NumPy linspace plotting it in Matplotlib

The NumPy linspace function returns evenly spaced numbers based on a specified interval.

The interval by default includes the starting value and ending value, but the ending value can be optionally excluded from the result.

To use the linspace function we will create a new script with the NumPy library imported as np.

Next we will call the linspace function using np.linspace( ) . Within the function, we will add two arguments.

The first argument is the starting value of the sequence and the second argument is the ending value of the sequence.

```
import numpy as np

#LinSpace(startpoint , endpoint)
numbers = np.linspace(0,10)
print(type(numbers))
print(numbers)
```

We will set the starting value to 0 and the ending value to 10.Remember, by default the linspace function will include the ending value, in this case 10, will be included in the resulting sequence.

```
<class 'numpy.ndarray'>
[ 0.         0.20408163  0.40816327  0.6122449   0.81632653  1.02040816
  1.2244898   1.42857143  1.63265306  1.83673469  2.04081633  2.24489796
  2.44897959  2.65306122  2.85714286  3.06122449  3.26530612  3.46938776
  3.67346939  3.87755102  4.08163265  4.28571429  4.48979592  4.69387755
  4.89795918  5.10204082  5.30612245  5.51020408  5.71428571  5.91836735
  6.12244898  6.32653061  6.53061224  6.73469388  6.93877551  7.14285714
  7.34693878  7.55102041  7.75510204  7.95918367  8.16326531  8.36734694
  8.57142857  8.7755102   8.97959184  9.18367347  9.3877551   9.59183673
  9.79591837 10.        ]
```

We will assign the result to a variable called numbers and then print the data type of thevariable to the console using the type function followed by printing the result numbers to the console.

The result is a NumPy array that contains 50 samples equally spaced from 0 to 50 where both the start and end values are included in the result.

The default number of samples in the result is set to 50, however we are able to change this value by adding another argument to the function.

Following the argument for the ending value, we will add the keyword num with the integer

25 to specify the number of samples to generate in the result. Setting the number of samples instead of setting a step size.

```
#LinSpace(startpoint , endpoint, numberofSamples)
numbers1 = np.linspace(start=0,stop=10,num=25)
print(type(numbers1))
print(numbers1)
```

```
<class 'numpy.ndarray'>
[ 0.          0.41666667  0.83333333  1.25        1.66666667  2.08333333
  2.5         2.91666667  3.33333333  3.75        4.16666667  4.58333333
  5.          5.41666667  5.83333333  6.25        6.66666667  7.08333333
  7.5         7.91666667  8.33333333  8.75        9.16666667  9.58333333
 10.        ]
```

Now the resulting array contains 25 samples ranging from 0 up to and including 10 that are equally spaced.

The next argument that we can pass to the function determines if the endpoint, in our case 10, should be included in the result.

The endpoint argument takes a boolean value, so passing the value False will exclude the ending value 10 from the resulting array.

```
#LinSpace(startpoint , endpoint, numberofSamples, endpoint(when False))
numbers2 = np.linspace(start=0,stop=10,num=25,endpoint=False)
print(type(numbers2))
print(numbers2)
```

```
<class 'numpy.ndarray'>
[0.  0.4 0.8 1.2 1.6 2.  2.4 2.8 3.2 3.6 4.  4.4 4.8 5.2 5.6 6.  6.4 6.8
 7.2 7.6 8.  8.4 8.8 9.2 9.6]
```

If we want to determine the step size between each sample in the array, we can add another

argument to the function called retstep which takes a boolean value. This argument is set to False by default, but if we specify the value True, the result will include the step size between each sample.

Note that the data type of the result is now a tuple which contains a Numpy array with the equally spaced samples and a float which is the step size between each sample.

```
#LinSpace(startpoint , endpoint, numberofSamples, endpoint(when False),retstep)
numbers3 = np.linspace(start=0,stop=10,num=25,endpoint=False,retstep=True)
print(type(numbers3))
print(numbers3)
```

```
<class 'tuple'>
(array([0. , 0.4, 0.8, 1.2, 1.6, 2. , 2.4, 2.8, 3.2, 3.6, 4. , 4.4, 4.8,
       5.2, 5.6, 6. , 6.4, 6.8, 7.2, 7.6, 8. , 8.4, 8.8, 9.2, 9.6]), 0.4)
>>>
```

Program:

LinespaceMatLab.py - C:/Users/Avinandan Bose/AppData/Local/Programs/Python/Python39/Machine Learning/LinespaceMatLab.py (3.9.6)

File Edit Format Run Options Window Help

```
import numpy as np

#LinSpace(startpoint , endpoint)
numbers = np.linspace(0,10)
print(type(numbers))
print(numbers)


#LinSpace(startpoint , endpoint, numberofSamples)
numbers1 = np.linspace(start=0,stop=10,num=25)
print(type(numbers1))
print(numbers1)

#LinSpace(startpoint , endpoint, numberofSamples, endpoint(when False))
numbers2 = np.linspace(start=0,stop=10,num=25,endpoint=False)
print(type(numbers2))
print(numbers2)

#LinSpace(startpoint , endpoint, numberofSamples, endpoint(when False),retstep)
numbers3 = np.linspace(start=0,stop=10,num=25,endpoint=False,retstep=True)
print(type(numbers3))
print(numbers3)
```

Output

```
<class 'numpy.ndarray'>
[ 0.         0.20408163  0.40816327  0.6122449   0.81632653  1.02040816
  1.2244898   1.42857143  1.63265306  1.83673469  2.04081633  2.24489796
  2.44897959  2.65306122  2.85714286  3.06122449  3.26530612  3.46938776
  3.67346939  3.87755102  4.08163265  4.28571429  4.48979592  4.69387755
  4.89795918  5.10204082  5.30612245  5.51020408  5.71428571  5.91836735
  6.12244898  6.32653061  6.53061224  6.73469388  6.93877551  7.14285714
  7.34693878  7.55102041  7.75510204  7.95918367  8.16326531  8.36734694
  8.57142857  8.7755102   8.97959184  9.18367347  9.3877551   9.59183673
  9.79591837 10.        ]
<class 'numpy.ndarray'>
[ 0.         0.41666667  0.83333333  1.25        1.66666667  2.08333333
  2.5         2.91666667  3.33333333  3.75        4.16666667  4.58333333
  5.          5.41666667  5.83333333  6.25        6.66666667  7.08333333
  7.5         7.91666667  8.33333333  8.75        9.16666667  9.58333333
 10.        ]
<class 'numpy.ndarray'>
[0.  0.4 0.8 1.2 1.6 2.  2.4 2.8 3.2 3.6 4.  4.4 4.8 5.2 5.6 6.  6.4 6.8
 7.2 7.6 8.  8.4 8.8 9.2 9.6]
<class 'tuple'>
(array([0. , 0.4, 0.8, 1.2, 1.6, 2. , 2.4, 2.8, 3.2, 3.6, 4. , 4.4, 4.8,
       5.2, 5.6, 6. , 6.4, 6.8, 7.2, 7.6, 8. , 8.4, 8.8, 9.2, 9.6]), 0.4)
```

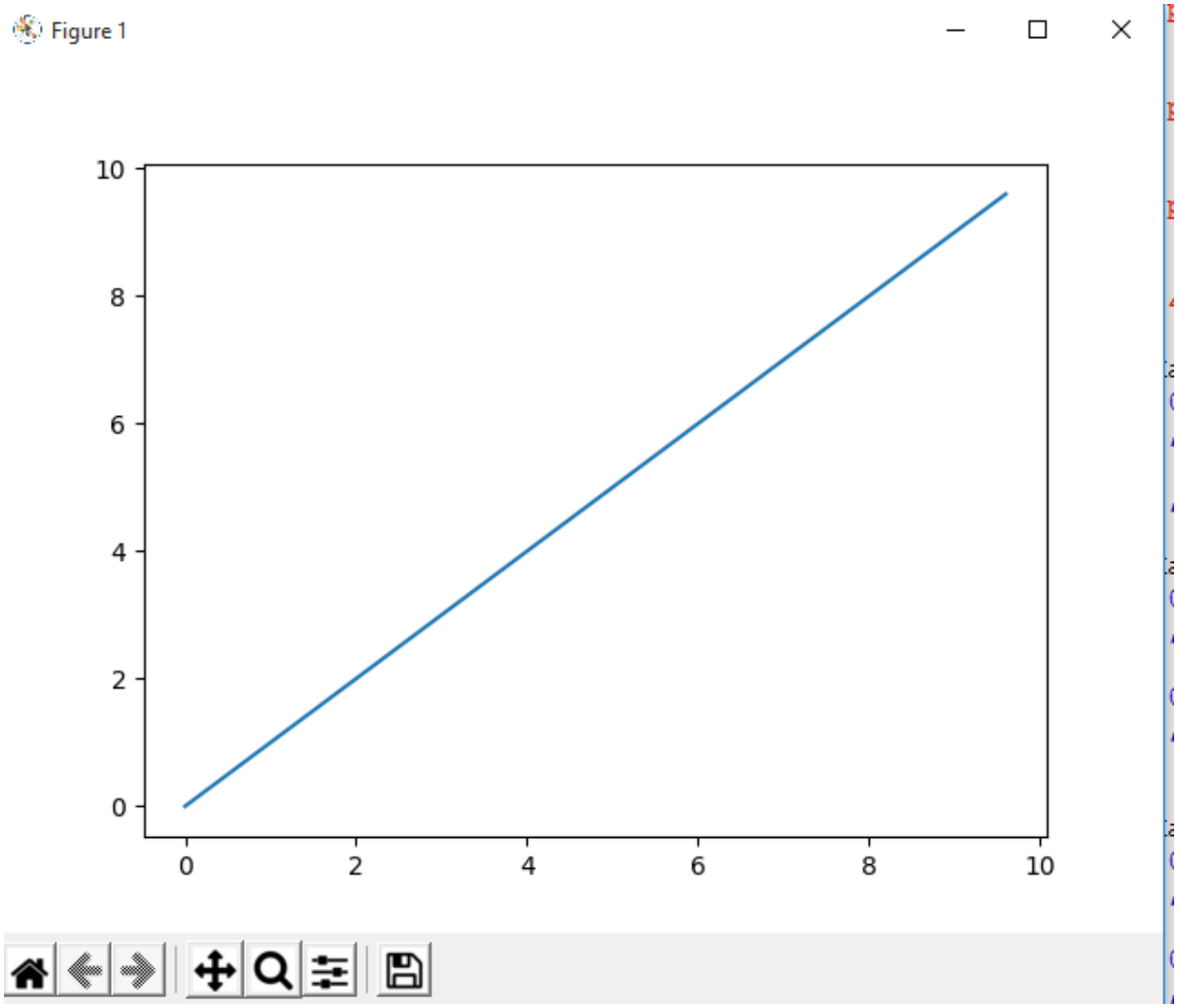
Plotting through Matplotlib tool

LinespaceMatLabPlt_1.py - C:/Users/Avinandan Bose/AppData/Local/Programs/Python/Python39/Machine Learning/Lines...

File Edit Format Run Options Window Help

```
import numpy as np
import matplotlib.pyplot as plt

N = 25
#LinSpace(startpoint , endpoint)
x = list(np.linspace(start=0,stop=10,num=25,endpoint=False))

y = list(np.linspace(start=0,stop=10,num=25,endpoint=False))

print("x:",x)
print("y:",y)

plt.plot(x,y)
plt.show()
```

Output



Note: Plotting will not occur if x and y dimension differs.