Python ile Fourier Dönüşümüne Giriş

1. Bölüm

Ön Bilgiler

Haluk Tanrıkulu

tanrikul@metu.edu.tr |MCNA Tech

Fourier Dönüşümün Temelleri

Öncelikle FT Python ile öğrenmek için ön hazırlık yapalım. Yani modülleri yükleyelim.
Bu bizim birinci adımımız olsun.

```
import numpy as np
import math
import matplotlib.pyplot as plt
import pylab as pl
from IPython import display
import time as ttime
import random
from mpl_toolkits.mplot3d import Axes3D
```

Anaconda ile bu modülleri yüklemiş olalım.

Öncelikle FT Python ile öğrenmek için ön hazırlık yapalım. Yani modülleri

Kompleks Sayılar (Complex Numbers)

Kompleks sayılar önemli önce ona bakalım.

```
# kompleks sayılar  real + imaginary şeklinde yazılır.
z = 4 + 3j
print(z)
# veya
z = 4 + 1j * 3
print(z)
```

```
# komplex (complex) fonksiyonunu kullanalım
z = complex(4,3)    # ya böyle
print(z)
z = complex('4+3j') # yada böyle
```

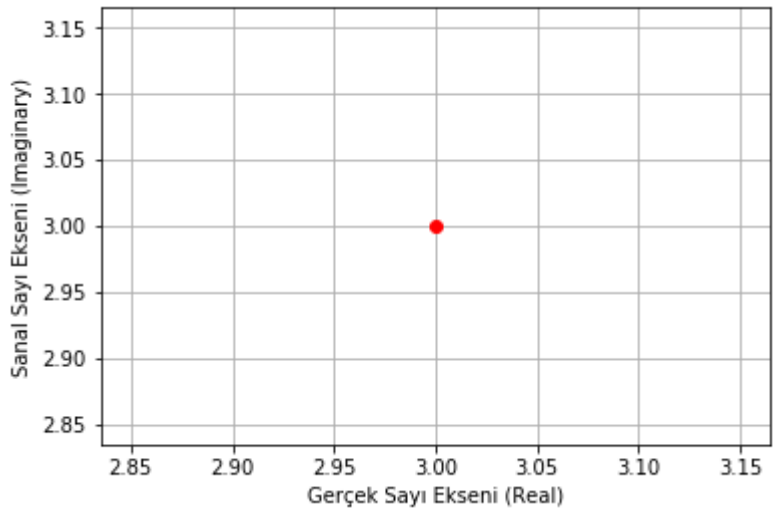
Çalıştıralım.

```
(4+3j)
(4+3j)
(4+3j)
(4+3j)
```

Çıkar.. nasıl yazarsak yazalım aynı sonuç çıktı...

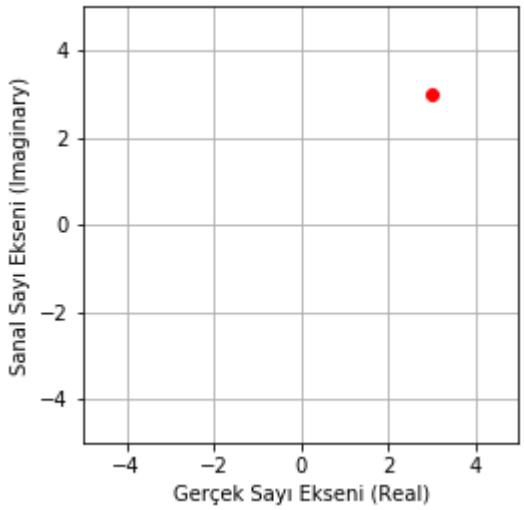
Komplex Sayının Gösterimi

```
# Komplex koordinat sisteminde bir komplex sayıyı gösterelim.

z =3 + 3j
plt.plot(np.real(z),np.imag(z),'ro')     # kırmızı o
plt.grid(True)
plt.xlabel('Gerçek Sayı Ekseni (Real)'), plt.ylabel('Sanal Sayı Ekseni (Imaginary)')
plt.show()
```




```
# çizimde bazı düzenlemeler yapalım.

z = 3 + 3j
plt.plot(np.real(z),np.imag(z),'ro')    # kırmızı o
```

```
plt.axis('square')
plt.axis([-5, 5, -5, 5])
plt.grid(True)
plt.xlabel('Gerçek Sayı Ekseni (Real)'), plt.ylabel('Sanal Sayı Ekseni (Imaginary)')
plt.show()
```



Magnitue (Mutlak Değerin Bulunması)

```
# Pisagor Teoreminden buluruz.
mag = np.sqrt( np.real(z)**2 + np.imag(z)**2 )
print( 'Magnitude Değeri : ',mag )

# yada doğrudan  abs() fonksiyonu kullanırız.
mag = np.abs(z)
print( 'Magnitude Değeri : ',mag )
```

Sonuç ne çıkar... Bakalım:

```
Magnitude Değeri :  4.242640687119285
Magnitude Değeri :  4.242640687119285
```


Açı Değerinin Bulunması

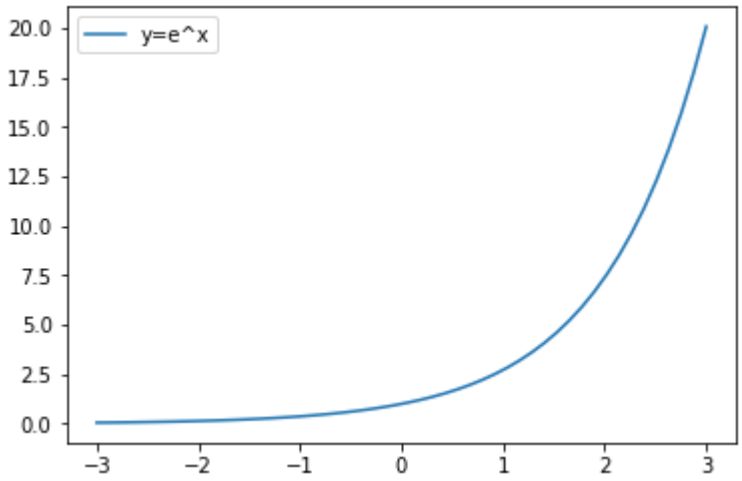
```
# arctanjant kullanalım.
mag = math.atan( np.imag(z) / np.real(z) )
print( 'Açı Değeri : ',phs )

# yada angle() fonksiyonu ile yapalım.
phs = np.angle(z)

print( 'Açı Değeri : ',phs )
```

Sonuç nedir?

```
Açı Değeri :  0.7853981633974483
Açı Değeri :  0.7853981633974483
```

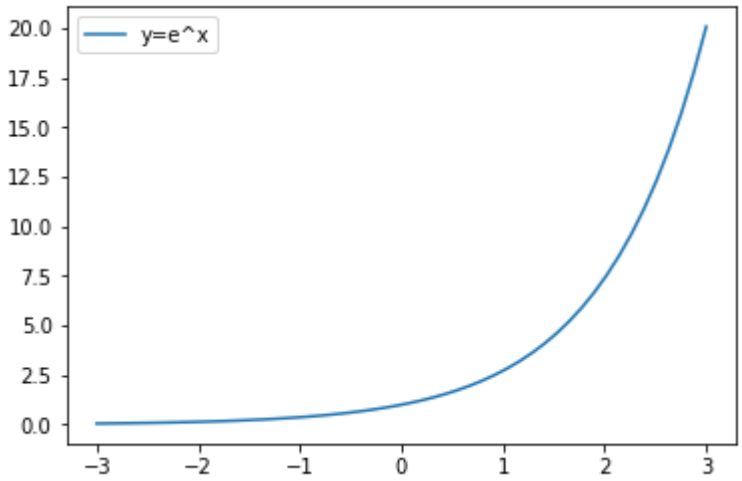
e ile Euler Formülüne Bakalım

```
x = np.linspace(-3,3,num=50)

plt.plot(x,np.exp(x),label='y=e^x')
plt.show()
```

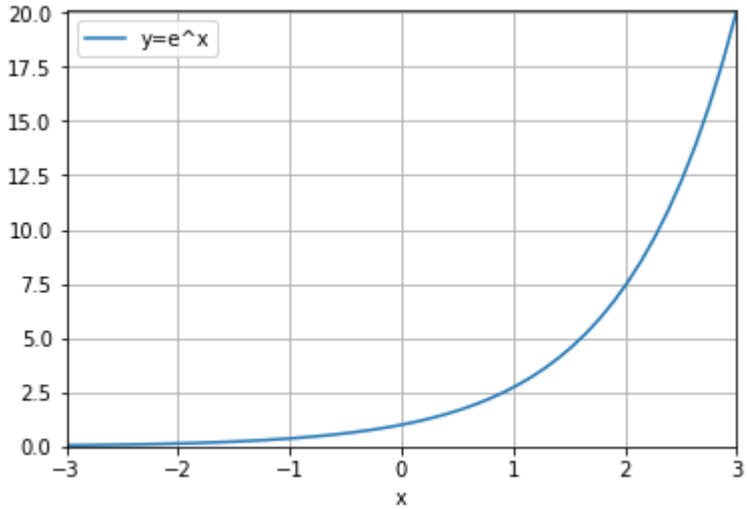


Bunu biraz değiştirelim. Bildiğimiz gibi yazalım.

```
plt.plot(x,np.e**(x),label='y=e^x')
plt.legend()
plt.show()
```

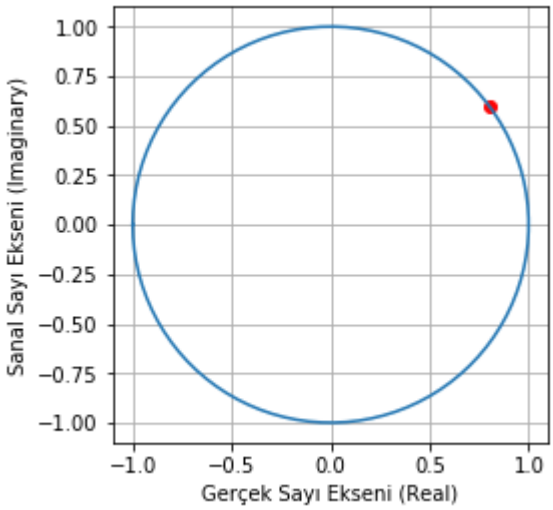



```
# Biraz düzeltme yapalım.
plt.plot(x,np.exp(x),label='y=e^x')
plt.axis([min(x),max(x),0,np.exp(x[-1])])
plt.grid(True)
plt.legend()
plt.xlabel('x')
plt.show()
```

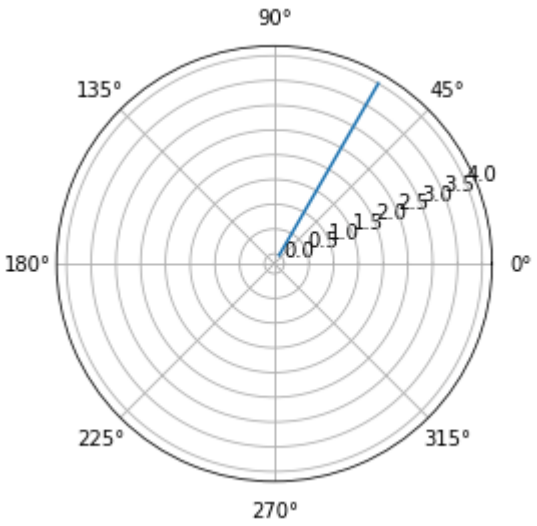


Şimdi Birim Çember üzerinde herhangi bir gerçel sayı olan k için [cos(k),sin(k)] değerlerini çizelim.

```
# Bir k değeri tanımlayalım.
k = 2/np.pi

# Euler gösterimi şu şekilde olsun.
euler = np.exp(1j*k)              #yani euler =  e^jk

# Bulunduğu noktayı çizelim.
plt.plot(np.cos(k),np.sin(k),'ro')

# referans birim çemberi çizelim.
x = np.linspace(-np.pi,np.pi,num=100)  #
plt.plot(np.cos(x),np.sin(x))

# biraz güzelleştirelim
plt.axis('square')
plt.grid(True)
plt.xlabel('Gerçek Sayı Ekseni (Real)'), plt.ylabel('Sanal Sayı Ekseni (Imaginary)')
plt.show()
```




```
# Euler formülü keyfi vektör büyüklüğü ile


m = 4                # genlik
k = np.pi/3          # açı
compnum = m*np.exp( 1j*k )

# extract magnitude and angle
mag = np.abs(compnum)
phs = np.angle(compnum)

plt.polar([phs,phs],[0,mag])
plt.show()
```



Sinüs dalgası ve kompleks sinüs dalgası

```
# Benzetim parametreleri
srate = 500;                          # örnekleme frekansı (Hz)
time  = np.arange(0.,2.,1./srate)  # zaman (sn)

# Sinüs dalgası parametreleri
```

```
for i in range(1, 4):

    freq = i                    # frekans in Hz
    ampl = 2                    # genlik  a.u.
    phas = np.pi/3              # faz radyan
freq = 3
# Sinüs dalgası oluşturalım
sinewave = ampl * np.sin( 2*np.pi * freq * time + phas )

plt.plot(time,sinewave,'k')
plt.xlabel('Time (sec.)')
plt.ylabel('Amplitude (a.u.)')
plt.show()
```