

part 9

numpy - 라이브러리 .import해서 불러와야함.

중요한 함수들은 그 함수들이 모여있는 라이브러리를 불러와야함.

함수가 들어있는 어떤 라이브러리를 . 패키지.

numpy 라는 패키지 라이브러리 속에 또 작은게 들어있을수 있음, 그 밑에 패키지 이름 A, B 해서 만들고 또 그안에 만들수도.

(점을 함수라고 생각)

numpy 안에 내부적으로 함수가 존재할수 있음. A안에 패키지가 또있다면. 패키지. 내부적으로 안에 있는걸 부를때는 “ numby.A.D.어떤함수” . numpy가 젤 위에 있는 상위개념이고 그 안에 또 패키지... 점이 그런 용도로 쓰임.

1) import numpy (np안에 있는 건 쓸수있음)

numpy-A-D-f를 쓰고 싶으면 import np를 했으니까

numpy.A.D.f라고 쓰면됨> (너무 복잡)

>불러올 때 젤 큰 깍데기를 불러오는 방법.

2) from Numpy Import A (> numpy 에 있는 A를 불러오자) (from을 쓸수도)

A.D.f (그러면 A를 그때부터 쓸수있음-)

from Numpy import A.D (이렇게도 가능)

>from 으로 import할 수도 있음

사진

In [2]

np라는 젤 큰거를 불러옴.

import해서 들어오고 numpy를 줄여서 쓰고 싶을 때 as.

np 라고 줄여서 쓰고 싶을 때 as

mat를 불러옴.

matplotlib가 젤 위에 있고(깍데기) 그안에 포함되있는게 - pyplot가 속해있음. 포함되어 있는 subset을 plt로 받아온다.

or

쓸 수 있는 다른 방법중 틀린 것-시험

from mat import pyplot as plt

-그 다음부터 plt쓸수 있음

(점은 포함관계로 되었다 라이브러리에)

numpy는 왜 필요? list 하고 아주 비슷한데 쓰는 이유는 수학적으로 계산도 할 수 있고 해서 처리할 모든 data는 다 np 처리를 해야

[3]

np.empty

empty는 함수 (괄호로 input을 받음)

np라는 젤 큰 라이브러리 안에 empty 라는 함수.-(그 아래에 subset 은 포함되지 않고)

list가 입력으로 들어감.

2 by 3 로 리스트로 .

가로 세로 행렬 : 직사각형의 숫자의 array

2행 3열 - 뽕뽕 옆으로

내부적으로 들어가는 것은 datatype을 .dtype을 int로 해서 하나 만들어라

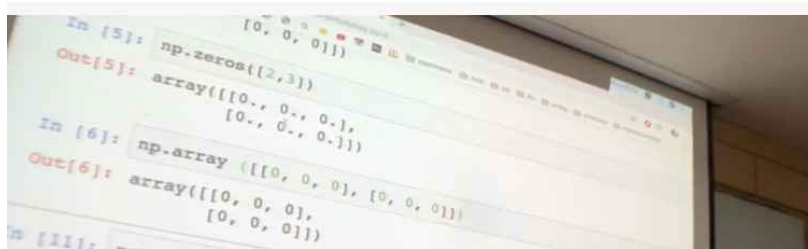
empty 라는 함수는 빈. (비었따 하지만 빈 건 아님)

out[2]

numpy속에 들어있는 함수를 이용했기 때문에 숫자가 만들어지는데 계산가능한 숫자

2by 3의 array로 되었음. - 3개짜리 list가 하나가 있고 그게 두줄이 있음.

int : dtype을 int로 해서. 안에 든 숫자는 랜덤한 숫자. 소수점 숫자 아님.



[4]

np라이브러리 속에 있는 zeros라는 함수를 이용. 거기 입력에 list로 2곱하기 3을 넣음.

이 함수는 2by 3의 행렬을 만드는데 0으로 채워진.

** [0,0,0]

>>그냥 0이 들어가있는 백타

이걸 2by 3로 만들고 싶을때는

** [[0,0,0] , [0,0,0]]

2행이 있고 3열이있는 리스트.

>>쓸수가 없음 계산안됨.

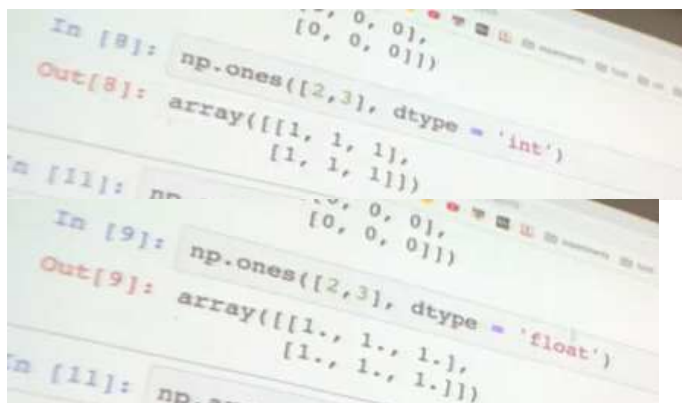
계산이 되는 array로 만드는 방법이

np.array ([[0,0,0] , [0,0,0]])

리스트에서 array로 바꿔줌

[4]와 같음. zeros함수와 같음.

np.array - list를 array 로 convert해줘라 >위에거랑 똑같은 결과



zeros대신 ones하면 1이 만들어짐. 1 다음에 점 점이 붙은건 int가 아니라 float. (dtype = int 안했을 때)

ones라는 함수가 디폴트로 datatype을 float로.

dtype= 'int' 적어주면 점이 사라짐

float64 로해도 변화는 없음. 소수점 몇째자리 까지 할까에 대한 정의. precision을 얼마나 밑에까지 할까에 대한 정의. - 오차를 허용하고 싶지 않을 때
 64- 정교하고 싶을 때 - 숫자를 길게 하면 메모리를 많이 차지함. 1.00000...
 정확도와 데이터 양 반비례.

```
In [5]: np.arange(5)
Out[5]: array([0, 1, 2, 3, 4])

In [6]: np.arange(0,10)
Out[6]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

In [7]: np.arange(0,10,2)
Out[7]: array([0, 2, 4, 6, 8])

In [8]: np.arange(0,10,2, dtype='float')
Out[8]: array([0., 2., 4., 6., 8.])
```

np에 있는 arange라는 함수는 계산이 될 수 있는 array를 만들어줌. 숫자가 다섯 개. 5라고 쓰는순간 index 5개가 만들어짐.(0부터)

arange (0,10) : 0부터 10까지 하면 10은 포함안되고 10 밑으로까지.

(0,10,2) : 2만큼 증가increment하면서 10미만까지 increment- 증가분

data type설정도 할수있음- float, float32, float64도 쓸수있음. - 얼마나 정확성을 높이느냐와 관련.

```
In [9]: np.linspace(0,10,5)
Out[9]: array([ 0.,  2.,  4.,  6.,  8., 10.])

In [10]: np.linspace(0,10,7)
Out[10]: array([ 0.,  1.66666667,  3.33333333,  5.,  6.66666667,  8.33333333, 10.])

In [15]: X= np.array([4,5,6])
X
Out[15]: array([4, 5, 6])

In [14]: X= np.array([[1,2,3],[4,5,6]])
X
Out[14]: array([[1, 2, 3],
               [4, 5, 6]])

In [17]: X= np.array([[1,2],[3,4],[5,6]])
X
Out[17]: array([[1, 2],
               [3, 4],
               [5, 6]])

In [18]: X= np.array([[[1,2],[3,4],[5,6]], [[1,2],[3,4],[5,6]]])
X
Out[18]: array([[[1, 2],
               [3, 4],
               [5, 6]],
               [[1, 2],
               [3, 4],
               [5, 6]]])

In [20]: X. ndim
Out[20]: 3
```

X. ndim - 3차원.

[9]

linspace - linear의 준말

linspace(0,10,6) : 0부터 10까지 0, 10포함(arange와 달리). 그걸 총 6개로 똑같이 나누어 준다.

linear space - 차이가 다 똑같음. 첫째에서 둘째로 가는

[11]

np. array : list에서 array 로 바꾸는거

백터를 만들고 싶을 때,

2by3를 만들고 싶을 때.(행렬을 만들고 싶을 때)

3by 2 만들고 싶을 때.

*백터면 1차원. 직사각형 2차원. 직육면체로 되면 3차원. 2차원 행렬 두 개 있으면 3차원

대괄호 두 개- 2차원. 대괄호 하나 더 쓰고 쉼표하고 같은거 쓰고 대괄호닫기. >3차원 -대괄호 3개.

차원

```
In [21]: X.shape
```

```
Out [21]: (2, 3, 2)
```

```
In [22]: X.dtype
```

```
Out [22]: dtype('int32')
```

```
In [23]: X.astype(np.float64)
```

```
Out [23]: array([[[1., 2.],
                  [3., 4.],
                  [5., 6.]],
                 [[1., 2.],
                  [3., 4.],
                  [5., 6.]])
```

```
In [25]: np.zeros_like(X)
```

```
Out [25]: array([[[0, 0],
                  [0, 0],
                  [0, 0]],
                 [[0, 0],
                  [0, 0],
                  [0, 0]]])
```

```
In [24]: X*0
```

```
Out [24]: array([[[0, 0],
                  [0, 0],
                  [0, 0]],
                 [[0, 0],
                  [0, 0],
                  [0, 0]]])
```

x. shape

(2,3,2)

2곱하기 3곱하기 2 의 차원이다. (3개니까 3차원인데

-3개니까 3차원인데 3곱하기 2가 2차원짜리 직사각형. 이게 두 개가 있다. 쉼표 큰 괄호속에있는게 두 개. (가 첫 번째 나오는 2) 그다음괄호속에 있는 3개의 리스트 안에 2개의 숫자.

X. dtype

: x속에 나오는 숫자가 어떤 type인가 >int /32 precision과 관련

타입을 바꾸고 싶을 때 astype 이라는 function .

-np.float64로 바꿔줘라

np. zeros_like / x를 만들었는데

모든 형태를 유지한채 숫자를 0으로 바꿔주라. : zeros like

X*0 해도/ 이제 계산이 가능하니 똑같이 만들어짐.

```
#data = np. random ~~
```

np 라는 큰 라이브러리 속에 random 이라는 sub패키지 속에 있는 normal이라는 함수를 쓴다.

from np import random. normal 도 가능

normal함수 - normal distribution을 만들어주는. 정규분포. 종모양 데이터를 만들어주는. 종 모양을 만들기위해 필요한게 두가지 정보 - mean값(평균) 과 얼마나 뚱뚱한지./ 0 - mean 1- 뚱뚱 100개의 데이터 / random한 데이터. 값(나올때마다 다름) / 총 백개의 데이터가 나옴.

data. ndim 은 1차원. data의 디멘션은 일차원- 대괄호 하나

data. shape- 총 100개의 숫자가 나옴.

100개의 데이터를 가지고 정규분포를 가지는지 플레팅을 할 수 있음.

mat라는 라이브러리 속에있는 plt라는 서브패키지를 import해올 것. plt - subpackage 속 세 속에 hist라는 함수를 씀.- 벡터 array 계산할수있는 data를 입력으로 하고

히스토그램: 늘 옵션에 bins 바구니를 총몇개로 할건가 정함. 바구니를 10개로 만들어 0기준 한쪽으론 -. 한쪽으론 + 값. 값들이 점점 쌓이는.

x축에다가 데이터값을 공으로 생각하고 던지면 바구니 속에 쑥 들어감. 위에 공들이 쌓임.

그렇게 그래프 그리는게 히스토그램.

그림 - 바구니 총 10개. range 속에 들어가는 값들 . y값에 해당되는 건 무조건 0을 포함한 정수값 .값들을 다 합하면 100개- 시험

<11/5>

Phasor

1초동안 얼마나 숫자가 뻑뻑하게 값들을 담을건가 - sampling rate

SR을 만으로 한다고 규정하면 1초동안 총 만개의 숫자를 담을거다.

1초에 얼마나 할 때 hz라는 단위.

list상에서는 어떠한 계산도 안되기 때문에 numpy 에 담는다.

```
np.array [ ]
```

다차원의 array를 만들 수 있다. 1차원이면 벡터 2차원이면 직사각형 3차원이면 volume

numpy - list에 숫자 담는것보다

np. array [] 하면 np형태의 데이터로 바뀐다.

Sound

다양한 pure tone의 합으로 복잡한 사인.

sinusoidal-사인 코사인처럼 곡선으로 생긴 phasor- sinusoidal function을 만들어내는 것 싸인하고 코사인에 들어가는 입력 - sin(입력)입력값은 degrees 가 들어가면 안되고 radians 가 들어가야함.

0부터 100파이 까지 사인 코사인 그래프를 그리면 총 몇 번의 반복? - 50번. 2파이가 반복되

니까 .

파이는 무리수

$\sin(\pi/4)$

오일러 공식

e도 무리수

\sin - 입력변하면 출력을 해주는 함수

$\sin \cos$ 처럼 e 저거도 함수 - e, I 는 fix 되었음. radian값이 변함으로써 어떤 값이 뺄어지는.

세타만 변하면 어떤 값이 나옴 e, I는 숫자. 세타에 파이 넣으면 다 숫자니까 숫자값이 나옴.

오일러 평선의 값들은 어떻게 그럴까.

복소수 평면

각도값을 그려서 만들고 몇 콤마 몇 하면 $0.3 + 0.8 I$ 이렇게 나옴

벡터의 정의는 숫자열 . 벡터값으로 표현이 됨.

t세타가 바뀔때 따라 원을 따라 .. 0 ~파이 ~ 2파이 ~ . 계속 도는.

projection : x축 a에 project를 하면 위에서 보는. > ---- 가로로 왔다갔다

y축에 project를 해서 b 보면 세로로 왔다갔다.

실수의 관점에서만 보겠다 하면 위에서 보면 됨. 1에서 시작해서 왔다갔다

허수만 보겠다 하면 0에서부터 위아래로 왔다갔다

실수 - \cos 허수 - \sin

\cos 그래프는 1부터 시작 \sin 은 0부터 시작

실수 - 1부터 시작 . 허수 - 0부터 시작해서 위 > 아래 0부터 올라갔다가 내려가는.

인풋이 세타 레디언 각도값.

pure tone에 넣는 입력값중에 frequency (1초에 몇 번 왔다갔다 하는가)

사인 세타를 쓰니까 각도값이 변한다고 할 때 시간의 개념이 안들어감 -각도값이기 때문에.

시간의 개념이 들어가야 1초에 몇 번 왔다갔다느냐해서 소리의 높이.

각도개념, 초개념을 같이 넣어줘야 진정한 소리가 나옴. 소리는 반드시 시간의 개념이 들어있어야.

sampling rate : 1초에 총 만개의 숫자로 표현.

#

<11/7>

크게 matplotlib 가 있고

import matplotlib.pyplot

해도 똑같은거.

젤큰 라이브러리 밑에 sub라이브러리.

#parameter setting

time 은

0부터 2 파이까지 만들어짐.

part 11

numpy 라이브러리를 import.

첫 번째 줄 . 플래팅 할 때 쓰는 라이브러릴. 큰 라이브러리 이름은 matplotlib . 큰 라이브러리 밑에 sub라이브러리. - 이것과 똑같은 역할) import matplotlib. pyplot
from을 쓰며 그 속에 pyplot을 부름.

Phasor에서 변수들을 parameter setting

변수에다 값을 담아 놓음.

amp 나 sr dur을 바꿀 때 이것만 고치면 결과값이 쉽게 바뀜 .이런게 parameter
time을 왜 만들까 - 사인 코사인 오일러 평선 등은 입력값이 각도값임 (degree 안돼
radian만) . time이 필요한이유는 각도값 만으로는 실체의 소리를 만들수없음.
사인 코사인으로 플래팅을 했을 때 어떻게 플랫 되는지 해볼거임.

sr은 1초에 몇 개가 들어나는지 - 1초라는 말이 들어가면 time과 관련있는거.

2차원 - 머 콤마 머 해서 2차원의 숫자가 들어가는 숫자가 두 개있는 벡터.

part 12

amp sr (얼마나 정보를 촘촘하게 할건가/숫자들이 1초동안 얼마나 나오는가) freq (얼마나
나 1초동안 왔다갔다 할까/shape 반복이 얼마나 왔다갔다 하는가)

어떻게 amp를 장착시킬까 - 증폭. / complex wave 소용돌이 치듯이 생김. 그게 얼마나 커
지는가.

그림

sr=10hz. 주어진 숫자가 10개. 이걸로 100번 왔다갔다 하는 걸 표현 할수 있을까
아무리 동그라미를 아껴도 5번의 왔다갔다 까지밖에 안됨.

주어진 숫자에 개수의 표현할 수 있는 주파수는 반밖에 안됨 멕시멈이.

1초에 웨이브를 두 개 만드는건 쉬움.

숫자가 주어져 있으면 무한대로 표현할 수 없음.

sr의 반에 해당되는것만 표현가능. Fre = 5hz 가 Maximum.

CD가 담는 음질은 sr = 44100Hz 1초에 주어진 숫자가 44100. 사람이 들을 수 있는 가청 주
파수가 2만. CD 음질이 저기에 고정.

NF 는 22050 hz. - 피치로 쳤을 때 아주 높은 소리. /사람이 그 위로는 들을 수 없음.

+유선전화도 소리가 디지털로 선을 따라감. 숫자값들이 움직임. 얼마나 뻑뻑하게 전송을 해야
하는지가 중요. sr을 얼마까지 해야하는가가 중요. / 유선전화의 sr은 8000hz. 4000까지 표
현가능.- 낮음. / 들을 수 있는 최대. /사람이 헛갈릴대 많음 엄만지 딸인지 말소리는 다 들
리지만.

휴대폰은 16000hz. NF 8000 까지 .

sr의 half 에 해당되는 NF까지가 숫자상으로 표현할 수 있는 fre의 Max. - CD음질의 sr를

가지고 박쥐 소리를 녹음하면 저기 들어가지 않음. 고주파가 담기지 않음.
초음파는 fre에 해당하는 애기 - 2만보다 훨씬 높은 소리.

pulse train :
그래프는 다 더했을 때 나오는 모양.

****frequency, sine wave, sr의 개념을 다시 보기 + fre 랑 sine wave의 관계.**

part 13
fre를 440hz로.
시간. 0.5초까지 만들었는데 0.01초까지 display.
parameter setting을 바꿔서 440hz(라 소리)를 880hz로 바꾸면 아까 소리와 높이가 같음.
- 1760hz로 하면 . 다 '라' 가 됨. -220,110 도 라. 배수로 하면 다 같은 음으로.

sin 대신 cos 을 쓰면 시작점이 달라짐. - 1부터 . 1초간 몇 번 왔다갔다 하는지는 같음
fre는 고정이기 때문에. / 두 소리는 둘다 '라' . 다르게 들리지 않음. / sin cos는 shape은 같지만 sin는 살짝 이동한거 $-\pi/2$ 90도 차이가 있음. / cos에서 90도 옆으로 이동하면 sin.
 $\pi/8$ 를 이동하면 소리가 달라질까 ? 노. 얼마큼 이동하든 소리는 다 같음.
phase 에 대한 거는 인식을 못한다. frequency 변화는 느끼지만 phase shift는 아님.

complex phasor 실행 -
complex number자체는 플래팅이 안됨. a ,b를 각각해서 2차원으로 플래팅.
>라 에 해당하는 소리가 나옴.
real = cos

Pulse train
F0를 하나 정하고. 오천까지(NF)
spectrum - 한 타임 포인트에서의 어떤 주파수 성분이 많은지 보여주는거. 낮은 주파수부터
있는 그림.
decreasing 하게 만들어야.
pulse train 만든걸 prat.

<11/19>

데이터 기계(인공지능,함수) 데이터
함수
데이터 부분은 벡터-숫자열로 되었음. 숫자열이 앞뒤 같을 필요는 없음.
음성 text
>음성이 들어가서 text 형태로 나오는 >>음성인식
text 음성 - 음성합성
일본어text 한국어text - 기계 번역

-알파고 같은 경우 바둑상태가 입력으로 들어가서 몇 번째 수를 뽑아야 하는가가 나옴.

중간에 있는게 인공지능, 함수/어떤 형태를 지닐까 -행렬의 형태

입력벡터를 출력벡터로 만들어주는 함수의 역할. -인공지능

음성 텍스트 사람이름,, 뭐든 될수있음 벡터는

모든 데이터가 벡터로 되어야하는건 행렬 곱을 하기 위해서.

행렬의 형태를 거치면 다른형태가 될수있음- 이미지가 들어가면 누구의 얼굴

5 3 0 1 -1 0 2 -3 6

0 1 3

3 -5 7

2 3 4

4 x 1(세로가 몇 개있냐와 관련) 4x 3 가로가 몇 개 있냐와 관련.

젤 왼쪽 컬럼이 왼쪽에 들어가 곱하고 더해서 오른쪽벡터가 됨.

중간계 행렬함수. 왼쪽이 입력(음성) 이라 생각하고 함수를 곱하기해서 통과하고 , 행렬과 행렬의 곱. 입력이 함수를 통해서 출력이 됨.

기계가 많은 데이터로부터 학습을 해서 얻어내는 것이라고 생각 - 인공지능.

인공지능- 행렬의 곱. 입력벡터를

Linear algebra 선형대수 - 행렬

모든 인공지능이 선형대수.

<11/21>

A x = b

1 3

5 1 6 2 0 15 29 3

6 -1 3 9 1 33 19 1

33 3 -1

입력

출력

3 by 2. 2 by 3__ 3 by 3

-바깥에 있는게 결과값.

A =

[1 2

-1 0

3 5]

3 by 2

col - ws = 3차원

spanning 시키면 2차원의 plane이 나옴. = column space - cv를 표현하고 원점과 연결시키면

cs(2차원) 와 ws(3차원) 는 안똑같아. - 한차원 남는게 left null space

LC으로 표현가능한 모든걸 spanning . 벡터 두 개를 LC 하면 B가 나오는데 B가 될 수 있는
거, all possible Bs 가 column space. -

row space

ws 는 2차원. spanning해내는 공간은 2차원을 넘어가지 않음.

rv 가

column I 2 개 - cs와 같음. = rank

column 으로 하든 row로 하든 I 한 벡터들의 숫자는 늘 같음 - rank

3 x 2 (ws) df

2 = 2 = cs . I 한것의 개수

1 0 > null space

$x\mathbf{A} = \mathbf{0} \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$

A라는 matrix 가 있는데 뭐를 곱해도 0이 되는 것.

x에는 1by 3가 와야됨. (3차원)