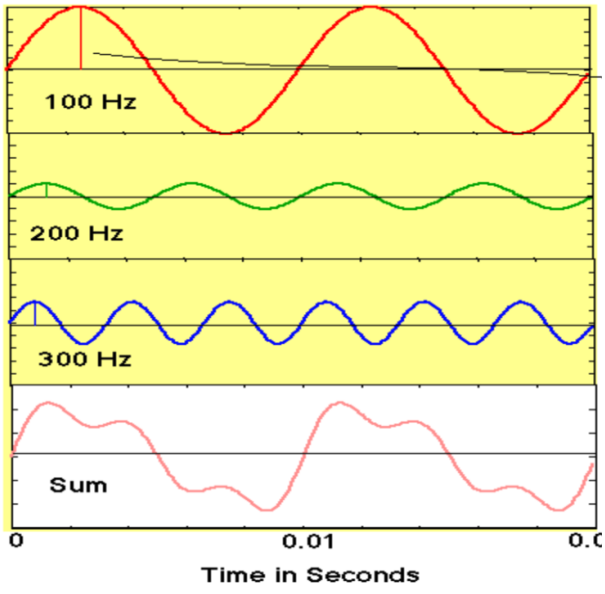
1. 소리는 숫자임. 등간격에 따라 할당된 숫자로 소리를 표현함. 그 점을 이어 그래프로 표현한 것. >>> 첫 수업, Praat을 이용하여 '안녕하세요'를 녹음한 것을 통해 알 수 있음
2. Consonants (자음)
    A. 소리랑 철자랑은 다름!
    B. 유의해야 할 표기는 j =/= 져 / j = 여
3. Vowels (모음)
    A. monophthong (단모음) / diphthong (이중모음): 6개가 있음
    B. 모든 모음은 voiced임
4. Phonetics
    A. A study on speech: speech는 사람과 관련된 발화임. 동물 소리에는 speech라 하지 않음.
    B. Articulatory (mouth), 조음 / Acoustic (through air): 물리적 영역, 음향 / Auditory (ear), 청각
5. Articulation
    A. Vocal tract: Nasal tract / oral tract
    B. Upper VT: lip ~ teeth ~ alveolar ridge ~ hard palate(구개) ~ soft palate(velum) ~ uvula ~ pharynx ~ larynx
    C. Lower VT: tongue structure (tip, blade 등) ~ epi(덮개)glottis
    D. Ex. 코로 숨 쉴 때 velum은 lower → '아' 소리 내면 올라감
    E. 사람은 기도(식도x)를 이용해서 소리가 남.
    F. Larynx: Voice box. Voiced/Voiceless: vibration 차이
    G. lips / tongue tip / tongue body: 메이저 조음에 관련된 것
    H. Constriction Location (where) / Constriction Degree (how much)
    I. CD: Upper part ~ Lower part: Stops > fricatives > approximants > vowels
    J. English 발음은 Constrictors, CD, CL에 따라 달라짐
        i. Ex. Velum raised, glottis opened, tongue tip, alveolar, stop = [t]
    K. cf. 모든 모음은 constrictor로 tongue body만 씀
6. Phonemes
        1. individual sounds that form words ex. /s aɪ k oʊ/
7. Praat
    A. Duration(sec), Intensity(dB), Pitch(Hz), Formant(Hz)
    B. 모든 시그널(신호)은 다르게 생긴 여러 사인 웨이브의 합으로 표현됨
    C. 이 세상에 존재하는 모든 신호는 sine wave = f(frequency, magnitude)
    D. Magnitude=amplitude



E. 사인웨이브(SW) 1은 크기는 크지만 freq. 작음 → slow. 1초에 저 모양이 100번 들어감

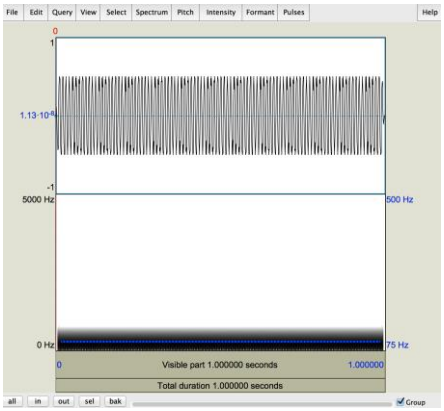
F. SW2는 2배 빠름.

G. Sum SW도 1초에 100번 반복됨.

H. x축은 시간, y축은 voltage(=value값)

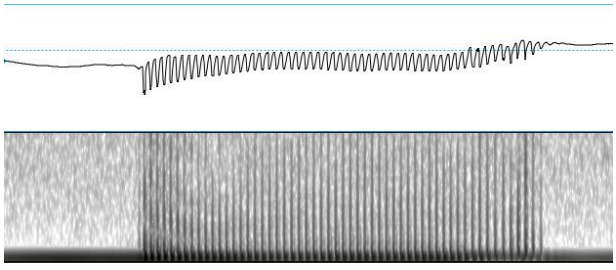
I. 합쳐서 Sum SW로 만드는 것 = synthesis / 그걸 어떤 것들로 이루어졌는데 스펙트럼으로 표현하는 것= analysis

J. 막대그래프로 만들면 스펙트럼: x축은 frequency, y축은 amplitude



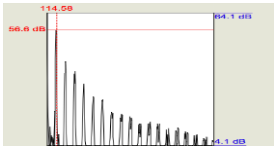
K. 검게 칠해진 것: spectrogram. 스펙트럼을 time으로 visualize한 것임. 스펙트럼 자체는 시간 개념이 없기 때문.

L. Human voice는 여러 사인 웨이브가 합해져 있음



M. Source의 filter를 어떻게 만드느냐에 따라 '아' 소리도 만들고 '이' 소리도 만듦.

N. Source만 가지고도 spectral analysis 가능
O. F0 = fundamental frequency
P. 스펙트럼의 x축: time, y축: frequency
Q. 진한 것이 amp가 큰 것임
R. 이건 gradually decrease harmonics. 저주파에서만 에너지가 강함.



S. Human voice source consists of harmonics!
T. Vocal tract의 shape에 의해서 소리가 만들어지는데 이걸 filtered된다고 말함
U. Peak / Mountain
V. 어디에 산맥이 나타나는가는 입모양에 따라 달라짐. 특정 높이의 소리는 특정한 입모양과 맞춰질 수밖에 없음. 그치만 같이 '아' 소리 내는 김씨, 이씨, 박씨의 산맥 패턴은 같음
W. 첫 번째 산맥 = f1 (=/= f0)
X. Praat을 통해 synthesize, combine source하는 법 배움

<Starting Python from Oct. 01st, 2019>

A. Variable
   A. 모든 language는 '단어'와 '문법'(어떻게 combine 하느냐)이 있음. '단어'는 그릇. '사과'라는 정보를 담으면 >> 사과가 됨. -→ 컴퓨터 언어에서 이것은 Variable(변수). 숫자, 문자 등을 담을 수 있음.
   B. 문법: 1) 변수라는 그릇에 어떤 정보를 assign 하는 것 2) conditioning 하는 것=if절 3) 여러 번 반복하게 하는 것=for절 (loop) 4) 함수 (어떤 입력을 넣으면 어떤 출력이 나오는 것).
   C. Directory 가려면 anaconda prompt >> jupyter notebook >> 원하는 directory를 browse >> New >> Python3
   D. = 표시는 오른쪽에 있는 정보를 왼쪽에 있는 variable로 assign한다는 뜻
   E. Print 함수는 ( )에 변수를 넣으면 그 정보를 print out 하는 것 ex. Print(a) 에서 입력은 a
   F. Variable은 unique 하기 때문에 처음에 a=1하고 다음 셀에 a=2 쓰면 a=2로 overwrite됨. 근데 a=1셀을 클릭하고 run하면 그 아래 a=2 셀 있어도 print(a)=1 나옴
   G. 문자는 그냥 쓰면 무조건 변수취급임. 그래서 정보값으로 하고 싶으면 ' ' 해야 함
   H. Shift+enter = 실행
   I. Love = 2 / b = love / print(b) = 2 : love라는 변수에 2값을 넣고, 다시 b라는 변수에 love가 가진 정보값을 도출하게 b=love라는 식을 세우면, print(b)하면 love의 정보값인 2가 나옴. 그리고 엔터 대신에 a=1; b=2; c=3도 가능
   J. 한 셀에 a=1 엔터 b=2 엔터 c=3 하고 같은 셀에 엔터 c 이렇게 쓰면 마지막 c는 print(c)랑 같은 뜻임
   K. [ ] 이거는 리스트를 표현하는 것

L. Type( )하면 각 변수가 무슨 타입인지 알려줌 ex. a=1 → type(a): int / a=1.2 → type(a)=float 실수 / a=[1,2,3] → type(a)=list / a='love' → type(a)=str 스트링 / a=[1,2,3,'love'] → type(a)=list 즉, 리스트는 숫자만일 필요는 없음 / [ ] 이거를 ( ) 얘로 바꿔도 됨. Type은 tuple임. Tuple과 list은 완전 똑같은데, 다만 보안에 더 강함.

M. Dictionary ex. a = ('a': 'apple', 'b': 'banana') 여기엔 2개가 들어있는 것, 표제어 : 설명 쌍으로 되어있음. → type(a) = dict

N. Variable 이름 적고 대괄호 쓰고 안에 index 쓰면 그 값을 가져옴. a=[1,2,3]. Print(a[1])=2

O. Dict의 정보를 access 할 때는, 페어에서 앞부분을 인덱스로 씀. 그니까 0,1,2 안 씀. Ex. a={"a":"apple", "b"="orange"}; print(a["a"]) = apple

P. S='abcdef' ; print(s[1:3]) = bc(1번부터 3번 직전까지)

Q. #을 붙이면 그 뒤가 실행이 안 됨. 쓰고 싶은 노트 쓰면 됨. / Code대신에 markdown으로 바꾸면 노트로 쓸 수 있음

<10.29 시험 리뷰>

1. 영어의 nasal consonant는 발화할 때 velum이 lowered 되어서 nasal tract으로 air flow가 있지만, oral tract는 막혀서 air flow가 없다. (True)
2. 일반적으로 영어 모음 /a/를 발화할 때, nasal tract의 air flow는 차단된다. (T)
3. 다음 중 발화시 air pressure와 가장 관계가 있는 것은? (Intensity)
4. 영어의 자음 중 코를 막고 숨을 쉴 때와 가장 유사한 articulation 상태인 음소는? (h)
5. 영어 /h/는 다음 중 어떤 articulator에서 constriction을 가지는가? (lips/tongue tip/tongue body/**none**)
6. 영어 음소 중에서 tongue body애서 constriction을 가지며 constriction degree는 fricative, constriction location은 velar인 자음의 개수는? (0개)
7. 영어 자음 /v/의 articulator의 constriction location을 bilabial로 바꾸고 constriction degree로 approximant로 바꾸면 어떤 음소가 되는지 쓰시오. (w)
8. /s/, /l/은 major articulators (lips, tongue tips, tongue body) 중에서 공통적으로 **tongue tip**에서 조음된다.
9. 어떤 모음의 pitch가 128Hz이다. F1은 128Hz보다 반드시 크다. (T)
10. 어떤 화자의 /j/ 모음을 120Hz의 pitch로 발화했다. 같은 화자가 똑같은 pitch로 /a/ 모음을 발화했다. /i/와 /a/ 모음에 대해, 0Hz~5000Hz 사이에 몇 개의 harmonics가 존재하는지 구하고, 각각의 개수를 a와 b라고 할 때, b-a는? (0)
11. a=[[1],[2,3].[4,5,6]]

    n=0

    for bin a:

        for d in b:

            n+=1

    print(n)

    답: 6

12. 12-13번 총 몇 개의 unique한 함수가 있는가? (3개)
13. 몇 번 실행 되는가? (6번)

```
n=[1,2,3,4]
for i in range(len(n)):
    print(i)
```

14. 14-16번

```
a=[1,2,[3,4]]
```

위의 코드를 실행하면 결과가 4이다.

```
b=[1,'a',{'a':[3,6], 'b':[d]}, 'b']
print(b[-2]['b'][-1])
```

15. C=[1,'a',{'a':'abc', 'b':def'}]

```
Print(c[-1]['b'][-2])
```

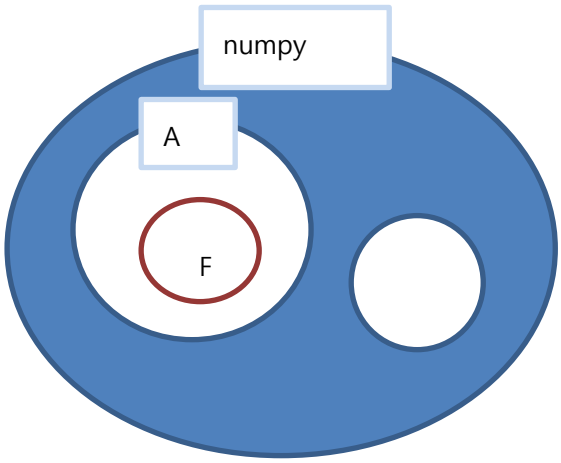
답은 2

16. 비슷한 문제

<중간고사 이후>

A. 숫자열을 가리켜 벡터라고 함.
B. 사진을 픽셀로 나누면 숫자 값을 넣을 수 있음. 직사각형의 형태로 행과 열: 행렬
C. 모든 데이터는 벡터의 형태로 되어야 함
D. 사진 흑백은 행렬데이터가 1장, 컬러는 3장(RGB)로 이루어짐
E. Array는 계산 가능한 것으로 바꿔 줌

<Numpy>



A. Import numpy as np #넘파이를 줄여쓰기 위해
B. Import matplotlib.pyplot as plt 얘는
   from matplotlib import pyplot as plt 얘로 바꿔서 쓸 수 있음
C. Per sec의 개념이 들어가면 항상 Hz 단위를 써야 함. Sampling rate(1초에 몇 개 조각으로 나누는가)도 그러함.

In [9]:
```
np.empty([2,3], dtype='int')

#[2,3] 2x3 2행 3열의 가로로 긴 것
```

Out[9]:
```
array([[72683148,        0,        0],
       [       0,        1,        0]])
```

In [10]:
```
np.zeros([2,3])

#0으로 채워진 행렬 리스트가 만들어짐
```

Out[10]:
```
array([[0., 0., 0.],
       [0., 0., 0.]])
```

In [11]:
```
#(0, 0, 0), (0, 0, 0) 얘를 2by3의 리스트를 만들고 싶으면
#쟤 자체는 계산이 안 되니까 쓸모가 없음, 그래서 계산 가능한 형태로 만드는 것

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

Out[11]:
```
array([[0, 0, 0],
       [0, 0, 0]])
```

In [12]:
```
np.ones([2,3])

#ones, zeros의 default type은 float이니까 1. 이렇게 나올 것
```

Out[12]:
```
array([[1., 1., 1.],
       [1., 1., 1.]])
```

In [13]:
```
np.ones([2,3], dtype='int')

#이러면 소수점 사라짐!
```

Out[13]:
```
array([[1, 1, 1],
       [1, 1, 1]])
```

In [19]:
```
np.arange(5)
```

In [25]:
```
np.linspace(0,10,6)

#얘는 arange랑 다르게 0, 10 포함, 또은 0,10 포함하여 6개로 똑같이 나눠준다
```

Out[25]:
```
array([ 0.,  2.,  4.,  6.,  8., 10.])
```

In [26]:
```
np.linspace(0,10,7)
```

Out[26]:
```
array([ 0.        ,  1.66666667,  3.33333333,  5.        ,  6.66666667,
        8.33333333, 10.        ])
```

In [31]:
```
x=np.array([[1,2],[3,4],[5,6]])
x
```

Out[31]:
```
array([[1, 2],
       [3, 4],
       [5, 6]])
```

In [33]:
```
#3차원도 표기가 가능함, 2차원은 대괄호가 위처럼 양옆이 2개씩 붙어있음, 3차원은 3개씩,
#2차원 행렬이 2개 나오면서 3차원이 됨
x=np.array([[[1,2],[3,4],[5,6]],[[7,8],[9,10],[10,11]]])
x
```

Out[33]:
```
array([[[ 1,  2],
        [ 3,  4],
        [ 5,  6]],

       [[ 7,  8],
        [ 9, 10],
        [10, 11]]])
```

In [34]:
```
x.ndim

#3차원이라는 것을 알려줌
```

Out[34]:
```
3
```

In [37]:
```
x.astype(np.float64)

#원래는 int32 data type인데 이걸 바꾸고 싶으면 astype함수를 씀
```

Out[37]:
```
array([[[ 1.,  2.],
        [ 3.,  4.],
        [ 5.,  6.]],

       [[ 7.,  8.],
        [ 9., 10.],
        [10., 11.]]])
```

In [38]:
```
np.zeros_like(x)

#형태를 유지하고 내용물을 0으로 바꾸고 싶을 때: zeros_like함수

#근데 이거는 x*0 이렇게 해도 됨
```

Out[38]:
```
array([[[0, 0],
        [0, 0],
        [0, 0]],

       [[0, 0],
        [0, 0],
        [0, 0]]])
```

In [46]:
```
#numpy 안의 random 패키지 안의 normal이라는 함수
#from numpy import random 이런 식으로도 쓸 수 있음?

data=np.random.normal(0,1,100)
#정규분포 모양의 데이터를 만들어주는 것임!!shape자체를 만들어주는 게 아니라
#normal(mean,standard deviation,data size)
```

In [52]:
```
data=np.random.normal(0,1,100)
print(data)
plt.hist(data, bins=10)
plt.show()
```

3.Numpy I/O

In [59]:
```
a=np.random.randint(0,10,[2,3])
b=np.random.random([2,3])
np.savez('test',a,b)

#test라고 적으면 파일로 저장됨
```

In [58]:
```
!ls -al test*
#위의 파일이 만들어졌는지 확인하는것
```

```
'ls'은(는) 내부 또는 외부 명령, 실행할 수 있는 프로그램, 또는
배치 파일이 아닙니다.
```

In [60]:
```
del a,b       #variable이 assign 안 된 상태로 없어짐
%who          #지금 available variable
```

```
data     np      plt     x       y
```

In [61]:
```
npzfiles=np.load('test.npz')     #이러면 불러올 수 있을 것에 npzfiles는 variable이름이니까 맘가나 바꿔도 됨
npzfiles.files
```

Out[61]:
```
['arr_0', 'arr_1']
```

In [62]:
```
npzfiles['arr_0']
```

Out[62]:
```
array([[7, 5, 4],
       [9, 8, 9]])
```

In [ ]:
```
data=np.loadtxt('regression.csv', delimiter=',',skiprows=1, dtype={'type':('x':'y'), 'formats'})

#skiprows 1번째 줄 뺀다

#csv는 comma separated value 콤마로 분류되어 있는 것.
#regression file을 깃헙에서 가져와서 해볼 것!!!
#그 파일을 다운 받아서
```