

## -English consonants and vowels

				1	2							
p	pie	pea		lowercase <i>p</i>	i	i	heed	he	bead	heat	keyed	lowercase <i>i</i>
t	tie	tea		lowercase <i>t</i>	ɪ	ɪ	hid		bid	hit	kid	small capital <i>I</i>
k	kye	key		lowercase <i>k</i>	eɪ	eɪ	hayed	hay	bayed	hate	Cade	lowercase <i>e</i>
b	by	bee		lowercase <i>b</i>	ɛ	ɛ	head		bed			epsilon
d	dye	D		lowercase <i>d</i>	æ	æ	had		bad	hat	cad	ash
g	guy			lowercase <i>g</i>	ɑ	ɑ	hard		bard	heart	card	script <i>a</i>
m	my	me	ram	lowercase <i>m</i>	ɑ	ɒ	hod		bod	hot	cod	turned script <i>a</i>
n	nigh	knee	ran	lowercase <i>n</i>	ɔ	ɔ	hawed	haw	bawd		cawed	open <i>o</i>
ŋ			rang	eng (or angm)	u	u	hood				could	upsilon
f	fie	fee		lowercase <i>f</i>	θ	θ	hoed	hoe	bode		code	lowercase <i>o</i>
v	vie	V		theta	u	u	who'd	who	booed	hoot	cood	lowercase <i>u</i>
θ	thigh			eth	Λ	Λ	Hudd		bud	hut	cud	turned <i>v</i>
ð	thy	thee	listen	lowercase <i>s</i>	ʒ	ʒ	herd	her	bird	hurt	curd	reversed epsilon
s	sigh	sea	mizzen	lowercase <i>z</i>	ai	ai	hide	high	bide	height		lowercase <i>a</i> (+I)
z		Z	mission	esh (or long <i>z</i> )	au	au		how	bowed		cowed	(as noted above)
ʃ (š)	shy	she	vision	long <i>z</i> (or yog)	ɔɪ	ɔɪ		(a)hoy	Boyd			(as noted above)
ʒ (ž)				lowercase <i>l</i>	ɪr	ɪə		here	beard			(as noted above)
l	lie	lee		lowercase <i>w</i>	ɛr	ɛə		hair	bared		cared	(as noted above)
w	why	we		lowercase <i>r</i>	aɪr	aə	hired	hire				(as noted above)
r	rye			lowercase <i>j</i>								
j (y)	high	he		lowercase <i>h</i>								
h												
Note also the following:												
tʃ (tš)	chi(me)	chea(p)										
dʒ (dž)	ji(ve)	G										
				Note also:								
				ju	ju	hued	hue	Bude		cued	(as noted above)	

-Phonetics: a study on speech

1. Articulatory phonetics(from mouth): how to produce speech
2. Acoustic phonetics(through air): how to transmit speech
3. Auditory phonetics(to ear):how to hear speech

-upper vocal tract: Lip, teeth, alveolar ridge, hard palate, soft palate(velum), uvular, pharynx wall, larynx

-lower vocal tract: lip, tongue tip, blade, front, center, back, root, epiglottis(음식이 기도로 가는것을 막음)

-larynx

1. Voiced: vibration- 성대가 붙어있을 때
2. Voiceless:no vibration-성대가 떨어져 있을 때

-velum lowered: nasal, breathing

-constriction

1.Constriction location

- 1) Lips-bilabial, labiodental
- 2) tongue body- palatal, velar cf)모든 모음은 tongue body를 사용
- 3) tongue tip- (inter)dental, alveolar, retroflex, palato-alveolar

2.Constriction degree

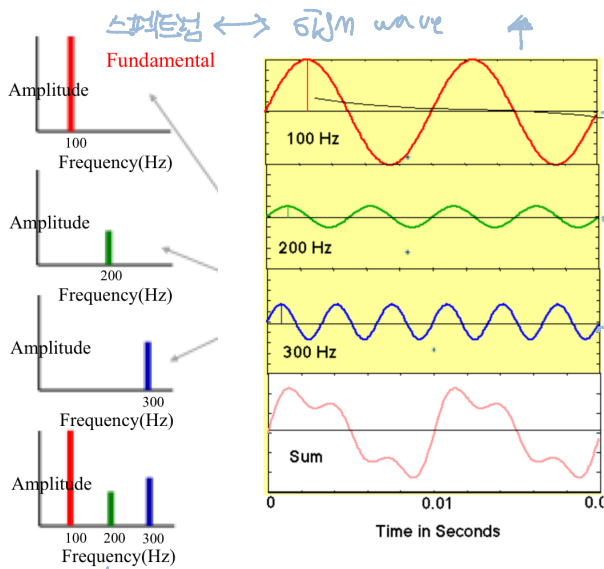
Stops>fricative>approximants>vowel

-phonemes: individual sounds that form words

## \* Spectrum

모든 소리는 spectrum으로 이루어져 있음.

time x value (frequency)



simplex (Pure) tone

Complex tone

: Simplex tones을 합성한 결과.

다) Phasor 기전용

Spectrum: 숫자 입력.

또는 sine wave

frequency x amplitude (magnitude)

(주파수 x 그 주파수의 크기)

## \* Source (목구멍에서만 나오는)

- Human voice source consists of **harmonics** 배음 증가
- A complex tone = sum of pure tones at integer multiples of the lowest pure tone
- the lowest pure tone
  - Fundamental frequency (F0) = Pitch
  - rate of vibration of the larynx
  - the number of opening-closing cycles of the larynx per second (보통 인의 진동수)
- Amplitude of pure tones gradually decreases

소리를 줄이는 기능.

다) 여 F0 ↑ → 등음성  
남 F0 ↓ → 탁음성

## \* Filter

- Compare spectrums between audio and EGG

- EGG: gradual decreasing

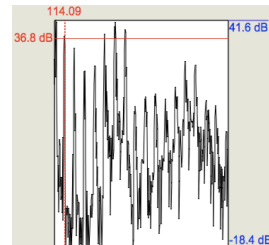
- audio: peaks/mountains and valleys

harmonics인 것 그대로이나, amplitude가 gradual decrease가 아님 → curved.

- Because it is filtered by the vocal tract (VT)

- peaks/mountains: frequencies VT likes = formants

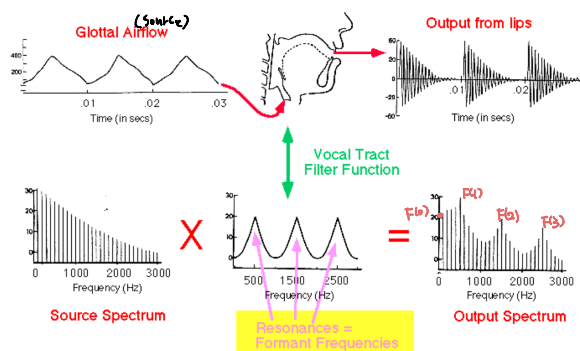
- valleys: frequencies VT does NOT like



## \* Synthesizing Source

- convert to mono → 직접에 sine wave와 주기 일치, 높이도 직접에 것만 비슷.
- 무한대로 가면, pulse train.

## \* Source-filter theory



Formants: Peaks of frequency.

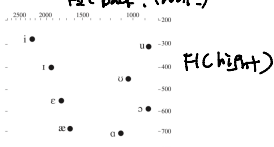
## \* Spectrogram

- Dark band: mountains = Formants

## \* F1, F2만으로 모음구분 및

F2 (back-front)

F1 (high-low)



코딩 = 자용재 (ex. 폰에있는 프로그램들)

언어 ~~의~~

사람의 언어 [ 단어 : 정보 (단어는 정보를 담는 그릇)  
= 문법

컴퓨터 언어 [ 변수 : 정보 ex) 숫자, 문자...  
= 문법

① 변수에 정보를 assign

② 조건 (if)

③ 반복 (for)

★ ④ 함수 : 입력 → 출력

$x = y$  ← y를 x로 assign. ex)  $a = 1$  / **Run** (단축키: Shift + enter)

들어갔는지 확인 : `print(a)` (a : 위에 열거했 / select ⓧ = delete.

셀 앞부분을 클릭하고 파란바드이면 (b : 아래에 " /

문자 입력 : ' ' ~~의~~

• `love = 2`   `b = love`   `print(b) = 2`

• `a=1`  
`b=1` → run → 3   *마지막 하나는 print하셈.*  
`c=3`  
`c`  
`//`

`a=1; b=2; c=3;`

• 한 변수에 여러개 넣는법 → 대괄호 ex) `a = [1, 2, 3, 5]`  
(list)   *괄호 안으로 (숫자) tuple*

• `type(변수)` → 어떤 type 인지 ex) `int`, `list`, `float`, `str`  
(숫자)   *소수점이 있는 숫자.*   (문자)

• `a = [1, 'love', [1, 'bye']]`

• `a = {'a': 'apple', 'b': 'banana'}`   *표제어: 설명.*  
↳ `type: dict (dictionary)`

A=1, b=1 c=a+b  
C=2

A=[1,2]  
B=[3,4]  
C=a[0]+b[0]  
C  
-4(1+3)

C=a[1]+b[1]  
C  
-6

A=1.2 a=int(a) print(a) -1  
A='123' print(a[1]) -1. Q. ???  
A=123 print(a[1]) - error ← 이전 index는, 문자나 숫자에만 적용

Index: 내부적인부분

Dict에있는 정보를 가져올때는 pair중의 앞부분을 index의 수단으로 쓴다. 그러면 뒷부분이 나옴.

Type(a[0])

S='abcdef'  
N=[100,200,300]

Print s[0] 제일 첫 번째 꺼 .....  
s[-1] 제일 마지막 꺼 .....  
S[1:3] = 첫번째에서 세번째의 직전까지 / b,c  
1: = 첫번째에서 끝까지  
:3 = 첫번째에서 세번째 의 직전까지

Len[] = 길이

S.upper() - 대문자 됨

R index ??

Strip: 쓸데없는거 없애주는  
Split: ' ' -스페이스를 사용해서 나눠라

a='123' print(a[1]) → 2 (그냥 문자에이저드, 두번째꺼)  
(two가 아니라 이런 문자면)

( )로 묶으면 tuple, [ ]로 묶으면 list.

dic → {"a": "apple" }

print(a["a"])

S="abcdef"  
print(s[1:3], s[1:], s[:3], s[:])  
bc bcdef abc abcdef.

S.upper()

result = s.find( ) : 문자가 포함. 몇번까지? \* 0부터 시작  
result = s.index(?)  
S = s.strip() ← print(s)  
tokens = s.split(' ') ← print(tokens)  
S = ' '.join(tokens)  
S = s.replace('this', 'that')

```
a= [1,2,3,4]
for i in a:      # for in : 라는 문법은 in 뒤에 있는 것을 i에 할당하는 것
    print(i)     # for과 if 다음에는 indent를 하는 것이 매우 중요하다.
1234
```

```
a=[1,2,3,4]
for i in range(4): #range는 index를 만들어주는 것 range(4): 0부터 4개 (0,1,2,3)
    print(a[i])    #range(len(a)) 라고 일반적으로 함.
                  # len(a): 그냥 몇개지
1234
```

```
a=['red','green','blue','purple']
for s in a:
    print(s)
red green blue purple
```

```
a=['red','green','blue','purple']
for s in range(len(a)):
    print(a[s])
red green blue purple
```

```
a=['red','green','blue','purple']
for i,s in enumerate(a) :          #enumerate: 숫자를 부여
    print(i,s)
0 red 1 green 2 blue 3 purple
```

```
a=['red','green','blue','purple']
b=[0.2, 0.3, 0.1, 0.4]
for i, s in enumerate(a) : #앞에있는게 번호, 뒤에있는게 element. enumerate 없으면 변수 2개 못씀
    print("{}:{}".format(s, b[i]*100))
red: 20.0% green:30.0% blue:10.0% purple:40.0%
```

```
a=['red','green','blue','purple']
b=[0.2, 0.3, 0.1, 0.4]
for s,i in zip(a, b):
    print("{}:{}".format(s, i*100))
red: 20.0% green:30.0% blue:10.0% purple:40.0%
```

```
a=0
if a==0:
    print("yay!")
yay!
```

```
if a!=0:
    print("yay!")
else:
    print("no")
no
```

```
#시험문제
for i in range(1,3):
    for j in range(3,5):
        print(i*j)
3468
```

```
for i in range(1,3):
    print(i)
    for j in range(3,5):
        print(i*j)
```

1 3 4 2 6 8

순서: 1: 1을프린트, 1을 3,4와 곱함

```
for i in range(1,3):
    for j in range(3,5):
        if j>=4:
            print(i*j)
```

4 8

```
for i in range(1,3):
    if i >= 3:
        for j in range(3,5):
            print(i*j)
```

아무것도안나옴

# Numpy

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

```
np.empty([2,3], dtype='int') → 2행 3열의 행렬 random
np.zeros([2,3]) → 0으로 구성된 행렬
np.arange(0,10,2, dtype='float64') → 0~10전까지, 2씩증가 array([0., 2., 4., 6., 8.])
np.linspace(0,10,6, dtype=float) → 0~10까지를 6개로 나눌. array([ 0., 2., 4., 6., 8., 10.])
```

```
X = np.array([[1,2,3],[4,5,6]]) → 행렬에 들어가는 것들을 직접 쓰는 방법
```

```
X.astype(np.float64) → 행렬에 들어가는 것들의 type 바꿀
```

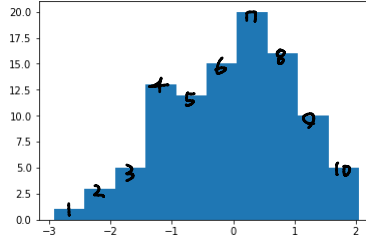
```
np.zeros_like(X) → 행렬에 들어가는 것들을 0으로 바꿈
```

```
data = np.random.normal(0,1,100) → 정규분포
```

```
print(data)
```

```
plt.hist(data, bins=10) → histogram 생성, 바구니가 10개
```

```
plt.show()
```



```
X = np.ones([2, 3, 4])
```

```
Y = X.reshape(-1, 3, 2)
```

```
np.allclose(X.reshape(-1, 3, 2), Y)
```

```
a = np.random.randint(0, 10, [2, 3])
```

```
b = np.random.random([2, 3])
```

```
np.savez("test", a, b)
```

```
del a, b
```

```
%who # Print all interactive variables
```

```
npzfiles = np.load("test.npz")
npzfiles.files
```

```
['arr_0', 'arr_1']
```

```
npzfiles['arr_0']
```

```
np.savetxt("regression_saved.csv", data, delimiter=",")
```

```
!ls -al regression_saved.csv
```

```
arr = np.random.random([5,2,3])
```

```
print(type(arr)) <class 'numpy.ndarray'>
```

```
print(len(arr)) 5
```

```
print(arr.shape) (5, 2, 3)
```

```
print(arr.ndim) 3
```

```
print(arr.size) 30
```

```
print(arr.dtype) float64
```

```
a = np.arange(1, 5)
```

```
b = np.arange(9, 5, -1)
```

```
print(a - b) [-8 -6 -4 -2]
```

```
print(a * b) [9 16 21 24]
```

```
a = np.arange(1, 10).reshape(3,3)
```

```
b = np.arange(9, 0, -1).reshape(3,3)
```

```
a.sum(), np.sum(a) a.sum(axis=0), np.sum(a, axis=0) a.sum(axis=1), np.sum(a, axis=1)
```

In [118]:

입력을 degree로 받지 않고 radiant로 받는다.

```
from matplotlib import pyplot as plt= import matplotlib.pyplot
```

## Phasor : sin, cos 의 한 주기

In [160]:

```
# parameter setting
amp = 1          # range [0.0, 1.0]
sr = 10000       # sampling rate, Hz =해상도
dur = 0.5        # in seconds
freq = 100.0     # sine frequency, Hz =1 초에 몇번 왔다갔다
```

In [161]:

```
# generate time 시간의 개념이 있어야 실질적 소리를 만들 수 있음
```

```
t = np.arange(1, sr * dur+1)/sr #이런거보고 tic 의개수구하기
```

In [162]:

t=0.0001,0.0002,...0.5000 (duration이 0.5니까 이만큼 하는 것, sr10000이라서 이렇게나눔)

t=np.arange(1,sr+1) 하면 만개가 생기는데 그러면 1초. 그러나 dur이 0.5

```
t=np.arange(1, sr*dur+1)
```

이거를 다시 sr로 나누어야 하나의 단위가됨

```
t=np.arange(1, sr*dur+1)/sr
```

e-04=1/10000 e-01=1/10

Theta: x축, (0~2파이) s: 결과값 = 총 7개

```
Theta=np.arange(0, 2*np.pi, 0.1)/더 뽁뽁하게 만드려면 여기서 세번째꺼를 더 작게
```

```
# generate phase
theta = t * 2*np.pi * freq
time의 벡터 사이즈= 세타의 벡터 사이즈 (t에다가 뭔가를 곱하기만 한 게 세타니까)
```

t가 1이고 1에 2파이를 곱하면 그냥 한바퀴도는 것. 거기에다가 몇 바퀴 도는지 정하는 게 freq

```
# generate signal by cosine-phasor
s = np.sin(theta)
```

In [164]:

```
fig = plt.figure()
```



```
ax = fig.add_subplot(111) # Subplot: 221: 2행2열짜리, 1:첫번째 꺼
```

```
ax.plot(t[0:1000], s[0:1000], '.')
```

# t,s 를 1000 개만 플라팅:점들의개수  
총 1000 개/[     ]안의 숫자들 달라지면 안됨/ 강 t,s 라고만하면 태극문양안만들어짐  
#.대신 -하거나 아무것도없으면 선으로연결

Subplot: 221: 2행2열짜리, 1:첫번째 꺼

```
ax.set_xlabel('time (s)')
```

label 은 그냥 이름 정하는 것

```
ax.set_ylabel('real')
```

<IPython.core.display.Javascript object>

Equidistance= x축에서는 하지만, y축에서는 하지않음.

linear하면(선처럼 생기면, 1차함수) x,y축에서 둘다 equidistance.

```
Text(0, 0.5, 'real')
```

*# generate signal by complex-phasor*  
`c = np.exp(theta*1j)`  
exp 가 오일러함수 그위에 theta\*1j 가 들어감. j 는 i 임

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot(t[0:1000], c.real[0:1000], c.imag[0:1000], '.')
```

```
# a+bi 에서 real: a, imag: b
```

#projection

time은 항상 볼 수 밖에 없음

real만 보려면: 위에서 = cos함수와 유사

imaginary = sin