영어음성학

2016130766 김효주

1~3주차

먼저 Praat이라는 프로그램을 다루는 것을 배웠다. Praat은 음성을 분석하는 데 사용되는 프로그램이다. 그 다음주에는 기본적인 영어 음성학에 대해서 배웠다. 음성학(Phonetics)과 음운론(Phonology)은 비슷해보이지만 차이가 있다. Phonology는 절차적, 인지적인 부분을 다루지만 phonetics는 더 깊고 물리학적인 부분을 다룬다. 영어의 모든 소리는 유성음(voiced sound)과 무성음(voiceless sound)으로 나눠지는데 이는 목이 떨리는 것과 떨리지 않는 것으로 구분할 수 있다. 모음은 전부 유성음에 해당된다. 또한 코로 나오는 소리와 그렇지 않은 소리가 있는데 코로 나오는 소리인 비음에는 m, n, ŋ이 있다. 모음은 monophthongs (단모음)와 diphthongs (복모음)로 나눠진다. 조음 기관에는 lips, tongue tip, tongue body, velum, larynx가 있다. Epiglotis는 기도로 가는 길을 막는 뚜껑 역할을 해서 침을 삼길 때 등의 경우에 기도로 이물질이 들어가 숨이 막히는 일을 방지해준다. 비음을 발음할 때는 velum이 lowered 된다. 모든 모음과 비음이 아닌 자음을 발음할 때는 velum은 raised되고 nasal track은 막힌다. 코로 숨을 쉴 때는 velum이 lowered 되고 nasal track은 열린다. Larynx는 무성음과 유성음을 구분하는 조음기관 역할을 한다.

4주차

Praat에서 분석할 때는 특정 부분을 선택한 후 선택한 시점에 주파수별로 어떤 성분이 많은지 분석한다. x축은 시간, y축은 frequency가 나온다. F0은 맨 처음 주파수를 말한다. 그 이후 sine wave는 점점 작아지는데 이는 배수의 형태로 amplitude는 점점 작아지지만 frequency는 점점 커지는 형태를 갖는다. 모든 사람의 소리가 그렇지만 모든 소리가 그런 것은 아니다. Source의 경우 고주파로 갈 수록 에너지가 약해진다. Filter의 경우에도 전반적으로는 그렇지만 별개로 어떤 곳에서는 에너지가 낮고 어떤 곳에서는 에너지가 높다. Simplex tone들의 합은 complex tone이라고 한다. Filter의 특징은 vocal tract가 입모양이라는 것이다. Vocal tract는 vocal tube 와 같은 맥락으로 이해하면 편하고 filter을 거쳐 filtered 된다고 말한다. 까만 부분을 peak라고 하고 이것은 산맥에 해당된다. 흰색은 valley라고 한다. 산맥은 불규칙적으로 일어나는 것이 아니라 사람들이 각각 ‘ㅏ’라고 할 때와 ‘ㅣ’라고 할 때 패턴이 모두 같다. 소리 별 패턴은 다르지만 누가 그 소리를 내든 동일한 패턴을 보인다는 것이다. 저주파로부터 첫 번째 산맥, 두 번째 산맥, 세 번째 산맥 등을 각각 first formant(f1), second formant(f2), third formant(f3) 등으로 표시한다. 이 때 위에서 언급한 F0과 헷갈리지 않는 것이 중요하다. 피아노 건반이나 기타줄처럼 특정 입모양은 특정 음의 높이를 내도록 소리가 정해져 있다. Voice source를 만들어 100hz, 200hz, …, 1000hz 10개를 combine to stereo하면 pure tone을 들었을 때보다 더 사람 목소리와 비슷해진다. 그러나 이는 그냥 합쳐서 들은 것일 뿐 numerically하게 더해진 상태는 아니다. 이를 convert to mono하면 합쳐진 complex tone이 된다. 반복 주기는 첫 번째 만들었던 것(F0)과 일치한다. 소리도 100hz와 일치한다. 그래서 F0을 fundamental sound, pitch라고 한다. 만약 이런 식으로 무한대로 계속 합하게 되면 한 쪽만 peak가 있는 형태가 되어 부드럽지 않게 되고 점점 무너지게 될 것이다. 즉 하나의 peak가 있고 나머지는 다 0인 형태인 것이다. 이것을 pulse라고 하고 이런 것이 여러 개 늘어져 있는 것을 pulse train이라고 한다. f1과 f2를 나타낸 그래프를 보면 소리마다 간격이 다른데 이 간격 차이는 누가 녹음하든 큰 차이가 없다. f1은 height, f2는 frontness 또는 backness를 결정한다. 영어가 더 backness가 크며 한국어는 상대적으로 더 적게 활용하는 것을 알 수 있다.

5주차

반복되는 것을 자동화하기 위해 코딩

모든 언어는 **단어 & 문법**이 있음. 단어 combine - 이런 저런 정보가 됨 → commuication

단어란 무엇인가 - 단어 속에는 의미가 포함되어 있음. 정보가 들어있음. 정보를 담는 그릇. 어떤 단어를 선택할 것인가 등 - 문법, combination. 숫자, 문자 등 정보 담을 수 있음. 사람과 기계와의 communication을 위해 문법이 필요함. 사람의 문법은 규칙들이 매우 많지만 그에 비해 기계와의 문법은 그렇게 어렵지 않음.

① 변수에 정보를 넣는 것 (variable assignment)

② if 문법 사용 (if conditioning)

③ 자동화의 가장 중요한 부분 - 여러번 반복 : for (for loop)

④ 함수 - 내부적으로 돌아가는 것이 뭔지는 몰라도 어떤 입력을 넣으면 내가 원하는 결과가 출력되는 것. praat, 자동차 등은 입출력이 확실히 있음 - 함수의 예시가 될 수 있음. 정보의 종류는 숫자와 글자 두 가지.

=: 왼쪽에 있는 변수에 오른쪽의 정보를 assign 한다. a=1 이면 a라는 variable에 1이라는 정보를 assign하는 것.

Print는 누군가 만들어 놓은 함수: 어떤 변수를 그 안에 넣으면 그 속에 있는 게 뭔지 print out해주는 변수. 함수 이름 + 괄호 안에 변수 이름. ex) print(a). print라는 함수의 입력은 a. 출력은 1.

Jupyter notebook에서 입력창을 선택하고 ‘b’를 누르면 밑에 새 줄 생김. ‘a’를 누르면 위에 새 줄 생김. 셀 지우려면 선택하고 ‘x’. 선택하는 셀 따로 실행 가능. a라는 변수는 하나밖에 없으니 다른 변수가 필요하면 직접 만들거나 해야 함. 그냥 알파벳 쓰면 그냥 변수, 숫자는 그냥 숫자. 문자를 쓰려면 인용 필수. 실행: 선택 + shift + enter

6주차

리스트: [ ]

튜플: ( )

딕셔너리: 여러 개 넣는 것은 리스트와 똑같지만 pair로 여러 개 넣음. { }

[ ] 역할: 여러 개 중 정보 취합 선택 필요할 때 번호를 정해주는 것. 0번째가 제일 처음 원소.

;: 두 줄 적을 것을 한 줄에 적고 싶을 때

내부적인 정보 부분 추출할 때는 리스트 이름 적은 다음 대괄호, 그 안에는 인덱스

Pair set에서 앞부분을 인덱스로 쓴다. 숫자 대신 pair의 앞부분이 인덱스 표현 방법이 되는 것.

인덱스: 제일 처음 것은 무조건 0, 제일 마지막은 무조건 -1, [1:3]이면 첫번째에서 두번째까지. [a:b]: a번째에서 b번째 직전까지.

rindex: 여러 개일 때 제일 마지막 것의 인덱스를 해라 (right index)

split(‘ ‘): ‘ ‘(space)를 구분점으로 해서 잘라라. 간격 기준으로 잘라서 리스트 만들어줌.

split과 join 한 쌍으로 유용하게 쓰임. 리스트로 되어 있는 것을 문장으로 복구하고 싶을 때 join 사용.

replace(‘this’, ‘that’) : this를 다 that으로 바꾸기

Function - len, type, for, if 등

For i in a : a에 있는 것을 하나씩 i에 할당하고 밑의 명령 실행.

in뒤에 range함수 : range 뒤에 어떤 숫자가 나오면 리스트를 만드는 것. 예를 들어 4면 0부터 3까지 리스트. 4는 인덱스를 의미하는 것. 루프를 돌면서 i가 처음에는 0을 받고 그래서 a의 0번째 원소를 출력하는 것. 위랑 똑같아야 함. ① in 뒤에 그대로 주는 방법. For 다음에 있는 variable 계속 바뀜. ② for 다음에 리스트 그대로 쓰는 게 아니라 range함수를 이용해 인덱스를 주는 방법. 만약 그냥 print(i)라고 하면 0, 1, 2, 3 출력될 것. 더 일반적으로는 len(a)라고 적으면 길이를 그대로 받음. 더 flexible. 첫번째 것은 0, 두번째는 1, … for loop의 내용은 반드시 인덴트 들어가야 함.

enumerate: 번호. 앞에 것이 번호, 뒤에 것이 자기 자신 element. 그냥 a 이런식으로 되어 있으면 i, s 이런 식으로 쓸 수 없음.

9주차

데이터: 숫자 데이터, 이미지, 영상, 음성 데이터

숫자 아닌 데이터: 숫자로 표현됨. 숫자의 열로 표현. 그런 숫자의 열이 벡터. 그림의 경우, 그림 확대했을 때 보이는 정사각형 하나가 숫자 하나. 검정: 0, 흰색: 10, 회색: 그 사이의 숫자. Rgb 세 장으로 표현. 모든 색 다 표현 가능. 동영상은 그게 계속 움직이는 것. 흑백 이미지: 2차원. Color image: 3차원. 시간까지 있으면 4차원 됨. 차원의 개념 알아야 함. 소리: 영상처럼 소리가 어떻게 vactorized 되는지 표현.

numpy라는 library. 그 안에 package a, b, c, … 등 만들 수 있음. numpy.a.d.~~~ 이런 식으로 쓸 수 있음. '.'은 그 패키지 안에 있는 것이라는 의미. import numpy 라고 쓸 수 있고 from numpy라고도 쓸 수 있음. 그 안에 있는 a 불러온다면 from numpy import a.d 라고 표현한다. np.zeros라고 쓰면 그 행렬 안의 모든 원소가 0이 된다. 이와 비슷하게 np.ones라고 하면 모든 원소가 1이 되는데 이 경우 '1.'이라고 표시된다. 이 때 1.이라는 것은 이 원소의 format이 int가 아니라 float이라는 것이다. zeros라는 함수의 default가 float이라는 것을 알 수 있다. 1로 표기하기 위해서는 dtype = 'int'라고 적어주면 된다. 정교한 표현을 하고 싶을 때는 float64와 같은 형식을 사용할 수 있다. 대신 이는 메모리를 많이 차지한다. 정확도 및 데이터 표현 가능한 양과 데이터가 사용하는 메모리 용량은 반비례한다는 것을 알 수 있다. arrange는 계산 가능한 array이다. np.arrange(5)라고 쓰면 5개의 인덱스가 만들어진다. range를 정해줄 수도 있다. 예를 들어 0부터 10까지 정한다고 하면, np.arrange(0,10)이라고 표현할 수 있다. np.arange(0, 10, 2)와 같이 표시한다면 0부터 10까지 증가분이 2인 인덱스를 만드는 것이다. linspace는 linear space라는 뜻으로 처음부터 끝까지 다 같은 간격을 만들어주는 것이다. 이 경우 지정한 범위의 처음 숫자와 마지막 숫자까지 포함하게 된다. 배열로 2차원 뿐만 아니라 3차원 표시도 가능하다. np.array() 안에 대괄호가 두 개 붙어 있으면 2차원, 세 개 붙어 있으면 3차원인 것이다. X.ndim은 X의 차원을 알려준다. X.shape은 X의 모양을 알려준다. normal은 normal distribution, 즉 정규분포를 만들어주는 것이다. 그 모양을 만들어주는 게 아니라 정규분포를 위한 데이터를 직접 만들라고 명령하는 것이다. 예를 들어, data = np.random.normal(0, 1, 100)과 같이 표시할 수 있다. 정규분포 모양을 만들기 위해서는 평균과 그래프가 얼마나 넓은지를 알아야 한다. 0과 1이 각각 그것을 알려준다. 100은 100개의 데이터를 만들라는 의미이다. allclose는 두 개의 배열을 비교해 똑같을 경우 true를 출력해준다. random.random은 random한 것을 지정한 배열만큼 출력하는 것이다. savez는 파일로 저장하는 것이다. variable은 지우지 않는 한 메모리에 그대로 남아있는데 그 variable을 assign되지 않은 상태로 지우고 싶을 때는 del을 사용한다. #who는 지금 사용 가능한 assign된 변수가 무엇인지를 알려준다. delimeter은 분리, skiprows는 줄을 빼는 것을 의미한다.

10주차

inusodial이란 sine, cosine wave 또는 그들로 구성된 곡선들을 말한다. Sinsoidal function을 만드는 것을 phasor이라고 한다. sin함수의 입력값은 sin( )와 같이 표현할 때 괄호 안에 들어가는 값을 말하는데 이는 radian 값으로 만들어줘야 하고 이는 cosine 함수도 마찬가지이다. 0부터 2pi는 0 °부터 360 ° 와 같이 표시하는데 전자를 radian, 후자를 degree라고 표현한다. 오일러 공식은 다음과 같다. $e^{\theta i} = \cos(\theta) + \sin(\theta)i$. 여기서 e는 2.71…로 자연로그의 밑 부분에 해당한다. i는 imaginary의 약자로 허수이며 √(-1)의 값을 가진다. i와 e는 고정되어 있고 theta의 radian 값에 따라 값이 변형되는 함수인 것이다. $f(\theta) = e^{\theta i}$ 라는 함수의 경우 i와 e는 고정된 숫자이고 θ도 숫자이기 때문에 결과값도 숫자값이 나와야 한다. 복소수는 a + bi 로 표현되며 어떤 수든 표현 가능하다.

matplotlib는 numpy를 import하는 과정에서 가장 큰 라이브러리이다. 그 밑에 sublibrary가 있고 from을 써서 그 안에 있는 pyplot을 불러올 수 있다. 각도의 값을 먼저 만들기 위해서는 0부터 2pi까지 각도값 벡터를 만들어야 한다. 이는 arange를 통해서 만들 수 있다. sine이라는 함수는 numpy 안에 있는 함수를 불러오는 것이므로 np.sin을 사용한다. figure을 function으로 해서 fig이라는 것을 만들어준다. fig 먼저 만들고 subplot을 하면 화면분할을 할 수 있다. 221이란 총 22(2행 * 2열)로 나누는데 그 중 첫 번째 것을 선택하는 것이다. subplot 바로 밑에 실제 plotting하려고 하는 x (theta)와 y (sine 함수의 결과)의 값을 직접 적어주면 된다. theta는 우리가 0부터 2pi까지 만든 것이다. s는 그 결과값이다. sin 함수의 결과라는 것을 알 수 있다. 0부터 시작하면 너무 sparse하다는 문제가 있어 더 빽빽하게 만들어주어야 한다. 간격 0.1로 바꿔주면 이를 해결할 수 있다. x축과 y축 이름 달아주기는 set_xlabel, set_ylable을 통해 할 수 있다. x축 상황에서는 equidistant하고 y축 상황에서는 그렇지 않다. x축에서도 equidistant하고 y축에서도 equidistant 하다는 것은 linear한 경우이고 linear하지 않은 경우에는 그렇지 않다. line같이 생긴 것의 예시는 y = 2x와 같은 경우가 있다. line의 형태는 y = ax로 나타난다. 이를 제외한 모든 함수의 x와 y의 관계는 nonlinear하다. 그래프는 기본적으로 선으로 나타나는데 '.'를 추가하면 점으로 나타난다. theta 범위가 2pi 말고 10pi면 다섯번 도는 것(10pi = 2pi * 5)이다. 소리는 시간이 있어야 한다. 시간이 없으면 얼마간의 시간동안 나타나는 건지 모르니까 소리를 만들 수 없다.

11주차

spectrogram에서 x축은 time, y축은 frequency에 해당된다. 스펙토그램에서 한 슬라이스만 잘랐다고 생각하고 그 시점에서 어떤 frquency 성분이 많은가를 분석해야 한다. 이때는 x축을 frequency, y축을 time으로 둔다. frequency 스펙트럼은 100hz부터 5000hz까지 있을 것이다. 점점 고주파로 가도록 그래프 만들고 그 다음에 산맥을 하나 만든다. 그 다음 위치에 계속해서 산맥을 만든다. 이 작업을 하면 사람 목소리로 바뀌게 된다.

440hz는 '라'소리가 나고, 배수는 다 같은 음이 나온다. 즉, 옥타브만 달라지는 것이다.

sine의 라와 cosine의 라는 똑같고 소리 변화가 없다. 그래프에서 시작점만 다르다. 모양은 같지만 그냥 이동시킨 것이다. pi/2 차이가 난다. 만약 pi/8 만 이동시킨다고 해도 소리는 바뀌지 않는다. 얼마나 이동시키는지는 소리와 상관이 없다. 이 각도를 phase라고 함. 우리 귀가 phase에는 sensitive하지 않다.

phase shift는 알 수 없다. frequency에 sensitive 하다. complex-phasor에서 나온 건 바로 plotting 할 수 없다. real 값만 가능하며, imaginary part를 y축으로 해서 이차원 상에서 찍는 방법으로 plotting 한다.

bandwidth는 산맥의 폭을 결정하는 요소이다.

gradually decreasing 하게 만들어야 한다. 큰 산을 안 보이는 쪽까지 만들면 부드럽게 만들어진다.

입술이 있기 때문에 소리가 더 공명되어 퍼져나갈 수 있는 것이다.

12주차

m*n행렬은 m행 n열로 이루어진 행렬이다. Column vector은 세로로 길게 생긴 벡터로 항상 1열이기에 열벡터라고도 한다. 어떤 점은 벡터로 나타낼 수 있고 기하, 즉 그래프 상에서도 나타낼 수 있다. 2차원에서 점 하나, 3차원에서 점 하나 이런 식으로 만약 100차원이라는 벡터가 있다고 해도 차원이 늘어날 뿐 점이 하나인 것은 동일하다. 벡터의 multiplication 역시 벡터와 기하 둘 다로 나타낼 수 있다. 벡터의 경우 곱해지는 수와 각 원소를 곱한 값을 계산해 나타낸다. 기하의 경우 역시 원래 있던 자리에서 곱해지는 수 만큼을 배하여 표시하면 벡터의 곱 수치와 같은 결과가 나온다. 만약 0.5가 곱해지는 수라면 행렬의 경우에는 모든 원소와 곱한 값을 수치로 나타내고 기하의 경우에는 원래 위치에서 반을 줄여 그래프에 나타내는 것인데 둘은 동일한 값이 되는 것이다. 덧셈의 경우에도 두 가지 모두로 나타낼 수 있다. Linear combination이란 벡터들에 상수를 곱하거나 더한 조합을 나타낸다. 나누기나 제곱승 연산 등은 해당되지 않는다. 무수하게 많은 벡터들이 만들어내는 공간을 vector space라고 한다. 예를 들어, (3, 4)와 같은 차원의 벡터를 무한하게 만든다면 그 vector space가 차지하는 공간은 2차원을 다 덮을 것이다. 즉, 모든 가능한 벡터들이 다 덮여진 공간이 vector space인 것이다. 만약 전체 차원 공간의 일부분만이라고 가정한다면 linear combination 했을 때 어떤 공간에 점이 찍힐지 모른다. Vector space의 규정은 linear combination의 결과까지 포함해야 한다. 즉, 한 차원의 전체로 생각해야지 일부분만은 불가능한 것이다. n차원의 공간은 n개의 component로 구성된다. 이 모든 공간이 채워져 있는 부분을 n차원의 vector space라고 한다. Column vector (2, 1)과 (-1, 3)이 있다고 하자. 이를 2차원의 공간에서 찍은 후 두 점을 가지고 linear combination을 하면 또 다른 지점이 생길 것이다. 이 과정을 무한번 반복하면 모든 공간이 채워지게 되기 때문에 두 column vector은 plain이 될 것이다. 이게 바로 column vector에 의한 column space이다. Column space는 column vector보다 차원이 높을 수 없다. 평면 나오는 것을 확장시키면 column space가 되는데 이를 spanning이라고 한다. 원점과 column vector을 다 연결하면 삼각형이 만들어진다. 그것은 평면 위에 있고 확장시킬 수 있는데 그것이 column space이다. Column vector들이 같은 선 상에 있지 않으면 independent하다고 한다. 두 column vector가 한 선 상에 있으면 dependent하다고 한다. 원점과 두 벡터를 연결했을 때 삼각형이 만들어져야 spanning하고 평면 만들기가 가능하다.