음성프로젝트
세 미 나

ToBig's 13기 신민정

음성&DSP 기초

안뇽?
만반잘부

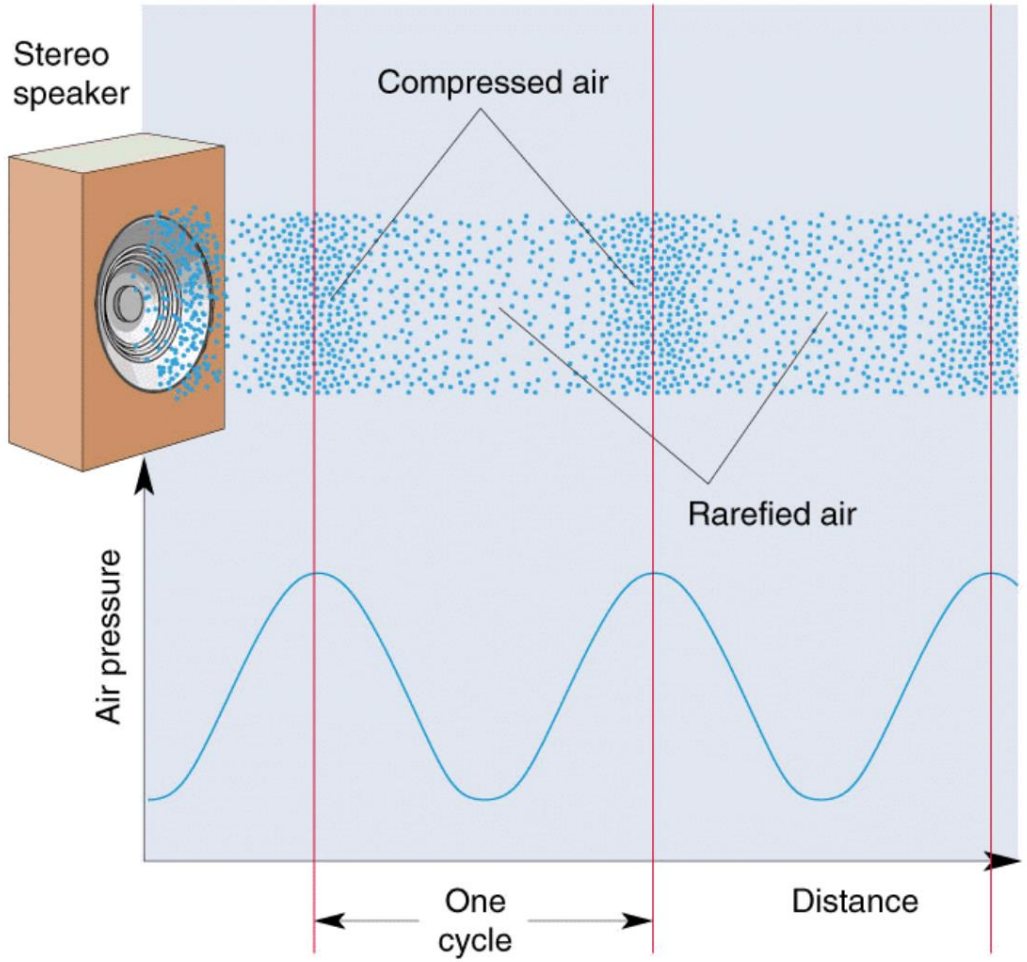
Contents

01 음성신호의 이해

Unit 01 | 음성신호의 이해

소리 Sound

소리 = 진동으로 인한 공기의 압축
압축이 얼마냐 됐느냐 = Wave

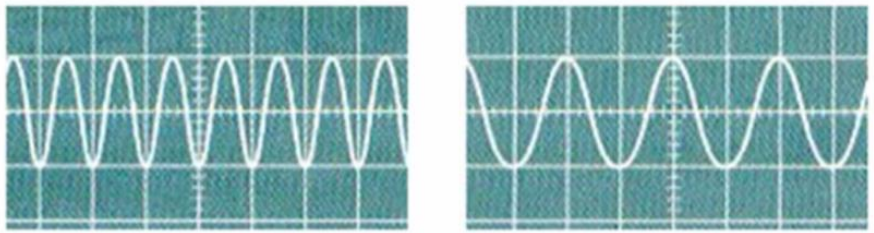


Unit 01 | 음성신호의 이해

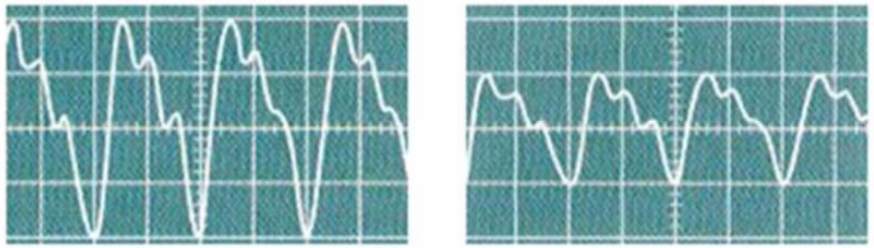
소리의 물리량

- Amplitude(진폭)
  소리 진폭의 세기
- Frequency(주파수)
  소리 떨림의 빠르기
- Tone-Color(Wave form)
  소리 파동의 모양

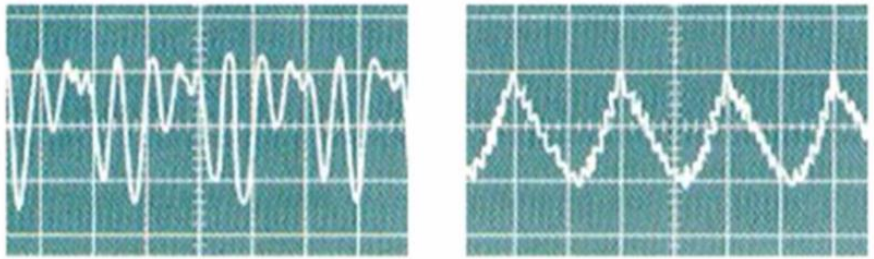
이상 소리의 (물리적) 3요소



높이가 다른 두 소리



세기가 다른 두 소리

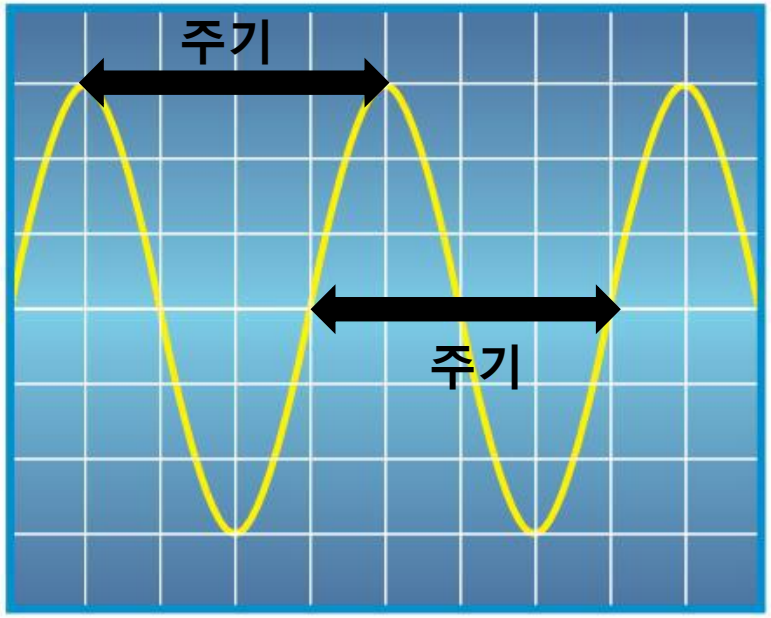


맵시가 다른 두 소리

Unit 01 | 음성신호의 이해

Frequncy : Number of compressed

주기(Period) : 파동이 한번 진동하는데 걸리는 시간
주파수(Frequency) : 1초 동안 진동한 횟수



주기 = 1/주파수

높은소리 = High Freq
낮은소리 = Low Freq

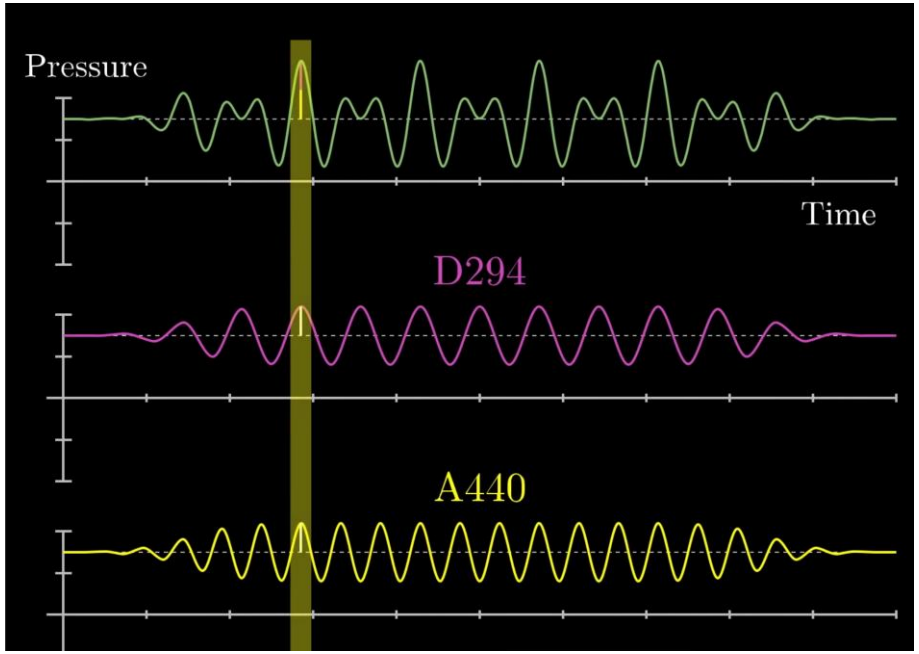
Unit 01 | 음성신호의 이해

Tone-Color (=Waveform)

파형의 모양 = 소리의 음색

모양은 다 다른거 아니야? 무슨 규칙이라도 있나??

소리신호 = 규칙적(주기) 신호들의 합



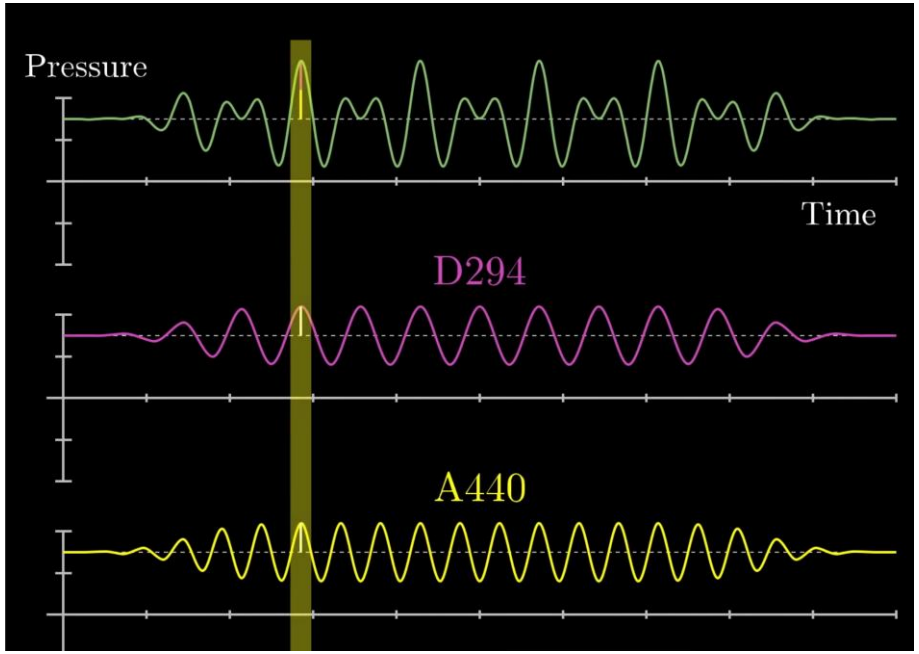
Unit 01 | 음성신호의 이해

Tone-Color (=Waveform)

파형의 모양 = 소리의 음색

모양은 다 다른거 아니야? 무슨 규칙이라도 있나??

소리신호 = 규칙적(주기) 신호들의 합

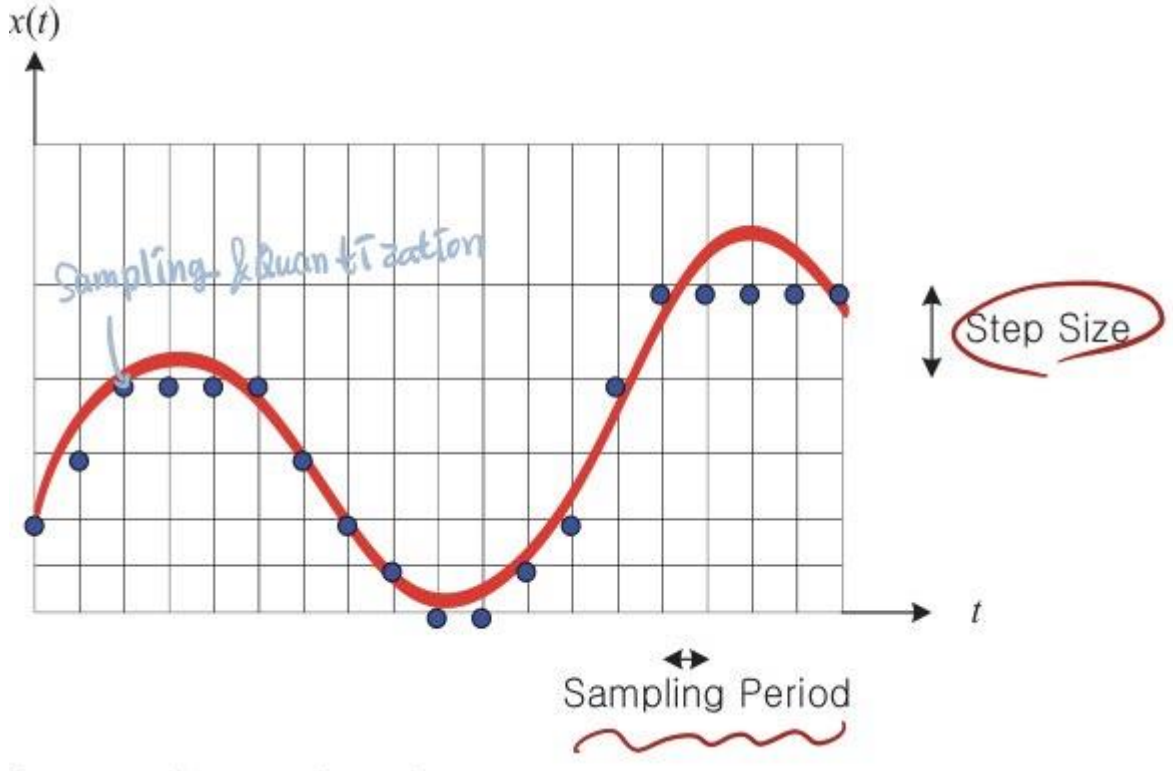


Unit 01 | 음성신호의 이해

컴퓨터가 소리를 인식하는 방법 (Analog to Digital)

Sampling : 연속적인 signa을
몇 개의 신호만 뽑아서 표현하자!
Sampling rate = 1초당 sampling 횟수

Quantization : Amplitude를
real value level에 맞게 조정하자!
(spectogram으로 바로 보기 때문에 Quantization은 거의 사용하지 않음. Light한 model에선 필요하다)



Unit 01 | 음성신호의 이해

Sampling rate

Sampling rate 우리가 오또케 설정해??8ㅁ8

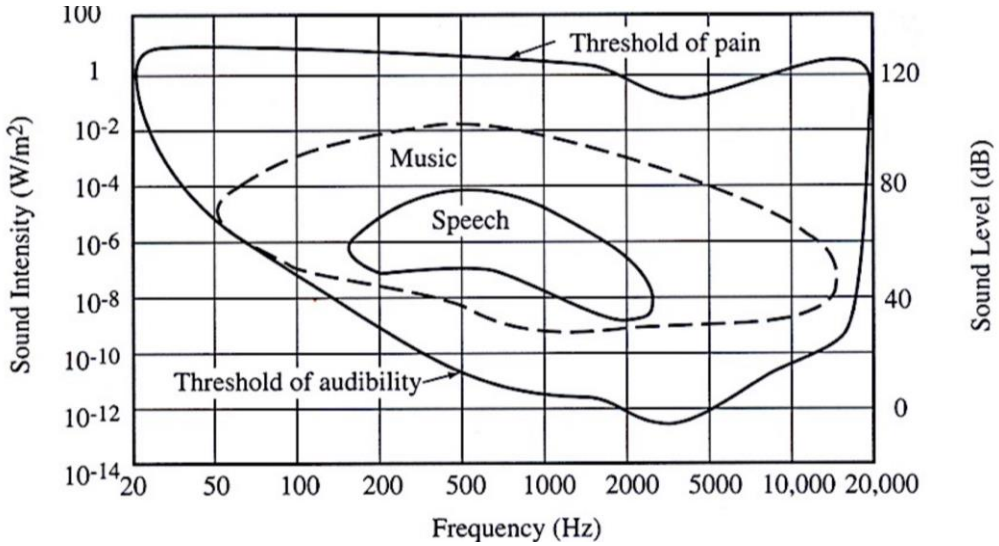
Unit 01 | 음성신호의 이해

Sampling rate

Sampling rate 우리가 오또케 설정해??8ㅁ8

-> $f_s \geq 2f_{\max}$: Nyquist Sampling Theory

일반적으로 16KHz(Speech), 22.05KHz, 44.1KHz(Music)



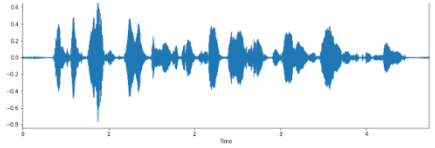
Down sampling/ Up sampling등
더 많은 sampling이론은 딥러닝에 쓰이는
Data(spectrogram)에서는 조금 거리가 있는
이야기여서 궁금한 사람은 따로 카톡!

02 Time-to-Frequency

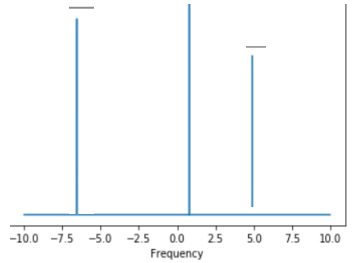
이제 당신은 음성 신호 고수!
마스터가 되볼까나??

Unit 02 | Time-to-Frequency

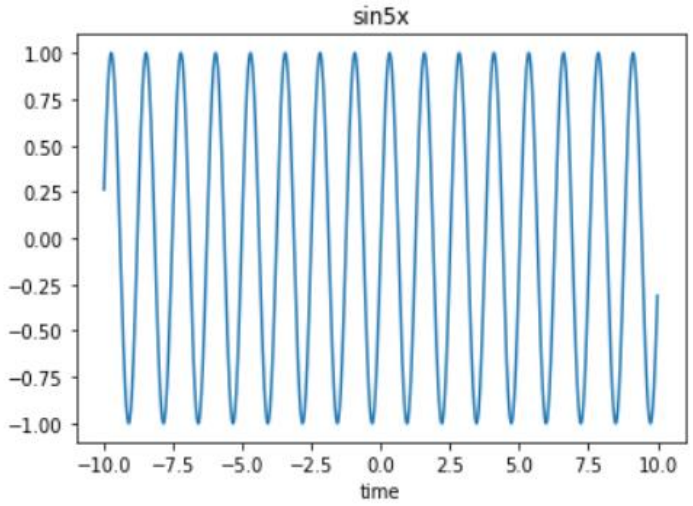
푸리에 변환 (Fourier Transform)

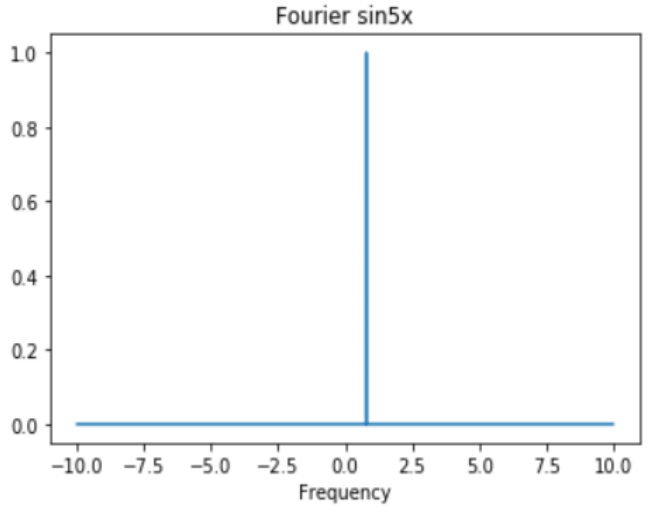


Time domain -> Frequency domain



소리 = 주기 신호의 합 = 주기?아하!주파수 역수??time domain에서 주기의 합이라구?? 그럼 Frequency domain에서는 seperate하게 보여지겠네?? = 응! 맞아!





Unit 02 | Time-to-Frequency

푸리에 변환 (Fourier Transform)

Time domain -> Frequency domain

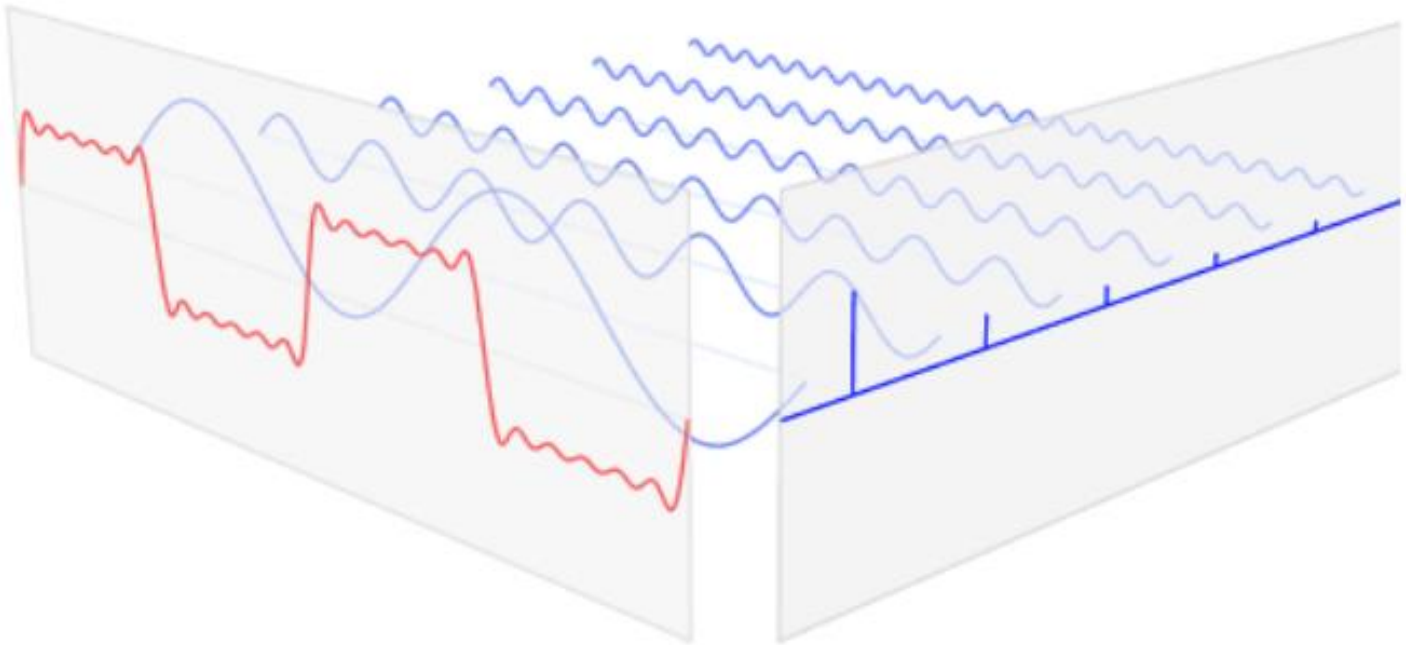


그림 1. 푸리에 변환 (그림출처: 위키피디아)

Unit 02 | Time-to-Frequency

푸리에 변환 (Fourier Transform)

Fourier Transform의 종류

- Continuous-Time Fourier Transform(CTFT)
- Discrete-Time Fourier Transform(DTFT)
- Discrete Fourier Transform(DFT)

Unit 02 | Time-to-Frequency

푸리에 변환 (Fourier Transform) - CTFT

- **Continuous-Time Fourier Transform(CTFT)**

basis signal = $e^{j2\pi f_0 t}$

correlation = $\int_{-\infty}^{\infty} x(t)y^*(t)dt$

$$x(t) \Leftrightarrow X(f)$$

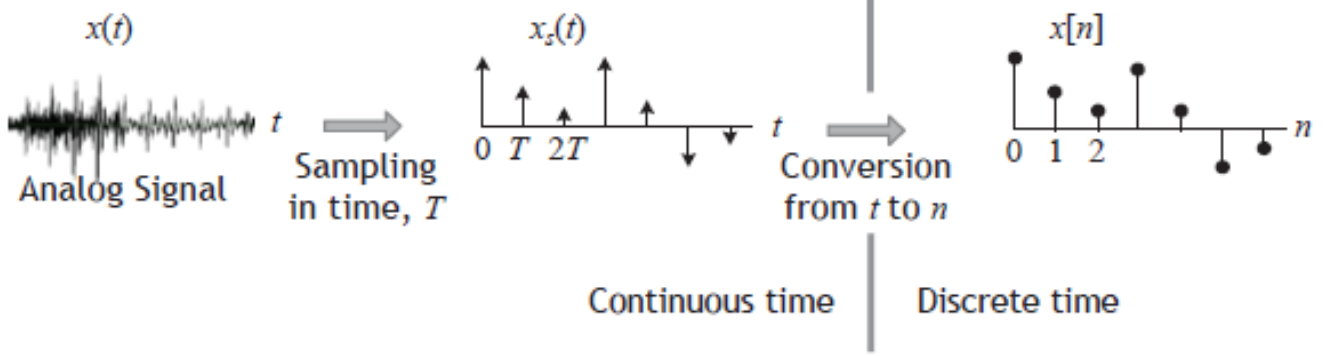
t : time in sec
f : frequency in Hz

$$X(f) = \int_{-\infty}^{\infty} x(t)e^{-j2\pi ft}dt$$

correlation between $x(t)$ and basis frequency signal $e^{j2\pi ft}$

$$x(t) = \int_{-\infty}^{\infty} X(f)e^{j2\pi ft}df$$

- **Conversion from $x(t)$ to $x[n]$**



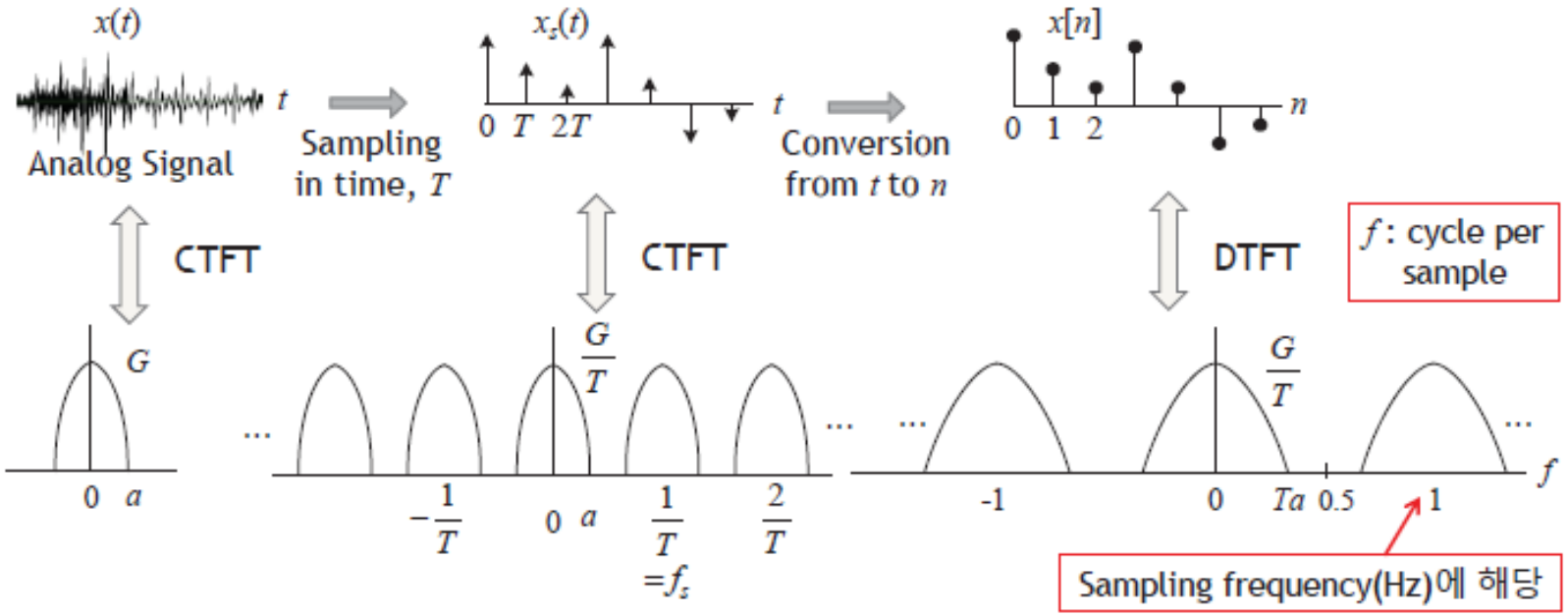
Unit 02 | Time-to-Frequency

푸리에 변환 (Fourier Transform) - DTFT

- **Discrete-Time Fourier Transform(DTFT)**
  - DT(discrete-time) waveform : $x[n]$
  - CF (continuous-frequency) spectrum : $X(f)$

$$X(f) = \sum_{n=-\infty}^{\infty} x[n] e^{-j2\pi fn}$$

$$x[n] = \int_{-0.5}^{0.5} X(f) e^{j2\pi fn} df$$

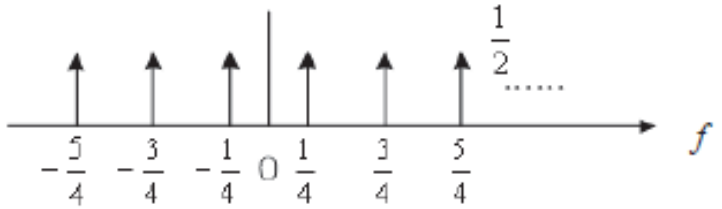


Unit 02 | Time-to-Frequency

푸리에 변환 (Fourier Transform) - DTFT

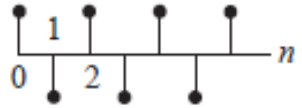
- DTFT spectrum 성질
  - f = 1.0 마다 반복
    - $-0.5 < f < 0.5$ 가 모든 spectrum 정보를 포함
  - Cosine 신호 성질

$$\cos(2\pi\frac{9}{4}n) = \cos(2\pi\frac{5}{4}n) = \cos(2\pi\frac{1}{4}n)$$

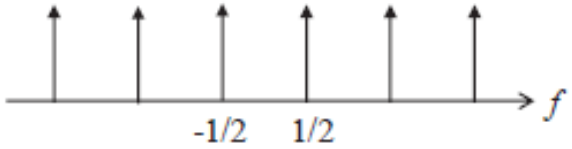


  - 최고 주파수 = 0.5

Signal with the highest Frequency



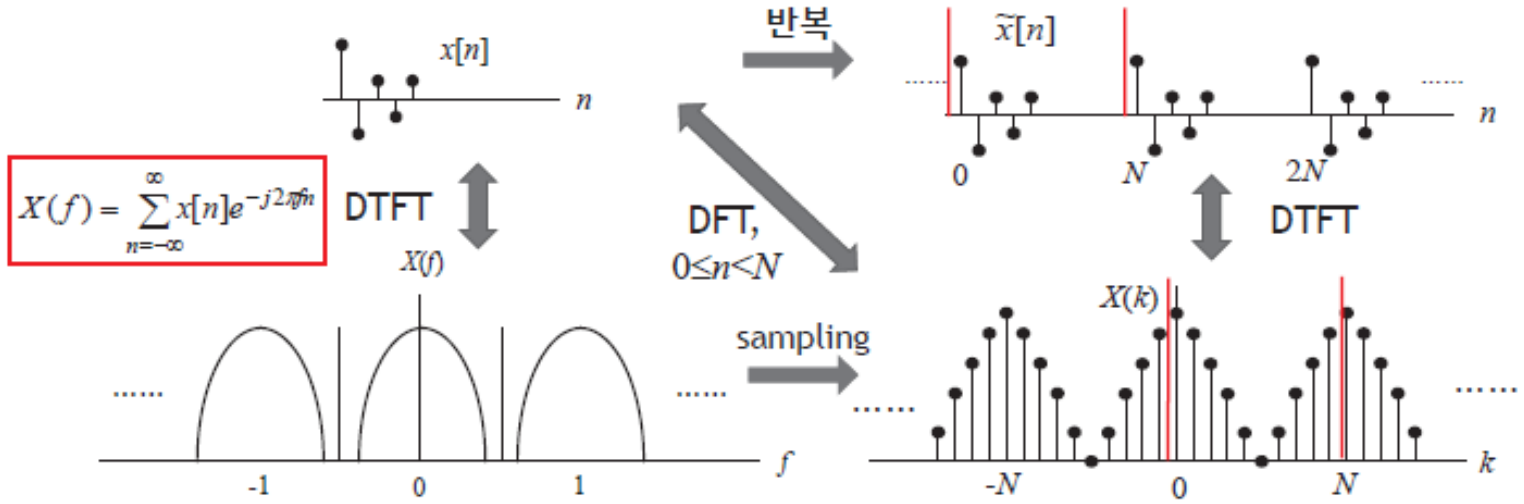
- 두 domain를 효율적으로 활용하는 능력 필요!



Unit 02 | Time-to-Frequency

푸리에 변환 (Fourier Transform) - DFT

- Digital 처리를 위하여 time와 frequency domain에서 모두 sampling
  - 그에 따라 time/frequency domain이 모두 주기 신호가 되어야 함



$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn}$$

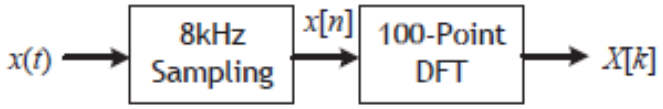
$$x[n] = \frac{1}{N} \sum_{n=0}^{N-1} X[k] e^{j\frac{2\pi}{N}kn}, 0 \le n < N$$

Unit 02 | Time-to-Frequency

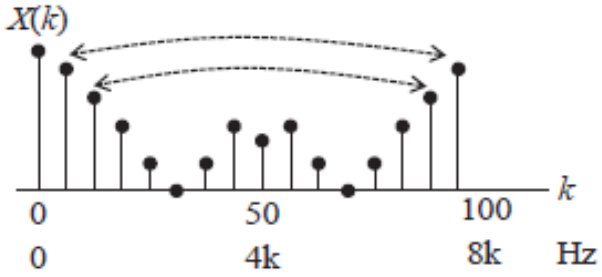
푸리에 변환 (Fourier Transform) - DFT

- N-point DFT
  - Time domain : N time samples
  - Frequency domain : N spectral coefficients
  - $X[k]$ 에서 $k = N$ 이 f = 1.0에 해당함
  - N 이 주파수 resolution 결정

- Example



  - Spectral resolution = 8k/100 = 80Hz
  - 200Hz low-pass filtering



Unit 02 | Time-to-Frequency

푸리에 변환 (Fourier Transform)

이걸 다 알라구...?8ㅁ8

언제 뭘 쓰는건데...??

도와줘 쿼카 ㅠㅠㅠㅠ

Unit 02 | Time-to-Frequency

푸리에 변환 (Fourier Transform)

이걸 다 알라구...?8ㅁ8

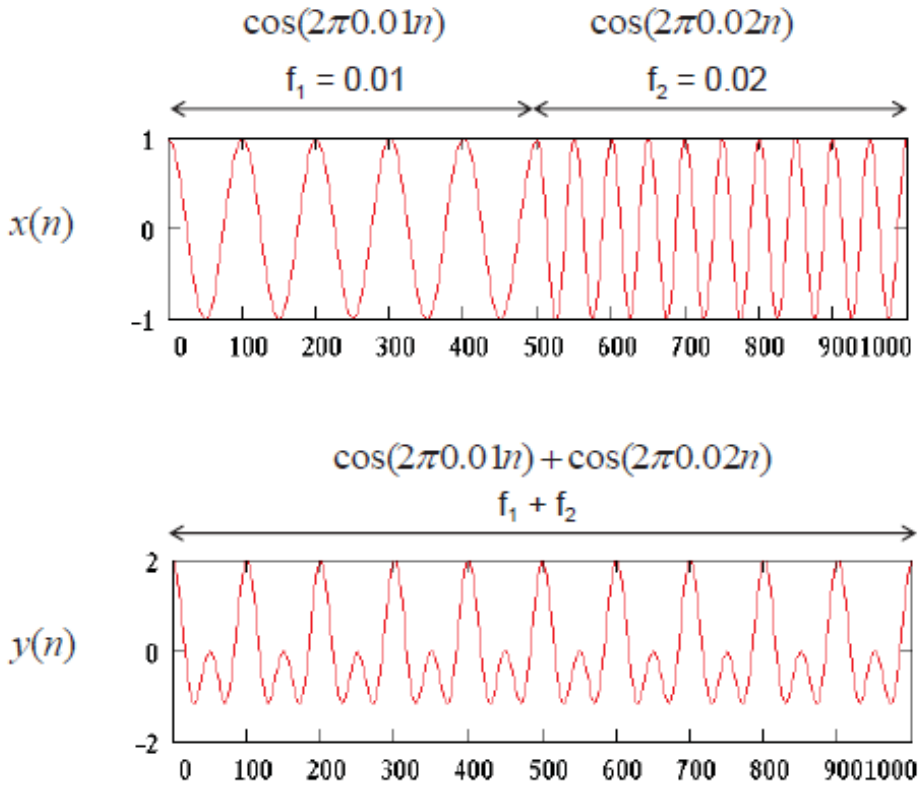
언제 뭘 쓰는건데...??

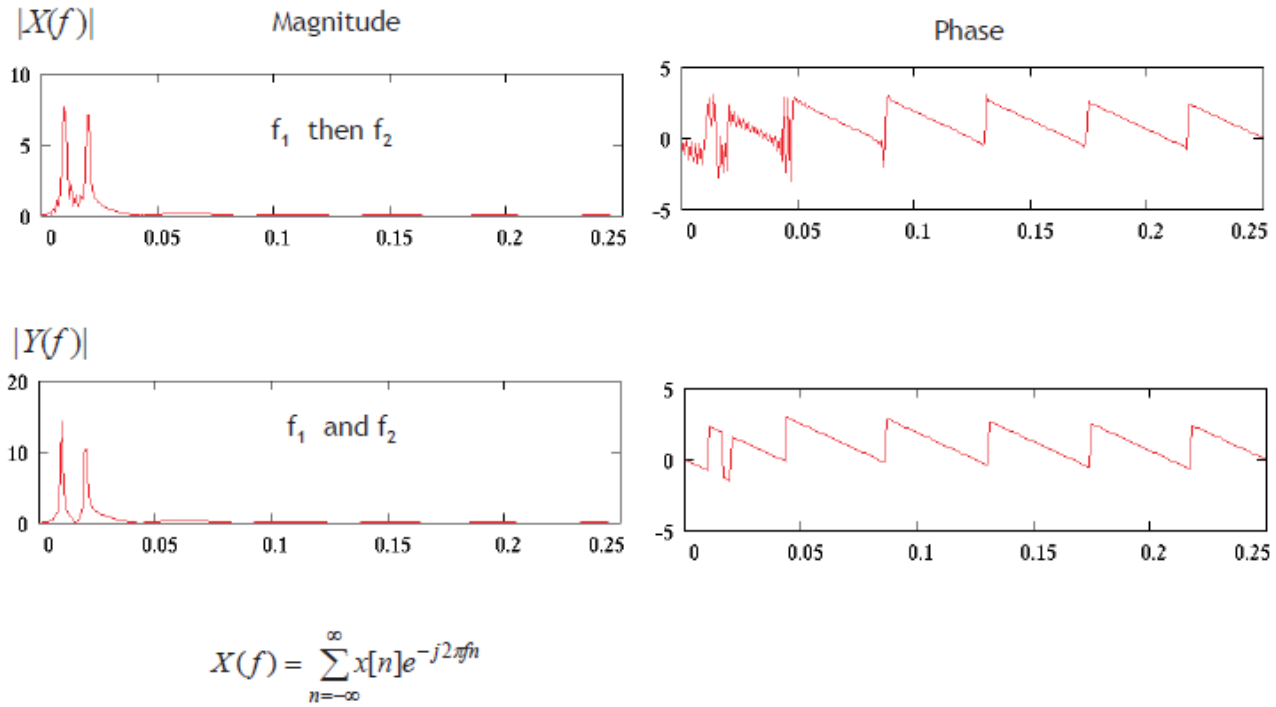
도와줘 쿼카 ㅠㅠㅠㅠ

Unit 02 | Time-to-Frequency

Shor-Time Fourier Transform (STFT)

그냥FT의 문제점



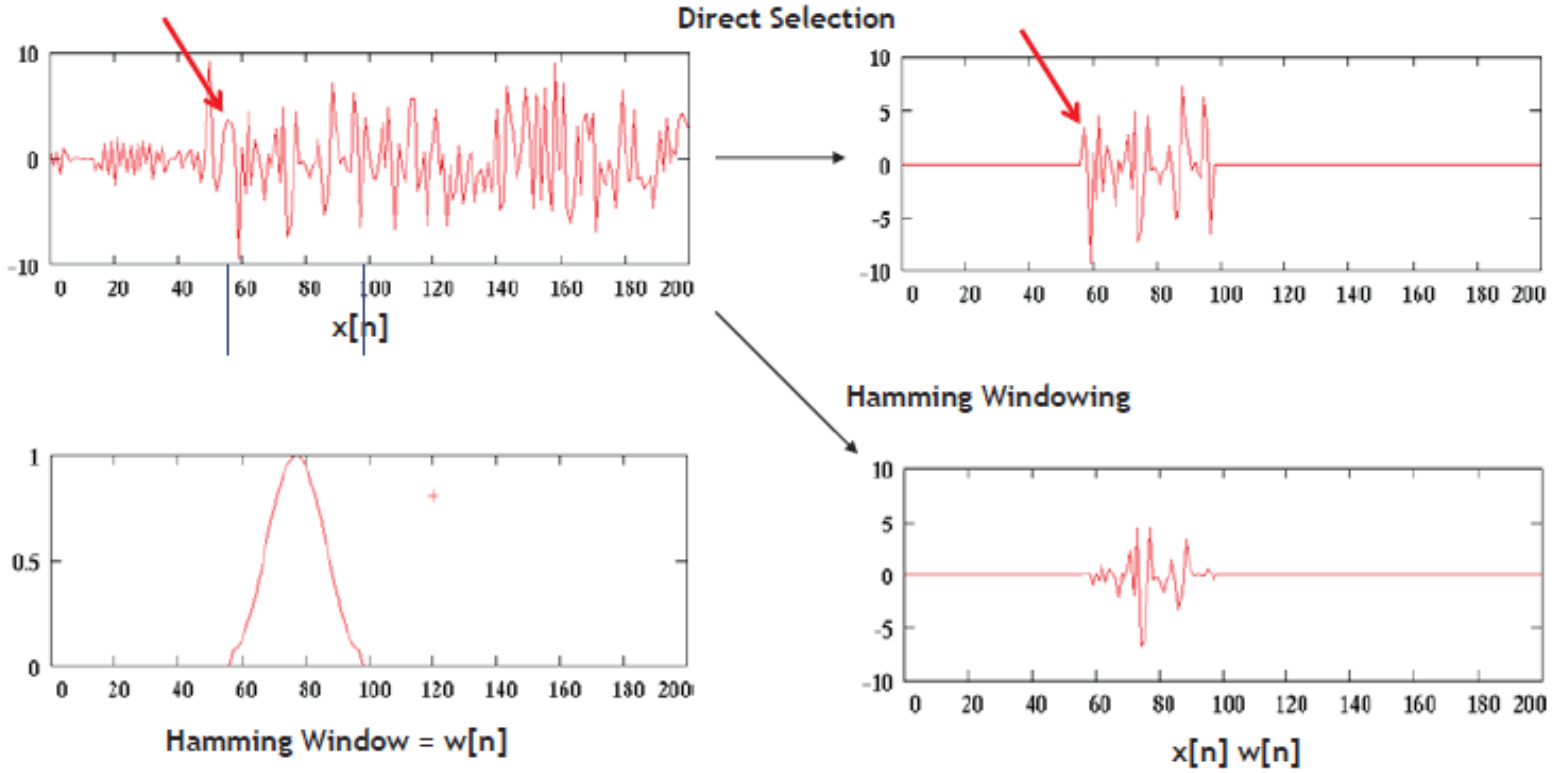


$$X(f) = \sum_{n=-\infty}^{\infty} x[n] e^{-j2\pi fn}$$

Unit 02 | Time-to-Frequency

Shor-Time Fourier Transform (STFT)

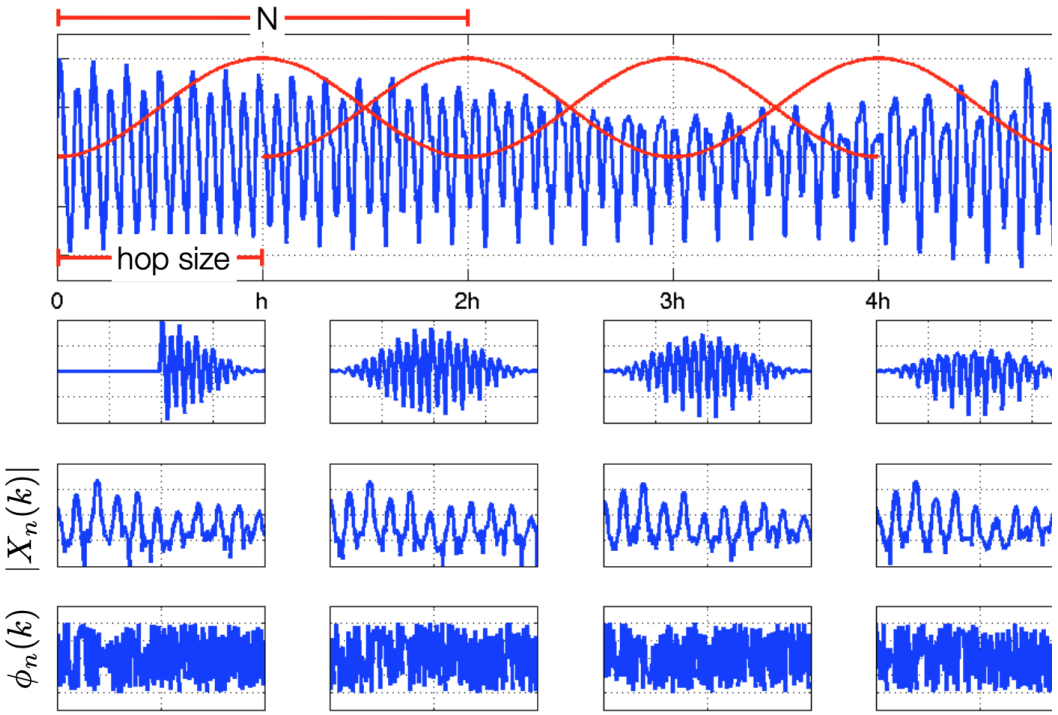
STFT : 짧은 시간 영역에서 한정된 FT 실행



Unit 02 | Time-to-Frequency

Shor-Time Fourier Transform (STFT)

$$X(l,k) = \sum_{n=0}^{N-1} w(n)x(n+lH)\exp^{\frac{-2\pi kn}{N}}$$



- N : FFT size
  - Window를 얼마나 많은 주파수 밴드로 나누는가 입니다.
- Duration
  - 샘플링 레이트를 window로 나눈 값입니다.
  - $T = window/SR$
  - T(Window) = 5T(Signal), duration은 신호주기보다 5배 이상 길게 잡아야한다.
  - 440Hz 신호의 window size는 5*(1/440)이 됩니다.
- $w(n)$: Window function
  - 일반적으로 Hann window가 쓰입니다.
- n : Window size
  - Window 함수에 들어가는 Sample의 양입니다.
  - 작을수록 Low-frequency resolution을 가지게 되고, high-time resolution을 가집니다.
  - 길수록 High-frequency, low time resolution을 가집니다.
- H : Hop size
  - 윈도우가 겹치는 사이즈입니다. 일반적으로는 1/4정도를 겹치게 합니다.

STFT의 결과는 즉 시간의 흐름(Window)에 따른 Frequency영역별 Amplitude를 반환합니다.

Unit 02 | Time-to-Frequency

Shor-Time Fourier Transform (STFT)

이 식 실화야...? $$X(l,k) = \sum_{n=0}^{N-1} w(n)x(n+lH)\exp\frac{-2\pi kn}{N}$$

이걸 code로 구현해서

몇십기가 data에 다 적용하라고...??8ㅁ8

쿼카야 도와줘 ㅠㅠㅠㅠㅠ

Unit 02 | Time-to-Frequency

Shor-Time Fourier Transform (STFT)

이 식 실화야...? $X(l,k) = \sum_{n=0}^{N-1} w(n)x(n+lH)\,\mathrm{ex}$

이걸 code로 구현해서

몇십기가 data에 다 적용하라고...??8ㅁ8

쿼카야 도와줘 ㅠㅠㅠㅠㅠ

Unit 02 | Time-to-Frequency

Shor-Time Fourier Transform (STFT)

```
# STFT
S = librosa.core.stft(samples, n_fft=1024, hop_length=512, win_length=1024)
```

```
S.shape, len(S[0]), S[0][0]
```

```
((513, 44), 44, (-0.2504628+0j))
```

```
sample_rate/512
```

```
43.06640625
```

```
# phase 에 대한 정보를 날린다.
D = np.abs(S)**2
```

```
D.shape
```

```
(513, 44)
```

Unit 02 | Time-to-Frequency

Shor-Time Fourier Transform (STFT)

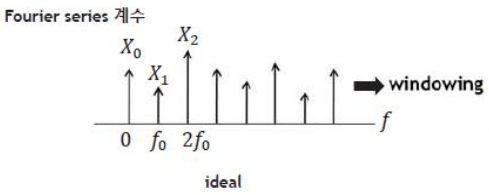
오호! STFT로 time을 고려한 frequenc의 특성을 볼 수 있구나!

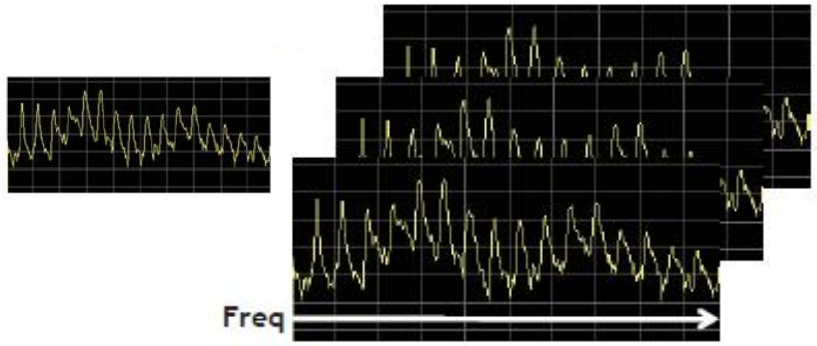
근데 이거 왜하고 있는거지...?

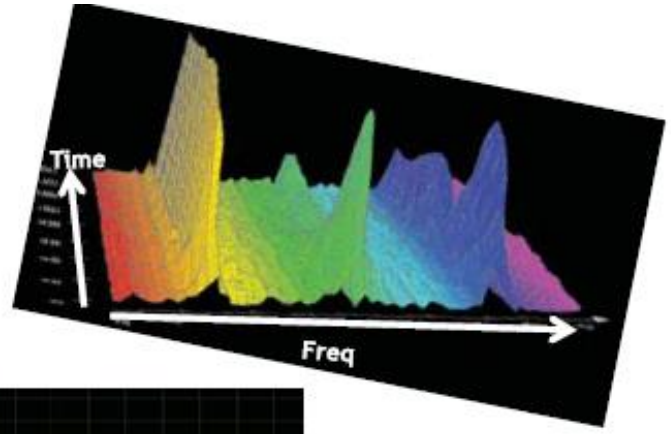
03 Spectogram

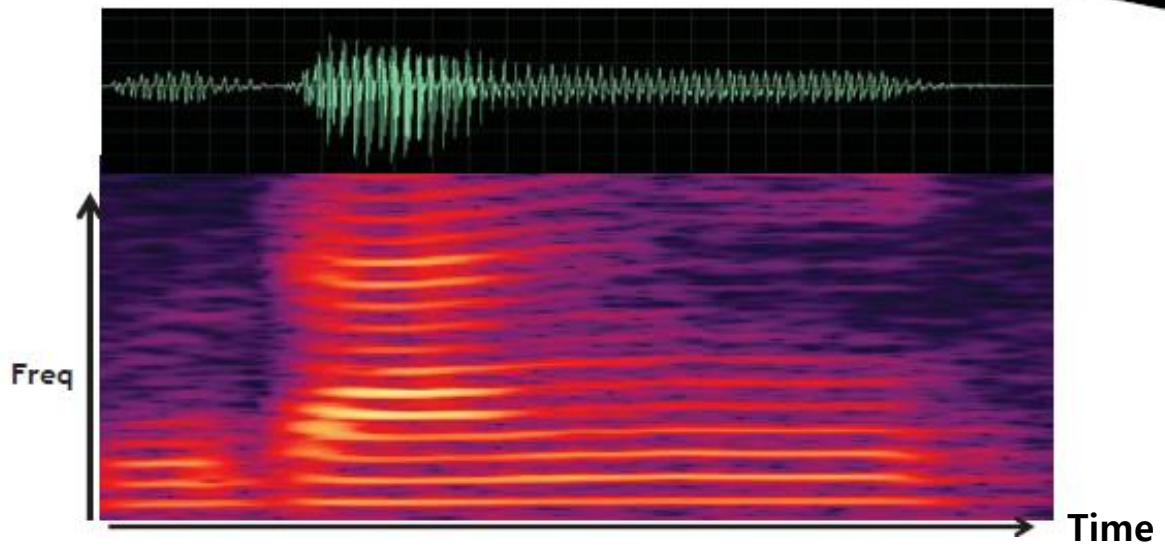
Unit 03 | Spectogram

Spectogram STFT를 배운 이유 -> spectogram을 그리기 위하여!!









Unit 03 | Spectogram

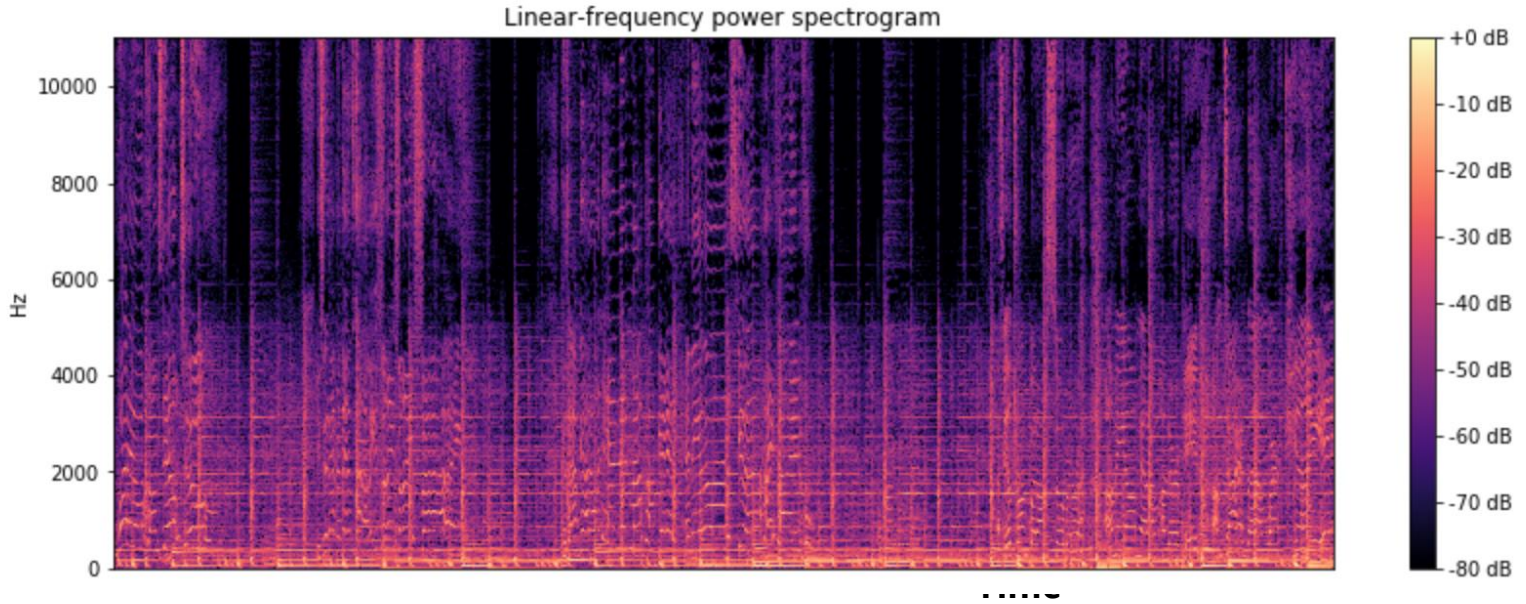
Spectogram

```
D = librosa.amplitude_to_db(np.abs(librosa.stft(y)), ref=np.max)
D[0:1] ,D.shape
```

```
(array([[-57.238033, -68.030106, -58.63585 , ..., -60.651947, -35.656086,
         -29.75666 ]], dtype=float32), (1025, 1293))
```

```
fig = plt.figure(figsize = (14,5))
librosa.display.specshow(D, y_axis='linear')
plt.colorbar(format='%+2.0f dB')
plt.title('Linear-frequency power spectrogram')
```

```
Text(0.5, 1.0, 'Linear-frequency power spectrogram')
```



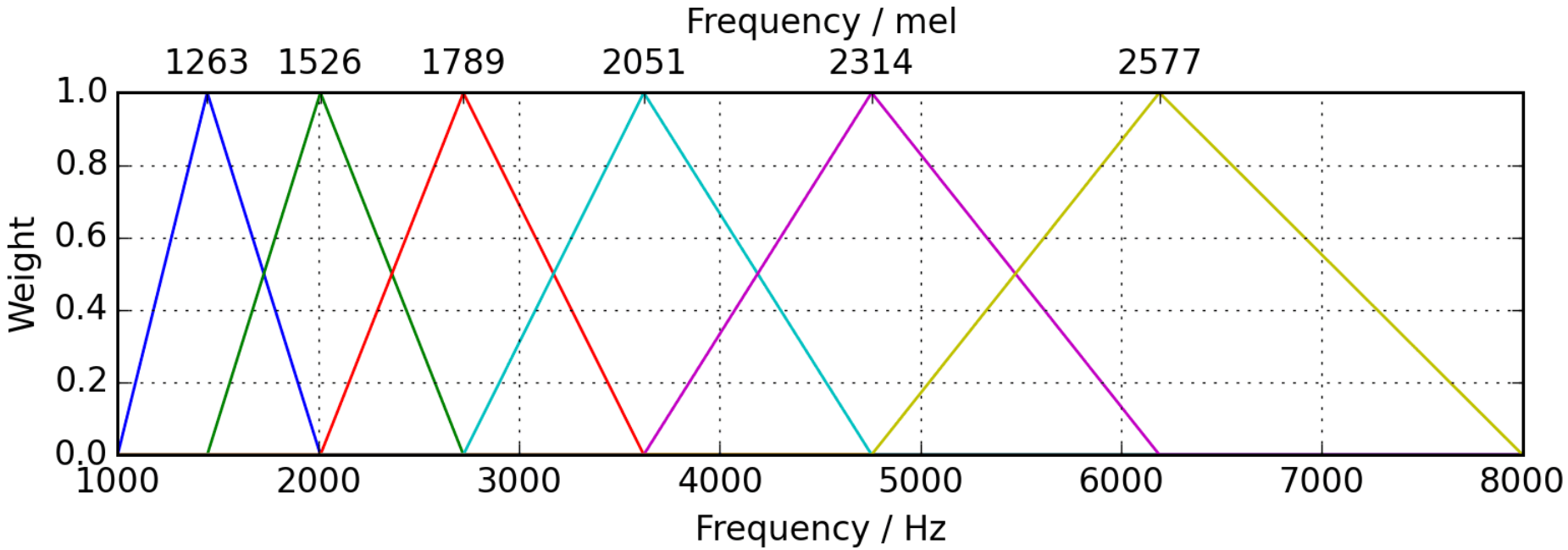
Unit 03 | Spectogram

Mel spectogram

Mel filter bank

사람은 인접한 주파수를 크게 구별하지 못한다!

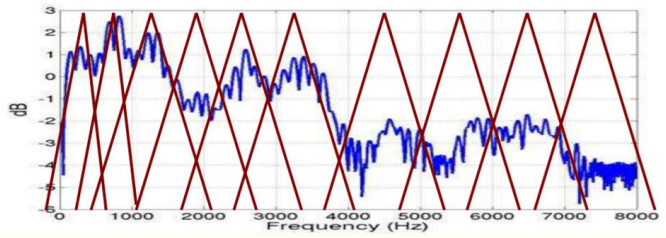
우리의 인지기관이 categorical한 구분을 하기 때문!
멜 스펙트럼은 주파수 단위를 다음 공식에 따라 멜 단위로 바꾼 것을 의미
이 멜 필터는 **저주파** 주변에서 얼마만큼 에너지가 있는지를 알려줍니다.

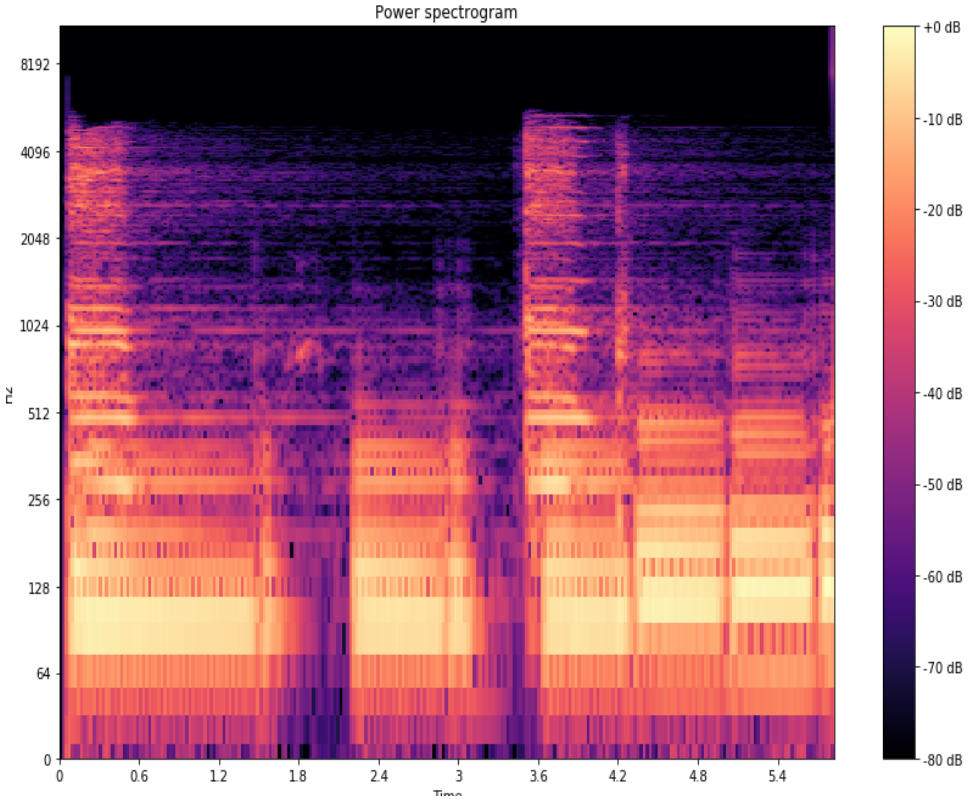


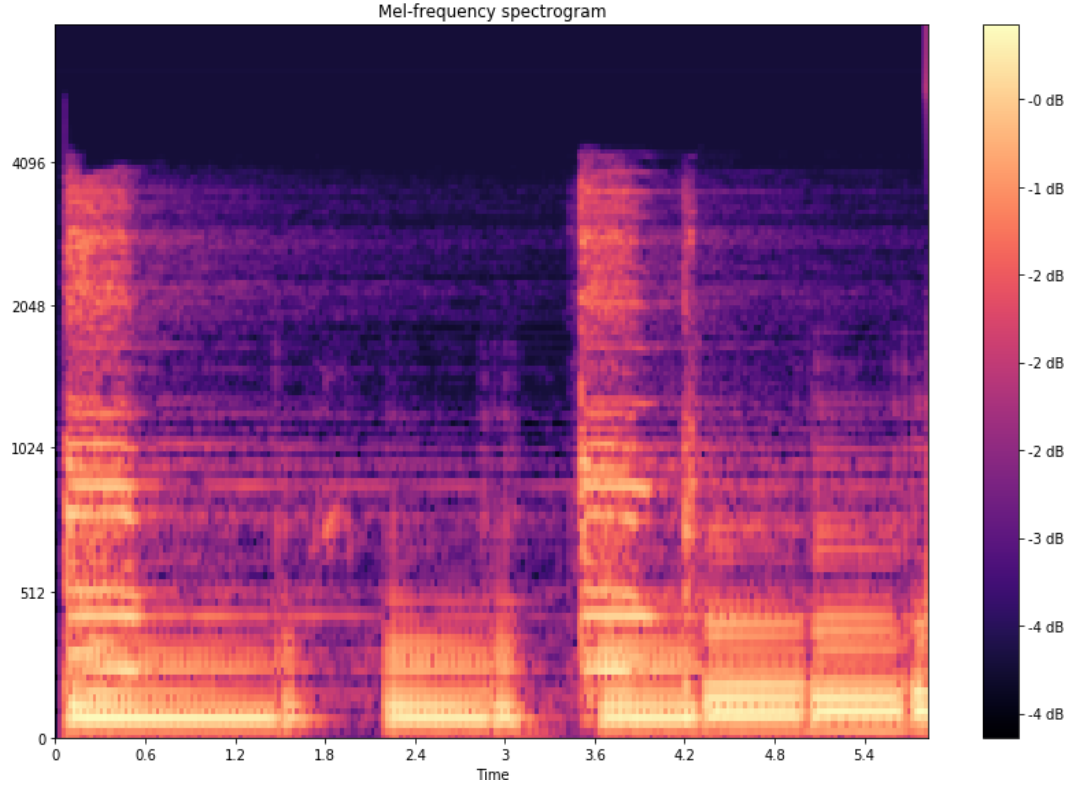
Time

Unit 03 | Spectogram

Mel spectogram







Unit 03 | Spectogram

Mel spectogram

```
# STFT
S = librosa.core.stft(samples, n_fft=1024, hop_length=512, win_length=1024)
# phase 에 대한 정보를 날린다.
D = np.abs(S)**2
```

```
# mel spectrogram (512 --> 40)
mel_basis = librosa.filters.mel(sample_rate, 1024, n_mels=40)
mel_S = np.dot(mel_basis, D)
mel_S.shape
```

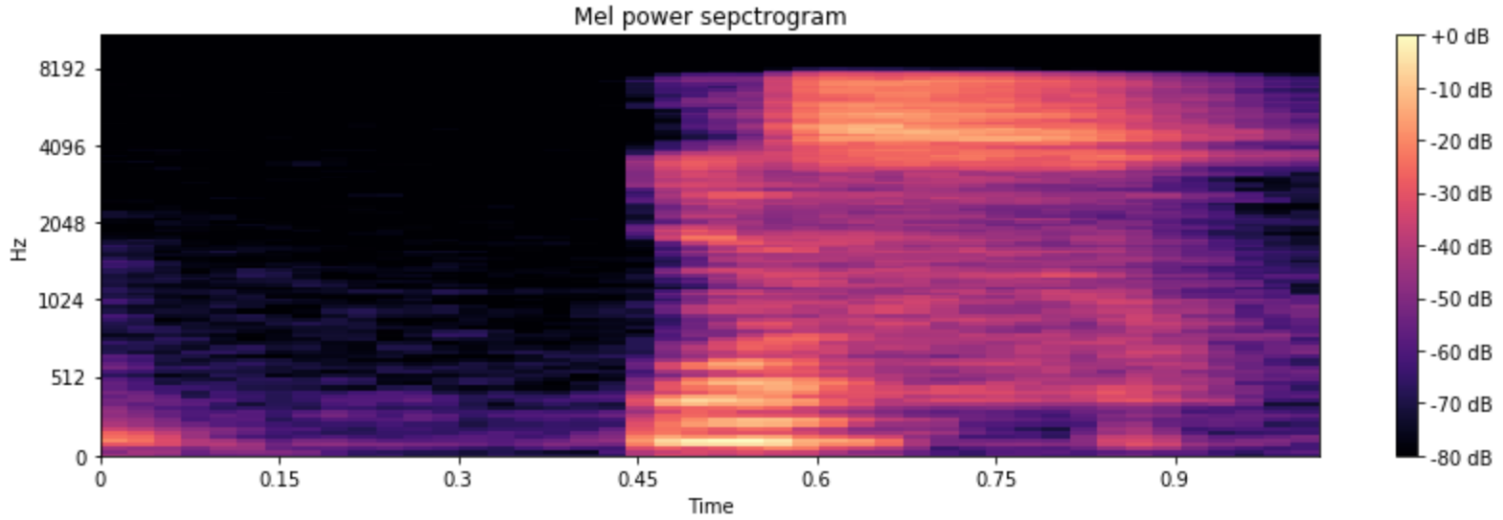
```
(40, 44)
```

Time

Unit 03 | Spectogram

Mel spectogram

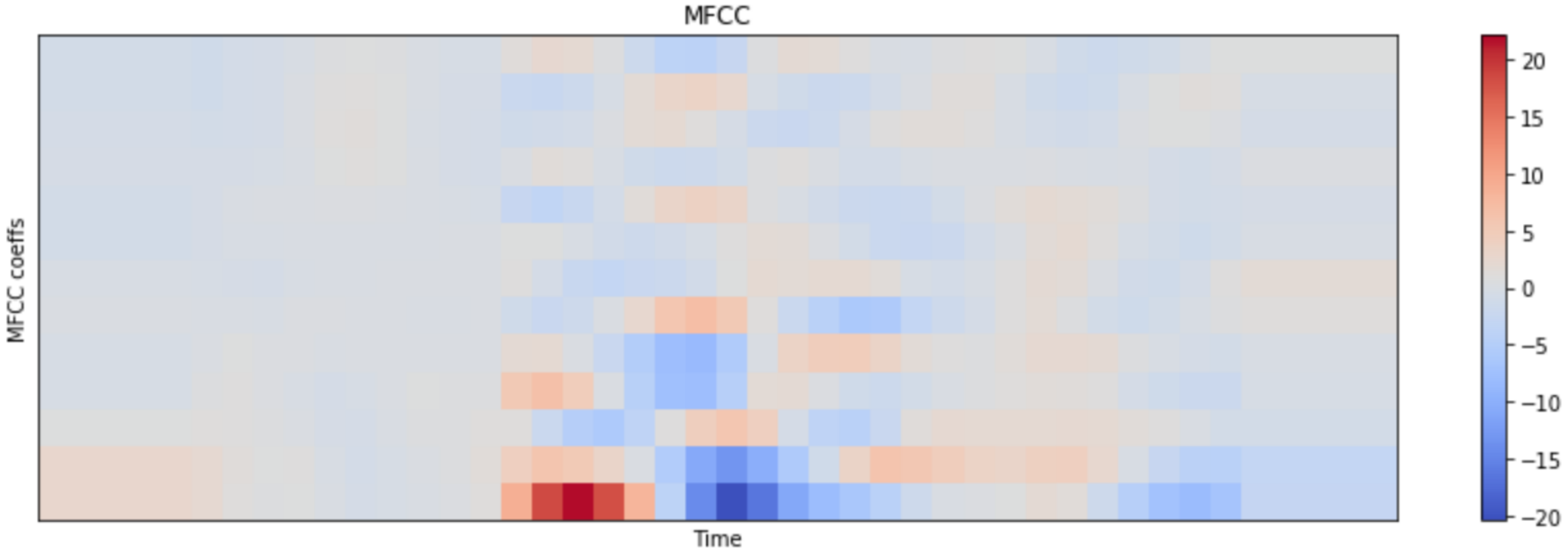
```
import librosa.display

S = librosa.feature.melspectrogram(samples, sr=sample_rate, n_mels = 128)
log_S = librosa.power_to_db(S, ref=np.max)
plt.figure(figsize=(12,4))
librosa.display.specshow(log_S, sr=sample_rate, x_axis='time', y_axis='mel')
plt.title('Mel power sepctrogram')
plt.colorbar(format='%+02.0f dB')
plt.tight_layout()
```



Unit 03 | Spectogram

MFCC

1. Mel-spectogram에 Log적용
2. Mel-log-spectrogram list전체에 DCT(Descrete Cosine Transform)적용
3. 얻어진 Coefficient에서 앞에서부터 N개만 추출



Unit 03 | Spectogram

MFCC

```
# mfcc (DCT)
mfcc = librosa.feature.mfcc(S=log_mel_S, n_mfcc=13)
mfcc = mfcc.astype(np.float32)    # to save the memory (64 to 32 bits)
mfcc[0]
```

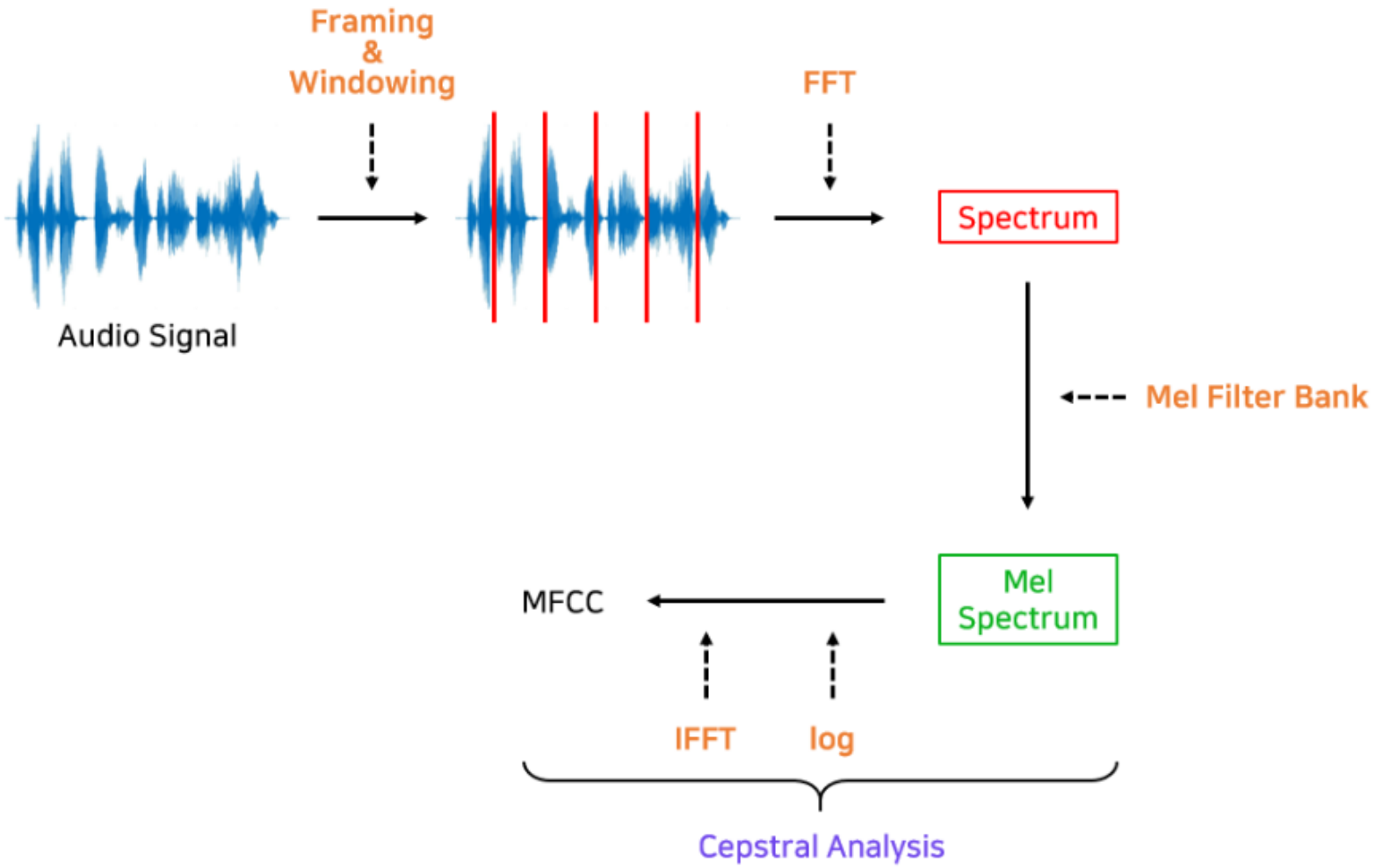
```
array([-352.24487, -366.56326, -374.9281 , -390.60297, -385.9468 ,
       -392.3261 , -398.55487, -396.6922 , -395.0417 , -398.3241 ,
       -400.8156 , -399.5283 , -400.10876, -405.3044 , -407.25635,
       -410.77112, -409.39105, -405.62488, -400.28088, -386.88754,
       -256.587  , -184.99794, -167.65366, -155.98474, -151.11472,
       -136.15807, -139.98293, -145.12239, -152.257  , -160.66386,
       -164.44904, -166.04997, -172.92793, -178.01353, -180.90056,
       -187.92828, -179.37764, -190.39357, -217.45445, -235.33223,
       -262.61517, -282.51657, -303.91086, -313.25308], dtype=float32)
```

```
mfcc = librosa.feature.mfcc(S=log_S, n_mfcc=13)
delta2_mfcc = librosa.feature.delta(mfcc, order=2)
print(delta2_mfcc.shape)

plt.figure(figsize=(12,4))
librosa.display.specshow(delta2_mfcc)
plt.ylabel('MFCC coeffs')
plt.xlabel('Time')
plt.title('MFCC')
plt.colorbar()
plt.tight_layout()
```

```
(13, 44)
```

Unit 03 | Spectogram

정리



04 Data Argumentation

파이팅!!!끝이 보인당!

Unit 04 | Data Argumentation

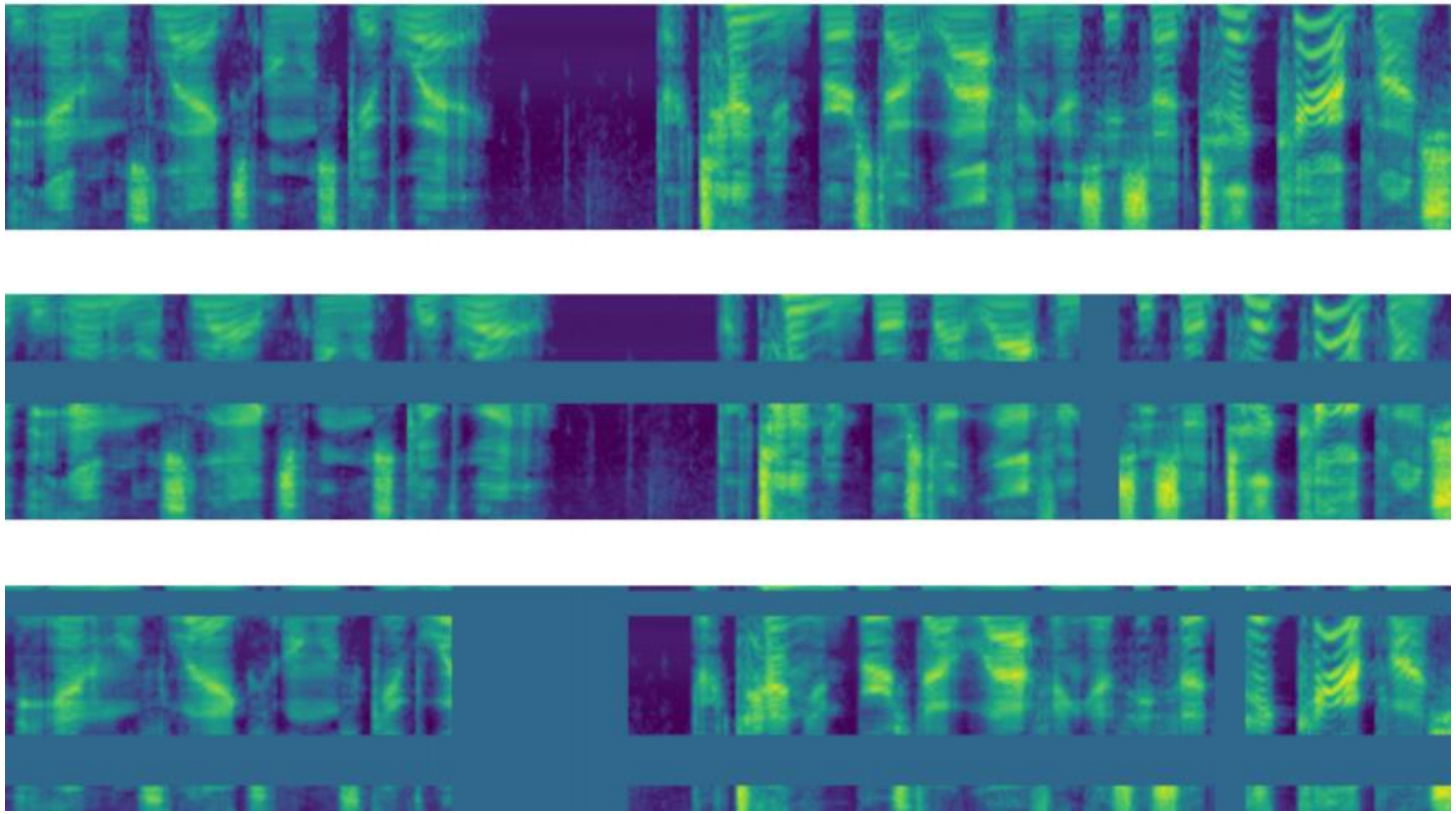
Data Argumentation (Audio)

- Noise Injection
- Shifiting Time
- Changing Pitch
- Changing Speed
- Stretch
- 등등등등…

Unit 04 | Data Argumentation

Data Argumentation (spectrogram as image)

SOTA = Masking



05 실습코드

Reference

참고자료

- 광운대 전자공학과 박호종교수님 강의자료
- 투빅스 12기 박진혁님 강의
- T아카데미 토크ON세미나 74회 '오디오 딥러닝 입문자를 위한 디지털신호처리 이해' 도승헌님

- https://brightwon.tistory.com/11 (MFCC)
- https://medium.com/@makcedward/data-augmentation-for-audio-76912b01fdf6 (data argumentation-audio)
- https://pdfs.semanticscholar.org/fd4d/7b6b37f8ffa2ee73af11675b8cec1e659bda.pdf (data argumentation-audio)

+ 민정쓰 머리

들어주셔서 감사합니다.