Analisi Audio Tempo/Frequenza

Elaborazione dell'audio digitale

Ingegneria del Cinema, Informatica e Telecomunicazioni

Antonio Servetti servetti@polito.it
Internet Media Group http://media.polito.it
Dip. di Automatica ed Informatica
Politecnico di Torino

Sommario

- ANALISI A "BREVE TERMINE" (o a blocchi/frame brevi)
  - ✓ Suddivisione in frame: lunghezza , hop-size, ecc.
  - ✓ Estrazione delle feature per ogni frame
- Attività / Esempio: Onset Detection
- Analisi tempo/frequenza
  - ✓ Trasformata di Fourier a "breve termine" (STFT)
  - ✓ Lo spettrogramma
  - ✓ Librerie per analisi tempo/frequenza

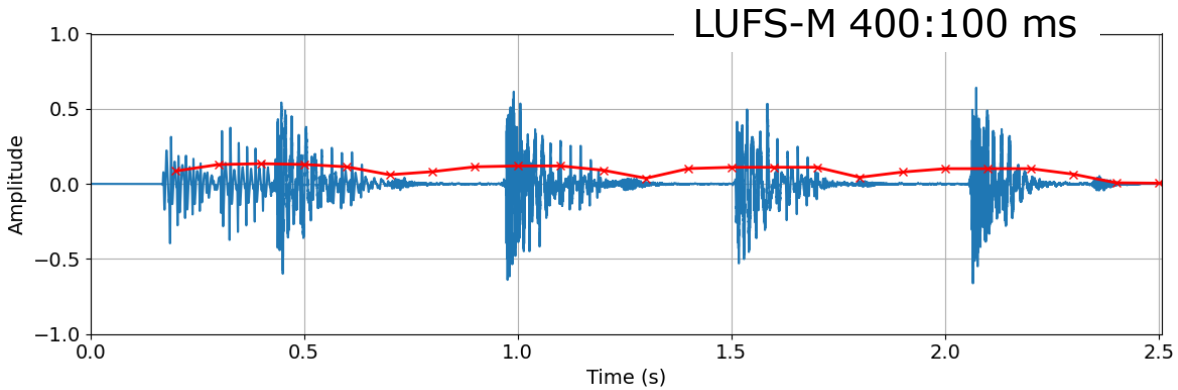
Bibliografia

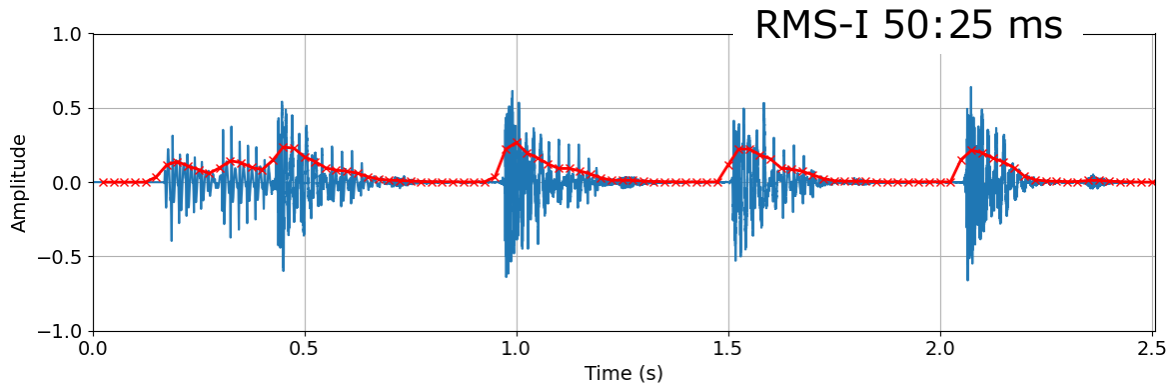
- De Poli, "Algorithms for Sound Music Computing"
  - ✓ 1.5 Short-time Fourier analysis
    - 1.5.1 The Discrete Fourier Transform
      - 1.5.1.2 Resolution, leakage and zero-padding
    - 1.5.2 The Short-Time Fourier Transform
      - 1.5.2.2 Windowing and the uncertainty principle
- Muller, "Fundamentals of Music Processing"
  - ✓ Ch.2.5 Short-Time Fourier Transform
    - 2.5.1 Definition of the STFT
    - 2.5.2 Spectrogram Representation
    - 2.5.3 Discrete version of the STFT
    - Code: https://audiolabs-erlangen.de/resources/MIR/FMP/C0/C0.html

ANALISI A "BREVE TERMINE"

Analisi a "breve termine"

- Per catturare l'andamento nel tempo delle caratteristiche del segnale audio (energia, pitch, spettro) è necessario analizzare il segnale su brevi intervalli (quasi-stazionari)
- Quanto brevi e quanto spesso?
  - ✓ La lunghezza degli intervalli dipende dalle applicazioni
    - LUFS (momentary) 400 ms ogni 100 ms
    - LUFS (short-term) 3000 ms ogni 1000 ms



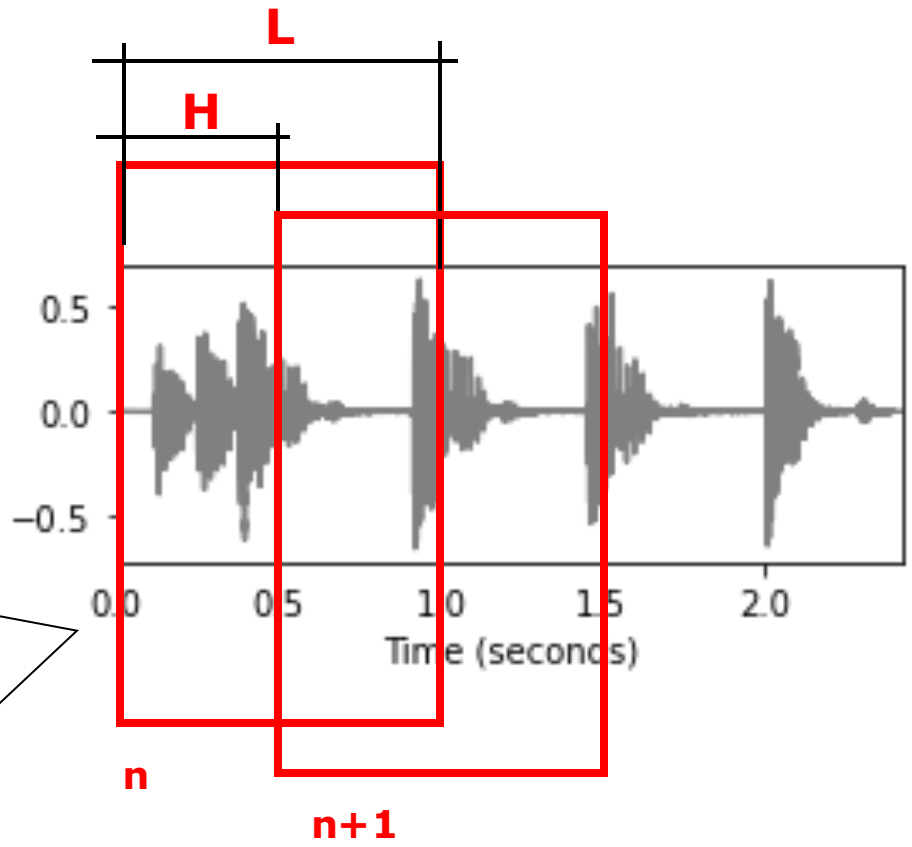


Analisi "a blocchi / finestre"

- Viene fatta "a finestre" di *breve* durata dove il segnale viene suddiviso in blocchi di L campioni parzialmente sovrapposti
- L'inizio di ogni finestra i–esima (0..M) è a distanza di H campioni (i*H)
- Overlap tipicamente 75%, 50%, 25%

L: durata, in campioni (o in ms)
H: passo, in campioni (o in ms)

n: indice della finestra / frame
$\mathbf{i_n}$: indice del campione iniziale della finestra
avanza di H (campioni) per ogni frame
$\mathbf{t_n}$: istante temporale iniziale della finestra
avanza di H/Fs (ms) per ogni frame



Analisi "a blocchi / finestre" in Python

```
# def time_framing_fn(x, L, H, win=None, fn=None):

y = pad_to_multiple(y,L) # padding with zeros last portion

i_v   = np.arange(0, len(y) - L+1, H)  # index of first sample of each frame
n_v   = np.arange(0, len(i_v))         # index of frame
fm_v = np.zeros(len(n_v))             # one value per frame

for n in n_v:
  i = i_v[n]
  frame = y[ i : i+L ]
  frame_w = frame * win # windowed frame
  fn_v[n] = fn(frame_w)
```

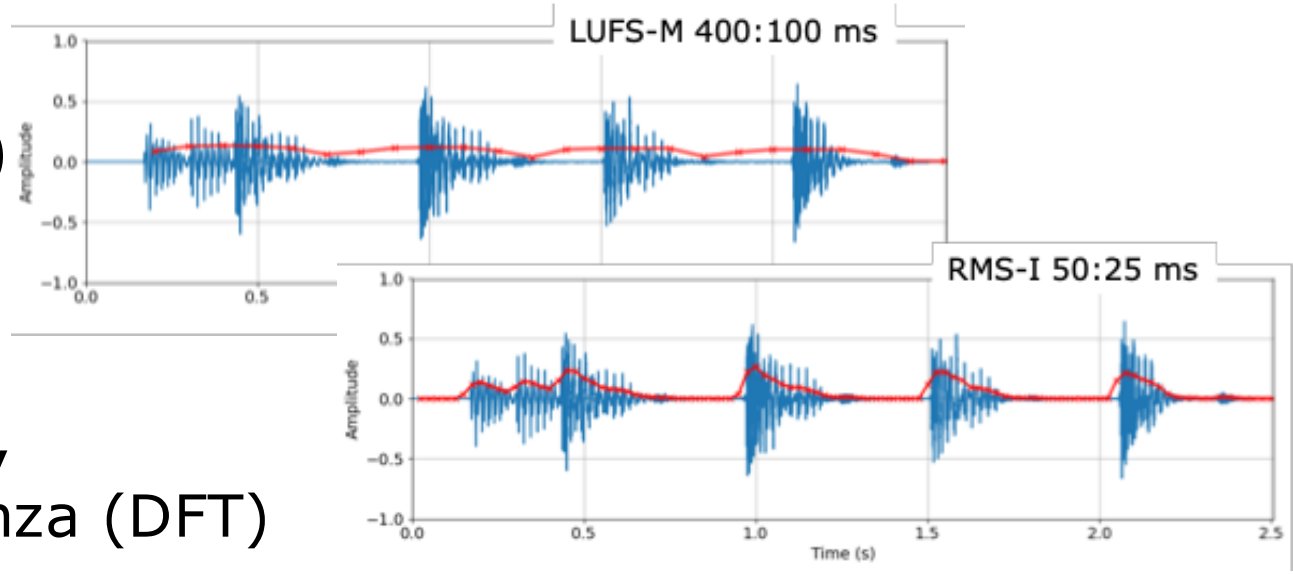
len(y) – (L+1)
ensures last frame of length L

```
e.g. L = 100
i_v = [ 0 50 100 150 200 250 .. ]
n_v = [ 0 1  2   3   4   5 .. ]

frame contains idxs [ 0 1 2 .. 99]
```

```
def pad_to_multiple(array, mult):
    length = len(array)
    target = ((length + mult - 1) // mult) * mult
    return np.pad(array, (0, target - length), mode='constant')
```

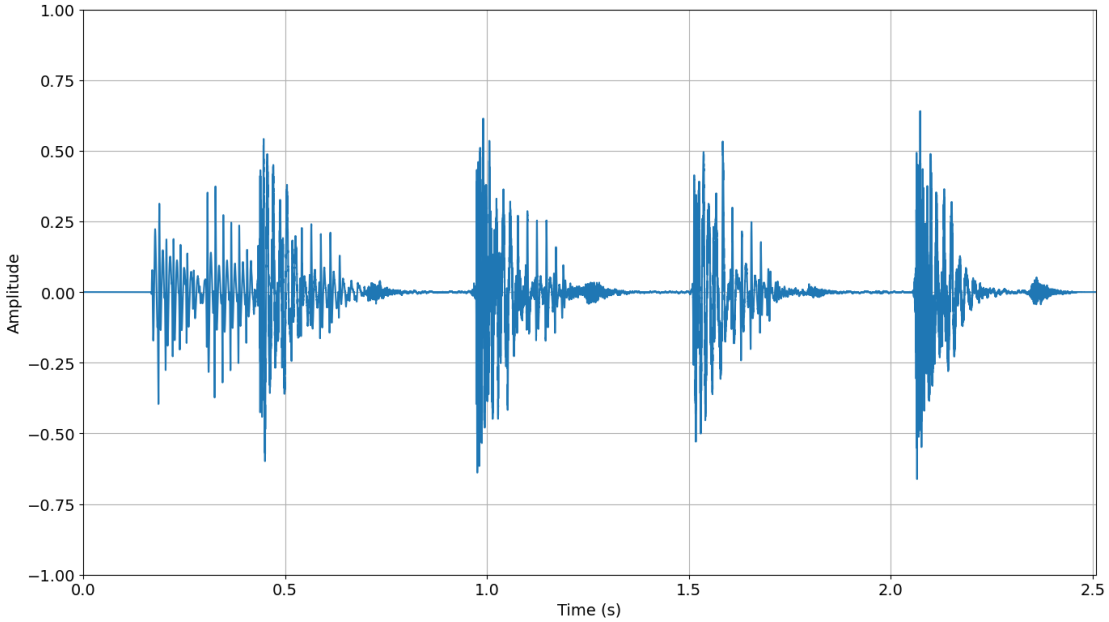
Trade-off lunghezza finestra di analisi

- La scelta della lunghezza della finestra L (e del passo H) dipende dall'applicazione
- "corta":
  - ✓ migliore risoluzione nel tempo, peggiore risoluzione in frequenza (DFT)
  - ✓ valori dall'andamento più instabile, soggetti a variazioni repentine
- "lunga":
  - ✓ rischio di non-stazionarietà nel frame singolo (fondere due fonemi)
  - ✓ non riesco a catturare cambiamenti rapidi (smussare transienti)



Onset Detection

- Consideriamo un esempio di analisi chiamato **Onset Detection**
  - ✓ L'idea è quella di identificare l'inizio di un suono, cioè un rapido cambiamento dopo un periodo di (quasi) silenzio
  - ✓ Ad esempio suoni percussivi in un brano musicale, (simile nella voce si può parlare di voice-activity-detection)



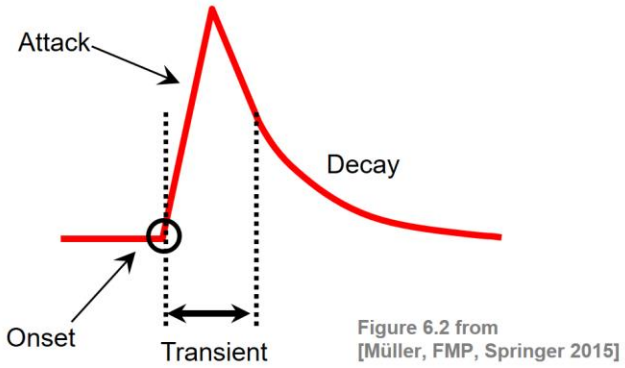
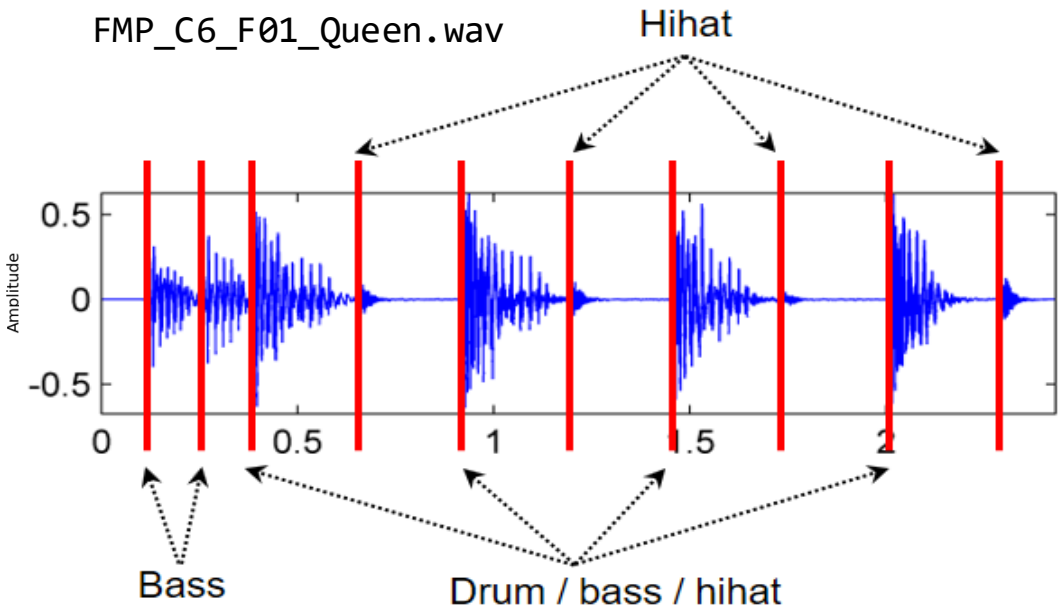


Figure 6.2 from [Müller, FMP, Springer 2015]

Onset Detection

- Consideriamo un esempio di analisi chiamato **Onset Detection**
  - ✓ L'idea è quella di identificare l'inizio di un suono, cioè un rapido cambiamento dopo un periodo di (quasi) silenzio
  - ✓ Ad esempio suoni percussivi in un brano musicale, (simile nella voce si può parlare di voice-activity-detection)



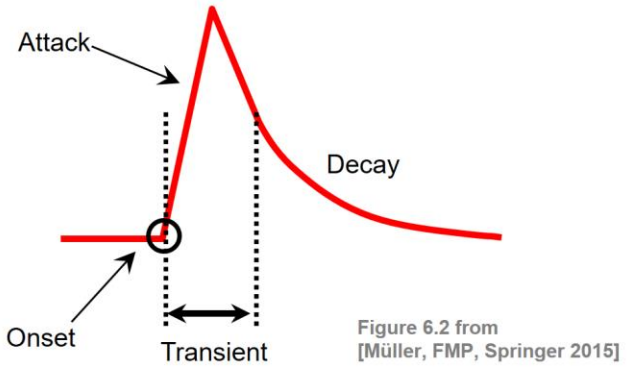
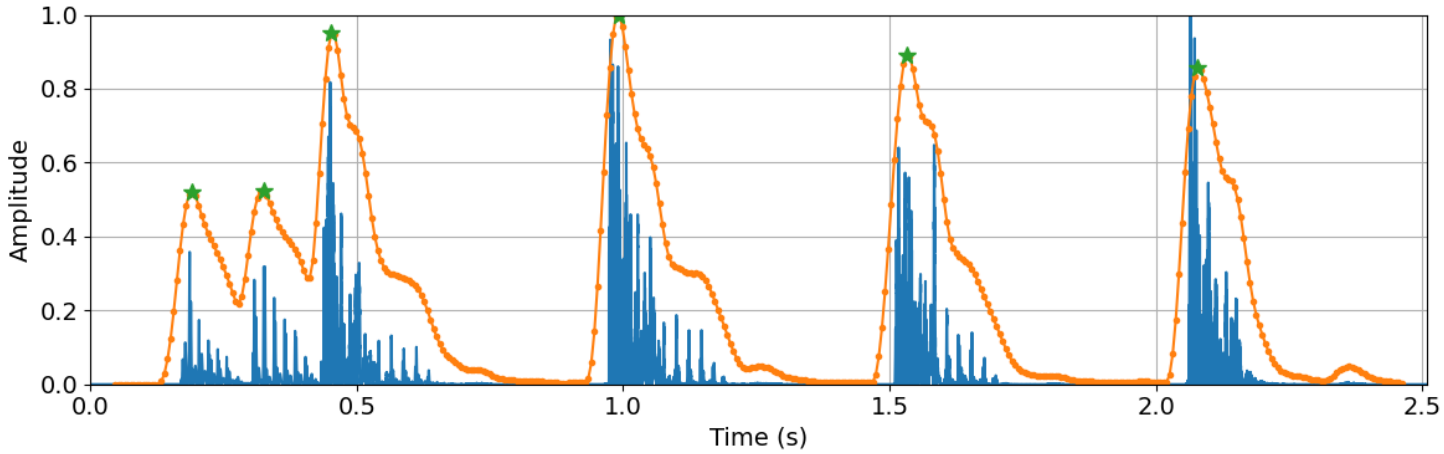


Figure 6.2 from [Müller, FMP, Springer 2015]

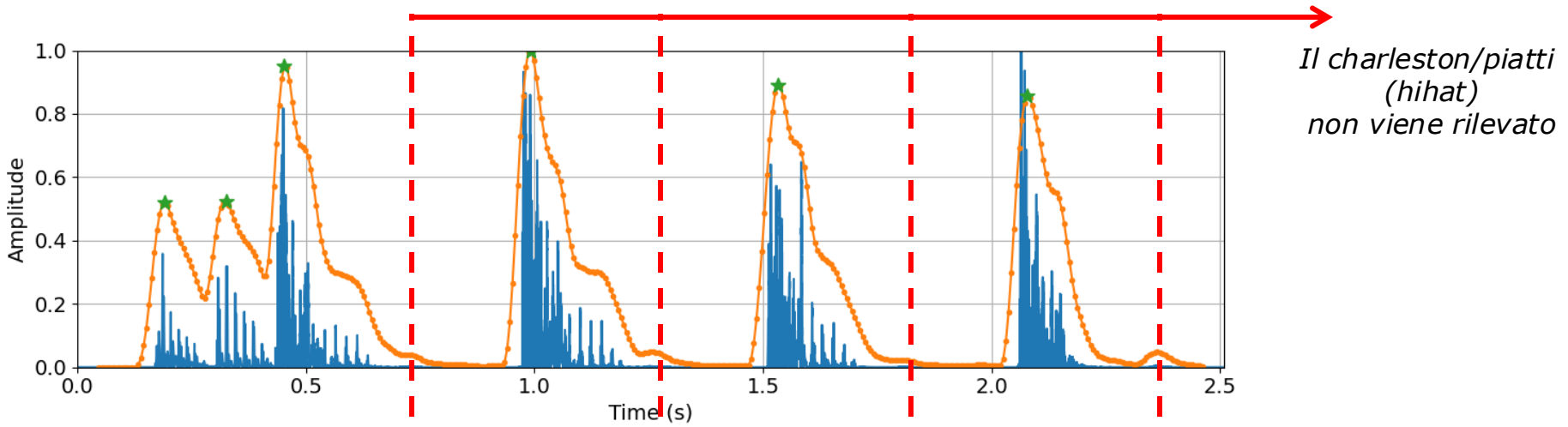
Esercizio: onset detection

- Utilizzare l'algoritmo [Analisi "a blocchi / finestre" in Python]
  - ✓ Per calcolare l'RMS su blocchi sovrapposti dell'audio
  - ✓ Provare differenti valori di L ed H e identificare quelli più adatti
- Rappresentare graficamente
  - ✓ Il quadrato dei valori del segnale (y^2) – linea blu
  - ✓ I valori dell'RMS calcolati per ogni frame – punti arancioni



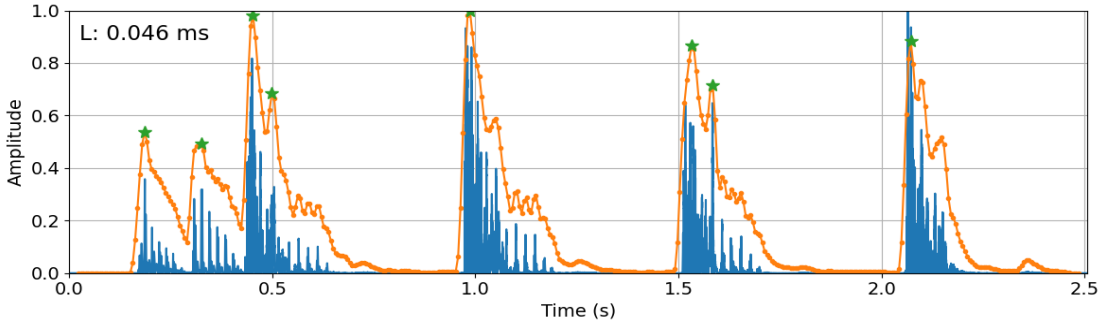
Esercizio: onset detection

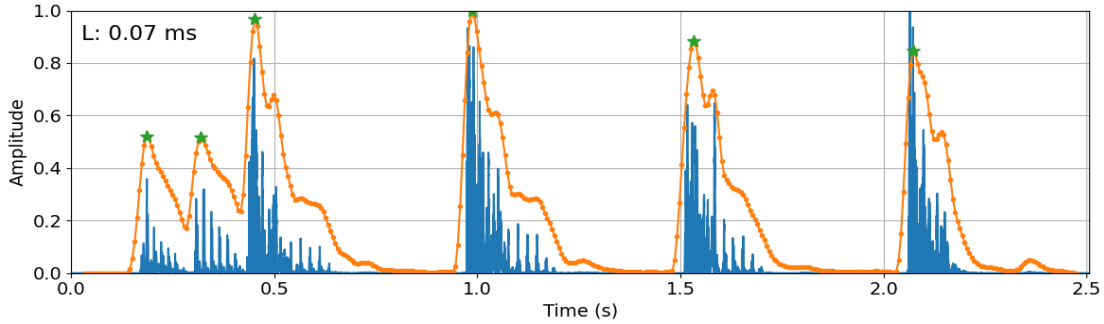
- Calcolare i picchi dell'RMS per identificare i punti di onset
  - ✓ scipy.signal.find_peaks
- Rappresentare graficamente
  - ✓ Il quadrato dei valori del segnale (y^2) – linea blu
  - ✓ I valori dell'RMS calcolati per ogni frame – punti arancioni
  - ✓ I punti di onset identificati – stelle verdi

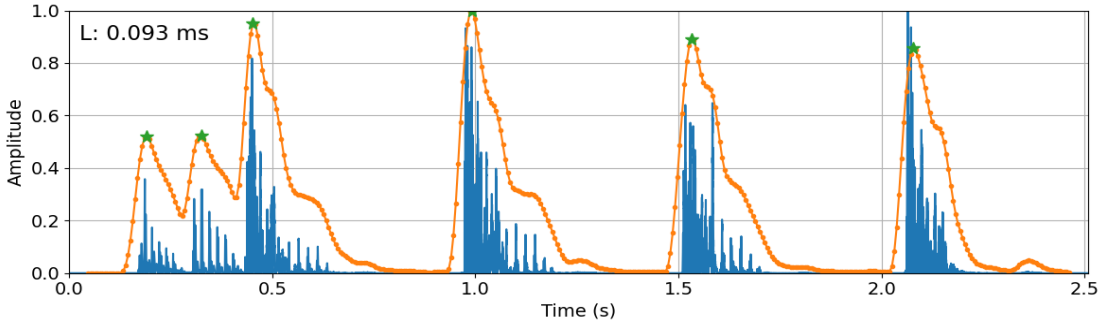


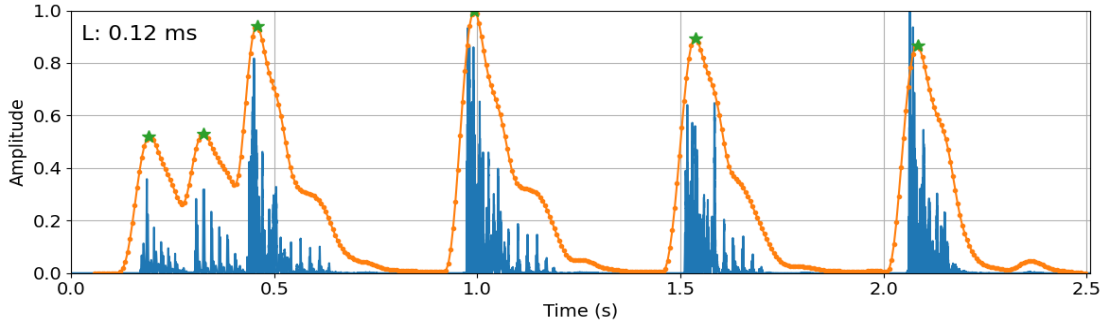
Il charleston/piatti (hihat) non viene rilevato

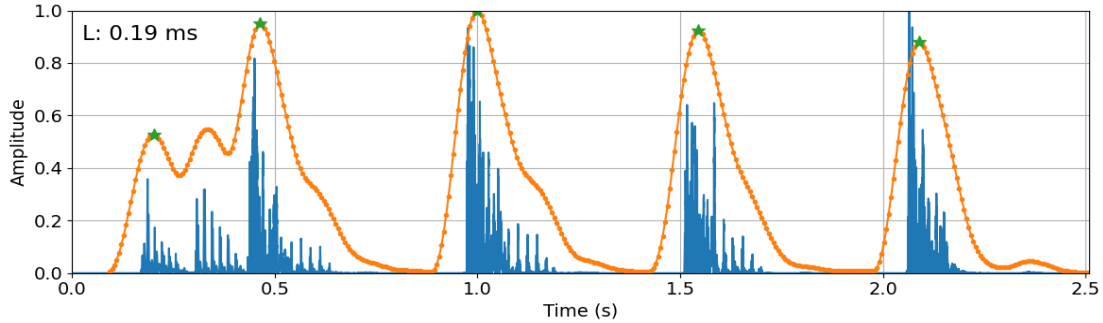
Scelta di lunghezza (L)











Python – time_framing_fn

```
def pad_to_multiple(array, multiple):
  length = len(array)
  target = ((length + multiple - 1) // multiple) * multiple
  return np.pad(array, (0, target - length), mode='constant')

def time_framing_fn(x, L, H, win=None, fn=np.max):
  win = signal.windows.boxcar(L) if win is None else win # rectangular
  y = pad_to_multiple(x,L)                  # padding right
  i_v = np.arange(0, len(y) - L+1, H) # index of first sample of each frame
  n_frames = len(i_v)                       # index of frame
  fn_v = np.zeros(len(n_v))                 # initialize feature array
  for n in arange(n_frames):
    i = i_v[n]
    frame = y[ i : i+L ]
    frame_w = frame * win                   # windowed frame
    fn_v[n] = fn(frame_w)
  return (i_v, fn_v)
```

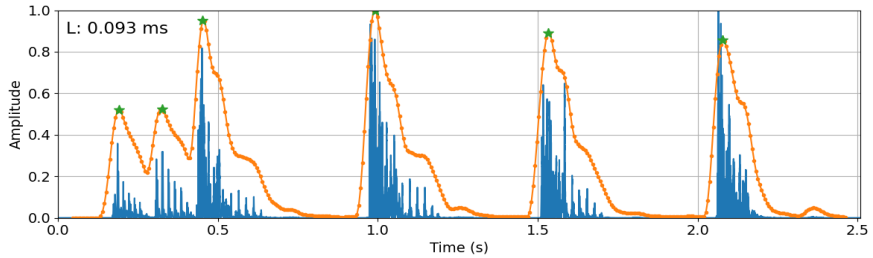
Novelty function

- Spesso viene calcolata una funzione detta di "novelty"
- In questo caso può essere semplicemente la derivata dei valori della feature
  - ✓ Nel dominio digitale la derivata può essere calcolata con la differenza
  - ✓ In questo caso ci si può limitare ai valori positivi, mettendo a zero quelli negativi

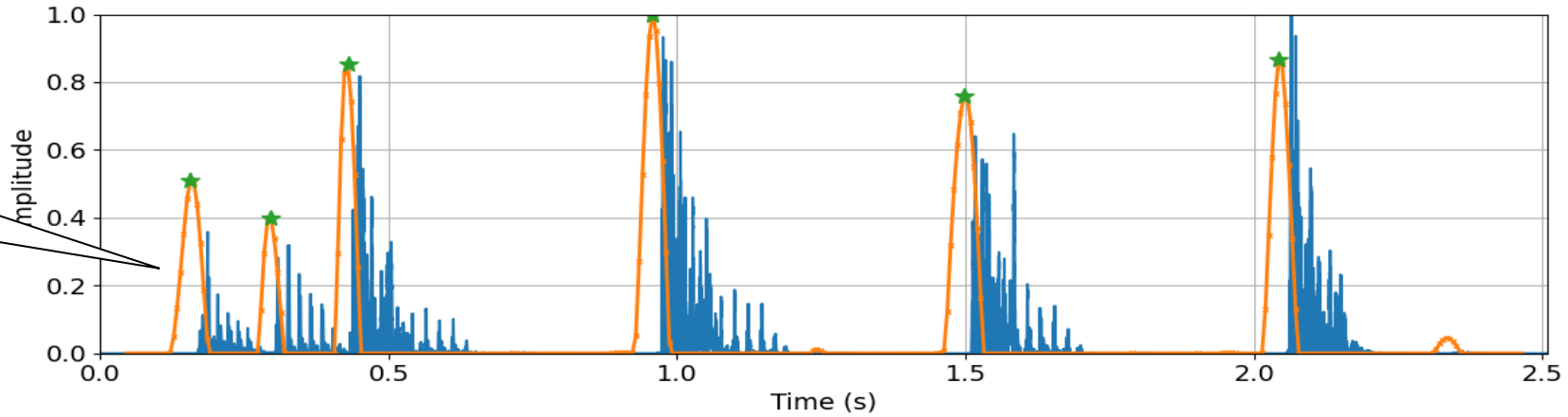
```
vals_diff = np.concatenate( (np.diff(vals), np.array([0])) )
novelty_v = np.copy(vals_diff)
novelty_v[novelty_v < 0] = 0
```

Novelty function

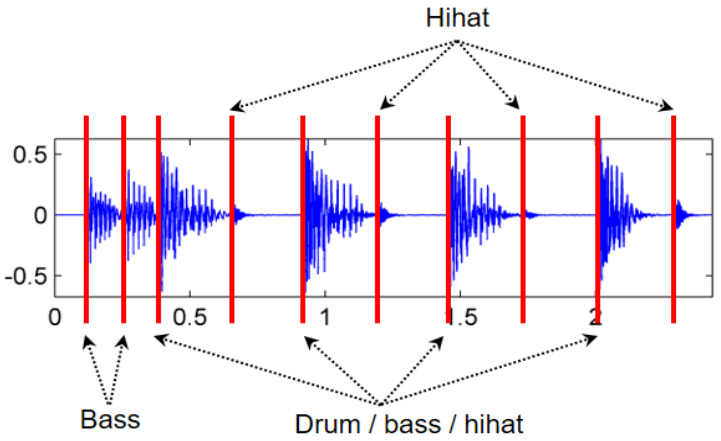
- Spesso viene calcolata una funzione detta di "novelty"
- Ad esempio, la derivata dei valori della feature
  - ✓ Nel dominio digitale la derivata può essere calcolata con la differenza
  - ✓ In questo caso ci si può limitare ai valori positivi, mettendo a zero quelli negativi



Mette in risalto i picchi, "nasconde" i non picchi

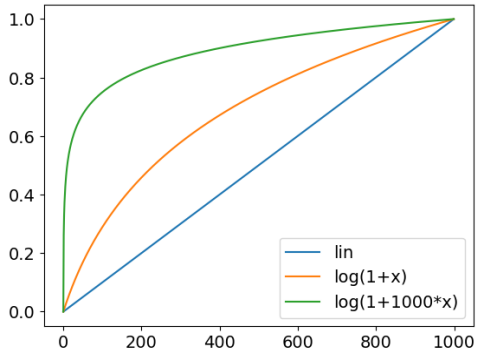


Compressione logaritmica



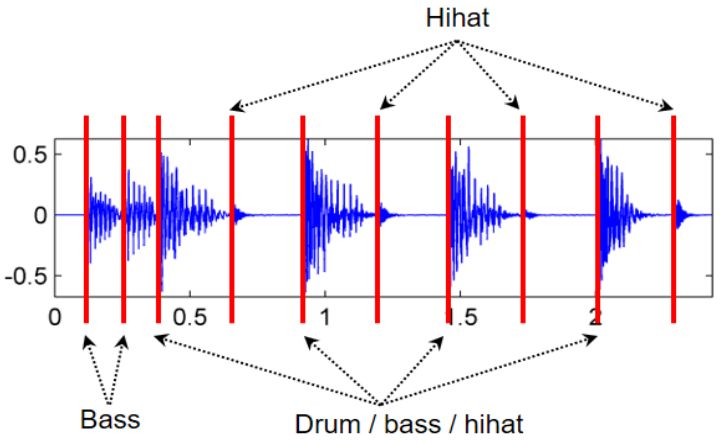
- Alcuni onset percepiti dal nostro orecchio, e.g. hihat, non vengono tuttavia riconosciuti
- La percezione umana è in scala logaritmica, è consigliabile utilizzare la scala in dB, o meglio, una funzione di *compressione logaritmica* del tipo

$$F_G(v) = \log(1 + G \cdot v)$$

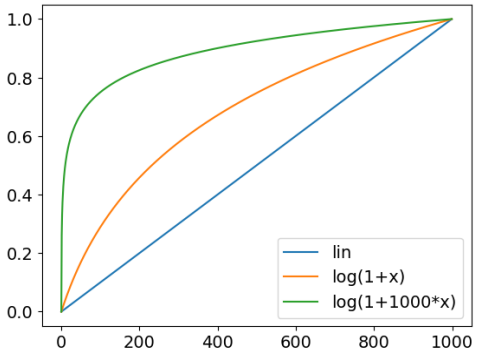


- ✓ Il **log** comprime la dinamica ed esalta i valori "piccoli" che altrimenti rimarrebbero "nascosti".
- ✓ **G** è un fattore di espansione, più è grande più esalta i valori piccoli, ma un valore troppo alto potrebbe esaltare anche il semplice rumore
- ✓ Il **+1** serve a dare stabilità numerica ed evitare la zona [0-1]

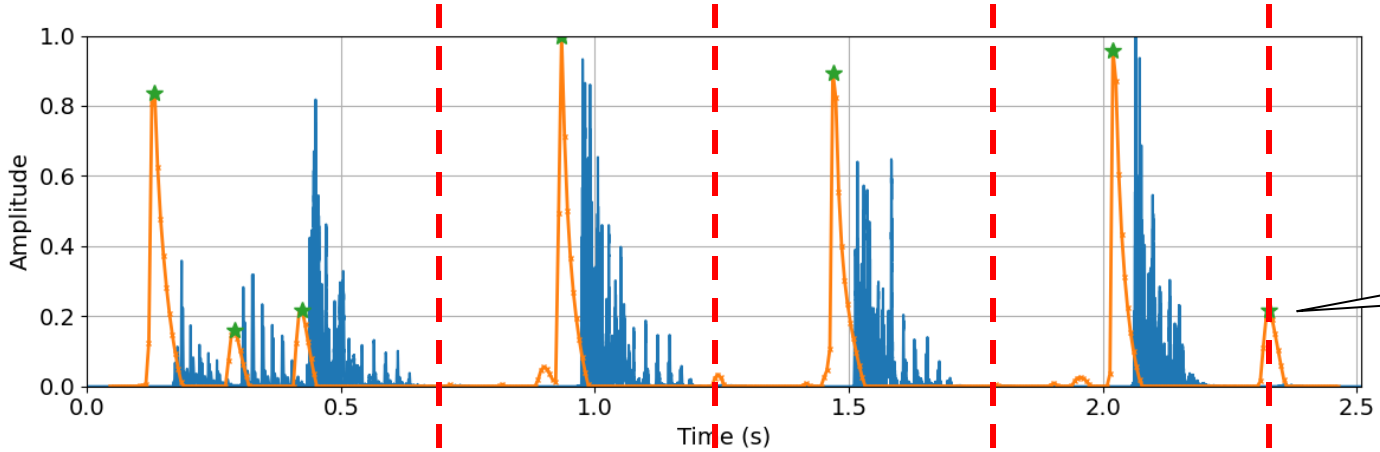
Compressione logaritmica



- Alcuni onset percepiti dal nostro orecchio, e.g. hihat, non vengono tuttavia riconosciuti
- La percezione umana è in scala logaritmica, è consigliabile utilizzare la scala in dB, o meglio, una funzione di *compressione logaritmica* del tipo




```
def log_g(x,g):
    return np.log(1 + g * x))
```



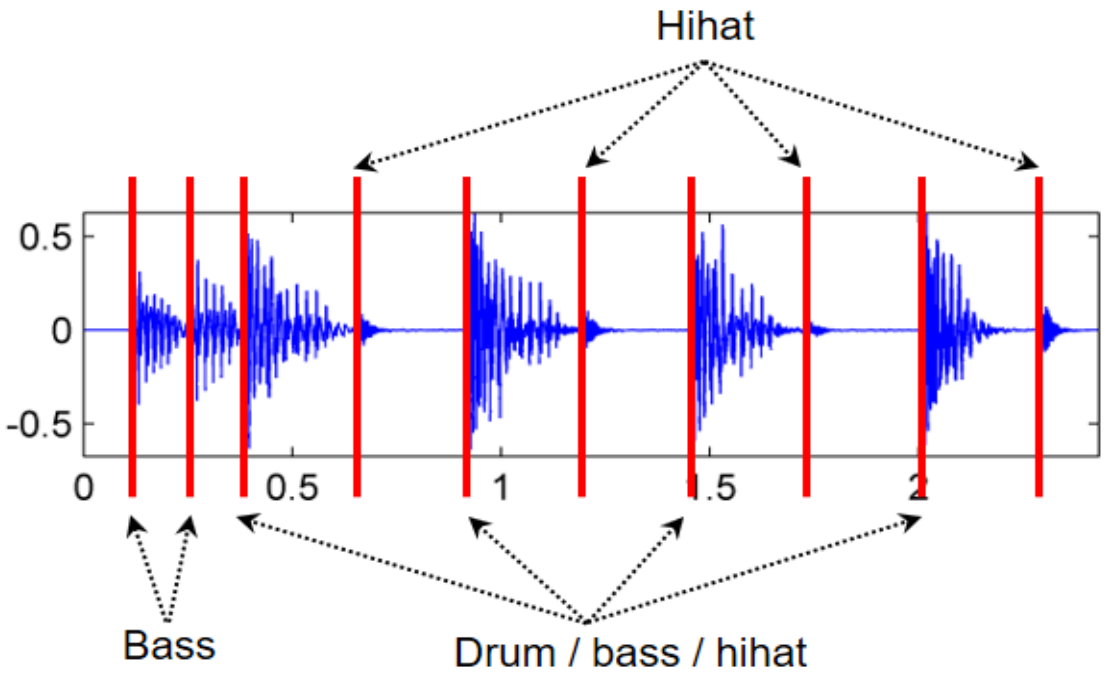
Un hihat viene identificato

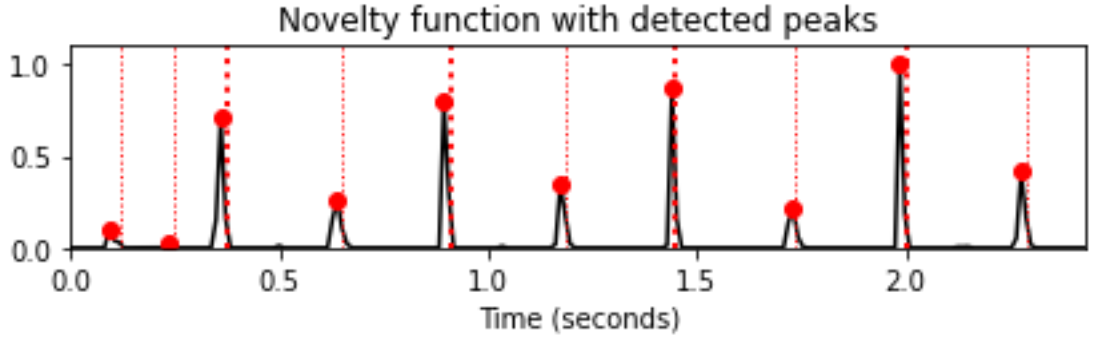
$$F_G(v) = \log(1 + G \cdot v)$$

Onset detection

- Per una trattazione più approfonditae l'analisi di novelty functions basate su differenti anche nel dominio della frequenza si rimanda al testo di Muller [FMP] e al link

  https://www.audiolabs-erlangen.de/resources/MIR/FMP/C6/C6S1_OnsetDetection.html





ANALISI TEMPO/FREQUENZA

Analisi tempo/frequenza

- La trasformata di Fourier (DFT) applicata su un intervallo del segnale ne **fotografa** lo spettro mediandone il contenuto spettrale sull'intervallo, ma perdendo il concetto di tempo.
- Per analizzare l'evoluzione temporale dello spettro di un segnale è necessario definire una trasformata su **due** dimensioni: *frequenza* e *tempo*
- Si definisce quindi la Short-Time (discrete) Fourier Transform, la Trasformata di Fourier a Breve Termine

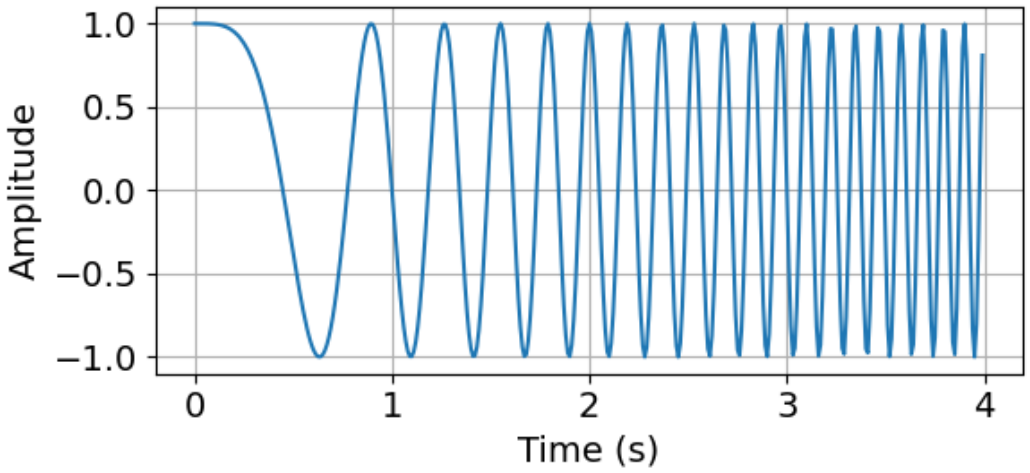
$$X[k,m] = \sum_{n=0}^{L-1} w[n - mH]x[n]\, e^{-j\frac{2\pi k}{N}n}$$

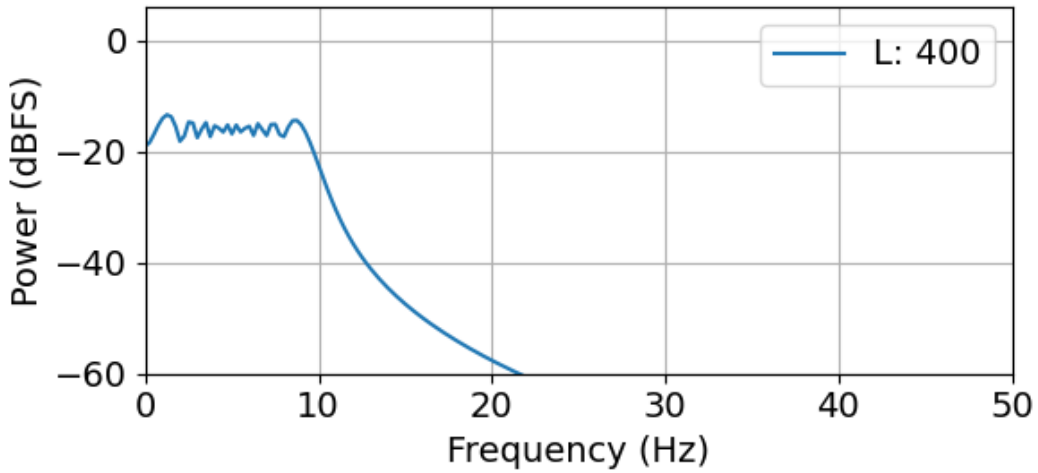
Analisi tempo/frequenza

- La trasformata di Fourier (DFT) applicata su un intervallo del segnale ne **fotografa** lo spettro mediandone il contenuto spettrale sull'intervallo, ma perdendo il concetto di tempo.

- Esempio: CHIRP

```
x = signal.chirp(t, f0=0, t1=4, f1=10, method='linear')
```





Trasformata di Fourier di breve termine

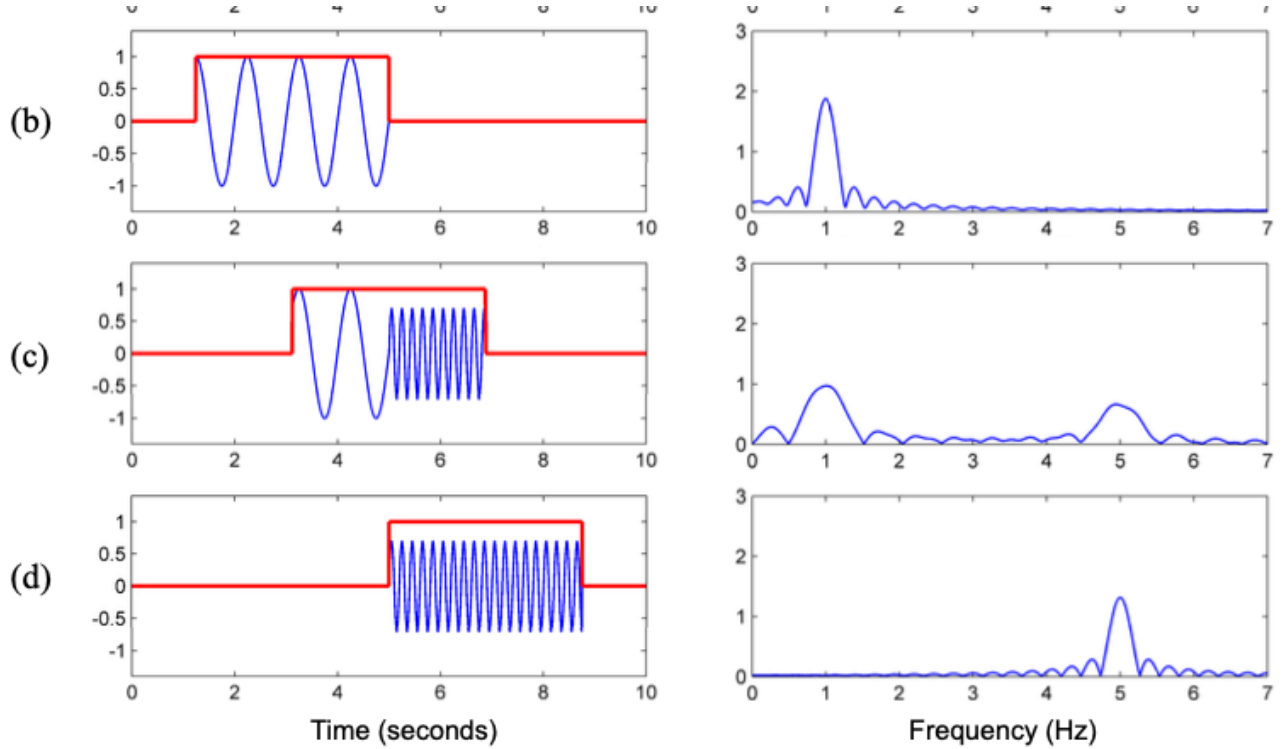
$$X[k,m] = \sum_{n=0}^{L-1} w[n - mH] \cdot x[n] \cdot e^{-j\frac{2\pi k}{N}n}$$

- La ST-DFT è la DFT di un segnale x[n] troncato su una finestra temporale m-esima w[n-m*H]
  - ✓ m è l'indice del frame (finestra)
  - ✓ L è la lunghezza del frame (finestra)
  - ✓ H è l'hop-size, il passo di avanzamento tra un frame e il successivo
  - ✓ N è la lunghezza della DFT dove $0 \leq k \leq N-1$
- Il risultato è uno spettro per ogni valore di *m*

Reference:

Esempio: ST-DFT

- L'utilizzo della ST-DFT permette di studiare **l'evoluzione temporale** del contenuto in frequenza



X [: , (m-1)*H]

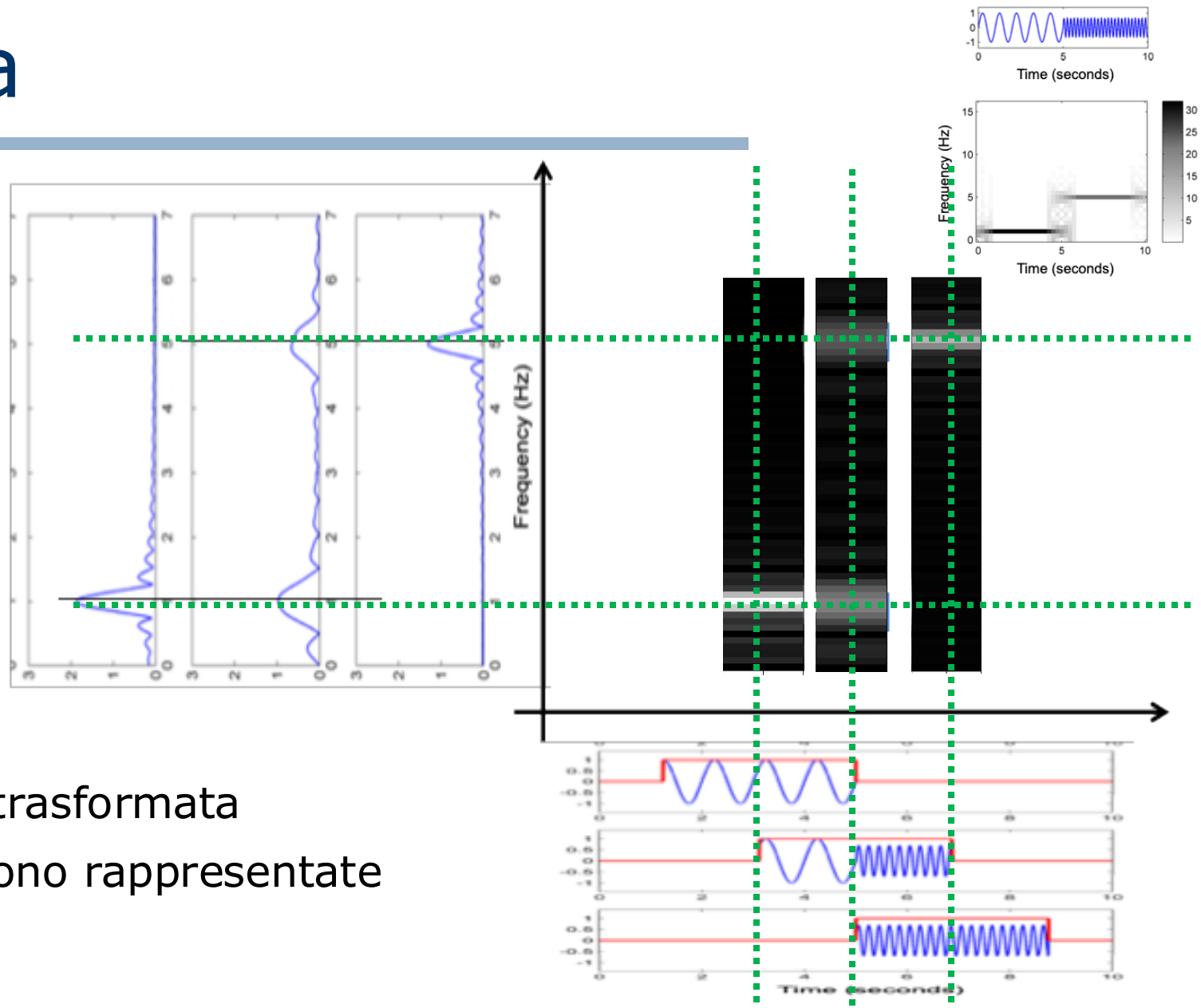
X [: , (m)*H]

X [: , (m+1)*H]

Muller, Fig.2.8

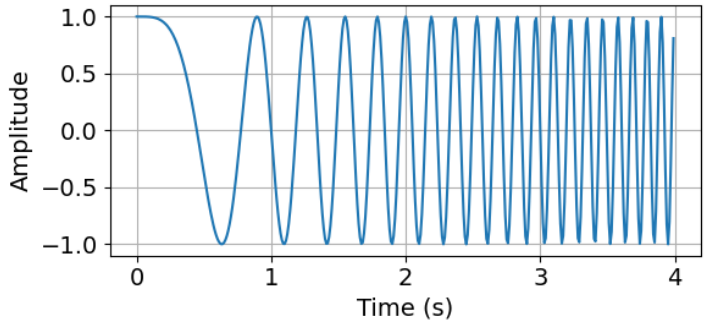
Spettrogramma

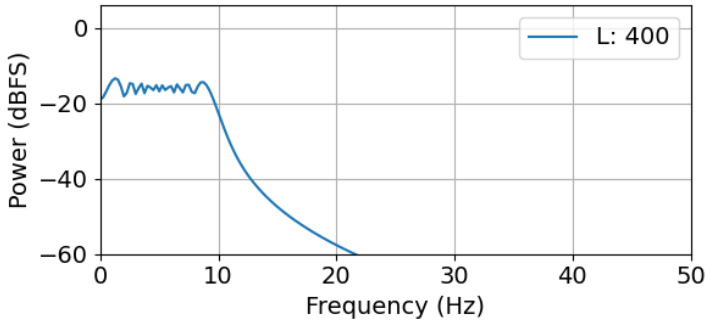
- Lo spettrogramma è la rappresentazione del segnale nel dominio tempo / frequenza
- Sull'asse delle X abbiamo il tempo corrispondente alle "finestre"
- Sull'asse delle Y abbiamo le frequenze corrispondenti ai bin della trasformata
- Le intensità dello spettro sono rappresentate da colori o scala di grigi

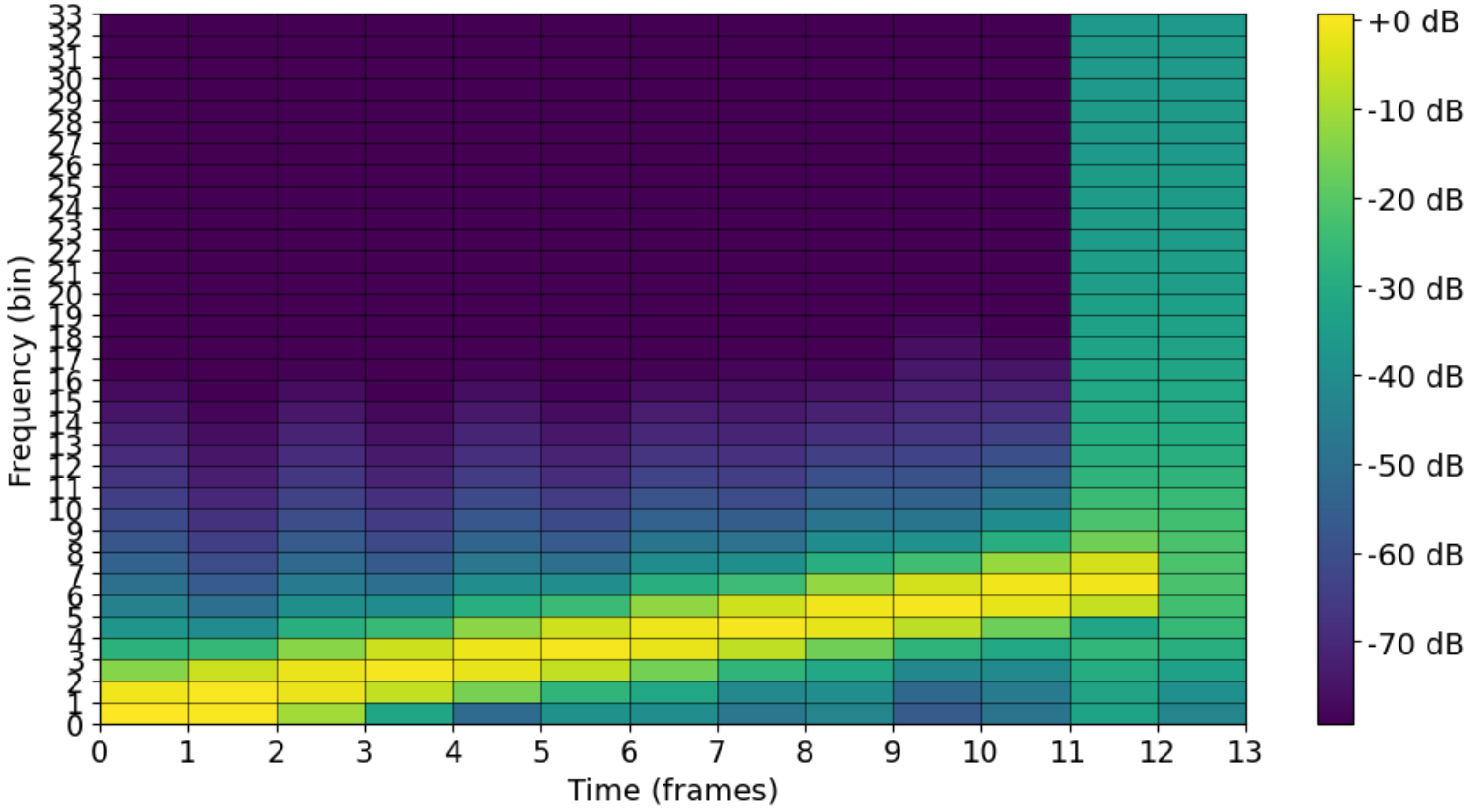


Esempio: chirp

- Fs=100, dur=4
- L=64, H=32

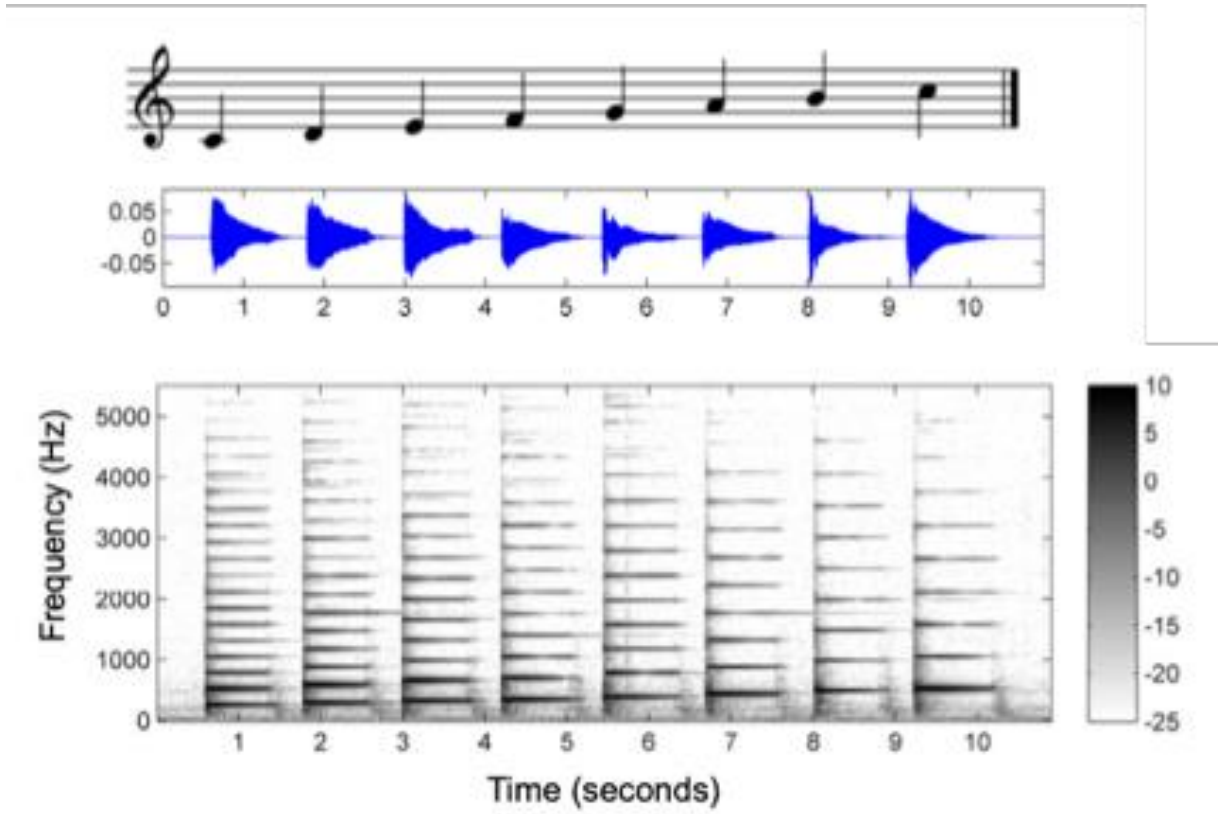






Esempi: spettrogramma

Scala di DO maggiore suonata al pianoforte



Muller, Fig.2.10

Esempi: spettrogramma

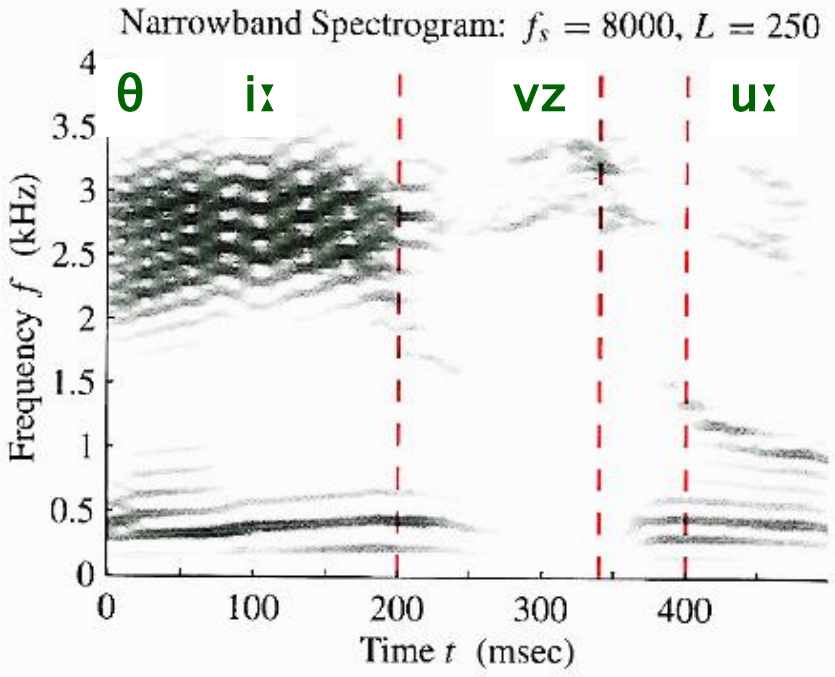


Figure 13-25: Spectrogram of speech signal; sampling frequency $f_s = 8000$ Hz, FFT length $N = 400$, and Hann window of length $L = 250$ (equivalent to 31.25 msec). Spectrum slices at times $nT_s = 200$, 340, and 400 msec are shown individually in Fig. 13-26.

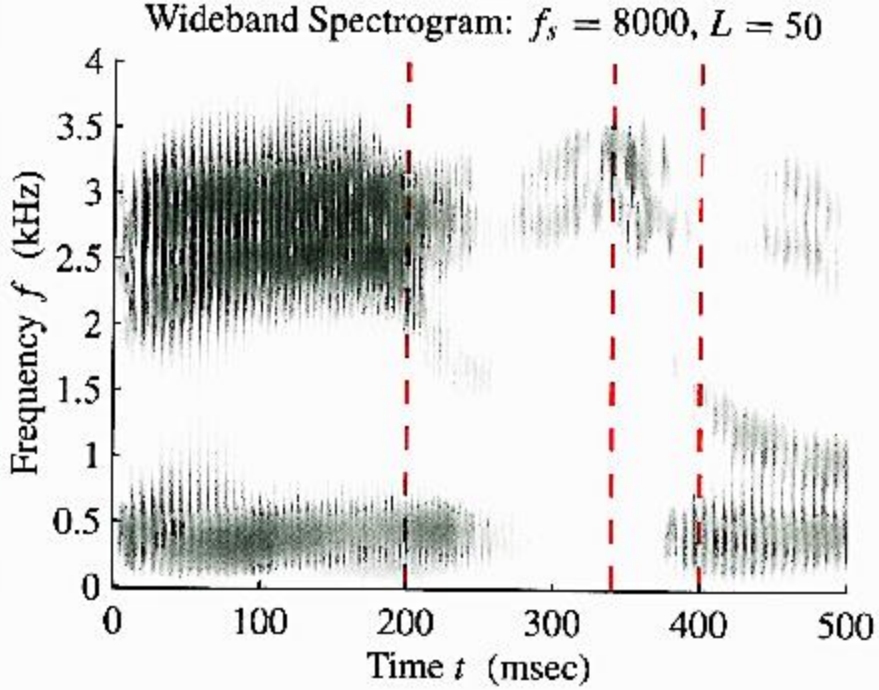


Figure 13-27: Spectrogram of speech signal; sampling frequency $f_s = 8000$ Hz, window length $L = 50$ (6.25 msec). Spectrum slices at times $nT_s = 200$, 340, and 400 msec are shown individually in Fig. 13-28.

Esempi: spettrogramma - narrow

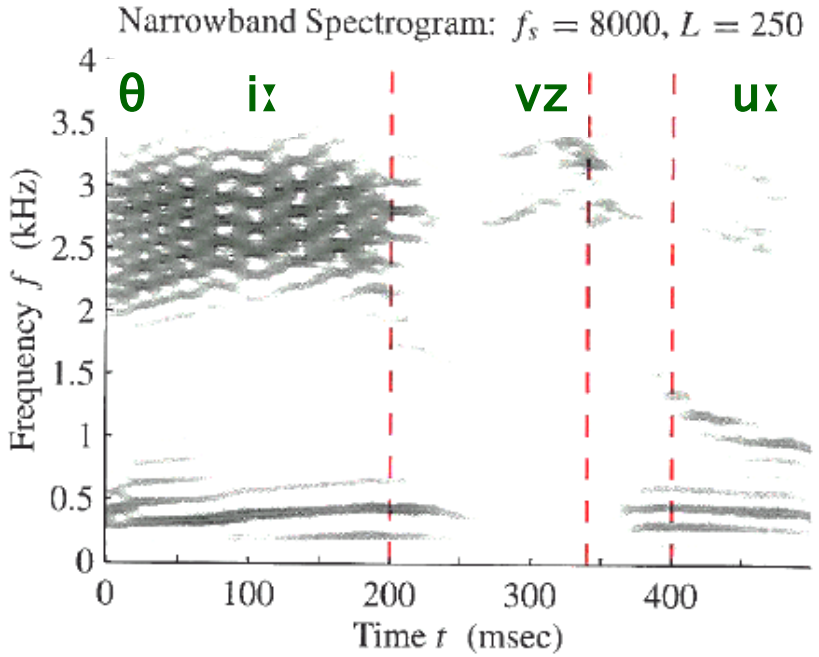


Figure 13-25: Spectrogram of speech signal; sampling frequency $f_s = 8000$ Hz, FFT length $N = 400$, and Hann window of length $L = 250$ (equivalent to 31.25 msec). Spectrum slices at times $nT_s = 200$, 340, and 400 msec are shown individually in Fig. 13-26.

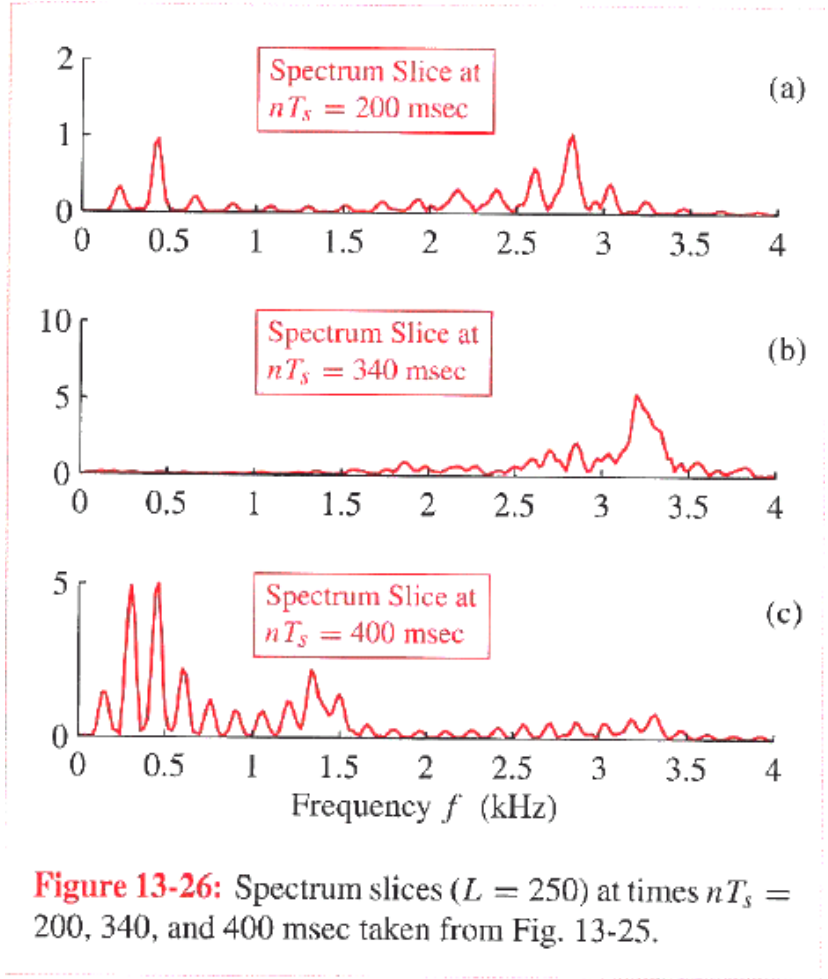


Figure 13-26: Spectrum slices ($L = 250$) at times $nT_s = 200$, 340, and 400 msec taken from Fig. 13-25.

Esempi: spettrogramma - wide

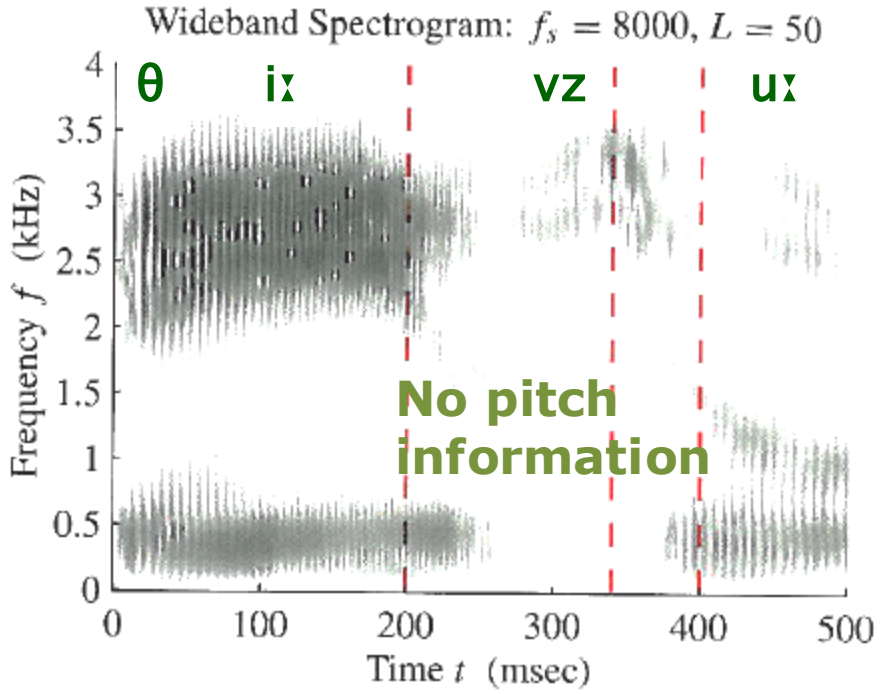


Figure 13-27: Spectrogram of speech signal; sampling frequency $f_s = 8000$ Hz, window length $L = 50$ (6.25 msec). Spectrum slices at times $nT_s = 200$, 340, and 400 msec are shown individually in Fig. 13-28.

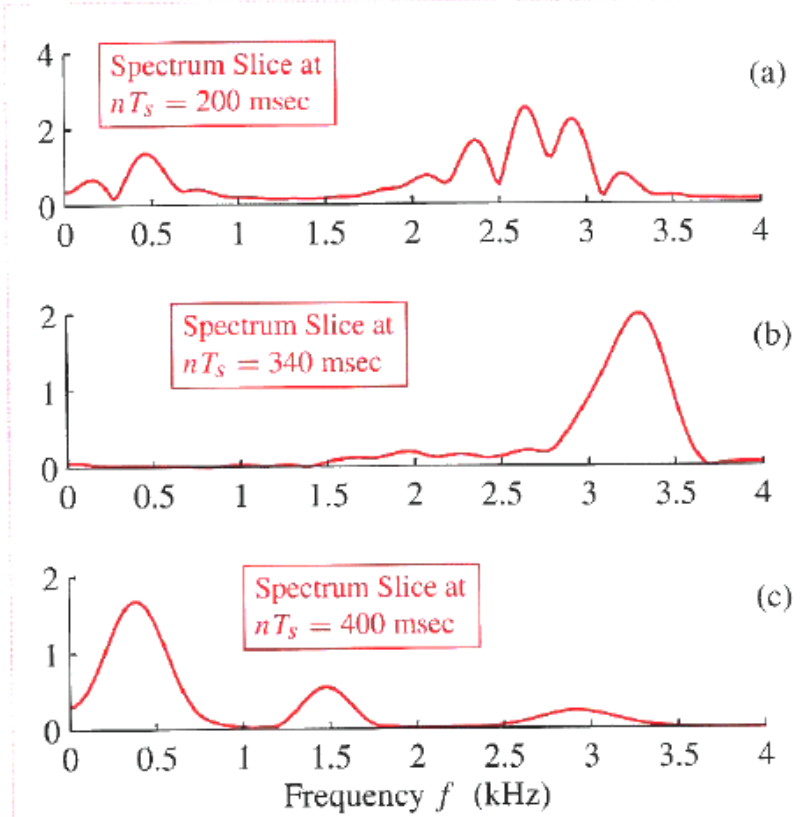
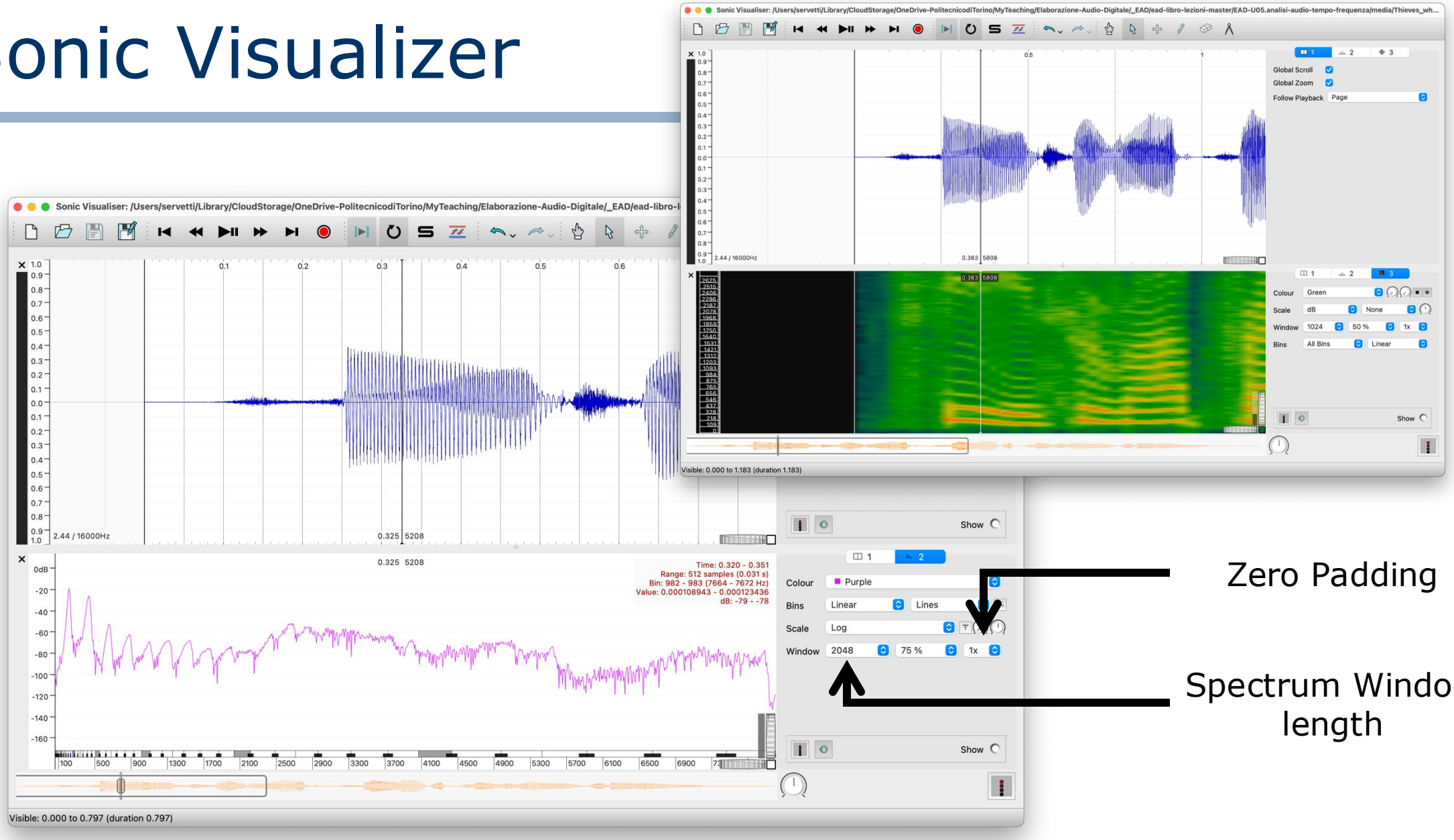


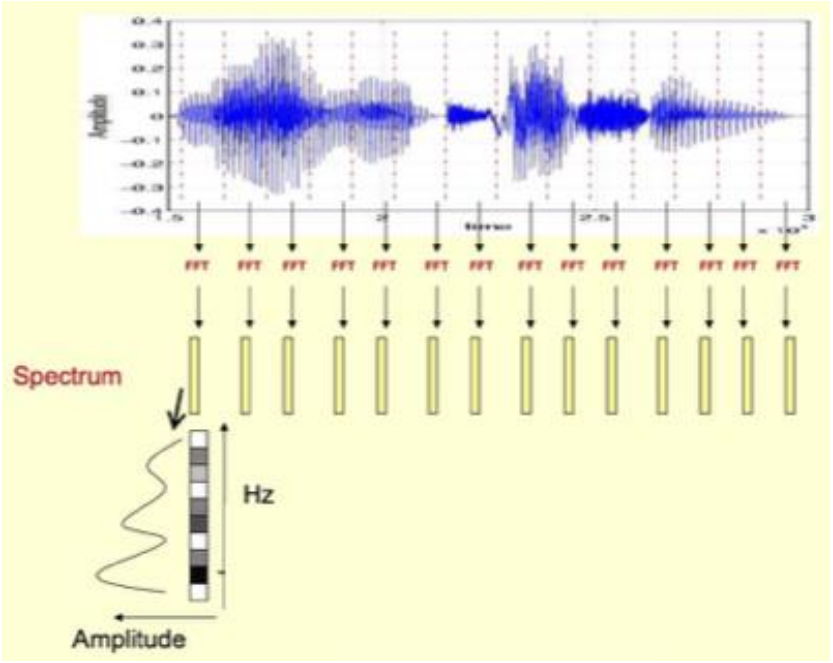
Figure 13-28: Spectrum slices ($L = 50$) at times $nT_s = 200$, 340, and 400 msec taken from Fig. 13-27.

Sonic Visualizer



Calcolo dello spettrogramma

- Estrarre frame ad intervalli temporali successivi parz. sovrapposti
  - ✓ Analisi a finestre parzialmente sovrapposte
- Calcolarne lo spettro pesato con una finestra
  - ✓ Utilizzare una finestra non rettangolare per "pesare" ciascun frame
  - ✓ w = scipy.signal.windows.hann(L)
- Memorizzare ciascuno spettro nelle colonne di una matrice
- Rappresentare la matrice come una immagine
  - ✓ plt.imshow(S)



Calcolo dello spettrogramma

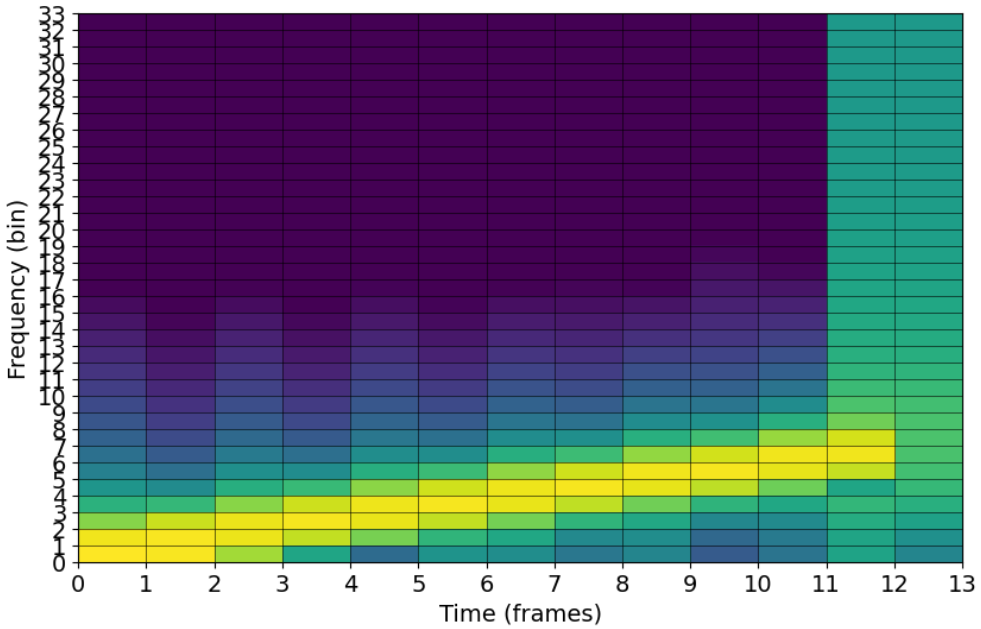
```
i_v = np.arange(0, len(y)- L+1, H)
n_frames = len(i_v)
S = np.zeros((n_frames, N//2+1))
```

Indice del primo campione di ogni frame: 0, H, 2H, 3H

Dimensione dello spettrogramma:
len(n_v) frame temporali
N//2+1 coefficienti spettrali

```
for n in np.arange(n_frames):
  i = i_v[n]
  frame = y[i:i+L]
  win = scipy.signal.windows.hann(L)
  frame_w = frame * win
  X, f = dft(frame_w, Fs, N)
  S[i] = X
```

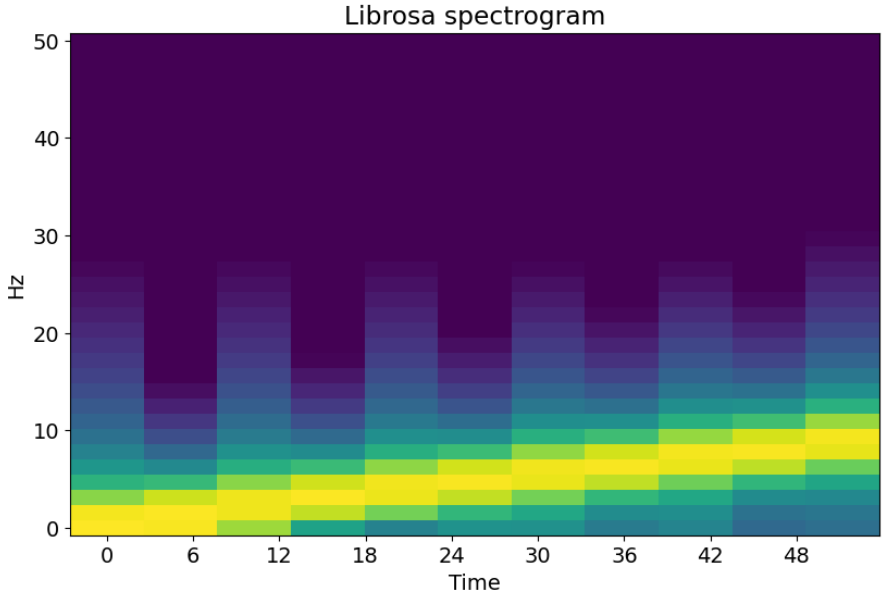
Finestra e finestratura




```
plt.imshow(S.T, origin='lower', aspect='auto', cmap='gray_r')
```

Plot spettrogramma librosa

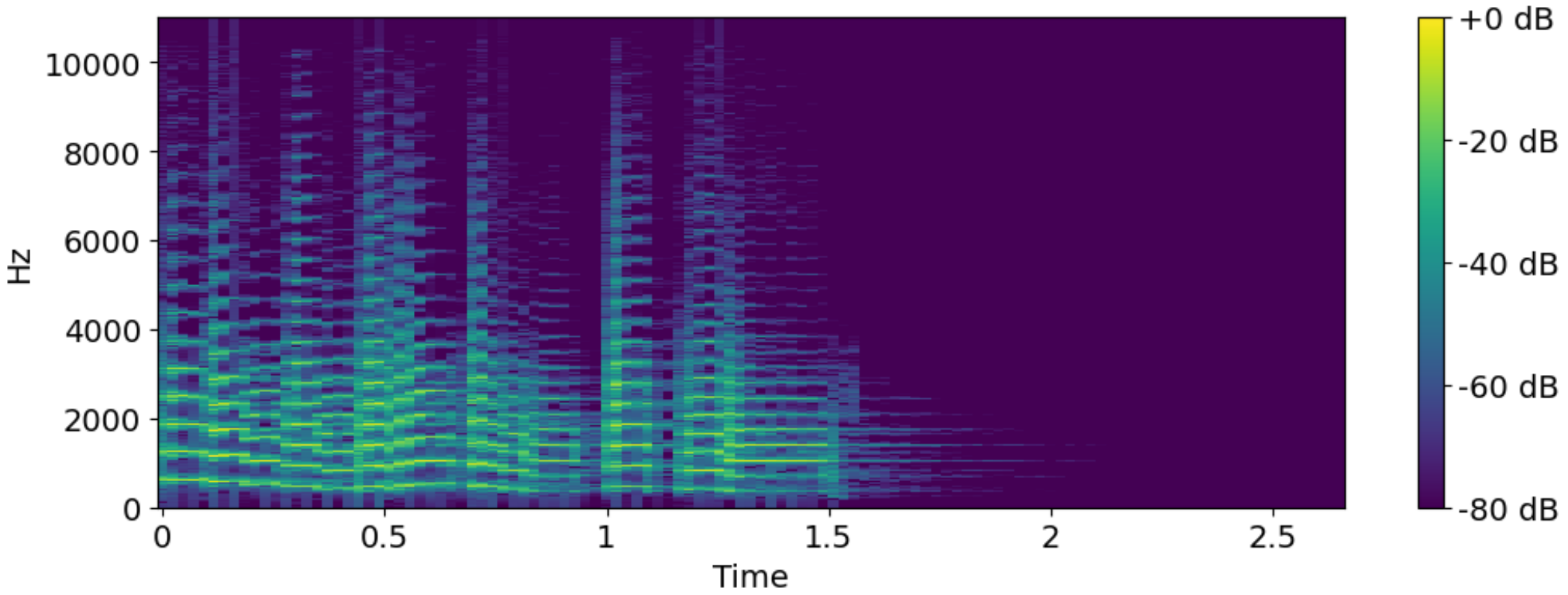
```
# Compute STFT => returns complex valued
  spectrogram
S = librosa.stft(x, win_length=L, hop_length=H,
  n_fft=N, window='hann')

# Convert to dB where 0 dB corresponds to max value
S_db = librosa.amplitude_to_db(np.abs(S), ref=np.max)

# Compute freq and time axes values
freqs = librosa.fft_frequencies(sr=Fs, n_fft=N)
times = librosa.frames_to_time(np.arange(S.shape[1]), sr=Fs, hop_length=H)

# Display spectrogram with librosa function specshow function
img = librosa.display.specshow( S_db, sr=sr,
      cmap="viridis", x_axis='time', y_axis='linear', ax=ax) # y_axis='log'
fig.colorbar(img, format="%+2.f dB")
```



https://librosa.org/doc/0.11.0/auto_examples/plot_display.html

Esempio

- Le alte frequenze sono le prime ad "esaurirsi/estinguersi" con il passare del tempo




```
y, sr = librosa.load(librosa.ex('trumpet'))
```

Principio di indeterminazione

- Effetto della lunghezza del frame L
  - ✓ Con una finestra corta
    si ha buona risoluzione nel tempo
    (punto di "attacco")
    ma scarsa in frequenza
  - ✓ Con una finestra lunga
    si ha scarsa risoluzione nel tempo
    ma buona risoluzione in frequenza
    (distinzione frequenza toni)

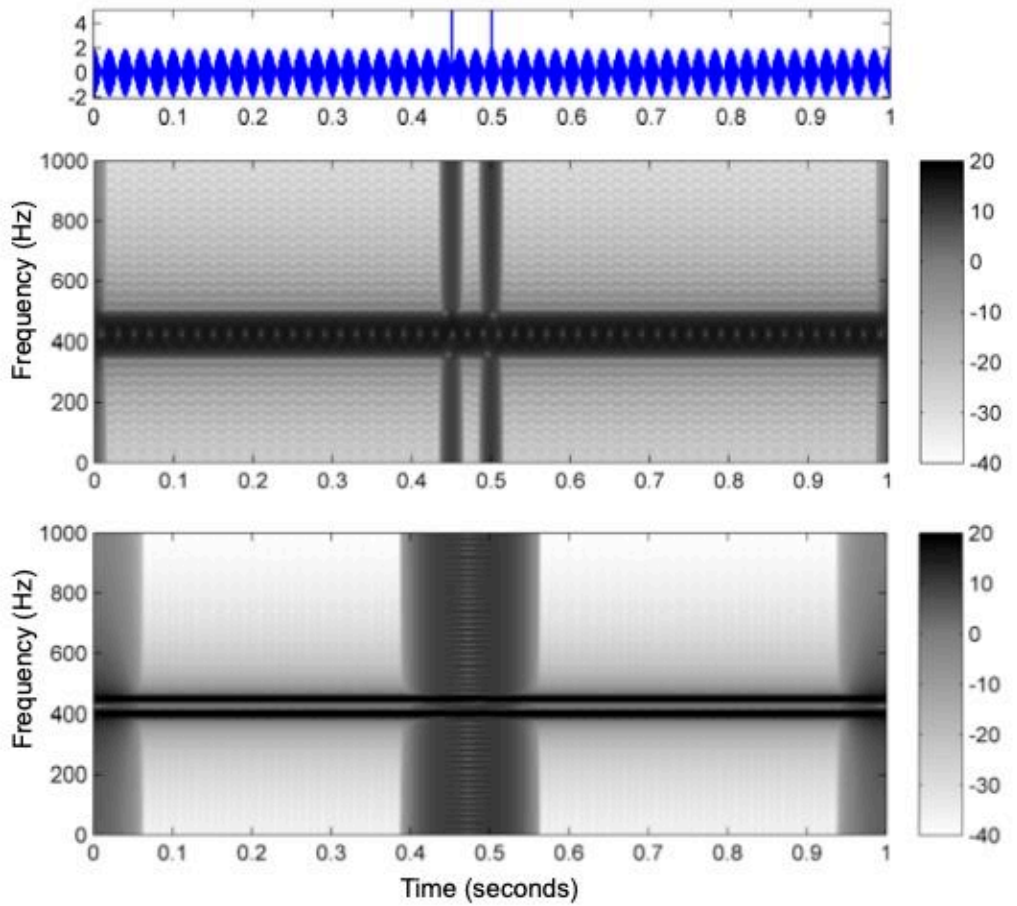
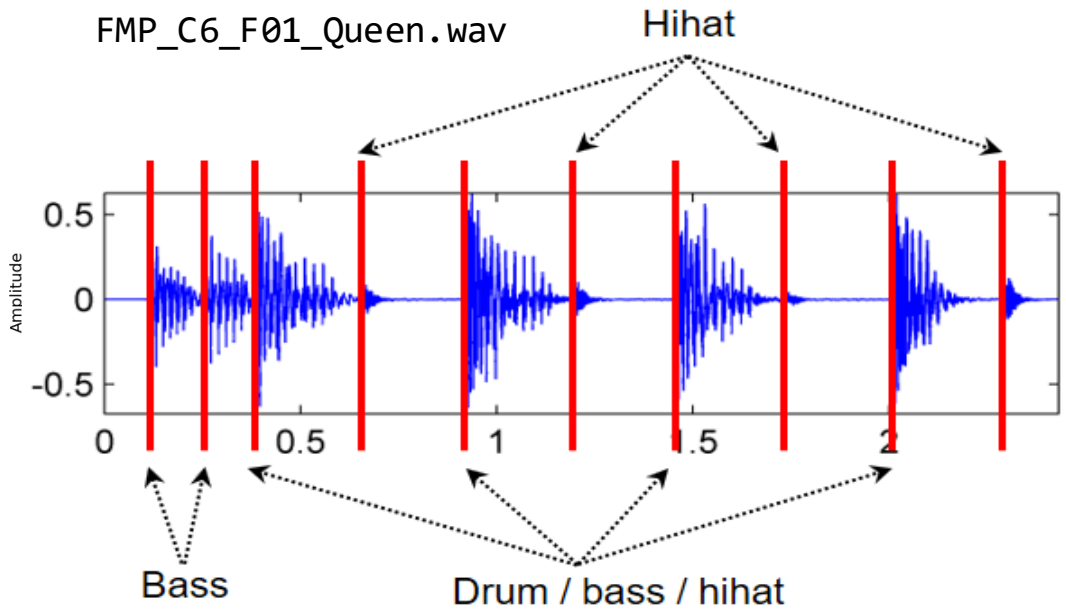


Fig. 2.32 Spectrogram using different window sizes. **(a)** Signal. **(b)** Spectrogram with short Hann window (32 ms). **(c)** Spectrogram with long Hann window (128 ms).

Onset Detection

- Consideriamo un esempio di analisi chiamato **Onset Detection**
  - ✓ L'idea è quella di identificare l'inizio di un suono, cioè un rapido cambiamento dopo un periodo di (quasi) silenzio
  - ✓ Ad esempio suoni percussivi in un brano musicale, (simile nella voce si può parlare di voice-activity-detection)



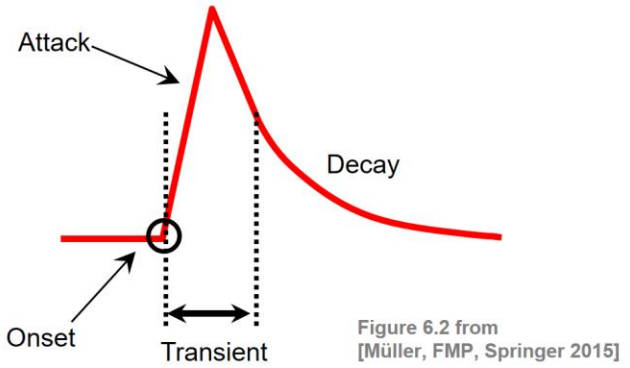


Figure 6.2 from [Müller, FMP, Springer 2015]