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

- EFFETTO AUTOTUNE

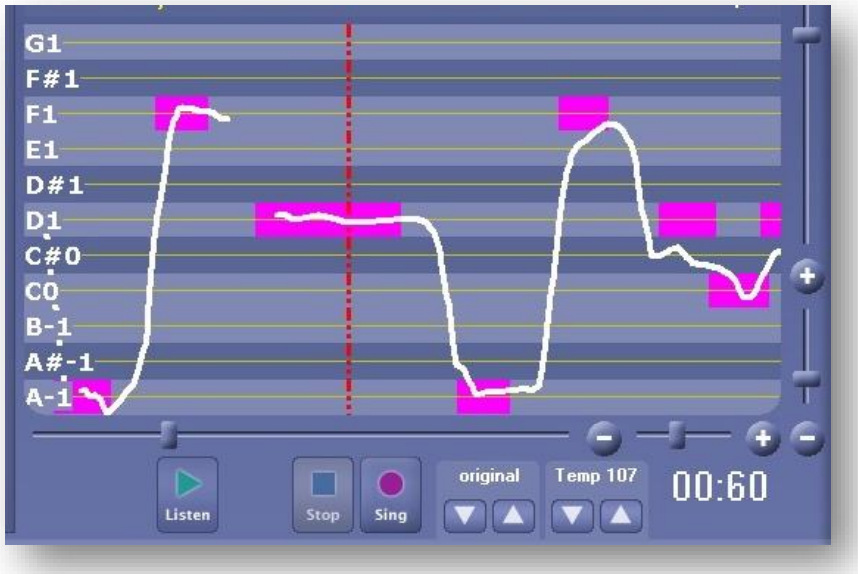
- 1. Identificazione del pitch
  - ✓ Tecnica dell'autocorrelazione
- 2. Mapping del pitch
- 3. Trasposizione del pitch
  - ✓ Tecnica Time Domain – Pitch Synchronous Overlap and Add

Bibliografia

- Identificazione del pitch
  - ✓ Rabiner, "On the Use of Autocorrelation Analysis for Pitch Detection", 1977
  - ✓ Benesty, "Handbook of Speech Processing – Ch. 10 Pitch and Voicing Determination", 2007
- Time-domain Pitch Synchronous Overlap Add
  - ✓ E. Moulines and F. Charpentier, "*Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones*," Speech communication, 1990.
  - ✓ Dutoit, "An Introduction to Text-to-Speech Synthesis – Ch.10 Time-Domain Algorithms ", Springer, 1997
- Muller, "A Review of Time-Scale Modification of Music Signals", 2016

Effetto Autotune

- Utilizzo di tecniche di **intonazione forzata** e **trasposizione del pitch**
- Algoritmo
  - ✓ Stima del pitch su porzioni di 10-40 ms
  - ✓ Definizione del pitch target
    - La nota più vicina o quella desiderata
  - ✓ Modifica dell'intonazione
    - Variazione pitch ma stessa durata



Effetto Autotune

- Cheer effect – 1998 album "Believe"
  - ✓ Autotune con parametri "alterati" per non essere una correzione ma diventare un effetto (e.g., tempo di "attacco" immediato, timbro sintetico, robotizzazione)

- Plugins
  - ✓ Antares Auto-Tune software (Cheer 1998)
  - ✓ Melodyne (commerciale)
  - ✓ Graillon 3 (versione free)

Reference: https://www.youtube.com/watch?v=nZXRV4MezEw

Scala musicale

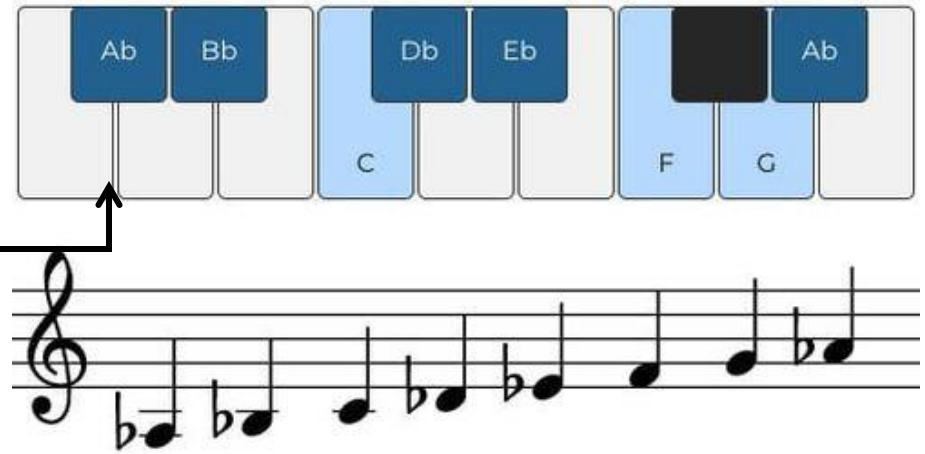
- La scala musicale è definita come 7 note con una determinata relazione di altezza (pitch) a partire da una nota "tonica"
- Esempio: scala di Do maggiore
  - ✓ Do – Re – Mi – Fa – Sol – La – Si –
  - ✓ T T S T T T S
- Esempio: scala di La♭ maggiore
  - ✓ La♭ - Si♭ - Do - Re♭ - Mi♭ - Fa – Sol

Analogia con
i tasti del pianoforte

MIDI

T T S T T T S

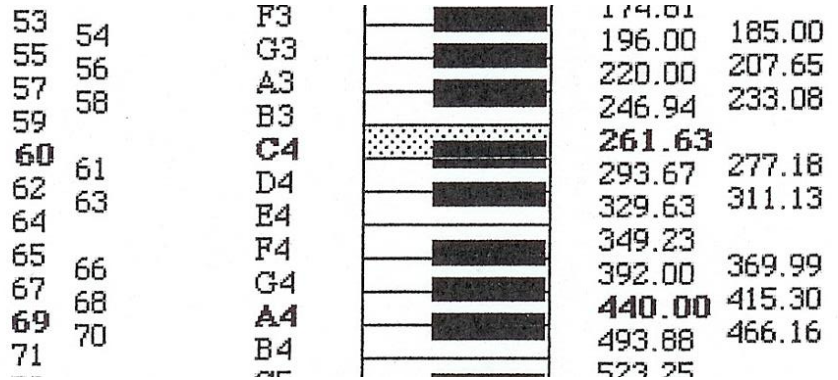
56 (57) 58 (59) 60 61 (62) 63 (64) 65 (66) 67 68



Reference: https://it.wikipedia.org/wiki/Scala_maggiore

Intonazione forzata

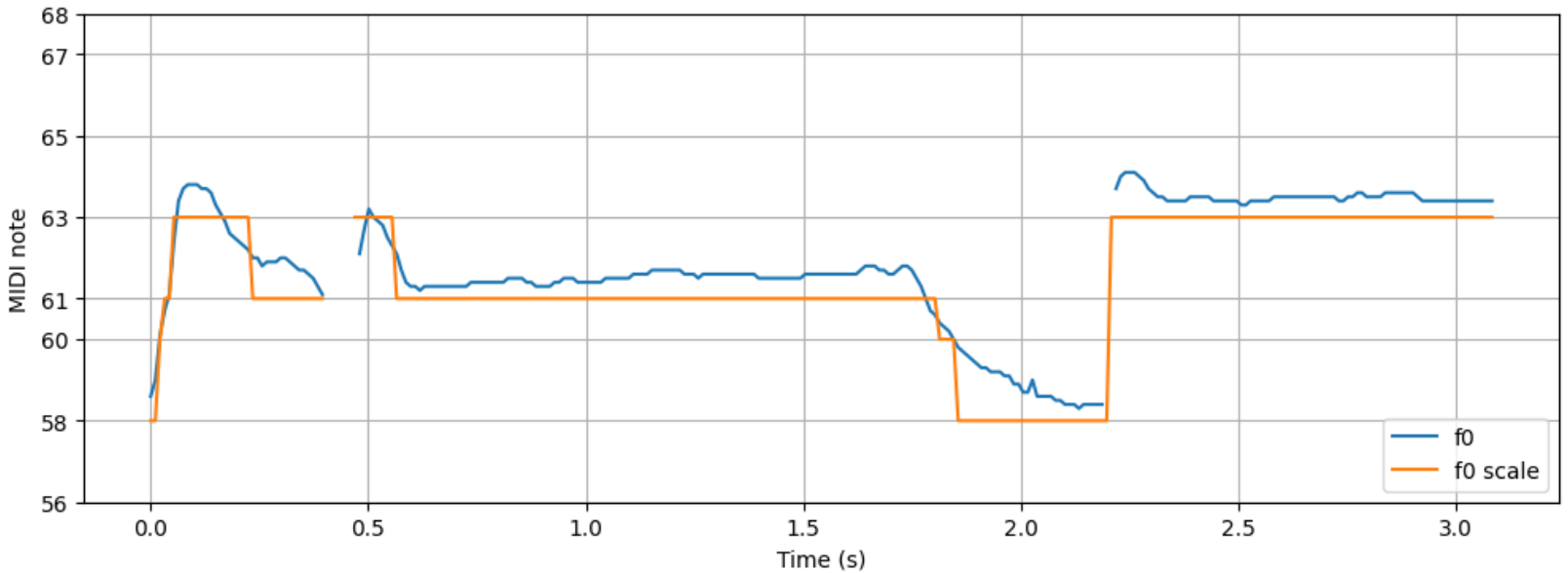
- Correzione dell'intonazione in modo da corrispondere sempre ad una nota della scala musicale usata



63: 311.17 Hz --

61: 277.18 Hz --
60: 261.63 Hz --

58: 233.08 Hz --

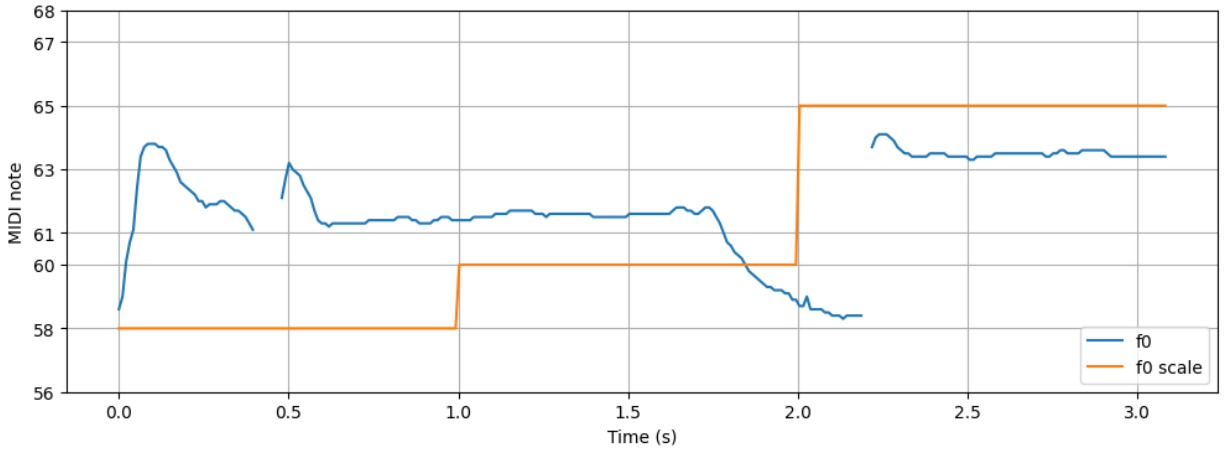


autotune-letitgo-bad.wav autotune-letitgo-bad.tuned.wav

Effetto pitch shifting - arbitrario

- Correzione dell'intonazione secondo valori arbitrari

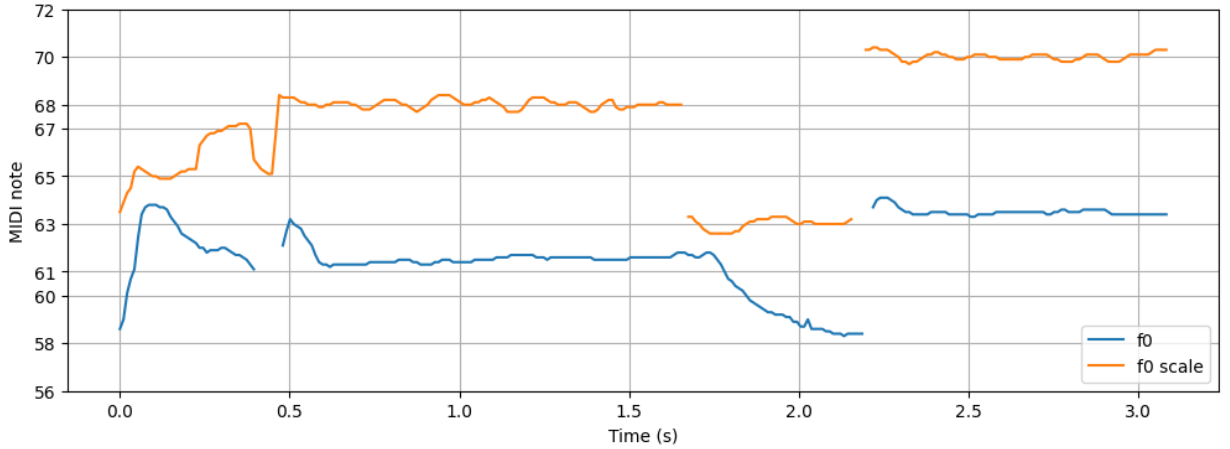
autotune-letitgo-bad.custom.wav



- Correzione dell'intonazione secondo valori "clonati"

autotune-letitgo-good.wav

autotune-letitgo-bad.clone.wav



Fasi dell'autotune

- 1. Identificazione del pitch
- 2. Mapping del pitch orginale sul nuovo pitch
  - ✓ Intonazione forzata
- 3. Trasposizione del pitch

Identificazione del pitch

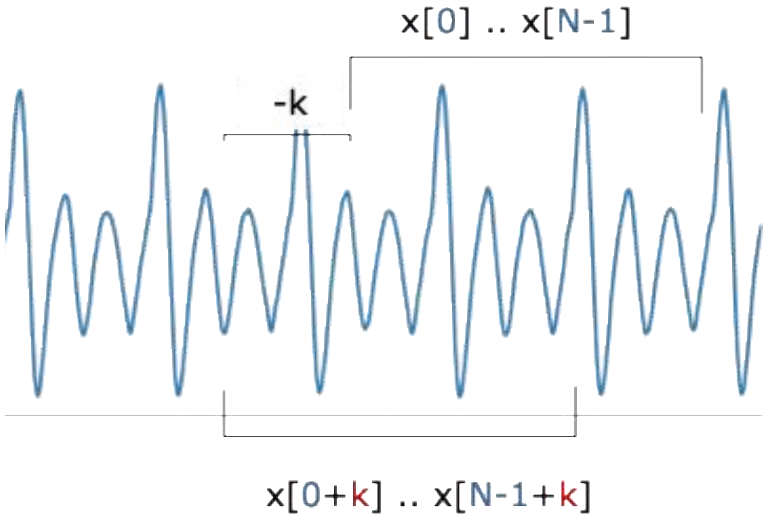
Identificazione del pitch (1/periodo)

- Periodo: "l'intervallo più lungo per cui il segnale si ripete uguale"
- L'identificazione del periodo consiste nel

selezionare un intervallo del segnale e
cercare a che distanza si trova la replica più simile

- Occorre introdurre un criterio di somiglianza (MSE)
  - ✓ Usiamo k<0
    perchè guardiamo nel passato

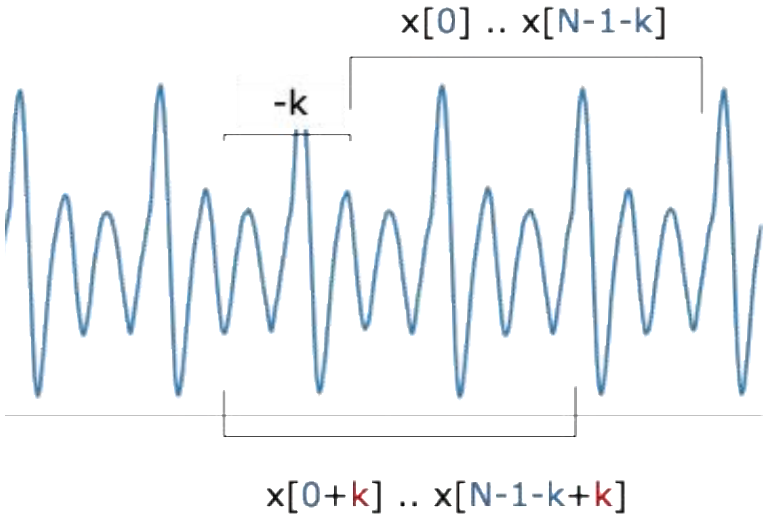
$$MSE[k] = \frac{1}{N}\sum_{n=0}^{N-1}(x[n] - x[n+k])^2$$



Identificazione del pitch (1/periodo)

- **Metodo della differenza, MSE (corretto)**
  - ✓ Criterio di somiglianza / differenza: MSE
  - ✓ Ritardo (-k) tra 0 e N
  - ✓ Normalizzazione <u>corretta</u>
    - Solo per la parte di segnale che si sovrappone (N-k) che equivale al numero di somme che vengono effettuate

$$MSE[k] = \frac{1}{N-k} \sum_{n=0}^{N-1-k} (x[n] - x[n+k])^2$$



Ricerca MSE minimo

$$MSE[k] = \frac{1}{N-k} \sum_{n=0}^{N-k-1} (x[n] - x[n+k])^2$$

- Occorre trovare il valore di **k** che minimizza l'MSE
  - ✓ Cioè il valore di k per cui la derivata dell'MSE si annulla
- Espandendo i calcoli e considerando un segnale stazionario otteniamo che l'MSE equivale a …

$$MSE[k] = \frac{1}{N-k} \sum_{n=0}^{N-k-1} (x^2[n] + x^2[n+k] - 2x[n]x[n+k])$$

$$MSE[k] = \frac{1}{N-k} \cdot \left[\sum_{n=0}^{N-k-1} x[n]x[n] + \sum_{n=0}^{N-k-1} x[n+k]x[n+k] - 2 \sum_{n=0}^{N-k-1} x[n]x[n+k] \right]$$

$$MSE[k] = \frac{1}{N-k} \cdot [R_{xx}[0] + R_{x_k x_k}[0] - 2 \cdot R_{xx}[k]]$$

Autocorrelazione (k)

$$R_{xx}[k] = \sum_{n=0}^{N-k-1} (x[n] \cdot x[n+k])$$

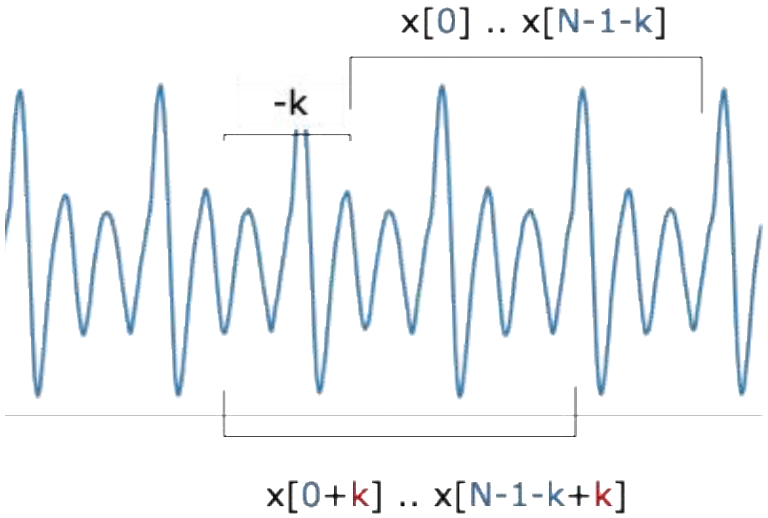
Ricerca MSE minimo

$$R_{xx}[k] = \sum_{n=0}^{N-k-1} (x[n] \cdot x[n+k])$$

- Introducendo l'autocorrelazione di x con ritardo k come $R_{xx}[k]$
- Se il segnale è stazionario l'autocorrelazione dipende solo dal ritardo k e non dal tempo assoluto: $R_{xx}[0]=R_{x_k x_k}[0]$

$$MSE[k] = \frac{1}{N-k} \cdot \left[R_{xx}[0] + R_{x_k x_k}[0] - 2 \cdot R_{xx}[k]\right] = \frac{2}{N-k} \cdot [R_{xx}[0] - R_{xx}[k]]$$

- Quindi l'MSE è minimo quando è massimo $R_{xx}[k]$
- Occorre individuare il ritardo/lag **k** per cui è massimo il prodotto tra x[n] e x[n+k]



Esempio: calcolo autocorrelazione

- Sia dato un segnale sig di lunghezza 10 campioni
- Dato un valore di k, e.g. -4
- Calcolare il prodotto tra ciascun campione di x[n] e x[n+(-k)]
- Sommare i valori ottenuti

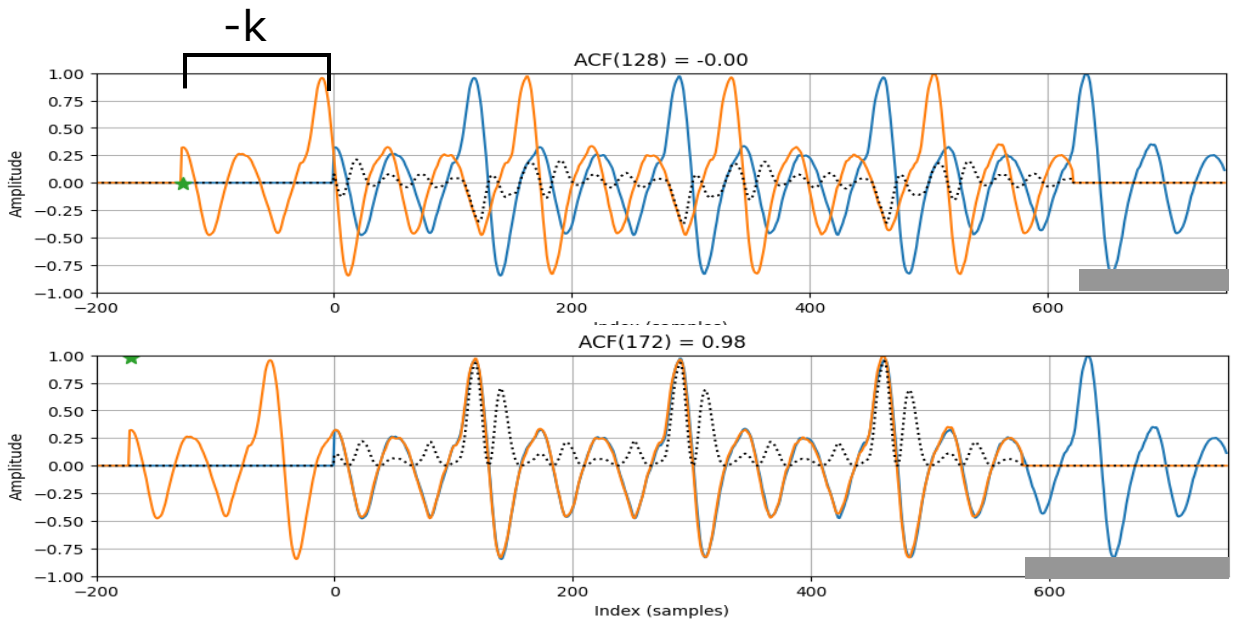
$$R_{xx}[k] = \sum_{n=0}^{N-k-1} (x[n] \cdot x[n+k])$$

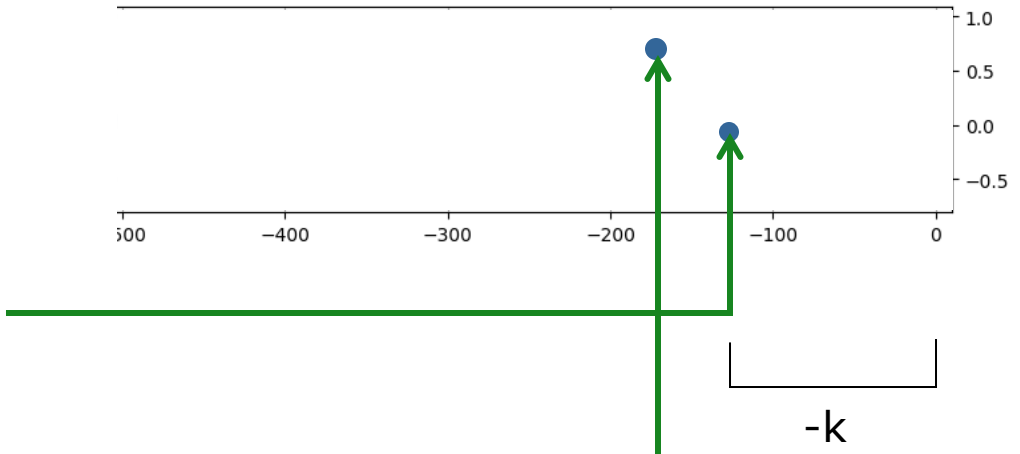
```
#       0123456789012345678 9
# sig   ----------********** => sig[:-k]
# sig_k ------**********---- => sig_k[k:]
# k     0987654321 0
```

Autocorrelazione

GSheet

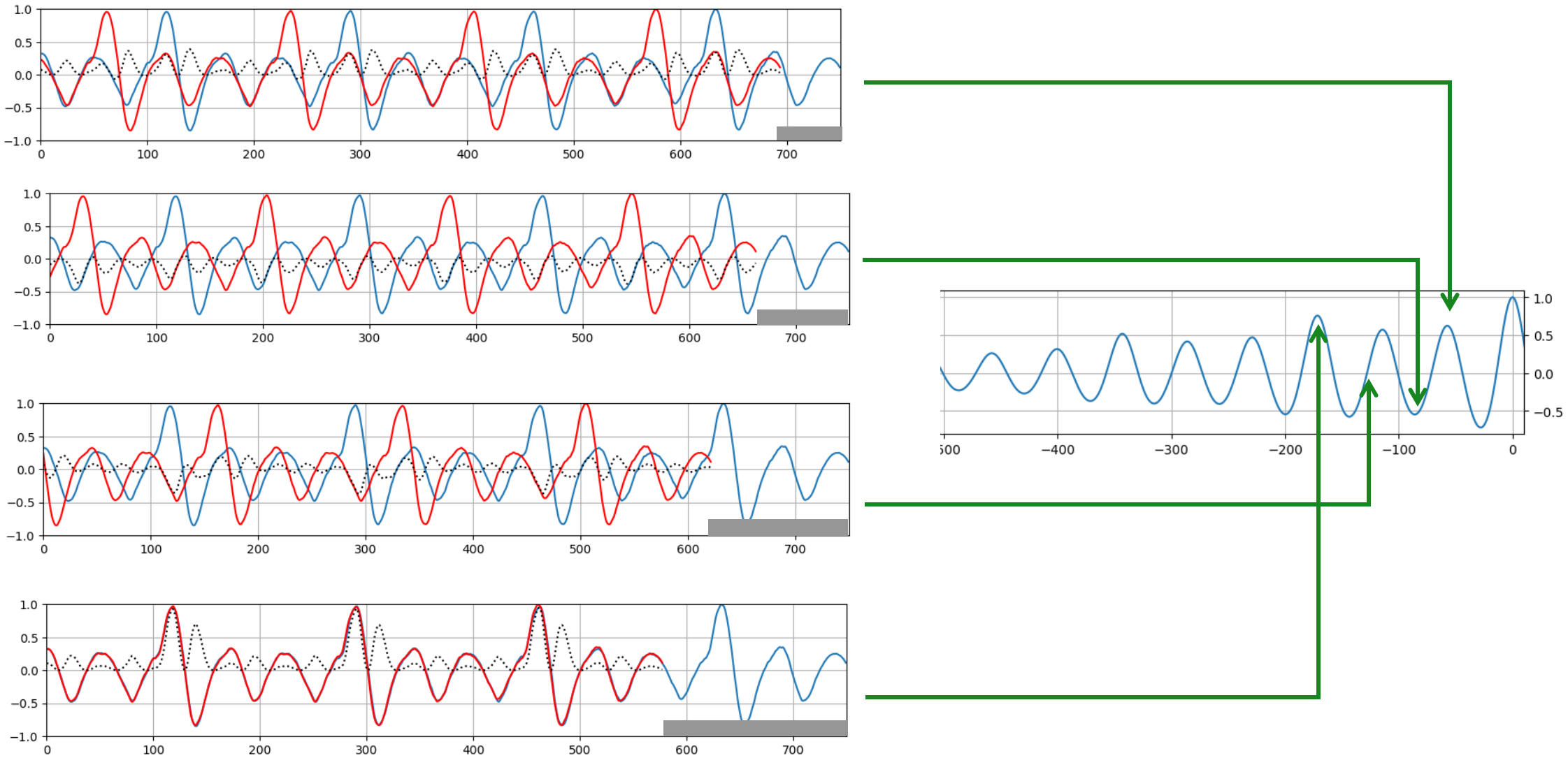
- Calcolo dell'autocorrelazione con lag -k: $R_{xx}[-k]$
  - ✓ Media del prodotto (tratteggiato) tra x[n] (blu) e x[n-k] (rosso) sulla finestra di analisi
- Da ripetere per diversi valori di k





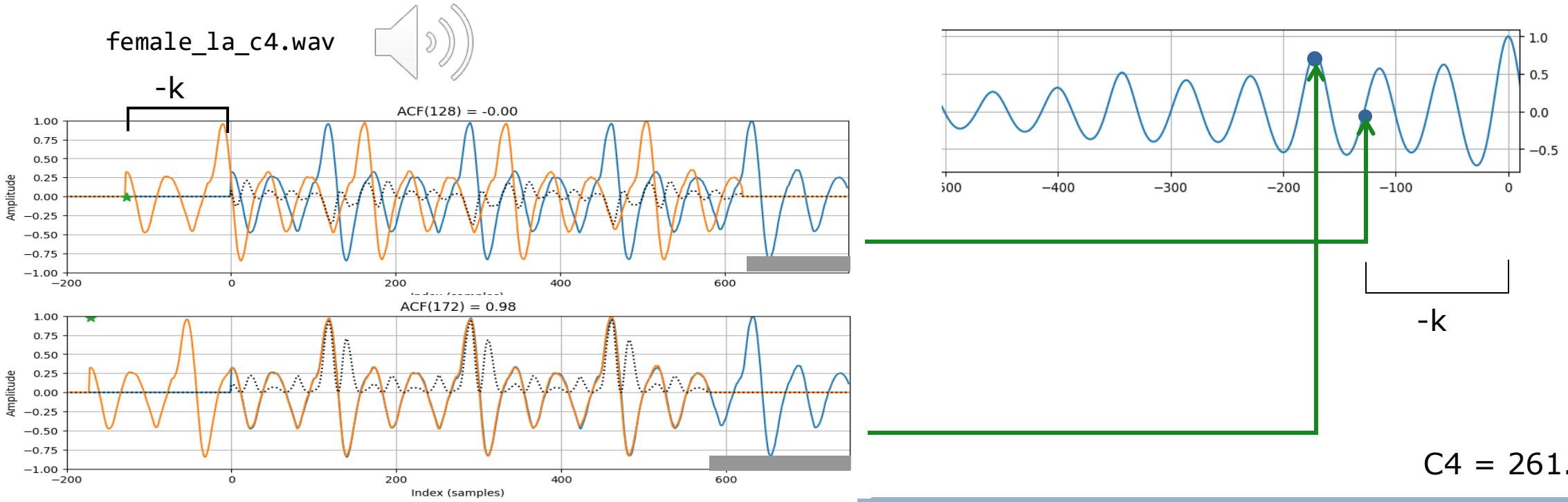
Autocorrelazione

56, 87, 128, 172



Autocorrelazione

- Il picco dell'autocorrelazione identifica il periodo T
- Il pitch è l'inverso del periodo 1/T
- Esempio: k = 172, fs = 44100 Hz, fs*1/k = 256,39 Hz

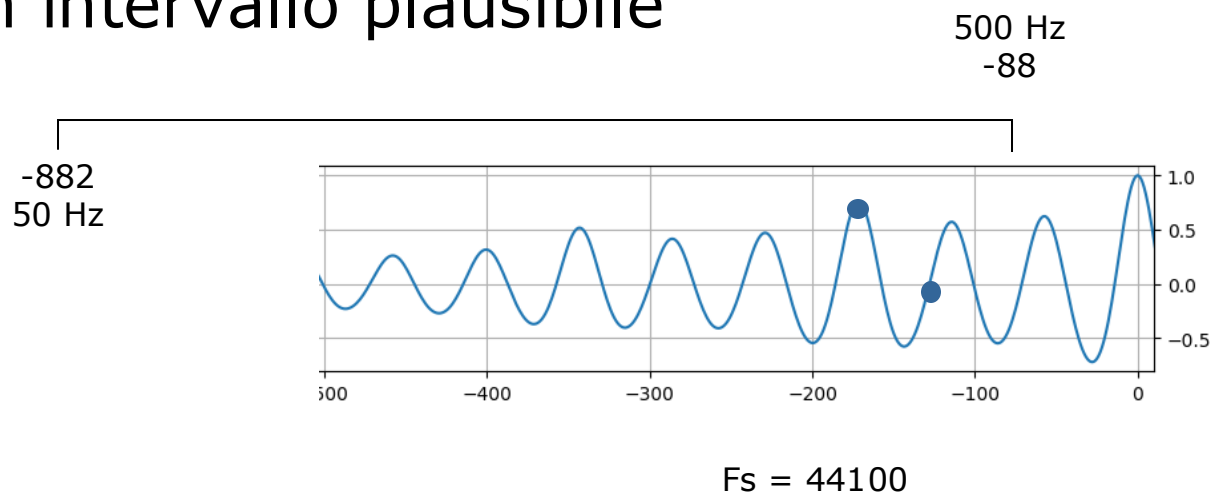


Identificazione del pitch - caveats

- Spesso si preferisce lavorare con l'autocorrelazione normalizzata, così, indipendentemente dal volume, il valore di picco sarà 1
- L'algoritmo non è infallibile, anzi …
- La presenza delle armoniche
  - ✓ Porta spesso a identificare valori che sono multipli del pitch effettivo perchè le armoniche creano picchi, talvolta più alti, a lag minori
  - ✓ E' convieniente definire un intervallo "ragionevole" in cui effettuare la ricerca del "lag" corretto
- Il segnale è spesso poco-stazionario
  - ✓ La ricerca del periodo, per essere corretta, richiede che questo si ripeta un certo numero di volte e non ci siano transitori

Pitch range

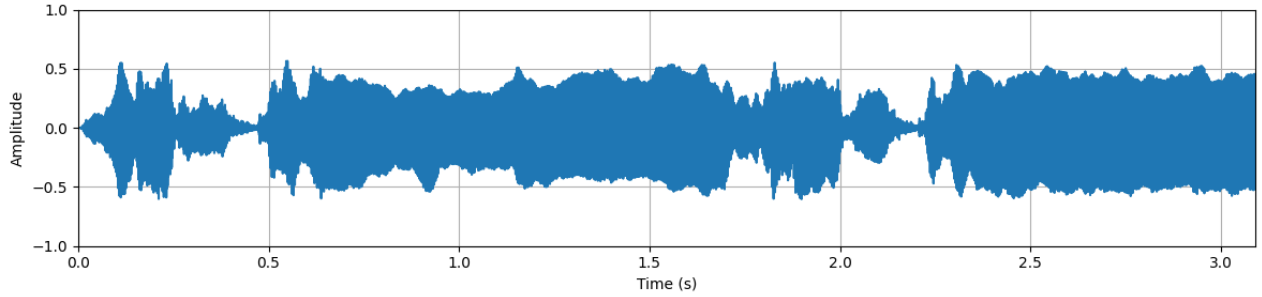
- L'autocorrelazione ha, per definizione, il massimo in k=0
  - ✓ Quando il segnale è moltiplicato per sè stesso
- E' buona norma non "prendere" qualsiasi massimo, ma ricercare il massimo in un intervallo plausibile
- Per la voce il pitch è
  - ✓ 50-150 Hz maschi adulti
  - ✓ 150-250 Hz femmine adulte
  - ✓ 300-500 Hz bambini
- A che ritardo corrisponde?
  - ✓ f_0 = Fs / k => k = Fs / F_0

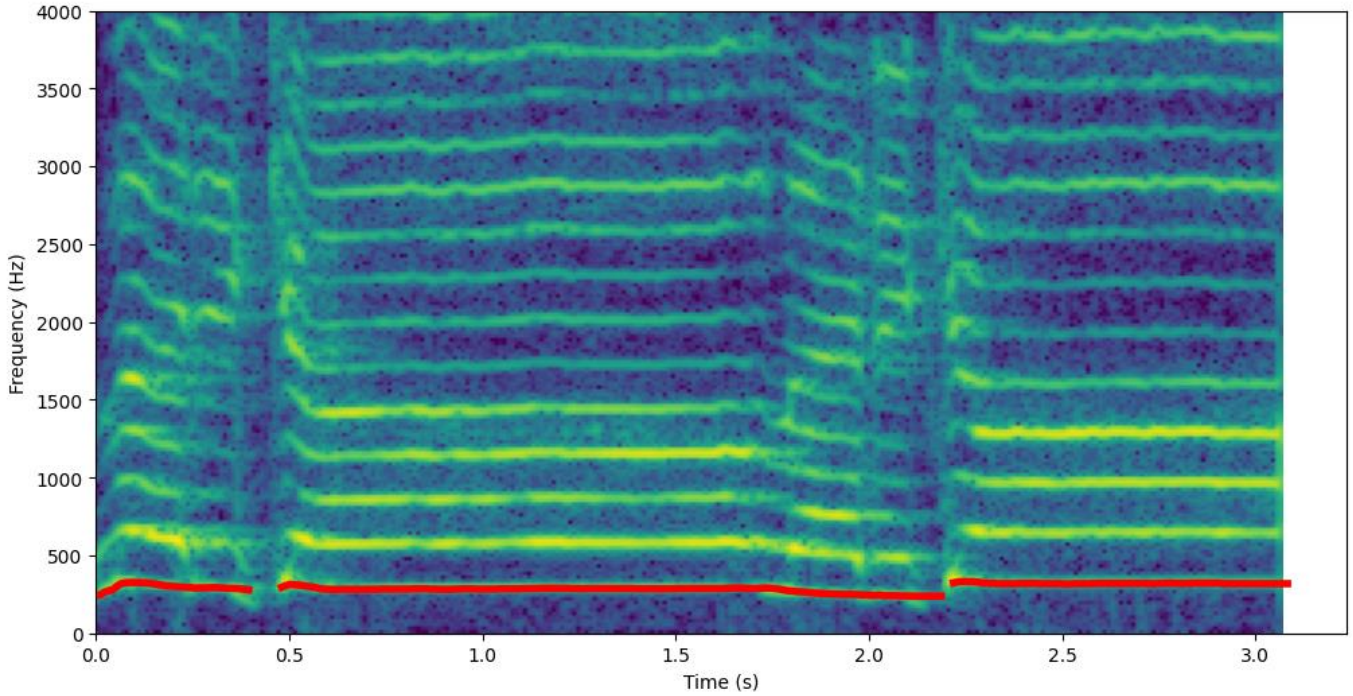


$$\frac{Fs}{f0_{max}} \leq k \leq \frac{Fs}{f0_{min}}$$

Esercizio

- autotune-letitgo-bad.wav

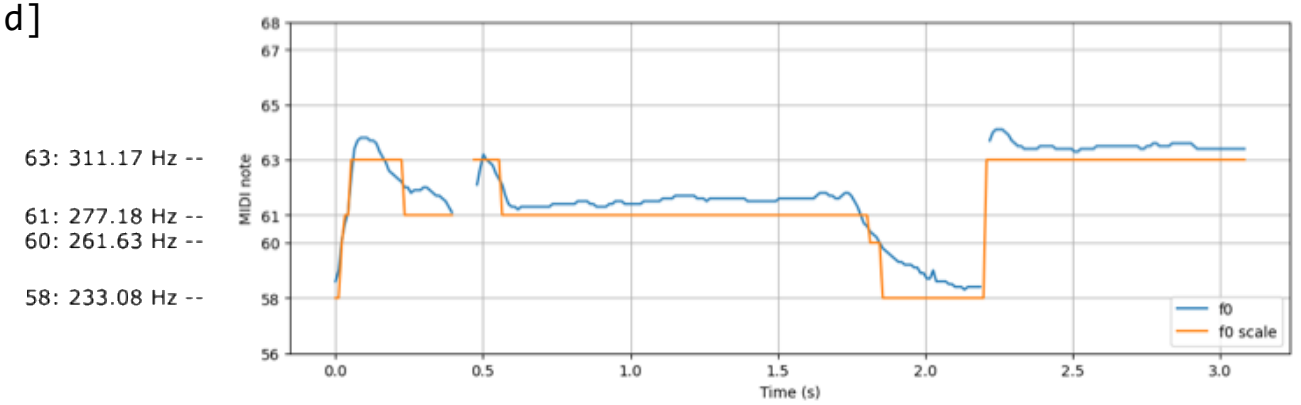




Mapping del pitch

Mapping del pitch

```
def pitch_to_scale(pitch, scale):
  # To properly perform pitch rounding to the nearest degree from the scale, we need to repeat
  # the first degree raised by an octave. Otherwise, pitches slightly lower than the base degree
  # would be incorrectly assigned.
  degrees = librosa.key_to_degrees(scale)
  degrees = np.concatenate( (degrees, [degrees[0] + 12]) ) # add next octave key degree
  midi_note = librosa.hz_to_midi(pitch)
  # Subtract the multiplicities of 12 so that we have the real-valued pitch class of the input pitch.
  degree = midi_note % 12
  # Find the closest pitch class from the scale.
  degree_id = np.argmin(np.abs(degrees - degree))
  # Calculate the difference between the input pitch class and the desired pitch class.
  degree_difference = degree - degrees[degree_id]
  # Shift the input MIDI note number
  # by the calculated difference.
  midi_note -= degree_difference
  # print('midi_note', midi_note)
  # Convert to Hz.
  return librosa.midi_to_hz(midi_note)
```



Trasposizione del pitch

Trasposizione del pitch

- L'obiettivo della *trasposizione del pitch* è quello di modificare l'altezza fondamentale f0 senza alterare la durata del segnale
- Esistono diversi algoritmi
  - ✓ Nel dominio del tempo: PSOLA e varianti
  - ✓ Nel dominio della frequenza: Phase Vocoder e varianti (*)
  - ✓ Basati su modelli del segnale (analisi e ri-sintesi)
    - Sine plus noise modeling
    - LPC vocoder (eccitazione + filtro)
    - Reti neurali

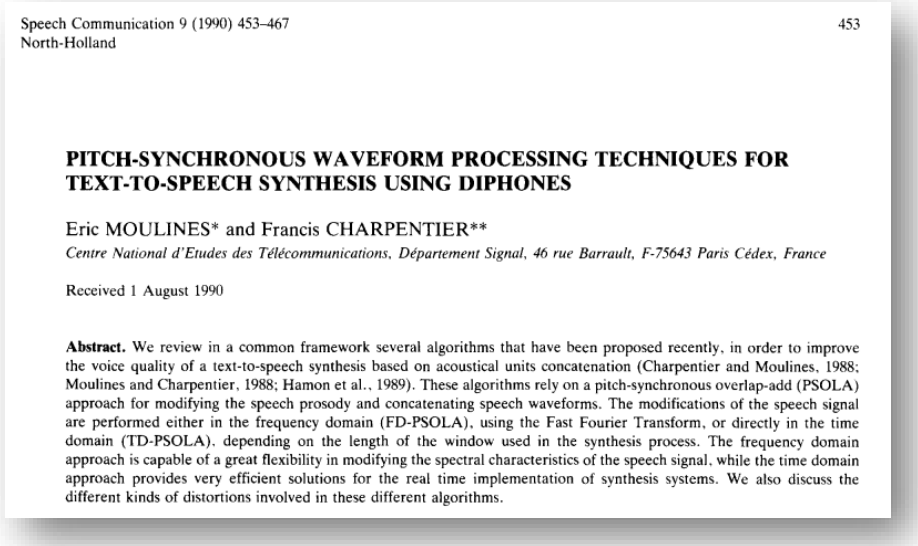
N.B. Diventano problematici se "c'è più di un pitch"

(*) altera le formanti, effetto chipmunk

Time-domain Pitch-Synchronous OLA

- Approccio di pitch shifting implementato nel dominio del tempo
- Si basa sull'identificazione di intervalli del segnale "sincroni" con il pitch (periodo T) del segnale originale (cioè dei periodi)
- E sulla copia degli stessi a distanza T' nel segnale sintetizzato per creare l'effetto pitch trasposto 1/T'

- E. Moulines, F. Charpentier,
  "Pitch synchronous waveform processing techniques for text-to-speech synthesis using diphones",
  Speech Communications, 1990



Speech Communication 9 (1990) 453–467 453
North-Holland

PITCH-SYNCHRONOUS WAVEFORM PROCESSING TECHNIQUES FOR TEXT-TO-SPEECH SYNTHESIS USING DIPHONES

Eric MOULINES* and Francis CHARPENTIER**

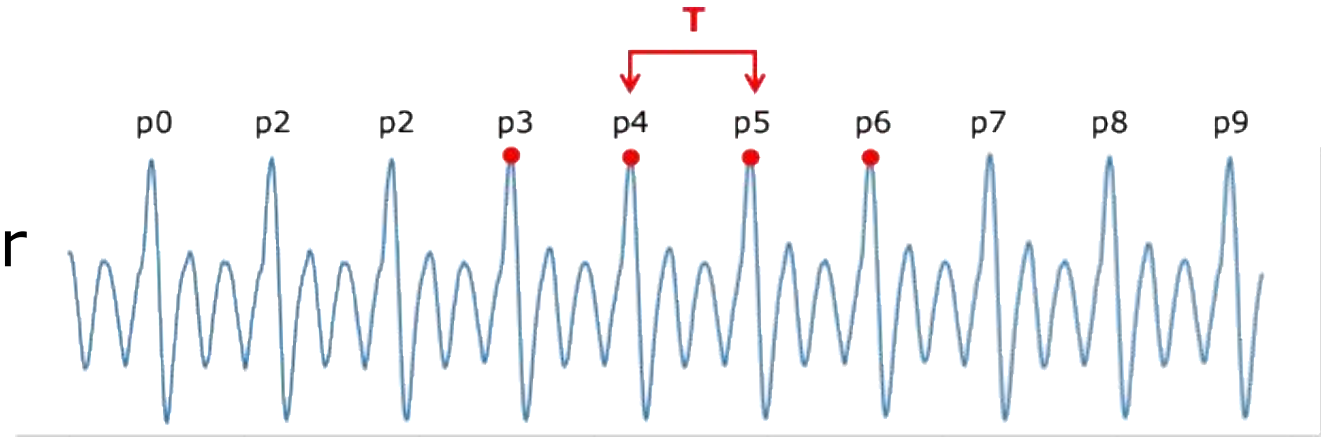
Centre National d'Etudes des Télécommunications, Département Signal, 46 rue Barrault, F-75643 Paris Cédex, France

Received 1 August 1990

Abstract. We review in a common framework several algorithms that have been proposed recently, in order to improve the voice quality of a text-to-speech synthesis based on acoustical units concatenation (Charpentier and Moulines, 1988; Moulines and Charpentier, 1988; Hamon et al., 1989). These algorithms rely on a pitch-synchronous overlap-add (PSOLA) approach for modifying the speech prosody and concatenating speech waveforms. The modifications of the speech signal are performed either in the frequency domain (FD-PSOLA), using the Fast Fourier Transform, or directly in the time domain (TD-PSOLA), depending on the length of the window used in the synthesis process. The frequency domain approach is capable of a great flexibility in modifying the spectral characteristics of the speech signal, while the time domain approach provides very efficient solutions for the real time implementation of synthesis systems. We also discuss the different kinds of distortions involved in these different algorithms.

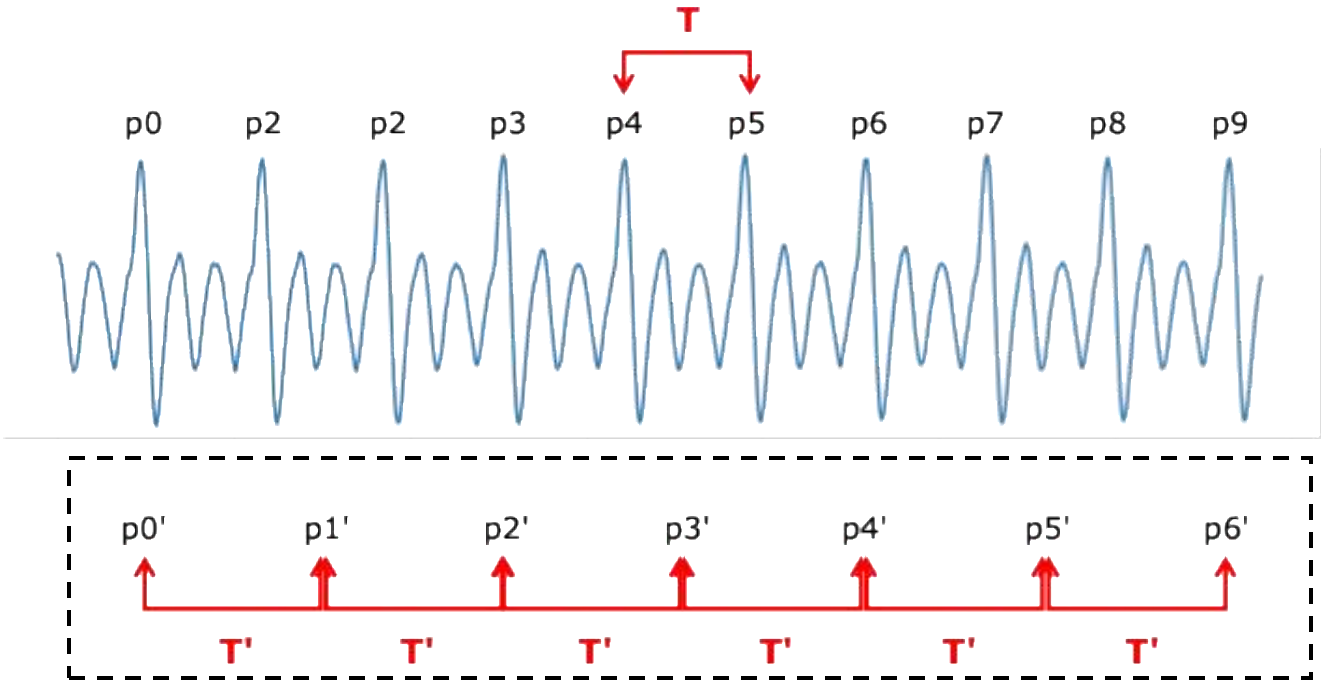
TD-PSOLA

- Identificazione del pitch originale: 1/T
- Identificazione dei picchi per ogni periodo del segnale originale



TD-PSOLA

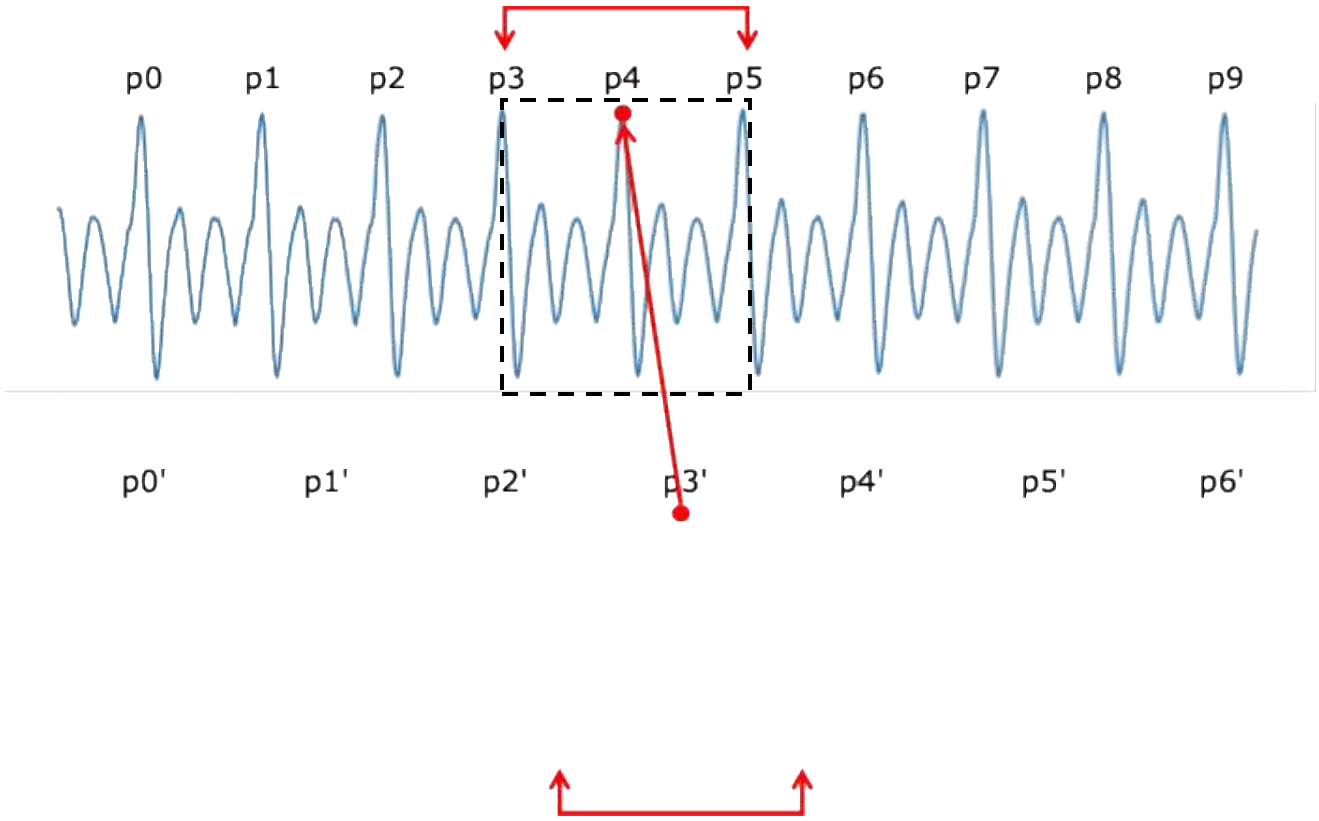
- Definizione
  del pitch target: 1/T'
- Identificazione dei nuovi picchi del segnale trasposto
  - ✓ Se T'>T saranno in numero minore, spaziati di T'



TD-PSOLA – overlap/add

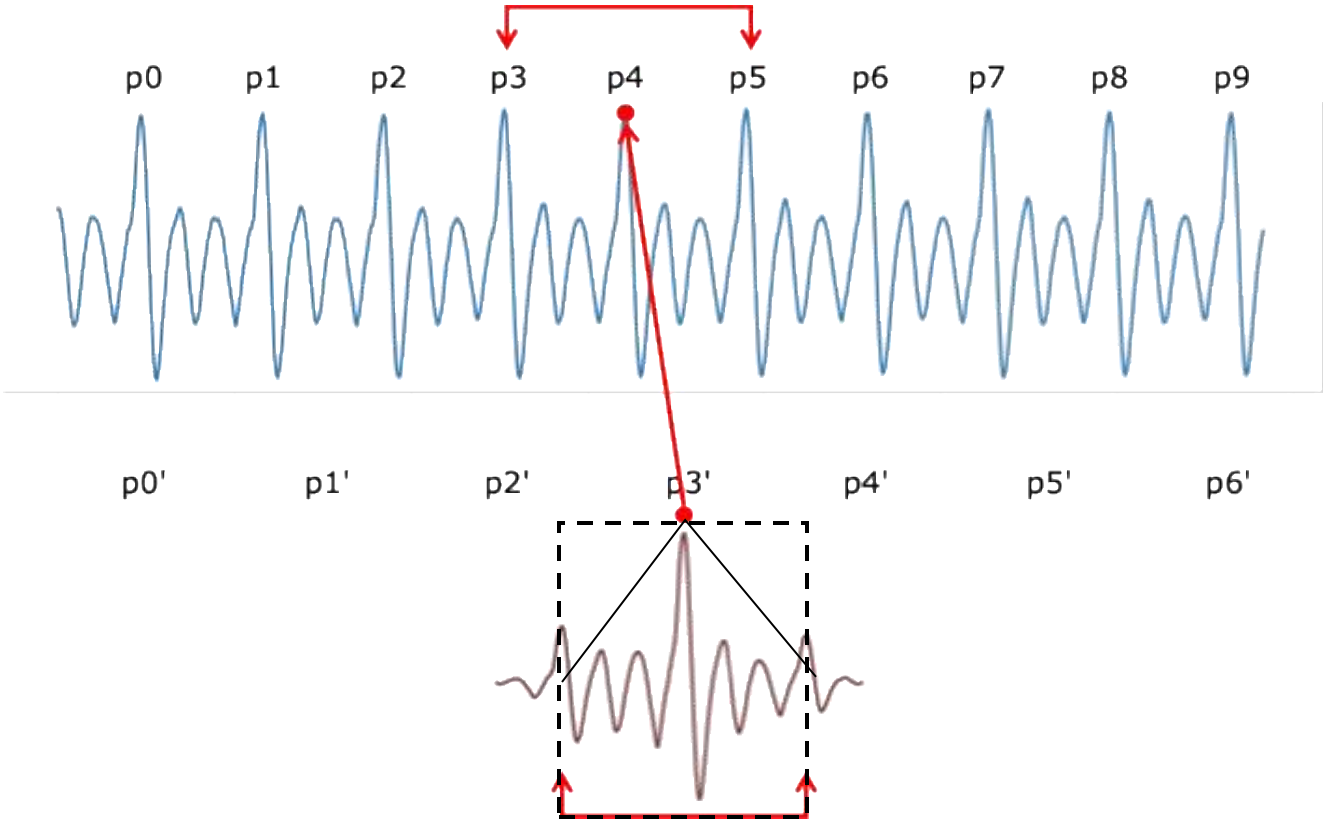
Come viene creato il nuovo segmento centrato in p'[i]?

- Viene identificato il picco p[j] più vicino del segnale originale
- Da tale picco si estrae un segmento di due periodi



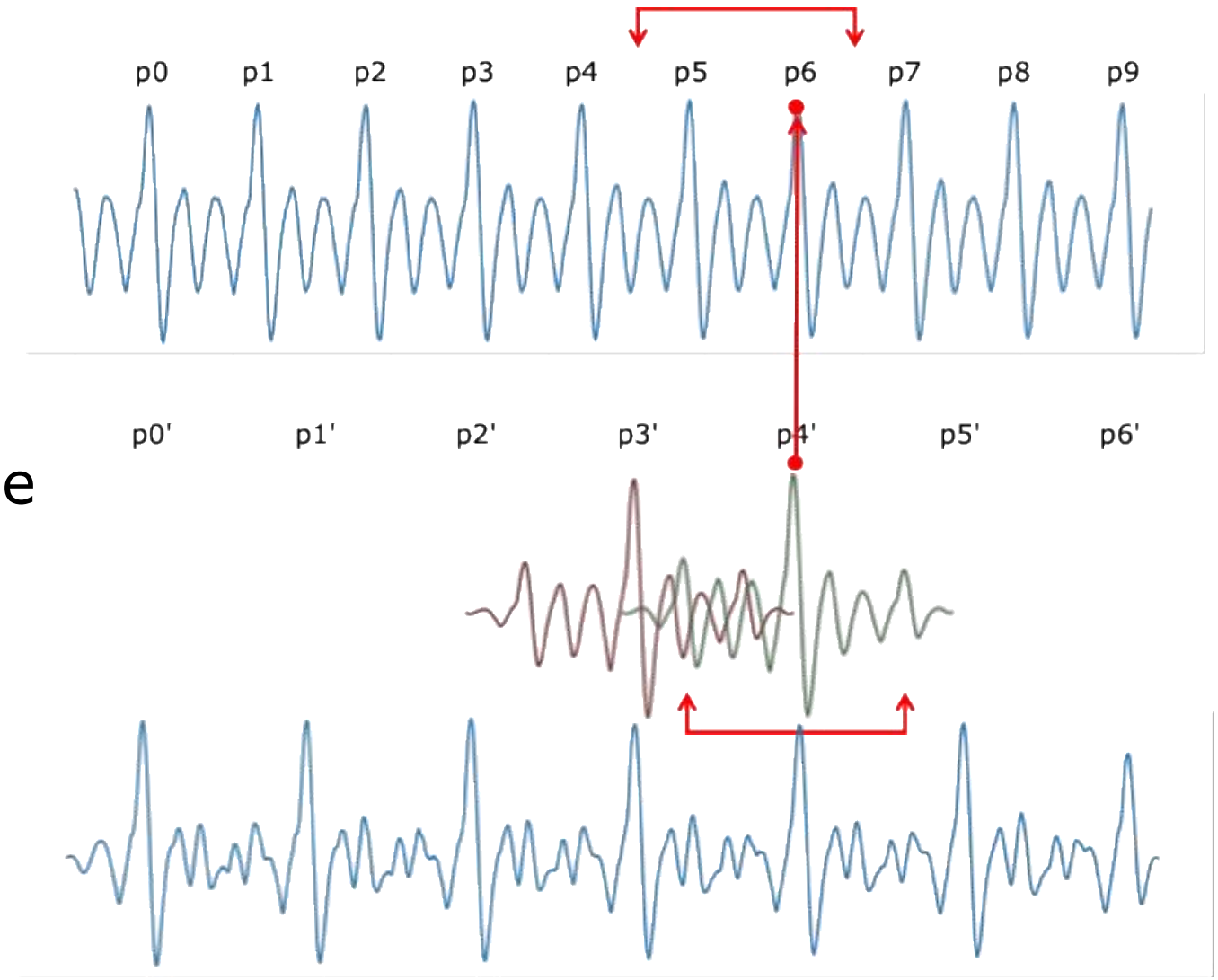
TD-PSOLA – overlap/add

- Il segmento viene pesato con una finestra triangolare
- E sommato al segnale in uscita nella posizione centrata in p[i]'



TD-PSOLA – overlap/add

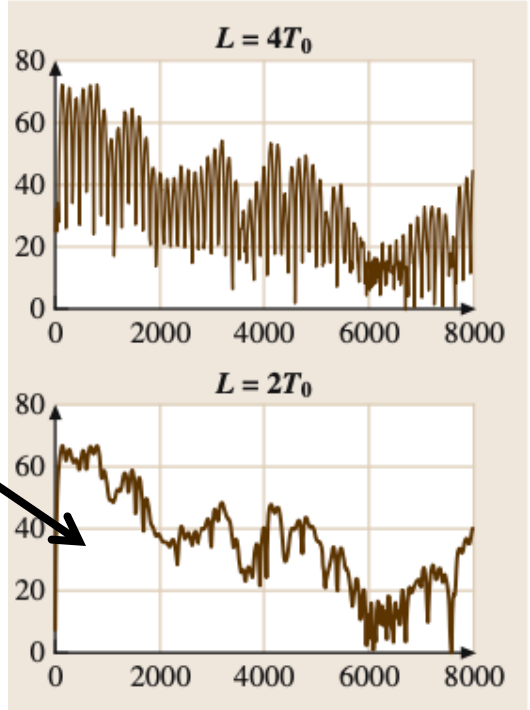
- Il tutto si ripete anche per il picco successivo
- Le operazioni successive di somma portano alla creazione del nuovo segnale con i picchi spaziati T'



TD-PSOLA - Limiti

Tutt'altro che scontato

- Serve la presenza di un pitch
  e capacità di identificarne correttamente i periodi
  - ✓ Voce umana o strumento singolo
  - ✓ Algoritmo del pitch robusto (~)
- Mantenimento del timbro
  - ✓ Utilizzo di segmenti L=2T, FFT circa inviluppo
  - ✓ Centratura sugli impulsi della glottide (~)
- Caveats
  - ✓ Ratio comprese tra 0.5 e 2 per la voce
  - ✓ Attenzione a porzioni non vocalizzate e transitori



Reference: Zolzer, "DAFX: Digital Audio Effects"

Esercizio

- Trasposizione del pitch
  - ✓ Custom
  - ✓ Autotune
  - ✓ Re-tune