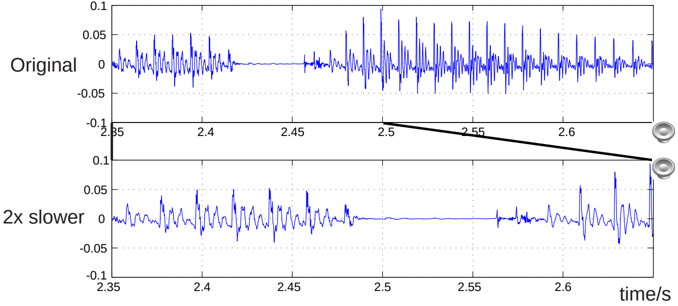
E85.2607: Lecture 6 – Time-segment Processing

Variable speed replay

How can we modify a sound to make 'faster' or 'slower'?
i.e. song played at half tempo
Why not just slow it down?

- $x_s(t) = x_o(vt)$, v = speedup factor ($> 1 \rightarrow$ faster)
- equivalent to playback at a different sampling rate (`resample` in Matlab)

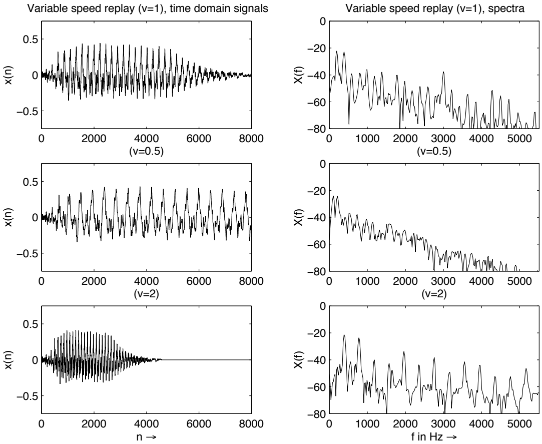


$v = \frac{1}{2}$

Source: Dan Ellis, EE6820 Lecture 1

Variable speed replay in the frequency domain

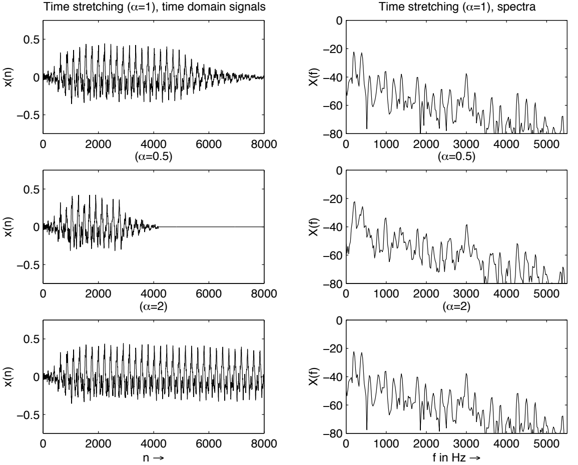
- Compress in time → shift frequencies higher
  - “chipmunk” effect
- Beware aliasing
  - LPF before resampling



Source: DAFX: Fig. 7.1

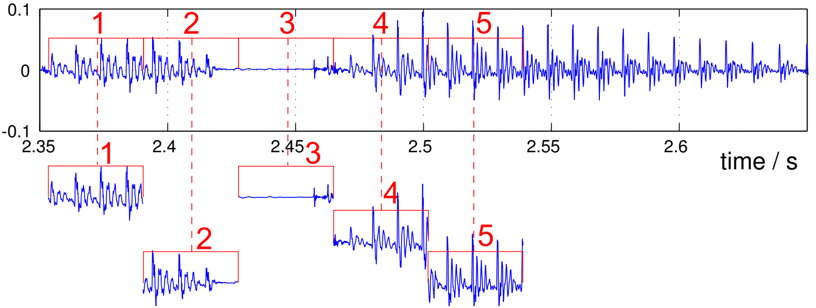
Timescale modification

- Want to change time axis
- Leave pitch alone



Source: DAFX: Fig. 7.4

Block processing



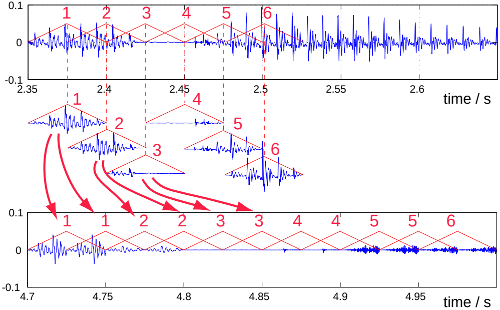
- Chop signal into short segments
- Resample individual segments, but keep time axis
  - Will this work?

```
c = 1;
for i = 0:N:length(x)
  y(:,c) = x(i + [1:N]);
  c = c + 1;
end
```

TSM in the time-domain

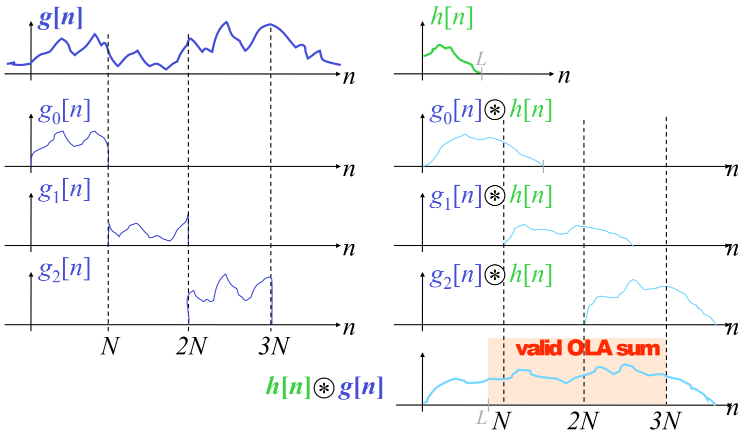
- Problem: want to preserve local time structure but alter global time structure
- Repeat or discard segments
  - but: artifacts from abrupt edges
- Cross-fade & overlap-add (OLA)

$$y^m[mL+n] = y^{m-1}[mL+n] + w[n] \cdot x\left[\lfloor vm \rfloor L + n\right]$$



Source: Dan Ellis, EE6820 Lecture 1

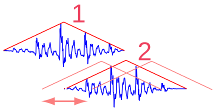
Aside: Fast convolution (`fftfilt`)



Source: Dan Ellis, EE4810: Lecture 3

Use FFT to implement block-wise convolution: $O(N \log N)$ vs $O(N^2)$

Idea: allow some leeway in placing window to optimize alignment of waveforms to minimize artifacts



K_m maximizes alignment of 1 and 2

Hence,

$$y^m[mL+n] = y^{m-1}[mL+n] + w[n] \cdot x\left[\lfloor vm \rfloor L + n + K_m\right]$$

Where K_m chosen by cross-correlation (`xcorr`):

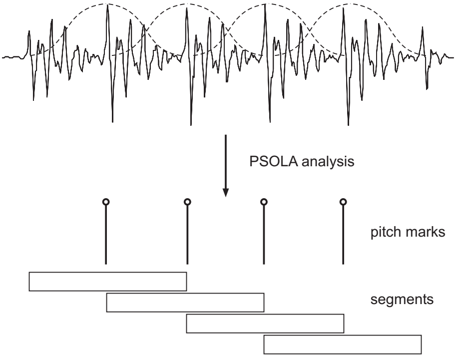
$$K_m = \underset{0 \le K \le K_u}{\operatorname{argmax}} \frac{\sum_{n=0}^{N_{ov}} y^{m-1}[mL+n] \cdot x\left[\lfloor vm \rfloor L + n + K\right]}{\sqrt{\sum (y^{m-1}[mL+n])^2 \sum (x\left[\lfloor vm \rfloor L + n + K\right])^2}}$$

Source: Dan Ellis, EE6820 Lecture 1

Pitch-synchronous OLA (PSOLA): Analysis

Assuming input signal has pitch, can get even cleaner TSM

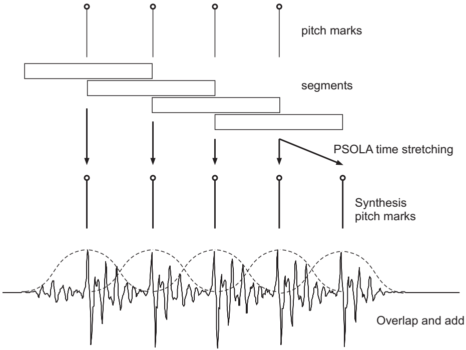
1. Identify pitch period T using auto-correlation
2. Find peaks corresponding to pitch pulse
3. Segment signal using length $2T$ window centered on each pitch pulse



Source: DAFX: Fig. 7.9

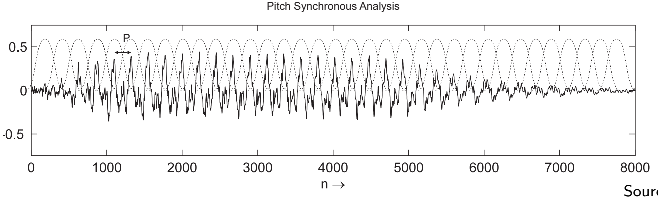
PSOLA: Synthesis

1. Set up grid of pitch marks for output segments
2. Match up closest input segment to each output pitch mark
3. Overlap-add...



Source: DAFX: Fig. 7.10

PSOLA gotchas

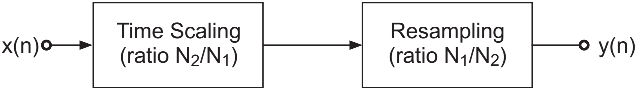


Source: DAFX: Fig. 7.8

- PSOLA analysis assumes constant pitch
  - Use more sophisticated pitch tracker
- What if there is no pitch?
  - Fall back to OLA/SOLA
  - Complicates analysis – how to decide whether given section has pitch or not?
- Tends to introduce artifacts on noisy sections
  - Repeating noisy segments introduces artificial periodicity (e.g. repeated transients)
  - Fix by reversing repeated segments

Pitch-shifting

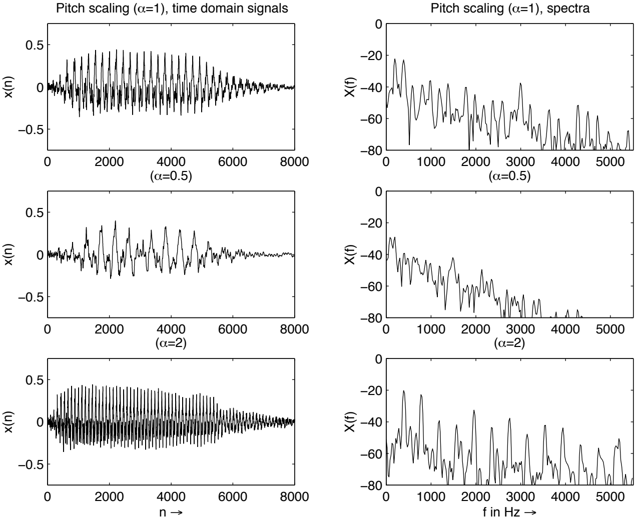
- Leave the time-axis alone, but shift frequency-axis
- Approach: Resample to shift both, then correct time axis
- or vice-versa



Source: DAFX: Fig. 7.13

$$v = \frac{N_2}{N_1}$$

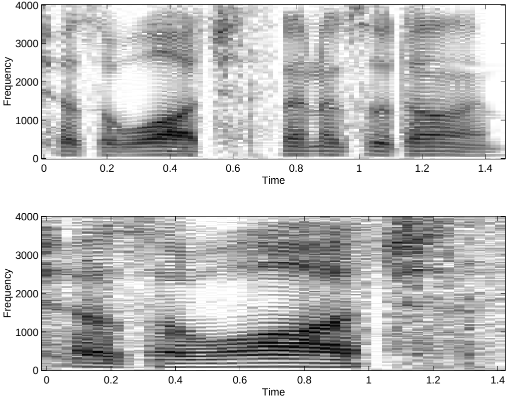
Pitch-shifting example



Source: DAFX: Fig. 7.12

TSM with the Spectrogram

Just stretch out the spectrogram?



how to resynthesize?

spectrogram is only $|Y[k, m]|$

Source: Dan Ellis, EE6820 Lecture 1

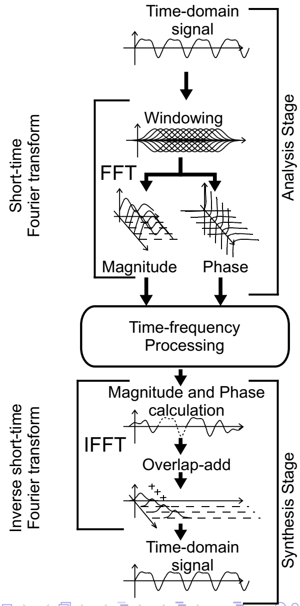
A look ahead: Phase vocoder

- Time/pitch modification in the time-frequency domain
- Like OLA, but also take DFT of each segment
  - Like STFT, but special attention paid to phase...
- Magnitude from 'stretched' spectrogram:

$$|Y[k,m]| = |X[k,vr]|$$

- But preserve phase increment between slices:

$$\dot{\theta}_Y[k,m] = \dot{\theta}_X[k,vm]$$



Reading

DAFX 7.1–7.4 TSM, SOLA, PSOLA

DAFX 8.1–8.2 Phase vocoder basics