

Digital Signal Processing for Music

Part 22: Time-stretching and Pitch-shifting

alexander lerch

time stretching & pitch shifting

introduction

■ time stretching

change playback speed/tempo without changing pitch

■ pitch shifting

change pitch without changing tempo/playback speed

■ terms

- time/pitch scaling
- time expansion/compression

time stretching & pitch shifting

introduction

Georgia Tech | Center for Music Technology
College of Design

■ time stretching

change playback speed/tempo without changing pitch

■ pitch shifting

change pitch without changing tempo/playback speed

■ terms

- time/pitch scaling
 - time expansion/compression

time stretching & pitch shifting

introduction

Georgia Tech Center for Music Technology
College of Design

■ time stretching

change playback speed/tempo without changing pitch

■ pitch shifting

change pitch without changing tempo/playback speed

■ terms

- time/pitch scaling
 - time expansion/compression

time stretching & pitch shifting

applications

- **beat matching:** align tempo of two or more audio files (mashup)
- **key lock:** “align” pitch of two or more audio files (mashup)
- **pitch/time correction:** edit intonation, frequency deviation, vibrato, glissando
- **video frame rate conversion**
- **sample player/libraries**
- **sound design**
- **educational software:** pitch and timing visualization

time stretching & pitch shifting applications

- **beat matching:** align tempo of two or more audio files (mashup)
- **key lock:** “align” pitch of two or more audio files (mashup)
- **pitch/time correction:** edit intonation, frequency deviation, vibrato, glissando
- **video frame rate conversion**
- **sample player/libraries**
- **sound design**
- **educational software:** pitch and timing visualization

time stretching & pitch shifting applications

- **beat matching:** align tempo of two or more audio files (mashup)
- **key lock:** “align” pitch of two or more audio files (mashup)
- **pitch/time correction:** edit intonation, frequency deviation, vibrato, glissando
- video frame rate conversion
- sample player/libraries
- sound design
- educational software: pitch and timing visualization

time stretching & pitch shifting applications

- **beat matching:** align tempo of two or more audio files (mashup)
- **key lock:** “align” pitch of two or more audio files (mashup)
- **pitch/time correction:** edit intonation, frequency deviation, vibrato, glissando
- **video frame rate conversion**
- sample player/libraries
- sound design
- educational software: pitch and timing visualization

time stretching & pitch shifting applications

- **beat matching:** align tempo of two or more audio files (mashup)
- **key lock:** “align” pitch of two or more audio files (mashup)
- **pitch/time correction:** edit intonation, frequency deviation, vibrato, glissando
- **video frame rate conversion**
- **sample player/libraries**
- **sound design**
- **educational software:** pitch and timing visualization

time stretching & pitch shifting applications

- **beat matching:** align tempo of two or more audio files (mashup)
- **key lock:** “align” pitch of two or more audio files (mashup)
- **pitch/time correction:** edit intonation, frequency deviation, vibrato, glissando
- **video frame rate conversion**
- **sample player/libraries**
- **sound design**
- **educational software:** pitch and timing visualization

time stretching & pitch shifting

applications

- **beat matching:** align tempo of two or more audio files (mashup)
- **key lock:** “align” pitch of two or more audio files (mashup)
- **pitch/time correction:** edit intonation, frequency deviation, vibrato, glissando
- **video frame rate conversion**
- **sample player/libraries**
- **sound design**
- **educational software:** pitch and timing visualization

time stretching & pitch shifting

stretch and pitch factors

$$s = \frac{t_{output}}{t_{input}}$$
$$p = \frac{f_{output}}{f_{input}}$$

examples:

- half speed: $s = 2$
- half pitch: $p = \frac{1}{2}$
- semitone up/down:

$$p_u = 2^{1/12} = 1.059 \quad p_d = 2^{-1/12} = 0.9439$$

- 100 BPM → 75 BPM: $s = \frac{4}{3}$

time stretching & pitch shifting

stretch and pitch factors

$$s = \frac{t_{output}}{t_{input}}$$
$$p = \frac{f_{output}}{f_{input}}$$

examples:

- *half speed*: $s = 2$
- *half pitch*: $p = \frac{1}{2}$
- *semitone up/down*:

$$p_u = 2^{1/12} = 1.059 \quad p_d = 2^{-1/12} = 0.9439$$

- *100 BPM → 75 BPM*: $s = \frac{4}{3}$

time stretching & pitch shifting

stretch and pitch factors

$$s = \frac{t_{output}}{t_{input}}$$
$$p = \frac{f_{output}}{f_{input}}$$

examples:

- half speed: $s = 2$
- half pitch: $p = \frac{1}{2}$
- semitone up/down:

$$p_u = 2^{1/12} = 1.059 \quad p_d = 2^{-1/12} = 0.9439$$

- 100 BPM → 75 BPM: $s = \frac{4}{3}$

time stretching & pitch shifting

stretch and pitch factors

$$s = \frac{t_{output}}{t_{input}}$$
$$p = \frac{f_{output}}{f_{input}}$$

examples:

- half speed: $s = 2$
- half pitch: $p = \frac{1}{2}$
- semitone up/down:

$$p_u = 2^{1/12} = 1.059 \quad p_d = 2^{-1/12} = 0.9439$$

- 100 BPM → 75 BPM: $s = \frac{4}{3}$

time stretching & pitch shifting

stretch and pitch factors

$$s = \frac{t_{output}}{t_{input}}$$
$$p = \frac{f_{output}}{f_{input}}$$

examples:

- half speed: $s = 2$
- half pitch: $p = \frac{1}{2}$
- semitone up/down:

$$p_u = 2^{1/12} = 1.059 \quad p_d = 2^{-1/12} = 0.9439$$

- 100 BPM → 75 BPM: $s = \frac{4}{3}$

time stretching & pitch shifting

stretch and pitch factors

$$s = \frac{t_{output}}{t_{input}}$$
$$p = \frac{f_{output}}{f_{input}}$$

examples:

- half speed: $s = 2$
- half pitch: $p = \frac{1}{2}$
- semitone up/down:

$$p_u = 2^{1/12} = 1.059 \quad p_d = 2^{-1/12} = 0.9439$$

- 100 BPM → 75 BPM: $s = \frac{4}{3}$

time stretching & pitch shifting

stretch and pitch factors

Georgia Tech | Center for Music Technology
College of Design

$$s = \frac{t_{output}}{t_{input}}$$

$$p = \frac{f_{output}}{f_{input}}$$

examples:

- *half speed*: $s = 2$
 - *half pitch*: $p = \frac{1}{2}$
 - *semitone up/down*:

$$p_u = 2^{1/12} = 1.059 \quad p_d = 2^{-1/12} = 0.9439$$

- $100 \text{ BPM} \rightarrow 75 \text{ BPM}$: $s = \frac{4}{3}$

time stretching & pitch shifting

Georgia Tech | Center for Music Technology
College of Design

$$s = \frac{t_{output}}{t_{input}}$$

$$p = \frac{f_{output}}{f_{input}}$$

examples:

- *half speed*: $s = 2$
 - *half pitch*: $p = \frac{1}{2}$
 - *semitone up/down*:

$$p_\mu = 2^{1/12} = 1.059 \quad p_d = 2^{-1/12} = 0.9439$$

- $100 \text{ BPM} \rightarrow 75 \text{ BPM}$: $s = \frac{4}{3}$

time stretching & pitch shifting

stretch and pitch factors

Georgia Tech Center for Music Technology
College of Design

$$s = \frac{t_{output}}{t_{input}}$$

$$p = \frac{f_{output}}{f_{input}}$$

examples:

- *half speed*: $s = 2$
 - *half pitch*: $p = \frac{1}{2}$
 - *semitone up/down*:

$$p_u = 2^{1/12} = 1.059 \quad p_d = 2^{-1/12} = 0.9439$$

- $100 \text{ BPM} \rightarrow 75 \text{ BPM}$: $s = \frac{4}{3}$

time stretching & pitch shifting resampling

■ traditional: resampling

- change inter-sample 'distance' by **interpolation**
- keep **playback sample rate constant**
- audio example



⇒ tempo change results in pitch change (and vice versa)

$$s = \frac{1}{p}$$

time stretching & pitch shifting resampling

■ traditional: resampling

- change inter-sample 'distance' by **interpolation**
- keep **playback sample rate constant**
- audio example
 - ▶ original
 - ▶ resample

⇒ tempo change results in pitch change (and vice versa)

$$s = \frac{1}{p}$$

time stretching & pitch shifting resampling

■ traditional: resampling

- change inter-sample 'distance' by **interpolation**
- keep **playback sample rate constant**

- audio example

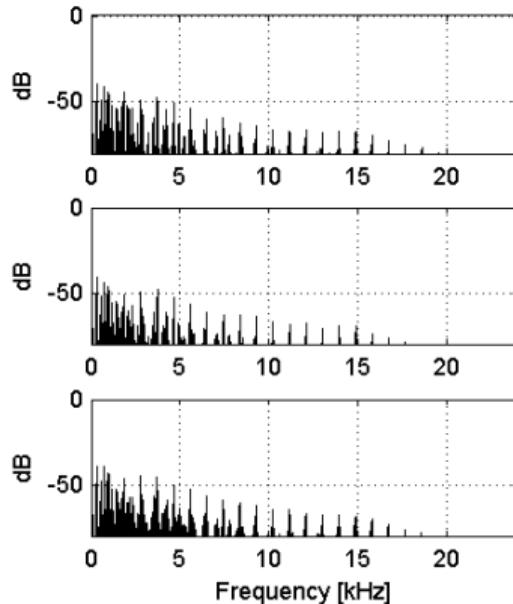
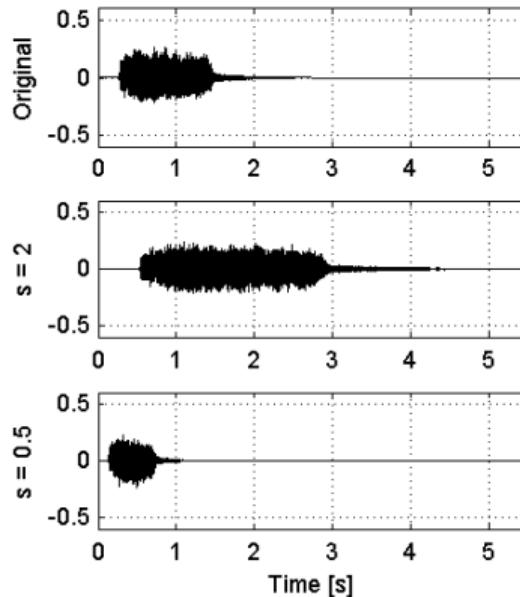
- ▶ original
- ▶ resample

⇒ tempo change results in pitch change (and vice versa)

$$s = \frac{1}{p}$$

time stretching & pitch shifting

stretching: frequency domain



OLA

introduction

overlap and add approaches for, e.g.,

- granular synthesis
- time/frequency synthesis and processing
- time-stretching and pitch-shifting

OLA

introduction

overlap and add approaches for, e.g.,

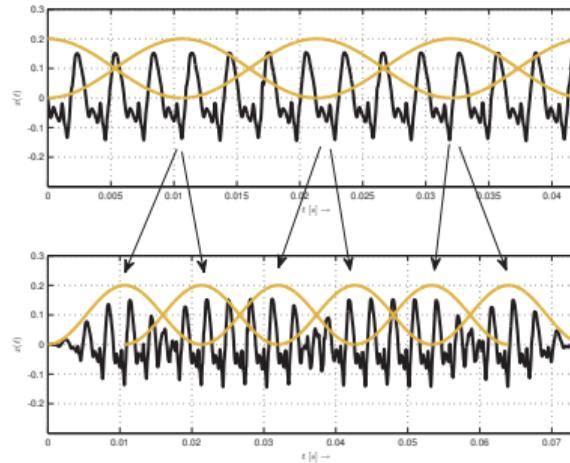
- granular synthesis
- time/frequency synthesis and processing
- **time-stretching and pitch-shifting**

OLA

time stretching

overlap and add

- 1 split input signal into overlapping blocks
- 2 duplicate or discard blocks depending on stretch factor

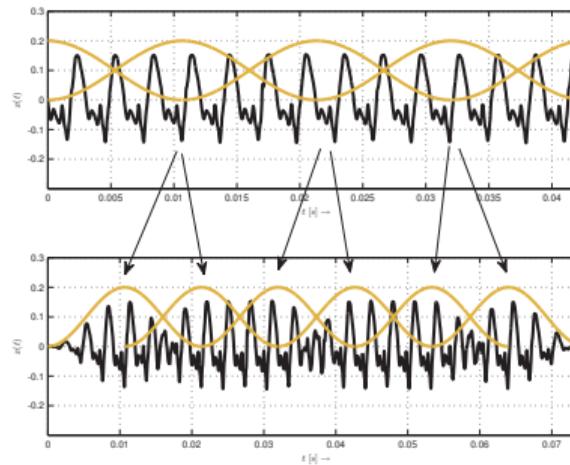


OLA

time stretching

overlap and add

- 1 **split input signal into overlapping blocks**
- 2 **duplicate or discard blocks** depending on stretch factor



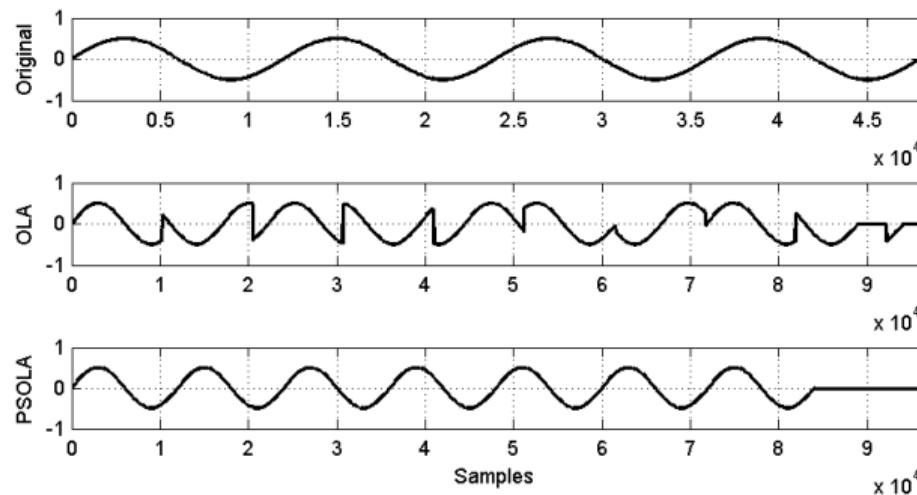
- orig
- $s = 4/3$

PSOLA

time stretching

pitch synchronous overlap and add

- use the OLA principle, but
- **adapt block length** to fundamental period length

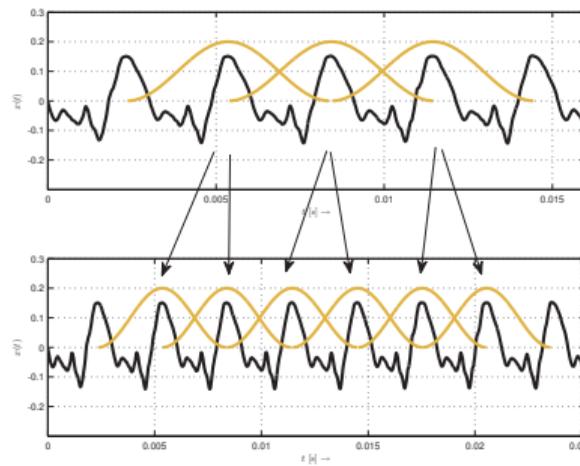


PSOLA

time stretching

pitch synchronous overlap and add

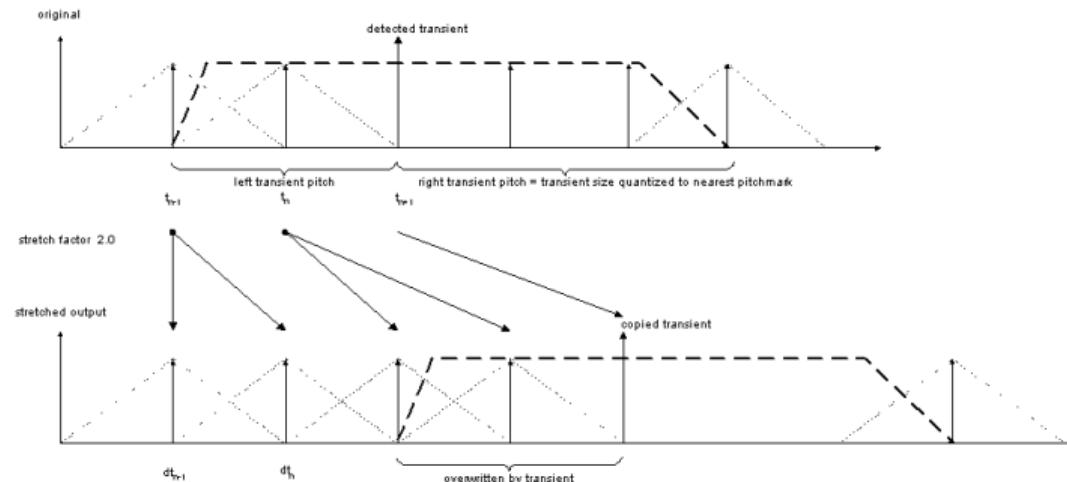
- use the OLA principle, but
- **adapt block length** to fundamental period length



- 🔊 orig
- 🔊 $s = 4/3$

PSOLA

transient copying



PSOLA

time stretching summary

■ processing steps

- 1 detect *fundamental frequency/period length*
- 2 set *pitch marks*
- 3 intelligently *select blocks* to be repeated/discarded

■ advantages

- *high granularity* — modify audio on period length resolution
- *high quality*

■ problems

- quality depends on *pitch tracking* reliability
- quality and timbre depends on *pitch mark positioning*
- works only for *monophonic* input signals
 - ▶ polyphonic and noisy segments
 - ▶ reverberation and overlapping tones
- *noise, plosives* require special consideration
- *copying artifacts* (double transients, timing deviations)

■ typical applications

- standard approach for vocal editing tools

PSOLA

time stretching summary

■ processing steps

- 1 detect *fundamental frequency/period length*
- 2 set *pitch marks*
- 3 intelligently *select blocks* to be repeated/discarded

■ advantages

- *high granularity* — modify audio on period length resolution
- *high quality*

■ problems

- quality depends on *pitch tracking* reliability
- quality and timbre depends on *pitch mark positioning*
- works only for *monophonic* input signals
 - ▶ polyphonic and noisy segments
 - ▶ reverberation and overlapping tones
- *noise, plosives* require special consideration
- *copying artifacts* (double transients, timing deviations)

■ typical applications

- standard approach for vocal editing tools

PSOLA

time stretching summary

■ processing steps

- 1 detect *fundamental frequency/period length*
- 2 set *pitch marks*
- 3 intelligently *select blocks* to be repeated/discarded

■ advantages

- *high granularity* — modify audio on period length resolution
- *high quality*

■ problems

- quality depends on *pitch tracking* reliability
- quality and timbre depends on *pitch mark positioning*
- works only for *monophonic* input signals
 - ▶ polyphonic and noisy segments
 - ▶ reverberation and overlapping tones
- *noise, plosives* require special consideration
- *copying artifacts* (double transients, timing deviations)

■ typical applications

- standard approach for vocal editing tools

PSOLA

time stretching summary

■ processing steps

- 1 detect *fundamental frequency/period length*
- 2 set *pitch marks*
- 3 intelligently *select blocks* to be repeated/discarded

■ advantages

- *high granularity* — modify audio on period length resolution
- *high quality*

■ problems

- quality depends on *pitch tracking* reliability
- quality and timbre depends on *pitch mark positioning*
- works only for *monophonic* input signals
 - ▶ polyphonic and noisy segments
 - ▶ reverberation and overlapping tones
- *noise, plosives* require special consideration
- *copying artifacts* (double transients, timing deviations)

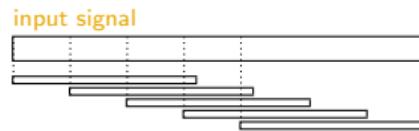
■ typical applications

- standard approach for vocal editing tools

time stretching

phase vocoder

- 1 **split input signal into overlapping blocks**
- 2 **compute magnitude and phase spectrum of each block**
- 3 **change overlap ratio** between blocks depending on stretch factor
- 4 **keep the magnitude, adapt the phase per bin** to the block's new time stamp



time stretching

phase vocoder

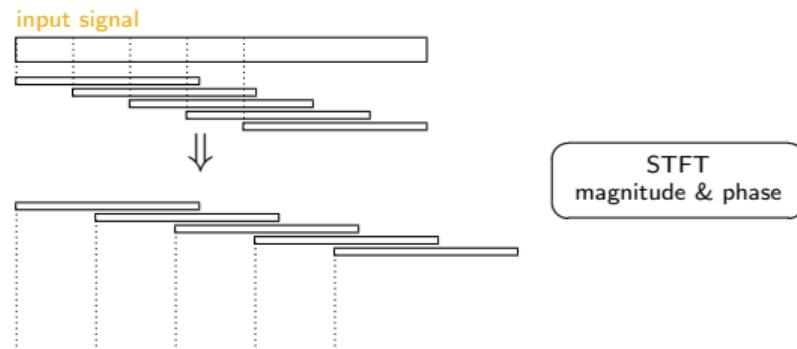
- 1 **split input signal into overlapping blocks**
- 2 **compute magnitude and phase spectrum of each block**
- 3 **change overlap ratio** between blocks depending on stretch factor
- 4 **keep the magnitude, adapt the phase per bin** to the block's new time stamp



time stretching

phase vocoder

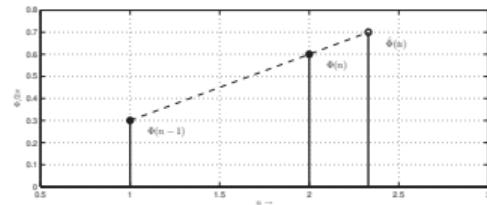
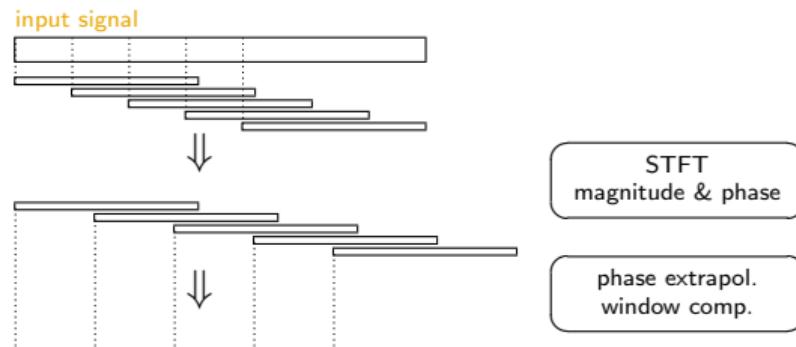
- 1 **split input** signal into overlapping blocks
- 2 compute **magnitude and phase spectrum** of each block
- 3 **change overlap ratio** between blocks depending on stretch factor
- 4 keep the magnitude, adapt the phase per bin to the block's new time stamp



time stretching

phase vocoder

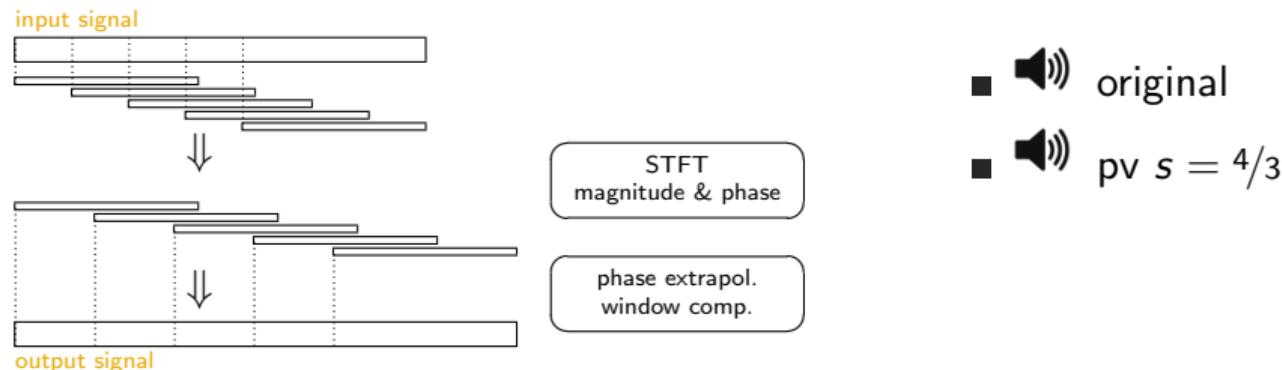
- 1 **split input** signal into overlapping blocks
- 2 compute **magnitude and phase spectrum** of each block
- 3 **change overlap ratio** between blocks depending on stretch factor
- 4 keep the magnitude, **adapt the phase per bin** to the block's new time stamp



time stretching

phase vocoder

- 1 **split input** signal into overlapping blocks
- 2 compute **magnitude and phase spectrum** of each block
- 3 **change overlap ratio** between blocks depending on stretch factor
- 4 keep the magnitude, **adapt the phase per bin** to the block's new time stamp



time stretching

frequency reassignment: relation of phase and frequency 1/2

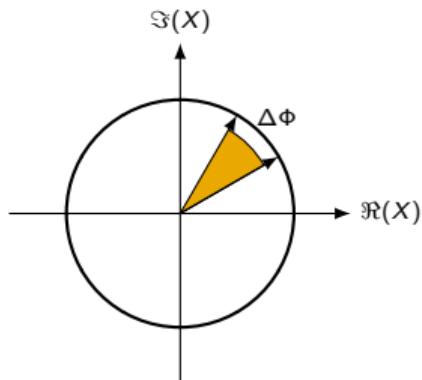
■ phasor representation:

- 1 sine value is defined by magnitude and phase
- 2 decreasing the amplitude \Rightarrow shorter vector
- 3 increasing the frequency \Rightarrow increasing speed



time stretching

frequency reassignment: relation of phase and frequency 2/2



- frequency and phase change closely related:

- time for full rotation is period length T with

$$f = \frac{1}{T}$$

- time for fractional rotation $\Delta\Phi$ is corresponding fraction of period length

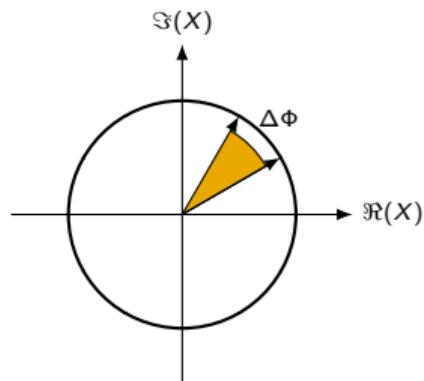
$$f = \frac{\Delta\Phi}{\Delta t}$$

- in other words:

$$\begin{aligned}\Phi(t) &= \omega \cdot t \\ \Rightarrow \frac{d\Phi(t)}{dt} &= \omega = 2\pi f\end{aligned}$$

time stretching

frequency reassignment: relation of phase and frequency 2/2



- frequency and phase change closely related:
 - time for full rotation is period length T with

$$f = \frac{1}{T}$$

- time for fractional rotation $\Delta\Phi$ is corresponding fraction of period length

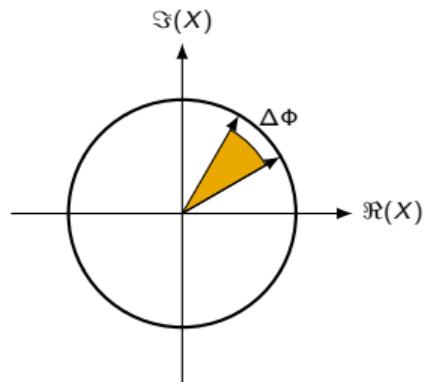
$$f = \frac{\Delta\Phi}{\Delta t}$$

- in other words:

$$\begin{aligned}\Phi(t) &= \omega \cdot t \\ \Rightarrow \frac{d\Phi(t)}{dt} &= \omega = 2\pi f\end{aligned}$$

time stretching

frequency reassignment: relation of phase and frequency 2/2



- frequency and phase change closely related:

- time for full rotation is period length T with

$$f = \frac{1}{T}$$

- time for fractional rotation $\Delta\Phi$ is corresponding fraction of period length

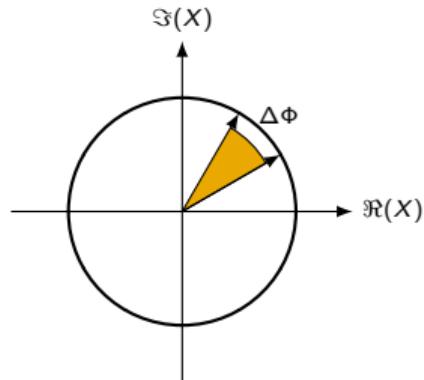
$$f = \frac{\Delta\Phi}{\Delta t}$$

- in other words:

$$\begin{aligned}\Phi(t) &= \omega \cdot t \\ \Rightarrow \frac{d\Phi(t)}{dt} &= \omega = 2\pi f\end{aligned}$$

time stretching

frequency reassignment: relation of phase and frequency 2/2



- frequency and phase change closely related:
 - time for full rotation is period length T with
$$f = \frac{1}{T}$$
 - time for fractional rotation $\Delta\Phi$ is corresponding fraction of period length
$$f = \frac{\Delta\Phi}{\Delta t}$$
 - in other words:

$$\begin{aligned}\Phi(t) &= \omega \cdot t \\ \Rightarrow \frac{d\Phi(t)}{dt} &= \omega = 2\pi f\end{aligned}$$

time stretching

frequency reassignment: principles

frequency domain:

- instead of using the bin frequency

$$f(k) = k * \frac{f_s}{\mathcal{K}}$$

- we use the phase of each bin $\Phi(k, n)$
- to compute the frequency from the phase difference of neighboring blocks

$$\omega_I(k, n) \propto \Phi(k, n) - \Phi(k, n - 1)$$

- $\omega_I(k, n)$ is called **instantaneous frequency** per block per bin

time stretching

frequency reassignment: principles

frequency domain:

- instead of using the bin frequency

$$f(k) = k * \frac{f_S}{\mathcal{K}}$$

- we use the phase of each bin $\Phi(k, n)$
- to compute the frequency from the phase difference of neighboring blocks

$$\omega_I(k, n) \propto \Phi(k, n) - \Phi(k, n - 1)$$

- $\omega_I(k, n)$ is called **instantaneous frequency** per block per bin

time stretching

frequency reassignment: scaling factor

- instantaneous frequency calculation has to take into account
 - hop size \mathcal{H}
 - sample rate f_S

$$\omega_I(k, n) = \frac{\Delta\Phi_u(k, n)}{\mathcal{H}} \cdot f_S$$

- problem: phase ambiguity

$$\Phi(k, n) = \Phi(k, n) + j \cdot 2\pi$$

⇒ *phase unwrapping*

time stretching

frequency reassignment: scaling factor

- instantaneous frequency calculation has to take into account
 - hop size \mathcal{H}
 - sample rate f_S

$$\omega_I(k, n) = \frac{\Delta\Phi_u(k, n)}{\mathcal{H}} \cdot f_S$$

- problem: phase ambiguity

$$\Phi(k, n) = \Phi(k, n) + j \cdot 2\pi$$

⇒ *phase unwrapping*

time stretching

frequency reassignment: scaling factor

- instantaneous frequency calculation has to take into account
 - hop size \mathcal{H}
 - sample rate f_S

$$\omega_I(k, n) = \frac{\Delta\Phi_u(k, n)}{\mathcal{H}} \cdot f_S$$

- problem: phase ambiguity

$$\Phi(k, n) = \Phi(k, n) + j \cdot 2\pi$$

\Rightarrow *phase unwrapping*

time stretching

frequency reassignment: phase unwrapping

1 compute unwrapped phase $\Phi_u(k, n)$

- estimate unwrapped bin phase

$$\hat{\Phi}(k, n) = \Phi(k, n - 1) + \underbrace{2\pi k \cdot \frac{\mathcal{H}}{\mathcal{K}}}_{=\omega_k \cdot \frac{\mathcal{H}}{f_s}}$$

- unwrap phase by shifting current phase to estimate's range

$$\Phi_u(k, n) = \hat{\Phi}(k, n) + \text{princarg} [\Phi(k, n) - \hat{\Phi}(k, n)]$$

2 compute unwrapped phase difference

$$\begin{aligned}\Delta\Phi_u(k, n) &= \Phi_u(k, n) - \Phi(k, n - 1) \\ &= \hat{\Phi}(k, n) + \text{princarg} [\Phi(k, n) - \hat{\Phi}(k, n)] - \Phi(k, n - 1) \\ &= \frac{2\pi k}{\mathcal{K}} \mathcal{H} + \text{princarg} \left[\Phi(k, n) - \Phi(k, n - 1) - \frac{2\pi k}{\mathcal{K}} \mathcal{H} \right]\end{aligned}$$

time stretching

frequency reassignment: phase unwrapping

1 compute unwrapped phase $\Phi_u(k, n)$

- estimate unwrapped bin phase

$$\hat{\Phi}(k, n) = \Phi(k, n - 1) + \underbrace{2\pi k \cdot \frac{\mathcal{H}}{\mathcal{K}}}_{= \omega_k \cdot \frac{\mathcal{H}}{f_s}}$$

- unwrap phase by shifting current phase to estimate's range

$$\Phi_u(k, n) = \hat{\Phi}(k, n) + \text{princarg} [\Phi(k, n) - \hat{\Phi}(k, n)]$$

2 compute unwrapped phase difference

$$\begin{aligned}\Delta\Phi_u(k, n) &= \Phi_u(k, n) - \Phi(k, n - 1) \\ &= \hat{\Phi}(k, n) + \text{princarg} [\Phi(k, n) - \hat{\Phi}(k, n)] - \Phi(k, n - 1) \\ &= \frac{2\pi k}{\mathcal{K}} \mathcal{H} + \text{princarg} [\Phi(k, n) - \Phi(k, n - 1) - \frac{2\pi k}{\mathcal{K}} \mathcal{H}]\end{aligned}$$

time stretching

frequency reassignment: phase unwrapping

1 compute unwrapped phase $\Phi_u(k, n)$

- estimate unwrapped bin phase

$$\hat{\Phi}(k, n) = \Phi(k, n - 1) + \underbrace{2\pi k \cdot \frac{\mathcal{H}}{\mathcal{K}}}_{= \omega_k \cdot \frac{\mathcal{H}}{f_s}}$$

- unwrap phase by shifting current phase to estimate's range

$$\Phi_u(k, n) = \hat{\Phi}(k, n) + \text{princarg} [\Phi(k, n) - \hat{\Phi}(k, n)]$$

2 compute unwrapped phase difference

$$\begin{aligned}\Delta\Phi_u(k, n) &= \Phi_u(k, n) - \Phi(k, n - 1) \\ &= \hat{\Phi}(k, n) + \text{princarg} [\Phi(k, n) - \hat{\Phi}(k, n)] - \Phi(k, n - 1) \\ &= \frac{2\pi k}{\mathcal{K}} \mathcal{H} + \text{princarg} \left[\Phi(k, n) - \Phi(k, n - 1) - \frac{2\pi k}{\mathcal{K}} \mathcal{H} \right]\end{aligned}$$

time stretching

frequency reassignment: problems

■ overlapping spectral components

- sinusoidal components often overlap (spectral leakage, several instruments playing the same pitch, ...)
 - ⇒ incorrect phase estimate
 - ▶ increase STFT length to make spectrum as sparse as possible

■ inaccurate phase unwrapping

- unwrapping algorithm is based on assumption of similarity between predicted and measured phase
 - ⇒ incorrect phase estimate
 - ▶ decrease hop size to make time steps as close as possible

time stretching

frequency reassignment: problems

■ overlapping spectral components

- sinusoidal components often overlap (spectral leakage, several instruments playing the same pitch, ...)
 - ⇒ incorrect phase estimate
 - ▶ increase STFT length to make spectrum as sparse as possible

■ inaccurate phase unwrapping

- unwrapping algorithm is based on assumption of similarity between predicted and measured phase
 - ⇒ incorrect phase estimate
 - ▶ decrease hop size to make time steps as close as possible

time stretching

frequency reassignment: problems

■ overlapping spectral components

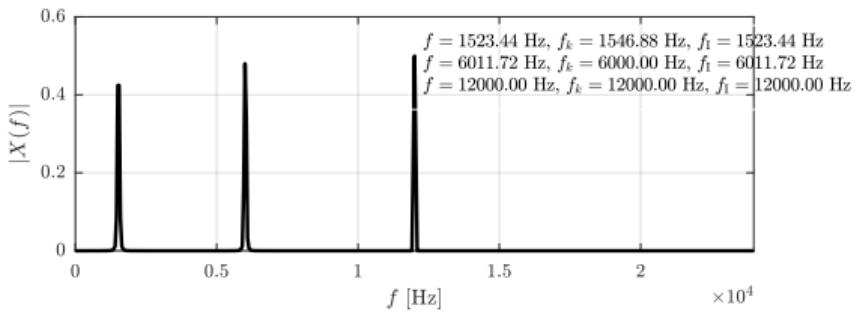
- sinusoidal components often overlap (spectral leakage, several instruments playing the same pitch, ...)
 - ⇒ incorrect phase estimate
 - ▶ increase STFT length to make spectrum as sparse as possible

■ inaccurate phase unwrapping

- unwrapping algorithm is based on assumption of similarity between predicted and measured phase
 - ⇒ incorrect phase estimate
 - ▶ decrease hop size to make time steps as close as possible

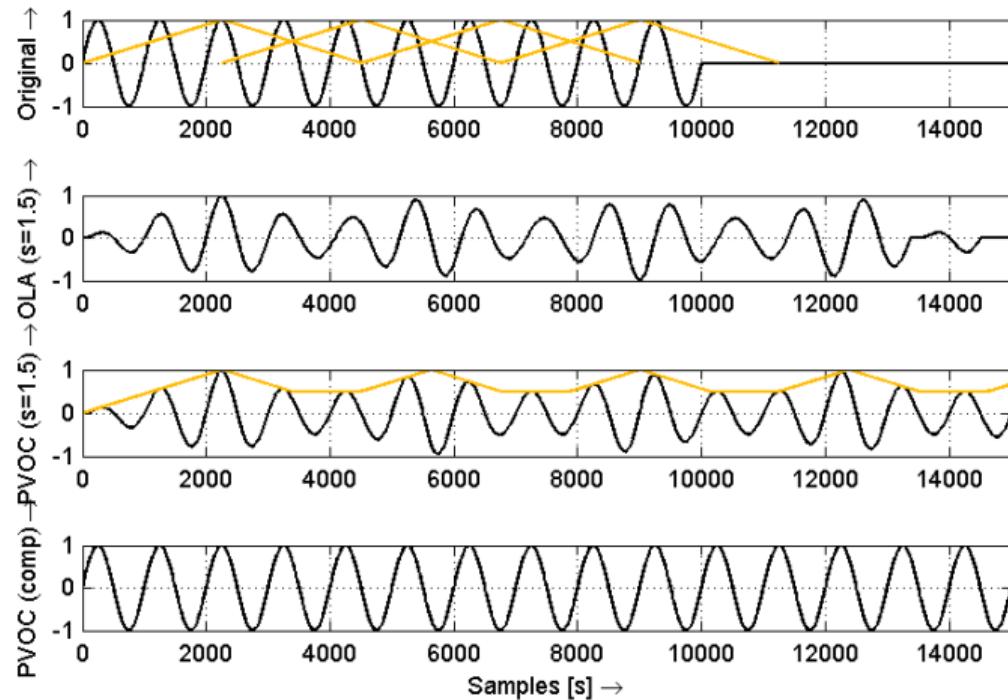
time stretching

frequency reassignment: example



time stretching

phase vocoder window compensation



time stretching

phase vocoder — properties & artifacts

■ advantages:

- allows *Polyphonic input* (assumption: no overlapping harmonics)
- absolute *timing stability* (i.e., sample resolution)

■ disadvantages:

- *low granularity* — FFT block size
- artifacts:
 - 1 phasing
 - 2 transient smearing/doubling

time stretching

phase vocoder — properties & artifacts

■ advantages:

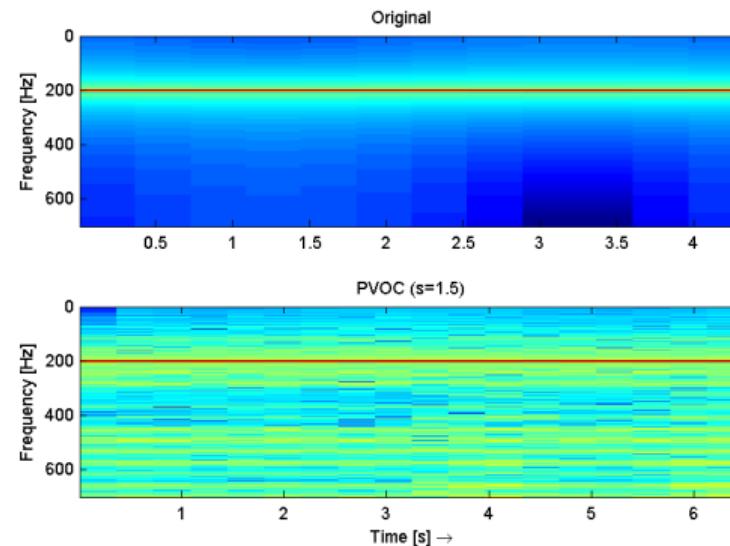
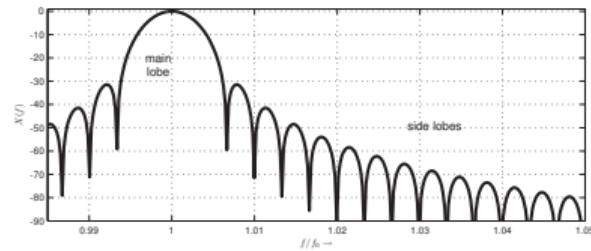
- allows *Polyphonic input* (assumption: no overlapping harmonics)
- absolute *timing stability* (i.e., sample resolution)

■ disadvantages:

- *low granularity* — FFT block size
- artifacts:
 - 1 phasing
 - 2 transient smearing/doubling

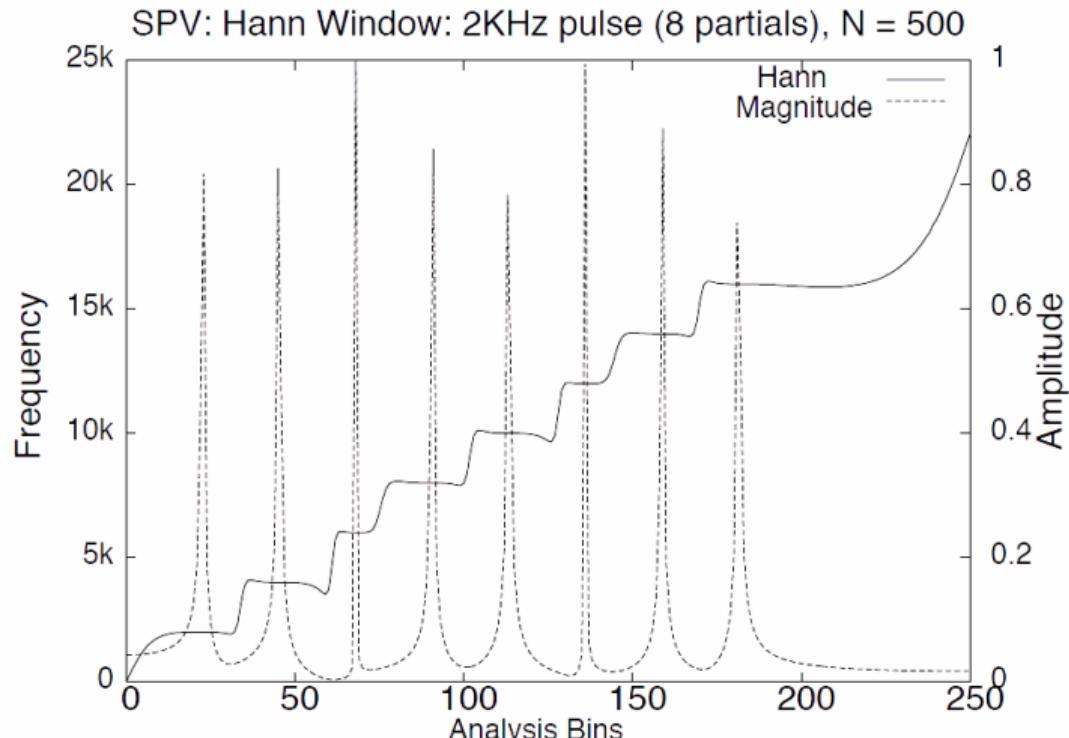
time stretching

phase vocoder artifacts: phasing — spectral leakage 1/3



time stretching

phase vocoder artifacts: phasing — spectral leakage 2/3



time stretching

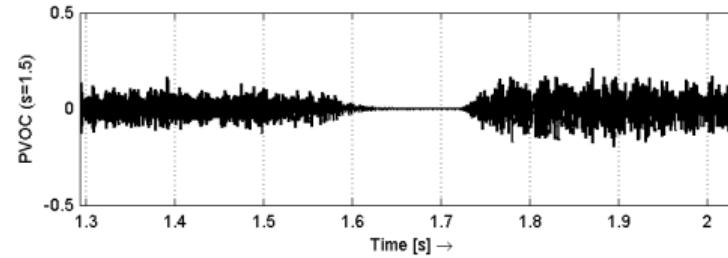
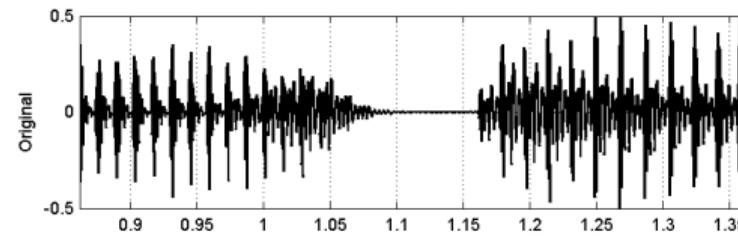
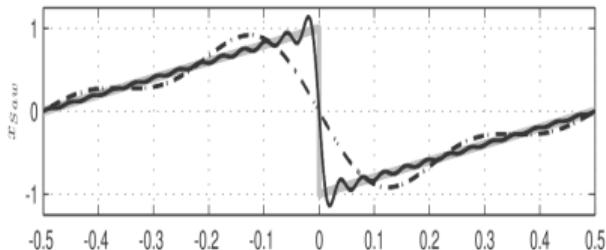
phase vocoder artifacts: phasing — spectral leakage 3/3

⇒ use *frequency reassignment* for grouping and phase sync

- original 
- $pv\ s = 4/3$ 
- grouped phase 

time stretching

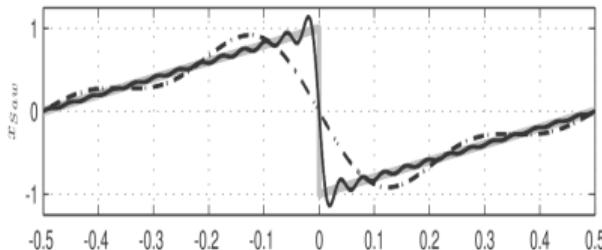
phase vocoder artifacts: phasing — unsynced harmonics



⇒ use *harmonic analysis* for grouping and phase

time stretching

phase vocoder artifacts: phasing — unsynced harmonics



⇒ use *harmonic analysis* for grouping and phase sync

- original
- pv $s = 4/3$
- synced phase

time stretching

phase vocoder artifacts: interchannel phasing

- phase estimation between channels slightly off due to

- numerical inaccuracies (cumulative!)
- overlapping frequency components

⇒ change in spatial image

-  original
-  $pv\ s = \frac{3}{2}$

time stretching

phase vocoder artifacts: interchannel phasing

- phase estimation between channels slightly off due to

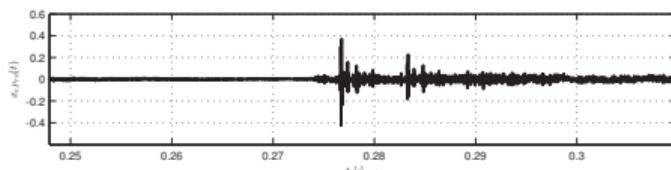
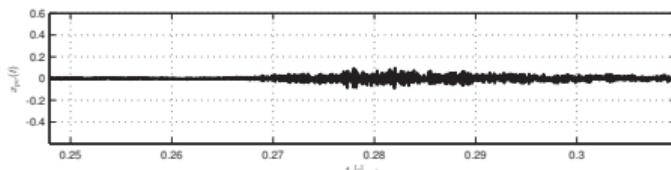
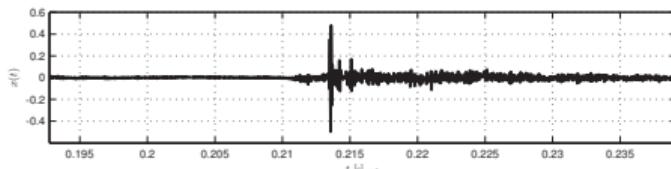
- numerical inaccuracies (cumulative!)
- overlapping frequency components

⇒ change in spatial image

-  original
-  pv $s = \frac{3}{2}$

time stretching

phase vocoder artifacts: transient smearing



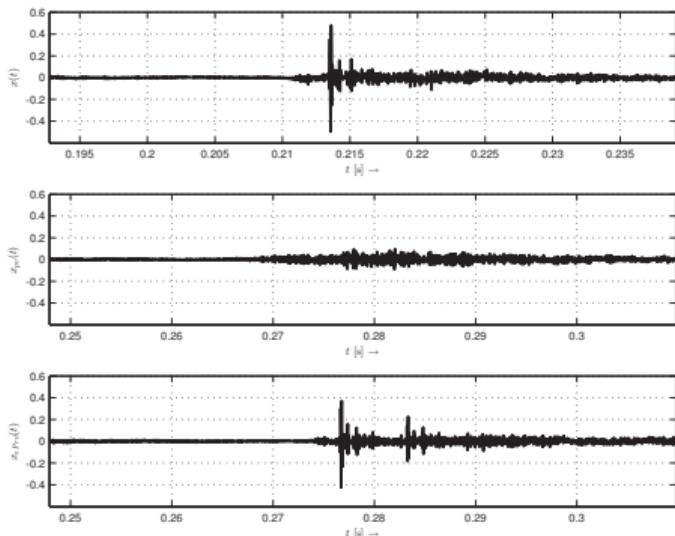
- original
- pv $s = 3/2$

⇒ detect transients and *reset phase per bin*

- original
- pv $s = 4/3$
- phase reset

time stretching

phase vocoder artifacts: transient smearing



- original
- pv $s = 3/2$

⇒ detect transients and *reset phase per bin*

- original
- pv $s = 4/3$
- phase reset

time stretching

inherent problems

- stretching the audio data can lead to “**non-natural**” results

- examples

- tempo dependent *timing variations*
- other performance related aspects may get inappropriate lengths and speed: *vibrato, tremolo, glissando*

time stretching

inherent problems

- stretching the audio data can lead to “**non-natural**” results

- examples

- tempo dependent *timing variations*
- other performance related aspects may get inappropriate lengths and speed: *vibrato, tremolo, glissando*

time stretching

inherent problems

- stretching the audio data can lead to “**non-natural**” results

- examples

- tempo dependent *timing variations*
- other performance related aspects may get inappropriate lengths and speed: *vibrato, tremolo, glissando*

pitch shifting

standard approach

■ definition:

change pitch without changing tempo

■ method:

combine stretching and *sample rate conversion* (interpolation)

- 1 change length with stretching
- 2 resample to compensate for length difference

■ implementation: differentiate "external" and "internal" parameters

- *extern*: stretch s_e and pitch p_e
- *intern*: stretch s_i and resample r_i

■ example: pitch shift factor $p = 4/3$

- time stretch (increase length/decrease tempo) $s = 4/3$
- resample (decrease length, increase pitch) $r = 3/4$

⇒ audio:



pitch shifting

standard approach

■ definition:

change pitch without changing tempo

■ method:

combine stretching and *sample rate conversion* (interpolation)

- 1 change length with stretching
- 2 resample to compensate for length difference

■ implementation: differentiate “external” and “internal” parameters

- *extern*: stretch s_e and pitch p_e
- *intern*: stretch s_i and resample r_i

■ example: pitch shift factor $p = 4/3$

- 1 time stretch (increase length/decrease tempo) $s = 4/3$

- 2 resample (decrease length, increase pitch) $r = 3/4$

⇒ audio:



pitch shifting

standard approach

■ definition:

change pitch without changing tempo

■ method:

combine stretching and *sample rate conversion* (interpolation)

- 1 change length with stretching
- 2 resample to compensate for length difference

■ implementation: differentiate “external” and “internal” parameters

- *extern*: stretch s_e and pitch p_e
- *intern*: stretch s_i and resample r_i

■ example: pitch shift factor $p = 4/3$

- 1 *time stretch* (increase length/decrease tempo) $s = 4/3$
- 2 *resample* (decrease length, increase pitch) $r = 3/4$

⇒ audio:

▶ OLA 

▶ phase vocoder 

pitch shifting

standard approach

■ definition:

change pitch without changing tempo

■ method:

combine stretching and *sample rate conversion* (interpolation)

- 1 change length with stretching
- 2 resample to compensate for length difference

■ implementation: differentiate “external” and “internal” parameters

- *extern*: stretch s_e and pitch p_e
- *intern*: stretch s_i and resample r_i

■ example: pitch shift factor $p = 4/3$

- 1 *time stretch* (increase length/decrease tempo) $s = 4/3$

- 2 *resample* (decrease length, increase pitch) $r = 3/4$

⇒ audio:



pitch shifting

standard approach: stretch, pitch, resample factor examples

- extern: $s_e = 1$ $p_e = 2$

pitch shifting

standard approach: stretch, pitch, resample factor examples

■ extern: $s_e = 1$ $p_e = 2$

intern: $s_i = 2$ $r_i = \frac{1}{2}$

pitch shifting

standard approach: stretch, pitch, resample factor examples

- extern: $s_e = 1$ $p_e = 2$
intern: $s_i = 2$ $r_i = \frac{1}{2}$
- extern: $s_e = 1$ $p_e = \frac{4}{3}$

pitch shifting

standard approach: stretch, pitch, resample factor examples

- extern: $s_e = 1$ $p_e = 2$

intern: $s_i = 2$ $r_i = \frac{1}{2}$

- extern: $s_e = 1$ $p_e = \frac{4}{3}$

intern: $s_i = \frac{4}{3}$ $r_i = \frac{3}{4}$

pitch shifting

standard approach: stretch, pitch, resample factor examples

- extern: $s_e = 1$ $p_e = 2$
intern: $s_i = 2$ $r_i = \frac{1}{2}$
- extern: $s_e = 1$ $p_e = \frac{4}{3}$
intern: $s_i = \frac{4}{3}$ $r_i = \frac{3}{4}$
- extern: $s_e = \frac{1}{2}$ $p_e = 2$

pitch shifting

standard approach: stretch, pitch, resample factor examples

- extern: $s_e = 1$ $p_e = 2$

intern: $s_i = 2$ $r_i = \frac{1}{2}$

- extern: $s_e = 1$ $p_e = \frac{4}{3}$

intern: $s_i = \frac{4}{3}$ $r_i = \frac{3}{4}$

- extern: $s_e = \frac{1}{2}$ $p_e = 2$

intern: $s_i = 1$ $r_i = \frac{1}{2}$

pitch shifting

standard approach: stretch, pitch, resample factor examples

- extern: $s_e = 1$ $p_e = 2$
intern: $s_i = 2$ $r_i = \frac{1}{2}$
- extern: $s_e = 1$ $p_e = \frac{4}{3}$
intern: $s_i = \frac{4}{3}$ $r_i = \frac{3}{4}$
- extern: $s_e = \frac{1}{2}$ $p_e = 2$
intern: $s_i = 1$ $r_i = \frac{1}{2}$
- extern: $s_e = 2$ $p_e = 2$

pitch shifting

standard approach: stretch, pitch, resample factor examples

- extern: $s_e = 1$ $p_e = 2$

intern: $s_i = 2$ $r_i = \frac{1}{2}$

- extern: $s_e = 1$ $p_e = \frac{4}{3}$

intern: $s_i = \frac{4}{3}$ $r_i = \frac{3}{4}$

- extern: $s_e = \frac{1}{2}$ $p_e = 2$

intern: $s_i = 1$ $r_i = \frac{1}{2}$

- extern: $s_e = 2$ $p_e = 2$

intern: $s_i = 4$ $r_i = \frac{1}{2}$

pitch shifting

frequency domain approach

1 STFT

- 2 magnitude and phase
- 3 magnitude and instantaneous frequency
- 4 resample both magnitude and frequency spectrum acc. to pitch factor
- 5 magnitude and phase
- 6 complex spectrum
- 7 IFFT and OLA

pitch shifting

frequency domain approach

1 STFT

2 magnitude and phase

3 magnitude and instantaneous frequency

4 resample both magnitude and frequency spectrum acc. to pitch factor

5 magnitude and phase

6 complex spectrum

7 IFFT and OLA

pitch shifting

frequency domain approach

- 1 STFT
- 2 magnitude and phase
- 3 magnitude and instantaneous frequency
- 4 resample both magnitude and frequency spectrum acc. to pitch factor
- 5 magnitude and phase
- 6 complex spectrum
- 7 IFFT and OLA

pitch shifting

frequency domain approach

- 1 STFT
- 2 magnitude and phase
- 3 magnitude and instantaneous frequency
- 4 resample both magnitude and frequency spectrum acc. to pitch factor
- 5 magnitude and phase
- 6 complex spectrum
- 7 IFFT and OLA

pitch shifting

frequency domain approach

- 1 STFT
- 2 magnitude and phase
- 3 magnitude and instantaneous frequency
- 4 resample both magnitude and frequency spectrum acc. to pitch factor
- 5 magnitude and phase
- 6 complex spectrum
- 7 IFFT and OLA

pitch shifting

frequency domain approach

- 1 STFT
- 2 magnitude and phase
- 3 magnitude and instantaneous frequency
- 4 resample both magnitude and frequency spectrum acc. to pitch factor
- 5 magnitude and phase
- 6 complex spectrum
- 7 IFFT and OLA

pitch shifting

frequency domain approach

- 1 STFT
- 2 magnitude and phase
- 3 magnitude and instantaneous frequency
- 4 resample both magnitude and frequency spectrum acc. to pitch factor
- 5 magnitude and phase
- 6 complex spectrum
- 7 IFFT and OLA

pitch shifting

formant preservation: time domain

■ idea

- signal is pulse train filtered by transfer function
 - ▶ pulse train determines fundamental frequency
 - ▶ transfer function determines formant shape/timbre characteristics

■ approach

- change grain/pulse distance
- grain “content” not modified
→ freq domain not modified

⇒ “same” spectrum,
different pitch

pitch shifting

formant preservation: time domain

■ idea

- signal is pulse train filtered by transfer function
 - ▶ pulse train determines fundamental frequency
 - ▶ transfer function determines formant shape/timbre characteristics

■ approach

- change grain/pulse distance
- grain “content” not modified
→ freq domain not modified

⇒ “same” spectrum,
different pitch

pitch shifting

formant preservation: time domain

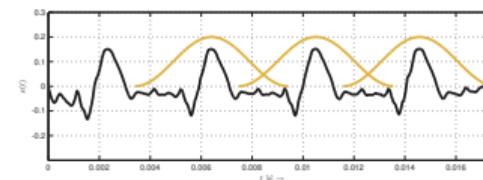
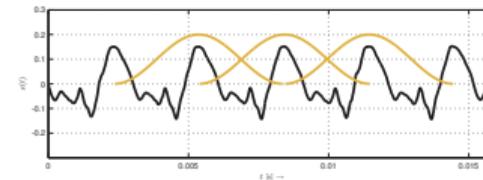
■ idea

- signal is pulse train filtered by transfer function
 - ▶ pulse train determines fundamental frequency
 - ▶ transfer function determines formant shape/timbre characteristics

■ approach

- change grain/pulse distance
- grain “content” not modified
→ freq domain not modified

⇒ “same” spectrum,
different pitch



pitch shifting

formant preservation: time domain

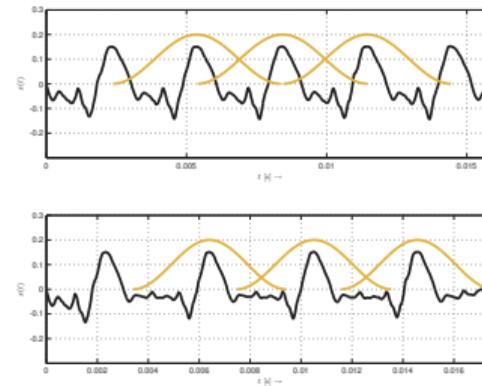
■ idea

- signal is pulse train filtered by transfer function
 - ▶ pulse train determines fundamental frequency
 - ▶ transfer function determines formant shape/timbre characteristics

■ approach

- change grain/pulse distance
- grain “content” not modified
→ freq domain not modified

⇒ **“same” spectrum,
different pitch**



pitch shifting

formant preservation: frequency domain

■ idea

- preserve spectral envelope

■ approach

- 1 measure spectral envelope
- 2 apply inverse envelope (whitening)
- 3 pitch shift
- 4 apply spectral envelope

pitch shifting

formant preservation: frequency domain

■ idea

- preserve spectral envelope

■ approach

- 1 measure spectral envelope
- 2 apply inverse envelope (whitening)
- 3 pitch shift
- 4 apply spectral envelope

pitch shifting

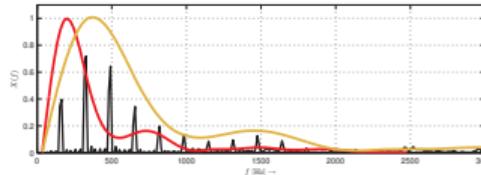
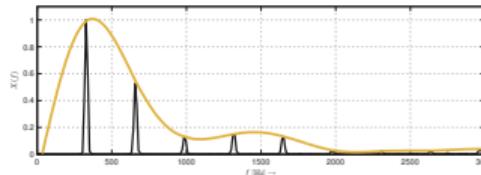
formant preservation: frequency domain

■ idea

- preserve spectral envelope

■ approach

- 1 measure spectral envelope
- 2 apply inverse envelope (whitening)
- 3 pitch shift
- 4 apply spectral envelope



pitch shifting

spectral envelope estimate

■ approaches

- LPC coefficients
- spectral maxima

■ potential issues

- *Polyphonic input audio:*
'superposition' of envelopes
- *Very high/low pitch factors:* high frequency boost/cut
- *estimate resolution*
 - ▶ too coarse → loss of timbre characteristics
 - ▶ too fine → impress pitch characteristics (harmonic pattern) on spectrum

pitch shifting

spectral envelope estimate

■ approaches

- LPC coefficients
- spectral maxima

■ potential issues

- *Polyphonic input* audio:
'superposition' of envelopes
- *Very high/low pitch factors*: high frequency boost/cut
- *estimate resolution*
 - ▶ too coarse → loss of timbre characteristics
 - ▶ too fine → impress pitch characteristics (harmonic pattern) on spectrum

pitch shifting

audio examples

	OLA	PSOLA	PVOC
orig			
resample $p = 4/3$			
formant $p = 4/3$			

summary

time stretching and pitch shifting

- pitch stretching and pitch shifting are largely equivalent algorithms
 - same artifacts
 - same workload
- monophonic time-stretching with PSOLA-based approaches is
 - easier to solve
 - has very bad artifacts if pitch tracker is off
- polyphonic time-stretching with PV-based approaches is
 - complicated due to tradeoffs (e.g., frequency vs time resolution)
- general challenges:
 - noisy and transient signals
 - resulting timbre changes
 - perceived naturalness of result
 - time resolution/accuracy due to blocked processing