DSP-CIS

Part-IV : Filter Banks & Time-Frequency Transforms

Chapter-14 : Time-Frequency Analysis & Scaling

Marc Moonen
Dept. E.E./ESAT-STADIUS, KU Leuven
marc.moonen@kuleuven.be
www.esat.kuleuven.be/stadius/

Part-IV : Filter Banks & Time-Frequency Transforms

Overview

Time-Frequency Analysis

- Short-time Fourier Transform (STFT)
- Weighted OverLap-Add (WOLA)
- Wavelet Analysis & Wavelet Filter Banks

Time/Frequency Scaling of Speech/Audio Signals

- Problem Statement & Approaches
- STFT-Based Time Scaling

Short-Time Fourier Transform

Starting point is **Discrete-Time Fourier Transform**

$$U(e^{j\omega}) = \sum_{k=-\infty}^{+\infty} u[k] \cdot e^{-j\omega k} \qquad 0 \le \omega \le 2\pi$$

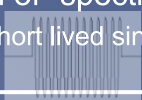
= infinitely long sequence u[k] is evaluated at infinitely many frequencies

Inversion/reconstruction/synthesis (=filter bank jargon) is..

$$u[k] = \frac{1}{2\pi} \int_0^{2\pi} U(e^{j\omega}) \cdot e^{j\omega k}\, d\omega$$

= u[k] represented as weighted sum of (orthogonal) basis functions $e^{j\omega k}$

- Notion of `spectrum that varies with time' not accommodated (e.g. `short lived sine' will correspond to non-localized spectrum)



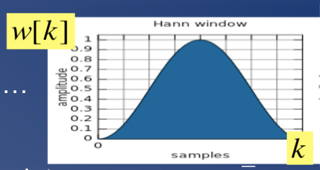
Short-Time Fourier Transform

Tool to fill this need = **Short-Time Fourier Transform**

Starting point is now…

$$U(e^{j\omega},k)=\sum_{\bar{k}=-\infty}^{+\infty} u[\bar{k}].w[k-\bar{k}].e^{-j\omega\bar{k}} \qquad 0\le\omega\le 2\pi \quad , \quad -\infty<k<+\infty$$

where w[k] is your favorite window function typically with `compact support' (=FIR), e.g. …



- Reversed (check formula) window slides past the data

  For each window position k, compute Discrete-Time Fourier Transform

PS: If w[k]=1 (all *k*) then this is just Discrete-Time Fourier Transform, for all window positions

Short-Time Fourier Transform

Tool to fill this need = **Short-Time Fourier Transform**

Starting point is now…

$$U(e^{j\omega},k)=\sum_{\bar{k}=-\infty}^{+\infty} u[\bar{k}].w[k-\bar{k}].e^{-j\omega\bar{k}} \qquad 0\le\omega\le 2\pi \quad , \quad -\infty<k<+\infty$$

Question: What would an inversion formula look like, here?

In the following slides, will provide a

filter bank derivation of STFT,

also leading to a simple

inversion formula (based on perfect reconstruction theory)

Short-Time Fourier Transform

First, **rewrite** formula as…

$$U(e^{j\omega},k) = e^{-j\omega k}.\sum_{\bar{k}=-\infty}^{+\infty} u[\bar{k}].w[k-\bar{k}].e^{j\omega(k-\bar{k})}$$

where the phase factor $e^{-j\omega k} = z^{-k}\big|_{z=e^{j\omega}}$ can effectively be removed

Interpretation-1: 'windowed' signal segment with window positioned at time k is shifted to time zero (*) before computing discrete-time Fourier transform (DTFT) , so that the DTFT indeed gets multiplied by $e^{j\omega k} = z^{k}\big|_{z=e^{j\omega}}$

Interpretation-2: modulate window instead of input signal

So from now on will use

$$U(e^{j\omega},k) = \sum_{\bar{k}=-\infty}^{+\infty} u[\bar{k}].w[k-\bar{k}].e^{j\omega(k-\bar{k})}$$

Short-Time Fourier Transform

$$U(e^{j\omega},k) = \sum_{\bar{k}=-\infty}^{+\infty} u[\bar{k}].w[k-\bar{k}].e^{j\omega(k-\bar{k})}$$

This correspond to performing a **convolution** of u[k] with (infinitely many) filters

$$h_\omega[k] = w[k].e^{j\omega k} \qquad 0 \le \omega \le 2\pi$$

In practice, will compute this for a discrete set of (N) frequencies…

$\omega_n = n.\frac{2\pi}{N}$ n=0,…,N-1

leading to a finite set of filters… $h_n[k] = h_0[k].e^{j(n.2\pi/N).k} \quad , \quad h_0[k] = w[k]$

This is a **DFT-modulated analysis bank** (Chapter-12, p.14)

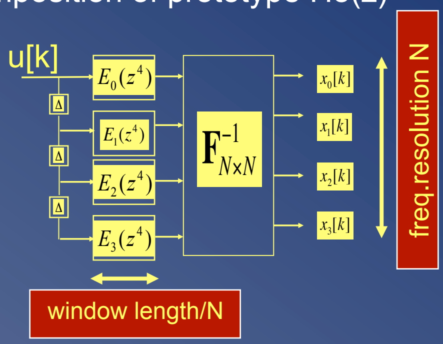
N channels, prototype defined by window function, with subband signals

$$x_n[k] = \sum_{\bar{k}=-\infty}^{+\infty} u[\bar{k}].h_n[k-\bar{k}] \qquad n = 0,\ldots,N-1$$

Short-Time Fourier Transform

Hence, can use efficient implementation based on polyphase decomposition of prototype Ho(z)

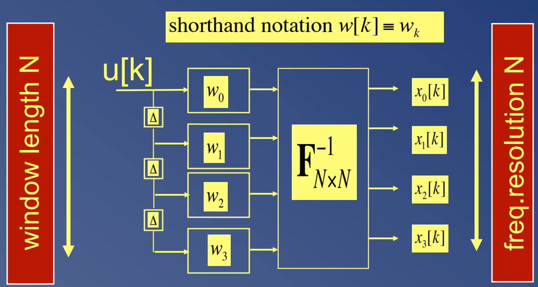
(Chapter-12, p.17)



$$H_0(z) = \sum_{n=0}^{N-1} z^{-n}.E_n(z^N)$$

Short-Time Fourier Transform

In practice, mostly **window length=freq.resolution=N**



(=DFT-modulated FB with 1-tap polyphase components of prototype)

PS: also remember F^{-1} (inverse DFT-matrix) is not very different from F (DFT-matrix) …

Will use this from now on...

Short-Time Fourier Transform

In practice **analysis is done only for k=0,D,2D,...**
(i.e. decimation D = ‘window shift’ > 1)

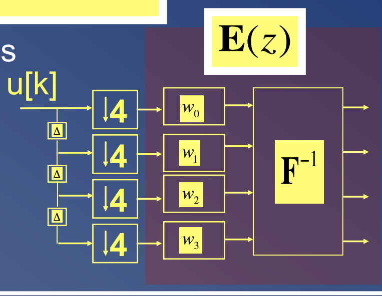
- If **maximally decimated** (D=N) then analysis FB operation corresponds to **Short-Time Fourier Transform (STFT)**

$$x_n[k] = \sum_{\bar{k}=-\infty}^{+\infty} u[\bar{k}].h_n[N.k-\bar{k}] \qquad n=0,\ldots,N-1$$

$x_n[k]$ = decimated subband signals
= `**STFT-coefficients**’

= infinitely long sequence u[k] is evaluated at N frequencies, infinitely many times (i.e. for infinitely many window positions)

...to be compared to p.4

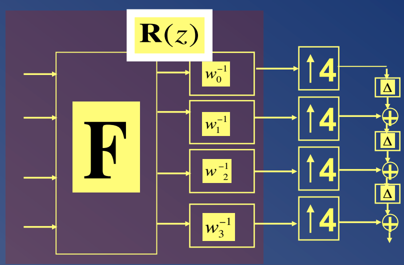


Short-Time Fourier Transform

- Corresponding (PR) DFT-modulated synthesis FB is

$$\mathbf{E}(z) = \mathbf{F}^{-1}.diag\left[w_i\right] \quad \Rightarrow \quad \mathbf{R}(z) = diag\left[w_i^{-1}\right].\mathbf{F}$$

i.e. synthesis prototype filter $f_o[k]$
= ‘synthesis window’
= $w_0^{-1}, w_1^{-1}, w_2^{-1}, \ldots, w_{N-1}^{-1}$



- Synthesis FB operation then corresponds to **Inverse STFT**

$$u[k] = \sum_{n=0}^{N-1} \sum_{\bar{k}=-\infty}^{+\infty} x_n[\bar{k}].f_n[k-N.\bar{k}]$$

...to be compared to p.4

Overview

Time-Frequency Analysis

- Short-time Fourier Transform (STFT)
- Weighted OverLap-Add (WOLA)
- Wavelet Analysis & Wavelet Filter Banks

Time/Frequency Scaling of speech/audio signals

- Problem Statement & Approaches
- STFT-Based Time Scaling

Weighted OverLap-Add

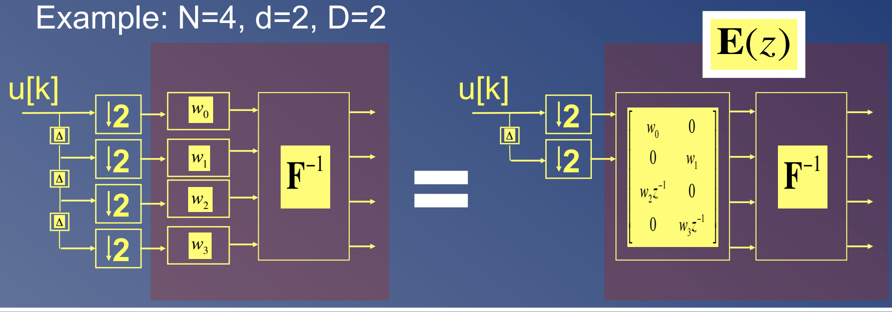
Continued from p.12…

- If **oversampled** (D=N/d), then analysis FB operation is

$$x_n[k] = \sum_{\bar{k}=-\infty}^{+\infty} u[\bar{k}].h_n[\frac{N}{d}.k - \bar{k}] \qquad n = 0,\ldots,N-1$$

d=2,3,4,.. corresponds to decimation D (=window shift) of N/2, N/3, N/4,..
i.e. 50%, 66%, 75%,.. 'window overlap'

Example: N=4, d=2, D=2



Weighted OverLap-Add

Example: N=4, d=2, D=2 (continued)

Corresponding (PR) synthesis filter bank (also DFT-modulated) is..

$$\mathbf{E}(z)=\mathbf{F}^{-1}.\begin{bmatrix} w_0 & 0 \\ 0 & w_1 \\ w_2 z^{-1} & 0 \\ 0 & w_3 z^{-1} \end{bmatrix} \quad \Rightarrow \quad \mathbf{R}(z)=\begin{bmatrix} v_0 z^{-1} & 0 & v_2 & 0 \\ 0 & v_1 z^{-1} & 0 & v_3 \end{bmatrix}.\mathbf{F}$$

PR condition then defines **synthesis window** v_k

$$\mathbf{R}(z).\mathbf{E}(z) = z^{-\delta}.\mathbf{I} \overset{\delta=1}{\Longrightarrow} \begin{cases} w_0 v_0 + w_2 v_2 = 1 \\ w_1 v_1 + w_3 v_3 = 1 \end{cases}$$

..which has many solutions. A usefull solution (see below) will be (**)

$$v_0=\frac{w_0}{w_0^2+w_2^2},\quad v_2=\frac{w_2}{w_0^2+w_2^2},\quad v_1=\frac{w_1}{w_1^2+w_3^2},\quad v_3=\frac{w_3}{w_1^2+w_3^2}$$

Weighted OverLap-Add

- **PR condition**
  can be generalized for other oversampling factors D=N/d

$$\sum_{k=-\infty}^{\infty}(w_{i+\frac{N}{d}k}).(v_{i+\frac{N}{d}k})=1 \quad \text{for} \quad i=0..(\frac{N}{d}-1)$$

- Synthesis FB operation corresponds to **Inverse Transform**

$$u[k]=\sum_{n=0}^{N-1}\sum_{\bar{k}=-\infty}^{+\infty} x_n[\bar{k}].f_n[k-\frac{N}{d}.\bar{k}]$$

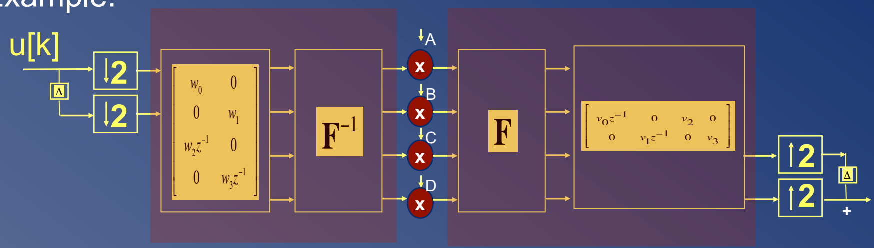
This analysis/synthesis is referred to as

Weighted OverLap-Add (WOLA)

Weighted OverLap-Add

A WOLA filter bank is often used for 'subband processing'…
(a.k.a. 'frequency domain processing')

Example:



…where the scalar multipliers (A,BC,D) for instance result from a per-subband noise reduction strategy (see Speech&Audio course)

Input-output characteristic then **approximates** a linear time-invariant filtering

(better approximation with better (more frequency selective) analysis/synthesis filters)

Weighted OverLap-Add

A WOLA filter bank is often used for 'subband processing'…
(a.k.a. 'frequency domain processing')

A special case for d=2 (50% overlap) is the '**overlap-save**'...

$$\mathbf{E}(z)=\mathbf{F}^{-1}.\begin{bmatrix} I \\ z^{-1}.I \end{bmatrix} \qquad \mathbf{R}(z)=\begin{bmatrix} 0 & I \end{bmatrix}.\mathbf{F}$$

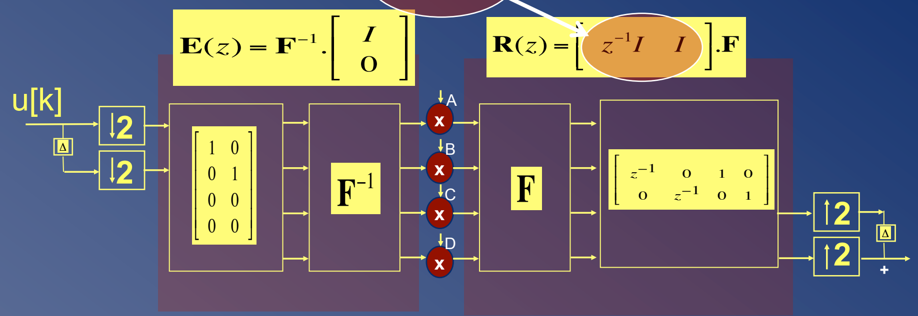
…and '**overlap-add**' filter bank

$$\mathbf{E}(z)=\mathbf{F}^{-1}.\begin{bmatrix} I \\ 0 \end{bmatrix} \qquad \mathbf{R}(z)=\begin{bmatrix} z^{-1}I & I \end{bmatrix}.\mathbf{F}$$

as used for **frequency domain filter realizations** (see Chapter 13)

Weighted OverLap-Add

Hence note that with an (unweighted) overlap-add (or –save) FB…



(despite the poor analysis/synthesis filter bank characteristics!)

…the subband processing with (e.g.) scalar multipliers (A,B,C,D) can be made to correspond **exactly** to a linear time-invariant filtering, **iff** the scalars jointly satisfy a specific condition (see Chapter 13)

Overview

Time-Frequency Analysis

- Short-time Fourier Transform (STFT)
- Weighted OverLap-Add (WOLA)
- Wavelet Analysis & Wavelet Filter Banks

Time/Frequency Scaling of speech/audio signals

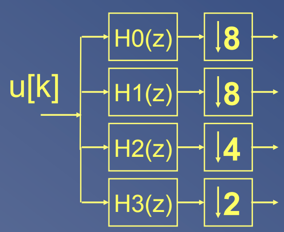
- Problem Statement & Approaches
- STFT-Based Time Scaling

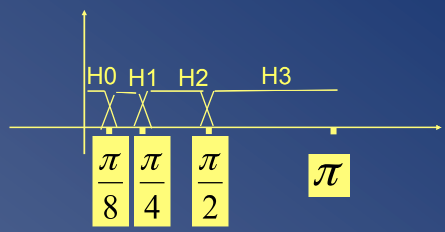
Wavelet Filter Banks & Wavelets

For some time-frequency analysis applications (e.g. in audio), would like to have a **non-uniform** filter bank

(instead of the uniform DFT-modulated filter bank of STFT)

hence also with **non-uniform** (maximum) decimation, e.g…

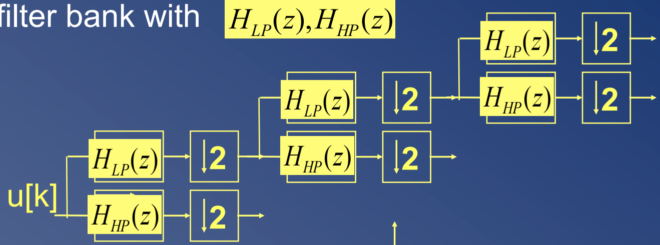




- **Non-uniform filters** = low frequency resolution at high frequencies, high frequency resolution at low frequencies (as human hearing)
- **Non-uniform decimation** = high time resolution at high frequencies, low time resolution at low frequencies

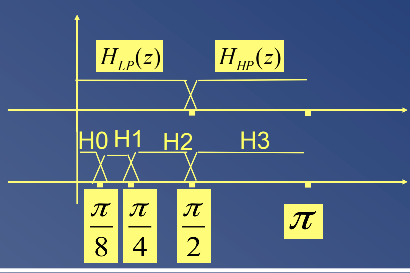
Wavelet Filter Banks & Wavelets

This can be built as a tree-structure, based on a 2-channel filter bank with $H_{LP}(z), H_{HP}(z)$



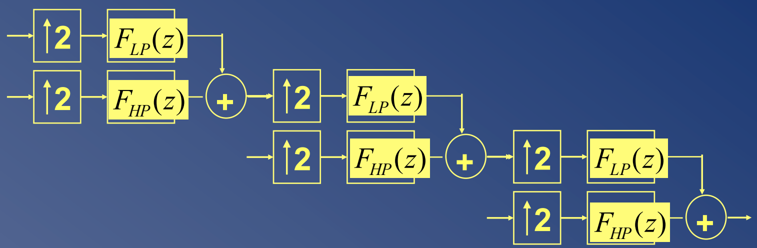
(by applying Noble identities)

$$H_3(z) = H_{HP}(z)$$
$$H_2(z) = H_{HP}(z^2).H_{LP}(z)$$
$$H_1(z) = H_{HP}(z^4).H_{LP}(z^2).H_{LP}(z)$$
$$H_0(z) = H_{LP}(z^4).H_{LP}(z^2).H_{LP}(z)$$



Wavelet Filter Banks & Wavelets

Similar synthesis bank can be constructed with $F_{LP}(z), F_{HP}(z)$



- If $H_{LP}(z), H_{HP}(z)$ and $F_{LP}(z), F_{HP}(z)$ form a PR FB (delay δ=0), then the complete analysis/synthesis structure is PR (why?)
- Example : `Haar' wavelet (after Alfred Haar) (compare to 2-channel DFT)

$$H_{Haar} = H_{HP} = \frac{1}{\sqrt{2}}(1 - z^{-1}) \qquad H_{LP} = \frac{1}{\sqrt{2}}(1 + z^{-1})$$

Wavelet Filter Banks & Wavelets

- Analysis bank corresponds to
  Discrete-Time Wavelet Transform (DTWT)

$$x_0[k] = \sum_{\bar{k}=-\infty}^{+\infty} u[\bar{k}].h_0[2^{N-1}.k - \bar{k}]$$

$x_k[n]$ = `DTWT-coefficients'

$$x_n[k] = \sum_{\bar{k}=-\infty}^{+\infty} u[\bar{k}].h_n[2^{N-n}.k - \bar{k}] \qquad n = 1,\ldots,N-1 \quad -\infty < k < +\infty$$

- With a corresponding (PR) synthesis filter bank, the reconstruction/synthesis formula (inverse DTWT) is

$$u[k] = \sum_{\bar{k}=-\infty}^{+\infty} x_0[\bar{k}].f_0[k - 2^{N-1}.\bar{k}] + \sum_{n=1}^{N-1} \sum_{\bar{k}=-\infty}^{+\infty} x_n[\bar{k}].f_n[k - 2^{N-n}.\bar{k}]$$

…to be compared to p.4 & p.11-12

Wavelet Filter Banks & Wavelets

- Reconstruction formula may be viewed as an expansion of u[k], using a set of basis functions (infinitely many)

$$b_{0,m}[k] = f_0[k - 2^{N-1}.m]$$
$$b_{n,m}[k] = f_n[k - 2^{N-n}.m] \qquad n = 1...N-1 \ , \ m = -\infty ... +\infty$$

- If the 2-channel filter bank is paraunitary, then this basis is orthonormal (which is a desirable property) :

$$\sum_{k=-\infty}^{+\infty} b_{n,m}[k].b^*_{n',m'}[k] = \delta(n-n').\delta(m-m')$$

=\`Orthonormal wavelet basis'

Wavelet Filter Banks & Wavelets

Skip this slide

Not treated here…

- \`Continuous wavelet transform' (CWT) of a continuous time function u(t)

$$x_{CWT}(p,q) = \frac{1}{\sqrt{|p|}} \int_{-\infty}^{\infty} u(t).h(\frac{q-t}{p})dt$$

h(t)=prototype
p,q are real-valued **continuous** variables
p introduces \`dilation' of prototype, q introduces \`shift' of prototype

- \`Discrete wavelet transform' (DWT) is CWT with discretized p,q

$$x_{DWT}(\bar{p},\bar{q}) = x_{CWT}(a^{\bar{p}}, a^{\bar{p}}.T\bar{q}) = a^{-\bar{p}/2} \int_{-\infty}^{\infty} u(t).h(T\bar{q} - a^{-\bar{p}}t).dt$$

T is sampling interval
p-bar, q-bar are real-valued **integer** variables mostly a=2

Overview

Time-Frequency Analysis

- Short-time Fourier Transform (STFT)
- Weighted OverLap-Add (WOLA)
- Wavelet Analysis & Wavelet Filter Banks

Time/Frequency Scaling of speech/audio signals

- Problem Statement & Approaches
- STFT-Based Time Scaling

Time Scaling & Frequency Scaling

- **Time Scaling**
  - Modify time domain attributes (tempo/duration) of a speech/audio signal, without modifying perceived frequency domain attributes (pitch), i.e. without introducing frequency distortion
  - Compression/expansion
  - Applications : movie post-synchronization (synchronization with video signal), dictation (synchronization with typing speed), fast rendering (e.g. in answering machines) ,…
- **Frequency Scaling** **(='dual' problem)**
  - Modify frequency domain attributes (pitch) of a speech/audio signal, without modifying perceived time domain attributes (tempo/duration)
  - A.k.a. `Pitch shifting'
  - Applications : games, karaoke, Doppler-effects in 3D audio…

Time Scaling & Frequency Scaling

- **Remember**: $u(t) \overset{FT}{\leftrightarrow} U(\Omega) \quad \Rightarrow \quad u(\alpha.t) \overset{FT}{\leftrightarrow} \frac{1}{|\alpha|}.U(\frac{\Omega}{\alpha})$

  scaling with alpha in the time domain ~ scaling with 1/alpha in frequency domain

  Hence straightforward scaling in the time or frequency domain does not provide a solution for the intended time or frequency scaling
- PS: Continuous-time signal u'(t)=u(α.t) is obtained by α times faster (α>1) or slower (α<1) playback of u(t).
- PS: Discrete-time signal (when sampling rate is to be kept constant and α can be non-integer) u'[k]=u[α.k] is obtained by re-sampling/digital interpolation of u[k] (ps: watch out for aliasing when α>1 !)
- Will consider only **Time Scaling**. Frequency Scaling can then be done as follows: If u^(t) is a time-scaled version of u(t) (e.g. duration increased by factor α, frequency attributes unchanged), then u^(α.t) is a frequency-scaled version of u(t) (i.e. duration unchanged, frequency attributes scaled by α).

Time Scaling

- **Different approaches**
  - Signal modeling based (`parametric')
    - Example : speech production model (LPC), …
    - Not treated here
  - Time/Frequency-analysis based (`non-parametric')
    - STFT-based
    - Wavelet transform based
    - …

  Will consider STFT-based approach only…

Overview

Time-Frequency Analysis

- Short-time Fourier Transform (STFT)
- Weighted OverLap-Add (WOLA)
- Wavelet Analysis & Wavelet Filter Banks

Time/Frequency Scaling of speech/audio signals

- Problem Statement & Approaches
- STFT-Based Time Scaling

STFT-Based Time Scaling

General procedure is…

1. Apply STFT to input signal u[k] (p.7, with (n,k) instead of $(k,\bar{k})$ for clarity)

$$U_{STFT}(e^{j\omega},n)=\sum_{k=-\infty}^{+\infty}u[k].w[n-k].e^{j\omega(n-k)}$$

=estimates frequency content in neighborhood of n (for n € grid)

2. Apply time axis transformation

$$U_{STFT}^{scaled}(e^{j\omega},n)=f\{U_{STFT}(e^{j\omega},n)\}$$

Will use simple transformation here…

$$U_{STFT}^{scaled}(e^{j\omega},n)=U_{STFT}(e^{j\omega},\alpha.n)$$

PS: requires interpolation ← if α.n is non-integer

3. Apply inverse STFT (to be defined)

$$u^{scaled}[k]=\ldots$$

STFT-Based Time Scaling

How to compute inverse STFT here?

- Usually, parameters are chosen such that $U_{STFT}^{scaled}(e^{j\omega},n)$ corresponds to an **oversampled** STFT (=WOLA)
- In an oversampled STFT (p.14), the number of `subband samples' is larger than number of full-band samples (=time-domain samples), hence STFT is `redundant' and so not straightforwardly invertible…
- In general, there does not exist any $u^{scaled}[k]$ for which $U_{STFT}^{scaled}(e^{j\omega},n)$ is the STFT (for all n)
  (≈ overdetermined set of linear equations)
- ➔Compute `maximally close' time-domain signal, in a **least squares** sense !

STFT-Based Time Scaling

Least Squares Problem is…

Given $U_{STFT}^{scaled}(e^{j\omega},n) \quad n \in \text{grid}$

compute time-domain sequence $\hat{u}[k]$ with

$$\hat{u}[k] \overset{STFT}{\leftrightarrow} \hat{U}_{STFT}(e^{j\omega},n)$$

such that

$$\sum_{n\in\text{grid}} \left\{ \int_0^{2\pi} \left| \hat{U}_{STFT}(e^{j\omega},n) - U_{STFT}^{scaled}(e^{j\omega},n) \right|^2 d\omega \right\} \quad \text{is minimized}$$

In words:
Compute a time-domain sequence $\hat{u}[k]$ such that its STFT is optimally close to $U_{STFT}^{scaled}(e^{j\omega},n)$ for all (i.e. summed over all) window positions n € grid

STFT-Based Time Scaling

For each n € grid define $u_n^{scaled}[k]$ as inverse DTFT(p.4) of $U_{STFT}^{scaled}(e^{j\omega},n)$ followed by a shift to time n

(to compensate for (*) p.7)

$$u_n^{scaled}[k+n] \overset{\text{inv.DTFT \& shift}}{=} \frac{1}{2\pi}\int_0^{2\pi} U_{STFT}^{scaled}(e^{j\omega},n)\cdot e^{j\omega k}.d\omega$$

so that…

$$U_{STFT}^{scaled}(e^{j\omega},n) \overset{\text{shift \& DTFT}}{=} \sum_{k=-\infty}^{+\infty} u_n^{scaled}[k].e^{j\omega(n-k)}$$

Then with…

$$\hat{U}_{STFT}(e^{j\omega},n) \overset{\substack{STFT\\(page 7)}}{=} \sum_{k=-\infty}^{+\infty} \hat{u}[k].w[n-k].e^{j\omega(n-k)}$$

the least-squares criterion…

$$\sum_{n\in \text{grid}} \{\int_0^{2\pi} \left|\hat{U}_{STFT}(e^{j\omega},n)-U_{STFT}^{scaled}(e^{j\omega},n)\right|^2 d\omega\}$$

can be replaced by…
(=Parceval's theorem)

$$\sum_{n\in \text{grid}} \{\sum_{k=-\infty}^{+\infty} \left|\hat{u}[k]\cdot w[n-k]-u_n^{\text{scaled}}[k]\right|^2\}$$

STFT-Based Time Scaling

Now

$$\sum_{n\in \text{grid}} \{\sum_{k=-\infty}^{+\infty} \left|\hat{u}[k]\cdot w[n-k]-u_n^{scaled}[k]\right|^2\} = \sum_{k=-\infty}^{+\infty} \{\sum_{n\in \text{grid}} \left|\hat{u}[k]\cdot w[n-k]-u_n^{scaled}[k]\right|^2\}$$

which corresponds to a separate least-squares problem for each $\hat{u}[k]$, i.e.

$$\sum_{n\in \text{grid}} \left|\hat{u}[k]\cdot w[n-k]-u_n^{scaled}[k]\right|^2$$

Least-squares solution is (see Chapter-8, p.7)

$$\hat{u}[k] = \frac{\sum_{n\in \text{grid}} w[n-k].u_n^{scaled}[k]}{\sum_{n\in \text{grid}} w[n-k]^2}$$

STFT-Based Time Scaling

$$\hat{u}[k] = \frac{\sum_{n \in \text{grid}} w[n-k].u_n^{scaled}[k]}{\sum_{n \in \text{grid}} w[n-k]^2}$$

In words: For each considered window position n € grid, compute inverse Fourier transform of $U_{STFT}^{scaled}(e^{j\omega}, n)$ and shift to position n, resulting in signals $u_n^{scaled}[k]$. Then u^[k] is a weighted sum of these sequences.

PS: Compare weights in this formula to (**) p.15 & try to establish link…!

PS: Procedure corresponds to STFT inversion if $U_{STFT}^{scaled}(e^{j\omega}, n)$ is a valid STFT (then $u_n^{scaled}[k] = w[n-k].\hat{u}[k], \quad \forall n$) and (**) p.15 provides PR.

STFT-Based Time Scaling

- **Method-1**: OLA synthesis (\`overlap-add')

Observe that Fourier- and inverse Fourier cancel each other…

$$u_n^{scaled}[k+n] \overset{p.35}{=} \frac{1}{2\pi}\int_0^{2\pi} U_{STFT}^{scaled}(e^{j\omega}, n)\cdot e^{j\omega k}.d\omega \overset{p.32}{=} \frac{1}{2\pi}\int_0^{2\pi} U_{STFT}(e^{j\omega}, \alpha.n)\cdot e^{j\omega k}.d\omega \overset{p.7}{=} u[k+\alpha.n].w[k]$$

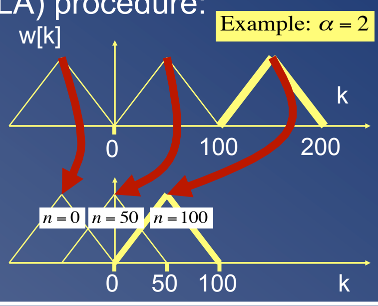
$$\Rightarrow u_n^{scaled}[(k-n)+n] = u[(k-n)+\alpha.n].w[(k-n)]$$

$$\Rightarrow u_n^{scaled}[k] = u[k+(\alpha-1).n].w[k-n]$$

This leads to simple time-domain (OLA) procedure:

$$\hat{u}[k] = \frac{\sum_{n \in \text{grid}} w[n-k].u_n^{scaled}[k]}{\sum_{n \in \text{grid}} w[n-k]^2} = \frac{\sum_{n \in \text{grid}} w[n-k]^2.u[k+(\alpha-1).n]}{\sum_{n \in \text{grid}} w[n-k]^2}$$

PS: requires interpolation if α.n is non-integer



STFT-Based Time Scaling

- **Method-1**: Does not seem to work well, because repositioning of signal segments destroys time structure (phase relation) across segments (example : applying procedure to a pure sine, results in harmonic distortion)

Hence in practice variants are used…

- **Method-2:** Only use magnitude information from STFT, add phase information based on an iterative procedure
- **Method-3:** Synchronized OLA, `SOLA'
  Reposition segments as in OLA, but then apply small additional re-alignment such that each re-aligned segment has maximum correlation with already formed portion of output signal (to restore phase relation across segments)
- **Method-4:** Pitch-Synchronous OLA, `PSOLA', …
- **Method-5:** Waveform Similarity OLA, `WSOLA', …
- etc… (details omitted)

Last Slide