KU LEUVEN

Digital Signal Processing 2

Les 4: Adaptieve filtering

Prof. dr. ir. Toon van Waterschoot

Faculteit Industriële Ingenieurswetenschappen
ESAT – Departement Elektrotechniek
KU Leuven, Belgium

Digital Signal Processing 2: Vakinhoud

- Les 1: Eindige woordlengte
- Les 2: Lineaire predictie
- Les 3: Optimale filtering
- Les 4: Adaptieve filtering
- Les 5: Detectieproblemen
- Les 6: Spectrale signaalanalyse
- Les 7: Schattingsproblemen 1
- Les 8: Schattingsproblemen 2
- Les 9: Sigma-Deltamodulatie
- Les 10: Transformatiecodering

Les 4: Adaptieve filtering

- **Linear adaptive filtering algorithms**

  RLS, steepest descent, LMS, …

- **Case study: Adaptive notch filters for acoustic feedback control**

  acoustic feedback problem, acoustic feedback control, adaptive notch filters, ANF-LMS algorithm …

Les 4: Adaptieve filtering

- **Linear adaptive filtering algorithms**

  S. V. Vaseghi, *Multimedia Signal Processing*

  – Ch. 9, “Adaptive Filters: Kalman, RLS, LMS”
    - Section 9.3, “Sample Adaptive Filters”
    - Section 9.4, “RLS Adaptive Filters”
    - Section 9.5, “The Steepest-Descent Method”
    - Section 9.6, “LMS Filter”
- **Case study: Adaptive notch filters for acoustic feedback control**

  Course notes:

  T. van Waterschoot, “Adaptive notch filters for acoustic feedback control”, *Course Notes Digital Signal Processing-2*, KU Leuven, Faculty of Engineering Technology, Dept. ESAT, Oct. 2014.

KU LEUVEN

Les 4: Adaptieve filtering

- **Linear adaptive filtering algorithms**

  RLS, steepest descent, LMS, …

- **Case study: Adaptive notch filters for acoustic feedback control**

  acoustic feedback problem, acoustic feedback control, adaptive notch filters, ANF-LMS algorithm …

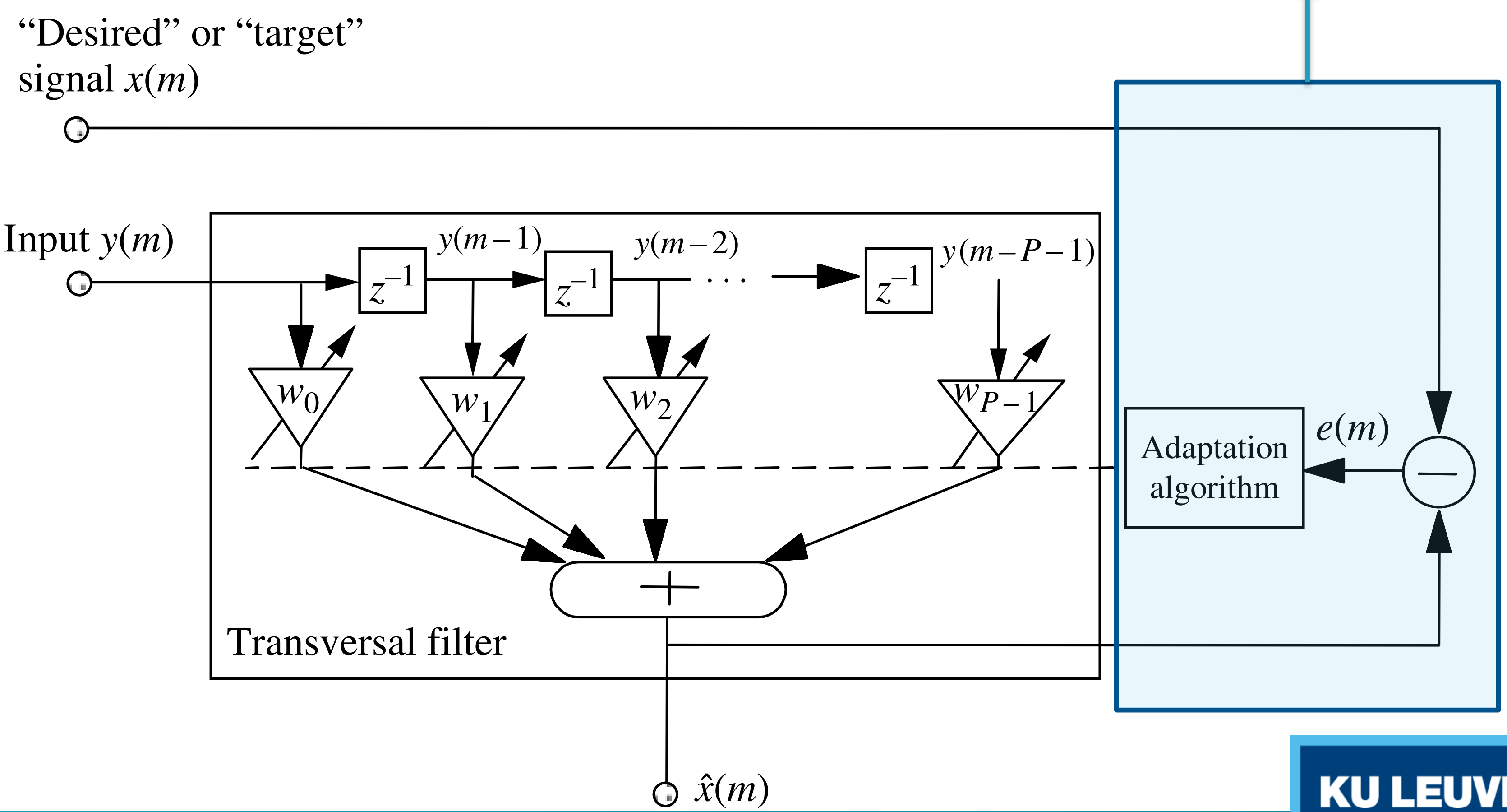
Linear adaptive filtering algorithms

- Adaptive filtering concept
- Recursive Least Squares (RLS) adaptive filters
- Steepest Descent method
- Least Mean Squares (LMS) adaptive filters
- Computational complexity

Adaptive filtering concept (1)

- **FIR adaptive filter**
  - signal flow graph

this part is different from Wiener filter



Adaptive filtering concept (2)

- **Adaptive filtering concept**
  - adaptive filter = time-varying optimal filter
  - filter coefficients are updated whenever new input/desired signal sample (or block of samples) is provided
  - general updating scheme:

optimal filter (time t) = optimal filter (time t-1) + adaptation gain * error

- **Design choices:**
  - FIR/IIR structure
  - filter order
  - cost function
  - adaptation algorithm

Linear adaptive filtering algorithms

- Adaptive filtering concept
- Recursive Least Squares (RLS) adaptive filters
- Steepest Descent method
- Least Mean Squares (LMS) adaptive filters
- Computational complexity

Recursive Least Squares (RLS) algorithm (1)

- **Online Wiener/LS filter implementation**
  - starting point: Wiener filter or least squares estimate

$$\mathbf{w} = \hat{\mathbf{R}}_{\mathbf{yy}}^{-1}\hat{\mathbf{r}}_{\mathbf{yx}} = (\mathbf{Y}^T\mathbf{Y})^{-1}\mathbf{Y}^T\mathbf{x}$$

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{P-1} \end{bmatrix}, \quad \mathbf{Y} = \begin{bmatrix} \mathbf{y}^T(0) \\ \mathbf{y}^T(1) \\ \vdots \\ \mathbf{y}^T(N-1) \end{bmatrix}, \quad \mathbf{y}(m) = \begin{bmatrix} y(m) \\ y(m-1) \\ \vdots \\ y(m-P+1) \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

  - how can we implement this filter in online applications?
    - at time m, only data $\{x(0), y(0), \ldots, x(m), y(m)\}$ are available
    - optimal filter coefficients **w** might be time-varying

Recursive Least Squares (RLS) algorithm (2)

- **Recursive time update of correlation matrix/vector**
  - consider the LS estimate at time *m*: $\mathbf{w}(m) = \hat{\mathbf{R}}_{\mathbf{yy}}^{-1}(m)\hat{\mathbf{r}}_{\mathbf{yx}}(m)$
  - the correlation matrix/vector can be computed recursively as

$$\hat{\mathbf{R}}_{\mathbf{yy}}(m) = \sum_{n=0}^{m} \mathbf{y}(n)\mathbf{y}^T(n) = \hat{\mathbf{R}}_{\mathbf{yy}}(m-1) + \mathbf{y}(m)\mathbf{y}^T(m)$$

$$\hat{\mathbf{r}}_{\mathbf{yx}}(m) = \sum_{n=0}^{m} \mathbf{y}(n)x(n) = \hat{\mathbf{r}}_{\mathbf{yx}}(m-1) + \mathbf{y}(m)x(m)$$

  - if optimal filter *w* is time-varying, use "forgetting" mechanism:

$$\hat{\mathbf{R}}_{\mathbf{yy}}(m) = \sum_{n=0}^{m} \lambda^{m-n}\mathbf{y}(n)\mathbf{y}^T(n) = \lambda\hat{\mathbf{R}}_{\mathbf{yy}}(m-1) + \mathbf{y}(m)\mathbf{y}^T(m)$$

$$0 \ll \lambda < 1$$

$$\hat{\mathbf{r}}_{\mathbf{yx}}(m) = \sum_{n=0}^{m} \lambda^{m-n}\mathbf{y}(n)x(n) = \lambda\hat{\mathbf{r}}_{\mathbf{yx}}(m-1) + \mathbf{y}(m)x(m)$$

KU LEUVEN

Recursive Least Squares (RLS) algorithm (3)

- **Matrix inversion lemma (MIL)**
  - since autocorrelation matrix needs to be inverted at each time *m*, recursive computation of inverse matrix is desired
  - define inverse autocorrelation matrix $\mathbf{\Phi}_{\mathbf{yy}}(m) = \hat{\mathbf{R}}_{\mathbf{yy}}^{-1}(m)$
  - Matrix inversion lemma:

$$\hat{\mathbf{R}}_{\mathbf{yy}}(m) = \lambda \hat{\mathbf{R}}_{\mathbf{yy}}(m-1) + \mathbf{y}(m)\mathbf{y}^T(m)$$

$$\Updownarrow$$

$$\mathbf{\Phi}_{\boldsymbol{yy}}(m) = \lambda^{-1}\mathbf{\Phi}_{\boldsymbol{yy}}(m-1) - \lambda^{-1}\mathbf{k}(m)\mathbf{y}^T(m)\mathbf{\Phi}_{\boldsymbol{yy}}(m-1)$$

with gain vector

$$\mathbf{k}(m) = \frac{\lambda^{-1}\mathbf{\Phi}_{\boldsymbol{yy}}(m-1)\mathbf{y}(m)}{1+\lambda^{-1}\mathbf{y}^T(m)\mathbf{\Phi}_{\boldsymbol{yy}}(m-1)\mathbf{y}(m)}$$

Recursive Least Squares (RLS) algorithm (4)

- **Recursive time update of filter coefficients**
  - plug recursive update for $\hat{\mathbf{r}}_{\mathbf{yx}}(m)$ into optimal filter solution:

$$\mathbf{w}(m) = \mathbf{\Phi}_{\mathbf{yy}}(m)[\lambda \mathbf{r}_{\mathbf{yx}}(m-1) + \mathbf{y}(m)x(m)]$$

  - replacing inverse autocorrelation matrix by

$$\mathbf{\Phi}_{\boldsymbol{yy}}(m) = \lambda^{-1}\mathbf{\Phi}_{\boldsymbol{yy}}(m-1) - \lambda^{-1}\mathbf{k}(m)\mathbf{y}^T(m)\mathbf{\Phi}_{\boldsymbol{yy}}(m-1)$$

results in a recursive time update of filter coefficients

$$\mathbf{w}(m) = \underbrace{\mathbf{\Phi}_{\mathbf{yy}}(m-1)\mathbf{r}_{\mathbf{yx}}(m-1)}_{\mathbf{w}(m-1)} + \mathbf{k}(m)\underbrace{\left[x(m) - \mathbf{y}^T(m)\mathbf{w}(m-1)\right]}_{e(m)}$$

KU LEUVEN

Recursive Least Squares (RLS) algorithm (5)

- **RLS algorithm**
  - input: $y(m), x(m)$
  - initialization: $\boldsymbol{\Phi}_{\boldsymbol{yy}}(0) = \delta \mathbf{I}, \mathbf{w}(0) = 0$
  - recursion (for m = 1,2,…):
    - adaptation gain: $\mathbf{k}(m) = \dfrac{\lambda^{-1}\boldsymbol{\Phi}_{\boldsymbol{yy}}(m-1)\mathbf{y}(m)}{1+\lambda^{-1}\mathbf{y}^T(m)\boldsymbol{\Phi}_{\boldsymbol{yy}}(m-1)\mathbf{y}(m)}$
    - error signal: $e(m) = x(m) - \mathbf{w}^T(m-1)\mathbf{y}(m)$
    - filter coefficient update: $\mathbf{w}(m) = \mathbf{w}(m-1) + \mathbf{k}(m)e(m)$
    - inverse input autocorrelation matrix update

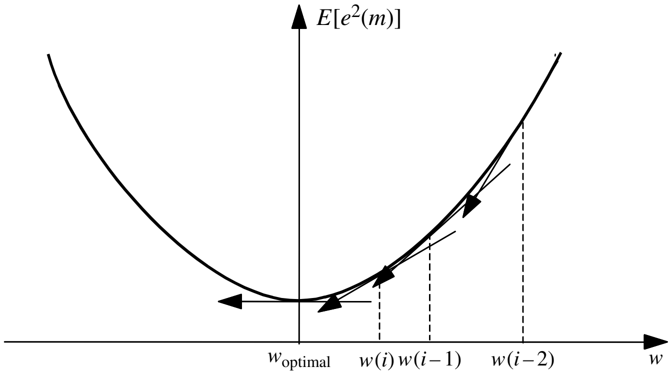
$$\boldsymbol{\Phi}_{\boldsymbol{yy}}(m) = \lambda^{-1}\boldsymbol{\Phi}_{\boldsymbol{yy}}(m-1) - \lambda^{-1}\mathbf{k}(m)\mathbf{y}^T(m)\boldsymbol{\Phi}_{\boldsymbol{yy}}(m-1)$$

Linear adaptive filtering algorithms

- Adaptive filtering concept
- Recursive Least Squares (RLS) adaptive filters
- Steepest Descent method
- Least Mean Squares (LMS) adaptive filters
- Computational complexity

Steepest Descent method (1)

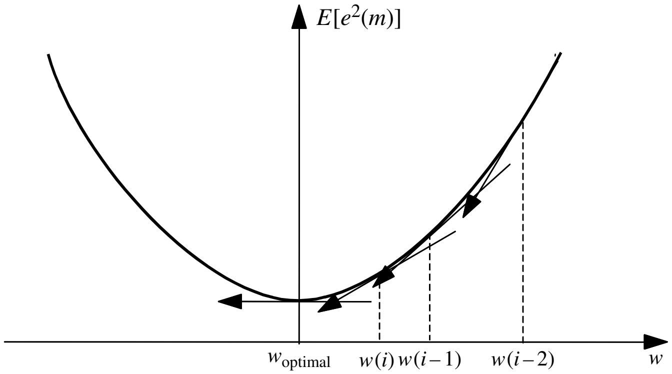
- **Steepest Descent (SD) method**
  - RLS algorithm
    - recursive implementation of LS optimal filter
    - calcuation of RLS adaptation gain is computationally expensive
  - Steepest Descent method
    - iterative implementation of Wiener filter
    - autocorrelation matrix inversion avoided to reduce complexity
  - idea: step-wise minimization of MSE cost function



Steepest Descent method (2)

- **Steepest Descent (SD) method**
  - idea: step-wise minimization of MSE cost function
  - optimal step = "steepest descent" direction
    = negative gradient direction

$$
\begin{aligned}
\mathbf{w}(m+1) &= \mathbf{w}(m) + \mu \left[-\frac{\partial E\{e^2(m)\}}{\partial \mathbf{w}(m)} \right] \\
&= \mathbf{w}(m) + \mu \left[\mathbf{r_{yx}} - \mathbf{R_{yy}} \mathbf{w}(m) \right]
\end{aligned}
$$



Steepest Descent method (3)

- **SD convergence & step size**
  - consider SD filter error vector w.r.t. optimal Wiener filter

$$\tilde{\mathbf{w}}(m) = \mathbf{w}(m) - \mathbf{w}_0$$

  - SD method produces filter estimates resulting in error update

$$\tilde{\mathbf{w}}(m+1) = [\mathbf{I} - \mu \mathbf{R}_{\mathbf{yy}}]\tilde{\mathbf{w}}(m)$$

  - SD convergence properties thus depend on
    - step size μ
    - input autocorrelation matrix $\mathbf{R}_{\mathbf{yy}}$

Steepest Descent method (4)

- **SD convergence & step size**
  - consider eigenvalue decomposition of autocorrelation matrix

$$\mathbf{R_{yy}} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$$

$$\mathbf{\Lambda} = \text{diag}\{\lambda_{\max}, \ldots, \lambda_{\min}\}$$

  - stable adaptation (i.e. $\tilde{\mathbf{w}}(m+1) < \tilde{\mathbf{w}}(m)$) is guaranteed if

$$0 < \mu < \frac{2}{\lambda_{\max}}$$

  - convergence rate is inversely proportional to

$$\text{eigenvalue spread} = \frac{\lambda_{\max}}{\lambda_{\min}}$$

  - *note:* eigenvalue spread = measure for magnitude of power spectrum variations

KU LEUVEN

Linear adaptive filtering algorithms

- Adaptive filtering concept
- Recursive Least Squares (RLS) adaptive filters
- Steepest Descent method
- Least Mean Squares (LMS) adaptive filters
- Computational complexity

Least Mean Squares (LMS) algorithm (1)

- **LMS filter**
  - Steepest Descent method
    - iterative implementation of Wiener filter
    - correlation matrix/vector assumed to be known a priori
  - Least Mean Squares (LMS) algorithm
    - recursive implementation of Steepest Descent method
    - gradient of instantaneous squared error i.o. mean squared error

$$
\begin{aligned}
\mathbf{w}(m+1) &= \mathbf{w}(m) + \mu \left[-\frac{\partial e^2(m)}{\partial \mathbf{w}(m)} \right] \\
&= \mathbf{w}(m) + \mu \left[\mathbf{y}(m) e(m) \right]
\end{aligned}
$$

- surprisingly simple algorithm!

KU LEUVEN

Least Mean Squares (LMS) algorithm (2)

- **LMS variations**
  - Leaky LMS algorithm
    - leakage factor $\alpha < 1$ results in improved stability and tracking

$$\mathbf{w}(m+1) = \alpha \mathbf{w}(m) + \mu \left[\mathbf{y}(m) e(m)\right]$$

  - Normalized LMS (NLMS) algorithm
    - step size normalization results in power-independent adaptation
    - small regularization parameter δ avoids division by zero

$$\mathbf{w}(m+1) = \mathbf{w}(m) + \frac{\mu}{\mathbf{y}^T(m)\mathbf{y}(m) + \delta} \left[\mathbf{y}(m) e(m)\right]$$

    - input power = $\mathbf{y}^T(m)\mathbf{y}(m) = \|\mathbf{y}(m)\|_2^2 = \sum_{k=0}^{P-1} y^2(m-k)$

Linear adaptive filtering algorithms

- Adaptive filtering concept
- Recursive Least Squares (RLS) adaptive filters
- Steepest Descent method
- Least Mean Squares (LMS) adaptive filters
- Computational complexity

Computational complexity

	RLS	LMS	Leaky LMS	NLMS
number of multiplications	$O(P^2)$	$3P+1$	$4P+1$	$4P+2$

Les 4: Adaptieve filtering

- **Linear adaptive filtering algorithms**

  RLS, steepest descent, LMS, …

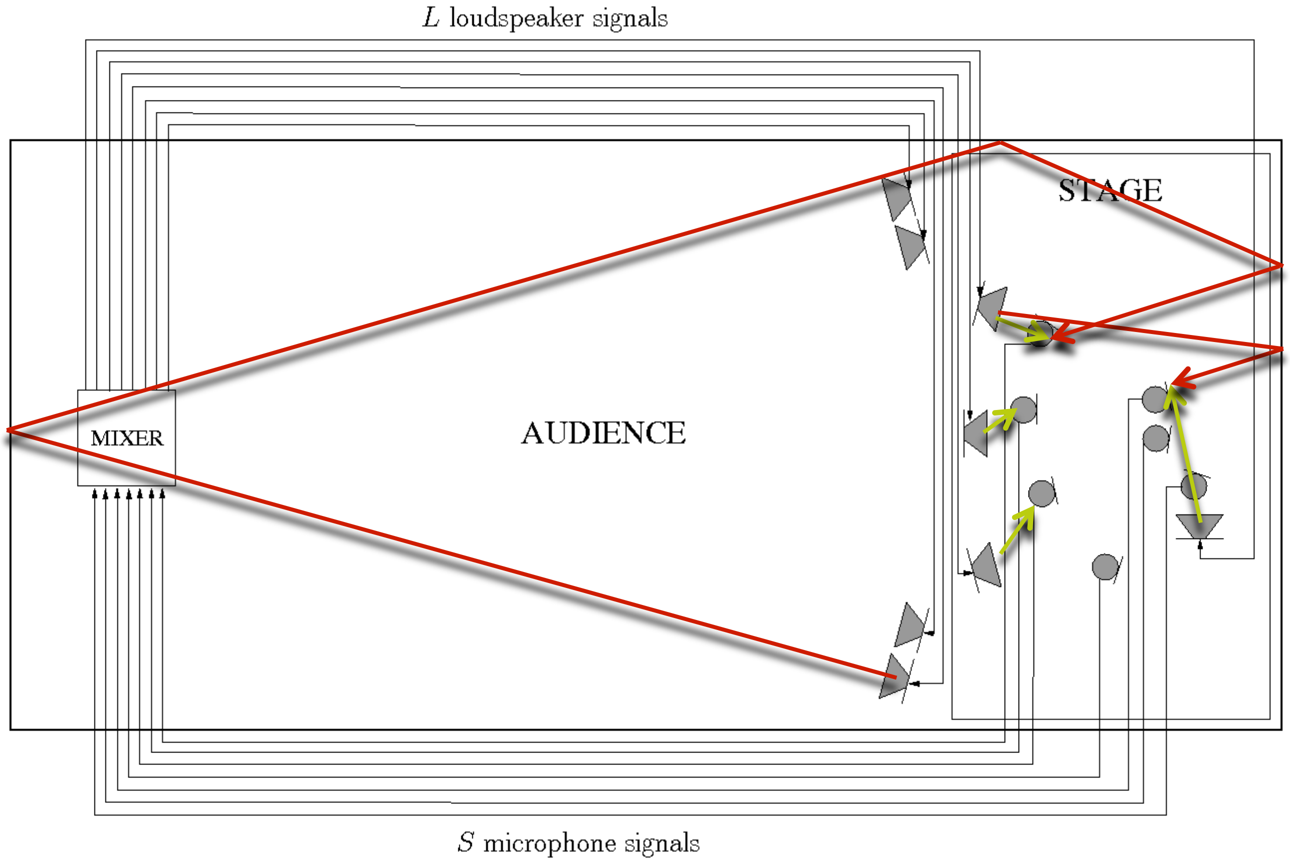
- **Case study: Adaptive notch filters for acoustic feedback control**

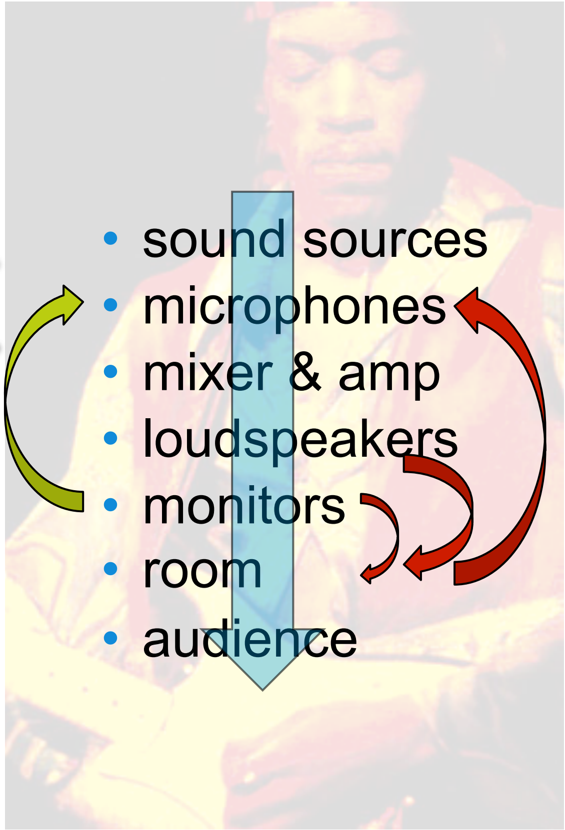
  acoustic feedback problem, acoustic feedback control, adaptive notch filters, ANF-LMS algorithm …

Case study: Adaptive notch filters

- Introduction
  - sound reinforcement
  - acoustic feedback
- Acoustic feedback control
- Adaptive notch filtering
- ANF-LMS algorithm

Introduction (1): Sound reinforcement (1)

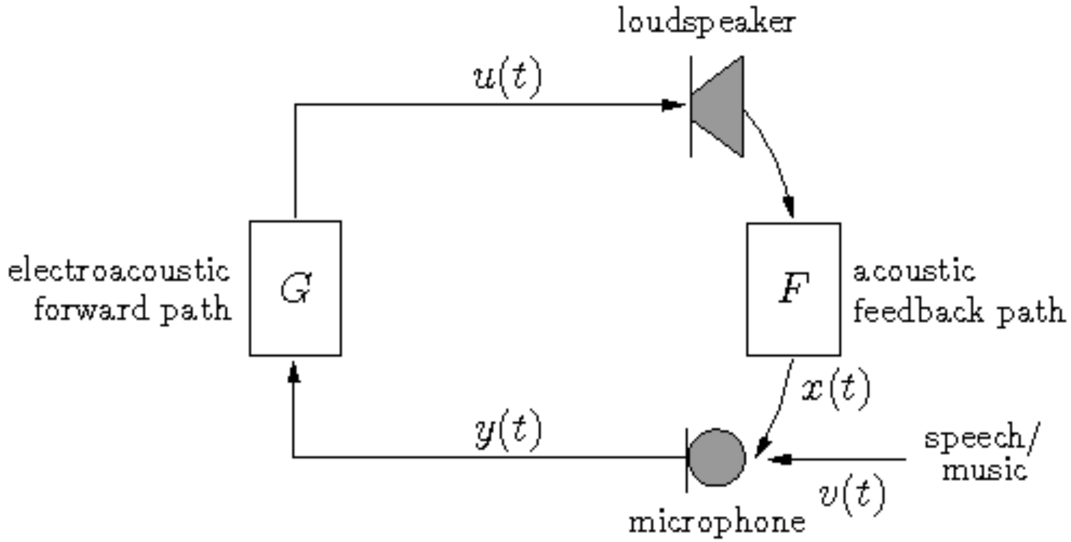




Goal: to deliver sufficiently high sound level
and best possible sound quality to audience

Introduction (2): Sound reinforcement (2)

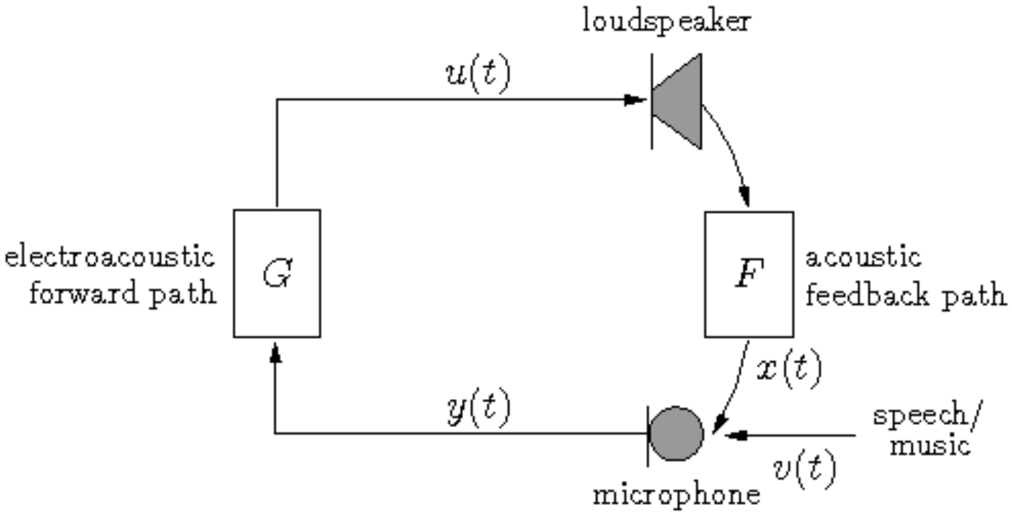
- We will restrict ourselves to the **single-channel case** (= single loudspeaker, single microphone)



KU LEUVEN

Introduction (3): Sound reinforcement (3)

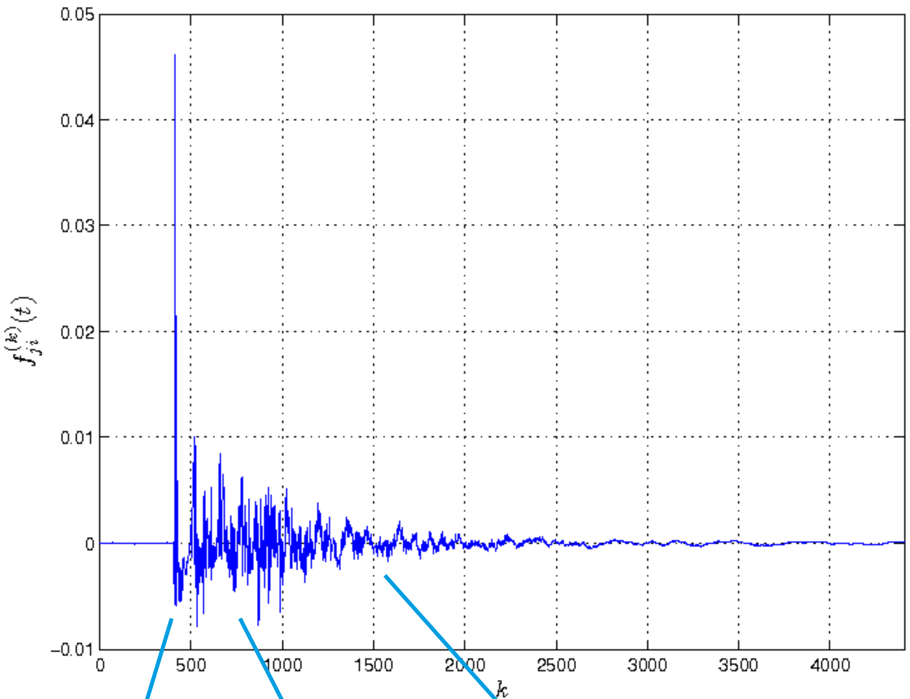
- **Assumptions:**
  - loudspeaker has linear & flat response
  - microphone has linear & flat response
  - forward path (amp) has linear & flat response
  - acoustic feedback path has linear response
- **But:** acoustic feedback path has **non-flat** response



Introduction (4): Sound reinforcement (4)

- Acoustic feedback path response: example room (36 m^3)

impulse response

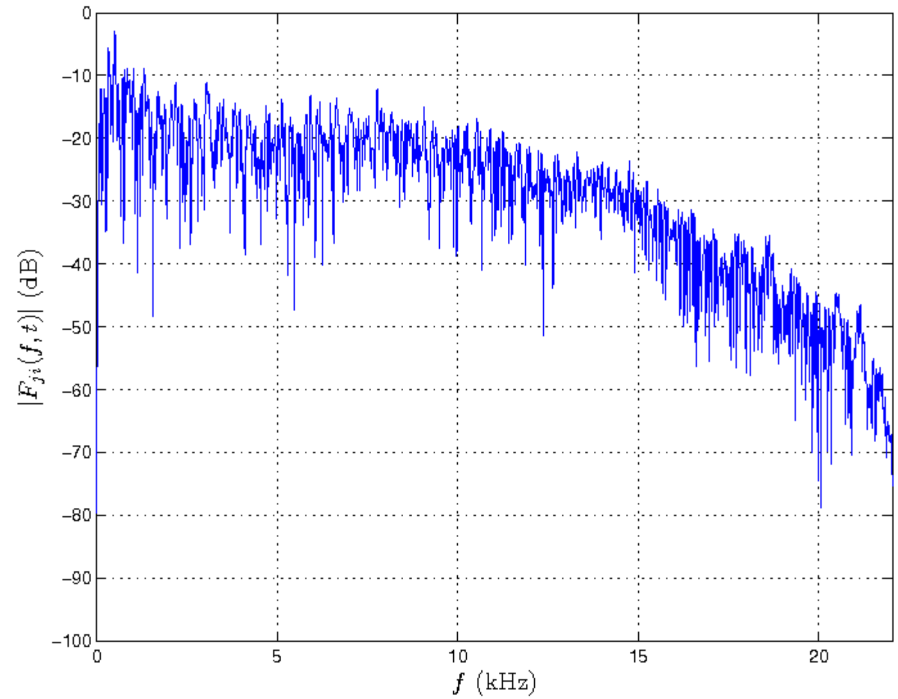


direct coupling

early reflections

diffuse sound field

frequency magnitude response



peaks/dips = anti-nodes/nodes of standing waves
peaks ~10 dB above average, and separated by ~10 Hz

Introduction (5): Acoustic feedback (1)

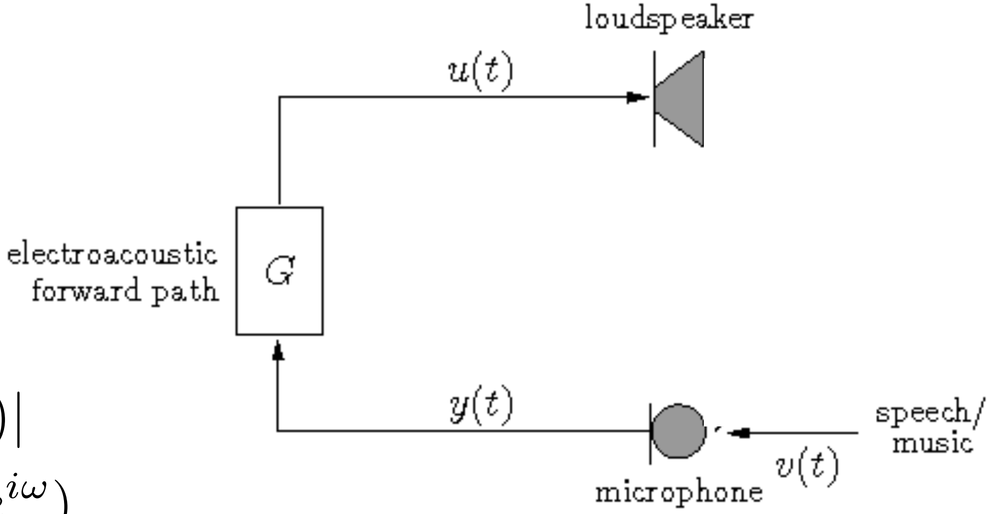
- **"Desired" system transfer function:**

$$\frac{U(z)}{V(z)} = G(z)$$

- **Closed-loop system transfer function:**

$$\frac{U(z)}{V(z)} = \frac{G(z)}{1 - G(z)F(z)}$$

  - spectral coloration
  - acoustic echoes
  - risk of instability



- **"Loop response":**
  - loop gain $|G(e^{i\omega})F(e^{i\omega})|$
  - loop phase $\angle G(e^{i\omega})F(e^{i\omega})$

KU LEUVEN

Introduction (6): Acoustic feedback (2)

- **Nyquist stability criterion:**
  - if there exists a radial frequency ω for which

    $$\begin{cases} |G(e^{i\omega})F(e^{i\omega})| \geq 1 \\ \angle G(e^{i\omega})F(e^{i\omega}) = n2\pi, n \in \mathbb{Z} \end{cases}$$

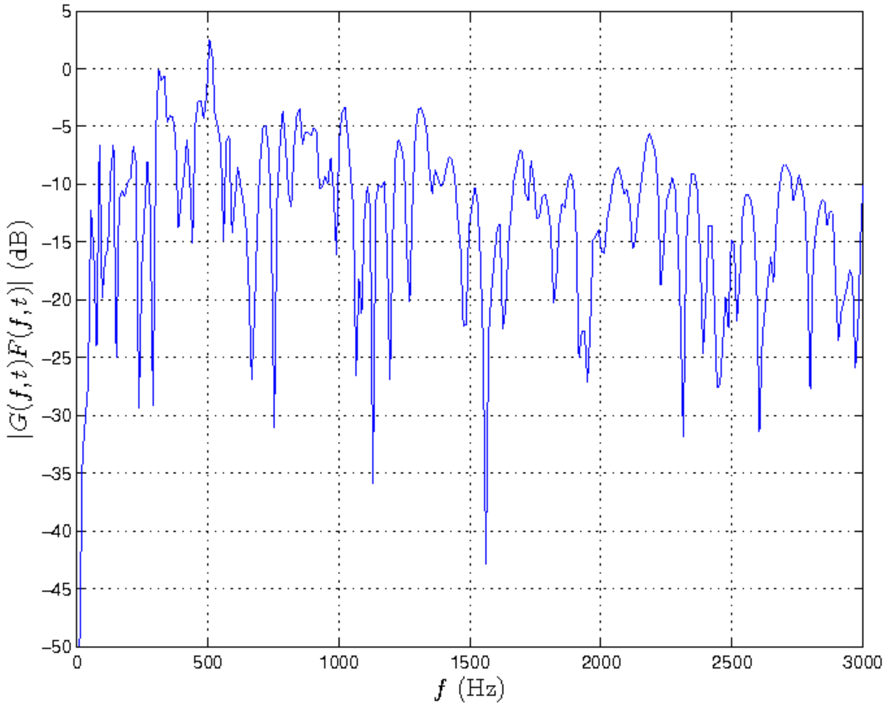
    then the closed-loop system is **unstable**
  - if the unstable system is excited at the critical frequency ω, then an oscillation at this frequency will occur **= howling**
- **Maximum stable gain (MSG):**
  - maximum forward path gain before instability
  - primarily determined by peaks in frequency magnitude response $|F(e^{i\omega})|$ of the room
  - 2-3 dB gain margin is desirable to avoid **ringing**

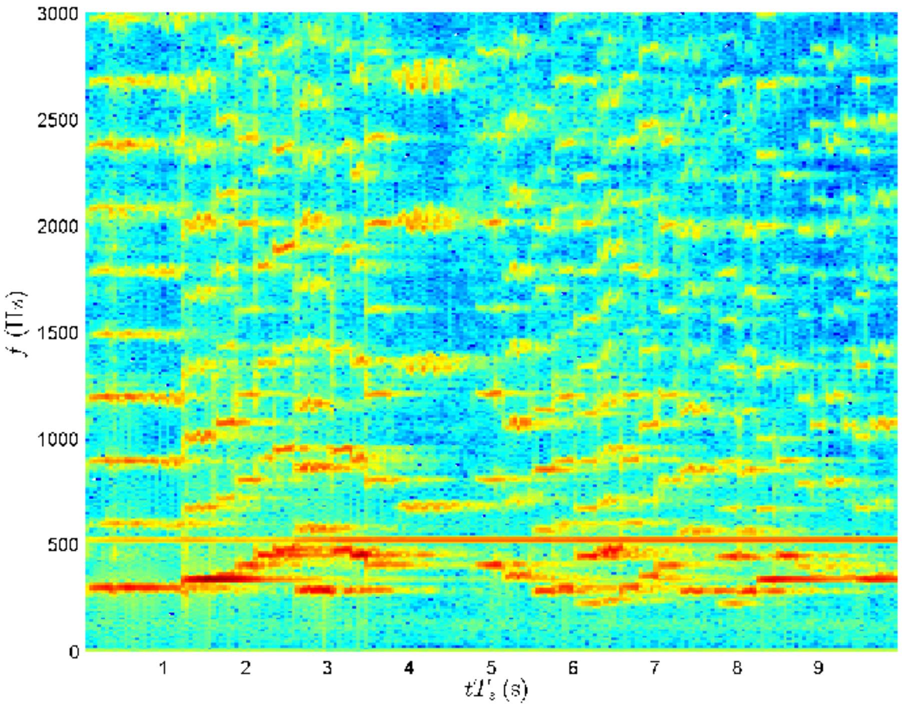
KU LEUVEN

Introduction (7): Acoustic feedback (3)

- **Example of closed-loop system instability:**

loop gain $|G(e^{i\omega})F(e^{i\omega})|$ loudspeaker spectrogram $U(e^{i\omega},t)$





KU LEUVEN

Case study: Adaptive notch filters

- Introduction
- Acoustic feedback control
- Adaptive notch filtering
- ANF-LMS algorithm

Acoustic feedback control (1)

- **Goal of acoustic feedback control**
  = to solve the acoustic feedback problem
  - either completely (to remove acoustic coupling)
  - or partially (to remove howling from loudspeaker signal)
- **Manual acoustic feedback control:**
  - proper microphone/loudspeaker selection & positioning
  - a priori room equalization using 1/3 octave graphic EQ filters
  - ad-hoc discrete room modes suppression using notch filters
- **Automatic acoustic feedback control:**
  - no intervention of sound engineer required
  - different approaches can be classified into four categories

Acoustic feedback control (2)

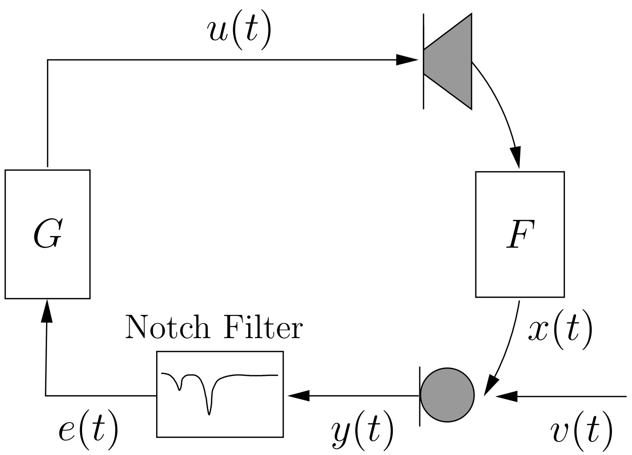
1. phase modulation (PM) methods
   - smoothing of “loop gain” (= closed-loop magnitude response)
   - phase/frequency/delay modulation, frequency shifting
   - well suited for reverberation enhancement systems (low gain)
2. spatial filtering methods
   - (adaptive) microphone beamforming for reducing direct coupling
3. gain reduction methods
   - (frequency-dependent) gain reduction after howling detection
   - most popular method for sound reinforcement applications
4. room modeling methods
   - adaptive inverse filtering (AIF): adaptive equalization of acoustic feedback path response
   - adaptive feedback cancellation (AFC): adaptive prediction and subtraction of feedback (≠howling) component in microphone signal

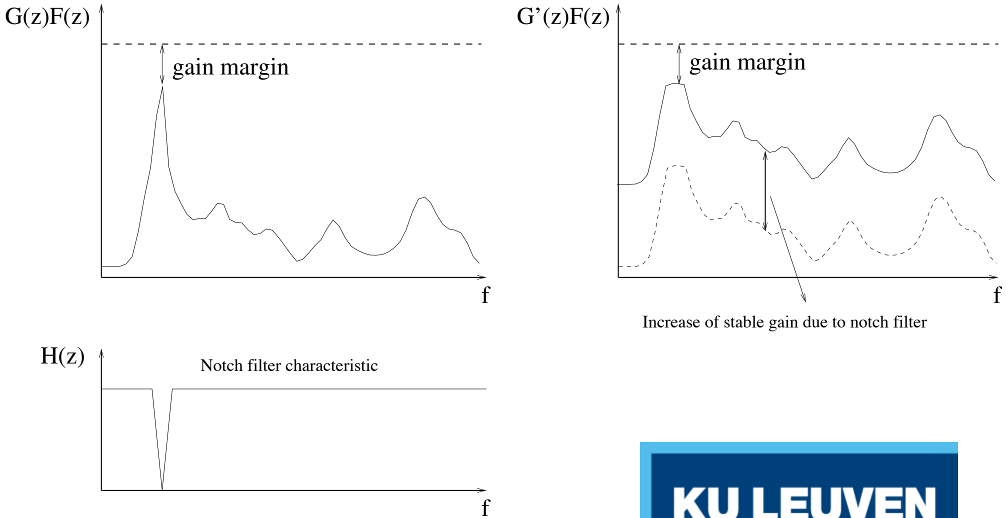
Case study: Adaptive notch filters

- Introduction
- Acoustic feedback control
- Adaptive notch filtering
- ANF-LMS algorithm

Adaptive notch filtering (1)

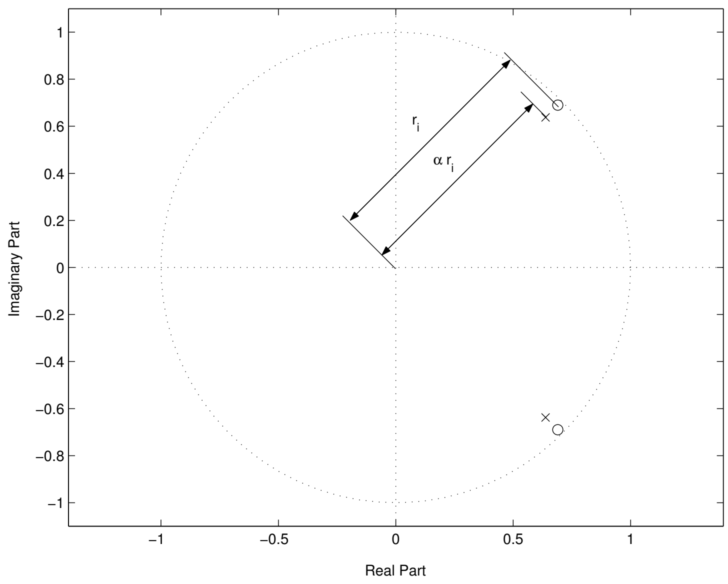
- **Gain reduction methods**
  - automation of the actions a sound engineer would undertake
- **Classification of gain reduction methods**
  - automatic gain control (full-band gain reduction)
  - automatic equalization (1/3 octave bandstop filters)
  - notch filtering (NF) (1/10-1/60 octave filters)

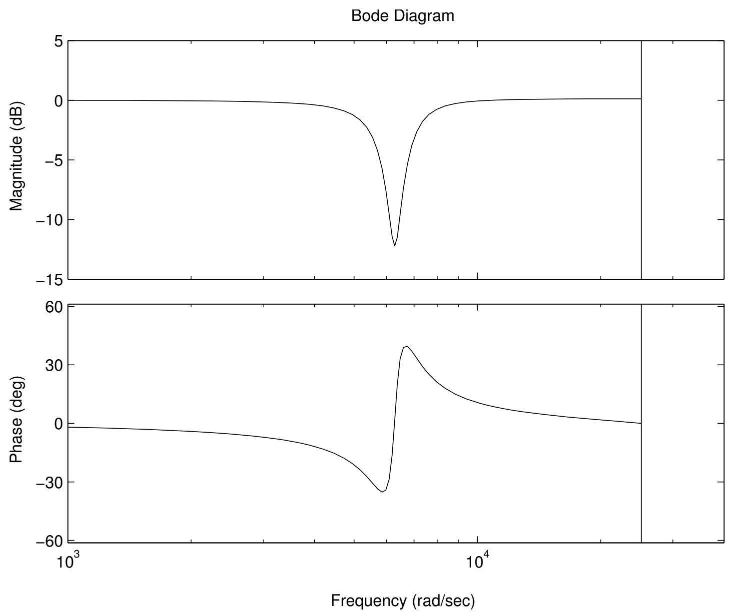




Adaptive notch filtering (2)

- **Adaptive notch filter**
  - filter that automatically finds & removes narrowband signals
  - based on constrained second-order pole-zero filter structure
  - **constraint 1:** poles and zeros lie on same radial lines





$$z_i = r_i e^{\pm j\omega_i}$$

$$p_i = \alpha r_i e^{\pm j\omega_i}$$

Adaptive notch filtering (3)

- **ANF transfer function**
  - cascade of constrained second-order pole-zero filters:

$$H(z^{-1}) = \frac{\prod_{i=1}^{2n}(1 - z_i z^{-1})}{\prod_{i=1}^{2n}(1 - \alpha z_i z^{-1})} \qquad \text{where } 0 \leq \alpha < 1$$

$$= \frac{1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_{2n} z^{-2n}}{1 + \alpha a_1 z^{-1} + \alpha^2 a_2 z^{-2} + \ldots + \alpha^{2n} a_{2n} z^{-2n}}$$

  - **constraint 2:** zeros are forced to lie on unit circle

$$H(z^{-1}) = \frac{1 + a_1 z^{-1} + \ldots + a_n z^{-n} + \ldots + a_1 z^{-2n+1} + z^{-2n}}{1 + \rho a_1 z^{-1} + \ldots + \rho^n a_n z^{-n} + \ldots + \rho^{2n-1} a_1 z^{-2n+1} + \rho^{2n} z^{-2n}}$$

$$= \frac{A(z^{-1})}{A(\rho z^{-1})}$$

  - "pole radius" ρ = "debiasing parameter" α

KU LEUVEN

Adaptive notch filtering (4)

- **ANF coefficient estimation**
  - coefficient vector $\theta = [a_1 \;\; a_2 \;\; \ldots \;\; a_n]^T$
  - least squares (LS) cost function:

$$
\begin{aligned}
\hat{\theta} &= \arg\min_{\theta} V_N(\theta) \\
&= \arg\min_{\theta} \sum_{t=1}^{N} e^2(\theta, t) \\
&= \arg\min_{\theta} \sum_{t=1}^{N} \left[\frac{A(\theta, z^{-1})}{A(\theta, \rho z^{-1})} y(t) \right]^2
\end{aligned}
$$

Case study: Adaptive notch filters

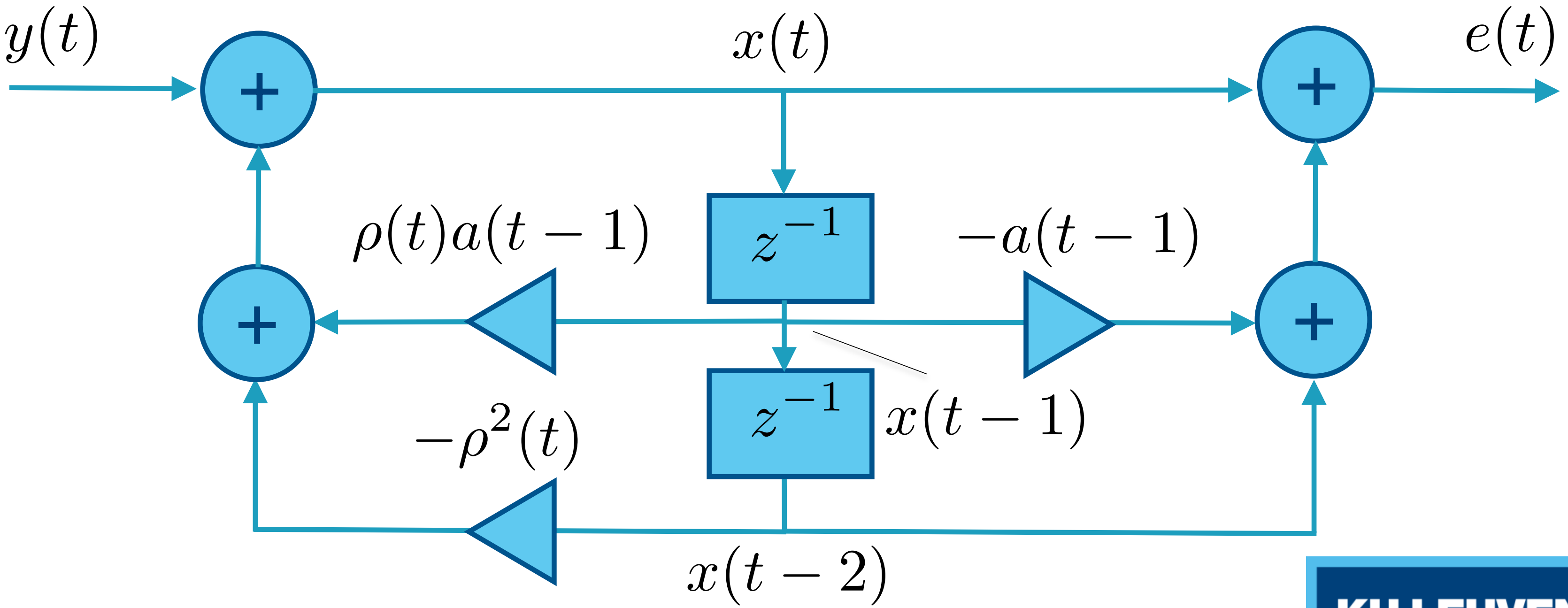
- Introduction
- Acoustic feedback control
- Adaptive notch filtering
- ANF-LMS algorithm

KU LEUVEN

ANF-LMS algorithm (1)

- **2^{nd} order constrained ANF implementation**
  - Direct-Form II implementation of second-order ANF

$$\begin{aligned} x(t) &= y(t) + \rho(t)a(t-1)x(t-1) - \rho^2(t)x(t-2) \\ e(t) &= x(t) - a(t-1)x(t-1) + x(t-2) \end{aligned}$$



ANF-LMS algorithm (2)

- **ANF filter coefficient update**

$$\begin{aligned} x(t) &= y(t) + \rho(t)a(t-1)x(t-1) - \rho^2(t)x(t-2) \\ \boxed{e(t)} &\boxed{= x(t) - a(t-1)x(t-1) + x(t-2)} \end{aligned}$$

  - adaptation strategy: only FIR portion of filter is adapted, coefficients are then copied to IIR portion of filter
  - LMS filter coefficient update:

$$\begin{aligned} a_{upd}(t) &= \arg\min_a (e^2(t)) \\ &= \arg\left(\frac{d}{da}e^2(t) = 0\right) \\ &= \arg\left(2e(t)\frac{d}{da}e(t) = 0\right) \\ &= \arg\Big(2e(t)(-x(t-1)) = 0\Big) \end{aligned}$$

ANF-LMS algorithm (2)

- **2^{nd} order ANF-LMS algorithm**

Algorithm 1 : 2^{nd} order ANF-LMS algorithm

Input step size μ, initial pole radius $\rho(0)$, final pole radius $\rho(\infty)$, exponential decay time constant λ, input data $\{y(t)\}_{t=1}^{N}$, initial conditions $x(0), x(-1), a(0)$

Output 2^{nd} order ANF parameter $\{a(t)\}_{t=1}^{N}$

1: **for** $t = 1, \ldots, N$ **do**

2: $\quad \rho(t) = \lambda\rho(t-1) + (1-\lambda)\rho(\infty)$

3: $\quad x(t) = y(t) + \rho(t)a(t-1)x(t-1) - \rho^2(t)x(t-2)$

4: $\quad e(t) = x(t) - a(t-1)x(t-1) + x(t-2)$

5: $\quad a(t) = a(t-1) + 2\mu e(t)x(t-1)$

6: **end for**
