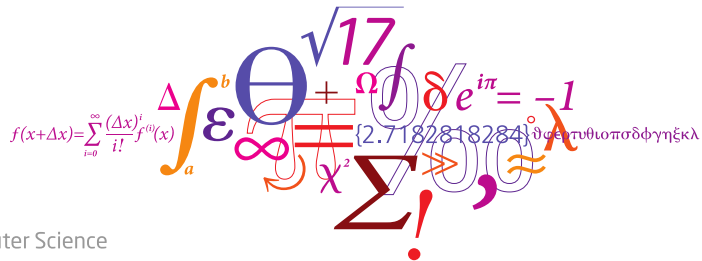
02471 Machine Learning for Signal Processing

Stochastic processes and linear filtering

Tommy Sonne Alstrøm

Cognitive Systems section



DTU Compute
Department of Applied Mathematics and Computer Science

Outline

- Course admin
- Last Week
- Frequency analysis
- The Wiener filter
- Typical applications of filtering
- Linear filtering using mean-squared error
- Stochastic Processes
- Next Week

Material: ML 2.4, 4.1–4.3, 4.5, 4.7

Administrative notes

- Problem set 1 deadline Sunday, 22/9.
  - Approved/Not Approved. Not approved solutions can hand in again, deadline 6/10.
  - In groups of 2 (or 3 max), everyone needs to hand in individually, but clearly write on DTU Learn who you worked with. In this case, you can hand in identical files.
  - ONLY upload PDFs.
  - We aim to have corrections ready by Thursday 26/9.

Feedback

- Type of feedback
  - Mention one thing that worked?
  - Mention one thing should be improved (both in current lecture and last weeks exercise)?
  - Mention one thing you would change if you gave the lecture.
- Some people are dropping the course . If you do, please let me know the reasons. Approx 15 people left the course.

Last Week

Derivatives with respect to vectors (exercise 2.1)

Two commonly used rules

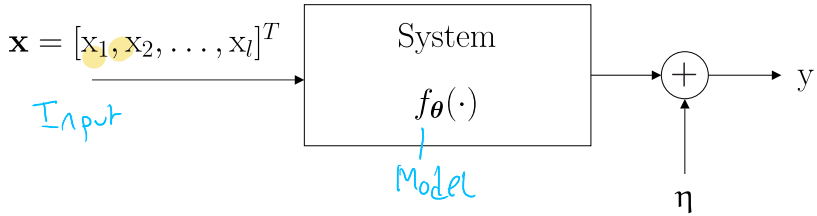
$$\frac{\partial \boldsymbol{a}^T \boldsymbol{x}}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{x}^T \boldsymbol{a}}{\partial \boldsymbol{x}} = \boldsymbol{a}$$

$$\frac{\partial \boldsymbol{x}^T A \boldsymbol{x}}{\partial \boldsymbol{x}} = \left(A + A^T\right) \boldsymbol{x}$$

Things to remember

- It is an extension of normal calculus. The derivatives are simply organized in vectors and matrices.
- When you look-up rules, it is important to be aware of the vector layout convention.
- We will use matrix calculus regularly in the course to derive update rules.
- Check Appendix A for useful rules and identities when doing exercises.
- $(AB)^T = B^T A^T$, $(B^T)^T = B$!

What is parameter estimation



We considered the general system

$$\mathrm{y} = f_{\boldsymbol{\theta}}(\boldsymbol{x}) + \eta$$

And limited the examples to the linear model that lead to the linear regression model

$$f_\theta(x) = \theta_0 + \theta_1 x_1 + \cdots + \theta_l x_l$$

$$\mathrm{y} = \boldsymbol{\theta}^T \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix} + \eta$$

Remarks on notation:

- y is a random variable, y is a scalar.
- $\mathbf{x}$ is a random vector, $\boldsymbol{x}$ is a vector
- $\boldsymbol{\theta}$ is a parameter vector.
- η is a random variable (noise).

Estimate θ from data?

Approaches:

- Point estimate of $\boldsymbol{\theta}$, denoted $\hat{\boldsymbol{\theta}}$, by loss functions (such as the squared error).
- Maximum likelihood point estimate of $\boldsymbol{\theta}$, denoted $\hat{\boldsymbol{\theta}}_{\text{ML}}$, where the underlying data generation process is considered by a choice of probability density function to model the data and noise, $p(\mathcal{D}|\boldsymbol{\theta})$.

Week 9

Estimating θ from loss functions

Least-Squares (LS) loss function

$$J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N}(y_n - f_{\boldsymbol{\theta}}(\boldsymbol{x}_n))^2$$

$$\boldsymbol{\theta}_* = \arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

The solution then becomes

LS Estimate

$$\boldsymbol{\theta}_* = (X^T X)^{-1} X^T \boldsymbol{y}$$

Linear Regr.

Biased estimation

Theory that justify biased estimation

Define the following biased estimator:

$$\hat{\theta}_b = (1+\alpha)\hat{\theta}_u$$

Choose α in the range

$$-\frac{2\mathrm{MSE}(\hat{\theta}_u)}{\mathrm{MSE}(\hat{\theta}_u)+\theta_o^2} < \alpha < 0$$

Then $|\hat{\theta}_b| < |\hat{\theta}_u|$ and

$$\mathrm{MSE}(\hat{\theta}_b) < \mathrm{MSE}(\hat{\theta}_u)$$

The Ridge regression approach is an example of a norm shrinking loss the results in biased estimator:

Ridge regression

$$\mathcal{L}(\boldsymbol{\theta}, \lambda) = \sum_{n=1}^{N} \left(y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2 + \lambda \|\boldsymbol{\theta}\|_2^2$$

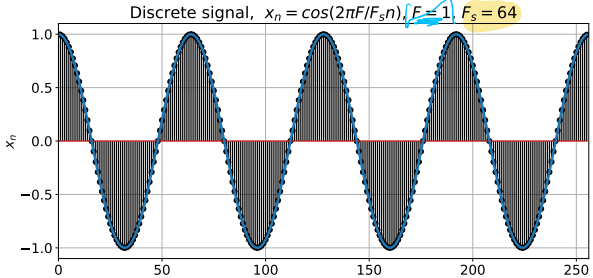
Weeks

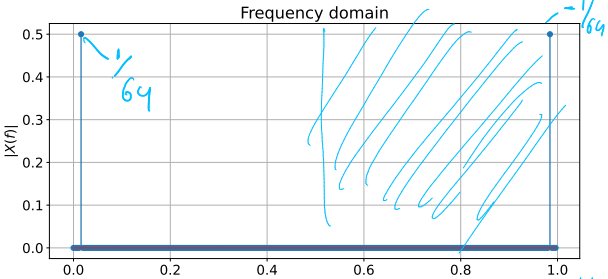
Last week summary

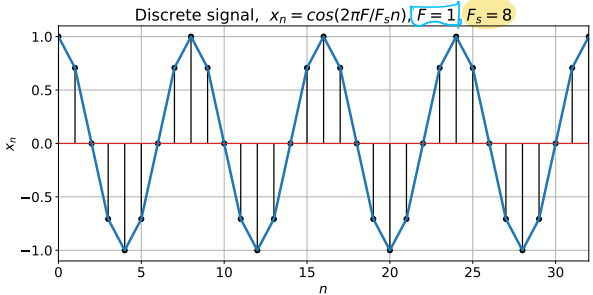
- Matrix calculus is equivalent to ordinary calculus and makes your life easier (promise!).
- If we focus on the MSE, we can decompose the loss into a bias and variance term that offers a direct interpretation of the source of error.
- Biased estimation can, with suitable model choices, result in a lower MSE compared to unbiased estimation.

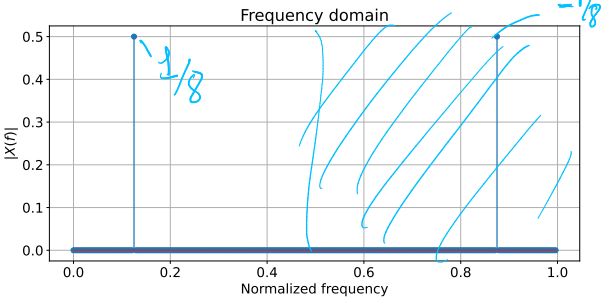
Frequency analysis

Frequency analysis

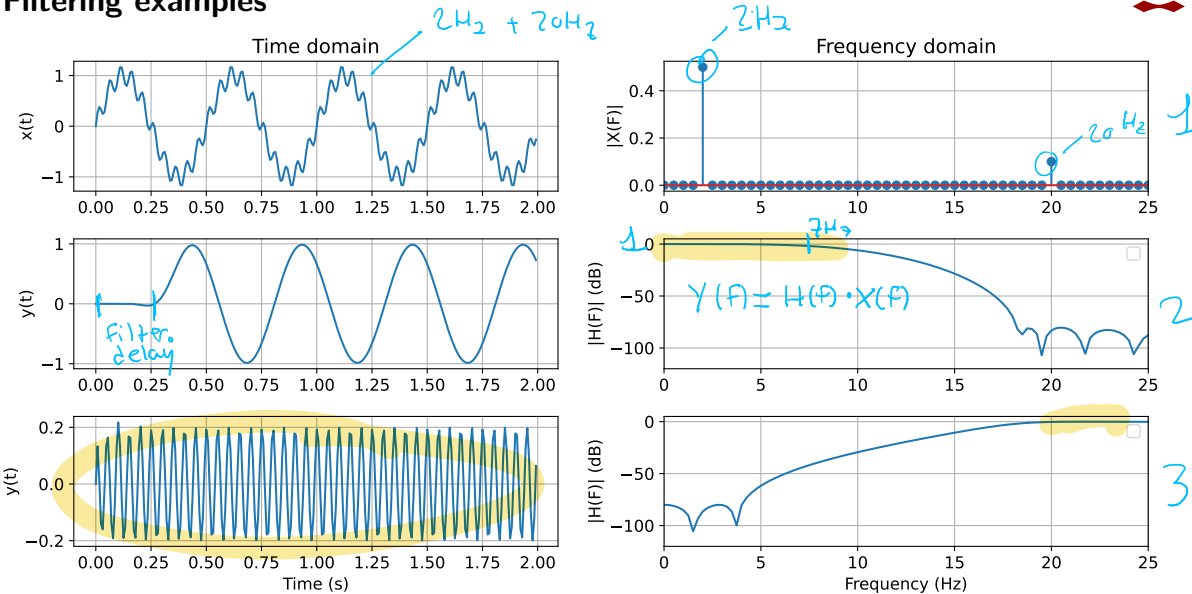






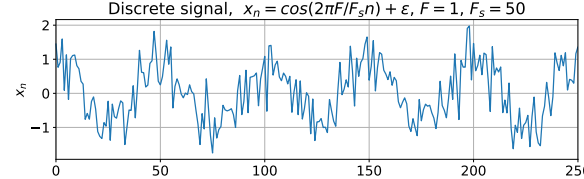


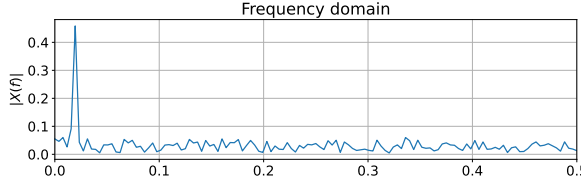
Filtering examples

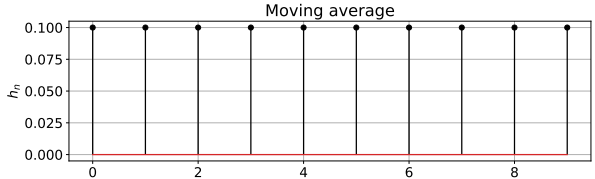


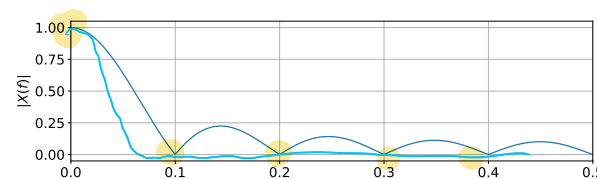
Husk at nævne DB

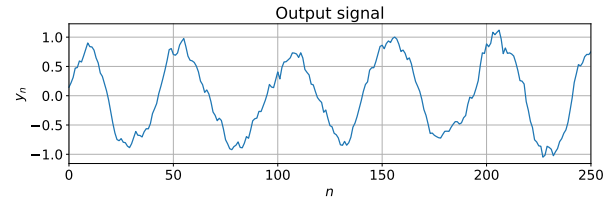
Filtering example – moving average

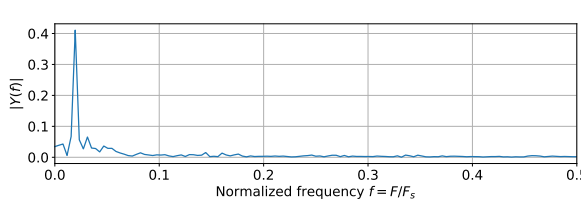








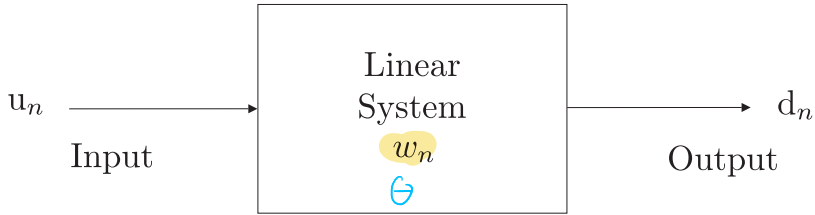




The Wiener filter

This week

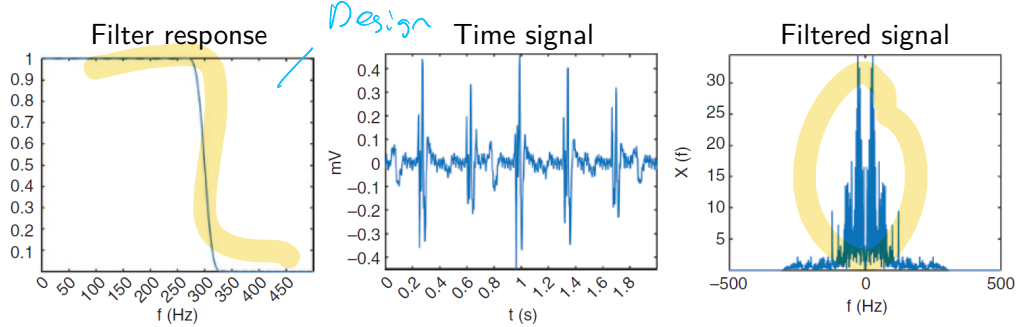
This weeks system



The input signal u_n, also written as $u_n = u(n)$, or often in DSP literature denoted as $x(n)$, is now considered to be a stochastic process.

Two approaches to filter design

Design filter with specific frequency response:

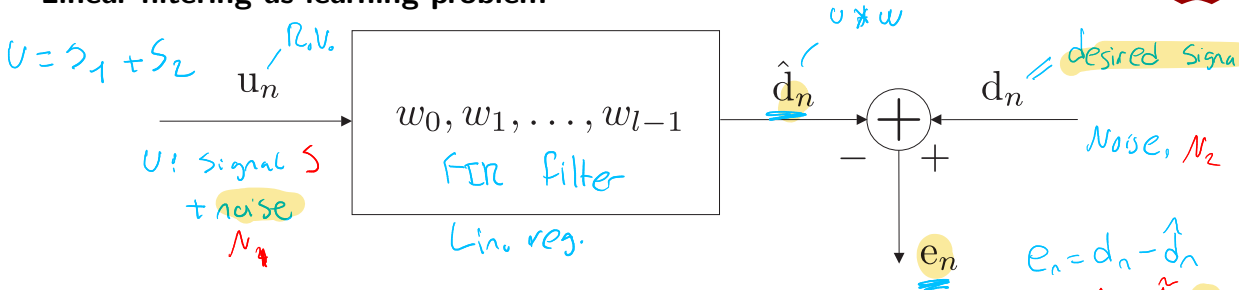


Alternative idea!

Create a filter as a learning problem, ie. one that minimized a cost function.

Discuss pros and cons, where could the two approached be applicable?

Linear filtering as learning problem



The filter weights are learned such that the error is minimized

$$\mathrm{e}_n = \mathrm{d}_n - \hat{\mathrm{d}}_n$$

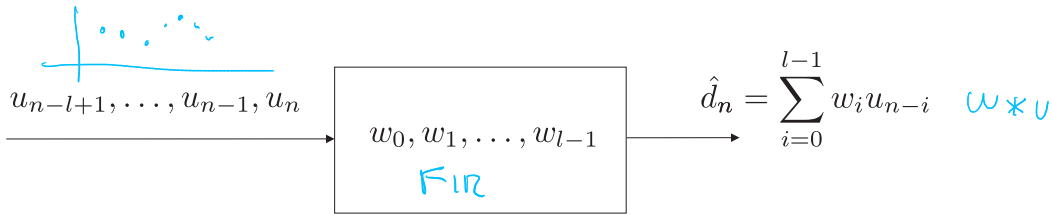
If we use an LTI filter and minimize the mean squared error, the filter is called a Wiener filter.

$$\hat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \mathbb{E}[\mathrm{e}_n] = \mathbb{E}\left[\left(\mathrm{d}_n - \hat{\mathrm{d}}_n\right)^2\right]$$

To use such filter, we need to specify a desired signal, d_n.

Linear filtering as learning problem

Once the filter coefficients are learned, the filter works like an ordinary linear time-invariant filter, i.e.



A word on notation:

- u_n is a stochastic process.
- u_n is a realization (concrete measurements) of the stochastic process, u_n.

More on this topic later.

Summary

There are two basic approaches to filter design:

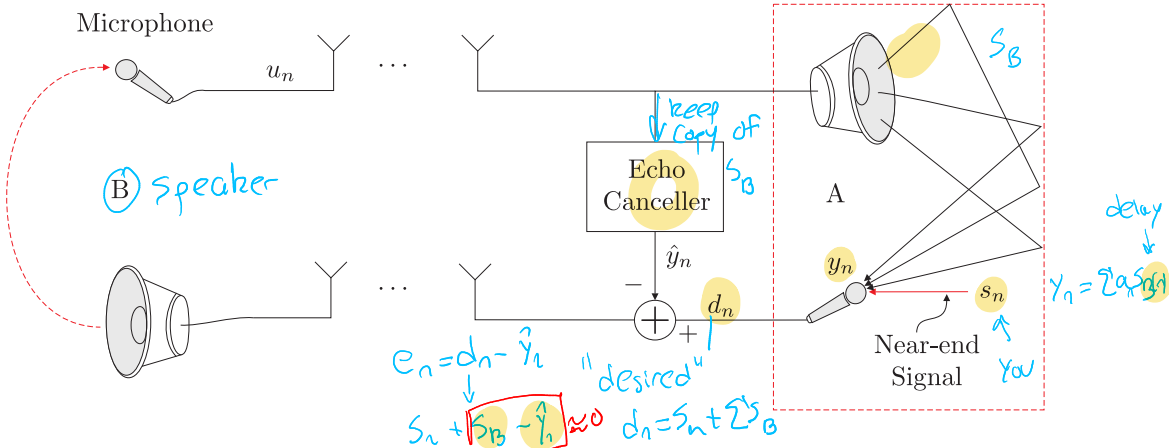
- Design a filter with the correct frequency response.
- Setup a learning problem, and learn the optimal filter from the data.
  - The linear filter the minimizes the mean squared error (MSE) is called the Wiener filter.

Typical applications of filtering

Typical applications of filtering (ML sec 4.7)

- Echo cancellation
- Noise cancellation
- System identification
- Interference cancellation
- Channel equalization

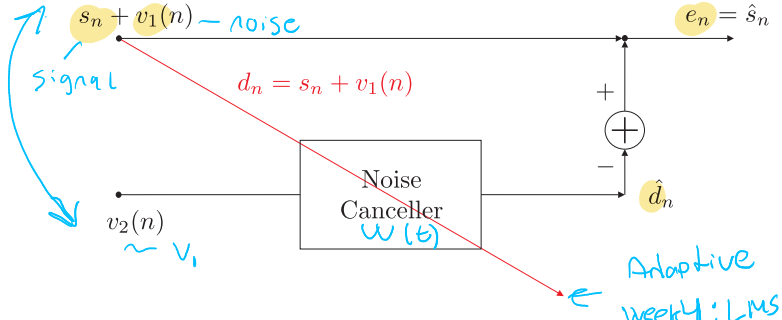
Echo cancellation



Application; e.g. video conference or telephone call with load speaker (location A). Without echo canceling, the audio would loop back to location B.

Noise cancellation

Goal: Remove unwanted noise $v_1(n)$ from a signal, $s_n + v_1(n)$



We have access to:

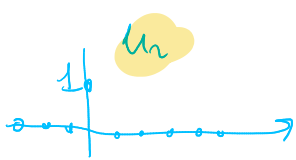
- The signal of interest, $s_n + v_1(n)$, where we want to suppress $v_1(n)$.
- A noise source $v_2(n)$, that is statistically related to $v_1(n)$ but statistically unrelated to s_n. Note: the signals $s_n + v_1(n)$ and $v_2(n)$ can change place.

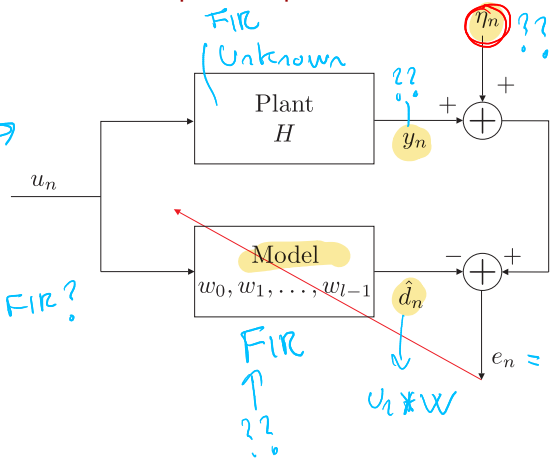
We will work with this setup in the exercise.

System identification

$Y_n = U_n * H = H$ $\quad Y_0 = H_0, \; Y_1 = H_1$

Goal: Model the impulse response of the unknown plant H





η_n ?? $\simeq E[\eta_n] = 0$

Aim: Analyze it, Simplify H

d_n desired signal $\quad d_n = y_n + \eta_n$

What is w if H is an unknown FIR?

$w = H$

We have access to:

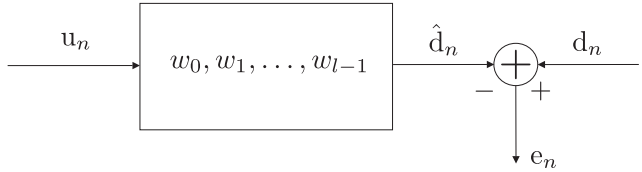
- The input signal u_n
- The noisy output d_n

Summary

- The linear filter setup has been successfully applied in many areas.
- We need to either know the statistical properties of our signal, or have training data.
- We can analyze the performance of the setups from a theoretical viewpoint if we model the signals as stochastic processes.

Linear filtering using mean-squared error

Linear filtering



$u_n = s_n + \eta_n = \text{signal} + \text{noise}$

- Filtering $d_n = s_n$
- Smoothing $d_n = s_{n-D}$
- Prediction $d_n = s_{n+D}$ Forecasting

The wiener filter

The filter which minimizes the mean-squared error is the Wiener filter

The normal equations

We want to obtain an estimate of $\boldsymbol{\theta}$ to the linear model (remember, $\mathbf{x}_n$ is random)

$$y(n) := \hat{\mathrm{d}}_n = \boldsymbol{\theta}^T \mathbf{x}_n$$

that minimize the mean squared error $J(\boldsymbol{\theta}) = \mathbb{E}\left[\left(\mathrm{d}_n - \hat{\mathrm{d}}_n\right)^2\right]$, i.e

$$\boldsymbol{\theta}_* := \arg\min_{\theta} J(\boldsymbol{\theta})$$

Note, $\boldsymbol{\theta}_*$ is used and not $\hat{\boldsymbol{\theta}}$ (as last week), because when we know the statistical properties of the signals, we can find the optimal solution, and not just an estimate based on one specific dataset.

Minimizing the cost function

ex 2.1.3

To avoid notation clutter, we drop the n subscript, and obtain:

$$
\begin{aligned}
\hat{\mathrm{d}} &= \boldsymbol{\theta}^T \mathbf{x} \\
J(\boldsymbol{\theta}) &= \mathbb{E}\Big[(\mathrm{d} - \hat{\mathrm{d}})^2\Big] \\
\nabla J(\boldsymbol{\theta}) &= \nabla \mathbb{E}\Big[(\mathrm{d} - \boldsymbol{\theta}^T\mathbf{x})(\mathrm{d} - \boldsymbol{\theta}^T\mathbf{x})\Big] \quad \text{complete the square} \\
&= \nabla \mathbb{E}\Big[\mathrm{d}^2 - \mathrm{d}\boldsymbol{\theta}^T\mathbf{x} - \boldsymbol{\theta}^T\mathbf{x}\mathrm{d} - \boldsymbol{\theta}^T\mathbf{x}\boldsymbol{\theta}^T\mathbf{x}\Big] \quad \mathbf{x}^T\boldsymbol{\theta} \\
&= \nabla \mathbb{E}\Big[\mathrm{d}^2 - 2\boldsymbol{\theta}^T\mathbf{x}\mathrm{d} + \boldsymbol{\theta}^T\mathbf{x}\mathbf{x}^T\boldsymbol{\theta}\Big] \\
&= \nabla_{\boldsymbol{\theta}}\Big(\mathbb{E}\big[\mathrm{d}^2\big] - 2\boldsymbol{\theta}^T\mathbb{E}[\mathbf{x}\mathrm{d}] + \boldsymbol{\theta}^T\mathbb{E}\Big[\mathbf{x}\mathbf{x}^T\Big]\boldsymbol{\theta}\Big) \quad \text{Apply E[.]} \\
&= -2\mathbb{E}[\mathbf{x}\mathrm{d}] + 2\mathbb{E}\Big[\mathbf{x}\mathbf{x}^T\Big]\boldsymbol{\theta} = \mathbf{0} \quad \checkmark \text{ apply rules from 2.1.3} \\
\Rightarrow \mathbb{E}\Big[\mathbf{x}\mathbf{x}^T\Big]\boldsymbol{\theta} &= \mathbb{E}[\mathbf{x}\mathrm{d}] \quad \text{Solution ??}
\end{aligned}
$$

M2.4

$\theta = E[xx^T]^{-1} \cdot E[xd]$!!

We recognize these terms as auto-correlation and cross-correlation functions.

Vis et eksempel

Minimizing the cost function

ML 4.5

Normal Equations

$$\Sigma_x \boldsymbol{\theta}_* = \boldsymbol{r}_{dx} \quad (\boldsymbol{r}_{dx} \text{ is denoted } \boldsymbol{p} \text{ in the book})$$

$$\boldsymbol{p} = [r_{dx}(0)\ r_{dx}(1)\ \cdots\ r_{dx}(l-1)]^T$$ cross-corr

$$\Sigma_x = \begin{bmatrix} r_x(0) & r_x(1) & \cdots & r_x(l-1) \\ r_x(1) & r_x(0) & \cdots & r_x(l-2) \\ \vdots & \vdots & \ddots & \vdots \\ r_x(l-1) & r_x(l-2) & \cdots & r_x(0) \end{bmatrix}$$ auto-corr

To obtain the solution, $\boldsymbol{\theta}_*$ we get

$(X^T X)^{-1} X y$

$$\boldsymbol{\theta}_* = \Sigma_x^{-1} \boldsymbol{r}_{dx}$$

Relate this result to our Wiener filter: x is our input signal, d is our target signal, i.e our desired signal. So to apply a Wiener filter, we need to quantities: the auto-correlation of our input signal, and the cross correlation between our input signal and desired signal.

Theory needed

What can we do now?

- We know how to optimally learn the filter coefficients if we know the statistical properties of the signals.

Theory needed - how do get to know the statistical properties?

- Stochastic processes, since the signals we just described are random, we can use tools from probability theory.

Summary

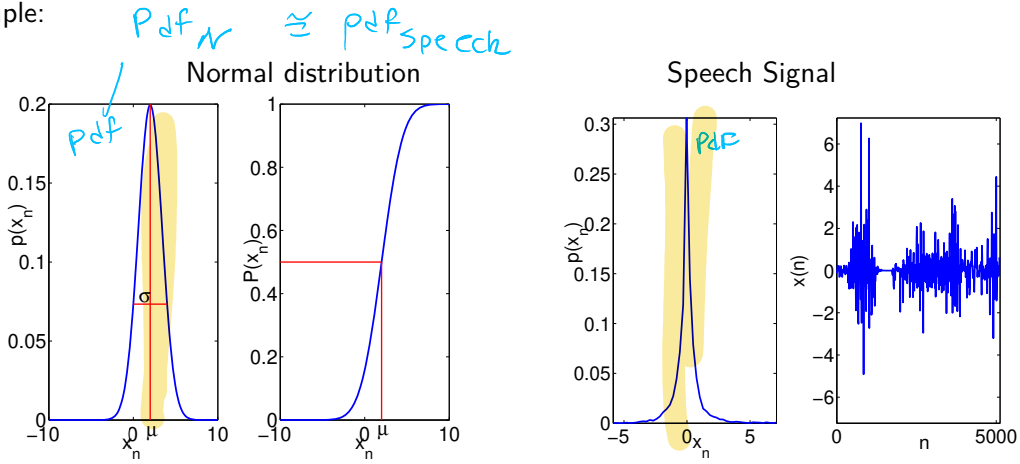
- We minimized the mean squared error for a linear filter, to obtain the Wiener filter solution.
- The solution is $\boldsymbol{\theta}_* = \Sigma_x^{-1} \boldsymbol{r}_{dx}$.
- We need to carry out the following actions:
  - Specify the desired signal.
  - Compute the autocorrelation function for the input signal.
  - Compute the cross-correlation function between the input signal and desired signal.

Stochastic Processes

Motivation

- Many real-world signals can be adequately be described as a stochastic process.

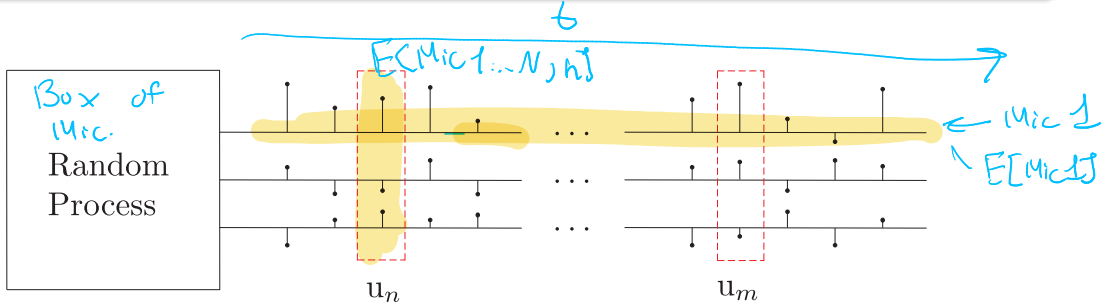
Example:



Stochastic Processes

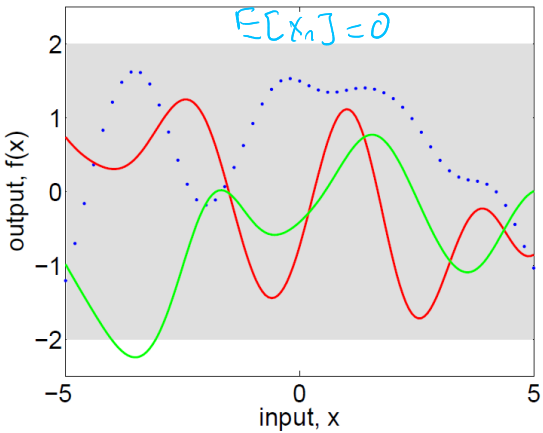
Stochastic process

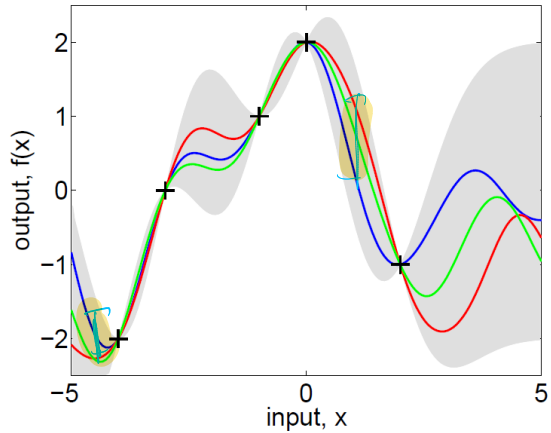
A stochastic process (or random process) can be considered as an ensamble of sequences, and individual examples are known as realizations.



u_n is the random variable at time instant n, and u_m is the random variable at time instant m. If the underlying probability density function does not change over time, the process is stationary.

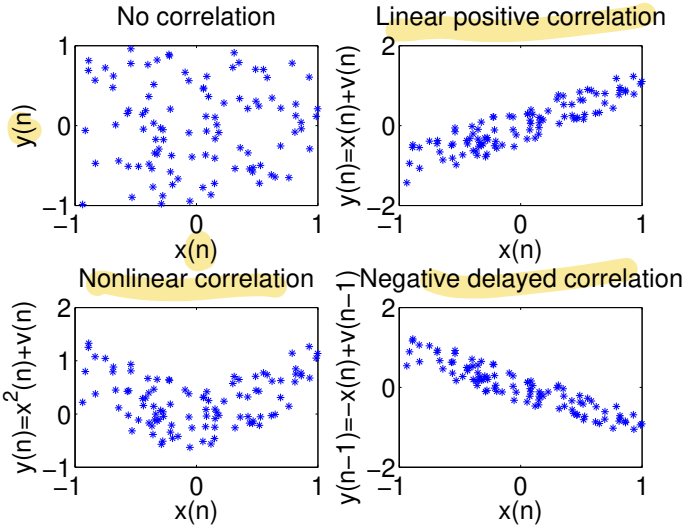
Example of a stochastic process





Ref: "Gaussian Processes for Machine Learning", Carl Edward Rasmussen and Christopher K. I. Williams MIT Press, 2006.

Concept of correlation



Important statistics

ML 2.4

Mean at Time n

$$\mu_n := \mathbb{E}[\mathrm{u}_n] = \int_{-\infty}^{\infty} u_n p(u_n) du_n$$

Autocovariance at time instants n, m

$$\mathrm{cov}(n,m) := \mathbb{E}[(\mathrm{u}_n - \mathbb{E}[\mathrm{u}_n])\,(\mathrm{u}_m - \mathbb{E}[\mathrm{u}_m])]$$

Autocorrelation at time instants n, m

$$r_u(n,m) := \mathbb{E}[\mathrm{u}_n \mathrm{u}_m]$$

Cross-correlation at time instants n, m

$$r_{uv}(n,m) := \mathbb{E}[\mathrm{u}_n \mathrm{v}_m]$$

These quantities are difficult to obtain in practice, so we impose certain assumptions about our signal of interest.

Weaker requirements – wide-sense stationarity

Wide-Sense Stationarity (WSS)

A stochastic process is said to be wide-sense stationary if the mean value is constant over time, and the autocovariance / autocorrelation sequences only depend on the differences of involved time instances

That means we can simply the following

For WSS processes

$$\mu_n = \mu$$

$$r_u(n, n-k) = r_u(k) = \mathbb{E}[\mathrm{u}_n \mathrm{u}_{n-k}]$$

def.

$$r_{uv}(n, n-k) = r_{uv}(k) = \mathbb{E}[\mathrm{u}_n \mathrm{v}_{n-k}]$$

Computing the statistics in practice

Definitions of estimators

$$\mu_u = \frac{1}{N}\sum_{n=1}^{N} u_n$$

$$r_{uv}(k) = \frac{1}{N}\sum_{n=k}^{N-1} u_n v_{n-k},\ k = 0, 1, \cdots, N-1 \quad (asymptotically\ unbiased)$$

$$r'_{uv}(k) = \frac{1}{N-k}\sum_{n=k}^{N-1} u_n v_{n-k},\ k = 0, 1, \cdots, N-1 \quad (unbiased)$$

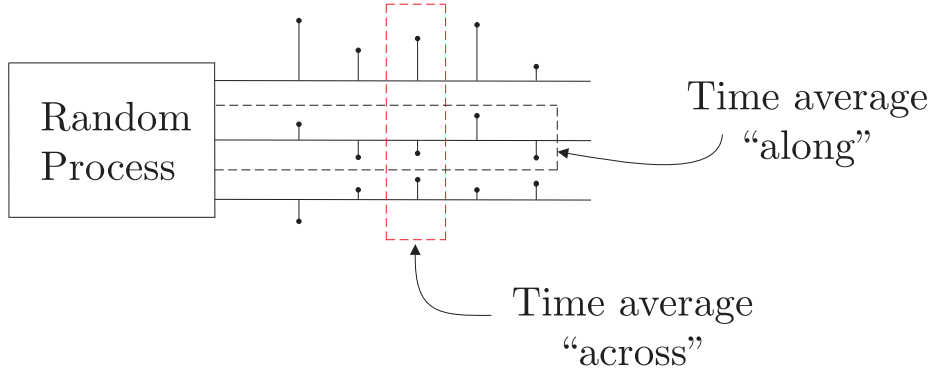
Note: cross-correlation functions can both be defined as normalized and un-normalized.
E.g. in time-series analysis, often the normalized cross-correlation is used.
See more:
`https://en.wikipedia.org/wiki/Cross-correlation#Normalization`

Weaker requirements – ergodic processes

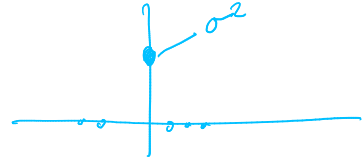
Mean-ergodic and covariance-ergodic

A stochastic process is said to be mean-ergodic and covariance-ergodic if the mean and covariance can be determined by any one of the realizations. Ergodic processes are also WSS.

Ensemble averages "across the process" can be obtained as time averages "along the process"



Example WSS process



White noise sequence

The white-noise sequence is both mean ergodic and covariance ergodic, and defined as

$$\mathbb{E}[\eta] = 0 \quad \text{and} \quad r(k) = \begin{cases} \sigma_\eta^2, & \text{if } k = 0, \\ 0, & \text{if } k \neq 0. \end{cases}$$

vis det visuelt 2025

Example WSS process, that is not covariance-ergodic

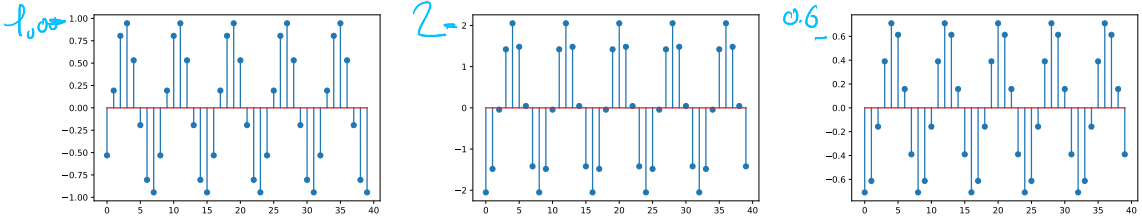
Consider the process

$$x_n = a\cos(n\lambda) + b\sin(n\lambda)$$

$$a \sim \mathcal{N}(0, \sigma_a^2),\ b \sim \mathcal{N}(0, \sigma_b^2),\ \lambda \in [0, \pi]$$

This process is WSS ($r_x(k) = \cos(\lambda n)$), mean-ergodic, but not covariance-ergodic.

Example realizations



Autoregressive models

An autoregressive process of order l, denoted as AR(l), follows the following

Autoregressive process (asymptotically WSS)

def 1

$$u_n + a_1 u_{n-1} + \cdots + a_l u_{n-l} = \eta_n$$

noise

Time Series Analysis

or identically written as (neglecting the sign of a's)

def 2

Current Time

$$u_n = \sum_{k=1}^{l} a_k u_{n-k} + \eta_n$$

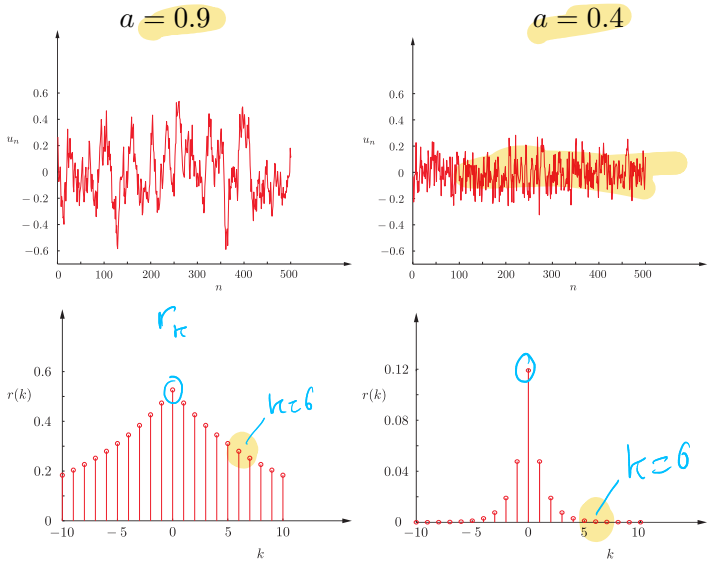
where η_n is a white noise process with variance σ_η^2, i.e. the samples at η_n and η_m are uncorrelated for any $n \neq m$, and $\mathbb{E}[\eta] = 0$.

This is a very powerful model, that can model many real-life signals well, as the model simply specify that the value at instant n, is a linear combination of the l previous samples, added with (measurement) noise.

We will work heavily with an AR(1) process in the exercises

ex 3 PS 2

Example of AR(1) process and the autocorrelation function



Stochastic processes summary

- We work with wide-sense stationary processes, which means their mean is constant over time, and their correlation functions can be computed with only the time lag taken into account.
- For real signals, we assume they are mean-ergodic and covariance-ergodic (a locally sensible assumption), so we can compute statistics from only one realization.
- As signal model of particular interest, we use an AR(l) process, which models many real-world phenomena adequately.
- Note: for a rigorous treatment of stochastic processes, DTU offers the course, "02407 Stochastic Processes - Probability 2".

Lecture summary

- Instead of designing filters by hand, we can train the filters from data. If we use a LTI filter and the Mean Squared Error as our cost function, the filter is called Wiener filter (or linear filtering).
- We apply theory from stochastic processes to analyze this setup and benchmark the systems.
- The stochastic processes we work with, will be wide-sense stationary and ergodic. That means we can estimate the mean and correlation functions using only "one" realization (example). This makes the system applicable to real data.
- Many of the applications require adaptive filtering to be useful in practice. This will be the topic for the next two weeks.

W 4+5

Next week

Exam 2022, 2.2 2.3

Material: ML 2.6, 5.1–5.5.1 (skip 5.3.1), 5.9.

- Adaptive filtering
- Stochastic Gradient descent
- Least-Mean-Squares (LMS) adaptive algorithm, and normalized LMS (NLMS)