Statistical, stochastic, probabilistic.

ESSP class notes

Quiz : 25%
Assignments : 30%
Final Exam : 40%
Class Participation : 5%

relative subjective
A, > 85 F < 40.

Read Hayes Chap 2

Statistical Digital Signal Processing and Modeling by Monson H Hayes.

↳ Chapter 3 for the first few classes

Thursday 4:30 Tut // Office hours
↳ Including tuts. etc.

Signals are not deterministic → WE NEED TO CAPTURE RANDOMNESS

model as a random process.

$x(t)$ → [System] → $y(t)$

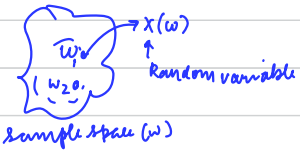
this signal unlike before is not deterministic

RANDOM

A given instance of a random process is a signal.

But collective instances makes it a random process.

Random Variable: A function that maps a variable to some real numbers in a sample space



Discrete	Continuous
* Bernaulli	* Uniform
* Poisson	* Gaussian
* Binomial	* exponential
* Geometric	

Central limit Thm: Mean of a series of numbers will be like a gaussian (as no of nos → ∞)

Noise represented by gaussian due to CLT

How to describe a RV? CDF → cumulative distributive fn.

PDF/PMF:
↳ may or may not exist [not all CDFs are differentiable]

$f_X(x)$ → PDF

$F_X(x)$ → CDF

mean, expectation, moments -->

CDF and PDF are deterministic functions that represent a random variable.

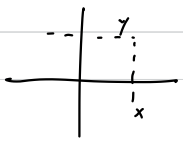
Random Vectors

Random variable are single dimensioned.

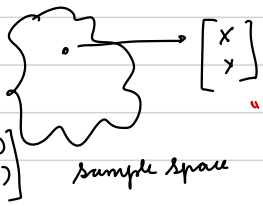
$\underline{Z} = \begin{bmatrix} X \\ Y \end{bmatrix}$ → Random vector

this pair sum as a vector.

Joint CDF → $F_{X,Y}(x,y) = \text{prob}(X \le x \ \& \ Y \le y)$



Random vectors is mapping from a sample space to the vector.



"I pick a random person, & note down height (x), weight (y)"

CDF

POF / PMF (If both are discrete)

mean $\underline{\mu} = \begin{bmatrix} E(X) \\ E(Y) \end{bmatrix}$

covariance [more than 1] $\Sigma = \begin{bmatrix} \text{var}(X) & \rho \\ \rho & \text{var}(Y) \end{bmatrix}$ $\rho = E[(X - E(X))(Y - E(Y))]$

how X and Y vary together.

$\text{var}(X) = E[(x - E(x))^2] = E(X^2) - E^2(X)$

$\underline{\mu} = E(\underline{z})_{2\times 1}$ $\quad \underline{z} \to [2 \times 1]$

$\Sigma = E\left[(\underline{z} - E(\underline{z}))(\underline{z} - E(\underline{z}))^T\right]_{2\times 2}$ symmetric

replace 2 with 'n' for generalization.

Random process / signal

map your sample space to a signal

$X(\omega) \to$ random process

NOW

$X(\omega, t) \to$ signal is a function of time

$X_1(\omega_1, t) \to$ one sample

$X_2(\omega_2, t) \to$ different sample.

Everytime a signal is different.

Randomness and structure

$X_1(\omega_1, t)$ → person A saying "one" [ω_1 is changing amplitude values t w/ time]

$X_2(\omega_2, t)$ → person B saying "one"

X_1 and X_2 has randomness + some structure

In this course we take t as discrete [digital]

$X_1(\omega_1, n)$, $X_2(\omega_2, n)$

Random process → collection of random variable

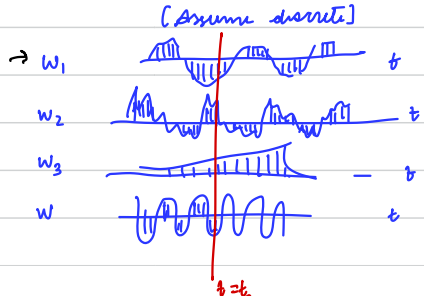
$X(\omega, t)$ has two interpretations

① random experiment → (ω fix) → $X(\omega, n)$

② At each (time fix) instant we have a random variable.

Collection for all ω → random proc $X(\omega=\omega_i, t)$

[Assume discrete]



① You pick ω, and you get a signal wrt time.

② You fix time and all the diff ω → can be modelled as a random variable.

[doesn't have to be variation with time] → [space.]

$X(\omega, t=t_0)$ collection for all t_i random process.

⇒ mean = $\mu(t) = E[X(\omega, t)]$

mean signal, function of time

⇒ mean $\mu(n) = E[X(n)]$

⇒ Autocorrelation

$R_X(n_1, n_2) = E[X(n_1)X(n_2)]$

↳ signal at two different points behavior ↴

⇒ auto covariance $C_X(n_1, n_2) = E[[X(n_1) - \mu(n_1)][X(n_2) - \mu(n_2)]$

↑ partial descriptors

We should be able describe: (Joint PDF)

$F_{X(n_1)\cdots X(n_N)}(x_1 \cdots x_n)$

fully describe the random process

↳ specify joint pdf for any $N = 1, 2 \rightarrow$

every set $\{n_1, n_2 \cdots n_N\}$

and every value $\{x_1, x_2 \cdots x_N\}$

Every possible subset

Random Processes

Stationary

nature of randomness is not changing with time

↑ statistics is not changing

non-stationary

nature of randomness is changing with time.

↑ State are changing

Tend to be stationary in short duration, non stationary in long duration

stock market. You want your prices to go up with time eventually.

Optimal linear filters → stationary for short duration

Adaptive filters → non stationary

Nonlinear

Complex random variable

$j = \sqrt{-1}$

$z = x + jy \equiv \begin{bmatrix} x \\ y \end{bmatrix}$

complex random process $Z(\omega, n)$

$E(z) = E(x) + jE(y)$

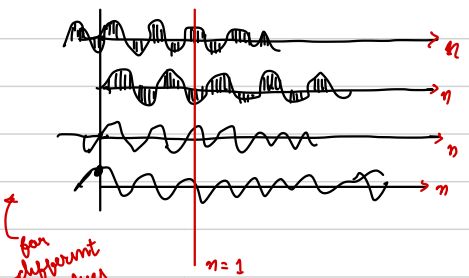
$\text{var}(z) = E[|z - E(z)|^2]$

$= \text{var}(X) + \text{var}(Y)$

Example

$X(n) = A\sin(\omega n + \phi)$ (A, ω fixed) $\quad \phi \sim U[-\pi, \pi]$

↳ phase is randomly uniform from $-\pi$ to π



for different ϕ values

$X(n=1) = A\sin(\omega + \phi) = F(\phi)$ ↑ function of ϕ

$\mu(n) = E(X(n)) \overset{?}{=} 0$ for $\forall n$.

Prove

$E(X(n)) = \int_{-\pi}^{\pi} g(\alpha) f_\phi(\alpha)$ ($f_\phi = 1/2\pi$)

for very n, $X(n)$ is a random variable

$\frac{1}{2\pi}\int_{-\pi}^{\pi} A\sin(\omega n + \alpha)\, d\alpha$

$\frac{[-A\cos(\omega n + \alpha)]_{-\pi}^{\pi}}{2\pi}$

$= 0$

Autocorrelation:

$X(n_1)X(n_2) = A^2 \sin(\omega n_1 + \phi)\sin(\omega n_2 + \phi)$

$= g_1(\phi)$

$A + B = (\omega n_1 + \phi)2$

$A - B = (\omega n_2 + \phi)2$

$A = \omega n_1 + \omega n_2 + 2\phi$

$B = \omega n_1 - \omega n_2$

$-\frac{1}{2}A^2\left[\cos(\omega n_1 + \omega n_2 + 2\phi) - \cos(\omega n_1 - \omega n_2)\right]$

Mean

$\frac{A^2}{2\pi}\int_{-\pi}^{\pi} \sin(\omega n_1 + \phi)\sin(\omega n_2 + \phi)\, d\phi$

$\frac{A^2}{4\pi}\int_{-\pi}^{\pi} \cos(\omega n_1 - \omega n_2)\, d\phi - \cos(\omega n_1 + \omega n_2 + 2\phi)\, d\phi$

$\frac{A^2}{4\pi}\left[\cos(\omega n_1 - \omega n_2)\cdot 2\pi - \frac{1}{2}\left[\sin(\omega n_1 + \omega n_2 + 2\phi)\right]_{-\pi}^{\pi}\right]$ (the sine term $\to 0$)

$\boxed{\frac{A^2}{2}\cos(\omega n_1 - \omega n_2)} = \frac{A^2}{2}\cos(\omega(n_2 - n_1))$

↑ Autocorrelation. ↑ function of the time difference.

Wide sense stationary random processes.

Practice problems

① $X(n) = A e^{j(\omega n + \phi)}$ $\quad \phi \sim U(-\pi, \pi)$

② $X(n) = a$ $\quad a \sim U(-1, 1)$

③ $X(n) = e^{-an}$ $\quad a \sim U(-1, 1)$

Find the mean function and the autocorrelation.

check if they are stationary or not.

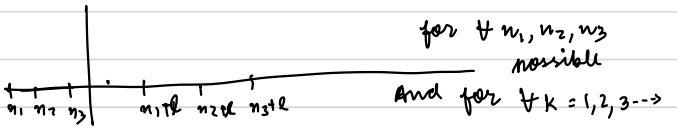
Stationary → SSS [Strict sense stationary]

Stationary → WSS [wide sense stationary]

SSS → All joint pdfs dont change w/ time

WSS → the first and second order moment [mean & covariance] doesnt change with time.

SSS { Joint pdf } $f_{X(n_1), X(n_2) \cdots X(n_k)}(\alpha_1 \cdots \alpha_k) = f_{X(n_1+\ell), X(n_2+\ell) \cdots X(n_k+\ell)}(\alpha_1 \cdots \alpha_k)$



for $\forall n_1, n_2, n_3$ possible

and for $\forall k = 1, 2, 3 \cdots$

WSS { $\mu(n)$ is constant for $\forall n$ [independent of n] and the auto correlation depends only on the time difference.

$r_{xx}(n_1, n_2) = r_{xx}(n_1+\ell, n_2+\ell) \quad \forall \ell$.

$= r_{xx}(n_1 - n_2, 0)$

$r_{xx}(k)$

where k represents the difference b/w n_1 and n_2

Auto correlation

$r_{xx}(n_1, n_2) = E[X(n_1) X(n_2)]$ [real]

$r_{xx}(n_1, n_2) = E[X(n_1) X^*(n_2)]$ [complex]

cross correlation

$r_{xy}(n_1, n_2) = E[X(n_1) y^*(n_2)]$

here the order matters.

Autocovariance

$$C_{xx}(n_1, n_2) = E\left[\left(X(n_1) - E(X(n_1))\right)\left(X(n_2) - E(X(n_2))\right)^*\right]$$

Cross covariance

$$C_{xy}(n_1, n_2) = E\left[\left(X(n_1) - E(X(n_1))\right)\left(Y(n_2) - E(Y(n_2))\right)^*\right]$$

Ex of usage: ↳ cross correlation b/w the input and the output.

$C_{xx}(n_1, n_2) = r_{xx}(n_1, n_2) - \mu_x(n_1)\mu_x^*(n_2)$

[can we prove ?]

Two random processes are said to be uncorrelated if

$C_{xy}(n_1, n_2) = 0 \quad \forall n_1, n_2$

[weaker condition than independence. Independence will imply this]

$r_{xy}(n_1, n_2) = \mu_x(n_1)\mu_y^*(n_2)$

iid random process: independent & identically distributed

$X(n_1)$ and $X(n_2)$ is IID rv $\forall n_1$ and n_2 $(n_1 \neq n_2)$

IID ⇒ SSS → identically distributed, same marginal distribution, so joint pdf is same through translation.

⇓ WSS.

Usually used to model noise etc, with extremely erratic values.

high fluctuation, no correlation from one step to another.

IID Gaussian is a common assumption for noise

For WSS the autocorrelation = $r_x(k)$. [for any other condition this is a 2D signal]

properties of $r_x(k)$: $= E[x(n+k)x^*(n)]$

① $r_x(k) = r_x^*(-k)$

② $r_x(0) > 0$

③ $r_x(0) \geq |r_x(k)|$ ↑ highest at zero.

Show

Jointly WSS

If X and Y are WSS and $r_{xy}(n_1, n_2) = r_{xy}(n_1+\ell, n_2+\ell)$

Matrix

WSS → { mean μ, auto correlation $r_x(k)$ }

consider the x $\quad x = [x[0], x[1], x[2], \cdots x[N]]^T$

outerproduct = xx^H (x^H hermitian)

$R_x = E[xx^H]$

Autocorrelation. $xx^H = \begin{bmatrix} x[0]x^*[0] & x[0]x^*[1] & \cdots \\ x[1]x^*[0] & x[1]x^*[1] & \cdots \\ x[2]x^*[0] & x[2]x^*[1] & \cdots \\ x[3]x^*[0] & x[3]x^*[1] & \cdots \end{bmatrix}$

all of these are random variables.

$r_x[k] = E[x[n+k]x^*[n]]$ → notation for function

$R_x = E[xx^H]$ → notation for matrix

$$R_x = \begin{bmatrix} r_x[0] & r_x[-1] & r_x[-2] & \cdots \\ r_x[1] & r_x[0] & \cdots & \\ r_x[2] & & r_x[0] & \\ \vdots & & & \\ r_x[N] & & & \end{bmatrix}$$

$r_x[k]$ not symmetric in complex, symmetric in real case.

we know that $r_x[k] = r_x^*[-k]$

$r_x[-1] = r_x^*[1]$

$$R_x = \begin{bmatrix} r_x[0] & r_x^*[1] & r_x^*[2] & \cdots \\ r_x[1] & r_x[0] & \cdots & \\ \vdots & & & \\ r_x[N] & r_x[N-1] & & r_x[0] \end{bmatrix}$$

Hermitian → transpose & complex conjugate

Properties

→ $R_x^H = R_x$ → Hermitian

→ All diagonals are constant : Toeplitz matrix

⇒ $a^H R_x a$ for any vector $a \in \mathbb{C}^{N+1}$ $a \neq 0$

$a^H R_x a \geq 0$ → Non negative definite

Proof: $a^H R_x a = a^H E[xx^H] a \geq 0$.

a^H is constant in contrast to randomness shown by x.

$$E[a^H x x^H a] = E[(a^H x)(a^H x)^*]$$

($a^H x$, $x^H a$: complex scalars)

$$= E[|a^H x|^2]$$

this value always greater than or equal to zero.

Ergodic Process

Lets say you have a RV, and you generate samples of RV. How do you estimate mean?

$X \sim RV$.

$E(X)$ → ensemble mean [for a large sample → this is the mean]

WSS

RP → $X(t)$

try to figure out if random process is stationary or not

mean $\mu(t)$?

↳ take a bunch of signals at time t and avg it out

↳ take a lot of time steps.

⇓ ENSEMBLE MEAN.

WSS → $\mu(t) = E[x(t)]$

But lets say you have only one realization of the random process say $x(t)$ [$x[n]$ → since we deal w/ discrete]

What if we take avg of this signal across time.

N is large

$$\frac{1}{2N+1} \sum_{n=-N}^{N} x[n]$$ [completely different from ensemble mean]

Eg $\lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} X[n]$ converges ll

Any realization

WSS

↑ temporal Average.

↑ IFF this happens: The RP is said to be Ergodic in mean.

If the temporal avg over all time converges to the Ensemble average → Ergodic in mean.

Example:

$X(t) = \cos(\omega t + \phi)$ → wide sense stationary

$\phi \sim U(-\pi, \pi)$

Average over all signals → 0.

Average over all time → 0

Ergodic in mean.

Example 2 -

$X(t) = A$ $A \sim U(-1, 1)$

$\mu(t) = 0$. [average of uniform - -).

But temporal Avg = A

so, $\frac{1}{2N+1} \sum_{n=-N}^{N} x[n]$ ~~converges~~ → 0.

Not ergodic in mean.

Ergodicity in auto correlation

IF the time average

$$\lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} X[n+k] x^*[n]$$ converges $r_x[k]$

$E[x[n+k] x^*[n]]$ ← multiple

We shall implicitly assume this in future

Fourier Transforms

$$S(e^{j\omega}) = \sum_{n=-\infty}^{\infty} s(n)\, e^{-j\omega n}$$

[DTFT]

$$s[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} S(e^{j\omega})\, e^{j\omega n}\, d\omega$$

keep in mind: Dirichlet conditions

A lot of random processes fourier transform may not exist.

(1) $X[n]$ WSS μ, $r_x[n]$

$X[n]$: random; $r_x[n]$: realizable

Power Spectral Density → Fourier transform $(r_x(n))$

$$S_x(e^{j\omega}) = \text{DTFT}(r_x[n])$$

$$S_x(e^{j\omega}) = \sum_{k=-\infty}^{\infty} r_x[k]\, e^{-j\omega k}$$

PSD

For deterministic signals:-

Energy → $\sum_n |x[n]|^2$ Power = $\frac{1}{2N+1} \sum_{-N}^{N} |x[n]|^2$

localized

For random signals:-

$$(x[n])^2 = x[n]\, x^*[n]$$

for random: $P_x = E\left\{ \frac{1}{2N+1} \sum_{n=-N}^{N} |x^2[n]| \right\} = r_x(0) = E[|x(0)|^2]$

PSD: $S_x(e^{j\omega}) = \sum_{k=-\infty}^{\infty} r_x[k]\, e^{-j\omega n}$

inversely

$$r_x[k] = \frac{1}{2\pi} \int_{-\pi}^{\pi} S(e^{j\omega})\, e^{j\omega k}\, d\omega$$

$$r_x[0] = \frac{1}{2\pi} \int_{-\pi}^{\pi} S(e^{j\omega})\, d\omega$$

PSD measures the power at diff frequencies of random processes.

(2) $X[n] \xrightarrow{FT} X(e^{j\omega})$

On avg what is the power in fourier transform?

compute $P(e^{j\omega}) = |X(e^{j\omega n})|^2$ $\quad X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x(i)\, e^{-j\omega n}\, d\omega$

NOT complete

$$P_N(e^{j\omega}) = \frac{1}{2N+1} \left| \sum_{-N}^{N} x[n]\, e^{-j\omega n} \right|^2$$

$$E\left[P_N(e^{j\omega})\right] = E\left[\frac{1}{2N+1} \left| \sum_{n=-N}^{N} x[n]\, e^{-j\omega n} \right|^2 \right]$$

take $\lim_{N\to\infty}$ & expand

Learn for random signals

$$S_x(e^{j\omega}) = \sum_{-\infty}^{\infty} r_x[k]\, e^{-j\omega n}$$

$$\lim_{N\to\infty} E\left[P_N(e^{j\omega})\right]$$

If $x(t)$ is real $S_x(e^{j\omega})$ is even.

White Noise

$r_x[k] = \sigma^2 \delta[k]$ $\qquad \delta = \begin{cases} 1 & k=0 \\ 0 & k=1 \end{cases}$

random variable is only correlated to itself.

Power spectral density

$$S_x(e^{j\omega}) = \frac{1}{2N+1} \sum_{n=-N}^{N} r_x[k]\, d\omega$$

$$S_x(e^{j\omega}) = \sigma^2$$

all frequency in white noise has same power spectral density for

White gaussian → special case of white noise

white noise is defined for any distribution.

$X[n] \rightarrow [h[n]] \rightarrow Y[n]$

$X[n]$: random process, ensemble of random signals

$h[n]$: LTI, impulse response

$Y[n]$: random process, ensemble of random signal

$$Y[n] = X[n] * h[n]$$

$$Y[n] = X[n] * \text{(impulse train)}$$

$$y[n] = \sum_{k=-\infty}^{\infty} h[k]\, x[n-k]$$

Let $x[n]$ be wide sense stationary. find mean and autocorrelation fn of $Y[n]$ [compute]

$$E[Y[n]] = E\left[\sum_{k=-\infty}^{\infty} h[k]\, x[n-k] \right]$$

$$E[Y[n]] = \left[\sum_{k=-\infty}^{\infty} h[k] \right] E(x[n-k])$$

↑ constant mean μ

$$E[y[n]] = \mu \sum_{k=-\infty}^{\infty} h[k]$$

↑ as long as this is finite, output of LTI also has a constant mean

Math 15-M

Autocorrelation of $y[n]$:

$E(y(n)\, y(n+k))$

Before that find cross correlation of input and output

$r_{yx}[n+k, n]$

$y(n) = \sum_{l=-\infty}^{\infty} x[l]\, h[n-l]$

By definition

$$r_{yx}[n+k, n] = E\left(y[n+k]\, x^*[n]\right)$$

$$\sum_{l=-\infty}^{\infty} E\left(x(l)\, h(n+k-l)\, x^*[n]\right)$$

$$\sum_{l=-\infty}^{\infty} E\left[x(l)\, x^*[n]\right) E\left(h(n+k-l)\right)$$

(h deterministic)

$$\sum_{l=-\infty}^{\infty} r_x(l-n)\, h(n+k-l)$$

$$\sum_{l=-\infty}^{\infty} r_x(l-n)\, h(k-(l-n))$$

$l - n = m$

$$\sum_{m+n=-\infty}^{\infty} r_x(m)\, h(k-m)$$

$$r_{yx}[n+k, n] = \sum_{m=-\infty}^{\infty} r_x(m)\, h(k-m) = r_x[k] * h[k]$$

$l - m = n$

Autocorrelation:

$$r_y[n+k, n] = E\left[y[n+k]\, y^*[n]\right]$$

$$E\left[\left[y[n+k] \cdot \sum_{l=-\infty}^{\infty} x^*[l]\, h^*[n-l]\right]\right]$$

$$\sum_{l=-\infty}^{\infty} h^*[n-l]\; E\left[y[n+k]\, x^*[l]\right]$$

$E[y[n+k]\, x^*[l]] \leftarrow r_{yx}[n+k, l]$

$$\sum_{l=-\infty}^{\infty} h^*[n-l]\left(r_x[n+k-l] * h[n+k-l]\right)$$

$n - l \to m$

$$r_y[n+k, n] = \sum_{m=-\infty}^{\infty} h^*[m]\left(r_x[m+k] * h[m+k]\right)$$

$g(k+m)$ ⟶

$$r_y[n+k, n] = \sum_{m=-\infty}^{\infty} h^*[m]\, g(k+m)$$

- correlation -

$$r_y[n+k, n] = g[k] * h^*[-k]$$

$$r_y[n+k, n] = r_x[k] * h[k] * h^*[-k]$$

$$\begin{cases} f(k) * g(k) = \sum_n f(n) g(k-n) \\ f(k) * g(-k) = \sum_n f(n) g(k+n) \end{cases}$$

If $r_x[k] * h_x[k]$ is finite; that our autocorrelation depends only on time difference.

So, then LTI of WSS is WSS.

Conditions for stationary

↳ If $\sum |h[k]| < \infty$

↳ If LTI system is stable then output is WSS.

Taking it into frequency domain

[Power spectral density] ⟶ DTFT

$$S_y(\omega) = S_x(\omega) \cdot H(\omega) \cdot H^*(\omega)$$

$$S_y(\omega) = S_x(\omega)\, |H(\omega)|^2$$

for a stable LTI system.

$$\begin{cases} H(\omega) = \sum_{n=-\infty}^{\infty} h[n]\, e^{-j\omega n} \\ H^*(\omega) \sum_{n=-\infty}^{\infty} h^*[n]\, e^{j\omega n} \\ H^*(\omega) = \sum_{k=-\infty}^{\infty} h^*[-k]\, e^{-j\omega k} \end{cases} \quad n = -k$$

Generating a random variable ?

$x[n]$ is a white process $\Rightarrow$ $r_x[k] = \sigma_x^2 \delta[k]$

PSD $= \sigma_x^2$

If input of an LTI system is a white process:-

$$S_y(\omega) = \sigma_x^2\, |H(\omega)|^2$$

↗ has a signature of the system's fourier transform

For generating:

given $r[k] \longleftrightarrow S(\omega)$

→ Give white process as input

→ Choose an $H(\omega)$ which would give us a specified autocorrelation / power spectral density

[choose a system]

→ Not necessarily practical or easy

$h(n) \longleftrightarrow H(z) \to \frac{Y(z)}{X(z)}$ [rational] ↗ FIR ↘ IIR

$H(\omega)$

white process

$v[n] \longrightarrow$ [LTI] $\longrightarrow y(n)$

→ **MA $h[n]$ is FIR and causal**

Moving Average (MA) RP:-

$y[n] = \sum_{k=0}^{q} h[k]\, v[n-k]$ $\quad v[n] = \sum_{i=0}^{2} b_i\, v[n-i]$

$y[n] = h[0]v[n] + h[1]v[n-1] \cdots h[q]v[n-q]$

window of length $q+1$. Averaging the past $q+1$ inputs. As n changes the location of avg changes.

Using specific notation:-

$y[n] = b_0 v[n] + b_1 v[n-1] + \cdots b_q v[n-q]$

Taking z transform on both sides.

$Y(z) = b_0 V(z) + b_1 V(z) z^{-1} \cdots b_q V(z) z^{-q}$

$Y(z) = V(z)\left[b_0 + b_1 z^{-1} \cdots b_q z^{-q}\right]$

$\frac{Y(z)}{V(z)} = b_0 + b_1 z^{-1} \cdots b_q z^{-q}$

$\frac{Y(z)}{V(z)} = \sum_{i=0}^{q} b_i z^{-i} = B_q(z)$.

$PSD = \sigma_w^2 \times |H(w)|^2 \quad z = e^{jw}$

$PSD = \sigma_w^2 \left|B_q(z = e^{jw})\right|^2$

⇒ **Autoregressive (AR) RP ⇔ IIR filters**

$y[n] + a_1 y[n-1] + a_2 y[n-2] \cdots a_p y[n-p] = v[n]$

output at previous times is recycled again in the system.

$Y(z)\left[1 + a_1 z^{-1} + a_2 z^{-2} \cdots a_p z^{-p}\right] = V(z)$

$\frac{Y(z)}{V(z)} = \frac{1}{1 + a_1 z^{-1} + a_2 z^{-2} \cdots a_p z^{-p}}$

$\frac{Y(z)}{V(z)} = \frac{1}{A_p(z)}$

→ compression of audio signals.
→ modeling of practical signals (like speech processing) [Linear Predictive coding]
↳ predictive process

$S_y(w) = |H(w)|^2 \sigma_w^2 = \frac{\sigma_w^2}{A_p^2(e^{jw})}$

③ ARMA

$y[n] + \sum_{i=1}^{p} a_i y[n-i] = \sum_{i=0}^{q} b_i v[n-i]$

← AR → ← MA →

$\frac{Y(z)}{V(z)} = H(z) = \frac{B_q(z)}{A_p(z)}$

$\Rightarrow S_y(w) = \left|\frac{B_q(z = e^{jw})}{A_p(z = e^{jw})}\right|^2 \sigma_w^2$

→ Show that MA random process is always wss whereas AR may or may not be wss.

↳ FIR filter is always stable
IIR filter is not necessarily stable.

You can also show the mean & Ac being constant & time invariant resp for FIR but not necessary for IIR.

AR processes

Assume wss.

Q) given autocorrelation $r[k]$ find coefficients

$x[n] + a_1 x[n-1] + a_2 x[n-2] \cdots a_p x[n-p] = v[n]$

$x[n] + a_1 x[n-1] + a_2 x[n-2] + \cdots a_p x[n-p] = v[n]$

[multiply with $x^*[n-l]$] and take expectation on both sides.

$E\left[x[n]x^*[n-l] + a_1 x[n-1]x^*[n-l] \cdots a_p x[n-p]x^*[n-l]\right] = E[v[n]x^*[n-l]]$

$r_x[l] + a_1 r_x[l-1] + a_2 r_x[l-2] \cdots a_p r_x[l-p] = E[v[n]x^*[n-l]]$

$E[v[n]x^*[n-l]] = \sum_{i=0}^{i=p} r_x[l-i]$

AR is a causal system. $x[n]$ depends on $v[n]$ and $x[n-1]$ depends on previous $v[n-1]$ and so on.

$$E[v[n]\, x^*[n-l]] = \begin{cases} \sigma_v^2 & l = 0 \\ 0 & l = 1 \cdots p \end{cases}$$

$x^*[n-l]$ would not have seen $v[n]$ which is in future; this is a causal system.

$l=0$ } $x[n]$ is correlated to $v[n]$.
also $E[v[n]\, x^*[n]] = \sigma_v^2$

AR random process

$$x[n] + a_1 x[n-1] + a_2 x[n-2] = v[n]$$

Given $= \{ r_x[0], r_x[1], r_x[2] \}$

find $= \{ a_1, a_2, \sigma_v^2 \}$

multiply w/ $x^*[n]$

$$x[n]x^*[n] + a_1 x[n-1]\, x^*[n] + a_2 x[n-2]\, x^*[n] = v[n]\, x^*[n]$$

taking expectation on both sides:

$$r_x[0] + a_1 r_x[1] + a_2 r_x[2] = E[v[n]\, x^*[n]] = \sigma_v^2$$

$$r_x[0] + a_1 r_x[1] + a_2 r_x[2] = \sigma_v^2$$

$$\boxed{r_x[0] + a_1 r_x^*[1] + a_2 r_x^*[2] = \sigma_v^2}$$

Now multiply w/ $x^*[n-1]$ and take expectation

$$x[n]\, x^*[n-1] + a_1 x[n-1]\, x^*[n-1] + a_2 x[n-2]\, x^*[n-1] = v[n]\, x^*[n-1]$$

$$\boxed{r_x(1) + a_1 r_x(0) + a_2 r_x(-1) = 0}$$

Now multiply w/ $x^*[n-2]$ and take expectation

$$r_x[n]\, x^*[n-2] + a_1 x[n-1]\, x^*[n-2] + a_2 x[n-2]\, x^*[n-2] = 0$$

$$\boxed{r_x(2) + a_1 r_x(1) + a_2 r_x(0) = 0}$$

$v(n)$ is not correlated w/ any other term $n > n_0$.

$$\begin{bmatrix} r_x(0) & r_x^*(1) \\ r_x(1) & r_x(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} r_x[1] \\ r_x[2] \end{bmatrix}$$

$R_x \underline{a} = \underline{r}$ → Yule Walker Equation

$$r_x[0] + a_1 r_x^*[1] + a_2 r_x^*[2] = \sigma_v^2$$

$x[n+1] = y[n+1] - a_1 x[n] - a_2 x[n-1] - \cdots - a_p x[n-p+1]$

white process is uncorrelated w/ its past.

Random process are correlated w/ their past

Your prediction is the linear value based on the signal in its past.

$\hat{x}(n+1)$ → Don't need to transfer complete signal, only pass what else is needed for decoding

LPC

For an p^{th} ordered process

$$\begin{bmatrix} r_x[0] & r_x[1] & \cdots & r_x^*[p-1] \\ r_x[0] & r_x[0] & \cdots & r_x^*[p-2] \\ & \ddots & \ddots & \\ & & & \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} -r_x(1) \\ r_x(2) \\ \vdots \\ r_x(p) \end{bmatrix}$$

Ex: realisation of $x[n]$ that estimates $r_x[k]$?
if we assume its ergodic (in AC)
then we can estimate by random processes.

Estimate $r_x(k)$

$$\hat{r}_x[k] = \frac{1}{N-k+1} \sum_{n=0}^{N-k} x[n+k]\, x^*[n] = \frac{1}{N} \sum_{n=k}^{N} x[n]\, x^*[n+k]$$

$$r_x[k] = E[x[n+k]\, x^*[n]] = E[x[n]\, x^*[n+k]] = E[x[k]\, x^*[0]]$$

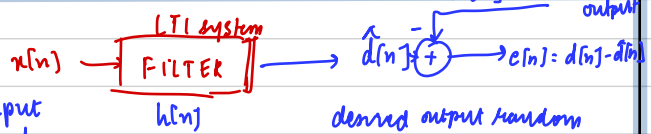
Optimum linear filters : MMSE filtering

LTI filter

Stochastic filters: process random process

minimum mean squared error

FIR causal filters



input random process

desired output random process

m(n) stochastic

Example: audio → music(n) stochastic; noise stochastic

noise RP removed.

x[n] and d[n] are WSS, RP are jointly WSS.

$\hat{d}[n]$ is also a WSS, e[n] is also a WSS.

to minimize the error signal : Minimize mean square error

find me a filter which minimizes error $E\left[|e[n]|^2\right] = J$ ← average power of signal

J doesn't depend on n since e[n] is WSS.

$h[k]$, $k = 0, 1, \dots, p$

We assume x and d are jointly WSS

$$J = E\left[e[n]\, e^*[n]\right]$$

$$\hat{d}[n] = \sum_{i=0}^{p} h[i]\, x[n-i]$$

$$e[n] = d[n] - \sum_{i=0}^{p} h[i]\, x[n-i]$$

$$J = E\left[\left(d[n] - \sum_{i=0}^{p} h[i]\, x[n-i]\right)\left(d^*[n] - \sum_{i=0}^{p} h^*[i]\, x^*[n-i]\right)\right]$$

$$J = E\left[d[n]\, d^*[n] - \sum_{i=0}^{p} h[i]\, x[n-i]\, d^*[n] - \sum_{i=0}^{p} \left(h^*[i]\, x^*[n-i]\right) d[n] + \sum_{i=0}^{p}\sum_{j=0}^{p} h[i]\, x[n-i]\, h^*[j]\, x^*[n-j] \right]$$

$r_x[k] = E\left[x[n]\, x^*[n-k]\right]$

$r_{dx}[k] = E\left[d[n]\, x^*[n-k]\right]$

$r_{dx}^*[k] = E\left[d^*[n]\, x[n-k]\right]$

$$J = r_d[0] - \sum_{i=0}^{p} h[i]\, r_{dx}^*[i] - \sum_{i=0}^{p} h^*[i]\, r_{dx}[i] + \sum_{i=0}^{p}\sum_{j=0}^{p} h[i]\, h^*[j]\, r_x[j-i]$$

double summation is a scalar

scalar → $h^H M h$

matrix → $h M h^H$

$\sum\sum$ ← outer product

$$h = \begin{bmatrix} h[0] \\ h[1] \\ \vdots \\ h[p] \end{bmatrix} \qquad r_{dx} = \begin{bmatrix} r_{dx}[0] \\ \vdots \\ r_{dx}[p] \end{bmatrix}$$

$h^{*T} = h^H$

$$J = r_d[0] - h^T r_{dx}^* - h^H r_{dx} + h^H R_x h$$

← maintaining dimensionality

$h^H R_x h$

$$\begin{bmatrix} h^*[0] & h^*[1] & \cdots & h^*[p] \end{bmatrix} \begin{bmatrix} r_x[0] & r_x^*[1] & \cdots & \\ r_x[1] & & & \\ & & & \end{bmatrix} \begin{bmatrix} h[0] \\ h[1] \\ \vdots \\ h[p] \end{bmatrix}$$

$$J = r_d[0] - h^T r_{dx}^* - h^H r_{dx} + h^H R_x h$$

Adding and subtracting $r_{dx}^H R_x^{-1} r_{dx}$

$$J = r_d[0] - h^T r_{dx}^* - h^H r_{dx} + h^H R_x h + r_{dx}^H R_x^{-1} r_{dx} - r_{dx}^H R_x^{-1} r_{dx}$$

$$J = \left(r_d[0] - r_{dx}^H R_x^{-1} r_{dx}\right) + \left(h - R_x^{-1} r_{dx}\right)^H R_x \left(h - R_x^{-1} r_{dx}\right)$$

← +ve → ↓ →

We need to find vector h such that J is minimized. The first terms doesn't depend on 'h' at all

$a^H R_x a \geq 0$

So for minimizing, zero.

So; $\boxed{h = R_x^{-1} r_{dx}}$ Wiener filter

Optimum MMSE filter.

discovered by Norbert wiener

So :

$$\boxed{J_{min} = r_d[0] - r_{dx}^H R_x^{-1} r_{dx}}$$

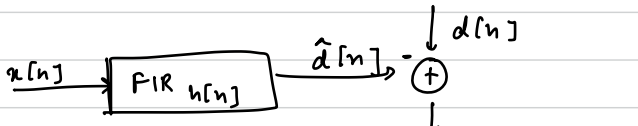
J never negative (computation of a square)

↳ Noise cancellation
↳ Noise clipping
↳ prediction of random
↳ deconvolution

WIENER FILTERS: ONLY FOR STATIONARY SIGNALS

March 22nd

FIR Wiener Filter



$J = E(|e^2[n]|)$ minimize

$$\hat{d}[n] = \sum_{i=0}^{p} h[i]\, x[n-i]$$

Solution we obtain:-

$$\underline{h} = R_x^{-1} \underline{r_{dx}}$$

wiener hopf equations

$$J = E(e[n]\, e^*[n])$$

to minimize J, $\frac{\partial J}{\partial h[k]} = 0 \ \forall k$

Complex derivatives (2.3.10).

$f(z) = |z|^2 = zz^*$ ← not differentiable wrt z.

$\frac{\partial f(z)}{\partial z}$ doesn't exist

the way to deal w/ this is to treat z, z^* as independent variables

$$g(z, z^*) = zz^*$$

$$\frac{\partial}{\partial z}\, g(z, z^*) = 0 \quad [\text{treats the } z^* \text{ as independent}]$$

for minimizing J, we treat $e(n)$, $e^*(n)$ as different variables

$$\frac{\partial J}{\partial h[k]} = E\left[\frac{\partial e[n]}{\partial h[k]}\, e^*[n]\right] = 0$$

$$\frac{\partial e[n]}{\partial h[k]} = \frac{\partial}{\partial h[k]}\left(d[n] - \hat{d}[n]\right)$$

$$= 0 - x[n-k]$$

so;

$$E[x[n-k]\, e^*[n]] = 0$$

orthogonality property

$$E\left[d^*[n] - \sum_{i=-\infty}^{\infty} h[i]\, x^*[n-i]\; x[n-k]\right]$$

$$r_{dx}^*[k] - \sum_{i=0}^{p} h[i]\, r_x[k-i] = 0$$

$\forall\ k = 0, 1, \cdots p.$

$$\sum_{i=0}^{p} h[i]\, r_x[k-i] = r_{dx}[k]$$

$\forall\ k = 0, 1 \cdots p.$

$$R_x \underline{h} = \underline{r_{dx}}$$

* Wiener – Hopf equation

$\underline{h}$: default column vector

All stationary signals-

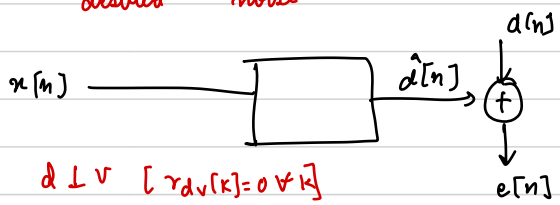
$\{r_x[0], r_x[1] \cdots r_x[p]\}$, $\{r_{dx}[0], r_{dx}[1] \cdots r_{dx}[p]\}$ $\xrightarrow[\text{eqn}]{W\text{-}H}$ $\{h[0] \ldots h[p]\}$

* Applications

1) Noise filtering

$x[n] = d[n] + v[n]$

desired, noise



$d \perp v$ $[r_{dv}[k] = 0 \ \forall k]$

$$r_{dx}[k] = E[d[n]\, x^*[n-k]]$$

$$= E[d[n](d^*[n-k] + v^*[n-k])]$$

$$r_{dx} = r_d[k] + 0$$

$$r_x[k] = E[x[n]\, x^*[n-k]]$$

$$= E[(d[n] + v[n])(d^*[n-k] + v^*[n-k])]$$

$$E[d[n]\, d^*[n-k]] + E[v[n]\, v^*[n-k]]$$

$$+ 0$$

$$r_x[k] = r_d[k] + r_v[k]$$

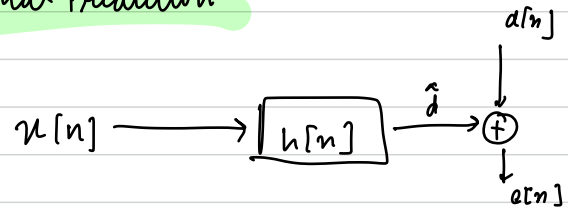
from WH eqn;

$$(R_d + R_v)\, \underline{h} = \underline{r_{dx}}$$

You can use Ergodicity to estimate desired signal statistics.

[how to obtain r_{dx} & r_x] not perf → adaptive filter

2) Signal Prediction



$d[n] = x[n+1]$

Why is prediction possible?

future sample has correlation to the older samples.

$$r_{dx} = E[d[n]\, x^*[n-k]]$$

$$E[x[n+1]\, x^*[n-k]] = r_x[k+1]$$

$$R_x \underline{h} = \underline{r_x}$$

same for yule walker AR process (in a way it's doing prediction)

$$x[n] + \sum_{i=1}^{p} a_i x[n-i] = v[n]$$

$$x[n] = v[n] - \sum_i a_i x[n-i]$$

$p=1$

$$\begin{bmatrix} r_x[0] & r_x^*[1] \\ r_x[1] & r_x[0] \end{bmatrix} \begin{bmatrix} h[0] \\ h[1] \end{bmatrix} = \begin{bmatrix} r_x[1] \\ r_x[2] \end{bmatrix}$$

solve for $h[0]$, $h[1]$

$$\hat{x}[n+1] = h[1]\, x[n-1] + h[0]\, x[n]$$

what if $x[n] = v[n]$; then $r_x[k] = \sigma_v^2 \delta[k]$

only zero for $r_x[0]$.

$r_x[1] = r_x[2] = 0$.

So here we will get $h[0]=0, h[1]=0$, proving that it can't be predicted —

Quality of prediction

$$J_{min} = r_d[0] - \underline{r_{dx}}^H R_x^{-1} \underline{r_{dx}}$$

3) Noise Cancellation

Noise is always changing (statistically)

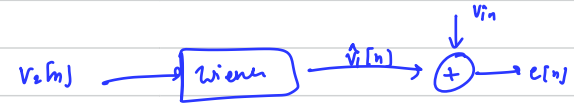
We assume noise is stationary.

Assume random process, deterministic signal

Primary $x[n] = s[n] + v_1[n]$ ← Ambient & not sensor noise

Secondary → $v_2[n]$ (ideal scenario)

Assume: v_1 & v_2 are correlated noises



Predict $v_1[n]$ with $v_2[n]$

Noise cancellation:

$$y[n] = x[n] - \hat{v}_1[n]$$

$$y[n] = s[n] + v_1[n] - \hat{v}_1[n]$$

$$R_{v_2} \underline{h} = r_{v_1 v_2}$$ ↑ not clear as

$$r_{v_1 v_2}[k] = E[v_1[n]\, v_2^*[n-k]]$$

$$E[(x[n] - s[n])\, v_2^*[n-k]]$$

$$E[x(n)\, v_2^*[n-k]] - s[n]\, E[v_2[n-k]]$$ (the second term → 0)

$$r_{v_1 v_2}[k] = E[s[n]\, v_2^*[n-k]]$$

$$= r_{x v_2}[k]$$

FROM TEXTBOOK

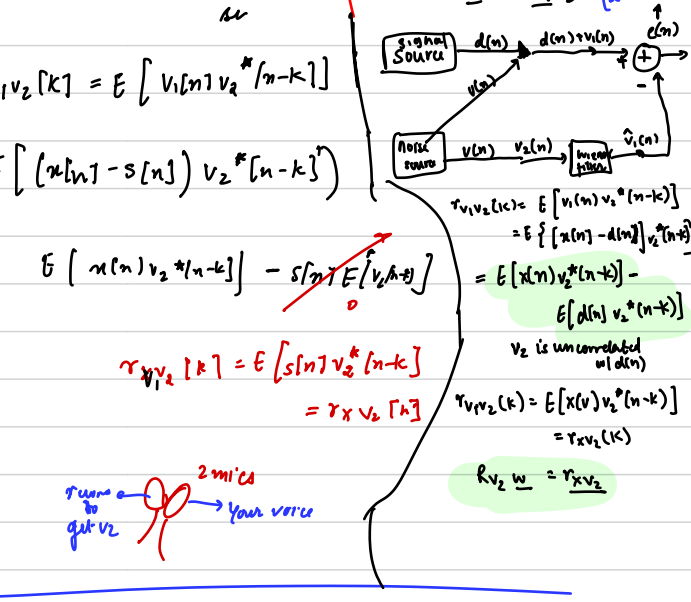
$x[n] = d[n] + v_1[n]$ (noise)

in filter autocorrelating noise must be known. Here, the info is obtained from a secondary source

→ $v_2(n)$ $v_2(n)$ & $v_1(n)$ are correlated. --- but not equal

$\hat{d}(n) = x(n) - \hat{v}_1(n)$

$R_{v_2} \underline{w} = r_{v_1 v_2}$ $d(n) - [x(n) - \hat{v}_1(n)]$



$r_{v_1 v_2}(k) = E[v_1(n)\, v_2^*(n-k)]$

$= E\{[x(n) - d[n]]\, v_2^*(n-k)\}$

$= E[x(n) v_2^*(n-k)] - E[d[n]\, v_2^*(n-k)]$

v_2 is uncorrelated w/ $d(n)$

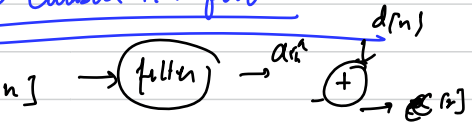
$r_{v_1 v_2}(k) = E[x(n) v_2^*(n-k)]$

$= r_{x v_2}(k)$

$R_{v_2} \underline{w} = \underline{r_{x v_2}}$

Causality is good, but not necessary if there a good filter.

Non Causal IIR filters



$$\hat{d}[n] = x[n] * h[n] \quad \sum_{i=-\infty}^{\infty} h[i]\, x[n-i]$$

$\min J = E(|e[n]|^2)$

By orthogonality principle

$E(e[n]\, x^*[n-k]) = 0$

$$E\left(\left(d[n] - \sum_{i=-\infty}^{\infty} h[i]\, x^*[n-k]\right)\right)$$

$$r_{dx} = \sum_{i=-\infty}^{\infty} h[i]\, r_x[k-i] = h(k) * R(k)$$

Summation limits !!

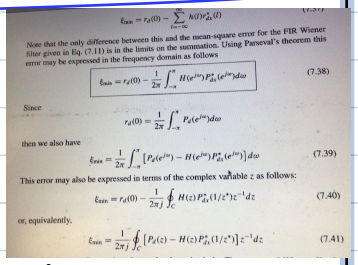
take DTFT

$S_{dx}(\omega) = H(\omega) S_x(\omega)$

$H(\omega) = \frac{S_{dx}(\omega)}{S_x(\omega)}$ → Wiener filter

↓ cross power spectral density.

$S_{dx}(\omega) \xleftrightarrow{DTFT} r_{dx}[k]$



Note that the only difference between this and the mean-square error for the FIR Wiener filter given in Eq. (7.11) is in the limits on the summation. Using Parseval's theorem this error may be expressed in the frequency domain as follows

$$\xi_{min} = r_d(0) - \frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{j\omega}) P_{dx}^*(e^{j\omega}) d\omega \quad (7.38)$$

Since

$$r_d(0) = \frac{1}{2\pi}\int_{-\pi}^{\pi} P_d(e^{j\omega}) d\omega$$

then we also have

$$\xi_{min} = \frac{1}{2\pi}\int_{-\pi}^{\pi} [P_d(e^{j\omega}) - H(e^{j\omega}) P_{dx}^*(e^{j\omega})] d\omega \quad (7.39)$$

This error may also be expressed in terms of the complex variable z as follows:

$$\xi_{min} = r_d(0) - \frac{1}{2\pi j}\oint_C H(z) P_{dx}^*(1/z^*) z^{-1} dz \quad (7.40)$$

or, equivalently,

$$\xi_{min} = \frac{1}{2\pi j}\oint_C [P_d(z) - H(z) P_{dx}^*(1/z^*)] z^{-1} dz \quad (7.41)$$

Applications

→ Noise Cancelling

$x[n] = d[n] + v[n]$ $\quad d \perp v$

$r_{dx}[k] = r_d[k] \rightarrow S_{dx}(\omega) = S_d(\omega)$

$r_x[k] = r_d[k] + r_v[k] \rightarrow S_x(\omega)$

$= S_d(\omega) + S_v(\omega)$

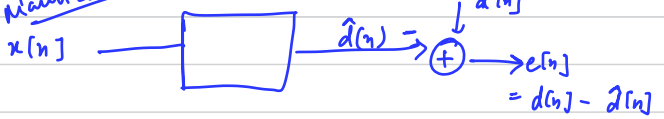
$H(\omega) = \frac{S_x(\omega)}{S_d(\omega) + S_v(\omega)}$ for non-zero filtering.

for a constant desired signal,

If noise is dominating; $H(\omega)$ ↓

If noise is less $H(\omega)$ ↑

recap March 26th



wiener filter → $\min J = E[|e(n)^2|]$

↳ not a function of n $= E[|e(n)|^2]$

This is all under the assumption that $x[n]$ and $d[n]$ are WSS and jointly WSS.

↳ $R_x \underline{h} = \underline{r_{dx}}$

Mar 26th

What if stationarity does not hold?

$e[n]^2$ is a function of n, so J is a function of n.

Now the filter is time dependant

$J[n] = E[|e[n]|^2]$ depends on n.

minimize $J[n]$

for $h[0] \cdots h[p]$

range of values for which operation holds

$e[n] = d[n] - \sum_{i=0}^{p} h[i] x[n-i]$

how min $J[n]$

$h_n[0] \cdots h_n[p]$ → dependance on n.

$\underline{h_n}^* = \arg\min_{h_n} J[n]$

NOT CONJUGATE; SPECIFIC SOLUTION (also a vector)

Using orthogonality principle

$E[e[n] x^*[n-k]] = 0 \quad \forall k = 0 \cdots p.$

min J represented by $e[n] = d[n] - \sum_{i=0}^{p} h[i] x[n-i]$

$\sum_{i=0}^{p} h_n[i] \cdot E\{x[n-i] x^*[n-k]\} = E\{d[n] x^*[n-k]\}$

↑ here we could directly like this as R_x since its not wss.

$r_x[n-i, n-k]$ is better notation

In general;

$\underline{R_x[n]}\ \underline{h} = \underline{r_{dx}(n)}$

changes w/ n.

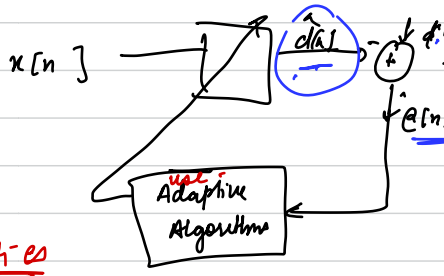
$\underline{h_n} = \underline{R_x^{-1}[n]}\ \underline{r_{dx}[n]}$

a lot of entries still probably are similar to signal at a previous time step.

$h_{n+1} = h_n + \Delta h_n$

↳ Concept of adaptive filters

Less computationally expensive as compared to optimal filters.



means keeps happening

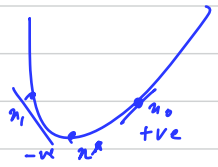
Properties

→ If signals are stationary, adaptation filters eventually tends to converge to wiener filters.

→ Adaptive filter should adapt to changing statistics

Gradient Descent / Steepest descent

→ Iterative optimisation technique



To reach a point of minima; change/move x opposite to the gradient.

$x_0 \to x_0$

$x_1 \to x_0 - \mu f'(x_0)$ where μ is learning rate $\mu > 0$.

$x_2 \to x_1 - \mu f'(x_1)$

→ As you come closer $f'(x_n)$ is very close to zero → converge.

→ Depends on learning rate

2D →

$f(x,y) \to x^*, y^*$ for argmin;

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \qquad h = \begin{bmatrix} x \\ y \end{bmatrix}$$

$f(\underline{h})$

$\underline{h}_{n+1} = \underline{h}_n - \mu \nabla f(\underline{h}_n)$

points against the direction of steepest increase.

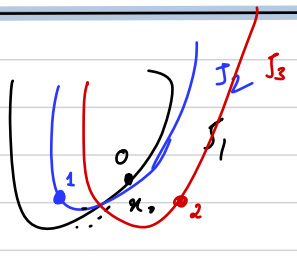
Gradient Descent on Adaptive Filters

$J[n] = \mathbb{E}[|e[n]|^2]$

$\underline{h}_{n+1} = \underline{h}_n - \mu \nabla J[n]$ $\quad J \leftarrow h_n$

$$\underline{h}_n = \begin{bmatrix} h_n[0] \\ \vdots \\ h_n[p] \end{bmatrix}$$ → change all entries as time goes on.

The filter keeps improving for changing statistics. This is not exactly like weights and biases grad descent. Our $J[n]$ keeps changing for different n. We operate here assuming it doesn't change too much.



A better way to understand this ---→

Our solution may not be optimal, but it will adapt as per the error.

This is a very idealistic way of looking at this.

$$J[n] = E\left[|e[n]|^2\right] = E\left[e[n]\,e^*[n]\right]$$

$$\nabla_{h_n} J[n] = E\left[\nabla_{h_n} e[n]\; e^*[n]\right]$$

$$e[n] = d[n] - \sum_{i=0}^{p} h_n[i]\, x[n-i]$$

$$\nabla_{h_n} e[n] = \begin{bmatrix} -x[n] \\ \{\text{wrt } h_n[0]\} \\ -x[n-1] \\ \{\text{wrt } h_n[1]\} \\ -x[n-2] \\ \{\text{wrt } h_n[2]\} \\ \vdots \\ -x[n-p] \end{bmatrix} = \begin{bmatrix} -x[n] \\ -x[n-1] \\ \vdots \\ -x[n-p] \end{bmatrix} = -\underline{x}_n$$

$$\underline{h}_{n+1} = \underline{h}_n + \mu E\left[e^*[n]\,\underline{x}[n]\right]$$

{for each n - so not a vector}

Formula 1

The steepest-descent adaptive filter converges to the wiener filter in the limit [for stationary signals]

$$\lim_{n\to\infty} \underline{h}_n = R_x^{-1} r_{dx} \quad \text{for } 0 < \mu < \frac{2}{\lambda_{max}}$$

where λ_{max} the the largest eigenvalue of R_x.

28/3

→ Adaptive filters are used for stationary signals since they converge to a solution.

→ You might also have cases where the matrix is "too large" to invert → not stable

for that you'll have to see condition number of e.g. matrix → $\frac{\lambda_{max}}{\lambda_{min}}$ → solution.

The steepest-descent adaptive filter converges to the wiener filter in the limit [for stationary signals]

$$\lim_{n\to\infty} \underline{h}_n = R_x^{-1} \underline{r}_{dx} \quad \text{for } 0 < \mu < \frac{2}{\lambda_{max}}$$

where λ_{max} the the largest eigenvalue of R_x.

GRADIENT DESCENT : gradient fixed $\nabla E[e[n]x^*[n]]$

Proof:

$$\underline{h}_{n+1} = \underline{h}_n + \mu E[e[n]\underline{x}^*[n]] \quad \leftarrow \underline{r}_{dx} - R_x \underline{h}_n$$

For stationary case: $E\left\{\left(d(n) - \sum_{i=0}^{p} h(i)\, x(n-i)\right) x^*[n]\right\}$

$$\underline{r}_{dx} - R_x \underline{h}_n$$

$$\underline{h}_{n+1} = \underline{h}_n + \mu(\underline{r}_{dx} - R_x \underline{h}_n)$$

$h_{opt} = R_x^{-1} r_{dx}$

$$\underline{h}_{n+1} - \underline{h}_{opt} = (I - \mu R_x)\underline{h}_n + \mu R_x \underline{h}_{opt} - \underline{h}_{opt}$$

$$\underline{h}_{n+1} - \underline{h}_{opt} = (I - \mu R_x)\underline{h}_n - (I - \mu R_x)\underline{h}_{opt}$$

$$\underline{h}_{n+1} - \underline{h}_{opt} = (I - \mu R_x)(\underline{h}_n - \underline{h}_{opt})$$

$\leftarrow C_{n+1} \rightarrow$ $\quad \leftarrow C_n \rightarrow$

$$C_{n+1} = (I - \mu R_x) C_n$$

so

$$C_{n+1} = \underbrace{(I - \mu R_x)}_{A}{}^{n+1} C_0$$

For convergence

as $n \to \infty$, $I - \mu R_x < 0$; $A \to 0$;

$C_{n+1} \to 0$; difference b/w our filter and optimal filter $\to 0$.

otherwise similar to SVD

EVD ONLY FOR SQUARE MATRIX

EVD $\longrightarrow$ $\boxed{R_x = V D V^H}$

eigenvalue decomposition

where $VV^H = I$ $\longrightarrow$ unitary V, $V^H = V^{-1}$

if R_x is symmetric and positive definite; we can write the EVD.

$R_x \underline{v} = \lambda \underline{v}$ $\quad D \to$ contains all eigenvalues of R_x.

So; putting this into the previous equation

$$I - \mu R_x = VV^H - \mu V D V^H$$

$$V(I - \mu D)V^H$$

so $(I - \mu R_x)^{n+1} = \left(V(I-\mu D)V^H\right)^{n+1}$

when you expand this

in the form of $V(I-\mu D)V^H V(I-\mu D)V^H$ etc ....

$V(I-\mu D)^{n+1} V^H$

the entries of this is

$$\begin{bmatrix} (1-\mu\lambda_1)^{n+1} & 0 & 0 & \cdots & 0 \\ 0 & (1-\mu\lambda_2)^{n+1} & & & \\ 0 & 0 & (1-\mu\lambda_3)^{n+1} & & \\ \vdots & & & \ddots & \\ & & & & (1-\mu\lambda_{p+1})^{n+1} \end{bmatrix}$$

where $\lambda_i > 0$ [positive definite]

so for this to converge as $n \to \infty$.

$-1 < 1 - \mu\lambda_i < 1$ for $i = 1 \cdots p+1$

$-2 < -\mu\lambda_i < 0$

$0 < \mu\lambda_i < 2$

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

hence $\mu < \frac{2}{\lambda_{max}}$ } proved

LMS filter [Least-mean squared filter]

$$\underline{h}_{n+1} = \underline{h}_n + \mu E(e[n]x^*[n])$$

$\uparrow r_{dx} - R_x \underline{h}_n$

we might not have these statistics.

To estimate this we can assume ergodicity.

$$E(e[n]x^*[n]) \approx \frac{1}{N} \sum_{K=0}^{N-1} e[n-k]\, x^*[n-k]$$

AT TIME n; I want to use the past $N-1$ values

This kind of approx works if data is not changing that much.

LMS says that :

$$E[e(n)\,x^*(n)] \approx e(n)\,x^*(n)$$

very crude approximation; but works out practically. Decreases computation significantly

LMS → $\underline{h}_{n+1} = \underline{h}_n + \mu\, e(n)\, x^*(n)$

STOCHASTIC GRADIENT DESCENT

In gradient descent you look at the best gradient, and that remains fixed $E[e(n)x^*(n)]$ but here every n value has a different gradient → adds some randomness to the process // stochasticity

Convergence of LMS

For jointly WSS processes $x[n]$ and $d[n]$, the LMS converges in the mean if $\lim_{n\to\infty} E\{\underline{h}_n\} \to \underline{h}_{opt}$

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

ensuring this is not easy; since this is different for every R_x → Hard to compute -

Side note: before it said that $\underline{h}_n \to \underline{h}_{opt}$ here it says on average; it will converge @ infinity

$$\lambda_{max} < \sum_{i=1}^{p+1} \lambda_i = \text{Trace}(R_x) \quad \text{All } \lambda_i \text{ are +ve.}$$

$$= (p+1)\, E[|x(n)|^2] = (p+1)\, r_x(0)$$

Now using ergodicity $E[|x(n)|^2] \approx \frac{1}{N}\sum_{i=0}^{N-1} |x(n-i)|^2$

So;

$$\lambda_{max} < (p+1) \times \frac{1}{(p+1)} \sum_{i=0}^{p} |x[n-i]|^2$$

$$\lambda_{max} < \|x[n]\|_2^2$$

ℓ_2 norm square of a vector -

Norm of A vector:

$$\underline{v} = \begin{bmatrix} v_1 \\ \vdots \\ v_N \end{bmatrix} \qquad \ell_2 \text{ norm}: \|\underline{v}\|_2 = \sqrt{|v_1|^2 + |v_2|^2 \cdots |v_N|^2}$$

$$0 < \mu < \frac{2}{\|x\|_2^2} \rightarrow \boxed{0 < \beta < 2}$$

Normalised LMS adaptive filter

$$\underline{h}_{n+1} = \underline{h}_n + \underbrace{\mu \|\underline{x}\|_2^2}_{\beta} \left(\frac{e[n]\,\underline{x}[n]}{\|\underline{x}[n]\|_2^2} \right)$$

$$\underline{h}_{n+1} = \underline{h}_n + \beta\, e(n)\, \frac{\underline{x}(n)}{\|\underline{x}\|_2^2}$$

Effective μ is changing w/ time -

Problem:

If signal energy goes to zero; change will go very large.

For stability reasons; $\epsilon > 0$

$$\underline{h}_{n+1} = \underline{h}_n + \beta\, e[n]\, \frac{\underline{x}[n]}{\|\underline{x}\|_2^2 + \epsilon}$$

April 6th

Recap

LMS: $\underline{h}_{n+1} = \underline{h}_n + \mu\, e(n) x(n)$ $\quad 0 < \mu < \frac{2}{\lambda_{max}}$

N-LMS: (normalised)

$$\underline{h}_{n+1} = \underline{h}_n + \beta\, e[n]\, \frac{\underline{x}[n]}{\|\underline{x}[n]\|_2^2 + \epsilon} \qquad 0 < \beta < 2$$

Learning curve

$$J = E[|e[n]|^2]$$

Wiener: $J_{min} = r_d[0] - r_{dx}^H R_x^{-1} r_{dx}$

$\underline{h}_{opt} = R_x^{-1} r_{dx}$

convergence → filter is fixed

$e[n] = d[n] - \underline{x}^H(n)\underline{h}(n)$ (desired; stochastic; desired)

Adaptive Filters:

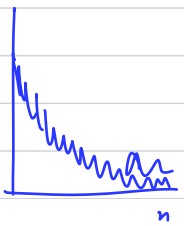
min J that can be achieved $\geq J_{min}$ (wiener) $\quad \underline{h}$ is also stochastic

$e[n] = d[n] - \underline{x}[n] \cdot \underline{h}[n]$ (deterministic; stochastic; stochastic)

Learning curve is the plot of J vs n.

Initialize the filter with something $[\underline{h}_0]$

$\lim_{n\to\infty} E\{\underline{h}_n\} = \underline{h}_{opt}$



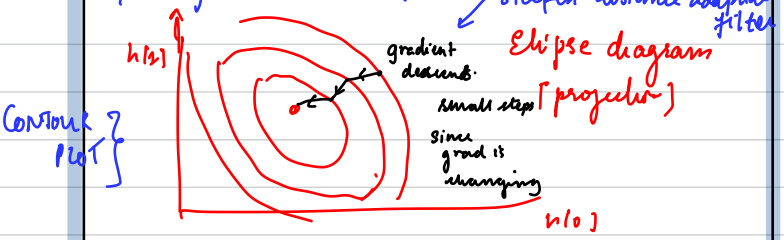
Contour plots

$J = \overline{E[|e[n]|^2]}$

↳ varying $h[0]$ & $h[2]$

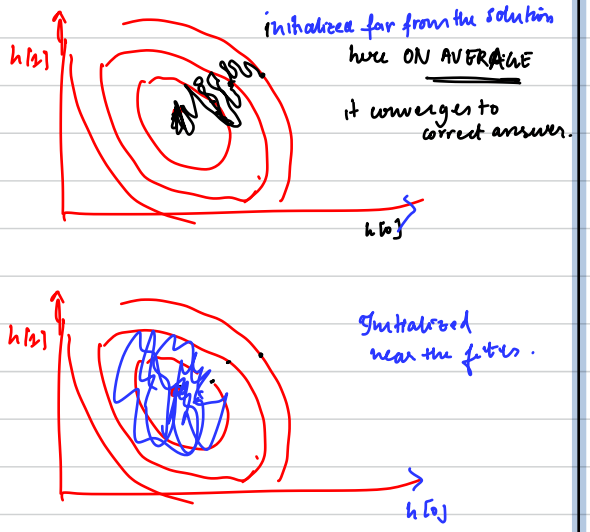
[3d graph look at scaled]

? Contour plots → representing 3D plots as 2D by taking horizontal cuts?



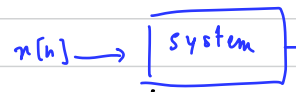
If gradient doesn't change, you can take large steps.

LMS filter [on average moving towards centre]



Applications

System Identification



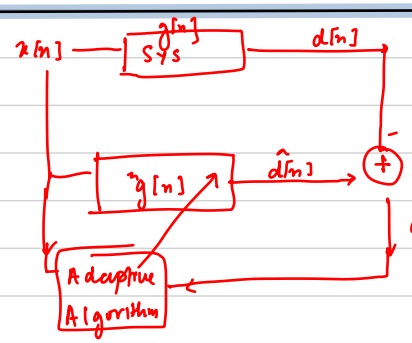
linear = $g[n]$

$d[n] = g[n] * x[n]$

from $x[n], d[n]$ finding $g[n]$ is process of **deconvolution**

$D(\omega) = G(\omega)X(\omega)$

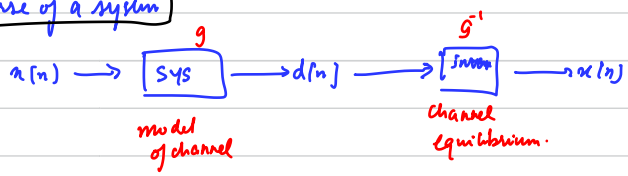
$G(\omega) = \frac{D(\omega)}{X(\omega)}$ given that $X(\omega) \neq 0$.



if error is minimised then the system g configured is close to actual system.

We can use this to also find an inverse system

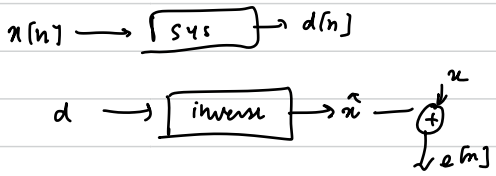
Inverse of a system



g^{-1} doesn't always exist

↑ $f[n]$ → converged if similar

$d[n] * f[n] = x[n] * g[n] * f[n] \approx x[n]$

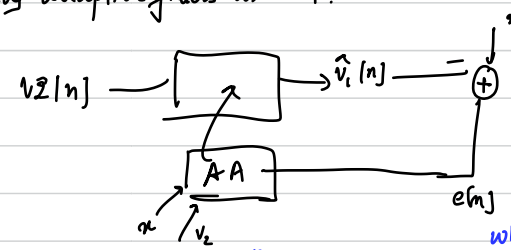


Noise Cancelling:

Primary: $x[n] = s[n] + v_1[n]$

Secondary $v_2[n]$

Using adaptive filters as well.



$\hat{v}_1[n] = z[n]$

why? $v_2[n]$ and $s[n]$ are uncorrelated

$J[n] = E[|e[n]|^2]$

$= E[|s[n] + v_1[n] - z[n]|^2]$ assuming real

$E[|s[n]|^2 + |v_1[n] - z[n]|^2 + 2s(n)(v_1(n) - z(n))]$

$E[|s(n)|^2]$ ← const $+ E[(v_1(n) - z(n))^2]$

$+ 2E(s(n)[v_1(n) - z(n)])$ ← zero as uncorrelated

$J[n] = E[|s[n]|^2] + E[|(v_1(n) - z(n)|^2]$

$|s[n]|^2$: constant; $|(v_1(n) - z(n)|^2$: term of minimization

Noise Cancellation example

[see slides]

$h_{n+1} = h_n + \mu\, e[n]\, x[n]$?

↳ $\mu\, \text{sgn}(e(n))\, x[n]$

↳ reduce computations (hardware complications)

Noise cancellation without secondary source

Primary : $x[n] = s[n] + v_1[n]$

Exploit fact that signal is uncorrelated w/ $v_1[n]$

Assumption : in general signal is correlated for long periods when compared to noise (for large k)

$r_s[k]$ → more non-zero values, spread out

$r_v[k]$ → larger k more likely to be zero.

Go backwards in time; signal can predict its self but noise cannot.

Condition number: how easy it is to solve a set of equations for a filter

April 12th

Excess mean squared error : error caused by using LMS instead of gradient error

Stationary signals: Wiener filter has [minimum mean squared error]

All these filters work on MMSE

$J = E(|e(n)|^2)$

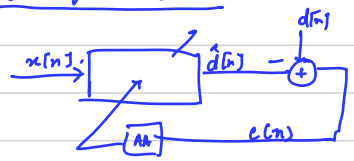
- Gradient Descent adaptive filter
- Stochastic gradient descent [LMS, NLMS] (both)
- Newton's [needs the statistics of the signals]

What if we don't have the statistics?

↳ Least squares error [LSE]

↳ Both; RLS → both;

Least squared errors



$h_{n+1} = h_n + \nabla h$

Loss function :

$$J[n] = \sum_{i=0}^{n} |e[i]|^2$$

→ different to most since you are trying to minimise the total error upto that point

computationally expensive ?

so

Recursive least square error (RLS) filters

↳ compared to LMS ↳ faster in convergence ↳ slightly more instable

→ If $d[n]$ is non linear version of $x[n]$ then LTI systems can't estimate it.

→ Neural networks have non linearities ; work in similar ways.

Non adaptive version of :

Least squares error $J[n] = \sum_{i=0}^{n} |e[i]|^2$

$x[n] \longrightarrow [h[n]] \longrightarrow \hat{d}[n]$

$= \|\underline{e}\|_2^2$

$\underline{e} = [e(0)\ e(1)\ \cdots\ e(n)]^T$

$e(i) = d(i) - \sum_{k=0}^{p} h[k]\, x[i-k]$

put this in a vector form

if k changes ; the filter is still the same

using vector notation

each place you evaluate for

$$\begin{bmatrix} e(0) \\ e(1) \\ \vdots \\ e(n) \end{bmatrix} = \begin{bmatrix} d(0) - \sum_{k=0}^{p} h[k]\, x[-k] \\ \vdots \\ d(n) - \sum_{k=0}^{p} h[k]\, x[n-k] \end{bmatrix}$$

$$\begin{bmatrix} e(0) \\ e(1) \\ \vdots \\ e(n) \end{bmatrix} = \begin{bmatrix} d(0) \\ d(1) \\ \vdots \\ d(n) \end{bmatrix} - \begin{bmatrix} x(0) & x(-1) & \cdots & x(-p) \\ x(1) & x(0) & \cdots & x(1-p) \\ x(2) & & & x(2-p) \\ \vdots & & & \vdots \\ x(n) & x(n-1) & \cdots & x(n-p) \end{bmatrix} \begin{bmatrix} h[0] \\ h[1] \\ \vdots \\ h[p] \end{bmatrix}$$

$\underline{e} = \underline{d} - X\underline{h}$

find $\underline{h}$ such that $J = \|e\|_2^2$ minimizes.

This analysis can be done for real/complex; here we continue w/ real :

$J = e^T e = (\underline{d} - X\underline{h})^T (\underline{d} - X\underline{h})$

$J = \underline{d}^T\underline{d} - \underline{h}^T X^T \underline{d} - \underline{d}^T X \underline{h} + \underline{h}^T X^T X \underline{h}$

$$J = \underline{d}^T\underline{d} - \underline{h}^T\underline{x}^T\underline{d} - \underline{d}^T\underline{x}\,\underline{h} + \underline{h}^T\underline{x}^T\underline{x}\,\underline{h}$$

To find h, $\boxed{\nabla_h J = 0}$

In general

If $f(\underline{h}) = \underline{h}^T\underline{a} = \underline{a}^T\underline{h}$

$\nabla_h f(\underline{h}) = \underline{a}$

$g(\underline{h}) = \underline{h}^T M \underline{h}$ (M: symmetric)

$\nabla g(\underline{h}) = M\underline{h} + \underline{h}^T M$

$= 2M\underline{h}$ (if not symmetric)

(if symmetric)

$$\nabla_h J = 0 - \underline{x}^T\underline{d} - \underline{d}^T\underline{x} + 2\underline{x}^T\underline{x}\,\underline{h}$$

$$\nabla_h J = -2\underline{x}^T\underline{d} + 2\underline{x}^T\underline{x}\,\underline{h} = 0$$

$$\underline{x}^T\underline{x}\,\underline{h} = \underline{x}^T\underline{d}$$

analogous to wiener

assuming $\underline{x}^T\underline{x}$ is invertible

$$\underline{h} = (\underline{x}^T\underline{x})^{-1}\underline{x}^T\underline{d}$$

← pseudo inverse of x

$x^+ = (x^Tx)^{-1}x^T$

$\underline{h} = \underline{x}^+\underline{d}$

→ has some properties of inversion; but not all.

Linear Regression

$\underline{x}\,\underline{h} = \underline{d}$ If x was invertible you'd get something like this. But if it's not we can have pseudo inverse.

at time n → solve for the best h at time n.

$J(\underline{h}_n) = \underline{e}_n^T\underline{e}_n$

$\underline{h}_n = (\underline{x}_n^T\underline{x}_n)^{-1}\underline{x}_n^T\underline{d}_n$

$R_n = \underline{x}_n^T\underline{x}_n = \sum_{i=0}^{n} \underline{x}_i^T\underline{x}_i$

at time $n+1$

$\underline{h}_{n+1} = (\underline{x}_{n+1}^T\underline{x}_{n+1})^{-1}\underline{x}_{n+1}^T\underline{d}_{n+1}$

every time instance you have to re-calculate.

It won't change much; only rows will be added.

Solving for h_n everytime is not efficient

Recursive Least Squares

We can show that (assuming WSS and ergodicity)

for large n → $\underline{x}_n^T\underline{x}_n = nR_x$

(0,0 term $x[0]^2 + x[1]^2 + \cdots x[n]$)

$= n r_x[k]$ (ergodicity in autocorr.)

here $\underline{h}_{n+1}$ depends on specific $\underline{x}_n^T$ $\underline{d}_n$ as well as statistics. Even for the same stationary process, at every time step filter will be different. More data driven.

Similarly; for large n; $\underline{x}_n^T\underline{d}_n \to r_{dx}$

ergodicity in cross correlation

W/K: $\text{inv}(A) + \text{inv}(B) \neq \text{inv}(A+B)$

Matrix Inversion Lemma

Woodbury matrix Identity

A, B → positive definite matrices $(m \times m)$

C → $m \times n$

D → positive definite $N \times N$

A, B, D are invertible

If $A = B^{-1} + CD^{-1}C^T$

$\Rightarrow A^{-1} = B - BC(D + C^TBC)^{-1}C^TB$

now $\underline{x}_{n+1}^T\underline{x}_{n+1}$ (augmented by a row) $= \underline{x}_n^T\underline{x}_n + \underline{z}\,\underline{z}^T$

$R_{n+1} = R_n + \underline{z}\,\underline{z}^T$

solve this recursively

$$X_{n+1} = \begin{bmatrix} X_n \\ \underline{z}^T \end{bmatrix}$$

$C = I$

now

$$\left[X_{n+1}^T X_{n+1}\right]^{-1} = \left[X_n^T X_n\right]^{-1} - \left[X_n^T X_n\right]\left(\underline{z}\,\underline{z}^T + X_n^T X_n\right)^{-1}\left[X_n^T X_n\right]$$

April 16th

$\underline{x}_i^T = [x[i], x[i-1] \cdots x[i-p]]$ $R_n \geq p+1$ (outer product; ip → scalar, op → matrix)

$X_n = \begin{bmatrix} \underline{x}_0^T \\ \underline{x}_1^T \\ \vdots \end{bmatrix} \to (n+1) \times (p+1)$

We have a way to give less and less importance to previous.

$\lambda \in (0,1)$

$J[n] = e^2[n] + \lambda e^2(n-1) + \lambda^2 e^2(n-2) \cdots \to$

"constant window, long time diff doesn't affect"

$$J[n] = \sum_{i=0}^{n} \lambda^i e^2(n-i)$$

Exponentially decaying weights

similar equations; but the terms will have some λ value.

So least square filter is now: →

$R_n^d \underline{h}_n = \underline{r}_n^d$

$$R_n^d = \sum_{i=0}^{n} \lambda^{n-i}\underline{x}_i\underline{x}_i^T$$

lowest weightage to x_0x_1 highest to $n-1, n$

we can apply matrix inversion lemma

Keep in mind---

$\underline{x}_i^T = [x[i], x[i-1] \cdots x[i-p]]$ $R_n \geq p+1$ (outer product; ip → scalar, op → matrix)

$X_n = \begin{bmatrix} \underline{x}_0^T \\ \underline{x}_1^T \\ \vdots \end{bmatrix} \to (n+1) \times (p+1)$

$r_n = x^T d_n$

$\underline{z} \equiv \underline{x}^T$ → check green equation

vectors.

matrix inversion lemma

$$R_n = \lambda R_{n-1} + \underline{x}_n\,\underline{x}_n^T$$

We get the following RLS filter

Assuming signals & filters are all real-valued

scalar

① $h_n = h_{n-1} + \alpha[n]\,k_n$

② $\alpha[n] = d[n] - h_{n-1}^T x_n$ (error)

③ $k_n = \frac{\frac{1}{\lambda} P_{n-1} x_n}{1 + \frac{1}{\lambda} x_n^T P_{n-1} x_n}$ $\quad k_n \propto P_{n-1} x_n$

④ $P_n = \frac{1}{\lambda} P_{n-1} - \frac{1}{\lambda} k_n x_n^T P_{n-1}$ $\quad P_n = (R_n)^{-1}$

this stuff is due to matrix inversion lemma

---

p+1 order filter — LMS ⇒ $h_{n+1} = h_n + \mu e[n] x_n$ $\quad O(p)$ $\quad e(n) = d(n) - h_n^T x_n$

p+1 order filter — RLS ⇒ $h_{n+1} = h_n + \alpha[n+1] \cdot k_{n+1}$ $\quad \propto P_n x_{n+1}$ $\quad \leftarrow O(p^2) \rightarrow$

↑ better convergence

$d(n+1) - h_n^T x_{n+1}$ ← a priori error

if I continue to use old filter w/ new data, how much error do I have?

In LMS you choose and arbitrary step size μ.

Here $P[n]$ is a data-based modified step ⇒

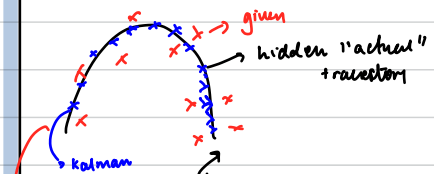
For RLS → Initialize h_0 and P_0 → start w/ arbitrary values

YOU NEED ENOUGH n to check if matrices are invertible

Check what happens when $\lambda = 0$ —

↳ Same applications as other filters ?

Kalman filter



Another form of noisy data

Kalman filter can provide a smooth trajectory for this

example: you have access of stream of velocity

↳ You can get position information ← linear relation

Examples:

position from angle deviations

↑ non linear relation b/w quantities

Transition Model: → senses how x evolves w/ time

$x_{n+1} = f(x_n, w_n)$ ← noise

how does the input change w/ time (depend on prev values)

Observation Model

$y_n = g(x_n, v_n)$

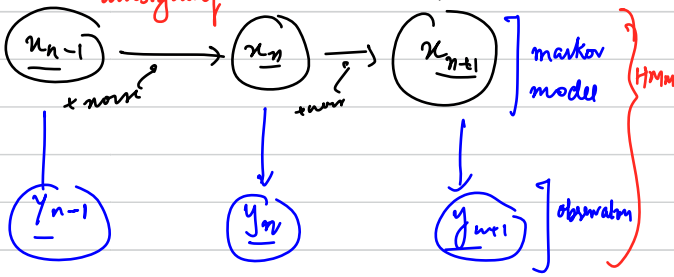
(how does the output depend on the input)

State evolution w/ time follows a markov model.

Kalman filter based on state space model ↗ markov model.

x_n → true state (unknown) ← hidden → only depends on previous state. ← markov

ambiguity



Hidden Markov Model

for Kalman filter

f, g have to be linear

w_n gaussian noise

Another Algorithm

Kalman Filter

1) State model / evolution

$x_{n+1} = f(x_n, w_n)$

special case ↳ $x_{n+1} = A_n x_n + w_n$ } linear & Gaussian

2) Observation model

Specifies $\begin{cases} y_n = g(x_n, v_n) \\ y_n = C_n x_n + v_n \end{cases}$ } linear and gaussian

KF is applicable for linear and gaussian models.

f and g in general are non linear & non gaussian

In KF

$A_n \rightarrow N \times N$ matrix (can change w/ time)

$C_n \rightarrow M \times N$ matrix } non necessarily stationary

$w_n \sim \mathcal{N}(0, Q)$

$v_n \sim \mathcal{N}(0, \tilde{Q}^{-1})$

Example: AR(1) process → $N(0)$.

State eqn. $x[n] = a_1 x[n-1] + w[n]$

Observation eqn / observing $x[n]$: $y[n] = x[n] + v[n]$

Given noisy observation $y[n]$, a_1 is known, estimate $x[n]$

$w \sim N(0, \sigma_w^2)$, $v \sim N(0, \sigma_v^2)$

Apply KF on $y[n]$ to estimate $x[n]$

not necessarily stationary

Example

$$AR(p): x[n] = \sum_{i=1}^{p} a_i x[n-i] + w[n]$$

$$y[n] = x[n] + v[n]$$

AR(p) process

$$\underline{x}_n = \begin{bmatrix} x[n] \\ x[n-1] \\ \vdots \\ x[n-p] \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & \cdots & a_p \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} x[n-1] \\ x[n-2] \\ \vdots \\ x[n-p-1] \end{bmatrix} + \begin{bmatrix} w[n] \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\underline{x}_n = A \underline{x}_{n-1} + \underline{w}_n$$

apply KF on $y[n]$ to estimate $x[n]$

Kalman filters

1) $\underline{x}_n = A_n \underline{x}_n + \underline{w}_n$
2) $\underline{y}_n = C_n \underline{x}_n + \underline{v}_n$

} Linear & gaussian.

→ estimate $\underline{x}_n$ if $\underline{y}_n$ measured.

→ more adv filters allow for non linearity

You have 2 info

↳ you know how your set is evolving

↳ $y[n]$

Solution

Let state estimate at time $n-1$ be given by $\hat{x}(n-1|n-1)$

① Prediction step

$$\hat{x}(n|n-1) = A_n \hat{x}(n-1|n-1)$$

[rough idea].

② Update step $ somehow solve for C_n

$$\hat{x}(n|n) = k_1(n)\, \hat{x}(n|n-1) + k(n)\, y_n$$

$k_1(n)$: some process of prediction step

$k(n)$: some process of y_n (measurement) [mimic C_n^{-1}, but C_n^{-1} doesn't exist]

Kind of like a weighted average.

found $k_1[n]$ and $k[n]$ matrices to get $\hat{x}(n|n)$ as the optimum linear estimate of $\underline{x}_n$

Prediction Error

$$e(n|n-1) = \underline{x}_n - \hat{x}(n|n-1)$$

Update Error

$$e(n|n) = \underline{x}_n - \hat{x}(n|n)$$

Derivation of eqn

$$E(e(n|n-1)) = E(x_n - \hat{x}(n|n-1))$$

$$= E(A_n x_{n-1} + w_n - A_n \hat{x}(n-1|n-1))$$

$$E[A_n(x_{n-1} - \hat{x}(n-1|n-1)] + 0$$

→ tends to zero.

WHY? LATER

① $E(e(n|n-1)) = A_n E(e(n-1|n-1)) = 0$

② $E(e(n|n)) = E(x_n - k_1(n)\hat{x}(n|n-1) - k(n) y_n)$

$$= E(x_n - k_1(n)(x_n - e(n|n-1)) - k(n)(C_n x_n + v_n))$$

$$= E[(I - k_1(n) - k(n) C_n) x_n] - k_1(n) E(e(n|n-1)) + k(n) E(v_n)$$

($E(e(n|n-1)) \to 0$, $E(v_n) \to 0$)

I want to see this to zero.

$I - k_1(n) + k(n) C_n = 0$

get relation b/w k_1 and k

$$k_1(n) = I - k(n) C_n$$

Kalman is a minimum MSE filter.

Statistics of error

$$P(n|n-1) = E(e(n|n-1) e^T(n|n-1))$$

$$P(n|n) = E(e(n|n) e^T(n|n))$$

} Error covariance matrices.

$$\hat{x}(n|n-1) = A_n \hat{x}(n-1|n-1)$$

we can do this differently by assume linear gaussian assumption.

This method finds computes the mean and covariance of your state vector → gaussian density

when it gaussian, this is optimal.
- if it not gaussian you can still.

April 23rd Kalman Filters (Summary)

$\hat{x}(n|n)$ & $P(n|n)$
$\hat{x}(n|n-1)$ & $P(n|n-1)$

State → state transition matrix

$$x_n = A_{n-1} x_{n-1} + w_n \quad \text{[true and unseen]}$$

Measurement → observation matrix

$$y_n = C_n x_n + v_n$$

Algorithm → Recursive state estimation algorithm -

Initialization: $\hat{x}(0|0)$ & $P(0|0)$ → uncertainty in obtaining the state.
↳ estimate

for $n = 1, 2 \ldots$

Prediction ↓

$$\hat{x}(n|n-1) = A_{n-1} \hat{x}(n-1|n-1)$$
$$P(n|n-1) = A_{n-1} P((n-1)|n-1) A_{n-1}^H + Q\hat{w}$$

does not give us any observation

Update: {incorporating what you have measured to make your estimate more accurate}

Compute $K(n)$

↳ Chapter 7 J

to find $K(n)$ you can use MMSE (minimize $E[\|e(n|n)\|_2^2]$)

$$\hat{x}(n|n) = \hat{x}(n|n-1) + K(n)\left[y_n - C_n \hat{x}(n|n-1)\right]$$

$$P(n|n) = \left[1 - K(n) C_n\right] P(n|n-1)$$

$K(n)$ doesn't depend on observation → CAN BE PRECOMPUTED

Mean square error

$$\Phi(n|n-1) = \text{Cov}\left(e(n|n-1)\right)$$
$$= \text{Cov}\left(\hat{x}(n|n-1)\right)$$

$$P(n|n-1) = \text{Cov}\left(e(n|n)\right) = \text{Cov}\left(\hat{x}(n|n)\right)$$

second order statistics.

$$E\left[\hat{x}(n|n)\right] = x_n$$

Show that: —

where P is covariance ⟺ represents uncertainty

$$P(n|n-1) = A_{n-1} P(n-1|n-1) A_{n-1}^T + Q_w(n)$$

$$P(n|n) = \left(I - K(n) C_n\right) P(n|n-1)$$

Update step

$$\hat{x}(n|n) = \left(I - K(n) C_n\right) \hat{x}(n|n-1) + K(n)\, y(n)$$

$$\hat{x}(n|n) = \hat{x}(n|n-1) + K(n)\left[y_n - C_n \hat{x}(n|n-1)\right]$$

kalman gain matrix. how much should error be optimized so update

To find $K(n)$ use a fn of trace: →
we use MMSE approach.
ie: Find $K(n)$ that minimizes

$$J = E\left(\|e(n|n)\|_2^2\right)$$

after some vector calculus to solve

$$\frac{\partial J}{\partial K} = 0$$

we get → initialize to zero and keep updating

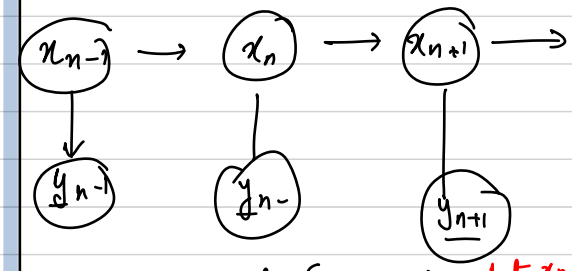
$$K(n) = P(n|n-1) C_n^T \left[C_n P(n|n-1) C_n^T + Q_v(n)\right]^{-1}$$

Examples of Kalman: IIT dog, trajectory of car, drawing on ipad.

The same goes for $P(n|n)$ → uncertainties (they dont depend on changing states & x values)

Kalman filtering is a special case of Bayesian filtering (recursive filter)

Bayesian filtering → Not restricted to linear & gaussian noise



State : $\underline{x}_n = f_n(\underline{x}_{n-1}, \underline{w}_n)$ — state $\underline{x}_n$ for given n is a random vector

Observation: $\underline{y}_n = g_n(\underline{x}_n, \underline{v}_n)$

Given that ik how x_{n-1} (almost) → x_n and observation of x_n

Can I find $P(x_{n-1}) \rightarrow P(x_n)$??

$P(x_{n-1}) = P(x_{n-1} | y_1, y_2 \cdots y_{n-1}) \rightarrow P(x_{n-1}|y_{1:n-1})$

$P(x_n) = P(x_n | y_1 \cdots y_{n-1}, y_n) \rightarrow P(x_n|y_{1:n})$

** estimate this

Initialize $P(x_0)$

y → doesn't exist, you haven't make any observation

Transition model

$P(x_n|x_{n-1})$ → since random noise driven evolution. → probability of obj going to n^{th} state from $n-1^{th}$ state

Observation → we observe y

$P(y_n|x_n)$ → given the state x in l.

Bayes Rule

$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$

also $P(A|B,C) = \frac{P(B|A,C)P(A|C)}{P(B|C)}$

$P(\underline{x}_n|\underline{y}_{1:n}) = \frac{P(\underline{y}_n|\underline{x}_n, y_{1:n-1}) P(\underline{x}_n|\underline{y}_{1:n-1})}{P(y_n|y_{1:n-1})}$

$P(y_n|y_{1:n-1})$ ↑ doesn't depend on x. can be not explained.

y_n depends only on x_n when x_n is present. If x_n was not present we could explain purpose of $y_{1:n-1}$

$P(\underline{x}_n|\underline{y}_{1:n}) = \eta\, P(\underline{y}_n|\underline{x}_n) P(\underline{x}_n|\underline{y}_{1:n-1})$

← likelihood fn →

Total probability law

$P(x) = \int_y P(x,y)\,dy$

$\int P(x|y)P(y)\,dy$

So: ← from total prob law

$P(\underline{x}_n|\underline{y}_{1:n-1}) = \int_{\underline{x}_{n-1}} P(x_n, x_{n-1}|\underline{y}_{1:n-1})\, d\underline{x}_{n-1}$

$= \int_{\underline{x}_{n-1}} P(\underline{x}_n|x_{n-1}, \underline{y}_{1:n-1})\, P(x_{n-1}|\underline{y}_{1:n-1})\, d\underline{x}_{n-1}$

↑ doesn't depend on y when prev (x_{n-1}) is present.

$= \int_{\underline{x}_{n-1}} P(\underline{x}_n|\underline{x}_{n-1})\, P(\underline{x}_{n-1}|\underline{y}_{1:n-1})\, d\underline{x}_{n-1}$

Soooo

$P(\underline{x}_n|\underline{y}_{1:n}) = \eta\, P(\underline{y}_n|\underline{x}_n) \int_{\underline{x}_{n-1}} P(\underline{x}_n|\underline{x}_{n-1})\, P(\underline{x}_{n-1}|\underline{y}_{n-1})\, d\underline{x}_{n-1}$

← posterior distribution → ← prior knowledge [prior distribution] →

Recursive bayesian filtering

$\eta = \frac{1}{P(\underline{y}_n|\underline{y}_{1:n-1})}$ → constant → doesn't depend on $\underline{x}_n$

↳ will scale it, not change shape

→ kalman is a special case

$P(\underline{x}_n|\underline{x}_{n-1})$ → transition → gaussian

$P(y_n|x_n)$ → gaussian

$P(x_{n-1}|\underline{y}_{1:n-1})$ → gaussian

$P(\underline{x}_n|\underline{y}_{1:n})$ ↳ gaussian

$\hat{x}(n|n)$ & $P(n|n)$ → mean & covariance

$\hat{x}(n|n-1)$ & $P(n|n-1)$ → mean & covariance of integral

$P(x_{n-1} \mid y_{1:n-1})$

$P(x_n \mid y_{1:n})$

Types of Bayesian filters –

Particle filter; Kalman filter

Another type of filtering & Extended Kalman filter

non linear function & → but taylor series expansion

Kalman filter from Bayesian?

@ $T = n-1$

$P(x_{n-1} \mid y_{1:n-1}) \sim$

$\mathcal{N}(x_{n-1}; \hat{x}(n-1|n-1), P(n-1|n-1))$

$\mathcal{N}(x; \mu, \Sigma)$

$\Rightarrow P(x)$

Transition

$x_n = A_{n-1} x_{n-1} + w_n$ (gaussian, gaussian; cov of this)

x_n ← constant and x_{n-1}

$P(x_n \mid x_{n-1}) \sim \mathcal{N}(x_n; A_{n-1} x_{n-1}; Q_w^n)$

Observation $y_n = C_n x_n + v_n$

$P(y_n \mid x_n)$

$\sim \mathcal{N}(y_n; C_n x_n, Q_v^n)$

So;

$P(x_n \mid y_{1:n})$ is also gaussian

After

$P(x_n \mid y_{1:n})$

$\sim \mathcal{N}(x_n; \hat{x}(n|n), P(n|n))$

mean is the point where gaussian distribution is maximum.

MAP & maximum aposteriori distribution

↑ is what you calculate using the Kalman filter. For gaussian maximum is at the mean.

April 25th

Examples

scalar: x_n and y_n

state model: $x_n = x_{n-1} + w_n$ → $0, \sigma_w^2$

observation model: $y_n = x_n + v_n$ → $0, \sigma_v^2$

} Example Brownian motion

Initialize $\hat{x}(0|0)$ $P(0|0)$

At time n;

Prediction Equation:

* $\hat{x}(n|n-1) = \hat{x}(n-1|n-1)$
* $P(n|n-1) = P(n|n-1) + \sigma_w^2$ ← the uncertainty measure changes by random variance

Update

Kalman Gain: $K_n = \frac{P(n-1|n-1) + \sigma_w^2}{P(n-1|n-1) + \sigma_w^2 + \sigma_v^2}$ ← uncertainty in state

uncertainty in observation $1 - K_n = \frac{-\sigma_v^2}{P(n-1|n-1) + \sigma_w^2 + \sigma_v^2}$

$x_0 = 0$

$\hat{x}(n|n) = \hat{x}(n|n-1) + K_n (y_n - \hat{x}(n|n-1))$

$\hat{x}(n|n) = (1 - K_n)\hat{x}(n|n-1) + K_n y_n$

Case $\sigma_v^2 \gg P[n-1|n-1] \cdot + \sigma_w^2$

$1 - K(n) \gg K_n$

less uncertain prediction

more uncertain prediction

here $A_w = I$, $C_n = I$ Q_w^v, Q_w^w

$$\hat{x}(n|n) = (1 - K_n)\hat{x}(n-1|n-1) + K_n y_n$$
$$= (1-K_n)\left[(1 - K_n)\hat{x}(n-2|n-2) + K_{n-1} y\right] + K_n y$$

as time goes on dependence on initialising reduces.

$e(n|n) = x_n - \hat{x}_{n|n}$

we want estimate to be unbaised;

$E(\hat{x}(n|n)) = x_n$

$E(e(n|n)) = E(x_n) - E(\hat{x}(n|n))$

$E(e(n|n)) = x_n - x_n$

$E(e(n|n)) = 0$

↳ here our estimate is unbiased.

spatial correlation ✓

no temporal correlation

$Q_w^n = E[w_n w_n^T]$

across time w_n & w_{n+1} may not be correlated too

[Example: noise at different sensors