DCSP-14: Matched Filter

Jianfeng Feng

Department of Computer Science Warwick Univ., UK

Jianfeng.feng@warwick.ac.uk

http://www.dcs.warwick.ac.uk/~feng/dcsp.html
Filters

- Stop or allow to pass for certain signals
Filter Types

There are four main classes of filter in widespread use:

- Lowpass
- highpass
- bandpass
- bandstop filters.

- The name are self-explanatory
- This four types are shown in the next figure
Filter Types: TFs

(i) $H(\omega) = 1$ for $0 \leq \omega \leq 2\pi - \omega_c$ and $2\pi - \omega_c \leq \omega \leq 2\pi$

(ii) $H(\omega)$ is constant for $0 \leq \omega \leq 2\pi$

(iii) $H(\omega) = 1$ for $0 \leq \omega \leq \omega_c$ and $2\pi - \omega_c \leq \omega \leq 2\pi$

(iv) $H(\omega)$ is constant for $0 \leq \omega \leq 2\pi - \omega_c$ and $2\pi - \omega_c \leq \omega \leq 2\pi$
General Filter Design

- Using Chebyshev or Butterworth polynomials to approximate a square wave function in the frequency domain.
- Any function can be approximated by polynomials.
- Optimal filter design aims to minimize these ripples.
- We are not discussing them here in details, but the principle is the same as our simple filter.

Ideal case
Polynomial approximation
Filters

- Stop or allow to pass for certain signals as we have talked before

- Detect certain signals such as radar etc (pattern recognitions)
Ex: Detect it when it comes close
Assume that we can detect

\[ S = (s_1, s_2, s_3, s_4, s_5, s_6) \]

To find \( a_i \)

\[
y(0) = a_0 x(0) + a_1 x(-1) + a_2 x(-2) + a_3 x(-3) + a_4 x(-4) + a_5 x(-5)
\]

so that we can find \( S \), more precisely

\[
Y(0) = a_0 s_6 + a_1 s_5 + a_2 s_4 + a_3 s_3 + a_4 s_2 + a_5 s_1
\]
At time -5 we have \( x(-5) = S_1 = 1 \) (length)

\[
Y(-5) = a_0 \cdot 1 + a_1 \cdot x(-6) + a_2 \cdot x(-7) + a_3 \cdot x(-8) + a_4 \cdot x(-9) + a_5 \cdot x(-10)
\]
At time -4 we have $x(-4) = S_2 = 2$ (length)

$Y(-5) = a_0 1 + a_1 x(-6) + a_2 x(-7) + a_3 x(-8) + a_4 x(-9) + a_5 x(-10)$

$Y(-4) = a_0 2 + a_1 1 + a_2 x(-6) + a_3 x(-7) + a_4 x(-8) + a_5 x(-9)$
At time -3 we have \( x(-3) = S_3 = 3 \) (length)

\[
Y(-5) = a_0 1 + a_1 x(-6) + a_2 x(-7) + a_3 x(-8) + a_4 x(-9) + a_5 x(-10)
\]

\[
Y(-4) = a_0 2 + a_1 1 + a_2 x(-6) + a_3 x(-7) + a_4 x(-8) + a_5 x(-9)
\]

\[
Y(-3) = a_0 3 + a_1 2 + a_2 1 + a_3 x(-4) + a_4 x(-5) + a_5 x(-6)
\]
At time -2 we have \( x(-2) = S_4 = 4 \) (length)
At time $-1$ we have $x(-1) = S_5 = 8$ (length)
At time 0 we have $x(0) = S_6 = 6$ (length)
The input signal is

\[ S = (s_1, s_2, s_3, s_4, s_5, s_6) \]

\[ = (1, 2, 3, 4, 8, 6) \]
Matched filters: Question

To find $a_i$

$$y(0) = a_0 6 + a_1 8 + a_2 4 + a_3 3 + a_4 2 + a_5 1$$

to detect that $S = (s_1, s_2, s_3, s_4, s_5, s_6)$ is here
Matched filters: visualization

\[ a_0 \cdot x(0) + a_1 \cdot x(-1) + a_2 \cdot x(-2) + a_3 \cdot x(-3) + a_4 \cdot x(-4) + a_5 \cdot x(-5) = y(0) \]

Matched !!!

incoming signals is matched by the coefficients
Data requirement

Without loss of generality, we can assume that

\[ S_i^2 = \frac{1}{N} \sum_{i=1}^{N} = 1 \] (normalized signals)

and we can assume that

\[ a_i^2 = \frac{1}{N} \sum_{i=1}^{N} = 1 \] (normalized coefficients)
Ideas

Remember that

\[ \sum_{i=0}^{N-1} (a_i - s_{N-i})^2 = 0 \]

If and only if \( a_i = s_{N-i} \) for all \( i \) which essentially says that the coefficients \( a \) and the incoming signal \( s \) are identical (matched, \( a_i = s_{N-i} \)).

Or equivalently

\[ a_0 S_N + a_1 S_{N-1} + \ldots + a_{N-1} S_1 = 1 \]

if and only if \( a_i = s_{N-i} \) for all \( i \)

\[ a_0 S_N + a_1 S_{N-1} + \ldots + a_{N-1} S_1 < 1 \]

otherwise
Matched filter

- Define $a_i = s_{N-i}$ (reversing the order)

$$y(n) = a_0 x(n) + a_1 x(n-1) + \ldots + a_N x(n-N)$$

So when the signal arrives we have

$$y(n) = S_N x(n) + S_{N-1} x(n-1) + \ldots + S_1 x(n-N)$$

$$= S_N S_N + S_{N-1} S_{N-1} + \ldots + S_1 S_1$$
At time $-5$ we have $y(-5) = 0.0877 \times 0.5262 = 0.0461$

$Y(-5) = a_0 x(-5) + a_1 x(-6) + \ldots$

$S = (0.0877 \ 0.1754 \ 0.2631 \ 0.3508 \ 0.7016 \ 0.5262)$

$a = (0.5262 \ 0.7016 \ 0.3508 \ 0.2631 \ 0.1754 \ 0.0877)$

$x = \begin{pmatrix} s_1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$

At time $-5$ we have $y(-5) = 0.0877 \times 0.5262 = 0.0461$
\[ Y(-4) = a_0 x(-4) + a_1 x(-5) + a_2 x(-6) + \ldots = 0.1538 \]

\[ \begin{array}{cccccc}
0.5262 & 0.1754 & 0.7016 & 0.0877 & 0.2631 & 0 \\
\end{array} \]

\[ \begin{array}{cccccc}
S = (0.0877 & 0.1754 & 0.2631 & 0.3508 & 0.7016 & 0.5262) \\
\end{array} \]

\[ \begin{array}{cccccc}
a = (0.5262 & 0.7016 & 0.3508 & 0.2631 & 0.1754 & 0.0877) \\
\end{array} \]

\[ \begin{array}{cccccc}
S_2 & S_1 & 0 & 0 & 0 & 0 \\
\end{array} \]
At time -3 we have

\[ Y(-3) = a_0 x(-3) + a_1 x(-4) + a_2 x(-5) + a_3 x(-6) \ldots = 0.2923 \]

\[
\begin{align*}
S &= (0.0877 \ 0.1754 \ 0.2631 \ 0.3508 \ 0.7016 \ 0.5262) \\
\mathbf{a} &= (0.5262 \ 0.7016 \ 0.3508 \ 0.2631 \ 0.1754 \ 0.0877)
\end{align*}
\]
At time -2 we have

\[ Y(-2) = a_0 x(-2) + a_1 x(-3) + a_2 x(-4) + a_3 x(-5) + a_4 x(-6) \ldots = \]

\[
\begin{array}{ccccccc}
0.5262 & 0.3508 & 0.7016 & 0.2631 & 0.3508 & 0.1754 & 0.2631 & 0.3508 \\
\end{array}
\]

\[
S = (0.0877 \ 0.1754 \ 0.2631 \ 0.3508 \ 0.7016 \ 0.5262) \\
\]

\[
a = (0.5262 \ 0.7016 \ 0.3508 \ 0.2631 \ 0.1754 \ 0.0877) \\
\]

\[
s_4 \quad s_3 \quad s_2 \quad s_1 \quad 0 \quad 0 \\
\]
At time $-1$ we have

$$y(-1) = a_0 x(-1) + a_1 x(-2) + a_2 x(-3) + a_3 x(-4) + a_4 x(-5) \ldots$$

$$S = (0.0877 \ 0.1754 \ 0.2631 \ 0.3508 \ 0.7016 \ 0.5262)$$

$$a = (0.5262 \ 0.7016 \ 0.3508 \ 0.2631 \ 0.1754 \ 0.0877)$$
S = (0.0877 0.1754 0.2631 0.3508 0.7016 0.5262)

a = (0.5262 0.7016 0.3508 0.2631 0.1754 0.0877)

At time 0 we have
Y(0) = a_0 x(0) + a_1 x(-1) + a_2 x(-2) + a_3 x(-3) + a_4 x(-4) + a_5 x(-5) + a_6 x(-6) = 1
At time 1 we have
\[ y(1) = a_0 x(1) + a_1 x(0) + a_2 x(-1) + a_3 x(-2) + a_4 x(-3) + a_5 x(-4) + a_6 x(-5) = 0.7961 \]
Outcome
Matlab Demo

- Matched-filter-demo
- It is a simple idea, but useful
- A bit theory first
Auto-correlation Function

\[ y(n) = \sum_{m=-\infty}^{\infty} a_m x(m) \]

\[ = s(m)s(n - m) \sum_{m=-\infty}^{\infty} \]

\[ = s(m)s(n + m) \sum_{m=-\infty}^{\infty} \]

\[ = r_{ss}(n) \]
Detection theory

- A means to quantify the ability to discern between information-bearing patterns and noise

- Patterns: stimulus in human, signal in machines

- Noise: random patterns that distract from the information
Can it be useful?

Any signal here?
Filter output
Filter output

```
s = rand(1,100);
a = s/sqrt(s'*s);
k = 100;
N = 100*k;
sigma = 0.2;
x = randn(1,N)*sigma;
signal = zeros(1,N);
for i = 3*N/k+1:N*4/k
    x(i) = a(i-3*N/k) + x(i);
signal(i) = a(i-3*N/k);
end
for i = 1:N*(k-1)/k
    c(i) = x([i:i+N/k-1])*s';
end
figure(1)
plot(x);
hold on
plot(signal,'r');
figure(2)
plot(c)
```
Matched filter

• The result is amazing

• It depends on SNR

• We will not go into details, but you might be able to investigate it using Matlab
Tomorrow
DCSP-15: Wiener Filter and Kalman Filter*

Jianfeng Feng

Department of Computer Science Warwick Univ., UK

Jianfeng.feng@warwick.ac.uk

http://www.dcs.warwick.ac.uk/~feng/dcsp.html
Norbert Wiener: Harvard awarded Wiener a PhD, when he was 17.
Example

The RGB format stores three color values, R, G and B, for each pixel.

```
RGB = imread('bush.png');
size(RGB)
an =
   1500   1200     3

imshow(RGB)
```
Example
Setup

Recorded signal $x(n,m) = s(n,m) + \xi(n,m)$

for example, an image of 1500 X1200
Setup

```
RGB = imread('bush.png');
I = rgb2gray(RGB);
J = imnoise(I,'gaussian',0,0.005);
figure, imshow(I),
figure, imshow(J)
```
To find a constant $a(m,n)$ such that

$$E \left( a(m,n) x(m,n) - s(m,n) \right)^2$$

as small as possible

$$y(m,n) = a(m,n) x(m,n)$$

Different from the filter before, $a(m,n)$ depends on $(m,n)$:

Adaptive filter
Setup

\[ E(ax - s)^2 = E(a^2 x^2 - 2axs + s^2) \]
\[ = a^2 Ex^2 - 2aExs + Es^2 \]

The quantity above is minimized if the derivative of it with respect to a is zero

\[ 2aEx^2 \quad 2Exs = 0 \]
\[ a = \frac{Exs}{Ex^2} = \frac{E(s + \text{)}s}{Es^2 + E^2} = \frac{Es^2}{Es^2 + E^2} = \frac{2}{2} E = \frac{2}{2} \] (optimal Wiener gain)
• Without loss of generality, we assume that $E_s=0$ and $E_{\xi}=0$ for simplicity of formulation.

• $S, \xi$ are independent
Algorithm

The filtered output is given by

\[ y(n_1, n_2) = m(n_1, n_2) + \sigma^2(n_1, n_2) \cdot \frac{\mathbb{E}^2(x(n_1, n_2) - m(n_1, n_2))}{\sigma^2(n_1, n_2)} \]

Note that the coefficient \( a \) depends on the position

\( \sigma \) is the local variance, \( \mathbb{E}\xi^2 \) is the global variance (noise)
Algorithm

\[ \mu = \frac{1}{NM} \sum_{n_1, n_2} x(n_1, n_2) \]

\[ \sigma^2 = \frac{1}{NM} \sum_{n_1, n_2} x^2(n_1, n_2) - \mu^2 \]

where summation is over an area of \( N \) and \( M \)

3X 3 area
(N=3, M=3)
Algorithm in Matlab

• `wiener2 lowpass` - filters an intensity image that has been degraded by constant power additive noise.

• `wiener2` uses a pixelwise adaptive Wiener method based on statistics estimated from a local neighborhood of each pixel.
Example

```matlab
RGB = imread('bush.png');
I = rgb2gray(RGB);
J = imnoise(I,'gaussian',0, 0.005);
K = wiener2(J,[5 5]);
figure, imshow(J),
figure, imshow(K)
```
Further issues

- Large M and N reduce noise, but blur the photo
- small M and N will not affect noise
- optimal size should be used

N=M=50

N=M=2
Further issues

• A more general version of Wiener: Kalman filter

• Kaman filter is widely used in many areas (still a very hot research topic)

• As an introductory course, we only give you a taste of its flavour
Kalman Filter: find out $x_k$

- Have two variables: state variable $x$ which is not observable

$$x_k = F_k \ x_{k-1} + B_k \ u_k + w_k$$

- Observable variable $z$ which we can observe, but is contaminated by noise

$$z_k = H_k \ x_k + v_k$$

where $w_k \sim N(0,Q_k)$ and $v_k \sim N(0,R_k)$ are noise, $F_k$ and $H_k$ are constants.
Kalman Filter

For given $X_{k-1|k-1}$ and $P_{k-1|k-1}$

Predict

• Predicted (a priori) state estimate $X_{k|k-1} = F_{k-1} X_{k-1|k-1} + B_k u_k$

• Predicted (a priori) estimate covariance $P_{k|k-1} = F_{k-1} P_{k-1|k-1} F_{k-1} + Q_k$

Update

• measurement residual $y_k = z_k - H_k X_{k|k-1}$

• measurement residual covariance $S_k = H_k P_{k|k-1} H_k + R_k$

• optimal Kalman gain $K_k = P_{k|k-1} H_k (S_k)^{-1}$

• updated (a posteriori) state estimate $X_{k|k} = X_{k|k-1} + K_k y_k$

• updated (a posteriori) estimate covariance $P_{k|k} = (I - K_k H_k) P_{k|k-1}$
Kalman Filter

- It is one of the most useful filters in the literature, as shown below

- We do not have time to go further, but do read around
Applications

- Attitude and Heading Reference Systems
- Autopilot
- Battery state of charge (SoC) estimation [1][2]
- Brain-computer interface
- Chaotic signals
- Tracking and Vertex Fitting of charged particles in Particle Detectors [35]
- Tracking of objects in computer vision
- Dynamic positioning
- Economics, in particular macroeconomics, time series, and econometrics
- Inertial guidance system
- Orbit Determination
- Power system state estimation
- Radar tracker
- Satellite navigation systems
- Seismology [3]
- Sensorless control of AC motor variable-frequency drives
- Simultaneous localization and mapping
- Speech enhancement
- Weather forecasting
- Navigation system
- 3D modeling
- Structural health monitoring
- Human sensorimotor processing[36]
Final slide

• Information theory

• FT (continuous, discrete, discrete + finite)

• Filters (band related filters, matched, Wiener + Kalman)

No lectures next week, but drop me a line if you need help for your assignment