TSRT14: Sensor Fusion Lecture 1

— Course overview
— Estimation theory for linear models

Gustaf Hendeby

gustaf.hendeby@liu.se

LINKÖPING UNIVERSITY

Course Overview

LINKÖPING
UNIVERSITY

Course Goals: study handbook

The student should after the course have the ability to describe the most important methods and algorithms for sensor fusion, and be able to apply these to sensor network, navigation and target tracking applications. More specifically, after the course the student should have the ability to:

- Understand the fundamental principles in estimation and detection theory.
- Implement algorithms for parameter estimation in linear and nonlinear models.
- Implement algorithms for detection and estimation of the position of a target in a sensor network.
- Apply the Kalman filter to linear state space models with a multitude of sensors.
- Apply nonlinear filters (extended Kalman filter, unscented Kalman filter, particle filter) to nonlinear or non-Gaussian state space models.
- Implement basic algorithms for simultaneous localization and mapping (SLAM).
- Describe and model the most common sensors used in sensor fusion applications.
- Implement the most common motion models in target tracking and navigation applications.
- Understand the interplay of the above in a few concrete real applications.

Lecture Plan

Le	Content	Ch
1	Course overview. Estimation theory for linear models	1, 2
2	Estimation theory for nonlinear models	3
3	Cramér-Rao lower bound. Models for sensor network	4
4	Detection theory. Filter theory	5, 6
5	Modeling and motion models	12–14
6	Kalman filter. Kalman filter approximations (EKF, UKF)	7, 8
7	Point-mass filter and particle filter	9
8	Particle filter theory. Marginalized particle filter	9
9	Simultaneous localization and mapping (SLAM)	11
10	Sensors and filter validation. Industry case study	14, 15

Course Administration

- Course homepage: `http://www.control.isy.liu.se/student/tsrt14/`
- Examination:
  - Two labs, including one lab report (make sure to sign up!)
  - Exam with a mixture of theoretical and computer exercises
- Contact information:
  - Examiner: Gustaf Hendeby (`gustaf.hendeby@.liu.se`)
  - Teaching assistant: Chuan Huang (`chuan.huang@liu.se`)

Gustaf Hendeby

Chuan Huang

Course Material

- Literature: *Statistical Sensor Fusion.* Fredrik Gustafsson. Studentlitteratur, 3 ed, 2018.
- Exercises: *Statistical Sensor Fusion. Exercises.* Christian Lundquist, Zoran Sjanic, and Fredrik Gustafsson. Studentlitteratur, 2015.
- Lab 1 and 2 compendium (links on the homepage)
- Matlab toolbox *Signal and Systems Lab* (link on the homepage)
- Toolbox manual
- Android app *Sensor Fusion app* available in Google Play Store and Matlab and Java real-time interface files.

Course Material

- Literature: *Statistical Sensor Fusion.* Fredrik Gustafsson. Studentlitteratur, 3 ed, 2018.
- Exercises: *Statistical Sensor Fusion. Exercises.* Christian Lundquist, Zoran Sjanic, and Fredrik Gustafsson. Studentlitteratur, 2015.
- Lab 1 and 2 compendium (links on the homepage)
- Matlab toolbox *Signal and Systems Lab* (link on the homepage)
- Toolbox manual
- Android app *Sensor Fusion app* available in Google Play Store and Matlab and Java real-time interface files.
- Videos from distance mode are available as a **complement** to the lectures (link on the homepage).

Changes Since 2023

- Continued lab improvements:
  - Technical improvements
  - Better motivation and explanation (today!)
- Course expectations:
  - Quiz to help clarifying prerequisites, and, if needed, suggestions for suitable material.

Le 1: linear estimation theory and sensor networks

Whiteboard:

- Basic definitions and derivations.

Slides:

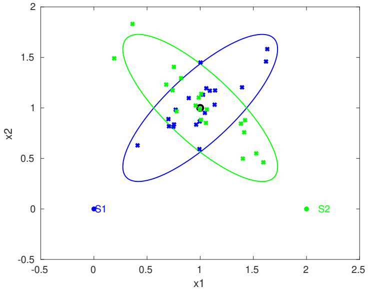
- Summaries
- Examples
- Code examples
- Algorithms

Goal:

- Localization of a device, a user, or an object in a sensor network!

A Simple Sensor Network Example

Triangulation, as used by seamen for a long time. Assume two sensors that each measure bearing to target accurately, and range less accurately (or *vice versa*). How to fuse these measurements?

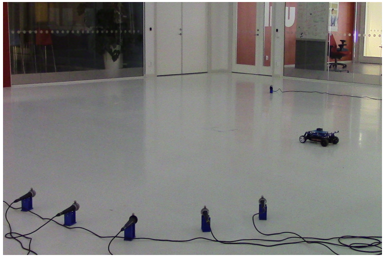


- Use all four measurements with the same weight — LS. Poor range information may destroy the estimate.
- Discard uncertain range information, gives a triangulation approach with algebraic solution.
- Weigh the information according to its precision — WLS.

A Standard Example

Laboration 1, but similar principles used in radio and underwater applications.

Vehicle sends out regular acoustic ‘pings’. Time synchronized microphones detect ping arrival times.



- **Localization:** Can the vehicle be located if the microphone positions are known? If the ping times are known? If the ping times are unknown?
- **Mapping:** Can the microphone positions be estimated if the vehicle position is known?
- **Simultaneous Localization and Mapping (SLAM):** Can both the target and microphone positions be estimated?

Another Standard Example

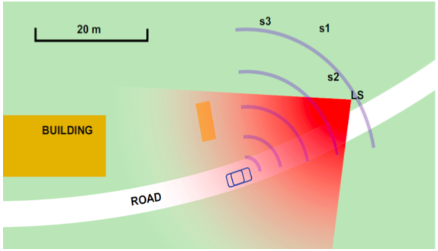
WiFi base stations in indoor environment.

Receiver gets BSSID (identity) and RSS (received signal strength).

- Can the receiver be positioned if the WiFi base station positions are known?
- Can the receiver be positioned if the WiFi base station positions are unknown, but there is a map over RSS as a function of position (fingerprinting)?

A Hard Example

A speaker sends out a $10\,\text{kHz}$ ($120\,\text{dB!}$) tone.
Several vehicles pass by.
A microphone network computes Doppler frequencies.



- Can the targets' speeds be estimated?
- Can the targets' positions be estimated?

Estimation Theory for Linear Models

LINKÖPING UNIVERSITY

Chapter 2 overview

- Linear model $y_k = H_k x + e_k$
- WLS theory
- WLS off-line versus on-line forms
- The fusion formula (and safe fusion)
- Overview of algorithms with examples
- Tricks: transform to linear model and conditionally linear model

Prerequisite Quiz

bitly

`https://bit.ly/4alWxUa`

WLS Summary

Linear model

$$
\begin{aligned}
y_k &= H_k x + e_k, & \operatorname{cov}(e_k) &= R_k, \quad k = 1, \dots, N,\\
\mathbf{y} &= \mathbf{H}x + \mathbf{e}, & \operatorname{cov}(\mathbf{e}) &= \mathbf{R}.
\end{aligned}
$$

WLS loss function

$$
V^{WLS}(x) = \sum_{k=1}^{N} (y_k - H_k x)^T R_k^{-1} (y_k - H_k x) = (\mathbf{y} - \mathbf{H}x)^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{H}x).
$$

Solution in batch form

$$
\begin{aligned}
\hat{x} &= \Big(\sum_{k=1}^{N} H_k^T R_k^{-1} H_k\Big)^{-1} \sum_{k=1}^{N} H_k^T R_k^{-1} y_k &&= (\mathbf{H}^T \mathbf{R}^{-1} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{R}^{-1} \mathbf{y},\\
P &= \Big(\sum_{k=1}^{N} H_k^T R_k^{-1} H_k\Big)^{-1} &&= (\mathbf{H}^T \mathbf{R}^{-1} \mathbf{H})^{-1}.
\end{aligned}
$$

Sequential WLS

The WLS estimate can be computed recursively in the space/time sequence y_k. Suppose the estimate $\hat{x}_{k-1}$ with covariance P_{k-1} is based on observations $y_{1:k-1}$, where we initiate with $\hat{x}_0$ and P_0 (a 'prior'). A new observation is fused using

$$\begin{aligned}\hat{x}_k &= \hat{x}_{k-1} + P_{k-1}H_k^T(H_kP_{k-1}H_k^T + R_k)^{-1}(y_k - H_k\hat{x}_{k-1}),\\ P_k &= P_{k-1} - P_{k-1}H_k^T(H_kP_{k-1}H_k^T + R_k)^{-1}H_kP_{k-1}.\end{aligned}$$

Note that the fusion formula can be used as an alternative. In fact, the derivation is based on the information fusion formula applying the matrix inversion lemma.

The Fusion Formula

The fusion formula for two independent estimates is

$$\left.\begin{aligned} \mathsf{E}(\hat{x}_1) = \mathsf{E}(\hat{x}_2) = x \\ \mathsf{cov}(\hat{x}_1) = P_1 \\ \mathsf{cov}(\hat{x}_2) = P_2 \end{aligned}\right\} \implies \begin{cases} \hat{x} = P(P_1^{-1}\hat{x}_1 + P_2^{-1}\hat{x}_2), \\ P = (P_1^{-1} + P_2^{-1})^{-1}. \end{cases}$$

If the estimates are not independent, the safe fusion (or covariance intersection algorithm) provides a pessimistic lower bound accounting for worst case correlation.

A Simple Sensor Network Example, cont'd (1/3)

Code for triangulation in *Signal and Systems Lab*:

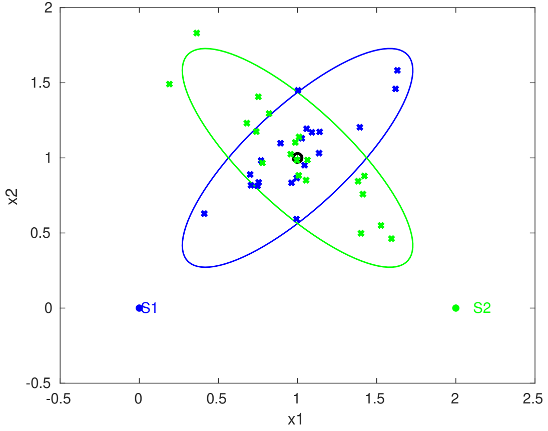
```
p1 = [0; 0];   % S1
p2 = [2; 0];   % S2
x0 = [1; 1];   % Actual state

X1 = ndist(x0, 0.1*[1 0.8; 0.8, 1]);
X2 = ndist(x0, 0.1*[1 -0.8; -0.8, 1]);

x1 = rand(X1, 20);
x2 = rand(X2, 20);

plot2(X1, X2, 'conf', 90,...
      'legend', 'off', 'markersize', 18);
hold on;
plot(x0(1), x0(2), 'ko', 'linewidth', 2);
plot2(empdist(x1), empdist(x2),...
      'legend', 'off', 'linewidth', 2);
axis([-0.5, 2.5, -0.5, 2]);

plot(p1(1), p1(2), '.b',...
     p2(1), p2(2), '.g',...
     'markersize', 15);
text(p1(1)+.01, p1(2), 'S1', 'Color', 'b');
text(p2(1)+.11, p2(2), 'S2', 'Color', 'g');
```



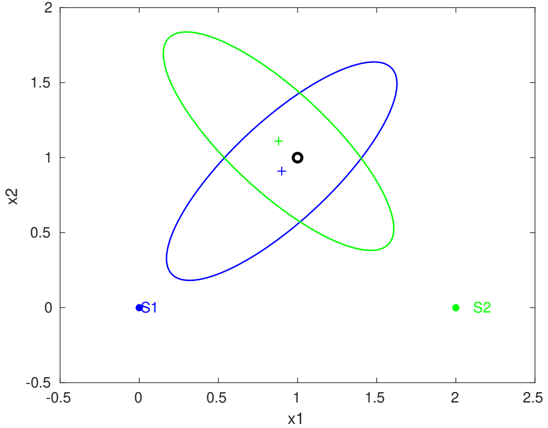
Experiment setup

A Simple Sensor Network Example, cont'd (2/3)

Code for triangulation in *Signal and Systems Lab*:

```
plot2(Xhat1, Xhat2, 'conf', 90,...
      'legend', 'off', 'markersize', 18);
hold on;
plot(x0(1), x0(2), 'ko', 'linewidth', 2);
axis([-0.5, 2.5, -0.5, 2]);

plot(p1(1), p1(2), '.b',...
     p2(1), p2(2), '.g',...
     'markersize', 15);
text(p1(1)+.01, p1(2), 'S1', 'Color', 'b');
text(p2(1)+.11, p2(2), 'S2', 'Color', 'g');
```



Estimates

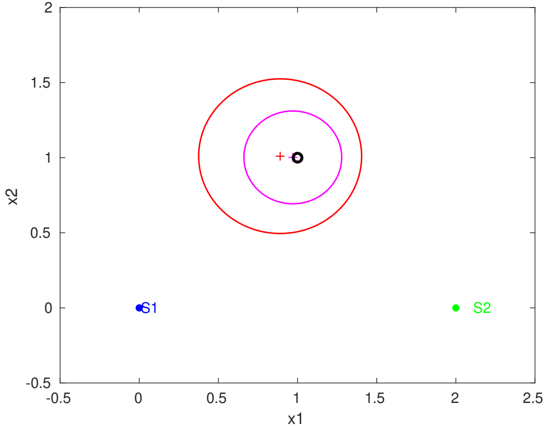
A Simple Sensor Network Example, cont'd (3/3)

Code for triangulation in *Signal and Systems Lab*:

```
Xhat3 = 0.5*Xhat1 + 0.5*Xhat2;  % LS
Xhat4 = fusion(Xhat1, Xhat2);  % WLS

plot2(Xhat3, Xhat4, 'col', 'rm', 'legend', 'off')
hold on;
plot(x0(1), x0(2), 'ko', 'linewidth', 2);
axis([-0.5, 2.5, -0.5, 2]);

plot(p1(1), p1(2), '.b',...
     p2(1), p2(2), '.g',...
     'markersize', 15);
text(p1(1)+.01, p1(2), 'S1', 'Color', 'b');
text(p2(1)+.11, p2(2), 'S2', 'Color', 'g');
```



Combine using LS and WLS

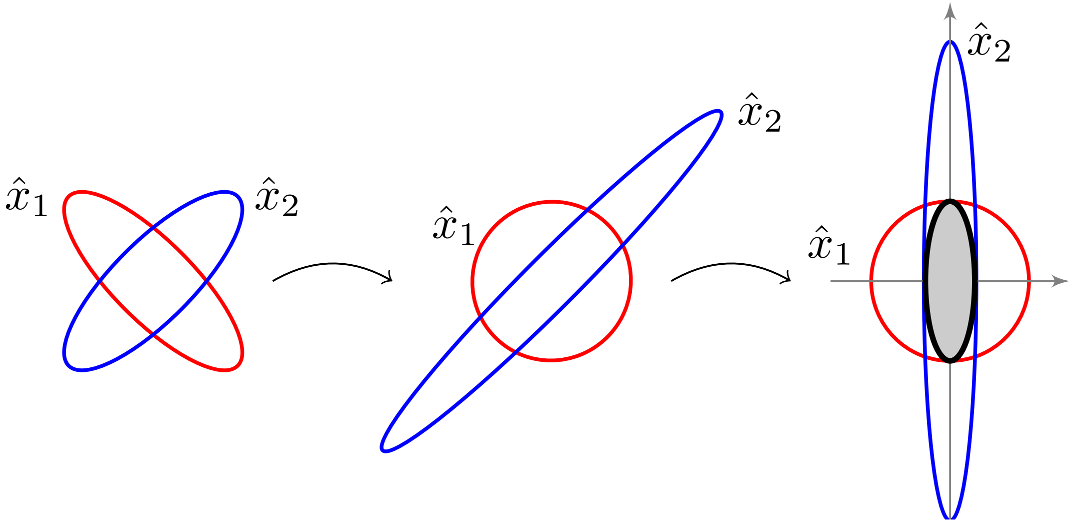
Safe Fusion

Given two unbiased estimates $\hat{x}_1, \hat{x}_2$ with covariance matrices P_1 and P_2, respectively.

Algorithm

1. SVD: $P_1 = U_1 D_1 U_1^T$.
2. SVD: $D_1^{-1/2} U_1^T P_2 U_1 D_1^{-1/2} = U_2 D_2 U_2^T$.
3. Transformation matrix: $T = U_2^T D_1^{-1/2} U_1^T$.
4. State transformation: $\hat{\hat{x}}_1 = T\hat{x}_1$ and $\hat{\hat{x}}_2 = T\hat{x}_2$.
   The covariances of these are $\operatorname{cov}(\hat{\hat{x}}_1) = I$ and $\operatorname{cov}(\hat{\hat{x}}_2) = D_2$.
5. For each component $i = 1, 2, \ldots, n_x$, let
   $$\begin{aligned} \hat{\hat{x}}^i &= \hat{\hat{x}}_1^i, \quad D^{ii} = 1 \qquad \text{if} \quad D_2^{ii} \geq 1, \\ \hat{\hat{x}}^i &= \hat{\hat{x}}_2^i, \quad D^{ii} = D_2^{ii} \quad \text{if} \quad D_2^{ii} < 1. \end{aligned}$$
6. Inverse state transformation:
   $$\hat{x} = T^{-1}\hat{\hat{x}}, \quad P = T^{-1} D T^{-T}$$

Algorithm Illustration

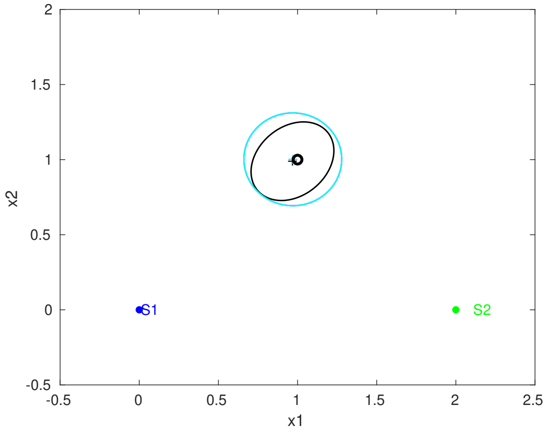


A Simple Sensor Network Example, cont'd

```
Xhat5 = fusion(Xhat1, Xhat4);  % X1 used twice
Xhat6 = safefusion(Xhat1, Xhat4);  % Safe fusion

plot2(Xhat4, Xhat5, Xhat6,...
      'col', 'mkc', 'legend', 'off');
hold on;
plot(x0(1), x0(2), 'ko', 'linewidth', 2);
axis([-0.5, 2.5, -0.5, 2]);

plot(p1(1), p1(2), '.b',...
     p2(1), p2(2), '.g',...
     'markersize', 15);
text(p1(1)+.01, p1(2), 'S1', 'Color', 'b');
text(p2(1)+.11, p2(2), 'S2', 'Color', 'g');
```



Effect of double counting and safe fusion

Summary

LINKÖPING
UNIVERSITY

Summary Lecture 1

- Linear model on batch form:

$$\mathbf{y} = \mathbf{H}x + \mathbf{e}, \quad \text{cov}(\mathbf{e}) = \mathbf{R}.$$

- WLS minimizes the loss function

$$V^{WLS}(x) = (\mathbf{y} - \mathbf{H}x)^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{H}x).$$

- WLS solution

$$\hat{x} = \left(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{y}, \quad P = \left(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\right)^{-1}.$$

- LS special case with $\mathbf{R} \propto I$ and gives larger P.
- The fusion formula for two **independent** estimates is

$$\mathsf{E}(\hat{x}_1) = \mathsf{E}(\hat{x}_2) = x, \quad \text{cov}(\hat{x}_1) = P_1, \quad \text{cov}(\hat{x}_2) = P_2 \Rightarrow$$
$$\hat{x} = P\left(P_1^{-1}\hat{x}_1 + P_2^{-1}\hat{x}_2\right), \quad P = \left(P_1^{-1} + P_2^{-1}\right)^{-1}.$$

- If the estimates are not independent, P is larger than indicated.

Gustaf Hendeby

gustaf.hendeby@liu.se

www.liu.se

LINKÖPING UNIVERSITY