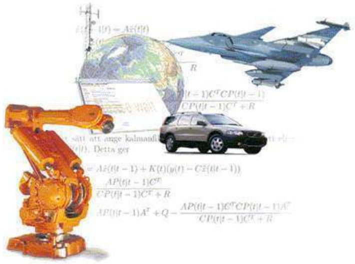
# Part 1 - Introduction and Strategies for state inference



Thomas Schön

Division of Automatic Control
Linköping University
Sweden



## Hinting at the potential – state estimation (I/II)



Fighter aircraft navigation using particle filters together with Saab.

The task is to find the aircraft position using information from several sensors:

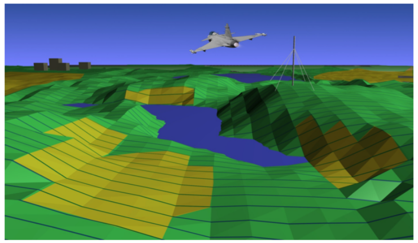
- Inertial sensors
- Radar
- Terrain elevation database

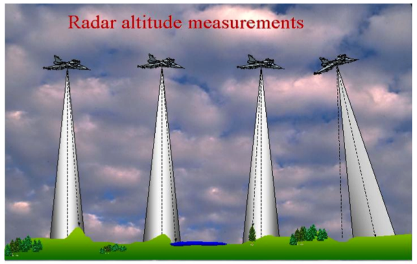
This **sensor fusion** problem requires a nonlinear state estimation problem to be solved, where we want to compute $p(x_t \mid y_{1:t})$.



## Hinting at the potential – state estimation (II/II)








Key theory that allowed us to do this

- Particle filter (Part 3 of this course)
- Rao-Blackwellized particle filter

Details of this particular example (and results using real flight data from Gripen) are provided in

Thomas Schön, Fredrik Gustafsson, and Per-Johan Nordlund. **Marginalized Particle Filters for Mixed Linear/Nonlinear State-Space Models**. *IEEE Transactions on Signal Processing*, 53(7):2279-2289, July 2005.



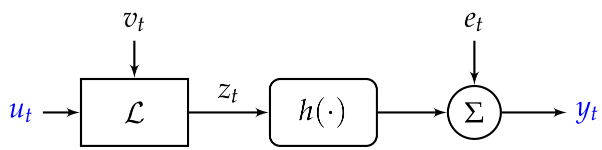
## Hinting at the potential – system identification (I/IV)



The theory provided tomorrow (Part 4) allows us to perform inference in state space models (SSMs)

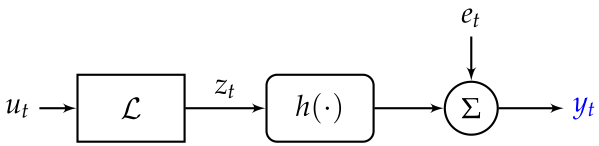
$$x_{t+1} \mid x_t \sim f_\theta(x_{t+1} \mid x_t, u_t) \qquad y_t \mid x_t \sim h_\theta(y_t \mid x_t, u_t)$$

Consider the special case of a Wiener model (a linear Gaussian state space (LGSS) model followed by a static nonlinearity)



## Hinting at the potential – system identification (II/IV)

Consider the blind problem,




The **task** is to learn the parameters of the linear system $\mathcal{L}$ and find the nonlinearity $h(\cdot)$ (entire function has to be learned) based only on the output measurements $y_{1:T} \triangleq \{y_1, \dots, y_T\}$.

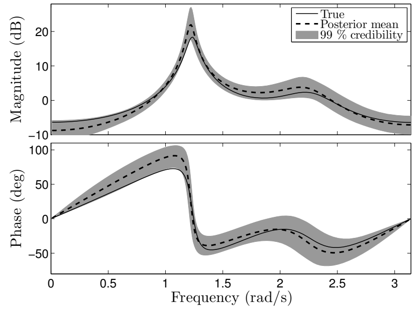
We do not impose any assumption on the nonlinearity and allow for colored noises.

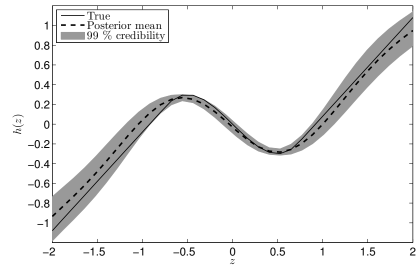




## Hinting at the potential – system identification (III/IV)

Using a PMCMC method (introduced in Part 4) we can compute the posterior distribution $p(\theta \mid y_{1:T})$, where $\theta$ contains the unknown parameters and the unknown measurement function.




**Show movie**





## Hinting at the potential – system identification (IV/IV)

Key theory that allowed us to do this:

- Particle MCMC (Part 4)
- Particle smoothing/ backward simulation (Part 3)
- Gaussian processes (Not covered in this course)

Details of this particular example are available in

Fredrik Lindsten, Thomas B. Schön and Michael I. Jordan, **A semiparametric Bayesian approach to Wiener system identification**. *Proceedings of the 16th IFAC Symposium on System Identification (SYSID)*, Brussels, Belgium, July, 2012.





## Important Message!

*Given the computational tools that we have today it can be rewarding to resist the linear Gaussian convenience!!*

## The aim of this course



The **aim of this course** is to provide an introduction to the theory and application of (new) computational methods for inference in dynamical systems.

The **key computational methods** we refere to are,

- Sequential Monte Carlo (SMC) methods (e.g., particle filters and particle smoothers) for nonlinear state inference problems.
- Expectation maximisation (EM) and Markov chain Monte Carlo (MCMC) methods for nonlinear system identification.

Course home page:

`users.isy.liu.se/rt/schon/course_CIDSusyd.html`



## Outline of the course



Part 1 Modelling and strategies for inferring states and parameters
- a) Modelling dynamical systems using SSMs
- b) Strategies for state inference

Part 2 EM and MCMC introduced by learning LGSS models
- a) Maximum likelihood (ML) learning using Expectation Maximisation (EM)
- b) Bayesian learning using Gibbs sampling (MCMC)

Part 3 Sequential Monte Carlo (SMC)
- a) Basic sampling (rejection sampling, importance sampling)
- b) Particle filter (PF)
- c) Particle smoother (PS)

Part 4 Learning nonlinear dynamical models
- a) Maximum likelihood learning using EM and PS
- b) Bayesian learning using particle MCMC (PMCMC)



## Outline - Part 1



1. Modelling dynamical systems
   - a) Nonlinear state space model (SSM)
   - b) Linear Gaussian state space (LGSS) model
   - c) Conditionally linear Gaussian state space (CLGSS) model
2. Strategies for state inference
   - a) Forward computations
   - b) Backward computations



## 1. Representing an SSM using pdf's



**Definition (State space model (SSM))**

A state space model (SSM) consists of a Markov process $\{x_t\}_{t\geq 1}$ and a measurement process $\{y_t\}_{t\geq 1}$, related according to

$$\begin{aligned} x_{t+1} \mid x_t &\sim f_{\theta,t}(x_{t+1} \mid x_t, u_t), \\ y_t \mid x_t &\sim h_{\theta,t}(y_t \mid x_t, u_t), \\ x_1 &\sim \mu_\theta(x_1), \end{aligned}$$

where $x_t \in \mathbb{R}^{n_x}$ denotes the state, $u_t \in \mathbb{R}^{n_u}$ denotes a known deterministic input signal, $y_t \in \mathbb{R}^{n_y}$ denotes the observed measurement and $\theta \in \Theta \subseteq \mathbb{R}^{n_\theta}$ denotes any unknown (static) parameters.

## 2. Representing SSM using difference equations



In engineering literature, the SSM is often written in terms of a difference equation and an accompanying measurement equation,

$$
\begin{aligned}
x_{t+1} &= \widetilde{f}_{\theta,t}(x_t, u_t) + v_{\theta,t}, \\
y_t &= \widetilde{h}_{\theta,t}(x_t, u_t) + e_{\theta,t},
\end{aligned}
$$



## 3. Representing SSM using a graphical model (I/II)



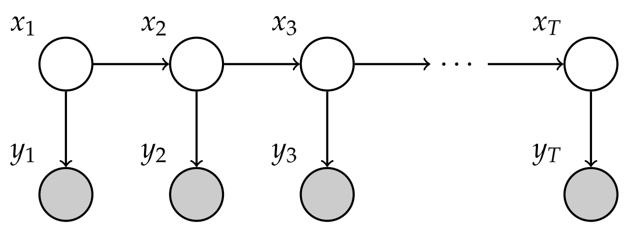



Figure: Graphical model for the SSM. Each stochastic variable is encoded using a node, where the nodes that are filled (gray) corresponds to variables that are observed and nodes that are not filled (white) are latent variables. The arrows pointing to a certain node encodes which variables the corresponding node are conditioned upon.

The SSM is an instance of a graphical model called **Bayesian network**, or **belief network**.



## 3. Representing SSM using a graphical model (II/II)



A Bayesian network directly describes how the joint distribution of all the involved variables (here $p(x_{1:T}, y_{1:T})$) is decomposed into a product of factors,

$$
p(x_{1:T}, y_{1:T}) = \prod_{t=1}^{T} p(x_t \mid \mathrm{pa}(x_t)) \prod_{t=1}^{T} p(y_t \mid \mathrm{pa}(y_t)),
$$

where $\mathrm{pa}(x_t)$ denotes the set of parents to $x_t$.

$$
p(x_{t:T}, y_{1:T}) = \mu(x_1) \prod_{t=1}^{T-1} f_{\theta,t}(x_{t+1} \mid x_t) \prod_{t=1}^{T} h_{\theta,t}(y_t \mid x_t).
$$

Graphical models offers a powerful framework for modeling, inference and learning,

Bishop, C. M. (2006). **Pattern Recognition and Machine Learning**. Springer.
Koller, D. and Friedman, N. (2009). **Probabilistic Graphical Models: Principles and Techniques**. MIT Press.



## The LGSS model



**Definition (Linear Gaussian State Space (LGSS) model)**

The time invariant linear Gaussian state space (LGSS) model is defined by

$$
\begin{aligned}
x_{t+1} &= Ax_t + Bu_t + v_t, \\
y_t &= Cx_t + Du_t + e_t,
\end{aligned}
$$

where $x_t \in \mathbb{R}^{n_x}$ denotes the state, $u_t \in \mathbb{R}^{n_u}$ denotes the known input signal and $y_t \in \mathbb{R}^{n_y}$ denotes the observed measurement. The initial state and the noise are distributed according to

$$
\begin{pmatrix} x_1 \\ v_t \\ e_t \end{pmatrix} \sim \mathcal{N}\left( \begin{pmatrix} \mu \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} P_1 & 0 & 0 \\ 0 & Q & S \\ 0 & S^T & R \end{pmatrix} \right).
$$

## Notation for Gaussian (Normal) variables



The pdf of a Gaussian variable is denoted $\mathcal{N}(x \mid \mu, \Sigma)$, i.e.,

$$\mathcal{N}(x \mid \mu, \Sigma) \triangleq \frac{1}{(2\pi)^{n/2}\sqrt{\det \Sigma}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

---

See the appendix of the lecture notes for basic theorems needed in manipulating Gaussian variables.



## CLGSS model



**Definition (Conditionally linear Gaussian state space (CLGSS) model)**

Assume that the state $x_t$ of an SSM can be partitioned according to $x_t = \begin{pmatrix} s_t^\mathsf{T} & z_t^\mathsf{T} \end{pmatrix}^\mathsf{T}$. The SSM is then a CLGSS model if the conditional process $\{z_t \mid s_{1:t}\}_{t \geq 1}$ is described by an LGSS model.

Conditioned on part of the state vector, the rest of the state behaves like an LGSS model.

This can be exploited in deriving inference algorithms!

The $z_t$-process is conditionally linear, motivating the name *linear state* for $z_t$ and *nonlinear state* for $s_t$.



## SLGSS model (I/II)



**Definition (Switching linear Gaussian state space (SLGSS))**

The SLGSS model is defined according to

$$\begin{aligned} z_{t+1} &= A^{s_t} z_t + B^{s_t} u_t + v^{s_t}, \\ y_t &= C^{s_t} z_t + D^{s_t} u_t + e^{s_t}, \\ s_t &\sim p(s_t \mid s_{t-1}, z_{t-1}), \end{aligned}$$

where $z_t \in \mathbb{R}^{n_x}$ denotes the state, $s_t \in \{1, \ldots, S\}$ denotes the switching variable, $u_t \in \mathbb{R}^{n_u}$ denotes the known input signal and $y_t \in \mathbb{R}^{n_y}$ denotes the observed measurement. The initial state $x_1$ and the noise are distributed according to

$$\begin{pmatrix} x_1 \\ v^{s_t} \\ e^{s_t} \end{pmatrix} \sim \mathcal{N}\left( \begin{pmatrix} \mu \\ \bar{v}^{s_t} \\ \bar{e}^{s_t} \end{pmatrix}, \begin{pmatrix} P_1 & 0 & 0 \\ 0 & Q^{s_t} & S^{s_t} \\ 0 & (S^{s_t})^\mathsf{T} & R^{s_t} \end{pmatrix} \right).$$



## SLGSS model (II/II)



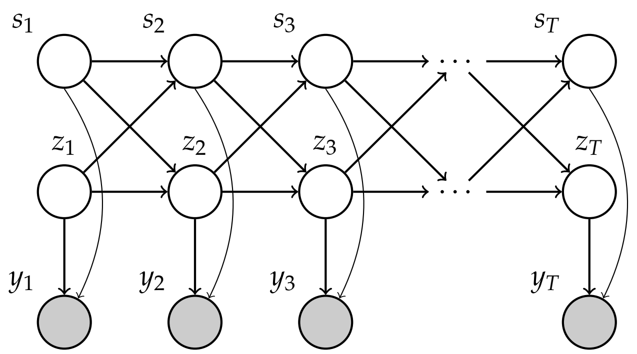



Figure: Graphical model for the switching linear Gaussian state space (SLGSS) model.

# Mixed Gaussian state space (MGSS) model

**Definition (Mixed Gaussian state space (MGSS) model)**

The MGSS model is defined according to

$$x_{t+1} = f_t(s_t) + A_t(s_t)z_t + v_t(s_t),$$
$$y_t = h_t(s_t) + C_t(s_t)z_t + e_t(s_t),$$

where

$$x_t = \begin{pmatrix} s_t \\ z_t \end{pmatrix}, \qquad f_t(s_t) = \begin{pmatrix} f_t^s(s_t) \\ f_t^z(s_t) \end{pmatrix}, \qquad A_t(s_t) = \begin{pmatrix} A_t^s(s_t) \\ A_t^z(s_t) \end{pmatrix}.$$

The noises are distributed according to

$$v_t(s_t) \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} Q^s(s_t) & Q^{sz}(s_t) \\ Q^{sz}(s_t)^{\mathsf{T}} & Q^z(s_t) \end{pmatrix}\right) = \mathcal{N}(0, Q(s_t))$$
$$e_t(s_t) \sim \mathcal{N}(0, R(s_t)).$$





# State inference

Table: Probability density functions for the most commonly encountered state inference problems (filtering, prediction and smoothing).

| **Name** | **Probability density function** |
|---|---|
| Filtering | $p(x_t \mid y_{1:t})$ |
| Prediction | $p(x_{t+1} \mid y_{1:t})$ |
| $k$-step prediction | $p(x_{t+k} \mid y_{1:t})$ |
| Joint smoothing | $p(x_{1:T} \mid y_{1:T})$ |
| Marginal smoothing | $p(x_t \mid y_{1:T}), t \leq T$ |
| Fixed-lag smoothing | $p(x_{t-l+1:t} \mid y_{1:t}), l > 0$ |
| Fixed-interval smoothing | $p(x_{r:t} \mid y_{1:T}), r < t \leq T$ |





# Forward computations

Summarizing this development, we have **measurement update**

$$p(x_t \mid y_{1:t}) = \frac{\overbrace{h(y_t \mid x_t)}^{\text{measurement}} \overbrace{p(x_t \mid y_{1:t-1})}^{\text{prediction pdf}}}{p(y_t \mid y_{1:t-1})},$$

and **time update**

$$p(x_t \mid y_{1:t-1}) = \int \underbrace{f(x_t \mid x_{t-1})}_{\text{dynamics}} \underbrace{p(x_{t-1} \mid y_{1:t-1})}_{\text{filtering pdf}} \mathrm{d}x_{t-1},$$





# Backward computations

By marginalizing

$$p(x_{1:T} \mid y_{1:T}) = p(x_T \mid y_{1:T}) \prod_{t=1}^{T-1} \frac{f(x_{t+1} \mid x_t) p(x_t \mid y_{1:t})}{p(x_{t+1} \mid y_{1:t})}.$$

w. r. t. $x_{1:t-1}$ and $x_{t+1:T}$ we obtain the following expression for the marginal smoothing pdf

$$p(x_t \mid y_{1:T}) = p(x_t \mid y_{1:t}) \int \frac{f(x_{t+1} \mid x_t) p(x_{t+1} \mid y_{1:T})}{p(x_{t+1} \mid y_{1:t})} \mathrm{d}x_{t+1}.$$