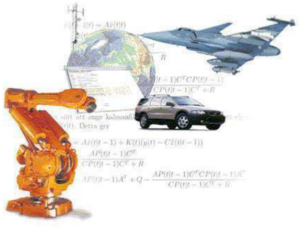
Nonlinear system identification enabled via sequential Monte Carlo



Thomas Schön (with significant input from Fredrik Lindsten)

Division of Automatic Control
Linköping University
Sweden

Joint work with (alphabetical order): **Michael I. Jordan** (UC Berkeley), **Fredrik Lindsten** (Linköping University), **Lennart Ljung** (Linköping University), **Brett Ninness** (University of Newcastle, Australia) and **Adrian Wills** (MRA, Newcastle, Australia).

The problem – identification of state space models

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

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

We observe

$$y_{1:T} \triangleq \{y_1, \ldots, y_T\},$$

(leaving the latent variables $x_{1:T}$ unobserved).

Identification problem: Find θ based on $y_{1:T}$.

Strategies for inference in latent variable models

Alternate between updating θ and updating $x_{1:T}$.

Frequentists:

- Find $\widehat{\theta}_{\text{ML}} = \arg\max_{\theta} p_{\theta}(y_{1:T})$.
- Use e.g. the expectation maximization (EM) algorithm.

Bayesians:

- Find $p(\theta \mid y_{1:T})$.
- Use e.g. Gibbs sampling.

Outline

1. Maximum Likelihood (ML) identification
    - Problem formulation
    - Solution using EM and a particle smoother
2. Bayesian identification
    - Problem formulation
    - Gibbs sampling
3. Sequential Monte Carlo (SMC), the particle filter
4. Particle Gibbs with ancestor sampling (PG-AS)

The sequential Monte Carlo samplers are fundamental to **both** the ML and the Bayesian approaches.

Problem formulation – ML

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

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

Identification problem: Find θ based on $y_{1:T}$.

ML amounts to solving,

$$\widehat{\theta}^{\text{ML}} = \arg\max_{\theta} \; \log p_\theta(y_{1:T})$$

where the log-likelihood function is given by

$$\log p_\theta(y_{1:T}) = \sum_{t=1}^{T} \log p_\theta(y_t \mid y_{1:t-1})$$

Expectation Maximization (EM) – Strategy in SSMs

The **EM** algorithm computes ML estimates of unknown parameters in probabilistic models involving latent variables.

The latent variables in an SSM are given by the states,

$$\{x_1, \ldots, x_T\}.$$

Strategy: Use the *structure* inherent in the SSM to separate the original problem into *two closely linked subproblems*, each of which is hopefully in some sense more tractable than the original problem.

EM – the algorithm

Algorithm 1 EM for identifying nonlinear dynamical systems

```
1. Initialise: Set $i = 1$ and choose an initial $\theta^1$.
2. While not converged do:
   (a) Expectation (E) step: Compute
       $Q(\theta, \theta^i) = \mathrm{E}_{\theta^i} [\log p_\theta(x_{1:T}, y_{1:T}) \mid y_{1:T}]$
       $= \int \log p_\theta(x_{1:T}, y_{1:T}) \underbrace{p_{\theta^i}(x_{1:T} \mid y_{1:T})} \mathrm{d}x_{1:T}$
       using PS (forward filter/backward simulation, FFBS).
   (b) Maximization (M) step: Compute $\theta^{i+1} = \arg\max_{\theta \in \Theta} Q(\theta, \theta^i)$
   (c) $i \leftarrow i + 1$
```

Thomas B. Schön, Adrian Wills and Brett Ninness. **System Identification of Nonlinear State-Space Models**. *Automatica*, 47(1):39-49, January 2011.

Problem formulation – Bayesian

Consider a Bayesian SSM (θ is now a random variable with a prior density $p(\theta)$)

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

Identification problem: Compute the posterior $p(\theta, x_{1:T} \mid y_{1:T})$, or one of its marginals.

The **key challenge** is that there is no closed form expression available for the posterior.

Gibbs sampler for SSMs

Markov chain Monte Carlo (MCMC) methods allow us to generate samples from a target distribution by simulating a Markov chain.

Gibbs sampling (blocked) for SSMs amounts to iterating

- Draw $\theta[m] \sim p(\theta \mid x_{1:T}[m-1], y_{1:T})$,
- Draw $x_{1:T}[m] \sim p(x_{1:T} \mid \theta[m], y_{1:T})$.

The above procedure results in a Markov chain,

$$\{\theta[m], x_{1:T}[m]\}_{m \geq 1}$$

with $p(\theta, x_{1:T} \mid y_T)$ as its stationary distribution!

Gibbs sampler for a general SSM

What would a Gibbs sampler for a general nonlinear/non-Gaussian SSM look like?

- Draw $\theta[m] \sim p(\theta \mid x_{1:T}[m-1], y_{1:T})$; **OK!**
- Draw $x_{1:T}[m] \sim p(x_{1:T} \mid \theta[m], y_{1:T})$. **Hard!**

Problem: $p(x_{1:T} \mid \theta[m], y_{1:T})$ not available!

Idea: Approximate $p(x_{1:T} \mid \theta[m], y_{1:T})$ using a sequential Monte Carlo method!

Outline

1. Maximum Likelihood (ML) identification
    - Problem formulation
    - Solution using EM and a particle smoother
2. Bayesian identification
    - Problem formulation
    - Gibbs sampling
3. **Sequential Monte Carlo (SMC), the particle filter**
4. Particle Gibbs with ancestor sampling (PG-AS)

The sequential Monte Carlo samplers are fundamental to **both** the ML and the Bayesian approaches.

The particle filter

The particle filter provides an approximation of the filter PDF $p(x_t \mid y_{1:t})$, when the state evolves according to an SSM,

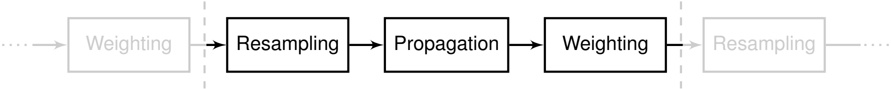
$$\begin{aligned} x_{t+1} \mid x_t &\sim f_t(x_{t+1} \mid x_t), \\ y_t \mid x_t &\sim g_t(y_t \mid x_t), \\ x_1 &\sim \mu(x_1). \end{aligned}$$

The particle filter maintains an empirical distribution made up N samples (particles) and corresponding weights

$$\widehat{p}^N(x_t \mid y_{1:t}) = \sum_{i=1}^{N} w_t^i \delta_{x_t^i}(x_t).$$

"Think of each particle as one simulation of the system state. Only keep the good ones."

The particle filter

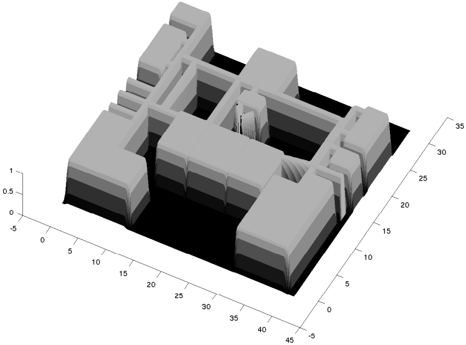


1. **Resampling:** $\{x^i_{1:t-1}, w^i_{t-1}\}_{i=1}^N \to \{\tilde{x}^i_{1:t-1}, 1/N\}_{i=1}^N$.
2. **Propagation:** $x^i_t \sim R_t(x_t \mid \tilde{x}^i_{1:t-1})$ and $x^i_{1:t} = \{\tilde{x}^i_{1:t-1}, x^i_t\}$.
3. **Weighting:** $w^i_t = W_t(x^i_{1:t})$.

The result is a new weighted set of particles $\{x^i_{1:t}, w^i_t\}_{i=1}^N$.

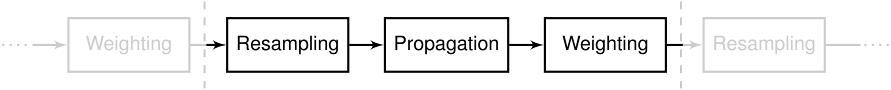
Example – Indoor localization

Aim: Compute the position of a person moving around indoors using sensors (inertial, magnetometer and radio) located in an ID badge and a map.



Show movie

The particle filter – alternative formulation



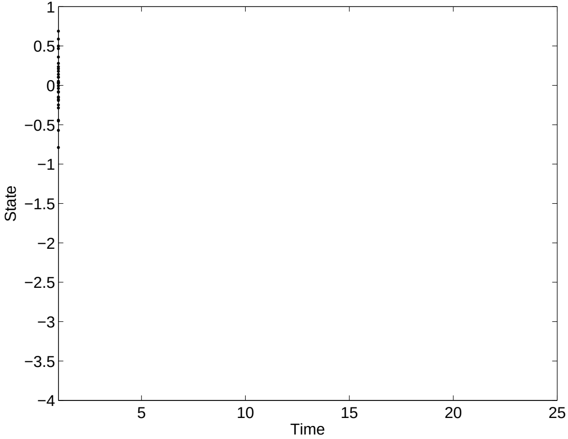
1. **Resampling + Propagation:**

$$(a_t^i, x_t^i) \sim M_t(a_t, x_t) = \frac{w_{t-1}^{a_t}}{\sum_l w_{t-1}^l} R_t(x_t \mid x_{1:t-1}^{a_t}).$$

2. **Weighting:** $w_t^i = W_t(x_{1:t}^i)$.

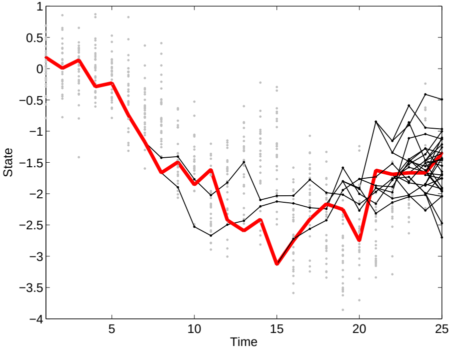
The result is a new weighted set of particles $\{x_{1:t}^i, w_t^i\}_{i=1}^N$.

The particle filter – illustrating particle degeneracy



Sampling based on the PF

With $P(x'_{1:T} = x^i_{1:T}) \propto w^i_T$ we get, $x'_{1:T} \overset{\text{approx.}}{\sim} p(x_{1:T} \mid \theta, y_{1:T})$.



Problems

Problems with this approach,

- Based on a PF ⇒ approximate sample.
- Does not leave $p(\theta, x_{1:T} \mid y_{1:T})$ invariant!
- Relies on large N to be successful.
- A lot of wasted computations.

To get around these problems,

Use a conditional particle filter (CPF). One prespecified path is retained throughout the sampler.

Christophe Andrieu, Arnaud Doucet and Roman Holenstein, **Particle Markov chain Monte Carlo methods**, *Journal of the Royal Statistical Society: Series B*, 72:269-342, 2010.

Particle Markov Chain Monte Carlo (PMCMC)

The idea underlying **PMCMC** is to make use of a certain SMC sampler to construct a Markov kernel leaving the joint smoothing distribution $p(x_{1:T} \mid \theta, y_{1:T})$ invariant.

This Markov kernel is then used in a standard MCMC algorithm (e.g. Gibbs, results in the **Particle Gibbs (PG)**).

Three SMC samplers leaving $p(x_{1:T} \mid \theta, y_{1:T})$ invariant:

1. Conditional particle filter (CPF)

   Christophe Andrieu, Arnaud Doucet and Roman Holenstein, **Particle Markov chain Monte Carlo methods**, *Journal of the Royal Statistical Society: Series B*, 72:269-342, 2010.

2. CPF with backward simulation (CPF-BS)

   Fredrik Lindsten and Thomas B. Schn. **On the use of backward simulation in the particle Gibbs sampler**. *Proc. of the 37th Internat. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, Kyoto, Japan, March 2012.

3. **CPF with ancestor sampling (CPF-AS)**

   Fredrik Lindsten, Michael I. Jordan and Thomas B. Schön, **Ancestor sampling for particle Gibbs**, *Proceedings of Neural Information Processing Systems (NIPS)*, Lake Tahoe, NV, US, December, 2012.

Conditional PF with ancestor sampling

Algorithm CPF w. ancestor sampling (CPF-AS), conditioned on $x^\star_{1:T}$

```
1. Initialize ($t = 1$):
    (a) Draw $x_1^i \sim R_1^\theta(x_1)$ for $i \neq N$ and set $x_1^N = x_1^\star$.
    (b) Set $w_1^i = W_1^\theta(x_1^i)$ for $i = 1, \dots, N$.
2. for $t = 2, \dots, T$:
    (a) Draw $(a_t^i, x_t^i) \sim M_t^\theta(a_t, x_t)$ for $i \neq N$ and set $x_t^N = x_t^\star$.
    (b) Draw $a_t^N$ with $P(a_t^N = i) \propto w_{t-1}^i p(x_t^\star \mid \theta, x_{t-1}^i)$.
    (c) Set $x_{1:t}^i = \{x_{1:t-1}^{a_t^i}, x_t^i\}$ and $w_t^i = W_t^\theta(x_{1:t}^i)$ for $i = 1, \dots, N$.
```

(The red text highlights the difference to the standard PF)

Conditional PF with ancestor sampling

Theorem

For any $N \geq 2$, the procedure;

(i) *Run CPF-AS$(x^{\star}_{1:T})$;*

(ii) *Sample $P(x'_{1:T} = x^{i}_{1:T}) \propto w^{i}_{T}$;*

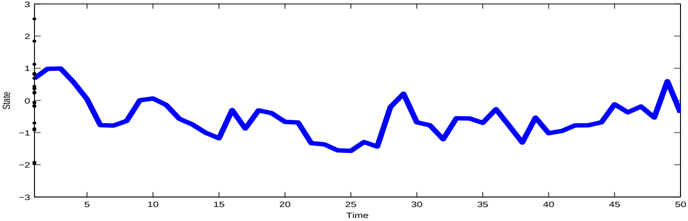
defines a Markov kernel on X^T which leaves $p(x_{1:T} \mid \theta, y_{1:T})$ invariant.

Three additional reasons for using CPF-AS:

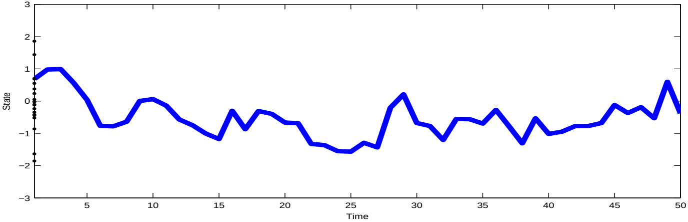
1. Significantly improves the mixing compared to CPF.
2. The computational complexity is linear in N.
3. Opens up for non-Markovian models.

CPF vs. CPF-AS

CPF



CPF-AS



Particle Gibbs with ancestor sampling

Bayesian identification: Gibbs + CPF-AS = PG-AS

Algorithm PG-AS: Particle Gibbs with ancestor sampling

1. **Initialize:** Set $\{\theta[0], x_{1:T}[0]\}$ arbitrarily.
2. **For** $m \geq 1$**, iterate:**
   (a) Draw $\theta[m] \sim p(\theta \mid x_{1:T}[m-1], y_{1:T})$.
   (b) Run CPF-AS$(x_{1:T}[m-1])$, targeting $p(x_{1:T} \mid \theta[m], y_{1:T})$.
   (c) Sample with $P(x_{1:T}[m] = {x_{1:T}}^i) \propto w_T^i$.

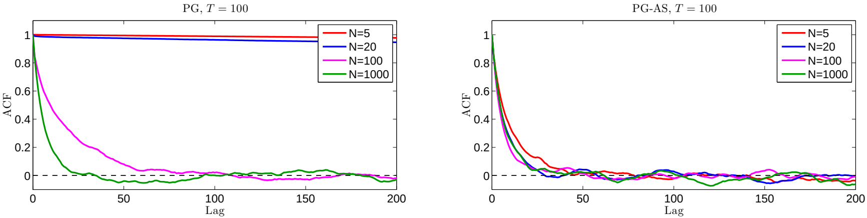
For any number of particles $N \geq 2$, the Markov chain $\{\theta[m], x_{1:T}[m]\}_{m\geq 1}$ has stationary distribution $p(\theta, x_{1:T} \mid y_{1:T})$.

Toy example – stochastic volatility (I/II)

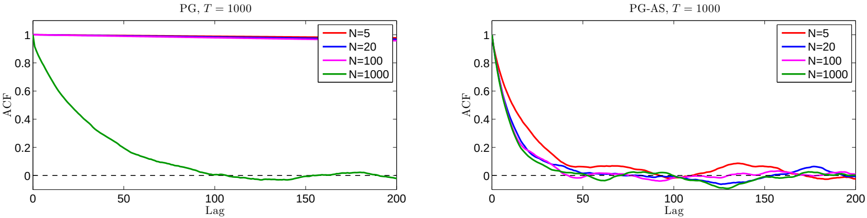
Consider the stochastic volatility model,

$$x_{t+1} = 0.9x_t + w_t, \qquad w_t \sim \mathcal{N}(0, \theta),$$
$$y_t = e_t \exp\left(\frac{1}{2}x_t\right), \qquad e_t \sim \mathcal{N}(0, 1).$$

Let us study the ACF for the estimation error, $\widehat{\theta} - \mathrm{E}\left[\theta \mid y_{1:T}\right]$



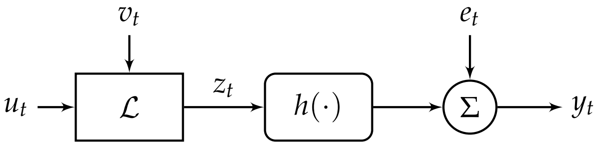
Toy example – stochastic volatility (II/II)



Some observations:

- We want the ACF to decay to zero as rapidly as possible (indicates good mixing in the PG sampler).
- Note the superior mixing of PG-AS compared to PG-CPF (already for just $N = 5$ particles!).

Example – semiparametric Wiener model (I/III)



Parametric LGSS and a nonparametric static nonlinearity:

$$x_{t+1} = \underbrace{\begin{pmatrix} A & B \end{pmatrix}}_{\Gamma} \begin{pmatrix} x_t \\ u_t \end{pmatrix} + v_t, \qquad v_t \sim \mathcal{N}(0, Q),$$

$$z_t = Cx_t.$$

$$y_t = h(z_t) + e_t, \qquad e_t \sim \mathcal{N}(0, R).$$

Example – semiparametric Wiener model (II/III)

Everything is learned from the data, by introducing the possibility to switch specific model components on and off.

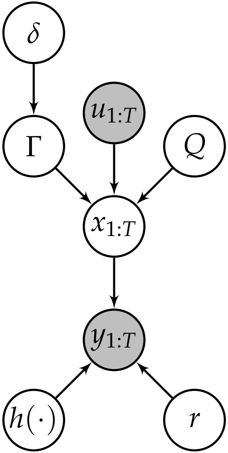
"Parameters": $\theta = \{A, B, Q, \delta, h(\cdot), r\}$.

Bayesian model specified by priors

- Sparseness prior (ARD) on $\Gamma = [A\ B]$,
- Inverse-Wishart prior on Q and r
- Gaussian process prior on $h(\cdot)$,

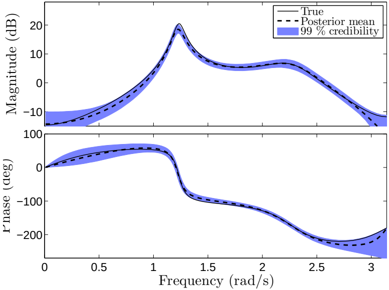
$$h(\cdot) \sim \mathcal{GP}(z, k(z, z')).$$

Inference using PG-AS with $N = 15$ particles. $T = 1\,000$ measurements. We ran $15\,000$ MCMC iterations and discarded $5\,000$ as burn-in.

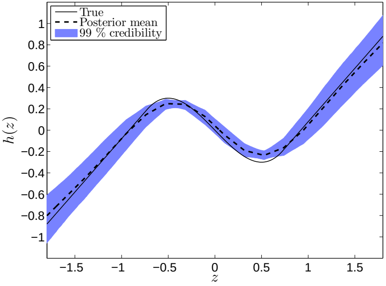


Example – semiparametric Wiener model (III/III)

Show movie



Bode diagram of the 4th-order linear system. Estimated mean (dashed black), true (solid black) and 99% credibility intervals (blue).



Static nonlinearity (non-monotonic), estimated mean (dashed black), true (black) and the 99% credibility intervals (blue).

Fredrik Lindsten, Thomas B. Schön and Michael I. Jordan. **Bayesian semiparametric Wiener system identification**. *Automatica*, 49(7): 2053-2063, July 2013.

Stochastic approximation EM

Assume for the time being that we can sample from $p_{\theta}(x_{1:T} \mid y_{1:T})$.

Stochastic approximation EM (SAEM): Replace the E-step with,

$$\widehat{Q}_m(\theta) = \widehat{Q}_{m-1}(\theta) + \gamma_m \left(\frac{1}{M} \sum_{j=1}^{M} \log p_{\theta}(\tilde{x}_{1:T}^{j}, y_{1:T}) - \widehat{Q}_{m-1}(\theta) \right),$$

where $\tilde{x}_{1:T}^{j} \overset{\text{i.i.d.}}{\sim} p_{\theta}(x_{1:T} \mid y_{1:T})$ for $j = 1, \dots, M$.

SAEM converges to a maximum of $p_{\theta}(y_{1:T})$ for any $M \geq 1$ under standard stochastic approximation conditions.

B. Delyon, M. Lavielle and E. Moulines, **Convergence of a stochastic approximation version of the EM algorithm**, *The Annals of Statistics*, 27:94-128, 1999.

Stochastic approximaiton EM using CPF-AS

- **Bad news:** We cannot sample from $p_\theta(x_{1:T} \mid y_{1:T})$.
- **Good news:** It is enough to sample from a uniformly ergodic Markov kernel, leaving $p_\theta(x_{1:T} \mid y_{1:T})$ invariant.

We can use CPF-AS to sample the states!

Results in an interesting and useful **combination** of frequentist and Bayesian ideas. We will see more combinations like this in the future.

Fredrik Lindsten. **An efficient stochastic approximation EM algorithm using conditional particle filters**. *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013.

Conclusions

- EM-PS for ML learning in nonlinear SSMs.
- Conditional particle filters (CPF) are useful for identification!
- CPF-AS defines a kernel on X^T leaving $p_\theta(x_{1:T} \mid y_{1:T})$ invariant.
- CPF-AS consists of two parts:
    - **Conditioning:** Ensures correct stationary distribution for any N.
    - **Ancestor sampling:** Mitigates path degeneracy and enables movement around the conditioned path.
- PG-AS for Bayesian inference and SAEM-AS for maximum likelihood inference. Both work with very few particles!

- We are working on a book project,

  Thomas B. Schön and Fredrik Lindsten, **Computational learning in dynamical systems**, 2013.

  Send me an e-mail if you are interested in a draft.
- Course: `users.isy.liu.se/rt/schon/course_CIDS.html`

Some references

Forthcoming book

Thomas B. Schön and Fredrik Lindsten, **Computational learning in dynamical systems**, 2013.

Novel introduction of PMCMC (very nice paper!)

Christophe Andrieu, Arnaud Doucet and Roman Holenstein, **Particle Markov chain Monte Carlo methods**, *Journal of the Royal Statistical Society: Series B*, 72:269-342, 2010.

Self-contained introduction to BS and AS (not limited to SSMs)

Fredrik Lindsten and Thomas B. Schön, **Backward simulation methods for Monte Carlo statistical inference**, *Foundations and Trends in Machine Learning*, 2013 (accepted for publication).

PG-AS (and the Wiener identification example)

Fredrik Lindsten, Michael I. Jordan and Thomas B. Schön, **Ancestor sampling for particle Gibbs**, *Proceedings of Neural Information Processing Systems (NIPS)*, Lake Tahoe, NV, US, December, 2012.

Fredrik Lindsten, Thomas B. Schön and Michael I. Jordan. **Bayesian semiparametric Wiener system identification**. *Automatica*, 2013, 49(): 2053-2063.

ML identification of nonlinear SSMs (and Wiener example)

Thomas B. Schön, Adrian Wills and Brett Ninness. **System Identification of Nonlinear State-Space Models**. *Automatica*, 47(1):39-49, January 2011.

Adrian Wills, Thomas B. Schön, Lennart Ljung and Brett Ninness. **Identification of Hammerstein-Wiener Models**. *Automatica*, 49(1): 70-81, January 2013.

MATLAB code is available from our web-site.