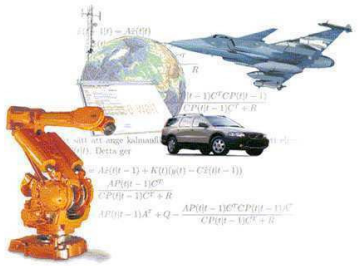
Part 4 - Nonlinear system identification



Thomas Schön

Division of Automatic Control
Linköping University
Sweden

The aim – Part 4

The **aim in part 4** is to show how EM together with SMC and MCMC together with SMC can be used to solve challenging nonlinear system identification problems.

In other words, we will here make use of most of the building blocks introduced throughout the course in order to show how they can be combined to solve nonlinear system identification problems.

Outline

1. Computing ML estimates using EM and PS.
   a) Derive the algorithm
   b) Example - parametric Wiener model
2. Computing Bayesian estimates using particle MCMC (PMCMC)
   a) Particle Metropolis Hastings (PMH)
   b) Example - semiparametric Wiener model

ML problem formulation

Task: Compute the ML estimate of the parameters θ in the SSM

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

The ML estimate is obtained by solving the following optimisation problem,

$$\widehat{\theta}^{\mathrm{ML}} = \arg\max_\theta \; L_\theta(y_{1:N}),$$

where the log-likelihood function is given by

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

EM revisited

The expectation maximisation (EM) algorithm computes ML estimates of unknown parameters in probabilistic models involving latent variables.

Algorithm 1 Expectation Maximization (EM)

1. **Initialise:** Set $i = 1$ and choose an initial θ^1.
2. **While** not converged **do:**
   (a) **Expectation (E) step:** Compute

$$\mathcal{Q}(\theta, \theta^i) = \mathrm{E}_{\theta^i}\left[\log p_\theta(Z, Y) \mid Y\right] = \int \log p_\theta(Z, Y) p_{\theta^i}(Z \mid Y) \mathrm{d}Z$$

   (b) **Maximization (M) step:** Compute $\theta^{i+1} = \arg\max_{\theta \in \Theta} \mathcal{Q}(\theta, \theta^i)$

   (c) $i \leftarrow i + 1$

EM for nonlinear system identification

The **key property** rendering EM an appealing approach for computing maximum likelihood estimates in nonlinear SSMs is that the intermediate quantity $\mathcal{Q}(\theta, \theta^i)$ and its derivatives can be approximated arbitrarily well using particle smoothers.

EM provides a **strategy** for breaking down the problem into two manageable subproblems

1. A nonlinear state smoothing problem
2. A nonlinear optimisation problem

each of which can be handled using readily available algorithms.

Approximation of the $\mathcal{Q}$-function

The intermediate quantity $\mathcal{Q}(\theta, \theta^i)$ is approximated according to (using the particle smoother (FFBSi))

$$\widehat{\mathcal{Q}}(\theta, \theta^i) = \widehat{I}_1(\theta, \theta^i) + \widehat{I}_2(\theta, \theta^i) + \widehat{I}_3(\theta, \theta^i),$$

where

$$\widehat{I}_1(\theta, \theta^i) = \frac{1}{N} \sum_{i=1}^{N} \log \mu_\theta(x_1^i),$$

$$\widehat{I}_2(\theta, \theta^i) = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T-1} \log f_\theta(x_{t+1}^i \mid x_t^i),$$

$$\widehat{I}_3(\theta, \theta^i) = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \log h_\theta(y_t \mid x_t^i).$$

Maximisation (M) step

Use a numerical nonlinear optimisation algorithm, e.g., BFGS. The gradient is computed according to

$$\nabla_\theta \mathcal{Q}(\theta, \theta^i) = \nabla_\theta I_1(\theta, \theta^i) + \nabla_\theta I_2(\theta, \theta^i) + \nabla_\theta I_3(\theta, \theta^i),$$

and based on $\widehat{\mathcal{Q}}(\theta, \theta^i)$ it is straightforward to approximate these gradients according to,

$$\nabla_\theta I_1(\theta, \theta^i) \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \log \mu_\theta(x_1^i),$$

$$\nabla_\theta I_2(\theta, \theta^i) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T-1} \nabla_\theta \log f_\theta(x_{t+1}^i \mid x_t^i),$$

$$\nabla_\theta I_3(\theta, \theta^i) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log h_\theta(y_t \mid x_t^i).$$

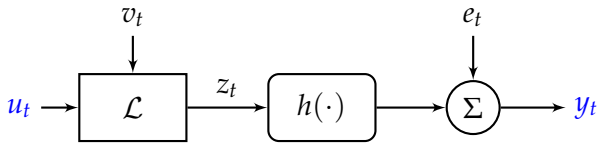
EM for nonlinear system identification

Algorithm 2 EM for nonlinear system identification

1. **Initialise:** Set $i = 1$ and choose an initial θ^1.
2. **While** not converged **do:**
   (a) **Expectation (E) step:** Run a PF and a FFBSi PS and compute
   $$\widehat{\mathcal{Q}}(\theta, \theta^i) = \widehat{I}_1(\theta, \theta^i) + \widehat{I}_2(\theta, \theta^i) + \widehat{I}_3(\theta, \theta^i)$$
   (b) **Maximization (M) step:** Compute $\theta^{i+1} = \arg\max_{\theta \in \Theta} \mathcal{Q}(\theta, \theta^i)$ using an off-the-shelf numerical optimisation algorithm.
   (c) $i \leftarrow i + 1$

A Specific Example – The Wiener Problem (I/II)

As an example we will study how to learn a **Wiener model**.



A Wiener model is a linear dynamical model ($\mathcal{L}$) followed by a static nonlinearity ($h(\cdot)$).

Learning problem: Find $\mathcal{L}$ and $h(\cdot)$ based on $\{u_{1:T}, y_{1:T}\}$.

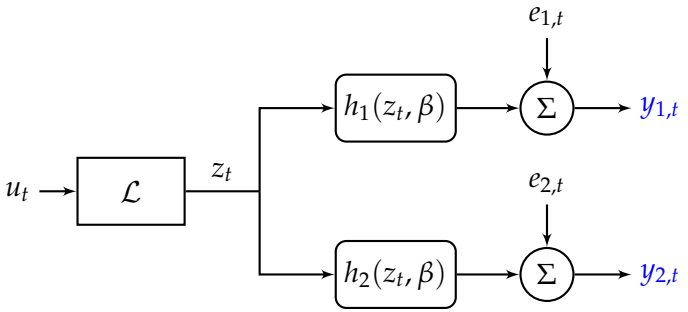
A Specific Example – The Wiener Problem (II/II)

Most of the existing work deals with special cases of the general problem. Typical restrictions imposed are:

- The nonlinearity $h(\cdot)$ is assumed to be invertible.
- The measurement noise e_t is absent.
- The LGSS model is deterministic (v_t is absent).
- The LGSS model is stochastic, but v_t is assumed white.

Using EM + PS we do not have to impose any of these assumptions.

Example – blind Wiener learning



$$x_{t+1} = \begin{pmatrix} A & B \end{pmatrix} \begin{pmatrix} x_t \\ u_t \end{pmatrix}, \qquad u_t \sim \mathcal{N}(0, Q),$$
$$z_t = Cx_t, \qquad y_t = h(z_t, \beta) + e_t, \qquad e_t \sim \mathcal{N}(0, R).$$

Learning problem: Find $\mathcal{L}$ and β, r_1, r_2 based on $\{y_{1,1:T}, y_{2,1:T}\}$.

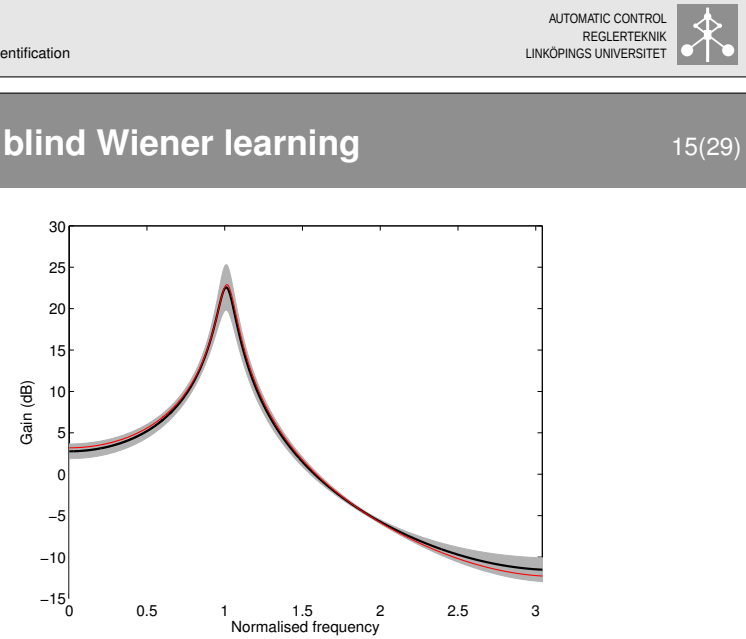
Example – blind Wiener learning

The linear system ($\mathcal{L}$) is given by

$$x_{t+1} = \begin{pmatrix} 1 & -0.9 \\ 1 & 0 \end{pmatrix} x_t + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u_t,$$
$$z_t = \begin{pmatrix} 1 & 0.3 \end{pmatrix} x_t.$$

Complex poles implies a resonant system. The nonlinearities are a saturation and a dead zone, respectively,

$$h_1(z_t, \beta) = \begin{cases} \beta_1 & : z_t < \beta_1 \\ z_t & : \beta_1 \leq z_t \leq \beta_2 \\ \beta_2 & : z_t > \beta_2 \end{cases} \quad h_2(z_t, \beta) = \begin{cases} z_t - \beta_3 & : z_t < \beta_3 \\ 0 & : \beta_3 \leq z_t \leq \beta_4 \\ z_t - \beta_4 & : z_t < \beta_4 \end{cases}$$



Example – blind Wiener learning

The measurements are given by

$$y_t = \begin{pmatrix} y_{1,t} \\ y_{2,t} \end{pmatrix} = \begin{pmatrix} h_1(z_t, \beta) \\ h_2(z_t, \beta) \end{pmatrix} + e_t, \qquad e_t \sim \mathcal{N}\left(0, \begin{pmatrix} r_1 & 0 \\ 0 & r_2 \end{pmatrix}\right),$$

The task is to learn this model based on $T = 1000$ measurements of the output ("blind" case), $y_{1:1000}$.

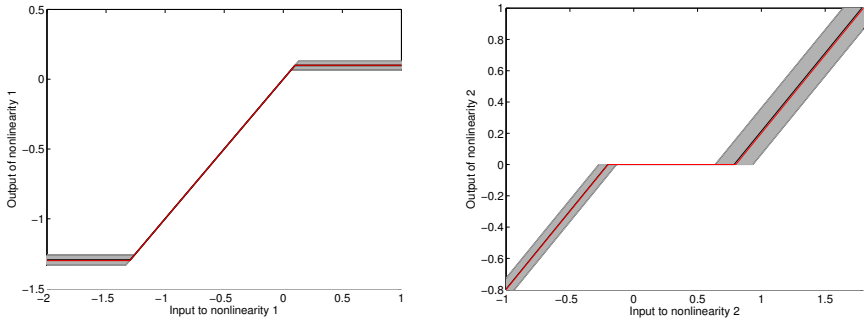
The input is chosen as $u_t \sim \mathcal{N}(0, 1)$. Initial values for the measurement variance are $\widehat{r}_1 = \widehat{r}_2 = 0.1$. The initial values for $\widehat{\eta}$ were chosen as $\widehat{\eta}_i = \frac{\eta_i^\star}{10}$, to reflect that they are unknown. The LGSS model is initialised via a subspace algorithm based on the measurements $\{y_{1,1}, \cdots, y_{1,T}\}$ from the dead zone nonlinearity.

Employ the EM alg. with $N = 100$ particles. The algorithm was terminated after just 100 iterations. Plots below are based on 100 realisations of data.

Example – blind Wiener learning

Bode plot of estimated mean (black), true system (red) and the result for all 100 realisations (gray).

Example – blind Wiener learning



Estimated mean (black), true static nonlinearity (red) and the result for all 100 realisations (gray).

Adrian Wills, Thomas B. Schön, Lennart Ljung and Brett Ninness. **Identification of Hammerstein-Wiener Models**. *Automatica*, 2012. (Accepted for publication)

Bayesian inference using Particle Markov chain Monte Carlo (PMCMC)

Bayesian inference in a general SSM

The **task** is to compute the pdf $p(\theta \mid y_{1:T})$ for a model on the form

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

Can we set up an MCMC sampler to solve this problem?

Directly targeting $\pi(\theta) = p(\theta \mid y_{1:T})$ is not possible, since $p(\theta \mid y_{1:T})$ cannot be evaluated pointwise. Recall,

$$p(\theta \mid y_{1:T}) = \frac{p(y_{1:T} \mid \theta)p(\theta)}{p(y_{1:T})},$$

where it is not possible to pointwise evaluate the likelihood $p(y_{1:T} \mid \theta)$ for the SSM above.

Bayesian inference in a general SSM

Way forward: Target $\pi(\theta, x_{1:T}) = p(\theta, x_{1:T} \mid y_{1:T})$ instead. In order to understand why this works, note that

$$p(\theta, x_{1:T} \mid y_{1:T}) = \frac{p(x_{1:T}, y_{1:T} \mid \theta)p(\theta)}{p(y_{1:T})},$$

where

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

This means that we can now evaluate the target density pointwise.

Bayesian inference in a general SSM

We need one more thing for a working MCMC algorithm

How do we create a proposal distribution capable of proposing samples from relevant parts of the state space?

The state space $\mathcal{X}^T \times \Theta$ is huge. Hence, it is key that it is explored in an efficient manner!

How can this be done?

Make use of

- the model,
- and the observed measurements $y_{1:T}$.

Together they provide a lot of information about which parts of the state space that are most interesting.

Bayesian inference in a general SSM

Indeed, using SMC algorithms we can then turn the information in

- the model,
- and the observed measurements $y_{1:T}$

into an efficient proposal distribution that captures what we know about where in the state space to propose new samples.

The result is a family of algorithms referred to as **particle MCMC (PMCMC)**.

The **fundamental idea underlying PMCMC** is to make use of an SMC sampler to construct a proposal for an MCMC sampler.

We will focus on the **particle Metropolis Hastings (PMH)** sampler in this course.

SMC again

Notation: $\mathbf{w}_t \triangleq \{w_t^1, \ldots, w_t^N\}$.

Algorithm 3 Sequential Monte Carlo (SMC)

1. **Initialise:** Sample $x_1^i \sim Q_1(x_1)$ and set $w_1^i = W_1(x_1^i)$. Set $t = 1$.
2. **For** $t = 2 : T$ **do:**
   (a) Resampling: $a_t^i \sim R(a_t \mid \mathbf{w}_{t-1})$.
   (b) Sample from the proposal kernel: $x_t^i \sim Q_t(x_t \mid x_{1:t-1}^{a_t^i})$ and set $x_{1:t}^i = \{x_{1:t-1}^{a_t^i}, x_t^i\}$.
   (c) Weighting: $w_{t+1}^i = W_t(x_{t+1}^i, \tilde{x}_{t-1}^{a_t^i})$.

Here: a_t^i to denote the **index** of the parent/ancestor at time $t-1$ of particle x_t^i.

SMC again

Just like any algorithm that is used to generate random numbers there is an **underlying distribution** also for the SMC sampler that encodes the probabilistic properties of the involved stochastic variables.

The SMC sampler generates a realisation of the random variables $\mathbf{X_{1:T}}$ and $\mathbf{A_{2:T}}$, where the pdf is

$$\psi(\mathbf{x}_{1:T}, \mathbf{a}_{2:T}) = \underbrace{\left\{\prod_{i=1}^{N} Q_1(x_1^i)\right\}}_{\text{initialisation}} \left\{\prod_{t=2}^{T}\prod_{i=1}^{N} \underbrace{R(a_t^i \mid \mathbf{w}_{t-1})}_{\text{Resampling}} \underbrace{Q_t\left(x_t^i \mid x_{1:t-1}^{a_t^i}\right)}_{\text{Proposing new particles}}\right\}$$

and it is defined on the space $\mathcal{X}^{TN} \times \{1, \ldots, N\}^{(T-1)N}$.

PMMH target

Use the following target distribution

$$\phi(\theta, \mathbf{x}_{1:T}, \mathbf{a}_{2:T}, k) \triangleq \frac{\pi(\theta, x_{1:T}^k)}{N^T} \frac{\psi^\theta(\mathbf{x}_{1:T}, \mathbf{a}_{2:T})}{Q_1^\theta(x_1^{b_1^k}) \prod_{t=2}^{T} M_t^\theta\left(a_t^k, x_t^{b_t^k}\right)}$$

Using this target we can now show that the acceptance probability is given by

$$a = \min\left(1, \frac{\widehat{p}(y_{1:T} \mid \theta^*) p(\theta^*) q(\theta[m-1] \mid \theta^*)}{\widehat{p}(y_{1:T} \mid \theta[m-1]) p(\theta[m-1]) q(\theta^* \mid \theta[m-1])}\right)$$

PMMH algorithm

1. **Initialise:** Set $\theta[0]$ and run an SMC sampler targeting $p(x_{1:T} \mid \theta[0], y_{1:T})$, sample $x_{1:T}[0] \sim \widehat{p}^N(x_{1:T} \mid \theta[0], y_{1:T})$ and compute $\widehat{p}(y_{1:T} \mid \theta[0])$.
2. **for** $m = 1$ **to** M **do**
   (a) Sample $\theta^* \sim q(\theta \mid \theta[m-1])$.
   (b) Run an SMC sampler targeting $p(x_{1:T} \mid \theta^*, y_{1:T})$, sample $x^*_{1:T} \sim \widehat{p}^N(x_{1:T} \mid \theta^*, y_{1:T})$ and compute the $\widehat{p}(y_{1:T} \mid \theta^*)$.
   (c) Compute the acceptance probability

$$a = \min\left(1, \frac{\widehat{p}(y_{1:T} \mid \theta^*)p(\theta^*)q(\theta[m-1] \mid \theta^*)}{\widehat{p}(y_{1:T} \mid \theta[m-1])p(\theta[m-1])q(\theta^* \mid \theta[m-1])}\right)$$

   (d) With probability a, set the next state $z[m] = \{\theta[m], x_{1:T}[m]\}$ of the Markov chain to $\{\theta^*, x^*_{1:T}\}$ and $\widehat{p}(y_{1:T} \mid \theta[m]) = \widehat{p}(y_{1:T} \mid \theta^*)$ and with probability $1-a$ set $z[m] = \{\theta[m-1], x_{1:T}[m-1]\}$ and $\widehat{p}(y_{1:T} \mid \theta[m]) = \widehat{p}(y_{1:T} \mid \theta[m-1])$.

PMMH

The PMMH sampler is a **standard MCMC sampler on a non-standard space**.

Put slightly differently, the PMMH sampler is a **standard MCMC sampler targeting a non-standard target distribution**.

We can also derive a Particle Gibbs (PG) sampler with backward simulation (PG-BS). This is in fact what we used for the blind Wiener example mentioned in the introduction of Part 1.

References for PMCMC

General references for PMCMC:

- Christophe Andrieu, Arnaud Doucet and Roman Holenstein. **Particle Markov chain Monte Carlo methods**. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269-342, June 2010.
- Fredrik Lindsten and Thomas B. Schön. **On the use of backward simulation in the particle Gibbs sampler**. *Proceedings of the 37th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Kyoto, Japan, March 2012.
- Fredrik Lindsten, Michael I. Jordan and Thomas B. Schön, **Ancestral Sampling for Particle Gibbs**. *Proceedings of Neural Information Processing Systems (NIPS)*, Lake Tahoe, NV, US, December, 2012. (Accepted for publication)

Using PMCMC for nonlinear system identification:

- 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.

The aim of this course

The **aim of this course** has been 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 (particle filters and particle smoothers) for nonlinear state inference problems.
- Expectation maximisation (EM) and Markov chain Monte Carlo (MCMC) methods for nonlinear system identification.

Great opportunities for research!!

Much interesting research remains to be done in solving nonlinear inference/learning problems using SMC and/or MCMC methods!!

Feel free to contact me (now or later) in case you have questions or want to discuss things.

All feedback (small and big) on how to improve the lectures and the lecture notes are very welcome

Thank you for listening!!