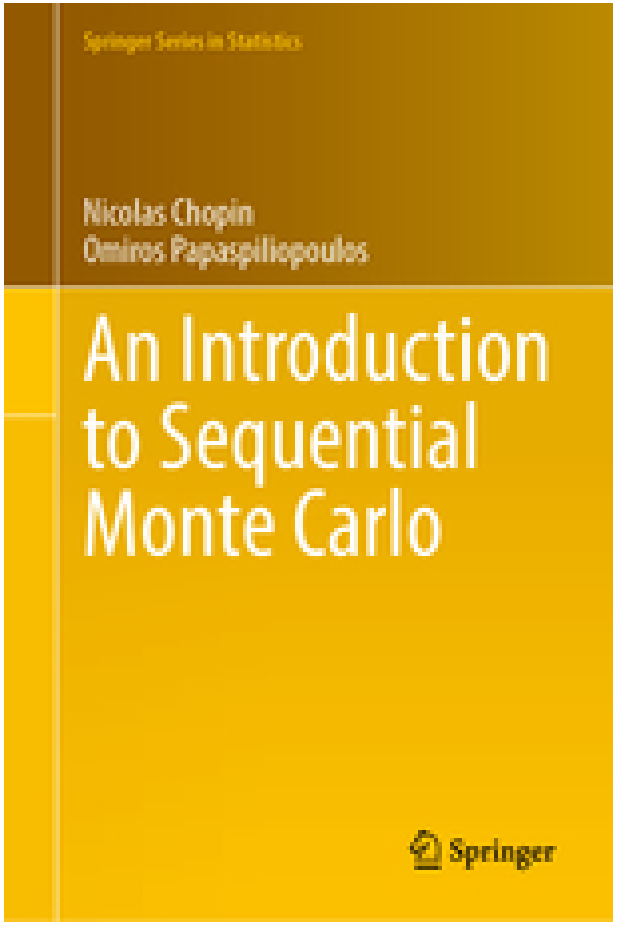
An introduction to Sequential Monte Carlo

nicolas.chopin@ensae.fr

Course based on



(Mis)conceptions about particle filters

- Something useful only for very specific models (hidden Markov models, state-space models);

(Mis)conceptions about particle filters

- Something useful only for very specific models (hidden Markov models, state-space models);
- Or alternatively something as versatile as MCMC

(Mis)conceptions about particle filters

- Something useful only for very specific models (hidden Markov models, state-space models);
- Or alternatively something as versatile as MCMC
- Which one it is?

Quick look

Let's have a quick look at a particle filter.

Structure

Algorithm 1: Generic PF algorithm

`Operations involving index` n `must be performed for` $n = 1, \ldots, N$.

$X_0^n \sim \mathbb{M}_0(\mathrm{d}x_0)$

$w_0^n \leftarrow G_0(X_0^n)$

$W_0^n \leftarrow w_0^n / \sum_{m=1}^N w_0^m$

for $t = 1$ **to** T **do**

- $A_t^{1:N} \sim \texttt{resample}(W_{t-1}^{1:N})$
- $X_t^n \sim M_t(X_{t-1}^{A_t^n}, \mathrm{d}x_t)$
- $w_t^n \leftarrow G_t(X_{t-1}^{A_t^n}, X_t^n)$
- $W_t^n \leftarrow w_t^n / \sum_{m=1}^N w_t^m$

Comments

- All particle filters have (essentially) this structure. (Let's ignore variations based on alternative resampling schemes, etc.)

Comments

- All particle filters have (essentially) this structure. (Let's ignore variations based on alternative resampling schemes, etc.)
- The user must specify:
  - kernel $M_t(x_{t-1}, \mathrm{d}x_t)$: that's how we simulate particle X_t^n, given a certain ancestor $X_{t-1}^{A_t^n}$;
  - Function $G_t(x_{t-1}, x_t)$; that's how we reweight/grade particle X_t^n (and its ancestor).

Comments

- All particle filters have (essentially) this structure. (Let's ignore variations based on alternative resampling schemes, etc.)
- The user must specify:
  - kernel $M_t(x_{t-1}, \mathrm{d}x_t)$: that's how we simulate particle X_t^n, given a certain ancestor $X_{t-1}^{A_t^n}$;
  - Function $G_t(x_{t-1}, x_t)$; that's how we reweight/grade particle X_t^n (and its ancestor).
- Easy part: **How**. Less easy: **Why**

Introduction to state-space models

nicolas.chopin@ensae.fr

Section 1

Presentation of state-space models

Objectives

The aim of this chapter is to define state-space models, give examples of such models from various areas of science, and discuss their main properties.

A first definition (with functions)

A time series model that consists of two discrete-time processes $\{X_t\} := (X_t)_{t \geq 0}$, $\{Y_t\} := (Y_t)_{t \geq 0}$, taking values respectively in spaces $\mathcal{X}$ and $\mathcal{Y}$, such that

$$X_t = K_t(X_{t-1}, U_t, \theta), \quad t \geq 1$$
$$Y_t = H_t(X_t, V_t, \theta), \quad t \geq 0$$

where K_0, K_t, H_t, are determistic functions, $\{U_t\}$, $\{V_t\}$ are sequences of i.i.d. random variables (*noises*, or *shocks*), and $\theta \in \Theta$ is an unknown parameter.

A first definition (with functions)

A time series model that consists of two discrete-time processes $\{X_t\} := (X_t)_{t \geq 0}$, $\{Y_t\} := (Y_t)_{t \geq 0}$, taking values respectively in spaces $\mathcal{X}$ and $\mathcal{Y}$, such that

$$X_t = K_t(X_{t-1}, U_t, \theta), \quad t \geq 1$$
$$Y_t = H_t(X_t, V_t, \theta), \quad t \geq 0$$

where K_0, K_t, H_t, are determistic functions, $\{U_t\}$, $\{V_t\}$ are sequences of i.i.d. random variables (*noises*, or *shocks*), and $\theta \in \Theta$ is an unknown parameter.

This is a popular way to define SSMs in Engineering. Rigorous, but not sufficiently general.

A second definition (with densities)

$$
\begin{aligned}
p_\theta(x_0) &= p_0^\theta(x_0) \\
p_\theta(x_t|x_{0:t-1}) &= p_t^\theta(x_t|x_{t-1}) \quad t \geq 1 \\
p_\theta(y_t|x_{0:t}, y_{0:t-1}) &= f_t^\theta(y_t|x_t)
\end{aligned}
$$

A second definition (with densities)

$$
\begin{aligned}
p_\theta(x_0) &= p_0^\theta(x_0) \\
p_\theta(x_t|x_{0:t-1}) &= p_t^\theta(x_t|x_{t-1}) \quad t \geq 1 \\
p_\theta(y_t|x_{0:t}, y_{0:t-1}) &= f_t^\theta(y_t|x_t)
\end{aligned}
$$

Not so rigorous (or not general enough): some models are such that $X_t|X_{t-1}$ does not admit a probability density (with respect to a fixed dominating measure).

Section 2

Examples of state-space models

Signal processing: tracking, positioning, navigation

X_t is position of a moving object, e.g.

$$X_t = X_{t-1} + U_t, \quad U_t \sim \mathcal{N}_2(0, \sigma^2 I_2),$$

and Y_t is a measurement obtained by e.g. a radar,

$$Y_t = \text{atan}\left(\frac{X_t(2)}{X_t(1)}\right) + V_t, \quad V_t \sim \mathcal{N}_1(0, \sigma_Y^2).$$

and $\theta = (\sigma^2, \sigma_Y^2)$.

Signal processing: tracking, positioning, navigation

X_t is position of a moving object, e.g.

$$X_t = X_{t-1} + U_t, \quad U_t \sim \mathcal{N}_2(0, \sigma^2 I_2),$$

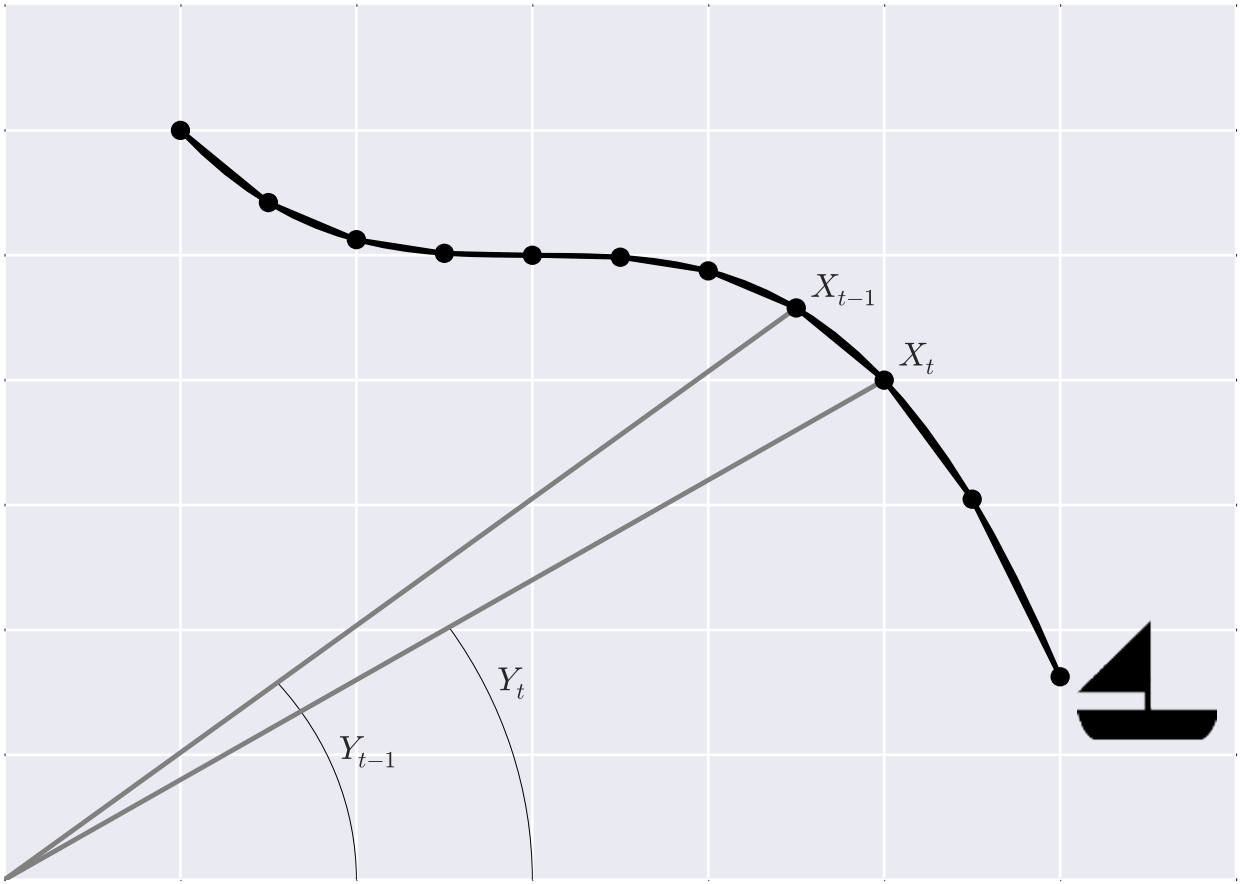
and Y_t is a measurement obtained by e.g. a radar,

$$Y_t = \text{atan}\left(\frac{X_t(2)}{X_t(1)}\right) + V_t, \quad V_t \sim \mathcal{N}_1(0, \sigma_Y^2).$$

and $\theta = (\sigma^2, \sigma_Y^2)$.

(This is called the **bearings-only tracking** model.)

Corresponding plot



GPS

In GPS applications, the velocity v_t of the vehicle is observed, so motion model is (some variation of):

$$X_t = X_{t-1} + v_t + U_t, \qquad U_t \sim \mathcal{N}_2(0, \sigma^2 I_2).$$

Also Y_t usually consists of more than one measurement.

More advanced motion model

A random walk is too erratic for modelling the position of the target; assume instead its velocitity follows a random walk. Then define:

$$X_t = \begin{pmatrix} I_2 & I_2 \\ 0_{2\times 2} & I_2 \end{pmatrix} X_{t-1} + \begin{pmatrix} 0_2 \\ U_t \end{pmatrix}, \quad U_t \sim \mathcal{N}_2(0, \sigma^2 I_2),$$

with obvious meanings for vector 0_2 and matrices $0_{2\times 2}$, I_2.

More advanced motion model

A random walk is too erratic for modelling the position of the target; assume instead its velocitity follows a random walk. Then define:

$$X_t = \begin{pmatrix} I_2 & I_2 \\ 0_{2\times 2} & I_2 \end{pmatrix} X_{t-1} + \begin{pmatrix} 0_2 \\ U_t \end{pmatrix}, \quad U_t \sim \mathcal{N}_2(0, \sigma^2 I_2),$$

with obvious meanings for vector 0_2 and matrices $0_{2\times 2}$, I_2.

Note: $X_t(1)$ and $X_t(2)$ (position) are deterministic functions of X_{t-1}: no probability density for $X_t|X_{t-1}$.

multi-target tracking

Same ideas except $\{X_t\}$ now represent the position (and velocity if needed) of a set of targets (of random size); i.e. $\{X_t\}$ is a point process.

Time series of counts (neuro-decoding, astrostatistics, genetics)

- Neuro-decoding: Y_t is a vector of d_y counts (spikes from neuron k),

$$Y_t(k)|X_t \sim \mathcal{P}(\lambda_k(X_t)), \quad \log \lambda_k(X_t) = \alpha_k + \beta_k X_t,$$

  and X_t is position+velocity of the subject's hand (in 3D).

- astro-statistics: Y_t is number of photon emissions; intensity varies over time (according to an auto-regressive process)

- Y_t is the number of 'reads', which is a noisy measurement of the transcription level X_t at position t in the genome;

Time series of counts (neuro-decoding, astrostatistics, genetics)

- Neuro-decoding: Y_t is a vector of d_y counts (spikes from neuron k),

$$Y_t(k)|X_t \sim \mathcal{P}(\lambda_k(X_t)), \quad \log \lambda_k(X_t) = \alpha_k + \beta_k X_t,$$

  and X_t is position+velocity of the subject's hand (in 3D).

- astro-statistics: Y_t is number of photon emissions; intensity varies over time (according to an auto-regressive process)

- Y_t is the number of 'reads', which is a noisy measurement of the transcription level X_t at position t in the genome;

Note: 'functional' definition of state-space models is less convenient in this case.

Stochastic volatility (basic model)

Y_t is log-return of asset price, $Y_t = \log(p_t/p_{t-1})$,

$$Y_t | X_t = x_t \sim \mathcal{N}(0, \exp(x_t))$$

where $\{X_t\}$ is an auto-regressive process:

$$X_t - \mu = \phi(X_{t-1} - \mu) + U_t, \quad U_t \sim \mathcal{N}(0, \sigma^2)$$

and $\theta = (\mu, \phi, \sigma^2)$.

Stochastic volatility (basic model)

Y_t is log-return of asset price, $Y_t = \log(p_t/p_{t-1})$,

$$Y_t | X_t = x_t \sim \mathcal{N}(0, \exp(x_t))$$

where $\{X_t\}$ is an auto-regressive process:

$$X_t - \mu = \phi(X_{t-1} - \mu) + U_t, \quad U_t \sim \mathcal{N}(0, \sigma^2)$$

and $\theta = (\mu, \phi, \sigma^2)$.

Take $|\phi| < 1$ and $X_0 \sim N(\mu, \sigma^2/(1-\rho^2))$ to impose stationarity.

Stochastic volatility (variations)

- Student dist' for noises
- skewness: $Y_t = \alpha X_t + \exp(X_t/2) V_t$
- leverage effect: correlation between U_t and V_t
- multivariate extensions

Panel data (Heiss, 2008)

For each individual i, we observe the SRHS (self-rated health status), $Y_{it} = k$ iff $\alpha_{k-1} < Y_{it}^\star \leq \alpha_k$, where $Y_{it}^\star$ is the actual health:

$$Y_{it}^\star = Z_{it}\beta + X_{it} + V_{it}, \quad V_{it} \sim \mathcal{N}(0,1)$$

and individual effect X_{it} is an AR(1):

$$X_{it} = \rho X_{i(t-1)} + \sigma\sqrt{1-\rho^2}U_{it}, \quad U_{it} \sim \mathcal{N}(0,1).$$

Panel data (Heiss, 2008)

For each individual i, we observe the SRHS (self-rated health status), $Y_{it} = k$ iff $\alpha_{k-1} < Y_{it}^\star \leq \alpha_k$, where $Y_{it}^\star$ is the actual health:

$$Y_{it}^\star = Z_{it}\beta + X_{it} + V_{it}, \quad V_{it} \sim \mathcal{N}(0,1)$$

and individual effect X_{it} is an AR(1):

$$X_{it} = \rho X_{i(t-1)} + \sigma\sqrt{1-\rho^2}U_{it}, \quad U_{it} \sim \mathcal{N}(0,1).$$

Note: large number of univariate state-space models.

Nonlinear dynamic systems in Ecology, Epidemiology, and other fields

$Y_t = X_t + V_t$, where $\{X_t\}$ is some complex nonlinear dynamic system. In Ecology for instance,

$$X_t = X_{t-1} + \theta_1 - \theta_2 \exp(\theta_3 X_{t-1}) + U_t$$

where X_t is log of population size. For some values of θ, process is nearly chaotic.

Nonlinear dynamic systems: Lokta-Volterra

Predator-prey model, where $\mathcal{X} = (\mathbb{Z}^+)^2$, $X_t(1)$ is the number of preys, $X_t(2)$ is the number of predators, and, working in continuous-time:

$$\begin{aligned} X_t(1) &\xrightarrow{\theta_1} 2X_t(1) \\ X_t(1) + X_t(2) &\xrightarrow{\theta_2} 2X_t(2), \quad t \in \mathbb{R}^+ \\ X_t(2) &\xrightarrow{\theta_3} 0 \end{aligned}$$

(but Y_t still observed in discrete time.)

Nonlinear dynamic systems: Lokta-Volterra

Predator-prey model, where $\mathcal{X} = (\mathbb{Z}^+)^2$, $X_t(1)$ is the number of preys, $X_t(2)$ is the number of predators, and, working in continuous-time:

$$\begin{aligned} X_t(1) &\xrightarrow{\theta_1} 2X_t(1) \\ X_t(1) + X_t(2) &\xrightarrow{\theta_2} 2X_t(2), \quad t \in \mathbb{R}^+ \\ X_t(2) &\xrightarrow{\theta_3} 0 \end{aligned}$$

(but Y_t still observed in discrete time.)

Intractable dynamics: We can simulate from $X_t|X_{t-1}$, but we can't compute $p_t(x_t|x_{t-1})$; see also compartmental models in Epidemiology.

State-space models with an intractable or degenerate observation process

We have seen models such that $X_t|X_{t-1}$ is intractable; $Y_t|X_t$ may be intractable as well. Let

$$X'_t = (X_t, Y_t), \quad Y'_t = Y_t + V_t, \quad V_t \sim \mathcal{N}(0, \sigma^2)$$

and use $\{(X'_t, Y'_t)\}$ as an approximation of $\{(X_t, Y_t)\}$.

State-space models with an intractable or degenerate observation process

We have seen models such that $X_t|X_{t-1}$ is intractable; $Y_t|X_t$ may be intractable as well. Let

$$X'_t = (X_t, Y_t), \quad Y'_t = Y_t + V_t, \quad V_t \sim \mathcal{N}(0, \sigma^2)$$

and use $\{(X'_t, Y'_t)\}$ as an approximation of $\{(X_t, Y_t)\}$.

$\Rightarrow$ Connection with ABC (likelihood-free inference).

Finite state-space models (aka hidden Markov models)

$\mathcal{X} = \{1, \ldots, K\}$, uses in e.g.

- speech processing; X_t is a word, Y_t is an acoustic measurement (possibly the earliest application of HMMs). Note K is quite large.
- time-series modelling to deal with heterogenity (e.g. in medecine, X_t is state of patient)
- rediscovered in Economics as Markov-switching models; there X_t is the state of the Economy (recession, growth), and Y_t is some economic indicator (e.g. GDP) which follows an ARMA process (with parameters that depend on X_t)
- also related: change-point models

Finite state-space models (aka hidden Markov models)

$\mathcal{X} = \{1, \ldots, K\}$, uses in e.g.

- speech processing; X_t is a word, Y_t is an acoustic measurement (possibly the earliest application of HMMs). Note K is quite large.
- time-series modelling to deal with heterogenity (e.g. in medecine, X_t is state of patient)
- rediscovered in Economics as Markov-switching models; there X_t is the state of the Economy (recession, growth), and Y_t is some economic indicator (e.g. GDP) which follows an ARMA process (with parameters that depend on X_t)
- also related: change-point models

Note: Not of direct interest to us, as sequential analysis may be performed *exactly*.

Section 3

Sequential analysis of state-space models

Definition

The phrase *state-space models* refers not only to its definition (in terms of $\{X_t\}$ and $\{Y_t\}$) but also to a particular **inferential scenario**: $\{Y_t\}$ is observed (data denoted $y_0, \ldots$), $\{X_t\}$ is not, and one wishes to recover the X_t's given the Y_t's, often sequentially (over time).

Filtering, prediction, smoothing

Conditional distributions of interest (at every time t)

- Filtering: $X_t | Y_{0:t}$
- Prediction: $X_t | Y_{0:t-1}$
- data prediction: $Y_t | Y_{0:t-1}$
- fixed-lag smoothing: $X_{t-h:t} | Y_{0:t}$ for $h \geq 1$
- complete smoothing: $X_{0:t} | Y_{0:t}$
- likelihood factor: density of $Y_t | Y_{0:t-1}$ (so as to compute the full likelihood)

Parameter estimation

All these tasks are usually performed for a fixed θ (assuming the model depends on some parameter θ). To deal additionally with parameter uncertainty, we could adopt a Bayesian approach, and consider e.g. the law of (θ, X_t) given $Y_{0:t}$ (for filtering). But this is often more involved.

Formal notations

- $\{X_t\}$ is a Markov process with initial law $P_0(\mathrm{d}x_0)$, and Markov kernel $P_t(x_{t-1}, \mathrm{d}x_t)$.
- $\{Y_t\}$ has conditional distribution $F_t(x_t, \mathrm{d}y_t)$, which admits probability density $f_t(y_t|x_t)$ (with respect to common dominating measure $\nu(\mathrm{d}y_t)$).
- when needed, dependence on θ will be made explicit as follows: $P_t^\theta(x_{t-1}, \mathrm{d}x_t)$, $f_t^\theta(y_t|x_t)$, etc.

Applications of SMC beyond state-space models

nicolas.chopin@ensae.fr

Bayesian sequential estimation

For a Bayesian model, with parameter θ, data $Y_0, \ldots, Y_t, \ldots$ (no latent variables), we would like to approximate recursively the posterior $p_t(\theta|y_{0:t})$. Could we treat this as a **filtering** problem, where

$$X_t = \Theta$$

is a constant process?

Tempering

We wish to simulate from (or compute the normalising constant of):

$$\pi(\mathrm{d}\theta) \propto \mu(\mathrm{d}\theta) \exp\{-V(\theta)\}$$

To do so, we introduce a **tempering** sequence:

$$\mathbb{P}_t(\mathrm{d}\theta) \propto \mu(\mathrm{d}\theta) \exp\{-\lambda_t V(\theta)\}$$

where $0 = \lambda_0 < \ldots < \lambda_T = 1$, and use SMC to target recursively $\mathbb{P}_0$, $\mathbb{P}_1$, ...

Plot of tempering sequence

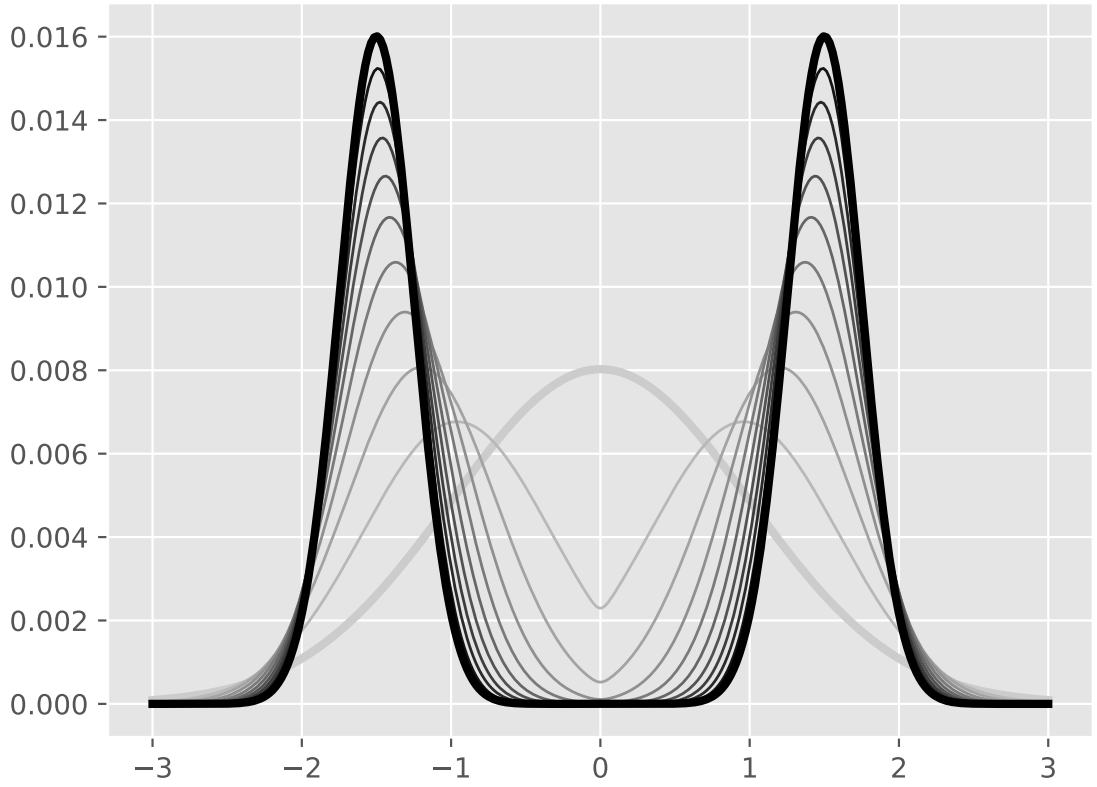


Figure 1: From a Gaussian to a mixture of two Gaussians

Fundamental question

In all these applications, how are we going to set the Markov kernels M_t to simulate the particles?

Fundamental question

In all these applications, how are we going to set the Markov kernels M_t to simulate the particles?

Hint: use MCMC.

Laying out the foundations: importance sampling, resampling, Feynman-Kac

nicolas.chopin@ensae.fr

Section 1

Importance sampling

Basic identity

$$\int_{\mathcal{X}} \varphi(x) q(x) \mathrm{d}x = \int_{\mathcal{X}} \varphi(x) \frac{q(x)}{m(x)} m(x) \mathrm{d}x$$

Basic identity

$$\int_{\mathcal{X}} \varphi(x) q(x) \mathrm{d}x = \int_{\mathcal{X}} \varphi(x) \frac{q(x)}{m(x)} m(x) \mathrm{d}x$$

Warning: valid only if $m(x) = 0 \Rightarrow q(x) = 0$.

Basic identity

$$\int_{\mathcal{X}} \varphi(x) q(x) \mathrm{d}x = \int_{\mathcal{X}} \varphi(x) \frac{q(x)}{m(x)} m(x) \mathrm{d}x$$

Warning: valid only if $m(x) = 0 \Rightarrow q(x) = 0$.

Normalised IS estimator:

$$\frac{1}{N} \sum_{n=1}^{N} w(X^n) \varphi(X^n)$$

where $X^n \sim m$, $w(x) = q(x)/m(x)$.

Auto-normalised importance sampling

If m, q may be computed only up to constant, i.e. $q(x) = q_u(x)/Z_q$, $m(x) = m_u(x)/Z_m$, where Z_q, Z_m are intractable, consider instead:

$$\begin{aligned}\int_{\mathcal{X}} \varphi(x) q(x)\,\mathrm{d}x &= \frac{\int_{\mathcal{X}} \varphi(x)\frac{q(x)}{m(x)} m(x)\,\mathrm{d}x}{\int_{\mathcal{X}} \frac{q(x)}{m(x)} m(x)\,\mathrm{d}x} \\ &= \frac{\int_{\mathcal{X}} \varphi(x)\frac{q_u(x)}{m_u(x)} m(x)\,\mathrm{d}x}{\int_{\mathcal{X}} \frac{q_u(x)}{m_u(x)} m(x)\,\mathrm{d}x}\end{aligned}$$

Auto-normalised importance sampling

If m, q may be computed only up to constant, i.e. $q(x) = q_u(x)/Z_q$, $m(x) = m_u(x)/Z_m$, where Z_q, Z_m are intractable, consider instead:

$$\begin{aligned}\int_{\mathcal{X}} \varphi(x) q(x)\,\mathrm{d}x &= \frac{\int_{\mathcal{X}} \varphi(x)\frac{q(x)}{m(x)} m(x)\,\mathrm{d}x}{\int_{\mathcal{X}} \frac{q(x)}{m(x)} m(x)\,\mathrm{d}x} \\ &= \frac{\int_{\mathcal{X}} \varphi(x)\frac{q_u(x)}{m_u(x)} m(x)\,\mathrm{d}x}{\int_{\mathcal{X}} \frac{q_u(x)}{m_u(x)} m(x)\,\mathrm{d}x}\end{aligned}$$

which suggests the following, based on $w(x) = q_u(x)/m_u(x)$:

$$\sum_{n=1}^{N} W^n \varphi(X^n), \qquad W^n = \frac{w(X^n)}{\sum_{m=1}^{N} w(X^m)}.$$

Change of measure

In a more general setting, the proposal and the target may be probability measures $\mathbb{M}(\mathrm{d}x)$, $\mathbb{Q}(\mathrm{d}x)$, and provided that $\mathbb{M}$ dominates $\mathbb{Q}$, we way reweight according to a function proportional to the **Radon-Nykodim derivative**.

Change of measure

In a more general setting, the proposal and the target may be probability measures $\mathbb{M}(\mathrm{d}x)$, $\mathbb{Q}(\mathrm{d}x)$, and provided that $\mathbb{M}$ dominates $\mathbb{Q}$, we way reweight according to a function proportional to the **Radon-Nykodim derivative**.

This is equivalent to applying a change of measure:

$$\mathbb{Q}(\mathrm{d}x) = \frac{1}{L}\mathbb{M}(\mathrm{d}x)w(x)$$

where $L = \mathbb{M}(w) \in (0, \infty)$.

Approximating moments, or approximating a distribution?

Since, for any function φ, we have

$$\sum_{n=1}^{N} W^n \varphi(X^n) \approx \mathbb{Q}(\varphi)$$

we could say that:

$$\mathbb{Q}^N(\mathrm{d}x) \approx \mathbb{Q}(\mathrm{d}x)$$

where $\mathbb{Q}^n$ is the following **random distribution**:

$$\mathbb{Q}^N(\mathrm{d}x) = \sum_{n=1}^{N} W^n \delta_{X^n}(\mathrm{d}x)$$

(In particular, $\mathbb{Q}^N(\varphi) = \sum_{n=1}^{N} W^n \varphi(X^n)$.)

ESS (Effective sample size)

A popular criterion:

$$\mathrm{ESS} = \frac{1}{\sum_{n=1}^{N} (W^n)^2} = \frac{\left(\sum_{n=1}^{N} w(X^n)\right)^2}{\sum_{n=1}^{N} w(X^n)^2}$$

which has several justifications:

- $\mathrm{ESS} \in [1, N]$.
- If $w(x) = \mathbb{1}_A(x)$, ESS is number of non-zero weights.
- N/ESS converges to the **chi-square (pseudo-)distance** of q relative to m: $\int_{\mathcal{X}} m(q/m - 1)^2$.

Curse of dimensionality in importance sampling

Now assume that both m and q are densities of IID variables $X_0, \ldots, X_T$; then

$$\frac{q(x)}{m(x)} = \prod_{t=0}^{T} \frac{q_1(x_t)}{m_1(x_t)}$$

and the variance of the weights is of the form $r^{T+1} - 1$, with $r \geq 1$.

Curse of dimensionality in importance sampling

Now assume that both m and q are densities of IID variables $X_0, \ldots, X_T$; then

$$\frac{q(x)}{m(x)} = \prod_{t=0}^{T} \frac{q_1(x_t)}{m_1(x_t)}$$

and the variance of the weights is of the form $r^{T+1} - 1$, with $r \geq 1$.

IID scenario not completely fictitious.

Section 2

Feynman-Kac

Feynman-Kac structure

Consider the following **generic** class of distributions: for each $t \geq 0$:

- $\mathbb{M}_t(\mathrm{d}x_{0:t})$ is the distribution of a *Markov* process $\{X_t\}$; with density:

$$= m_0(x_0)m_1(x_1|x_0)\ldots m_t(x_t|x_{t-1})$$

- $\mathbb{Q}_t(\mathrm{d}x_{0:t})$ is the distribution that corresponds to the following **change of measure**, that is the distribution with density

$$= \frac{1}{L_t} m_0(x_0)m_1(x_1|x_0)\ldots m_t(x_t|x_{t-1}) \left\{ \prod_{s=0}^{t} G_s(x_{s-1}, x_s) \right\}$$

How to approximate the $\mathbb{Q}_t$'s?

Importance sampling? Curse of dimensionality.

How to approximate the $\mathbb{Q}_t$'s?

Importance sampling? Curse of dimensionality.

However, if we are only interested in certain **marginal distributions** of the $\mathbb{Q}_t$, we might be able to express our calculations in a much smaller dimension. This is the key observation.

Forward recursion

Suppose we have computed the marginal density $q_{t-1}(x_{t-1})$ (of variable X_{t-1} with respect to $\mathbb{Q}_{t-1}$). Then:

1. Extend:

$$q_{t-1}(x_{t-1}, x_t) = q_{t-1}(x_{t-1}) m_t(x_t | x_{t-1}).$$

2. Embrace (the next potential function):

$$q_t(x_{t-1}, x_t) \propto q_{t-1}(x_{t-1}, x_t) G_t(x_{t-1}, x_t)$$

3. Extinguish (marginalize out X_{t-1})

$$q_t(x_t) = \int_{\mathcal{X}} q_t(x_{t-1}, x_t) \mathrm{d}x_{t-1}$$

Why do we care?

Let's go back to state-space models. The smooting distribution at time t is the distribution of $X_{0:t}$ given $Y_{0:t} = y_{0:t}$, and has the expression:

$$\propto p_0(x_0) \prod_{s=1}^{t} p_t(x_t|x_{t-1}) \prod_{s=0}^{t} f_t(y_t|x_t)$$

hence, the same structure as $\mathbb{Q}_t(\mathrm{d}x_{0:t})$ provided we take:

- $m_t(x_t|x_{t-1}) = p_t(x_t|x_{t-1})$
- $G_t(x_{t-1}, x_t) = f_t(y_t|x_t)$

Why do we care?

Let's go back to state-space models. The smooting distribution at time t is the distribution of $X_{0:t}$ given $Y_{0:t} = y_{0:t}$, and has the expression:

$$\propto p_0(x_0) \prod_{s=1}^{t} p_t(x_t|x_{t-1}) \prod_{s=0}^{t} f_t(y_t|x_t)$$

hence, the same structure as $\mathbb{Q}_t(\mathrm{d}x_{0:t})$ provided we take:

- $m_t(x_t|x_{t-1}) = p_t(x_t|x_{t-1})$
- $G_t(x_{t-1}, x_t) = f_t(y_t|x_t)$

In particular, the forward recursion may be used to compute recursively the filtering distributions.

Practical implementations of the forward recursions

- finite state-space: replace integrals by sums, exact calculations, complexity $\mathcal{O}(K^2)$ per time step (Baum-Petrie);
- linear-Gaussian state-space models: propagating mean/variance through the **Kalman filter**;
- other state-space models: importance sampling and resampling $\Rightarrow$ particle filters.

Section 3

Resampling

Motivation

$$\mathbb{Q}_0^N(\mathrm{d}x_0) = \sum_{n=1}^N W_0^n \delta_{X_0^n}, \qquad X^n \sim \mathbb{M}_0, \qquad W_0^n = \frac{w_0(X_0^n)}{\sum_{m=1}^N w_0(X_0^m)},$$

and now interested in

$$(\mathbb{Q}_0 M_1)(\mathrm{d}x_{0:1}) = \mathbb{Q}_0(\mathrm{d}x_0) M_1(x_0, \mathrm{d}x_1).$$

Two solutions:

First solution

Extend $X_1^n \sim M_1(X_0^n, \mathrm{d}x_1)$.

This ignores the intermediate approx $\mathbb{Q}_0^N(\mathrm{d}x_0)$.

Second solution: resampling

$$\mathbb{Q}_0^N(\mathrm{d}x_0)M_1(x_0,\mathrm{d}x_1) = \sum_{n=1}^N W_0^n M_1(X_0^n,\mathrm{d}x_1)$$

and now we **sample** from this approximation:

$$\frac{1}{N}\sum_{n=1}^N \delta_{\widetilde{X}_{0:1}^n}, \qquad \text{where } \widetilde{X}_{0:1}^n \sim \mathbb{Q}_0^N(\mathrm{d}x_0)M_1(x_0,\mathrm{d}x_1).$$

One way to obtain such samples is to do:

$$\widetilde{X}_{0:1}^n = (X_0^{A_1^n}, X_1^n), \qquad A_1^{1:N} \sim \mathcal{M}(W_0^{1:N}), \qquad X_1^n \sim M_1(X_0^{A_1^n},\mathrm{d}x_1)$$

Why resample??

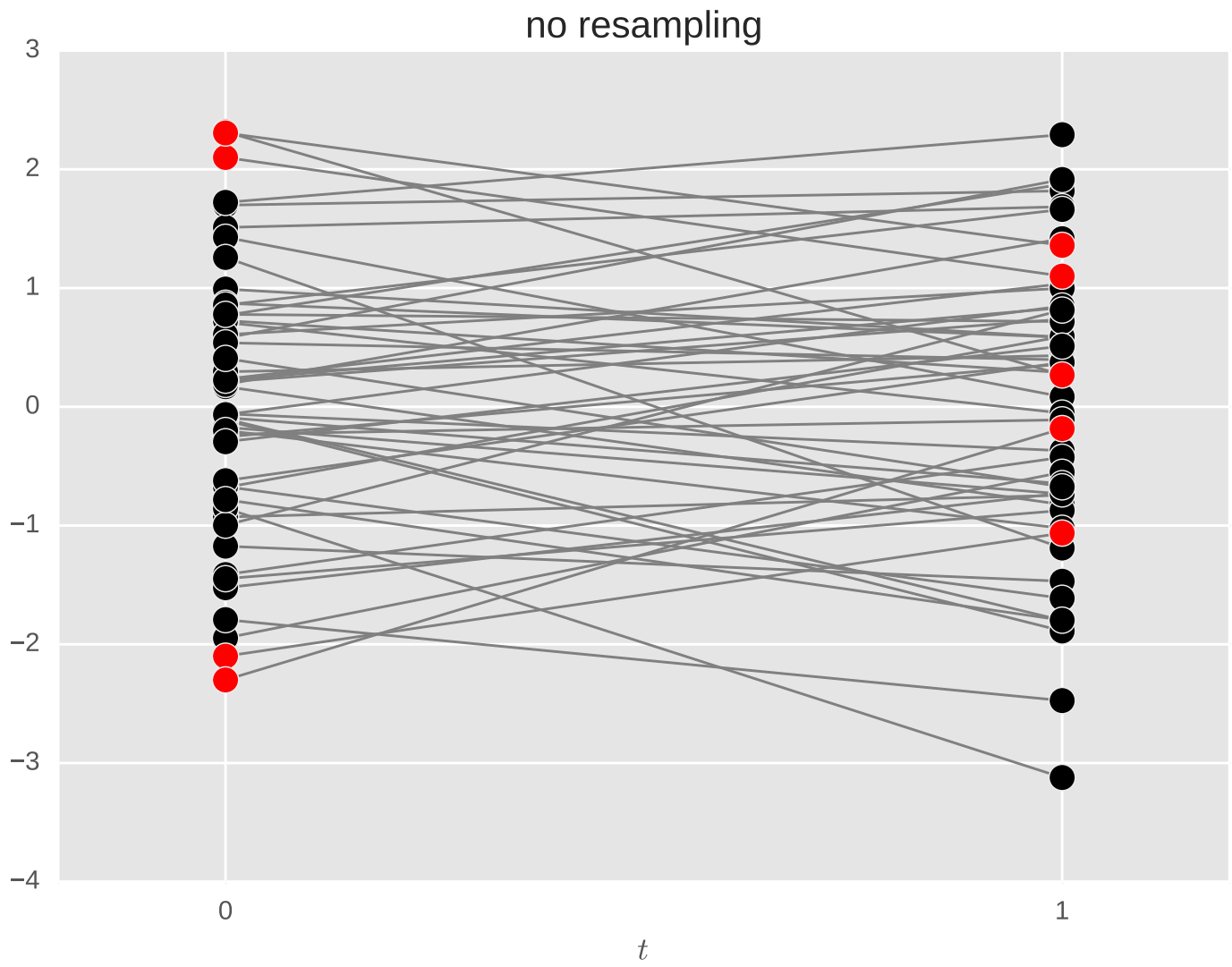
Toy example

- $\mathcal{X} = \mathbb{R}$, $\mathbb{M}_0$ is $\mathcal{N}(0,1)$, $w_0(x) = \mathbb{1}(|x| > \tau)$; thus $\mathbb{Q}_0$ is a truncated Gaussian distribution
- $M_1(x_0, \mathrm{d}x_1)$ so that $X_1 = \rho X_0 + \sqrt{1-\rho^2}U$, with $U \sim N(0,1)$
- $\varphi(x_1) = x_1$; note that $(\mathbb{Q}_0 M_1)(\varphi) = 0$

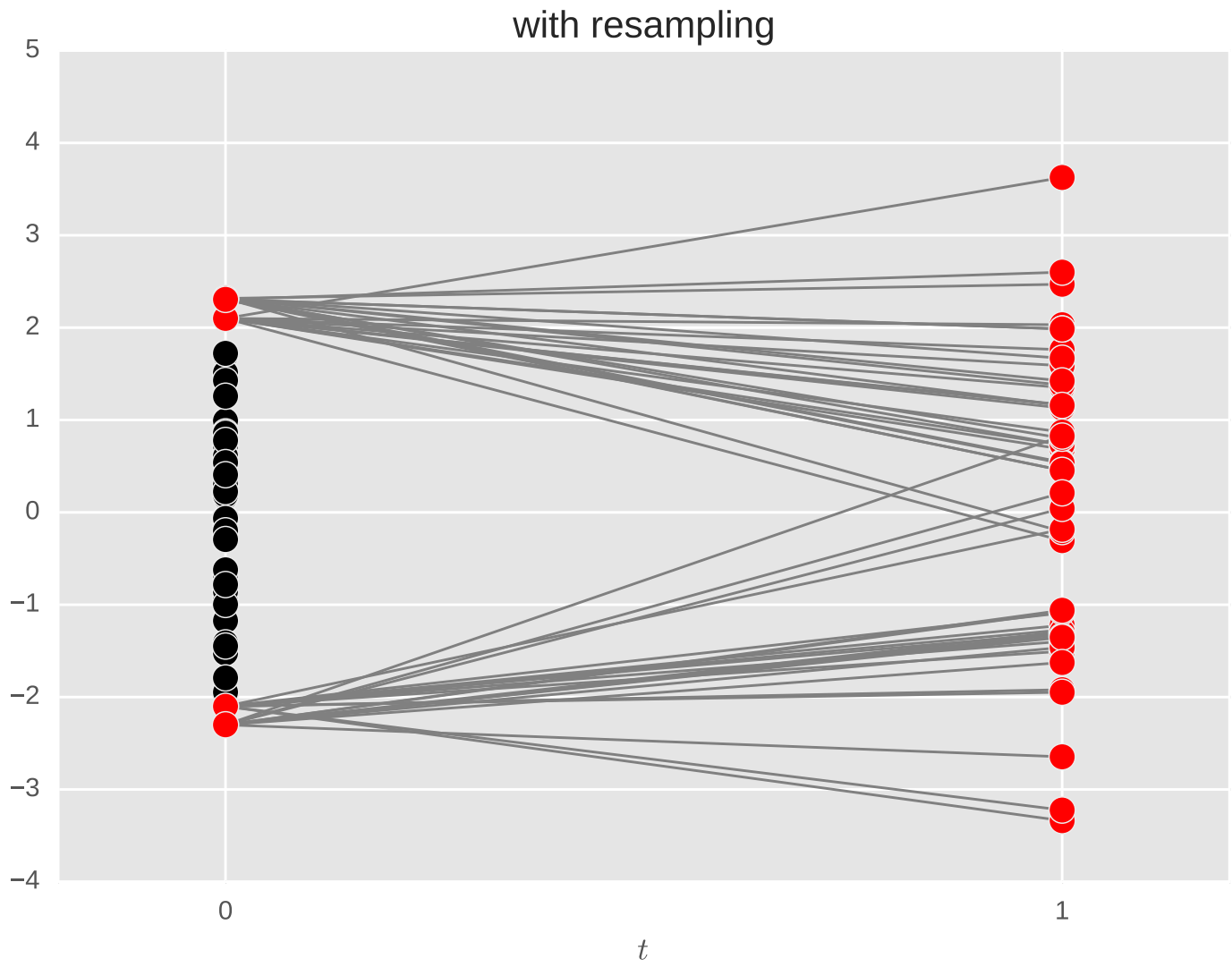
$$\widehat{\varphi}_{\mathrm{IS}} = \sum_{n=1}^{N} W_0^n X_1^n, \qquad (X_0^n, X_1^n) \sim \mathbb{M}_0 M_1$$

$$\widehat{\varphi}_{\mathrm{IR}} = N^{-1} \sum_{n=1}^{N} X_1^n, \qquad X_1^n \sim \mathbb{Q}_0^N M_1$$

No resampling



With resampling



Bottom line

Resampling sacrifices the past to save the present.

Resampling adds noise

Resampling amounts to replacing

$$\mathbb{Q}^N(\mathrm{d}x) = \sum_{n=1}^{N} W^n \delta_{X^n}(\mathrm{d}x)$$

with

$$\frac{1}{N} \sum_{n=1}^{N} O^n \delta_{X^n}(\mathrm{d}x)$$

where O^n is the **number of off-springs** of particle n. Note O^n is random, takes values in $\{1, \ldots, N\}$, and is such that

$$\mathbb{E}[O^n] = NW^n$$

(Unbiasedness property of resampling)

Resampling: drawbacks?

Assume that $W^n \approx 1/N$ (weights are nearly constant). Then

$$O^n \approx \text{Binomial}(N, 1/N)$$

and in particular

$$P(O^n = 0) = (1 - \frac{1}{N})^N \approx e^{-1} \approx 0.37$$

which seems quite **wasteful**.

Solutions:

- Resample only when ESS is low.
- Use better (=less noisy) resampling schemes.

How to resample

nicolas.chopin@ensae.fr

Section 1

Multinomial resampling

Inverse CDF

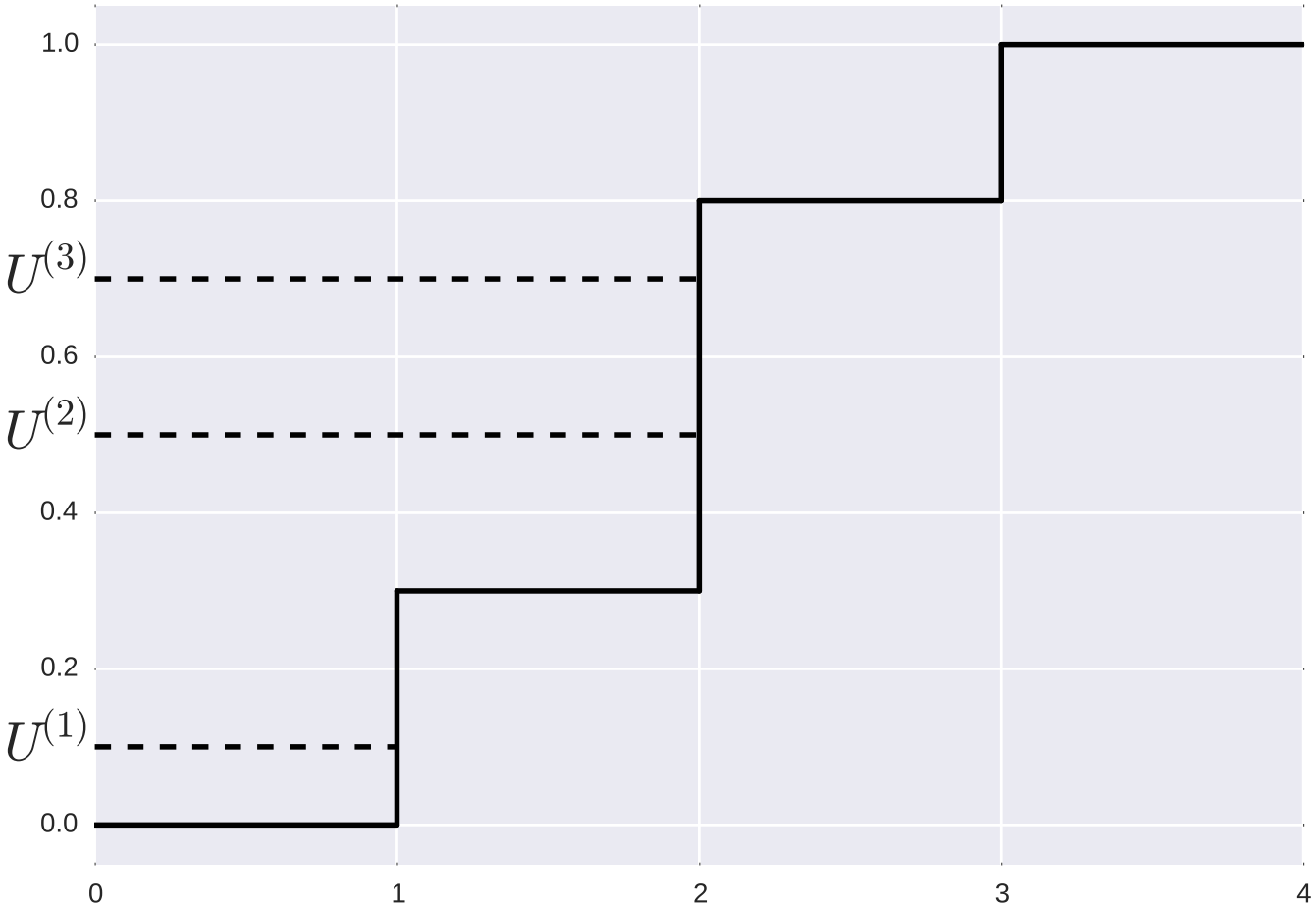


Figure 1: CDF $F(x) = \sum_{n=1}^{N} W_t^n \mathbb{1}\{n \leq x\}$

inverse CDF algorithm

In the resampling step, we must simulate N times from $\mathcal{M}(W_t^{1:N})$, the multinomial distribution that generates label n with probability W_t^n.

Inverse transform method: generate N uniform variates U^m, $m \in 1:N$, and set A_t^m according to:

$$C^{n-1} \leq U^m \leq C^n \quad \Leftrightarrow \quad A_t^m = n$$

where the C^n's are the cumulative weights:

$$C^0 = 0, \quad C^n = C^{n-1} + W_t^n.$$

inverse CDF algorithm

In the resampling step, we must simulate N times from $\mathcal{M}(W_t^{1:N})$, the multinomial distribution that generates label n with probability W_t^n.

Inverse transform method: generate N uniform variates U^m, $m \in 1:N$, and set A_t^m according to:

$$C^{n-1} \leq U^m \leq C^n \quad \Leftrightarrow \quad A_t^m = n$$

where the C^n's are the cumulative weights:

$$C^0 = 0, \quad C^n = C^{n-1} + W_t^n.$$

This suggests resampling costs $\mathcal{O}(N^2)$; however, if we are given uniforms that are already **sorted**, $U^{(1)} < \ldots < U^{(N)}$, then only $2N$ comparisons need to be performed.

Resampling based on sorted uniforms

Algorithm 1: Inverse CDF (finite distribution)

Input: $W^{1:N}$, and $0 < U^{(1)} < \ldots < U^{(N)} < 1$.

Output: $A^{1:N}$ (N indices in $1:N$).

Function `icdf`$(W^{1:N}, U^{(1:N)})$**:**

- $s \leftarrow W^1$, $m \leftarrow 1$
- **for** $n = 1$ **to** N **do**
  - **while** $s < U^{(n)}$ **do**
    - $m \leftarrow m + 1$
    - $s \leftarrow s + W^m$
  - $A^n \leftarrow m$

How to generate sorted uniforms

- Generate N uniforms, then sort: $\mathcal{O}(N \log N)$ complexity (not so bad).

How to generate sorted uniforms

- Generate N uniforms, then sort: $\mathcal{O}(N \log N)$ complexity (not so bad).
- $\mathcal{O}(N)$ complexity by using properties of the Poisson process: see next slide.

Uniform spacings algorithm

Algorithm 2: Uniform spacings

Input: Integer N.

Output: Ordered sequence $0 < U^{(1)} < \ldots < U^{(N)} < 1$.

Function `uniform_spacings(`N`)`**:**

- $S^0 \leftarrow 0$
- **for** $n = 1$ **to** $(N+1)$ **do**
  - $E^n \sim \mathcal{E}(1)$
  - $S^n \leftarrow S^{n-1} + E^n$
- **for** $n = 1$ **to** N **do**
  - $U^{(n)} \leftarrow S^n / S^{N+1}$

Section 2

Alternative resampling schemes

Motivation for alternative resampling schemes

We motivated resampling as a way to sample the ancestor X_{t-1}^n from the joint distribution:

$$\sum_{n=1}^{N} W^n \delta_{X_0^n}(\mathrm{d}x_0) M_1(X_0^n, \mathrm{d}x_1)$$

Now imagine $W_{t-1}^n = 1/N$ for all n. The probability of never selecting ancestor X_{t-1}^n is $(1 - 1/N)^N \approx \exp(-1) \approx 0.37$. Seems quite wasteful.

Let $O^n = \sum_{m=1}^N \mathbb{1}\{A^m = n\}$ (number of offsprings). We would like to derive resampling schemes such that

$$\mathbb{E}[O^n] = NW^n$$

while having lower variance than multinomial resampling.

Residual resampling

Let $\mathrm{frac}(x) = x - \lfloor x \rfloor$, and take

$$O^n = \lfloor NW^n \rfloor + \widetilde{O}^n$$

with $\widetilde{O}^n$ taking values in $\mathbb{Z}^+$, and such that $\mathbb{E}[\widetilde{O}^n] = \mathrm{frac}(NW^n)$.

To generate the $\widetilde{O}^n$, use multinomial resampling, based on weights $r^n = \mathrm{frac}(NW^n)/R$, with $R = \sum_{n=1}^N \mathrm{frac}(NW^n)$.

Residual resampling: the algorithm

Input: normalised weights $W^{1:N}$

Output: $A^{1:N} \in 1:N$

(a) Compute $r^n = \mathrm{frac}(NW^n)$ (for each $n \in 1:N$) and $R = \sum_{n=1}^{N} r^n$.

(b) Construct $A^{1:(N-R)}$ as a vector of size $(N-R)$ that contains $\lfloor NW^n \rfloor$ copies of value n for each $n \in 1:N$.

(c) Sample $A^{N-R+1:N} \sim \mathcal{M}(r^{1:N}/R)$ using multinomial resampling.

Stratified and systematic resampling

We defined multinomial resampling as some operation involving N sorted **IID** uniforms $U^{(n)}$. Taking instead

$$U^{(n)} \sim \mathcal{U}\left(\frac{n-1}{N}, \frac{n}{N}\right)$$

gives **stratified resampling**. Reducing randomness further, taking

$$U^{(n)} = (n - 1 + U)/N$$

(based on a *single* uniform $U \sim \mathcal{U}(0,1)$) leads to **systematic resampling**.

Which resampling scheme to use in practice?

- systematic resampling is pretty popular: easy to implement, fast, seems to "work well".
- Recent theoretical study suggests stratified resampling has better properties.
- In practice, what matters most is to avoid multinomial resampling.

Particle filtering

nicolas.chopin@ensae.fr

Section 1

Objectives

Objectives

- introduce a generic PF algorithm for a given Feynman-Kac model $\{(M_t, G_t)\}_{t=0}^{T}$
- discuss the different algorithms one may obtain for a given state-space model, by using different Feynman-Kac formalisms.
- give more details on the implementation, complexity, and so on of the algorithm.

Section 2

The algorithm

Input

- A Feynman-Kac model $\{(M_t, G_t)\}_{t=0}^{T}$ such that:
  - the weight function G_t may be evaluated pointwise (for all t);
  - it is possible to simulate from $M_0(\mathrm{d}x_0)$ and from $M_t(x_{t-1}, \mathrm{d}x_t)$ (for any x_{t-1} and t)
- The number of particles N

Structure

Algorithm 1: Basic PF algorithm

At time 0:

(a) Generate $X_0^n \sim M_0(\mathrm{d}x_0)$.

(b) Compute $w_0^n = G_0(X_0^n)$, and $W_0^n = w_0^n / \sum_{m=1}^N w_0^m$.

Recursively, for $t = 1, \ldots, T$:

(a) Generate ancestor variables $A_t^n \sim \mathcal{M}(W_{t-1}^{1:N})$.

(b) Generate $X_t^n \sim M_t(X_{t-1}^{A_t^n}, \mathrm{d}x_t)$.

(c) Compute $w_t^n = G_t(X_{t-1}^{A_t^n}, X_t^n)$, and

$W_t^n = w_t^n / \sum_{m=1}^N w_t^m$.

Output

the algorithm delivers the following approximations at each time t:

$$\frac{1}{N}\sum_{n=1}^{N}\delta_{X_t^n} \quad \text{approximates } \mathbb{Q}_{t-1}(\mathrm{d}x_t)$$

$$\mathbb{Q}_t^N(\mathrm{d}x_t) = \sum_{n=1}^{N} W_t^n \delta_{X_t^n} \quad \text{approximates } \mathbb{Q}_t(\mathrm{d}x_t)$$

$$L_t^N = \prod_{s=0}^{t}\left(\frac{1}{N}\sum_{n=1}^{N} w_t^n\right) \quad \text{approximates } L_t$$

Some comments

- by *approximates*, we mean: for any test function φ, the quantity

$$\mathbb{Q}_t^N(\varphi) = \sum_{n=1}^{N} W_t^n \varphi(X_t^n)$$

converges to $\mathbb{Q}_t(\varphi)$ as $N \to +\infty$ (at the standard Monte Carlo rate $\mathcal{O}_P(N^{-1/2})$).

Some comments

- by *approximates*, we mean: for any test function φ, the quantity

$$\mathbb{Q}_t^N(\varphi) = \sum_{n=1}^{N} W_t^n \varphi(X_t^n)$$

converges to $\mathbb{Q}_t(\varphi)$ as $N \to +\infty$ (at the standard Monte Carlo rate $\mathcal{O}_P(N^{-1/2})$).

- complexity is $\mathcal{O}(N)$ per time step.

Section 3

Particle algorithms for a given state-space model

Principle

We now consider a given state-space model:

- with initial law $P_0(\mathrm{d}x_0)$ and Markov kernel $P_t(x_{t-1}, \mathrm{d}x_t)$ for $\{X_t\}$;
- with conditional probability density $f_t(y_t|x_t)$ for $Y_t|X_t$

and discuss how the choice of a particular Feynman-Kac formalism leads to more or less efficient particle algorithms.

The bootstrap filter

Bootstrap Feynman-Kac formalism:

$$M_t(x_{t-1}, \mathrm{d}x_t) = P_t(x_{t-1}, \mathrm{d}x_t), \quad G_t(x_{t-1}, x_t) = f_t(y_t|x_t)$$

then $\mathbb{Q}_t$ is the filtering distribution, L_t is the likelihood of $y_{0:t}$, and so on.

The resulting algorithm is called the **bootstrap filter**, and is particularly simple to interpret: we sample particles from Markov transition $P_t(x_{t-1}, \mathrm{d}x_t)$, and we reweight particles according to how compatible they are with the data.

The bootstrap filter: algorithm

All operations to be performed for all $n \in 1:N$.

At time 0:

(a) Generate $X_0^n \sim P_0(\mathrm{d}x_0)$.

(b) Compute $w_0^n = f_0(y_0|X_0^n)$ and $W_0^n = w_0^n / \sum_{m=1}^N w_0^m$.

Recursively, for $t = 1, \ldots, T$:

(a) Generate ancestor variables $A_t^n \in 1:N$ independently from $\mathcal{M}(W_{t-1}^{1:N})$.

(b) Generate $X_t^n \sim P_t(X_{t-1}^{A_t^n}, \mathrm{d}x_t)$.

(c) Compute $w_t^n = f_t(y_t|X_t^n)$ and $W_t^n = w_t^n / \sum_{m=1}^N w_t^m$.

The bootstrap filter: output

$$\frac{1}{N}\sum_{n=1}^{N}\varphi(X_t^n) \quad \text{approximates } \mathbb{E}[\varphi(X_t)|Y_{0:t-1}=y_{0:t-1}]$$

$$\sum_{n=1}^{N}W_t^n\varphi(X_t^n) \quad \text{approximates } \mathbb{E}[\varphi(X_t)|Y_{0:t}=y_{0:t}]$$

$$L_t^N=\prod_{s=0}^{t}\left(\frac{1}{N}\sum_{n=1}^{N}w_t^n\right) \quad \text{approximates } p(y_{0:t})$$

The bootstrap filter: pros and cons

Pros:

- particularly simple
- does not require to compute the density $X_t|X_{t-1}$: we can apply it to models with **intractable dynamics**

Cons:

- We simulate particles **blindly**: if $Y_t|X_t$ is very informative, few particles will get a non-negligible weight.

The guided PF

Guided Feynman-Kac formalism: M_t is a user-chosen **proposal** kernel such that $M_t(x_{t-1}, \mathrm{d}x_t)$ dominates $P_t(x_{t-1}, \mathrm{d}x_t)$, and

$$G_t(x_{t-1}, x_t) = \frac{f_t(y_t|x_t) P_t(x_{t-1}, \mathrm{d}x_t)}{M_t(x_{t-1}, \mathrm{d}x_t)} \qquad (1)$$
$$= \frac{f_t(y_t|x_t) p_t(x_t|x_{t-1})}{m_t(x_t|x_{t-1})}$$

(assuming in the second line that both kernels admit a density wrt a common measure). We still have that $\mathbb{Q}_t(\mathrm{d}x_t)$ is the filtering distribution, and L_t is the likelihood.

We call the resulting algorithm the **guided particle filter**, as in practice we would like to choose M_t so as to **guide** particles to regions of high likelihood.

The guided PF: choice of M_t (local optimality)

Suppose that (G_s, M_s) have been chosen to satisfy (1) for $s \leq t-1$. Among all pairs (M_t, G_t) that satisfy (1), the Markov kernel

$$M_t^{\mathrm{opt}}(x_{t-1}, \mathrm{d}x_t) \propto f_t(y_t|x_t) P_t(x_{t-1}, \mathrm{d}x_t)$$

minimises the variance of the weights, $\mathrm{Var}\left[G_t(X_{t-1}^{A_t^n}, X_t^n)\right]$.

Interpretation and discussion of this result

- M_t^{opt} is simply the law of X_t given X_{t-1} and Y_t. In a sense it is the perfect compromise between the information brought by $P_t(x_{t-1}, \mathrm{d}x_t)$ and by $f_t(y_t|x_t)$.
- In most practical cases, M_t^{opt} is not tractable, hence this result is mostly indicative (on how to choose M_t).
- Note also that the local optimality criterion is debatable. For instance, we do not consider the effect of *future* datapoints.

First example: Gaussian / Gaussian

Assume:

$$X_t|X_{t-1} = x_{t-1} \sim N\left(h(x_{t-1}), \sigma_X^2\right)$$
$$Y_t|X_t = x_t \sim N(x_t, \sigma_Y^2)$$

Then

$$X_t|X_{t-1} = x_{t-1}, Y_t = y_t \sim N\left(v \times \left\{\frac{h(x_{t-1})}{\sigma_X^2} + \frac{y_t}{\sigma_Y^2}\right\}, v\right)$$

with $\frac{1}{v} = \frac{1}{\sigma_X^2} + \frac{1}{\sigma_Y^2}$.

First example: Gaussian / Gaussian

Assume:

$$X_t|X_{t-1} = x_{t-1} \sim N\left(h(x_{t-1}), \sigma_X^2\right)$$
$$Y_t|X_t = x_t \sim N(x_t, \sigma_Y^2)$$

Then

$$X_t|X_{t-1} = x_{t-1}, Y_t = y_t \sim N\left(v \times \left\{\frac{h(x_{t-1})}{\sigma_X^2} + \frac{y_t}{\sigma_Y^2}\right\}, v\right)$$

with $\frac{1}{v} = \frac{1}{\sigma_X^2} + \frac{1}{\sigma_Y^2}$.

Useful e.g. for Ricker Model in Ecology (non-linear h).

Second example: stochastic volatility

There, the log-density of $X_t|X_{t-1}, Y_t$ is (up to a constant):

$$-\frac{1}{2\sigma^2}\left\{x_t - \mu - \phi(x_{t-1} - \mu)\right\}^2 - \frac{x_t}{2} - \frac{e^{-x_t}}{2}y_t^2$$

We can use $e^{x-x_0} \approx 1 + (x - x_0) + (x - x_0)^2/2$ to get a Gaussian approximation.

Third example: bearings-only tracking

In that case, $P_t(x_{t-1}, \mathrm{d}x_t)$ imposes deterministic constraints:

$$X_t(k) = X_{t-1}(k) + X_{t-1}(k+2), \quad k = 1, 2$$

We can choose a M_t that imposes the same constraints. However, in this case, we find that

$$M_t^{\mathrm{opt}}(x_{t-1}, \mathrm{d}x_t) = P_t(x_{t-1}, \mathrm{d}x_t).$$

Discuss.

Guided particle filter pros and cons

Pro:

- may work much better that bootstrap filter when $Y_t|X_t$ is informative (provided we are able to derive a good proposal).

Cons:

- requires to be able to compute density $p_t(x_t|x_{t-1})$.
- sometimes local optimality criterion is not so sound.

The auxiliary particle filter

In the auxiliary Feynman-Kac formalism, an extra degree of freedom is gained by introducing **auxiliary** function η_t, and set:

$$G_0(x_0) = f_0(y_0|x_0)\frac{P_0(\mathrm{d}x_0)}{M_0(\mathrm{d}x_0)}\eta_0(x_0),$$

$$G_t(x_{t-1}, x_t) = f_t(y_t|x_t)\frac{P_t(x_{t-1}, \mathrm{d}x_t)}{M_t(x_{t-1}, \mathrm{d}x_t)}\frac{\eta_t(x_t)}{\eta_{t-1}(x_{t-1})}\,.$$

so that

$$\mathbb{Q}_t(\mathrm{d}x_{0:t}) \propto \mathbb{P}(\mathrm{d}x_{0:t}|Y_{0:t} = y_{0:t})\eta_t(x_t)$$

and we recover the filtering distribution by reweighting by $1/\eta_t$.

Idea: choose η_t so that G_t is as constant as possible.

Output of APF

Let $\tilde{w}_t^n := w_t^n / \eta_t(X_t^n)$, $\tilde{W}_t^n := \tilde{w}_t^n / \sum_{m=1}^N \tilde{w}_t^m$, then

$$\frac{1}{\sum_{m=1}^N \frac{\tilde{W}_t^m}{f(y_t|X_t^m)}} \sum_{n=1}^N \frac{\tilde{W}_t^n}{f_t(y_t|X_t^n)} \varphi(X_t^n) \quad \text{approx.} \quad \mathbb{E}[\varphi(X_t)|Y_{0:t-1} = y_{0:t-1}]$$

$$\sum_{n=1}^N \tilde{W}_t^n \varphi(X_t^n) \quad \text{approx.} \quad \mathbb{E}[\varphi(X_t)|Y_{0:t} = y_{0:t}]$$

$$L_t^N \times N^{-1} \sum_{n=1}^N \tilde{w}_t^n \quad \text{approx.} \quad p(y_{0:t})$$

Local optimality for M_t and η_t

For a given state-space model, suppose that (G_s, M_s) have been chosen to satisfy (1) for $s \leq t-2$, and M_{t-1} has also been chosen. Among all pairs (M_t, G_t) that satisfy (1) and functions η_{t-1}, the Markov kernel

$$M_t^{\mathrm{opt}}(x_{t-1}, \mathrm{d}x_t) = \frac{f_t(y_t|x_t)}{\int_{\mathcal{X}} f(y_t|x')\, P_t(x_{t-1}, \mathrm{d}x')} P_t(x_{t-1}, \mathrm{d}x_t)$$

and the function

$$\eta_{t-1}^{\mathrm{opt}}(x_{t-1}) = \int_{\mathcal{X}} f(y_t|x')\, P_t(x_{t-1}, \mathrm{d}x')$$

minimise $\mathrm{Var}\left[G_t(X_{t-1}^{A_t^n}, X_t^n)/\eta_t(X_t^n)\right]$.

Interpretation and discussion

- We find again that the optimal proposal is the law of X_t given X_{t-1} and Y_t. In addition, the optimal auxiliary function is the probability density of Y_t given X_{t-1}.
- For this ideal algorithm, we would have

$$G_t(x_{t-1}, x_t) = \eta_t^{\mathrm{opt}}(x_t);$$

the density of Y_{t+1} given $X_t = x_t$; not constant, but intuitively less variable than $f_t(y_t|x_t)$ (as in the bootstrap filter).

Example: stochastic volatility

We use the same ideas as for the guided PF: Taylor expansion of log-density, then we integrate wrt x_t.

APF pros and cons

Pros:

- usually gives some extra performance.

Cons:

- a bit difficult to interpret and use;
- they are some (contrived) examples where the auxiliary particle filter actually performs worse than the bootstrap filter.

Note on the generality of APF

From the previous descriptions, we see that:

- the guided PF is a particular instance of the auxiliary particle filter (take $\eta_t = 1$);
- the bootstrap filter is a particular instance of the guided PF(take $M_t = P_t$).

This is why some recent papers focus on the APF.

Which resampling to use in practice?

- Systematic resampling is fast, easy to implement, and seems to work best; but no supporting theory.
- We **do** have some theoretical results regarding the fact that multinomial resampling is dominated by most other resampling schemes. (So don't use it!)
- On the other hand, multinomial resampling is easier to study formally (because again it is based on IID sampling).

Section 4

When to resample?

Resampling or not resampling, that is the question

For the moment, we resample every time. When we introduced resampling, we explained that the decision to resample was based on a trade-off: adding noise at time $t-1$, while hopefully reducing noise at time t (assuming that $\{X_t\}$ forgets its past).

Resampling or not resampling, that is the question

For the moment, we resample every time. When we introduced resampling, we explained that the decision to resample was based on a trade-off: adding noise at time $t-1$, while hopefully reducing noise at time t (assuming that $\{X_t\}$ forgets its past).

We do know that never resample would be a bad idea: consider $M_t(x_{t-1}, \mathrm{d}x_t)$ defined such that the X_t are IID $\mathcal{N}(0,1)$, $G_t(x_t) = \mathbb{1}(x_t > 0)$. (More generally, recall the curse of dimensionality of importance sampling.)

The ESS recipe

Trigger the resampling step whenever the variability of the weights is too large, as measured by e.g. the ESS (effective sample size):

$$\mathrm{ESS}(W_t^{1:N}) := \frac{1}{\sum_{n=1}^N (W_t^n)^2} = \frac{\{\sum_{n=1}^N w_t(X^n)\}^2}{\sum_{n=1}^N w_t(X^n)^2}.$$

Recall that $\mathrm{ESS}(W_t^{1:N}) \in [1, N]$, and that if k weights equal one, and $N - k$ weights equal zero, then $\mathrm{ESS}(W_t^{1:N}) = k$.

PF with adaptive resampling

(Same operations at $t = 0$.)

Recursively, for $t = 1, \ldots, T$:

(a) **If** $\mathrm{ESS}(W_{t-1}^{1:N}) < \gamma N$

generate ancestor variables $A_t^{1:N}$ from resampling distribution $\mathcal{RS}(W_{t-1}^{1:N})$, and set $\hat{W}_{t-1}^n = 1/N$;

Else (no resampling)

set $A_t^n = n$ and $\hat{W}_{t-1}^n = W_{t-1}^n$

(b) Generate $X_t^n \sim M_t(X_{t-1}^{A_t^n}, \mathrm{d}x_t)$.

(c) Compute $w_t^n = (N\hat{W}_{t-1}^n) \times G_t(X_{t-1}^{A_t^n}, X_t^n)$, $L_t^N = L_{t-1}^N \{N^{-1} \sum_{n=1}^N w_t^n\}$, $W_t^n = w_t^n / \sum_{m=1}^N w_t^m$.

Section 5

Numerical experiments

Linear Gaussian example

$$X_t = \rho X_{t-1} + \sigma_X U_t$$
$$Y_t = X_t + \sigma_Y V_t$$

with $\rho = 0.9$, $\sigma_X = 1$, $\sigma_Y = 0.2$.

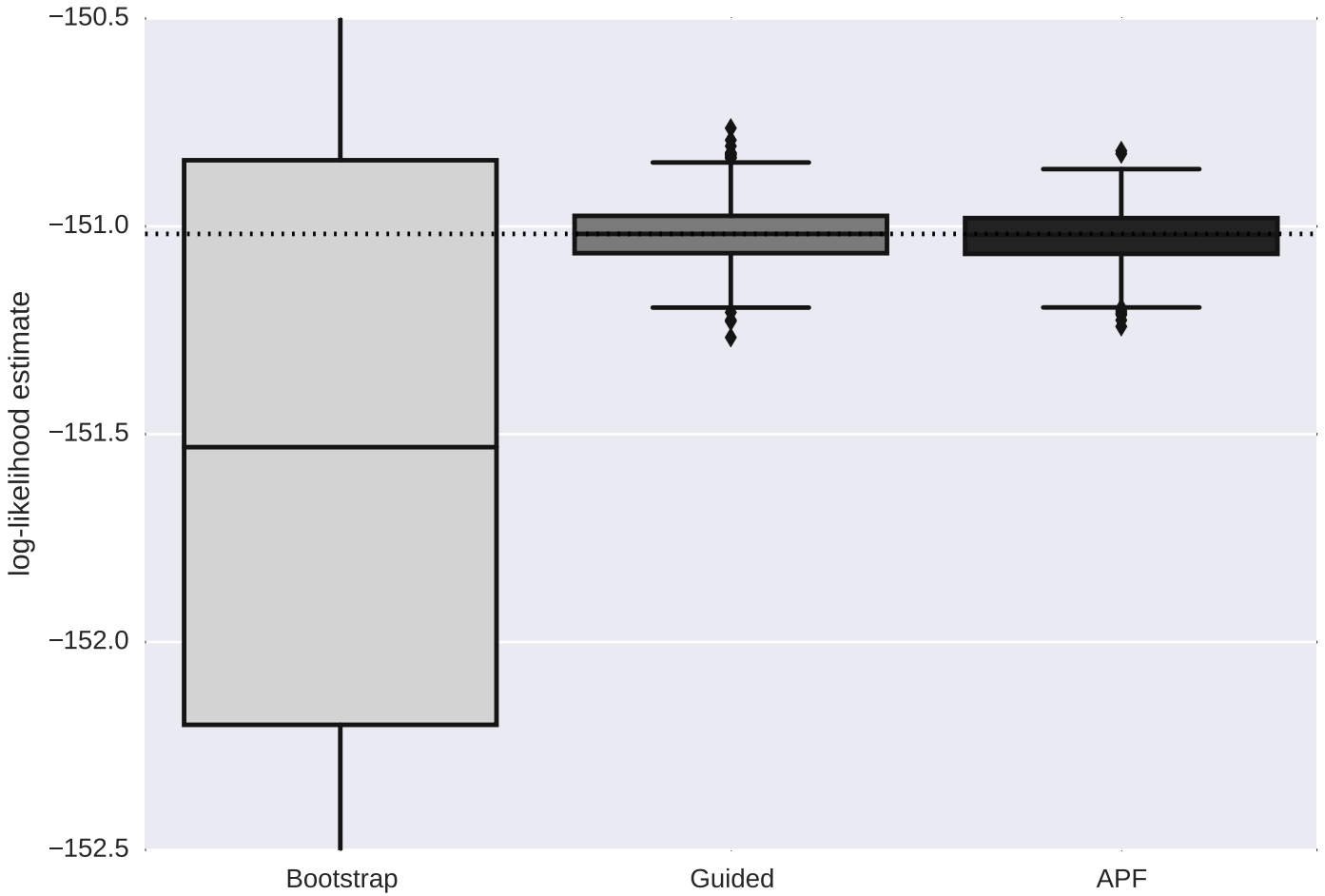
Linear Gaussian example

$$X_t = \rho X_{t-1} + \sigma_X U_t$$
$$Y_t = X_t + \sigma_Y V_t$$

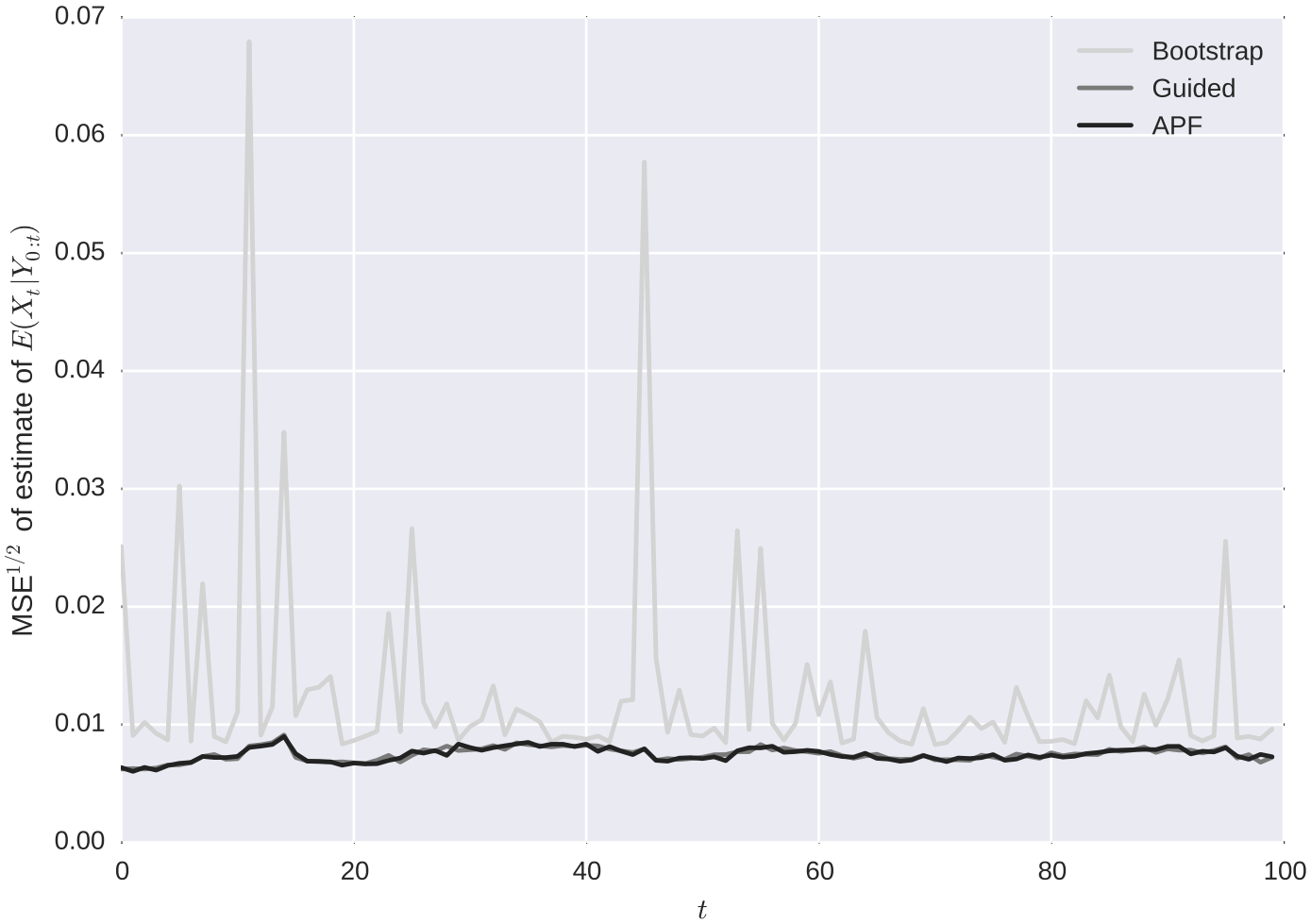
with $\rho = 0.9$, $\sigma_X = 1$, $\sigma_Y = 0.2$.

We can implement the perfect guided filter and the perfect APF.

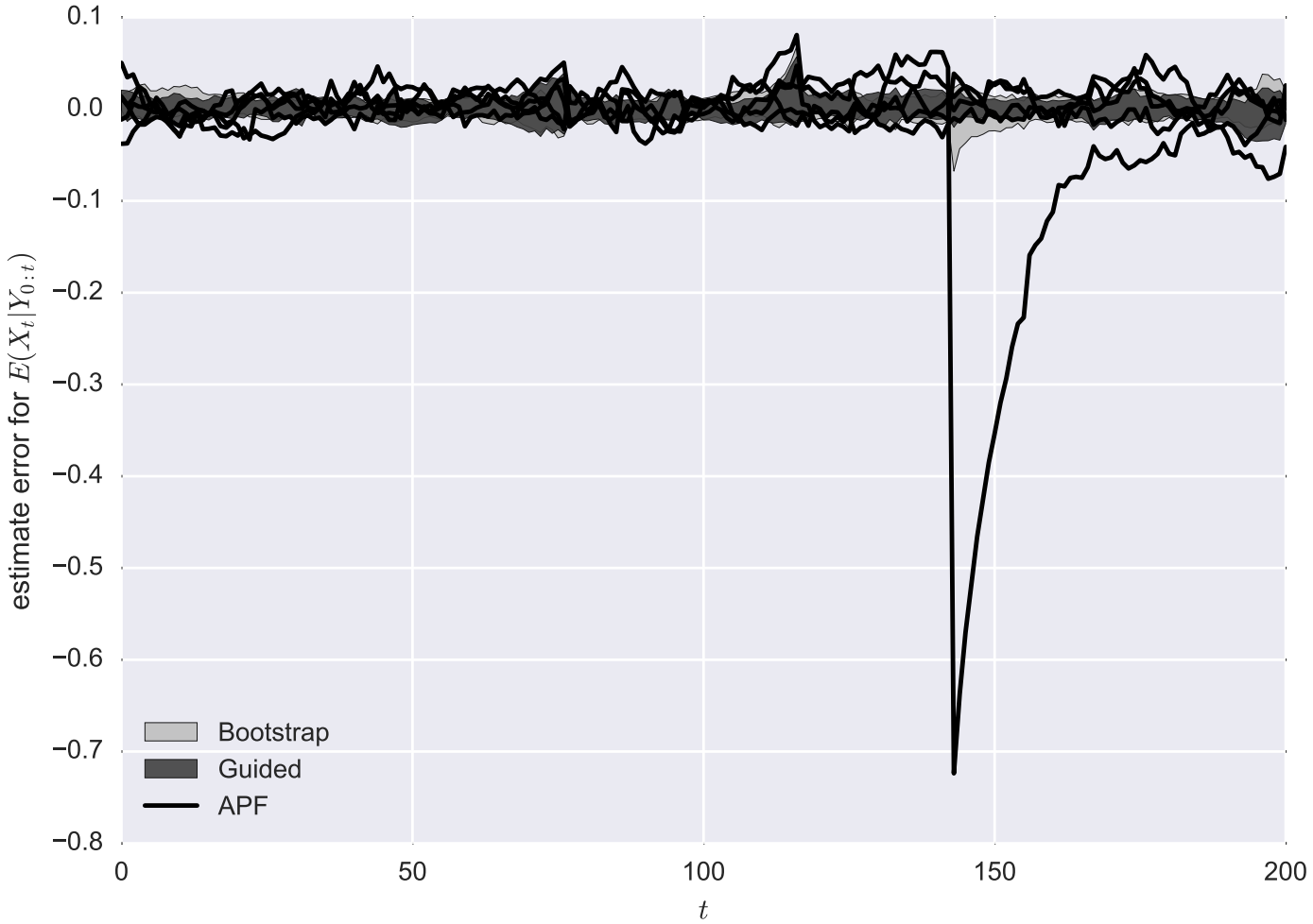
Likelihood


−150.5
−151.0
−151.5
−152.0
−152.5
log-likelihood estimate
Bootstrap
Guided
APF

Filtering


0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
0
20
40
60
80
100
t
$\mathrm{MSE}^{1/2}$ of estimate of $E(X_t|Y_{0:t})$
Bootstrap
Guided
APF

Stochastic volatility



Particle smoothing

nicolas.chopin@ensae.fr

Section 1

Introduction

Objective

transform/extend particle *filtering* algorithms so as to approximate the smoothing distribution $\mathbb{P}_t(\mathrm{d}x_{0:t}|Y_{0:t} = y_{0:t})$ for a given state-space model.

Distinctions

- on-line vs off-line smoothing: recursively, at each time t; or only at some final time T. . . .
- fixed lag vs complete smoothing: recover the law of $X_{t-h:t}$ versus the law of the complete trajectory $X_{0:t}$ (in both cases, given the data $y_{0:t}$.

Distinctions

- on-line vs off-line smoothing: recursively, at each time t; or only at some final time T. . . .
- fixed lag vs complete smoothing: recover the law of $X_{t-h:t}$ versus the law of the complete trajectory $X_{0:t}$ (in both cases, given the data $y_{0:t}$.
- class of test functions? some algorithms will apply only to *additive* functions.

An important motivation

Assuming densities for process $\{X_t\}$, the score can be expressed as the smoothing expectation of an additive function:

$$\frac{\partial}{\partial \theta} \log p_T^\theta(y_{0:T}) = \mathbb{E}^\theta \left[\varphi_T(X_{0:T}) \right] | \, Y_{0:T} = y_{0:T}]$$

with

$$\varphi_T(x_{0:T}) = \frac{\partial}{\partial \theta} \left\{ \log p_0^\theta(x_0) + \sum_{t=1}^{T} \log p_t^\theta(x_t | x_{t-1}) + \sum_{t=0}^{T} \log f_t^\theta(y_t | x_t) \right\}.$$

Important requirement

Most smoothing algorithms will require the Markov kernel $P_t(x_{t-1}, \mathrm{d}x_t)$

1. to admit a probability density $p_t(x_t|x_{t-1})$ (with respect to a fixed measure)
2. such that this PDF is computable for any x_{t-1}, x_t.

Three classes of algorithms

1. forward-only (on-line) smoothing

Three classes of algorithms

1. forward-only (on-line) smoothing
2. Backward sampling (a.k.a. FFBS for forward filtering, backward sampling; off-line)

Three classes of algorithms

1. forward-only (on-line) smoothing
2. Backward sampling (a.k.a. FFBS for forward filtering, backward sampling; off-line)
3. Two-filter smoothing (off-line)

Three classes of algorithms

1. forward-only (on-line) smoothing
2. Backward sampling (a.k.a. FFBS for forward filtering, backward sampling; off-line)
3. Two-filter smoothing (off-line)

Section 2

Forward-only smoothing

$\mathcal{O}(N)$ forward-only smoothing

Simplest approach: we carry forward $X_{t-h:t}$ (fixed-lag), or $X_{0:t}$ within our particle filtering algorithm.

$\mathcal{O}(N)$ forward-only smoothing

Simplest approach: we carry forward $X_{t-h:t}$ (fixed-lag), or $X_{0:t}$ within our particle filtering algorithm.

Pros: simple, complexity is $\mathcal{O}(N)$.

Con: degeneracy.

Degeneracy

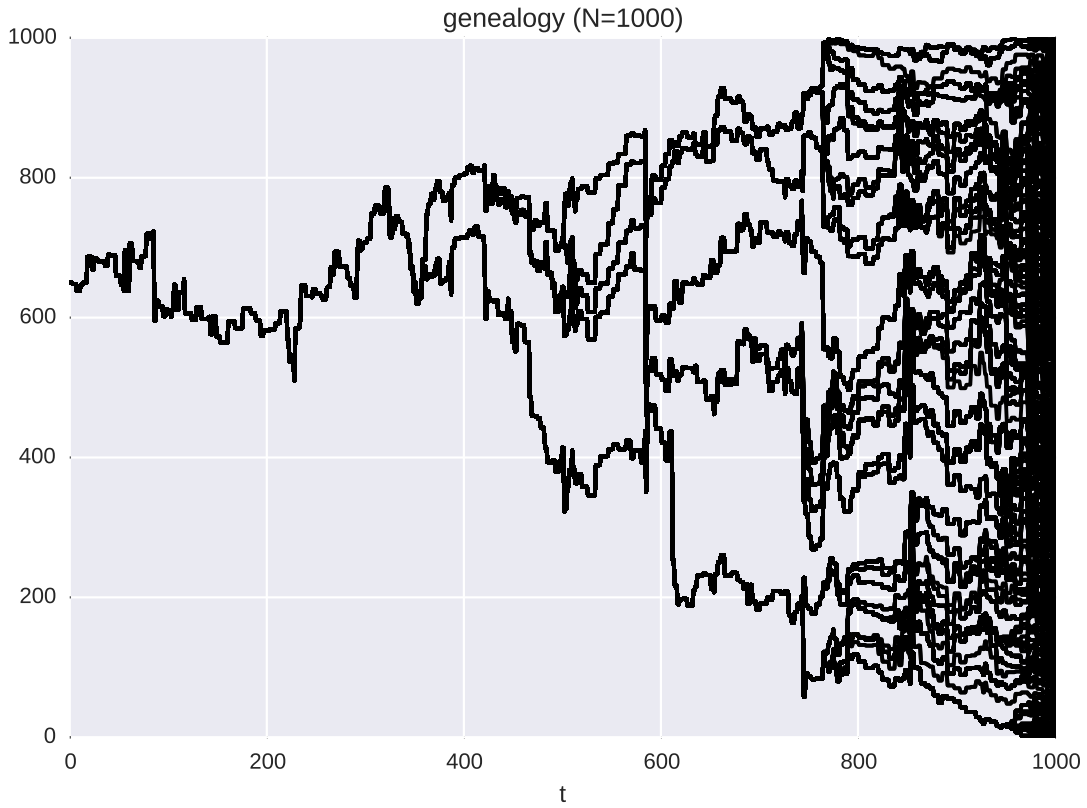


Figure 1: Genealogy of a single run of the bootstrap filter for model: $Y_t|X_t \sim \text{Poisson}(e^{X_t})$, $X_t|X_{t-1} \sim N(\mu + \rho(X_{t-1} - \mu), \sigma^2)$.

$\mathcal{O}(N^2)$ forward-only smoothing for additive functions

For $\varphi_t(x_{0:t}) = \psi_0(x_0) + \sum_{s=1}^{t} \psi_s(x_{s-1}, x_s)$ we have:

Proposition

For $t \geq 0$, let

$$\Phi_t(x_t) := \mathbb{E}[\varphi_t(X_{0:t}) | X_t = x_t, Y_{0:t} = y_{0:t}],$$

then

$$\mathbb{E}[\varphi_t(X_{0:t}) | Y_{0:t} = y_{0:t}] = \mathbb{E}[\Phi_t(X_t) | Y_{0:t} = y_{0:t}]$$

and the Φ_t's may be computed recursively as: $\Phi_0(x_0) = \psi_0(x_0)$,

$$\Phi_t(x_t) = \mathbb{E}\left[\Phi_{t-1}(X_{t-1}) + \psi_t(X_{t-1}, x_t) | X_t = x_t, Y_{0:t-1} = y_{0:t-1}\right]$$

for $t>0$.

$\mathcal{O}(N^2)$ forward-only smoothing: algorithm

Algorithm 1: $\mathcal{O}(N^2)$ on-line smoothing (additive functions)

At iteration $t \in 0:T$ of a particle filtering algorithm:

for $n = 1$ **to** N **do**

  if $t = 0$ **then**

    $\Phi_0^n(X_0^n) \leftarrow \psi_0(X_0^n)$

  else

    $$\Phi_t^n(X_t^n) \leftarrow \frac{\sum_{m=1}^N W_{t-1}^m p_t(X_t^n|X_{t-1}^m)\{\Phi_{t-1}^N(X_{t-1}^m)+\psi_t(X_{t-1}^m,X_t^n)\}}{\sum_{m=1}^N W_{t-1}^m p_t(X_t^n|X_{t-1}^m)}$$

return $\sum_{n=1}^N W_t^n \Phi_t^N(X_t^n)$ (as an approximation of $\mathbb{E}[\varphi_t(X_{0:t})|Y_{0:t} = y_{0:t}]$)

Numerical illustration

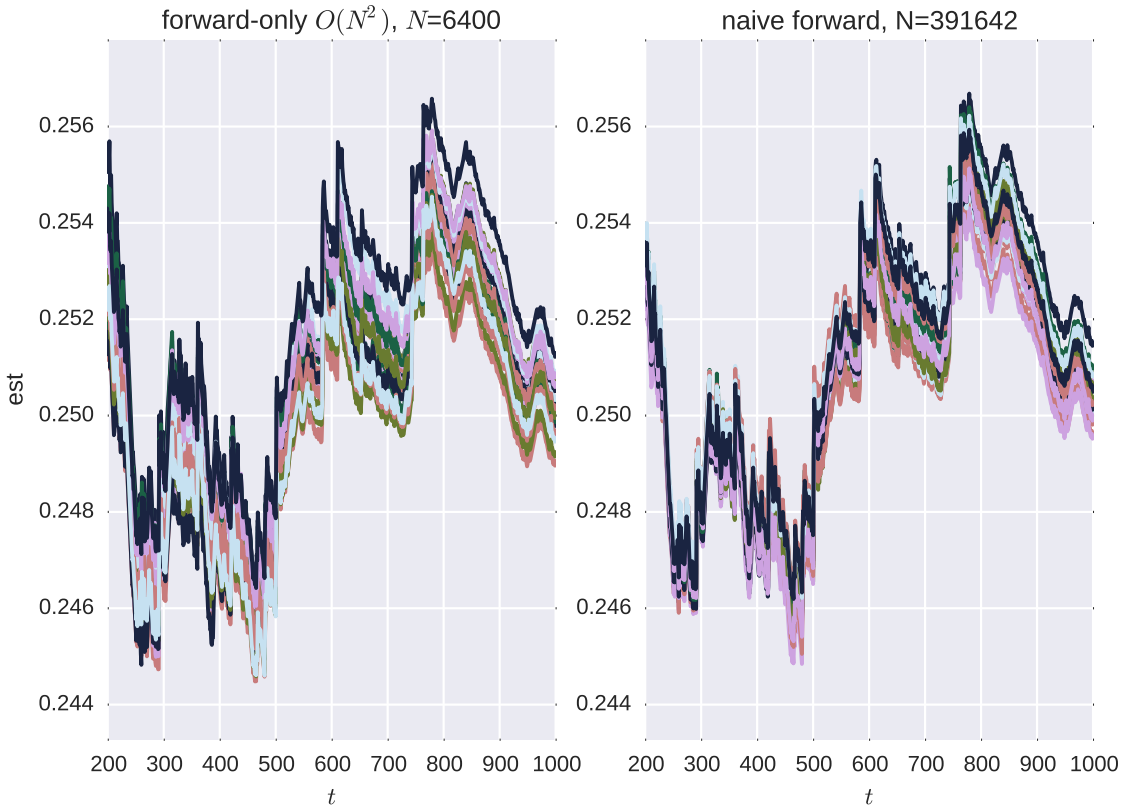


Figure 2: Smoothing expectation vs time, Same model, function φ_t is score wrt σ^2.

Section 3

Backward sampling

Principle

Recall the backward decomposition:

$$\mathbb{P}_T(\mathrm{d}x_{0:T}|Y_{0:T} = y_{0:T}) = \mathbb{P}_T(\mathrm{d}x_T|Y_{0:T} = y_{0:T}) \prod_{t=0}^{T-1} \overleftarrow{P}_{t|t}(x_{t+1}, \mathrm{d}x_t)$$

where $\overleftarrow{P}_{t|t}(x_{t+1}, \mathrm{d}x_t) \propto p_{t+1}(x_{t+1}|x_t)\mathbb{P}(\mathrm{d}x_t|Y_{0:t} = y_{0:t})$

Principle

Recall the backward decomposition:

$$\mathbb{P}_T(\mathrm{d}x_{0:T}|Y_{0:T} = y_{0:T}) = \mathbb{P}_T(\mathrm{d}x_T|Y_{0:T} = y_{0:T}) \prod_{t=0}^{T-1} \overleftarrow{P}_{t|t}(x_{t+1}, \mathrm{d}x_t)$$

where $\overleftarrow{P}_{t|t}(x_{t+1}, \mathrm{d}x_t) \propto p_{t+1}(x_{t+1}|x_t)\mathbb{P}(\mathrm{d}x_t|Y_{0:t} = y_{0:t})$

Idea: plug particle approximation:

$$\sum_{n=1}^{N} W_t^n \delta_{X_t^n} \approx \mathbb{P}_t(\mathrm{d}x_t|y_{0:t}).$$

Smoothing skeleton

$$\mathbb{P}_T^N(\mathrm{d}x_{0:T}|Y_{0:T}=y_{0:T}) := \left\{\sum_{n=1}^N W_T^n \delta_{X_T^n}(\mathrm{d}x_T)\right\} \prod_{t=0}^{T-1} \overleftarrow{P}_{t|T}^N(x_{t+1},\mathrm{d}x_t)$$

with

$$\overleftarrow{P}_{t|T}^N(x_{t+1},\mathrm{d}x_t) \propto \sum_{n=1}^N W_t^N p_{t+1}(x_{t+1}|X_t^n)\delta_{X_t^n}.$$

The skeleton is a discrete distribution, with support of size N^{T+1}.

Sampling from the skeleton

Algorithm 2: FFBS

Input: Output of a particle filter: particles $X_0^{1:N}, \ldots, X_T^{1:N}$, weights $W_0^{1:N}, \ldots, W_T^{1:N}$.

Output: trajectory $(X_0^{B_0}, \ldots, X_T^{B_T})$.

$B_T \sim \mathcal{M}(W_T^{1:N})$

for $t = (T-1)$ **to** 0 **do**

$\widehat{w}_t^n \leftarrow W_t^n p_{t+1}(X_{t+1}^{B_{t+1}} | X_t^n)$ and $\widehat{W}_t^n = \widehat{w}_t^n / \sum_{m=1}^N \widehat{w}_t^m$ for $n = 1, \ldots, N$

$B_t \sim \mathcal{M}(\widehat{W}_t^{1:N})$

Notes

- cost of simulating **one** trajectory is $\mathcal{O}(TN)$.

Notes

- cost of simulating **one** trajectory is $\mathcal{O}(TN)$.
- In practice, we sample M times from the skeleton, and compute the following estimates

$$\frac{1}{M}\sum_{m=1}^{M}\varphi(\tilde{X}_{0:T}^{m}) \approx \mathbb{P}_T^N\left(\varphi(X_{0:T})|Y_{0:T}=y_{0:T}\right)$$

$$\approx \mathbb{E}[\varphi(X_{0:T})|Y_{0:T}=y_{0:T}]$$

Notes

- cost of simulating **one** trajectory is $\mathcal{O}(TN)$.
- In practice, we sample M times from the skeleton, and compute the following estimates

$$\frac{1}{M}\sum_{m=1}^{M}\varphi(\tilde{X}_{0:T}^{m}) \approx \mathbb{P}_T^N\left(\varphi(X_{0:T})|Y_{0:T}=y_{0:T}\right)$$
$$\approx \mathbb{E}[\varphi(X_{0:T})|Y_{0:T}=y_{0:T}]$$

- Often, people take $M=N$, so $\mathcal{O}(N^2)$ complexity.

FFBS-reject

If we know some constant C_t such that $p_t(x_t|x_{t-1}) \leq C_t$, then we may use rejection to sample the B_t^n's: sample $B_t \sim \mathcal{M}(W_t^{1:N})$, accept with probability $p_{t+1}(X_{t+1}^{B_{t+1}}|X_t^{B_t})/C_t$.

FFBS-reject

If we know some constant C_t such that $p_t(x_t|x_{t-1}) \leq C_t$, then we may use rejection to sample the B_t^n's: sample $B_t \sim \mathcal{M}(W_t^{1:N})$, accept with probability $p_{t+1}(X_{t+1}^{B_{t+1}}|X_t^{B_t})/C_t$.

It has been claimed that such an approach has $\mathcal{O}(N)$ complexity. However, its running time is **random**, and may have infinite expectation or variance (behaves like a mixture of Geometric distributions).

FFBS-reject

If we know some constant C_t such that $p_t(x_t|x_{t-1}) \leq C_t$, then we may use rejection to sample the B_t^n's: sample $B_t \sim \mathcal{M}(W_t^{1:N})$, accept with probability $p_{t+1}(X_{t+1}^{B_{t+1}}|X_t^{B_t})/C_t$.

It has been claimed that such an approach has $\mathcal{O}(N)$ complexity. However, its running time is **random**, and may have infinite expectation or variance (behaves like a mixture of Geometric distributions).

Also, rejection is hard to implement efficiently in Python/R.

FFBS-MCMC

- Choose randomly one of the existing trajectories: $B_T \sim \mathcal{M}(W_T^{1:N})$, $B_t \leftarrow A_{t+1}^{B_{t+1}}$ (recursively).
- Apply a **single** Metropolis step to this trajectory, with proposal $\mathcal{M}(W_t^{1:N})$ at iteration t.

FFBS-MCMC

- Choose randomly one of the existing trajectories: $B_T \sim \mathcal{M}(W_T^{1:N})$, $B_t \leftarrow A_{t+1}^{B_{t+1}}$ (recursively).
- Apply a **single** Metropolis step to this trajectory, with proposal $\mathcal{M}(W_t^{1:N})$ at iteration t.

This leads to an algorithm with **deterministic**, $\mathcal{O}(N)$ complexity; see Dau & C. (2023, Annals of Statistics) for details.

Connection between FFBS and $\mathcal{O}(N^2)$ forward smoothing

The $\mathcal{O}(N^2)$ forward smoothing algorithm for additive functions φ_t we presented before computes **exactly** the expectation of φ_t w.r.t. the skeleton.

Section 4

Two-filter smoothing

Two-filter smoothing: basic identity

Recall (FK chapter):

$$\mathbb{P}(X_t \in \mathrm{d}x_t | Y_{0:T} = y_{0:T}) \propto p_T(y_{t+1:T}|x_t)\mathbb{P}(X_t \in \mathrm{d}x_t | Y_{0:t} = y_{0:t}).$$

We can approximate $\mathbb{P}(X_t \in \mathrm{d}x_t | Y_{0:t} = y_{0:t})$ with a **forward** particle filter, but what about $p_T(y_{t+1:T}|x_t)$?

Two-filter smoothing: backward recursion

In the FK chapter, we also derived the following recursion (taking $p_T(y_{T+1:T}|x_T) = 1$):

$$p_T(y_{t+1:T}|x_t) = \int_{\mathcal{X}} f_{t+1}(y_{t+1}|x_{t+1}) p_T(y_{t+2:T}|x_{t+1}) P_{t+1}(x_t, \mathrm{d}x_{t+1}) .$$

which looks suspiciously similar to the forward recursion of Feynman-Kac models.

Idea: run a **backward** particle algorithm to recursively approximate this quantity.

Information filter

Note that $p_T(y_{t+1:T}|x_t)$ is **not** (necessarily) proportional to a PDF. Hence we introduce some (user-chosen) distribution $\gamma_t(\mathrm{d}x_t)$ dist' and tracks the sequence

$$\gamma_{t|T}(\mathrm{d}x_t) \propto \gamma_t(\mathrm{d}x_t) p_T(y_{t+1:T}|x_t).$$

Problem: how to choose the γ_t's for *best* performance?

Two-filter estimate

To approximate the smoothing expectation of function $\varphi_{t+1}(X_t, X_{t+1})$, plug the two particle approximations in the two-filter identity:

$$\frac{1}{\sum_{m,n=1}^{N} \omega_t^{m,n}} \sum_{m,n=1}^{N} \omega_t^{m,n} \varphi_{t+1}(\overleftarrow{X}_{t+1}^m, X_t^n)$$

with $\omega_t^{m,n} = W_t^n \overleftarrow{W}_t^m / \gamma_t(\overleftarrow{X}_t^m)$, and

$$\sum_{n=1}^{N} W_t^n \delta_{X_t^n}(dx_t) \approx \mathbb{P}_t(\mathrm{d}x_t | y_{0:t}), \quad \text{forward filter,}$$

$$\sum_{m=1}^{N} \overleftarrow{W}_{t+1}^m \delta_{\overleftarrow{X}_{t+1}^m}(dx_{t+1}) \approx \gamma_{t+1|T}(\mathrm{d}x_{t+1}), \quad \text{backward information filter.}$$

Complexity

Cost to compute previous estimate is in general $\mathcal{O}(N^2)$. There exists a recent method to obtain a $\mathcal{O}(N)$ complexity.

Section 5

Conclusion

Practical recommendations

- Off-line, known transition density: FFBS-MCMC

Practical recommendations

- Off-line, known transition density: FFBS-MCMC
- Off-line, unknown transition density: see Dau & C (2023) for a coupling approach.

Practical recommendations

- Off-line, known transition density: FFBS-MCMC
- Off-line, unknown transition density: see Dau & C (2023) for a coupling approach.
- On-line, additive functions: the $\mathcal{O}(N^2)$ algorithm (expensive).

References

- Dau, H.-D., and Chopin, N. (2023) On backward smoothing algorithms, Annals of Statistics.

Maximum likelihood estimation

nicolas.chopin@ensae.fr

Section 1

Main ideas

Maximum likelihood estimation

Now, the considered state-space model depends on some unknown parameter θ; dependence on θ is made explicit in the notations.

We'd like to compute:

$$\widehat{\theta}_T \in \arg\max_{\theta \in \Theta} p_T^\theta(y_{0:T}).$$

Specific difficulties

- Asymptotic theory (for state-space models) is very technical, and relies on strong assumptions.
- The likelihood function is typically not well-behaved.
- The likelihood function (and related quantities) are not tractable.

Log-likelihood plot

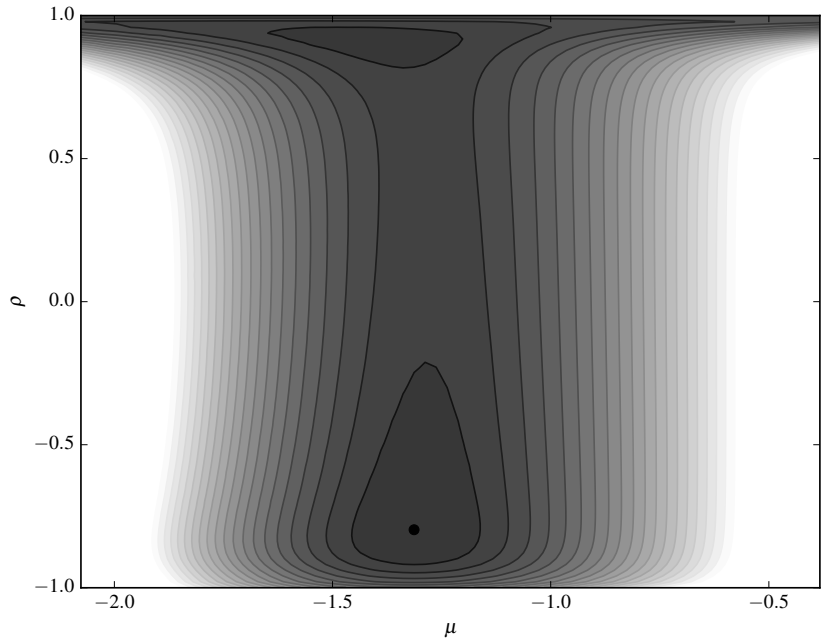


Figure 1: log-likelihood function of a stochastic volatility model for real-data ($\sigma = 0.178$).

Main approaches

- Gradient-free optimisation (Nelder-Mead);
- Gradient-based optimisation (gradient descent);
- the EM algorithm.

Section 2

Gradient-free optimisation

Likelihood estimate

Recall that, in a guided filter

$$L_T^N = \prod_{t=0}^{T} \left(\frac{1}{N} \sum_{n=1}^{N} w_t^n \right)$$

is an estimate of the density of $Y_{0:T}$ (i.e. the likelihood).

In fact, this estimate is unbiased, and its variance grows with time.

Error grows with time

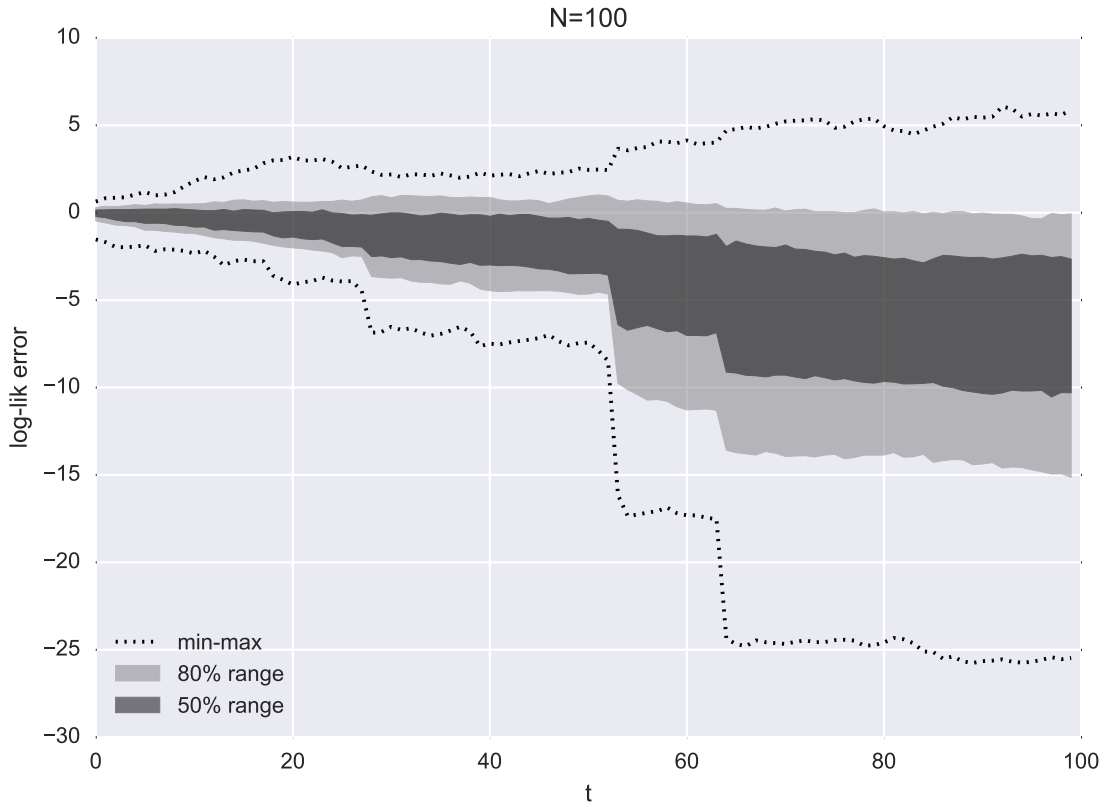


Figure 2: log-likelihood error versus time (bootstrap filter, $N = 100$, linear Gaussian model)

Common random numbers

A nice trick when dealing with noisy optimisation is to "freeze" the random numbers for all the evaluations of the noisy target.

Unfortunately, the CRN trick is not very useful in our context: even with frozen random numbers, a particle estimate is a discontinuous function of θ. (Discuss.)

Hürzeler and Künsch

The CRN would apply nicely to a likelihood estimate based on the following identity:

$$\frac{p_T^\theta(y_{0:T})}{p_T^{\theta_0}(y_{0:T})} = \mathbb{E}_{\mathbb{P}_T^{\theta_0}}\left[\frac{p_T^\theta(X_{0:T}, y_{0:T})}{p_T^{\theta_0}(X_{0:T}, y_{0:T})} \,\middle|\, Y_{0:T} = y_{0:T}\right] \tag{1}$$

where

$$p_T^\theta(x_{0:T}, y_{0:T}) = p_0^\theta(x_0) \prod_{t=1}^{T} p_t^\theta(x_t|x_{t-1}) \prod_{t=0}^{t} f_t^\theta(y_t|x_t)\,.$$

but note that we are then dealing with a curse of dimensionality...

Applying H&K to a stochastic volatility model

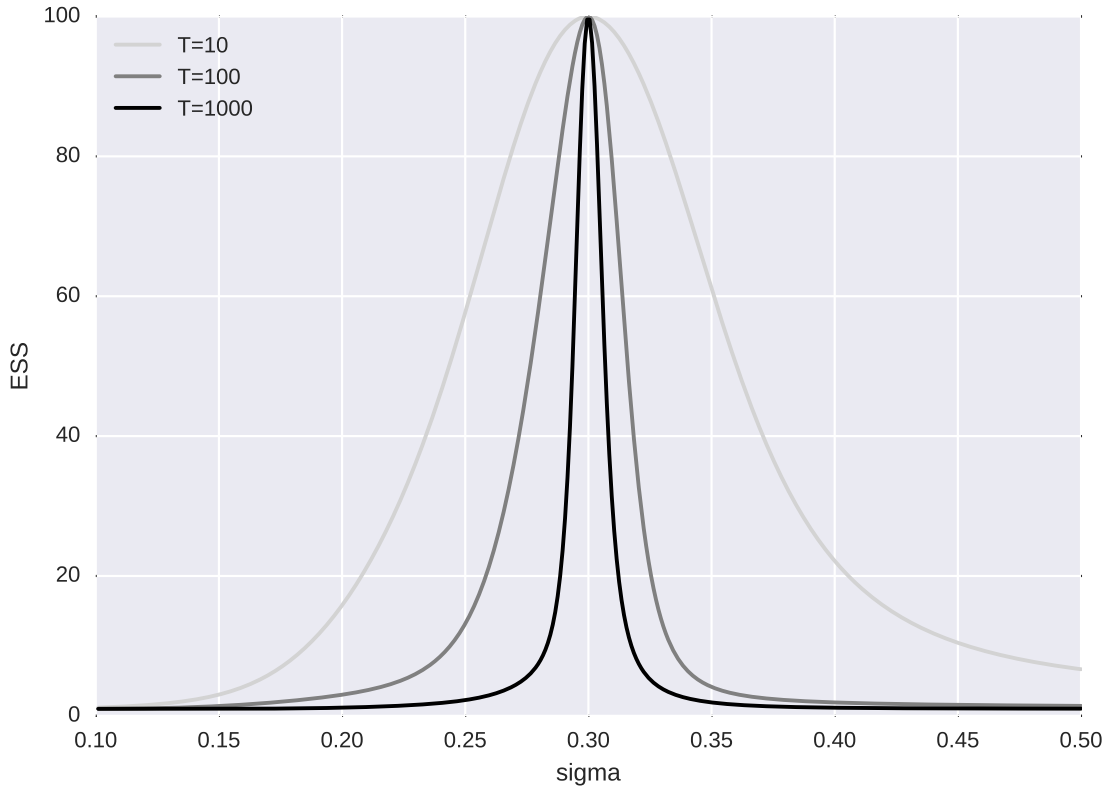


Figure 3: ESS as a function of σ, for the IS estimate of the previous slide

Practical recipe

- Brute force approach: take N large enough so that the noise of likelihood estimates become negligible. Then apply the simplex (Nelder-mead) algorithm.
- As always, the algorithm may converge to a local mode, depending on the starting point.
- Seems to work reasonably well when $\dim(\theta)$ is not too large.

Section 3

Gradient-based approaches

Gradient descent

Gradient *ascent* maximises a function h by iterating:

$$\theta_n = \theta_{n-1} + \gamma_n \nabla h(\theta_{n-1})$$

In our case, recall that we may express the gradient of the log-likelihood as a smoothing expectation (see previous chapter).

Section 4

The EM algorithm

The EM algorithm

For any model on a pair (X, Y) such that we observe only Y, the EM algorithm iterates:

$$\theta_n = \arg\max_{\theta \in \Theta} \mathbb{E}_{\mathbb{P}_{\theta_{n-1}}} \left[\log p^{\theta}(X, y) | Y = y \right]$$

where $p^{\theta}(x, y)$ is the joint density of (X, Y) (for parameter θ).

EM algorithm for exponential families

If the joint density belongs to a natural exponential family:

$$p^{\theta}(x, y) = \exp\{\theta^T s(x, y) - \psi(\theta) - \xi(x)\}$$

the EM update takes the following (simpler) form:

$$\nabla\psi(\theta) = \mathbb{E}_{\mathbb{P}^{\theta_{n-1}}}\left[s(X, y) \,|\, Y = y\right] . \tag{2}$$

EM algorithm for state-space models

There, $X = X_{0:T}$, $Y = Y_{0:T}$, and, assuming again an exponential family for the joint, the EM update amounts to computing a smoothing expectation:

$$\theta_n = (\nabla \psi)^{-1} \left(\mathbb{E}_{\mathbb{P}^{\theta_{n-1}}} \left[s(X_{0:T}, y_{0:T}) \mid Y_{0:T} = y_{0:T} \right] \right)$$

Section 5

Conclusion

Conclusion

- no clear winner;
- most approaches rely on computing smoothing estimates; this implies in particular that kernel $P_t^\theta(x_{t-1}, \mathrm{d}x_t)$ admits a tractable density $p_t^\theta(x_t|x_{t-1})$.
- Maximum likelihood estimation may not be the best way to assess parameter uncertainty in the context of state-space models.

Particles as auxiliary variables: PMCMC and related algorithms

nicolas.chopin@ensae.fr

Particles as auxiliary variables: PMCMC and related algorithms

nicolas.chopin@ensae.fr

Outline

Tractable models

For a standard Bayesian model, defined by (a) prior $p(\theta)$, and (b) likelihood $p(y|\theta)$, a standard approach is to use the Metropolis-Hastings algorithm to sample from the posterior

$$\pi(\theta) \propto p(\theta)p(y|\theta).$$

Metropolis-Hastings (Gaussian random walk proposal)

From current point θ_m

1. Sample $\theta_\star \sim N(\theta_m, \Sigma)$
2. With probability $1 \wedge r$, take $\theta_{m+1} = \theta_\star$, otherwise $\theta_{m+1} = \theta_m$, where

$$r = \frac{p(\theta_\star)p(y|\theta_\star)}{p(\theta_m)p(y|\theta_m)}$$

This generates a Markov chain which leaves $p(\theta|y)$ invariant.

Intractable models

This generic approach cannot be applied in the following situations:

1. The likelihood is $p(y|\theta) = h_\theta(y)/Z(\theta)$, where $Z(\theta)$ is an intractable normalising constant; e.g. log-linear models, network models, Ising models.
2. The likelihood $p(y|\theta)$ is an intractable integral

$$p(y|\theta) = \int_{\mathcal{X}} p(y, x|\theta)\, dx.$$

3. The likelihood is even more complicated, because it corresponds to some scientific model involving some complicate **generative** process (scientific models, "likelihood-free inference", ABC).

Outline

General framework

Consider posterior

$$\pi(\theta, x) \propto p(\theta)p(x|\theta)p(y|x, \theta)$$

where typically x is of much larger dimension than θ.
One potential approach to sample from the posterior is **Gibbs sampling**: iteratively sample $\theta|x, y$, then $x|\theta, y$. However, there are many cases where Gibbs is either difficult to implement, or quite inefficient.
Instead, we would like to sample **marginally** from

$$\pi(\theta) \propto p(\theta)p(y|\theta), \quad p(y|\theta) = \int_{\mathcal{X}} p(x, y|\theta)\, dx$$

but again $p(y|\theta)$ is intractable...

Importance sampling

I cannot compute $p(y|\theta)$, but I can compute an **unbiased** estimator of this quantity:

$$\hat{p}(y|\theta) = \frac{1}{N}\sum_{n=1}^{N} \frac{p(x^n|\theta)p(y|x^n,\theta)}{q(x^n)}, \quad x^{1:N} \overset{iid}{\sim} q(x)$$

using **importance sampling**.

The pseudo-marginal approach

GIMH (Beaumont, 2003)

From current point θ_m

1. Sample $\theta_\star \sim N(\theta_m, \Sigma)$
2. With prob. $1 \wedge r$, take $\theta_{m+1} = \theta_\star$, otherwise $\theta_{m+1} = \theta_m$, with

$$r = \frac{p(\theta_\star)\hat{p}(y|\theta_\star)}{p(\theta_m)\hat{p}(y|\theta_m)}$$

Note that $\hat{p}(y|\theta_\star)$ is based on independent samples generated at iteration m.
Question: Is GIMH a **non-standard** HM sampler w.r.t. **standard** target $\pi(\theta)$?

Validity of GIMH

Property 1

The following function

$$\bar{\pi}(\theta, x^{1:N}) = \prod_{n=1}^{N} q(x^n) \frac{p(\theta)\hat{p}(y|\theta)}{p(y)}$$

is a joint PDF, whose θ-marginal is $\pi(\theta) \propto p(\theta)p(y|\theta)$.

Proof: Direct consequence of unbiasedness; fix θ then

$$\int \prod_{n=1}^{N} q(x^n) p(\theta)\hat{p}(y|\theta)\, dx^{1:N} = p(\theta)\mathbb{E}\left[\hat{p}(y|\theta)\right] = p(\theta)p(y|\theta)$$

GIMH as a Metropolis sampler

Property 2

GIMH is a Metropolis sampler with respect to joint distribution $\bar{\pi}(\theta, x^{1:N})$. The proposal density is $N(\theta_\star; \theta_m, \Sigma) \prod_{n=1}^N q(x_\star^n)$.

Proof: current point is $(\theta_m, x_m^{1:N})$, proposed point is $(\theta_\star, x_\star^{1:N})$ and HM ratio is

$$r = \frac{\cancel{\prod_{n=1}^N q(x_\star^n)} p(\theta_\star) \hat{p}(y|\theta_\star) \cancel{\prod_{n=1}^N q(x_m^n)}}{\cancel{\prod_{n=1}^N q(x_m^n)} p(\theta_m) \hat{p}(y|\theta_m) \cancel{\prod_{n=1}^N q(x_\star^n)}}$$

Thus, GIMH is a **standard** Metropolis sampler w.r.t. **non-standard** (extended) target $\bar{\pi}(\theta, x^{1:N})$.

There is more to life than this

Property 3

Extend $\bar{\pi}(\theta, x^{1:N})$ with $k|\theta, x^{1:N} \propto \pi(\theta, x^k)/q(x^k)$, then,

- the marginal dist. of (θ, x^k) is $\pi(\theta, x)$.
- Conditional on (θ, x^k), $x_n \sim q$ for $n \neq k$, independently.

Proof: let

$$\bar{\pi}(\theta, x^{1:N}, k) = \left\{ \prod_{n=1}^{N} q(x^n) \right\} \frac{\pi(\theta, x^k)}{q(x^k)} = \left\{ \prod_{n \neq k} q(x^n) \right\} \pi(\theta, x^k)$$

then clearly the sum w.r.t. k gives $\bar{\pi}(\theta, x^{1:N})$, while the above properties hold.

We can do Gibbs!

One consequence of Property 3 is that we gain the ability to perform **Gibbs**, in order to regenerate the $N-1$ non-selected points x^n, $n \neq k$. More precisely:

1. Sample $k \sim \pi(k|\theta, x^{1:N}) \propto \pi(\theta, x^k)/q(x^k)$
2. regenerate $x^n \sim q$, for all $n \neq k$.

Could be useful for instance to avoid "getting stuck", because say the current value $\hat{\pi}(\theta)$ is too high.

Main lessons

- We can replace an intractable quantity by an unbiased estimate, **without introducing any approximation**.
- In fact, we can do more: with Proposition 3, we have obtained that
  1. it is possible to sample from $\pi(\theta, x)$ jointly;
  2. it is possible to do a Gibbs step where the $N-1$ x^n, $n \neq k$ are regenerated (useful when GIMH "get stucks"?)
- but careful, it is possible to get it wrong...

Unbiasedness without an auxiliary variable representation

This time, consider instead a target $\pi(\theta)$ (no x), involving an intractable *denominator*; an important application is Bayesian inference on likelihoods with intractable normalising constants:

$$\pi(\theta) \propto p(\theta)p(y|\theta) = p(\theta)\frac{h_\theta(y)}{Z(\theta)}$$

Liang & Lin (2010)'s sampler

From current point θ_m

1. Sample $\theta_\star \sim H(\theta^m, \mathrm{d}\theta_\star)$
2. With prob. $1 \wedge r$, take $\theta_{m+1} = \theta_\star$, otherwise $\theta_{m+1} = \theta_m$, with

$$r = \left(\frac{\widehat{Z(\theta_m)}}{Z(\theta_\star)}\right)\frac{p(\theta_\star)h_{\theta_\star}(y)h(\theta^m|\theta_\star)}{p(\theta_m)h_{\theta_m}(y)h(\theta_\star|\theta^m)}.$$

Outline

PMCMC: introduction

PMCMC (Andrieu et al., 2010) is akin to GIMH, except a more complex proposal mechanism is used: a PF (particle filter).
The same remarks will apply:

- Unbiasedness (of the likelihood estimated provided by the PF) is only an intermediate result for establishing the validity of the whole approach.
- Unbiasedness is not enough to give you intuition on the validity of e.g. Particle Gibbs.

Objective

Objectives

Sample from

$$p(\mathrm{d}\theta, \mathrm{d}x_{0:T}|y_{0:T})$$

for a given state-space model.

Why are these models difficult?

Because the likelihood is intractable

$$p_T^\theta(y_{0:T}) = \int \prod_{t=0}^{T} f_t^\theta(y_t|x_t) \prod_{t=1}^{T} p_t^\theta(x_t|x_{t-1}) p_0^\theta(x_0)$$

Feynman-Kac formalism

Taking $\{M_t^\theta, G_t^\theta\}_{t \geq 0}$ so that

- $M_t^\theta(x_{t-1}, \mathrm{d}x_t)$ is a Markov kernel (for fixed θ), with density $m_t^\theta(x_t|x_{t-1})$
- and

$$G_t^\theta(x_{t-1}, x_t) = \frac{f_t^\theta(y_t|x_t) p_t^\theta(x_t|x_{t-1})}{m_t^\theta(x_t|x_{t-1})}$$

we obtain the Feynman-Kac representation associated to a guided PF that approximates the filtering distribution at every time t.

If we take $m_t^\theta(x_t|x_{t-1}) = p_t^\theta(x_t|x_{t-1})$, we recover the bootstrap filter (which does not require to be able to evaluate $p_t^\theta(x_t|x_{t-1})$ pointwise).

Particle filters: pseudo-code

All operations to be performed for all $n \in 1:N$.
At time 0:

(a) Generate $X_0^n \sim M_0^\theta(\mathrm{d}x_0)$.

(b) Compute $w_0^n = G_0^\theta(X_0^n)$, $W_0^n = w_0^n / \sum_{m=1}^N w_0^m$, and $L_0^N = N^{-1} \sum_{n=1}^N w_0^n$.

Recursively, for $t = 1, \ldots, T$:

(a) Generate ancestor variables $A_t^n \in 1:N$ independently from $\mathcal{M}(W_{t-1}^{1:N})$.

(b) Generate $X_t^n \sim M_t^\theta(X_{t-1}^{A_t^n}, \mathrm{d}x_t)$.

(c) Compute $w_t^n = G_t^\theta(x_{t-1}, x_t)$, $W_t^n = w_t^n / \sum_{m=1}^N w_t^m$, and $L_t^N(\theta) = L_{t-1}^N(\theta) \times \{N^{-1} \sum_{n=1}^N w_t^n\}$.

Unbiased likelihood estimator

A by-product of PF output is that

$$L_T^N(\theta) = \left(\frac{1}{N} \sum_{n=1}^{N} G_0^\theta(X_0^n) \right) \prod_{t=1}^{T} \left(\frac{1}{N} \sum_{n=1}^{N} G_t^\theta(x_{t-1}, x_t) \right)$$

is an **unbiased** estimator of the likelihood $L_T(\theta) = p(y_{0:T}|\theta)$.

(Not trivial, see e.g Proposition 7.4.1 in Pierre Del Moral's book.)

PMCMC

Breakthrough paper of Andrieu et al. (2011), based on the unbiasedness of the PF estimate of the likelihood.

Marginal PMCMC

From current point θ_m (and current PF estimate $L_T^N(\theta_m)$):

1. Sample $\theta_\star \sim H(\theta_m, \mathrm{d}\theta_\star)$
2. Run a PF so as to obtain $L_T^N(\theta_\star)$, an unbiased estimate of $L_T(\theta_\star) = p(y_{0:T}|\theta_\star)$.
3. With probability $1 \wedge r$, set $\theta_{m+1} = \theta_\star$, otherwise $\theta_{m+1} = \theta_m$ with

$$r = \frac{p(\theta_\star) L_T^N(\theta_\star) h(\theta_m|\theta_\star)}{p(\theta_m) L_T^N(\theta_m) h(\theta_\star|\theta_m)}$$

Validity

Property 1

Let $\psi_{T,\theta}(\mathrm{d}x_{0:T}^{1:N}, \mathrm{d}a_{1:T}^{1:N})$ be the joint dist' of all the the rv's generated by a PF (for fixed θ), then

$$\pi_T(\mathrm{d}\theta, \mathrm{d}x_{0:T}^{1:N}, \mathrm{d}a_{1:T}^{1:N}) = \frac{p(\mathrm{d}\theta)}{p(y_{0:T})} \psi_{T,\theta}(\mathrm{d}x_{0:T}^{1:N}, \mathrm{d}a_{1:T}^{1:N}) L_T^N(\theta)$$

is a joint pdf, such that the θ-marginal is $p(\theta|y_{0:T})\mathrm{d}\theta$.

Proof: fix θ, and integrate wrt the other variables:

$$\begin{aligned} \int \pi_T(\cdot) &= \frac{p(\theta)}{p(y_{0:T})} \mathbb{E}\left[L_T^N(\theta)\right] \mathrm{d}\theta \\ &= \frac{p(\theta)p(y_{0:T}|\theta)}{p(y_{0:T})} \mathrm{d}\theta = p(\theta|y_{0:T})\mathrm{d}\theta \end{aligned}$$

More direct proof for $T = 1$

$$\psi_{1,\theta}(\mathrm{d}x_{0:1}^{1:N}, \mathrm{d}a_1^{1:N}) = \prod_{n=1}^{N} M_0^\theta(\mathrm{d}x_0^n) \left\{ \prod_{n=1}^{N} M_1^\theta(x_0^{a_1^n}, \mathrm{d}x_1^n) W_{0,\theta}^{a_1^n} \mathrm{d}a_1^n \right\}$$

with $W_{0,\theta}^n = G_0^\theta(x_0^n) / \sum_{m=1}^N G_0^\theta(x_0^m)$. So

$$\pi_1(\cdot) = \frac{p(\theta)}{p(y_{0:t})} \psi_{1,\theta}(\cdot) \left\{ \frac{1}{N} \sum_{n=1}^{N} G_0^\theta(x_0^n) \right\} \left\{ \frac{1}{N} \sum_{n=1}^{N} G_1^\theta(x_0^{a_1^n}, x_1^n) \right\}$$

$$= \frac{p(\theta)}{N^2 p(y_{0:t})} \sum_{n=1}^{N} G_1^\theta(x_0^{a_1^n}, x_1^n) M_1^\theta(x_0^{a_1^n}, x_1^n) \frac{G_0^\theta(x_0^{a_1^n})}{\sum_{m=1}^N G_0^\theta(x_0^m)} \left\{ \sum_{m=1}^{N} G_0^\theta(x_0^m) \right.$$

$$\times M_0^\theta(\mathrm{d}x_0^{a_1^n}) \left\{ \prod_{i \neq a_1^n} M_0^\theta(\mathrm{d}x_0^i) \right\} \left\{ \prod_{i \neq n} M_1^\theta(x_0^{a_1^i}, \mathrm{d}x_1^i) W_1^{a_1^i} \mathrm{d}a_1^i \right\}$$

Interpretation

$$\pi_1(\mathrm{d}\theta, \mathrm{d}x_{0:1}^{1:N}, \mathrm{d}a_1^{1:N}) = \frac{1}{N} \times \left[\frac{1}{N} \sum_{n=1}^{N} p(\mathrm{d}\theta, \mathrm{d}x_0^{a_1^n}, \mathrm{d}x_1^n | y_{0:1}) \right.$$
$$\left. \prod_{i \neq a_1^n} M_0^\theta(\mathrm{d}x_0^i) \left\{ \prod_{i \neq n} M_1^\theta(x_0^{a_1^i}, \mathrm{d}x_1^i) W_0^{a_1^i} \right\} \right]$$

which is a mixture distribution, with probability $1/N$ that path n follows $p(\mathrm{d}\theta, \mathrm{d}x_{0:1}|y_{0:1})$, A_1^n is Uniform in $1:N$, and other paths follows a conditional SMC distribution (the distribution of a particle filter conditional on one trajectory being fixed). From this calculation, one easily deduce the unbiasedness property (directly!) but also properties similar to those of the GIMH.

Additional properties (similar to GIMH)

Property 2

Marginal PMCMC is a Metropolis sampler with invariant distribution π_T, and proposal distribution $h(\theta_\star|\theta)\mathrm{d}\theta_\star\psi_{T,\theta_\star}(\cdot)$. (In particular, it leaves invariant the posterior $p(\mathrm{d}\theta|y_{0:T})$.)

Proof: write the MH ratio, same type of cancellations as for GIMH.

Additional properties (similar to GIMH)

Property 3

If we extend π_T by adding component $k \in 1:N$ with conditional probability $\propto W_T^k$, then the joint pdf $\pi_T(\mathrm{d}\theta, \mathrm{d}x_{0:T}^{1:N}, \mathrm{d}a_{1:T-1}^{1:N}, \mathrm{d}k)$ is such that

(a) $(\theta, X_{0:T}^\star) \sim p(\mathrm{d}\theta, \mathrm{d}x_{0:T}|y_{0:T})$ marginally; and

(b) Given $(\theta, X_{0:T}^\star)$, the $N-1$ remaining trajectories follow the conditional SMC distribution.

where $X_{0:T}^\star$ is the $k-$th **complete** trajectory: $X_t^\star = X_t^{B_t}$ for all t, with $B_T = k$, $B_{T-1} = A_T^k$, ... $B_0 = A_1^{B_1}$.

Outline

CSMC

- The formalisation of PMCMC offers the possibility to regenerate the $N-1$ trajectories that have not been selected; this is essentially a Gibbs step, conditional on θ, and the selected trajectory $X^{\star}_{0:T}$.
- This CSMC step cannot be analysed with the same tools as marginal PMCMC, as in Andrieu and Vihola (2012).

From now on, we drop θ from the notations.

Algorithmic description ($T = 1$)

Assume selected trajectory is $X_{0:1}^\star = (X_0^1, X_1^1)$; i.e. $k = 1$, $A_1^k = 1$.
At time $t = 0$:

(a) sample $X_0^n \sim M_0(\mathrm{d}x_0)$ for $n \in 2 : N$.

(b) Compute weights $w_0^n = G_0(X_0^n)$ and normalise, $W_0^n = w_0^n / \sum_{m=1}^N w_0^m$.

At time $t = 1$:

(a) Sample $A_1^{2:N}$ $\mathcal{M}(W_0^{1:N})$.

(b) Sample $X_1^n \sim M_1(X_1^{A_0^n}, \mathrm{d}x_1)$ for $n \in 2 : N$.

(c) Compute weights $w_1^n = G_1(X_0^{A_1^n}, X_1^n)$ and normalise, $W_1^n = w_1^n / \sum_{m=1}^N w_1^m$.

(d) select new trajectory k with probability W_1^k.

then return $\tilde{X}_{0:1}^\star = (X_0^{A_1^k}, X_1^k)$.

Some remarks

- One may show that the CSMC update does not depend on the labels of the frozen trajectory. This is why we set these arbitrarily to $(1, \ldots, 1)$. Formally, this means that the CSMC kernel is such that $K^N_{\mathrm{CSMC}} : \mathcal{X}^T \to \mathcal{P}(\mathcal{X}^T)$.
- This remains true for other resampling schemes (than multinomial); see next two* slides for an example

Properties of the CSMC kernel

Theorem

Under appropriate conditions, one has, for any $\varepsilon > 0$,

$$\left| K^N_{\mathrm{CSMC}}(\varphi)(x_{0:T}) - K^N_{\mathrm{CSMC}}(\varphi)(x'_{0:T}) \right| \leq \varepsilon$$

for N large enough, and $\varphi : \mathcal{X}^T \to [-1, 1]$.

This implies uniform ergodicity. Proof based on a coupling construction.

Assumptions

- G_t is upper bounded, $G_t(x_t) \leq g_t$.
- We have

$$\int M_0(dx_0) G_0(x_0) \geq \frac{1}{g_0}, \quad \int M_t(x_{t-1}, dx_t) G_t(x_t) \geq \frac{1}{g_t}$$

But no assumptions on the kernels M_t.

Backward sampling

Nick Whiteley (in his RSS discussion of PMCMC) suggested to add an extra **backward** step to CSMC, where one tries to modify (recursively, backward in time) the ancestry of the selected trajectory.
In our $T = 1$ example, and for multinomial resampling, this amounts to draw A_1^k from

$$\mathbb{P}(A_1^k = a | k, x_{0:1}^{1:N}) \propto W_0^a m_1(x_1^k | x_0^a)$$

where $m_1(x_1^k | x_0^a)$ is the PDF at point x_1^k of $M_1(x_0^a, dx_1)$, then return $x_{0:1}^\star = (x_0^a, x_1^k)$.

BS for other resampling schemes

More generally, BS amounts to draw a_1^k from

$$P(a_1^k = a|k, x_{1:2}^{1:N}) \propto \rho_1(W_1^{1:N}; a_1^k = a|a_1^{-k}) m_2(x_1^a, x_2^k)$$

where a_1^{-k} is $a_1^{1:N}$ minus a_1^k.

So we need to be able the conditional probability $\rho_1(W_1^{1:N}; a_1^k = a|a_1^{-k})$ for alternative resampling schemes.

Why BS would bring an improvement?

C. and Singh (2014) prove that CSMC+BS dominates CSMC in efficiency ordering (i.e. asymptotic variance). To do so, they prove that these two kernels are reversible; see Tierney (1998), Mira & Geyer (1999).

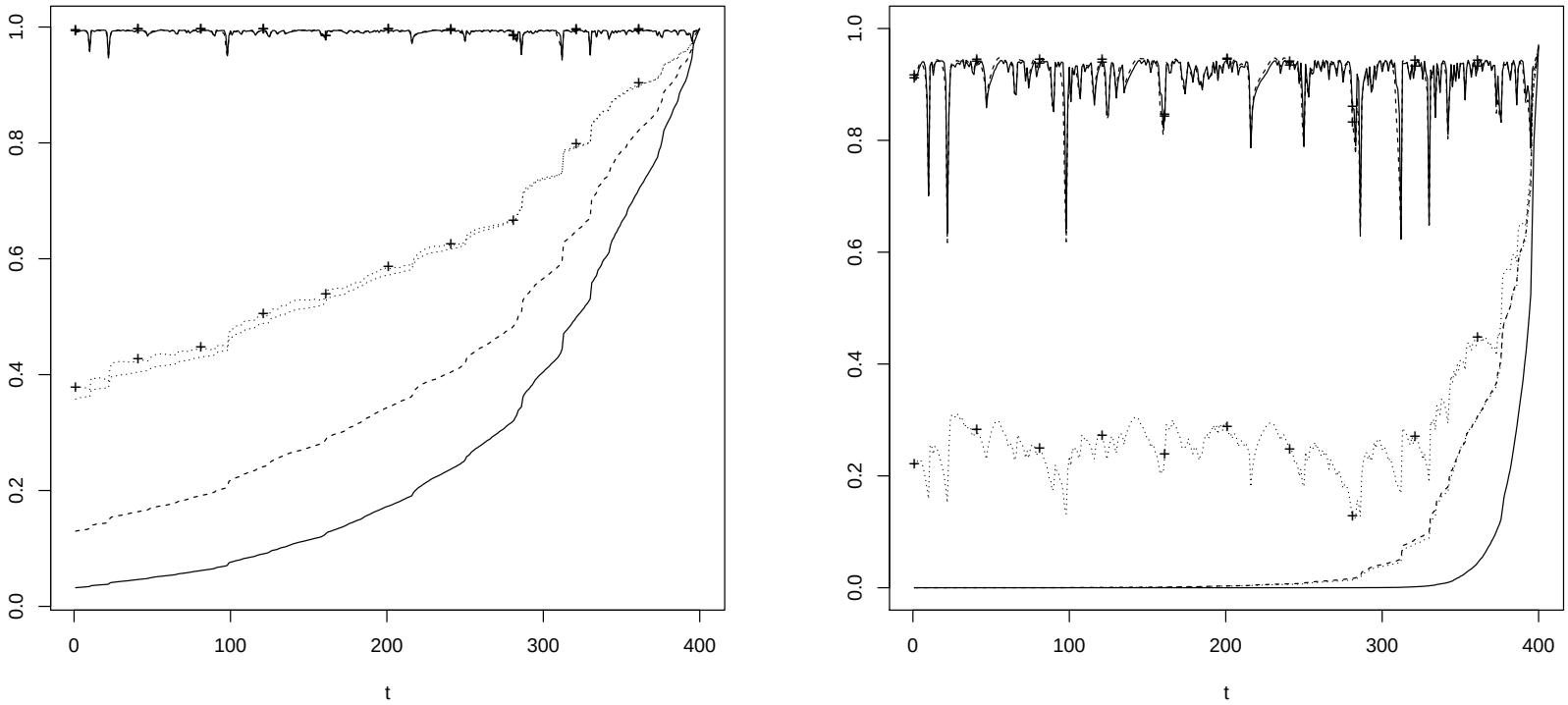
Simulations

See the plots in next slide, based on the following simple state-space model, with $\theta = (\mu, \phi, \sigma)$:

$$x_t - \mu = \phi(x_{t-1} - \mu) + \sigma\epsilon_t, \quad \epsilon_t \sim N(0,1)$$

$$y_t | x_t \sim \text{Poisson}(e^{x_t})$$

Update rate of X_t



Left: $N = 200$, right: $N = 20$. Solid line: multinomial, Dashed line: residual; Dotted line: Systematic. Crosses mean BS has been used.

Conclusion

- When the backward step is possible, it should be implemented, because it improves mixing dramatically. In that case, multinomial resampling is good enough.
- When the backward step cannot be implemented, switching to systematic resampling helps.

But what's the point of PG?

It's a bit the same discussion as marginal Metropolis (in θ-space) versus Gibbs:

- Gibbs does not work so well when they are strong correlations (here between θ and $X^{\star}_{0:T}$);
- Metropolis requires a good proposal to work well.

In some cases, combining the two is helpful: in this way, the CSMC update will refresh the particle system, which may help to get "unstuck".

PMCMC: Practical calibration

nicolas.chopin@ensae.fr

Section 1

Toy example

Example

- $X_0 \sim \mathcal{N}(0, \sigma_X^2)$, and

$$X_t = \rho X_{t-1} + U_t, \quad U_t \sim \mathcal{N}(0, \sigma_X^2)$$
$$Y_t = X_t + V_t, \quad V_t \sim \mathcal{N}(0, \sigma_Y^2)\,,$$

- Prior for $\theta = (\rho, \sigma_X^2, \sigma_Y^2)$: $\rho \sim \mathcal{U}([-1, 1])$, σ_X^2, $\sigma_Y^2 \sim \mathcal{IG}(2, 2)$.
- Data simulated from the model: $T + 1 = 100$, $\rho = 0.9$, $\sigma_X = 1$, $\sigma_Y = 0.2$.

Considered algorithms

- Gaussian random walk PMMH (for various values of N); covariance of proposal is τI_3, for various values of τ.
- "ideal" random walk Metropolis (i.e. true likelihood is computed using Kalman), with same proposal.

MSJD (mean squared jumping distance) vs acceptance rate

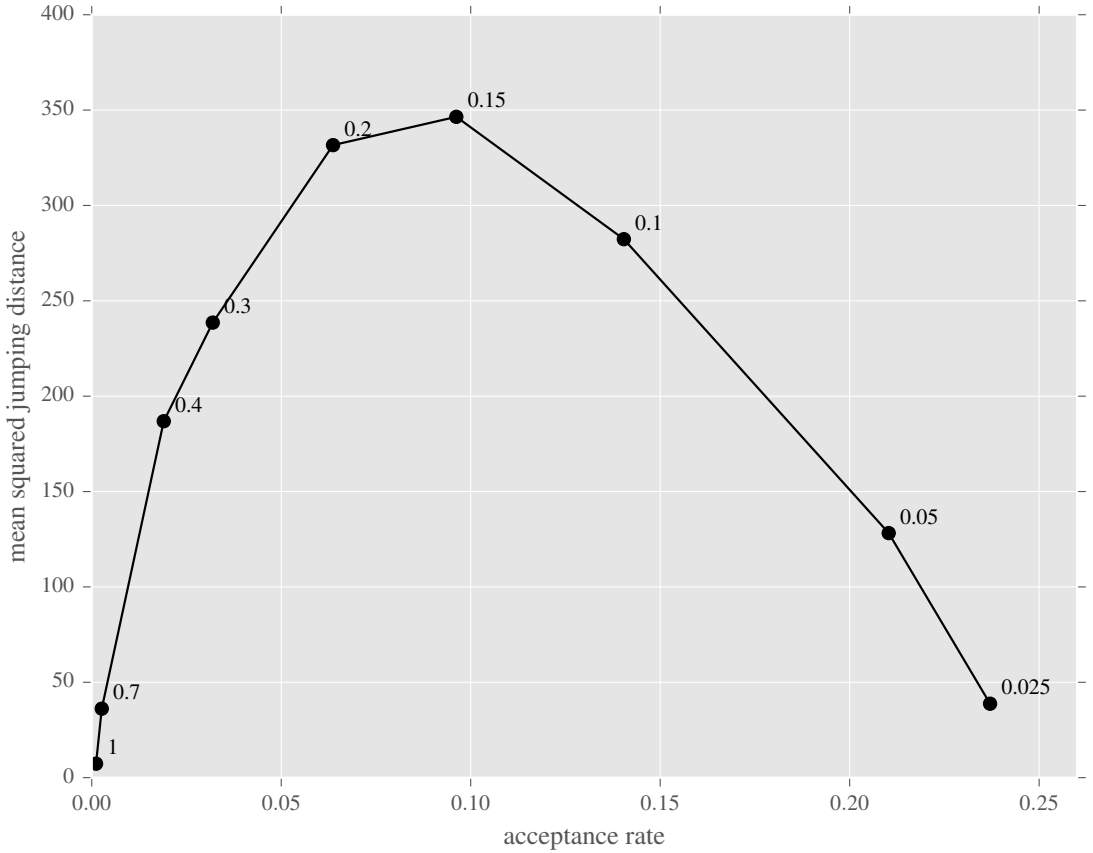


Figure 1: Mean squared jumping distance versus acceptance rate, for PMMH with $N = 100$, and different random walk scales τ; the value of τ is printed next to each dot.

Acceptance rate vs N

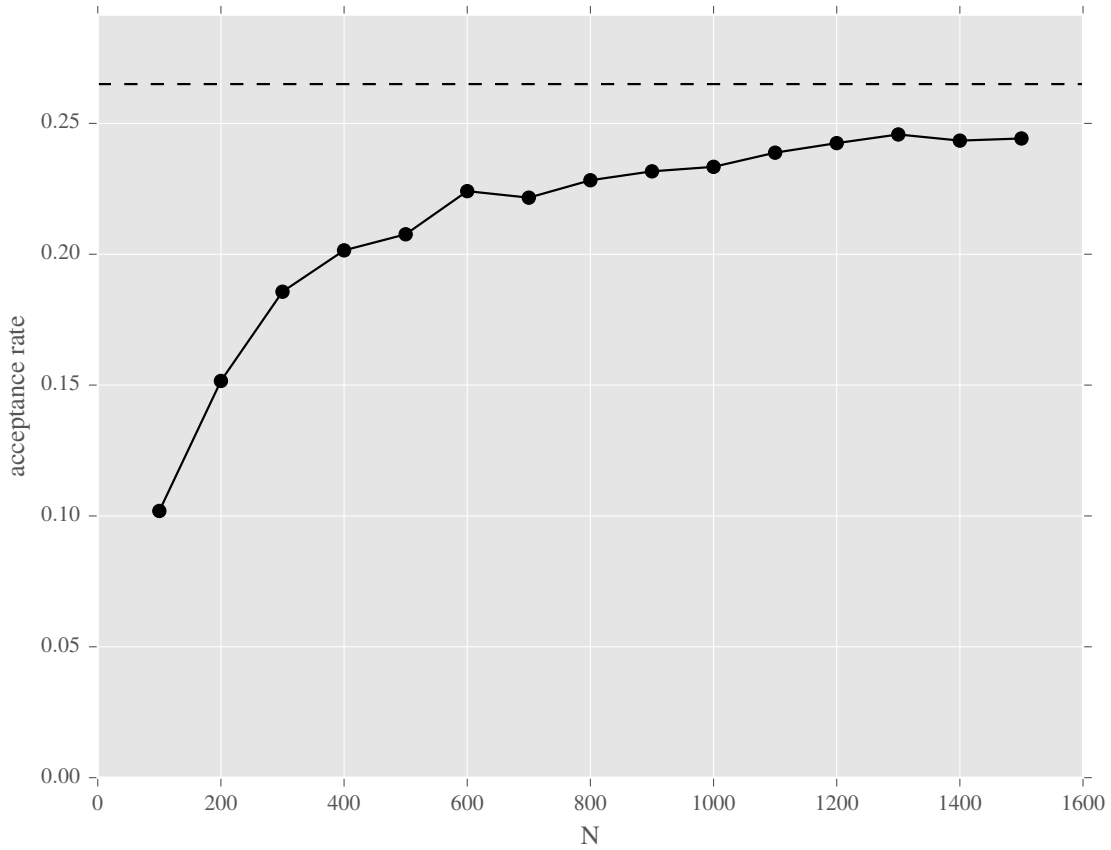


Figure 2: Acceptance rate versus N, for the random walk PMMH algorithm; dashed line gives the acceptance rate of the ideal sampler.

ACFs

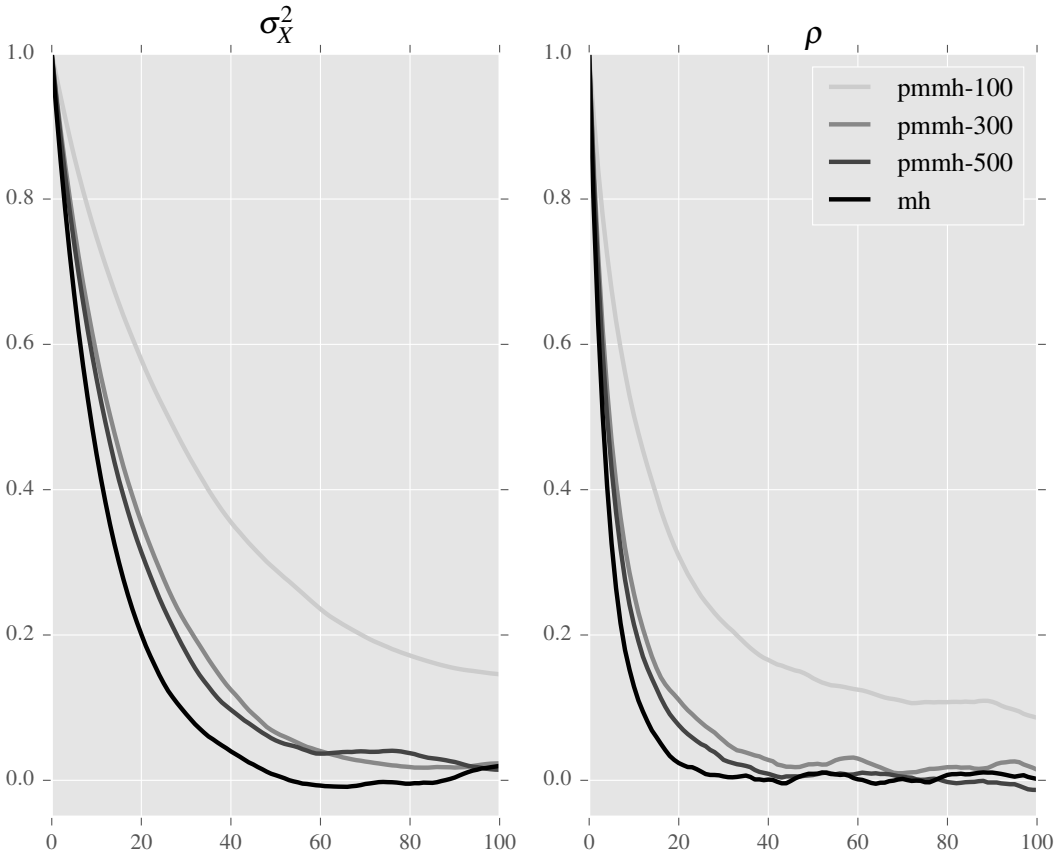


Figure 3: ACFs (auto-correlation function) of two components of θ of the ideal sampler and selected PMMH samplers (based on the bootstrap filter for $N = 100, 300, 500$), in the linear Gaussian toy example.

MSJD vs log-likelihood variance

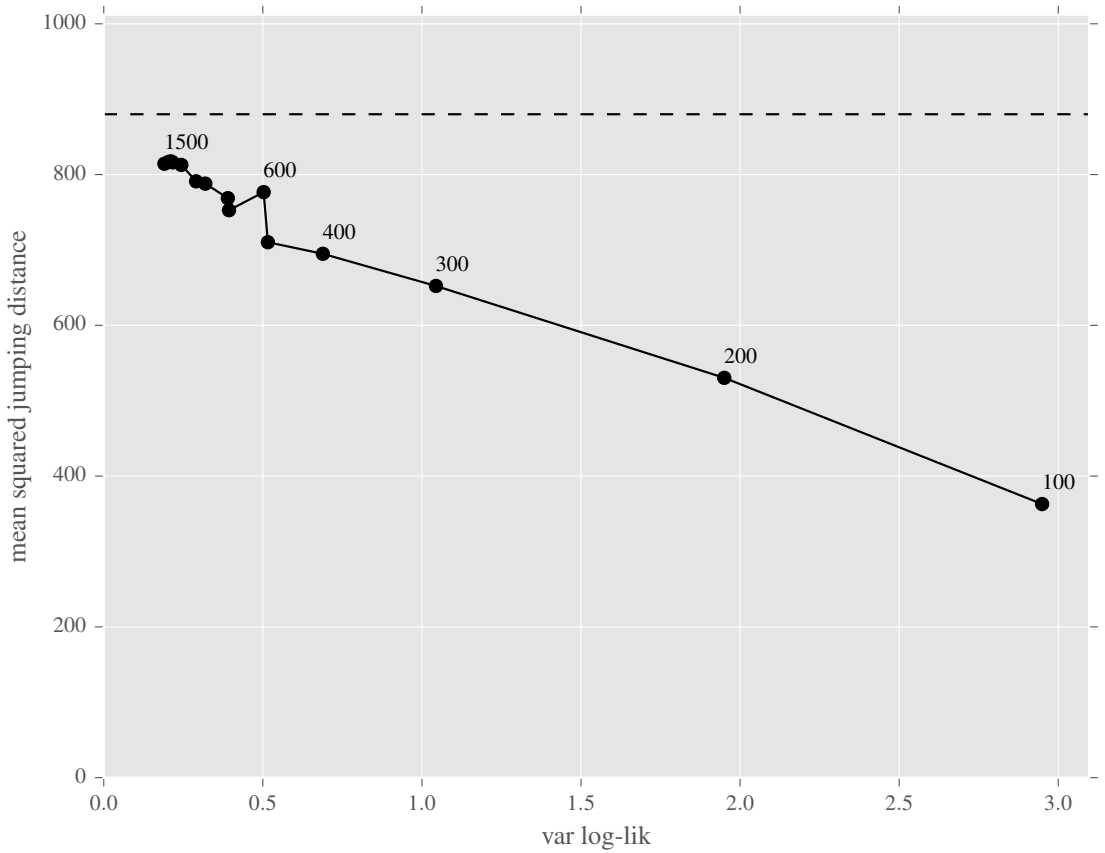


Figure 4: Mean squared jumping distance versus log-likelihood variance, for the PMMH algorithm ($N = 100, \ldots, 1000$). The latter quantity is in fact the average (over 10 values of θ sampled from the posterior by the

Change prior to $\sigma_Y^2 \sim \mathrm{G}(1/2, 1/2)$

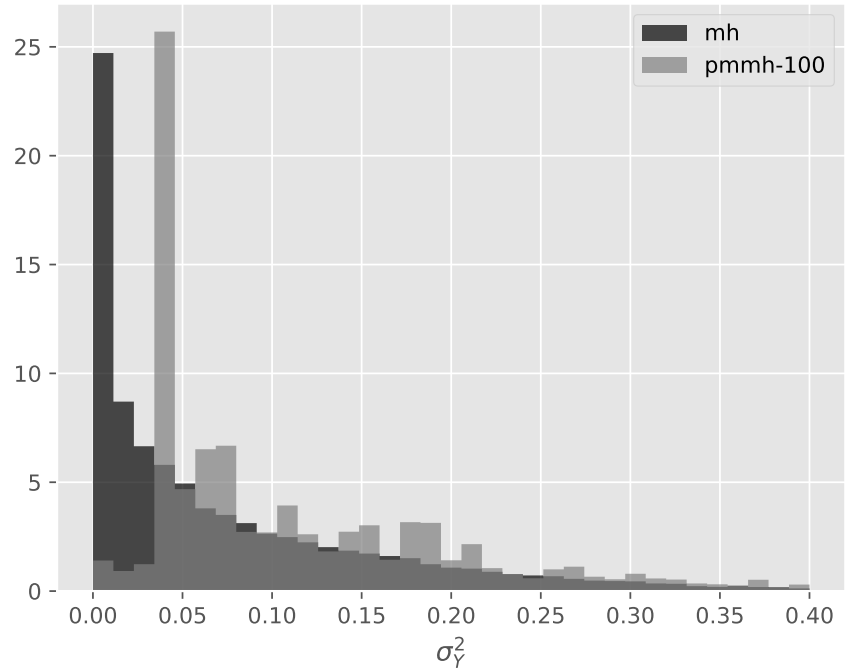


Figure 5: Marginal posterior distribution of σ_Y^2, as estimated by the ideal sampler (black), and PMMH with $N = 100$ (grey), for the alternative prior.

Section 2

Theta-logistic model in Ecology

The model

- $X_0 \sim N(0,1)$ (for simplicity) and

$$X_t = X_{t-1} + \tau_0 - \tau_1 \exp(\tau_2 X_{t-1}) + U_t, \qquad U_t \sim N(0, \sigma_X^2)$$
$$Y_t = X_t + V_t, \qquad V_t \sim N(0, \sigma_Y^2)$$

- Prior for $\theta = (\tau_0, \tau_1, \tau_2, \sigma_X^2, \sigma_Y^2)$: $\tau_i \sim \mathcal{N}_+(0,1)$ (a normal distribution truncated to $\mathbb{R}^+$), σ_X^2, $\sigma_Y^2 \sim IG(2,1)$.
- Real data.

Shape of posterior

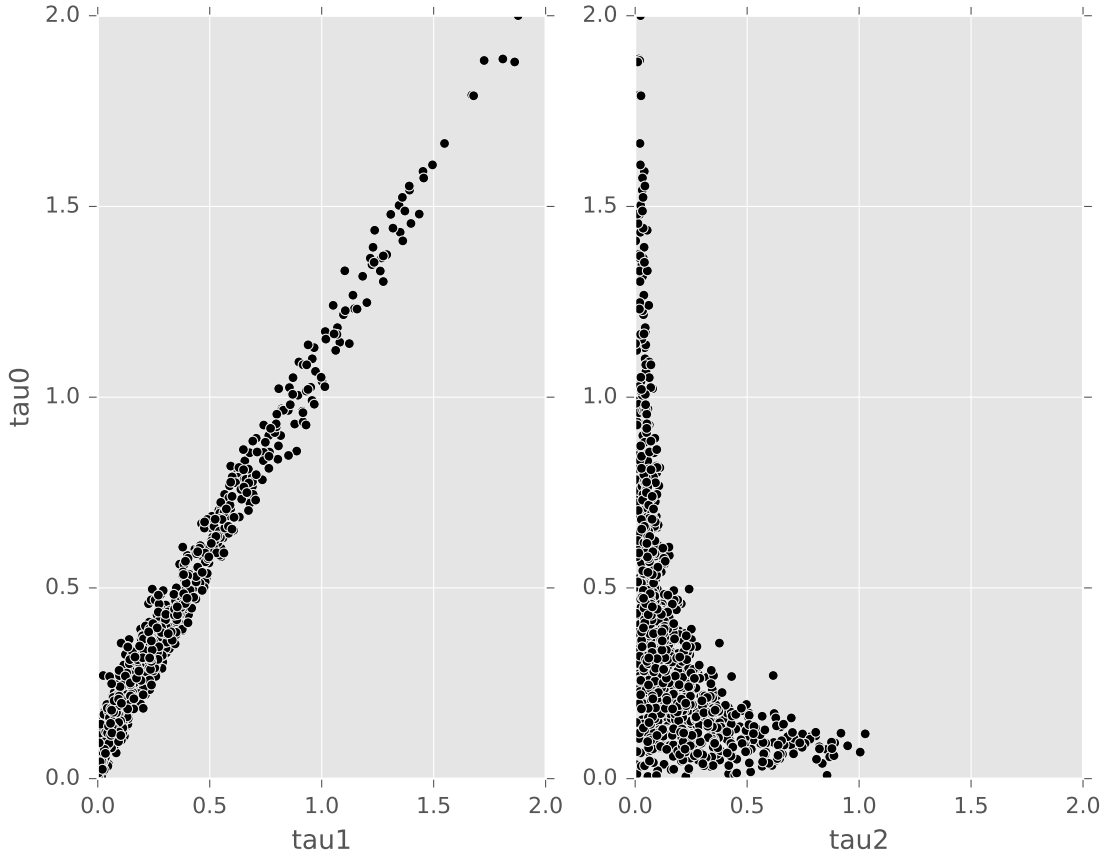


Figure 6: Selected pair plots for the output of the Particle Gibbs sampler (with backward step) in the theta-logistic example.

Results

PMMH almost impossible to calibrate for good performance here; instead we obtained good performance from Particle Gibbs.

Section 3

Conclusions

Recommendations

If you wish to implement PMMH:

- Try to design a PF such the variance of the log-likelihood estimate is $\ll 1$, for θ's that are *representative* of the posterior. For this, you may need to increase N, and/or use a better proposal (guided filter), and/or use SQMC.
- Then calibrate the random walk proposal so as to obtain e.g. a high value for the MSJD.
- Adaptive strategies may really help in this case; alternatively, consider Particle Gibbs or SMC^2.

(Waste-free) sequential Monte Carlo samplers

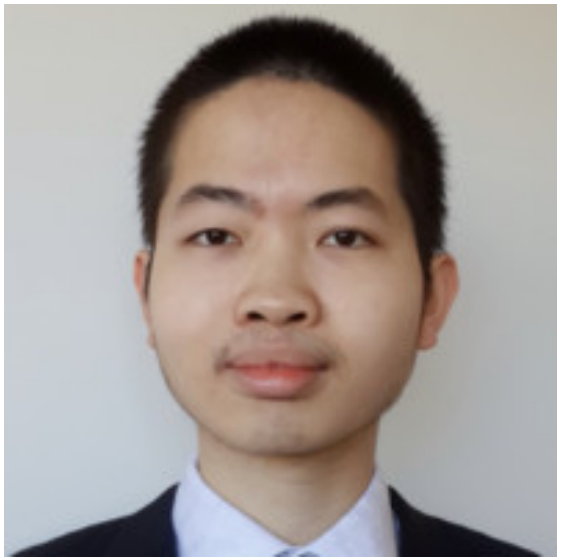
nicolas.chopin@ensae.fr

Section 1

Introduction

Two talks in one

1. Overview of SMC samplers (and how/why they may be overlooked)
2. Proposed improvement: waste-free SMC, joint work with:



Hai-Dang Dau

Section 2

SMC: motivation

SMC: what is it?

SMC = an algorithm that mixes importance sampling and MCMC techniques so as to approximate a **sequence** of distributions.

SMC: what is it?

SMC = an algorithm that mixes importance sampling and MCMC techniques so as to approximate a **sequence** of distributions.

In some problems, you already have a sequence of distributions of interest.

SMC: what is it?

SMC = an algorithm that mixes importance sampling and MCMC techniques so as to approximate a **sequence** of distributions.

In some problems, you already have a sequence of distributions of interest.

In other problems, you may have a single distribution of interest, and you need to **design a sequence** that ends at the target. (I will give some recommendations.)

SMC: why?

- often only requirement is: being able to compute pointwise the (un-normalised) target density.
- parallelisable;
- estimates of the normalising constants;
- adaptive (see 2nd part);
- competitive (e.g. C. and Ridgway, 2017; Buchholz et al, 2020).

PAC-Bayesian learning (see Alquier, 2021)

A ML method based on a pseudo-posterior:

$$\pi(\theta) \propto \mu(\theta) \exp\{-\lambda R_n(\theta)\}$$

where $R_n(\theta)$ is the empirical risk for parameter θ. For instance, for a classification task:

$$R_n(\theta) = \sum_{i=1}^{n} \mathbf{1}\{Y_i s_\theta(X_i) < 0\}$$

and s_θ could be e.g. $s_\theta(x) = \theta^T x$.

How do we choose λ?

PAC-Bayesian learning (see Alquier, 2021)

A ML method based on a pseudo-posterior:

$$\pi(\theta) \propto \mu(\theta) \exp\{-\lambda R_n(\theta)\}$$

where $R_n(\theta)$ is the empirical risk for parameter θ. For instance, for a classification task:

$$R_n(\theta) = \sum_{i=1}^{n} \mathbf{1}\{Y_i s_\theta(X_i) < 0\}$$

and s_θ could be e.g. $s_\theta(x) = \theta^T x$.

How do we choose λ?

$\Rightarrow$ Consider a sequence of values $0 = \lambda_0 < \ldots < \lambda_T$, use SMC to approximate the corresponding sequence of PAC-Bayes posteriors (and do e.g. cross validation).

PAC-Bayesian learning (see Alquier, 2021)

A ML method based on a pseudo-posterior:

$$\pi(\theta) \propto \mu(\theta) \exp\{-\lambda R_n(\theta)\}$$

where $R_n(\theta)$ is the empirical risk for parameter θ. For instance, for a classification task:

$$R_n(\theta) = \sum_{i=1}^{n} \mathbf{1}\{Y_i s_\theta(X_i) < 0\}$$

and s_θ could be e.g. $s_\theta(x) = \theta^T x$.

How do we choose λ?

$\Rightarrow$ Consider a sequence of values $0 = \lambda_0 < \ldots < \lambda_T$, use SMC to approximate the corresponding sequence of PAC-Bayes posteriors (and do e.g. cross validation).

See also Chernozhukov and Hong (2003), Bissiri et al (2016).

Sequential Bayesian estimation (and model choice)

Parametric model, prior $\mu(\theta)$. Data arrive sequentially: $Y_0, Y_1, ...$

Sequential Bayesian estimation (and model choice)

Parametric model, prior $\mu(\theta)$. Data arrive sequentially: $Y_0, Y_1, ...$

Consider sequence of posterior distributions:

$$\pi_t(\theta) = \frac{1}{p(y_{0:t})} \mu(\theta) p(y_{0:t}|\theta)$$

which may be used to infer θ sequentially, and also to perform **model choice**, through the marginal likelihood:

$$p(y_{0:t}) = \int \mu(\theta) p(y_{0:t}|\theta) d\theta$$

ABC (Approximate Bayesian Computation)

Model described only through a **simulator**: $y \sim p_\theta(y)$. ABC posterior:

$$\pi_\varepsilon(\theta, y) \propto \mu(\theta) p_\theta(y) \mathbf{1}\left\{d(y, y^\star) \leq \varepsilon\right\}$$

ABC (Approximate Bayesian Computation)

Model described only through a **simulator**: $y \sim p_\theta(y)$. ABC posterior:

$$\pi_\varepsilon(\theta, y) \propto \mu(\theta) p_\theta(y) \mathbf{1}\left\{d(y, y^\star) \leq \varepsilon\right\}$$

Use a sequence $\varepsilon_0 > \varepsilon_1 > \ldots > \varepsilon_T$.

What if I don't have a sequence

Given a target distribution π, and a base distribution μ, one may **interpolate** between the two through tempering:

$$\begin{aligned}\pi_\lambda(\theta) &\propto \mu(\theta)^{1-\lambda}\pi(\theta)^\lambda \\ &= \frac{1}{Z_\lambda}\mu(\theta)\exp\left\{-\lambda V(\theta)\right\}\end{aligned}$$

where $V(\theta) := -\log\{\pi(\theta)/\mu(\theta)\}$.

What if I don't have a sequence

Given a target distribution π, and a base distribution μ, one may **interpolate** between the two through tempering:

$$\begin{aligned}\pi_\lambda(\theta) &\propto \mu(\theta)^{1-\lambda}\pi(\theta)^\lambda \\ &= \frac{1}{Z_\lambda}\mu(\theta)\exp\{-\lambda V(\theta)\}\end{aligned}$$

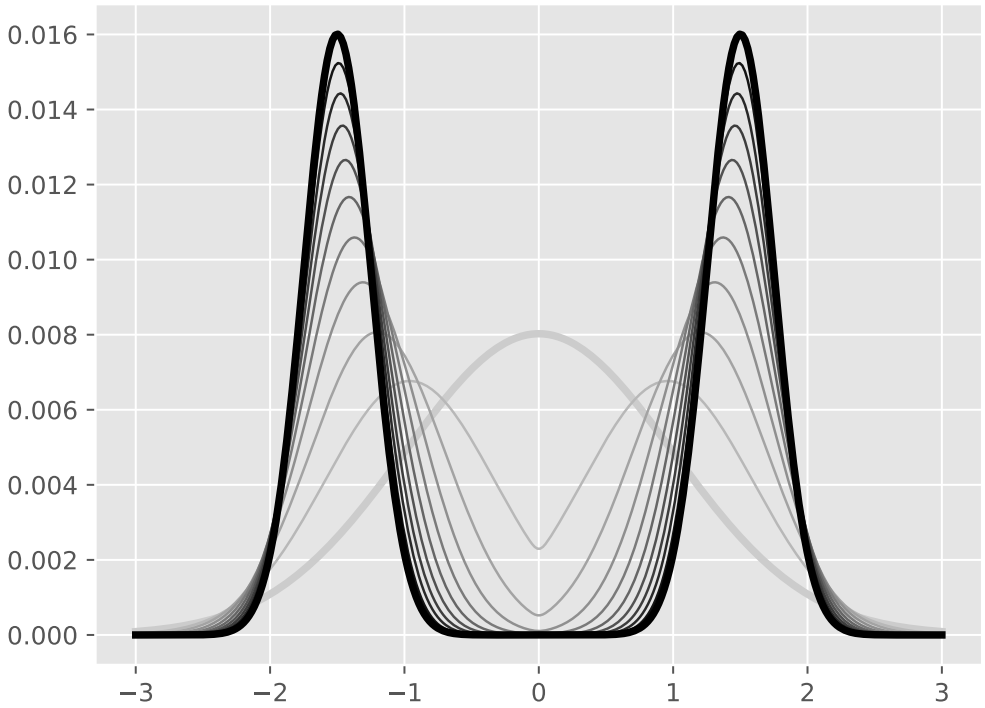
where $V(\theta) := -\log\{\pi(\theta)/\mu(\theta)\}$.

$\Rightarrow$ Introduce the **tempering** sequence:

$$\pi_t(\theta) = \frac{1}{Z_{\lambda_t}}\mu(\theta)\exp\{-\lambda_t V(\theta)\}$$

where $0 = \lambda_0 < \ldots, \lambda_T = 1$.

Pictorial representation



Tempering sequence interpolating between $N(0,1)$ and a mixture of two Gaussians.

Two critical points regarding tempering

- You can set the λ_t's automatically. **Critical** for good performance.
- Consider two Gaussian distributions in dimension d. Then:
  - their χ^2-distance is $\mathcal{O}(e^d)$.
  - however, I can design $T = \mathcal{O}(d^{1/2})$ intermediate distributions, such that the χ^2-distance between π_t and π_{t+1} is $\mathcal{O}(1)$.

Tempering lifts the curse of dimensionality; see next chapter.

Global optim

To minimise function V, consider again sequence

$$\pi_t(\theta) \propto \mu(\theta) \exp\{-\lambda_t V(\theta)\}$$

where this time $\lambda_t \to +\infty$.

Global optim

To minimise function V, consider again sequence

$$\pi_t(\theta) \propto \mu(\theta) \exp\{-\lambda_t V(\theta)\}$$

where this time $\lambda_t \to +\infty$.

Example: variable selection, θ is a binary vector (whether predictor is included or not), and $V(\theta)$ is e.g. BIC).

Global optim

To minimise function V, consider again sequence

$$\pi_t(\theta) \propto \mu(\theta) \exp\{-\lambda_t V(\theta)\}$$

where this time $\lambda_t \to +\infty$.

Example: variable selection, θ is a binary vector (whether predictor is included or not), and $V(\theta)$ is e.g. BIC).

Connection with genetic programming.

Section 3

SMC samplers

SMC: more than sequential importance sampling

Now that we have a certain sequence (π_t) of distribution: we could:

- sample $\theta^n \sim \pi_0(\theta) = \mu(\theta)$ at time 0.
- move to target π_1 through importance sampling; assign to particle θ^n weight:

$$w_1^n \propto \frac{\pi_1(\theta^n)}{\pi_0(\theta^n)}$$

- move to target π_2 through a second IS step:

$$w_2^n \propto w_1^n \frac{\pi_2(\theta^n)}{\pi_1(\theta^n)}$$

SMC: more than sequential importance sampling

Now that we have a certain sequence (π_t) of distribution: we could:

- sample $\theta^n \sim \pi_0(\theta) = \mu(\theta)$ at time 0.
- move to target π_1 through importance sampling; assign to particle θ^n weight:

$$w_1^n \propto \frac{\pi_1(\theta^n)}{\pi_0(\theta^n)}$$

- move to target π_2 through a second IS step:

$$w_2^n \propto w_1^n \frac{\pi_2(\theta^n)}{\pi_1(\theta^n)}$$

However, this boils down to IS from π_0 to π_T. Usual weight degeneracy.

SMC: resample / move steps

At time $t-1$, we have a **weighted** sample that approximates π_{t-1}:

$$\sum_{n=1}^{N} W_{t-1}^n \varphi(\theta_{t-1}^n) \approx \pi_{t-1}(\varphi)$$

where $W_{t-1}^n = w_{t-1}^n / \sum_{m=1}^N w_{t-1}^m$ (normalised weights).

In order to **rejuvenate** the sample:

- resample: draw with replacement from set of N particles, according to the weights.
- move the resampled particles according to a MCMC kernel that leaves π_{t-1} invariant.

SMC sampler algorithm

Operations involving n are performed for $n = 1, \ldots, N$.

---

for $t \leftarrow 0$ **to** T **do**

  if $t = 0$ **then**

    $\theta_0^n \sim \pi_{-1}$

  else

    $A_t^{1:N} \sim \texttt{resample}(N, W_{t-1}^{1:N})$

    $\theta_t^n \sim M_t(\theta_{t-1}^{A_t^n}, d\theta_t)$ (M_t leaves invariant π_{t-1})

  $w_t^n \leftarrow \pi_t(\theta_t^n)/\pi_{t-1}(\theta_t^n)$

  $W_t^n \leftarrow w_t^n / \sum_{m=1}^N w_t^m$

An example of a MCMC kernel: random walk Metropolis

The algorithm (with input: θ):

- Generate $\theta^p \sim N(\theta, \Sigma)$
- With probability $\alpha \wedge 1$, return θ^p, otherwise return θ, where

$$\alpha = \frac{\pi(\theta^p)}{\pi(\theta)}$$

defines a Markov kernel that leaves invariant π.

An example of a MCMC kernel: random walk Metropolis

The algorithm (with input: θ):

- Generate $\theta^p \sim N(\theta, \Sigma)$
- With probability $\alpha \wedge 1$, return θ^p, otherwise return θ, where

$$\alpha = \frac{\pi(\theta^p)}{\pi(\theta)}$$

defines a Markov kernel that leaves invariant π.

Choice of Σ critical for good performance. If $\pi \approx N(\mu, S)$, recommended to take $\Sigma = cS$, with $c = (2.38)^2/d$.

An example of a MCMC kernel: random walk Metropolis

The algorithm (with input: θ):

- Generate $\theta^p \sim N(\theta, \Sigma)$
- With probability $\alpha \wedge 1$, return θ^p, otherwise return θ, where

$$\alpha = \frac{\pi(\theta^p)}{\pi(\theta)}$$

defines a Markov kernel that leaves invariant π.

Choice of Σ critical for good performance. If $\pi \approx N(\mu, S)$, recommended to take $\Sigma = cS$, with $c = (2.38)^2/d$.

Many other types of MCMC (e.g. Gibbs, HMC, NUTS, etc.).

Practical implementation within a SMC sampler

A default strategy in SMC samplers is use (as the MCMC kernel that moves the particles at time t) k step of random walk Metropolis, with

$$\Sigma = c \times \widehat{\Sigma}$$

and $\widehat{\Sigma}$ is the empirical covariance matrix of the weighted particles.

Practical implementation within a SMC sampler

A default strategy in SMC samplers is use (as the MCMC kernel that moves the particles at time t) k step of random walk Metropolis, with

$$\Sigma = c \times \widehat{\Sigma}$$

and $\widehat{\Sigma}$ is the empirical covariance matrix of the weighted particles.

Note how easy it is to properly tune the MCMC kernel.

Practical implementation within a SMC sampler

A default strategy in SMC samplers is use (as the MCMC kernel that moves the particles at time t) k step of random walk Metropolis, with

$$\Sigma = c \times \widehat{\Sigma}$$

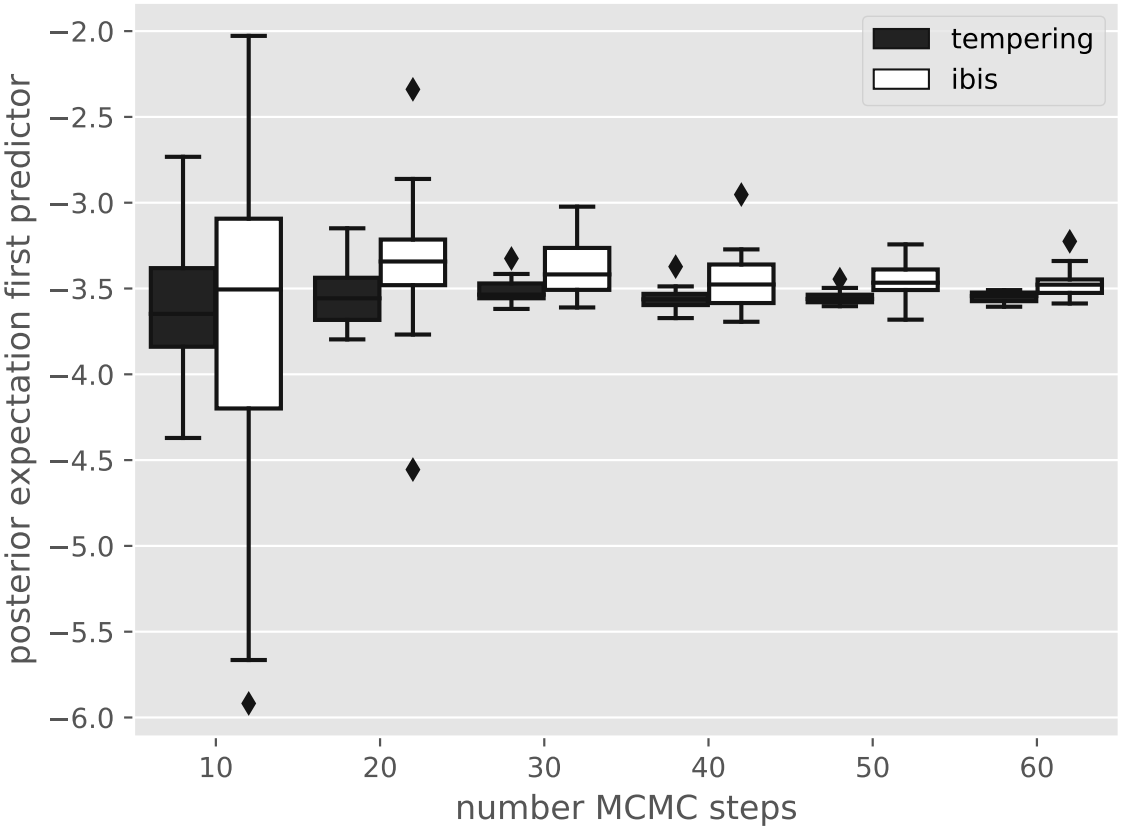
and $\widehat{\Sigma}$ is the empirical covariance matrix of the weighted particles.

Note how easy it is to properly tune the MCMC kernel.

But: how to choose k?

Numerical experiment: logistic regression, sonar dataset

For details, see Chap. 16 of C. and Papaspiliopoulos (2020).



Numerical experiment (ii)

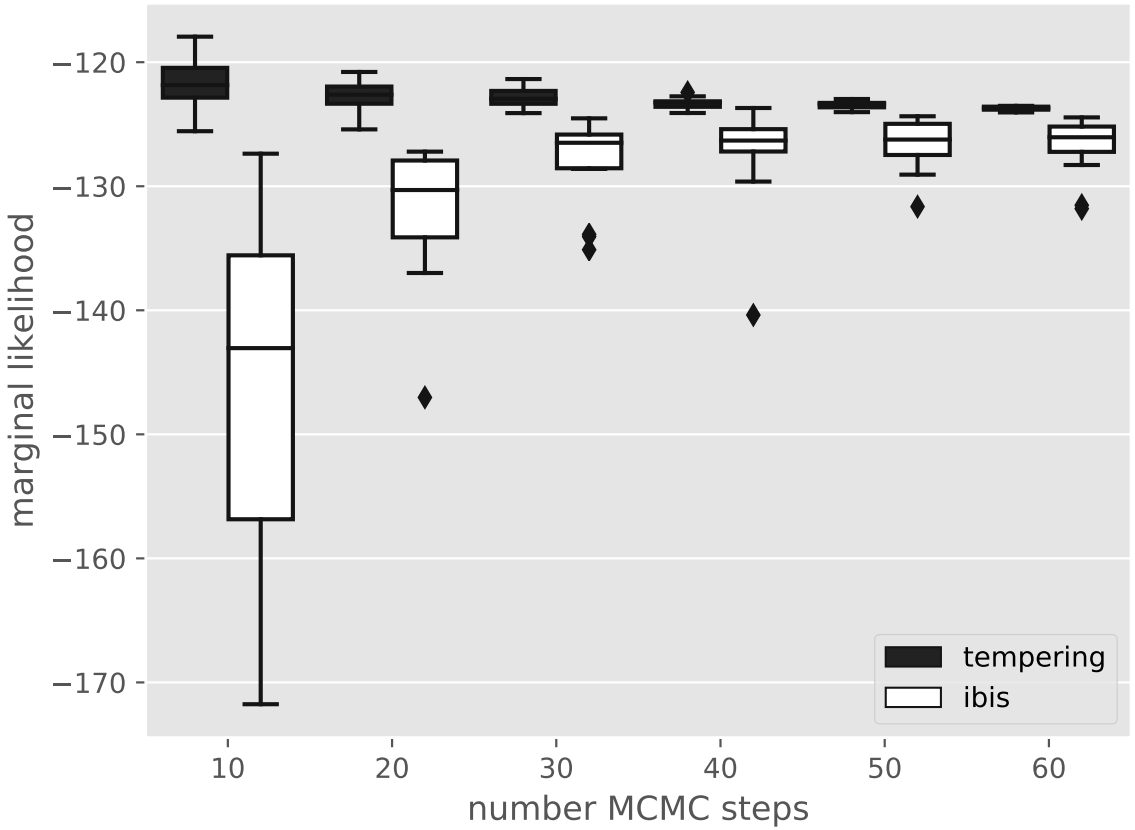
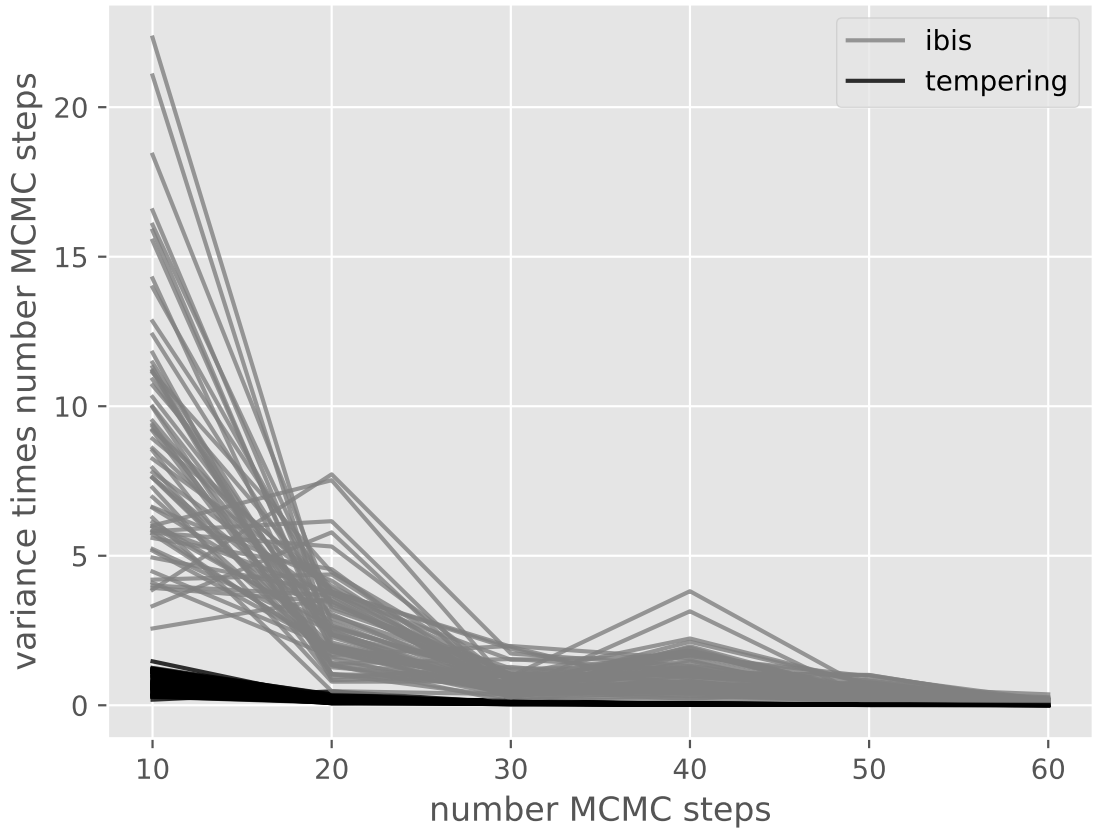


Figure 2: Same plot for log marginalising constant

Numerical experiment (iii)



Questions / remarks

- How do we choose k in practice? (without doing many pilot runs)
- Should we have the same k at all iterations?
- If k is large, not using the intermediate states seem **wasteful**.

Section 4

Waste-free SMC

Basic idea

Let M, P two integers such that $N = MP$.

At iteration t:

- resample M particles;
- apply $P - 1$ MCMC steps to each of the M resampled particles;
- keep all the intermediate steps $\Rightarrow$ N particles. **No waste.**

Basic idea

Let M, P two integers such that $N = MP$.

At iteration t:

- resample M particles;
- apply $P - 1$ MCMC steps to each of the M resampled particles;
- keep all the intermediate steps $\Rightarrow$ N particles. **No waste.**

Questions

- Validity?
- how to choose pair (M, P) (for a given N)?

P fixed, $M \to \infty$

In that regime, waste-free SMC is equivalent to a standard SMC sampler, where the target at time t corresponds to the distribution of a stationary Markov chain of length P.

P fixed, $M \to \infty$

In that regime, waste-free SMC is equivalent to a standard SMC sampler, where the target at time t corresponds to the distribution of a stationary Markov chain of length P.

Implies that:

- algorithm converges (as $N \to \infty$, while keeping P fixed)
- estimate of normalising constant is unbiased.

P fixed, $M \to \infty$

In that regime, waste-free SMC is equivalent to a standard SMC sampler, where the target at time t corresponds to the distribution of a stationary Markov chain of length P.

Implies that:

- algorithm converges (as $N \to \infty$, while keeping P fixed)
- estimate of normalising constant is unbiased.

However, this regime usually does not lead to best performance.

M fixed (or grows slowly), $P \to \infty$

$$\sqrt{N}\left(\frac{1}{N}\sum_{n=1}^{N}\varphi(\theta_t^n) - \pi_{t-1}(\varphi)\right) \Rightarrow \mathcal{N}\left(0, \tilde{\mathcal{V}}_t(\varphi)\right)$$

$$\sqrt{N}\left(\sum_{n=1}^{N} W_t^n \varphi(\theta_t^n) - \pi_t(\varphi)\right) \Rightarrow \mathcal{N}\left(0, \mathcal{V}_t(\varphi)\right)$$

where

$$\tilde{\mathcal{V}}_t(\varphi) = v_\infty(M_{t-1}, \varphi)$$

$$\mathcal{V}_t(\varphi) = \tilde{\mathcal{V}}_t\left(\bar{G}_t(\varphi - \pi_t\varphi)\right),$$

and $v_\infty(M_t, \varphi)$ is the asymptotic variance of a **stationary** Markov chain (ξ_t) with kernel M_t:

$$v_\infty(M_t, \varphi) = \mathrm{Var}\left(\varphi(\xi_0)\right) + 2\sum_{}^{\infty} \mathrm{Cov}\left(\varphi(\xi_0), \varphi(\xi_p)\right).$$

Asymptotic variances

Note that these asymptotic variances:

- do not depend on M;
- do not depend on previous iterations;
- suggest the following interpretation of the N particles: as M independent **stationary** chains of length P.

Asymptotic variances

Note that these asymptotic variances:

- do not depend on M;
- do not depend on previous iterations;
- suggest the following interpretation of the N particles: as M independent **stationary** chains of length P.

$\Rightarrow$ We can use adapt standard (initial sequence, spectral, etc.) variance estimators for MCMC chains to get a single-run estimate of the variance of our particle estimates.

Practical implications

- Take $M \ll N$.
- Take $M \geq$ the number of cores on a parallel machine.
- Single-run variance estimate based on the $M-$chain interpretation.

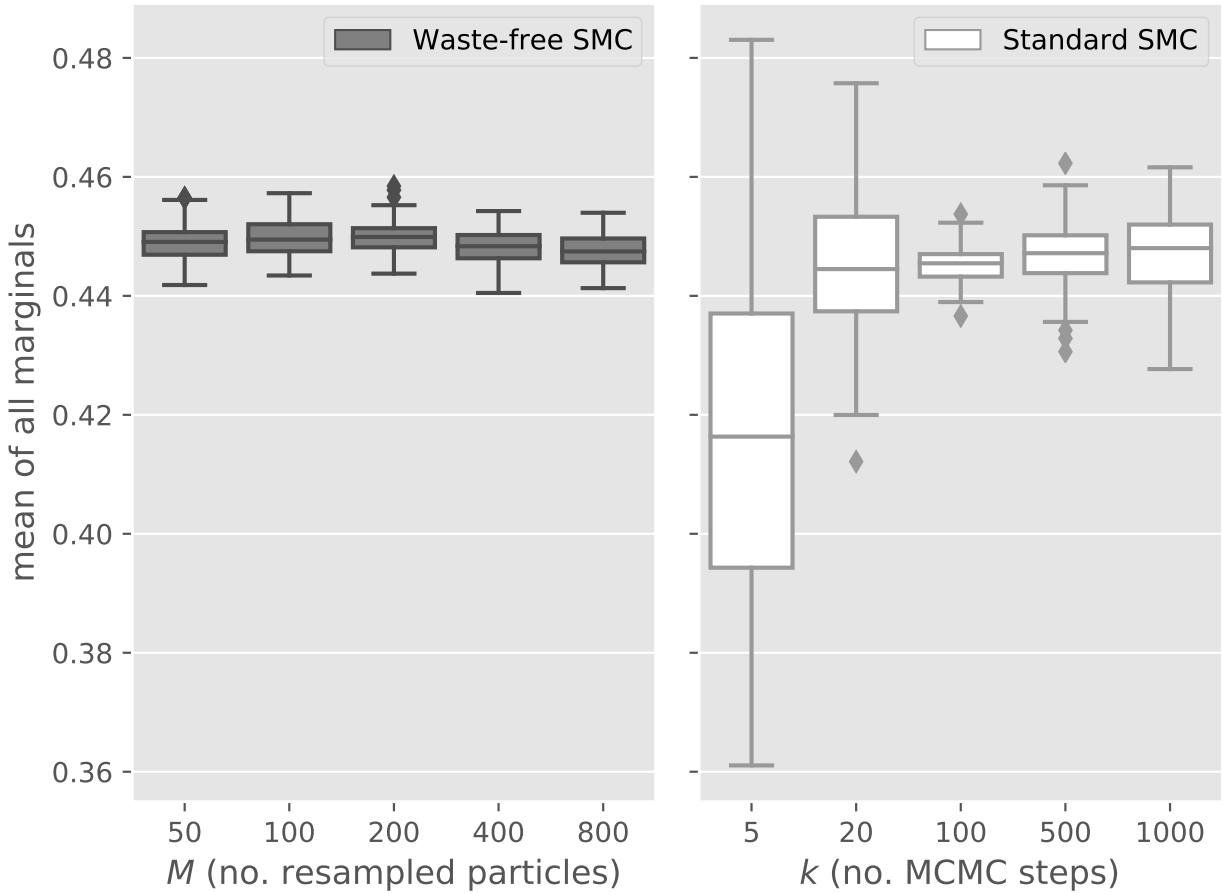
Section 5

Numerical experiments

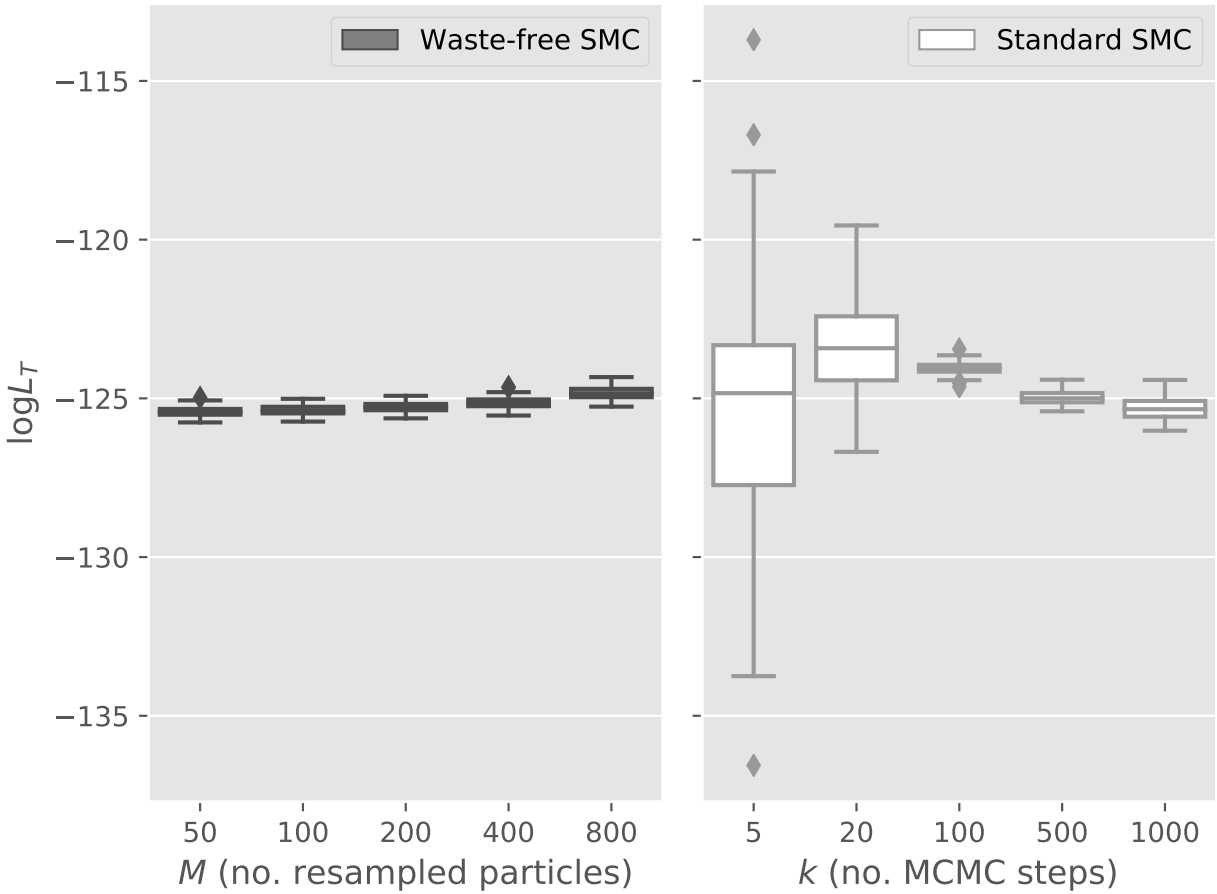
Numerical experiment I

- target: posterior of a logistic regression, sonar dataset ($d = 63$, challenging, see C. and Ridgway, 2017).
- sequence: tempering (automatic).
- MCMC steps: random walk Metropolis (calibrated on particles).
- independent runs of standard SMC (varying k, the number of MCMC steps), and waste-free SMC (varying M). Same number of likelihood evaluations: $N = 2 \times 10^5 / k$ (standard SMC), and $N = 2 \times 10^5$ for waste-free.

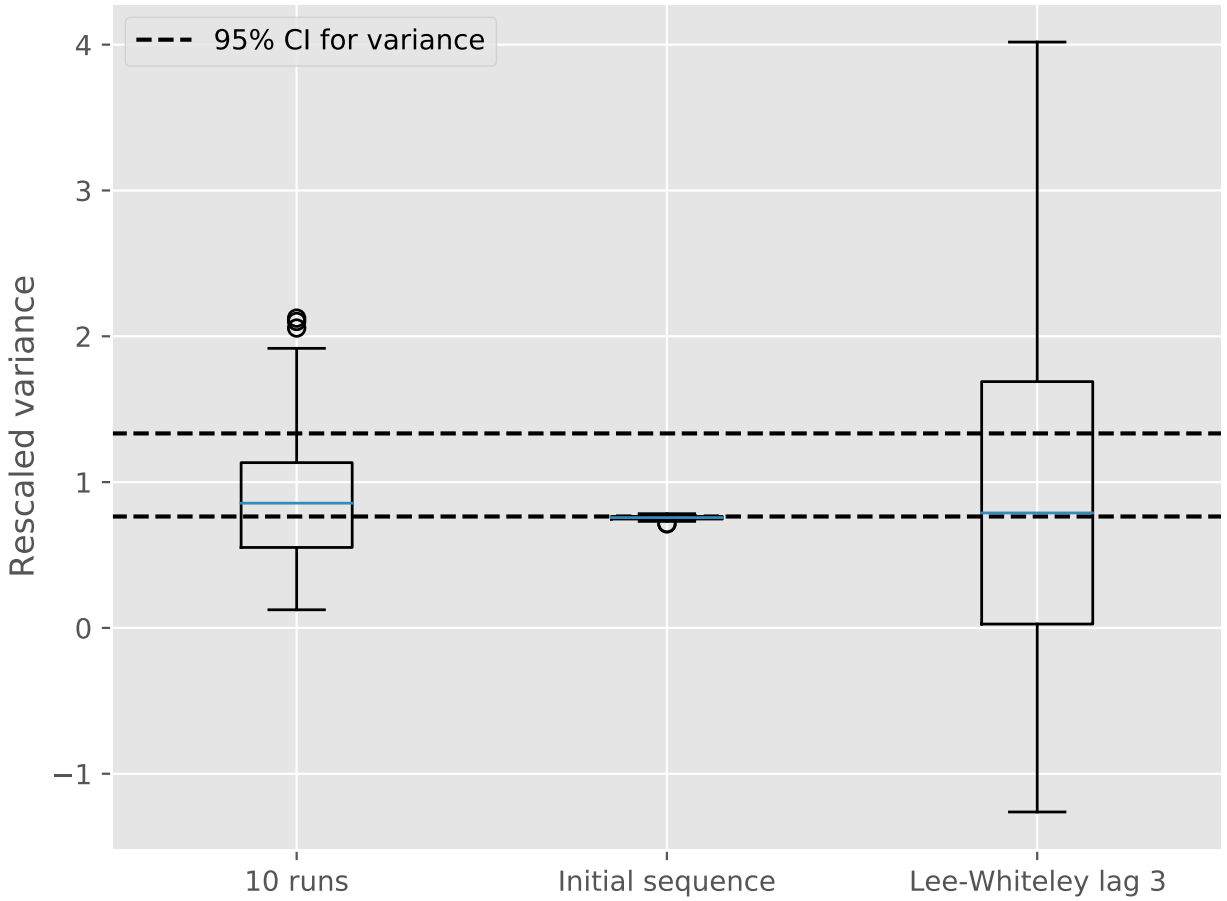
Posterior expectation of the average of the components



Log marginal likelihood



Variance estimators



Numerical experiment II

1	3	2
2	1	3
3	2	1

A Latin square of size 3.

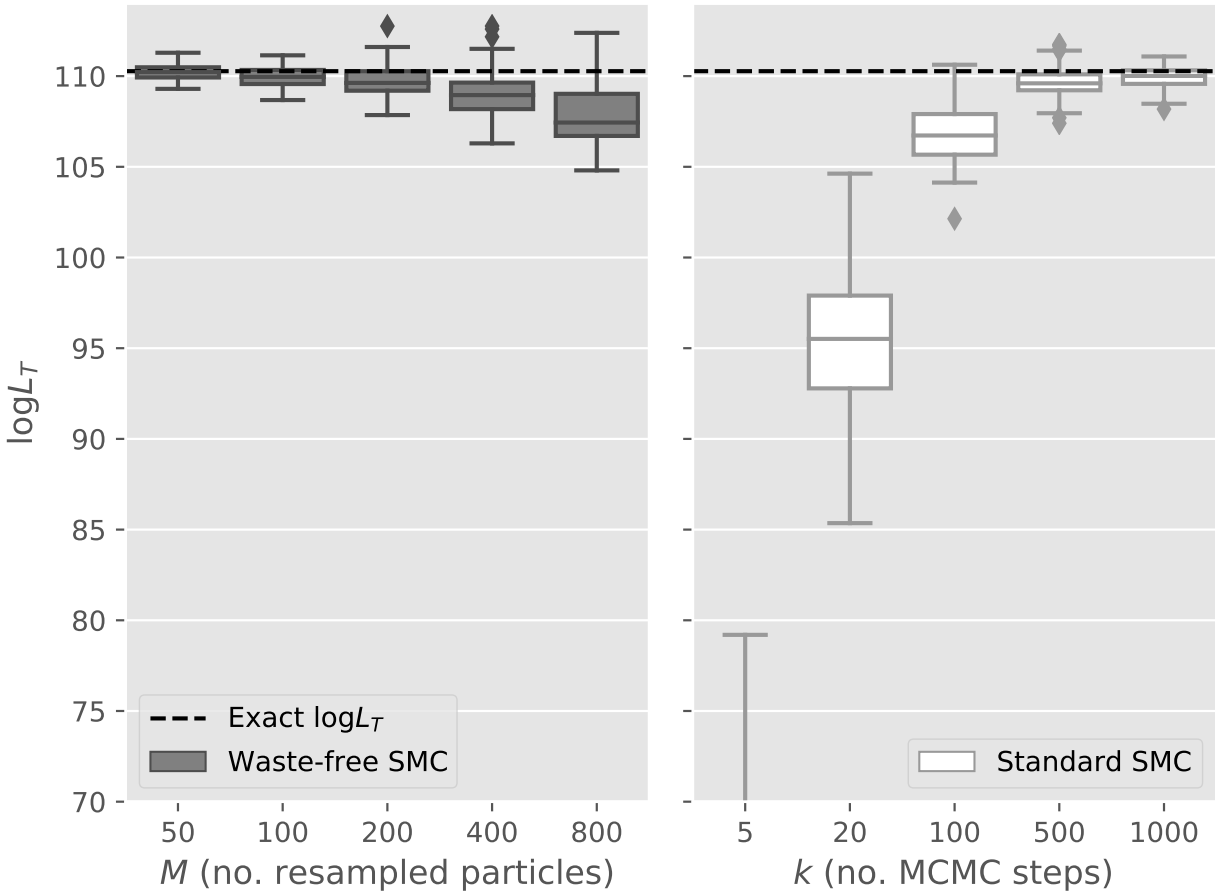
- Objective: counting the number $I(d)$ of latin squares of size d.
- Sequence:

$$\pi_t(\theta) = \frac{1}{L_t}\mu(\theta)\exp\{-\lambda_t V(\theta)\}$$

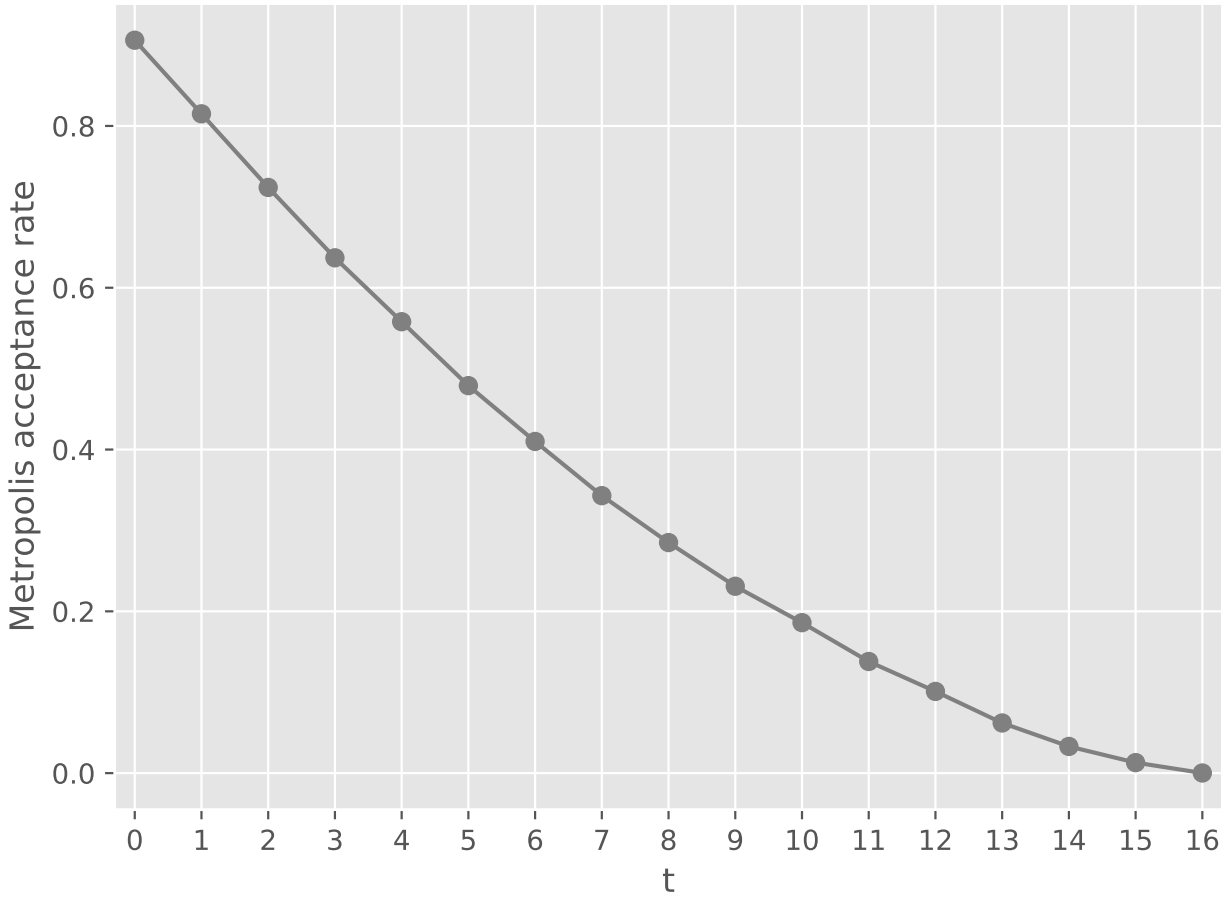
where $V(\theta) \geq 0$, $= 0$ iff θ is a Latin square. As $\lambda_t \to \infty$, L_t goes to $I(d)$.

- MCMC kernels: Metropolis, swap two entries.

Results ($d = 11$)


110
105
100
95
90
85
80
75
70
$\log L_T$
Exact $\log L_T$
Waste-free SMC
Standard SMC
50
100
200
400
800
M (no. resampled particles)
5
20
100
500
1000
k (no. MCMC steps)

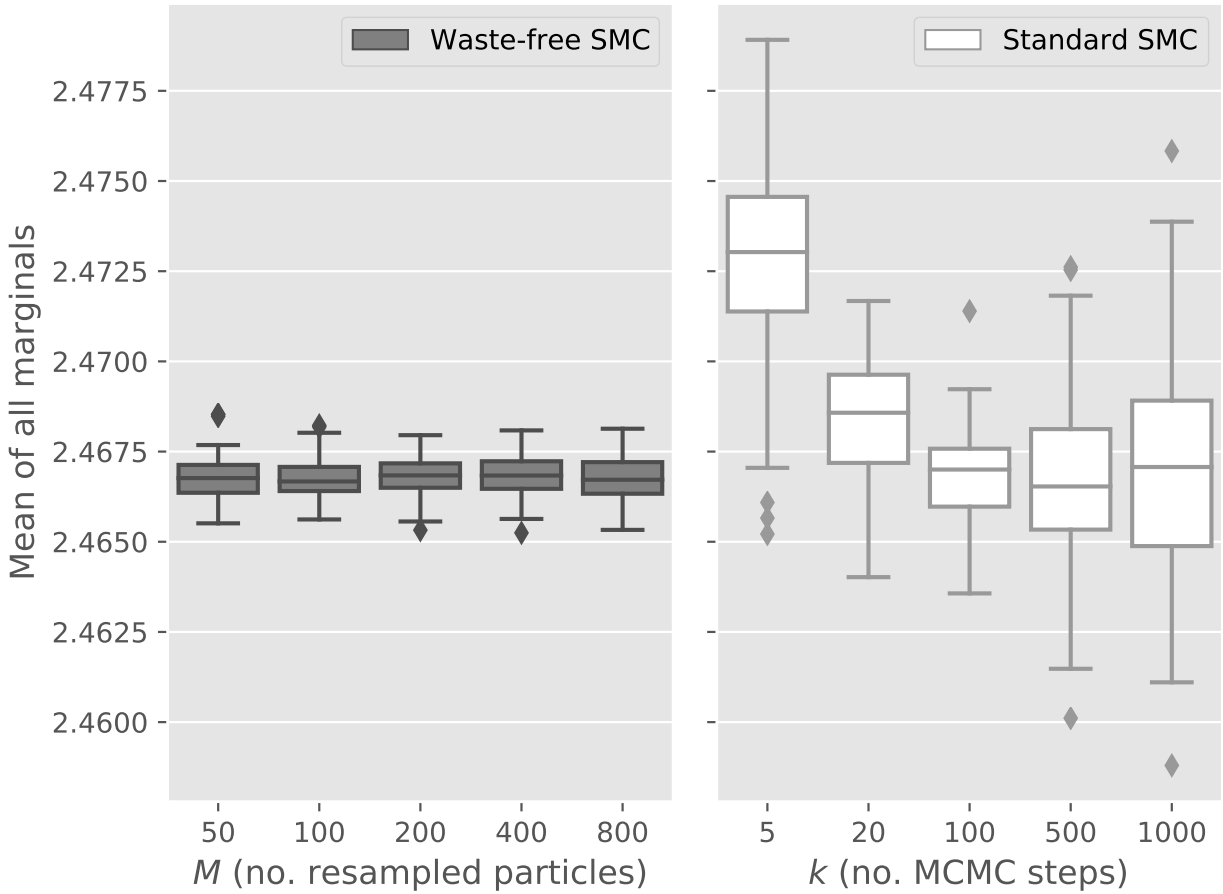
Acceptance rate vs iteration



Numerical experiment III

- Objective: evaluate orthant probability $\mathbb{P}(X \geq 0)$, $X \sim \mathcal{N}(\mu, \Sigma)$ and/or sample from the corresponding truncated Gaussian dist.
- Sequence: increasing dimension.
- MCMC steps: Gibbs

Results


Waste-free SMC
Standard SMC
Mean of all marginals
2.4775
2.4750
2.4725
2.4700
2.4675
2.4650
2.4625
2.4600
50
100
200
400
800
M (no. resampled particles)
5
20
100
500
1000
k (no. MCMC steps)

Conclusion: waste-free

- Same advantages as standard SMC:
  - automatable;
  - unbiased estimate of the normalising constant;
  - parallelisable (up to M cores)

Conclusion: waste-free

- Same advantages as standard SMC:
  - automatable;
  - unbiased estimate of the normalising constant;
  - parallelisable (up to M cores)
- In addition:
  - even more automatable (no need to choose k)
  - single-run variance estimators (**not** based on genealogy)

Conclusion: waste-free

- Same advantages as standard SMC:
  - automatable;
  - unbiased estimate of the normalising constant;
  - parallelisable (up to M cores)
- In addition:
  - even more automatable (no need to choose k)
  - single-run variance estimators (**not** based on genealogy)

In particular, **tempering SMC** is a very good default strategy if you have a single target distribution.

Implementations

- Python: https://github.com/nchopin/particles

Implementations

- Python: https://github.com/nchopin/particles
- Blackjax

Implementations

- Python: https://github.com/nchopin/particles
- Blackjax
- STAN? (on Santa's wishlist)

Reference

Dau H.D. and C. N. (2022). Waste-free Sequential Monte Carlo, Journal of the Royal Statistical Society Series B: Statistical Methodology, Volume 84, Issue 1, pages 114–148, https://doi.org/10.1111/rssb.12475

Towards a better understanding of tempering SMC

(Based on joint work with Francesca Crucinio and Anna Korba)

Tempering distributions as a parametric model

Given μ and π, the tempering distributions:

$$\begin{aligned}\pi_\lambda(\theta) &\propto \mu(\theta)^{1-\lambda}\pi(\theta)^\lambda \\ &\propto \mu(\theta)\exp\{-\lambda V(\theta)\}\end{aligned}$$

where

$$V(\theta) := -\log\{\pi(\theta)/\mu(\theta)\}.$$

define an **(exponential) parametric model**, with parameter $\lambda \in [0, 1]$.

Tempering distributions as a parametric model

Given μ and π, the tempering distributions:

$$\begin{aligned}\pi_\lambda(\theta) &\propto \mu(\theta)^{1-\lambda}\pi(\theta)^\lambda \\ &\propto \mu(\theta)\exp\{-\lambda V(\theta)\}\end{aligned}$$

where

$$V(\theta) := -\log\{\pi(\theta)/\mu(\theta)\}.$$

define an **(exponential) parametric model**, with parameter $\lambda \in [0, 1]$.

The **Fisher information** of this model is $I(\lambda) := \mathrm{var}_\lambda[V(\theta)]$.

Using the Fisher information

Proposition

For any convex, twice-differentiable function f, and $\lambda' \approx \lambda$,

$$D_f(\pi_\lambda | \pi_{\lambda'}) = \frac{f''(1) I(\lambda)}{2} (\lambda' - \lambda)^2 + \mathcal{O}\left((\lambda' - \lambda)^3\right)$$

where $D_f(\mu, \pi) := \mathbb{E}_\mu[f(\pi/\mu)]$ (f-**divergence**).

Using the Fisher information

Proposition

For any convex, twice-differentiable function f, and $\lambda' \approx \lambda$,

$$D_f(\pi_\lambda | \pi_{\lambda'}) = \frac{f''(1) I(\lambda)}{2} (\lambda' - \lambda)^2 + \mathcal{O}\left((\lambda' - \lambda)^3\right)$$

where $D_f(\mu, \pi) := \mathbb{E}_\mu[f(\pi/\mu)]$ (f-**divergence**).

This applies in particular:

- to the KL divergence, $f(x) = x \log x$;
- the χ^2-divergence, $f(x) = (x-1)^2$.

Equi-distant distributions

In practice, we'd like to choose $0 = \lambda_0 < \ldots < \lambda_T = 1$ so that

$$D_f(\pi_{\lambda_{t-1}}, \pi_{\lambda_t}) \simeq c$$

which implies:

$$\lambda_t - \lambda_{t-1} \simeq \frac{c'}{\sqrt{I(\lambda_{t-1})}}$$

for a certain $c' > 0$.

Practical implications

- If μ and π are IID distributions with d components, then $I(\lambda) = \mathcal{O}(d)$. $\Rightarrow$ Length of tempering sequence is $\mathcal{O}(d^{1/2})$.

Practical implications

- If μ and π are IID distributions with d components, then $I(\lambda) = \mathcal{O}(d)$. $\Rightarrow$ Length of tempering sequence is $\mathcal{O}(d^{1/2})$.
- the sequence (λ_t) may behave in a very different ways;

Practical implications

- If μ and π are IID distributions with d components, then $I(\lambda) = \mathcal{O}(d)$. $\Rightarrow$ Length of tempering sequence is $\mathcal{O}(d^{1/2})$.
- the sequence (λ_t) may behave in a very different ways;
- At iteration t, we could use the empirical variance of the **log-weights** to approximate $I(\lambda_{t-1})$, and decide the next exponent λ_t.

Practical implications

- If μ and π are IID distributions with d components, then $I(\lambda) = \mathcal{O}(d)$. $\Rightarrow$ Length of tempering sequence is $\mathcal{O}(d^{1/2})$.
- the sequence (λ_t) may behave in a very different ways;
- At iteration t, we could use the empirical variance of the **log-weights** to approximate $I(\lambda_{t-1})$, and decide the next exponent λ_t.
- However, in practice, the usual ESS recipe works well too.

Plots

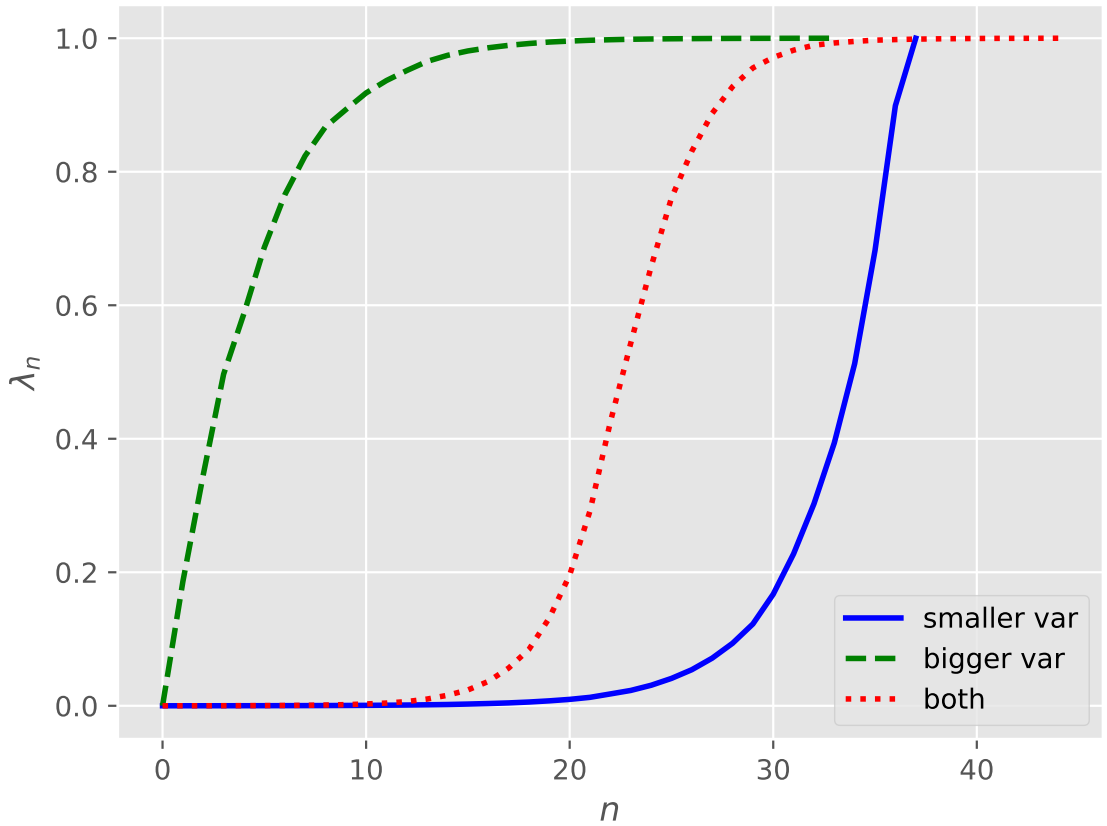


Figure 1: ESS-adapted tempering sequences, $\mu = N_d(0, I_d)$, $\pi = N_d(0, \Sigma)$, where $\Sigma = 10^2 I_d$ (bigger), $\Sigma = 10^{-2} I_d$ (smaller), and a mix of the two.

Other considerations

One can interpret tempering SMC as a numerical approximation of entropic mirror descent.

Other considerations

One can interpret tempering SMC as a numerical approximation of entropic mirror descent.

See the following manuscript for more details:

NC, crucinio F and Korba A (2023). A connection between Tempering and Entropic Mirror Descent, arXiv:2310.11914.

SMC^2: sequential estimation of state-space models

nicolas.chopin@ensae.fr

Objectives

1. to derive sequentially

$$p(\mathrm{d}\theta, \mathrm{d}x_{0:t} | Y_{0:t} = y_{0:t}), \quad p(y_{0:t}), \quad \text{for all } t \in \{0, \ldots, T\}$$

2. to obtain a black box algorithm (automatic calibration).

Main tools of our approach

- Particle filter algorithms for state-space models (this will be to estimate the likelihood, for a fixed θ).
- Iterated Batch Importance Sampling for sequential Bayesian inference for parameters (this will be the theoretical algorithm we will try to approximate).

Both are sequential Monte Carlo (SMC) methods.

IBIS (C., 2001)

SMC method for particle approximation of the sequence $p(\theta|y_{0:t})$, $t = 0 : T$. Based on the sequence of importance sampling steps:

$$\frac{p(\theta|y_{0:t})}{p(\theta|y_{0:t-1})} \propto p(y_t|y_{0:t-1}, \theta)$$

but doing only IS steps would not well. Resampling alone will not help, because θ is not an ergodic process.

$\Rightarrow$ introduces an artificial dynamics by moving the θ particles through a MCMC step (that leaves $p(\theta|y_{0:t})$ invariant).

In next slide, operations with superscript m must be understood as operations performed for all $m \in 1 : N_\theta$, where N_θ is the total number of θ-particles.

Sample θ^m from $p(\theta)$ and set $\omega^m \leftarrow 1$. Then, at time $t = 0, \ldots, T$

(a) Compute incremental weights

$$u_t(\theta^m) = p(y_t | y_{0:t-1}, \theta^m), \quad L_t = \frac{1}{\sum_{m=1}^{N_\theta} \omega^m} \times \sum_{m=1}^{N_\theta} \omega^m u_t(\theta^m),$$

(b) Update the importance weights,

$$\omega^m \leftarrow \omega^m u_t(\theta^m). \tag{1}$$

(c) If some degeneracy criterion is fulfilled, sample $\tilde{\theta}^m$ independently from the mixture distribution

$$\frac{1}{\sum_{m=1}^{N_\theta} \omega^m} \sum_{m=1}^{N_\theta} \omega^m K_t(\theta^m, \cdot).$$

Finally, replace the current weighted particle system:

$$(\theta^m, \omega^m) \leftarrow (\tilde{\theta}^m, 1).$$

Observations

- Cost of lack of ergodicity in θ: the occasional MCMC move
- Still, in regular problems resampling happens at diminishing frequency (logarithmically)
- K_t is an MCMC kernel invariant wrt $\pi(\theta \mid y_{1:t})$. Its parameters can be chosen using information from current population of θ-particles
- L_t is a MC estimator of the model evidence
- Infeasible to implement for state-space models: intractable incremental weights, and MCMC kernel

Our algorithm: SMC^2

We provide a generic (black box) algorithm for recovering the sequence of parameter posterior distributions, but as well filtering, smoothing and predictive.

We give next a pseudo-code; the code seems to only track the parameter posteriors, but actually it does all other jobs. Superficially, it looks an approximation of IBIS, but in fact it does not produce any systematic errors (unbiased MC).

Sample θ^m from $p(\theta)$ and set $\omega^m \leftarrow 1$. Then, at time $t = 0, \ldots, T$,

(a) For each particle θ^m, perform iteration t of the PF: If $t = 0$, sample independently $X_0^{1:N_x,m}$ from ψ_{0,θ^m}, and compute

$$\hat{p}(y_0|\theta^m) = \frac{1}{N_x} \sum_{n=1}^{N_x} w_0^\theta(x_0^{n,m});$$

If $t > 0$, sample $\left(X_t^{1:N_x,m}, A_t^{1:N_x,m}\right)$ from ψ_{t,θ^m} conditional on $\left(X_{0:t-1}^{1:N_x,m}, A_{1:t-1}^{1:N_x,m}\right)$, and compute

$$\hat{p}(y_t|y_{1:t-1}, \theta^m) = \frac{1}{N_x} \sum_{n=1}^{N_x} w_t^\theta(X_{t-1}^{A_t^{n,m},m}, X_t^{n,m}).$$

(b) Update the importance weights,

$$\omega^m \leftarrow \omega^m \hat{p}(y_t | y_{0:t-1}, \theta^m)$$

(c) If some degeneracy criterion is fulfilled, sample $\left(\tilde{\theta}^m, \tilde{X}_{0:t}^{1:N_x,m}, \tilde{A}_{1:t}^{1:N_x}\right)$ independently from

$$\frac{1}{\sum_{m=1}^{N_\theta} \omega^m} \sum_{m=1}^{N_\theta} \omega^m K_t \left\{ \left(\theta^m, x_{0:t}^{1:N_x,m}, a_{1:t}^{1:N_x,m}\right), \cdot \right\}$$

Finally, replace current weighted particle system:

$$(\theta^m, X_{0:t}^{1:N_x,m}, A_{1:t}^{1:N_x,m}, \omega^m) \leftarrow (\tilde{\theta}^m, \tilde{X}_{0:t}^{1:N_x,m}, \tilde{A}_{1:t-1}^{1:N_x,m}, 1)$$

Observations

- It appears as approximation to IBIS. For $N_x = \infty$ it is IBIS.
- However, no approximation is done whatsoever. This algorithm really samples from $p(\theta|y_{0:t})$ and all other distributions of interest.
- The validity of algorithm is essentially based on two results: i) the particles are weighted due to unbiasedness of PF estimator of likelihood; ii) the MCMC kernel is appropriately constructed to maintain invariance wrt to an expanded distribution which admits those of interest as marginals; it is a Particle MCMC kernel.
- The algorithm does not suffer from the path degeneracy problem due to the MCMC updates.

The MCMC step

(a) Sample $\tilde{\theta}$ from proposal kernel, $\tilde{\theta} \sim h(\theta, d\tilde{\theta})$.

(b) Run a new PF for $\tilde{\theta}$: sample independently $(\tilde{X}_{0:t}^{1:N_x}, \tilde{A}_{1:t}^{1:N_x})$ from $\psi_{t,\tilde{\theta}}$, and compute $\hat{L}_t(\tilde{\theta}, \tilde{X}_{0:t}^{1:N_x}, \tilde{A}_{1:t-1}^{1:N_x})$.

(c) Accept the move with probability

$$1 \wedge \frac{p(\tilde{\theta})\hat{L}_t(\tilde{\theta}, \tilde{X}_{0:t}^{1:N_x}, \tilde{A}_{1:t}^{1:N_x})h(\tilde{\theta}, \theta)}{p(\theta)\hat{L}_t(\theta, X_{0:t}^{1:N_x}, A_{1:t}^{1:N_x})h(\theta, \tilde{\theta})} .$$

It can be shown that this is a standard Hastings-Metropolis kernel with proposal

$$q_\theta(\tilde{\theta}, \tilde{x}_{0:t}^{1:N_x}, \tilde{a}_{1:t}^{1:N_x}) = h(\theta, \tilde{\theta})\psi_{t,\tilde{\theta}}(\tilde{x}_{0:t}^{1:N_x}, \tilde{a}_{1:t}^{1:N_x})$$

invariant w.r.t. to an extended distribution $\pi_t(\theta, x_{0:t}^{1:N_x}, a_{1:t}^{1:N_x})$.

Some advantages of the algorithm

- Immediate estimates of filtering and predictive distributions
- Immediate and sequential estimator of model evidence.
- Easy recovery of smoothing distributions.
- Principled framework for automatic calibration of N_x.
- Population Monte Carlo advantages.

Validity

SMC^2 is simply a SMC sampler with respect to the sequence:

$$\pi_t(\mathrm{d}\theta, \mathrm{d}x_{0:t}^{1:N_x}, \mathrm{d}a_{1:t}^{1:N_x})$$

- the reweigthing step $t-1 \to t$ (a) extends the dimension, by sampling $X_t^{1:N}, a_t^{1:N}$; and (b) computes $\pi_t(\cdot)/\pi_{t-1}(\cdot)$.
- The move step is a PMCMC step that leaves π_t invariant.

How to choose N_x?

PMCMC: valid whatever N_x, but one needs to take $N_x = O(T)$ in order to obtain a non-negligible acceptance rate. This is related to the following type of results (Cérou et al, 2011; Whiteley, 2011):

$$\mathrm{Var}[\hat{p}(y_{0:T}|\theta)] \leq \frac{CT}{N_x}.$$

For SMC^2, this suggests that one should start with a small value, then increases N_x progressively. But:

1. how to increase N_x at a given time?
2. when should we increase N_x?

How to increase N_x

Two possible strategies to replace our PF's of size N_x with PF's of size N'_x at iteration t:

1. exchange step: generate a new PF of size N'_x, then do an importance sampling step in order to swap the old PF and the new PF.
2. a CSMC (Particle Gibbs step), when we select one trajectory, throw away the $N_x - 1$ remaining ones, and regenerate $N'_x - 1$ new trajectories using CSMC.

The latter should suffer less from weigh degeneracy, but it suffers from a higher memory cost, i.e. $O(TN_xN_\theta)$ at time t.

When to increase N_x?

Currently, we monitor the acceptance rate of the PMCMC rejuvenation step; when it's too small, we trigger an exchange step (from N_x to $2N_x$).

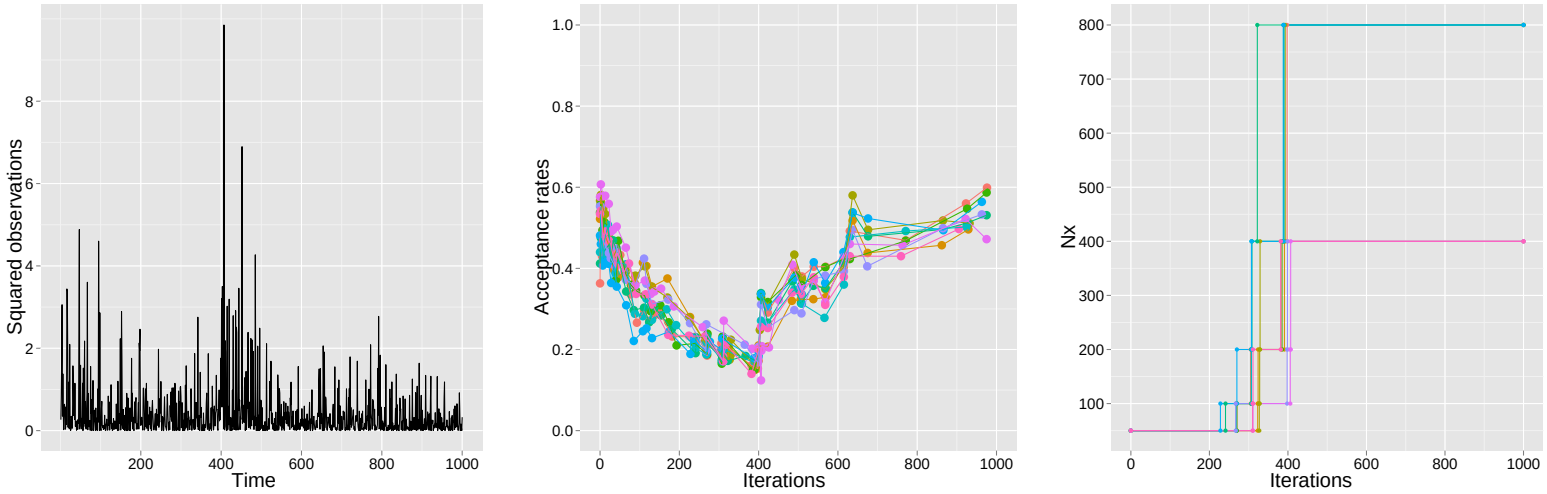
We're working on more refined versions based on PG steps, and better criteria to determine when and by how much we should increase N_x (on-going work).

Complexity

The overall complexity of SMC^2 is $O(N_\theta T^2)$ if run until time T:

1. The cost of iteration t without a rejuvenation step is $O(N_\theta N_x)$;
2. as explained before, we need to increase N_x progressively, $N_x = O(t)$;
3. The cost of the PMCMC rejuvenation step is $O(tN_\theta N_x)$, but we obtained the following result: if it is triggered whenever ESS$< \gamma$, and $N_x = O(t)$, then the occurence times are geometric $(\tau^k,\ k = 1, 2, \ldots)$.

Numerical illustrations: SV



(a) (b) (c)

Figure: Squared observations (synthetic data set), acceptance rates, and illustration of the automatic increase of N_x.

▸ See the model

Numerical illustrations: SV

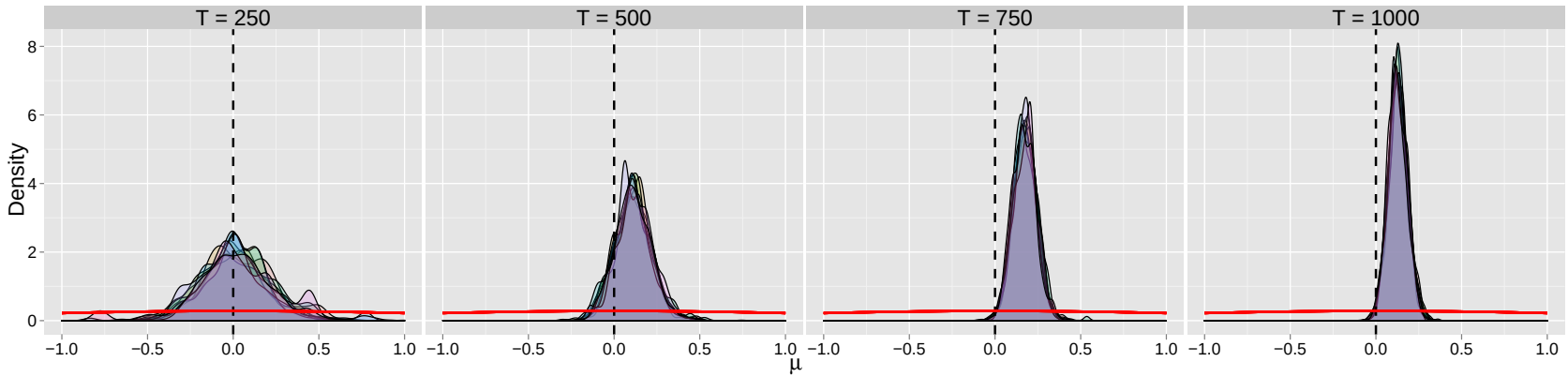


Figure: Concentration of the posterior distribution for parameter μ.

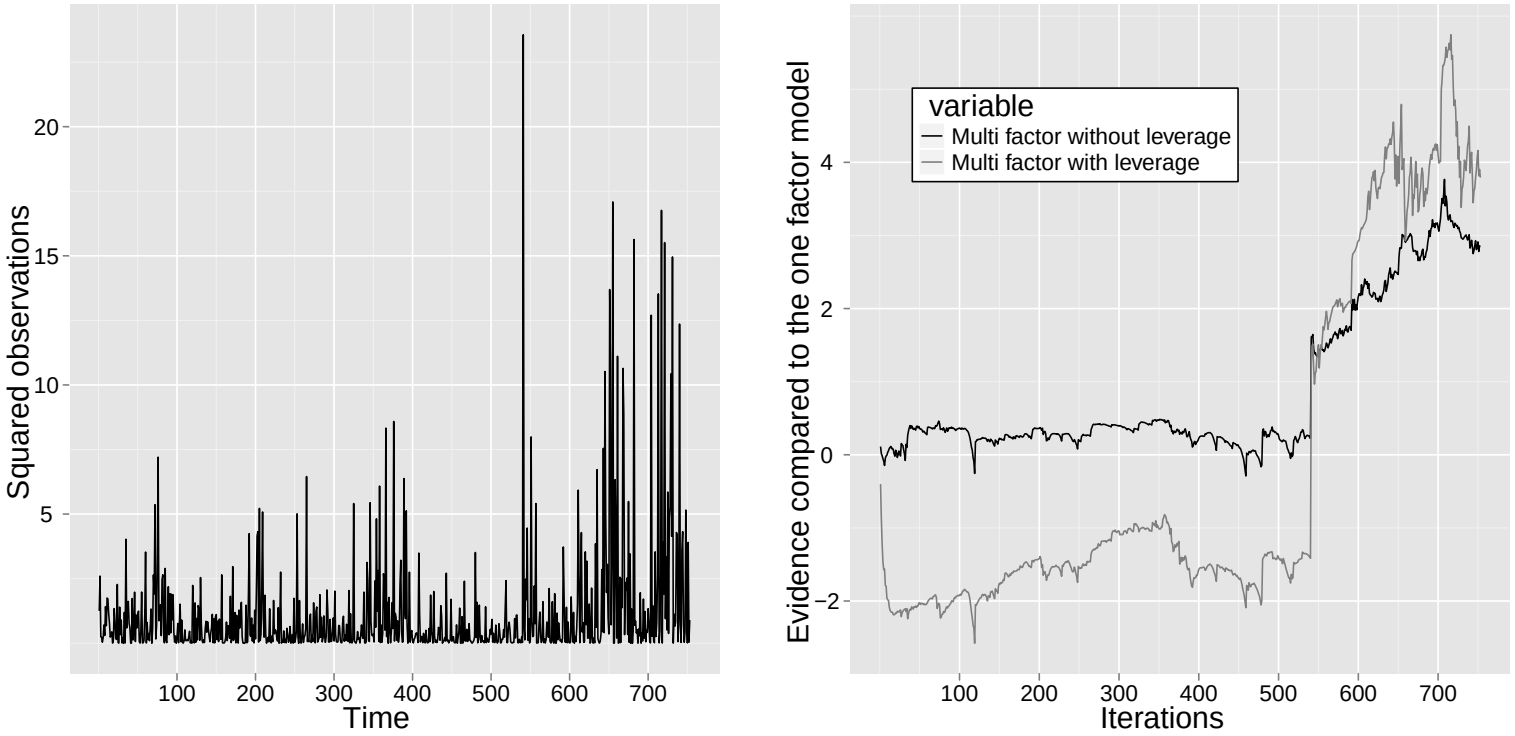
Numerical illustrations: SV

Multifactor model

$$y_t = \mu + \beta v_t + v_t^{1/2}\epsilon_t + \rho_1 \sum_{j=1}^{k_1} e_{1,j} + \rho_2 \sum_{j=1}^{k_2} e_{2,j} - \xi(w\rho_1\lambda_1 + (1-w)\rho_2\lambda_2)$$

where $v_t = v_{1,t} + v_{2,t}$, and $(v_i, z_i)_{n=1,2}$ are following the same dynamics with parameters $(w_i\xi, w_i\omega^2, \lambda_i)$ and $w_1 = w$, $w_2 = 1 - w$.

Numerical illustrations: SV



(a) (b)

Figure: S&P500 squared observations, and log-evidence comparison between models (relative to the one-factor model).

Athletics records model

$$g(y_{1:2,t}|\mu_t,\xi,\sigma) = \{1 - G(y_{2,t}|\mu_t,\xi,\sigma)\} \prod_{n=1}^{2} \frac{g(y_{i,t}|\mu_t,\xi,\sigma)}{1 - G(y_{i,t}|\mu_t,\xi,\sigma)}$$

$$x_t = (\mu_t, \dot{\mu}_t)', \quad x_{t+1} \mid x_t, \nu \sim \mathcal{N}(Fx_t, Q),$$

with

$$F = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \text{ and } Q = \nu^2 \begin{pmatrix} 1/3 & 1/2 \\ 1/2 & 1 \end{pmatrix}$$

$$G(y|\mu,\xi,\sigma) = 1 - \exp\left[-\left\{1 - \xi\left(\frac{y-\mu}{\sigma}\right)\right\}_+^{-1/\xi}\right]$$

Numerical illustrations

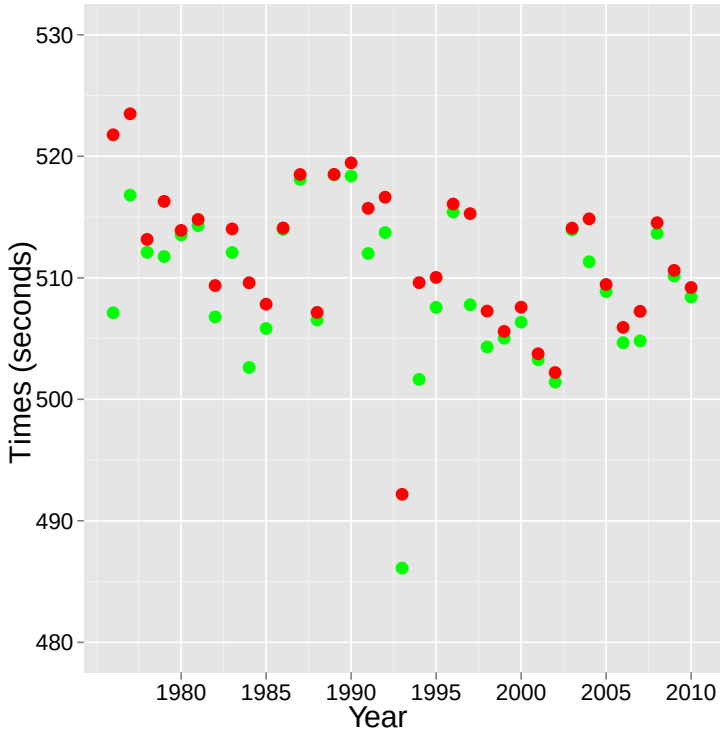


Figure: Best two times of each year, in women's 3000 metres events between 1976 and 2010.

Numerical illustrations: Athletics records

Motivating question

How unlikely is Wang Junxia's record in 1993?

A smoothing problem

We want to estimate the likelihood of Wang Junxia's record in 1993, given that we observe a better time than the previous world record. We want to use all the observations from 1976 to 2010 to answer the question.

Note

We exclude observations from the year 1993.

▸ See the model

Numerical illustrations

Some probabilities of interest

$$p_t^y = \mathbb{P}(y_t \leq y | y_{1976:2010})$$
$$= \int_\Theta \int_{\mathcal{X}} G(y|\mu_t, \theta) p(\mu_t | y_{1976:2010}, \theta) p(\theta | y_{1976:2010}) \, d\mu_t d\theta$$

The interest lies in $p_{1993}^{486.11}$, $p_{1993}^{502.62}$ and $p_t^{cond} := p_t^{486.11} / p_t^{502.62}$.

Numerical illustrations

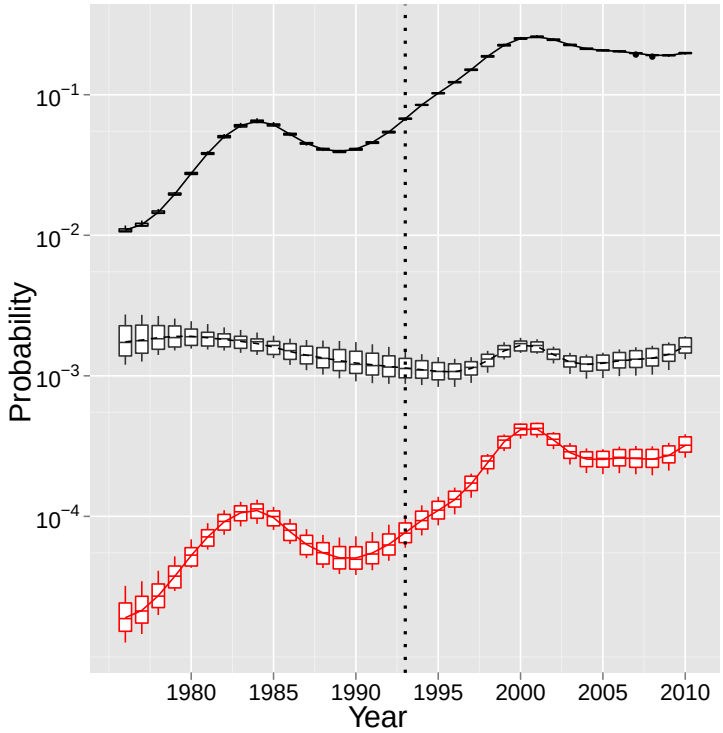


Figure: Estimates of the probability of interest (top) $p_t^{502.62}$, (middle) p_t^{cond} and (bottom) $p_t^{486.11}$, obtained with the SMC^2 algorithm. The y-axis is in log scale, and the dotted line indicates the year 1993 which motivated the study.