Particle Learning and Smoothing

Hedibert Freitas Lopes
The University of Chicago Booth School of Business

September 18th 2009
Instituto Nacional de Pesquisas Espaciais
São José dos Campos, Brazil

Joint with Nick Polson, Carlos Carvalho and Mike Johannes

Outline of the talk

Let the general dynamic model be

$$\begin{aligned} \text{Observation equation} &: p(y_{t+1}|x_{t+1}, \theta) \\ \text{System equation} &: p(x_{t+1}|x_t, \theta) \end{aligned}$$

We will talk about....

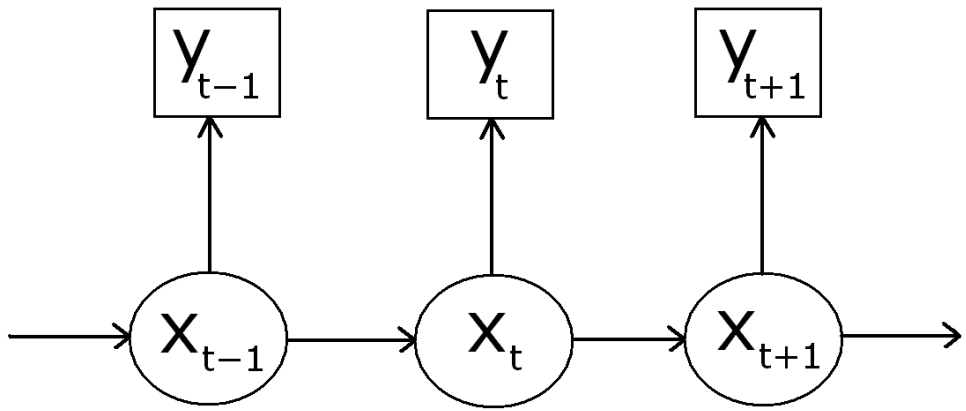
- MCMC in normal dynamic linear models.
- Particle filters: learning states x_{t+1}.
- Particle filters: learning parameters θ.
- **Particle Learning (PL) framework.**
- More general dynamic models.
- Final remarks.

Normal dynamic linear model (NDLM)

West and Harrison (1997):

$$\begin{aligned} y_{t+1}|x_{t+1} &\sim N(F_{t+1}x_{t+1}, \sigma^2_{t+1}) \\ x_{t+1}|x_t &\sim N(G_{t+1}x_t, \tau^2_{t+1}) \end{aligned}$$

Hidden Markovian structure



Sequential learning

$$p(x_t|y^t) \Longrightarrow p(x_{t+1}|y^t) \Longrightarrow p(y_{t+1}|x_t) \Longrightarrow p(x_{t+1}|y^{t+1})$$

where $y^t = (y_1, \ldots, y_t)$.

Example i. Local level model

Let $\theta = (\sigma^2, \tau^2)$, $x_0 \sim N(m_0, C_0)$ and

$$\begin{aligned} y_{t+1}|x_{t+1}, \theta &\sim N(x_{t+1}, \sigma^2) \\ x_{t+1}|x_t, \theta &\sim N(x_t, \tau^2) \end{aligned}$$

Kalman filter recursions

- Posterior at t: $(x_t|y^t) \sim N(m_t, C_t)$
- Prior at $t+1$: $(x_{t+1}|y^t) \sim N(m_t, R_{t+1})$
- Predictive at $t+1$: $(y_{t+1}|y^t) \sim N(m_t, Q_{t+1})$
- Posterior at $t+1$: $(x_{t+1}|y^{t+1}) \sim N(m_{t+1}, C_{t+1})$

where $R_{t+1} = C_t + \tau^2$, $Q_{t+1} = R_{t+1} + \sigma^2$, $A_{t+1} = R_{t+1}/Q_{t+1}$, $C_{t+1} = A_{t+1}\sigma^2$, and $m_{t+1} = (1 - A_{t+1})m_t + A_{t+1}y_{t+1}$.

Example i. Backward smoothing

For $t = n$, $x_n|y^n \sim N(m_n^n, C_n^n)$, where

$$\begin{aligned} m_n^n &= m_n \\ C_n^n &= C_n \end{aligned}$$

For $t < n$, $x_t|y^n \sim N(m_t^n, C_t^n)$, where

$$\begin{aligned} m_t^n &= (1 - B_t)m_t + B_t m_{t+1}^n \\ C_t^n &= (1 - B_t)C_t + B_t^2 C_{t+1}^n \end{aligned}$$

and

$$B_t = \frac{C_t}{C_t + \tau^2}$$

Example i. Backward sampling

For $t = n$, $x_n|y^n \sim N(a_n^n, R_n^n)$, where

$$\begin{aligned} a_n^n &= m_n \\ R_n^n &= C_n \end{aligned}$$

For $t < n$, $x_t|x_{t+1}, y^n \sim N(a_t^n, R_t^n)$, where

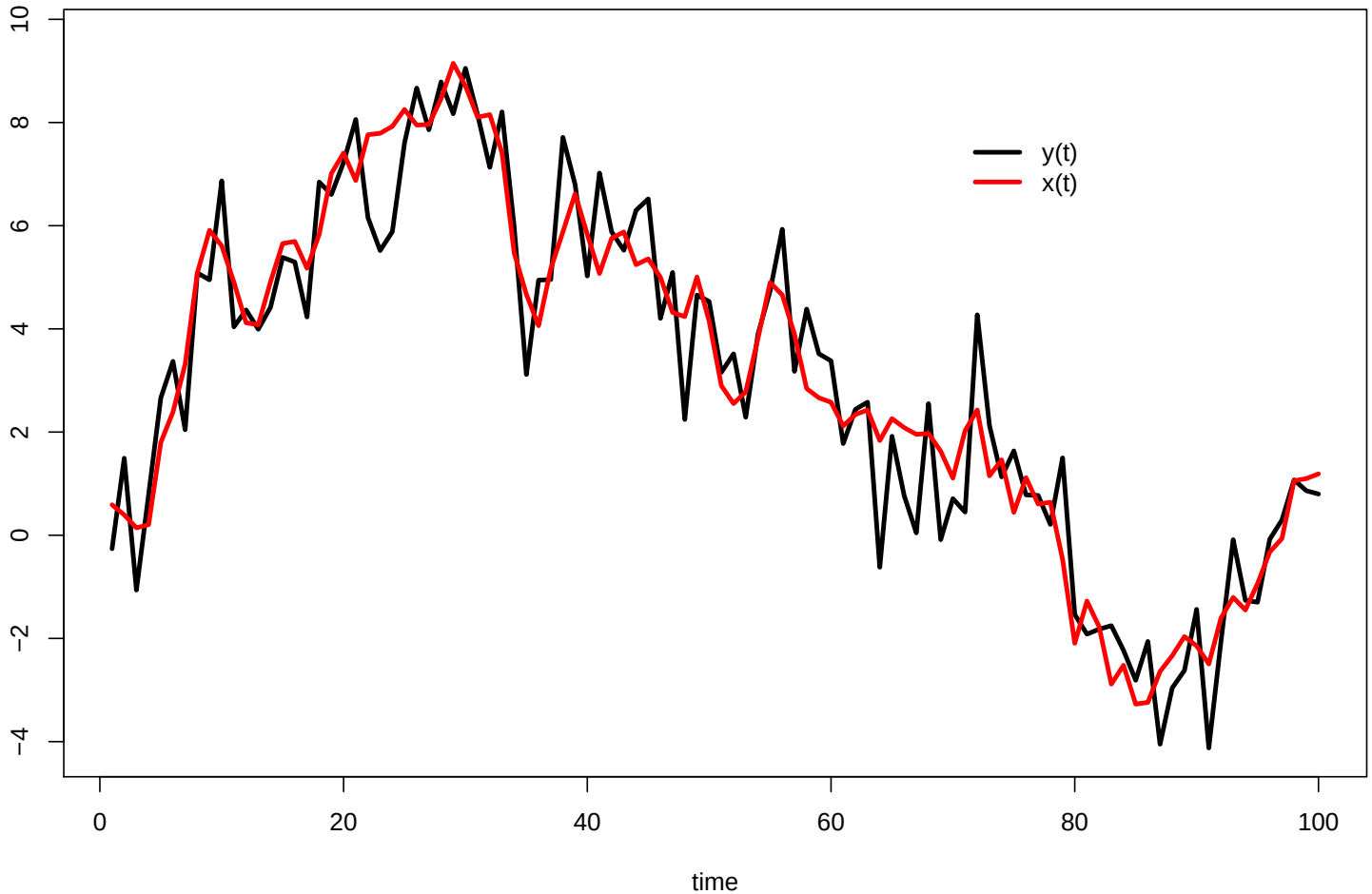
$$\begin{aligned} a_t^n &= (1 - B_t)m_t + B_t x_{t+1} \\ R_t^n &= B_t \tau^2 \end{aligned}$$

and

$$B_t = \frac{C_t}{C_t + \tau^2}$$

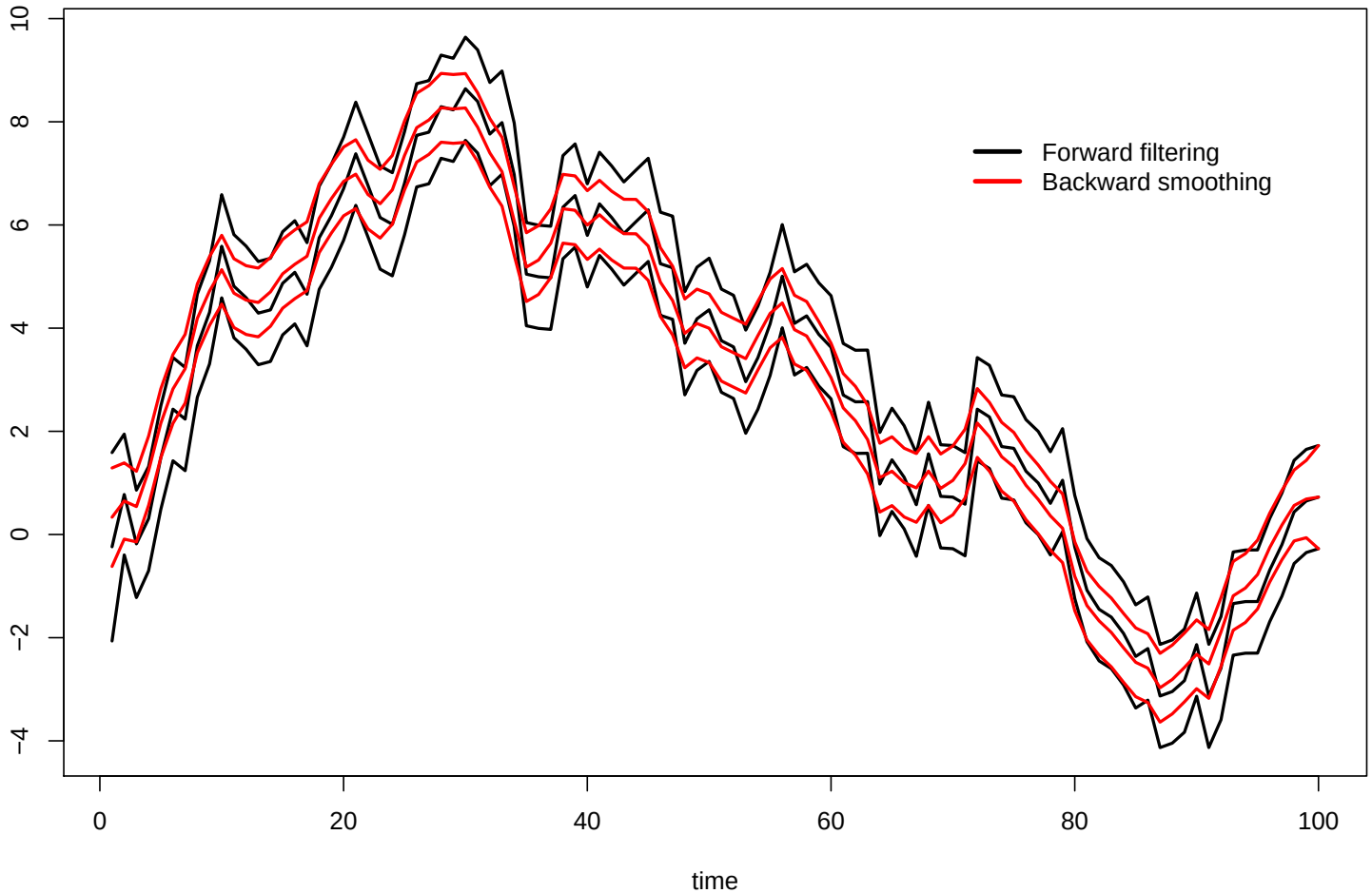
This is basically the Forward filtering, backward sampling (FFBS) algorithm commonly used to sample from $p(x^n|y^n)$ (Carter and Kohn, 1994 and Frühwirth-Schnatter, 1994).

Example i. $n = 100$, $\sigma^2 = 1.0$, $\tau^2 = 0.5$ and $x_0 = 0$



Example i. $p(x_t|y^t, \theta)$ and $p(x_t|y^n, \theta)$ for $t \leq n$.

$m_0 = 0.0$ and $C_0 = 10.0$



Non-Gaussian, nonlinear dynamic models

The dynamic model is

$$p(y_{t+1}|x_{t+1}) \quad \text{and} \quad p(x_{t+1}|x_t)$$

for $t = 1, \ldots, n$ and $p(x_0)$.

Prior and **posterior** at time $t+1$, i.e.

$$\begin{aligned} p(x_{t+1}|y^t) &= \int p(x_{t+1}|x_t)p(x_t|y^t)dx_t \\ p(x_{t+1}|y^{t+1}) &\propto p(y_{t+1}|x_{t+1})p(x_{t+1}|y^t) \end{aligned}$$

are usually **unavailable in closed form**.

Over the last 20 years:

- Carlin, Polson and Stoffer (1992) for more general DMs;
- Carter and Kohn (1994) and Frühwirth-Schnatter (1994) for conditionally Gaussian DLMs;
- Gamerman (1998) for generalized DLMs.

The Bayesian boostrap filter (BBF)

Gordon, Salmond and Smith (1993) use a **propagate-sample** scheme to go from $p(x_t|y^t)$ to $p(x_{t+1}|y^t)$ to $p(x_{t-1}|y^{t-1})$.

$$\{x_t^1, \ldots, x_t^N\} \sim p(x_t|y^t)$$

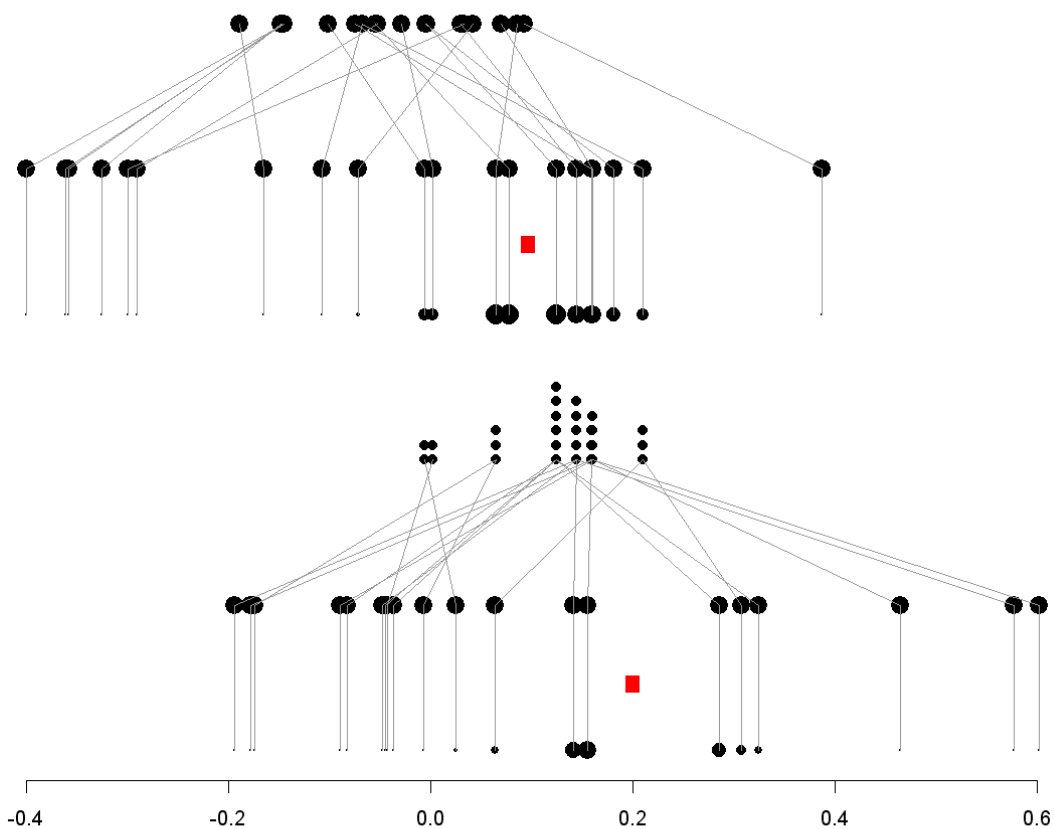
$$\tilde{x}_{t+1}^i \sim p(x_{t+1}|x_t^i)$$

$$\omega_{t+1}^i \propto p(y_{t+1}|\tilde{x}_{t+1}^i)$$

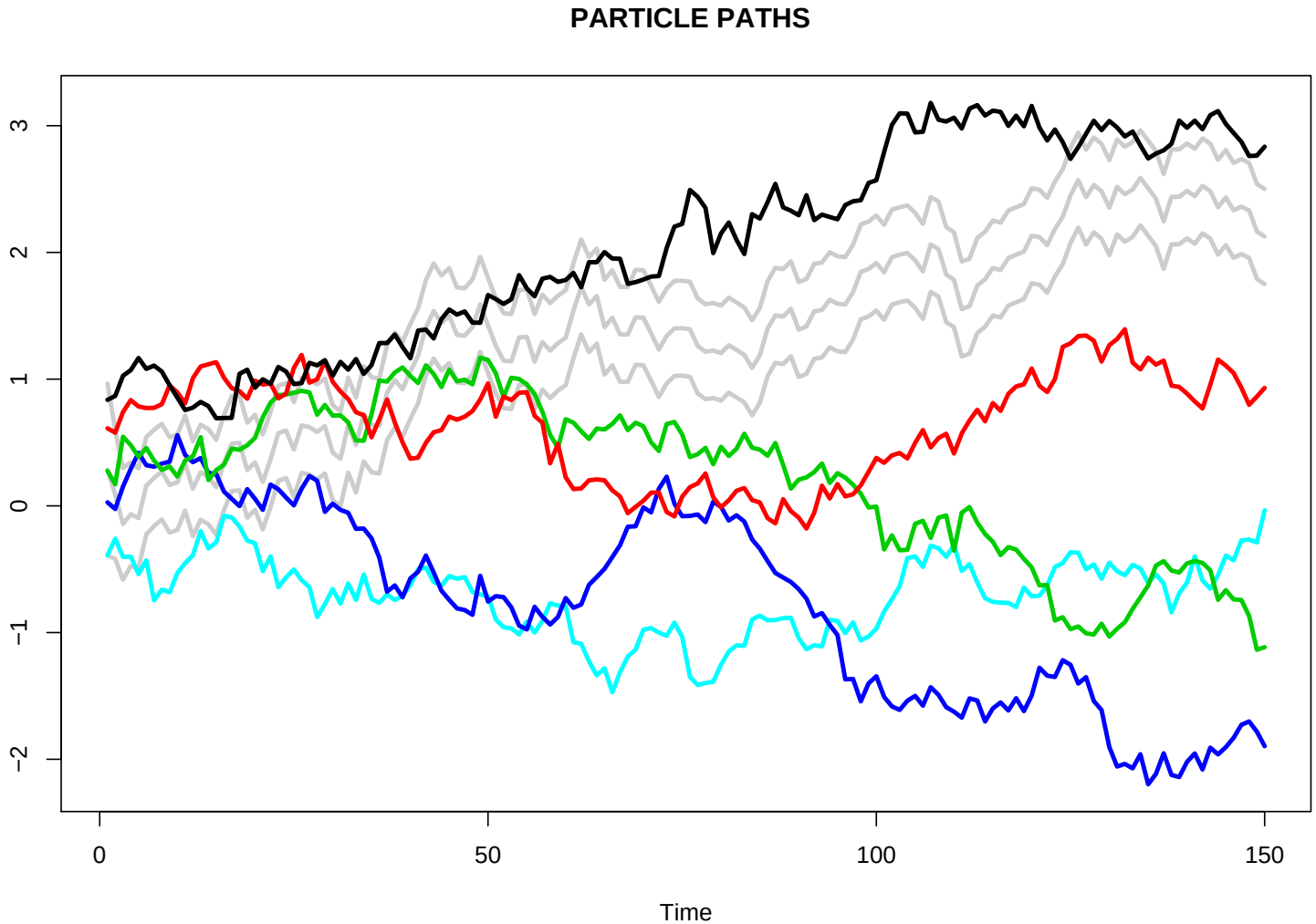
$$\{x_{t+1}^1, \ldots, x_{t+1}^N\} \sim p(x_{t+1}|y^{t+1})$$

$$\tilde{x}_{t+2}^i \sim p(x_{t+2}|x_{t+1}^i)$$

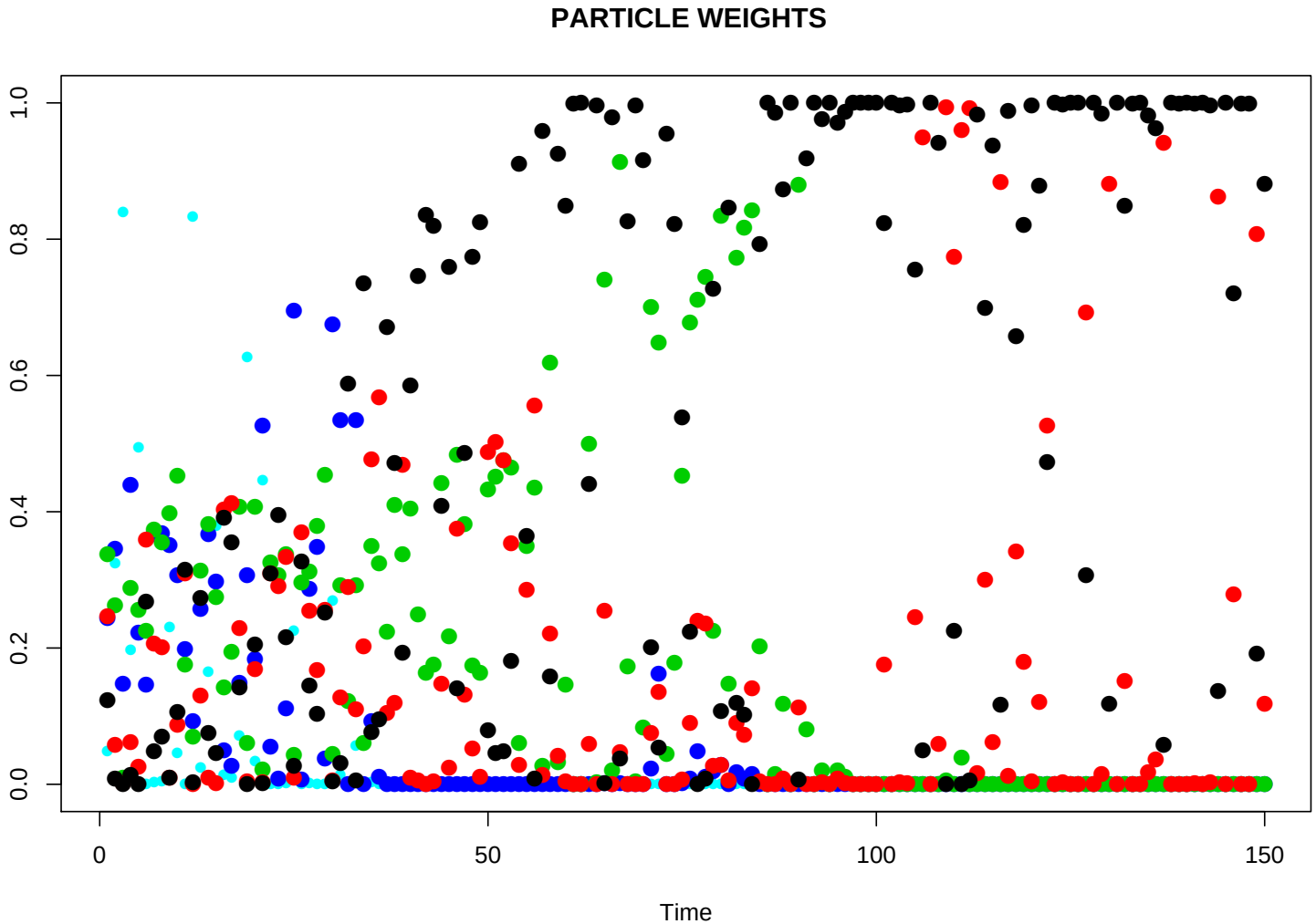
$$\omega_{t+2}^i \propto p(y_{t+2}|\tilde{x}_{t+2}^i)$$



Resample or not resample?



Weights



Fully adapted BBF (FABBF)

- Posterior at t: $\{x_t^{(i)}\}_{i=1}^N \sim p(x_t|y^t)$.
- **Propagate** Draw $\{\tilde{x}_{t+1}^{(i)}\}_{i=1}^N$ from

$$p(x_{t+1}|x_t^{(i)}, y_{t+1}).$$

- **Resample** Draw $\{x_{t+1}^{(j)}\}_{j=1}^N$ from $\{\tilde{x}_{t+1}^{(i)}\}_{i=1}^N$ with weights

$$w_{t+1}^{(i)} \propto p(y_{t+1}|x_t^{(i)}).$$

- Posterior at $t+1$: $\{x_{t+1}^{(i)}\}_{i=1}^N \sim p(x_{t+1}|y^{t+1})$.

The auxiliary particle filter (APF)

- Posterior at t: $\{x_t^{(i)}\}_{i=1}^N \sim p(x_t|y^t)$.
- **Resample** Draw $\{\tilde{x}_t^{(j)}\}_{j=1}^N$ from $\{x_t^{(i)}\}_{i=1}^N$ with weights

$$w_{t+1}^{(i)} \propto p(y_{t+1}|g(x_t^{(i)}))$$

  where $g(x_t) = E(x_t|x_{t-1})$.
- **Propagate** Draw $\{x_{t+1}^{(i)}\}_{i=1}^N$ from

$$p(x_{t+1}|\tilde{x}_t^{(i)})$$

  and resample (SIR) with weights

$$\omega_{t+1}^{(j)} \propto \frac{p(y_{t+1}|x_{t+1}^{(j)})}{p(y_{t+1}|g(x_t^{(k^j)}))}.$$

- Posterior at $t+1$: $\{x_{t+1}^{(i)}\}_{i=1}^N \sim p(x_{t+1}|y^{t+1})$.

How about $p(\theta|y^n)$?

Two-step strategy: On the first step, approximate $p(\theta|y^n)$ by

$$p^N(\theta|y^n) = \frac{p^N(y^n|\theta)p(\theta)}{p(y^n)} \propto p^N(y^n|\theta)p(\theta)$$

where $p^N(y^n|\theta)$ is a SMC approximation to $p(y^n|\theta)$. Then, on the 2nd step, sample θ via a MCMC scheme or a SIR scheme[1].

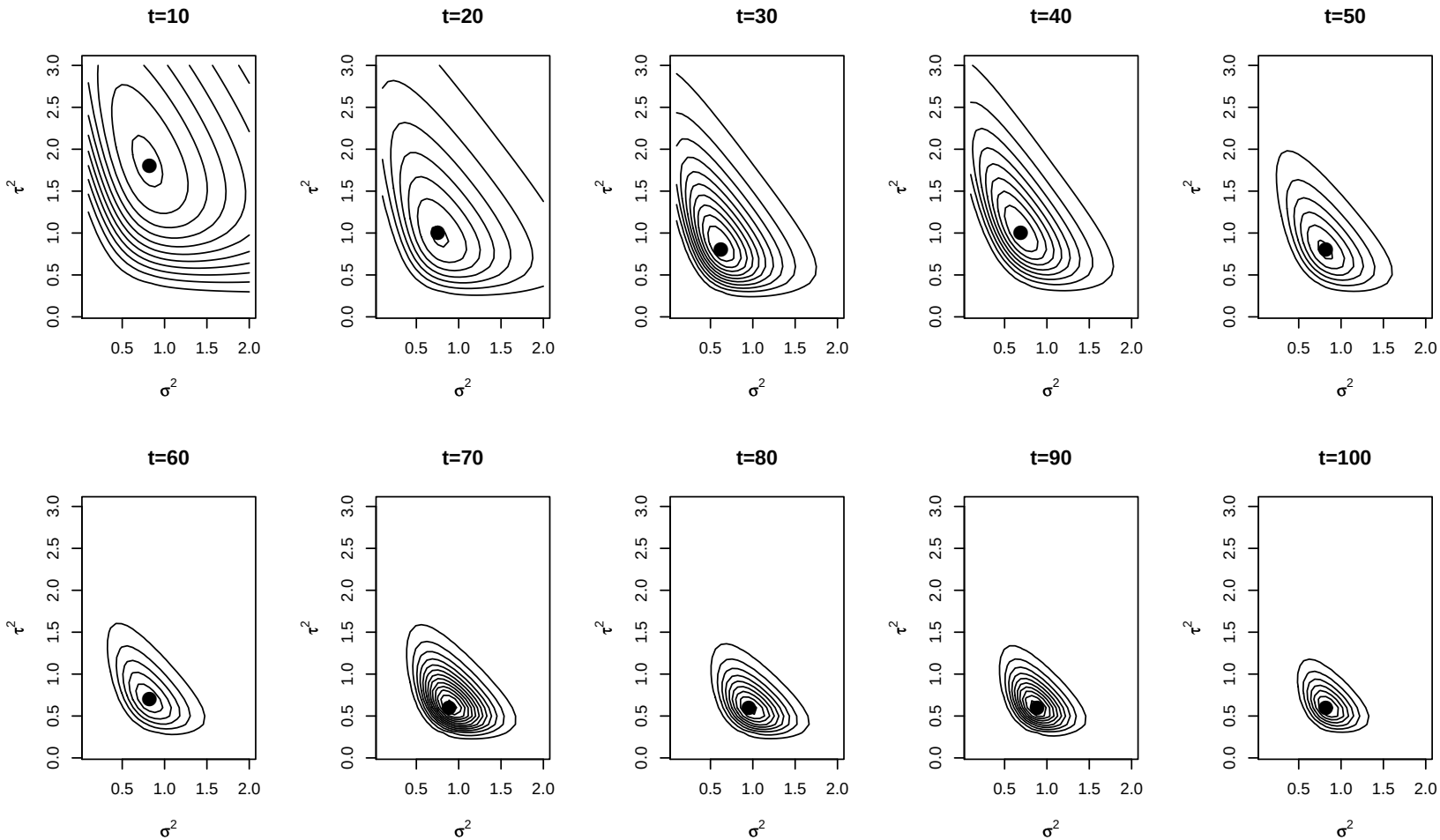
Problem 1: SMC looses its appealing sequential nature.

Problem 2: Overall sampling scheme is sensitive to $p^N(y|\theta)$.

[1] See Fernándes-Villaverde and Rubio-Ramírez (2007) "Estimating Macroeconomic Models: A Likelihood Approach", DeJong, Dharmarajan, Liesenfeld, Moura and Richard (2009) "Efficient Likelihood Evaluation of State-Space Representations" for applications of this two-step strategy to DSGE and related models.

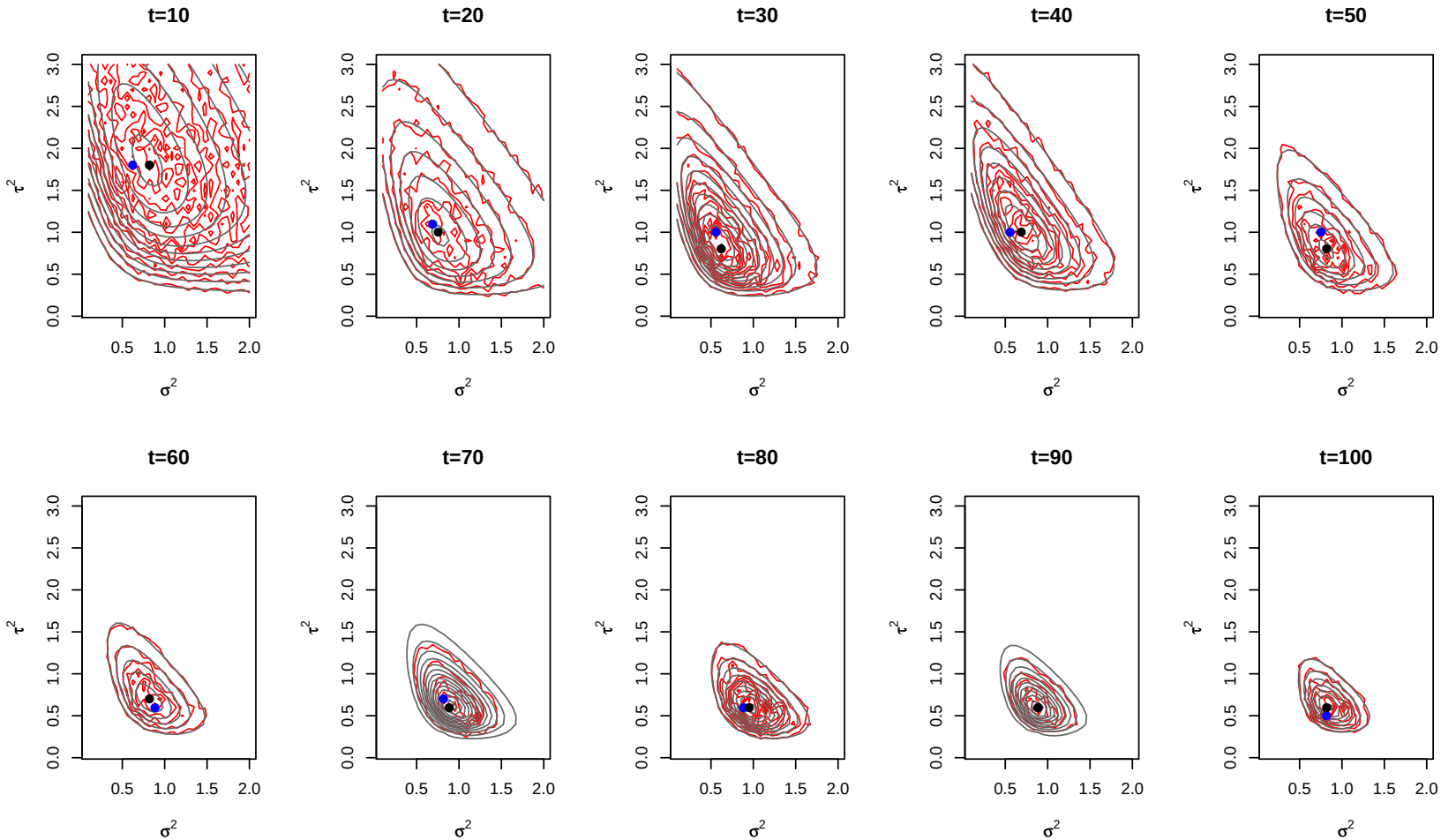
Example ii. Exact integrated likelihood $p(y^n|\sigma^2, \tau^2)$

$n = 100, x_0 = 0, \sigma^2 = 1, \tau^2 = 0.5$ and $x_0 \sim N(0.0, 100)$
30×30 grid: $\sigma^2 = (0.1, \ldots, 2)$ and $\tau^2 = (0.1, \ldots, 3)$



Example ii. Approximated $p^N(y^n|\sigma^2, \tau^2)$

Based on $N = 1000$ particles



Another (perhaps more natural) idea

Sequentially learning x_t and θ.

$$\begin{aligned}
\text{Posterior at t} &: p(x_t|\theta, y^t)p(\theta|y^t) \\
&\Downarrow \\
\text{Prior ate t+1} &: p(x_{t+1}|\theta, y^t)p(\theta|y^t) \\
&\Downarrow \\
\text{Posterior at t+1} &: p(x_{t+1}|\theta, y^{t+1})p(\theta|y^{t+1})
\end{aligned}$$

Advantages:

Sequential updates of $p(\theta|y^t)$, $p(x_t|y^t)$ and $p(\theta, x_t|y^t)$

Sequential h-steps ahead forecast $p(y_{t+h}|y^t)$

Sequential approximations for $p(y_t|y^{t-1})$

Sequential Bayes factors

$$B_{12t} = \frac{\prod_{j=1}^{t} p(y_j|y^{j-1}, M_1)}{\prod_{j=1}^{t} p(y_j|y^{j-1}, M_2)}$$

Liu and West filter (LWF)

$\{(x_t, \theta)^{(i)}\}_{i=1}^N \sim p(x_t, \theta | y^t)$.

Compute $\bar{\theta}$, V and $m(\theta^{(i)}) = a\theta^{(i)} + (1-a)\bar{\theta} \ \ \forall i$.

Resampling

- Compute $g(x_t^{(i)}) = E(x_{t+1} | x_t^{(i)}, m(\theta^{(i)})) \ \ \forall i$.
- Compute $w_{t+1}^{(i)} = p(y_{t+1} | g(x_t^{(j)}), m(\theta^{(j)})) \ \ \forall i$.
- Resample $(\tilde{x}_t, \tilde{\theta})^{(i)}$ from $\{(x_t, \theta, w_{t+1})^{(j)}\}_{j=1}^N$.

Sampling

- Sample $\tilde{\tilde{\theta}}^{(i)} \sim N(m(\tilde{\theta}^{(i)}), (1-a^2)V) \ \ \forall i$
- Sample $\tilde{x}_{t+1}^{(i)} \sim p(x_{t+1} | \tilde{x}_t^{(i)}, \tilde{\tilde{\theta}}^{(i)}) \ \ \forall i$.
- Compute $\omega_{t+1}^{(i)} = p(y_{t+1} | \tilde{x}_{t+1}^{(i)}, \tilde{\tilde{\theta}}^{(i)}) / p(y_{t+1} | g(\tilde{x}_t^{(i)}), m(\tilde{\theta}^{(i)})) \forall i$.
- Resample $(x_{t+1}, \theta)^{(i)}$ from $\{(\tilde{x}_{t+1}, \tilde{\theta}, \omega_{t+1})^{(j)}\}_{j=1}^N$.

$\{(x_{t+1}, \theta)^{(i)}\}_{i=1}^N \sim p(x_{t+1}, \theta | y^{t+1})$.

Choosing a

They approximate $p(\theta|y^t)$ by a N-component mixture of normals

$$p(\theta|y^t) = \sum_{i=1}^{N} \omega_t^{(i)} f_N(\theta|a\theta^{(i)} + (1-a)\bar{\theta}, (1-a^2)V)$$

where $\bar{\theta}$ and V approximate mean and variance of $(\theta|y^t)$.

A discount factor argument is used to set up the shrinkage quantity

$$a = \frac{3\delta - 1}{2\delta}.$$

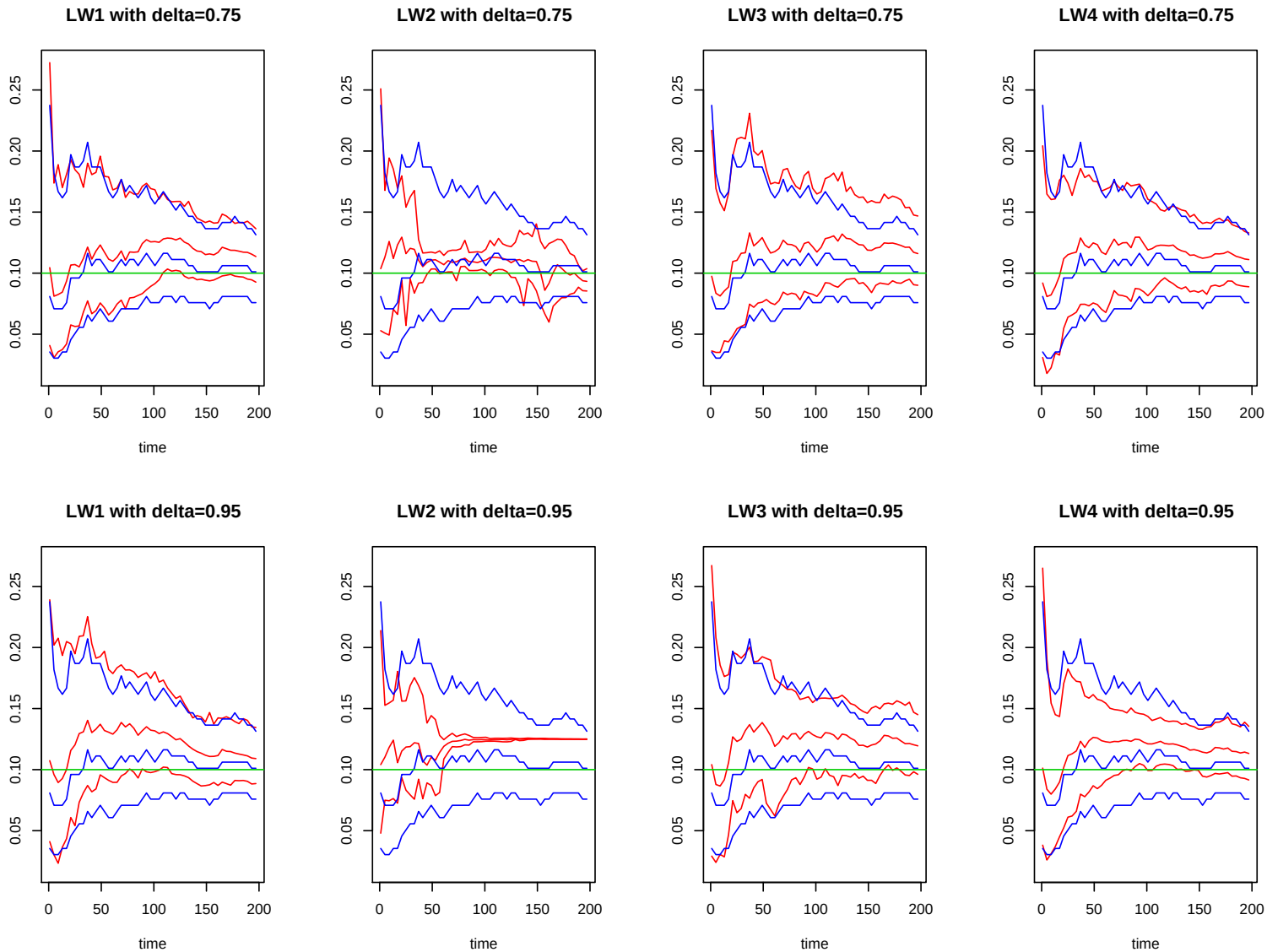
For example,

- $\delta = 0.50$ leads to $a = 0.500$
- $\delta = 0.75$ leads to $a = 0.833$
- $\delta = 0.95$ leads to $a = 0.974$
- $\delta = 1.00$ leads to $a = 1.000$.

When $a = 1.0$, the particles will degenerate over time.

Example iii. $N = 2000, x_0 = 25, \sigma^2 = .1, \tau^2 = .05$

LW1:log σ^2, LW2:σ^2, LW3:LW1 + o.p. LW4:LW2 + o.p.[2]



[2]o.p.=optimal propagation

Particle Learning (PL)

- Advantages:
  - Sequentially learning about (x_t, θ);
  - A real/practical alternative to MCMC methods;
  - Sequential model assessment;
  - Applicable in a wide range of dynamic and static models.

- Key issues in our approach:
  - Reverse the Kalman filter logic: resample/propagate;
  - Estimation of "fixed", unknown parameters;
  - Work with conditional sufficient statistics;
  - Use SMC for smoothing.

Parameter learning[3]

It is assumed that

$$p\left(\theta|x^t, y^t\right) = p\left(\theta|s_t\right),$$

where s_t is a recursively defined sufficient statistic (SS),

$$s_{t+1} = \mathcal{S}\left(s_t, x_{t+1}, y_{t+1}\right).$$

- SS are just another state with a deterministic evolution;
- "Filtering" SS provides a mechanism for replenishing the parameters avoiding degeneracy;
- Reduction of the variance of re-sampling weights: Rao-Blackwellization.

[3] Storvik (2002) and Fearnhead (2002)

Reverse the Kalman filter logic

Traditional logic for filtering (Kalman, 1960): Predict/Update

$$p\left(x_{t+1}|y^t\right) = \int p\left(x_{t+1}|x_t\right) p\left(x_t|y^t\right) dx_t$$
$$p\left(x_{t+1}|y^{t+1}\right) \propto p\left(y_{t+1}|x_{t+1}\right) p\left(x_{t+1}|y^t\right).$$

By Bayes rule, we can reverse this logic through:

$$p\left(x_t|y^{t+1}\right) \propto p\left(y_{t+1}|x_t\right) p\left(x_t|y^t\right)$$
$$p\left(x_{t+1}|y^{t+1}\right) = \int p\left(x_{t+1}|x_t, y_{t+1}\right) p\left(x_t|y^{t+1}\right) dx_t$$

• We will first re-sample (smooth) and then propagate. Since information in y_{t+1} is used in both steps, the algorithm will be more efficient.

The general approach

- Assume at time t, $\left\{(x_t, s_t)^{(i)}\right\}_{i=1}^{N}$ approximates $p^N(x_t, s_t|y^t)$;

- Once y_{t+1} is observed, the re-sample/propagation rule is
  - Resampling

$$p\left(x_t, s_t|y^{t+1}\right) \propto p\left(y_{t+1}|x_t, s_t\right) p\left(x_t, s_t|y^t\right)$$

  - Propagation

$$p\left(x_{t+1}|y^{t+1}\right) = \int p\left(x_{t+1}|x_t, s_t, y_{t+1}\right) p\left(x_t, s_t|y^{t+1}\right) dx_t ds_t$$

Perfect adaptation

Target: $p\left(x_{t+1}, x_t | y^{t+1}\right)$

IS weights:

$$w \propto \frac{p\left(x_{t+1}|x_t, y_{t+1}\right) p\left(y_{t+1}|x_t\right) p\left(x_t|y^t\right)}{q_1\left(x_t|y_{t+1}\right) q_2\left(x_{t+1}|x_t, y_{t+1}\right)}$$

$$w \propto \frac{p\left(x_{t+1}|x_t, y_{t+1}\right) p\left(y_{t+1}|x_t\right) p\left(x_t|y^t\right)}{p\left(x_t|y_{t+1}\right) p\left(x_{t+1}|x_t, y_{t+1}\right) p\left(x_t|y^t\right)}$$

$$= 1$$

- This is the ideal scenario and should serve as a guiding principle in the construction of sequential Monte Carlo filters[4].
- Marginalization is key!

[4] Perfect adaptation is also discussed in Pitt and Shephard (1999) and Johansen and Doucet (2008).

The general PL algorithm

- Posterior at t: $\Phi_t \equiv \left\{(x_t, \theta)^{(i)}\right\}_{i=1}^{N} \sim p(x_t, \theta | y^t)$.
- Compute, for $i = 1, \ldots, N$,

$$w_{t+1}^{(i)} \propto p(y_{t+1} | x_t^{(i)}, \theta^{(i)})$$

- Resample from Φ_t with weights w_{t+1}: $\tilde{\Phi}_t \equiv \left\{(\tilde{x}_t, \tilde{\theta})^{(i)}\right\}_{i=1}^{N}$.
- Propagate states

$$x_{t+1}^{(i)} \sim p(x_{t+1} | \tilde{x}_t^{(i)}, \tilde{\theta}^{(i)}, y_{t+1})$$

- Update sufficient statistics

$$s_{t+1}^{(i)} = \mathcal{S}(s_t^{(i)}, x_{t+1}^{(i)}, y_{t+1})$$

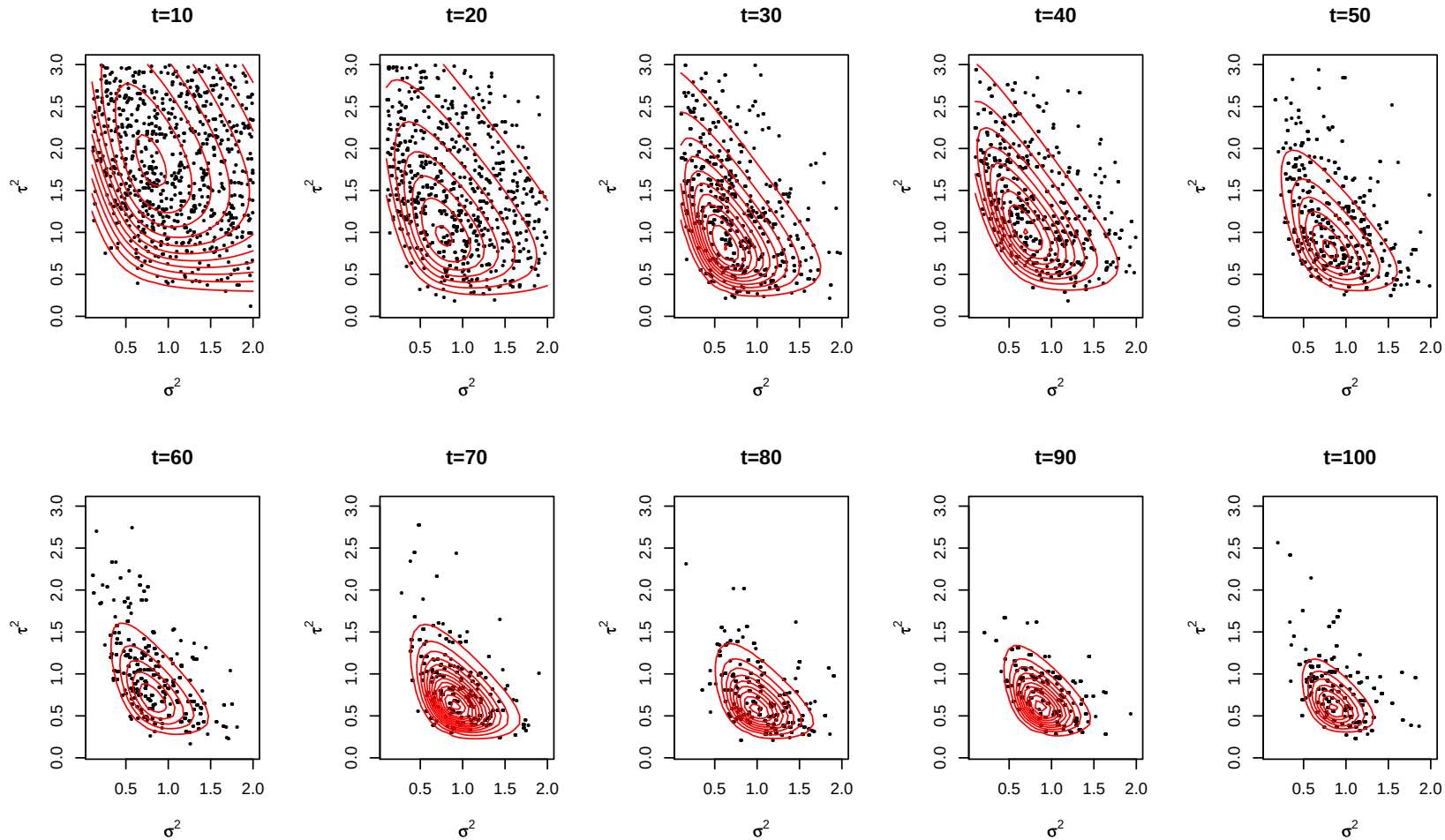
- Sample parameters

$$\theta^{(i)} \sim p(\theta | s_{t+1}^{(i)})$$

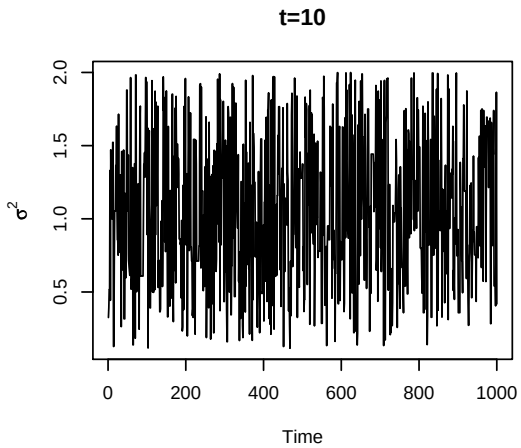
Example ii. $p^N(\theta|y^t)$ via PL for x_t and MCMC for (σ^2, τ^2)

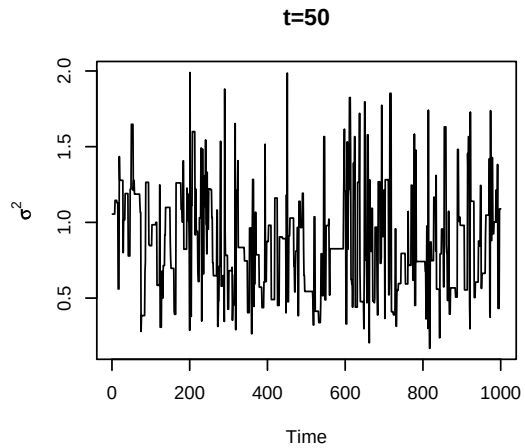
Prior: $\sigma^2 \sim U(0.1, 2)$ and $\tau^2 \sim U(0.1, 3)$.

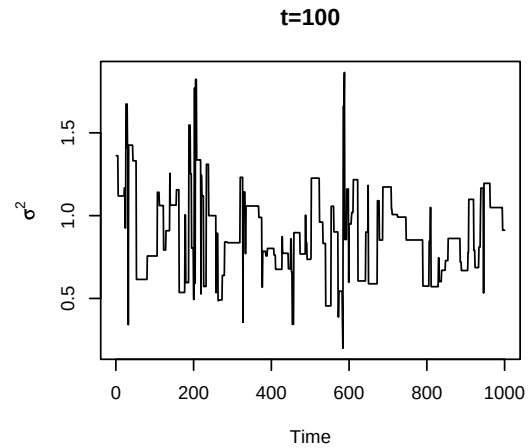
Based on $N = 1000$ particles.

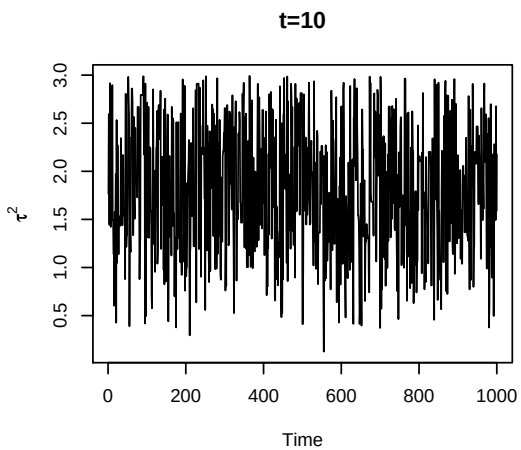


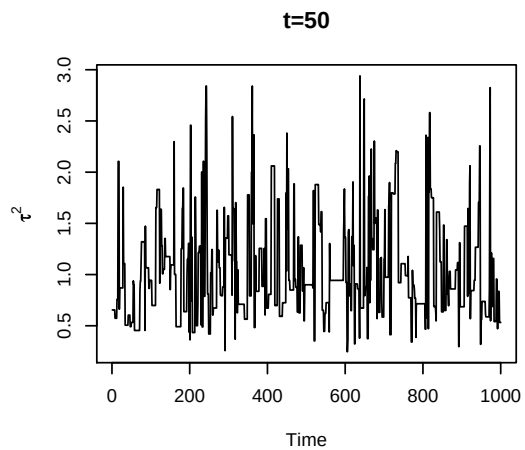
Example ii. Trace plots for σ^2 and τ^2

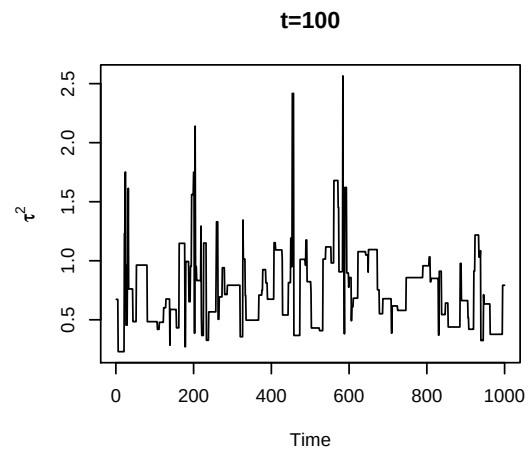








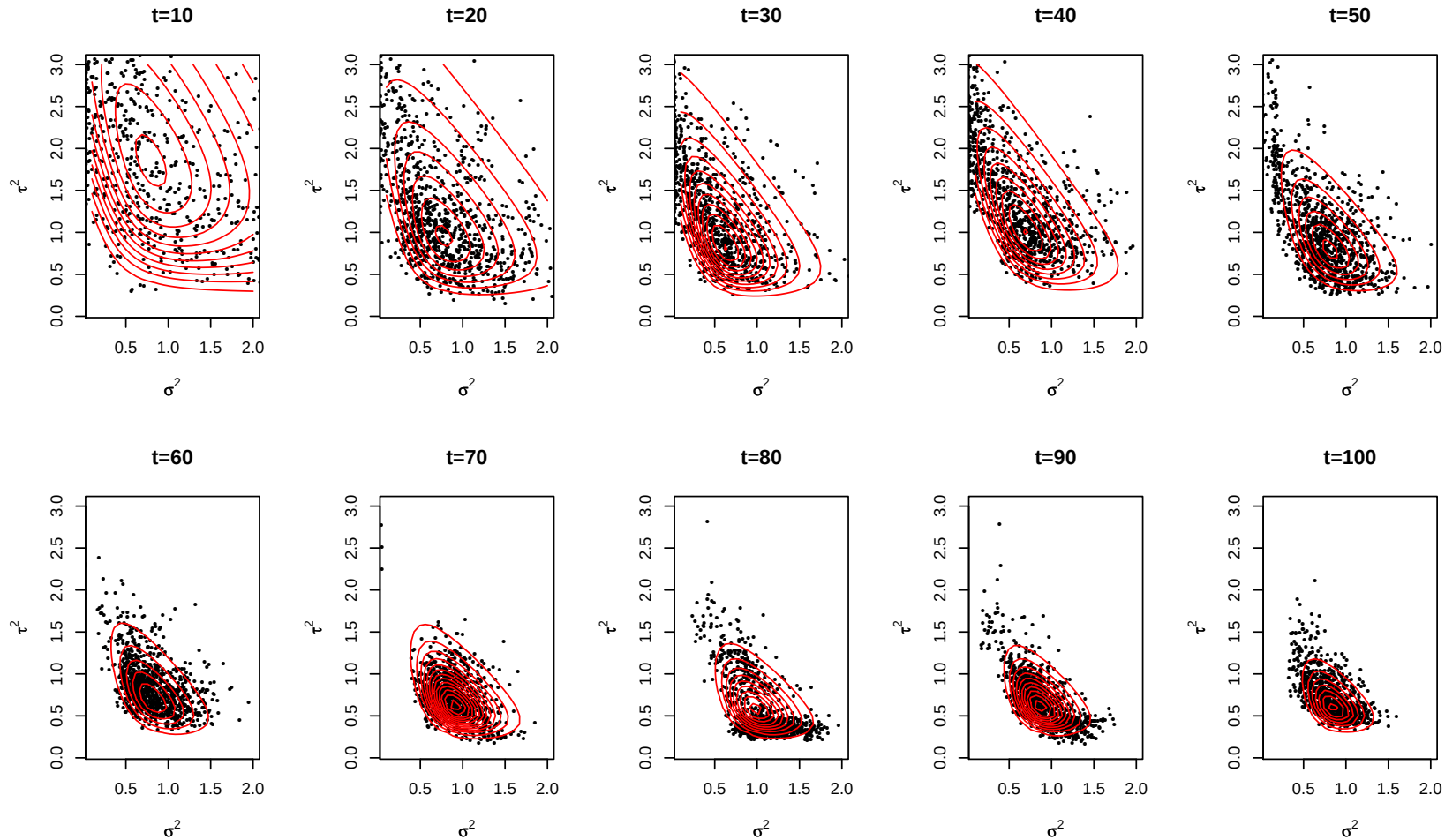




Example ii. $p^N(\theta|y^t)$ via PL for (x_t, σ^2, τ^2)

Prior: $\sigma^2 \sim U(0.1, 2)$ and $\tau^2 \sim U(0.1, 3)$.

Based on $N = 1000$ particles.



Example iv. Comparing BBF, APF, FABBF and PL

Simulation
$M = 20$ data sets with $n = 100$ observations each, $\tau^2 = 0.013$ and $\sigma^2 \in \{5, 0.13, 0.013, 0.0065, 0.0013\}$ from

$$\begin{aligned} y_{t+1}|x_{t+1} &\sim N(x_{t+1}, \sigma^2) \\ x_{t+1}|x_t &\sim N(x_t, \tau^2) \end{aligned}$$

with $x_0 = 0$.

Particle filters: $R = 20$ replications of $N = 1000$ particles.

Prior set up: $x_0 \sim N(0, 10)$.

Example iv. Log relative mean square error

Let q_α^t be such that

$$Pr(x_t < q_t^\alpha | y^t) = \alpha$$

and $\alpha = (0.05, 0.25, 0.5, 0.75, 0.95)$.

Then, the average MSE for filter f, time t and quantile 100α-percentile is

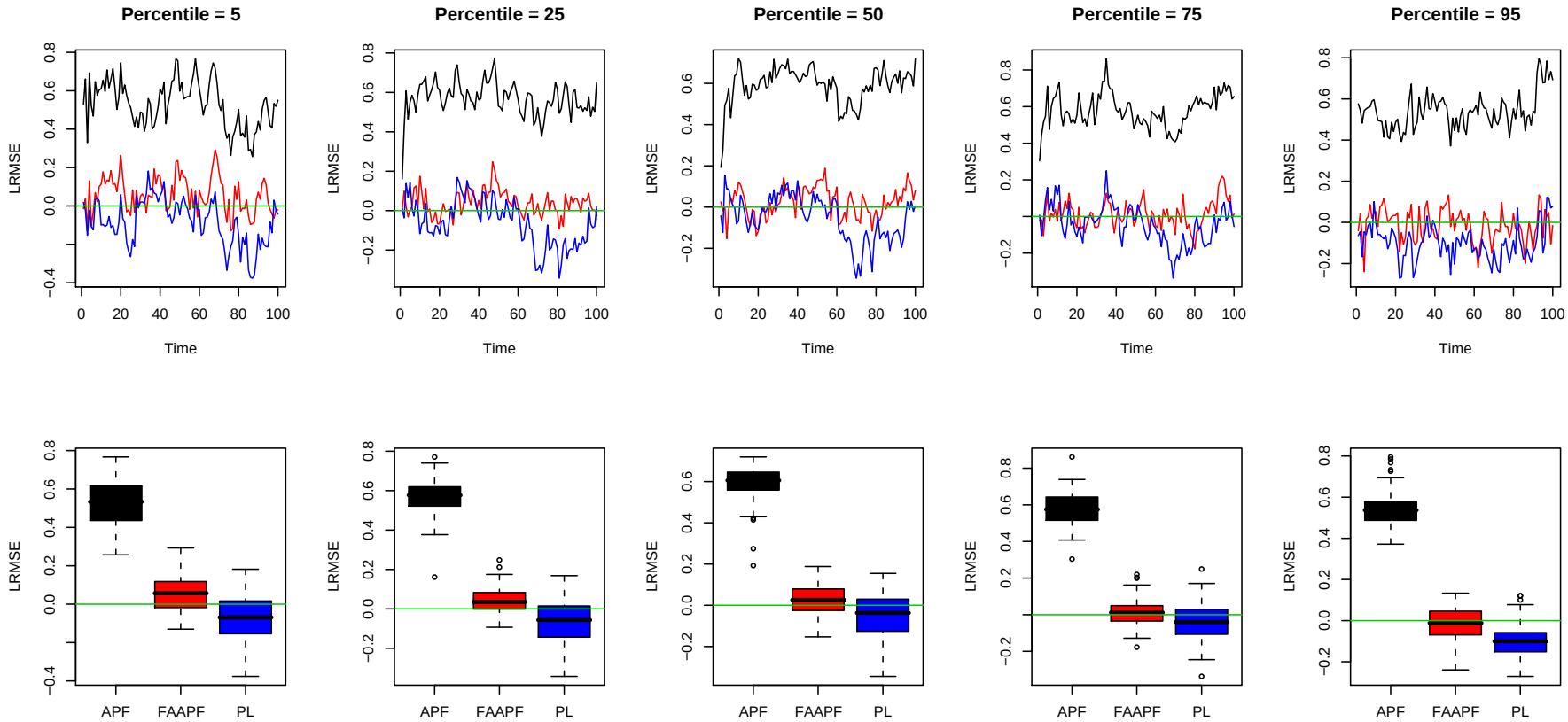
$$MSE_{t,f}^\alpha = \frac{1}{MR} \sum_{i=1}^{M} \sum_{j=1}^{R} (\hat{q}_{tij,f}^\alpha - q_{it}^\alpha)^2$$

We compare filters f_1 and f_2 based on log relative MSE, i.e.

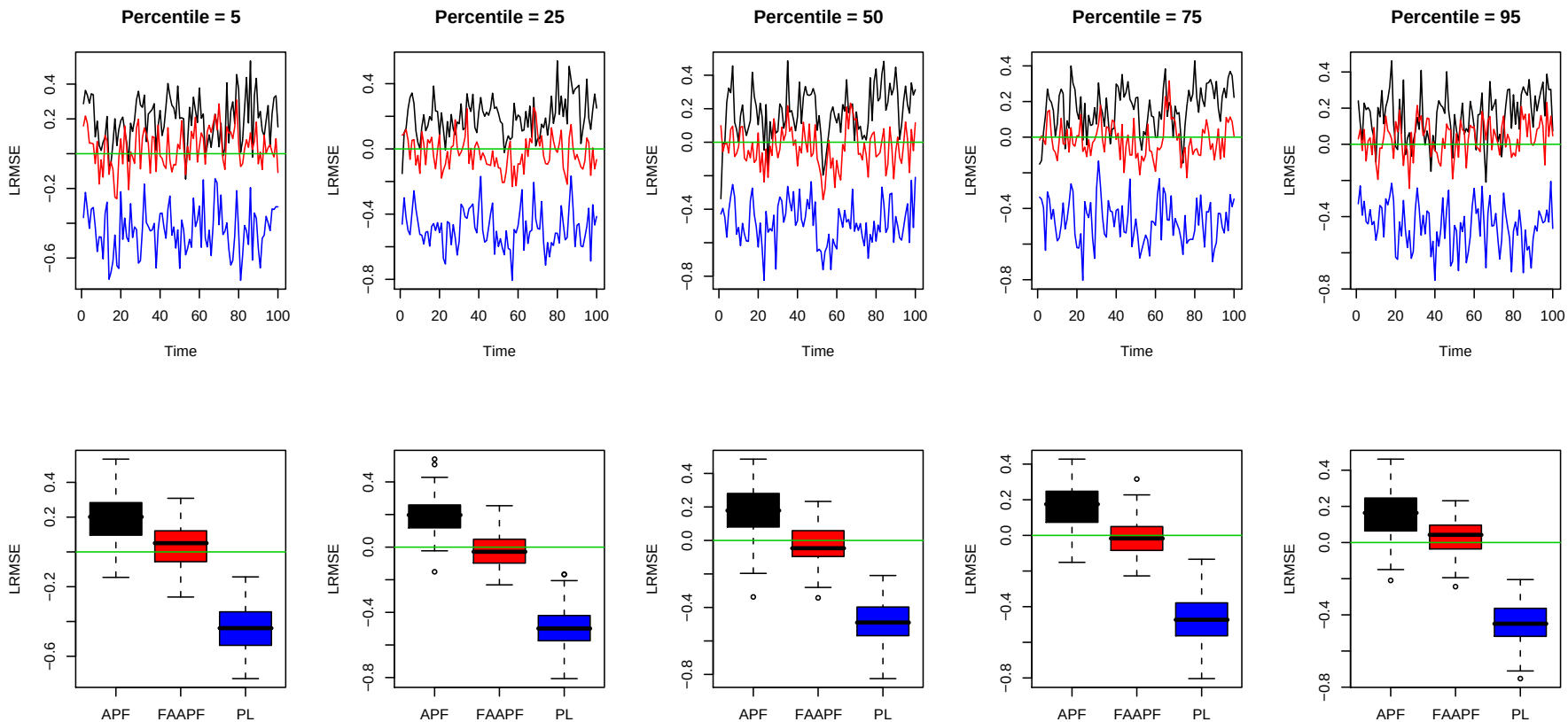
$$LRMLE_{t,f_1,f_2}^\alpha = \log MSE_{t,f_1}^\alpha - MSE_{t,f_2}^\alpha$$

Example iv. $\sigma^2 = 5.0$ and $\tau/\sigma = 0.05$

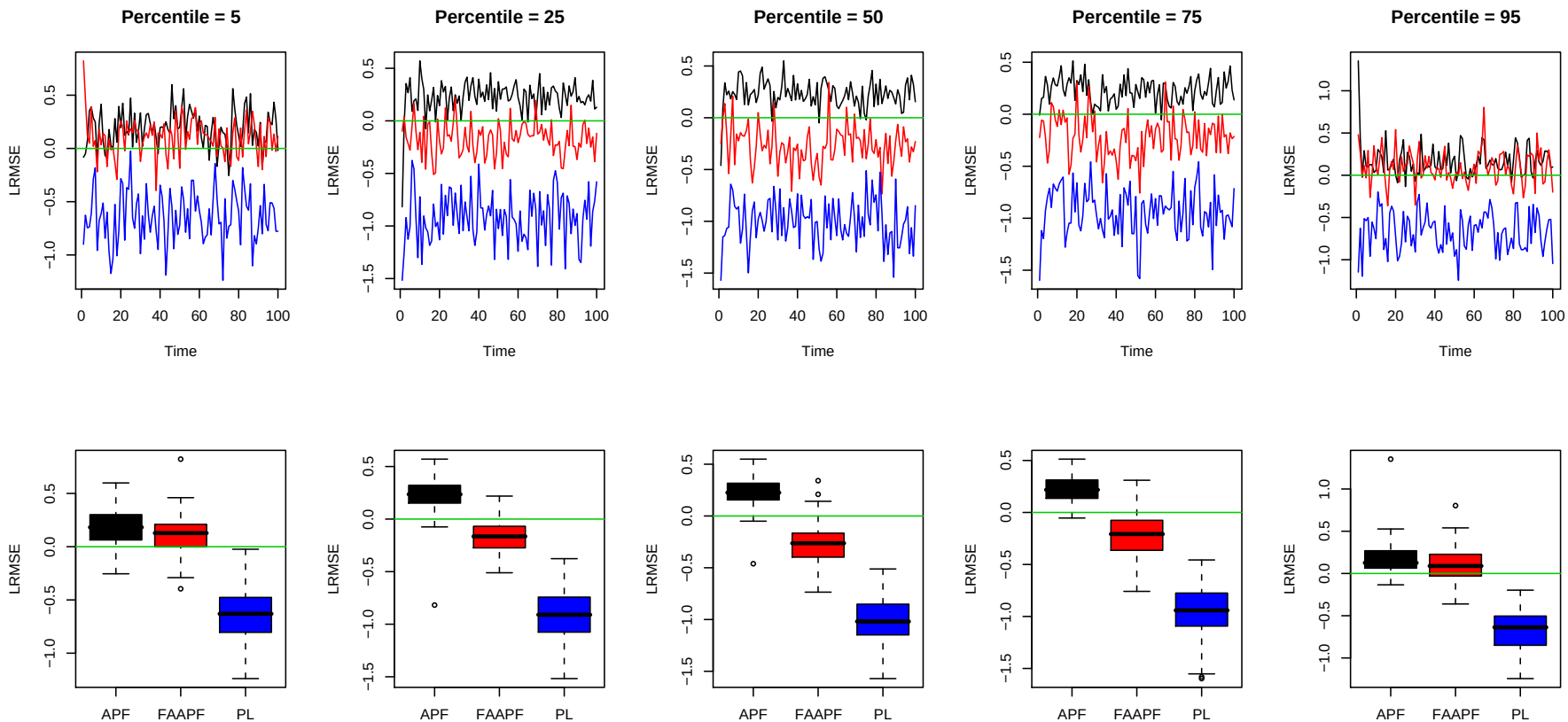
(APF,BBF), (FABBF,BBF) and (PL,BBF)



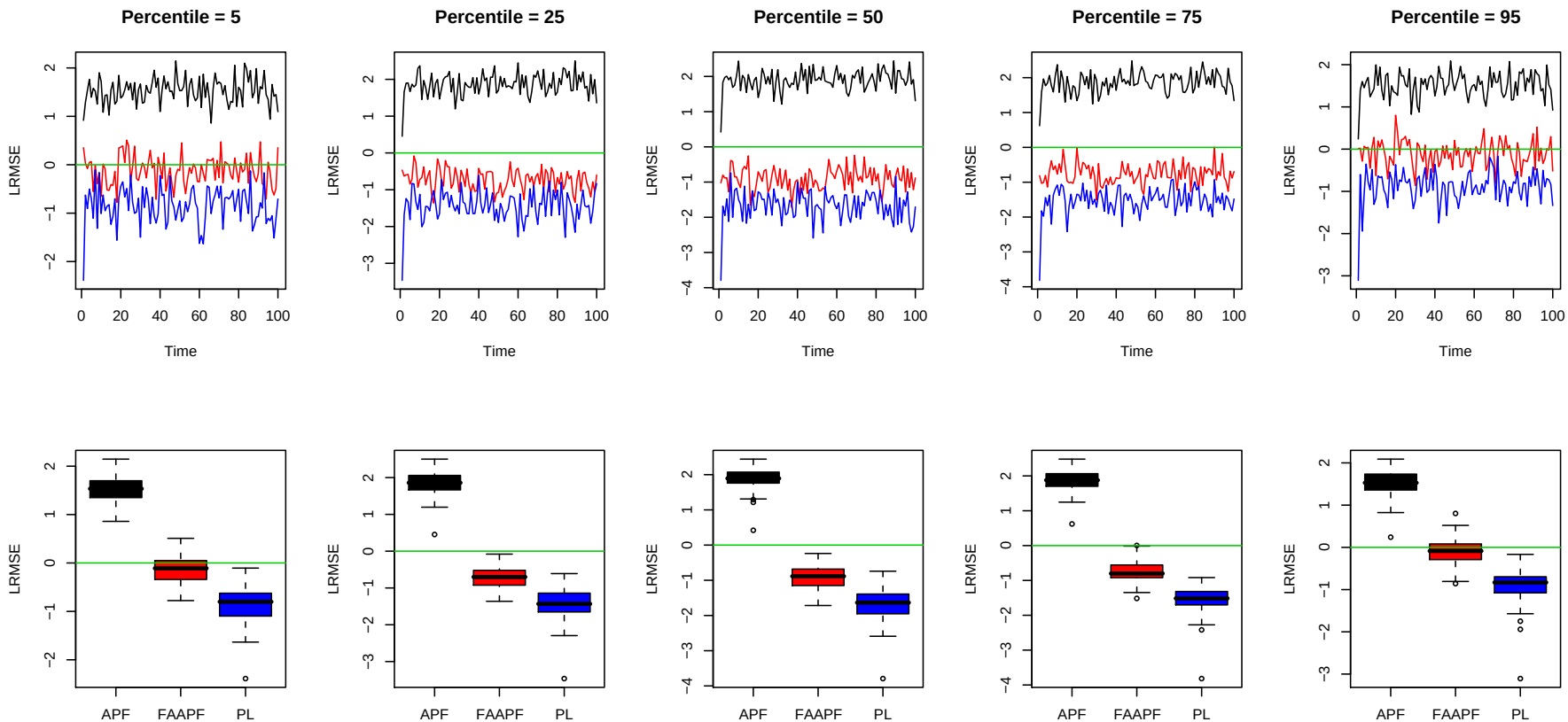
Example iv. $\sigma^2 = 0.13$ and $\tau/\sigma = 0.32$



Example iv. $\sigma^2 = 0.013$ and $\tau/\sigma = 1.00$



Example iv. $\sigma^2 = 0.0013$ and $\tau/\sigma = 3.16$



SMC smoothers

SMC smoothers are alternatives to MCMC in state-space models.

Godsill, Doucet and West (2004) "Monte Carlo smoothing for non-linear time series" introduced an $O(TN^2)$ algorithm that relies on

- Forward particles, and
- Backward re-weighting via evolution equation.

See also Briers, Doucet and Maskell's (2009) "Smoothing Algorithms for State-Space Models" for other $O(TN^2)$ smoothers.

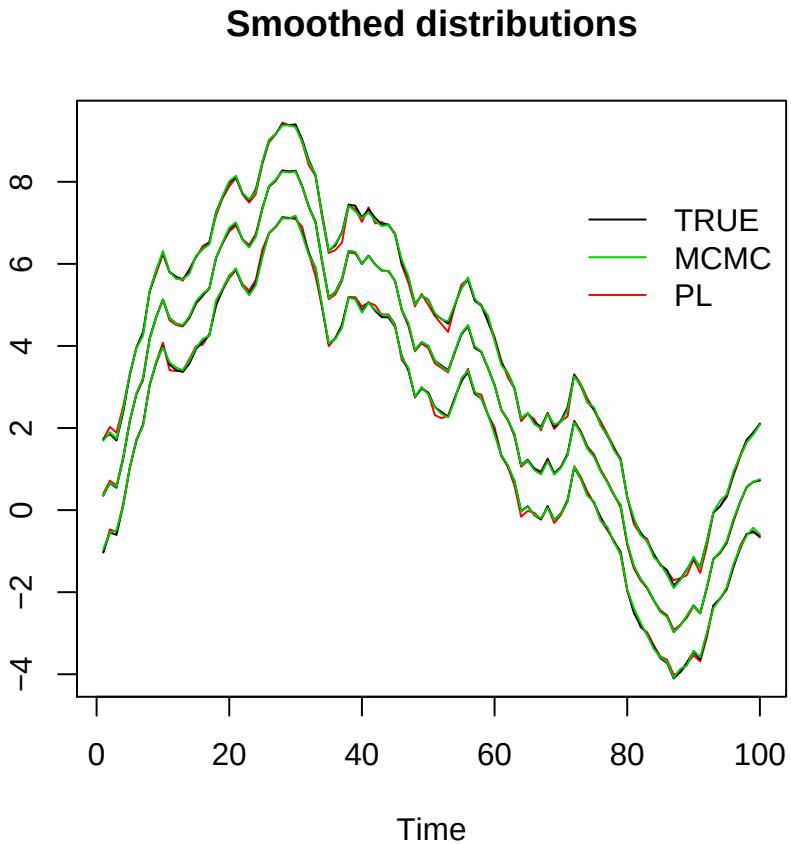
An $O(TN)$ smoothing algorithm is introduced by Fearnhead, Wyncoll and Tawn's (2008) "A sequential smoothing algorithm with linear computational cost".

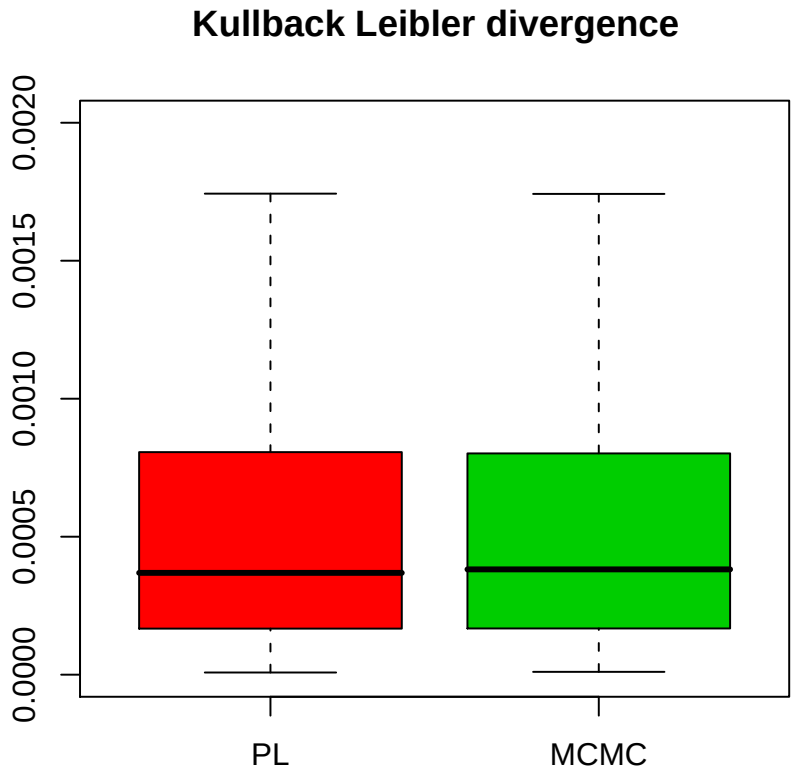
Example v. PL and MCMC (filtering and smoothing)

$n = 100$, $\sigma^2 = 1, \tau^2 = 0.5$ and $x_0 = 0$ and $x_0 \sim N(0, 100)$.

$N = 1000$ PL particles

$M = 1000$ MCMC draws, after discarding the first $M_0 = 1000$.





Example v. Computing time (in seconds)

$N = M_0 = M = 500$

n	PL	MCMC
100	9.3	4.7
200	18.8	9.1
500	47.7	23.4
1000	93.9	46.1

$n = 100$ and $N = M_0 = M$

N	PL	MCMC
500	9.3	4.7
1000	32.8	9.6
2000	127.7	21.7

MacBook 2.4GHz Intel Duo processor, 4GB 1067MHz memory.

Example vi. Sample-resample or PL?

Three time series of length $T = 1000$ were simulated from

$$\begin{aligned} y_t|x_t, \sigma^2 &\sim N(x_t, \sigma^2) \\ x_t|x_{t-1}, \tau^2 &\sim N(x_{t-1}, \tau^2) \end{aligned}$$

with $x_0 = 0$ and (σ^2, τ^2) in $\{(0.1, 0.01), (0.01, 0.01), (0.01, 0.1)\}$. Throughout σ^2 is kept fixed.

The independent prior distributions for x_0 and τ^2 are $x_0 \sim N(m_0, V_0)$ and $\tau^2 \sim IG(a, b)$, for $a = 10$, $b = (a+1)\tau_0^2$, $m_0 = 0$ and $V_0 = 1$, where τ_0^2 is the true value of τ^2 for a given study.

We also include BBF in the comparison, for completion.

In all filters τ^2 is sampled offline from $p(\tau^2|S_t)$ where S_t is the vector of conditional sufficient statistics.

Example vi. Mean absolute error

The three filters are rerun $R = 100$ times, all with the same seed within run, for each one of the three simulated data sets. Five different number of particles N were considered: 250, 500, 1000, 2000 and 5000.

Mean absolute errors (MAE) taken over the 100 replications are constructed by comparing percentiles of the true sequential distributions $p(x_t|y^t)$ and $p(\tau^2|y^t)$ to percentiles of the estimated sequential distributions $p_N(x_t|y^t)$ and $p_N(\tau^2|y^t)$.

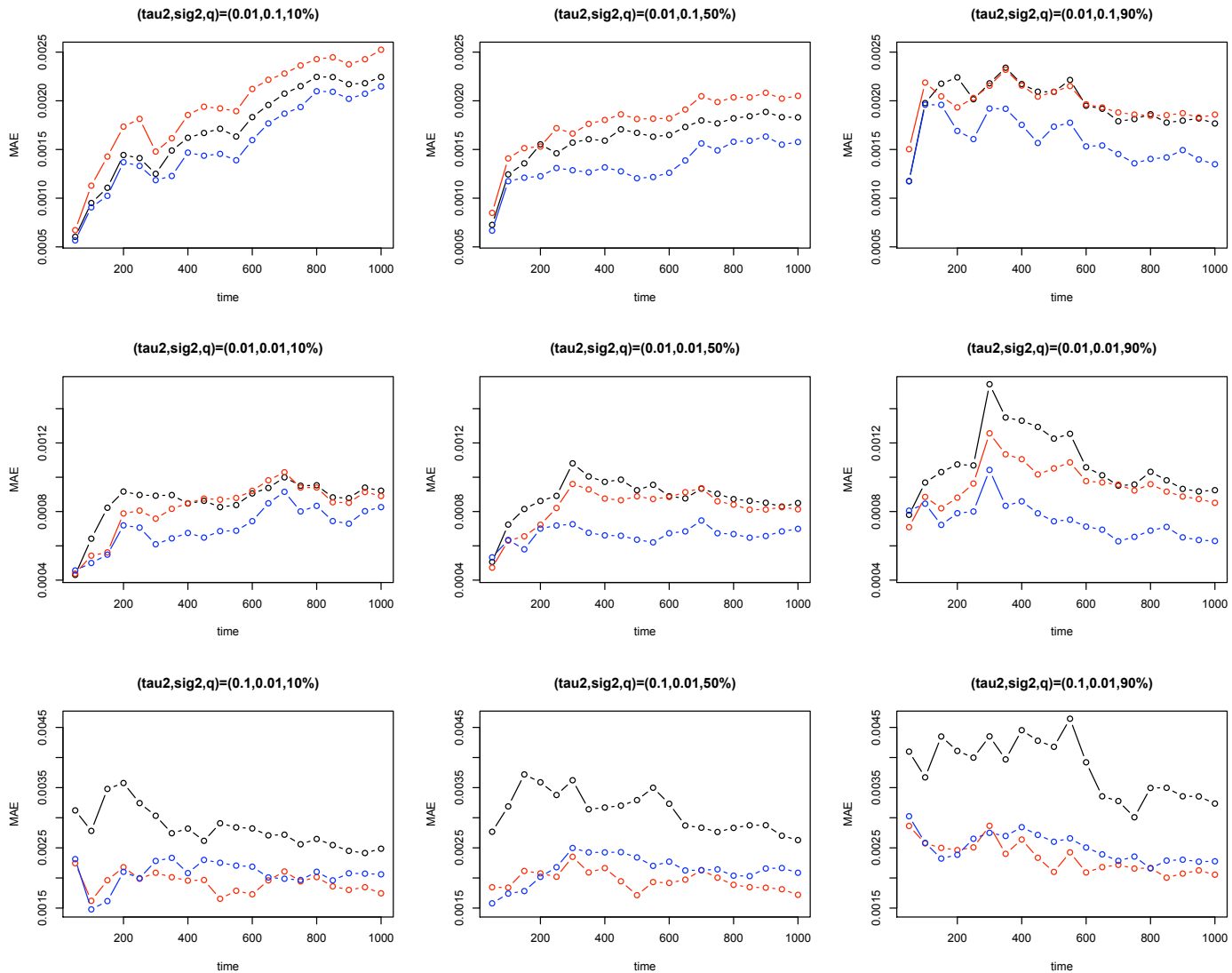
For $\alpha = 0.1, 0.5, 0.9$, true and estimated values of $q^x_{t,\alpha}$ and $q^{\tau^2}_{t,\alpha}$ were computed, for $Pr(x_t < q^x_{t,\alpha}|y^t) = Pr(\tau^2 < q^{\tau^2}_{t,\alpha}|y^t) = \alpha$.

For a in $\{x, \tau^2\}$ and α in $\{0.01, 0.50, 0.99\}$,

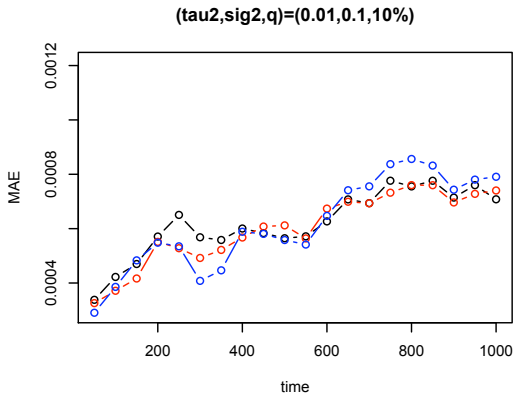
$$MAE^a_{t,\alpha} = \frac{1}{R}\sum_{r=1}^{R} |q^a_{t,\alpha} - \hat{q}^a_{t,\alpha,r}|$$

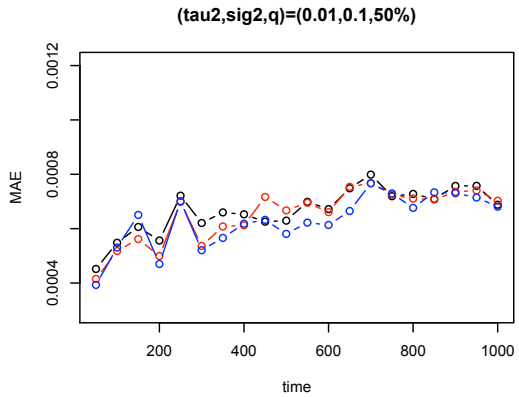
Example vi. $M = 500$ and learning τ^2.

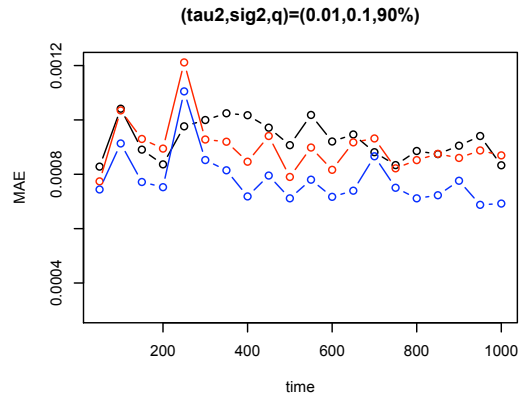
BBF,sample-resample,PL.

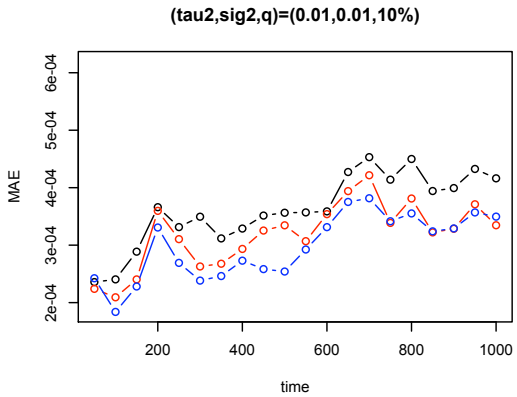


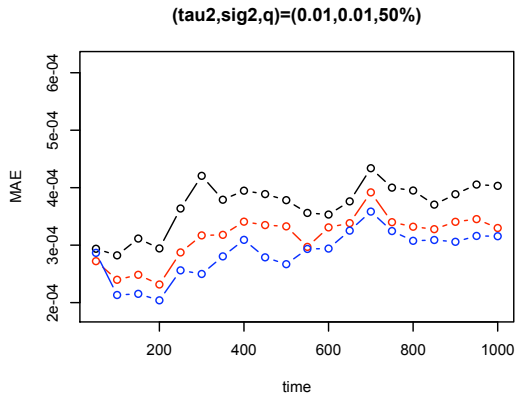
Example vi. $M = 5000$ and learning τ^2.

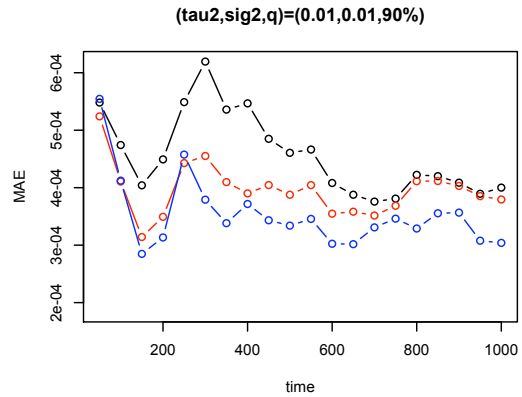


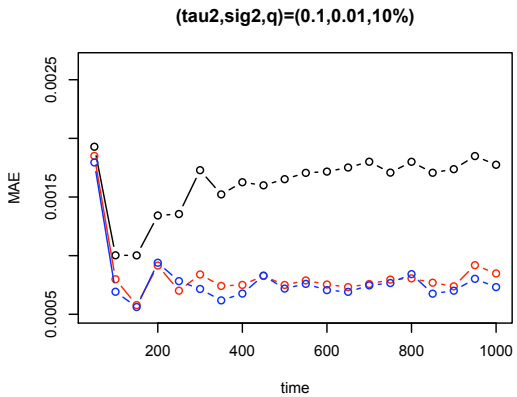


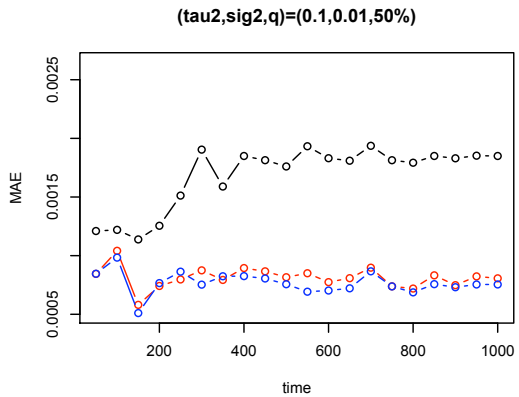


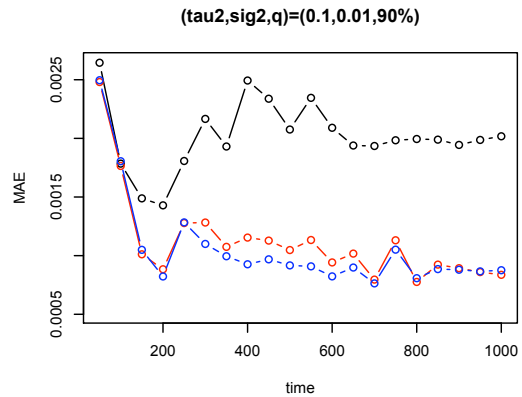




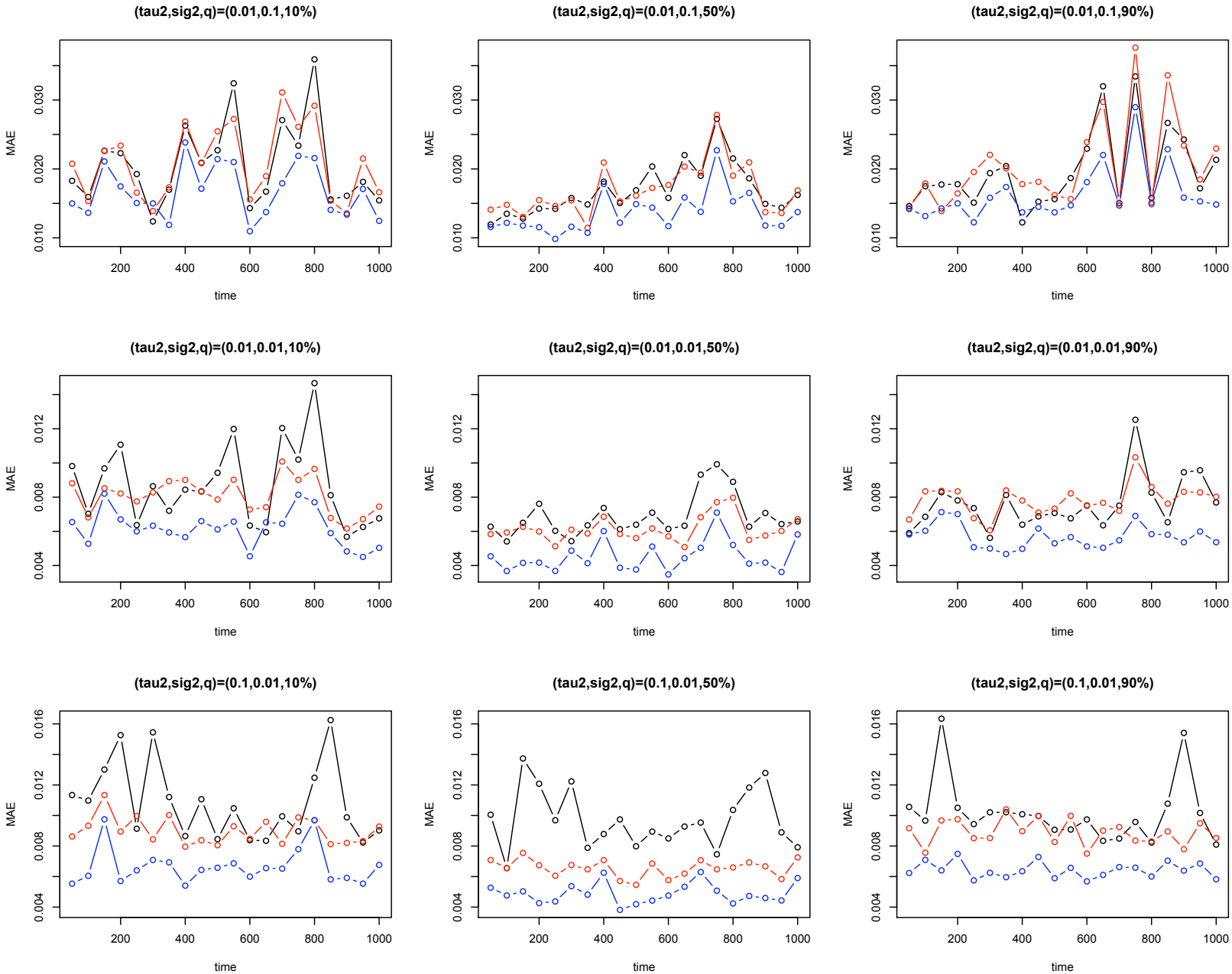




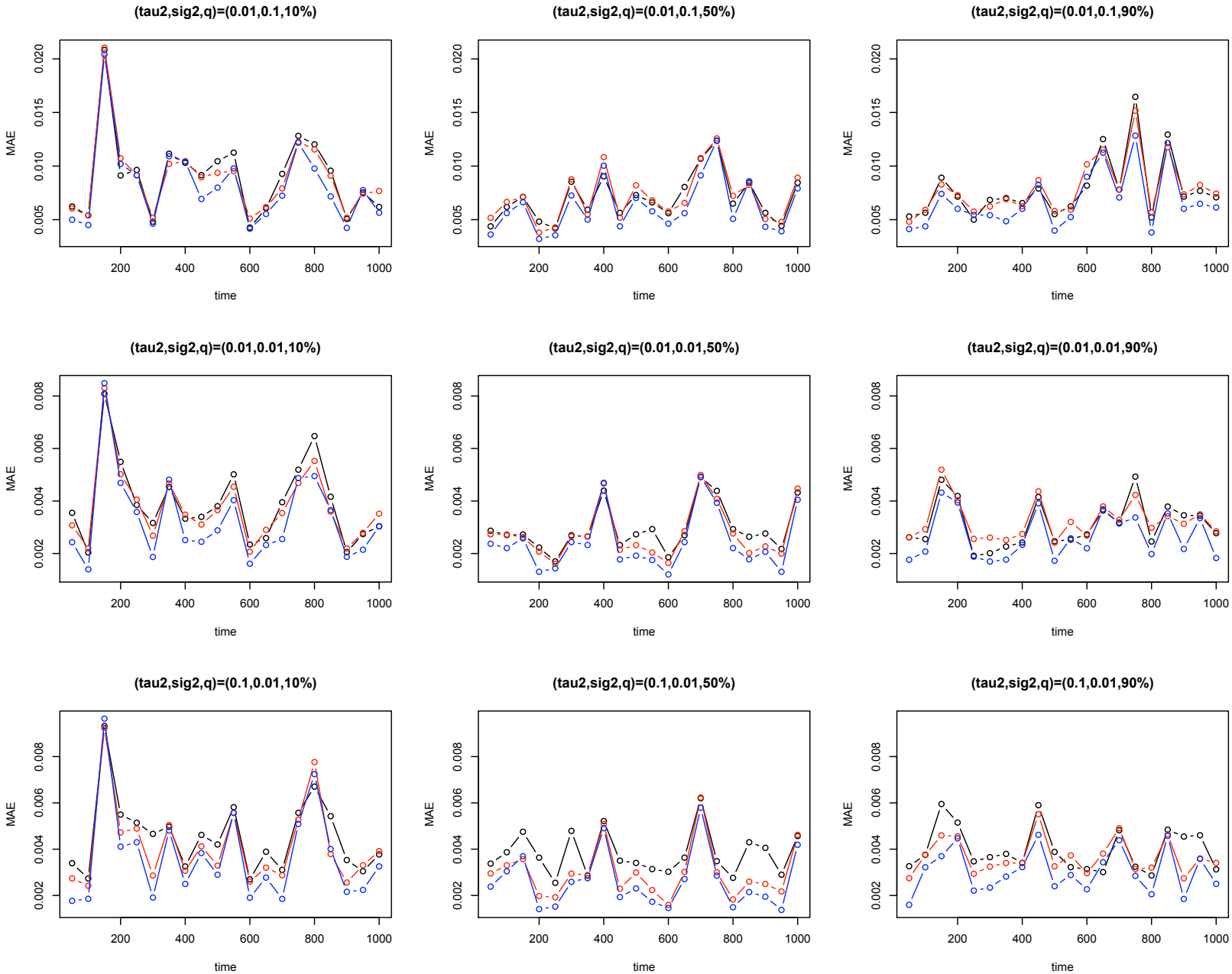




Example vi. $M = 500$ and learning x_t.



Example vi. $M = 5000$ and learning x_t.



Example vii. Computing sequential Bayes factors

A time series y_t is simulated from a *AR(1) plus noise* model:

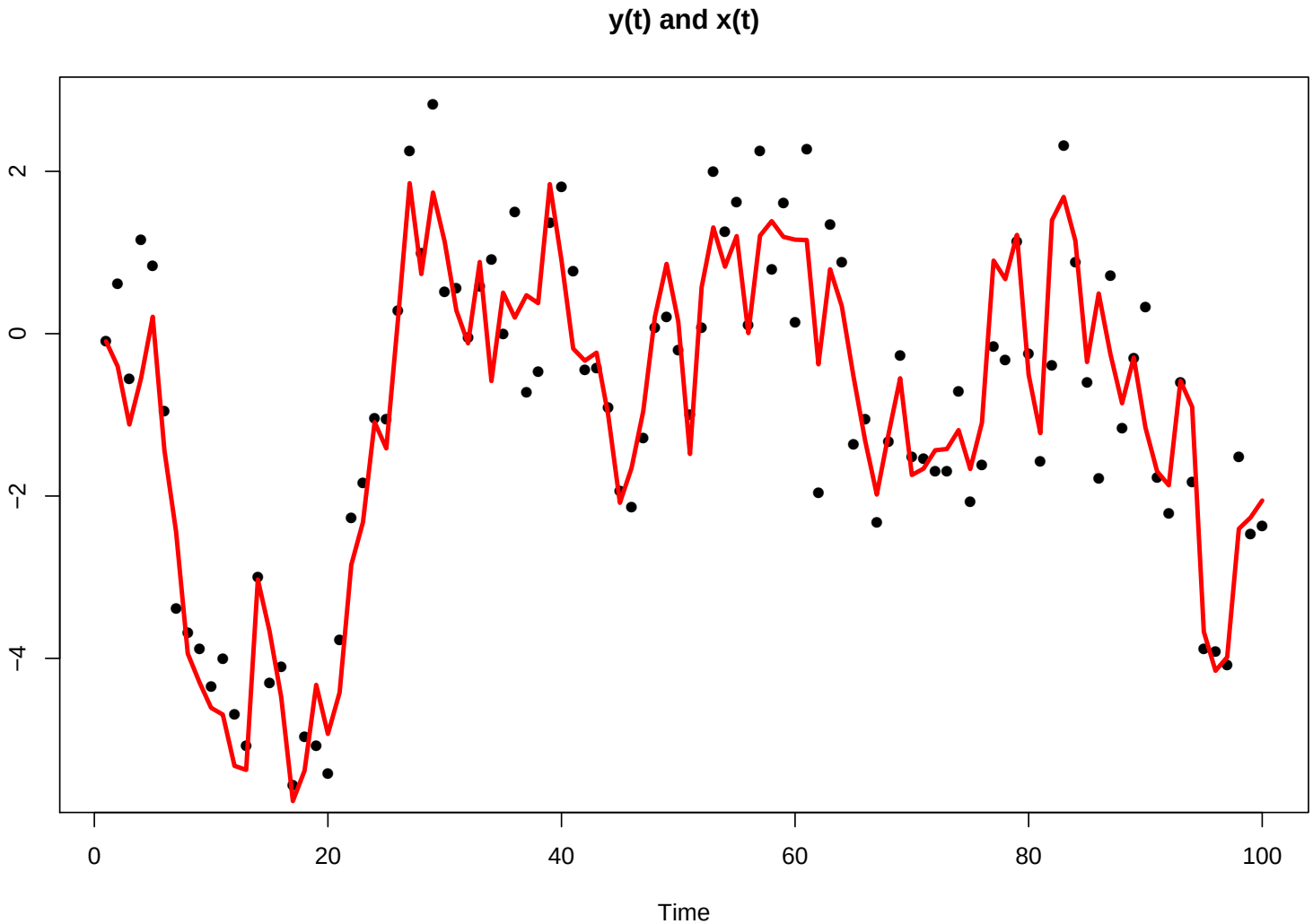
$$\begin{aligned}(y_{t+1}|x_{t+1},\theta) &\sim N(x_{t+1},\sigma^2)\\ (x_{t+1}|x_t,\theta) &\sim N(\beta x_t,\tau^2)\end{aligned}$$

for $t = 1,\ldots,T$.

We set $T = 100$, $x_0 = 0$, $\theta = (\beta,\sigma^2,\tau^2) = (0.9, 1.0, 0.5)$.

σ^2 and τ^2 are kept known and the independent prior distributions for β and x_0 are both $N(0,1)$.

Example vii. Simulated data



Example vii. PL pure filter versus PL

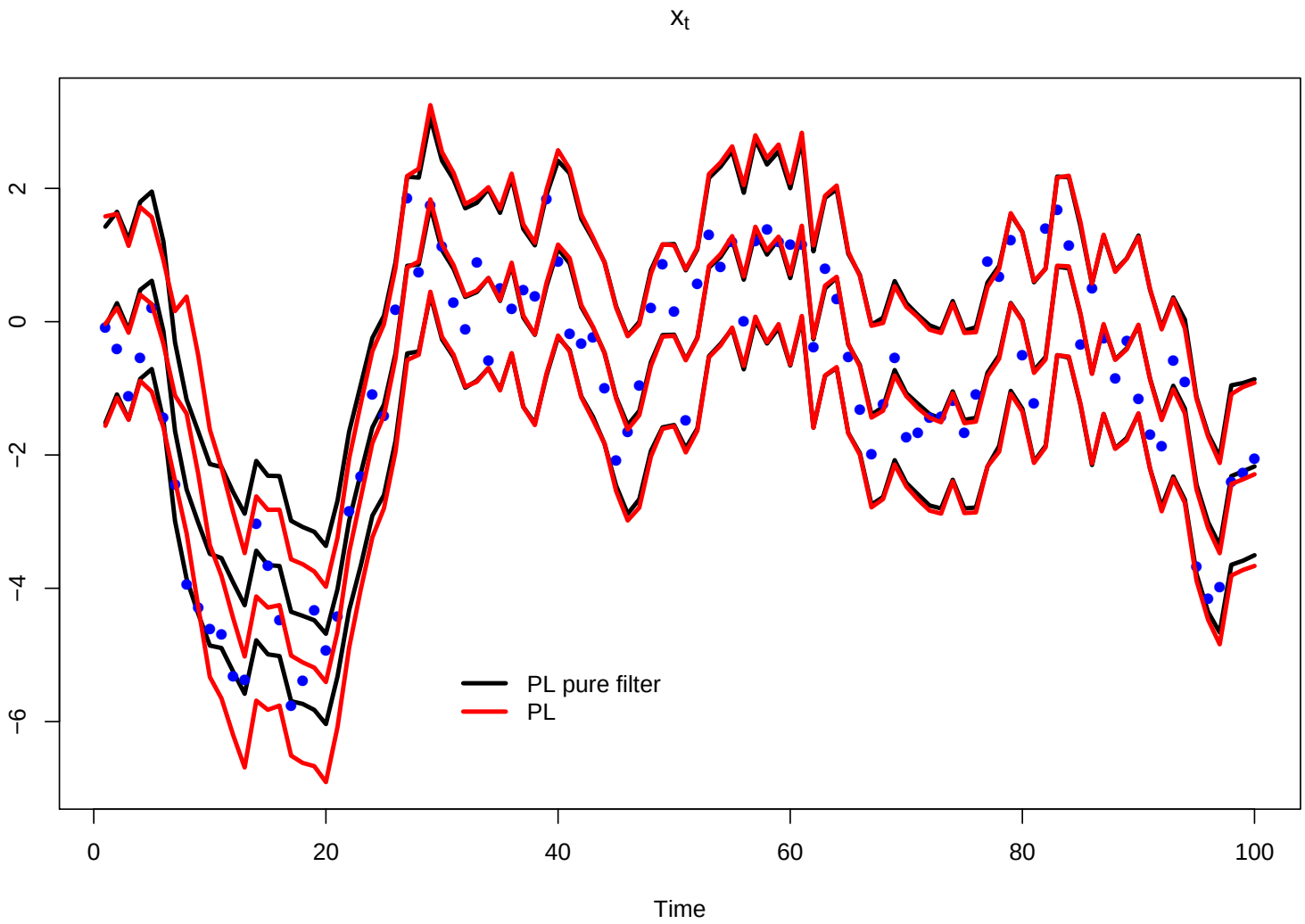
We run two filters:

- PL pure filter - our particle learning algorithm for learning x_t and keeping β fixed;
- PL - our particle learning algorithm for learning x_t and β sequentially.

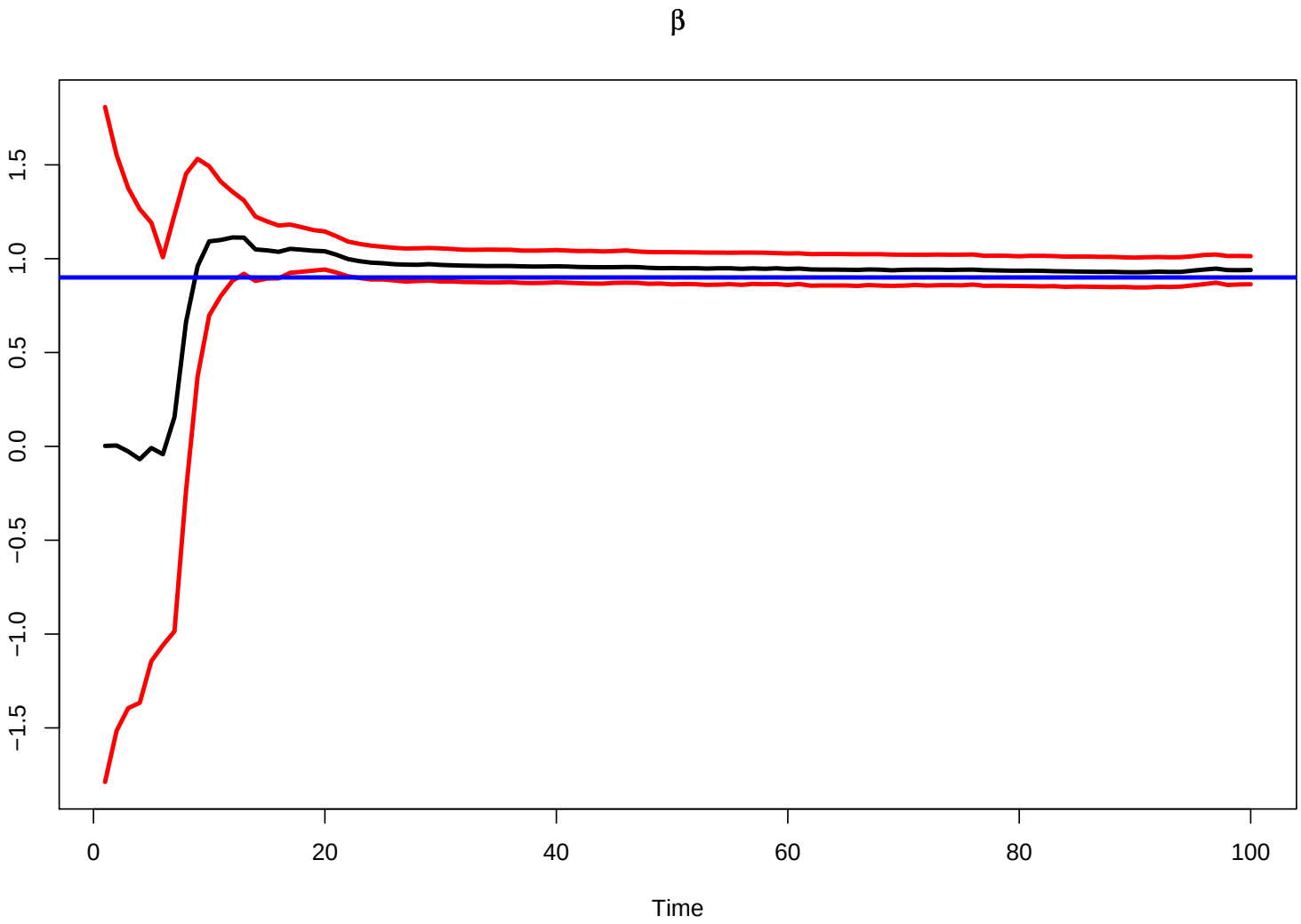
The filters are based on $N = 10,000$ particles.

Example vii. PL pure filter versus PL

β was fixed at the true value.

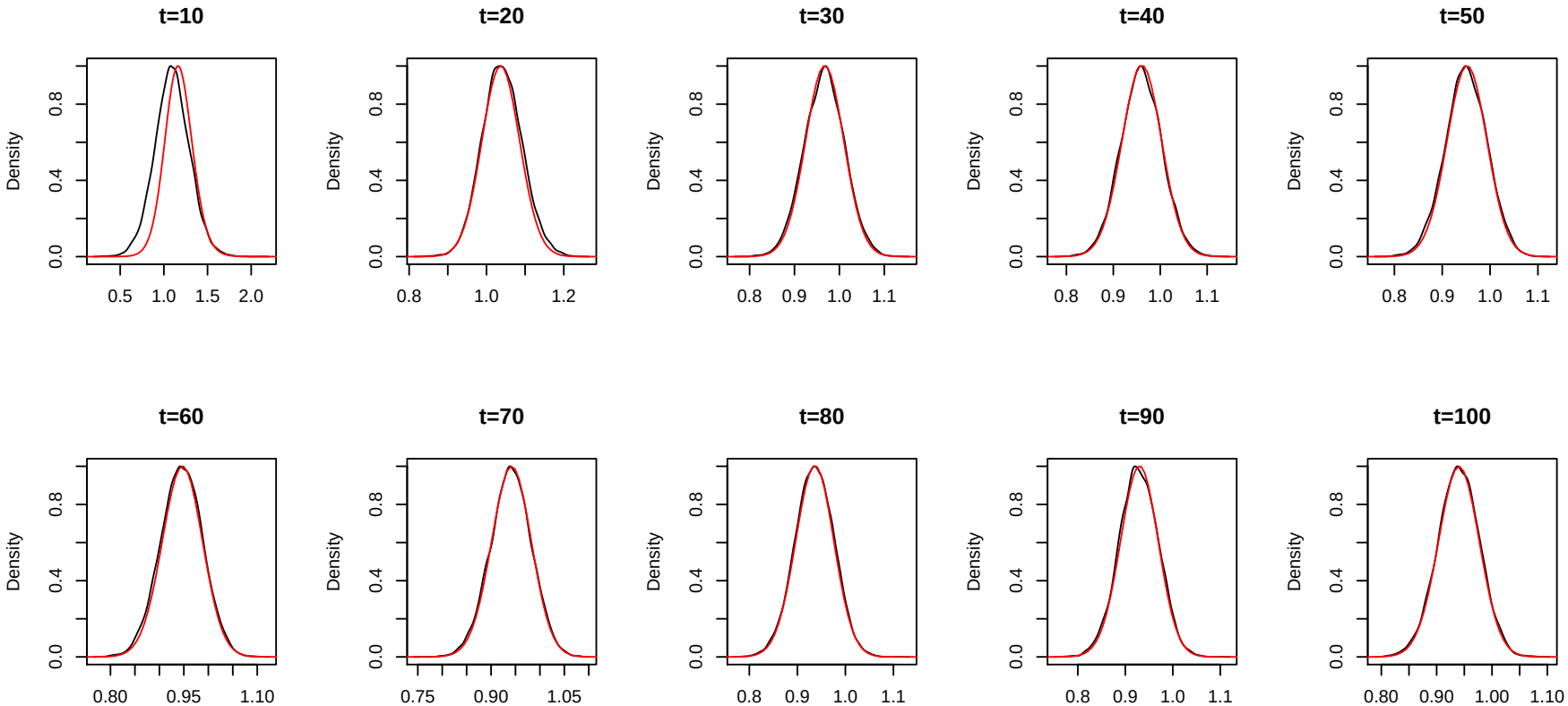


Example vii. PL - learning β

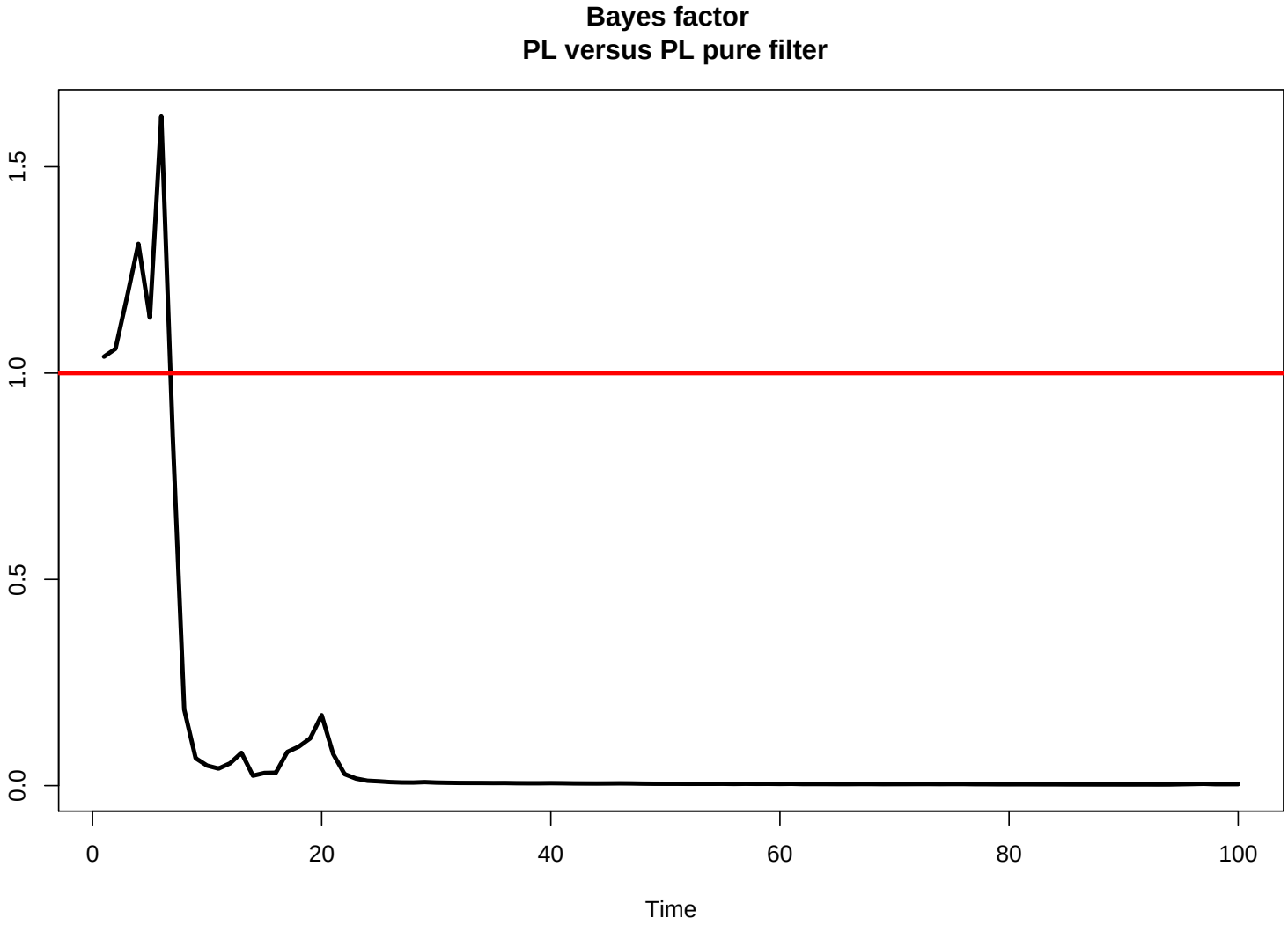


Example vii. PL - learning β

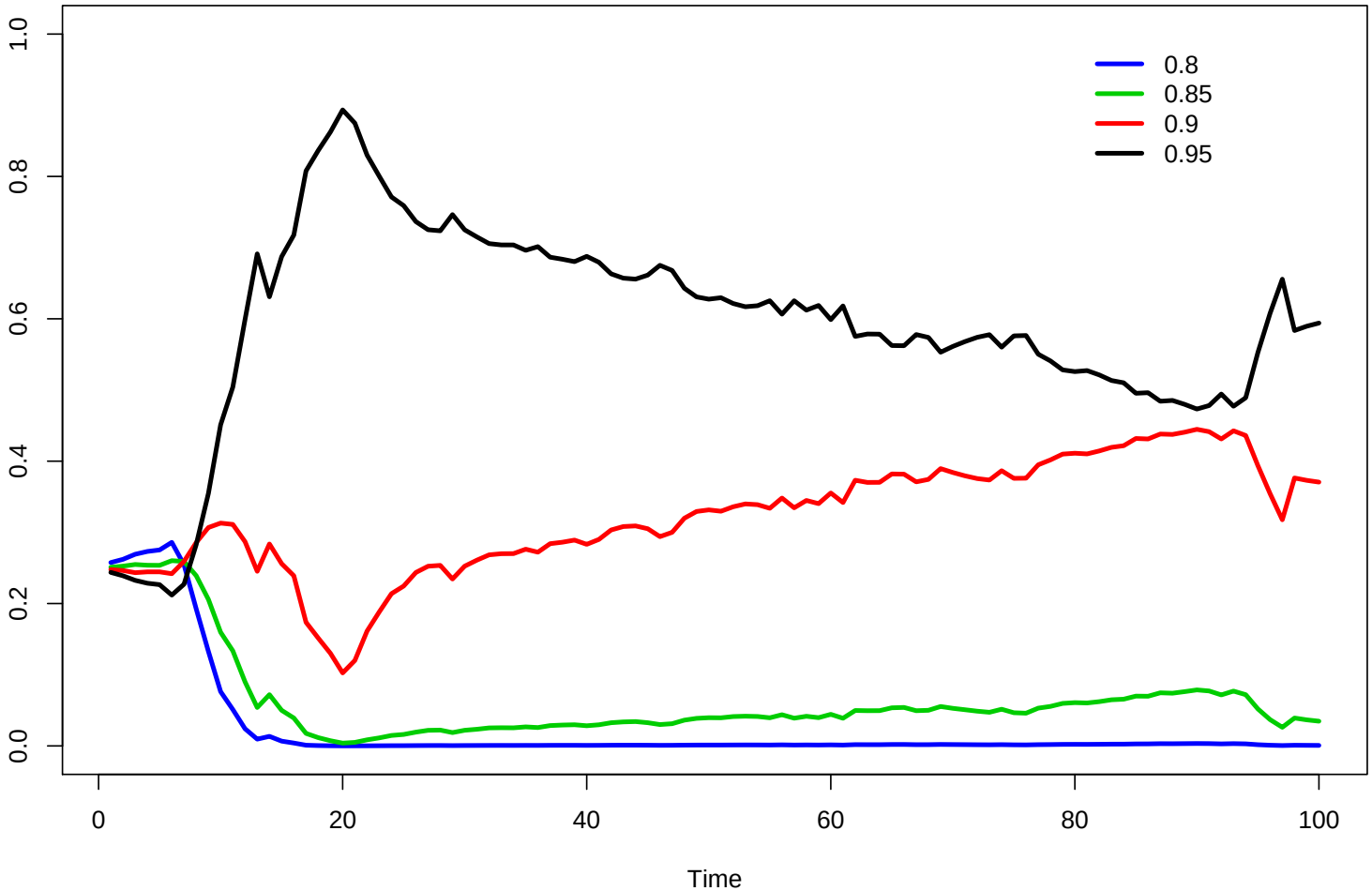
Comparing $p^N(\beta|y^t)$ with true $p(\beta|y^t)$.



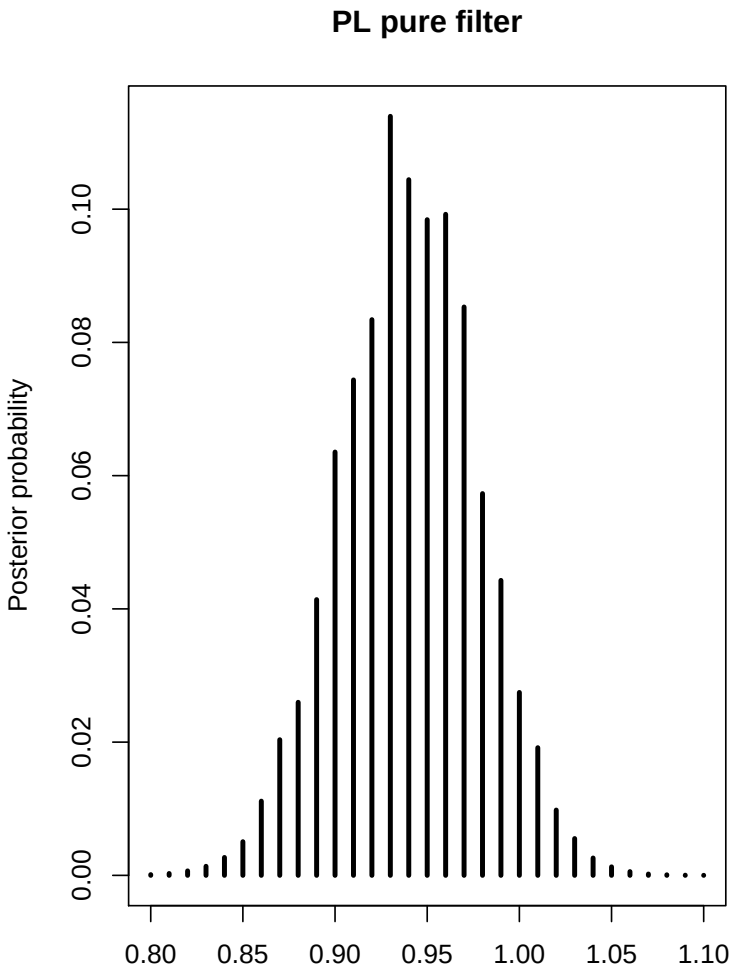
Example vii. Sequential Bayes factor

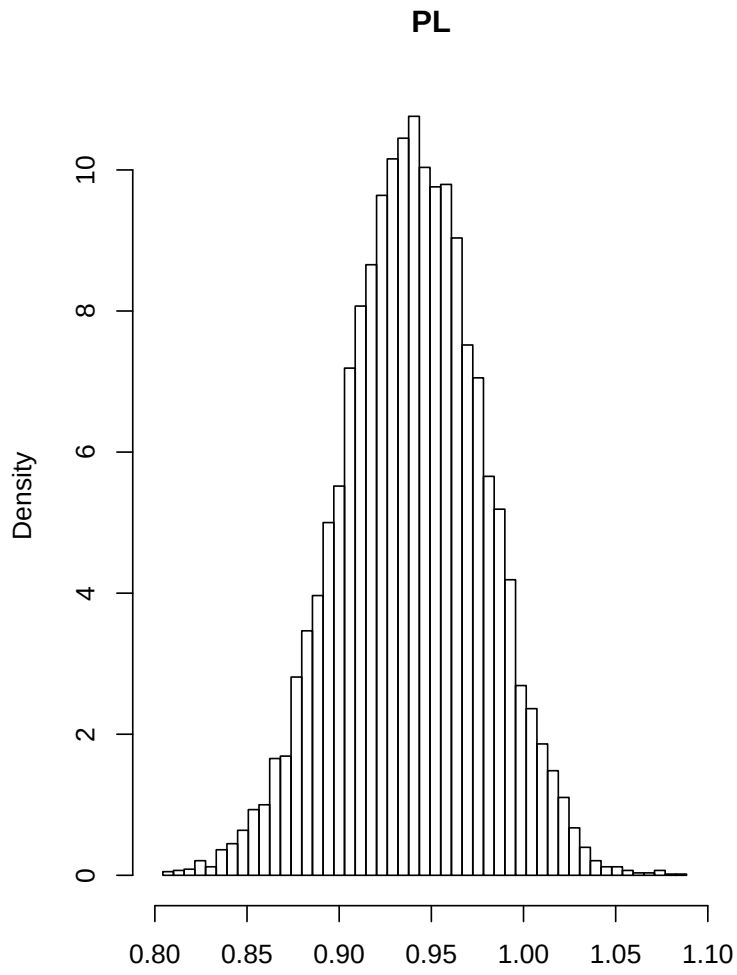


Example vii. Posterior model probabilities: 4 models



Example vii. Posterior model probabilities: 31 models





PL in Conditional Dynamic Linear Models (CDLM)

The model is

$$y_{t+1} = F_{\lambda_{t+1}} x_{t+1} + \epsilon_{t+1} \quad \text{where} \quad \epsilon_{t+1} \sim \mathcal{N}\left(0, V_{\lambda_{t+1}}\right)$$
$$x_{t+1} = G_{\lambda_{t+1}} x_t + \epsilon^x_{t+1} \quad \text{where} \quad \epsilon^x_{t+1} \sim \mathcal{N}\left(0, W_{\lambda_{t+1}}\right)$$

The error distribution

$$p\left(\epsilon_{t+1}\right) = \int \mathcal{N}\left(0, V_{\lambda_{t+1}}\right) p\left(\lambda_{t+1}\right) d\lambda_{t+1}$$

The augmented latent state is

$$\lambda_{t+1} \sim p(\lambda_{t+1}|\lambda_t)$$

PL extends Liu and Chen's (2000) "Mixture of Kalman Filters".

Algorithm

Step 1 (Re-sample): Generate an index $k(i) \sim \mathit{Multi}\left(w^{(i)}\right)$ where

$$w^{(i)} \propto p\left(y_{t+1} | (s_t^x, \theta)^{(i)}\right)$$

Step 2 (Propagate): States

$$\lambda_{t+1} \sim p\left(\lambda_{t+1} | \left(\lambda_t, \theta\right)^{k(i)}, y_{t+1}\right)$$

$$x_{t+1} \sim p(x_{t+1} | \left(x_t, \theta\right)^{k(i)}, \lambda_{t+1}, y_{t+1})$$

Step 3 (Propagate): Sufficient Statistics

$$s_{t+1}^x = \mathcal{K}\left(s_t^x, \theta, \lambda_{t+1}, y_{t+1}\right)$$

$$s_{t+1} = \mathcal{S}\left(s_t, x_{t+1}, \lambda_{t+1}, y_{t+1}\right)$$

Example A. Dynamic factor with switching loadings[5]

For $t = 1, \ldots, T$, the model is defined as follows:

- Observation equation

$$y_t | z_t, \theta \sim N(\gamma_t x_t, \sigma^2 I_2)$$

- State equations

$$\begin{aligned} x_t | x_{t-1}, \theta &\sim N(x_{t-1}, \sigma_x^2) \\ \lambda_t | \lambda_{t-1}, \theta &\sim Ber((1-p)^{1-\lambda_{t-1}} q^{\lambda_{t-1}}) \end{aligned}$$

where $z_t = (x_t, \lambda_t)'$.

Factor loadings: $\gamma_t = (1, \beta_{\lambda_t})'$.

Parameters: $\theta = (\beta_1, \beta_2, \sigma^2, \sigma_x^2, p, q)'$.

[5] Lopes and Carvalho (2007) and Lopes, Salazar and Gamerman (2008)

Example A. Conditionally conjugate prior

$$
\begin{aligned}
(\beta_i|\sigma^2) &\sim N\left(b_{i0}, \sigma^2 B_{i0}\right) \qquad \text{for } i = 1, 2, \\
\sigma^2 &\sim IG\left(\frac{\nu_{00}}{2}, \frac{d_{00}}{2}\right) \\
\sigma_x^2 &\sim IG\left(\frac{\nu_{10}}{2}, \frac{d_{10}}{2}\right) \\
p &\sim Beta(p_1, p_2) \\
q &\sim Beta(q_1, q_2) \\
x_0 &\sim N(m_0, C_0)
\end{aligned}
$$

Example A. Particle representation

At time t, particles

$$\left\{(x_t, \lambda_t, \theta, s_t^x, s_t)^{(i)}\right\}_{i=1}^{N}$$

approximating

$$p\left(x_t, \lambda_t, \theta, s_t^x, s_t | y^t\right)$$

where

- $s_t^x = \mathcal{S}(s_{t-1}^x, \theta)$ are state sufficient statistics
- $s_t = \mathcal{S}(s_{t-1}, x_t, \lambda_t)$ are fixed parameter sufficient statistics

Example A. Re-sampling $(x_t, \lambda_t, \theta, s_t^x, s_t)$

Let us redefine $\beta_i = (1, \beta_i)'$ whenever necessary.

Draw an index $k(i) \sim Multi(\omega^{(i)})$ with weights

$$\omega^{(i)} \propto p(y_{t+1}|(s_t^x, \lambda_t, \theta)^{k(i)})$$

with

$$p(y_{t+1}|m_t, C_t, \lambda_t, \theta) = \sum_{j=1}^{2} f_N\left(y_{t+1}; \beta_j m_t, V_j\right) Pr\left(\lambda_{t+1} = j|\lambda_t, \theta\right)$$

where $V_j = (C_t + \sigma_x^2)\beta_j\beta_j' + \sigma^2 I_2$, m_t and C_t are components of s_t^x and f_N denotes the normal density function.

Example A. Propagating states

Draw auxiliary state λ_{t+1}

$$\lambda_{t+1}^{(i)} \sim p(\lambda_{t+1}|(s_t^x, \lambda_t, \theta)^{k(i)}, y_{t+1})$$

where

$$Pr(\lambda_{t+1} = j|s_t^x, \lambda_t, \theta, y_{t+1}) \propto f_N\left(y_{t+1}; \beta_j m_t, V_j\right) p\left(\lambda_{t+1} = j|\lambda_t, \theta\right).$$

Draw state x_{t+1} conditionally on λ_{t+1}

$$x_{t+1}^{(i)} \sim p(x_{t+1}|\lambda_{t+1}^{(i)}, (s_t^x, \theta)^{k(i)}, y_{t+1})$$

by a simply Kalman filter update.

Example A. Updating sufficient statistics for states, s^x_{t+1}

The Kalman filter recursion yield

$$m_{t+1} = m_t + A_{t+1}(y_{t+1} - \beta_{\lambda_{t+1}} m_t)$$
$$C_{t+1} = C_t + \sigma_x^2 - A_{t+1} Q_{t+1}^{-1} A'_{t+1}$$

where

$$Q_{t+1} = (C_t + \sigma_x^2)\gamma_{t+1}\gamma'_{t+1} + \sigma^2 I_2$$
$$A_{t+1} = (C_t + \sigma_x^2)\gamma'_{t+1} Q_{t+1}^{-1}$$

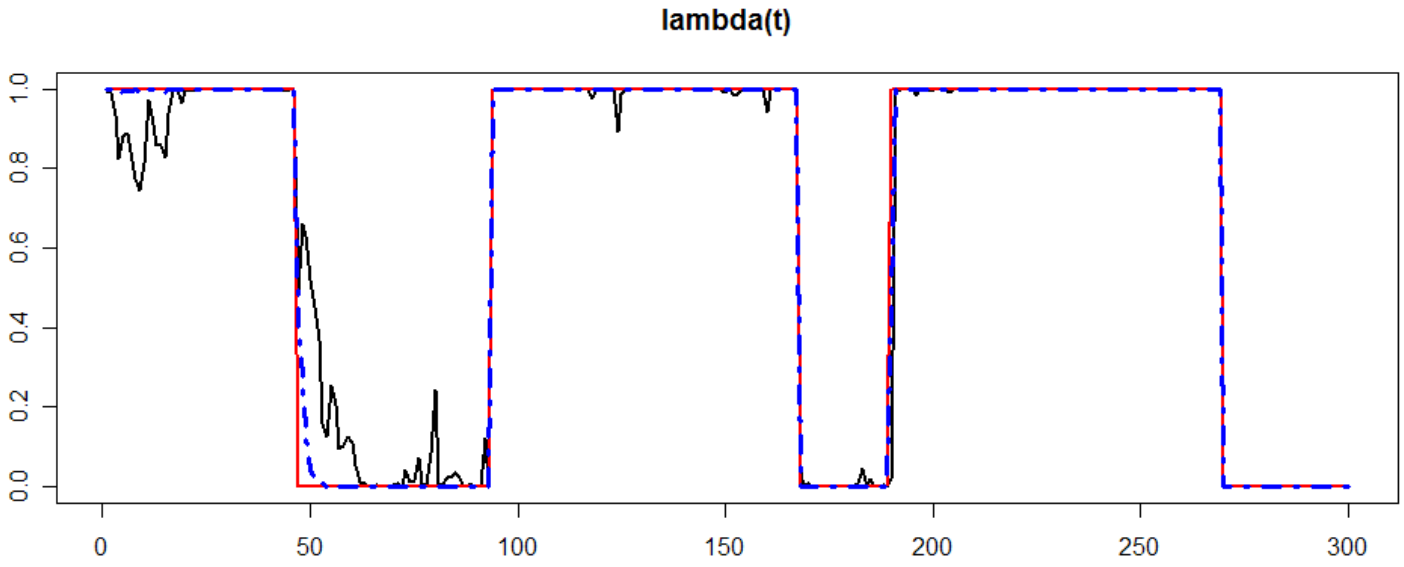
Example A. Updating suff. statistics for parameters, s_{t+1}

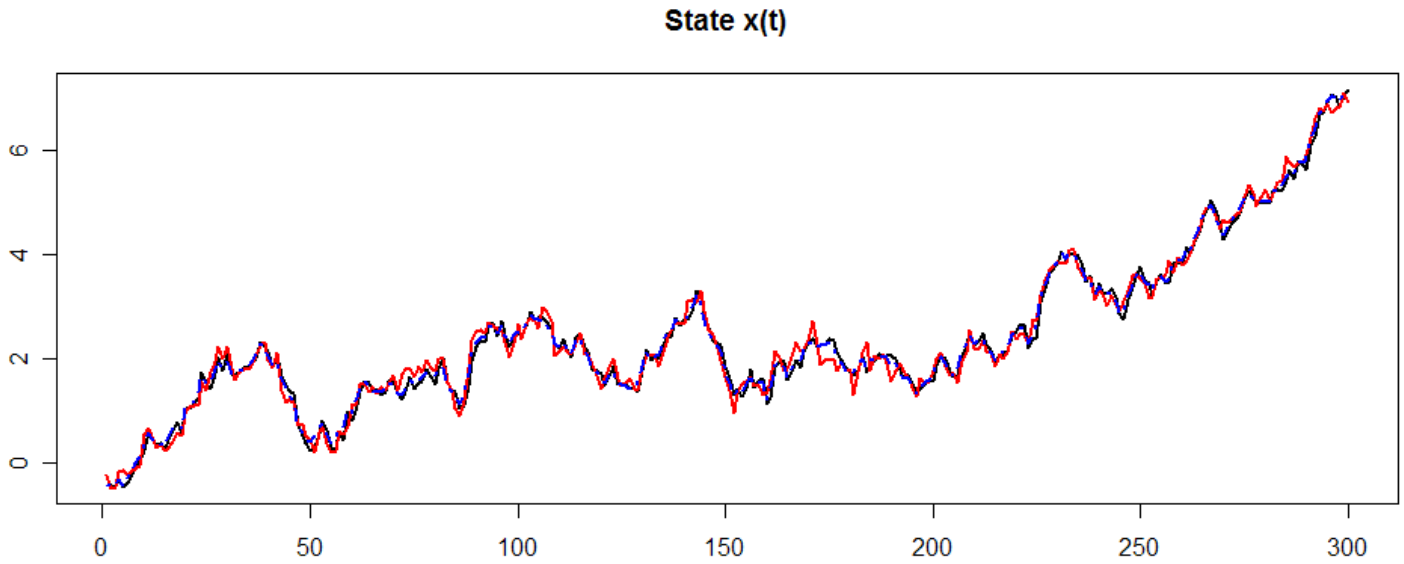
Recall that $s_{t+1} = \mathcal{S}(s_t, x_{t+1}, \lambda_{t+1})$. Then,

$$
\begin{aligned}
(\beta_i|\sigma^2, s_{t+1}) &\sim N\left(b_{i,t+1}, \sigma^2 B_{i,t+1}\right) \qquad \text{for } i = 1, 2, \\
(\sigma^2|s_{t+1}) &\sim IG\left(\frac{\nu_{0t}}{2}, \frac{d_{0,t+1}}{2}\right) \\
(\sigma_x^2|s_{t+1}) &\sim IG\left(\frac{\nu_{1t}}{2}, \frac{d_{1,t+1}}{2}\right) \\
(p|s_{t+1}) &\sim Beta(p_{1,t+1}, p_{2,t+1}) \\
(q|s_{t+1}) &\sim Beta(q_{1,t+1}, q_{2,t+1})
\end{aligned}
$$

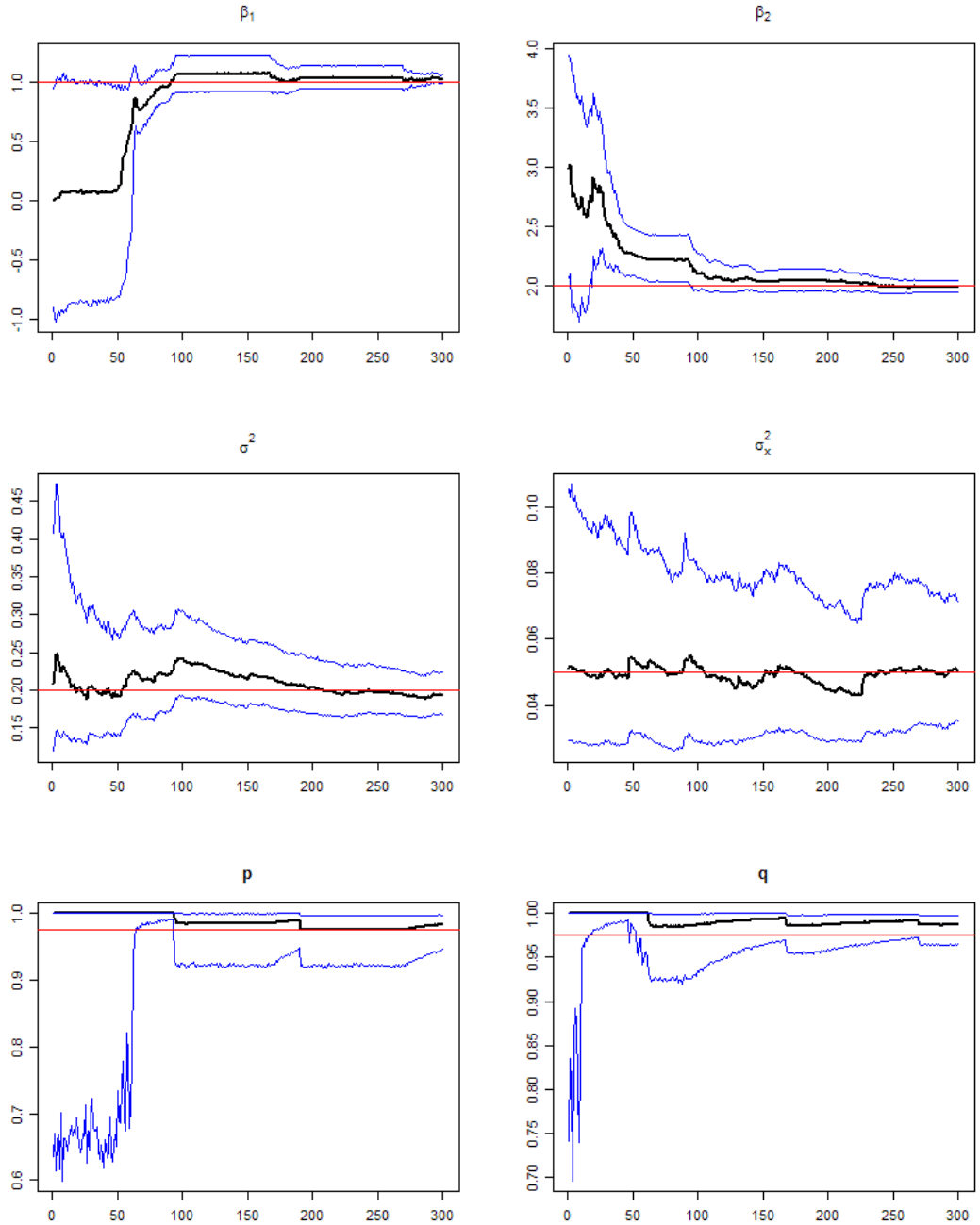
where $\mathbb{I}_{\lambda_{t+1}=i} = \mathbb{I}_i$, $\mathbb{I}_{\lambda_t=i,\lambda_{t+1}=j} = \mathbb{I}_{ij}$, $\nu_{it} = \nu_{i,t-1} + 1$,
$B_{i,t+1}^{-1} = B_{it}^{-1} + x_{t+1}^2$, $B_{i,t+1}^{-1} b_{i,t+1} = B_{it}^{-1} b_{it} + x_{t+1} y_{t+1,2} \mathbb{I}_i$,
$p_{i,t+1} = p_{it} + \mathbb{I}_{1i}$ (similarly for $q_{i,t+1}$) for $i = 1, 2$,
$d_{0,t+1} = d_{0,t} + (y_{t+1,1} - x_{t+1})^2 +$
$\sum_{j=1}^{2} \left[(y_{t+1,2} - b_{j,t+1} x_{t+1}) y_{t+1,2} + B_{j,t+1}^{-1} b_{j,t+1}\right] \mathbb{I}_j$, and
$d_{1,t+1} = d_{1,t} + (x_{t+1} - x_t)^2$.

Example A. Filtering and smoothing for states





Example A. Sequential parameter learning



PL in (state) non-linear normal dynamic models

The model now is

$$y_{t+1} = F_{\lambda_{t+1}} x_{t+1} + \epsilon_{t+1} \quad \text{where} \quad \epsilon_{t+1} \sim \mathcal{N}\left(0, V_{\lambda_{t+1}}\right)$$
$$x_{t+1} = G_{\lambda_{t+1}} Z\left(x_t\right) + \omega_{t+1} \quad \text{where} \quad \omega_{t+1} \sim \mathcal{N}\left(0, W_{\lambda_{t+1}}\right)$$

where ϵ_{t+1} and λ_{t+1} are modeled as before.

Algorithm:

Step 1 (Re-sample): Generate an index $k(i) \sim Multi\left(w^{(i)}\right)$ where

$$w^{(i)} \propto p\left(y_{t+1} | (x_t, \theta)^{(i)}\right)$$

Step 2 (Propagate):

$$\lambda_{t+1} \sim p\left(\lambda_{t+1} | \left(\lambda_t, \theta\right)^{k(i)}, y_{t+1}\right)$$
$$x_{t+1} \sim p\left(x_{t+1} | \left(x_t, , \theta\right)^{k(i)}, \lambda_{t+1}, y_{t+1}\right)$$
$$s_{t+1} = \mathcal{S}\left(s_t, x_{t+1}, \lambda_{t+1}, y_{t+1}\right)$$

Example B. Fat-tailed nonlinear model[6]

Let

$$y_{t+1} = x_{t+1} + \sigma\sqrt{\lambda_{t+1}}\epsilon_{t+1} \quad \text{where} \quad \lambda_{t+1} \sim \mathcal{IG}\left(\frac{\nu}{2}, \frac{\nu}{2}\right)$$

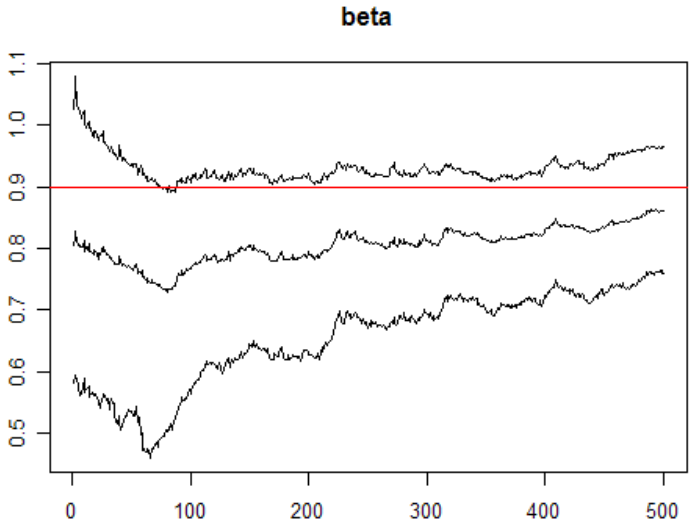
$$x_{t+1} = g(x_t)\beta + \sigma_x u_{t+1} \qquad \text{where} \quad g(x_t) = \frac{x_t}{1 + x_t^2}$$

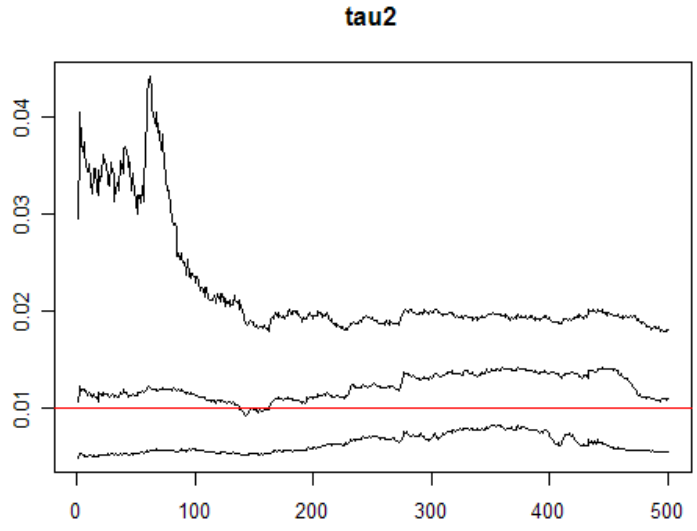
where ϵ_{t+1} and u_{t+1} are independent standard normals and ν is known.

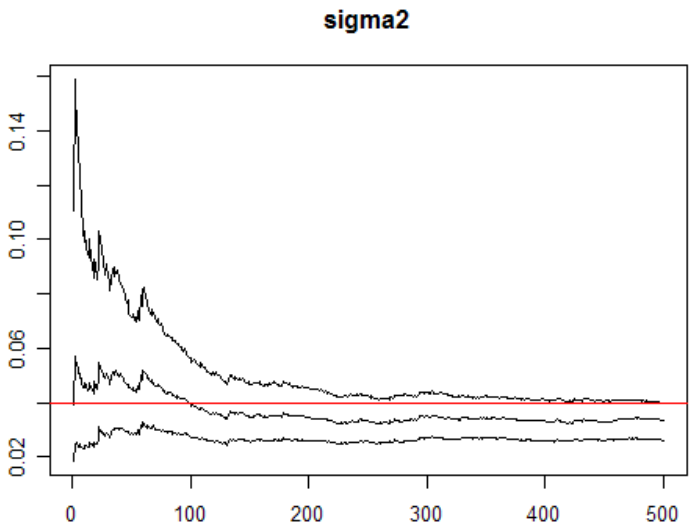
The observation error term is non-normal $\sqrt{\lambda_{t+1}}\epsilon_{t+1} \sim t_\nu$.

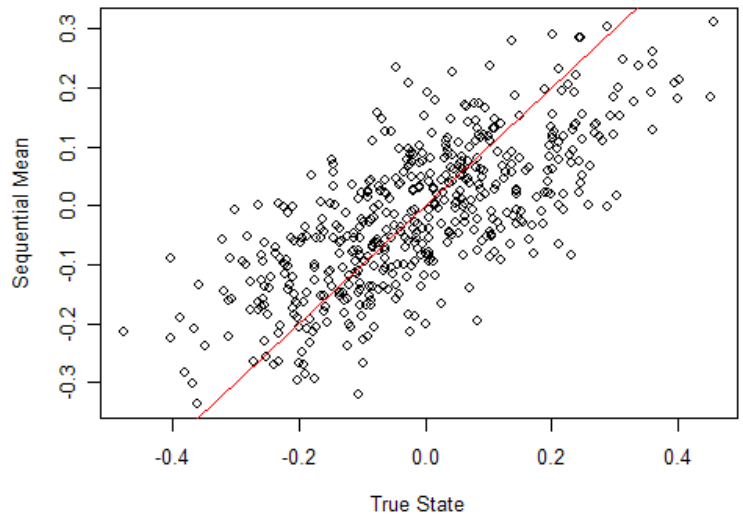
[6] From deJong et al (2007)

Example B. Sequential inference









Example C. Dynamic multinomial logit model

Let us study the multinomial logit model[7]

$$P\left(y_{t+1}=1|\beta_{t+1}\right)=\frac{e^{F_t\beta_t}}{1+e^{F_t\beta_t}} \quad \text{and} \quad \beta_{t+1} \quad =\phi\beta_t+\sigma_x\epsilon^{\beta}_{t+1}$$

where $\beta_0 \sim N(0,\sigma^2/(1-\rho^2))$. Scott's (2007) data augmentation structure leads to a mixture Kalman filter model

$$\begin{aligned} y_{t+1} &= \mathbb{I}(z_t \geq 0) \\ z_{t+1} &= Z_t\beta+\epsilon_{t+1} \ \text{ where } \ \epsilon_{t+1} \sim -ln\mathcal{E}(1) \end{aligned}$$

Here ϵ_t is an extreme value distribution of type 1 where $\mathcal{E}(1)$ is an exponential of mean one. The key is that it is easily to simulate $p(z_t|\beta,y_t)$ using

$$z_{t+1}=-\ln\left(\frac{\ln U_i}{1+e^{\beta_i\beta}}-\frac{\ln V_i}{e^{\beta_i\beta}}\mathcal{I}_{y_{t+1}=0}\right)$$

[7] Carvalho, Lopes and Polson (2008)

Example C. 10-component mixture of normals

Frunwirth-Schnatter and Schnatter (2007) uses a 10-component mixture of normals:

$$p(\epsilon_t) = e^{-\epsilon_t - e^{-\epsilon_t}} \approx \sum_{j=1}^{10} w_j \mathcal{N}(\mu_j, s_j^2)$$

Hence conditional on an indicator λ_t we can analyze

$$y_t = \mathbb{I}(z_t \geq 0) \quad \text{and} \quad z_t = \mu_{\lambda_t} + Z_t\beta + s_{\lambda_t}\epsilon_t$$

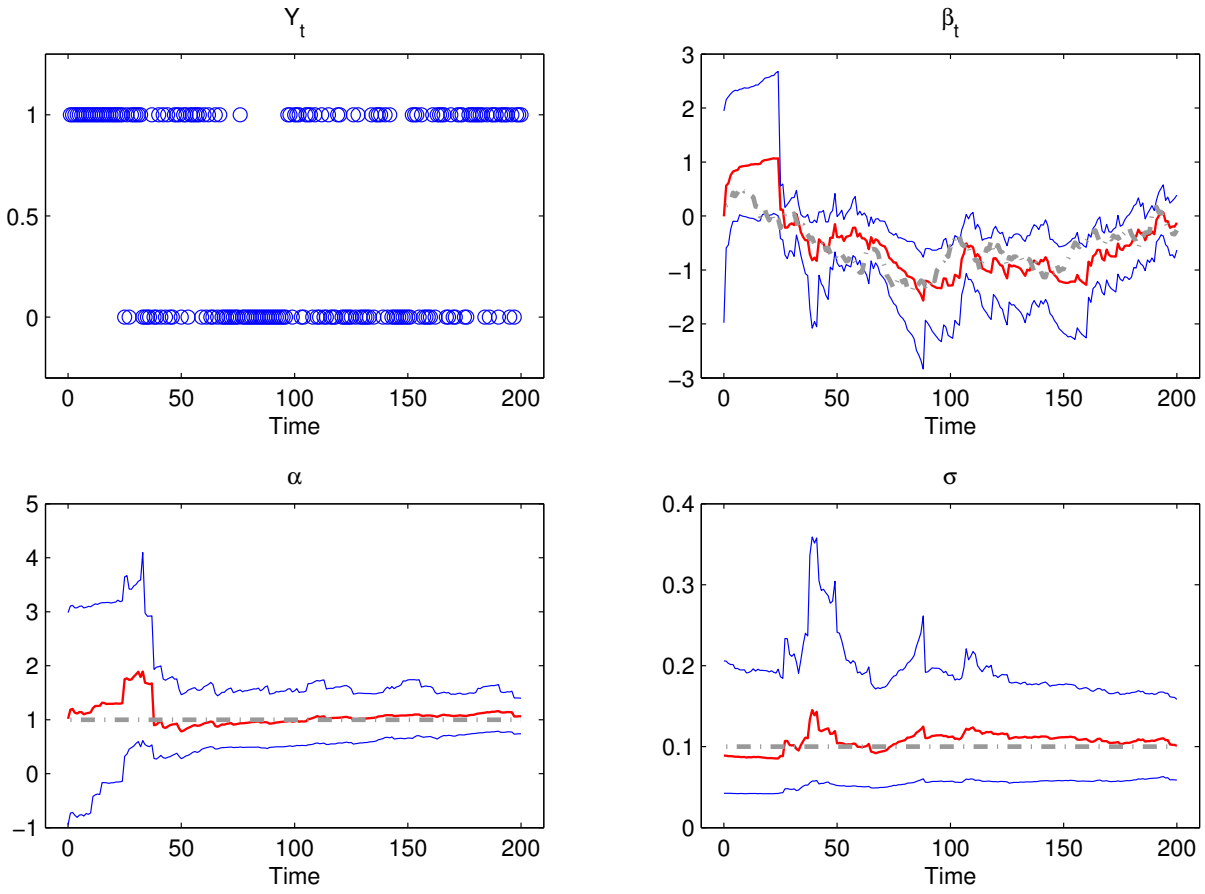
where $\epsilon_t \sim N(0,1)$ and $Pr(\lambda_t = j) = w_j$. Also,

$$\begin{aligned} s_{t+1}^{\beta} &= \mathcal{K}\left(s_t^{\beta}, z_{t+1}, \lambda_{t+1}, \theta, y_{t+1}\right) \\ p(y_{t+1}|s_t^{\beta}, \theta) &= \sum_{\lambda_{t+1}} p(y_{t+1}|s_t^{\beta}, \lambda_{t+1}, \theta) \end{aligned}$$

for re-sampling. Propagation now requires

$$\begin{aligned} \lambda_{t+1} &\sim p\left(\lambda_{t+1}|(s_t^{\beta}, \theta)^{k(i)}, y_{t+1}\right) \\ z_{t+1} &\sim p\left(z_{t+1}|(s_t^{\beta}, \theta)^{k(i)}, \lambda_{t+1}, y_{t+1}\right) \\ \beta_{t+1} &\sim p\left(z_{t+1}|(s_t^{\beta}, \theta)^{k(i)}, \lambda_{t+1}, z_{t+1}\right) \end{aligned}$$

Example C. Simulated exercise



PL based on 30,000 particles.

Example D. Sequential Bayesian Lasso

We develop a sequential version of Bayesian Lasso[8] for a simple problem of signal detection. The model takes the form

$$\begin{aligned}(y_t|\theta_t) &\sim N(\theta_t, 1)\\ p(\theta_t|\tau) &= (2\tau)^{-1}\exp\left(-|\theta_t|/\tau\right)\end{aligned}$$

for $t = 1, \ldots, n$ and $\tau^2 \sim IG(a_0, b_0)$.

Data augmentation: It is easy to see that

$$p(\theta_t|\tau) = \int p(\theta_t|\tau, \lambda_t)p(\lambda_t)d\lambda_t$$

where

$$\begin{aligned}\lambda_t &\sim Exp(2)\\ \theta_t|\tau, \lambda_t &\sim N(0, \tau^2\lambda_t)\end{aligned}$$

[8] Carlin and Polson (1991) and Hans (2008)

Example D. Data augmentation

The natural set of latent variables is given by the augmentation variable λ_{n+1} and conditional sufficient statistics leading to

$$Z_n = (\lambda_{n+1}, a_n, b_n)$$

The sequence of variables λ_{n+1} are i.i.d. and so can be propagated directly with $p(\lambda_{n+1})$.

The conditional sufficient statistics (a_{n+1}, b_{n+1}) are deterministically determined based on parameters $(\theta_{n+1}, \lambda_{n+1})$ and previous values (a_n, b_n).

Example D. PL algorithm

1. After n observations: $\left\{(Z_n, \tau)^{(i)}\right\}_{i=1}^{N}$.
2. Draw $\lambda_{n+1}^{(i)} \sim Exp(2)$.
3. Resample old particles with weights

$$w_{n+1}^{(i)} \propto p(y_{n+1}; 0, 1+\tau^{2(i)}\lambda_{n+1}^{(i)}).$$

4. Sample $\theta_{n+1}^{(i)} \sim N(m_n^{(i)}, C_n^{(i)})$, where $m_n^{(i)} = C_n^{(i)} y_{n+1}$ and $C_n^{-1} = 1 + \tilde{\tau}^{-2(i)}\tilde{\lambda}_{n+1}^{-1(i)}$.
5. Suff. stats: $a_{n+1}^{(i)} = \tilde{a}_n^{(i)} + 1/2$, $b_{n+1}^{(i)} = \tilde{b}_n^{(i)} + \theta_{n+1}^{2(i)}/(2\tilde{\lambda}_{n+1}^{(i)})$.
6. Sample (offline) $\tau^{2(i)} \sim IG(a_{n+1}, b_{n+1})$.
7. Let $Z_{n+1}^{(i)} = (\lambda_{n+1}^{(i)}, a_{n+1}^{(i)}, b_{n+1}^{(i)})$.
8. After $n+1$ observations: $\left\{(Z_{n+1}, \tau)^{(i)}\right\}_{i=1}^{N}$.

Example D. Sequential Bayes factor

As the Lasso is a model for sparsity we would expect the evidence for it to increase when we observe $y_t = 0$.

We can sequentially estimate $p(y_{n+1} \mid y^n, \text{lasso})$ via

$$p(y_{n+1} \mid y^n, \text{lasso}) = \frac{1}{N} \sum_{i=1}^{N} p(y_{n+1} \mid (\lambda_n, \tau)^{(i)})$$

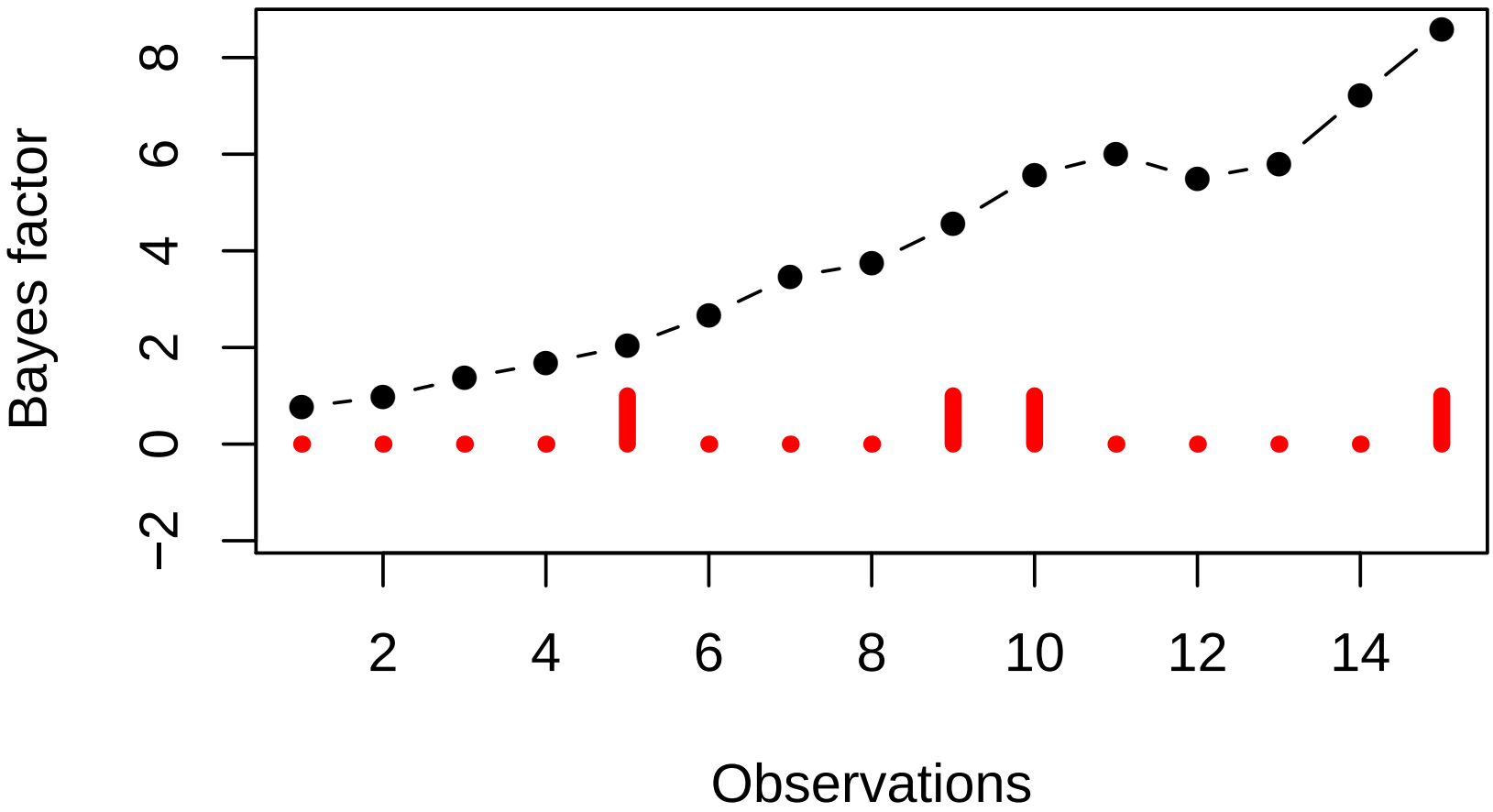
with predictive $p(y_{n+1} \mid \lambda_n, \tau) \sim N(0, \tau^2 \lambda_n + 1)$.

This leads to a sequential Bayes factor

$$BF_{n+1} = \frac{p(y^{n+1} \mid \text{lasso})}{p(y^{n+1} \mid \text{normal})}.$$

Example D. Simulated data

Data based on $\theta = (0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1)$ and priors $\tau^2 \sim IG(2, 1)$ for the double exponential case and $\tau^2 \sim IG(2, 3)$ for the normal case, reflecting the ratio of variances between those two distributions.



Final remarks

PL is a general framework for sequential Bayesian inference in dynamic and static models.

PL is able to deal with filtering and learning and reduce the accumulation of error.

The loose definition of sufficient statistics and the flexibility to freely augment x_t makes PL a competitive alternative to MCMC in highly structured models.

A powerful by-product of PL (and SMC in general) over MCMC schemes, is its ability to sequentially produce model comparison, assessment indicators.