Sequential Monte Carlo Methods for Conditional Linear Gaussian Models

Review of Kalman Filter For Linear Gaussian Models
Conditional Linear Gaussian Models
SMC for Time Series Models
Partially Observed Linear Gaussian Models

Prof. Nicholas Zabaras
University of Notre Dame
Notre Dame, IN, USA

Email: nzabaras@gmail.com
URL: https://www.zabaras.com/

October 25, 2017

Contents

References

- C.P. Robert & G. Casella, Monte Carlo Statistical Methods, Chapter 11
- J.S. Liu, Monte Carlo Strategies in Scientific Computing, Chapter 3, Springer-Verlag, New York.
- A. Doucet, N. De Freitas & N. Gordon (eds), Sequential Monte Carlo in Practice, Springer-Verlag: 2001
- A. Doucet, N. De Freitas, N.J. Gordon, An introduction to Sequential Monte Carlo, in SMC in Practice, 2001
- D. Wilkison, Stochastic Modelling for Systems Biology, Second Edition, 2006
- E. Ionides, Inference for Nonlinear Dynamical Systems, PNAS, 2006
- J.S. Liu and R. Chen, Sequential Monte Carlo methods for dynamic systems, JASA, 1998
- A. Doucet, Sequential Monte Carlo Methods, Short Course at SAMSI
- A. Doucet, Sequential Monte Carlo Methods & Particle Filters Resources
- Pierre Del Moral, Feynman-Kac models and interacting particle systems (SMC resources)
- A. Doucet, Sequential Monte Carlo Methods, Video Lectures, 2007
- N. de Freitas and A. Doucet, Sequential MC Methods, N. de Freitas and A. Doucet, Video Lectures, 2010

References

- M.K. Pitt and N. Shephard, Filtering via Simulation: Auxiliary Particle Filter, JASA, 1999
- A. Doucet, S.J. Godsill and C. Andrieu, On Sequential Monte Carlo sampling methods for Bayesian filtering, Stat. Comp., 2000
- J. Carpenter, P. Clifford and P. Fearnhead, An Improved Particle Filter for Non-linear Problems, IEE 1999.
- A. Kong, J.S. Liu & W.H. Wong, Sequential Imputations and Bayesian Missing Data Problems, JASA, 1994
- O. Cappe, E. Moulines & T. Ryden, Inference in Hidden Markov Models, Springer-Verlag, 2005
- W Gilks and C. Berzuini, Following a moving target: MC inference for dynamic Bayesian Models, JRSS B, 2001
- G. Poyadjis, A. Doucet and S.S. Singh, Maximum Likelihood Parameter Estimation using Particle Methods, Joint Statistical Meeting, 2005
- N Gordon, D J Salmond, AFM Smith, Novel Approach to nonlinear non Gaussian Bayesian state estimation, IEE, 1993
- Particle Filters, S. Godsill, 2009 (Video Lectures)
- R. Chen and J.S. Liu, Predictive Updating Methods with Application to Bayesian Classification, JRSS B, 1996

References

- C. Andrieu and A. Doucet, Particle Filtering for Partially Observed Gaussian State-Space Models, JRSS B, 2002
- R Chen and J Liu, Mixture Kalman Filters, JRSSB, 2000
- A Doucet, S J Godsill, C Andrieu, On SMC sampling methods for Bayesian Filtering, Stat. Comp. 2000
- N. Kantas, A.D., S.S. Singh and J.M. Maciejowski, An overview of sequential Monte Carlo methods for parameter estimation in general state-space models, in Proceedings IFAC System Identification (SySid) Meeting, 2009
- C. Andrieu, A.Doucet & R. Holenstein, Particle Markov chain Monte Carlo methods, JRSS B, 2010
- C. Andrieu, N. De Freitas and A. Doucet, Sequential MCMC for Bayesian Model Selection, Proc. IEEE Workshop HOS, 1999
- P. Fearnhead, MCMC, sufficient statistics and particle filters, JCGS, 2002
- G. Storvik, Particle filters for state-space models with the presence of unknown static parameters, IEEE Trans. Signal Processing, 2002

References

- C. Andrieu, A. Doucet and V.B. Tadic, Online EM for parameter estimation in nonlinear-non Gaussian state-space models, Proc. IEEE CDC, 2005
- G. Poyadjis, A. Doucet and S.S. Singh, Particle Approximations of the Score and Observed Information Matrix in State-Space Models with Application to Parameter Estimation, *Biometrika*, 2011
- C. Caron, R. Gottardo and A. Doucet, On-line Changepoint Detection and Parameter Estimation for Genome Wide Transcript Analysis, Technical report 2008
- R. Martinez-Cantin, J. Castellanos and N. de Freitas. Analysis of Particle Methods for Simultaneous Robot Localization and Mapping and a New Algorithm: Marginal-SLAM. International Conference on Robotics and Automation
- C. Andrieu, A.D. & R. Holenstein, Particle Markov chain Monte Carlo methods (with discussion), JRSS B, 2010
- A Doucet, Sequential Monte Carlo Methods and Particle Filters, List of Papers, Codes, and Viedo lectures on SMC and particle filters
- Pierre Del Moral, Feynman-Kac models and interacting particle systems

References

- P. Del Moral, A. Doucet and A. Jasra, Sequential Monte Carlo samplers, JRSSB, 2006
- P. Del Moral, A. Doucet and A. Jasra, Sequential Monte Carlo for Bayesian Computation, Bayesian Statistics, 2006
- P. Del Moral, A. Doucet & S.S. Singh, Forward Smoothing using Sequential Monte Carlo, technical report, Cambridge University, 2009
- A. Doucet, Short Courses Lecture Notes (A, B, C)
- P. Del Moral, Feynman-Kac Formulae, Springer-Verlag, 2004
- Sequential MC Methods, M. Davy, 2007
- A Doucet, A Johansen, Particle Filtering and Smoothing: Fifteen years later, in Handbook of Nonlinear Filtering (edts D Crisan and B. Rozovsky), Oxford Univ. Press, 2011
- A. Johansen and A. Doucet, A Note on Auxiliary Particle Filters, Stat. Proba. Letters, 2008.
- A. Doucet et al., Efficient Block Sampling Strategies for Sequential Monte Carlo, (with M. Briers & S. Senecal), JCGS, 2006.
- C. Caron, R. Gottardo and A. Doucet, On-line Changepoint Detection and Parameter Estimation for Genome Wide Transcript Analysis, Stat Comput. 2011.

References

- C.P. Robert & G. Casella, Monte Carlo Statistical Methods, Chapter 11
- J.S. Liu, Monte Carlo Strategies in Scientific Computing, Chapter 3, Springer-Verlag, New York.
- A. Doucet, N. De Freitas & N. Gordon (eds), Sequential Monte Carlo in Practice, Springer-Verlag: 2001
- A. Doucet, N. De Freitas, N.J. Gordon, An introduction to Sequential Monte Carlo, in SMC in Practice, 2001
- D. Wilkison, Stochastic Modelling for Systems Biology, Second Edition, 2006
- E. Ionides, Inference for Nonlinear Dynamical Systems, PNAS, 2006
- J.S. Liu and R. Chen, Sequential Monte Carlo methods for dynamic systems, JASA, 1998
- A. Doucet, Sequential Monte Carlo Methods, Short Course at SAMSI
- A. Doucet, Sequential Monte Carlo Methods & Particle Filters Resources
- Pierre Del Moral, Feynman-Kac models and interacting particle systems (SMC resources)
- A. Doucet, Sequential Monte Carlo Methods, Video Lectures, 2007
- N. de Freitas and A. Doucet, Sequential MC Methods, N. de Freitas and A. Doucet, Video Lectures, 2010

References

- M.K. Pitt and N. Shephard, Filtering via Simulation: Auxiliary Particle Filter, JASA, 1999
- A. Doucet, S.J. Godsill and C. Andrieu, On Sequential Monte Carlo sampling methods for Bayesian filtering, Stat. Comp., 2000
- J. Carpenter, P. Clifford and P. Fearnhead, An Improved Particle Filter for Non-linear Problems, IEE 1999.
- A. Kong, J.S. Liu & W.H. Wong, Sequential Imputations and Bayesian Missing Data Problems, JASA, 1994
- O. Cappe, E. Moulines & T. Ryden, Inference in Hidden Markov Models, Springer-Verlag, 2005
- W Gilks and C. Berzuini, Following a moving target: MC inference for dynamic Bayesian Models, JRSS B, 2001
- G. Poyadjis, A. Doucet and S.S. Singh, Maximum Likelihood Parameter Estimation using Particle Methods, Joint Statistical Meeting, 2005
- N Gordon, D J Salmond, AFM Smith, Novel Approach to nonlinear non Gaussian Bayesian state estimation, IEE, 1993
- Particle Filters, S. Godsill, 2009 (Video Lectures)
- R. Chen and J.S. Liu, Predictive Updating Methods with Application to Bayesian Classification, JRSS B, 1996

References

- C. Andrieu and A. Doucet, Particle Filtering for Partially Observed Gaussian State-Space Models, JRSS B, 2002
- R Chen and J Liu, Mixture Kalman Filters, JRSSB, 2000
- A Doucet, S J Godsill, C Andrieu, On SMC sampling methods for Bayesian Filtering, Stat. Comp. 2000
- N. Kantas, A.D., S.S. Singh and J.M. Maciejowski, An overview of sequential Monte Carlo methods for parameter estimation in general state-space models, in Proceedings IFAC System Identification (SySid) Meeting, 2009
- C. Andrieu, A.Doucet & R. Holenstein, Particle Markov chain Monte Carlo methods, JRSS B, 2010
- C. Andrieu, N. De Freitas and A. Doucet, Sequential MCMC for Bayesian Model Selection, Proc. IEEE Workshop HOS, 1999
- P. Fearnhead, MCMC, sufficient statistics and particle filters, JCGS, 2002
- G. Storvik, Particle filters for state-space models with the presence of unknown static parameters, IEEE Trans. Signal Processing, 2002

References

- C. Andrieu, A. Doucet and V.B. Tadic, Online EM for parameter estimation in nonlinear-non Gaussian state-space models, Proc. IEEE CDC, 2005
- G. Poyadjis, A. Doucet and S.S. Singh, Particle Approximations of the Score and Observed Information Matrix in State-Space Models with Application to Parameter Estimation, *Biometrika*, 2011
- C. Caron, R. Gottardo and A. Doucet, On-line Changepoint Detection and Parameter Estimation for Genome Wide Transcript Analysis, Technical report 2008
- R. Martinez-Cantin, J. Castellanos and N. de Freitas. Analysis of Particle Methods for Simultaneous Robot Localization and Mapping and a New Algorithm: Marginal-SLAM. International Conference on Robotics and Automation
- C. Andrieu, A.D. & R. Holenstein, Particle Markov chain Monte Carlo methods (with discussion), JRSS B, 2010
- A Doucet, Sequential Monte Carlo Methods and Particle Filters, List of Papers, Codes, and Viedo lectures on SMC and particle filters
- Pierre Del Moral, Feynman-Kac models and interacting particle systems

References

- P. Del Moral, A. Doucet and A. Jasra, Sequential Monte Carlo samplers, JRSSB, 2006
- P. Del Moral, A. Doucet and A. Jasra, Sequential Monte Carlo for Bayesian Computation, Bayesian Statistics, 2006
- P. Del Moral, A. Doucet & S.S. Singh, Forward Smoothing using Sequential Monte Carlo, technical report, Cambridge University, 2009
- P. Del Moral, Feynman-Kac Formulae, Springer-Verlag, 2004
- Sequential MC Methods, M. Davy, 2007
- A Doucet, A Johansen, Particle Filtering and Smoothing: Fifteen years later, in Handbook of Nonlinear Filtering (edts D Crisan and B. Rozovsky), Oxford Univ. Press, 2011
- A. Johansen and A. Doucet, A Note on Auxiliary Particle Filters, Stat. Proba. Letters, 2008.
- A. Doucet et al., Efficient Block Sampling Strategies for Sequential Monte Carlo, (with M. Briers & S. Senecal), JCGS, 2006.
- C. Caron, R. Gottardo and A. Doucet, On-line Changepoint Detection and Parameter Estimation for Genome Wide Transcript Analysis, Stat Comput. 2011.

Review of the Kalman Filter for Linear Gaussian Models

Kalman Filter

- As an alternative to SMC, the Kalman filter can provide an optimal estimation for linear Gaussian models.

- Consider a linear Gaussian state-space model,

$$x_n = A_n x_{n-1} + B_n v_n, \; v_n \overset{i.i.d.}{\sim} \mathcal{N}(0, I_x)$$

$$y_n = C_n x_n + D_n w_n, \; w_n \overset{i.i.d.}{\sim} \mathcal{N}(0, I_y)$$

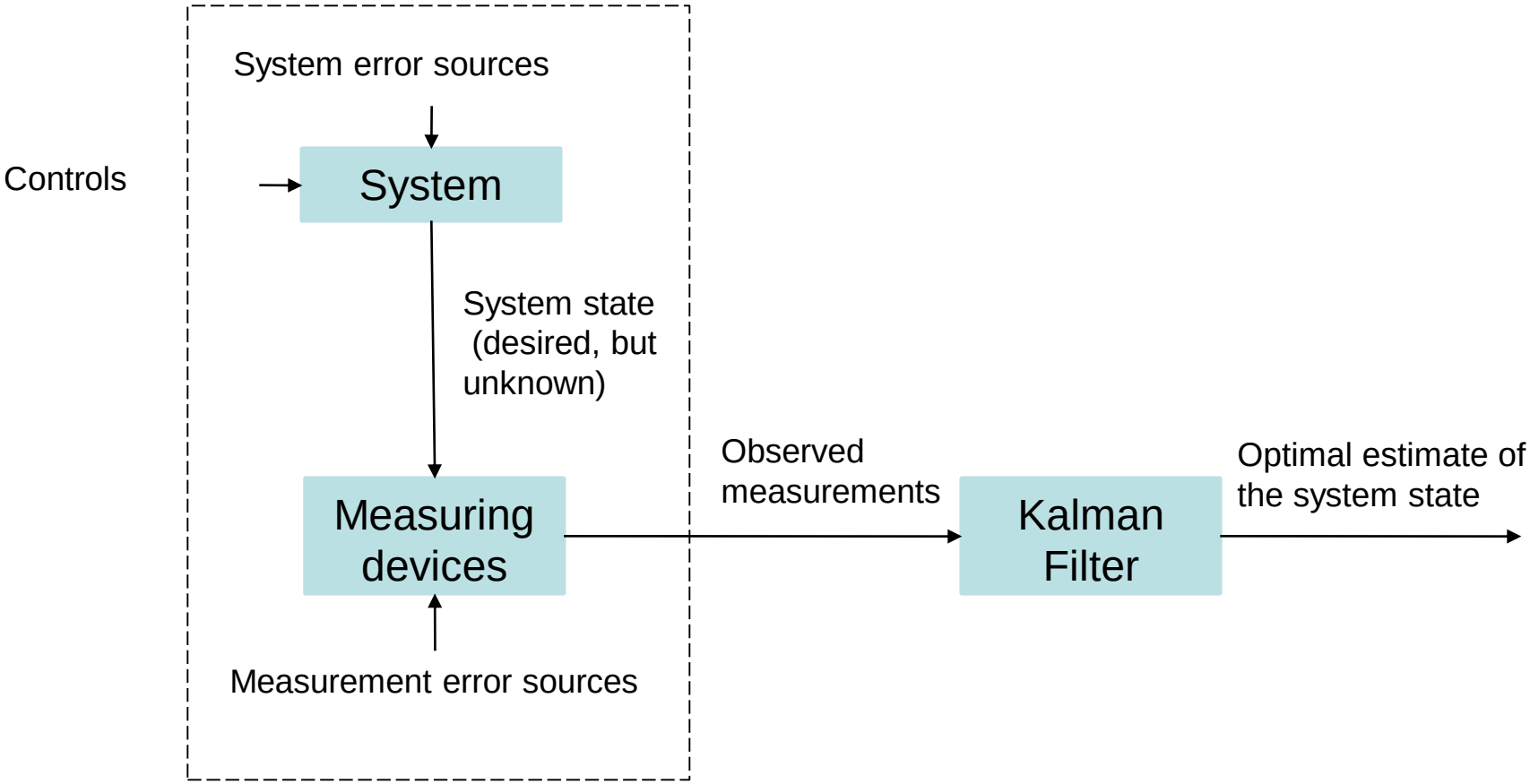
- At each time step, we have a posterior distribution $p(x_n \mid \boldsymbol{y}_{1:n})$ that is subject to a Gaussian distribution.

- The means and covariance matrices of the posteriors can be obtained in a recursive way using the Kalman filter.

  - T. Fletcher, The Kalman Filter Explained
  - N. Shimkin, Derivations of the discrete time Kalman Filter
  - T. Lacey, Tutorial: The Kalman Filter

Kalman Filter

- The Kalman filter provides an efficient "recursive" way to estimate the internal state of a linear dynamic system from a series of noisy measurements.



Kalman Filter

- A Kalman filter combines all available measurement data, plus prior knowledge about the system and measuring device to produce an estimate of the desired variables in such a manner that the error is minimized statistically.

- Kalman filter has been proved to be optimal in cases where the state space model is linear and Gaussian.

- As in SMC, the word "recursive" means that the Kalman filter does not require all previous data to be kept in storage and reprocessed every time a new measurement is taken.

- The Kalman filter can be constructed by minimizing the posterior estimation error of the unknown system states. However, in the following slides, we directly give the algorithm by deriving the distributions of the system states conditioned on the observation data.

Linear Gaussian State-Space Model

- In a generalized linear Gaussian model,

$$x_n = A_n x_{n-1} + B_n v_n, \;\; v_n \overset{i.i.d.}{\sim} \mathcal{N}(0, I_x)$$

$$y_n = C_n x_n + D_n w_n, \;\; w_n \overset{i.i.d.}{\sim} \mathcal{N}(0, I_y)$$

  the state equation and observation equation are:

$$p(x_{n+1} \mid x_n) = \mathcal{N}(A_n x_n, B_n B_n^T)$$

$$p(y_n \mid x_n) = \mathcal{N}(C_n x_n, D_n D_n^T)$$

- The matrices A_n, B_n, C_n and D_n are known but can change with time.

- We wish to infer the probability distribution of x_n given the observations up to y_n, i.e.

$$\textit{Target distribution}: \quad p(x_n \mid \mathbf{y}_{1:n})$$

Linear Gaussian State-Space Model

- Let us start from an initial distribution of x_0 with

$$p(x_0) = \mathcal{N}(x_0 \mid \mu_0, \Sigma_0)$$

- The target distribution $p(x_n \mid \boldsymbol{y}_{1:n})$ can be found recursively

$$p(x_n \mid \boldsymbol{y}_{1:n}) = p(x_n \mid \boldsymbol{y}_{1:n-1}, y_n) \propto p(x_n, y_n \mid \boldsymbol{y}_{1:n-1})$$

$$= \int_{x_{n-1}} p(x_n, x_{n-1}, y_n \mid \boldsymbol{y}_{1:n-1}) dx_{n-1} = \int_{x_{n-1}} p(x_n \mid x_{n-1}) p(x_{n-1}, y_n \mid \boldsymbol{y}_{1:n-1}) dx_{n-1}$$

$$= \int_{x_{n-1}} \underbrace{p(x_n \mid x_{n-1})}_{\text{Gaussian}} \underbrace{p(y_n \mid x_{n-1})}_{\text{Gaussian}} \underbrace{p(x_{n-1} \mid \boldsymbol{y}_{1:n-1})}_{\text{Gaussian}} d\boldsymbol{x}_{n-1}$$

- Obviously, the integration will yield Gaussians, i.e.

$$p(x_n \mid \boldsymbol{y}_{1:n}) = \mathcal{N}(x_n \mid \mu_{n|n}, \Sigma_{n|n})$$

and the task is to derive recursive expressions for $\mu_{n|n}$ and $\Sigma_{n|n}$.

Linear Gaussian State-Space Model

- Suppose at time step $n-1$, we have the target distribution as

$$p\left(x_{n-1} \mid \boldsymbol{y}_{1:n-1}\right) = \mathcal{N}\left(\mu_{n-1|n-1}, \Sigma_{n-1|n-1}\right)$$

- Given data $\boldsymbol{y}_{1:n-1}$, the prediction of x_n is

$$p\left(x_n \mid \boldsymbol{y}_{1:n-1}\right) = p\left(A_n x_{n-1} + B_n v_n \mid \boldsymbol{y}_{1:n-1}\right)$$

which is also a Gaussian distribution with mean

$$\begin{aligned}
\mu_{n|n-1} = \mathbb{E}\left(x_n \mid \boldsymbol{y}_{1:n-1}\right) &= \mathbb{E}\left(A_n x_{n-1} + B_n v_n \mid \boldsymbol{y}_{1:n-1}\right) \\
&= \mathbb{E}\left(A_n x_{n-1} \mid \boldsymbol{y}_{1:n-1}\right) \\
&= A_n \mathbb{E}\left(x_{n-1} \mid \boldsymbol{y}_{1:n-1}\right) \\
&= A_n \mu_{n-1|n-1}
\end{aligned}$$

and covariance

$$\begin{aligned}
\Sigma_{n|n-1} = Cov\left(x_n \mid \boldsymbol{y}_{1:n-1}\right) &= Cov\left(A_n x_{n-1} + B_n v_n \mid \boldsymbol{y}_{1:n-1}\right) \\
&= Cov\left(A_n x_{n-1} \mid \boldsymbol{y}_{1:n-1}\right) + Cov\left(B_n v_n \mid \boldsymbol{y}_{1:n-1}\right) \\
&= A_n Cov\left(x_{n-1} \mid \boldsymbol{y}_{1:n-1}\right) A_n^T + B_n B_n^T \\
&= A_n \Sigma_{n-1|n-1} A_n^T + B_n B_n^T
\end{aligned}$$

Linear Gaussian State-Space Model

- Given data $\boldsymbol{y}_{1:n-1}$, since $y_n = C_n x_n + D_n w_n$, the conditional vector

$$\left(\begin{pmatrix} x_n \\ y_n \end{pmatrix} \middle| \boldsymbol{y}_{1:n-1} \right)$$

  is a Gaussian distribution.

- For $p(y_n \mid \boldsymbol{y}_{1:n-1})$, we have

$$\mathbb{E}(y_n \mid \boldsymbol{y}_{1:n-1}) = \mathbb{E}(C_n x_n + D_n w_n \mid \boldsymbol{y}_{1:n-1}) = C_n \mu_{n|n-1}$$

$$\begin{aligned} Cov(y_n \mid \boldsymbol{y}_{1:n-1}) &= Cov(C_n x_n + D_n w_n \mid \boldsymbol{y}_{1:n-1}) \\ &= Cov(C_n x_n \mid \boldsymbol{y}_{1:n-1}) + Cov(D_n w_n \mid \boldsymbol{y}_{1:n-1}) \\ &= C_n Cov(x_n \mid \boldsymbol{y}_{1:n-1}) C_n^T + D_n D_n^T \\ &= C_n \Sigma_{n|n-1} C_n^T + D_n D_n^T \end{aligned}$$

Linear Gaussian State-Space Model

- In addition, the covariance matrices of x_n and y_n conditioned on $\boldsymbol{y}_{1:n-1}$ are

$$\begin{aligned} Cov\left(x_n \mid \boldsymbol{y}_{1:n-1}, y_n \mid \boldsymbol{y}_{1:n-1}\right) &= Cov\left(x_n \mid \boldsymbol{y}_{1:n-1}, C_n x_n + D_n w_n \mid \boldsymbol{y}_{1:n-1}\right) \\ &= Cov\left(x_n \mid \boldsymbol{y}_{1:n-1}, C_n x_n \mid \boldsymbol{y}_{1:n-1}\right) \\ &= Cov\left(x_n \mid \boldsymbol{y}_{1:n-1}, x_n \mid \boldsymbol{y}_{1:n-1}\right) C_n^T \\ &= Var\left(x_n \mid \boldsymbol{y}_{1:n-1}\right) C_n^T \\ &= \Sigma_{n|n-1} C_n^T \end{aligned}$$

  and similarly

$$Cov\left(y_n \mid \boldsymbol{y}_{1:n-1}, x_n \mid \boldsymbol{y}_{1:n-1}\right) = C_n \Sigma_{n|n-1}$$

- For a random vector $(x,y)^T$ that is distributed in a multivariate Gaussian distribution, we have

$$\begin{aligned} \begin{pmatrix} x \\ y \end{pmatrix} &\sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} \Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_{yy} \end{bmatrix}\right) \\ &\sim \mathcal{N}\left(\begin{bmatrix} \mathbb{E}(x) \\ \mathbb{E}(y) \end{bmatrix}, \begin{bmatrix} Var(x) & Cov(x,y) \\ Cov(y,x) & Var(y) \end{bmatrix}\right) \end{aligned}$$

Linear Gaussian State-Space Model

- Thus, the Gaussian distribution for $\left(\begin{pmatrix} x_n \\ y_n \end{pmatrix} \middle| \boldsymbol{y}_{1:n-1} \right)$ is

$$\left(\begin{pmatrix} x_n \\ y_n \end{pmatrix} \middle| \boldsymbol{y}_{1:n-1} \right) \sim \mathcal{N}\left(\begin{bmatrix} \mathbb{E}(x_n \mid \boldsymbol{y}_{1:n-1}) \\ \mathbb{E}(y_n \mid \boldsymbol{y}_{1:n-1}) \end{bmatrix}, \begin{bmatrix} Var(x_n \mid \boldsymbol{y}_{1:n-1}) & Cov(x_n \mid \boldsymbol{y}_{1:n-1}, y_n \mid \boldsymbol{y}_{1:n-1}) \\ Cov(y_n \mid \boldsymbol{y}_{1:n-1}, x_n \mid \boldsymbol{y}_{1:n-1}) & Var(y_n \mid \boldsymbol{y}_{1:n-1}) \end{bmatrix} \right)$$

$$\sim \mathcal{N}\left[\begin{pmatrix} \mu_{n|n-1} \\ C_n \mu_{n|n-1} \end{pmatrix}, \begin{pmatrix} \Sigma_{n|n-1} & \Sigma_{n|n-1} C_n^T \\ C_n \Sigma_{n|n-1} & C_n \Sigma_{n|n-1} C_n^T + D_n D_n^T \end{pmatrix} \right]$$

- The conditional distribution for a random vector $(x,y)^T$ is

$$x \mid y \sim \mathcal{N}(\mu, \Sigma)$$

where

$$\mu = \mu_x + \Sigma_{xy} \Sigma_{yy}^{-1} (y - \mu_y)$$

$$\Sigma = \Sigma_{xx} - \Sigma_{xy} \Sigma_{yy}^{-1} \Sigma_{yx}$$

Linear Gaussian State-Space Model

- Then according to the properties of conditional multivariate Gaussian distribution, the distribution

$$p\left(x_n \mid y_n, \boldsymbol{y}_{1:n-1}\right) \equiv p\left(x_n \mid \boldsymbol{y}_{1:n}\right)$$

is a Gaussian distribution with mean

$$\mu_{n|n} = \mu_{n|n-1} + \Sigma_{n|n-1} C_n^T S_n^{-1} \left(y_n - C_n \mu_{n|n-1}\right)$$

and covariance

$$\Sigma_{n|n} = \Sigma_{n|n-1} - \Sigma_{n|n-1} C_n^T S_n^{-1} C_n \Sigma_{n|n-1}$$

where

$$S_n = C_n \Sigma_{n|n-1} C_n^T + D_n D_n^T$$

- As a result, we finish the update from $p\left(x_{n-1} \mid \boldsymbol{y}_{1:n-1}\right) = \mathcal{N}\left(\mu_{n-1|n-1}, \sigma^2_{n-1|n-1}\right)$ to

$$p\left(x_n \mid \boldsymbol{y}_{1:n}\right) = \mathcal{N}\left(\mu_{n|n}, \Sigma_{n|n}\right)$$

in a recursive form.

Kalman Filter Algorithm

- The procedure to estimate $p(x_n \mid \boldsymbol{y}_{1:n})$ using Kalman Filter proceeds as follows:
  - Choose initial distribution $p(x_0) = \mathcal{N}(\mu_0, \Sigma_0)$ i.e.

$$\mu_{0|0} = \mu_0 \quad , \quad \Sigma_{0|0} = \Sigma_0$$

At step *n-1*

  - Predict the distribution $p(x_n \mid \boldsymbol{y}_{1:n-1}) = \mathcal{N}(x_n \mid \mu_{n|n-1}, \Sigma_{n|n-1})$ where

$$\mu_{n|n-1} = A_n \mu_{n-1|n-1} \qquad \Sigma_{n|n-1} = A_n \Sigma_{n-1|n-1} A_n^T + B_n B_n^T$$

  - Given a new observation y_n, update to the distribution

$$p(x_n \mid \boldsymbol{y}_{1:n}) = \mathcal{N}(x_n \mid \mu_{n|n}, \Sigma_{n|n})$$

where

$$\mu_{n|n} = \mu_{n|n-1} + \Sigma_{n|n-1} C_n^T S_n^{-1} (y_n - C_n \mu_{n|n-1})$$

$$\Sigma_{n|n} = \Sigma_{n|n-1} - \Sigma_{n|n-1} C_n^T S_n^{-1} C_n \Sigma_{n|n-1}$$

$$S_n = C_n \Sigma_{n|n-1} C_n^T + D_n D_n^T$$

Integrating Kalman Filter and SMC

- Since SMC algorithm is also recursive, it is possible to integrate the Kalman filter into SMC.

- Next we will develop an algorithm that combines SMC and Kalman filter for conditionally linear Gaussian models.

Conditional Linear Gaussian Model

Conditional Dynamic Linear Model

- A conditional dynamic linear model (CDLM) can be generally defined as

$$Z_n = A(X_n)Z_{n-1} + B(X_n)V_n \quad , \quad Z_1 \sim \mathcal{N}(m_1, \Sigma_1), \quad V_n \overset{i.i.d.}{\sim} \mathcal{N}(0, \Sigma_v)$$

$$Y_n = C(X_n)Z_n + D(X_n)W_n \quad , \quad W_n \overset{i.i.d.}{\sim} \mathcal{N}(0, \Sigma_w)$$

  where all coefficient matrices are functions of the indicator random variable X_n. Both $\{X_n, Z_n\}, n \geq 1$, are unobserved.

- The X_n, which can be either continuous or discrete, is a latent indicator process with certain probabilistic structure. For example:

$$X_1 \sim \mu, \quad X_n \mid X_{n-1} = x \sim f(. \mid x)$$

- Here Z_n is an unobserved Markov process and Y_n is the corresponding observation.

- An important feature of the CDLM is that given the trajectory of the indicator variable X_n, the system is Gaussian and linear.
  - R.Chen,J.S.Liu, Mixture Kalman Filters, Journal of the Royal Statistical Society, 2000,62B:p493

Conditional Dynamic Linear Models

- Finite Markov Chain in colored and white noises (ion channel):
  - X_n finite Markov chain

$$Z_n = AZ_{n-1} + BV_n \,,\; Y_n = Z_n + X_n + W_n$$

- Flat fading channels (comms): X_n finite Markov chain

$$Z_n = AZ_{n-1} + BV_n \,,\; Y_n = Z_n X_n + W_n$$

- Time varying autoregressions in noise (speech denoising)

$$X_n = X_{n-1} + E_n \,,$$
$$Z_n = A(X_n) Z_{n-1} + B(X_n) V_n,$$
$$Y_n = Y_{n-L+1:n}^T Z_n + W_n$$

Conditional Dynamic Linear Model

- Since this is an extension of the linear dynamic Gaussian model, we could estimate using SMC methods.

- Consider the posterior distribution:
  - Since the indicator variable X_n and unobserved states Z_n are both unknown, we have the following posterior distribution

$$p(\boldsymbol{x}_{1:n}, \boldsymbol{z}_{1:n} \mid \boldsymbol{y}_{1:n}) \propto p(\boldsymbol{x}_{1:n}, \boldsymbol{z}_{1:n}, \boldsymbol{y}_{1:n}) = p(\boldsymbol{x}_{1:n})\, p(\boldsymbol{z}_{1:n} \mid \boldsymbol{x}_{1:n})\, p(\boldsymbol{y}_{1:n} \mid \boldsymbol{x}_{1:n}, \boldsymbol{z}_{1:n})$$

$$= p(\boldsymbol{x}_{1:n}) \times \prod_{k=2}^{n} \mathcal{N}\left(z_k; A(x_k) z_{k-1}, B(x_k) B^T(x_k)\right) \times \prod_{k=1}^{n} \mathcal{N}\left(y_k; C(x_k) z_k, D(x_k) D^T(x_k)\right)$$

  - This is similar to the framework for dynamic models discussed in earlier lectures, except that this model is controlled by the random indicator variable X_n.

  - If X_n is an unobserved Markov process: $X_1 \sim \mu$, $X_i \mid (X_{i-1} = x) \sim f(. \mid x)$, then:

$$p(\boldsymbol{x}_{1:n}) = \mu(x_1) \prod_{k=2}^{n} f(x_k \mid x_{k-1})$$

Conditional Dynamic Linear Model

- This is similar to the framework for dynamic (HMM) models discussed earlier, except that this model is controlled by the random indicator variable X_n.

- $\{Z_n, X_n\}$ is a Markov process and the observations $\{Y_n\}$ are conditionally independent given $\{Z_n, X_n\}$.

$$p\left(\boldsymbol{z}_{1:n}, \boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right) \propto$$

$$\mu(x_1)\prod_{k=2}^{n} f(x_k \mid x_{k-1})$$

$$\times\, p\left(z_1\right) \times \prod_{k=2}^{n} \mathcal{N}\left(z_k; A(x_k) z_{k-1}, B(x_k) B^T(x_k)\right) \times \prod_{k=1}^{n} \mathcal{N}\left(y_k; C(x_k) z_k, D(x_k) D^T(x_k)\right)$$

Conditional Dynamic Linear Model

- One can use the SMC method to sample from

$$p\left(\boldsymbol{x}_{1:n}, \boldsymbol{z}_{1:n} \mid \boldsymbol{y}_{1:n}\right)$$

- This approach would not use the particular structure of the CDLM and thus is very inefficient.

- An important feature of the CDLM is that given the trajectory of an indicator variable X_n, the system is Gaussian and linear.

- We can see that

$$p\left(\boldsymbol{x}_{1:n}, \boldsymbol{z}_{1:n} \mid \boldsymbol{y}_{1:n}\right) = p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right) \underbrace{p\left(\boldsymbol{z}_{1:n} \mid \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n}\right)}_{Gaussian}$$

- Conditioned on $\boldsymbol{x}_{1:n}$, the distribution $p\left(\boldsymbol{z}_{1:n} \mid \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n}\right)$ is a Gaussian distribution.

Conditional Dynamic Linear Model

- It is only necessary to estimate through SMC the marginal distribution

$$p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right) \propto p\left(\boldsymbol{x}_{1:n}\right) p\left(\boldsymbol{y}_{1:n} \mid \boldsymbol{x}_{1:n}\right)$$

where the likelihood $p(\boldsymbol{y}_{1:n}|\boldsymbol{x}_{1:n})$ is also given by Kalman filter because this is a standard linear Gaussian model conditioned on $\boldsymbol{x}_{1:n}$.

- Accordingly, when simulating CDLM with SMC, **the target distribution is the marginal for the indicator variable $\boldsymbol{x}_{1:n}$**

$$p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right) \propto p\left(\boldsymbol{x}_{1:n}\right) p\left(\boldsymbol{y}_{1:n} \mid \boldsymbol{x}_{1:n}\right)$$

- We need to be able to compute up to a normalizing constant this target $p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right)$ distribution for SMC to apply. This may be possible even if $\{X_n\}$ is non-Markovian. Also note that

$$p(\boldsymbol{y}_{1:n}|\boldsymbol{x}_{1:n}) \neq \prod_{k=1}^{n} p(y_k|x_k)$$

Kalman Filter

- Recall the Kalman filter. Consider the standard linear Gaussian model with time-varying coefficients

$$x_n = A_n x_{n-1} + B_n v_n, \;\; v_n \overset{i.i.d.}{\sim} \mathcal{N}(0, I_x)$$
$$y_n = C_n x_n + D_n w_n, \;\; w_n \overset{i.i.d.}{\sim} \mathcal{N}(0, I_y)$$

- If the coefficients are fixed at each time step, we can use Kalman filter to sequentially estimate the mean and covariance

$$\mu_{t|t-1} = A_t \mu_{t-1|t-1}$$
$$\sigma^2_{t|t-1} = A_t \Sigma_{t-1|t-1} A_t^T + B_t B_t^T$$
$$y_{t|t-1} = C_t \mu_{t|t-1}$$
$$S_t = C_t \Sigma_{t|t-1} C_t^T + D_t D_t^T$$
$$\mu_{t|t} = \mu_{t|t-1} + \Sigma_{t|t-1} C_t^T S_t^{-1} (y_t - y_{t|t-1})$$
$$\Sigma_{t|t} = \Sigma_{t|t-1} - \Sigma_{t|t-1} C_t^T S_t^{-1} C_t \Sigma_{t|t-1}$$

$\Rightarrow$

$$p(x_{t-1} \mid \boldsymbol{y}_{1:t-1}) = \mathcal{N}(\mu_{t-1|t-1}, \Sigma_{t-1|t-1})$$
$$p(x_t \mid \boldsymbol{y}_{1:t-1}) = \mathcal{N}(\mu_{t|t-1}, \Sigma_{t|t-1})$$
$$p(y_t \mid \boldsymbol{y}_{1:t-1}) = \mathcal{N}(y_{t|t-1}, S_t)$$
$$p(x_t \mid \boldsymbol{y}_{1:t}) = \mathcal{N}(\mu_{t|t}, \Sigma_{t|t})$$

Sequential Importance Sampling

- To sample from our target $\pi_n\left(\boldsymbol{x}_{1:n}\right) = p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right)$, we can use sequential importance sampling:

$$\boldsymbol{X}_{1:n}^{(i)} \sim q_n\left(\boldsymbol{x}_{1:n}\right)$$

where

$$q_n\left(\boldsymbol{x}_{1:n}\right) = q_{n-1}\left(\boldsymbol{x}_{1:n-1}\right) q_n\left(x_n \mid \boldsymbol{x}_{1:n-1}\right) = q_1\left(x_1\right) \prod_{k=2}^{n} q_k\left(x_k \mid \boldsymbol{x}_{1:k-1}\right)$$

- This is valid regardless if $\{X_n\}$ is a Markov process or not.

- The weights at step n can be computed sequentially as:

$$w_n\left(\boldsymbol{x}_{1:n}\right) = \frac{\pi_n\left(\boldsymbol{x}_{1:n}\right)}{q_n\left(\boldsymbol{x}_{1:n}\right)} = \underbrace{\frac{\pi_{n-1}\left(\boldsymbol{x}_{1:n-1}\right)}{q_{n-1}\left(\boldsymbol{x}_{1:n-1}\right)}}_{w_{n-1}\left(\boldsymbol{x}_{1:n-1}\right)} \frac{\pi_n\left(\boldsymbol{x}_{1:n}\right)}{\pi_{n-1}\left(\boldsymbol{x}_{1:n-1}\right) q_n\left(x_n \mid \boldsymbol{x}_{1:n-1}\right)}$$

$$\propto w_{n-1}\left(\boldsymbol{x}_{1:n-1}\right) \frac{\pi_n\left(\boldsymbol{x}_{1:n}\right)}{\pi_{n-1}\left(\boldsymbol{x}_{1:n-1}\right) q_n\left(x_n \mid \boldsymbol{x}_{1:n-1}\right)}$$

Sequential Importance Sampling Resampling

- Thus, the **SIS Resampling algorithm** for CDLM is

  At time step n, for i=1,2,…,N (index for particles)

  - Sampling
    generate $X_n^{(i)}$ from a proposal distribution $q\left(x_n \mid \boldsymbol{X}_{1:n-1}^{(i)}\right)$

  - Updating
    set $\boldsymbol{X}_{1:n}^{(i)} = \left\{\boldsymbol{X}_{1:n-1}^{(i)}, X_n^{(i)}\right\}$. The importance weight is

$$W_n^{(i)} \propto \underbrace{W_{n-1}\left(\boldsymbol{X}_{1:n-1}^{(i)}\right)}_{W_{n-1}^{(i)}} \frac{\pi_n\left(\boldsymbol{X}_{1:n}^{(i)}\right)}{\pi_{n-1}\left(\boldsymbol{X}_{1:n-1}^{(i)}\right) q_n\left(X_n^{(i)} \mid \boldsymbol{X}_{1:n-1}^{(i)}\right)}$$

  - Resampling
    If variance of weights is high, resample $\left\{\boldsymbol{X}_{1:n}^{(i)}, W_n^{(i)}\right\}$ to obtain a new population of particles and weights $\left\{\boldsymbol{X}_{1:n}^{(i)}, W_n^{(i)} = 1/N\right\}$

Sequential Importance Sampling

- As we did for dynamical state systems before, we select the importance distribution to minimize the variance of the incremental importance weight conditional upon $\boldsymbol{x}_{1:n-1}$

$$w_n\left(\boldsymbol{x}_{1:n}\right) = w_{n-1}\left(\boldsymbol{x}_{1:n-1}\right)\frac{\pi_n\left(\boldsymbol{x}_{1:n}\right)}{\pi_{n-1}\left(\boldsymbol{x}_{1:n-1}\right)q_n\left(x_n \mid \boldsymbol{x}_{1:n-1}\right)}$$

$$= w_{n-1}\left(\boldsymbol{x}_{1:n-1}\right)\frac{\pi_n\left(\boldsymbol{x}_{1:n-1}\right)}{\pi_{n-1}\left(\boldsymbol{x}_{1:n-1}\right)}\frac{\pi_n\left(x_n \mid \boldsymbol{x}_{1:n-1}\right)}{q_n\left(x_n \mid \boldsymbol{x}_{1:n-1}\right)}$$

- The incremental weight needs to be computed with cost independent of n to avoid increasing complexity with time.

- Thus as before the optimal importance distribution is:

$$q_n\left(x_n \mid \boldsymbol{x}_{1:n-1}\right) = \pi_n\left(x_n \mid \boldsymbol{x}_{1:n-1}\right)$$

Conditional Dynamic Linear Model

- We can estimate our target distribution $\pi_n\left(\boldsymbol{x}_{1:n}\right) = p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right)$ in this general SMC framework.

- The importance distribution at time n is given as: $q\left(x_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n-1}\right)$ and

$$p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right) \propto p\left(\boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n}\right) = p\left(\boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}, x_n, y_n\right) = p\left(\boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}\right) p\left(x_n, y_n \mid \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}\right)$$
$$= p\left(\boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}\right) p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right) f\left(x_n \mid x_{n-1}\right)$$

- The importance weights are now updated as follows:

$$w_n\left(\boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n}\right) \propto w_{n-1}\left(\boldsymbol{x}_{1:n-1}\right) \frac{p\left(\boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n}\right)}{p\left(\boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}\right) q_n\left(x_n \mid \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n}\right)}$$
$$= w_{n-1}\left(\boldsymbol{x}_{1:n-1}\right) \frac{p\left(\boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}\right) p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right) f\left(x_n \mid x_{n-1}\right)}{p\left(\boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}\right) q_n\left(x_n \mid \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n}\right)}$$
$$= w_{n-1}\left(\boldsymbol{x}_{1:n-1}\right) \frac{p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right) f\left(x_n \mid x_{n-1}\right)}{q_n\left(x_n \mid \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n}\right)}$$

SISR for the Conditionally Linear Gaussian Model

- Consider at time n-1 that we have $\left\{ \boldsymbol{X}_{1:n-1}^{(i)}, W_{n-1}^{(i)} \right\}$

  At time step n, for i=1,2,…,N (index for particles)

  - Sampling
    generate $X_n^{(i)}$ from a proposal distribution $q\left(X_n^{(i)} \mid \boldsymbol{y}_{1:n}, \boldsymbol{X}_{n-1}^{(i)} \right)$

  - Updating
    set $\boldsymbol{X}_{1:n}^{(i)} = \left\{ \boldsymbol{X}_{1:n-1}^{(i)}, X_n^{(i)} \right\}$. The importance weight is

$$W_n^{(i)} \propto \underbrace{W_{n-1}\left(\boldsymbol{X}_{1:n-1}^{(i)} \right)}_{W_{n-1}^{(i)}} \frac{p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n}^{(i)} \right) f\left(X_n^{(i)} \mid X_{n-1}^{(i)} \right)}{q_n\left(X_n^{(i)} \mid \boldsymbol{X}_{1:n-1}^{(i)}, \boldsymbol{y}_{1:n} \right)}$$

  - Resampling
    If variance of weights is high, resample $\left\{ \boldsymbol{X}_{1:n}^{(i)}, W_n^{(i)} \right\}$ to obtain a new population of particles and weights $\left\{ \boldsymbol{X}_{1:n}^{(i)}, W_n^{(i)} = 1/N \right\}$

SISR for the Conditionally Linear Gaussian Model

- Based on our weight update formula,

$$W_n^{(i)} \propto \underbrace{W_{n-1}\left(\boldsymbol{X}_{1:n-1}^{(i)}\right)}_{W_{n-1}^{(i)}} \frac{p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n}^{(i)}\right) f\left(X_n^{(i)} \mid X_{n-1}^{(i)}\right)}{q_n\left(X_n^{(i)} \mid \boldsymbol{X}_{1:n-1}^{(i)}, \boldsymbol{y}_{1:n}\right)}$$

  the algorithm requires that we save in memory $\left\{\boldsymbol{X}_{1:n}^{(i)}\right\}$ at each n in order to compute $w_n^{(i)} = w_n\left(\boldsymbol{X}_{1:n}^{(i)}, \boldsymbol{y}_{1:n}\right)$.

- In addition a good proposal $q_n\left(X_n^{(i)} \mid \boldsymbol{X}_{1:n-1}^{(i)}, \boldsymbol{y}_{1:n}\right)$ should depend on $p\left(y_n \mid \boldsymbol{X}_{1:n}^{(i)}, \boldsymbol{y}_{1:n-1}\right)$

- The increasing complexity with n at this point implies that this algorithm is not practical.

SMC Algorithm for CLGM

- In the calculation of importance weight
  - $f\left(X_n^{(i)} \mid X_{n-1}^{(i)}\right)$ is prescribed in the model
  - $q\left(X_n^{(i)} \mid \boldsymbol{y}_{1:n}, \boldsymbol{X}_{1:n-1}^{(i)}\right)$ is the proposal distribution designed by us
  - the problem is how to calculate $p\left(y_n \mid \boldsymbol{X}_{1:n}^{(i)}, \boldsymbol{y}_{1:n-1}\right)$

- Implementing the previous algorithm would require storing the whole paths $\left\{\boldsymbol{X}_{1:n}^{(i)}\right\}$ which is not practical.

Rao-Blackwellised SMC for CDLM

- The key is to realize that $p\left(y_n \mid \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n-1}\right)$ is a Gaussian distribution of mean $\mu\left(\boldsymbol{x}_{1:n}\right)$ and covariance $\Sigma\left(\boldsymbol{x}_{1:n}\right)$. Indeed note that:

$$p\left(y_n \mid \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n-1}\right) = \int p\left(y_n, z_{n-1}, z_n \mid \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n-1}\right) d\boldsymbol{z}_{n-1:n} = \int \underbrace{p\left(y_n \mid z_{n-1}, z_n, \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n-1}\right)}_{g\left(y_n \mid z_n, x_n\right)} \underbrace{p\left(z_n, z_{n-1} \mid x_n, \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}\right)}_{p\left(z_n \mid z_{n-1}, \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n-1}\right) p\left(z_{n-1} \mid \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n-1}\right)} d\boldsymbol{z}_{n-1:n} =$$

$$\int g\left(y_n \mid z_n, x_n\right) f\left(z_n \mid z_{n-1}\right) \underbrace{p\left(z_{n-1} \mid \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}\right)}_{\text{Gaussian}} d\boldsymbol{z}_{n-1:n}$$

Its mean and covariance can be computed by the Kalman filter (the model is conditionally linear Gaussian)

$$p\left(z_{n-1} \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n-1}\right) =$$
$$\mathcal{N}\left\{z_{n-1}; \mu_{n-1}\left(\boldsymbol{x}_{1:n-1}\right), \Sigma_{n-1}\left(\boldsymbol{x}_{1:n-1}\right)\right\}$$

Instead of $\boldsymbol{X}_{1:n}^{(i)}$, we only need to save

$$X_n^{(i)}, \mu_n\left(\boldsymbol{X}_{1:n}^{(i)}\right) \text{ and } \Sigma_n\left(\boldsymbol{X}_{1:n}^{(i)}\right)$$

- $X_n^{(i)}, \mu_n\left(\boldsymbol{X}_{1:n}^{(i)}\right) \text{ and } \Sigma_n\left(\boldsymbol{X}_{1:n}^{(i)}\right)$ are sufficient statistics summarizing all information about $\left(\boldsymbol{X}_{1:n}^{(i)}, \boldsymbol{y}_{1:n}\right)$. This is the only info to be saved at *n*.
- Rao-Blackwellised SMC is now widely used in robotics to address the SLAM problem.
  - A. Doucet, et al., Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks, Report, 2000.

Revised Implementation of the SISR for the CLGM

- Consider time $n-1$ when we have $\left\{X_{n-1}^{(i)}, \mu_{n-1}^{(i)}, \Sigma_{n-1}^{(i)}, W_{n-1}^{(i)}\right\}$

At time step n, for $i = 1,2,\ldots,N$ (index for particles)

- Sampling:
  sample $X_n^{(i)}$ from a proposal distribution $q\left(. \mid y_n, X_{n-1}^{(i)}, \mu_{n-1}^{(i)}, \Sigma_{n-1}^{(i)}\right)$

- Updating:
  set $\boldsymbol{X}_{1:n}^{(i)} = \left\{\boldsymbol{X}_{1:n-1}^{(i)}, X_n^{(i)}\right\}$. The importance weight is

$$W_n^{(i)} \propto W_{n-1}^{(i)} \frac{p\left(y_n \mid X_n^{(i)}, \mu_{n-1}^{(i)}, \Sigma_{n-1}^{(i)}\right) f\left(X_n^{(i)} \mid X_{n-1}^{(i)}\right)}{q_n\left(X_n^{(i)} \mid X_{n-1}^{(i)}, \mu_{n-1}^{(i)}, \Sigma_{n-1}^{(i)}, y_n\right)}$$

- Kalman Filter: Use the Kalman filter to compute $\left\{\mu_n^{(i)}, \Sigma_n^{(i)}\right\}, i = 1,\ldots,N$

- Resampling:
  If variance of weights is high, resample $\left\{X_n^{(i)}, \mu_n^{(i)}, \Sigma_n^{(i)}, W_n^{(i)}\right\}$ to obtain a new population of particles and weights $\left\{X_n^{(i)}, \mu_n^{(i)}, \Sigma_n^{(i)}, W_n^{(i)} = 1/N\right\}$

SMC Algorithm for CDLM

- The Kalman step updates are reviewed below
  - Kalman Step
    obtain $KF_n^{(i)} \equiv \left(\mu_n\left(\boldsymbol{X}_{1:n}^{(i)}\right), \Sigma_n\left(\boldsymbol{X}_{1:n}^{(i)}\right)\right)$ by a one-step Kalman filter, conditional on $\left\{KF_{n-1}^{(i)}, y_n, X_n^{(i)}\right\}$

$$p\left(z_{n-1} \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n-1}^{(i)}\right) = \mathcal{N}\left(\mu_{n-1}^{(i)}, \Sigma_{n-1}^{(i)}\right)$$

$$p\left(z_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n}^{(i)}\right) = \mathcal{N}\left(\mu_{n|n-1}, \Sigma_{n|n-1}\right)$$

$$p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n}^{(i)}\right) = \mathcal{N}\left(y_{n|n-1}, S_n\right)$$

$$p\left(z_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{X}_{1:n}^{(i)}\right) = \mathcal{N}\left(\mu_n^{(i)}, \Sigma_n^{(i)}\right)$$

$$\mu_{n|n-1} = A_n \mu_{n-1}^{(i)}$$

$$\Sigma_{n|n-1} = A_n \Sigma_{n-1}^{(i)} A_n^T + B_n B_n^T$$

$$y_{n|n-1} = C_n \mu_{n|n-1}$$

$$S_n = C_n \Sigma_{n|n-1} C_n^T + D_n D_n^T$$

$$\mu_n^{(i)} = \mu_{n|n-1} + \Sigma_{n|n-1} C_n^T S_n^{-1}\left(y_n - y_{n|n-1}\right)$$

$$\Sigma_n^{(i)} = \Sigma_{n|n-1} - \Sigma_{n|n-1} C_n^T S_n^{-1} C_n \Sigma_{n|n-1}$$

- Recall our dynamical system:

$$x_n = A_n x_{n-1} + B_n v_n, \; v_n \overset{i.i.d.}{\sim} \mathcal{N}\left(0, I_x\right)$$

$$y_n = C_n x_n + D_n w_n, \; w_n \overset{i.i.d.}{\sim} \mathcal{N}\left(0, I_y\right)$$

Revised Implementation of the SISR for the CLGM

- Our Monte Carlo approximation of $p\left(x_n \mid \boldsymbol{y}_{1:n}\right)$ is given as:

$$Inference\ on\ X_n\colon \hat{p}_N(x_n|\boldsymbol{y}_{1:n}) = \sum_{i=1}^{N} W_n^{(i)} \delta_{X_n^{(i)}}(x_n)$$

- Inference on Z_n is based on the following (requires N Kalman filters):

$$Inference\ on\ Z_n\colon \hat{p}_N(z_n|\boldsymbol{y}_{1:n}) = \int \hat{p}_N(z_n, \boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}) d\boldsymbol{x}_{1:n}$$

$$= \int \hat{p}_N(z_n|\boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n}) \hat{p}_N(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}) d\boldsymbol{x}_{1:n} = \sum_{i=1}^{N} W_n^{(i)} \hat{p}_N\left(z_n|X_n^{(i)}, \boldsymbol{y}_{1:n}\right)$$

$$= \sum_{i=1}^{N} W_n^{(i)} \mathcal{N}\left(z_n|\mu_n^{(i)}, \Sigma_n^{(i)}\right), \ where\ \hat{p}_N(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}) = \sum_{i=1}^{N} W_n^{(i)} \delta_{\boldsymbol{X}_{1:n}^{(i)}}(\boldsymbol{x}_{1:n})$$

- Note using the Eq. above we can write:

$$\widehat{\mathbb{E}}_N(Z_n|\boldsymbol{y}_{1:n}) = \sum_{i=1}^{N} W_n^{(i)} \mu_n^{(i)}$$

$$\widehat{Cov}_N(Z_n|\boldsymbol{y}_{1:n}) = \sum_{i=1}^{N} W_n^{(i)} \left(\Sigma_n^{(i)} + \mu_n^{(i)} \mu_n^{(i)^T}\right) - \widehat{\mathbb{E}}(Z_n|\boldsymbol{y}_{1:n}) \widehat{\mathbb{E}}(Z_n|\boldsymbol{y}_{1:n})^T$$

Rao-Blackwellised SMC for CLGM

- Using Kalman filter, we can estimate $\mu_n(\boldsymbol{x}_{1:n})$ and $\Sigma_n(\boldsymbol{x}_{1:n})$ in a recursive way. At time n+1, given the estimated X_{n+1} and observation data y_{n+1}, $\mu_{n+1}(\boldsymbol{x}_{1:n+1})$ and $\Sigma_{n+1}(\boldsymbol{x}_{1:n+1})$ are completely determined (a deterministic procedure).

- In practice instead of $\boldsymbol{X}_{1:n}^{(i)}, \boldsymbol{y}_{1:n}$, we only save $X_n^{(i)}, \mu_n\left(\boldsymbol{X}_{1:n}^{(i)}\right) and\ \Sigma_n\left(\boldsymbol{X}_{1:n}^{(i)}\right)$

- In other words, $\left(X_n^{(i)}, \mu_n^{(i)}, \Sigma_n^{(i)}\right)$ are sufficient statistics summarizing all information about $\left(\boldsymbol{X}_{1:n}^{(i)}, \boldsymbol{y}_{1:n}\right)$.

- The Kalman filter can be also used to construct proposal distributions.

Conditional Linear Gaussian Model

- The optimal importance distribution is given as:

$$q\left(x_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n-1}\right) = p\left(x_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n-1}\right) = p\left(x_n \mid y_n, \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n-1}\right)$$

$$= \frac{p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right) f\left(x_n \mid x_{n-1}\right)}{\int p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right) f\left(x_n \mid x_{n-1}\right) dx_n}$$

$$\propto p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right) f\left(x_n \mid x_{n-1}\right)$$

- The importance weights are now updated as follows:

$$w_n\left(\boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n}\right) \propto w_{n-1}\left(\boldsymbol{x}_{1:n-1}\right) \frac{p\left(\boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n}\right)}{p\left(\boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}\right) q_n\left(x_n \mid \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n}\right)}$$

$$= w_{n-1}\left(\boldsymbol{x}_{1:n-1}\right) \frac{p\left(\boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}\right) p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right) f\left(x_n \mid x_{n-1}\right)}{p\left(\boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n-1}\right) q_n\left(x_n \mid \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n}\right)}$$

$$= w_{n-1}\left(\boldsymbol{x}_{1:n-1}\right) \int p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right) f\left(x_n \mid x_{n-1}\right) dx_n$$

Importance Distribution

- Consider the case when the latent variable X_n takes values in a finite discrete set I. The optimal importance distribution

$$q\left(x_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n-1}\right) = p\left(x_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n-1}\right)$$

was decomposed in the previous slide as:

$$p\left(x_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n-1}\right) \propto p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right) \underbrace{p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right)}_{Kalman\ filter}$$

Here $p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right)$ is a normal distribution which is estimated by Kalman filter

$$p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right) = \mathcal{N}\left(y_n \mid y_{n|n-1}, S_n\right)$$

and $p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right)$ is generally predefined in the model.

Importance Weights

- We can directly sample from this proposal distribution by inspecting all the possible values of x_n and calculate the probability densities.

$$v_i = p\left(x_n = i \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n-1}\right), \forall i \in I$$

where v_i is the normalized probability

$$v_i = \frac{p\left(x_n = i \mid \boldsymbol{x}_{1:n-1}\right) p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n-1}, x_n\right)}{\sum_j p\left(x_n = j \mid \boldsymbol{x}_{1:n-1}\right) p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n-1}\right)}$$

Then we sample x_n according to probability $\{v_i\}$

- The importance weights are then:

$$w_n^{(i)} = w_{n-1}^{(i)} \frac{p\left(y_n \mid \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n-1}\right) p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right)}{p\left(x_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n-1}\right)} = w_{n-1}^{(i)} \frac{p\left(y_n \mid \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n-1}\right) p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right)}{\frac{p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right) p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right)}{\sum_j p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n-1}, x_n = j\right) p\left(x_n = j \mid \boldsymbol{x}_{1:n-1}\right)}}$$

$$= w_{n-1}^{(i)} \sum_j \underbrace{p\left(x_n = j \mid \boldsymbol{x}_{1:n-1}\right) p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n-1}, x_n = j\right)}_{v_j}$$

SMC Algorithm for CLGM

- In this case, by estimating the normal distributions, e.g.

$$p\left(z_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{X}_{1:n}^{(i)}\right) = \mathcal{N}\left(\mu_{n|n}, \sigma_{n|n}^2\right), \quad p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n}^{(i)}\right) = \mathcal{N}\left(y_{n|n-1}, S_n\right)$$

  the Kalman filter serves for two purposes

  - estimate the probability $p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right)$ to make samples of x_n and calculate the importance weight

  - Use the posterior distribution $p\left(z_n \mid \boldsymbol{y}_{1:n}, x_n\right)$ to estimate z_n which is the Markov state of interest.

- For generality, we use $KF_n = \left(\mu_n\left(\boldsymbol{x}_{1:n}\right), \Sigma_n\left(\boldsymbol{x}_{1:n}\right)\right)$ as an abbreviation of the mean and variance of the distribution

$$p\left(z_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n}\right) = \mathcal{N}\left(\mu_n\left(\boldsymbol{x}_{1:n}\right), \Sigma_n\left(\boldsymbol{x}_{1:n}\right)\right)$$

SMC Algorithm for CLGM

- This distribution is updated by the Kalman filter, and the other distributions we may need, e.g. $p(y_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n})$, are byproduct of the update process.

- Thus $\mu_n(\boldsymbol{x}_{1:n}), \Sigma_n(\boldsymbol{x}_{1:n})$ are sufficient statistics summarizing all the information about $\boldsymbol{x}_{1:n}$ and $\boldsymbol{y}_{1:n}$. We can use the information provided by $KF_n = (\mu_n(\boldsymbol{x}_{1:n}), \Sigma_n(\boldsymbol{x}_{1:n}))$ to construct the proposal distribution and calculate the importance weight.

- Now the proposal distribution can be formulated in a generalized form

$$q\left(x_n \mid y_n, X_{n-1}^{(i)}, KF_{n-1}^{(i)}\right)$$

  which means the distribution is constructed based on knowledge about previous latent variables X_{n-1} and probability densities derived by Kalman filter.

SMC Algorithm for CDLM

- **SIS Resampling algorithm** combined with Kalman Filter for CDLM
- At time *n*
  - Sampling Step
    generate $X_n^{(i)}$ from $q\left(X_n^{(i)} \mid y_n, X_{n-1}^{(i)}, KF_{n-1}^{(i)}\right)$ and update $\boldsymbol{X}_{1:n}^{(i)} = \left(\boldsymbol{X}_{1:n-1}^{(i)}, X_n^{(i)}\right)$
  - Kalman Step
    obtain $KF_n^{(i)}$ by a one-step Kalman filter, conditional on $\left\{KF_{n-1}^{(i)}, y_n, X_n^{(i)}\right\}$
  - Updating Step
    the importance weight $$w_n^{(i)} = w_{n-1}^{(i)} \underbrace{\frac{p\left(y_n \mid X_n^{(i)}, KF_{n-1}^{(i)}\right) f\left(X_n^{(i)} \mid \boldsymbol{X}_{1:n-1}^{(i)}\right)}{q\left(X_n^{(i)} \mid y_n, X_{n-1}^{(i)}, KF_{n-1}^{(i)}\right)}}_{p\left(y_n \mid X_n^{(i)}, KF_{n-1}^{(i)}\right) = \sum_{j \in I} p\left(y_n \mid X_n^{(i)} = j, KF_{n-1}^{(i)}\right) f\left(X_n^{(i)} = j \mid \boldsymbol{X}_{1:n-1}^{(i)}\right)}$$
  - Resampling
    resample a new set of KF_t from $\left\{KF_n^{(1)}, \cdots, KF_n^{(N)}\right\}$ with probability proportional to the weights
  - the posterior distribution of z_n is $p\left(z_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{X}_{1:n}^{(i)}\right) = \mathcal{N}\left(\mu_n\left(\boldsymbol{X}_{1:n}^{(i)}\right), \Sigma_n\left(\boldsymbol{X}_{1:n}^{(i)}\right)\right)$

Example

- In many occasions, the indicator variable X_n takes values in a finite discrete set.

- A simple case is

$$Z_n = AZ_{n-1} + B(X)V_n \quad , \quad V_n \sim i.i.d.\ \mathcal{N}(0,1)$$
$$Y_n = CZ_n + DW_n \quad , \quad W_n \sim i.i.d.\ \mathcal{N}(0,1)$$

where X is a binary indicator with probability

$$X = \begin{cases} 1, & p(X=1) = \theta \\ 2, & p(X=2) = 1-\theta \end{cases}$$

and

$$B(1) = b_1, B(2) = b_2$$

Example

- As discussed earlier, a reasonable importance density for X_n is its predictive distribution

$$q\left(x_n|y_n, X_{n-1}^{(i)}, KF_{n-1}^{(i)}\right) \doteq p\left(x_n|\boldsymbol{X}_{1:n-1}^{(i)}, y_{1:n}\right)$$
$$\propto p\left(y_n|\boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n-1}^{(i)}, x_n\right) p\left(x_n|\boldsymbol{X}_{1:n-1}^{(i)}\right)$$

- We know that $p\left(x_n = j \mid \boldsymbol{X}_{1:n-1}^{(i)}\right) = p\left(x_n = j\right)$ is the prior transition probability for the indicator which is prescribed in the model.

- In this example, we first use Kalman filter to obtain the distributions

$$p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n-1}^{(i)}, x_n = 1\right)$$
$$p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n-1}^{(i)}, x_n = 2\right)$$

- Then we can calculate the probability density of

$$v_i = p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n-1}^{(i)}, x_n = i\right) p\left(x_n = i\right)$$

Example

- By normalizing p_1 and p_2, we obtain:

$$p_1 = \frac{v_1}{v_1 + v_2} \quad , \quad p_2 = \frac{v_2}{v_1 + v_2}$$

In other words, the proposal distribution is

$$q(x_n = 1) = p_1 \quad , \quad q(x_n = 2) = p_2$$

Then we can sample x_n from it.

- The posterior distribution of z_n is directly obtained by Kalman filter

$$p\left(z_n \mid \boldsymbol{y}_{1:n}, X_n^{(i)}\right) = \mathcal{N}\left(\mu_{n|n}, \sigma_{n|n}^2\right)$$

Example

- The algorithm for CLGM with indicator variable in a discrete state space
- At time step *n*, for i=1,2,…,N (index for particles)

  - Updating
    For j=1 and 2, estimate the following distribution using Kalman filter

$$p\left(z_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{X}_{1:n-1}^{(i)}, X_n^{(i)} = j\right) = \mathcal{N}\left(\mu_{n|n}, \sigma_{n|n}^2\right)$$

$$p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n-1}^{(i)}, X_n^{(i)} = j\right) = \mathcal{N}\left(y_{n|n-1}, S_n\right)$$

  - Sampling
    calculate probability densities

$$v_i = p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{X}_{1:n-1}^{(i)}, X_n^{(i)} = j\right) p\left(X_n^{(i)} = j\right)$$

    normalize them to get $p_1 = \dfrac{v_1}{v_1 + v_2}$, $p_2 = \dfrac{v_2}{v_1 + v_2}$

    and sample $X_n^{(i)}$ with probability p_1 and p_2.

Example

Calculate the importance weight

$$w_n^{(i)} = w_{n-1}^{(i)} \times \sum_j v_j^{(i)}$$

- Resampling
  calculate ESS and make resampling when ESS < ESS_{th}.

Estimate the Hidden Markov States

- In fact, we are more interested with estimating the unobserved Markov states z_n with posterior density

$$p\left(z_n \mid \boldsymbol{y}_{1:n}\right)$$

- Since

$$p\left(z_n \mid \boldsymbol{y}_{1:n}\right) = \int p\left(z_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n}\right) p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right) d\boldsymbol{x}_{1:n}$$

where $z_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{x}_{1:n} \sim \mathcal{N}\left(\mu_n\left(\boldsymbol{x}_{1:n}\right), \Sigma_n\left(\boldsymbol{x}_{1:n}\right)\right)$, the main idea in estimating state variables is to use the weighted sample of the indicators x_n.

$$\hat{p}_N(z_n|\boldsymbol{y}_{1:n}) = \sum w_n^{(i)} \mathcal{N}\left(\mu_n\left(\boldsymbol{X}_{1:n}^{(i)}\right), \Sigma_n\left(\boldsymbol{X}_{1:n}^{(i)}\right)\right)$$

here $w_n^{(i)}$ are normalized importance weights.

- Therefore, after sampling the indicators, we can directly use the above approximation to sample z_n.

Example

- Let us return to our earlier example:

$$Z_n = 0.9Z_{n-1} + B(X)V_n \quad , \quad V_n \sim i.i.d.\ \mathcal{N}(0,1)$$
$$Y_n = Z_n + 0.3W_n \quad , \quad W_n \sim i.i.d.\ \mathcal{N}(0,1)$$

where

$$X = \begin{cases} 1, & p(X=1) = 0.7 \\ 2, & p(X=2) = 0.3 \end{cases}$$

and

$$B(1) = 0.5, B(2) = 1.5$$

Example

The implementation of the algorithm is summarized as follows:

- In this example,

$$p\left(x_n = j \mid X_{n-1}^{(i)}\right) = \pi_j \qquad \text{where } \pi_1 = 0.7 \text{ and } \pi_2 = 0.3$$

- Using Kalman filter, $v_j^{(i)}$ can be calculated with the following formulae

$$\mu_+ = A\mu_{n-1}^{(i)} \quad ; \quad \Sigma_+ = A^2\Sigma_{n-1}^{(i)} + \left(B\left(X_{n-1}^{(i)}\right)\right)^2$$

$$v_j^{(i)}\left(X_n^{(i)} = j\right) = \frac{\pi_j}{\sqrt{2\pi}\sqrt{\Sigma_+ + \sigma_w^2}} \exp\left\{-\frac{\left(y_n - \mu_+\right)^2}{2\left(\Sigma_+ + \sigma_w^2\right)}\right\}$$

where $\sigma_w = 0.3$ is the standard deviation in the observation equation.

chapter 11 (A theoretical framework for sequential importance sampling with resampling, J. Liu, R. Chen and T. Logvinenko) in *Doucet A, Freitas N, Gordon N. Sequential Monte Carlo methods in practice. New York: Springer. 2001*

Example

- Sampling Step
  - we sample $X_n^{(i)}$ with probability proportional to $v_j^{(i)}$.

$$v_j^{(i)}\left(X_n^{(i)}=j\right)=\frac{\pi_j}{\sqrt{2\pi}\sqrt{\Sigma_+ +\sigma_w^2}}\exp\left\{-\frac{\left(y_n-\mu_+\right)^2}{2\left(\Sigma_+ +\sigma_w^2\right)}\right\}$$

- The importance weight is calculated by

$$w_n^{(i)}=w_{n-1}^{(i)}\times\sum_j v_j^{(i)}$$

Example

- Updating Step
  - To calculate $KF_n^{(i)} = \left\{ \mu_n^{(i)}\left(\boldsymbol{X}_{1:n}^{(i)}\right), \Sigma_n^{(i)}\left(\boldsymbol{X}_{1:n}^{(i)}\right), X_n^{(i)} \right\}$, we have

$$\Sigma_n^{(i)}\left(X_n^{(i)} = j\right) = \left(\frac{1}{\Sigma_+} + \frac{1}{\sigma_w^2}\right)^{-1}$$

$$\mu_n^{(i)}\left(X_n^{(i)} = j\right) = \left(\frac{\mu_+}{\Sigma_+} + \frac{y_n}{\sigma_w^2}\right)\Sigma_n^{(i)}$$

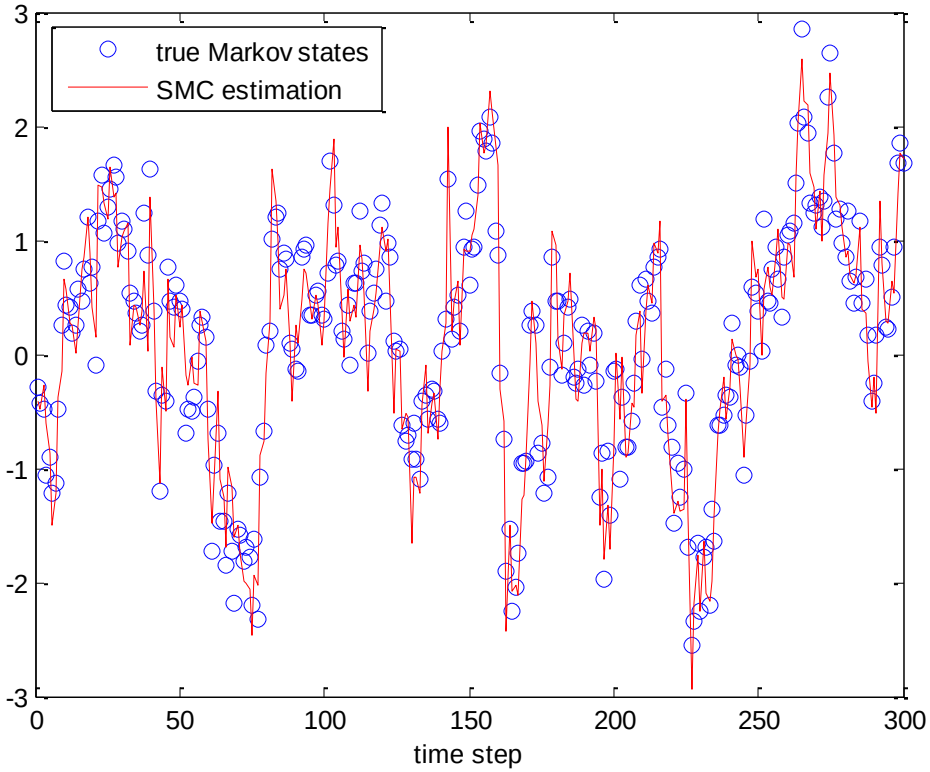
- Resampling
  - calculate ESS and resample a new set of x_n and KF_n with probability proportional to the weights

- The posterior mean of z_n is obtained by Kalman filter when x_n is sampled

$$p\left(z_n \mid \boldsymbol{y}_{1:n}, \boldsymbol{X}_{1:n}^{(i)}\right) = \mathcal{N}\left(\mu_n\left(\boldsymbol{X}_{1:n}^{(i)}\right), \Sigma_n\left(\boldsymbol{X}_{1:n}^{(i)}\right)\right)$$

Example

- The number of time steps is set to be 300 and the ESS_{th} = 240.
- 500 particles are used to estimate the state variable z_n.
- Through SMC, we can estimate the state variable accurately, even though there are several sudden jumps occurring in the system.
- The figure gives the posterior mean of the z_n.



A MatLab implementation can be found here

RAO-BLACKWELLISED PARTICLE FILTER (RBPF)

RBPF

- In some models we can partition the hidden variables x_t and z_t such that we can analytically integrate out z_t provided we know the values of $\mathbf{x}_{1:t}$.
- Thus we only have to sample $\mathbf{x}_{1:t}$ and can represent $p(z_t|\mathbf{x}_{1:t})$ parametrically. Each particle *s now* represents a value for $\boldsymbol{x}_{1:t}^{s}$ and a distribution of the form $p(z_t|\boldsymbol{y}_{1:t}, \boldsymbol{x}_{1:t}^{s})$.
- These hybrid particles are called **distributional particles** or **collapsed particles.**
- The advantage of this approach is that we reduce the dimensionality of the space in which we are sampling, which reduces the variance of our estimate.
- This technique is known as **Rao-Blackwellised particle filtering** or **RBPF.**

- Koller, D. and N. Friedman (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

RBPF for Switching LG-SSMs

- As an example of the RBPF we consider an application to the switching linear dynamical system (SLDS).

- We can represent $p(z_t|\boldsymbol{y}_{1:t}, \boldsymbol{x}^s_{1:t})$ using a mean and covariance matrix for each particle s, where here $x_t \in \{1, \ldots, K\}$.

- If we propose from the prior $q(x_t = k|x^s_{t-1})$ the weight update becomes

$$w^s_t \propto w^s_{t-1} p(y_t|x_t = k, \boldsymbol{x}^s_{1:t-1}, \boldsymbol{y}_{1:t-1}) = w^s_{t-1} L^s_{t,k}$$

$$L^s_{t,k} = \int p(y_t|x_t = k, z_t, \boldsymbol{y}_{1:t-1}, \boldsymbol{x}^s_{1:t-1}) p(z_t|x_t = k, \boldsymbol{y}_{1:t-1}, \boldsymbol{x}_{1:t-1}, \boldsymbol{x}^s_{1:t-1}) dz_t$$

- The $L^s_{t,k}$ is the predictive density for the new observation y_t conditioned on $x_t = k$ and $\boldsymbol{x}^s_{1:t-1}$. In SLDS models, this can be computed using the normalization constant of the Kalman filter $p(y_t|\boldsymbol{y}_{1;t-1}, \boldsymbol{x}_{1:t}) = \mathcal{N}(y_t|\boldsymbol{C}_t \boldsymbol{\mu}_{t|t-1}, \boldsymbol{S}_t)$.

- Chen, R. and S. Liu (2000). Mixture Kalman filters. *J. Royal Stat. Soc. B*.
- Doucet, A., N. de Freitas, and N. J. Gordon (2001). *Sequential Monte Carlo Methods in Practice*. Springer Verlag.

RBPF for SLDS (Mixture of Kalman Filters)

$for\ s = 1{:}S$

$\quad k \sim p(x_t | x_{t-1}^s);$

$\quad x_t^s := k;$

$\quad \left(\boldsymbol{\mu}_t^s, \boldsymbol{\Sigma}_t^s, L_{t,k}^s\right) = KFUpdate(\boldsymbol{\mu}_{t-1}^s, \boldsymbol{\Sigma}_{t-1}^s, y_t, \boldsymbol{\theta}_k);$ // One for each particle

$\quad w_t^s = w_{t-1}^s L_{tk}^s;$

$Normalize\ weights{:}\ w_t^s = \dfrac{w_t^s}{\sum_{s'} w_t^{s'}};$

$Compute\ \hat{S}_{eff} = \dfrac{1}{\sum_{s=1}^{S} (w_t^s)^2};$

$If\ \hat{S}_{eff} < S_{min}\ then$

$\quad Resample\ S\ indices\ \boldsymbol{\pi} \sim \boldsymbol{w}_t;$

$\quad x_t^s = x_t^{\pi}, \boldsymbol{\mu}_t^s = \boldsymbol{\mu}_t^{\pi}, \boldsymbol{\Sigma}_t^s = \boldsymbol{\Sigma}_t^{\pi};$

$\quad w_t^s = {}^1/_S;$

RBPF for Switching LG-SSMs

- If K is small, we can compute the optimal proposal distribution as

$$p(x_t = k|\boldsymbol{x}_{1:t-1}^s, \boldsymbol{y}_{1:t}) = \hat{p}_{t-1}^s(x_t = k|y_t) = \frac{\hat{p}_{t-1}^s(y_t|x_t = k)\hat{p}_{t-1}^s(x_t = k)}{\hat{p}_{t-1}^s(y_t)}$$

$$= \frac{L_{tk}^s p(x_t = k|x_{t-1}^s)}{\sum_{k'} L_{tk'}^s p(x_t = k'|x_{t-1}^s)}$$

- Here we have denoted $\hat{p}_{t-1}^s(\cdot) = p(\cdot|\boldsymbol{x}_{1:t-1}^s, \boldsymbol{y}_{1:t-1})$
- We sample from $p(q_t = k|\boldsymbol{x}_{1:t-1}^s, \boldsymbol{y}_{1:t})$ and assign the resulting particle the following weight

$$w_t^s \propto w_{t-1}^s p(y_t|\boldsymbol{x}_{1:t-1}^s, \boldsymbol{y}_{1:t-1}) = w_{t-1}^s \sum_k [L_{tk}^s p(x_t = k|x_{t-1}^s)]$$

- Since the weights of the particles are independent of the new value that is actually sampled for x_t we can compute these weights first, and use them to decide which particles to propagate.

- de Freitas, N., R. Dearden, F. Hutter, R. Morales-Menendez, J. Mutch, and D. Poole (2004). Diagnosis by a waiter and a mars explorer. *Proc. IEEE 92*(3)..
- Fearnhead, P. (2004). Exact Bayesian curve fitting and signal segmentation. *IEEE Trans. Signal Processing 53*, 2160–2166.

RBPF for Switching LG-SSMs

$$w_t^s \propto w_{t-1}^s p(y_t|\boldsymbol{x}_{1:t-1}^s, \boldsymbol{y}_{1:t-1}) = w_{t-1}^s \sum_k [L_{tk}^s p(x_t = k|x_{t-1}^s)]$$

- We choose the fittest particles at $t-1$ using information y_t from time t.
- We pass each sample in the prior through all K models to get K posteriors one per sample. The normalization constants allow us to compute the optimal weights. We then resample S indices.
- Finally, for each old particle s that is chosen, we sample one new state $x_t^s = k$, and use the corresponding posterior from the K possible alternative that we have already computed. This method needs $\mathcal{O}(KS)$ storage, but has the advantage that each particle is chosen using the latest information y_t.
- A further improvement can be obtained by exploiting the fact that the state space is discrete. The resampling method of (Fearnhead 2004) avoids duplicating particles.

One Step of Look-Ahead RBPF for SLDS

$for\ s = 1{:}S$

$\quad For\ k = 1{:}K\ do;$

$\quad\quad \left[\left(\boldsymbol{\mu}_t^s, \boldsymbol{\Sigma}_t^s, L_{t,k}^s\right) = KFUpdate(\boldsymbol{\mu}_{t-1}^s, \boldsymbol{\Sigma}_{t-1}^s, y_t, \boldsymbol{\theta}_k);\right.$

$\quad w_t^s = w_{t-1}^s \sum_k [L_{tk}^s p(x_t = k|x_{t-1}^s)]$

$Normalize\ weights{:}\ w_t^s = \dfrac{w_t^s}{\sum_{s'} w_t^{s'}};$

$Resample\ S\ indices\ \boldsymbol{\pi} \sim \boldsymbol{w}_t$

$For\ s \in \pi\ do$

$\quad Compute\ optimal\ proposal\ p(k|\boldsymbol{x}_{1:t-1}^s, \boldsymbol{y}_{1:t-1}) = \dfrac{L_{tk}^s p(x_t = k|x_{t-1}^s)}{\sum_{k'} L_{tk'}^s p(x_t = k'|x_{t-1}^s)};$

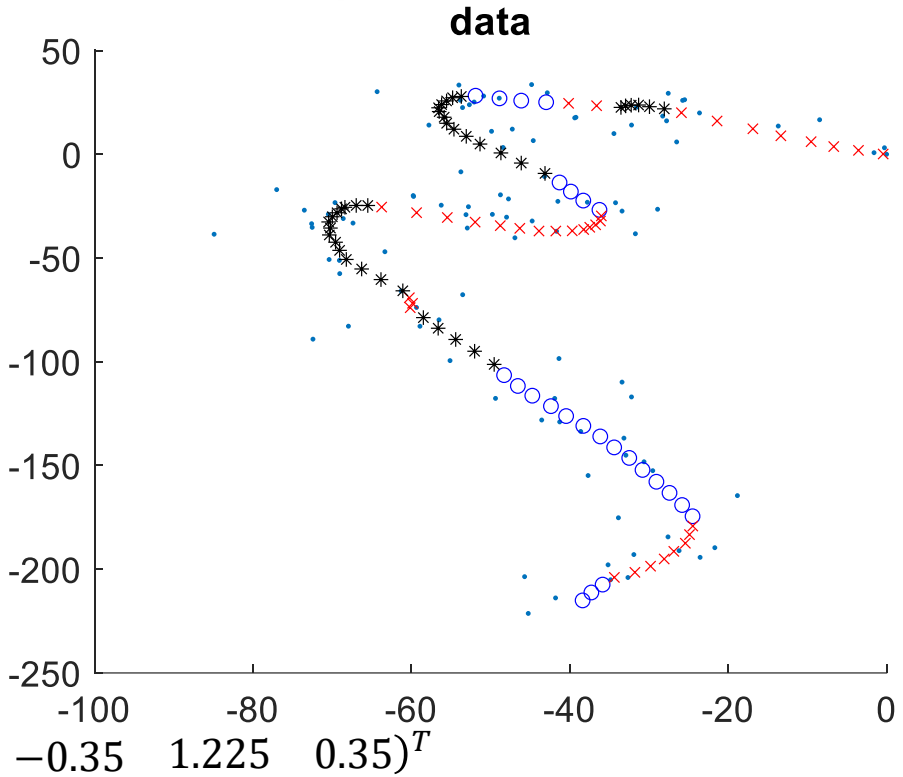
$\quad Sample\ k \sim p(k|\boldsymbol{y}_{1:t}, \boldsymbol{x}_{1:t-1})$

$\quad x_t^s = q_t^\pi, \boldsymbol{\mu}_t^s = \boldsymbol{\mu}_{tk}^s, \boldsymbol{\Sigma}_t^s = \boldsymbol{\Sigma}_{tk}^s;$

$\quad w_t^s = {}^1/_S;$

Tracking of an Object

- One application of SLDS is to track moving objects that have piecewise linear dynamics.
- q_t can specify if the object is flying normally or is taking evasive action (**maneuvering target tracking**)
- In this example we add a three-state discrete Markov chain which controls the input of the system.



$u_t = 1, \boldsymbol{B}_1 = (0 \quad 0 \quad 0 \quad 0)^T, \boldsymbol{B}_2 = (-1.225 \quad -0.35 \quad 1.225 \quad 0.35)^T$

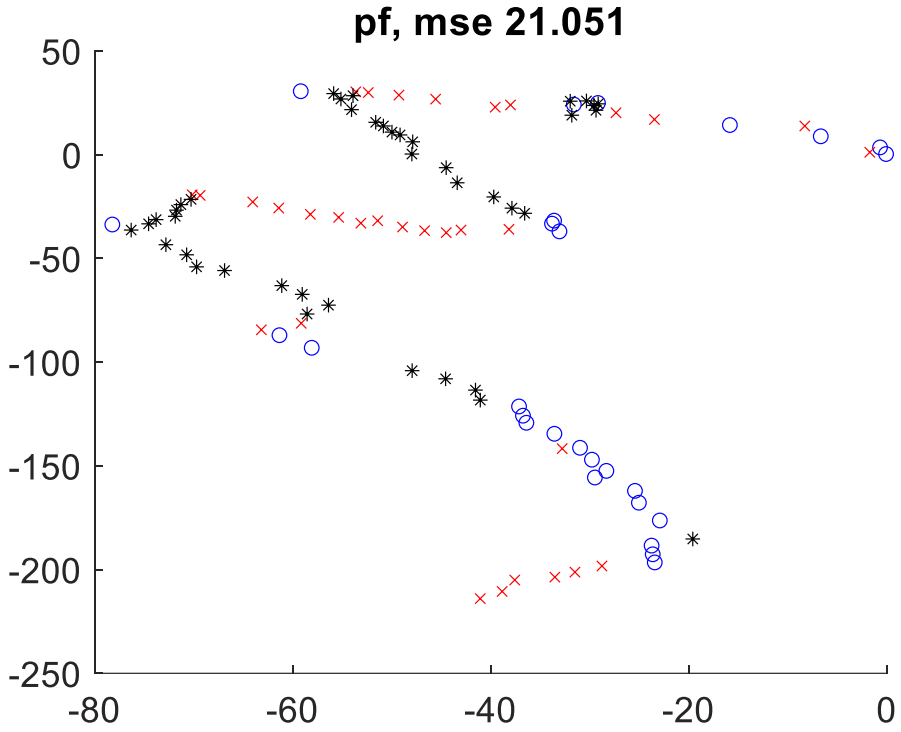
$\boldsymbol{B}_3 = (1.225 \quad 0.35 \quad -1.225 \quad -0.35)^T$

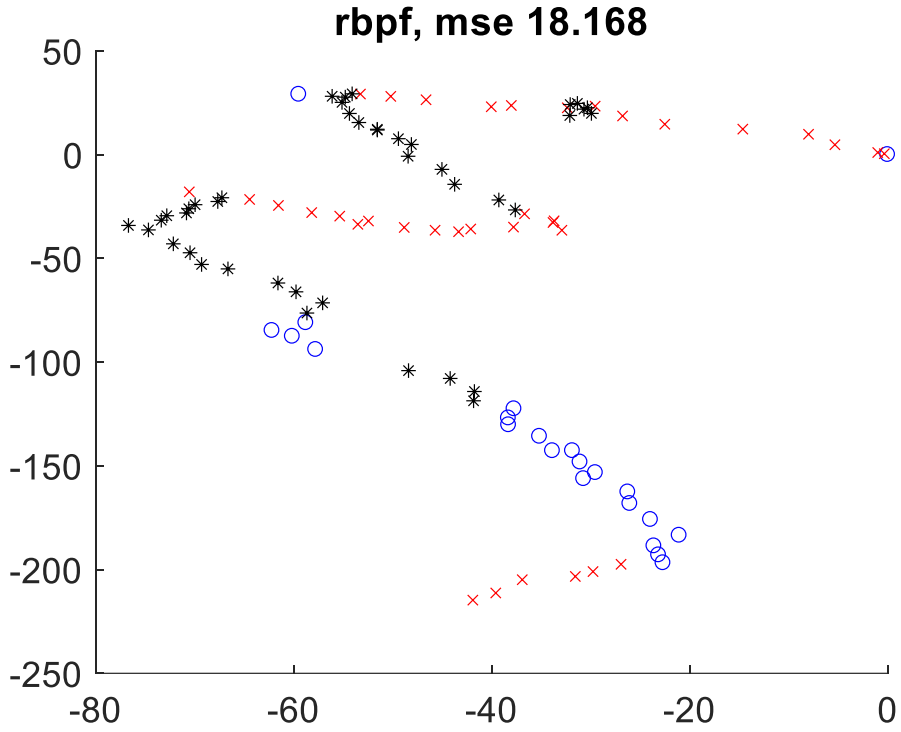
$$p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}, x_k = k, \boldsymbol{u}_t, \theta) = \mathcal{N}(\boldsymbol{z}_t|\boldsymbol{A}_k\boldsymbol{z}_{t-1} + \boldsymbol{B}_k\boldsymbol{u}_k, \boldsymbol{Q}_k)$$

Run rbpfManueuverDemo

- The system turns in different directions depending on the discrete state.
- The Fig shows the true state from a sample run starting at (0,0). colored symbols denote the discrete state, and the location of the symbol denotes the (*x*, *y*) location. Small dots represent noisy observations

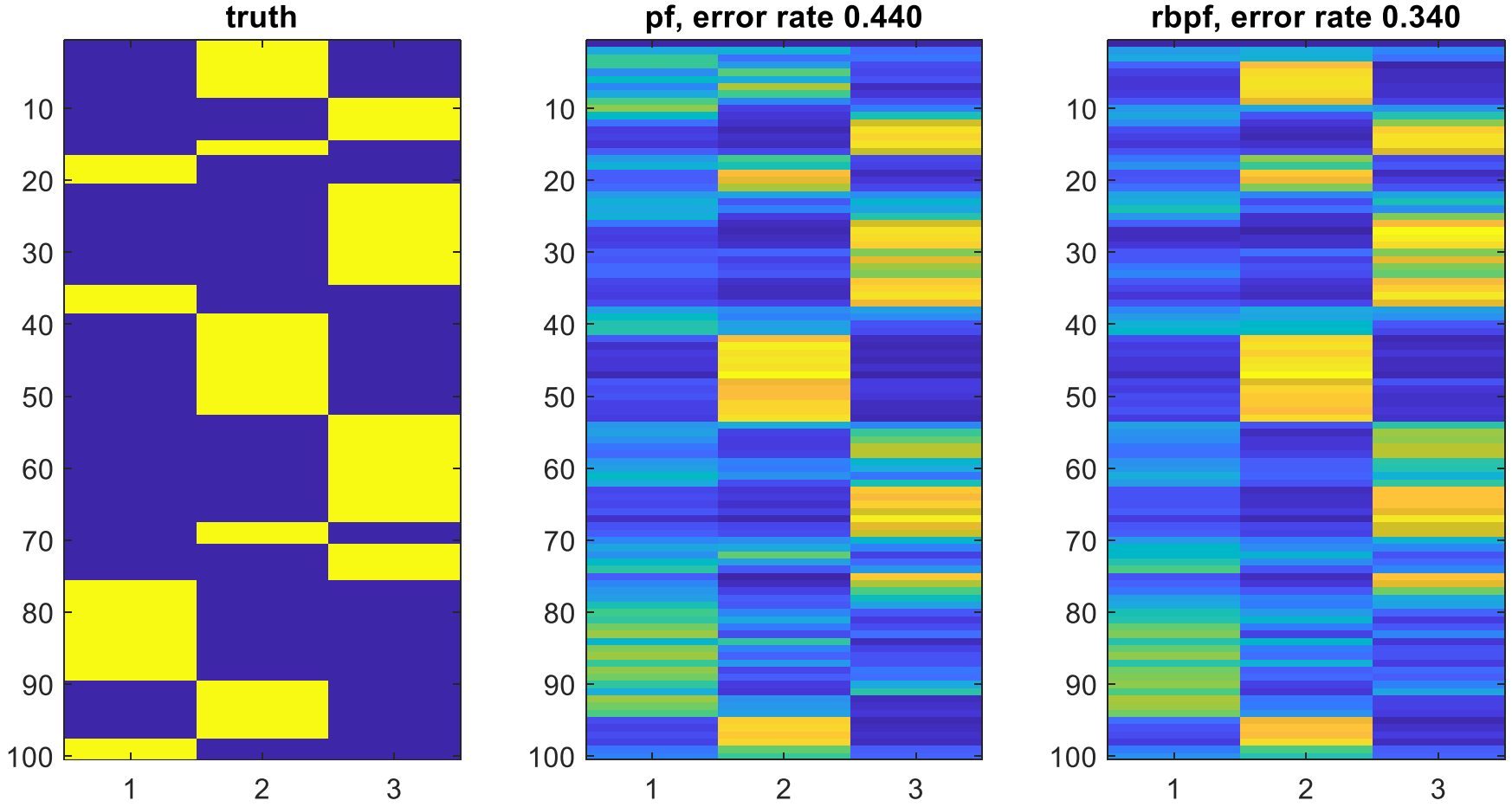
Tracking of an Object





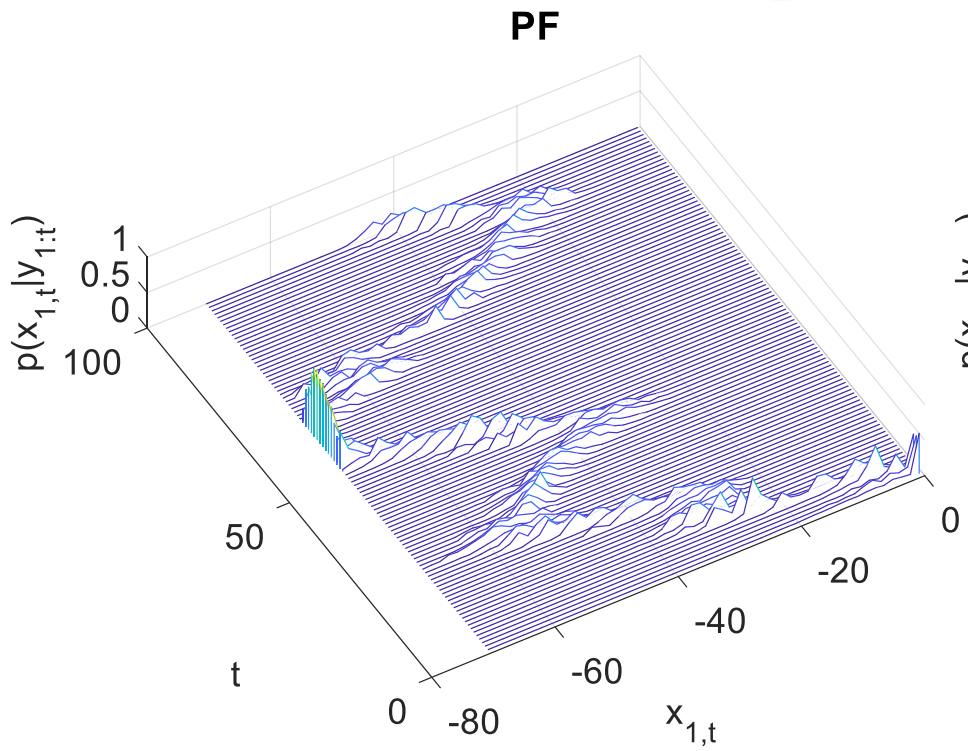
- (Left) shows the state computed using PF (500 particles) where the proposal is to sample from the prior. The colored symbols denote the MAP of the state and the location of the symbol denotes the MMSE (minimum mean square error) estimate (posterior mean).
- (Right) shows the estimate computing using RBPF with 500 particles, using the optimal proposal distribution. RBPF has slightly better performance, although it is also slightly slower.

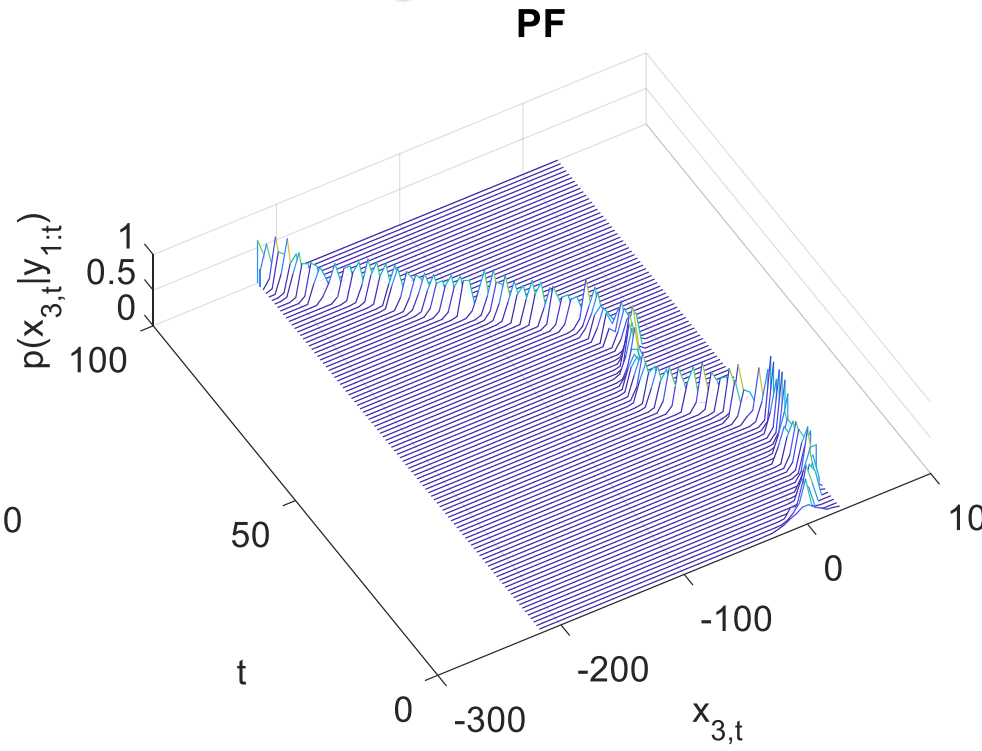
Distribution over Discrete States



- The particle filter estimate of the belief state (2nd column) is not as accurate as the RBPF estimate (3nd column) in the beginning, although after the first few observations performance is similar.

Tracking of an Object

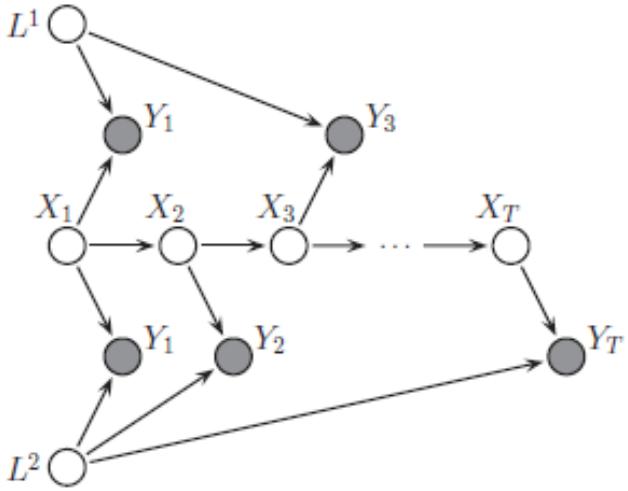




- The posterior over the locations. For simplicity, we use the PF estimate, which is a set of weighted samples, but we could also have used the RBPF estimate, which is a set of weighted Gaussians.

Fast Slam

- In the problem of simultaneous localization and mapping or SLAM for mobile robotics, the main problem with the Kalman filter implementation is that it is cubic in the number of landmarks.



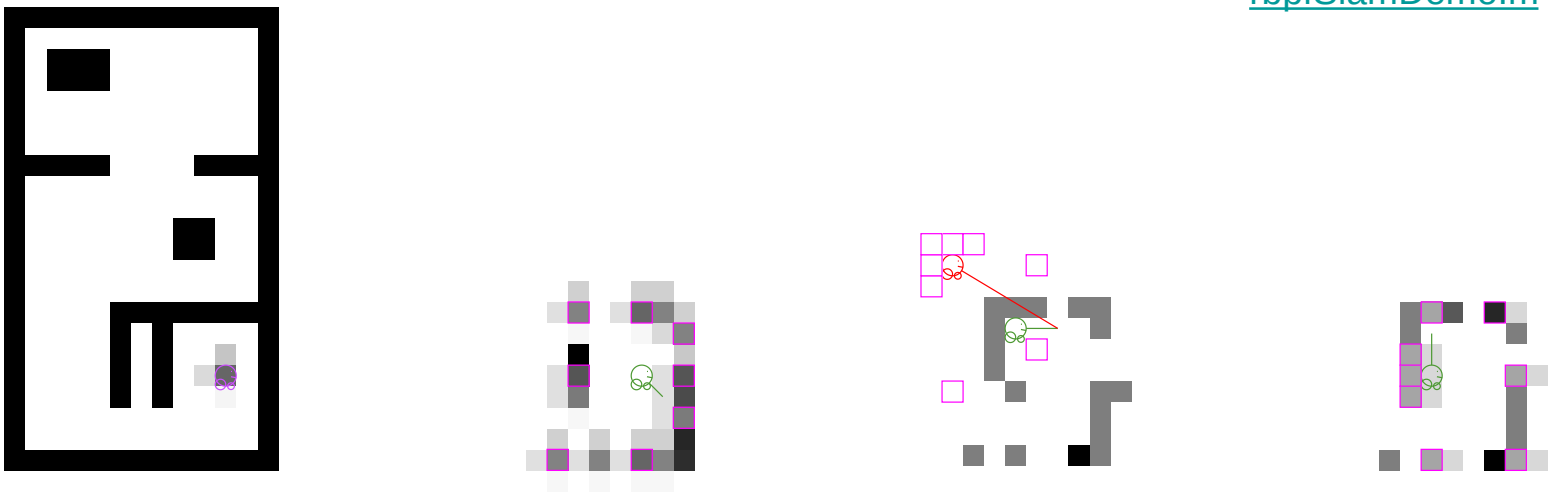
- Conditional on knowing the robot's path, $\boldsymbol{q}_{1:t}$ where $\boldsymbol{q}_t \in \mathbb{R}^2$, the landmark locations $\mathbf{z} \in \mathbb{R}^{2L}$ are independent (the landmarks don't move). That is, $p(\boldsymbol{z}|\boldsymbol{q}_{1:t}, \boldsymbol{y}_{1:t}) = \prod_{l=1}^{L}(z_l|\boldsymbol{q}_{1:t}, \boldsymbol{y}_{1:t})$
- Consequently we can use RBPF, where we sample the robot's trajectory $\boldsymbol{q}_{1:t}$ and we run L independent 2d Kalman filters inside each particle. This takes $\mathcal{O}(L)$ time per particle.
- The number of particles needed for good performance is quite small so the algorithm is essentially linear in the number of particles.

Fast Slam

- This technique has the additional advantage that it is easy to use sampling to handle the data association ambiguity, and that it allows for other representations of the map, such as occupancy grids.
- This idea first discussed in Murphy 2000 and was subsequently extended and made practical in (Thrun et al. 2004).

What is out there **What the robot sees** **What the robot sees** **What the robot sees**

rbpfSlamDemo.m



- Murphy, K. (2000). Bayesian map learning in dynamic environments. In *NIPS*, Volume 12.
- Thrun, S., M. Montemerlo, D. Koller, B. Wegbreit, J. Nieto, and E. Nebot (2004). Fastslam: An efficient solution to the simultaneous localization and mapping problem with unknown data association. *J. of Machine Learning Research 2004*.

SMC for Time Series Models

Time Series Models

- The framework developed for conditionally linear dynamic models can also be applied to **Time Series Models**.

- In statistics, a `time series' is a sequence of data points, measured typically at successive times spaced at uniform time intervals.

- Models for time series data can have many forms and represent different stochastic processes. The most common (linear) time series models include

  - Autoregressive models (AR models)

  - Moving Average models (MV models)

    and the combination of the above two models :

  - Autoregressive Moving Average models (ARMV models)

- A.Doucet,P.Duvaut. Bayesian estimation of state-space models applied to deconvolution of Bernoulli-Gaussian process. Signal Processing. 1997,57:p147

AR Models

- An AR(1) process $\{x_n\}$ is defined by the conditional relation

$$x_n = \mu + \rho\left(x_{n-1} - \mu\right) + \varepsilon_n$$

where ε_n is an iid sequence of random variables with mean 0 and variance σ^2 (white noise sequence).

- A generalization of the AR(1) model is obtained by increasing the lag dependence on the past states.

- An AR(p) process is defined by the conditional (against the past) representation

$$x_n = \mu + \sum_{i=1}^{p} \rho_i \left(x_{n-i} - \mu\right) + \varepsilon_n$$

- This natural generalization assumes that the past p values of the process influence the current state of the process.

MA Models

- A second type of time series model that still enjoys linear dependence and closed-form expression is the MA(q) model.

- An MA(1) process is such that, conditionally on the past

$$x_n = \mu + \varepsilon_n - \upsilon \varepsilon_{n-1}$$

where ε_n is an iid sequence of random variables with mean 0 and variance σ^2 (white noise sequence).

- The mean and variance of the MA(1) model are

$$\mathbb{E}(x_n) = \mu \ , \ Var(x_n) = \left(1+\upsilon^2\right)\sigma^2$$

- A natural extension of the MA(1) model is to increase the dependence on the past innovations, namely to introduce the MA(q) process defined by

$$x_n = \mu + \varepsilon_n - \sum_{i=1}^{q} \upsilon_i \varepsilon_{n-i}$$

ARMA Models

- A straightforward extension of both previous models are the ARMA(p,q) models, where the states $\{x_n\}$ are conditionally defined as

$$x_n = \mu + \sum_{i=1}^{p} \rho_i \left(x_{n-i} - \mu \right) + \varepsilon_n - \sum_{j=1}^{q} \upsilon_j \varepsilon_{n-j}$$

where the noise $\{\varepsilon_n\}$ are independent.

- Example : ARMA(2,1) model
  - the state variable z_n is conditionally defined as

$$z_k = a_1 z_{k-1} + a_2 z_{k-2} + v'$$

where the noise is a two-component normal mixture

$$v' \sim \lambda \mathcal{N}\left(0, \sigma_0^2\right) + \left(1-\lambda\right) \mathcal{N}\left(0, \alpha^2 \sigma_0^2\right) \quad , 0 < \lambda < 1$$

  - the observation equation

$$y_k = z_k + w \qquad w \sim i.i.d.\ \mathcal{N}\left(0, \sigma_w^2\right)$$

ARMA Models

- Let us simulate the ARMA(2,1) model with the SMC method
- First, we can consider the variance v' in the state equation to be generated from $\mathcal{N}\left(0,\sigma_0^2\right)$ with probability λ or from $\mathcal{N}\left(0,\alpha^2\sigma_0^2\right)$ with probability 1-λ.
- In other words, consider an auxiliary random binary variable u
- Assume

$$\Pr(u=1)=\lambda$$
$$\Pr(u=2)=1-\lambda$$
$$v' \sim \mathcal{N}\left(0,\sigma_v^2\right)$$

- At each time step, we make sample from *u*, thus

$$\sigma_v = \begin{cases} \sigma_0 \text{ , when } u=1 \\ \alpha\sigma_0 \text{, when } u=2 \end{cases}$$

ARMA Models

- Transform the ARMA model into a conditionally linear dynamic model
- Let us set $x_k = \begin{pmatrix} z_k \\ z_{k-1} \end{pmatrix}$, the state equation can be rewritten as

$$x_k = \begin{bmatrix} a_1 & a_2 \\ 1 & 0 \end{bmatrix} x_{k-1} + \begin{pmatrix} \sigma_v \\ 0 \end{pmatrix} v$$

where $v \sim i.i.d.\ \mathcal{N}(0,1)$ and σ_v is controlled by a random indicator variable u={1,2}

$$\sigma_v = \begin{cases} \sigma_0 \text{ , when } u = 1 \\ \alpha\sigma_0, \text{ when } u = 2 \end{cases}$$

$$\Pr(u=1) = \lambda \ , \ \Pr(u=2) = 1-\lambda$$

- The observation equation

$$y_k = \begin{pmatrix} 1 & 0 \end{pmatrix}^T x_k + w \qquad w \sim i.i.d.\ \mathcal{N}\left(0, \sigma_w^2\right)$$

Example

- Given coefficients, we have

$$x_k = \begin{bmatrix} 1.51 & -0.55 \\ 1 & 0 \end{bmatrix} x_{k-1} + \begin{pmatrix} \sigma_v \\ 0 \end{pmatrix} v$$

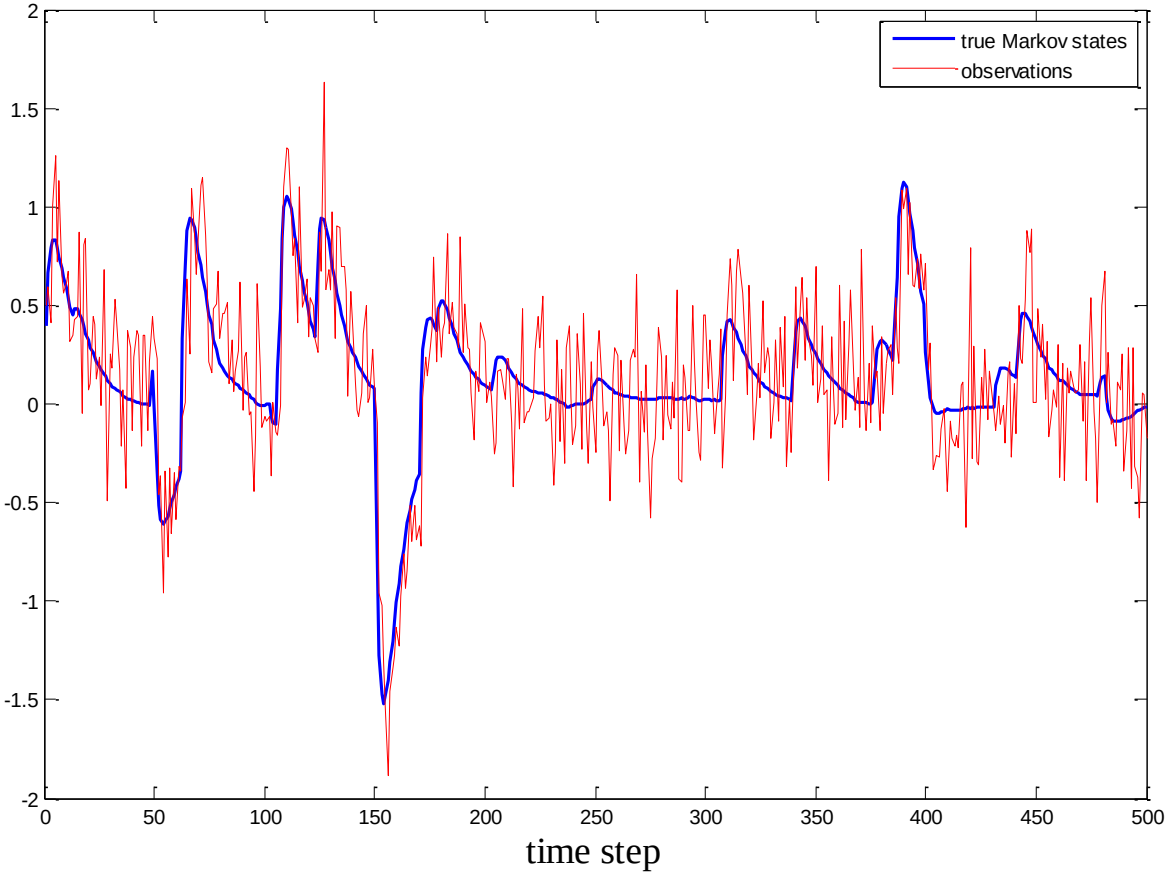
$$\sigma_v = \begin{cases} 0.3 \text{ , when } u = 1 \\ 0.003, \text{ when } u = 2 \end{cases}$$

$$\Pr(u = 1) = 0.05 \ , \ \Pr(u = 2) = 0.95$$

$$y_k = \begin{pmatrix} 1 & 0 \end{pmatrix} x_k + w \quad w \sim i.i.d. \ \mathcal{N}\left(0, 0.25^2\right)$$

Example

- From the above equations, a 500-time-steps realization is generated.
- The red line represents the observations; while the blue line denotes the true Markov states.



See here for a MatLab implementation

SMC For the ARMA Model

- Now we can use the SMC framework for conditionally linear dynamic model to simulate the ARMA model.

- The algorithm is similar with the one for the previous example, except the expressions for mean and covariance should be transformed into matrix form.

- Suppose at time *n*, we know .

$$KF_{n-1}^{(i)} = \left\{ \mu_{n-1}^{(i)}\left(\boldsymbol{X}_{1:n-1}\right), \Sigma_{n-1}^{(i)}\left(\boldsymbol{X}_{1:n-1}\right), \boldsymbol{X}_{1:n-1}^{(i)} \right\}$$

- First, let us estimate

$$v_j^{(i)} \propto p\left(y_n \mid KF_{n-1}^{(i)}, x_n = i\right) p\left(x_n = j \mid X_{n-1}^{(i)}\right)$$

based on which we make samples for x_n and compute the importance weights. In this example,

$$p\left(x_n = j \mid X_{n-1}^{(i)}\right) = \pi_j \qquad \text{where } \pi_1 = 0.05 \text{ and } \pi_2 = 0.95$$

Example

- Here $p\left(y_n \mid KF_{n-1}^{(i)}, x_n = j\right)$ is a byproduct of the Kalman filter

$$p\left(y_n \mid KF_{n-1}^{(i)}, x_n = j\right) = p\left(y_n \mid \boldsymbol{y}_{1:n-1}, \boldsymbol{x}_{1:n}\right) \sim \mathcal{N}\left(y_{n|n-1}, S_n\right)$$

  Sampling Step

  - we sample $x_n^{(i)}$ with probability proportional to $v_j^{(i)}$.

- The importance weight is calculated by

$$w_n^{(i)} = w_{n-1}^{(i)} \times \sum_j v_j^{(i)}$$

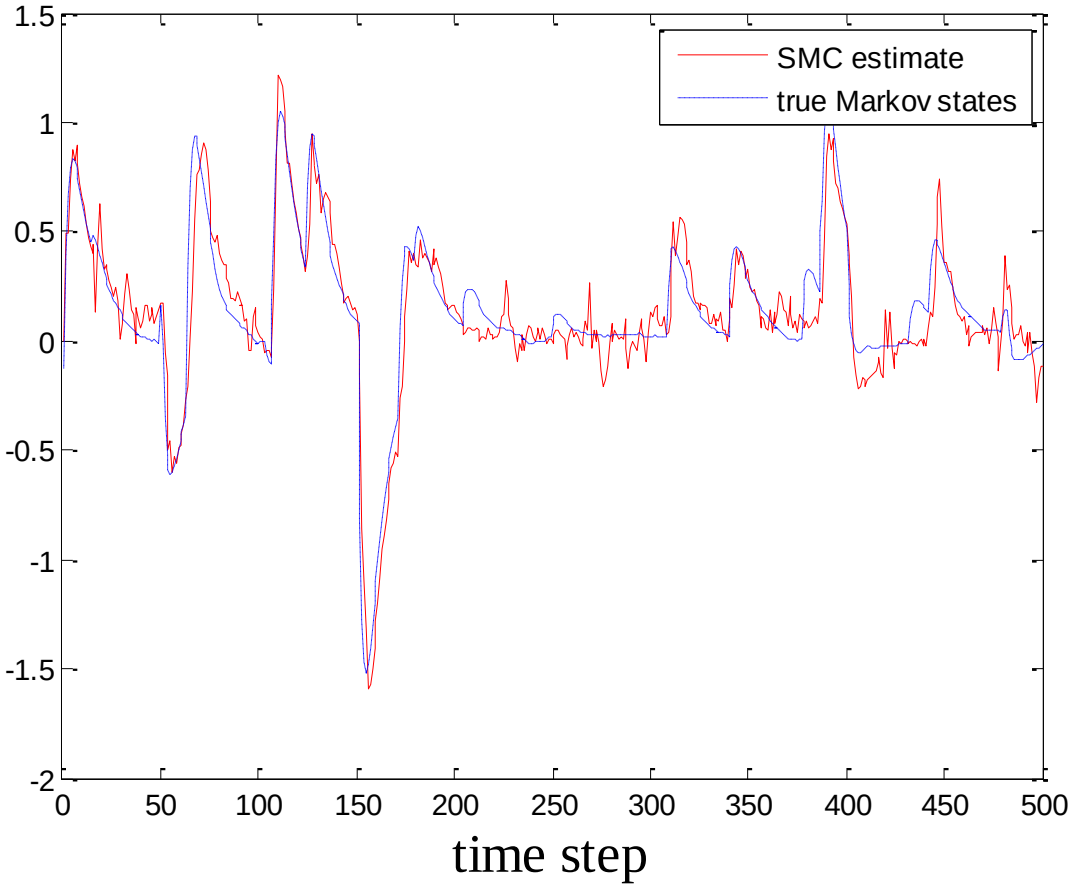
- Updating from $KF_{n-1}^{(i)} \rightarrow KF_n^{(i)}$

- Resampling

  - resample a new set of KF_n from $\left\{KF_n^{(1)}, \cdots, KF_n^{(N)}\right\}$ with probability proportional to the weights

Example

- We use 2000 particles to estimate the hidden Markov states $x_{1:n}$.



Partially Observed Linear Gaussian Models

Partially Observed Linear Gaussian Models

- Many data analysis tasks involve estimating the state of a dynamic model when only partial observations are available.

- Except in a few special cases, including linear Gaussian state space models, on-line state estimation is a problem that does not admit a closed form solution.

- In this section, we apply the SMC methods into partially observed linear Gaussian state-space models.

Unobserved processes

Observations

$$
\begin{aligned}
Z_n &= AZ_{n-1} + BV_n \quad , \quad V_n \sim i.i.d.\ \mathcal{N}(0, I) \\
X_n &= CZ_n + DW_n \quad , \quad W_n \sim i.i.d.\ \mathcal{N}(0, I) \\
Y_n &\mid (Z_n, X_n) \sim g(\cdot \mid X_n)
\end{aligned}
$$

- We want to estimate *sequentially in time* some characteristics of the posterior distribution $p(\boldsymbol{x}_{1:n}, \boldsymbol{z}_{1:n} \mid \boldsymbol{y}_{1:n})$

Partially Observed Linear Gaussian Models

- In this model, $\boldsymbol{x}_{1:n}$ and $\boldsymbol{z}_{1:n}$ are unknown variables. Given observation data $\boldsymbol{y}_{1:n}$, the posterior distribution

$$p\left(\boldsymbol{x}_{1:n}, \boldsymbol{z}_{1:n} \mid \boldsymbol{y}_{1:n}\right) = p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right) \underbrace{p\left(\boldsymbol{z}_{1:n} \mid \boldsymbol{x}_{1:n}\right)}_{Gaussian}$$

- Since

$$\begin{aligned} Z_n &= AZ_{n-1} + BV_n \quad , \quad V_n \sim i.i.d.\ \mathcal{N}(0, I) \\ X_n &= CZ_n + DW_n \quad , \quad W_n \sim i.i.d.\ \mathcal{N}(0, I) \\ Y_n &\mid (Z_n, X_n) \sim g(\cdot \mid X_n) \end{aligned}$$

  is a standard linear Gaussian model, $p\left(\boldsymbol{z}_{1:n} \mid \boldsymbol{x}_{1:n}\right)$ is a Gaussian distribution. Its mean and covariance matrix can be estimated analytically by the Kalman filter.

- Thus the task is to estimate the posterior $p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right)$.

$$p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right) \propto \underbrace{p\left(\boldsymbol{y}_{1:n} \mid \boldsymbol{x}_{1:n}\right)}_{Likelihood} \quad \underbrace{p\left(\boldsymbol{x}_{1:n}\right)}_{Prior\ Computed\ by\ Kalman}$$

- We will perform SMC on $\pi_n\left(\boldsymbol{x}_{1:n}\right) = p\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right)$

Partially Observed Linear Gaussian Models

- At time *n-1*, assume we have *N* particles $\left\{\boldsymbol{X}_{1:n-1}^{(i)}\right\}_{i=1}^{N}$ distributed according to $p\left(\boldsymbol{x}_{1:n-1} \mid \boldsymbol{y}_{1:n-1}\right)$.

- At time *n*, we extend each particle $\boldsymbol{X}_{1:n-1}^{(i)}$ by sampling $X_n^{(i)}$ according to a proposal distribution (importance distribution)

$$q_n\left(X_n | \boldsymbol{X}_{1:n-1}^{(i)}, \boldsymbol{y}_{1:n}\right)$$

- Thus, each particle $\widetilde{\boldsymbol{X}}_{1:n}^{(i)}$ is distributed according to

$$p(\boldsymbol{x}_{1:n-1}|\boldsymbol{y}_{1:n-1}) q_n\left(x_n | \boldsymbol{X}_{1:n-1}^{(i)}, \boldsymbol{y}_{1:n}\right)$$

The importance weight

$$W_n^{(i)} = \frac{p(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n})}{p(\boldsymbol{x}_{1:n-1}|\boldsymbol{y}_{1:n-1}) q_n\left(x_n | \boldsymbol{X}_{1:n-1}^{(i)}, \boldsymbol{y}_{1:n}\right)} \propto W_{n-1}^{(i)} \frac{p\left(X_n^{(i)} | \boldsymbol{X}_{1:n-1}^{(i)}\right) g\left(y_n | X_n^{(i)}\right)}{q_n\left(X_n^{(i)} | \boldsymbol{X}_{1:n-1}^{(i)}, \boldsymbol{y}_{1:n}\right)}$$

- If variance of the weights $\left\{W_n^{(i)}\right\}$ is high, resample $\left\{W_n^{(i)}, \boldsymbol{X}_{1:n}^{(i)}\right\}$ to obtain $\left\{1/N, \boldsymbol{X}_{1:n}^{(i)}\right\}$

Revised Implementation of the SISR

- Consider time n-1 when we have $\left\{ \boldsymbol{X}_{1:n-1}^{(i)}, W_{n-1}^{(i)} \right\}$

  At time step n, for i=1,2,…,N (index for particles)

  - Sampling:
    sample $X_n^{(i)}$ from a proposal distribution $q\left(. \mid y_{1:n}, X_{1:n-1}^{(i)}\right)$

  - Updating:
    set $\boldsymbol{X}_{1:n}^{(i)} = \left\{ \boldsymbol{X}_{1:n-1}^{(i)}, X_n^{(i)} \right\}$. The importance weight is

$$W_n^{(i)} \propto W_{n-1}^{(i)} \frac{g\left(y_n \mid X_n^{(i)}\right) p\left(X_n^{(i)} \mid \boldsymbol{X}_{1:n-1}^{(i)}\right)}{q_n\left(X_n^{(i)} \mid y_n, \boldsymbol{X}_{1:n-1}^{(i)}\right)}$$

  - Resampling:
    If variance of weights is high, resample $\left\{ \boldsymbol{X}_{1:n}^{(i)}, W_n^{(i)} \right\}$ to obtain a new population of particles and weights $\left\{ \boldsymbol{X}_{1:n}^{(i)}, W_n^{(i)} = 1/N \right\}$

Partially Observed Linear Gaussian Models

- The optimal proposal distribution is

$$q_n\left(X_n|\boldsymbol{X}_{1:n-1}^{(i)},\boldsymbol{y}_{1:n}\right)=p\left(X_n|\boldsymbol{X}_{1:n-1}^{(i)},\boldsymbol{y}_{1:n}\right)\propto p\left(X_n^{(i)}|\boldsymbol{X}_{1:n-1}^{(i)}\right)g\left(y_n|X_n^{(i)}\right)$$

- The cost looks to increase with n even if you focus on $p\left(x_n \mid \boldsymbol{y}_{1:n}\right)$

- As we discussed earlier for conditional Gaussian models, the key is to observe that:

$$p(x_n|\boldsymbol{x}_{1:n-1})=\int p(x_n|z_n)f(z_n|z_{n-1})p(z_{n-1}|\boldsymbol{x}_{1:n-1})d\boldsymbol{z}_{n-1:n}$$

- $p(z_{n-1}|\boldsymbol{x}_{1:n-1})$ is a Gaussian distribution with mean m_{n-1} and covariance Σ_{n-1} computed from the Kalman filter associated to the "virtual" observations.

- We don't need to store $\boldsymbol{X}_{1:n}^{(i)}$ but only the sufficient statistics

$$\left(\boldsymbol{X}_n^{(i)},\mu_n\left(\boldsymbol{X}_{1:n}^{(i)}\right),\Sigma_n\left(\boldsymbol{X}_{1:n}^{(i)}\right)\right)=\left(\boldsymbol{X}_n^{(i)},\boldsymbol{\mu}_n^{(i)},\boldsymbol{\Sigma}_n^{(i)}\right)$$

Revised Implementation of the SISR

- Consider time n-1 when we have $\left\{X_{n-1}^{(i)}, \mu_{n-1}^{(i)}, \Sigma_{n-1}^{(i)} = \Sigma_{n-1}, W_{n-1}^{(i)}\right\}$

  At time step n, for i=1,2,…,N (index for particles)

  - Sampling:
    sample $X_n^{(i)}$ from a proposal distribution $q\left(. \mid y_n, X_{n-1}^{(i)}, \mu_{n-1}^{(i)}, \Sigma_{n-1}\right)$

  - Updating:
    set $\boldsymbol{X}_{1:n}^{(i)} = \left\{\boldsymbol{X}_{1:n-1}^{(i)}, X_n^{(i)}\right\}$. The importance weight is

$$W_n^{(i)} \propto W_{n-1}^{(i)} \frac{g\left(y_n \mid X_n^{(i)}\right) p\left(X_n^{(i)} \mid X_{n-1}^{(i)}, \mu_{n-1}^{(i)}, \Sigma_{n-1}\right)}{q_n\left(X_n^{(i)} \mid y_n, X_{n-1}^{(i)}, \mu_{n-1}^{(i)}, \Sigma_{n-1}\right)}$$

  - Kalman Filter: Use the Kalman filter to compute $\left\{\mu_n^{(i)}, \Sigma_n\right\}, i = 1, \ldots, N$

  - Resampling:
    If variance of weights is high, resample $\left\{X_n^{(i)}, \mu_n^{(i)}, \Sigma_n, W_n^{(i)}\right\}$ to obtain a new population of particles and weights $\left\{X_n^{(i)}, \mu_n^{(i)}, \Sigma_n, W_n^{(i)} = 1/N\right\}$
    Note: The computational complexity ≠ N Kalman filters as $\Sigma_n^{(i)}$ is independent of i.

Revised Implementation of the SISR

- Our Monte Carlo approximation of $p\left(x_n \mid \boldsymbol{y}_{1:n}\right)$ is given as:

$$Inference\ on\ X_n\colon\ \hat{p}_N(x_n|\boldsymbol{y}_{1:n}) = \sum_{i=1}^{N} W_n^{(i)} \delta_{X_n^{(i)}}(x_n)$$

- Inference on Z_n is based on the following (requires N Kalman filters):

$$Inference\ on\ Z_n\colon\ \hat{p}_N(z_n|\boldsymbol{y}_{1:n}) = \int \hat{p}_N(z_n, \boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}) d\boldsymbol{x}_{1:n}$$

$$= \int \hat{p}_N(z_n|\boldsymbol{x}_{1:n}) \hat{p}_N(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}) d\boldsymbol{x}_{1:n} = \sum_{i=1}^{N} W_n^{(i)} \hat{p}_N\left(z_n|X_n^{(i)}, \boldsymbol{y}_{1:n}\right)$$

$$= \sum_{i=1}^{N} W_n^{(i)} \mathcal{N}\left(z_n; \mu_n^{(i)}, \Sigma_n\right), where\ \hat{p}_N(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}) = \sum_{i=1}^{N} W_n^{(i)} \delta_{\boldsymbol{X}_{1:n}^{(i)}}(\boldsymbol{x}_{1:n})$$

- Note using the Eq. above we can write:

$$\widehat{\mathbb{E}}_N(Z_n|\boldsymbol{y}_{1:n}) = \sum_{i=1}^{N} W_n^{(i)} \mu_n^{(i)}$$

$$\widehat{Cov}_N(Z_n|\boldsymbol{y}_{1:n}) = \sum_{i=1}^{N} W_n^{(i)} \left(\Sigma_n + \mu_n^{(i)} \mu_n^{(i)^T}\right) - \widehat{\mathbb{E}}(Z_n|\boldsymbol{y}_{1:n}) \widehat{\mathbb{E}}(Z_n|\boldsymbol{y}_{1:n})^T$$

Example: Dynamic Tobit Model

- Consider the Dynamic Tobit Model

$$Z_0 \sim \mathcal{N}\left(0, \frac{\sigma_v^2}{1-\phi^2}\right)$$

Hidden Markov states

$$Z_n = \phi Z_{n-1} + \sigma_v V_n \quad , \quad v_n \sim i.i.d.\ \mathcal{N}(0,1)$$

$$X_n = Z_n + \sigma_w W_n \quad , \quad \varepsilon_n \sim i.i.d.\ \mathcal{N}(0,1)$$

Observations

$$Y_n = \max\{X_n, 0\}$$

- We set parameters

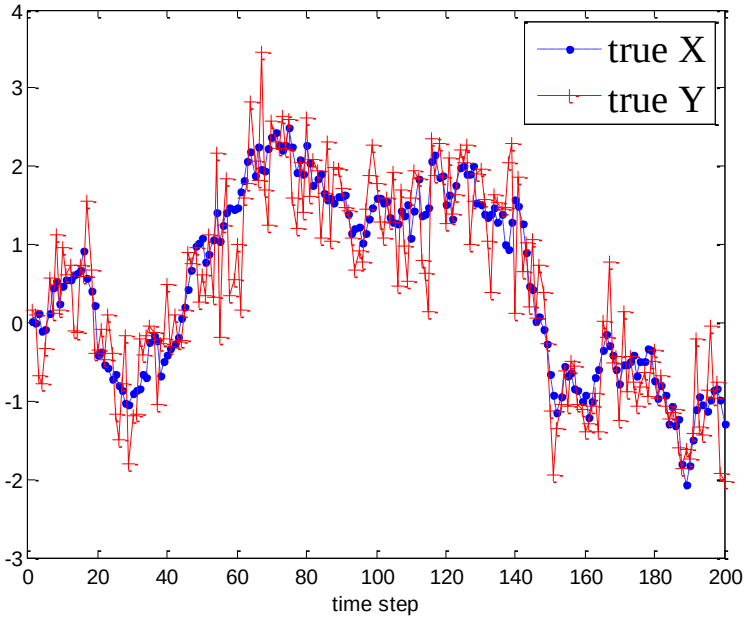
$$\phi = 0.99 \ , \ \sigma_v^2 = 0.05 \ , \ \sigma_w^2 = 0.3$$

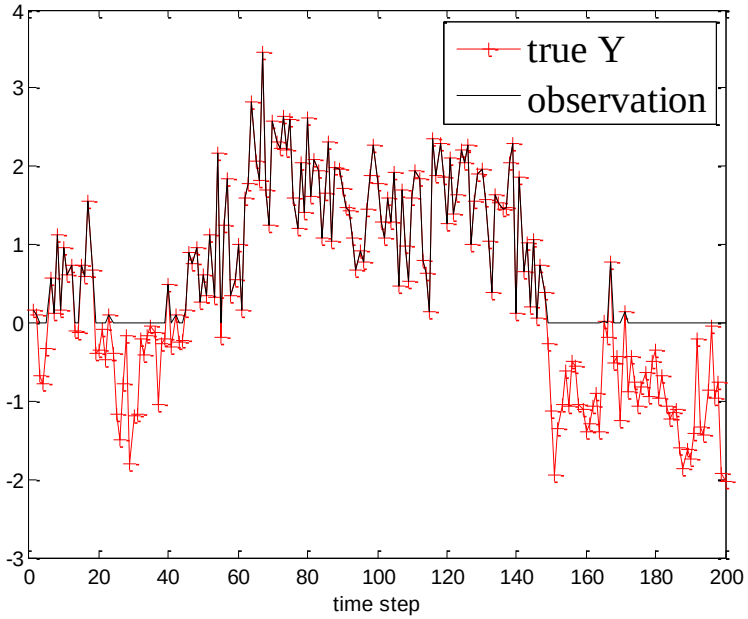
- Andrieu C and Doucet A. Particle filtering for partially observed Gaussian state space models. Journal Of The Royal Statistical Society Series, 2002(64): 827-836

Dynamic Tobit Model

- We simulated T = 200 time steps to obtain "true" states $\{x_n\}$ and $\{z_n\}$ as well as the observation $\{y_n\}$.

- When $x_n<0$, the corresponding observation data are just zero, which means that the real process $\{x_n\}$ are "partially observed".





Dynamic Tobit Model

- To sample from $p(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n})$ at time n, we discuss two conditions
  - Case I : when x_n>0, we directly have

$$y_n = x_n$$

since $y_n = \max\{x_n, 0\}$.

Due to the deterministic relationship between y_n and x_n, the importance weight is simply

$$If\ \ y_n > 0 \cdots then: w_n(\boldsymbol{x}_{1:n}) \propto w_{n-1}(\boldsymbol{x}_{1:n-1}) \mathcal{N}(x_n; m_{n|n-1}, \Sigma_{x,n|n-1})$$

$$If\ \ y_n = 0 \cdots then: q(x_n \mid \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n}) \propto p(x_n \mid \boldsymbol{x}_{1:n-1}) \mathbb{I}_{\mathbb{R}^+}(x_n)$$

$$w_n(\boldsymbol{x}_{1:n}) \propto w_{n-1}(\boldsymbol{x}_{1:n-1}) \Phi\left(-x_{n|n-1} / \sqrt{\Sigma_{x,n|n-1}}\right)$$

Dynamic Tobit Model

- To sample from $p(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n})$ at time n, we discuss two conditions

  - Case II : when y_n=0, we have

$$p(y_n = 0 \mid x_n \le 0) = 1 \quad , \quad p(y_n = 0 \mid x_n > 0) = 0$$

The importance distribution is defined as

$$\begin{aligned} q_n(x_n \mid \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n}) &= p(x_n \mid \boldsymbol{x}_{1:n-1}, y_n = 0) \\ &\propto p(x_n \mid \boldsymbol{x}_{1:n-1}) \mathbb{I}_{(-\infty,0)}(x_n) \\ &= \mathcal{N}(x_n; m_{n|n-1}, \Sigma_{x,n|n-1}) \mathbb{I}_{(-\infty,0)}(x_n) \end{aligned}$$

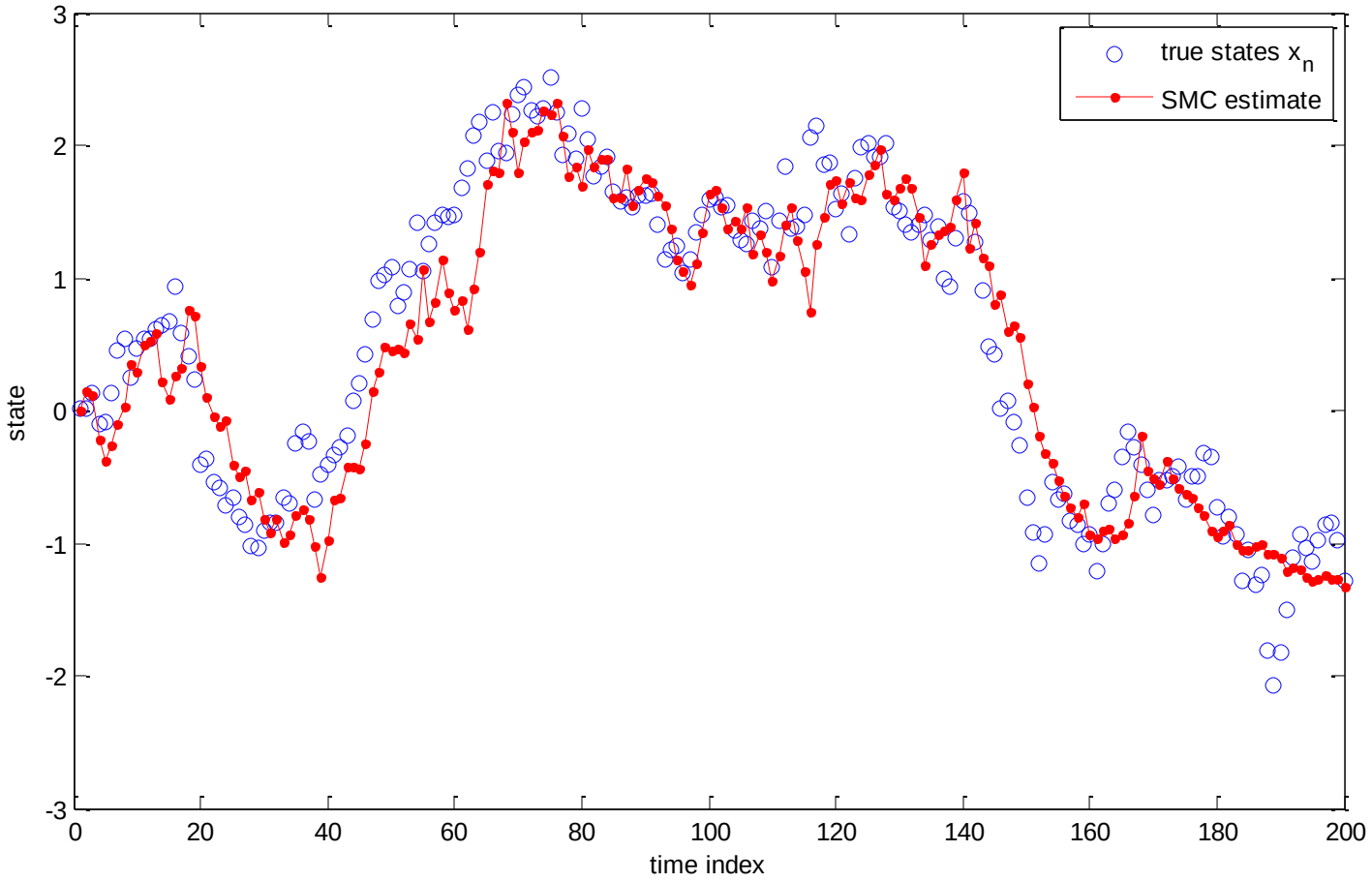
The importance weight is simply

$$\begin{aligned} w(\boldsymbol{x}_{1:n}) \propto p(y_n = 0 \mid \boldsymbol{x}_{1:n-1}) &= \int p(x_n \mid \boldsymbol{x}_{1:n-1}) p(y_n = 0 \mid x_n) dx_n \\ &= \int_{-\infty}^{0} p(x_n \mid \boldsymbol{x}_{1:n-1}) dy_n \\ &= \Phi\left(-x_{n|n-1} / \sqrt{S_n}\right) \end{aligned}$$

where Φ(.) is the cumulative density function (CDF) of the standard normal distribution.

Results

- We simulated *N=500* particles. The posterior mean and corresponding true states for the process $\{x_n\}$ are depicted below

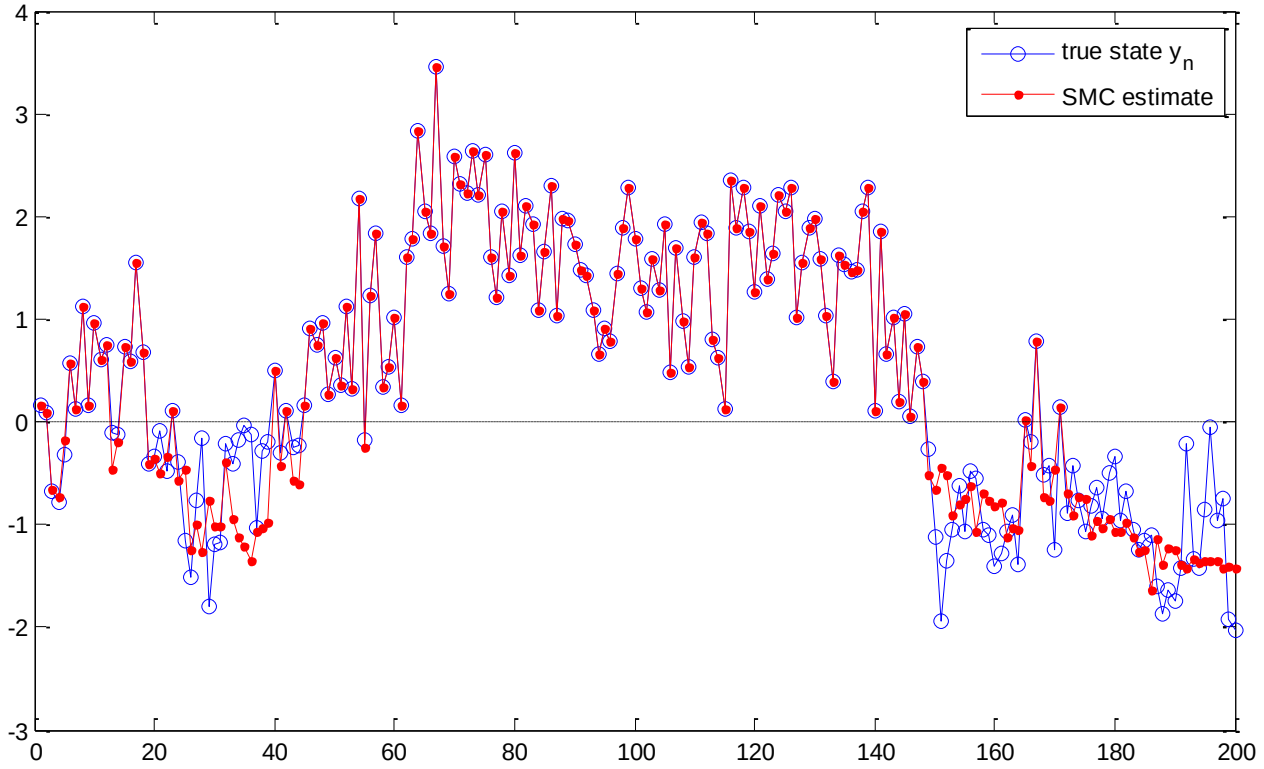


A MatLab implementation of the Tobit model is given here

Example

- We simulated *N=500* particles. The posterior mean and corresponding true states for the process $\{y_n\}$ are depicted below.

- The unobserved states ($y_n < 0$, the plot below the dashed line) are estimated by the SMC method.



Example: Dynamic Probit Model

- We analyze here a non-stationary binary time series Tokyo rainfall data which consists of T = 366 observations with y_n=1 indicating that it rained on the n-th day of the year and y_n=0 otherwise.

- We model y_n by using a dynamic probit model

$$\Pr\left(Y_n = 1 \,|\, \alpha_n\right) = \Phi\left(Z_n\right)$$

where α_n is modeled by using a second-order random walk

$$\alpha_n = 2\alpha_{n-1} - \alpha_{n-2} + \sigma_v V_n \quad , \quad V_n \sim i.i.d.\ \mathcal{N}\left(0,1\right)$$

- Then an artificial latent process $\{X_n\}$ is introduced such that

$$X_n = \alpha_n + W_n \quad , \quad W_n \sim i.i.d.\ \mathcal{N}\left(0,1\right)$$

and define

$$Y_n = \mathbb{I}_{(0,+\infty)}\left(X_n\right)$$

Dynamic Probit Model

- It is easy to check that

$$\Pr\left(Y_n = 1 \mid \alpha_n\right) = \Pr\left(X_n > 0 \mid \alpha_n\right) = \Pr\left(W_n > -\alpha_n\right) = \Phi\left(\alpha_n\right)$$

- Thus the probability of daily rainfall is conditioned on a process $\{\alpha_n\}$.

- The task is to estimate $\{\alpha_n\}$ and $\{\Phi(\alpha_n)\}$ based on observation data.

- Manrique, Aurora, and Neil Shephard. 1998. "Likelihood inference for limted dependent processes." Econometrics Journal 1: C174-C202.
- Andrieu, Christophe, Doucet, Arnaud, Particle filtering for partially observed Gaussian state space models, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 2002

Dynamic Probit Model

- It is easy to check thatType equation here.

$$\Pr\left(Y_n = 1 \mid \alpha_n\right) = \Pr\left(X_n > 0 \mid \alpha_n\right) = \Pr\left(W_n > -\alpha_n\right) = \Phi\left(\alpha_n\right)$$

- Consider the dynamic model

$$\begin{aligned} \alpha_n &= 2\alpha_{n-1} - \alpha_{n-2} + \sigma_v V_n \quad , \quad V_n \sim i.i.d.\ \mathcal{N}(0,1) \\ X_n &= \alpha_n + W_n \quad , \quad W_n \sim i.i.d.\ \mathcal{N}(0,1) \\ Y_n &= \mathbb{I}_{(0,+\infty)}\left(X_n\right) \end{aligned}$$

- Take $Z_n \triangleq (\alpha_n, \alpha_{n-1})^T$, we can rewrite the equations into a state-space model

$$Z_n = \begin{bmatrix} 2 & -1 \\ 1 & 0 \end{bmatrix} Z_{n-1} + \sigma_v \begin{bmatrix} V_n \\ 0 \end{bmatrix} \quad , \quad v_n \sim i.i.d.\ N(0,1)$$

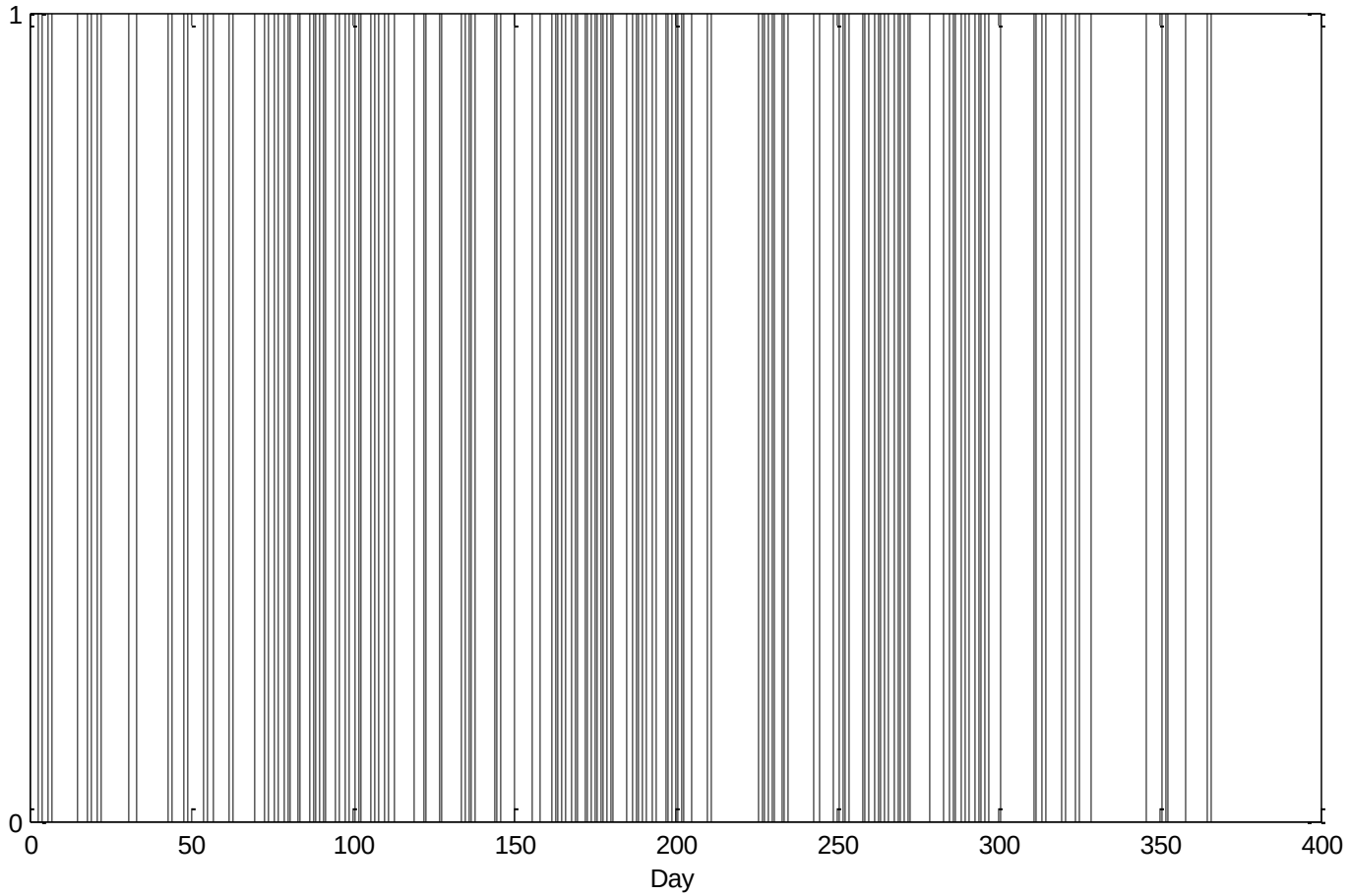
$$X_n = (1 \quad 0)^T \tilde{Z}_n + W_n$$

$$Y_n = \mathbb{I}_{(0,+\infty)}(X_n)$$

- In this example, we set $\sigma_v^2 = 0.01$.

Dynamic Probit Model

- The binary observation data for the rainfall in a year (366 days) are given as (1 – rainy, 0 – no rain)



366 days in a year

Dynamic Probit Model

- The introduction of X_n being artificial, it is necessary to apply SMC to the estimation of $p(\boldsymbol{z}_{1:t}|\boldsymbol{y}_{1:n})$ and not $p(\boldsymbol{z}_{1:n}, \boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n})$. The motivation for introducing X_n comes from the fact that we can now use the optimal density as importance density:

$$p\left(x_n \mid \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n}\right) = p\left(x_n \mid \boldsymbol{x}_{1:n-1}, y_n\right) \propto \begin{cases} p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right) \mathbb{I}_{[0,+\infty)}\left(x_n\right) & , \quad \text{if} \quad y_n = 1 \\ p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right) \mathbb{I}_{(-\infty,0)}\left(x_n\right) & , \quad \text{if} \quad y_n = 0 \end{cases}$$

- This is a truncated Gaussian. Using the general equation

$$\begin{aligned} w\left(\boldsymbol{x}_{1:n}\right) \propto p\left(y_n \mid \boldsymbol{x}_{1:n-1}\right) &= \int p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right) p\left(y_n \mid x_n\right) dx_n \\ &= \int_0^{\infty} p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right) dx_n \\ &= 1 - \Phi\left(-\frac{m_{n|n-1}}{\sqrt{S_n}}\right) \end{aligned}$$

one obtains:

$$w\left(\boldsymbol{x}_{1:n}\right) \propto p\left(y_n \mid \boldsymbol{x}_{1:n-1}\right) = \left(1 - \Phi\left(-\frac{m_{n|n-1}}{\sqrt{S_n}}\right)\right)^{y_n} \Phi\left(-\frac{m_{n|n-1}}{\sqrt{S_n}}\right)^{1-y_n}$$

Dynamic Probit Model

- The importance weight is defined as

$$p\left(x_n \mid \boldsymbol{x}_{1:n-1}, \boldsymbol{y}_{1:n}\right) = p\left(x_n \mid \boldsymbol{x}_{1:n-1}, y_n\right) \propto \begin{cases} p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right) \mathbb{I}_{[0,+\infty)}\left(x_n\right) & , \quad \text{if} \quad y_n = 1 \\ p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right) \mathbb{I}_{(-\infty,0)}\left(x_n\right) & , \quad \text{if} \quad y_n = 0 \end{cases}$$

  which is a truncated Gaussian distribution

- The importance weight is

  - when $y_n = 1$

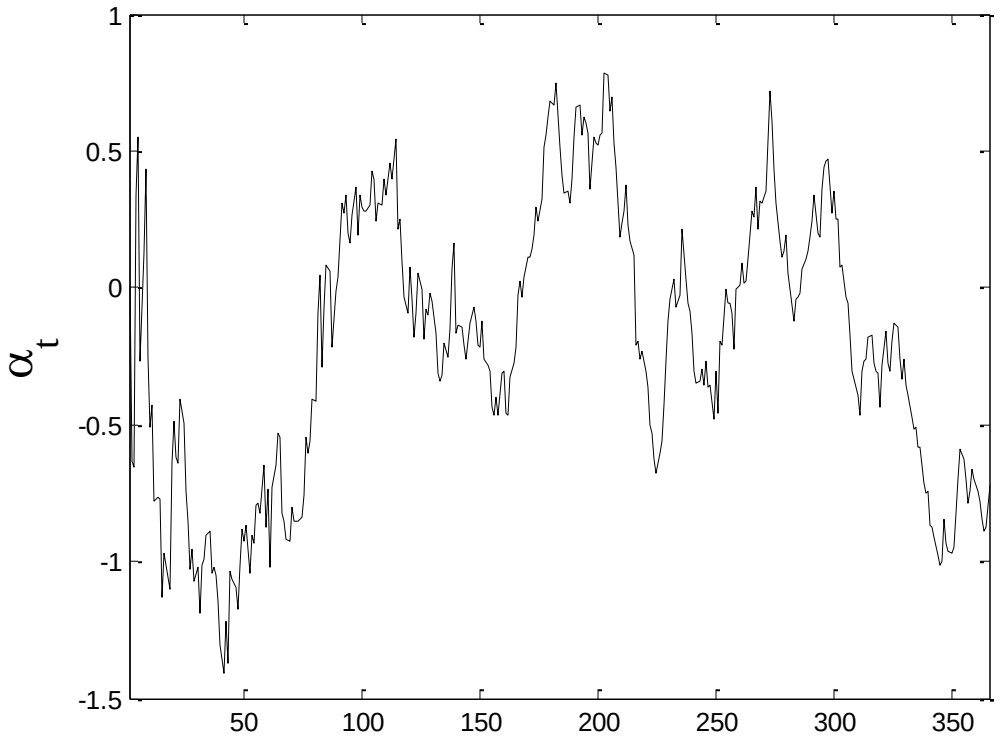
$$\begin{aligned} w\left(\boldsymbol{x}_{1:n}\right) \propto p\left(y_n \mid \boldsymbol{x}_{1:n-1}\right) &= \int p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right) p\left(y_n \mid x_n\right) dx_n \\ &= \int_0^{\infty} p\left(x_n \mid \boldsymbol{x}_{1:n-1}\right) dx_n \\ &= 1 - \Phi\left(-\frac{m_{n|n-1}}{\sqrt{S_n}}\right) \end{aligned}$$

  - when $y_n = 0$

$$\begin{aligned} w\left(\boldsymbol{x}_{1:n}\right) &\propto \int_{-\infty}^{0} p\left(x_n \mid \boldsymbol{y}_{1:n-1}\right) dx_n \\ &= \Phi\left(-\frac{m_{n|n-1}}{\sqrt{S_n}}\right) \end{aligned}$$

Results

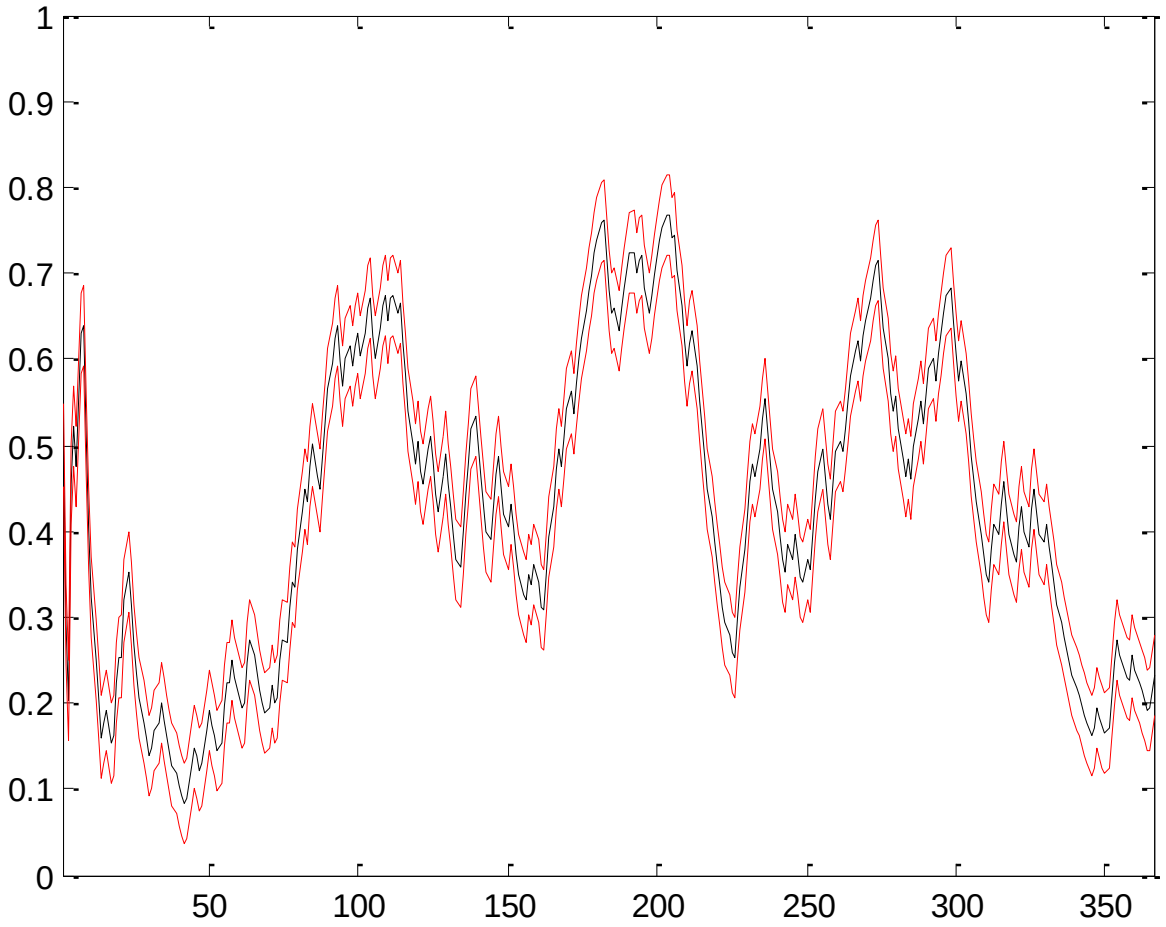
- The posterior mean of $\{z_n\}$ is



A MatLab implementation of the Probit model is given here

Results

- Here the expectation $\mathbb{E}\{\Phi(\alpha_n)|\boldsymbol{x}_{1:n}\}$ (black line) and $\mathbb{E}\{\Phi(z_n)|\boldsymbol{x}_{1:n}\} \pm \sqrt{\text{var}\{\Phi(\alpha_n)|\boldsymbol{x}_{1:n}\}}$ (red lines) are plotted



Sequential Monte Carlo Fixed-Lag Smoothing Approximation

- P. Del Moral, A. Doucet & S.S. Singh, Forward Smoothing using Sequential Monte Carlo, technical report, Cambridge University, 2009
- Godsill, Doucet, and West: Monte Carlo Smoothing for Nonlinear Time Series. 157, Journal of the American Statistical Association, March 2004, Vol. 99, No. 465.
- P. Del Moral, A. Doucet & S.S. Singh, Forward Smoothing using Sequential Monte Carlo, technical report, Cambridge University, 2009
- Paul Fearnhead David Wyncoll Jonathan Tawn *Biometrika*, Volume 97, Issue 2, 1 June 2010, Pages 447–464, A sequential smoothing algorithm with linear computational cost.
- Mark Briers Arnaud DoucetEmail author Simon Maskell, Smoothing algorithms for state–space models Annals of the Institute of Statistical Mathematics February 2010, 62:61

Smoothing Approximation

- Note that $p(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}) = p(x_n|\boldsymbol{y}_{1:n}) \prod_{k=1}^{n-1} p(x_k|\boldsymbol{x}_{k+1:n}, \boldsymbol{y}_{1:n})$. This defines the general smoothing problem.
  - Particular estimate of interest $p(x_k|\boldsymbol{y}_{1:n})$
- Godsill et al. proposed a solution to this smoothing problem by proposing a particle approximation to each term in the product $\prod_{k=1}^{n-1} p(x_k|\boldsymbol{x}_{k+1:n}, \boldsymbol{y}_{1:n})$.
- Using the Markov property note the following:

$$p(x_k|\boldsymbol{x}_{k+1:n}, \boldsymbol{y}_{1:n}) = p(x_k|x_{k+1}, \boldsymbol{y}_{1:k}) = \frac{p(x_k, x_{k+1}|\boldsymbol{y}_{1:k})}{p(x_{k+1}|\boldsymbol{y}_{1:k})}$$
$$\propto f(x_{k+1}|x_k)\, p(x_k|\boldsymbol{y}_{1:k})$$

- Here prior=$f(x_{k+1}|x_k)$ and filtering distribution = $p(x_k|\boldsymbol{y}_{1:k})$.

- Godsill, Doucet, and West: Monte Carlo Smoothing for Nonlinear Time Series. 157, Journal of the American Statistical Association, March 2004, Vol. 99, No. 465.

Smoothing Approximation

$$p(x_k|\boldsymbol{x}_{k+1:n}, \boldsymbol{y}_{1:n}) \propto f(x_{k+1}|x_k)p(x_k|\boldsymbol{y}_{1:k})$$

- Here prior=$f(x_{k+1}|x_k)$ and filtering distribution = $p(x_k|\boldsymbol{y}_{1:k})$.

- Given a particle approximation $\left\{W_k^{(i)}, x_k^{(i)}\right\}_{i=1}^N$ of the filtering density $p(x_k|\boldsymbol{y}_{1:k})$ and a sample X_{k+1}, we can obtain a particle approximation of $p(x_k|\boldsymbol{x}_{k+1:n}, \boldsymbol{y}_{1:n})$ by updating the weights as follows:

$$w_{k|k+1}^{(i)} = W_k^{(i)} f\left(X_{k+1}|X_k^{(i)}\right) \text{ and } W_{k|k+1}^{(i)} = \frac{w_{k|k+1}^{(i)}}{\sum_i w_{k|k+1}^{(i)}}$$

- Recalling that $p(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}) = p(x_n|\boldsymbol{y}_{1:n}) \prod_{k=1}^{n-1} p(x_k|\boldsymbol{x}_{k+1:n}, \boldsymbol{y}_{1:n})$, we can implement an algorithm for computing a particle approximation of $p(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n})$.

Smoothing Approximation: Algorithm

$$p(x_k|\boldsymbol{x}_{k+1:n}, \boldsymbol{y}_{1:n}) \propto f(x_{k+1}|x_k)p(x_k|\boldsymbol{y}_{1:k})$$

$$w_{k|k+1}^{(i)} = W_k^{(i)} f\left(X_{k+1}|X_k^{(i)}\right) \text{ and } W_{k|k+1}^{(i)} = \frac{w_{k|k+1}^{(i)}}{\sum_i w_{k|k+1}^{(i)}}$$

$$p(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}) = p(x_n|\boldsymbol{y}_{1:n}) \prod_{k=1}^{n-1} p(x_k|\boldsymbol{x}_{k+1:n}, \boldsymbol{y}_{1:n})$$

- Draw $\hat{X}_n$ from the particle approximation $\left\{W_n^{(i)}, x_n^{(i)}\right\}_{i=1}^N$ of $p(x_n|\boldsymbol{y}_{1:n})$
- For $k = n-1, \ldots, 1$
  - Calculate $w_{k|k+1}^{(i)} = W_k^{(i)} f\left(\hat{X}_n|X_k^{(i)}\right)$ for i=1,2,…,n
  - Draw $\hat{X}_k$ with probability $\propto w_{k|k+1}^{(i)}$
- Then $\hat{\boldsymbol{X}}_{1:n} = \{\hat{X}_1, \ldots, \hat{X}_n\}$ is a sample from $p(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n})$
- Further independent realizations are obtained by repeating this procedure as many times as needed.

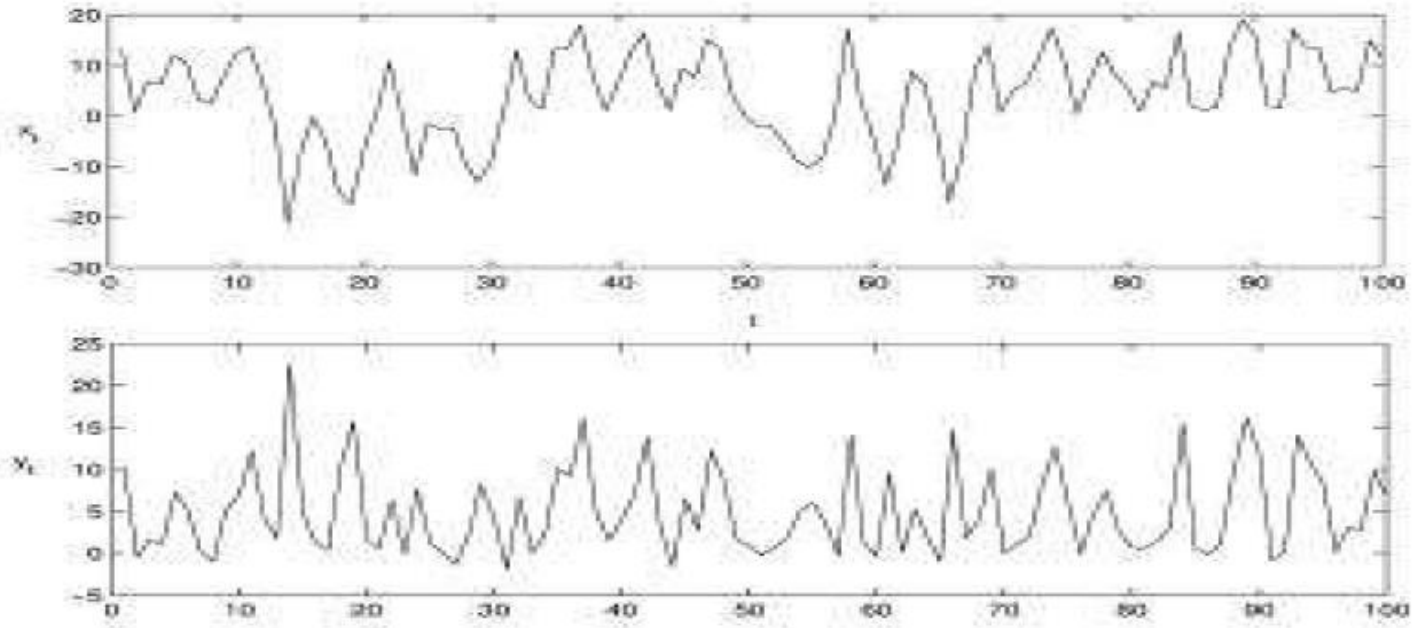
Smoothing Approximation

$$X_k = \frac{X_{k-1}}{2} + 25\frac{X_{k-1}}{1+X_{k-1}^2} + 8cos(1.2k) + u_k,\ u_k \sim \mathcal{N}(0,10)$$

$$Y_k = \frac{X_k^2}{20} + w_k,\ w_k \sim \mathcal{N}(0,1)$$

$x_n\ (true)$

y_n



- Godsill, Doucet, and West: Monte Carlo Smoothing for Nonlinear Time Series. 157, Journal of the American Statistical Association, March 2004, Vol. 99, No. 465.

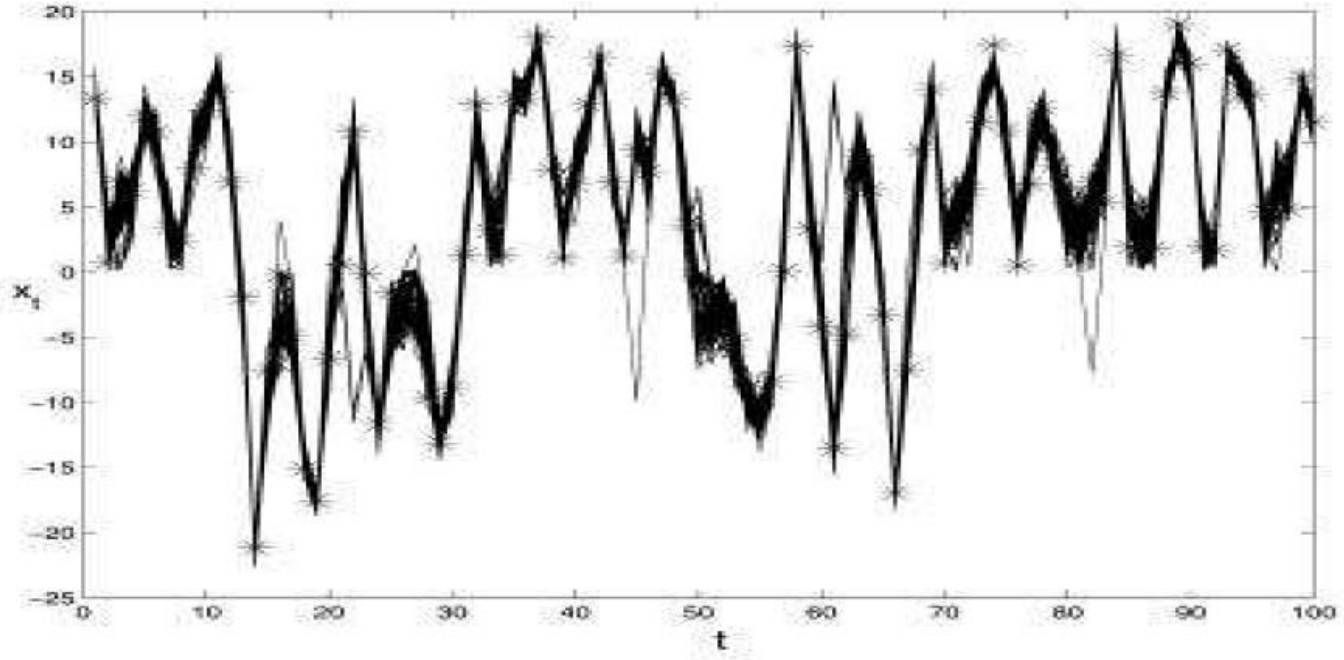
Smoothing Approximation

$$X_k = \frac{X_{k-1}}{2} + 25\frac{X_{k-1}}{1+X_{k-1}^2} + 8cos(1.2k) + u_k,\ u_k \sim \mathcal{N}(0,10)$$

$$Y_k = \frac{X_k^2}{20} + w_k,\ w_k \sim \mathcal{N}(0,1)$$

Smoothing trajectories drawn from

$p(\boldsymbol{x}_{1:100}|\boldsymbol{y}_{1:100})$



- Godsill, Doucet, and West: Monte Carlo Smoothing for Nonlinear Time Series. 157, Journal of the American Statistical Association, March 2004, Vol. 99, No. 465.

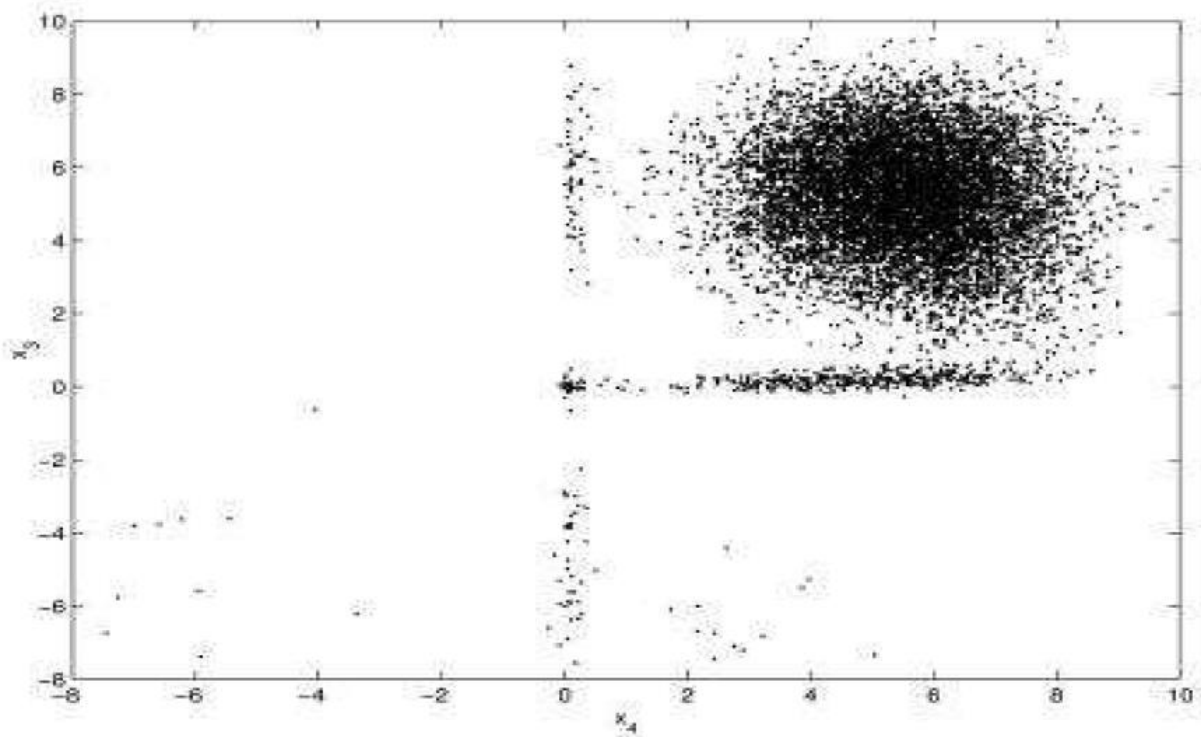
Smoothing Approximation

$$X_k = \frac{X_{k-1}}{2} + 25\frac{X_{k-1}}{1+X_{k-1}^2} + 8cos(1.2k) + u_k,\ u_k \sim \mathcal{N}(0,10)$$

$$Y_k = \frac{X_k^2}{20} + w_k,\ w_k \sim \mathcal{N}(0,1)$$

Scatter plot of samples drawn from

$p(\boldsymbol{x}_{3:4}|\boldsymbol{y}_{1:100})$



- Godsill, Doucet, and West: Monte Carlo Smoothing for Nonlinear Time Series. 157, Journal of the American Statistical Association, March 2004, Vol. 99, No. 465.

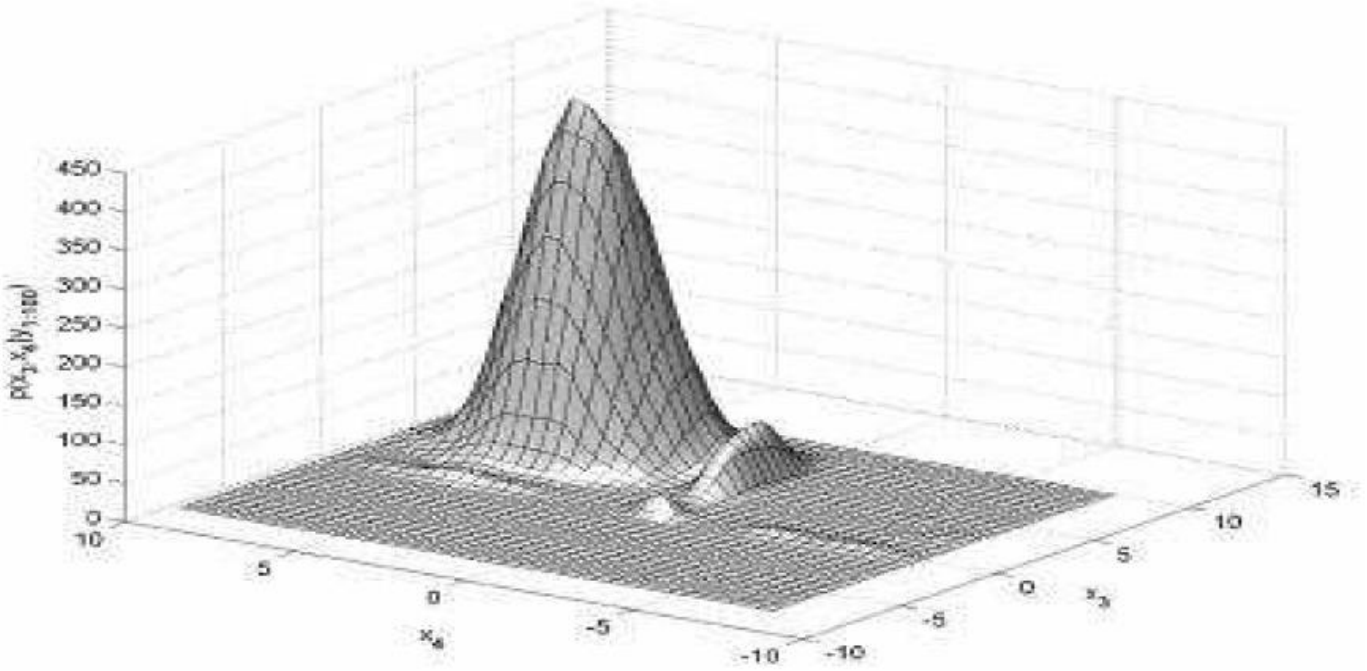
Smoothing Approximation

$$X_k = \frac{X_{k-1}}{2} + 25\frac{X_{k-1}}{1+X_{k-1}^2} + 8cos(1.2k) + u_k,\ u_k \sim \mathcal{N}(0,10)$$

$$Y_k = \frac{X_k^2}{20} + w_k,\ w_k \sim \mathcal{N}(0,1)$$

Kernel density estimate of

$p(\boldsymbol{x}_{3:4}|\boldsymbol{y}_{1:100})$



- Godsill, Doucet, and West: Monte Carlo Smoothing for Nonlinear Time Series. 157, Journal of the American Statistical Association, March 2004, Vol. 99, No. 465.

ONLINE PARAMETER ESTIMATION

Online Bayesian Parameter Estimation

- Assume that our state model is defined with some unknown static parameter θ with some prior $p(\theta)$:

$$X_1 \sim \mu(.) \; and \; X_n \,|\, (X_{n-1} = x_{n-1}) \sim f_\theta(x_n \,|\, x_{n-1})$$

$$Y_n \,|\, (X_n = x_n) \sim g_\theta(y_n \,|\, x_n)$$

- Given data $\boldsymbol{y}_{1:n}$, inference now is based on:

$$p(\theta, \boldsymbol{x}_{1:n} \,|\, \boldsymbol{y}_{1:n}) = p(\theta \,|\, \boldsymbol{y}_{1:n})\, p_\theta(\boldsymbol{x}_{1:n} \,|\, \boldsymbol{y}_{1:n}),$$

$$where$$

$$p(\theta \,|\, \boldsymbol{y}_{1:n}) \propto p_\theta(\boldsymbol{y}_{1:n})\, p(\theta)$$

- We can use standard SMC but on the extended space $Z_n=(X_n,\theta_n)$.

$$f(z_n \,|\, z_{n-1}) = \delta_{\theta_{n-1}}(\theta_n)\, f_\theta(x_n \,|\, x_{n-1}),\; g(y_n \,|\, z_n) = g_\theta(y_n \,|\, x_n)$$

- Note that θ is a static parameter –does not involve with n.

Online Bayesian Parameter Estimation

- For fixed θ, using our earlier error estimates

$$Var[\log \hat{p}_\theta(\boldsymbol{y}_{1:n})] \leq \frac{Cn}{N}$$

- In a Bayesian context, the problem is even more severe as

$$p(\theta \mid \boldsymbol{y}_{1:n}) \propto p_\theta(\boldsymbol{y}_{1:n})\, p(\theta)$$

- Exponential stability assumption cannot hold as $\theta_n = \theta_1$.

- To mitigate this problem, introduce MCMC steps on θ.

- C. Andrieu, N. De Freitas and A. Doucet, Sequential MCMC for Bayesian Model Selection, Proc. IEEE Workshop HOS, 1999
- P. Fearnhead, MCMC, sufficient statistics and particle filters, JCGS, 2002
- W Gilks and C. Berzuini, Following a moving target: MC inference for dynamic Bayesian Models, JRSS B, 2001
- Storvik, G., 2002, Particle filters in state space models with the presence of unknown static parameters, IEEE. Trans. of Signal Processing 50, 281—289.
- Eric Jacquier, Michael Johannes, Nicholas Polson, MCMC maximum likelihood for latent state models, In Journal of Econometrics, Volume 137, Issue 2, 2007, Pages 615-640.
- T. Vercauteren et al, Batch and Sequential Bayesian Estimators of the Number of Active Terminals in an IEEE 802.11 Network.

Recursive Bayesian Parameter Estimation using SMC

- We set $\theta \sim \pi(\theta)$, and try to estimate $p(\theta, \boldsymbol{x}_{1:n} | \boldsymbol{y}_{1:n})$ using sequential Monte Carlo.
- SMC is applicable as we know $p(\theta, \boldsymbol{x}_{1:n} | \boldsymbol{y}_{1:n})$ up to normalizing factor.
- However, SMC does not work as we only sample particles in Θ space at time $n = 1$ and never modify their locations. Thus after a few time steps, $p(\theta \,|\boldsymbol{y}_{1:n})$ is approximated by a single particle.
- The higher the dimensionality of θ, the faster the degeneracy arises.
- A potential solution is by adding artificial noise, e.g. $\theta_k = \theta_{k-1} + \varepsilon_k$. This modifies however the target distributions.

Artificial Dynamics for θ

- A solution consists of perturbing the location of the particles $\{\theta^{(i)}\}$ in a way that does not modify their distributions; i.e. if at time n

$$\theta_n^{(i)} \sim p(\theta \mid \boldsymbol{y}_{1:n})$$

then we would like a transition kernel such that if

$$\theta_n^{'(i)} \mid \theta_n^{(i)} \sim M_n(\theta_n^{(i)}, .)$$

Then:

$$\theta_n^{'(i)} \sim p(\theta \mid \boldsymbol{y}_{1:n})$$

- In Markov chain language, we want $M_n(\theta, \theta')$ to be $p(\theta \mid \boldsymbol{y}_{1:n})$ invariant.

Artificial Dynamics Using MCMC

- There is a whole literature on the design of such kernels known as Markov chain Monte Carlo e.g. the Metropolis-Hastings algorithm.

- We cannot use these algorithms directly as $p\left(\theta \mid \boldsymbol{y}_{1:n}\right)$ would need to be known up to a normalizing constant but $p\left(\boldsymbol{y}_{1:n} \mid \theta\right) \equiv p_{\theta}\left(\boldsymbol{y}_{1:n}\right)$ is unknown.

- However, we can use a simple Gibbs update.

$$\theta_n^{(i)} \sim p\left(\theta \mid \boldsymbol{y}_{1:n}, \boldsymbol{X}_{1:n}^{(i)}\right)$$

- Indeed note that if $\left(\boldsymbol{X}_{1:n}^{(i)}, \theta_n^{(i)}\right) \sim p\left(\theta, \boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right)$ then if $\theta_n^{'(i)} \sim p\left(\theta \mid \boldsymbol{y}_{1:n}, \boldsymbol{X}_{1:n}^{(i)}\right)$, we have:

$$\left(\boldsymbol{X}_{1:n}^{(i)}, \theta_n^{'(i)}\right) \sim p\left(\theta, \boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right)$$

- Indeed note that:

$$\int p\left(\theta, \boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right) p\left(\theta' \mid \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n}\right) d\theta = p\left(\theta', \boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right)$$

SMC with MCMC for Parameter Estimation

- Given an approximation at time n-1:

$$\hat{p}(\theta, \boldsymbol{x}_{1:n-1}|\boldsymbol{y}_{1:n-1}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{\left(\theta_{n-1}^{(i)}, \boldsymbol{X}_{1:n-1}^{(i)}\right)}(\theta, \boldsymbol{x}_{1:n-1})$$

- Sample $\tilde{X}_n^{(i)} \sim f_{\theta_{n-1}^{(i)}}\left(x_n|X_{n-1}^{(i)}\right)$, set $\tilde{X}_{1:n}^{(i)} \sim \left(\boldsymbol{X}_{1:n-1}^{(i)}, \tilde{X}_n^{(i)}\right)$ and then approximate:

$$\tilde{p}(\theta, \boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}) = \sum_{i=1}^{N} W_n^{(i)} \delta_{\left(\theta_{n-1}^{(i)}, \tilde{\boldsymbol{X}}_{1:n}^{(i)}\right)}(\boldsymbol{x}_{1:n}),\ W_n^{(i)} \propto g_{\theta_{n-1}^{(i)}}\left(y_n|\tilde{X}_n^{(i)}\right)$$

- Resample $\boldsymbol{X}_{1:n}^{(i)} \sim \tilde{p}(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n})$, then sample $\theta_n^{(i)} \sim p\left(\theta|\boldsymbol{y}_{1:n}, \boldsymbol{X}_{1:n}^{(i)}\right)$ to obtain

$$\hat{p}(\theta, \boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{\left(\theta_n^{(i)}, \boldsymbol{X}_{1:n}^{(i)}\right)}(\theta, \boldsymbol{x}_{1:n})$$

SMC with MCMC for Parameter Estimation

- Consider the following model:

$$X_{n+1} = \theta X_n + \sigma_v V_{n+1}, V_n \sim \mathcal{N}(0,1)$$
$$Y_n = X_n + \sigma_w W_n, W_n \sim \mathcal{N}(0,1)$$
$$X_1 \sim \mathcal{N}(0, \sigma_0^2)$$

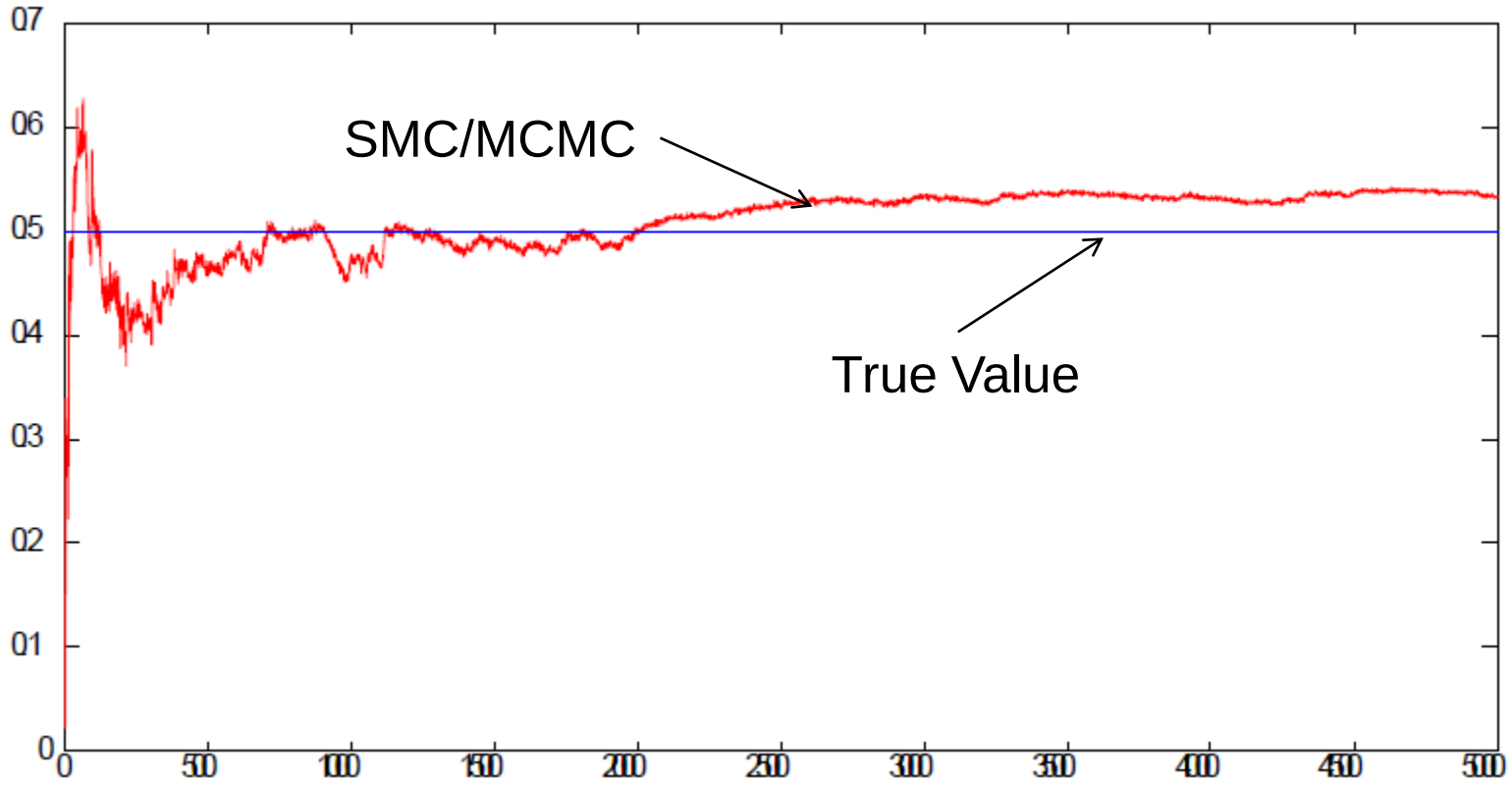
- We set the prior on θ as $\theta \sim \mathcal{U}(-1,1)$.

- In this case,

$$p(\theta \mid \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n}) \propto \mathcal{N}(\theta \mid m_n, \sigma_n^2) \mathbb{I}_{(-1,1)}(\theta)$$
$$m_n = \sigma_n^2 \left(\sum_{k=2}^{n} x_k x_{k-1} \right), \sigma_n^{-2} = \sum_{k=2}^{n-1} x_k^2$$

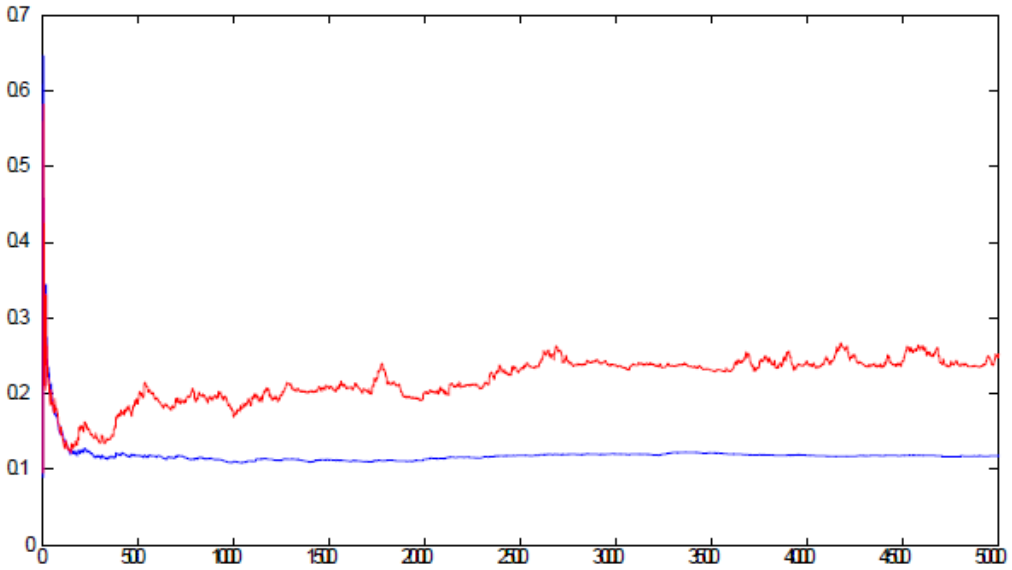
SMC with MCMC for Parameter Estimation

- We use SMC with Gibbs step. The degeneracy problem remains. SMC estimate is shown of $\mathbb{E}[\theta|\mathbf{y}_{1:n}]$ as n increases (From A. Doucet, lecture notes). The parameter converges to the wrong value.



SMC with MCMC for Parameter Estimation

- The problem with this approach is that although we move θ according to $p(\theta \mid \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n})$, we are still relying implicitly on the approximation of the joint distribution $p(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n})$
- As n increases, the SMC approximation of $p(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n})$ deteriorates and the algorithm cannot converge towards the right solution. If it does converge, the results may be incorrect.



Sufficient statistics $\frac{1}{n}\sum_{k=1}^{n}\mathbb{E}\left[X_k^2 \mid \boldsymbol{y}_{1:n}\right]$ computed exactly through the Kalman filter (blue) vs SMC approximation (red) for a fixed value of θ.

Marginal Metropolis Hastings Algorithm

- Consider the target:

$$p\left(\theta, \boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right) = p\left(\theta \mid \boldsymbol{y}_{1:n}\right) p_{\theta}\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right)$$

- We use the following proposal distribution:

$$q\left(\left(\boldsymbol{x}_{1:n}^{*}, \theta^{*}\right) \mid \left(\boldsymbol{x}_{1:n}, \theta\right)\right) = q\left(\theta^{*} \mid \theta\right) p_{\theta^{*}}\left(\boldsymbol{x}_{1:n}^{*} \mid \boldsymbol{y}_{1:n}\right)$$

- Then the acceptance probability becomes:

$$\min\left(1, \frac{p\left(\theta^{*}, \boldsymbol{x}_{1:n}^{*} \mid \boldsymbol{y}_{1:n}\right) q\left(\left(\boldsymbol{x}_{1:n}, \theta\right) \mid \left(\boldsymbol{x}_{1:n}^{*}, \theta^{*}\right)\right)}{p\left(\theta, \boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right) q\left(\left(\boldsymbol{x}_{1:n}^{*}, \theta^{*}\right) \mid \left(\boldsymbol{x}_{1:n}, \theta\right)\right)}\right) =$$

$$\min\left(1, \frac{p_{\theta^{*}}\left(\boldsymbol{y}_{1:n}\right) p\left(\theta^{*}\right) q\left(\theta \mid \theta*\right)}{p_{\theta}\left(\boldsymbol{y}_{1:n}\right) p\left(\theta\right) q\left(\theta^{*} \mid \theta\right)}\right)$$

- We will use SMC approximations to compute

$$p_{\theta}\left(\boldsymbol{y}_{1:n}\right), \text{and } p_{\theta}\left(\boldsymbol{x}_{1:n} \mid \boldsymbol{y}_{1:n}\right)$$

Marginal Metropolis Hastings Algorithm

- Step 1:

$$Given: \left\{\theta^{(i-1)}, \boldsymbol{X}_{1:n}^{(i-1)}, \hat{p}_{\theta^{(i-1)}}(\boldsymbol{y}_{1:n})\right\}$$
$$Sample: \theta^* \sim q\left(\theta|\theta^{(i-1)}\right)$$
$$Run\ an\ SMC\ to\ obtain: \hat{p}_{\theta^*}(\boldsymbol{y}_{1:n}), p_{\theta^*}(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n})$$

- Step 2:

$$Sample: \boldsymbol{X}_{1:n}^* \sim \hat{p}_{\theta^*}(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n})$$

- Step 3: With probability $\min\left(1, \frac{\hat{p}_{\theta^*}(\boldsymbol{y}_{1:n})p(\theta^*)q\left(\theta^{(i-1)}|\theta^*\right)}{\hat{p}_{\theta^{(i-1)}}(\boldsymbol{y}_{1:n})p(\theta^{(i-1)})q(\theta^*|\theta^{(i-1)})}\right)$

$$Set: \left\{\theta^{(i)}, \boldsymbol{X}_{1:n}^{(i)}, \hat{p}_{\theta^{(i)}}(\boldsymbol{y}_{1:n})\right\} = \{\theta^*, \boldsymbol{X}_{1:n}^*, \hat{p}_{\theta^*}(\boldsymbol{y}_{1:n})\}$$
$$Otherwise: \left\{\theta^{(i)}, \boldsymbol{X}_{1:n}^{(i)}, \hat{p}_{\theta^{(i)}}(\boldsymbol{y}_{1:n})\right\} = \left\{\theta^{(i-1)}, \boldsymbol{X}_{1:n}^{(i-1)}, \hat{p}_{\theta^{(i-1)}}(\boldsymbol{y}_{1:n})\right\}$$

- The advantage of SMC is that it builds automatically efficient very high-dimensional proposal distributions based only on low-dimensional proposals.

Recursive Parameter Estimation

Recursive MLE Parameter Estimation

- The log likelihood can be written as:

$$\ell_n(\theta) = \log p_\theta(\boldsymbol{Y}_{1:n}) = \sum_{k=1}^{n} \log p_\theta\left(Y_k \mid \boldsymbol{Y}_{1:k-1}\right)$$

- Here we compute:

$$p_\theta\left(Y_k \mid \boldsymbol{Y}_{1:k-1}\right) = \int g_\theta\left(Y_k \mid x_k\right) p_\theta\left(x_k \mid \boldsymbol{Y}_{1:k-1}\right) dx_k$$

- Under regularity assumptions $\left\{X_n, Y_n, p_\theta\left(x_n \mid \boldsymbol{Y}_{1:n-1}\right)\right\}$ is an homogeneous Markov chain whish converges towards its invariant distribution:

$$\lim_{n\to\infty} \frac{\ell_n(\theta)}{n} = \ell(\theta) = \int \log \int g_\theta(y \mid x)\mu(dx)\lambda_{\theta,\theta^*}(dy, d\mu)$$

Robbins-Monro Algorithm for MLE

- We can maximize $\ell(\theta)$ by using the gradient and the Robbins-Monro algorithm:

$$\theta_n = \theta_{n-1} + \gamma_n \nabla \log p_{\theta_{1:n-1}}(Y_n \mid \boldsymbol{Y}_{1:n-1})$$

where:

$$\nabla p_\theta(Y_n \mid \boldsymbol{Y}_{1:n-1}) = \int \nabla g_\theta\left(Y_n \mid x_n\right) p_\theta\left(x_n \mid \boldsymbol{Y}_{1:n-1}\right) dx_n + \int g_\theta\left(Y_n \mid x_n\right) \nabla p_\theta\left(x_n \mid \boldsymbol{Y}_{1:n-1}\right) dx_n$$

- We thus need to approximate the signed measures: $\left\{\nabla p_\theta\left(x_n \mid \boldsymbol{Y}_{1:n-1}\right)\right\}$

Marginal Importance Sampling for Sensitivity Estimation

- An alternative identity is the following:

$$\nabla p_\theta(x_n \mid \boldsymbol{Y}_{1:n-1}) = \nabla \log p_\theta(x_n \mid \boldsymbol{Y}_{1:n-1})\, p_\theta(x_n \mid \boldsymbol{Y}_{1:n-1})$$

- It suggests using an SMC approximation of the form:

$$\widehat{\nabla p_\theta}(x_n \mid \boldsymbol{Y}_{1:n-1}) = \frac{1}{N}\sum_{i=1}^{N} \beta_n^{(i)} \delta_{X_n^{(i)}}(x_n)$$

$$\beta_n^{(i)} = \widehat{\nabla \log p_\theta}(X_n^{(i)} \mid \boldsymbol{Y}_{1:n-1}) = \frac{\widehat{\nabla p_\theta}(X_n^{(i)} \mid \boldsymbol{Y}_{1:n-1})}{\hat{p}_\theta(X_n^{(i)} \mid \boldsymbol{Y}_{1:n-1})}$$

- Such an approximation relies on a pointwise approximation of both the filter and its derivative. The computational complexity is $\mathcal{O}(N^2)$ as

$$\hat{p}_\theta(X_n^{(i)} \mid \boldsymbol{Y}_{1:n-1}) \propto \int f_\theta(X_n^{(i)} \mid x_{n-1}) \hat{p}_\theta(x_{n-1} \mid \boldsymbol{Y}_{1:n-1}) dx_{n-1} = \frac{1}{N}\sum_{j=1}^{N} f_\theta(X_n^{(i)} \mid X_{n-1}^{(j)})$$

Example: Stochastic Volatility Model

- Consider the following model:

$$X_n = \phi X_{n-1} + \sigma_v V_n$$

$$Y_n = \beta \exp\left(\frac{X_n}{2}\right) W_n$$

- We have $\theta = \{\phi, \sigma_v, \beta\}$. We use SMC with N=1000 for batch and on-line estimation.

- For Batch Estimation:

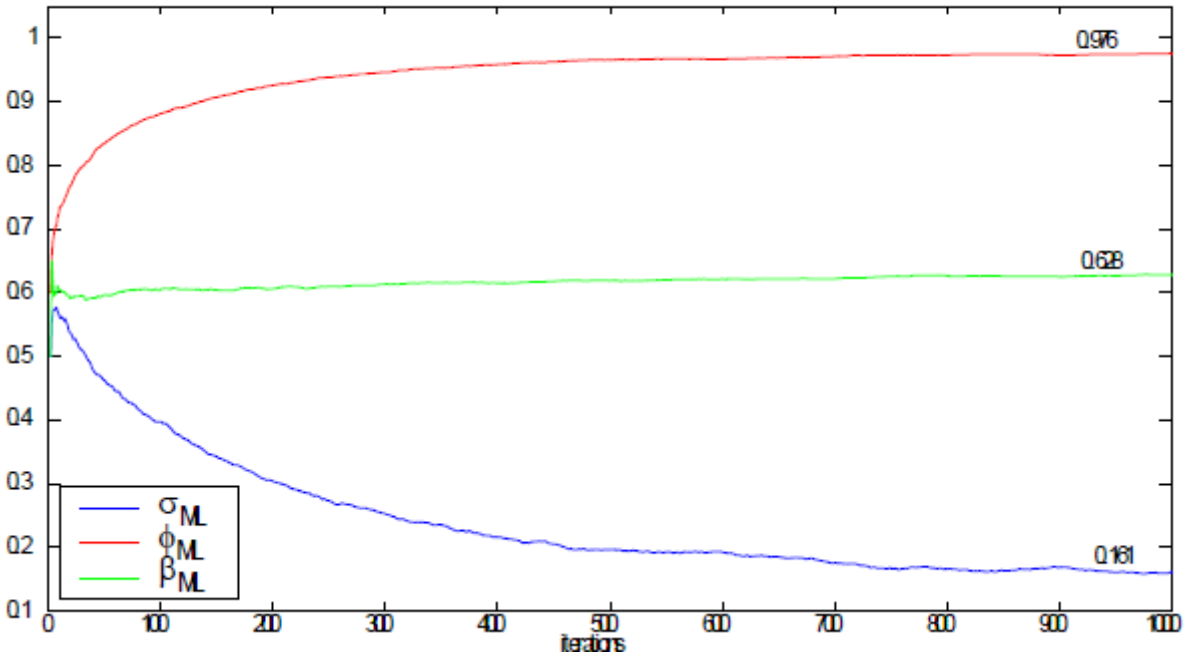
$$\theta_n = \theta_{n-1} + \gamma_n \widehat{\nabla \log p}_{\theta_{n-1}}(\boldsymbol{Y}_{1:n})$$

- For online estimation:

$$\theta_n = \theta_{n-1} + \gamma_n \widehat{\nabla \log p}_{\boldsymbol{\theta}_{1:n-1}}(Y_n | \boldsymbol{Y}_{1:n-1})$$

Example: Stochastic Volatility Model

- Batch Parameter Estimation for Daily Exchange Rate Pound/Dollar 81-85



Example: Stochastic Volatility Model

- Recursive parameter estimation for SV model. The estimates converge towards the true values.