

From Kalman to Particle Filtering

Eric Moulines

Centre de Mathématiques Appliquées

October 9, 2019

bssm: Bayesian Inference of Non-linear and Non-Gaussian State Space Models in R

Jouni Helske and Matti Vihola, University of Jyväskylä, Department of Mathematics and Statistics, Finland

September 23, 2019

Introduction

State space models (SSM) are latent variable models which are commonly applied in analysing time series data due to their flexible and general framework (cf. Durbin and Koopman 2012). For R (R Core Team 2016), there is large number of packages available for state space modelling, especially for the two special cases. First special case is linear-Gaussian SSM (LGSSM) where both the observation and state densities are Gaussian with linear relationships with the states. Another special case is SSM with discrete state space, which are sometimes called hidden Markov models (HMM). We do not consider HMMs in this paper. What is special about these two class of models is that the marginal likelihood function, and the conditional state distributions (conditioned on the observations) of these models are analytically tractable, making inference relatively straightforward. See for example (Petrис and Petrone 2011, @Tusell2010, @helske2017, @HelskeHelske2017) for review of some of the R packages dealing with these type of models. The R package `bssm` is designed for Bayesian inference of general state space models with non-Gaussian and/or non-linear observational and state equations. The package aims to provide easy-to-use and efficient functions for fully Bayesian inference of common time series models such basic structural time series model (BSM) (Harvey 1989) with exogenous covariates, simple stochastic volatility models, and discretized diffusion models, making it straightforward and efficient to make predictions and other inference in a Bayesian setting.

Package bssm

Part I

State-Space Models and Filtering

1 State-Space Models

- The Functional Representation
- The Markovian Representation

2 The Linear Prediction Approach

- A Very Simple (Static) Example
- The General Kalman Filter
- Virtues and Limitations

3 The Bayesian Approach

- The General Filtering Recursion
- The Linear Gaussian State-Space Case
- The Finite State Space Case

4 Approximate Gaussian Filters

- The Extended Kalman Filter (EKF)
- The Unscented Kalman Filter (UKF)

Roadmap

1 State-Space Models

- The Functional Representation
- The Markovian Representation

2 The Linear Prediction Approach

- A Very Simple (Static) Example
- The General Kalman Filter
- Virtues and Limitations

3 The Bayesian Approach

- The General Filtering Recursion
- The Linear Gaussian State-Space Case
- The Finite State Space Case

4 Approximate Gaussian Filters

- The Extended Kalman Filter (EKF)
- The Unscented Kalman Filter (UKF)

State-Space Models

State-space modelling is widely used in

- Times series analysis
- Signal processing
- Machine learning

The model features a latent (or unobserved) state process and is specified by a pair of recurrence relations.

Alternatively it can be defined using the Markov property, hence the name **Hidden Markov Model (HMM)** also used to refer to this approach.

The Functional Representation: (1) State Process

State Evolution (or Dynamic) Equation

$$X_k = a_{k-1}(X_{k-1}, U_{k-1}),$$

where,

- 1 $a_k(\cdot, \cdot)$ is a (possibly vector-valued) function that may be time-varying, hence the subscript $k - 1$;
- 2 the dynamic noise $\{U_k\}_{k \geq 0}$ is independent and identically distributed (i.i.d.);
- 3 the initial state X_0 is distributed under a prior distribution ν .

The Functional Representation: (2) Observations

Measurement Equation

$$Y_k = b_k(X_k, V_k) ,$$

where,

- 1 $b_k(\cdot, \cdot)$ is another vector-valued function that can also be time varying^{*};
- 2 the measurement noise $\{V_k\}_{k \geq 0}$ is i.i.d. and independent from $\{U_k\}$ and X_0 .

*Note that a_k and b_k could also depend of previous observations Y_1, \dots, Y_{k-1} , which is not considered here.

The Linear State-Space Model

Unless otherwise specified, the term **State-Space Model** is most often used for **linear models** of the form^{*}:

$$\begin{aligned} X_k &= A_{k-1}X_{k-1} + R_{k-1}U_{k-1}, \\ Y_k &= B_kX_k + S_kV_k. \end{aligned}$$

it is often assumed (though not necessary) that X_0 , $(U_k)_{k \geq 0}$ and $(V_k)_{k \geq 0}$ are Gaussian.

^{*}One can also include deterministic terms (trends, non-zero means) that are omitted here for simplicity.

Markovian Representation

Equivalently, the model may be described in terms of the structure of the distribution of the joint process $(X_k, Y_k)_{k \geq 0}$.

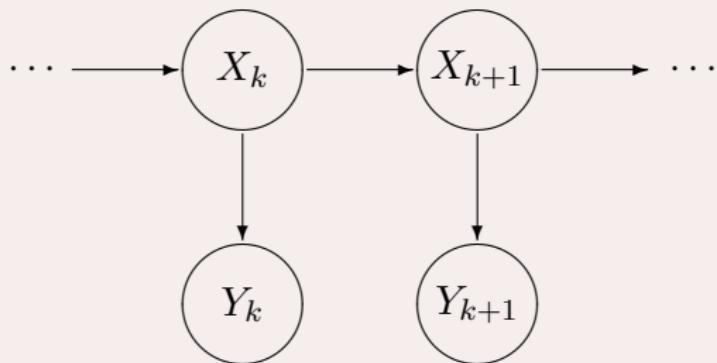
$(X_k, Y_k)_{k \geq 0}$ is a Markov process such that **the conditional distribution of (X_k, Y_k) given $(X_i, Y_i)_{0 \leq i \leq k-1}$ only depends on X_{k-1}** .

Markovian Representation

Equivalently, the model may be described in terms of the structure of the distribution of the joint process $(X_k, Y_k)_{k \geq 0}$.

$(X_k, Y_k)_{k \geq 0}$ is a Markov process such that **the conditional distribution of (X_k, Y_k) given $(X_i, Y_i)_{0 \leq i \leq k-1}$ only depends on X_{k-1}** .

Bayesian Network (Directed Graphical Model) Representation



Transition Kernel

- The conditional distribution is conveniently represented by a **Markov kernel**, Q_k , defined by

$$\Pr(X_{k+1} \in A \mid X_k = x_k) = Q_k(x_k, A).$$

In words, $Q_k(x_k, A)$ is the probability of transitioning from x_k to a (measurable) set A .

Transition Kernel

- The conditional distribution is conveniently represented by a **Markov kernel**, Q_k , defined by

$$\Pr(X_{k+1} \in A | X_k = x_k) = Q_k(x_k, A).$$

In words, $Q_k(x_k, A)$ is the probability of transitioning from x_k to a (measurable) set A .

- In most situations of interest, there exists a **transition density function** q_k such that

$$Q_k(x, A) = \int_A q_k(x, x') dx' , \text{ for all } x \in \mathcal{X} \text{ and } A \in \mathcal{X}.$$

In words, $q_k(X_k, \cdot)$ is the conditional density of X_{k+1} given X_k .

Conditional Measurement Likelihood

- Similarly,

$$\Pr(Y_k \in A | X_k = x_k) = G_k(x_k, A) = \int_A g_k(x_k, y) dy ,$$

where $g_k(X_k, \cdot)$ is the conditional density of Y_k given X_k .

Conditional Measurement Likelihood

- Similarly,

$$P(Y_k \in A | X_k = x_k) = G_k(x_k, A) = \int_A g_k(x_k, y) dy ,$$

where $g_k(X_k, \cdot)$ is the conditional density of Y_k given X_k .

- Together these two equations define the conditional density of (X_k, Y_k) given $(X_i, Y_i)_{0 \leq i \leq k-1}$ as

$$q_{k-1}(X_{k-1}, x_k) g_k(x_k, y_k) .$$

It is easy to relate the **state-space equations** to the **Markovian representation**; the latter is useful for defining the filtering equations in the general case.

Example (Stochastic Volatility Model)

$$X_k = \phi X_{k-1} + \sigma U_{k-1} \quad U_{k-1} \sim \mathcal{N}(0, 1)$$

$$Y_k = \beta \exp(X_k/2) V_k \quad V_k \sim \mathcal{N}(0, 1)$$

$$q(x_{k-1}, x_k) = \mathcal{N}(x_k; \phi x_{k-1}, \sigma^2),$$

$$= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x_k - \phi x_{k-1})^2}{2\sigma^2}\right].$$

$$g(x_k, y_k) = \mathcal{N}(y_k; 0, \beta^2 \exp(x_k)),$$

$$= \frac{1}{\sqrt{2\pi\beta^2}} \exp\left[-\frac{y_k^2}{2\beta^2} \exp(-x_k) - \frac{1}{2}x_k\right].$$

Filtering

- **Filtering** relates to the task of inferring the latent (hidden) state X_k from the observations $Y_{0:k} = Y_0, \dots, Y_k$.
- The filtering problem may be solved by **recursion on k** .
- The filtering recursion can be determined by two approaches:
 - 1 The L^2 or **linear prediction** approach in the case of **linear state-space models** only.
 - 2 The **Bayesian** approach, which is more general.

Filtering

- Filtering relates to the task of inferring the latent (hidden) state X_k from the observations $Y_{0:k} = Y_0, \dots, Y_k$.
- The filtering problem may be solved by recursion on k .
- The filtering recursion can be determined by two approaches:
 - 1 The L^2 or linear prediction approach in the case of linear state-space models only.
 - 2 The Bayesian approach, which is more general.
- The Bayesian approach builds on the Markovian representation while the linear prediction approach only uses the state-space representation.
- The linear prediction equations are recovered in the Bayesian approach when assuming a linear state-space model with Gaussian noises.

Roadmap

1 State-Space Models

- The Functional Representation
- The Markovian Representation

2 The Linear Prediction Approach

- A Very Simple (Static) Example
- The General Kalman Filter
- Virtues and Limitations

3 The Bayesian Approach

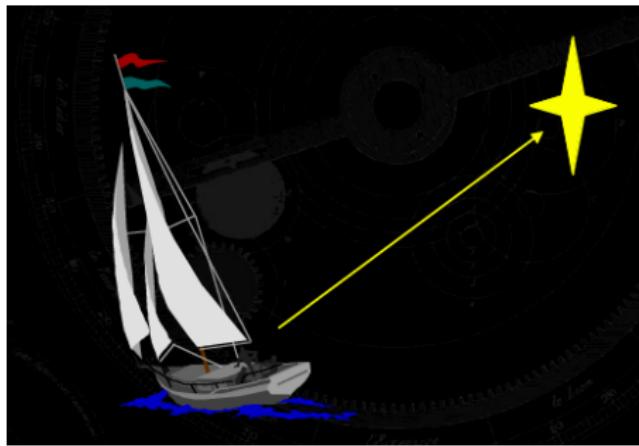
- The General Filtering Recursion
- The Linear Gaussian State-Space Case
- The Finite State Space Case

4 Approximate Gaussian Filters

- The Extended Kalman Filter (EKF)
- The Unscented Kalman Filter (UKF)

A very simple example

To understand the main argument, consider first a basic model with **static state** $X_k = X$.



Measurement

- We get a first measurement: $Y_1 = X + V_1$, where
 - 1 X is the unknown position of the ship
 - 2 V_1 is the **measurement** error assumed to be zero mean $E[V_1] = 0$ with finite variance $E[V_1^2] = \sigma_1^2$.
- In a Bayesian setting, the unknown ship position is modeled as a random variable. The distribution of this random variable traduces the a priori information presents before obtaining the first measurement. This distribution is referred to as the **prior distribution**.
- We assume here that $E[X] = 0$ and $E[X^2] = \Sigma_0$. A large value of Σ_0 means that we have little prior knowledge on the initial position.
- In addition X is assumed to be **uncorrelated** from the observation noise.

The optimal linear mean square error estimator

- We have two sources of information: the **prior information** and the **measurement**. How to fuse in a statistically sound way these information?
- The **optimal linear mean square error estimator** of the true position is $\hat{X}_{1|1} = \hat{a}_1 Y_1$ where \hat{a}_1 is chosen to minimize the MSE

$$\hat{a}_1 = \operatorname{argmin}_a \text{MSE}(a) \quad \text{MSE}(a) \stackrel{\text{def}}{=} \mathbb{E}[(X - aY_1)^2].$$

- $\hat{X}_{1|1}$ minimizes the variance of the prediction error.

Solving the optimal linear MSE problem

- The function $a \mapsto \text{MSE}(a)$ is quadratic in a . This function has a unique minimum characterized by

$$\frac{d\text{MSE}(a)}{da} = 0 \Leftrightarrow \hat{a}_1 = \frac{\text{E}[XY_1]}{\text{E}[Y_1^2]}.$$

- The optimal linear MSE estimator is thus given by

$$\hat{a}_1 = \frac{\text{E}[X^2]}{\text{E}[X^2] + \text{E}[V_1^2]} = \frac{\Sigma_0}{\Sigma_0 + \sigma_1^2}.$$

Computing the variance of the estimation error

- Using that $E[(X - \hat{a}_1 Y_1)Y_1] = 0$, the error variance is given by

$$\begin{aligned}\Sigma_{1|1} &= E[(X - \hat{a}_1 Y_1)^2] = E[(X - \hat{a}_1 Y_1)X] = (1 - \hat{a}_1)\Sigma_0 \\ &= \frac{\sigma_1^2 \Sigma_0}{\sigma_1^2 + \Sigma_0},\end{aligned}$$

showing that the observation (as expected) reduces the uncertainty on the state position.

- If $\Sigma_0 \gg \sigma_1^2$, the error variance is almost equal to the measurement error variance: the prior information is diffuse and Y_1 is the best bet that we can make.

A geometric interpretation

- The space of square integrable random variables over a given probability space (Ω, \mathcal{F}, P) , equipped with the scalar product $\langle Z_1, Z_2 \rangle = E(Z_1 Z_2)$ is an **Hilbert space**. This space is denoted $L^2(\Omega, \mathcal{F}, P)$.
- The associated norm is denoted $\|Z\|^2 = \langle Z, Z \rangle$.

Projection

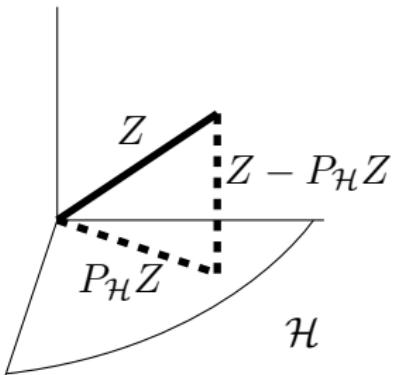
- Let \mathcal{H} be a (closed) linear subspace of $L^2(\Omega, \mathcal{F}, P)$ and $Z \in L^2(\Omega, \mathcal{F}, P)$ be a vector. The **projection** of Z onto \mathcal{H} is the unique vector $P_{\mathcal{H}}Z$ belonging to \mathcal{H} satisfying

$$\|Z - P_{\mathcal{H}}Z\|^2 \leq \|Z - T\|^2 \quad \text{for all } T \in \mathcal{H}.$$

- The projection is characterized by the **orthogonality relation**

$$\langle Z - P_{\mathcal{H}}Z, T \rangle = 0 \quad \text{for all } T \in \mathcal{H}.$$

- The projector $P_{\mathcal{H}}$ is a **linear** operator over $L^2(\Omega, \mathcal{F}, P)$.



The optimal linear MSE as a projection

- The **optimal linear mean-square estimator** of X based on Y_1 is the **projection** of the random variable X on the one-dimensional subspace spanned by the vector Y_1 ,

$$\hat{X}_{1|1} = P_{\text{span}(Y_1)} X$$

where $\text{span}(Y_1)$ is the one-dimensional space of random variables colinear to Y_1 ,

$$\text{span}(Y_1) \stackrel{\text{def}}{=} \{Z : Z = aY_1\} .$$

- Computing the projection on a one-dimensional subspace consists in computing the scalar product and dividing by the norm:

$$P_{\text{span}(Y_1)} X = \frac{\langle Y_1, X \rangle}{\|Y_1\|^2} Y_1 .$$

A second measurement

- Assume that we have at hand a second measurement $Y_2 = X + V_2$, where V_2 is uncorrelated with X and V_1 .
- The **optimal linear mean square error estimator** is the **orthogonal projection** onto the space spanned by the random variables Y_1, Y_2 ,

$$\hat{X}_{2|2} = P_{\text{span}(Y_1, Y_2)} X ,$$
$$\text{span}(Y_1, Y_2) \stackrel{\text{def}}{=} \{Z : Z = a_1 Y_1 + a_2 Y_2\} .$$

- This time we have to compute the projection on a two-dimensional subspace. This operation is a little bit less straightforward because Y_1 and Y_2 are not orthogonal!

How to compute the projection?

- To compute the projection, it is worthwhile to construct an orthogonal basis of $\text{span}(Y_1, Y_2)$,

$$\text{span}(Y_1, Y_2) = \text{span}(Y_1) \oplus \text{span}(\epsilon_2)$$

where $\epsilon_2 = Y_2 - P_{\text{span}(Y_1)}Y_2$ is the **innovation**. Indeed,

$$P_{\text{span}(Y_1, Y_2)}Z = P_{\text{span}(Y_1)}Z + \|\epsilon_2\|^{-2}\langle Z, \epsilon_2 \rangle \epsilon_2 .$$

How to compute the innovation?

- Since the projection is a linear operator and $Y_2 = X + V_2$,

$$P_{\text{span}(Y_1)} Y_2 = P_{\text{span}(Y_1)} X + P_{\text{span}(Y_1)} V_2 .$$

- Since V_2 is uncorrelated with Y_1 ,

$$\langle V_2, Y_1 \rangle = 0$$

and the orthogonality characterization implies that

$$P_{\text{span}(Y_1)} V_2 = 0.$$

- Hence, the innovation writes

$$\epsilon_2 = Y_2 - P_{\text{span}(Y_1)} X = Y_2 - \hat{X}_{1|1} .$$

- The variance of the innovation is given by

$$\|\epsilon_2\|^2 = \Sigma_{1|1} + \sigma_2^2 .$$

Update equation

- Hence, the optimal linear MSE estimator can be computed as

$$\hat{X}_{2|2} = \hat{X}_{1|1} + K_2 \epsilon_2 ,$$

where K_2 is the **Kalman gain** defined as

$$\begin{aligned} K_2 &= \|\epsilon_2\|^{-2} \langle \epsilon_2, X \rangle \\ &= (\Sigma_{1|1} + \sigma_2^2)^{-1} \langle X - \hat{X}_{1|1} + V_2, X \rangle \\ &= (\Sigma_{1|1} + \sigma_2^2)^{-1} \Sigma_{1|1} . \end{aligned}$$

- $K_2 \epsilon_2$ is the correction to the **optimal linear MSE** which is associated to the observation Y_2 and K_2 is the **magnitude** of the correction.

Riccati's equations

- To complete the recursion, it is required to update sequentially the variance of the error

$$\Sigma_{2|2} \stackrel{\text{def}}{=} \|X - \hat{X}_{2|2}\|^2 .$$

- The orthogonality principle implies that $\langle X - \hat{X}_{2|2}, \hat{X}_{2|2} \rangle = 0$. Hence

$$\begin{aligned}\Sigma_{2|2} &= \langle X - \hat{X}_{1|1} - K_2 \epsilon_2, X \rangle \\ &= \Sigma_{1|1} - K_2 \langle X - \hat{X}_{1|1}, X \rangle \\ &= (1 - K_2) \Sigma_{1|1} .\end{aligned}$$

- In this case,

$$\Sigma_{2|2}^{-1} = \Sigma_{1|1}^{-1} + \sigma_2^{-2} .$$

Recap

Starting from the estimate obtained after the first measurement:

$$\hat{X}_{1|1}, \Sigma_{1|1}$$

Recap

Starting from the estimate obtained after the first measurement:

$$\hat{X}_{1|1}, \Sigma_{1|1}$$

- Innovation : $\epsilon_2 = Y_2 - \hat{X}_{1|1}$

Recap

Starting from the estimate obtained after the first measurement:

$$\hat{X}_{1|1}, \Sigma_{1|1}$$

- Innovation : $\epsilon_2 = Y_2 - \hat{X}_{1|1}$
- Kalman gain computation: $K_2 = \Sigma_{1|1} (\Sigma_{1|1} + \sigma_2^2)^{-1}$

Recap

Starting from the estimate obtained after the first measurement:

$$\hat{X}_{1|1}, \Sigma_{1|1}$$

- Innovation : $\epsilon_2 = Y_2 - \hat{X}_{1|1}$
- Kalman gain computation: $K_2 = \Sigma_{1|1} (\Sigma_{1|1} + \sigma_2^2)^{-1}$
- State estimate update: $\hat{X}_{2|2} = \hat{X}_{1|1} + K_2 \epsilon_2$

Recap

Starting from the estimate obtained after the first measurement:

$$\hat{X}_{1|1}, \Sigma_{1|1}$$

- Innovation : $\epsilon_2 = Y_2 - \hat{X}_{1|1}$
- Kalman gain computation: $K_2 = \Sigma_{1|1} (\Sigma_{1|1} + \sigma_2^2)^{-1}$
- State estimate update: $\hat{X}_{2|2} = \hat{X}_{1|1} + K_2 \epsilon_2$
- Error covariance update: $\Sigma_{2|2} = (I - K_2) \Sigma_{1|1}$.

These recursions define a basic instance of the **Kalman filter equations**.

Back to the General Case

State Equation

The **state evolution (process) equation** models an unknown physical stochastic phenomenon as the output of a linear dynamical state excited by white noise:

$$X_k = A_{k-1}X_{k-1} + R_{k-1}U_{k-1} .$$

- X_k : state at time k
- A_{k-1} : transition matrix (taking the state at time $k - 1$ to the new state at time k)
- U_{k-1} : state noise at time $k - 1$.

Measurement Equation

The **measurement equation** describes the dependence of the observations on the state:

$$Y_k = B_k X_k + S_k V_k .$$

- Y_k : observation vector at time k ,
- B_k : measurement matrix at time k ,
- V_k : measurement noise at time k .

Applications

- 1 Satellite-orbit estimation, tracking radar
- 2 Aided inertial navigation system using global positioning system (GPS)
- 3 Lots of computer vision applications (extracting lip motions, etc)
- 4 Channel estimation in wireless communications
- 5 Finance, econometry
- 6 ...

Statement

Given

1 prescribed initial conditions, and

2 the entire sequence of observations $Y_{0:n} = Y_0, \dots, Y_n$

estimate the minimum mean-square error estimate $\hat{X}_{k|n}$ and the associated error covariance $\Sigma_{k|n}$ of the state X_k at time k .

Statement

Given

1 prescribed initial conditions, and

2 the entire sequence of observations $Y_{0:n} = Y_0, \dots, Y_n$

estimate the minimum mean-square error estimate $\hat{X}_{k|n}$ and the associated error covariance $\Sigma_{k|n}$ of the state X_k at time k .

Different goals:

- **Prediction:** $k > n$,
- **Filtering:** $k = n$,
- **Smoothing:** $0 \leq k < n$.

Assumptions

$$\begin{aligned}X_{k+1} &= A_k X_k + R_k U_k , \\Y_k &= B_k X_k + S_k V_k .\end{aligned}$$

- 1 **White Noise:** $\{U_k\}_{k \geq 0}$ and $\{V_k\}_{k \geq 0}$ are white noise.
- 2 **Decorrelation:** $\{U_k\}_{k \geq 0}$ and $\{V_k\}_{k \geq 0}$ are uncorrelated.
- 3 **Initial Conditions:** X_0 has prior mean $\hat{X}_{0|-1}$ (the initial guess) and covariance $\Sigma_{0|-1}$ and is uncorrelated with $\{U_k\}_{k \geq 0}$ and $\{V_k\}_{k \geq 0}$.

Kalman Filter principle

- An estimate $\hat{X}_{k|k}$ of the state X_k is to be computed from the data $Y_{0:k}$ so as to minimize the mean-square error in the estimate.
- The Kalman filter recursion computes $\hat{X}_{k|k}$ as an explicit function **only** of
 - 1 the measurement Y_k ,
 - 2 the previous estimate $\hat{X}_{k-1|k-1}$ and
 - 3 the covariance of the state-estimation error
 $\Sigma_{k-1|k-1} = \text{Cov}(X_k - \hat{X}_{k|k-1})$.

Properties

- The Kalman filter is the **optimal mean-square error tracker of the hidden state of a linear dynamical state-space model.**
- This approach leads to a recursive solution that provides an estimate which is equivalent to the estimate obtained by processing all the data simultaneously.

Properties

- The Kalman filter is the **optimal mean-square error tracker of the hidden state of a linear dynamical state-space model.**
- This approach leads to a recursive solution that provides an estimate which is equivalent to the estimate obtained by processing all the data simultaneously.
- If both the dynamic noise, the measurement noise and the initial state are jointly Gaussian distributed, then the Kalman filter is also optimal in the Bayesian (maximum a posteriori) sense: the filtering distribution of the state X_k given the observation Y_0, \dots, Y_k , denoted $\phi_{k|k}$, is Gaussian with mean $\hat{X}_{k|k}$ and covariance $\Sigma_{k|k}$,

$$\phi_{k|k} = \mathcal{N}(\hat{X}_{k|k}, \Sigma_{k|k}).$$

Prediction

Use the state equation to compute

- 1 the predicted state $\hat{X}_{k|k-1}$

$$\hat{X}_{k-1|k-1} \rightarrow \hat{X}_{k|k-1} = A_{k-1} \hat{X}_{k-1|k-1},$$

Prediction

Use the state equation to compute

- 1 the predicted state $\hat{X}_{k|k-1}$

$$\hat{X}_{k-1|k-1} \rightarrow \hat{X}_{k|k-1} = A_{k-1} \hat{X}_{k-1|k-1} ,$$

- 2 the prediction error covariance

$$\Sigma_{k-1|k-1} \rightarrow \Sigma_{k|k-1} = A_{k-1} \Sigma_{k-1|k-1} A_{k-1}^t + R_{k-1} R_{k-1}^t .$$

Innovation

- Use the observation Y_k and the measurement equation to compute the forward prediction error termed the **innovation** and its covariance

$$\epsilon_k = Y_k - B_k \hat{X}_{k|k-1} \quad \Gamma_k = B_k \Sigma_{k|k-1} B_k^t + S_k S_k^t .$$

Innovation

- Use the observation Y_k and the measurement equation to compute the forward prediction error termed the **innovation** and its covariance

$$\epsilon_k = Y_k - B_k \hat{X}_{k|k-1} \quad \Gamma_k = B_k \Sigma_{k|k-1} B_k^t + S_k S_k^t .$$

- The innovation is another way to represent the observations by retaining only the part which cannot be predicted linearly from the past.
 - Principle of orthogonality: $E[\epsilon_k Y_l^t] = 0$, for $l = 0, \dots, k-1$.
 - Orthogonalization: $E[\epsilon_k \epsilon_l^t] = 0$, for $l = 0, \dots, k-1$.
 - Information preservation:

$$\{Z : Z = \sum_{i=0}^k \alpha_i Y_i\} = \{Z : Z = \sum_{i=0}^k \beta_i \epsilon_i\} .$$

Correction

- Use the innovation to update (*i.e.*, correct) the minimum mean-square estimate of the state related linearly to the observables,

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k \epsilon_k ,$$

where K_k is the so-called **Kalman Gain**.

- Since $\hat{X}_{k|k-1}$ and I_k are **orthogonal**, $E[\hat{X}_{k|k-1} \epsilon_k^t] = 0$, K_k is given by

$$\begin{aligned} K_k &= E[X_k \epsilon_k^t] (E[\epsilon_k \epsilon_k^t])^{-1} \\ &= \Sigma_{k|k-1} B_k^t (B_k \Sigma_{k|k-1} B_k^t + S_k S_k^t)^{-1} . \end{aligned}$$

- $K_k \epsilon_k$ is the incremental change applied to the *a priori* state estimate $\hat{X}_{k|k-1}$ to produce the *a posteriori* state estimate $\hat{X}_{k|k}$.

Riccati's equations

- The last step is to update the *a posteriori* state error covariance $X_k - \hat{X}_{k|k}$.
- Since $X_k - \hat{X}_{k|k}$ is orthogonal to $\hat{X}_{k|k}$,

$$\begin{aligned}\Sigma_{k|k} &= E \left((X_k - \hat{X}_{k|k})(X_k - \hat{X}_{k|k})^t \right) \\ &= \Sigma_{k|k-1} - K_k E(\epsilon_k (X_k - \hat{X}_{k|k-1})^t) \\ &= (I - K_k B_k) \Sigma_{k|k-1},\end{aligned}$$

which is referred to as the Riccati equation.

Summary

Kalman Filter algorithm

1 **State Predictor** $\hat{X}_{k|k-1} = A_{k-1}\hat{X}_{k-1|k-1}$

2 **Prediction Error Covariance**

$$\Sigma_{k|k-1} = A_{k-1}\Sigma_{k-1|k-1}A_{k-1}^t + R_{k-1}R_{k-1}^t$$

3 **Innovation** $\epsilon_k = Y_k - B_k\hat{X}_{k|k-1}$

4 **Kalman Gain** $K_k = \Sigma_{k|k-1}B_k^t (B_k\Sigma_{k|k-1}B_k^t + S_kS_k^t)^{-1}$

5 **State Filter** $\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k\epsilon_k$

6 **Filtering Error Covariance** $\Sigma_{k|k} = (I - K_kB_k)\Sigma_{k|k-1}$

Virtues

- Kalman filtering has established itself as an indispensable tool in the study of dynamical systems in both control and signal processing.
- It offers a highly elegant framework for state estimation, building on geometric intuitions and requiring only first and second order moment assumptions.
- When considering a physical system that is linear and governed by Gaussian distributions for both dynamic noise and measurement noise, the Kalman filter is also optimal in the Bayesian sense.

From Linear to General State-Space Models

- In many practical applications, either the linearity or the Gaussianity assumptions fail to be true.
- If the state-space model is still linear but the state measurement noise are non Gaussian, the Kalman filter can still be applied but can be far from being optimal. This is the case in particular when outliers are present in the observation noise or if the distribution.
- If the state or measurement equations are non-linear, the situation is even worse, since the Kalman filter can no longer be applied as it stands.
- There are however situations where deviations from the linear state-space case are mild: in this case, the overall structure of the KF can be kept to develop **approximate filtering algorithms**.

Roadmap

1 State-Space Models

- The Functional Representation
- The Markovian Representation

2 The Linear Prediction Approach

- A Very Simple (Static) Example
- The General Kalman Filter
- Virtues and Limitations

3 The Bayesian Approach

- The General Filtering Recursion
- The Linear Gaussian State-Space Case
- The Finite State Space Case

4 Approximate Gaussian Filters

- The Extended Kalman Filter (EKF)
- The Unscented Kalman Filter (UKF)

The Bayesian Approach

We will now use the Markovian representation of the model:

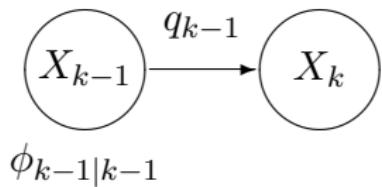
- $q_k(X_k, \cdot)$ is the conditional density of X_{k+1} given X_k ,
- $g_k(X_k, \cdot)$ is the conditional density of Y_k given X_k .

Non-Linear Filtering principle

- The **filtering density** $\phi_{k|k}$ corresponding the posterior distribution of the state X_k given the observations $Y_{0:k} = Y_0, \dots, Y_k$, is to be computed.
- This posterior distribution can be computed as an explicit function **only** of
 - 1 the observation Y_k ,
 - 2 the previous filtering density $\phi_{k-1|k-1}$.
- The problem parallels the Kalman filtering problem and can be solved using a **related prediction-correction** mechanism.

Prediction

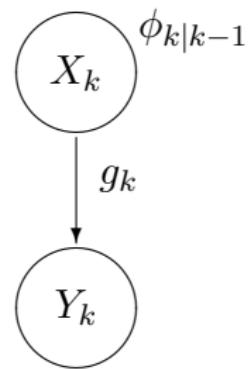
- Use the state equation to compute the density $\phi_{k|k-1}$ of the **predictive distribution**, that is, the posterior distribution of X_k given $Y_{0:k-1}$:



$$\phi_{k|k-1}(x_k) = \int \phi_{k-1|k-1}(x_{k-1}) q_{k-1}(x_{k-1}, x_k) dx_{k-1}$$

Correction

- Use the current observation Y_k to compute the **filtering density** $\phi_{k|k}$ from the predictive density $\phi_{k|k-1}$ using Bayes' rule:
 - $\phi_{k|k-1}$ is the **prior** distribution for the state X_k ,
 - $g_k(X_k, \cdot)$ is the **conditional** distribution of the observation given the state,



Bayes' rule therefore implies that

$$\phi_{k|k}(x_k) = \frac{\phi_{k|k-1}(x_k)g_k(x_k, Y_k)}{\int \phi_{k|k-1}(x_k)g_k(x_k, Y_k)dx_k}.$$

Example (Non-linear Growth Model)

We consider a simple nonlinear time series model which has been used extensively in the literature for benchmarking numerical filtering techniques.

The state-space equations are as follows:

$$X_t = \frac{X_{t-1}}{2} + 25 \frac{X_{t-1}}{1 + X_{t-1}^2} + 8 \cos(1.2t) + U_{t-1},$$

$$Y_t = \frac{X_t^2}{20} + V_t,$$

where $U_t \sim \mathcal{N}(0, \sigma_u^2)$ and $V_t \sim \mathcal{N}(0, \sigma_v^2)$ and here $\sigma_u = 5$ and $\sigma_v = 2$ are considered fixed and known. The initial state distribution is $x_0 \sim \mathcal{N}(0, \sigma_u^2)$.

Non-linear Growth Model Contd.

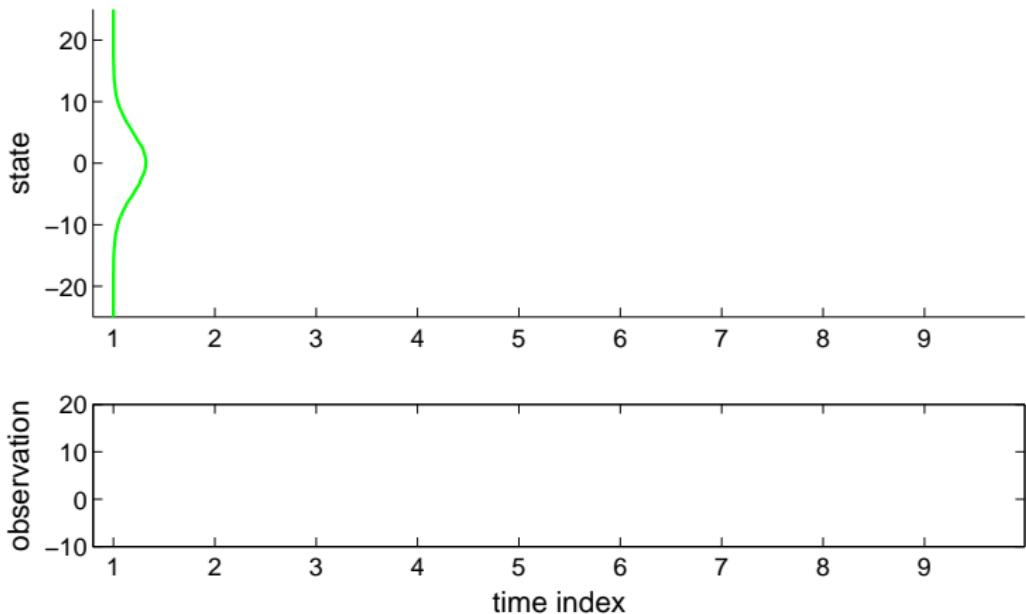
The representation of the model in terms of transition densities $q(x_{t-1}, x_t)$ and $g(x_t, y_t)$ is given by:

$$q(x_{t-1}, x_t) = \mathcal{N} \left(x_t ; \frac{x_{t-1}}{2} + 25 \frac{x_{t-1}}{1+x_{t-1}^2} + 8 \cos(1.2t), \sigma_u^2 \right)$$

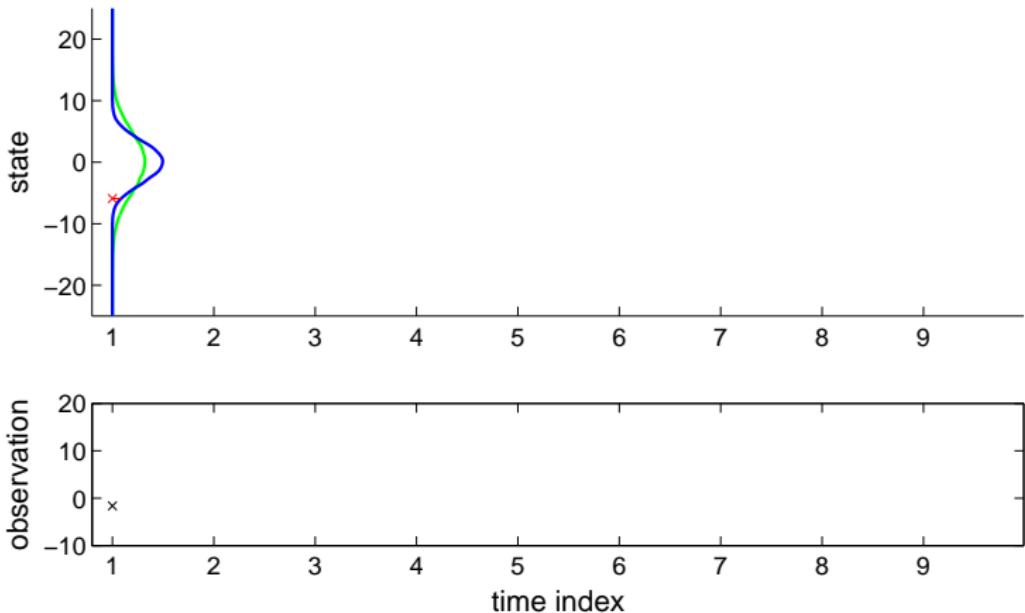
$$g(x_t, y_t) = \mathcal{N} \left(y_t ; \frac{x_t^2}{20}, \sigma_v^2 \right)$$

The form of these densities was straightforward to obtain in this simple case.

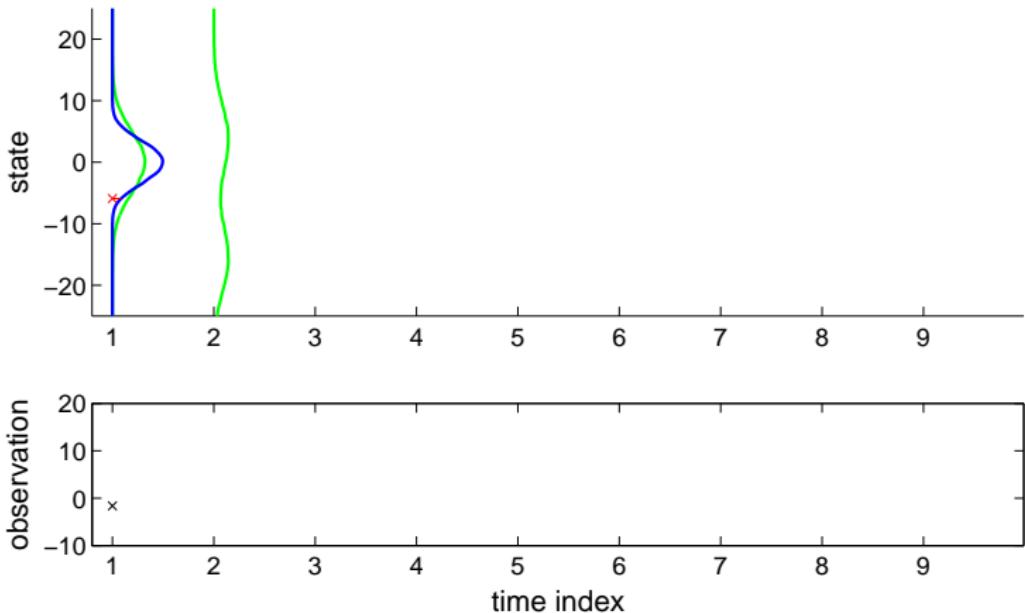
For more complex cases, a Jacobian term might be required when either X_t or Y_t is no more additive in U_t or V_t , respectively.



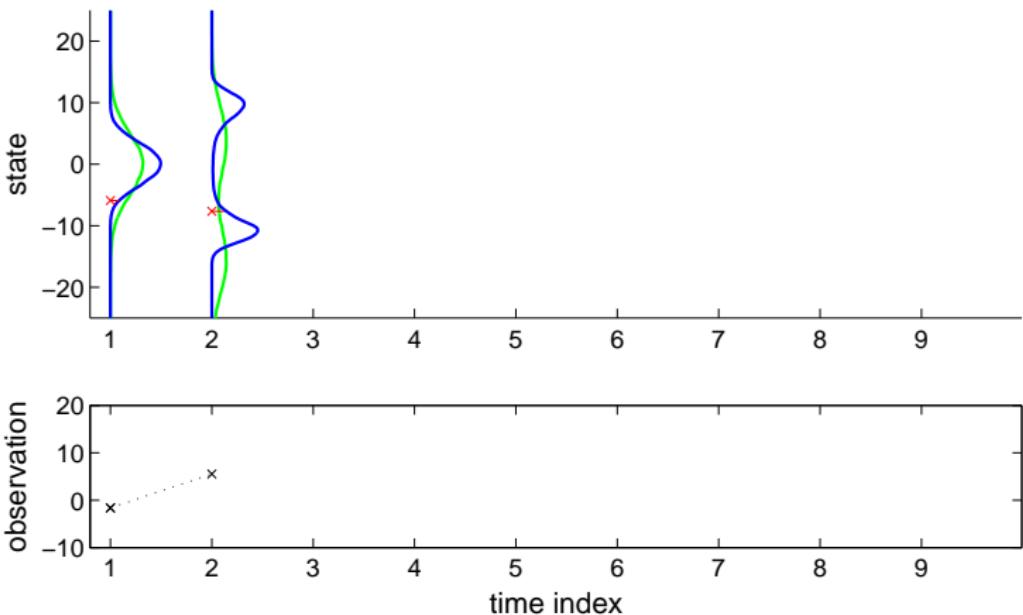
Predictive and filtering densities in the non-linear growth model



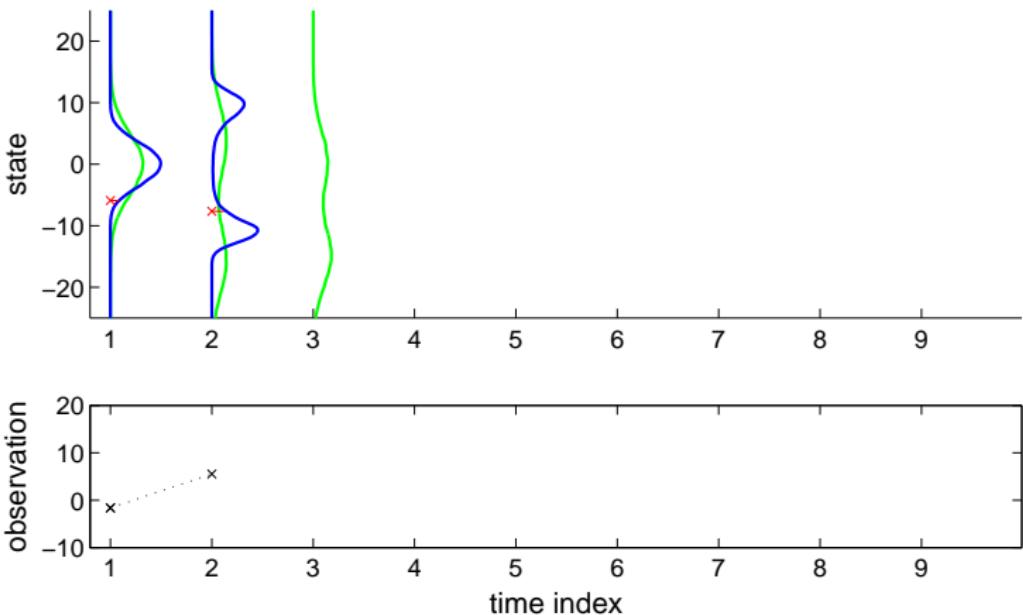
Predictive and filtering densities in the non-linear growth model



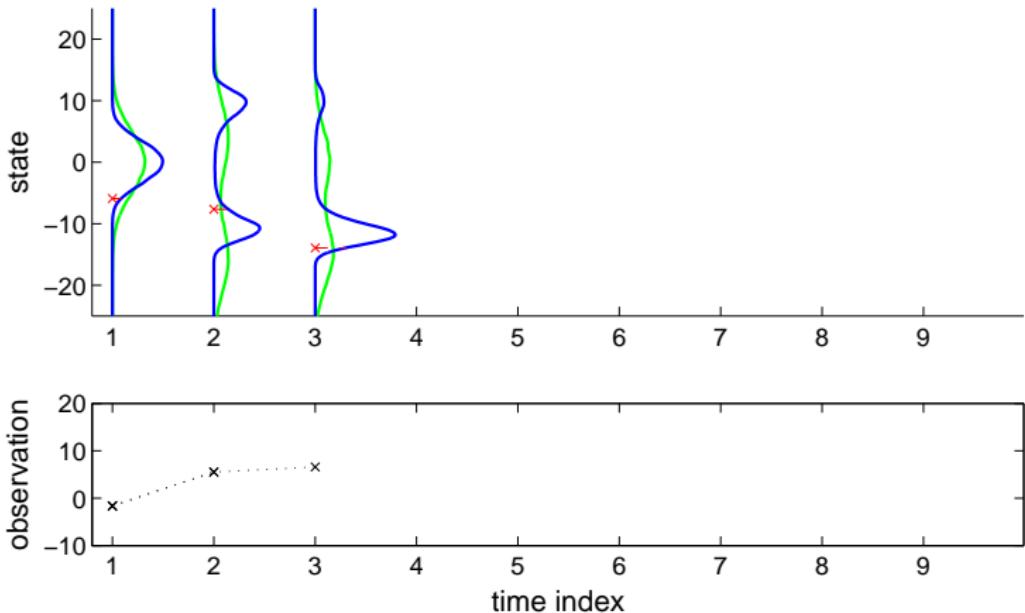
Predictive and filtering densities in the non-linear growth model



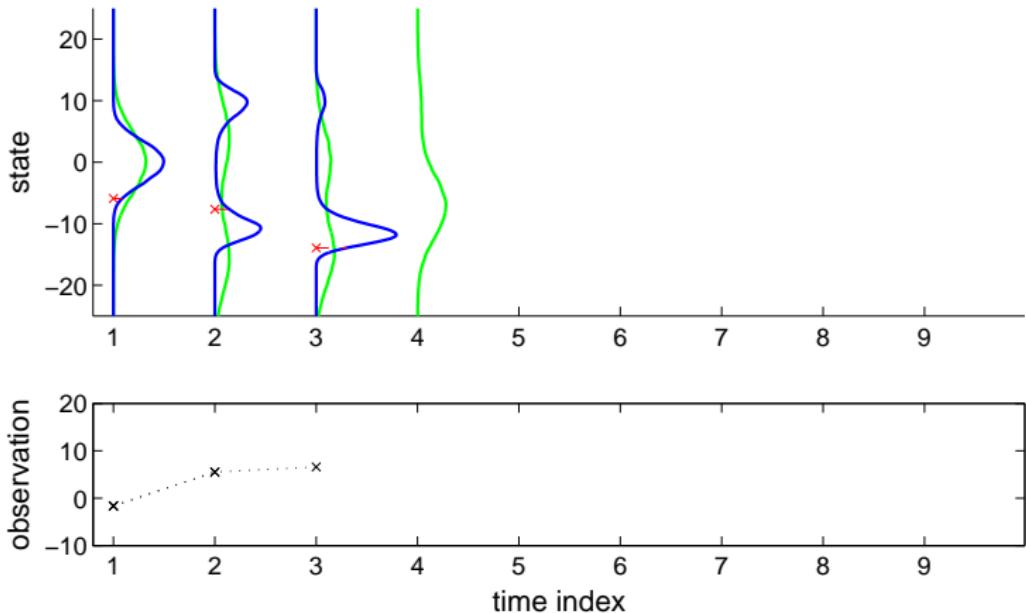
Predictive and filtering densities in the non-linear growth model



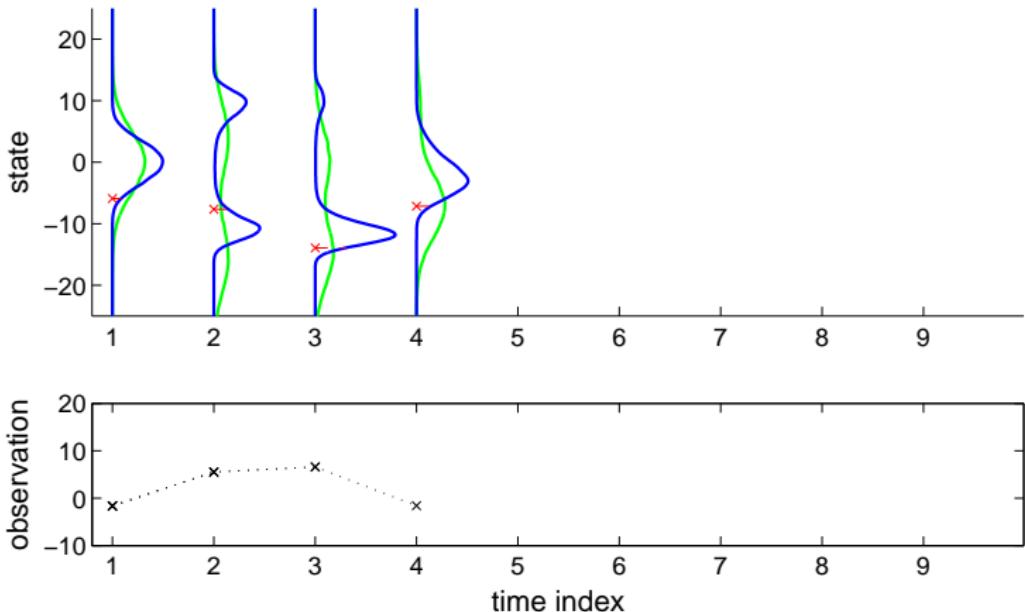
Predictive and filtering densities in the non-linear growth model



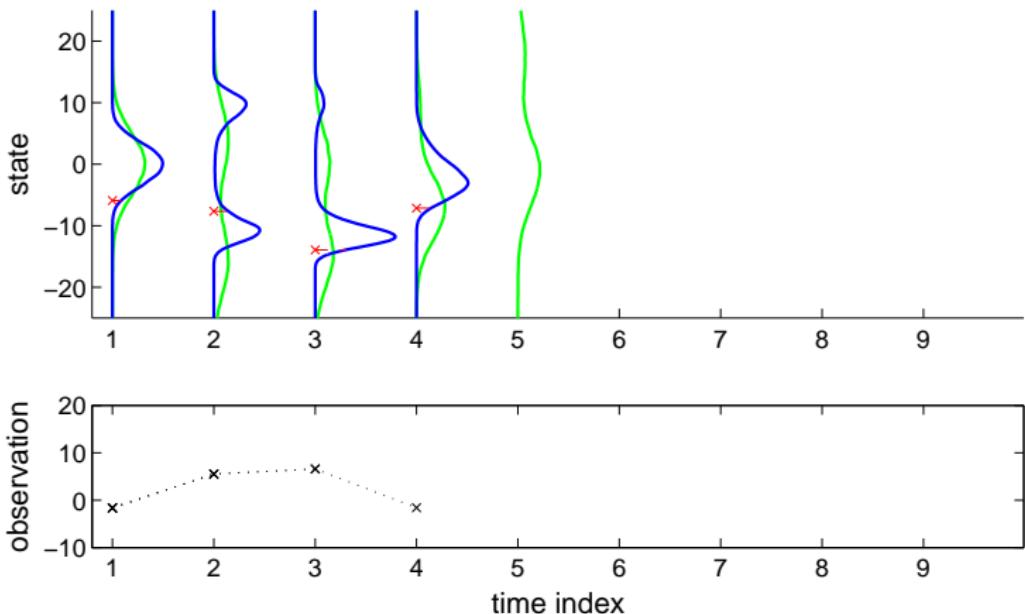
Predictive and filtering densities in the non-linear growth model



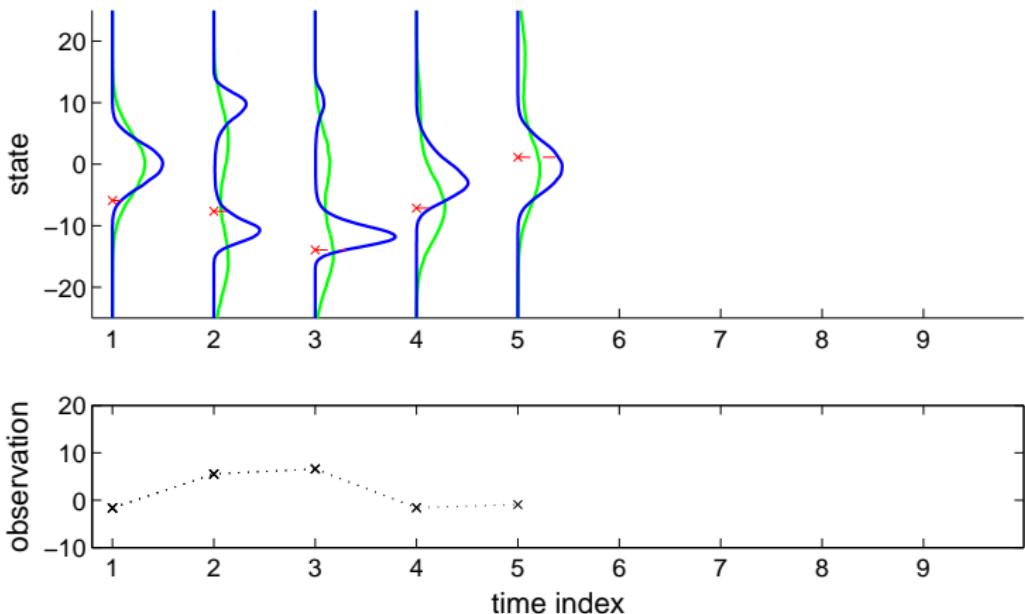
Predictive and filtering densities in the non-linear growth model



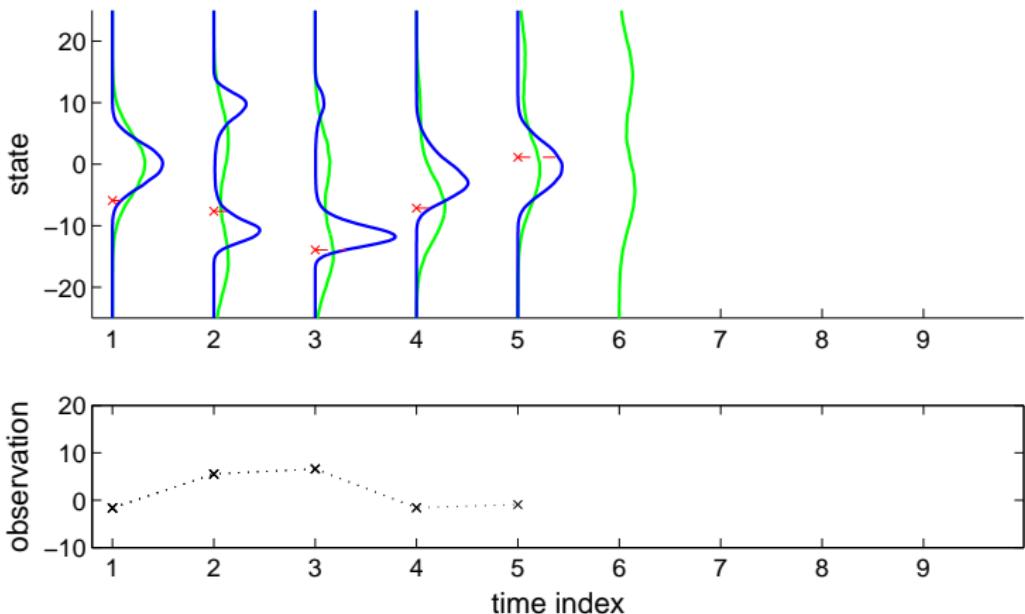
Predictive and filtering densities in the non-linear growth model



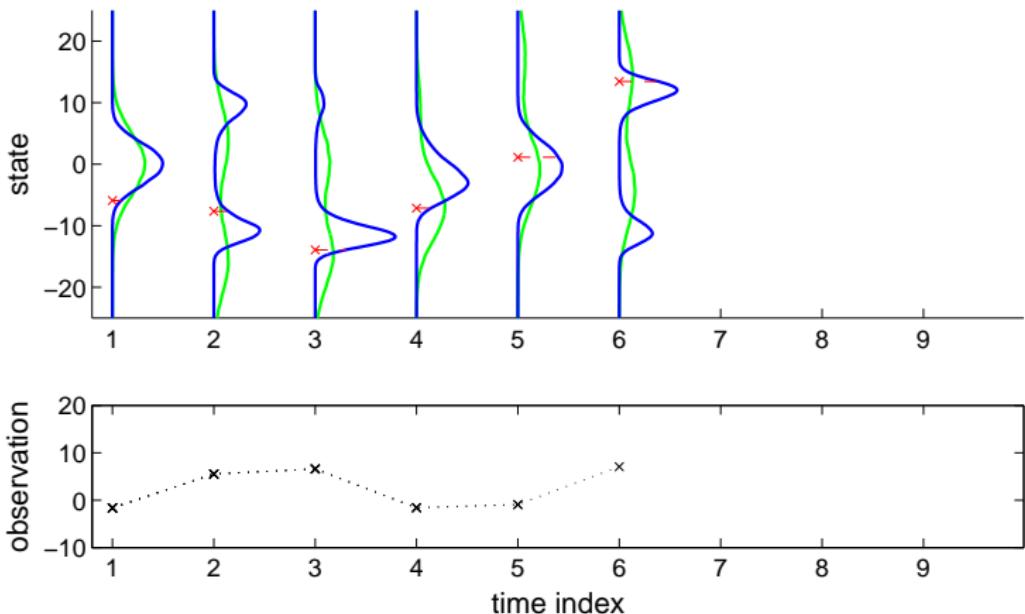
Predictive and filtering densities in the non-linear growth model



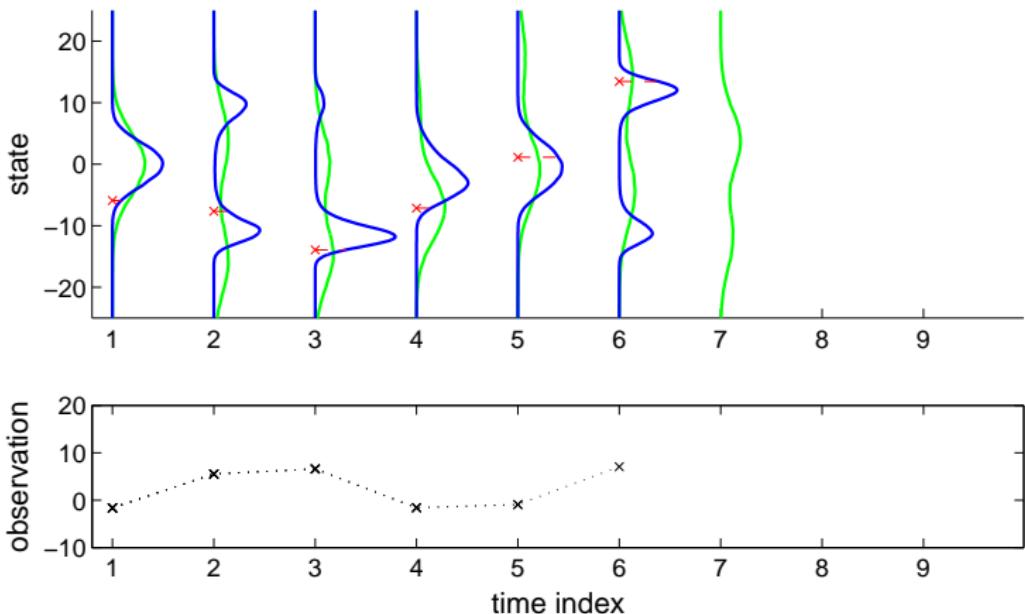
Predictive and filtering densities in the non-linear growth model



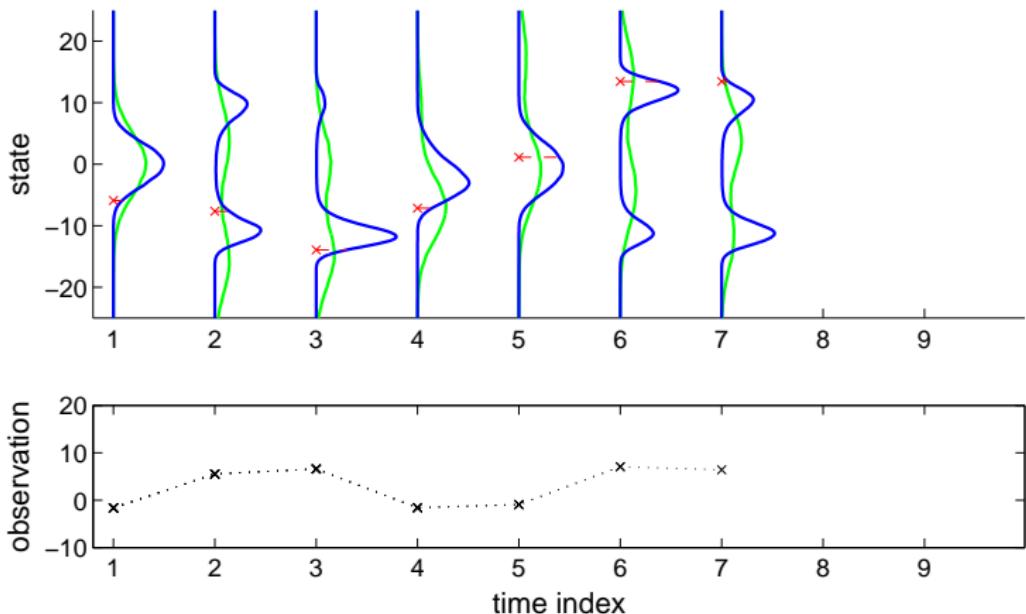
Predictive and filtering densities in the non-linear growth model



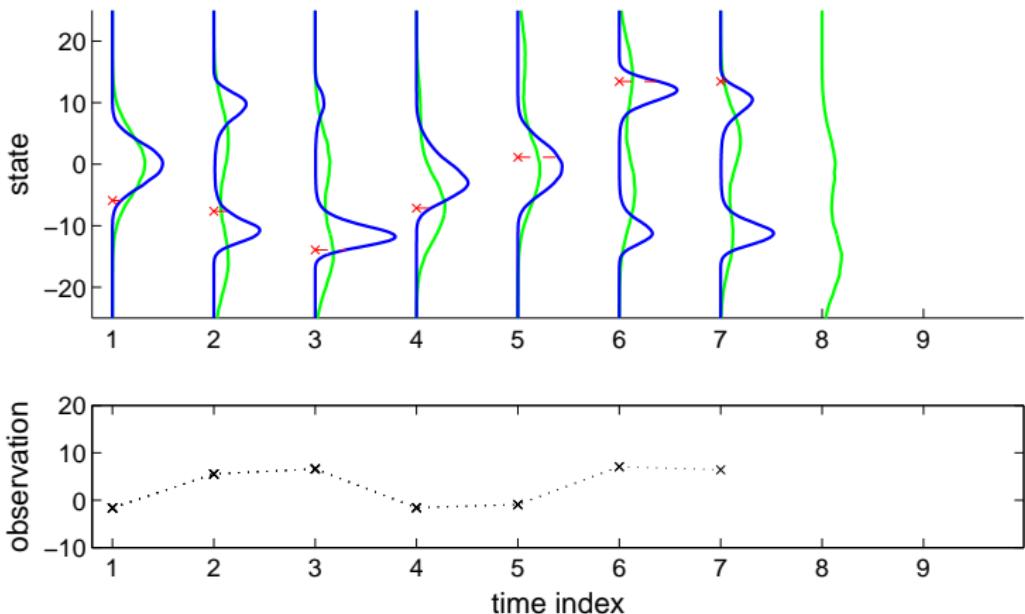
Predictive and filtering densities in the non-linear growth model



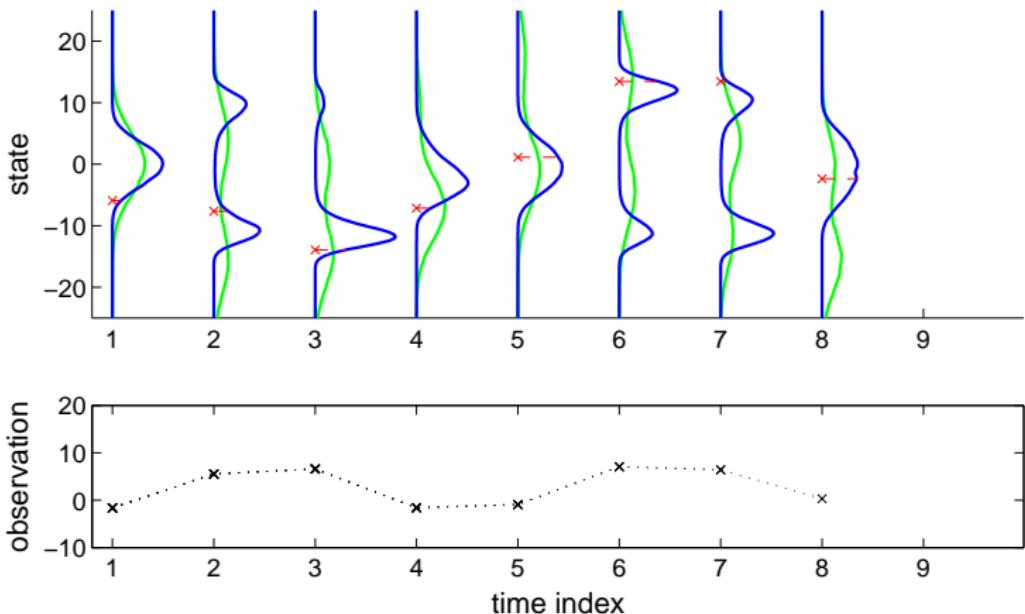
Predictive and filtering densities in the non-linear growth model



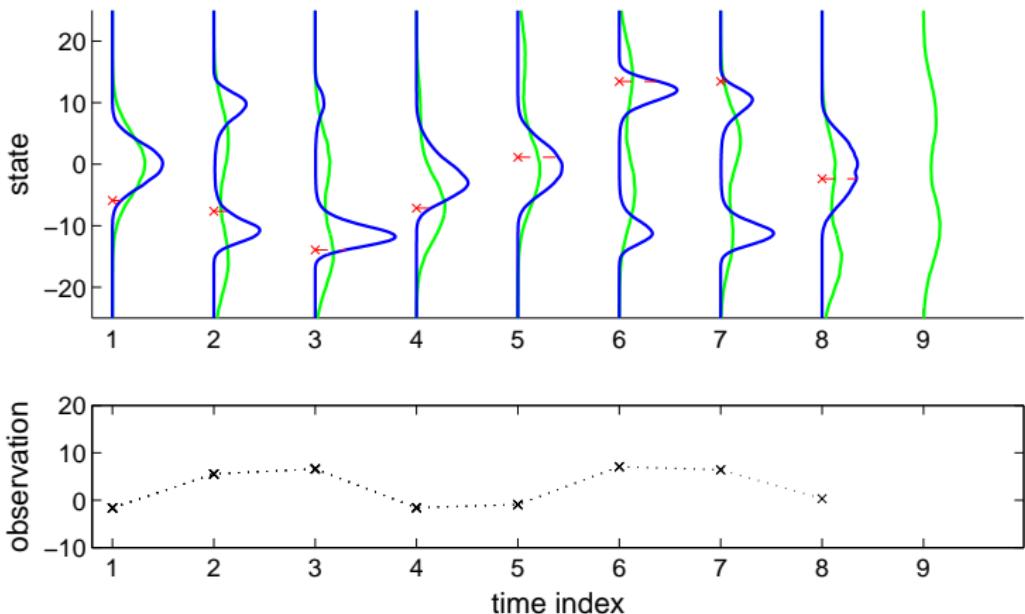
Predictive and filtering densities in the non-linear growth model



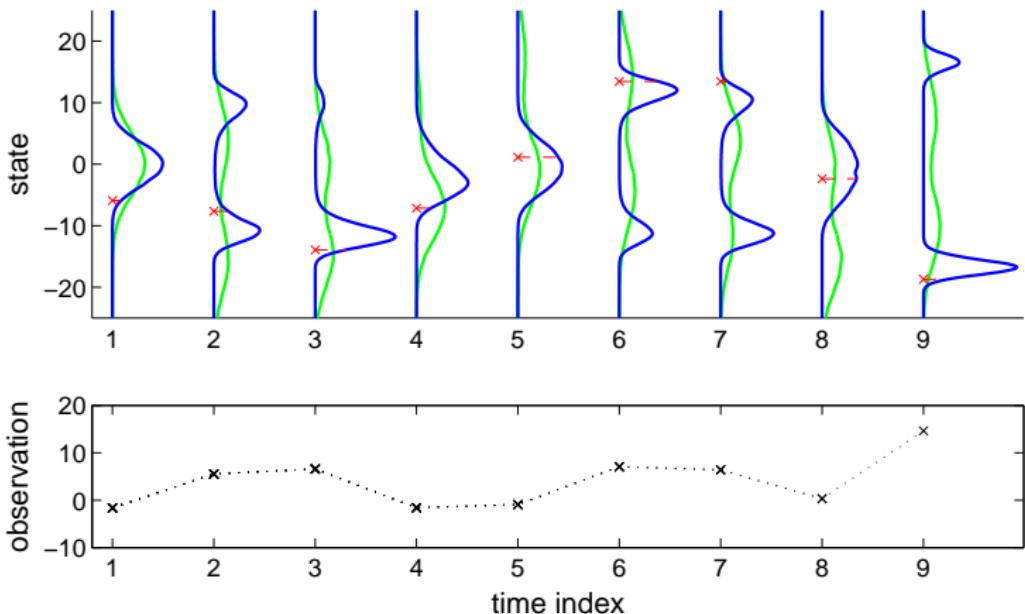
Predictive and filtering densities in the non-linear growth model



Predictive and filtering densities in the non-linear growth model



Predictive and filtering densities in the non-linear growth model



Predictive and filtering densities in the non-linear growth model

When the Model is Linear with Gaussian Noises

Prediction

- $X_k = A_{k-1}X_{k-1} + R_{k-1}U_{k-1}$
- together with $\phi_{k-1|k-1} = \mathcal{N}(\hat{X}_{k-1|k-1}, \Sigma_{k-1|k-1})$

imply that the predictive distribution is also Gaussian with mean and covariance given by

$$\hat{X}_{k|k-1} = A_{k-1}\hat{X}_{k-1|k-1} ,$$

$$\Sigma_{k|k-1} = A_{k-1}\Sigma_{k-1|k-1}A_{k-1}^t + R_{k-1}R_{k-1}^t .$$

Correction

Applying Bayes' rule in the linear Gaussian model

$$Y_k = B_k X_k + S_k V_k$$

is then a classic signal processing problem which leads to a new Gaussian distribution with mean and covariance defined as

$$\begin{aligned}\hat{X}_{k|k} &= \underbrace{\Sigma_{k|k-1} B_k^t (B_k \Sigma_{k|k-1} B_k^t + S_k S_k^t)^{-1}}_{K_k} \\ &\quad \times \underbrace{(Y_k - B_k \hat{X}_{k|k-1})}_{\epsilon_k},\end{aligned}$$

$$\Sigma_{k+1|k} = \Sigma_{k|k-1} - \Sigma_{k|k-1} B_k^t (B_k \Sigma_{k|k-1} B_k^t + S_k S_k^t)^{-1} B_k \Sigma_{k|k-1}.$$

When the State Takes a Finite Number of Values

- The corresponding model is usually referred to an **Hidden Markov Model**.
- The filtering recursion is a computational procedure known as the **forward pass of the forward-backward (or Baum-Welch) procedure**.

(Normalized) Forward Recursion

Assume $X = \{1, \dots, r\}$.

Initialization For $i = 1, \dots, r$,

$$\phi_{0|-1}(i) = \nu(i) .$$

Forward Recursion For $k = 0, \dots, n$,

$$c_k = \sum_{i=1}^r \phi_{k|k-1}(i) g_k(i, Y_k) ,$$

$$\phi_{k|k}(j) = \phi_{k|k-1}(j) g_k(j, Y_k) / c_k ,$$

$$\phi_{k+1|k}(j) = \sum_{i=1}^r \phi_{k|k}(i) q_k(i, j) ,$$

for each $j = 1, \dots, r$.

- The computational cost of the filtering procedure is of order $r^2 \times n$.
- A similar procedure can be used to compute the **smoothing probabilities** $\phi_{k|n}(i) = P(X_k = i | Y_{0:n})$ proceeding by backward recursion.

(Markovian Form of) Backward Recursion

Given stored values of $\phi_{0|0}, \dots, \phi_{n|n}$ and starting from the end of the data record, do

Initialization $\phi_{n|n}(j)$, for $j = 1, \dots, r$.

Backward Recursion For $k = n - 1, \dots, 0$,

- Compute, for $j, i = 1, \dots, r$,

$$\begin{aligned} B_k(j, i) &= P(X_k = i | X_{k+1} = j, Y_{0:k}) \\ &= \frac{\phi_{k|k}(i) q_k(i, j)}{\sum_{m=1}^r \phi_{k|k}(m) q_k(m, j)}. \end{aligned}$$

-

$$\phi_{k|n}(i) = \sum_{j=1}^r \phi_{k+1|n}(j) B_k(j, i),$$

for $i = 1, \dots, r$.

Roadmap

1 State-Space Models

- The Functional Representation
- The Markovian Representation

2 The Linear Prediction Approach

- A Very Simple (Static) Example
- The General Kalman Filter
- Virtues and Limitations

3 The Bayesian Approach

- The General Filtering Recursion
- The Linear Gaussian State-Space Case
- The Finite State Space Case

4 Approximate Gaussian Filters

- The Extended Kalman Filter (EKF)
- The Unscented Kalman Filter (UKF)

Approximate Gaussian (or assumed density) filters work by approximating the filtering and prediction distributions by Gaussian distributions.

- 1 Hence, these allow for closed-form updates of mean and covariance matrix parameters, as in Kalman filter.
- 2 The obtained results however are only approximate.
- 3 These approaches are more successful in the presence of moderately non-Gaussian noises (with unimodal and light tail noise densities) and in the presence of mildly non-linear state and observation equations.

Extended Kalman Filter (EKF)

- The EKF approach dates back to the 1970's.
- A prototypal example of use is the **non-linear space model with additive noises**, given by

$$\begin{aligned}X_{k+1} &= a(X_k) + R_k U_k , \\Y_k &= b(X_k) + S_k V_k ,\end{aligned}$$

where

- a, b are non-linear vector-valued functions, supposed to be continuously differentiable.
- $\{U_k\}_{k \geq 0}$ and $\{V_k\}_{k \geq 0}$ are uncorrelated **white noises**.
- X_0 has prior mean $\hat{X}_{0|-1}$ and covariance $\Sigma_{0|-1}$ and is uncorrelated with $\{U_k\}_{k \geq 0}$ and $\{V_k\}_{k \geq 0}$.

Extended Kalman Filter: Prediction

- Mimicking the Kalman filter recursion, it seems reasonable to adapt the state prediction as

$$\hat{X}_{k-1|k-1} \rightarrow \hat{X}_{k|k-1} = a(\hat{X}_{k-1|k-1}) .$$

Extended Kalman Filter: Prediction

- 1 Mimicking the Kalman filter recursion, it seems reasonable to adapt the state prediction as

$$\hat{X}_{k-1|k-1} \rightarrow \hat{X}_{k|k-1} = a(\hat{X}_{k-1|k-1}) .$$

- 2 Linearizing the state equation at $\hat{X}_{k-1|k-1}$

$$X_k - \hat{X}_{k|k-1} \approx A_{k-1}(X_{k-1} - \hat{X}_{k-1|k-1}) + R_{k-1}U_{k-1} ,$$

where $A_{k-1} \stackrel{\text{def}}{=} \nabla a(\hat{X}_{k-1|k-1})$.

Extended Kalman Filter: Prediction

- 1 Mimicking the Kalman filter recursion, it seems reasonable to adapt the state prediction as

$$\hat{X}_{k-1|k-1} \rightarrow \hat{X}_{k|k-1} = a(\hat{X}_{k-1|k-1}) .$$

- 2 Linearizing the state equation at $\hat{X}_{k-1|k-1}$

$$X_k - \hat{X}_{k|k-1} \approx A_{k-1}(X_{k-1} - \hat{X}_{k-1|k-1}) + R_{k-1}U_{k-1} ,$$

where $A_{k-1} \stackrel{\text{def}}{=} \nabla a(\hat{X}_{k-1|k-1})$.

- 3 This suggests to approximate the a priori state error covariance by

$$\Sigma_{k-1|k-1} \rightarrow \Sigma_{k|k-1} = A_{k-1}\Sigma_{k-1|k-1}A_{k-1}^T + R_{k-1}R_{k-1}^t .$$

Extended Kalman Filter: Innovation

- 1 The next step is to compute the **innovation**, defined as a proxy for the prediction error,

$$\epsilon_k = Y_k - b(\hat{X}_{k|k-1}) .$$

Extended Kalman Filter: Innovation

- 1 The next step is to compute the **innovation**, defined as a proxy for the prediction error,

$$\epsilon_k = Y_k - b(\hat{X}_{k|k-1}) .$$

- 2 Linearizing the measurement equation at $\hat{X}_{k|k-1}$ yields

$$\epsilon_k = b(X_k) - b(\hat{X}_{k|k-1}) + S_k V_k \approx B_k (X_k - \hat{X}_{k|k-1}) + S_k V_k ,$$

where $B_k \stackrel{\text{def}}{=} \nabla b(\hat{X}_{k|k-1})$.

Extended Kalman Filter: Innovation

- 1 The next step is to compute the **innovation**, defined as a proxy for the prediction error,

$$\epsilon_k = Y_k - b(\hat{X}_{k|k-1}) .$$

- 2 Linearizing the measurement equation at $\hat{X}_{k|k-1}$ yields

$$\epsilon_k = b(X_k) - b(\hat{X}_{k|k-1}) + S_k V_k \approx B_k (X_k - \hat{X}_{k|k-1}) + S_k V_k ,$$

where $B_k \stackrel{\text{def}}{=} \nabla b(\hat{X}_{k|k-1})$.

- 3 The innovation covariance may thus be approximated by

$$\Gamma_k = B_k \Sigma_{k|k-1} B_k^T + S_k S_k^t .$$

Extended Kalman Filter: Correction

- 1 The correction mechanism outlined in the Kalman filtering recursion derivation suggests to compute the posterior state estimate from the prior state estimate by

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k \epsilon_k ,$$

where K_k is the **Kalman Gain**.

Extended Kalman Filter: Correction

- 1 The correction mechanism outlined in the Kalman filtering recursion derivation suggests to compute the posterior state estimate from the prior state estimate by

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k \epsilon_k ,$$

where K_k is the **Kalman Gain**.

- 2 Here ϵ_k is no longer exactly orthogonal to $\hat{X}_{k|k-1}$ but it is (hopefully !) approximately so, which suggests to compute The Kalman Gain as

$$K_k = \Sigma_{k|k-1} B_k^T \left(B_k \Sigma_{k|k-1} B_k^T + S_k S_k^T \right)^{-1} .$$

Extended Kalman Filter: Correction

- 1 The correction mechanism outlined in the Kalman filtering recursion derivation suggests to compute the posterior state estimate from the prior state estimate by

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k \epsilon_k ,$$

where K_k is the **Kalman Gain**.

- 2 Here ϵ_k is no longer exactly orthogonal to $\hat{X}_{k|k-1}$ but it is (hopefully !) approximately so, which suggests to compute The Kalman Gain as

$$K_k = \Sigma_{k|k-1} B_k^T \left(B_k \Sigma_{k|k-1} B_k^T + S_k S_k^T \right)^{-1} .$$

- 3 Similarly, the error covariance can be evaluated using the Riccati equations.

The Extended Kalman Filter

- Exploiting the assumption that the transformations are locally quasi-linear, the EKF simply linearizes all non-linear transformation and substitutes Jacobian matrices for the linear transformations in the KF recursions.
- Although the EKF mimics the elegant and computationally efficient recursive updates of the KF, it suffers from several drawbacks.

Extended Kalman Filter

- Linearized transformations are only reliable if the error propagation can be well approximated by a linear function: if this condition does not hold, the linearized approximation can be extremely poor.
- Contrary to the Kalman filter, the EKF is not guaranteed to be stable and divergence may occur if either the initial estimation error or the disturbance noise are not small enough (the linearization then become invalid).
- As a consequence, additional tricks should be used to avoid divergence, making the algorithm difficult to implement and to tune... and the EKF is only reliable for systems which are almost linear on the time-scale of the updates.
- Linearization can be applied only if the Jacobian matrix exists. However, this is not always the case... some systems contain discontinuities (process might be jump-linear, measurements may be quantized, etc.)

The Unscented Kalman Filter (UKF)

- To address the obvious EKF deficiencies, alternative approximate filtering techniques have been introduced.
- These methods are sub-optimal in the sense that they do not compute the posterior distribution of the state, but compute recursively proxies for the posterior state mean and covariance, mimicking the KF **prediction-correction mechanism**.
- One of the most successful approach so far is the so-called Unscented Kalman Filter, introduced by Uhlman and Julier (1996).

The Unscented Transformation

- The Unscented transformation is a method for propagating quantities approximating the mean and the covariance of a random variable X which undergoes a nonlinear transformation.
- The idea is to consider a set of points (the **sigma-points**) with weights,

$$\mathcal{S} = \{(\xi^i, \omega^i), i = 1, \dots, p\}, \quad \sum_{i=1}^p \omega^i = 1.$$

- The sigma-points are selected so as to represent the first and second order moments.

The Unscented Transformation

The sigma-points and the weights are chosen in such a way that:

- 1 The ensemble average $E[X]$ is equal to the weighted average of the σ -points

$$E[X] = \sum_{i=1}^p \omega^i \xi^i .$$

- 2 The covariance $Cov[X]$ is equal to the outer product of the sigma-points

$$Cov(X) = \sum_{i=1}^p \omega^i \left(\xi^i - \sum_{i=1}^p \omega^i \xi^i \right)^{\otimes 2} ,$$

where for a vector x , $x^{\otimes 2} = xx^T$.

Choice of the Sigma-Points

One set of points that satisfies the conditions is a symmetric set of $p = 2N_x$ points that lie on the $\sqrt{N_x}$ -th covariance contour, i.e. for $i = 1, \dots, N_x$,

$$\xi^i = \mathbb{E}[X] + \left(\sqrt{N_x \text{Cov}(X)} \right)_i, \quad \omega^i = 1/2N_x,$$

$$\xi^{i+N_x} = \mathbb{E}[X] - \left(\sqrt{N_x \text{Cov}(X)} \right)_i, \quad \omega^{i+N_x} = 1/2N_x,$$

where $(\left(\sqrt{N_x \text{Cov}[X]} \right)_i)$ is the i -th row or column of the square root of the covariance matrix $\text{Cov}[X]$ scaled by N_x . Other sets of sigma-points are considered in Julier and Uhlmann (2004).

The Unscented Transformation

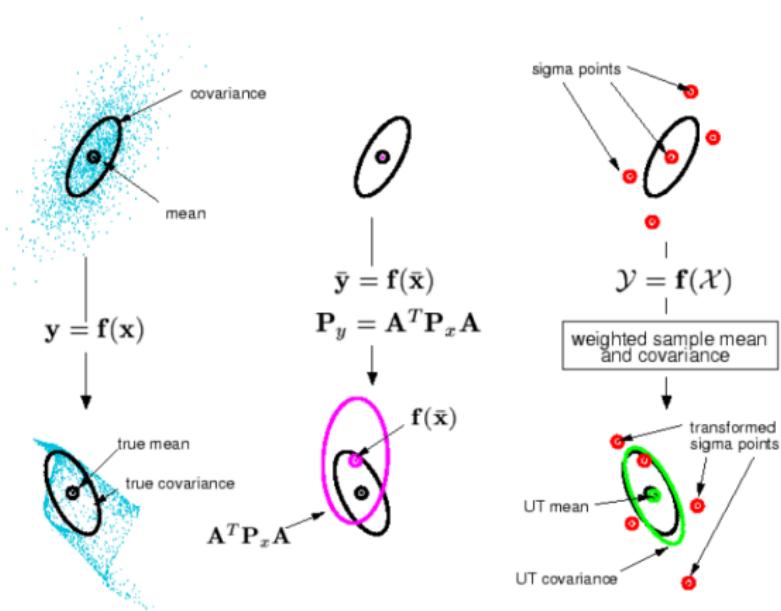
- 1 Instantiate each point through the function f of interest to yield the set of transformed sigma points: $\tilde{\xi}^i = f(\xi^i)$.
- 2 The mean of $Y = f(X)$, $E[Y]$, is approximated by the weighted average of the transformed points:

$$\widehat{E[Y]} = \sum_{i=1}^p \omega^i \tilde{\xi}^i .$$

- 3 The covariance of Y , $Cov(Y)$ is the weighted outer product of the transformed points

$$\widehat{Cov[Y]} = \sum_{i=1}^p \omega^i \left(\tilde{\xi}^i - \sum_{i=1}^p \omega^i \tilde{\xi}^i \right)^{\otimes 2} .$$

The Unscented Transformation



From E. Wan and R. van der Merwe, AS-SPCC2000

Unscented Transformation vs. Linearization

- 1 The computational cost of the algorithm is the same order of magnitude as the EKF. The most expensive operations are calculating the matrix square root and the outer products required to compute the covariance of the projected sigma points.
- 2 Any set of sigma points that encodes the mean and covariance correctly calculates the projected mean and covariance correctly to the second order **without the need to calculate any derivatives**.
- 3 The algorithm can be used with discontinuous transformations.

The Unscented Kalman Filter: Principles

- The Unscented Kalman Filter (UKF) is a straightforward extension of the Unscented Transformation to the recursive estimation.
- The state distribution is again represented by its mean and covariance which are represented using **sigma-points**.

The Unscented Kalman Filter: Implementation

- The state is redefined as the concatenation of the original state and noise variables: $X_k^a = (X_k^T \ U_k^T V_k^T)^T$.
- The UT sigma-point selection scheme is applied to this new augmented state: the mean and covariance of this augmented state is

$$\mu_k^a = (\hat{X}_{k|k}^T, 0^T, 0^T)^T, \quad \Sigma_{k|k}^a = \text{diag}(\Sigma_{k|k}, R_k R_k^t, S_k S_k^t).$$

- These sigma-points are propagated through the non-linear system of state-space equations

$$\begin{pmatrix} X_{k+1} \\ Y_{k+1} \end{pmatrix} = \begin{pmatrix} a_k(X_k) + U_k \\ b_{k+1}(a_k(X_k) + U_k) + V_{k+1} \end{pmatrix}$$

to produce the transformed sigma-points, which are used to approximate the state mean and covariance conditioned on Y_{k+1} .

Part II

From Monte Carlo Methods to Particle Filtering

- 5 Monte Carlo Methods
- 6 Importance Sampling
- 7 From Importance Sampling to Sampling Importance Resampling
- 8 Sequential Importance Sampling Principles
 - Selection of the Proposal Kernel
 - Degeneracy of the weights
- 9 Sequential Importance Sampling/Resampling
 - Principles
 - Particle Approximation in Action

Roadmap

5 Monte Carlo Methods

6 Importance Sampling

7 From Importance Sampling to Sampling Importance Resampling

8 Sequential Importance Sampling Principles

- Selection of the Proposal Kernel
- Degeneracy of the weights

9 Sequential Importance Sampling/Resampling

- Principles
- Particle Approximation in Action

An Historical Perspective

- Numerical methods that are known as Monte Carlo methods can be loosely described as **statistical simulation methods**, where statistical simulation is defined in quite general terms to be **any method that utilizes sequences of random numbers to perform the simulation**.
- Monte Carlo methods have been used for centuries, but only in the past several decades has the technique gained the status of a full-fledged numerical method capable of addressing the most complex applications.
- The name **Monte Carlo** was coined during the Manhattan Project of World War II, because of the similarity of statistical simulation to games of chance !

Monte-Carlo Versus Discretisation

- Statistical simulation methods may be contrasted to conventional numerical discretization methods, which typically are applied to ordinary or partial differential equations that describe some underlying physical or mathematical system. The only requirement is that the **system be described by probability distributions**.
- Once the distributions are known, the Monte Carlo simulation can proceed by **random sampling from the distributions**. Many simulations are then performed (**multiple trials or histories**) and the desired result is taken as an average over the number of observations.

Monte-Carlo Integration

- 1 **Objective** Given a probability density μ , how to evaluate numerically $\int_X f(x)\mu(x)dx$ for arbitrary functions f ?
- 2 **The Monte Carlo Answer**
 - Draw an independent sample ξ^1, \dots, ξ^N from μ .
 - Compute $N^{-1} \sum_{i=1}^N f(\xi^i)$.
- 3 This technique is applicable only when direct sampling from the distribution μ is feasible.

A Radically Simple Example

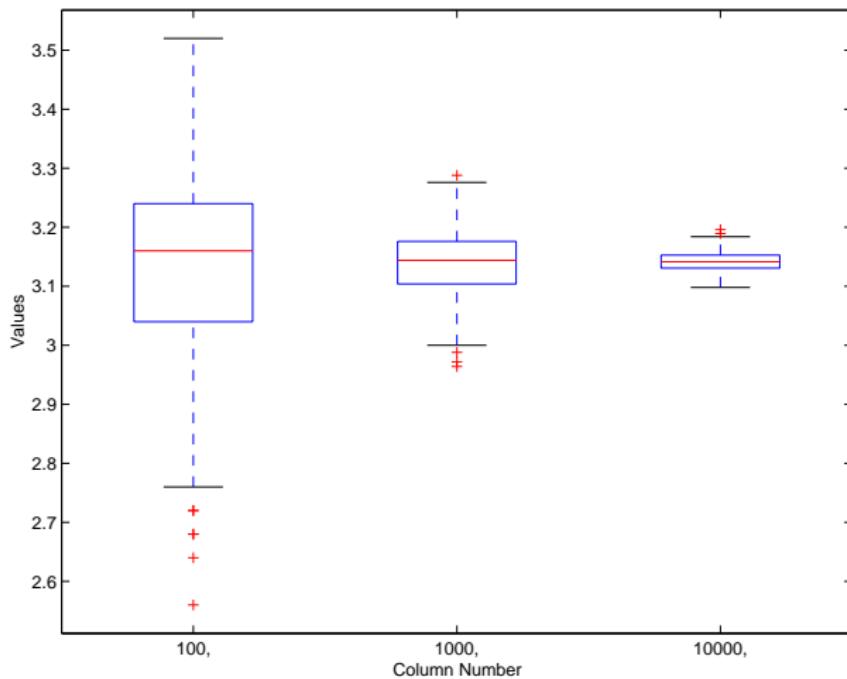
- **Objective** Compute $\pi \dots$ using the Monte-Carlo method. For this purpose, set $f(x_1, x_2) = \mathbb{1}\{x_1^2 + x_2^2 \leq 1\}$ and let μ be the uniform distribution on the square $[-1, +1]^{\times 2}$. Then,

$$\frac{\pi}{4} = \frac{1}{4} \iint \mathbb{1}\{x_1^2 + x_2^2 \leq 1\} dx_1 dx_2$$

- Draw an independent sample ξ^1, \dots, ξ^N from the uniform distribution over the square $\mathcal{U}([0, 1])^{\otimes 2}$ and compute the number of hits in the disk !

$$\hat{S}_N = N^{-1} \sum_{i=1}^N \mathbb{1}\{|\xi^i|^2 \leq 1\}$$

- A first Monte-Carlo estimator is given by $\hat{\pi}_N = 4\hat{S}_N$.



Boxplot for 1000 replications of the Monte-Carlo estimator of π for different sample sizes

A Radically Simple Example

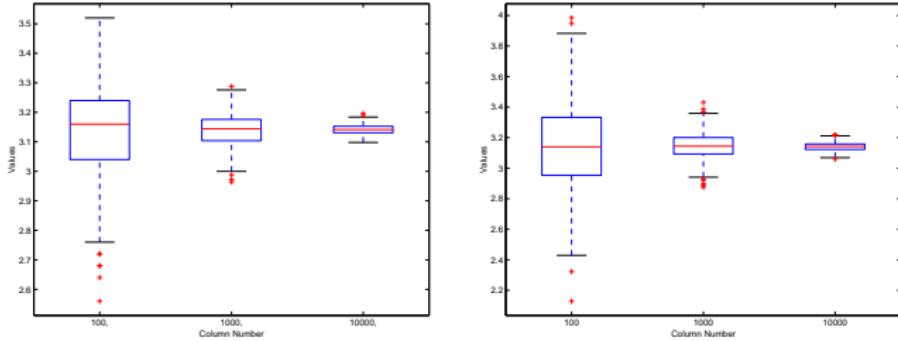
- There are many other ways, using the same random numbers to evaluate π ; indeed

$$\frac{1}{4} \iint_{[0,1]^{x^2}} (x_1^2 + x_2^2)^p \mathbb{1}\{x_1^2 + x_2^2 \leq 1\} dx_1 dx_2 = \frac{\pi}{2(p+2)} .$$

- We might consider for example the Monte-Carlo estimator $2(p+2)\hat{S}^{(p)}$, with

$$\hat{S}_n^{(p)} = N^{-1} \sum_{i=1}^N |\xi^i|^p \mathbb{1}\{|\xi^i|^2 \leq 1\}$$

where we average the norm raised to the power p of all the points that hit the disk.



Boxplot for 1000 replications of the Monte-Carlo estimator of π for different sample sizes; left panel $p = 0$. Right panel: $p = 2$

Output Error Analysis

- By the **Law of Large Numbers** (LLN),
 $\hat{\mu}_N(f) = N^{-1} \sum_{i=1}^N f(\xi^i)$ converges in probability and almost surely to $\mu(f) = \int f(x)\mu(x)dx$, provided that $\mu|f| < \infty$.
- LLN is not enough to provide an estimate of the error.
- In many practical applications however, one can predict the statistical error (the “variance”) in this average result, and hence an estimate of the number of Monte Carlo trials that are needed to achieve a given error.

Output Error Analysis

- The key tool for output error analysis is the **Central Limit Theorem** (CLT) which shows that, provided that $\mu(f^2) < \infty$,

$$\frac{1}{\sqrt{N}} \sum_{i=1}^N \{f(\xi^i) - \mu(f)\} \xrightarrow{\mathcal{D}} \mathcal{N}(0, \sigma^2(\mu, f)) ,$$

where the **asymptotic variance** is given by

$$\sigma^2(\mu, f) = \mu \{ [f - \mu(f)]^2 \} .$$

- Roughly speaking, the size of the error decreases as the square root of the sample size N !... the convergence can thus be pretty slow but does not depend upon the dimension of the state-space X

Roadmap

5 Monte Carlo Methods

6 Importance Sampling

7 From Importance Sampling to Sampling Importance Resampling

8 Sequential Importance Sampling Principles

- Selection of the Proposal Kernel
- Degeneracy of the weights

9 Sequential Importance Sampling/Resampling

- Principles
- Particle Approximation in Action

Importance Sampling: General Principle

- Sample from an **instrumental (proposal) distribution** ν and apply a **change-of-measure formula** to account for the fact that ν differs from the **target distribution** μ .

Importance Sampling: General Principle

- Sample from an **instrumental (proposal) distribution** ν and apply a **change-of-measure formula** to account for the fact that ν differs from the **target distribution** μ .
- If the instrumental distribution ν pdf **covers** the pdf μ , the IS estimate is

$$\mu(f) = \int f(x) \mu(x) dx = \int f(x) \frac{\mu(x)}{\nu(x)} \nu(x) dx ,$$

where $W(x) = \frac{\mu(x)}{\nu(x)}$ is the **importance function (or importance ratio)**.

Importance Sampling: the Algorithm

- **Sampling** Draw an independent sample ξ^1, \dots, ξ^N from the distribution ν .

Importance Sampling: the Algorithm

- **Sampling** Draw an independent sample ξ^1, \dots, ξ^N from the distribution ν .
- **Weighting** Compute the **importance weights**

$$\omega^i \propto \frac{\mu(\xi^i)}{\nu(\xi^i)}, \quad \text{for } i = 1, \dots, N.$$

Importance Sampling: the Algorithm

- **Sampling** Draw an independent sample ξ^1, \dots, ξ^N from the distribution ν .
- **Weighting** Compute the **importance weights**

$$\omega^i \propto \frac{\mu(\xi^i)}{\nu(\xi^i)}, \quad \text{for } i = 1, \dots, N.$$

- Compute the **weighted** average
- **Weighted Monte Carlo Approximation**

$$\hat{\mu}_N^{\text{IS}}(f) = \frac{\sum_{i=1}^N f(\xi^i) \frac{\mu(\xi^i)}{\nu(\xi^i)}}{\sum_{i=1}^N \frac{\mu(\xi^i)}{\nu(\xi^i)}},$$

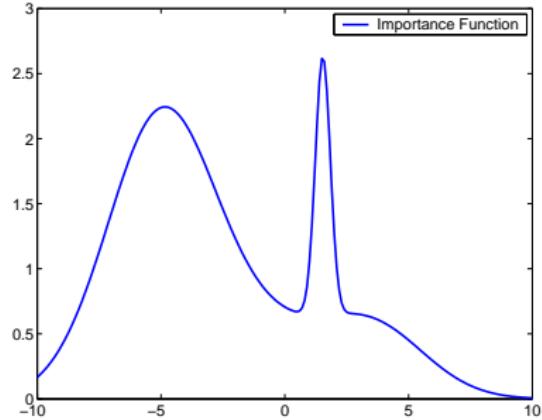
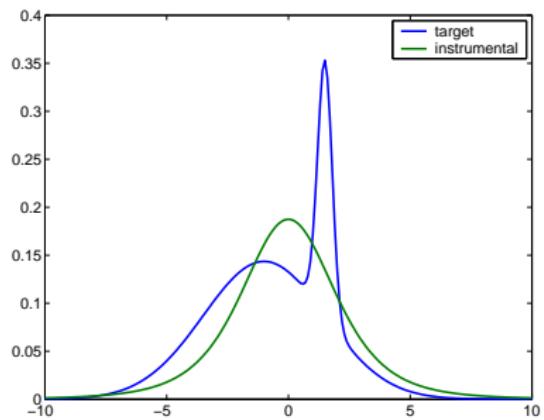
Importance Sampling

- The IS construction is universal but the performance strongly depends on the choices of the **target distribution** μ , the **instrumental distribution** ν and the function f .

Importance Sampling

- The IS construction is universal but the performance strongly depends on the choices of the **target distribution** μ , the **instrumental distribution** ν and the function f .
- **Idea** choose the importance distribution ν so that it generates values that are in the region where $f \frac{\mu}{\nu}$ is large.

Importance Sampling



target: mixture of gaussian. instrumental: student-t with 4 degrees of freedom (scale 2)

► More on Student's t distribution

Student's t-distribution

- Student's t pdf is

$$\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)} \frac{1}{\sqrt{\nu\pi}} \left(1 + \frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}$$

where ν is the number of degrees of freedom and Γ is... the Gamma function.

Student's t-distribution

- Student's t pdf is

$$\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)} \frac{1}{\sqrt{\nu\pi}} \left(1 + \frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}$$

where ν is the number of degrees of freedom and Γ is... the Gamma function.

- Student- t are often used in IS because

Student's t-distribution

- Student's t pdf is

$$\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)} \frac{1}{\sqrt{\nu\pi}} \left(1 + \frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}$$

where ν is the number of degrees of freedom and Γ is... the Gamma function.

- Student- t are often used in IS because
 - 1 the student t -pdf **polynomially decaying tails**... the importance function is bounded as soon as the target distribution has tail thinner than the Student's t tails.

Student's t-distribution

- Student's t pdf is

$$\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)} \frac{1}{\sqrt{\nu\pi}} \left(1 + \frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}$$

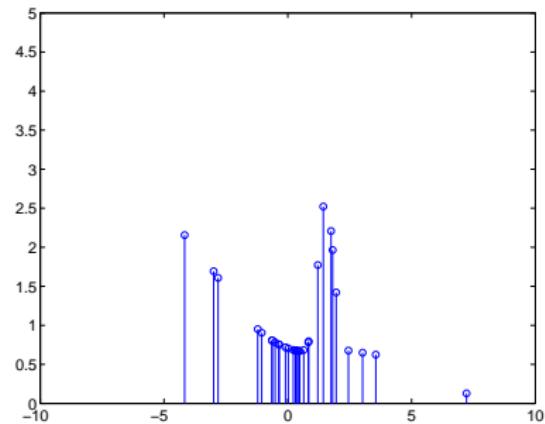
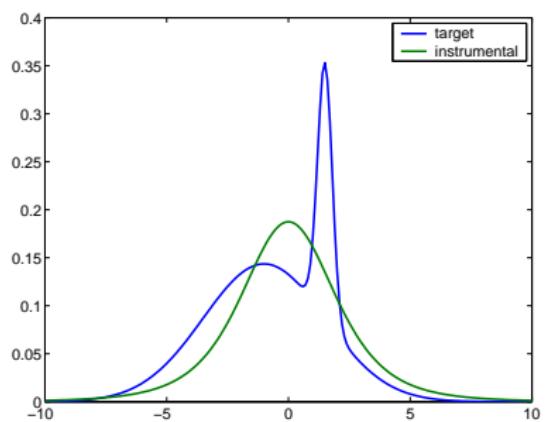
where ν is the number of degrees of freedom and Γ is... the Gamma function.

- Student- t are often used in IS because
 - 1 the student t -pdf **polynomially decaying tails**... the importance function is bounded as soon as the target distribution has tail thinner than the Student's t tails.
 - 2 it is pretty simple to sample from t -distribution (see Devroye (1988))

Student's t-distribution: History

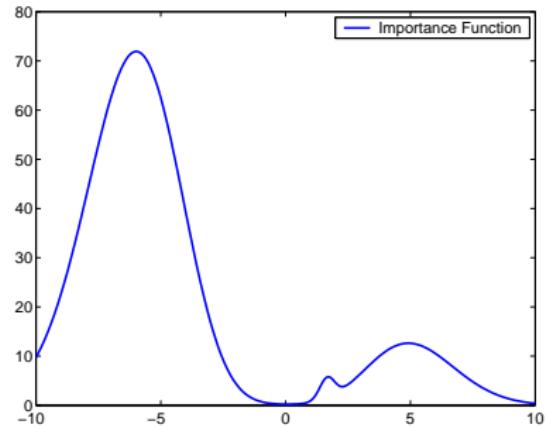
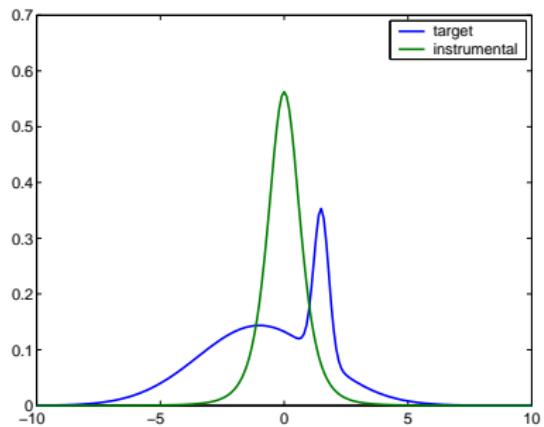
- W. S. Gossett (1908) discovered the distribution through his work at the Guinness brewery. At that time, Guinness did not allow its staff to publish, so Gossett used the pseudonym Student.

Importance Sampling



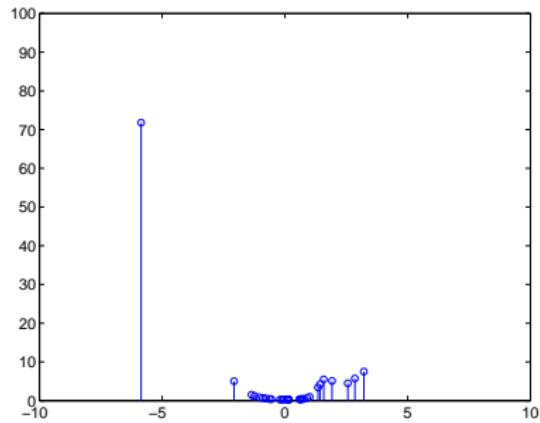
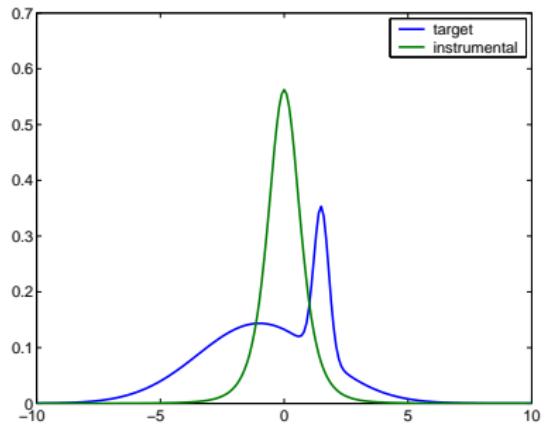
target: mixture of gaussian. instrumental: student-t with 4 degrees of freedom (scale 2)

Importance Sampling



target: mixture of gaussian. instrumental: student-t with 4 degrees of freedom (scale 1)

Importance Sampling



target: mixture of gaussian. instrumental: student-t with 4 degrees of freedom (scale 1)

Importance Sampling

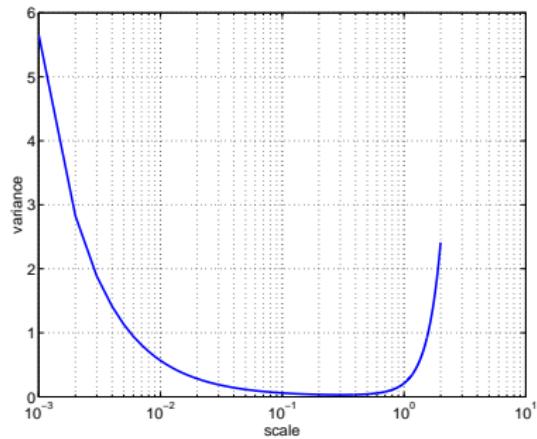
- Under appropriate conditions self-normalized IS is asymptotically normal

$$\sqrt{N}(\hat{\mu}_N^{\text{IS}}(f) - \mu(f)) \xrightarrow{\mathcal{D}} \mathcal{N}(0, \sigma^2(\nu, f)) ,$$

$$\sigma^2(\nu, f) = \text{Var}_{\nu}(W\{f - \mu(f)\}) = \nu \left((W)^2 (f - \mu(f))^2 \right) ,$$

where $W(x) = \mu(x)/\nu(x)$.

Importance Sampling



Variance of the IS estimate of the mean as a function of the scale of the instrumental distribution

Importance Sampling: Example

target distribution: Cauchy distribution $\mu = C(0, 1)$

instrumental distribution: Gaussian $\nu = N(0, 1)$.

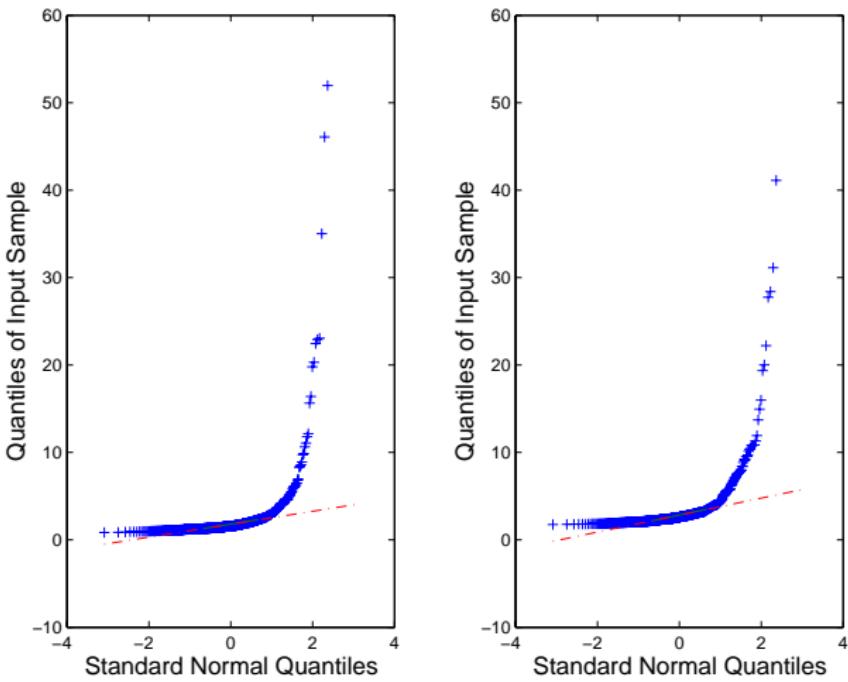
importance weight function:

$$W(x) = \sqrt{2\pi} \frac{\exp(-x^2/2)}{\pi (1 + x^2)}$$

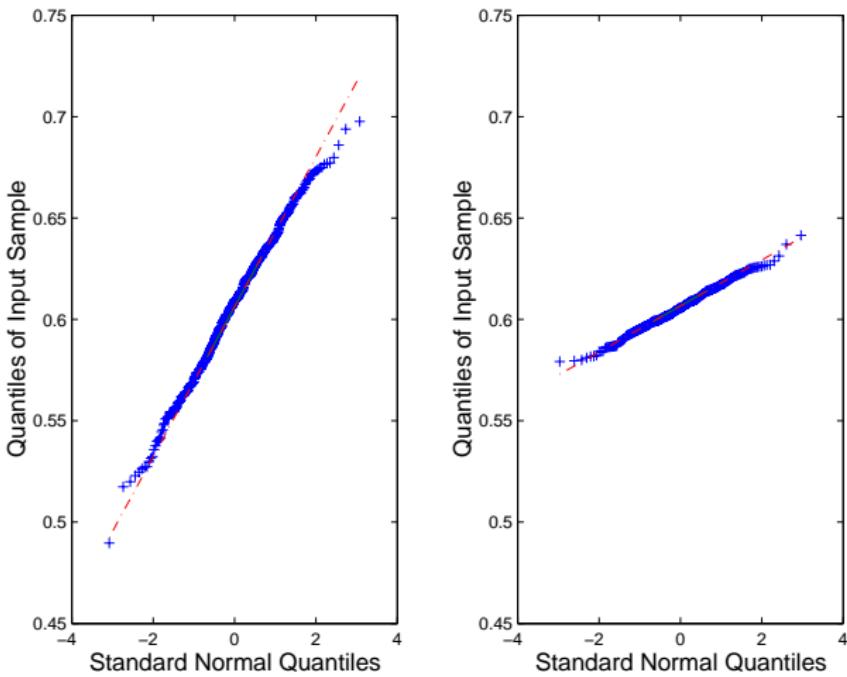
The importance weight function is obviously very badly behaved !

In particular,

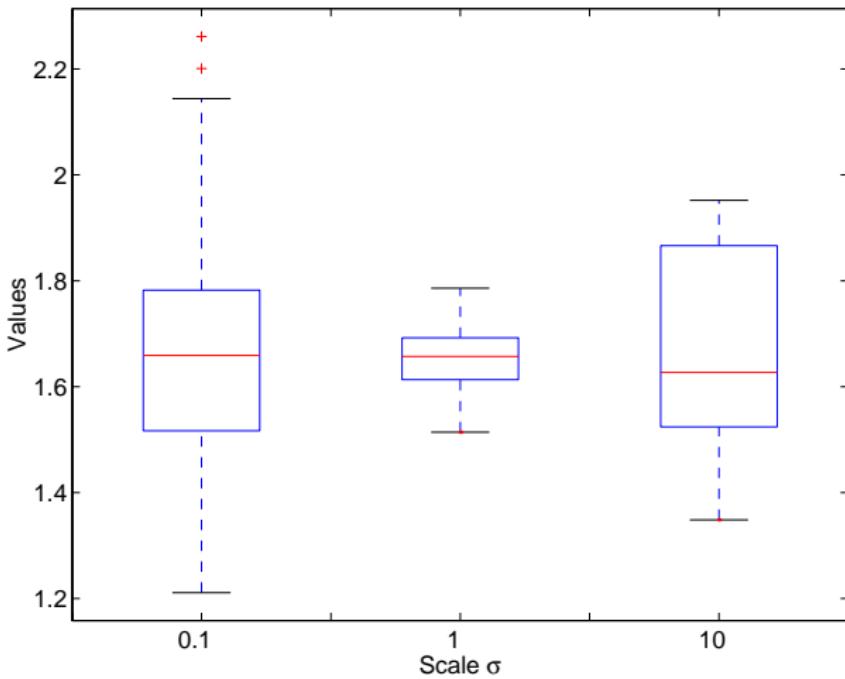
$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} (W(x))^2 \exp(-x^2/2) dx = \infty.$$



Quantile-quantile plot of the sample quantile of the normalized IS estimator versus the theoretical quantile from a normal distribution. Number of Monte-Carlo experiments: $m = 500$. Left panel: sample size $N = 100$. Right Panel: sample size $N = 1000$.



Quantile-quantile plot of the sample quantile of the normalized IS estimator versus the theoretical quantile from a normal distribution. Number of Monte-Carlo experiments: $m = 500$. Left panel: sample size $N = 100$. Right Panel: sample size $N = 1000$.



box and whisker plot of the normalized IS estimator for different values of the scale parameter $\sigma = 0.1, 1, 10$. Sample size $N = 1000$. Number of Monte-Carlo experiments: $m = 500$.

Roadmap

5 Monte Carlo Methods

6 Importance Sampling

7 From Importance Sampling to Sampling Importance Resampling

8 Sequential Importance Sampling Principles

- Selection of the Proposal Kernel
- Degeneracy of the weights

9 Sequential Importance Sampling/Resampling

- Principles
- Particle Approximation in Action

Sampling Importance Resampling (SIR)

- While IS is originally designed to approximate integrals like $\mu(f)$ it can also be used to sample from the distribution μ .

Sampling Importance Resampling (SIR)

- While IS is originally designed to approximate integrals like $\mu(f)$ it can also be used to sample from the distribution μ .
- Algorithm:

Sampling: Draw an i.i.d. sample $\tilde{\xi}^1, \dots, \tilde{\xi}^M$ from ν .

Sampling Importance Resampling (SIR)

- While IS is originally designed to approximate integrals like $\mu(f)$ it can also be used to **sample from the distribution μ** .
- **Algorithm:**

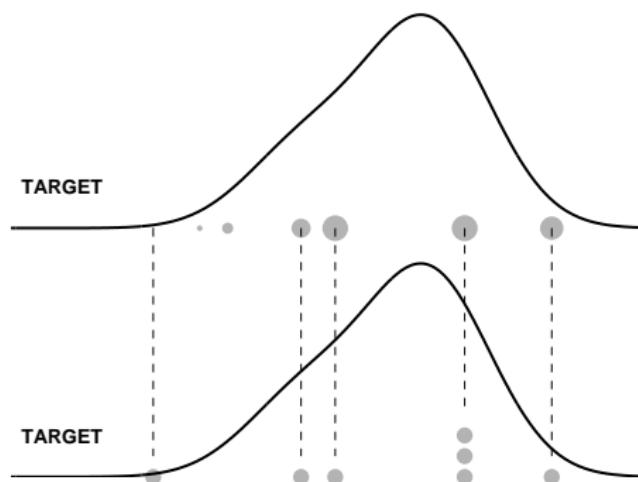
Sampling: Draw an i.i.d. sample $\tilde{\xi}^1, \dots, \tilde{\xi}^M$ from ν .

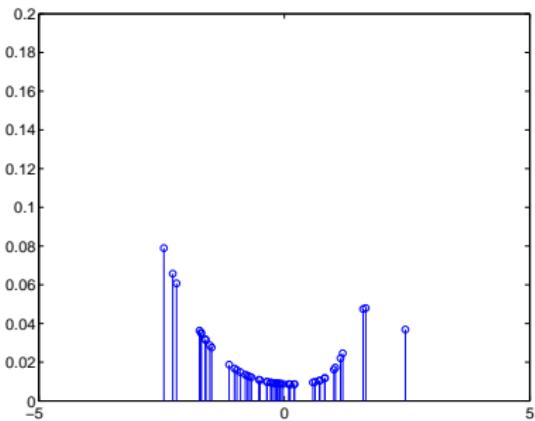
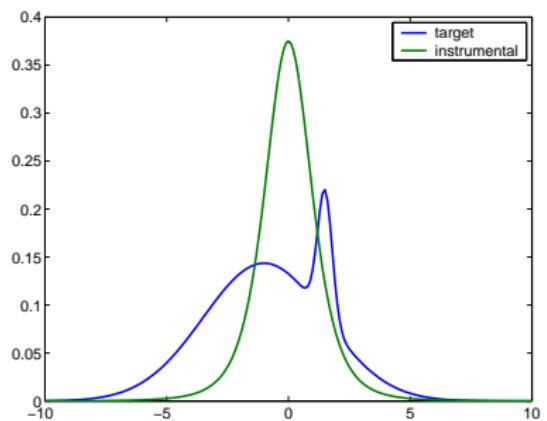
Weighting: Compute the (normalized) importance weights
 $\omega^i = W(\tilde{\xi}^i) / \sum_{j=1}^M W(\tilde{\xi}^j)$ for $i = 1, \dots, M$,
where $W(x) = \mu(x)/\nu(x)$.

Resampling: Draw (I^1, \dots, I^N) in the set $\{1, \dots, M\}$ with
probabilities $(\omega^1, \dots, \omega^M)$ and set, $\xi^i = \tilde{\xi}^{I^i}$.

Sampling Importance Resampling

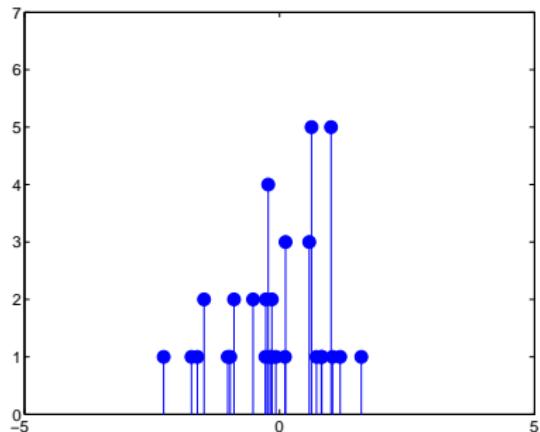
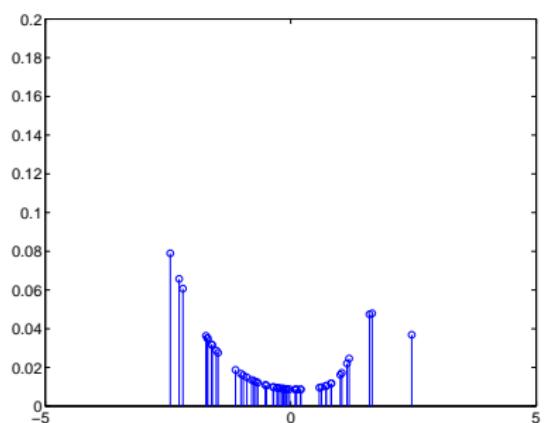
- In the resampling operation, the **bad points** $\tilde{\xi}^i$, as measured by the **importance function** $W(\tilde{\xi}^i)$, are discarded
- The **good points** are selected and perhaps replicated with probability proportional to their importance measured using the **fitness** function $W(\tilde{\xi}^i)$.
- The SIR paradigm is thus a statistical application of the Darwinian **survival of the fittest** principle...





target: mixture of gaussian; instrumental: student- t with scale
1

► More on Student's t distribution



target: mixture of gaussian; instrumental: student- t with scale
1

SIR Unbiasedness

- the SIR estimator $M^{-1} \sum_{i=1}^M f(\xi^i)$ may be alternatively expressed as

$$\frac{1}{M} \sum_{i=1}^M f(\xi^i) = \frac{1}{N} \sum_{i=1}^N N_i f(\tilde{\xi}^i)$$

where N_i is the number of times the particle $\tilde{\xi}^i$ is replicated
(if $N_i = 0$ the particle is actually discarded).

SIR Unbiasedness

- the SIR estimator $M^{-1} \sum_{i=1}^M f(\xi^i)$ may be alternatively expressed as

$$\frac{1}{M} \sum_{i=1}^M f(\xi^i) = \frac{1}{N} \sum_{i=1}^N N_i f(\tilde{\xi}^i)$$

where N_i is the number of times the particle $\tilde{\xi}^i$ is replicated
 (if $N_i = 0$ the particle is actually discarded).

- By construction, the conditional expectation of N_i
 proportional to the importance weight,

$$\mathbb{E}[N_i | \tilde{\xi}^1, \dots, \tilde{\xi}^M] = \omega^i \propto W(\tilde{\xi}^i) .$$

N_i is therefore conditionally unbiased..

SIR Unbiasedness

the SIR estimator $N^{-1} \sum_{i=1}^N f(\xi^i)$ is **conditionally on** $(\tilde{\xi}^1, \dots, \tilde{\xi}^M)$ an unbiased estimator of the IS estimator

$$\mathbb{E} \left[N^{-1} \sum_{i=1}^N f(\xi^i) \mid \tilde{\xi}^1, \dots, \tilde{\xi}^M \right] = \sum_{i=1}^M \frac{W(\tilde{\xi}^i)}{\sum_{j=1}^N W(\tilde{\xi}^j)} f(\tilde{\xi}^i).$$

SIR Variance

- ... but the mean squared error of the SIR estimator is always larger than that of the IS estimator

$$\begin{aligned} \text{E} \left(N^{-1} \sum_{i=1}^N f(\xi^i) - \mu(f) \right)^2 &= \\ \text{E} \left(N^{-1} \sum_{i=1}^N f(\xi^i) - \sum_{i=1}^M \omega^i f(\tilde{\xi}^i) \right)^2 + \text{E} \left(\sum_{i=1}^M \omega^i f(\tilde{\xi}^i) - \mu(f) \right)^2 \end{aligned}$$

where $\omega^i \propto W(\tilde{\xi}^i)$

SIR Variance

- ... but the mean squared error of the SIR estimator is always larger than that of the IS estimator

$$\begin{aligned} \mathrm{E} \left(N^{-1} \sum_{i=1}^N f(\xi^i) - \mu(f) \right)^2 &= \\ \mathrm{E} \left(N^{-1} \sum_{i=1}^N f(\xi^i) - \sum_{i=1}^M \omega^i f(\tilde{\xi}^i) \right)^2 + \mathrm{E} \left(\sum_{i=1}^M \omega^i f(\tilde{\xi}^i) - \mu(f) \right)^2 \end{aligned}$$

where $\omega^i \propto W(\tilde{\xi}^i)$

- sampling always brings in noise...

SIR: Large Sample Behavior

Theorem

$\hat{\mu}_{\nu,N}^{\text{SIR}}(f)$ is a consistent estimate of $\mu(f)$ for μ -integrable functions f . If $\lim_{N \rightarrow \infty} M/N = \alpha$ for some $\alpha \geq 1$ and that W and fW are in $L^2(\mathsf{X}, \nu)$, then

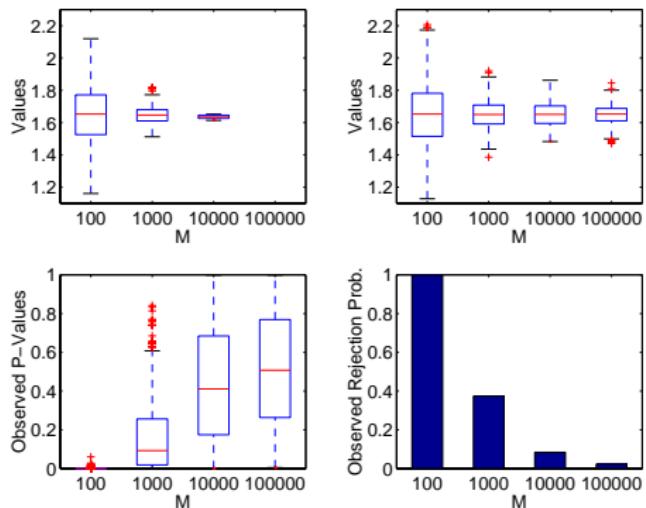
$$\sqrt{N}(\hat{\mu}_{\nu,M,N}^{\text{SIR}}(f) - \mu(f)) \xrightarrow{\mathcal{D}} \mathcal{N}(0, \tilde{\sigma}^2(f))$$

$$\tilde{\sigma}^2(f) = \underbrace{\text{Var}_\mu(f)}_{\text{variance of resampling}} + \alpha^{-1} \underbrace{\text{Var}_\nu(W \{f - \mu(f)\})}_{\text{variance of IS}}.$$

Example: Sampling a Gaussian from a Cauchy

- Target distribution μ : standard Gaussian
- Instrumental distribution ν : standard Cauchy.
- We set $N = 1,000$ and investigate the impact of the size M of the instrumental sample on the accuracy of the SIR estimator for $f(x) = \exp(-x)$.
- We display the box and whisker plots obtained from 500 independent Monte Carlo replications of the IS and SIR estimators of $\mu(f)$, for instrumental sample sizes $M = 100, 1,000, 10,000$ and $100,000$.

Sampling a Gaussian from a Cauchy



Top left: Box and whisker plot of the IS estimate. Top right: Box and whisker plot of the SIR estimate. Bottom left: Observed p -values of the Kolmogorov-Smirnov goodness of fit test of the null hypothesis that the distribution is standard Gaussian. Bottom right: Observed rejection probabilities of the null hypothesis at significance level 5%.

Alternative Resampling Schemes

- There are other resampling schemes that guarantee conditional unbiasedness

$$\mathbb{E} \left[N^i \mid \tilde{\xi}^1, \dots, \tilde{\xi}^M \right] = N\omega^i \quad i = 1, \dots, M .$$

Alternative Resampling Schemes

- There are other resampling schemes that guarantee conditional unbiasedness

$$\mathbb{E} \left[N^i \mid \tilde{\xi}^1, \dots, \tilde{\xi}^M \right] = N\omega^i \quad i = 1, \dots, M .$$

- Some of these have lower conditional variance and / or are easier to implement.

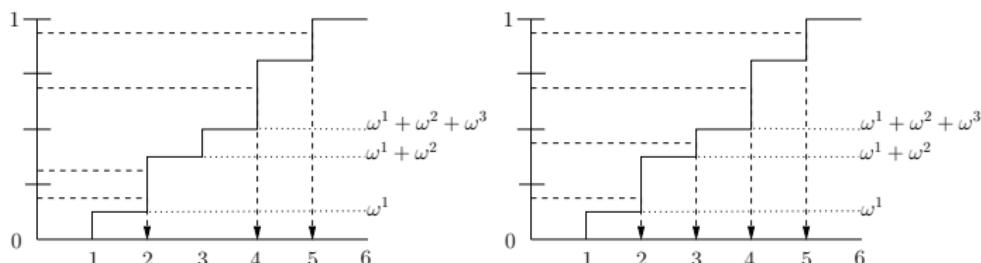
Alternative Resampling Schemes

- There are other resampling schemes that guarantee conditional unbiasedness

$$\mathbb{E} \left[N^i \mid \tilde{\xi}^1, \dots, \tilde{\xi}^M \right] = N\omega^i \quad i = 1, \dots, M .$$

- Some of these have lower conditional variance and / or are easier to implement.
- Several useful ideas can be derived from sampling strategy in survey...

Alternative Resampling Schemes



Principle of **stratified sampling** (left) and **systematic sampling** (right). Note: the latter does not always reduce the conditional variance.

Efficient Sampling from a Multinomial Distribution

- When N is large, efficient techniques for generating the multinomial draw should be used to keep the overall computational burden acceptable.
- In particular, the naive solutions consisting in drawing N independent random variables from the multinomial distribution $\text{Mult}(1, \omega^1, \dots, \omega^N)$ and adding them should be avoided because of its high computational cost.
- In fact, it is easy to derive an algorithm to sample of multinomial distribution $\text{Mult}(N, \omega^1, \dots, \omega^N)$ using an *ordered* sample of the uniform distribution.

Efficient Sampling from a Multinomial Distribution

- Generate the order statistics of $U_{(1)} \leq U_{(2)} \leq \cdots \leq U_{(N)}$ of N i.i.d. uniform random variables on $[0, 1]$,
- Set $p^0 = 0$ and $\bar{N}^0 = 0$. For $i \in \{1, \dots, N\}$ do
 - Compute $p^i = \sum_{j=1}^{i-1} \omega^j = p^{i-1} + \omega^i$,
 - Compute $N^i = \sup\{j \geq 0, U_{(j+\bar{N}^i)} \leq p^i\}$ and set $\bar{N}^i = \bar{N}^{i-1} + N^i$.

Sampling the Order Statistics of the Uniform Distribution

Let $0 \leq U_{(1)} \leq \dots \leq U_{(N)} \leq 1$ be the order statistics of U_1, U_2, \dots, U_N , a sequence of i.i.d. uniform $[0, 1]$ random variables. Then, if $U_{(N+1)} = 1$,

- $\left\{ \left(U_{(i)} / U_{(i+1)} \right)^i, i \in \{1, \dots, N\} \right\}$ is distributed as U_1, \dots, U_N .
- $U_N^{1/N}, U_N^{1/N} U_{N-1}^{1/N-1}, \dots, U_N^{1/N} \dots U_1^{1/1}$ is distributed as $U_{(N)}, \dots, U_{(1)}$.

Roadmap

- 5 Monte Carlo Methods
- 6 Importance Sampling
- 7 From Importance Sampling to Sampling Importance Resampling
- 8 Sequential Importance Sampling Principles
 - Selection of the Proposal Kernel
 - Degeneracy of the weights
- 9 Sequential Importance Sampling/Resampling
 - Principles
 - Particle Approximation in Action

- Particle filtering is rooted in Monte Carlo simulation and Bayesian estimation procedures.

- Particle filtering is rooted in Monte Carlo simulation and Bayesian estimation procedures.
- Contrary to the Kalman filter, a particle filter is a way to approximate (and not to compute exactly) the posterior distribution of the state given the observation.

- Particle filtering is rooted in **Monte Carlo simulation** and **Bayesian estimation** procedures.
- Contrary to the Kalman filter, a particle filter is a way to **approximate** (and not to compute exactly) the posterior distribution of the state given the observation.
- As such, it lacks the mathematical rigour that is an inherent characteristic of Kalman filtering.

- A particle refers to a **sample point** of the state space, so the posterior distribution is represented by a set of points: the particle filter therefore favours parallelism.

- A particle refers to a **sample point** of the state space, so the posterior distribution is represented by a set of points: the particle filter therefore favours parallelism.
- Moreover, case studies by many investigators working in diverse fields have demonstrated superior performance of particle filters over Kalman filters.

- A particle refers to a **sample point** of the state space, so the posterior distribution is represented by a set of points: the particle filter therefore favours parallelism.
- Moreover, case studies by many investigators working in diverse fields have demonstrated superior performance of particle filters over Kalman filters.
- The caveat, however, is that care has to be exercised in how certain design issues such as the choice of proposal distribution (basic to Monte-Carlo simulation) and the resampling step are implemented.

Sequential Importance Sampling

- **Principle** use Importance Sampling to approximate the filtering distribution in non-linear state space models.

Sequential Importance Sampling

- Principle use Importance Sampling to approximate the filtering distribution in non-linear state space models.
- Key remark Importance Sampling can be implemented sequentially, due to the particular structure of $\phi_{k|k}$.

Sequential Importance Sampling

- Principle use Importance Sampling to approximate the filtering distribution in non-linear state space models.
- Key remark Importance Sampling can be implemented sequentially, due to the particular structure of $\phi_{k|k}$.
- The corresponding algorithm is known as sequential importance sampling (SIS).

HMM Notations

Recall that an hidden Markov model is such that^{*}

$$X_{k+1} \sim q(X_k, \cdot) \quad \text{state equation}$$

$$Y_k \sim g(X_k, \cdot) \quad \text{measurement equation}$$

where

- $\{X_k\}_{k \geq 0}$ is a Markov chain with transition density q and initial distribution ν
- the observations $\{Y_k\}_{k \geq 0}$ are independent conditionally to $\{X_k\}_{k \geq 0}$: $g(X_k, Y_k)$ is the conditional density of Y_k given X_k .

*For simplicity, we will consider homogeneous models such that q and g do not depend on the time index k .

Filtering Recursion

$$\phi_{0:k|k}(x_{0:k}) = \frac{\phi_{0:k-1|k-1}(x_{0:k-1})t_{k-1}(x_{k-1}, x_k)}{\int \cdots \int \phi_{0:k-1|k-1}(x_{0:k-1})t_{k-1}(x_{k-1}, x'_k)dx'_k}$$

where, t_k is the (unnormalized) transition kernel density given by

$$t_{k-1}(x, x') = q(x, x')g(x', Y_k).$$

In the sequel, we sometimes use $g_k(x)$ as a shorthand notation for $g(x, Y_k)$.

Choice of the Instrumental Distribution

- Denote by $\{r_k\}_{k \geq 0}$ a family of transition density functions and ρ_0 a probability measure on X .

Choice of the Instrumental Distribution

- Denote by $\{r_k\}_{k \geq 0}$ a family of transition density functions and ρ_0 a probability measure on X .
- Assume that $\phi_{0|0}(x)/\rho_0(x) < \infty$ and for all $k \geq 0$ and all $(x, x') \in X \times X$, $t_k(x, x')/r_k(x, x') < \infty$.
- The inhomogeneous Markov chain with initial distribution ρ_0 and transition density $(r_k)_{k \geq 0}$ defines the following probability density on X^{k+1}

$$\rho_{0:k}(x_0, \dots, x_k) = \rho_0(x_0) \prod_{\ell=1}^k r_{\ell-1}(x_{\ell-1}, x_\ell) .$$

Sequential Computation of the Importance Function

- The importance function is then defined as

$$\frac{\phi_{0:k|k}(x_{0:k})}{\rho_{0:k}(x_{0:k})} \propto \frac{\phi_{0|0}(x_0)}{\rho_0(x_0)} \prod_{\ell=1}^k \frac{t_{\ell-1}(x_{\ell-1}, x_\ell)}{r_{\ell-1}(x_{\ell-1}, x_\ell)},$$

- It can be computed sequentially

$$\frac{\phi_{0:k|k}(x_{0:k})}{\rho_{0:k}(x_{0:k})} \propto \frac{\phi_{0:k-1|k-1}(x_{0:k-1})}{\rho_{0:k-1}(x_{0:k-1})} \underbrace{\frac{t_{k-1}(x_{k-1}, x_k)}{r_{k-1}(x_{k-1}, x_k)}}_{W_{k-1}(x_{k-1}, x_k)}.$$

Sequential Importance Sampling Algorithm

- 1 [Sampling] For $i = 1, \dots, N$ draw ξ_{k+1}^i from $r_k(\xi_k^i, \cdot)$.

Sequential Importance Sampling Algorithm

- 1 [Sampling] For $i = 1, \dots, N$ draw ξ_{k+1}^i from $r_k(\xi_k^i, \cdot)$.
- 2 [Weighting] Update the importance weight

$$\omega_{k+1}^i = \omega_k^i W_k(\xi_k^i, \xi_{k+1}^i) ,$$

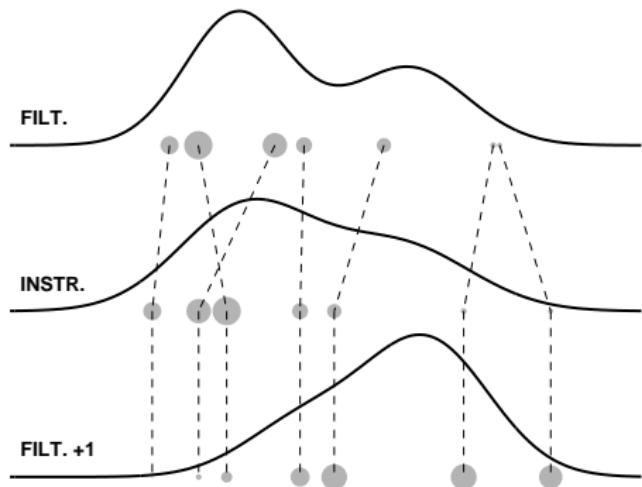
where $W_k(x_k, x_{k+1})$ is the **incremental importance weight**

Sequential Importance Sampling Algorithm

- 1 [Sampling] For $i = 1, \dots, N$ draw ξ_{k+1}^i from $r_k(\xi_k^i, \cdot)$.
- 2 [Weighting] Update the importance weight

$$\omega_{k+1}^i = \omega_k^i W_k(\xi_k^i, \xi_{k+1}^i) ,$$

where $W_k(x_k, x_{k+1})$ is the **incremental importance weight**
the points ξ_k^i are called **particles**; the trajectories $\xi_{0:k}^i$ path
particles.



One step of the SIS algorithm.

Guidelines

- The support of the proposal distribution should cover that of the posterior distribution; in other words, the proposal should have a **broad** distribution (long-tailed behavior).

Guidelines

- The support of the proposal distribution should cover that of the posterior distribution; in other words, the proposal should have a **broad** distribution (long-tailed behavior).
- The sampling procedure should be easy to implement.

Guidelines

- The support of the proposal distribution should cover that of the posterior distribution; in other words, the proposal should have a **broad** distribution (long-tailed behavior).
- The sampling procedure should be easy to implement.
- Account is taken of transition prior and likelihood, as well as most recent observation data.

Guidelines

- The support of the proposal distribution should cover that of the posterior distribution; in other words, the proposal should have a **broad** distribution (long-tailed behavior).
- The sampling procedure should be easy to implement.
- Account is taken of transition prior and likelihood, as well as most recent observation data.
- ideally, the proposal distribution is close (in shape) to the true posterior... however, satisfying all these goals is not easy !

The Prior Kernel

- The performance of SIS depends on the choice of r_k .

The Prior Kernel

- The performance of SIS depends on the choice of r_k .
- The most obvious solution is the **prior kernel** $r_k = q$, which is usually simple to sample from.

The Prior Kernel

- The performance of SIS depends on the choice of r_k .
- The most obvious solution is the **prior kernel** $r_k = q$, which is usually simple to sample from.
- The incremental weight

$$W_k(x, x') = g_{k+1}(x'), \quad \forall (x, x') \in \mathcal{X} \times \mathcal{X}.$$

does not depend on the past particle position !

The Prior Kernel

- The prior kernel is often computationally simple and is thus often hard to beat...

The Prior Kernel

- The prior kernel is often computationally simple and is thus often hard to beat...
- It is suboptimal if the observations are **informative**, i.e. the observation is telling much about the next state.

The Prior Kernel

- The prior kernel is often computationally simple and is thus often hard to beat...
- It is suboptimal if the observations are **informative**, i.e. the observation is telling much about the next state.
- It is also sensitive to the presence of **outliers** in the observations.

The Optimal Kernel

- The **optimal kernel** is defined in such a way that the incremental weight does depend only on **the current particle** ξ_k^i .
- This kernel is the conditional distribution distribution of the next state given the observation and the previous state:

$$t_k(x_k, x_{k+1}) = \frac{q(x_k, x_{k+1})g(x_{k+1}, Y_{k+1})}{\int q(x_k, x'_{k+1})g(x'_{k+1}, Y_{k+1})dx'_{k+1}}$$

- The importance weight is given by

$$W_k(x_k, x_{k+1}) = \int q(x_k, x_{k+1})g(x_{k+1}, Y_{k+1})dx_{k+1} = \gamma_k(x_k) .$$

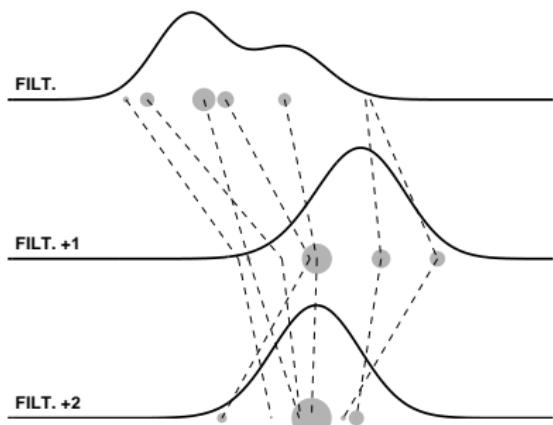
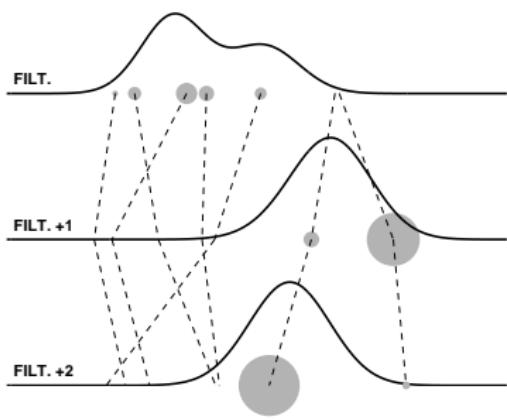
The Optimal kernel

- This is the only choice for which the conditional variance of the new weights is equal to zero !

The Optimal kernel

- This is the only choice for which the conditional variance of the new weights is equal to zero !
- Unfortunately, computing γ_k is usually not feasible in models where implementing the filtering recursion is problematic

The Optimal Kernel is More Robust to Outliers



The optimal kernel proposes particles in the regions where the filtering density has most of its mass.

Local Approximation of the Optimal Kernel

- Local approximations can be constructed in two steps:
 - 1 locate the **high-density regions** of the (multivariate) optimal distribution to spot important regions;

Local Approximation of the Optimal Kernel

- Local approximations can be constructed in two steps:
 - 1 locate the **high-density regions** of the (multivariate) optimal distribution to spot important regions;
 - 2 create an **overdispersed approximation** around the modes

Local Approximation of the Optimal Kernel

- Local approximations can be constructed in two steps:
 - 1 locate the **high-density regions** of the (multivariate) optimal distribution to spot important regions;
 - 2 create an **overdispersed approximation** around the modes
- Because the process is repeated for each particle, the procedure should be **SIMPLE**.

Application to the Stochastic Volatility Model

$$\begin{aligned} X_{k+1} &= \phi X_k + \sigma U_k & |\phi| < 1 , \\ Y_k &= \beta \exp(X_k/2) V_k , \end{aligned}$$

where

- 1 $\{U_k\}_{k \geq 0}$ and $\{V_k\}_{k \geq 0}$ are independent standard Gaussian white noise processes.
- 2 $X_0 \sim \mathcal{N}(0, \sigma^2/(1 - \phi^2))$.

Stochastic Volatility Model

- The prior transition $X_k|X_{k-1} = x$ is Gaussian with mean ϕx and variance σ^2 ,

$$q(x, x') = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x' - \phi x)^2}{2\sigma^2}\right)$$

Stochastic Volatility Model

- The prior transition $X_k|X_{k-1} = x$ is Gaussian with mean ϕx and variance σ^2 ,

$$q(x, x') = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x' - \phi x)^2}{2\sigma^2}\right)$$

- The conditional likelihood $Y_k|X_k = x'$ is Gaussian with mean 0 and variance $\beta^2 \exp(x')$,

$$g_{k+1}(x') = \frac{1}{\sqrt{2\pi\beta^2}} \exp\left(-\frac{Y_{k+1}^2}{2\beta^2} \exp(-x') - \frac{1}{2}x'\right) ,$$

and the incremental weight $\gamma_k(x)$ is not available in closed form.

Stochastic Volatility Model (contd.)

- The function $x' \mapsto \log(q(x, x')g_{k+1}(x'))$ is (strictly) **log-concave** and thus **unimodal**.

Stochastic Volatility Model (contd.)

- The function $x' \mapsto \log(q(x, x')g_{k+1}(x'))$ is (strictly) **log-concave** and thus **unimodal**.
- The mode $m_k(x)$ of the optimal transition density is the unique solution of the non-linear equation,

$$-\frac{1}{\sigma^2}(x' - \phi x) + \frac{Y_k^2}{2\beta^2} \exp(-x') - \frac{1}{2} = 0.$$

Stochastic Volatility Model (contd.)

- The function $x' \mapsto \log(q(x, x')g_{k+1}(x'))$ is (strictly) **log-concave** and thus **unimodal**.
- The mode $m_k(x)$ of the optimal transition density is the unique solution of the non-linear equation,

$$-\frac{1}{\sigma^2}(x' - \phi x) + \frac{Y_k^2}{2\beta^2} \exp(-x') - \frac{1}{2} = 0.$$

- The solution of this equation can be computed numerically.

Stochastic Volatility Model (contd.)

- We might use, for instance, as instrumental kernel a t -distribution with $\eta = 5$ degrees of freedom, located at the mode $m_k(x)$.

Stochastic Volatility Model (contd.)

- We might use, for instance, as instrumental kernel a t -distribution with $\eta = 5$ degrees of freedom, located at the mode $m_k(x)$.
- The scale can be taken as the inverse of the negated second-order derivative of $x' \mapsto \log q(x, x')g_k(x')$ evaluated at the mode $m_k(x)$,

$$\sigma_k^2(x) = \left(\frac{1}{\sigma^2} + \frac{Y_{k+1}^2}{2\beta^2} \exp[-m_k(x)] \right)^{-1}.$$

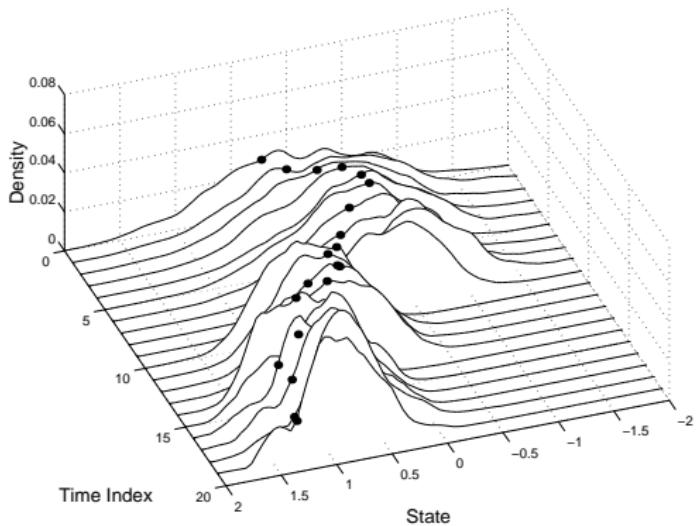
Stochastic Volatility Model (contd.)

- We might use, for instance, as instrumental kernel a t -distribution with $\eta = 5$ degrees of freedom, located at the mode $m_k(x)$.
- The scale can be taken as the inverse of the negated second-order derivative of $x' \mapsto \log q(x, x')g_k(x')$ evaluated at the mode $m_k(x)$,

$$\sigma_k^2(x) = \left(\frac{1}{\sigma^2} + \frac{Y_{k+1}^2}{2\beta^2} \exp[-m_k(x)] \right)^{-1}.$$

- The incremental weight may easily be evaluated once $m_k(x)$ and $\sigma_k^2(x)$ have been computed (note that it now depends both on x and x').

Application to the Stochastic Volatility Model (contd.)



Waterfall representation of the sequence of estimated filtering distribution with actual state (1,000 particles).

- When there is no obvious way to locate the modes, a possible solution consists in using classical non-linear filtering procedures to approximate t_k .

- When there is no obvious way to locate the modes, a possible solution consists in using classical non-linear filtering procedures to approximate t_k .
- These include in particular the **Extended Kalman filter** (EKF), the **Unscented Kalman filter** (UKF), etc.

Extended Kalman Filter

- Consider a non-linear state-space model of the form

$$\begin{aligned} X_{k+1} &= a(X_k) + U_k , & U_k &\sim \mathcal{N}(0, \sigma_U^2 I) , \\ Y_k &= b(X_k) + V_k , & V_k &\sim \mathcal{N}(0, \sigma_V^2 I) , \end{aligned}$$

where a, b are non-linear functions.

Extended Kalman Filter

- Consider a non-linear state-space model of the form

$$\begin{aligned} X_{k+1} &= a(X_k) + U_k, & U_k &\sim \mathcal{N}(0, \sigma_U^2 I), \\ Y_k &= b(X_k) + V_k, & V_k &\sim \mathcal{N}(0, \sigma_V^2 I), \end{aligned}$$

where a, b are non-linear functions.

- The EKF approximate this model by a **non-linear** Gaussian state-space model with **linear** measurement equation...

Extended Kalman Filter

- Consider a non-linear state-space model of the form

$$\begin{aligned} X_{k+1} &= a(X_k) + U_k, & U_k &\sim \mathcal{N}(0, \sigma_U^2 I), \\ Y_k &= b(X_k) + V_k, & V_k &\sim \mathcal{N}(0, \sigma_V^2 I), \end{aligned}$$

where a, b are non-linear functions.

- The EKF approximate this model by a **non-linear** Gaussian state-space model with **linear** measurement equation...
- Since the measurement equations is linear, we are then back to a model for which the optimal kernel may be determined exactly.

Extended Kalman Filter

- We will adopt the approximation

$$X_{k+1} = a(X_{k-1}) + U_k ,$$

$$Y_k \approx b(a(X_{k-1})) + B(X_{k-1})(X_k - a(X_{k-1})) + V_k ,$$

where $B(x)$ is the matrix of partial derivatives of $b(x)$ at $a(x)$.

Extended Kalman Filter

- We will adopt the approximation

$$X_{k+1} = a(X_{k-1}) + U_k ,$$

$$Y_k \approx b(a(X_{k-1})) + B(X_{k-1})(X_k - a(X_{k-1})) + V_k ,$$

where $B(x)$ is the matrix of partial derivatives of $b(x)$ at $a(x)$.

- Note that the measurement equation depends both on the current state X_k and on the previous one X_{k-1} and thus departs from the HMM assumptions.

Extended Kalman Filter

- On the other hand, **when conditioning** on the value of X_{k-1} the structure of both models are exactly similar !

Extended Kalman Filter

- On the other hand, **when conditioning** on the value of X_{k-1} the structure of both models are exactly similar !
- Hence the posterior distribution of the state X_k given $X_{k-1} = x$ and Y_k is a Gaussian distribution with mean $m_k(x)$ and covariance matrix $\Gamma_k(x)$ which can be evaluated according to

$$\begin{aligned} K_k(x) &= \sigma_U^2 B^t(x) (\sigma_U^2 B(x) B^t(x) + \sigma_V^2)^{-1}, \\ m_k(x) &= a(x, 0) + K_k(x) \{Y_k - b(a(x))\}, \\ \Gamma(x) &= (I - K_k(x) B(x)) R(x) R^t(x). \end{aligned}$$

Stochastic Growth Model

$$X_k = f_{k-1}(X_{k-1}) + U_{k-1} \quad U_k \sim \mathcal{N}(0, 3)$$

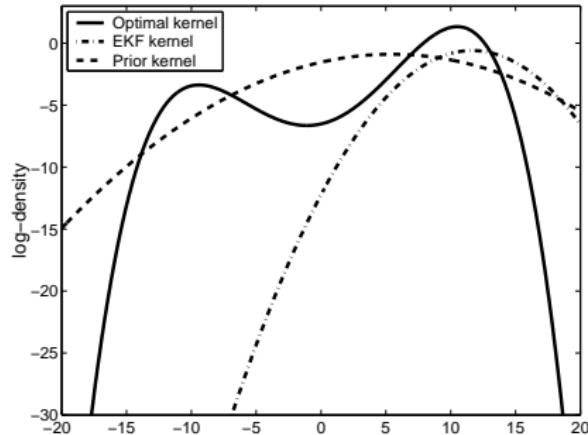
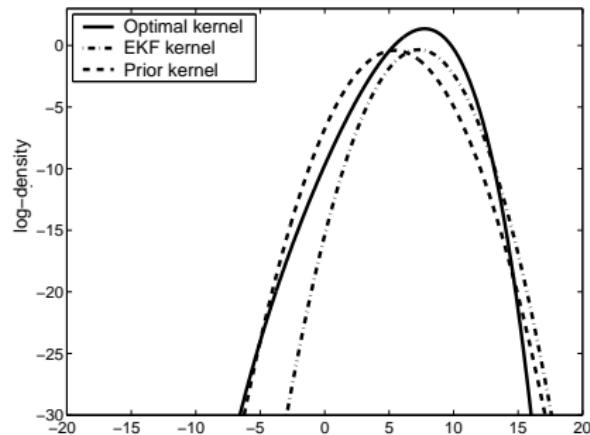
$$Y_k = 0.05X_k^2 + V_k \quad V_k \sim \mathcal{N}(0, 1).$$

$$f_{k-1}(x) = .5X_{k-1} + 25 \frac{X_{k-1}}{1 + X_{k-1}^2} + 8 \cos(1.2(k-1)).$$

The likelihood adds a twist to the problem.

- $Y_k \leq 0$: the log-likelihood is unimodal and symmetric about 0.
- $Y_k > 0$: the likelihood is symmetric about 0 with modes at $\pm(Y_k/b_0)^{1/2}$.

Stochastic Growth Model



Log-density of the optimal kernel (solid line), EKF approximation of the optimal kernel (dashed-dotted line) and the prior kernel (dashed line) for two different values of the state noise variance σ_u^2 : left, $\sigma_u^2 = 1$; right, $\sigma_u^2 = 10$.

Weight Degeneracy

- The normalized importance weights measure the pertinence of each particle.

Weight Degeneracy

- The normalized importance weights measure the pertinence of each particle.
- Small importance weight implies that the associated particle is far from the main body of the posterior distribution and contributes poorly.

Weight Degeneracy

- The normalized importance weights measure the pertinence of each particle.
- Small importance weight implies that the associated particle is far from the main body of the posterior distribution and contributes poorly.
- If there are too many such ineffective particles, the Monte-Carlo approximation is unreliable.

Weight Degeneracy

- The normalized importance weights measure the pertinence of each particle.
- Small importance weight implies that the associated particle is far from the main body of the posterior distribution and contributes poorly.
- If there are too many such ineffective particles, the Monte-Carlo approximation is unreliable.
- Empirically, this phenomenon “always” happens when n gets larger (N being fixed).

Roadmap

- 5 Monte Carlo Methods
- 6 Importance Sampling
- 7 From Importance Sampling to Sampling Importance Resampling
- 8 Sequential Importance Sampling Principles
 - Selection of the Proposal Kernel
 - Degeneracy of the weights
- 9 Sequential Importance Sampling/Resampling
 - Principles
 - Particle Approximation in Action

Resampling

- The solution, proposed by (Gordon, Salmond & Smith, 1993), to avoid the degeneracy of the importance weights is to regularly **resample** the particles according to their importance weights.

Resampling

- The solution, proposed by (Gordon, Salmond & Smith, 1993), to avoid the degeneracy of the importance weights is to regularly **resample** the particles according to their importance weights.
- The basic idea of resampling is to
 - 1 **eliminate** particles which have small importance weights,

Resampling

- The solution, proposed by (Gordon, Salmond & Smith, 1993), to avoid the degeneracy of the importance weights is to regularly **resample** the particles according to their importance weights.
- The basic idea of resampling is to
 - 1 **eliminate** particles which have small importance weights,
 - 2 **replicate** particles which have large importance weights in proportion of their relevance.

Resampling

- The solution, proposed by (Gordon, Salmond & Smith, 1993), to avoid the degeneracy of the importance weights is to regularly **resample** the particles according to their importance weights.
- The basic idea of resampling is to
 - 1 **eliminate** particles which have small importance weights,
 - 2 **replicate** particles which have large importance weights in proportion of their relevance.
- Resampling concentrates the particles in regions of the state space which are pertinent and avoids exploration of highly improbable areas.

Resampling

- This idea is clearly rooted in the sampling importance resampling (SIR) technique.

Resampling

- This idea is clearly rooted in the sampling importance resampling (SIR) technique.
- Contrary to standard SIR, the aim of the resampling step is not to draw a sample but rather to avoid weight degeneracy.

Resampling

- This idea is clearly rooted in the sampling importance resampling (SIR) technique.
- Contrary to standard SIR, the aim of the resampling step is not to draw a sample but rather to avoid weight degeneracy.
- **Drawback:** Resampling introduces unnecessary noise into the algorithm, and this extra noise might be far from negligible.

Sequential Importance Sampling with Resampling (SISR)

■ Sampling

- Draw $(\tilde{\xi}_{k+1}^1, \dots, \tilde{\xi}_{k+1}^N)$ from the instrumental kernel:
 $\tilde{\xi}_{k+1}^i \sim r_k(\xi_k^i, \cdot), i = 1, \dots, N.$

Sequential Importance Sampling with Resampling (SISR)

■ Sampling

- Draw $(\tilde{\xi}_{k+1}^1, \dots, \tilde{\xi}_{k+1}^N)$ from the instrumental kernel:
 $\tilde{\xi}_{k+1}^i \sim r_k(\xi_k^i, \cdot), i = 1, \dots, N.$
- Update importance weights

$$\omega_{k+1}^i = \omega_k^i W_k(\xi_k^i, \tilde{\xi}_{k+1}^i)$$

Sequential Importance Sampling with Resampling (SISR)

■ Sampling

- Draw $(\tilde{\xi}_{k+1}^1, \dots, \tilde{\xi}_{k+1}^N)$ from the instrumental kernel:
 $\tilde{\xi}_{k+1}^i \sim r_k(\xi_k^i, \cdot), i = 1, \dots, N.$
- Update importance weights

$$\omega_{k+1}^i = \omega_k^i W_k(\xi_k^i, \tilde{\xi}_{k+1}^i)$$

■ Resampling (Optional):

- Draw indices $(I_{k+1}^1, \dots, I_{k+1}^N)$ with probabilities of success proportional to

$$\omega_{k+1}^1, \dots, \omega_{k+1}^N.$$

and reset the importance weights ω_{k+1}^i to a constant value.

The Bootstrap Filter

SISR applied with

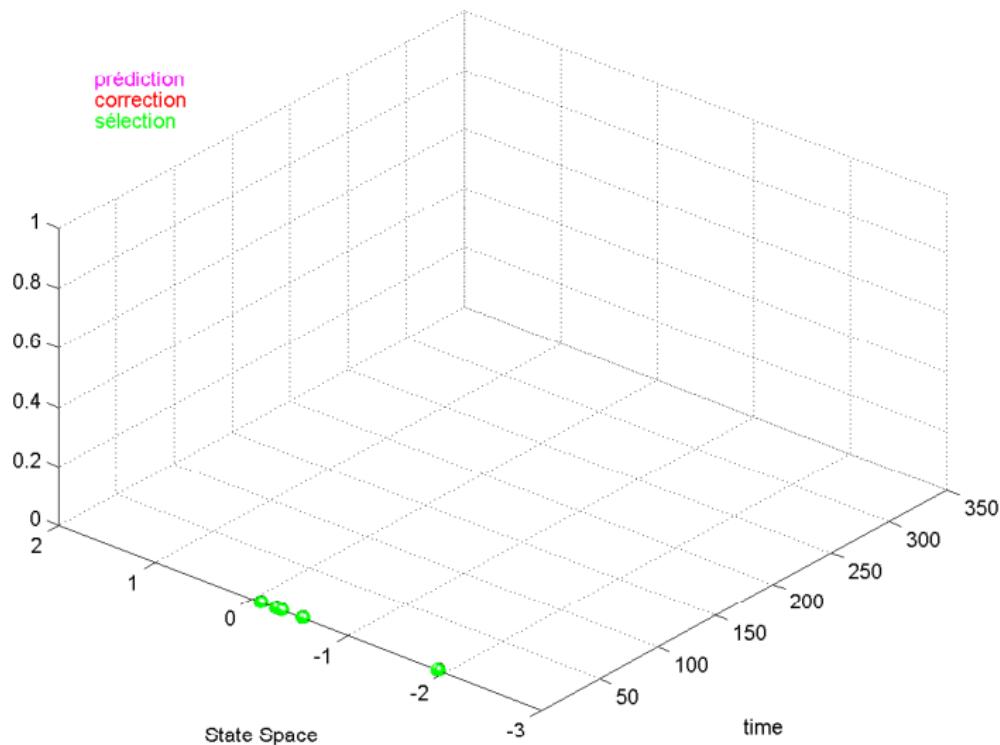
- 1 $r_k = q$ (proposition from the prior dynamics),
- 2 Resampling at each time step

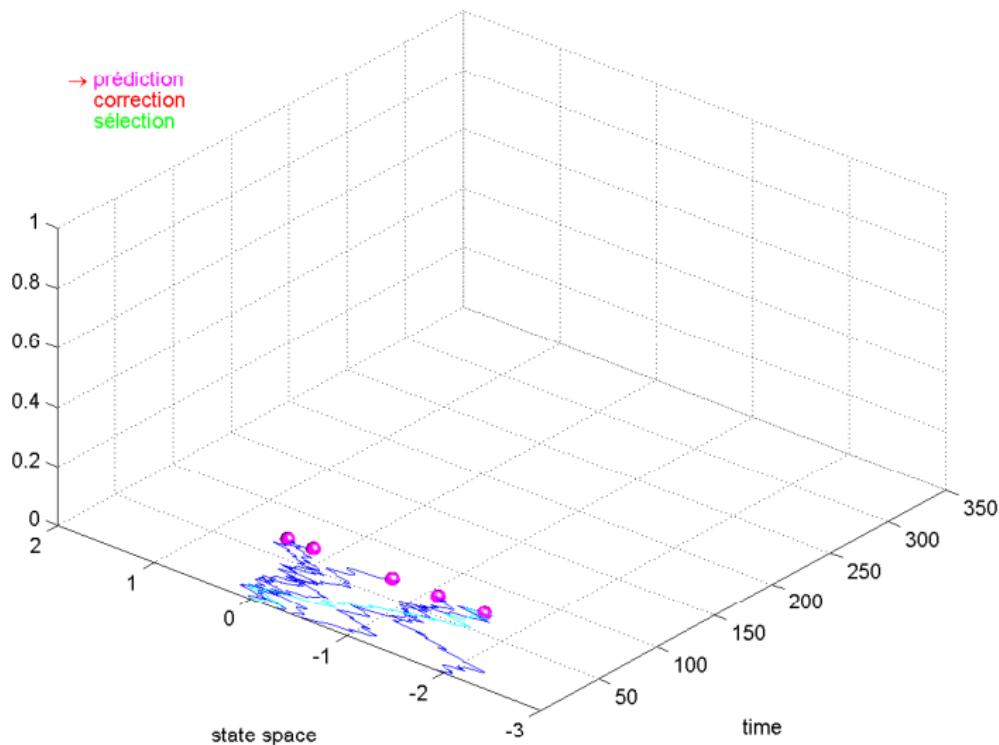
is the simplest yet functional algorithm proposed in the seminal paper of Gordon, Salmond, and Smith (1993).

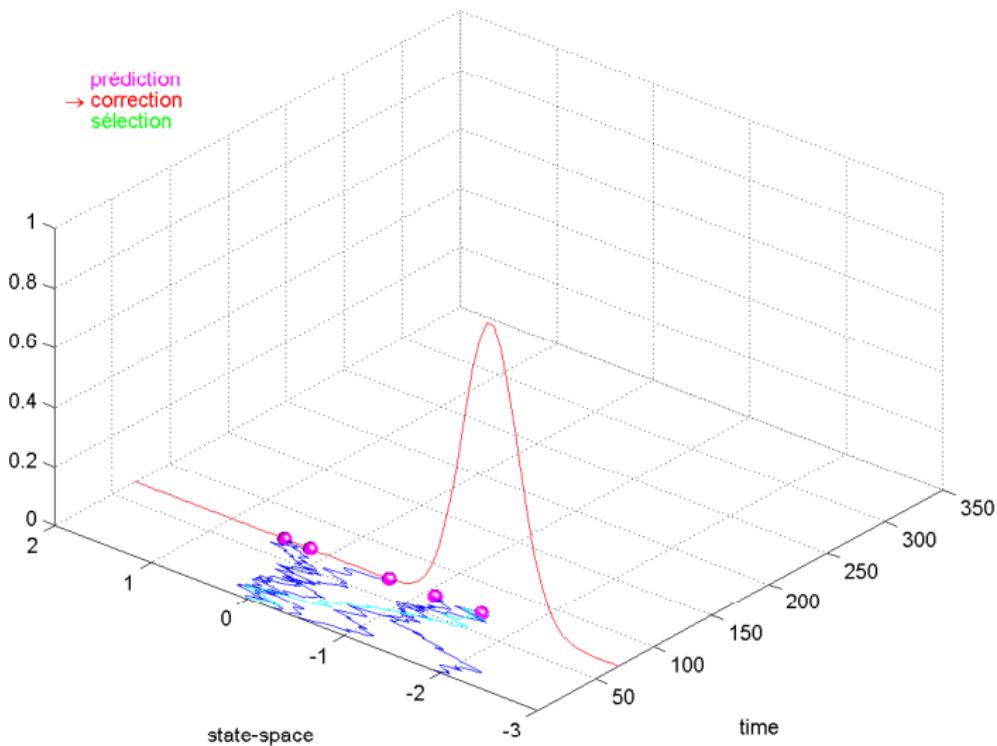
In this simple application of SISR, termed **bootstrap filter**, the unnormalized importance weights are given by

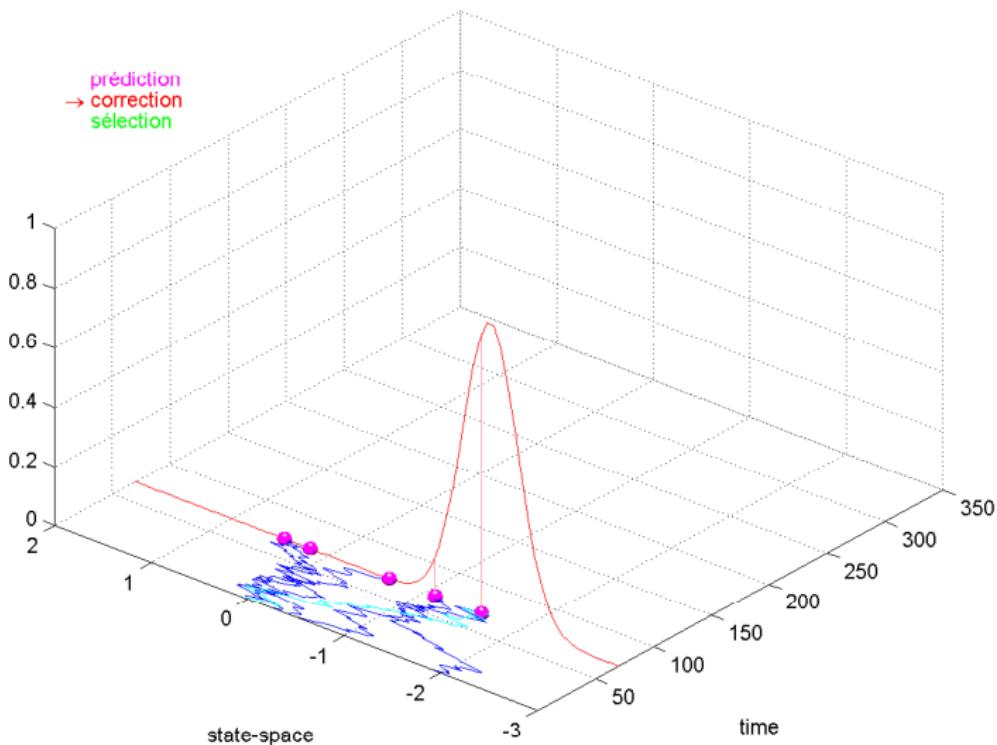
$$\omega_k^i = g(\xi_k^i, Y_k) ,$$

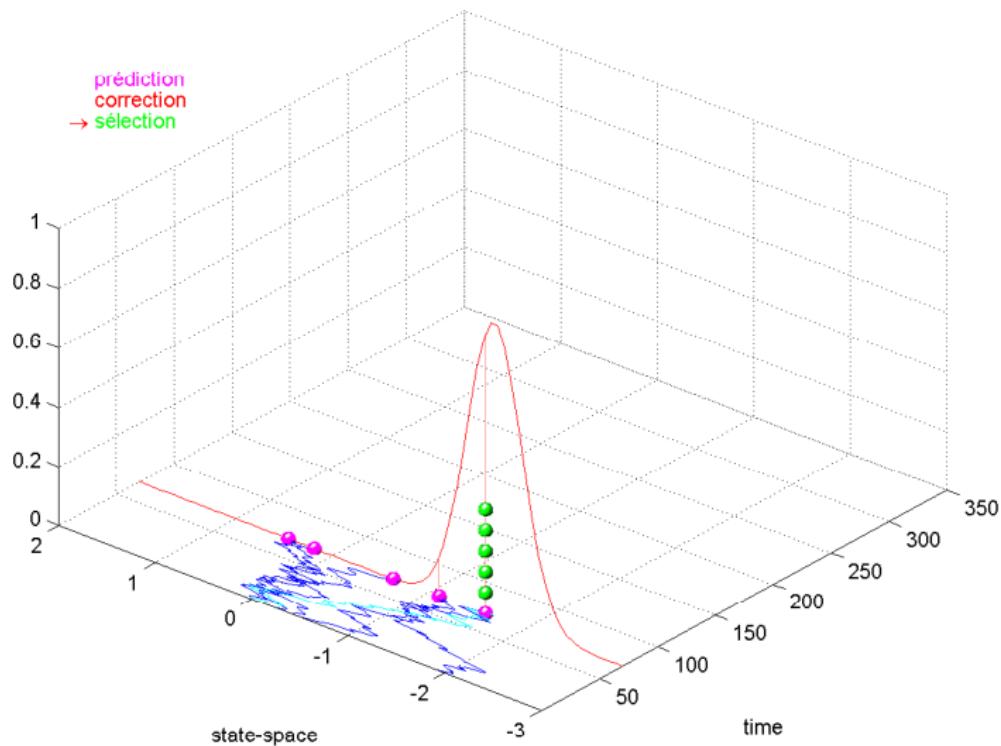
i.e., the likelihood of the current observation conditional upon the particle position ξ_k^i .

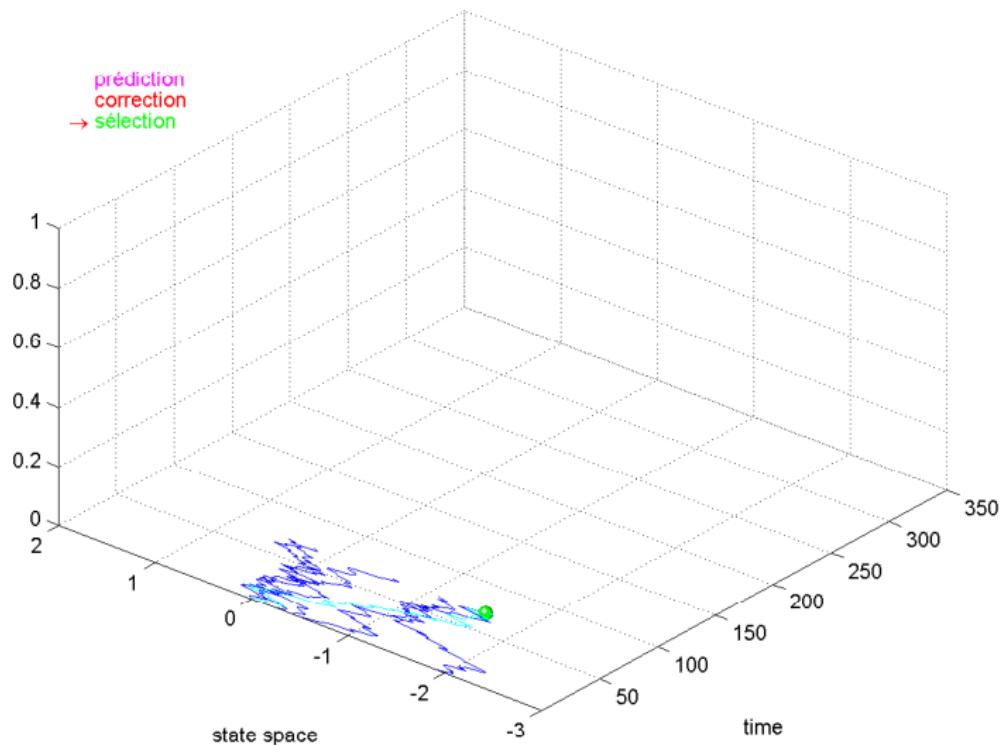


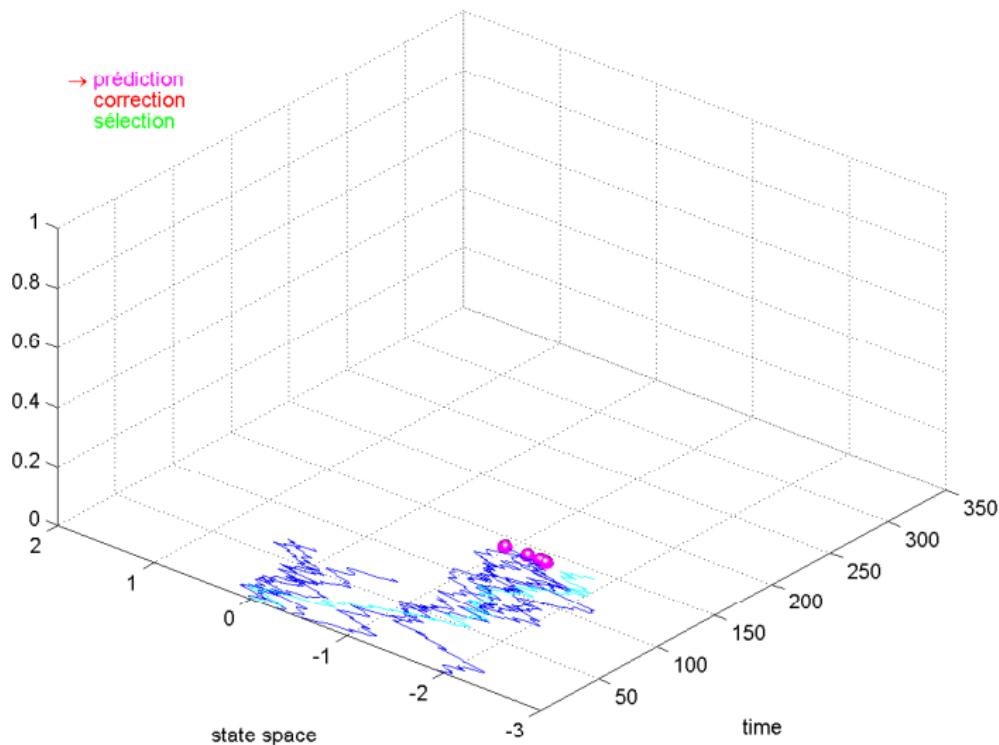


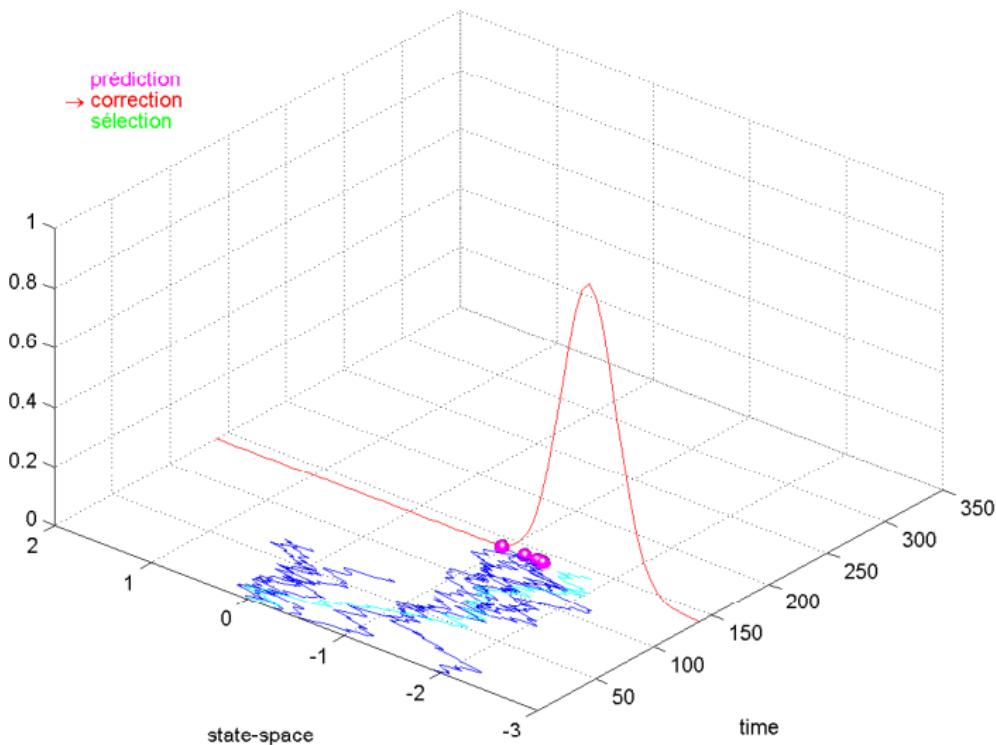


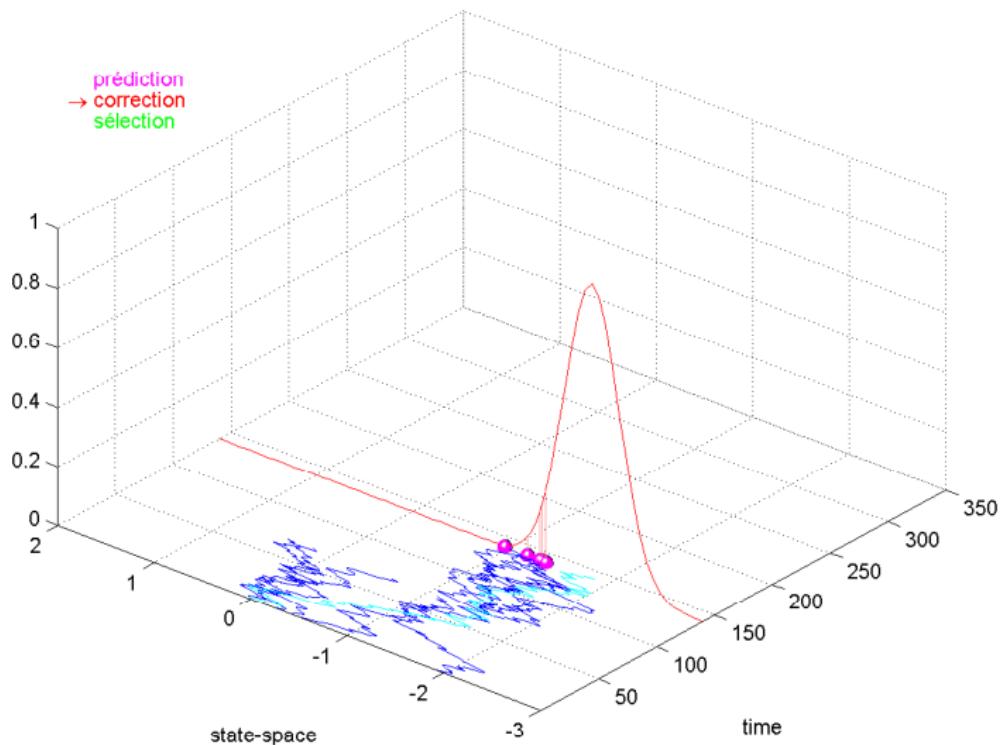


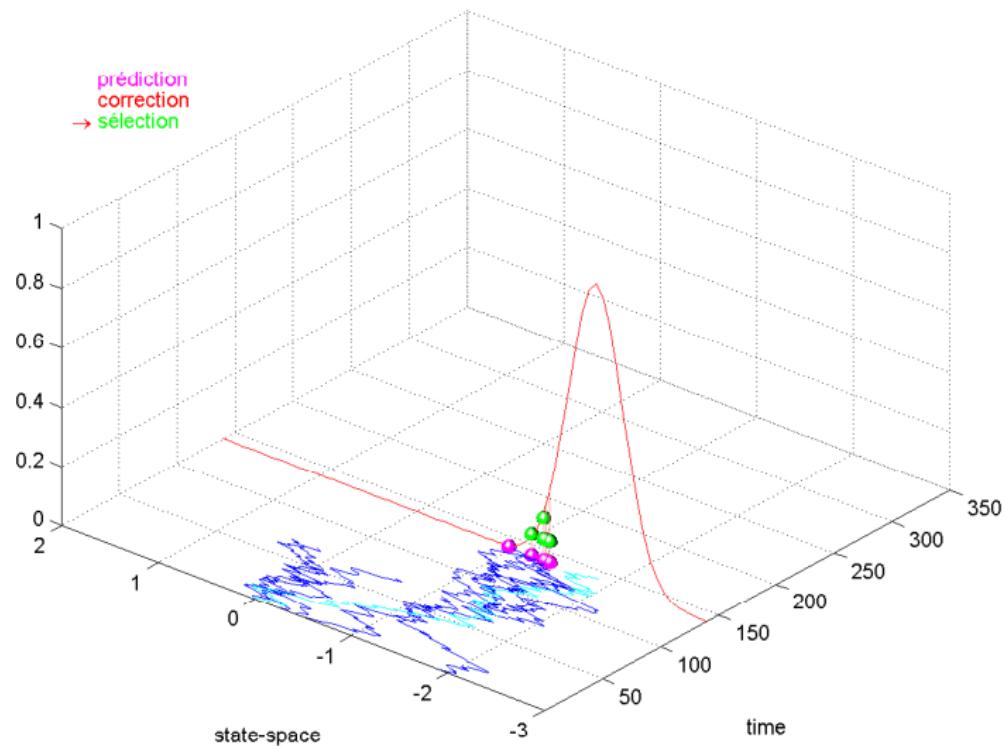


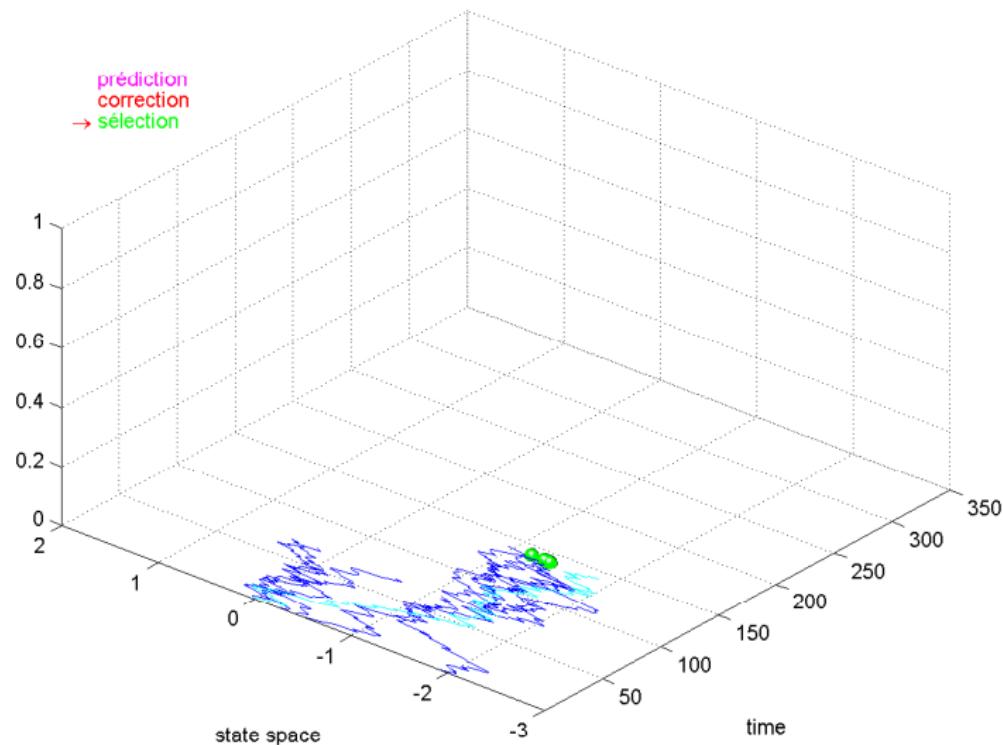


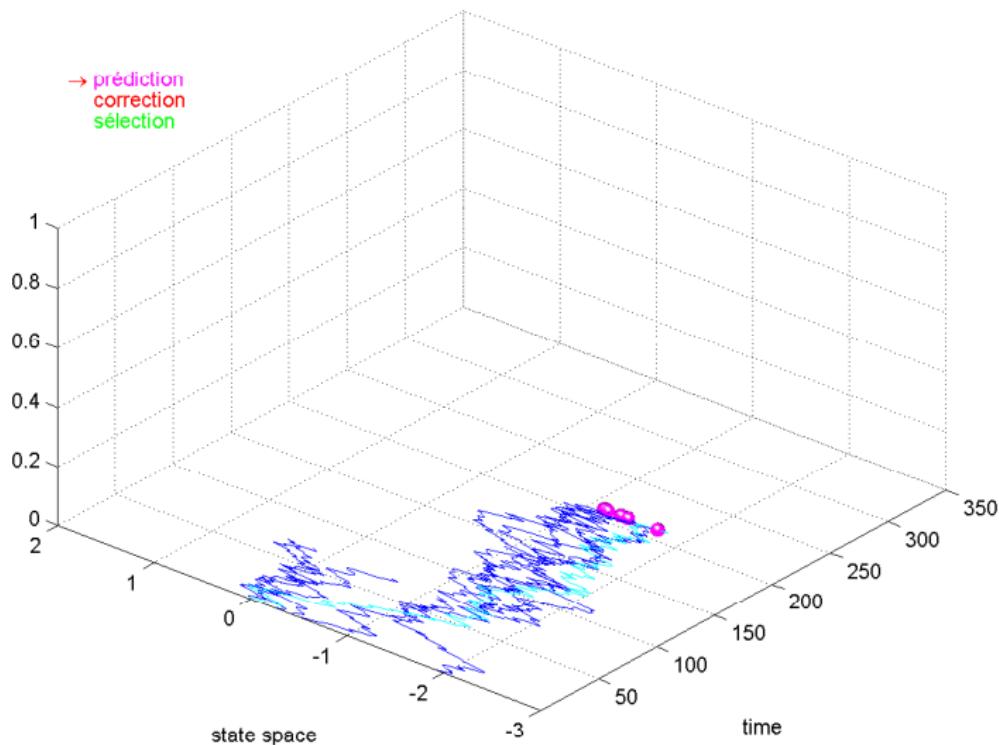


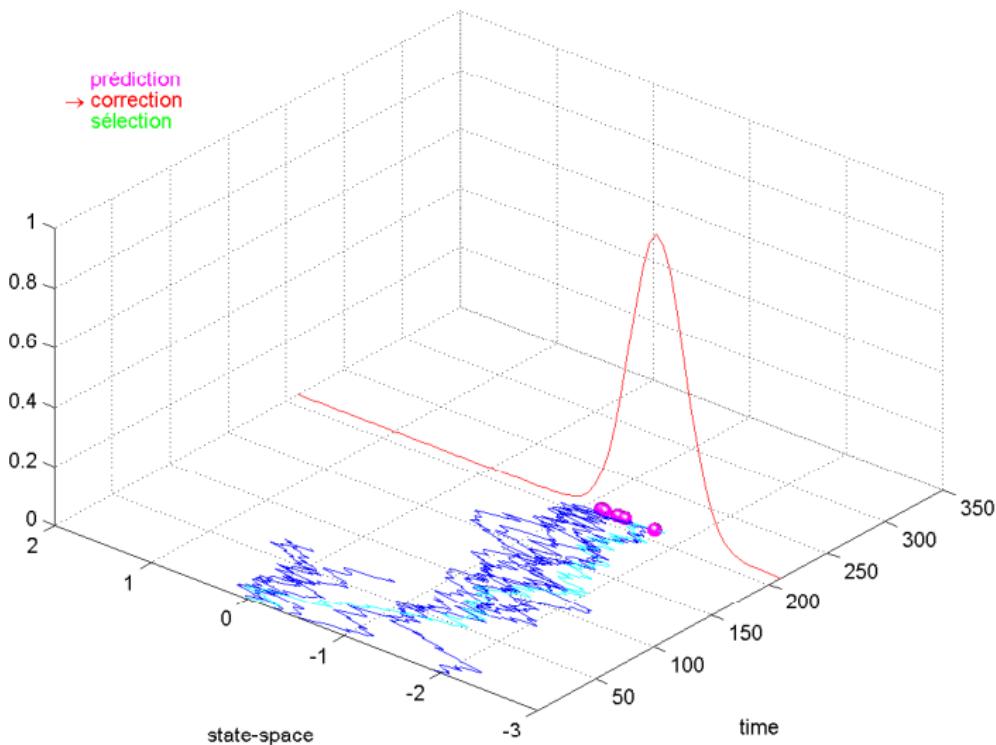


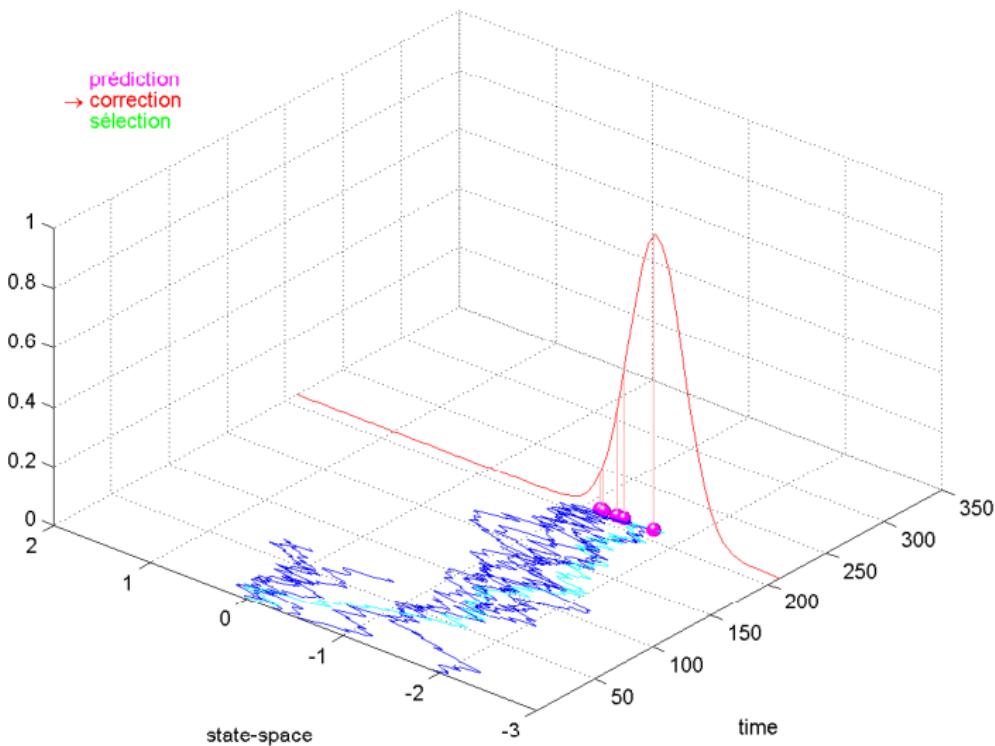


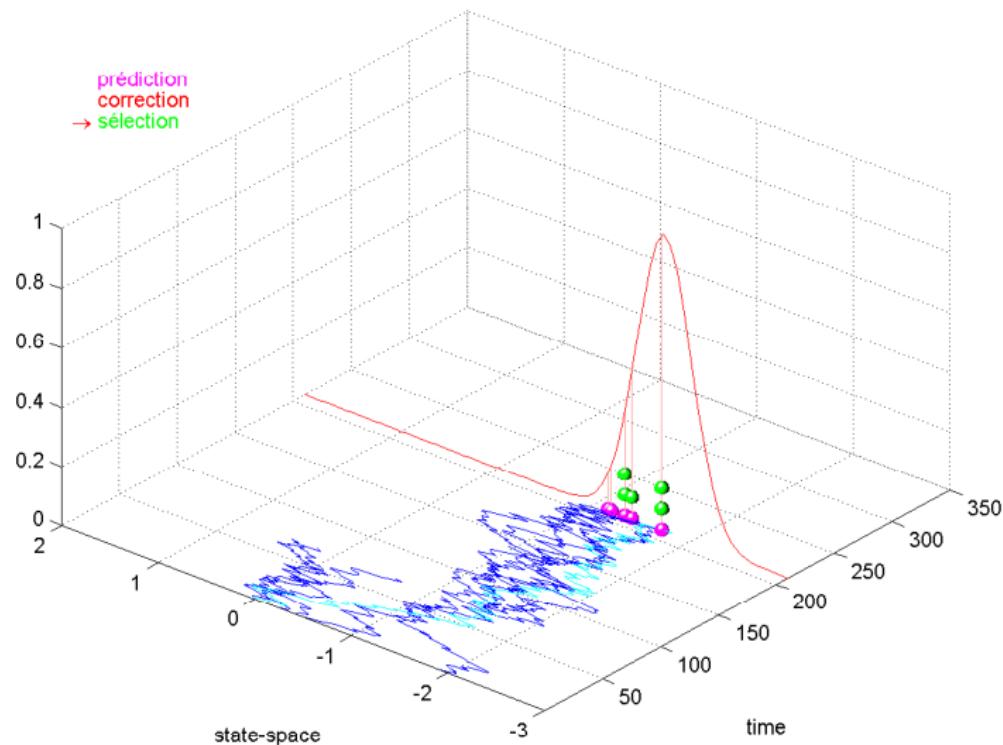


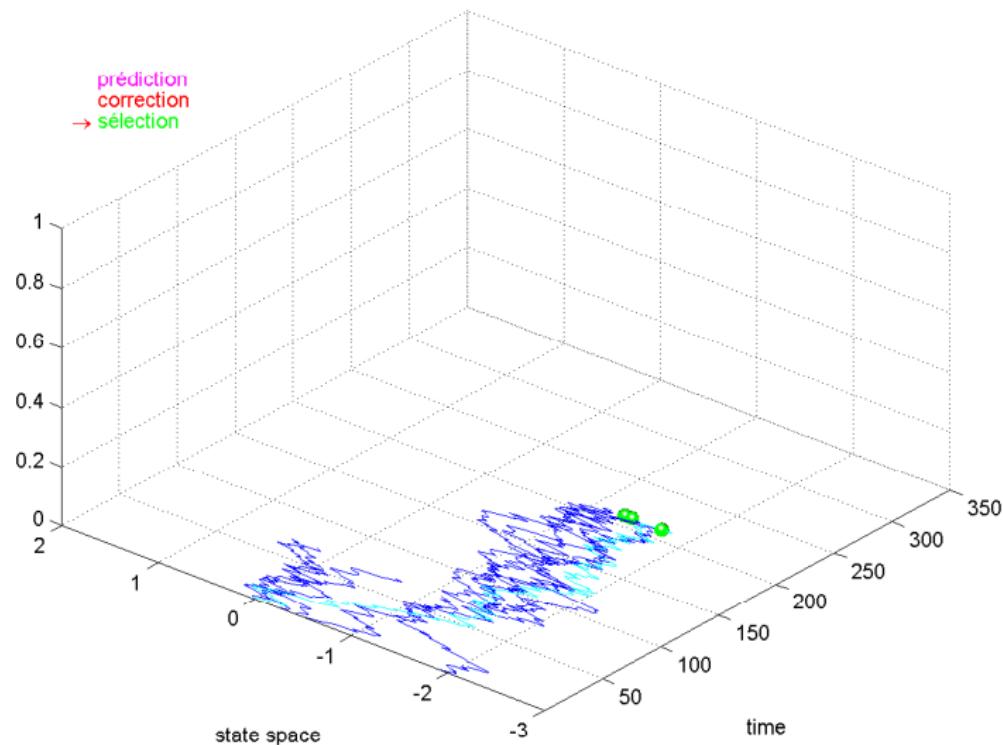


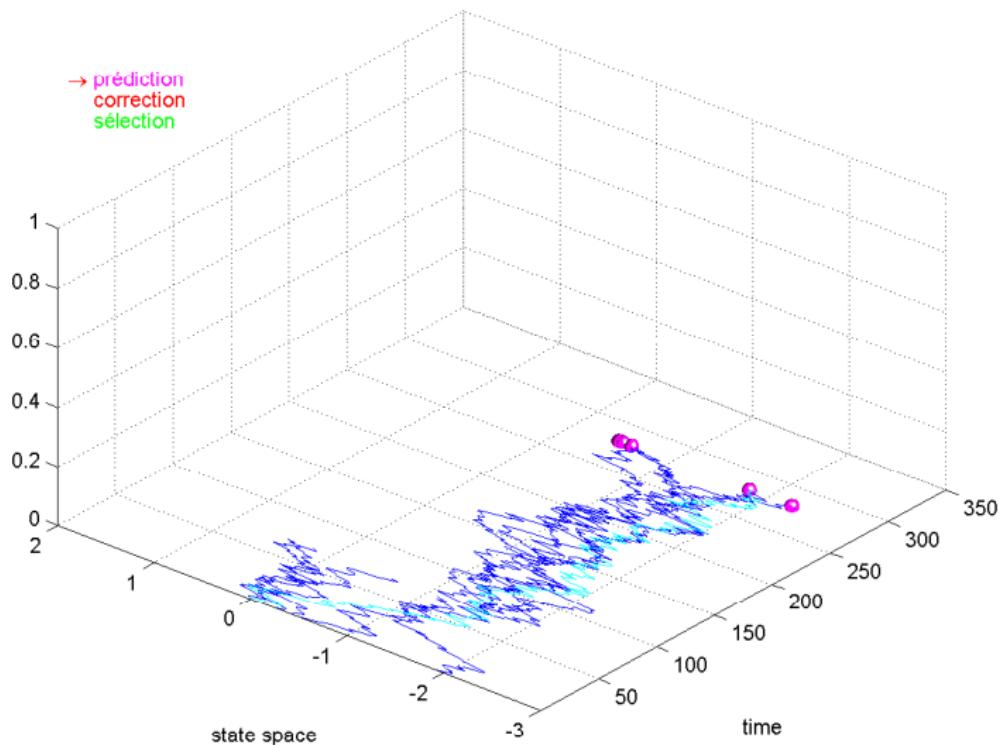


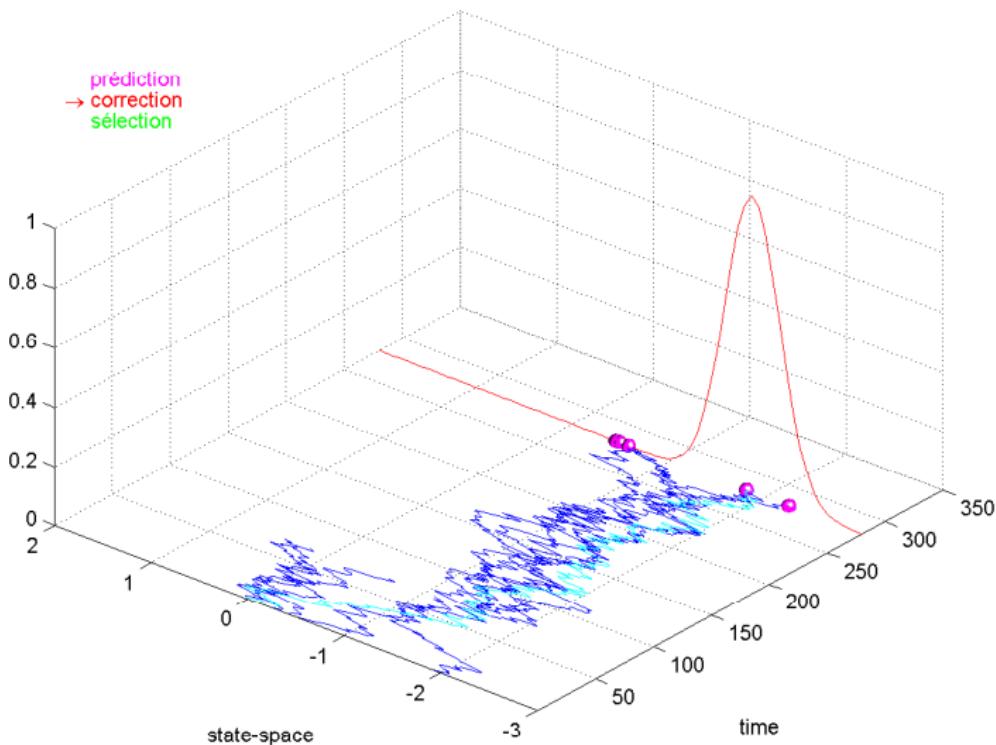


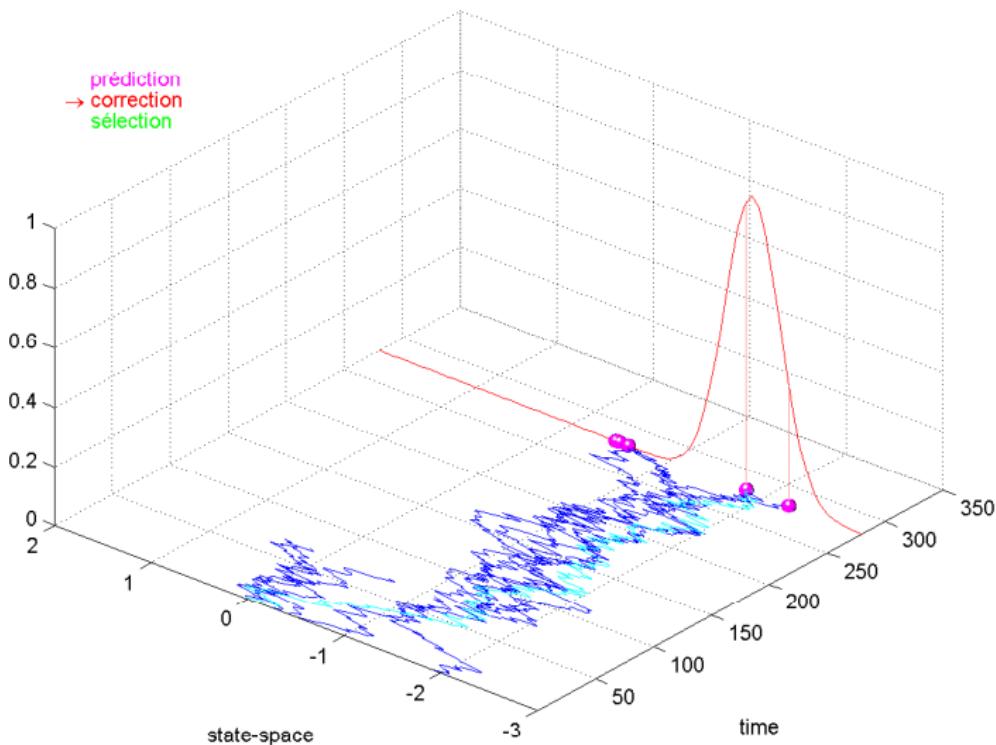


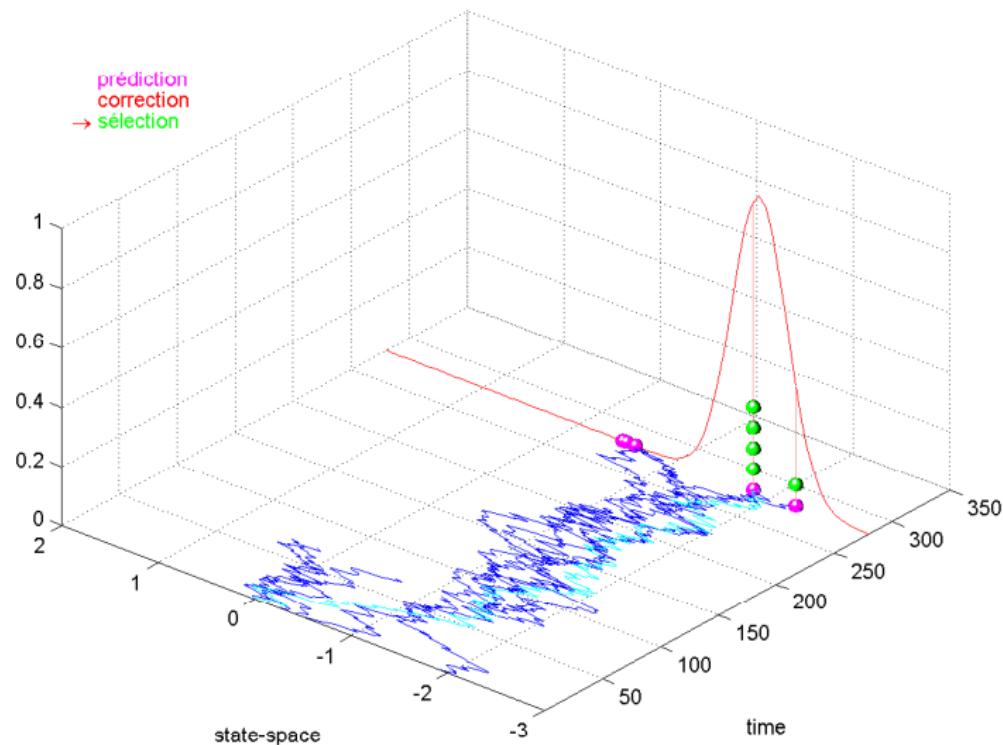


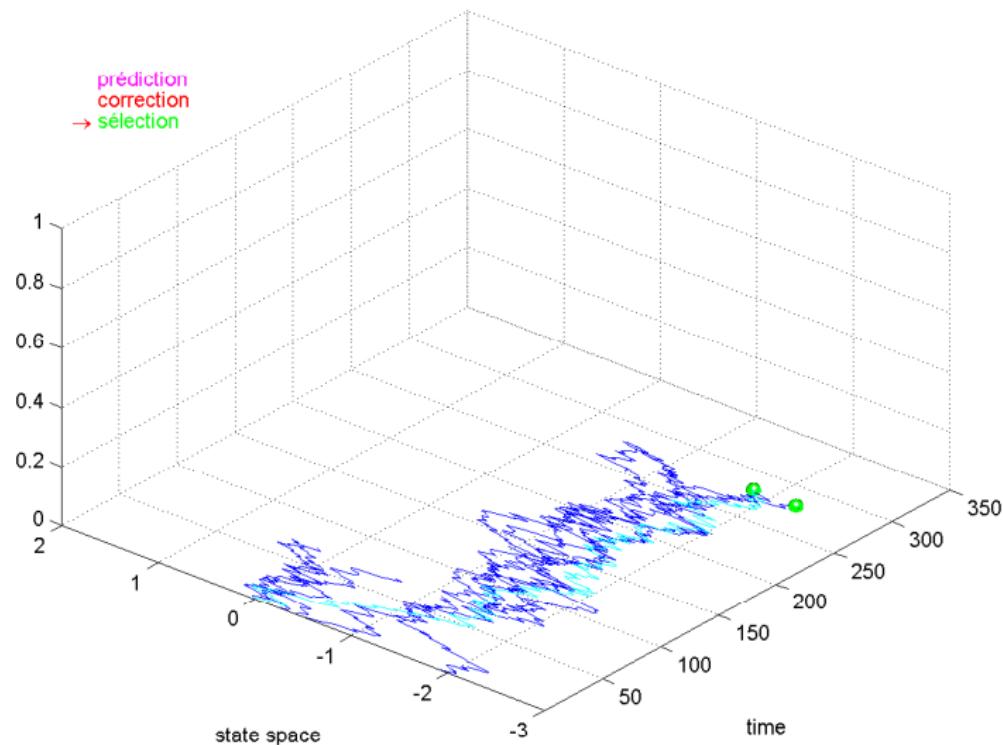


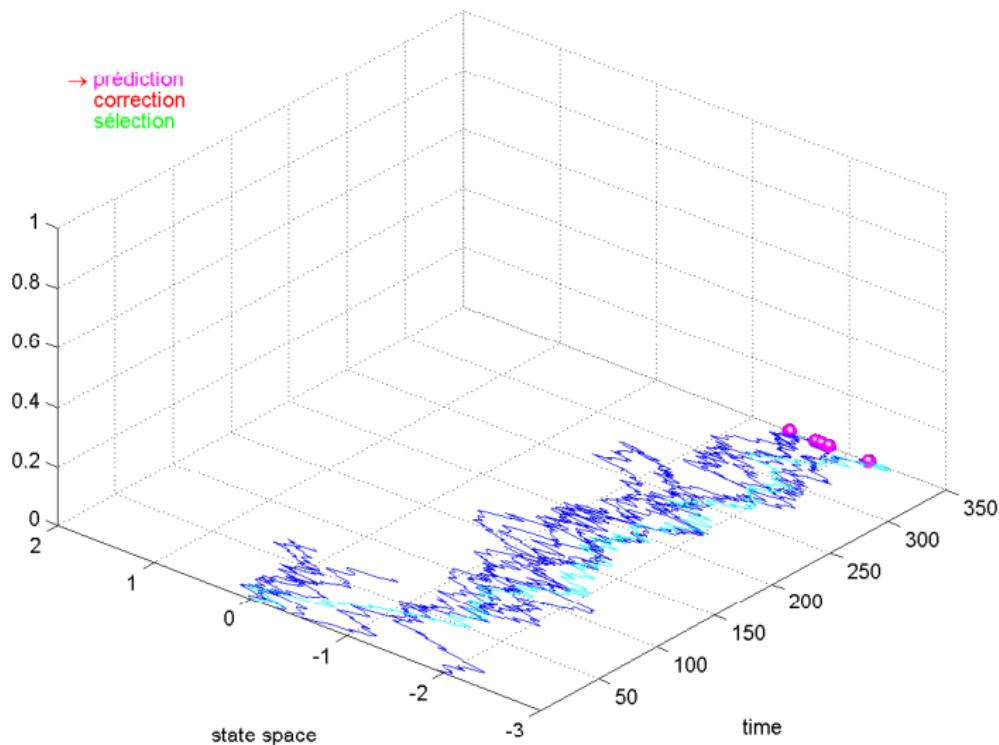


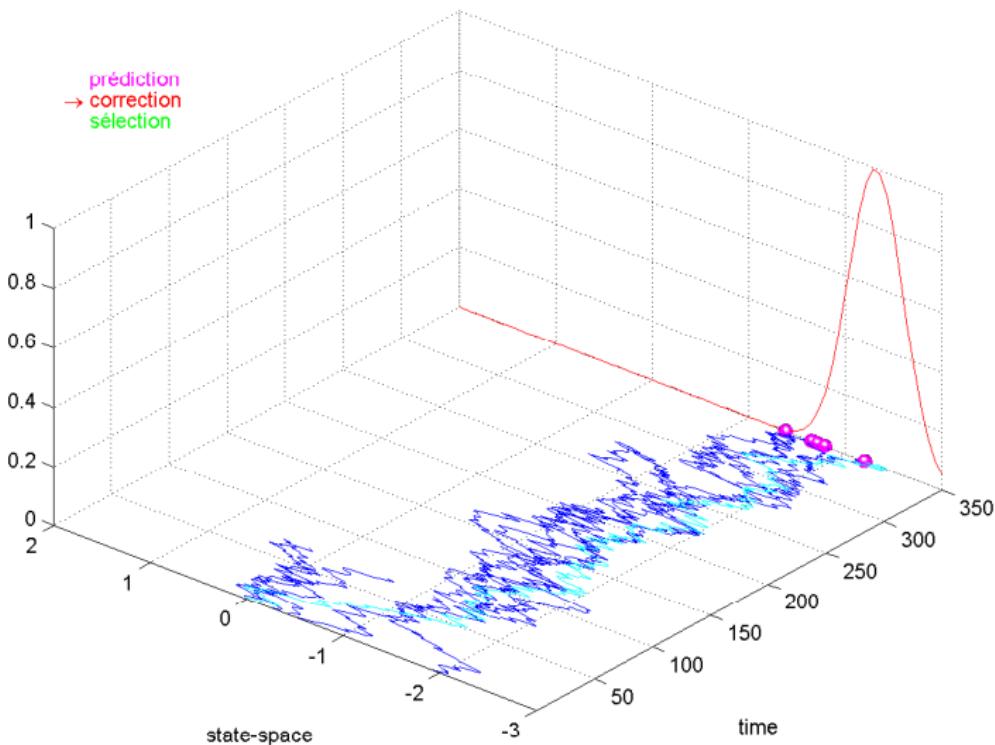


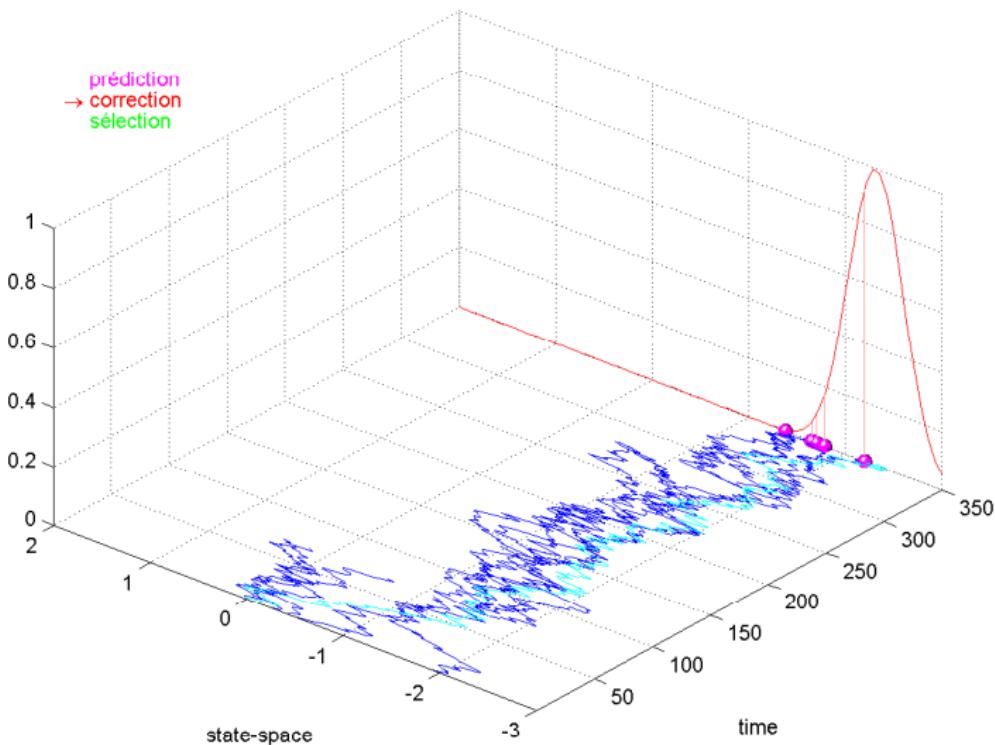












Part III

Case Studies

10 Stochastic Volatility Model

11 Terrain Navigation

- The Navigation Task
- The Model
- Illustrations

12 Tracking of a Region in a Video Sequence

- Aims and Computation of Visual Features
- (Non-Linear) State-Space Modelling

13 Robot Localization

Roadmap

10 Stochastic Volatility Model

11 Terrain Navigation

- The Navigation Task
- The Model
- Illustrations

12 Tracking of a Region in a Video Sequence

- Aims and Computation of Visual Features
- (Non-Linear) State-Space Modelling

13 Robot Localization

Application to the Stochastic Volatility Model

Recall the **stochastic volatility model**

$$\begin{aligned} X_{k+1} &= \phi X_k + \sigma U_k & |\phi| < 1 , \\ Y_k &= \beta \exp(X_k/2) V_k , \end{aligned}$$

where

- 1 $\{U_k\}_{k \geq 0}$ and $\{V_k\}_{k \geq 0}$ are independent standard Gaussian white noise, processes.
- 2 $X_0 \sim \mathcal{N}(0, \sigma^2 / (1 - \phi^2))$.

Implementing the Bootstrap Filter

Is easy as

- 1 The dynamic equation is that of a standard AR model, hence one can simulate ξ_{k+1}^i from

$$q(\tilde{\xi}_k^i, x) = \mathcal{N}(x; \phi \xi_k^i, \sigma^2)$$

- 2 The conditional likelihood of the observation has already been obtained as

$$g_{k+1}(\xi_{k+1}^i) = \frac{1}{\sqrt{2\pi\beta^2}} \exp\left(-\frac{Y_{k+1}^2}{2\beta^2} \exp(-\xi_{k+1}^i) - \frac{1}{2}\xi_{k+1}^i\right)$$

We illustrate the implementation of the algorithm using the example of MATLAB code.

The Bootstrap Filter in Matlab

```
% The Boostrap Filter for Stochastic Volatility.  
%  
% stovolBootstrap(Y, nPart, beta, phi, sigm)  
%     Y(1:nSamp): observations  
%     nPart: number of particles  
%     beta: parameter of the observation equation  
%     phi, sigma: parameters of the state equation  
  
for iSamp = 2:nSamp  
    % Propagating the particles  
    partPos = phi*partPos + sigm*randn(nPart,1);  
    % Computing the normalized weights  
    weight = exp( ...  
        -0.5*(partPos+(Y(iSamp))^2/beta^2*exp(-partPos)) );  
    weight = weight/sum(weight);  
    % Resampling  
    ind = resampling(weight, nPart, 'indp');  
    partPos = partPos(ind);  
end
```

Resampling

```
% I = resample(p, n [, method]); where method is one of
%      'indp' (independent)
%      'stra' (stratified)

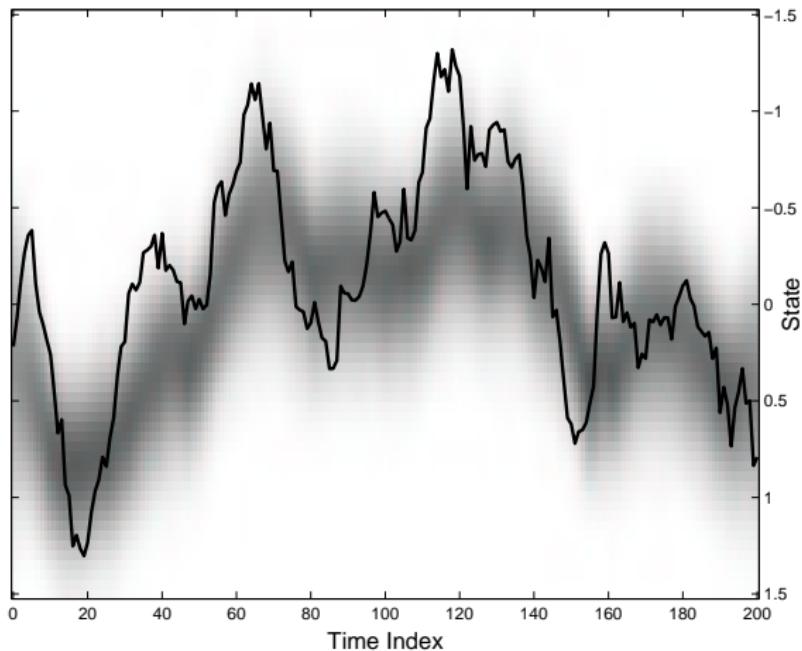
switch lower(method)
    case 'indp'
        % 1. Independent (or multinomial) resampling
        U = sort(rand(1,n));
    case 'stra'
        % 2. Stratified resampling
        U = rand(1, n);
        U = ((0:n-1)+U)/n;
end
I = invertCDF(p, U);
```

Bulk Inversion of the Discrete CDF (With Linear Cost)

```
% Inverts a discrete cumulative distribution function for
% an input vector of *increasing* numbers between 0 and 1.
% I = invertCDF(p, U). In: p probability vector, U
% vector of increasing numbers between 0 and 1; Out:
% I vector of indices between 1 and length(p).

n = length(U);
% Compute discrete CDF
F = cumsum(p);
% Compute (increasing) indices
I = zeros(n, 1);
for t = 1:n
    if (t == 1)
        I(t) = 1;
    else
        I(t) = I(t-1);
    end
    while (U(t) >= F(I(t)))
        I(t) = I(t)+1;
    end
end
```

With resampling the Particle Filter is Stable on the Long-Term



Actual state and particle densities (grey level) using $N = 1,000$ particles.

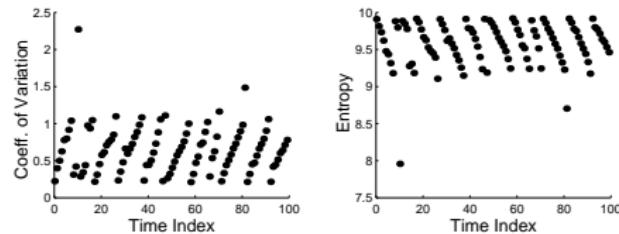
Optional Resamplings

In usual cases, the variance of the approximation can be significantly reduced by making resampling less frequent:

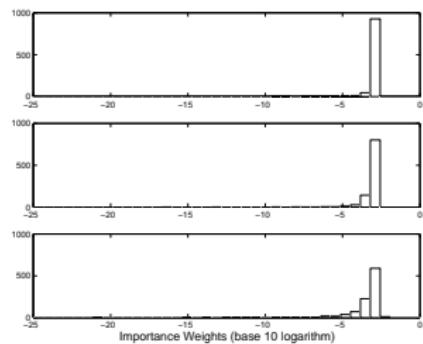
```
for iSamp = 2:nSamp
    % Propagating the particles
    partPos = phi*partPos + sigm*randn(nPart,1);
    % Computing the normalized weights
    weight = weight .* exp( ...
        -0.5*(partPos+(Y(iSamp))^2/beta^2*exp(-partPos)) );
    weight = weight/sum(weight);
    % Optional resampling
    if (RESAMPLING_FLAG)
        ind = resampling(weight, nPart, 'indp');
        partPos = partPos(ind);
        weight = ones(1, nPart)/nPart;
    end
end
```

Example

One can use the criteria discussed about importance sampling (CV, ESS, Entropy) to trigger sampling based on the current distribution of particle weights



Coefficient of variation (left) and binary entropy of the normalized importance weights as a function of the number of iterations when using resampling triggered by $\text{CV}_N(\omega) > 1$.



Histograms of the base 10 logarithm of the normalized importance weights after (from top to bottom) 1, 10 and 100 iterations.

Roadmap

10 Stochastic Volatility Model

11 Terrain Navigation

- The Navigation Task
- The Model
- Illustrations

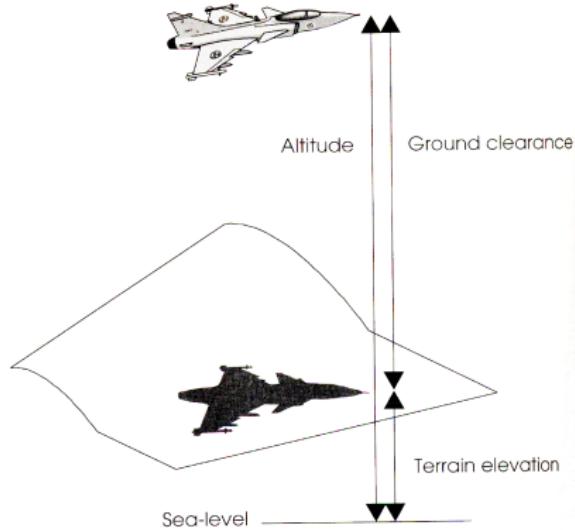
12 Tracking of a Region in a Video Sequence

- Aims and Computation of Visual Features
- (Non-Linear) State-Space Modelling

13 Robot Localization

Terrain Elevation Measurement

The aircraft altitude (over mean sea level) is measured by a pressure meter and the ground clearance distance by a radar altimeter. The difference is treated as measurement of the **terrain elevation** beneath the aircraft.



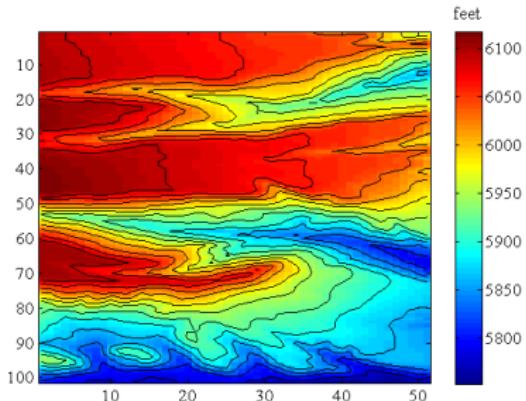
Reference Map

The measurement is compared to a digital terrain elevation map, and the matching positions in the map are used to determine aircraft position estimates.

The elevation map is viewed as a **highly nonlinear** function of the 2D position

$$b : (x, y) \rightarrow h$$

(in practice, it is known in a finite number of points and interpolated in between).



State and Observation Equations

- The unobservable state consists of the plane position and speed components, projected onto the map

$$X_k = [X_{1,k} \quad X_{2,k} \quad \dot{X}_{1,k} \quad \dot{X}_{2,k}]^T$$

- The observation Y_k is the measured elevation at position $(X_{1,k}, X_{2,k})$ modelled as

$$Y_k = b \left([1 \quad 0 \quad 0 \quad 0] X_k, [0 \quad 1 \quad 0 \quad 0] X_k \right) + V_k$$

where V_k is a Gaussian measurement noise.

Linear Dynamic Modelling by Second Order Taylor Expansion

The speed components are assumed to be mostly constant over a sampling period and follow a random walk model; the dynamic model is then obtained using a second order Taylor expansion:

$$x_1((k+1)T) = x_1(kT) + \dot{x}_1(kT)T + \frac{T^2}{2}\epsilon_{1,x}$$

$$\dot{x}_1((k+1)T) = \dot{x}_1(kT) + T\epsilon_{1,x}$$

Linear Dynamic Modelling by Second Order Taylor Expansion

The speed components are assumed to be mostly constant over a sampling period and follow a random walk model; the dynamic model is then obtained using a second order Taylor expansion:

$$\begin{aligned}x_1((k+1)T) &= x_1(kT) + \dot{x}_1(kT)T + \frac{T^2}{2}\epsilon_{1,x} \\ \dot{x}_1((k+1)T) &= \dot{x}_1(kT) + T\epsilon_{1,x}\end{aligned}$$

Hence,

$$\begin{bmatrix} X_{1,k} \\ X_{2,k} \\ X_{3,k} \\ X_{4,k} \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{1,k-1} \\ X_{2,k-1} \\ X_{3,k-1} \\ X_{4,k-1} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix} \begin{bmatrix} \epsilon_{1,k} \\ \epsilon_{2,k} \end{bmatrix}$$

Nonlinear Dynamic Modelling Assuming Constant Speed and Course (Variant)

In this alternative model the speed (modulus) and course are assumed to be quasi constant, both following a random walk model:

$$\text{Course: } C_{k+1} = \arg \tan(X_{2,k}/X_{1,k}) + \epsilon_{1,k}$$

$$\text{Speed: } S_{k+1} = \sqrt{X_{3,k}^2 + X_{4,k}^2} + \epsilon_{2,k}$$

Nonlinear Dynamic Modelling Assuming Constant Speed and Course (Variant)

In this alternative model the speed (modulus) and course are assumed to be quasi constant, both following a random walk model:

$$\text{Course: } C_{k+1} = \arg \tan(X_{2,k}/X_{1,k}) + \epsilon_{1,k}$$

$$\text{Speed: } S_{k+1} = \sqrt{X_{3,k}^2 + X_{4,k}^2} + \epsilon_{2,k}$$

$$X_{1,k+1} = X_{1,k} + T X_{3,k+1}$$

$$X_{2,k+1} = X_{2,k} + T X_{4,k+1}$$

$$X_{3,k+1} = S_{k+1} \cos(C_{k+1})$$

$$X_{4,k+1} = S_{k+1} \sin(C_{k+1})$$

Bootstrap Filter (For the Linear Dynamic Model)

- To propagate the i -th particle, one uses the dynamic model

$$\xi_k^i = A_k \xi_{k-1}^i + R_k U_k$$

where U_k is a two-dimensional Gaussian vector.

Bootstrap Filter (For the Linear Dynamic Model)

- To propagate the i -th particle, one uses the dynamic model

$$\xi_k^i = A_k \xi_{k-1}^i + R_k U_k$$

where U_k is a two-dimensional Gaussian vector.

- The weights are computed from the conditional likelihood function

$$\omega_k^i \propto \omega_{k-1}^i \exp \left(-\frac{1}{2\sigma^2} (Y_k - b(\xi_k^i))^2 \right)$$

(where $b(\xi_k^i)$ is obtained from the elevation map).

Bootstrap Filter (For the Linear Dynamic Model)

- To propagate the i -th particle, one uses the dynamic model

$$\xi_k^i = A_k \xi_{k-1}^i + R_k U_k$$

where U_k is a two-dimensional Gaussian vector.

- The weights are computed from the conditional likelihood function

$$\omega_k^i \propto \omega_{k-1}^i \exp\left(-\frac{1}{2\sigma^2}(Y_k - b(\xi_k^i))^2\right)$$

(where $b(\xi_k^i)$ is obtained from the elevation map).

- Resampling is required to avoid weight degeneracy.

Demo in MATLAB | demos/progs_LevelMap

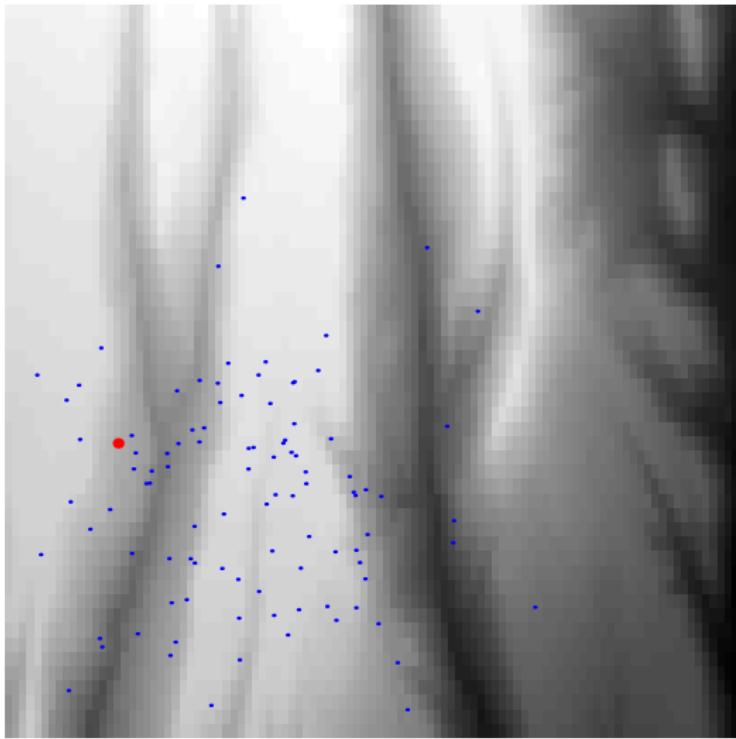
- Results with resampling.

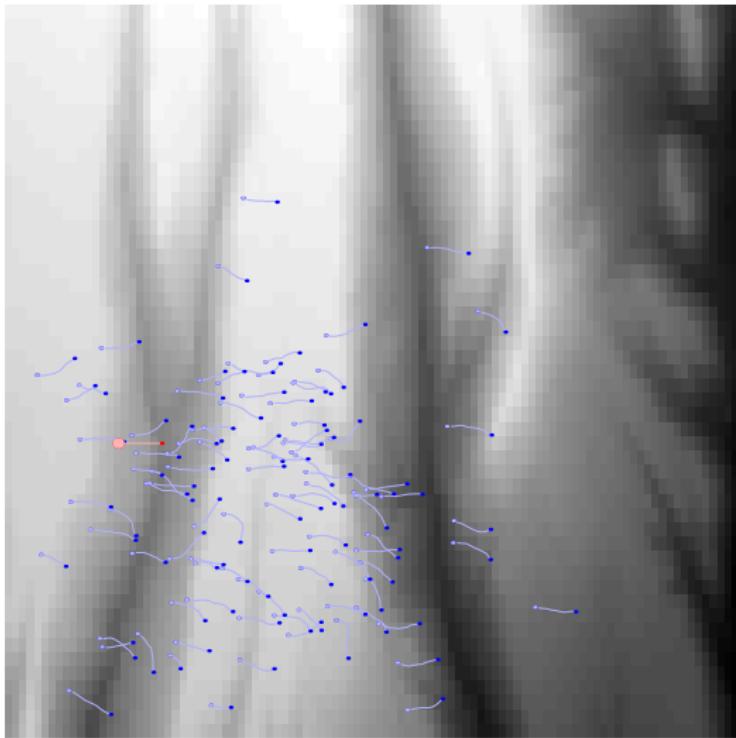
Demo in MATLAB | demos/progs_LevelMap

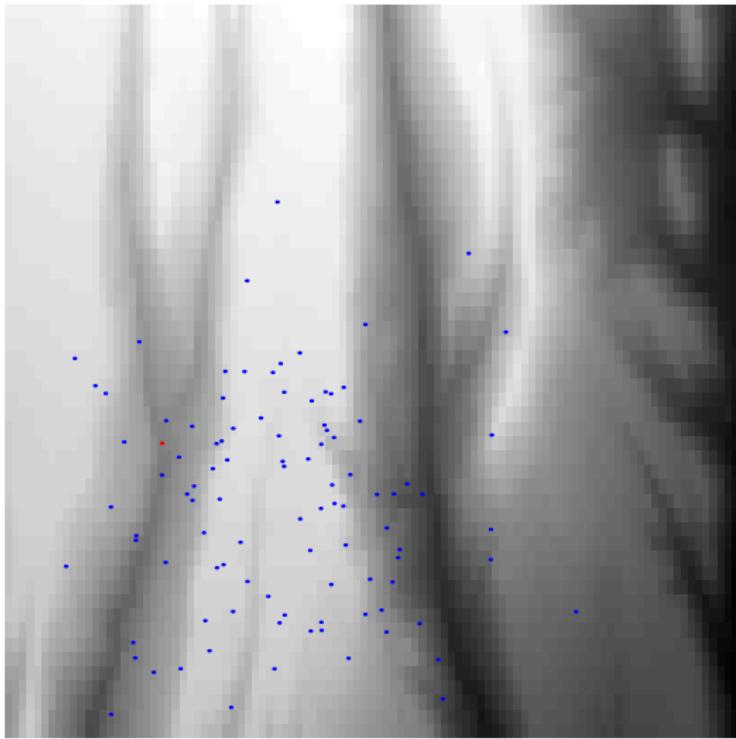
- Results with resampling.
- Comparison with results obtained without resampling (using basic SIS).

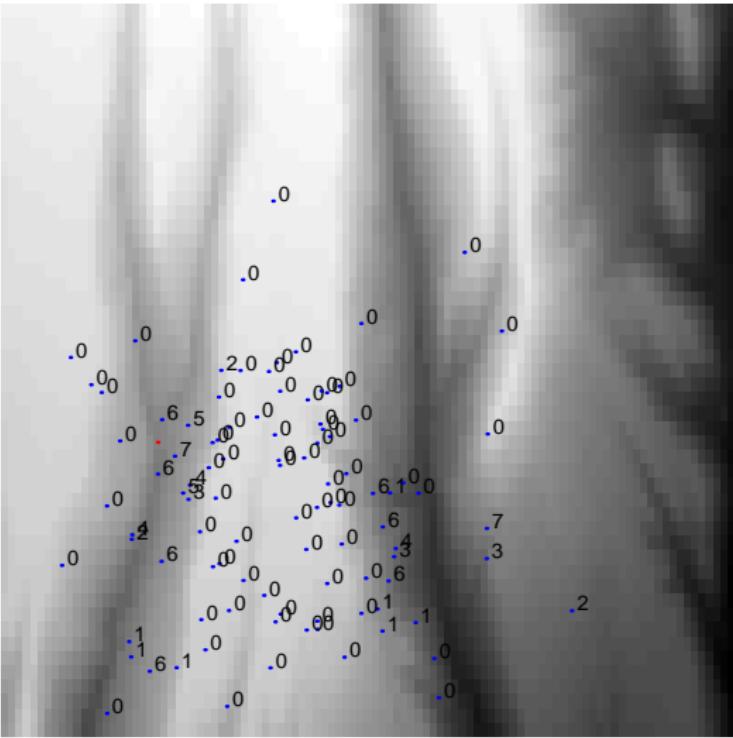
Illustration of the Resampling Process

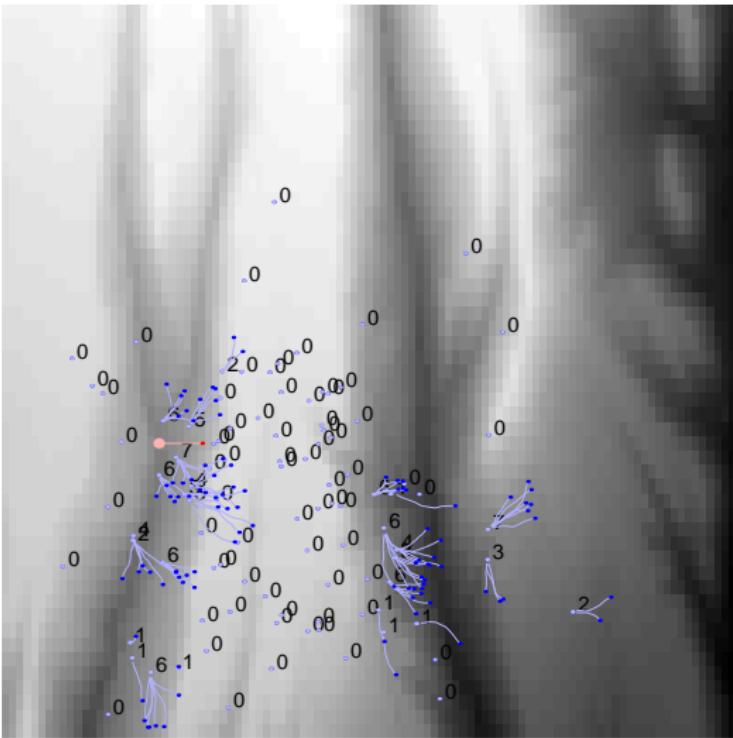
Simulation results by F. Campillo using the second (non-linear) dynamic model.

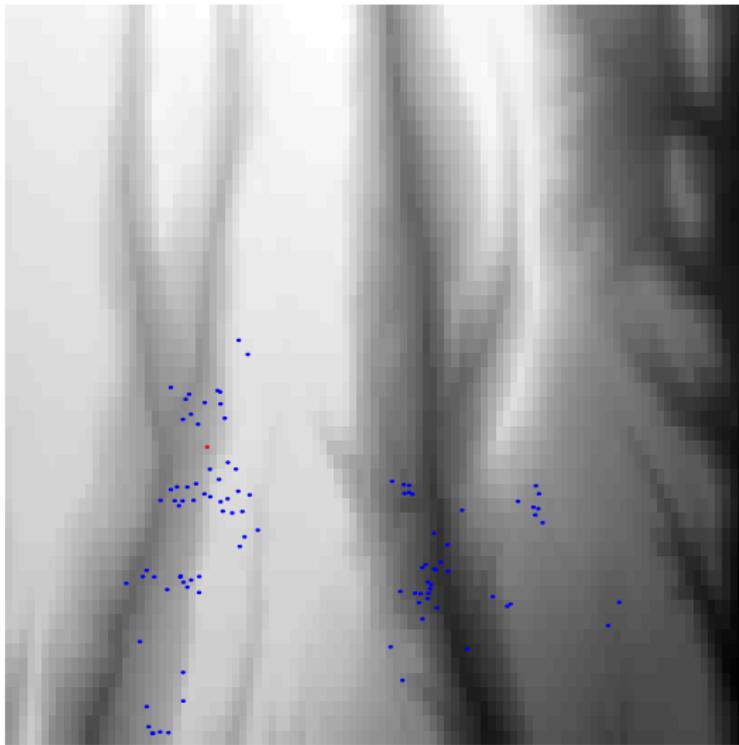


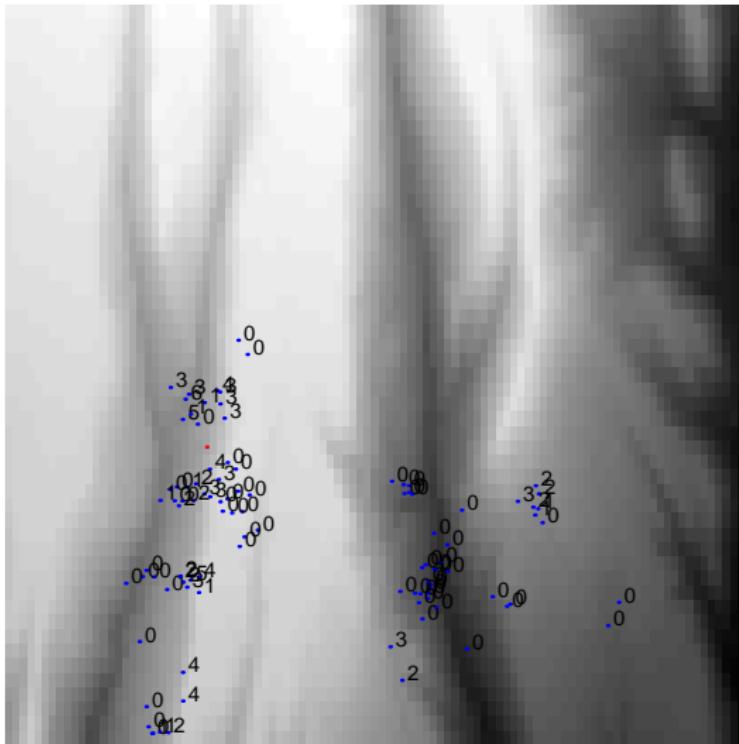


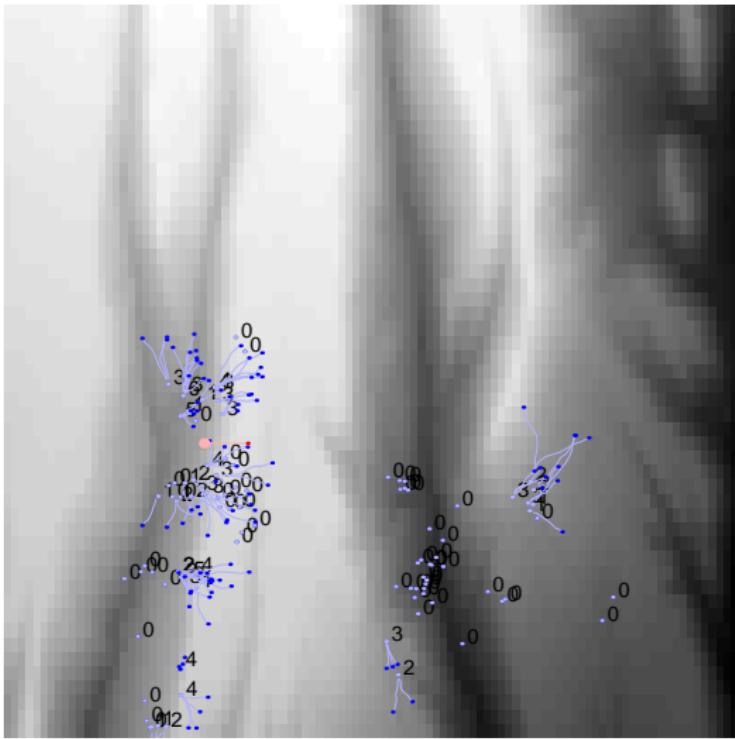


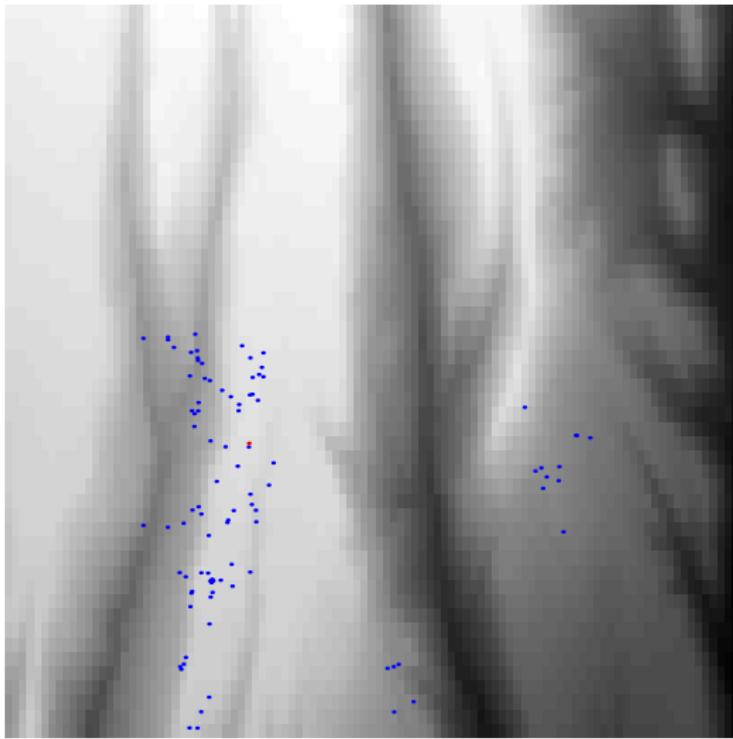


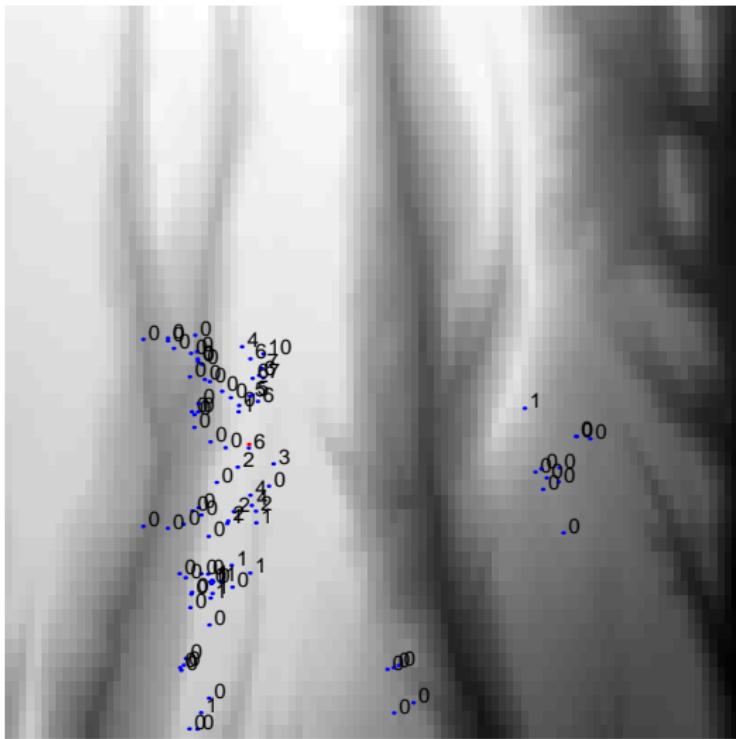


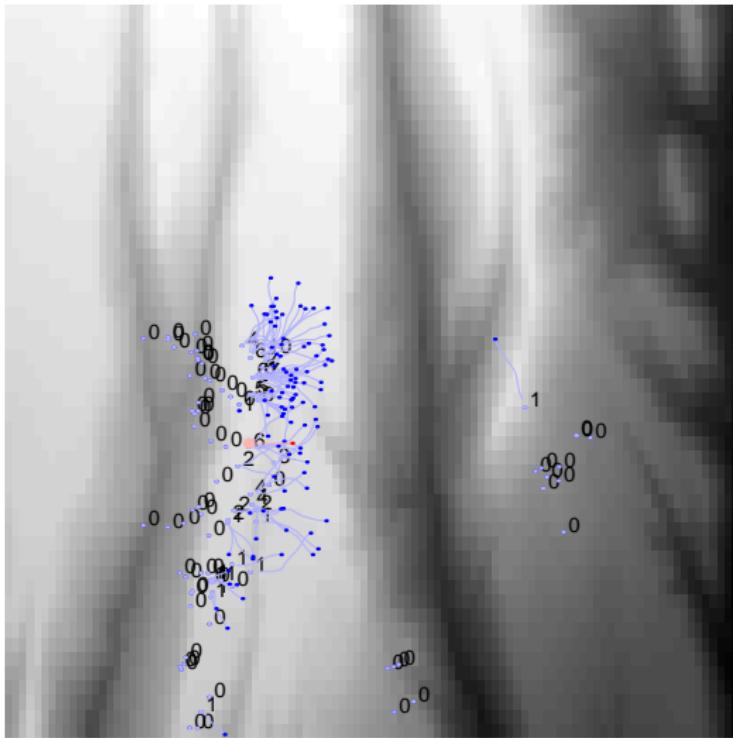


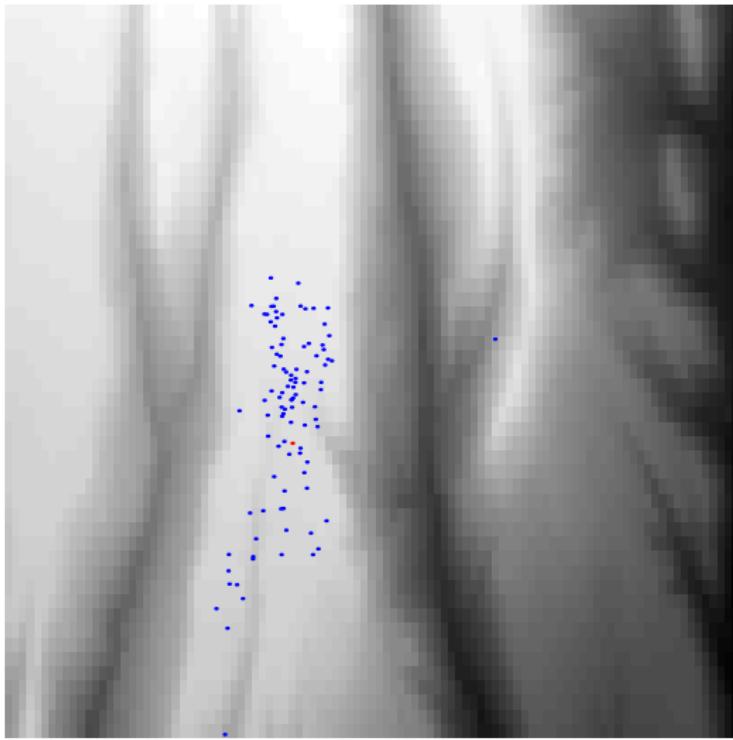


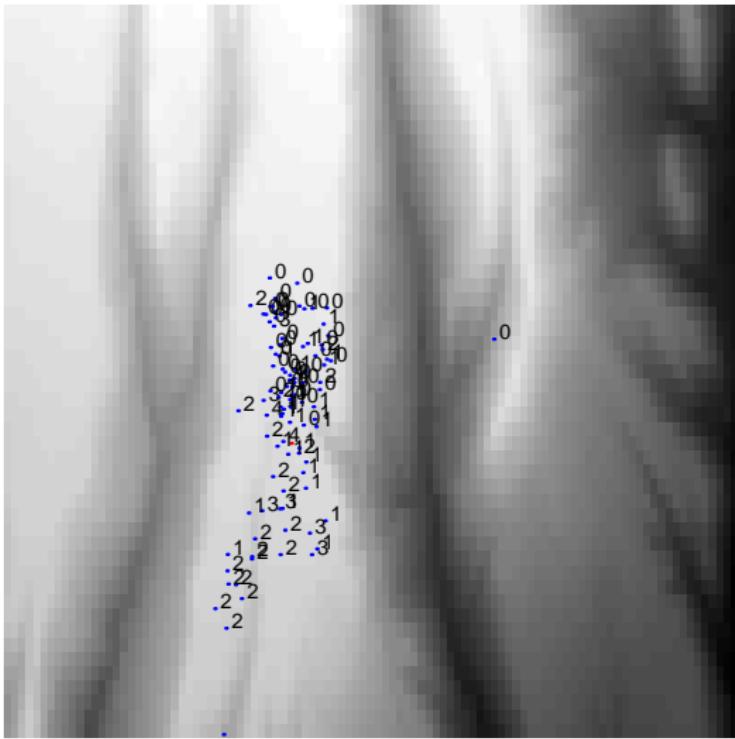


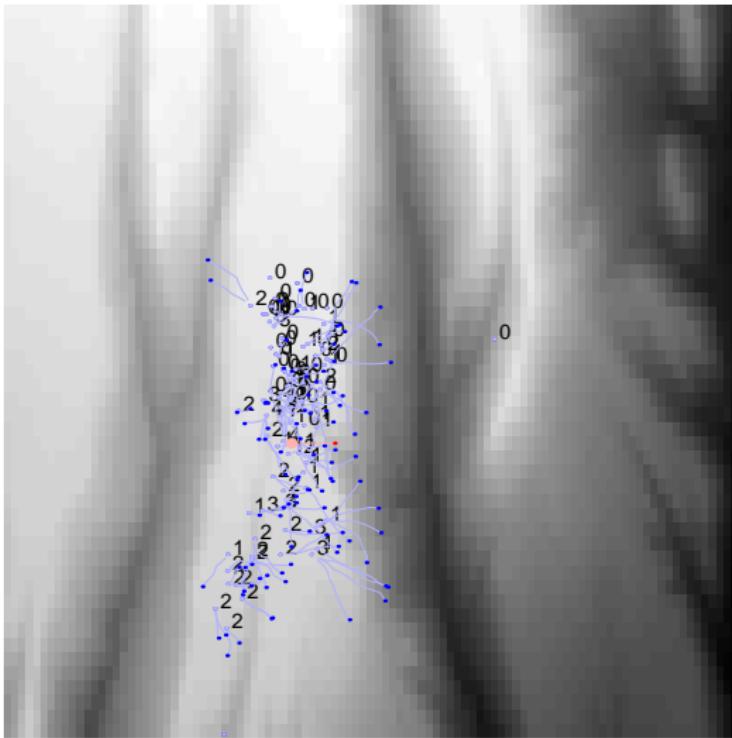


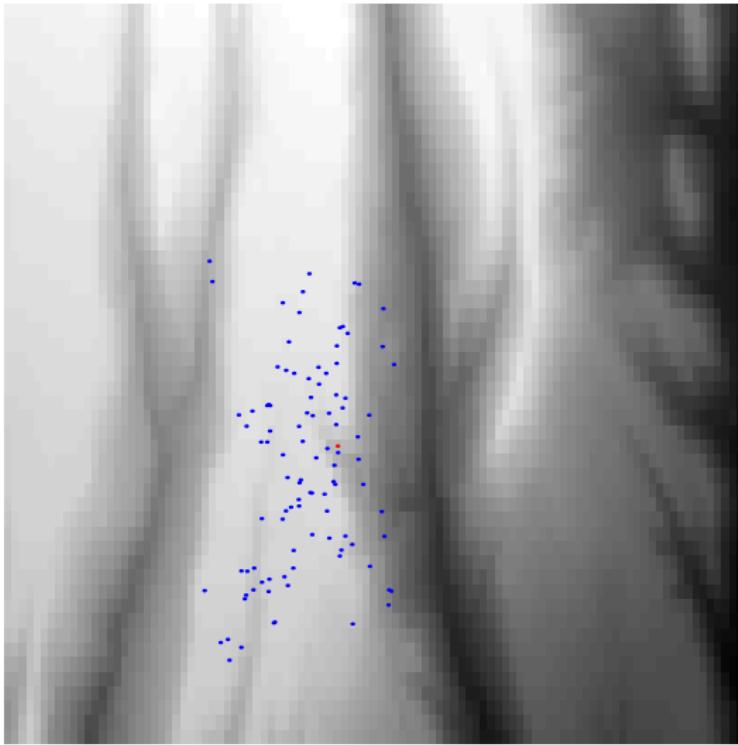


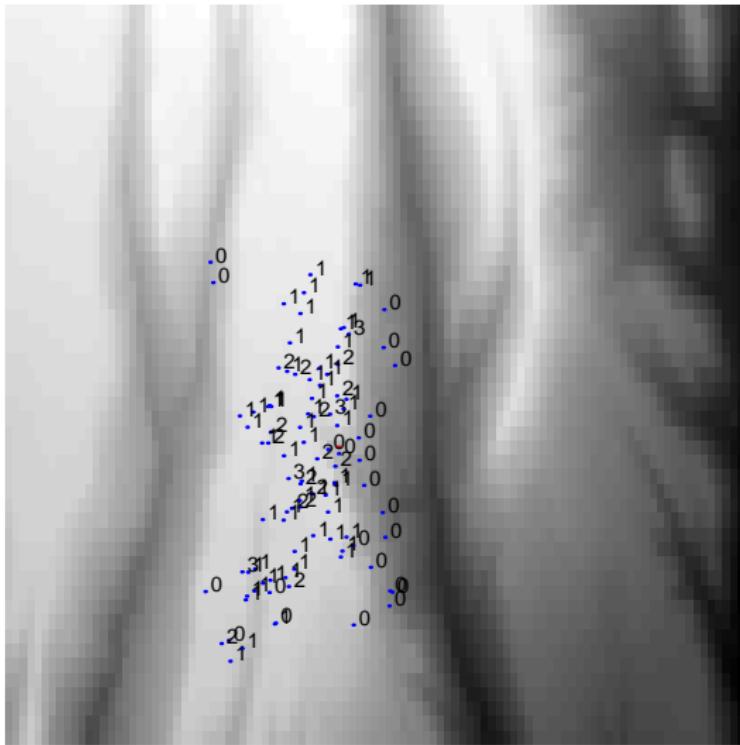


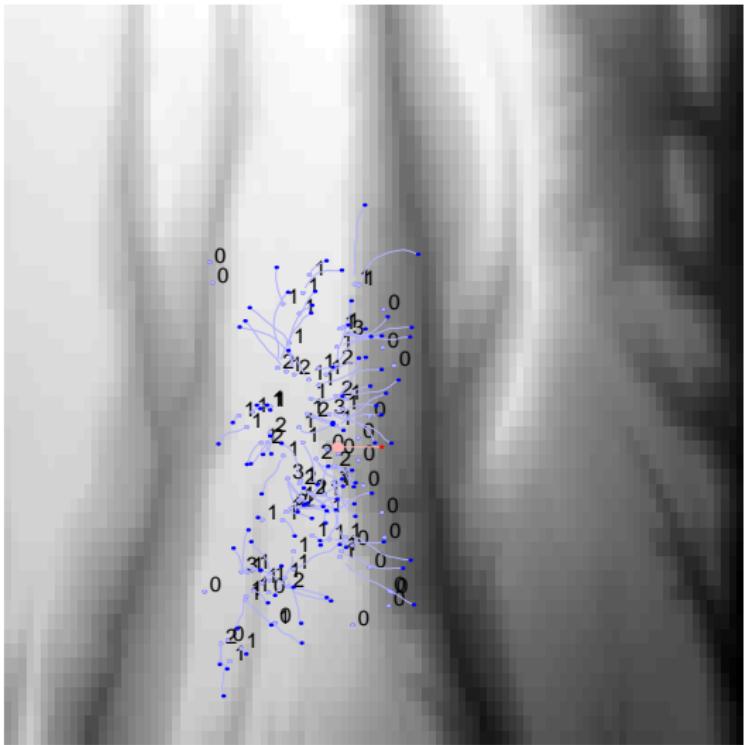


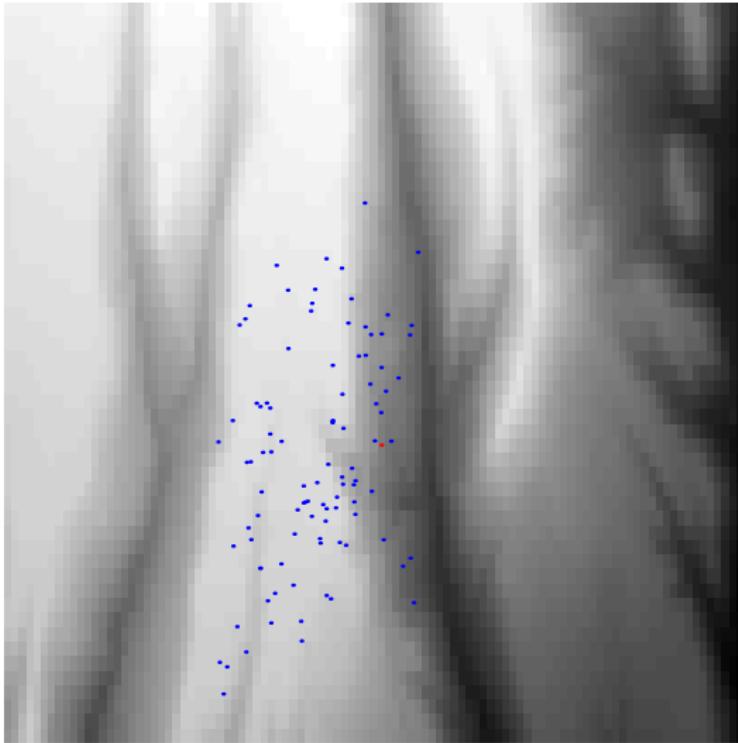


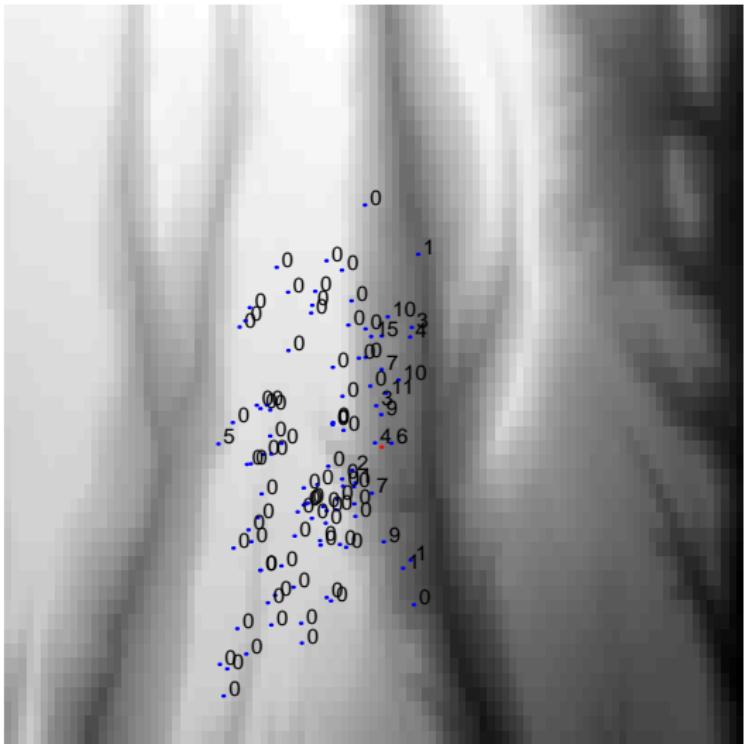


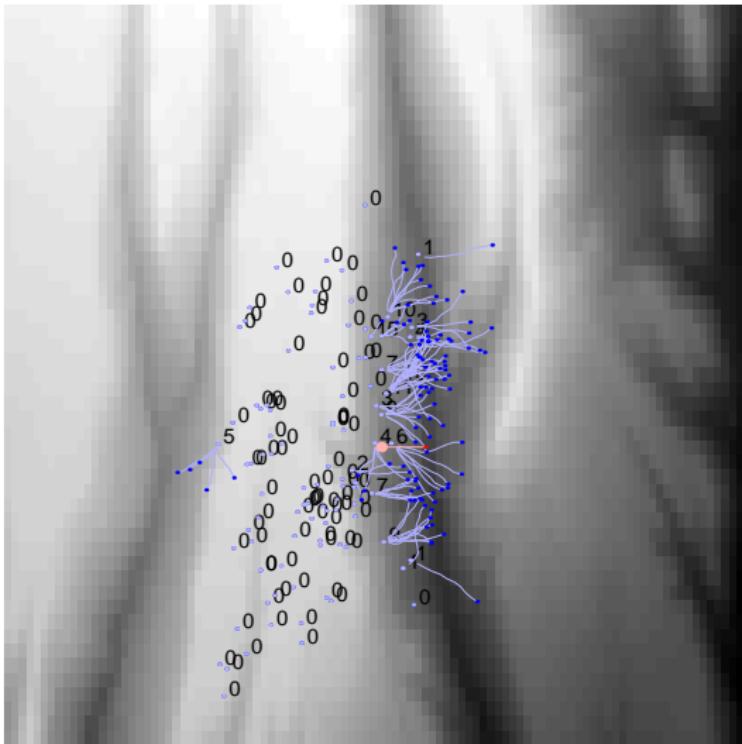


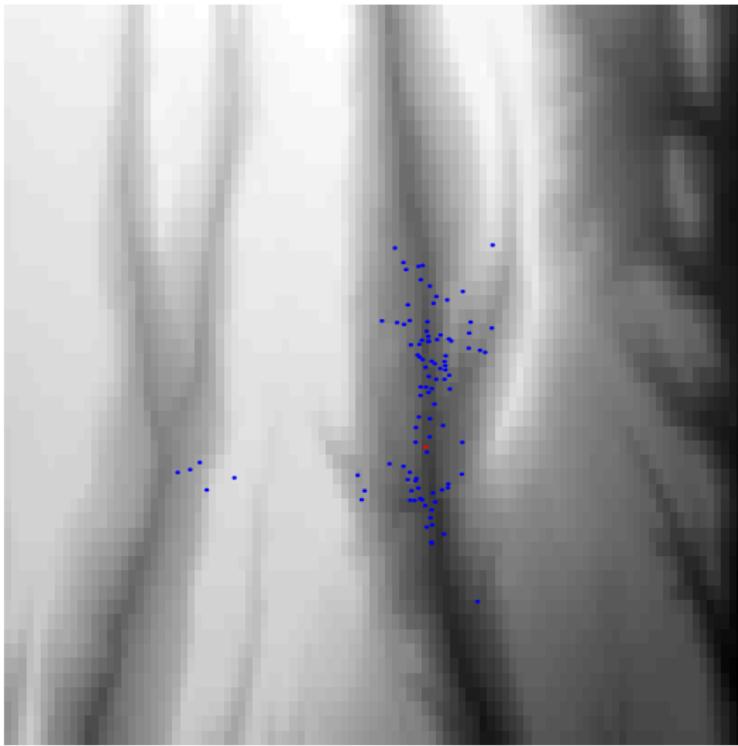


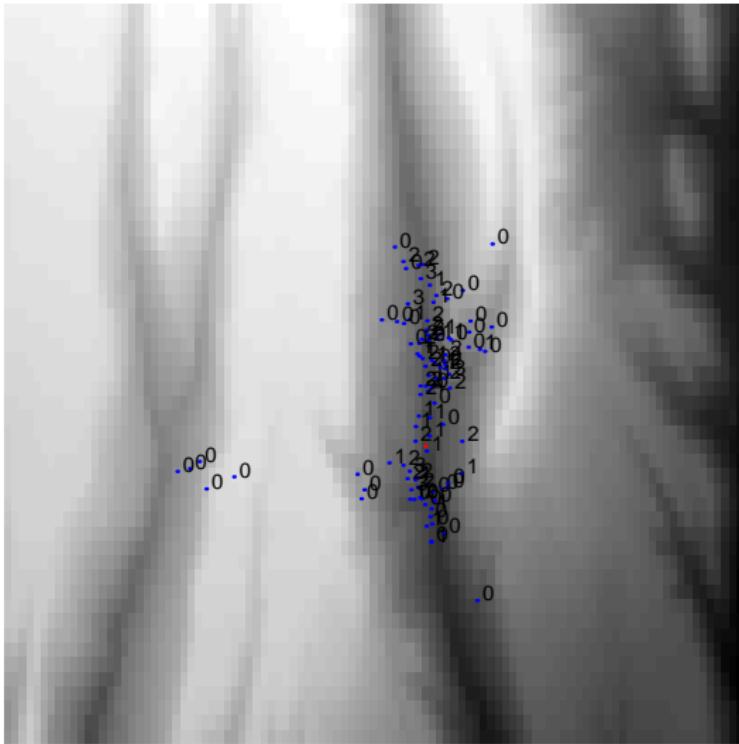


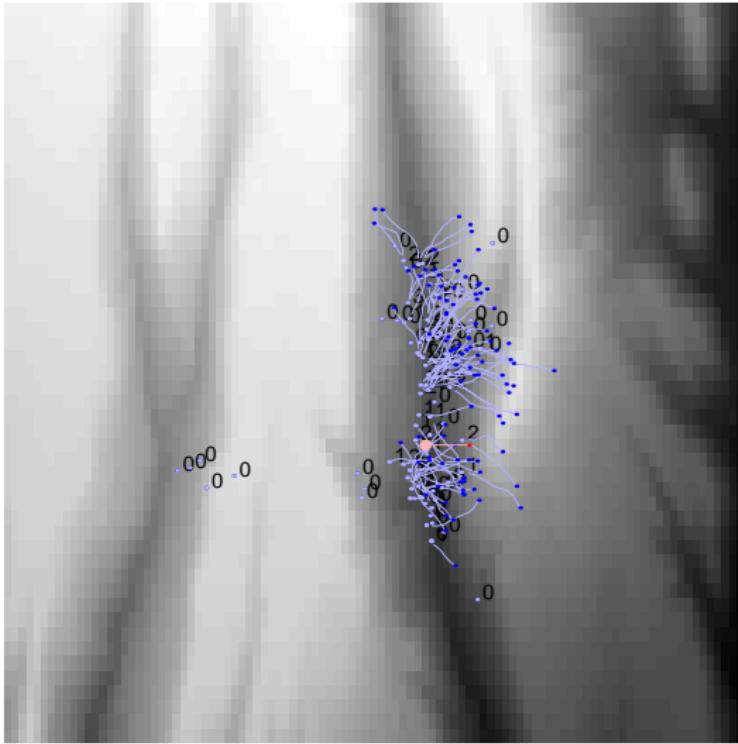


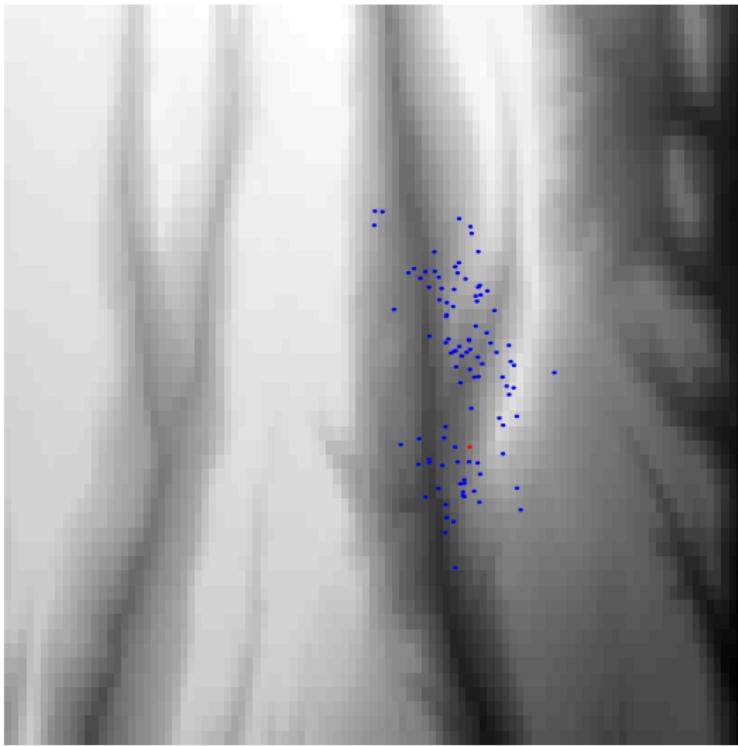


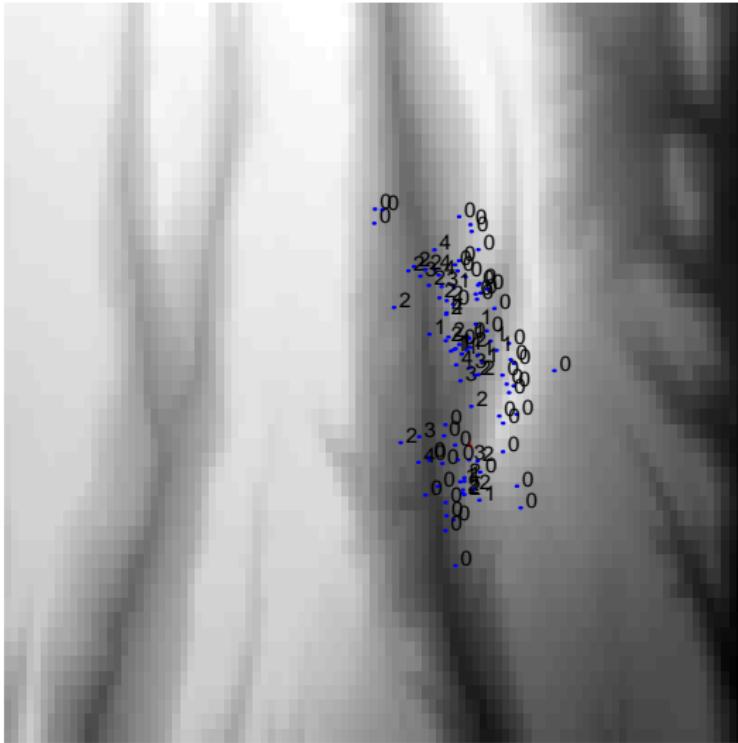


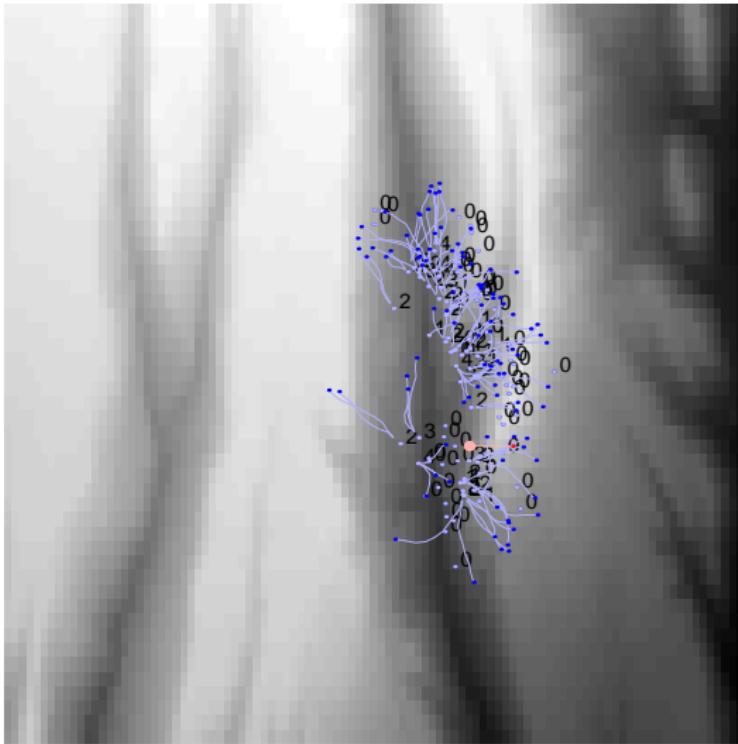


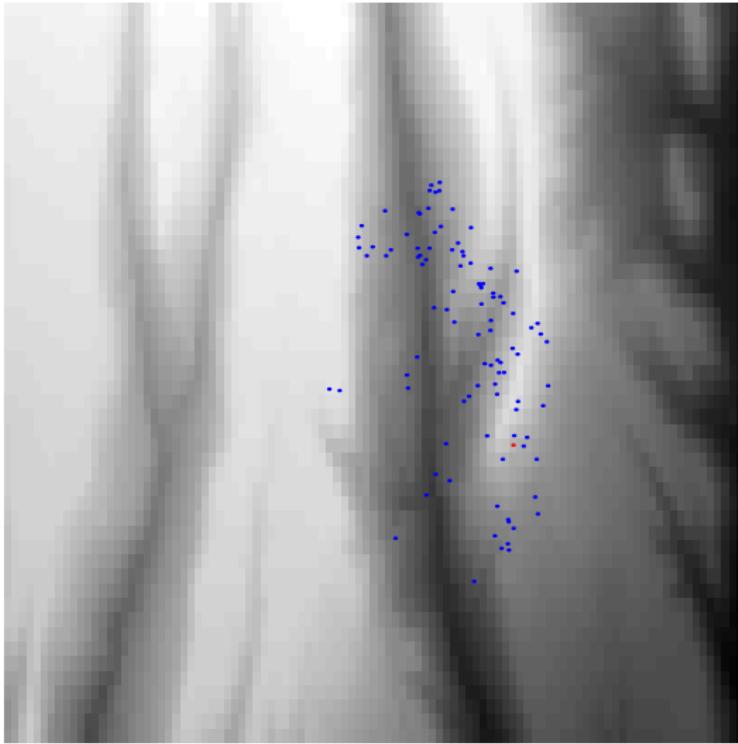


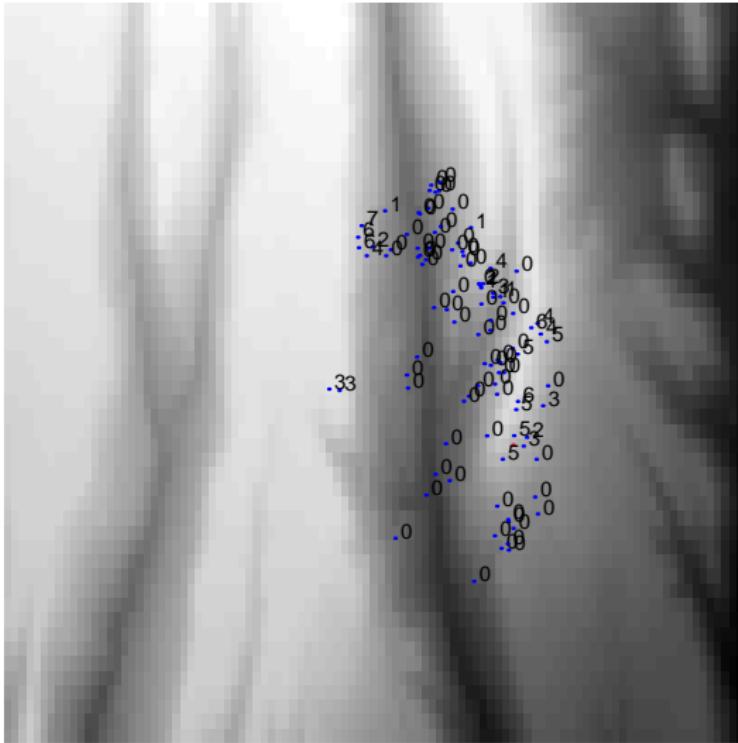


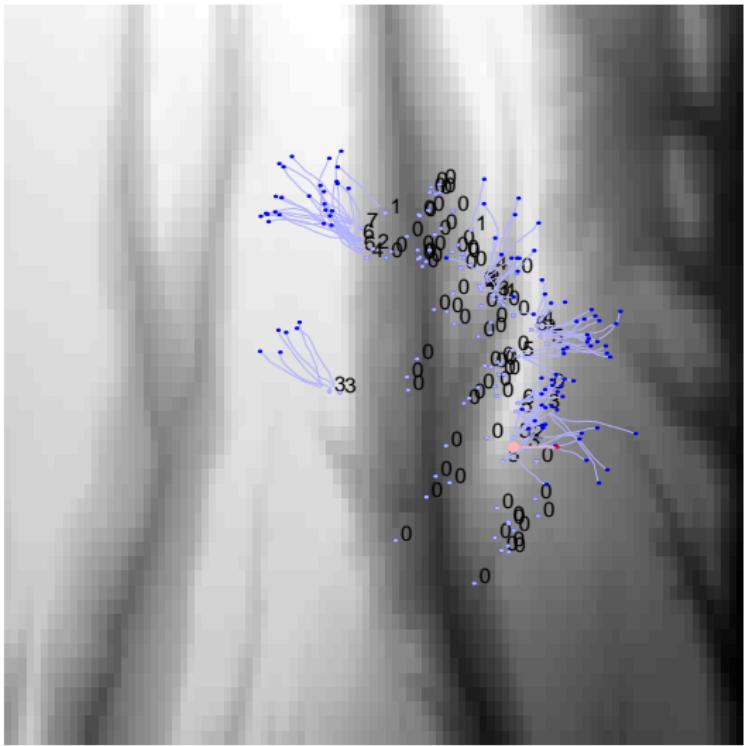


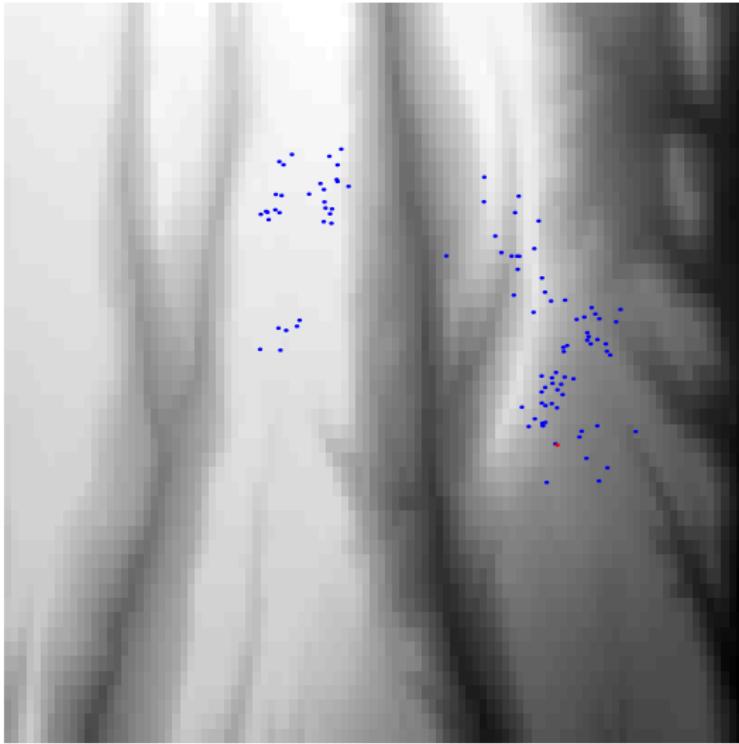


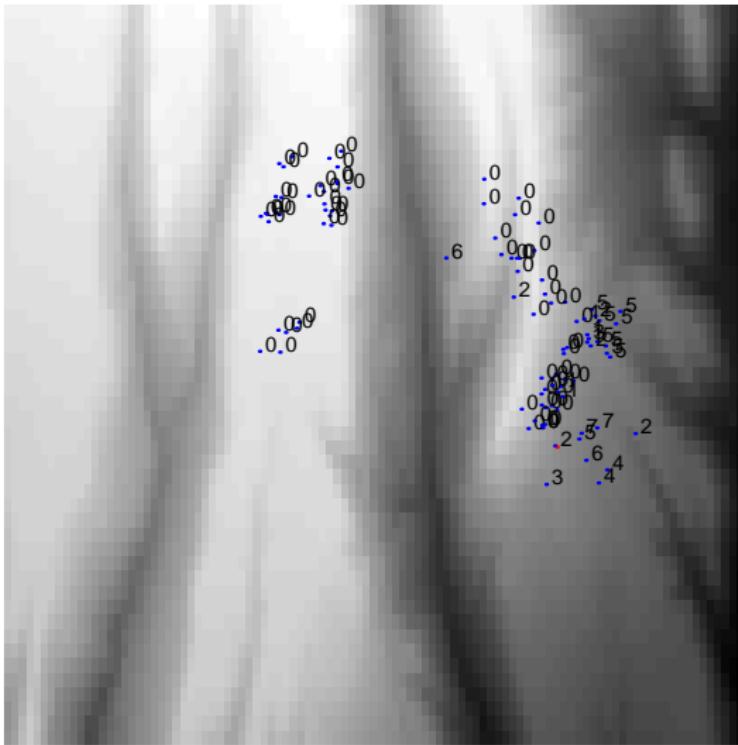


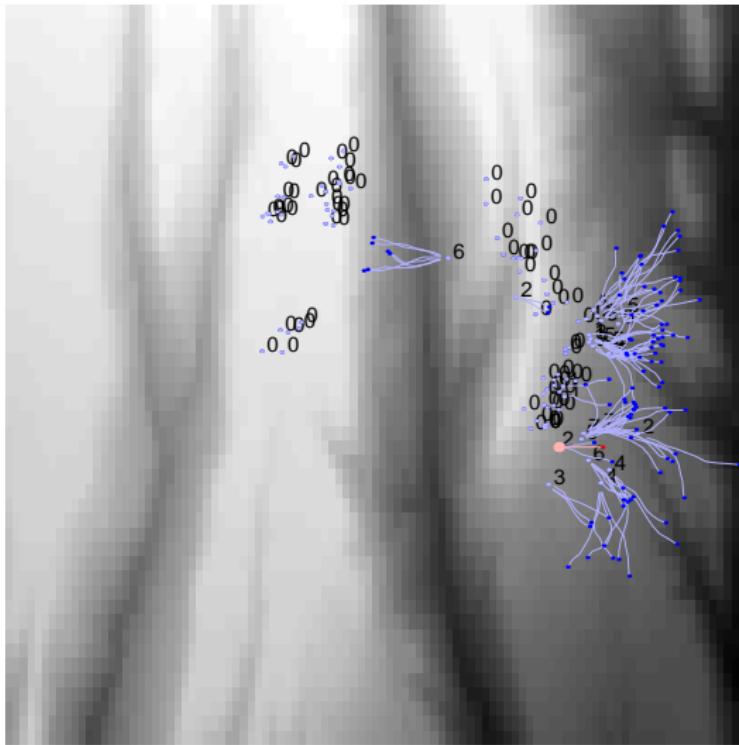


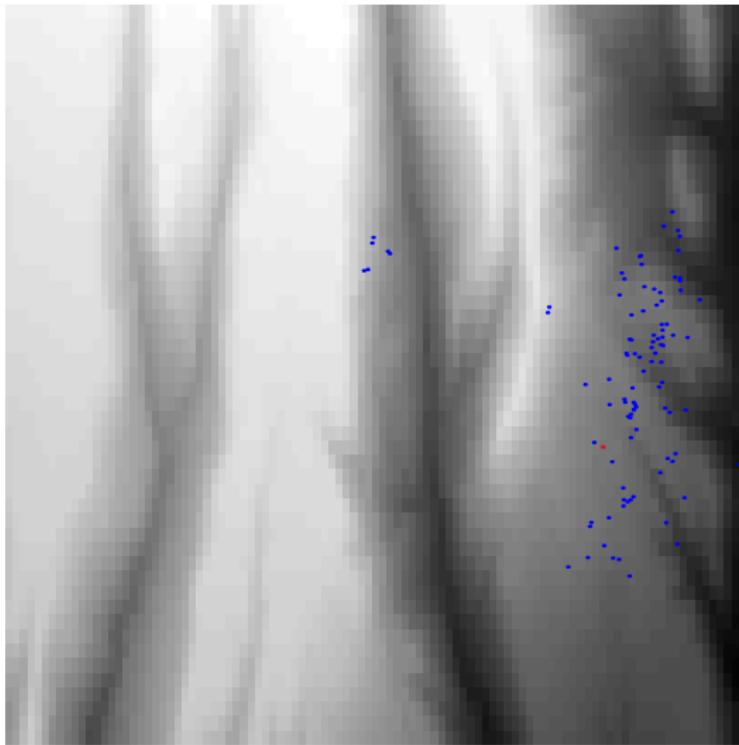


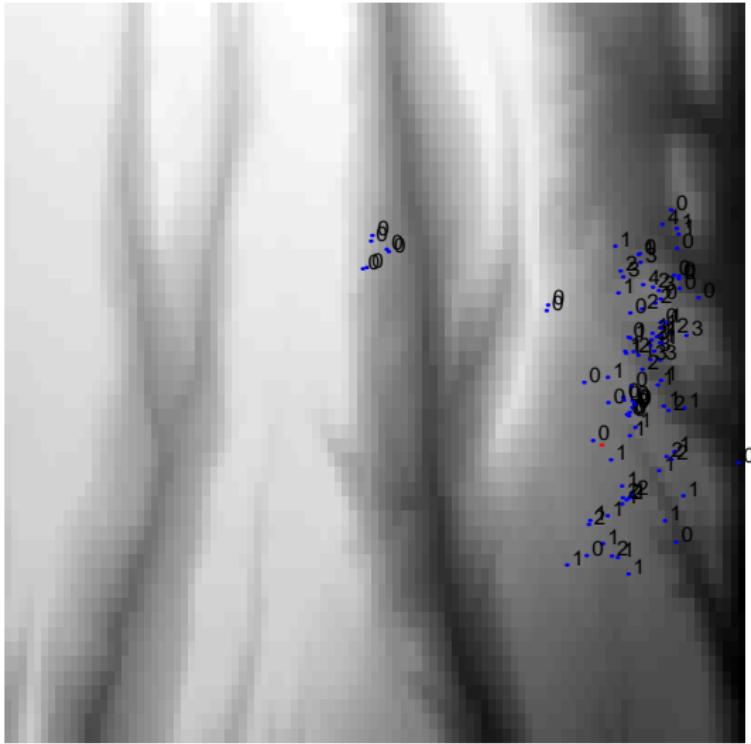


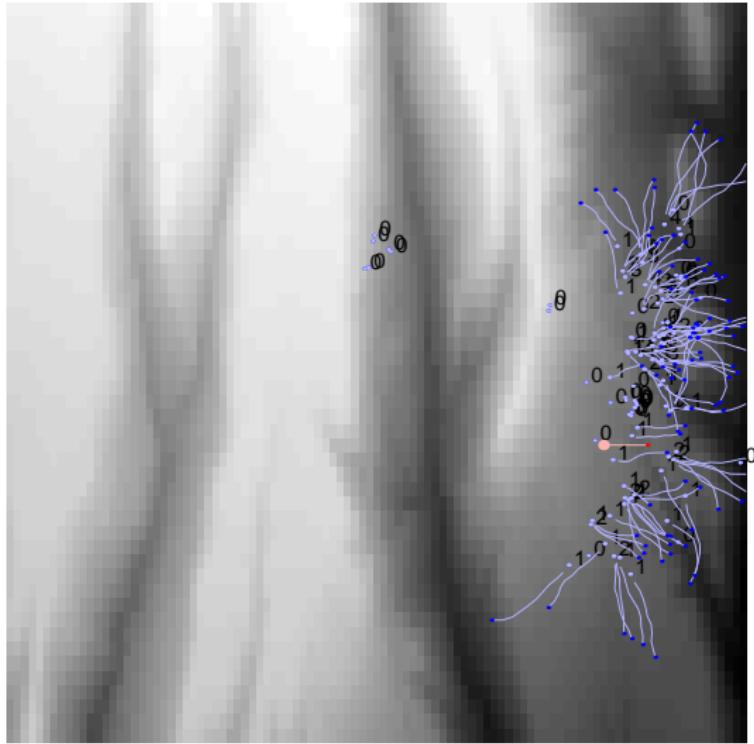












Roadmap

10 Stochastic Volatility Model

11 Terrain Navigation

- The Navigation Task
- The Model
- Illustrations

12 Tracking of a Region in a Video Sequence

- Aims and Computation of Visual Features
- (Non-Linear) State-Space Modelling

13 Robot Localization

The Task

The goal is to track a user-defined arbitrary shaped region in a video sequence.



We illustrate here a simple method based on the sole **color and luminosity** information.

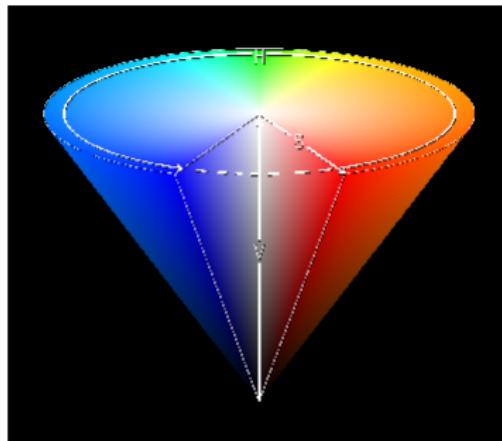
- + Does not rely on a shape model and can be used for arbitrary shaped regions or objects
- + Is very fast (can be run in real time on simple hardware)

We illustrate here a simple method based on the sole **color and luminosity** information.

- + Does not rely on a shape model and can be used for arbitrary shaped regions or objects
- + Is very fast (can be run in real time on simple hardware)
- May generate tracking error in the case of occlusion or coincidence with a similarly colored background

Choice of the Color Representation

Empirically, the so-called HSV (Hue, Saturation, Value) representation of the image provides more robust features for computer vision applications than RGB (in the presence, for instance, of illumination variations)



HSV Cone

Choice of the Visual Features

The visual features consist of histograms computed on the area of interest:

- For H and S, we use a joint histogram with $N_h \times N_s$ bins.
- For V, we use a separate histogram with N_v bins.

These histograms are stacked together in a vector of dimension $L = N_h N_s + N_v$, where typically $N_h = N_s = N_v = 10$, and hence $L = 110$.

Choice of the Visual Features

The visual features consist of histograms computed on the area of interest:

- For H and S, we use a joint histogram with $N_h \times N_s$ bins.
- For V, we use a separate histogram with N_v bins.

These histograms are stacked together in a vector of dimension $L = N_h N_s + N_v$, where typically $N_h = N_s = N_v = 10$, and hence $L = 110$.

To allow for the comparison of regions of different scales the feature vector is normalized (ie. its coordinates sum to 1 rather than twice the number of pixels in the region of interest).

Choice of the Visual Features

The visual features consist of histograms computed on the area of interest:

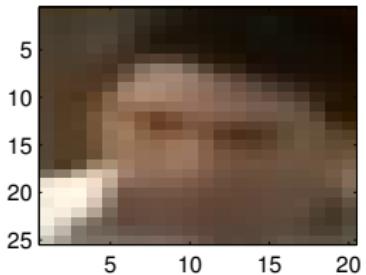
- For H and S, we use a joint histogram with $N_h \times N_s$ bins.
- For V, we use a separate histogram with N_v bins.

These histograms are stacked together in a vector of dimension $L = N_h N_s + N_v$, where typically $N_h = N_s = N_v = 10$, and hence $L = 110$.

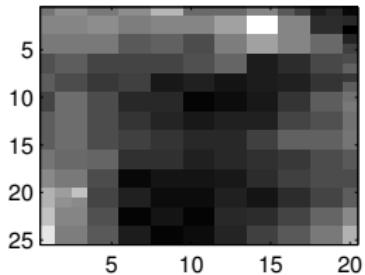
To allow for the comparison of regions of different scales the feature vector is normalized (ie. its coordinates sum to 1 rather than twice the number of pixels in the region of interest).

By comparison, if the area of interest is of size 60×10 the original number of HSV value is 1,800.

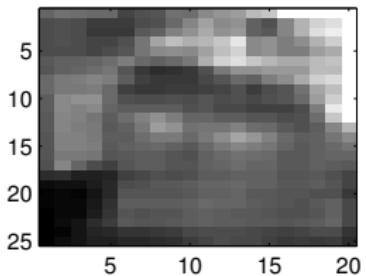
Imagette



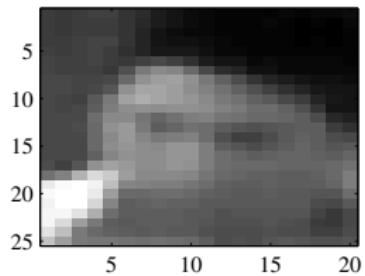
Teinte H



Saturation S



Intensité V



Area and histogram values in the 3 HSV projections.

Conditional Likelihood of the Observation

Let x_k denote a possible position of the region of interest (in pixel coordinates) in the image Y_k observed at time k .

- The **conditional likelihood** is chosen as a decreasing function of the difference between the reference vector of features b^{ref} and the vector of features $b(x_k, Y_k)$ computed on the image Y_k assuming that the area of interest is located at position x_k .

Conditional Likelihood of the Observation

Let x_k denote a possible position of the region of interest (in pixel coordinates) in the image Y_k observed at time k .

- The **conditional likelihood** is chosen as a decreasing function of the difference between the reference vector of features b^{ref} and the vector of features $b(x_k, Y_k)$ computed on the image Y_k assuming that the area of interest is located at position x_k .

Typically,

$$p(Y_k|x_k) \propto \exp(-\lambda d^2)$$

where (for instance)

$$d^2 = \frac{1}{2} \sum_{\ell=1}^L \left(\sqrt{b_\ell(Y_k, x_k)} - \sqrt{b_\ell^{\text{ref}}} \right)^2$$

is the so-called (squared) Hellinger distance.

Conditional Likelihood of the Observation

Let x_k denote a possible position of the region of interest (in pixel coordinates) in the image Y_k observed at time k .

- The **conditional likelihood** is chosen as a decreasing function of the difference between the reference vector of features b^{ref} and the vector of features $b(x_k, Y_k)$ computed on the image Y_k assuming that the area of interest is located at position x_k .

Typically,

$$p(Y_k|x_k) \propto \exp(-\lambda d^2)$$

where (for instance)

$$d^2 = \frac{1}{2} \sum_{\ell=1}^L \left(\sqrt{b_\ell(Y_k, x_k)} - \sqrt{b_\ell^{\text{ref}}} \right)^2$$

is the so-called (squared) Hellinger distance.

The constant λ is tuned empirically so as to achieve the best tracking performance.

State Modelling

To incorporate sensible prior information on the (relative) displacement of the area of interest, the state is extended to include five coordinates:

$$X_k = \begin{bmatrix} X_{1,k} \\ X_{2,k} \\ X_{3,k} \\ X_{4,k} \\ X_{5,k} \end{bmatrix}$$

where

- $X_{1:2,k}$ represent the coordinates of the center of the region of interest (taken to be rectangular in the simulations)
- $X_{3:5,k}$ are the speed components
- $X_{5,k} \in \mathbb{R}^+$ corresponds to the relative scale factor of the region of interest (where the scale of 1 corresponds to the whole image)

Dynamic Equations

The prior on the system dynamic is chosen to favor constant speed straight line trajectories and quasi-constant scales:

$$\begin{bmatrix} X_{1,k} \\ X_{2,k} \\ X_{3,k} \\ X_{4,k} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \delta & 0 \\ 0 & 1 & 0 & \delta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{1,k-1} \\ X_{2,k-1} \\ X_{3,k-1} \\ X_{4,k-1} \end{bmatrix} + \begin{bmatrix} \frac{\delta^2}{2} & 0 \\ 0 & \frac{\delta^2}{2} \\ \delta & 0 \\ 0 & \delta \end{bmatrix} \begin{bmatrix} \epsilon_{1,k} \\ \epsilon_{2,k} \end{bmatrix}$$

$$X_{5,k} = X_{5,k-1} + \nu_k$$

Demo in MATLAB | demos/progs_HSVtracking

- Illustration of tracking performance and robustness to scale variations and temporary occlusions

Roadmap

10 Stochastic Volatility Model

11 Terrain Navigation

- The Navigation Task
- The Model
- Illustrations

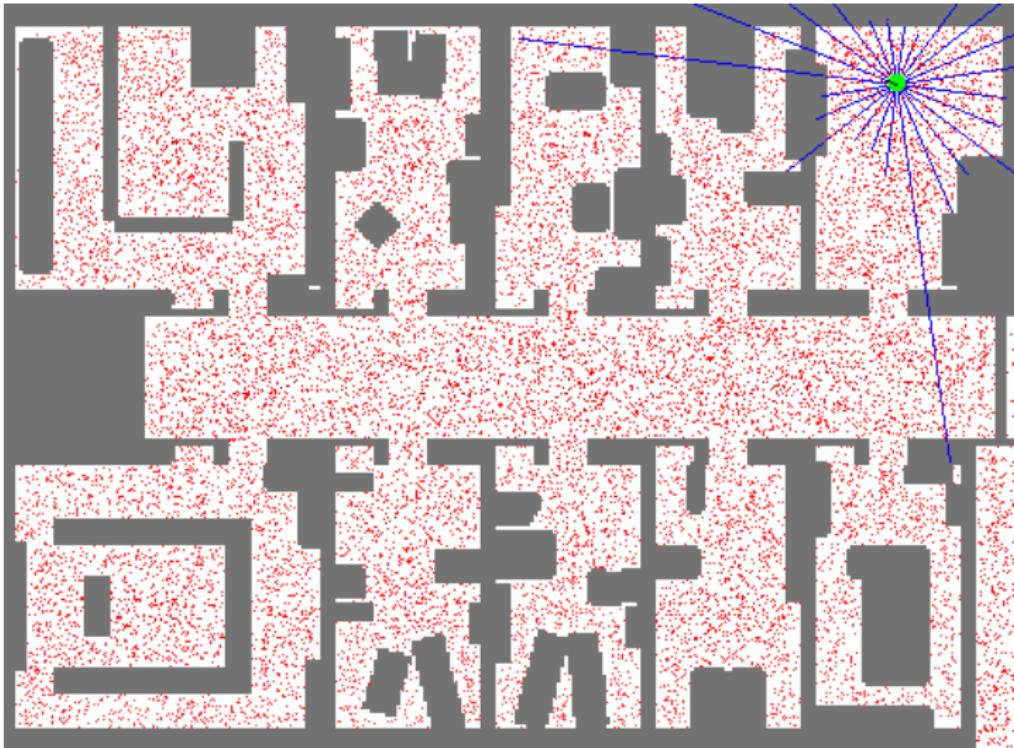
12 Tracking of a Region in a Video Sequence

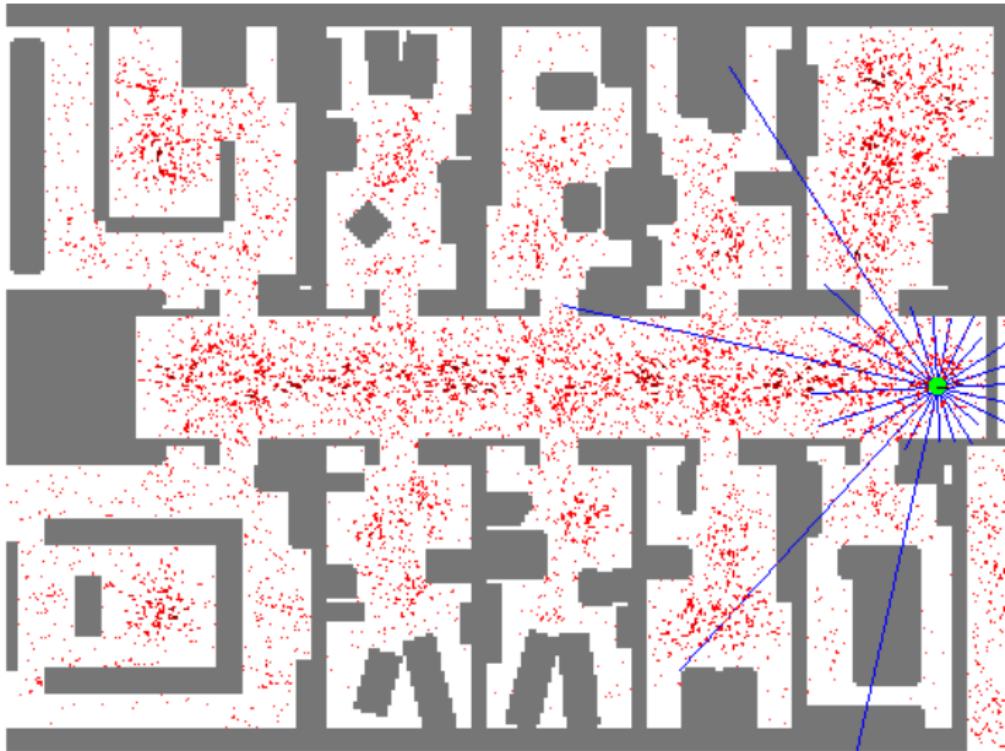
- Aims and Computation of Visual Features
- (Non-Linear) State-Space Modelling

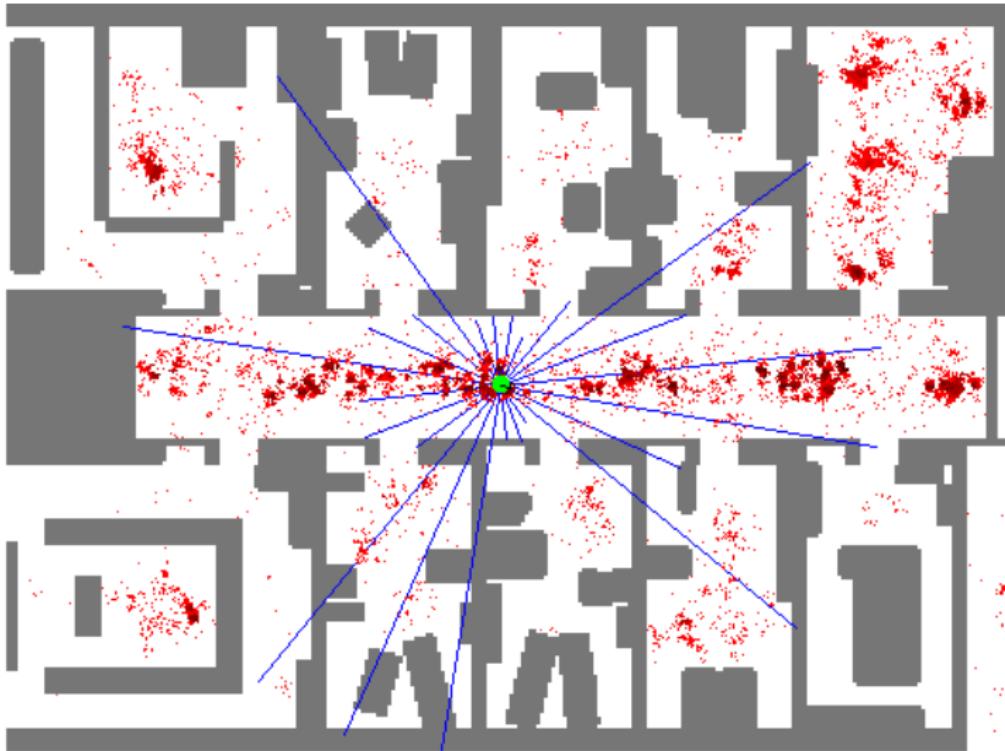
13 Robot Localization

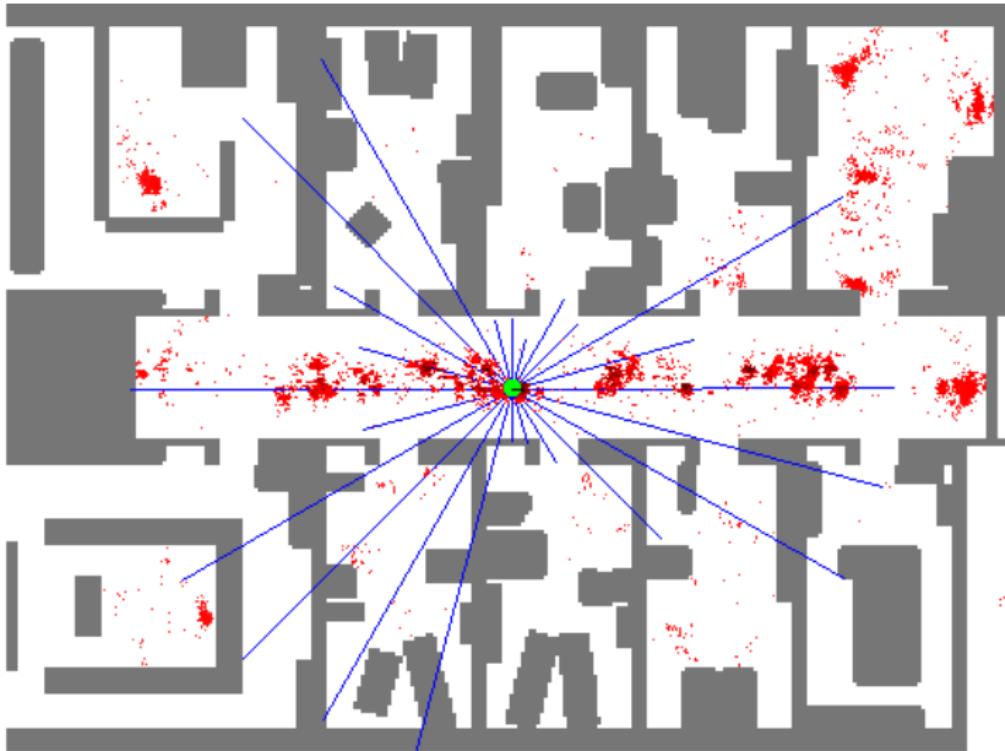
Robot Localization

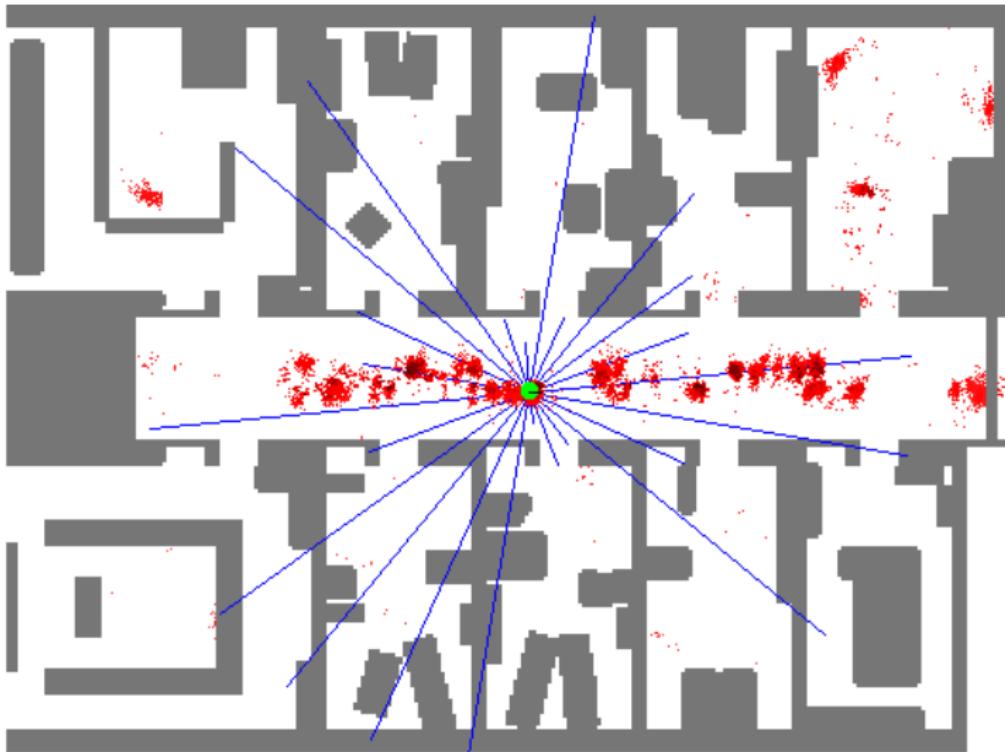
This demo by Sebastian Thrun shows particle localization on a map for a moving robot, using laser range finders.

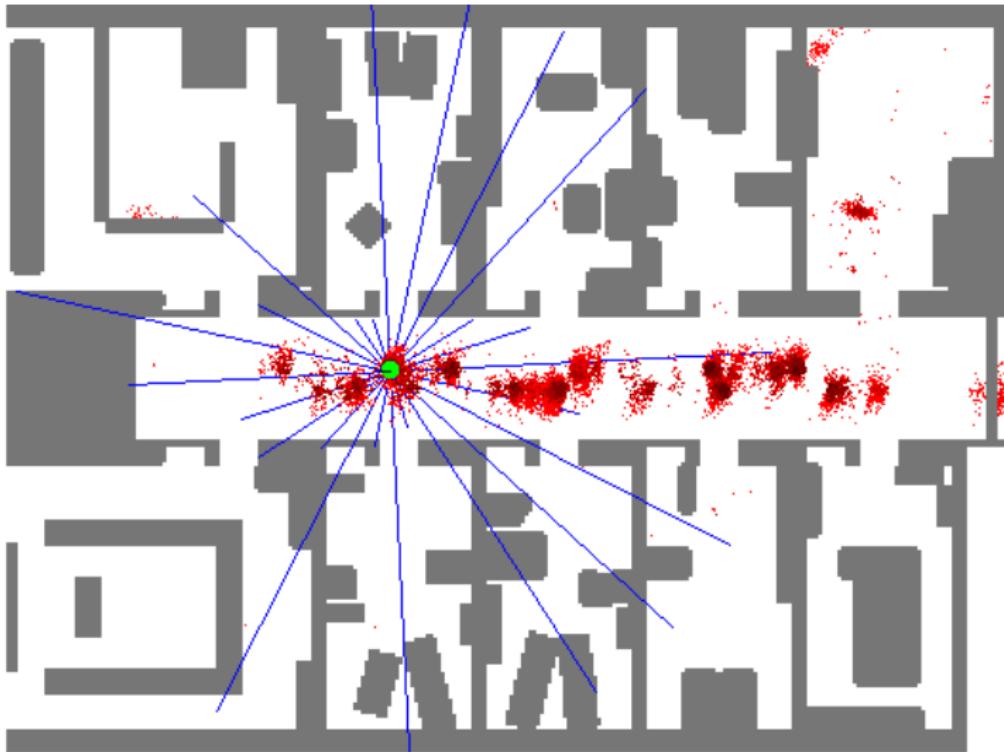


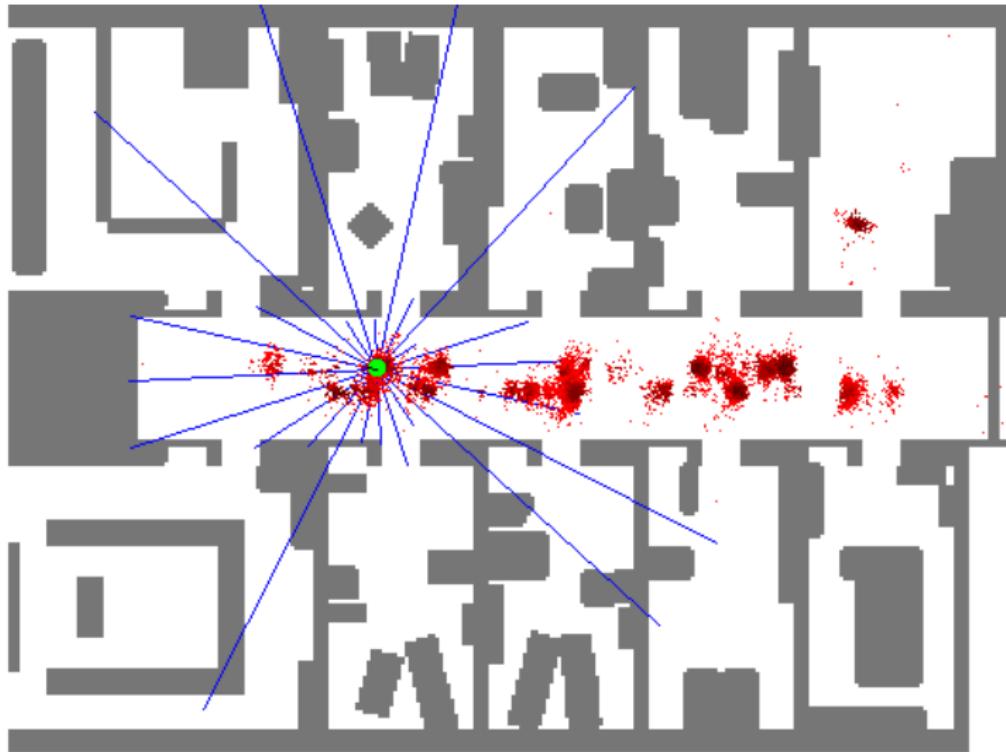


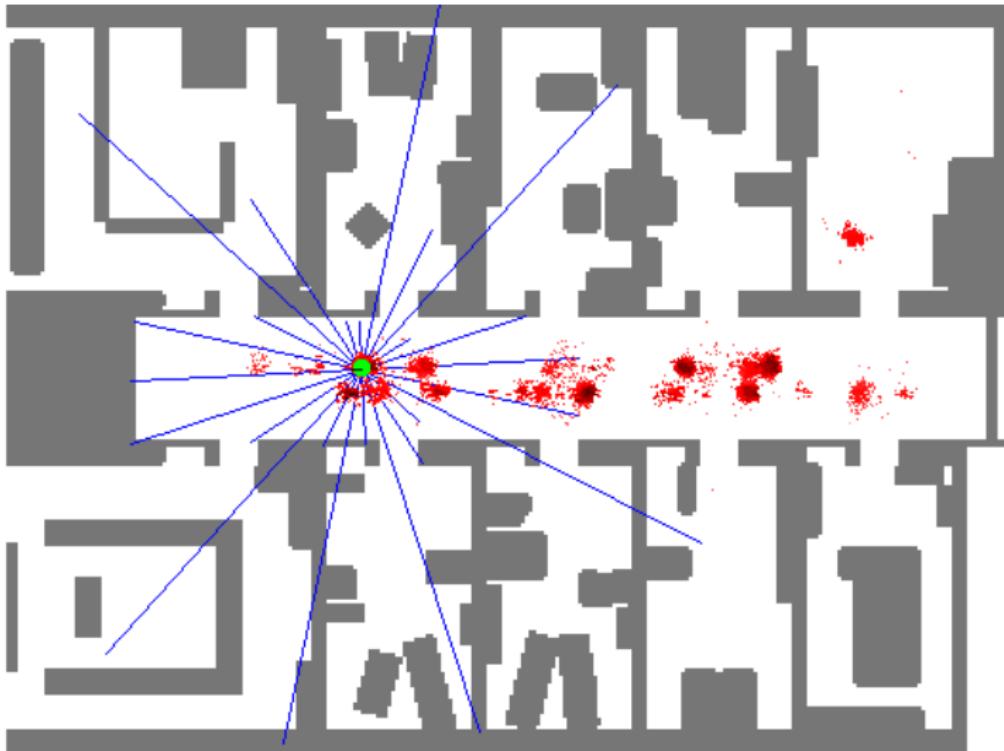


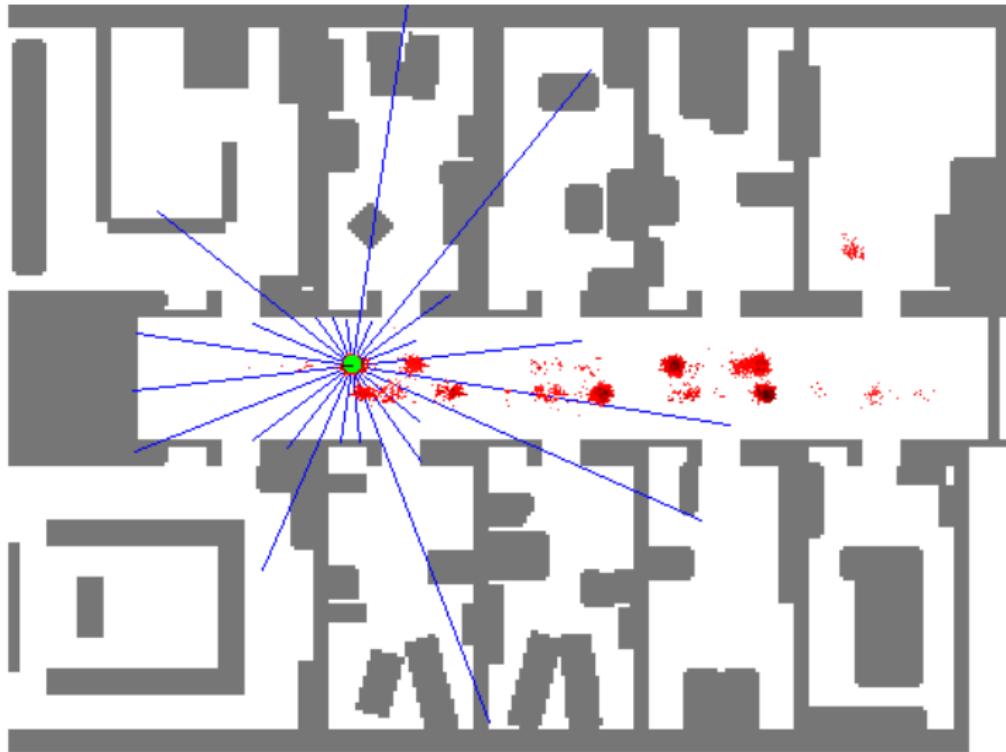


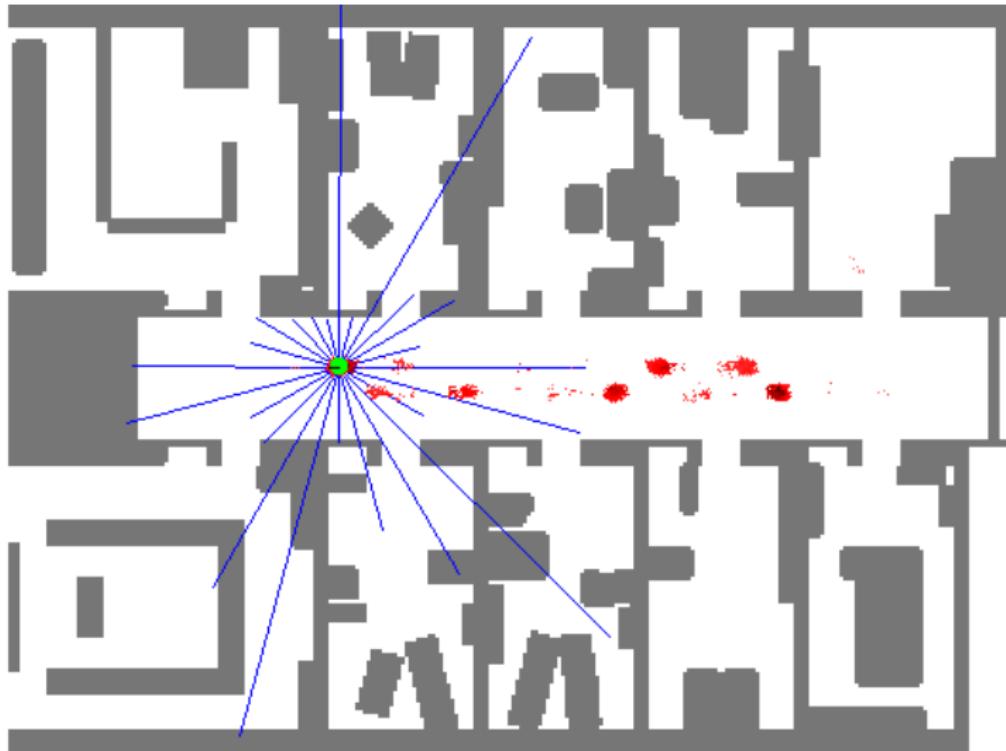


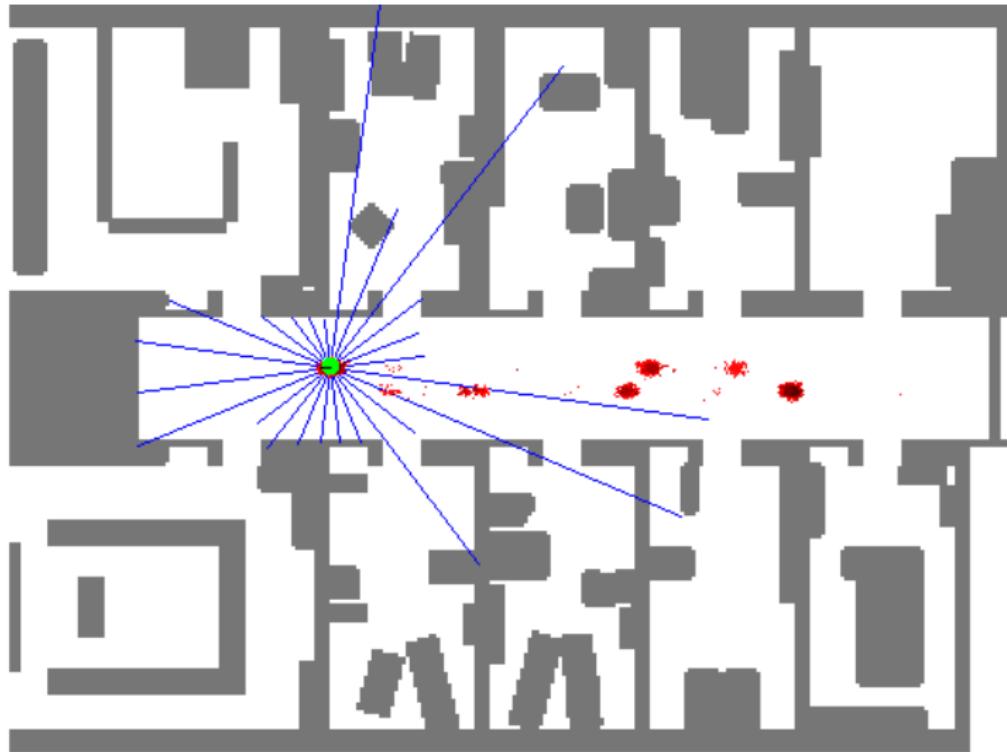


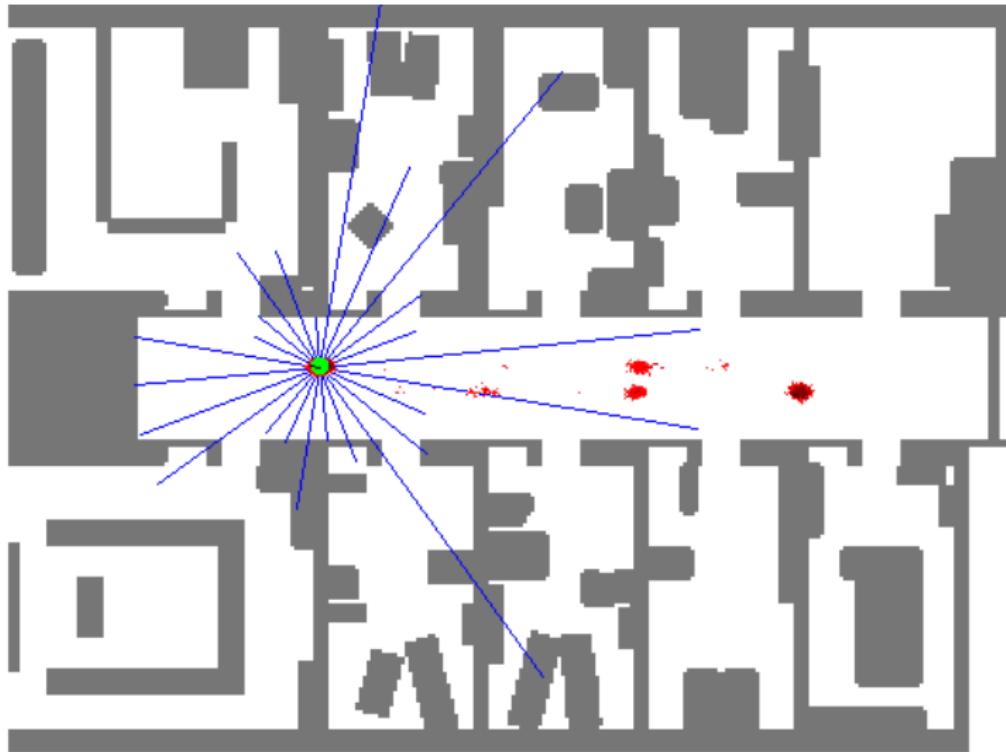


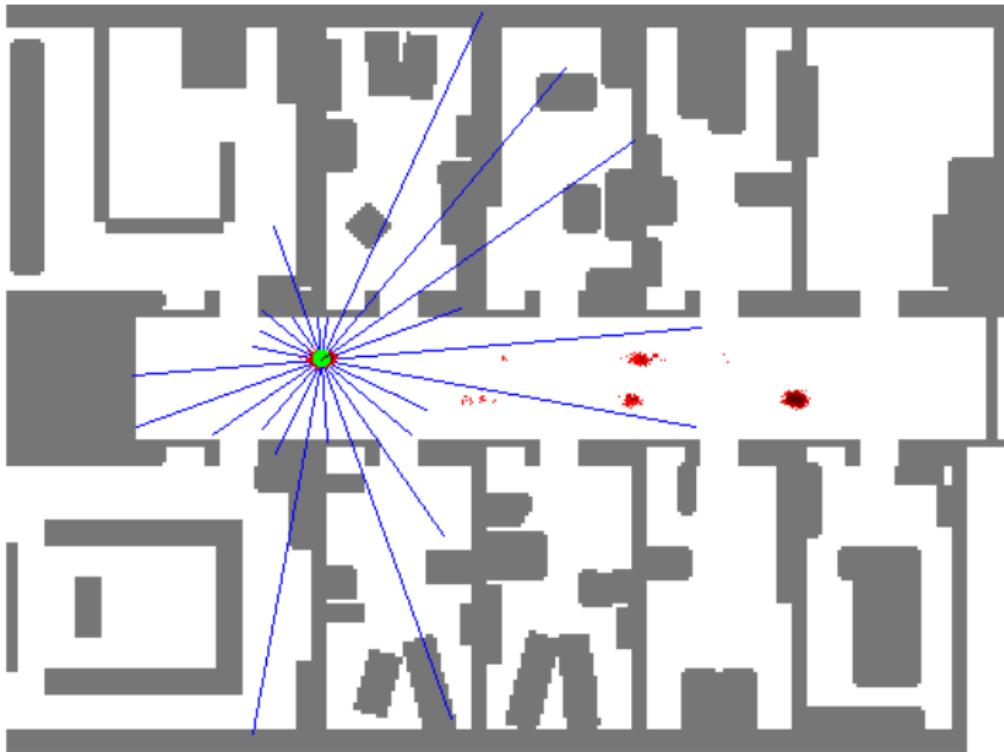


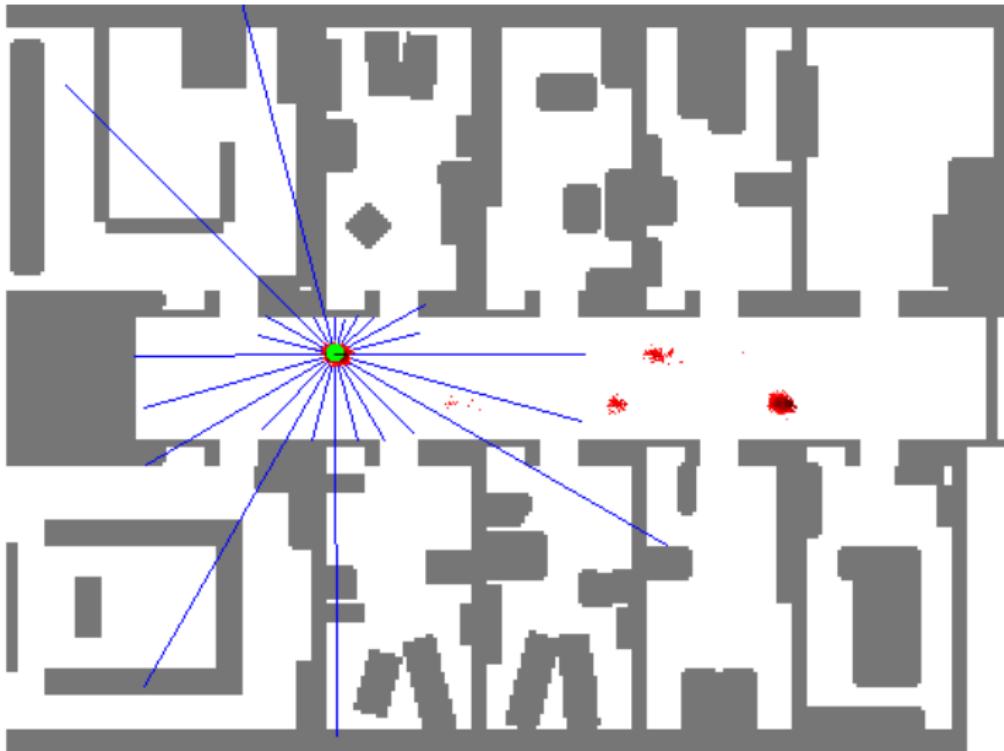


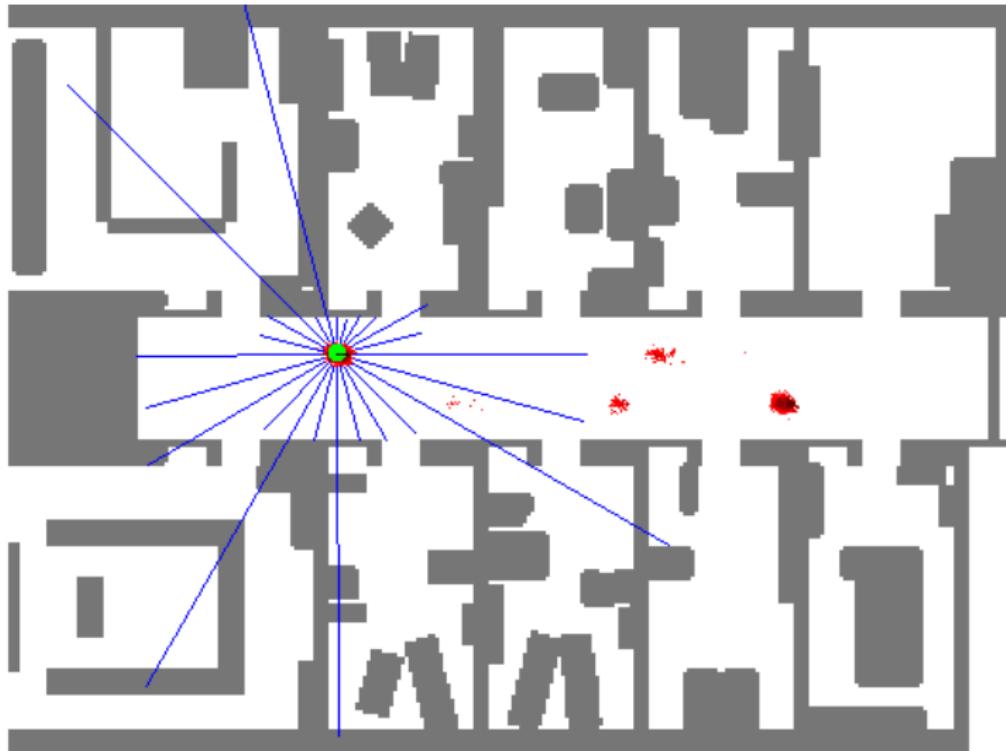


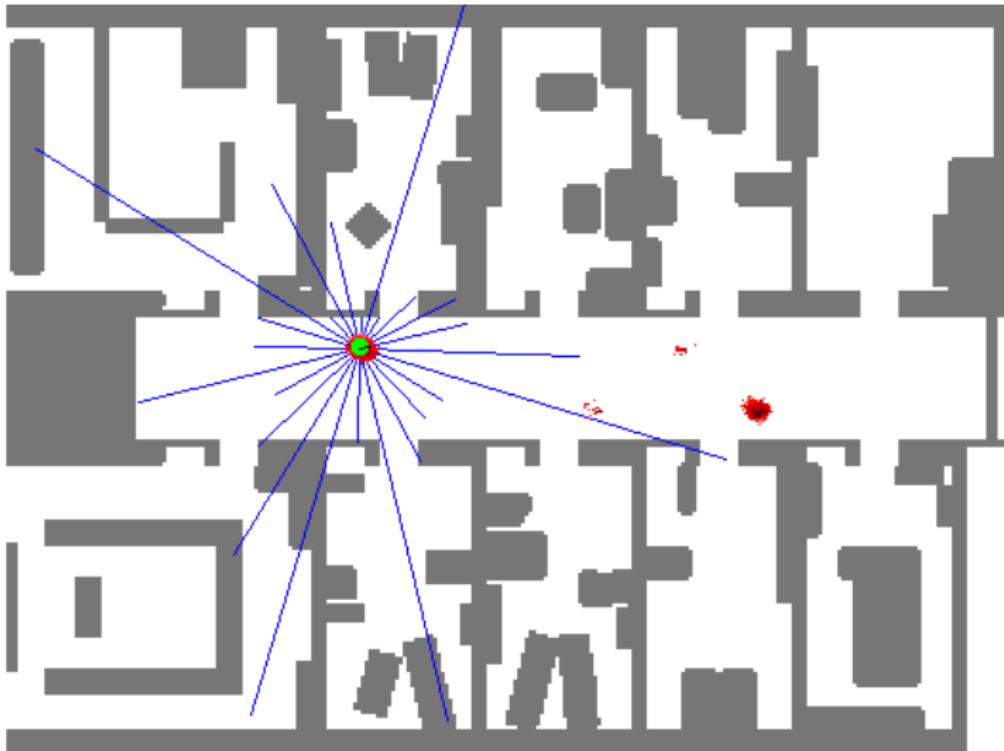


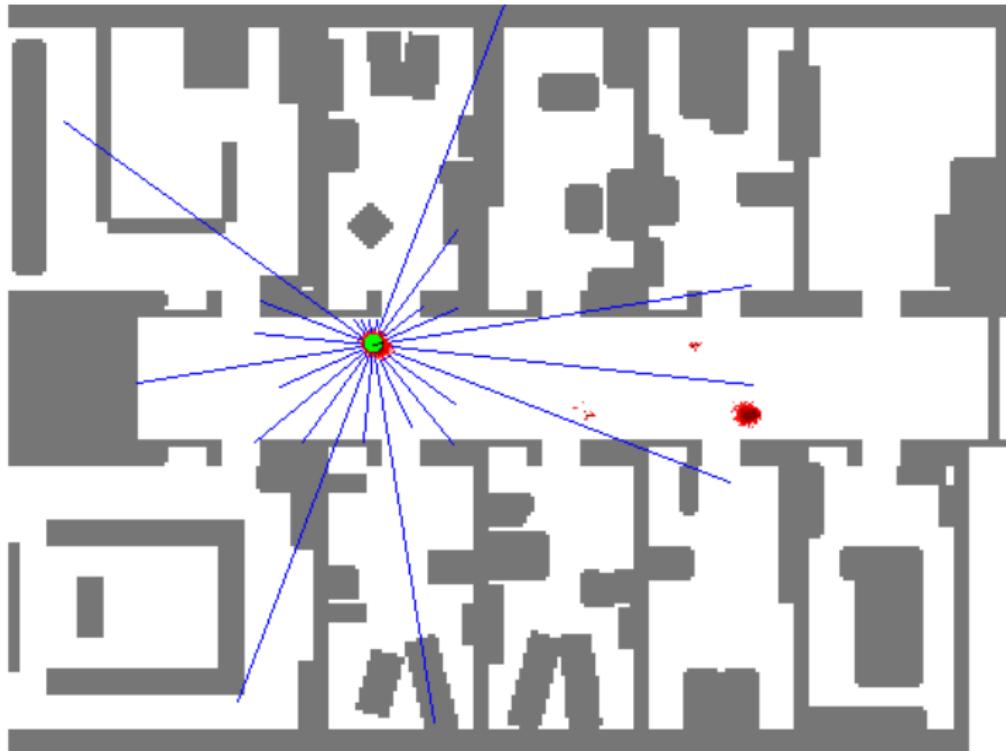


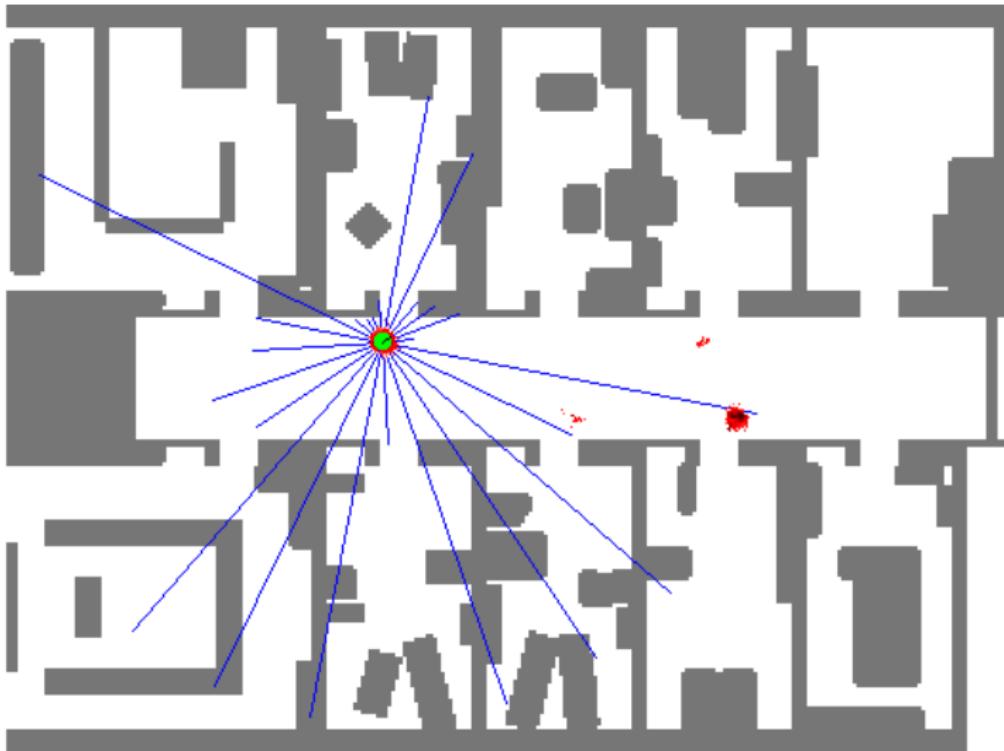


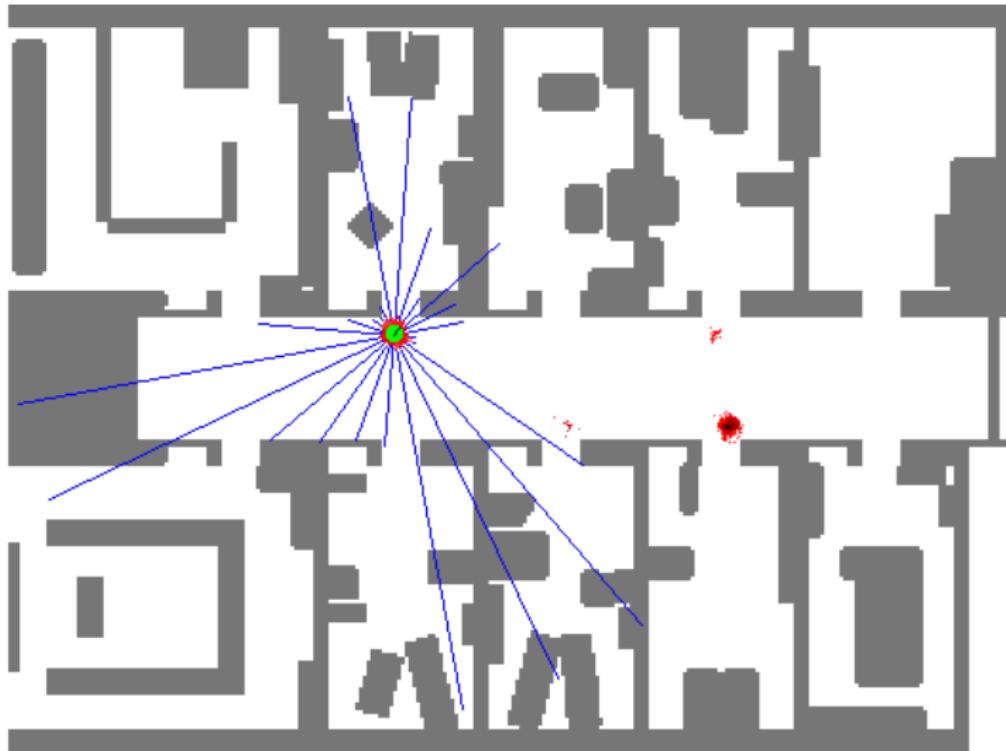


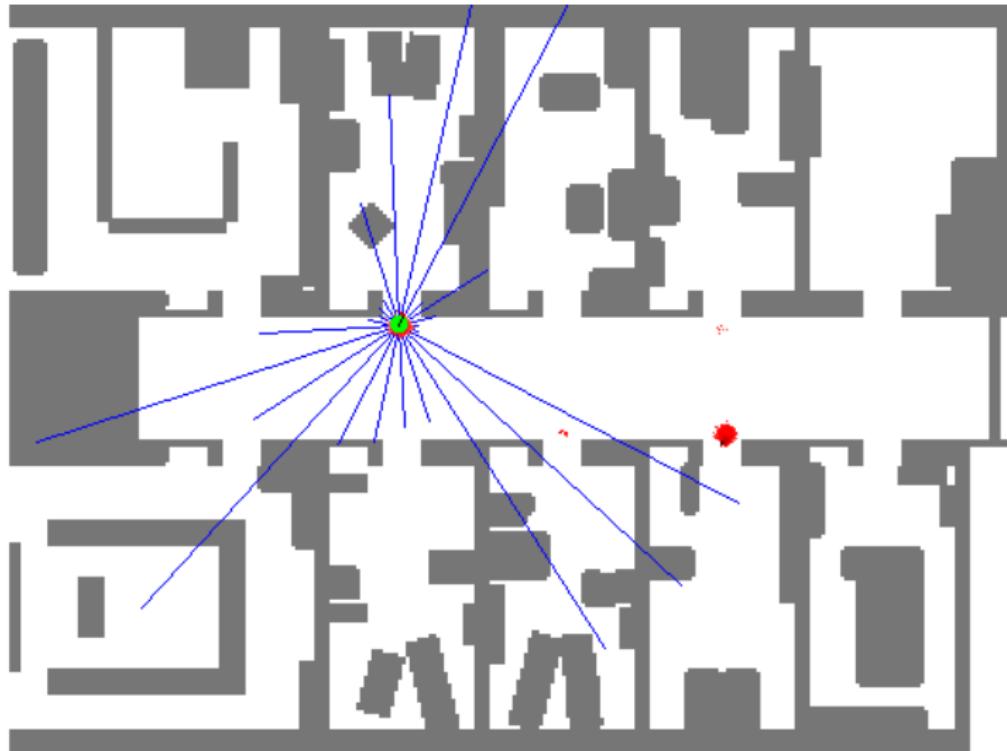


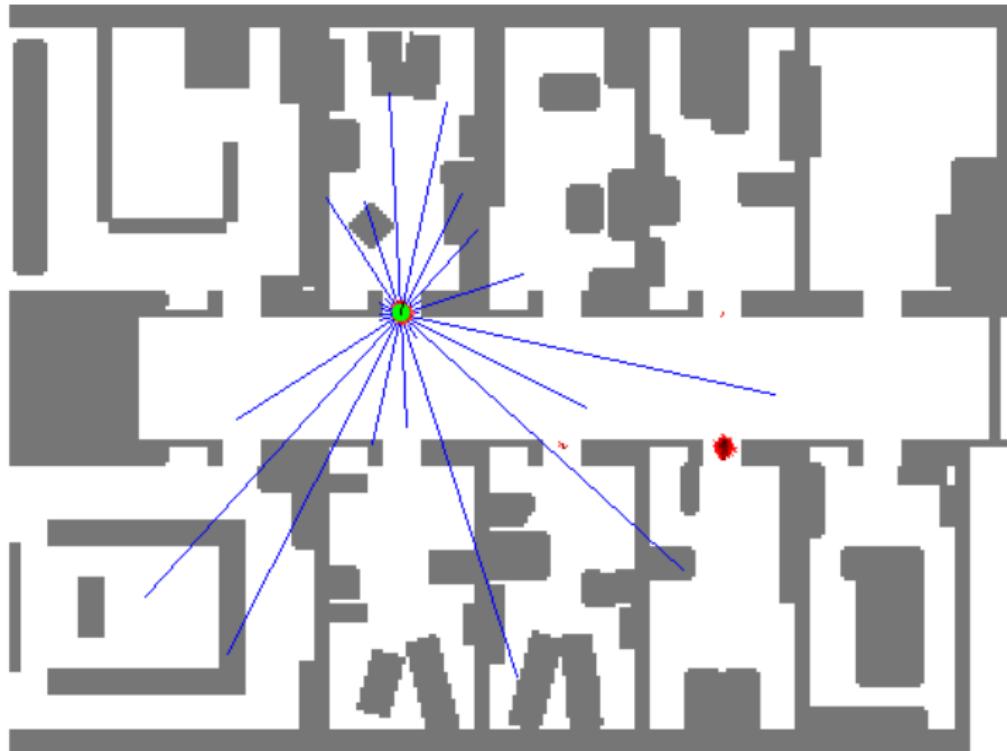


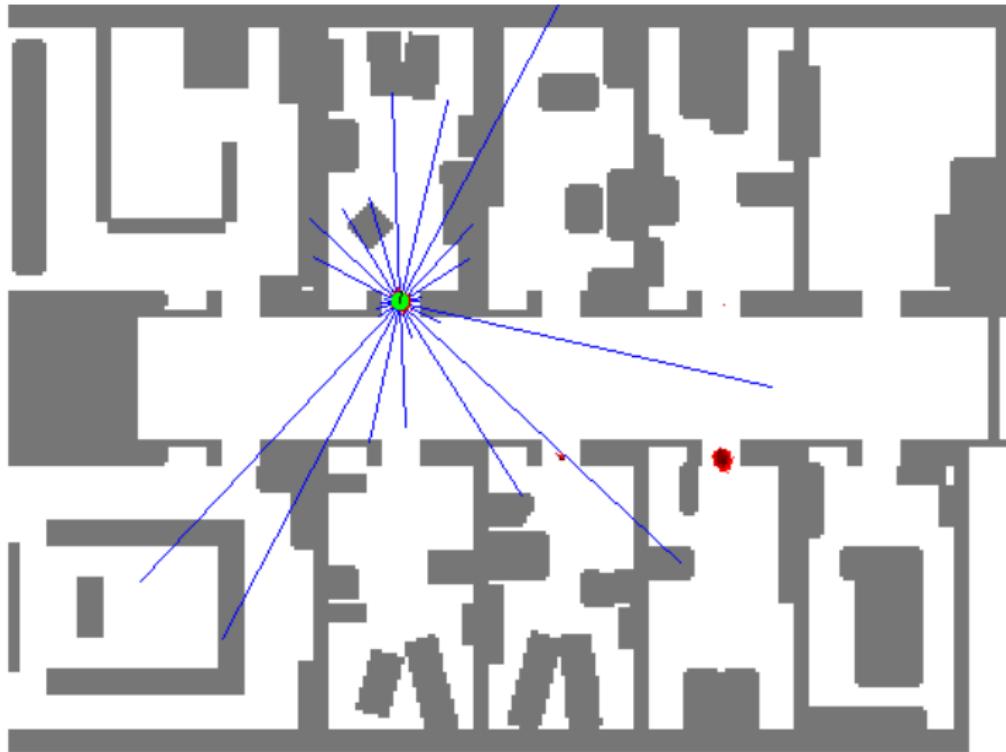


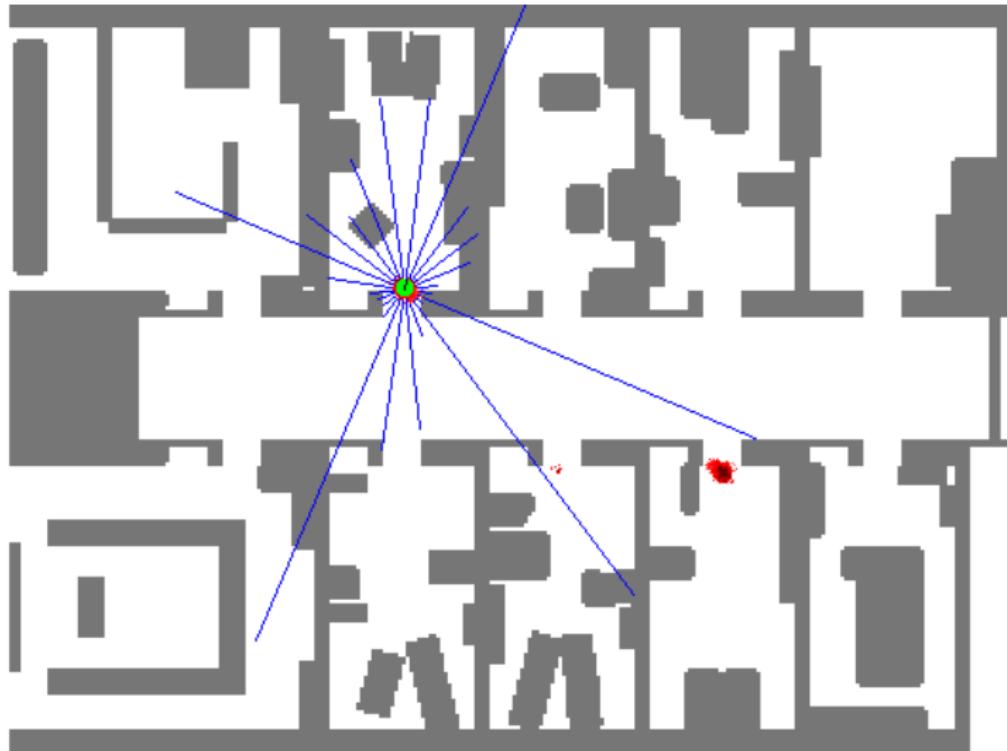


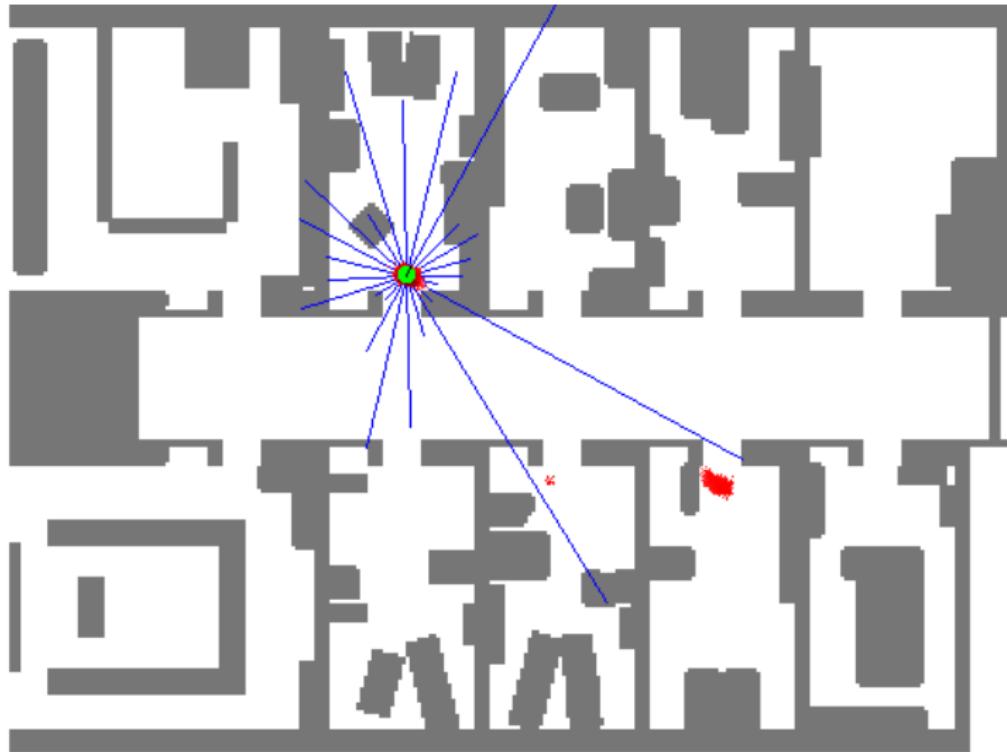


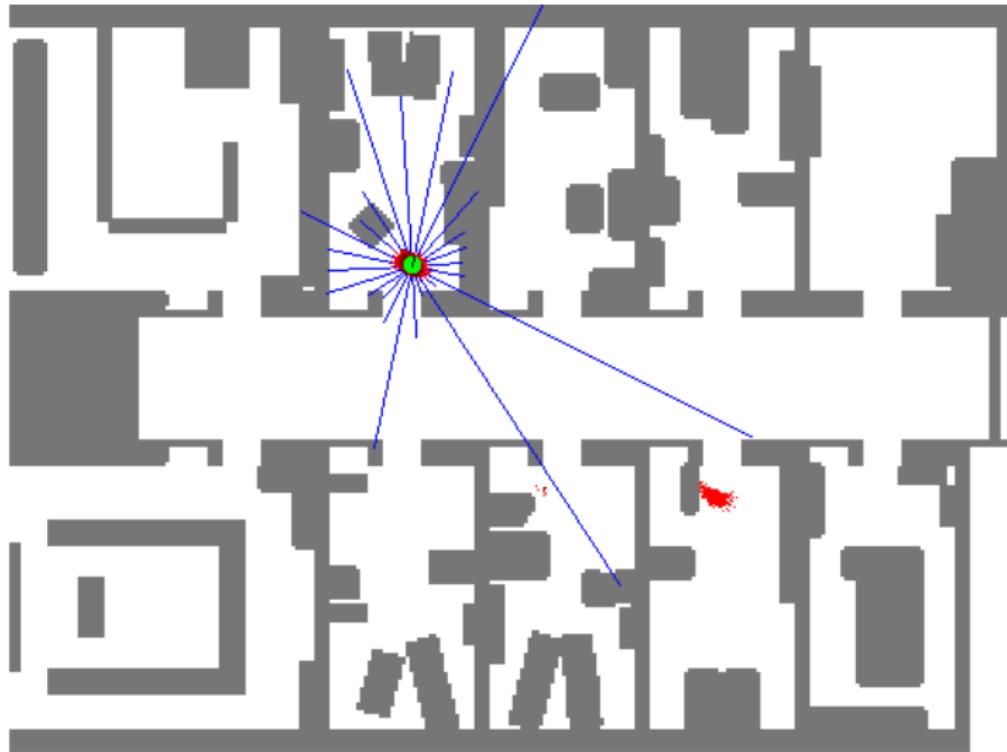


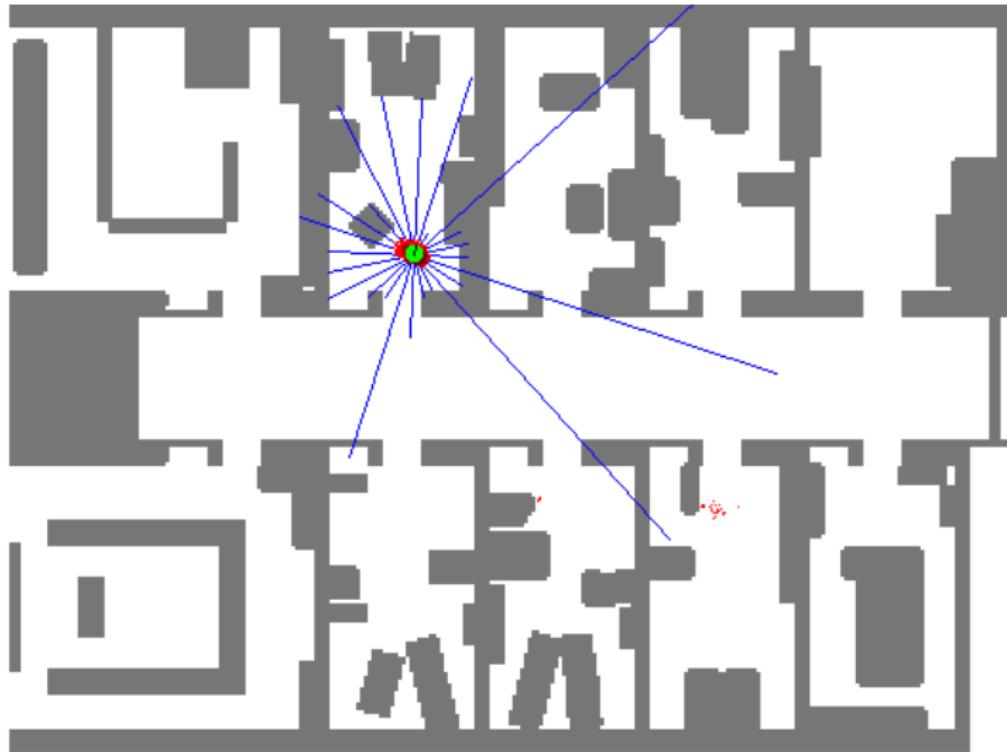


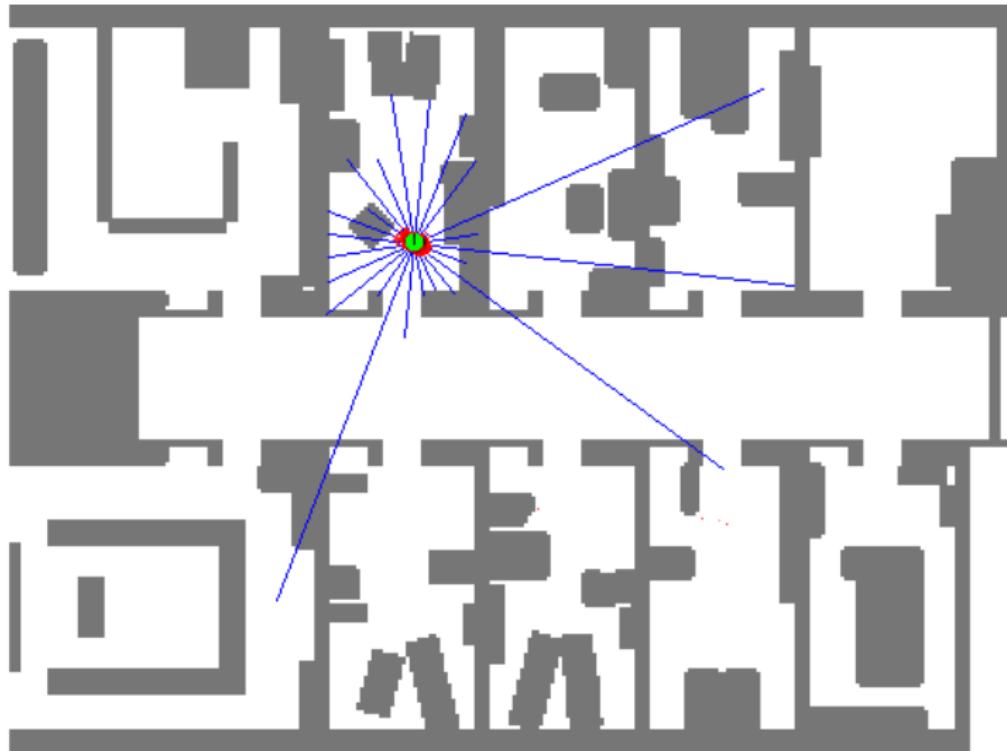


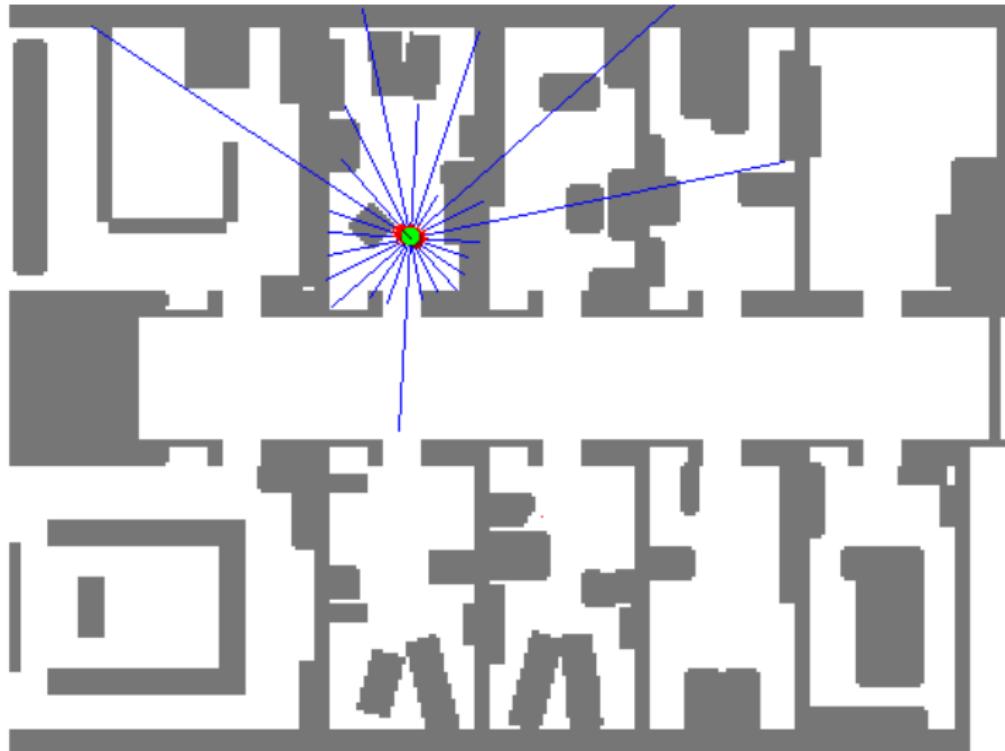


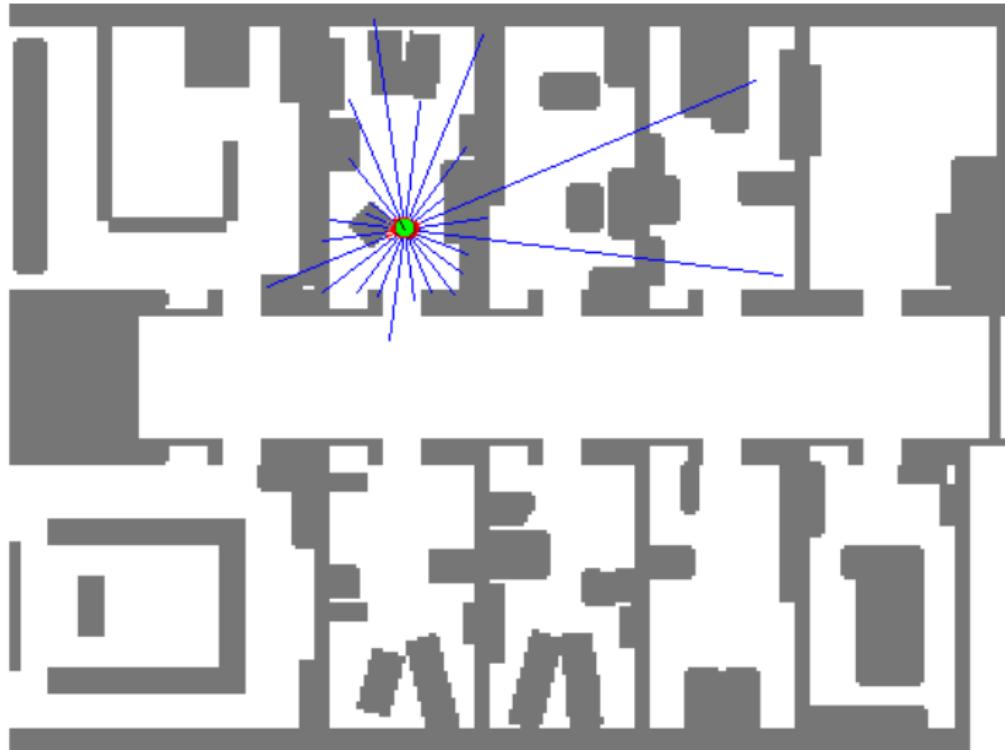












Part IV

Advanced Issues in Particle Filtering

14 The Auxiliary Particle Filter

- A New Interpretation of SISR
- The Auxiliary Trick
- Selecting the Weights in APF
- APF: Summary

Roadmap

14 The Auxiliary Particle Filter

- A New Interpretation of SISR
- The Auxiliary Trick
- Selecting the Weights in APF
- APF: Summary

Alternatives to SISR

- The resampling step in the SISR algorithm can be seen as a method to sample approximately under the distribution obtained when plugging the current particle approximation into the filtering update.
- This alternative way of thinking about resampling suggests several sequential Monte Carlo variants.

The Filtering Recursion Revisited

Recall that*

$$\phi_{k+1|k+1}(x) = \frac{\int \phi_{k|k}(dx_k) q(x_k, x) g(x, Y_{k+1})}{\iint \phi_{k|k}(dx_k) q(x_k, x') g(x', Y_{k+1}) dx'}.$$

Now, consider what happens when considering the **empirical filtering distribution**

$$\hat{\phi}_{k|k}^N = \sum_{i=1}^N \omega_k^i \delta_{\xi_k^i}$$

as an approximation to $\phi_{k|k}$ and plugging it into the previous relation.

* In this section we assume that the transition kernel q admits a transition density; this implies that the same is true for the actual filtering distributions $\phi_{k|k}$.

Filtering Target

One obtains a **mixture target pdf**

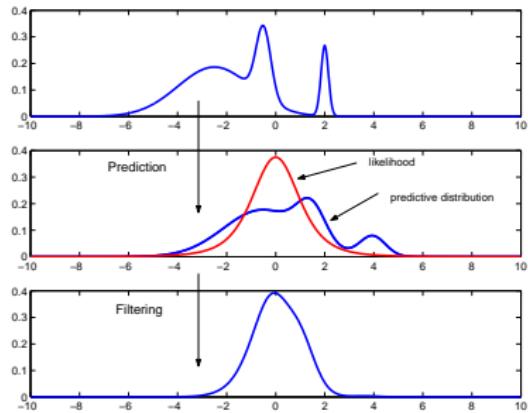
$$\begin{aligned}\phi_{k+1|k+1}^N(x) &= \frac{\sum_{i=1}^N \omega_k^i q(\xi_k^i, x) g(x, Y_{k+1})}{\sum_{i=1}^N \int \omega_k^i q(\xi_k^i, x) g(x, Y_{k+1}) dx} \\ &= \sum_{i=1}^N \frac{\omega_k^i \gamma_k(\xi_k^i)}{\sum_{j=1}^N \omega_k^j \gamma_k(\xi_k^j)} t_k(\xi_k^i, x),\end{aligned}$$

with

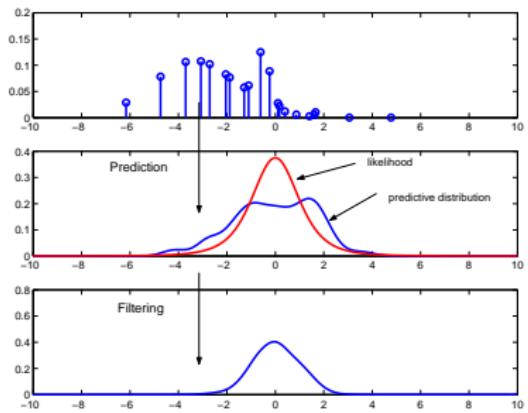
$$\begin{aligned}\gamma_k(x) &= \int q(x, dx') g(x', Y_{k+1}) dx' , \\ t_k(x, x') &= \frac{q(x, x') g(x', Y_{k+1})}{\gamma_k(x)},\end{aligned}$$

$t_k(x_k, x_{k+1})$ being the **optimal** proposal kernel.

Filtering Step



Approximation



The previous reinterpretation of SISR shows that resampling and particle update can be integrated into a single **global IS step that targets $\phi_{k+1|k+1}^N$** .

But,

- how to chose the global instrumental kernel
 $r_k^{\text{glob}}((\xi_k^{1:N}, \omega_k^{1:N}), x)$?
- as $\phi_{k+1|k+1}^N$ is a mixture of N densities, evaluation of the IS weights may necessitate of the order of N^2 computations, which is computationally unfeasible.

The previous reinterpretation of SISR shows that resampling and particle update can be integrated into a single **global IS step that targets $\phi_{k+1|k+1}^N$** .

But,

- how to chose the global instrumental kernel
 $r_k^{\text{glob}}((\xi_k^{1:N}, \omega_k^{1:N}), x)$?
- as $\phi_{k+1|k+1}^N$ is a mixture of N densities, evaluation of the IS weights may necessitate of the order of N^2 computations, which is computationally unfeasible.

Remark The global instrumental kernel that corresponds to SISR is

$$r_k^{\text{glob}}((\xi_k^{1:N}, \omega_k^{1:N}), x) = \sum_{i=1}^N \omega_k^i q(\xi_k^i, x).$$

The Auxiliary Trick

Assume that the global instrumental kernel is a mixture of the form

$$r_k^{\text{glob}}((\xi_k^{1:N}, \omega_k^{1:N}), x) = \sum_{i=1}^N \tau_k^i r_k^i(\xi_k^i, x).$$

To avoid the N^2 update, one can use **data augmentation**.

The Auxiliary Trick (Contd.)

Introducing the mixture component as an auxiliary variable:

$$r_k^{\text{aux}}((\xi_k^{1:N}, \omega_k^{1:N}), (i, x)) = \tau_k^i r_k^i(\xi_k^i, x)$$

defines a pdf on the **product space** $\{1, \dots, N\} \times X$, whose marginal density is

$$r_k^{\text{glob}}((\xi_k^{1:N}, \omega_k^{1:N}), x) = \sum_{i=1}^N r_k^{\text{aux}}((\xi_k^{1:N}, \omega_k^{1:N}), (i, x)).$$

The Auxiliary Trick (Contd.)

Introducing the mixture component as an auxiliary variable:

$$r_k^{\text{aux}}((\xi_k^{1:N}, \omega_k^{1:N}), (i, x)) = \tau_k^i r_k^i(\xi_k^i, x)$$

defines a pdf on the **product space** $\{1, \dots, N\} \times X$, whose marginal density is

$$r_k^{\text{glob}}((\xi_k^{1:N}, \omega_k^{1:N}), x) = \sum_{i=1}^N r_k^{\text{aux}}((\xi_k^{1:N}, \omega_k^{1:N}), (i, x)).$$

Likewise,

$$\phi_{k+1|k+1}^{n,\text{aux}}(i, x) = \frac{1}{c} \omega_k^i q(\xi_k^i, x) g(x, Y_{k+1}) ,$$

where $c = \sum_{i=1}^N \int \omega_k^i q(\xi_k^i, x) g(x, Y_{k+1}) dx$, is the auxiliary target whose marginal density is

$$\phi_{k+1|k+1}^N(x) = \sum_{i=1}^N \phi_{k+1|k+1}^{n,\text{aux}}(i, x) .$$

Auxiliary Sampling Algorithm (Pitt & Shephard, 1999)

Auxiliary Particle Filter (APF)

Repeat N times independently, for $i = 1, \dots, N$,

- 1 Sample the **ancestor's index** J_k^i in $\{1, \dots, N\}$ with probabilities $(\tau_k^1, \dots, \tau_k^N)$.
- 2 Sample the position of the **new particle** ξ_{k+1}^i from $r_k^i(\xi_k^{J_k^i}, x)$.
- 3 Compute the auxiliary IS weight

$$\omega_{k+1}^i \propto \frac{\omega_k^{J_k^i} q(\xi_k^{J_k^i}, \xi_{k+1}^i) g(\xi_{k+1}^i, Y_{k+1})}{\tau_k^{J_k^i} r_k^i(\xi_k^{J_k^i}, \xi_{k+1}^i)}.$$

How Does it Relate to Standard SISR?

Connection to bootstrap filter: proposing from the prior kernel ($r = q$),

Bootstrap Filter

- 1 $J_k^i \sim \text{Mult}(1; \omega_k^1, \dots, \omega_k^N)$ (resampling),
- 2 $\xi_{k+1}^i \sim q(\xi_k^{J_k^i}, x)$,
- 3 $\omega_{k+1}^i \propto g(\xi_{k+1}^i, Y_{k+1})$.

Global Proposal

- 1 $J_k^i \sim \text{Mult}(1; \tau_k^1, \dots, \tau_k^N)$,
- 2 $\xi_{k+1}^i \sim q(\xi_k^{J_k^i}, x)$,
- 3 $\omega_{k+1}^i \propto \frac{\sum_{i=1}^N \omega_k^i q(\xi_k^i, \xi_{k+1}^i) g(\xi_{k+1}^i, Y_{k+1})}{\sum_{j=1}^N \omega_k^j q(\xi_k^j, \xi_{k+1}^j)}$.

Auxiliary Particle Filter

- 1 $J_k^i \sim \text{Mult}(1; \tau_k^1, \dots, \tau_k^N)$,
- 2 $\xi_{k+1}^i \sim q(\xi_k^{J_k^i}, x)$,
- 3 $\omega_{k+1}^i \propto \frac{\omega_k^{J_k^i}}{\tau_k^{J_k^i}} g(\xi_{k+1}^i, Y_{k+1})$.

How Does it Relate to Standard SISR?

Connection to bootstrap filter: proposing from the prior kernel ($r = q$),

Bootstrap Filter

- 1 $J_k^i \sim \text{Mult}(1; \omega_k^1, \dots, \omega_k^N)$ (resampling),
- 2 $\xi_{k+1}^i \sim q(\xi_k^{J_k^i}, x)$,
- 3 $\omega_{k+1}^i \propto g(\xi_{k+1}^i, Y_{k+1})$.

Global Proposal

- 1 $J_k^i \sim \text{Mult}(1; \tau_k^1, \dots, \tau_k^N)$,
- 2 $\xi_{k+1}^i \sim q(\xi_k^{J_k^i}, x)$,
- 3 $\omega_{k+1}^i \propto \frac{\sum_{i=1}^N \omega_k^i q(\xi_k^i, \xi_{k+1}^i) g(\xi_{k+1}^i, Y_{k+1})}{\sum_{j=1}^N \omega_k^j q(\xi_k^j, \xi_{k+1}^j)}$.

Auxiliary Particle Filter

- 1 $J_k^i \sim \text{Mult}(1; \tau_k^1, \dots, \tau_k^N)$,
- 2 $\xi_{k+1}^i \sim q(\xi_k^{J_k^i}, x)$,
- 3 $\omega_{k+1}^i \propto \frac{\omega_k^{J_k^i}}{\tau_k^{J_k^i}} g(\xi_{k+1}^i, Y_{k+1})$.

- 1 The auxiliary view provides a new degree of freedom by allowing to chose the proposal weights τ_k^i
- 2 It has the same computational complexity as SISR
- 3 One can also use particle-dependent instrumental kernels r_k^i

IID sampling or Fully Adapted APF

If sampling from the optimal kernel

$$t_k(x, x') = \frac{q(x, x')g(x', Y_{k+1})}{\gamma_k(x)}$$

is feasible.

Standard SISR

- 1 $J_k^i \sim \text{Mult}(1; \omega_k^1, \dots, \omega_k^N)$
(resampling),
- 2 $\xi_{k+1}^i \sim t_k(\xi_k^{J_k^i}, x),$
- 3 $\omega_{k+1}^i \propto \gamma_k(\xi_k^i).$

Fully Adapted APF

- 1 $J_k^i \sim \text{Mult}(1; \tau_k^1, \dots, \tau_k^N),$ where
 $\tau_k^i \propto \gamma_k(\xi_k^i),$
- 2 $\xi_{k+1}^i \sim t_k(\xi_k^{J_k^i}, x),$
- 3 $\omega_{k+1}^i \propto 1.$

IID sampling or Fully Adapted APF

If sampling from the optimal kernel

$$t_k(x, x') = \frac{q(x, x')g(x', Y_{k+1})}{\gamma_k(x)}$$

is feasible.

Standard SISR

- 1 $J_k^i \sim \text{Mult}(1; \omega_k^1, \dots, \omega_k^N)$
(resampling),
- 2 $\xi_{k+1}^i \sim t_k(\xi_k^{J_k^i}, x),$
- 3 $\omega_{k+1}^i \propto \gamma_k(\xi_k^i).$

Fully Adapted APF

- 1 $J_k^i \sim \text{Mult}(1; \tau_k^1, \dots, \tau_k^N),$ where
 $\tau_k^i \propto \gamma_k(\xi_k^i),$
- 2 $\xi_{k+1}^i \sim t_k(\xi_k^{J_k^i}, x),$
- 3 $\omega_{k+1}^i \propto 1.$

The fully adapted APF resamples but does not necessitate the use of weights compared to standard SISR which resamples, propagates and then reweights.

The Art of APF Lies In the Choice of the Weights τ_k^1 , Usually Based on Y_{k+1}

Gaussian Model with Occlusion

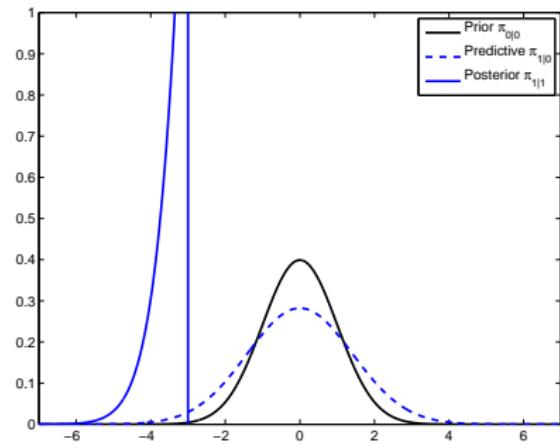
- $\phi_{0|0} = \mathcal{N}(0, 1)$ and $X_0^i \sim \nu$,
- $q(x_0, \cdot) = \mathcal{N}(\cdot; x_0, 1)$,
- $g(x_1, Y_1) = \mathbb{1}(x_1 \leq \kappa)$.

The target $\phi_{1|1}$ is the truncated Gaussian pdf

$$\mathcal{N}(x; 0, 2)\mathbb{1}(x \leq \kappa)/(2\Phi(\kappa/2))$$

The optimal proposal kernel $t(x, x')$ is

$$\mathcal{N}(x'; x, 1)\mathbb{1}(x' \leq \kappa)/\Phi(\kappa - x)$$



$$\kappa = -3$$

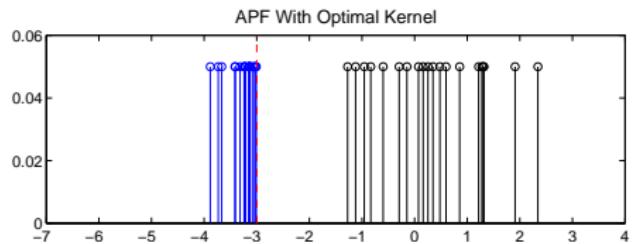
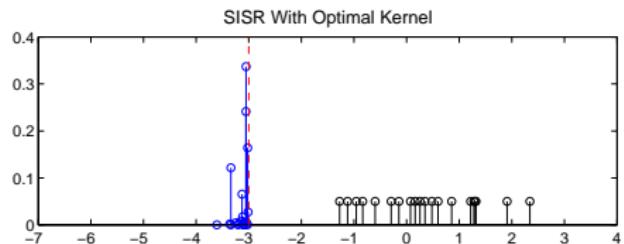
The Optimal Kernel in Action

SISR

- $\xi_1^i \sim \frac{\mathcal{N}(x; \xi_0^i, 1) \mathbb{1}(x \leq \kappa)}{\Phi(\kappa - \xi_0^i)},$
- $\omega_1^i \propto \Phi(\kappa - \xi_0^i).$

APF

- $\tau_0^i \propto \Phi(\kappa - \xi_0^i),$
- $J_0^i \sim \text{Mult}(1; \tau_0^1, \dots, \tau_0^N),$
- $\xi_1^i \sim \frac{\mathcal{N}(x; \xi_0^{J_0^i}, 1) \mathbb{1}(x \leq \kappa)}{\Phi(\kappa - \xi_0^{J_0^i})},$
- $\omega_1^i = 1/n.$

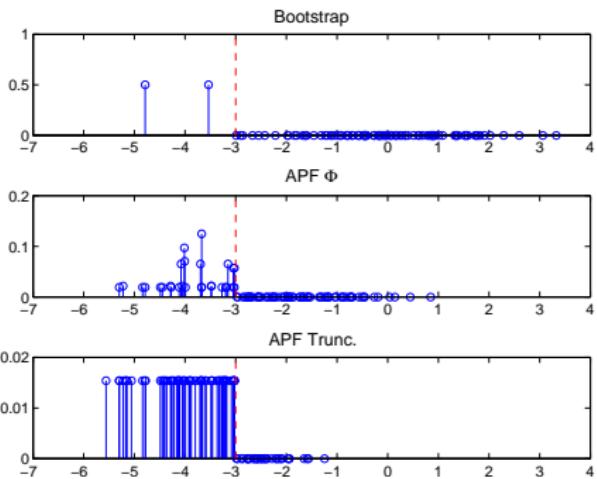


$n = 20$ particles

With the optimal kernel, APF is preferable to SISR (Douc & Moulines, 2008)

Compare three strategies that use the proposal $r = q$

- 1 Bootstrap filter (without resampling as $W_0^i = 1/n$),
- 2 APF with
 $\tau_k^i \propto \Phi(\kappa - \xi_0^i)$,
- 3 APF with
 $\tau_k^i \propto \max(\mathbb{1}(\xi_0^i \leq \kappa), \varepsilon)$,
and $\varepsilon = 10^{-3}$.



$n = 100$ particles

What About Principled Ways of Selecting the Weights?

Though question with few really practical answers :(

In Our Last Example

- Strategy 2 (APF Φ) is preferable, yielding an asymptotic decrease in variance (compared to the bootstrap filter) by a factor 2.3 (note that τ_k^i correspond in this case to the weights associated to the optimal proposal t).
- Strategy 3 (APF Trunc.) is worse than bootstrap with an increase of asymptotic variance by a factor 2.1.
- For moderate values of n ($50 < n < 500$), APF Φ is clearly preferable to the other two; APF Trunc. often display a deceptively high ESS (and very biased estimates).

A Word of Caution

In Our Example (Contd.)

- APF Trunc. with $\varepsilon = 0$ can be shown to be biased!

A Word of Caution

In Our Example (Contd.)

- APF Trunc. with $\varepsilon = 0$ can be shown to be biased!

Beware that SMC is based on IS and that excessive “optimization” in SMC may result in methods that are not consistent and/or robust.

More precise ideas on APF to be found in (Johansen & Doucet, 2008) and (Douc *et al.*, 2008)

Summary

- APF may be understood as an IS method that targets the propagation of the empirical filtering distribution using the exact filtering recursion.
- APF uses an auxiliary variable (ie. keeps track of the particle history) to keep weight computation tractable.
- APF offers a new degree of freedom by allowing to chose the resampling weights and proposal kernels.
- APF has the potential to improve over SISR but this depends on the choice of weights.
- Suitable setting of the weights depends on the model.

Part V

Smoothing Algorithms

15 Smoothing

- The Forward Filtering Backward Smoothing Algorithm
- The Forward Filtering Backward Simulation Algorithm

16 Smoothing Algorithms

- Path space method
- FFBS method
- FFBSi method
- Simulations

Roadmap

15 Smoothing

- The Forward Filtering Backward Smoothing Algorithm
- The Forward Filtering Backward Simulation Algorithm

16 Smoothing Algorithms

- Path space method
- FFBS method
- FFBSi method
- Simulations

Motivation

- smoothing is relevant in complex dynamical systems, since filtering alone will often yield only fairly uninformative state estimates, while the **lookahead** (when possible) yields more accurate estimates.
- The basic **filtering** version of the particle filter actually provides **smoothing for free**... the stored particle trajectories $\{\xi_{0:k}^i\}$ and their associated weights $\{\omega_k^i\}$ can be considered as a weighted sample from the joint **smoothing distribution** $\phi_{0:k|0:k}(x_{0:k})$.
- While this scheme can be successful for certain models and small lags L , resampling procedures will make this estimator very inaccurate !

Illustration for the Bootstrap Filter on a Toy Example

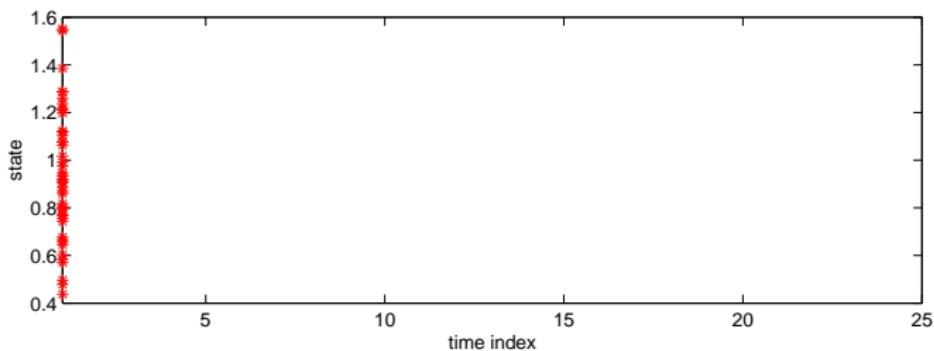
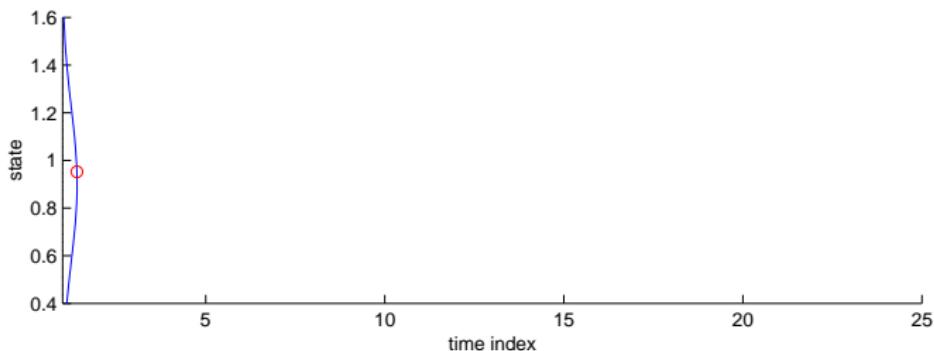
- Noisy AR(1) model

$$X_{k+1} - \mu = \phi(X_k - \mu) + \sigma U_k$$

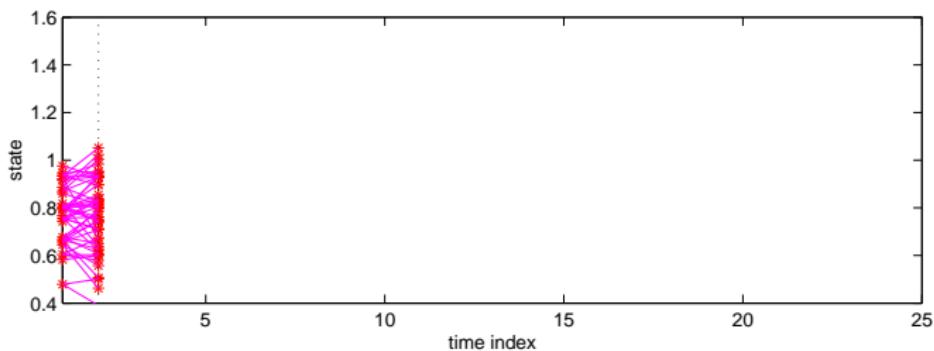
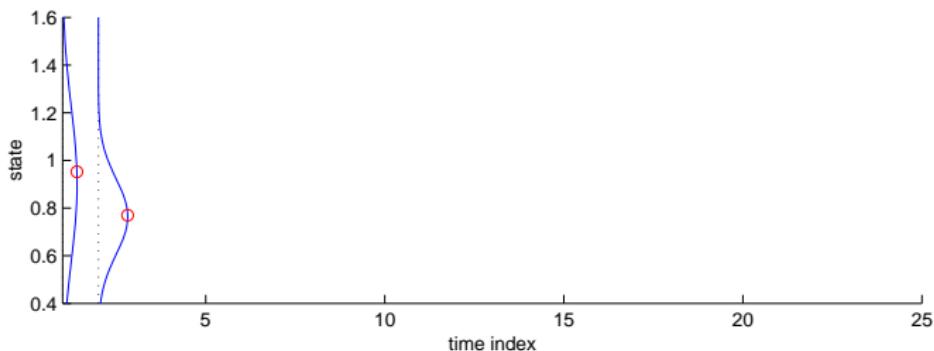
$$Y_k = X_k + \eta V_k$$

$$\mu = 0.9, \phi = 0.95, \sigma^2 = 0.01, \eta^2 = 0.02 = (\sigma^2 / (1 - \phi^2)) / 5$$

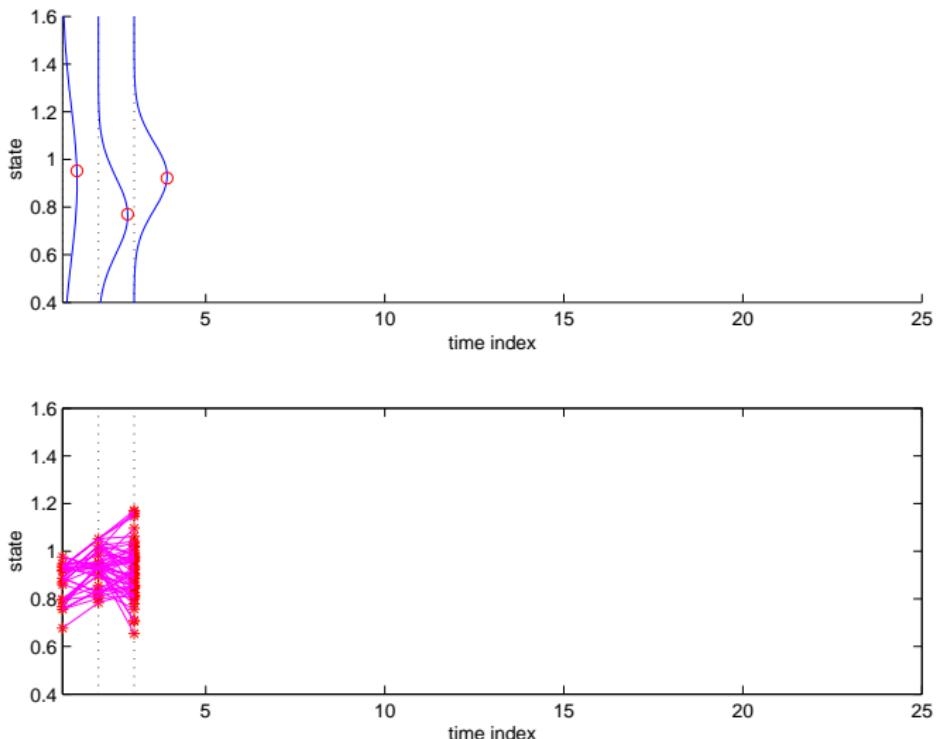
- To approximate the predictive distribution $\phi_{k+1|k}$, we use the bootstrap filter with $N = 50$ particles, plotting the full particle paths $\{\xi_{0:k}^i, \tilde{\xi}_{k+1}^i\}_{1 \leq i \leq N}$ for each time index.
- This example is used since we may also compute the actual filtering densities using Kalman filtering



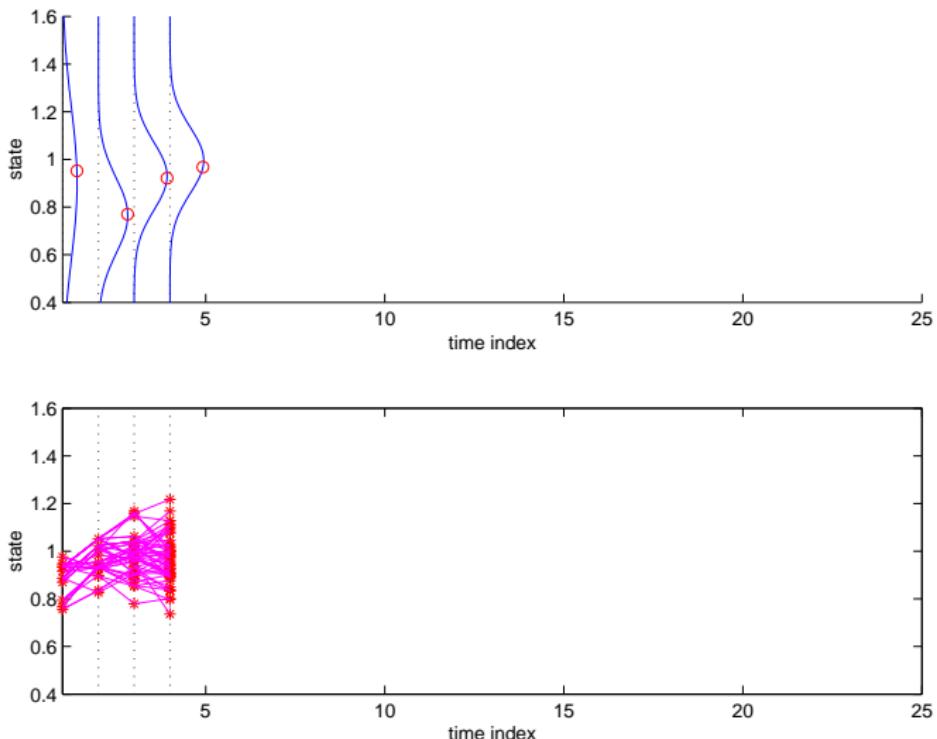
Predictive densities and evolution of the particle ancestry tree



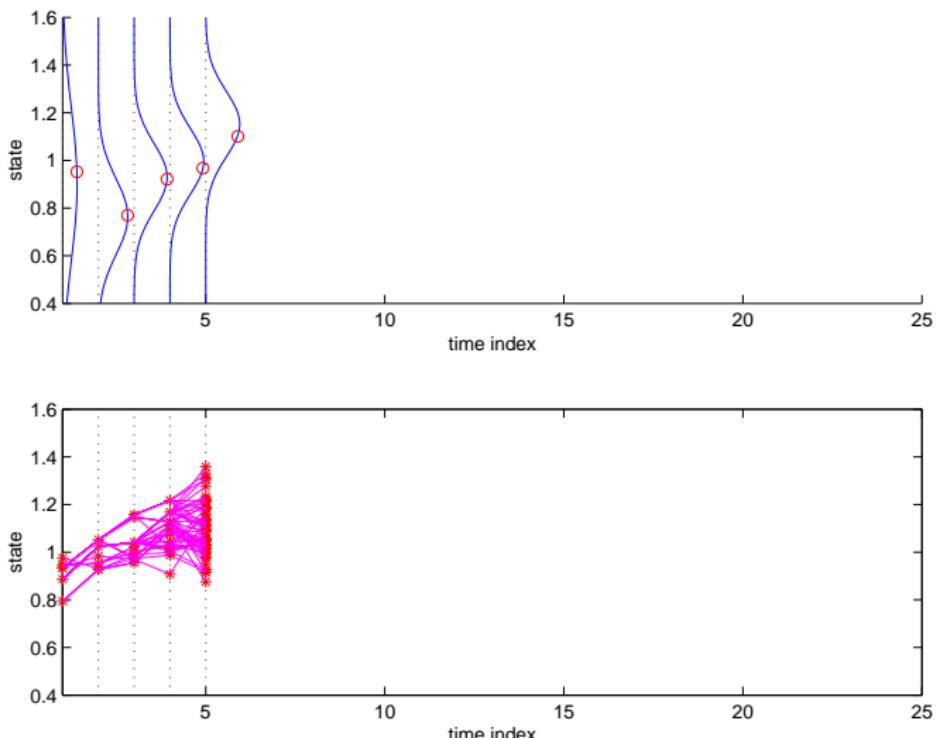
Predictive densities and evolution of the particle ancestry tree



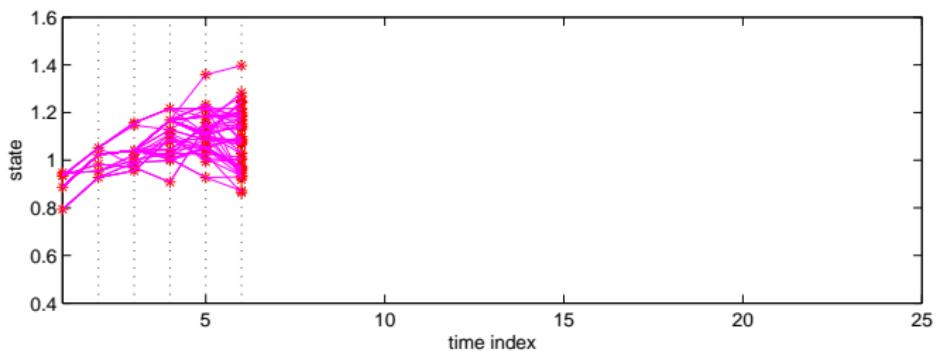
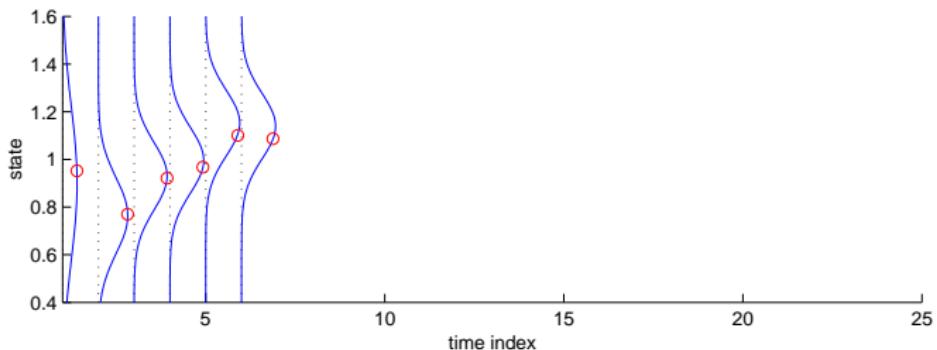
Predictive densities and evolution of the particle ancestry tree



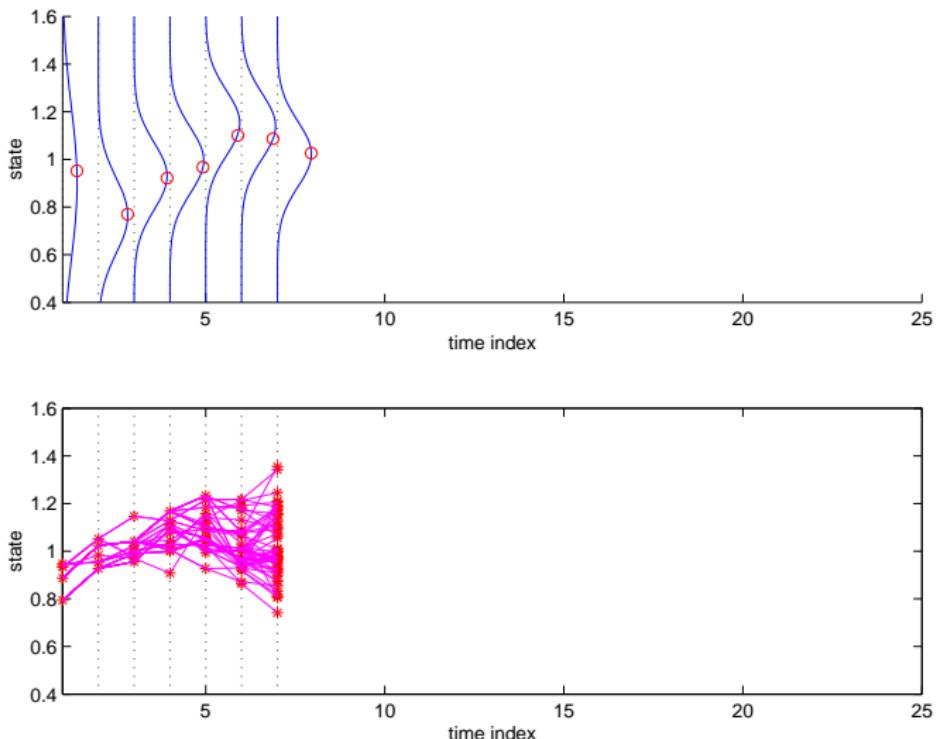
Predictive densities and evolution of the particle ancestry tree



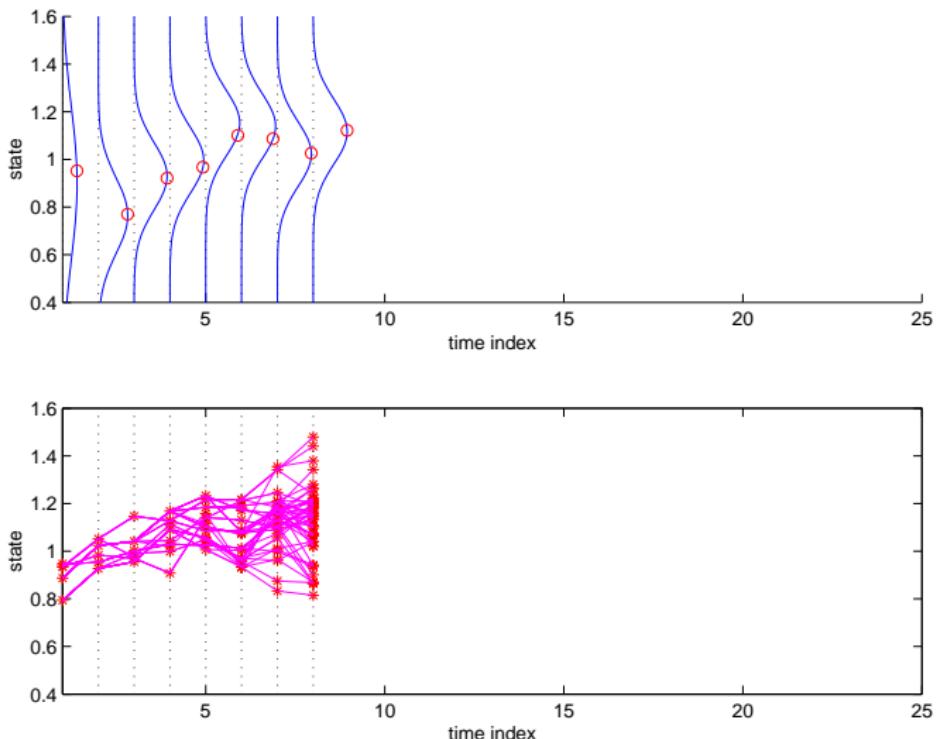
Predictive densities and evolution of the particle ancestry tree



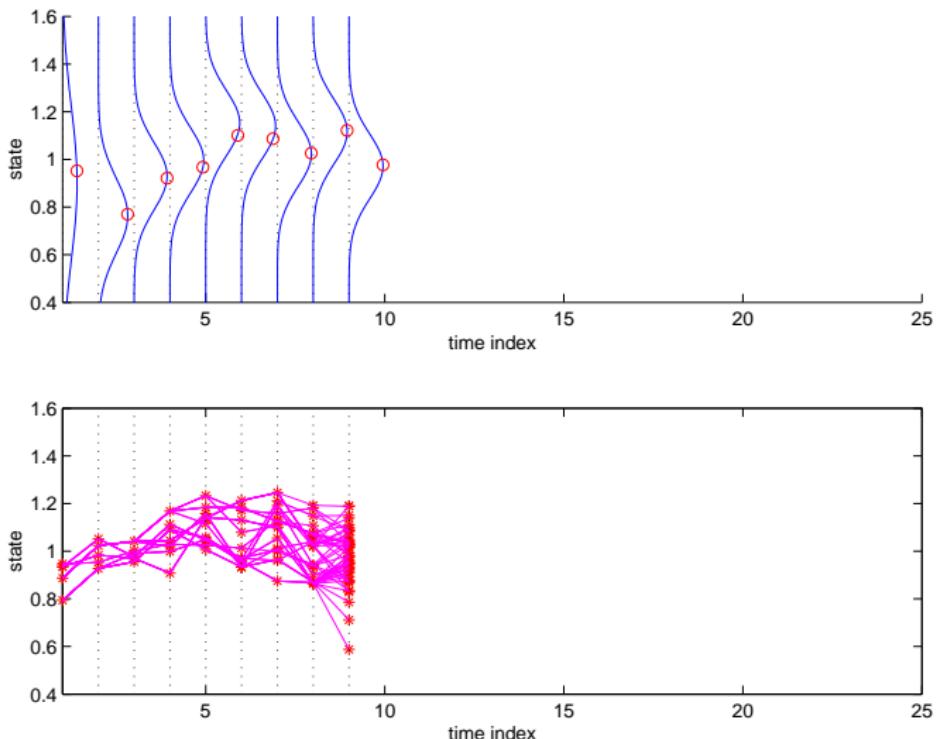
Predictive densities and evolution of the particle ancestry tree



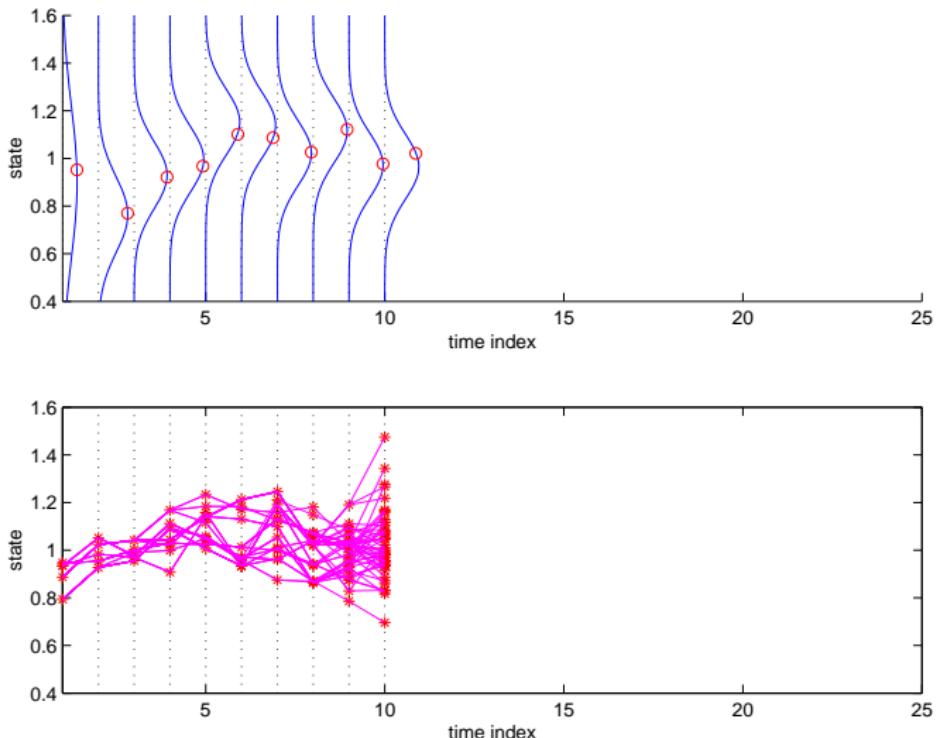
Predictive densities and evolution of the particle ancestry tree



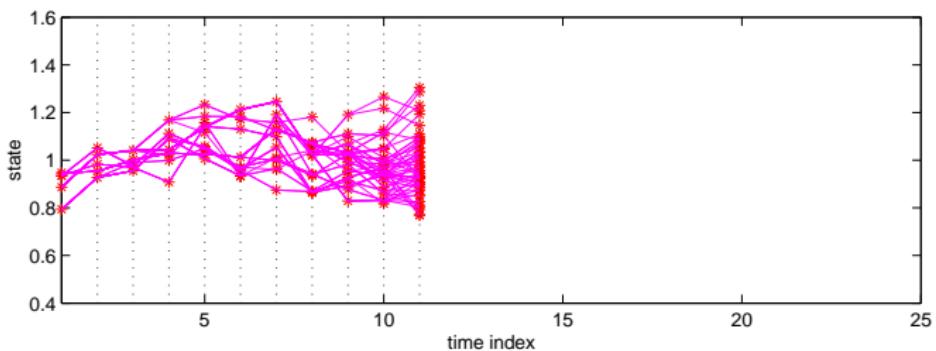
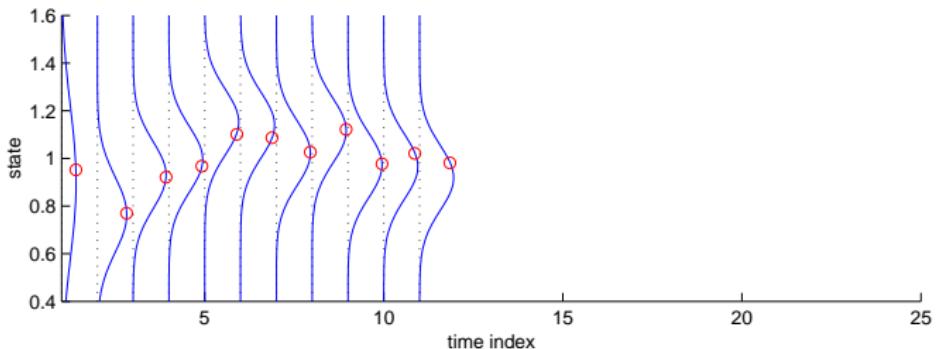
Predictive densities and evolution of the particle ancestry tree



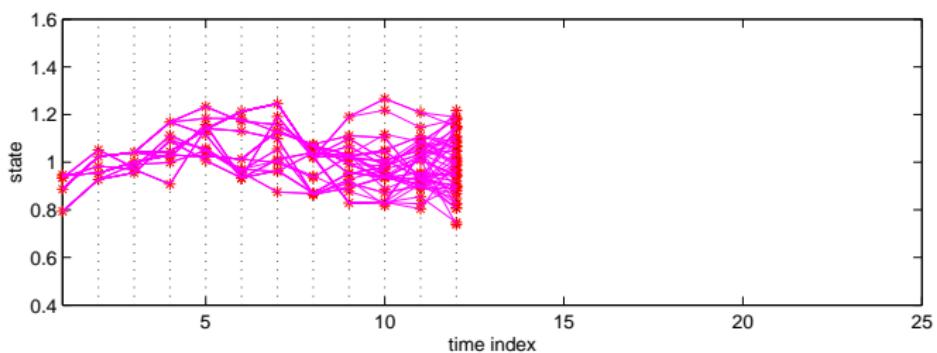
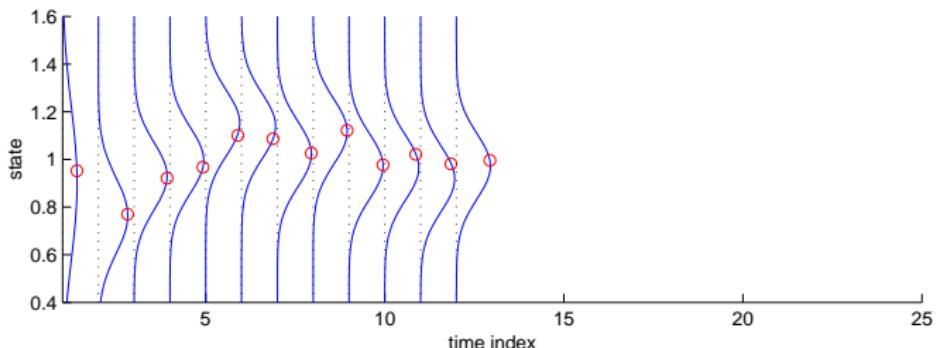
Predictive densities and evolution of the particle ancestry tree



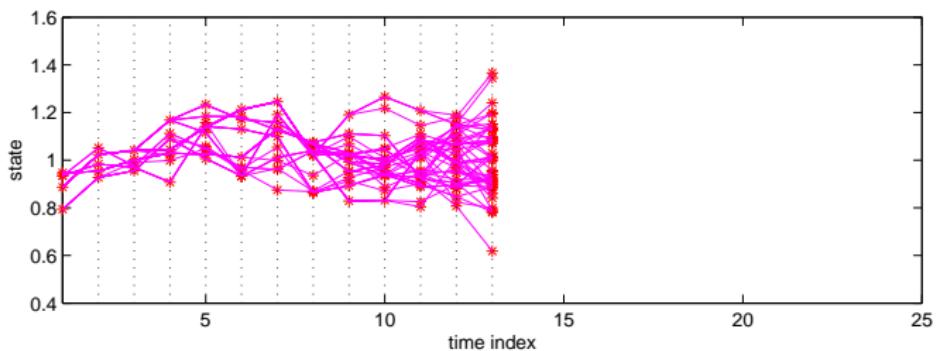
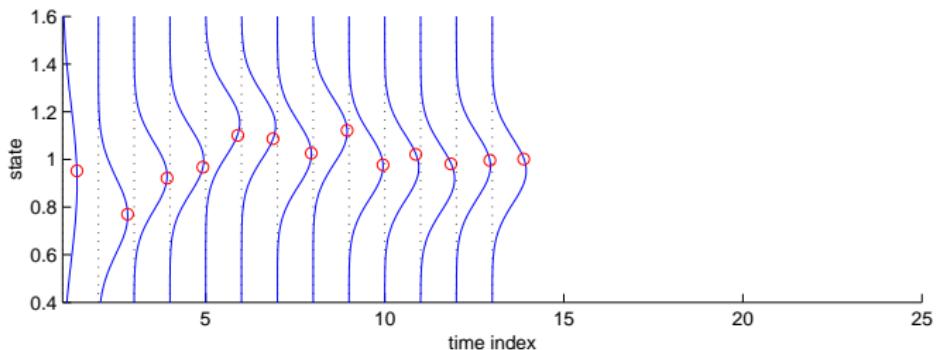
Predictive densities and evolution of the particle ancestry tree



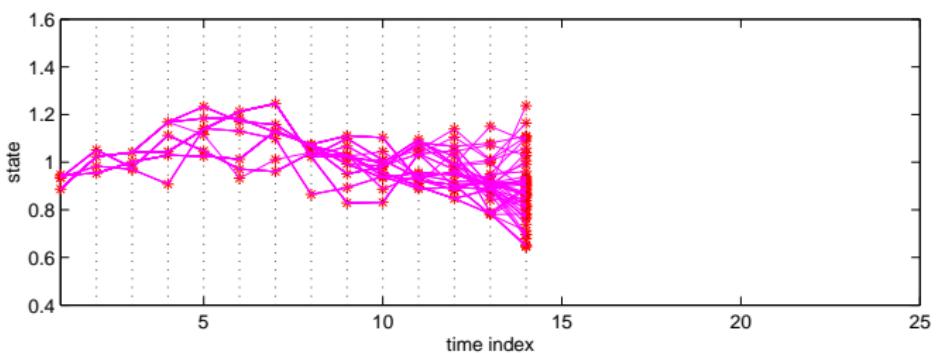
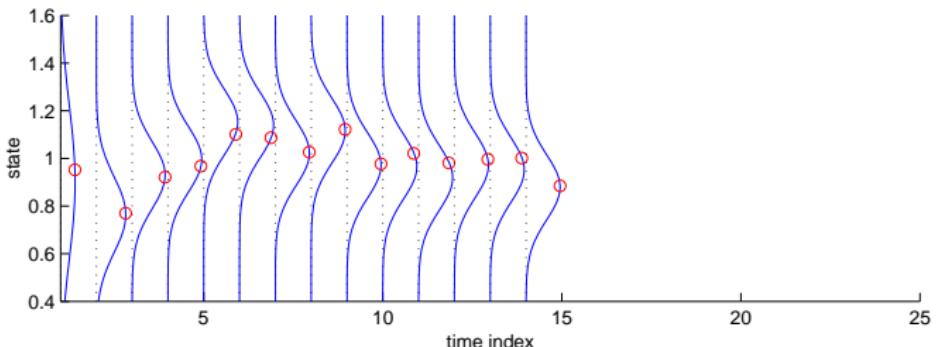
Predictive densities and evolution of the particle ancestry tree



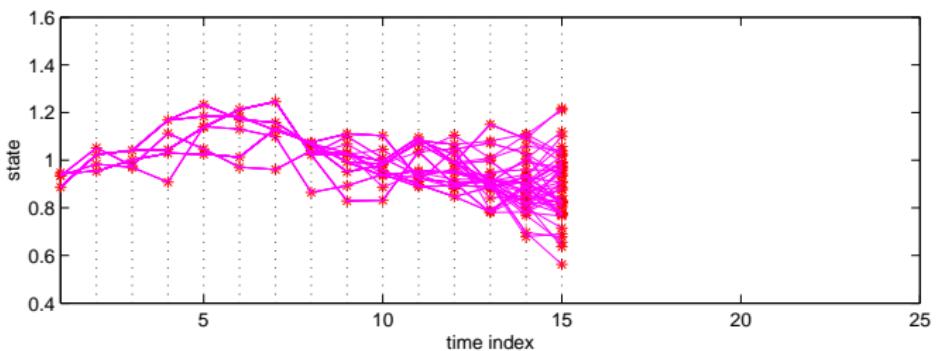
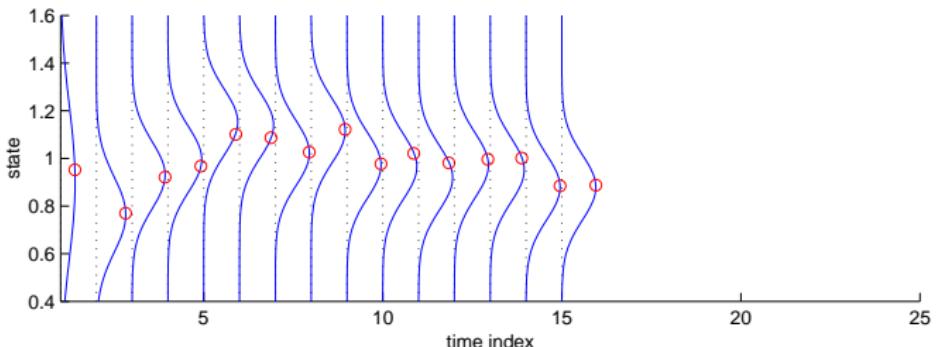
Predictive densities and evolution of the particle ancestry tree



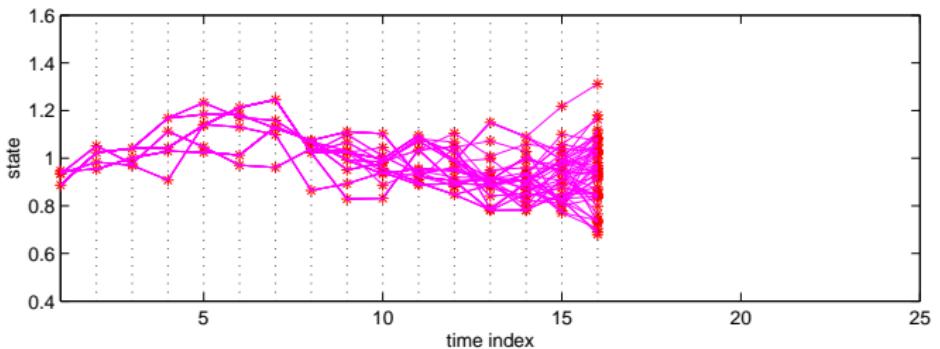
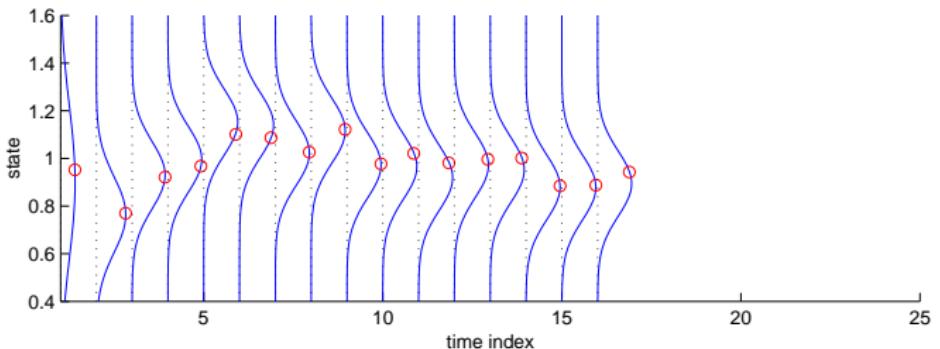
Predictive densities and evolution of the particle ancestry tree



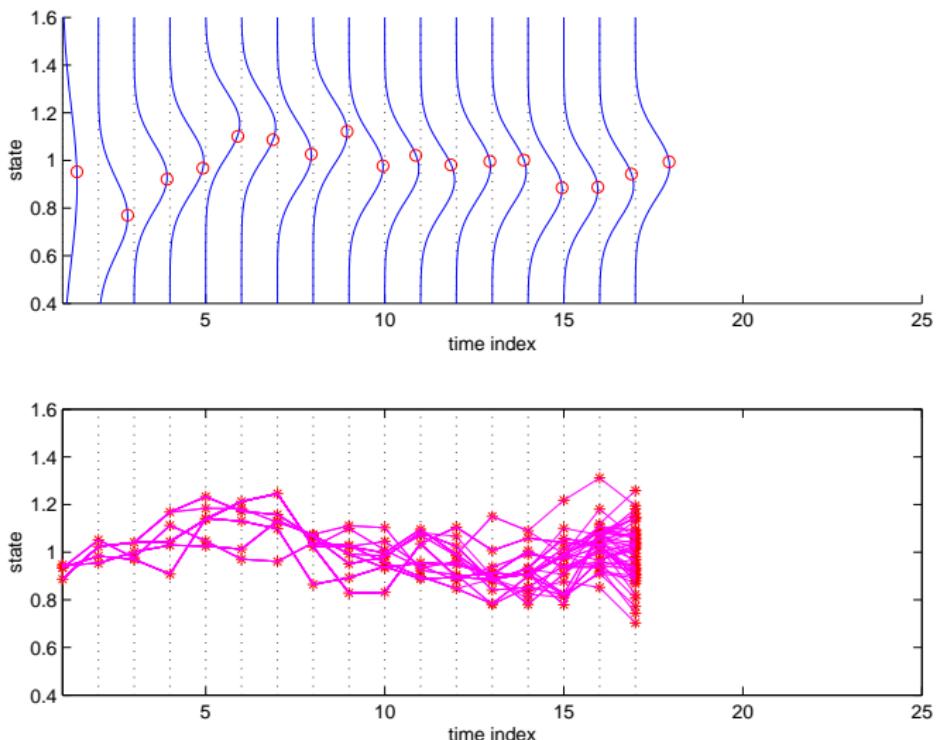
Predictive densities and evolution of the particle ancestry tree



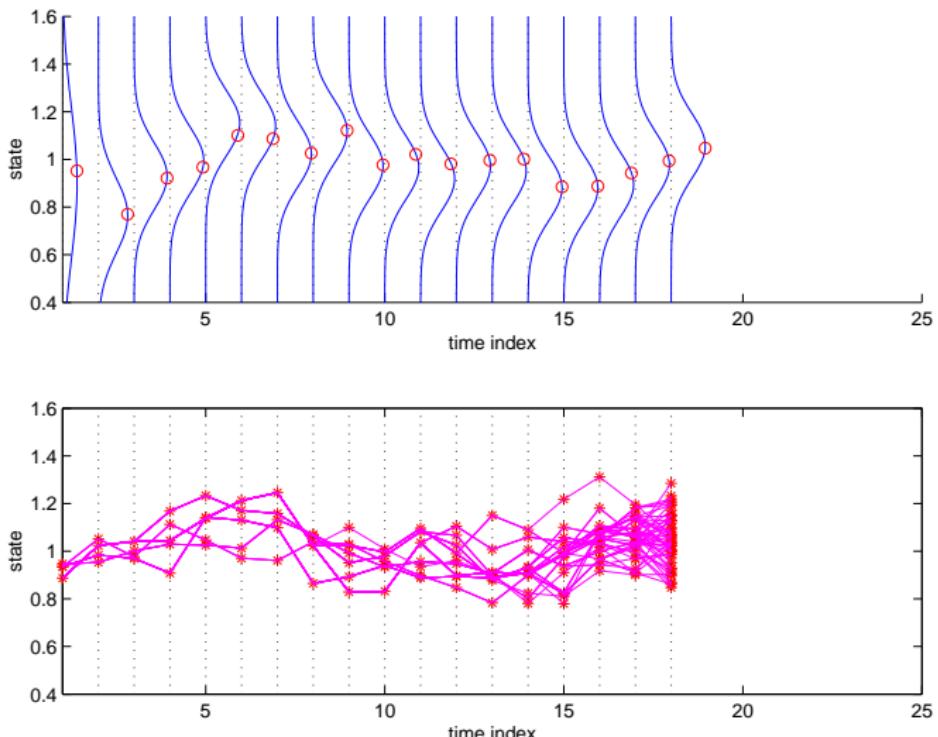
Predictive densities and evolution of the particle ancestry tree



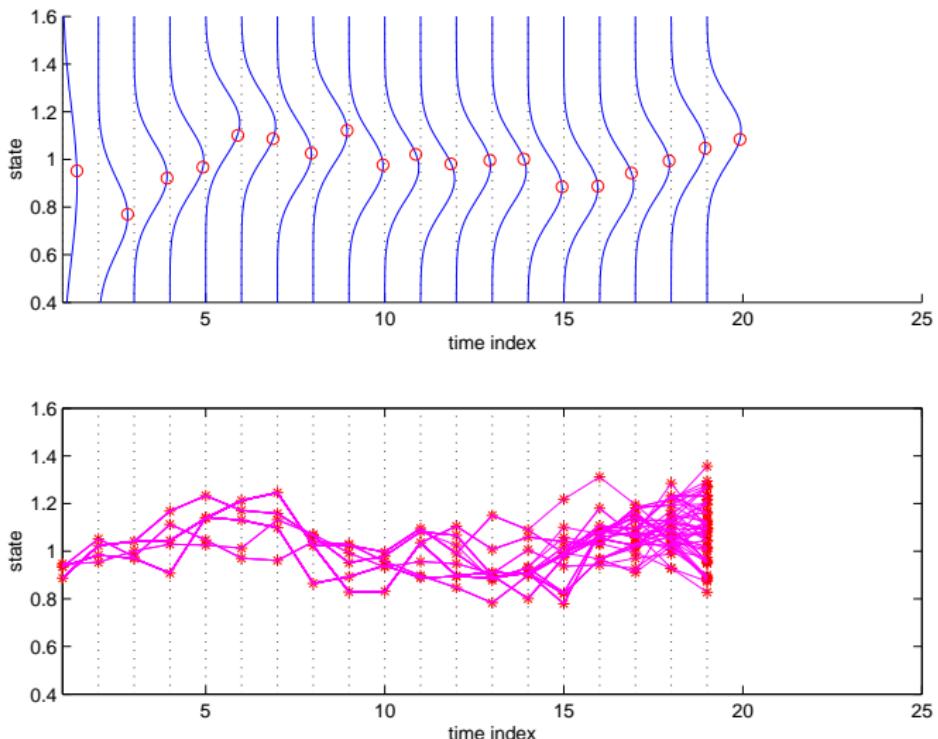
Predictive densities and evolution of the particle ancestry tree



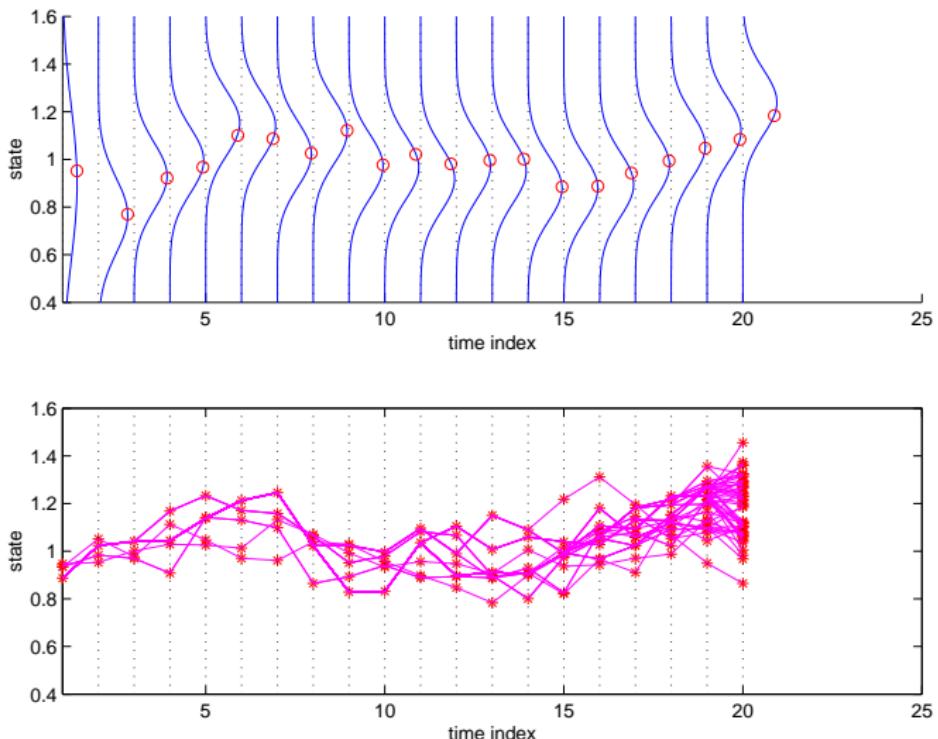
Predictive densities and evolution of the particle ancestry tree



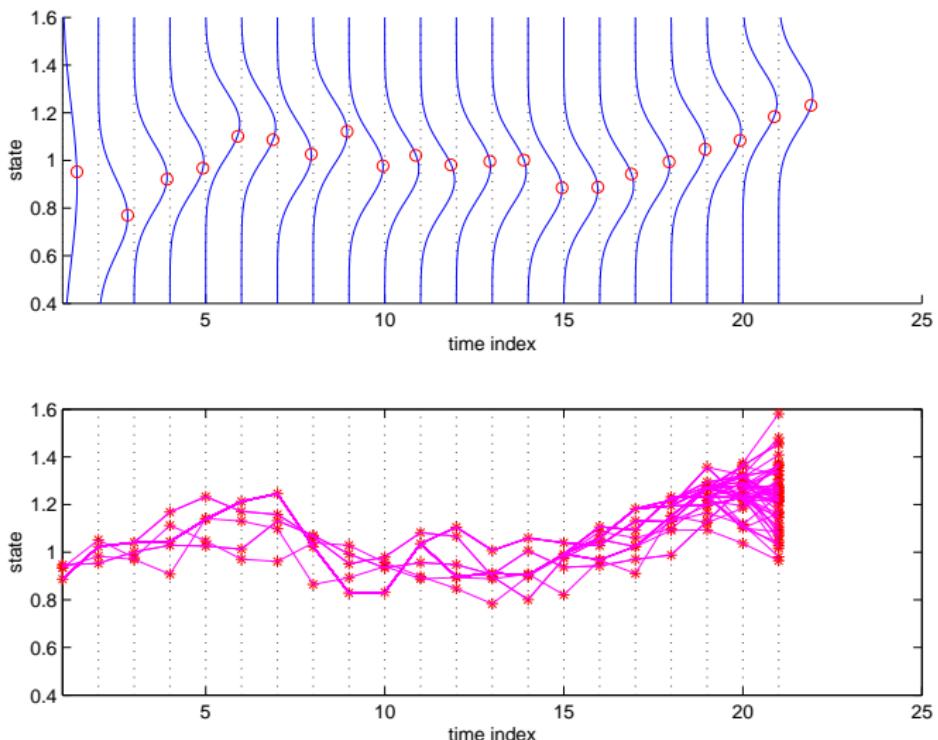
Predictive densities and evolution of the particle ancestry tree



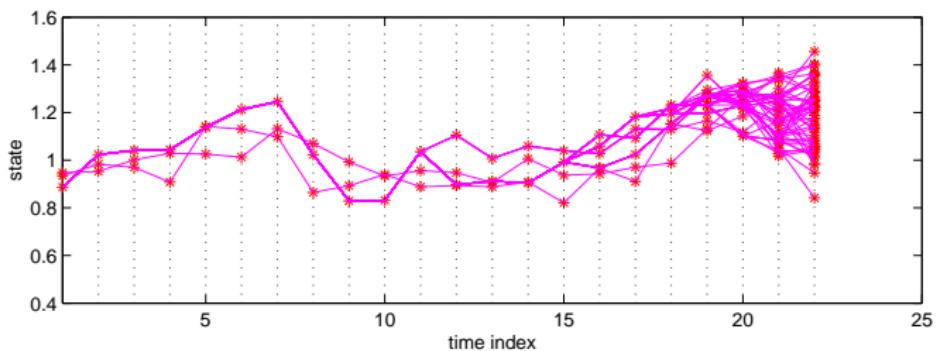
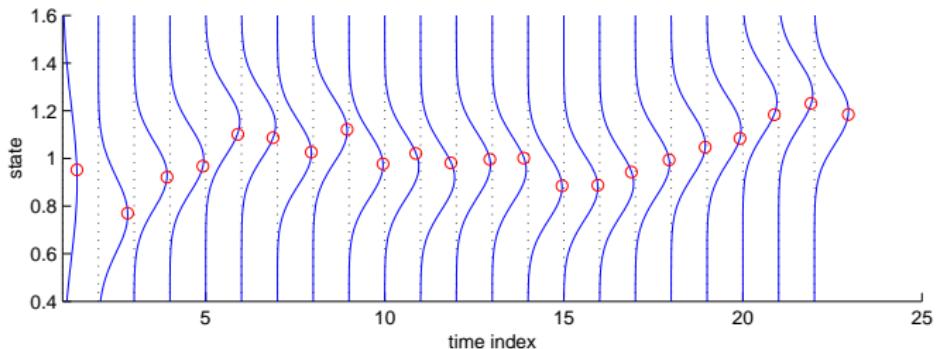
Predictive densities and evolution of the particle ancestry tree



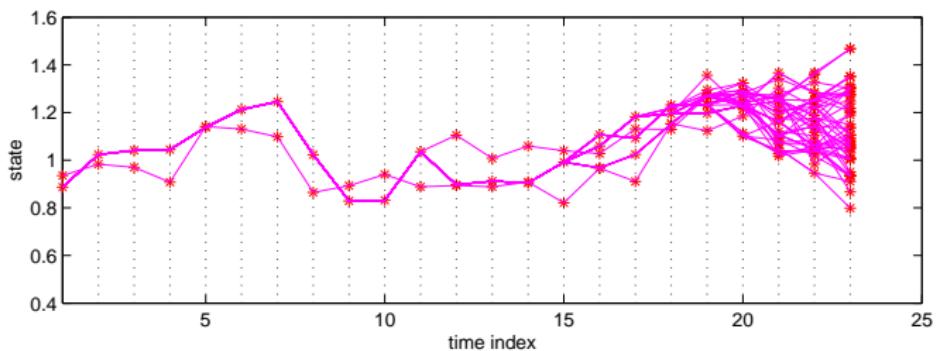
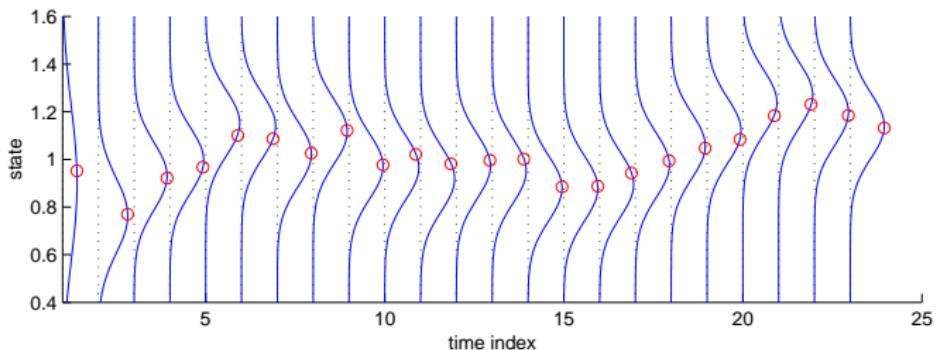
Predictive densities and evolution of the particle ancestry tree



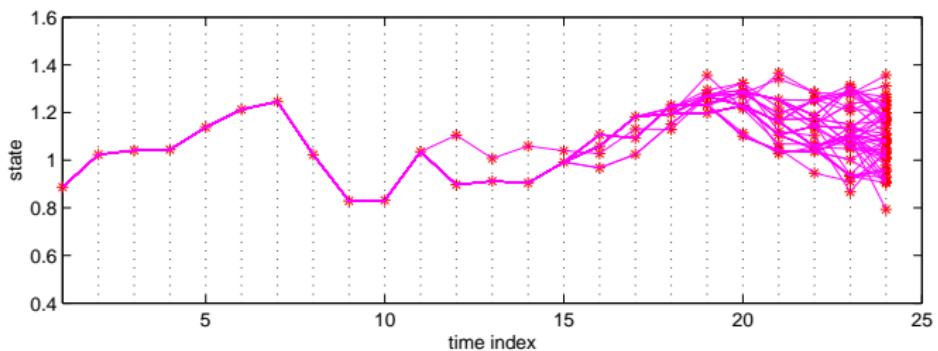
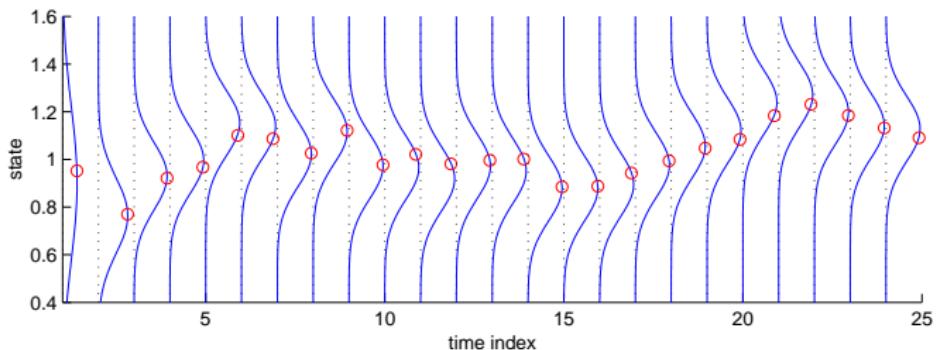
Predictive densities and evolution of the particle ancestry tree



Predictive densities and evolution of the particle ancestry tree

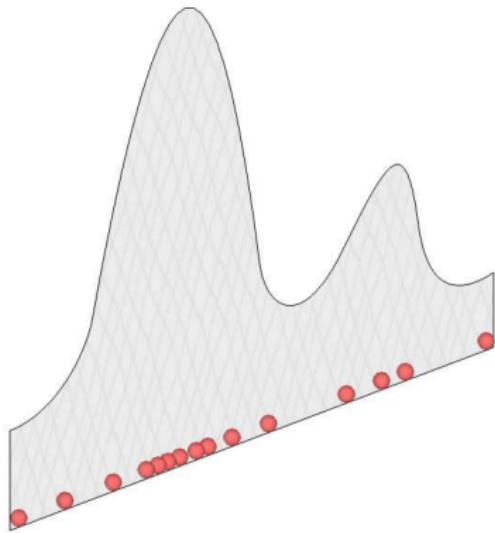


Predictive densities and evolution of the particle ancestry tree



Predictive densities and evolution of the particle ancestry tree

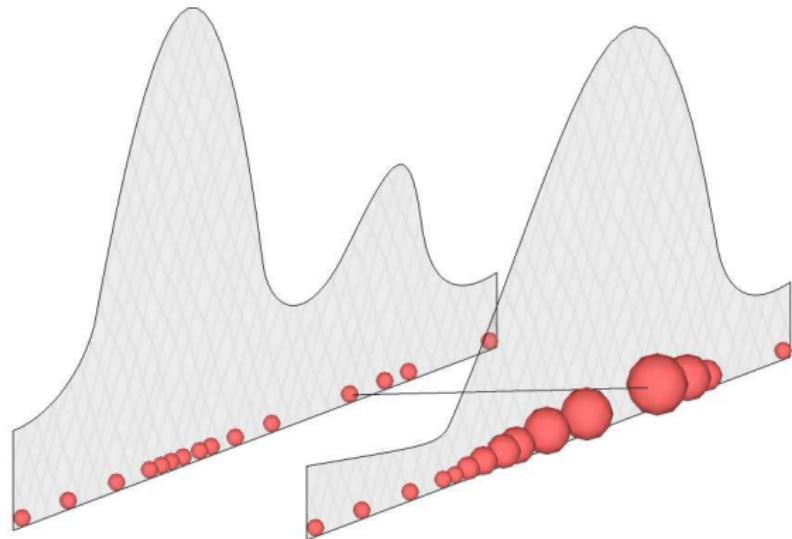
The Standard Bootstrap Filter Viewed as a Path Space Method



i

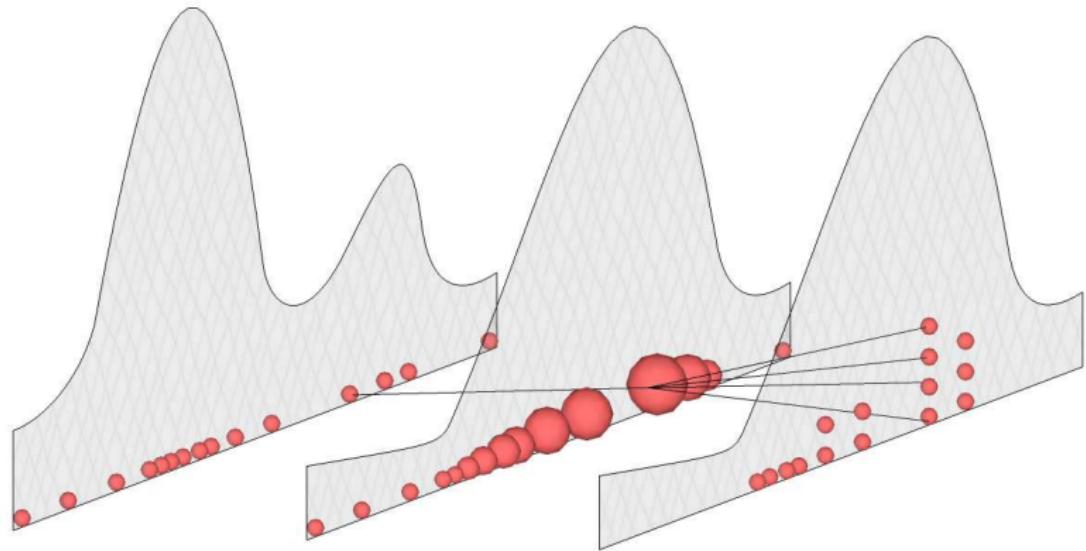
Particles at time $k + 1$ are sampled according to $\xi_{k+1}^i \sim q(\xi_s^{J_k^i}, .)$
where $\xi_k^{J_k^i}$ are the resampled particles at time k .

The Standard Bootstrap Filter Viewed as a Path Space Method



Weights, $\omega_{k+1}^i = g(\xi_{k+1}^i, Y_{k+1})$ are computed.

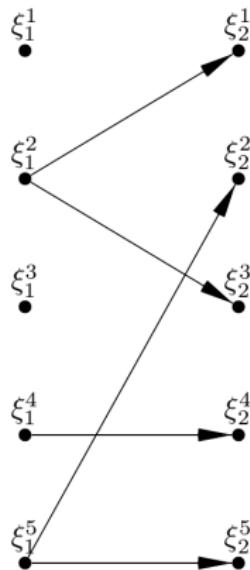
The Standard Bootstrap Filter Viewed as a Path Space Method



Particles are resampled by choosing J_{k+1}^i with probability proportional to ω_{k+1}^i and $\xi_{k+1}^{J_{k+1}^i}$ are used for time $k + 2$.

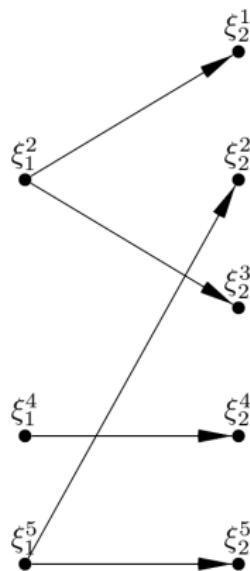
The Path Construction

Path degeneracy



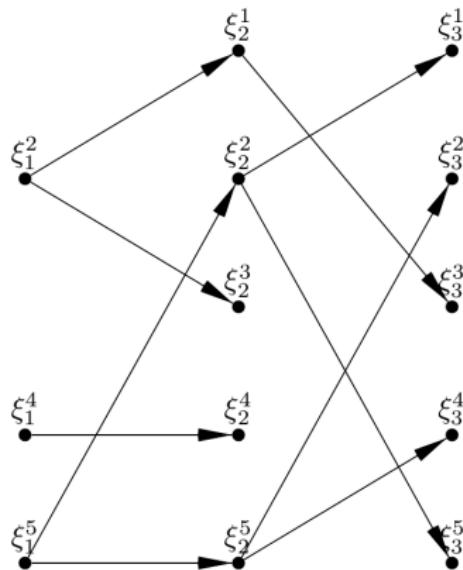
The Path Construction

Path degeneracy



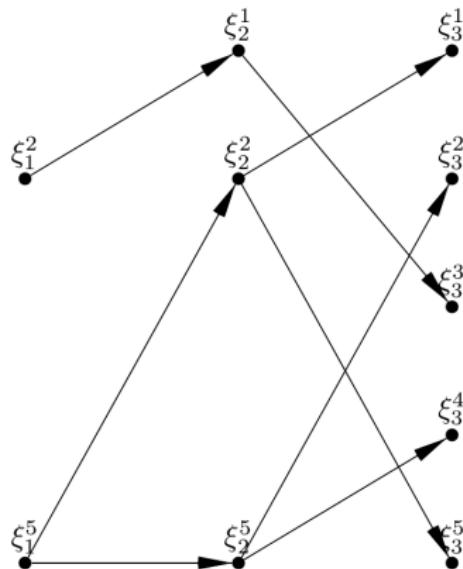
The Path Construction

Path degeneracy



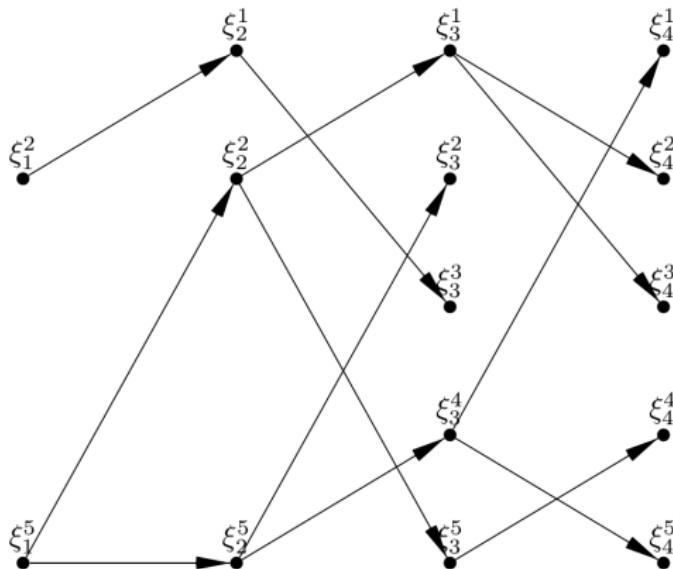
The Path Construction

Path degeneracy



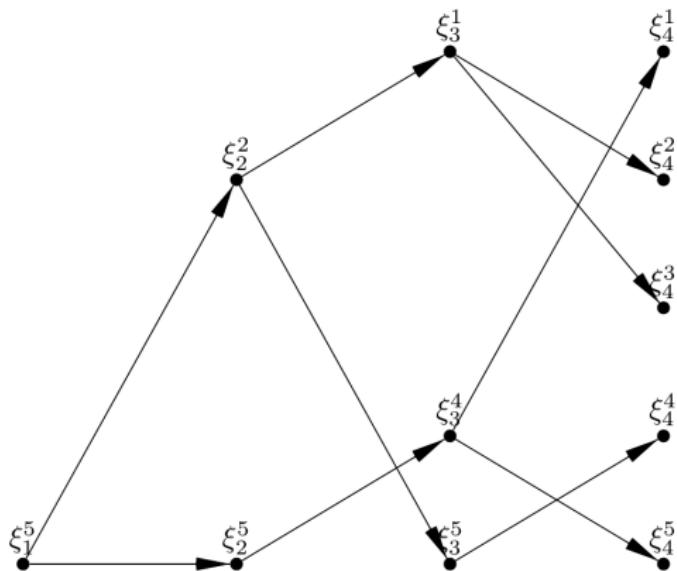
The Path Construction

Path degeneracy



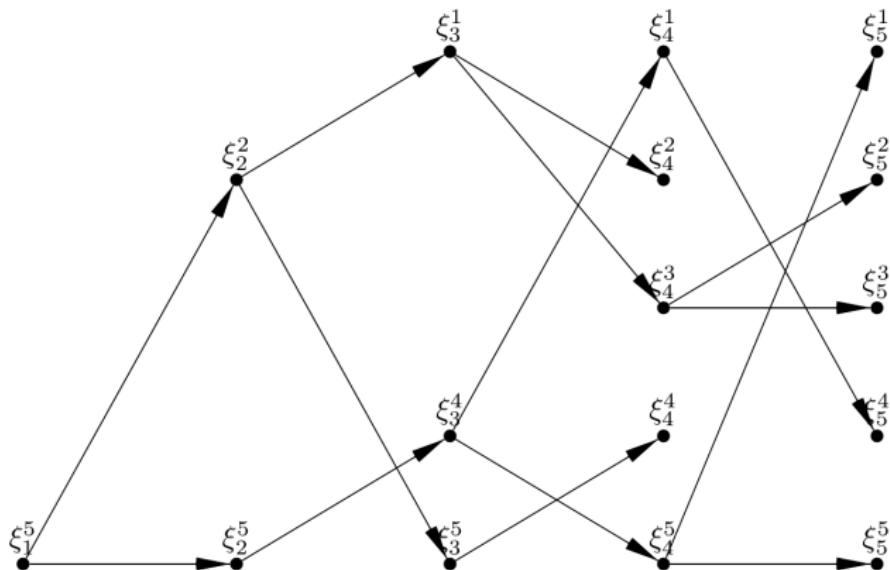
The Path Construction

Path degeneracy



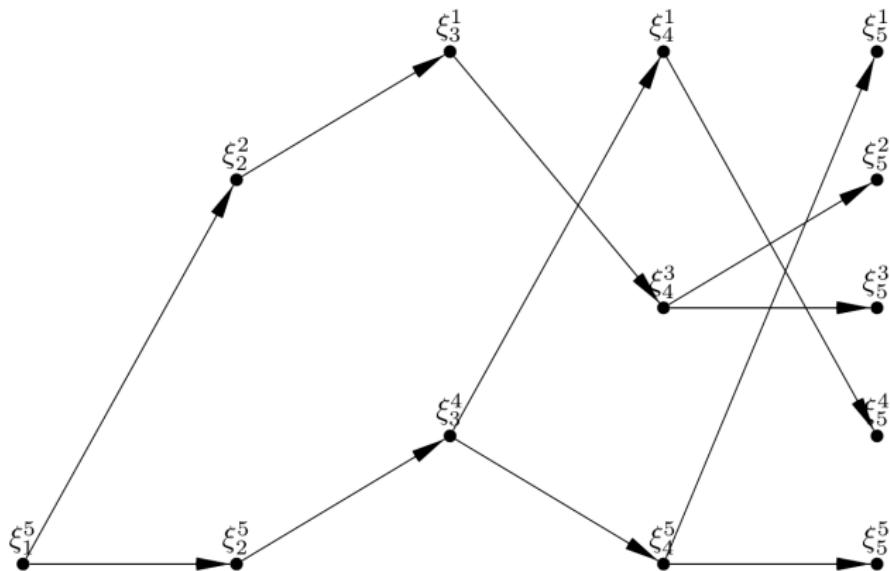
The Path Construction

Path degeneracy



The Path Construction

Path degeneracy



The Backward and the Forward Markov chains

- Conditionally on the observations $Y_{0:n}$, the state sequence $\{X_k\}_{k \geq 0}$ is a time-inhomogeneous Markov chain:

$$\mathrm{E} [f(X_k) | X_{0:k-1}, Y_{0:n}] = \mathrm{E} [f(X_k) | X_{k-1}, Y_{k:n}] .$$

- This property remains true in the **time-reversed** direction.

$$\mathrm{E} [f(X_k) | X_{k+1:n}, Y_{0:n}] = \mathrm{E} [f(X_k) | X_{k+1}, Y_{0:n}] .$$

- Given the final time index n and an initial distribution ν , the associated Markov kernel is the so-called **backward kernel** $\mathrm{B}_{\nu,k}(x_{k+1}, \cdot)$ defined as

$$\mathrm{b}_{\nu,k}(x_{k+1}, x) \stackrel{\text{def}}{=} \frac{\phi_{\nu,k|k}(x) q(x, x_{k+1})}{\int \phi_{\nu,k|k}(x) q(x, x_{k+1}) dx} .$$

Backward Smoothing: joint smoothing distribution

- Backward smoothing recursions can be thought of as the natural extension of the Rauch-Tung-Striebel backward smoothing recursions to nonlinear and non-Gaussian state space models.
- Idea: propagate the **smoothing distribution** backward in time.

$$\phi_{\nu,k:n|n}(x_{k:n}) = b_{\nu,k}(x_{k+1}, x_k) \phi_{\nu,k+1:n|n}(x_{k+1:n}) .$$

Backward Smoothing: marginal distribution

- The marginal smoothing distribution is obtained by marginalizing the joint smoothing distribution with respect to the state sequence.

$$\begin{aligned}\phi_{\nu,k|n}(x_k) &= \phi_{\nu,k|k}(x_k) \int \frac{\phi_{\nu,k+1|n}(x_{k+1})q(x_k, x_{k+1})dx_{k+1}}{\iint \phi_{\nu,k|k}(x_k)q(x_k, x_{k+1})dx_{k:k+1}} \\ &= \int \phi_{\nu,k+1|n}(x_{k+1})b_{\nu,k}(x_{k+1}, x_k)dx_{k+1}.\end{aligned}$$

- Gaussian Linear state space: Rauch-Tung-Striebel smoothing (the simplest but not necessarily the most efficient smoother).

The Forward Filtering Backward Smoothing Algorithm

- The Forward Filtering Backward Smoothing (FFBS) is a two-pass algorithm:
- **Forward pass:** The particle filter is executed, while storing the weighted samples $\{(\xi_k^i, \omega_k^i)\}_{i=1}^N$, approximating the filtering distributions $\phi_{\nu,k|k}$, $k = 0, \dots, n$. (all the particles and weights are kept!);
- **Backward pass:** Starting with the particle approximation of the filtering distribution $\phi_{\nu,k|k}$, $k = 0, \dots, n$, the importance weights are recursively updated backward in time by combining
 - 1 the particle position and weight of the fixed interval smoothing distribution $\phi_{\nu,k+1:n|n}$
 - 2 the particle position and weight of the filtering estimate $\phi_{\nu,k|k}$.
- Only the weights are modified, but the particle positions are kept unchanged.

The FFBS Algorithm for the Joint Smoothing Distribution

- For $1 \leq k \leq l \leq n$, define $\xi_{k:l}^{i_{k:l}} \stackrel{\text{def}}{=} (\xi_k^{i_k}, \dots, \xi_n^{i_n})$, the set of all possible particle paths
- The **FFBS** approximation of the joint smoothing distribution is

$$\hat{\phi}_{\nu,k:n|n}(dx_{k:n}) \propto \sum_{j_{k:n}=1}^N \omega_{k|n}^{j_{k:n}} \delta_{\xi_{k:n}^{j_{k:n}}}(dx_{k:n}),$$

- The support of the joint smoothing distribution is the set of all possible paths... this is why the depletion is expected to be less annoying !
- still, the support of the smoothing distribution is selected in the forward pass... this may cause a problem when the filtering and the smoothing distributions do not agree.

The FFBS Algorithm for the Joint Smoothing Distribution

- **Beware:** The number of such paths grows exponentially with the horizon $n - k$... of course, this is not implementable except if one is interested in estimating a **fixed** marginal smoothing distribution.
- The theory is however more transparent (and in fact the proofs essentially requires) to proceed with the joint distribution...
- A **practical** algorithm for the joint smoothing distribution will be derived later...

The Forward Filtering Backward Smoothing Algorithm

- Consider the approximation of the joint smoothing distribution:

$$\hat{\phi}_{\nu,k+1:n|n}(\mathrm{d}x_{k+1:n}) \propto \sum_{j_{k+1:n}=1}^N \omega_{k+1|n}^{j_{k+1:n}} \delta_{\xi_{k+1:n}^{j_{k+1:n}}}(\mathrm{d}x_{k+1:n}),$$

- Approximate the **backward smoothing kernel** by

$$\hat{B}_{\nu,k}(x_{k+1}, \mathrm{d}x_k) = \sum_{i=1}^N \frac{\omega_k^i q(\xi_k^i, x_{k+1})}{\sum_{\ell=1}^N \omega_k^\ell q(\xi_k^\ell, x_{k+1})} \delta_{\xi_k^i}(\mathrm{d}x_k)$$

The Forward Filtering Backward Smoothing Algorithm

- Substituting the **particle approximations** of the **backward kernel** and the **joint smoothing distribution** into the backward recursion yields to

$$\phi_{\nu,k:n|n}(f) = \int \cdots \int f(x_{k:n}) B_{\nu,k}(x_{k+1}, dx_k) \phi_{\nu,k+1:n|n}(dx_{k+1:n})$$

yields to the updating rule for the smoothing weights :

$$\omega_{k|n}^{j_{k:n}} = \frac{\omega_k^{j_k}(\xi_k^{j_k}, \xi_{k+1}^{j_{k+1}})}{\sum_{\ell=1}^N \omega_k^\ell q(\xi_k^\ell, \xi_{k+1}^{j_{k+1}})} \omega_{k+1|n}^{j_{k+1:n}}.$$

The FFBS as an importance sampling estimator

- The support of the joint smoothing estimators is the set of the N^{n-k+1} possible particle paths $\{\xi_{k:n}^{j_{k:n}}\}$.
- The importance weight of these **path particles** is computed as if the path particles $\xi_{k:n}^{j_{k:n}}$ were simulated by drawing forward in time, for $k < m \leq n$, $\xi_m^{j_m}$ in the set $\{\xi_m^i\}_{i=1}^N$ conditionally independently from the distribution

$$\Omega_{m-1}^{-1} \sum_{\ell=1}^N \omega_{m-1}^\ell q(\xi_{m-1}^\ell, \xi_m^i) , \quad i = 1, \dots, N ,$$

which approximates the predictive distribution $\phi_{\nu, m|m-1}$.

The FFBS for the marginal smoothing distribution

- Substituting the **particle approximations** of the **backward kernel** and the **marginal smoothing distribution** into the backward recursion yields to

$$\phi_{\nu,k|n}(f) = \int \cdots \int f(x_{k:}) B_{\nu,s}(x_{k+1}, dx_k) \phi_{\nu,k+1:n|n}(dx_{k+1}),$$

yields to the updating rule for the smoothing weights :

$$\omega_{k|n}^i = \sum_{j=1}^N \frac{\omega_k^i q(\xi_k^i, \xi_{k+1}^j)}{\sum_{\ell=1}^N \omega_k^\ell q(\xi_k^\ell, \xi_{k+1}^j)} \omega_{k+1|n}^j, \quad i = 1, \dots, N \quad (1)$$

The FFBS for the marginal smoothing distribution

- The complexity of this estimator of the marginal smoothing distribution is $O(N^2n)$, which is manageable only if the number of particles is moderate.
- When the dimension of the input space is not too large, this computational cost can be reduced to $N \log(N)$, but at the price of truncating the distribution and therefore introducing some bias.

The Backward Markov Kernel

- We define a probability distribution over the set of trajectories (indexed by $\{1, \dots, N\}^{n-k}$), which may be related to a discrete state space non homogeneous Markov chain over $\{1, \dots, N\}$.
- For $k \in \{0, \dots, n-1\}$, consider the Markov transition matrix $(W_k^{i,j})_{i,j=1}^N$ from $\{1, \dots, N\}$ to $\{1, \dots, N\}$

$$W_k^{i,j} = \frac{\omega_k^j q(\xi_k^j, \xi_{k+1}^i)}{\sum_{\ell=1}^N \omega_k^\ell q(\xi_k^\ell, \xi_{k+1}^i)}, \quad i, j = 1, \dots, N.$$

Backward Simulation

- Define the non-homogeneous backward Markov chain $\{J_u\}_{u=0}^n$ constructed recursively as follows.
 - 1 At time n , the random index J_n is drawn in $\{1, \dots, N\}$ with a probability proportional to $[\omega_n^1, \dots, \omega_n^N]$.
 - 2 At time $k \leq n - 1$, assuming that the vector of indices $J_{k+1:n}$ has been simulated, we draw the index J_k in the set $\{1, \dots, N\}$ with a probability $[W_k^{1,J_{k+1}}, \dots, W_k^{N,J_{k+1}}]$.
- More formally, the joint distribution of the vector of index $J_{k:n}$ is given by

$$\begin{aligned} P(J_{k:n} = i_{k:n} | \mathcal{F}_n) &= W_k^{i_k, i_{k+1}} \dots W_n^{i_{n-1}, i_n} \frac{\omega_n^{i_n}}{\Omega_n} \\ &= \varpi_{k:n}^{i_{k:n}} \frac{\omega_n^{i_n}}{\Omega_n}, \quad i_{k:n} \in \{1, \dots, N\}^{n-k+1}. \end{aligned}$$

FFBS vs FBBSi

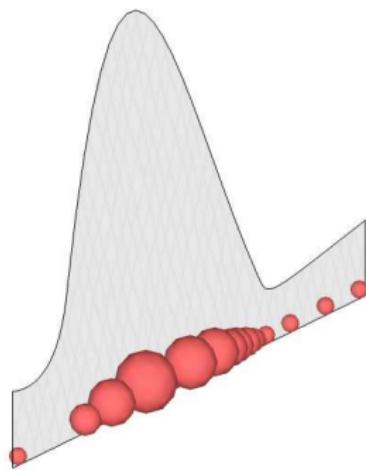
- The FFBS estimator of the joint smoothing distribution may be written as the conditional expectation

$$\hat{\phi}_{\nu,k:n|n}(f) = \mathbb{E} \left[f \left(\xi_{k:n}^{J_{k:n}} \right) \middle| \mathcal{F}_n \right].$$

- We may therefore construct an unbiased estimator of the FFBS estimator by drawing, conditionally independently from \mathcal{F}_n , N paths of $\{J_{k:n}^\ell\}_{\ell=1}^N$ of the non-homogeneous Markov chain introduced above.
- This sample yields to the following (practical) estimator of the joint fixed-interval smoothing distribution:

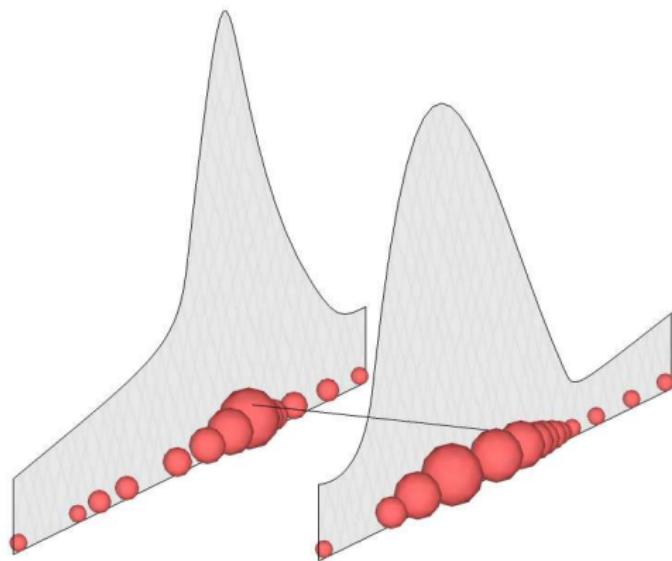
$$\hat{\phi}_{\nu,k:n|n}(f) = N^{-1} \sum_{\ell=1}^N f \left(\xi_{k:n}^{J_{k:n}^\ell} \right).$$

The FFBSi Method



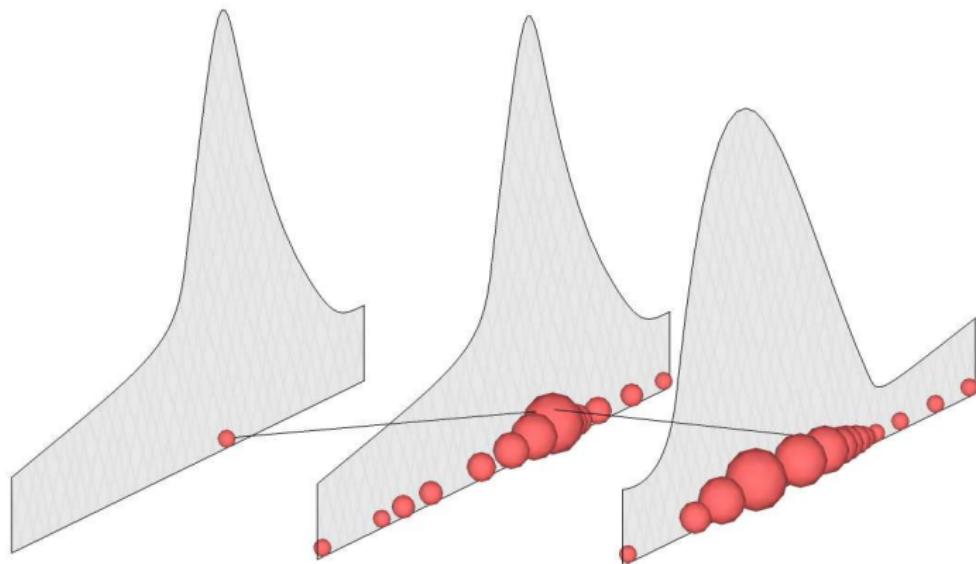
Particles $(\xi_k^j)_j$ and their weights $(\omega_k^j)_j$ at time k are drawn in the forward pass using the bootstrap filter.

The FFBSi Method



To sample the ascendant of the particle $\xi_{k+1}^{J_{k+1}^i}$, the weights of particles $(\xi_k^j)_j$ are recomputed by $(\omega_k^j q(\xi_k^j, \xi_{k+1}^{J_{k+1}^i}))_j$.

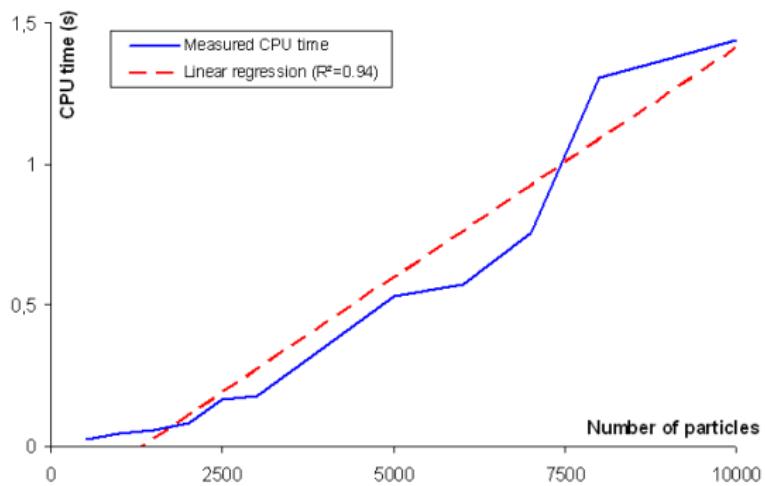
The FFBSi Method



The ascendant $\xi_k^{J_k^i}$ of the particle $\xi_{k+1}^{J_{k+1}^i}$ is sampled among the $(\xi_k^j)_j$ according to the previous weights.

FFBSi method: Complexity

- Drawing paths backward in time by computing all the weights leads to a $\mathcal{O}(N^2n)$ complexity.
- However an accept-reject procedure ensures a $\mathcal{O}(Nn)$ complexity as shown in the figure below.



Roadmap

15 Smoothing

- The Forward Filtering Backward Smoothing Algorithm
- The Forward Filtering Backward Simulation Algorithm

16 Smoothing Algorithms

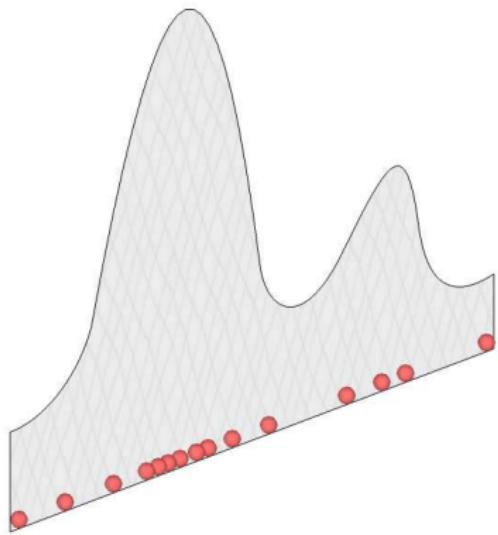
- Path space method
- FFBS method
- FFBSi method
- Simulations

Path space method: Bootstrap filter

Objective : Compute $\mathbb{E} \left[\sum_{s=1}^T h(X_{s-1}, X_s) \middle| Y_{0:T} \right]$

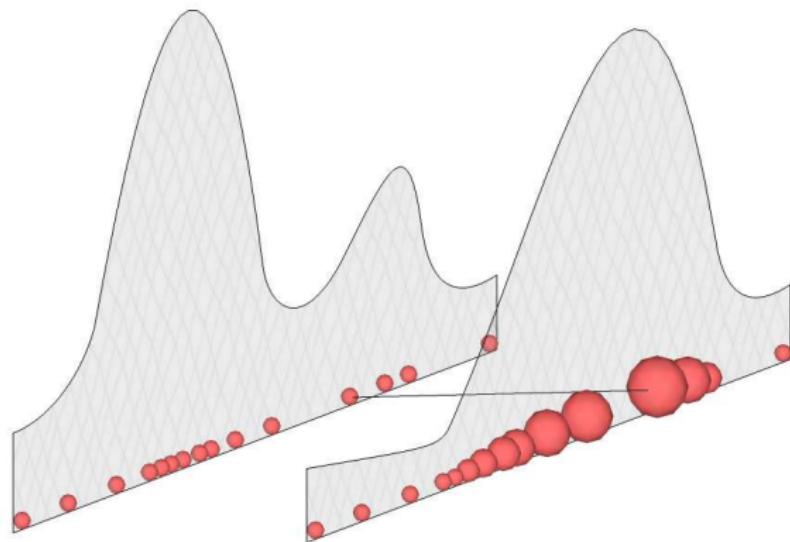
- The first idea to compute smoothed additive functionals is to use **interacting particle systems**.
- These weighted samples approximate recursively the sequence of **posterior distributions** of $X_{0:s}|Y_{0:s}$ using a combination of selection/mutation steps at each time index.

Path space method: Bootstrap filter



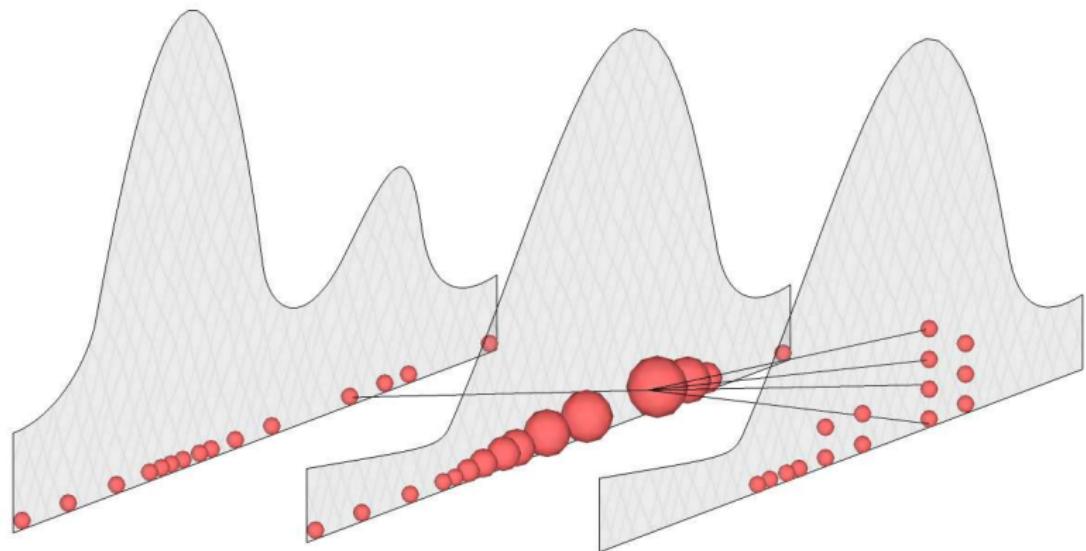
Particles at time $s + 1$ are sampled according to $\xi_{s+1}^i \sim Q(\xi_s^{I_s^i}, .)$
where $\xi_s^{I_s^i}$ are the resampled particles at time s .

Path space method: Bootstrap filter



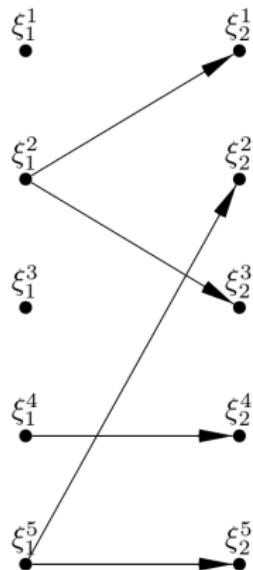
Weights, $\omega_{s+1}^i = g(\xi_{s+1}^i, Y_{s+1})$, are computed to match the target kernel proportional to $Q(\xi_s^{I_s^i}, \cdot)g(\cdot, Y_{s+1})$.

Path space method: Bootstrap filter

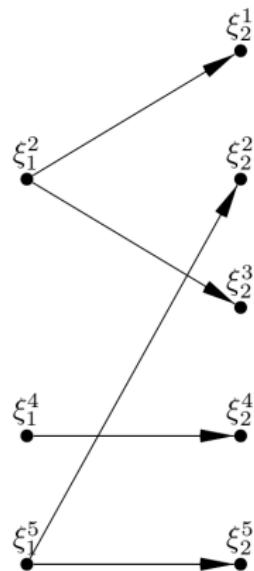


Particles are resampled by choosing I_{s+1}^i with probability proportional to ω_{s+1}^i and $\xi_{s+1}^{I_{s+1}^i}$ are used for time $s + 2$.

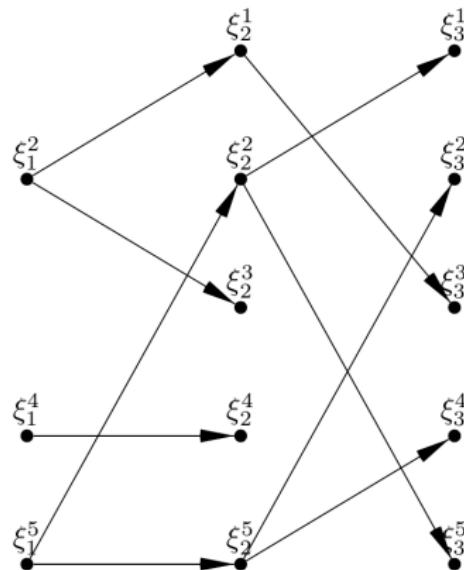
Path degeneracy



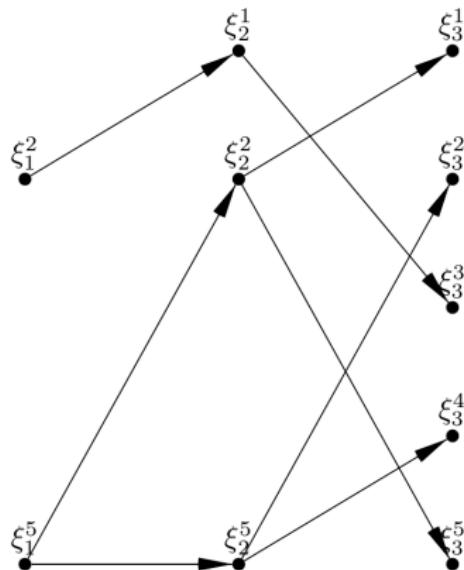
Path degeneracy



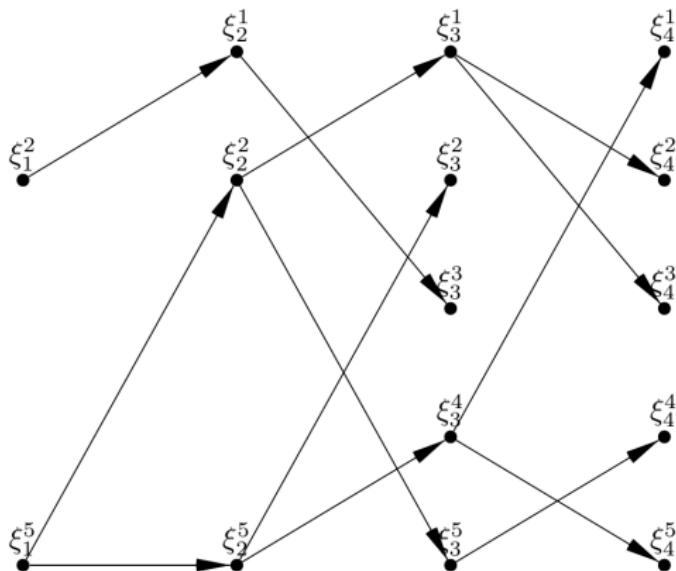
Path degeneracy



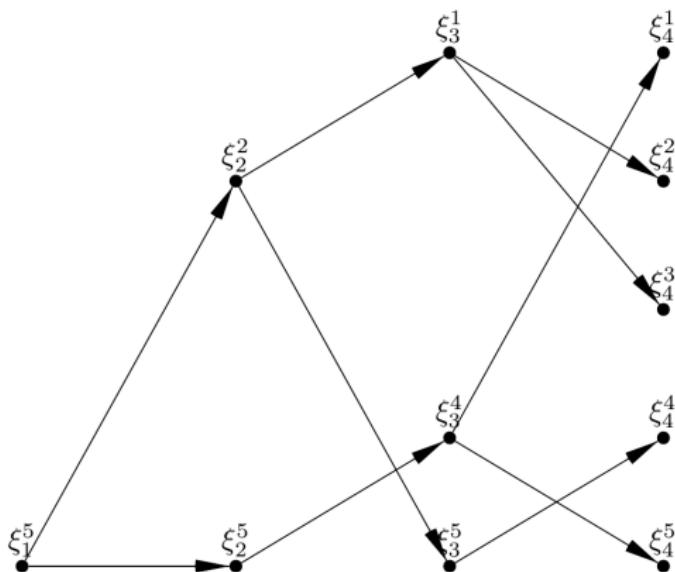
Path degeneracy



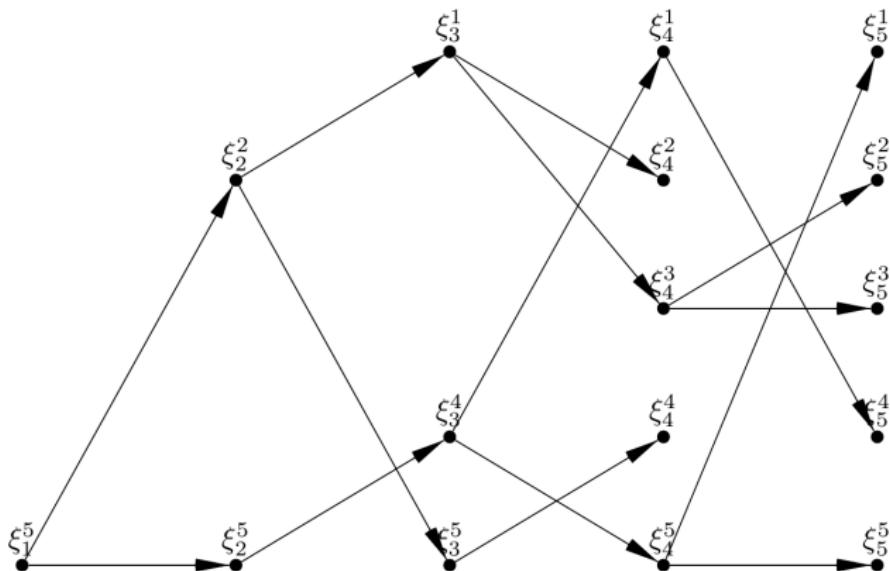
Path degeneracy



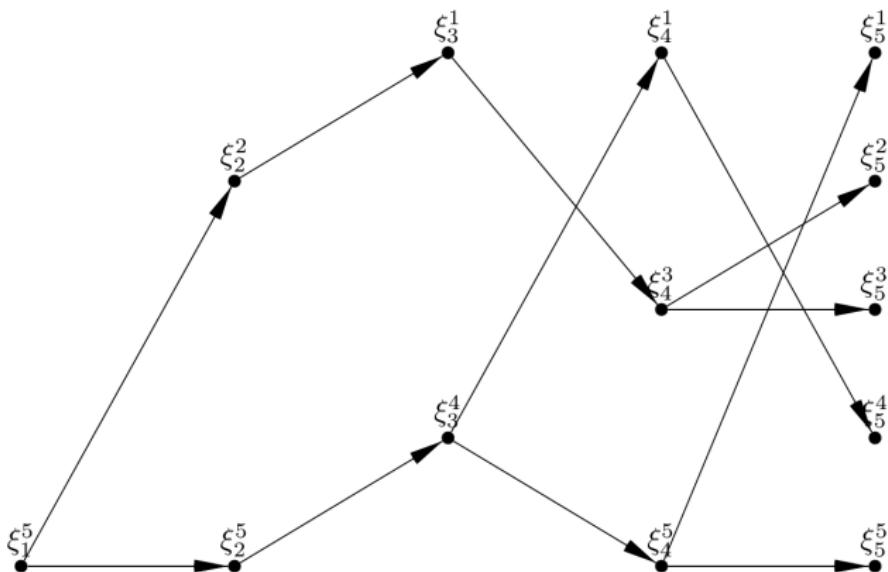
Path degeneracy



Path degeneracy



Path degeneracy



FFBS method

- The **Forward Filtering Backward Smoothing** algorithm can be used to compute approximations of joint smoothing distributions.
- The first step is to perform a **forward pass**, drawing particles (ξ_s^i) and their associated weights (ω_s^i) using the bootstrap filter.
- Then, introducing, for any probability distribution η ,

$$B_\eta(x, h) = \frac{\int \eta(dx') q(x', x) h(x')}{\int \eta(dx') q(x', x)}$$

the **joint smoothing distribution** may be computed recursively according to

$$\pi_{s:T|T}(h) = \int B_{\pi_s}(x, h)(x_{s+1}, dx_s) \pi_{s+1:T|T}(dx_{s+1:T}) h(x_{s+1:T}) .$$

FFBS method

The approximation of the **joint smoothing distribution** may be computed recursively backward as follows

- At time T the approximation is given by $\pi_{T|T}^N = \pi_T^N$.
- For any $s < T$, the filtering distribution is replaced by its **particle approximation** in the exact recursion.

$$B_{\pi_s^N}(x, h) = \sum_{i=1}^N \frac{\omega_s^i q(\xi_s^i, x)}{\sum_{\ell=1}^N \omega_s^\ell q(\xi_s^\ell, x)} h(\xi_s^i)$$

- This leads to

$$\pi_{s:T|T}^N(h) = \sum_{i_{s:T}=1}^N \left(\prod_{u=s+1}^T \frac{\omega_{u-1}^{i_{u-1}} q(\xi_{u-1}^{i_{u-1}}, \xi_u^{i_u})}{\sum_{\ell=1}^N \omega_{u-1}^\ell q(\xi_{u-1}^\ell, \xi_u^{i_u})} \right) \frac{\omega_T^{i_T}}{\Omega_T} h(\xi_s^{i_s}, \dots, \xi_T^{i_T}) .$$

FFBS method

- The estimator is impractical since the cardinality of its support **grows exponentially** with $T - s$.
- Results on this algorithm will be used to compute bounds for the **FFBSi** algorithm.
- This more practical approximation relies on the particular form of the **normalized smoothing** weights.

FFBSi method

- To overcome the path degeneracy issue the **Forward Filtering Backward Simulation** algorithm can also be used.
- The first step is to perform a **forward pass**, drawing particles (ξ_s^i) and their associated weights (ω_s^i) using the bootstrap filter.
- Then N paths of particle indices $(J_{0:T}^i)_{i \in \{1, \dots, N\}}$ are **independently drawn backward** from the set $\{1, \dots, N\}^{T+1}$.
 - 1 $(J_T^i)_i$ are chosen with probability proportional to $(\omega_T^j)_j$.
 - 2 $(J_s^i)_i$ are chosen with probability proportional to $(\omega_s^j q(\xi_s^j, \xi_{s+1}^{J_s^i}))_j$.

More on FFBSi

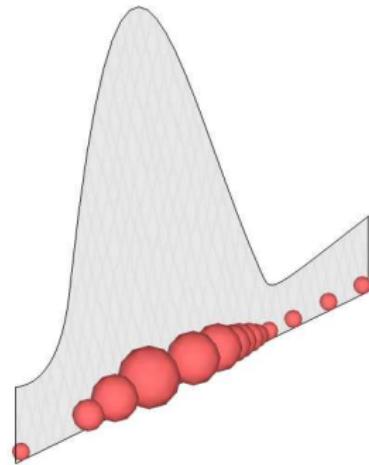
- We use the **backward kernel** given by :

$$B_{\phi_s}(x, dx') \stackrel{\text{def}}{=} \frac{q(x', x)\phi_s(dx')}{\int_X q(x', x)\phi_s(dx')}$$

- Starting with $\phi_T^N = \phi_{T:T|T}^N$, we obtain for any $0 \leq s \leq T - 1$ the joint smoothing distribution $\phi_{s:T|T}^N$ according to the following update formula :

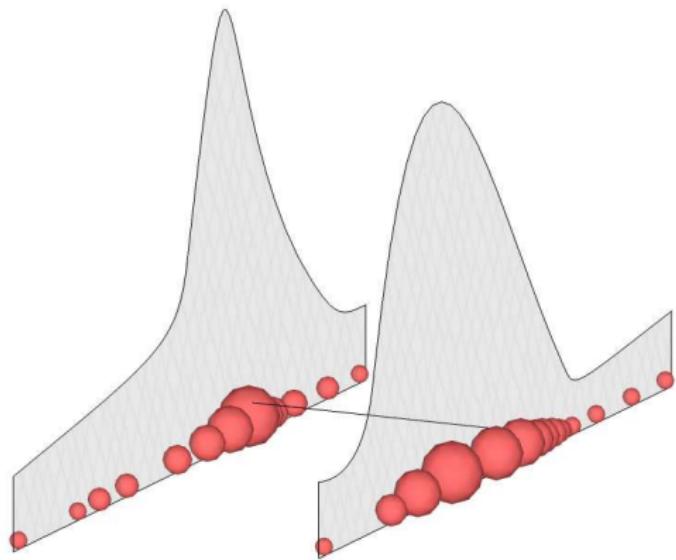
$$\phi_{s:T|T}^N(dx_{s:T}) \stackrel{\text{def}}{=} B_{\phi_s^N}(x_{s+1}, dx_s) \phi_{s+1:T|T}^N(dx_{s+1:T})$$

FFBSi method



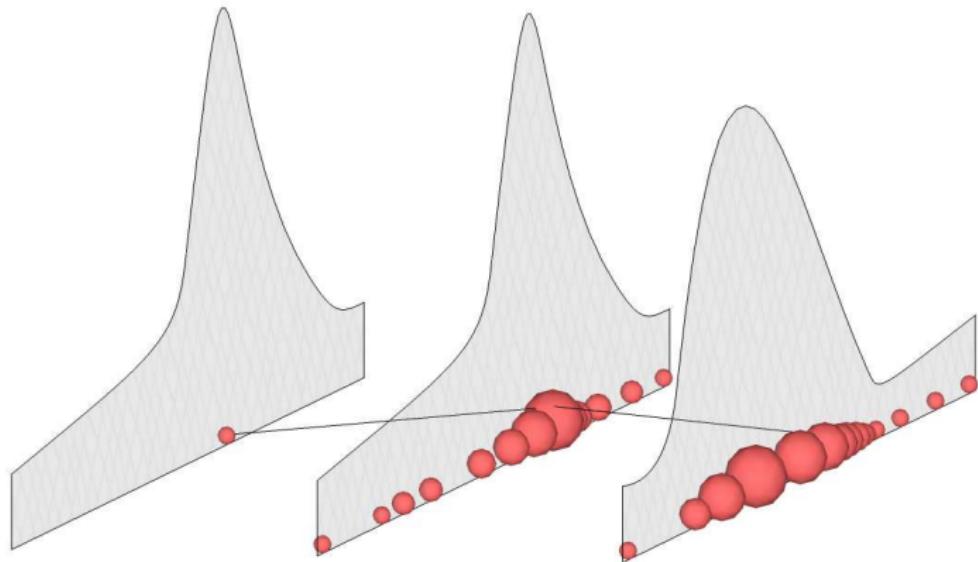
Particles $(\xi_s^j)_j$ and their weights $(\omega_s^j)_j$ at time s are drawn in the forward pass using the bootstrap filter.

FFBSi method



To sample the ascendant of the particle $\xi_{s+1}^{J_{s+1}^i}$, the weights of particles $(\xi_s^j)_j$ are recomputed by $(\omega_s^j q(\xi_s^j, \xi_{s+1}^{J_{s+1}^i}))_j$.

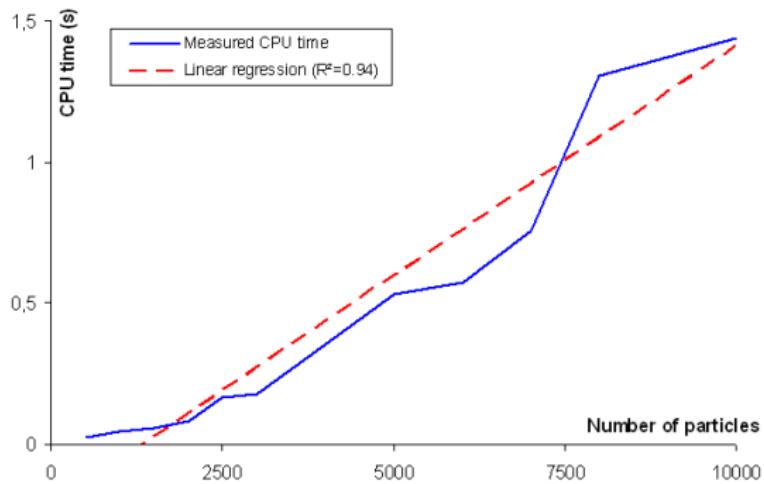
FFBSi method



The ascendant $\xi_s^{J_s^i}$ of the particle $\xi_{s+1}^{J_{s+1}^i}$ is sampled among the $(\xi_s^j)_j$ according to the previous weights.

FFBSi method: Complexity

- Drawing paths backward in time by computing all the weights leads to a $\mathcal{O}(N^2T)$ complexity.
- However an accept-reject procedure ensures a $\mathcal{O}(NT)$ complexity as shown in the figure below.



Parameter estimation

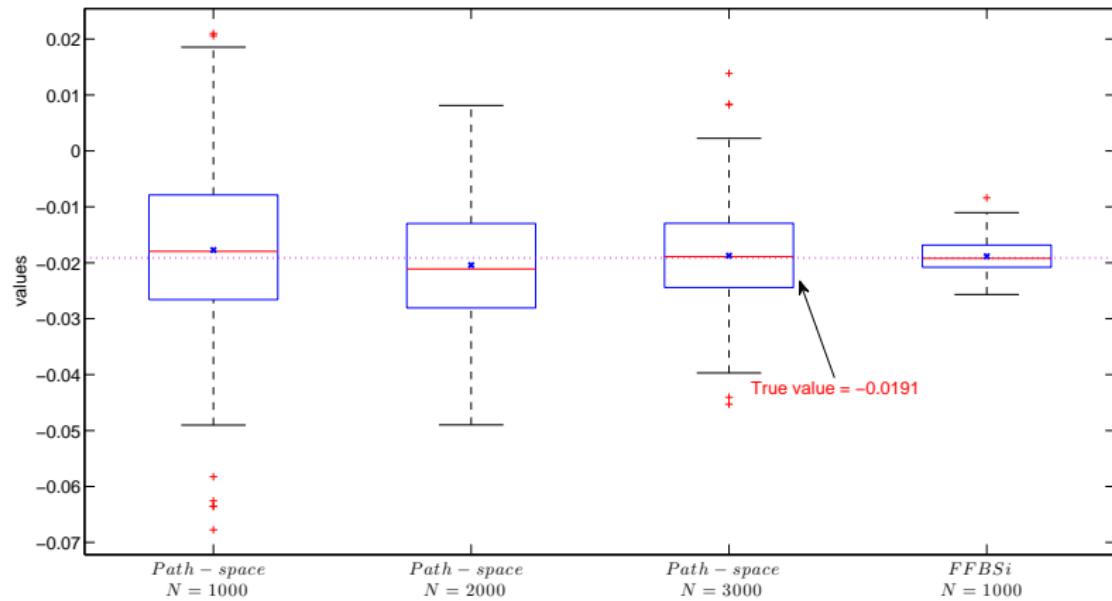
1 Linear gaussian model

- $X_{t+1} = \phi X_t + \sigma_u U_{t+1}$
- $Y_t = X_t + \sigma_v V_t$

U_t and V_t are independent standard gaussian random variables.

The three parameters $(\phi, \sigma_u, \sigma_v)$ are known and both algorithms are used to compute $\frac{1}{T} \sum_{s=1}^T \mathbb{E}[X_s | Y_{1:T}]$. Estimates are compared to the true value given by the Kalman smoother.

Smoothed additive functional estimate with $T = 1000$



Parameter estimation

2 Stochastic volatility model

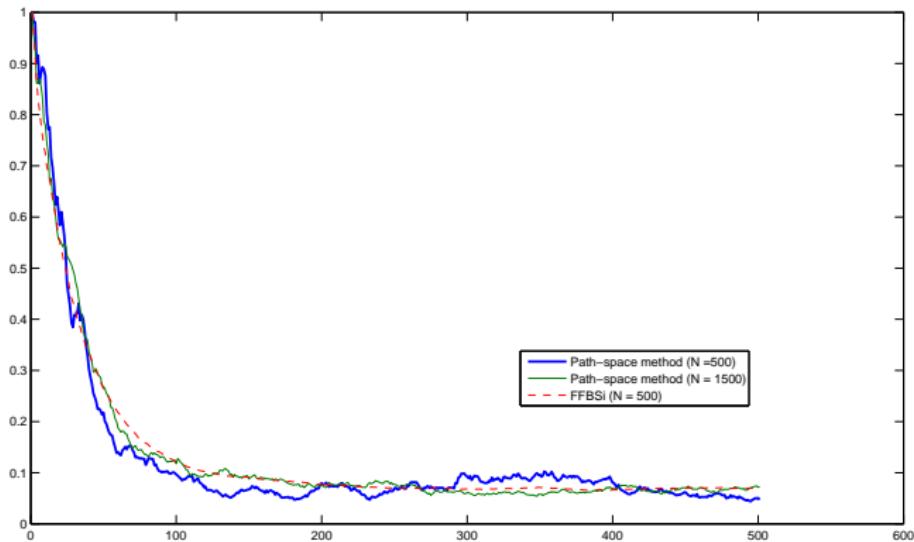
- $X_{t+1} = \phi X_t + \sigma U_{t+1}$
- $Y_t = \beta e^{\frac{X_t}{2}} V_t$

U_t and V_t are independent standard gaussian random variables.

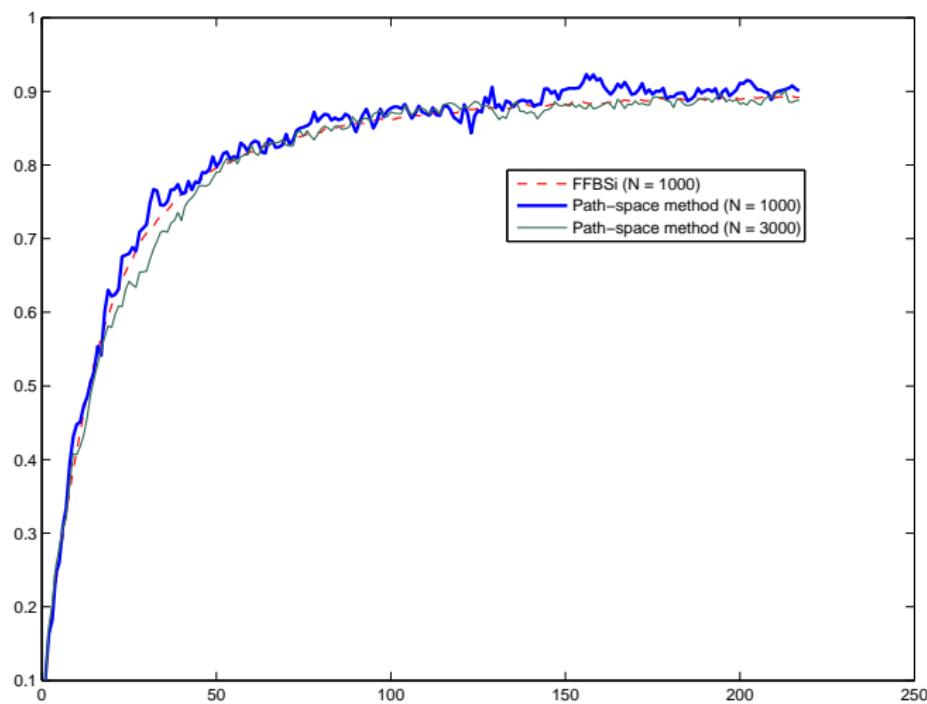
The three parameters (ϕ, σ, β) are estimated using the path space-method and the FFBSi algorithm as an approximation of the expectation step of the EM algorithm.

Observations are generated using $(\phi, \sigma^2, \beta^2) = (0.9, 0.1, 1)$ and each algorithm is initialized with $(\phi_0, \sigma_0^2, \beta_0^2) = (0.1, 1, 2)$.

Estimation of σ^2 with time horizon $T = 500$



Estimation of ϕ with time horizon $T = 1000$



What do we know about FFBS' stability ?

Recent results on smoothed additive functionals :

- 2009 - *Sequential Monte Carlo smoothing for general state space hidden Markov models*: R. Douc and al. used an innovating version of the forward filter backward simulation algorithm of complexity $\mathcal{O}(TN)$. The error L_q -norm is bounded by a function of T^2/N .
- 2009 - *A backward particle interpretation of Feynman-Kac Formulae*: P. Del Moral and al. obtained estimates with a **forward only** algorithm of complexity $\mathcal{O}(TN^2)$. The error L_q -norm is bounded by a function of T^2/N .
- 2009 - *Forward smoothing using sequential monte Carlo method*, P. Del Moral and al. derive a **$\mathcal{O}(TN^2)$ forward only** algorithm to perform this computation whose quadratic error can be controlled by terms of the form $(\frac{T}{N})^\ell$, $\ell \in \{1, \frac{3}{2}, 2\}$.

Part VII

Concluding Remarks

- 1 Sequential state estimation is of fundamental importance in the study of dynamical systems where the state is hidden and the only thing that we have is a set of observables dependent on the state.

- 1 Sequential state estimation is of fundamental importance in the study of dynamical systems where the state is hidden and the only thing that we have is a set of observables dependent on the state.
- 2 Depending on the system description, we may identify the following system hierarchy of increasing mathematical difficulty:

- 1 Sequential state estimation is of fundamental importance in the study of dynamical systems where the state is hidden and the only thing that we have is a set of observables dependent on the state.
- 2 Depending on the system description, we may identify the following system hierarchy of increasing mathematical difficulty:
 - 1 Linear, Gaussian

- 1 Sequential state estimation is of fundamental importance in the study of dynamical systems where the state is hidden and the only thing that we have is a set of observables dependent on the state.
- 2 Depending on the system description, we may identify the following system hierarchy of increasing mathematical difficulty:
 - 1 Linear, Gaussian
 - 2 Nonlinear, Gaussian

- 1 Sequential state estimation is of fundamental importance in the study of dynamical systems where the state is hidden and the only thing that we have is a set of observables dependent on the state.
- 2 Depending on the system description, we may identify the following system hierarchy of increasing mathematical difficulty:
 - 1 Linear, Gaussian
 - 2 Nonlinear, Gaussian
 - 3 Linear, non-Gaussian

- 1 Sequential state estimation is of fundamental importance in the study of dynamical systems where the state is hidden and the only thing that we have is a set of observables dependent on the state.
- 2 Depending on the system description, we may identify the following system hierarchy of increasing mathematical difficulty:
 - 1 Linear, Gaussian
 - 2 Nonlinear, Gaussian
 - 3 Linear, non-Gaussian
 - 4 Nonlinear, non-Gaussian

- 1 The Kalman filter provides an optimal solution (optimal in the Bayesian sense) to the linear, Gaussian case in a highly elegant and rigorous manner.

- 1 The Kalman filter provides an optimal solution (optimal in the Bayesian sense) to the linear, Gaussian case in a highly elegant and rigorous manner.
- 2 For nonlinear systems with Gaussian distributions, we have two options, depending on the degree of nonlinearity:
 - 1 Extended Kalman filter for nonlinearities of a mild sort, where first-order approximations to the process and measurement equations are adequate.
 - 2 Unscented Kalman filter, when the first-order approximations are violated.

- 1 Particle filters, rooted in Monte Carlo simulation and Bayesian estimation, provide a practical solution to the more general case of nonlinear, non-Gaussian systems.

- 1 Particle filters, rooted in Monte Carlo simulation and Bayesian estimation, provide a practical solution to the more general case of nonlinear, non-Gaussian systems.
- 2 However, this relatively new family of sequential-state estimation tools are more difficult to implement and control than the Kalman Filter.

- 1 Particle filters, rooted in Monte Carlo simulation and Bayesian estimation, provide a practical solution to the more general case of nonlinear, non-Gaussian systems.
- 2 However, this relatively new family of sequential-state estimation tools are more difficult to implement and control than the Kalman Filter.
- 3 Particle filters rely on importance samples drawn from an instrumental or proposal distribution.

- 1 Particle filters, rooted in Monte Carlo simulation and Bayesian estimation, provide a practical solution to the more general case of nonlinear, non-Gaussian systems.
- 2 However, this relatively new family of sequential-state estimation tools are more difficult to implement and control than the Kalman Filter.
- 3 Particle filters rely on importance samples drawn from an instrumental or proposal distribution.
- 4 A key to a successful design of particle filters is the choice of the instrumental or proposal distribution.

Some General References on Particle Filtering

- Doucet, A., De Freitas, N. and Gordon, N. (eds.) (2001), *Sequential Monte Carlo Methods in Practice*, Springer.
- Ristic, B., Arulampalam, M. and Gordon, A. (2004), *Beyond Kalman Filters: Particle Filters for Target Tracking*, Artech House.
- Cappé, O., Moulines, E. and Rydén, T. (2005), *Inference in Hidden Markov Models*, Springer.
- Doucet, A., Godsill, S. and Andrieu, C. (2000), On sequential Monte-Carlo sampling methods for Bayesian filtering, *Stat. Comput.*, **10**:197–208.
- Arulampalam, M., Maskell, S., Gordon, N. and Clapp, T. (2002), A tutorial on particle filters for on line non-linear/non-Gaussian Bayesian tracking, *IEEE Trans. Signal Process.*, **50**, 241–254.
- Cappé, O., Godsill, S. J. and Moulines, E. (2007), An overview of existing methods and recent advances in sequential Monte Carlo, *IEEE Proc.*, **95**:899–924.