Filtering, data assimilation, SDE’s

Hans R. Künsch

Seminar für Statistik, ETH Zürich

Singapore, January 2008
Contents

1. Summary of last week’s lecture
2. State space models
3. Some basic results on SDE’s
Statistical methods to handle uncertainty in differential equation models:

- Least squares estimation of unknown parameters (finite dimensional).
- Bayesian estimation of unknown inputs (infinite dimensional).
- Recursive estimation of state of a system with incomplete and noisy data (filtering, data assimilation).
- Explicit inclusion of a bias term.
- Time-varying parameters.
- Stochastic differential equations (SDE’s): Noise is included in the dynamics.
Take up two topics in some more detail

- Filtering for state space models: Kalman, Ensemble Kalman and Particle Filters.
- SDE’s: Ito formula, Simulation, Girsanov’s theorem, Estimation of parameters.
General state space models

Consist of an unobserved state sequence \((x_t)\) and an observation sequence \((y_t)\) of the following form

\[
x_t = g(x_{t-1}) + U_t,
\]
\[
y_t = h(x_t) + V_t
\]

where the system noises \(U_t\) and observation noises \(V_t\) are all mutually independent.

Graphical representation:

\[
\ldots 
\rightarrow x_{t-1} 
\rightarrow x_t
\rightarrow x_{t+1}
\rightarrow \ldots
\]
\[
\downarrow
\downarrow
\downarrow
\]
\[
\ldots 
\rightarrow y_{t-1}
\rightarrow y_t
\rightarrow y_{t+1}
\rightarrow \ldots
\]
(\(x_t\)) is Markovian, (\(y_t\)) is not.

Usually \(x_t\) is high-dimensional.

Often components of \(x_t\) associated with some spatial location.

State evolution can be deterministic, i.e. \(U_t = 0\).

In a fluid model, \(x_t\) can be the velocity field on a fine grid, \(y_t\) velocity measured on a coarse grid.

Noise need not be additive.

\(g, h\) and noise distributions could depend on \(t\).
Use observations \((y_1, \ldots y_s)\) and knowledge about the dynamics to determine an unobserved state \(x_t\)

\[
\begin{align*}
    s < t & \quad \text{Prediction, Forecasting} \\
    s = t & \quad \text{Filtering, Nowcasting, Analysis} \\
    s > t & \quad \text{Smoothing, Backcasting, Reanalysis.}
\end{align*}
\]

Both available information and unknown variable change with time.
Predictive and filter distributions: Recursions

Describe imperfect knowledge about \(x_t\) by probability distributions.

Predictive distribution \(\mu_t = \text{distribution of } x_t \text{ given } y_1, \ldots, y_{t-1}\).
Filter distribution \(\nu_t = \text{distribution of } x_t \text{ given } y_1, \ldots, y_t\).
\(\nu_0 = \text{initial distribution of } x_0\).

Propagation, Forecast

\[
\mu_t(dx_t) = \int f_U(x_t - g(x_{t-1}))\nu_{t-1}(dx_{t-1}) \, dx_t.
\]

Update, Analysis

\[
\nu_t(dx_t) = \frac{f_V(y_t - h(x_t))\mu_t(dx_t)}{\int f_V(y_t - h(x))\mu_t(dx)}.
\]
Linear Gaussian model: Kalman filter

Exact solution is available if

\[ g(x) = Gx, \ h(x) = Hx, \ U_t \sim \mathcal{N}(0, R), \ V_t \sim \mathcal{N}(0, Q). \]

If \( \nu_0 = \mathcal{N}(0, P^f_0) \), then for all \( t \)

\[ \mu_t = \mathcal{N}(m^p_t, P^p_t), \quad \nu_t = \mathcal{N}(m^f_t, P^f_t). \]

Recursions for means and covariances:

\[ m^p_t = Gm^f_{t-1}, \quad P^p_t = GP^f_{t-1}G^T + R \]

and

\[ m^f_t = m^p_t + K_t(y_t - Hm^p_t), \quad P^f_t = (I - K_tH)P^p_t \]

where \( K_t \) is the gain matrix

\[ K_t = P^p_t H^T(Q + HP^p_t H^T)^{-1}. \]
Monte Carlo methods represent $\mu_t$ and $\nu_t$ by samples (ensembles) $(\tilde{x}_{t,j}; 1 \leq j \leq N)$ and $(x_{t,j}; 1 \leq j \leq N)$ which evolve in time.

**Propagation:** From $(x_{t-1,j})$ to $(\tilde{x}_{t,j})$.

$$\tilde{x}_{t,j} = g(x_{t-1,j}) + U_{t,j}$$

where $U_{t,j}$ are independent replicates of $U_t$.

**Update:** From $(\tilde{x}_{t,j})$ to $(x_{t,j})$. Two methods, particle and ensemble filters.
Particle filter update

If $\mu_t$ is discrete with values $(\tilde{x}_{t,j})$ and weights $1/N$, then by Bayes formula $\nu_t$ is also discrete with the same values, but weights

$$w_{t,j} = \frac{f_V(y_t - h(\tilde{x}_{t,j}))}{\sum_k f_V(y_t - h(\tilde{x}_{t,k}))}.$$

Could propagate a weighted discrete $\nu_t$, but weights degenerate quickly during the iterations. To overcome this, use resampling: Choose $\tilde{x}_{t,j}$ approximately $NW_{t,j}$ times. High weight particles are multiplied, low weight particles die out. See separate figure.

If the evolution of states is stochastic, ties are broken during the next propagation step. Otherwise, need to add a bit of noise.
Consider

\[ x_t = 0.5x_{t-1} + 25 \frac{x_{t-1}}{1 + x_{t-1}^2} + \gamma \cos(1.2t) + U_t, \quad y_t = \frac{x_t^2}{20} + V_t. \]

which goes back to Andrade Netto et al., IEEE Trans. Autom. Control (1978). \( y_t \) carries no information about the sign of \( x_t \).

In another window a simulation from the model and 10% and 90% filter quantiles are shown.
Particle filter is inefficient if weights before resampling vary widely. As in importance sampling, can use a wrong transition density $q(x_t \mid x_{t-1,j})$ in the propagation step and correct with the weights

$$w_{t,j} \propto \frac{f_V(y_t - h(\tilde{x}_{t,j})) f_U(\tilde{x}_{t,j} - g(x_{t-1,j}))}{q(\tilde{x}_{t,j} \mid x_{t-1,j})}.$$

Various ideas for constructing good $q'$s (that depend on $y_t$) exist.
Assume a linear observation function with Gaussian noise:

\[ h(x) = Hx, \quad V \sim \mathcal{N}(0, Q). \]

The Kalman filter update gives

\[ \nu_t = \mathcal{N}(m_t^f, P_t^f) \text{ where } \]

\[ m_t^f = m_t^p + K_t(y_t - Hm_t^p), \quad P_t^f = (I - K_t H) P_t^p. \]

Note: This is only correct if also \( \mu_t \) is Gaussian.

The ensemble Kalman filter estimates \( m_t^p \) and \( P_t^p \) from \( (\tilde{x}_{t,j}) \), computes \( m_t^f \) and \( P_t^f \) and then samples \( (x_{t,j}) \) from \( \mathcal{N}(m_t^f, P_t^f) \).
Implementing the ensemble filter

Sampling is possible without computing $P_t^f$ or taking its square root. Can take $z_j \sim \mathcal{N}(m_t^p, P_t^p)$ and $V_j \sim \mathcal{N}(0, Q)$ and put

$$x_{t,j} = z_j + K_t(y_t - Hz_j + V_j).$$

To reduce the error due to the Gaussian assumption and to avoid taking the square root of $P_t^p$, take $z_j = \tilde{x}_{t,j}$. (Still need to estimate $P_t^p$ to compute $K_t$).

Changing $y_t$ results in an additive shift of the filter sample. Not correct if the shape of $\nu_t$ depends on $y_t$. 

H.R. Künsch, ETHZ
Summary of last week's lecture
State space models
Some basic results on SDE's
Definitions and problems
Exact filtering
Particle filter
Ensemble filter
Unknown parameters

Illustration of particle and ensemble updates

H.R. Künsch, ETHZ
Difficulties in high dimensions

Size of the sample $N$ is typically much smaller than dimensions of the state vector $x$ and observation vector $y$.

Ensemble filter can cope with this. Can reduce the error in estimating $P_t^p$ by sparsity assumptions.

Particle filter typically degenerates: Updating puts all the weights on one value in the sample. Challenge: Modify the particle filter so that it also works in high dimensions.
The likelihood function

Let $g$, $h$, $f_U$ and $f_V$ depend on an unknown parameter $\theta$.

Cannot compute the log likelihood function:

$$
\log L(\theta) = \sum_t \log p(y_t \mid y_1, \ldots, y_{t-1}, \theta)
\quad = \sum_t \log \int f_V(y_t - h(x_t))d\mu_t(x_t)
$$

(In the last expression both the integrand and $\mu_t$ depend on $\theta$.)
Offline estimation

This means: All observations are available at the beginning.

Options

- Estimate $\theta$ and $(x_t)$ by MCMC,
- Estimate $\theta$ by stochastic EM,
- Approximate the likelihood by filtering and maximize it.

Efficient implementations of all these options are challenging.
Online estimation

Observations become available sequentially.
Options
- Take $\theta$ as part of the state vector with deterministic evolution $\theta_t = \theta_{t-1}$.
- Approximate $\nu_t$ and its derivative w.r. to $\theta$ by Monte Carlo methods and use recursive optimization.

Again, this is not easy.
An SDE has the following form

\[ dx_t = g(x_t)dt + \sigma(x_t)dB_t \]

where the increments \( dB(t) \) of Brownian motion are jointly Gaussian with

\[ E[dB_t] = 0, \quad E[dB_t dB_s^T] = \delta_{ts} I \, dt \]

(\( I \) is the identity matrix, \( x_t \) and \( g \) are vectors, \( \sigma \) is a matrix).

Rigorously, an SDE is defined as an integral equation. I use the Ito version of the stochastic integral.

The solution of an SDE is a Markov process in continuous time.
Ito formula

ODE’s and SDE’s are different! E.g. the “intuitive” result

\[ d(f(x_t)) = \frac{\partial f}{\partial x}(x_t)dx_t \]

is no longer correct. One has to expand \( f \) up to second order, using

\[ dx_t dx_t^T = \sigma(x_t)\sigma(x_t)^T dt. \]

Example: If \( z_t = \exp(\lambda t + \sigma B_t) \), then

\[ dz_t = (\lambda + \sigma dB_t)z_t + \frac{1}{2}\sigma^2 z_t dt. \]
Simulating an SDE

To simulate the solution of an SDE, the easiest method is the Euler approximation

\[ x_{t+h} = x_t + h \, g(x_t) + \mathcal{N}(0, h \, \sigma(x_t)\sigma(x_t)^T) \].

There are better alternatives, but these are different from those used to solve ODE’s.
Gareth Roberts and coworkers found a way to simulate “exactly” the solution from an SDE of the following form

\[ dx_t = g(x_t)dt + dB_t, \quad g = \frac{\partial G}{\partial x}. \]

Exactly means that we can generate values \( x_{t_i} \) on an arbitrary grid which can be refined later on.

This can be done by simulating Brownian motion and using importance sampling. The importance weights are given by the density of the distribution of \( x \) with respect to that of \( B \). For this, we need Girsanov’s formula.
Girsanov’s formula

This density is equal to

$$
\exp \left( \int_0^t g(B_s) dB_s - \frac{1}{2} \int_0^t g(B_s)^2 ds \right).
$$

Ito’s formula allows to get rid of the stochastic integral:

$$
\int_0^t g(B_s) dB_s = G(B_t) - G(B_0) - \frac{1}{2} \int_0^t \Delta G(B_s) ds.
$$

To compute this, we still need the whole path of $B$. But can generate random unbiased weights that need $B$ only at a finite number of (random) time points.
Statistics for SDE’s

A key quantity in statistical applications of SDE’s are the transition densities for \( p(x_t \mid x_s) \). Exact computation requires solving a PDE (Fokker-Planck). Aproximate computations

- Euler approximation

\[
p(x_t \mid x_s) \approx \phi(x_t; x_s + (t - s)g(x_s), (t - s)\sigma(x_s)\sigma(x_s)^T).
\]

- Importance sampling approximations with additional intermediate time points.

- Unbiased estimation as in the case of exact simulation.
Thank you for your attention.