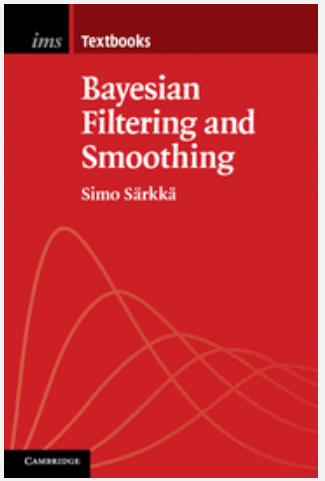
Tutorial: Bayesian Filtering and Smoothing

EUSIPCO 2014, Lisbon, Portugal

Simo Särkkä

Aalto University, Finland
becs.aalto.fi/~ssarkka/

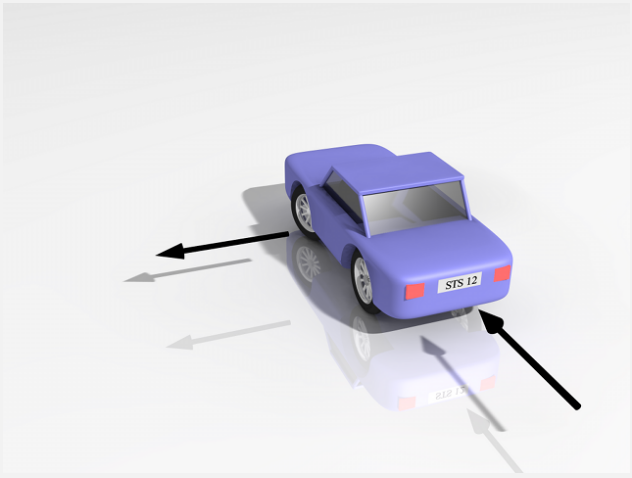
September 1, 2014



Learning Outcomes

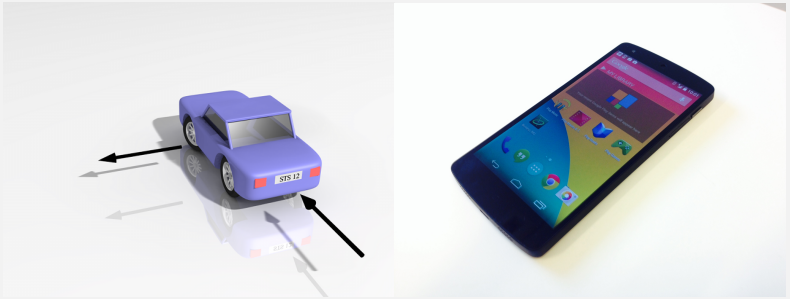
1. Principles of Bayesian inference in dynamic systems
2. Construction of probabilistic state space models
3. Bayesian filtering of state space models
4. Bayesian smoothing of state space models
5. Parameter estimation in state space models

Recursive Estimation of Dynamic Processes



- Dynamic, that is, time varying phenomenon - e.g., the motion state of a car or smart phone.
- The phenomenon is measured - for example by a radar or by acceleration and angular velocity sensors.
- The purpose is to compute the state of the phenomenon when only the measurements are observed.
- The solution should be recursive, where the information in new measurements is used for updating the old information.

Bayesian Modeling of Dynamics

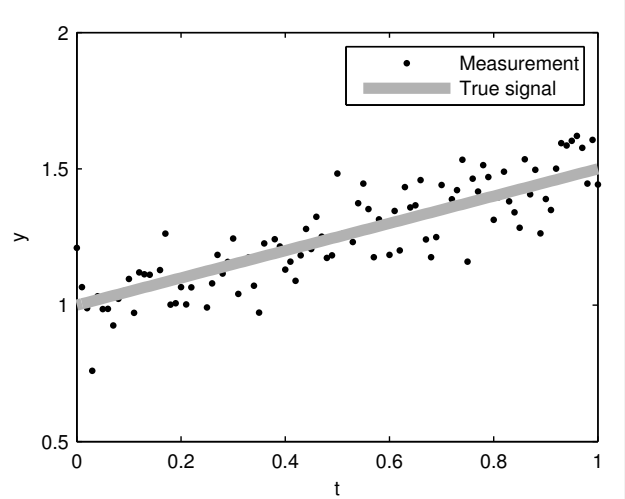


- The laws of physics, biology, epidemiology etc. are typically differential equations.
- Uncertainties and unknown sub-phenomena are modeled as stochastic processes:
  - Physical phenomena: differential equations + uncertainty $\Rightarrow$ stochastic differential equations.
  - Discretized physical phenomena: Stochastic differential equations $\Rightarrow$ stochastic difference equations.
  - Naturally discrete-time phenomena: Systems jumping from step to another.
- Stochastic differential and difference equations can be represented in stochastic state space form.

Bayesian Modeling of Measurements

- The relationship between measurements and phenomenon is mathematically modeled as a probability distribution.
- The measurements could be (in ideal world) computed if the phenomenon was known (forward model).
- The uncertainties in measurements and model are modeled as random processes.
- The measurement model is the conditional distribution of measurements given the state of the phenomenon.

Batch Linear Regression [1/2]



- Consider the linear regression model

$$y_k = \theta_1 + \theta_2\, t_k + \varepsilon_k, \qquad k = 1, \ldots, T,$$

with $\varepsilon_k \sim \mathrm{N}(0, \sigma^2)$ and $\boldsymbol{\theta} = (\theta_1, \theta_2) \sim \mathrm{N}(\mathbf{m}_0, \mathbf{P}_0)$.

- In probabilistic notation this is:

$$p(y_k \mid \boldsymbol{\theta}) = \mathrm{N}(y_k \mid \mathbf{H}_k\, \boldsymbol{\theta}, \sigma^2)$$
$$p(\boldsymbol{\theta}) = \mathrm{N}(\boldsymbol{\theta} \mid \mathbf{m}_0, \mathbf{P}_0),$$

where $\mathbf{H}_k = (1 \;\; t_k)$.

Batch Linear Regression [2/2]

- The Bayesian batch solution by the Bayes' rule:

$$\begin{aligned} p(\theta \mid y_{1:T}) &\propto p(\theta) \prod_{k=1}^{T} p(y_k \mid \theta) \\ &= \mathrm{N}(\theta \mid \mathbf{m}_0, \mathbf{P}_0) \prod_{k=1}^{T} \mathrm{N}(y_k \mid \mathbf{H}_k \theta, \sigma^2). \end{aligned}$$

- The posterior is Gaussian

$$p(\theta \mid y_{1:T}) = \mathrm{N}(\theta \mid \mathbf{m}_T, \mathbf{P}_T).$$

- The mean and covariance are given as

$$\begin{aligned} \mathbf{m}_T &= \left[\mathbf{P}_0^{-1} + \frac{1}{\sigma^2}\mathbf{H}^T\mathbf{H}\right]^{-1} \left[\frac{1}{\sigma^2}\mathbf{H}^T\mathbf{y} + \mathbf{P}_0^{-1}\mathbf{m}_0\right] \\ \mathbf{P}_T &= \left[\mathbf{P}_0^{-1} + \frac{1}{\sigma^2}\mathbf{H}^T\mathbf{H}\right]^{-1}, \end{aligned}$$

where $\mathbf{H}_k = (1 \; t_k)$, $\mathbf{H} = (\mathbf{H}_1; \mathbf{H}_2; \ldots; \mathbf{H}_T)$, $\mathbf{y} = (y_1; \ldots; y_T)$.

Recursive Linear Regression [1/4]

- Assume that we have already computed the posterior distribution, which is conditioned on the measurements up to $k-1$:

$$p(\theta \mid y_{1:k-1}) = \mathrm{N}(\theta \mid \mathbf{m}_{k-1}, \mathbf{P}_{k-1}).$$

- Assume that we get the kth measurement y_k. Using the equations from the previous slide we get

$$\begin{aligned} p(\theta \mid y_{1:k}) &\propto p(y_k \mid \theta)\, p(\theta \mid y_{1:k-1}) \\ &\propto \mathrm{N}(\theta \mid \mathbf{m}_k, \mathbf{P}_k). \end{aligned}$$

- The mean and covariance are given as

$$\mathbf{m}_k = \left[\mathbf{P}_{k-1}^{-1} + \frac{1}{\sigma^2}\mathbf{H}_k^T\mathbf{H}_k\right]^{-1}\left[\frac{1}{\sigma^2}\mathbf{H}_k^T y_k + \mathbf{P}_{k-1}^{-1}\mathbf{m}_{k-1}\right]$$

$$\mathbf{P}_k = \left[\mathbf{P}_{k-1}^{-1} + \frac{1}{\sigma^2}\mathbf{H}_k^T\mathbf{H}_k\right]^{-1}.$$

Recursive Linear Regression [2/4]

- By the matrix inversion lemma (or Woodbury identity):

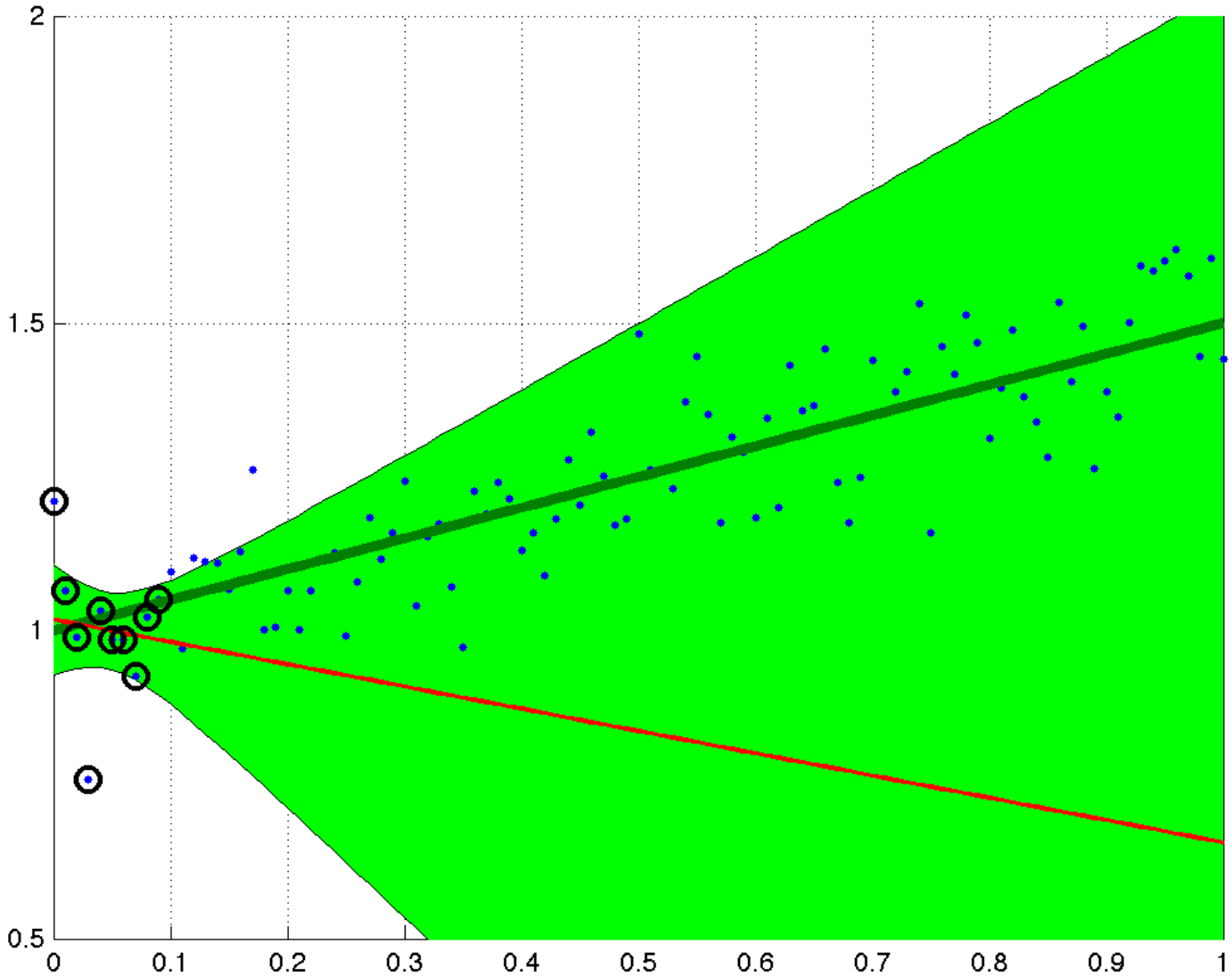
$$\mathbf{P}_k = \mathbf{P}_{k-1} - \mathbf{P}_{k-1}\mathbf{H}_k^T \left[\mathbf{H}_k \mathbf{P}_{k-1} \mathbf{H}_k^T + \sigma^2\right]^{-1} \mathbf{H}_k \mathbf{P}_{k-1}.$$

- Now the equations for the mean and covariance reduce to

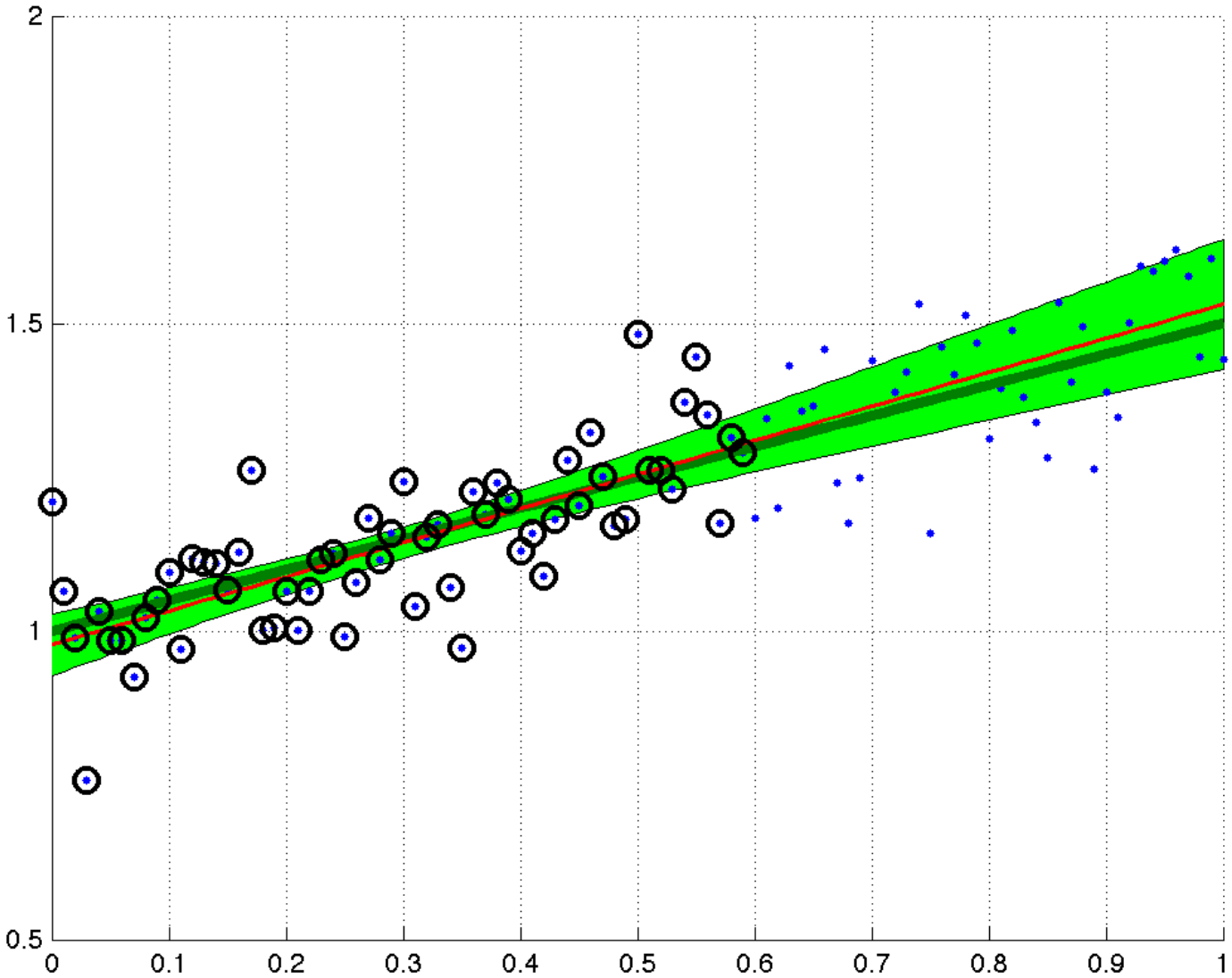
$$\begin{aligned}
S_k &= \mathbf{H}_k \mathbf{P}_{k-1} \mathbf{H}_k^T + \sigma^2 \\
\mathbf{K}_k &= \mathbf{P}_{k-1} \mathbf{H}_k^T S_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_{k-1} + \mathbf{K}_k [y_k - \mathbf{H}_k \mathbf{m}_{k-1}] \\
\mathbf{P}_k &= \mathbf{P}_{k-1} - \mathbf{K}_k S_k \mathbf{K}_k^T.
\end{aligned}$$

- Computing these for $k = 0, \ldots, T$ gives exactly the linear regression solution.
- A special case of Kalman filter.

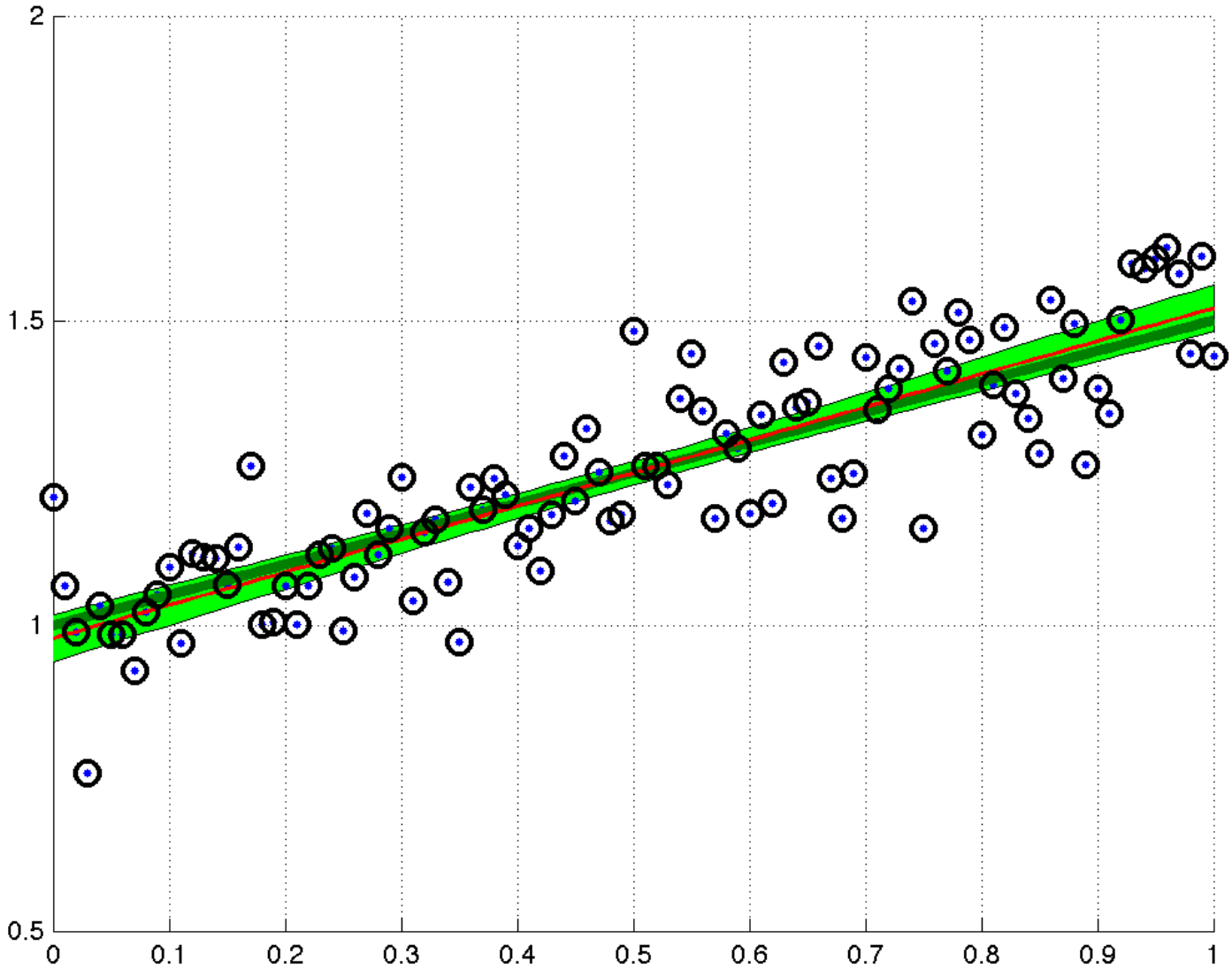
Recursive Linear Regression [3/4]



Recursive Linear Regression [3/4]

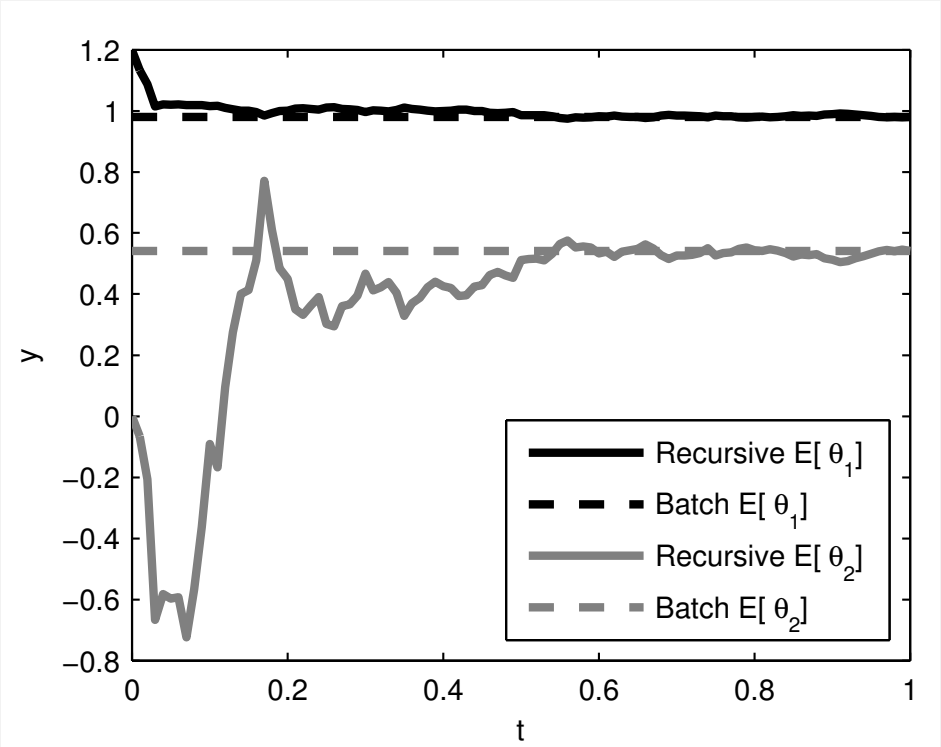


Recursive Linear Regression [3/4]



Recursive Linear Regression [4/4]

Convergence of the recursive solution to the batch solution – on the last step the solutions are exactly equal:



Batch vs. Recursive Estimation [1/2]

General batch solution:

- Specify the measurement model:

$$p(\mathbf{y}_{1:T} \mid \boldsymbol{\theta}) = \prod_k p(\mathbf{y}_k \mid \boldsymbol{\theta}).$$

- Specify the prior distribution $p(\boldsymbol{\theta})$.
- Compute posterior distribution by the Bayes' rule:

$$p(\boldsymbol{\theta} \mid \mathbf{y}_{1:T}) = \frac{1}{Z} p(\boldsymbol{\theta}) \prod_k p(\mathbf{y}_k \mid \boldsymbol{\theta}).$$

- Compute point estimates, moments, predictive quantities etc. from the posterior distribution.

Batch vs. Recursive Estimation [2/2]

General recursive solution:

- Specify the measurement likelihood $p(\mathbf{y}_k \mid \theta)$.
- Specify the prior distribution $p(\theta)$.
- Process measurements $\mathbf{y}_1, \ldots, \mathbf{y}_T$ one at a time, starting from the prior:

$$
\begin{aligned}
p(\theta \mid \mathbf{y}_1) &= \frac{1}{Z_1} p(\mathbf{y}_1 \mid \theta) p(\theta) \\
p(\theta \mid \mathbf{y}_{1:2}) &= \frac{1}{Z_2} p(\mathbf{y}_2 \mid \theta) p(\theta \mid \mathbf{y}_1) \\
p(\theta \mid \mathbf{y}_{1:3}) &= \frac{1}{Z_3} p(\mathbf{y}_3 \mid \theta) p(\theta \mid \mathbf{y}_{1:2}) \\
&\vdots \\
p(\theta \mid \mathbf{y}_{1:T}) &= \frac{1}{Z_T} p(\mathbf{y}_T \mid \theta) p(\theta \mid \mathbf{y}_{1:T-1}).
\end{aligned}
$$

- The result at the last step is the batch solution.

Advantages of Recursive Solution

- The recursive solution can be considered as the online learning solution to the Bayesian learning problem.
- Batch Bayesian inference is a special case of recursive Bayesian inference.
- The parameter can be modeled to change between the measurement steps $\Rightarrow$ basis of filtering theory.

Drift Model for Linear Regression [1/3]

- Let assume Gaussian random walk between the measurements in the linear regression model:

$$\begin{aligned} p(y_k \mid \theta_k) &= \mathrm{N}(y_k \mid \mathbf{H}_k \, \theta_k, \sigma^2) \\ p(\theta_k \mid \theta_{k-1}) &= \mathrm{N}(\theta_k \mid \theta_{k-1}, \mathbf{Q}) \\ p(\theta_0) &= \mathrm{N}(\theta_0 \mid \mathbf{m}_0, \mathbf{P}_0). \end{aligned}$$

- Again, assume that we already know

$$p(\theta_{k-1} \mid y_{1:k-1}) = \mathrm{N}(\theta_{k-1} \mid \mathbf{m}_{k-1}, \mathbf{P}_{k-1}).$$

- The joint distribution of θ_k and θ_{k-1} is (due to Markovianity of dynamics!):

$$p(\theta_k, \theta_{k-1} \mid y_{1:k-1}) = p(\theta_k \mid \theta_{k-1}) \, p(\theta_{k-1} \mid y_{1:k-1}).$$

Drift Model for Linear Regression [2/3]

- Integrating over θ_{k-1} gives:

$$p(\theta_k \mid y_{1:k-1}) = \int p(\theta_k \mid \theta_{k-1})\, p(\theta_{k-1} \mid y_{1:k-1})\, d\theta_{k-1}.$$

- This equation for Markov processes is called the Chapman-Kolmogorov equation.
- Because the distributions are Gaussian, the result is Gaussian

$$p(\theta_k \mid y_{1:k-1}) = \mathrm{N}(\theta_k \mid \mathbf{m}_k^-, \mathbf{P}_k^-),$$

where

$$\begin{aligned} \mathbf{m}_k^- &= \mathbf{m}_{k-1} \\ \mathbf{P}_k^- &= \mathbf{P}_{k-1} + \mathbf{Q}. \end{aligned}$$

Drift Model for Linear Regression [3/3]

- As in the pure recursive estimation, we get

$$\begin{aligned} p(\theta_k \mid y_{1:k}) &\propto p(y_k \mid \theta_k)\, p(\theta_k \mid y_{1:k-1}) \\ &\propto \mathrm{N}(\theta_k \mid \mathbf{m}_k, \mathbf{P}_k). \end{aligned}$$

- After applying the matrix inversion lemma, mean and covariance can be written as

$$\begin{aligned} S_k &= \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \sigma^2 \\ \mathbf{K}_k &= \mathbf{P}_k^- \mathbf{H}_k^T S_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k [y_k - \mathbf{H}_k \mathbf{m}_k^-] \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k S_k \mathbf{K}_k^T. \end{aligned}$$

- Again, we have derived a special case of the Kalman filter.
- The batch version of this solution would be much more complicated.

State Space Notation

- In the previous slide we formulated the model as

$$p(\theta_k \mid \theta_{k-1}) = \mathrm{N}(\theta_k \mid \theta_{k-1}, \mathbf{Q})$$
$$p(y_k \mid \theta_k) = \mathrm{N}(y_k \mid \mathbf{H}_k \, \theta_k, \sigma^2)$$

- But in Kalman filtering and control theory the vector of parameters θ_k is usually called "state" and denoted as $\mathbf{x}_k$.
- More standard state space notation:

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) = \mathrm{N}(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{Q})$$
$$p(y_k \mid \mathbf{x}_k) = \mathrm{N}(y_k \mid \mathbf{H}_k \, \mathbf{x}_k, \sigma^2)$$

- Or equivalently

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{q}_{k-1}$$
$$y_k = \mathbf{H}_k \, \mathbf{x}_k + r_k,$$

where $\mathbf{q}_{k-1} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$, $r_k \sim \mathrm{N}(0, \sigma^2)$.

Kalman Filter [1/2]

- The canonical Kalman filtering model is

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) = \mathrm{N}(\mathbf{x}_k \mid \mathbf{A}_{k-1}\,\mathbf{x}_{k-1}, \mathbf{Q}_{k-1})$$
$$p(\mathbf{y}_k \mid \mathbf{x}_k) = \mathrm{N}(\mathbf{y}_k \mid \mathbf{H}_k\,\mathbf{x}_k, \mathbf{R}_k).$$

- More often, this model can be seen in the form

$$\mathbf{x}_k = \mathbf{A}_{k-1}\,\mathbf{x}_{k-1} + \mathbf{q}_{k-1}$$
$$\mathbf{y}_k = \mathbf{H}_k\,\mathbf{x}_k + \mathbf{r}_k.$$

- The Kalman filter actually calculates the following distributions:

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}) = \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_k^-, \mathbf{P}_k^-)$$
$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) = \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_k, \mathbf{P}_k).$$

Kalman Filter [2/2]

- Prediction step of the Kalman filter:

$$\begin{aligned}\mathbf{m}_k^- &= \mathbf{A}_{k-1}\,\mathbf{m}_{k-1} \\ \mathbf{P}_k^- &= \mathbf{A}_{k-1}\,\mathbf{P}_{k-1}\,\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1}.\end{aligned}$$

- Update step of the Kalman filter:

$$\begin{aligned}\mathbf{S}_k &= \mathbf{H}_k\,\mathbf{P}_k^-\,\mathbf{H}_k^T + \mathbf{R}_k \\ \mathbf{K}_k &= \mathbf{P}_k^-\,\mathbf{H}_k^T\,\mathbf{S}_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\,[\mathbf{y}_k - \mathbf{H}_k\,\mathbf{m}_k^-] \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\,\mathbf{S}_k\,\mathbf{K}_k^T.\end{aligned}$$

- These equations can be derived from the general Bayesian filtering equations.

Probabilistic State Space Models [1/2]

- Generic non-linear state space models

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1})$$
$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{r}_k).$$

- Generic Markov models

$$\mathbf{x}_k \sim p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})$$
$$\mathbf{y}_k \sim p(\mathbf{y}_k \,|\, \mathbf{x}_k).$$

- Continuous-discrete state space models involving stochastic differential equations:

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}, t) + \mathbf{w}(t)$$
$$\mathbf{y}_k \sim p(\mathbf{y}_k \,|\, \mathbf{x}(t_k)).$$

Probabilistic State Space Models [2/2]

- Non-linear state space model with unknown parameters:

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}, \theta)$$
$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{r}_k, \theta).$$

- General Markovian state space model with unknown parameters:

$$\mathbf{x}_k \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \theta)$$
$$\mathbf{y}_k \sim p(\mathbf{y}_k \mid \mathbf{x}_k, \theta).$$

- Parameter estimation will be considered later – for now, we will attempt to estimate the state.
- Why Bayesian filtering and smoothing then?

Bayesian Filtering, Prediction and Smoothing

- In principle, we could just use the (batch) Bayes' rule

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_T \mid \mathbf{y}_1, \ldots, \mathbf{y}_T) = \frac{p(\mathbf{y}_1, \ldots, \mathbf{y}_T \mid \mathbf{x}_1, \ldots, \mathbf{x}_T)\, p(\mathbf{x}_1, \ldots, \mathbf{x}_T)}{p(\mathbf{y}_1, \ldots, \mathbf{y}_T)},$$

- Curse of computational complexity: complexity grows more than linearly with number of measurements (typically we have $O(T^3)$).
- Hence, we concentrate on the following:
  - Filtering distributions:

$$p(\mathbf{x}_k \mid \mathbf{y}_1, \ldots, \mathbf{y}_k), \qquad k = 1, \ldots, T.$$

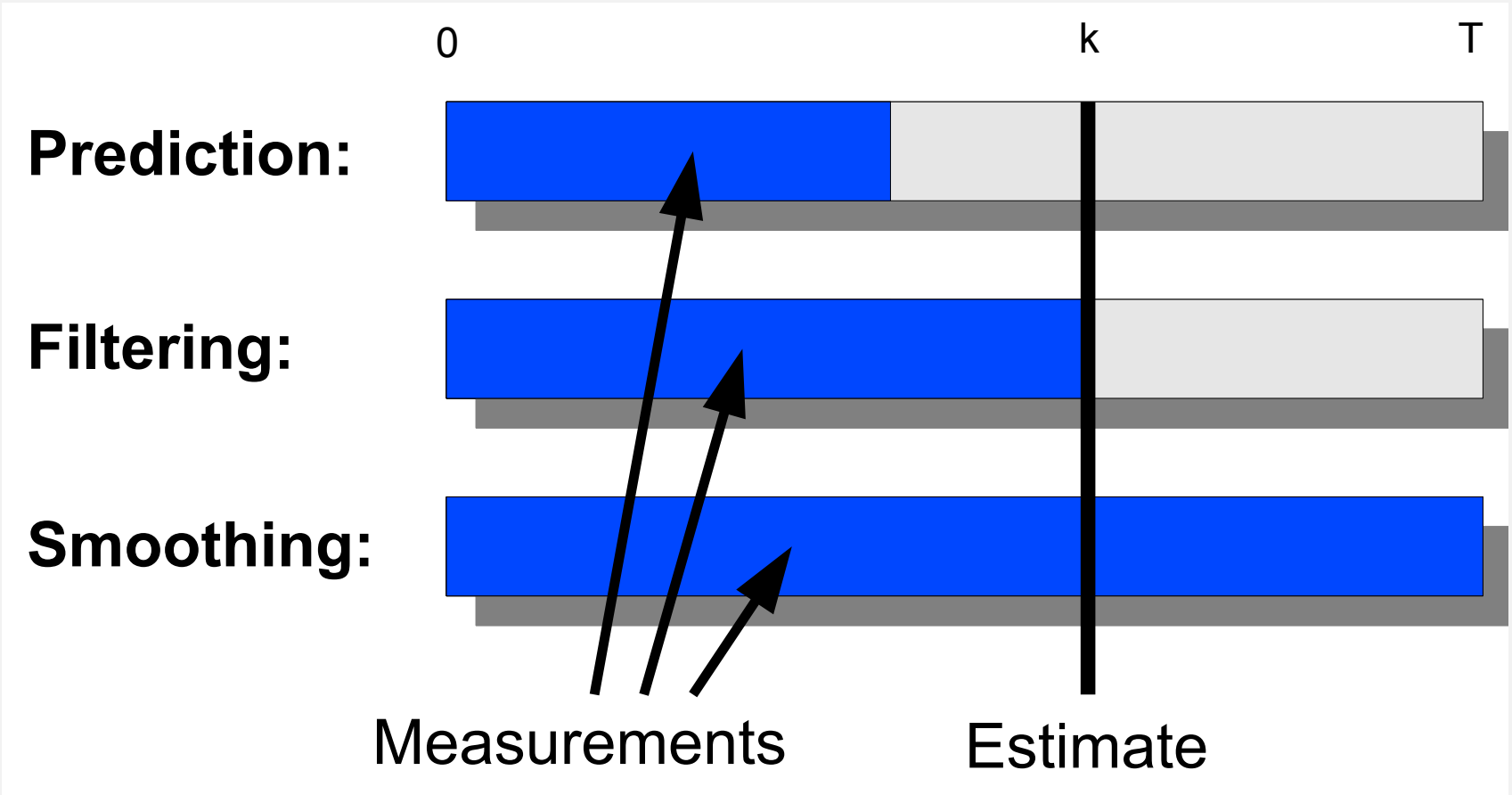
  - Prediction distributions:

$$p(\mathbf{x}_{k+n} \mid \mathbf{y}_1, \ldots, \mathbf{y}_k), \qquad k = 1, \ldots, T, \quad n = 1, 2, \ldots,$$

  - Smoothing distributions:

$$p(\mathbf{x}_k \mid \mathbf{y}_1, \ldots, \mathbf{y}_T), \qquad k = 1, \ldots, T.$$

Bayesian Filtering, Prediction and Smoothing (cont.)



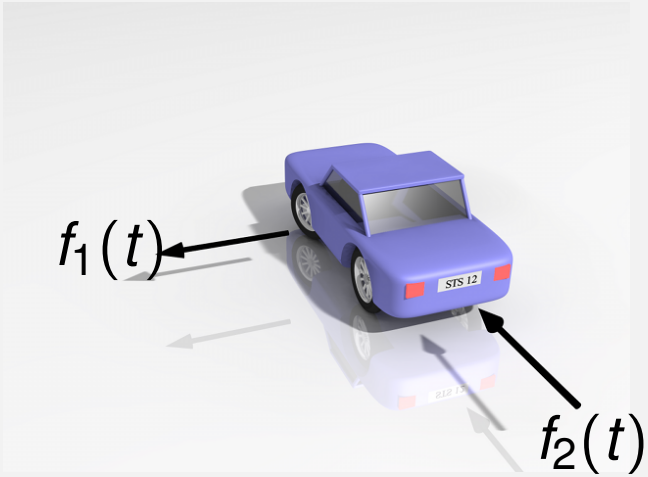
Filtering Algorithms

- Kalman filter is the classical optimal filter for linear-Gaussian models.
- Extended Kalman filter (EKF) is linearization based extension of Kalman filter to non-linear models.
- Unscented Kalman filter (UKF) is sigma-point transformation based extension of Kalman filter.
- Gauss-Hermite and Cubature Kalman filters (GHKF/CKF) are numerical integration based extensions of Kalman filter.
- Particle filter forms a Monte Carlo representation (particle set) to the distribution of the state estimate.
- Grid based filters approximate the probability distributions on a finite grid.
- Mixture Gaussian approximations are used, for example, in multiple model Kalman filters and Rao-Blackwellized Particle filters.

Smoothing Algorithms

- Rauch-Tung-Striebel (RTS) smoother is the closed form smoother for linear Gaussian models.
- Extended, statistically linearized and unscented RTS smoothers are the approximate nonlinear smoothers corresponding to EKF, SLF and UKF.
- Gaussian RTS smoothers: cubature RTS smoother, Gauss-Hermite RTS smoothers and various others
- Particle smoothing is based on approximating the smoothing solutions via Monte Carlo.
- Rao-Blackwellized particle smoother is a combination of particle smoothing and RTS smoothing.

Dynamic Model for a Car [1/3]



- The dynamics of the car in 2d (x_1, x_2) are given by the Newton's law:

$$\mathbf{f}(t) = m\,\mathbf{a}(t),$$

where $\mathbf{a}(t)$ is the acceleration, m is the mass of the car, and $\mathbf{f}(t)$ is a vector of (unknown) forces acting the car.

- We shall now model $\mathbf{f}(t)/m$ as a 2-dimensional white noise process:

$$\mathrm{d}^2 x_1/\mathrm{d}t^2 = w_1(t)$$
$$\mathrm{d}^2 x_2/\mathrm{d}t^2 = w_2(t).$$

Dynamic Model for a Car [2/3]

- If we define $x_3(t) = \mathrm{d}x_1/\mathrm{d}t$, $x_4(t) = \mathrm{d}x_2/\mathrm{d}t$, then the model can be written as a first order system of differential equations:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}}_{\mathbf{F}} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} + \underbrace{\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}}_{\mathbf{L}} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}.$$

- In shorter matrix form:

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{F}\,\mathbf{x} + \mathbf{L}\,\mathbf{w}.$$

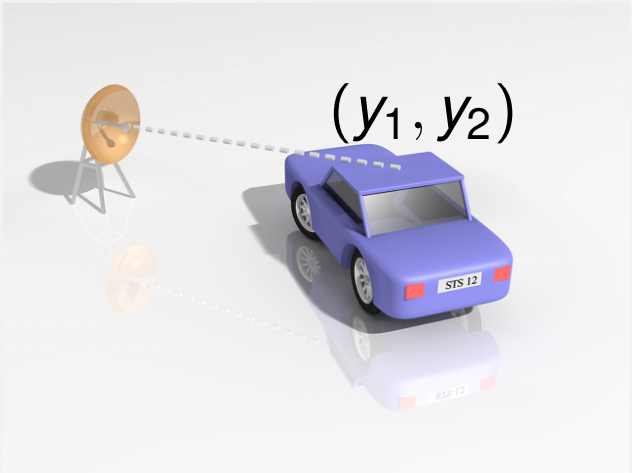
Dynamic Model for a Car [3/3]

- If the state of the car is measured (sampled) with sampling period Δt it suffices to consider the state of the car only at the time instances $t \in \{0, \Delta t, 2\Delta t, \ldots\}$.
- The dynamic model can be discretized, which leads to the linear difference equation model

$$\mathbf{x}_k = \mathbf{A}\,\mathbf{x}_{k-1} + \mathbf{q}_{k-1},$$

where $\mathbf{x}_k = \mathbf{x}(t_k)$, $\mathbf{A}$ is the transition matrix and $\mathbf{q}_k$ is a discrete-time Gaussian noise process.

Measurement Model for a Car



- Assume that the position of the car (x_1, x_2) is measured and the measurements are corrupted by Gaussian measurement noise $e_{1,k}, e_{2,k}$:

$$y_{1,k} = x_{1,k} + e_{1,k}$$
$$y_{2,k} = x_{2,k} + e_{2,k}.$$

- The measurement model can be now written as

$$\mathbf{y}_k = \mathbf{H}\,\mathbf{x}_k + \mathbf{e}_k, \qquad \mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Model for Car Tracking

- The dynamic and measurement models of the car now form a linear Gaussian filtering model:

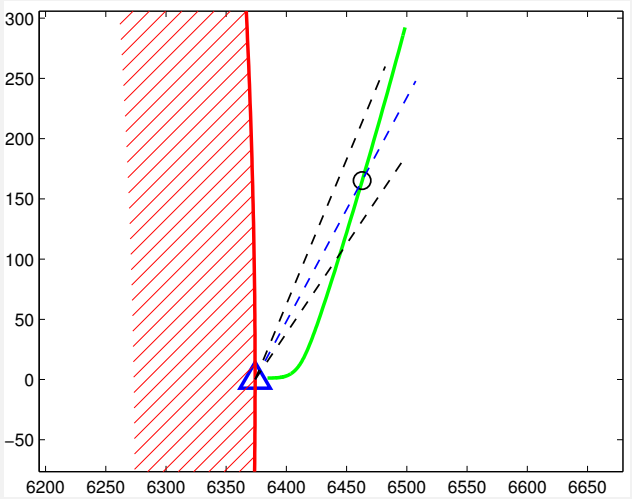
$$\mathbf{x}_k = \mathbf{A}\,\mathbf{x}_{k-1} + \mathbf{q}_{k-1}$$
$$\mathbf{y}_k = \mathbf{H}\,\mathbf{x}_k + \mathbf{r}_k,$$

  where $\mathbf{q}_{k-1} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ and $\mathbf{r}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{R})$.
- The posterior distribution is Gaussian

$$p(\mathbf{x}_k \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_k) = \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k).$$

- The mean $\mathbf{m}_k$ and covariance $\mathbf{P}_k$ of the posterior distribution can be computed by the Kalman filter.
- The whole history of the states can be estimated with the Rauch–Tung–Striebel smoother.

Re-Entry Vehicle Model [1/3]



- Gravitation law:

$$\mathbf{f} = m\,\mathbf{a}(t) = -\frac{G\,m\,M\,\mathbf{r}(t)}{|\mathbf{r}(t)|^3}.$$

- If we also model the friction and uncertainties:

$$\mathbf{a}(t) = -\frac{G\,M\,\mathbf{r}(t)}{|\mathbf{r}(t)|^3} - D(\mathbf{r}(t))\,|\mathbf{v}(t)|\,\mathbf{v}(t) + \mathbf{w}(t).$$

Re-Entry Vehicle Model [2/3]

- If we define $\mathbf{x} = (x_1 \; x_2 \; \frac{\mathrm{d}x_1}{\mathrm{d}t} \; \frac{\mathrm{d}x_2}{\mathrm{d}t} \;)^T$, the model is of the form

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}) + \mathbf{L}\,\mathbf{w}(t).$$

  where $\mathbf{f}(\cdot)$ is non-linear – do not confuse $\mathbf{f}(\cdot)$ with the force! – we just ran out of letters.

- The radar measurement:

$$r = \sqrt{(x_1 - x_r)^2 + (x_2 - y_r)^2} + e_r$$
$$\theta = \tan^{-1}\left(\frac{x_2 - y_r}{x_1 - x_r}\right) + e_\theta,$$

  where $e_r \sim \mathrm{N}(0, \sigma_r^2)$ and $e_\theta \sim \mathrm{N}(0, \sigma_\theta^2)$.

Re-Entry Vehicle Model [3/3]

- By suitable numerical integration scheme the model can be approximately written as discrete-time state space model:

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1})$$
$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{r}_k),$$

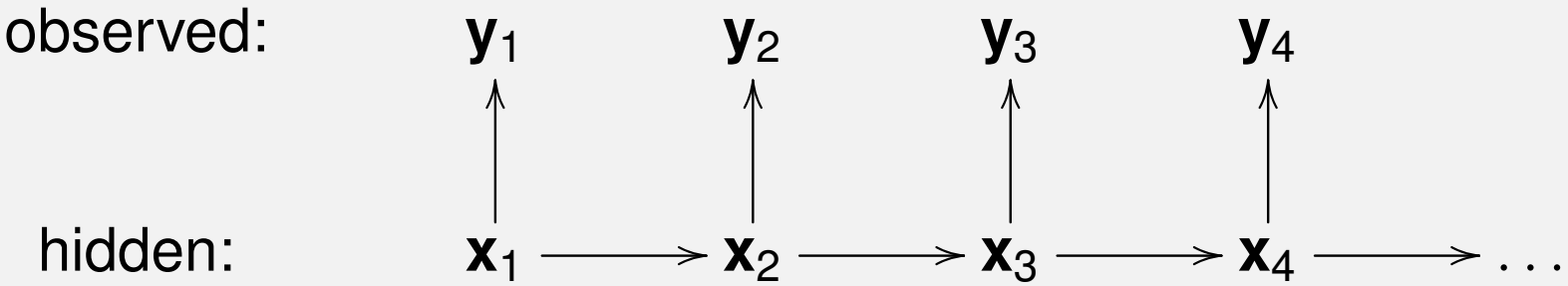
  where $\mathbf{y}_k$ is the vector of measurements, and $\mathbf{q}_{k-1} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ and $\mathbf{r}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{R})$.
- The tracking of the space vehicle can be now implemented by, e.g., extended Kalman filter (EKF), unscented Kalman filter (UKF) or particle filter.
- The history of states can be estimated with non-linear smoothers.

Probabilistic State Space Models: General Model

- General probabilistic state space model:

$$\text{dynamic model: } \mathbf{x}_k \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$$
$$\text{measurement model: } \mathbf{y}_k \sim p(\mathbf{y}_k \mid \mathbf{x}_k)$$

- $\mathbf{x}_k = (x_{k1}, \ldots, x_{kn})$ is the state and $\mathbf{y}_k = (y_{k1}, \ldots, y_{km})$ is the measurement.
- Has the form of hidden Markov model (HMM):



Probabilistic State Space Models: Example

Example (Gaussian random walk)

Gaussian random walk model can be written as

$$\begin{aligned} x_k &= x_{k-1} + w_{k-1}, \quad & w_{k-1} &\sim \mathrm{N}(0, q) \\ y_k &= x_k + e_k, & e_k &\sim \mathrm{N}(0, r), \end{aligned}$$

where x_k is the hidden state and y_k is the measurement. In terms of probability densities the model can be written as

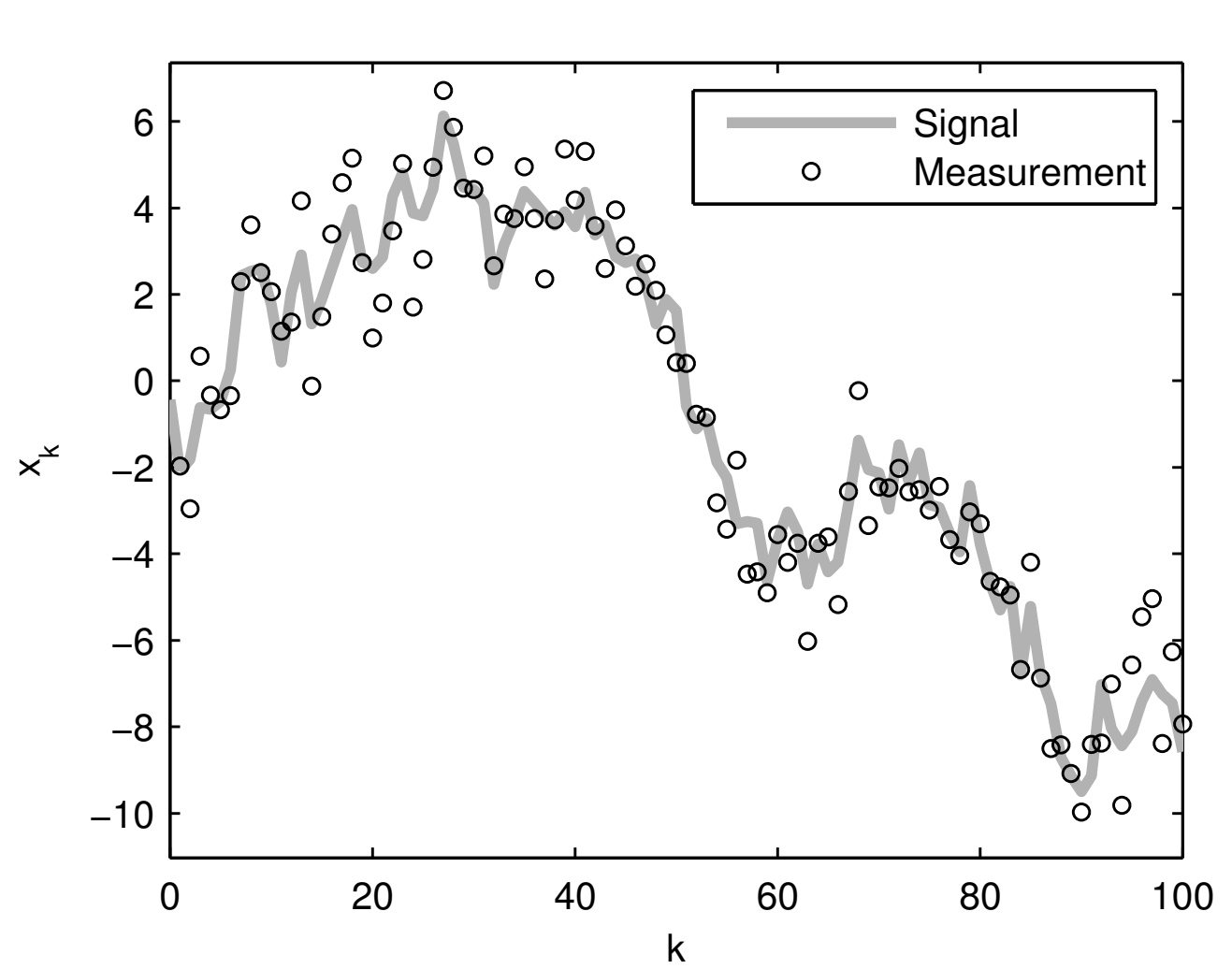
$$p(x_k \mid x_{k-1}) = \frac{1}{\sqrt{2\pi q}} \exp\left(-\frac{1}{2q}(x_k - x_{k-1})^2\right)$$

$$p(y_k \mid x_k) = \frac{1}{\sqrt{2\pi r}} \exp\left(-\frac{1}{2r}(y_k - x_k)^2\right)$$

which is a discrete-time state space model.

Probabilistic State Space Models: Example (cont.)

Example (Gaussian random walk (cont.))



Linear Gaussian State Space Models

- General form of linear Gaussian state space models:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{A}\,\mathbf{x}_{k-1} + \mathbf{q}_{k-1}, & \mathbf{q}_{k-1} &\sim \mathrm{N}(0, \mathbf{Q}) \\ \mathbf{y}_k &= \mathbf{H}\,\mathbf{x}_k + \mathbf{r}_k, & \mathbf{r}_k &\sim \mathrm{N}(0, \mathbf{R}) \end{aligned}$$

- In probabilistic notation the model is:

$$\begin{aligned} p(\mathbf{y}_k \mid \mathbf{x}_k) &= \mathrm{N}(\mathbf{y}_k \mid \mathbf{H}\,\mathbf{x}_k, \mathbf{R}) \\ p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) &= \mathrm{N}(\mathbf{x}_k \mid \mathbf{A}\,\mathbf{x}_{k-1}, \mathbf{Q}). \end{aligned}$$

- Surprisingly general class of models – linearity is from measurements to estimates, not from time to outputs.

Non-Linear State Space Models

- General form of non-linear Gaussian state space models:

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1})$$
$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{r}_k).$$

- $\mathbf{q}_k$ and $\mathbf{r}_k$ are typically assumed Gaussian.
- Functions $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ are non-linear functions modeling the dynamics and measurements of the system.
- Equivalent to the generic probabilistic models of the form

$$\mathbf{x}_k \sim p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})$$
$$\mathbf{y}_k \sim p(\mathbf{y}_k \,|\, \mathbf{x}_k).$$

Bayesian Optimal Filter: Principle

- Bayesian optimal filter computes the distribution

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k})$$

- Given the following:
  1. Prior distribution $p(\mathbf{x}_0)$.
  2. State space model:

$$\mathbf{x}_k \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$$
$$\mathbf{y}_k \sim p(\mathbf{y}_k \mid \mathbf{x}_k),$$

  3. Measurement sequence $\mathbf{y}_{1:k} = \mathbf{y}_1, \ldots, \mathbf{y}_k$.
- Computation is based on recursion rule for incorporation of the new measurement $\mathbf{y}_k$ into the posterior:

$$p(\mathbf{x}_{k-1} \mid \mathbf{y}_{1:k-1}) \longrightarrow p(\mathbf{x}_k \mid \mathbf{y}_{1:k})$$

Bayesian Optimal Filter: Formal Equations

Optimal filter

- Initialization: The recursion starts from the prior distribution $p(\mathbf{x}_0)$.
- Prediction: by the Chapman-Kolmogorov equation

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})\, p(\mathbf{x}_{k-1} \,|\, \mathbf{y}_{1:k-1})\, \mathrm{d}\mathbf{x}_{k-1}.$$

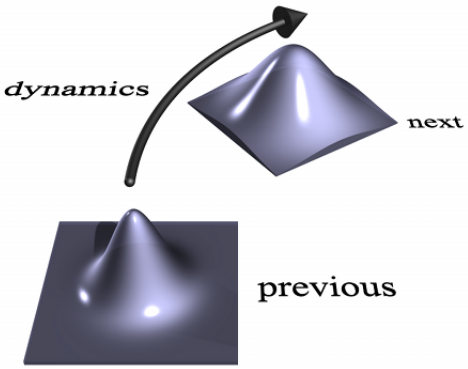
- Update: by the Bayes' rule

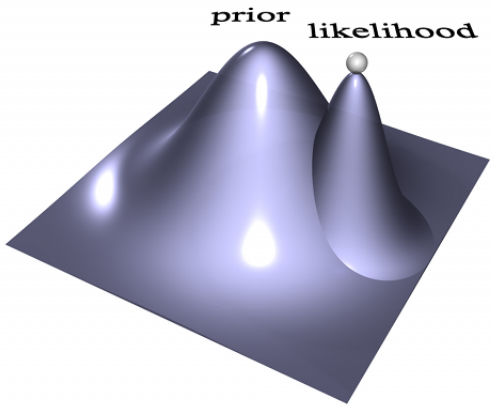
$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}) = \frac{1}{Z_k} p(\mathbf{y}_k \,|\, \mathbf{x}_k)\, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}).$$

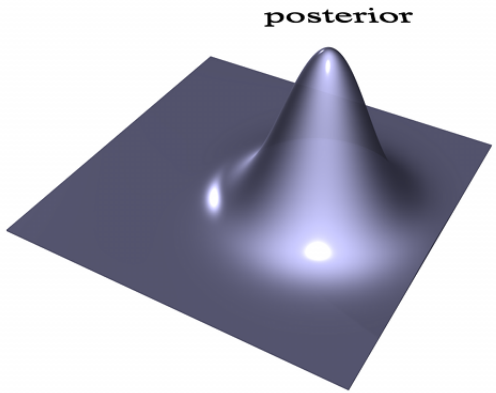
- The normalization constant $Z_k = p(\mathbf{y}_k \,|\, \mathbf{y}_{1:k-1})$ is given as

$$Z_k = \int p(\mathbf{y}_k \,|\, \mathbf{x}_k)\, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1})\, \mathrm{d}\mathbf{x}_k.$$

Bayesian Optimal Filter: Graphical Explanation







On prediction step the distribution of previous step is propagated through the dynamics.

Prior distribution from prediction and the likelihood of measurement.

The posterior distribution after combining the prior and likelihood by Bayes' rule.

Kalman Filter: Model

- Gaussian driven linear model, i.e., Gauss-Markov model:

$$\mathbf{x}_k = \mathbf{A}_{k-1}\,\mathbf{x}_{k-1} + \mathbf{q}_{k-1}$$
$$\mathbf{y}_k = \mathbf{H}_k\,\mathbf{x}_k + \mathbf{r}_k,$$

- $\mathbf{q}_{k-1} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_{k-1})$ white process noise.
- $\mathbf{r}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{R}_k)$ white measurement noise.
- $\mathbf{A}_{k-1}$ is the transition matrix of the dynamic model.
- $\mathbf{H}_k$ is the measurement model matrix.
- In probabilistic terms the model is

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) = \mathrm{N}(\mathbf{x}_k \mid \mathbf{A}_{k-1}\,\mathbf{x}_{k-1}, \mathbf{Q}_{k-1})$$
$$p(\mathbf{y}_k \mid \mathbf{x}_k) = \mathrm{N}(\mathbf{y}_k \mid \mathbf{H}_k\,\mathbf{x}_k, \mathbf{R}_k).$$

- Kalman filter computes

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) = \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_k, \mathbf{P}_k)$$

Kalman Filter: Equations

Kalman Filter

- Initialization: $\mathbf{x}_0 \sim \mathrm{N}(\mathbf{m}_0, \mathbf{P}_0)$
- Prediction step:

$$\begin{aligned}
\mathbf{m}_k^- &= \mathbf{A}_{k-1}\,\mathbf{m}_{k-1} \\
\mathbf{P}_k^- &= \mathbf{A}_{k-1}\,\mathbf{P}_{k-1}\,\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1}.
\end{aligned}$$

- Update step:

$$\begin{aligned}
\mathbf{v}_k &= \mathbf{y}_k - \mathbf{H}_k\,\mathbf{m}_k^- \\
\mathbf{S}_k &= \mathbf{H}_k\,\mathbf{P}_k^-\,\mathbf{H}_k^T + \mathbf{R}_k \\
\mathbf{K}_k &= \mathbf{P}_k^-\,\mathbf{H}_k^T\,\mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\,\mathbf{v}_k \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\,\mathbf{S}_k\,\mathbf{K}_k^T.
\end{aligned}$$

Non-Linear Gaussian State Space Model

Basic Non-Linear Gaussian State Space Model is of the form:

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}) + \mathbf{q}_{k-1}$$
$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{r}_k$$

- $\mathbf{x}_k \in \mathbb{R}^n$ is the state
- $\mathbf{y}_k \in \mathbb{R}^m$ is the measurement
- $\mathbf{q}_{k-1} \sim \mathrm{N}(0, \mathbf{Q}_{k-1})$ is the Gaussian process noise
- $\mathbf{r}_k \sim \mathrm{N}(0, \mathbf{R}_k)$ is the Gaussian measurement noise
- $\mathbf{f}(\cdot)$ is the dynamic model function
- $\mathbf{h}(\cdot)$ is the measurement model function

The Idea of Extended Kalman Filter

- In EKF, the non-linear functions are linearized as follows:

$$\mathbf{f}(\mathbf{x}) \approx \mathbf{f}(\mathbf{m}) + \mathbf{F}_{\mathbf{x}}(\mathbf{m})\,(\mathbf{x} - \mathbf{m})$$
$$\mathbf{h}(\mathbf{x}) \approx \mathbf{h}(\mathbf{m}) + \mathbf{H}_{\mathbf{x}}(\mathbf{m})\,(\mathbf{x} - \mathbf{m})$$

  where $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$, and $\mathbf{F}_{\mathbf{x}}$, $\mathbf{H}_{\mathbf{x}}$ are the Jacobian matrices of $\mathbf{f}$, $\mathbf{h}$, respectively.
- Only the first terms in linearization contribute to the approximate means of the functions $\mathbf{f}$ and $\mathbf{h}$.
- The second term has zero mean and defines the approximate covariances of the functions.
- Can be generalized into approximation of a non-linear transform.

Linear Approximation of Non-Linear Transform

The linear Gaussian approximation to the joint distribution of $\mathbf{x}$ and $\mathbf{y} = \mathbf{g}(\mathbf{x}) + \mathbf{q}$, where $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ and $\mathbf{q} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ is

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu}_L \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_L \\ \mathbf{C}_L^T & \mathbf{S}_L \end{pmatrix} \right),$$

where

$$\begin{aligned} \boldsymbol{\mu}_L &= \mathbf{g}(\mathbf{m}) \\ \mathbf{S}_L &= \mathbf{G}_{\mathbf{x}}(\mathbf{m})\, \mathbf{P}\, \mathbf{G}_{\mathbf{x}}^T(\mathbf{m}) + \mathbf{Q} \\ \mathbf{C}_L &= \mathbf{P}\, \mathbf{G}_{\mathbf{x}}^T(\mathbf{m}). \end{aligned}$$

EKF Equations

Extended Kalman filter

- Prediction:

$$\begin{aligned}\mathbf{m}_k^- &= \mathbf{f}(\mathbf{m}_{k-1}) \\ \mathbf{P}_k^- &= \mathbf{F}_{\mathbf{x}}(\mathbf{m}_{k-1})\,\mathbf{P}_{k-1}\,\mathbf{F}_{\mathbf{x}}^T(\mathbf{m}_{k-1}) + \mathbf{Q}_{k-1}.\end{aligned}$$

- Update:

$$\begin{aligned}\mathbf{v}_k &= \mathbf{y}_k - \mathbf{h}(\mathbf{m}_k^-) \\ \mathbf{S}_k &= \mathbf{H}_{\mathbf{x}}(\mathbf{m}_k^-)\,\mathbf{P}_k^-\,\mathbf{H}_{\mathbf{x}}^T(\mathbf{m}_k^-) + \mathbf{R}_k \\ \mathbf{K}_k &= \mathbf{P}_k^-\,\mathbf{H}_{\mathbf{x}}^T(\mathbf{m}_k^-)\,\mathbf{S}_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\,\mathbf{v}_k \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\,\mathbf{S}_k\,\mathbf{K}_k^T.\end{aligned}$$

Principle of Unscented Transform [1/4]

- Problem: Determine the mean and covariance of y:

$$x \sim \mathrm{N}(\mu, \sigma^2)$$
$$y = \sin(x)$$

- Recall the linearization based approximation:

$$y = \sin(\mu) + \frac{\partial \sin(\mu)}{\partial \mu}(x - \mu) + \ldots$$

which gives

$$\mathrm{E}[y] \approx \mathrm{E}[\sin(\mu) + \cos(\mu)(x - \mu)] = \sin(\mu)$$
$$\mathrm{Cov}[y] \approx \mathrm{E}[(\sin(\mu) + \cos(\mu)(x - \mu) - \sin(\mu))^2] = \cos^2(\mu)\,\sigma^2.$$

Principle of Unscented Transform [2/4]

- Form 3 sigma points as follows:

$$\begin{aligned} \mathcal{X}^{(0)} &= \mu \\ \mathcal{X}^{(1)} &= \mu + \sigma \\ \mathcal{X}^{(2)} &= \mu - \sigma. \end{aligned}$$

- Let's select some weights $W^{(0)}, W^{(1)}, W^{(2)}$ such that the original mean and variance can be recovered by

$$\begin{aligned} \mu &= \sum_i W^{(i)} \, \mathcal{X}^{(i)} \\ \sigma^2 &= \sum_i W^{(i)} \, (\mathcal{X}^{(i)} - \mu)^2. \end{aligned}$$

Principle of Unscented Transform [3/4]

- We use the same formula for approximating the moments of $y = \sin(x)$ as follows:

$$\mu = \sum_i W^{(i)} \sin(\mathcal{X}^{(i)})$$

$$\sigma^2 = \sum_i W^{(i)} (\sin(\mathcal{X}^{(i)}) - \mu)^2.$$

- For vectors $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ the generalization of standard deviation σ is the Cholesky factor $\mathbf{L} = \sqrt{\mathbf{P}}$:

$$\mathbf{P} = \mathbf{L}\,\mathbf{L}^T.$$

- The sigma points can be formed using columns of $\mathbf{L}$ (here c is a suitable positive constant):

$$\mathcal{X}^{(0)} = \mathbf{m}$$

$$\mathcal{X}^{(i)} = \mathbf{m} + c\,\mathbf{L}_i$$

$$\mathcal{X}^{(n+i)} = \mathbf{m} - c\,\mathbf{L}_i$$

Principle of Unscented Transform [4/4]

- For transformation $\mathbf{y} = \mathbf{g}(\mathbf{x})$ the approximation is:

$$\mu_y = \sum_i W^{(i)}\, \mathbf{g}(\mathcal{X}^{(i)})$$

$$\Sigma_y = \sum_i W^{(i)}\, (\mathbf{g}(\mathcal{X}^{(i)}) - \mu_y)\, (\mathbf{g}(\mathcal{X}^{(i)}) - \mu_y)^T.$$

- It is convenient to define transformed sigma points:

$$\mathcal{Y}^{(i)} = \mathbf{g}(\mathcal{X}^{(i)})$$

- Joint moments of $\mathbf{x}$ and $\mathbf{y} = \mathbf{g}(\mathbf{x}) + \mathbf{q}$ are then approximated as

$$\mathrm{E}\left[\begin{pmatrix} \mathbf{x} \\ \mathbf{g}(\mathbf{x}) + \mathbf{q} \end{pmatrix}\right] \approx \sum_i W^{(i)} \begin{pmatrix} \mathcal{X}^{(i)} \\ \mathcal{Y}^{(i)} \end{pmatrix} = \begin{pmatrix} \mathbf{m} \\ \mu_y \end{pmatrix}$$

$$\mathrm{Cov}\left[\begin{pmatrix} \mathbf{x} \\ \mathbf{g}(\mathbf{x}) + \mathbf{q} \end{pmatrix}\right]$$

$$\approx \sum_i W^{(i)} \begin{pmatrix} (\mathcal{X}^{(i)} - \mathbf{m})\, (\mathcal{X}^{(i)} - \mathbf{m})^T & (\mathcal{X}^{(i)} - \mathbf{m})\, (\mathcal{Y}^{(i)} - \mu_y)^T \\ (\mathcal{Y}^{(i)} - \mu_y)\, (\mathcal{X}^{(i)} - \mathbf{m})^T & (\mathcal{Y}^{(i)} - \mu_y)\, (\mathcal{Y}^{(i)} - \mu_y)^T + \mathbf{Q} \end{pmatrix}$$

Unscented Transform [1/3]

Unscented transform

The unscented transform approximation to the joint distribution of $\mathbf{x}$ and $\mathbf{y} = \mathbf{g}(\mathbf{x}) + \mathbf{q}$ where $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ and $\mathbf{q} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ is

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu}_U \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_U \\ \mathbf{C}_U^T & \mathbf{S}_U \end{pmatrix} \right),$$

where the sub-matrices are formed as follows:

1. Form the sigma points as

$$\begin{aligned} \mathcal{X}^{(0)} &= \mathbf{m} \\ \mathcal{X}^{(i)} &= \mathbf{m} + \sqrt{n + \lambda} \left[\sqrt{\mathbf{P}} \right]_i \\ \mathcal{X}^{(i+n)} &= \mathbf{m} - \sqrt{n + \lambda} \left[\sqrt{\mathbf{P}} \right]_i, \quad i = 1, \ldots, n \end{aligned}$$

Unscented transform (cont.)

2. Propagate the sigma points through $\mathbf{g}(\cdot)$:

$$\mathcal{Y}^{(i)} = \mathbf{g}(\mathcal{X}^{(i)}), \quad i = 0, \ldots, 2n.$$

3. The sub-matrices are then given as:

$$\boldsymbol{\mu}_U = \sum_{i=0}^{2n} W_i^{(m)} \, \mathcal{Y}^{(i)}$$

$$\mathbf{S}_U = \sum_{i=0}^{2n} W_i^{(c)} \, (\mathcal{Y}^{(i)} - \boldsymbol{\mu}_U) \, (\mathcal{Y}^{(i)} - \boldsymbol{\mu}_U)^T + \mathbf{Q}$$

$$\mathbf{C}_U = \sum_{i=0}^{2n} W_i^{(c)} \, (\mathcal{X}^{(i)} - \mathbf{m}) \, (\mathcal{Y}^{(i)} - \boldsymbol{\mu}_U)^T.$$

Unscented Transform [3/3]

Unscented transform (cont.)

- λ is a scaling parameter defined as $\lambda = \alpha^2 (n + \kappa) - n$.
- α and κ determine the spread of the sigma points.
- Weights $W_i^{(m)}$ and $W_i^{(c)}$ are given as follows:

$$
\begin{aligned}
W_0^{(m)} &= \lambda/(n+\lambda) \\
W_0^{(c)} &= \lambda/(n+\lambda) + (1 - \alpha^2 + \beta) \\
W_i^{(m)} &= 1/\{2(n+\lambda)\}, \quad i = 1, \ldots, 2n \\
W_i^{(c)} &= 1/\{2(n+\lambda)\}, \quad i = 1, \ldots, 2n,
\end{aligned}
$$

- β can be used for incorporating prior information on the (non-Gaussian) distribution of $\mathbf{x}$.

Unscented Kalman Filter (UKF): Algorithm [1/4]

Unscented Kalman filter: Prediction step

1. Form the sigma points:

$$
\begin{aligned}
\mathcal{X}_{k-1}^{(0)} &= \mathbf{m}_{k-1}, \\
\mathcal{X}_{k-1}^{(i)} &= \mathbf{m}_{k-1} + \sqrt{n+\lambda}\,\left[\sqrt{\mathbf{P}_{k-1}}\right]_i \\
\mathcal{X}_{k-1}^{(i+n)} &= \mathbf{m}_{k-1} - \sqrt{n+\lambda}\,\left[\sqrt{\mathbf{P}_{k-1}}\right]_i, \quad i = 1, \ldots, n.
\end{aligned}
$$

2. Propagate the sigma points through the dynamic model:

$$
\hat{\mathcal{X}}_k^{(i)} = \mathbf{f}(\mathcal{X}_{k-1}^{(i)}). \quad i = 0, \ldots, 2n.
$$

Unscented Kalman Filter (UKF): Algorithm [2/4]

Unscented Kalman filter: Prediction step (cont.)

3. Compute the predicted mean and covariance:

$$\mathbf{m}_k^- = \sum_{i=0}^{2n} W_i^{(m)} \, \hat{\mathcal{X}}_k^{(i)}$$

$$\mathbf{P}_k^- = \sum_{i=0}^{2n} W_i^{(c)} \, (\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-) \, (\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-)^T + \mathbf{Q}_{k-1}.$$

Unscented Kalman Filter (UKF): Algorithm [3/4]

Unscented Kalman filter: Update step

1. Form the sigma points:

$$
\begin{aligned}
\mathcal{X}_k^{-(0)} &= \mathbf{m}_k^-, \\
\mathcal{X}_k^{-(i)} &= \mathbf{m}_k^- + \sqrt{n+\lambda}\,\left[\sqrt{\mathbf{P}_k^-}\right]_i \\
\mathcal{X}_k^{-(i+n)} &= \mathbf{m}_k^- - \sqrt{n+\lambda}\,\left[\sqrt{\mathbf{P}_k^-}\right]_i, \quad i = 1, \ldots, n.
\end{aligned}
$$

2. Propagate sigma points through the measurement model:

$$
\hat{\mathcal{Y}}_k^{(i)} = \mathbf{h}(\mathcal{X}_k^{-(i)}), \quad i = 0, \ldots, 2n.
$$

Unscented Kalman Filter (UKF): Algorithm [4/4]

Unscented Kalman filter: Update step (cont.)

3. Compute the following:

$$\boldsymbol{\mu}_k = \sum_{i=0}^{2n} W_i^{(m)} \, \hat{\mathcal{Y}}_k^{(i)}$$

$$\mathbf{S}_k = \sum_{i=0}^{2n} W_i^{(c)} \, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k) \, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T + \mathbf{R}_k$$

$$\mathbf{C}_k = \sum_{i=0}^{2n} W_i^{(c)} \, (\mathcal{X}_k^{-(i)} - \mathbf{m}_k^-) \, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T$$

$$\mathbf{K}_k = \mathbf{C}_k \, \mathbf{S}_k^{-1}$$

$$\mathbf{m}_k = \mathbf{m}_k^- + \mathbf{K}_k \, [\mathbf{y}_k - \boldsymbol{\mu}_k]$$

$$\mathbf{P}_k = \mathbf{P}_k^- - \mathbf{K}_k \, \mathbf{S}_k \, \mathbf{K}_k^T.$$

Gaussian Moment Matching [1/2]

- Consider the transformation of $\mathbf{x}$ into $\mathbf{y}$:

$$\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$$
$$\mathbf{y} = \mathbf{g}(\mathbf{x}).$$

- Form Gaussian approximation to $(\mathbf{x}, \mathbf{y})$ by directly approximating the integrals:

$$\mu_M = \int \mathbf{g}(\mathbf{x})\ \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P})\, d\mathbf{x}$$
$$\mathbf{S}_M = \int (\mathbf{g}(\mathbf{x}) - \mu_M)\, (\mathbf{g}(\mathbf{x}) - \mu_M)^T\ \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P})\, d\mathbf{x}$$
$$\mathbf{C}_M = \int (\mathbf{x} - \mathbf{m})\, (\mathbf{g}(\mathbf{x}) - \mu_M)^T\ \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P})\, d\mathbf{x}.$$

Gaussian Moment Matching [2/2]

Gaussian moment matching

The moment matching based Gaussian approximation to the joint distribution of $\mathbf{x}$ and the transformed random variable $\mathbf{y} = \mathbf{g}(\mathbf{x}) + \mathbf{q}$ where $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ and $\mathbf{q} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ is given as

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \mu_M \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_M \\ \mathbf{C}_M^T & \mathbf{S}_M \end{pmatrix} \right),$$

where

$$\begin{aligned}
\mu_M &= \int \mathbf{g}(\mathbf{x})\, \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P})\, d\mathbf{x} \\
\mathbf{S}_M &= \int (\mathbf{g}(\mathbf{x}) - \mu_M)\, (\mathbf{g}(\mathbf{x}) - \mu_M)^T\, \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P})\, d\mathbf{x} + \mathbf{Q} \\
\mathbf{C}_M &= \int (\mathbf{x} - \mathbf{m})\, (\mathbf{g}(\mathbf{x}) - \mu_M)^T\, \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P})\, d\mathbf{x}.
\end{aligned}$$

Gaussian Filter [1/3]

Gaussian filter prediction

Compute the following Gaussian integrals:

$$\mathbf{m}_k^- = \int \mathbf{f}(\mathbf{x}_{k-1})\, \mathrm{N}(\mathbf{x}_{k-1} \,|\, \mathbf{m}_{k-1}, \mathbf{P}_{k-1})\, d\mathbf{x}_{k-1}$$

$$\mathbf{P}_k^- = \int (\mathbf{f}(\mathbf{x}_{k-1}) - \mathbf{m}_k^-)\, (\mathbf{f}(\mathbf{x}_{k-1}) - \mathbf{m}_k^-)^T \times \mathrm{N}(\mathbf{x}_{k-1} \,|\, \mathbf{m}_{k-1}, \mathbf{P}_{k-1})\, d\mathbf{x}_{k-1} + \mathbf{Q}_{k-1}.$$

Gaussian Filter [2/3]

Gaussian filter update

1. Compute the following Gaussian integrals:

$$\mu_k = \int \mathbf{h}(\mathbf{x}_k)\, \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-)\, d\mathbf{x}_k$$

$$\mathbf{S}_k = \int (\mathbf{h}(\mathbf{x}_k) - \mu_k)\, (\mathbf{h}(\mathbf{x}_k) - \mu_k)^T\, \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-)\, d\mathbf{x}_k + \mathbf{R}_k$$

$$\mathbf{C}_k = \int (\mathbf{x}_k - \mathbf{m}_k^-)\, (\mathbf{h}(\mathbf{x}_k) - \mu_k)^T\, \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-)\, d\mathbf{x}_k.$$

2. Then compute the following:

$$\mathbf{K}_k = \mathbf{C}_k\, \mathbf{S}_k^{-1}$$
$$\mathbf{m}_k = \mathbf{m}_k^- + \mathbf{K}_k\, (\mathbf{y}_k - \mu_k)$$
$$\mathbf{P}_k = \mathbf{P}_k^- - \mathbf{K}_k\, \mathbf{S}_k\, \mathbf{K}_k^T.$$

Gaussian Filter [3/3]

- Special case of assumed density filtering (ADF).
- Multidimensional Gauss-Hermite quadrature $\Rightarrow$ Gauss Hermite Kalman filter (GHKF).
- Cubature integration $\Rightarrow$ Cubature Kalman filter (CKF).
- Monte Carlo integration $\Rightarrow$ Monte Carlo Kalman filter (MCKF).
- Gaussian process / Bayes-Hermite Kalman filter: Form Gaussian process regression model from set of sample points and integrate the approximation.
- Linearization (EKF), unscented transform (UKF), central differences, divided differences can be considered as special cases.
- Note that all of these lead to Gaussian approximations

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) \approx \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_k, \mathbf{P}_k)$$

Spherical Cubature Rules

- The spherical cubature rule is exact up to third degree:

$$\begin{aligned}
&\int \mathbf{g}(\mathbf{x})\ \mathrm{N}(\mathbf{x} \mid \mathbf{m}, \mathbf{P})\, d\mathbf{x} \\
&= \int \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}}\,\boldsymbol{\xi})\ \mathrm{N}(\boldsymbol{\xi} \mid \mathbf{0}, \mathbf{I})\, d\boldsymbol{\xi} \\
&\approx \frac{1}{2n} \sum_{i=1}^{2n} \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}}\,\boldsymbol{\xi}^{(i)}),
\end{aligned}$$

  where

$$\boldsymbol{\xi}^{(i)} = \begin{cases} \sqrt{n}\,\mathbf{e}_i & , \quad i = 1, \ldots, n \\ -\sqrt{n}\,\mathbf{e}_{i-n} & , \quad i = n+1, \ldots, 2n, \end{cases}$$

  where $\mathbf{e}_i$ denotes a unit vector to the direction of coordinate axis i.

- A special case of unscented transform!

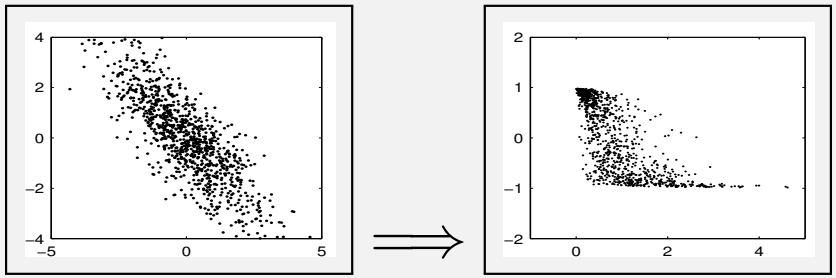
Multidimensional Gauss–Hermite Rules

- Cartesian product of classical Gauss–Hermite quadratures gives

$$
\begin{aligned}
&\int \mathbf{g}(\mathbf{x})\ \mathrm{N}(\mathbf{x} \mid \mathbf{m}, \mathbf{P})\, d\mathbf{x} \\
&= \int \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}}\,\boldsymbol{\xi})\ \mathrm{N}(\boldsymbol{\xi} \mid \mathbf{0}, \mathbf{I})\, d\boldsymbol{\xi} \\
&= \int \cdots \int \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}}\,\boldsymbol{\xi})\ \mathrm{N}(\xi_1 \mid 0, 1)\, d\xi_1 \times \cdots \times \mathrm{N}(\xi_n \mid 0, 1)\, d\xi_n \\
&\approx \sum_{i_1, \ldots, i_n} W^{(i_1)} \times \cdots \times W^{(i_n)} \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}}\,\boldsymbol{\xi}^{(i_1, \ldots, i_n)}).
\end{aligned}
$$

- $\boldsymbol{\xi}^{(i_1, \ldots, i_n)}$ are formed from the roots of Hermite polynomials.
- $W^{(i_j)}$ are the weights of one-dimensional Gauss–Hermite rules.

Particle Filtering: Principle



- Animation: Kalman vs. Particle Filtering:
  - ▸ Kalman filter animation
  - ▸ Particle filter animation

Sequential Importance Resampling: Idea

- Sequential Importance Resampling (SIR) (= particle filtering) is concerned with models

$$\mathbf{x}_k \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$$
$$\mathbf{y}_k \sim p(\mathbf{y}_k \mid \mathbf{x}_k)$$

- The SIS algorithm uses a weighted set of *particles* $\{(w_k^{(i)}, \mathbf{x}_k^{(i)}) \ : \ i = 1, \ldots, N\}$ such that

$$\mathrm{E}[\mathbf{g}(\mathbf{x}_k) \mid \mathbf{y}_{1:k}] \approx \sum_{i=1}^{N} w_k^{(i)} \mathbf{g}(\mathbf{x}_k^{(i)}).$$

- Or equivalently

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) \approx \sum_{i=1}^{N} w_k^{(i)} \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}),$$

  where $\delta(\cdot)$ is the Dirac delta function.
- Uses importance sampling sequentially.

Sequential Importance Resampling: Algorithm

Sequential Importance Resampling

- Draw point $\mathbf{x}_k^{(i)}$ from the importance distribution:

$$\mathbf{x}_k^{(i)} \sim \pi(\mathbf{x}_k \mid \mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{1:k}), \qquad i = 1, \ldots, N.$$

- Calculate new weights

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)})\, p(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{1:k})}, \qquad i = 1, \ldots, N,$$

  and normalize them to sum to unity.
- If the effective number of particles is too low, perform resampling.

Sequential Importance Resampling: Bootstrap filter

- In bootstrap filter we use the dynamic model as the importance distribution

$$\pi(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{1:k}) = p(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)})$$

and resample at every step:

Bootstrap Filter

- Draw point $\mathbf{x}_k^{(i)}$ from the dynamic model:

$$\mathbf{x}_k^{(i)} \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{(i)}), \qquad i = 1, \ldots, N.$$

- Calculate new weights

$$w_k^{(i)} \propto p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)}), \qquad i = 1, \ldots, N,$$

and normalize them to sum to unity.

- Perform resampling.

Sequential Importance Resampling: Optimal Importance Distribution

- The optimal importance distribution is

$$\pi(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{1:k}) = p(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k)$$

- Then the weight update reduces to

$$w_k^{(i)} \propto w_{k-1}^{(i)}\, p(\mathbf{y}_k \mid \mathbf{x}_{k-1}^{(i)}), \qquad i = 1, \ldots, N.$$

- The optimal importance distribution can be used, for example, when the state space is finite.

Sequential Importance Resampling: Importance Distribution via Kalman Filtering

- We can also form a Gaussian approximation to the optimal importance distribution:

$$p(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k) \approx \mathrm{N}(\mathbf{x}_k^{(i)} \mid \tilde{\mathbf{m}}_k^{(i)}, \tilde{\mathbf{P}}_k^{(i)}).$$

  by using a single prediction and update steps of a Gaussian filter starting from a singular distribution at $\mathbf{x}_{k-1}^{(i)}$.
- We can also replace above with the result of a Gaussian filter $\mathrm{N}(\mathbf{m}_{k-1}^{(i)}, \mathbf{P}_{k-1}^{(i)})$ started from a random initial mean.
- A very common way seems to be to use the previous sample as the mean: $\mathrm{N}(\mathbf{x}_{k-1}^{(i)}, \mathbf{P}_{k-1}^{(i)})$.
- A particle filter with UKF proposal has been given name unscented particle filter (UPF) – you can invent new PFs easily this way.

Rao-Blackwellized Particle Filter: Idea

- Rao-Blackwellized particle filtering (RBPF) is concerned with conditionally Gaussian models:

$$
\begin{aligned}
p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_{k-1}) &= \mathrm{N}(\mathbf{x}_k \mid \mathbf{A}_{k-1}(\mathbf{u}_{k-1})\,\mathbf{x}_{k-1}, \mathbf{Q}_{k-1}(\mathbf{u}_{k-1})) \\
p(\mathbf{y}_k \mid \mathbf{x}_k, \mathbf{u}_k) &= \mathrm{N}(\mathbf{y}_k \mid \mathbf{H}_k(\mathbf{u}_k)\,\mathbf{x}_k, \mathbf{R}_k(\mathbf{u}_k)) \\
p(\mathbf{u}_k \mid \mathbf{u}_{k-1}) &= (\text{any given form}),
\end{aligned}
$$

  where
  - $\mathbf{x}_k$ is the state
  - $\mathbf{y}_k$ is the measurement
  - $\mathbf{u}_k$ is an arbitrary latent variable
- Given the latent variables $\mathbf{u}_{1:T}$ the model is Gaussian.
- The RBPF uses SIR for the latent variables and computes the conditionally Gaussian part in closed form with Kalman filter.

Bayesian Smoothing Problem

- Probabilistic state space model:

$$\text{measurement model: } \mathbf{y}_k \sim p(\mathbf{y}_k \mid \mathbf{x}_k)$$
$$\text{dynamic model: } \mathbf{x}_k \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$$

- Assume that the filtering distributions $p(\mathbf{x}_k \mid \mathbf{y}_{1:k})$ have already been computed for all $k = 0, \ldots, T$.
- We want recursive equations of computing the smoothing distribution for all $k < T$:

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:T}).$$

- The recursion will go backwards in time, because on the last step, the filtering and smoothing distributions coincide:

$$p(\mathbf{x}_T \mid \mathbf{y}_{1:T}).$$

Bayesian Optimal Smoothing Equations

Bayesian Optimal Smoothing Equations

The Bayesian optimal smoothing equations consist of prediction step and backward update step:

$$p(\mathbf{x}_{k+1} \mid \mathbf{y}_{1:k}) = \int p(\mathbf{x}_{k+1} \mid \mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{y}_{1:k})\, \mathrm{d}\mathbf{x}_k$$

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:T}) = p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) \int \left[\frac{p(\mathbf{x}_{k+1} \mid \mathbf{x}_k)\, p(\mathbf{x}_{k+1} \mid \mathbf{y}_{1:T})}{p(\mathbf{x}_{k+1} \mid \mathbf{y}_{1:k})} \right] \mathrm{d}\mathbf{x}_{k+1}$$

The recursion is started from the filtering (and smoothing) distribution of the last time step $p(\mathbf{x}_T \mid \mathbf{y}_{1:T})$.

Rauch-Tung-Striebel Smoother

Backward recursion equations for the smoothed means $\mathbf{m}_k^s$ and covariances $\mathbf{P}_k^s$:

$$
\begin{aligned}
\mathbf{m}_{k+1}^- &= \mathbf{A}_k \, \mathbf{m}_k \\
\mathbf{P}_{k+1}^- &= \mathbf{A}_k \, \mathbf{P}_k \, \mathbf{A}_k^T + \mathbf{Q}_k \\
\mathbf{G}_k &= \mathbf{P}_k \, \mathbf{A}_k^T \, [\mathbf{P}_{k+1}^-]^{-1} \\
\mathbf{m}_k^s &= \mathbf{m}_k + \mathbf{G}_k \, [\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-] \\
\mathbf{P}_k^s &= \mathbf{P}_k + \mathbf{G}_k \, [\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-] \, \mathbf{G}_k^T,
\end{aligned}
$$

- $\mathbf{m}_k$ and $\mathbf{P}_k$ are the mean and covariance computed by the Kalman filter.
- The recursion is started from the last time step T, with $\mathbf{m}_T^s = \mathbf{m}_T$ and $\mathbf{P}_T^s = \mathbf{P}_T$.

Extended Rauch-Tung-Striebel Smoother

The equations for the extended RTS smoother are

$$
\begin{aligned}
\mathbf{m}_{k+1}^{-} &= \mathbf{f}(\mathbf{m}_k) \\
\mathbf{P}_{k+1}^{-} &= \mathbf{F}_{\mathbf{x}}(\mathbf{m}_k)\, \mathbf{P}_k\, \mathbf{F}_{\mathbf{x}}^{T}(\mathbf{m}_k) + \mathbf{Q}_k \\
\mathbf{G}_k &= \mathbf{P}_k\, \mathbf{F}_{\mathbf{x}}^{T}(\mathbf{m}_k)\, [\mathbf{P}_{k+1}^{-}]^{-1} \\
\mathbf{m}_k^{s} &= \mathbf{m}_k + \mathbf{G}_k\, [\mathbf{m}_{k+1}^{s} - \mathbf{m}_{k+1}^{-}] \\
\mathbf{P}_k^{s} &= \mathbf{P}_k + \mathbf{G}_k\, [\mathbf{P}_{k+1}^{s} - \mathbf{P}_{k+1}^{-}]\, \mathbf{G}_k^{T},
\end{aligned}
$$

where the matrix $\mathbf{F}_{\mathbf{x}}(\mathbf{m}_k)$ is the Jacobian matrix of $\mathbf{f}(\mathbf{x})$ evaluated at $\mathbf{m}_k$.

Gaussian Rauch-Tung-Striebel Smoother

The equations for the Gaussian RTS smoother are

$$\begin{aligned}
\mathbf{m}_{k+1}^- &= \int \mathbf{f}(\mathbf{x}_k)\, \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k)\, d\mathbf{x}_k \\
\mathbf{P}_{k+1}^- &= \int [\mathbf{f}(\mathbf{x}_k) - \mathbf{m}_{k+1}^-]\, [\mathbf{f}(\mathbf{x}_k) - \mathbf{m}_{k+1}^-]^T \\
&\qquad \times \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k)\, d\mathbf{x}_k + \mathbf{Q}_k \\
\mathbf{D}_{k+1} &= \int [\mathbf{x}_k - \mathbf{m}_k]\, [\mathbf{f}(\mathbf{x}_k) - \mathbf{m}_{k+1}^-]^T \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k)\, d\mathbf{x}_k \\
\mathbf{G}_k &= \mathbf{D}_{k+1}\, [\mathbf{P}_{k+1}^-]^{-1} \\
\mathbf{m}_k^s &= \mathbf{m}_k + \mathbf{G}_k\, (\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-) \\
\mathbf{P}_k^s &= \mathbf{P}_k + \mathbf{G}_k\, (\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-)\, \mathbf{G}_k^T.
\end{aligned}$$

Particle Smoothing: Direct SIR

- The smoothing solution can be obtained from SIR by storing the whole state histories into the particles.
- Special care is needed on the resampling step.
- The smoothed distribution approximation is then of the form

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:T}) \approx \sum_{i=1}^{N} w_T^{(i)} \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}),$$

  where $\mathbf{x}_k^{(i)}$ is the kth component in $\mathbf{x}_{1:T}^{(i)}$.
- Unfortunately, the approximation is often quite degenerate.

Particle Smoothing: Backward Simulation

Backward simulation particle smoother

Given the weighted set of particles $\{w_k^{(i)}, \mathbf{x}_k^{(i)}\}$ representing the filtering distributions:

- Choose $\tilde{\mathbf{x}}_T = \mathbf{x}_T^{(i)}$ with probability $w_T^{(i)}$.
- For $k = T-1, \ldots, 0$:
  1. Compute new weights by
  $$w_{k|k+1}^{(i)} \propto w_k^{(i)} p(\tilde{\mathbf{x}}_{k+1} \mid \mathbf{x}_k^{(i)})$$
  2. Choose $\tilde{\mathbf{x}}_k = \mathbf{x}_k^{(i)}$ with probability $w_{k|k+1}^{(i)}$

Given S iterations resulting in $\tilde{\mathbf{x}}_{1:T}^{(j)}$ for $j = 1, \ldots, S$ the smoothing distribution approximation is

$$p(\mathbf{x}_{1:T} \mid \mathbf{y}_{1:T}) \approx \frac{1}{S} \sum_j \delta(\mathbf{x}_{1:T} - \tilde{\mathbf{x}}_{1:T}^{(j)}).$$

Particle Smoothing: Reweighting

Reweighting Particle Smoother

Given the weighted set of particles $\{w_k^{(i)}, x_k^{(i)}\}$ representing the filtering distribution, we can form approximations to the marginal smoothing distributions as follows:

- Start by setting $w_{T|T}^{(i)} = w_T^{(i)}$ for $i = 1, \ldots, n$.
- For each $k = T-1, \ldots, 0$ do the following:
  - Compute new importance weights by

$$w_{k|T}^{(i)} \propto \sum_j w_{k+1|T}^{(j)} \frac{w_k^{(i)} p(\mathbf{x}_{k+1}^{(j)} \mid \mathbf{x}_k^{(i)})}{\left[\sum_l w_k^{(l)} p(\mathbf{x}_{k+1}^{(j)} \mid \mathbf{x}_k^{(l)})\right]}.$$

At each step k the marginal smoothing distribution can be approximated as

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:T}) \approx \sum_i w_{k|T}^{(i)} \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}).$$

Bayesian estimation of parameters

- State space model with unknown parameters $\boldsymbol{\theta} \in \mathbb{R}^d$:

$$\begin{aligned}
\boldsymbol{\theta} &\sim p(\boldsymbol{\theta}) \\
\mathbf{x}_0 &\sim p(\mathbf{x}_0 \mid \boldsymbol{\theta}) \\
\mathbf{x}_k &\sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \boldsymbol{\theta}) \\
\mathbf{y}_k &\sim p(\mathbf{y}_k \mid \mathbf{x}_k, \boldsymbol{\theta}).
\end{aligned}$$

- We approximate the marginal posterior distribution:

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

- The key is the prediction error decomposition:

$$p(\mathbf{y}_{1:T} \mid \boldsymbol{\theta}) = \prod_{k=1}^{T} p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1}, \boldsymbol{\theta})$$

- Luckily, the Bayesian filtering equations allow us to compute $p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1}, \boldsymbol{\theta})$ efficiently.

Bayesian estimation of parameters (cont.)

Recursion for marginal likelihood of parameters

The marginal likelihood of parameters is given by

$$p(\mathbf{y}_{1:T} \mid \theta) = \prod_{k=1}^{T} p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1}, \theta)$$

where the terms can be solved via the recursion

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \theta) = \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \theta)\, p(\mathbf{x}_{k-1} \mid \mathbf{y}_{1:k-1}, \theta)\, \mathrm{d}\mathbf{x}_{k-1}$$

$$p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1}, \theta) = \int p(\mathbf{y}_k \mid \mathbf{x}_k, \theta)\, p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \theta)\, \mathrm{d}\mathbf{x}_k$$

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k}, \theta) = \frac{p(\mathbf{y}_k \mid \mathbf{x}_k, \theta)\, p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \theta)}{p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1}, \theta)}.$$

Energy function

- The energy function:

$$\varphi_T(\boldsymbol{\theta}) = -\log p(\mathbf{y}_{1:T} \mid \boldsymbol{\theta}) - \log p(\boldsymbol{\theta}).$$

- The posterior distribution can be recovered via

$$p(\boldsymbol{\theta} \mid \mathbf{y}_{1:T}) \propto \exp(-\varphi_T(\boldsymbol{\theta})).$$

- The energy function can be evaluated recursively as follows:
  - Start from $\varphi_0(\theta) = -\log p(\boldsymbol{\theta})$.
  - At each step $k = 1, 2, \ldots, T$ compute the following:

$$\varphi_k(\boldsymbol{\theta}) = \varphi_{k-1}(\boldsymbol{\theta}) - \log p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1}, \boldsymbol{\theta})$$

- For linear models, we can evaluate the energy function exactly with help of Kalman filter.
- In non-linear models we can use Gaussian filters or particle filters for approximating the energy function.

Methods for parameter estimation

- MAP and ML-estimates can be computed by direct optimization of the energy function (or posterior).
- Derivatives of the energy function can be computed via sensitivity equations or Fisher's identity.
- Markov chain Monte Carlo (MCMC) methods can be used to sample from the posterior once the energy function is known.
- When particle filter approximation and MCMC is combined we get the exact particle Markov chain Monte Carlo (PMCMC) method.
- EM-algorithm can be used for computing MAP or ML-estimates when energy function is not available.

Summary

- Probabilistic state space models can be used to model various dynamic phenomena, e.g., dynamics of a car or re-entry vehicle.
- Bayesian filtering and smoothing methods solve Bayesian inference problems on state space models recursively.
- Kalman filter is the closed form linear Gaussian filtering solution.
- Extended Kalman filter (EKF) is linearization based extension of Kalman filter to non-linear models.
- Unscented Kalman filter (UKF) is sigma-point transformation based extension of Kalman filter.
- Gauss-Hermite and Cubature Kalman filters (GHKF/CKF) are numerical integration based extensions of Kalman filter.
- Particle filter forms a Monte Carlo representation (particle set) to the distribution of the state estimate.

Summary (cont.)

- Rauch-Tung-Striebel (RTS) smoother is the closed form smoother for linear Gaussian models.
- Extended, unscented, cubature, and related RTS smoothers are the approximate nonlinear smoothers for non-linear models.
- Particle smoothing is based on approximating the smoothing solutions via Monte Carlo.
- The marginal posterior distribution of state-space model parameters can be computed from the results of Bayesian filter.
- Given the marginal posterior, we can, e.g., compute MAP/ML estimates or use MCMC methods (or even EM-algorithms).
- For non-linear/non-Gaussian models the parameter posterior can be approximated with non-linear Kalman filters and particle filters.

More information on the topic

Book on Bayesian filtering and smoothing

S. Särkkä (2013). *Bayesian Filtering and Smoothing*. Cambridge University Press.

- ✓ Also freely available ONLINE at `becs.aalto.fi/~ssarkka/`