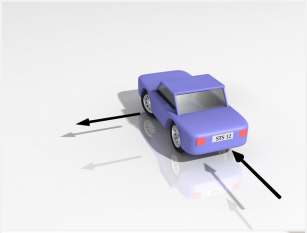
Lecture 9: Recap of the Course Topics

Simo Särkkä

April 13, 2016

Contents

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.

Probabilistic State Space Models: General Model

- General probabilistic state space model:

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

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

$$\begin{tikzcd} \text{observed:} & \mathbf{y}_1 & \mathbf{y}_2 & \mathbf{y}_3 & \mathbf{y}_4 & \\ \text{hidden:} & \mathbf{x}_1 \arrow[u] \arrow[r] & \mathbf{x}_2 \arrow[u] \arrow[r] & \mathbf{x}_3 \arrow[u] \arrow[r] & \mathbf{x}_4 \arrow[u] \arrow[r] & \ldots \end{tikzcd}$$

Probabilistic State Space Models: Example

Example (Gaussian random walk)

Gaussian random walk model can be written as

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

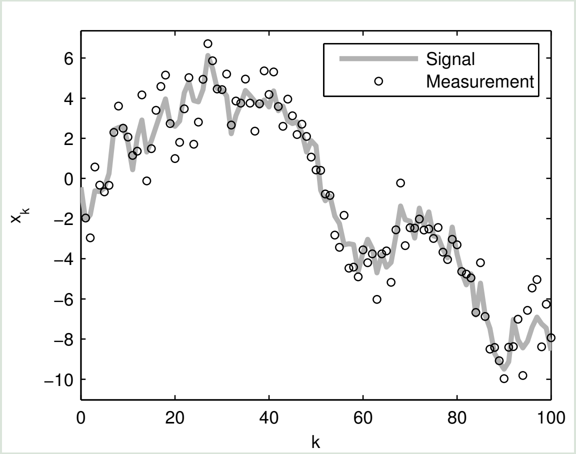
where x_k is the hidden state and y_k is the measurement. In terms of probability densities the model can be written as

$$\begin{aligned} p(x_k \mid x_{k-1}) &= \frac{1}{\sqrt{2\pi q}} \exp\left(-\frac{1}{2q}(x_k - x_{k-1})^2\right) \\ p(y_k \mid x_k) &= \frac{1}{\sqrt{2\pi r}} \exp\left(-\frac{1}{2r}(y_k - x_k)^2\right) \end{aligned}$$

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:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{A}\,\mathbf{x}_{k-1} + \mathbf{q}_{k-1}, & \mathbf{q}_{k-1} &\sim \mathrm{N}(0, \mathbf{Q}) \\ \mathbf{y}_k &= \mathbf{H}\,\mathbf{x}_k + \mathbf{r}_k, & \mathbf{r}_k &\sim \mathrm{N}(0, \mathbf{R}) \\ \mathbf{x}_0 &\sim \mathrm{N}(\mathbf{m}_0, \mathbf{P}_0). \end{aligned}$$

- In probabilistic notation the model is:

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

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

Non-Linear State Space Models

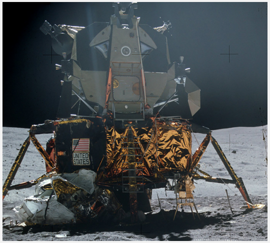
- General form of non-linear Gaussian state space models:

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

- $\mathbf{q}_k$ 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

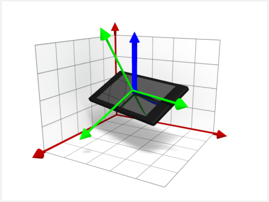
$$\mathbf{x}_k \sim p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})$$
$$\mathbf{y}_k \sim p(\mathbf{y}_k \,|\, \mathbf{x}_k).$$

Navigation of Lunar Module



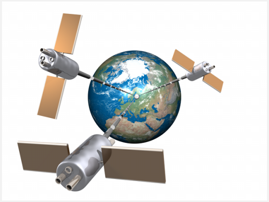
- The navigation system of Eagle lunar module AGC was based on an optimal filter - this was in the year 1969.
- The dynamic model was Newton's gravitation law.
- The measurements at lunar landing were the radar readings.
- On rendezvous with the command ship the orientation was computed with gyroscopes and their biases were also compensated with the radar.
- The optimal filter was an extended Kalman filter.

Smartphone Sensor Fusion



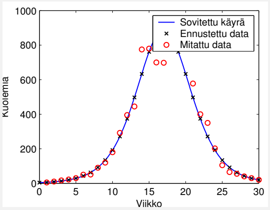
- Acceleration and angular velocity can, in principle, be integrated to give position and orientation.
- The known gravitation direction used for orientation tracking.
- Accelerometer can also be used to detect steps – gives a measurement of speed/distance.
- Barometer can be used to for local height tracking.
- Kalman/particle filters used for the sensor fusion.
- Bayesian filters also used for radio/magnetic map matching.

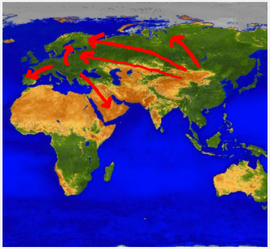
Satellite Navigation (GPS)



- The dynamic model in GPS receivers is often the Newton's second law where the force is completely random, that is, the Wiener velocity model.
- The measurements are time delays of satellite signals.
- The optimal filter computes the position and the accurate time.
- Also the errors caused by multi path can be modeled and compensated.
- Acceleration and angular velocity measurements are sometimes used as extra measurements.

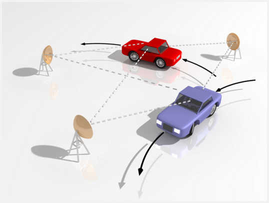
Spread of Disease

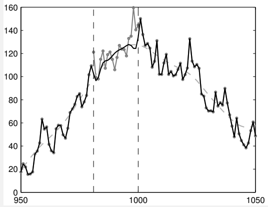




- Spreading of a disease in population can be modeled by differential equations.
- Modeling the unknown parameters and phenomena as random processes leads to stochastic dynamic model.
- The measurements in this case are the number of dead or recovered individuals.
- Optimal filter is used for computing the unknown parameters and the number of susceptible and infected individuals.
- It is also possible to predict when the maximum of the epidemic and how many casualties there will be.

Other Applications





- Target tracking, where one or many targets are tracked with many passive sensors - air surveillance.
- Time series prediction, where the model parameters are estimated from the measured time series and the unknown part is predicted using these parameters.
- Analysis and restoration of audio signals.
- Telecommunication systems - optimal receivers, signal detectors.
- State estimation of control systems - chemical processes, auto pilots, control systems of cars.

Bayesian Filtering, Prediction and Smoothing

- Recursively computable marginal distributions:
  - Filtering distributions:

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

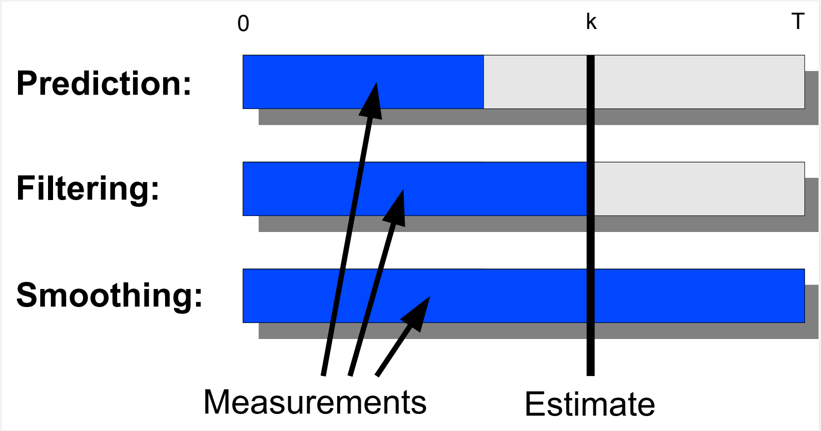
  - Prediction distributions:

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

  - Smoothing distributions:

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

Bayesian Filtering, Prediction and Smoothing (cont.)



Algorithms for Computing the Solutions

- Closed form solutions:
  - Kalman filter is the exact filter for linear-Gaussian models.
  - The Rauch–Tung–Striebel smoother (RTSS) is the exact smoother for linear-Gaussian models.
  - Grid filters and smoothers are solutions to Markov models with finite state spaces.
- Approximations:
  - Extended Kalman filter (EKF) uses linearization.
  - The extended Rauch–Tung–Striebel smoother (ERTSS).
  - Unscented Kalman filter (UKF) is unscented (sigma-point) transform based extension of Kalman filter.
  - The unscented Rauch–Tung–Striebel smoother (URTSS).
  - Gauss-Hermite and Cubature Kalman filters (GHKF/CKF) and smoothers use numerical integration.
  - Particle filters and smoothers use Monte Carlo.
  - Mixture Gaussian approximations are used, for example, in Rao-Blackwellized Particle filters.

Bayesian Filter: Principle

- Bayesian 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:

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

  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 Filter: Formal Equations

Bayesian filter

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

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

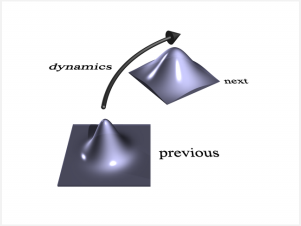
- Update: by the Bayes' rule

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

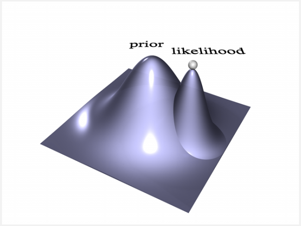
- The normalization constant $Z_k = p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1})$ is given as

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

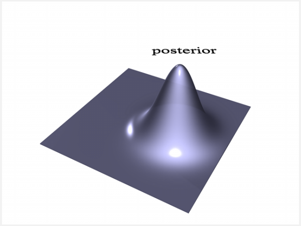
Bayesian Filter: Graphical Explanation



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:

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

- $\mathbf{q}_{k-1} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_{k-1})$ 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

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

- Kalman filter computes

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

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:

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

- $\mathbf{x}_k \in \mathbb{R}^n$ 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 **f** and **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 Transforms

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:

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

The Idea of Statistically Linearized Filter

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

$$\mathbf{f}(\mathbf{x}) \approx \mathbf{b}_f + \mathbf{A}_f\,(\mathbf{x} - \mathbf{m})$$
$$\mathbf{h}(\mathbf{x}) \approx \mathbf{b}_h + \mathbf{A}_h\,(\mathbf{x} - \mathbf{m})$$

  where $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$.

- The parameters $\mathbf{b}_f$, $\mathbf{A}_f$ and $\mathbf{b}_h$, $\mathbf{A}_h$ are chosen to minimize the mean squared errors of the form

$$\mathrm{MSE}_f(\mathbf{b}_f, \mathbf{A}_f) = \mathrm{E}[||\mathbf{f}(\mathbf{x}) - \mathbf{b}_f - \mathbf{A}_f\,\delta\mathbf{x}||^2]$$
$$\mathrm{MSE}_h(\mathbf{b}_h, \mathbf{A}_h) = \mathrm{E}[||\mathbf{h}(\mathbf{x}) - \mathbf{b}_h - \mathbf{A}_h\,\delta\mathbf{x}||^2]$$

  where $\delta\mathbf{x} = \mathbf{x} - \mathbf{m}$.

- Describing functions of the non-linearities with Gaussian input.

Statistically Linearized Filter

Statistically linearized filter

- Prediction (expectations w.r.t. $\mathbf{x}_{k-1} \sim \mathrm{N}(\mathbf{m}_{k-1}, \mathbf{P}_{k-1})$):

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

- Update (expectations w.r.t. $\mathbf{x}_k \sim \mathrm{N}(\mathbf{m}_k^-, \mathbf{P}_k^-)$):

$$\begin{aligned}
\mathbf{v}_k &= \mathbf{y}_k - \mathrm{E}[\mathbf{h}(\mathbf{x}_k)] \\
\mathbf{S}_k &= \mathrm{E}[\mathbf{h}(\mathbf{x}_k)\, \delta\mathbf{x}_k^T]\, (\mathbf{P}_k^-)^{-1}\; \mathrm{E}[\mathbf{h}(\mathbf{x}_k)\, \delta\mathbf{x}_k^T]^T + \mathbf{R}_k \\
\mathbf{K}_k &= \mathrm{E}[\mathbf{h}(\mathbf{x}_k)\, \delta\mathbf{x}_k^T]^T\, \mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\, \mathbf{v}_k \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\, \mathbf{S}_k\, \mathbf{K}_k^T.
\end{aligned}$$

Statistically Linearized Filter (cont.)

- If the function $\mathbf{g}(\mathbf{x})$ is differentiable, we have

$$\mathrm{E}[\mathbf{g}(\mathbf{x})\,(\mathbf{x}-\mathbf{m})^T] = \mathrm{E}[\mathbf{G}_x(\mathbf{x})]\,\mathbf{P},$$

  where $\mathbf{G}_x(\mathbf{x})$ is the Jacobian of $\mathbf{g}(\mathbf{x})$, and $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$.
- In practice, we can use the following property for computation of the expectation of the Jacobian:

$$\begin{aligned} \boldsymbol{\mu}(\mathbf{m}) &= \mathrm{E}[\mathbf{g}(\mathbf{x})] \\ \frac{\partial \boldsymbol{\mu}(\mathbf{m})}{\partial \mathbf{m}} &= \mathrm{E}[\mathbf{G}_x(\mathbf{x})]. \end{aligned}$$

- The resulting filter resembles EKF very closely.
- Related to replacing Taylor series with Fourier-Hermite series in the approximation.

Linearization Based Gaussian Approximation

- Problem: Determine the mean and covariance of y:

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

- Linearization based approximation:

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

which gives

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

Principle of Unscented Transform [1/3]

- 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 [2/3]

- 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):

$$\begin{aligned} \mathcal{X}^{(0)} &= \mathbf{m} \\ \mathcal{X}^{(i)} &= \mathbf{m} + c\,\mathbf{L}_i \\ \mathcal{X}^{(n+i)} &= \mathbf{m} - c\,\mathbf{L}_i \end{aligned}$$

Principle of Unscented Transform [3/3]

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

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

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

- 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

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

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

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

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 [2/3]

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:

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

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 **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:

$$
\begin{aligned}
\boldsymbol{\mu}_k &= \sum_{i=0}^{2n} W_i^{(m)} \hat{\mathcal{Y}}_k^{(i)} \\
\mathbf{S}_k &= \sum_{i=0}^{2n} W_i^{(c)} (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k) (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T + \mathbf{R}_k \\
\mathbf{C}_k &= \sum_{i=0}^{2n} W_i^{(c)} (\mathcal{X}_k^{-(i)} - \mathbf{m}_k^-) (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T \\
\mathbf{K}_k &= \mathbf{C}_k \, \mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k \, [\mathbf{y}_k - \boldsymbol{\mu}_k] \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k \, \mathbf{S}_k \, \mathbf{K}_k^T.
\end{aligned}
$$

Gaussian Moment Matching [1/2]

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

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

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

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

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

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

where

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

Gaussian Filter [1/3]

Gaussian filter prediction

Compute the following Gaussian integrals:

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

Gaussian Filter [2/3]

Gaussian filter update

1. Compute the following Gaussian integrals:

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

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

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

2. Then compute the following:

$$\begin{aligned}\mathbf{K}_k &= \mathbf{C}_k\,\mathbf{S}_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\,(\mathbf{y}_k - \boldsymbol{\mu}_k) \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\,\mathbf{S}_k\,\mathbf{K}_k^T.\end{aligned}$$

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:

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

  where

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

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

- A special case of unscented transform!

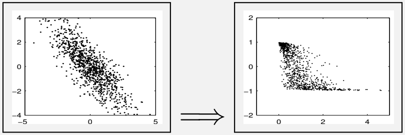
Multidimensional Gauss–Hermite Rules

- 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

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

- 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:

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

- 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 Smoothing Equations

Bayesian Smoothing Equations

The Bayesian 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

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

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

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
  $$w_{k|k+1}^{(i)} \propto w_k^{(i)} p(\tilde{\mathbf{x}}_{k+1} \,|\, \mathbf{x}_k^{(i)})$$
  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

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

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

$$\begin{aligned}
p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \theta) &= \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \theta)\, p(\mathbf{x}_{k-1} \mid \mathbf{y}_{1:k-1}, \theta)\, \mathrm{d}\mathbf{x}_{k-1} \\
p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1}, \theta) &= \int p(\mathbf{y}_k \mid \mathbf{x}_k, \theta)\, p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \theta)\, \mathrm{d}\mathbf{x}_k \\
p(\mathbf{x}_k \mid \mathbf{y}_{1:k}, \theta) &= \frac{p(\mathbf{y}_k \mid \mathbf{x}_k, \theta)\, p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \theta)}{p(\mathbf{y}_k \mid \mathbf{y}_{1:k-1}, \theta)}.
\end{aligned}$$

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.

The End

What is beyond this (or is there?):

- Even more approximate filters and smoothers.
- Even more methods for parameter inference.
- “Learning” of the dynamic and measurement functions.
- Square root filters/smoothers.
- Information filters/smoothers.
- Convergence/consistency of filters/smoothers.
- A lot of state-space models for real-world systems.
- Continuous-time dynamic models.
- Spatio-temporal systems.
- Stochastic (optimal) control theory.