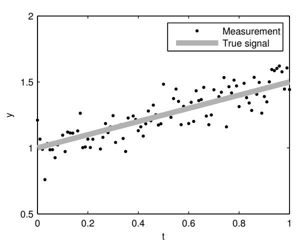
Lecture 2: From Linear Regression to Kalman Filter and Beyond

Simo Särkkä

February 3, 2016

Contents

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:

$$\begin{aligned} p(y_k \,|\, \boldsymbol{\theta}) &= \mathrm{N}(y_k \,|\, \mathbf{H}_k \, \boldsymbol{\theta}, \sigma^2) \\ p(\boldsymbol{\theta}) &= \mathrm{N}(\boldsymbol{\theta} \,|\, \mathbf{m}_0, \mathbf{P}_0), \end{aligned}$$

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

Batch Linear Regression [2/2]

- The Bayesian batch solution by the Bayes' rule:

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

- The posterior is Gaussian

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

- The mean and covariance are given as

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

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 $k-1$:

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

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

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

- The mean and covariance are given as

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

Recursive Linear Regression [2/4]

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

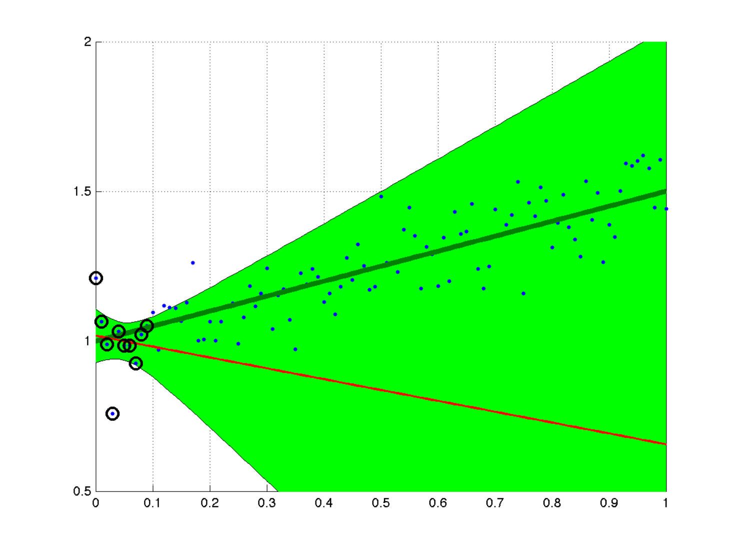
$$\mathbf{P}_k = \mathbf{P}_{k-1} - \mathbf{P}_{k-1}\mathbf{H}_k^\mathsf{T} \left[\mathbf{H}_k \mathbf{P}_{k-1} \mathbf{H}_k^\mathsf{T} + \sigma^2\right]^{-1} \mathbf{H}_k \mathbf{P}_{k-1}.$$

- Now the equations for the mean and covariance reduce to

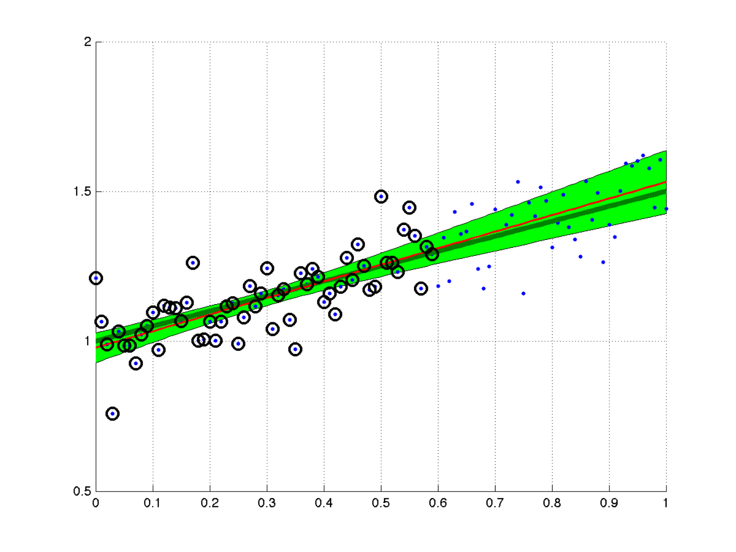
$$\begin{aligned}
S_k &= \mathbf{H}_k \mathbf{P}_{k-1} \mathbf{H}_k^\mathsf{T} + \sigma^2 \\
\mathbf{K}_k &= \mathbf{P}_{k-1} \mathbf{H}_k^\mathsf{T} S_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_{k-1} + \mathbf{K}_k [y_k - \mathbf{H}_k \mathbf{m}_{k-1}] \\
\mathbf{P}_k &= \mathbf{P}_{k-1} - \mathbf{K}_k S_k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

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

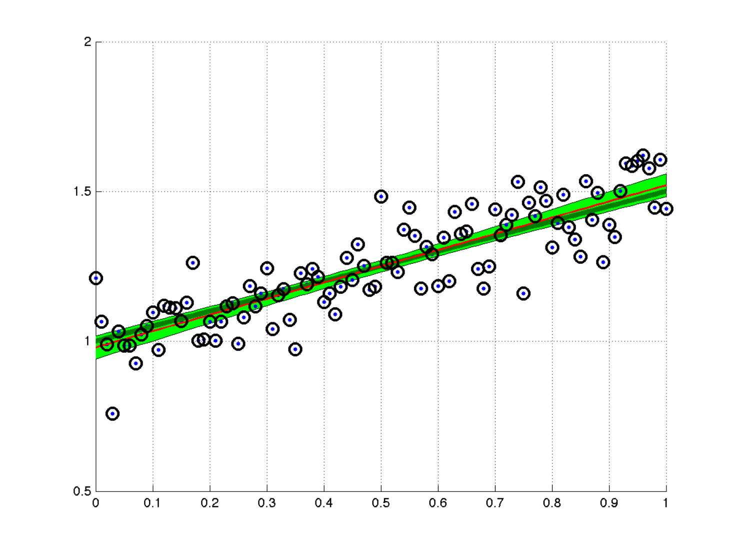
Recursive Linear Regression [3/4]



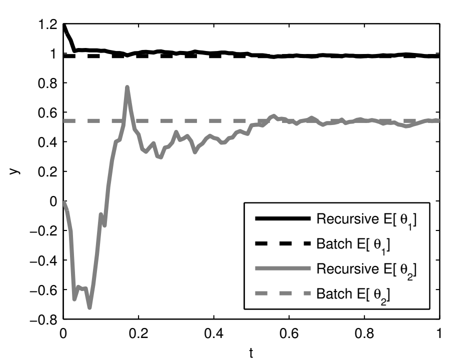
Recursive Linear Regression [3/4]



Recursive Linear Regression [3/4]



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

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

- 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:

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

- Again, assume that we already know

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

- The joint distribution of $\boldsymbol{\theta}_k$ and $\boldsymbol{\theta}_{k-1}$ is (due to Markovianity of dynamics!):

$$
p(\boldsymbol{\theta}_k, \boldsymbol{\theta}_{k-1} \mid y_{1:k-1}) = p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{k-1}) \, p(\boldsymbol{\theta}_{k-1} \mid y_{1:k-1}).
$$

Drift Model for Linear Regression [2/3]

- Integrating over $\boldsymbol{\theta}_{k-1}$ gives:

$$p(\boldsymbol{\theta}_k \mid y_{1:k-1}) = \int p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{k-1}) \, p(\boldsymbol{\theta}_{k-1} \mid y_{1:k-1}) \, d\boldsymbol{\theta}_{k-1}.$$

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

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

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

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

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

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

- 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

$$
\begin{aligned}
p(\boldsymbol{\theta}_k \,|\, \boldsymbol{\theta}_{k-1}) &= \mathrm{N}(\boldsymbol{\theta}_k \,|\, \boldsymbol{\theta}_{k-1}, \mathbf{Q}) \\
p(y_k \,|\, \boldsymbol{\theta}_k) &= \mathrm{N}(y_k \,|\, \mathbf{H}_k \, \boldsymbol{\theta}_k, \sigma^2)
\end{aligned}
$$

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

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

- Or equivalently

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

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

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

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

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

- The Kalman filter actually calculates the following distributions:

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

Kalman Filter [2/2]

- Prediction step of the Kalman filter:

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

- Update step of the Kalman filter:

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

- These equations will be derived from the general Bayesian filtering equations in the next lecture.

Probabilistic State Space Models [1/2]

- Generic non-linear state space models

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

- Generic Markov models

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

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

$$
\begin{aligned}
\frac{d\mathbf{x}}{dt} &= \mathbf{f}(\mathbf{x}, t) + \mathbf{w}(t) \\
\mathbf{y}_k &\sim p(\mathbf{y}_k \mid \mathbf{x}(t_k)).
\end{aligned}
$$

Probabilistic State Space Models [2/2]

- Non-linear state space model with unknown parameters:

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

- General Markovian state space model with unknown parameters:

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

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

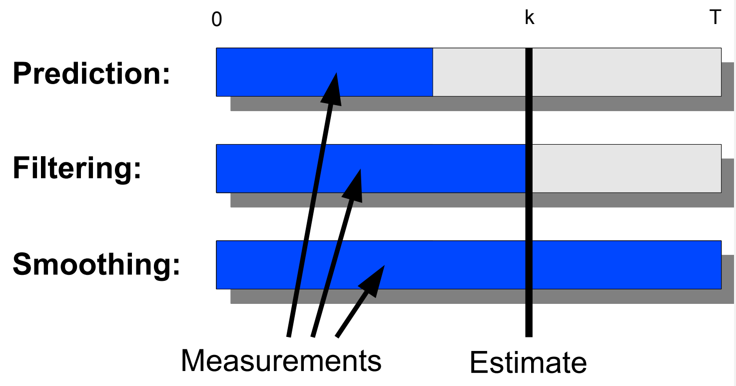
Bayesian Filtering, Prediction and Smoothing

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

$$
\begin{aligned}
&p(\mathbf{x}_1, \ldots, \mathbf{x}_T \mid \mathbf{y}_1, \ldots, \mathbf{y}_T) \\
&\qquad = \frac{p(\mathbf{y}_1, \ldots, \mathbf{y}_T \mid \mathbf{x}_1, \ldots, \mathbf{x}_T)\, p(\mathbf{x}_1, \ldots, \mathbf{x}_T)}{p(\mathbf{y}_1, \ldots, \mathbf{y}_T)},
\end{aligned}
$$

- 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.

Demonstration

Batch and recursive linear regression.

Linear and Linear in Parameters Models

- Basic linear regression model with noise ϵ_k:

$$y_k = a_0 + a_1\, s_k + \epsilon_k, \qquad k = 1, \ldots, N.$$

- Define matrix $\mathbf{H}_k = (1 \; s_k)$ and state $\mathbf{x} = (a_0 \; a_1)^T$:

$$y_k = \mathbf{H}_k\, \mathbf{x} + e_k, \qquad k = 1, \ldots, N.$$

- For notation sake we can also define $\mathbf{x}_k = \mathbf{x}$ such that $\mathbf{x}_k = \mathbf{x}_{k-1}$:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{x}_{k-1} \\ y_k &= \mathbf{H}_k\, \mathbf{x}_k + e_k. \end{aligned}$$

- Thus we have a linear Gaussian state space model, solvable with the basic Kalman filter.

Linear and Linear in Parameters Models (cont.)

- More general linear regression models:

$$y_k = a_0 + a_1 \, s_{k,1} + \cdots + a_d \, s_{k,d} + \epsilon_k, \qquad k = 1, \ldots, N.$$

- Defining matrix $\mathbf{H}_k = (1 \; s_{k,1} \; \cdots \; s_{k,d})$ and state $\mathbf{x}_k = \mathbf{x} = (a_0 \; a_1 \; \cdots \; a_d)^T$ gives linear Gaussian state space model:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{x}_{k-1} \\ y_k &= \mathbf{H}_k \, \mathbf{x}_k + \epsilon_k. \end{aligned}$$

- Linear in parameters models:

$$y_k = a_0 + a_1 \, f_1(s_k) + \cdots + a_d \, f_d(s_k) + \epsilon_k.$$

- Definitions $\mathbf{H}_k = (1 \; f_1(s_k) \; \cdots \; f_d(s_k))$ and $\mathbf{x}_k = \mathbf{x} = (a_0 \; a_1 \; \cdots \; a_d)^T$ again give linear Gaussian state space model.

Non-Linear and Neural Network Models

- Non-linearity in measurements models arises in generalized linear models, e.g.

$$y_k = g^{-1}(a_0 + a_1\, s_k) + \epsilon_k.$$

- The measurement model is now non-linear and if we define $\mathbf{x} = (a_0\ a_1)^T$ and $h(\mathbf{x}) = g^{-1}(x_1 + x_2\, s_k)$ we get non-linear Gaussian state space model:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{x}_{k-1} \\ y_k &= h(\mathbf{x}_k) + \epsilon_k. \end{aligned}$$

- Neural network models such as multi-layer perceptron (MLP) models can be also transformed into the above form.
- Instead of basic Kalman filter we need extended Kalman filter or unscented Kalman filter to cope with the non-linearity.

Adaptive Filtering Models

- In digital signal processing, a commonly used signal model is the autoregressive model

$$y_k = w_1 \, y_{k-1} + \cdots + w_d \, y_{k-d} + \epsilon_k,$$

- In adaptive filtering the weights w_i are estimated from data.
- If we define matrix $\mathbf{H}_k = (y_{k-1} \ \cdots \ y_{k-d})$ and state as $\mathbf{x}_k = (w_1 \ \cdots \ w_d)^T$, we get linear Gaussian state space model:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{x}_{k-1} \\ y_k &= \mathbf{H}_k \, \mathbf{x}_k + \epsilon_k. \end{aligned}$$

- The estimation problem can be solved with Kalman filter.
- The LMS algorithm can be interpreted as approximate version of this Kalman filter.

Adaptive Filtering Models (cont.)

- In time varying autoregressive models (TVAR) models the weights are time-varying:

$$y_k = w_{1,k}\, y_{k-1} + \cdots + w_{d,k}\, y_{k-d} + \epsilon_k,$$

- Typical model for the time dependence of weights:

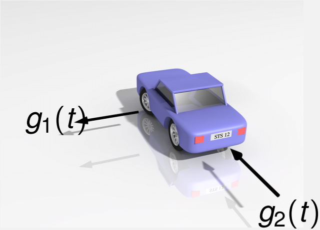
$$w_{i,k} = w_{i,k-1} + q_{k-1,i}, \quad q_{k-1,i} \sim \mathrm{N}(0, \sigma^2), \quad i = 1, \ldots, d.$$

- Can be written as linear Gaussian state space model with process noise $\mathbf{q}_{k-1} = \begin{pmatrix} q_{k-1,1} & \cdots & q_{k-1,d} \end{pmatrix}^T$:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{x}_{k-1} + \mathbf{q}_{k-1} \\ y_k &= \mathbf{H}_k\, \mathbf{x}_k + \epsilon_k. \end{aligned}$$

- More general (TV)ARMA models can be handled similarly.

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:

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

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

- We shall now model $\mathbf{g}(t)/m$ 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:

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

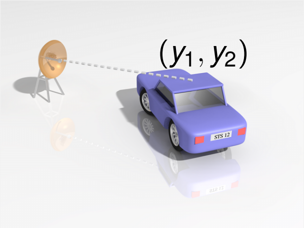
Dynamic Model for a Car [3/3]

- 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 $e_{1,k}, e_{2,k}$:

$$
\begin{aligned}
y_{1,k} &= x_{1,k} + e_{1,k} \\
y_{2,k} &= x_{2,k} + e_{2,k}.
\end{aligned}
$$

- 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:

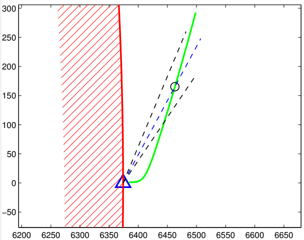
$$\begin{aligned}
\mathbf{x}_k &= \mathbf{A}\,\mathbf{x}_{k-1} + \mathbf{q}_{k-1} \\
\mathbf{y}_k &= \mathbf{H}\,\mathbf{x}_k + \mathbf{r}_k,
\end{aligned}$$

  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

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

- The mean $\mathbf{m}_k$ and covariance $\mathbf{P}_k$ of the posterior distribution can be computed by the Kalman filter.

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.
- The radar measurement:

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

  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:

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

  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.

Summary

- Linear regression problem can be solved as batch problem or recursively – the latter solution is a special case of Kalman filter.
- A generic Bayesian estimation problem can also be solved as batch problem or recursively.
- If we let the linear regression parameter change between the measurements, we get a simple linear state space model – again solvable with Kalman filtering model.
- By generalizing this idea and the solution we get the Kalman filter algorithm.
- By further generalizing to non-Gaussian models results in generic probabilistic state space models.
- Bayesian filtering and smoothing methods solve Bayesian inference problems on state space models recursively.