COMP 64101
Reasoning and Learning under Uncertainty

Omar Rivasplata

Department of Computer Science, University of Manchester
Manchester Centre for AI Fundamentals

Lecture 3 - Part 2

Topics: Gaussian filtering and smoothing

I know it's [Tuesday]. It's a good day for math!

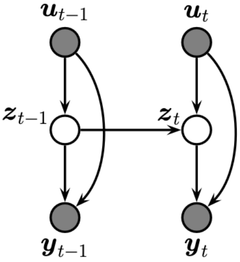
–Inspired by Max Mintz, UPenn.

- Intro, state-space models (§ 8.1)
- Inference for linear-Gaussian SSMs (§ 8.2)
- Inference based on local linearization (§ 8.3)
- More goodies on this (§ 8.4 – 8.7)

Nota bene: Section numbers refer to Murphy (2023).

8.1 Introduction: State Space Models (SSMs)

- Variables:
  - $\mathbf{z}_t$ are hidden (a.k.a. latent),
  - $\mathbf{y}_t$ are observations or outputs,
  - $\mathbf{u}_t$ are optional inputs.
- "Latent variable sequence models."



- CI properties given by the chain-structured graph (see figure).
- Joint distribution:

$$p(\mathbf{y}_{1:T}, \mathbf{z}_{1:T} | \mathbf{u}_{1:T}) = \left[p(\mathbf{z}_1 | \mathbf{u}_1) \prod_{t=2}^{T} p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_t) \right] \left[\prod_{t=1}^{T} p(\mathbf{y}_t | \mathbf{z}_t, \mathbf{u}_t) \right]$$

8.1.1 Inferential goals

- Posterior inference about the hidden states; a.k.a. **state estimation**.

- **Filtering:** Estimate $p(\mathbf{z}_t|\mathbf{y}_{1:t})$, sequential data.
- **Smoothing:** Estimate $p(\mathbf{z}_t|\mathbf{y}_{1:T})$, using offline data.

- **To do:** Read and learn about the various kinds of SSMs, inc. fixed-lag and fixed-interval smoothing (Murphy (2023), see Fig. 8.3 on p. 361)

8.1.2 Bayesian filtering equations

- **Bayes filter:** Algorithm for recursively computing belies $p(\mathbf{z}_t|\mathbf{y}_{1:t})$.
  - Sequential Bayesian updating, online data stream.

- **Prediction step:** $p(\mathbf{z}_t|\mathbf{y}_{1:t-1}) = \int p(\mathbf{z}_t|\mathbf{z}_{t-1})p(\mathbf{z}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{z}_{t-1}$
  - This is the Chapman-Kolmogorov equation.
  - computes the one-step-ahead for the latent state (hidden state).

- **Update step:** $p(\mathbf{z}_t|\mathbf{y}_{1:t}) = \frac{1}{Z_t}p(\mathbf{y}_t|\mathbf{z}_t)p(\mathbf{z}_t|\mathbf{y}_{1:t-1})$
  - This is just an application of Bayes' rule.
  - Z_t is normalising factor: $Z_t = \int p(\mathbf{y}_t|\mathbf{z}_t)p(\mathbf{z}_t|\mathbf{y}_{1:t-1})d\mathbf{z}_t$.

- **Note:** Can use Z_t compute the log-likelihood of the observations.

8.1.3 Bayesian smoothing equations

- Computing $p(\mathbf{z}_t|\mathbf{y}_{1:T})$, belief about the hidden state at time t given all the data, both past and future.
  - Fixed-interval smoothing, assumes offline data $\mathbf{y}_{1:T}$ are available.

- **Forward pass:** Filtering.
  - Just as before, for time steps from 1 to T.
  - Get $p(\mathbf{z}_t|\mathbf{y}_{1:t})$ from $p(\mathbf{z}_{t-1}|\mathbf{y}_{1:t-1})$, for $t = 1, \ldots, T$.

- **Backward pass:** Smoothing.
  - Backwards induction: Get $p(\mathbf{z}_{t+1}|\mathbf{y}_{1:T})$, then $p(\mathbf{z}_t|\mathbf{y}_{1:T})$ and so on.
  - Uses **joint smoothed distributions** over two consecutive time steps: $p(\mathbf{z}_t, \mathbf{z}_{t+1}|\mathbf{y}_{1:T}) = p(\mathbf{z}_t|\mathbf{z}_{t+1}, \mathbf{y}_{1:T})p(\mathbf{z}_{t+1}|\mathbf{y}_{1:T})$

- Method called **forwards filtering backwards smoothing** (FFBS).

8.1.4 The Gaussian ansatz

- Bayesian filtering and smoothing are computationally expensive.
- Their implementation needs integrals that are intractable in general.
- Two notable exceptions:
  - The state space is discrete (e.g. HMMs, to see in § 9.2).
  - The model is linear-Gaussian (e.g. Kalman Filter, to see next).
- Additive Gaussian noise:
  - $\mathbf{z}_t = f(\mathbf{z}_{t-1}, \mathbf{u}_t) + \mathcal{N}(\mathbf{0}, \mathbf{Q}_t)$
  - $\mathbf{y}_t = g(\mathbf{z}_t, \mathbf{u}_t) + \mathcal{N}(\mathbf{0}, \mathbf{R}_t)$
- The liner-Gaussian model:
  - Functions f and g are linear.
  - Noise is Gaussian, as above.

8.2 Inference for linear-Gaussian SSMs

- A linear-Gaussian SSM (or LG-SSM for short) has the following form:

$$p(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{u}_t) = \mathbf{F}_t\mathbf{z}_{t-1} + \mathbf{B}_t\mathbf{u}_t + \mathbf{b}_t + \mathcal{N}(\mathbf{z}_t|\mathbf{0}, \mathbf{Q}_t)$$
$$p(\mathbf{y}_t|\mathbf{z}_t, \mathbf{u}_t) = \mathbf{H}_t\mathbf{z}_t + \mathbf{D}_t\mathbf{u}_t + \mathbf{d}_t + \mathcal{N}(\mathbf{y}_t|\mathbf{0}, \mathbf{R}_t)$$

- Equivalently (as in Murphy (2023), p. 363):

$$p(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{u}_t) = \mathcal{N}(\mathbf{z}_t|\mathbf{F}_t\mathbf{z}_{t-1} + \mathbf{B}_t\mathbf{u}_t + \mathbf{b}_t, \mathbf{Q}_t)$$
$$p(\mathbf{y}_t|\mathbf{z}_t, \mathbf{u}_t) = \mathcal{N}(\mathbf{y}_t|\mathbf{H}_t\mathbf{z}_t + \mathbf{D}_t\mathbf{u}_t + \mathbf{d}_t, \mathbf{R}_t)$$

- Matrices $\mathbf{F}_t$, $\mathbf{B}_t$, $\mathbf{H}_t$, $\mathbf{D}_t$ and vectors $\mathbf{b}_t, \mathbf{d}_t$ make up the linear model.
- Covariance matrices $\mathbf{Q}_t$ and $\mathbf{R}_t$ define the Gaussian noise.

- Special case of a Gaussian Bayes net (details in § 4.2.3).

8.2.1.1 Example: Tracking and state estimation

- **To do:** Read this example (Murphy (2023), pp. 364 - 365).

8.2.2 The Kalman filter

- Algorithm for exact Bayesian filtering for linear-Gaussian SSMs.
- Sequential (online) setting: Computes $p(\mathbf{z}_t|\mathbf{y}_{1:t})$ for each t.
- Can perform the prediction and update steps in closed form!

- Named after its inventor: Rudolf Emil Kálmán (a.k.a. Rudy Kalman).

8.2.2.1 The Kalman filter: Prediction step

- a.k.a. **time update step**.
- This is the one-step-ahead prediction for the hidden state:

$$p(\mathbf{z}_t|\mathbf{y}_{1:t-1}, \mathbf{u}_{1:t}) = \mathcal{N}(\mathbf{z}_t|\boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1})$$

$$\boldsymbol{\mu}_{t|t-1} = \mathbf{F}_t \boldsymbol{\mu}_{t-1|t-1} + \mathbf{B}_t \mathbf{u}_t + \mathbf{b}_t$$

$$\boldsymbol{\Sigma}_{t|t-1} = \mathbf{F}_t \boldsymbol{\Sigma}_{t-1|t-1} \mathbf{F}_t^\top + \mathbf{Q}_t$$

- Notation:
  - $\boldsymbol{\mu}_{t|t'}$ denotes the posterior mean for time t, given $\mathbf{y}_{1:t'}$
  - $\boldsymbol{\Sigma}_{t|t'}$ denotes the posterior covariance for time t, given $\mathbf{y}_{1:t'}$

- **N.B.** There are other notations (Murphy (2023), footnote on p. 365).

8.2.2.2 The Kalman filter: Update step

- a.k.a. **measurement update step**.
- Computed using Bayes' rule, as follows:

$$p(\mathbf{z}_t|\mathbf{y}_{1:t}, \mathbf{u}_{1:t}) = \mathcal{N}(\mathbf{z}_t|\boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t})$$

$$\begin{aligned}
\hat{\mathbf{y}}_t &= \mathbf{H}_t\boldsymbol{\mu}_{t|t-1} + \mathbf{D}_t\mathbf{u}_t + \mathbf{d}_t \\
\mathbf{S}_t &= \mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^\top + \mathbf{R}_t \\
\mathbf{K}_t &= \boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^\top\mathbf{S}_t^{-1} \\
\boldsymbol{\mu}_{t|t} &= \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \hat{\mathbf{y}}_t) \\
\boldsymbol{\Sigma}_{t|t} &= \boldsymbol{\Sigma}_{t|t-1} - \mathbf{K}_t\mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1} \\
&= \boldsymbol{\Sigma}_{t|t-1} - \mathbf{K}_t\mathbf{S}_t\mathbf{K}_t^\top
\end{aligned}$$

- $\mathbf{K}_t$ is called the **Kalman gain matrix**.
- $\mathbf{e}_t = \mathbf{y}_t - \hat{\mathbf{y}}_t$ is the **residual error** or **innovation term**.

8.2.2.3 The Kalman filter: Posterior predictive

- Compute the one-step-ahead predictive density for latent states:

$$\begin{aligned} p(\mathbf{z}_t|\mathbf{y}_{1:t-1}) &= \int p(\mathbf{z}_t|\mathbf{z}_{t-1})p(\mathbf{z}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{z}_{t-1} \\ &= \mathcal{N}(\mathbf{z}_t|\mathbf{F}_t\boldsymbol{\mu}_{t-1|t-1}, \mathbf{F}_t\boldsymbol{\Sigma}_{t-1|t-1}\mathbf{F}_t^\top + \mathbf{Q}_t) \end{aligned}$$

- Convert to a prediction about observations by marginalizing out $\mathbf{z}_t$:

$$\begin{aligned} p(\mathbf{y}_t|\mathbf{y}_{1:t-1}) &= \int p(\mathbf{y}_t, \mathbf{z}_t|\mathbf{y}_{1:t-1})d\mathbf{z}_t \\ &= \int p(\mathbf{y}_t|\mathbf{z}_t)p(\mathbf{z}_t|\mathbf{y}_{1:t-1})d\mathbf{z}_t = \mathcal{N}(\mathbf{y}_t|\hat{\mathbf{y}}_t, \mathbf{S}_t) \end{aligned}$$

- **Note:** Can also compute the log-likelihood of the observations.

The Kalman filter: more goodies

- 8.2.2.4 Derivation
  - **To do:** Read about this (Murphy (2023), pp. 367 - 368).
- 8.2.2.5 Abstract formulation
  - **To do:** Read about this (Murphy (2023), pp. 368 - 369).
- 8.2.2.6 Numerical issues
  - **To do:** Read about this (Murphy (2023), p. 370).
- 8.2.2.7 Continuous-time version (Kalman-Bucy filter)
  - **To do:** Read about this (Murphy (2023), p. 370).

- See Algorithm 8.1 (LG system) and Algorithm 8.2 (Kalman filter).

8.2.3 The Kalman smoother (RTS smoother)

- Offline settings: Computes $p(\mathbf{z}_t|\mathbf{y}_{1:T})$ based on whole stream $\mathbf{y}_{1:T}$.
- For settings where we can wait until all the data $\mathbf{y}_{1:T}$ have arrived.
- Still can write distributions in closed form!

- Name: RTS stands for Rauch, Tung, and Striebel.

- RTSS = Rauch, Tung, and Striebel smoother.

8.2.3.1 The Kalman smoother: Algorithm

- The key update equations are as follows:

$$p(\mathbf{z}_t|\mathbf{y}_{1:T}) = \mathcal{N}(\mathbf{z}_t|\boldsymbol{\mu}_{t|T}, \boldsymbol{\Sigma}_{t|T})$$

$$\begin{aligned}
\boldsymbol{\mu}_{t+1|t} &= \mathbf{F}_t\boldsymbol{\mu}_{t|t} \\
\boldsymbol{\Sigma}_{t+1|t} &= \mathbf{F}_t\boldsymbol{\Sigma}_{t|t}\mathbf{F}_t^\top + \mathbf{Q}_{t+1} \\
\mathbf{J}_t &= \boldsymbol{\Sigma}_{t|t}\mathbf{F}_t^\top\boldsymbol{\Sigma}_{t+1|t}^{-1} \\
\boldsymbol{\mu}_{t|T} &= \boldsymbol{\mu}_{t|t} + \mathbf{J}_t\left(\boldsymbol{\mu}_{t+1|T} - \boldsymbol{\mu}_{t+1|t}\right) \\
\boldsymbol{\Sigma}_{t|T} &= \boldsymbol{\Sigma}_{t|t} + \mathbf{J}_t\left(\boldsymbol{\Sigma}_{t+1|T} - \boldsymbol{\Sigma}_{t+1|t}\right)\mathbf{J}_t^\top
\end{aligned}$$

The Kalman smoother: more goodies

- 8.2.3.2 Derivation
  - **To do:** Read about this (Murphy (2023), pp. 370 - 371).
- 8.2.3.3 Two-filter smoothing
  - **To do:** Read about this (Murphy (2023), p. 372).
- 8.2.3.4 Time and space complexity
  - **To do:** Read about this (Murphy (2023), p. 372).
- 8.2.3.4 Time and space complexity
  - **To do:** Read about this (Murphy (2023), p. 370).
- 8.2.3.5 Forwards filtering backwards sampling
  - **To do:** Read about this (Murphy (2023), p. 372).
- 8.2.4 Information form filtering and smoothing
  - **To do:** Read about this (Murphy (2023), pp. 372 - 375).

8.3 Inference based on local linearization

- Aim to extend the Kalman filter and smoother to the case where the system dynamics and/or the observation model are nonlinear.
- Continue to assume that the noise is additive Gaussian.
- Idea: Linearize the dynamics and observation models about the previous state estimate using a first order Taylor series expansion, and then to apply the standard Kalman filter equations.

- Intuition: approximating a stationary non-linear dynamical system with a non-stationary linear dynamical system.
- This approach is called the **extended Kalman filter** (EKF).

8.3.1 [Linearization]

- Let $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.
- Suppose $\mathbf{y} = g(\mathbf{x})$ with $g : \mathbb{R}^n \to \mathbb{R}^n$ differentiable, invertible.
- The PDF for $\mathbf{y}$ is (by the transformation formula):

$$p(\mathbf{y}) = |\det J(g^{-1})(\mathbf{y})|\, \mathcal{N}(g^{-1}(\mathbf{y})|\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Write $\mathbf{x} = \boldsymbol{\mu} + \boldsymbol{\delta}$ with $\boldsymbol{\delta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$.
- First-order Taylor series expansion of g around $\boldsymbol{\mu}$ gives:

$$g(\mathbf{x}) \approx g(\boldsymbol{\mu}) + J(g)(\boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu}) = g(\boldsymbol{\mu}) + J(g)(\boldsymbol{\mu})\boldsymbol{\delta}$$

  where $J(g)$ is the Jacobian of g, an $n \times n$ matrix.

- Can use this for approximating $\mathbb{E}[g(\mathbf{x})]$ and $\mathrm{Cov}(g(\mathbf{x}))$.
- See Algorithm 8.3 (Linear approximation to a joint Gaussian).

More goodies

- 8.3.2 The extended Kalman filter (EKF)
- 8.3.3 The extended Kalman smoother (EKS)
- 8.4 Inference based on the unscented transform
  - The unscented Kalman filter (UKF)
  - The unscented Kalman smoother (UKS)
- 8.5 Other variants of the Kalman filter
  - 8.5.1 General Gaussian filtering (and sub-cases)
  - 8.5.2 Conditional moment Gaussian filtering
  - 8.5.3 Iterated filters and smoothers
  - 8.5.4 Ensemble Kalman filter
  - 8.5.5 Robust Kalman filters
  - 8.5.6 Dual EKF
- And even more (§ 8.6, § 8.7).

References

Kevin P. Murphy. *Probabilistic Machine Learning: Advanced Topics*. MIT Press, 2023. URL `http://probml.github.io/book2`.