Probabilistic Time Series Analysis: Lab 3

Tim Kunisky

September 19, 2018

The Linear Gaussian Hidden Markov Model

The Linear Gaussian Hidden Markov Model

- High level: the Kalman filter is an *algorithm for computing certain conditional distributions* for certain time series models. It is not a model! It is not a way to do inference!

The Linear Gaussian Hidden Markov Model

- High level: the Kalman filter is an *algorithm for computing certain conditional distributions* for certain time series models. It is not a model! It is not a way to do inference!
- Let's review the model that we covered Kalman filtering for:

$$
\begin{array}{ccccccccc}
\cdots & \rightarrow & \boldsymbol{z}_{t-2} & \rightarrow & \boldsymbol{z}_{t-1} & \rightarrow & \boldsymbol{z}_t & \rightarrow & \cdots \\
 & & \downarrow & & \downarrow & & \downarrow & & \\
\cdots & \rightarrow & \boldsymbol{x}_{t-2} & \rightarrow & \boldsymbol{x}_{t-1} & \rightarrow & \boldsymbol{x}_t & \rightarrow & \cdots
\end{array}
$$

$$
\begin{aligned}
\boldsymbol{z}_t &= \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{w}_t \\
\boldsymbol{x}_t &= \boldsymbol{C}\boldsymbol{z}_t + \boldsymbol{v}_t \\
\boldsymbol{w}_t &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}) \\
\boldsymbol{v}_t &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}) \\
\boldsymbol{z}_0 &\sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma})
\end{aligned}
$$

Inference vs. Filtering vs. Smoothing

Inference vs. Filtering vs. Smoothing

- **Question:** Describe the goal of the inference task for this model.

Inference vs. Filtering vs. Smoothing

- **Question:** Describe the goal of the inference task for this model.
  - Kalman filtering will not do this for you! This is what we will use the EM algorithm for.

Inference vs. Filtering vs. Smoothing

- **Question:** Describe the goal of the inference task for this model.
    - Kalman filtering will not do this for you! This is what we will use the EM algorithm for.
- Once we know the parameters $\boldsymbol{A}, \boldsymbol{C}, \boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{\mu}_0, \Sigma$, we still need the distributions of the unobserved latent states $\boldsymbol{z}_t$.

Inference vs. Filtering vs. Smoothing

- **Question:** Describe the goal of the inference task for this model.
  - Kalman filtering will not do this for you! This is what we will use the EM algorithm for.
- Once we know the parameters $A, C, Q, R, \mu_0, \Sigma$, we still need the distributions of the unobserved latent states z_t.
- There are two basic settings:

Inference vs. Filtering vs. Smoothing

- **Question:** Describe the goal of the inference task for this model.
  - Kalman filtering will not do this for you! This is what we will use the EM algorithm for.
- Once we know the parameters $A, C, Q, R, \mu_0, \Sigma$, we still need the distributions of the unobserved latent states z_t.
- There are two basic settings:
  - **Filtering:** You have observations of $x_1, \ldots, x_t$, and want to compute the distribution of z_t, e.g. if you are trying to estimate something about a physical system "live" or "online."

Inference vs. Filtering vs. Smoothing

- **Question:** Describe the goal of the inference task for this model.
  - Kalman filtering will not do this for you! This is what we will use the EM algorithm for.
- Once we know the parameters $\boldsymbol{A}, \boldsymbol{C}, \boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{\mu}_0, \Sigma$, we still need the distributions of the unobserved latent states $\boldsymbol{z}_t$.
- There are two basic settings:
  - **Filtering:** You have observations of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t$, and want to compute the distribution of $\boldsymbol{z}_t$, e.g. if you are trying to estimate something about a physical system "live" or "online."
  - **Smoothing:** You have observations of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$, and want to compute the distribution of $\boldsymbol{z}_t$ with $t < T$, e.g. if you have gathered an entire time series of observations and want to estimate the latent states afterwards.

Filtering

Filtering

- **Goal:** $\mathbb{P}[\boldsymbol{z}_n \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \sim \mathcal{N}(\boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n})$, compute RHS parameters for all n.

Filtering

- **Goal:** $\mathbb{P}[\boldsymbol{z}_n \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \sim \mathcal{N}(\boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n})$, compute RHS parameters for all n.
- **Prediction Step:** $\mathbb{P}[\boldsymbol{z}_n \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n-1}] \sim \mathcal{N}(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1})$.

$$\begin{aligned}
\boldsymbol{\mu}_{n|n-1} &= \boldsymbol{A}\boldsymbol{\mu}_{n-1|n-1} \\
\boldsymbol{\Sigma}_{n|n-1} &= \boldsymbol{A}\boldsymbol{\Sigma}_{n-1|n-1}\boldsymbol{A}^\top + \boldsymbol{Q}
\end{aligned}$$

Filtering

- **Goal:** $\mathbb{P}[\boldsymbol{z}_n \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \sim \mathcal{N}(\boldsymbol{\mu}_{n|n}, \Sigma_{n|n})$, compute RHS parameters for all n.
- **Prediction Step:** $\mathbb{P}[\boldsymbol{z}_n \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n-1}] \sim \mathcal{N}(\boldsymbol{\mu}_{n|n-1}, \Sigma_{n|n-1})$.

$$
\begin{aligned}
\boldsymbol{\mu}_{n|n-1} &= \boldsymbol{A}\boldsymbol{\mu}_{n-1|n-1} \\
\Sigma_{n|n-1} &= \boldsymbol{A}\Sigma_{n-1|n-1}\boldsymbol{A}^\top + \boldsymbol{Q}
\end{aligned}
$$

- **Observation Step:** Add conditioning on $\boldsymbol{x}_n$.

$$
\begin{aligned}
\tilde{\boldsymbol{x}}_n &= \boldsymbol{x}_n - \boldsymbol{C}\boldsymbol{\mu}_{n|n-1} \\
\tilde{\boldsymbol{R}}_n &= \boldsymbol{C}\Sigma_{n|n-1}\boldsymbol{C}^\top + \boldsymbol{R} \\
\boldsymbol{K}_n &= \Sigma_{n|n-1}\boldsymbol{C}^\top \tilde{\boldsymbol{R}}_n^{-1} \\
\boldsymbol{\mu}_{n|n} &= \boldsymbol{\mu}_{n|n-1} + \boldsymbol{K}_n \tilde{\boldsymbol{x}}_n \\
\Sigma_{n|n} &= (\boldsymbol{I} - \boldsymbol{K}_n\boldsymbol{C})\Sigma_{n|n-1}
\end{aligned}
$$

Smoothing

Smoothing

- **Goal:** $\mathbb{P}[\boldsymbol{z}_n \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n)$, compute RHS parameters, assuming you know them for all smaller n.

Smoothing

- **Goal:** $\mathbb{P}[\boldsymbol{z}_n \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n)$, compute RHS parameters, assuming you know them for all smaller n.
- **Forward Path:** Run filtering on the entire sequence, from $n = 0$ to $n = N$.

Smoothing

- **Goal:** $\mathbb{P}[\boldsymbol{z}_n \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_n, \hat{\Sigma}_n)$, compute RHS parameters, assuming you know them for all smaller n.
- **Forward Path:** Run filtering on the entire sequence, from $n = 0$ to $n = N$.
- **Backward Path:** Step backwards from N to 1, making corrections to the filtering predictions iteratively. At N, the filtering and smoothing predictions match, and we propagate the "knowledge of the full series" backwards through the predictions.

$$
\begin{aligned}
\hat{\boldsymbol{\mu}}_N &= \boldsymbol{\mu}_{N|N} \\
\hat{\Sigma}_N &= \Sigma_{N|N} \\
\boldsymbol{F}_n &= \Sigma_{n|n} \boldsymbol{A}^\top \Sigma_{n+1|n}^{-1} \\
\hat{\boldsymbol{\mu}}_n &= \boldsymbol{\mu}_{n|n} + \boldsymbol{F}_n (\hat{\boldsymbol{\mu}}_{n+1} - \boldsymbol{A}\boldsymbol{\mu}_{n|n}) \\
\hat{\Sigma}_n &= \Sigma_{n|n} + \boldsymbol{F}_n (\hat{\Sigma}_{n+1} - \Sigma_{n+1|n}) \boldsymbol{F}_n^\top
\end{aligned}
$$

Inference: Review of the EM Algorithm

Inference: Review of the EM Algorithm

- This is a way of doing *maximum likelihood*, maximizing

$$L(\boldsymbol{\theta}) = \log \mathbb{P}[\boldsymbol{x} \mid \boldsymbol{\theta}] = \log \int_z \mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}] d\boldsymbol{z}.$$

Inference: Review of the EM Algorithm

- This is a way of doing *maximum likelihood*, maximizing

$$L(\boldsymbol{\theta}) = \log \mathbb{P}[\boldsymbol{x} \mid \boldsymbol{\theta}] = \log \int_{z} \mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}] d\boldsymbol{z}.$$

- The idea is to introduce a distribution over $\boldsymbol{z}$, called $\mathbb{Q}$, and take an inequality to make the optimization easier:

Inference: Review of the EM Algorithm

- This is a way of doing *maximum likelihood*, maximizing

$$L(\boldsymbol{\theta}) = \log \mathbb{P}[\boldsymbol{x} \mid \boldsymbol{\theta}] = \log \int_z \mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}] d\boldsymbol{z}.$$

- The idea is to introduce a distribution over $\boldsymbol{z}$, called $\mathbb{Q}$, and take an inequality to make the optimization easier:

$$\begin{aligned} L(\boldsymbol{\theta}) &= \log \int_z \mathbb{Q}[\boldsymbol{z}] \frac{\mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}]}{\mathbb{Q}[\boldsymbol{z}]} d\boldsymbol{z} \\ &= \log \mathbb{E}_{\mathbb{Q}} \left[\frac{\mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}]}{\mathbb{Q}[\boldsymbol{z}]} \right] \\ &\geq \mathbb{E}_{\mathbb{Q}} \left[\log \frac{\mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}]}{\mathbb{Q}[\boldsymbol{z}]} \right] \\ &= \mathbb{E}_{\mathbb{Q}} \left[\log \mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}] \right] - \mathbb{E}_{\mathbb{Q}} \left[\log \mathbb{Q}[\boldsymbol{z}] \right]. \end{aligned}$$

Inference: Review of the EM Algorithm

- Why would this be a good idea??

Inference: Review of the EM Algorithm

- Why would this be a good idea??
- When is there approximate equality in Jensen's inequality:

$$\log \mathbb{E}_{\mathbb{Q}}[f(\boldsymbol{z})] \approx \mathbb{E}_{\mathbb{Q}}[\log f(\boldsymbol{z})]?$$

Inference: Review of the EM Algorithm

- Why would this be a good idea??
- When is there approximate equality in Jensen's inequality:

$$\log \mathbb{E}_{\mathbb{Q}}[f(\boldsymbol{z})] \approx \mathbb{E}_{\mathbb{Q}}[\log f(\boldsymbol{z})]?$$

- **When $f(\boldsymbol{z}) \approx$ constant.**

Inference: Review of the EM Algorithm

- Why would this be a good idea??
- When is there approximate equality in Jensen's inequality:

$$\log \mathbb{E}_{\mathbb{Q}}[f(\boldsymbol{z})] \approx \mathbb{E}_{\mathbb{Q}}[\log f(\boldsymbol{z})]?$$

- **When $f(\boldsymbol{z}) \approx$ constant.**
- What does this mean for us? We want

$$\frac{\mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}]}{\mathbb{Q}[\boldsymbol{z}]} \approx c(\boldsymbol{x}, \boldsymbol{\theta})$$

Inference: Review of the EM Algorithm

- Why would this be a good idea??
- When is there approximate equality in Jensen's inequality:

$$\log \mathbb{E}_{\mathbb{Q}}[f(\boldsymbol{z})] \approx \mathbb{E}_{\mathbb{Q}}[\log f(\boldsymbol{z})]?$$

- **When $f(\boldsymbol{z}) \approx$ constant.**
- What does this mean for us? We want

$$\frac{\mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}]}{\mathbb{Q}[\boldsymbol{z}]} \approx c(\boldsymbol{x}, \boldsymbol{\theta})$$

or...

$$\mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}] \approx \mathbb{Q}[\boldsymbol{z}] c(\boldsymbol{x}, \boldsymbol{\theta}),$$

so the likelihood should approximately factorize!

Inference: Review of the EM Algorithm

- Why would this be a good idea??
- When is there approximate equality in Jensen's inequality:

$$\log \mathbb{E}_{\mathbb{Q}}[f(\boldsymbol{z})] \approx \mathbb{E}_{\mathbb{Q}}[\log f(\boldsymbol{z})]?$$

- **When $f(\boldsymbol{z}) \approx$ constant.**
- What does this mean for us? We want

$$\frac{\mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}]}{\mathbb{Q}[\boldsymbol{z}]} \approx c(\boldsymbol{x}, \boldsymbol{\theta})$$

  or...

$$\mathbb{P}[\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}] \approx \mathbb{Q}[\boldsymbol{z}]c(\boldsymbol{x}, \boldsymbol{\theta}),$$

  so the likelihood should approximately factorize!
- **General principle:** The EM algorithm encodes an approximation of the likelihood as factorizing, and optimizes over this approximation.

Coding Assignment

- Find `labs/lab3/lab3-student.ipynb` on the course Github. (It will be easier if you clone the whole repository.)
- Try running everything, but you will only be graded on what you fill in for the missing bits of code marked with a TODO. This time, these are all methods of one class for doing sampling, filtering, and smoothing for the model we talked about.