Robot Learning and Control 2

Linear Gaussian State Space Models

Takamitsu Matsubara[1]

[1] Robot Learning Lab

Nov. 15th 2024

Table of Contents

Today's notebook

Python notebooks using goolge colab (made by AP Sasaki) will be provided to help participants understanding the course.

https://colab.research.google.com/drive/1cjF42mssLJ6jONqzDOFmO4UkLouGq1QS?usp=sharing

Table of Contents

Dynamics in Real-world Robots

Dynamics:

- forces that produces movement
- the branch of mechanics concerned with the motion of bodies under the action of forces
- $\mathrm{state}_{t+1} = \mathit{Dynamics}(\mathrm{state}_t, \mathrm{action}_t)$
- $p(\mathrm{state}_{t+1} | \mathrm{state}_t, \mathrm{action}_t)$

Linear Gaussian Dynamics

The next-time state is deviated by the linear mapping from current state and zero-mean Gaussian noise:

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{w}_t \quad (1)$$

$$\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}), \mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \sigma_0^2\mathbf{I}) \quad (2)$$

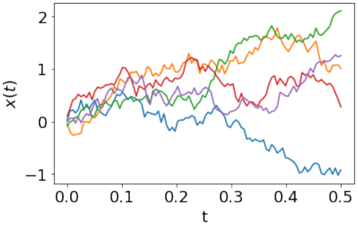


Figure: State trajectory with $A = 1$, $\sigma^2 = 0.1$, $\sigma_0^2 \approx 0$

Diversity of state expands over time → prediction becomes harder

Let's confirm this point in the model

Linear Gaussian Dynamics

Given the initial state distribution and linear Guassian dynamics,

$$p(\mathbf{x}_1) = \mathcal{N}(\mathbf{x}_1; \boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1}) \tag{3}$$

$$p(\mathbf{x}_2 \mid \mathbf{x}_1) = \mathcal{N}(\mathbf{x}_2; \mathbf{A}\mathbf{x}_1, \sigma^2\mathbf{I}) \tag{4}$$

One-step predictive state distribution is obtained as follows:

$$p(\mathbf{x}_2) = \int p(\mathbf{x}_2|\mathbf{x}_1)p(\mathbf{x}_1)\mathbf{x}_1 \quad \text{(marginalization)} \tag{5}$$

$$= \mathcal{N}(\mathbf{x}_2; \mathbf{A}\boldsymbol{\mu}, \underbrace{\sigma^2\mathbf{I}}_{\text{process}} + \underbrace{\mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{A}^T}_{\text{dynamics}}) \tag{6}$$

Both uncertainties in initial state and dynamics are marged, and it can be recursively applied over the time!
If $A \geq 1$, the uncertainty is expanded and expanded over the time!

Table of Contents

What is LGSSM

Linear Gaussian State Space Model: probabilistic dynamics with noisy observations, e.g.,

- locations of moving vehicle with GPS
- robot joint angles and angular velocities with encoders

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{w}_t \quad \text{(State transition model)} \tag{7}$$

$$\mathbf{y}_t = \mathbf{B}\mathbf{x}_t + \mathbf{v}_t \quad \text{(Observation model)} \tag{8}$$

where $\mathbf{x}$ is latent state variable, $\mathbf{y}$ is observed variable, $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$, $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ are process and observation noises. $\mathbf{Q}$ and $\mathbf{R}$ are symmetric positive definite matrices (covariance of Gaussian). Thus, the model parameter can be summarized as $\theta = \{\mathbf{A}, \mathbf{B}, \mathbf{Q}, \mathbf{R}, \mathbf{V}_0\}$

It can also be represented as:

$$p(\mathbf{x}_1) = \mathcal{N}(\mathbf{x}_1 \mid \boldsymbol{\mu}_0, \mathbf{V}_0) \tag{9}$$

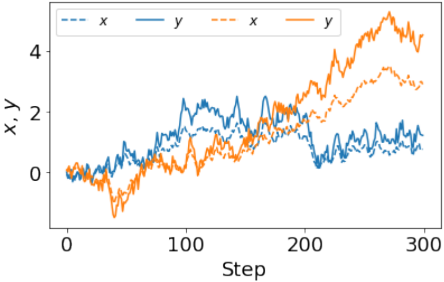
$$p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t \mid \mathbf{A}\mathbf{x}_{t-1}, \mathbf{Q}) \tag{10}$$

$$p(\mathbf{y}_t \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{y}_t \mid \mathbf{B}\mathbf{x}_t, \mathbf{R}) \tag{11}$$

Plot of LGSSM

Simulation setup:

- $\mu_0 = 0, V_0 = 0.1$
- $A = 1, Q = 0.1$
- $B = 1.5, R = 0.1$



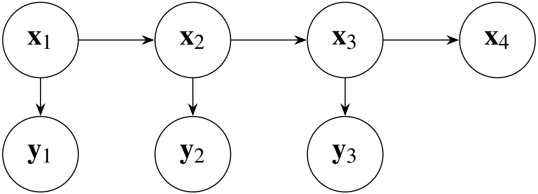
Some difference between x and y depending on the value of B and noises.

Table of Contents

Three State Estimation Problems in LGSSM

Estimate state variables given observations with a LGSSM model.

- **Prediction:** estimate future state given past observations
  $p(\mathbf{x}_k \mid \mathbf{y}_{1:t-1}), (k \geq t)$
- **Filtering:** estimate current state given until current observations
  $p(\mathbf{x}_t \mid \mathbf{y}_{1:t})$
- **Smoothing:** estimate past state given until current observations
  $p(\mathbf{x}_t \mid \mathbf{y}_{1:k}), (k \geq t)$

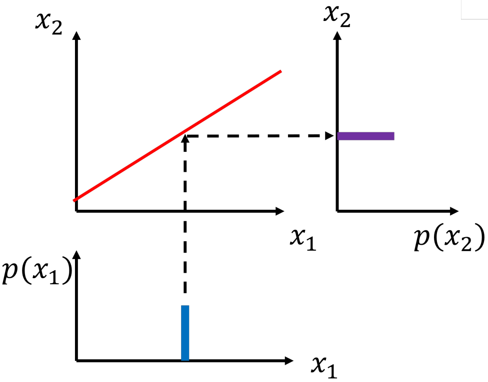


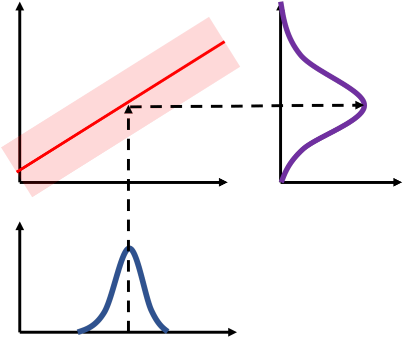
All problems' solutions can be derived using **Bayes rules of Gaussians!**

Recap: Prediction of Linear Gaussians

$$\boldsymbol{x}_2 = \boldsymbol{A}\boldsymbol{x}_1$$

$$p(\mathbf{x}_2) = \int p(\mathbf{x}_2|\mathbf{x}_1)p(\mathbf{x}_1)\mathbf{x}_1$$

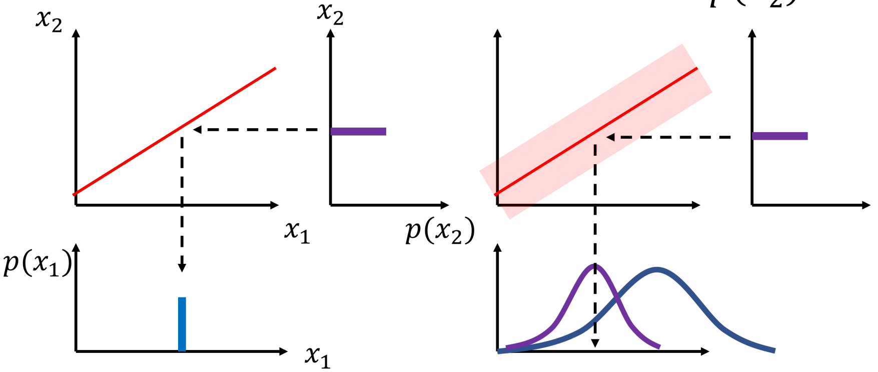




Recap: Bayes Posterior of Linear Gaussians

$$\boldsymbol{x}_1 = \boldsymbol{A}^{-1}\boldsymbol{x}_2$$

$$\boldsymbol{p}(\boldsymbol{x}_1|\boldsymbol{x}_2) = \frac{p(\boldsymbol{x}_2|\boldsymbol{x}_1)p(\boldsymbol{x}_1)}{p(\boldsymbol{x}_2)}$$



Prediction

Prediction: estimate future state given past observations

$$p(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}) = \int \underbrace{p(\mathbf{x}_t \mid \mathbf{x}_{t-1})}_{\text{state transition}} \underbrace{p(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1})}_{\text{(filtered) current state}} d\mathbf{x}_{t-1} \tag{12}$$

If $p(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1})$ follows a Gaussian distribution, through the Bayes rules of LGM, the solution is obtained in the form of Gaussian.

Prediction: algorithm

Given the current state estimation $p(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}) = \mathcal{N}(\mathbf{x}_{t-1} \mid \boldsymbol{\mu}_{t-1}, \mathbf{V}_{t-1})$, one-step ahead prediction is obtained as:

$$p(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}) = \int p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) p(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}) d\mathbf{x}_{t-1} = \mathcal{N}(\mathbf{x}_t; \bar{\boldsymbol{\mu}}_t, \bar{\mathbf{V}}_t) \tag{13}$$

where

$$\bar{\boldsymbol{\mu}}_t = \mathbf{A}\boldsymbol{\mu}_{t-1} \tag{14}$$

$$\bar{\mathbf{V}}_t = \mathbf{Q} + \mathbf{A}\mathbf{V}_{t-1}\mathbf{A}^T \tag{15}$$

The above algorithm can be recursively applied forward in time from the initial state

Exercise 2-1: Derive it by yourself (See eq. 25 in Lecture-1)

Filtering

Filtering: estimate current state given until current observations

$$p(\mathbf{x}_t \mid \mathbf{y}_{1:t}) = \frac{\overbrace{p(\mathbf{y}_t \mid \mathbf{x}_t)}^{\text{observation model}} \; \overbrace{p(\mathbf{x}_t \mid \mathbf{y}_{1:t-1})}^{\text{predicted state}}}{p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1})} \tag{16}$$

$$p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1}) = \int p(\mathbf{y}_t \mid \mathbf{x}_t) p(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}) d\mathbf{x}_t \tag{17}$$

If $p(\mathbf{x}_t \mid \mathbf{y}_{1:t-1})$ follows a Gaussian distribution, through the Bayes theorem of LGM, the solution is obtained in the form of Gaussian.

Filtering: algorithm (Kalman Filter)

Jeff Miller 2016

- Initialize

$$\mathbf{K}_1 = \mathbf{V}_0\mathbf{B}^T(\mathbf{B}\mathbf{V}_0\mathbf{B}^T + \mathbf{R})^{-1} \tag{18}$$

$$\boldsymbol{\mu}_1 = \boldsymbol{\mu}_0 + \mathbf{K}_1(\mathbf{y}_1 - \mathbf{B}\boldsymbol{\mu}_0) \tag{19}$$

$$\mathbf{V}_1 = (\mathbf{I} - \mathbf{K}_1\mathbf{B})\mathbf{V}_0, \quad \mathbf{P}_1 = \underbrace{\mathbf{A}\mathbf{V}_1\mathbf{A}^T + \mathbf{Q}}_{\text{predictive variance}} \tag{20}$$

- For $j = 2, \cdots, T$

$$\mathbf{K}_j = \mathbf{P}_{j-1}\mathbf{B}^T(\mathbf{B}\mathbf{P}_{j-1}\mathbf{B}^T + \mathbf{R})^{-1} \tag{21}$$

$$\boldsymbol{\mu}_j = \underbrace{\mathbf{A}\boldsymbol{\mu}_{j-1}}_{\text{predictive mean}} + \mathbf{K}_j(\mathbf{y}_j - \mathbf{B}\mathbf{A}\boldsymbol{\mu}_{j-1}) \tag{22}$$

$$\mathbf{V}_j = (\mathbf{I} - \mathbf{K}_j\mathbf{B})\mathbf{P}_{j-1}, \quad \mathbf{P}_j = \mathbf{A}\mathbf{V}_j\mathbf{A}^T + \mathbf{Q} \tag{23}$$

Then, $p(\mathbf{x}_j \mid \mathbf{y}_{1:j}) = \mathcal{N}(\mathbf{x}_j; \boldsymbol{\mu}_j, \mathbf{V}_j)$ for all $j = 1, ..., T$.

The above algorithm can be recursively applied **forward** in time from the initial state with prediction algorithm.

Smoothing (Kalman Smoother)

Smoothing: estimate past state given until current observations

$$
\begin{aligned}
p(\mathbf{x}_{t+1}, \mathbf{x}_t \mid \mathbf{y}_{1:t}) &= \overbrace{p(\mathbf{x}_{t+1} \mid \mathbf{x}_t)}^{\text{state transition}} \overbrace{p(\mathbf{x}_t \mid \mathbf{y}_{1:t})}^{\text{filtered}} && (24)\\
p(\mathbf{x}_t \mid \mathbf{x}_{t+1}, \mathbf{y}_{1:t}) &= \frac{p(\mathbf{x}_{t+1}, \mathbf{x}_t \mid \mathbf{y}_{1:t})}{p(\mathbf{x}_{t+1} \mid \mathbf{y}_{1:t})} && (25)\\
p(\mathbf{x}_t \mid \mathbf{x}_{t+1}, \mathbf{y}_{1:T}) &= p(\mathbf{x}_t \mid \mathbf{x}_{t+1}, \mathbf{y}_{1:t}) \ \text{(conditional independence)} && (26)\\
p(\mathbf{x}_t, \mathbf{x}_{t+1} \mid \mathbf{y}_{1:T}) &= p(\mathbf{x}_k \mid \mathbf{x}_{t+1}, \mathbf{y}_{1:T}) \overbrace{p(\mathbf{x}_{t+1} \mid \mathbf{y}_{1:T})}^{\text{previous smoothing}} && (27)\\
p(\mathbf{x}_t \mid \mathbf{y}_{1:T}) &= \int p(\mathbf{x}_t, \mathbf{x}_{t+1} \mid \mathbf{y}_{1:T}) d\mathbf{x}_{t+1} && (28)
\end{aligned}
$$

Smoothing: algorithm

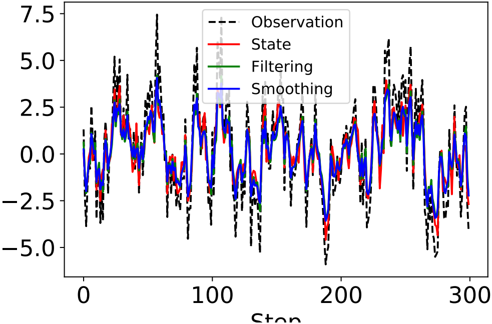
Jeff Miller 2016

- Initialize:
  - Compute Kalman filter algorithm to obtain $\boldsymbol{\mu}_j$,$\mathbf{V}_j$, and $\mathbf{P}_j$
  - $\hat{\boldsymbol{\mu}}_T = \boldsymbol{\mu}_T$, $\hat{\mathbf{V}}_T = \mathbf{V}_T$
- For $j = T-1, .., 1$:

$$
\begin{aligned}
\mathbf{C}_j &= \mathbf{V}_j \mathbf{A}^T \mathbf{P}_j^{-1} && (29)\\
\hat{\boldsymbol{\mu}}_j &= \mu_j + \mathbf{C}_j(\hat{\boldsymbol{\mu}}_{j+1} - \mathbf{A}\boldsymbol{\mu}_j) && (30)\\
\hat{\mathbf{V}}_j &= \mathbf{V}_j + \mathbf{C}_j(\hat{\mathbf{V}}_{j+1} - \mathbf{P}_j)\mathbf{C}_j^T && (31)
\end{aligned}
$$

Then, $p(\mathbf{x}_j \mid \mathbf{y}_{1:T}) = \mathcal{N}(\mathbf{x}_j \mid \hat{\mu}_j, \hat{\mathbf{V}}_j)$ for all $j = 1..T$.
The above algorithm can be recursively applied **backword** in time from the filtering results.

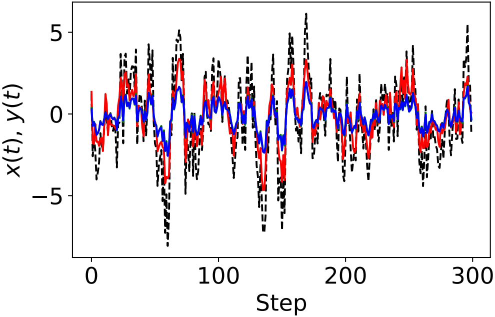
Plot of filtering and smoothing in LGSSM



It works well. The difference between filtering and smoothing are relatively small.

Plot of filtering and smoothing in LGSSM

If the parameters of the LGSSM is wrong, the results become very poor..



How can we set the parameters to fit to the data automatically?

Table of Contents

Parameter estimation problem of LGSSM

If we can observe **both state and observation** $\{\mathbf{Y}, \mathbf{X}\}$, its likehood becomes

$$p(\mathbf{Y}, \mathbf{X}; \theta) = p(\mathbf{x}_1; \theta) p(\mathbf{y}_1 \mid \mathbf{x}_1; \theta) \prod_{t=2}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}; \theta) p(\mathbf{y}_t \mid \mathbf{x}_t; \theta) \tag{32}$$

where θ is the parameter of the models. Then, its parameter estimation problem can be formulated as maximum (log-)likelihood;

$$\theta^* \leftarrow \arg\max_{\theta} \log p(\mathbf{Y}, \mathbf{X}; \theta) \tag{33}$$

This is tractable; it is *supervised learning* similar to linear Gaussian regression!

Parameter estimation problem of LGSSM

If **only observations** $\{\mathbf{Y}\}$ can be obtained as a more realistic setup, the problem becomes more difficult as an *unsupervised learning* problem.

Marginal likelihood (unobserved states $\mathbf{X}$ are marginalized):

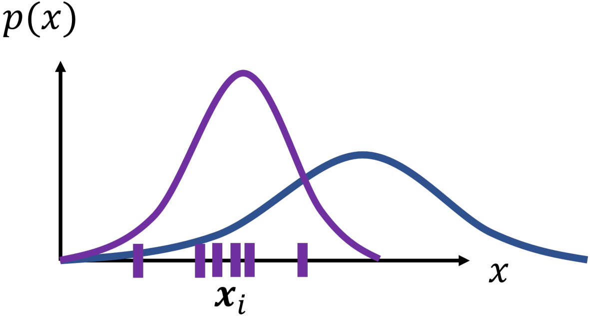
$$p(\mathbf{Y};\theta) = \int p(\mathbf{x}_1;\theta)p(\mathbf{y}_1 \mid \mathbf{x}_1;\theta) \prod_{t=2}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{t-1};\theta)p(\mathbf{y}_t \mid \mathbf{x}_t;\theta)d\mathbf{x}_{1:t} \quad (34)$$

$$\theta^* \leftarrow \arg\max_{\theta} \ln p(\mathbf{Y} \mid \theta) \quad (35)$$

Its optimization cannot be solved easily due to *the integral in log function*. Is there any alternative approaches solvable easiler?

Parameter estimation problem of LGSSM

$$\theta^* \leftarrow \mathrm{argm}ax_\theta \log \prod_i p(\boldsymbol{x}_i;\theta)$$



Fitting the mean and variance parameters of the Gaussian for observed data

Making a Lower Bound of Marginal Likelihood

Zoubin Ghahramani 1996

The marginal likelihood can be decomposed by arbitrary distribution $q(\mathbf{X})$:

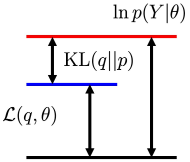
$$\ln p(\mathbf{Y} \mid \theta) = L(\theta, q) + \underbrace{KL(q \,\|\, p)}_{\text{Kullback Leibler divergence}} \tag{36}$$

where

$$L(q, \theta) = \int q(\mathbf{X}) \ln \frac{p(\mathbf{Y}, \mathbf{X} \mid \theta)}{q(\mathbf{X})} d\mathbf{X} \tag{37}$$

$$KL(q \,\|\, p) = -\int q(\mathbf{X}) \ln \frac{p(\mathbf{X} \mid \mathbf{Y}, \theta)}{q(\mathbf{X})} d\mathbf{X} \tag{38}$$

Since KL≥ 0, $L(q, \theta)$ is said as *lowerbound*.



Exercise 2-2, 2-2

2-2 Confirm the derivation of decomposition

2-3 Confirm non-negativity of KL divergence

Expectation-Maximization Algorithm

The lowerbound seems easiler to optimize since it directly applies logarizm function to each probability:

$$\ln p(\mathbf{Y}, \mathbf{X} \mid \theta) = \ln p(\mathbf{Y} \mid \mathbf{X}, \theta) p(\mathbf{X} \mid \theta) = \ln p(\mathbf{Y} \mid \mathbf{X}, \theta) + \ln p(\mathbf{X} \mid \theta) \tag{39}$$

The lowerbound **touches to the marginal likeliood** at $q_k(X) = p(X \mid Y; \theta_k)$ since $KL(q \parallel p)$ becomes 0.
So, if we find a better parameter $\hat{\theta}$ so that $L(q_k(X), \hat{\theta}) > L(q_k(X; \theta), \theta_k)$, it also holds $\ln p(\mathbf{Y}; \hat{\theta}) > \ln p(\mathbf{Y}; \theta_k)$

Thus, the parameter optimization can be formalized by the following alternative update scheme:

- E-step: $q_k(X) = p(X \mid Y; \theta_k)$ (**Kalman smoother**)
- M-step: $\theta_{k+1} \leftarrow \arg\max_\theta L(q_k(X), \theta)$

Derivation of M-step for "B"

The terms of lowerbound including the parameter matrix $\mathbf{B}$ is extracted, then

$$\frac{\partial L_B}{\partial \mathbf{B}} \propto \frac{\partial}{\partial \mathbf{B}} \sum_{t=1}^{T} E_{q_k}[(\mathbf{y}_t - \mathbf{B}\mathbf{x}_t)^T \mathbf{R}^{-1} (\mathbf{y}_t - \mathbf{B}\mathbf{x}_t)] \quad (40)$$

$$\propto \frac{\partial}{\partial \mathbf{B}} \sum_{t=1}^{T} E_{q_k}[-2\mathbf{y}_t^T \mathbf{R}^{-1} \mathbf{B}\mathbf{x}_t + Tr(\mathbf{x}_t^T \mathbf{B}^T \mathbf{R}^{-1} \mathbf{B}\mathbf{x}_t)] \quad (41)$$

$$= -\sum_{t=1}^{T} \mathbf{R}^{-1} \mathbf{y}_t \boldsymbol{\mu}_t^T + \sum_{t=1}^{T} \mathbf{R}^{-1} \mathbf{B}\mathbf{P}_t = 0 \quad (42)$$

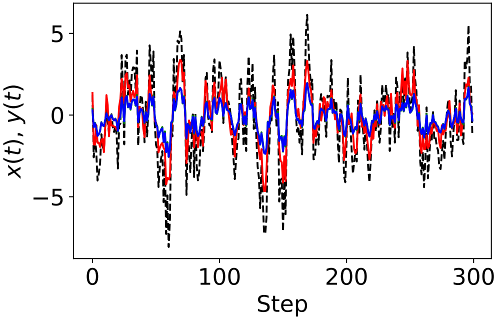
where $p(\mathbf{x}_t \mid \mathbf{Y}) = \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_t, \mathbf{V}_t)$ is **the solution of Kalman smoother!**
Thus, $\mathbf{B}^* = (\sum_{t=1}^{T} \mathbf{y}_t \boldsymbol{\mu}_t^T)(\sum_{t=1}^{T} \mathbf{P}_t)^{-1}$
$Tr(ABC) = Tr(BCA)$, $\frac{\partial}{\partial X} Tr(AXBX^T) = 2AXB$, $\mathbf{P}_t = \mathbf{V}_t + \boldsymbol{\mu}_t \boldsymbol{\mu}_t^T$,
$\frac{\partial}{\partial X} Tr(AXB) = A^T B^T$

Zoubin Ghahramani 1996

Exercise 2-4: Derivation of M-step for "A"

$$\frac{\partial L_A}{\partial \mathbf{A}} \propto \quad ... \tag{43}$$

Simulation Results



Let's see google colab

References

- Zoubin Ghahramani and Geoffrey E. Hinton: Parameter Estimation for Linear Dynamical Systemsm, Technical Report CRG-TR-92-2, 1996.
- Christopher M. Bishop: Pattern recognition and machine learning, 5th Edition. Information science and statistics, Springer 2007
- Jeffrey W. Miller (2016). Lecture Notes on Advanced Stochastic Modeling. Duke University, Durham, NC. (https://jwmi.github.io/ASM/6-KalmanFilter.pdf)
- Zoubin Ghahramani and Geoffrey E. Hinton: Parameter Estimation for Linear Dynamical Systems, U of Tronto, Technical Report CRG-TR-96-2.