Probabilistic Graphical Models

Factor Analysis and State Space Models

Eric Xing
Lecture 11, February 17, 2016

Reading: See class website
A road map to more complex dynamic models

- **Discrete** Y
- **Continuous** X

**Mixture model**
e.g., mixture of multinomials
- $y_t \rightarrow y_2 \rightarrow y_3 \rightarrow \ldots \rightarrow y_N$
  - $x_1, x_2, x_3, \ldots, x_N$

**HMM**
(for discrete sequential data, e.g., text)

**Mixture model**
e.g., mixture of Gaussians
- $y_t \rightarrow y_2 \rightarrow y_3 \rightarrow \ldots \rightarrow y_N$
  - $x_1, x_2, x_3, \ldots, x_N$

**HMM**
(for continuous sequential data, e.g., speech signal)

**Factorial HMM**

**Switching SSM**

**Factor analysis**

**State space model**

© Eric Xing @ CMU, 2005-2016
Recall multivariate Gaussian

- Multivariate Gaussian density:
  \[ p(x | \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp\left\{ -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right\} \]

- A joint Gaussian:
  \[ p\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} | \mu, \Sigma \right) = \mathcal{N}\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \middle| \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right) \]

- How to write down \( p(x_1) \), \( p(x_1|x_2) \) or \( p(x_2|x_1) \) using the block elements in \( \mu \) and \( \Sigma \)?
  - Formulas to remember:
    \[
    \begin{align*}
    p(x_2) &= \mathcal{N}(x_2 \middle| m_2^m, V_2^m) \\
    m_2^m &= \mu_2 \\
    V_2^m &= \Sigma_{22} \\
    p(x_1|x_2) &= \mathcal{N}(x_1 \middle| m_{1|2}, V_{1|2}) \\
    m_{1|2} &= \mu_1 + \Sigma_{12} \Sigma_{22}^{-1} (x_2 - \mu_2) \\
    V_{1|2} &= \Sigma_{11} - \Sigma_{12} \Sigma_{22}^{-1} \Sigma_{21}
    \end{align*}
    \]
The matrix inverse lemma

Consider a block-partitioned matrix: $M = \begin{bmatrix} E & F \\ G & H \end{bmatrix}$

First we diagonalize $M$

\[
\begin{bmatrix} I & -FH^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} E & F \\ G & H \end{bmatrix} \begin{bmatrix} I & 0 \\ -H^{-1}G & I \end{bmatrix} = \begin{bmatrix} E-FH^{-1}G & 0 \\ 0 & H \end{bmatrix}
\]

Schur complement: $M/H = E-FH^{-1}G$

Then we inverse, using this formula: $XYZ = W \Rightarrow Y^{-1} = ZW^{-1}X$

\[
M^{-1} = \begin{bmatrix} E & F \\ G & H \end{bmatrix}^{-1} = \begin{bmatrix} I & 0 \\ -H^{-1}G & I \end{bmatrix} \begin{bmatrix} (M/H)^{-1} & 0 \\ 0 & H^{-1} \end{bmatrix} \begin{bmatrix} I & -FH^{-1} \\ 0 & I \end{bmatrix}
\]

\[
= \begin{bmatrix} (M/H)^{-1} & -(M/H)^{-1}FH^{-1} \\ -H^{-1}G(M/H)^{-1} & H^{-1} + H^{-1}G(M/H)^{-1}FH^{-1} \end{bmatrix}
\begin{bmatrix} E^{-1} + E^{-1}F(M/E)^{-1}GE^{-1} & -E^{-1}F(M/E)^{-1} \\ -(M/E)^{-1}GE^{-1} & (M/E)^{-1} \end{bmatrix}
\]

Matrix inverse lemma

\[
(E-FH^{-1}G)^{-1} = E^{-1} + E^{-1}F(H-GE^{-1}F)^{-1}GE^{-1}
\]
Review: Some matrix algebra

- Trace and derivatives
  - Cyclical permutations
    \[ \text{tr}[A] \overset{\text{def}}{=} \sum_i a_{ii} \]
    \[ \text{tr}[ABC] = \text{tr}[CAB] = \text{tr}[BCA] \]
  - Derivatives
    \[ \frac{\partial}{\partial A} \text{tr}[BA] = B^T \]
    \[ \frac{\partial}{\partial A} \text{tr}[x^T A x] = \frac{\partial}{\partial A} \text{tr}[x x^T A] = x x^T \]

- Determinants and derivatives
  \[ \frac{\partial}{\partial A} \log |A| = A^{-1} \]
Factor analysis

- An unsupervised linear regression model

\[ p(x) = \mathcal{N}(x; 0, I) \]

\[ p(y|x) = \mathcal{N}(y; \mu + \Lambda x, \Psi) \]

where \( \Lambda \) is called a factor loading matrix, and \( \Psi \) is diagonal.

- Geometric interpretation

  - To generate data, first generate a point within the manifold then add noise. Coordinates of point are components of latent variable.
Marginal data distribution

- A marginal Gaussian (e.g., $p(x)$) times a conditional Gaussian (e.g., $p(y|x)$) is a joint Gaussian

- Any marginal (e.g., $p(y)$) of a joint Gaussian (e.g., $p(x,y)$) is also a Gaussian
  - Since the marginal is Gaussian, we can determine it by just computing its mean and variance. (Assume noise uncorrelated with data.)

\[
E[Y] = E[\mu + \Lambda X + W] \quad \text{where } W \sim \mathcal{N}(0, \Psi)
\]

\[
= \mu + \Lambda E[X] + E[W]
\]

\[
= \mu + 0 + 0 = \mu
\]

\[
\text{Var}[Y] = E[(Y - \mu)(Y - \mu)^T]
\]

\[
= E[(\mu + \Lambda X + W - \mu)(\mu + \Lambda X + W - \mu)^T]
\]

\[
= E[(\Lambda X + W)(\Lambda X + W)^T]
\]

\[
= \Lambda E[XX^T]\Lambda^T + E[WW^T]
\]

\[
= \Lambda \Lambda^T + \Psi
\]
FA = Constrained-Covariance Gaussian

- Marginal density for factor analysis ($y$ is $p$-dim, $x$ is $k$-dim):

  $$p(y \mid \theta) = \mathcal{N}(y; \mu, \Lambda \Lambda^T + \Psi)$$

- So the effective covariance is the low-rank outer product of two long skinny matrices plus a diagonal matrix:

  $$\text{Cov}[y] = \Lambda \Lambda^T + \Psi$$

- In other words, factor analysis is just a constrained Gaussian model. (If $\Psi$ were not diagonal then we could model any Gaussian and it would be pointless.)
Model

\[ p(x) = \mathcal{N}(x;\mathbf{0}, I) \]

\[ p(y|x) = \mathcal{N}(y; \mu + \Lambda x, \Psi) \]

Covariance between \( x \) and \( y \)

\[
\text{Cov}[X, Y] = E[(X - 0)(Y - \mu)^T] = E[X(\mu + \Lambda X + W - \mu)^T] \\
= E[XX^T \Lambda^T + XW^T] \\
= \Lambda^T
\]

Hence the joint distribution of \( x \) and \( y \):

\[
p\left( \begin{bmatrix} x \\ y \end{bmatrix} \right) = \mathcal{N} \left( \begin{bmatrix} x \\ y \end{bmatrix} \mid \begin{bmatrix} 0 \\ \mu \end{bmatrix}, \begin{bmatrix} I & \Lambda^T \\ \Lambda & \Lambda \Lambda^T + \Psi \end{bmatrix} \right)
\]

Assume noise is uncorrelated with data or latent variables.
Inference in Factor Analysis

- Apply the Gaussian conditioning formulas to the joint distribution we derived above, where

\[
\Sigma_{11} = I \\
\Sigma_{12} = \Sigma_{12}^T = \Lambda^T \\
\Sigma_{22} = (\Lambda\Lambda^T + \Psi)
\]

we can now derive the posterior of the latent variable \(x\) given observation \(y\), \(p(x|y) = \mathcal{N}(x | m_{1|2}, V_{1|2})\), where

\[
m_{1|2} = \mu_1 + \Sigma_{12} \Sigma_{22}^{-1} (y - \mu_2) \\
= \Lambda^T (\Lambda\Lambda^T + \Psi)^{-1} (y - \mu)
\]

\[
V_{1|2} = \Sigma_{11} - \Sigma_{12} \Sigma_{22}^{-1} \Sigma_{21} \\
= I - \Lambda^T (\Lambda\Lambda^T + \Psi)^{-1} \Lambda
\]

Applying the matrix inversion lemma

\[
(E^{-1}FH^{-1}G)^{-1} = E^{-1} + E^{-1}F(HGE^{-1}F)^{-1}GE^{-1}
\]

\[
\Rightarrow V_{1|2} = \left( I + \Lambda^T \Psi^{-1} \Lambda \right)^{-1} \\
m_{1|2} = V_{1|2} \Lambda^T \Psi^{-1} (y - \mu)
\]

- Here we only need to invert a matrix of size \(|x| \times |x|\), instead of \(|y| \times |y|\).
Geometric interpretation: inference is linear projection

- The posterior is:
  \[ p(x|y) = \mathcal{N}(x; m_{1|2}, V_{1|2}) \]
  \[ V_{1|2} = (I + \Lambda^T\Lambda)^{-1} \]
  \[ m_{1|2} = V_{1|2}\Lambda^T\Psi^{-1}(y - \mu) \]

- Posterior covariance does not depend on observed data \( y \)!
- Computing the posterior mean is just a linear operation:
Learning FA

- Now, assume that we are given \( \{y_n\} \) (the observation on high-dimensional data) only

- We have derived how to estimate \( x_n \) from \( P(X|Y) \)

- How can we learning the model?
  - Loading matrix \( \Lambda \)
  - Manifold center \( \mu \)
  - Variance \( \Psi \)
EM for Factor Analysis

- Incomplete data log likelihood function (marginal density of $y$)
  \[
  \ell(\theta, D) = -\frac{N}{2} \log |\Lambda \Lambda^T + \Psi| - \frac{1}{2} \sum_n (y_n - \mu)^T (\Lambda \Lambda^T + \Psi)^{-1} (y_n - \mu) \\
  = -\frac{N}{2} \log |\Lambda \Lambda^T + \Psi| - \frac{1}{2} \text{tr} \left[ (\Lambda \Lambda^T + \Psi)^{-1} S \right], \text{ where } S = \sum_n (y_n - \mu)(y_n - \mu)^T
  \]
  - Estimating $\mu$ is trivial: $\hat{\mu}^{ML} = \frac{1}{N} \sum_n y_n$
  - Parameters $\Lambda$ and $\Psi$ are coupled nonlinearly in log-likelihood

- Complete log likelihood
  \[
  \ell_c(\theta, D) = \sum_n \log p(x_n, y_n) = \sum_n \log p(x_n) + \log p(y_n \mid x_n) \\
  = -\frac{N}{2} \log |\Gamma| - \frac{1}{2} \sum_n x_n^T \Gamma x_n - \frac{N}{2} \log |\Psi| - \frac{1}{2} \sum_n (y_n - \Lambda x_n)^T \Psi^{-1} (y_n - \Lambda x_n) \\
  = -\frac{N}{2} \log |\Psi| - \frac{1}{2} \sum_n \text{tr}[x_n x_n^T] - \frac{N}{2} \text{tr}[S \Psi^{-1}], \text{ where } S = \frac{1}{N} \sum_n (y_n - \Lambda x_n)(y_n - \Lambda x_n)^T
  \]
E-step for Factor Analysis

- Compute $\langle \ell_e(\theta, D) \rangle_{p(x|y)}$

$$
\langle \ell_e(\theta, D) \rangle = -\frac{N}{2} \log|\Psi| - \frac{1}{2} \sum_n \text{tr}[\langle X_n X_n^T \rangle] - \frac{N}{2} \text{tr}[\langle S \Psi^{-1} \rangle]
$$

$$
\langle S \rangle = \frac{1}{N} \sum_n (y_n y_n^T - y_n \langle X_n^T \rangle \Lambda^T - \Lambda \langle X_n^T \rangle y_n^T + \Lambda \langle X_n X_n^T \rangle \Lambda^T)
$$

$$
\langle X_n \rangle = E[X_n | y_n]
$$

$$
\langle X_n X_n^T \rangle = \text{Var}[X_n | y_n] + E[X_n | y_n] E[X_n | y_n]^T
$$

- Recall that we have derived:

$$
V_{1/2} = \left( I + \Lambda^T \Psi^{-1} \Lambda \right)^{-1}
$$

$$
m_{1/2} = V_{1/2} \Lambda^T \Psi^{-1} (y - \mu)
$$

$$
\Rightarrow \quad \langle X_n \rangle = m_{x_n|y_n} = V_{1/2} \Lambda^T \Psi^{-1} (y_n - \mu) \quad \text{and} \quad \langle X_n X_n^T \rangle = V_{1/2} + m_{x_n|y_n} m_{x_n|y_n}^T
$$
M-step for Factor Analysis

- Take the derivates of the expected complete log likelihood wrt. parameters.
  - Using the trace and determinant derivative rules:

\[
\frac{\partial}{\partial \Psi^{-1}} \langle \ell_c \rangle = \frac{\partial}{\partial \Psi^{-1}} \left( -\frac{N}{2} \log|\Psi| - \frac{1}{2} \sum_n \text{tr} \left[ \langle X_n X_n^T \rangle \right] - \frac{N}{2} \text{tr} \left[ (S) \Psi^{-1} \right] \right) \\
= \frac{N}{2} \Psi - \frac{N}{2} \langle S \rangle \quad \Rightarrow \quad \Psi^{t+1} = \langle S \rangle
\]

\[
\frac{\partial}{\partial \Lambda} \langle \ell_c \rangle = \frac{\partial}{\partial \Lambda} \left( -\frac{N}{2} \log|\Psi| - \frac{1}{2} \sum_n \text{tr} \left[ \langle X_n X_n^T \rangle \right] - \frac{N}{2} \text{tr} \left[ (S) \Psi^{-1} \right] \right) = -\frac{N}{2} \Psi^{-1} \frac{\partial}{\partial \Lambda} \langle S \rangle \\
= -\frac{N}{2} \Psi^{-1} \frac{\partial}{\partial \Lambda} \left( \frac{1}{N} \sum_n (y_n y_n^T - y_n \langle X_n^T \rangle \Lambda^T - \Lambda \langle X_n^T \rangle y_n^T + \Lambda \langle X_n X_n^T \rangle \Lambda^T) \right) \\
= \Psi^{-1} \sum_n y_n \langle X_n^T \rangle - \Psi^{-1} \Lambda \sum_n \langle X_n X_n^T \rangle \quad \Rightarrow \quad \Lambda^{t+1} = \left( \sum_n y_n \langle X_n^T \rangle \right) \left( \sum_n \langle X_n X_n^T \rangle \right)^{-1}
\]
Model Invariance and Identifiability

- There is *degeneracy* in the FA model.
- Since $\Lambda$ only appears as outer product $\Lambda \Lambda^T$, the model is invariant to rotation and axis flips of the latent space.
- We can replace $\Lambda$ with $\Lambda Q$ for any orthonormal matrix $Q$ and the model remains the same: $(\Lambda Q)(\Lambda Q)^T = \Lambda(QQ^T)\Lambda^T = \Lambda \Lambda^T$.
- This means that there is no “one best” setting of the parameters. An infinite number of parameters all give the ML score!
- Such models are called *un-identifiable* since two people both fitting ML parameters to the identical data will not be guaranteed to identify the same parameters.
A road map to more complex dynamic models

- **Discrete**
  - $X$
  - $Y$
  - Mixture model (e.g., mixture of multinomials)
  - HMM (for discrete sequential data, e.g., text)

- **Continuous**
  - $X$
  - $Y$
  - Mixture model (e.g., mixture of Gaussians)
  - HMM (for continuous sequential data, e.g., speech signal)
  - Factor analysis
  - State space model

- **Factorial HMM**
  - $S_1 \rightarrow S_2 \rightarrow \cdots \rightarrow S_N$
  - $y_1 \rightarrow y_2 \rightarrow \cdots \rightarrow y_N$
  - $x_1 \rightarrow x_2 \rightarrow \cdots \rightarrow x_N$

- **Switching SSM**
  - $S_1 \rightarrow S_2 \rightarrow \cdots \rightarrow S_N$
  - $y_1 \rightarrow y_2 \rightarrow \cdots \rightarrow y_N$
  - $x_1 \rightarrow x_2 \rightarrow \cdots \rightarrow x_N$

© Eric Xing @ CMU, 2005-2016
State space models (SSM)

- A sequential FA or a continuous state HMM

\[ x_t = Ax_{t-1} + Gw_t \]
\[ y_t =Cx_{t-1} + v_t \]
\[ w_t \sim \mathcal{N}(0; Q), \quad v_t \sim \mathcal{N}(0; R) \]
\[ x_0 \sim \mathcal{N}(0; \Sigma_0), \]

This is a linear dynamic system.

- In general,

\[ x_t = f(x_{t-1}) + Gw_t \]
\[ y_t = g(x_{t-1}) + v_t \]

where \( f \) is an (arbitrary) dynamic model, and \( g \) is an (arbitrary) observation model.
LDS for 2D tracking

- Dynamics: new position = \textit{old position} + \Delta \times \textit{velocity} + \textit{noise} (constant velocity model, Gaussian noise)

\[
\begin{pmatrix}
    x_{t-1}^1 \\
    x_{t-1}^2 \\
    \dot{x}_t^1 \\
    \dot{x}_t^2
\end{pmatrix} =
\begin{pmatrix}
    1 & 0 & \Delta & 0 \\
    0 & 1 & 0 & \Delta \\
    0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
    x_{t-1}^1 \\
    x_{t-1}^2 \\
    \dot{x}_{t-1}^1 \\
    \dot{x}_{t-1}^2
\end{pmatrix} + \textit{noise}
\]

- Observation: project out first two components (we observe Cartesian position of object - linear!)

\[
\begin{pmatrix}
    y_t^1 \\
    y_t^2
\end{pmatrix} =
\begin{pmatrix}
    1 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
    x_t^1 \\
    x_t^2 \\
    \dot{x}_t^1 \\
    \dot{x}_t^2
\end{pmatrix} + \textit{noise}
\]
The inference problem 1

- Filtering → given \( y_1, \ldots, y_t \), estimate \( x_t: \) \( P(x_t \mid y_{1:t}) \)
  - The Kalman filter is a way to perform exact online inference (sequential Bayesian updating) in an LDS.
  - It is the Gaussian analog of the forward algorithm for HMMs:
    \[
    p(X_t = i \mid y_{1:t}) = \alpha_i^t \propto p(y_t \mid X_t = i) \sum_j p(X_t = i \mid X_{t-1} = j) \alpha_j^{t-1}
    \]
The inference problem 2

- Smoothing → given $y_1, \ldots, y_T$, estimate $x_t (t<T)$
- The Rauch-Tung-Strievel smoother is a way to perform exact off-line inference in an LDS. It is the Gaussian analog of the forwards-backwards (alpha-gamma) algorithm:

$$p(X_t = i \mid y_{1:T}) = \gamma_t^i \propto \sum_j \alpha_t^i P(X_{t+1}^j \mid X_t^j) \gamma_{t+1}^j$$
2D tracking

2D filtering

2D smoothing

© Eric Xing @ CMU, 2005-2016
Kalman filtering in the brain?
Kalman filtering derivation

- Since all CPDs are linear Gaussian, the system defines a large multivariate Gaussian.
  - Hence all marginals are Gaussian.
  - Hence we can represent the belief state $p(X_t|y_{1:t})$ as a Gaussian with mean and covariance .
  - It is common to work with the inverse covariance (precision) matrix ; this is called information form.

- Kalman filtering is a recursive procedure to update the belief state:
  - Predict step: compute $p(X_{t+1}|y_{1:t})$ from prior belief $p(X_t|y_{1:t})$ and dynamical model $p(X_{t+1}|X_t)$ --- time update
  - Update step: compute new belief $p(X_{t+1}|y_{1:t+1})$ from prediction $p(X_{t+1}|y_{1:t})$, observation $y_{t+1}$ and observation model $p(y_{t+1}|X_{t+1})$ --- measurement update
Kalman filtering derivation

- Kalman filtering is a recursive procedure to update the belief state:
  - Predict step: compute $p(X_{t+1}|y_{1:t})$ from prior belief $p(X_t|y_{1:t})$ and dynamical model $p(X_{t+1}|X_t)$ --- time update
  
  ![Diagram of Kalman filter predict step]

  - Update step: compute new belief $p(X_{t+1}|y_{1:t+1})$ from prediction $p(X_{t+1}|y_{1:t})$, observation $y_{t+1}$ and observation model $p(y_{t+1}|X_{t+1})$ --- measurement update

  ![Diagram of Kalman filter update step]
Predict step

- Dynamical Model: \[ x_{t+1} = Ax_t + Gw_t, \quad w_t \sim \mathcal{N}(0; Q) \]
  - One step ahead prediction of state:

- Observation model: \[ y_t = Cx_t + v_t, \quad v_t \sim \mathcal{N}(0; R) \]
  - One step ahead prediction of observation:
Predict step

- **Dynamical Model:** \( x_{t+1} = Ax_t + Gw_t, \quad w_t \sim \mathcal{N}(0; Q) \)
  - One step ahead prediction of state:
    \[
    \hat{x}_{t+1|t} = E(X_{t+1} \mid y_1, \ldots, y_t) = A\hat{x}_{t|t}
    \]
    \[
    P_{t+1|t} = E(X_{t+1} - \hat{x}_{t+1|t})(X_{t+1} - \hat{x}_{t+1|t})^T \mid y_1, \ldots, y_t
    \]
    \[
    = E(AX_t + Gw_t - \hat{x}_{t+1|t})(AX_t + Gw_t - \hat{x}_{t+1|t})^T \mid y_1, \ldots, y_t
    \]
    \[
    = AP_{t|t}A + GQG^T
    \]

- **Observation model:** \( y_t = Cx_t + v_t, \quad v_t \sim \mathcal{N}(0; R) \)
  - One step ahead prediction of observation:
    \[
    E(Y_{t+1} \mid y_1, \ldots, y_t) = E(CX_{t+1} + v_{t+1} \mid y_1, \ldots, y_t) = C\hat{x}_{t+1|t}
    \]
    \[
    E(Y_{t+1} - \hat{y}_{t+1|t})(Y_{t+1} - \hat{y}_{t+1|t})^T \mid y_1, \ldots, y_t = CP_{t+1|t}C^T + R
    \]
    \[
    E(Y_{t+1} - \hat{y}_{t+1|t})(X_{t+1} - \hat{x}_{t+1|t})^T \mid y_1, \ldots, y_t = CP_{t+1|t}
    \]
Update step

- Summarizing results from previous slide, we have
  \[ p(X_{t+1}, Y_{t+1}|y_{1:t}) \sim \mathcal{N}(m_{t+1}, V_{t+1}) \]
  where
  \[
  m_{t+1} = \begin{pmatrix}
  \hat{X}_{t+1|t} \\
  C\hat{X}_{t+1|t}
  \end{pmatrix}, \\
  V_{t+1} = \begin{pmatrix}
  P_{t+1|t} & P_{t+1|t}C^T \\
  CP_{t+1|t} & CP_{t+1|t}C^T + R
  \end{pmatrix},
  \]

- Remember the formulas for conditional Gaussian distributions:
  \[
p(x_1 \mid \mu, \Sigma) = \mathcal{N}
  \begin{pmatrix}
  x_1 \\
  x_2
  \end{pmatrix}
  \bigg|
  \begin{pmatrix}
  \mu_1 \\
  \mu_2
  \end{pmatrix}
  ,
  \begin{pmatrix}
  \Sigma_{11} & \Sigma_{12} \\
  \Sigma_{21} & \Sigma_{22}
  \end{pmatrix}
  
  p(x_2) = \mathcal{N}(x_2 \mid m_2^m, V_2^m) \\
  p(x_1 \mid x_2) = \mathcal{N}(x_1 \mid m_{1|2}, V_{1|2})
  \]
  \[
  m_2^m = \mu_2 \\
  V_2^m = \Sigma_{22} \\
  m_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(x_2 - \mu_2) \\
  V_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}
  \]
Kalman Filter

- Measurement updates:

\[
\hat{x}_{t+1|t+1} = \hat{x}_{t+1|t} + K_{t+1}(y_{t+1} - C\hat{x}_{t+1|t})
\]

\[
P_{t+1|t+1} = P_{t+1|t} - KCP_{t+1|t}
\]

where \(K_{t+1}\) is the *Kalman gain matrix*

\[
K_{t+1} = P_{t+1|t}C^T(CP_{t+1|t}C^T + R)^{-1}
\]

- Time updates:

\[
\hat{x}_{t+1|t} = A\hat{x}_{t|t}
\]

\[
P_{t+1|t} = AP_{t|t}A + GQG^T
\]

- \(K_t\) can be pre-computed (since it is independent of the data).
Example of KF in 1D

- Consider noisy observations of a 1D particle doing a random walk:
  \[ x_{t|t-1} = x_{t-1} + w, \ w \sim \mathcal{N}(0, \sigma_x) \quad z_t = x_t + v, \ v \sim \mathcal{N}(0, \sigma_z) \]

- KF equations:
  \[ P_{t+1|t} = AP_{t|t} A + GQG^T = \sigma_t + \sigma_x, \quad \hat{x}_{t+1|t} = A\hat{x}_{t|t} = \hat{x}_{t|t} \]
  \[ K_{t+1} = P_{t+1|t} C^T (CP_{t+1|t} C^T + R)^{-1} = (\sigma_t + \sigma_x)(\sigma_t + \sigma_x + \sigma_z) \]
  \[ \hat{x}_{t+1|t+1} = \hat{x}_{t+1|t} + K_{t+1}(z_{t+1} - C\hat{x}_{t+1|t}) = \frac{(\sigma_t + \sigma_x)z_t + \sigma_z \hat{x}_{t|t}}{\sigma_t + \sigma_x + \sigma_z} \]
  \[ P_{t+1|t+1} = P_{t+1|t} - KCP_{t+1|t} = \frac{(\sigma_t + \sigma_x)\sigma_z}{\sigma_t + \sigma_x + \sigma_z} \]
KF intuition

- The KF update of the mean is
  \[ \hat{x}_{t+1|t+1} = \hat{x}_{t+1|t} + K_{t+1} (z_{t+1} - C \hat{x}_{t+1|t}) = \frac{(\sigma_t + \sigma_x)z_t + \sigma_z \hat{x}_{t|t}}{\sigma_t + \sigma_x + \sigma_z} \]
  the term \((z_{t+1} - C \hat{x}_{t+1|t})\) is called the **innovation**

- New belief is convex combination of updates from prior and observation, weighted by Kalman Gain matrix:
  \[ K_{t+1} = P_{t+1|t} C^T (CP_{t+1|t} C^T + R)^{-1} \]

- If the observation is unreliable, \(\sigma_z\) (i.e., \(R\)) is large so \(K_{t+1}\) is small, so we pay more attention to the prediction.

- If the old prior is unreliable (large \(\sigma_t\)) or the process is very unpredictable (large \(\sigma_x\)), we pay more attention to the observation.
Complexity of one KF step

- Let $X_t \in \mathbb{R}^{N_x}$ and $y_t \in \mathbb{R}^{N_y}$.

- Computing $P_{t+1|t} = AP_{t|t}A + GQG^T$ takes $O(N_x^2)$ time, assuming dense $P$ and dense $A$.

- Computing $K_{t+1} = P_{t+1|t}C^T(CP_{t+1|t}C^T + R)^{-1}$ takes $O(N_y^3)$ time.

- So overall time is, in general, $\max \{N_x^2, N_y^3\}$.
The inference problem 2

- Smoothing → given $y_1, \ldots, y_T$, estimate $x_t$ (t<T)
  - The Rauch-Tung-Strievel smoother is a way to perform exact off-line inference in an LDS. It is the Gaussian analog of the forwards-backwards (alpha-gamma) algorithm:

$$p(X_t = i \mid y_{1:T}) = \gamma^i_t \propto \sum_j \alpha^i_j P(X_{t+1}^j \mid X_t^i) \gamma^j_{t+1}$$
Rauch-Tung-Strievel smoother

\[ \hat{x}_{t|t} = \hat{x}_{t|t} + L_t (\hat{x}_{t+1|t} - \hat{x}_{t+1|t}) \]
\[ P_{t|t} = P_{t|t} + L_t (P_{t+1|t} - P_{t+1|t}) L_t^T \]
\[ L_t = P_{t|t} A^T P_{t+1|t}^{-1} \]

- General structure: KF results + the difference of the "smoothed" and predicted results of the next step
- Backward computation: Pretend to know things at t+1 — such conditioning makes things simple and we can remove this condition finally

- The difficulty:
  \[ X_t | y_1, \ldots, y_T \]

- The trick:
  \[ E[X | Z] = E[E[X | Y, Z] | Z] \]
  \[ \text{Var}[X | Z] = \text{Var}[E[X | Y, Z] | Z] + E[\text{Var}[X | Y, Z] | Z] \]
  \[ \hat{x}_{t|T} \overset{\text{def}}{=} E[X_t | y_1, \ldots, y_T] = E[E[X_t | X_{t+1}, y_1, \ldots, y_T] | y_1, \ldots, y_T] = E[E[X_t | X_{t+1}, y_1, \ldots, y_T] | y_1, \ldots, y_T] = E[X_t | X_{t+1}, y_1, \ldots, y_T] \]

Same for \( P_{t|T} \)
RTS derivation

- Following the results from previous slide, we need to derive
  \[ p(X_{t+1}, X_t | y_{1:t}) \sim \mathcal{N}(m, V), \]
  where

  \[
  m = \begin{pmatrix}
  \hat{x}_{t|t} \\
  \hat{x}_{t+1|t}
  \end{pmatrix},
  \quad V = \begin{pmatrix}
  P_{t|t} & P_{t|t} A^T \\
  A P_{t|t} & P_{t+1|t}
  \end{pmatrix},
  \]

  - all the quantities here are available after a forward KF pass

- Remember the formulas for conditional Gaussian distributions:

  \[
  p(x_1) = \mathcal{N}(x_1 | m_1, V_{11})
  \]

- The RTS smoother

  \[
  \hat{x}_{t|T} = E[X_t | X_{t+1}, Y_1, \ldots, Y_t] = \hat{x}_{t|t} + L_t (\hat{x}_{t+1|T} - \hat{x}_{t+1|t})
  \]

  \[
  P_{t|T} = Var[\hat{x}_{t|T} | Y_{1:T}] + E[Var[X_t | X_{t+1}, Y_{1:t}] | Y_{1:T}]
  \]

  \[
  = P_{t|t} + L_t (P_{t+1|T} - P_{t+1|t}) L_t^T
  \]

  © Eric Xing @ CMU, 2005-2016
Learning SSMs

- Complete log likelihood

\[ \ell_c(\theta, D) = \sum_n \log p(x_n, y_n) = \sum_n \log p(x_0) + \sum_n \sum_t \log p(x_{n,t} \mid x_{n,t-1}) + \sum_n \sum_t \log p(y_{n,t} \mid x_{n,t}) \]

\[ = f_1(x_1; \Sigma_0) + f_2(\{x_t x_{t-1}^T, x_t^T, x_t : \forall t \}; A, Q, G) + f_3(\{x_t^T, x_t : \forall t \}; C, R) \]

- EM
  - E-step: compute \( \langle x_t x_{t-1}^T \rangle, \langle x_t x_t^T \rangle, \langle x_t \rangle \mid y_1, \ldots, y_T \)

  these quantities can be inferred via KF and RTS filters, etc., e.g., \( \langle x_t x_t^T \rangle \equiv \text{var}(x_t x_t^T) + E(x_t)^2 = P_{t \mid t} + \hat{x}_{t \mid t}^2 \)

  - M-step: MLE using

\[ \langle \ell_c(\theta, D) \rangle = f_1(\langle x_1 \rangle; \Sigma_0) + f_2(\{\langle x_t x_{t-1}^T \rangle, \langle x_t x_t^T \rangle, \langle x_t \rangle : \forall t \}; A, Q, G) + f_3(\{\langle x_t x_t^T \rangle, \langle x_t \rangle : \forall t \}; C, R) \]

c.f., M-step in factor analysis
Nonlinear systems

- In robotics and other problems, the motion model and the observation model are often nonlinear:
  \[ x_t = f(x_{t-1}) + w_t, \quad y_t = g(x_t) + v_t \]
- An optimal closed form solution to the filtering problem is no longer possible.
- The nonlinear functions \( f \) and \( g \) are sometimes represented by neural networks (multi-layer perceptrons or radial basis function networks).
- The parameters of \( f \) and \( g \) may be learned offline using EM, where we do gradient descent (back propagation) in the M step, c.f. learning a MRF/CRF with hidden nodes.
- Or we may learn the parameters online by adding them to the state space: \( x_t' = (x_t, \theta) \). This makes the problem even more nonlinear.
Extended Kalman Filter (EKF)

- The basic idea of the EKF is to linearize $f$ and $g$ using a second order Taylor expansion, and then apply the standard KF.
  - i.e., we approximate a stationary nonlinear system with a non-stationary linear system.

$$
x_t = f(\hat{x}_{t-1|t-1}) + A_{\hat{x}_{t-1|t-1}} (x_{t-1} - \hat{x}_{t-1|t-1}) + w_t
$$

$$
y_t = g(\hat{x}_{t|t-1}) + C_{\hat{x}_{t|t-1}} (x_t - \hat{x}_{t|t-1}) + v_t
$$

where $\hat{x}_{t|t-1} = f(\hat{x}_{t-1|t-1})$ and $A_{\hat{x}} \overset{\text{def}}{=} \frac{\partial f}{\partial x} \bigg|_{\hat{x}}$ and $C_{\hat{x}} \overset{\text{def}}{=} \frac{\partial g}{\partial x} \bigg|_{\hat{x}}$

- The noise covariance ($Q$ and $R$) is not changed, i.e., the additional error due to linearization is not modeled.
Online vs offline inference

filtering: $P(X(t) \mid y(1:t))$

Viterbi: $\arg\max_{x(1:t)} P(x(1:t) \mid y(1:t))$

prediction: $P(X(t+\delta) \mid y(1:t))$

fixed-lag smoothing: $P(X(t-\tau) \mid y(1:t))$

fixed interval smoothing (offline): $P(X(t) \mid y(1:T))$
The KF update of the mean is
\[ \hat{x}_{t+1|t+1} = A\hat{x}_{t|t} + K_{t+1}(y_{t+1} - C\hat{x}_{t+1|t}) \]

Consider the special case where the hidden state is a constant, \( x_t = \theta \), but the “observation matrix” \( C \) is a time-varying vector, \( C = x_t^T \).

- Hence the observation model at each time slide, \( y_t = x_t^T \theta + v_t \), is a linear regression.

We can estimate recursively using the Kalman filter:
\[ \hat{\theta}_{t+1} = \hat{\theta}_t + P_{t+1}R^{-1}(y_{t+1} - x_t^T \hat{\theta}_t)x_t \]

This is called the recursive least squares (RLS) algorithm.

We can approximate \( P_{t+1}R^{-1} \approx \eta_{t+1} \) by a scalar constant. This is called the least mean squares (LMS) algorithm.

We can adapt \( \eta_t \) online using stochastic approximation theory.