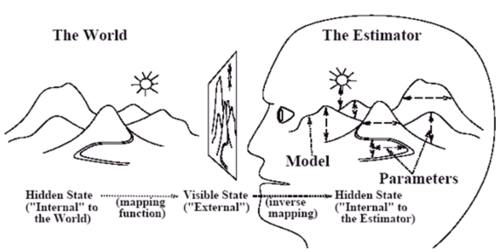
School of Computer Science
Carnegie Mellon

Probabilistic Graphical Models

Factor Analysis and State Space Models

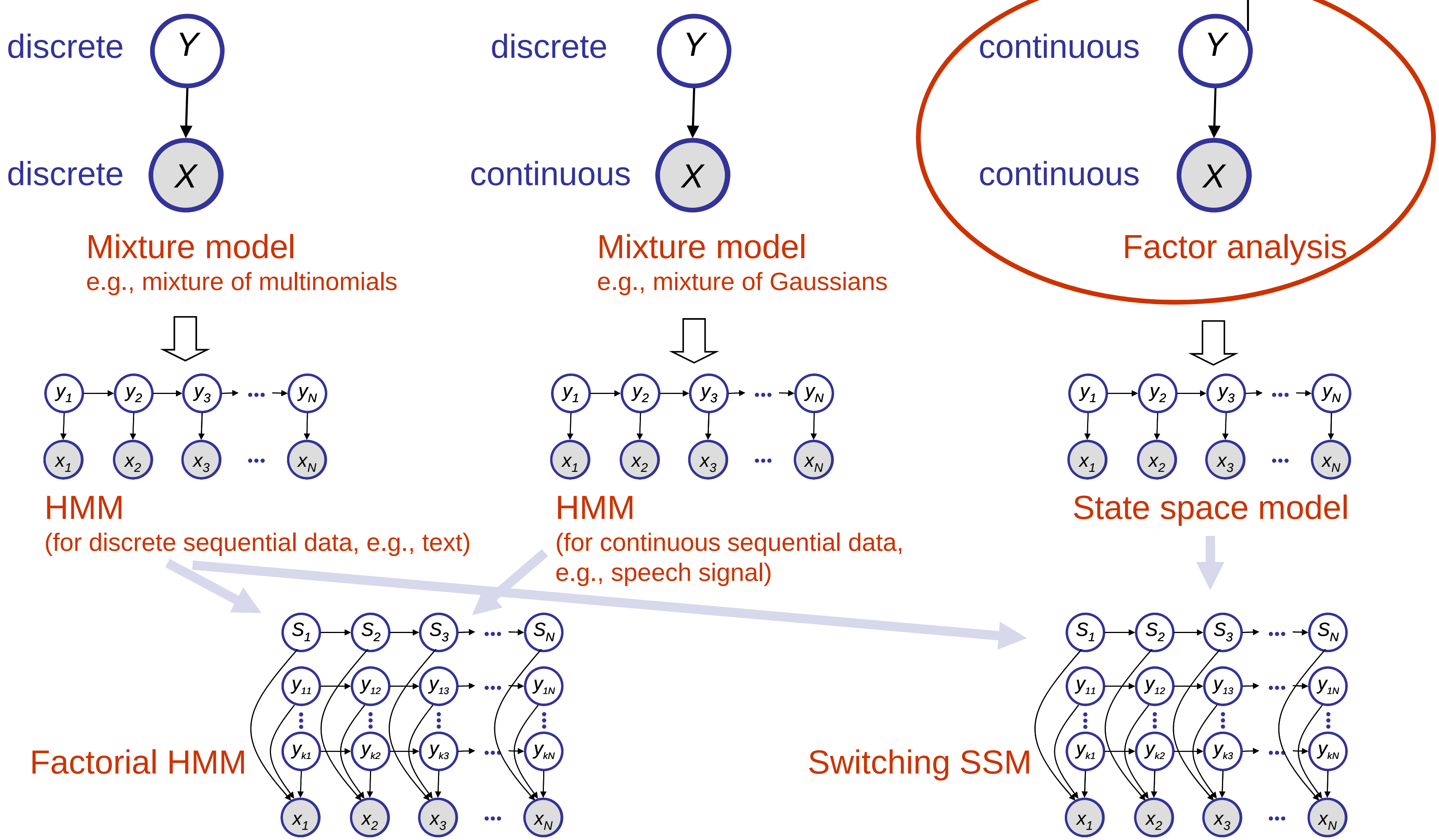
Eric Xing

Lecture 11, February 18, 2015



Reading: See class website

A road map to more complex dynamic models



Recall multivariate Gaussian

- Multivariate Gaussian density:

$$p(\mathbf{x}|\mu,\Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left\{-\tfrac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1}(\mathbf{x}-\mu)\right\}$$

- A joint Gaussian:

$$p\left(\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix} \middle| \mu,\Sigma\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix} \middle| \begin{bmatrix}\mu_1\\ \mu_2\end{bmatrix}, \begin{bmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{bmatrix}\right)$$

- How to write down $p(\mathbf{x}_1)$, $p(\mathbf{x}_1|\mathbf{x}_2)$ or $p(\mathbf{x}_2|\mathbf{x}_1)$ using the block elements in μ and Σ?
  - Formulas to remember:

$$p(\mathbf{x}_2) = \mathcal{N}(\mathbf{x}_2|\mathbf{m}_2^m, \mathbf{V}_2^m)$$
$$\mathbf{m}_2^m = \mu_2$$
$$\mathbf{V}_2^m = \Sigma_{22}$$

$$p(\mathbf{x}_1|\mathbf{x}_2) = \mathcal{N}(\mathbf{x}_1|\mathbf{m}_{1|2}, \mathbf{V}_{1|2})$$
$$\mathbf{m}_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \mu_2)$$
$$\mathbf{V}_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

Review: 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

$$\left(E-FH^{-1}G\right)^{-1} = E^{-1} + E^{-1}F\left(H-GE^{-1}F\right)^{-1}GE^{-1}$$

Review: Some matrix algebra

- Trace and derivatives $\mathrm{tr}[A] \overset{\mathrm{def}}{=} \sum_i a_{ii}$
  - Cyclical permutations

$$\mathrm{tr}[ABC] = \mathrm{tr}[CAB] = \mathrm{tr}[BCA]$$

  - Derivatives

$$\frac{\partial}{\partial A}\mathrm{tr}[BA] = B^T$$

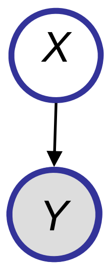
$$\frac{\partial}{\partial A}\mathrm{tr}[x^T A x] = \frac{\partial}{\partial A}\mathrm{tr}[x x^T A] = x x^T$$

- Determinants and derivatives

$$\frac{\partial}{\partial A}\log|A| = A^{-1}$$

Factor analysis

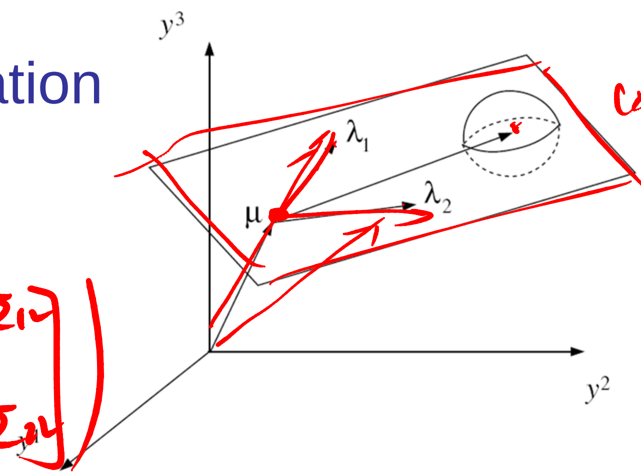
- An unsupervised linear regression model



$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; 0, I)$$
$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}; \mu + \Lambda\mathbf{x}, \Psi)$$

where Λ is called a factor loading matrix, and Ψ is diagonal.

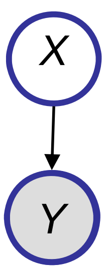
- Geometric interpretation



  - 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., $p(\mathbf{x})$) times a conditional Gaussian (e.g., $p(\mathbf{y}|\mathbf{x})$) is a joint Gaussian
- Any marginal (e.g., $p(\mathbf{y})$ of a joint Gaussian (e.g., $p(\mathbf{x},\mathbf{y})$) 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.)



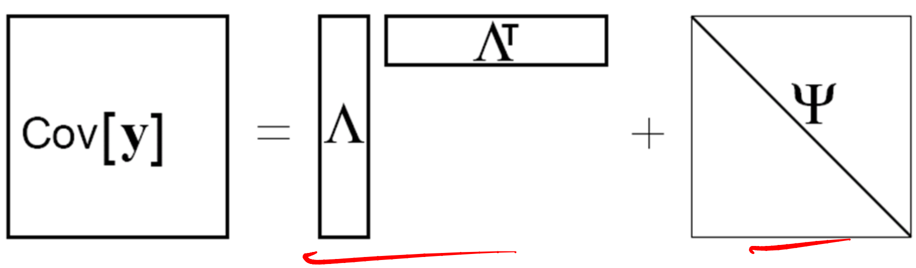
$$
\begin{aligned}
E[\mathbf{Y}] &= E[\mu + \Lambda\mathbf{X} + \mathbf{W}] \qquad \text{where } \mathbf{W} \sim \mathcal{N}(0, \Psi) \\
&= \mu + \Lambda E[\mathbf{X}] + E[\mathbf{W}] \\
&= \mu + 0 + 0 = \mu \\
Var[\mathbf{Y}] &= E\left[(\mathbf{Y}-\mu)(\mathbf{Y}-\mu)^T\right] \\
&= E\left[(\mu + \Lambda\mathbf{X} + \mathbf{W} - \mu)(\mu + \Lambda\mathbf{X} + \mathbf{W} - \mu)^T\right] \\
&= E\left[(\Lambda\mathbf{X} + \mathbf{W})(\Lambda\mathbf{X} + \mathbf{W})^T\right] \\
&= \Lambda E\left[\mathbf{X}\mathbf{X}^T\right]\Lambda^T + E\left[\mathbf{W}\mathbf{W}^T\right] \\
&= \Lambda\Lambda^T + \Psi
\end{aligned}
$$

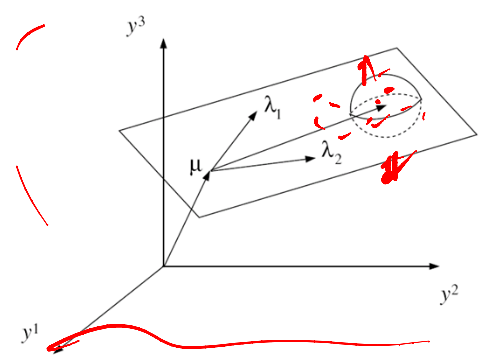
FA = Constrained-Covariance Gaussian

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

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

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





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

FA joint distribution

- Model

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

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

- Covariance between $\mathbf{x}$ and $\mathbf{y}$

$$\begin{aligned} Cov[\mathbf{X},\mathbf{Y}] &= E\left[(\mathbf{X}-0)(\mathbf{Y}-\mu)^T\right] = E\left[\mathbf{X}(\mu + \Lambda\mathbf{X} + \mathbf{W} - \mu)^T\right] \\ &= E\left[\mathbf{X}\mathbf{X}^T\Lambda^T + \mathbf{X}\mathbf{W}^T\right] \\ &= \Lambda^T \end{aligned}$$

- Hence the joint distribution of $\mathbf{x}$ and $\mathbf{y}$:

$$p\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix} \middle| \begin{bmatrix}0\\ \mu\end{bmatrix}, \begin{bmatrix} I & \Lambda^T \\ \Lambda & \Lambda\Lambda^T + \Psi\end{bmatrix}\right)$$

- Assume noise is uncorrelated with data or latent variables.

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

Inference in Factor Analysis

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

$$\Sigma_{11} = I$$

$$\Sigma_{12} = \Sigma_{12}{}^{T} = \Lambda^{T}$$

$$\Sigma_{22} = \left(\Lambda\Lambda^{T} + \Psi\right)$$

we can now derive the posterior of the latent variable $\mathbf{x}$ given observation $\mathbf{y}$, $p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x} \mid \mathbf{m}_{1|2}, \mathbf{V}_{1|2})$, where

$$\mathbf{m}_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{y} - \mu_2) = \Lambda^{T}\left(\Lambda\Lambda^{T} + \Psi\right)^{-1}(\mathbf{y} - \mu)$$

$$\mathbf{V}_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} = I - \Lambda^{T}\left(\Lambda\Lambda^{T} + \Psi\right)^{-1}\Lambda$$

Applying the matrix inversion lemma $\left(E\text{-}FH^{-1}G\right)^{-1} = E^{-1} + E^{-1}F\left(H\text{-}GE^{-1}F\right)^{-1}GE^{-1}$

$$\Rightarrow \quad \mathbf{V}_{1|2} = \left(I + \Lambda^{T}\Psi^{-1}\Lambda\right)^{-1} \qquad \mathbf{m}_{1|2} = \mathbf{V}_{1|2}\Lambda^{T}\Psi^{-1}(\mathbf{y} - \mu)$$

- Here we only need to invert a matrix of size $|\mathbf{x}|\times|\mathbf{x}|$, instead of $|\mathbf{y}|\times|\mathbf{y}|$.

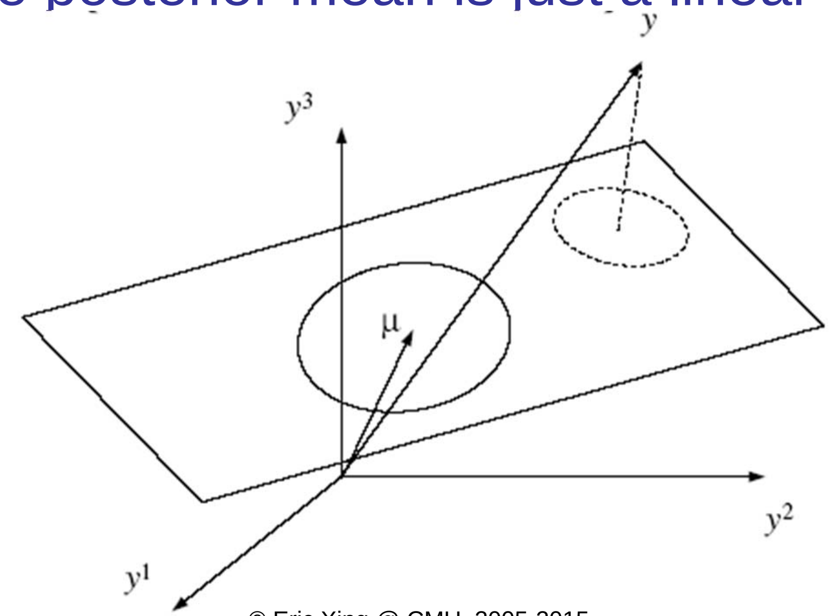
Geometric interpretation: inference is linear projection

- The posterior is:

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}; \mathbf{m}_{1|2}, \mathbf{V}_{1|2})$$

$$\mathbf{V}_{1|2} = \left(I + \Lambda^T \Psi^{-1} \Lambda\right)^{-1} \qquad \mathbf{m}_{1|2} = \mathbf{V}_{1|2} \Lambda^T \Psi^{-1} (\mathbf{y} - \mu)$$

- Posterior covariance does not depend on observed data $\mathbf{y}$!
- 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 Λ
  - Manifold center μ
  - Variance Ψ

EM for Factor Analysis

- Incomplete data log likelihood function (marginal density of y)

$$\ell(\theta, D) = -\frac{N}{2}\log\left|\Lambda\Lambda^T + \Psi\right| - \frac{1}{2}\sum_n (y_n - \mu)^T \left(\Lambda\Lambda^T + \Psi\right)^{-1} (y_n - \mu)$$

$$= -\frac{N}{2}\log\left|\Lambda\Lambda^T + \Psi\right| - \frac{1}{2}\mathrm{tr}\left[\left(\Lambda\Lambda^T + \Psi\right)^{-1}\mathbf{S}\right], \qquad \text{where } \mathbf{S} = \sum_n (y_n - \mu)(y_n - \mu)^T$$

  - Estimating μ is trivial: $\hat{\mu}^{ML} = \frac{1}{N}\sum_n y_n$
  - Parameters Λ and Ψ are coupled nonlinearly in log-likelihood

- Complete log likelihood

$$\ell_c(\theta, D) = \sum_n \log p(x_n, y_n) = \sum_n \log p(x_n) + \log p(y_n \mid x_n)$$

$$= -\frac{N}{2}\log|I| - \frac{1}{2}\sum_n x_n^T x_n - \frac{N}{2}\log|\Psi| - \frac{1}{2}\sum_n (y_n - \Lambda x_n)^T \Psi^{-1} (y_n - \Lambda x_n)$$

$$= -\frac{N}{2}\log|\Psi| - \frac{1}{2}\sum_n \mathrm{tr}\left[x_n x_n^T\right] - \frac{N}{2}\mathrm{tr}\left[\mathbf{S}\Psi^{-1}\right], \qquad \text{where } \mathbf{S} = \frac{1}{N}\sum_n (y_n - \Lambda x_n)(y_n - \Lambda x_n)^T$$

E-step for Factor Analysis

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

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

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

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

$$\langle X_n X_n^T \rangle = Var[X_n \mid y_n] + E[X_n \mid y_n] E[X_n \mid y_n]^T$$

- Recall that we have derived:

$$\mathbf{V}_{1|2} = \left(I + \Lambda^T \Psi^{-1} \Lambda\right)^{-1} \qquad \mathbf{m}_{1|2} = \mathbf{V}_{1|2} \Lambda^T \Psi^{-1} (\mathbf{y} - \mu)$$

$$\Rightarrow \quad \langle X_n \rangle = \mathbf{m}_{x_n|y_n} = \mathbf{V}_{1|2} \Lambda^T \Psi^{-1} (y_n - \mu) \quad \text{and} \quad \langle X_n X_n^T \rangle = \mathbf{V}_{1|2} + \mathbf{m}_{x_n|y_n} \mathbf{m}_{x_n|y_n}^T$$

M-step for Factor Analysis

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

$$\frac{\partial}{\partial \Psi^{-1}}\langle \ell_c \rangle = \frac{\partial}{\partial \Psi^{-1}}\left(-\frac{N}{2}\log|\Psi| - \frac{1}{2}\sum_n \mathrm{tr}\left[\langle X_n X_n^T \rangle\right] - \frac{N}{2}\mathrm{tr}\left[\langle \mathbf{S} \rangle \Psi^{-1}\right]\right)$$

$$= \frac{N}{2}\Psi - \frac{N}{2}\langle \mathbf{S} \rangle \qquad \Rightarrow \qquad \Psi^{t+1} = \langle \mathbf{S} \rangle$$

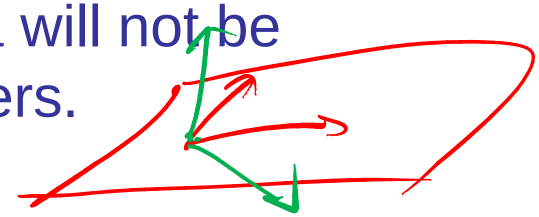
$$\frac{\partial}{\partial \Lambda}\langle \ell_c \rangle = \frac{\partial}{\partial \Lambda}\left(-\frac{N}{2}\log|\Psi| - \frac{1}{2}\sum_n \mathrm{tr}\left[\langle X_n X_n^T \rangle\right] - \frac{N}{2}\mathrm{tr}\left[\langle \mathbf{S} \rangle \Psi^{-1}\right]\right) = -\frac{N}{2}\Psi^{-1}\frac{\partial}{\partial \Lambda}\langle \mathbf{S} \rangle$$

$$= -\frac{N}{2}\Psi^{-1}\frac{\partial}{\partial \Lambda}\left(\frac{1}{N}\sum_n (y_n y_n^T - y_n \langle X_n^T \rangle \Lambda^T - \Lambda \langle X_n^T \rangle y_n^T + \Lambda \langle X_n X_n^T \rangle \Lambda^T)\right)$$

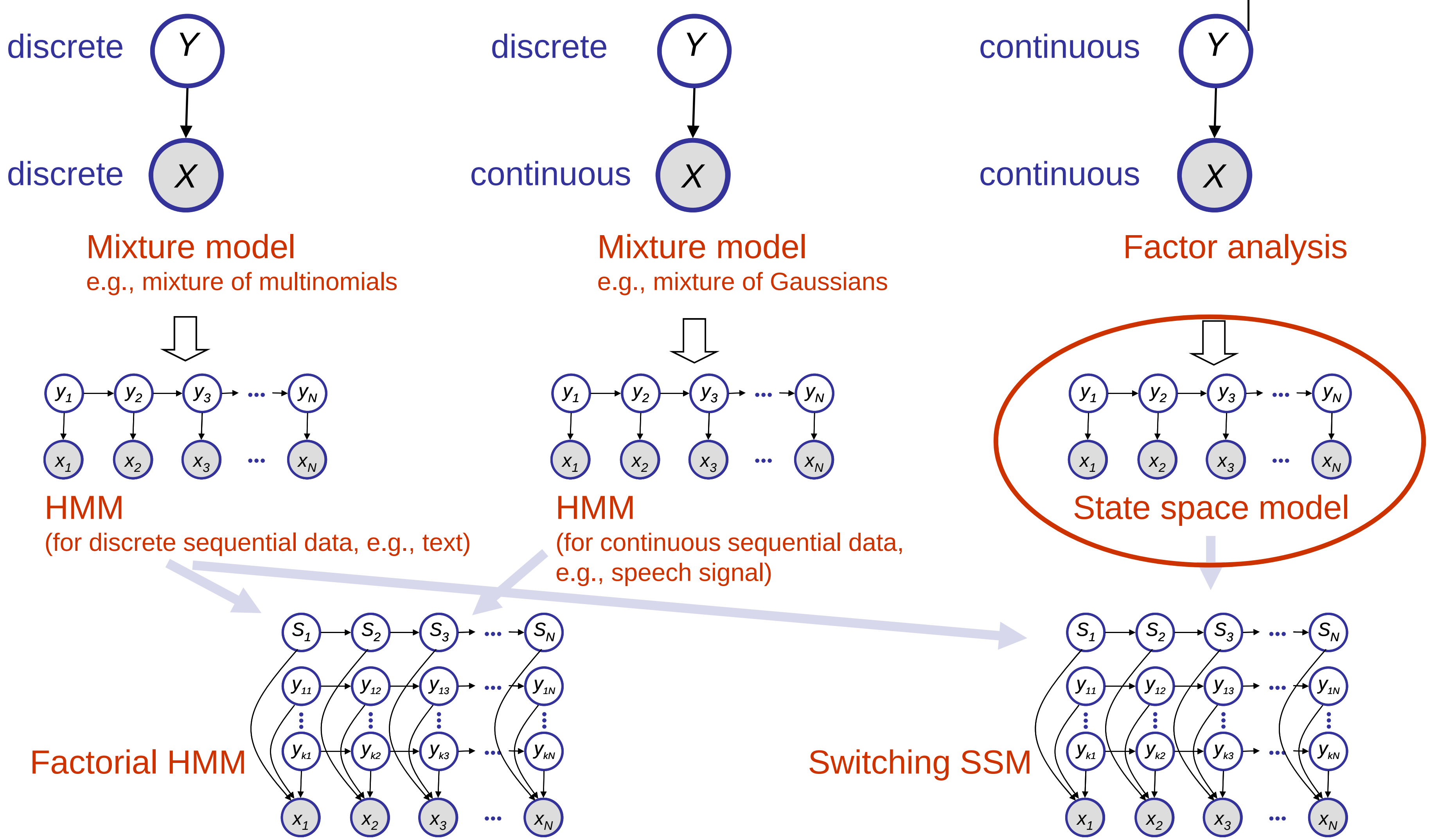
$$= \Psi^{-1}\sum_n y_n \langle X_n^T \rangle - \Psi^{-1}\Lambda \sum_n \langle X_n X_n^T \rangle \qquad \Rightarrow \qquad \Lambda^{t+1} = \left(\sum_n y_n \langle X_n^T \rangle \right)\left(\sum_n \langle X_n X_n^T \rangle \right)^{-1}$$

Model Invariance and Identifiability

- There is *degeneracy* in the FA model.
- Since Λ only appears as outer product $\Lambda\Lambda^T$, the model is invariant to rotation and axis flips of the latent space.
- We can replace Λ with ΛQ 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.

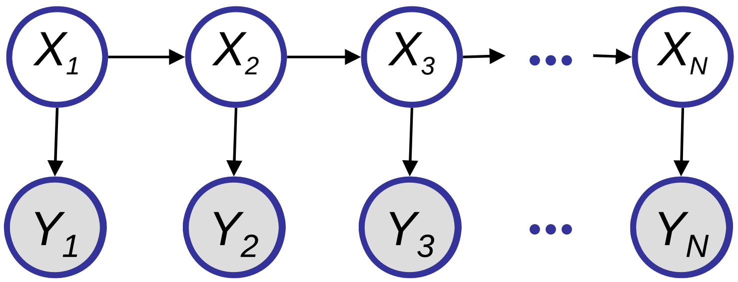


A road map to more complex dynamic models



State space models (SSM)

- A sequential FA or a continuous state HMM



$$\mathbf{x}_t = A\mathbf{x}_{t-1} + Gw_t$$
$$\mathbf{y}_t = C\mathbf{x}_{t-1} + v_t$$
$$w_t \sim \mathcal{N}(0; Q), \quad v_t \sim \mathcal{N}(0; R)$$
$$\mathbf{x}_0 \sim \mathcal{N}(0; \Sigma_0),$$

This is a linear dynamic system.

- In general,

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}) + Gw_t$$
$$\mathbf{y}_t = g(\mathbf{x}_{t-1}) + v_t$$

where f is an (arbitrary) dynamic model, and g is an (arbitrary) observation model

LDS for 2D tracking

- Dynamics: new position = *old position* + $\Delta \times$*velocity* + *noise* (constant velocity model, Gaussian noise)

$$\begin{pmatrix} x_t^1 \\ x_t^2 \\ \dot{x}_t^1 \\ \dot{x}_t^2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & \Delta & 0 \\ 0 & 1 & 0 & \Delta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{t-1}^1 \\ x_{t-1}^2 \\ \dot{x}_{t-1}^1 \\ \dot{x}_{t-1}^2 \end{pmatrix} + \text{noise}$$

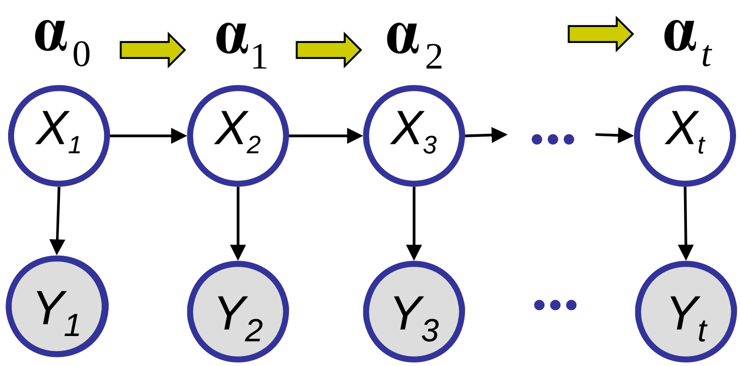
- Observation: project out first two components (we observe Cartesian position of object - linear!)

$$\begin{pmatrix} y_t^1 \\ y_t^2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_t^1 \\ x_t^2 \\ \dot{x}_t^1 \\ \dot{x}_t^2 \end{pmatrix} + \text{noise}$$

The inference problem 1

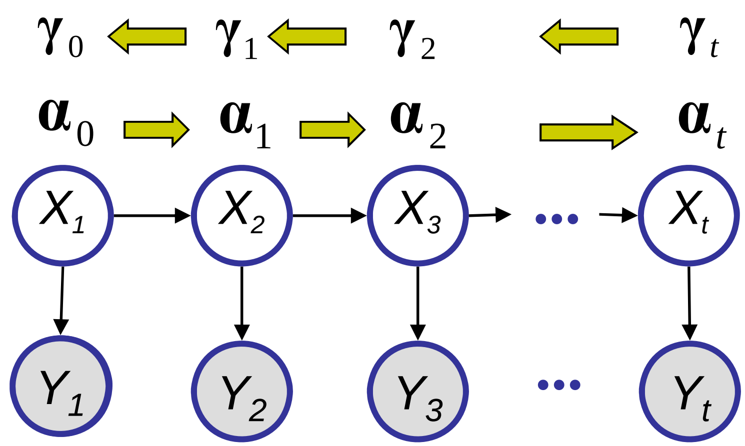
- Filtering → given $\mathbf{y}_1, \ldots, \mathbf{y}_t$, estimate x_t: $P(x_t \mid \mathbf{y}_{1:t})$
  - The Kalman filter is a way to perform exact online inference (sequential Bayesian updating) in an LDS.
  - It is the Gaussian analog of the forward algorithm for HMMs:

$$p(\mathrm{X}_t = i \mid \mathrm{y}_{1:t}) = \alpha_t^i \propto p(\mathrm{y}_t \mid \mathrm{X}_t = i) \sum_j p(\mathrm{X}_t = i \mid \mathrm{X}_{t-1} = j)\alpha_{t-1}^j$$



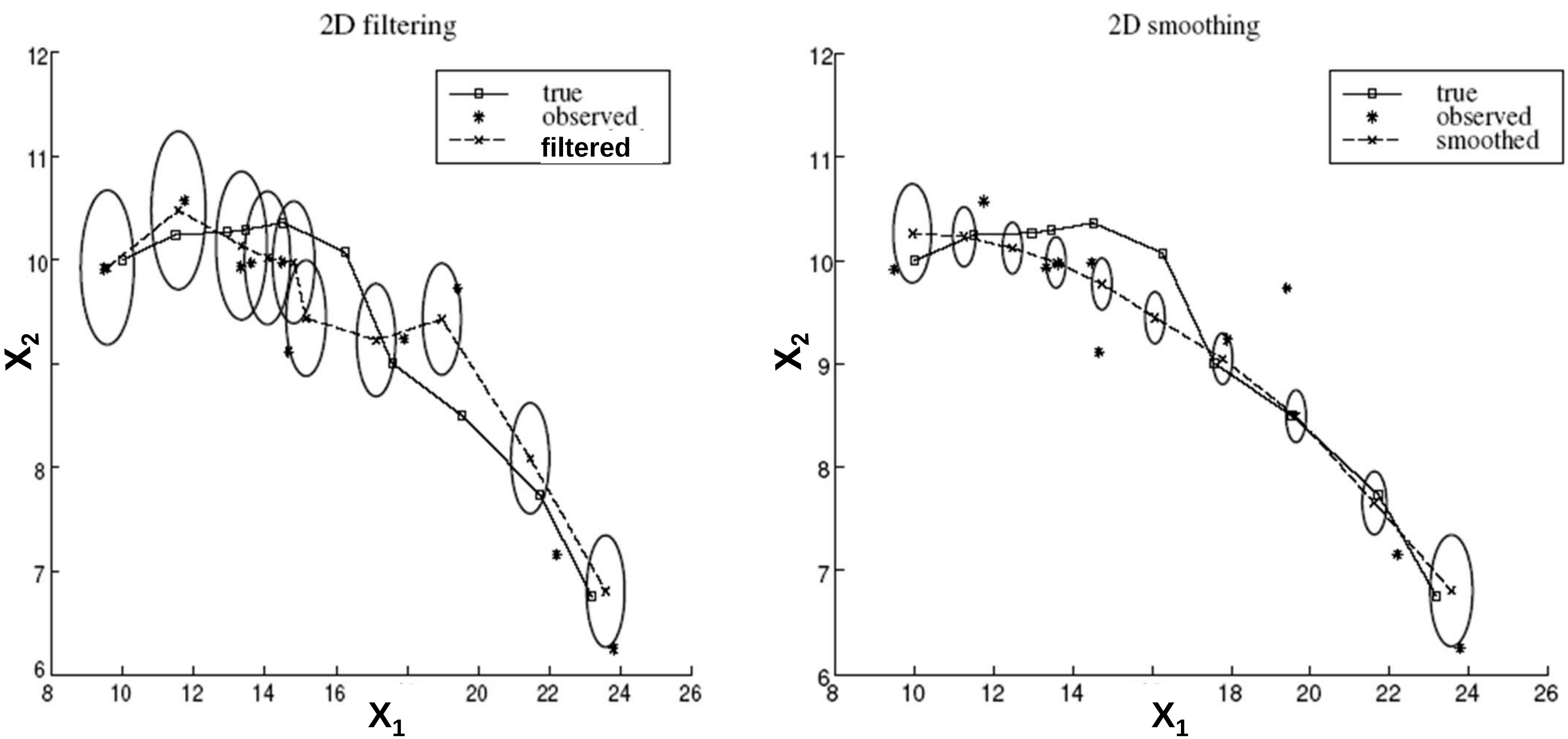
The inference problem 2

- Smoothing → given $\mathbf{y}_1, \ldots, \mathbf{y}_T$, 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:

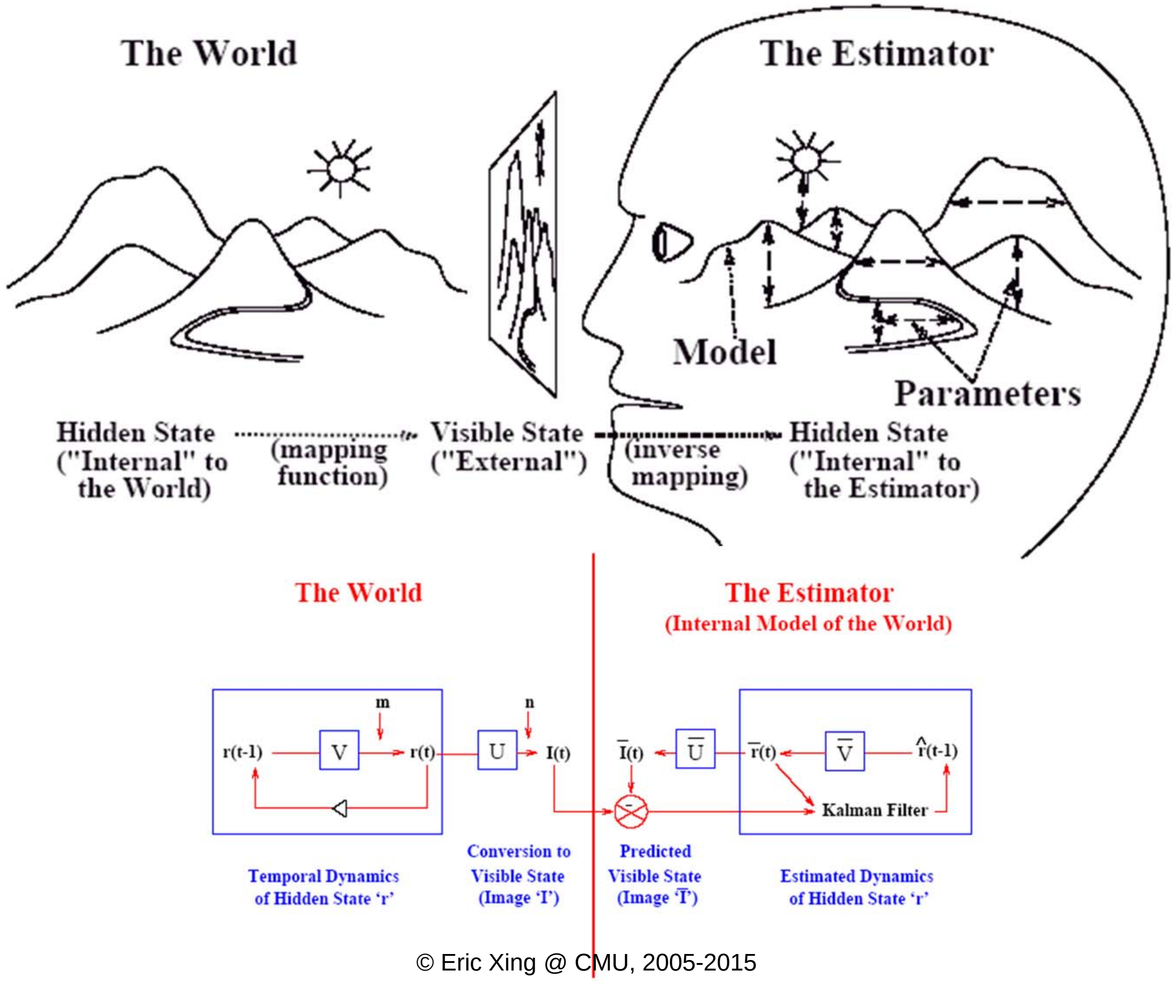


$$p(X_t = i \mid y_{1:T}) = \gamma_t^i \propto \sum_j \alpha_t^i P(X_{t+1}^j \mid X_i^j) \gamma_{t+1}^j$$

2D tracking


2D filtering
true
observed
filtered
X2
X1
2D smoothing
true
observed
smoothed
X2
X1

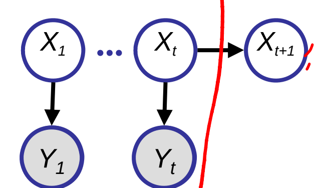
Kalman filtering in the brain?



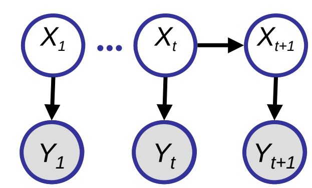
Kalman filtering derivation

- 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 $p(\mathbf{X}_t|\mathbf{y}_{1:t})$ as a Gaussian with mean $E(X_t|y_{1:t}) = \mu_{t|t}$ and covariance $E(X_t-\mu_{t|t})(X_t-\mu_{t|t})^T = P_{t|t}$.
  - 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 $p(\mathbf{X}_{t+1}|\mathbf{y}_{1:t})$ from prior belief $p(\mathbf{X}_t|\mathbf{y}_{1:t})$ and dynamical model $p(\mathbf{X}_{t+1}|\mathbf{X}_t)$ --- time update

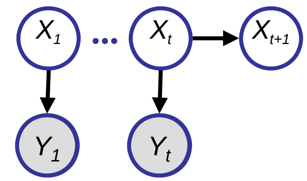


  - Update step: compute new belief $p(\mathbf{X}_{t+1}|\mathbf{y}_{1:t+1})$ from prediction $p(\mathbf{X}_{t+1}|\mathbf{y}_{1:t})$, observation $\mathbf{y}_{t+1}$ and observation model $p(\mathbf{y}_{t+1}|\mathbf{X}_{t+1})$ --- measurement update

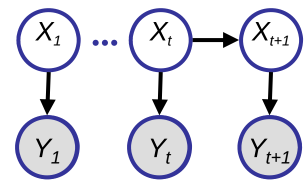


Kalman filtering derivation

- Kalman filtering is a recursive procedure to update the belief state:
  - Predict step: compute $p(\mathbf{X}_{t+1}|\mathbf{y}_{1:t})$ from prior belief $p(\mathbf{X}_t|\mathbf{y}_{1:t})$ and dynamical model $p(\mathbf{X}_{t+1}|\mathbf{X}_t)$ --- time update



  - Update step: compute new belief $p(\mathbf{X}_{t+1}|\mathbf{y}_{1:t+1})$ from prediction $p(\mathbf{X}_{t+1}|\mathbf{y}_{1:t})$, observation $\mathbf{y}_{t+1}$ and observation model $p(\mathbf{y}_{t+1}|\mathbf{X}_{t+1})$ --- measurement update



$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \sim \left(\begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right)$ $z_1 \to z_1 z_2 \to z_2 | z_1$, $\downarrow z_2$

$z_1 \sim X_t$.

$P(X_t | y_{1:t}) \Rightarrow P(X_{t+1} | y_{1:t}) \Rightarrow P(X_{t+1}, y_{t+1} | y_{1:t}) \Rightarrow$

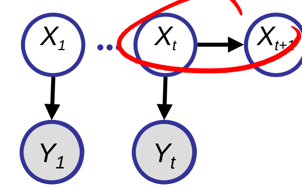
$P(y_{t+1}|X_{t+1})$ $P(y_{t+1}|y_{1:t})$

$X_{t+1} = f(X_t) = A X_t + w$

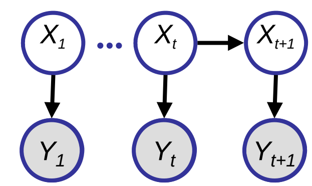
$y_{t+1} = C X_{t+1} + V$

Predict step

- Dynamical Model: $\mathbf{x}_{t+1} = A\mathbf{x}_t + Gw_t, \quad w_t \sim \mathcal{N}(0; Q)$
  - One step ahead prediction of state:

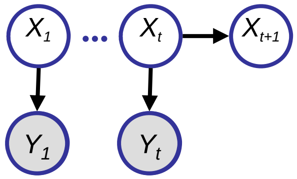


- Observation model: $\mathbf{y}_t = C\mathbf{x}_t + v_t, \quad v_t \sim \mathcal{N}(0; R)$
  - One step ahead prediction of observation:



Predict step

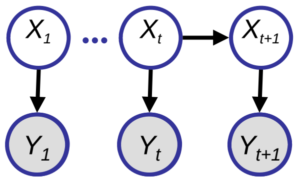
- Dynamical Model: $\mathbf{x}_{t+1} = A\mathbf{x}_t + Gw_t, \qquad w_t \sim \mathcal{N}(0; Q)$
  - One step ahead prediction of state:



$$\hat{\mathbf{x}}_{t+1|t} = E(\mathrm{X}_{t+1} \mid \mathbf{y}_1, \dots, \mathbf{y}_t) = A\hat{\mathbf{x}}_{t|t}$$

$$\begin{aligned} P_{t+1|t} &= E(\mathrm{X}_{t+1} - \hat{\mathbf{x}}_{t+1|t})(\mathrm{X}_{t+1} - \hat{\mathbf{x}}_{t+1|t})^T \mid \mathbf{y}_1, \dots, \mathbf{y}_t) \\ &= E(A\mathrm{X}_t + Gw_t - \hat{\mathbf{x}}_{t+1|t})(A\mathrm{X}_t + Gw_t - \hat{\mathbf{x}}_{t+1|t})^T \mid \mathbf{y}_1, \dots, \mathbf{y}_t) \\ &= AP_{t|t}A + GQG^T \end{aligned}$$

- Observation model: $\mathbf{y}_t = C\mathbf{x}_t + v_t, \quad v_t \sim \mathcal{N}(0; R)$
  - One step ahead prediction of observation:



$$E(\mathrm{Y}_{t+1} \mid \mathbf{y}_1, \dots, \mathbf{y}_t) = E(C\mathrm{X}_{t+1} + v_{t+1} \mid \mathbf{y}_1, \dots, \mathbf{y}_t) = C\hat{\mathbf{x}}_{t+1|t}$$

$$E(\mathrm{Y}_{t+1} - \hat{\mathbf{y}}_{t+1|t})(\mathrm{Y}_{t+1} - \hat{\mathbf{y}}_{t+1|t})^T \mid \mathbf{y}_1, \dots, \mathbf{y}_t) = CP_{t+1|t}C^T + \mathrm{R}$$

$$E(\mathrm{Y}_{t+1} - \hat{\mathbf{y}}_{t+1|t})(\mathrm{X}_{t+1} - \hat{\mathbf{x}}_{t+1|t})^T \mid \mathbf{y}_1, \dots, \mathbf{y}_t) = CP_{t+1|t}$$

Update step

- Summarizing results from previous slide, we have $p(\mathbf{X}_{t+1},\mathbf{Y}_{t+1}|\mathbf{y}_{1:t}) \sim \mathcal{N}(m_{t+1}, V_{t+1})$, where

$$m_{t+1} = \begin{pmatrix} \hat{x}_{t+1|t} \\ C\hat{x}_{t+1|t} \end{pmatrix}, \qquad V_{t+1} = \begin{pmatrix} P_{t+1|t} & P_{t+1|t}C^T \\ CP_{t+1|t} & CP_{t+1|t}C^T + R \end{pmatrix},$$

- Remember the formulas for conditional Gaussian distributions:

$$p\left(\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} \mid \mu, \Sigma\right) = \mathcal{N}\left(\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} \mid \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}\right)$$

$$p(\mathbf{x}_2) = \mathcal{N}(\mathbf{x}_2 \mid \mathbf{m}_2^m, \mathbf{V}_2^m) \qquad p(\mathbf{x}_1|\mathbf{x}_2) = \mathcal{N}(\mathbf{x}_1 \mid \mathbf{m}_{1|2}, \mathbf{V}_{1|2})$$

$$\mathbf{m}_2^m = \mu_2 \qquad \mathbf{m}_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \mu_2)$$

$$\mathbf{V}_2^m = \Sigma_{22} \qquad \mathbf{V}_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

Kalman Filter

- Measurement updates:

$$\hat{\mathbf{x}}_{t+1|t+1} = \hat{\mathbf{x}}_{t+1|t} + K_{t+1}(\mathrm{y}_{t+1} - \mathrm{C}\hat{\mathrm{x}}_{t+1|t})$$

$$P_{t+1|t+1} = P_{t+1|t} - KCP_{t+1|t}$$

  - 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:

$$\hat{\mathbf{x}}_{t+1|t} = A\hat{\mathbf{x}}_{t|t}$$

$$P_{t+1|t} = AP_{t|t} A + GQG^T$$

- 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:

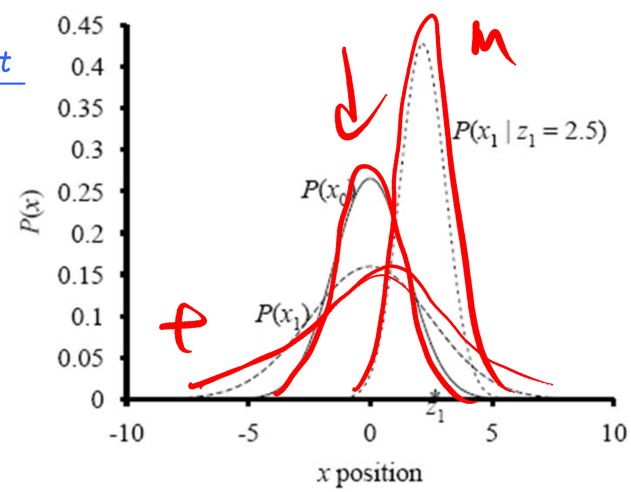
$$x_{t|t-1} = x_{t-1} + w, \quad w \sim \mathcal{N}(0, \sigma_x) \qquad z_t = x_t + v, \quad v \sim \mathcal{N}(0, \sigma_z)$$

- KF equations: $P_{t+1|t} = AP_{t|t}A + GQG^T = \sigma_t + \sigma_x \quad , \quad \hat{x}_{t+1|t} = A\hat{x}_{t|t} = \hat{x}_{t|t}$

$$K_{t+1} = P_{t+1|t}C^T(CP_{t+1|t}C^T + R)^{-1} = (\sigma_t + \sigma_x)(\sigma_t + \sigma_x + \sigma_z)$$

$$\hat{x}_{t+1|t+1} = \hat{x}_{t+1|t} + K_{t+1}(z_{t+1} - C\hat{x}_{t+1|t}) = \frac{(\sigma_t + \sigma_x)z_t + \sigma_z \hat{x}_{t|t}}{\sigma_t + \sigma_x + \sigma_z}$$

$$P_{t+1|t+1} = P_{t+1|t} - KCP_{t+1|t} = \frac{(\sigma_t + \sigma_x)\sigma_z}{\sigma_t + \sigma_x + \sigma_z}$$



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:

$$K_{t+1} = P_{t+1|t} C^T (C P_{t+1|t} C^T + R)^{-1}$$

- If the observation is unreliable, σ_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 σ_t) or the process is very unpredictable (large σ_x), we pay more attention to the observation.

Complexity of one KF step

- Let $x_t \in \mathbb{R}^{N_x}$ and $y_t \in \mathbb{R}^{N_y}$,

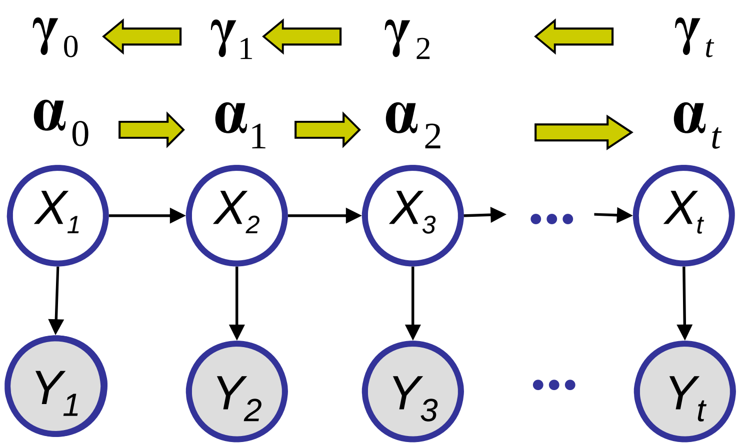
- Computing $P_{t+1|t} = AP_{t|t}A + GQG^T$ takes O(N_x^2) time, assuming dense P and dense A.

- 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}

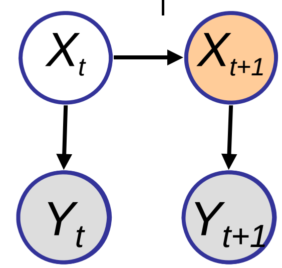
The inference problem 2

- Smoothing → given $\mathbf{y}_1, \ldots, \mathbf{y}_T$, 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:



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

Rauch-Tung-Strievel smoother



$$\hat{\mathbf{x}}_{t|T} = \hat{\mathbf{x}}_{t|t} + L_t(\hat{\mathbf{x}}_{t+1|T} - \hat{\mathbf{x}}_{t+1|t})$$

$$P_{t|T} = P_{t|t} + L_t\left(P_{t+1|T} - P_{t+1|t}\right)L_t^T \qquad L_t = P_{t|t}A^T P_{t+1|t}^{-1}$$

- 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: $x_t \mid y_1, \ldots, y_T$
- The trick:

$$E[X \mid Z] = E[E[X \mid Y, Z] \mid Z] \qquad \text{(Hw!)}$$

$$Var[X \mid Z] = Var[E[X \mid Y, Z] \mid Z] + E[Var[X \mid Y, Z] \mid Z]$$

$$\hat{x}_{t|T} \overset{\text{def}}{=} E[X_t \mid y_1, \ldots, y_T] = E[E[X_t \mid X_{t+1}, y_1, \ldots, y_T] \mid y_1, \ldots, y_T]$$
$$= E[E[X_t \mid X_{t+1}, y_1, \ldots, y_t] \mid y_1, \ldots, y_T]$$
$$= E[X_t \mid X_{t+1}, y_1, \ldots, y_t]$$

Same for $P_{t|T}$

RTS derivation

- Following the results from previous slide, we need to derive $p(\mathbf{X}_{t+1},\mathbf{X}_t|\mathbf{y}_{1:t}) \sim \mathcal{N}(m, V)$, where

$$m = \begin{pmatrix} \hat{x}_{t|t} \\ \hat{x}_{t+1|t} \end{pmatrix}, \qquad V = \begin{pmatrix} P_{t|t} & P_{t|t}A^T \\ AP_{t|t} & P_{t+1|t} \end{pmatrix},$$

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

- Remember the formulas for conditional Gaussian distributions:

$$p\left(\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} | \mu, \Sigma\right) = \mathcal{N}\left(\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} | \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}\right),$$

$$p(\mathbf{x}_2) = \mathcal{N}(\mathbf{x}_2 | \mathbf{m}_2^m, \mathbf{V}_2^m)$$
$$\mathbf{m}_2^m = \mu_2$$
$$\mathbf{V}_2^m = \Sigma_{22}$$

$$p(\mathbf{x}_1|\mathbf{x}_2) = \mathcal{N}(\mathbf{x}_1 | \mathbf{m}_{1|2}, \mathbf{V}_{1|2})$$
$$\mathbf{m}_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \mu_2)$$
$$\mathbf{V}_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

- The RTS smoother

$$\hat{x}_{t|T} = E[X_t | X_{t+1}, y_1, \ldots, y_t] = \hat{\mathbf{x}}_{t|t} + L_t(\hat{\mathbf{x}}_{t+1|T} - \hat{\mathbf{x}}_{t+1|t})$$

$$P_{t|T} \stackrel{\text{def}}{=} Var[\hat{x}_{t|T} | y_{1:T}] + E[Var[X_t | X_{t+1}, y_{1:t}] | y_{1:T}] = P_{t|t} + L_t(P_{t+1|T} - P_{t+1|t})L_t^T$$

Learning SSMs

- Complete log likelihood

$$\ell_c(\theta, D) = \sum_n \log p(x_n, y_n) = \sum_n \log p(x_1) + \sum_n \sum_t \log p(x_{n,t} \mid x_{n,t-1}) + \sum_n \sum_t \log p(y_{n,t} \mid x_{n,t})$$

$$= f_1(X_1; \Sigma_0) + f_2\left(\left\{X_t X_{t-1}^T, X_t X_t^T, X_t : \forall t\right\}, A, Q, G\right) + f_3\left(\left\{X_t X_t^T, X_t : \forall t\right\}, C, R\right)$$

- EM
  - E-step: compute $\left\langle X_t X_{t-1}^T \right\rangle, \left\langle X_t X_t^T \right\rangle, \left\langle X_t \right\rangle \Big| y_1, \dots y_T$

    these quantities can be inferred via KF and RTS filters, etc.,
    e,g., $\left\langle X_t X_t^T \right\rangle \equiv \mathrm{var}(X_t X_t^T) + \mathrm{E}(X_t)^2 = P_{t|T} + \hat{x}_{t|T}^2$

  - M-step: MLE using

    $\left\langle \ell_c(\theta, D) \right\rangle = f_1(\langle X_1 \rangle; \Sigma_0) + f_2\left(\left\{\left\langle X_t X_{t-1}^T \right\rangle, \left\langle X_t X_t^T \right\rangle, \langle X_t \rangle : \forall t\right\}, A, Q, G\right) + f_3\left(\left\{\left\langle X_t X_t^T \right\rangle, \langle X_t \rangle : \forall t\right\}, C, R\right)$

    c.f., M-step in factor analysis

Nonlinear systems

- In robotics and other problems, the motion model and the observation model are often nonlinear:

$$x_t = f(x_{t-1}) + w_t \quad , \qquad y_t = g(x_t) + v_t$$

- 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.

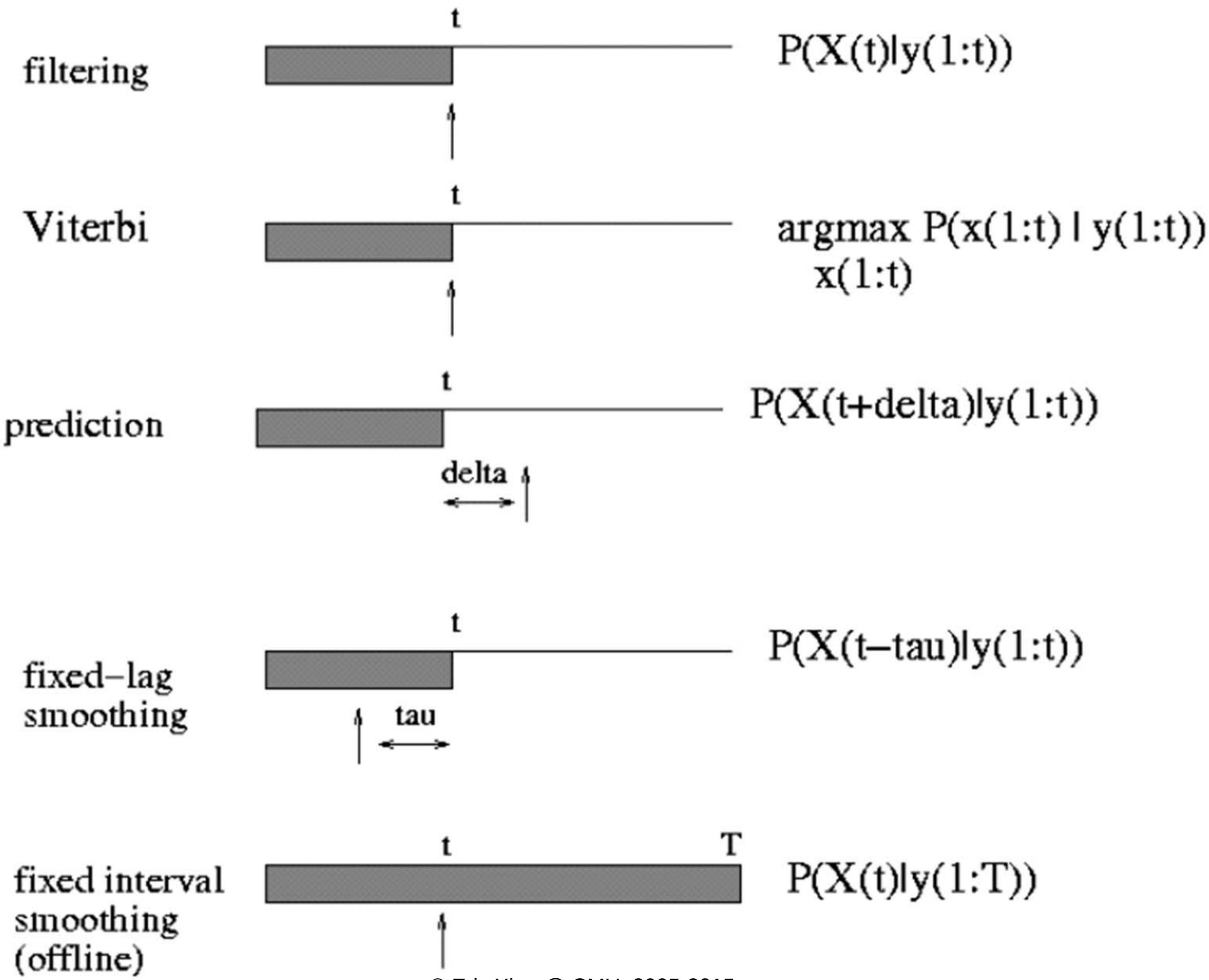
$$x_t = f(\hat{x}_{t-1|t-1}) + A_{\hat{x}_{t-1|t-1}}(x_{t-1} - \hat{x}_{t-1|t-1}) + w_t$$

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

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

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

Online vs offline inference



KF, RLS and LMS

- 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 η_t online using stochastic approximation theory.