DS-GA 3001.001

Probabilistic time series analysis

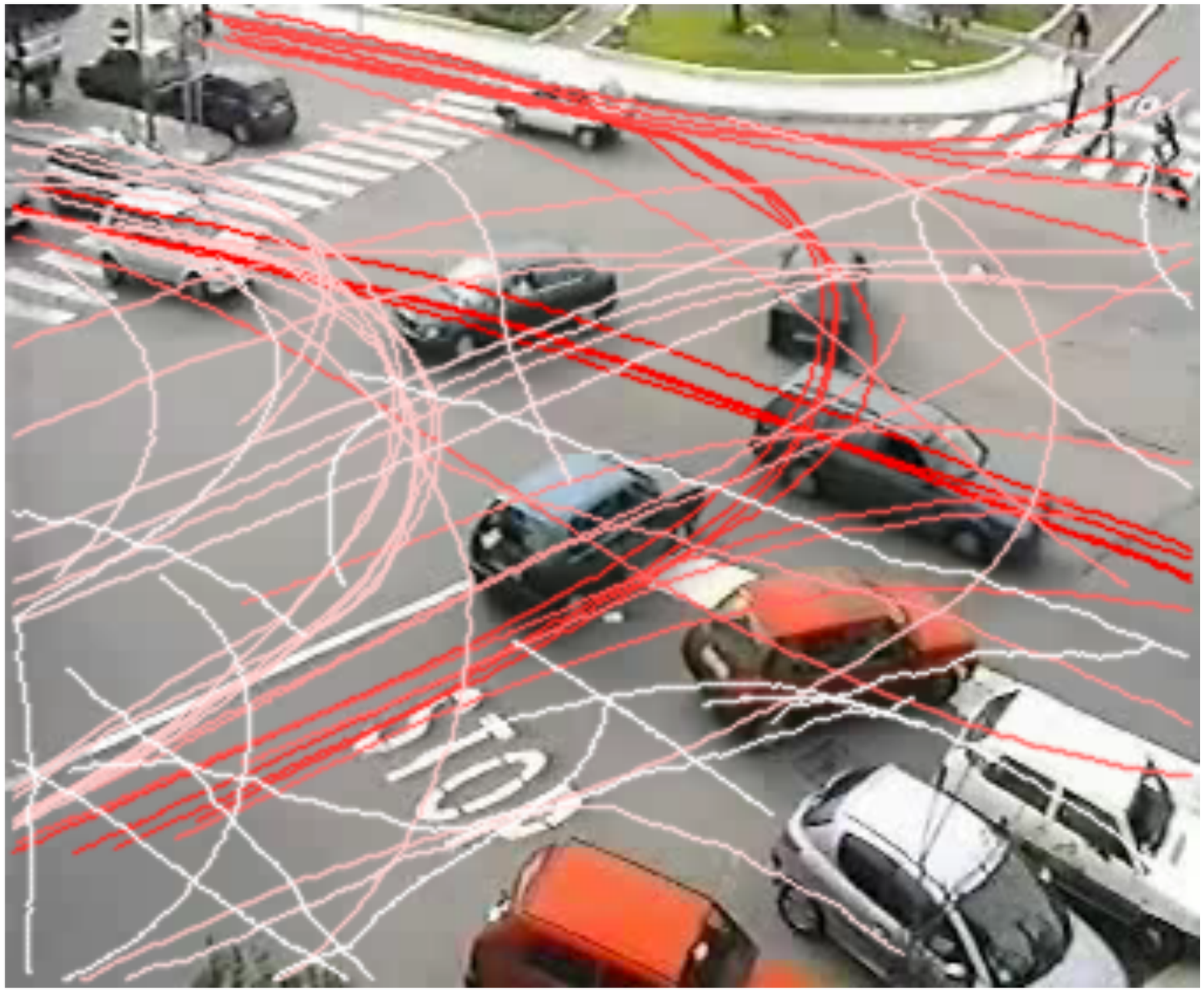
Lecture 3

Latent state model: Kalman filtering

Instructor: Cristina Savin

NYU, CNS & CDS

Most interesting data is complex and **multidimensional**



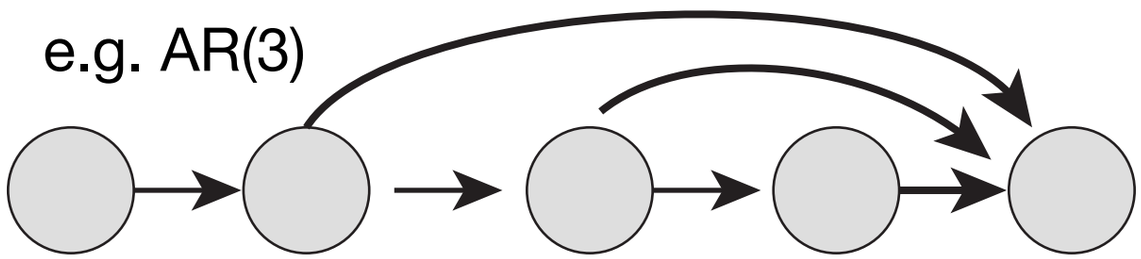
From Saunier & Sayed 2008

Directly model the dependencies across time, under certain **restrictions**.

Tractable

ARIMA

e.g. AR(3)

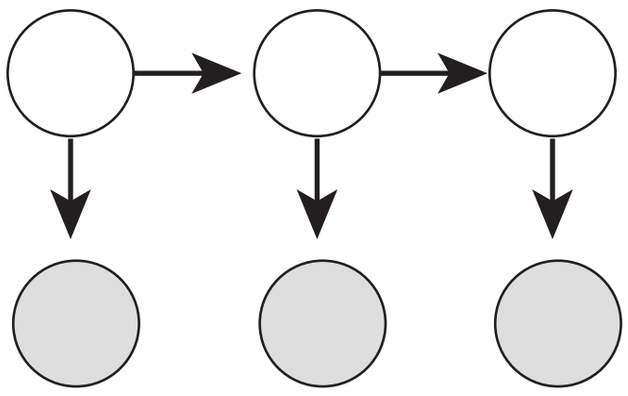


linear dependencies
stationary*
scalar variables

In the latent space the temporal dependencies are simple:

1st order Markov

Latent state models

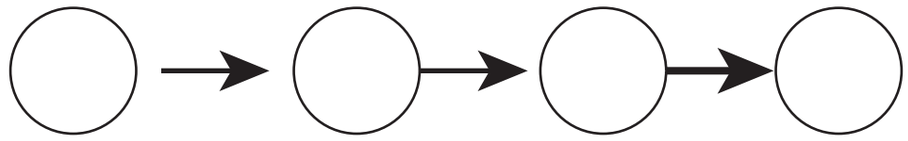


Goal: find a **latent** variable
that summarizes
the relevant **history**
while keeping math simple

The latent variable is either discrete for **hidden Markov models** (HMMs) or continuous for **latent dynamical systems** (LDS)

Markov property

e.g. first order Markov process



A sequence has the Markov property if, once we know the most recent value $\mathbf{x}_{t-1}$ the current state $\mathbf{x}_t$ is independent on the past history.

$$\mathrm{P}\left(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_{t-2}, \cdots, \mathbf{x}_0\right) = \mathrm{P}(\mathbf{x}_t|\mathbf{x}_{t-1})$$

This means that we have a simple factorization for the full sequence:

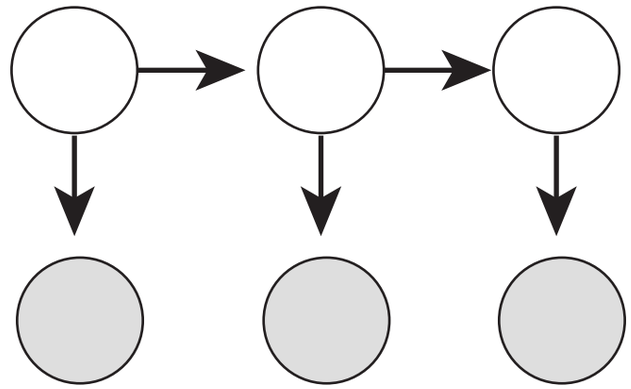
$$\mathrm{P}\left(\mathbf{x}_t, \mathbf{x}_{t-1}, \cdots, \mathbf{x}_0\right) = \prod_k \mathrm{P}\left(\mathbf{x}_k|\mathbf{x}_{k-1}\right)$$

Nice if individual components are simple (exponential family)

Reminder: chain rule

$$\mathrm{P}\left(\mathbf{x}_t, \mathbf{x}_{t-1}, \cdots, \mathbf{x}_0\right) = \prod_k \mathrm{P}\left(\mathbf{x}_k|\mathbf{x}_{k-1}, \cdots, \mathbf{x}_0\right)$$

Some useful conditional independencies

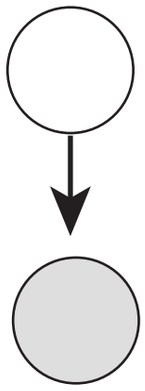


$$
\begin{aligned}
\mathrm{P}\left(\mathbf{x}_{1:t}|\mathbf{z}_i\right) &= \mathrm{P}\left(\mathbf{x}_{1:i}|\mathbf{z}_i\right)\mathrm{P}\left(\mathbf{x}_{i+1:t}|\mathbf{z}_i\right) \\
\mathrm{P}\left(\mathbf{x}_{1:i}|\mathbf{z}_{i+1},\mathbf{x}_{i+1}\right) &= \mathrm{P}\left(\mathbf{x}_{1:i}|\mathbf{z}_{i+1}\right) \\
\mathrm{P}\left(\mathbf{x}_{i+2:t}|\mathbf{z}_{i+1},\mathbf{x}_{i+1}\right) &= \mathrm{P}\left(\mathbf{x}_{i+2:t}|\mathbf{z}_{i+1}\right) \\
\mathrm{P}\left(\mathbf{x}_{1:i}|\mathbf{z}_i,\mathbf{z}_{i+1}\right) &= \mathrm{P}\left(\mathbf{x}_{1:i}|\mathbf{z}_i\right) \\
\mathrm{P}\left(\mathbf{x}_{i+1:t}|\mathbf{z}_i,\mathbf{z}_{i+1}\right) &= \mathrm{P}\left(\mathbf{x}_{i+1:t}|\mathbf{z}_{i+1}\right) \\
\mathrm{P}\left(\mathbf{x}_{1:t}|\mathbf{z}_i,\mathbf{z}_{i+1}\right) &= \mathrm{P}\left(\mathbf{x}_{1:i}|\mathbf{z}_i\right)\mathrm{P}\left(\mathbf{x}_{i+1}|\mathbf{z}_{i+1}\right)\mathrm{P}\left(\mathbf{x}_{i+2:t}|\mathbf{z}_{i+1}\right)
\end{aligned}
$$

https://www.youtube.com/watch?v=ofHn99_uBwA

Kalman filtering

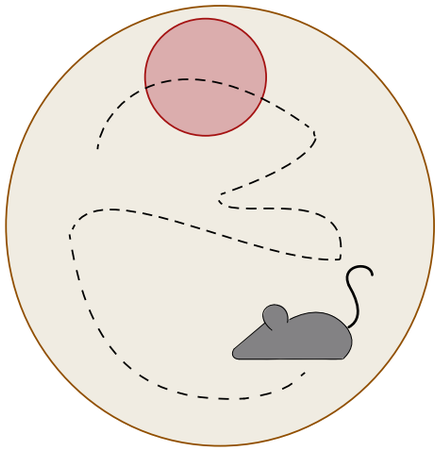
Kalman filtering motivation

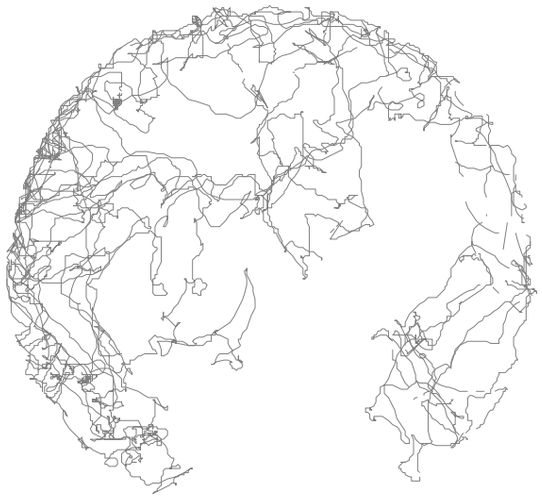


If I want to determine the value of a variable z,
based on a noisy measurement x,
I just take multiple measurements to get
a better estimate

Can’t do that if z changes over time,
but there is still hope if z changes slowly
(intuitively, some sort of weighted average)

Applications include: tracking objects (e.g., people, hands), navigation
Computer vision: tracking features in video, stabilizing depth measurements,
and many more





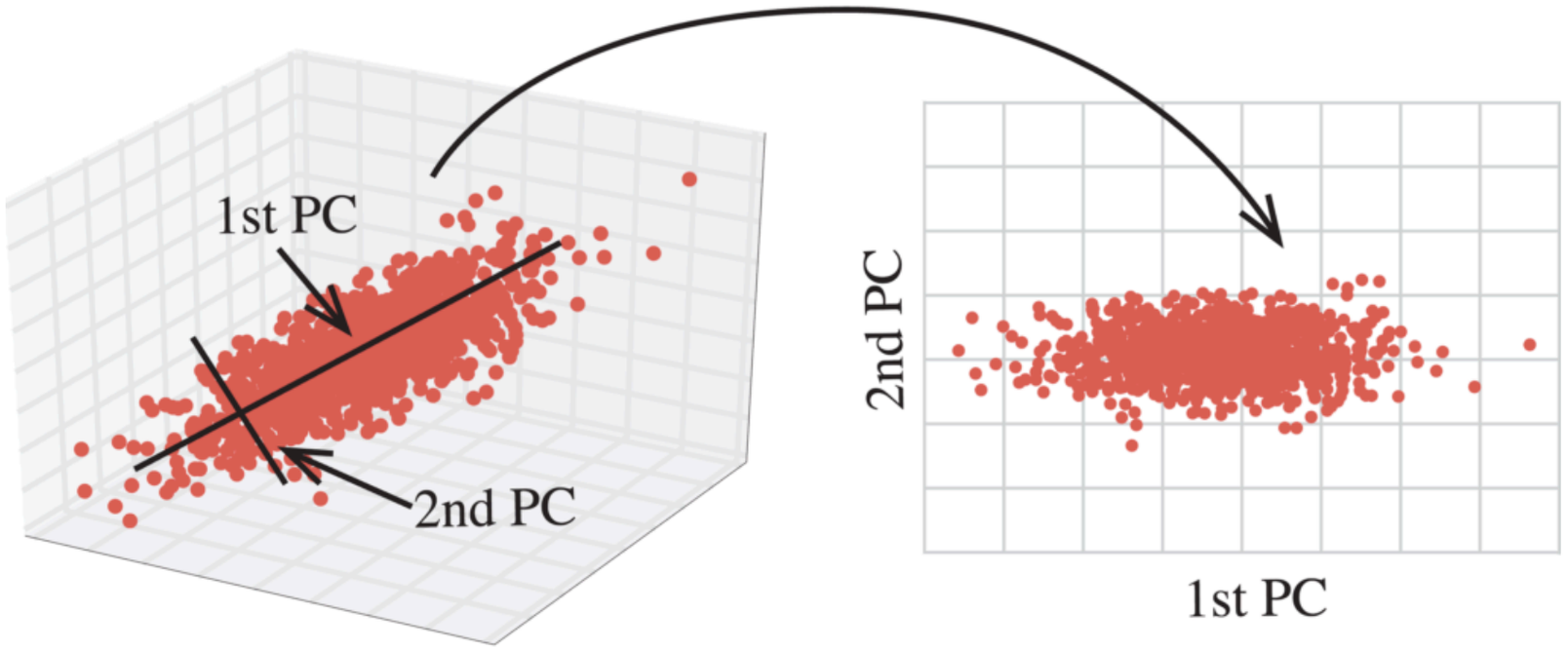
E.g. noisy measurements
of the animal's
position from top camera

OpenCV demo

Application : dimensionality reduction

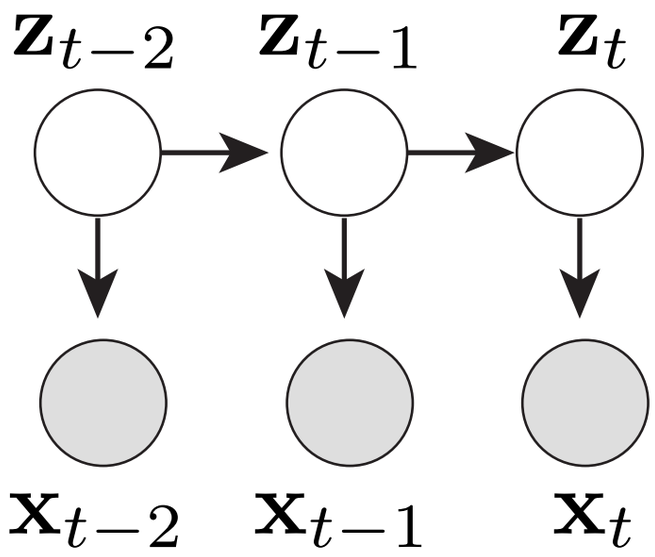
Often we have d < n, i.e. the data itself is low-dimensional.

Can think about this as a time-dependent version of PCA
(we'll make this link formal later in the course)



Play with in the lab!

Latent state models



d-dimensional

n-dimensional

$$\mathbf{z}_t = \mathbf{A}\mathbf{z}_{t-1} + \mathbf{w}_t$$
$$\mathbf{x}_t = \mathbf{C}\mathbf{z}_t + \mathbf{v}_t$$

Where the noise terms are iid Gaussian:

$$\mathbf{w}_t \sim \mathcal{N}(0, \mathbf{Q})$$
$$\mathbf{v}_t \sim \mathcal{N}(0, \mathbf{R})$$
$$\mathbf{z}_0 \sim \mathcal{N}(\mu_0, \Sigma)$$

We are still in a happy linear gaussian world!

The noise terms are essential elements of the model : without **w**, deterministic behavior given by leading eigenvalue of **A**; without **v**, no latents

Without loss of generality, we can take **Q** to be the identity*, **R** can be any positive definite matrix

*If Q is not diagonal, it is still positive definite, so it can be diagonalized $\mathbf{Q} = \mathbf{E}\mathbf{D}\mathbf{E}^{\mathbf{T}}$

With the variable change (rotating axes) we can always rewrite this so that the noise term is the identity $\mathbf{y} = \mathbf{D}^{-1/2}\mathbf{E}^{\mathbf{T}}\mathbf{x}$

LDS inference

We are given the data up to *s*, $\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_s\}$.

We want to estimate the latent state, $\mathbf{z}_t$.

If t>s this is prediction, if s=t **filtering**, if t<s **smoothing.**

$$\mathbf{z}_t|\mathbf{z}_{t-1} \sim \mathcal{N}(\mathbf{A}\mathbf{z}_{t-1}; \mathbf{Q})$$
$$\mathbf{x}_t|\mathbf{z}_t \sim \mathcal{N}(\mathbf{C}\mathbf{z}_t; \mathbf{R})$$

$$\mathrm{P}(\mathbf{z}_*, \mathbf{x}_*) = \mathrm{P}(\mathbf{z}_*)\prod_i \mathrm{P}(\mathbf{x}_i|\mathbf{z}_i)$$

* marks full series

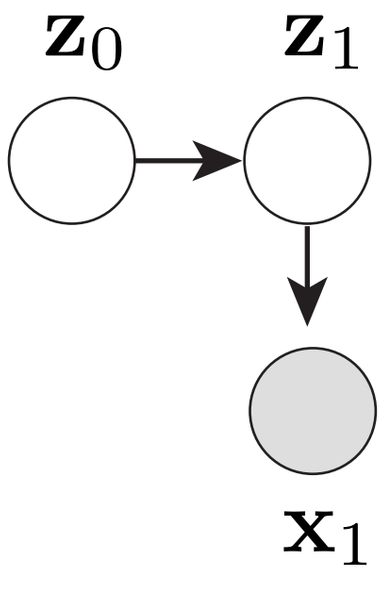
$$\mathrm{P}(\mathbf{z}_*) = \mathrm{P}(\mathbf{z}_0)\prod_i \mathrm{P}(\mathbf{z}_t|\mathbf{z}_{t-1})$$

Everything is Gaussian, so we could explicitly write it down.

Kalman filtering uses recursion to compute things more efficiently.

Note: This can be derived as a specific instance of message passing in trees (see e.g. Bishop, chp.6), but we're not going to do this here.

Intuitively, we can construct the forward pass by induction



2 sub-steps:

$\mathbf{z}_1 \big| \mathbf{z}_0$ **Take into account what we know about z_0**

$\mathbf{z}_1 \big| \mathbf{x}_1, \mathbf{z}_0$ **Take into account evidence x_1**

Basic Gaussian identities

a d-dimensional multidimensional gaussian (normal) density for $\mathbf{x}$ is:

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-d/2}|\boldsymbol{\Sigma}|^{-1/2} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]$$

E1. Rescaling and shifting gaussian variables

$$\mathrm{E}[\mathbf{A}\mathbf{x}+\mathbf{y}] = \mathbf{A}(\mathrm{E}[\mathbf{x}])+\mathbf{y}$$
$$\mathrm{Covar}[\mathbf{A}\mathbf{x}+\mathbf{y}] = \mathbf{A}(\mathrm{Covar}[\mathbf{x}])\mathbf{A}^T$$

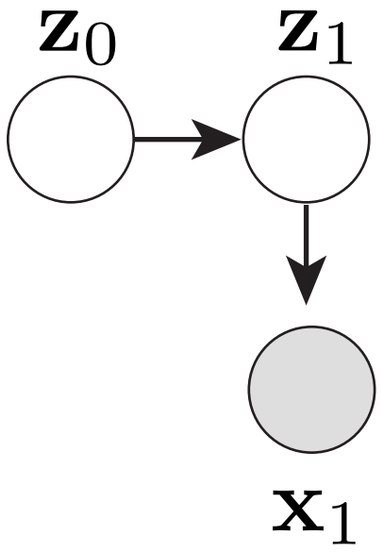
E2. Sum of 2 independent gaussian variables

$$\Sigma_c = \Sigma_a + \Sigma_b$$
$$\mu_c = \mu_a + \mu_b$$

E3. Conditioning

$$\mathbf{z} = \begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}\mathbf{a}\\ \mathbf{b}\end{bmatrix}, \begin{bmatrix}\mathbf{A} & \mathbf{C}\\ \mathbf{C}^T & \mathbf{B}\end{bmatrix}\right)$$
$$\mathbf{x}|\mathbf{y} \sim \mathcal{N}\left(\mathbf{a}+\mathbf{C}\mathbf{B}^{-1}(\mathbf{y}-\mathbf{b}), \mathbf{A}-\mathbf{C}\mathbf{B}^{-1}\mathbf{C}^T\right)$$

Intuitively, we can construct the forward pass by induction



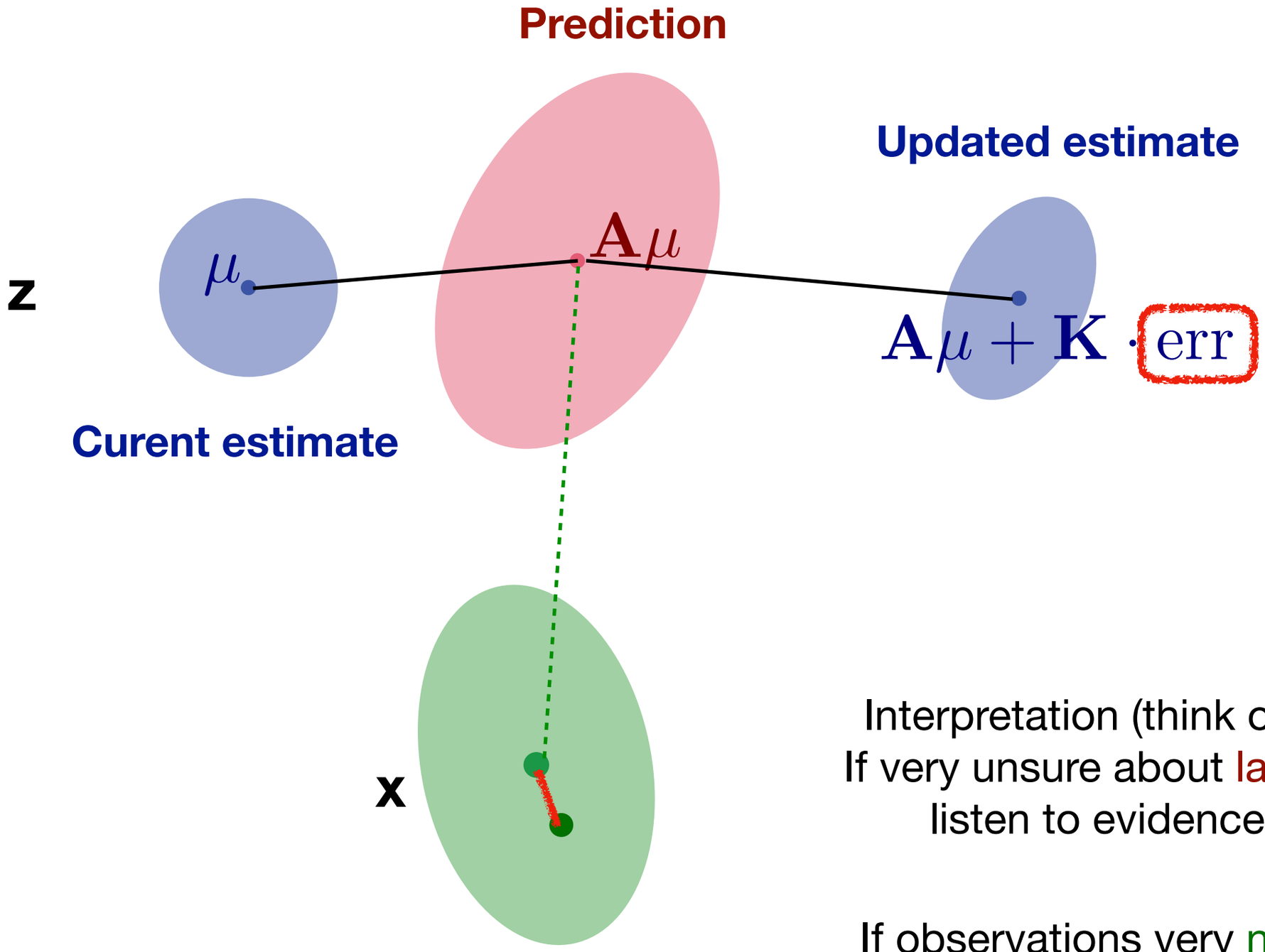
2 sub-steps:

Forecasting

$$\mathbf{z}_1|\mathbf{z}_0 \qquad \mu_{1|0} = A\mu_0$$
$$\Sigma_{1|0} = A\Sigma_0 A^{\mathrm{T}} + Q$$

Combine with new evidence

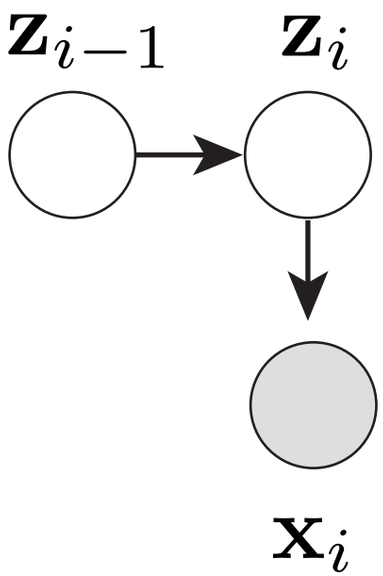
$$\mathbf{z}_1|\mathbf{x}_1,\, \mathbf{z}_0 \qquad \mu_{1|1} = \mu_{1|0} + \mathbf{K}(\mathbf{x}_1 - \mathbf{C}\mu_{1|0})$$
$$\Sigma_{1|1} = \Sigma_{1|0} - \mathbf{K}\mathbf{C}\Sigma_{1|0}$$
$$\mathbf{K} = \Sigma_{1|0}\mathbf{C}^{\mathrm{T}}(\mathbf{C}\Sigma_{1|0}\mathbf{C}^{\mathrm{T}} + R)^{-1}$$



Interpretation (think of scales):
If very unsure about latent state,
listen to evidence more

If observations very nosy (R large),
listen to evidence less

$$\mathbf{K} = \Sigma_{1|0}\mathbf{C}^{\mathrm{T}}(\mathbf{C}\Sigma_{1|0}\mathbf{C}^{\mathrm{T}} + R)^{-1}$$



Exactly in the same way, if we have a posterior distribution for $\mathbf{z}_{i-1}$, we can use it instead of $P(\mathbf{z}_0)$ to compute the posterior for $\mathbf{z}_i$.

$$\mu_{i|i} = \mu_{i|i-1} + \mathbf{K}_i(\mathbf{x}_i - \mathbf{C}\mu_{i|i-1})$$

$$\Sigma_{i|i} = \Sigma_{i|i-1} - \mathbf{K}_i\mathbf{C}\Sigma_{i|i-1}$$

$$\mathbf{K}_i = \Sigma_{i|i-1}\mathbf{C}^{\mathrm{T}}(\mathbf{C}\Sigma_{i|i-1}\mathbf{C}^{\mathrm{T}} + R)^{-1}$$ **Kalman gain**

The gain changes dynamically!

Notation: double index i|j:
i-we are estimating variable $\mathbf{z}_i$,
j-datapoints observed up to (including) $\mathbf{y}_j$

Smoothing - backward sweep

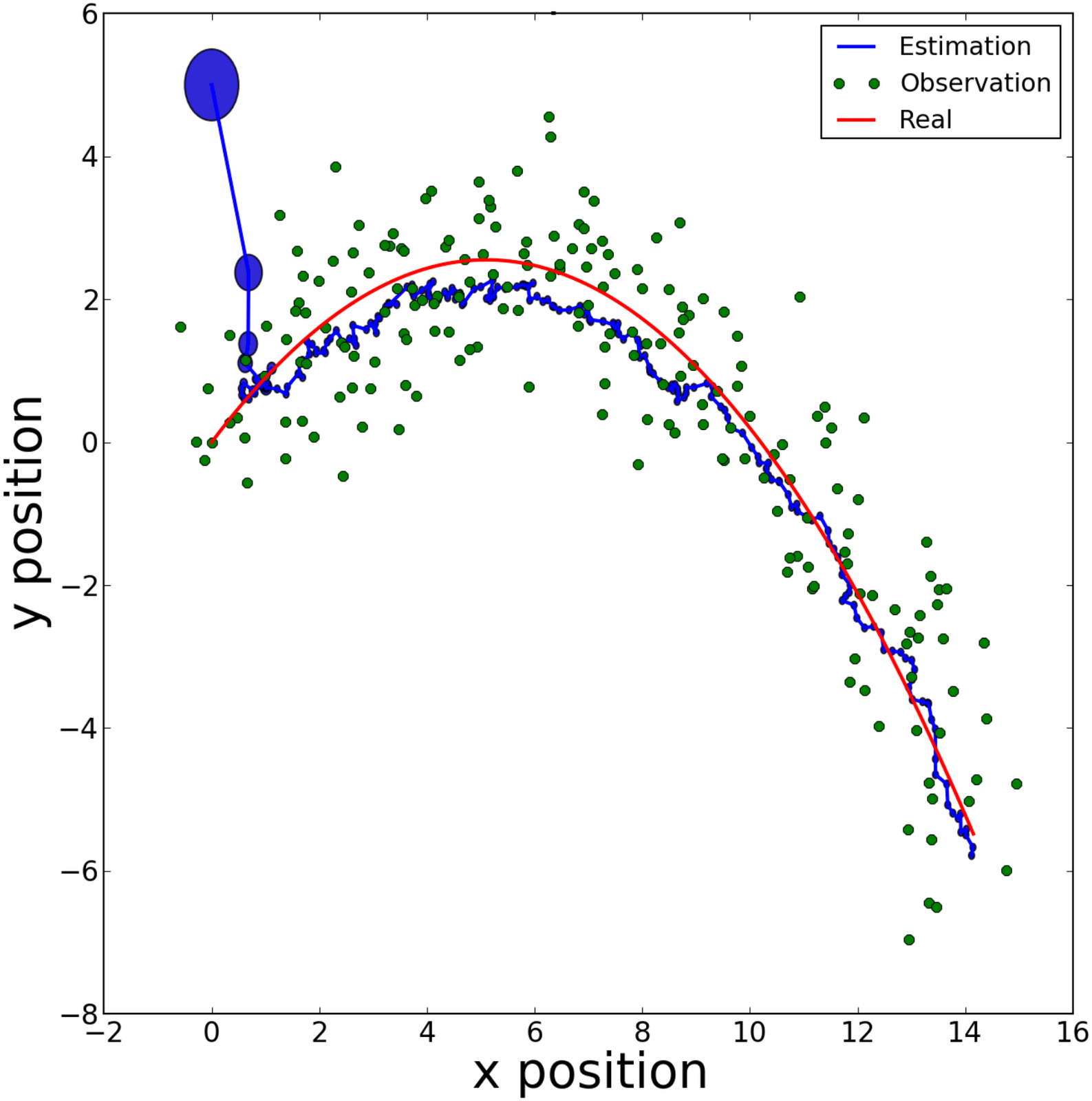
Same principle, but now we propagate information backwards in time, using the estimates we have already (we no longer need the data)

$$\mu_{i|t} = \mu_{i|i} + \mathbf{F}_i(\mu_{i+1|t} - \mu_{i+1|i})$$

$$\Sigma_{i|t} = \mathbf{F}_i(\Sigma_{i+1|t} - \Sigma_{i+1|i})\mathbf{F}_i^{\mathrm{T}} + \Sigma_{i|i}$$

$$\mathbf{F}_i = \Sigma_{i|i}\mathbf{A}^{\mathrm{T}}\Sigma_{i+1|i}^{-1}$$

See handout for proof


Estimation
Observation
Real
y position
x position
6
4
2
0
−2
−4
−6
−8
−2
0
2
4
6
8
10
12
14
16

How do we use this for prediction?

How do we use learn the parameters?

Estimating parameters via expectation maximization

Find parameters that are most consistent with the observed data

To simplify notation we use shorthand $\theta = \{\mathbf{A},\, \mathbf{C},\, \mathbf{Q},\, \mathbf{R}, \mu_0, \Sigma_0\}$

The goal is to find the parameters that maximize the (log) likelihood

$$\mathcal{L}(\theta) = \log \mathrm{P}(\mathbf{x}_*|\theta)$$

Which we get after **marginalizing out the latents**

$$\mathrm{P}(\mathbf{x}_*|\theta) = \int \mathrm{P}(\mathbf{x}_*, \mathbf{z}_*|\theta)\mathbf{dz}$$

General idea of EM

$$\log \int_z \mathrm{P}(\mathbf{x}, \mathbf{z}|\theta)\mathbf{dz} = \log \int_z Q(\mathbf{z}) \frac{\mathrm{P}(\mathbf{x},\mathbf{z}|\theta)}{Q(\mathbf{z})}\mathbf{dz} \quad = \log\left(\mathbb{E}_Q\left[\frac{\mathrm{P}(\mathbf{x},\mathbf{z}|\theta)}{Q(\mathbf{z})}\right]\right)$$

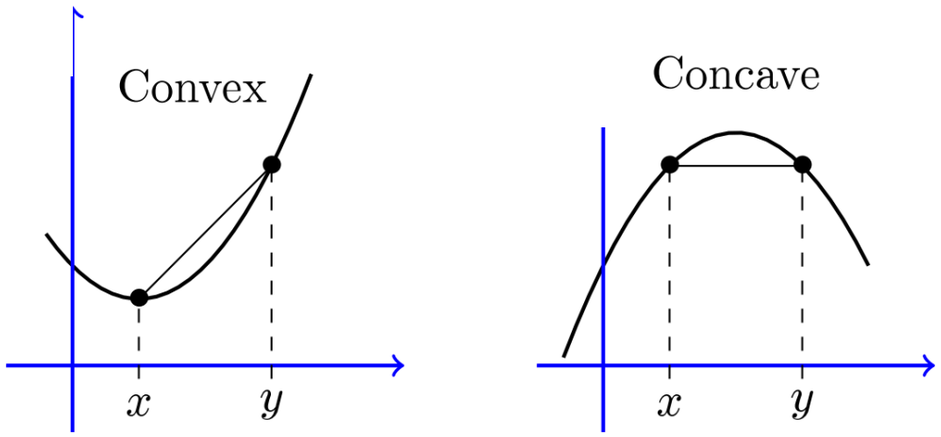
via **Jensen's Inequality** $\geq \int_z Q(\mathbf{z}) \log \frac{\mathrm{P}(\mathbf{x},\mathbf{z}|\theta)}{Q(\mathbf{z})}\mathbf{dz}$ $= \mathbb{E}_Q\left[\log \frac{\mathrm{P}(\mathbf{x},\mathbf{z}|\theta)}{Q(\mathbf{z})}\right]$

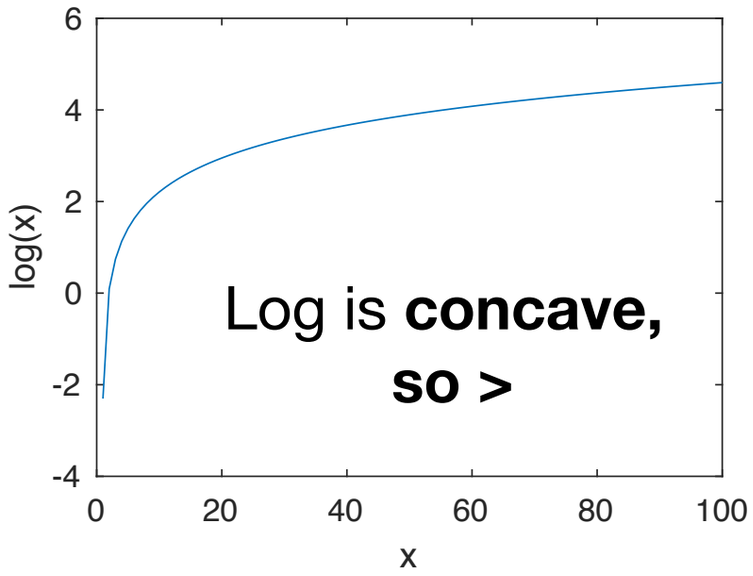
$$= \int_Q Q(\mathbf{z}) \log \mathrm{P}(\mathbf{x}, \mathbf{z}|\theta))\mathbf{dz} - \int_Q Q(\mathbf{z}) \log Q(\mathbf{z})\mathbf{dz}$$

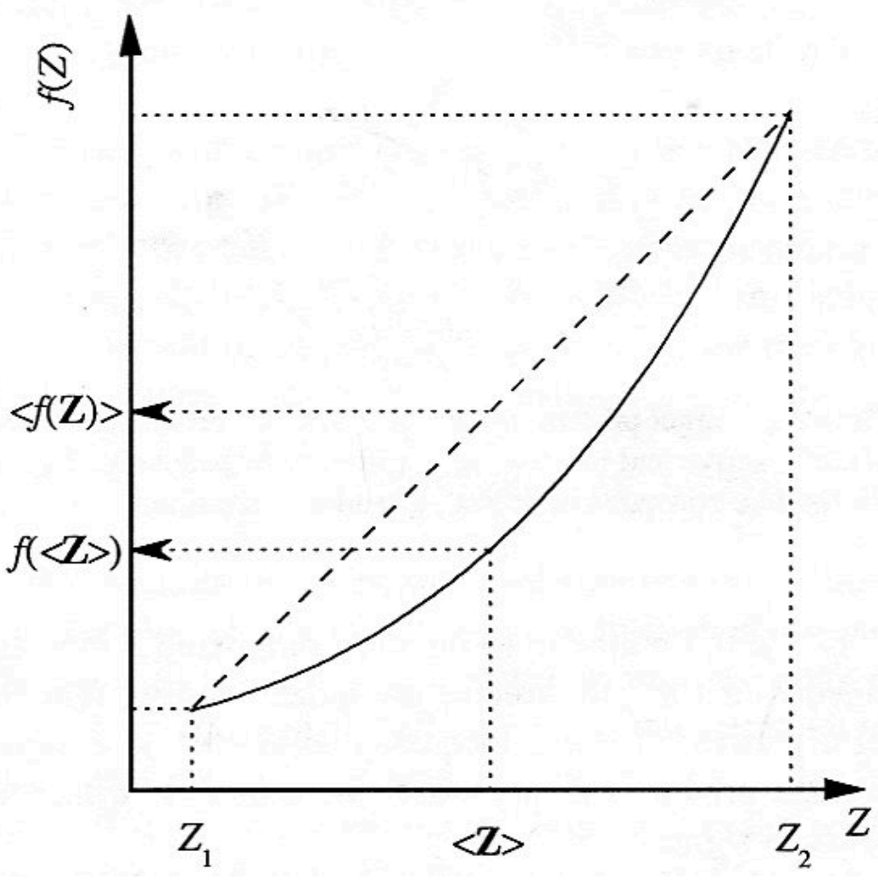
$$= \mathcal{F}(Q, \theta)$$ **Free energy**

Big picture: log of integral is not nice, so we replace it with bound on integral of log

Reminder: convex vs concave functions; Jensen's inequality







$$f(\mathbb{E}[x]) \leq \mathbb{E}[f(x)]$$

If f is **convex**

EM alternates between finding a good approximation Q,
and then changing the parameters to improve the approx likelihood

This is done coordinate-wise: improve one keeping the other fixed, then swap.

$$\textbf{E step:} \qquad Q_{k+1} \leftarrow \underset{Q}{\arg\max} \ \mathcal{F}(Q, \theta_k)$$

$$\textbf{M step:} \qquad \theta_{k+1} \leftarrow \underset{\theta}{\arg\max} \ \mathcal{F}(Q_{k+1}, \theta)$$

It is guaranteed that the likelihood will **never decrease** during this procedure.

General philosophy:
use simple parametric form to ‘guesstimate’ the state of the
latent variables, given current model parameters. Then use this fictitious
complete data to find new model parameters.

E-step: $Q_{k+1}(\mathbf{z}) = \mathrm{P}(\mathbf{z}|\mathbf{x}, \theta_k)$

The approximation is locally exact