good job with HW1!

Project proposals due **Febr. 27,** instructions on course page

HW2 (LDS) is online, due next week **March 2**

use the LDS handouts!

HW3 (HMMs) will also be posted this week, due next week **March 22**

may want to look at these before the midterm,
use the HMM handouts!

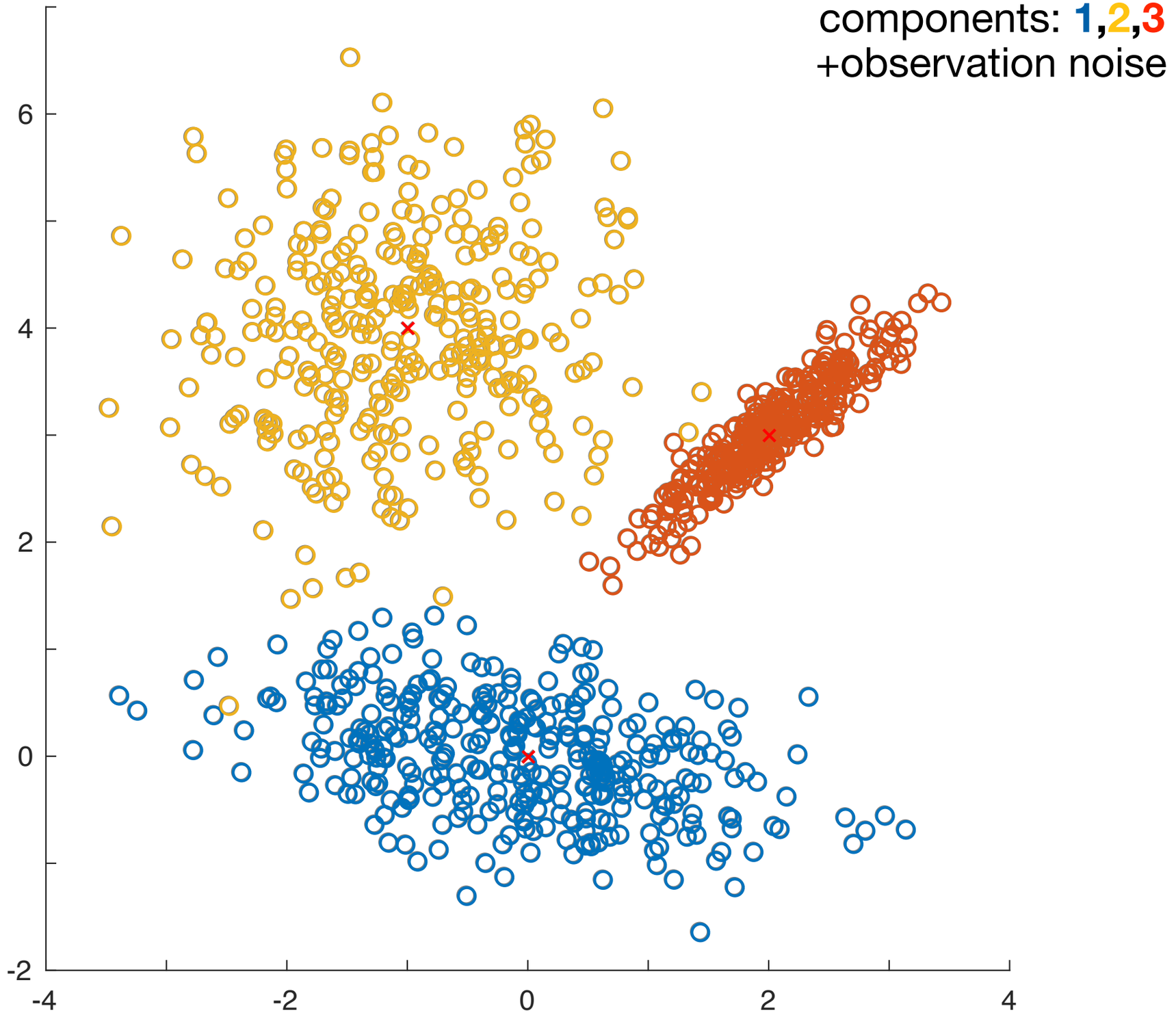
DS-GA 3001.008 Modelling time series data

L5. Hidden Markov models

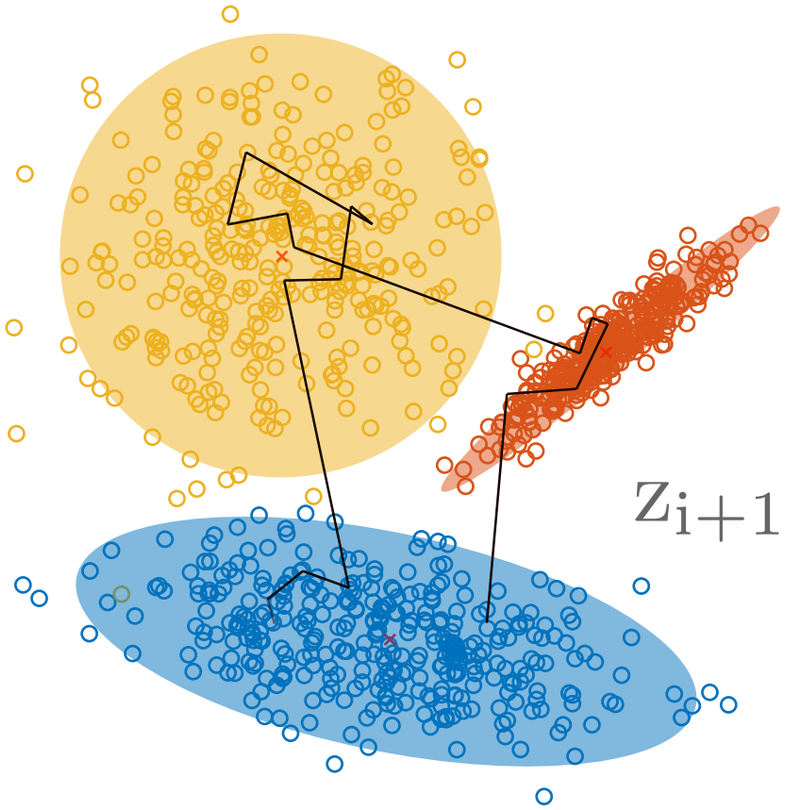
Instructor: Cristina Savin

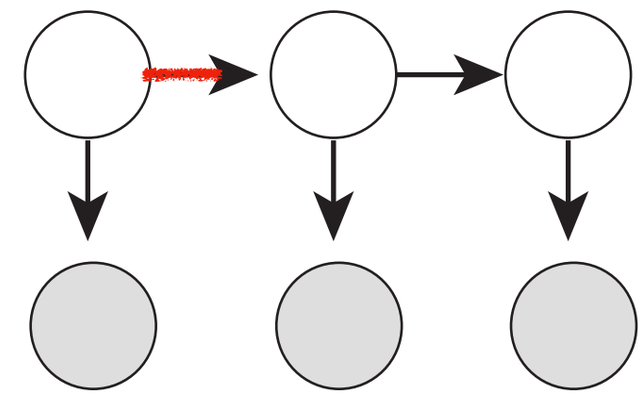
NYU, CNS & CDS

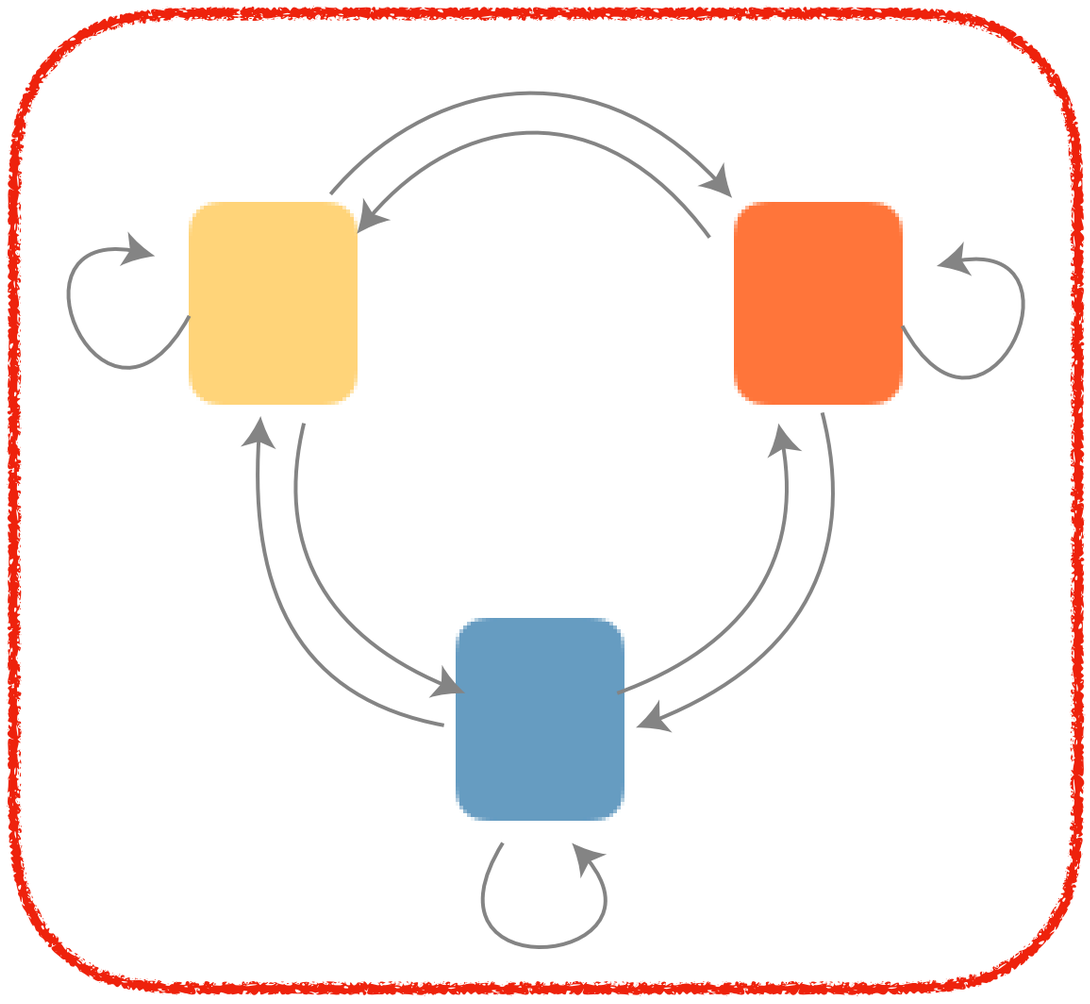
Reminder: mixture models



Hidden Markov models: mixture models, with **temporal dependencies** for components







Markov(1913) used frequency counts to compute the probability that the next letter is a vowel?

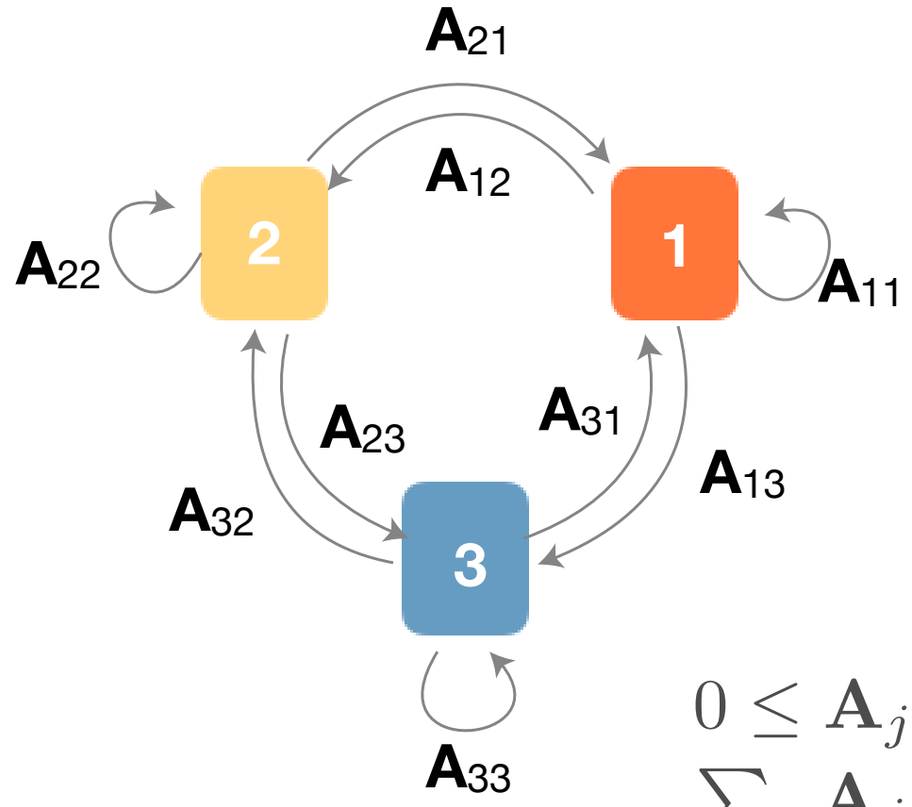
Applications:
speech recognition
natural language modelling
online handwritten recognition
structure of proteins, DNA

Hidden Markov models: details

K district components, $z_i = \{1,2,\ldots,K\}$

Transition probabilities: **A**

$$A_{jk} = \mathrm{P}(z_{i+1} = k | z_i = j)$$



$$0 \leq \mathbf{A}_{j,k} \leq 1$$
$$\textstyle\sum_k \mathbf{A}_{j,k} = 1$$

Alternative 1-in-K binary representation*:

$\mathbf{z}_i$ - K-dimensional binary vector

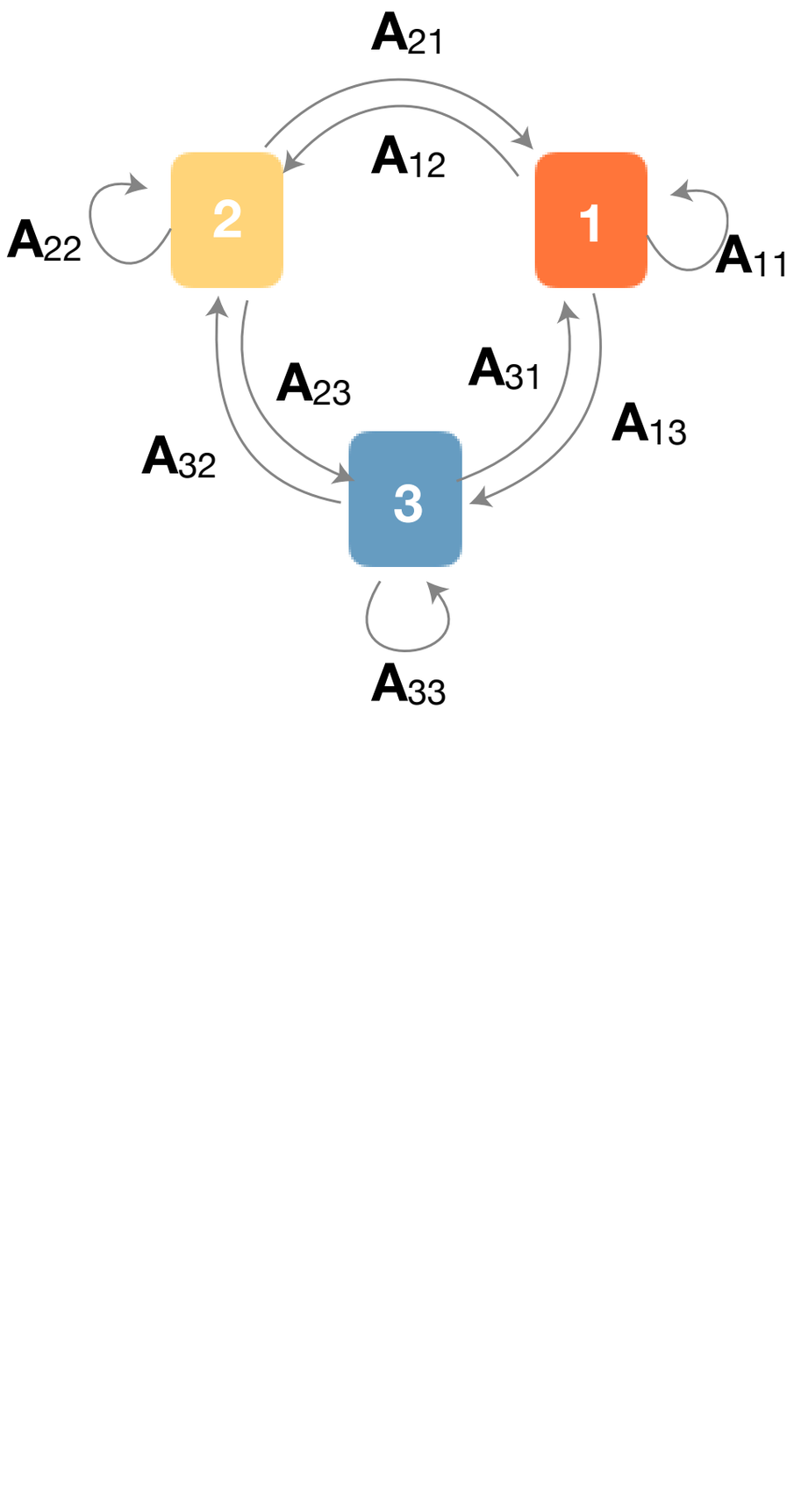
$$A_{jk} = \mathrm{P}(\mathbf{z}_{i+1,k} = 1 | \mathbf{z}_{i,j} = 1)$$

$$\mathrm{P}(\mathbf{z}_{i+1} = 1 | \mathbf{z}_i = 1) = \textstyle\prod_{j,k} \mathbf{A}_{j,k}^{z_{i,j} \cdot z_{i+1,k}}$$

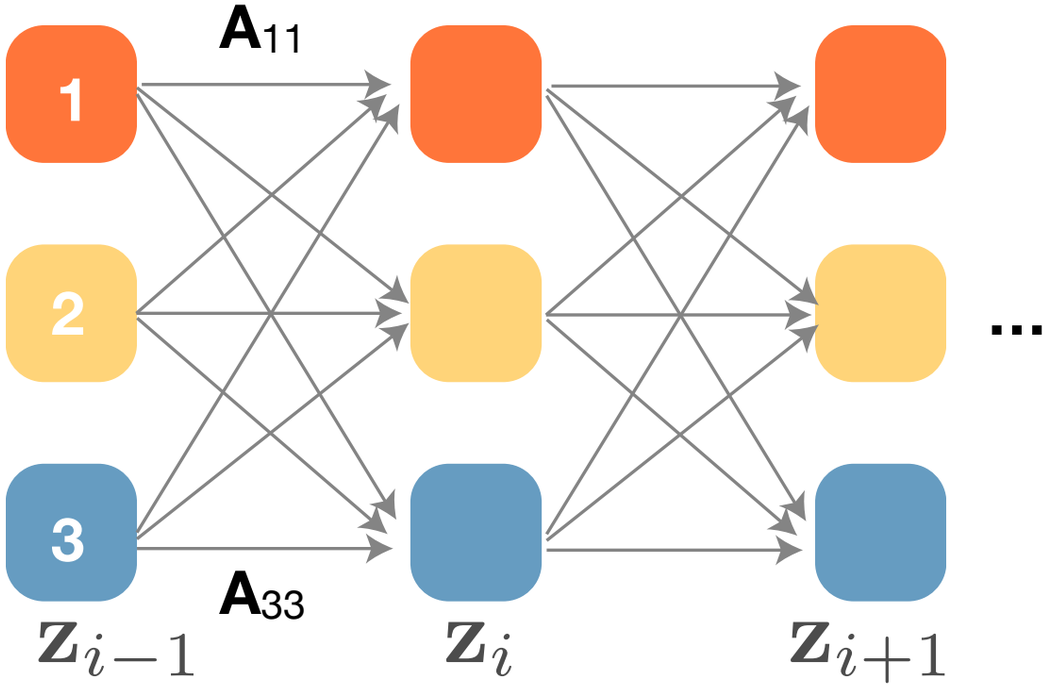
Initial state, $\mathbf{z}_0$ $\quad \mathrm{P}(\mathbf{z}_1) = \prod_k \pi_k^{z_{1,k}}$

Emissions model for observations $\mathbf{x}_i$ $\quad \mathrm{P}(\mathbf{x}_i | \mathbf{z}_i)$

*Note: This seems unnecessarily complicated at this point, but will turn out convenient when we want to represent beliefs about the state of the latent variables

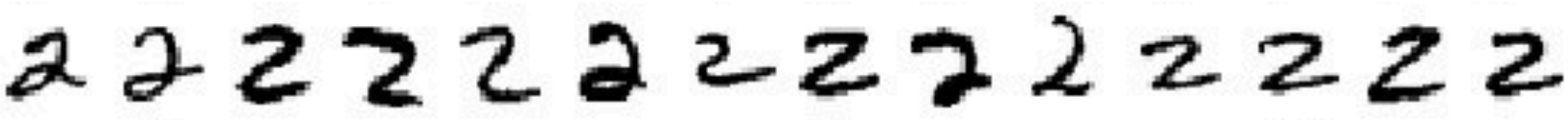


unfold over time:

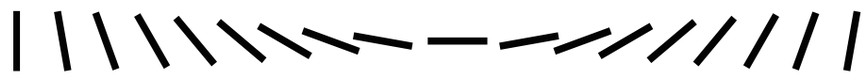


trellis representation

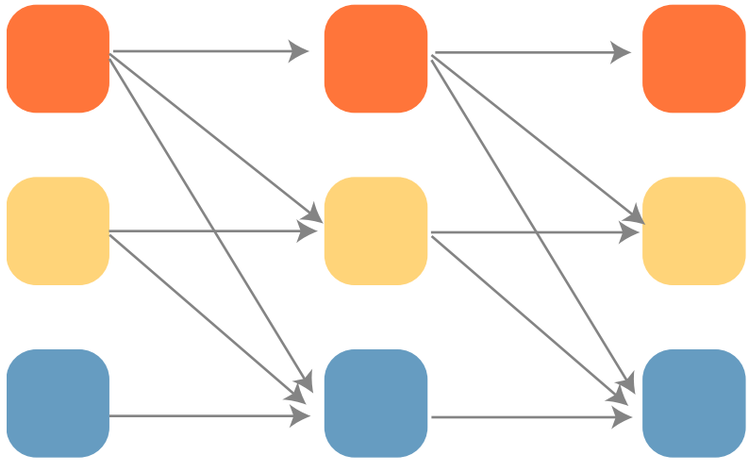
Example: real time digit recognition



z: lines fixes length, different orientations



natural ordering of components:
left-to-right HMMs



probabilistic transition between components: rescaling, variation in length

1. Inference

the full posterior is $\mathrm{P}(\mathbf{z}_{1:t}|\mathbf{x}_{1:t})$

$\mathbf{z}_{1:t}$ has K^t possible configurations, so computationally intractable

What we can do is:
compute exact **marginals**: alpha-beta algorithm
(as we did in LDS with Kalman smoother)

We may also want to determine the most likely sequence **(MAP)**
Viterbi algorithm

2. Parameter learning

This is just more **EM**

1. Inference: marginals

We represent posterior marginals and joints using

$$\begin{aligned} \gamma(\mathbf{z}_i) &= \mathrm{P}\left(\mathbf{z}_i|\mathbf{x}_*, \theta^{\mathrm{old}}\right) \\ \xi(\mathbf{z}_i, \mathbf{z}_{i+1}) &= \mathrm{P}\left(\mathbf{z}_i, \mathbf{z}_{i+1}|\mathbf{x}_*, \theta^{\mathrm{old}}\right) \end{aligned}$$

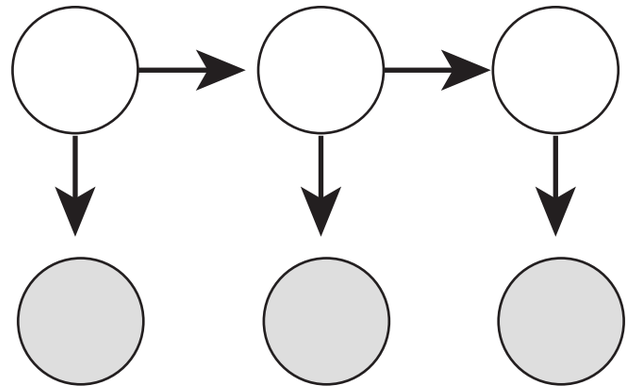
dimensionality?

With the corresponding expectations:

$$\begin{aligned} \gamma(z_{ik}) &= \mathbb{E}[z_{i,k}|\mathbf{x}_*, \theta^{\mathrm{old}}] = \sum_{\mathbf{z}_i} \gamma(\mathbf{z}_i) z_{i,k} \\ \xi(z_{i,j}, z_{i+1,k}) &= \mathbb{E}[z_{i,j} z_{i+1,k}|\mathbf{x}_*, \theta^{\mathrm{old}}] = \sum_{\mathbf{z}_i, \mathbf{z}_{i+1}} \xi(\mathbf{z}_i, \mathbf{z}_{i+1}) z_{i,j} z_{i+1,k} \end{aligned}$$

We seek an efficient way of computing these by using recursions
(dynamic programming, as for LDS)

Some useful conditional independencies



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

*Note: some intuitions using graphical model on board

First, let's look at marginals

$$\mathrm{P}\left(\mathbf{z}_i|\mathbf{x}_{1:t},\theta^{\mathrm{old}}\right) = \frac{\mathrm{P}\left(\mathbf{x}_{1:t}|\mathbf{z}_i\right)\mathrm{P}\left(\mathbf{z}_i\right)}{\mathrm{P}\left(\mathbf{x}_{1:t}\right)} = \frac{\mathrm{P}\left(\mathbf{x}_{1:i},\mathbf{z}_i\right)\mathrm{P}\left(\mathbf{x}_{i+1:t}|\mathbf{z}_i\right)}{\mathrm{P}\left(\mathbf{x}_{1:t}\right)} = \frac{\alpha(\mathbf{z}_i)\beta(\mathbf{z}_i)}{\mathrm{P}\left(\mathbf{x}_{1:t}\right)}$$

where we defined the 'messages'

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

And the normalizing constant is computed by marginalizing $\mathbf{z}_i$:

$$\mathrm{P}\left(\mathbf{x}_{1:t}\right) = \sum_{\mathbf{z}_k}\alpha(\mathbf{z}_k)\beta(\mathbf{z}_k)$$

Recursion equations:

$$\alpha(\mathbf{z}_i) = \mathrm{P}\left(\mathbf{x}_{1:i}|\mathbf{z}_i\right)\mathrm{P}\left(\mathbf{z}_i\right) = \mathrm{P}\left(\mathbf{x}_i|\mathbf{z}_i\right)\sum_{\mathbf{z}_{i-1}}\alpha(\mathbf{z}_{i-1})\mathrm{P}\left(\mathbf{z}_i|\mathbf{z}_{i-1}\right)$$

$$\beta(\mathbf{z}_i) = \mathrm{P}\left(\mathbf{x}_{i+1:t}|\mathbf{z}_i\right) = \sum_{\mathbf{z}_{i+1}}\beta(\mathbf{z}_{i+1})\mathrm{P}\left(\mathbf{x}_{i+1}|\mathbf{z}_{i+1}\right)\mathrm{P}\left(\mathbf{z}_{i+1}|\mathbf{z}_i\right)$$

With initial conditions:

$$\alpha(\mathbf{z}_1) = \mathrm{P}\left(\mathbf{x}_1|\mathbf{z}_1\right)\mathrm{P}\left(\mathbf{z}_1\right) = \prod_k(\pi_k\mathrm{P}\left(\mathbf{x}_1|\phi_k\right))^{z_{1k}}$$

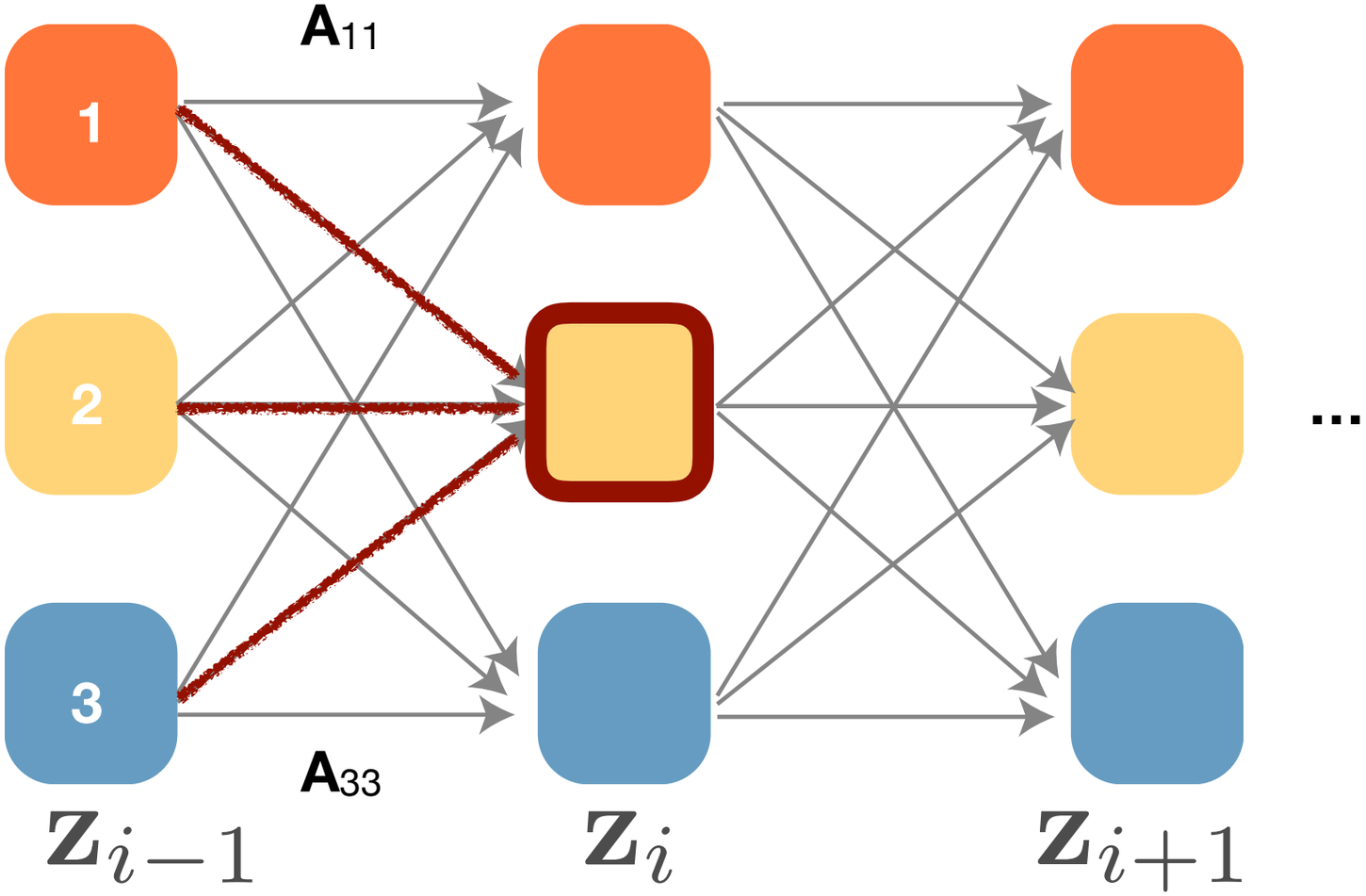
$$\beta(\mathbf{z}_t) = 1$$

These quantities are enough to also compute the joint:

$$\mathrm{P}\left(\mathbf{z}_i, \mathbf{z}_{i+1}|\mathbf{x}_{1:t}\right) = \frac{\alpha(\mathbf{z}_i)\mathrm{P}\left(\mathbf{z}_{i+1}|\mathbf{z}_i\right)\mathrm{P}\left(\mathbf{x}_{i+1}|\mathbf{z}_{i+1}\right)\beta(\mathbf{z}_{i+1})}{\mathrm{P}\left(\mathbf{x}_{1:t}\right)}$$

Proofs: on the board and in handout hmm.pdf

Interpretation: sum-product



$$\alpha(\mathbf{z}_i) \quad = \quad \mathrm{P}\left(\mathbf{x}_i|\mathbf{z}_i\right) \sum_{\mathbf{z}_{i-1}} \alpha(\mathbf{z}_{i-1}) \mathrm{P}\left(\mathbf{z}_i|\mathbf{z}_{i-1}\right)$$

consider **all possible ways** to reach state $\mathbf{z}_{i,k}$ and **sum** them up,
then combine with local evidence

Implementational caveat (a rather important one)

$$\alpha(\mathbf{z}_i) \quad = \quad \mathrm{P}\left(\mathbf{x}_{1:i}, \mathbf{z}_i\right)$$

joint distribution over increasingly many things,
the probability of any particular configuration is vanishingly small,
easy to go under machine precision

Solution: rescale messages to keep things in sensible range

$$\hat{\alpha}(\mathbf{z}_i) \quad = \quad \mathrm{P}\left(\mathbf{z}_i | \mathbf{x}_{1:i}\right) = \frac{\alpha(\mathbf{z}_i)}{\mathrm{P}\left(\mathbf{x}_{1:i}\right)}$$

$$\hat{\beta}(\mathbf{z}_i) \quad = \quad \frac{\mathrm{P}\left(\mathbf{x}_{i+1:t} | \mathbf{z}_i\right)}{\mathrm{P}\left(\mathbf{x}_{\mathbf{i+1:t}} | \mathbf{x}_{1:i}\right)} = \frac{\beta(\mathbf{z}_i)}{\prod_{j=i+1:t} c_j}$$

where we've introduced an intermediate variable

$$c_i \quad = \quad \mathrm{P}\left(\mathbf{x}_i | \mathbf{x}_{1:i-1}\right)$$

$$\mathrm{P}\left(\mathbf{x}_{1:i}\right) \quad = \quad \prod_{j=1:i} c_j$$

The updated recursion equation for $\hat{\alpha}$, and $\hat{\beta}$ become:

$$
\begin{aligned}
c_i\hat{\alpha}(\mathbf{z}_i) &= \mathrm{P}\left(\mathbf{x}_i|\mathbf{z}_i\right)\sum_{\mathbf{z}_{i-1}}\hat{\alpha}(\mathbf{z}_{i-1})\mathrm{P}\left(\mathbf{z}_i|\mathbf{z}_{i-1}\right) \\
c_{i+1}\hat{\beta}(\mathbf{z}_i) &= \sum_{\mathbf{z}_{i+1}}\hat{\beta}(\mathbf{z}_{i+1})\mathrm{P}\left(\mathbf{x}_{i+1}|\mathbf{z}_{i+1}\right)\mathrm{P}\left(\mathbf{z}_{i+1}|\mathbf{z}_i\right)
\end{aligned}
$$

check at home!

Finally the new expressions for the posterior marginals are:

$$
\begin{aligned}
\gamma(\mathbf{z}_i) &= \hat{\alpha}(\mathbf{z}_i)\hat{\beta}(\mathbf{z}_i) \\
\xi(z_{i,j}, z_{i+1,k}) &= {c_{i+1}}^{-1}\hat{\alpha}(\mathbf{z}_i)\mathrm{P}\left(\mathbf{z}_{i+1}|\mathbf{z}_i\right)\mathrm{P}\left(\mathbf{x}_{i+1}|\mathbf{z}_{i+1}\right)\hat{\beta}(\mathbf{z}_{i+1})
\end{aligned}
$$

Formal link to LDS:

Kalman filtering: $\hat{\alpha}(\mathbf{z}_i)$

Kalman smoothing: $\gamma(\mathbf{z}_i) = \hat{\alpha}(\mathbf{z}_i)\hat{\beta}(\mathbf{z}_i)$

1. Inference: MAP

$$\hat{\mathbf{z}}_{1:t} = \text{argmax}_{\mathbf{z}_{1:t}} \{\text{P}\left(\mathbf{z}_{1:t} | \mathbf{x}_{1:t}\right)\}$$

The Viterbi algorithm

(this is also an instance of dynamic programming)

General idea: somebody told you the probability of the most likely configuration is up to point i, and its end point,
can you use that to figure out the most likely configuration up to $\mathbf{z}_{i+1}$

Yet another instance of forward message passing:

why not conditional?
why log?

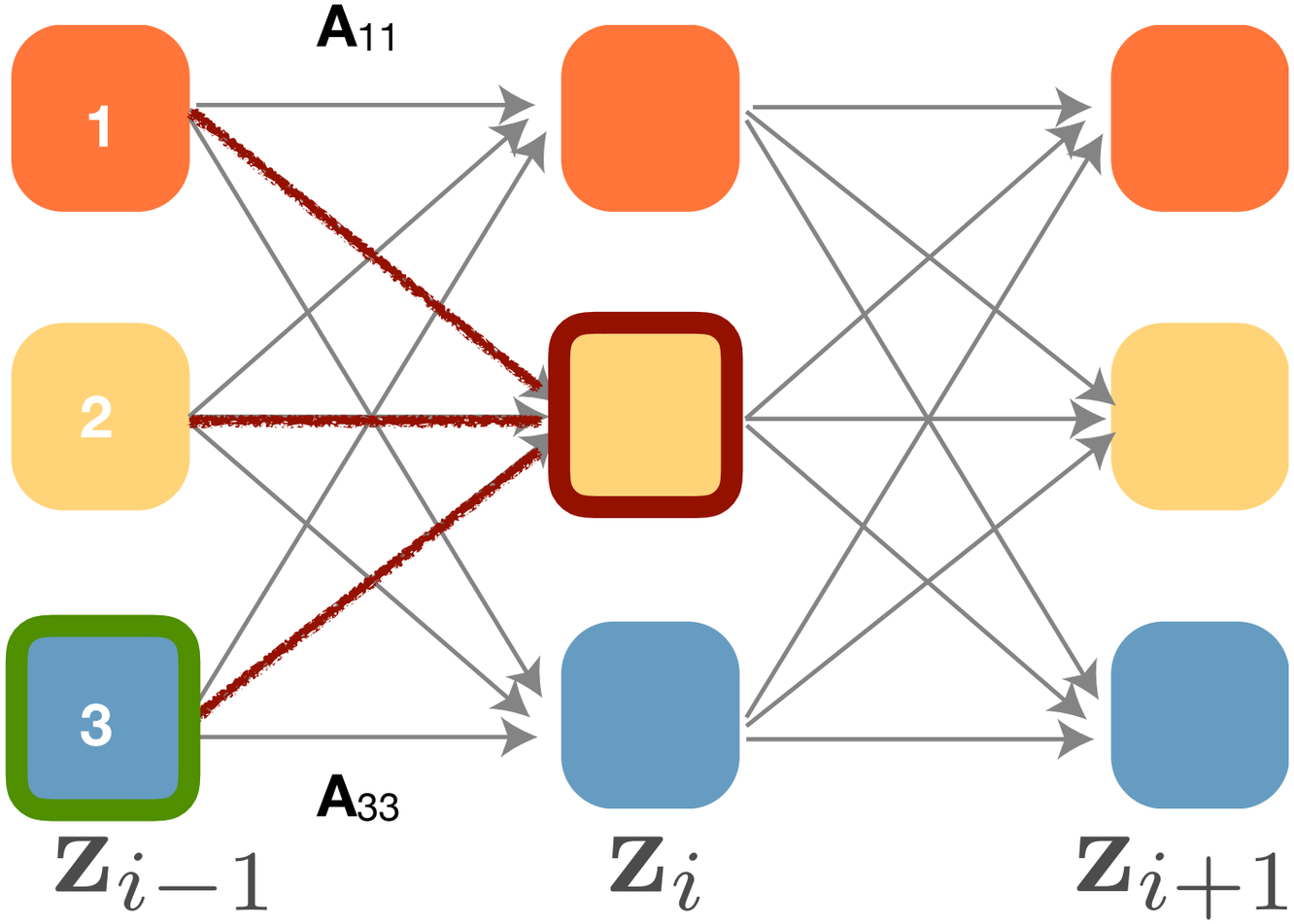
With recursion:

$$w_{i+1} = \log \text{P}\left(\mathbf{x}_{i+1} | \mathbf{z}_{i+1}\right) + \max_{\mathbf{z}_i} \{\log \text{P}\left(\mathbf{z}_{i+1} | \mathbf{z}_i\right) + w_i(\mathbf{z}_i)\}$$

And initial condition: $w_1 = \log \text{P}\left(\mathbf{z}_1\right) + \log \text{P}\left(\mathbf{x}_1 | \mathbf{z}_1\right)$

nterpretation: max-product

$$k_i^{\max} = \text{argmax}_{\mathbf{z}_i} \{\log \mathrm{P}\left(\mathbf{z}_{i+1}|\mathbf{z}_i\right) + w_i(\mathbf{z}_i)\}$$



...

+need to remember
how we got there

$$w_{i+1} = \log \mathrm{P}\left(\mathbf{x}_{i+1}|\mathbf{z}_{i+1}\right) + \max_{\mathbf{z}_i} \{\log \mathrm{P}\left(\mathbf{z}_{i+1}|\mathbf{z}_i\right) + w_i(\mathbf{z}_i)\}$$

consider **all possible ways** to reach state $\mathbf{z}_{i,k}$ and **pick the best**,
then combine with local evidence; also remember which was the best

2. Parameter learning: EM

Goal: find parameters that maximize $\mathrm{P}\left(\mathbf{x}|\theta\right) = \int_Z \mathrm{P}\left(\mathbf{x}, \mathbf{z}|\theta\right) \mathbf{dz}$

Remember the **general recipe**:

E-step: $q(\mathbf{z}) = \mathrm{P}\left(\mathbf{z}|\mathbf{x}, \theta^{\mathrm{old}}\right)$ fixed parameters θ^{old}

This we do using alpha-beta

M-step: $\mathcal{Q}(\theta, \theta^{\mathrm{old}}) = \int_Z \mathrm{P}\left(\mathbf{z}|\mathbf{x}, \theta^{\mathrm{old}}\right) \log \mathrm{P}\left(\mathbf{x}, \mathbf{z}|\theta\right)$

fixed q, optimize parameters

Separate terms for observation model, latent state transitions and initial state, optimize one at a time

Learning: EM

M-step:

In general:

$$\pi_k^{\text{new}} = \frac{\gamma(z_{1,k})}{\sum_j \gamma(z_{1,j})}$$

$$A_{jk}^{\text{new}} = \frac{\sum_i \xi(z_{i,j}, z_{i+1,k})}{\sum_{i,l} \xi(z_{i,j}, z_{i+1,l})}$$

For a simple *gaussian* observation model

$$\mu_k^{\text{new}} = \frac{1}{\sum_i \gamma(z_{i,k})} \sum_i \gamma(z_{i,k}) \mathbf{x}_i$$

$$\Sigma_k^{\text{new}} = \frac{1}{\sum_i \gamma(z_{i,k})} \sum_i \gamma(z_{i,k}) \mathbf{x}_i \mathbf{x}_i^{\text{t}} - \mu_k \mu_k^{\text{t}}$$

This is just a weighted version of empirical mean and covariance

What happens next:

HMMs lab: real data! (supervised setting, so model fit easier, no EM)
but you get to implement Viterbi inference

next week's lecture: putting what we've learned in a bigger picture,
variations, what can and can't we do with these models,
where do we go when they are not enough

I'll upload the ARIMA handout for next week's lab by end of the week