HMMs + LDSs

Some Final Thoughts on FA and PCA

- PCA has simpler understanding in terms of variance and orthogonality
- FA won’t be as dependent on scaling
- Important to check that scientific results are robust across methods

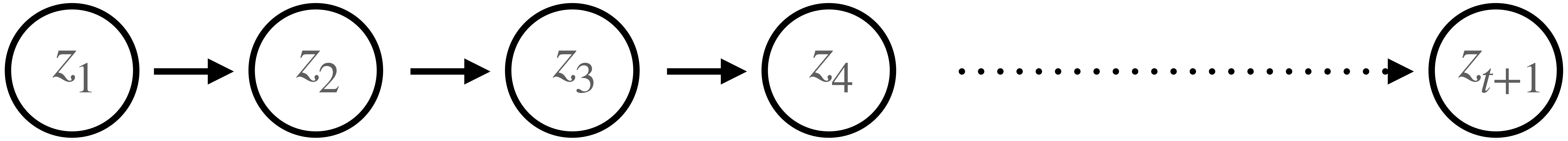
Markov Chains

$$z_1 \rightarrow z_2 \rightarrow z_3 \rightarrow z_4 \quad \cdots\cdots\cdots \rightarrow z_{t+1}$$

- The current state only depends on the past state

$$P(z_{t+1} \mid z_1, z_2, \ldots, z_t) = P(z_{t+1} \mid z_t)$$

Markov Chains



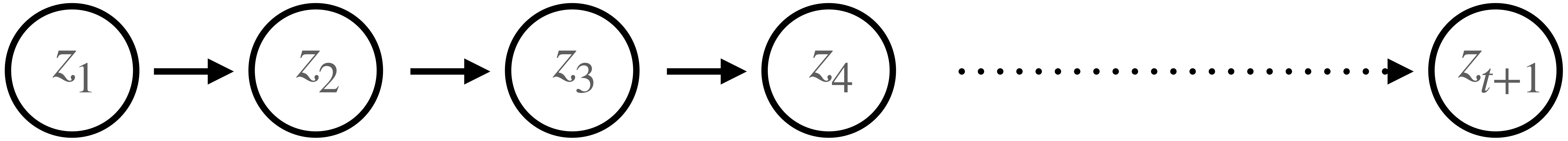
- The current state only depends on the past state

$$P(z_{t+1} \mid z_1, z_2, \ldots, z_t) = P(z_{t+1} \mid z_t)$$

- We can use the rules of independence to calculate the total probability:

$$P(z_1, z_2, \ldots, z_{t+1}) = P(z_1)P(z_2 \mid z_1) \ldots P(z_t \mid z_{t-1})P(z_{t+1} \mid z_t)$$

Markov Chains



- The current state only depends on the past state

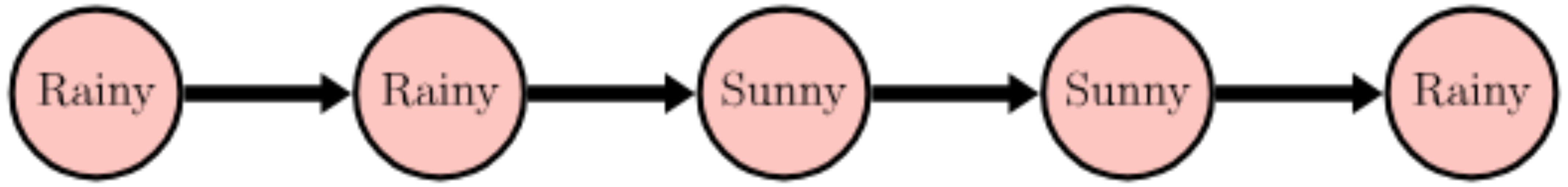
$$P(z_{t+1} \,|\, z_1, z_2, \ldots, z_t) = P(z_{t+1} \,|\, z_t)$$

- We can use the rules of independence to calculate the total probability:

$$P(z_1, z_2, \ldots, z_{t+1}) = P(z_1)P(z_2 \,|\, z_1) \ldots P(z_t \,|\, z_{t-1})P(z_{t+1} \,|\, z_t)$$

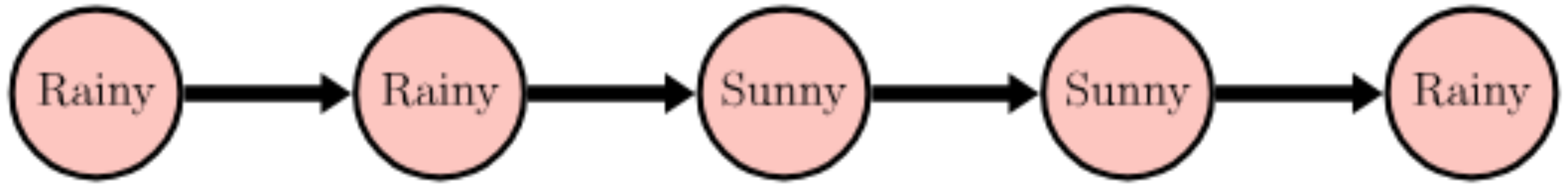
$$P(z_{1:T}) = P(z_1)\prod_{t=2}^{T} p(z_t \,|\, z_{t-1})$$

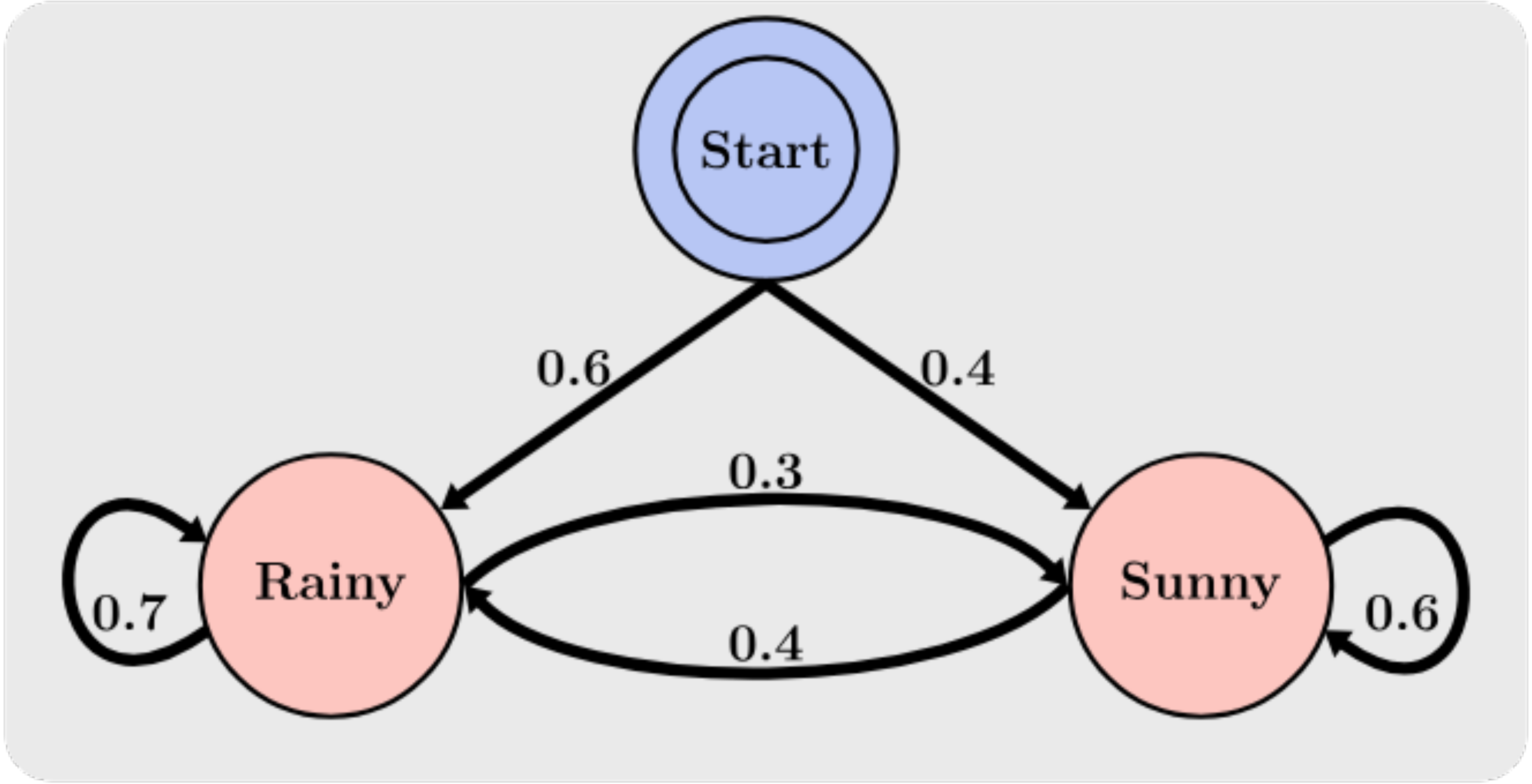
Markov Chains



https://nipunbatra.github.io/hmm/

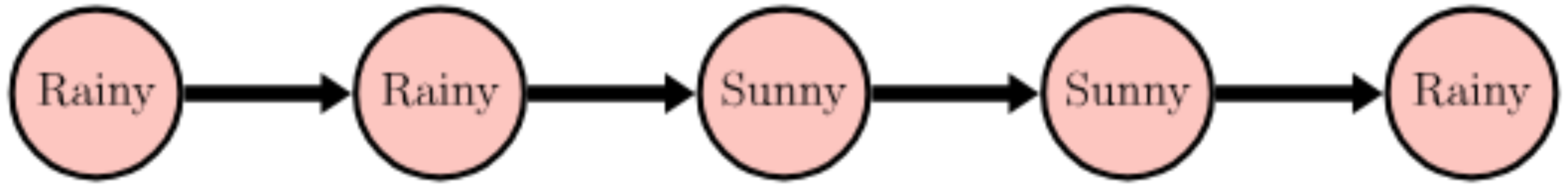
Markov Chains



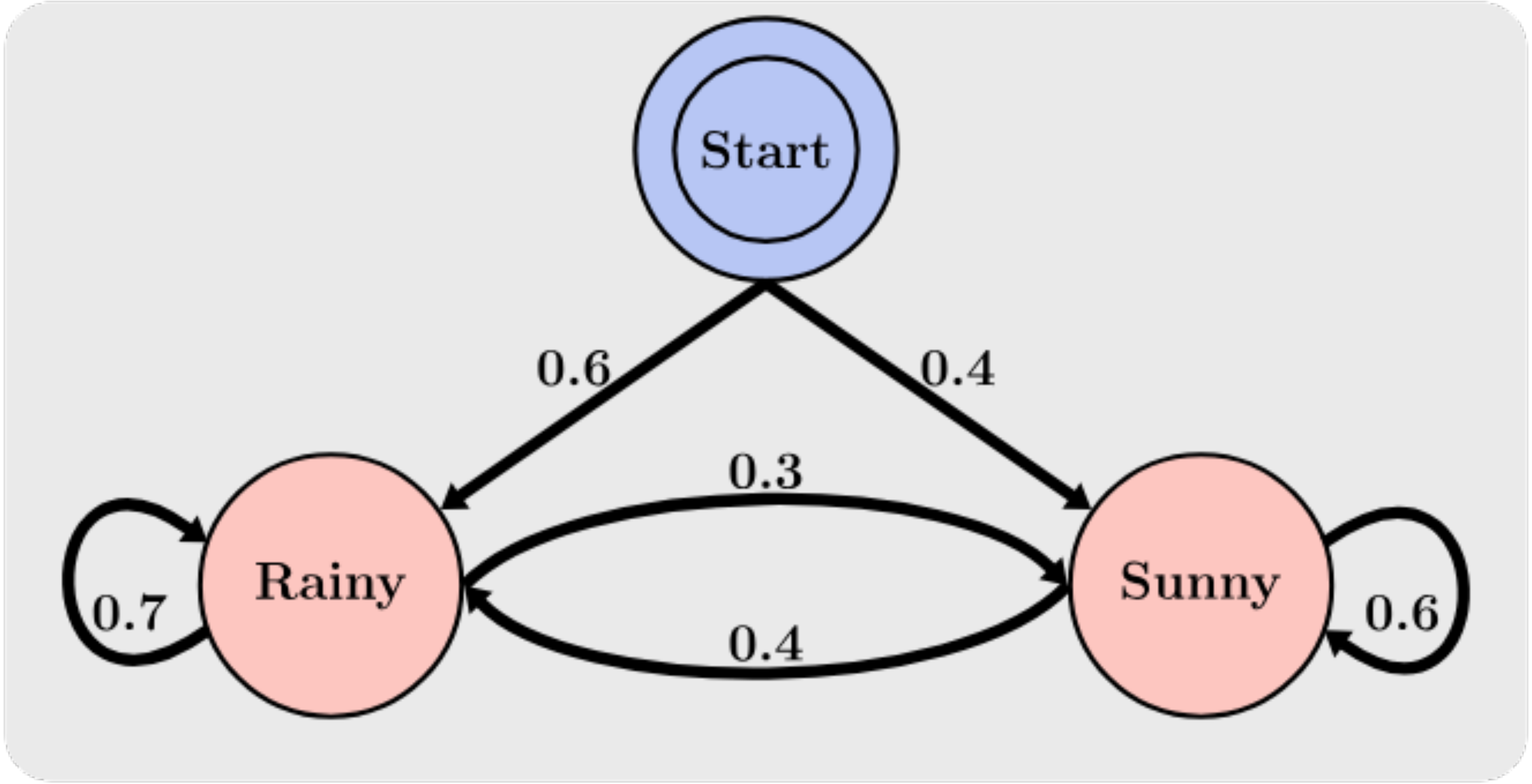


https://nipunbatra.github.io/hmm/

Markov Chains

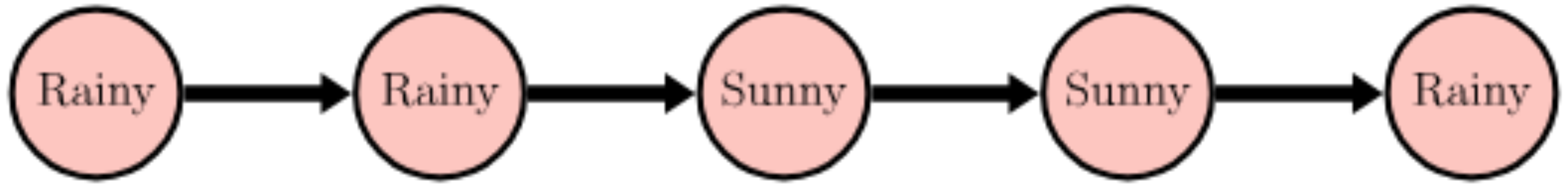


$$P(z_1, z_2, \ldots, z_{t+1}) = P(z_1)P(z_2 \mid z_1) \ldots P(z_t \mid z_{t-1})P(z_{t+1} \mid z_t)$$

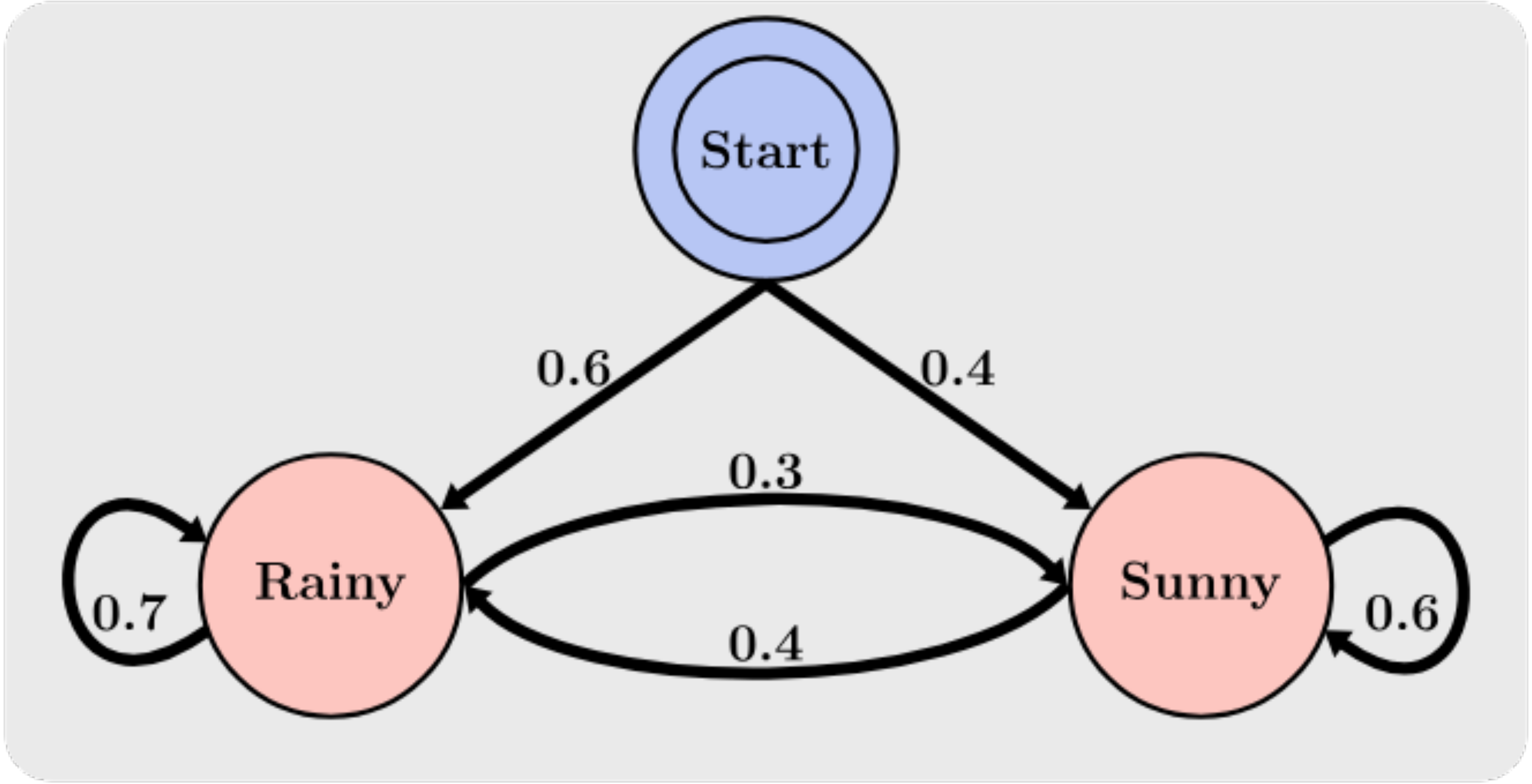


https://nipunbatra.github.io/hmm/

Markov Chains



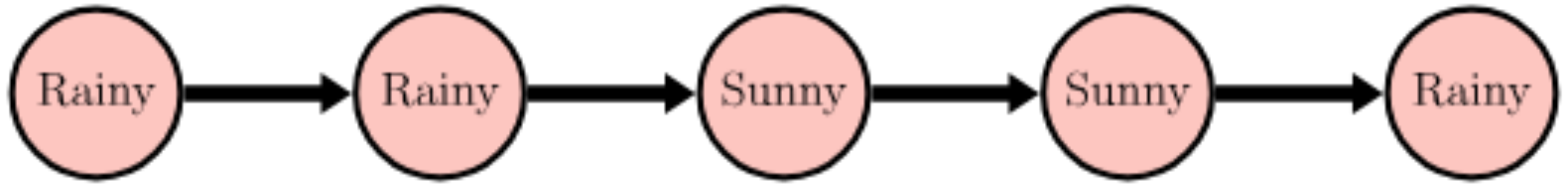
$$P(z_1, z_2, \ldots, z_{t+1}) = P(z_1)P(z_2 \mid z_1) \ldots P(z_t \mid z_{t-1})P(z_{t+1} \mid z_t)$$



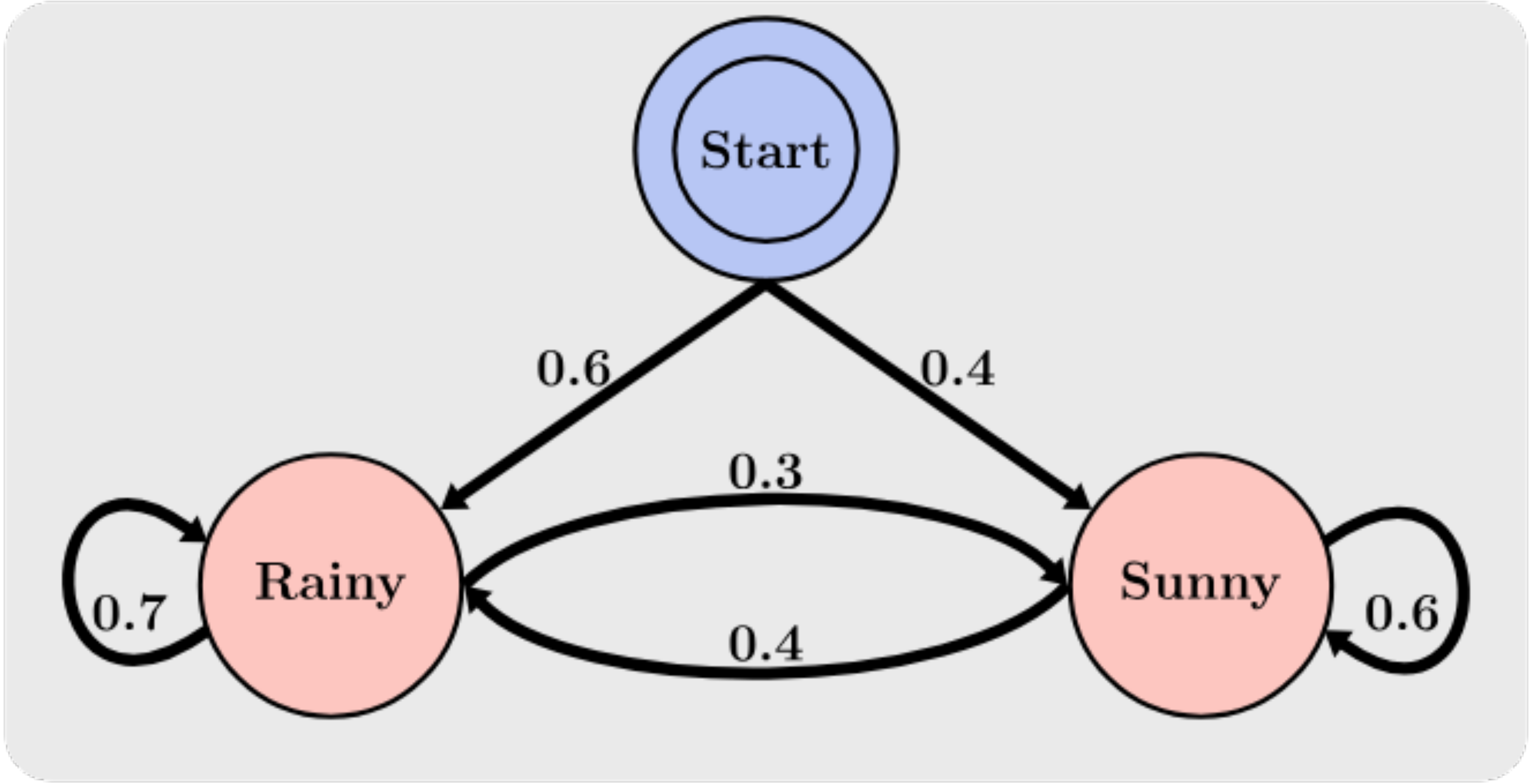
Initial Conditions

$$P(z_1 = R) = 0.6, \quad P(z_1 = S) = 0.4$$

Markov Chains



$$P(z_1, z_2, \ldots, z_{t+1}) = P(z_1)P(z_2 \mid z_1) \ldots P(z_t \mid z_{t-1})P(z_{t+1} \mid z_t)$$



Initial Conditions

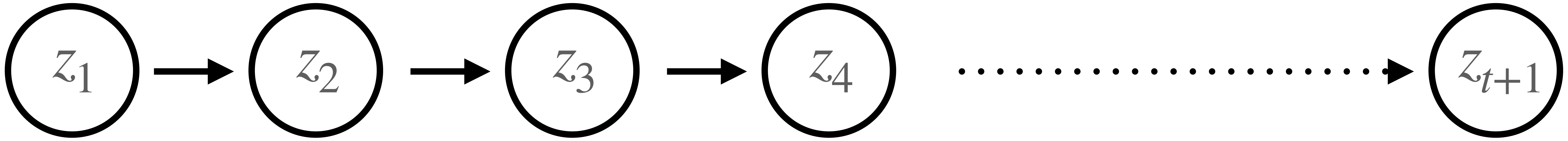
$$P(z_1 = R) = 0.6, \quad P(z_1 = S) = 0.4$$

Transitions Matrix

$P(z_{t+1} = R \mid z_t = R) = 0.7$	$P(z_{t+1} = S \mid z_t = R) = 0.3$
$P(z_{t+1} = S \mid z_t = R) = 0.4$	$P(z_{t+1} = S \mid z_t = S) = 0.6$

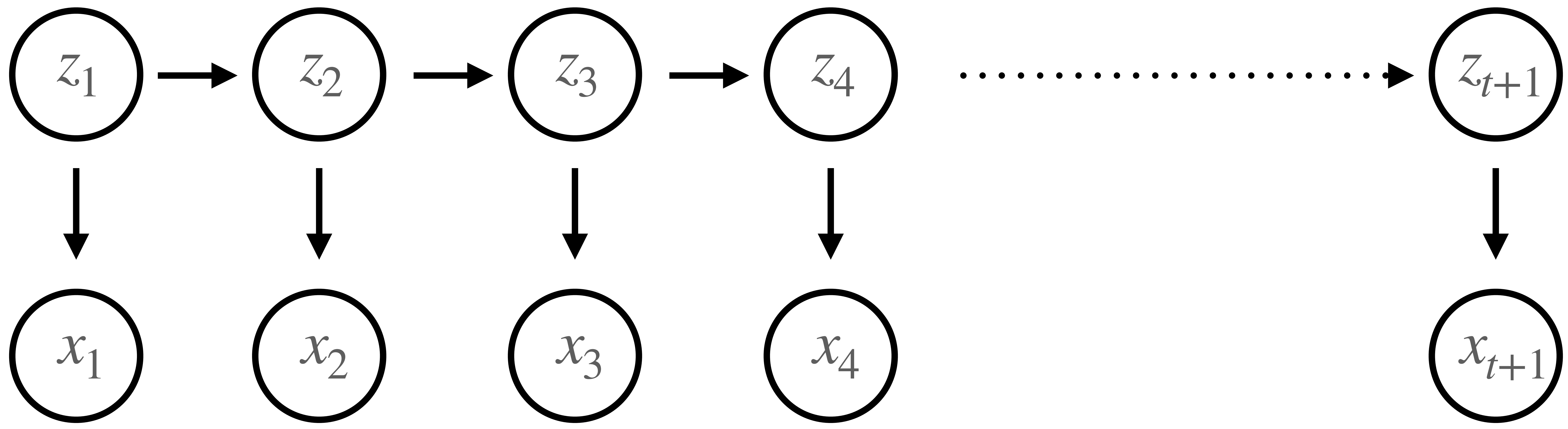
https://nipunbatra.github.io/hmm/

Hidden Markov Models



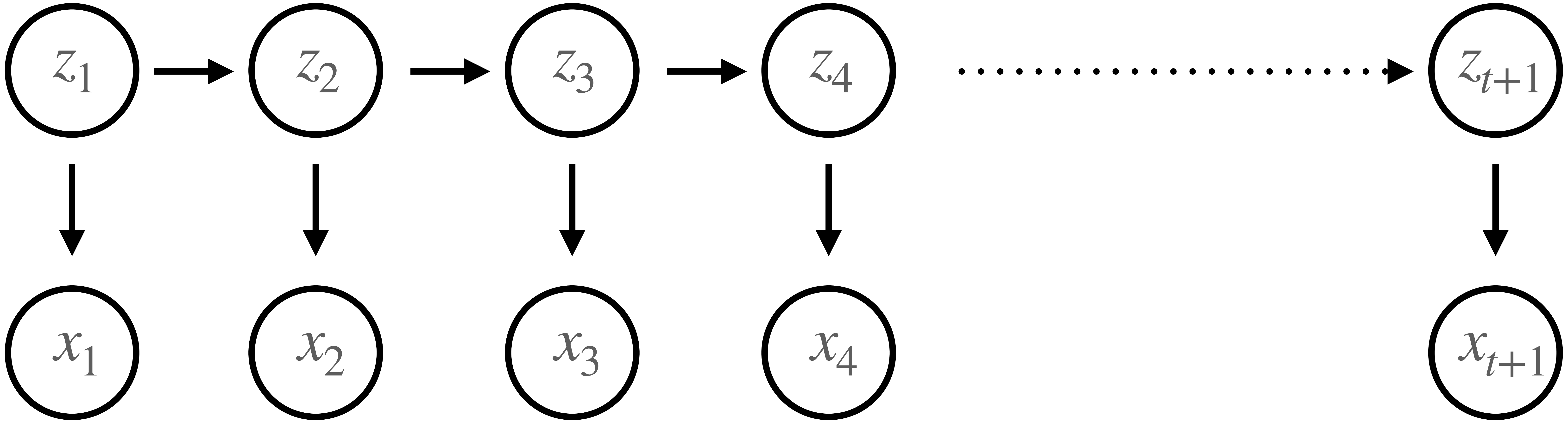
$$P(z_{1:T}) = P(z_1)\prod_{t=2}^{T} p(z_t \mid z_{t-1})$$

Hidden Markov Models



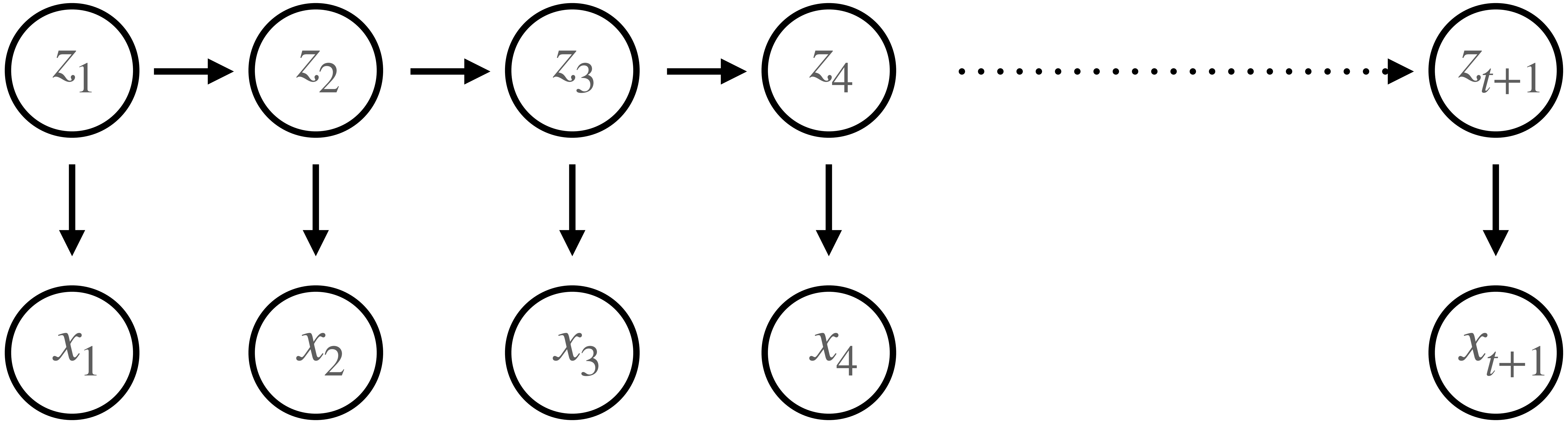
$$P(z_{1:T}) = P(z_1)\prod_{t=2}^{T} p(z_t \mid z_{t-1})$$

Hidden Markov Models



$$P(z_{1:T}, x_{1:T}) = P(z_1)\prod_{t=2}^{T} p(z_t \mid z_{t-1}) \prod_{t=1}^{T} p(x_t \mid z_t)$$

Hidden Markov Models



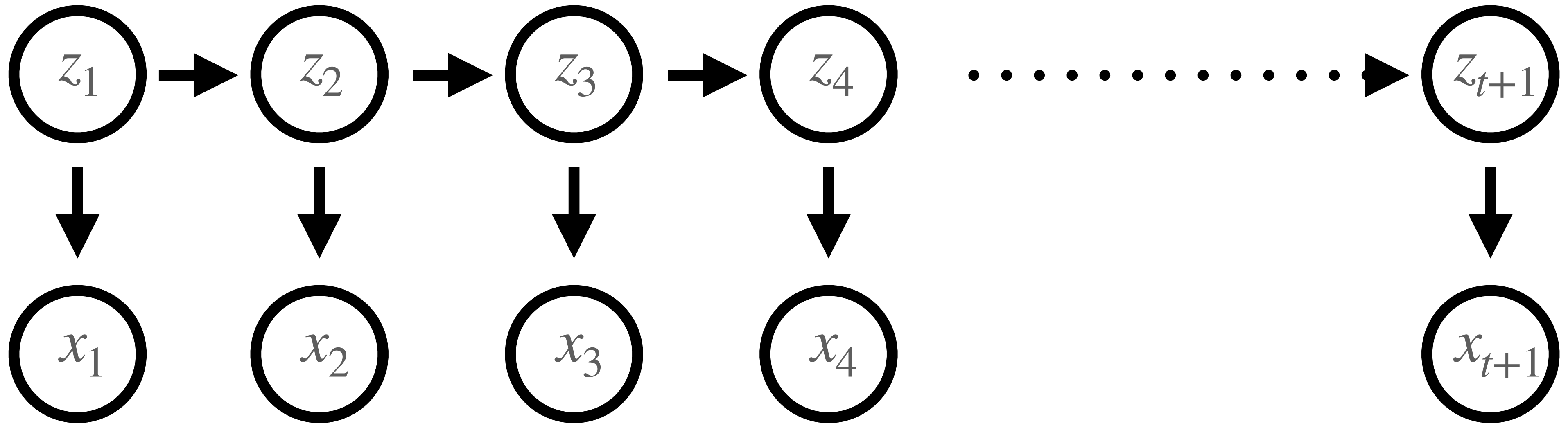
Initial Probabilities

Transition Probabilities

Emissions (Observation) Probabilities

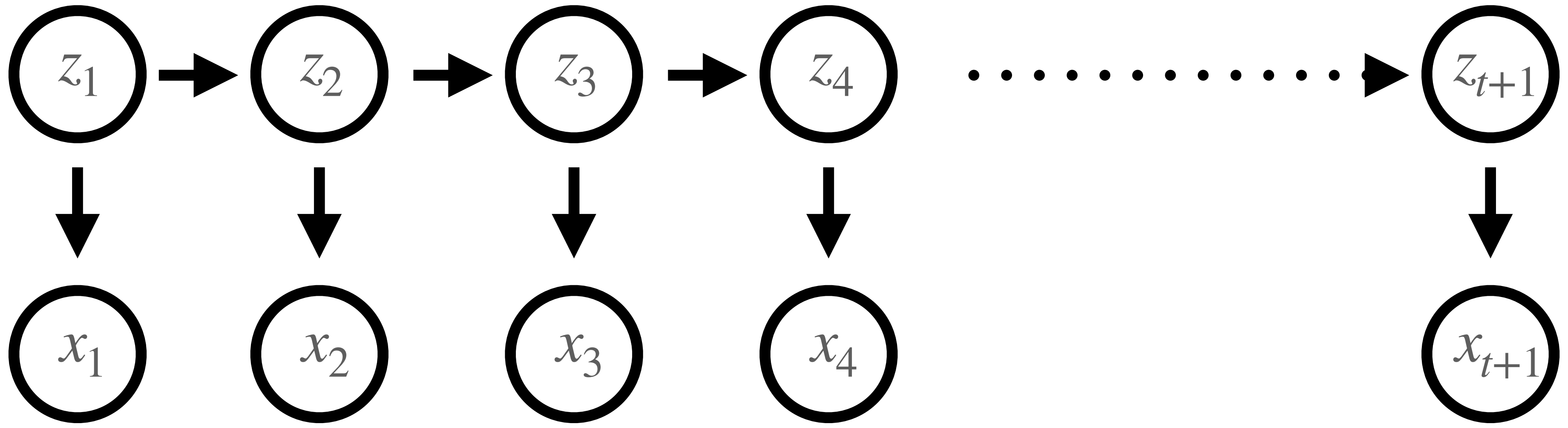
$$P(z_{1:T}, x_{1:T}) = P(z_1)\prod_{t=2}^{T} p(z_t \mid z_{t-1}) \prod_{t=1}^{T} p(x_t \mid z_t)$$

Hidden Markov Models: Emissions Models



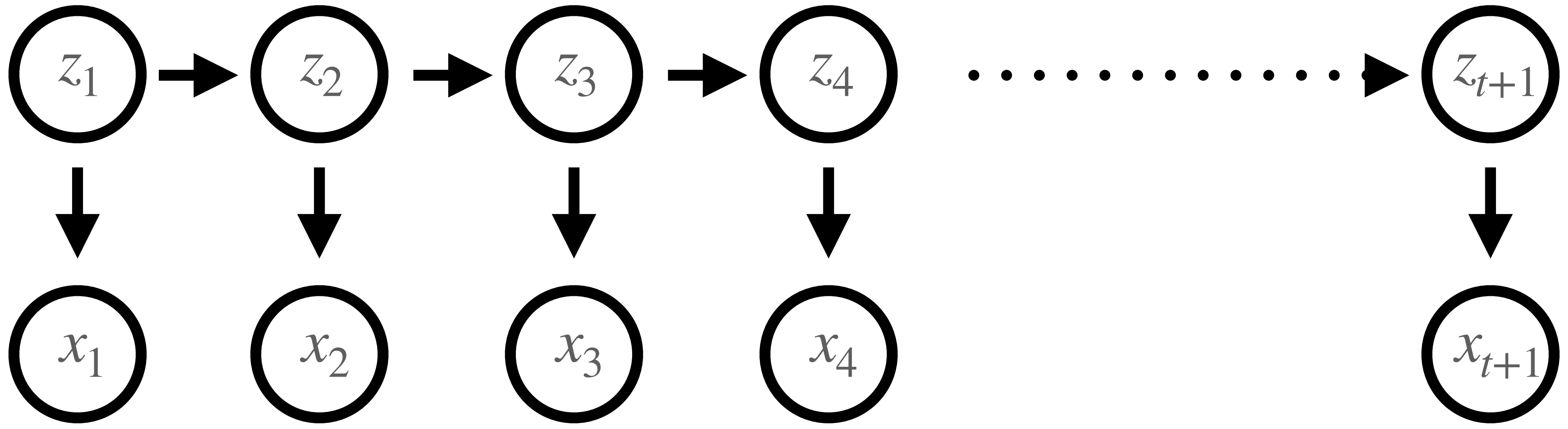
- $P(x \mid z)$ can take many different forms
    - Gaussian: $P(x_t \mid z_t) = \mathcal{N}(\mu_{z_t}, \sigma_{z_t})$

Hidden Markov Models: Emissions Models



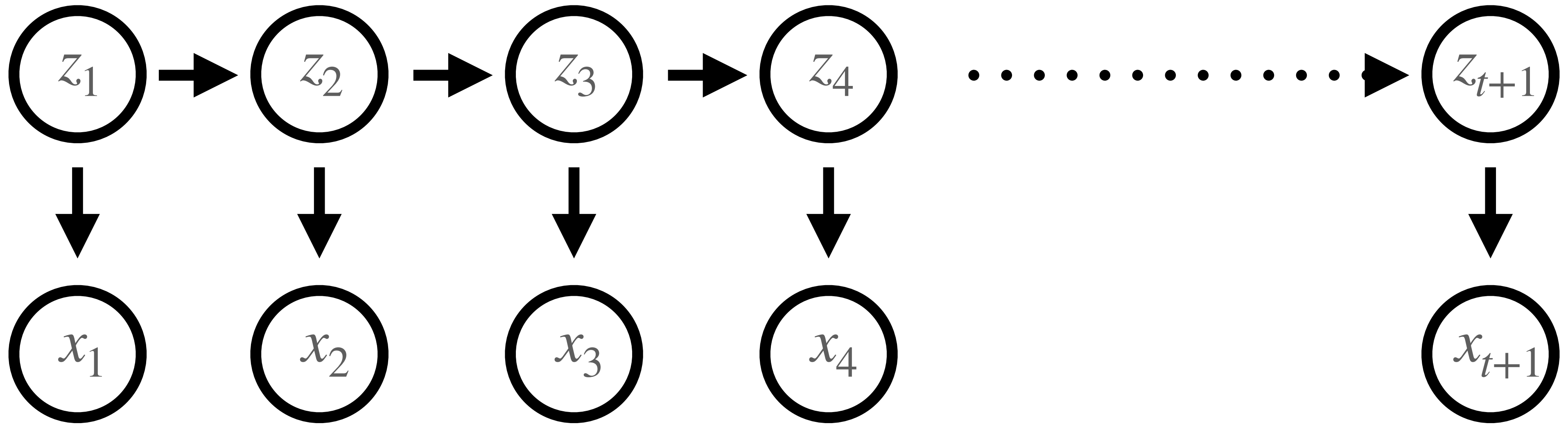
- $P(x \mid z)$ can take many different forms
  - Gaussian: $P(x_t \mid z_t) = \mathcal{N}(\mu_{z_t}, \sigma_{z_t})$
  - Bernoulli, Poisson, etc.

Hidden Markov Models: Emissions Models



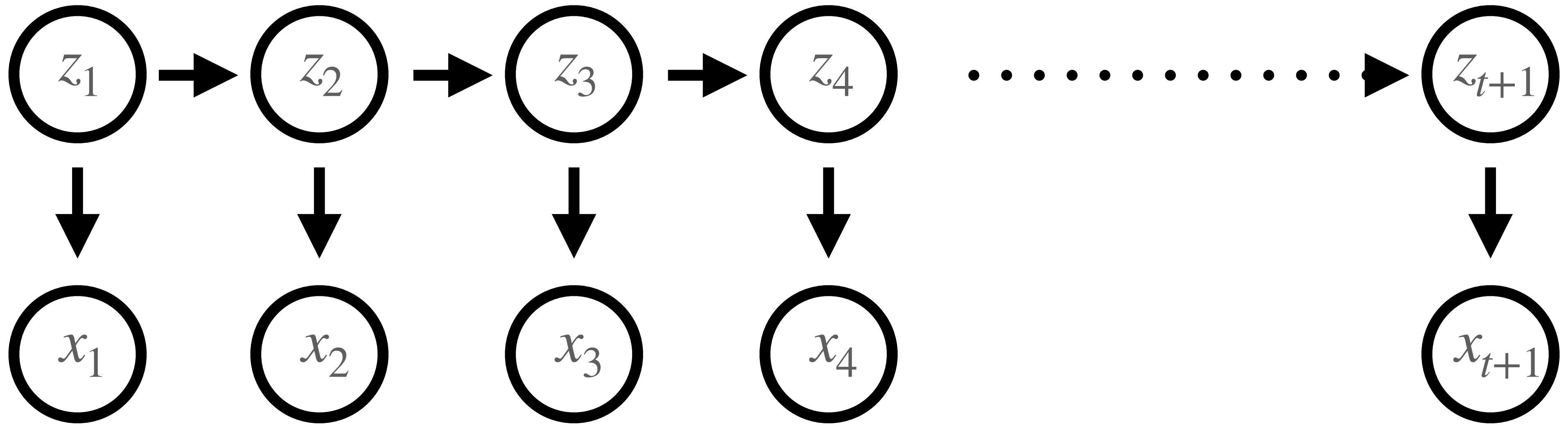
- $P(x \mid z)$ can take many different forms
  - Gaussian: $P(x_t \mid z_t) = \mathcal{N}(\mu_{z_t}, \sigma_{z_t})$
  - Bernoulli, Poisson, etc.
  - Autoregressive HMM (ARHMM):
    - Different dynamics in each discrete state: $P(x_t \mid z_t) = \mathcal{N}(y_{t-1} - A_{z_t} y_t, \sigma_{z_t})$

Hidden Markov Models: Emissions Models

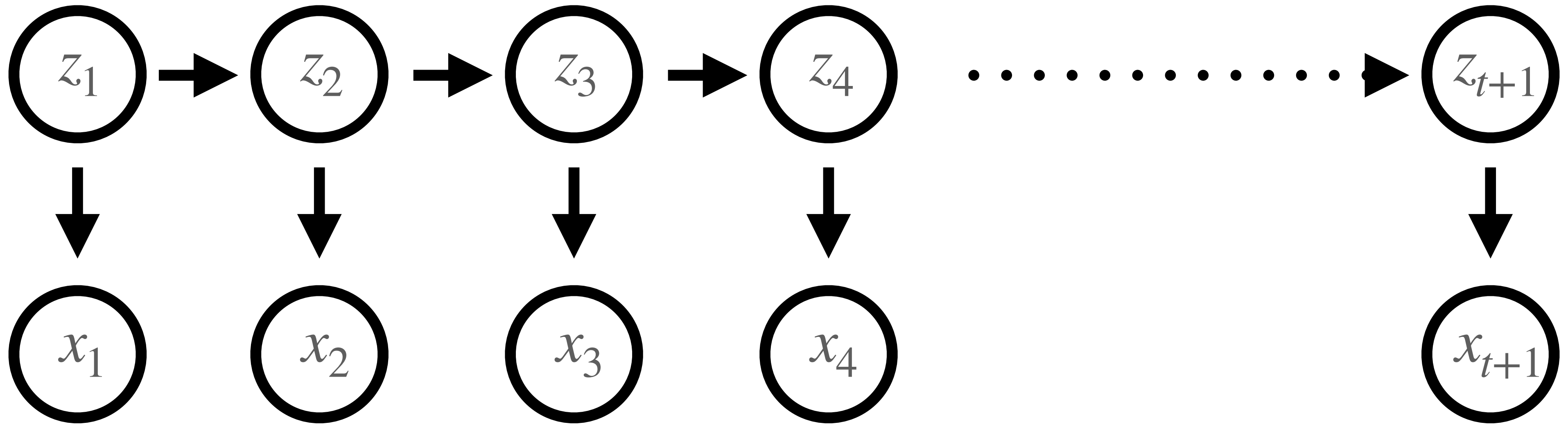


- $P(x \mid z)$ can take many different forms
  - Gaussian: $P(x_t \mid z_t) = \mathcal{N}(\mu_{z_t}, \sigma_{z_t})$
  - Bernoulli, Poisson, etc.
  - Autoregressive HMM (ARHMM):
    - Different dynamics in each discrete state: $P(x_t \mid z_t) = \mathcal{N}(y_{t-1} - A_{z_t} y_t, \sigma_{z_t})$
  - GLM-HMM:
    - different GLM weights in each discrete state

Simulating from an HMM

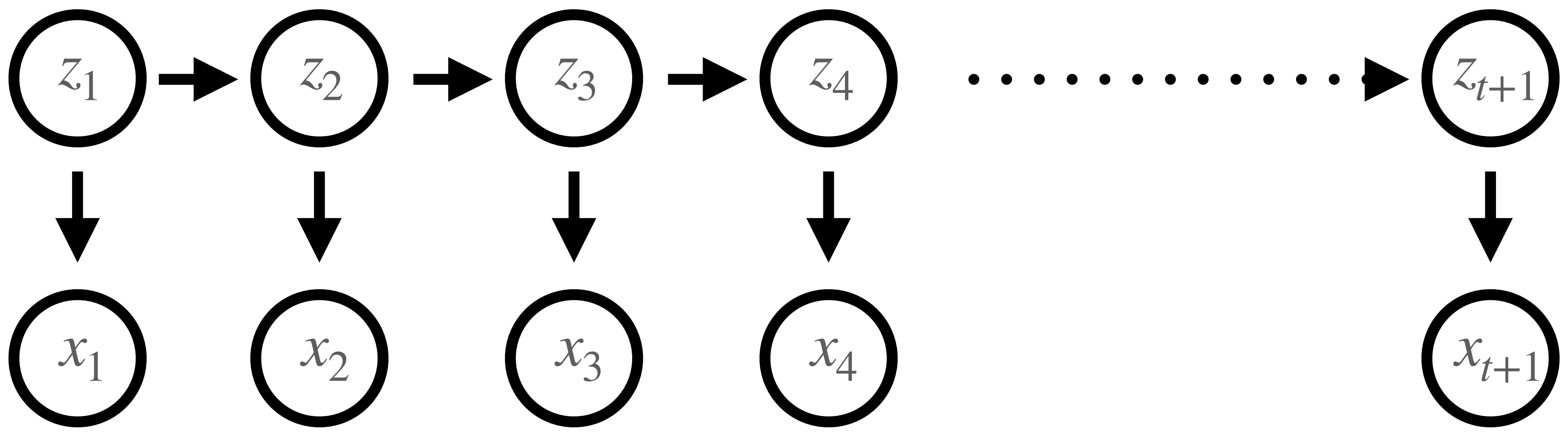


Simulating from an HMM



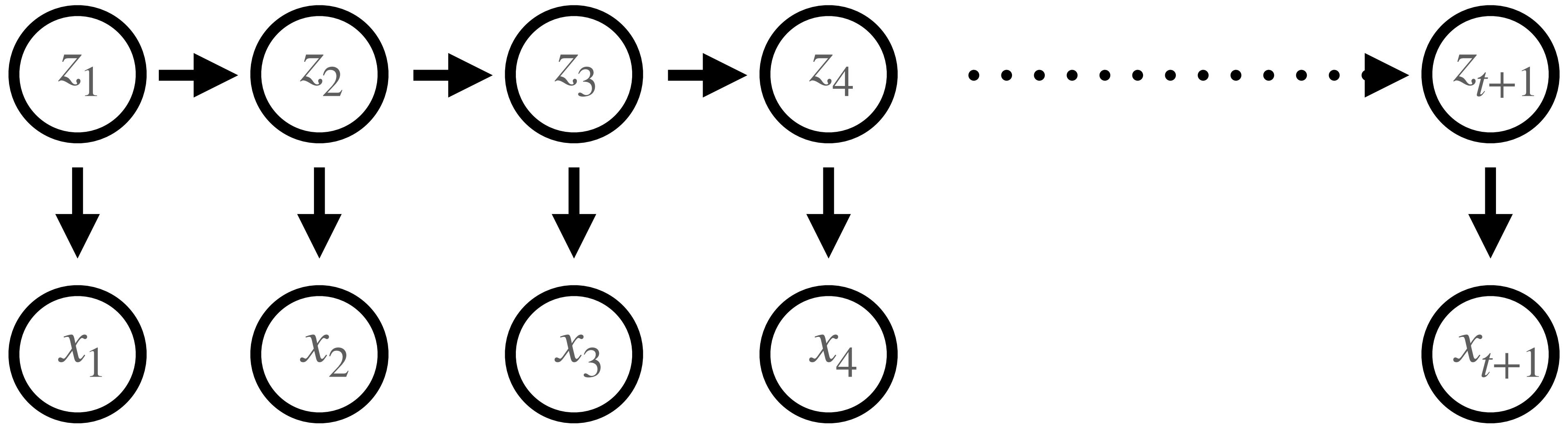
- Sample from $P(z_1)$

Simulating from an HMM



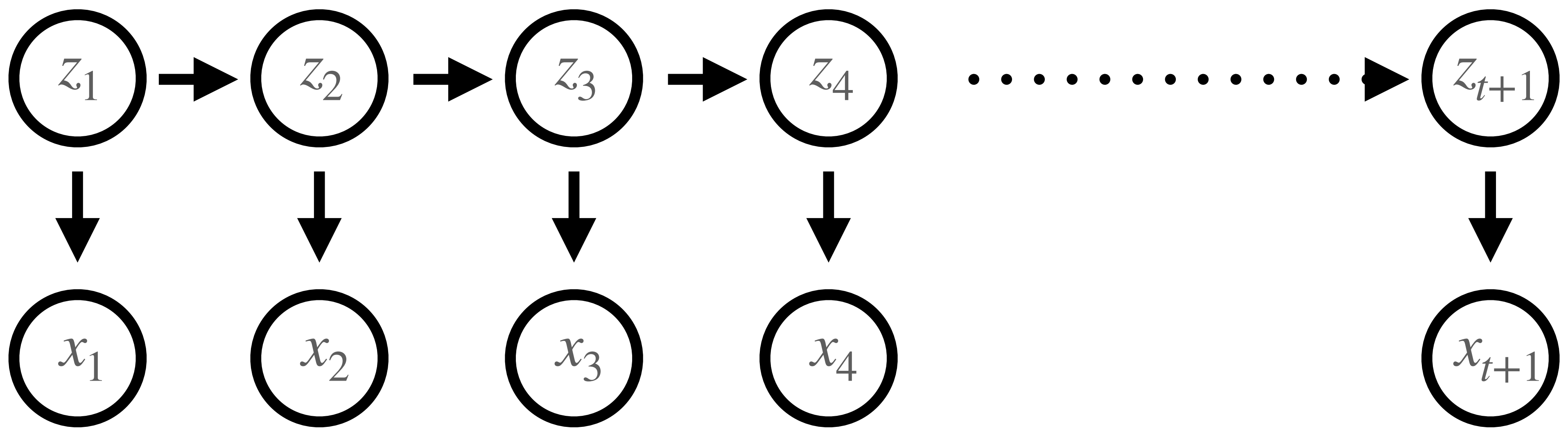
- Sample from $P(z_1)$
- Sample from $P(x_1 \mid z_1)$

Simulating from an HMM



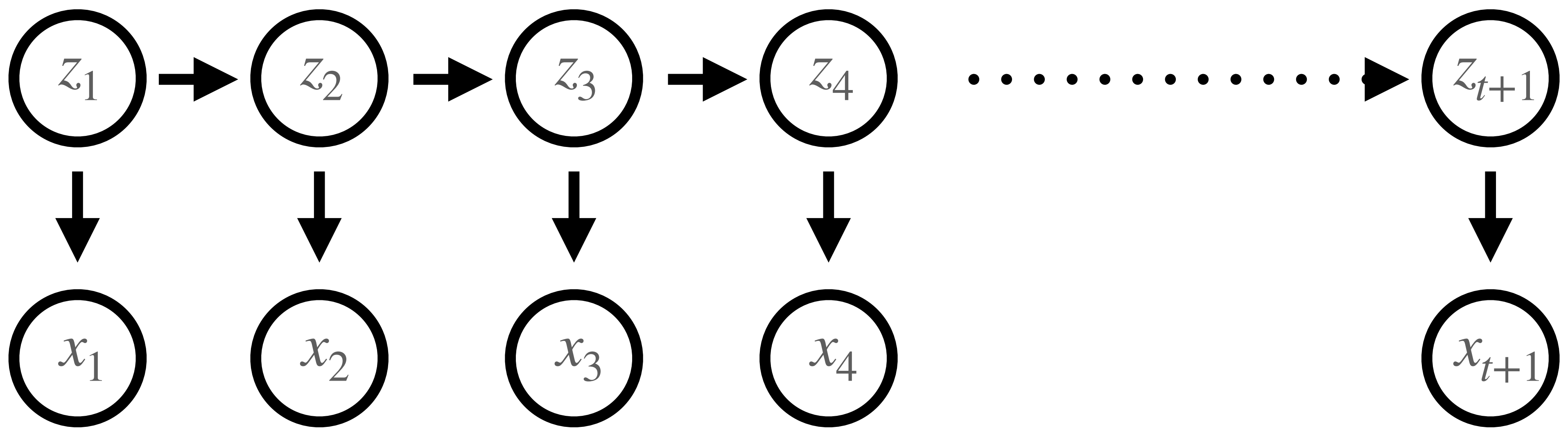
- Sample from $P(z_1)$
- Sample from $P(x_1 \mid z_1)$
- For all future time steps:
    - Sample $P(z_{t+1} \mid z_t)$
    - Sample $P(x_{t+1} \mid z_{t+1})$

HMM: Goals



- Given some data:
  - Fit the model!
    - Infer discrete latent states
    - Infer model parameters (transition probabilities, emissions model)
  - Determine the likelihood of the data given the model parameters

HMM: Goals

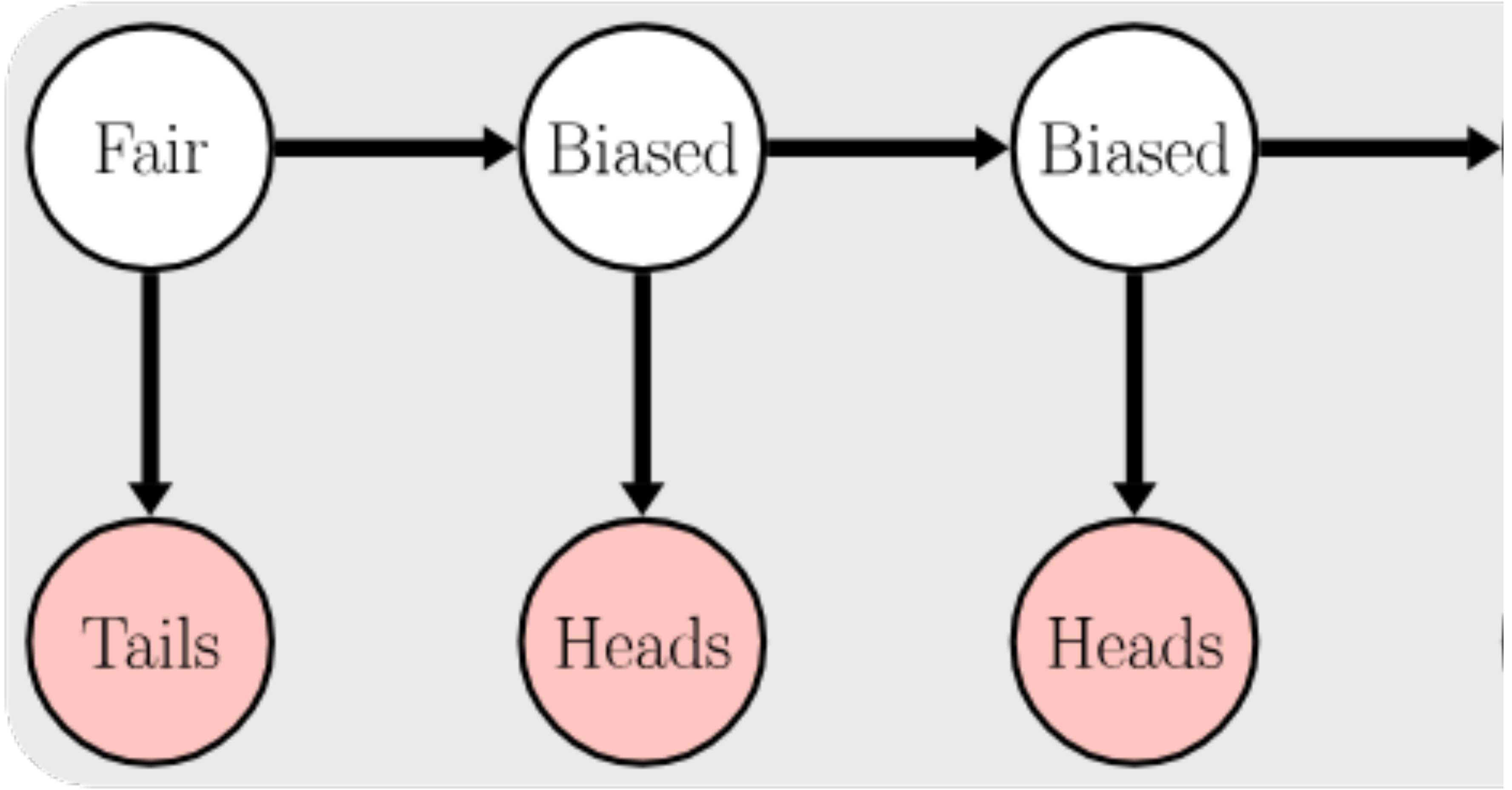


- Given some data:
  - Fit the model!
    - Infer discrete states
    - Infer model parameters (transition probabilities, emissions model)
  - **Determine the likelihood of the data given the model parameters**

HMMs: Evidence Likelihood

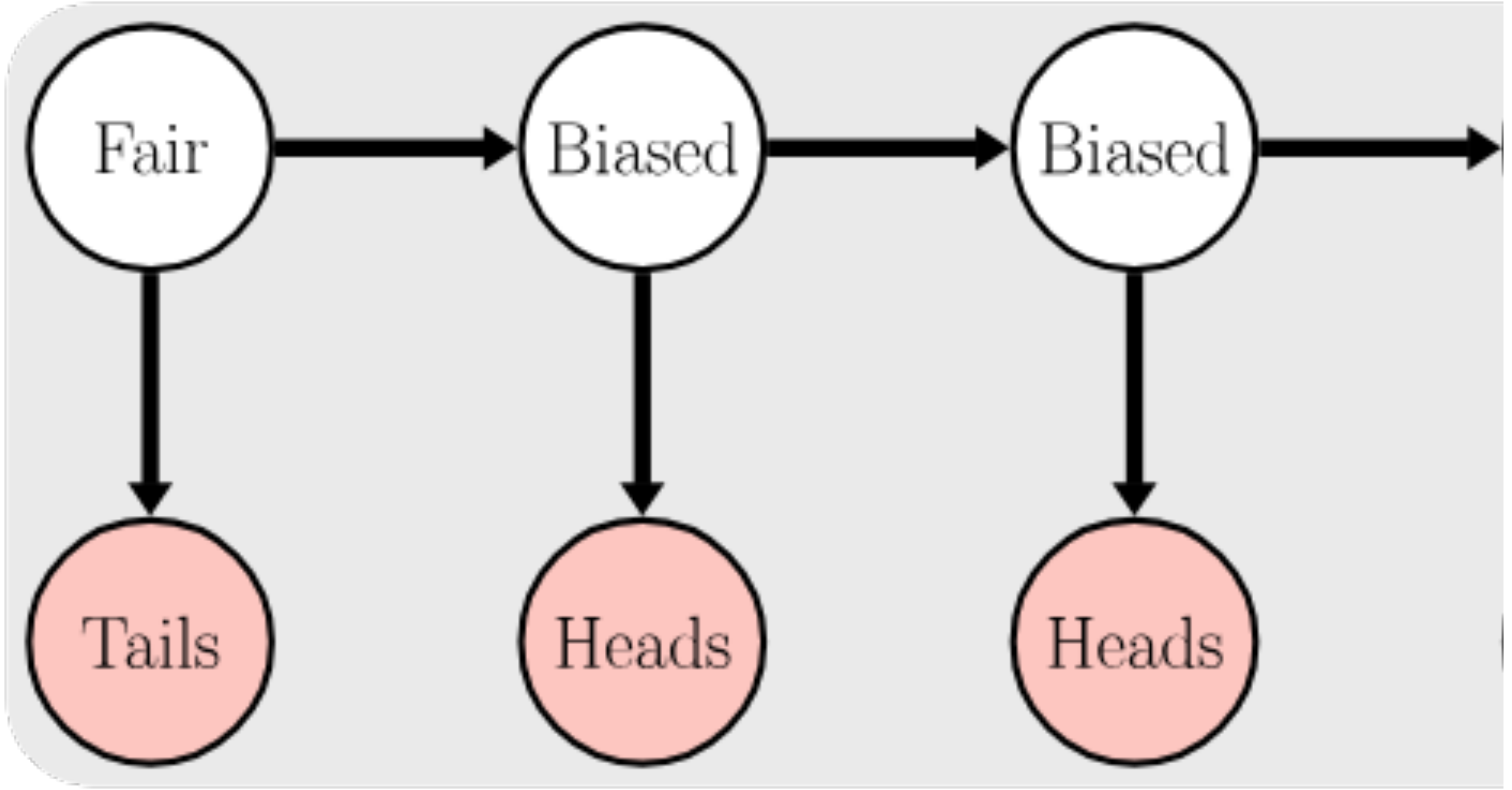
- Estimate the likelihood, $L(X|\theta)$, of the observed data, $X = \{x_1, x_2 \ldots . x_T\}$ given the HMM parameters θ

HMMs: Evidence Likelihood



https://nipunbatra.github.io/hmm/

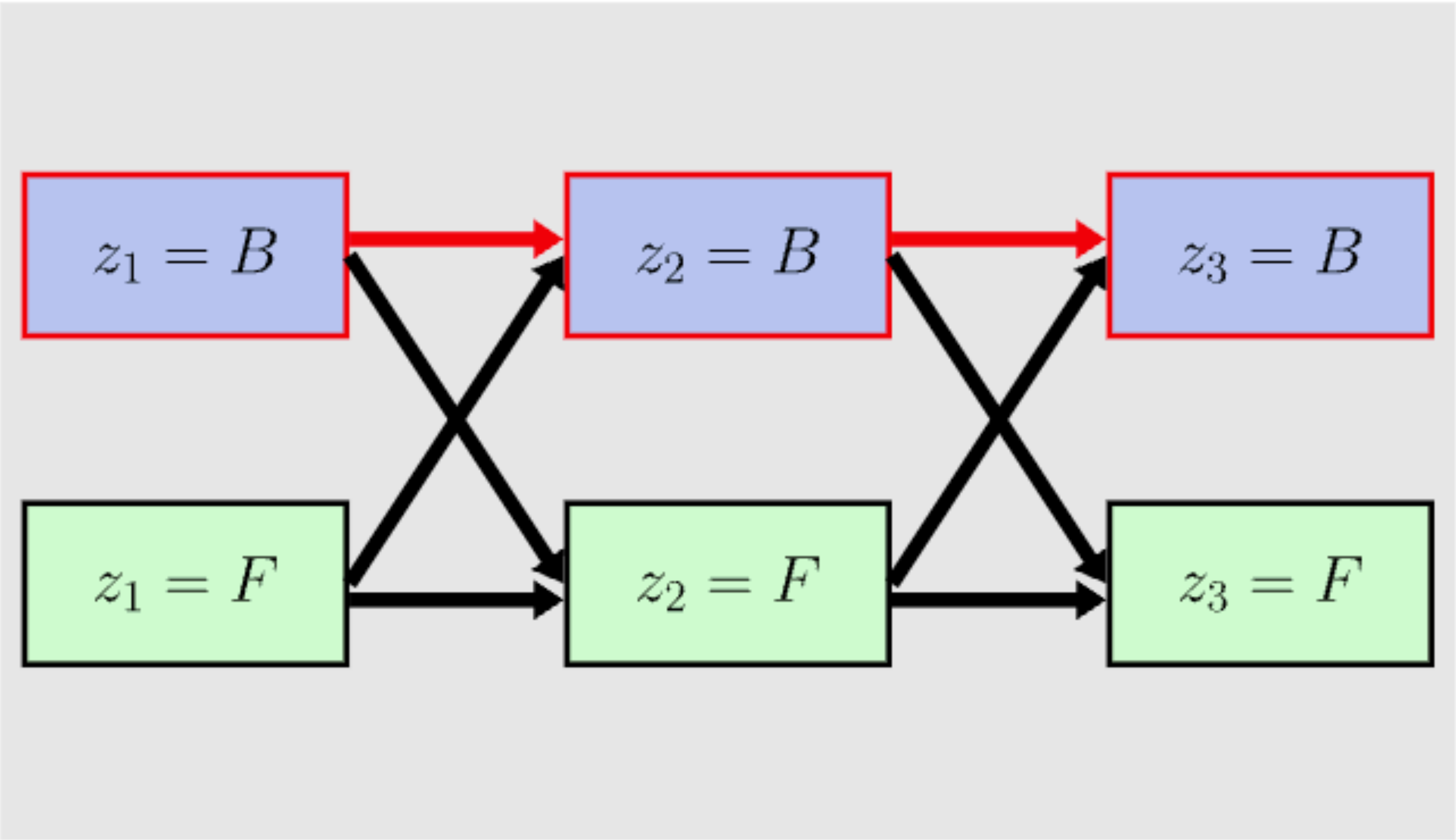
HMMs: Evidence Likelihood



- Example: How can we compute $L(HHH \mid \theta)$?

HMMs: Evidence Likelihood

- Example: How can we compute $L(HHH|\theta)$?



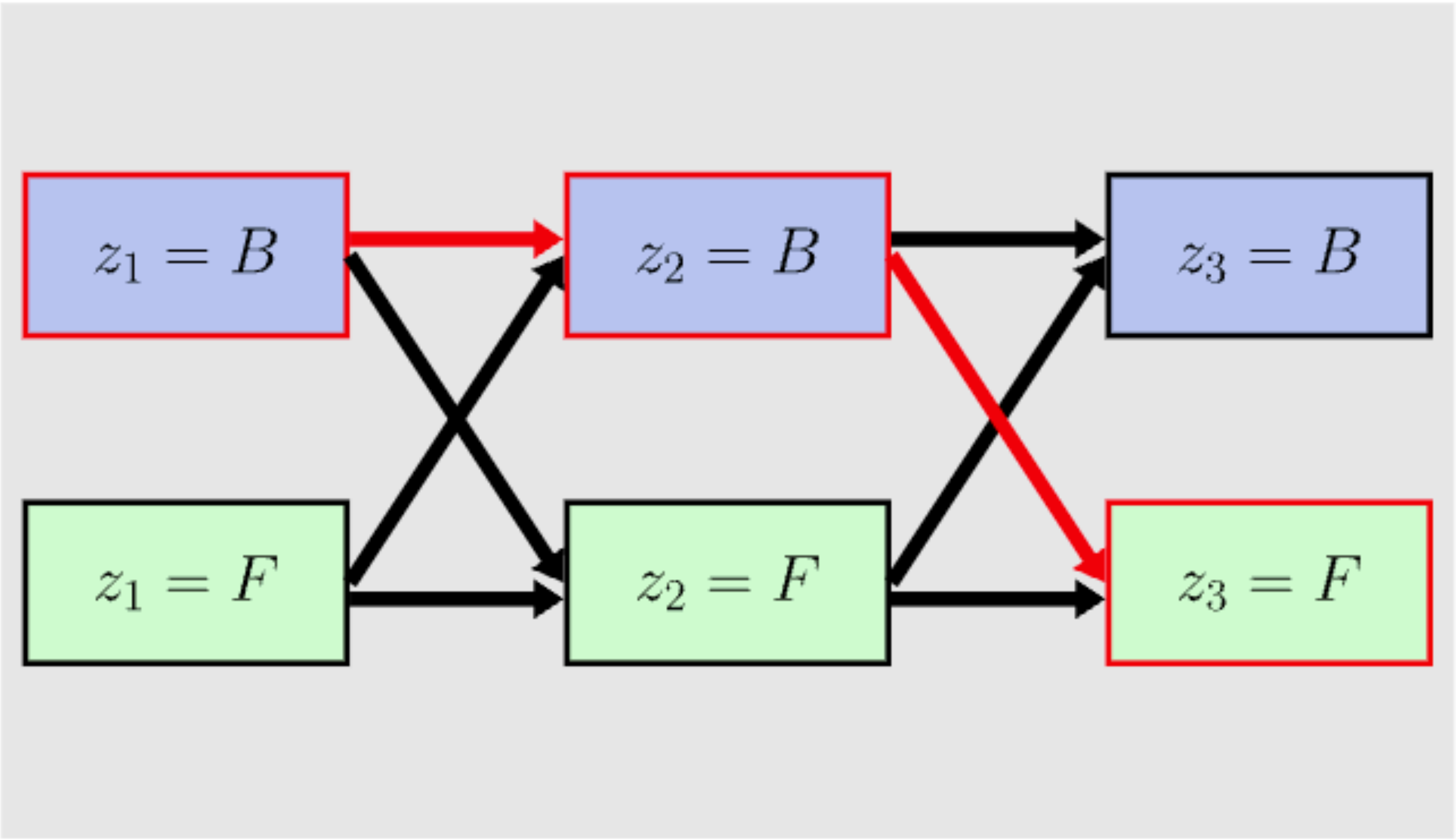
Path 1

$$P(HHH|BBB) = P(B)P(H|B)P(B|B)P(H|B)P(B|B)P(H|B)$$

https://nipunbatra.github.io/hmm/

HMMs: Evidence Likelihood

- Example: How can we compute $L(HHH \mid \theta)$?



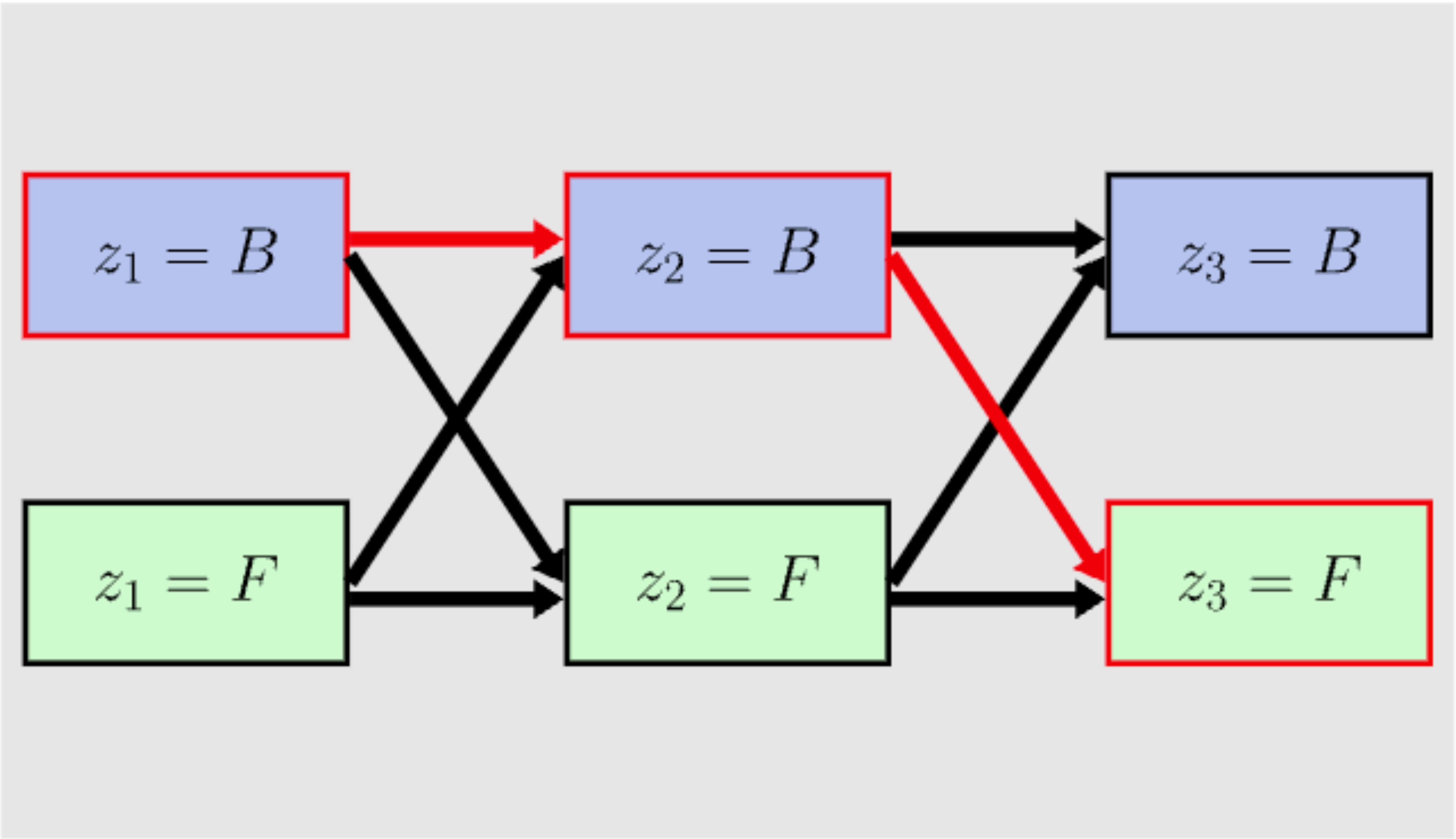
Path 2

$$P(HHH|BBF) = P(B)P(H|B)P(B|B)P(H|B)P(F|B)P(H|F)$$

https://nipunbatra.github.io/hmm/

HMMs: Evidence Likelihood

- Example: How can we compute $L(HHH \mid \theta)$?



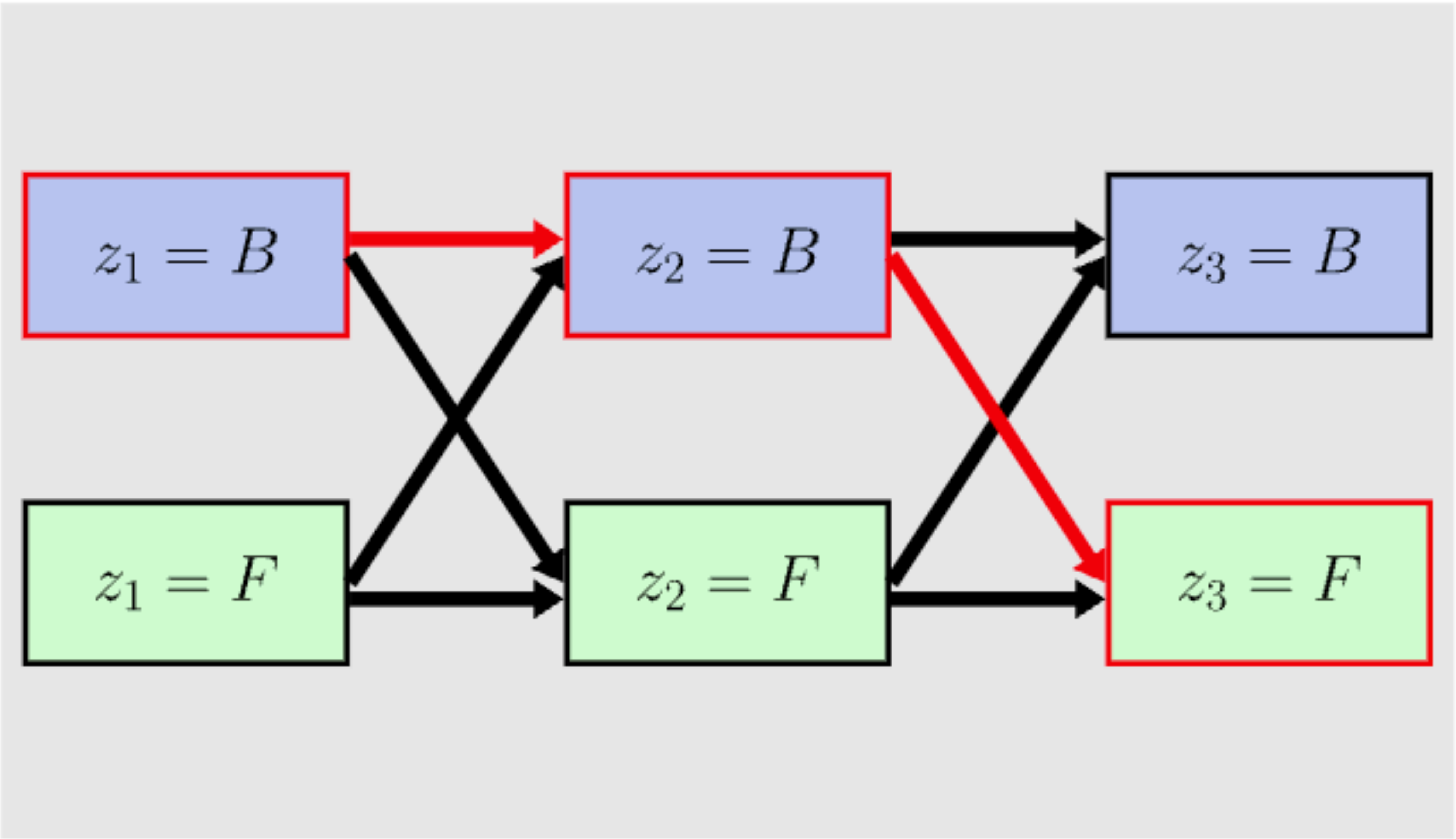
Path 2

$$P(HHH|BBF) = P(B)P(H|B)P(B|B)P(H|B)P(F|B)P(H|F)$$

- Here, for $K = 2$ and $T = 3$, there are $2^3 = 8$ possible paths
- $L(HHH \mid \theta)$ is the sum of probabilities across the 8 paths

HMMs: Evidence Likelihood

- Example: How can we compute $L(HHH \mid \theta)$?



Path 2

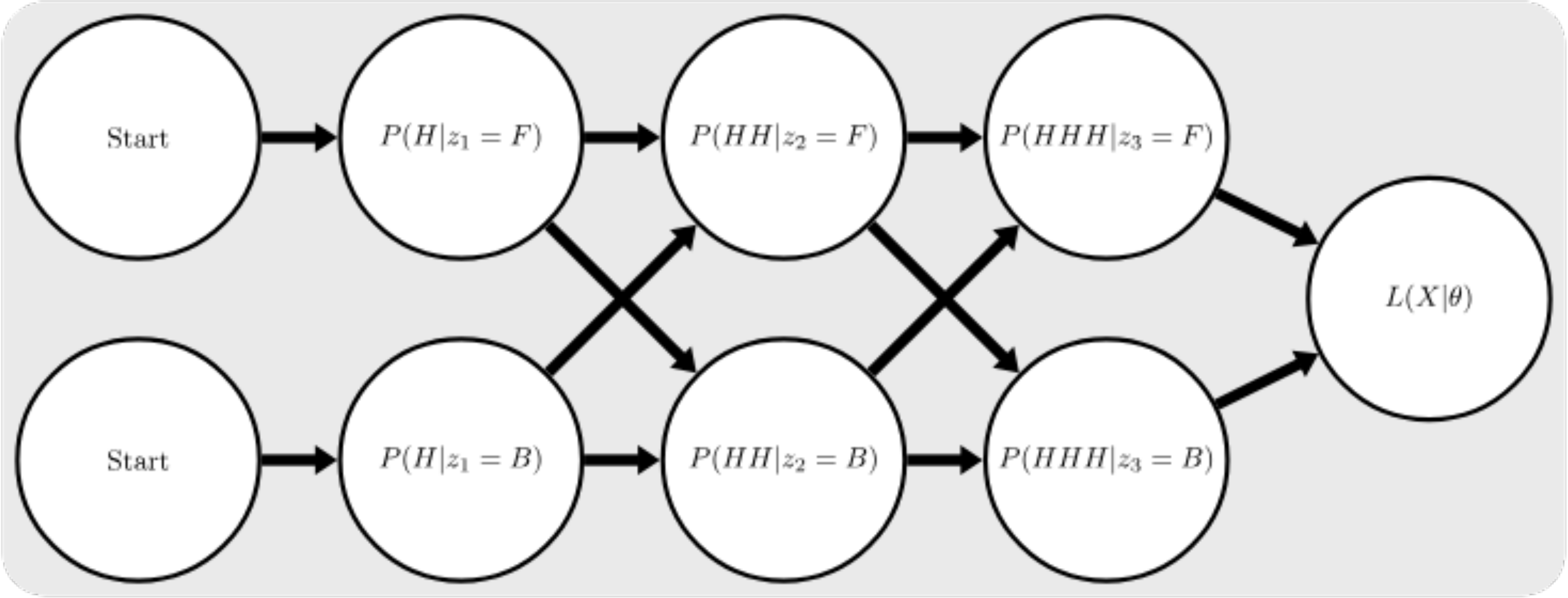
$P(HHH|BBF) = P(B)P(H|B)P(B|B)P(H|B)P(F|B)P(H|F)$

- Here, for $K = 2$ and $T = 3$, there are $2^3 = 8$ possible paths
- $L(HHH \mid \theta)$ is the sum of probabilities across the 8 paths
- In general, there are K^T paths

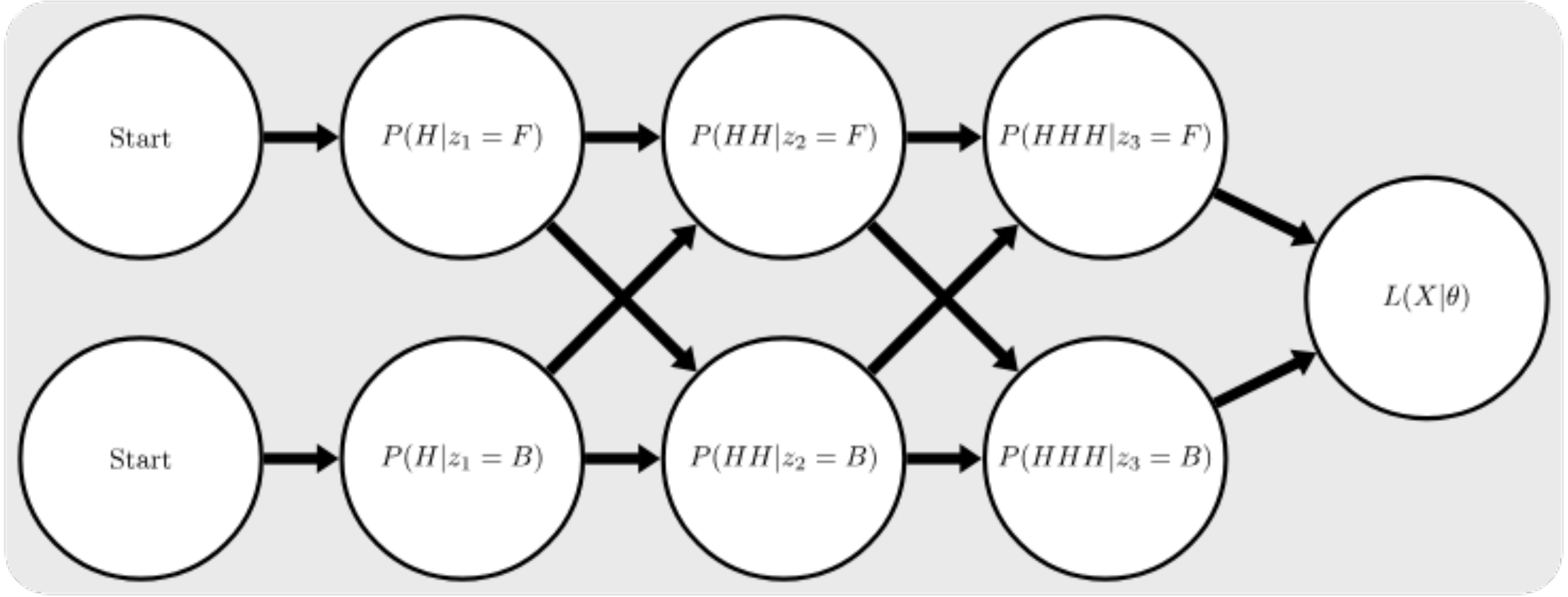
https://nipunbatra.github.io/hmm/

HMMs: Efficient Calculation via Forward Algorithm

HMMs: Efficient Calculation via Forward Algorithm



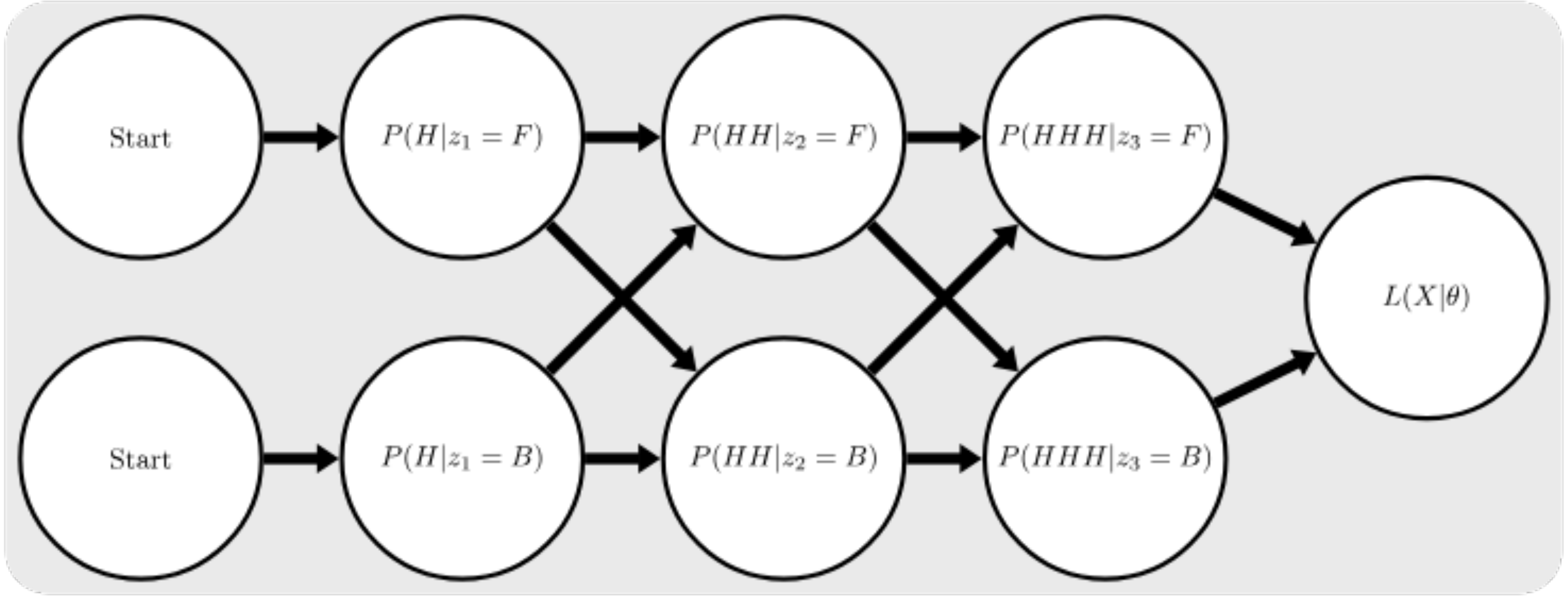
HMMs: Forward Algorithm



$$\alpha_t(i) = P(X_{1:t} | z_t = i)$$

- The probability of data up to time *t* given a particular latent at time *t*

HMMs: Forward Algorithm



$$\alpha_t(i) = P(X_{1:t} | z_t = i)$$

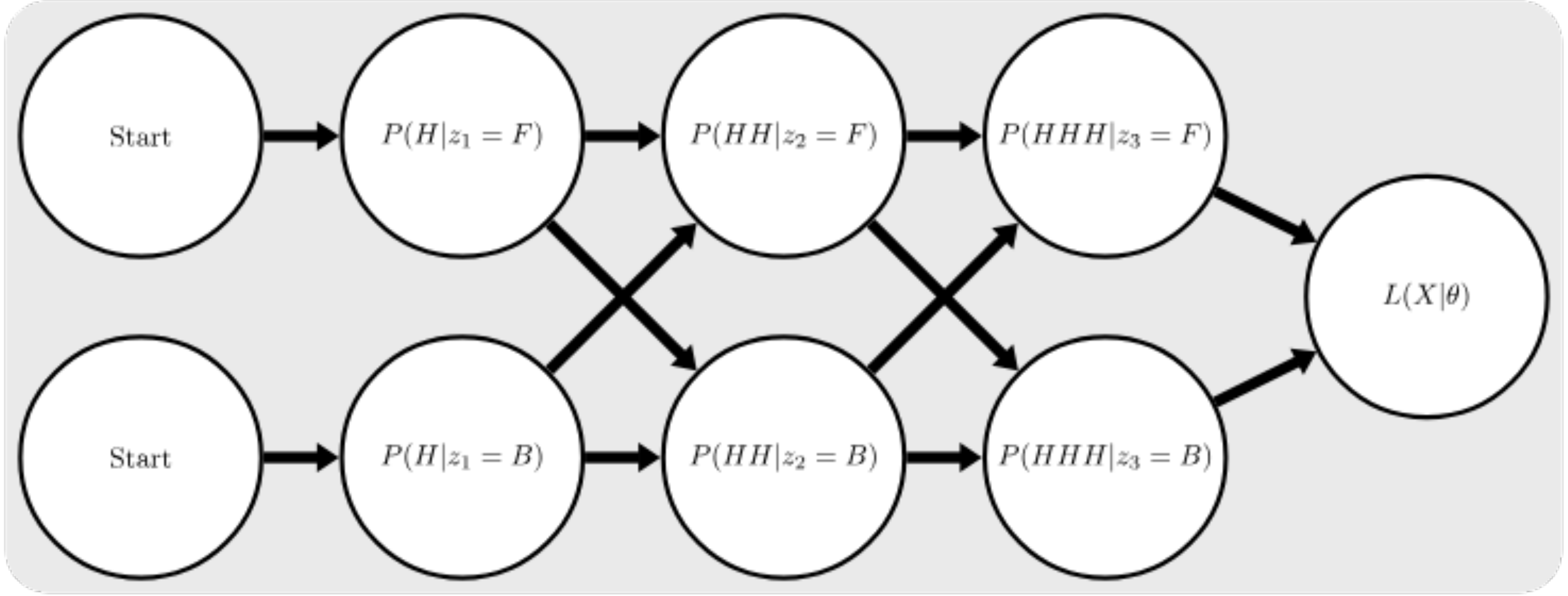
- The probability of data up to time *t* given a particular latent at time *t*

Initial Step:

$$\alpha_1(i) = P(x_1 = i) \cdot P(x_1 | z_1 = i)$$

$$= \pi_i \cdot \phi_j(x_1)$$

HMMs: Forward Algorithm



$$\alpha_t(i) = P(X_{1:t} \,|\, z_t = i)$$

- The probability of data up to time *t* given a particular latent at time *t*

<u>Initial Step:</u>

$$\alpha_1(i) = P(x_1 = i) \cdot P(x_1 \,|\, z_1 = i)$$

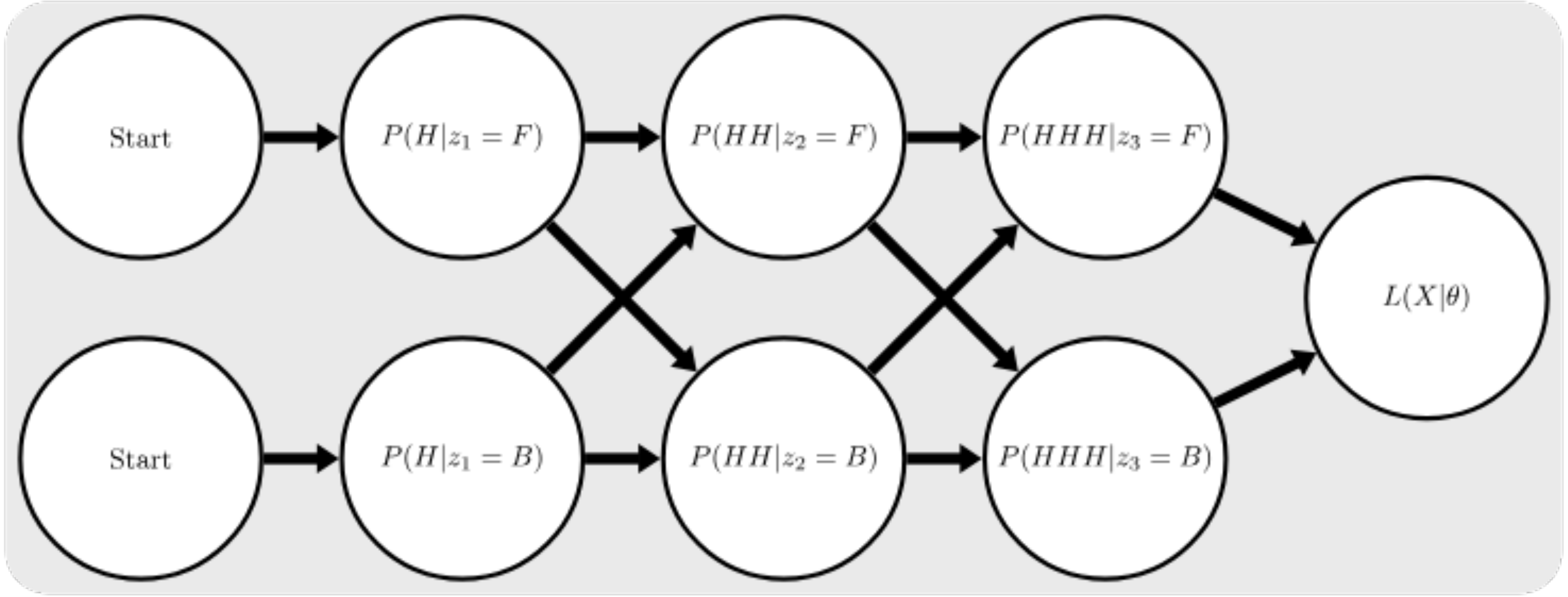
$$= \pi_i \cdot \phi_j(x_1)$$

<u>Future Steps:</u>

$$\alpha_{t+1}(j) = \{ \sum_{i=1}^{K} \alpha_t(i) \cdot P(x_{t+1} = j \,|\, P(x_t = i)\} \cdot P(x_{t+1} \,|\, z_{t+1} = j)$$

$$= \{ \sum_{i=1}^{K} \alpha_t(i) \cdot A_{ij}\} \cdot \phi_j(x_{t+1})$$

HMMs: Forward Algorithm



$$\alpha_t(i) = P(X_{1:t} | z_t = i)$$

- The probability of data up to time *t* given a particular latent at time *t*

<u>Initial Step:</u>

$$\alpha_1(i) = P(x_1 = i) \cdot P(x_1 | z_1 = i)$$

$$= \pi_i \cdot \phi_j(x_1)$$

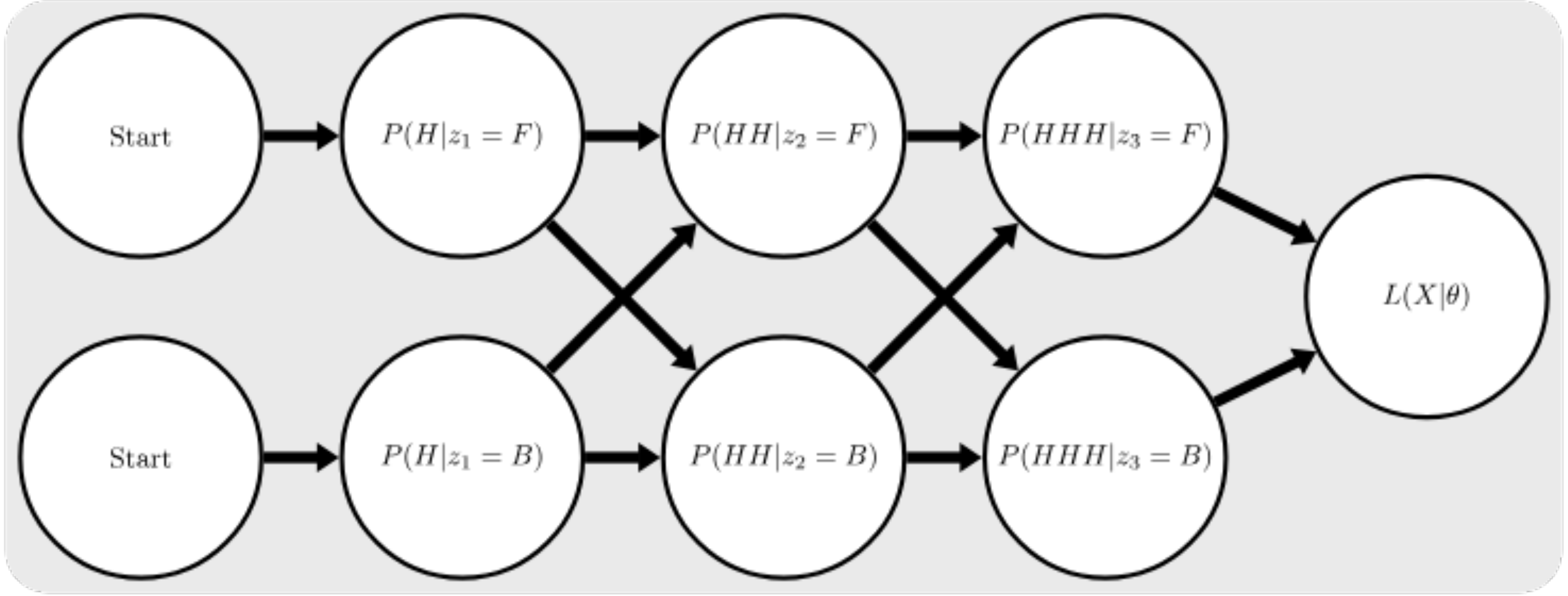
<u>Future Steps:</u>

$$\alpha_{t+1}(j) = \{ \sum_{i=1}^{K} \alpha_t(i) \cdot P(x_{t+1} = j | P(x_t = i)\} \cdot P(x_{t+1} | z_{t+1} = j)$$

$$= \{ \sum_{i=1}^{K} \alpha_t(i) \cdot A_{ij}\} \cdot \phi_j(x_{t+1})$$

$$\alpha_{t+1} = \phi_{t+1} \circ (A^T \alpha_t)$$

HMMs: Forward Algorithm



$$\alpha_t(i) = P(X_{1:t} | z_t = i)$$

- The probability of data up to time *t* given a particular latent at time *t*

Initial Step:

$$\alpha_1(i) = P(x_1 = i) \cdot P(x_1 | z_1 = i)$$

$$= \pi_i \cdot \phi_j(x_1)$$

Future Steps:

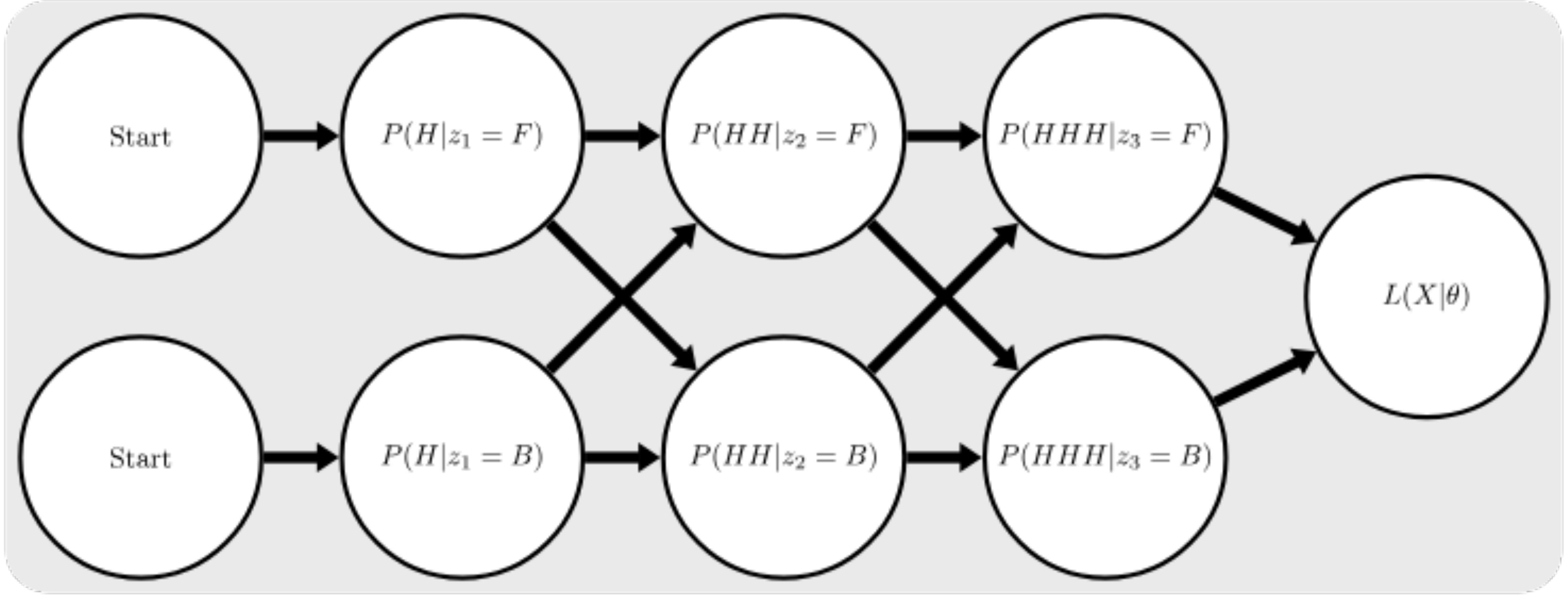
$$\alpha_{t+1}(j) = \{ \sum_{i=1}^{K} \alpha_t(i) \cdot P(x_{t+1} = j | P(x_t = i)\} \cdot P(x_{t+1} | z_{t+1} = j)$$

$$= \{ \sum_{i=1}^{K} \alpha_t(i) \cdot A_{ij}\} \cdot \phi_j(x_{t+1})$$

$$\alpha_{t+1} = \phi_{t+1} \circ (A^T \alpha_t)$$

- At each step, normalize so the α's sum to 1!

HMMs: Forward Algorithm



$$\alpha_t(i) = P(X_{1:t} | z_t = i)$$

- The probability of data up to time *t* given a particular latent at time *t*

Initial Step:

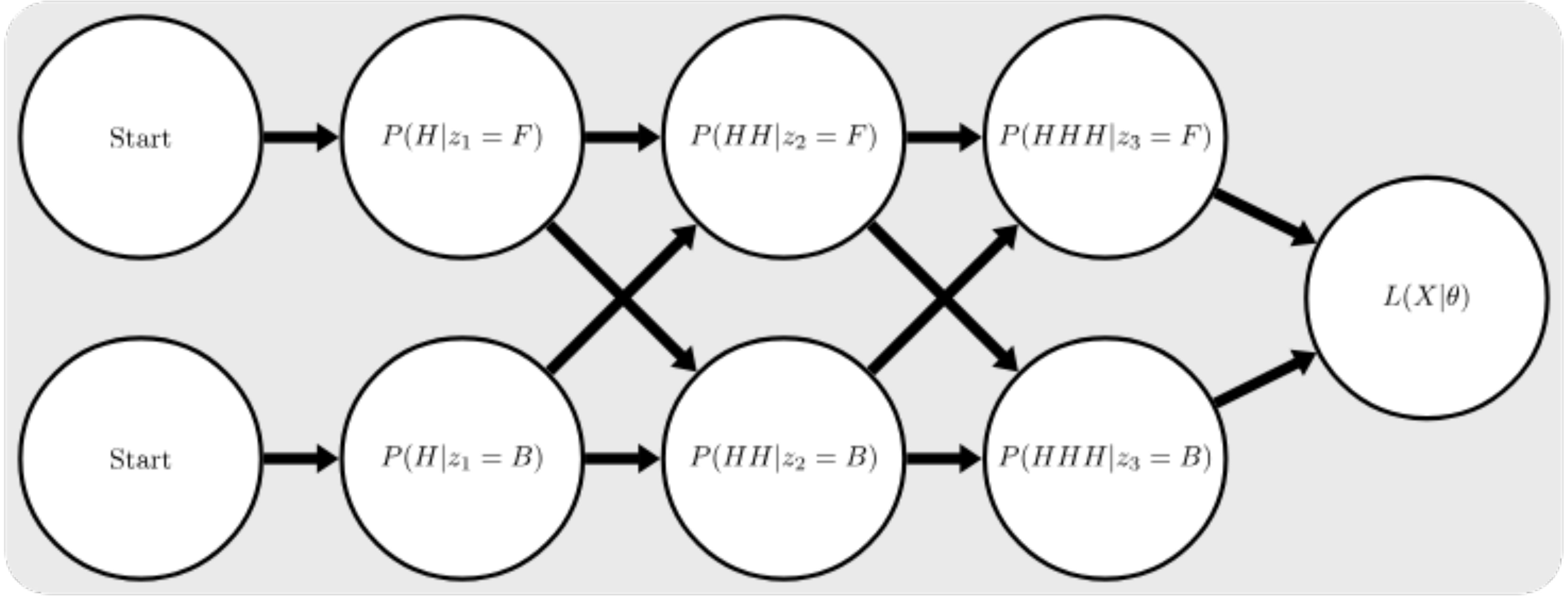
$$\alpha_1(i) = P(x_1 = i) \cdot P(x_1 | z_1 = i)$$
$$= \pi_i \cdot \phi_j(x_1)$$

Future Steps:

$$\alpha_{t+1}(j) = \{ \sum_{i=1}^{K} \alpha_t(i) \cdot P(x_{t+1} = j | P(x_t = i)\} \cdot P(x_{t+1} | z_{t+1} = j)$$
$$= \{ \sum_{i=1}^{K} \alpha_t(i) \cdot A_{ij}\} \cdot \phi_j(x_{t+1})$$
$$\alpha_{t+1} = \phi_{t+1} \circ (A^T \alpha_t)$$

- At each step, normalize so the α's sum to 1!
- We can view $\alpha_t(i)$ as the probability of being in latent state *i* given the data up to time *t*

HMMs: Forward Algorithm



$$\alpha_t(i) = P(X_{1:t} \,|\, z_t = i)$$

- The probability of data up to time *t* given a particular latent at time *t*

Initial Step:

$$\alpha_1(i) = P(x_1 = i) \cdot P(x_1 \,|\, z_1 = i)$$
$$= \pi_i \cdot \phi_j(x_1)$$

Future Steps:

$$\alpha_{t+1}(j) = \{ \sum_{i=1}^{K} \alpha_t(i) \cdot P(x_{t+1} = j \,|\, P(x_t = i) \} \cdot P(x_{t+1} \,|\, z_{t+1} = j)$$
$$= \{ \sum_{i=1}^{K} \alpha_t(i) \cdot A_{ij} \} \cdot \phi_j(x_{t+1})$$
$$\alpha_{t+1} = \phi_{t+1} \circ (A^T \alpha_t)$$

- At each step, normalize so the α's sum to 1!
- We can view $\alpha_t(i)$ as the probability of being in latent state *i* given the data up to time *t*
- Time complexity of $O(TK^2)$

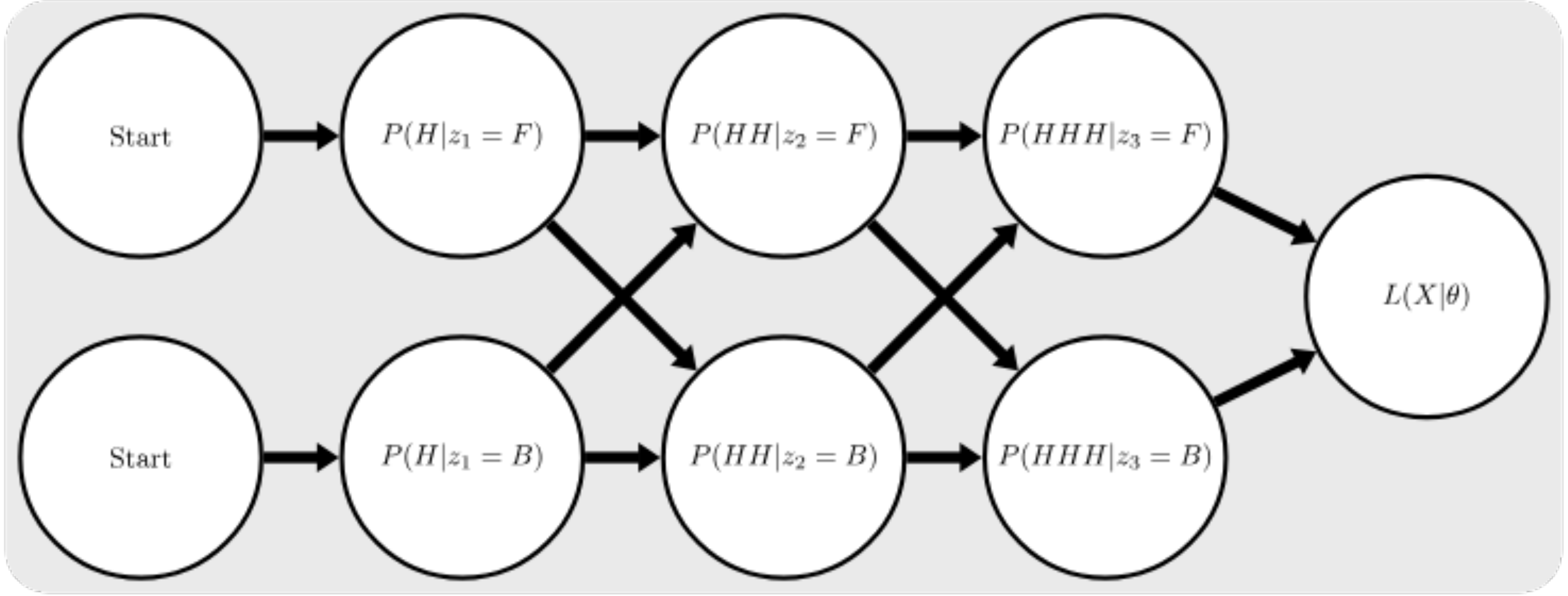
HMMs: EM Algorithm

- E-step: Learn expected states and transitions
- M-step: Optimize parameters

HMMs: E step

- Want to learn $P(z_t | X_{1:T})$
- Can get that from $P(X_{1:T} | z_t)$ and Bayes' Rule
- Forward algorithm gave us $P(X_{1:t} | z_t)$
- But what about $P(X_{t+1:T} | z_t)$?
  - Backward algorithm!

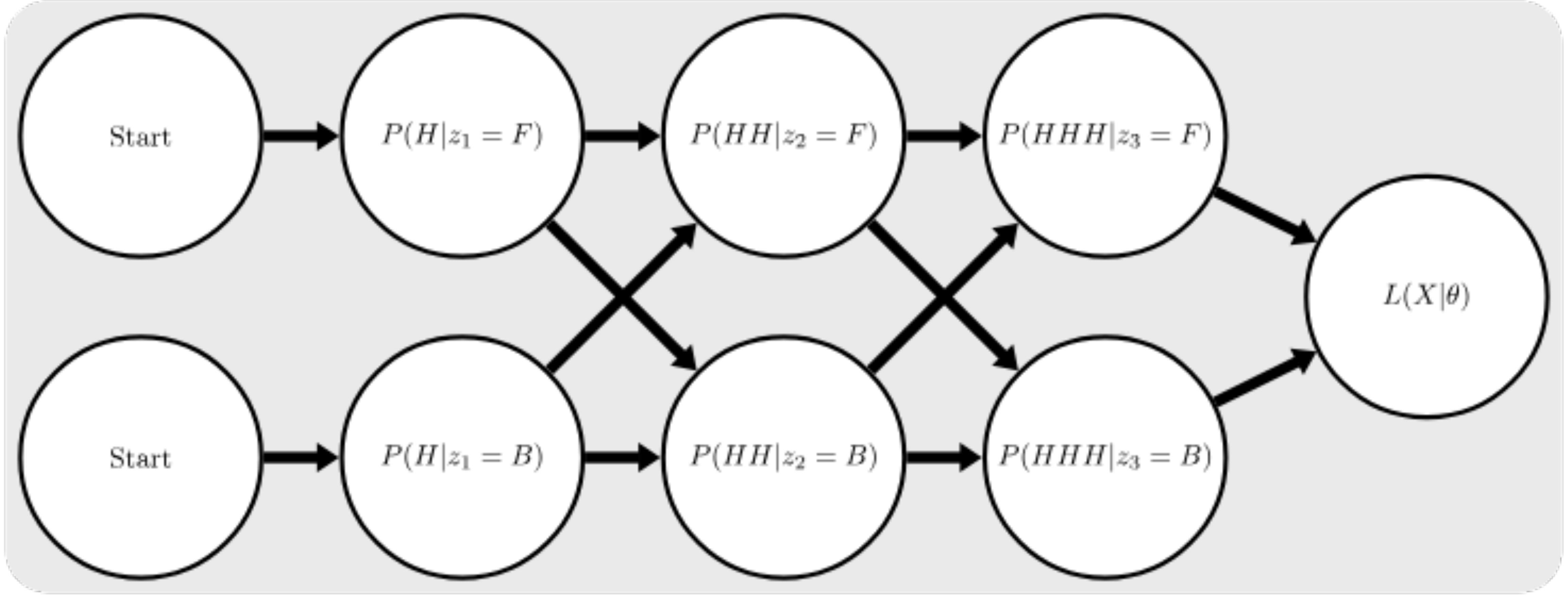
HMMs: Backward Algorithm



$$\beta_t(i) = P(X_{t+1:T} \,|\, z_t = i)$$

- The probability of data after time *t* given a particular latent at time *t*

HMMs: Backward Algorithm



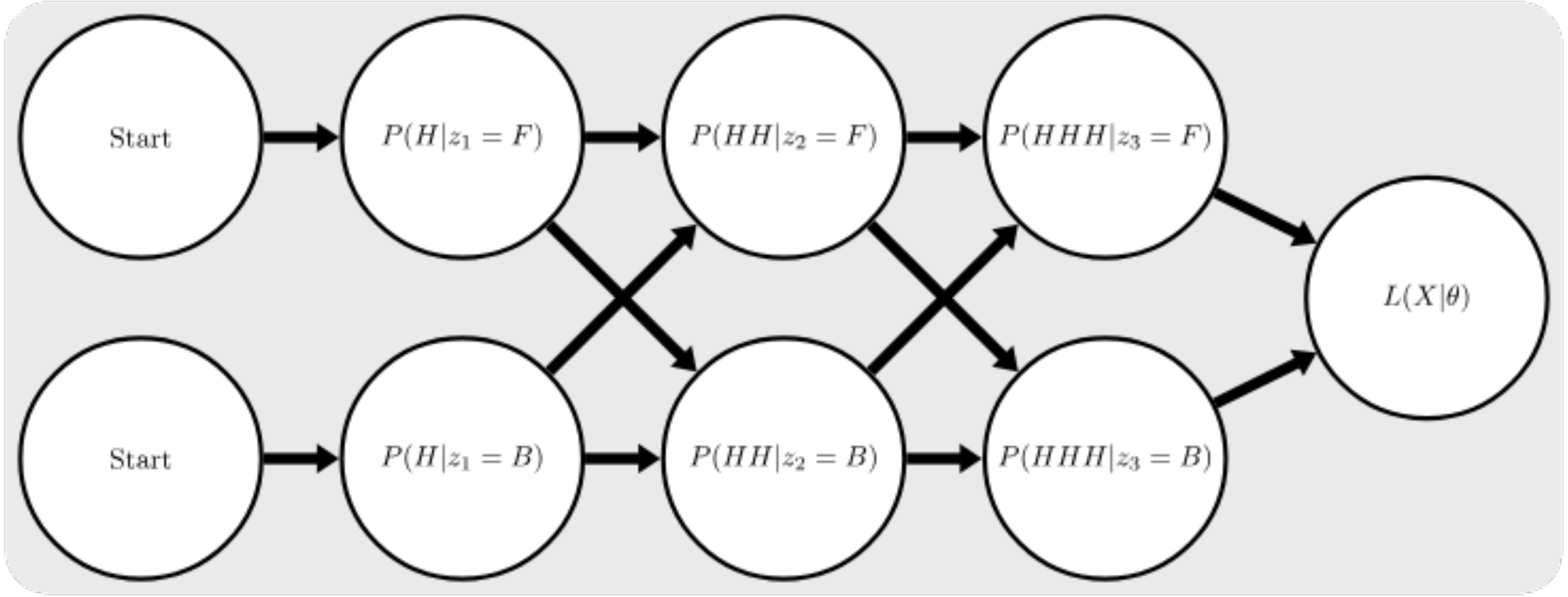
$$\beta_t(i) = P(X_{t+1:T} \mid z_t = i)$$

- The probability of data after time *t* given a particular latent at time *t*

<u>Initial Step:</u>

$$\beta_T(i) = 1$$

HMMs: Backward Algorithm



$$\beta_t(i) = P(X_{t+1:T} | z_t = i)$$

- The probability of data after time *t* given a particular latent at time *t*

Initial Step:

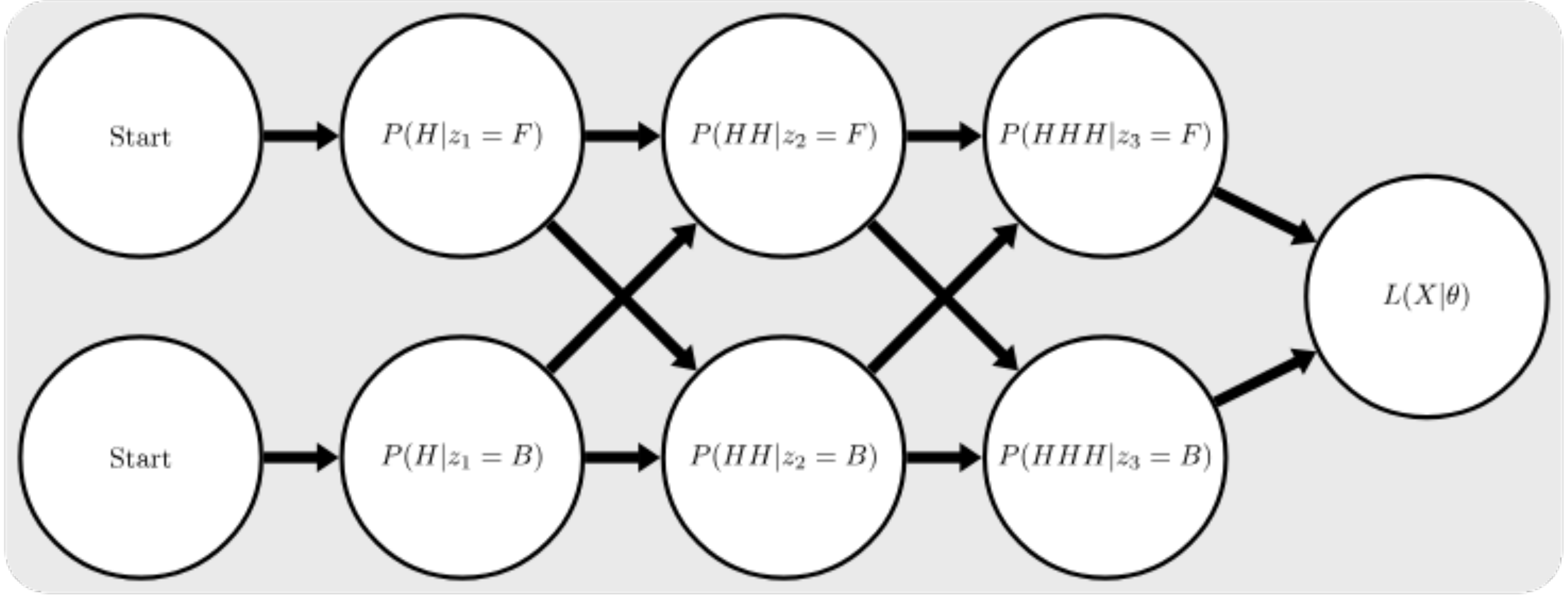
$$\beta_T(i) = 1$$

Future Steps:

$$\beta_t(j) = \sum_{i=1}^{K} \beta_{t+1}(i) \cdot P(x_{t+1} | z_{t+1} = i) \cdot P(x_{t+1} = i | P(x_t = j)$$

$$= \sum_{i=1}^{K} \beta_{t+1}(i) \cdot \phi_i(x_{t+1}) \cdot A_{ji}$$

HMMs: Backward Algorithm



$$\beta_t(i) = P(X_{t+1:T} | z_t = i)$$

- The probability of data after time *t* given a particular latent at time *t*

Initial Step:

$$\beta_T(i) = 1$$

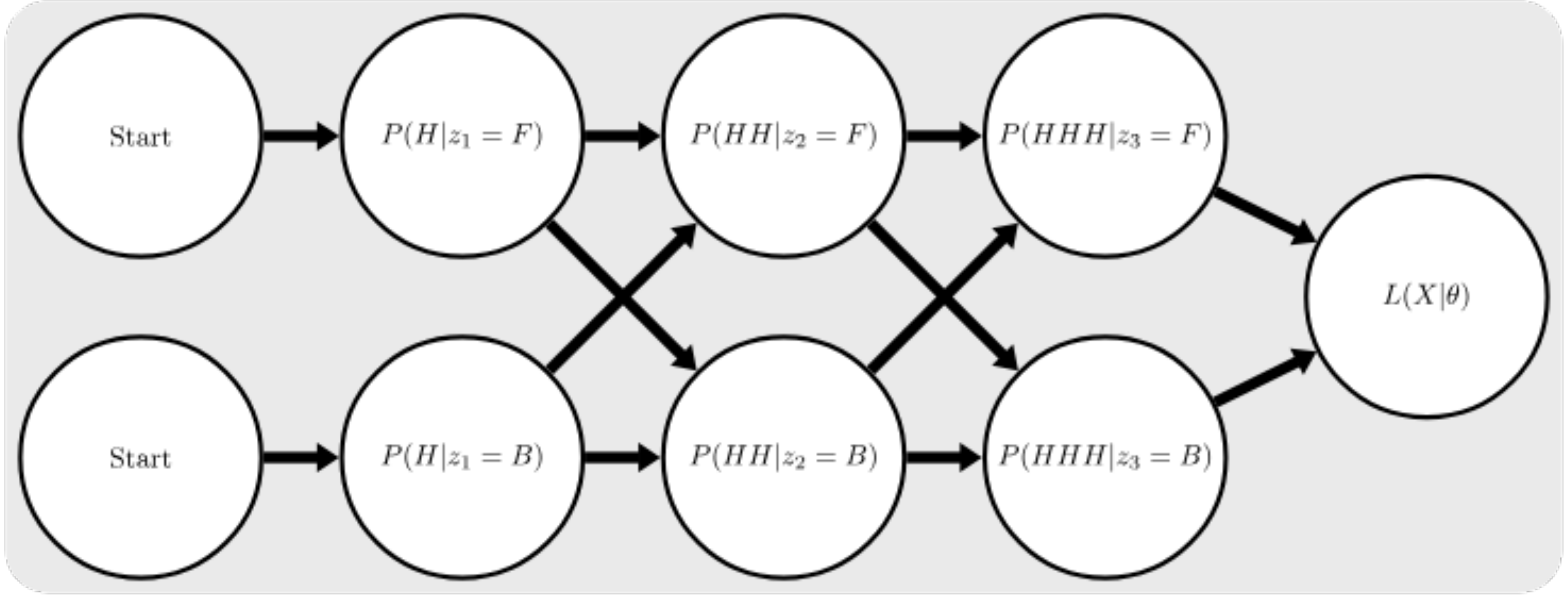
Future Steps:

$$\beta_t(j) = \sum_{i=1}^{K} \beta_{t+1}(i) \cdot P(x_{t+1} | z_{t+1} = i) \cdot P(x_{t+1} = i | P(x_t = j)$$

$$= \sum_{i=1}^{K} \beta_{t+1}(i) \cdot \phi_i(x_{t+1}) \cdot A_{ji}$$

$$\beta_t = A(\phi_{t+1} \circ \beta_{t+1})$$

HMMs: Backward Algorithm



$$\beta_t(i) = P(X_{t+1:T} | z_t = i)$$

- The probability of data after time *t* given a particular latent at time *t*

<u>Initial Step:</u>

$$\beta_T(i) = 1$$

<u>Future Steps:</u>

$$\beta_t(j) = \sum_{i=1}^{K} \beta_{t+1}(i) \cdot P(x_{t+1} | z_{t+1} = i) \cdot P(x_{t+1} = i | P(x_t = j)$$

$$= \sum_{i=1}^{K} \beta_{t+1}(i) \cdot \phi_i(x_{t+1}) \cdot A_{ji}$$

$$\beta_t = A(\phi_{t+1} \circ \beta_{t+1})$$

- At each step, normalize so the β's sum to 1!
- We can view $\beta_t(i)$ as the probability of being in latent state *i* given the data after time *t*

HMMs: E step

$$P(z_t = i \mid X_{1:T}) \propto P(X_{1:T} \mid z_t = i)$$ If the prior is uniform

$$= P(X_{1:t} \mid z_t = i) P(X_{t+1:T} \mid z_t = i)$$

$$= \alpha_t(i) \beta_t(i)$$

- Expected number of times of being in state *i* at time *t*

HMMs: E step

- Expected number of times of being in state *i* at time *t*
  - Let $\gamma_t(i) = P(z_t = i \mid X_{1:T})$

HMMs: E step

- Expected number of times of being in state *i* at time *t*
  - Let $\gamma_t(i) = P(z_t = i \mid X_{1:T})$

- Expected number of transitions from state *i* to *j* at time *t*.
  - $\epsilon_t(i,j) = P(z_t = i, z_{t+1} = j \mid X_{1:T})$

HMMs: M step

Updating π

$\hat{\pi}_k$ denotes the expected fraction of sequences with $z_1 = k$.

$$\hat{\pi}_{\mathbf{k}} = \frac{\text{Expected number of sequences that start with } z_k}{\text{Total number of sequences}}$$

$$\hat{\pi}_k = \frac{\sum_{n=1}^{N} \gamma_{n,1}(k)}{N}$$

Assume there are N sequences
and $\gamma_{n,t}$ is for sequence *n* at time *t*

https://nipunbatra.github.io/hmm/

HMMs: M step

Updating A

$\hat{A_{jk}}$ denotes the expected probability of transitions from state i to state j.

$$\begin{aligned}\hat{A_{jk}} &= \frac{\text{Expected number of transitions from state i to state j}}{\text{Expected number of transitions from state i}} \\ &= \frac{\sum_{n=1}^{N}\sum_{t=1}^{T_i-1}\epsilon_{n,t}(j,k)}{\sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{t=1}^{T_n-1}\epsilon_{n,t}(j,k)}\end{aligned}$$

https://nipunbatra.github.io/hmm/

HMMs: M step

- With Bernoulli observations:

Updating ϕ

$\hat{\phi}_{jl}$ denotes the expected probability of observing l from state j.

$$\hat{\phi}_{jl} = \frac{\text{Expected number of times in state j and observing l}}{\text{Expected number of times in state j}}$$

$$= \frac{\sum_{n=1}^{N} \sum_{t \text{ where } x_{nt}=l}^{T_i} \gamma_{n,t}(j)}{\sum_{n=1}^{N} \sum_{t=1}^{T_i} \gamma_{n,t}(j)}$$

https://nipunbatra.github.io/hmm/

HMMs: M step

- With Gaussian observations:

$$\mu_j = \frac{\text{Expected sum of observations in state } j}{\text{Expected number of times in state } j}$$

$$\mu_j = \frac{\sum_{n=1}^{N} \sum_{t=1}^{T} \gamma_{n,t}(j)\ x_{n,t}}{\sum_{n=1}^{N} \sum_{t=1}^{T} \gamma_{n,t}(j)}$$

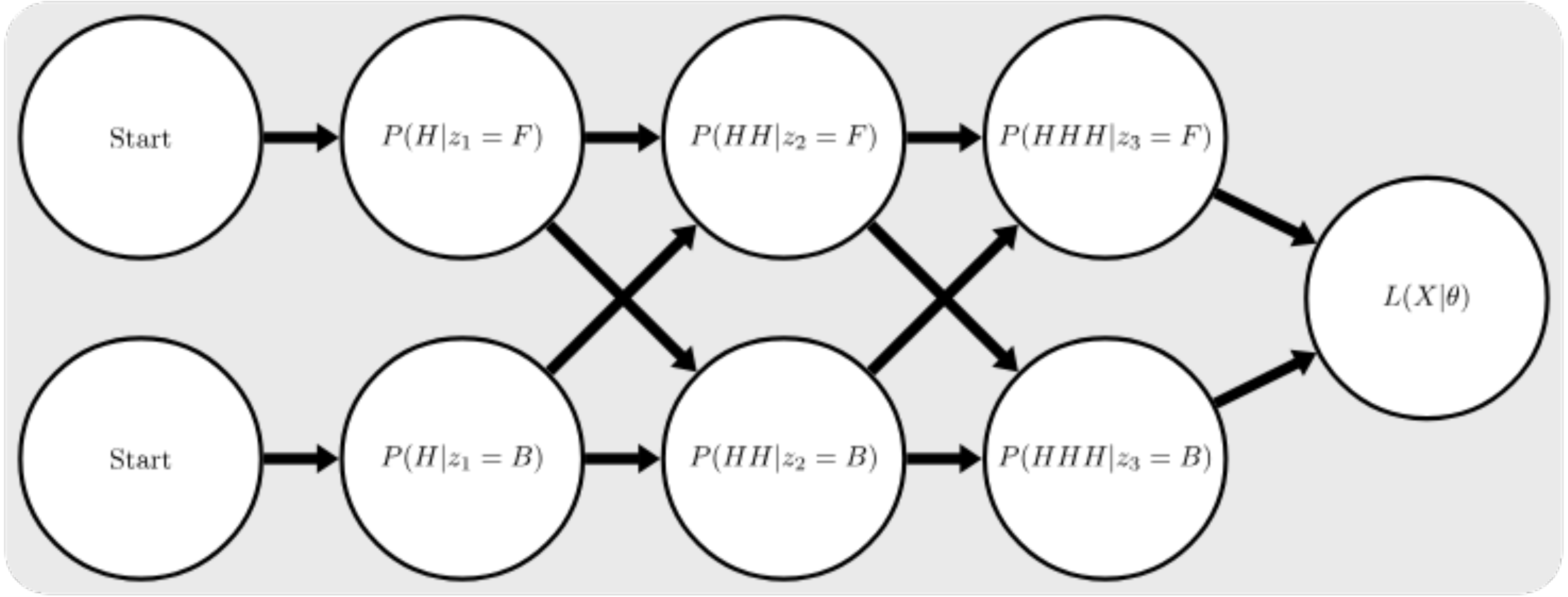
- Not shown: there is also a covariance parameter

HMMs: Viterbi Algorithm

- Another common goal is to find the single most likely sequence of discrete states: $\text{argmax}_{z_{1:T}} P(z_{1:T} | X_{1:T})$

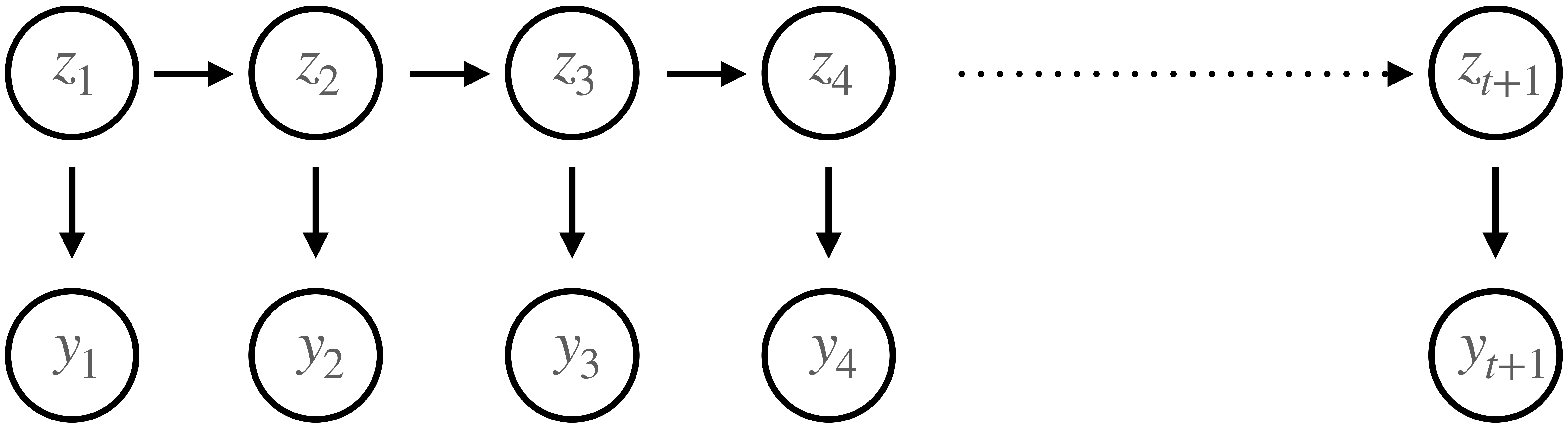
- Note that $P(z_t | X_{1:T})$ does not necessarily give this!

- Viterbi Algorithm is another iterative dynamic programming algorithm to find this sequence



Linear Dynamical Systems

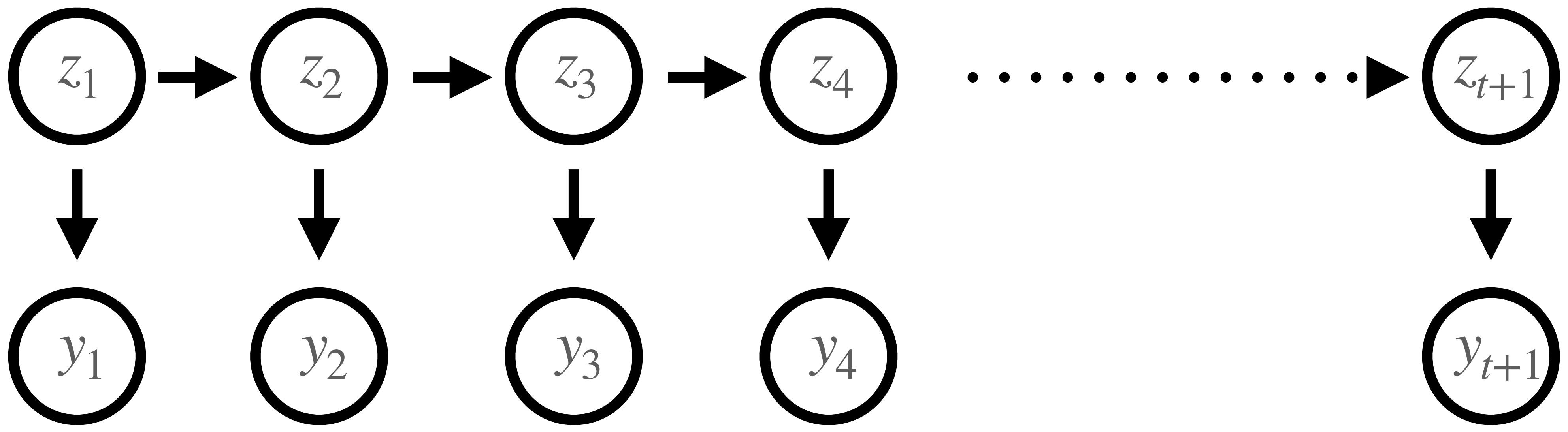
Linear Dynamical Systems



$$P(z_{1:T}, y_{1:T}) = P(z_1)\prod_{t=2}^{T} p(z_t \mid z_{t-1}) \prod_{t=1}^{T} p(y_t \mid z_t)$$

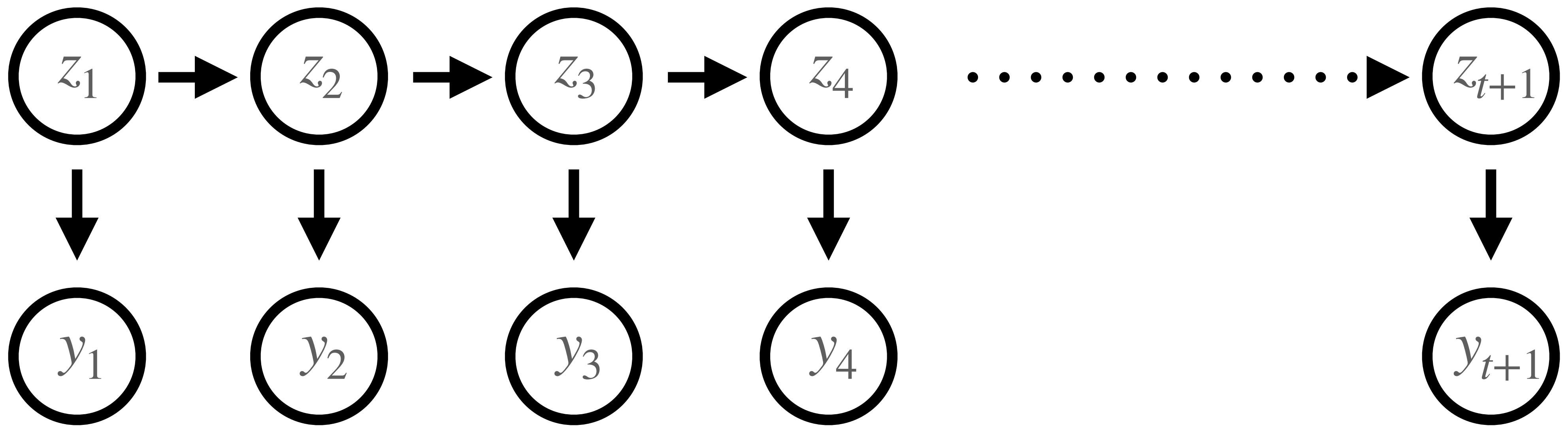
- Same graphical model as an HMM, but now z is continuous!
- Note the change in notation from *x* -> *y*

Linear Dynamical Systems



$$P(z_{1:T}, y_{1:T}) = P(z_1)\prod_{t=2}^{T} p(z_t \mid z_{t-1}) \prod_{t=1}^{T} p(y_t \mid z_t)$$

Linear Dynamical Systems



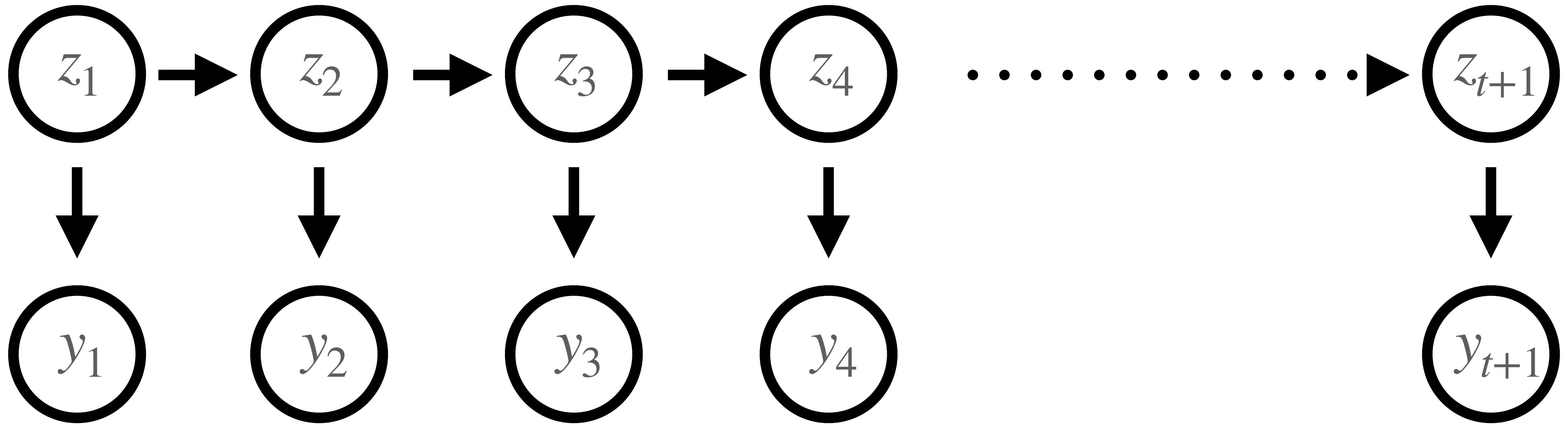
$$P(z_{1:T}, y_{1:T}) = P(z_1) \prod_{t=2}^{T} p(z_t \mid z_{t-1}) \prod_{t=1}^{T} p(y_t \mid z_t)$$

$\mathcal{N}(\boldsymbol{z}|\mathbf{0}, \mathbf{I})$

$\mathcal{N}(\boldsymbol{z}_t|\mathbf{F}_t \boldsymbol{z}_{t-1}, \mathbf{Q}_t)$

$\mathcal{N}(\boldsymbol{y}_t|\mathbf{H}_t \boldsymbol{z}_t, \mathbf{R}_t)$

Linear Dynamical Systems



$$P(z_{1:T}, y_{1:T}) = P(z_1) \prod_{t=2}^{T} p(z_t \,|\, z_{t-1}) \prod_{t=1}^{T} p(y_t \,|\, z_t)$$

$$\mathcal{N}(\boldsymbol{z}|\mathbf{0}, \mathbf{I}) \qquad \mathcal{N}(\boldsymbol{z}_t|\mathbf{F}_t \boldsymbol{z}_{t-1}, \mathbf{Q}_t) \qquad \mathcal{N}(\boldsymbol{y}_t|\mathbf{H}_t \boldsymbol{z}_t, \mathbf{R}_t)$$

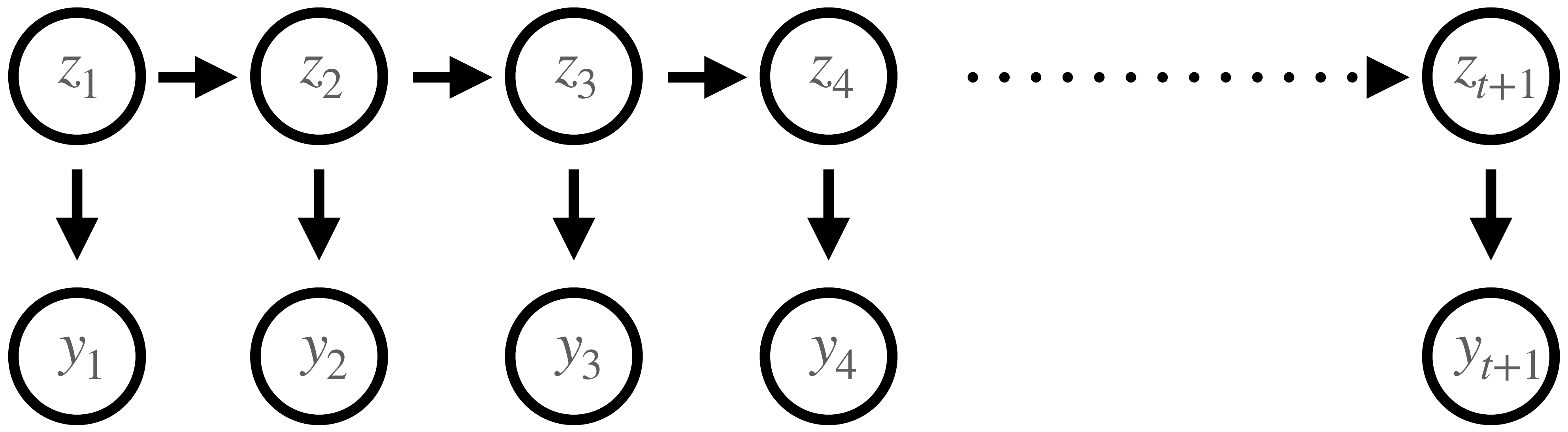
Or more fully:

$$\mathcal{N}(\boldsymbol{z}_t|\mathbf{F}_t \boldsymbol{z}_{t-1} + \mathbf{B}_t \boldsymbol{u}_t + \boldsymbol{b}_t, \mathbf{Q}_t)$$

Or more fully:

$$\mathcal{N}(\boldsymbol{y}_t|\mathbf{H}_t \boldsymbol{z}_t + \mathbf{D}_t \boldsymbol{u}_t + \boldsymbol{d}_t, \mathbf{R}_t)$$

Linear Dynamical Systems

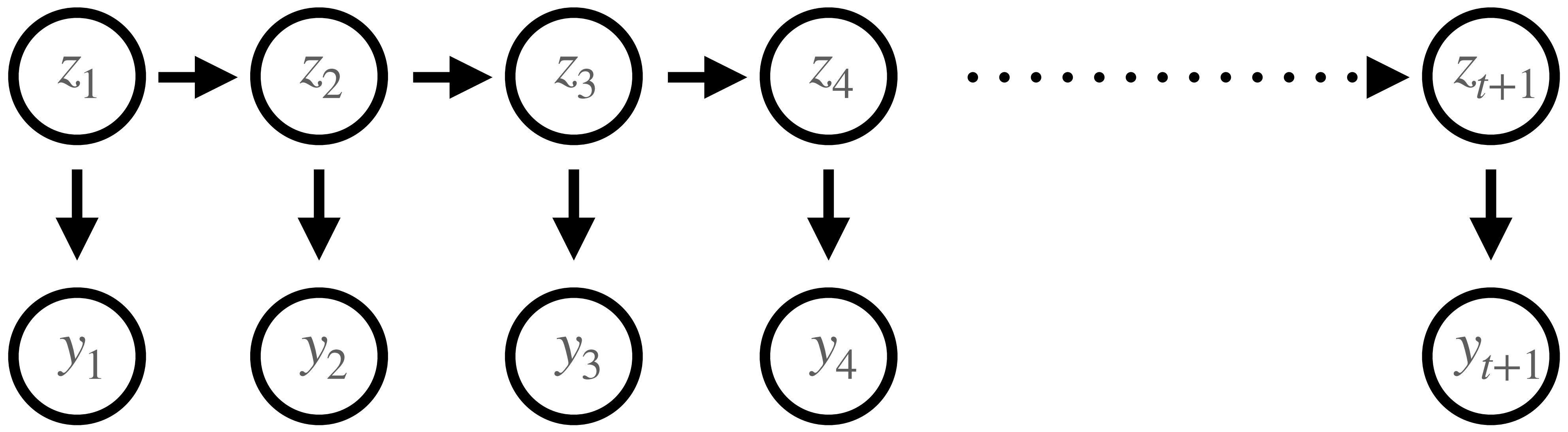


$$P(z_{1:T}, y_{1:T}) = P(z_1)\prod_{t=2}^{T} p(z_t \mid z_{t-1}) \prod_{t=1}^{T} p(y_t \mid z_t)$$

$$\mathcal{N}(\boldsymbol{z}|\mathbf{0}, \mathbf{I}) \qquad \mathcal{N}(\boldsymbol{z}_t|\mathbf{F}_t \boldsymbol{z}_{t-1}, \mathbf{Q}_t) \qquad \mathcal{N}(\boldsymbol{y}_t|\mathbf{H}_t \boldsymbol{z}_t, \mathbf{R}_t)$$

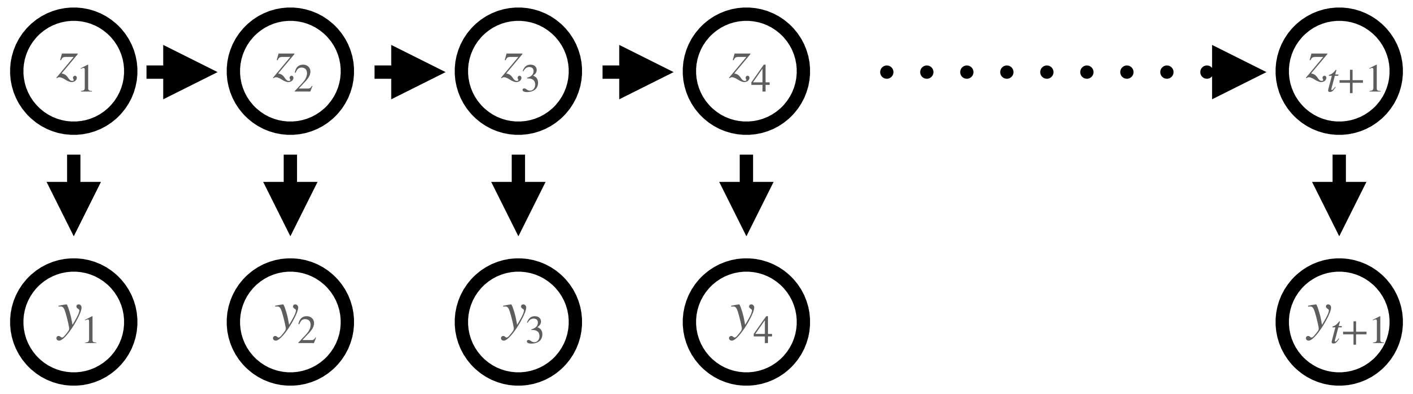
Just like the Factor Analysis Model,
but now with a dynamics term!

Simulating from an ~~HMM~~ LDS

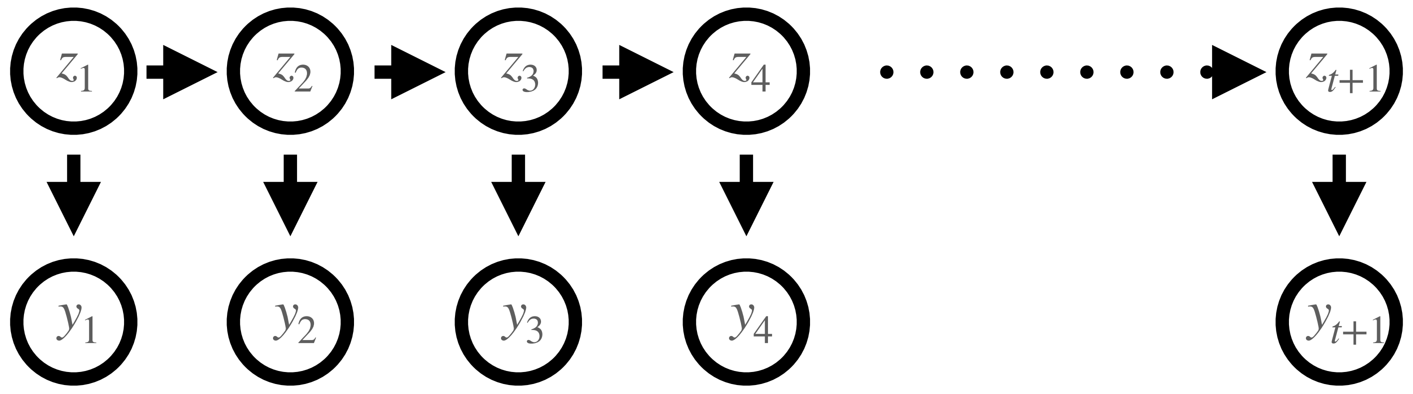


- Sample from $P(z_1)$
- Sample from $P(y_1 \mid z_1)$
- For all future time steps:
    - Sample $P(z_{t+1} \mid z_t)$
    - Sample $P(y_{t+1} \mid z_{t+1})$

LDS: Inferring the Latent States with a Kalman Filter



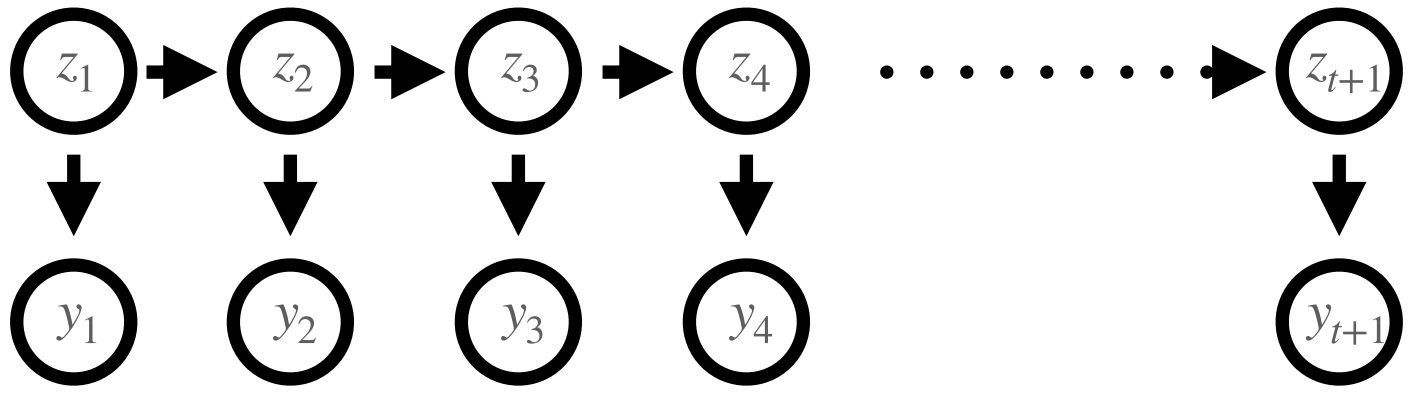
LDS: Inferring the Latent States with a Kalman Filter



- <u>Predict Step:</u>
    - Predict forward in time with dynamics model

- <u>Update Step:</u>
    - Use observations to modify dynamics-only prediction from above

LDS: Inferring the Latent States with a Kalman Filter

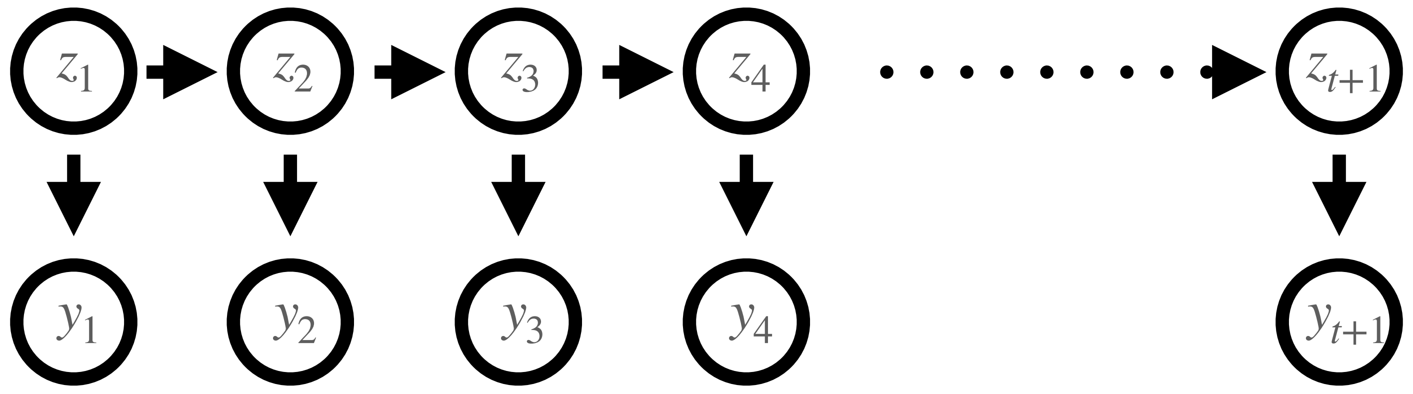


$$\mathcal{N}(\boldsymbol{z}_t|\mathbf{F}_t\boldsymbol{z}_{t-1}, \mathbf{Q}_t)$$

$$\mathcal{N}(\boldsymbol{y}_t|\mathbf{H}_t\boldsymbol{z}_t, \mathbf{R}_t)$$

- <u>Predict Step:</u>
  - Predict forward in time with dynamics model
- <u>Update Step:</u>
  - Use observations to modify dynamics-only prediction from above

LDS: Inferring the Latent States with a Kalman Filter



$$\mathcal{N}(\boldsymbol{z}_t | \mathbf{F}_t \boldsymbol{z}_{t-1}, \mathbf{Q}_t)$$

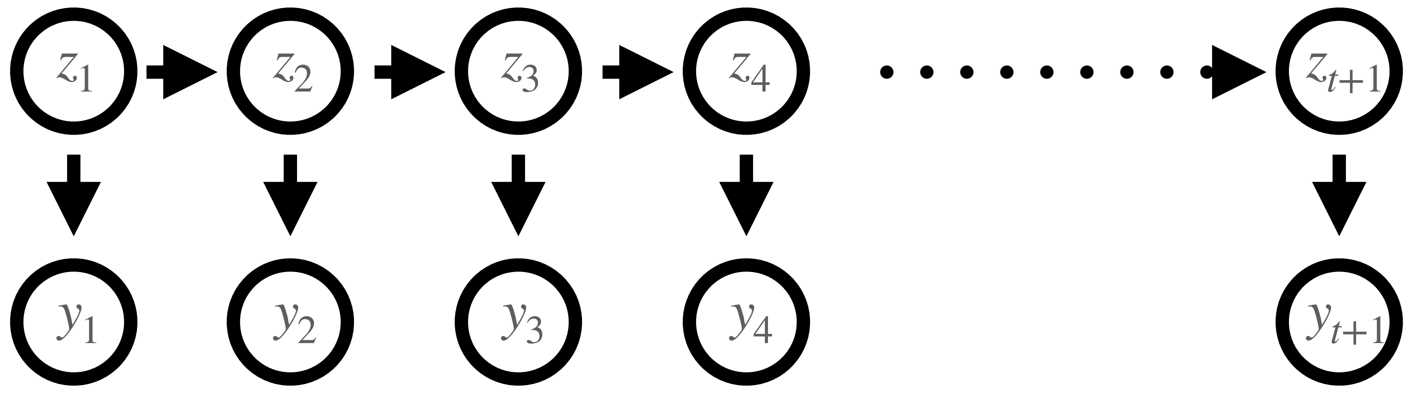
$$\mathcal{N}(\boldsymbol{y}_t | \mathbf{H}_t \boldsymbol{z}_t, \mathbf{R}_t)$$

- <u>Predict Step:</u>
  - Predict forward in time with dynamics model

$$\begin{aligned} p(\boldsymbol{z}_t | \boldsymbol{y}_{1:t-1}, \boldsymbol{u}_{1:t}) &= \mathcal{N}(\boldsymbol{z}_t | \boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1}) \\ \boldsymbol{\mu}_{t|t-1} &= \mathbf{F}_t \boldsymbol{\mu}_{t-1|t-1} + \mathbf{B}_t \boldsymbol{u}_t + \boldsymbol{b}_t \\ \boldsymbol{\Sigma}_{t|t-1} &= \mathbf{F}_t \boldsymbol{\Sigma}_{t-1|t-1} \mathbf{F}_t^\top + \mathbf{Q}_t \end{aligned}$$

- <u>Update Step:</u>
  - Use observations to modify dynamics-only prediction from above

LDS: Inferring the Latent States with a Kalman Filter



$$\mathcal{N}(\boldsymbol{z}_t|\mathbf{F}_t\boldsymbol{z}_{t-1}, \mathbf{Q}_t)$$

$$\mathcal{N}(\boldsymbol{y}_t|\mathbf{H}_t\boldsymbol{z}_t, \mathbf{R}_t)$$

- <u>Predict Step:</u>
  - Predict forward in time with dynamics model

$$p(\boldsymbol{z}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{u}_{1:t}) = \mathcal{N}(\boldsymbol{z}_t|\boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1})$$

$$\boldsymbol{\mu}_{t|t-1} = \mathbf{F}_t\boldsymbol{\mu}_{t-1|t-1} + \mathbf{B}_t\boldsymbol{u}_t + \boldsymbol{b}_t$$

$$\boldsymbol{\Sigma}_{t|t-1} = \mathbf{F}_t\boldsymbol{\Sigma}_{t-1|t-1}\mathbf{F}_t^{\mathsf{T}} + \mathbf{Q}_t$$

- <u>Update Step:</u>
  - Use observations to modify dynamics-only prediction from above

$$p(\boldsymbol{z}_t|\boldsymbol{y}_{1:t}, \boldsymbol{u}_{1:t}) = \mathcal{N}(\boldsymbol{z}_t|\boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t})$$

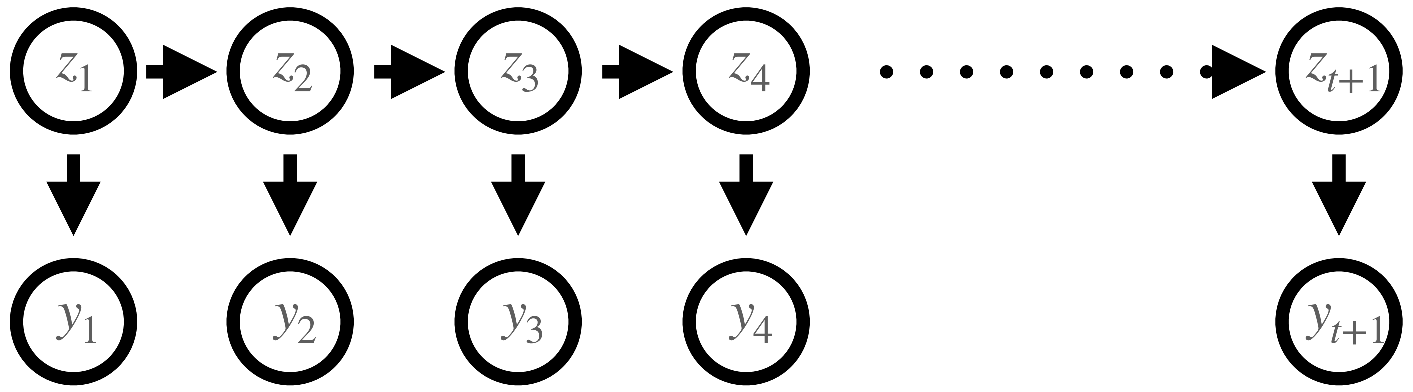
$$\hat{\boldsymbol{y}}_t = \mathbf{H}_t\boldsymbol{\mu}_{t|t-1} + \mathbf{D}_t\boldsymbol{u}_t + \boldsymbol{d}_t$$

$$\mathbf{S}_t = \mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^{\mathsf{T}} + \mathbf{R}_t$$

$$\mathbf{K}_t = \boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^{\mathsf{T}}\mathbf{S}_t^{-1}$$

$$\boldsymbol{\mu}_{t|t} = \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t(\boldsymbol{y}_t - \hat{\boldsymbol{y}}_t)$$

LDS: Inferring the Latent States with a Kalman Filter



$$\mathcal{N}(\boldsymbol{z}_t | \mathbf{F}_t \boldsymbol{z}_{t-1}, \mathbf{Q}_t)$$

$$\mathcal{N}(\boldsymbol{y}_t | \mathbf{H}_t \boldsymbol{z}_t, \mathbf{R}_t)$$

- <u>Predict Step:</u>
  - Predict forward in time with dynamics model

$$p(\boldsymbol{z}_t | \boldsymbol{y}_{1:t-1}, \boldsymbol{u}_{1:t}) = \mathcal{N}(\boldsymbol{z}_t | \boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1})$$

$$\boldsymbol{\mu}_{t|t-1} = \mathbf{F}_t \boldsymbol{\mu}_{t-1|t-1} + \mathbf{B}_t \boldsymbol{u}_t + \boldsymbol{b}_t$$

$$\boldsymbol{\Sigma}_{t|t-1} = \mathbf{F}_t \boldsymbol{\Sigma}_{t-1|t-1} \mathbf{F}_t^\mathsf{T} + \mathbf{Q}_t$$

- <u>Update Step:</u>
  - Use observations to modify dynamics-only prediction from above

$$p(\boldsymbol{z}_t | \boldsymbol{y}_{1:t}, \boldsymbol{u}_{1:t}) = \mathcal{N}(\boldsymbol{z}_t | \boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t})$$

$$\hat{\boldsymbol{y}}_t = \mathbf{H}_t \boldsymbol{\mu}_{t|t-1} + \mathbf{D}_t \boldsymbol{u}_t + \boldsymbol{d}_t$$

$$\mathbf{S}_t = \mathbf{H}_t \boldsymbol{\Sigma}_{t|t-1} \mathbf{H}_t^\mathsf{T} + \mathbf{R}_t$$

$$\mathbf{K}_t = \boldsymbol{\Sigma}_{t|t-1} \mathbf{H}_t^\mathsf{T} \mathbf{S}_t^{-1}$$

Observations Noise

Dynamics Noise

$$\boldsymbol{\mu}_{t|t} = \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t (\boldsymbol{y}_t - \hat{\boldsymbol{y}}_t)$$

LDS: Inferring the Latent States with a Kalman Smoother

- The Kalman Filter finds $p(z_t \mid y_{1:t}, u_{1:t})$
  - Comparable to the “forward” algorithm for HMMs

LDS: Inferring the Latent States with a Kalman Smoother

- The Kalman Filter finds $p(z_t | y_{1:t}, u_{1:t})$
  - Comparable to the "forward" algorithm for HMMs

- The Kalman Smoother finds $p(z_t | y_{1:T}, u_{1:T})$
  - Also includes a calculation going backwards in time, and is comparable to the "forward-backward" algorithm for HMMs

LDS: Model Fitting

- Most standard is EM
  - E-step is Kalman Smoother
  - M-step optimizes parameters
    - Can learn interesting things about the dynamics too!

- Subspace Identification (SSID) is also used

Resources

- Probabilistic Machine Learning Book 2, Chapter 29
  - https://probml.github.io/pml-book/book2.html
- Interactive HMM Website
  - https://nipunbatra.github.io/hmm/