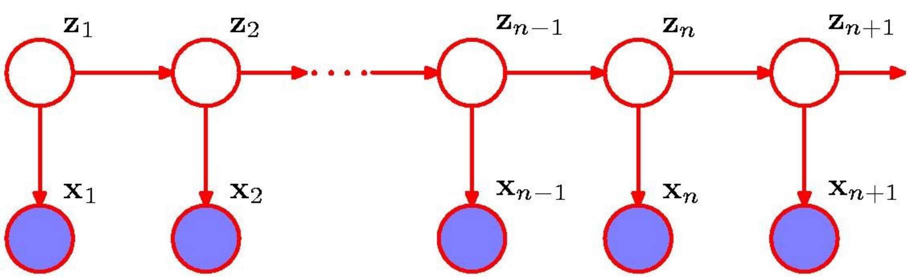
Hidden Markov Models

Terminology, Representation and Basic Problems



Christian Nørgaard Storm Pedersen

cstorm@birc.au.dk

The next two weeks

Hidden Markov models (HMMs):

Wed 3/11:	Terminology and basic algorithms.
Fri 5/11:	Implementing the basic algorithms.
Wed 10/11:	Implementing the basic algorithms, cont.
	Selecting model parameters and training.
Fri 12/11:	Selecting model parameters and training, cont.
	Extensions and applications.
Wed 17/11:	Talk about project.

We use Chapter 13 from Bishop's book “Pattern Recognition and Machine Learning”. Rabiner's paper “A Tutorial on Hidden Markov Models [...]” might also be useful to read.

BrightSpace and http://birc.au.dk/~cstorm/courses/ML_e21

What is machine learning?

Machine learning means different things to different people, and there is no general agreed upon core set of algorithms that must be learned.

For me, the core of machine learning is:

Building a mathematical model that captures some desired structure of the phenomenon/system that you are studying and you have data from.

Training the model (i.e. set the parameters of the model) based on existing data and knowledge to make it represent the system you are studying as good as possible.

Making predictions of the behavior of the system you are studying by using the trained model and new data, or study the trained model to learn about the system it is based upon

Data – Observations

A sequence of observations from a finite and discrete set, e.g. measurements of weather patterns, daily values of stocks, the composition of DNA or proteins, or ...

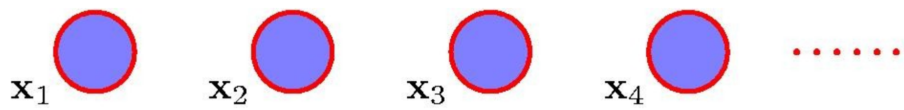
$$\mathbf{X} = \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$$

Typical question/problem: How likely is a given **X**, i.e. p(**X**)?

We need a model that describes how to compute p(**X**)

Simple Models (1)

Observations are independent and identically distributed



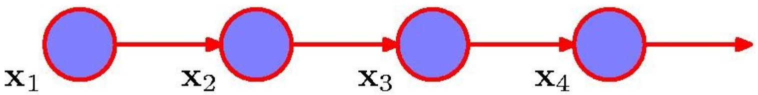
$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \prod_{n=1}^{N} p(\mathbf{x}_n)$$

Too simplistic for realistic modelling of many phenomena

Simple Models (2)

The *n*'th observation in a chain of observations is influenced only by the *n*-1'th observation, i.e.

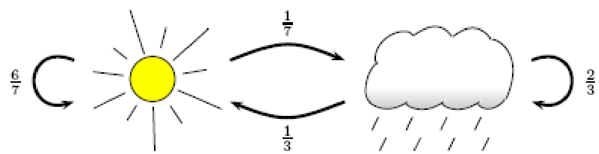
$$p(\mathbf{x}_n|\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}) = p(\mathbf{x}_n|\mathbf{x}_{n-1})$$



The chain of observations is a 1st-order Markov chain, and the probability of a sequence of *N* observations is

$$p(\mathbf{x}_1,\ldots,\mathbf{x}_N) = \prod_{n=1}^{N} p(\mathbf{x}_n|\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}) = p(\mathbf{x}_1)\prod_{n=2}^{N} p(\mathbf{x}_n|\mathbf{x}_{n-1})$$

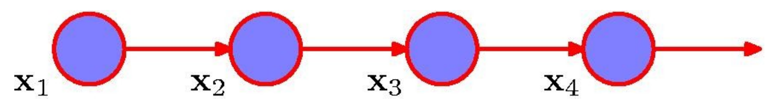
The model, i.e. $p(\mathbf{x}_n \mid \mathbf{x}_{n-1})$:



A sequence of observations:

The … by the n-1'th observation, i.e.

$$p(\mathbf{x}_n|\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}) = p(\mathbf{x}_n|\mathbf{x}_{n-1})$$



The chain of observations is a 1st-order Markov chain, and the probability of a sequence of N observations is

$$p(\mathbf{x}_1,\ldots,\mathbf{x}_N) = \prod_{n=1}^{N} p(\mathbf{x}_n|\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}) = p(\mathbf{x}_1)\prod_{n=2}^{N} p(\mathbf{x}_n|\mathbf{x}_{n-1})$$

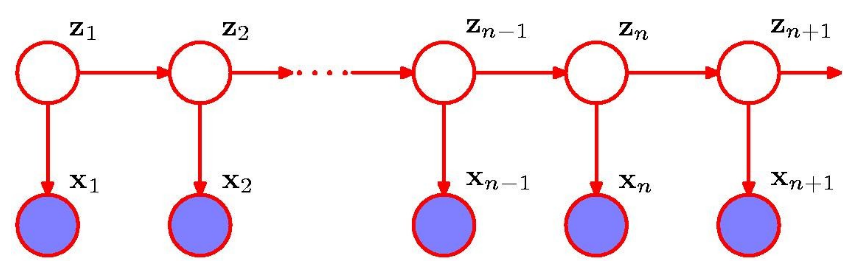
Hidden Markov Models

What if the *n*'th observation in a chain of observations is influenced by a corresponding hidden value/state?

Latent values

H H L L H

Observations

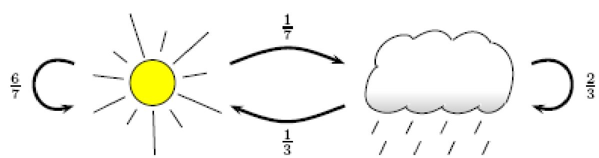


If the hidden values/states are discrete and form a Markov chain, then it is a hidden Markov model (HMM)

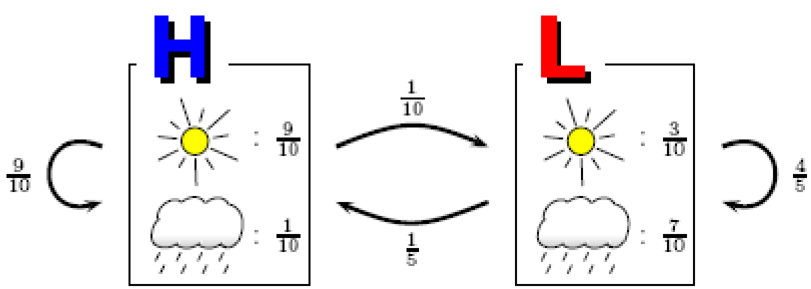
Hidden Markov Models

What if the *n*'th observation in a chain of observations is influenced by a corresponding hidden value/state?

Markov Model



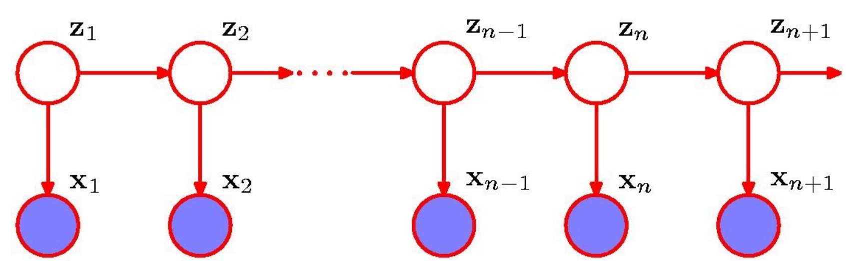
Hidden Markov Model



Latent values

H H L L H

Observations



If the hidden values/states are discrete and form a Markov chain, then it is a hidden Markov model (HMM)

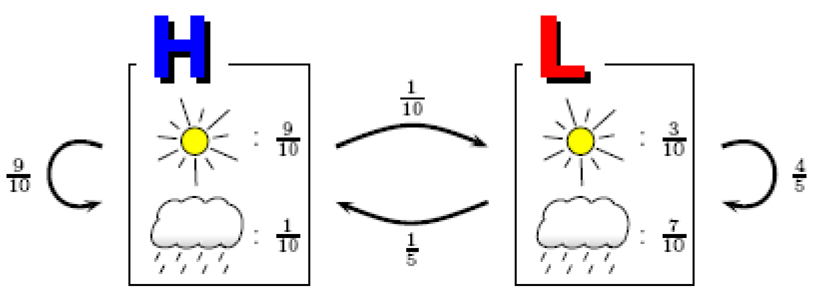
Hidden Markov Models

What if the *n*'th observation in a chain of observations is influenced

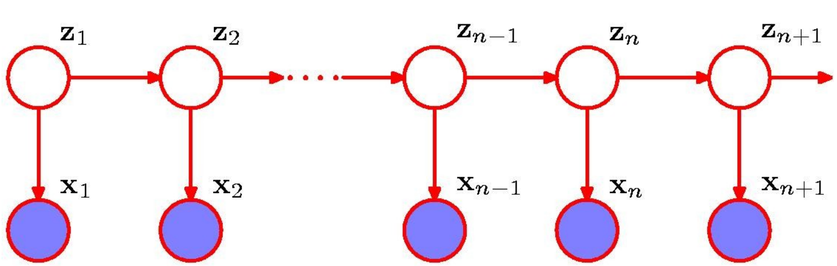
The joint distribution

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N, \mathbf{z}_1, \ldots, \mathbf{z}_N) = p(\mathbf{z}_1) \left[\prod_{n=2}^{N} p(\mathbf{z}_n | \mathbf{z}_{n-1}) \right] \prod_{n=1}^{N} p(\mathbf{x}_n | \mathbf{z}_n)$$

Hidden Markov Model



Observations



If the hidden values/states are discrete and form a Markov chain, then it is a hidden Markov model (HMM)

Hidden Markov Models

Emission probabilities

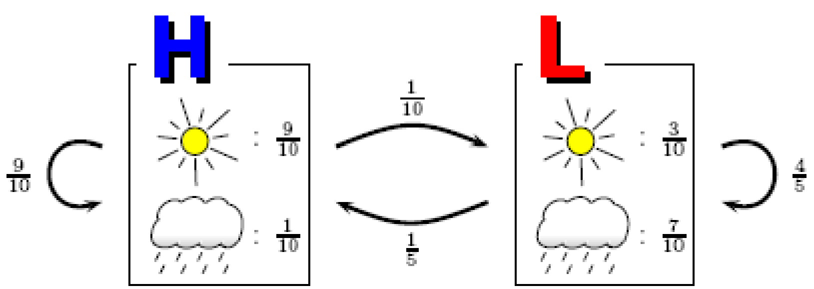
What if ... hain of c

Transition probabilities

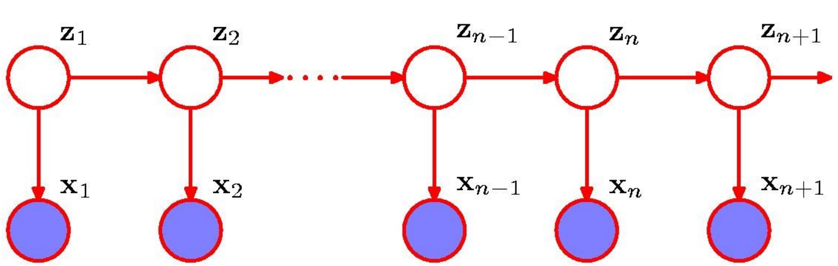
distribution

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N, \mathbf{z}_1, \ldots, \mathbf{z}_N) = p(\mathbf{z}_1) \left[\prod_{n=2}^{N} p(\mathbf{z}_n | \mathbf{z}_{n-1}) \right] \prod_{n=1}^{N} p(\mathbf{x}_n | \mathbf{z}_n)$$

Hidden Markov Model



Observations



If the hidden values/states are discrete and form a Markov chain, then it is a hidden Markov model (HMM)

Transition probabilities

Notation: In Bishop, the hidden states $\mathbf{z}_n$ are positional vectors, e.g. if $\mathbf{z}_n = (0,0,1)$ then the model in step n is in state $k=3$

Transition probabilities: If the hidden states are discrete with K states, the conditional distribution $p(\mathbf{z}_n \mid \mathbf{z}_{n-1})$ is a K x K table $\mathbf{A}$, and the marginal distribution $p(\mathbf{z}_1)$ describing the initial state is a K vector $\boldsymbol{\pi}$

The probability of going from state j to state k is:

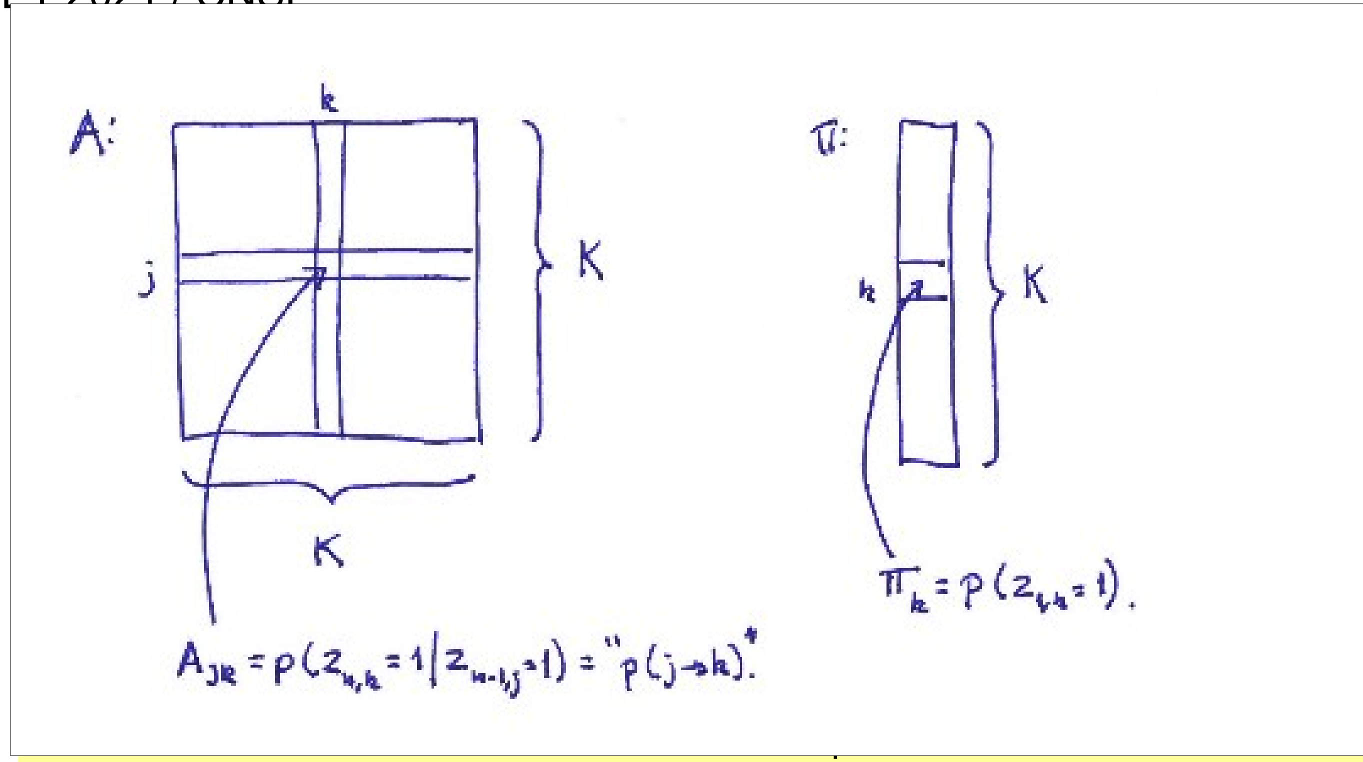
$$A_{jk} \equiv p(z_{nk} = 1 | z_{n-1,j} = 1)$$

$$\sum_k A_{jk} = 1$$

The probability of state k being the initial state is:

$$\pi_k \equiv p(z_{1k} = 1)$$

$$\sum_k \pi_k = 1$$



ities

ositional vectors, e.g.
e *k*=3

re discrete with *K*
a *K* x *K* table **A**, and
tial state is a *K*
vector **π**

The probability of going from state *j* to state *k* is:

$$A_{jk} \equiv p(z_{nk} = 1 | z_{n-1,j} = 1)$$

$$\sum_k A_{jk} = 1$$

The probability of state *k* being the initial state is:

$$\pi_k \equiv p(z_{1k} = 1)$$

$$\sum_k \pi_k = 1$$

Notat … s, e.g.

if $\mathbf{z}_n$ =

Trans … K

states … and

the m

vecto

The transition probabilities:

$$p(\mathbf{z}_n|\mathbf{z}_{n-1}, \mathbf{A}) = \prod_{k=1}^{K} \prod_{j=1}^{K} A_{jk}^{z_{n-1,j} z_{nk}}$$

$$p(\mathbf{z}_1|\pi) = \prod_{k=1}^{K} \pi_k^{z_{1k}}$$

The probability of going from state *j* to state *k* is:

$$A_{jk} \equiv p(z_{nk} = 1 | z_{n-1,j} = 1)$$

$$\sum_k A_{jk} = 1$$

The probability of state *k* being the initial state is:

$$\pi_k \equiv p(z_{1k} = 1)$$

$$\sum_k \pi_k = 1$$

This double product is $A_{j^*k^*}$ iff $z_{n-1,j^*} = 1$ and $z_{n,k^*} = 1$, because $A^1_{jk} = A_{jk}$ and $A^0_{jk} = 1$

The transition prob

Notat

if $\mathbf{z}_n$ =

, e.g.

Trans

states

the m

vecto

$$p(\mathbf{z}_n|\mathbf{z}_{n-1},\mathbf{A}) = \prod_{k=1}^{K}\prod_{j=1}^{K} A_{jk}^{z_{n-1,j}z_{nk}}$$

$$p(\mathbf{z}_1|\pi) = \prod_{k=1}^{K} \pi_k^{z_{1k}}$$

This product is similarly π_{k^*} iff $z_{1,k^*} = 1$

The probability of going from state *j* to state *k* is:

$$A_{jk} \equiv p(z_{nk}=1|z_{n-1,j}=1)$$

$$\sum_k A_{jk} = 1$$

The probability of state *k* being the initial state is:

$$\pi_k \equiv p(z_{1k}=1)$$

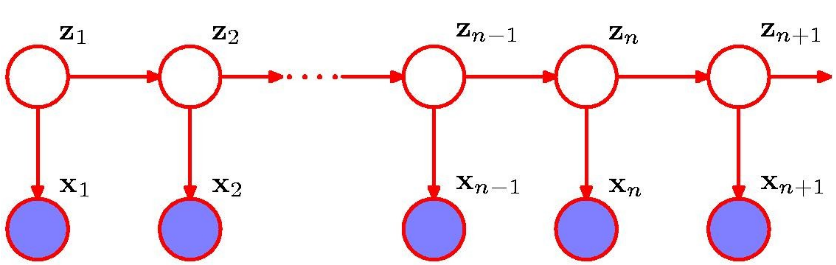
$$\sum_k \pi_k = 1$$

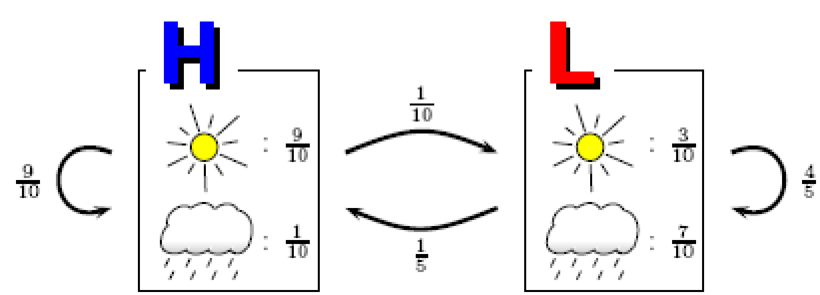
Emission probabilities

Emission probabilities: The conditional distributions of the observed variables $p(\mathbf{x}_n \mid \mathbf{z}_n)$ from a specific state

If the observed values $\mathbf{x}_n$ are discrete (e.g. D symbols), the emission probabilities $\boldsymbol{\Phi}$ is a KxD table of probabilities which for each of the K states specifies the probability of emitting each observable ...

$$p(\mathbf{x}_n|\mathbf{z}_n, \phi) = \prod_{k=1}^{K} p(\mathbf{x}_n|\phi_k)^{z_{nk}}$$

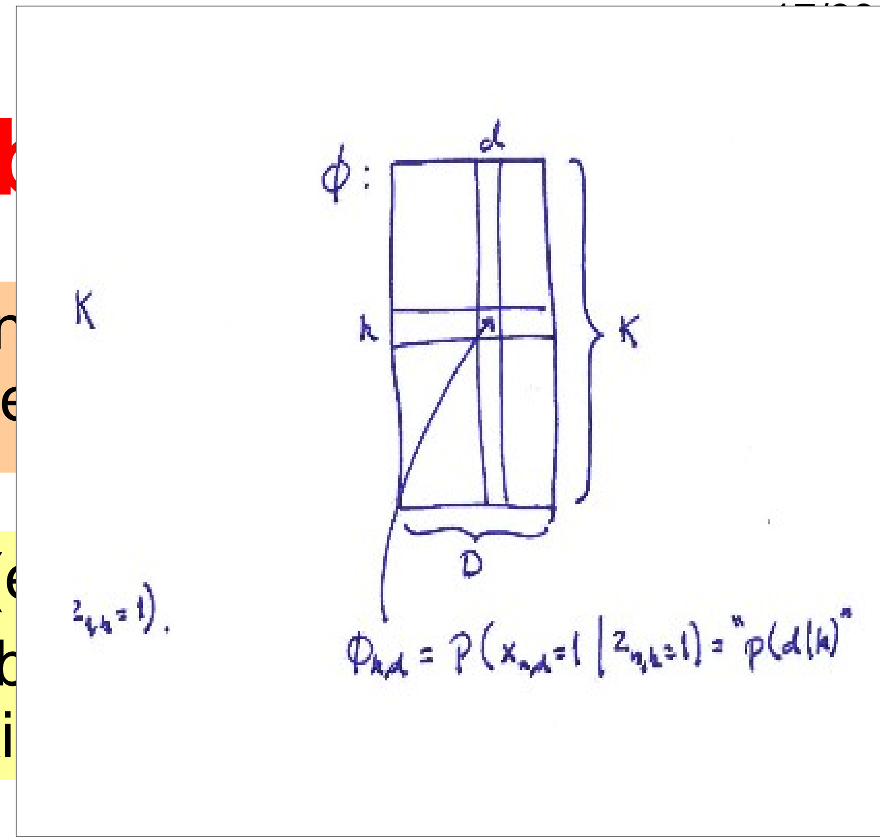




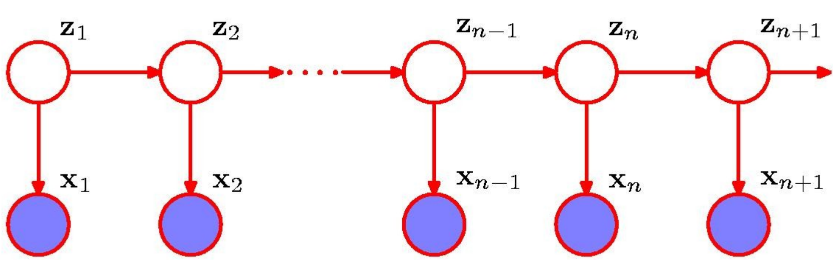
Emission prob

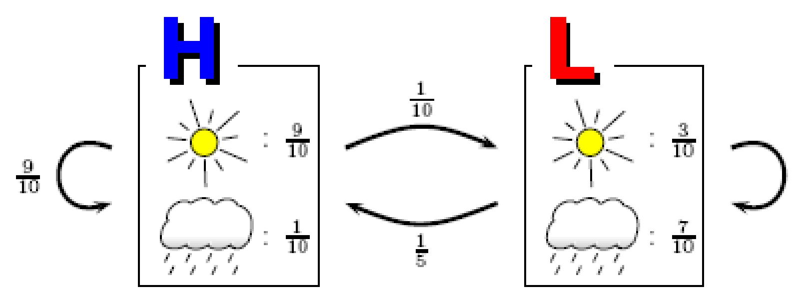
Emission probabilities: The condition
observed variables $p(\mathbf{x}_n \mid \mathbf{z}_n)$ from a spe

If the observed values $\mathbf{x}_n$ are discrete (e
probabilities $\boldsymbol{\Phi}$ is a *K*x*D* table of probab
states specifies the probability of emitti



$$p(\mathbf{x}_n|\mathbf{z}_n, \phi) = \prod_{k=1}^{K} p(\mathbf{x}_n|\phi_k)^{z_{nk}}$$





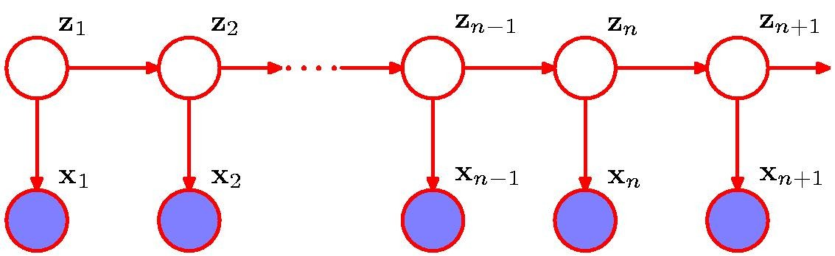
HMM joint probability distribution

$$p(\mathbf{X}, \mathbf{Z}|\mathbf{\Theta}) = p(\mathbf{z}_1|\pi)\left[\prod_{n=2}^{N} p(\mathbf{z}_n|\mathbf{z}_{n-1}, \mathbf{A})\right]\prod_{n=1}^{N} p(\mathbf{x}_n|\mathbf{z}_n, \phi)$$

Observables: $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$

Latent states: $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$

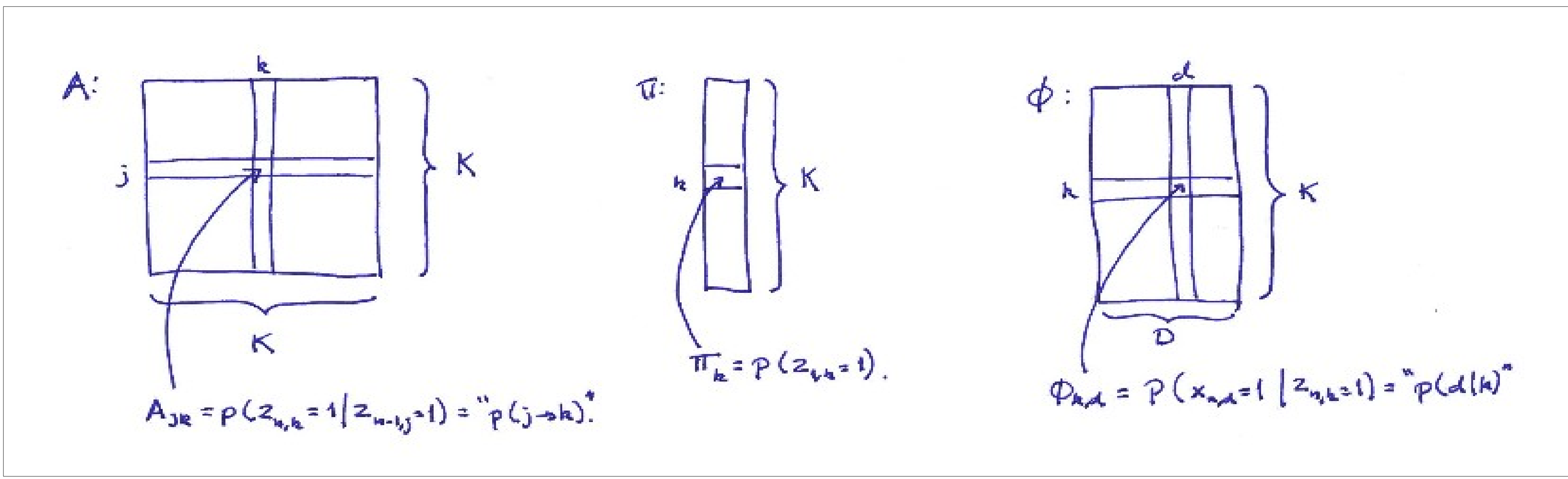
Model parameters: $\mathbf{\Theta} = \{\pi, \mathbf{A}, \phi\}$



If A and ***Φ*** are the same for all *n* then the HMM is *homogeneous*

HMM joint probability distribution

$$p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\Theta}) = p(\mathbf{z}_1 | \pi) \left[\prod_{n=2}^{N} p(\mathbf{z}_n | \mathbf{z}_{n-1}, \mathbf{A}) \right] \prod_{n=1}^{N} p(\mathbf{x}_n | \mathbf{z}_n, \phi)$$

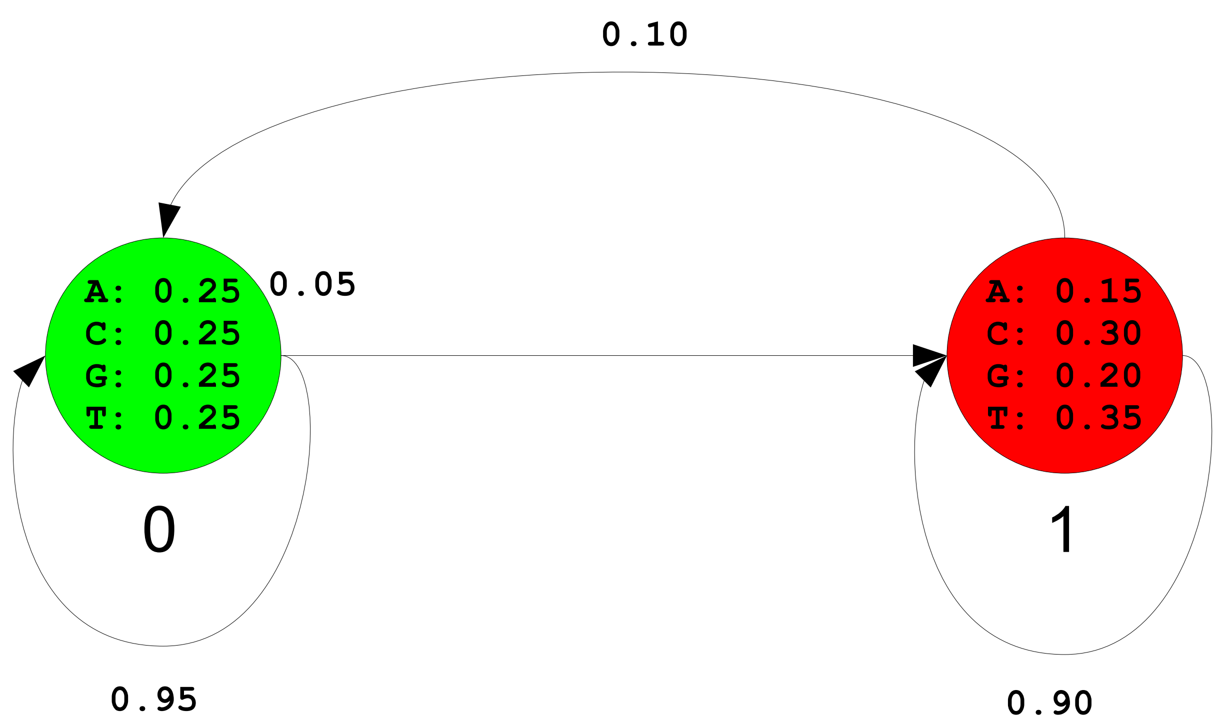


If A and ***Φ*** are the same for all *n* then the HMM is *homogeneous*

Example – 2-state HMM

Observable: {A, C, G, T}, States: {0,1}

$$A \quad \begin{matrix} 0.95 & 0.05 \\ 0.10 & 0.90 \end{matrix} \qquad \pi \quad \begin{matrix} 1.00 \\ 0.00 \end{matrix} \qquad \varphi \quad \begin{matrix} 0.25 & 0.25 & 0.25 & 0.25 \\ 0.15 & 0.30 & 0.20 & 0.35 \end{matrix}$$



Example – 7-state HMM

Observable: {A, C, G, T}, States: {0,1, 2, 3, 4, 5, 6}

A

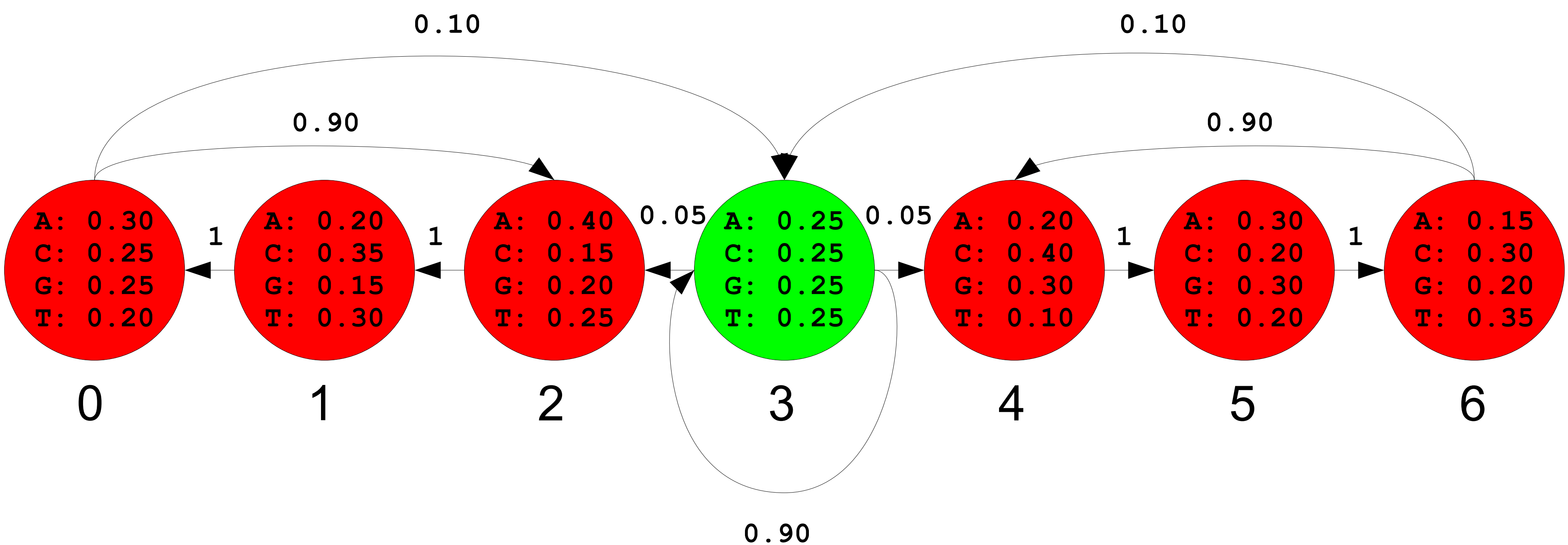
0.00	0.00	0.90	0.10	0.00	0.00	0.00
1.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	1.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.05	0.90	0.05	0.00	0.00
0.00	0.00	0.00	0.00	0.00	1.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	1.00
0.00	0.00	0.00	0.10	0.90	0.00	0.00

π

0.00
0.00
0.00
1.00
0.00
0.00
0.00

φ

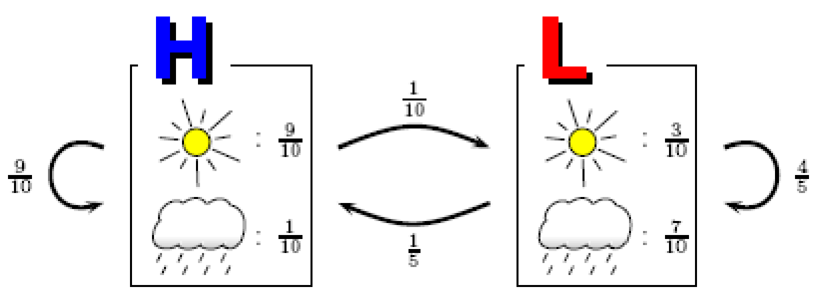
0.30	0.25	0.25	0.20
0.20	0.35	0.15	0.30
0.40	0.15	0.20	0.25
0.25	0.25	0.25	0.25
0.20	0.40	0.30	0.10
0.30	0.20	0.30	0.20
0.15	0.30	0.20	0.35



HMMs as a generative model

A HMM ***generates a sequence of observables*** by moving from latent state to latent state according to the transition probabilities and ***emitting an observable*** (from a discrete set of observables, i.e. a finite alphabet) from each latent state visited ***according to the emission probabilities*** of the state ...

Model *M*:



A run follows a sequence of states:

H H L L H

And emits a sequence of symbols:

Computing P(X,Z)

$$p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\Theta}) = p(\mathbf{z}_1 | \pi) \left[\prod_{n=2}^{N} p(\mathbf{z}_n | \mathbf{z}_{n-1}, \mathbf{A}) \right] \prod_{n=1}^{N} p(\mathbf{x}_n | \mathbf{z}_n, \phi)$$

```
def joint_prob(x, z):
       """
       Returns the joint probability of x and z
       """
       p = init_prob[z[0]] * emit_prob[z[0]][x[0]]
       for i in range(1, len(x)):
           p = p * trans_prob[z[i-1]][z[i]] * emit_prob[z[i]][x[i]]
       return p
```

Computing P(X,Z)

$$p(\mathbf{X}, \mathbf{Z}|\mathbf{\Theta}) = p(\mathbf{z}_1|\pi) \left[\prod_{n=2}^{N} p(\mathbf{z}_n|\mathbf{z}_{n-1}, \mathbf{A}) \right] \prod_{n=1}^{N} p(\mathbf{x}_n|\mathbf{z}_n, \phi)$$

```
def jo
```

```
[x[i]]
```

```
$ python hmm_jointprob.py hmm-7-state.txt test_seq100.txt
> seq100
p(x,z) =  1.8619524290102162e-65

$ python hmm_jointprob.py hmm-7-state.txt test_seq200.txt
> seq200
p(x,z) =  1.6175774997005771e-122

$ python hmm_jointprob.py hmm-7-state.txt test_seq300.txt
> seq300
p(x,z) =  3.0675430597843052e-183

$ python hmm_jointprob.py hmm-7-state.txt test_seq400.txt
> seq400
p(x,z) =  4.860704144302979e-247

$ python hmm_jointprob.py hmm-7-state.txt test_seq500.txt
> seq500
p(x,z) =  5.258724342206735e-306

$ python hmm_jointprob.py hmm-7-state.txt test_seq600.txt
> seq600
p(x,z) =  0.0
```

Computing P(X,Z)

$$p(\mathbf{X},\mathbf{Z}|\mathbf{\Theta}) = p(\mathbf{z}_1|\pi)\left[\prod_{n=2}^{N} p(\mathbf{z}_n|\mathbf{z}_{n-1},\mathbf{A})\right]\prod_{n=1}^{N} p(\mathbf{x}_n|\mathbf{z}_n,\phi)$$

```
def jo
```

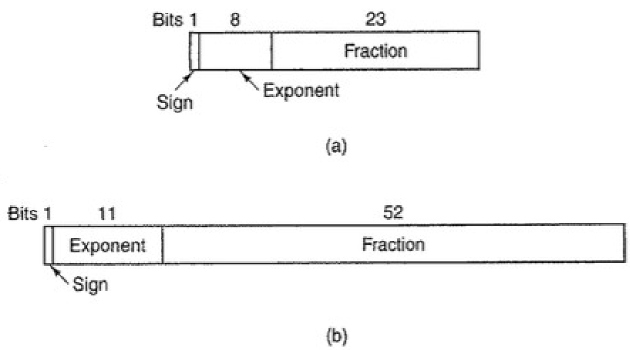
```
[x[i]]
```

```
$ python hmm_jointprob.py hmm-7-state.txt test_seq100.txt
> seq100
p(x,z) =  1.8619524290102162e-65

$ python hmm_jointprob.py hmm-7-state.txt test_seq200.txt
> seq200
p(x,z) =  1.6175774997005771e-122

$ python hmm_jointprob.py hmm-7-state.txt test_seq300.txt
> seq300
p(x,z) =  3.0675430597843052e-183

$ python hmm_jointprob.py hmm-7-state.txt test_seq400.txt
> seq400
p(x,z) =  4.860704144302979e-247

$ python hmm_jointprob.py hmm-7-state.txt test_seq500.txt
> seq500
p(x,z) =  5.258724342206735e-306

$ python hmm_jointprob.py hmm-7-state.txt test_seq600.txt
> seq600
p(x,z) =  0.0
```

Should be >0 by construction of **X** and **Z**

Representing numbers

A floating point number *n* is represented as $n = f * 2^e$ cf. the IEEE-754 standard which specify the range of *f* and *e*



Item	Single precision	Double precision
Bits in sign	1	1
Bits in exponent	8	11
Bits in fraction	23	52
Bits, total	32	64
Exponent system	Excess 127	Excess 1023
Exponent range	−126 to +127	−1022 to +1023
Smallest normalized number	2^{-126}	2^{-1022}
Largest normalized number	approx. 2^{128}	approx. 2^{1024}
Decimal range	approx. 10^{-38} to 10^{38}	approx. 10^{-308} to 10^{308}
Smallest denormalized number	approx. 10^{-45}	approx. 10^{-324}

Figure B-5. Characteristics of IEEE floating-point numbers.

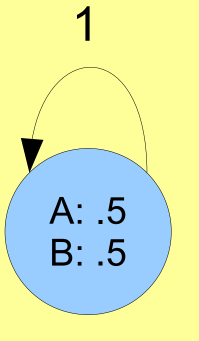
See e.g. Appendix B in Tanenbaum's Structured Computer Organization for further details.

The problem – Too small numbers

For the simple HMM, the joint-probability $p(\mathbf{X},\mathbf{Z})$ is

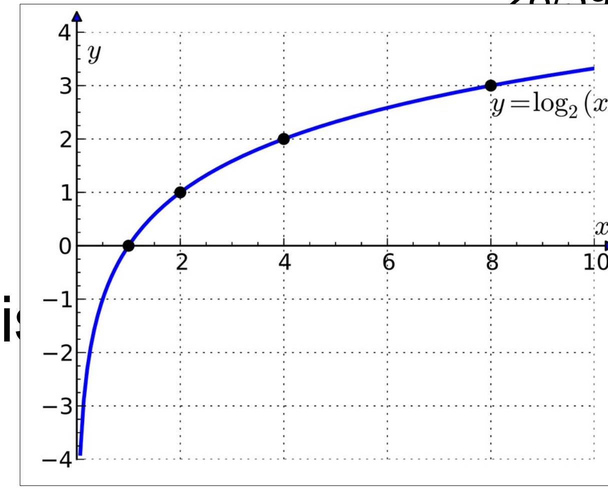
$$p(\mathbf{X},\mathbf{Z}) = 1 \cdot \prod_{n=2}^{N} 1 \cdot \prod_{n=1}^{N} \frac{1}{2} = \left(\frac{1}{2}\right)^{N} = 2^{-N}$$

If $N > 467$ then 2^{-N} is smaller than 10^{-324}, i.e. cannot be represented



A simple HMM

The problem – Too small



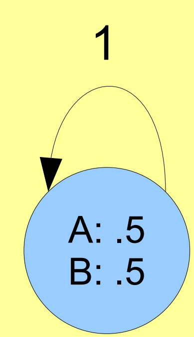
For the simple HMM, the joint-probability $p(\mathbf{X},\mathbf{Z})$ i

$$p(\mathbf{X},\mathbf{Z}) = 1 \cdot \prod_{n=2}^{N} 1 \cdot \prod_{n=1}^{N} \frac{1}{2} = \left(\frac{1}{2}\right)^{N} = 2^{-N}$$

If $N > 467$ then 2^{-N} is smaller than 10^{-324}, i.e. cannot be represented

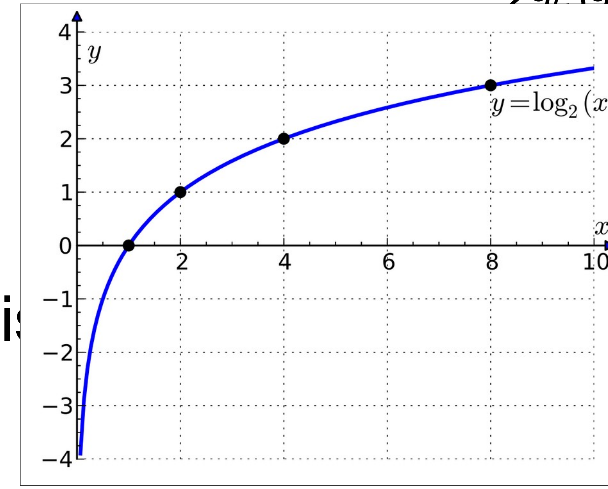
Does this mean that we cannot compute $p(\mathbf{X},\mathbf{Z})$ for slightly long sequence?

No, we use log-transform. Instead of computing $p(\mathbf{X},\mathbf{Z})$, we compute $\log p(\mathbf{X},\mathbf{Z})$



A simple HMM

The problem – Too small



For the simple HMM, the joint-probability *p*(**X**,**Z**) i

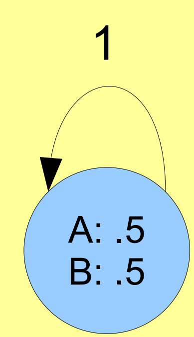
$$p(\mathbf{X},\mathbf{Z}) = 1 \cdot \prod_{n=2}^{N} 1 \cdot \prod_{n=1}^{N} \frac{1}{2} = \left(\frac{1}{2}\right)^{N} = 2^{-N}$$

If $N > 467$ then 2^{-N} is smaller than 10^{-324}, i.e. cannot be represented

No problem representing

log *p*(**X**,**Z**) = -*N*

as the decimal range is approx -10^{308} to 10^{308}



A simple HMM

Solution: Compute log P(X,Z)

$$p(\mathbf{X},\mathbf{Z}|\boldsymbol{\Theta}) = p(\mathbf{z}_1|\pi)\left[\prod_{n=2}^{N} p(\mathbf{z}_n|\mathbf{z}_{n-1},\mathbf{A})\right]\prod_{n=1}^{N} p(\mathbf{x}_n|\mathbf{z}_n,\phi)$$

Use log (XY) = log X + log Y, and define log 0 to be -inf

$$\log p(\mathbf{X},\mathbf{Z}|\boldsymbol{\Theta}) = \log p(\mathbf{z}_1|\pi) + \sum_{n=2}^{N} \log p(\mathbf{z}_n|\mathbf{z}_{n-1},\mathbf{A}) + \sum_{n=1}^{N} \log p(\mathbf{x}_n|\mathbf{z}_n,\phi)$$

Solution: Compute log P(X,Z)

$$\log p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\Theta}) = \log p(\mathbf{z}_1 | \pi) + \sum_{n=2}^{N} \log p(\mathbf{z}_n | \mathbf{z}_{n-1}, \mathbf{A}) + \sum_{n=1}^{N} \log p(\mathbf{x}_n | \mathbf{z}_n, \phi)$$

```
def log_joint_prob(self, x, z):
    """
    Returns the log transformed joint probability of x and z
    """
    logp = log(init_prob[z[0]]) + log(emit_prob[z[0]][x[0]])
    for i in range(1, len(x)):
        logp = logp + log(trans_prob[z[i-1]][z[i]]) + log(emit_prob[z[i]][x[i]])
    return logp
```

Solution: Compute log P(X,Z)

$$\log p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\Theta}) = \log p(\mathbf{z}_1 | \pi) + \sum_{n=2}^{N} \log p(\mathbf{z}_n | \mathbf{z}_{n-1}, \mathbf{A}) + \sum_{n=1}^{N} \log p(\mathbf{x}_n | \mathbf{z}_n, \phi)$$

```
def log_joint
    """
    Returns t
    """
    logp = lo
    for i in
        logp                                        [z[i]][x[i]])
    return lo
```

```
$ python hmm_log_jointprob.py hmm-7-state.txt test_seq100.txt
> seq100
log p(x,z) =  -149.04640541441395

$ python hmm_log_jointprob.py hmm-7-state.txt test_seq200.txt
> seq200
log p(x,z) =  -280.43445168576596

$ python hmm_log_jointprob.py hmm-7-state.txt test_seq300.txt
> seq300
log p(x,z) =  -420.25219508298494

$ python hmm_log_jointprob.py hmm-7-state.txt test_seq400.txt
> seq400
log p(x,z) =  -567.1573346564519

$ python hmm_log_jointprob.py hmm-7-state.txt test_seq500.txt
> seq500
log p(x,z) =  -702.9311499793356

$ python hmm_log_jointprob.py hmm-7-state.txt test_seq600.txt
> seq600
log p(x,z) =  -842.0056730984585
```

Using HMMs

- Determine the likelihood of a sequence of observations.
- Find a plausible underlying explanation (or decoding) of a sequence of observations.

Using HMMs

- Determine the likelihood of a sequence of observations.
- Find a plausible underlying explanation (or decoding) of a sequence of observations.

$$p(\mathbf{X}|\mathbf{\Theta}) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\mathbf{\Theta})$$

Using HMMs

- Determine the likelihood of a sequence of observations.
- Find a plausible underlying explanation (or decoding) of a sequence of observations.

$$p(\mathbf{X}|\mathbf{\Theta}) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\mathbf{\Theta})$$

The sum has K^N terms, but it turns out that it can be computed in O(K^2N) time, but first we will consider **decoding**

Decoding using HMMs

Given a HMM $\mathbf{\Theta}$ and a sequence of observations $\mathbf{X} = \mathbf{x}_1,\ldots,\mathbf{x}_N$, find a plausible explanation, i.e. a sequence $\mathbf{Z}^* = \mathbf{z}^*_1,\ldots,\mathbf{z}^*_N$ of values of the hidden variable.

Decoding using HMMs

Given a HMM $\mathbf{\Theta}$ and a sequence of observations $\mathbf{X} = \mathbf{x}_1,\ldots,\mathbf{x}_N$, find a plausible explanation, i.e. a sequence $\mathbf{Z}^* = \mathbf{z}^*_1,\ldots,\mathbf{z}^*_N$ of values of the hidden variable.

Viterbi decoding

$\mathbf{Z}^*$ is the overall most likely explanation of $\mathbf{X}$:

$$\mathbf{Z}^* = \arg\max_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z} | \mathbf{\Theta})$$

Decoding using HMMs

Given a HMM $\mathbf{\Theta}$ and a sequence of observations $\mathbf{X} = \mathbf{x}_1,\ldots,\mathbf{x}_N$, find a plausible explanation, i.e. a sequence $\mathbf{Z}^* = \mathbf{z}^*_1,\ldots,\mathbf{z}^*_N$ of values of the hidden variable.

Viterbi decoding

$\mathbf{Z}^*$ is the overall most likely explanation of $\mathbf{X}$:

$$\mathbf{Z}^* = \arg\max_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z} | \mathbf{\Theta})$$

Posterior decoding

$\mathbf{z}^*_n$ is the most likely state to be in the *n*'th step:

$$\mathbf{z}^*_n = \arg\max_{\mathbf{z}_n} p(\mathbf{z}_n | \mathbf{x}_1, \ldots, \mathbf{x}_N)$$

Summary

- Terminology of hidden Markov models (**HMMs**)
- **Viterbi-** and **Posterior decoding** for finding a plausible underlying explanation (sequence of hidden states) of a sequence of observation
- **Next**: Algorithms for computing the Viterbi and Posterior decodings efficiently