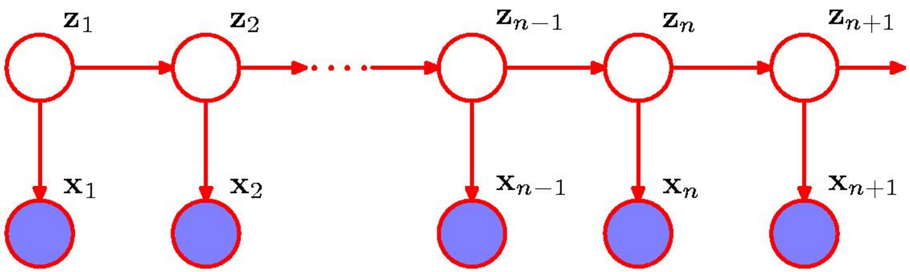
Hidden Markov Models

Terminology, Representation and Basic Problems



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 ...

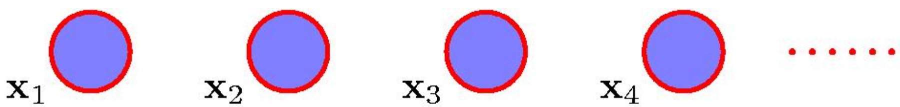
$$\mathbf{X} = \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$$

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



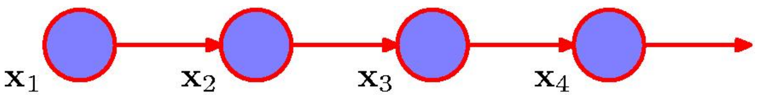
$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \prod_{n=1}^{N} p(\mathbf{x}_n)$$

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.

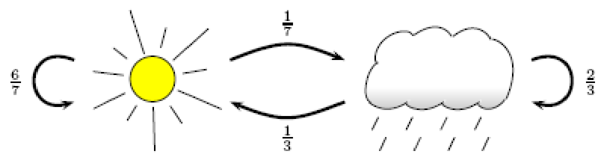
$$p(\mathbf{x}_n|\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}) = p(\mathbf{x}_n|\mathbf{x}_{n-1})$$



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

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

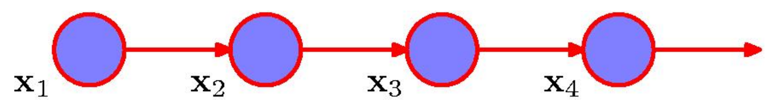
The model, i.e. $p(\mathbf{x}_n \mid \mathbf{x}_{n-1})$:



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

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

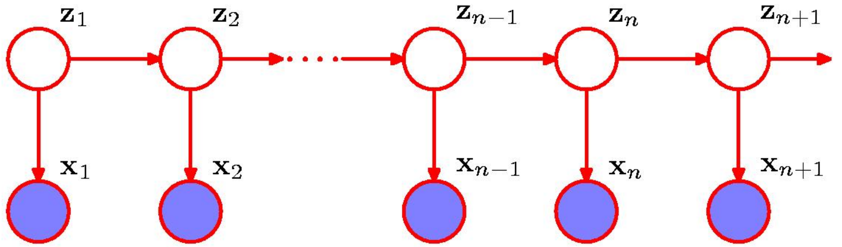
Hidden Markov Models

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

Latent values

H H L L H

Observations



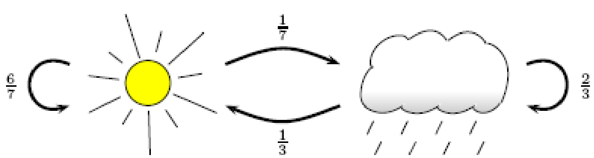
If the hidden variables are discrete and form a Markov chain, then it is a hidden Markov model (HMM)

Hidden Markov Models

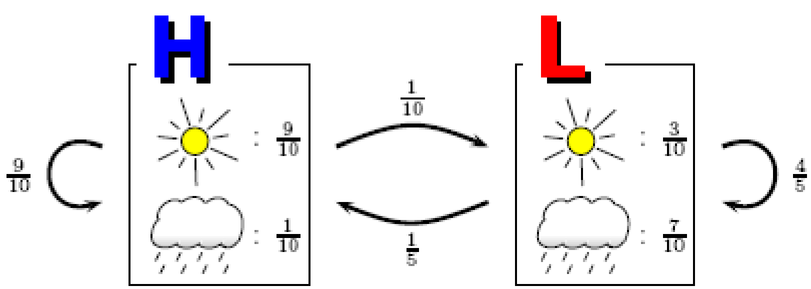
What if the *n*'th observation in a chain of observations is influenced by a corresponding hidden variable?

Latent values

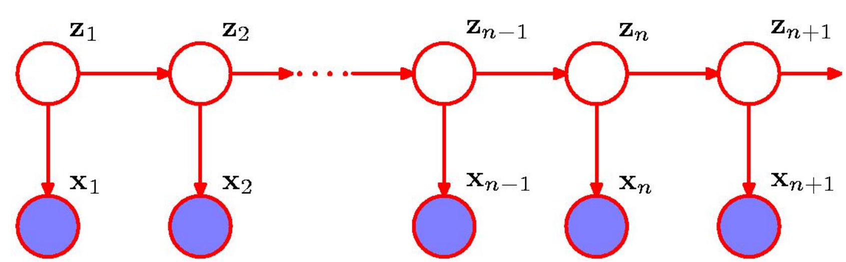
Markov Model



Hidden Markov Model



Observations



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

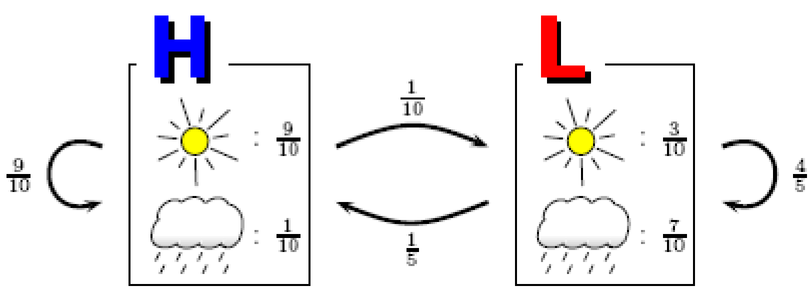
Hidden Markov Models

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

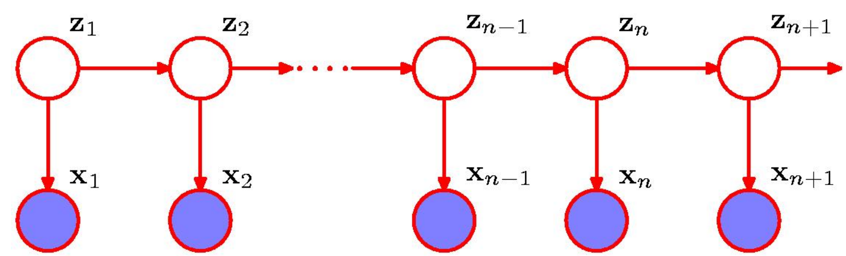
The joint distribution

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

Hidden Markov Model



Observations



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

Hidden Markov Models

Emission probabilities

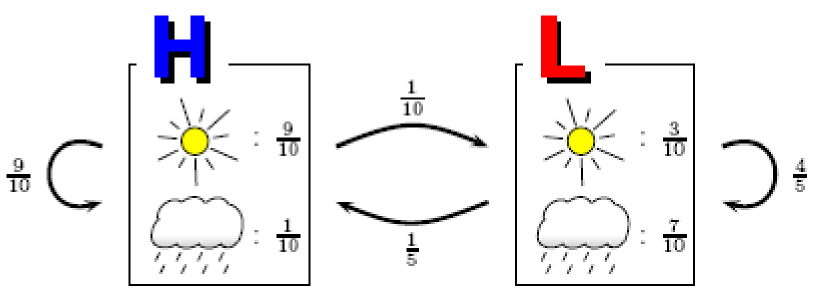
What if ... hain of c

Transition probabilities

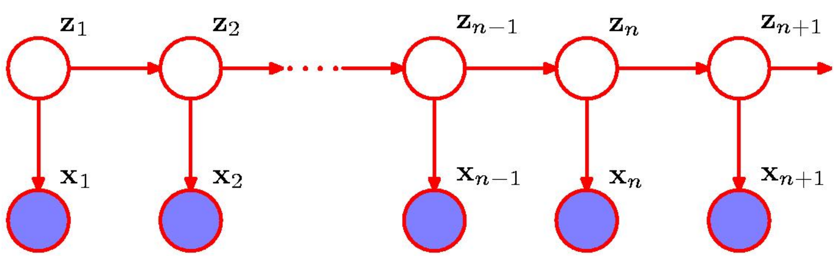
distribution

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

Hidden Markov Model



Observations



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

Transition probabilities

Notation: In Bishop, the hidden variables $\mathbf{z}_n$ 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 variables are discrete with K states, the conditional distribution $p(\mathbf{z}_n \mid \mathbf{z}_{n-1})$ is a K x K table **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:

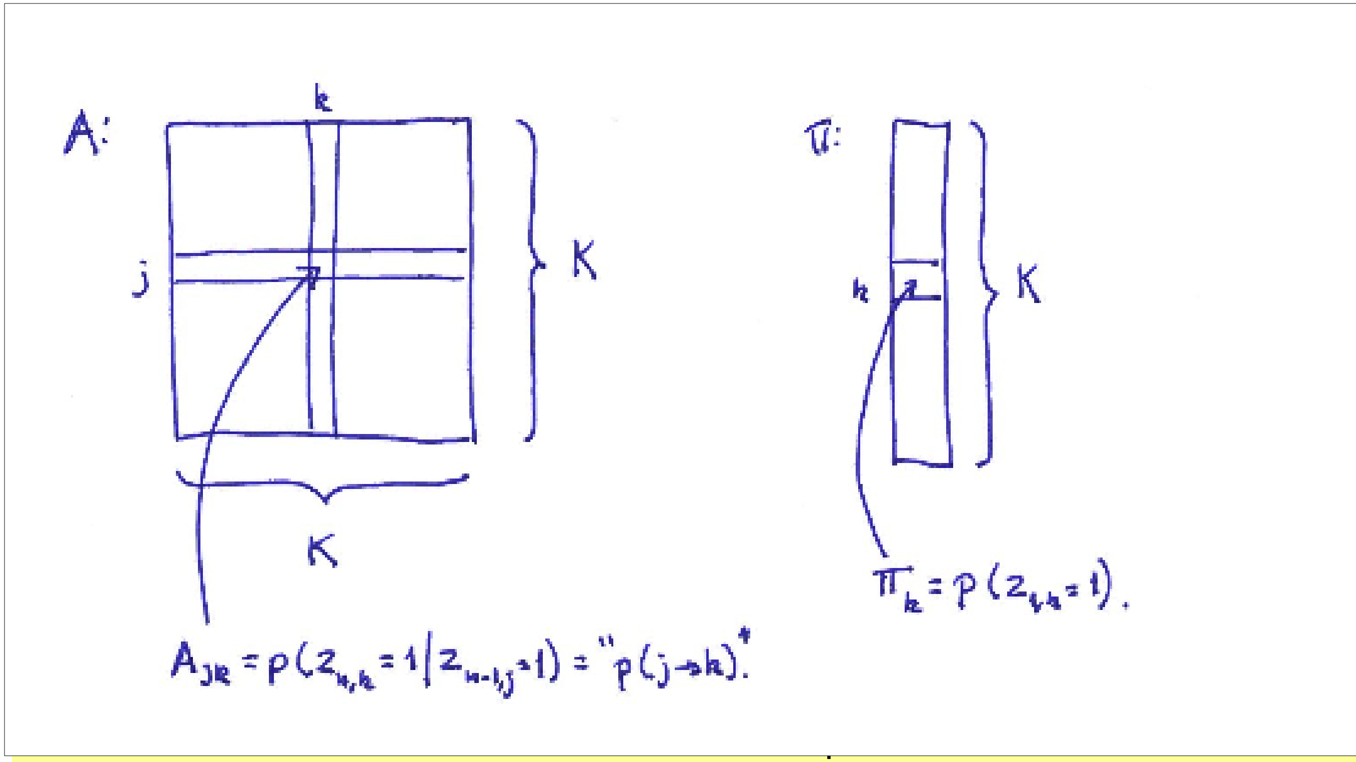
$$A_{jk} \equiv p(z_{nk} = 1 | z_{n-1,j} = 1)$$

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

The probability of state k being the initial state is:

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

$$\sum_k \pi_k = 1$$



ities

e positional vectors,
state k=3 ...

s are 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 tors,

e.g. if

Trans with *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:

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

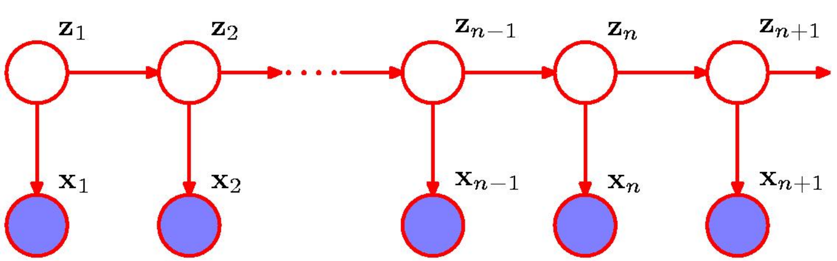
$$\sum_k \pi_k = 1$$

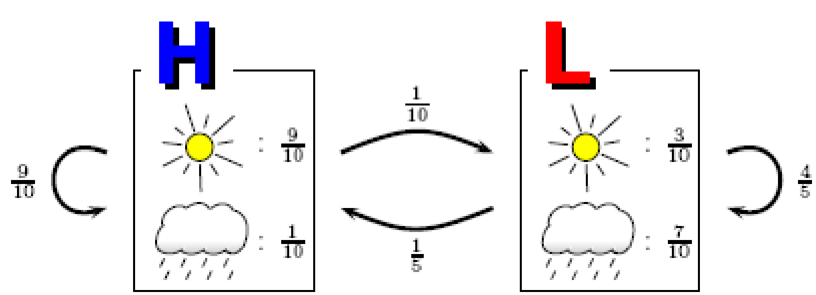
Emission probabilities

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 ...

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

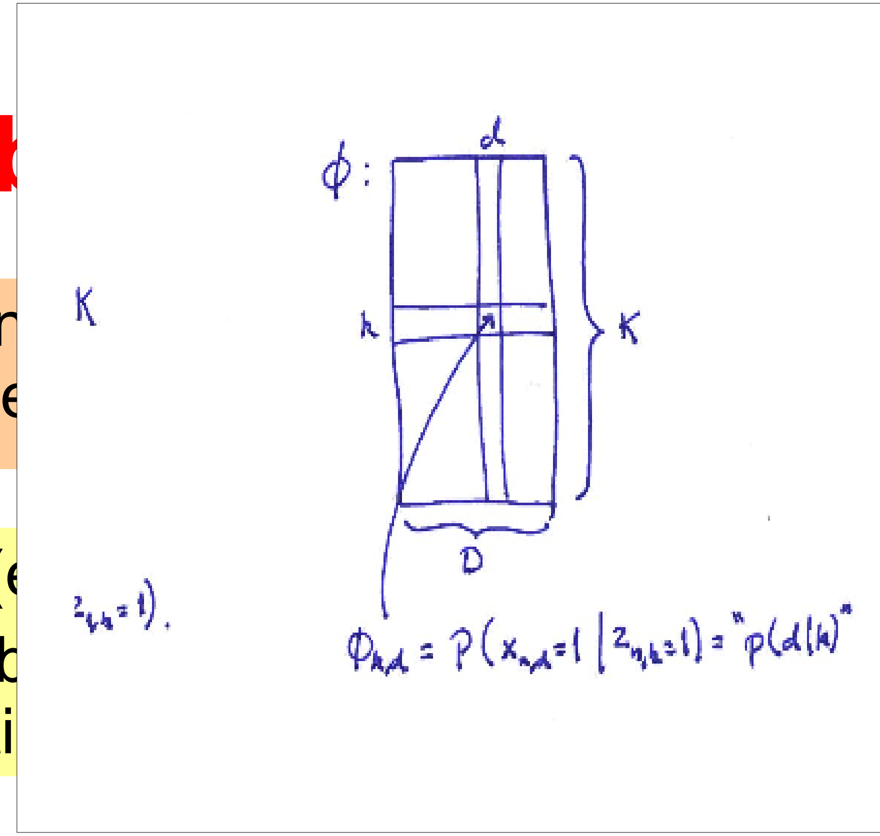




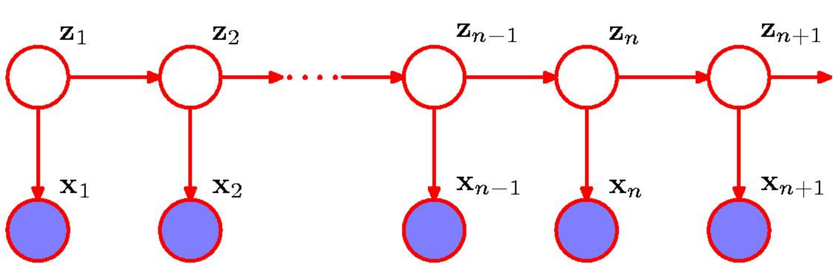
Emission prob

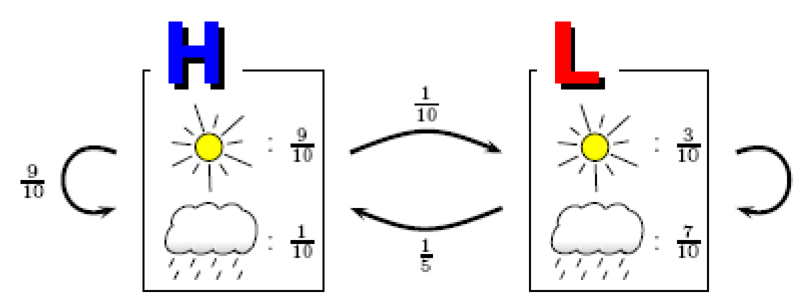
Emission probabilities: The condition
observed variables $p(\mathbf{x}_n \mid \mathbf{z}_n)$ from a spe

If the observed values $\mathbf{x}_n$ are discrete (
probabilities $\boldsymbol{\Phi}$ is a KxD 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:

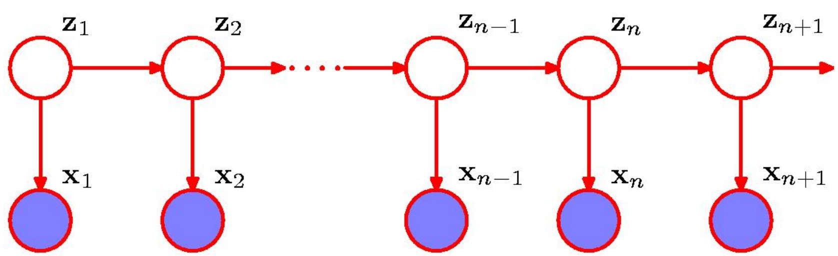
$$\mathbf{X} = \{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$$

Latent states:

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

Model parameters:

$$\mathbf{\Theta} = \{\pi,\mathbf{A},\phi\}$$

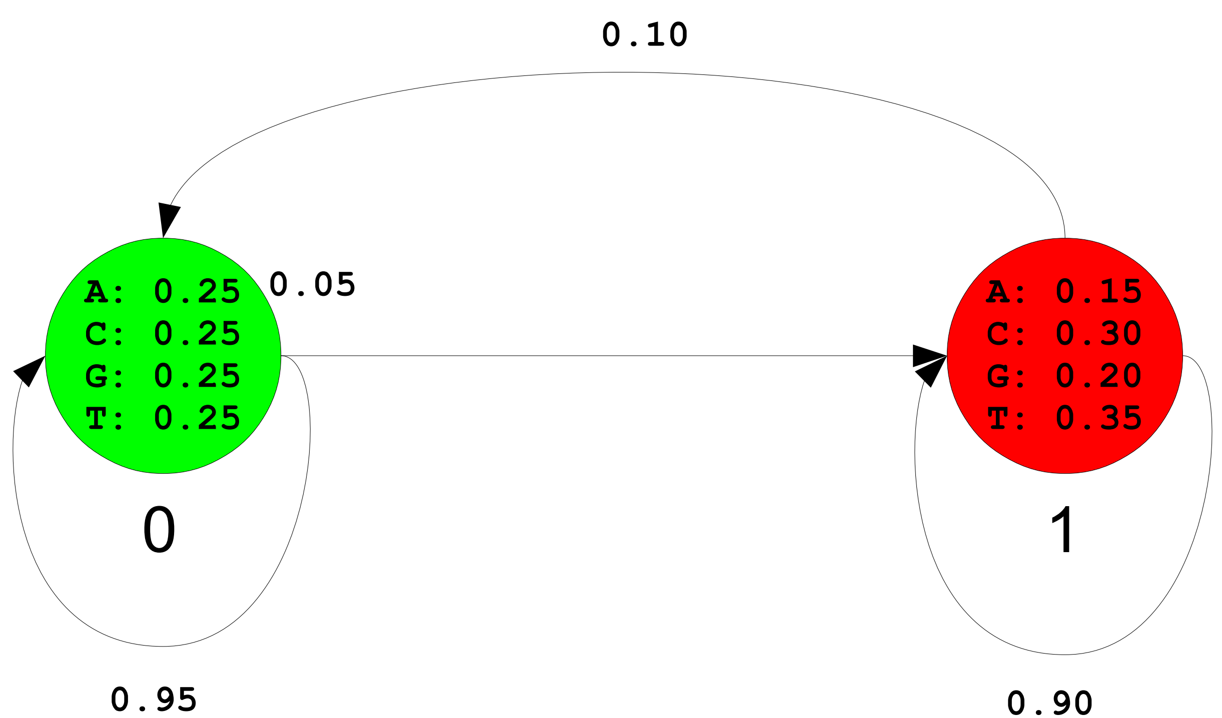


If A and ***Φ*** are the same for all *n* then the HMM is *homogeneous*

Example – 2-state HMM

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

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



Example – 7-state HMM

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

A

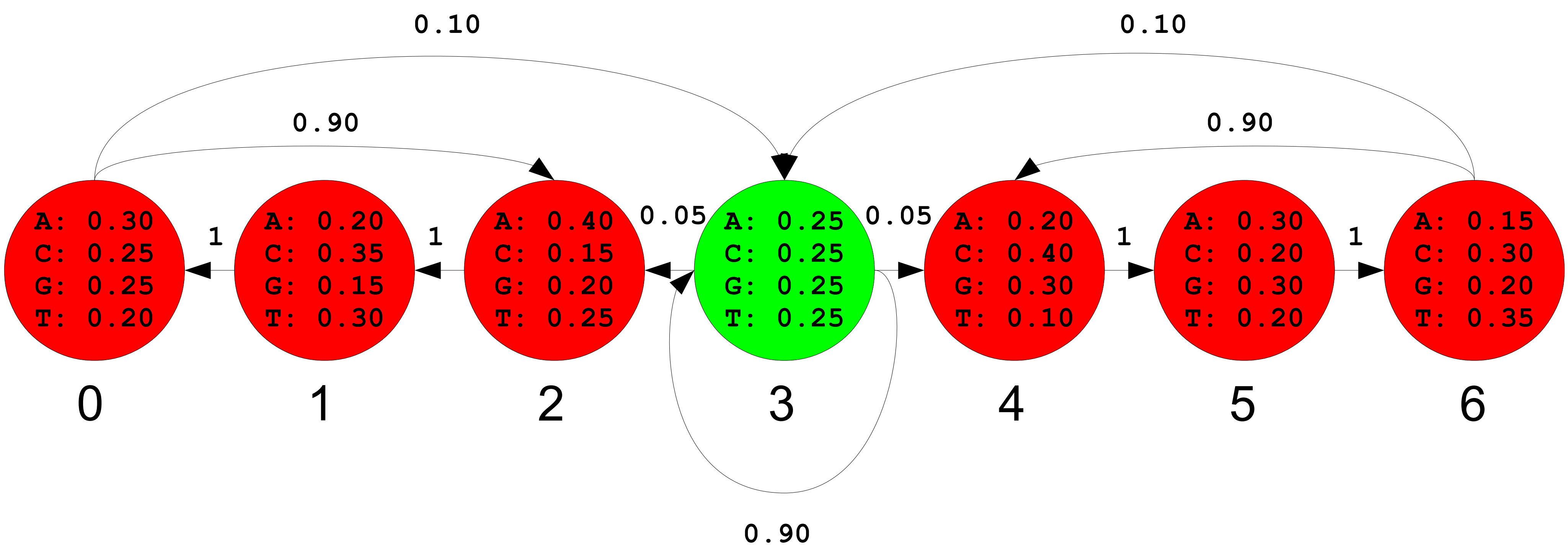
0.00	0.00	0.90	0.10	0.00	0.00	0.00
1.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	1.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.05	0.90	0.05	0.00	0.00
0.00	0.00	0.00	0.00	0.00	1.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	1.00
0.00	0.00	0.00	0.10	0.90	0.00	0.00

π

0.00
0.00
0.00
1.00
0.00
0.00
0.00

φ

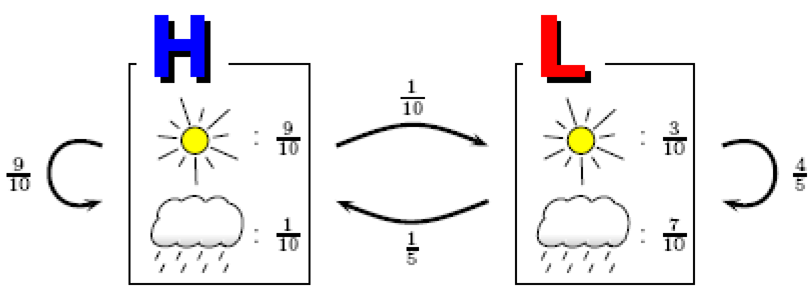
0.30	0.25	0.25	0.20
0.20	0.35	0.15	0.30
0.40	0.15	0.20	0.25
0.25	0.25	0.25	0.25
0.20	0.40	0.30	0.10
0.30	0.20	0.30	0.20
0.15	0.30	0.20	0.35



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)

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

```
def 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)

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

```
$ 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
```

```
def jo
[x[i]]
```

Computing P(X,Z)

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

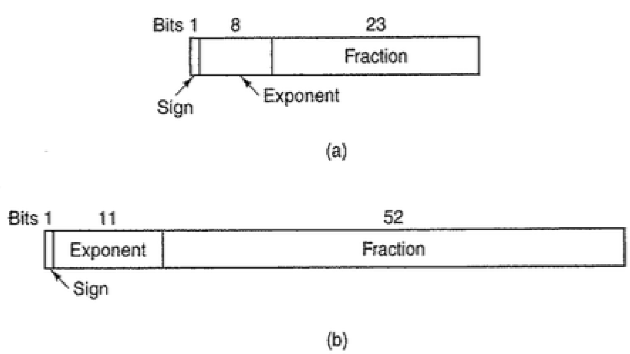
```
def jo
                                                         [x[i]]
```

```
$ 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.

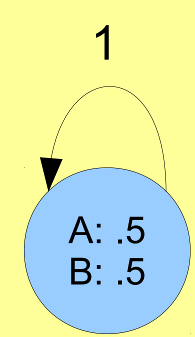
See e.g. Appendix B in Tanenbaum's Structured Computer Organization for further details.

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 numbers

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

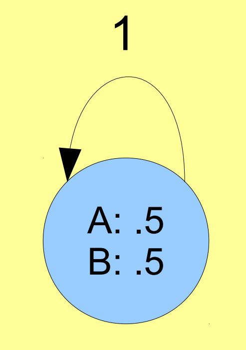
$$p(\mathbf{X},\mathbf{Z}) = 1 \cdot \prod_{n=2}^{N} 1 \cdot \prod_{n=1}^{N} \frac{1}{2} = \left(\frac{1}{2}\right)^{n} = 2^{-n}$$

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

No problem representing

$\log p(\mathbf{X},\mathbf{Z}) = -n$

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



A simple HMM

Solution: Compute log P(X,Z)

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

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

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

Solution: Compute log P(X,Z)

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

```
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.

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

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 $\boldsymbol{\Theta}$ 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

Z* is the overall most likely explanation of **X**:

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

Decoding using HMMs

Given a HMM $\boldsymbol{\Theta}$ 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 $\mathbf{X}$:

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

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