Markov Chains

CPSC 440/550: Advanced Machine Learning

cs.ubc.ca/~dsuth/440/24w2

University of British Columbia, on unceded Musqueam land

2024-25 Winter Term 2 (Jan–Apr 2025)

Last time

- VAEs
  - Deep latent variable model, e.g. $Z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $X \mid (Z = z) \sim \mathcal{N}(g_\theta(z), \sigma^2 \mathbf{I})$
  - Approximate (amortized) inference: $q_\phi(z \mid x) = \mathcal{N}(z; \boldsymbol{\mu}_\phi(x), \boldsymbol{\Sigma}_\phi(x))$
  - Maximizing ELBO encourages q to be consistent with p and p to maximize likelihood
- Transposed convolutions
  - Convolution-type layer that increases dimension
  - Useful for "learned upscaling" to do pixel-level labeling (fully-conv. nets)
  - Useful for decoder architecture in a VAE (latent $\to$ image)
- Representation learning
  - Autoencoders: encode to a low-dim latent, decode, try to match
  - Goal of representation learning: find a "useful" representation
    - Maybe: interpretable latent factors
    - Maybe: more meaningful distance between X values
    - Maybe: so that supervised tasks are easier to learn
  - AEs/VAEs: sometimes give a "useful" representation, sometimes not
    - Depends on complicated factors like the architecture, what SGD finds, ...

More representation learning

- There are lots of other ways to learn representations too!
- In some ways, the main topic of recent machine learning

ICLR 2025

The Thirteenth International Conference
on Learning Representations

- We'll cover many more
- But first, we'll want to cover another paradigm: sequence data and (relatedly) hierarchical models

Outline

Example: Vancouver Rain Data

- Consider density estimation on the "Vancouver Rain" dataset:

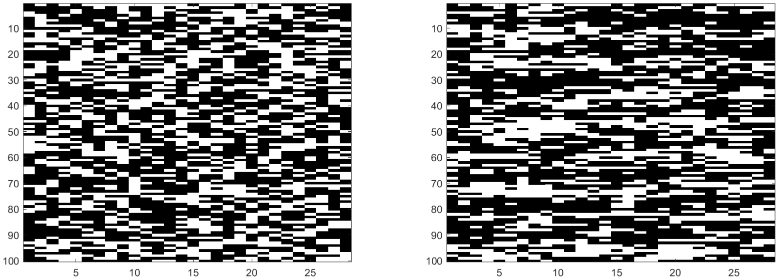
	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	...
Month 1	0	0	0	1	1	0	0	1	1	
Month 2	1	0	0	0	0	0	1	0	0	
Month 3	1	1	1	1	1	1	1	1	1	
Month 4	1	1	1	1	0	0	1	1	1	
Month 5	0	0	0	0	1	1	0	0	0	
Month 6	0	1	1	0	0	0	0	1	1	

- $x_j^{(i)} = 1$ if it rained on day j in month i, otherwise 0
  - Each row is a month, each column is a day of the month
  - This data covers from 1896 to 2004

- The strongest signals in the data:
  - It tends to rain more in the winter than the summer
  - If it rained yesterday, it's likely to rain today: $\Pr(X_j = X_{j-1}) \approx 70\%$

Rain Data with Product of Bernoullis

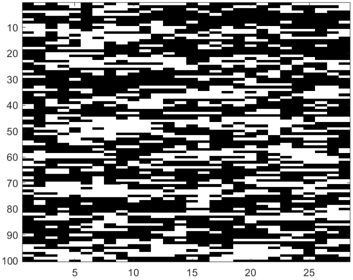
- With product of Bernoullis, we get $\Pr(X_j = \texttt{rain}) \approx 0.41$
  - Samples from product of Bernoullis model (left) vs. real data (right):

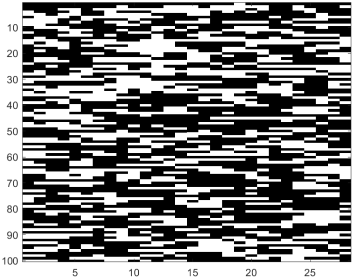


- Making days independent misses seasons and misses correlations

Markov Chains

- A better model for the between-day correlations is a Markov chain
  - Models $p(x_j \mid x_{j-1})$: probability of rain today given yesterday's value.
    - Captures dependency between adjacent days





- It can perfectly capture the "position-independent" between-day correlation
  - Only need a few parameters, and has a closed-form MLE

Rain Chain: Ingredients

- State space:
  - At time j, we can be in the `rain` state or the `not-rain` state
- Initial probabilities:

c	$\Pr(X_1 = c)$
`rain`	0.37
`not-rain`	0.63

- Transition probabilities (assumed to the same for all times j):

c_{old}	c_{new}	$\Pr(X_j = c_{\text{new}} \mid X_{j-1} = c_{\text{old}})$
`rain`	`rain`	0.65
`rain`	`not-rain`	0.35
`not-rain`	`rain`	0.25
`not-rain`	`not-rain`	0.75

- Because of "sum to 1" constraints, there are only 3 parameters in this model
- We're assuming that the order of features is meaningful
  - We're modeling dependency of each feature on the previous feature

Chain Rule of Probability

- By using the product rule, $p(a, b) = p(a)p(b \mid a)$, we can always decompose

$$\begin{aligned} p(x_1, x_2, \ldots, x_d) &= p(x_1)\, p(x_2, x_3, \ldots, x_d \mid x_1) \\ &= p(x_1)\, p(x_2 \mid x_1)\, p(x_3, x_4, \ldots, x_d \mid x_1, x_2) \\ &= p(x_1)\, p(x_2 \mid x_1)\, p(x_3 \mid x_2, x_1)\, p(x_4, x_5, \ldots, x_d \mid x_1, x_2, x_3) \end{aligned}$$

  and so on until we get

$$p(x_1, x_2, \ldots, x_d) = p(x_1)\, p(x_2 \mid x_1)\, p(x_3 \mid x_1, x_2) \cdots p(x_d \mid x_1, x_2, \ldots x_{d-1})$$

- This factorization is called the chain rule of probability
- This turns multivariate density estimation into a sequence of univariate problems
  - But with complicated conditioning...
  - For binary x_j, we'd need 2^d parameters for $p(x_d \mid x_1, x_2, \ldots, x_{d-1})$ alone
  - Or we could logistic regression / neural networks / etc to estimate conditionals

Markov Chains

- Markov chains simplify the distribution by assuming the Markov property:

$$p(x_j \mid x_{j-1}, x_{j-2}, \ldots, x_1) = p(x_j \mid x_{j-1}),$$

  that X_j is independent of the past given X_{j-1}
  - "Don't care what happened two days ago if you know what happened yesterday"

- The probability for a sequence $x_1, x_2, \cdots, x_d$ in a Markov chain simplifies to

$$\begin{aligned} p(x_1, x_2, \ldots, x_d) &= p(x_1)\, p(x_2 \mid x_1)\, p(x_3 \mid x_2, x_1) \cdots p(x_d \mid x_{d-1}, x_{d-2}, \ldots, x_1) \\ &= p(x_1)\, p(x_2 \mid x_1)\, p(x_3 \mid x_2) \cdots p(x_d \mid x_{d-1}) \end{aligned}$$

- Another way to write this joint probability is

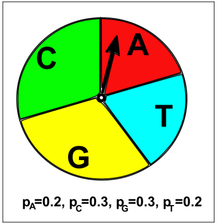
$$p(x_1, x_2, \ldots, x_d) = \underbrace{p(x_1)}_{\text{initial prob.}} \prod_{j=2}^{d} \underbrace{p(x_j \mid x_{j-1})}_{\text{transition prob.}}$$

Example: Modeling DNA Sequences

- A nice demo of independent vs. Markov for DNA sequences:
  - http://a-little-book-of-r-for-bioinformatics.readthedocs.io/en/latest/src/chapter10.html

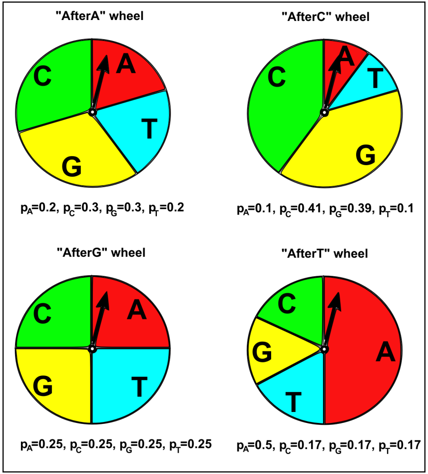
https://www.tes.com/lessons/WE5E9RncBhieAQ/dna

- Independent model for elements of sequence:



Example: Modeling DNA Sequences

- Transition probabilities in a Markov chain model for elements of sequence:



(visualizing transition probabilities based on previous symbol)

Markov Chains

- Markov chains are ubiquitous in sequence/time-series models:

9 Applications
- 9.1 Physics
- 9.2 Chemistry
- 9.3 Testing
- 9.4 Speech Recognition
- 9.5 Information sciences
- 9.6 Queueing theory
- 9.7 Internet applications
- 9.8 Statistics
- 9.9 Economics and finance
- 9.10 Social sciences
- 9.11 Mathematical biology
- 9.12 Genetics
- 9.13 Games
- 9.14 Music
- 9.15 Baseball
- 9.16 Markov text generators

Homogenous Markov Chains

- For rain data it makes sense to use a homogeneous Markov chain:
  - Transition probabilities $p(x_j \mid x_{j-1})$ are the same for all times j

- An example of parameter tying:
  1. You have more data available to estimate each parameter
     - Don't need to independently learn $p(x_3 \mid x_2)$ and $p(x_{24} \mid x_{23})$
  2. You can have training examples of different sizes
     - Same model can be used for any number of days
     - We could even treat the rain data as one long Markov chain ($n = 1$)

Homogenous Markov Chains

- With discrete states, we could use tabular parameterization for transitions,

$$\Pr(X_j = c' \mid X_{j-1} = c) = \theta_{c \to c'}$$

  where $\theta_{c \to c'} \geq 0$ and $\sum_{c=1}^{k} \theta_{c \to c'} = 1$ (and we use the same $\theta_{c' \to c}$ for all j)
  - Another way of putting this: $X_j \mid (X_{j-1} = c) \sim \mathrm{Cat}(\theta_{c \to 1}, \dots, \theta_{c \to k})$

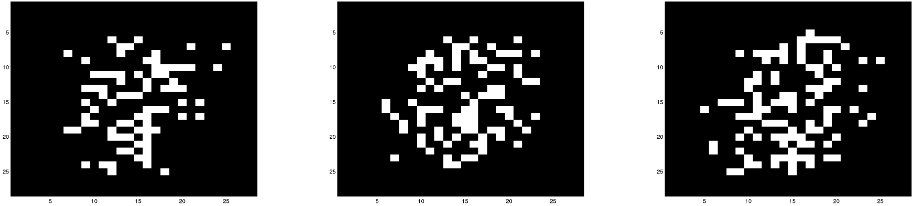
- MLE for homogeneous Markov chain with discrete x_j and tabular parameters:

$$\theta_{c \to c'} = \frac{\text{(number of transitions from } c \text{ to } c'\text{)}}{\text{(number of times we went from } c \text{ to anything)}};$$

  learning is just counting

Density Estimation for MNIST Digits

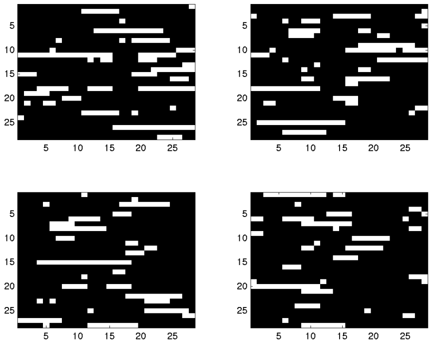
- We've previously considered density estimation for MNIST images of digits
- We saw that product of Bernoullis does terribly



- This model misses correlation between adjacent pixels
  - Could we capture this with a Markov chain?

Density Estimation for MNIST Digits

- Samples from a homogeneous Markov chain (putting rows into one long vector):



- Captures correlations between adjacent pixels in the same row.
  - But misses long-range dependencies in row and dependencies between rows
  - Also, "position independence"/homogeneity means it loses position information

Inhomogeneous Markov Chains

- We could allow a different $p(x_j \mid x_{j-1})$ for each j
  - This makes sense for digits data, but probably not for the rain data

- For discrete x_j we could use a tabular parameterization,

$$\Pr(X_j = c' \mid X_{j=1} = c) = \theta^{(j)}_{c \to c'}$$

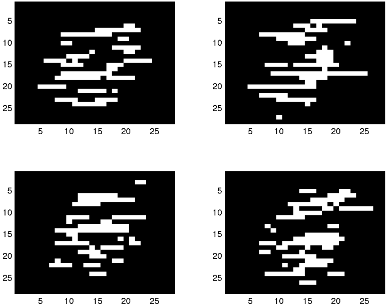
- MLE under this parameterization is given by

$$\theta^{j}_{c \to c'} = \frac{(\text{number of transitions from } c \text{ to } c' \text{ starting at } (j-1))}{(\text{number of times we saw } c \text{ at position } (j-1))},$$

- Inhomogeneous Markov chains include independent models as special case:
  - Use $p(x_j \mid x_{j-1}) = p(x_j)$ for all j; becomes a product of independent models

Density Estimation for MNIST Digits

- Samples from an inhomogeneous Markov chain fit to digits:



- We have correlations between adjacent pixels in rows, and position information
  - But it isn't capturing long-range dependencies or dependency between rows
  - Graphical models, soon, can address this

Training Markov Chains

- Some common setups for fitting the parameters of Markov chains:
  1. We have one long sequence, and fit parameters of a homogeneous Markov chain
     - Here, we just focus on the transition probabilities
  2. We have many sequences of different lengths, and fit a homogeneous chain
     - And we can use it to model sequences of any length
  3. We have many sequences of same length, and fit an inhomgeneous Markov chain
     - This allows "position-specific" effects
  4. We use domain knowledge to guess the initial and transition probabilities
     - Here we would be interested in inference in the model

Fun with Markov Chains

bonus!

- Markov Chains "Explained Visually":
  `http://setosa.io/ev/markov-chains`
- Snakes and Ladders:
  `http://datagenetics.com/blog/november12011/index.html`
- Candyland:
  `http://www.datagenetics.com/blog/december12011/index.html`
- Yahtzee:
  `http://www.datagenetics.com/blog/january42012/`
- Chess pieces returning home and K-pop vs. ska:
  `https://www.youtube.com/watch?v=63HHmjlh794`

Outline

Inference in Markov Chains

- Given a Markov chain model, these are the most common inference tasks:
  1. Sampling: generate sequences that follow the distribution
  2. Marginalization: compute probability of being in state c at time j
  3. Stationary distribution: probability of being in state c as j goes to ∞
     - Usually for homogeneous Markov chains
  4. Mode decoding: compute assignment of the x_j that has highest joint probability
     - Usually for inhomogeneous Markov chains (important for supervised learning)
  5. Conditioning: do any of the above, assuming $x_j = c$ for some j and c
     - For example, "filling in" missing parts of a sequence

Ancestral Sampling

- To sample dependent random variables we can use the chain rule of probability,

$$p(x_1, x_2, x_3, \ldots, x_d) = p(x_1)\, p(x_2 \mid x_1)\, p(x_3 \mid x_2, x_1) \cdots p(x_d \mid x_{d-1}, x_{d-2}, \ldots, x_1)$$

- The chain rule suggests the following sampling strategy:
  - Sample x_1 from $p(x_1)$
  - Given x_1, sample x_2 from $p(x_2 \mid x_1)$
  - Given x_1 and x_2, sample x_3 from $p(x_3 \mid x_2, x_1)$
  - ...
  - Given x_1 through x_{d-1}, sample x_d from $p(x_d \mid x_{d-1}, x_{d-2}, \ldots x_1)$

- This is called ancestral sampling
  - It's easy if conditional probabilities are simple, since sampling in 1D is usually easy
  - But may not be simple; binary conditional j has 2^j values of $\{x_1, x_2, \ldots, x_j\}$

Ancestral Sampling Examples

- For Markov chains the chain rule simplifies to

$$p(x_1, x_2, x_3, \ldots, x_d) = p(x_1)\, p(x_2 \mid x_1)\, p(x_3 \mid x_2) \cdots p(x_d \mid x_{d-1})$$

- This means ancestral sampling simplifies, too:
  1. Sample x_1 from initial probabilities $p(x_1)$
  2. Given x_1, sample x_2 from transition probabilities $p(x_2 \mid x_1)$
  3. Given x_2, sample x_3 from transition probabilities $p(x_3 \mid x_2)$
  4. ...
  5. Given x_{d-1}, sample x_d from transition probabilities $p(x_d \mid x_{d-1})$

Markov Chain Toy Example: CS Grad Career

- "Computer science grad career" Markov chain:
  - Initial probabilities:

State	Probability	Description
Industry	0.60	They work for a company or own their own company.
Grad School	0.30	They are trying to get a Masters or PhD degree.
Video Games	0.10	They mostly play video games.

  - Transition probabilities (from row to column):

From\to	Video Games	Industry	Grad School	Video Games (with PhD)	Industry (with PhD)	Academia	Deceased
Video Games	0.08	0.90	0.01	0	0	0	0.01
Industry	0.03	0.95	0.01	0	0	0	0.01
Grad School	0.06	0.06	0.75	0.05	0.05	0.02	0.01
Video Games (with PhD)	0	0	0	0.30	0.60	0.09	0.01
Industry (with PhD)	0	0	0	0.02	0.95	0.02	0.01
Academia	0	0	0	0.01	0.01	0.97	0.01
Deceased	0	0	0	0	0	0	1

- Here $\Pr(x_t = \text{"Grad School"} \mid x_{t-1} = \text{"Industry"}) = 0.01$

Example of Sampling x_1

- Initial probabilities are:
  - 0.1 (Video Games)
  - 0.6 (Industry)
  - 0.3 (Grad School)
  - 0 (Video Games with PhD)
  - 0 (Academia)
  - 0 (Deceased)
- So initial CDF is:
  - 0.1 (Video Games)
  - 0.7 (Industry)
  - 1 (Grad School)
  - 1 (Video Games with PhD)
  - 1 (Academia)
  - 1 (Deceased)

- To sample the initial state x_1:
  - First generate a number $u \sim \mathrm{Uniform}(0, 1)$, for example $u = 0.724$
  - Now find the first CDF value bigger than u, which in this case is "Grad School"

Example of Sampling x_2, Given x_1 = "Grad School"

- So we sampled x_1 = "Grad School"
  - To sample x_2, we'll use the "Grad School" row in transition probabilities:

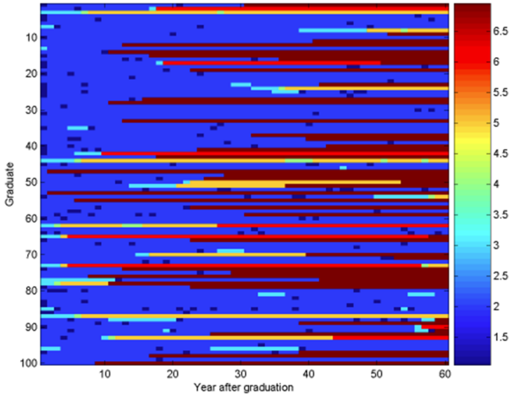
From\to	Video Games	Industry	Grad School	Video Games (with PhD)	Industry (with PhD)	Academia	Deceased
Video Games	0.08	0.90	0.01	0	0	0	0.01
Industry	0.03	0.95	0.01	0	0	0	0.01
Grad School	0.06	0.06	0.75	0.05	0.05	0.02	0.01
Video Games (with PhD)	0	0	0	0.30	0.60	0.09	0.01
Industry (with PhD)	0	0	0	0.02	0.95	0.02	0.01
Academia	0	0	0	0.01	0.01	0.97	0.01
Deceased	0	0	0	0	0	0	1

Example of Sampling x_2, Given $x_1 =$ "Grad School"

- Transition probabilities:
  - 0.06 (Video Games)
  - 0.06 (Industry)
  - 0.75 (Grad School)
  - 0.05 (Video Games with PhD)
  - 0.02 (Academia)
  - 0.01 (Deceased)
- So transition CDF is:
  - 0.06 (Video Games)
  - 0.12 (Industry)
  - 0.87 (Grad School)
  - 0.97 (Video Games with PhD)
  - 0.99 (Academia)
  - 1 (Deceased)
- To sample the second state x_2:
  - First generate a number $u \sim \text{Uniform}(0, 1)$, for example $u = 0.113$
  - Now find the first CDF value bigger than u, which in this case is "Industry"

Markov Chain Toy Example: CS Grad Career

- Samples from "computer science grad career" Markov chain:



- State 7 ("deceased") is called an absorbing state (no probability of leaving)
- Samples often give you an idea of what model knows (and what should be fixed)

Aside: Ancestral Sampling with Blocks of Variables

- We often factorize variables in terms of blocks of variables, as in

$$p(x_1, x_2, x_3, x_4, x_5, x_6) = p(x_1, x_2)\, p(x_3, x_4 \mid x_1, x_2)\, p(x_5, x_6 \mid x_1, x_2, x_3, x_4)$$

- With this factorization ancestral sampling takes the form
  1. Sample x_1 and x_2 from $p(x_1, x_2)$
  2. Given x_1 and x_2, sample x_3 and x_4 from $p(x_3, x_4 \mid x_2, x_1)$
  3. Given $x_{1:4}$, sample x_5 and x_6 from $p(x_5, x_6 \mid x_1, x_2, x_3, x_4)$

- For example, in Gaussian discriminant analysis we write

$$p(x, y) = p(y)\, p(x \mid y)$$

- Sampling from Gaussian discriminant analysis:
  1. Sample y from the categorical distribution $p(y)$
  2. Sample x from the multivariate Gaussian $p(x \mid y)$
- Sampling from a deep latent variable model: exactly the same form

Marginalization and Conditioning

- Given a density estimator, we often want to make probabilistic inferences:
  - Marginals: what is the probability that $X_j = c$?
    - What's the probability we're in industry 10 years after graduation?
  - Conditionals: what is the probability that $X_j = c$ given $X_{j'} = c'$?
    - What's the probability of industry after 10 years, if we immediately go to grad school?

- This is easy for simple independent models:
  - We directly model marginals $p(x_j)$
  - Conditionals are marginals: $p(x_j \mid x_{j'}) = p(x_j)$

- For Markov chains, it's more complicated
  - $p(x_4)$ depends on the values of x_1, x_2 and x_3
  - $p(x_4 \mid x_8)$ additionally depends on the values x_5, x_6, x_7, x_8

Monte Carlo Methods for Markov Chains

- We could use Monte Carlo approximations for inference in Markov chains:
  - Marginal $\Pr(x_j = c) \approx$ the portion of chains that were in state c at time j
  - Average value at time j is $\mathbb{E}[x_j] \approx$ average of samples $x_j^{(i)}$
  - $\Pr(5 \leq x_j \leq 10) \approx$ portion of chains with x_j between 5 and 10
    - This makes more sense for continuous states than evaluating equalities
  - $\Pr(x_j \leq 10, x_{j+1} \geq 10) \approx$ portion of chains where both happen

- Monte Carlo works for continuous states too (for inequalities and expectations)

- In typical settings Monte Carlo has slow convergence (like stochastic gradient)
  - For $\mathbb{E}[f(x)]$, the estimate $\frac{1}{n}\sum_{i=1}^{n} f(x^{(i)})$ has variance $\mathrm{Var}(f(x))/n$
    - If all samples look about the same ($\mathrm{Var}(f(X))$ is small), it converges quickly
    - If samples vary a lot, it can be painfully slow

Exact Marginal Calculation

- For discrete-state Markov chains, we can actually compute marginals directly
- We're given initial probabilities $\Pr(x_1 = c)$ for all c as part of the definition
- We can use transition probabilities to compute $p(x_2 = c)$ for all c:

$$p(x_2) = \underbrace{\sum_{x_1=1}^{k} p(x_2, x_1)}_{\text{marginalization rule}} = \sum_{x_1=1}^{k} \underbrace{p(x_2 \mid x_1)p(x_1)}_{\text{product rule}}$$

- Same calculation gives

$$p(x_3) = \sum_{x_2=1}^{k} p(x_3, x_2) = \sum_{x_2=1}^{k} p(x_3 \mid x_2)p(x_2)$$

- So we have $p(x_3)$ in terms of $p(x_2)$, and $p(x_2)$ in terms of $p(x_1)$, which we know

Exact Marginal Calculation

- Recursive formula for maginals at time j:

$$p(x_j) = \sum_{x_{j-1}=1}^{k} p(x_j \mid x_{j-1}) p(x_{j-1}),$$

  called the Chapman-Kolmogorov (CK) equations

- The CK equations can be implemented as matrix-vector multiplication:
  - Define $\pi^{(j)}$ as a vector containing the marginals at time j:

$$\pi_c^{(j)} = \Pr(x_j = c)$$

  - Define $T^{(j)}$ as a matrix containing the transition probabilities:

$$T_{c'c}^{(j)} = \Pr(x_j = c' \mid x_{j-1} = c)$$

  - Rule is just $\pi^{(j)} = T^j \pi^{(j-1)}$

Exact Marginal Calculation

- Implementing the CK equations as a matrix multiplication:

$$T^{(j)}\pi^{(j-1)} = \begin{bmatrix} \Pr(X_j = 1 \mid X_{j-1} = 1) & \Pr(X_j = 1 \mid X_{j-1} = 2) & \dots & \Pr(X_j = 1 \mid X_{j-1} = k) \\ \Pr(X_j = 2 \mid X_{j-1} = 1) & \Pr(X_j = 2 \mid X_{j-1} = 2) & \dots & \Pr(X_j = 2 \mid X_{j-1} = k) \\ \Pr(X_j = k \mid X_{j-1} = 1) & \Pr(X_j = k \mid X_{j-1} = 2) & \dots & \Pr(X_j = k \mid X_{j-1} = k) \end{bmatrix} \begin{bmatrix} \Pr(X_{j-1} = 1) \\ \Pr(X_{j-1} = 2) \\ \vdots \\ \Pr(X_{j-1} = k) \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{c=1}^{k} \Pr(X_j = 1 \mid X_{j-1} = c) \Pr(X_{j-1} = c) \\ \sum_{c=1}^{k} \Pr(X_j = 2 \mid X_{j-1} = c) \Pr(X_{j-1} = c) \\ \vdots \\ \sum_{c=1}^{k} \Pr(X_j = k \mid X_{j-1} = c) \Pr(X_{j-1} = c) \end{bmatrix} = \begin{bmatrix} \Pr(X_j = 1) \\ \Pr(X_j = 2) \\ \vdots \\ \Pr(X_j = k) \end{bmatrix} = \pi^{(j)}$$

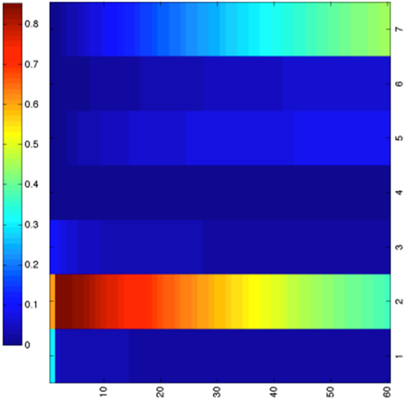
- Cost of multiplying a vector by a $k \times k$ matrix is $\mathcal{O}(k^2)$

- So cost to compute marginals up to time d is $\mathcal{O}(dk^2)$
  - This is fast, considering that last step sums over all k^d possible sequences

$$p(x_d) = \sum_{x_1=1}^{k} \sum_{x_2=1}^{k} \cdots \sum_{x_{j-1}=1}^{k} \sum_{x_{j+1}=1}^{k} \cdots \sum_{x_{d-1}=1}^{k} p(x_1, x_2, \dots, x_d)$$

Marginals in CS Grad Career

- CK equations can give all marginals $p(x_j = c)$ from CS grad Markov chain:



- Each row j is a state, and each column c is a year

Continuous-State Markov Chains

bonus!

- The CK equations also apply if we have continuous states:

$$p(x_j) = \int_{x_{j-1}} p(x_j \mid x_{j-1}) p(x_{j-1}) \mathrm{d}x_{j-1}$$

  but this integral may not have a closed-form solution

- Gaussian probabilities are an important special case:
  - If $p(x_{j-1})$ and $p(x_j \mid x_{j-1})$ are Gaussian, then $p(x_j)$ is Gaussian
    - Marginal of product of Gaussians
  - So we can write $p(x_j)$ in closed-form in terms of a mean and variance
    - Also works if states are vectors, with initial/transition following multivariate Gaussian

- If the probabilities are non-Gaussian, usually can't represent $p(x_j)$ distribution
  - Gaussian has the special property that it is its own conjugate prior
  - With other distributions, usually stuck using Monte Carlo or other approximations

Stationary Distribution

- A stationary distribution of a homogeneous Markov chain is a distribution π with

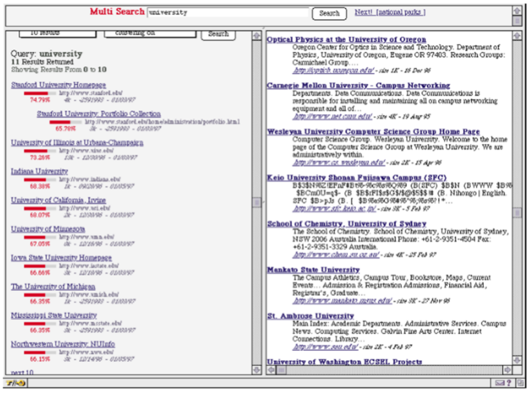
$$\pi(c) = \sum_{c'} p(x_j = c \mid x_{j-1} = c')\pi(c') \quad \text{or equivalently} \quad \pi = T\pi$$

- "Marginal probabilities don't change across time"
  - A stationary distribution is called an "invariant" distribution
  - Note this does not imply it converges to a single state
- Under certain conditions, marginals converge to a stationary distribution
  - $p(x_j = c) \to \pi(c)$ as j goes to ∞
  - If we fit a Markov chain to the rain example, we have $\pi(\texttt{rain}) = 0.41$
  - In the CS grad student example, we have $\pi(\texttt{deceased}) = 1$
- Stationary distribution is basis for Google's PageRank algorithm

Application: PageRank

bonus!

- Web search before Google:



http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf

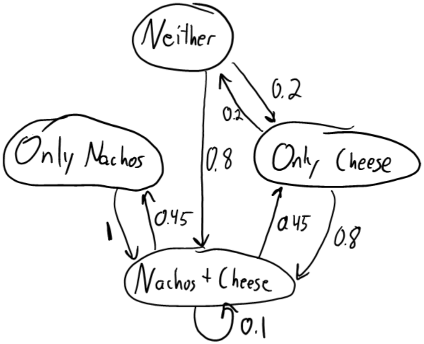
- It was also easy to fool search engines by copying popular websites

State Transition Diagram

bonus!

- State transition diagrams are common for visualizing homogenous Markov chains:

$$T = \begin{bmatrix} 0 & 0 & 0.2 & 0.8 \\ 0 & 0 & 0 & 1 \\ 0.2 & 0 & 0 & 0.8 \\ 0 & 0.45 & 0.45 & 0.1 \end{bmatrix}$$



- Each node is a state, each edge is a non-zero transition probability
  - For web-search, each node will be a webpage
- Cost of CK equations is only $O(z)$ instead of $O(k^2)$ if you have only z edges

Application: PageRank

bonus!

- Wikipedia's cartoon illustration of Google's PageRank:
  - Large face means higher rank

https://en.wikipedia.org/wiki/PageRank

- "Important webpages are linked from other important webpages"
- "A link is more meaningful if the webpage has few links"

Application: PageRank

bonus!

- Google's PageRank algorithm for measuring the importance of a website:
  - Stationary probability in "random surfer" Markov chain:
    - With probability α, surfer clicks on a random link on the current webpage
    - Otherwise, surfer goes to a completely random webpage
- To compute the stationary distribution, they use the power method:
  - Just start with some distribution, then repeatedly apply the CK equations
  - Iterations are faster than $O(k^2)$ due to sparsity of links
    - Transition matrix is "sparse plus rank-1," which allows fast multiplication
  - Can be easily parallelized

Application: Game of Thrones

bonus!

- PageRank can be used in other applications.
- "Who is the main character in the Game of Thrones books?"

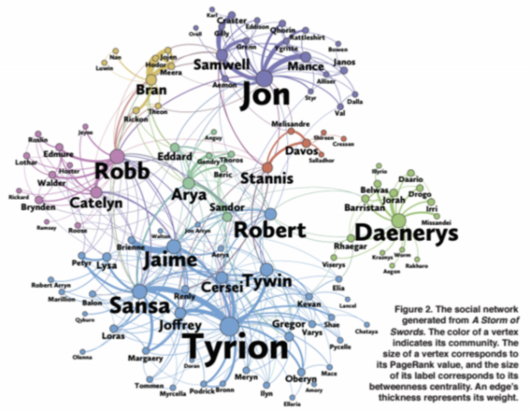


Figure 2. The social network generated from *A Storm of Swords*. The color of a vertex indicates its community. The size of a vertex corresponds to its PageRank value, and the size of its label corresponds to its betweenness centrality. An edge's thickness represents its weight.

http://qz.com/650796/mathematicians-mapped-out-every-game-of-thrones-relationship-to-find-the-main-character

Existence/Uniqueness of Stationary Distribution

bonus!

- Does a stationary distribution π exist and is it unique?
- Sufficient condition for existence/uniqueness: all $\Pr(X_j = c \mid X_{j'} = c') > 0$
  - PageRank satisfies this by adding probability $(1-\alpha)$ of jumping to a random page
- Weaker sufficient condition for existence and uniqueness is ergodicity:
  1. "Irreducible" (doesn't get stuck in part of the graph, e.g. at absorbing states)
  2. "Aperiodic" (probability of returning to state isn't on fixed intervals)

Summary

- Markov chains model dependencies between adjacent features
  - Set of possible states; initial probabilities; transition probabilities
- Chain rule of probability
  - Writes joint probability in terms of conditionals over "earlier" variables
- Markov assumption
  - Conditional independence from "past" times given previous time
- Homogeneous Markov chains: same transition probabilities across time
  - Allows sequences of different lengths; more data to estimate transition parameters
- Inhomogeneous Markov chains: transition probabilities can vary
- Ancestral sampling generates samples from multivariate distributions
  - Use chain rule of probability, sequentially sample variables from conditionals
- Chapman-Kolmogorov equations compute exact univariate marginals
  - For discrete or Gaussian Markov chains
- Stationary distribution of homogenous Markov chain
  - Marginals as time goes to ∞; basis of e.g. Google's PageRank method

- Next time: we finally use that dynamic programming prereq

Label Propagation as a Markov Chain Problem

bonus!

- Semi-supervised label propagation method has a Markov chain interpretation
  - We have $n + t$ states, one for each [un]labeled example
- Monte Carlo approach to label propagation ("adsorption"):
  - At time $t = 0$, set the state to the node you want to label
  - At time $t > 0$ and on a labeled node, output the label
    - Labeled nodes are absorbing states
  - At time $t > 0$ and on an unlabeled node i:
    - Move to neighbour j with probability proportional w_{ij} (or $\bar{w}_{ij}$)
- Final predictions are probabilities of outputting each label
  - Nice if you only need to label one example at a time (slow if labels are rare)
  - Common hack is to limit random walk time to bound runtime