DSCI 575:
Advanced Machine Learning

Markov Chains *and* Monte Carlo

Winter 2018

Example: Vancouver Rain Data

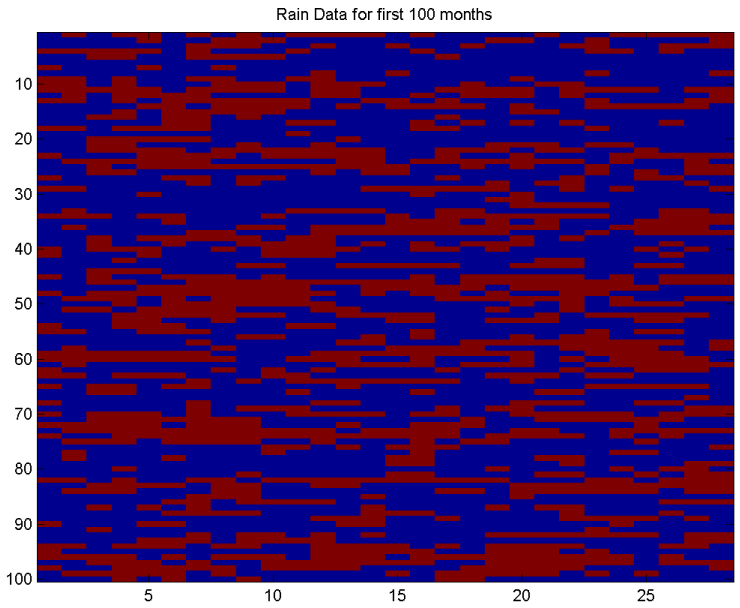
- Consider modeling the "Vancouver rain" dataset.

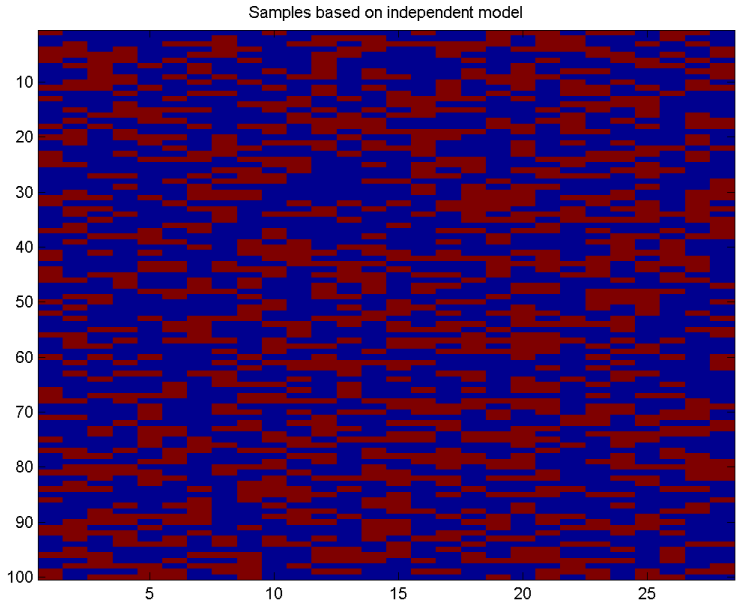
	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	…
Month 1	0	0	0	1	1	0	0	1	1	
Month 2	1	0	0	0	0	0	1	0	0	
Month 3	1	1	1	1	1	1	1	1	1	
Month 4	1	1	1	1	0	0	1	1	1	
Month 5	0	0	0	0	1	1	0	0	0	
Month 6	0	1	1	0	0	0	0	1	1	

- A time-series dataset where $x_t = 1$ if it rained on day 't'.
- The strongest signal in the data is the simple relationship:
  - If it rained yesterday, it's likely to rain today (> 50% chance that $x_{t-1} = x_t$).

Example: Vancouver Rain Data

- If we assume x_t are independent, we get $p(x_t = 1) = 0.41$ (sadly).
  - Real data vs. samples from independent Bernoulli model:

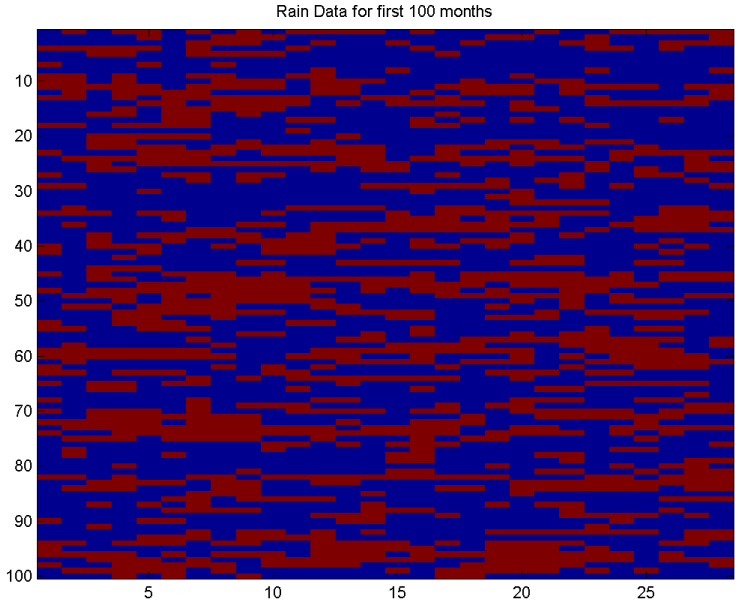


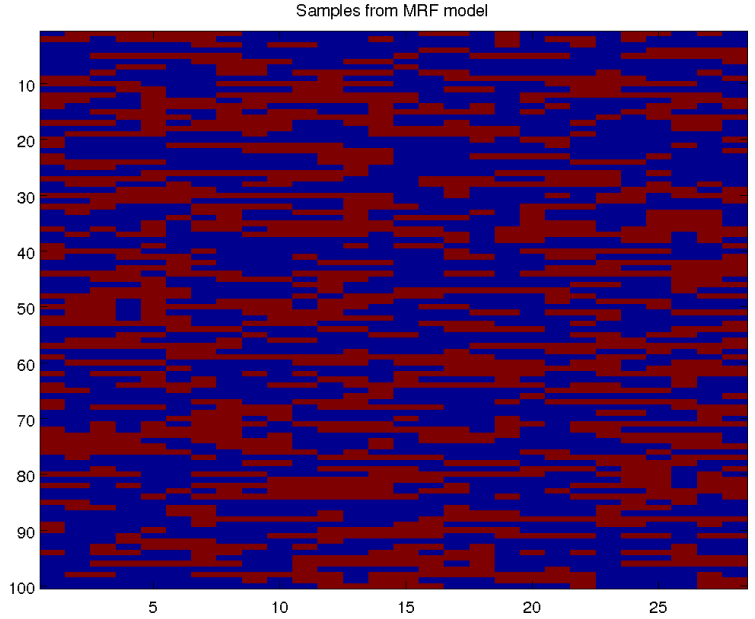


  - Making days independent misses correlation.

Markov Chain Model of Rain Data

- A better model for the rain data is a Markov chain:
  - Captures dependency of x_t on x_{t-1}.





  - We model $p(x_t \mid x_{t-1})$: probability of rain today given yesterday's value.

Markov Chain Ingredients (MEMORIZE)

- Markov chain ingredients:
  - State space:
    - Set of possible states (indexed by 's') we can be in at time 't' ("rain" or "not rain").
  - Initial probabilities:
    - $p(x_1 = s)$ that we start in state 's' at time 1.
  - Transition probabilities:
    - $p(x_t = s \mid x_{t-1} = s')$ that we move to state s from state s' at time 't'.
      - Probability that it rains today, given what happened yesterday.

- For PageRank: each webpage is a state 's'.
  - Initial probability is random.
  - Go to random page with probability α, otherwise go to random neighbour.

Markov Chain Probability and Markov Property

- Markov chain probability for a sequence $x_1, x_2, \ldots, x_d$:

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

- This assumes the Markov property:

$$p(x_t \mid x_1, x_2, x_3, \ldots, x_{t-1}) = p(x_t \mid x_{t-1})$$

  - That x_t is independent of the past given x_{t-1}.
    - To predict "rain", we only need to know whether it rained yesterday.

Markov Chain Applications

Homogeneous Markov Chains

- We usually assume that the Markov chain is homogeneous:
  - Transition probabilities $p(x_t = s \mid x_{t-1} = s')$ are same for all 't'.

- Given 'n' samples, MLE for homogeneous Markov chain is:

$$\text{Initial: } p(x_1 = s) = \frac{\text{number of times we start in state } s}{n}$$

$$\text{Transition: } p(x_t = s \mid x_{t-1} = s') = \frac{\text{number of times we went from } s' \text{ to } s}{\text{number of times we went from } s' \text{ to anything}}$$

- So given one or more sequences, learning is just counting.
  - Like in naïve Bayes.

Computation with Markov Chains

- Common things we do with Markov chains:
  - Sampling: generate sequences that follow the probability.
    - This is what our “random walk” algorithms are doing.
  - Inference: compute probability of being in state ‘s’ at time ‘t’.
  - Decoding: compute most likely sequence of states.
  - Conditioning: do any of the above, assuming $x_t = s$ for some ‘t’ and ‘s’.
    - For example, “filling in” missing parts of a sequence.
  - Stationary distribution: probability of being ‘s’ at ‘t’ goes to ∞.
    - PageRank.

Fun with Markov Chains

- 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

(pause)

Fundamental Problem: Sampling from a Density

- A fundamental problem in data science is sampling from a density.
  - Generating examples x_i that are distributed according to given density p(x).
  - Basically, the "opposite" of learning: going from a model to data.

$$p(x) = \begin{cases} 1 & \text{w.p. } 0.5 \\ 2 & \text{w.p. } 0.25 \\ 3 & \text{w.p. } 0.25 \end{cases} \Rightarrow X = \begin{bmatrix} 1 \\ 2 \\ 1 \\ 1 \\ 3 \\ 2 \\ 1 \\ 2 \end{bmatrix}$$

- Sometimes we use samples to "tell us what the model learning".
  - If the samples look like real data, then we have a good model.

- Samples can also be used in Monte Carlo estimation (later):
  - Replace complicated p(x) with samples to solve hard problems at test time.

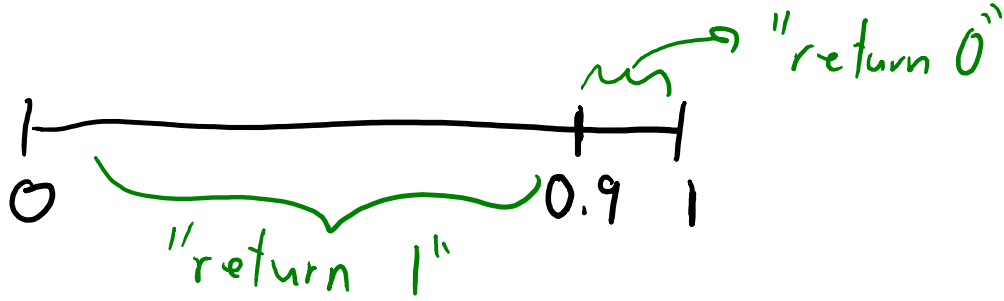
Simplest Case: Sampling from a Bernoulli

- Consider sampling from a Bernoulli, for example:

$$p(x=1) = 0.9 \qquad p(x=0) = 0.1$$

- Sampling methods assume we sample uniformly over [0,1].
  - Usually, a "pseudo-random" number is good enough (Python/R).

- How to use a uniform sample to sample from the Bernoulli above:
  1. Generate a uniform sample u ~ U(0,1).
  2. Set x = 1 If u ≤ 0.9.



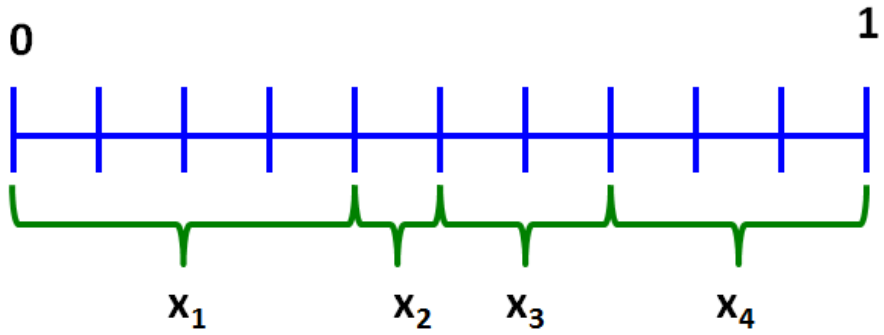
- With good uniform sampler, then we have x=1 with probability 0.9.

Sampling from a Categorical Distribution

- Consider a more general categorical density like:

$$p(x=1)=0.4 \quad p(x=2)=0.1 \quad p(x=3)=0.2 \quad p(x_4)=0.3$$

- We can divide the [0,1] interval based on probability values:



- If u ~ U(0,1), 40% of the time it lands in the x_1 region.
  - And 10% in x_2, 20% in x_3, and 30% in x_4.

Sampling from a Categorical Distribution

- Consider a more general categorical density like:

$$p(x=1)=0.4 \quad p(x=2)=0.1 \quad p(x=3)=0.2 \quad p(x_4)=0.3$$

- To sample from this categorical density we can use:
  1. Generate u ~ U(0,1).
  2. If u ≤ 0.4, output 1.
  3. If u ≤ 0.4 + 0.1, output 2.
  4. If u ≤ 0.4 + 0.1 + 0.2, output 3.
  5. Otherwise, output 4.

Sampling from a Categorical Distribution

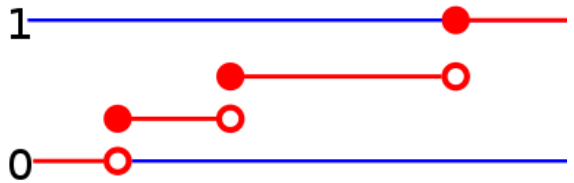
- General case for sampling from categorical:
  1. Generate u ~ U(0,1).
  2. If u ≤ p(x ≤ 1), output 1.
  3. If u ≤ p(x ≤ 2), output 2.
  4. If u ≤ p(x ≤ 3), output 3.
  5. …

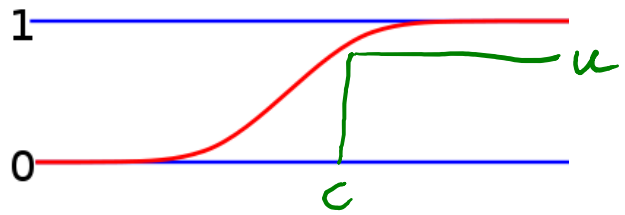
- The value p(x ≤ c) = p(x = 1) + p(x = 2) + … + p(x = c) is the CDF.
  - "Cumulative distribution function".

- The CDF can use be used to sample from continuous densities…

Inverse Transform Method (Exact 1D Sampling)

- We often use $F(c) = p(x \leq c)$ to denote the CDF:
  - F(c) is between 0 and 1, gives proportion of times 'x' is below 'c'.
  - F can be used for discrete and continuous variables:





- The inverse CDF ("quantile") function F^{-1} is its inverse:
  - Give u between 0 and 1, $F^{-1}(u)$ is the c such that $p(x \leq c) = u$.

- Inverse transform method for exact sampling in 1D (how you sample normal):
  - Sample $u \sim U(0,1)$.
  - Return $F^{-1}(u)$.

- Video on pseudo-random numbers and inverse-transform sampling from PBS.

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

Sampling from a Product Distribution

- Consider a model where the variables are independent:

$$p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2)p(x_3)p(x_4)$$

- Because of independence, we can sample independently:
  - Sample x_1 from $p(x_1)$.
  - Sample x_2 from $p(x_2)$.
  - Sample x_3 from $p(x_3)$.
  - Sample x_4 from $p(x_4)$.

- But we can't use this for Markov chains due to dependence.

Ancestral Sampling

- To sample dependent random variables we can use chain rule:

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

- The chain rule suggests the following sampling strategy:
  - Sample x_1 from $p(x_1)$.
  - Sample x_2 from $p(x_2 \mid x_1)$ for the sampled x_1.
  - Sample x_3 from $p(x_3 \mid x_2, x_1)$ for the sampled x_1 and x_2.
  - Sample x_4 from $p(x_4 \mid x_3, x_2, x_1)$ for the sampled $\{x_1, x_2, x_3\}$.

- This called ancestral sampling.
  - It's easy if conditional probabilities are simple.
  - But may not be simple, binary conditional 't' has 2^t values of $\{x_1, x_2, \ldots, x_t\}$

Example: Ancestral Sampling for Markov Chains

- For Markov chains the chain rule simplifies to:

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

- So ancestral sampling simplifies (only depends on last time):
  - Sample x_1 from $p(x_1)$.
  - Sample x_2 from $p(x_2 \mid x_1)$ for the sampled x_1.
  - Sample x_3 from $p(x_3 \mid x_2)$ for the sampled x_2.
  - Sample x_4 from $p(x_4 \mid x_3)$ for the sampled x_3.

- In PageRank and label propagation, this is the random walk.

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\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

  - So $p(x_t = \text{"Grad School"} \mid x_{t-1} = \text{"Industry"}) = 0.01$.

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 (Industry with PhD)
  - 0 (Acadmia)
  - 0 (Deceased)
- So initial CDF is:
  - 0.1 (Video Games)
  - 0.7 (Industry)
  - 1 (Grad School)
  - 1 (Video Games with PhD)
  - 1 (Industry with PhD)
  - 1 (Academia)
  - 1 (Deceased)
- To sample the initial state x_1:
  - First generate u, which we'll assume is u=0.724.
  - Now find 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 x2, we'll use the "Grad School" row in transition probabilities:

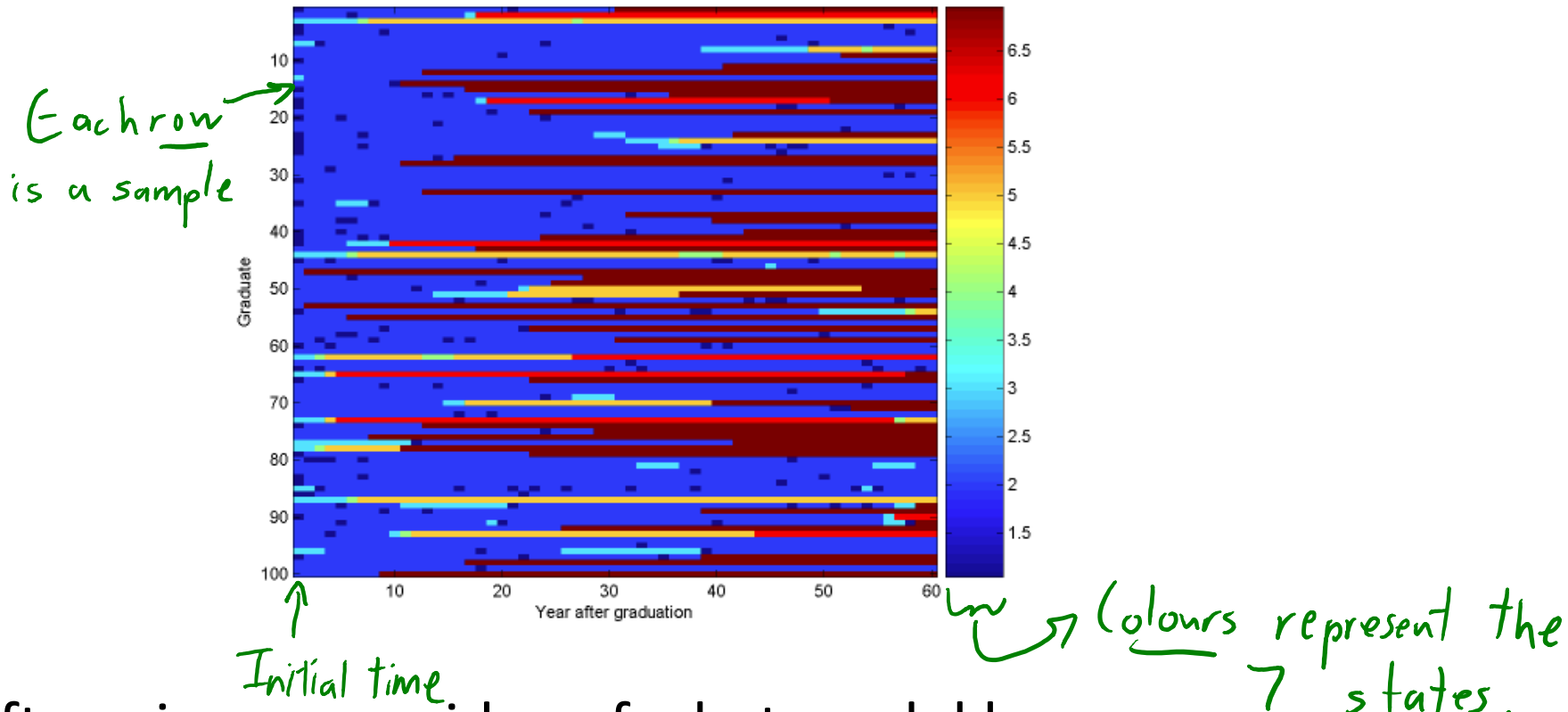
From\to	Video Games	Industry	Grad School	Video Games (with PhD)	Industry (with PhD)	Academia	Deceased
Video Games	0.08	0.90	0.01	0	0	0	0.01
Industry	0.03	0.95	0.01	0	0	0	0.01
Grad School	0.06	0.06	0.75	0.05	0.05	0.02	0.01
Video Games (with PhD)	0	0	0	0.30	0.60	0.09	0.01
Industry (with PhD)	0	0	0	0.02	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)
  - 0.05 (Industry with PhD)
  - 0.02 (Acadmia)
  - 0.01 (Deceased)
- So transition CDF is:
  - 0.06 (Video Games)
  - 0.12 (Industry)
  - 0.87 (Grad School)
  - 0.92 (Video Games with PhD)
  - 0.97 (Industry with PhD)
  - 0.99 (Academia)
  - 1 (Deceased)
- To sample the second state x_2:
  - First generate u, which we'll assume is u=0.113.
  - Now find first CDF value bigger than u, which in this case is "Industry".

Sampling from a Markov Chain

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



- Samples often give you an idea of what model knows.
  - And what should be fixed.

(pause)

Computing Marginals and Conditionals

- Given a joint probability 'p', we often want to do "inferences":
  - Marginals: what $p(x_{10} = \text{"industry"})$?
    - What is the probability we're in industry 10 years after graduation?
  - Conditionals: what $p(x_{10} = \text{"industry"} \mid x_1 = \text{"academia"})$.
    - What is the probability of industry after 10 years, if we go to grad school?
- Where are we going with this?
  - Faster calculation of PageRank: $p(x_\infty = \text{"www.nba.com"})$.
  - Bayesian machine learning requires doing calculation like this.

Monte Carlo: Inference by Sampling

- A basic Monte Carlo method for estimating probabilities of events:
  1. Generate a large number of samples from the model.

$$X = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

  2. Compute the frequency that the event happened in the samples.

$$p(x_2 = 1) \approx 3/4 \qquad p(x_3 = 0) \approx 0/4$$

- Monte Carlo is second most important class of ML algorithms.
  - Originally developed to build better atomic bombs ☹
    - Run physics simulator to “sample”, then see if it lead to a chain reaction.

Monte Carlo Method for Rolling Di

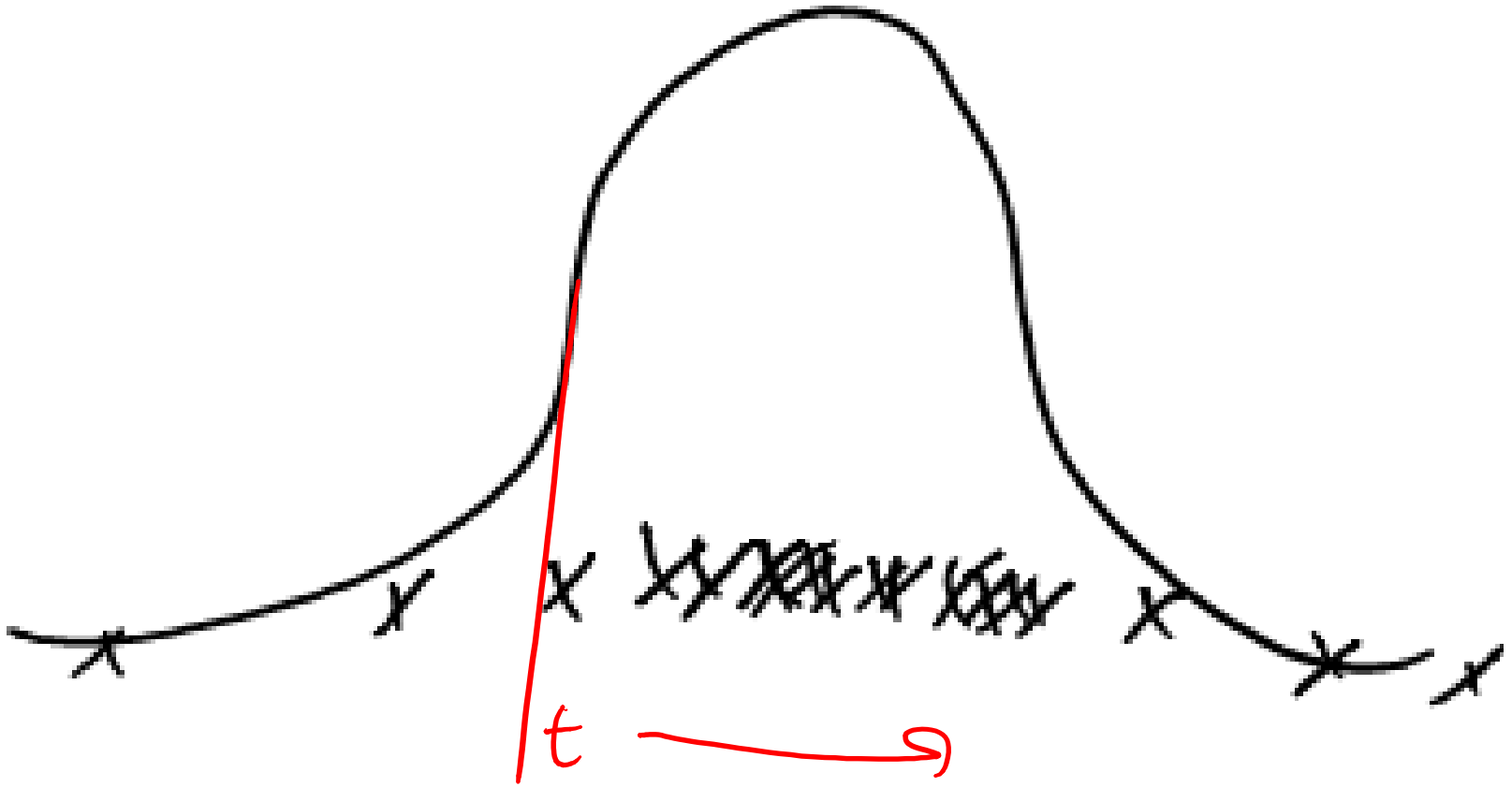
- Probability of event:
  - (number of samples where event happened) / (number of samples)

- Computing probability of a pair of dice rolling a sum of 7:
  - Roll two dice, check if the sum is 7.
  - Roll two dice, check if the sum is 7.
  - Roll two dice, check if the sum is 7.
  - Roll two dice, check if the sum is 7.
  - Roll two dice, check if the sum is 7.
  - …

- Monte Carlo estimate: fraction of samples where sum is 7.

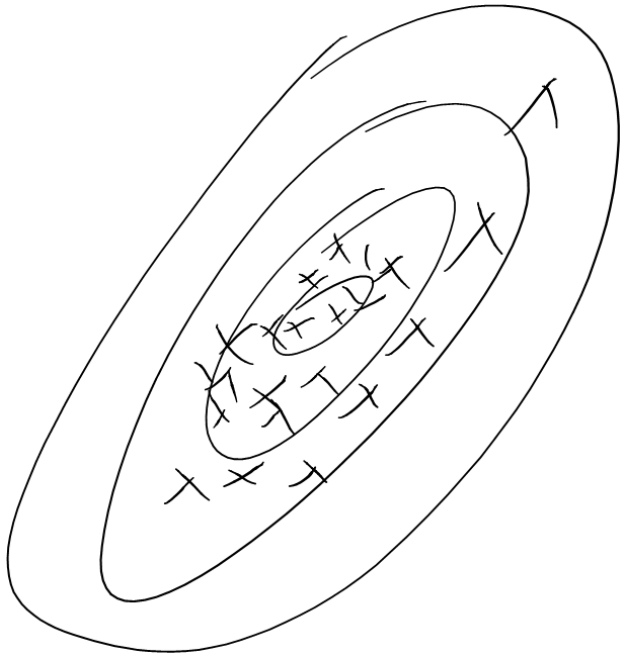
Monte Carlo Method for Inequalities

- Monte Carlo estimate of probability that variable is threshold:
  - Compute fraction of examples where sample is above threshold.



Monte Carlo Method for Mean

- A Monte Carlo approximation of the mean:
  - Approximate the mean by average of samples.

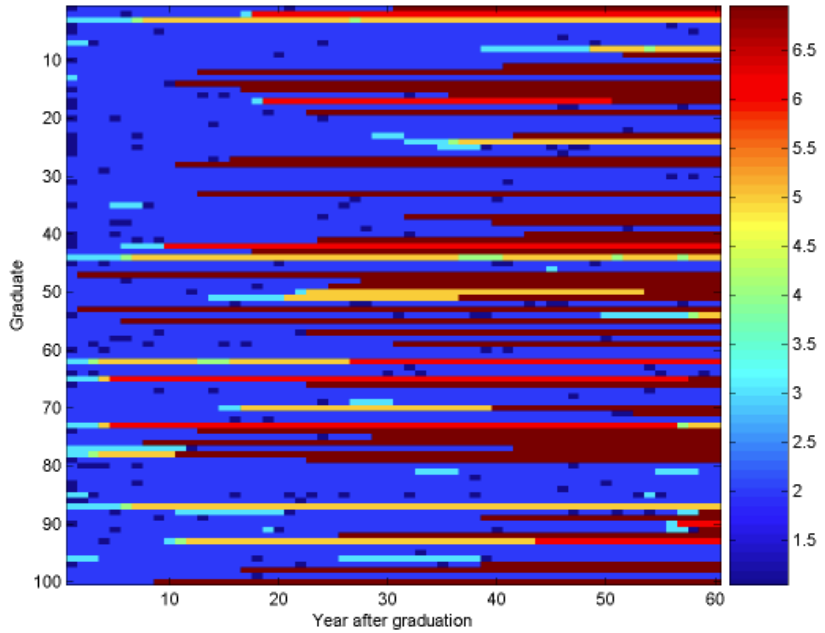


$$E[x] = \frac{1}{n} \sum_{i=1}^{n} x_i$$

- Visual demo of Monte Carlo approximation of mean and variance:

Monte Carlo *for* Markov Chains (not MCMC yet)

- Our samples from the CS grad student Markov chain:



- You can estimate probabilities by looking at frequencies in samples.
  - In how many out of 100 chains did we have x_{10} = "Industry"?
- This works for continuous states too.

Monte Carlo Methods

- Monte Carlo methods approximate expectations:

$$E[g(X)] = \underbrace{\sum_x p(x)g(x)}_{\text{discrete 'x'}} \quad \text{or} \quad E[g(X)] = \underbrace{\int_x p(x)g(x)dx}_{\text{continuous 'x'}}$$

- Computing mean is the special case of g(x) = x.
- Computing probability of any event 'A' is also a special case:
  - Set g(x) = 1 if 'A' happens and 0 if it does not happen.

- Monte Carlo method: generate 'n' IID samples x_i from p(x) and use:

$$E[g(x)] \approx \frac{1}{n}\sum_{i=1}^{n} g(x_i)$$

Monte Carlo Methods

- Monte Carlo estimate is unbiased approximation of expectation:

$$E\left[\frac{1}{n}\sum_{i=1}^{n} g(x_i)\right] = \frac{1}{n}\sum_{i=1}^{n} E[g(x_i)] = \frac{1}{n}\sum_{i=1}^{n} E[g(X)] = E[g(X)]$$

- The law of large numbers says that:
  - Unbiased approximators "converge" to expectation (in probabilistic sense).
  - So the more samples you get, the closer to the true value you start to get.

- Challenge with Monte Carlo methods:
  - It may be hard to generate IID samples.
  - This is where MCMC will fit in later.

Monte Carlo Methods for Markov Chain Inference

- Monte Carlo methods can approximate Markov chain expectations:
  - Marginal $p(x_{10}$ = "industry") is number samples with "industry" at time 10.
  - Average value at time 10, $\mathbf{E}[x_{10}]$, is approximate by average x_{10} in samples.
  - $p(x_{10} \leq 2)$ is approximated by frequency of x_{10} being less than 2.
  - $P(x_{10} \leq 2, x_{11} \geq 2)$ is approximated by frequency of both happening.

Monte Carlo for Conditional Probabilities

- We often want to compute conditional probabilities.
  - We can ask "what leads to $x_{10} = 4$?" with queries like $p(x_1 \mid x_{10} = 4)$.
  - We can ask "where does $x_{10} = 4$ lead?" with queries like $p(x_{100} \mid x_{10} = 4)$.

- Monte Carlo approach for estimating $p(x_t = s \mid x_{t'} = s')$:
  - Generate a large number of samples from the Markov chain.
  - Use Monte Carlo estimates of $p(x_t = s, x_{t'} = s')$ and $p(x_{t'} = s')$ to give:

$$p(x_t = s \mid x_{t'} = s') = \frac{p(x_t = s, x_{t'} = s')}{p(x_{t'} = s')} \approx \frac{(\text{number of samples with } x_t = s \text{ and } x_t = s')}{(\text{number of samples with } x_{t'} = s')}$$

(pause)

Last Time: Markov Chains and Monte Carlo

- Markov chains are a way to define a joint probability.

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

  - You've seen other ways to define joint probabilities, like mixture models.

- Monte Carlo is an algorithm for approximating expectations:

$$E[g(x)] \approx \frac{1}{n} \sum_{i=1}^{n} g(x_i)$$

  - You can use Monte Carlo to compute expectations in a Markov chain.
  - But Monte Carlo can be used with other probabilities, like mixture models.

Quick "Joint Probability Equations" Review

- Independence: if variables are $\{a_1, a_2, \ldots, a_n\}$ are independent then

$$p(a_1, a_2, \cdots, a_n) = p(a_1)p(a_2)\cdots p(a_n) = \prod_{i=1}^{n} p(a_i)$$

- Marginalization rule: summing/integrating out over one variable

$$p(a) = \sum_{b=1}^{K} p(a,b) \qquad p(a) = \int_b p(a,b)\,db$$

- Product rule: relates joint to conditional

$$p(a,b) = p(a|b)p(b)$$

- Bayes' rule: reverse conditionals

$$p(b|a) = \frac{p(a|b)p(b)}{p(a)} \propto p(a|b)p(b)$$

Exact Marginal Calculation

- Monte Carlo tends to converge very slowly.
  - You may need a huge number of samples.

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

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

  - We can repeat this calculation to obtain $p(x_3 = s)$ and subsequent marginals.

Exact Marginal Calculation

- Recursive formula for marginals at time 't':

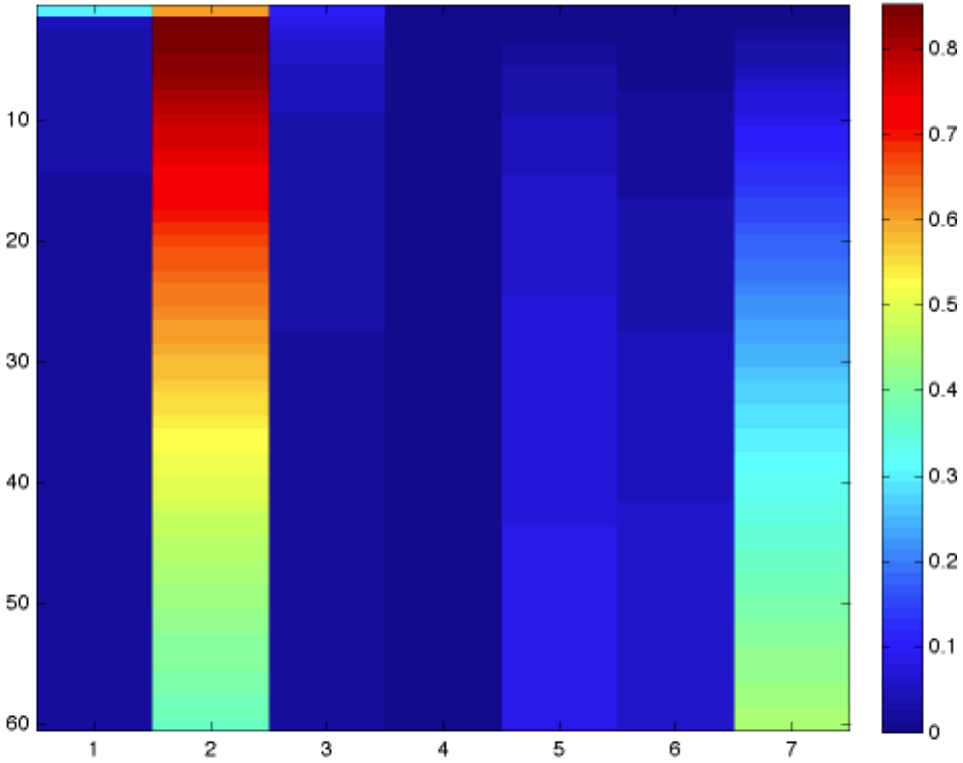
$$p(x_t = s) = \sum_{s'} p(x_t = s, x_{t-1} = s') = \sum_{s'} p(x_t = s \mid x_{t-1} = s') p(x_{t-1} = s')$$

  - Called Chapman-Kolmogorov (CK) equations.
    - There are similar equations if all probabilities are Gaussian.

- Cost:
  - Given previous time, CK equations for time 't' and state 's' cost O(k).
  - Given previous time, to compute $p(x_t = s)$ for all 's' costs $O(k^2)$.
  - So cost to compute marginals up to time 't' is $O(tk^2)$.
    - I think this is fast: there are k^t paths of length 't' that this sums over.

Marginals in CS Grad Career

- CK equations can give marginals $p(x_t = s)$ from CS grad chain:



- Each row is a year ‘t’ (first year at top), each column is a state ‘s’.

Stationary Distribution and PageRank

- A stationary distribution of a homogeneous Markov chain is a vector π satisfying:

$$\pi_s = \sum_{s'} p(x_t = s \mid x_{t-1} = s') \pi_{s'}$$

- "The probabilities don't change across time" (also called "invariant distribution").

- Under weak conditions, Markov chain marginals converge to stationary distribution.
  - $p(x_t = s)$ converges to π_s as 't' goes to ∞.
  - If we fit a Markov chain to the rain example, we have $\pi_{rain} = 0.41$.
  - In the CS grad student example, we have $\pi_{dead} = 1$.

- The PageRank is the stationary distribution of the "random surfer" Markov chain.
  - The "power method" used by Google repeatedly applies CK equations.
  - Faster than SVD which takes $O(k^3)$: iterations cost $O(z)$ where 'z' is number of links.

Uniqueness of Stationary Distribution

- Does a stationary distribution π exist and is it unique?

- Sufficient condition for existence/uniqueness is positive transitions:

$$p(x_t = s \mid x_{t-1} = s') > 0$$

  - "Damped" PageRank adds probability α of jumping to random page.

- Weaker sufficient conditions for existence/uniqueness ("ergodic"):
  - "Irreducible" (doesn't get stuck in part of the graph)
  - "Aperiodic" (probability of returning to state isn't on fixed intervals).

(pause)

Decoding in Markov Chains

- Decoding: finding the sequence with highest probability.
  - For fixed 't', find $\{x_1,x_2,\ldots,x_t\}$ that maximizes $p(x_1,x_2,..,x_t)$.

- For CS grad student (t = 60) the decoding is "industry" for all years.
  - Decoding often doesn't look like a typical sample.
  - It can also change if you increase 't'.

- Viterbi decoding is a dynamic programming algorithm.
  - Computes optimal decoding of Markov chain in $O(tk^2)$.
  - Has various applications like decoding digital TV.

Application: Voice Photoshop

- Application: Adobe VoCo uses Viterbi as part of synthesizing voices:

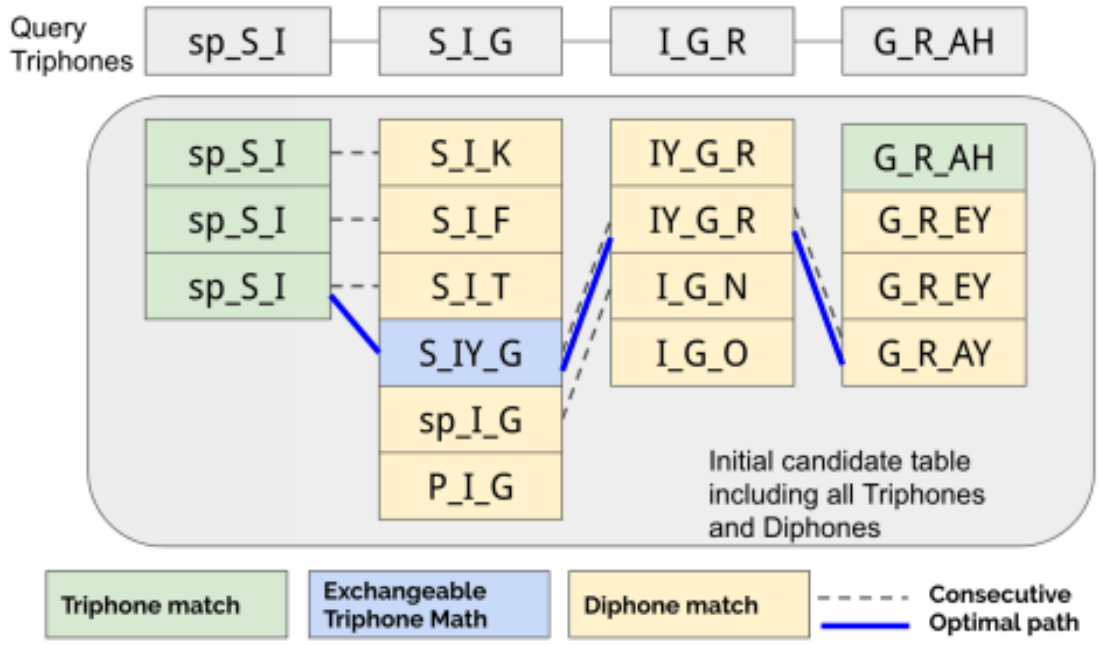


Fig. 7. Dynamic triphone preselection. For each query triphone (top) we find a candidate set of good potential matches (columns below). Good paths through this set minimize differences from the query, number and severity of breaks, and contextual mismatches between neighboring triphones.

http://gfx.cs.princeton.edu/pubs/Jin_2017_VTI/Jin2017-VoCo-paper.pdf

Summary

- Markov chains model dependency between states x_t across time.
  - Based on Markov assumption: “independence of past given last time”.
- Inverse transform can generate 1d samples.
- Ancestral sampling can generate d-dimensional samples.
- Monte Carlo methods approximate expectation using samples.
- CK equations compute exact marginals of Markov chain.
- Stationary distribution of homogeneous Markov chain (PageRank).