CHALMERS
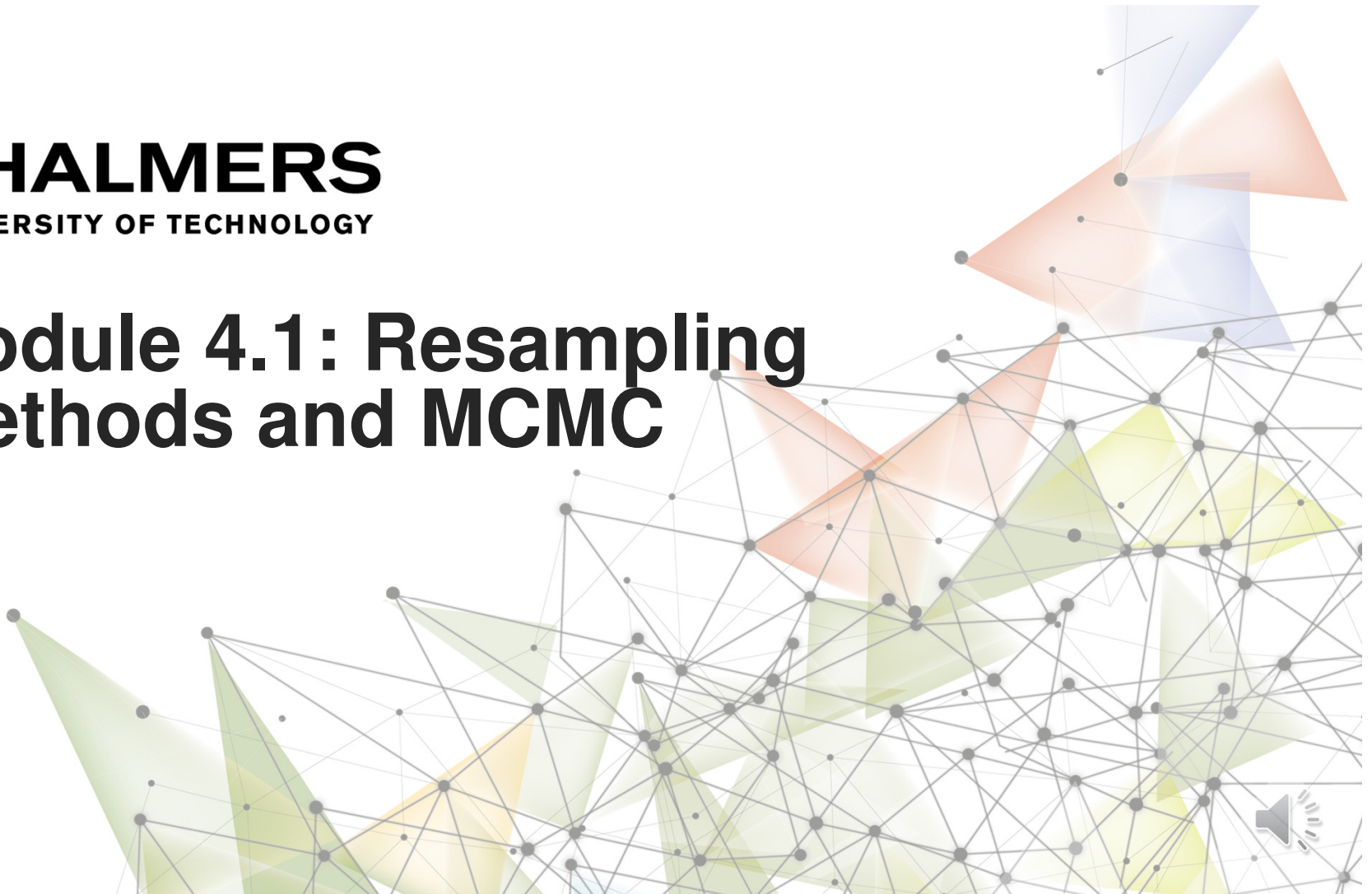### UNIVERSITY OF TECHNOLOGY

Statistical methods in Data Science and AI

Marina Axelson-Fisk

1 October, 2019

Module 4.1: Resampling methods and MCMC

Bayesian network inference

- **Approximate inference, simulation or _sampling_ is a hot topic in machine learning**

- **Basic idea:**

  - **Draw N samples from some sampling distribution $\hat{F}$**
  - **Compute an approximate posterior probability**
  - **Show that this converges to the true distribution F**

- **Why sample?**

  - **Learning: get samples from an unknonw distribution**
  - **Inference: faster than exact methods (if even possible)**

Sampling basics

- We want to make inference about about a population that is **too large** to observe completely.

- Or a distribution that is **too complex** to observe directly.

- We draw a **representative** sample (*i.i.d.* observations) and assume that the conclusions approximate those of the population/distribution.

- ***But what if the distribution is too complex to even sample from?***

Sampling statistics

Given a *sample* $X_1, X_2, \ldots, X_n$ from some distribution a *statistic* is a function $f(X_1, \ldots, X_n)$ of the sample.

Common exampels

- **Sample mean:**

$$\bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

- **Sample variance:**

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (X_i - \bar{X})^2$$

- **Sample range:**

$$r(X_1 \ldots, X_n) = \max\{X_1, \ldots, X_n\} - \min\{X_1, \ldots X_n\}$$

Simple sampling from a distribution

Inverse probability transformation

For a cumulative distribution function (cdf) F

- **Generate uniform** $U(0,1)$ **sample** $u_1, \dots, u_n$
- **Compute the inverse** F^{-1}
- **Sample from desired distribution is given by**

$$x_i = F^{-1}(u_i), i = 1, \dots, n$$

Example: sample from $\mathrm{Exp}(\lambda)$

We want to generate a sample for $X \sim \mathrm{Exp}(\lambda)$**.**
Distribution function
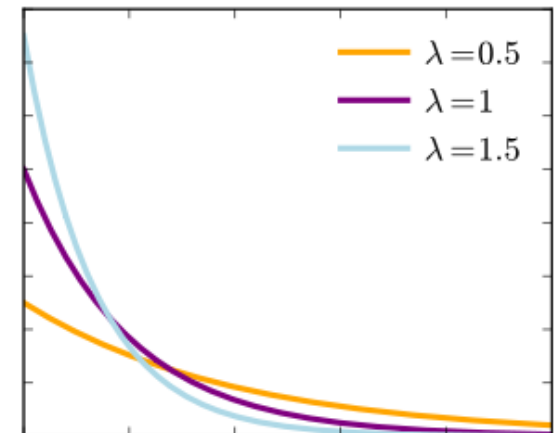
$$F(x) = 1 - e^{-\lambda x}$$

Inverse

$$y = 1 - e^{-\lambda x} \Leftrightarrow x = -\ln(1 - x)/\lambda$$

$$\Leftrightarrow F^{-1}(x) = -\ln(1 - x)/\lambda$$

Compute

$$x_i = -\ln(u_i)/\lambda$$

$$\boxed{U \sim U(0,1) \Rightarrow 1 - U \sim U(0,1)}$$

$\lambda = 0.5$
$\lambda = 1$
$\lambda = 1.5$

Resampling methods

Due to cheap rapid computing and new software *resampling methods* have become practical.

Common resampling methods

- **Permutation**: sampling without replacement to test hypotheses on the form "no effect"

- **Bootstrap**: sampling with replacement to establish more precise confidence intervals

- **Monte Carlo**: repeated sampling from populations with known characteristics to determine the sensitivity to those characteristics

Resampling methods

Advantages over both nonparametric and parametric methods:

- **Simpler**
- **More accurate**
- **Fewer assumptions**
- **Greater generalizability**
- **Answer questions not possible with traditional methods**
- **Conceptually simple**

Empirical bootstrap

Example

Given a sample $X_1, \ldots, X_n$ we estimate the mean μ and variance σ^2 by the sample mean and variance

$$\hat{\mu} = \bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

$$\hat{\sigma}^2 = s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (X_i - \bar{X})^2$$

A $(1 - \alpha)\%$ confidence interval for μ is then

$$\mu = \bar{x} \pm z_{1-\alpha/2} \frac{s}{\sqrt{n}}$$

Can we construct a confidence interval for the sample *median* in a similar way?
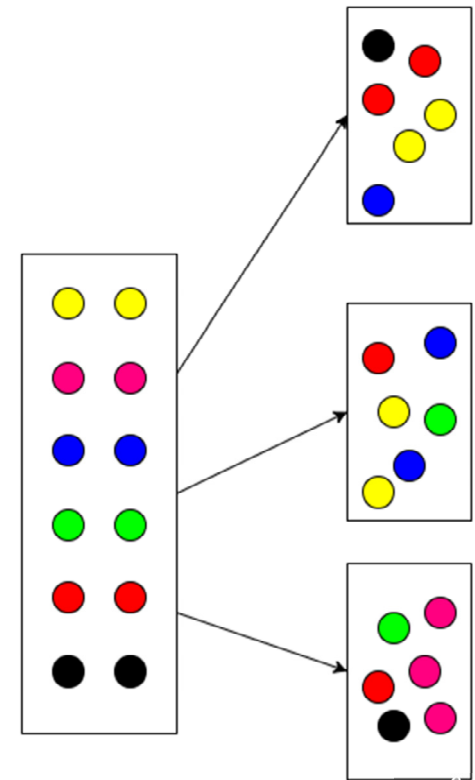
Empirical bootstrap

- **Given the sample** $X_1, \ldots, X_n$ **we** *sample with replacement* **from these** n **points.**

- **Resampled sample:** $X_1^{*(1)}, \ldots, X_n^{*(1)}$

- **Repeat this, creating** B ***bootstrap samples***

$$X_1^{*(1)}, \ldots, X_n^{*(1)}$$
$$X_1^{*(2)}, \ldots, X_n^{*(2)}$$
$$\ldots$$
$$X_1^{*(B)}, \ldots, X_n^{*(B)}$$

Empirical bootstrap
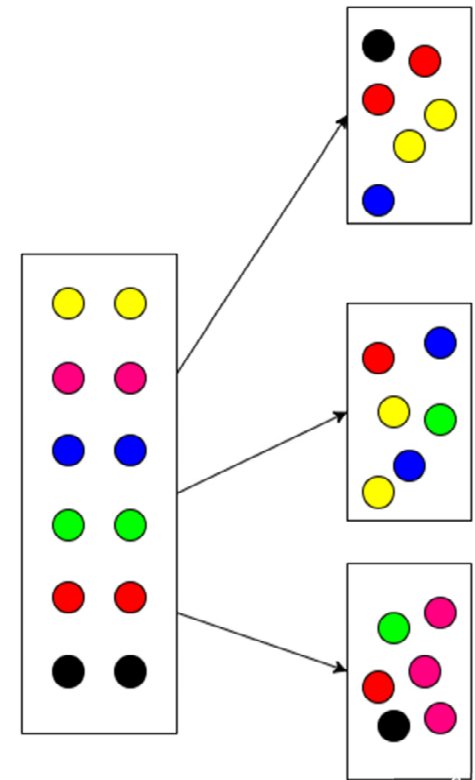
- **For each sample, compute the sample median**

$$M_n^{*(1)}, M_n^{*(2)}, \ldots, M_n^{*(B)}$$

- **Estimate the median using:** $\bar{M}_B = \frac{1}{B}\sum_{i=1}^b M_n^{*(i)}$

- **Boostrap estimate of the median variance:**

$$\widehat{\text{Var}}_B(M_n) = \frac{1}{B-1}\sum_{i=1}^{B}\left(M_n^{*(i)} - \bar{M}_B^*\right)^2$$
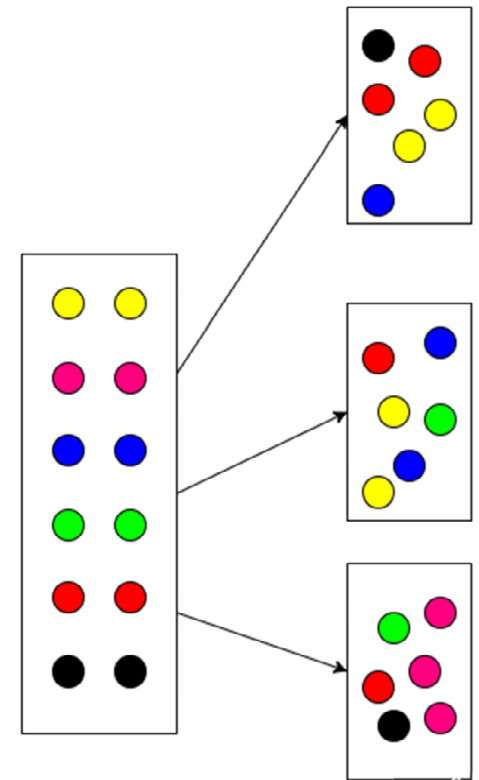
- **Bootstrap confidence interval of the median**

$$M_n \pm z_{1-\alpha/2} \cdot \sqrt{\widehat{\text{Var}}_B(M_n)}$$

Empirical bootstrap

- **Can be generalized to many other statistics**

  - **Sample quantiles**

  - **Interquartile range**

  - **Skewness (related to $E[X^3]$)**

  - **Kurtosis (related to $E[X^4]$)**

  - **…**

Monte Carlo simulation

- **We want to evaluate the integral**

$$F(x) = \int_0^1 e^{-x^3} dx$$

No closed form solution!

- *Riemann integration*:

Evaluate $f(x) = \exp(-x^3)$ **in evenly spread points**
$0 \leq x_1 < x_x < \cdots < x_K \leq 1$, **and compute**

$$\sum_{i=1}^{K} f(x_i) / K$$

- **Converges as** $K \rightarrow \infty$.

Monte Carlo simulation

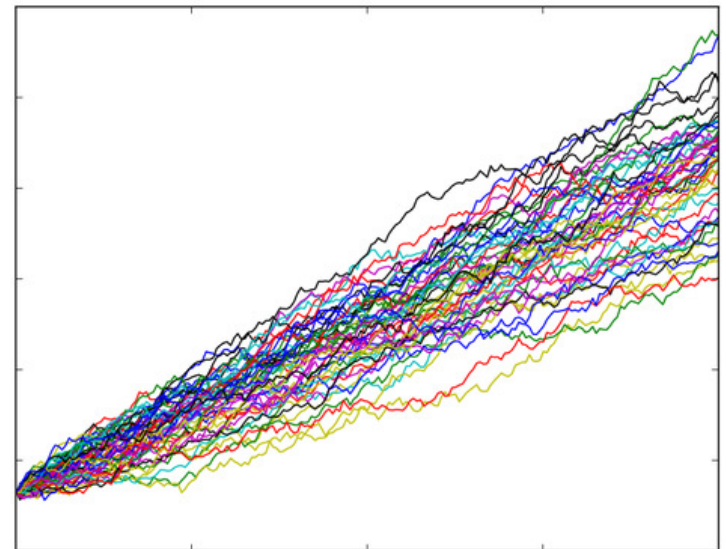- **We want to evaluate an integral**

$$I = \int f(x)dx$$

- **We choose a pdf $p(x)$ of X and compute**

$$I = \int f(x)p(x)dx = E[f(X)]$$

- **Generate i.i.d. $X_1, \dots, X_n \sim p$ and compute**

$$\hat{I}_n = \frac{1}{n}\sum_{i=1}^{n} f(X_i)$$

Monte Carlo simulation – example
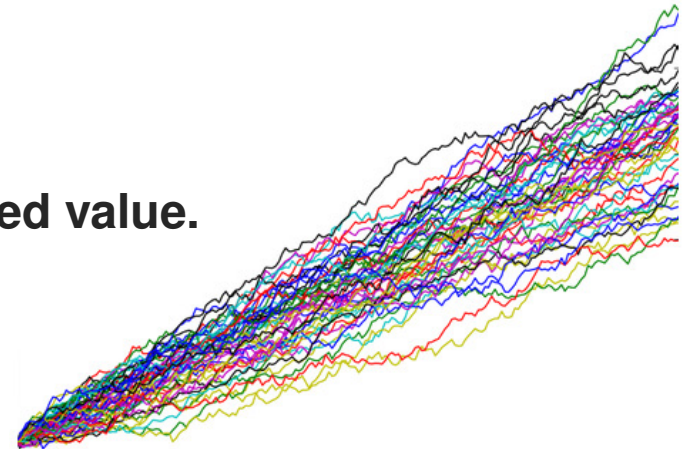
- **We want to compute the integral**

$$I = \int_0^1 e^{-x^3} dx$$

- **Let $U \sim U[0,1]$ and write**

$$I = \int_0^1 e^{-x^3} 1 \, dx = E\left[e^{-U^3}\right]$$

evaluating integral $\Leftrightarrow$ estimating the expected value.
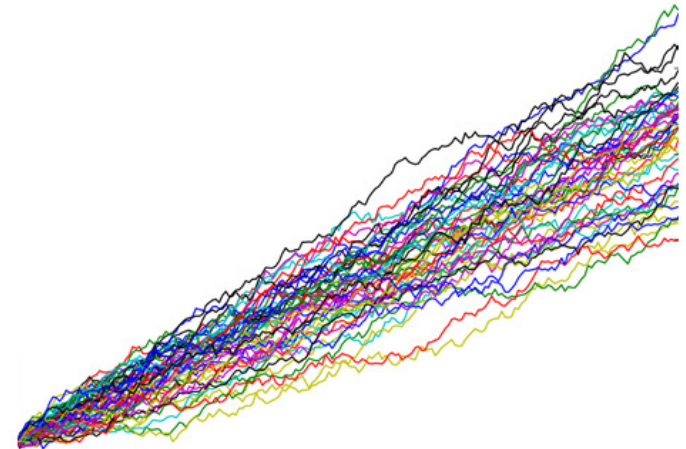
$$E\left[e^{-U^3}\right]$$

Monte Carlo simulation – example

- **Generate i.i.d. sample** $U_1, \dots, U_n \sim U[0,1]$**.**

- **Compute** $W_i = e^{-U_i^3}$ **and use**

$$\overline{W} = \frac{1}{n}\sum_{i=1}^{n} W_i = \frac{1}{n}\sum_{i=1}^{n} e^{-U_i^3}$$

- **Law of large numbers:** $\overline{W} \xrightarrow{P} E[W_i] = E\left[e^{-U^3}\right]$
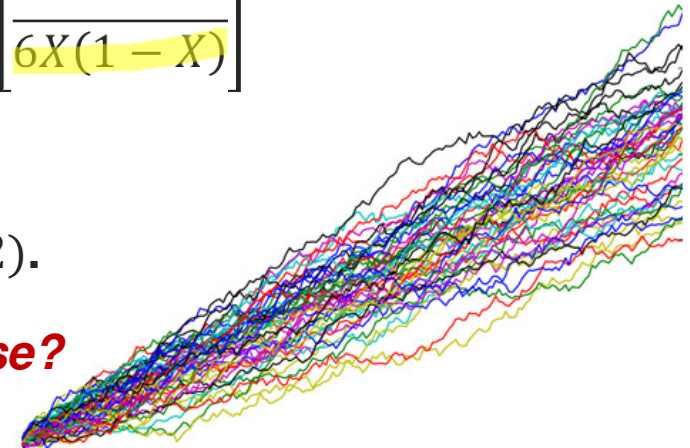
Monte Carlo simulation – example

- **Note that the function f changes if the pdf p changes:**

- **Since $\text{Beta}(2,2)$ has pdf $p(x) = 6x(1-x)$ we could write**

$$\int_0^1 e^{-x^3} dx = \int_0^1 \frac{e^{-x^3}}{6x(1-x)} \cdot 6x(1-x) dx = E\left[\frac{e^{-X^3}}{6X(1-X)}\right]$$

  where now $X \sim \text{Beta}(2,2)$.

- **Thus we can sample i.i.d. $X_1, \ldots, X_n \sim \text{Beta}(2,2)$.**

So which sampling distribution p should we use?

Monte Carlo simulation: bias and variance

- **Bias:**

$$E[\hat{I}_n] = I \Rightarrow \mathbf{bias} = 0$$

<div style="border: 1px solid red;">

Unbiased!

</div>

- **Variance**

$$\text{Var}(\hat{I}_n) = \frac{1}{n} \text{Var}(f(X_1))$$
$$= \frac{1}{n} (E[f^2(X_1)] - (E[f(X_1)])^2)$$
$$= \frac{1}{n} \left(\int f^2(x) p(x) dx - I^2 \right)$$

<div style="border: 1px solid red;">

The variance depends on the sample size n and sampling distribution p.
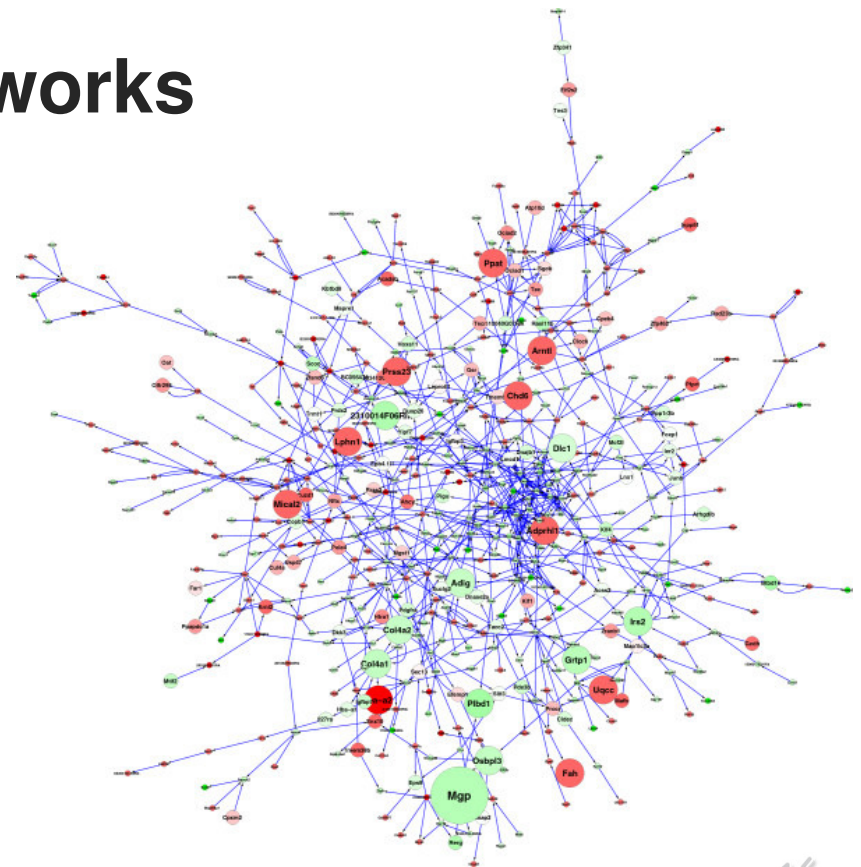
</div>

The quantity I^2 is fixed.

We choose a density p and a function f that minimizes the variance.

<div style="border: 1px solid red;">

How?

</div>

Sampling in Bayesian networks

- **Prior sampling**

- **Rejection sampling**

- **Likelihood weighting**

- **Gibbs sampling**

Prior sampling

- In a network of n nodes $X_1, \ldots, X_n$ a sample is generated by drawing in hierarchical order from
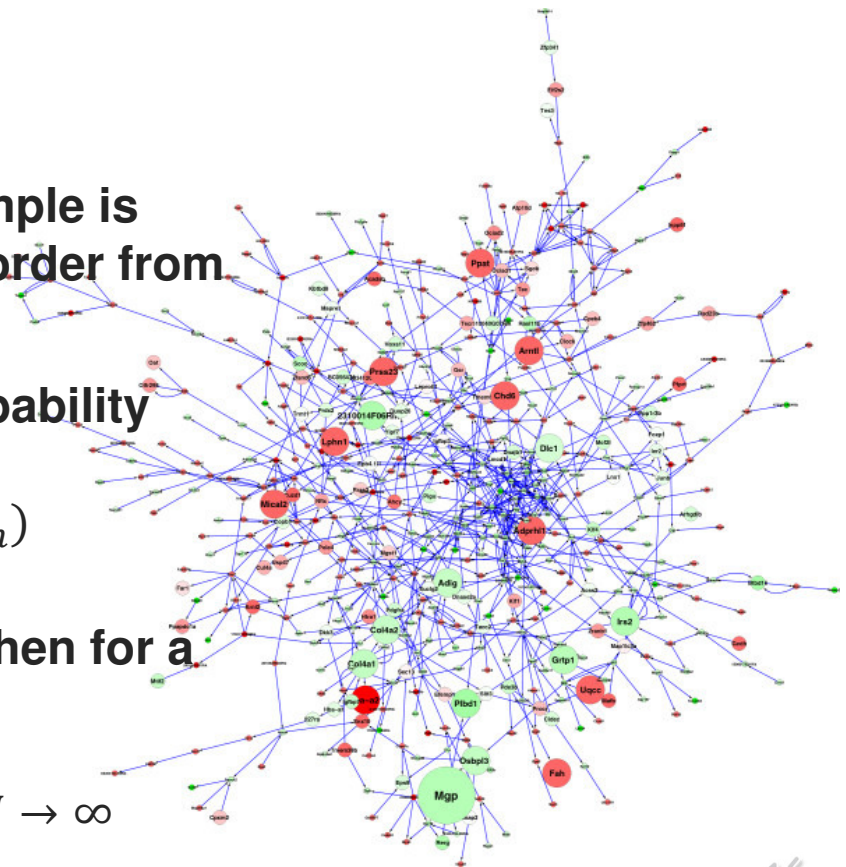$$P(X_i | \mathrm{pa}(X_i)), i = 1, \ldots, n$$

- Then samples are generated with probability
$$\prod_{i=1}^{n} P(X_i | \mathrm{pa}(X_i)) = P(X_1, \ldots, X_n)$$

- Repeat the process many times (N). Then for a certain event $(x_1, \ldots, x_n)$
$$\frac{\#\{(x_1, \ldots, x_n)\}}{N} \to P(x_1, \ldots, x_n) \text{ as } N \to \infty$$

Prior sampling: example

- Assume that we sample a Bayesian network with *true/false* variables A, B, C, D
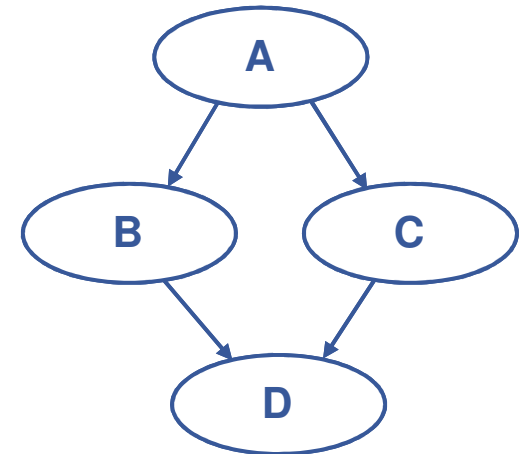
$$+a, -b, +c, +d$$
$$+a, +b, +c, +d$$
$$-a, +b, +c, -d$$
$$+a, -b, +c, +d$$
$$-a, -b, -c, +d$$

- To compute the probability $P(D)$ we see

$$\#\{+d\} = 4, \#\{-d\} = 1$$

- Thus: $\widehat{P}(D = +d) = 4/5 = 0.80 = 1 - \widehat{P}(D = -d)$

Rejection sampling (rejection/acceptance)

Assume we want to sample from some distribution $p(x)$

1. **Create a rejection distribution $q(x)$ that is easy to sample from and which satisfies**

$$Mq(x) \geq \tilde{p}(x)$$ **envelope**

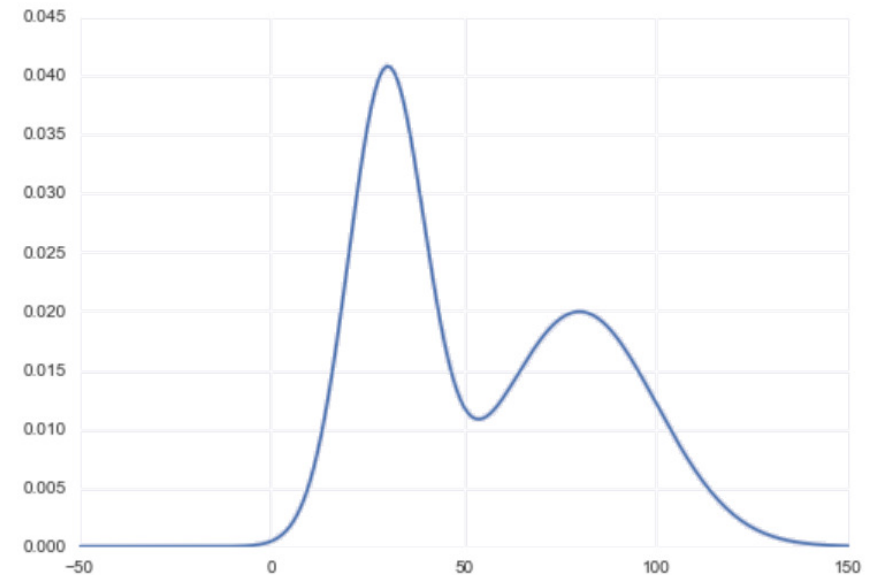**constant M, $\tilde{p}(x)$ = unnormalized $p(x)$**

$$p(x) = \tilde{p}(x)/Z_p \Leftrightarrow \tilde{p}(x) = p(x)Z_p$$

2. **Sample a random location $x \sim q(x)$ and a random height $u \sim U(0,1)$.**

3. **If $u \leq \dfrac{\tilde{p}(x)}{Mq(x)}$ keep the observation in sample.**

Example: rejection sampling

We want to sample from a Gaussian mixture

$$X + Y \sim N(30,10) + N(80,20)$$
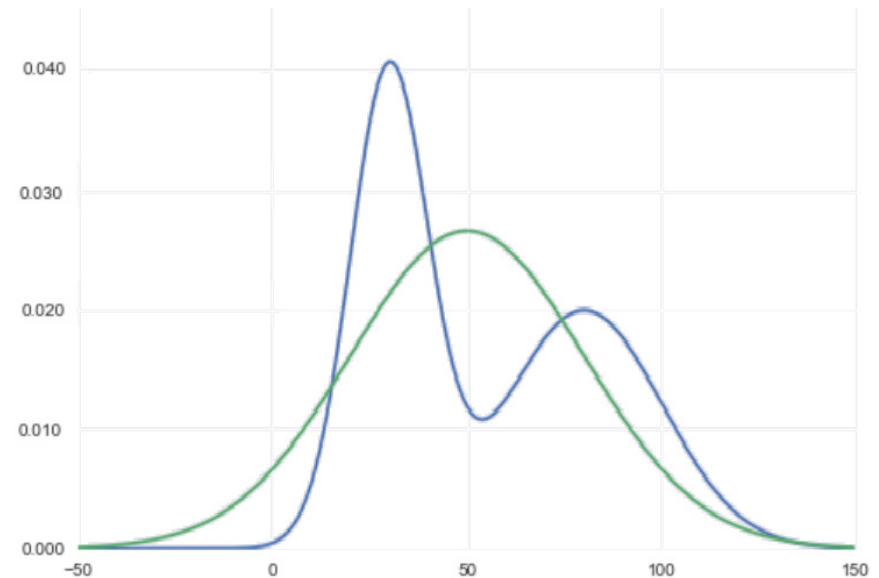
Example: rejection sampling

We want to sample from a Gaussian mixture

$$X + Y \sim N(30,10) + N(80,20)$$

Use $q(x) = N(50,30)$ as proposal distribution. (Unnormalized)

Bad choice since $q(x)$ is *not* enveloping $p(x)$.

Example: rejection sampling

We want to sample from a Gaussian mixture

$$X + Y \sim N(30,10) + N(80,20)$$

Use $q(x) = N(50,30)$ **as proposal distribution. (Unnormalized)**

Bad choice since $q(x)$ **is *not* enveloping** $p(x)$.

Add on a scaling factor

$$k = \max\big(p(x)/q(x)\big) \text{ for all } x$$
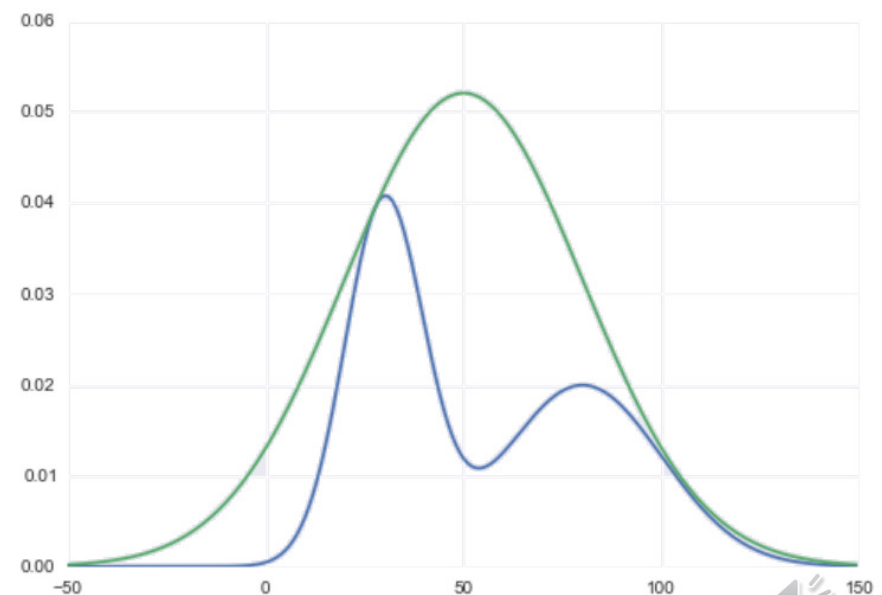
Example: rejection sampling

- **Produce a large sample from**

$$Z \sim q(x)$$

- **Uniformly pick the height**

$$u \sim U\big(0, kq(z)\big)$$

Now (z, u) **is uniform under**

$$kq(x)$$

- **Accept points** (z, u) **under the** $p(x)$ **curve.**
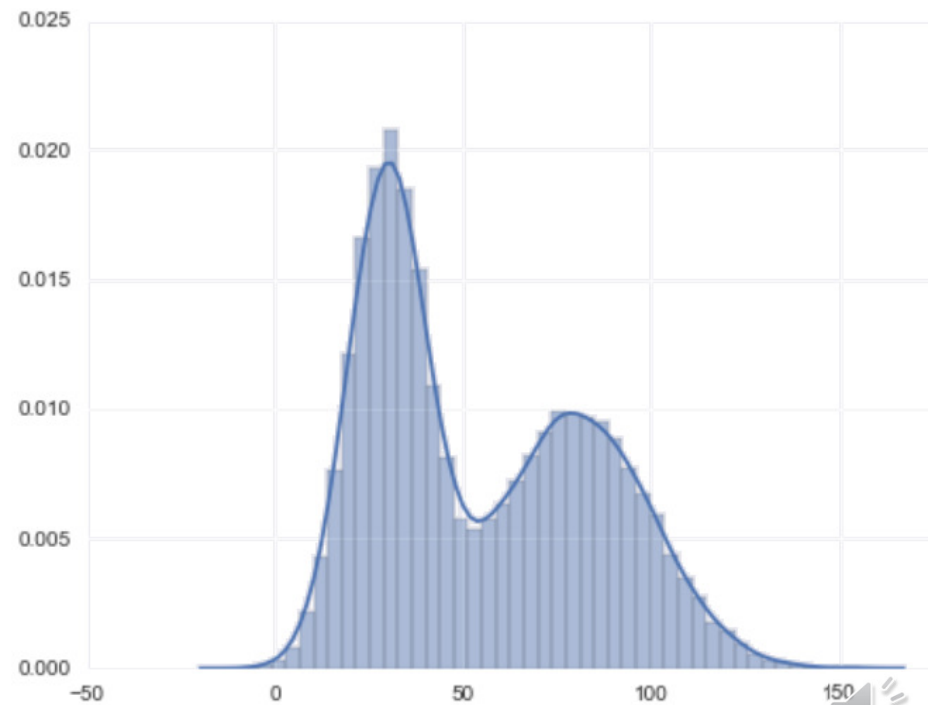
Rejection sampling

- **Unnormalized target distribution – no problem.**
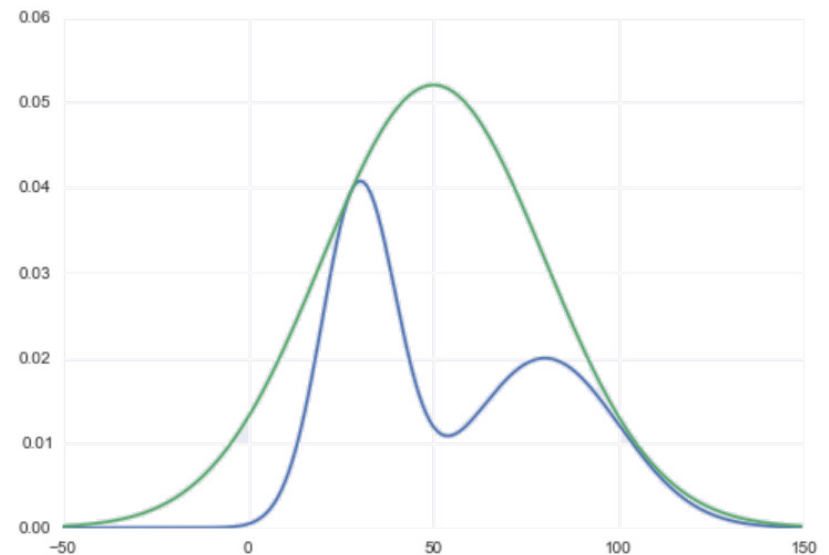
- **Drawbacks:**

    - **Need a closed form for q.**
    - **$q(x)$ has to envelope $p(x)$.**
    - **Need to know shape of $p(x)$.**
    - **Large M is time consuming.**

Rejection sampling in Bayesian statistics

- **Data set $\mathcal{D}$ and target distribution with parameter $\boldsymbol{\theta}$.**

- **We want to sample from the posterior**

$$p(\boldsymbol{\theta}|\mathcal{D}) = p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})/p(\mathcal{D})$$

- **We can use unnormalized target $\tilde{p}(\boldsymbol{\theta}) = p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})$**
  proposal distribution $q(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$
  and envelope $M = p(\mathcal{D}|\widehat{\boldsymbol{\theta}})$ where $\widehat{\boldsymbol{\theta}} = \arg\max p(\mathcal{D}|\boldsymbol{\theta})$,
  i.e. the MLE

- **Accept sampled points with probability**

$$P = \frac{p(\boldsymbol{\theta}|\mathcal{D})}{Mq(\boldsymbol{\theta})}$$
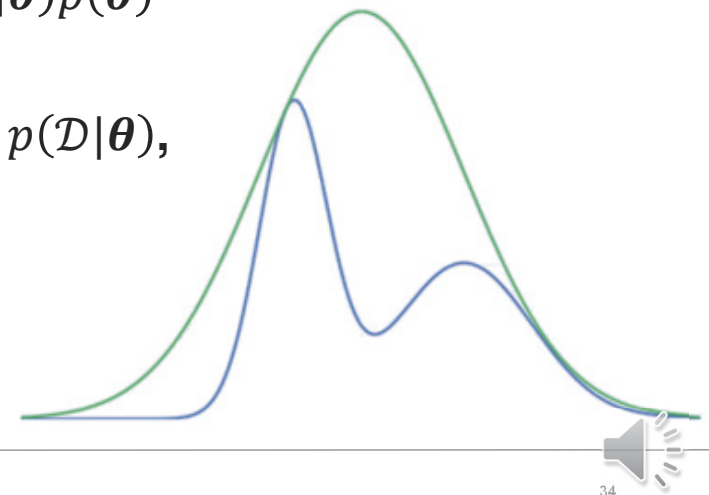
Sampling in Bayesian networks: example

We use an AI to diagnose the true cause of a fire alarm.

Variabels (all true/false):

- *Fire*: true when there is a fire
- *Alarm*: true when the alarm sounds
- *Smoke*: true when there is smoke
- *Leaving*: true if many people leave the building
- *Report*: true if reports of people leaving
- *Tampering*: true when alarm were tampered with
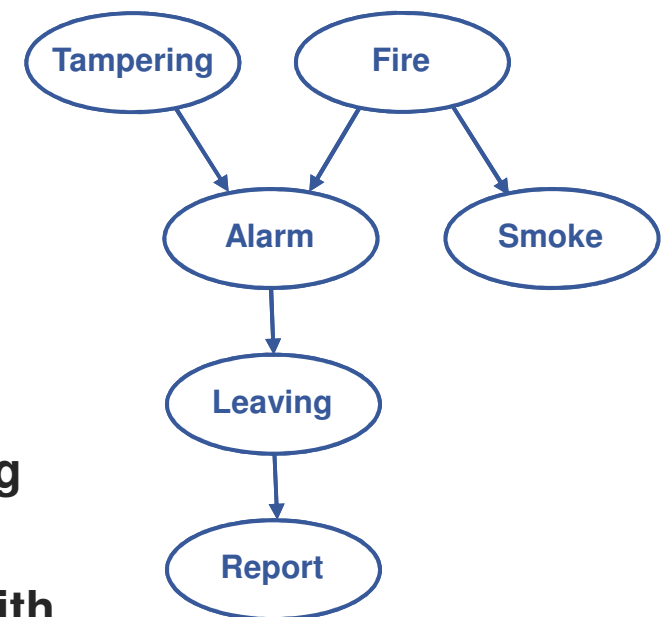
Conditional dependencies are given by the DAG.

Sampling in Bayesian networks: example

Factorization

$P(\text{tampering, fire, alarm, smoke, leaving, report}) =$
$\quad = P(\text{tampering})$
$\qquad \cdot P(\text{fire})$
$\qquad \cdot P(\text{alarm} \mid \text{tampering, fire})$
$\qquad \cdot P(\text{smoke} \mid \text{fire})$
$\qquad \cdot P(\text{leaving} \mid \text{alarm}) \cdot$
$\qquad \cdot P(\text{report} \mid \text{leaving})$

Sampling in Bayesian networks: example

$P(T)$
0.02

$P(F)$
0.01

T   **F**

F	T	$P(A\|F,T)$
true	true	0.5
true	false	0.99
false	true	0.85
false	false	0.0001

A   **S**

F	$P(S\|F)$
true	0.9
false	0.01

L

A	$P(L\|A)$
true	0.88
false	0.001

L	$P(R\|L)$
true	0.75
false	0.01

R
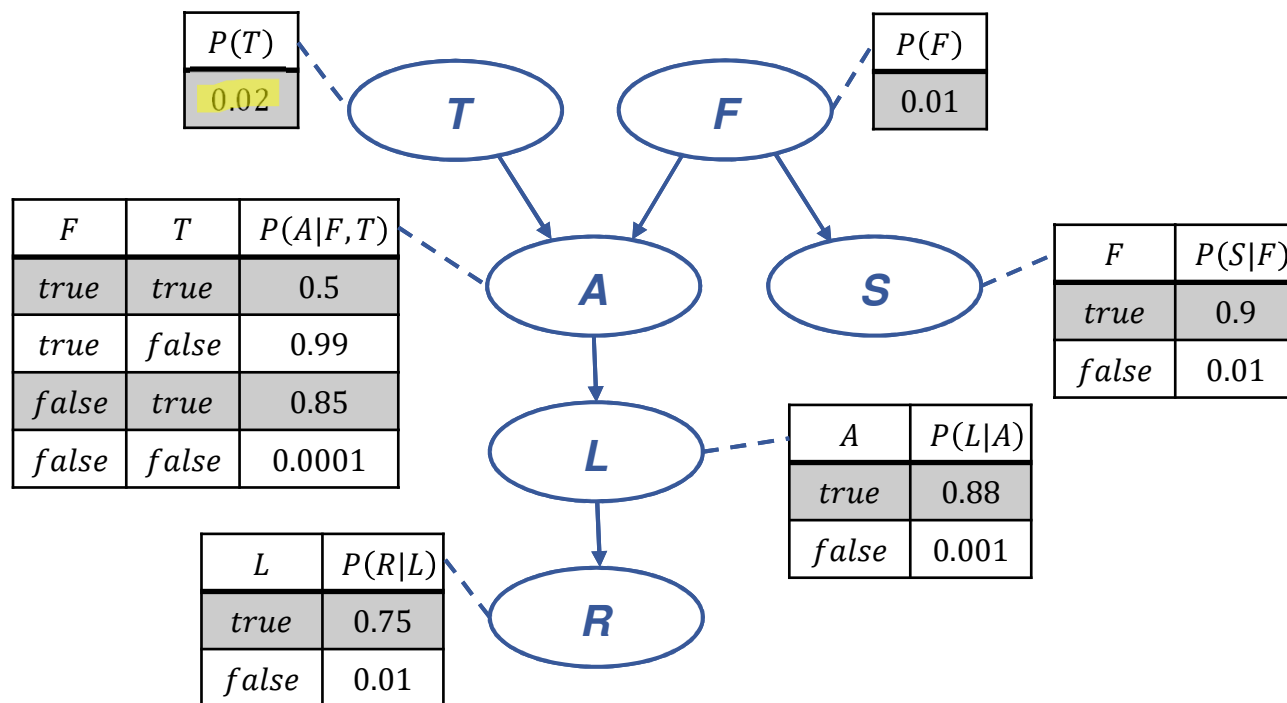
Forward sampling in Bayesian networks

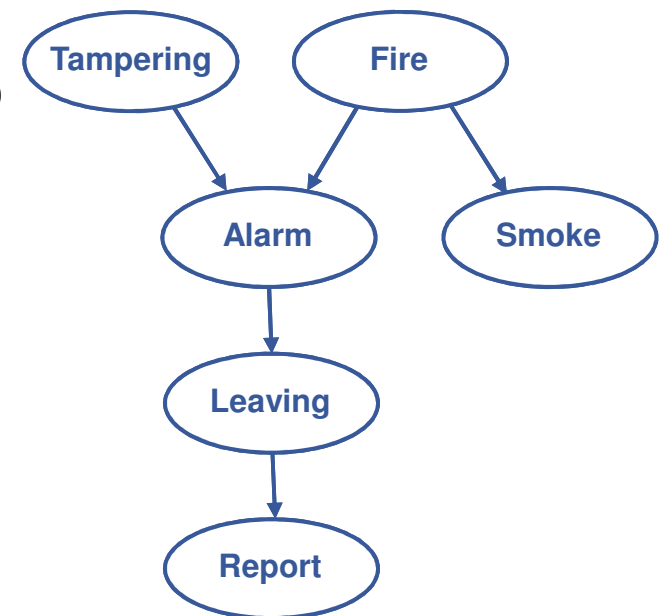A method to generate a sample of every variable so that each sample is generated in proportion to its probability.

Forward sampling

Assume variables $X_1, X_2, \ldots, X_n$ are network variables ordered so parents come before children.

1. Sample X_1 using its CDF
2. Sample X_2 given the value of X_1
3. …

Tampering
Fire
Alarm
Smoke
Leaving
Report

Forward sampling in Bayesian networks

Sampling from a single variable

**Assume variable X takes values $\{v_1, v_2, v_3, v_4\}$.
Assume**

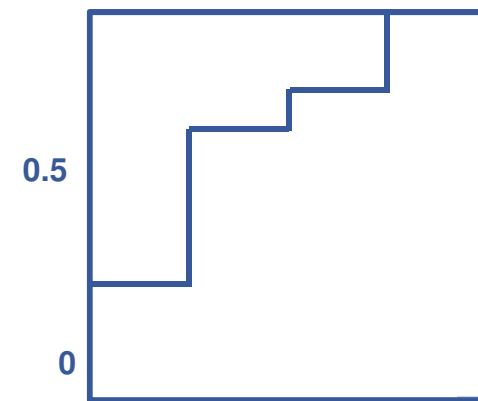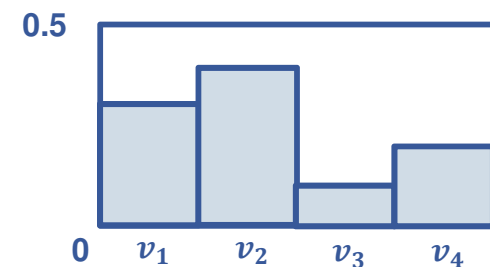$$P(X = v_1) = 0.3$$
$$P(X = v_2) = 0.4$$
$$P(X = v_3) = 0.1$$
$$P(X = v_4) = 0.2$$

Order the values, say $v_1 < v_2 < v_3 < v_4$.

Generate numbers from $u \sim U(0,1)$:

If $v_2 < u < v_3$, then v_3 etc.

Sampling in Bayesian networks: example

Forward sampling:

Sample	T	F	A	S	L	R
s_1	false	false	true	false	false	true
s_2	false	true	false	true	false	false
s_3	false	true	true	false	true	true
s_4	false	true	false	true	false	false
s_5	false	true	true	true	true	true
s_6	false	false	false	true	false	false
s_7	true	false	false	false	true	false
s_8	true	true	true	true	true	true
...						
s_{1000}	true	false	true	false	true	false

Sample from Tampering:
$$P(T) = 0.02, P(T^c) = 0.98$$
Draw from $U(0, 1) \Rightarrow u = 0.37$
$$\Rightarrow T = false$$

Sample from Fire:
$$P(F) = 0.01, P(F^c) = 0.99$$
$$\Rightarrow F = true$$

Sample from Alarm given parents
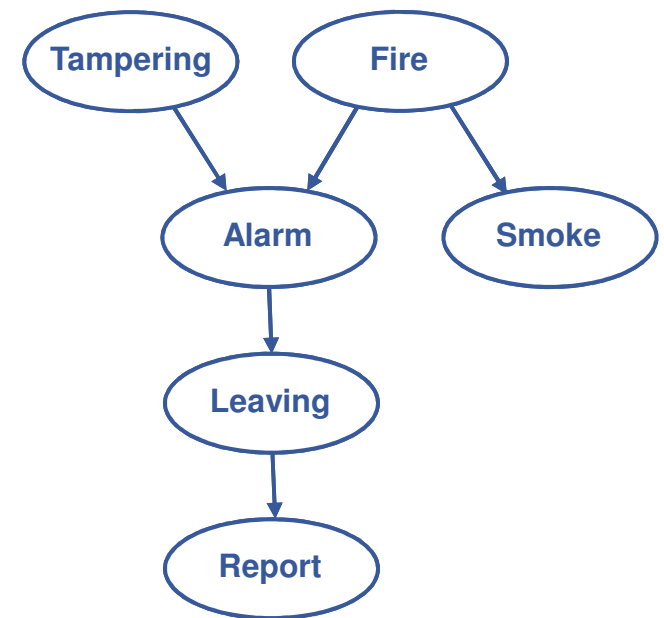$$P(A|T^c, F) = 0.99,$$
$$P(A^c|T^c, F) = 0.01$$
$$\Rightarrow A = true$$

etc.

Rejection sampling in Bayesian networks

Given some data D rejection sampling estimates the posterior

$$P(Y|D) = \frac{P(D|Y)P(Y)}{P(D)}$$

Sampling in Bayesian networks: example

Rejection sampling for $P(T|S, R^c)$

Sample	T	F	A	S	L	R	
s_1	false	false	true	false	X		
s_2	false	true	false	true	false	false	✓
s_3	false	true	true	false	X		
s_4	false	true	false	true	false	false	✓
s_5	false	true	true	true	true	true	X
s_6	false	false	false	true	false	false	✓
s_7	true	false	false	false	X		
s_8	true	true	true	true	true	true	X
...							
s_{1000}	true	false	true	false	X		

Sampling in Bayesian networks: example

Rejection sampling for $P(T|S, R^c)$

Sample	T	F	A	S	L	R	
s_1	false	false	true	false	**X**		
s_2	false	true	false	true	false	false	✓
s_3	false	true	true	false	**X**		
s_4	false	true	false	true	false	false	✓
s_5	false	true	true	true	true	true	**X**
s_6	false	false	false	true	false	false	✓
s_7	true	false	false	false	**X**		
s_8	true	true	true	true	true	true	**X**
...							
s_{1000}	true	false	true	false	**X**		

The probability $P(T|S, R^c)$ **is estimated using observed proportions:**

$$\hat{P}(T|S, R^c) = \frac{\#\{\text{accepted}, T\}}{\#\{\text{accepted}\}}$$

Sampling in Bayesian networks: example

Downside: Rejection sampling requires MANY samples

Sample	T	F	A	S	L	R
s_1	false	false	true	false	false	true
s_2	false	true	false	true	false	false
s_3	false	true	true	false	true	true
s_4	false	true	false	true	false	false
s_5	false	true	true	true	true	true
s_6	false	false	false	true	false	false
s_7	true	false	false	false	true	false
s_8	true	true	true	true	true	true
...						
s_{1000}	true	false	true	false	true	false

Example: estimate $P(T|S,F)$

Recall:

$P(F) = 0.01$
$P(S|F) = 0.9$
$P(S|F^c) = 0.01$

Out of 1000 samples:
$\approx$ **990** $F = false$
Of these 990
$\approx$ **1% = 10** $S = true$
$\Rightarrow$ **remaining 980 (98%) rejected**

Likelihood weighting in Bayesian networks

- **A kind of importance sampling where instead of accepting/rejecting a sample, we weigh its contribution to the sample.**

- **Instead of generating samples until enough many has been accepted:**

  - **Generate only samples that agree with evidence (what we condition on)**

  - **Weigh samples by their likelihood**

  - **Combine the corresponding weights in the final probability estimates.**
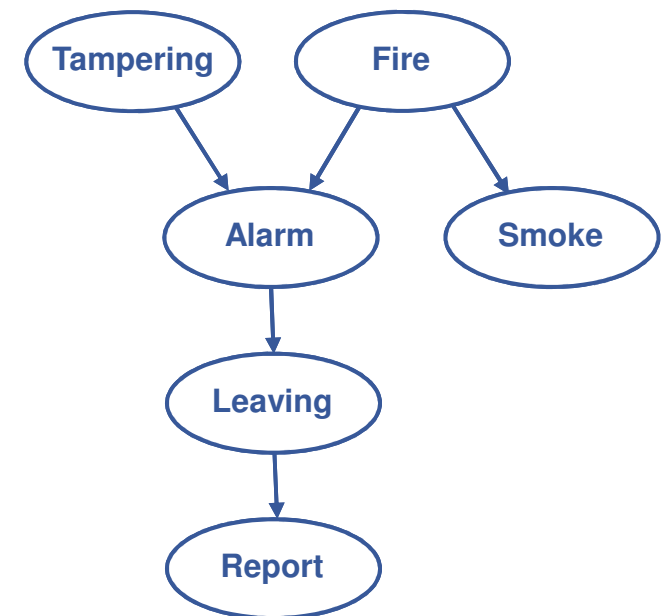
Likelihood weighting in Bayesian networks

If we wanted to estimate $P(\mathbf{X}|\mathbf{Y})$:

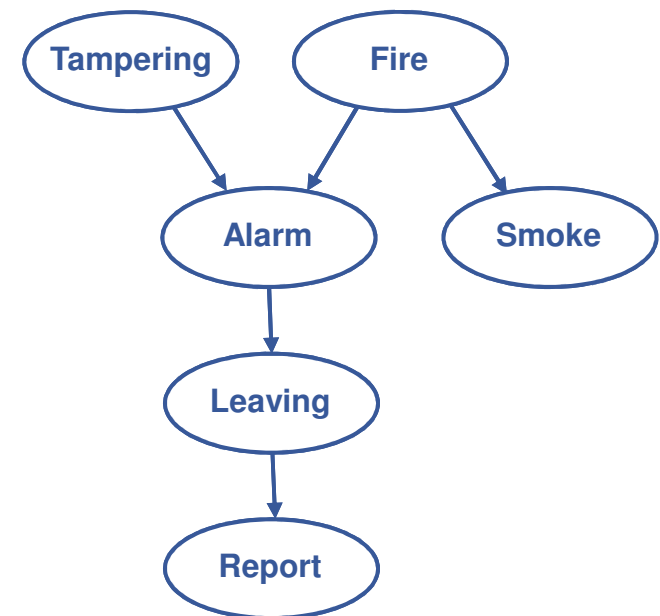- **Compute the sampling distribution of X with Y fixed**

$$S_{LW}(\mathbf{x}, \mathbf{y}) = \prod_{X_i \in \mathbf{X}} P(X_i | \mathrm{pa}(X_i))$$
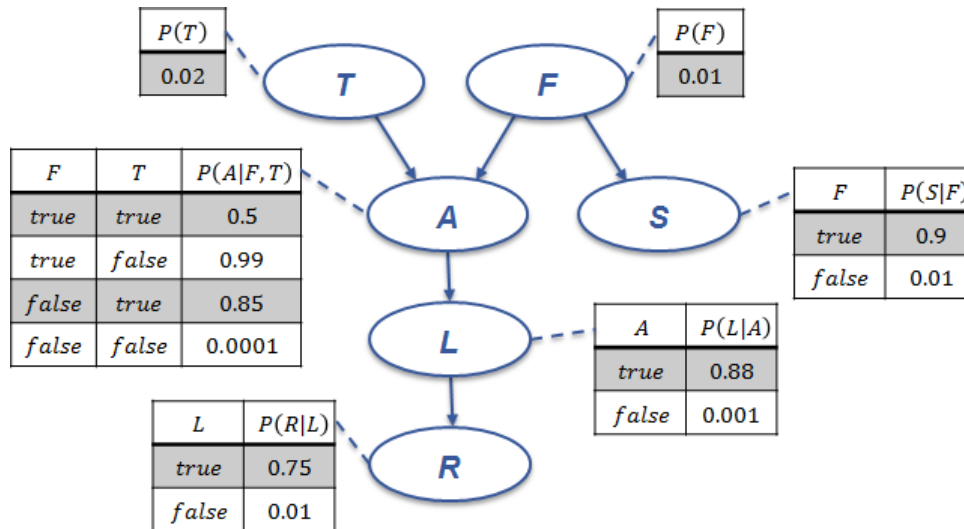
- **Compute sample weights**

$$w(\mathbf{x}, \mathbf{y}) = \prod_{Y_i \in \mathbf{Y}} P(Y_i | \mathrm{pa}(Y_i))$$

- **We get**

$$\hat{P}(\mathbf{X} = \mathbf{x}, \mathbf{Y} = \mathbf{y}) = S_{LW}(\mathbf{x}, \mathbf{y})w(\mathbf{x}, \mathbf{y})$$
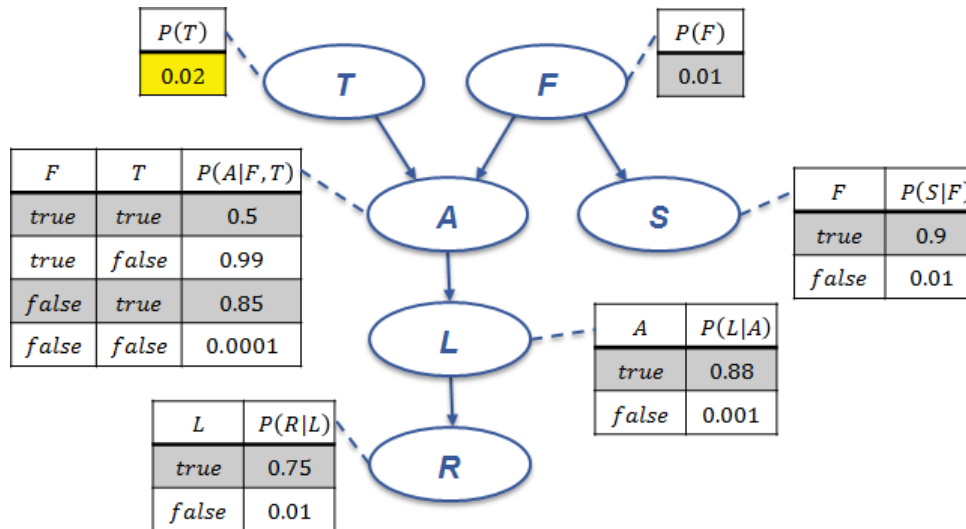
Likelihood weighting in Bayesian networks

P(T)
0.02

P(F)
0.01

F	T	P(A|F,T)
true	true	0.5
true	false	0.99
false	true	0.85
false	false	0.0001

F	P(S|F)
true	0.9
false	0.01

A	P(L|A)
true	0.88
false	0.001

L	P(R|L)
true	0.75
false	0.01

Example: estimate $P(T|S, R^c)$

Variables $S = true$ and $R = false$ are fixed *evidence*

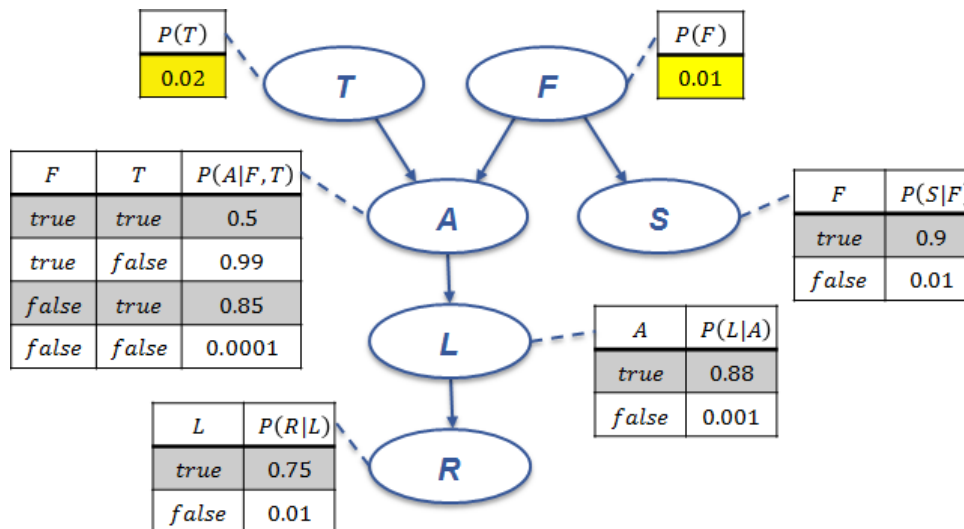**Generate sample:**

Likelihood weighting in Bayesian networks

P(T)
0.02

T

F

P(F)
0.01

F	T	P(A|F,T)
true	true	0.5
true	false	0.99
false	true	0.85
false	false	0.0001

A

S

F	P(S|F)
true	0.9
false	0.01

L

A	P(L|A)
true	0.88
false	0.001

L	P(R|L)
true	0.75
false	0.01

R

Example: estimate $P(T|S, R^c)$

Variables $S = true$ **and** $R = false$ **are fixed** *evidence*

Generate sample:

1. **Sample** T**: e.g** $T = false$

Likelihood weighting in Bayesian networks

Example: estimate $P(T|S, R^c)$

Variables $S = true$ **and** $R = false$ **are fixed** *evidence*

Generate sample:

1. **Sample** T**: e.g** $T = false$
2. **Sample** F**: e.g.** $F = true$

Likelihood weighting in Bayesian networks
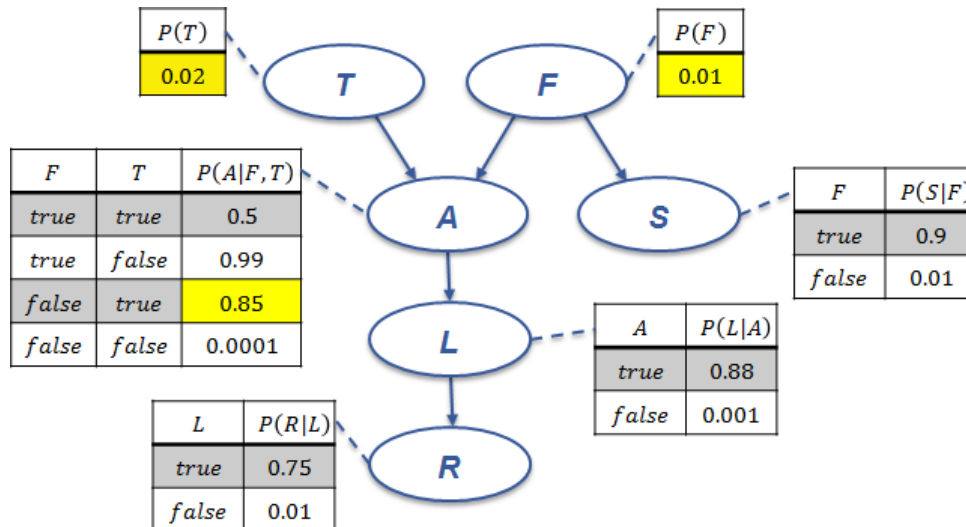
P(T)
0.02

P(F)
0.01

F	T	P(A|F,T)
true	true	0.5
true	false	0.99
false	true	0.85
false	false	0.0001

F	P(S|F)
true	0.9
false	0.01

A	P(L|A)
true	0.88
false	0.001

L	P(R|L)
true	0.75
false	0.01

Example: estimate $P(T|S, R^c)$

Variables $S = true$ **and** $R = false$ **are fixed** *evidence*

Generate sample:

1. **Sample** T: **e.g** $T = false$
2. **Sample** F: **e.g.** $F = true$
3. **Sample** $A|F, T^c$: **e.g.** $A = true$

Likelihood weighting in Bayesian networks
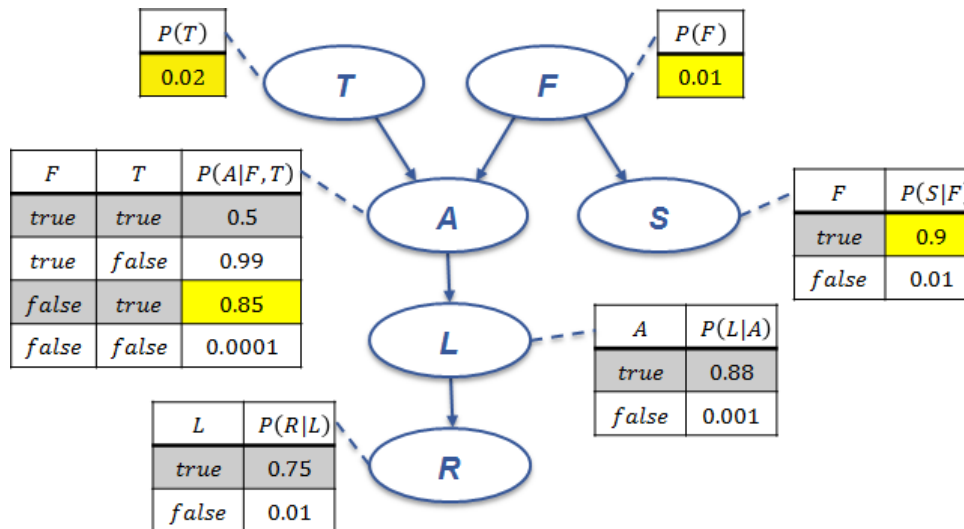
Example: estimate $P(T|S, R^c)$

Variables $S = true$ **and** $R = false$ **are fixed** *evidence*

Generate sample:

1. **Sample** T: **e.g** $T = false$
2. **Sample** F: **e.g.** $F = true$
3. **Sample** $A|F, T^c$: **e.g.** $A = true$
4. **Now:** $S = true$ **is evidence**
   weight $w = P(S|F) = 0.9$

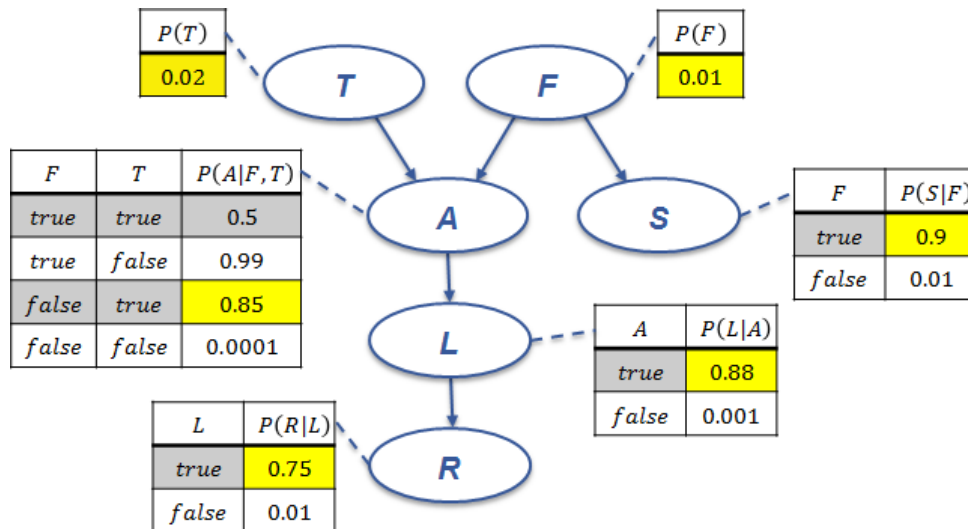# Likelihood weighting in Bayesian networks
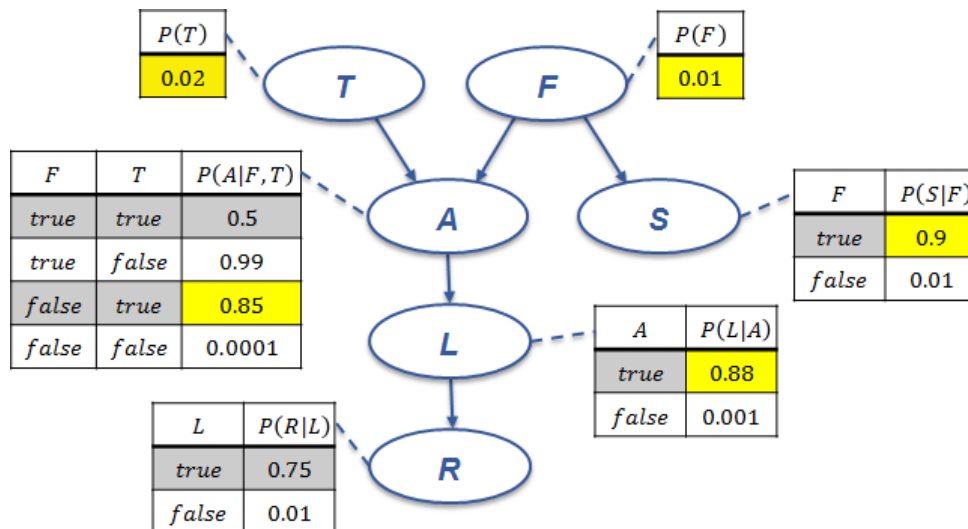
F	T	$P(A|F,T)$
true	true	0.5
true	false	0.99
false	true	0.85
false	false	0.0001

	$P(T)$
	0.02

	$P(F)$
	0.01

F	$P(S|F)$
true	0.9
false	0.01

A	$P(L|A)$
true	0.88
false	0.001

L	$P(R|L)$
true	0.75
false	0.01

Example: estimate $P(T|S, R^c)$

Variables $S = true$ **and** $R = false$ **are fixed** *evidence*

Generate sample:

1. **Sample** T: **e.g** $T = false$
2. **Sample** F: **e.g.** $F = true$
3. **Sample** $A|F, T^c$: **e.g.** $A = true$
4. **Now:** $S = true$ **is evidence**
   weight $w = P(S|F) = 0.9$
5. **Sample** $L|A$: **e.g.** $L = true$
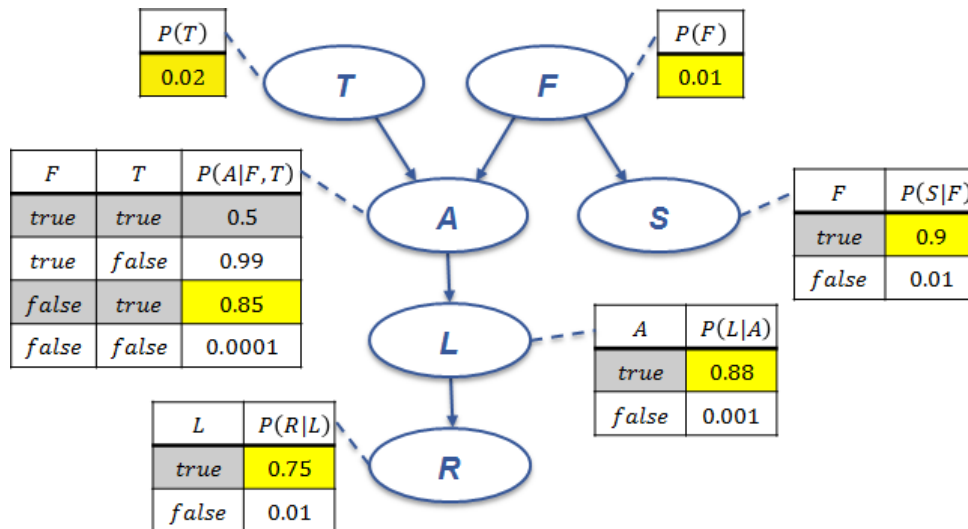
Likelihood weighting in Bayesian networks

P(T)
0.02

P(F)
0.01

F	T	P(A|F,T)
true	true	0.5
true	false	0.99
false	true	0.85
false	false	0.0001

F	P(S|F)
true	0.9
false	0.01

A	P(L|A)
true	0.88
false	0.001

L	P(R|L)
true	0.75
false	0.01

Example: estimate $P(T|S, R^c)$

Variables $S = true$ **and** $R = false$ **are fixed** *evidence*

Generate sample:

1. **Sample** T: **e.g** $T = true$
2. **Sample** F: **e.g.** $F = true$
3. **Sample** $A|F, T$: **e.g.** $A = false$
4. **Now:** $S = true$ **is evidence**
    weight $w = P(S|F) = 0.9$
5. **Sample** $L|A^c$: **e.g.** $L = true$
6. **Now:** $R = false$ **is evidence**
    $w = 0.9 \cdot P(R^c|L) = 0.9 \cdot 0.25 = 0.225$

Likelihood weighting in Bayesian networks

P(T)
0.02

P(F)
0.01

F	T	P(A|F,T)
true	true	0.5
true	false	0.99
false	true	0.85
false	false	0.0001

F	P(S|F)
true	0.9
false	0.01

A	P(L|A)
true	0.88
false	0.001

L	P(R|L)
true	0.75
false	0.01

Example: estimate $P(T|S, R^c)$

$$w = 0.9 \cdot P(R^c|L) = 0.9 \cdot 0.25 = 0.225$$

Sample $\{T, F, A^c, S, L, R^c\}$ **gets weight** $w = 0.225$

Draw new samples and add the weights for each specific state. E.g. if state $\{T, F, A^c, S, L, R^c\}$ **occurs** n **times the total weight for that state is** $n \cdot 0.225$.

To compute $P(T|S, R^c)$

$$P(T|S, R^c) = \frac{\text{sum of weights where } T = true}{\text{sum of all weights}}$$

Likelihood weighting: algorithm

1. **Initiate the weight of the sample:** $w = 1$
2. **Initiate the state of the sample** $x = \{\}$
3. **Initiate total weights:** $W(x) = 0$
4. **Sample node by node:**
   a) **If current node is evidence (conditioned on)**
      $$w = w \cdot P(currentNode|parents)$$
      Add state of node to x.
   b) **If not, sample from distribution to determine its state. Add state of node to** x.
5. **Add weight to total weight:** $W(x) = W(x) + w$

Compute estimate for specific probabilities

$$\hat{P}(X|Y) = \left(\sum_{x:X|Y} W(x) \right) / \left(\sum_{x} W(x) \right)$$
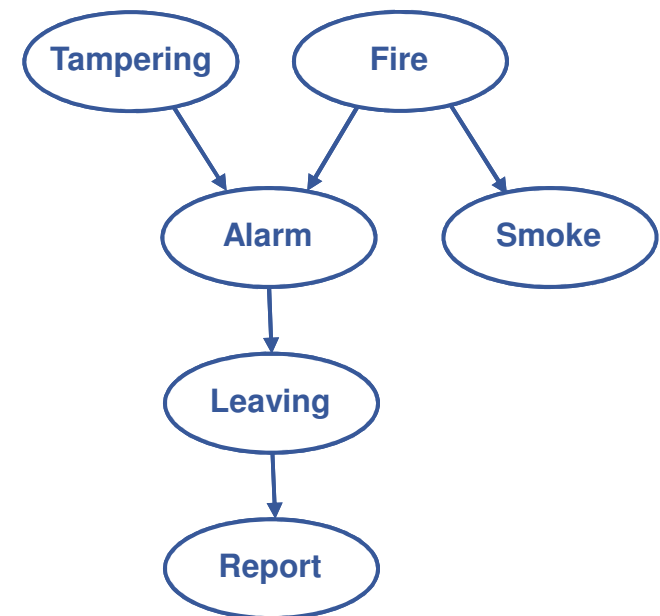
Likelihood weighting in Bayesian networks

- **More efficient than rejection sampling.**

- **But doesn't solve all our problems:**

  - **By fixing the evidence nodes, we influence the sampling of the downstream variables, not the upstream ones.**

We would like to consider the evidence when we sample every node!
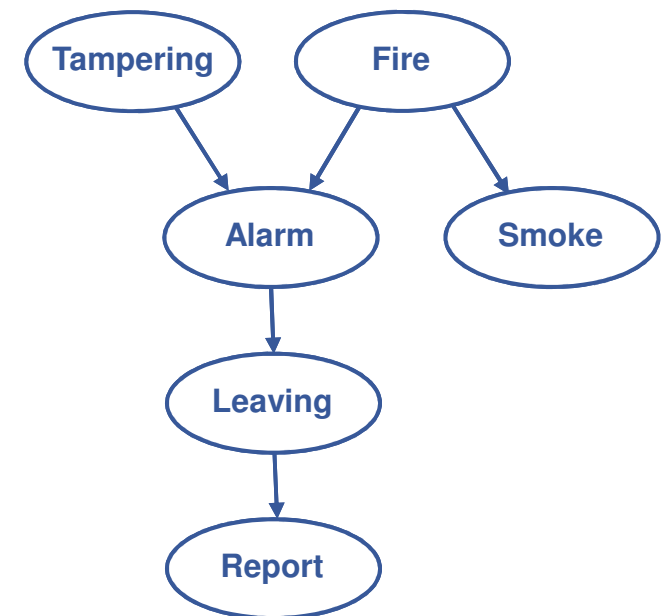
Markov chain Monte Carlo (MCMC)

- MCMC combines Monte Carlo simulation and Markov chains.

- **Idea**: Instead of generating every sample from scratch, we create samples that are similar to the previous one

- We construct a **Markov chain** that has the desired distribution as **stationary** distribution.
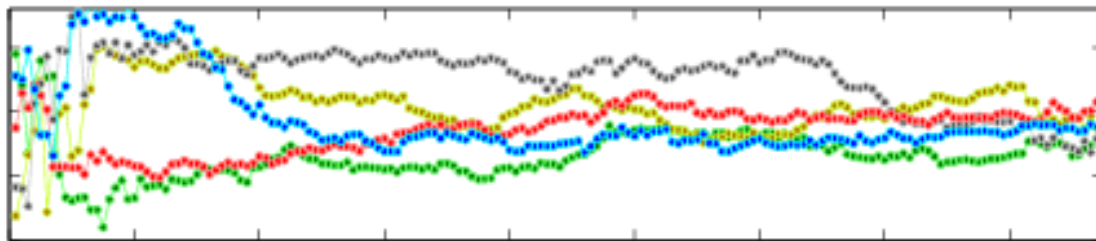
Markov processes

- A **random process** is a set of random variables $\{X(t)\}_{t \in T}$ developing over time t.

  - The time can be continuous or discrete
  - It can be actual time or some kind of spatial enumeration (e.g. characters in a text)

- A **Markov process** is a random process were conditioning on the current state, future states are independent of the past.
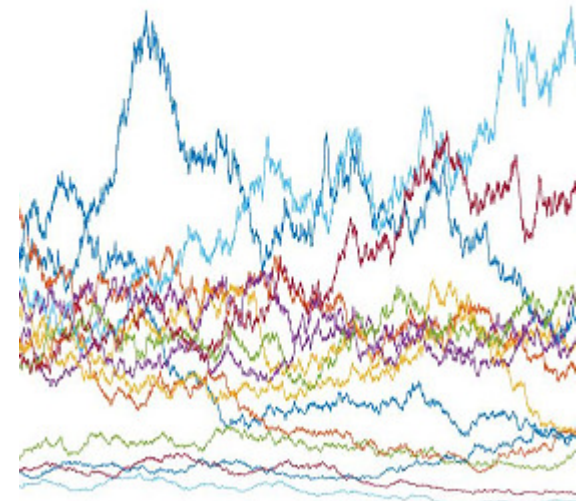
Markov chains

- A **Markov chain** is a discrete realization $X_1, X_2, X_3, \ldots$ of a Markov process, where the process jumps between states in some state space

$$S = \{s_1, \ldots, s_N\}$$

and where the next jump only depends on the current state

$$P(X_{n+1} = j | X_1 = x_1, \ldots, X_{n-1} = x_{n-1}, X_n = i)$$
$$= P(X_{n+1} = j | X_n = i)$$
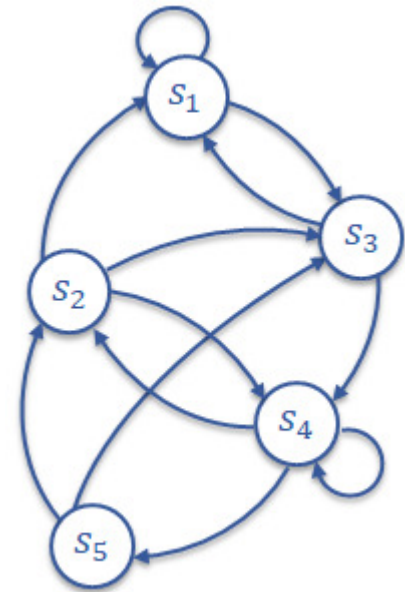
Markov chains

- **A Markov chain is defined by its**

  - **state space** $S = \{s_1, \ldots, s_N\}$

  - **initial state distribution**
    $$\pi_i = P(X_1 = s_i), i = 1, \ldots, n$$

  - **transition probabilities**
    $$p_{ij} = P(X_{n+1} = j | X_n = i)$$
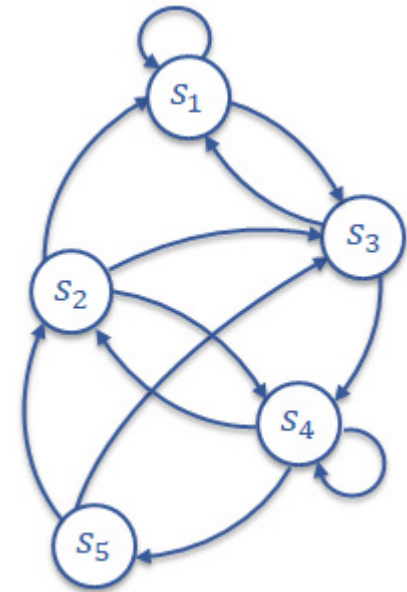
Markov chains

- A Markov chain is said be be *stationary* if its distribution of the state is independent of its starting condition.

- This means that we can start the chain in any initial state and if we run it long enough (burn-in period) we approach stationarity.

- Key idea of MCMCs: construct a Markov chain whose stationary distribution is the distribution we want to sample from.
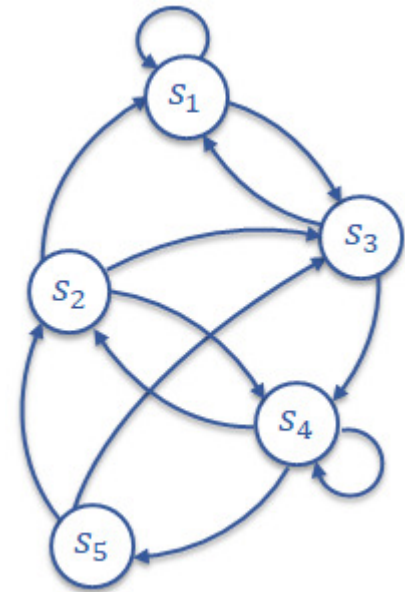
The Metropolis-Hastings algorithm

Assume that we want to sample from
$$p(\theta) = f(\theta)/K$$
where K is a normalizing constant that may be difficult to compute.

The idea is choose generate random numbers from some *proposal* distribution, which will be accepted or rejected according to some *acceptance* probability.
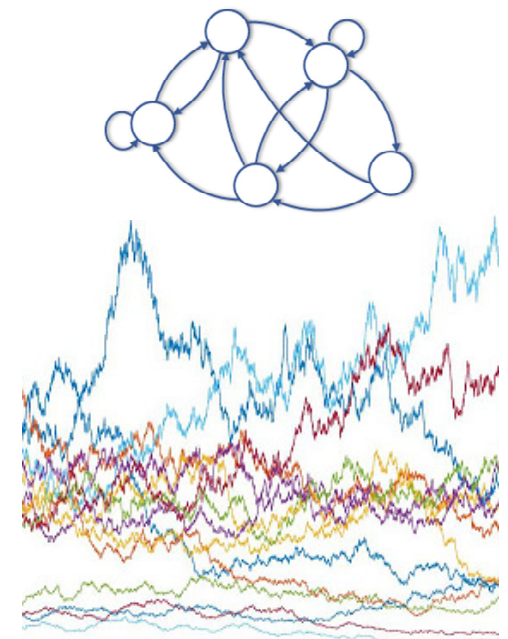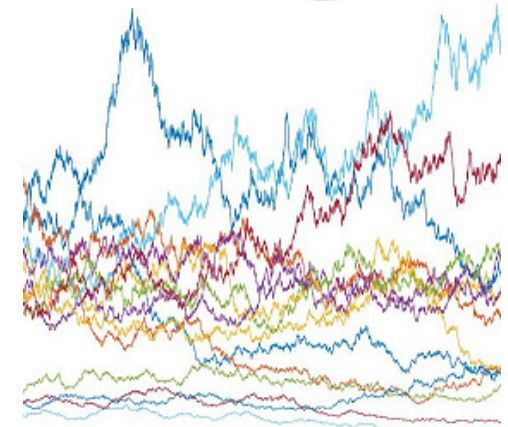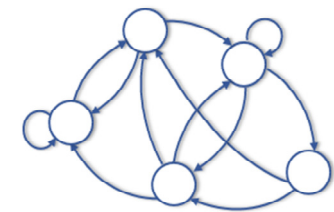
The Metropolis-Hastings algorithm

1. **Choose an initial value $\theta^{(0)}$ where $f(\theta^{(0)}) > 0$.**

2. **Use current value $\theta^{(k)}$, sample a candidate point θ^* according to some proposal prob $q(\theta^*|\theta^{(k)})$ (symmetric $q(x|y) = q(y|x)$.)**

3. **Compute the ratio**

$$\alpha = \frac{f(\theta^*)q(\theta^*|\theta^{(k)}}{f(\theta^{(k)})q(\theta^{(k)}|\theta^*)}$$

4. **If the jump increases the density ($\alpha > 1$), accept the point: $\theta^{(k+1)} = \theta^*$. If the jump decreases the density ($\alpha < 1$), accept the point with probability α. Else reject it. Return to step 2.**

The Metropolis-Hastings algorithm

- **Generates a Markov chain**
$$\left(\theta^{(0)}, \theta^{(1)}, \ldots, \theta^{(k)}, \ldots\right)$$
as the transition probability from $\theta^{(k)}$ to $\theta^{(k+1)}$ depends only on $\theta^{(k)}$.

- **After a burn-in period T samples from the vector $(\theta_{T+1}, \theta_{T+2}, \ldots, \theta_{T+n})$ are samples from $p(x)$.**
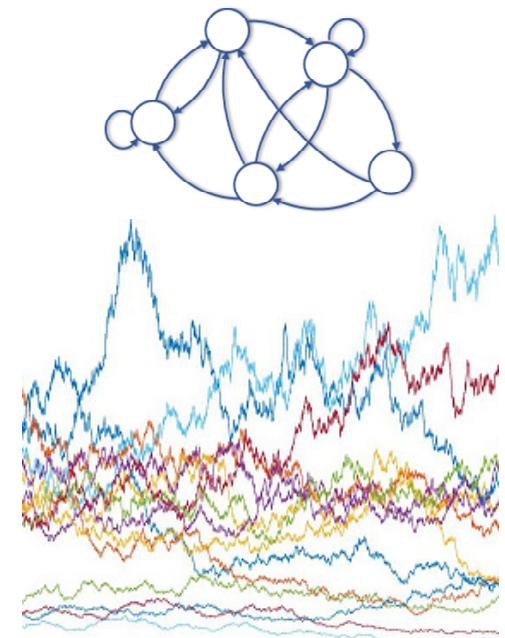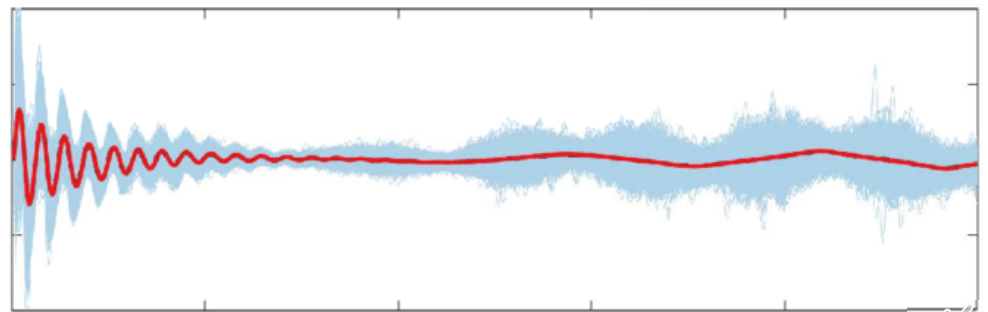
Burn-in period

A key issue in Metropolis-Hastings and MCMC is the number of steps until the chain reaches stationarity

- **this is called the burn-in period.**
- **typically the 1000-5000 first points are tossed.**

A poor choice of initial values can greatly increase the burn-in period.

- **Initiate in the centre of the distribution**

Gibbs sampling

- **The Gibbs sampler is a version of the Metropolis-Hastings algorithm that accepts all generated samples**

- **Key: all conditional distributions must have a nice form (conditionally conjugate likelihoods)**

- **Idea: resample one variable at a time, conditioned on the rest, but keep evidence fixed.**

- **Now samples are not independent, but sample averages are still consistent estimators.**
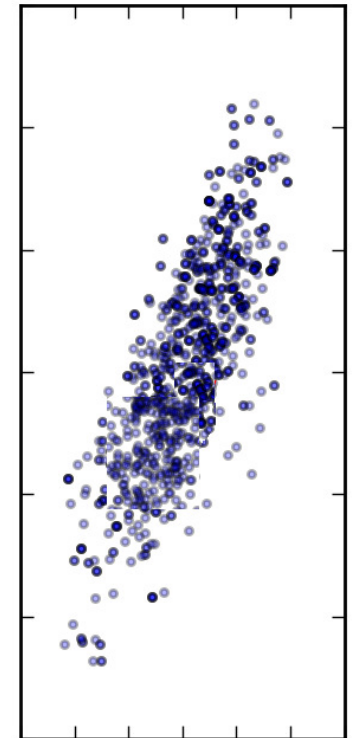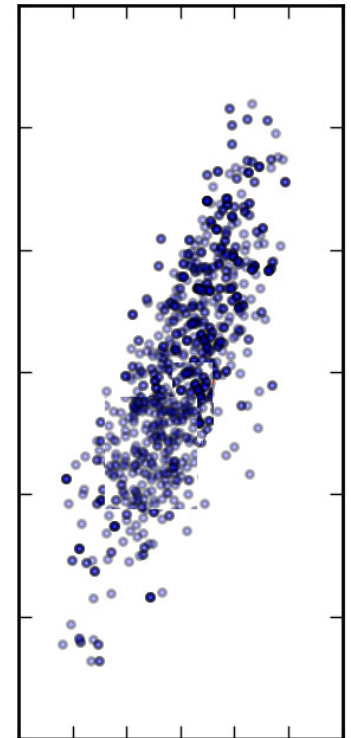
Example: Gibbs sampling

Assume we have a joint distribution of a discrete random variable $X \in \{0, 1, \dots n\}$ and a continuous random variable Y, $0 \leq Y \leq 1$, with joint density function

$$p(x, y) = \frac{n!}{(n-x)!\, x!} y^{x+\alpha-1}(1-y)^{n-x+\beta-1}$$

Complex joint distribution, but simple conditionals:

$$X|Y \sim \text{Bin}(n, y)$$
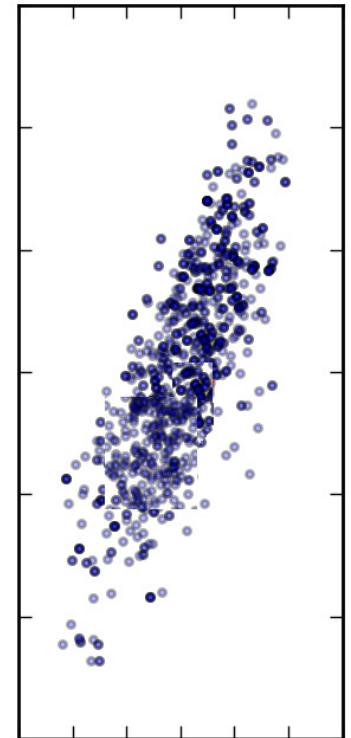$$Y|X \sim \text{Beta}(x + \alpha, n - x + \beta)$$

Example: Gibbs sampling

Let $n = 10$ and $\alpha = 1, \beta = 2$. Initiate with $y_0 = 1/2$.

1. **Generate x_0 from $\mathrm{Bin}(n, y_0) = \mathrm{Bin}(10, 1/2)$. Say we get $x_0 = 5$.**

2. **Generate y_1 from $\mathrm{Beta}(x_0 + \alpha, n - x_0 + \beta) = \mathrm{Beta}(6,7)$, giving $y_1 = 0.33$.**

3. **Generate x_1 from $\mathrm{Bin}(10, 0.33)$ giving $x_1 = 3$.**

And so on. The x_i are samples from the marginal posterior of X_i, and y_i from that of Y_i.

Gibbs sampling in a Bayesian network

Assume again that we want to estimate a probability
$P(T, A^c | S, R)$

1. **Initialization:**
   a. **Fix the evidence variables** $S = true, R = true$
   b. **Sample all other variables** T, F, A, L

2. **Repeat (as many times as wanted)**
   a. **Pick a non-evidence variable** X_i **uniformly**
   b. **Sample** x_i' **from** $P(X_i | x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$
   c. **Keep all other variables** $x_j' = x_j, \forall j \neq i$
   d. **The new sample is** $x_1', \ldots x_n'$

3. **Alternatively march through the variables in some predefined order.**