Simulation

Lecture 1

Perfect simulation I

What, Why, and How

Mark Huber
Claremont McKenna College
July, 2018

Supported by NSF grant DMS 1418495

What is perfect simulation/integration?

Perfect simulation

Stopping time rules that tell you when you have reached a point where you have a sample that comes exactly from your target distribution.

Perfect integration

Find an integral/expectation that has error bounds chosen by the user without regard to the (usually unknown) variance of your Monte Carlo samples.

Why perfect simulation?

Exact simulation from high dimensional distributions

Applies in situations where dependence between dimensions bounded. Markov random fields, weighted permutations, Stochastic Differential Equations.

Replaces the burn-in/warm-up/mixing steps in MCMC

Often mixing time dealt with by limited heuristics in stat applications, which works most but not all the time.

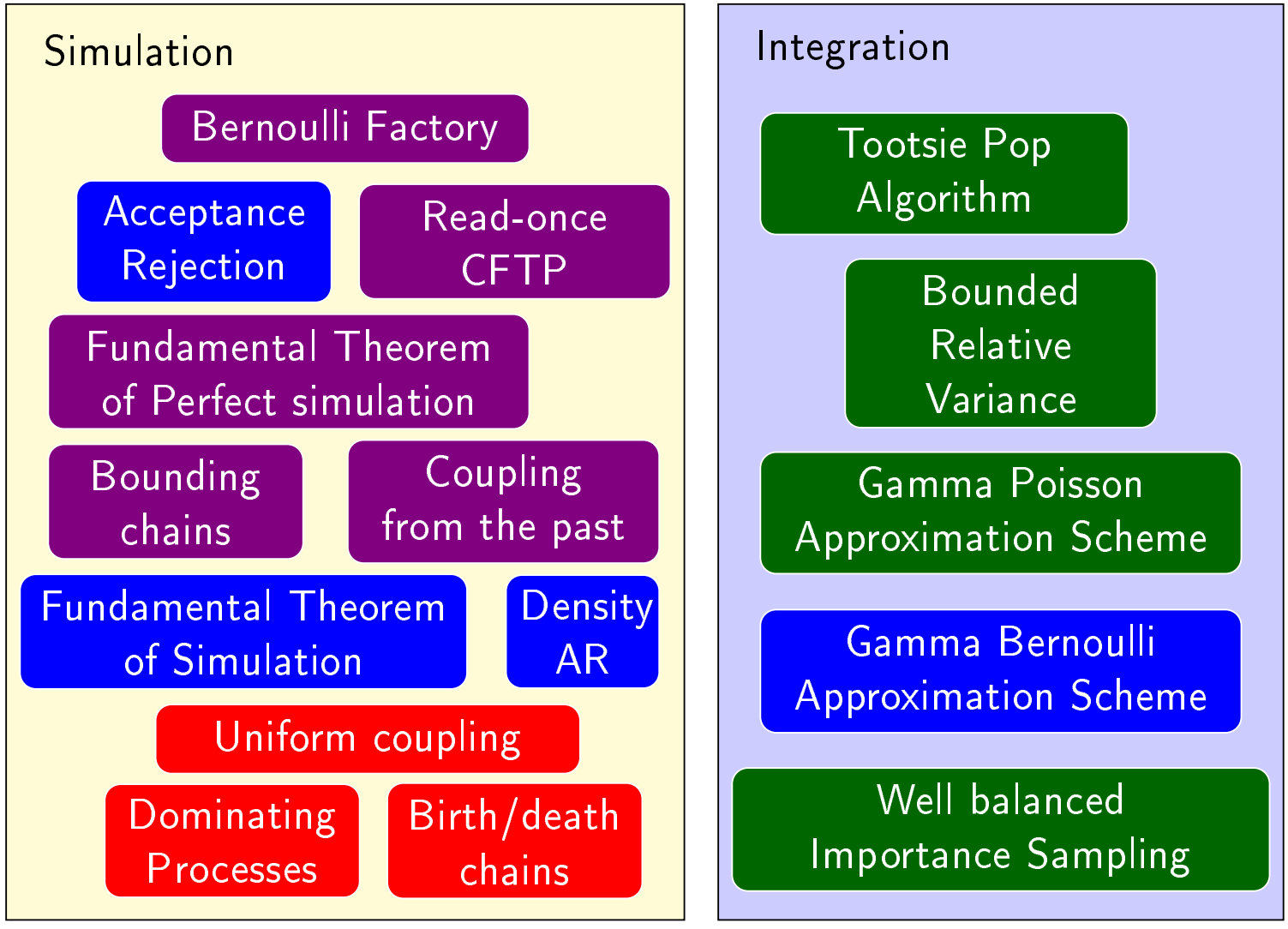
Allows for more complicated Markov chains

Often steps require exact draws from difficult distributions. For example, Zig-zag and slice samplers can need these algorithms as subroutines.

How to perfectly integrate

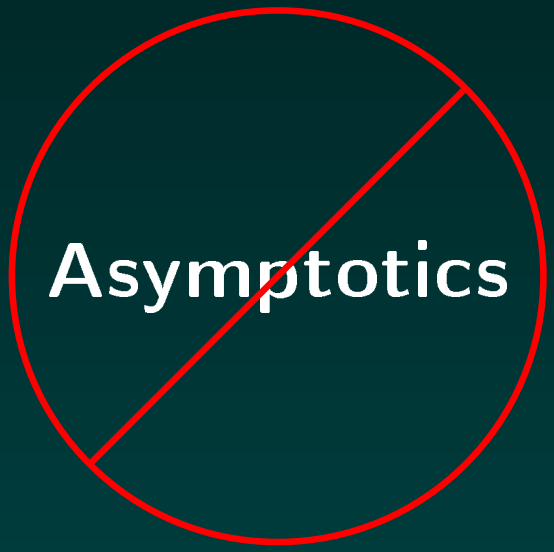
Overarching philosophy

Keep track of both the state and the distribution of the state, randomly changing both until you reach a point where you have the distribution you need.


Simulation
Bernoulli Factory
Acceptance Rejection
Read-once CFTP
Fundamental Theorem of Perfect simulation
Bounding chains
Coupling from the past
Fundamental Theorem of Simulation
Density AR
Uniform coupling
Dominating Processes
Birth/death chains
Integration
Tootsie Pop Algorithm
Bounded Relative Variance
Gamma Poisson Approximation Scheme
Gamma Bernoulli Approximation Scheme
Well balanced Importance Sampling

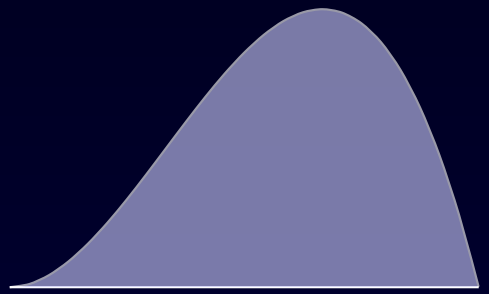
Simulation

We don't need no asymptotics

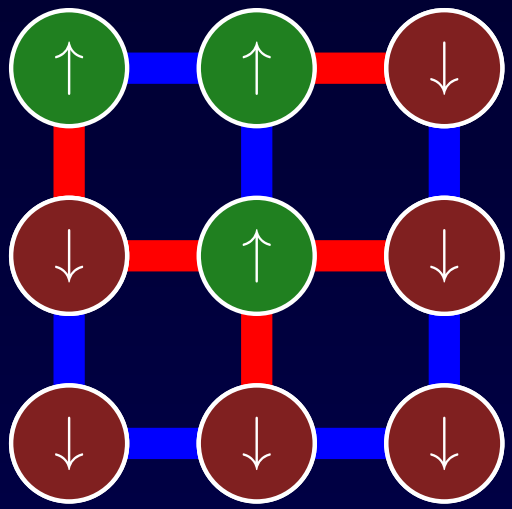


Four sampling problems

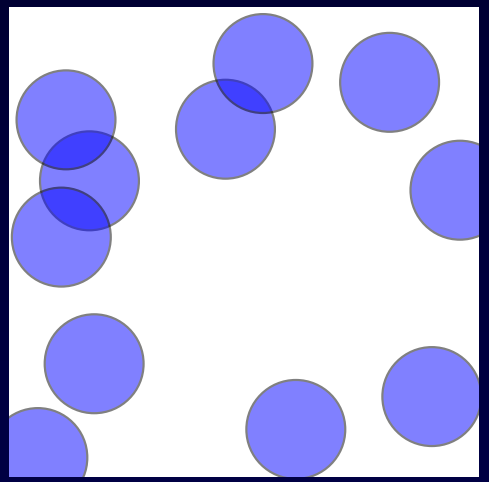
Unif$(\{1, \ldots, 5\})$



Beta$(3, 2)$



Ising model



Strauss point process

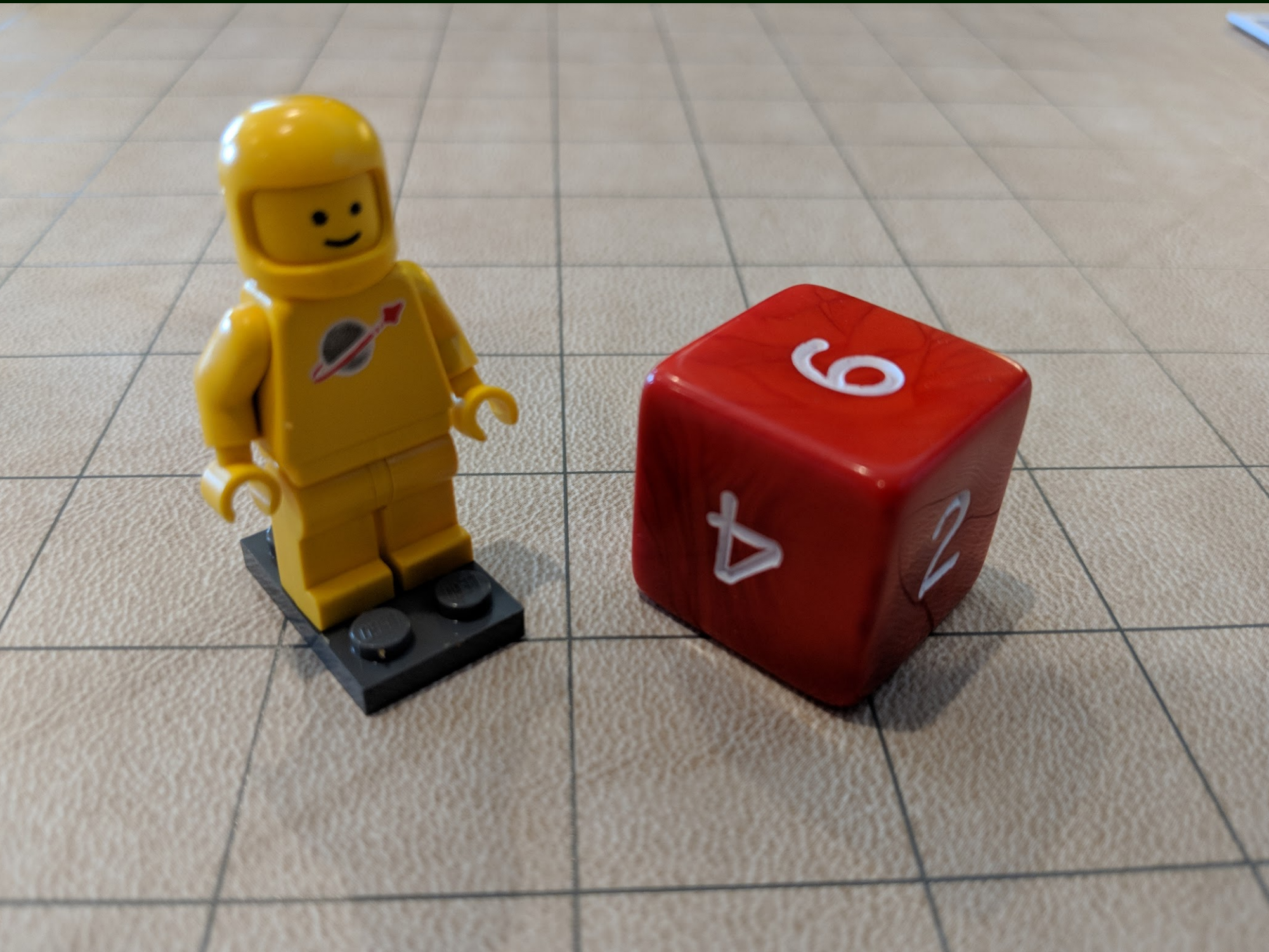
Start with the easy one

Suppose that I invite four friends over for a game of dice





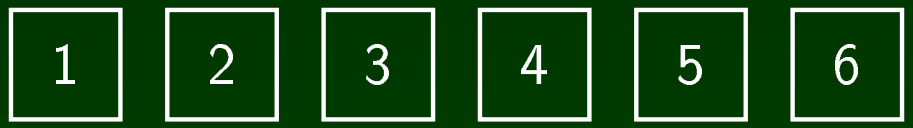
Question, how to decide who goes first?


6
4
2

Restating

Goal: sample uniformly from

Source of randomness: rolls of a fair sixed die, so uniform over

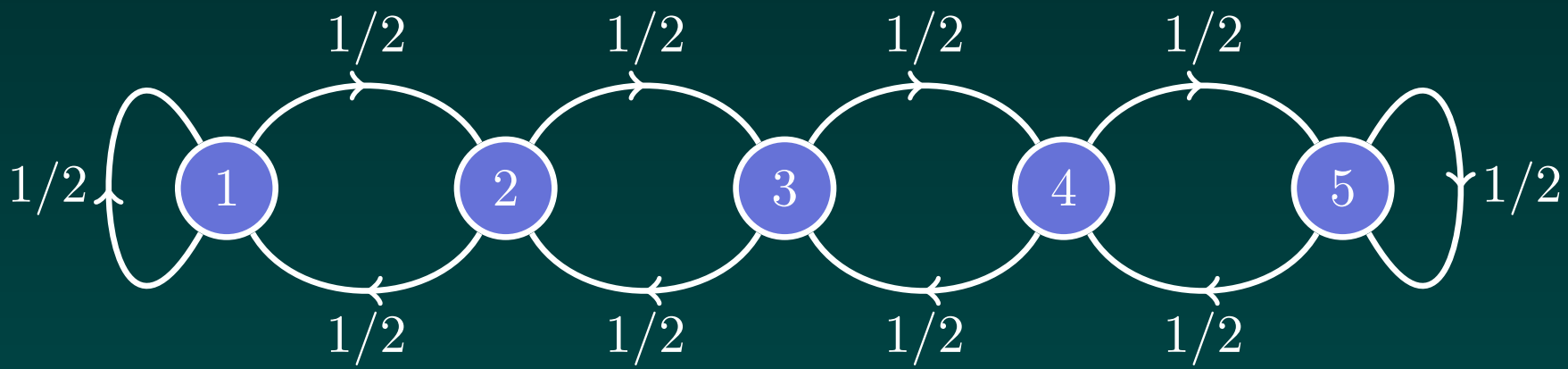


The MCMC approach

Use our random six sided die rolls to move one to the left or right

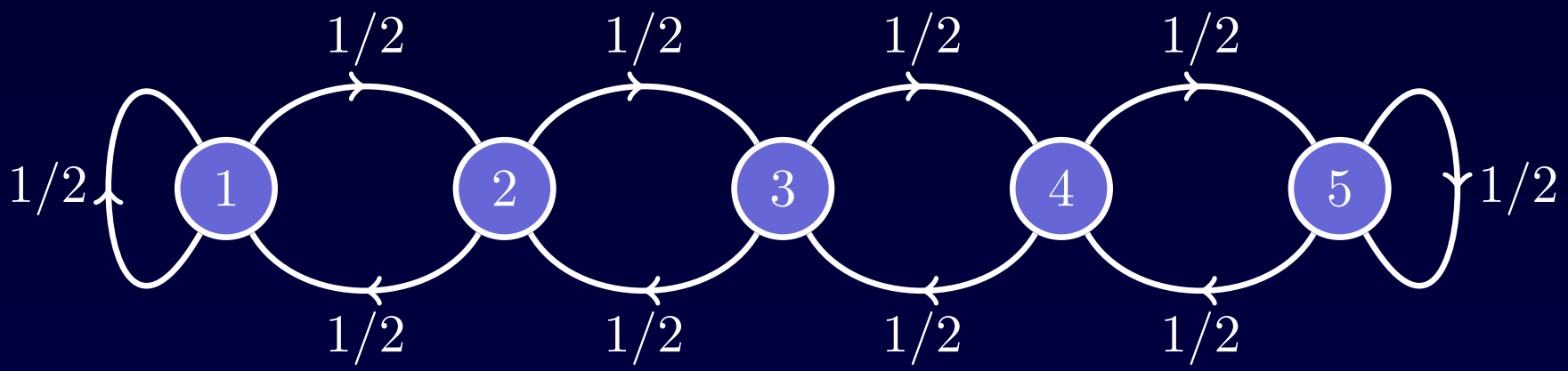
$$R_1, R_2, \ldots \overset{\text{iid}}{\sim} \text{Unif}(\{1, 2, 3, 4, 5, 6\})$$

- If $R_i \in \{1, 3, 5\}$ try to move one step to the right,
- Else try moving one step to to the left



Update function version

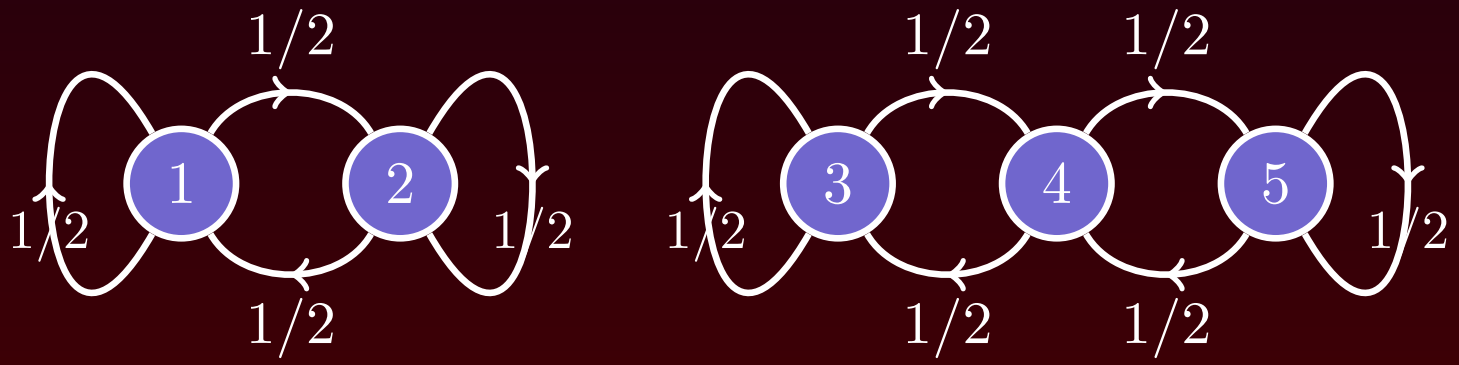
$$R_1, R_2, \ldots \overset{\text{iid}}{\sim} \mathsf{Unif}(\{1,2,3,4,5,6\})$$
$$D_t = \mathbb{1}(R_t \in \{1,3,5\}) - \mathbb{1}(R_t \in \{2,4,6\})$$
$$X_{t+1} = X_t + D_t \mathbb{1}(X_t + D_t \in \{1,\ldots,5\}) = \phi(X_t, R_t)$$



Here $\mathbb{1}(\text{expression}) = 1$ if the expression is true and 0 otherwise.

Problem with this approach

- Final state X_t never exactly uniform
- Do not know initially how many steps to take in order for state to be close to uniform
- For complex problem, could accidentally create chain that does not mix at all

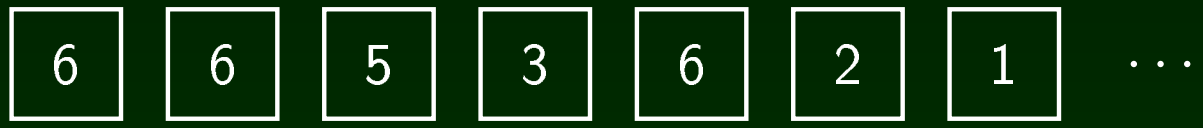


A better idea

1. Roll the six sided die until we get a number from 1 to 5
2. Output the number rolled

Viewed as a stopping time

Roll the die over and over again (let $R_i \overset{\text{iid}}{\sim} \text{Unif}(\{1, \ldots, 6\})$)



$T = 3$

Let T be first time number is not 6. Formally

$$T = \inf\{t : R_t \neq 6\},$$

here $\inf$ is the infimum that is the smallest number for which this happens–after finding the inf, our sample is

$$X = R_T$$

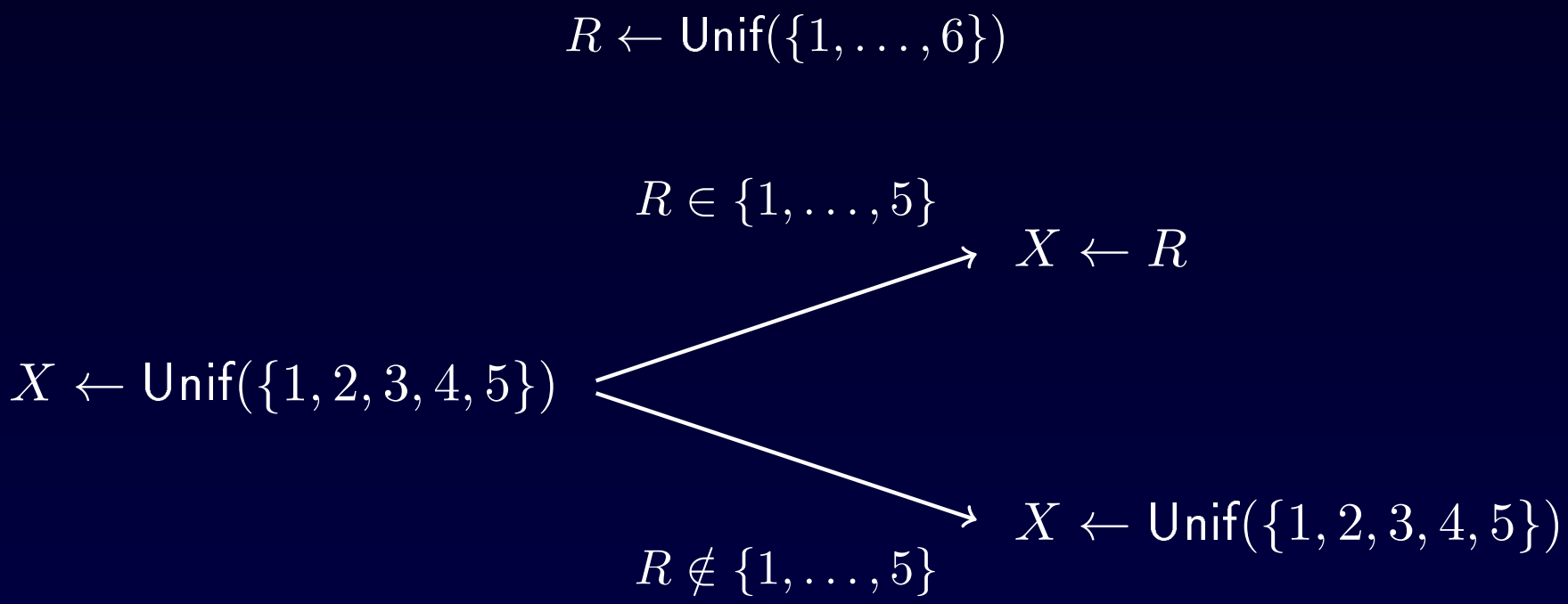
Call T a **stopping time**

Viewed as a mixture

$\mathsf{Unif}(\{1,\ldots,6\})$ $\mathsf{Unif}(\{1,\ldots,5\})$ $\mathsf{Unif}(\{6\})$

$\frac{6}{6}$ $\frac{5}{6}$ $\frac{1}{6}$

Viewed as branching process



Pros and cons

What have we lost

- Need a random number of tries to get our sample

What have we gained

- Answer (when arrives) comes exactly from desired distribution
- No need to build Markov chain or understand mixing time

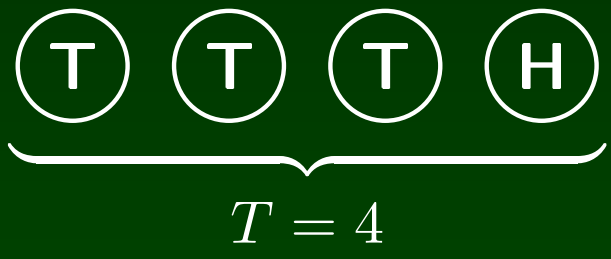
Perfect simulation

Definition

A **perfect simulation** algorithm uses a number of steps T in order to sample exactly from a target distribution π. Here T is an unbounded random variable that is finite with probability 1.

Example $T \sim \text{Geo}(1/2)$

1. Flip a fair coin until you get a head
2. Return the number of flips



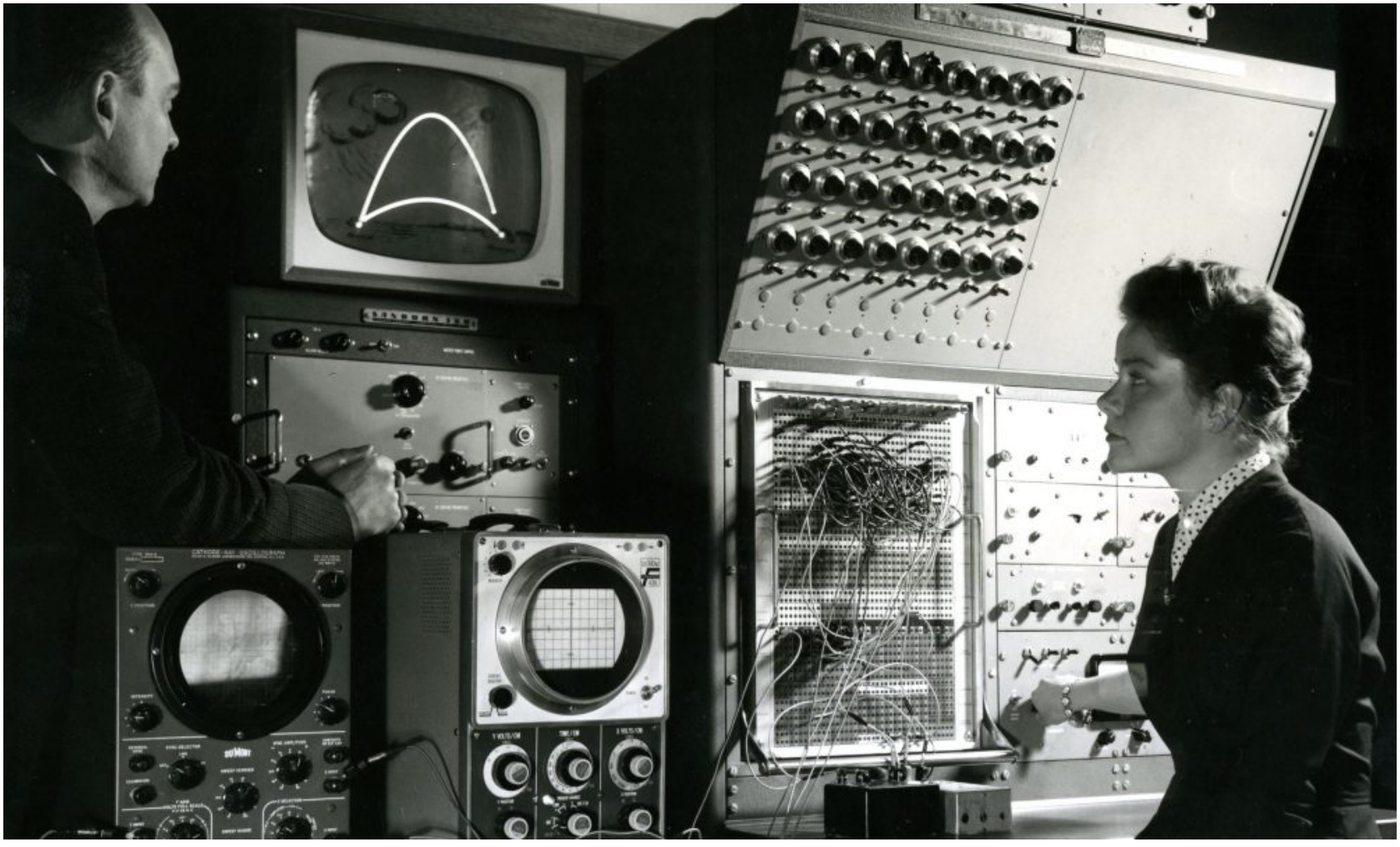


Photo credit: Rutgers School of Engineering

In pseudocode

Repeat version

`RepeatGeo`

1. $T \leftarrow 0$
2. Repeat
   - 2.1 Draw $C \leftarrow \text{Unif}(\{\text{tails,heads}\})$
   - 2.2 $T \leftarrow T + 1$
3. Until $C = \text{heads}$
4. Output T

Recursive definition

`RecursiveGeo`

1. Draw $C \leftarrow \text{Unif}(\{\text{tails,heads}\})$
2. If $C = \text{heads}$ output 1
3. Else
   - 3.1 $Y \leftarrow$ `RecursiveGeo`
   - 3.2 $T \leftarrow 1 + Y$

Viewed as a mixture

Let δ_x be distribution where

$$X \sim \delta_x \Leftrightarrow \mathbb{P}(X = x) = 1$$

$\text{Geo}(1/2)$ is a mixture of a fixed dist and a shifted version of itself

$$\text{Geo}(1/2) \sim (1/2)\delta_1 + (1/2)[1 + \text{Geo}(1/2)]$$

Perfect simulation algorithms take advantage of these mixtures!

Perfect simulation protocols

Definition

A **protocol** is a general framework for creating algorithms.

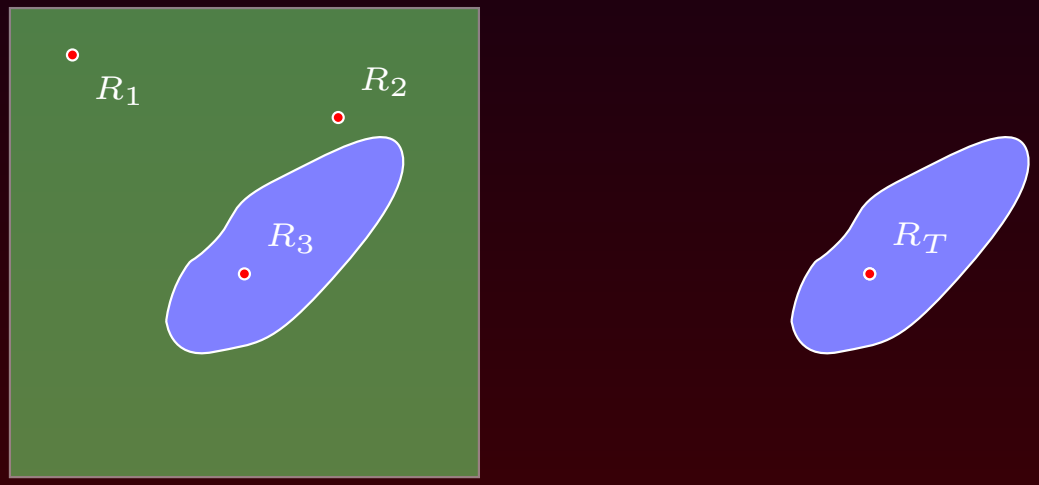
First perfect simulation protocol: Acceptance Rejection

> *John Von Neumann, Various Techniques used in connection with random digits,* J. Res. Nat. Bur. Stand. Appl. Math, *Series 3, 36–38 (1951)*

Uniform Acceptance Rejection

Theorem

Let $A \subseteq B$, $R_1, R_2, \ldots$ *Unif*(B), *and* $T = \inf\{t : R_t \in A\}$. *If* $\mathbb{P}(T < \infty) = 1$, *then* $R_T \sim$ *Unif*(A).



$T = 3$

AR in pseudocode repeat view

AR

1. Repeat
2. Draw $X \leftarrow \text{Unif}(B)$
3. Until $X \in A$
4. Output X

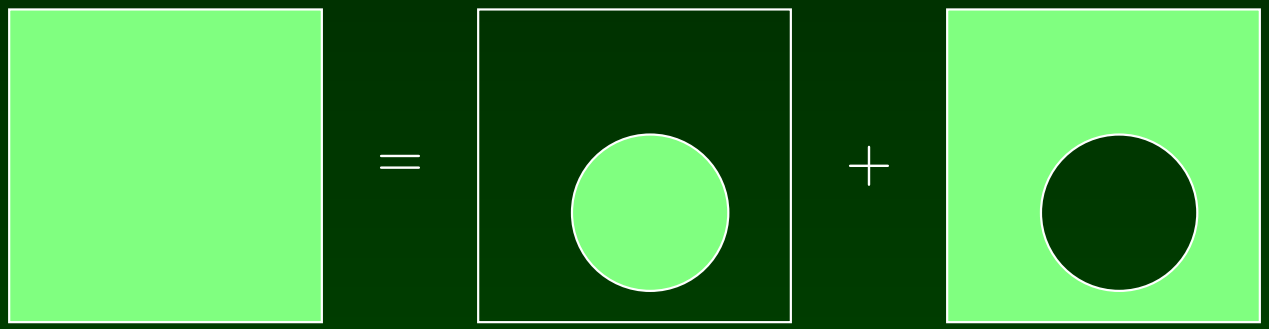
AR in pseudocode recursive view

`AR`

1. Draw $X \leftarrow \text{Unif}(B)$
2. If $X \notin A$, $X \leftarrow$ `AR`
3. Output X

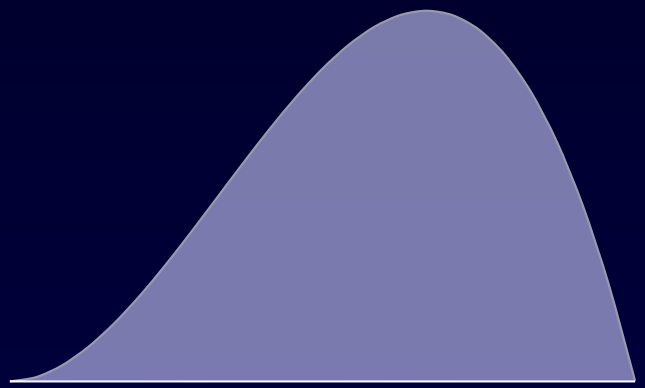
AR in mixture view

$$\mathsf{Unif}(A) \sim \frac{m(A)}{m(B)}\mathsf{Unif}(A) + \frac{m(B) - m(A)}{m(B)}\mathsf{Unif}(B)$$



Question: how to sample from densities with AR

$$f_X(s) = s^2(1-s)\mathbb{1}(s \in [0,1])$$



$\mathsf{Beta}(3,2)$

Fundamental Theorem of Simulation

The FTS: informally stated

Densities are an illusion (multivariate densities doubly so): all random variables can be viewed as uniform random variables.

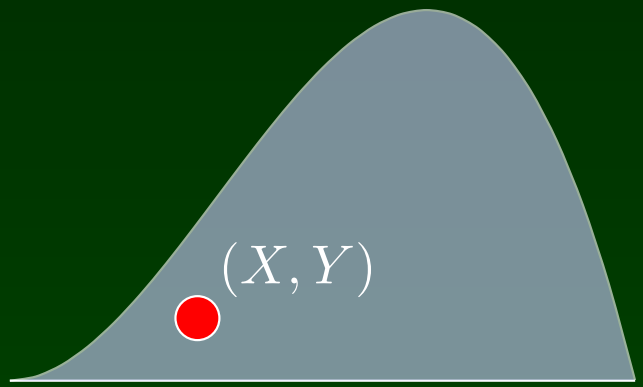
C. P. Robert and G. Casella, Monte Carlo Statistical Methods (Second Edition), Springer, p. 47, 2004

The FTS: In pictures

To draw $X \sim f_X$

1. Draw (X, Y) uniformly from region under the density curve
2. Output X

$$f_X(s) = s^2(1-s)\mathbb{1}(s \in [0,1])$$



Beta$(3, 2)$

C. P. Robert and G. Casella, Monte Carlo Statistical Methods (Second Edition), Springer, p. 47, 2004

The FTS: Stated formally

Theorem (Fundamental Theorem of Simulation)

Simulating X with density f is equivalent to drawing

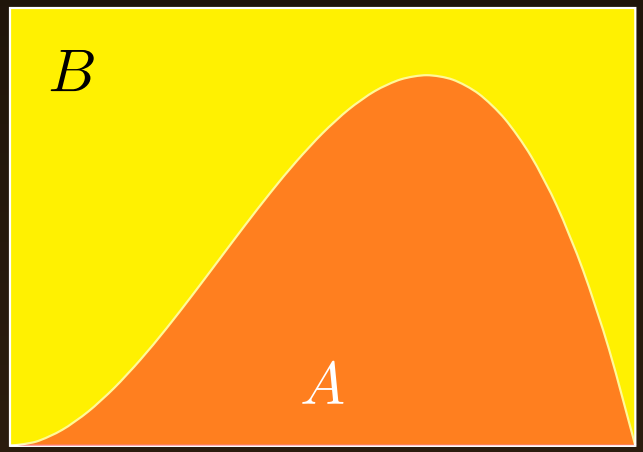
$$(X, Y) \sim \textit{Unif}(\{(x, y) : y \in [0, f(x)]\})$$

and keeping the X value.

C. P. Robert and G. Casella, Monte Carlo Statistical Methods (Second Edition), Springer, p. 47, 2004

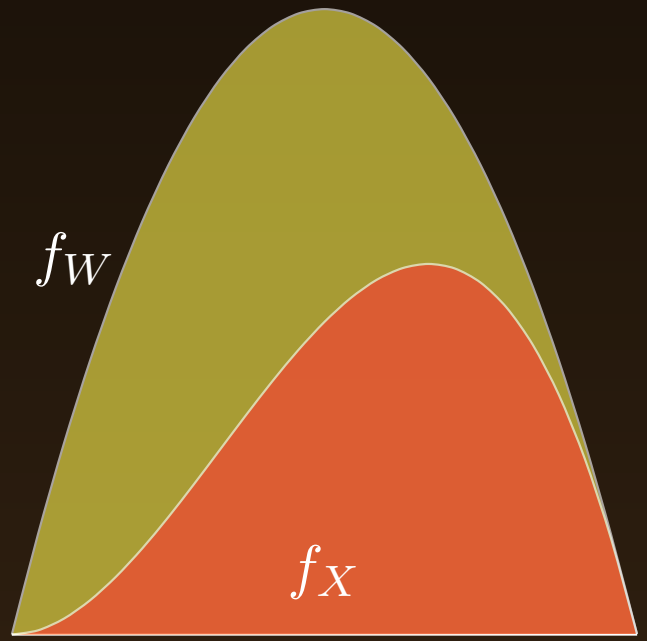
Drawing from densities with FTS and AR

1. Embed area under density A in a larger region B



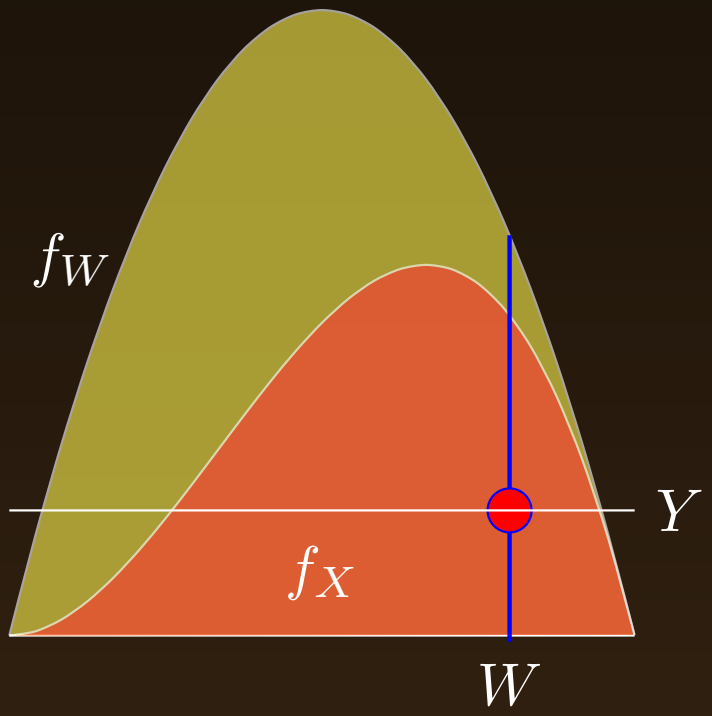
2. Draw (X, Y) uniformly from A using AR
3. Output X

What if B is also defined by a density?



Say that f_W **dominates** f_X if $f_W \geq f_X$

AR for dominated densities



1. Repeat
2. Draw $W \sim f_W$
3. Draw $Y \leftarrow \text{Unif}([0, f_W(W)])$
4. Until $Y \leq f_X(W)$
5. Output W

Once W has been drawn, the probability that a uniform draw on $[0, f_W(W)]$ is at most $f_X(W)$ is $f_X(W)/f_W(W)$.

AR for unnormalized densities

If $f_W(s) \geq f_X(s)$ for all s,

1. Repeat
2. Draw $W \leftarrow f_W$
3. Draw $U \leftarrow \text{Unif}([0,1])$
4. Until $U \leq f_X(W)/f_W(W)$
5. Ouptut W

Expected number of draws of W needed:

$$\frac{\int_\Omega f_W(s)\ ds}{\int_\Omega f_X(s)\ ds}$$

AR for normalized densities

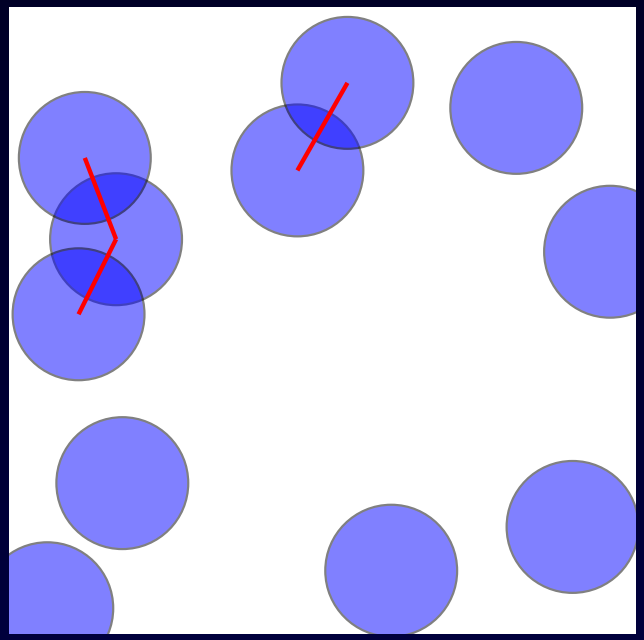
If $cf_W(s) \geq f_X(s)$ for all s, where f_W and f_X are normalized

1. Repeat
2. Draw $W \leftarrow f_W$
3. Draw $U \leftarrow \text{Unif}([0,1])$
4. Until $U \leq f_X(W)/[cf_W(W)]$
5. Ouptut W

Expected number of draws of W needed:

$$c$$

Problem #3: Strauss Point Process



Penalty is γ^3

A **Strauss point process** is a repulsive point process–so points are farther apart than for a Poisson process of equal intensity. It does this by having a factor of $\gamma < 1$ in the density for each pair of points within distance r of each other.

Strauss process viewed as density

A density consists of two things

1. A nonnegative function that puts more probability where the function is high
2. A reference measure

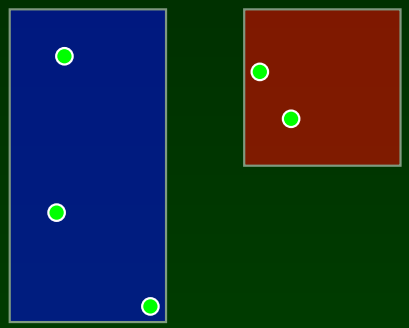
Example

- For continuous random variables over $\mathbb{R}$, the reference measure is dx, Lebesgue measure (aka length, area, etcetera) (so $\mu([3,7]) = 7 - 3 = 4$)
- For discrete random variables over $\{x_1, x_2, \ldots\}$, the reference measure is counting measure $\#$ (so $\#(\{a,b,c\}) = 3$)
- For the Strauss Point process, the reference measure is called a **Poisson point process**

What is a Poisson point process?

A **Poisson point process** (PPP) is a set of points P (possibly empty) with two properties.

1. The average number of points in A is proportional to λ and the measure of A.
2. For disjoint A and B, the number of points in A and B are independent.



Generating Poisson point processes

Ingredients

- Region A with area a
- Intensity λ

Method

1. Draw a Poisson random variable N with mean λa
2. Draw points $P_1, \ldots, P_N$ iid uniform over A.

Pseudocode for PPP

PPP

Input intensity λ, region A, *: Output:* $P \sim PPP(\lambda, A)$

1) Draw $N \leftarrow \text{Pois}(\lambda \cdot \text{area}(A))$
2) Draw $P_1, \ldots, P_N \leftarrow \text{iid Unif}(A)$
3) Output $\{P_1, \ldots, P_N\}$

Strauss process as a density

- Let $p = \{p_1, \ldots, p_k\}$ be a set of points in the region
- Let $t(p) = \#\{(i,j) : \mathrm{dist}(p_i, p_j) < R\}$
- Density of p wrt the unit rate Poisson process is

$$f_{\lambda,\gamma,R}(p) = \gamma^{t(p)}$$

AR for Strauss

AR method for Strauss

1. Repeat
2. Draw $P \leftarrow \mathbf{PPP}(\lambda)$
3. Draw $U \leftarrow \mathsf{Unif}([0,1])$
4. Until $U \leq \gamma^{t(P)}$

Why not use AR everywhere?

- Expected number of points in **PPP** draw:

$$\lambda \cdot \mathsf{area}(A)$$

- Probability a given point in space rejected because of close points

$$(1-\gamma)[1-\exp(-c_0 r^2 \lambda)]$$

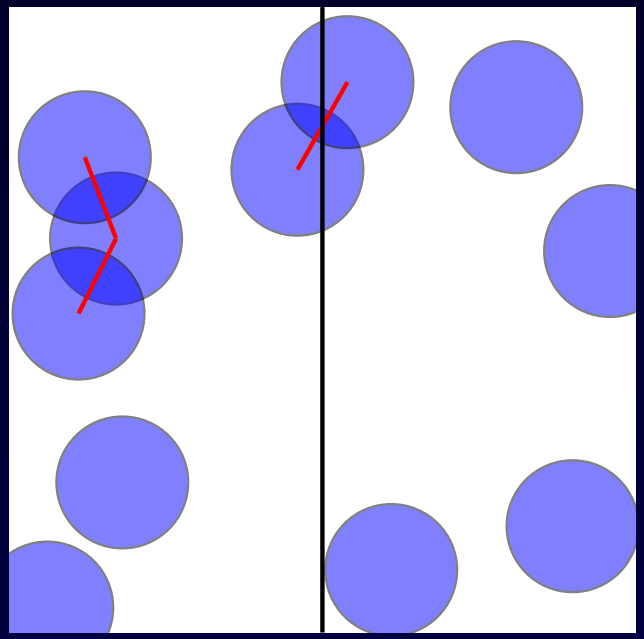
- Overall chance of acceptance for λr^2 small:

$$\approx \exp(-c_1(1-\gamma)\lambda^2 r^2 \mathsf{area}(A))$$

Divide and Conquer for AR

Density for AR is product of penalty factors

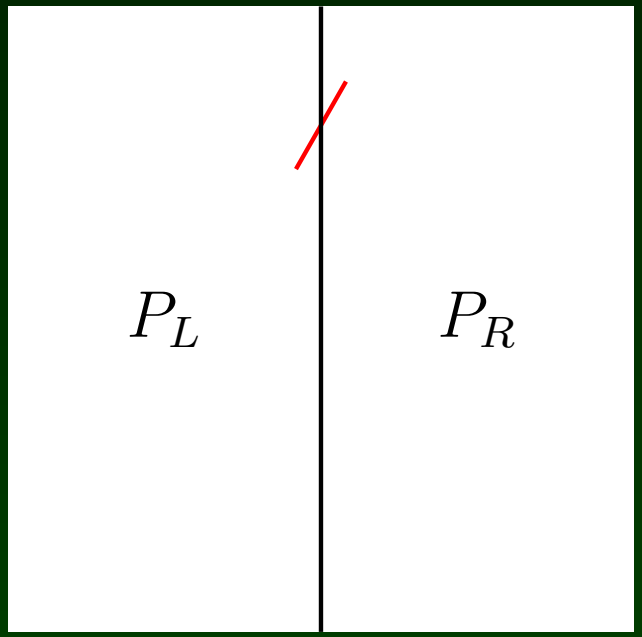
- Improve by breaking up region and then stitching together



Accept $\gamma^2 \cdot \gamma^0 \cdot \gamma^1$

Have to accept the left hand side penalties + right hand side penalties + penalties that cross from left hand side to the right hand side. Chance of accepting is $p_1 p_2 p_3$.

Stitching AR procedure



1 Repeat \\
2 Draw P_L from left half of region using AR
3 Draw P_R from right half of region using AR
4 Draw $U \leftarrow \mathsf{Unif}([0,1])$
5 Let $t(P_L, P_R)$ be number of pairs (p_ℓ, p_r) where $p_\ell \in P_L$, $p_r \in P_R$ with $\mathtt{dist}(p_\ell, p_r) < R$
6 Until $U \leq \gamma^{t(P_L,P_R)}$
7 Output $P_L \cup P_R$

Performance boost from stitching

Old way, expected running time

$$\frac{1}{p_1} \cdot \frac{1}{p_2} \cdot \frac{1}{p_3},$$

With stitching

$$\left[\frac{1}{p_1} + \frac{1}{p_2}\right] \frac{1}{p_3}$$

Can break down region even further, this improves running time up until we reach a point where

$$p_1 + p_2 \geq 1$$

Some run time results

	AR	AR stitching
$\lambda \cdot \text{area} = 10$	4.56 sec	3.31 sec
$\lambda \cdot \text{area} = 20$	132 sec	4.32 sec
$\lambda \cdot \text{area} = 40$	4.9 hrs	47.2 sec

Stitching more generally

If I was to sample from density that can be factored into two pieces that are independent and one that "stitches" the others together,

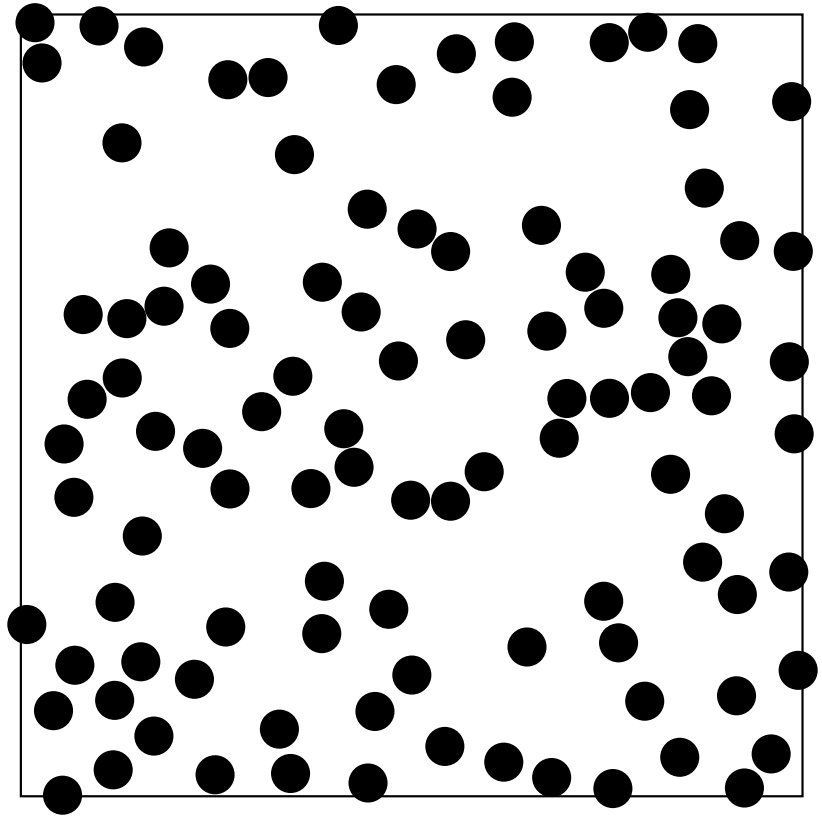
$$f(x_1, \ldots, x_n) = g(x_1, \ldots, x_k)h(x_{k+1}, \ldots, x_n)s(x_1, \ldots, x_n)$$

where s has known upper bound s'

1. Repeat
2. Draw A from g recursively
3. Draw B from h recursively
4. Draw $U \leftarrow \text{Unif}([0, 1])$
5. Until $U \leq s(A, B)/s'(A, B)$

A hard shell test

$\gamma = 0,\ N = 103, R = 0.05$



Chance of accepting naive AR: $< 10^{-19}$

Stitching AR gets sample in a few seconds

Variations on AR

Making AR efficient for specific problems

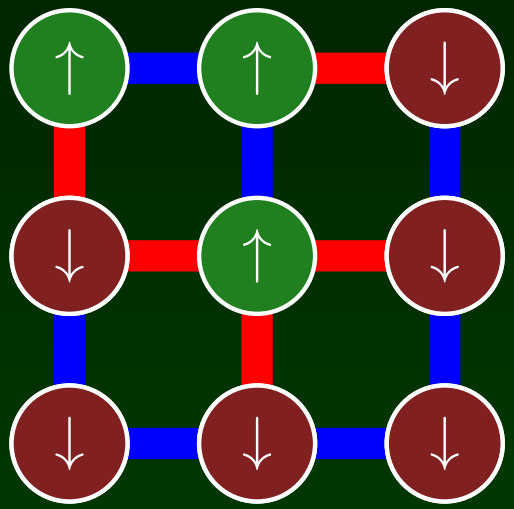
- Log-concave densities: Adaptive AR (Gilks & Wild 1992)
- Permanents (H. 2006)
- Stochastic Differential Equations (Beskos & Roberts 2006)

General techniques

- Divide and Conquer (Arratia & deSalvo 2015)
- Partially Recursive AR (H. 2016)

Dealing with bonuses rather than penalties

The Ising model over state space $\Omega = \{-1, 1\}^V$ can be viewed as giving a bonus to densities rather than a penalty:



Ising model

7 edges with like spins

Each like spin gives factor of $\exp(\beta)$

Each different spin gives factor of 1

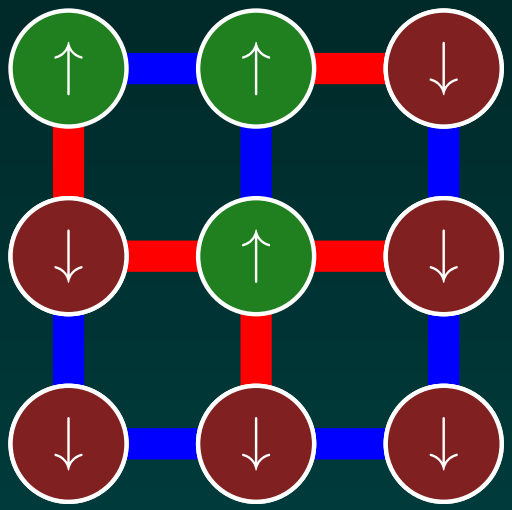
Density: $\exp(7\beta)$

Here the density with respect to uniform measure is

$$f_X(x) = \exp\left(\beta \sum_{\{i,j\} \in E} \mathbb{1}(x(i) = x(j))\right)$$

Solution

Multiply each factor by constant to make it at most 1: The Ising model over state space $\Omega = \{-1, 1\}^V$ can be viewed as giving a bonus at most 1:



Ising model

7 edges with like spins

Each like spin gives factor of 1

Each different spin gives factor of $\exp(-\beta)$

Density: $\exp(-5\beta)$

Here the density with respect to uniform measure is

$$g_X(x) = \exp\left(-\beta \sum_{\{i,j\}\in E} \mathbb{1}(x(i) \neq x(j))\right) = f_X(x)\exp(-\beta \#(E))$$

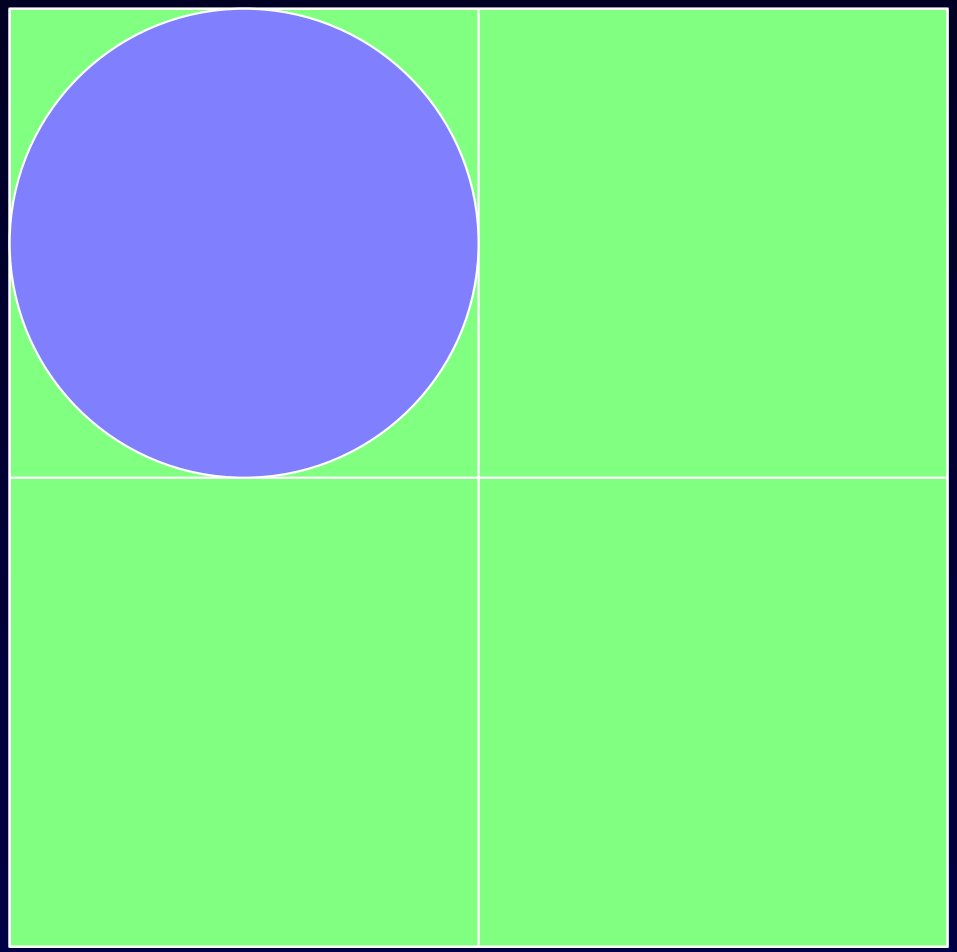
Integration

Inference as integration

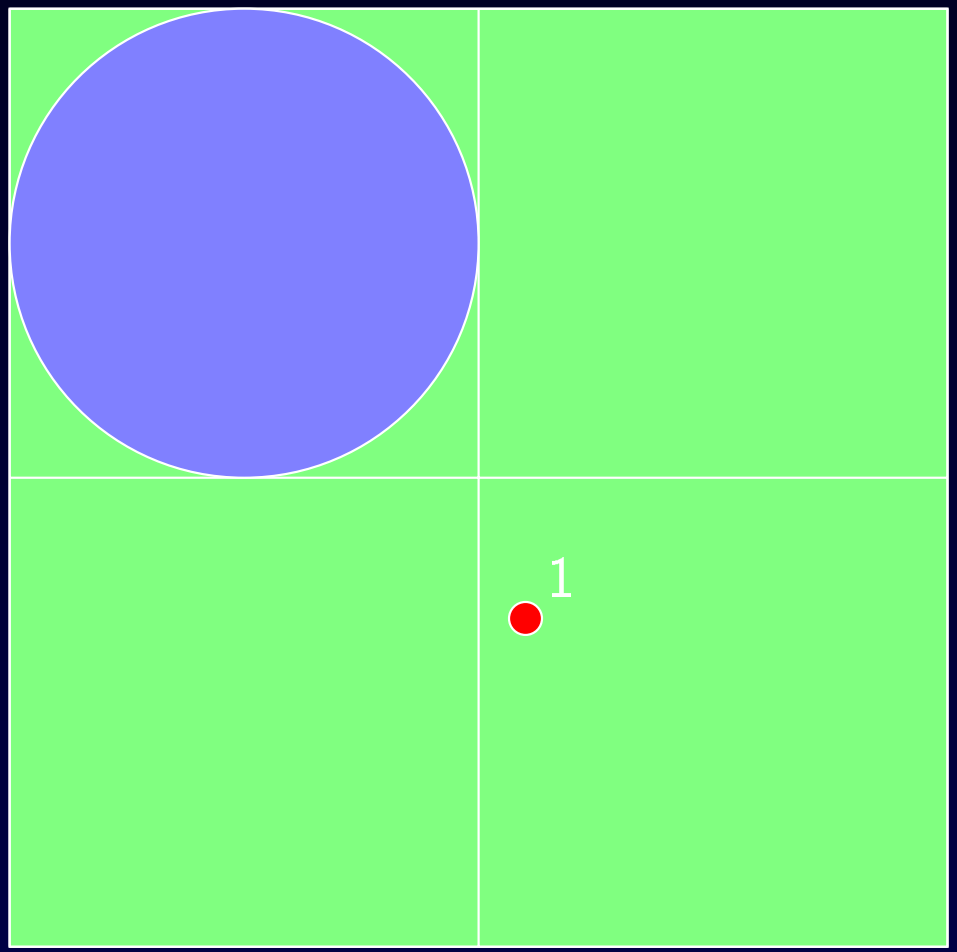
- Many statistical problems can be framed as integrals
  - Finding p-values: integrate high-dimensional likelihood
  - Finding MLE: integrate high-dimensional likelihood given up to normalizing constant
  - Finding Bayes factors: integrate high-dimensional posterior
- Typically dimension is number of data points
- Problem difficult when statistical model/likelihood does not have independence
- Monte Carlo can be used to estimate those integrals if we can draw from statistical model/posterior
- MCMC is a heuristic aimed at getting those draws
- Only a heuristic because mixing time is usually unknown

Saturday Morning Breakfast Cereal by Zach Weinersmith

Samples to Bernoulli draws

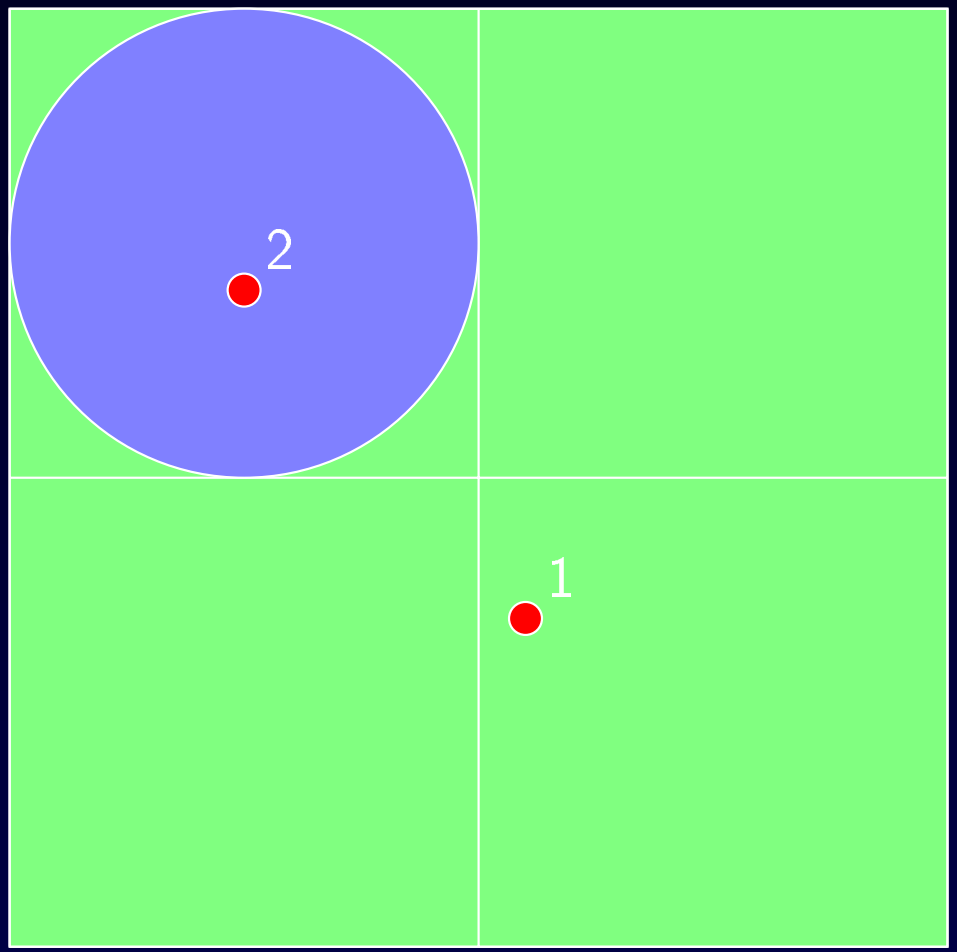


Samples to Bernoulli draws



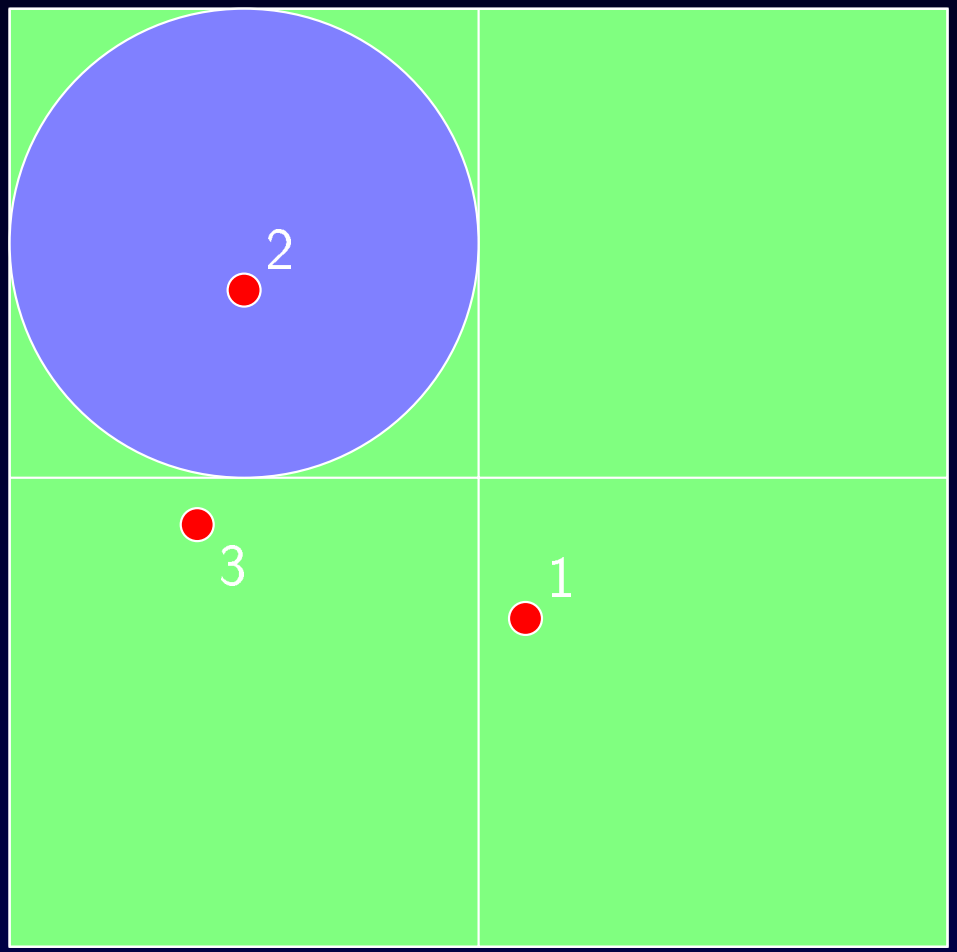
0

Samples to Bernoulli draws



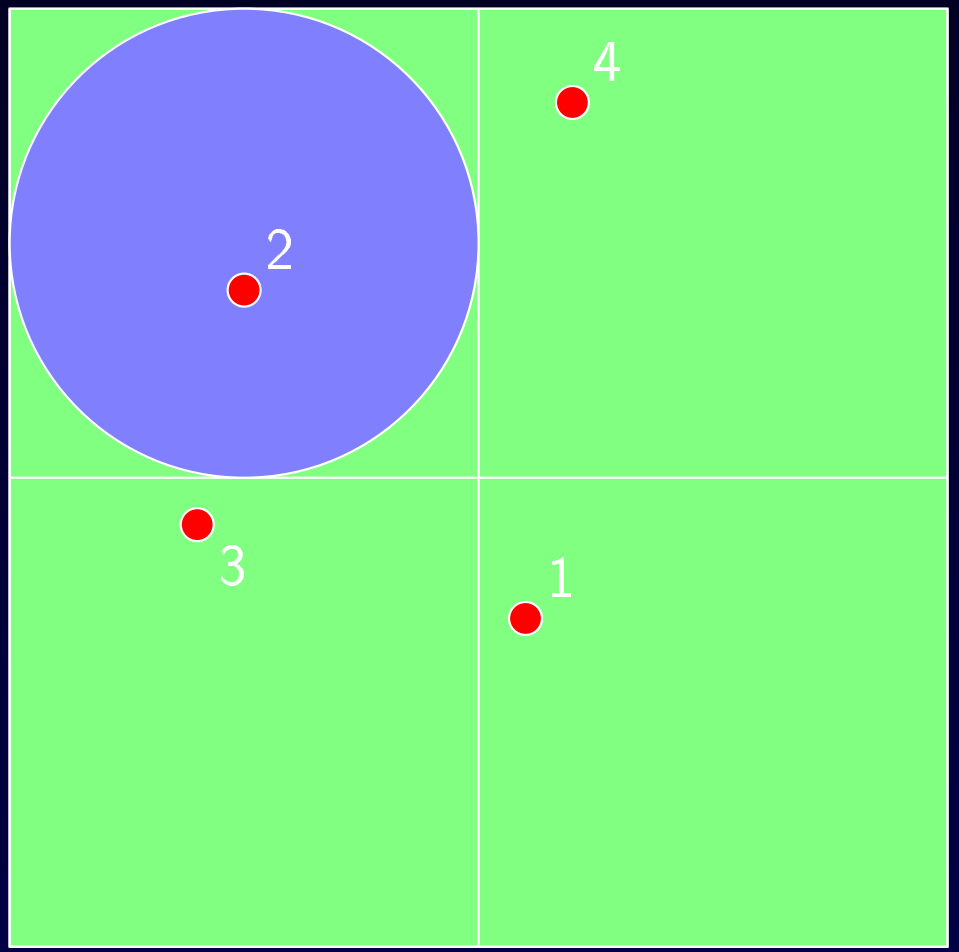
01

Samples to Bernoulli draws



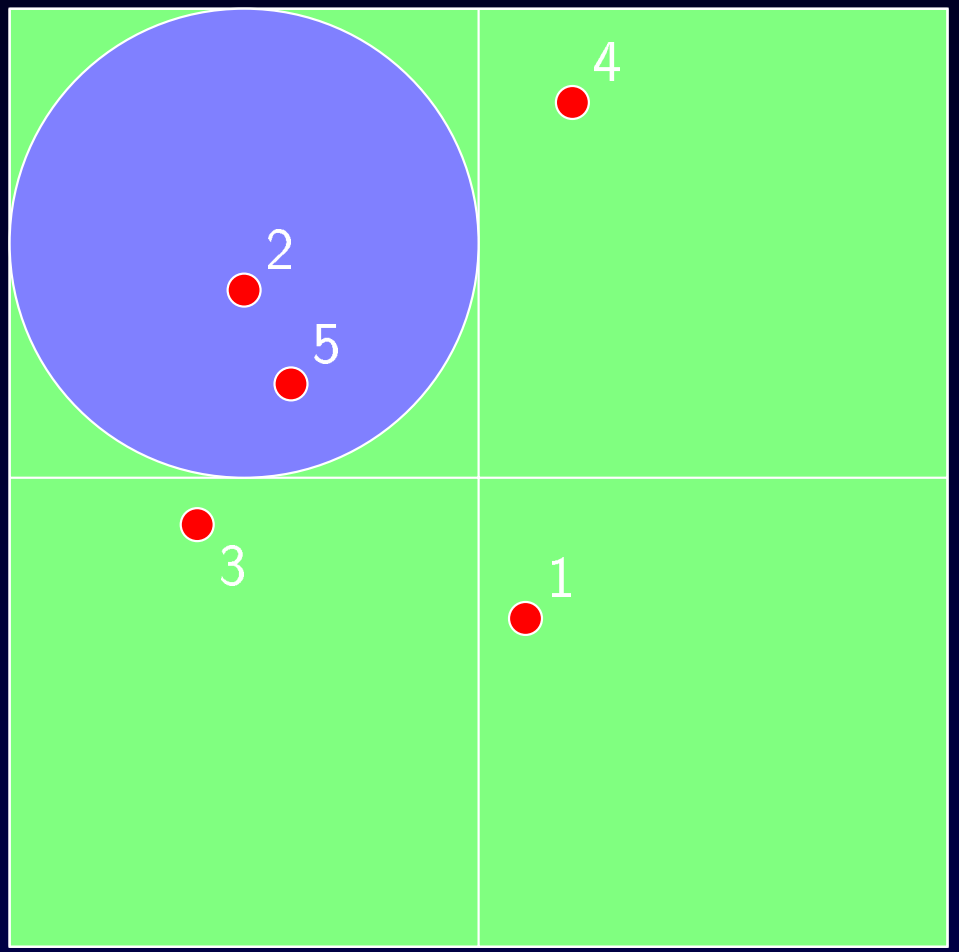
010

Samples to Bernoulli draws



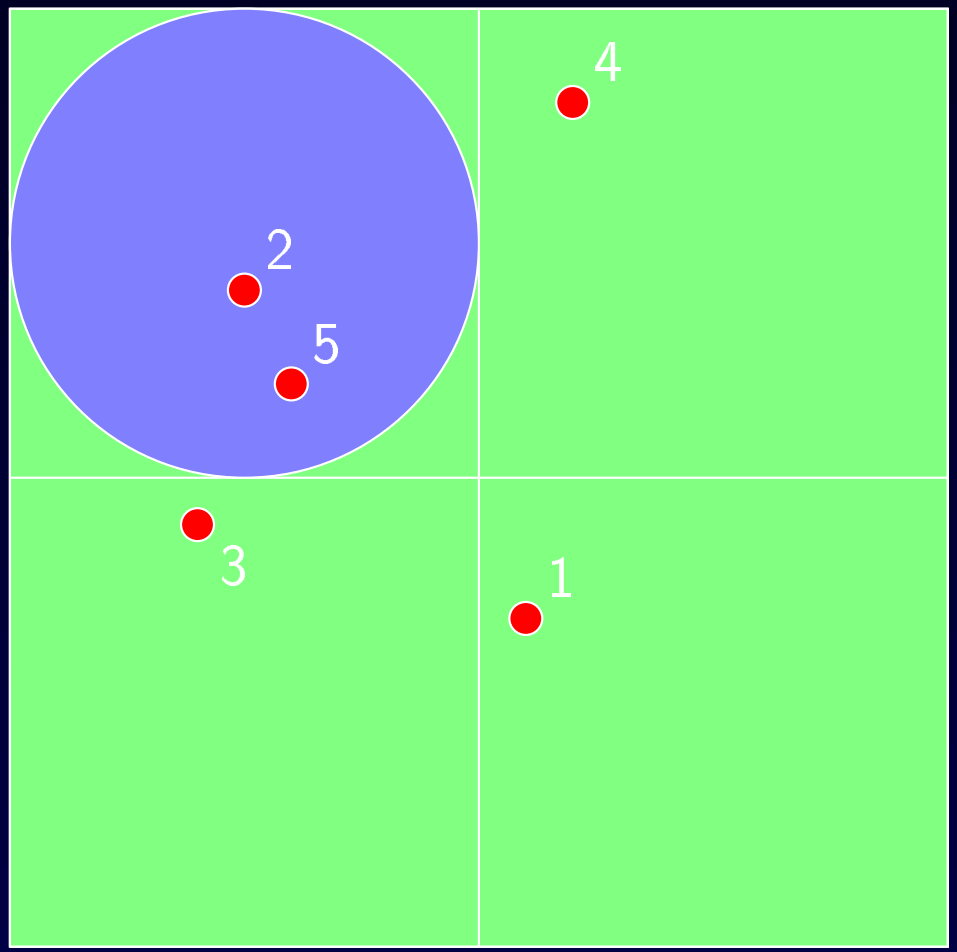
0100

Samples to Bernoulli draws



01001

Samples to Bernoulli draws



01001

Estimate

$\tau/8 = \pi/16 \approx 2/5$

Using AR to integrate

Start with

$$R_1, R_2, R_3, \ldots \sim \text{Unif}(B)$$

create

$$B_1, B_2, B_3, \ldots \sim \text{Bern}(\text{area}(A)/\text{area}(\text{B}))$$

by making

$$B_i = \mathbb{1}(R_i \in A)$$

Estimate $\hat{p}$ for mean of B_i gives

$$\hat{\text{area}}(A) = \hat{p} \cdot \text{area}(B)$$

Can also use with dominating densities

Given $f_X \leq f_W$, ability to draw from f_W:

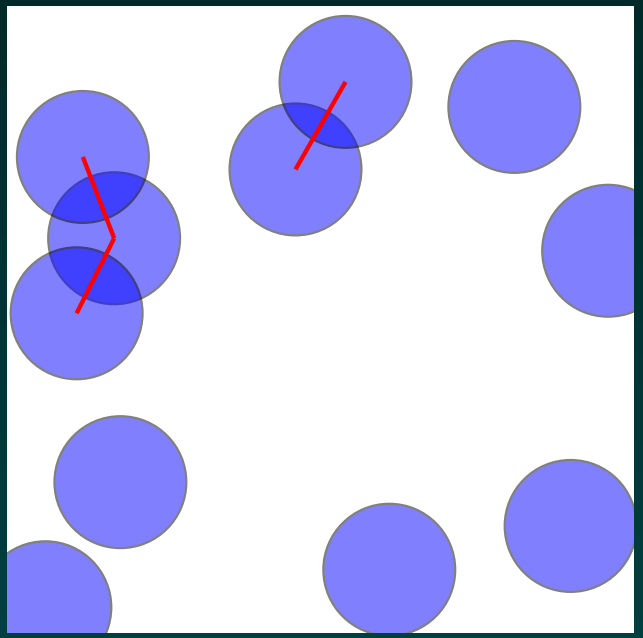
1. Draw $W \leftarrow f_W$
2. Draw $Y \leftarrow \text{Unif}([0, f_W(W)])$
3. Output $B \leftarrow \mathbb{1}(Y \leq f_X(W)$

Fact

The output B is an indicator (Bernoulli) random variable with mean $\int f_X(x)\, dx / \int f_W(w)\, dw$.

Statistical inference with Strauss

Question: If r and γ are known, what is the Maximum Likelihood Estimator for λ for this data?



Penalty is γ^3

A note about densities

Properties for MLE

- Need density with respect to a measure that does not change with parameter
- Previously gave density of Strauss with respect to rate λ Poisson point process
- Instead use density of Strauss with respect to unit rate PPP
- New normalized density:

$$g_{\lambda,\gamma,r}(p) = \frac{\lambda^{\#(p)}\gamma^{t(p)}}{Z_{\lambda,\gamma,r}}.$$

- MLE hard to find because $Z_{\lambda,\gamma,r}$ hard to calculate

---

Here $\#(S)$ counts the number of elements is the set S

Normalizing constant as integral

Strauss normalizing constant

$$Z_{\lambda,\gamma,r} = \int_{\Omega} \lambda^{\#(s)} \gamma^{t(s)} \, ds$$

Fact

Well known about PPP: for $\gamma = 1$ and region of area a,

$$Z_{\lambda,1,r} = \exp(\lambda a)$$

Using AR to estimate $Z_{\lambda,\gamma,r}$

Use AR with dominating density

$$h(s) = \lambda^{\#(s)} \geq \lambda^{\#(s)}\gamma^{t(s)}$$

For $W \sim h(s)$ and $U \sim \text{Unif}([0,1])$,

$$\begin{aligned}
\mathbb{P}(\text{accept } W) &= \mathbb{P}(U \leq \gamma^{t(W)}) \\
&= \mathbb{E}[\mathbb{1}(U \leq \gamma^{t(W)})] \\
&= Z_{\lambda,\gamma,r} \exp(-\lambda a)
\end{aligned}$$

Using Bernoulli random variables for perfect integration

Bernoulli random variables

- Note $\mathbb{1}(U \leq \gamma^{t(W)})$ is an **indicator** or **Bernoulli** r.v.
- Want a robust estimator for mean of a Bernoulli
- Here robust means error is controlled for all means in $[0, 1]$
- In fact, we can do better and set the distribution of the relative error (H. 2017, Feng, H., Ruan 2018)

User specified relative error

Definition

Say that an estimate $\hat{a}$ for a has **user specified relative error** or **USRE** if the distribution of $\hat{a}/a$ does not depend on $\hat{a}$, but only on parameters specified by the user in constructing $\hat{a}$.

Idea for USRE for Bernoulli means

1. Use Bernoulli r.v.'s to build Geometric r.v.'s with mean $1/p$
2. Use Geometric r.v.'s to build Exponential r.v.'s with mean $1/p$
3. Use USRE (MOM for instance) for Exponentials to estimate p

Gamma Bernoulli Approximation Scheme

Gamma_Bernoulli_Approximation_Scheme

Input: k *Output:* $\hat{p}_k$

---

1) $S \leftarrow 0,\ R \leftarrow 0.$
2) Repeat
3) $\quad X \leftarrow \mathsf{Bern}(p),\ A \leftarrow \mathsf{Exp}(1)$
4) $\quad S \leftarrow S + X,\ R \leftarrow R + A$
5) Until $S = k$
6) $\hat{p}_k \leftarrow (k-1)/R$

M. Huber, A Bernoulli mean estimate with known relative error distribution, *Random Struc. & Alg., arXiv:1309.5413, 50:173–182, 2017*

Relative error for GBAS

Theorem

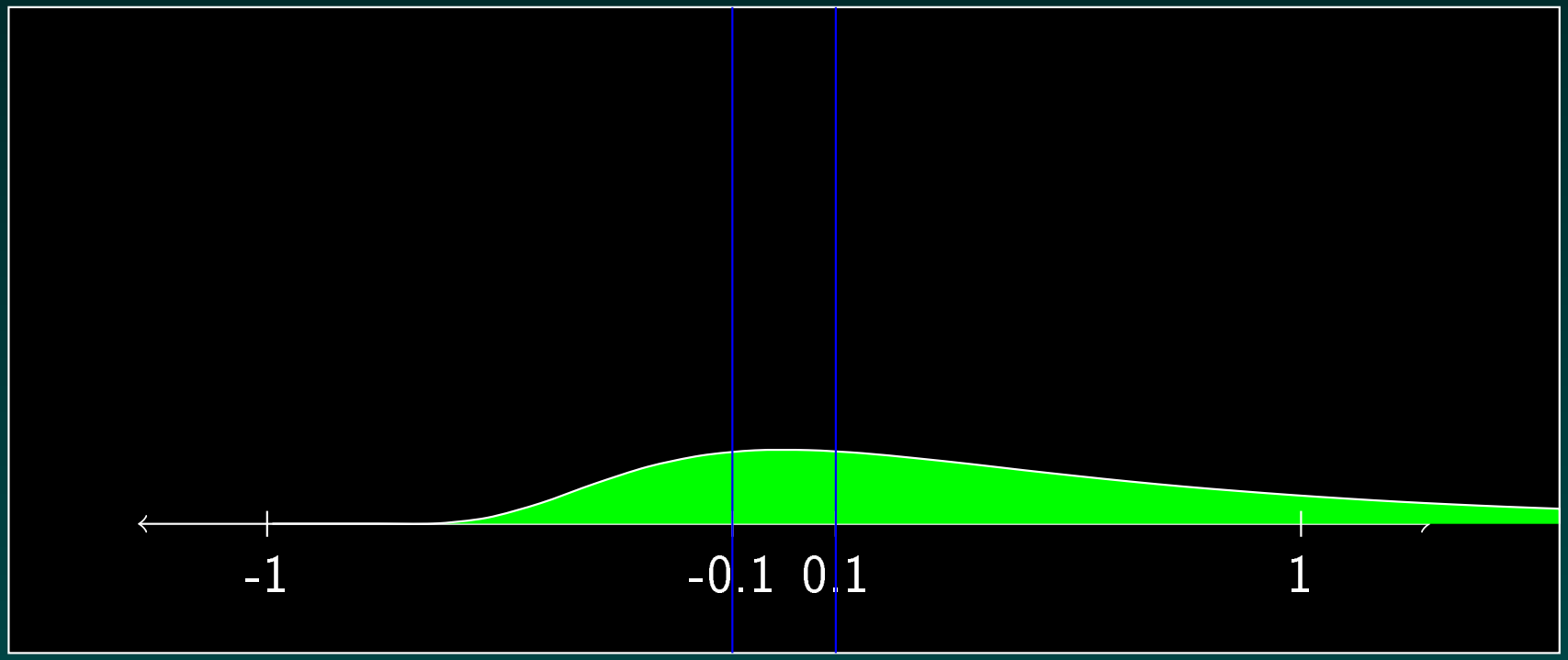
For GBAS, $\mathbb{E}[\hat{p}_k] = p$. Moreover for all $p \in [0, 1]$, the relative error is distributed as

$$\frac{p}{\hat{p}_k} \sim \textit{Gamma}(k, k-1)$$

Visualizing the relative error

User set $k = 5$

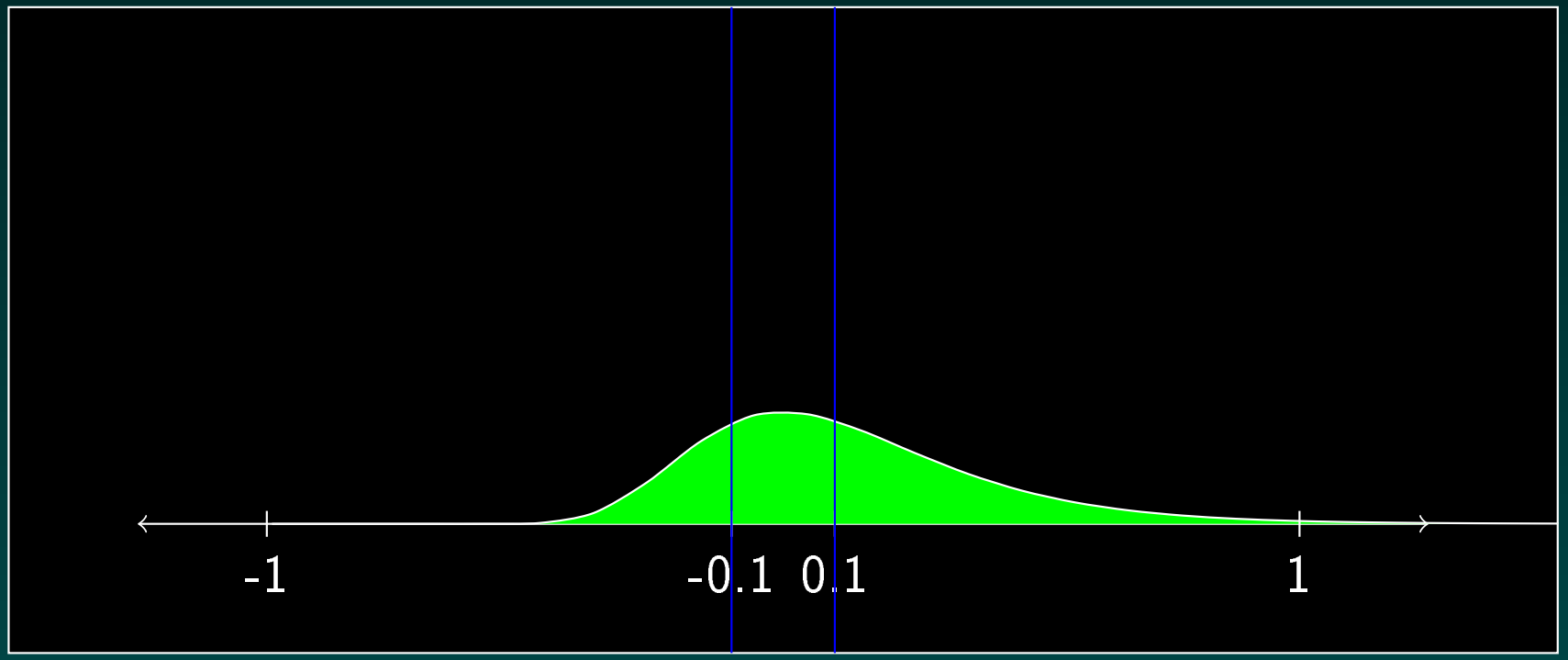
$\mathbb{P}(|\text{rel err}| > 0.1) \approx 92.6\%$



Visualizing the relative error

User set $k = 20$

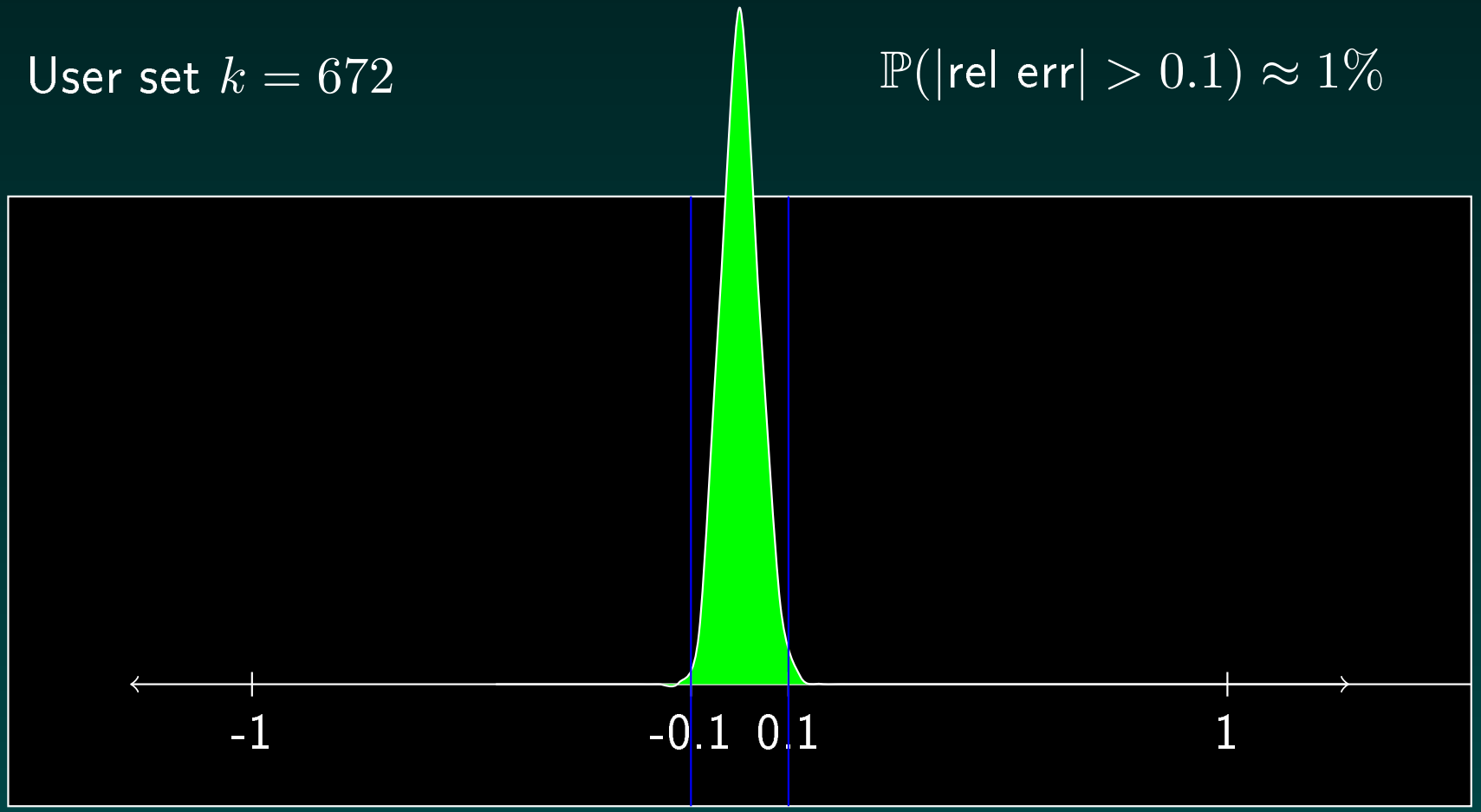
$\mathbb{P}(|\text{rel err}| > 0.1) \approx 66.1\%$



Visualizing the relative error

User set $k = 672$

$\mathbb{P}(|\text{rel err}| > 0.1) \approx 1\%$



Computing the error exactly with $c = 1$

```
pgamma(1/0.9,671,670,lower.tail=FALSE) + pgamma(1/1.1,671,670)
```

$$0.01002245$$

```
pgamma(1/0.9,672,671,lower.tail=FALSE) + pgamma(1/1.1,672,671)
```

$$0.00996848$$

Set $k = 672$ w/ prob 0.4159678, $k = 671$ w/ prob 0.5840322

$\mathbb{E}[k] = 671.416$ gives error 0.01000000

Accuracy of GBAS

Definition

An algorithm is an (ϵ, δ)-**randomized approximation scheme** if for all $\epsilon, \delta > 0$, the output $\hat{a}$ of the algorithm satisfies (with respect to true answer a)

$$\mathbb{P}\left(\left|\frac{\hat{a}}{a} - 1\right| > \epsilon\right) < \delta.$$

Performance of GBAS

Theorem

Let $\epsilon, \delta > 0$ *set* $c = 2\epsilon/[(1-\epsilon^2)\ln(1+2\epsilon/(1-\epsilon))] \approx 1 + (2/3)\epsilon^2$ *Then let* $\hat{a}$ *be the output of GBAS with* $k = \lceil 2\epsilon^{-2}\ln(\delta^{-1})\rceil$ *divided by* c*. Then* $\hat{a}$ *is an* $(\epsilon, \delta) - ras$*.*

J. Feng, M. Huber, and Y. Ruan. Monte Carlo with user-specified relative error. In P. W. Glynn and A. Owen, editors, *Proceedings in Mathematics & Statistics: Monte Carlo and Quasi-Monte Carlo methods*, 241, chapter 12. Springer. To appear.

Remarks about GBAS performance

- $\epsilon^{-2} \ln(\delta^{-1})$ factor part of Monte Carlo
- Only way to improve that is through Quasi-Monte Carlo
- Factor of 2 same as in denominator of normal distribution
- Use of c biases estimator slightly but makes it slightly faster (in higher order terms) than CLT

A natural next step

By the Fundamental Theorem of Probability
(aka The Law of Total Expectation)

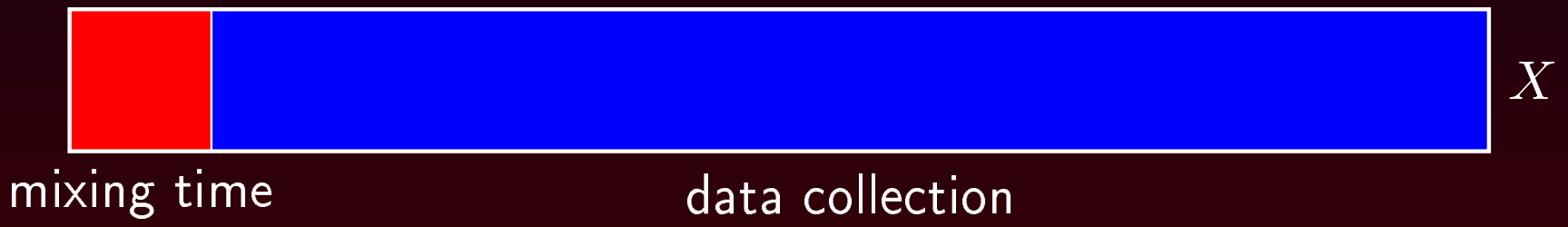
$$\mathbb{E}[\mathbb{1}(U \leq \gamma^{t(W)})] = \mathbb{E}[\mathbb{E}[\mathbb{1}(U \leq \gamma^{t(W)})|W]] = \mathbb{E}[\gamma^{t(W)}]$$

- Replacing a random variable with its expectation preserves unbiasedness, reduces variance
- Conditioning to reduce variance is often called *Rao-Blackwellization*
- Also the basis of *Importance Sampling*
- In Part IV I will discuss perfect integration with non-Bernoulli r.v.'s that apply to IS methods

Perfect simulation and MCMC

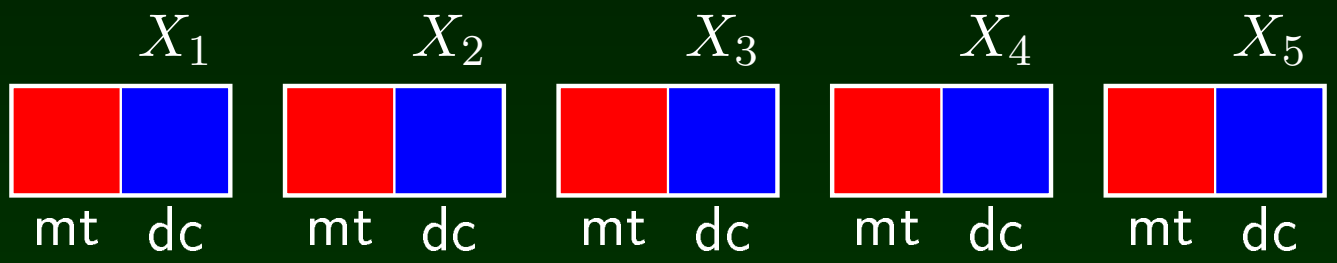
MCMC practice

- For complex hierarchical models MCMC often only viable way
- Often MCMC is run by optimists



Optimists devote 10% of time to burnin/mixing time

MCMC practice for pessimists

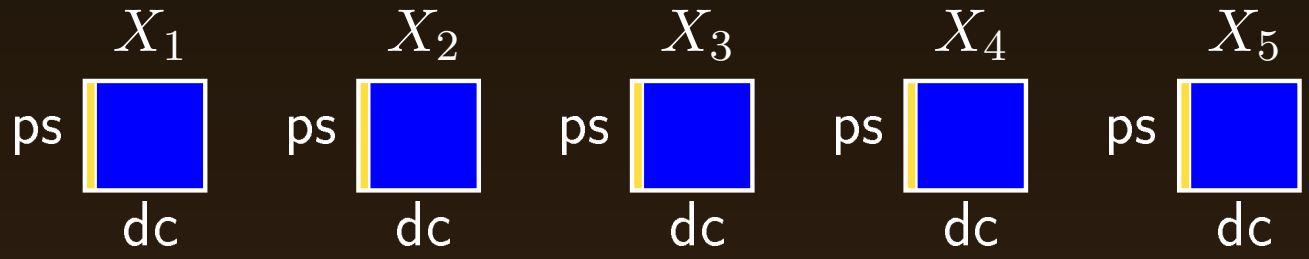


Without running your Markov chain multiple times, impossible to know anything about $\mathbb{V}(X_i)$

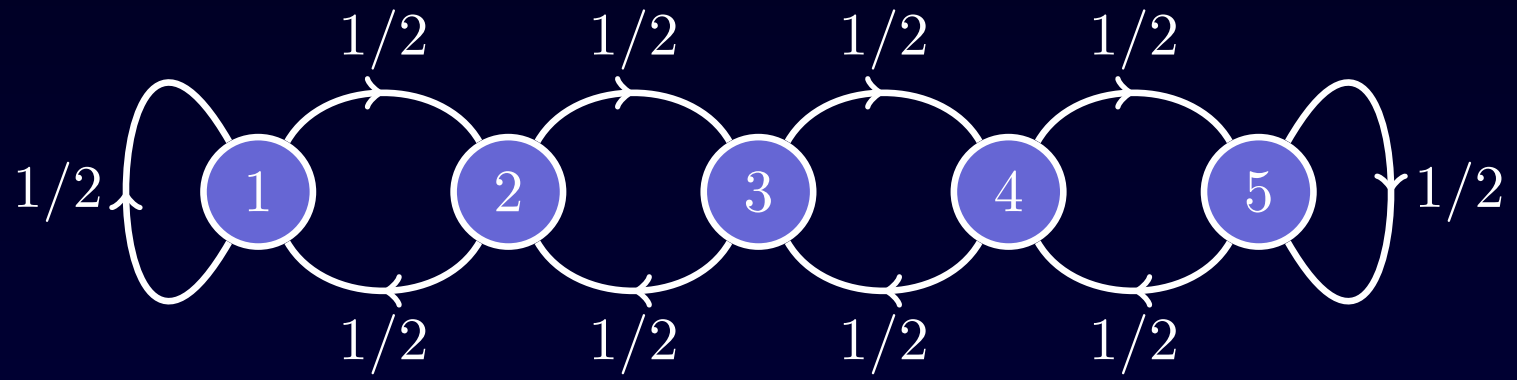
- Spending half our steps on mixing time
- Some sense "correct" amount: more and data collection decreases rapidly, less and gain at most twice as much data
- For safety half of time budget should be spent on mixing

How perfect simulation helps

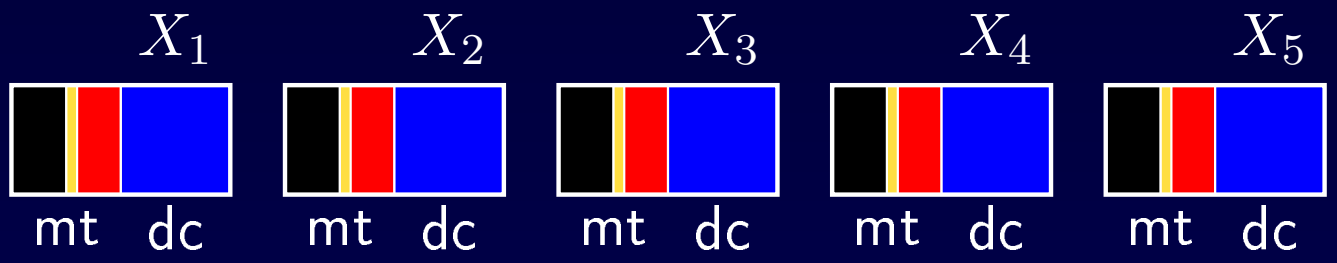
Replaces mixing entirely for Markov chain



Even imperfect but better starting point helps



Start state Unif($\{1, 5\}$) reduces mixing time by a factor of 4



Acceptance Rejection Variants

Variations on a theme

Modifications of basic AR

- Thinning Poisson processes (Preston 1977)
- Union bound AR
- Markov, Chebyshev, Chernoff AR (Bucklew 2004)
- Divide and Conquer AR (Min & Wang 2008)
- Partially Recursive AR (H. 2016)
- Minc's Inequality AR (H. 2006, H. & Law 2008)
- Exact draws from certain SDE's (Beskos et al. 2006)

If you can bound it, AR might be the answer!

Summary

Major points

- Acceptance Rejection is a widely applicable approach to generating samples exactly from a target distribution in a random amount of time
- Efficiency depends directly on close original problem is to an easy problem
- Can be directly turned into a robust integration method
- When used with MCMC, should use roughly one third to one half of computational time on perfect simulation
- AR can be exponential slow for high dimensional problems (will tackle this problem next lecture)