Monte Carlo: a tutorial

Art B. Owen

Stanford University

Some notation

X	random variable in $\mathbb{R}$
$\boldsymbol{X}$	random variable in $\mathbb{R}^d$
x, $\boldsymbol{x}$	observed values of X and $\boldsymbol{X}$
$\Pr(X = x)$	probability that random variable X takes value x
$X \sim F$ or p	X has distribution F or p
$\boldsymbol{X}_i \overset{\text{iid}}{\sim} F$	$\boldsymbol{X}_i$ independent and identically distributed as F
$\mathbb{E}(f(X))$	Expectation, e.g., $\int f(x)p(x)\,\mathrm{d}x$.
$\mathbf{U}(S)$	Uniform distribution on set S

there will be more notation

The MC idea(s)

Two versions:

Informal MC

Simulate some random process and watch what happens.

Formal MC

Express an unknown quantity μ as the solution

$$\begin{aligned}\mu &= \mathbb{E}(f(\boldsymbol{X})), \quad \boldsymbol{X} \sim p \\ &= \int f(\boldsymbol{x}) p(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}\end{aligned}$$

Then sample $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n \overset{\text{iid}}{\sim} p$ and take

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} f(\boldsymbol{X}_i).$$

Nagel-Schreckenberg traffic

- N vehicles in a circular track
- M possible positions $\{0, 1, \ldots, M-1\}$ mod M
- speed limit is $v_{\max}$, e.g., 5

The algorithm

For a car at $x \in \{0, 1, \ldots, M-1\}$ with velocity v and d spaces behind the car in front:

$$v \leftarrow \min(v+1, v_{\max})$$

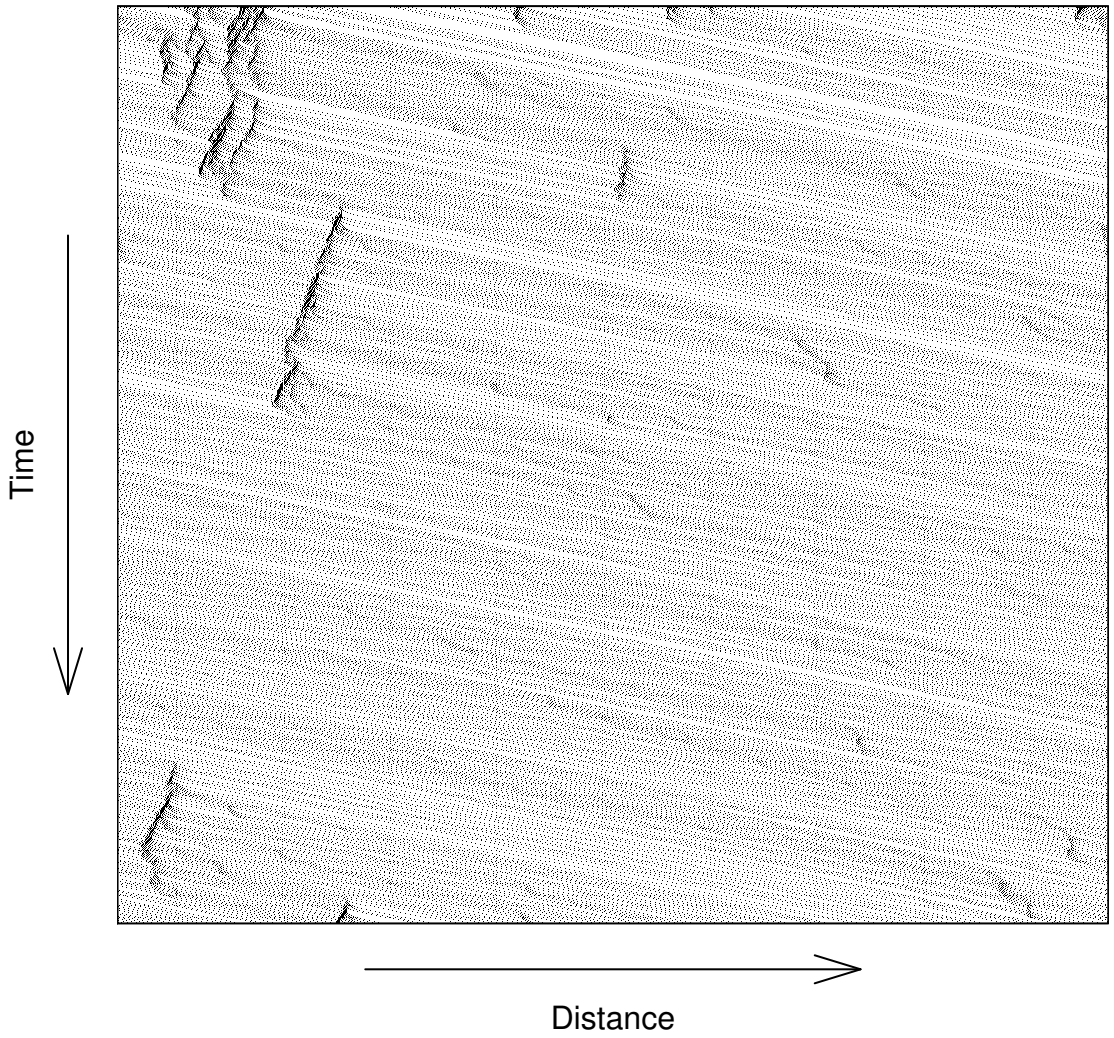
$$v \leftarrow \min(v, d-1)$$

$$v \leftarrow \max(0, v-1) \quad \text{with probability } p$$

$$x \leftarrow x + v \bmod M$$

Traffic results

Nagel–Schreckenberg traffic



- Dots = cars, start at top row
- Traffic jams 'emerge'
- and move backwards
- then disappear
- gaps move at the speed limit
- total flow not monotone in # cars
- one can elaborate the model
- (replace circular track by city map)

Average distance

For rectangle $R = [0, a] \times [0, b]$, let $\boldsymbol{X}, \boldsymbol{Z} \overset{\text{iid}}{\sim} \mathbf{U}(R)$. We want:

$$\begin{aligned}\mu(a, b) &= \mathbb{E}(\|\boldsymbol{X} - \boldsymbol{Z}\|) \\ &= \int_0^a \int_0^b \int_0^a \int_0^b \sqrt{(x_1 - z_1)^2 + (x_2 - z_2)^2}\, \mathrm{d}x_1\, \mathrm{d}x_2\, \mathrm{d}z_1\, \mathrm{d}z_2\end{aligned}$$

Quick and easy by Monte Carlo.

Also available analytically Ghosh (1951).

$$\mu(1, 3/5) = 0.4239 \qquad \text{from closed form}$$
$$\hat{\mu}(1, 3/5) = 0.4227 \qquad \text{from } n = 10{,}000 \text{ MC samples}$$

Relative error 0.0027

MC vs closed form

Exact solution generalizes to new a and b.

MC solution generalizes to more complicated regions or distances.

The closed form is brittle.

Properties of MC

1) MC works under minimal assumptions

   the desired mean must exist, then

   (law of large numbers) $\Pr(\lim_{n\to\infty} \hat{\mu}_n = \mu) = 1$

2) MC does not deliver extreme accuracy

   RMSE $\equiv \sqrt{\mathbb{E}((\hat{\mu} - \mu)^2)} = \sigma/\sqrt{n}$

   to cut RMSE by 10, we must raise n by 100

   a less serious flaw, when the problem is only posed to low accuracy

3) MC is very competitive in high dimensional or non-smooth problems

   (see next slide)

4) MC has extremely good error estimation

   (see slide after that)

MC vs. classic quadrature

For f on $[0,1]$ with $\geqslant r$ continuous derivatives,

quadrature gets $\int_0^1 f(x)\,\mathrm{d}x$ with error $O(n^{-r})$

e.g., $r = 4$ for Simpson's rule.

Iterated integrals

$$\int_{[0,1]^d} f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_0^1 \cdots \int_0^1 f(\boldsymbol{x})\,\mathrm{d}x_1 \cdots \mathrm{d}x_d$$

Use Fubini's rule in d dimensions:

$N = n^d$ points in a grid.

Error is $O(n^{-r}) = O(N^{-r/d})$

Monte Carlo

RMSE $= \sigma N^{-1/2}$ for any dimension d

Best possible rate is $O(N^{-1/2-r/d})$ Bakhvalov (1962)

MC is competitive for large d, low smoothness

Error estimation

$$\mu = \mathbb{E}(f(\boldsymbol{X})) \quad \text{and} \quad \sigma^2 = \mathrm{Var}(f(\boldsymbol{X})) \quad \boldsymbol{X} \sim p$$

Central limit theorem: $\hat{\mu} \dot{\sim} \mathcal{N}(\mu, \sigma^2/n)$

$$\lim_{n\to\infty} \Pr\left(\frac{\hat{\mu} - \mu}{\sigma/\sqrt{n}} \leqslant z\right) = \int_{-\infty}^{z} \frac{e^{-t^2/2}}{\sqrt{2\pi}}\,\mathrm{d}t$$

99% confidence interval

$$\Pr\left(\hat{\mu} - \frac{2.58\hat{\sigma}}{\sqrt{n}} \leqslant \mu \leqslant \hat{\mu} + \frac{2.58\hat{\sigma}}{\sqrt{n}}\right) = 0.99 + O(n^{-1})$$ Hall (1986) Ann. Stat.

Estimates $\hat{\mu}$ and $\hat{\sigma}$ from $f(\boldsymbol{X}_i)$

Estimation error at $O(n^{-1})$ is better than for $\hat{\mu}$ itself!

Example: MC for numerical integration

Consider the definite integral

$$\alpha = \int_{\mathbb{R}} f(x)\, dx.$$

Note that

$$\int_{\mathbb{R}} f(x)\, dx = \int_{\mathbb{R}} \frac{f(y)}{p(y)}\, p(y)\, dy = \int_{\mathbb{R}} h(y)\, p(y)\, dy = \mathbb{E}\left\{h(Y)\right\},$$

where Y is a random variable with density $p(y)$ and $h(y) = f(y)/p(y)$. Thus, it is possible to estimate α by

$$\widehat{\alpha} = \frac{1}{M} \sum_{m=1}^{M} h(Y_m),$$

where $Y_1, \cdots, Y_M$ are independent observations of Y.

Example: MC for numerical integration

Consider the integral

$$\begin{aligned} \alpha &= \int_0^{2\pi} \sin^2(x)\, dx \\ &= \int_0^{2\pi} 2\pi \sin^2(x) \frac{1}{2\pi}\, dx \\ &= \int_{\mathbb{R}} h(y)\, p(y) dy \end{aligned}$$

where

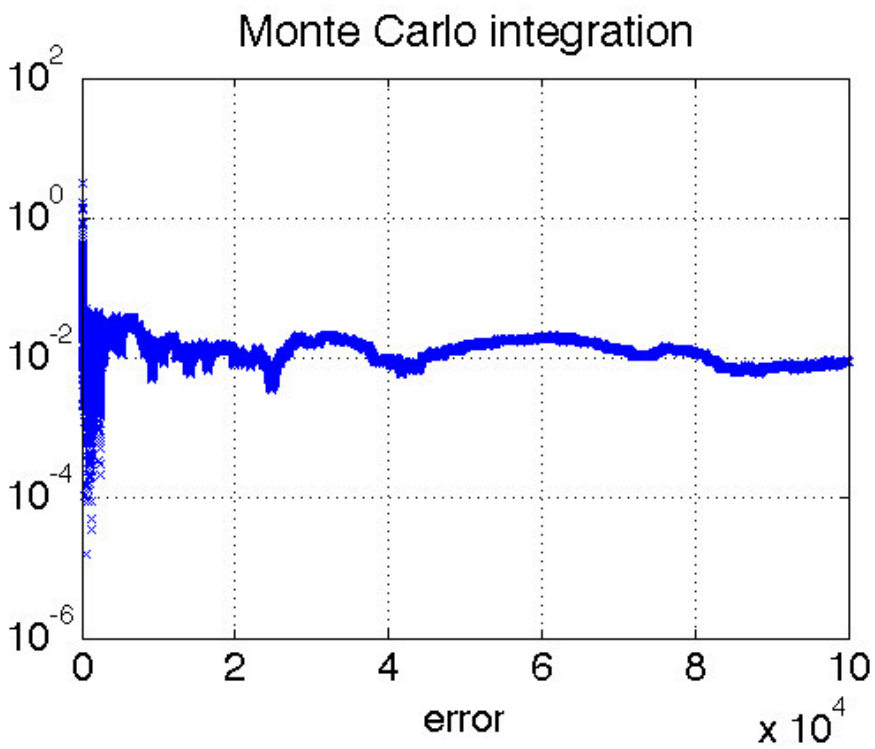
- $h(y) = 2\pi \sin^2(y)$
- $p(y) = \frac{1}{2\pi} \mathbb{1}_{(0,2\pi)}(y)$ i.e., $Y \sim \mathcal{U}(0, 2\pi)$.

Example: MC for numerical integration

Table: Evaluation of the integral via Monte Carlo Method.

samples	$\widehat{\alpha}$	$\lvert\alpha - \widehat{\alpha}\rvert$ **
10	3.7543	3.1392
100	3.2420	0.1004
1000	3.1349	0.0066
10000	3.1256	0.0159
100000	3.1324	0.0091

**Analytical solution $\alpha = \pi$

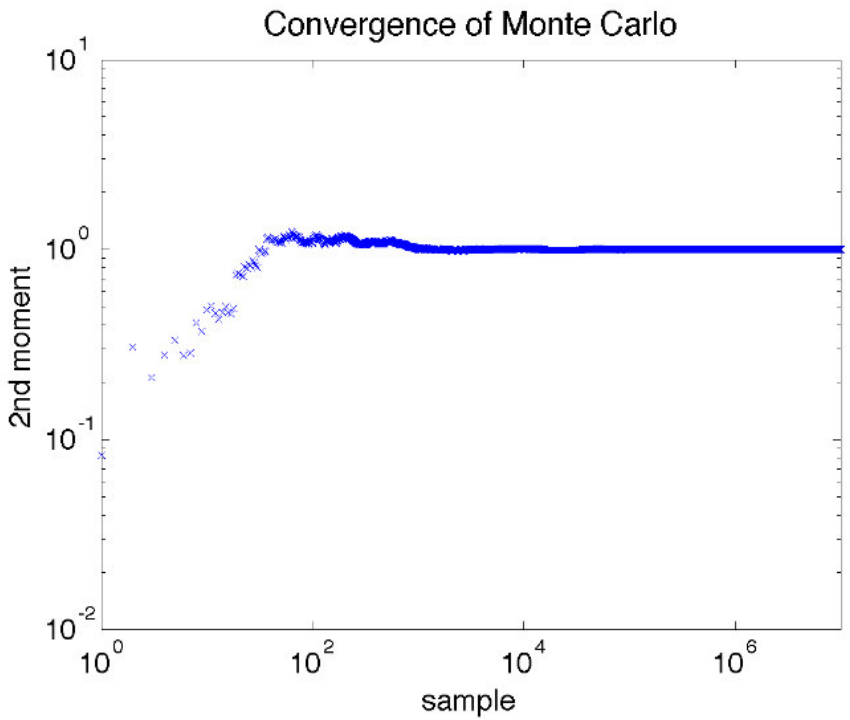


Analysis of Monte Carlo convergence

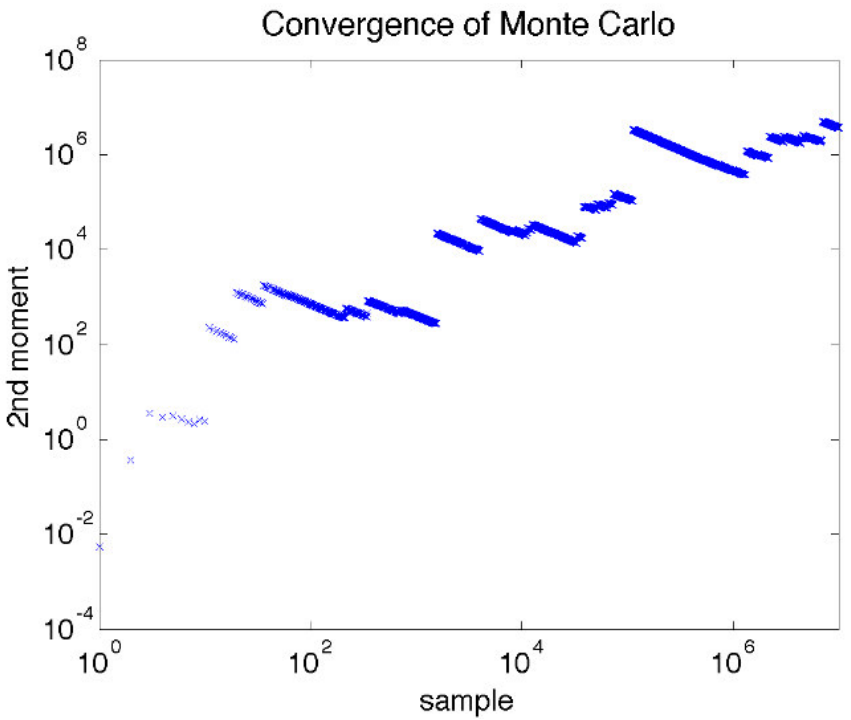
Convergence metric (2nd moment estimator):

$$\widehat{\alpha}_2^{(M)} = \frac{1}{M} \sum_{m=1}^{M} X_m^2$$

$X \sim \mathcal{N}(0,1)$



$X \sim \mathit{Cauchy}(0,1)$



Analysis of Monte Carlo convergence

Why is this behavior observed?

- $X \sim \mathcal{N}(0, 1)$

- $X \sim Cauchy(0, 1)$

Why is this behavior observed?

- $X \sim \mathcal{N}(0,1)$

$$\mathbb{E}\left\{X^2\right\} = \int_{-\infty}^{+\infty} \frac{x^2}{\sqrt{2\pi}\,\sigma} \exp\left\{-\frac{(k-\mu)^2}{2\,\sigma^2}\right\} dx = \mu^2+\sigma^2 < +\infty$$

- $X \sim \mathit{Cauchy}(0,1)$

Why is this behavior observed?

- $X \sim \mathcal{N}(0,1)$

$$\mathbb{E}\left\{X^2\right\} = \int_{-\infty}^{+\infty} \frac{x^2}{\sqrt{2\pi}\,\sigma} \exp\left\{-\frac{(k-\mu)^2}{2\,\sigma^2}\right\} dx = \mu^2+\sigma^2 < +\infty$$

- $X \sim Cauchy(0,1)$

$$\mathbb{E}\left\{X^2\right\} \propto \int_{-\infty}^{\infty} \frac{x^2}{1+x^2}\, dx = \int_{-\infty}^{\infty} dx - \pi = \infty$$

Randomness

- We need a source of randomness
- Physics offers several
- But true random numbers are not reproducible (or compressable (Kolmogorov))
- Some physical RNGs fail tests of randomness (Marsaglia)

Pseudo-randomness

- Most MC uses pseudo-random numbers
- I.e., deterministic computer programs that simulate randomness, reproducibly.
- There are many high quality and well tested RNGs. I like

  1) the Mersenne Twister of Matsumoto, Nishimura (1998),

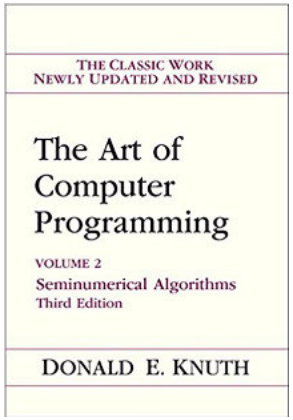
  2) and MRG32k3a of L'Ecuyer (1999),

  and there are other high quality RNGs.

Today's MC would be impossible without the efforts of people who work on the algebra of finite fields.

An early attempt to contruct a RNG

Donald E. Knuth



Knuth's "Super-random" algorithm

1. Take N to be the most significant digit of X, a 10-digit decimal number. Steps 2-13 are repeated exactly $N + 1$ times.
2. Let M be the second most significant digit of X. Jump to step $3 + M$.
3. If $X < 5 \times 10^9$, set $X = X + 5 \times 10^9$.
4. Replace X by $\lfloor X^2/10^5 \rfloor \mod 10^{10}$.
5. Replace X by $(1001001001 \times X) \mod 10^{10}$.
6. If $X < 10^8$ then $X = X + 9814055677$; otherwise $X = 10^{10} - X$.
7. Interchange the lower-order five digits of X with the higher-order five digits of X.
8. Replace X by $(1001001001 \times X) \mod 10^{10}$.
9. For each digit d of X, decrease d by 1 if $d > 0$.
10. If $X < 10^5$, set $X = X^2 + 99999$; otherwise $X = X - 99999$.
11. If $X < 10^9$, set $X = 10 \times X$ and repeat this step.
12. Replace X by the middle 10 digits of $X(X - 1)$.
13. If $N > 0$, decrease N by one and return to step 2. If $N = 0$, the algorithm terminates with the current value of X as the next value in the sequence.

R. W. Shonkwiler, and F. Mendivil, *Explorations in Monte Carlo Methods*, Springer, 2009.

Sequence generated by Knuth's algorithm

Sometimes complexity hides simple behaviour:

- The first time Knuth ran this algorithm it almost immediately converged to 6065038420 (a fixed point for the algorithm).
- After this, when he ran it with a different starting value, it converged to a cycle having length 3178!

Lessons from this example:

- Complexity is not a substitute for randomness.
- Random numbers should not be generated with a method chosen at random!

R. W. Shonkwiler, and F. Mendivil, *Explorations in Monte Carlo Methods*, Springer, 2009.

Linear Congruential Generators (LCG)

Fix the positive integer parameters m, a, c, and X_0

- m – modulus, $0 < m$
- a – multiplier, $0 \leq a < m$
- c – increment, $0 \leq c < m$
- X_0 – seed, $0 \leq x_0 < m$

Given an integer X_n , the following iteration is computed

$$X_{n+1} = (a\,X_n + c) \mod m, \qquad n \in \mathbb{N}.$$

R. W. Shonkwiler, and F. Mendivil, *Explorations in Monte Carlo Methods*, Springer, 2009.

Sequence generated by LCG

Example: $(m = 8,\ a = 5,\ c = 1,\ x_0 = 0)$

$$
\begin{array}{lll}
X_0 = 0 & X_8 = 0 & X_{16} = 0 \\
X_1 = 1 & X_9 = 1 & X_{17} = 1 \\
X_2 = 6 & X_{10} = 6 & X_{18} = 6 \\
X_3 = 7 & X_{11} = 7 & X_{19} = 7 \\
X_4 = 4 & X_{12} = 4 & X_{20} = 4 \\
X_5 = 5 & X_{13} = 5 & X_{21} = 5 \\
X_6 = 2 & X_{14} = 2 & X_{22} = 2 \\
X_7 = 3 & X_{15} = 3 & X_{23} = 3
\end{array}
\quad \ldots
$$

- maximum number of possible different outcome is equal to m
- in real applications, a very large m is taken (e.g. $m = 2^{32}$)

R. W. Shonkwiler, and F. Mendivil, *Explorations in Monte Carlo Methods*, Springer, 2009.

Choosing a Good Random Number Generator

- Like choosing a new car: for some people speed is preferred, while for others robustness and reliability are more important.
- For Monte Carlo simulation distributional properties of RNG are paramount, whereas in coding/cryptography unpredictability is crucial.
- As with cars, there are many poorly designed and outdated models available that should be avoided. Several of standard generators that come with popular programming languages and computing packages can be appallingly poor.

D. P. Kroese, T. Taimre, and Z. I. Botev, *Handbook of Monte Carlo Methods*, Wiley, 2011.

Basic (pseudo) RNGs

First: make sure your software is using a good and thoroughly tested RNG.

Typical usage

$x \leftarrow \texttt{rand}()$ // x is now a simulated $\mathbf{U}(0,1)$

```
rand:
    state ← update(state)
    return f(state)
```

Period

The state space is finite $\Rightarrow$ the RNG eventually repeats: $x_{i+M} = x_i$ for period M.

Use no more than $\sqrt{M}$ draws in one simulation. (L'Ecuyer)

Seed

Setting a seed (e.g. $\texttt{setseed}(s)$) lets you control the initial state of the RNG.

Getting the seed (e.g. $s \leftarrow \texttt{getseed}()$) lets you save state for later.

Streams

Sophisticated use of RNGs requires multiple independent streams

$$X_i^{(s)} \overset{\text{iid}}{\sim} \mathbf{U}(0,1), \quad i = 1, \ldots, N_s, \quad s = 1, \ldots, S$$

Coupling

Use stream 1 for coffee shop customer arrival:

- random wait . . . 148.7 seconds for next customer group
- it has 3 customers
- first one orders double decaf soy latte with bacon bits
- and so on until 10 pm

Use stream 2 for service times. Now compare two store configs on given customer stream.

Processes

Simulate S physical systems for N time steps

Use one stream per system

Later add systems (larger S) or do longer simulations (larger N) compatibly

Parallelism

May want one stream per processor.

Challenging to seed them all.

Still an area of active research.

Hellekalek (1998) warns “Don’t trust parallel Monte Carlo!”

I like RngStreams of L’Ecuyer, Simard, Chen & Kelton (2002)

lots of long random streams, tested

Also, the Mersenne Twister can be seeded

with output of a cryptographically secure RNG to make streams Matsumoto

Making random things

1) Random variables: $X \in \mathbb{R}$ but not $\mathbf{U}(0,1)$ (e.g. $\mathcal{N}(0,1)$)
2) Random vectors: $\boldsymbol{X} \in \mathbb{R}^d$
3) Random objects: graphs, permutations, projection matrices
4) Random processes: Brownian motion, Poisson, Cox, Chinese restaurant

Non-uniform random numbers

See Devroye (1986)

If the distribution has a name (normal, Poisson, Gamma, χ^2, beta, etc.)
it is probably already in Matlab or R or python or $\cdots$

Vectors and processes are another story

Inversion of the CDF

$F(x) = \Pr(X \leqslant x)$ invertible $\quad \Rightarrow \quad X \equiv F^{-1}(\mathbf{U}(0,1)) \sim F$

$$\begin{aligned}\Pr(X \leqslant x) &= \Pr(F^{-1}(U) \leqslant x) \\ &= \Pr(F(F^{-1}(U)) \leqslant F(x)) \\ &= \Pr(U \leqslant F(x)) \\ &= F(x)\end{aligned}$$

More generally

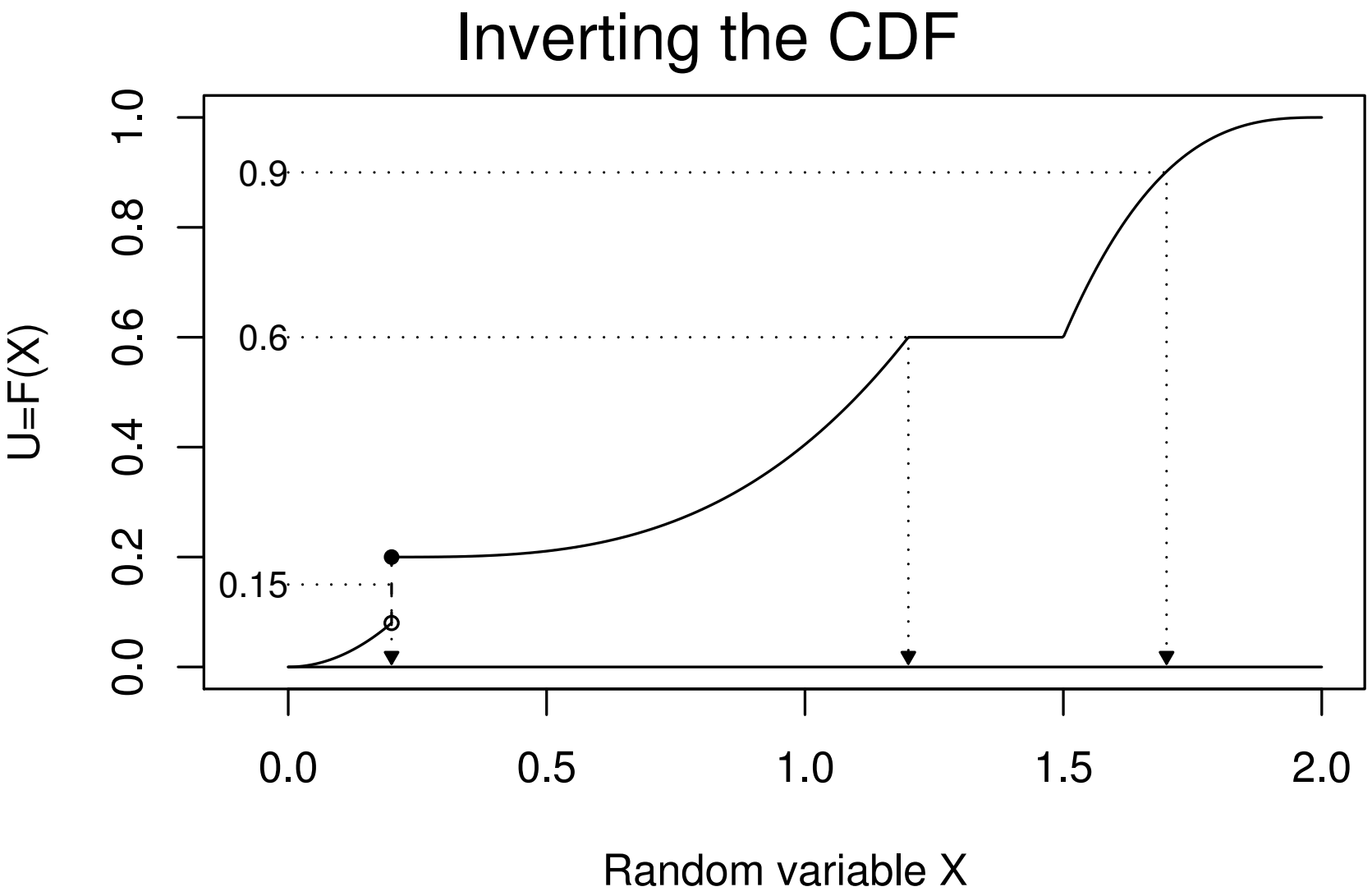
$F^{-1}(u)$ may not exist or may not be unique.

We solve both problems with:

$$F^{-1}(u) = \inf\{x \mid F(x) \geqslant u\}, \quad 0 < u < 1$$

$$X = F^{-1}(\mathbf{U}(0,1)) \sim F, \qquad \forall F$$

Inversion ctd



$$F^{-1}(u) = \inf\{x \mid F(x) \geqslant u\}$$

An example with the inverse transform method

Generate X from

$$p_X(x) = \begin{cases} 2\,x, & 0 \leq x \leq 1 \\ 0, & \text{otherwise.} \end{cases}$$

The CDF of X is given by

$$F_X(x) = x^2, \quad 0 \leq x \leq 1$$

and its inverse by

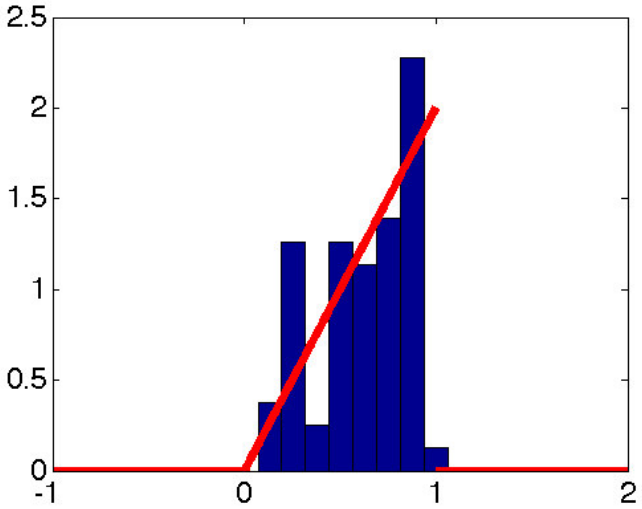
$$F_X^{-1}(u) = \sqrt{u}, \quad 0 \leq u \leq 1.$$

Algorithm :

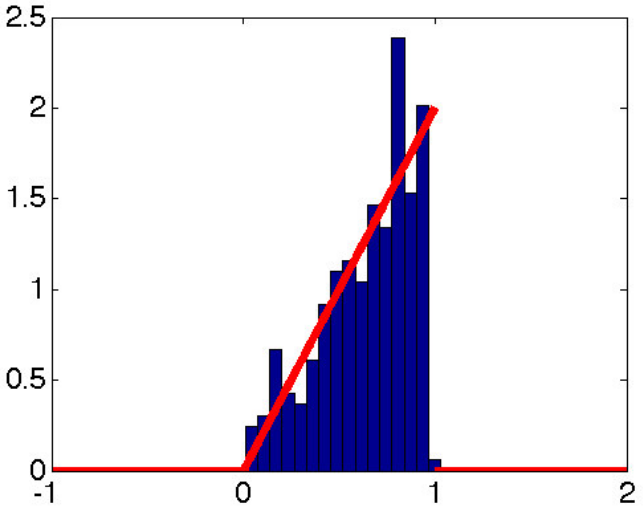
1. draw $U \sim \mathcal{U}(0,1)$
2. set $X = \sqrt{U}$

D. P. Kroese, T. Taimre, and Z. I. Botev, *Handbook of Monte Carlo Methods*, Wiley, 2011.

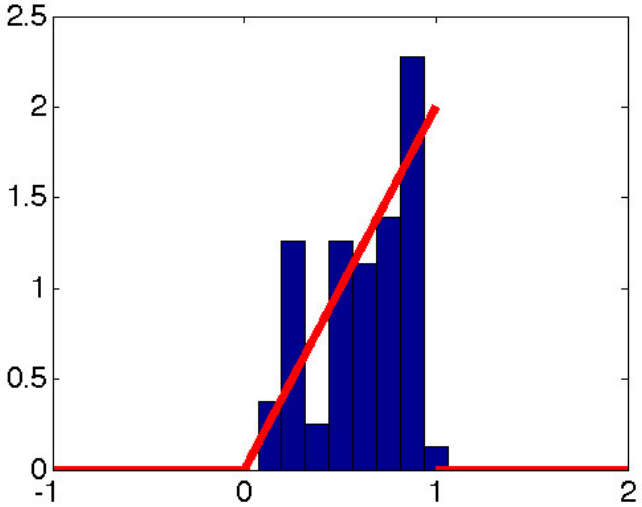
An example with the inverse transform method



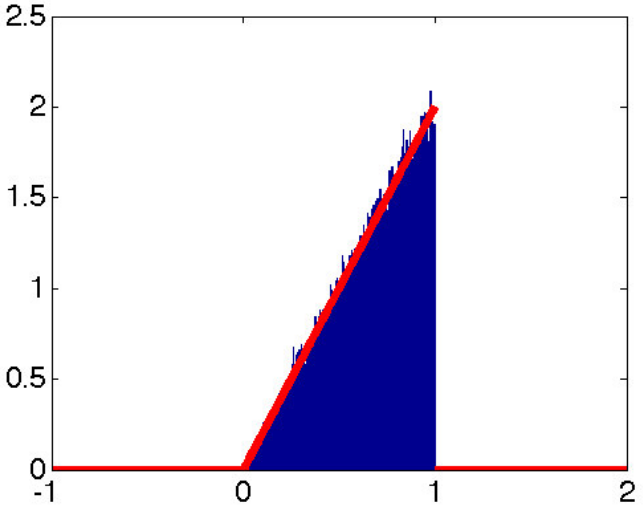
64 samples



256 samples



1024 samples



65536 samples

D. P. Kroese, T. Taimre, and Z. I. Botev, *Handbook of Monte Carlo Methods*, Wiley, 2011.

Many to one transformations

Box, Muller (1958)

$$Z = \sqrt{-2\log(U_1)}\cos(2\pi U_2) \sim \mathcal{N}(0,1)$$

Beta via ranks

Sort $U_1, \ldots, U_n \overset{\text{iid}}{\sim} \mathbf{U}(0,1)$ getting $U_{(1)} \leqslant U_{(2)} \leqslant \cdots \leqslant U_{(n)}$. Then

$$X = U_{(r)} \sim \text{Beta}(r, n-r+1)$$

$$f(x) \propto x^{r-1}(1-x)^{n-r+1} \quad 0 < x < 1$$

In reverse

Sample 10^{th} smallest of 10^{100} random variables via

$$F^{-1}(X), \quad X \sim \text{Beta}(10, 10^{100} - 9)$$

Devroye (1986) has a cornucopia of transformations.

Acceptance-rejection sampling

We want $X \sim f$

we can get $X \sim g$

where $f(x) \leqslant cg(x)$, known $c < \infty$.

Algorithm

Sample candidate $Y \sim g$

Accept $Y = y$ with probability $A(y) = f(y)/(cg(y)) \leqslant 1$

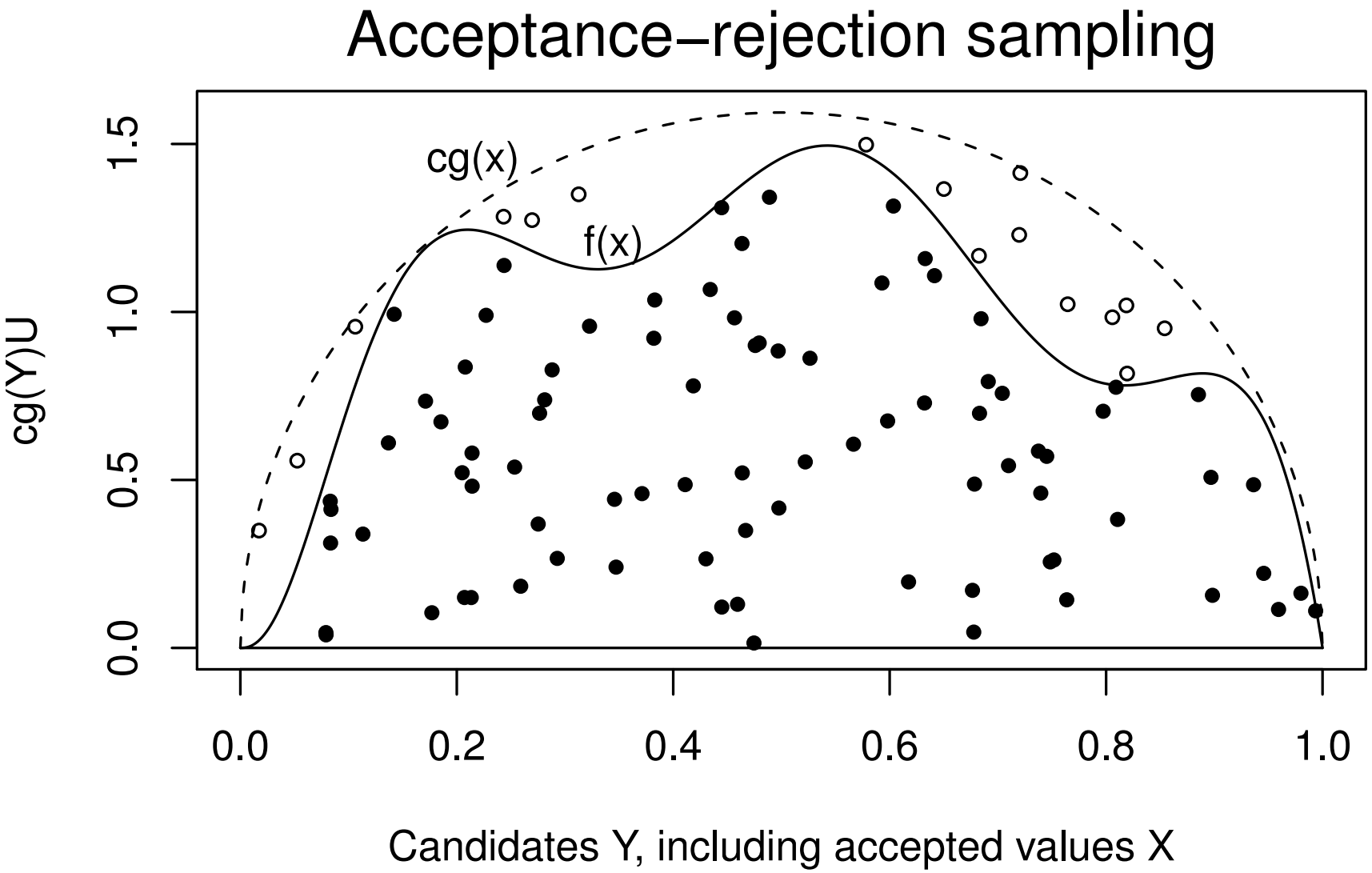
If accepted deliver $X = Y$. Else try again.

Outcome

Result has density $\propto g(x)A(x) = g(x)\frac{f(x)}{cg(x)} \propto f(x)$.

Nice proof in Knuth (1998)

Algorithm from von Neumann (1951)



The cost is proportional to $c = 1/\text{acceptance probability}$.

Geometry of acceptance-rejection

Define the region under Mh:

$$\mathcal{S}_M(h) = \{(x, z) \mid 0 \leqslant z \leqslant Mh(x),\ x \in \mathbb{R}\} \subset \mathbb{R}^2,$$

for $0 < M < \infty$ and a probability density function h

If $(X, Z) \sim \mathbf{U}(\mathcal{S}_M)$ then $X \sim h$

Conversely if $X \sim h$ and Z given $X = x$ is $\mathbf{U}(0, Mh(x))$ then $(X, Z) \sim \mathbf{U}(\mathcal{S}_M)$.

We sample uniformly under the envelope $cg(x)$.

Accepted points are uniform under $f(x)$.

Mixtures

Suppose that

$$f(x) = \sum_{j=1}^{J} \alpha_j f_j(x)$$

for $\alpha_j \geqslant 0$ and $\sum_{j=1}^{J} \alpha_j = 1$

If we can sample f_j then we can sample f:

1) Take random J with $\Pr(J = j) = \alpha_j$.
2) Deliver $X \sim f_J$.

machine-generated algorithms based on mixtures

1) Rectangle-tail-wedge Marsaglia, MacLaren, Bray (1964)
2) Ziggurat Marsaglia, Tsang (2000)
3) Adaptive rejection samp. Gilks, Wild (1992), Hörmann, Leydold, Derflinger (2004)

Random vectors

Random vectors can be very hard to generate.

This fact has motivated both Sequential Monte Carlo (SMC)

and Markov chain Monte Carlo (MCMC).

Sequential generation

Let $\boldsymbol{X} = (X_1, \ldots, X_d) \sim F$

we could sample X_j given $X_1, \ldots, X_{j-1}$ for $j = 1, \ldots, d$

Difficulties

1) Inversion (sequentially) . . . easier said than done. It requires lots of $F^{-1}(\cdot)$'s
2) Acceptance-rejection: c may grow exponentially with d
3) Transformations: we might not know any
4) Mixtures: geometry gets computationally problematic

Random vectors

There are good methods for the following important distributions

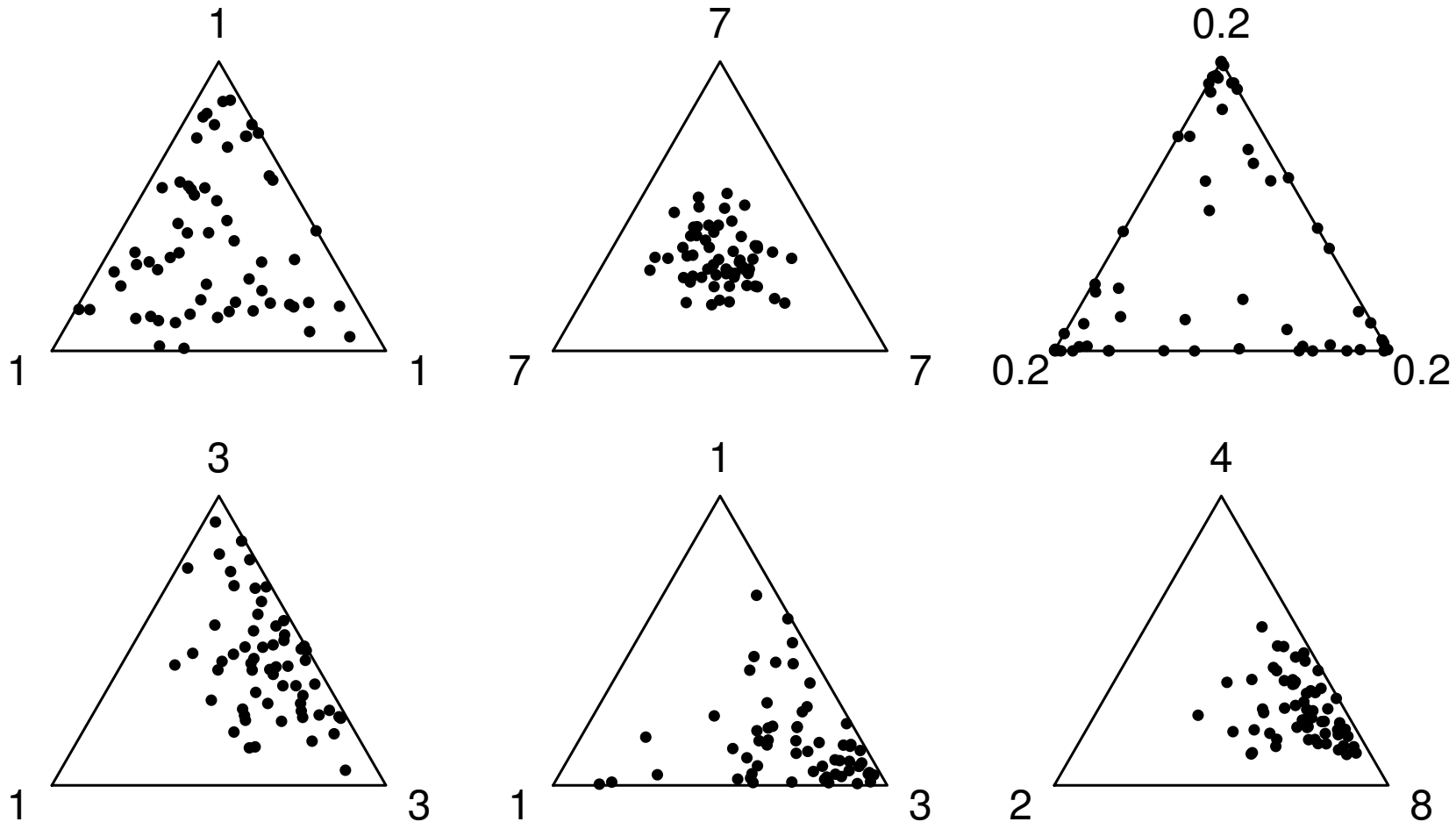
1) Multivariate normal $\mathcal{N}(\mu, \Sigma)$
2) Multivariate t: $\boldsymbol{X} = \mu + \mathcal{N}(0, \Sigma)/\sqrt{\chi^2_\nu/\nu}$
3) Multinomial[1]
4) Dirichlet[2]

[1] e.g. number of counts in bucket $j = 1, \ldots, J$ out of n independent trials, each bucket has probability π_j (think of somebody tabulating an imperfect roulette wheel)

[2] $\boldsymbol{X}$ has nonnegative components summing to 1 with density proportional to $\prod_{j=1}^{d} x_j^{\alpha_j - 1}$.

Dirichlet

Some Dirichlet samples



α_j on the corners, density $\propto \prod_{j=1}^{d} x_j^{\alpha_j - 1}$

The multivariate xxx distribution

There is no unique multivariate distribution with given margins.

E.g. Kotz, Balakrishnan & Johnson (2000) list 12 bivariate Gamma distributions.

Generalize one property $\cdots$ lose another.

Classic multivariate Poisson

$Z_j \sim \text{Poi}(\lambda_j)$ independent

$$\begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} Z_1 \\ Z_2 \\ Z_3 \\ Z_4 \\ Z_5 \\ Z_6 \end{pmatrix}$$

$\boldsymbol{X} = A\boldsymbol{Z}$ for binary matrix A and independent Poisson $\boldsymbol{Z}$.

X_i are dependent Poisson random variables.

$A_{ij} = 1$ encodes cause-effect relationships (cause $j \implies$ failure i).

We never get negative dependence for X_i and $X_{i'}$ this way.

Copula-marginal sampling

$\boldsymbol{X} = (X_1, \ldots, X_d)$ with $X_j \sim F_j$, known F_j

Glue them together with the dependence structure of $\boldsymbol{Y} \sim G$.

1) $\boldsymbol{Y} = (Y_1, \ldots, Y_d) \sim G$
2) $U_j = G_j(Y_j)$ G_j is CDF of Y_j
3) $X_j = F_j^{-1}(U_j)$
4) deliver $\boldsymbol{X} = (X_1, \ldots, X_d)$, so $X_j \sim F_j$

The vector $\boldsymbol{U} = (U_1, \ldots, U_d)$ is a 'copula', i.e., random vector with $U_j \sim \mathbf{U}(0,1)$

Gaussian copula

Also called NORTA (normal to anything) or the Nataf transformation Nataf (1962)

The Gaussian is convenient.

That doesn't mean it is correct.

The t copula has some advantages too.

(Which doesn't mean it is correct either.)

Tail independence

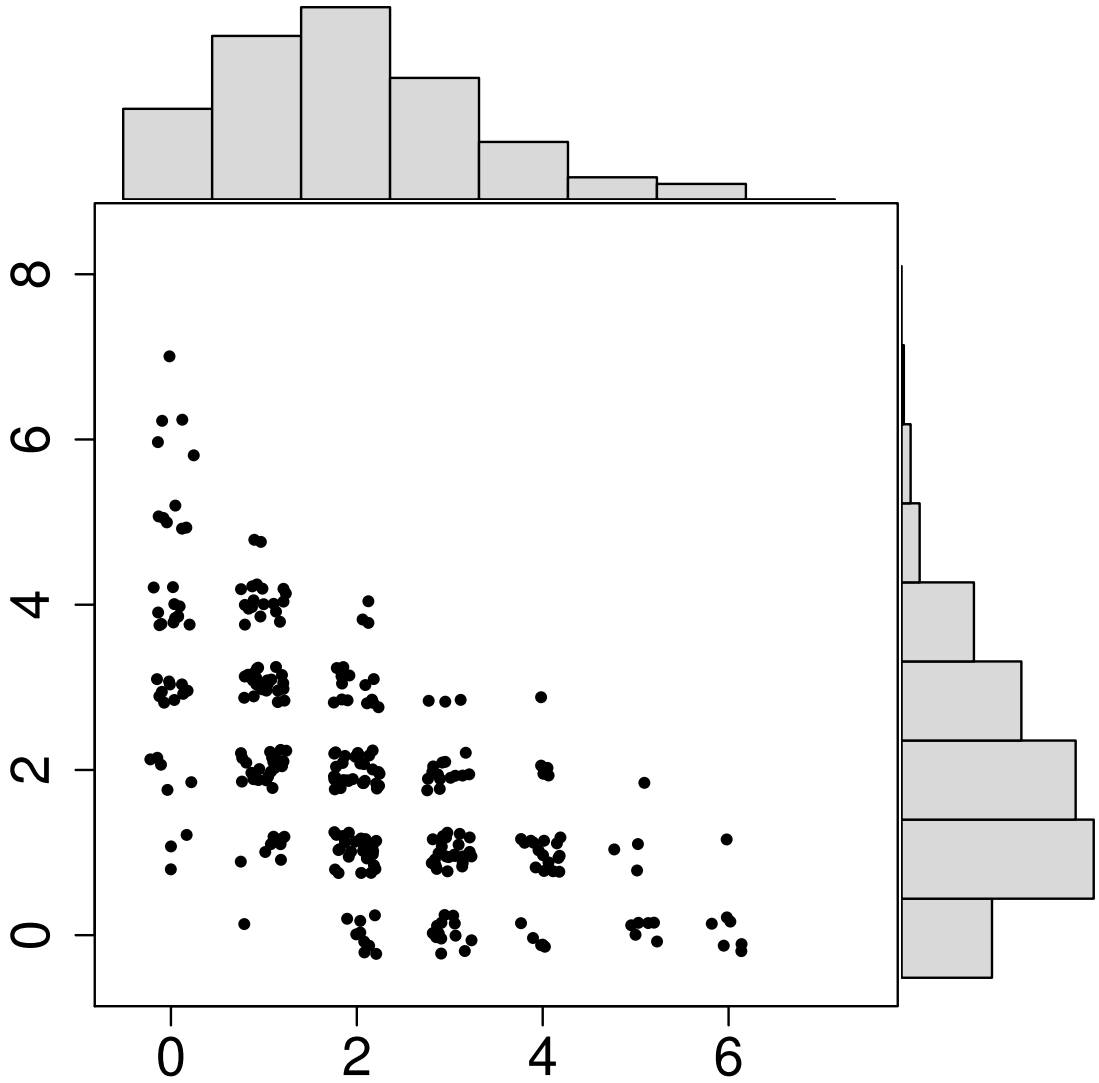
The Gaussian copula is poorly suited for finance and insurance because

$$\lim_{u\to 1^-} \Pr(X_j > F_j^{-1}(u) \mid X_k > F_k^{-1}(u)) = 0$$

When X_k gives a big loss, a big loss on X_j is unlikely (under this model)

McNeil, Frey, Embrechts (2005)

Poisson with Gaussian copula



$\boldsymbol{X} = (X_1, X_2)$

Each $X_j \sim \text{Poi}(2)$

Gaussian copula

$\mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & -0.7 \\ -0.7 & 1 \end{pmatrix}\right)$

Jittered

Random processes

Like a vector but the index has infinite cardinality

E.g. $X(t)$ is particle's position at time

$t \in [0, \infty)$, or,

$t \in \{0, 1, 2, \dots\}$.

We only get finitely points $t_1 < t_2 < \cdots < t_M$ on the trajectory

Challenges

1) Sampling consistently

2) and efficiently

3) biases, e.g.

$$\min\{X(t_1), X(t_2), \dots, X(t_M)\} \geqslant \min_{0 \leqslant t \leqslant 1} X(t)$$

Gaussian processes

For any $t_1, \ldots, t_M \in \mathcal{T} \subseteq \mathbb{R}^d$

$$\begin{pmatrix} X(t_1) \\ X(t_2) \\ \vdots \\ X(t_M) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu(t_1) \\ \mu(t_2) \\ \vdots \\ \mu(t_M) \end{pmatrix}, \begin{pmatrix} \Sigma(t_1,t_1) & \Sigma(t_1,t_2) & \cdots & \Sigma(t_1,t_M) \\ \Sigma(t_2,t_1) & \Sigma(t_2,t_2) & \cdots & \Sigma(t_2,t_M) \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma(t_M,t_1) & \Sigma(t_M,t_2) & \cdots & \Sigma(t_M,t_M) \end{pmatrix} \right)$$

Comments

- $\Sigma(\cdot,\cdot)$ has to be a positive definite function
- in principle we can choose to sample at any t_{j+1} given $X(t_1), \ldots, X(t_j)$
- in practice, computation favors special $\Sigma(\cdot,\cdot)$
- very special case: Brownian motion on $[0,\infty)$
- Brownian motion drives stochastic differential equations (Kloeden & Platen (1999))

New methods based on multilevel MC Giles et al.

Poisson processes

Random points $t_i \in \mathcal{T} \subseteq \mathbb{R}^d$ representing:

arrival times, flaws in a semiconductor, forest fire locations $\cdots$

$$N(A) = \text{Number of process points in } A \subset \mathcal{T}$$

For disjoint $A_1, \ldots, A_J \subseteq \mathcal{T}$

$$N(A_j) \sim \text{Poi}\left(\int_{A_j} \lambda(\boldsymbol{t})\,\mathrm{d}\boldsymbol{t}\right) \quad \text{independently}$$

$$\lambda(\boldsymbol{t}) \geqslant 0$$

Some sampling methods

Exponential spacings

For $\mathcal{T} = [0, \infty)$ (e.g., time) and $\lambda(t) = \lambda$ (constant) take

1) $T_0 \equiv 0$ (not part of the process)
2) For $j \geqslant 1, T_j = T_{j-1} + E_j/\lambda$, $E_j \overset{\text{iid}}{\sim} \exp(1)$
3) Until either $t_j > T$ or $j > N$

Finite integrated intensity

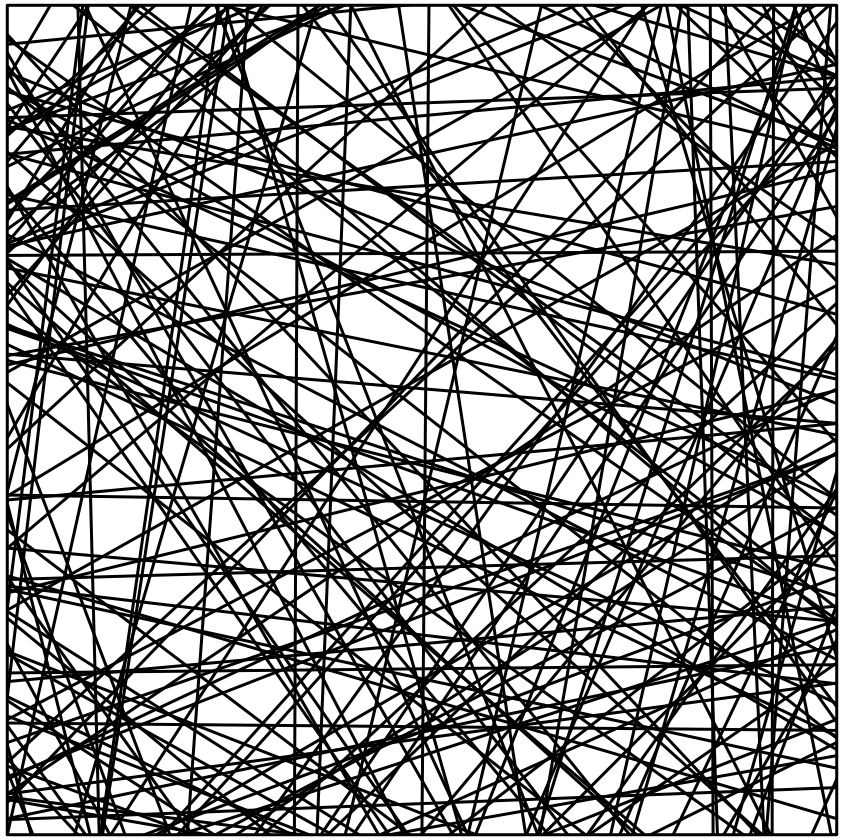
If $\int_{\mathcal{T}} \lambda(\boldsymbol{t})\,\mathrm{d}\boldsymbol{t} < \infty$:

1) $N \sim$ Poi $\left(\int_{\mathcal{T}} \lambda(\boldsymbol{t})\,\mathrm{d}\boldsymbol{t}\right)$
2) $\boldsymbol{T}_1, \ldots, \boldsymbol{T}_N \overset{\text{iid}}{\sim} f$ where $f(\cdot) \propto \lambda(\cdot)$.

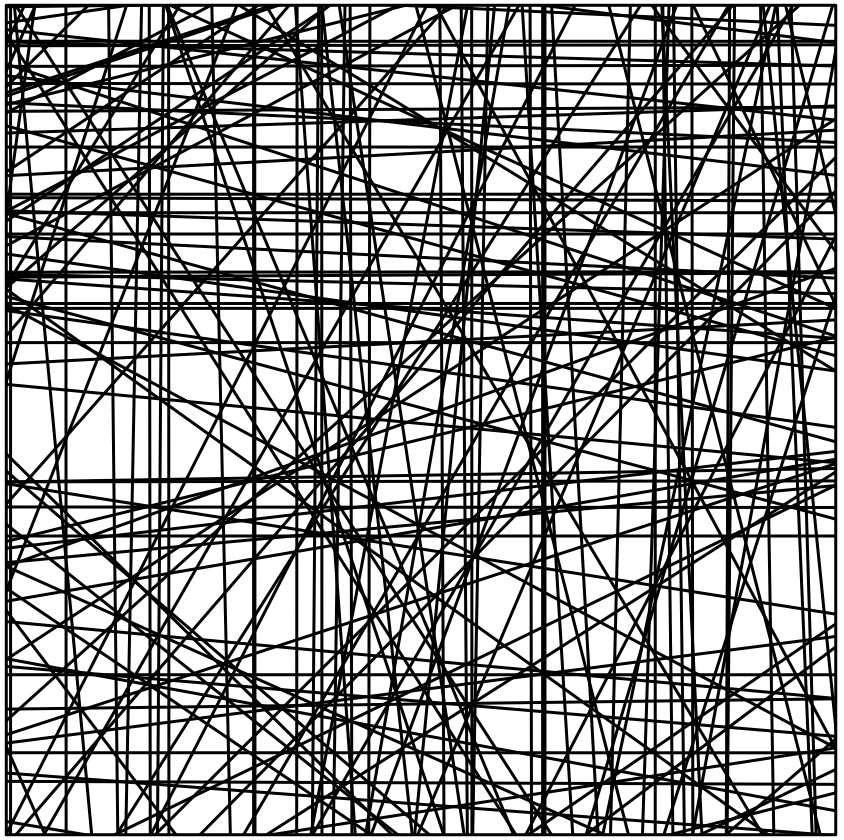
So they look like a random sample

no clustering, no avoidance

Poisson lines



Isotropic



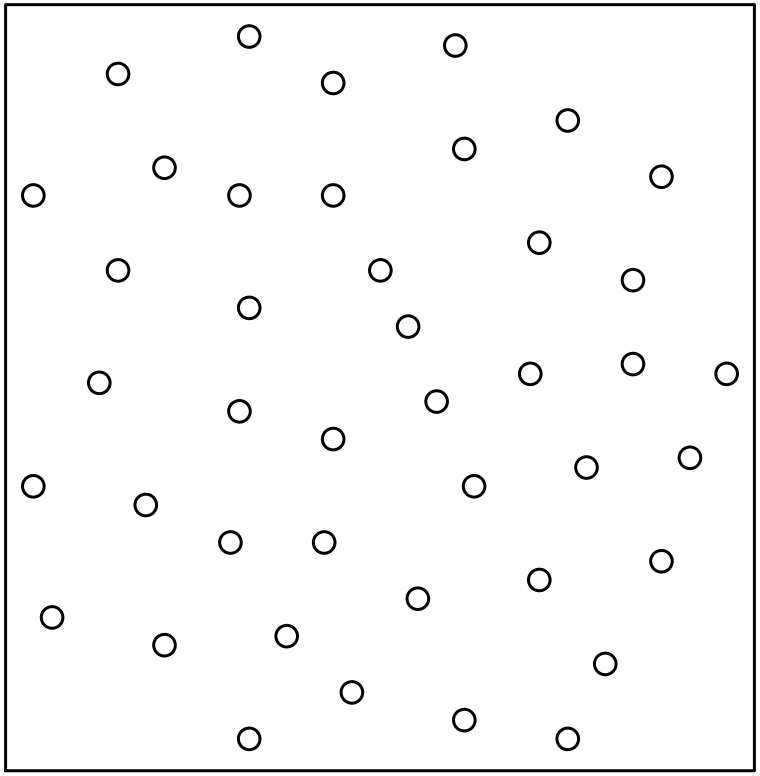
Non–isotropic

Polar coordinate definitions of the line follow Poisson process

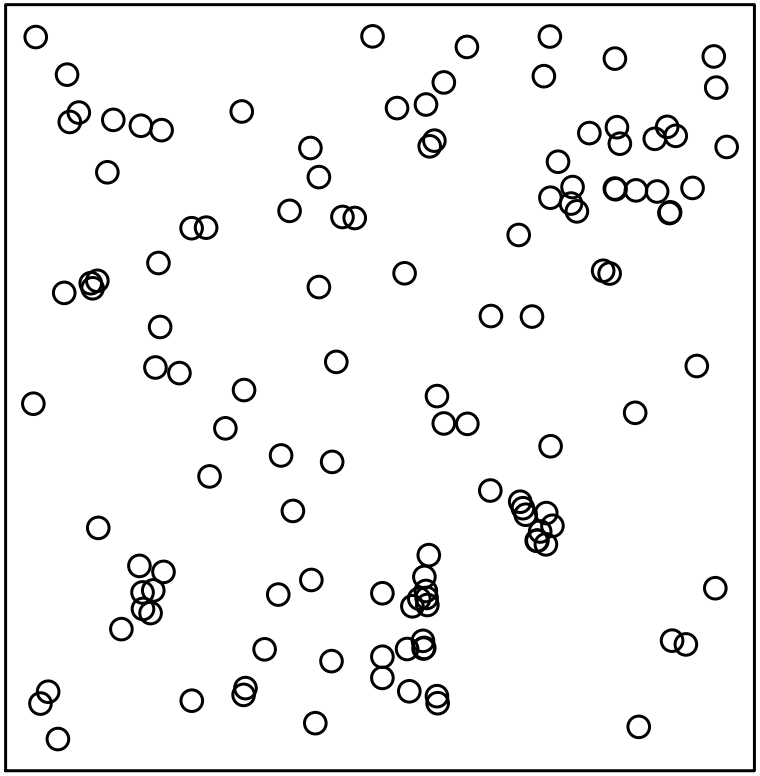
There are Poisson planes too

Non-Poisson points

Two Spatial Point Sets



Cell centers



Finnish pines

Centers of insect cells from Crick via Ripley. They avoid each other.

Locations of pine trees from Penttinen via van Lieshout (2004). They cluster.

Non-Poisson models

Clumping is easy. E.g. Cox model

1) $\lambda(\cdot) \sim$ Spatial process $\equiv$ random function

2) $\boldsymbol{T}_i \sim$ Poisson process(λ)

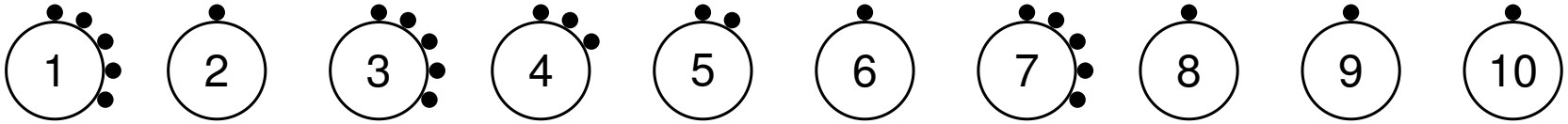
Avoidance is hard. E.g. hard shell model $\boldsymbol{T}_i \sim \mathbf{U}(\mathcal{T})$ subject to

$$\min_{1 \leqslant i < j \leqslant N(\mathcal{T})} \|\boldsymbol{T}_i - \boldsymbol{T}_j\| \geqslant \varepsilon$$

Has $O(N^2)$ constraints. May proceed via:

- Dart throwing, or,
- Markov chain Monte Carlo

Chinese restaurant process



- Start with unbounded tables and no customers
- For $k \geqslant 1$, the k^{th} customer
  - starts a new table with prob $\alpha/(\alpha + k - 1)$, or else,
  - joins existing table with prob $\propto$ # people there

Notes

CRP used in statistical machine learning

See Jordan (2005)

Counterpart: Indian buffet process

Still no known vegemite stochastic process (VSP)

Importance sampling

Consider $\mu = \mathbb{E}(f(\boldsymbol{X})) = \int_{\mathbb{R}^d} f(\boldsymbol{x}) p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$

where $f \approx 0$ outside of A and $\Pr(\boldsymbol{X} \in A)$ is tiny

Examples: rare events, small probabilities, spiky functions,

floods, power outages, way out of money options,

probability of network failure, etc.

The idea

Arrange for $\boldsymbol{X} \in A$ to happen more often.

Then adjust for bias.

Importance sampling ctd.

Probability density $q(\boldsymbol{x}) > 0$ whenever $f(\boldsymbol{x})p(\boldsymbol{x}) \neq 0$

$$\mu = \int f(\boldsymbol{x})p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int \frac{f(\boldsymbol{x})p(\boldsymbol{x})}{q(\boldsymbol{x})} q(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$$

$$\hat{\mu}_q = \frac{1}{n}\sum_{i=1}^{n} \frac{f(\boldsymbol{X}_i)p(\boldsymbol{X}_i)}{q(\boldsymbol{X}_i)}, \quad \boldsymbol{X}_i \overset{\text{iid}}{\sim} q$$

Variance

$$\mathrm{Var}(\hat{\mu}_q) = \frac{\sigma_q^2}{n} \quad \text{where} \quad \sigma_q^2 = \int \frac{(f(\boldsymbol{x})p(\boldsymbol{x}))^2}{q(\boldsymbol{x})}\,\mathrm{d}\boldsymbol{x} - \mu^2$$

small $q(\boldsymbol{x})$ are problematic

Variance again

$$\sigma_q^2 = \int \frac{(f(\boldsymbol{x})p(\boldsymbol{x}))^2}{q(\boldsymbol{x})}\,\mathrm{d}\boldsymbol{x} - \mu^2 = \int \frac{(f(\boldsymbol{x})p(\boldsymbol{x}) - \mu q(\boldsymbol{x}))^2}{q(\boldsymbol{x})}\,\mathrm{d}\boldsymbol{x}$$

Consequences

1) If $q(\boldsymbol{x}) \propto f(\boldsymbol{x})p(\boldsymbol{x})$ then $\sigma_q^2 = 0$
2) Best is $q(\boldsymbol{x}) \propto |f(\boldsymbol{x})p(\boldsymbol{x})|$
3) For safety, take $q(\boldsymbol{x})$ 'heavier tailed' than $p(\boldsymbol{x})$
   E.g. $p = \mathcal{N}(0,1)$ with $q = t_5$

Finding a good importance sampler is an art and a science.

Success is not assured.

A poor choice can give $\sigma_q^2 = \infty$ even when $\sigma^2 < \infty$

For adaptive importance sampling, see Evans & Swartz, Rubinstein

Main problems with MC

1) We often cannot sample $\boldsymbol{X} \overset{\text{iid}}{\sim} p$ for desired p
2) Sometimes $\sqrt{n}$ rate is too slow

Solutions (partial credit only)

1) **MCMC**: greatly expands the range of problems
   evolves out of acceptance-rejection
   uses dependent values
2) **QMC, RQMC:** improves the convergence rate
   evolves out of stratification
   exploits smoothness

Combinations

Latest MCMC ∩ QMC in Stanford thesis of S. Chen (2011)

Deeper theory than before

Good performance for GARCH and stochastic volatility models