# Reducing variance with averaging kernels: General theory and an application to quasirandom simulation of Markov chains

James Propp (UMass Lowell and UC Berkeley)

February 14, 2012

Slides for this talk are on-line at
http://jamespropp.org/mcqmc12.pdf

# Part I: Kernel smoothing

# The punchline

If you want to estimate the asymptotic mean value

$$\overline{X} = \lim_{n\to\infty} (X_1 + X_2 + \cdots + X_n)/n$$

of a deterministic or random numerical sequence $X_1, X_2, \ldots$, and there are patterns (especially periodicities) in the sequence, then there may be better way to estimate $\overline{X}$ than to take the unweighted average

$$\hat{X}_N = (X_1 + X_2 + \cdots + X_N)/N$$

for a single large $N$; weighted ("smoothed") averages such as

$$\widehat{X}_N = ((1)(N)X_1 + (2)(N-1)X_2 + \cdots + (N)(1)X_N) \,/\, \frac{N(N+1)(N+2)}{6}$$

often do better.

# A baby example from QMC integration

Let $f(x) = x(1-x)$ and $I = \int_0^1 f(x)dx = 1/6$, and let $X_n = f(x_n)$ where

$$(x_n)_1^\infty = (0, \frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \frac{1}{8}, \frac{5}{8}, \dots)$$

(the van der Corput sequence), so that $\overline{X} = I$.

Then for $N \approx 10^6$, it appears that $\hat{X}_N = I \pm O(1/N^{1.3})$ and $\widehat{X}_N = I \pm O(1/N^{1.7})$.

# A strange example ...

Suppose $U, V$ are independent Uniform($[0,1]$)'s and

$$X_n = \lfloor nU + V \rfloor - \lfloor (n-1)U + V \rfloor \in \{0, 1\},$$

so that $(X_n)_1^\infty$ is a sequence of 0's and 1's with random density $U$ and random phase $V$.

$X_n = \phi_U(nU + V)$ where $\phi_U(t)$ is 1 if the fractional part of $t$ lies in $(0, U)$ and 0 otherwise, so that $\int_0^1 \phi_U(t)\, dt = U$.

$$\begin{aligned} \overline{X} &= \lim_{n\to\infty} (X_1 + X_2 + \cdots + X_n)/n \\ &= \lim_{n\to\infty} (\lfloor nU + V \rfloor - \lfloor V \rfloor)/n \text{ (by telescoping)} \\ &= U. \end{aligned}$$

Then $\hat{X}_N = U \pm O(1/N)$ and $\widehat{X}_N = U \pm O(1/N^{3/2})$ (3/2 is exact).

# ... but an important one

This example may seem arcane, but it's actually natural; $(X_n)_1^\infty$ is an example of a **Sturmian sequence**, and we'll see that Sturmian bit-streams are good for driving Markov chains.

A Sturmian bit-sequence of density $p$ has the property that the sum of any $k$ consecutive bits is either the floor of $kp$ or the ceiling of $kp$. This is as low-discrepancy as an integer-valued random variable with expected value $kp$ can be!

The fact that we can empirically reconstruct (i.e., estimate) $\overline{X}$ with high accuracy from $X_1, X_2, \ldots$ translates into analogous reconstruction properties for Markov chains driven by $X_1, X_2, \ldots$ (as we'll see in parts II and III).

# Kernel Smoothing Kills Sinusoids

Claim: Suppose $f(t)$ is a constant plus a finite sum of sinusoids (possibly with irrational and incommensurable periods), and let $X_n = f(n)$, so that $\overline{X} = \lim_{n\to\infty}(f(1)+f(2)+\cdots+f(n))/n$. Then

$$\hat{X}_N = \overline{X} \pm O(1/N)$$

and

$$\widehat{X}_N = \overline{X} \pm O(1/N^2).$$

Proof idea: One can evaluate the unsmoothed and smoothed average for each sinusoid individually.

Kernel smoothing the data is tantamount to squaring the length of our time-series, and it's a linear procedure; we don't have to look at the data and try to guess what the periods are.

# Part II: Rotor-routing

# CUD vs. rotor-routing

Owen et al. ask: When can Completely Uniformly Distributed streams be used instead of IID streams for simulation?

P. et al. ask: When can periodic/Sturmian streams be used instead of IID streams?

# Gambler's ruin

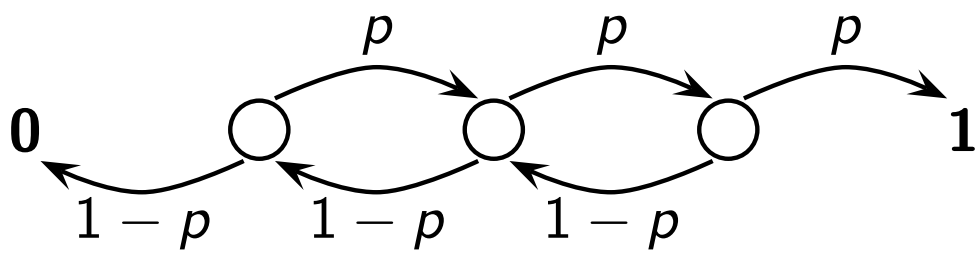
Consider the Markov chain with state-space $S = \{0, 1, 2, 3, 4\}$, where state 1 is the initial state, states 1, 2, and 3 are transient, and states 0 and 4 are absorbing:

$$p_{1,2} = p_{2,3} = p_{3,4} = p,$$

$$p_{1,0} = p_{2,1} = p_{3,2} = 1 - p,$$

$$p_{0,0} = p_{4,4} = 1.$$

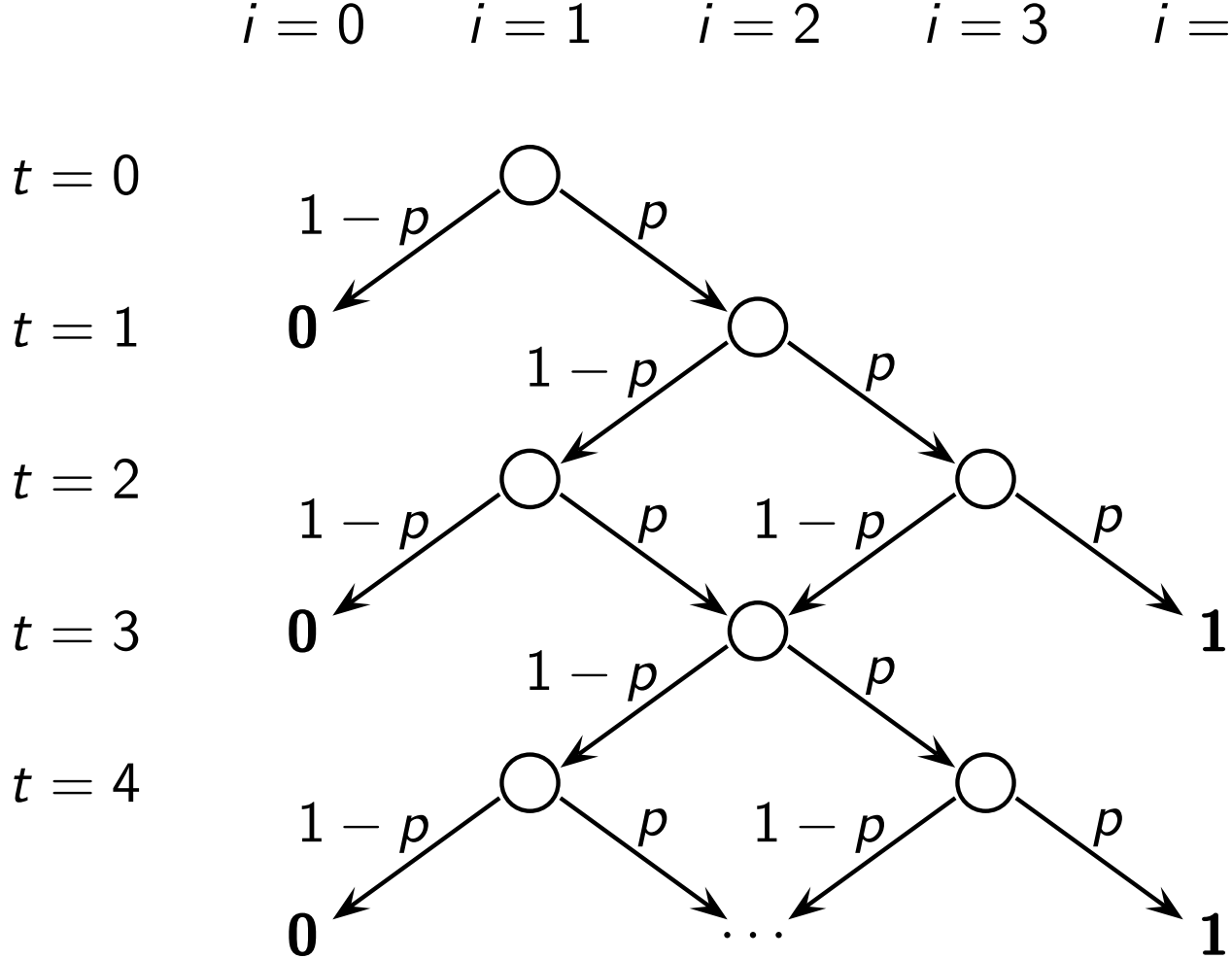
$i = 0$ $i = 1$ $i = 2$ $i = 3$ $i = 4$



# Lifting to space-time

We replace the chain by its "space-time lift" with "space-time states" $(i, t)$ in $S \times \mathbf{N}$ for $i \in S$ and $t \in \mathbf{N} = \{0, 1, 2, \dots\}$;

$$\begin{aligned} p_{(i,t),(i+1,t+1)} &= p, \\ p_{(i,t),(i-1,t+1)} &= 1 - p. \end{aligned}$$


$i = 0$
$i = 1$
$i = 2$
$i = 3$
$i = 4$
$t = 0$
$t = 1$
$t = 2$
$t = 3$
$t = 4$
$1 - p$
$p$
$\mathbf{0}$
$\mathbf{1}$
$\ldots$

# Escape probability via simulation

Our goal is to estimate empirically the escape probability $p_{\mathrm{esc}}$ (the probability of reaching 4 before reaching 0).

Of course in this case there's an exact formula for $p_{\mathrm{esc}}$; this is a pedagogical example and a proof-of-principle, not a genuine application!

We commence a new trial (starting a new particle at the top) each time the lifted chain reaches a state $(0, t)$ or a state $(4, t)$, outputting a $\mathbf{0}$ or a $\mathbf{1}$ respectively. When we've recorded $N$ output-bits, we compute their sum, $K$; $K/N$ is our empirical estimate of $p_{\mathrm{esc}}$.

# Ordinary Monte Carlo

The usual thing we do is drive the process using Bernoulli (IID) bit-streams of density $p$, one for each space-time state $(i, t)$ (with $i$ transient).

(Actually, what we *really* do is use a single bit-stream, but that's equivalent to the procedure I described, since the streams are all IID and independent of one another.)

Then $K/N = p_{\mathrm{esc}} \pm O(1/N^{1/2})$.

# Negatively Correlated Monte Carlo

But we get a better empirical estimate of $p_{\mathrm{esc}}$ if we use a **Sturmian** bit-stream of density $p$ for each space-time state $(i, t)$ with $i$ transient; the correlations don't hurt us, and the low discrepancy helps us.

"Tricky but true": Each output-bit is Bernoulli($p_{\mathrm{esc}}$).

There are dependencies between one trial and the next, so these Bernoulli($p_{\mathrm{esc}}$) output-bits are correlated, but these correlations are of the negative (variance-reducing) kind.

We get $K/N = p_{\mathrm{esc}} \pm O((\log N)/N)$.

# "What about states with more than two successors?"

By generalizing the class of Sturmian sequences we can design infinite sequences (or "streams") in any finite alphabet with prescribed frequencies summing to 1, such that for any $k$, the number of occurrences of any symbol in any length-$k$ subword of the infinite word takes on only finitely many values.

We can associate such a driving stream with each state $(i, t)$ of the lifted Markov chain, where the different symbols correspond to the allowed transitions from that state.

# From space-time rotors to space-routers

We do even better if we use a single driving stream for each space-state $i$, instead of each space-time state $(i, t)$.

(Individual trials are no longer Markovian — they have long-term self-correlations. But the escape probability for this non-Markov process can be shown to be the same as for the original Markov process.)

We get $K/N = p_{\mathrm{esc}} \pm O(1/N)$ (the log factor goes away).

# Rotor-routing for stationary probabilities

So far we've looked at *escape probabilities*.

Similarly, one can estimate the *stationary probability* of a particular state $i$ as $K/N$, where $K$ is the number of times state $i$ occurs during a run of length $N$.

For ordinary Monte Carlo, $K/N = \pi(i) \pm O(1/N^{1/2})$ (where $\pi(\cdot)$ is the stationary measure).

For rotor-routing, $K/N = \pi(i) \pm O(1/N)$.

The implicit constant can be large if the chain has many states. A better result from Holroyd-P. says that rotor-routing can be used to estimate probability ratios $\pi(i)/\pi(j)$ to within $O(1/N)$.

# Connections to previous work

The Holroyd-P. results apply to *all* Markov chains with finite state-space and to *some* (sufficiently recurrent) Markov chains with infinite state-space.

The Holroyd-P. results apply with *deterministic* rotor-settings. The results stated above for space-rotors (with *randomized* rotor-settings) are all direct consequences.

The results stated above for random space-time rotors require new arguments (not yet published).

# Can we do better?

If we are trying to assess $p_{\mathrm{esc}}$ (or some other probability) from a sequence of $N$ random (but not necessarily independent) bits, each with expected value $p_{\mathrm{esc}}$, the best we can hope to do (for geometry-of-numbers reasons) is estimate $p_{\mathrm{esc}}$ to within $O(1/N^2)$.

Monte Carlo has error $O(1/N^{1/2})$.

Rotor-router has error $O(1/N^1)$.

Can we get a better exponent?

# Part III: Kernel smoothing and rotor-routing

# Relative stationary probabilities

If one performs rotor-routing and records the order of the respective visits to states $i$ and $j$ (with $\pi(i)$ and $\pi(j)$ bounded away from 0) and performs smoothing on the resulting sequence in order to estimate $\pi(i)/\pi(j)$, one typically sees error falling off like $O(1/N^{1.5})$.

# Escape probabilities

Likewise, smoothing appears to give consistent improvement in estimates of $p_{\mathrm{esc}}$.

Typical performance is around $O(1/N^{1.5})$.

In fact, smoothed rotor-routing works well even when the state-space is infinite (so that the linear algebra method of computing $p_{\mathrm{esc}}$ is not applicable), e.g., random walk on $\mathbf{Z}^2$, where error for rotor-routing has been proved to fall like $O((\log N)/N)$ and error for smoothed rotor-routing seems to fall like $O(1/N^{1.5})$.

# I want more information (just 5 more slides)

I'd appreciate leads to relevant literature I may not know about.

I'd also welcome suggestions for problems to tackle with these methods.

# If you want more information (just 4 more slides)

See the slides from my longer talk on this subject: "Rotor-routing, smoothing kernels, and reduction of variance: Breaking the $O(1/n)$ barrier".

If you prefer a talk less focused on rotor-routing and more focused on kernel smoothing, and you have Mathematica, see the slides from my talk "How well can you see the slope of a digital line? (and other applications of averaging kernels)". (If you don't have Mathematica, you can download the Mathematica player and look at the Mathematica player version of the slides.)

For much more about rotor-routing, see my article with Ander Holroyd on rotor-routing: "Rotor Walks and Markov Chains", `arXiv:0904.4507`.

# Summary (just 3 more slides)

For determining escape probabilities, mean hitting times, and relative stationary probabilities of Markov chains, the error of rotor-router estimates (using space-rotors) is provably $O(1/N)$ for all finite Markov chains and all sufficiently recurrent infinite (discrete) Markov chains (sometimes with an extra factor of $\log N$).

If one applies smoothing, the error can (apparently) be brought as low as $1/N^{1.5}$ (this has only been proved in very special cases, but experimental evidence supports the general claim).

# Summary (just 2 more slides)

These problems are in in regimes where exact methods (for finite state spaces) or iterative methods (for infinite state spaces) beat rotor-routing, even with the extra accuracy provided by smoothing.

Most (though not all) of the successes of the method are in a toy domain where simulation runs are long compared to the "effective size" of the state space of the Markov chain, so that most of the states (measured by mass, not cardinality) get visited many times.

This is far from the domain of ordinary MCQMC — but the fact that such a simple trick as rotor-routing works as well as it does suggests that *if* the fundamental idea can be applied in other settings, simulation error might be reduced.

# Summary (last slide)

But in any case, if you're using MCQMC and your runs exhibit periodicities, you should try smoothing them with an averaging kernel!