

Numerical simulation of random curves - lecture 4

Tom Kennedy

Department of Mathematics, University of Arizona

Supported by NSF grant DMS-0501168

<http://www.math.arizona.edu/~tgk>

Implementing the Pivot Algorithm

Two bottlenecks:

- Check proposed pivot for intersections
- Carry out the pivot if accepted

Checking:

vague idea: $d = \|\omega(i) - \omega(j)\|_1$

$d \neq 0 \Rightarrow \omega(i) \neq \omega(j)$

$\omega(i') \neq \omega(j')$, if $|i - i'| + |j - j'| < d$

l is “time” at which the pivot is done.

Inductive assumption:

We have times i, j ($j < l < i$) such that

$\omega(i') \neq \omega(j')$ if $j < j' < l < i' < i$

Try to decrease j or increase i so that above still holds.

Implementation - continued

$$\omega(i') \neq \omega(j') \quad \text{if } j < j' < l < i' < i$$

Procedure for **increasing** i :

$$m_i = \text{dist}(\omega(i), \{\omega(k) : j < k < l\})$$

$m_i = 0 \Rightarrow$ self-intersection. $m_i > 0 \Rightarrow$ can increase i by m_i .

Try to compute a lower bound b_i on m_i

Loop on j' from $l - 1$ down to j . Set $b_i = N$.

During the loop,

$$b_i \leq \text{dist}(\omega(i), \{\omega(k) : j' < k < l\})$$

$$d = \|\omega(i) - \omega(j')\|.$$

Pick $s < d$.

$$\text{dist}(\omega(i), \{\omega(k) : j' - s \leq k \leq j'\}) \geq d - s.$$

So $b_i \rightarrow \min\{b_i, d - s\}$, $j' \rightarrow j' - s$

How to choose s ?

Simplest choice: $s = d/2$.

Faster: If $d < b_i$ take $s = d/2$.

If $d \geq b_i$, take $s = d - b_i$.

Better choices ???

Implementation - continued

Carrying out pivots

Pivot $\omega \rightarrow \bar{\omega}$ by

$$\bar{\omega}(j) = \begin{cases} \omega(j), & \text{for } j \leq l \\ g[\omega(j) - \omega(l)] + \omega(l), & \text{for } j \geq l \end{cases}$$

l is pivot time

g is a lattice symmetry which fixes 0

Could just keep list of the l 's and g 's for the accepted pivots.

Need to be able to find current $\omega(i)$ fast.

Suppose we have accepted n pivots and pivot times are

$$l_1 < l_2 < \cdots < l_n.$$

Let ω be walk after these pivots, ω' the walk before these pivots.

Segments from l_i to l_{i+1} are rigid.

Data structure

- the “old” walk ω' .
- n = number of pivots accepted, but not carried out yet.
- pivot times: $l_1 < l_2 < \dots < l_n$
- lattice symmetries: g_1, g_2, \dots, g_n
- lattice sites: x_1, x_2, \dots, x_n

$$\omega(j) = g_i \omega'(j) + x_i, \quad \text{for } l_i \leq j \leq l_{i+1}$$

Choose N_{pivot} large, but $\ll N$.

As pivots are accepted, update above.

When $n = N_{pivot}$, carry out all the pivots

Theory : N_{pivot} should go like \sqrt{N}

Practice : $N_{pivot} = \sqrt{N/40}$.

Assorted Tricks

1. Computing random variables.

Some RV's are cheap : end to end dist.

Some are expensive: distance from walk to a fixed point.

Use fact that walk is nearest neighbor.

2. Picking the pivot time

If the RV is “supported” near the start, it pays to make the probability distribution for picking the pivot time non-uniform.

4 *Practical Issues*

Thermalization or burn-in

For SAW we start with a walk that is a line.

For Ising we start with random configuration (infinite T).

These are atypical. More precisely, for most RV's we care about the value of the RV is way out in the tail of the distribution of the RV.

Need to run the chain until it is in equilibrium.

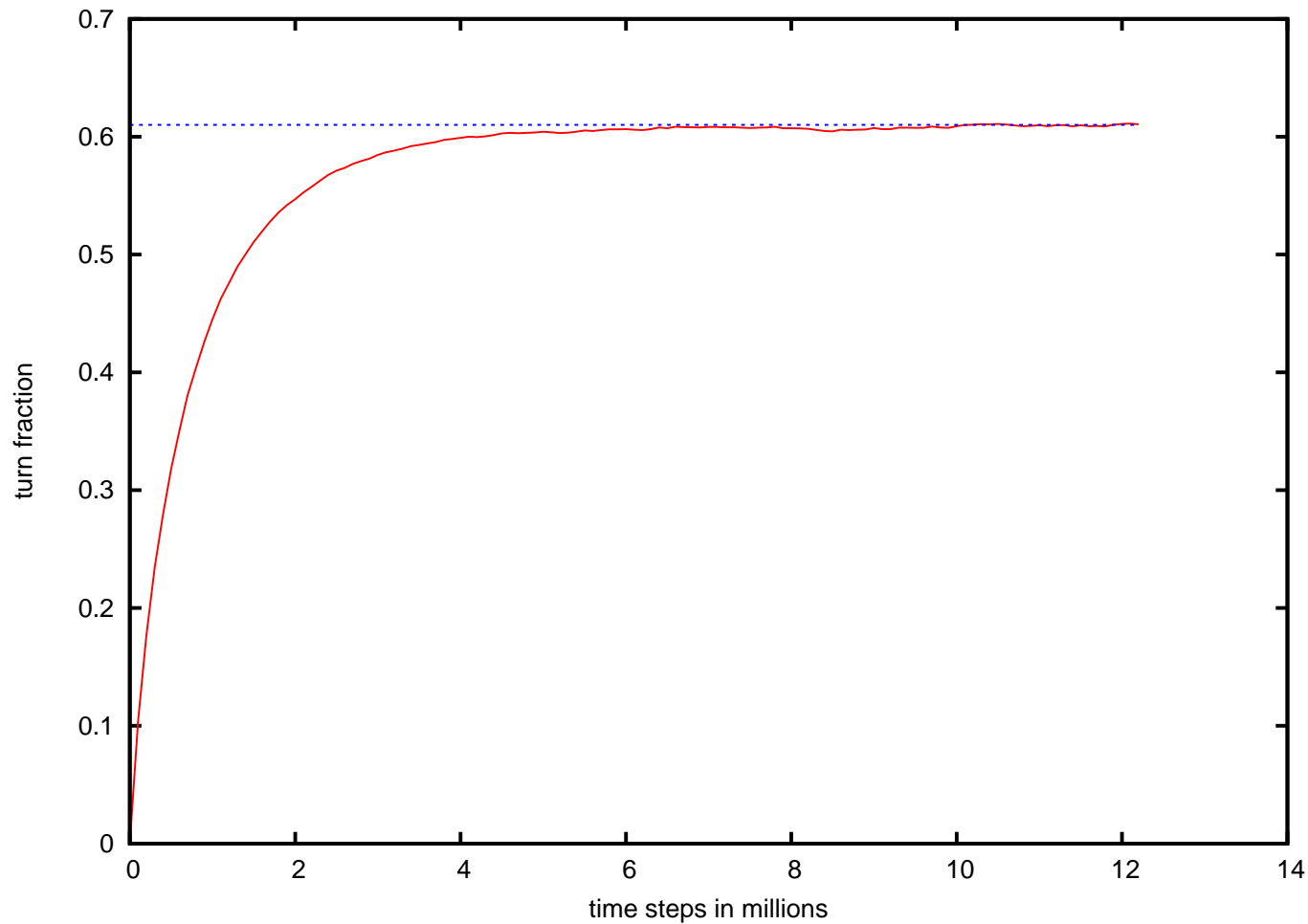
Crude, practical approach: plot random variable as function of time.

Of course, you can save walk or spin config after your run to use as initial state for your next run.

Example of thermalization - SAW

100,000 step SAW. Start with line.

RV is fraction of steps that turn rather than go straight.



Convergence

$$\frac{1}{n} \sum_{k=1}^n f(X_k) \rightarrow \sum_x f(x) \pi_x$$

But $f(X_k)$ are not independent.

How correlated - auto correlation time

Should do careful time series analysis.

Quick and dirty approach - batched means

Systematic errors

Convergence of previous slide was the convergence of the Markov chain.

Typically there are other limits to be taken, possibly even a double limit.

SAW: send number of steps to ∞ , then lattice spacing to 0.

Hard to estimate. Example: exponent ν for 3d SAW.

Madras, Sokal : '88 : $\nu = 0.592 \pm 0.003$

Li, Madras, Sokal '95: $\nu = 0.5877 \pm 0.0006$

Random number generation

Never assume a random number generator is good.

Dan Goodin in San Francisco, 21 May 2008 18:47

It's been more than a week since Debian patched a massive security hole in the library the operating system uses to create cryptographic keys for securing email, websites and administrative servers. Now the hard work begins, as legions of admins are saddled with the odious task of regenerating keys too numerous for anyone to estimate.

The flaw in Debian's random number generator means that OpenSSL keys generated over the past 20 months are so predictable that an attacker can correctly guess them in a matter of hours. Not exactly a comforting thought when considering the keys in many cases are the only thing guarding an organization's most precious assets. Obtain the key and you gain instant access to trusted administrative accounts and the ability to spoof or spy on sensitive email and web servers.

Code

Time spent thinking about the code before you write any code is time very well spent.

Document the code so that in five years you can look at it and figure out what is going on.

Modularize (object-oriented programming)

C++ lets you create data structures that help do this.

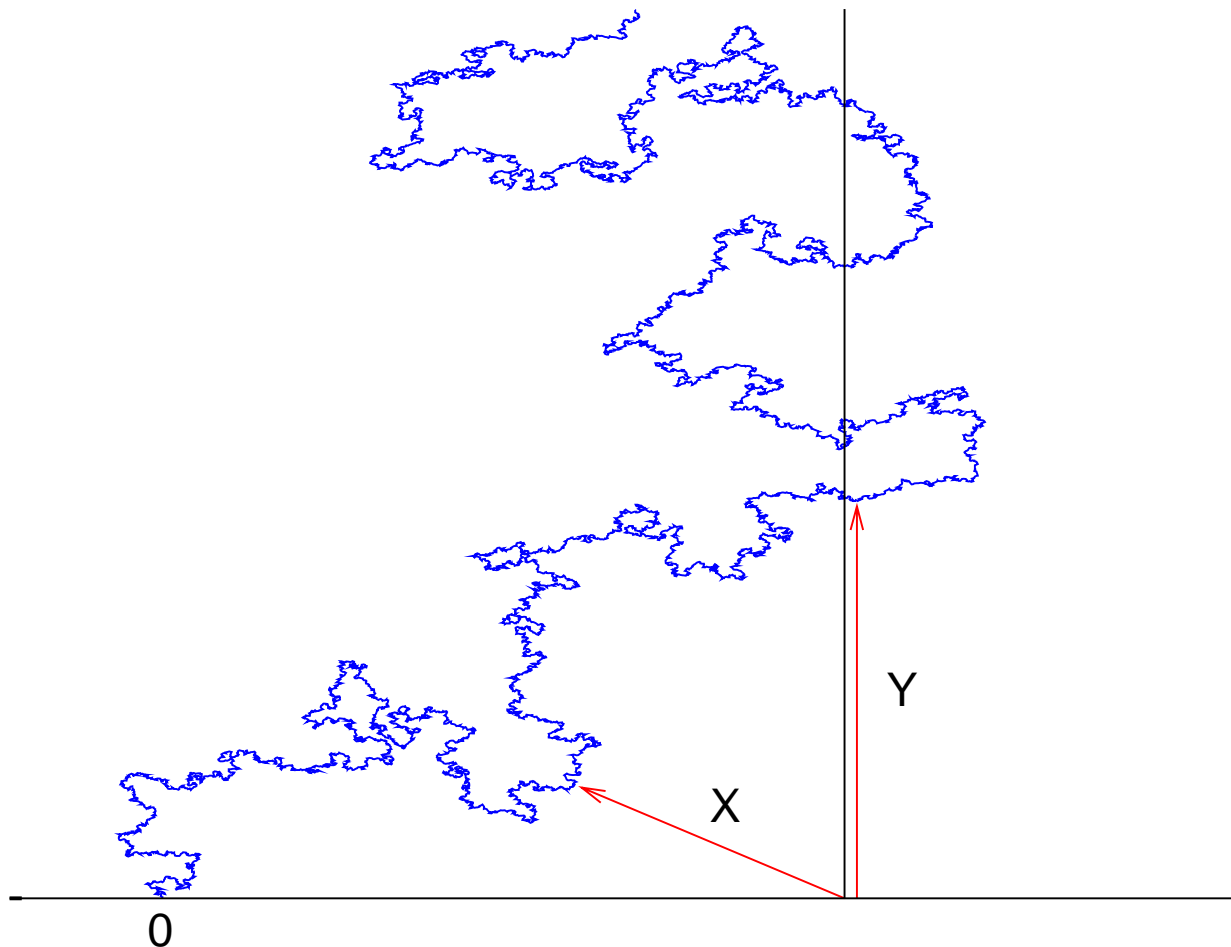
Example of SAW. Classes: [point](#), [walk](#)

Details of lattice only appear at low level.

$$SAW = SLE_{8/3}$$

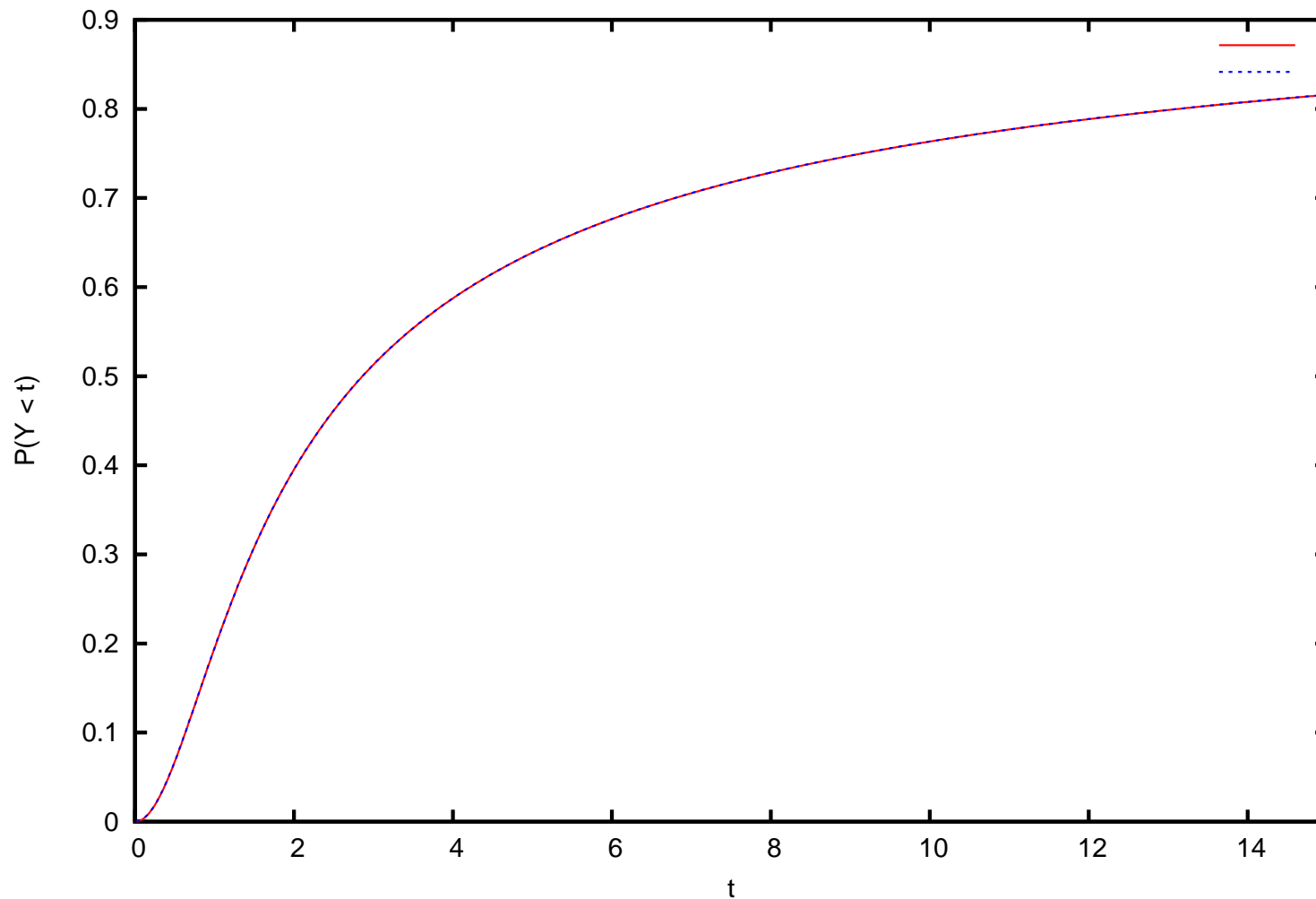
Lawler, Schramm and Werner conjectured that the scaling limit of the SAW is $SLE_{8/3}$.

Monte Carlo simulations of the SAW support their equivalence as unparameterized curves.



Distribution of Y

Graph is of the cumulative distribution function, $P(Y \leq t)$ as a function of t .



Natural parameterization of SAW

SAW has a natural parameterization. Let $W(n)$ be infinite SAW on unit lattice. Define

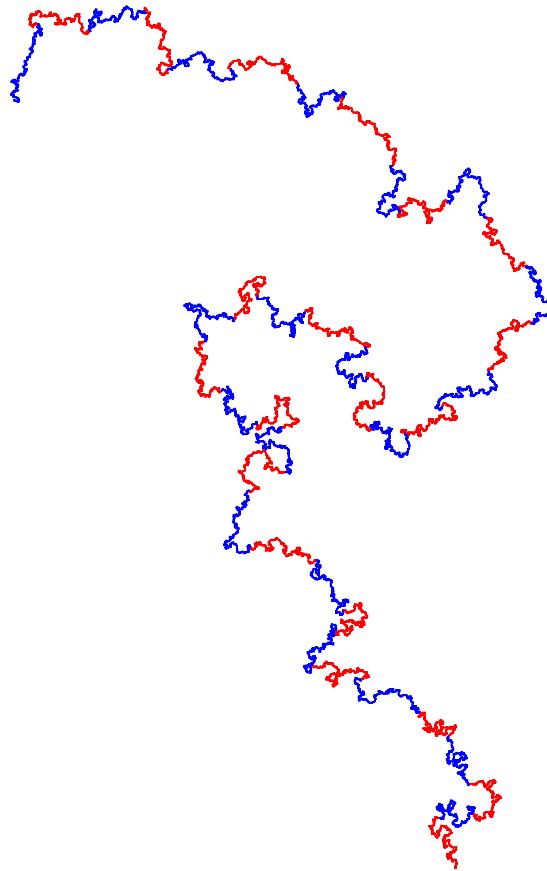
$$\omega(t) = \lim_{n \rightarrow \infty} n^{-\nu} W(nt), \quad E \omega(t)^2 = c' t^{2\nu}$$

Natural parameterization of SAW

SAW has a natural parameterization. Let $W(n)$ be infinite SAW on unit lattice. Define

$$\omega(t) = \lim_{n \rightarrow \infty} n^{-\nu} W(nt), \quad E \omega(t)^2 = c' t^{2\nu}$$

SAW: 100,000 steps, 40 segments of 2,500 steps



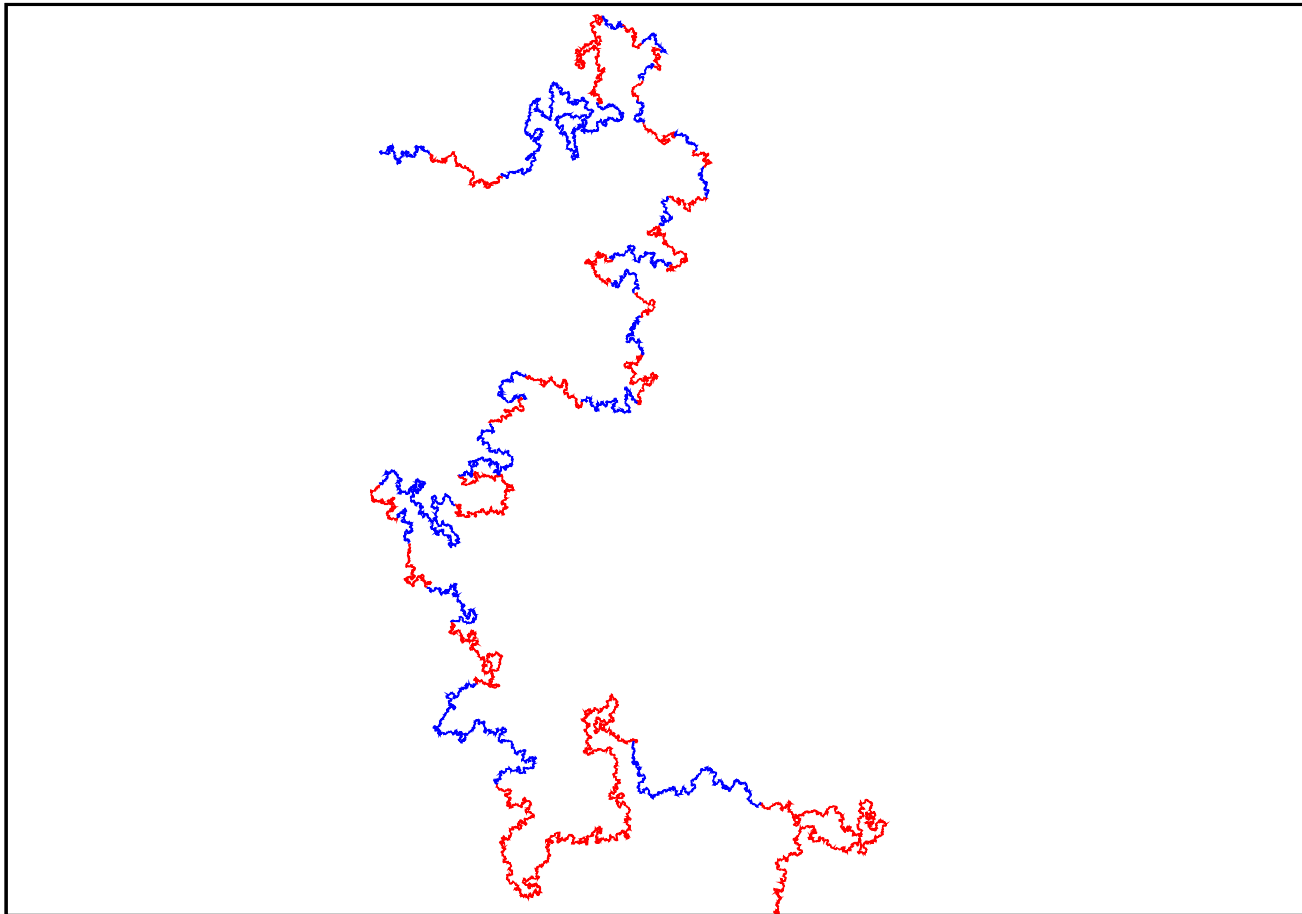
Capacity parameterization of SLE

SLE is usually parameterized by half-plane capacity.

If we divide it into segments of equal change in capacity, we get

Capacity parameterization of SLE

SLE is usually parameterized by half-plane capacity.
If we divide it into segments of equal change in capacity, we get



Parameterizations

How are SAW and SLE parameterized?

- SLE is usually parameterized so that

$$\mathit{hcap}(\gamma[0, t]) = 2t, \quad E \gamma(t)^2 = ct$$

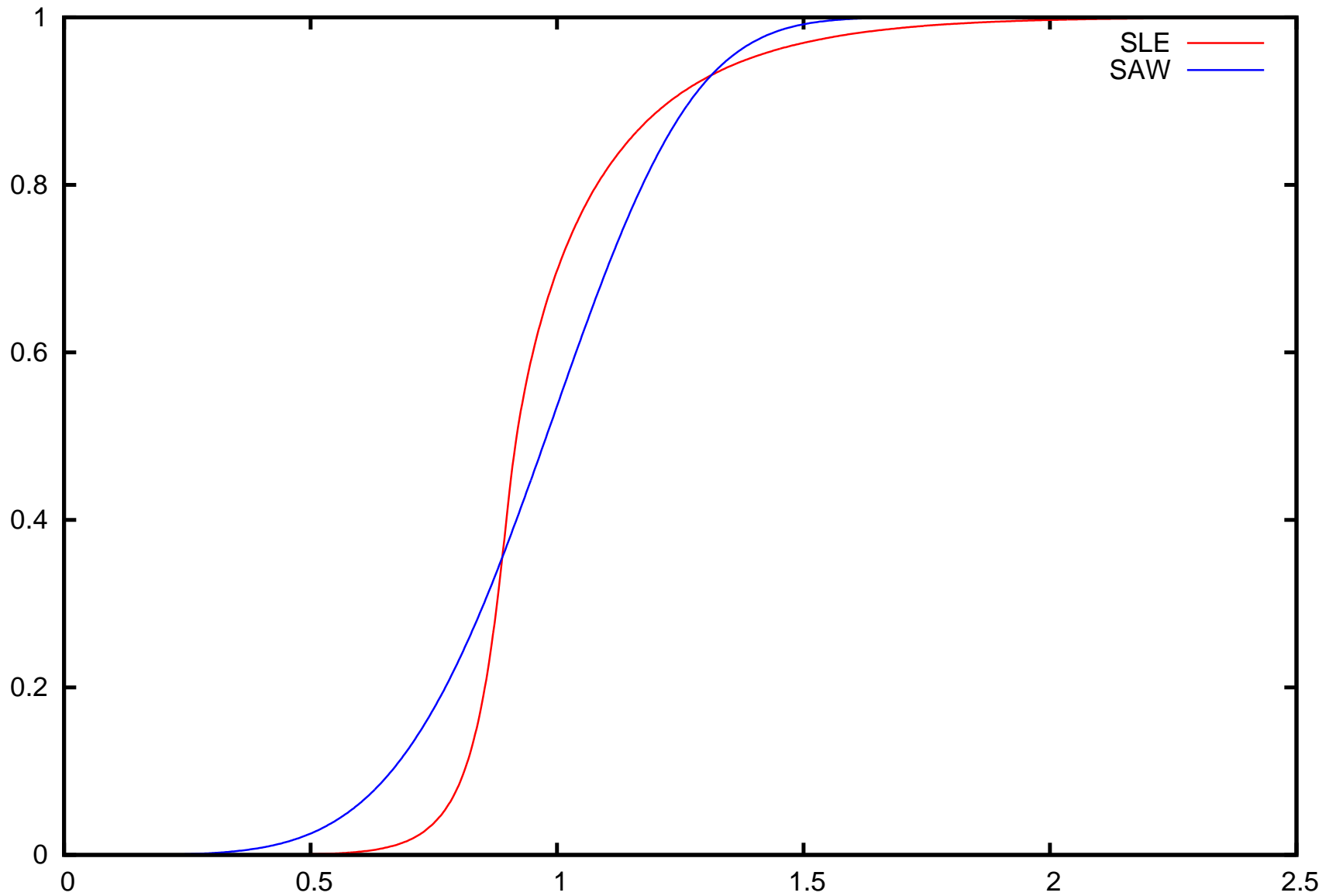
hcap is half-plane capacity. (Coef of $1/z$ in Laurent expansion of $g_t(z)$ about ∞ .)

- SAW has a natural parameterization. Let $W(n)$ be infinite SAW on unit lattice. Define

$$\omega(t) = \lim_{n \rightarrow \infty} n^{-\nu} W(nt), \quad E \omega(t)^2 = c't^{2\nu}$$

Compare $\gamma(1)$ and $\omega(1)$, rescaled so $|\gamma(1)|$ and $|\omega(1)|$ have mean one.

Distributions of distance from 0 to $\gamma(1), \omega(1)$



How to reparameterize?

How should we parameterize SAW and $\text{SLE}_{8/3}$ so the parameterized curves have the same distribution?

- Easy question: Keep the capacity parameterization for SLE, reparameterize SAW.

Let C be the random time on the SAW where

$$(1) \quad \text{hcap}(\omega[0, C]) = 2$$

Compare $\omega(C)$ and $\gamma(1)$.

- Interesting question: Use natural parameterization for SAW, reparameterize SLE.

How to reparameterize SLE

Use *p*th variation with $p = 1/\nu$:

Let $0 = t_0^n < t_1^n < t_2^n \cdots < t_{k_n}^n = t$ be sequence of partitions of $[0, t]$.

$$fvar(\gamma[0, t]) = \lim_{n \rightarrow \infty} \sum_j |\gamma(t_j^n) - \gamma(t_{j-1}^n)|^{1/\nu}$$

With $\nu = 1/2$ this is the quadratic variation. For Brownian motion it is non-random and proportional to t .

How to reparameterize SLE

Use *p*th variation with $p = 1/\nu$:

Let $0 = t_0^n < t_1^n < t_2^n \cdots < t_{k_n}^n = t$ be sequence of partitions of $[0, t]$.

$$fvar(\gamma[0, t]) = \lim_{n \rightarrow \infty} \sum_j |\gamma(t_j^n) - \gamma(t_{j-1}^n)|^{1/\nu}$$

With $\nu = 1/2$ this is the quadratic variation. For Brownian motion it is non-random and proportional to t .

Conjecture: For the scaling limit of the SAW and other lattice models,

$$fvar(\omega[0, t]) = ct$$

How to reparameterize SLE

Use *p*th variation with $p = 1/\nu$:

Let $0 = t_0^n < t_1^n < t_2^n \cdots < t_{k_n}^n = t$ be sequence of partitions of $[0, t]$.

$$fvar(\gamma[0, t]) = \lim_{n \rightarrow \infty} \sum_j |\gamma(t_j^n) - \gamma(t_{j-1}^n)|^{1/\nu}$$

With $\nu = 1/2$ this is the quadratic variation. For Brownian motion it is non-random and proportional to t .

Conjecture: For the scaling limit of the SAW and other lattice models,

$$fvar(\omega[0, t]) = ct$$

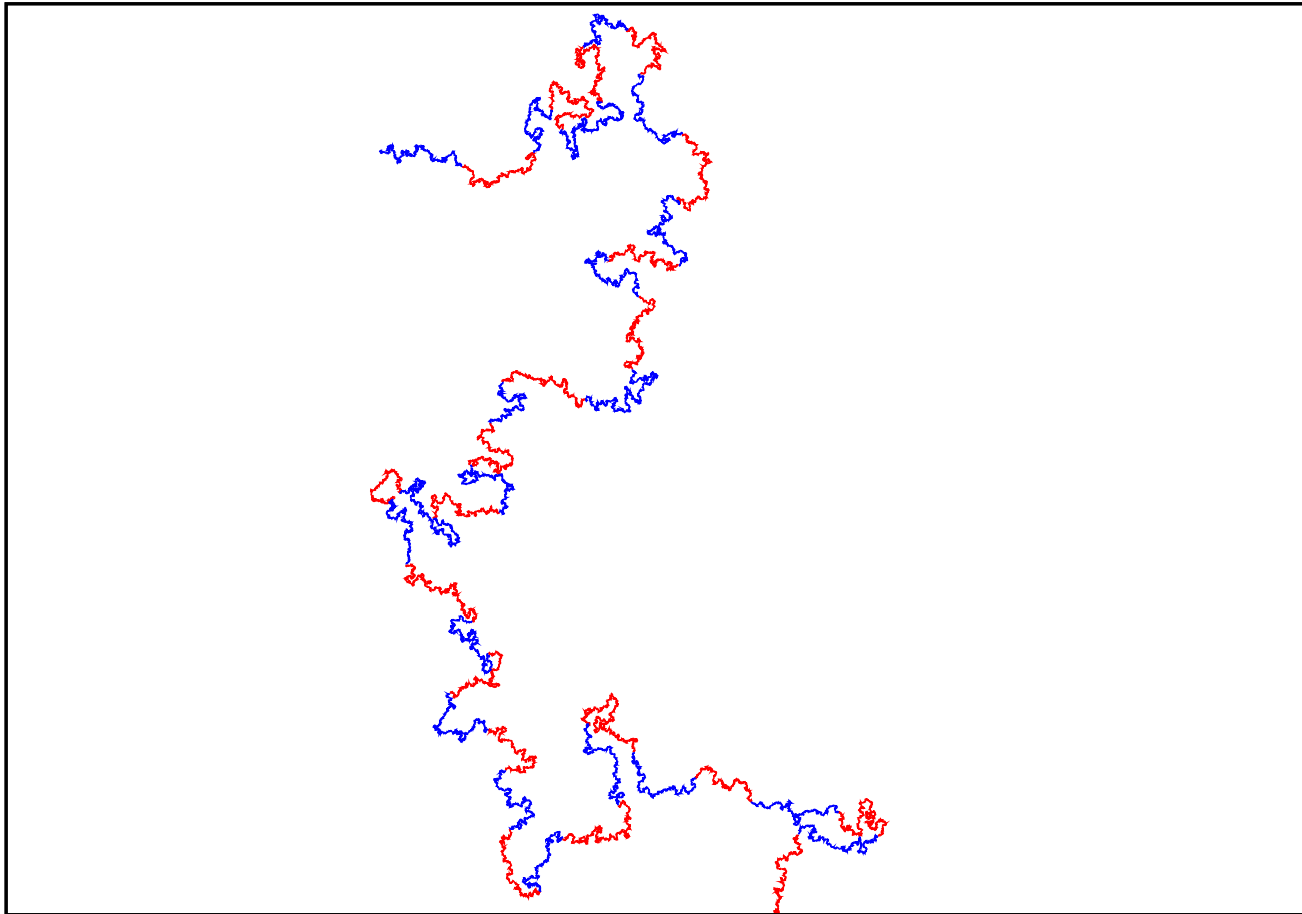
Key point: The $1/\nu$ variation of $\omega[0, t]$ is not random.

Fractal variation parameterization of SLE

If we take the same SLE curves from previous slide and parameterize them by p -variation and divide it into segments of equal variation, we get

Fractal variation parameterization of SLE

If we take the same SLE curves from previous slide and parameterize them by p -variation and divide it into segments of equal variation, we get



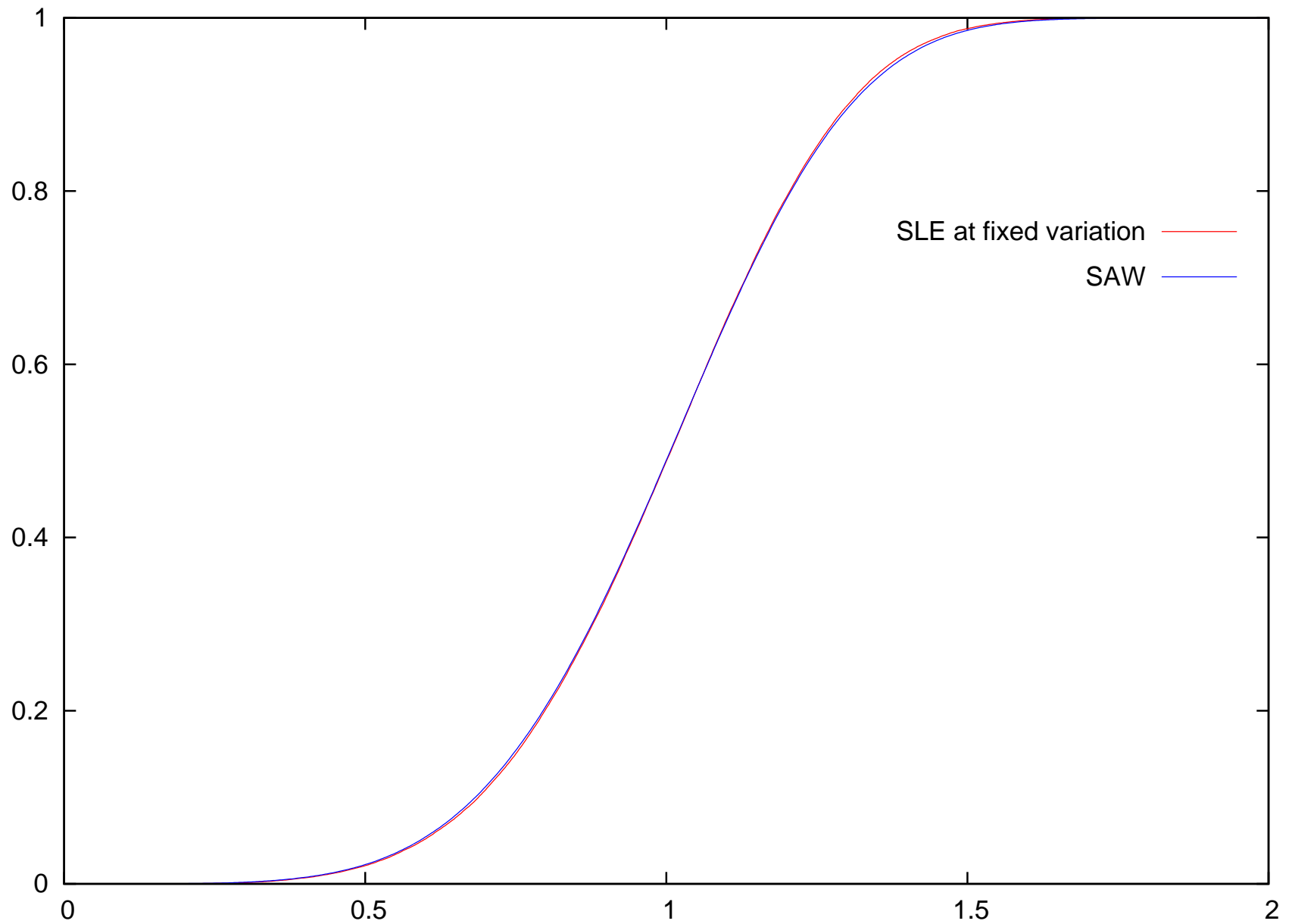
Comparing at constant fractal variation

Look at $\omega(1)$. This should be the same as looking at SAW at when fractal variation equals c .

Let T be the random time on the SLE when the fractal variation is c .

Compare $\omega(1)$ and $\gamma(T)$.

Distributions of distances for $\gamma(T), \omega(1)$



5.2 Future simulations - homework

- Length or natural parameterization: There are multiple definitions. Are they the same?
- SAW: There is no proof that it is SLE with $\kappa = 8/3$. All the simulations supporting this conjecture are for SAW with fixed number of steps in unbounded domain. Can you test SLE prediction for SAW in bounded domain with variable number of steps.
- SAW: use the fast algorithm to study things like ν in 3d, careful time series analysis for the autocorrelation time, ...
- SAW vs. “full plane SLE”
- Endpoint distribution of SAW: does SLE have anything to say about this?
- For SLE, $U_t = \sqrt{\kappa}B_t$. Let $l(t)$ be “length” of $\gamma[0, t]$. Let $V_{l(t)} = U_t$. What is the process V_l ?
- Bond avoiding walk can be simulated with essentially the same algorithm as SAW. It should have the same scaling limit.