June 1, 2008. This is very preliminary version of these notes. In particular the references are grossly incomplete. You should check my website to see if a newer version of the notes is available.

1 Introduction

These notes are about how to do various simulations that are related to the Schramm-Loewner Evolution (SLE) in some way. They are meant to be pedagogic in nature. Some of the algorithms we discuss are well know and so our discussion of them will be brief, referring the reader to various articles for more detail. For some of the algorithms that are not as well known, our discussion will be more detailed. Our goal is to provide the reader with a "how-to" guide that will enable him or her to do "state of the art" simulations related to SLE.

We begin with simulating SLE itself. Then we consider what might be called the inverse problem. Given a model of random curves, compute the Loewner driving process that describes them to see if the model is SLE. Then we turn our attention to various lattice models that are believed or proven to be described by SLE when they are at their critical point in two dimensions.

2 Simulations involving conformal maps

2.1 Lowener equation crash course

Let $\mathbb{H}$ denote the upper half of the complex plane,

$$\mathbb{H} = \{z : Im(z) > 0\} \tag{1}$$

If γ is a curve in $\mathbb{H}$ that starts at 0 and is simple (does not intersect itself), then $\mathbb{H} \setminus \gamma[0,t]$ is a simply connected domain. So there is a conformal map g_t from it onto $\mathbb{H}$. Thus such simple curves may be described by one parameter families of conformal maps. We start with a brief review of Loewner's equation for g_t and some of its properties. For a more detailed review from a physics point of view we refer the reader to [3, 9, 19]. For a full mathematical treatment we refer the reader to [14].

2.1.1 Loewner equation as correspondence between curves and driving functions

The Loewner equation provides a means for encoding curves in the upper half plane that do not intersect themselves by a real-valued function. Let $\gamma(t)$ be such a simple curve with $0 \leq t < \infty$. Let $\gamma[0,t]$ denote the image of γ up to time t. Then $\mathbb{H} \setminus \gamma[0,t]$ is a simply connected domain. So there is a conformal map g_t from this domain to $\mathbb{H}$. This map is not unique. We choose the

map that satisfies

$$g_t(z) = z + \frac{C(t)}{z} + O(\frac{1}{|z|^2}), \qquad z \to \infty \tag{2}$$

The coefficient $C(t)$ is increasing in t, and we can parameterize the curve so that $C(t) = 2t$. Then g_t satisfies the differential equation

$$\frac{\partial g_t(z)}{\partial t} = \frac{2}{g_t(z) - U_t}, \qquad g_0(z) = z \tag{3}$$

for some real valued function U_t on $[0, \infty)$. The function U_t is often called the driving function.

If our simple curve in the half plane is random, then the driving function U_t is a stochastic process. Schramm discovered that if the scaling limit of a two-dimensional model is conformally invariant and satisfies a certain Markov property, then this stochastic driving process must be a Brownian motion with mean zero [21]. The only thing that is not determined is the variance. Schramm named this process stochastic Loewner evolution or SLE; it is now often referred to as Schramm-Loewner evolution.

The solution to (3) need not exist for all times t since the denominator can go to zero. We let K_t be the set of points z in $\mathbb{H}$ for which the solution to this equation no longer exists at time t. If the driving function U_t is sufficiently nice, this set will be a curve $K_t = \gamma[0, t]$. But even for a continuous U_t it need not be [31]. In particular, when U_t is a Brownian motion, K_t may not be a curve. Even in the cases where U_t is not sufficiently nice, in our simulations our approximation to U_t will be nice enough that it produces a curve. So in the following we will always take K_t to be a curve, but the reader should keep in mind that in some cases this curve is approximating a more complicated set, more precisely it is approximating a curve that generates a more complicated set. ??

2.1.2 Breaking Loewner equation up into sequence of compositions

Let $t, s > 0$. The map g_{t+s} maps $\mathbb{H} \setminus \gamma[0, t + s]$ onto $\mathbb{H}$. We can do this in two stages. We first apply the map g_s. This maps $\mathbb{H} \setminus \gamma[0, s]$ onto $\mathbb{H}$, and it maps $\mathbb{H} \setminus \gamma[0, t+s]$ onto $\mathbb{H} \setminus g_s(\gamma[s, t+s])$. Let $\bar{g}_t$ be the conformal map that maps $\mathbb{H} \setminus g_s(\gamma[s, t + s])$ onto $\mathbb{H}$ with the usual hydrodynamic normalization. By the uniqueness of these maps,

$$g_{s+t} = \bar{g}_t \circ g_s, \quad i.e., \quad \bar{g}_t = g_{s+t} \circ g_s^{-1} \tag{4}$$

Then

$$\frac{d}{dt}\bar{g}_t(z) = \frac{d}{dt} g_{s+t} \circ g_s^{-1}(z) = \frac{2}{g_{s+t} \circ g_s^{-1}(z) - U_{s+t}} = \frac{2}{\bar{g}_t(z) - U_{s+t}} \tag{5}$$

Note that $\bar{g}_0(z) = z$. Thus $\bar{g}_t(z)$ is obtained by solving the Loewner equation with driving function $\bar{U}_t = U_{s+t}$. This driving function starts at U_s, and so the $\gamma(t)$ associated with $\bar{g}_t$ starts growing at U_s.

We now introduce a partition of the time interval $[0, \infty)$: $0 = t_0 < t_1 < t_2 < \cdots t_n < \cdots$, and define

$$\bar{g}_k = g_{t_k} \circ g_{t_{k-1}}^{-1} \tag{6}$$

So

$$g_{t_k} = \bar{g}_k \circ \bar{g}_{k-1} \circ \bar{g}_{k-2} \circ \cdots\cdots \bar{g}_2 \circ \bar{g}_1 \tag{7}$$

By the remarks above, $\bar{g}_k(z)$ is obtained by solving the Loewner equation with driving function $U_{t_{k-1}+t}$ for $t = 0$ to $t = \Delta_k$, where $\Delta_k = t_k - t_{k-1}$. Note that $\bar{g}_k$ maps $\mathbb{H}$ minus a "cut" starting at $U_{t_{k-1}}$ to $\mathbb{H}$. If we consider

$$g_k(z) = \bar{g}_k(z + U_{t_{k-1}}) - U_{t_{k-1}}, \tag{8}$$

it is obtained by solving the Loewner equation with driving function $U_{t_{k-1}+t} - U_{t_{k-1}}$ for $t = 0$ to $t = \Delta_k$. This driving function starts at 0 and ends at δ_k where $\delta_k = U_{t_k} - U_{t_{k-1}}$. So this conformal map takes $\mathbb{H}$ minus a cut starting at the origin onto $\mathbb{H}$. The inverse of this map,

$$g_k^{-1}(z) = \bar{g}_k^{-1}(z + U_{t_{k-1}}) - U_{t_{k-1}}, \tag{9}$$

takes $\mathbb{H}$ and introduces a cut which begins at the origin.

There are two general types of simulations we would like to do. Given a a driving function we want to find the curve it generates. And given a curve we want to find the corresponding driving function. For both problems the key idea is the same. We approximate the driving function on the interval $[t_{k-1}, t_k]$ by a function for which the Loewner equation may be explicitly solved. So the maps $\bar{g}_k$ and g_k can be found explicitly. Eq. (7) can then be used to approximate g_t. In the next section we consider some driving functions for which the Loewner equation can be explicitly solved.

2.1.3 Some explicitly solvable cases of Loewner eq.

One of the simplest explicit solutions of the Loewner equation is what we will refer to as "tilted slits." For $x_l, x_r > 0$ and $0 < \alpha < 1$, define

$$f(z) = (z + x_l)^{1-\alpha}(z - x_r)^{\alpha},$$

Then f maps $\mathbb{H}$ to $\mathbb{H} \setminus \Gamma$ where Γ is a line segment from 0 to a point $re^{i\alpha\pi}$. There are two degrees of freedom for the line segment - its length r and α. This map sends $[-x_l, x_r]$ onto Γ. Unfortunately, its inverse cannot be explicitly computed. For the inverse to satisfy the normalization (2), we must have $(1 - \alpha)x_l = \alpha x_r$. Straightforward calculation shows if we let

$$f_t(z) = \left(z + 2\sqrt{t}\sqrt{\frac{\alpha}{1-\alpha}}\right)^{1-\alpha} \left(z - 2\sqrt{t}\sqrt{\frac{1-\alpha}{\alpha}}\right)^{\alpha}$$

then it produces a slit with capacity $2t$. We know from the general theory that $g_t = f_t^{-1}$ satisfies the Loewner equation (3) for some driving function U_t. Some calculation then shows that

$$U_t = c_\alpha \sqrt{t} \tag{10}$$

where

$$c_\alpha = 2\frac{1-2\alpha}{\sqrt{\alpha(1-\alpha)}} \tag{11}$$

The change in the driving function over the time interval $[0, \Delta]$ is

$$\delta = c_\alpha \sqrt{\Delta} \tag{12}$$

The original map ϕ had three real degrees of freedom, α, x, y. The normalization (2) reduces this to two real degrees of freedom, α and t. So if are given δ and Δ, then the map is completely determined.

An even simpler solution of the Loewner equation is "vertical slits." Let

$$g_t(z) = \sqrt{(z-\delta)^2 + 4t} + \delta$$

Then it is easy to see that g_t satisfies Lowener's equation with a constant diving function, $U_t = \delta$. Since the driving function does not start at 0, the curve will not start at the origin. The curve is just a vertical slit from δ to $\delta + 2i\sqrt{t}$. Using vertical slits means that we approximate the driving function by a discontinuous piecewise constant function. This will produce a K_t which is not a curve.

2.2 Simulating SLE, more generally driving function $\Rightarrow$ curve

2.2.1 The zipping algorithm

Our primary motivation is to simulate SLE, i.e., to compute the curve when the driving function is Brownian motion. But the following discussion applies to any U_t.

One method for simulating SLE is to numerically solve the Loewner equation for many initial conditions and see which initial conditions are in K_t and which are not. See Vincent Beffara's web page for more on this.

Let $0 = t_0 < t_1 < t_2 < \cdots < t_n$ be a partition of the time interval $[0, t]$. The approximation to the SLE trace is given by $\gamma(t) = g_t^{-1}(U_t)$. Let $z_k = g_{t_k}^{-1}(U_{t_k})$. We will only consider the points on this curve which correspond to times $t = t_k$. One could consider other points on the curve, but the distance between consecutive z_k is already of the order of the error in our approximation, so there is little reason to consider more points. The points z_k are given by

$$z_k = \bar{g}_1^{-1} \circ \bar{g}_2^{-1} \circ \cdots\cdots \bar{g}_{k-1}^{-1} \circ \bar{g}_k^{-1}(U_{t_k}) \tag{13}$$

Recall that if we solve the Loewner equation with driving function $U_{t_{k-1}+t} - U_{t_{k-1}}$ for $t = 0$ to $t = \Delta_k$, we get $g_k(z)$ where

$$g_k(z) = \bar{g}_k(z + U_{t_{k-1}}) - U_{t_{k-1}} \tag{14}$$

Define

$$h_k(z) = \bar{g}_k(z) - \delta_k = \bar{g}(z + U_{t_{k-1}}) - U_{t_k} \tag{15}$$

where $\delta_k = U_{t_k} - U_{t_{k-1}}$. Then

$$h_k \circ h_{k-1} \circ \cdots\cdots \circ h_1(z_k) = \bar{g}_k \circ \bar{g}_{k-1} \circ \cdots\cdots \circ \bar{g}_1(z_k) - U_{t_k} = 0. \tag{16}$$

Let

$$f_k = h_k^{-1} \tag{17}$$

So

$$z_k = f_1 \circ f_2 \circ \cdots \circ f_k(0) \tag{18}$$

As noted before, g_k maps $\mathbb{H}$ minus a small curve onto $\mathbb{H}$. The driving function ends at δ_k, so g_k sends the tip of the curve to δ_k. It follows that $h_k(z) = g_k(z) - \delta_k$ maps $\mathbb{H}$ minus the small curve onto $\mathbb{H}$ and sends the tip to the origin. So $f_k = h_k^{-1}$ maps $\mathbb{H}$ onto $\mathbb{H}$ minus the small curve and sends the origin to the tip of the curve.

Thus we have the following simple picture for eq. (18). The first map f_k welds together a small interval on the real axis containing the origin to produce a small cut. The origin is mapped to the tip of this cut. The second map f_{k-1} welds together a (possibly different) small interval in such a way that it produces another small cut. The original cut is moved away from the origin with its base being at the tip of the new cut. This process continues. Each map introduces a new small cut whose tip is attached to the image of the base of the previous cut.

As discussed before, we define U_t on each time interval $t_{k-1} \leq t \leq t_k$ so that $g_k(z)$ may be explicitly computed. There are two constraints on g_k. The curve must have capacity $2\Delta_k$ and g_k must map the tip of the curve to δ_k. Given any simple curve satisfying these two constraints and starting at the origin, it will be the solution of the Loewner equation for some driving function which goes from 0 to δ_k over the time interval $[0, \Delta_k]$. Hence (18) gives points on a curve that approximates the curve corresponding to the exact driving function. Different choices of how we define U_t on each time interval give us different discretizations. As we will see, this choice will not have a significant effect. Of much greater importance is how we choose the Δ_k and δ_k.

If we want to simulate SLE, the δ_k should be chosen so that the stochastic process U_t will converge to $\sqrt{\kappa}$ times Brownian motion as $N \to \infty$. One possibility is take the δ_k to be independent normal random variables with mean zero and variance $\kappa\Delta_k$. If we do this, then U_t and $\sqrt{\kappa}B_t$ will have the same distributions if we only consider times chosen from the t_k. Another possibility is to take the δ_k to be independent random variables with $\delta_k = \pm\sqrt{\kappa\Delta_k}$ where the choices of $+$ and $-$ are equally probable. If we use this choice with the uniform partition of the time interval, then we are approximating the Brownian motion by a simple random walk.

The simplest choice for Δ_k is to use a uniform partition of the time interval. For values of κ which are not too large this works reasonbly well. Figure 1 shows a simulation with $\kappa = 8/3$ with $10,000$ equally spaced time intervals.

However, for larger values of κ, uniform Δ_k are a disaster. Figure 2 shows a simulation with $\kappa = 6$ and $10,000$ equally spaced time intervals. Clearly something has gone wrong. To see just how badly wrong things have gone the reader should compare this figure with figure 3 which used the same sample of Brownian motion.

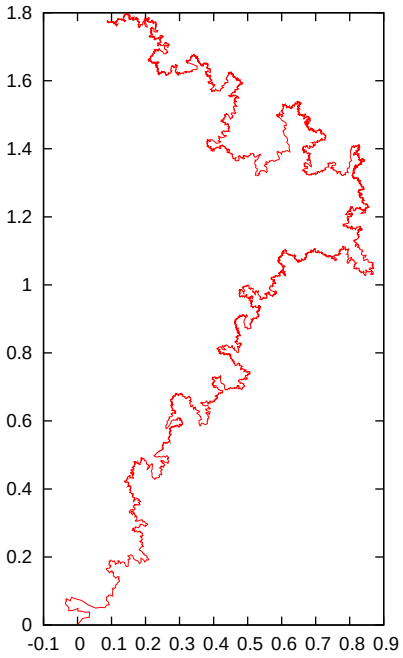


Figure 1: SLE with $\kappa = 8/3$ with fixed Δt. There are $10,000$ points.

To understand the effect seen in (2) we give another equivalent definition of the (half plane) capacity C of a set A. Before we defined it by

$$g(z) = z + \frac{C}{z} + O(\frac{1}{z^2})$$

where g maps $\mathbb{H} \setminus A$ onto $\mathbb{H}$, usual normalizations. A more intuitive definition is

$$C = \lim_{y \to \infty} y \, E^{iy}[Im(B_\tau)]$$

where B_t is two-dimensional Brownian motion started at iy. The stopping time τ is the first time the Brownian motion hits A or $\mathbb{R}$. From the point of view of the Brownian motion, parts of the curve can be well hidden by earlier parts of the curve and so have very little capacity. So what looks like a "long" section of the curve has very little capacity and so gets very few points approximating it.

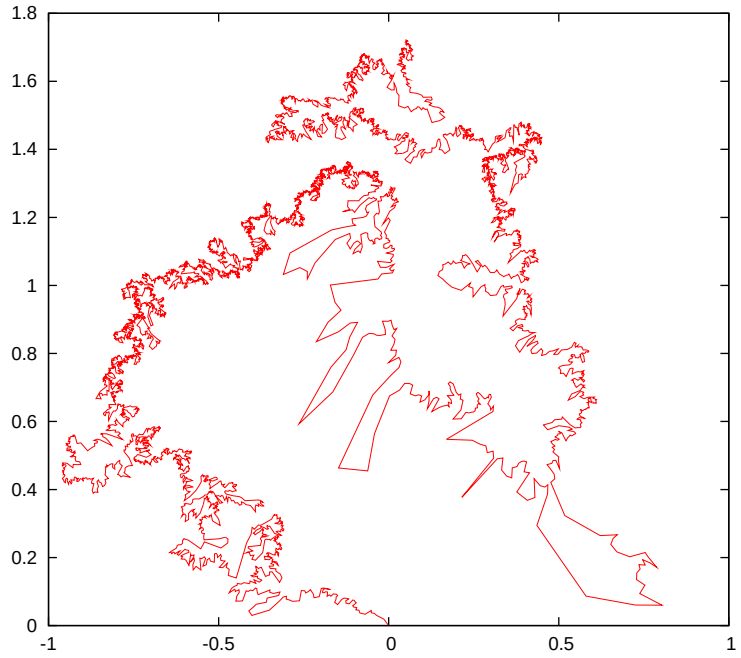


Figure 2: SLE with $\kappa = 6$ with fixed Δt. There are $10,000$ points.

2.2.2 Adaptive Δ_k

To do better we will choose use non-uniform Δ_k. In fact they will depend on the sample of the Brownian motion and so we refer to this method as "adaptive Δ_k." I learned this idea from Stephen Rhöde [32].

Fix a spatial scale $\epsilon > 0$. We start with uniform partition of the time and compute the points z_k along the curve. Look for points z_k such that $|z_k - z_{k-1}| \geq \epsilon$. For these time intervals $[t_{k-1}, t_k]$, divide the interval into two equal intervals. We then sample the Brownian motion at midpoint of $[t_{k-1}, t_k]$ using a Brownian bridge. Then we compute new the new points z_k. (There will of course be more of them than before.) We repeat this until we have $|z_k - z_{k-1}| \leq \epsilon$ for all k.

MORE: Explain Brownian bridge.

2.2.3 Effect of the choice of the cut

In this section we compare the curves we get using tilted slits for the elementary maps with the curves we get using vertical slits. To do this we carry out the adaptive simulation described in the previous section using tilted slits. We then use the same Δ_k and δ_k, i.e., the same partition of the time interval and the same sample of Brownian motion, but with vertical slits. For $\kappa = 8/3$, figure 2.2.3 shows the tilted slits curve vs. the vertical slits curve. There are about

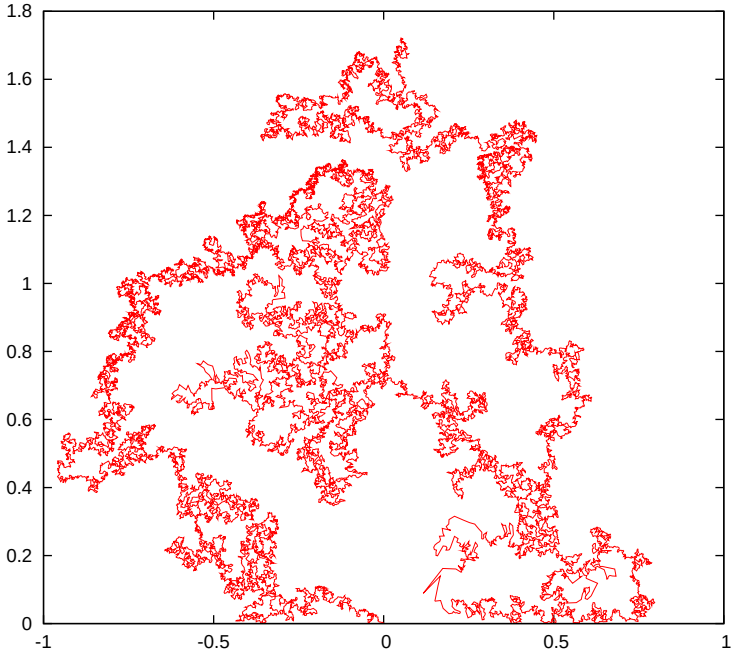


Figure 3: SLE with $\kappa = 6$ with adaptive Δt. There are $35,000$ points.

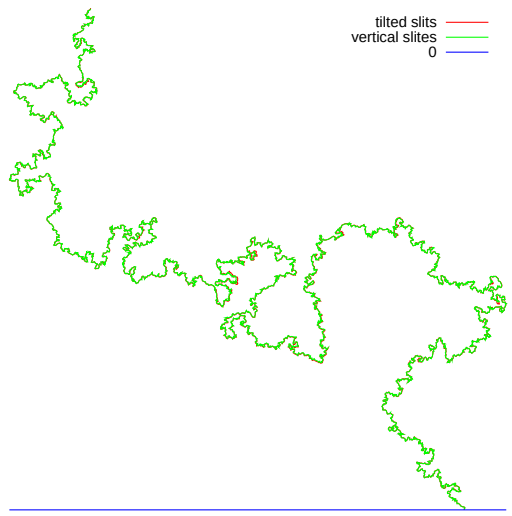
$35,000$ points on the curves. The vertical slits do not produce a curve. What we have plotted is the following. We compute the points z_k and then simply connected them with a straight line. 2.2.3 shows an enlargment of part of the figure.

In the first figure it is almost impossible to distinguish the two curves. Even in the enlargement the difference is quite small. Figures 2.2.3 and 2.2.3 show the same thing with $\kappa = 6$. In the enlargement one can see deviations between the two curves. The size of the deviations is roughly on the same scale as the distance between adjacent points on the curve.

2.3 Inverse SLE: curve $\Rightarrow$ driving function

We now consider what one might call the inverse problem. Given a simple curve γ, we want to compute the corresponding driving function. As we will see, it is natural to call this the unzipping algorithm.

One motivation for doing this is that it gives a way to test if a given model is in fact SLE. One generates a bunch of random curves in the model, computes their driving functions and then tests if they could be samples of a Brownian motion. There is a long list of models whose scaling limit is described by SLE. For many of these models this has been proved and for the others it is supported by theoretical arguments and simulations. More recent work has considered models for which the connection with SLE is more speculative. The possibility that domain walls in spin glass ground states are SLE curves was studied numerically both by Amoruso, Hartman,



Hastings, and Moore [1] and by Bernard, Le Doussal, and Middleton [7]. Bernard, Boffetta, Celani and Falkovich considered simulations of certain isolines in two-dimensional turbulence [5] and surface quasi-geostrophic turbulence [6].

In almost all applications, the parameterization of the curve is not the parameterization by capacity. Let g_s be the conformal map which takes the half plane minus $\gamma[0,s]$ onto the half plane, normalized so that for large z

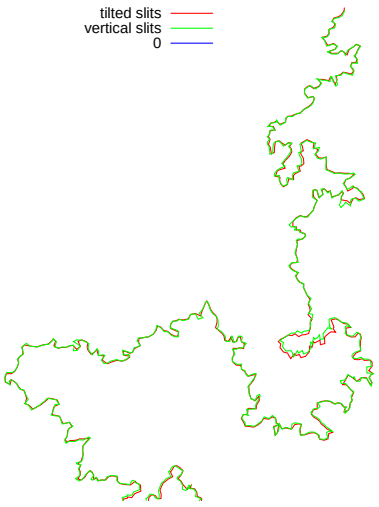
$$g_s(z) = z + \frac{C(s)}{z} + O(\frac{1}{z^2}), \tag{19}$$

The coefficient $C(s)$ is the half-plane capacity of $\gamma[0,s]$. The value of the driving function at t is $U_t = g_s(\gamma(s))$. Thus computing the driving function essentially reduces to computing this uniformizing conformal map. We will describe the "zipper algorithm" for doing this [13, 18]. A comment on terminology is in order. We use "zipper algorithm" to refer to all the various algorithms we can get from different choices of the curve γ_{k+1}. Marshall and Rohde [18] use "zipper" to refer only to the choice using tilted slits. Another approach to computing the driving function may be found in [24].

Let $z_0, z_1, \cdots, z_n$ be points along the curve with $z_0 = 0$. In many applications these are lattice sites. Our goal is to find a sequence of conformal maps h_i, $i = 1, 2, \cdots, n$ such that $h_k \circ h_{k-1} \circ \cdots \circ h_1(z_k) = 0$. Then $h_k \circ h_{k-1} \circ \cdots \circ h_1$ sends $\mathbb{H} \setminus \gamma$ to $\mathbb{H}$ where γ is some curve that passes through $z_0, z_1, \cdots z_k$, . Suppose that the conformal maps $h_1, h_2, \cdots, h_k$ have been defined with these properties. Let

$$w_{k+1} = h_k \circ h_{k-1} \circ \cdots \circ h_1(z_{k+1}) \tag{20}$$

Then w_{k+1} is close to the origin. We define h_{k+1} to be a conformal map sends $\mathbb{H} \setminus \gamma_{k+1}$ to $\mathbb{H}$ where γ_{k+1} is a short simple curve that ends at w_{k+1}. We also require that h_{k+1} sends w_{k+1}



to the origin. As before we choose this curve so that h_{k+1} is explicitly known. Two possible choices are "tilted slits" and "vertical slits." Note that for both of these maps there were two real degrees of freedom. They will be determined by the condition that $h_{k+1}(w_{k+1}) = 0$.

Let $2\Delta_i$ be the capacity of the map h_i, and δ_i the final value of the driving function for h_i. So

$$h_i(z) = z - \Delta_i + \frac{2\delta_i}{z} + O(\frac{1}{z^2}) \tag{21}$$

Then

$$h_k \circ h_{k-1} \circ \cdots \circ h_1(z) = z - U_t + \frac{2t}{z} + O(\frac{1}{z^2}) \tag{22}$$

where

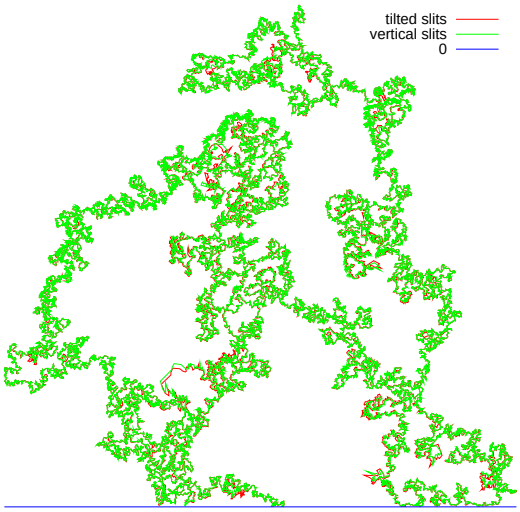
$$t = \sum_{i=1}^{k} \Delta_i \tag{23}$$

$$U_t = \sum_{i=1}^{k} \delta_i \tag{24}$$

Thus the driving function of the curve is obtained by "adding up" the driving functions of the elementary conformal maps h_i.

2.4 Faster: blocking functions to get faster algorithm

In this section we show how to greatly speed up both the algorithm for compute the curve γ given the driving function U_t and the algorithm for computing the driving function U_t given a curve γ. We start with the first algorithm. One of the main motivations is a fast algorithm for simulating SLE, but our fast algorithm is applicable to other driving functions as well.



Recall that points on the approximation to the SLE trace or more generally the curve γ are given by eq. (18) which says

$$z_k = f_1 \circ f_2 \circ \cdots \circ f_k(0) \tag{25}$$

The number of operations needed to compute a single z_k is proportional to k. So to compute all the points z_k with $k = 1, 2, \cdots N$ requires a time $O(N^2)$.

It is important to note that the computation of z_k does not depend on any of the other z_j. So we can compute a subset of the points z_k if we desire. (As an extreme example, if we are only interested in $z_N = \gamma(t_N)$, the time required for the computation is $O(N)$ not $O(N^2)$.) For the timing tests in this paper we compute the points z_{jd} with $j = 1, 2, \cdots, N/d$ where d is some integer. But we emphasize that our algorithm works for any choice of the set of points to compute.

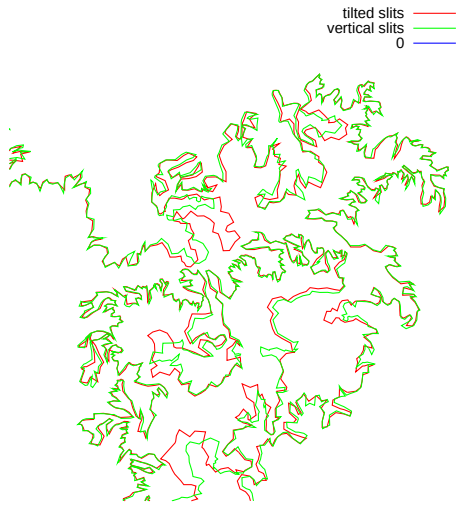
For the above algorithm the time grows as N^2/d. We will refer to the time it takes to compute one point on the SLE trace as the "time per point computed" (TPPC). The (TPPC) grows as N for the naive algorithm. We use the TPPC throughout these notes to study the efficiency. It is a natural measure since it depends on how finely we discretize the time interval but not on the number of points we choose to compute. The total time to compute the SLE trace is given by the number of points we want to compute on it times the TPPC. Our goal is to develop an algorithm for which the TPPC is $O(N^p)$ with $p < 1$.

Our algorithm begins by grouping the functions in (25) into blocks. The number of functions in a block will be denoted by b. Let

$$F_j = f_{(j-1)b+1} \circ f_{(j-1)b+2} \circ \cdots \circ f_{jb} \tag{26}$$

If we write k as $k = mb + l$ with $0 \le l < b$, then we have

$$z_k = F_1 \circ F_2 \circ \cdots \circ F_m \circ f_{mb+1} \circ f_{mb+2} \circ \cdots \circ f_{mb+l}(0) \tag{27}$$



The number of compositions in (27) is smaller than the number in (25) by roughly a factor of b. Unfortunately, even though the f_i are relatively simple, the F_j cannot be explicitly computed. Our strategy is to approximate the f_i by functions whose compositions can be explicitly computed to give an explicit approximation to F_j. For large z, $f_i(z)$ is well approximated by its Laurent series about ∞. One could approximate f_i by truncating this Laurent series. This is the spirit of our approach, but our approximation is slightly different.

Let $\gamma : [0,t] \to \mathbb{H}$ be a curve in the upper half plane with $\gamma(0) = 0$. Let $f(z)$ be the conformal map from $\mathbb{H}$ onto $\mathbb{H} \setminus \gamma[0,t]$, We assume that f is normalized is the same way as our f_i, i.e., $f(\infty) = \infty$, $f'(\infty) = 1$ and $f(0) = \gamma(t)$. Let $a, b > 0$ be such that $[-a, b]$ is mapped onto the slit $\gamma[0,t]$. So f is real valued on $(-\infty, -a]$ and $[b, \infty)$. By the Schwartz reflection principle, f has an analytic continuation to $\mathbb{C} \setminus [-a, b]$, which we will simply denote by f. Let $R = \max\{a, b\}$, so f is analytic on $\{z : |z| > R\}$ and maps ∞ to itself. Thus $f(1/z)$ is analytic on $\{z : 0 < |z| < 1/R\}$ and our assumptions on f imply it has a simple pole at the origin with residue 1. So we have

$$f(1/z) = 1/z + \sum_{k=0}^{\infty} c_k \, z^k \tag{28}$$

This gives the Laurent series of f about ∞.

$$f(z) = z + \sum_{k=0}^{\infty} c_k \, z^{-k} \tag{29}$$

If we truncate this Laurent series, it will be a good approximation to f for large z. At first sight, this Laurent series is the natural approximation to use for f. However, we will use a different but closely related representation.

Define $\hat{f}(z) = 1/f(1/z)$. Since $f(z)$ does not vanish on $\{|z| > R\}$, $\hat{f}(z)$ is analytic in $\{z : |z| < 1/R\}$. Our assumptions on f imply that $\hat{f}(0) = 0$ and $\hat{f}'(0) = 1$. So $\hat{f}$ has a power

series of the form

$$\hat{f}(z) = \sum_{j=0}^{\infty} a_j z^j \tag{30}$$

with $a_0 = 0$ and $a_1 = 1$. It is not hard to show that $1/R$ is the radius of convergence of this power series. We will refer to this power series as the "hat power series" of f. Note that the coefficients of the hat power series of f are the coefficients of the Laurent series of $1/f$.

Need better name for hat power series

The primary advantage of this hat power series over the Laurent series is its behavior with respect to composition.

$$(f_1 \circ f_2)\hat{}(z) = \frac{1}{f_1((f_2(1/z))} = \frac{1}{f_1(1/\hat{f}_2(z))} = \hat{f}_1(\hat{f}_2(z)) \tag{31}$$

Thus

$$(f_1 \circ f_2)\hat{} = \hat{f}_1 \circ \hat{f}_2 \tag{32}$$

Our approximation for $f(z)$ is to replace $\hat{f}(z)$ by the truncation of its power series at order n. So

$$f(z) = \frac{1}{\hat{f}(1/z)} \approx \left[\sum_{j=0}^{n} a_j z^{-j}\right]^{-1} \tag{33}$$

For each f_i we compute the power series of $\hat{f}_i$ to order n. We then use them and (32) to compute the power series of $\hat{F}_j$ to order n. Let $1/R_j$ be the radius of convergence for the power series of $\hat{F}_j$. (R_j is easy to compute. It is the smallest positive number such that $F_j(R_j)$ and $F_j(-R_j)$ are both real.) Now consider equation (27). If z is large compared to R_j, then $F_j(z)$ is well approximated using its hat power series. We introduce a parameter $L > 1$ and use the hat power series to compute $F_j(z)$ whenever $|z| \geq LR_j$. When $|z| < LR_j$, we just use (26) to compute $F_j(z)$. The argument of F_j is the result of applying the previous conformal maps to 0, and so is random. Thus whether or not we can approximate a particular F_j using its hat power series depends on the randomness and on which z_k we are computing.

As part of the algorithm we must compute R_j. This is easy. R_j is the smallest positive number such that $F_j(R_j)$ and $F_j(-R_j)$ are both real.

In addition to the choice of simple curves we use (tilted slits, vertical slits, ...), there are three parameters in our algorithm. b is the number of functions composed in a block. n is the order at which we truncate our series approximation. L is the scale that determines when we use series for F_j. The parameter b has little effect on the accuracy of the algorithm and we should choose it to make the algorithm run as quickly as possible. Eq. (27) suggests that b should vary with N as $\sqrt{N}$ and experiments bear this out. For unzipping the SAW with $N = 1,000$ to $500,000$, we have used $b = 20$ to 200.

The choice of n involves a tradeoff of speed vs. accuracy. Larger n means more terms in the series, hence slower but more accurate computations. We typically use $n = 12$.

The parameter L will determine how fast the series converges. Roughly speaking, the series will converges at least as fast as the geometric series $\sum_n L^{-n}$. The choice of L also involves a tradeoff of speed vs. accuracy. Larger L means the seris converges faster and so is more accurate. But it also means that we use the block functions F_j less frequently, and so the computation is slower. We typically use $L = 4$.

To study the speed of the new algorithm, we have simulated SLE with $\kappa = 6$ and up to $1,000,000$ points on the curve. Figure 4 shows the time per point computed vs. N the number of points. The curve for the naive algorithm is well fit by a line with slope 1 as we expect. The curve for our faster algorithm is fit by a line with slope around 0.4. In this computation we use different choices of the block size b for different N to optimize the speed.

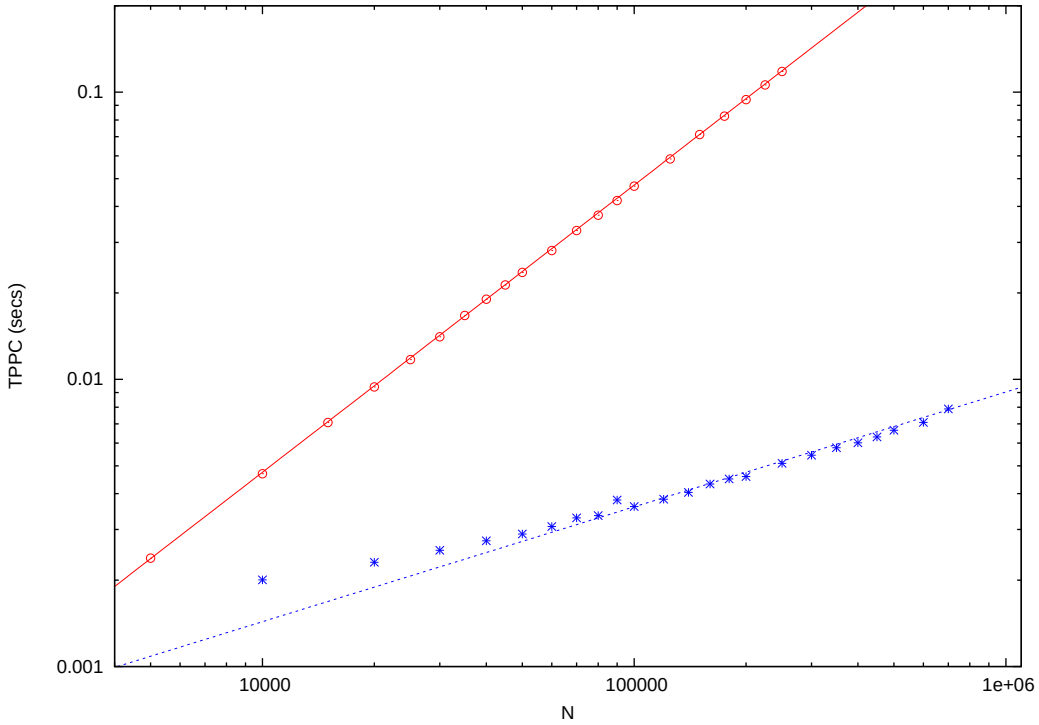


Figure 4: Time per point computed (TPPC) as a function of N, the number of subintervals in the partition of the time interval for $\kappa = 8/3$. The top curve is the naive algorithm; the bottom curve is the algorithm using blocking. The lines shown have slopes 1 and 0.4.

To study the accuracy of our series approximation we compute an SLE curve for $\kappa = 6$ with and without the series approximation. We use the same Brownian motion sample path for both curves. The two curves are shown in figure 5. They cannot be distinguished. An enlargement is shown in 6.

We now consider the zipper algorithm for computing the driving function of a given curve. The number of operations needed to compute a single w_{k+1} is proportional to k. So to compute

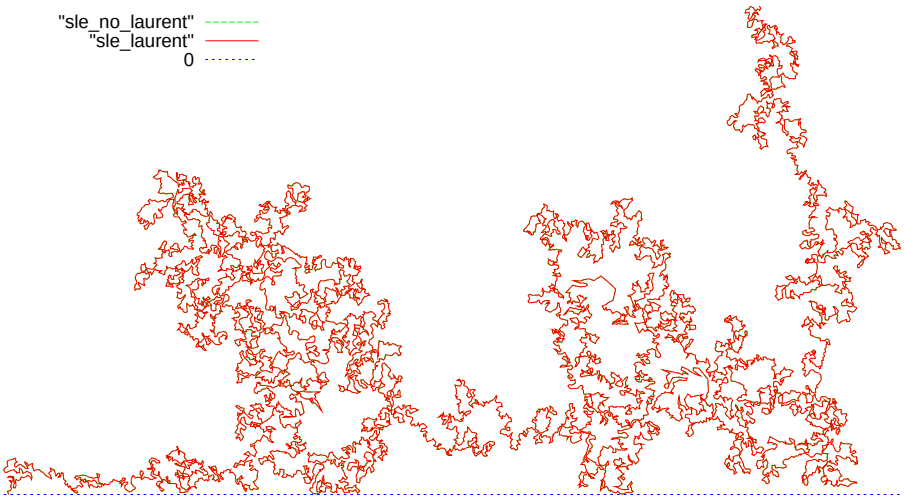


Figure 5: Two curves for SLE with $\kappa = 6$ are shown. They use the same Brownian motion sample path but one uses the series approximation and the other does not.

all the points w_{k+1} requires a time $O(N^2)$. The idea for improving this is the same as before - we group the functions we are composing into blocks and approximate the composition of the functions in a block using the hat power series. The only minor difference is that the order of the conformal maps in (20) is the opposite of that in (18). We continue to denote the number of functions in a block by b. Let

$$H_j = h_{jb} \circ h_{jb-1} \circ \cdots \circ h_{(j-1)b+2} \circ h_{(j-1)b+1} \tag{34}$$

If we write k as $k = mb + r$ with $0 \le r < b$, then (20) becomes

$$w_{k+1} = h_{mb+r} \circ h_{mb+r-1} \circ \cdots \circ h_{mb+1} \circ H_m \circ H_{m-1} \circ \cdots \circ H_1(z_{k+1}) \tag{35}$$

As before, the h_i are relatively simple, but the composition H_j cannot be explicitly computed. We approximate h_i by its hat power series and compute the compositions in (34) just once rather than every time we compute a w_k.

Recall that h_i is normalized so that $h_i(\infty) = \infty$ and $h_i'(\infty) = 1$. It maps $\mathbb{H}$ minus a simple curve near the origin to $\mathbb{H}$, sending the tip of the curve to the origin. Let h denote such a conformal map. Let R be the largest distance from the origin to a point on the curve. Then h is analytic on $\{z \in \mathbb{H} : |z| > R\}$. Note that h is real valued on the real axis. By the Schwarz reflection principle it may be analytically continued to $\{z \in \mathbb{C} : |z| > R\}$. Moreover, it does not vanish on this domain.

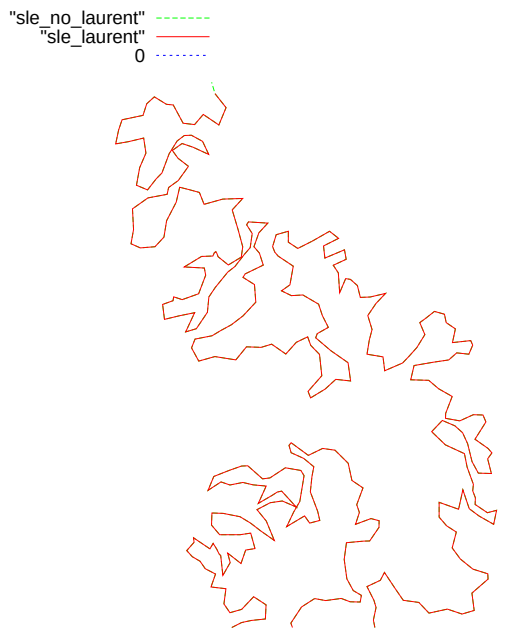


Figure 6: An enlargement of a part of the previous figure.

So if we let $\hat{h}(z) = 1/h(1/z)$, then $\hat{h}(z)$ is analytic in $\{z \in \mathbb{C} : |z| < 1/R\}$. Our assumptions on h imply that $\hat{h}(0) = 0$ and $\hat{h}'(0) = 1$. So $\hat{h}$ has a power series of the form

$$\hat{h}(z) = \sum_{j=1}^{\infty} a_j z^j \tag{36}$$

with $a_1 = 1$. The radius of convergence of this power series is $1/R$. Note that the coefficients of this power series are the coefficients of the Laurent series of $1/h$.

As before, the advantage of working with the power series of $\hat{h}$ is its behavior with respect to composition. $(h_1 \circ h_2)\hat{} = \hat{h_1} \circ \hat{h_2}$ Our approximation for $h_i(z)$ is to replace $\hat{h_i}(z)$ by the truncation of its power series at order n. So

$$h_i(z) = \frac{1}{\hat{h_i}(1/z)} \approx \left[\sum_{j=1}^{n} a_j z^{-j} \right]^{-1} \tag{37}$$

For each h_i we compute the power series of $\hat{h_i}$ to order n. We then use them to compute the power series of $\hat{H}_j$ to order n. Let $1/R_j$ be the radius of convergence for the power series of $\hat{H}_j$. Now consider equation (35). If z is large compared to R_j, then $H_j(z)$ is well approximated using the power series of $\hat{H}_j$. We introduce a parameter $L > 1$ and use this series to compute $H_j(z)$ whenever $|z| \geq LR_j$. When $|z| < LR_j$, we just use (34) to compute $H_j(z)$. The argument of

H_j is the result of applying the previous conformal maps to some z_{k+1}, and so is random. Thus whether or not we can approximate a particular H_j by its series depends on the randomness and on which w_{k+1} we are computing.

We need to compute R_j. Consider the images of $z_{(j-1)b}, z_{(j-1)b+1}, \cdots z_{jb-1}$ under the map $H_{j-1} \circ H_{j-2} \circ \cdots \circ H_1$. The domain of the conformal map H_j is the half-plane $\mathbb{H}$ minus some curve Γ_j which passes through the images of these points. The radius R_j is the maximal distance from the origin to a point on Γ_j. This distance should be very close to or even equal to the maximum distance from the origin to images of $z_{(j-1)b}, z_{(j-1)b+1}, \cdots z_{jb-1}$ under $H_{j-1} \circ H_{j-2} \circ \cdots \circ H_1$. So in our algorithm we approximate R_j by the maximum of these distances.

The improvement in the speed of the zipper algorithm from using our power series approximation is shown in table 1 and figure 7. In these timing tests we use a single SAW with one million steps. We time how long it takes to unzip the first N steps with and without the power series approximation. We do the computations using the power series approximation for different choices for the block length, namely $b = 20, 30, 40, 50, 75, 100, 200, 300$, and report the fastest time. The last column in the table indicates the block length that achieves the fastest time. As a rule of thumb, a good choice for the block length (at least for the SAW) is $b = \sqrt{N}/4$. The next to last column in the table gives the factor by which the use of the power series approximation reduces the time needed for the computation. These timing tests were done on a PC with a 3.4 GHz Pentium 4 processor.

Without the power series approximation the time is $O(N^2)$. This is seen clearly in the log-log plot in figure 7 where the data for unzipping without the power series approximation is fit quite well by a line with slope 2. The data for unzipping using the power series approximation is fit by a line with slope 1.35. This indicates that the time required when the power series are used is approximately $O(N^{1.35})$.

2.5 Zipper algorithm for finding conformal map of given domain

The zipper algorithm of [13, 18] is actually an algorithm for finding a conformal map of a given simply connected domain to a standard simply connected domain, e.g., the half plane or unit disc. The algorithm we described in previous sections is only part of this algorithm for finding conformal maps. In this section we review the full algorithm for finding conformal maps. This has nothing to do with SLE or random curves in the plane. We include this discussion since the ideas of the previous section for speeding up the algorithm apply here as well.

MORE explain zipper

2.6 Open problems/projects - homework

In this section we list some projects related to the algorithms we have discussed.

- We have only discussed the simulation of chordal SLE. The simulation of radial SLE is similar. Can one use the ideas we used to speed up the simulation of chordal SLE to speed up the simulation of radial SLE?

N	time 1	time 2	factor	block length
1,000	0.21	0.43	0.50	20
2,000	0.86	0.95	0.91	20
5,000	5.44	3.00	1.81	20
10,000	21.44	7.41	2.89	30
20,000	85.65	18.31	4.68	40
50,000	534.8	62.6	8.54	50
100,000	2128	158	13.45	75
200,000	8562	437	19.59	100
500,000	53516	1674	31.98	200
1,000,000	214451	4675	45.87	200

Table 1: The time (in seconds) needed to unzip a SAW with N steps without using the power series approximation is shown in the second column (time 1) The time using the power series approximation is shown in the third column (time 2). The fourth column (factor) is the ratio of these two times. The block length used is in the last column.

- Given a simple (no self-intersections) loop, its interior is a simply connected region. So there is a conformal map from it to the unit disc. There is also a conformal map from the exterior of the loop to the exteriof of the unit disc. Together these two maps define a homeomorphism of the unit circle onto itself. This function is sometimes called the fingerprint. If the simple loop is random, the fingerprint will be random. Can you simulate this stochastic process ?

- Instead of taking the scaling limit at the critical point, one can consider off critical models and take the scaling limit in such a way that it has a finite correlation length. What can you say about the driving process for this scaling limit ? For percolation it is know to be rather nasty [20]. See also [4].

- There are several methods for numerically computing the conformal map of a given simply connected domain onto a standard domain, like the unit disc. **refs** How does our faster version of the zipper algorithm compare with them?

- Use the ideas presented here for a fast simulation of the SLE(κ,ρ) processes.

- The code in sim_SLE is not very user friendly. Is there a better way to package this, e.g., as Matlab code, or Mathematica code.

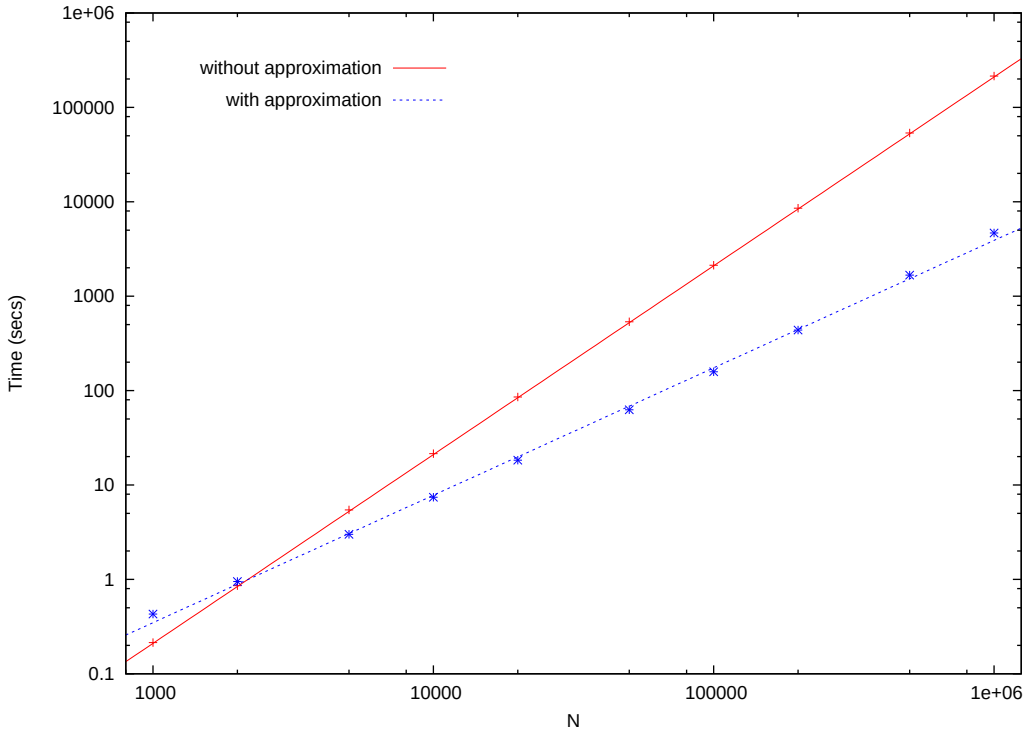


Figure 7: The points are the time (in seconds) needed to unzip a SAW with N steps with and without the power series approximation. The lines have slopes 2 and 1.35.

Acknowledgments:
I thank Don Marshall and Stephen Rohde for useful discussions about the zipper algorithm. Talks and interactions during visits to Banff ?? and to the Kavli Institute for Theoretical Physics in September, 2006 contributed to the research included in these notes. The opportunity to present this material at the 2008 Enrage summer school at IHP is warmly acknowledged. This research was supported in part by the National Science Foundation under grant DMS-0201566,DMS-0501168.

References

[1] C. Amoruso, A. K. Hartman, M. B. Hastings, and M. A. Moore, Conformal invariance and SLE in two-dimensional Ising spin glasses, *Phys. Rev. Lett.* **97**, 267202 (2006). Archived as `arXiv:cond-mat/0601711`.

[2] R. Bauer, Discrete Loewner evolution. Archived as math.PR/0303119 in arXiv.org.

[3] Bernard,Bauer review

[4] Bernard,Bauer off-critical paper

[5] D. Bernard, G. Boffetta, A. Celani, and G. Falkovich, Conformal invariance in two-dimensional turbulence, *Nature Physics* **2**, 124 (2006). Archived as `arXiv:nlin.CD/0602017`.

[6] D. Bernard, G. Boffetta, A. Celani, and G. Falkovich, Inverse turbulent cascades and conformally invariant curves. Archived as `arXiv:nlin.CD/0609069`.

[7] D. Bernard, P. Le Doussal, and A. A. Middleton, Are domain walls in 2D spin glasses described by stochastic Loewner evolutions?, *Phys. Rev. B* **76**, 020403(R) (2007). Archived as `arXiv:cond-mat/0611433`.

[8] F. Camia and C. M. Newman, Critical percolation exploration path and SLE(6): a proof of convergence, preprint. Archived as `arXiv:math.PR/0604487`.

[9] Cardy review

[10] Chung, Markov chains

[11] T. Kennedy, A fast algorithm for simulating the chordal Schramm-Loewner evolution, *J. Statist. Phys.* **128**, 1125–1137 (2007). Archived as `arXiv:math.PR/0508002`.

[12] T. Kennedy, A faster implementation of the pivot algorithm for self-avoiding walks, *J. Statist. Phys.* **106**, 407–429 (2002). Archived as `arXiv:cond-mat/0109308`

[13] R. Kühnau, Numerische Realisierung konformer Abbildungen durch "Interpolation", *Z. Angew. Math. Mech.* **63**, 631-637 (1983).

[14] G. Lawler, *Conformally Invariant Processes in the Plane*, *Mathematical Surveys and Monographs, vol. 114*, American Mathematical Society, 2005.

[15] G. Lawler, O. Schramm, and W.Werner, On the scaling limit of planar self-avoiding walk, *Fractal Geometry and Applications: a Jubilee of Benoit Mandelbrot, Part 2*, 339–364, *Proc. Sympos. Pure Math. 72*, Amer. Math. Soc., Providence, RI, 2004. Archived as `arXiv:math.PR/0204277`.

[16] G. Lawler, O. Schramm, and W. Werner, Conformal invariance of planar loop-erased random walks and uniform spanning trees, *Ann. Probab.* **32**, 939–995 (2004). Archived as `arXiv:math.PR/0112234`.

[17] N. Madras and G. Slade, *The Self-Avoiding Walk*, Birkhäuser, Boston-Basel-Berlin, 1993.

[18] D. E. Marshall and S. Rohde, Convergence of a variant of the Zipper algorithm for conformal mapping, *SIAM J. Numer. Anal.* to appear.

[19] Nienhuis review

[20] P. Nolin, W. Werner Off critical percolation paper.

[21] O. Schramm, Scaling limits of loop-erased random walks and uniform spanning trees, *Israel J. Math.* **118**, 221–288 (2000). Archived as `arXiv:math.PR/9904022`.

[22] S. Smirnov, Critical percolation in the plane, *C. R. Acad. Sci. Paris Sér. I Math.* **333**, 239 (2001).

[23] S. Smirnov, Towards conformal invariance of 2D lattice models, Proceedings of the International Congress of Mathematicians, Vol. II, 1421-1451, Eur. Math. Soc., Zurich, 2006. Archived as `arXiv:0708.0032v1 [math-ph]`.

[24] J. Tsai, The Loewner driving function of trajectory arcs of quadratic differentials, preprint. Archived as `arXiv:0704.1933v2 [math.CV]`.

[25] D. Zhan, The scaling limits of planar LERW in finitely connected domains, *Ann. Probab.* to appear. Archived as `arXiv:math.PR/0610304`.

[26] S. Rohde, O. Schramm, Basic properties of SLE. Archived as math.PR/0106036 in arXiv.org.

[27] W. Werner, Random planar curves and Schramm-Loewner evolutions To appear in Springer Lecture Notes. Archived as math.PR/0303354 in arXiv.org.

[28] Werner - Utah

[29] Grimmett - book

[30] Convergence of a variant of the Zipper algorithm for conformal mapping, with S. Rohde, SIAM J. Numer. Anal. 45(2007), 2577-2609.

[31] The Loewner differential equation and slit mappings, with S. Rohde, Jour. Amer. Math. Soc., 18(2005), 763-778.

[32] S. Rohde, private communication