

# Simulation of Brownian motion at first-passage times

Zaeem A. Burq and Owen D. Jones  
*Department of Mathematics and Statistics*  
*University of Melbourne*

June 2006

## Abstract

We show how to simulate Brownian motion not on a regular time grid, but on a regular spatial grid. That is, when it first hits points in  $\delta\mathbf{Z}$  for some  $\delta > 0$ . Central to our method is an algorithm for the perfect simulation of  $\tau$ , the first time Brownian motion hits  $\pm 1$ . This work is motivated by boundary hitting problems for time-changed Brownian motion, such as appear in mathematical finance when pricing barrier-options.

**Key words:** boundary hitting; exact simulation; perfect simulation.

## 1 Introduction

Let  $W = \{W(t)\}_{t \geq 0}$  be a standard 1-dimensional Brownian motion;  $W(0) = 0$ . We define level- $\delta$  first-passage times:

$$\begin{aligned}\tau_\delta(0) &= 0; \\ \tau_\delta(n+1) &= \inf\{t > \tau_\delta(n) : |W(t) - W(\tau_\delta(n))| \geq \delta\}.\end{aligned}$$

We call  $(\tau_\delta, X_\delta) = (\{\tau_\delta(n)\}_{n=0}^\infty, \{W(\tau_\delta(n))\}_{n=0}^\infty)$  the  $\delta$ -level skeleton of  $W$ ; see Figure 1 for an example. In this paper we show how to simulate  $(\tau_\delta, X_\delta)$ .

Usually Brownian motion is simulated on a regular time lattice, viz  $\{W(n\epsilon)\}_{n=0}^\infty$ . There is an inherent problem with using a simulation like this, that arises in situations where we wish to know the first time  $W$  crosses a given boundary, namely the sample path from  $W(n\epsilon)$  to  $W((n+1)\epsilon)$  is unbounded. Using the  $\delta$ -level skeleton overcomes this problem.

Our particular motivation comes from mathematical finance, where time-changed Brownian motion is used as a model for financial assets (see [HPR97] for a survey), and boundary hitting problems appear, for example, when pricing barrier-options or estimating the ruin probability of an insurance fund.

The following properties of the  $\delta$ -level skeleton are clear. Let  $Z_\delta(n) = W(\tau_\delta(n)) - W(\tau_\delta(n-1))$  then  $Z_\delta(1), Z_\delta(2), \dots$  are i.i.d. with  $\mathbf{P}(Z_\delta(1) = -\delta) = \mathbf{P}(Z_\delta(1) = \delta) = 1/2$ . From the strong Markov property, the differences  $u_\delta(n) = \tau_\delta(n) - \tau_\delta(n-1)$  are i.i.d. with the same distribution as  $\tau_\delta(1)$ , since  $\tau_\delta(0) = 0$ . Since  $W$  is self-similar with index  $1/2$ , that is since  $W \stackrel{fdd}{=} \{a^{-1/2}W(at)\}_{t \geq 0}$  for any  $a > 0$ , we have

$$(\tau_\delta, X_\delta) \stackrel{fdd}{=} (\delta^2\tau_1, \delta X_1).$$

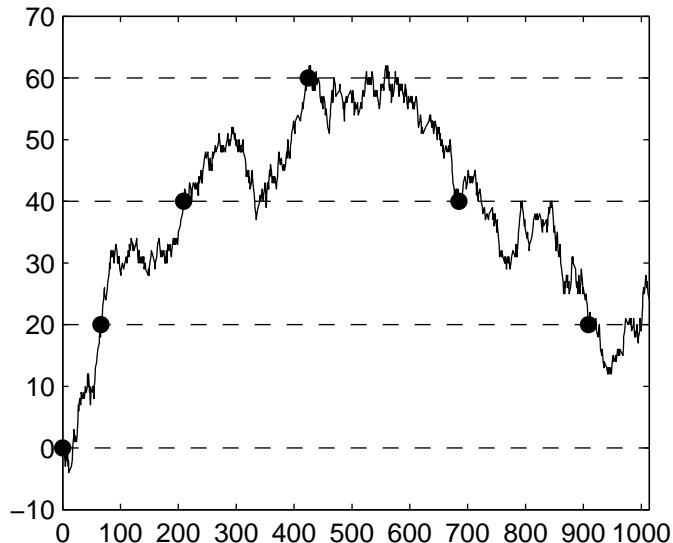


Figure 1: Sample Brownian motion path and  $\delta$ -level skeleton. The marked points are  $(\tau_\delta(n), W(\tau_\delta(n)))$  for  $\delta = 20$  and  $n = 0, 1, \dots$

Thus the problem of simulating  $(\tau_\delta, X_\delta)$  reduces in essence to the problem of simulating  $\tau := \tau_1(1)$ .

It can be shown that  $\tau$  satisfies the recurrence relation [Ha95, Jo96]

$$\tau \stackrel{d}{=} \sum_{i=1}^N \tau^{(i)} / 4, \quad (1)$$

where the  $\tau^{(i)}$  are i.i.d. as  $\tau$  and  $N/2 \sim \text{geometric}(1/2)$  (that is,  $\mathbf{P}(N = 2k) = 1/2^k$  for  $k = 1, 2, \dots$ ). This relation is obtained from the rather remarkable result that  $\tau$  is the normed-limit of a branching process with offspring distribution  $N$ . This result immediately gives rise to an approximate method for simulating  $\tau$ : simulate  $n$  generations of the branching process then divide by  $4^n$ . However this approach is slow and the simulation error can not be bounded. Devroye & Neininger [DN02] give a technique for the perfect simulation of random variables satisfying recurrence relations such as (1). In what follows we derive an algorithm for the perfect simulation of  $\tau$  that is similar in spirit to that of Devroye & Neininger, though simpler, as we are able to make use of additional specific results on the distribution of  $\tau$ . More importantly, we show that the expected number of function calls required to simulate  $\tau$  using our algorithm is finite, unlike the algorithm of [DN02].

In Section 2 we derive some useful results on the distribution of  $\tau$  then in Section 3 we give our simulation algorithm. A `Matlab` implementation of the algorithm is given in the Appendix.

## 2 The first-passage time $\tau$

We consider  $\tau := \tau_1$  where  $\tau_\delta = \inf\{t \geq 0 : |W(t)| \geq \delta\}$ . For the remainder of the paper  $\tau$  and  $\tau_\delta$  will refer to a single first-passage time, rather than a sequence of them.

The Laplace transform of  $\tau$  can easily be calculated by applying the martingale stopping theorem to the exponential martingale  $M_\lambda(t) = \exp\{\lambda W(t) - \lambda^2 t/2\}$  at  $\tau$ :

$$\psi(\lambda) := \mathbf{E} \exp\{-\lambda\tau\} = \frac{1}{\cosh \sqrt{2\lambda}}.$$

Inverting the Laplace transform ([BS02], Formula 1.3.0.2) gives the probability density function of  $\tau$ :

$$f(t) = \sum_{k=-\infty}^{\infty} (-1)^k g_{(1+2k)}(t), \quad t \geq 0, \quad (2)$$

where

$$g_y(t) = \frac{y}{\sqrt{2\pi t^3}} \exp\{-y^2/2t\}, \quad t \geq 0.$$

Differentiating  $\psi$  at 0 we get  $\mathbf{E}\tau = 1$ .

The next three lemmas give a description of the shape of  $f$ .

**Lemma 1** *The distribution  $f$  of  $\tau$  is unimodal.*

**Proof** We use Theorem 1 of [Ya78], which shows that a self-decomposable distribution is necessarily unimodal. A random variable  $X$  is self-decomposable if for any  $c \in (0, 1)$  there exists a  $Y$ , independent of  $X$ , such that  $X \stackrel{d}{=} cX + Y$ .

For any fixed  $c \in (0, 1)$ , we can write

$$\tau_1 = \tau_{\sqrt{c}} + \eta_{\sqrt{c},1},$$

where  $\eta_{\sqrt{c},1} := \inf\{t > 0 : |W(t + \tau_{\sqrt{c}})| \geq 1\}$ .

It follows immediately from (2) that  $\tau_{\sqrt{c}} \stackrel{d}{=} c\tau_1$ .

Let  $\mathcal{F}(T)$  be the  $\sigma$ -algebra generated by  $W$  from time 0 up to the stopping time  $T$ . By the strong Markov property and the symmetry of  $W$ ,

$$\begin{aligned} \mathbf{P}(\eta_{\sqrt{c},1} \leq t | \mathcal{F}(\tau_{\sqrt{c}})) &= \mathbf{P}(\eta_{\sqrt{c},1} \leq t | W(\tau_{\sqrt{c}})) \\ &= \mathbf{P}(|W(s)| \geq 1 \text{ for some } s \leq t | W(0) = \sqrt{c}). \end{aligned}$$

Hence  $\tau_{\sqrt{c}}$  and  $\eta_{\sqrt{c},1}$  are independent. The self-decomposability of  $\tau$  and thus the result follows.

**Lemma 2** *Right tail asymptotics. For any  $\epsilon > 0$  we have, for  $\gamma = \pi^2/8$ ,*

$$f(t) = o(e^{-(\gamma-\epsilon)t}) \text{ as } t \rightarrow \infty.$$

**Proof** First note that

$$\int_0^\infty e^{-\lambda t} df(t) = \lambda \int_0^\infty e^{-\lambda t} f(t) dt = \lambda \psi(\lambda) = \frac{\lambda}{\cosh \sqrt{2\lambda}}.$$

We know that  $\cosh(\theta) = 0$  when  $\theta = i(\frac{\pi}{2} + n\pi)$ , for  $n \in \mathbf{Z}$ . Thus  $\lambda \psi(\lambda)$  has singularities when  $\sqrt{2\lambda} = i(\frac{\pi}{2} + n\pi)$ , for  $n \in \mathbf{Z}$ , or

$$\lambda = -\frac{1}{2} \left( \frac{\pi}{2} + n\pi \right)^2, \quad n \in \mathbf{Z}.$$

The largest pole is  $-\pi^2/8$ , occurring when  $n = -1, 0$ , so  $\lambda\psi(\lambda)$  converges in the half plane

$$\Re(\lambda) > \frac{\pi^2}{8} =: \gamma.$$

The lemma now follows from classical transform theory; see for example Theorem 2.2a of [Wi46].

**Lemma 3** *Left tail asymptotics.* For all  $n \geq 1$

$$f(t) = o(t^n) \text{ as } t \rightarrow 0.$$

**Proof** Taking the positive terms in the expansion (2) of  $f$ , then using the fact that  $e^{-x} \leq m!/x^m$ , for any  $m \geq 0$  and all  $x \in [0, \infty)$ , we have

$$\begin{aligned} f(t) &\leq \sum_{k=-\infty}^{\infty} g_{(1+4k)}(t) \\ &\leq \sum_{k=-\infty}^{\infty} \frac{m! 2^m t^{m-3/2}}{\sqrt{2\pi} (1+4k)^{2m-1}} \\ &= c_m t^{m-3/2}, \end{aligned}$$

where, for  $m \geq 2$ ,

$$c_m = \frac{m! 2^m}{\sqrt{2\pi}} \sum_{k=-\infty}^{\infty} \frac{1}{(1+4k)^{2m-1}} < \infty.$$

As  $m$  can be taken arbitrary large, the result follows.

Define

$$f_n(t) = \sum_{k=-n}^n (-1)^k g_{(1+2k)}(t), \quad t \geq 0.$$

Our final lemma concerns the manner in which  $f_n(t) \rightarrow f(t)$  as  $n \rightarrow \infty$ .

**Lemma 4** *The sequence of remainders  $\epsilon_n(t) = f(t) - f_n(t)$ ,  $n = 1, 2, \dots$ , oscillates about 0 for  $n \geq N(t) := t(\log 3)/4 \vee 3$ .*

**Proof** Let  $h_n(t) = f_n(t) - f_{n-1}(t)$ , then

$$\begin{aligned} h_n(t) &= \frac{(-1)^n}{\sqrt{2\pi t^3}} \left( (1-2n)e^{-(1-2n)^2/(2t)} + (1+2n)e^{-(1+2n)^2/(2t)} \right) \\ &= \frac{(-1)^{n+1}}{\sqrt{2\pi t^3}} e^{-(1+4n^2)/(2t)} \left( (2n-1)e^{2n/t} - (2n+1)e^{-2n/t} \right). \end{aligned}$$

In the last expression, the first factor has an oscillating sign, while the second factor is strictly positive as soon as

$$\frac{2n+1}{2n-1} < e^{4n/t}.$$

For  $n \geq 2$  the LHS of the inequality is strictly bounded above by 3, so we have that  $h_n(t)$  oscillates for

$$n \geq n_0(t) := t(\log 3)/4 \vee 2.$$

Next note that for  $n \geq n_0(t)$ ,

$$|h_n(t)| = \frac{1}{\sqrt{2\pi t^3}} \left( (2n-1)e^{-(2n-1)^2/(2t)} - (2n+1)e^{-(2n+1)^2/(2t)} \right). \quad (3)$$

Thus, for  $n \geq n_0(t)$ ,

$$\begin{aligned} |h_n(t)| > |h_{n+1}(t)| &\iff (2n-1)e^{-(2n-1)^2/(2t)} - (2n+1)e^{-(2n+1)^2/(2t)} \\ &> (2n+1)e^{-(2n+1)^2/(2t)} - (2n+3)e^{-(2n+3)^2/(2t)} \\ &\iff (2n-1)e^{2n/t} > (4n+2)e^{-2n/t} - (2n+3)e^{-(6n+4)/t} \\ &\iff (2n-1)e^{2n/t} > (4n+2)e^{-2n/t} \\ &\iff e^{4n/t} > \frac{4n+2}{2n-1}. \end{aligned}$$

For  $n \geq 3$ , the RHS of the final inequality is strictly bounded above by 3, so we have that  $|h_n(t)|$  is strictly decreasing for

$$n \geq n_1(t) := t(\log 3)/4 \vee 3.$$

For  $n \geq n_1(t)$  we have that  $h_n(t)$  oscillates about 0 and  $|h_n(t)|$  is strictly decreasing. It follows that for  $n \geq n_1(t)$  we must have  $f_{n+k}(t)$  between  $f_n(t)$  and  $f_{n+1}(t)$  for all  $k > 1$ , and thus the limit  $f(t)$  must lie between  $f_n(t)$  and  $f_{n+1}(t)$ . Putting  $N(t) = n_1(t)$  the result follows.

### 3 Simulation algorithm

We use a rejection algorithm with iterative-squeezing to simulate  $\tau$ . Suppose that  $g$  is a density such that  $f(t) \leq ag(t)$  on  $[0, \infty)$ , for some  $a \geq 1$ , and that we can generate random variables from the distribution  $g$ . Using a standard rejection algorithm we generate  $U \sim U(0, 1)$  and  $V \sim g$  until  $f(V) > ag(V)U$ , then return  $V$ . In our case we can not calculate  $f$  exactly, however from Lemma 4 we have, for  $n \geq N(V)$ ,

$$f_n(V) < f_{n+1}(V) < ag(V)U \implies f(V) < ag(V)U \text{ and} \quad (4)$$

$$f_n(V) > f_{n+1}(V) > ag(V)U \implies f(V) > ag(V)U. \quad (5)$$

Since  $f_n(V) \rightarrow f(V)$  as  $n \rightarrow \infty$ , we have that with probability 1, either (4) or (5) occurs eventually. Using this we obtain the following algorithm. It can be proved correct in the same manner as the standard rejection algorithm, see for example [De86].

```

REPEAT
  Generate  $U \sim U(0, 1)$  and  $V \sim g$ 
   $n = N(V)$ 
  Resolved = FALSE
  REPEAT
    IF  $f_n(V) < f_{n+1}(V) < ag(V)U$  THEN
      Resolved = TRUE
      Accept = FALSE
    ELSE IF  $f_n(V) > f_{n+1}(V) > ag(V)U$  THEN
      Resolved = TRUE
      Accept = TRUE

```

```

    END IF
    n = n + 1
UNTIL Resolved
UNTIL Accept
Return V

```

To complete the algorithm we need the sampling distribution  $g$  and constant  $a \geq \sup_{t \geq 0} f(t)/g(t)$ . We used the Gamma distribution. Let

$$g(t; \alpha, \lambda) = \frac{\lambda^\alpha t^{\alpha-1} e^{-\lambda t}}{\Gamma(\alpha)}, \quad t \geq 0.$$

For any  $\epsilon > 0$  we have

$$g(t; \alpha, \lambda) = o(e^{-(\lambda-\epsilon)t}) \text{ as } t \rightarrow \infty.$$

And for the left tail we have

$$g(t; \alpha, \lambda) = O(t^{\alpha-1}) \text{ as } t \rightarrow 0.$$

Setting  $\lambda = \gamma - \epsilon$ , where  $\gamma = \pi^2/8$  from Lemma 2 and  $\epsilon > 0$  is small, we get a Gamma density whose right tail dominates that of  $f$ . Since the left tail of the Gamma density has polynomial decay, it will dominate the left tail of  $f$  for any choice of  $\alpha > 0$  and  $0 < \lambda < \gamma$ , from Lemma 3. Thus, as  $f$  and  $g$  are bounded on  $[0, \infty)$ , for any  $\alpha > 0$  and  $0 < \lambda < \gamma$  there exists an  $a$  such that  $f \leq ag$  on  $[0, \infty)$ .

The time taken by the rejection algorithm is proportional to  $a$  (see below). Numerical minimisation of  $\sup_{t \geq 0} f(t)/g(t; \alpha, \lambda)$  subject to  $\alpha > 0$  and  $0 < \lambda < \gamma$  gives, to 6 decimal places,

$$a = 1.243707 \text{ for } \alpha = 1.088870 \text{ and } \lambda = 1.233701.$$

For these values of  $\alpha$  and  $\lambda$ , the supremum of  $f(t)/g(t; \alpha, \lambda)$  occurred at  $t = 0.52495$  (to 5 decimal places). The optimal  $\lambda$  is tight against the upper constraint  $\gamma$ . Figure 3 gives plots of  $f$  and  $ag$ .

### 3.1 Efficiency

Let  $I_n(t)$  be the interval spanned by two successive approximations  $f_{n-1}(t)$  and  $f_n(t)$  of  $f(t)$ :

$$I_n(t) = (f_{n-1}(t) \wedge f_n(t), f_{n-1}(t) \vee f_n(t)).$$

For a given realisation of the pair  $(U, V)$  where  $U \sim U(0, 1)$  and  $V \sim g$ , let

$$K(U, V) = \min\{n \geq N(V) : ag(V)U \notin I_n(V)\}.$$

$K(U, V)$  is the number of terms in the truncated series expansion  $f_n(V)$  of  $f(V)$  needed by our algorithm to reach the accept/reject decision.

Let  $\{(U_i, V_i)\}$  be an i.i.d. sequence with,  $U_i \sim U(0, 1)$  and  $V_i \sim g$  independent. Let  $\eta = \min\{i : (U_i, V_i) \text{ is accepted}\}$ , then the number of function calls made by our algorithm, in order to simulate  $\tau$ , is proportional to

$$Z = \sum_{i=1}^{\eta} K(U_i, V_i).$$

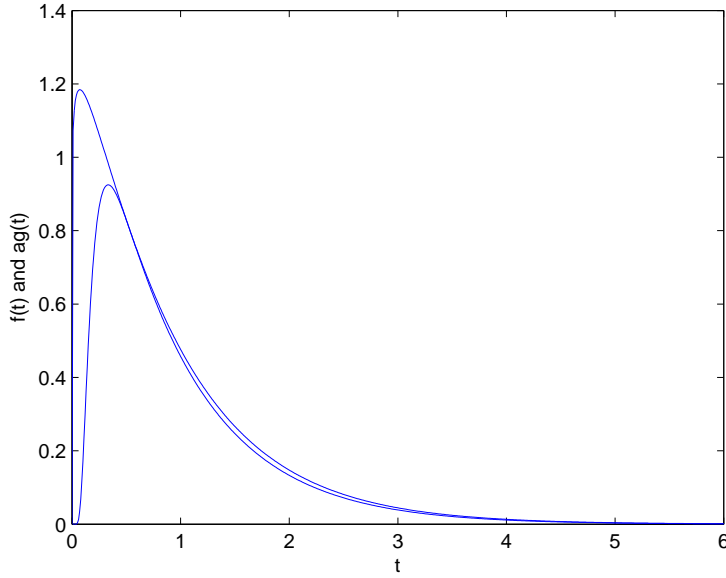


Figure 2: The density  $f(t)$  of  $\tau$  and dominating curve  $ag(t)$  where  $g$  is a  $\Gamma(1.088870, 1.233701)$  density and  $a = 1.243707$ .

By Wald's Equation (see for example [De86] Theorem II.3.5),  $\mathbf{E}Z = \mathbf{E}\eta\mathbf{E}K(U, V)$ . Clearly  $\eta$  has a Geometric distribution with mean  $a$ , so the number of function calls made by our algorithm is finite if and only if  $\mathbf{E}K(U_1, V_1) < \infty$ .

For  $k \geq N(V)$  we have

$$\mathbf{P}(K(U, V) \geq k | V) = \mathbf{P}(ag(V)U \in I_k(V) | V) = \frac{|I_k(V)|}{ag(V)} = \frac{|h_k(V)|}{ag(V)},$$

where, using the notation of Lemma 4,  $h_k(t) = f_k(t) - f_{k-1}(t)$ . Thus

$$\begin{aligned} \mathbf{E}(K(U, V) | V) &= \sum_{k=1}^{\infty} \mathbf{P}(K(U, V) \geq k | V) \\ &\leq N(V) + \sum_{k=N(V)}^{\infty} \frac{|h_k(V)|}{ag(V)} \\ &= N(V) + \frac{(2N(V) - 1)e^{-(2N(V)-1)^2/(2V)}}{ag(V)\sqrt{2\pi V^3}} \\ &\quad \text{(telescoping sum; see (3)).} \end{aligned}$$

For  $V$  large,  $2N(V) - 1 = O(V)$ , so we can find positive constants  $c_1, \dots, c_5$  such that

$$\mathbf{E}(K(U, V) | V) \leq c_1 + c_2V + \frac{(c_3 + c_4V)e^{-c_5(V+1/V)}}{g(V)V^{3/2}},$$

whence

$$\mathbf{E}K(U, V) \leq c_1 + c_2\mathbf{E}V + \int_0^{\infty} \frac{(c_3 + c_4v)e^{-c_5(v+1/v)}}{v^{3/2}} dv < \infty.$$

Noting that the number of operations required to calculate  $f_n(t)$  is proportional to  $n$ , we have proven:

**Proposition 5** *The expected number of operations needed to simulate  $\tau$  is finite.*

## References

- [BS02] Borodin, A. N. and Salminen, P., *Handbook of Brownian Motion: Facts and Formulae*. Birkhauser, 2002.
- [De86] Devroye, L., *Non-Uniform Random Variate Generation*. Springer Verlag, 1986.
- [DN02] Devroye, L. and Neininger, R., Density approximation and exact simulation of random variables that are solutions of fixed-point equations. *Adv. Appl. Probab.*, **34**, pp. 441–468, 2002.
- [Ha95] Hambly, B.M., On constant tail behaviour for the limiting random variable in a supercritical branching process. *J. Appl. Probab.*, **32**, pp. 267–273, 1995.
- [HPR97] Hurst, S.R., Platen, E. and Rachev, S.R., Subordinated Markov models: a comparison. *Fin. Eng. Japanese Markets*, **4**, pp. 97–124, 1997.
- [Jo96] Jones, O.D., A random walk construction of Brownian motion with drift. Research Report 72, Centre for Statistics, The University of Queensland. November 1996
- [Wi46] Widder, D.V., *The Laplace Transform*. Princeton University Press, 1946.
- [Ya78] Yamazato, M., Unimodality of infinitely divisible distributions of class  $L$ . *Ann. Probab.*, **6**, pp. 523–531, 1978.

## Appendix: Matlab code

We give a Matlab implementation of our algorithm.

The function `xing` returns a single simulated observation of  $\tau$ . It uses the functions `gamrnd` and `gampdf` from the Statistics Toolbox. Note that `gamrnd` and `gampdf` use the scale parameter  $\beta = 1/\lambda = 0.810570$ .

The function `BM_xing` returns a time vector `t` and space vector `Wt` such that `Wt(i)` is Brownian motion observed at time `t(i)`. The function takes an optional input `scale`, with default value 1, which gives the spatial resolution of the simulated process.

```
function X = xing
accepted = 0;
while ~accepted
    X = gamrnd(1.088870, 0.810570);
    Y = rand*1.243707*gampdf(X, 1.088870, 0.810570);
    sqrt2piX3 = sqrt(2*pi*X^3);
    N = max([ceil(0.275*X), 3]);
    K = (1+2*[-N:N]);
    fN0 = sum((-1).^[-N:N].*K.*exp(-K.^2./(2*X)))/sqrt2piX3;
    N = N + 1;
    fN1 = fN0 + (-1)^N*((1-2*N)*exp(-(1-2*N)^2/(2*X)) ...
        + (1+2*N)*exp(-(1+2*N)^2/(2*X)))/sqrt2piX3;
    while sign((Y - fN0)*(Y - fN1)) == -1
        fN0 = fN1;
        N = N + 1;
        fN1 = fN0 + (-1)^N*((1-2*N)*exp(-(1-2*N)^2/(2*X)) ...
            + (1+2*N)*exp(-(1+2*N)^2/(2*X)))/sqrt2piX3;
    end
    if Y <= fN1, accepted = 1; end
end

function [t, Wt] = BM_xing(n, scale)
t = zeros(n+1, 1);
for i = 1:n
    t(i+1) = t(i) + xing;
end
Wt = [0; cumsum(sign(rand(n, 1) - 0.5))];
if nargin > 1
    t = scale^2.*t;
    Wt = scale.*Wt;
end
```