Brownian Motion Simulation Project in R

Zhijun Yang
Faculty Adivisor: David Aldous

Historically, Brownian motion is named after the botanist Robert Brown, who discovered it through observing through a microscope at particles found in pollen grains in water, and founded strange patterns of movement of the particles in 1827. However, it is not until 1905, Brownian motion started to emerge into mainsteam science community when Albert Einstein published a paper to fully explain this phenomenona among his other three legendary papers in that year.The first rigorous construction of mathematical Brownian motion is due to Wiener, so Brownian motion is also been called Wiener Process.

Preliminary and Definitions

Defition of Brownian Motion (in lower dimension case)
A real-valued stochastic process $\{B(t) : t \geq 0\}$ is called a (linear) Brownian Motion with started in x $\in \Re$ if the following holds:
(i) B(0) = x
(ii) the process has independent increment, i.e. for all time $0 \leq t_1 \leq t_2 \leq ... \leq t_n$ the increments B(t_n) $-$ B(t_{n-1}), ... ,B(t_2) $-$ B(t_1) are independent random variables.
(iii) for all t $\geq$ 0 and h $>$ 0, the increments $B(t + h) - B(t)$ are normally distributed with expectation zero and variance h
(iv) the function t $\rightarrow$ B(t) is continous
In addition, we say that $\{B(t) : t \geq 0\}$ is a standard Brownian Motion if x=0.

From the definition above, we immediately see that Brownian motion is a continous time Stochastic process, and in fact it is commonly known as among these simplest continous time stochastic process, let's recall that a stochastic preocess is defined as

Definition of Stochastic Process
Let T be a set, (E, Σ) be a measurable space. A stochastic process indexed bt T, taking its values in (E, Σ), is a family of measurable mappings X_t, t $\in$ T, from a probability space (Ω, $\mathscr{F}$, P) into (E ,Σ). The space (E, Σ) is called the state space.

In short, a Stochastic process, whose behavior implied by its another name, Random process, gives a randomly determinant outcome. In general we can just view a stochastic process as just a collection of randome variables $\{X_t : t \in T\}$ evolving in the state space.
Stochastic process can be divided into two categories:
(i) Discrete time and Discrete space
(ii) Continous time and Continous space
Since Brownian motion is Continous time Process, we will focus on some properties of continous time Stochastic Process. Thus we are expecting that Brownian motion as a stochastic process, whose behavior is determined by a collection of random variables varies in time.

In addition, I think some useful concepts that we will use in develop our further studies of Brownian motion simulation and application are Martingale and Markov Chain, which we will briefly introduce here.

I will use the brief introduction of Martingale stated in Chapter 5 of Durrett's *Probability Theory and Examples*: "A martingale X_n can be thought of as the fortune at time n of a player who is betting on a fair game; submartingales (supermartingales) as the outcome of betting on a favorable (unfavorable) game. There are two basic facts about martingales. The first is that you cannot make money betting on them, and in particular if you choose to stop playing at some bounded time N, then your expected winnings EX_N are equal to your initial fortune X_0."
It is from this definition, immediately we are expecting that Brownian motion is a sort of Martingale, whose behavior cannot be determined by past. And it is intuitive to think that the expectation of Brownian motion movement will be its original position.
A more rigorous definition of martingale can be stated as the following:
Definiton of Martingale
A stochastic process X is said to be a Martingale w.r.t a filtration $\mathscr{F}_n$ (a filtration is an increaseing sequence of σ-fields.) if X_n is
(i) E$|X_n| < \infty$
(ii) X_n is adapted to $\mathscr{F}_n$ (**Note :** a sequence X_n is said to adapted to $\mathscr{F}_n$ if $X_n \in \mathscr{F}_n$ for all n.)
(iii) E$(X_{n+1}|\ \mathscr{F}_n)$ =X_n for all n

Now, we are introducing our last concept here, which among other definition made here is closely related to Brownian Motion. A Marknov chain, simply speaking, is a system that transite from one state to another, among with a countable number of possible states.

Definition of Markov Chain

A Markov Chain is a sequence of random variabes $X_1, X_2, X_3, ...$ such that
P$(X_{n+1} = X \mid X_1 = x_1, X_2 = x_2, ..., X_n = x_n\ )$ = P$(X_{n+1} = x \mid X_n = x_n)$
where the x_i are taken value from a countable set S, called the state space of the chain.

Again, immediately from the definition, we can see that markov chain has the memoryless property, that its future state is only determined by its current state. Markov chain has various application in modeling the real world phenomenona. Markov chain are defined by discrete-time and continous-time markov chain, it is a great tool and concept to help us study Brownian motion. For example, markov chain leads to random walk process which gives us insightful information about Brownian motion. In fact, standard Brownian motion (Wiener process) is just scaling limit of random walk in dimension.

Since this research mainly focus on simulation and application of Brownian motion, we will end our introduction of major concepts and definitions here. But we will constantly refer back to those concepts in our discussion later and some relevant consequences and theorems will also be explore later.

Examples and Simple Simulation

After those introduction, let's start with an simple examples of simulation of Brownian Motion produced by me.

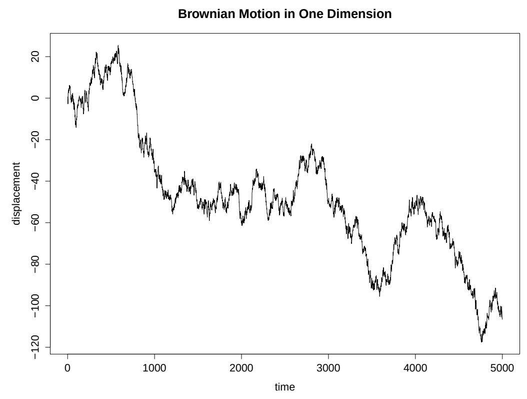
Example of A Simple Simulation of Brownian Motion

Like all the physics and mathematical problem, we first consider the simple case in one dimension. We see from (ii), (iii) of definition of Brownian motion. Brownian motion in one dimension is composed of cumulated sumummation of a sequence of normally distributed random displacements, that is Brownian motion can be simulated by successive adding terms of random normal distribute numbernamely:

$$\text{X}(0) \backsim \text{N}(0,\sigma^2)$$
$$\text{X}(1) \backsim \text{X}(0) + \text{N}(0,\sigma^2)$$
$$\text{X}(2) \backsim \text{X}(1) + \text{N}(0, \sigma^2)$$
$$.......$$

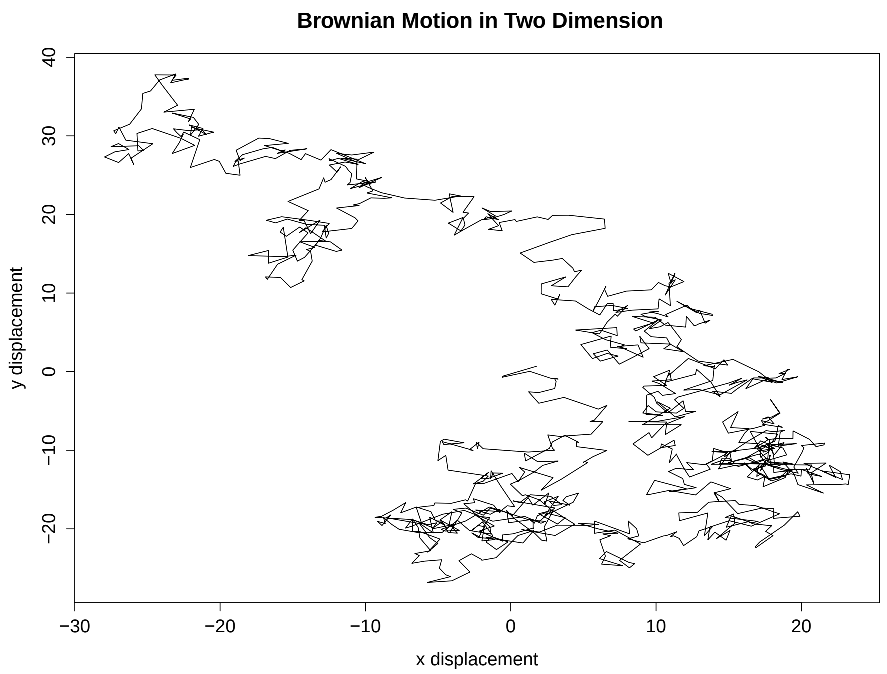
where N denotes normal distribution, and each N are independent, where in a simple Brownian motion case $\sigma = 0$.
We define time as a number of discrete interval, as the length of each interval become arbitrary small, the

process become continous in nature. Let's call the N as the number of discrete time interval. And I wrote code and simulated the graph when taken N = 1000, 5000 in R statistical programming language see appendix for coding.

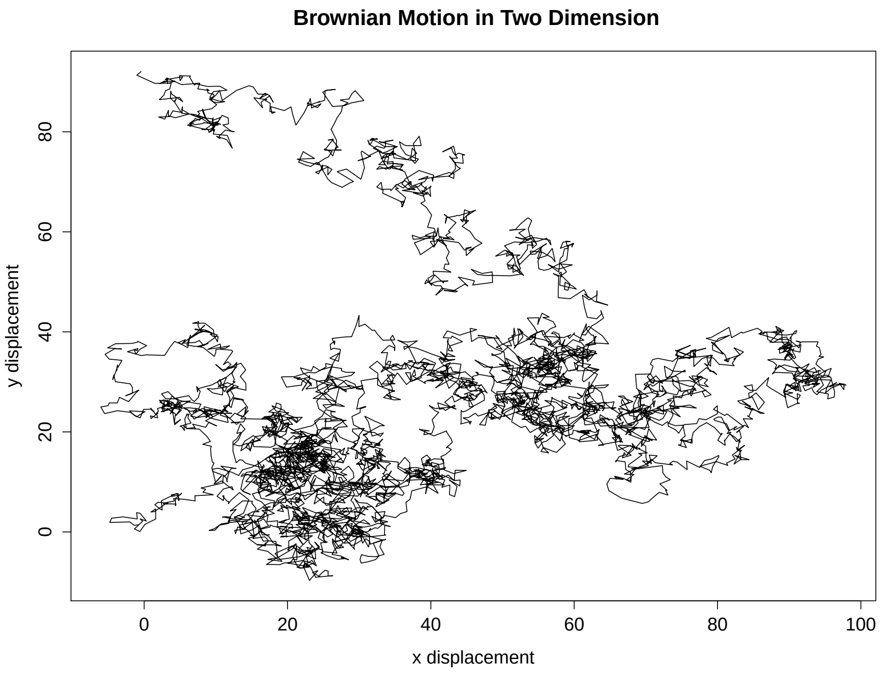


We see that as N become larger, then "motion" produced by our code become larger in scale. And when N is greater than 1000, the produced graph has not much essential difference. But when we take N small, such as less than 500, the graph does not actually quiet like Brownian motion in real physical situation.

We now come to second step of what usually being teach at a math or physcis class, ascending our situation into higher dimensions. One of the nature thing that come to my mind is that add more variables into our code to produce our simulation. And indeed, this is true. To simulate a Brownian motion in 2 dimesnion, it is just add a independent copy of displacement variable of one-dimension into our code.



Here is another graphical simulation with N=5000, we can see there are not essential differnce from the case N=1000. But However, you can try yourself, that with N less than 500, the graph does not quiet look like the actually Brownian motion in real physical situation.



Again, from the graph I have produced, I am convinced that the bahavior of Brownian motion is not quiet predictable. It future state is only depends on the present state. There are no such trend or patterns we can actually extract from its past behavior. And this fact is actually predicted by its properties in definition.

If we consider Brownian motion as a real physical phenomonena, we immediately notice that the change of movement of particle is random distributed in certain time interval. One of the most relevant distribution for this kind come into my mind is Poisson distribution. Simply speaking, a physical particle traveling thorugh a certain region, at time t, the probability that it been hited by another micro-paritcle follows a poisson distribution, and that is also the moment its trajectory change. Recall that a poisson distribution Poission(λ) is given by:

Definition of Poisson Distribution

A discrete random variable X is said to be as Poission(λ) if its distribution mass function is given by:

$$\mathrm{P(X = k)} = \frac{\lambda^k e^{-\lambda}}{k!}$$

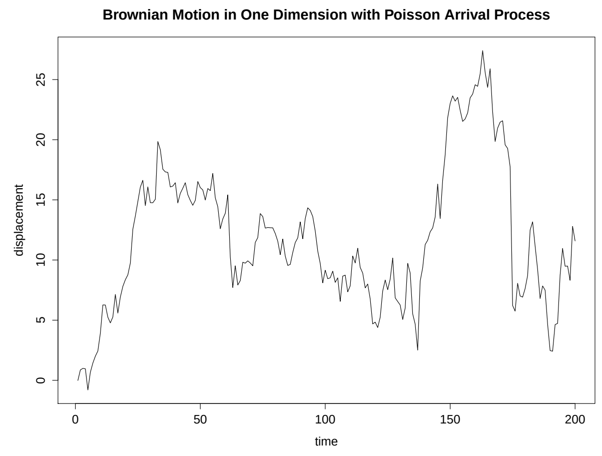
Notice immediately from the definition we see that Poisson distribution is a Lẽvy process, that is its increment at time t1 is independent of any of its increment at time t2, where t1 $\neq$ t2. And in addition it is a stationary precess. Thus it has the propery that its probability that an event happens at t = k1 is same as t = k2. That is a poisson distribution of future event is independent of its past event. That is the exactly the property that we want to employ into our simulation. Specifically, we want to model a Poisson arrival process of displacement incrementation thorughout our time interval.

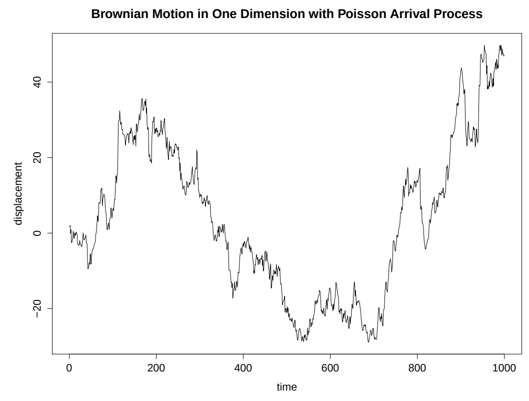
Simulation of Brownian Motion with Displacement

The following are my coding simulation of Brownian motion implemented poisson arrival process suing R staistiscal language. I will briefly explain my code after the simulation.
let's start to explore the simulation in one-dimension and let us taken the parameter λ in the poisson process to be 1.

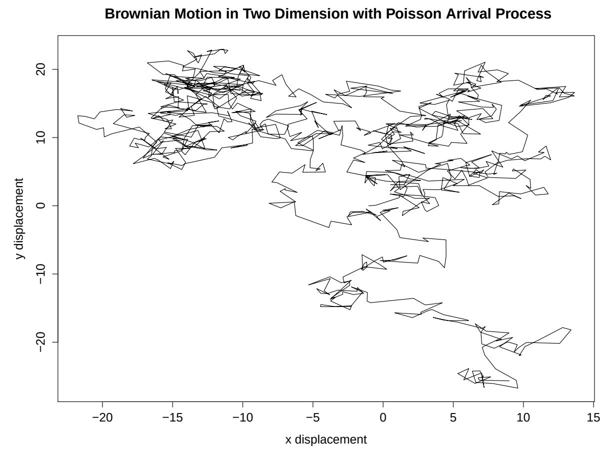
Here we produce a vector (sequence) of poisson random variables *at*, which is consisting of different time interval that a particle travels before change its trajectory. If the value of ith entry of the vector is 0, this means the trajectory change instantly, and if it is any number greater than 0, it means the particle travels for such number of time before its trajectory change, thus we multiple the time interval with the ith entry of random normal vector to represent that the particle travel at speed for such time length, which is its displacement for that time interval. And the graph produced is quiet amazing.

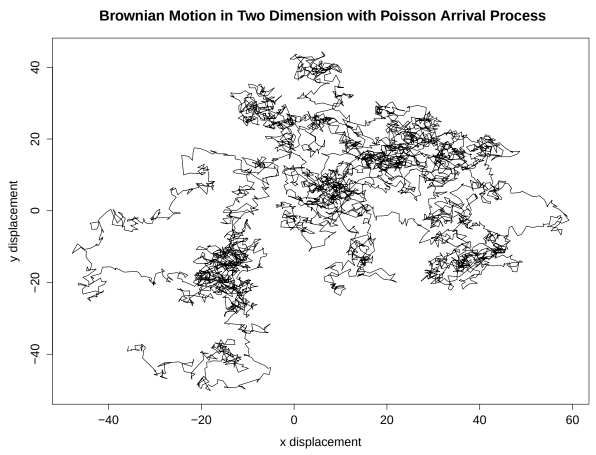
The following graphics represent we take N=200 and N=1000. As you can check that when N is greater than 1000, there are not much difference between graph been produced, and when N is less than 200, the graph does not look quiet like the real situation.



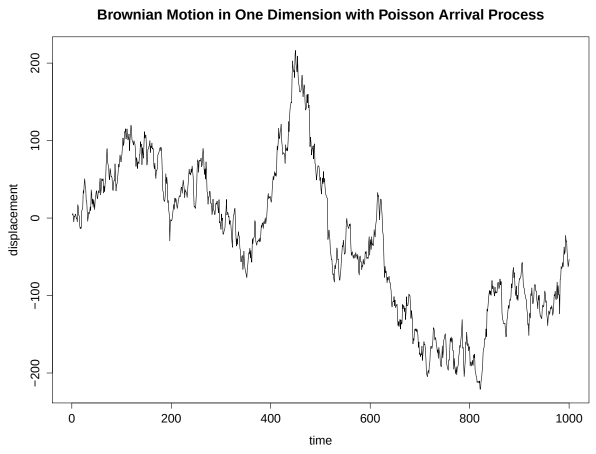


And like before, we are also goint to simulate this in two-dimension case. I adjust our situation to 2 dimension with $\lambda = 1$, and when taken N = 1000 and N = 5000.



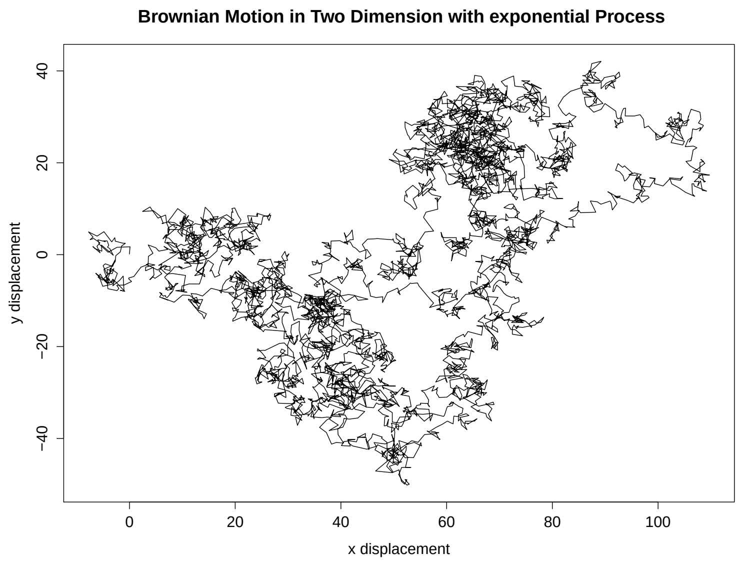


Another thing we can explore is that we can change the value of parameter λ. From the knowledge we have about Poission distribution we know that λ is also the expectation of Poission(λ) distribution, thus by choosing a relative large λ, we get a relative large time inverval between the micro-particel hit our particular pariticular, this means the graph might be more "smooth out". The following are the graph when we take our λ to be 10 instead of 1 and when N is taken the same as 1000:



It might be not obvious at first sight, but notice the scale of the displacement axis (the y-axis). You will find that it is [-200,200] instead of [-20,40]. The realtive large scale verify our intuition is correct. That the change of poisson paremeter λ scale the graph.

In addition, Instead of just implementing Poisson distribution into our implementation, we also are quiet expecting the exponential distribution of time interval bewteen micro-particle hit our particle, this have some real application in phycics since many paritcle are emitted from particluar source follows directly from a exponential distribution. And in fact, expenential distribution is closely related to possion distribution. Since this is introduced in most probabilty textbook, we are not going to explore more about this except leaving a two-dimension Brownian motion trajectory with exponential arrival process. In fact, we can implement any distribution model here for time arriving process into our model here to give different model of Brownian motion based on diferent physics situation.
The following code will produce a Brownian motion trajectory in two dimension with exponential arriving process.

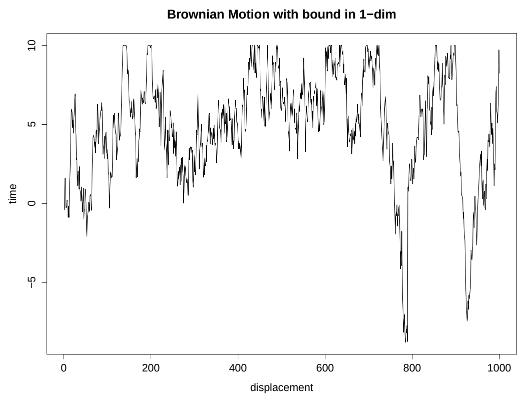


Actually, according to the Theory, the more realistic situation is that we replace the type of distribution of arrival process with mean squared displacement of the particle, which is proportion to the time interval, this quantity can be computed by $< |\vec{r}(t+r) - \vec{r}(t)|^2 > = 2\text{dD}\tau$, where d,D,$\tau$ are corresponding to number of dimensions, diffusion coefficient and time interval. If we replace $\sqrt{2dD\tau}$ into our *at* vector, it would be quiet realistic for any diffusion situation. However, this approach has been extensively researched, so we will not give a more detail discussion about it.

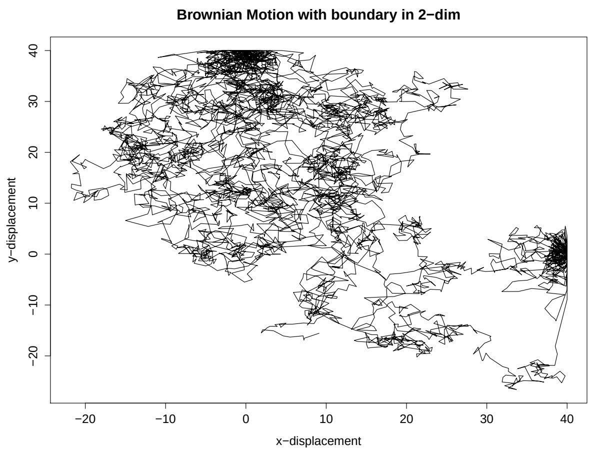
Sometimes, in a real physics and statisitcal situation, we want to our solution be bounded. To take this into real world situation, that is certain range is given, and any movement that try to exit this range is not allowed. And in our case deal with Brownian motion, we can adjust this into our code and thus give our Brownian Motion trajectory a "bound".

Simulation of Brownian Motion with Boundary

The following code define an upper bound and a lower bound for the Brownian motion, namely Upperbound = 10 and Lowerbound=-10, without the bound, you can try yourself that the dispacement will usually goes beyond 20 or -20. Under this condition, our model of Brownian motion will only stayed in the range.



We see from above graph that the Brownian motion did hit the boundary several times (at upperbound displacement = 10 and lowerbound displacement = -10), and been "bounced back" or stayed at the boundary for a while. it is more realistic than our first simulation. And this case is even much more realsitic in the two dimensional case, where a particle is fthought to be flowing thorugh a bounded region. The following graph are obtained when we have taken N=5000 and upper/lower bound of both x,y axis to be 40 and -40.



Appendix: Coding

1. Simple Brownian Motion Code in One Dimension *N = 1000;*
dis = rnorm(N, 0, 1);
dis = cumsum(dis);
plot(dis, type= "l",main= "Brownian Motion in One Dimension", xlab="time", ylab="displacement")

2. Simple Brownian Motion Code in Two Dimension *N = 1000;*
xdis = rnorm(N, 0 ,1);
ydis = rnorm(N, 0 ,1);
xdis = cumsum(xdis);
ydis = cumsum(ydis);
plot(xdis, ydis, type="l", main =" Brownian Motion in Two Dimension", xlab="x displacement", ylab = "y displacement");

3.Brownian Motion with Displacement Code in One Dimension *N = 1000;*
dis = rnorm(N, 0, 1);
at = rpois(N,1)
for(i in 1:N){
if(at[i] != 0){
*dis[i]= dis[i]*at[i];*
} }
dis = cumsum(dis)
plot(dis, type= "l",main= "Brownian Motion in One Dimension with Poisson Arrival Process", xlab="time", ylab="displacement")

4. Brownian Motion With Displacement Code in Two Dimension *N = 1000;*

```
xdis = rnorm(N, 0 ,1);
ydis = rnorm(N, 0 ,1);
xdis = cumsum(xdis);
ydis = cumsum(ydis);
at = rpois(N,1)
for(i in N)
if(rpois[i] != 0)
xdis[i] = xdis[i]*at[i];
ydis[i] = ydis[i]*at[i];

plot(xdis, ydis, type=”l”, main =” Brownian Motion in Two Dimension with Poisson Ar-
rival Process”, xlab=”x displacement”, ylab = ”y displacement”);
```

5. Brownian Motion with Exponenetial Arrive Displacement N = 1000;

```
ub = 10; lb = -10;
xdis = rnorm(N, 0 ,1);
xdis1 = rep(1,N);
xdis1[1] = xdis[1];
for(i in 1:N){
if(xdis1[i] + xdis[i+1] ¿ ub){
xdis1[i+1] = ub;
else
if(xdis1[i] + xdis[i+1] ¡ lb){
xdis[i+1] = lb;
}else{
xdis1[i+1] = xdis1[i] +xdis[i+1];
}}}
plot(xdis1, type=”l”,main=”Brownian Motion with bound in 1-dim”, xlab=”displacement”,ylab=”time”)
```

6.Brownian Motion with Boundary Coding N = 1500;

```
ub = 10; lb = -10;
xdis = rnorm(N, 0 ,1); ydis = rnorm(N, 0, 1);
xdis1 = rep(1,N); ydis1 = rep (1,N);
xdis1[1] = xdis[1]; ydis1[1] = ydis[1];
for(i in 1:N){
if(xdis1[i] + xdis[i+1] ¿ ub){
xdis1[i+1] = ub;
ydis1[i+1] = (ydis1[i] + ydis[i+1])/2;
}else{
if(xdis1[i] + xdis[i+1] ¡ lb){
xdis1[i+1] = lb;
ydis1[i+1] = (ydis1[i] + ydis[i+1])/2;
}else{
if( ydis1[i] + ydis[i+1] ¿ ub){
ydis1[i+1] = ub;
xdis1[i+1] = (xdis1[i] + xdis[i+1])/2;
}else{if(ydis1[i] + ydis[i + 1] < lb){
ydis[i+1] = lb;
xdis1[i+1] =(xdis1[i] + xdis[i+1])/2;
}else{
xdis1[i+1] = xdis1[i] + xdis[i+1];
ydis1[i+1] = ydis1[i] + ydis[i+1];
}}}}}
plot(xdis1, ydis1, type=”l”,main=”Brownian Motion with boundary in 2-dim”, xlab=”displacement”,ylab=”
```

7. Brownian Motion with Boundary Code

```
N = 1000;
xdis = rnorm(N, 0 ,1);
ydis = rnorm(N, 0 ,1);
xdis = cumsum(xdis);
ydis = cumsum(ydis);
at = rexp(N,1)      Here we define exponential or other type of distribution here to give a
new arriving model for Brownian motion
for(i in N)
if(rpois[i] != 0)
xdis[i] = xdis[i]*at[i];
ydis[i] = ydis[i]*at[i];

plot(xdis, ydis, type=”l”, main =” Brownian Motion in Two Dimension with Poisson Ar-
rival Process”, xlab=”x displacement”, ylab = ”y displacement”);
```

Reference

”Investigations on the Theory of Brownian Movement” Einstein, A. New York: Dover, 1956.

Brownian Motion, Peter Morter and Yuval Peres, Cambridge Press, 2010.

Brownian Motion and Stochastic Calculus, Ioannis Karatzas and Steven E. Shreve, Second Edition, Springer Verlag Press, 1991.

An introduction to Stochastic Modeling, Howard M.Taylor and Samuel Karlin, Third Edition, Academic Press, 1998.

Continous Martingales and Brownian Motion, Daniel Revuz and Marc Yor, Third Edition, Springer Press, 1999.

Simple Branching Process Simualtion in Random Walk

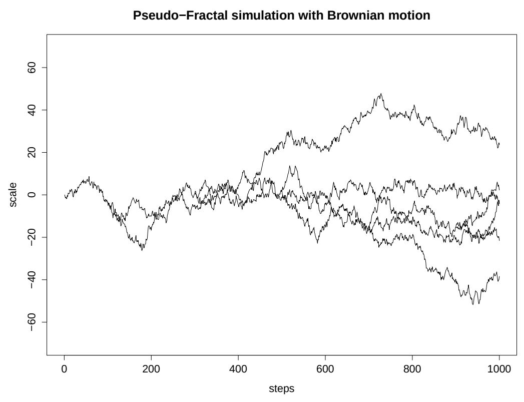
Zhijun Yang
Faculty Adivisor: David Aldous

A Simple Branch Process in Brownian Motion

Here we are going to write a simplication of this branching process for start. Let's consider the case in one-dimension, the simulation starts from a trajectory of Brownian Motion, where I am going to make indifference copies of the some process in smaller scales on the same graph.

In the following code, I take the number steps to be 1, that I am just going to take only one smaller copies of my Wiener (Brownian Motion) process for simplicity. And the number of smaller copies on the our interval is 5. And for simplicaity and consistency, we are dealing with Simple Brownian Motion (Given the variance of norm distribution to be 1) for all our graphs. The code is given below

```
N = 1000;
dis = rnorm(N, 0 ,1);
dis = cumsum(dis);
plot(dis, type = "l", main="Pseudo-fractal simulation with Brownian motion",xlim=c(0,1000), ylim=c(-
70,70))
fnorm = matrix(nrow= 5, ncol = N)
for(i in 1:5){
for(j in 1:N){
fnorm[i,j] = rnorm(1,0,1)
}}
node = c(100,300,500,700,900);
fdis = matrix(nrow= 5, ncol = N)
for(i in 1:5){
fdis[i,] = cumsum(fnorm[i,]) + dis[node[i]]
}
for(i in 1:5){
lines(node[i]:N,fdis[i,1:(N-node[i]+1)])
}
```



An Iterative Branching Process Simulation

Now we will modify our code to produce a better simulation of the "Branching Process" of Brownian Motion.

First again, consider thecode of BM, BMF function, I will add one more input called variance that controls the variation of internal brownian motion movement.

```
BM = function(N,var){
cumsum(rnorm(N, 0 ,var));
}
BMF = function(start, end, total,var){
N = end - start + 1;
dis = rep(NA, total);
replace = BM(N,var)
for(i in start:end){
dis[i] = replace[i + 1 - start];
}
return(dis)
}
```

In the following code we give the original trajectory with Simple Brownian Motion process.

```
total = 1000;
xdis = BMF(1,1000,total,1)
plot(xdis, type = "l", main="Pseudo-Fractal Simulation with Brownian motion with 2 iteration", ylim =
c(-100, 100), xlab = "steps",ylab="scale", lwd= 1.4)
```

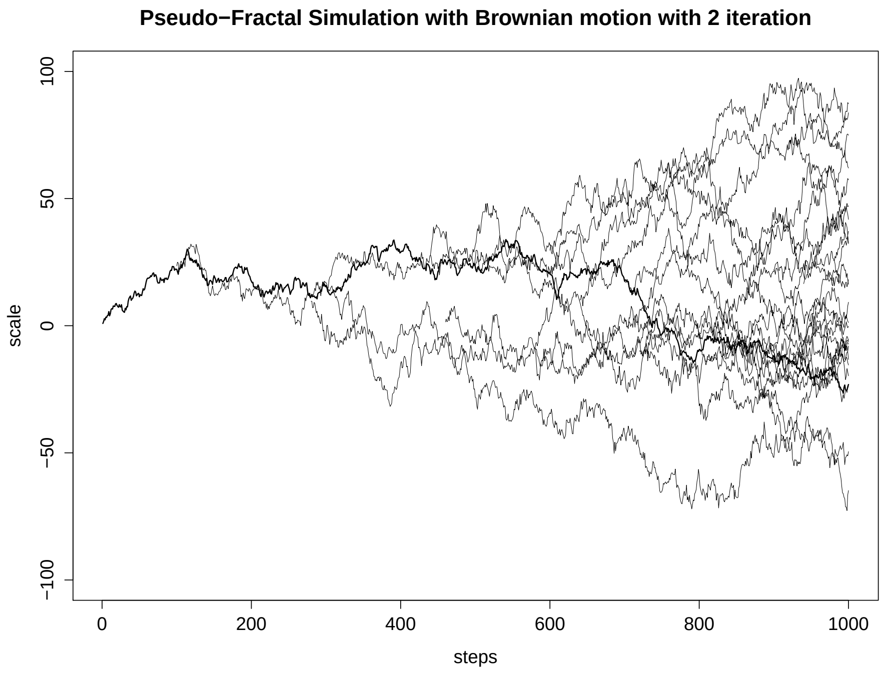
Now for the first iteration of fractal, we give these trajectory with Brownian motion of variance 1.5 instead of 1, and give the branches less thickness than the original branch (stem). Since it is expected that the "offspring" employs more variation than the "parent".

```
N = 5
fxdis1 = matrix(nrow=5, ncol=total);
node = c(100,300,500,700,900);

for(i in 1:N){
n = node[i];
xreplace = BMF(n,total,total,1.5);
yreplace = BMF(n,total,total,1.5);
fxdis1[i,] = (xreplace + xdis[n]);
}

for(i in 1:N){
lines(fxdis1[i,], lwd = 0.5);
}
```

The same idea applys to the second iteration of our coding, we gives the Brownian Motion with variance 2, and also less thickness than the original and first iteration.

```
node1 = 1:N
fxdis2 = rep(NA,total)
for(j in 1:N){
for(i in 1:N){
node1[i] = node[j] + i*(total-node[j])/N;
replace = BMF(node1[i],total,total,2);
n = node1[i];
fxdis2 = replace + fxdis1[i,n];
lines(fxdis2,lwd = 0.2);
}}
```

And the graph produced is given by below, as we see it is a much better simulation plot than before. The branches farther away from the stems have more variation and less thickness.

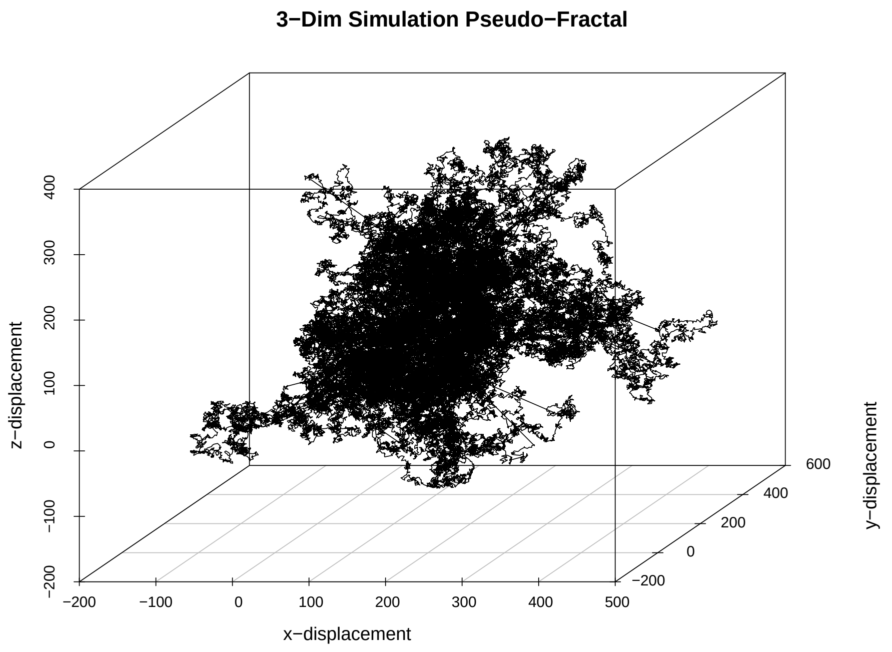


Simulation of Branching Process of Random Walk in three-dimension

The given code is written in R to produce a simulation of the same type but in the three dimension, I added the package "3d scatterlot" to produce the three dimensional plot in R. Anythingelse is just straight forward, just add one more varibales in our computing, then simulating the result.

```
BM = function(N, var){
```

```
cumsum(rnorm(N, 0 ,var));
}
total = 2000; WE TAKE THE VARIACNE OF WEINER PROCESS TO BE LARGE IN STEM
xdis = BM(total,5);
ydis = BM(total,5);
zdis = BM(total,5);
N = 20 WE TAKE THE STEM TO HAVE 20 BRANCHES AND VARIANCE TO BE LESS
fxdis1 = rep(0,N*total);
fydis1 = rep(0,N*total);
fzdis1 = rep(0,N*total);
for(i in 1:(N-1)){
xreplace = BM(total,2);
yreplace = BM(total,2);
zreplace = BM(total,2);
fxdis1[((i-1)*total+1):(i*total)] = (xreplace + xdis[(i-1)*(total/N)+1]);
fydis1[((i-1)*total+1):(i*total)] = (yreplace + ydis[(i-1)*(total/N)+1]);
fzdis1[((i-1)*total+1):(i*total)] = (zreplace + zdis[(i-1)*(total/N)+1]);
}
fxdis1 = c(xdis,fxdis1);
fydis1 = c(ydis,fydis1);
fzdis1 = c(xdis,fzdis1);
library(scatterplot3d)
scatterplot3d(fxdis1,fydis1,fzdis1,type="l", main="3-Dim Simulation Pseudo-Fractal",xlab="x-displacement",ylab="y-
displacement",zlab="z-displacement")
```



Reference

Reversible Markov Chains and Random Walks on Graphs, David Aldous and Jim Fill.

Brownian Motion, Peter Morter and Yuval Peres, Cambridge Press, 2010.

Brownian Motion and Stochastic Calculus, Ioannis Karatzas and Steven E. Shreve, Second Edition, Springer Verlag Press, 1991.

Continous Martingales and Brownian Motion, Daniel Revuz and Marc Yor, Third Edition, Springer Press, 1999.

Random walk in random and non-random environments (Third Edition), Pal Rvsz (2013), World Scientific Pub Co. ISBN 978-981-4447-50-8