Brownian Motion

Brownian Motion Context

We say that a stochastic process $(W_t)_{t \geq 0}$ is a standard Brownian Motion if:

- $W_0 = 0$
- $t \to W_t$ are continuous
- The increments are independent: $\forall 0 \leq s \leq t, W_t - W_s$ is independent of W_s
  - same with $t_1 \leq t_2 \leq \ldots \leq t_k, (W_{t_i+x} - W_{t+i})_{x \in [1,k]}$ are independent
- The law of $W_t - W_s$ is $\mathcal{N}(0, t-s)$.

Simulating Brownian Motions

GOAL

We are interested in implementing a Brownian Motion process model using two different algorithms.

METHOD

Simulation method 1:

We simulate a Brownian Motion trajectory by:

1. Introducing a time length n and a time step t_i, $\forall i [0, n]$ such that $t_i = \eta . i$ with η a fixed, discretization timestep
2. We set $W_0 = 0$
3. We set $W_{t_1} = \sqrt{t_1} . \mathcal{N}(0, 1)$
4. Then, $\forall i \in [2, \frac{T_{max}}{\eta}]$, $W_{t_i} = W_{t_i - 1} + \sqrt{t_i - t_{i-1}} . \mathcal{N}(0, 1)$

Simulation method 2:

We can also rely on Donsker’s Theorem (https://en.wikipedia.org/wiki/Donsker%27s_theorem):

1. Let $X_1, X_2, \ldots$ be a sequence of IID random variable with mean 0 and variance 1 (in our case, $\forall i \in \{1, 2, \ldots\}$, $X_i \in \{-1, 1\}$ and $P(X_i = 1) = P(X_i = -1) = \frac{1}{2}$).
2. Let $n \in \mathbb{N}$, $S_n = \sum^n{}_{i=1} X_i$ be the stochastic process known as a random walk
3. Let the diffusively rescaled random walk (partial sum process) $W^{(n)}(t)$ such that:

$$W^{(n)}(t) = \frac{1}{\sqrt{n}} . S_{\lfloor nt \rfloor}, \, t \in [0, 1]$$

Function implementations:

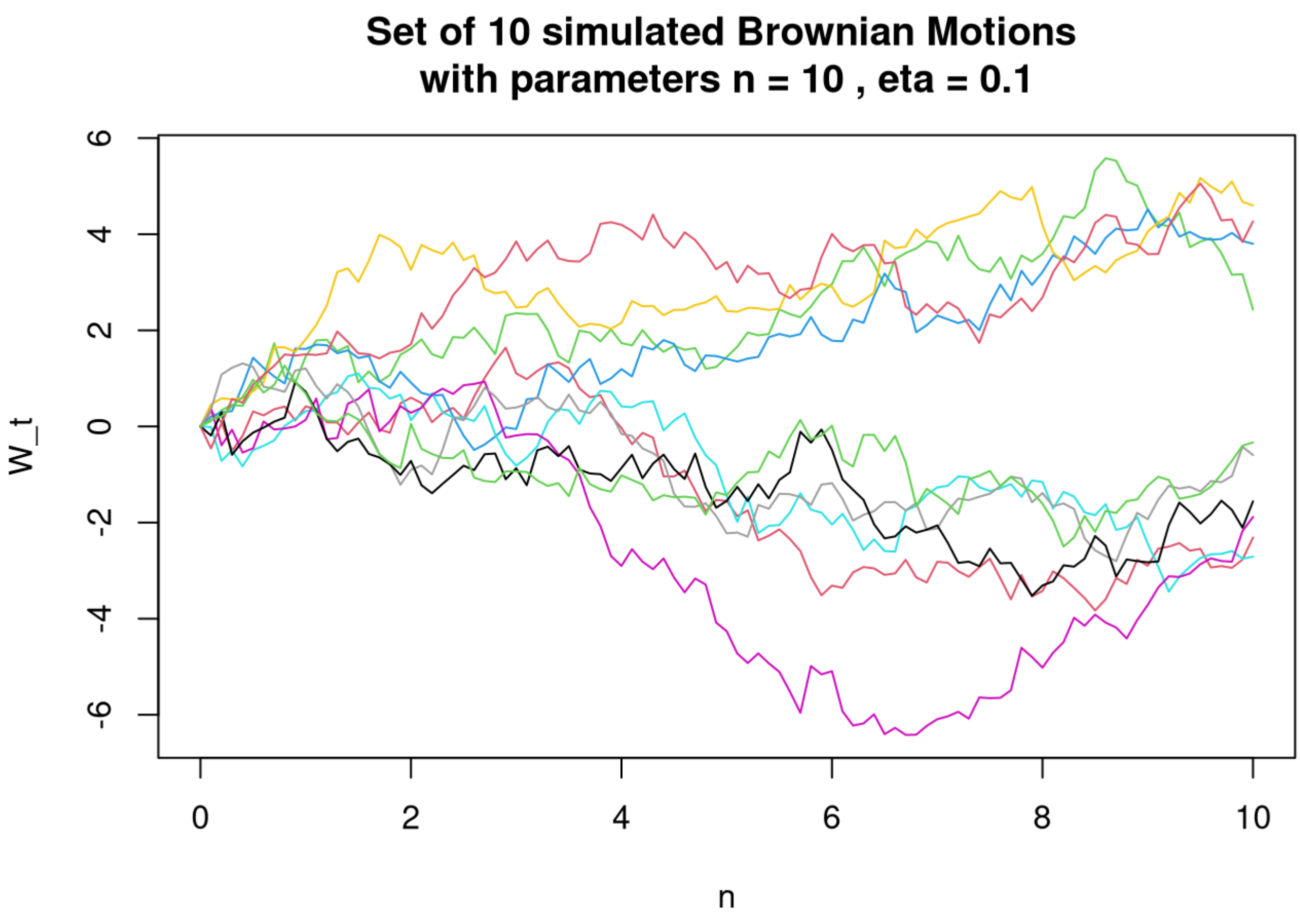
```
standard_simulation <- function(n, eta, graph=F) {
  ### Simulate a brownian motion using an iterative process
  timesteps = seq(0, n, eta)
  # Initializes the brownian motion
  w0 = 0
  w1 = sqrt(eta) * rnorm(1)
  brownian_motion = c(w0, w1)
  # Populates the brownian motion
  for (i in timesteps[3:length(timesteps)]) { # /!\ R is 1-valued
    new_step = brownian_motion[length(brownian_motion)] + sqrt(eta) * rnorm(1)
    brownian_motion = c(brownian_motion, new_step)
  }
  if (graph) {
    plot(timesteps, brownian_motion, type="l",
         main="Simulated browian motion",
         xlab="n", ylab="W_t")
  }
  brownian_motion
}

donsker_simulation <- function(n, eta, graph=F) {
  ### Simulate a brownian motion based on donsker's theorem
  timesteps = seq(0, 1, eta)
  # Computes the steps of a donsker-based brownian motion
  X = sample(c(-1, 1), n, T, prob = c(1/2, 1/2))
  S = cumsum(X)
  W = unlist(apply(matrix(timesteps), 1, function(x) {1/sqrt(n)*S[round(n*x)]}))
  if (graph) {
    par(mar=c(2.5,2.5,2.5,2.5)) # deals with margin error
    plot(timesteps, c(0, W), type="l",
         main="Simulated browian motion via Donsker's Theorem",
         xlab="n", ylab="W_t")
  }
  c(0, W)
}

simulate <- function (simulation_function, n_simulations, time_length, timestep, dons
ker_n=NULL) {
  ### Simulate a set of brownian motions and display the results given a
  ### simulation method.
  time_range = seq(0, time_length, timestep)
  simulations = c()
  # Computes the simulations
  for (sim in 1:n_simulations) {
    if (simulation_function == "standard") {
      s = standard_simulation(time_length, timestep)
      title = "simulated Brownian Motions"
    } else {
      if (is.null(donsker_n)) {
        donsker_n = time_length/timestep
      }
```

```
      s = donsker_simulation(donsker_n, timestep)
      title = "Brownian Motions (simulated via Donsker's Theorem)"
    }
    simulations = c(simulations, s)
  }
  simulations = t(matrix(simulations, ncol=n_simulations))
  # Plots
  ylim = c(min(simulations), max(simulations))
  plot(time_range, simulations[1,], col=2, type="l", ylim=ylim,
       xlab="n", ylab="W_t",
       main=paste("Set of", n_simulations, title,
                  "\nwith parameters n =", time_length, ", eta =", timestep))
  for (sim in 2:n_simulations) {
    lines(time_range, simulations[sim,], col=sim+1)
  }
  simulations
}
```

RESULTS - Simulation with Algorithm 1

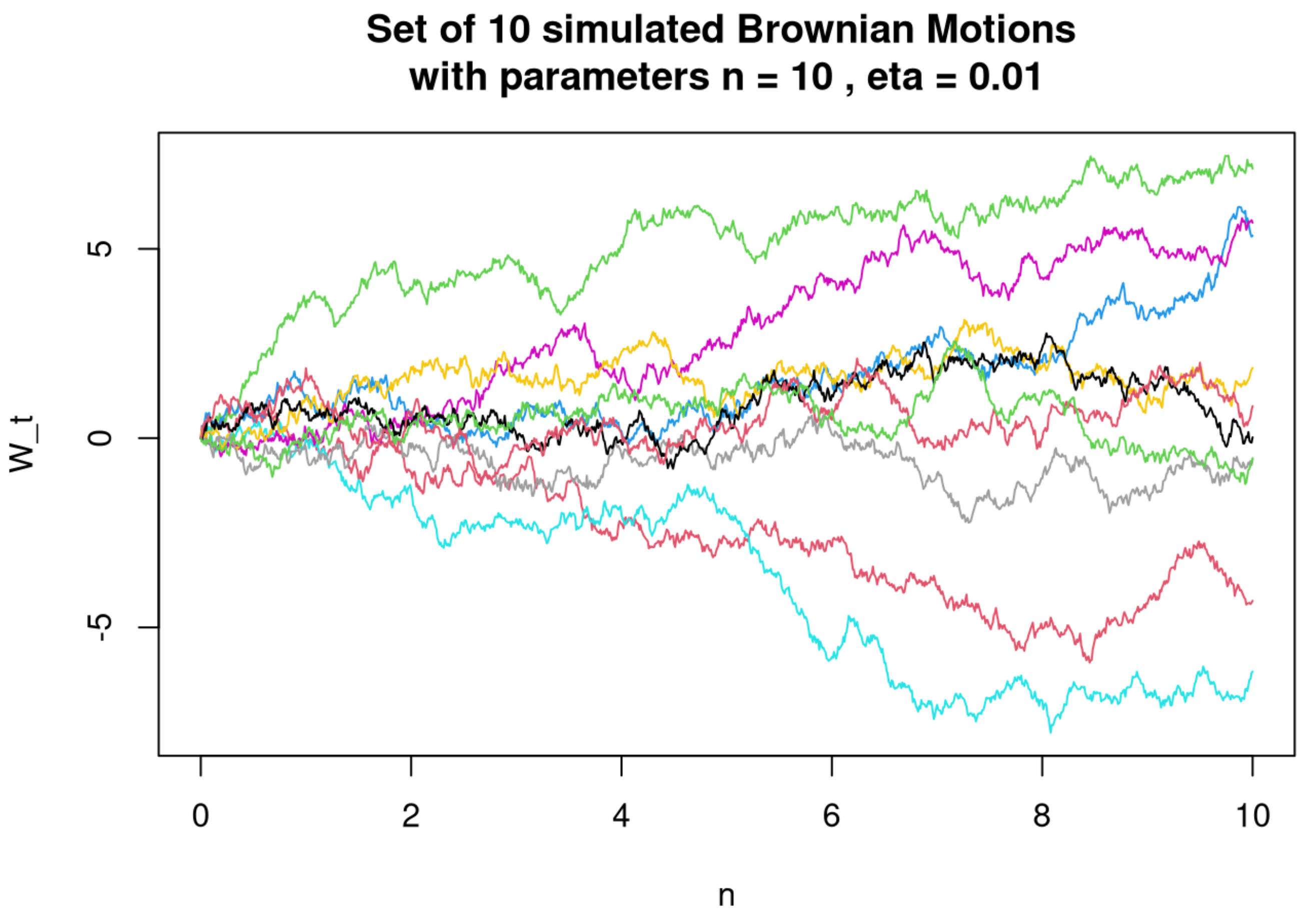
We generate 10 brownian motions with parameters $n = 10$ and $\eta = 0.1$.

```
algo1_simulations = simulate("standard", 10, 10, 0.1)
```



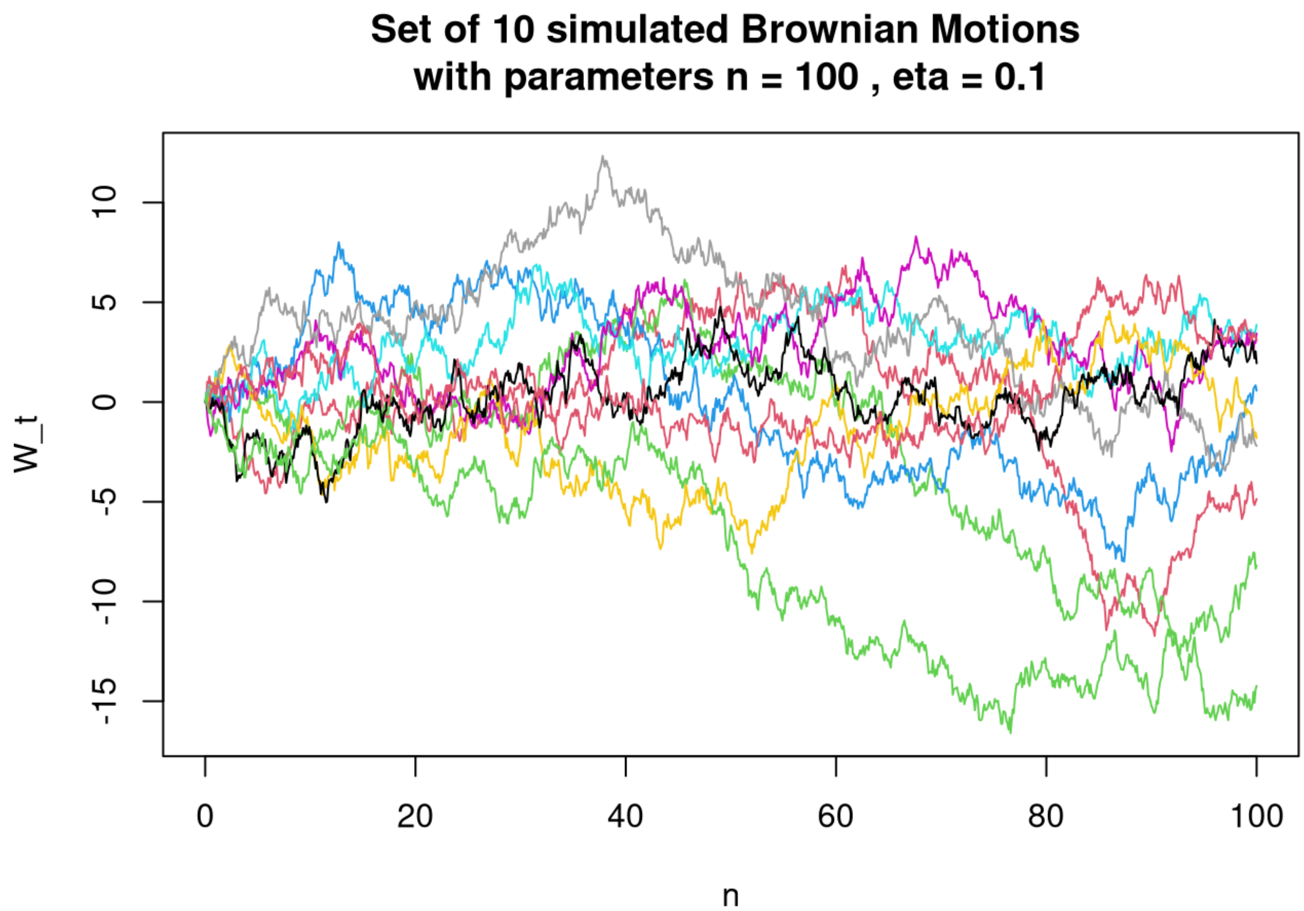
We generate 10 brownian motions with parameters $n = 10$ and $\eta = 0.01$.

```
algo1_simulations = simulate("standard", 10, 10, 0.01)
```



We generate 10 brownian motions with parameters $n = 100$ and $\eta = 0.1$.

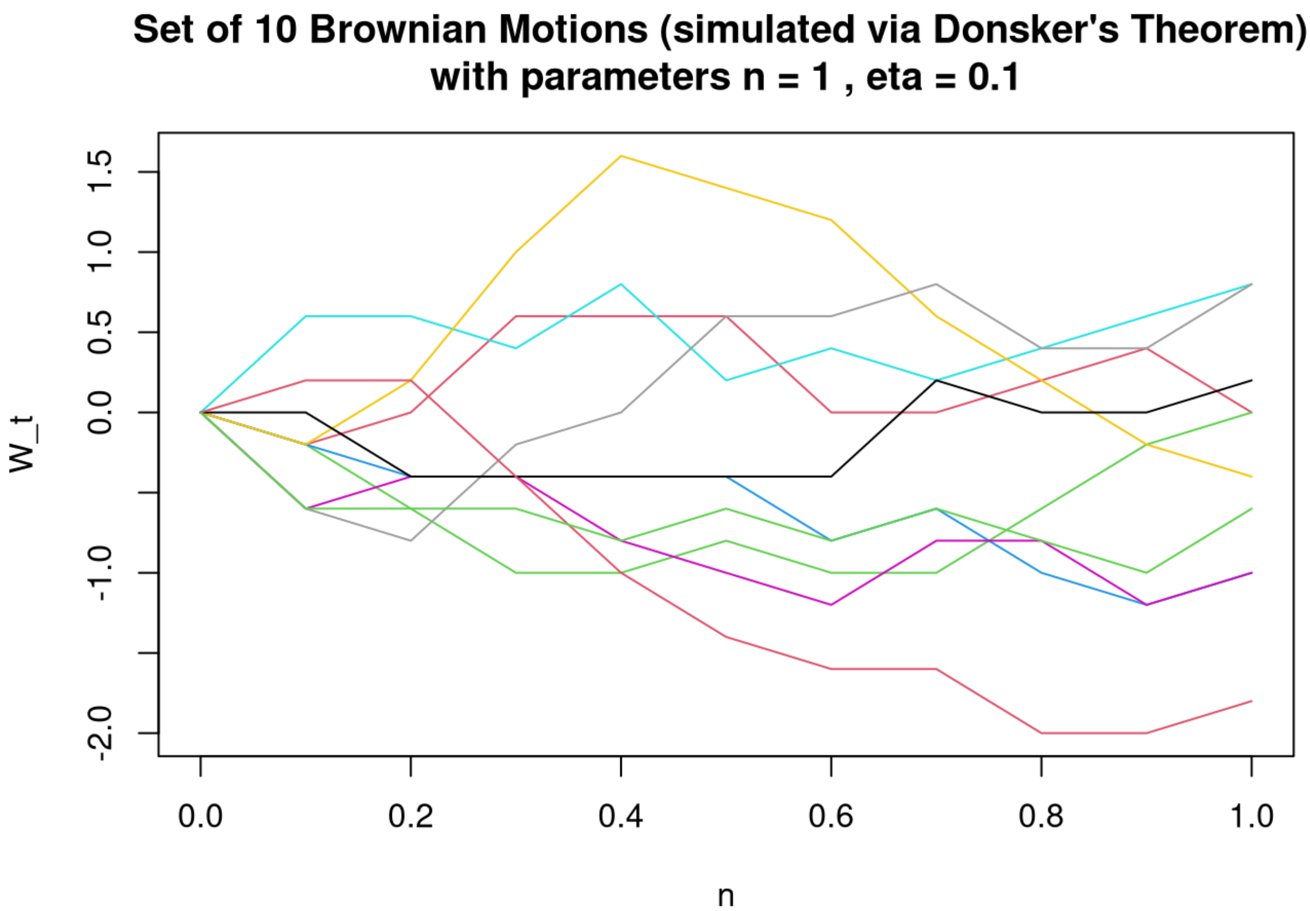
```
algo1_simulations = simulate("standard", 10, 100, 0.1)
```



RESULTS - Simulation with Algorithm 2

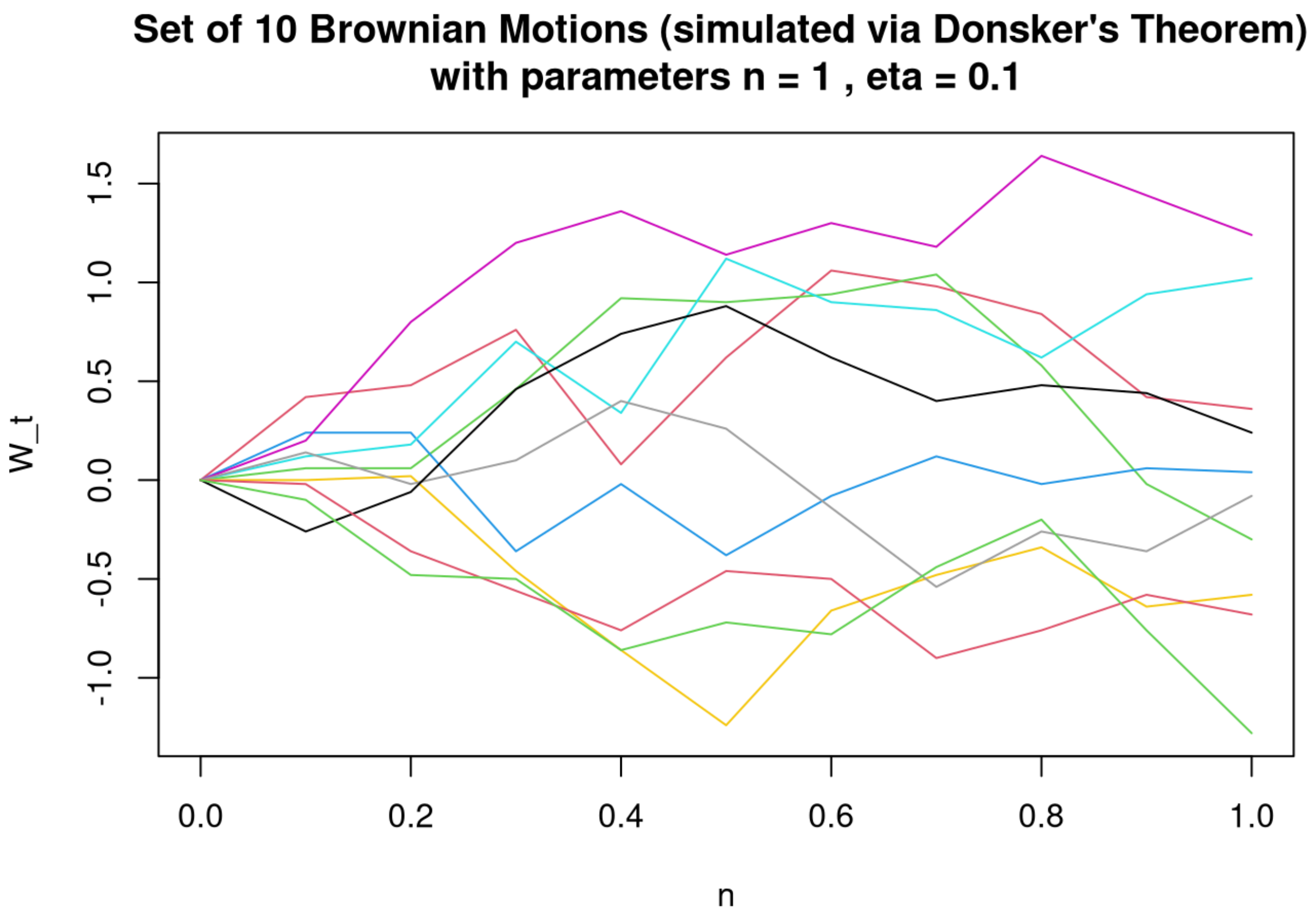
We generate 10 brownian motions with parameters $donsker_n = 100$ and $\eta = 0.1$, using the Donsker's theorem.

```
# Note: by the Donsker's theorem, the time length is set to 1
algo2_simulations = simulate("donsker", 10, time_length=1, 0.1, donsker_n=100)
```



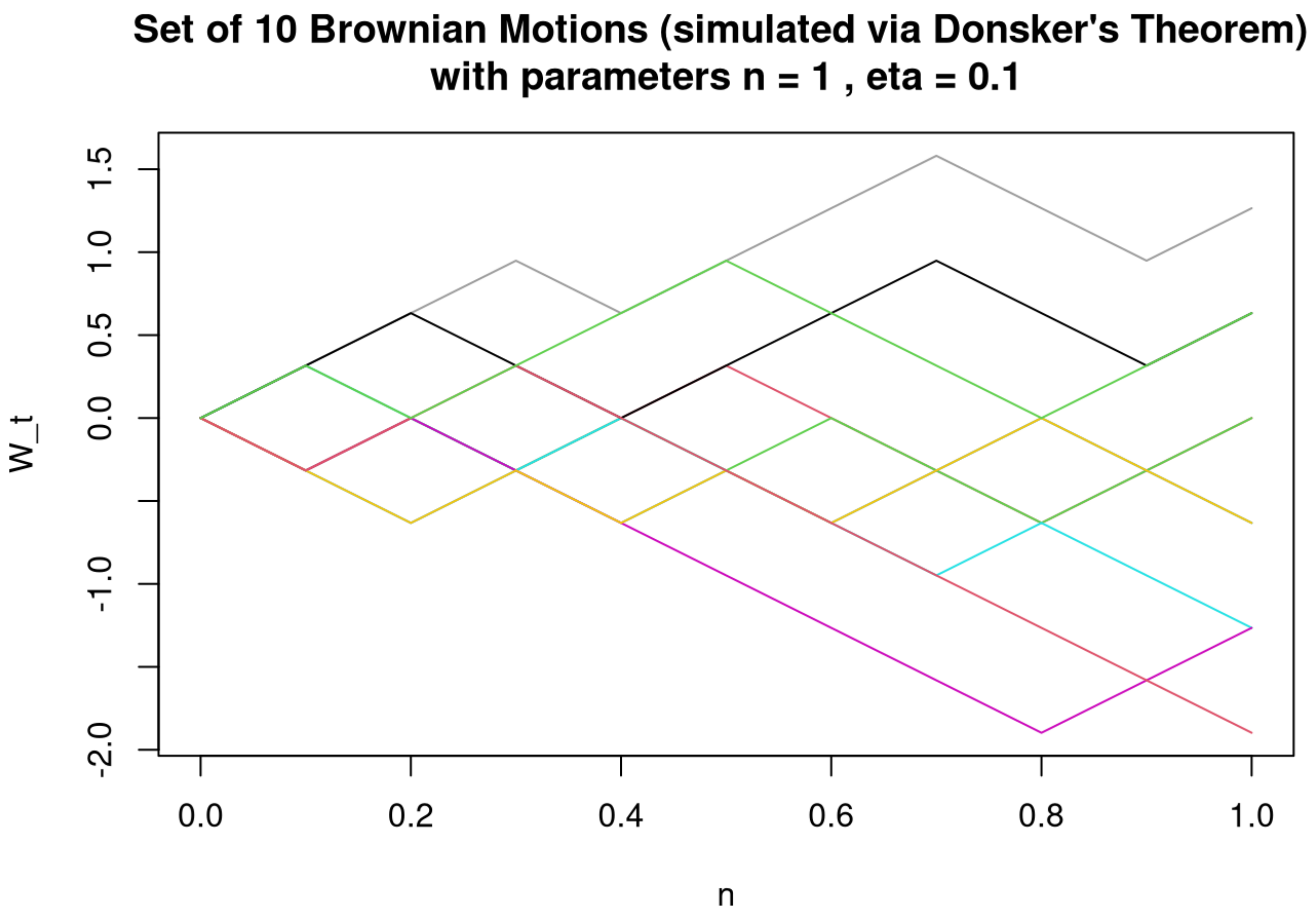
We generate 10 brownian motions with parameters $donsker_n = 10000$ and $\eta = 0.1$, using the Donsker's theorem.

```
# Note: by the Donsker's theorem, the time length is set to 1
algo2_simulations = simulate("donsker", 10, time_length=1, 0.1, donsker_n=10000)
```



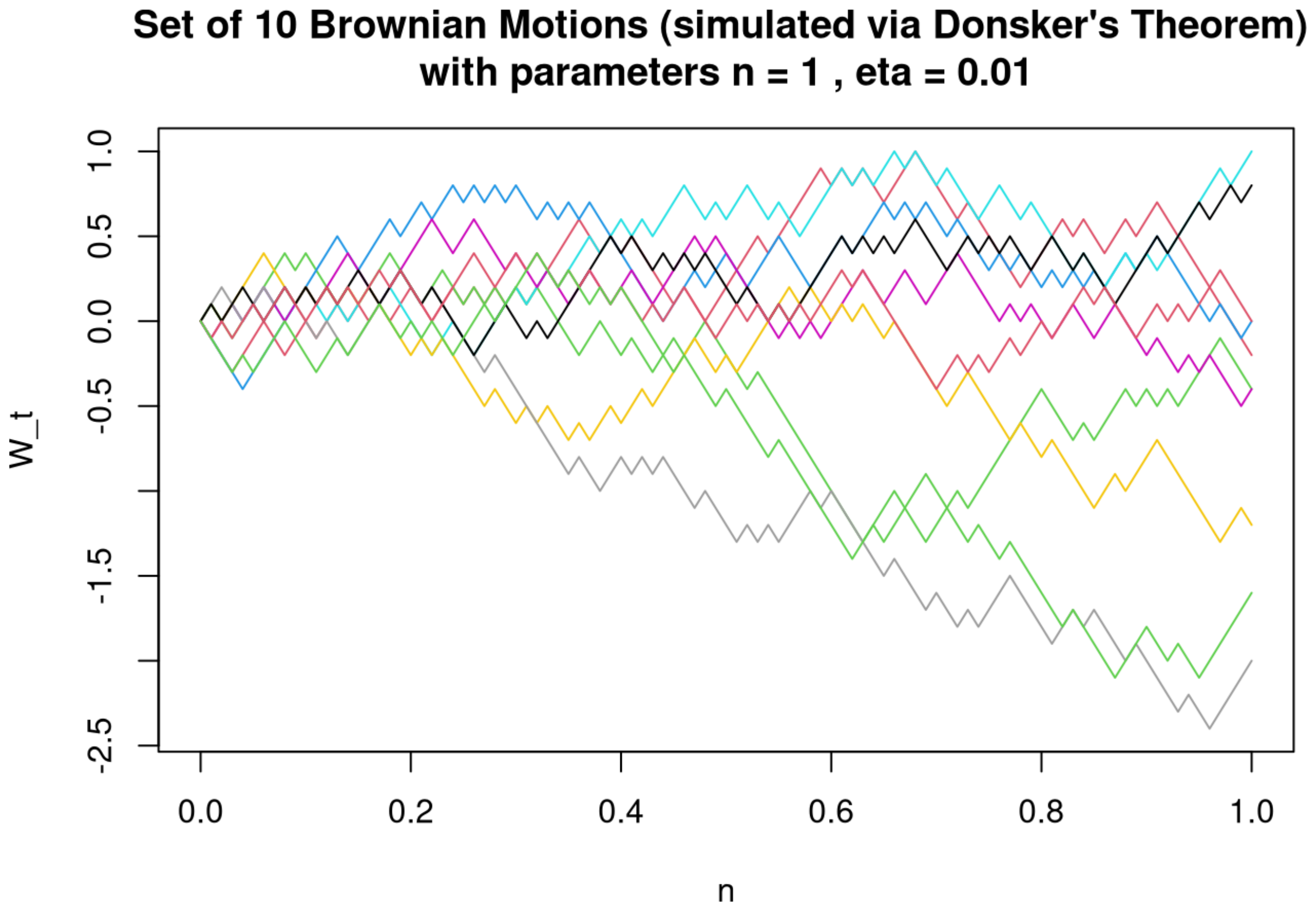
We generate 10 brownian motions with parameters $donsker_n = 10$ and $\eta = 0.1$, using the Donsker's theorem.

```
# Note: by the Donsker's theorem, the time length is set to 1
algo2_simulations = simulate("donsker", 10, time_length=1, 0.1, donsker_n=10)
```



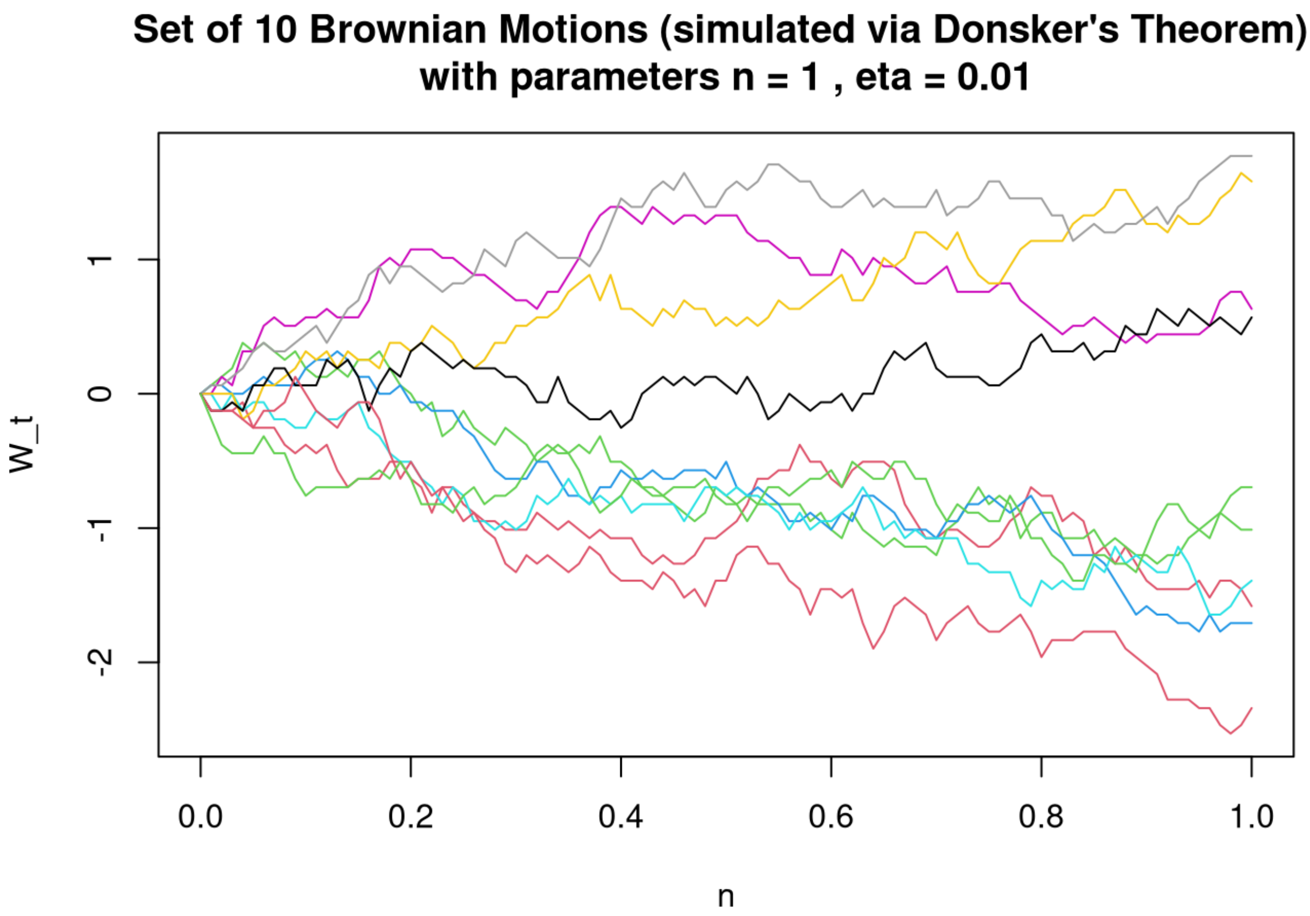
We generate 10 brownian motions with parameters $donsker_n = 100$ and $\eta = 0.01$, using the Donsker's theorem.

```
# Note: by the Donsker's theorem, the time length is set to 1
algo2_simulations = simulate("donsker", 10, time_length=1, 0.01, donsker_n=100)
```



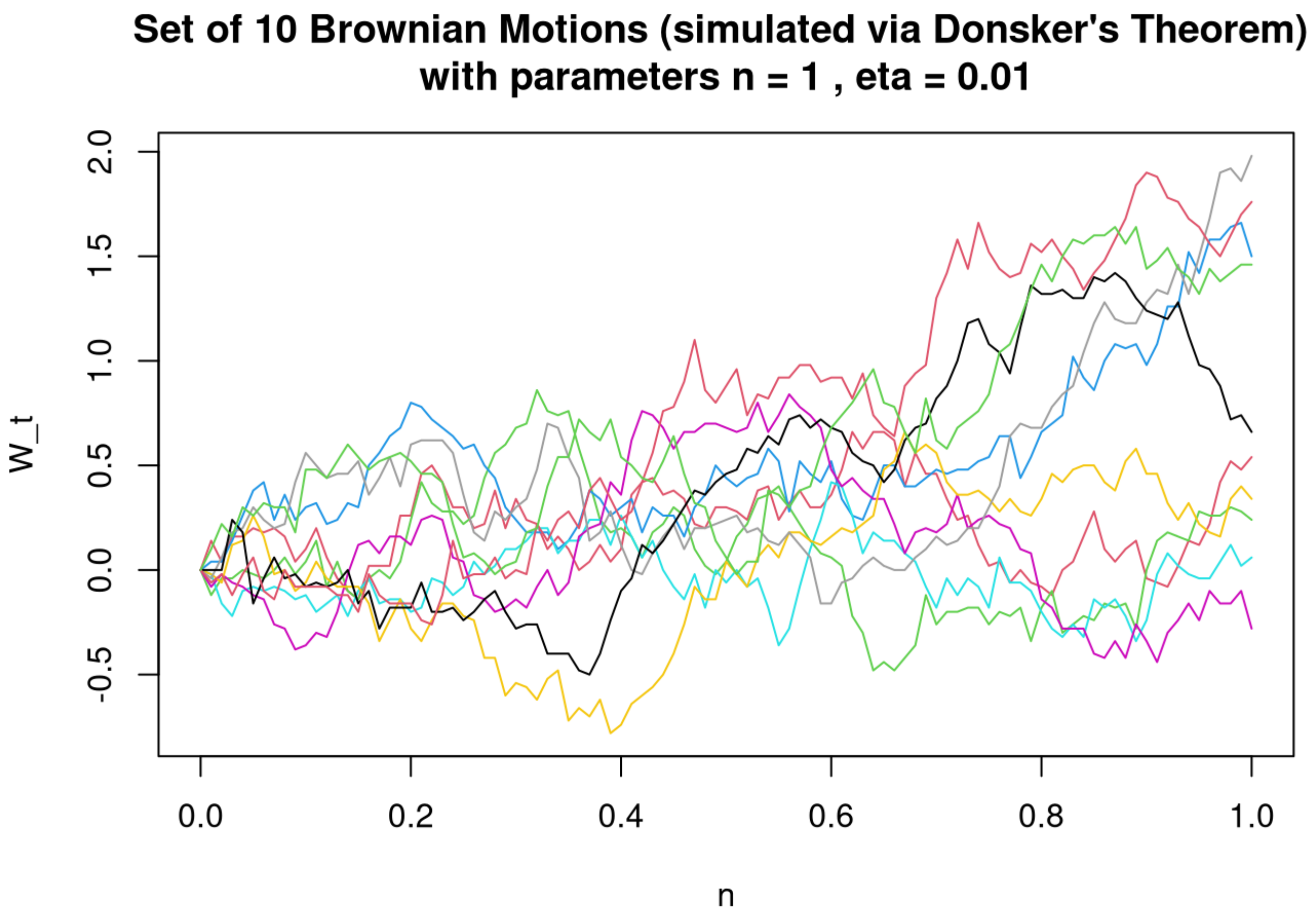
We generate 10 brownian motions with parameters $donsker_n = 1000$ and $\eta = 0.01$, using the Donsker's theorem.

```
# Note: by the Donsker's theorem, the time length is set to 1
algo2_simulations = simulate("donsker", 10, time_length=1, 0.01, donsker_n=1000)
```



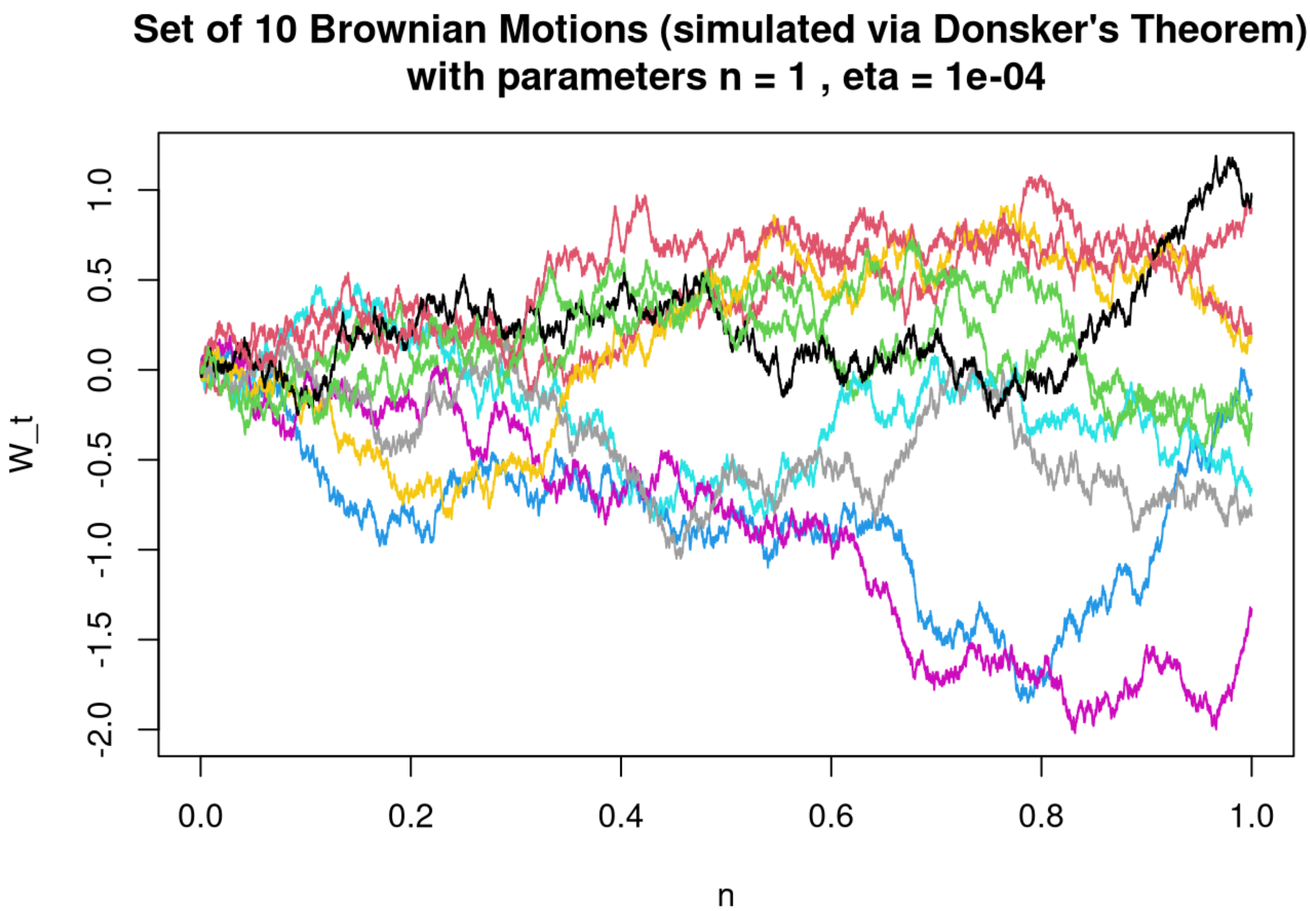
We generate 10 brownian motions with parameters $donsker_n = 10000$ and $\eta = 0.01$, using the Donsker's theorem.

```
# Note: by the Donsker's theorem, the time length is set to 1
algo2_simulations = simulate("donsker", 10, time_length=1, 0.01, donsker_n=10000)
```



We generate 10 brownian motions with parameters $donsker_n = 10000$ and $\eta = 0.0001$, using the Donsker's theorem.

```
# Note: by the Donsker's theorem, the time length is set to 1
algo2_simulations = simulate("donsker", 10, time_length=1, 0.0001, donsker_n=10000)
```



COMMENTS

We have shown two ways to yield a Brownian Motion. It is interesting to note that the algorithm relying on the Donsker's Theorem only produces a Brownian Motion on the interval $[0, 1]$ with a resulting range for W_t restricted around a mean of 0.

In order to obtain a Brownian Motion on a larger scale (e.g. $n \in [0, 10]$), it is interesting to consider how to rescale the resulting simulations (whether we only scale the x-axis, or whether we should rescale both x and y-axes).