Matlab Exercise I: Simulating Brownian Motion

Ruben Biesheuvel[†] and Mars Geuze[‡]

[†]Student number 4076680
[‡]Student number 4109419

February 15, 2015

1

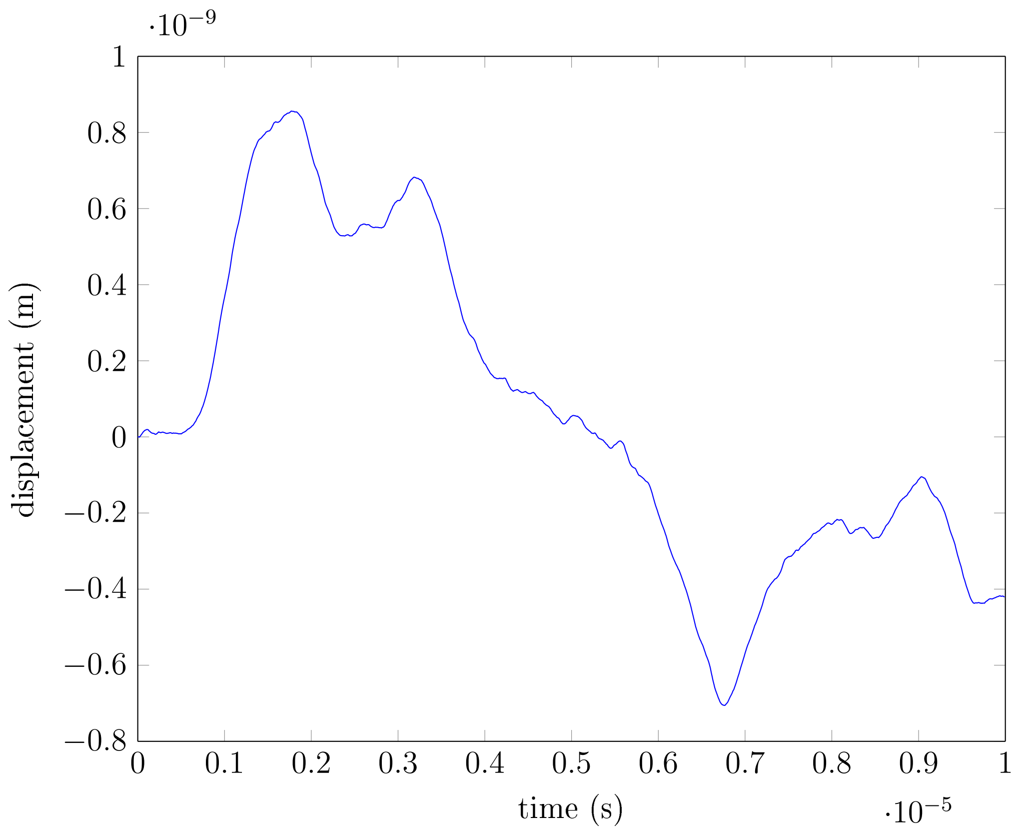
$$\beta_1 = -\frac{2m + \gamma \Delta t}{m + \gamma \Delta t} \tag{1}$$

$$\beta_2 = \frac{m}{m + \gamma \Delta t} \tag{2}$$

$$\beta_3 = \frac{\sqrt{2 k_B T \gamma \Delta t}}{\frac{m}{\Delta t} + \gamma} \tag{3}$$

2

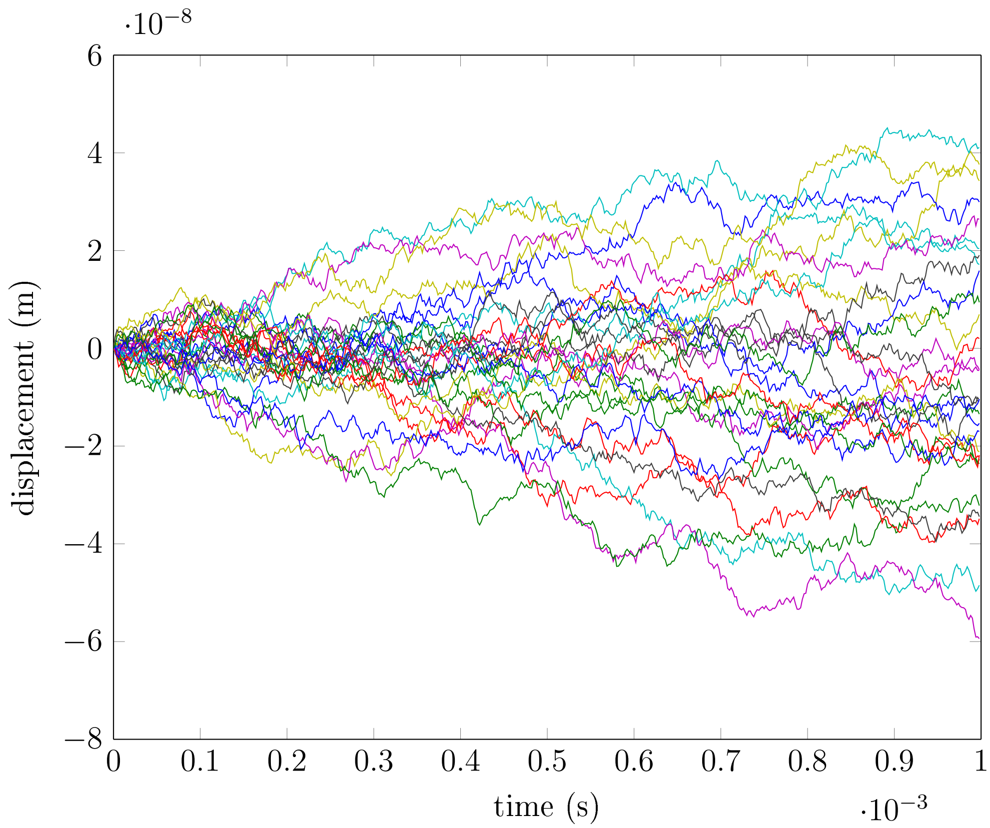
```matlab
%% Constants

N = 1e5;
dt = 1e-8; % s
R = 1e-6; % m
kB = 1.38e-23; % J/K
T = 300; % K
eta = 1e-3; % Pa s
rho = 2.6e3; % kg/m^3
gamma = 6*pi*R*eta; % Pa m s

% Compute particle mass in kg (nb. particles are spheres)
m = (4/3)*pi*R^3*rho;

% Expressions for the coefficients in terms of the constants given above
beta1 = -(2*m + gamma*dt)/(m + gamma*dt);
beta2 = m/(m + gamma*dt);
beta3 = sqrt(2*kB*T*gamma/dt)/(m/dt^2 + gamma/dt);

% Initialize signal vector (x) and generate white noise samples vector (w)
N2 = 1e3;
x = zeros(N2,1);
w = randn(N2,1);

% Simulate the difference equation
for k = 3:N2
    x(k) = - beta1*x(k-1) - beta2*x(k-2) + beta3*w(k);
end

% Plot the result as a function of time
time = (0:dt:dt*(N2-1));
plot(time,x);
xlabel('time (s)');
ylabel('displacement (m)');
```



$\cdot 10^{-9}$
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
displacement (m)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
time (s)
$\cdot 10^{-5}$

3

```
% Initialize number of simulation (L) and the signal and noise matrices (x
    and w)
L = 30;
x = zeros(N,L);
w = randn(N,L);

% Simulate the difference equation
for l = 1:L
    for k = 3:N
        x(k,l) = - beta1*x(k-1,l) - beta2*x(k-2,l) + beta3*w(k,l);
    end
end

% Plot the results
time = (0:dt:dt*(N-1));
plot(time,x);
xlabel('time (s)');
ylabel('displacement (m)');
```



4

```matlab
% Initialize the number of simulation (vector of L values)
% and the number of samples in each simulation (vector of H values)
Ls = [30,300,3000];
Hs = [10^3,10^4,10^5];

% Take L the maximum number of simulations we need to compare
% (we can take subsets for the lower values of L)
L = max(Ls);

% Note that N (defined above) is equal to the maximum h

% Initialize signal and noise matrices (x and w)
x = zeros(N,L);
w = randn(N,L);

% Simulate the difference equation L times
for l = 1:L
    for k = 3:N
        x(k,l) = - beta1*x(k-1,l) - beta2*x(k-2,l) + beta3*w(k,l);
    end
end

% Take different subsets of the data for each combination of the
% L and h parameters and plot as a histogram
for i = 1:length(Ls)
    L = Ls(i);

    for j = 1:length(Hs)
        h = Hs(j);

        % Select appropriate subplot
        subplot(length(Ls), length(Hs), (i-1)*length(Hs)+j);

        % Plot the histogram
        hist(x(h,1:L),sqrt(L));
        xlim([min(min(x)) max(max(x))]);
        xlabel('displacement (m)');
        title(sprintf('L=%d, h=%d', L, h));

    end
end
```

5

```
Kappa = zeros(3,3);
Sigma = zeros(3,3);
for i = 1:length(Ls)
    L = Ls(i);

    for j = 1:length(Hs)
        h = Hs(j);

        % Repeat hist command to get the data
        [counts, centers] = hist(x(h,(1:L)),sqrt(L));

        % Calculate initial estimates of sigma and kappa
        sigma0 = std(x(h,(1:L)));
        kappa0 = counts(round(length(counts)/2));

        % Define the objective function which we'll try to minimize
        % p is a vector with fit parameters:
        %     p(1) standard deviation (sigma)
        %     p(2) scale factor (kappa)
        fobj = @(p) sum( (counts - p(2)*exp(-centers.^2/2/p(1)^2)).^2 );

        % Fit the Gaussian through the histogram data
        p_opt = fminsearch(fobj, [sigma0, kappa0]);
        sigma = p_opt(1);
        kappa = p_opt(2);
        Sigma(i,j) = sigma;
        Kappa(i,j) = kappa;
        % Select appropriate subplot
        subplot(length(Ls), length(Hs), (i-1)*length(Hs)+j);

        % Plot the Gaussian on top of the histogram
        hold all;
        xg = linspace(min(min(x)), max(max(x)), 1000);
        yg = kappa.*exp(-(xg.^2)./(2.*sigma.^2));
        plot(xg, yg, 'r');
        xlabel(sprintf('sigma = %.2e (m)', sigma));
        hold off;

    end
end
```

6

Table 1: h $= 10^3$

Parameter \ L	30	300	3000
$\hat{\kappa}$	8.3	3.8×10^1	1.6×10^2
$\hat{\sigma}$	1.8×10^{-9}	2.1×10^{-9}	2.0×10^{-9}

Table 2: h $= 10^4$

Parameter \ L	30	300	3000
$\hat{\kappa}$	8.4	4.2×10^1	1.5×10^2
$\hat{\sigma}$	8.1×10^{-9}	6.7×10^{-9}	6.8×10^{-9}

Table 3: h $= 10^5$

Parameter \ L	30	300	3000
$\hat{\kappa}$	8.1	4.3×10^1	1.8×10^2
$\hat{\sigma}$	2.4×10^{-8}	2.1×10^{-8}	2.1×10^{-8}

7

From the Tables Table 1, Table 2 and Table 3 it can be concluded that within one drawing the height of the Gaussian fit ($\hat{\kappa}$) does not vary with the distance, but the standard deviation ($\hat{\sigma}$) does increase within one drawing when increasing the distance.

With more drawings, the height of the Gaussian fit goes up, because the fit is not normalized.

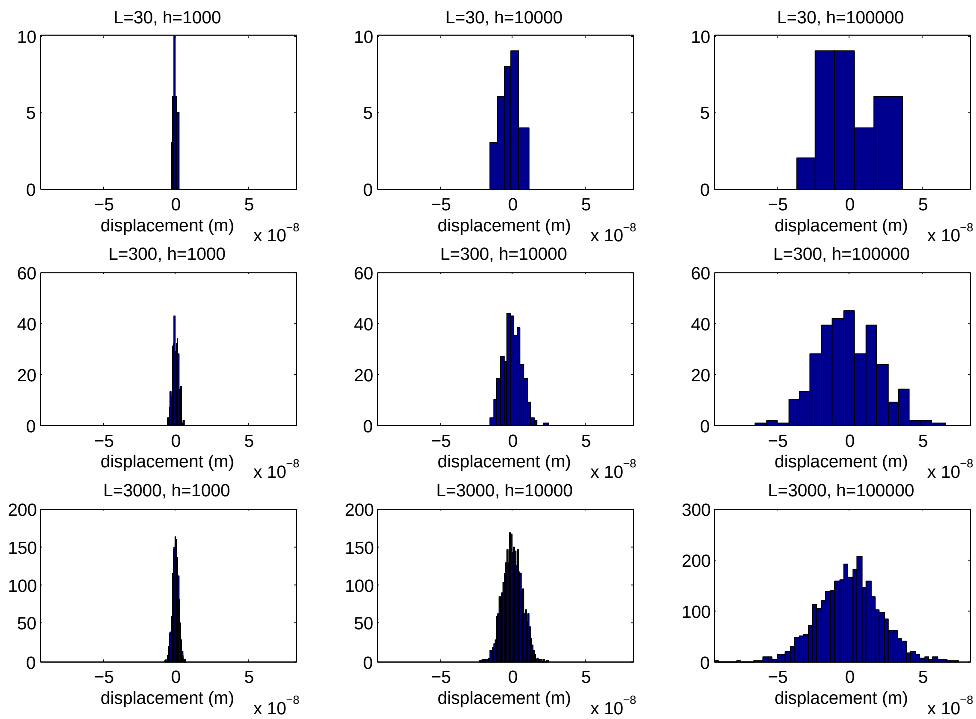


Figure 1: The histograms of the distribution of particles at varying distances (h) and different amount of particles (L)

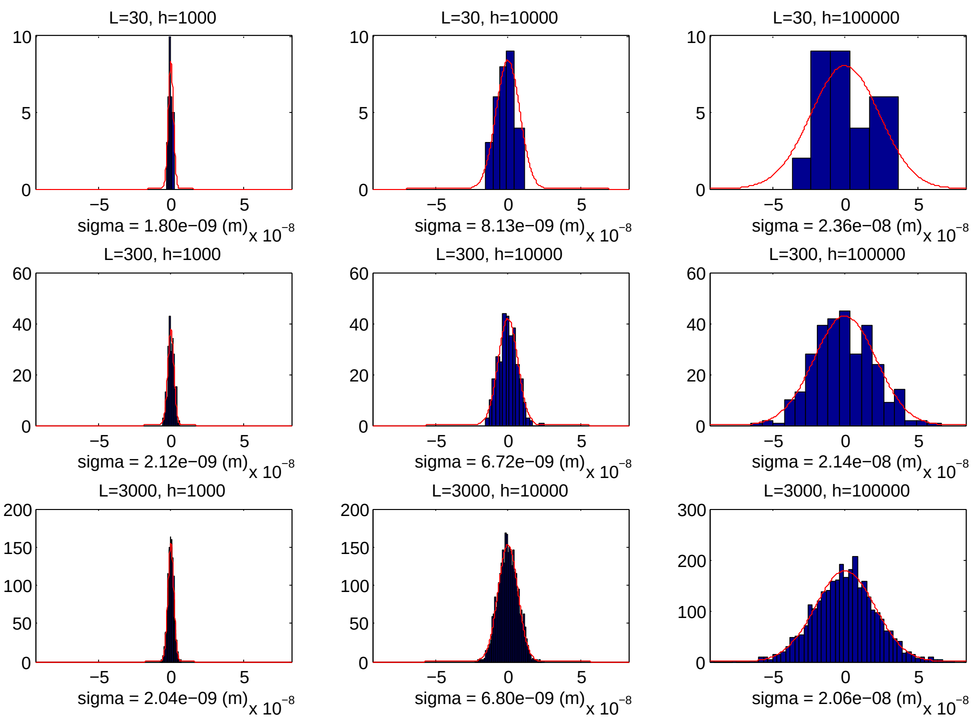


Figure 2: The histograms of the distribution of particles at varying distances (h) and different amount of particles (L), with the Gaussian fits as estimated by $\hat{\kappa}$ and $\hat{\sigma}$