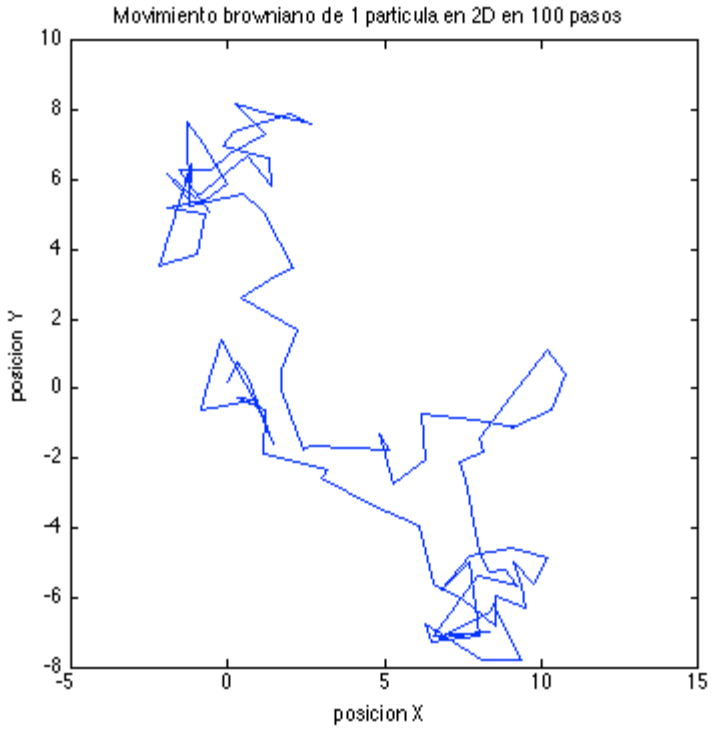
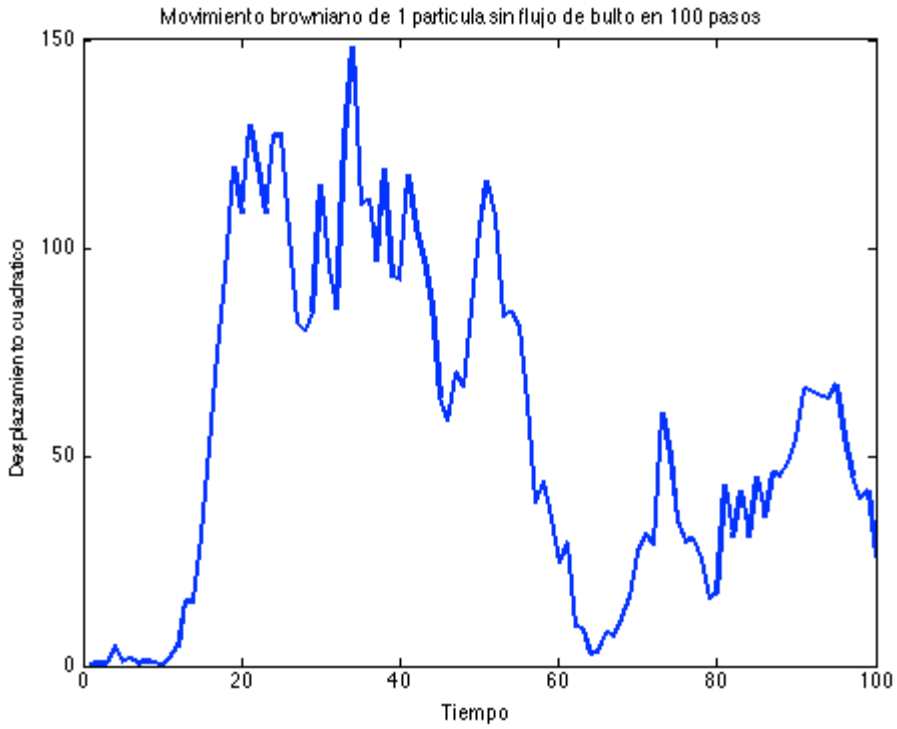
Contents


```
% Script: Simulacion del movimiento de particulas brownianas
% http://labs.physics.berkeley.edu/mediawiki/index.php/Simulating_Brownian_Motion

clear all
close all
clc

% Simulacion en 2D
N = 100; % N pasos
% Structura particula
particle = struct();
particle.x = cumsum(randn(N,1));
particle.y = cumsum(randn(N,1));

figure(1)
plot(particle.x, particle.y,'b','linewidth',1);
axis square
ylabel('posicion Y')
xlabel('posicion X')
title(['Movimiento browniano de 1 particula en 2D en ' num2str(N) ' pasos'])

dsquared = particle.x.^2 + particle.y.^2;

figure(2)
plot(dsquared,'linewidth',2)
title(['Movimiento browniano de 1 particula sin flujo de bulto en ' num2str(N) ' pasos'])
ylabel('Desplazamiento cuadratico')
xlabel('Tiempo')
```





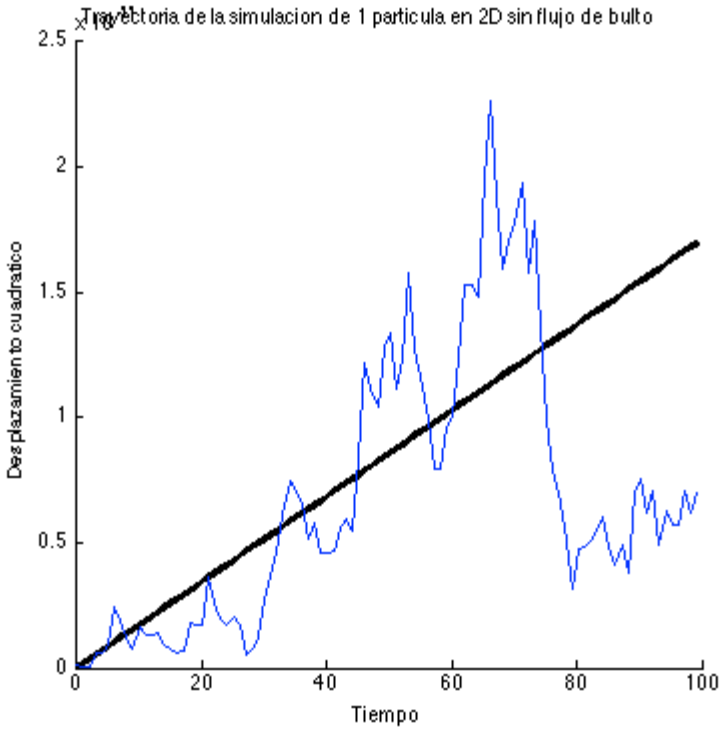
Valor teorico del coeficiente de difusion D

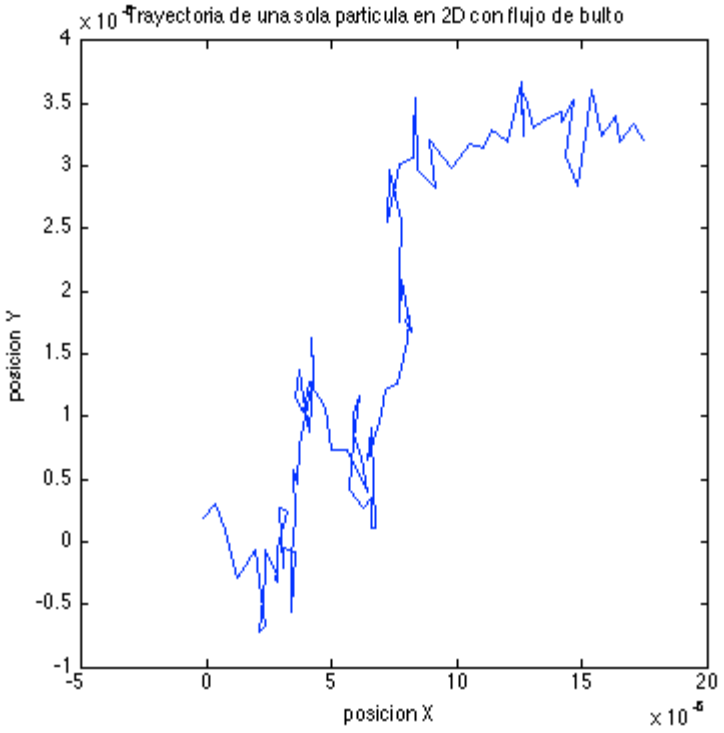
radio en metros

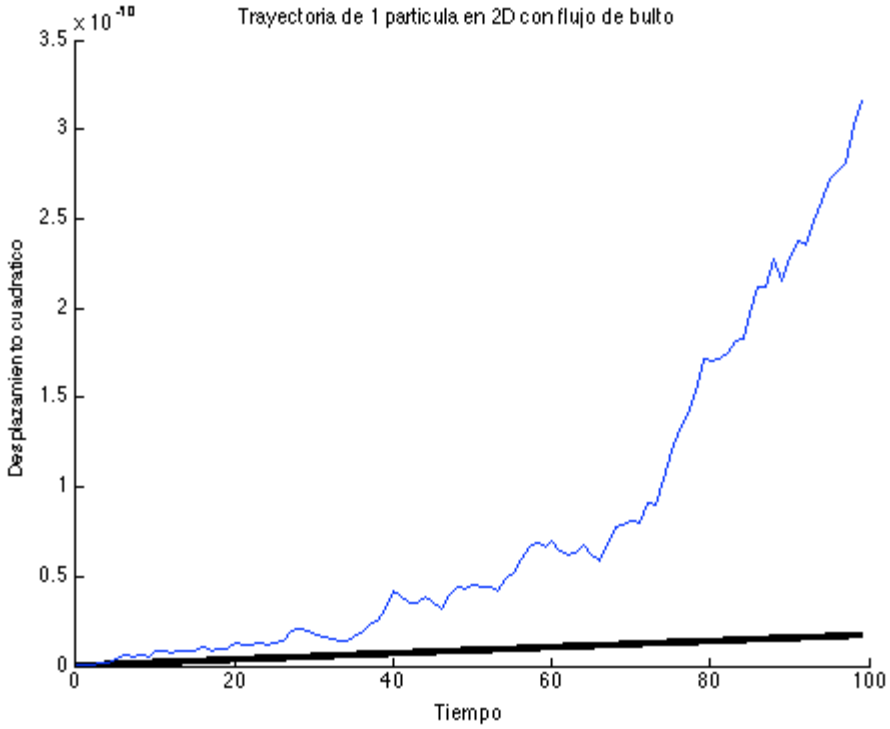
```
d = 1.0e-6;
```

```
% viscosidad del agua en unidades SI (Pascales-segundos) a 293 K
eta = 1.0e-3;
% Constante de Boltzmann
kB = 1.38e-23;
T = 293;
% Temperatura en grados Kelvin
D = kB*T/(3*pi*eta*d);

% Simulacion 2 dimensional
dimensions = 2;
% Intervalo de tiempo en segundos
tau = 0.1;
% Vector tiempo para graficar
time = tau*1:N;

k = sqrt(D*dimensions*tau);
dx = k*randn(N,1);
dy = k*randn(N,1);
x = cumsum(dx);
y = cumsum(dy);
dSquaredDisplacement = dx.^2 + dy.^2;
squaredDisplacement = x.^2 + y.^2;

figure(3)
plot(x,y);
% Grafica del desplazamiento cuadratico
clf
hold on
plot(time,(0:1:(N-1))*2*k^2,'k','LineWidth',3);
% grafica de la linea teorica
plot(time, squaredDisplacement);
hold off
axis square
xlabel('Tiempo')
ylabel('Desplazamiento cuadratico')
title('Trayectoria de la simulacion de 1 particula en 2D sin flujo de bulto');

% Estimando D a partir de datos simulados

simulatedD = mean(dSquaredDisplacement)/(2*dimensions*tau);

% Incertidumbre en la simulacion

standardError = std(dSquaredDisplacement)/(2*dimensions*tau*sqrt(N));
actualError = D - simulatedD;

% Error sistematico -- Flujo de bulto en el solvente

dx = dx + 0.6*k;
dy = dy + 0.2*k;
x = cumsum(dx);
y = cumsum(dy);
dSquaredDisplacement = dx.^2 + dy.^2;
squaredDisplacement = x.^2 + y.^2;

simulatedD = mean(dSquaredDisplacement)/(2*dimensions*tau);
```

```
standarError = std(dSquaredDisplacement)/(2*dimensions*tau*sqrt(N)) - simulatedD;
actualError = D - simulatedD;

figure(4)
plot(x,y);
xlabel('posicion X')
ylabel('posicion Y')
title('Trayectoria de una sola particula en 2D con flujo de bulto');
axis square

% Desplazamiento cuadratico en presencia de flujo de bulto

figure(5)
hold on;
% grafica de linea teorica
plot(time,(0:1:(N-1))*2*k^2,'k','LineWidth',3);
plot(time, squaredDisplacement);
hold off;
xlabel('Tiempo');
ylabel('Desplazamiento cuadratico');
title('Trayectoria de 1 particula en 2D con flujo de bulto');
```



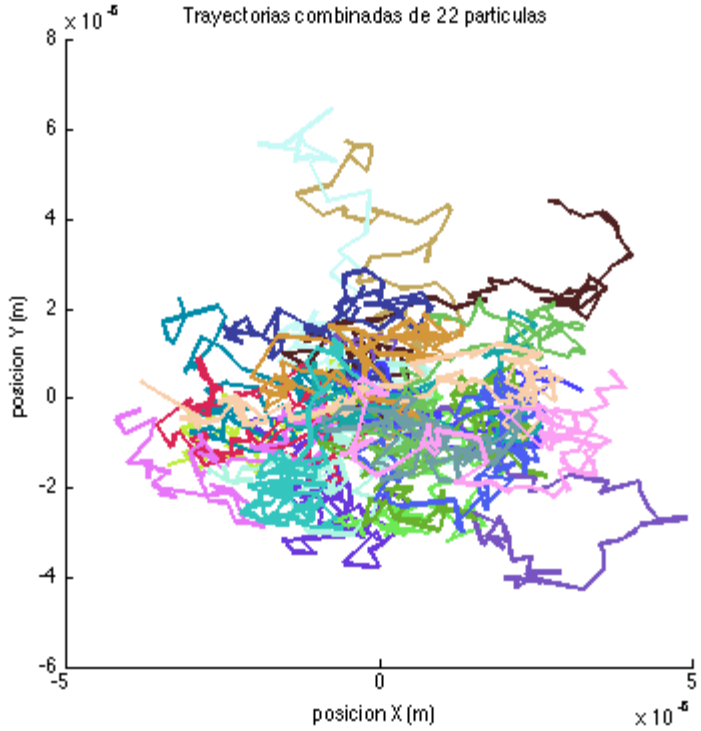


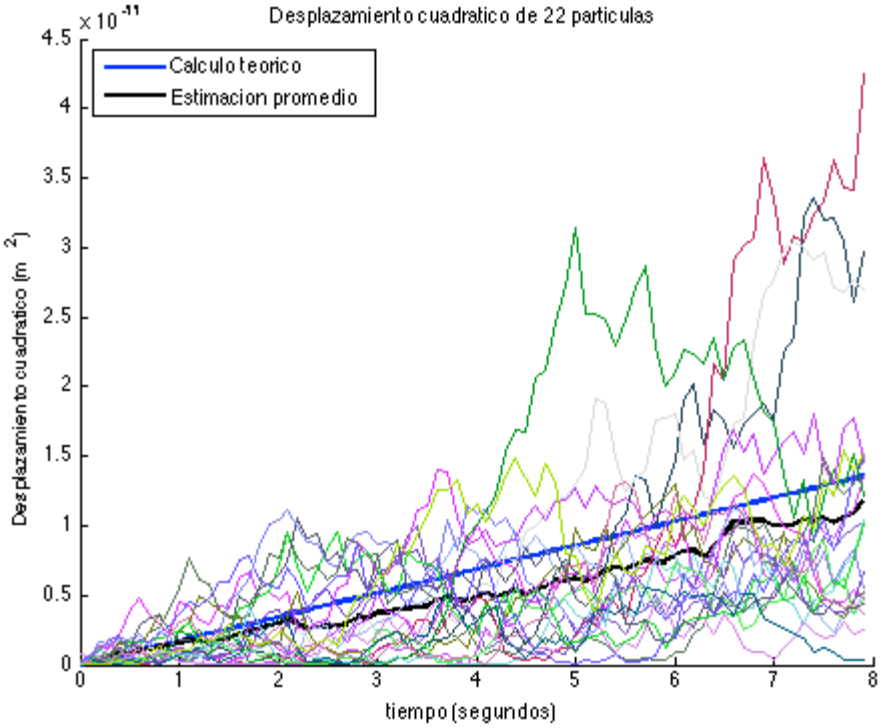


Simulacion de varias particulas

```
% Usando un ciclo (loop) para generar multiples conjuntos de datos

particleCount = 22;
```

```
N = 80;
tau = 0.1;
time = 0:tau:(N-1)*tau;
% Arreglo vacio (structura) para guardar resultados
particle = { };
for i = 1:particleCount
  particle{i} = struct();
  particle{i}.dx = k * randn(1,N);
  particle{i}.x = cumsum(particle{i}.dx);
  particle{i}.dy = k * randn(1,N);
  particle{i}.y = cumsum(particle{i}.dy);
  particle{i}.drsquared = particle{i}.dx.^2 + particle{i}.dy.^2;
  particle{i}.rsquared = particle{i}.x.^2 + particle{i}.y.^2;
  particle{i}.D = mean(particle{i}.drsquared)/(2*dimensions*tau);
  particle{i}.standardError = std( particle{i}.drsquared)/(2*dimensions*tau*sqrt(N));
end

figure(6)
hold on
for i = 1:particleCount
  plot(particle{i}.x,particle{i}.y,'color',rand(1,3),'linewidth',2);
end
xlabel('posicion X (m)');
ylabel('posicion Y (m)');
title(['Trayectorias combinadas de ' num2str(particleCount) ' particulas'])
axis square
hold off

% Desplazamiento cuadratico
% Calculo del conjunto promedio
rsquaredSum = zeros(1,N);

for i = 1:particleCount
  rsquaredSum = rsquaredSum + particle{i}.rsquared;
end
ensembleAverage = rsquaredSum / particleCount;
% crear una grafica
figure(7);
hold on
% grafica de linea teorica
plot(time,(0:1:(N-1))*2*k^2,'b','LineWidth',2);
% grafica del conjunto promedio
plot(time,ensembleAverage,'k','LineWidth',2);
legend('Calculo teorico','Estimacion promedio','location','NorthWest');
for i = 1:particleCount
  % grafica de la trayectoria de cada particula
  plot(time, particle{i}.rsquared,'color',rand(1,3));
end
xlabel('tiempo (segundos)');
ylabel('Desplazamiento cuadratico (m^2)');
title(['Desplazamiento cuadratico de ' num2str(particleCount) ' particulas']);
hold off
```



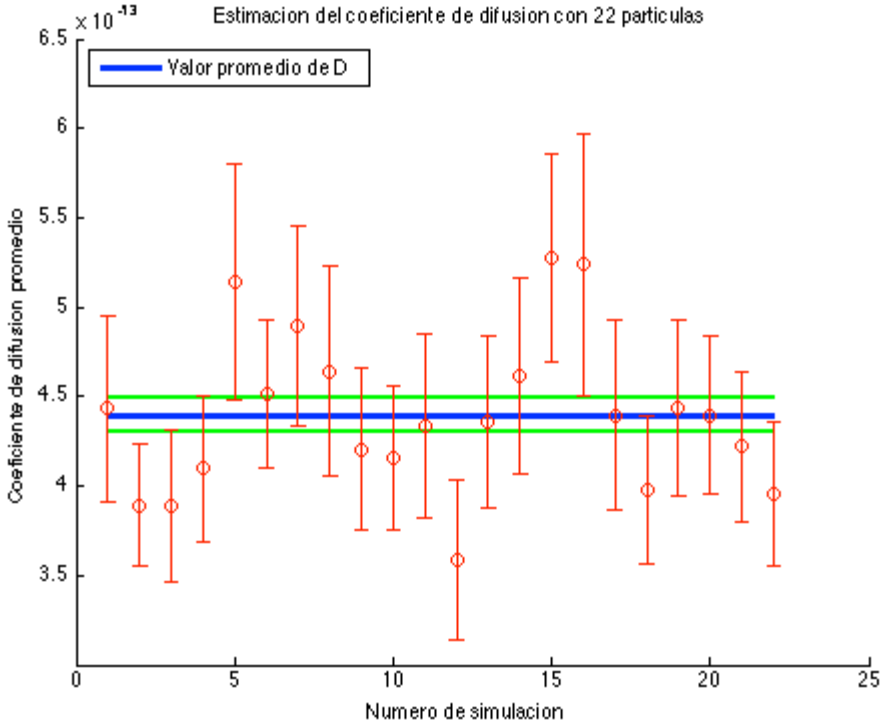


Estimacion del valor de D

```
clear D e dx;
% Obtener el valor de D a partir de cada simulacion
% y guardarlo en una sola matriz llamada D
```

```
for i = 1:particleCount
  D(i) = particle{i}.D;
  dx(i,:) = particle{i}.dx;
  e(i) = particle{i}.standardError;
end
% Calculo del estimado de D y su incertidumbre
averageD = mean(D)
uncertainty = std(D)/sqrt(particleCount)
% borrado de cada grafica usando 'clf';
figure(8)
% clf
hold on
% grafica del estimado de D
plot(averageD*ones(1,particleCount),'b','linewidth',3);
% grafica de la barra de limite superior
plot((averageD + uncertainty)*ones(1,particleCount),'g-','linewidth',2);
% grafica de la barra del limite inferior
plot((averageD - uncertainty)*ones(1,particleCount),'g-','linewidth',2);
% grafica de los valores de D con barras de error
errorbar(D,e,'ro');

xlabel('Numero de simulacion');
ylabel('Coeficiente de difusion promedio');
title(['Estimacion del coeficiente de difusion con ' num2str(particleCount) ' particulas'])
legend('Valor promedio de D', 'location', 'NorthWest');
hold off
```

```
averageD =

   4.3937e-13


uncertainty =

   9.4987e-15
```



Published with MATLAB® 7.14