Final Examination: Computational Physics

Pablo Lopez Duque

December 11^{th}, 2020

1 Problem 1

A Fokker-Plank equation describes the diffusion of a particle in a potential $U(x)$, with

$$\frac{\partial n}{\partial t} = D\frac{\partial^2 n}{\partial x^2} + \frac{1}{\zeta}\frac{\partial}{\partial x}(n\frac{\partial U}{\partial x})$$

For $U(x) = -fx$, with reflecting boundary conditions at $x = \pm L/2$, analytically determine the steady state solution for $n_\infty(x)$. Numerically integrate the Fokker-Plank equation using any method you choose (for any nonzero ζ and f you choose), and show that your predicted steady state solution is recovered as $t \to \infty$.

We can rewrite this Fokker-Planck equation as:

$$\frac{\partial n}{\partial t} = D\frac{\partial^2 n}{\partial x^2} + \frac{1}{\zeta}\frac{\partial n}{\partial x}\left(\frac{\partial U}{\partial x}\right) + \frac{1}{\zeta}n\left(\frac{\partial^2 U}{\partial x^2}\right)$$

Notice that $\frac{\partial U}{\partial x} = -f$ and $\frac{\partial^2 U}{\partial x^2} = 0$:

$$\frac{\partial n}{\partial t} = D\frac{\partial^2 n}{\partial x^2} - \frac{f}{\zeta}\frac{\partial n}{\partial x}$$

Now, let's use separation of variables: $n(x,t) = \chi(x)T(t)$, $\Rightarrow \frac{\partial n}{\partial t} = \chi(x)\dot{T}(t)$, $\frac{\partial^2 n}{\partial x^2} = \chi''(x)T(t)$, $\frac{\partial n}{\partial x} = \chi'(x)T(t)$

$$\chi(x)\dot{T}(t) = D\chi''(x)T(t) - \frac{f}{\zeta}\chi'(x)T(t)$$

$$\frac{\dot{T}(t)}{T(t)} = D\frac{\chi''(x)}{\chi(x)} - \frac{f}{\zeta}\frac{\chi'(x)}{\chi(x)} = -\kappa$$

So, the temporal solution is $T(t) = e^{-\frac{\kappa}{D}t}$, whereas the spatial equation becomes:

$$D\chi''(x) - \frac{f}{\zeta}\chi'(x) + \kappa\chi(x) = 0$$

The characteristic equation for the system is: $D\lambda^2 - \frac{f}{\zeta}\lambda + \kappa = 0$. Hence the solutions are given by the roots:

$$\lambda = \frac{1}{2D}\left(\frac{f}{\zeta} \pm \sqrt{\left(\frac{f}{\zeta}\right)^2 - 4D\kappa}\right)$$

Hence, the spatial solution is:

$$\chi(x) = c_1 e^{\frac{1}{2D}\left(\frac{f}{\zeta} + \sqrt{\left(\frac{f}{\zeta}\right)^2 - 4D\kappa}\right)} + c_2 e^{\frac{1}{2D}\left(\frac{f}{\zeta} - \sqrt{\left(\frac{f}{\zeta}\right)^2 - 4D\kappa}\right)}$$

Now, we can apply the reflecting boundary conditions. Hence:

$$\frac{\partial \chi}{\partial x} = \frac{c_1}{2}\left(\frac{f}{D\zeta} + \sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\right) e^{\frac{1}{2}\left(\frac{f}{D\zeta} + \sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\right)x} + \frac{c_2}{2}\left(\frac{f}{D\zeta} - \sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\right) e^{\frac{1}{2}\left(\frac{f}{D\zeta} - \sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\right)x}$$

$$\frac{\partial \chi}{\partial x}|_{\frac{L}{2}} = 0 = e^{\frac{f}{2D\zeta}x}\left[\frac{c_1}{2}\left(\frac{f}{D\zeta} + \sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\right) e^{\frac{1}{2}\sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\frac{L}{2}} + \frac{c_2}{2}\left(\frac{f}{D\zeta} - \sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\right) e^{-\frac{1}{2}\sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\frac{L}{2}}\right]$$

The exponential term outside the parenthesis cannot be zero for any finite value of the parameters. Hence, the term inside the parenthesis must be zero:

$$c_1 = -c_2 \frac{\left(\frac{f}{D\zeta} - \sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\right)}{\left(\frac{f}{D\zeta} + \sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\right)} e^{-\sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\frac{L}{2}}$$

In order to understand better the solution, let's consider the limit for $f \to 0$.

$$\chi(x) = c_1 e^{\frac{1}{2}\left(\frac{f}{D\zeta} + \sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\right)x} + c_2 e^{\frac{1}{2}\left(\frac{f}{D\zeta} - \sqrt{\left(\frac{f}{D\zeta}\right)^2 - \frac{4\kappa}{D}}\right)x}$$

$$\chi(x) = c_1 e^{(i\sqrt{\frac{\kappa}{D}})x} + c_2 e^{-i(\sqrt{\frac{\kappa}{D}})x} = Asin\left(\sqrt{\frac{\kappa}{D}}x\right) + Bcos\left(\sqrt{\frac{\kappa}{D}}x\right)$$

which yields the solution for the diffusion only equation. After applying the BCs in this case, we get:

$$A = Btan\left(\sqrt{\frac{\kappa}{D}}\frac{L}{2}\right); \qquad k = \left(\frac{2n\pi}{L}\right)^2 D$$

where $n = 0, 1, 2, ..$, and the $n = 0$ solution gives the constant distribution expected for the thermodynamic limit(because it leads to $\kappa = 0$ meaning the temporal part vanishes).
Similarly, we can analyze the limit where f dominates:

$$\chi'(x) - \frac{\kappa\zeta}{f}\chi(x) = 0$$

So, the solution is:

$$\chi(x) = Ae^{\left(\frac{\kappa\zeta}{f}x\right)}$$

$$\chi'(\pm\frac{L}{2}) = 0 = A\frac{\kappa\zeta}{f}exp\left(\pm\frac{\kappa\zeta}{f}\frac{L}{2}\right)$$

$$\Rightarrow \kappa = 0$$

which yields terms that converge to a constant in the thermodynamic limit. Hence, the general solution converges to a constant as well.

So, now let's get back to the original problem, by replacing the coefficient we found, we have:

$$\chi(x) = -c_2 \frac{\left(\frac{f}{D\zeta} - i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\right)}{\left(\frac{f}{D\zeta} + i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\right)} e^{\frac{1}{2}\left(\frac{f}{D\zeta} + i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\right)x - i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\frac{L}{2}} + c_2 e^{\frac{1}{2}\left(\frac{f}{D\zeta} - i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\right)x}$$

Hence:

$$\frac{\partial\chi}{\partial x} = c_2\left(\frac{f}{D\zeta} - i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\right)\frac{1}{2}\left[-e^{\frac{1}{2}\left(\frac{f}{D\zeta} + i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\right)x - i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\frac{L}{2}} + e^{\frac{1}{2}\left(\frac{f}{D\zeta} - i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\right)x}\right]$$

$$\frac{\partial\chi}{\partial x} = c_2\left(\frac{f}{D\zeta} - i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\right)\frac{1}{2}e^{\frac{f}{2D\zeta}x}\left[-e^{\frac{i}{2}\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}(x-L)} + e^{-\frac{i}{2}\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}x}\right]$$

$$\frac{\partial \chi}{\partial x} = c_2 \left(\frac{f}{D\zeta} - i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2} \right) \frac{1}{2} e^{\frac{f}{2D\zeta}x} e^{-\frac{i}{2}\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\frac{L}{2}} \left[-e^{\frac{i}{2}\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}(x - \frac{L}{2})} + e^{-\frac{i}{2}\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}(x - \frac{L}{2})} \right]$$

$$\frac{\partial \chi}{\partial x} = ic_2 \left(\frac{f}{D\zeta} - i\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2} \right) e^{\frac{f}{2D\zeta}x} e^{-\frac{i}{2}\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}\frac{L}{2}} \sin\left(\frac{1}{2}\sqrt{\frac{4\kappa}{D} - \left(\frac{f}{D\zeta}\right)^2}(x - \frac{L}{2}) \right)$$

So the BC at $x = L/2$ yields a condition on κ:

$$\sqrt{\frac{\kappa}{D} - \left(\frac{f}{2D\zeta}\right)^2} L = n\pi \Rightarrow \boxed{\kappa = \left(\frac{n\pi}{L}\right)^2 D + \frac{1}{D}\left(\frac{f}{2\zeta}\right)^2}$$

where $n = 0, 1, 2, \ldots$. So:

$$\chi_n(x) = c_2 \frac{\left(\frac{f}{D\zeta} - i\sqrt{\frac{n\pi}{L}}\right)}{\left(\frac{f}{D\zeta} + i\sqrt{\frac{n\pi}{L}}\right)} e^{\frac{f}{2D\zeta}x} e^{-\frac{i}{2}\sqrt{\frac{n\pi}{L}}\frac{L}{2}} \left[-e^{\left(\frac{i}{2}\sqrt{\frac{n\pi}{L}}\right)x} + e^{-\frac{i}{2}\sqrt{\frac{n\pi}{L}}x} \right]$$

$$n(x,t) = c_2 + \sum_n \chi_n(x) e^{-\kappa_n t}$$

The general solution is a linear combination of the terms $\chi(x)$ and $T(t)$. So, the behavior of the system depends on the relationship between the coefficients D, f and ζ.
In order to implement the numerical solution, I will use the Crank Nicolson algorithm. This involves setting up the parameters for the system, constructing the Hamiltonian and the CN matrices for the system and iteratively multiplying until the desired number of steps. The code is shown below with appropriate comments.

```
%  Program to solve the Fokker-Planck equation using a parameter f that
%  accounts for a constant force.
clear;

%% * Initialize parameters (grid spacing, time step, etc.)
Nprime = 100;   %input('Enter number of grid points: ');
N=Nprime+2;     %adding the ghost cells to apply Neummann BCs.
L = 100;        % System extends from -L/2 to L/2
delx = L/(N-1-2); % Grid cell size
x = delx*(0:N-1) - delx*(N-1)/2;  % Coordinates  of grid points
D = input('Enter D: ');  % diffusion coefficient. Larger means faster diffusion
zeta=1;
f=1;                %force
tau =0.01;     %input('Enter time step: ');

%% * Set up the Hamiltonian operator matrix
FPdiff = zeros(N);  % Set all elements to zero
coeff = D/(delx^2);
for i=2:(N-1)
   FPdiff(i,i-1) = coeff+f/(2*zeta*delx);
   FPdiff(i,i) = -2*coeff;  % Set interior rows
   FPdiff(i,i+1) = coeff-f/(2*zeta*delx);
end
% First and last rows for Neummann BCs

FPdiff(1,:) = FPdiff(3,:);
FPdiff(N,:) = FPdiff(N-2,:);
%Diff(1,1) = -2*coeff;
%Diff(1,2) = 2*(coeff);
```

```
%Diff(N,N) = -2*coeff;
%Diff(N,N-1) = 2*(coeff);
%% * Compute the Crank-Nicolson matrix
dCN = ( inv(eye(N) - .5*tau*FPdiff) * (eye(N) + .5*tau*FPdiff) );

%% * Initialize the function at t=0.
%I chose a Gaussian wave packet as the starting function, but a a
%function will serve as well
x0 = -L/4;                % Location of the center of the wavepacket
sigma0 = L/20;            % Standard deviation of the wavefunction
Norm_rho = 1/(sqrt(sigma0*sqrt(pi)));  % Normalization
Rho = Norm_rho * exp(-(x'-x0).^2/(2*sigma0^2)); %to add velocity .*exp(i_imag*k0*x')
der=diff(Rho);
der2=diff(Rho,2);
%% * Plot the initial density
figure(1); clf;
plot(x,Rho,'-');
hold all
plot(x(1:N-1)',der,'+');
plot(x(1:N-2)',der2,'o');
title('Initial density');
xlabel('x');  ylabel('\rho(x)'); legend('initial');
drawnow;
saveas(gcf,'Final_1Neumann_init','epsc');
pause(1);
hold off
%% * Initialize loop and plot variables
max_iter = input('Enter number of time steps(length of simulation): '); %choose for how ...
    long to run
plot_iter = max_iter/20; % How many plots we display
p_plot(:,1) = Rho;       % Record initial condition
iplot = 1;
figure(2); clf;
axisV = [-L/2 L/2 0 max(p_plot)]; % Fix axis min and max

%% * Loop over desired number of steps
for iter=1:max_iter
  %* Compute new density function using the Crank-Nicolson scheme
  Rho = dCN*Rho;
  %Diff(1,:) = Diff(3,:);
  %Diff(N,:) = Diff(N-2,:);
  %dCN = ( inv(eye(N) - .5*tau*Diff) * (eye(N) + .5*tau*Diff) );

  %* Periodically record values for plotting
  if( rem(iter,plot_iter) < 1 )
        iplot = iplot+1;
        tplot(iplot) = iter*tau;
        p_plot(:,iplot) = Rho;
        hold on;
        plot(x(2:N-1),p_plot(2:N-1,iplot)); % Display snap-shot of P(x)
        xlabel('x'); ylabel('P(x,t)');
        title(sprintf('after %g iterations',iter));
        drawnow;
        saveas(gcf,'Final_1Neumann_plots','epsc');
    end
end
```

Now, let's discuss the results. Figure 2 shows the results for $f = 1, \zeta = 1, D = 10$, that is when diffusion dominates over drift. As expected, the steady state distribution is a constant. This is the expectation for any values of the parameters but the dynamics should change depending on the strength of the force and the diffusion coefficient. However, my simulation goes near zero whenever f is larger than D as depicted in figure 1. This should be an issue in the implementation of the boundary conditions, i tried both implementations discussed in class, but both suffered the same issue and produced identical results.

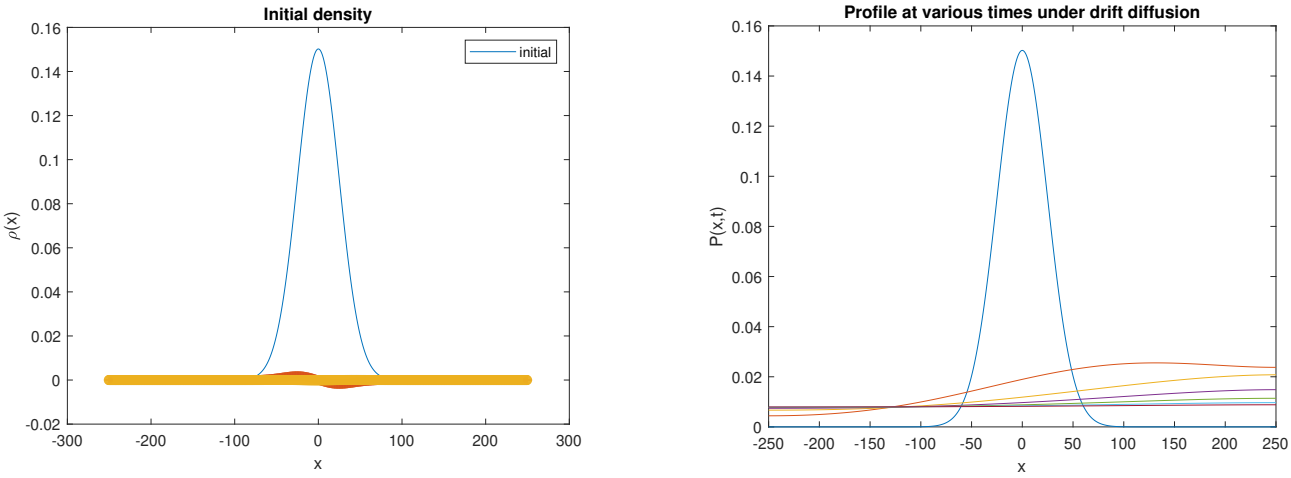

Figure 1: LEFT: initial density distribution, first derivative(red) and second derivative(yellow). RIGHT: Distribution for various times for $f = 1, \zeta = 1, D = 100$.

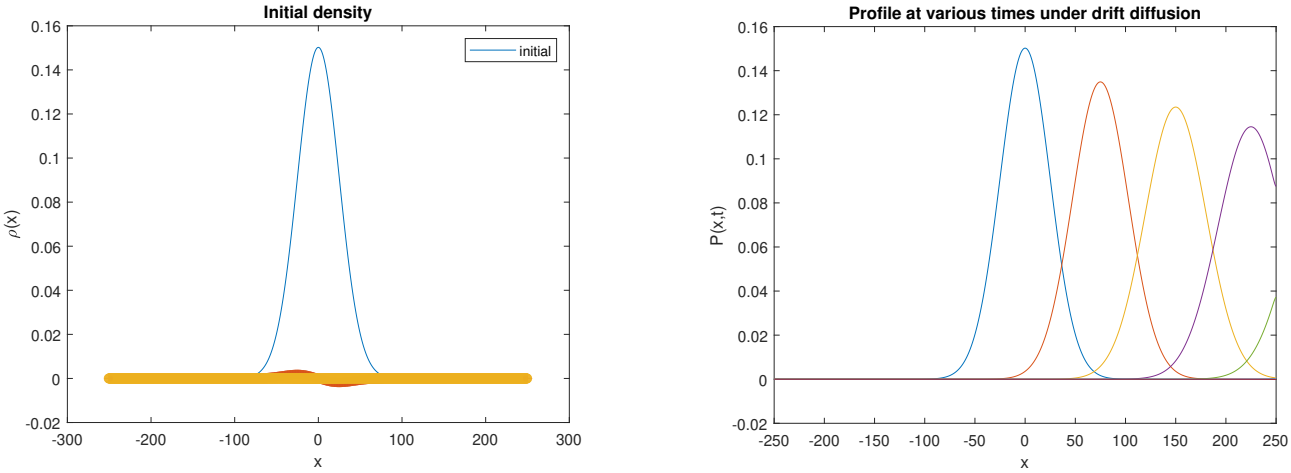

Figure 2: LEFT: initial density distribution, first derivative(red) and second derivative(yellow). RIGHT: Distribution for various times for $f = 1, \zeta = 1, D = 1$.

2 Problem 2

Numerically solve for the wavefunction of a particle approaching a δ function potential, satisfying:

$$-\frac{\hbar^2}{2m}\frac{\partial^2\Psi}{\partial x^2} + u\delta(x) = i\hbar\frac{\partial\Psi}{\partial t}$$

with the natural units $\hbar = m = 1$, using a Crank-Nicolson scheme. Use the the periodic boundary condition $\Psi(-L/2) = \Psi(L/2)$ for $L = 1000$ (in dimensionless units), and take the initial condition to be a traveling Gaussian packet $\Psi(x.0) = e^{ipx/\hbar - (x-x_0)^2/2\sigma^2}$ representing a particle moving towards the δ function barrier with momentum p. You should use $x0 = -L/4$, $p = 1$, and $\sigma = 20$ (all in dimensionless units). Show that for $u = 0.5$ and $u = 2.0$ the wavepacket is partially transmitted and partially reflected. Numerically compute the amplitude of the transmitted and reflected packets for $u = 0.5$ and 2.0.

The setup for the problem is identical to the one previously used for the class and homework. However, in this case, we must account for the δ potential. So, we must include the effect of the potential in our scheme. Recall the Hamiltonian operator is given by:

$$H_{jk} = -\frac{\hbar^2}{2m}\frac{\delta_{j+1,k} - 2\delta_{jk} + \delta_{j-1,k}}{\Delta x^2} + V_{jk}\delta_{jk}$$

So, we can replace $V_{jk} = u\delta(x)$ in the Hamiltonian and use an approximate version of the δ function. We can basically model it as $1/\Delta x^2$, the inverse if the grid step squared. Then, we can insert it in the expression for the CN algorithm:

$$\Psi^{n+1} = (\mathbf{I} + \frac{i\Delta t}{2\Delta x})^-1(\mathbf{I} - \frac{i\Delta t}{2\Delta x})\Psi^n$$

and solve the system considering periodic BCs. Firstly, one must set up the parameters for the system:

```
%% * Initialize parameters (grid spacing, time step, etc.)
i_imag = sqrt(-1);     % Imaginary i
N = 1000 ;%input('Enter number of grid points: ');
L = 1000;               % System extends from -L/2 to L/2
h = L/(N-1);            % Grid size
x = h*(0:N-1) - L/2;  % Coordinates  of grid points
h_bar = 1;   mass = 1; % Natural units
u=2.0;
tau = 1;%input('Enter time step: ');
```

Then, we initialize the Hamiltonian matrix taking into account the delta function potential only at the origin (center of our grid) and also initialize the Crank Nicolson matrix:

```
%* Set up the Hamiltonian operator matrix
ham = zeros(N);  % Set all elements to zero
coeff = -h_bar^2/(2*mass*h^2);
for i=2:(N-1)
  ham(i,i-1) = coeff;
  ham(i,i) = -2*coeff;  % Set interior rows
  ham(i,i+1) = coeff;
end
% First and last rows for periodic boundary conditions
ham(1,N) = coeff;   ham(1,1) = -2*coeff; ham(1,2) = coeff;
ham(N,N-1) = coeff; ham(N,N) = -2*coeff; ham(N,1) = coeff;
%Center must include the Δ function
ham(N/2,N/2)=-2*coeff+u*1/h^2;

%* Compute the Crank-Nicolson matrix
dCN = ( inv(eye(N) + .5*i_imag*tau/h_bar*ham)(eye(N) - .5*i_imag*tau/h_bar*ham) );
```

Then, to initialize the wave function:

```
%* Initialize the wavefunction
x0 = -L/4;          % Location of the center of the wavepacket
  % Average velocity of the packet
k0 = 1;        % Average wavenumber. Recall p=\hbar k, since \hbar=1 p, k are equivalent
sigma0 = 20;   % Standard deviation of the wavefunction
Norm_psi = 1/(sqrt(sigma0*sqrt(pi)));  % Normalization
psi = Norm_psi * exp(i_imag*k0*x') .* exp(-(x'-x0).^2/(2*sigma0^2));
```

And finally, we iteratively update the wavefunction:

```
for iter=1:max_iter
  %* Compute new wave function using the Crank-Nicolson scheme
   psi = dCN*psi;
  %* Periodically record values for plotting
   if( rem(iter,plot_iter) < 1 )
     iplot = iplot+1;
     p_plot(:,iplot) = psi.*conj(psi);
     plot(x,p_plot(:,iplot));      % Display snap-shot of P(x)
     xlabel('x'); ylabel('P(x,t)');
     title(sprintf('Finished %g of %g iterations',iter,max_iter));
     axis(axisV); drawnow;
   end
end
```

The resulting plots show two possible scenarios depending on the value for u. For $u = 0.5$ the interaction is still small and the transmitted wavefunction is larger than the reflected one. However, for $u = 2$, the interaction becomes greater and the transmission probability is smaller than the reflection probability. Physically, this is the expected behavior, since the delta function "strength" u determines how is the interaction between the incoming and outgoing waves. Figures 3 and 4 show a snapshot of the two waves. A better idea on what going on is presented on the attached gif files.

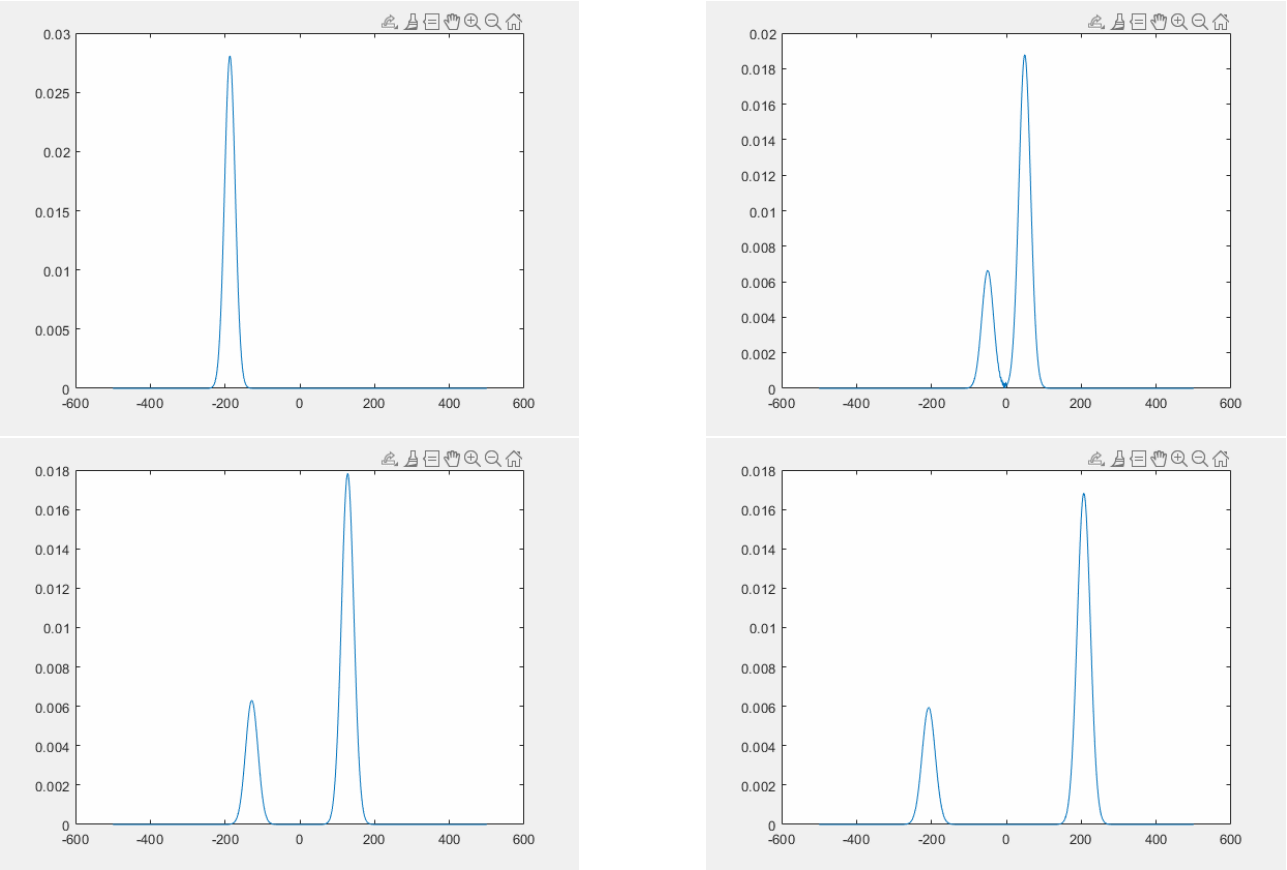

Figure 3: Snapshots for the interaction between the wavepacket and a delta potential of strength $u = 0.5$

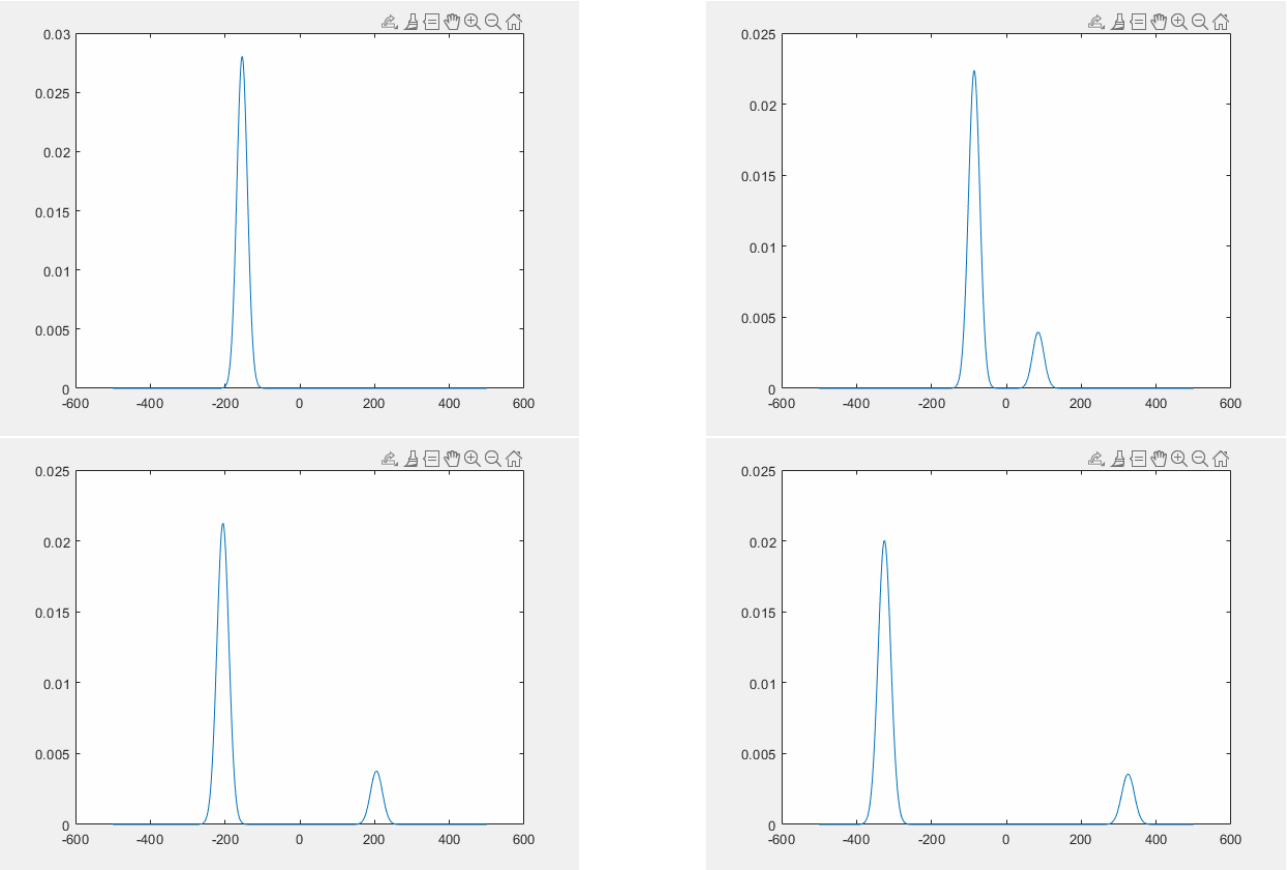

Figure 4: Snapshots for the interaction between the wavepacket and a delta potential of strength $u = 2$

Finally, the values for the transmission and reflection probabilities are calculated based on the amplitudes of the incoming and outgoing wave packets. For $u = 0.5$ the reflection probability is $R \approx 0.2611$ and the transmission is $T \approx 0.7389$. Meanwhile, for $u = 2$, the coefficients are: $R = 0.8496$ and $T = 0.1504$. This makes sense since a greater transmission is expected for a "weaker" delta potential and conversely greater reflection is expected for a stronger potential.

3 Problem 3

In this problem you will simulate the dynamics of a 2D Ising model, with Hamiltonian $\beta H = -J\sum_{<ij>} s_i s_j$. In all problems, you should simulate a square system of $N = 50 \times 50$ on a square lattice, using periodic boundary conditions. All simulations should include 5×10^5 iterations, and include 100 runs per simulation.

The code for this problem was modified from the ising.m code provided in class. The parameters for the system were modified accordingly:

```
clear;
rng(2,'twister');

N=50;
nstep=500000;
printstep=1;
nprint=floor(nstep/printstep);
ncorr=floor((nstep-10^4)/printstep);
nrun=100;

energy=zeros(nstep,1);
t=zeros(nprint,1);
avm=zeros(nprint,1);
avmsq=zeros(nprint,1);
ave=zeros(nprint,1);
avesq=zeros(nprint,1);
corr=zeros(ncorr,1);
%where we store averages

J=0.1;
b=0;
pbc=true;
Tc=0;
```

The remaining modifications will be discussed when relevant to the question.

(a) **Starting from a random initial state (50% spins up, and 50% spins down), plot the average energy as a function of time for J = 0.1, 0.2, 0.3, and 0.4. Plot the energy as a function of the iteration number n, $< E_n >$, as well as the fluctuations in the energy, $< E_n^2 > - < E_n >^2$. Do the fluctuations depend significantly on n and/or J?**

The modifications from the original code were only cosmetic. There were additions for calculating the average of the energy squared and the fluctuations, implemented in a similar fashion to the average energy:

```
    if(mod(step,printstep)==0)
          k=floor(step/printstep);
          t(k)=step;
          avm(k)=avm(k)+abs(m)/N/N;
          avmsq(k)=avmsq(k)+(abs(m)/N/N)*(abs(m)/N/N);
          ave(k)=ave(k)+e/J/N/N;
          avesq(k)=avesq(k)+(e/J/N/N)*(e/J/N/N);
          fluct(k)=avesq(k)-ave(k)^2;
          %save the averages
      end
```

The following figures, 5-8 show the initial and final spin configurations, the average energy for a single run, the average energy for all runs, the average energy squared and the fluctuations in the energy as a function of the iteration number(i.e. the time).

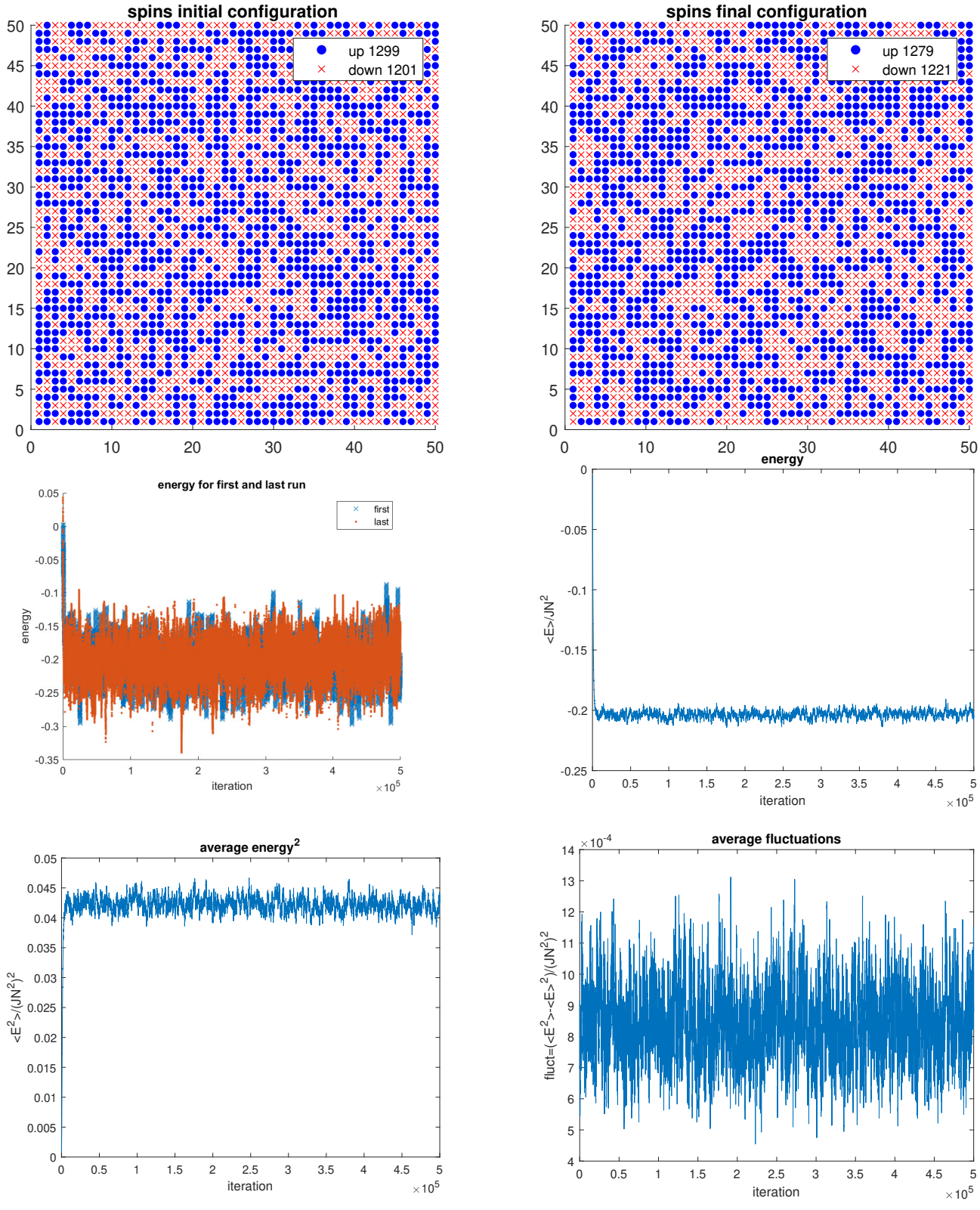

Figure 5: Initial and final spin configurations, average energy for a specific run, average energy over all runs, average n squared, and average fluctuations for J=0.1

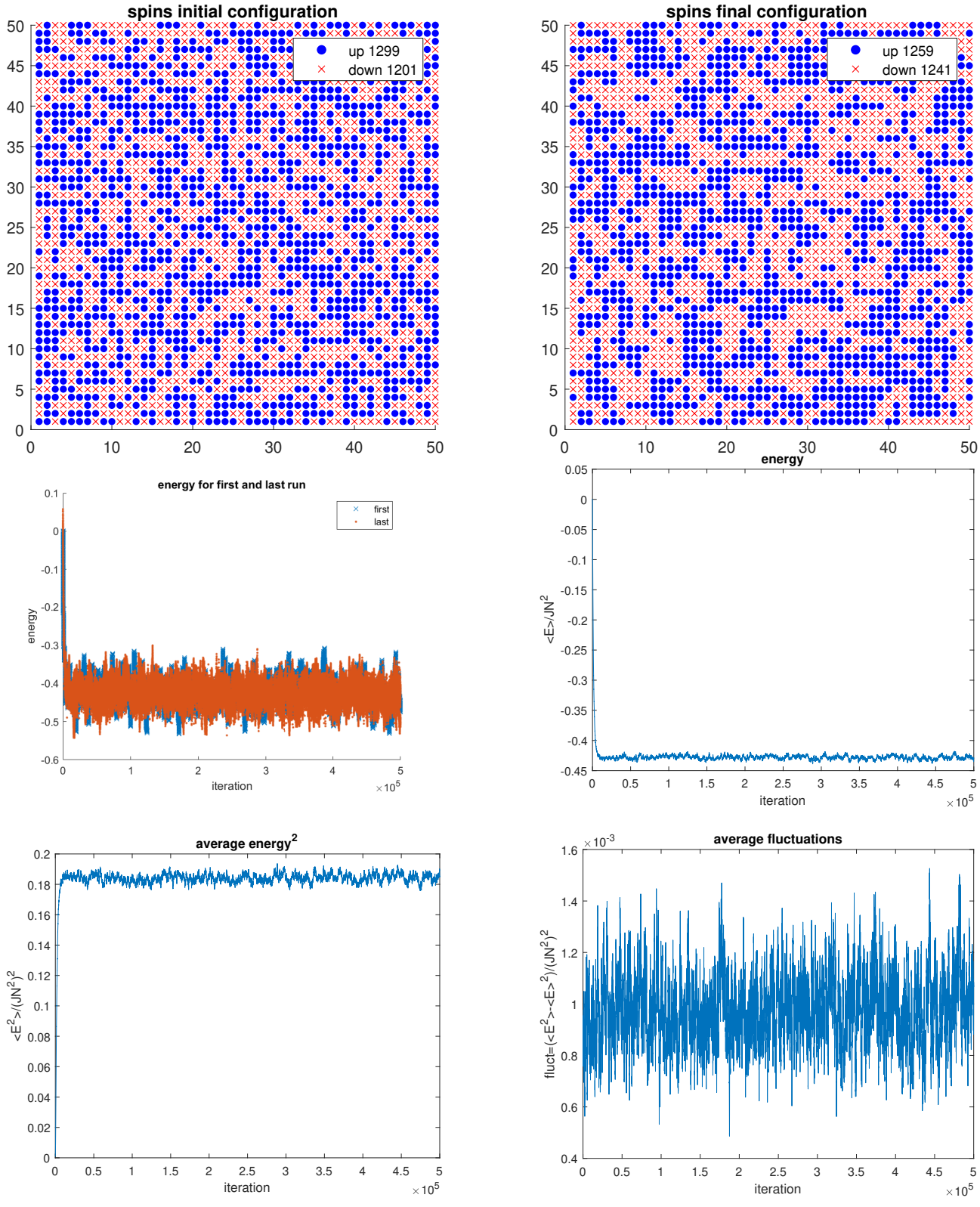

Figure 6: Initial and final spin configurations, average energy for a specific run, average energy over all runs, average n squared, and average fluctuations for J=0.2

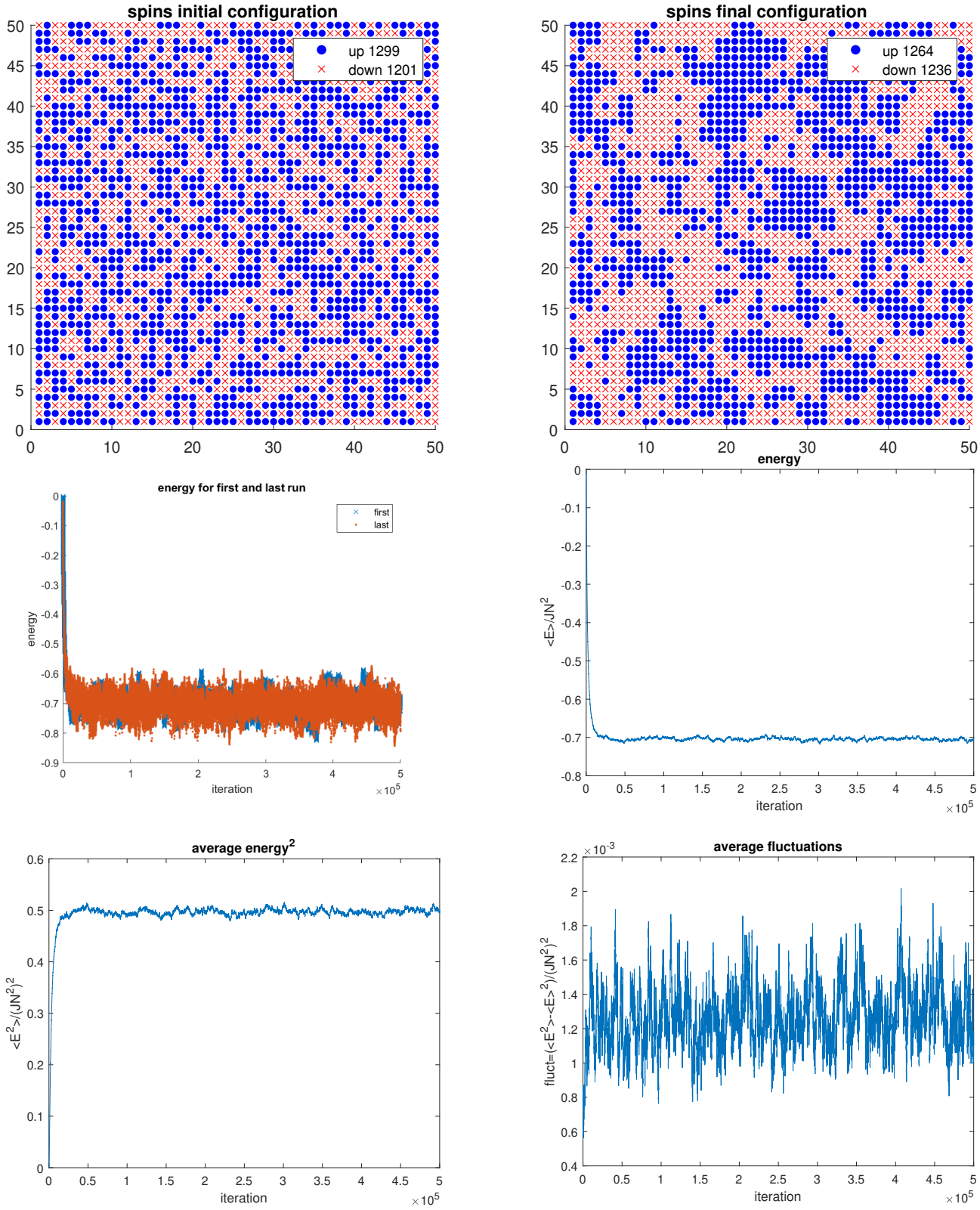

Figure 7: Initial and final spin configurations, average energy for a specific run, average energy over all runs, average n squared, and average fluctuations for J=0.3

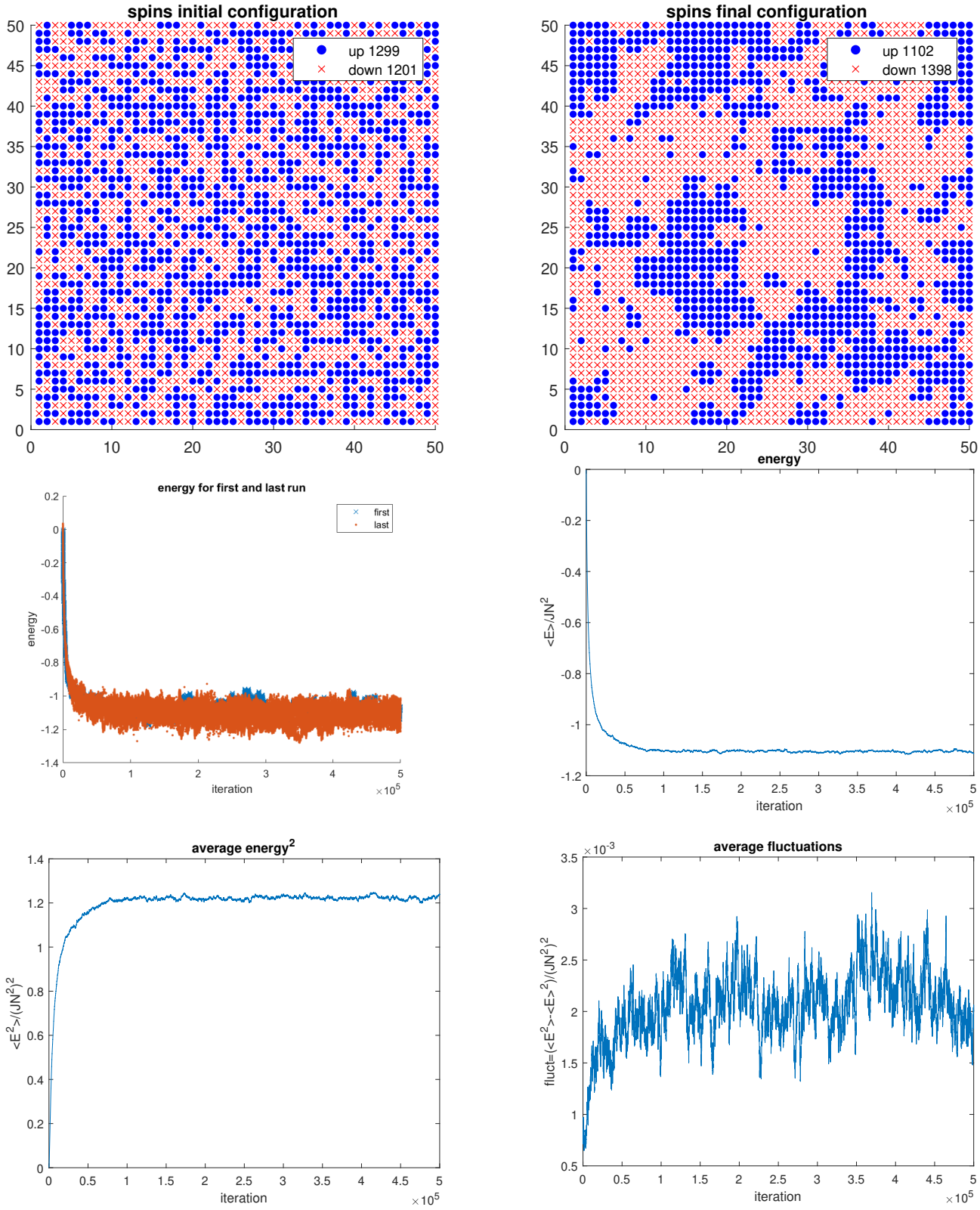

Figure 8: Initial and final spin configurations, average energy for a specific run, average energy over all runs, average n squared, and average fluctuations for J=0.4

As it can be seen, the fluctuations depend significantly on J, as J increases the fluctuations also increase. However, since the average energy also increases for increasing J, the fluctuations have less influence over the value of the energy. Thus, the fluctuations become less noticeable on the energy vs iteration plot for larger J. Meanwhile, the fluctuations do not depend on the iteration number n. For the case $J = 0.4$, there is a small variation of the fluctuations with n, but this is due to the magnetization not being equilibrated yet.

(b) **Fit your simulation result to a single exponential curve, $< E_n >= Ae^{-t/\tau_e} + B$, with τ_e the equilibration 'time' (use A, B, and τ_e as fitting parameters). What are the values of τ_e you find? Are they constant, or do they vary with J?**

For J=0.1, $\tau_e = 894.8$ as shown in figure 9. The resulting fit parameters are:

```
General model:
     f(x) = a*exp(-x/b)+c
Coefficients (with 95% confidence bounds):
       a =      0.1949  (0.1945, 0.1953)
       b =       894.8  (892.3, 897.4)
       c =     -0.2034  (-0.2034, -0.2034)
Goodness of fit:
  SSE: 4.44
  R-square: 0.7906
  Adjusted R-square: 0.7906
  RMSE: 0.00298
```

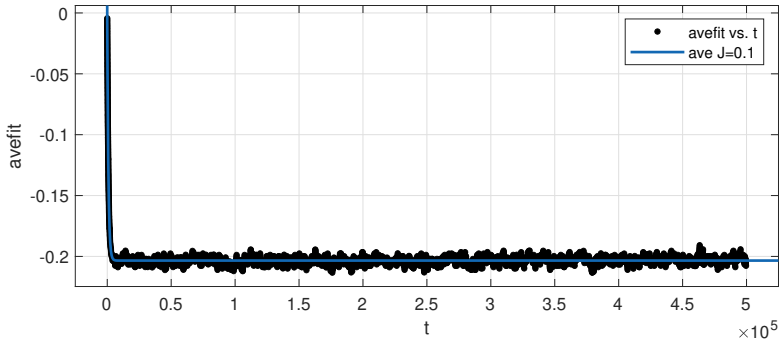

Figure 9: Fit for the average energy. Case J=0.1

For J=0.2, $\tau_e = 1444$ as shown in figure 10. The resulting fit parameters are:

```
General model:
     f(x) = a*exp(-x/b)+c
Coefficients (with 95% confidence bounds):
       a =      0.3839  (0.3836, 0.3843)
       b =        1444  (1443, 1446)
       c =      -0.428  (-0.428, -0.428)
Goodness of fit:
  SSE: 5.09
  R-square: 0.9575
  Adjusted R-square: 0.9575
  RMSE: 0.003191
```

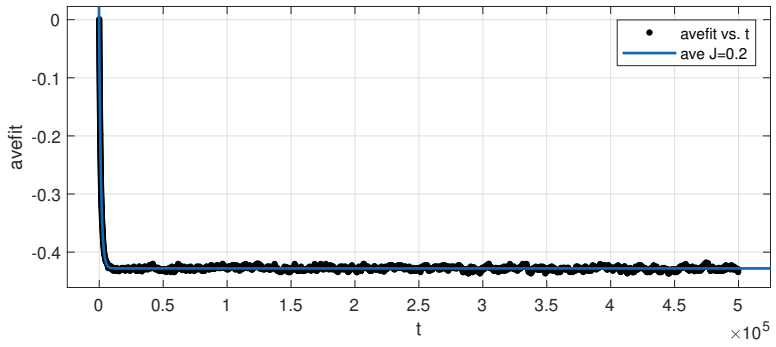

Figure 10: Fit for the average energy. Case J=0.2

For J=0.3, $\tau_e = 4069$ as shown in figure 11. The resulting fit parameters are:

```
General model:
     f(x) = a*exp(-x/b)+c
Coefficients (with 95% confidence bounds):
       a =      0.3227  (0.3225, 0.3229)
       b =        4069  (4065, 4073)
       c =     -0.7041  (-0.7041, -0.7041)
Goodness of fit:
  SSE: 7.713
  R-square: 0.9844
  Adjusted R-square: 0.9844
  RMSE: 0.003928
```

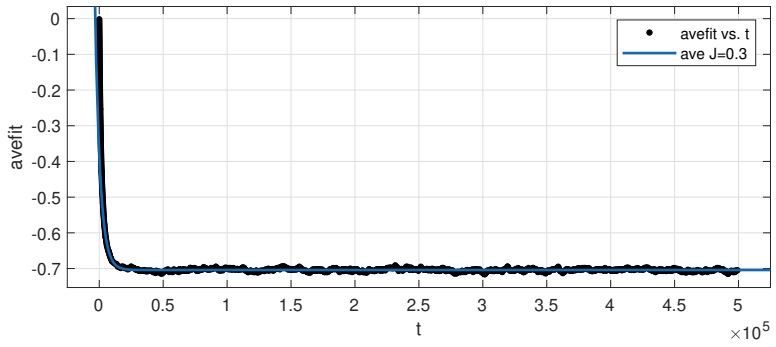

Figure 11: Fit for the average energy. Case J=0.3

Finally, for J=0.4, $\tau_e = 7471$ as shown in figure 12. The resulting fit parameters are:

```
General model:
     f(x) = a*exp(-x/b)+c
Coefficients (with 95% confidence bounds):
       a =      0.7371  (0.7369, 0.7373)
       b =        7471  (7468, 7474)
       c =      -1.104  (-1.104, -1.104)
Goodness of fit:
  SSE: 8.584
  R-square: 0.9963
```

```
    Adjusted R-square: 0.9963
    RMSE: 0.004143
```

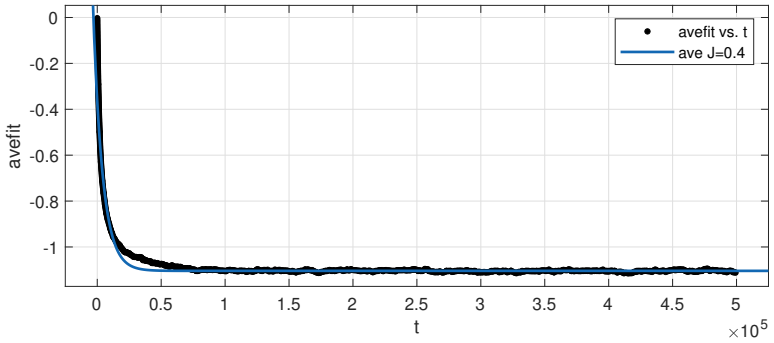

Figure 12: Fit for the average energy. Case J=0.4

As it can be noticed, the value of τ_e increases with J, which means that whenever the coupling constant is greater, it takes more time for the system to reach equilibrium. This is completely sensible, since whenever we flip spins, they feel a stronger coupling to their neighbors, the fluctuations are expected to be greater and the system would take more time to equilibrate due to these fluctuations.

(c) **In (a), you should have found the single exponentially decay doesn't fit well for all of the curves. Does a bi-exponential fit, $< E_n >= Ae^{-n/m_1} + Be^{-n/m_2}$ better fit the data?**

The only case for which one gets a better fit is for $J = 0.4$. This can be noticed by comparing the figures, clearly, for small times, the exponential fit is not good enough. Still, we get a very good R^2 value due to the high number of points for larger values of the "time". Figure 13 shows the advantages of the new fit.

```
General model:
     f(x) = a1*exp(-x/b1)+c+a2*exp(-x/b2)
Coefficients (with 95% confidence bounds):
       a1 =      0.2205  (0.2202, 0.2208)
       a2 =      0.6954  (0.6951, 0.6957)
       b1 =   2.395e+04  (2.392e+04, 2.397e+04)
       b2 =        3462  (3460, 3465)
       c =      -1.105  (-1.105, -1.105)
Goodness of fit:
  SSE: 5.905
  R-square: 0.9975
  Adjusted R-square: 0.9975
  RMSE: 0.003436
```

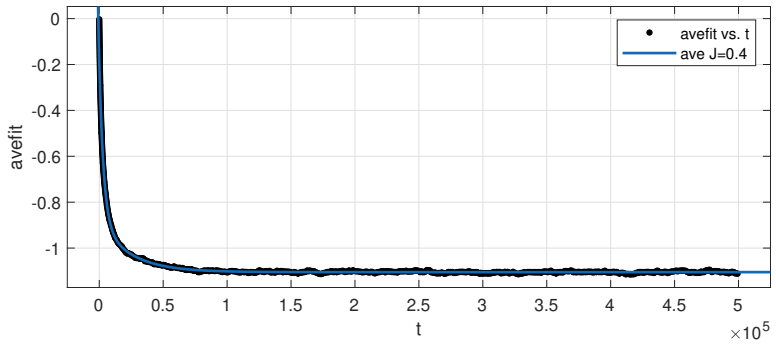

Figure 13: Bi-exponential fit for the average energy. Case J=0.4

(d) **In (b-c), you should have found equilibration on a timescale $\tau_e = 10^4$ for most of the simulations. For the same values of J, take $\tau_e = 10^4$ and compute $C(n) =< m_e m_{n+\tau_e} >$, the correlation between magnetization at two different times at equilibrium. Fit these to an exponential decay, $C(n) = De^{-n/\tau_C}$ with D and τ_C fitting parameters. Are the equilibration times from (b) the same as the correlation times you found here?**

No, the correlation times are different. The difference increases with increasing J.

```
General model:
     f(x) = a*exp(-x/b)+c
Coefficients (with 95% confidence bounds):
       a =      0.8599  (0.8458, 0.8739)
       b =         955  (933, 977.1)
       c =       1.104  (1.103, 1.104)
Goodness of fit:
  SSE: 5988
  R-square: 0.01024
  Adjusted R-square: 0.01023
  RMSE: 0.1105
```

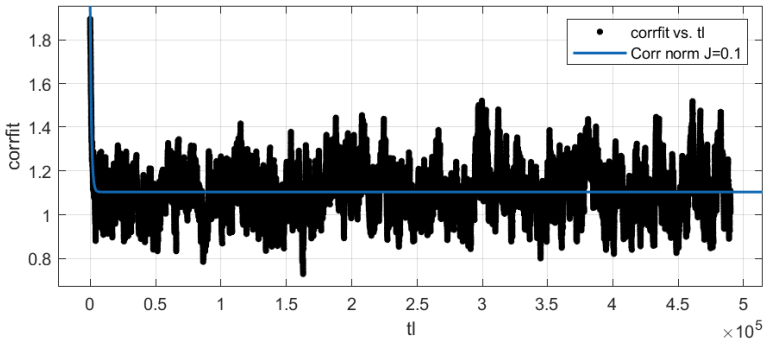

Figure 14: Fit for the correlation. Case J=0.1

```
J=0.2
General model:
     f(x) = a*exp(-x/b)+c
Coefficients (with 95% confidence bounds):
```

```
        a =       1.987  (1.967, 2.006)
        b =        1734  (1709, 1758)
        c =       2.203  (2.203, 2.204)
Goodness of fit:
  SSE: 2.101e+04
  R-square: 0.06847
  Adjusted R-square: 0.06846
  RMSE: 0.2071
```

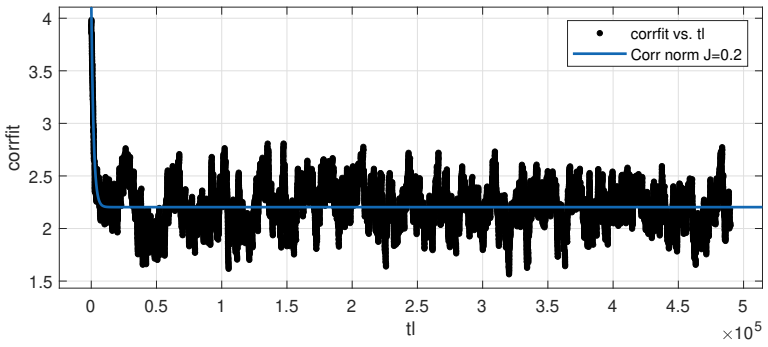

Figure 15: Fit for the correlation. Case J=0.2

```
J=0.3
General model:
     f(x) = a*exp(-x/b)+c
Coefficients (with 95% confidence bounds):
        a =       1.684  (1.675, 1.694)
        b =   2.804e+04  (2.779e+04, 2.829e+04)
        c =       4.552  (4.551, 4.553)

Goodness of fit:
  SSE: 8.836e+04
  R-square: 0.2643
  Adjusted R-square: 0.2643
  RMSE: 0.4247
```

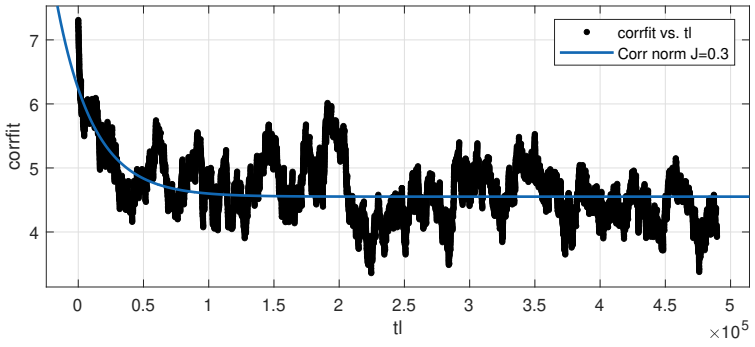

Figure 16: Fit for the correlations. Case J=0.3

For the case J=0.4, the correlations are clearly not exponential. Since, the magnetization has not equili-

brated after all iterations, it cannot be expected to get a meaningful correlation.

(e) **Compute the mean $< m >$ and standard deviation of the mean $\sigma_m = (< m^2 > - < m >^2)/\sqrt{n_{run}}$ for $J = 0.1, 0.2$, and 0.3 using the procedure described in class: discard the equilibration data and sampling the data at a rate inversely proportional to the correlation time.**

For $J = 0.1$, the resulting mean magnetization is $-4.9508e-04$ and the standard deviation is $8.3565e-05$. For $J = 0.2$, the mean magnetization is $-8.4828e-04$ and the standard deviation is $9.7203e-05$. Finally, for $J = 0.3$ the mean magnetization is -0.0015 and the standard deviation is $1.2602e-04$.