# 7 Dec 23 - Activity: The Ising Model

import numpy as np  
import matplotlib.pyplot as plt  
import random as random

---------------------------------------------------------------------------  
  
ModuleNotFoundError Traceback (most recent call last)  
  
/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb Cell 2 line 1  
----> <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W1sZmlsZQ%3D%3D?line=0'>1</a> import numpy as np  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W1sZmlsZQ%3D%3D?line=1'>2</a> import matplotlib.pyplot as plt  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W1sZmlsZQ%3D%3D?line=2'>3</a> import random as random  
  
  
ModuleNotFoundError: No module named 'numpy'

cellLength = 20  
simulationSteps = 1000000  
couplingConstant = 1.0 ## J  
temperature = 1.0  
  
def calculateEnergy(spinArray):  
 '''Calculate all the pairwise energy interactions and sum them up  
 Do rows and columns separately and add them up.'''  
   
 rowNeighborInteractionEnergy = np.sum(spinArray[0:cellLength-1,:]\*spinArray[1:cellLength,:])  
 columnNeighborInteractionEnergy = np.sum(spinArray[:,0:cellLength-1]\*spinArray[:,1:cellLength])  
   
 totalInteractionEnergy = rowNeighborInteractionEnergy+columnNeighborInteractionEnergy  
   
 return -couplingConstant\*totalInteractionEnergy  
  
## Create an empty square array  
spinArray = np.empty([cellLength,cellLength], int)  
  
## Populate it with random spins  
for row in range(cellLength):  
 for column in range(cellLength):  
 if random.random()<0.5:  
 spinArray[row,column] = +1  
 else:  
 spinArray[row,column] = -1  
  
# Calculate the initial energy and magnetization   
energyAtStep = calculateEnergy(spinArray)  
magnetizationAtStep = np.sum(spinArray)  
  
## Show the spin array   
## Black is spin up and white is spin down  
plt.figure(figsize=(8,8))  
c = plt.pcolor(spinArray, cmap='Greys')  
plt.axis('square')

---------------------------------------------------------------------------  
  
NameError Traceback (most recent call last)  
  
/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb Cell 3 line 1  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W2sZmlsZQ%3D%3D?line=14'>15</a> return -couplingConstant\*totalInteractionEnergy  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W2sZmlsZQ%3D%3D?line=16'>17</a> ## Create an empty square array  
---> <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W2sZmlsZQ%3D%3D?line=17'>18</a> spinArray = np.empty([cellLength,cellLength], int)  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W2sZmlsZQ%3D%3D?line=19'>20</a> ## Populate it with random spins  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W2sZmlsZQ%3D%3D?line=20'>21</a> for row in range(cellLength):  
  
  
NameError: name 'np' is not defined

## Hold onto the values of the magnetization   
## for each step in the simulation  
magnetizationArray = np.zeros(simulationSteps)  
  
## Monte Carlo Loop  
for step in range(simulationSteps):  
   
 ## Store the magnetization at this step  
 magnetizationArray[step] = magnetizationAtStep  
   
 ## Store the energy before swapping the spin randomly  
 oldEnergy = energyAtStep  
   
 ## Select a spin from the cell  
 ithSpin = random.randrange(cellLength)  
 jthSpin = random.randrange(cellLength)  
   
 ## Flip the spin of that one site  
 spinArray[ithSpin,jthSpin] = -spinArray[ithSpin,jthSpin]  
   
 ## Calculate the energy after that change  
 energyAtStep = calculateEnergy(spinArray)  
 deltaE = energyAtStep - oldEnergy  
   
 ## If the change resulted in an increase in the total energy,  
 ## evaluate whether to accept the value or not  
 if deltaE > 0.0:  
   
 probabilityOfFlip = np.exp(-deltaE/temperature)  
   
 ## If the the random value is lower than the probability,  
 ## reverse the change to the spin, and recalculate the energy  
 if random.random()>probabilityOfFlip:  
   
 spinArray[ithSpin,jthSpin] = -spinArray[ithSpin,jthSpin]  
 energyAtStep = oldEnergy  
 continue  
   
 magnetizationAtStep = np.sum(spinArray)

---------------------------------------------------------------------------  
  
NameError Traceback (most recent call last)  
  
/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb Cell 4 line 3  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W3sZmlsZQ%3D%3D?line=0'>1</a> ## Hold onto the values of the magnetization   
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W3sZmlsZQ%3D%3D?line=1'>2</a> ## for each step in the simulation  
----> <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W3sZmlsZQ%3D%3D?line=2'>3</a> magnetizationArray = np.zeros(simulationSteps)  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W3sZmlsZQ%3D%3D?line=4'>5</a> ## Monte Carlo Loop  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W3sZmlsZQ%3D%3D?line=5'>6</a> for step in range(simulationSteps):  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W3sZmlsZQ%3D%3D?line=6'>7</a>   
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W3sZmlsZQ%3D%3D?line=7'>8</a> ## Store the magnetization at this step  
  
  
NameError: name 'np' is not defined

plt.figure(figsize=(8,8));  
c = plt.pcolor(spinArray, cmap='Greys');  
plt.axis('square');

---------------------------------------------------------------------------  
  
NameError Traceback (most recent call last)  
  
/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb Cell 5 line 1  
----> <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W4sZmlsZQ%3D%3D?line=0'>1</a> plt.figure(figsize=(8,8));  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W4sZmlsZQ%3D%3D?line=1'>2</a> c = plt.pcolor(spinArray, cmap='Greys');  
 <a href='vscode-notebook-cell:/Users/caballero/repos/teaching/phy415fall23/content/4\_distributions/activity-ising\_model.ipynb#W4sZmlsZQ%3D%3D?line=2'>3</a> plt.axis('square');  
  
  
NameError: name 'plt' is not defined

plt.figure(figsize=(8,6))

plt.plot(magnetizationArray) plt.ylabel(‘Magnetization’) plt.xlabel(‘Simulation Steps’)