Queens College, CUNY, Department of Computer Science
Software Engineering
CSCI 370
Spring 2019
Instructor: Dr. Sateesh Mane

***** UPDATE *** due Friday Feb. 22, 2019**

1 Project 1

- **All students work individually for this project, there are no teams.**
- This document describes a mathematical calculation involving a lot of computation.
- *This project does not require a GUI or a database.*
- This project requires statistical sampling using pseudorandom numbers.
- To reduce the overall computation time, the application should perform parallel processing.
- You are responsible for configuring how your application implements parallel processing.
- You are responsible to design your program code to perform the computations in parallel.
- **Your code will be tested by running it on the Mars server.**

1.1 Project report

- Submit your solution in a zip archive with the following name.

  `StudentId_first_last_CS370_project1_Spring2019.zip`

- **Your project zip archive must contain all your program source code.**
- I will attempt to compile and run your program on the Mars server.
  A failing grade will be awarded if your code does not compile and run successfully.
- Your project report must state the values of N_M and N_f to answer Sec. 1.9.
- Your project report must state the values of B and C to answer all challenges that you solve.
- Your project report must contain screenshots/graphs to answer all challenges that you solve.
- For challenge #4, your project report must state the highest peak value of $\langle m \rangle$.
- For challenge #5, your project report must state the lowest minimum value of $\langle c_p \rangle$.

1.2 Introduction

- We shall implement a version of the Metropolis algorithm for simulated annealing.
- The Metropolis algorithm is a technique to find good solutions for hard problems such as the traveling salesman problem.
- We have a large set of 'nodes' and we wish to minimize some function of those nodes.
- In formal computer science we might speak of a graph instead, but I shall say nodes.
- The problem is hard because we must search a large number of configurations of the nodes.
- Suppose there are n nodes $(x_1, \ldots, x_n)$.
- Label a configuration of the nodes by $\boldsymbol{z}$ and the function by $f(\boldsymbol{z})$.
- We wish to find the optimum configuration $\boldsymbol{z}_{\text{opt}}$ which minimizes f.
- The Metropolis algorithm proceeds as follows.

1. Set a probability parameter p where $0 \leq p \leq 1$. This will be explained below.
2. Begin by guessing a candidate configuration $\boldsymbol{z}_0$ and compute the value of $f(\boldsymbol{z}_0)$.
3. Select a node x_i at random and change its value, to get a new configuration $\boldsymbol{z}_1$.
   (a) If $f(\boldsymbol{z}_1) \leq f(\boldsymbol{z}_0)$, replace $\boldsymbol{z}_0$ by $\boldsymbol{z}_1$ as our candidate configuration.
   (b) If $f(\boldsymbol{z}_1) > f(\boldsymbol{z}_0)$, ***accept*** $\boldsymbol{z}_1$ ***with probability*** p**.**
   (c) That is to say, generate a random number $0 \leq r < 1$ and accept $\boldsymbol{z}_1$ if $r \leq p$.
4. **Therefore the Metropolis algorithm is not greedy.**
   (a) The Metropolis algorithm always accepts a change which reduces the value of $f(\boldsymbol{z})$ and *sometimes* accepts a change which increases the value of $f(\boldsymbol{z})$.
   (b) Hence the Metropolis algorithm does not get trapped in a false local minimum.
5. Repeat and iterate the above procedure to find new configurations $\boldsymbol{z}_2$, $\boldsymbol{z}_3$, etc.
6. **When does the algorithm terminate?**
   (a) There is no fixed rule: *we have to make a decision when to stop the iteration.*
   (b) In practice we have to be satisfied by a final configuration $\boldsymbol{z}_f$ which we believe is sufficiently close to the optimum for our purposes.
7. **The value of p can change during the iterations.**
   (a) Initially we might set p to a high value, to explore a large area of the configuration space to find good candidates.
   (b) As the algorithm settles down to a good candidate, we decrease the value of p to reduce the probability of leaving the good region of the configuration space.
   (c) Hence the application of the Metropolis algorithm to practical problems is an art, not a science.

1.3 Some quantum physics . . . *don't panic*

- **These are definitions, nothing to compute here.**
- We shall calculate some thermodynamic properties of a set of quantum spins.
  1. **A quantum spin s_i is a variable which can take only two values $s_i = \pm 1$.**
  2. There are n quantum spins s_i, $i = 1, \ldots, n$.
  3. The spins are arranged in a circle, so each spin has two neighbors (circular linked list).
  4. For $1 < i < n$, the neighbors of spin s_i are s_{i-1} and s_{i+1}.
  5. The neighbors of the first spin s_1 are s_n and s_2.
  6. The neighbors of the last spin s_n are s_{n-1} and s_1.
- Denote a configuration of the spins by $\boldsymbol{\sigma} = (s_1, \ldots, s_n)$.
- There are 2^n configurations $\boldsymbol{\sigma}$, from $(1, \ldots, 1)$ to $(-1, \ldots, -1)$.
- The **magnetization per spin** is given by the sum of spins in $\boldsymbol{\sigma}$.

$$m(\boldsymbol{\sigma}) = \frac{1}{n} \sum_{i=1}^{n} s_i \,. \tag{1.3.1}$$

- The **pair correlation per spin** is given by the sum of products of neighboring spins in $\boldsymbol{\sigma}$. (Remember $s_{n+1} = s_1$ in the last term, because of the circular list.)

$$c_p(\boldsymbol{\sigma}) = \frac{1}{n} \sum_{i=1}^{n} s_i s_{i+1} \,. \tag{1.3.2}$$

- Hence by construction $-1 \le m \le 1$ and $-1 \le c_p \le 1$.
- For convenience we shall always choose n to be an even number.
- Then the following should be obvious, but as I said, don't panic.
  1. When all the spins are up $s_i = +1$ then both $m = 1$ and $c_p = 1$.
  2. When all the spins are down $s_i = -1$ then $m = -1$.
  3. When all the spins are down $s_i = -1$ *then again* $c_p = 1$, because $(-1) \times (-1) = +1$ for all pairs.
  4. When the spins alternate in sign $+ - + - \ldots$ then $m = 0$.
  5. When the spins alternate in sign $+ - + - \ldots$ then $c_p = -1$ because for each pair we obtain either $(+1) \times (-1)$ or $(-1) \times (+1)$, which equals -1 for every pair.
- The reason I write c_p instead of just c is because there are other correlations, with a larger gap between the spins. What I have called c_p is c_1, with a gap of 1, but c_2, etc. also exist:

$$c_2 = \frac{1}{n} \sum_{i=1}^{n} s_i s_{i+2} \,, \qquad c_3 = \frac{1}{n} \sum_{i=1}^{n} s_i s_{i+3} \qquad \text{etc.} \tag{1.3.3}$$

- *We shall not study the other correlations c_2, etc. in this project.*

1.4 Thermodynamic definitions . . . *again, don't panic*

- **These are definitions, nothing to compute here.**
- **We wish to calculate the thermodynamic average values of both m and c_p.**
- The thermodynamic average values are denoted by $\langle m \rangle$ and $\langle c_p \rangle$.
- They are the weighted average values of m and c_p at a temperature T.
  1. Introduce parameters B and C.
  2. The energy of a configuration of spins $\boldsymbol{\sigma}$ is given as follows.

$$E(\boldsymbol{\sigma}) = -\sum_{i=1}^{n} \big(B s_i + C s_i s_{i+1} \big) \,. \tag{1.4.1}$$

  3. The energy has a linear term s_i and a quadratic term of nearest-neighbor pairs $s_i s_{i+1}$.
  4. The weight factor associated with the configuration $\boldsymbol{\sigma}$ is $e^{-E(\boldsymbol{\sigma})/T}$.
  5. Hence the averages for $\langle m \rangle$ and $\langle c_p \rangle$ are given by the following weighted sums:

$$\langle m \rangle = \sum_{\boldsymbol{\sigma}} m(\boldsymbol{\sigma})\, e^{-E(\boldsymbol{\sigma})/T} \Big/ (\text{sum of weight factors}) \,, \tag{1.4.2a}$$

$$\langle c_p \rangle = \sum_{\boldsymbol{\sigma}} c_p(\boldsymbol{\sigma})\, e^{-E(\boldsymbol{\sigma})/T} \Big/ (\text{sum of weight factors}) \,. \tag{1.4.2b}$$

- One way to compute the averages is by brute force.
  1. We sum over all the configurations of the spins $\boldsymbol{\sigma}$ and compute $\langle m \rangle$ and $\langle c_p \rangle$.
  2. There are totally 2^n configurations $\boldsymbol{\sigma}$ of the spins.
- Brute force will yield the exact theoretical answer.
- Brute force is possible for $n \lesssim 20$, but for $n = 100$ it is impractical.
- Instead we shall employ the Metropolis algorithm, which will yield an approximate answer.
- As with the traveling salesman problem, brute force will yield the exact answer, but the computation is too slow. Instead we employ a faster algorithm, but the answer is approximate.

1.5 Metropolis algorithm

- **Important: details of computation are given in this section.**
- We employ the Metropolis algorithm to compute $\langle m \rangle$ and $\langle c_p \rangle$ as follows.
- **Begin by setting the spins in an initial configuration $\boldsymbol{\sigma}_0$.**
  1. We shall call $\boldsymbol{\sigma}_0$ our current configuration.
  2. It will be discussed later how to select a 'good' choice for $\boldsymbol{\sigma}_0$.
- **Choose an integer $1 \leq i \leq n$ using a uniform pseudorandom number distribution.**
  1. *Flip the spin s_i:* if $s_i = 1$, flip it to -1, else if $s_i = -1$, flip it to 1.
  2. This gives us a new configuration of spins $\boldsymbol{\sigma}_1$.
- Compute the energy difference using eq. (1.4.1): $\Delta E(\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1) = E(\boldsymbol{\sigma}_1) - E(\boldsymbol{\sigma}_0)$.
  1. *If you think carefully you will realize that the difference $\Delta E(\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1)$ can be computed efficiently without computing $E(\boldsymbol{\sigma}_1)$ and $E(\boldsymbol{\sigma}_0)$ separately.*
  2. The efficient calculation will also avoid problems of underflow/overflow.
  3. **Extra credit will be given to students who compute $\Delta E(\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1)$ efficiently.**
- Next we must make a decision whether or not to replace $\boldsymbol{\sigma}_0$ by $\boldsymbol{\sigma}_1$.
  1. If $\Delta E(\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1) < 0$ then replace $\boldsymbol{\sigma}_0$ by $\boldsymbol{\sigma}_1$ as our current configuration.
  2. If $\Delta E(\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1) > 0$ then compute $p = e^{-\Delta E(\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1)/T}$. Note that $0 < p < 1$.
  3. **Generate a pseudorandom number r using a uniform distribution $0 \leq r < 1$.**
  4. If $r < p$ then replace $\boldsymbol{\sigma}_0$ by $\boldsymbol{\sigma}_1$ as our current configuration.
  5. Else reject $\boldsymbol{\sigma}_1$ and retain $\boldsymbol{\sigma}_0$ as our current configuration.
- **Repeat the above process multiple times to update the current configuration.**
  1. As noted in Sec. 1.2, *when do we stop the iteration?*
  2. We would like each spin to have a chance of being flipped.
  3. Hence we should try at least n flips.
  4. **Set an integer parameter $N_f \geq 1$ and perform $n \times N_f$ flips.**
  5. Note that not all of the spin flips will result in an update of the current configuration.
- After completing all the flips, let the current configuration be $\boldsymbol{\sigma}_*$.
  1. **Compute the magnetization per spin using $\boldsymbol{\sigma}_*$ and eq.** (1.3.1)**.**
  2. **Compute the pair correlation per spin using $\boldsymbol{\sigma}_*$ and eq.** (1.3.2)**.**
  3. *These are our values for the thermodynamic averages $\langle m \rangle$ and $\langle c_p \rangle$.*
- Basically, instead of summing over all spin configurations (brute force), the Metropolis algorithm says the answer is the magnetization and pair correlation per spin of an 'optimal' configuration.

1.6 Initial spin configuration $\boldsymbol{\sigma}_0$

- **Use the following rule to set the initial spin configuration $\boldsymbol{\sigma}_0$.**
  1. **If $C \geq 0$, point all the spins up $s_i = 1$.**
  2. **If $C < 0$, alternate the signs of the spins $s_i = (-1)^i$.**
- Look at the expression for the energy in eq. (1.4.1).
  1. If $C > 0$, the spins like to be parallel $++++\ldots$ or $----\ldots$ because then $s_i s_{i+1} = 1$ and $C s_i s_{i+1} > 0$.
  2. If $C < 0$, the spins like to be antiparallel $+-+-\ldots$ or $-+-+\ldots$ because then $s_i s_{i+1} = -1$ and $C s_i s_{i+1} > 0$.
  3. The case $C = 0$ is ambiguous but we can go with parallel spins.
- Note also that there is no purpose in using $B < 0$.
  1. If we replace $B \to -B$, just flip all the spins $s_i \to -s_i$ and we obtain the same result.
  2. This is because $B s_i = (-B)(-s_i)$ and the product of spins is not affected if we flip all the spins: $s_i s_{i+1} = (-s_i)(-s_{i+1})$.
  3. Hence we shall use $B > 0$ in our calculations.
  4. Note that if $B > 0$ then the spins like up point up $s_i = 1$ because then $B s_i > 0$.
  5. If $B = 0$, the term in C will determine the spin configuration.
- Of course, if $B > 0$ and $C < 0$, the B term in the energy in eq. (1.4.1) would like all the spins to point up $++++\ldots$ but the C term would like the spins to alternate in sign $+-+-\ldots$, hence there is a competition between the two contributions to the energy.
- Do not worry, the Metropolis algorithm will deal with it.

1.7 Threads

- The values of n, B, C and T are given as inputs to the program.
- It is not enough to call to the Metropolis algorithm only once. We need better accuracy.
  1. **Write a main program (or calling application) to spawn N_t threads.**
  2. **In each thread, call the Metropolis algorithm N_M times (in a loop).**
  3. **Write a function to implement the Metropolis algorithm given in Sec. 1.5.**
- **Use $n = 100$ spins in all calculations.** This is too big for brute force.
- **Use $N_t = 1000$ threads.** I have found this is more or less optimal on the Mars server.
- The values of N_M and N_f (flips in the Metropolis algorithm) will be specified later.
- The output of the above calculations is as follows.
  1. Each thread returns a pair of arrays of lengrh N_M for $\langle m \rangle$ and $\langle c_p \rangle$.
  2. There are totally N_t threads, hence N_t such pairs of arrays.
- **Calculate the mean from each thread and the global mean μ_m of the $\langle m \rangle$ data.**

$$m[j] = \frac{1}{N_M} \sum_{k=1}^{N_M} \langle m \rangle_k \qquad \text{for each thread } j = 1, \ldots, N_t \tag{1.7.1a}$$

$$\mu_m = \frac{1}{N_t} \sum_{j=1}^{N_t} m[j] \qquad \text{(sum over all threads)}\,. \tag{1.7.1b}$$

- **Calculate the mean from each thread and the global mean μ_c of the $\langle c_p \rangle$ data.**

$$c[j] = \frac{1}{N_M} \sum_{k=1}^{N_M} \langle c_p \rangle_k \qquad \text{for each thread } j = 1, \ldots, N_t \tag{1.7.2a}$$

$$\mu_c = \frac{1}{N_t} \sum_{j=1}^{N_t} c[j] \qquad \text{(sum over all threads)}\,. \tag{1.7.2b}$$

- However, although this gives us values for μ_m and μ_c by summing over all the data from all the threads, we do not really know how accurate our answers are.
  1. That is why I asked you to calculate the values $m[j]$ and $c[j]$ from each thread.
  2. Save the arrays $m[j]$ and $c[j]$.
  3. We shall compute standard deviations using the arrays $m[j]$ and $c[j]$.
  4. This is our statistical error analysis, and it is necessary.
  5. The calculation is slighly complicated and will be described in Sec. 1.9.
- ***Do you understand?***

1.8 Sample graphs

- The theoretical formulas for both $\langle m \rangle$ and $\langle c_p \rangle$ are known.

  1. For $C = 0$, the formulas are as follows.

$$\langle m \rangle_{\text{theory}} = \frac{e^{B/T} - e^{-B/T}}{e^{B/T} + e^{-B/T}}\,, \qquad \langle c_p \rangle_{\text{theory}} = \langle m \rangle^2_{\text{theory}}\,. \tag{1.8.1}$$

  2. For $B = 0$, the formulas are as follows.

$$\langle m \rangle_{\text{theory}} = 0\,, \qquad \langle c_p \rangle_{\text{theory}} = \frac{e^{C/T} - e^{-C/T}}{e^{C/T} + e^{-C/T}}\,. \tag{1.8.2}$$

- I used $n = 100$ spins and set $B = 1$ and $C = 0$ and plotted the values of $\langle m \rangle$ and $\langle c_p \rangle$ (actually the averages μ_m and μ_c from Sec. 1.7) as functions of the temperatute T in Fig. 1. The curves were computed using eq. (1.8.1).

- I used $n = 100$ spins and set $B = 0$ and $C = -1$ and plotted the value $\langle c_p \rangle$ (actually the average μ_c from Sec. 1.7) as a function of the temperatute T in Fig. 2. The value of $\langle m \rangle$ is zero in this case and the simulation results for μ_m are very close to zero. The curve was computed using eq. (1.8.2).

- In both Figs. 1 and 2, I plotted ten points $T = 0.1, 0.3, \ldots, 1.9$ in steps $\Delta T = 0.2$.

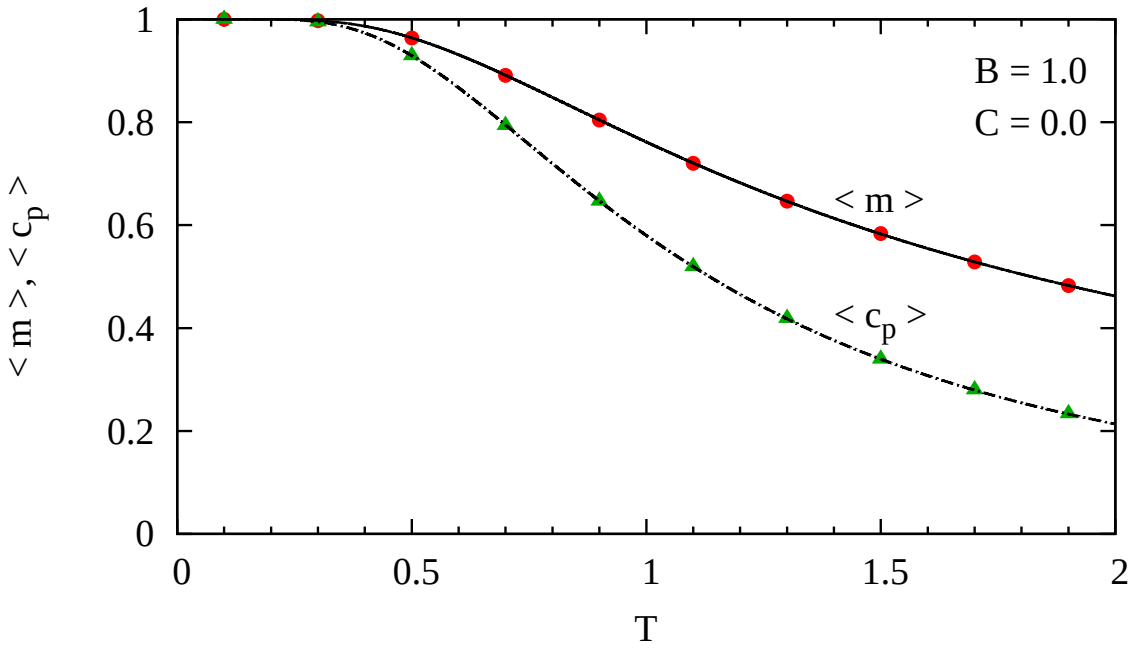

Figure 1: Average magnetization $\langle m \rangle$ (circles) and pair correlation $\langle c_p \rangle$ (triangles) per spin as a function of the temperature T. The points are from simulation data and the curves are the theoretical values.

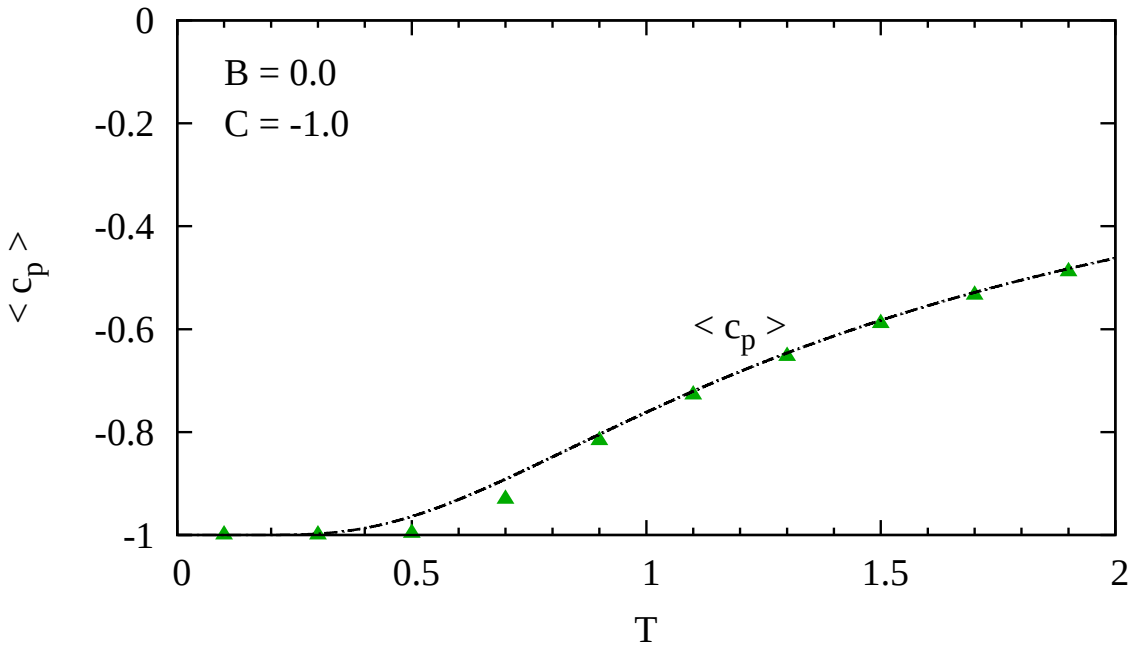

Figure 2: Average pair correlation $\langle c_p \rangle$ (triangles) per spin as a function of the temperature T. The points are from simulation data and the curve is the theoretical value. The average magnetization per spin $\langle m \rangle$ is zero in this case.

1.9 Values of N_M and N_f: statistical error analysis

- We shall use Fig. 2, i.e. the case $(B, C) = (0, -1)$, to find suitable values for N_M and N_f.
  1. Specifically, set $T = 1.9$ (the last point in Fig. 2).
  2. We shall find suitable values for N_M and N_f to fit that data point for $T = 1.9$.
- **Calculate the following.**
  1. For $T = 1.9$ and $(B, C) = (0, -1)$, use eq. (1.8.2) to calculate the value of $\langle c_p \rangle_{\text{theory}}$.
  2. Denote this value by c_p^*. It is a negative number.
  3. Compute the *relative error* r_c as follows. See eq. (1.7.2) and sum $c[j]$ over all threads.

$$r_c = \frac{1}{N_t} \sum_{j=1}^{N_t} \frac{c[j] - c_p^*}{c_p^*} \,. \tag{1.9.1}$$

  4. Compute the variance σ_c^2 of the relative error as follows.

$$\sigma_c^2 = \frac{1}{N_t} \sum_{j=1}^{N_t} \left(\frac{c[j] - c_p^*}{c_p^*} - r_c \right)^2 \,. \tag{1.9.2}$$

  5. Both the values of r_c and σ_c depend on the values of N_M and N_f.
- **Find values for N_M and N_f such that the following are satisfied.**

$$|r_c| \leq 0.02 \,, \qquad \sigma_c \leq 0.02 \,. \tag{1.9.3}$$

  1. **You can find values for N_M and N_f by any method you like (guesswork?).**
  2. Some hints are given below, **but you do not have to use them.**
  3. *As long as you find values for N_M and N_f which satisfy eq.* (1.9.3), that is good enough.
  4. **The solution for N_M and N_f is not unique.**
  5. **All valid solutions will be accepted.**
- Here is a suggestion how to find the values of N_M and N_f **but you do not have to use it.**
- **Set $T = 1.9$ and $B = 0$ and $C = -1$ below.**
- First let us find a suitable value of N_f to satisfy eq. (1.9.3).
  1. **Fill the following table to determine the value of N_f.**
  2. Set the value of N_M to some small number, say $N_m = 10$, to speed up the calculations.
  3. Don't set $N_M = 1$, too small.
  4. Then fill the following table until you obtain $|r_c| < 0.02$.

5. Denote the final value of N_f by N_f^*.

$T = 1.9$, $(B, C) = (0, -1)$, $N_M = 10$		
N_f	r_c	$\lvert r_c \rvert \times \sqrt{N_f}$
...	...	...
$\vdots$		
N_f^*	$\lvert r_c \rvert < 0.02$	...

6. If you have done your work correctly, you should find the numbers in the last column are approximately equal.
7. You should also find that if you change the value of N_M, maybe to 50, the results for r_c remain almost the same.

- The value of N_f^* tells the Metropolis algorithm how many flips to try to obtain a good answer.
- Next let us find a suitable value of N_M to satisfy eq. (1.9.3).

1. **Fill the following table to determine the value of N_M.**
2. Set the value of N_f to some small number, say $N_f = 10$, to speed up the calculations.
3. Don't set $N_f = 1$, too small.
4. Then fill the following table until you obtain $\sigma_c < 0.02$.
5. Denote the final value of N_M by N_M^*.

$T = 1.9$, $(B, C) = (0, -1)$, $N_f = 10$		
N_M	σ_c	$\sigma_c \times \sqrt{N_M}$
...	...	...
$\vdots$		
N_M^*	$\sigma_c < 0.02$	...

6. If you have done your work correctly, you should find the numbers in the last column are approximately equal.
7. You should also find that if you change the value of N_f, maybe to 50, the results for σ_c remain almost the same.

- If you have done your work correctly, you should find that if you plot ten points for $T = 0.1, 0.3, \ldots, 1.9$ using your values of N_M^* and N_f^*, your graph will closely match Fig. 2. The data points at $T = 0.5$ and $T = 0.7$ will not match the theory curve well. It is difficult to match those data points accurately; we will require much larger values for N_M^* and N_f^*.
- It is much easier to fit Fig. 1 than Fig. 2. If you have done your work correctly, you should find that if you use your values of N_M^* and N_f^*, you should also obtain a good match to Fig. 1. Remember that to fit Fig. 1 you must use the initial configuration $s_i = 1$ for the spins.

1.10 Challenges

- If $B > 0$ and $C > 0$ the graphs are boring. From eq. (1.4.1), if $B > 0$ then the spins want to be up $s_i = 1$ and if $C > 0$ the spins want to be parallel. Hence the combined effect is all the spins want to point up $s_i = 1$ and the results are boring.

- We obtain more interesting graphs if $B > 0$ and $C < 0$. From eq. (1.4.1), if $B > 0$ then the spins want to be up $s_i = 1$ but if $C < 0$, neighboring spins want to have opposite signs. This sets up a competition between the linear and quadratic terms in the energy in eq. (1.4.1), which leads to more interesting results.

    1. The graph in Fig. 3 used $B > 0$ and $C < 0$. The points are from simulation data. The curves are from theoretical formulas. The value of $\langle m \rangle$ equals zero at $T = 0$ and exhibits a peak at $T > 0$. *The peak is unique. The graph of* $\langle m \rangle$ *does not have multiple peaks.*
    2. The graph in Fig. 4 used *different values* $B > 0$ and $C < 0$. The points are from simulation data. The curves are from theoretical formulas. The value of $\langle c_p \rangle$ equals 1 at $T = 0$, *goes negative as* T *increases* and exhibits a minimum. *The minimum is unique. The value of* $\langle c_p \rangle$ *does not have multiple minima.*

- **Challenges:**

    1. **Find values $B > 0$ and $C < 0$ such that the graph of $\langle m \rangle$ equals zero at $T = 0$ and exhibits a peak at $T > 0$.**
    2. **Find values $B > 0$ and $C < 0$ such that the graph of $\langle c_p \rangle$ equals 1 at $T = 0$, goes negative as T increases and exhibits a minimum.**

- **For both cases, state your values of B and C and plot graphs as in Figs. 3 and 4.**

- There is more than one value of (B, C) which works, for each challenge.
  All valid solutions will be accepted.

- **Notes:**

    1. Use the values of N_M^* and N_f^* that you found in Sec. 1.9, for all the challenges.
    2. *For prelimimary scans, it is better to use smaller values of* N_M *and* N_f*, to speed up the calculations.* Once you have found 'good' values for B and C, then plot the graph using the values of N_M^* and N_f^* that you found in Sec. 1.9.
    3. **Use $0.01 < T \leq 2$ in your graphs.** Avoid wasting time using too large a range.
    4. You cannot set $T = 0$, the exponential $e^{-E(\boldsymbol{\sigma})/T}$ will exhibit overflow/underflow.

    5. **Plot 20 points in each graph,** $T = 0.01, 0.06, 0.11, \ldots$.
    6. **Use $0.1 \leq B \leq 2$ for all your calculations to answer the challenges.**
       All the challenges can be solved using values $0.1 \leq B \leq 2$.
    7. Given a value of B, you have to find a value of $C < 0$ which satisfies each challenge.
    8. *Because* $C < 0$*, remember to set the initial spin configuration* $\boldsymbol{\sigma}_0$ *to* $s_i = (-1)^i$.

1.11 More challenges

- *The challenges in this section may be very hard.*
- Do not waste time if you get stuck.
- **Extra credit will be given to students who solve the challenges in this section.**
- Look again at Figs. 3 and 4.
  1. Notice that in Fig. 3, $\langle m \rangle = 0$ and $\langle c_p \rangle = -1$ at $T = 0$.
  2. Notice that in Fig. 4, both $\langle m \rangle = \langle c_p \rangle = 1$ at $T = 0$.
- **Challenge #3:**
  Find values for $B > 0$ and $C < 0$ such that $\langle m \rangle \neq 0$ and $\langle m \rangle \neq 1$ at $T = 0$, also $\langle c_p \rangle \neq 1$ and $\langle c_p \rangle \neq -1$ at $T = 0$, i.e. $0 < \langle m \rangle < 1$ and $-1 < \langle c_p \rangle < 1$ at $T = 0$.
- In practice you cannot set $T = 0$, but try $T = 0.01$ (or $T = 0.001$ if you can manage that).
- **Challenge #4:**
  (Extension of Challenge #1.) Find values for $B > 0$ and $C < 0$ such that $\langle m \rangle = 0$ at $T = 0$ and $\langle m \rangle$ has a peak at $T > 0$.
  What is the highest peak value you can obtain for $\langle m \rangle$?
- **Challenge #5:**
  (Extension of Challenge #2.) Find values for $B > 0$ and $C < 0$ such that $\langle c_p \rangle = 1$ at $T = 0$ and $\langle c_p \rangle$ goes negative and has a minimum for $T > 0$.
  What is the lowest minimum (most negative value) you can obtain for $\langle c_p \rangle$?

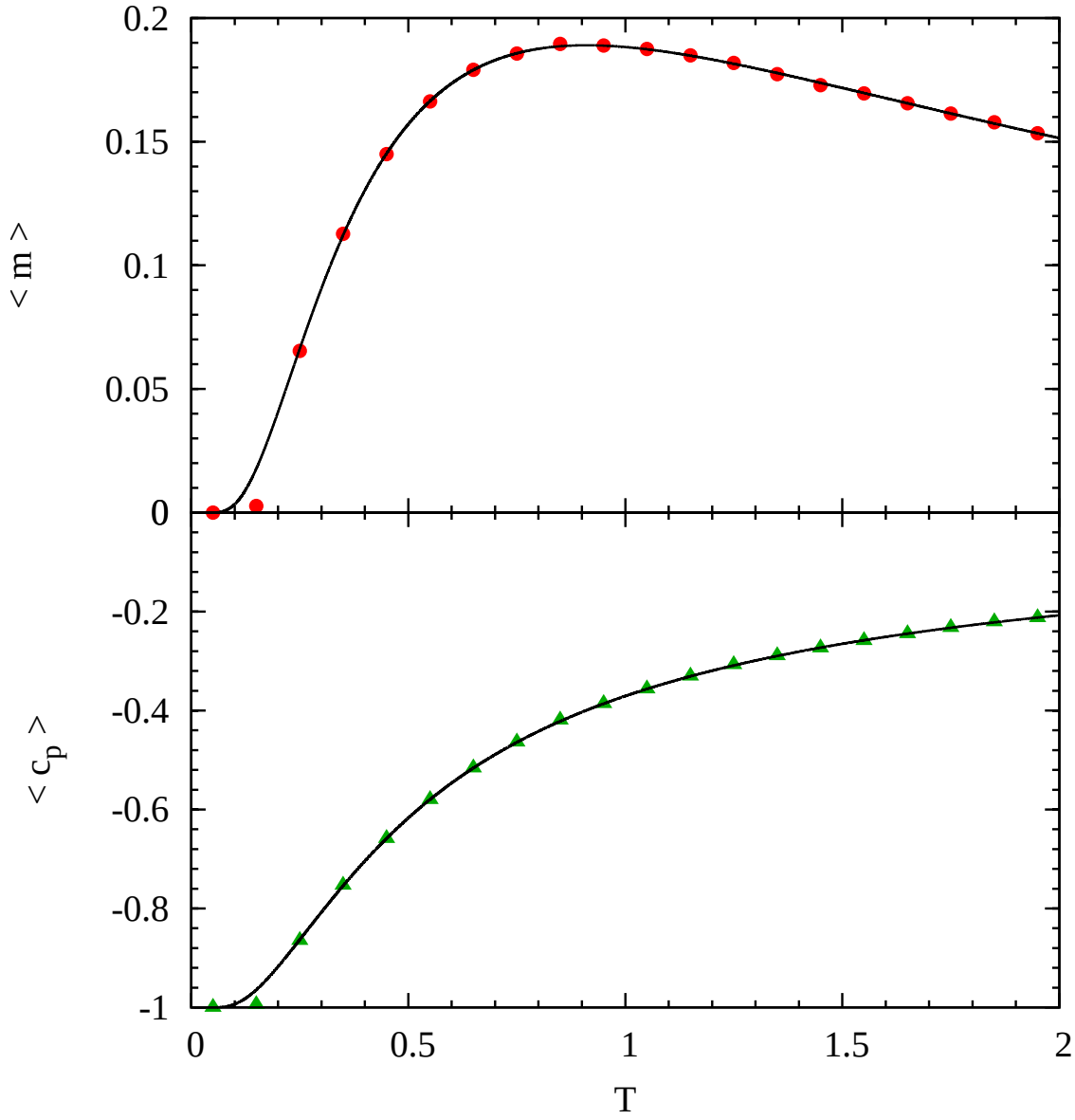

Figure 3: Average magnetization $\langle m \rangle$ (circles) and pair correlation $\langle c_p \rangle$ (triangles) per spin as a function of the temperature T. The points are from simulation data and the curves are from theoretical formulas. The value of $\langle m \rangle$ equals zero at $T = 0$ and exhibits a peak.

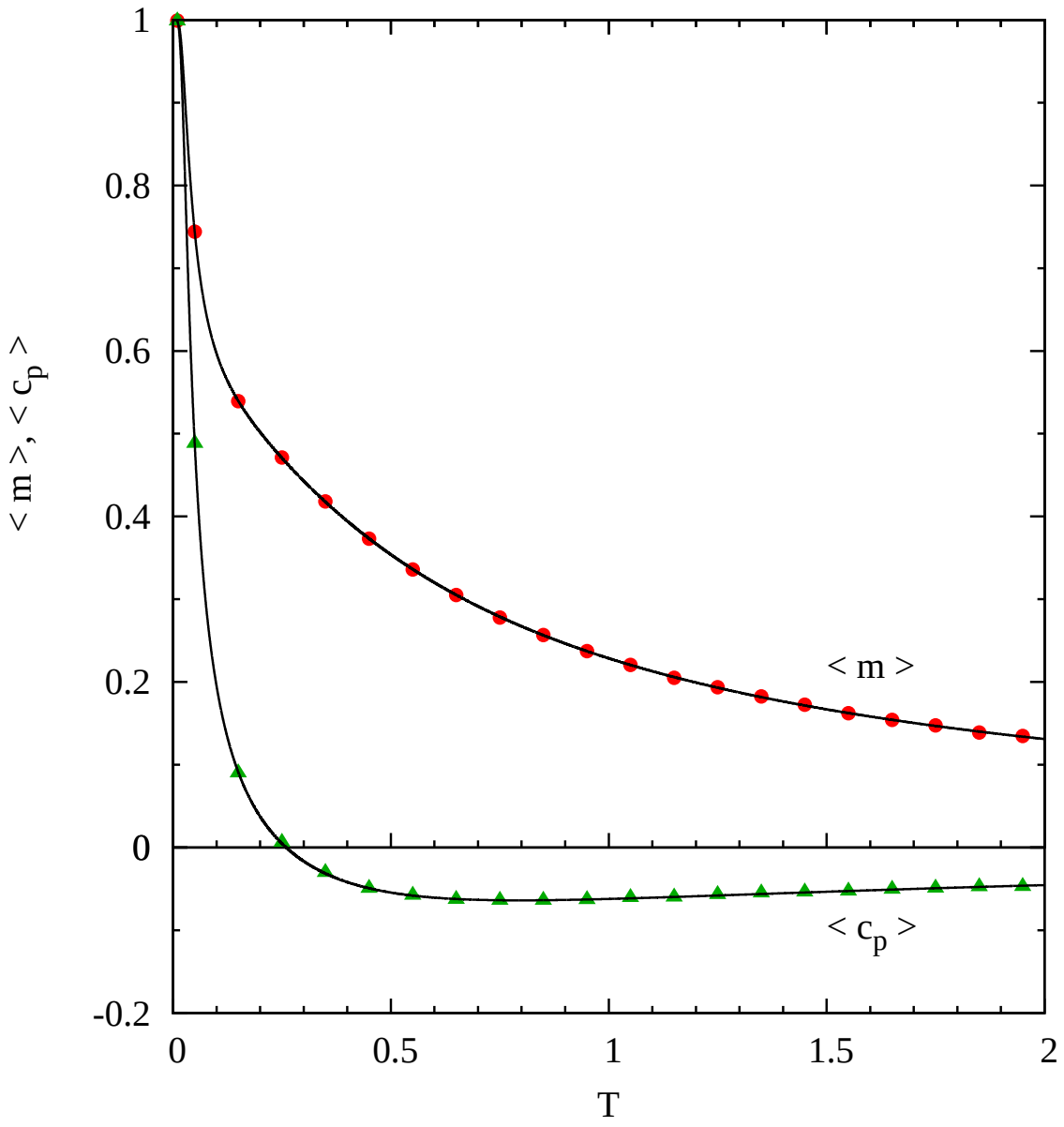

Figure 4: Average pair correlation $\langle c_p \rangle$ (triangles) per spin as a function of the temperature T. The points are from simulation data and the curves are from theoretical formulas. The value of $\langle c_p \rangle$ equals 1 at $T = 0$, goes negative as T increases and exhibits a minimum.