24-623/12-623 2015 HW#5

Total points: 40
Assigned: November 2, 2015.
Due: November 12, 2015, midnight to Blackboard. Please use the Blackboard discussion board to ask questions of the instructor or the other students.

1. (15 points) If we define $f(x, y) = 1$ for $x^2 + y^2 \leq 1$ and $f(x, y) = 0$ for $x^2 + y^2 > 1$, then

$$\int_{-1}^{1} \int_{-1}^{1} f(x, y) dx dy = \pi.$$

Write a C++ program to compute π by evaluating this double integral using random Monte Carlo sampling. Use $N = 10$, 100, 1000, and 10,000 samples. In each case, run your code five times and report the average value of π and the estimated uncertainty. Based on your results, predict how many samples you would need to compute π to twenty significant digits.

2. (25 points) In this problem, you will use Monte Carlo simulation to study the properties of the single oscillators from HW#2, which had potential energies (i) $U = x^2/2$ and (ii) $U = x^4 - 2x^2 + 1$.

(a) Determine $\langle U \rangle$, $\langle x \rangle$, and $\langle x^2 \rangle$ for $\beta = 0.1$, 1, 5, and 10 for oscillator (i) by directly evaluating the appropriate integrals (either analytically or numerically). Write a C++ code that uses Metropolis Monte Carlo to compute $\langle U \rangle$, $\langle x \rangle$, and $\langle x^2 \rangle$ for $\beta = 0.1$, 1, 5, and 10 for (i) and compare to the direct predictions. Describe how you debugged your code and how you chose the parameters you used to collect your data (e.g., maximum step size, number of trial moves). Compare your results to the behavior you observed in HW#2.

(b) Repeat part (a) for oscillator (ii). Compare your results to the behavior you observed in HW#2.

BONUS WORK:

(5 points) The Metropolis acceptance/rejection criterion is not unique. There are other acceptance/rejection criteria that can be used to perform Monte Carlo calculations in the NVT ensemble. Repeat 2.(a) by computing the acceptance probability for all moves as

$$acc(old \rightarrow new) = \frac{\exp(-\beta \delta E/2)}{\exp(-\beta \delta E/2) + \exp(\beta \delta E/2)},$$

where $\delta E = U(new) - U(old)$. Simulations performed using this criterion are called Kawasaki Monte Carlo. In addition to implementing Kawasaki Monte Carlo in your C++ code, show analytically that this criterion satisfies detailed balance.