Due Wednesday November 11th, 2020.

– – – – – – – – – – – – – – – ⋆ – – – – – – – – – – – – – –

1. **python etc**

   (a) Imagine that your home-made computer uses only 4 bits to represent signed integers, using the two's complement system. What is the largest (positive) integer and the smallest (negative) integer that could be stored in this computer?

   (b) Learn about the **reduce** command/function. (In python3, it is in the functools package.)

   Use reduce with a lambda expression (anonymous function) as the first argument and a list of numbers as the second argument, to produce

   (i) the sum of all the elements of the list.

   (ii) the largest element of the list.

   (Of course, each of these tasks can be done much more easily without reduce! The idea here is to get practice in designing anonymous functions.)

2. **Transition matrix for a Metropolis chain**

   Consider the 2-site Ising model

$$H = -J\sigma_1\sigma_2$$

where the two classical spins, σ_1 and σ_2, can each take values ± 1.

We are performing Markov chain Monte Carlo on this (admittedly toy) model for inverse temperature β. At every step, one proposes a flip of one of the two spins chosen at random. The state space consists of 4 configurations:

$$A = (+,+) \qquad B = (+,-) \qquad C = (-,+) \qquad D = (-,-)$$

where, e.g., $(+,-)$ means $\sigma_1 = +1$ and $\sigma_2 = -1$.

(a) Write down the 4×4 transition matrix, also known as the row stochastic matrix or the Markov matrix.

Hint 1: In the Metropolis process, a transition probability is the product of the proposal probability and the acceptance probability. Make sure to work out both, for each pair of states.

Hint 2: Each row should sum to 1.

(b) Based on the energies of the 4 configurations, what are the probabilities of these configurations in thermal equilibrium?

You might feel the urge to define $x = e^{-2\beta J}$.

(c) Form a row vector combining the probabilities in the stationary (thermal) state.

Check that this is the left eigenvector of your stochastic matrix, corresponding to eigenvalue 1.

3. **QR method for eigenvalues**

The eigenvalues of a matrix can be found by successive QR decompositions. In an earlier assignment, you showed that if A is QR-decomposed as $A = Q_1R_1$, then the matrix $A^{(1)} = R_1Q_1$ has the same eigenvalues as A. We could do this repeatedly: We define $A^{(0)} = A$ and the sequence of matrices $A^{(k)}$ as

$$\begin{aligned} A^{(k-1)} &= Q_kR_k \qquad &&\text{(QR decomp of } A^{(k-1)}\text{)} \\ A^{(k)} &= R_kQ_k \qquad &&\text{(inverting order: next matrix defined)} \end{aligned}$$

The $A^{(k)}$ each have the same eigenvalues. In addition, they have the property that successive $A^{(k)}$ are more and more diagonally dominant. If A is hermitian, for large k the iteration gives diagonal matrices. (If A is not hermitian, the large-k result might be instead a triangular matrix, which is good enough for obtaining eigenvalues.)

This is known as the QR method for obtaining eigenvalues.

We will explore the convergence of this method using numpy.linalg.qr.

Create a 10×10 matrix with all elements chosen from a random normal distribution. Symmetrize the matrix by adding its transpose to itself. Set up the iteration described above. At each step. record a measure of the diagonal dominance of the matrix. I suggest

$$D = \frac{\text{sum of squares of diagonal elements}}{\text{sum of squares of all elements}} = \frac{\sum_i |A_{ii}|^2}{\sum_i \sum_j |A_{ij}|^2}$$

which should converge to 1 as the matrix becomes more and more diagonal.

If you think of an alternate measure that you prefer, please define and explain carefully.

(a) Present beautiful plots of D (or your alternate measure) against iteration number.

(b) Present a program that performs this iteration and uses the value of D as a stopping criterion, i.e., stops only when $1 - D$ gets smaller than a tolerance.

(c) In the QR iteration, *why* do the matrices approach diagonal form?