# Stuff on a ring

In this assignment, we will investigate the application of Monte Carlo and PDE techniques on the same problem (essentially).

On a 1-dimensional ring of circumference $L$ (i.e., and interval of length $L$ with periodic boundary conditions), consider the following two cases:

1. A density field $P$ that satisfies the following partial differential equations:

$$\frac{\partial P}{\partial t} = D\frac{\partial^2 P}{\partial x^2}$$

   (the diffusion or heat equation), where $t$ is time, $x$ is the coordinate along the ring ($0 \leq x < L$) and $D$ is the diffusion constant.

   The initial condition is such that all density is concentrated at $x = 0$, i.e.,

$$P(x, t = 0) = \delta(x).$$

2. Alternatively, divide the ring up into segments of length $\Delta x$. $Z$ walkers are randomly hopping from segment to segments.

   At each time step, which is defined to take a time $\Delta t$, these walkers move to the next segments to the left or the right with probability $p$ (thus, the probability of staying is $1 - 2p$).

   Initially, all walkers are on segment $n = 0$.

To solve case 1 numerically, we will discretize time using a time step of the same size $\Delta t$ as in case 2, and we will discretize space using the same spacing $\Delta x$ as well. If we use forward Euler for the time step, and the numerical second derivate from the PDE lecture notes, and we equate $D = \frac{p\Delta x^2}{\Delta t}$, then it can be shown that these two cases become equivalent, in the sense that $P$ is the probablity distribution for the random walkers. We therefore expect

$$P(n\Delta x) = \lim_{Z\to\infty} \frac{a_n}{Z\Delta x}$$

Where $a_n$ is the number of walkers on segment $n$.

You are to test this equivalence in this assignment!

To get you started, you are given the codes for two applications, `diffring` and `walkring`, which provide a framework to solving the two cases above separately.

The codes are already modularized into the following files

| | |
|---|---|
| `diffring.cc` | Driver for the diffusion code |
| `diffring.ini` | Parameters file read in by the code |
| `diffring_parameters.h` | Module to read in the parameter file |
| `diffring_parameters.cc` | for the diffusion code |
| `diffring_timestep.h` | Module to do one diffusion time step |
| `diffring_timestep.cc` | |
| `diffring_output.h` | Module to do output |
| `diffring_output.cc` | |
| `walkring.cc` | Driver for the random walk code |
| `walkring.ini` | Parameters file read in by the code |
| `walkring_parameters.cc` | Module to read in the parameter file |
| `walkring_parameters.h` | for the random walkers |
| `walkring_timestep.cc` | Module to do one random walk time step |
| `walkring_timestep.h` | |
| `walkring_output.cc` | Module to do output |

| | |
|---|---|
| `walkring_output.h` | |
| `inifile.h` | Module for reading ini files |
| `ticktock.h` | Module for timing |
| `ticktock.cc` | |
| `sparkline.h` | Module to produce inline graphs |
| `sparkline.cc` | (uses unicode tricks) |
| `Makefile` | Combined makefile for the two applications |

Note that the code requires the following libraries:

- rarray
- boost
- openblas

There are a few pieces missing in `diffring_timestep.cc` and `walkring_timestep.cc`:

1. `diffring_timestep.cc` misses the initialization of the matrix $F$ that performs the time evolution (blas) matrix-vector multiplication in the `perform_time_step`. function. Note that most of the matrix is similar to the one shown on page 17–19 on the slides of lecture 14. To implement the periodic boundary conditions, the first and last row and the first and last column differ. In particular, if the number of grid points is $N$, one should make sure that one satifies:

$$P_{i+1}[N-1] = P_i[N-1] + \frac{D\Delta t}{\Delta x^2}\Big\{P_i[N-2] + P_i[0] - 2P_i[N-1]\Big\}$$

$$P_{i+1}[0] = P_i[0] + \frac{D\Delta t}{\Delta x^2}\Big\{P_i[N-1] + P_i[1] - 2P_i[0]\Big\}$$

   This should introducing non-zero elements `F[0][N-1]` and `F[N-1][0]`.

2. `diffring_timestep.cc` also misses the appropriate (blas) matrix-vector multiplication in the `perform_time_step` function.

3. `walkring.cc` misses the (random) dynamics of walkers on the ring in its `perform_time_step` function.

Your task is to

1. Provide the missing pieces (note that periodic boundary conditions imply the matrix for case 1 is not 'banded')

2. See how many walkers you need to get close to the equivalence.

3. Write a little report on what you did, which method you'd prefer and why.

Due date: March 24. Please include all relevant files in your submission, including the ones already provided. Putting all files in a zip or tar file is the easiest (do not use rar).