

Physics 420/580: Computational Physics

Final Exam

Tuesday, December 8, 2009
in class 14:00–15:20, in lab 15:30–17:00

Student's Name: _____

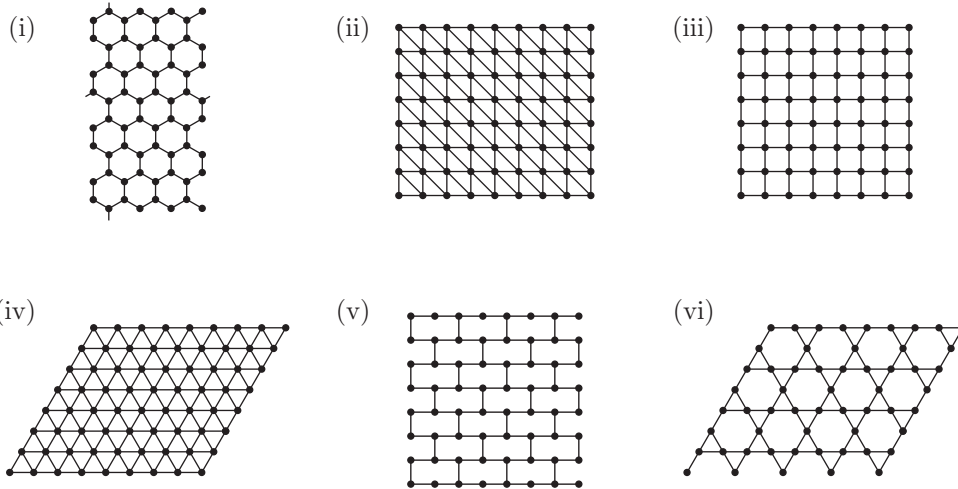
7 points	multiple choice	questions 1–7
23 points	long answer	questions 8–12
30 points	programming	questions 13–15

60 points

Multiple Choice Questions (7 points)

Answer by circling one of (a), (b), (c), etc. Please be clear about which one you have selected.

1. Which of the following lattices are topologically equivalent?



- (a) (ii) and (iii); (iv) and (vi)
- (b)** (i) and (v); (ii) and (iv)
- (c) (ii), (iii), and (v); (iv) and (vi)
- (d) all of them
- (e) none of them

Topologically equivalent lattices have the same connectivity, even if they don't have the same spatial arrangement or spatial symmetries. You can think of them as being continuously deformable one into the other. The honeycomb and brick-wall lattices are equivalent in this way; so are the triangular lattice and the square lattice with a nw-se connection across each plaquette.

2. In conventional mathematical notation, the matrix

$$A = \begin{pmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ \vdots & & & \vdots \\ A_{m,1} & A_{m,2} & \cdots & A_{m,n} \end{pmatrix}$$

has elements $A_{i,j}$ indexed by row $i = 1 \dots m$ and column $j = 1 \dots n$. Suppose that the elements are stored contiguously in memory in column-major order—i.e.,

$$\boxed{A_{1,1} | A_{2,1} | \dots | A_{m,1} | A_{1,2} | A_{2,2} | \dots | A_{m,2} | \dots | A_{1,n} | A_{2,n} | \dots | A_{m,n}}$$

—and you want to continue to refer to them using 1-based indexing. What is the correct replacement for \dots in the following program?

```

#include <cstdint>
using std::size_t;
#include <cmath>
using std::sqrt;

const size_t m = 7;
const size_t n = 4;
double A[m*n];
double frobenius_norm = 0.0;
for (size_t i = 1; i <= m; ++i)
    for (size_t j = 1; j <= n; ++j)
    {
        const double a = ... ;
        frobenius_norm += a*a;
    }
forbenius_norm = sqrt(forbenius_norm);

```

- (a) $A[i*n+j]$
- (b) $A[i+m*j]$
- (c) $A[(i-1)*n+(j-1)]$
- (d) $A[i+m*j-m-1]$

Column-major order with zero-based indexing is achieved with $i + mj$. Here, we simply offset i and j by one: $(i - 1) + m(j - 1) = i + mj - m - 1$.

3. A very accurate Runge-Kutta integrator (high order and small time step) is used to simulate a pendulum released from rest at angle $\theta = 50^\circ$. The pendulum is damped and subject to a strong, sinusoidal driving force whose frequency is close to the natural frequency of the pendulum. Two instances of the program are run, tracing out the pendulum trajectory from time $t = 0$ to $t = 20$. Program A runs to completion without interruption. Program B is stopped at $t = 10$, at which point it dumps its current configuration (3 digits of precision) to the file `snapshot.xml`.

```

$ cat snapshot.xml
<pendulum>
  <angle units="degrees"> 15.3 </angle>
  <angular_velocity units="degrees_per_second">
    -0.0138
  </angular_velocity>
</pendulum>

```

Program B is started up again with initial conditions read in from `snapshot.xml` and allowed to run from $t = 10$ to $t = 20$. How do the trajectories of A and B compare?

- (a) The two trajectories diverge beyond $t = 10$

- (b) Trajectory B experiences a slight hiccup at $t = 10$ but quickly reconverges on trajectory A
- (c) There is no difference whatsoever between the two trajectories

Restarting program B based on a configuration stored to only 3 digits precision introduces a very small perturbation to the trajectory. Since the behaviour of the damped, forced pendulum is chaotic, the trajectories will quickly diverge.

4. A field ϕ in three-dimensional space is restricted to a cubic grid of points, uniformly spaced by $\Delta x = \Delta y = \Delta z$: i.e., $\phi(i \cdot \Delta x, j \cdot \Delta x, k \cdot \Delta x) = \phi_{i,j,k}$. What is an appropriate finite-difference approximation to the Laplacian $\nabla^2\phi$?

- (a) $\frac{1}{(\Delta x)^3} [\phi_{i+1,j,k} + \phi_{i-1,j,k} + \phi_{i,j+1,k} + \phi_{i,j-1,k} + \phi_{i,j,k+1} + \phi_{i,j,k-1} - 3\phi_{i,j,k}]$
- (b) $\frac{1}{(\Delta x)^3} [\phi_{i+1,j,k} + \phi_{i-1,j,k} + \phi_{i,j+1,k} + \phi_{i,j-1,k} + \phi_{i,j,k+1} + \phi_{i,j,k-1} - 6\phi_{i,j,k}]$
- (c) $\frac{1}{(\Delta x)^2} [\phi_{i+1,j,k} + \phi_{i-1,j,k} + \phi_{i,j+1,k} + \phi_{i,j-1,k} + \phi_{i,j,k+1} + \phi_{i,j,k-1} - 6\phi_{i,j,k}]$
- (d) $\frac{1}{2\Delta x} [\phi_{i+1,j,k} - \phi_{i-1,j,k} + \phi_{i,j+1,k} - \phi_{i,j-1,k} + \phi_{i,j,k+1} - \phi_{i,j,k-1}]$

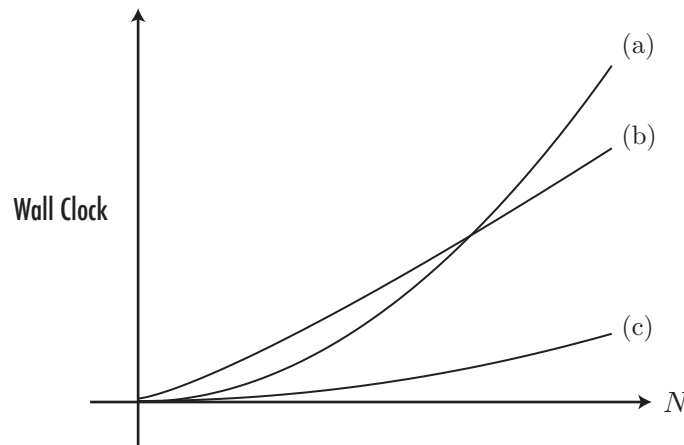
The second-order partial derivative has the three-point approximation

$$\frac{\partial^2 \phi}{\partial x^2} \approx \frac{1}{\Delta x} \left(\frac{\phi_{i+1,j,k} - \phi_{i,j,k}}{\Delta x} - \frac{\phi_{i,j,k} - \phi_{i-1,j,k}}{\Delta x} \right) = \frac{\phi_{i+1,j,k} + \phi_{i-1,j,k} - 2\phi_{i,j,k}}{\Delta x^2}$$

Thus, the Laplacian, which is just the sum $\partial^2\phi/\partial x^2 + \partial^2\phi/\partial y^2 + \partial^2\phi/\partial z^2$, is given by (c).

For the next two questions, answer by writing the corresponding letter in the spaces provided. Each of (a), (b), (c) [and (d) for question 6] should appear exactly once.

5. The execution time of a molecular dynamics simulation of a Lennard-Jones gas is plotted as a function of the number of particles N . Identify which curve belongs to which algorithm.



- (a) Standard algorithm running on one computer

- (c) Parallel algorithm running on five computers
- (b) Barnes-Hutt algorithm

A simulation of N particles interacting via long-range forces will scale as $N(N - 1)/2$, which is just the count of all possible particle pairs. The same algorithm running on five computers will scale (at best) as $N(N - 1)/10$. Since these two curves have the same N dependence and only a different prefactor, they will never cross. Thus, it must be that the parallel algorithm is (c) and the standard algorithm is either (a) or (b). Since Barnes-Hutt scales as $N \log N$, it must grow slower than the standard algorithm at large N . We conclude that the Barnes-Hutt curve is (b).

6. A sleep-deprived PHYS 420/580 student (working late on an assignment, of course) is trying to compute the kinetic energy $\frac{1}{2}mv^2$ of a point particle of mass $m = 5/2$ and velocity $v = 3$. Her four attempts yield the following code snippets. In each case, determine the value assigned to `kinetic_energy`.

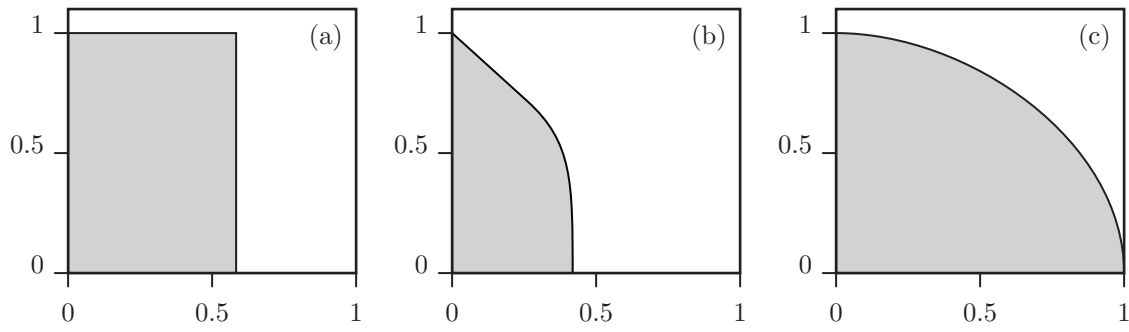
```
const double m = 2.5;
const double v = 3.0;
```

- (a) `const double kinetic_energy = 0.5*v*m*v;`
- (b) `const double kinetic_energy = (1/2)*m*v*v;`
- (c) `const double kinetic_energy = m*v^2/2.0;`
- (d) `#include <cmath>`
`using std::pow;`
`const double kinetic_energy = m*pow(2.0,v)/2;`

- (b) 0
- (d) 10.0
- (a) 11.25
- (c) Fails to compile

Version (a) gives the correct value, which is $1/2 \times 5/2 \times 3^2 = 45/4 = 11.25$. The integer division $1/2$ in (b) yields 0. Version (c) is syntactically wrong: there is no exponentiation operation in C/C++. The exponentiation *function* `pow(2.0,v)`, used in version (d), computes 2^v , not v^2 as the programmer intended.

7. Suppose that we have an $L \times L$ square lattice of cells, some fraction of which have been randomly removed. The set of cells connected to their neighbours by moves to the north, south, east, or west form a cluster. Which of the plots below correctly depicts the fraction belonging to the largest cluster as a function of the fraction of cells removed?



The percolation problem has a *continuous* transition at a nontrivial critical fraction $0 < p_c < 1$. In (a), the transition is first order; in (c), $p_c = 1$.

Long Answer Questions (23 points)

8. (3 points) The following array of binary cells has length $L = 24$.

0 0 0 0 0 0 1 0 0 0 1 1 0 1 1 0 0 0 0 1 0 0 0 0

(a) Perform a coarse-graining step $(b, b') \rightarrow (b + b')/2$ on pairs of cells to produce a new array of length $L/2 = 12$ and then again to produce an array of length $L/4 = 6$.

0 0 0 $\frac{1}{2}$ 0 1 $\frac{1}{2}$ $\frac{1}{2}$ 0 $\frac{1}{2}$ 0 0
 0 $\frac{1}{4}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{4}$ 0

(b) What is the complete set of possible cell values after n such coarse-graining steps.

$\{0, 1/2^n, 1/2^{n-1}, \dots, 1/2, 1\}$

(c) Implement a cellular automata rule that updates pairs of sites via $\boxed{1}\boxed{0} \leftrightarrow \boxed{0}\boxed{1}$ using a two-site Margolus scheme. Show me the evolution during the first five time steps.

0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	1	1	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	1	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	1	1	0	0	0	1	0	0	0	0	0	0

9. (4 points) Read over the following C++ class definition.

```
class rtnl
{
private:
    signed long int n;
    unsigned long int d;
public:
    rtnl(signed long int n_, unsigned long int d_) : n(n_), d(d_) {}
    double fp(void) const { return (long double)(n)/(long double)(d); }
    friend rtnl operator*(const rtnl &x, const rtnl &y)
    {
        return rtnl(x.n*y.n,x.d*y.d);
    }
    friend rtnl operator+(const rtnl &x, const rtnl &y)
    {
        return rtnl(x.n*y.d+x.d*y.n,x.d*y.d);
    }
};
```

- (a) Explain what number concept the `rtnl` class implements.

Rational numbers

- (b) Critique the implementation. Suggest several ways it might fail.

(i) The implementation assumes that the sign information is stored in the numerator only. The denominator is an `unsigned` type. The constructor calls `rtnl(-3,5)` and `rtnl(3,-5)` do not both produce $-3/5$. Calls to `operator+` may also break this sign convention and thus return an incorrect result. (ii) There is no special handling of the case when the denominator is zero. (iii) Operations on pairs of `rtnl` objects can lead to overflow of the `n` and `d` data members.

- (c) Suggest an obvious improvement that could be made to the `operator+` and `operator*` functions.

The common factors in the numerator and denominator of the temporary object should be cancelled before it is returned. This will stave off overflow problems.

- (d) Does an `rtnl` object have more or less numerical precision than the standard 32- and 64-bit floating point types (`float`, `double`)?

The `float` and `double` types retain 23 and 52 significant bits, respectively. This is standardized on all modern computer platforms. The size of a `long int`, on the other hand, isn't fixed by the C++ specification. Nonetheless, it's almost always 32 bits wide. So, an `rtnl` object has $32 + 31 = 63$ non-sign storage bits. But not all of them are significant, since the rational representation is many to one, e.g., $n/d = 2/2 = 5/5 = 57/57 = 1$. Also, unlike the floating-point types, a fixed-width rational does not retain its relative precision over many orders of magnitude. For a rough estimate, we can determine the smallest "machine epsilon" from $n/d = (d + 1)/d = 1 + 1/d = 1 + \epsilon$ by substituting the largest value of $d = n - 1$. This implies about 31 bits of precision—somewhere between a `float` and a `double`.

10. (5 points) Consider a system of $2N$ particles governed by the classical Hamiltonian

$$H = \sum_{i=1}^{2N} \frac{\mathbf{p}_i^2}{2m} + \sum_{i=1}^{2N} \sum_{j=1}^{2N} \frac{1}{2} (1 - \delta_{ij}) V(\mathbf{r}_i - \mathbf{r}_j) + \sum_{i=1}^N \frac{K}{2} (|\mathbf{r}_{2i} - \mathbf{r}_{2i-1}| - d)^2.$$

The pair potential $V(\mathbf{r}_i - \mathbf{r}_j) = U(r_{ij}) = U(-r_{ij})$ is a function of the distance between particles i and j and is mildly repulsive. In the last term, $d > 0$ is a small distance and $K \geq 0$ is a Hooke's law constant.

The corresponding classical equations of motion are

$$\begin{aligned} \dot{\mathbf{r}}_i &= \frac{\partial H}{\partial \mathbf{p}_i} = \frac{\mathbf{p}_i}{m} \\ \dot{\mathbf{p}}_i &= -\frac{\partial H}{\partial \mathbf{r}_i} = -\sum_{j \neq i} U'(r_{ij}) \hat{\mathbf{r}}_{ij} + K(|\mathbf{r}_{i+1} - \mathbf{r}_i| - d) \hat{\mathbf{r}}_{i+1,i} \quad (i \text{ odd}) \\ \dot{\mathbf{p}}_i &= -\frac{\partial H}{\partial \mathbf{r}_i} = -\sum_{j \neq i} U'(r_{ij}) \hat{\mathbf{r}}_{ij} - K(|\mathbf{r}_{i-1} - \mathbf{r}_i| - d) \hat{\mathbf{r}}_{i,i-1} \quad (i \text{ even}) \end{aligned}$$

where $\hat{\mathbf{r}}_{ij} = (\mathbf{r}_i - \mathbf{r}_j)/|\mathbf{r}_i - \mathbf{r}_j|$.

- (a) Argue that $m \frac{d^2}{dt^2}(\mathbf{r}_3 + \mathbf{r}_4)$ is independent of K while $m \frac{d^2}{dt^2}(\mathbf{r}_4 + \mathbf{r}_5)$ is not.

The mass times acceleration of a particle is

$$m\ddot{\mathbf{r}} = m \frac{d}{dt} \left(\frac{\mathbf{p}}{m} \right) = \dot{\mathbf{p}}.$$

Hence,

$$\begin{aligned} m \frac{d^2}{dt^2}(\mathbf{r}_3 + \mathbf{r}_4) &= \dot{\mathbf{p}}_3 + \dot{\mathbf{p}}_4 \\ &= -2 \sum_{j \neq i} U'(r_{ij}) \hat{\mathbf{r}}_{ij} + K(|\mathbf{r}_4 - \mathbf{r}_3| - d) \hat{\mathbf{r}}_{4,3} - K(|\mathbf{r}_3 - \mathbf{r}_4| - d) \hat{\mathbf{r}}_{4,3} \\ &= -2 \sum_{j \neq i} U'(r_{ij}) \hat{\mathbf{r}}_{ij} \end{aligned}$$

On the other hand,

$$\begin{aligned} m \frac{d^2}{dt^2}(\mathbf{r}_4 + \mathbf{r}_5) &= \dot{\mathbf{p}}_4 + \dot{\mathbf{p}}_5 \\ &= -2 \sum_{j \neq i} U'(r_{ij}) \hat{\mathbf{r}}_{ij} - K(|\mathbf{r}_3 - \mathbf{r}_4| - d) \hat{\mathbf{r}}_{4,3} + K(|\mathbf{r}_6 - \mathbf{r}_5| - d) \hat{\mathbf{r}}_{6,5} \end{aligned}$$

- (b) Explain how this system behaves as the parameter K is tuned from zero to infinity. What is unphysical about the small K limit?

$K = 0$ describes a system of $2N$ non-interacting particles. As K is ramped up, interactions arise between the particles pairs $(1, 2), (3, 4), \dots (2N - 1, 2N)$ that favour a pairwise spatial separation d . In the limit $K \rightarrow \infty$, the spring constant becomes infinitely stiff: the system behaves as a gas of N diatomic molecules, modelled as rigid dumbbells. The Hamiltonian doesn't make physical sense for small K , since each particle is interacting with only a single partner, which may be at much longer distances than neighbouring particles.

- (c) Use a time-scale argument to explain why it's so difficult to simulate this system numerically at very large K .

At large K , there are extremely fast oscillations (natural frequency $\omega = \sqrt{2K/m}$) of each diatomic pair around its equilibrium distance d . This completely dominates the choice of time step Δt , which will end up being much smaller than the time scale for motion of the molecular centres of mass.

- (d) Explain how a Lagrange multiplier trick can be used to simulate the $K = \infty$ limit efficiently.

The non-interacting Hamiltonian is augmented by a constraint term

$$H = H_0 - \sum_{i=1}^N \lambda_i (|\mathbf{r}_{2i} - \mathbf{r}_{2i-1}| - d)^2.$$

The equations of motion—which now include λ -dependent fictitious forces—are solved, ensuring that

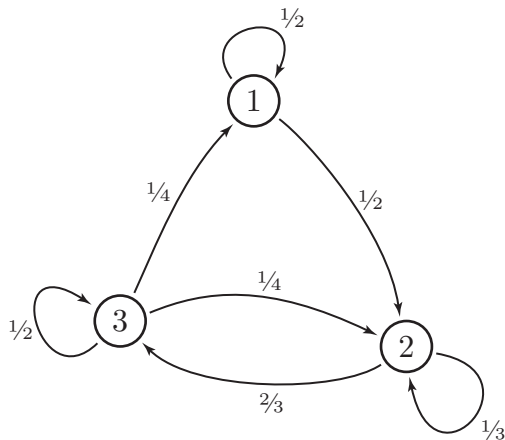
$$\frac{\partial H}{\partial \lambda_i} = (|\mathbf{r}_{2i} - \mathbf{r}_{2i-1}| - d)^2 = 0$$

at each time step.

11. (5 points) A Markov process on a three-state system is governed by the matrix of transition weights,

$$T = \begin{pmatrix} 1/2 & 0 & 1/4 \\ 1/2 & 1/3 & 1/4 \\ 0 & 2/3 & 1/2 \end{pmatrix}.$$

Each element T_{ij} represents the probability of moving from state j (at time step n) to state i (at time step $n + 1$).



We use $\pi_i^{(n)}$ to denote the chance of finding a walker in state i at time n . In the long-time limit, the random process settles into a steady state distribution

$$\pi^* = \lim_{n \rightarrow \infty} \pi^{(n)} = \lim_{n \rightarrow \infty} T^n \pi^{(0)} = \frac{1}{\sqrt{29}} \begin{pmatrix} 2 \\ 3 \\ 4 \end{pmatrix}.$$

- (a) What properties does T possess that make it valid transition matrix?

We have to be able to interpret the elements as probabilities: (i) $T_{ij} \geq 0$. (ii) $\sum_i T_{ij} = 1$ for each j .

- (b) A collection of random walkers is initialized at time $n = 0$ such that all the walkers are in state 1; i.e.,

$$\pi^{(0)} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}.$$

What fraction of the walkers are in state 3 two time steps later?

We compute $\pi_3^{(2)} = (T^2 \pi^{(0)})_3 = 1/3$ as follows.

$$\pi^{(1)} = T \pi^{(0)} = \begin{pmatrix} 1/2 & 0 & 1/4 \\ 1/2 & 1/3 & 1/4 \\ 0 & 2/3 & 1/2 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1/2 \\ 1/2 \\ 0 \end{pmatrix}$$

$$\pi^{(2)} = T\pi^{(1)} = \begin{pmatrix} 1/2 & 0 & 1/4 \\ 1/2 & 1/3 & 1/4 \\ 0 & 2/3 & 1/2 \end{pmatrix} \begin{pmatrix} 1/2 \\ 1/2 \\ 0 \end{pmatrix} = \begin{pmatrix} 1/4 \\ 5/12 \\ 1/3 \end{pmatrix}$$

(c) Show that T and

$$T' = \begin{pmatrix} 1/2 & 1/6 & 1/8 \\ 1/4 & 1/3 & 3/8 \\ 1/4 & 1/2 & 1/2 \end{pmatrix}$$

lead to the same steady state distribution. Which of T and T' satisfy detailed balance?

π^* is stationary under action by both T and T' .

$$T\pi^* = \frac{1}{\sqrt{29}} \begin{pmatrix} 1/2 & 0 & 1/4 \\ 1/2 & 1/3 & 1/4 \\ 0 & 2/3 & 1/2 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \\ 4 \end{pmatrix} = \frac{1}{\sqrt{29}} \begin{pmatrix} 2 \\ 3 \\ 4 \end{pmatrix} = \pi^*$$

$$T'\pi^* = \frac{1}{\sqrt{29}} \begin{pmatrix} 1/2 & 1/6 & 1/8 \\ 1/4 & 1/3 & 3/8 \\ 1/4 & 1/2 & 1/2 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \\ 4 \end{pmatrix} = \frac{1}{\sqrt{29}} \begin{pmatrix} 2 \\ 3 \\ 4 \end{pmatrix} = \pi^*$$

Detailed balance requires that the flow rate between every two states be equal in each direction. We want to check that $R_{ij} = T_{ij}\pi_j^*$ is the same as $R_{ji} = T_{ji}\pi_i^*$ or, in other words, that the matrix R is its own transpose. This does not hold for R derived from T , but it does for R' derived from T' .

$$R_{ij} = T_{ij}\pi_j^* = \frac{1}{\sqrt{29}} \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 2 & 2 \end{pmatrix}_{ij}$$

$$R'_{ij} = T'_{ij}\pi_j^* = \frac{1}{\sqrt{29}} \begin{pmatrix} 1 & 1/2 & 1/2 \\ 1/2 & 1 & 3/2 \\ 1/2 & 3/2 & 2 \end{pmatrix}_{ij}$$

(d) Find the steady state distribution for

$$T'' = \begin{pmatrix} 1/2 & 0 & 0 \\ 1/2 & 1/3 & 0 \\ 0 & 2/3 & 1 \end{pmatrix}.$$

Hint: the answer will be obvious if you draw a diagram analogous to the one shown at the beginning of question 11.

State 1 can transition to state 2; state 2 can transition to state 3; but state 3 remains fixed for all time. Eventually, all weight accumulates in state 3.

$$\pi^* = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

12. (6 points) The diffusion equation

$$\frac{\partial}{\partial t}u(x, t) = D\frac{\partial^2}{\partial x^2}u(x, t)$$

has a general solution of the form

$$u(x, t) = \int_{-\infty}^{\infty} \frac{dk}{2\pi} \hat{u}(k) e^{ikx} e^{-Dk^2 t}.$$

The factor $e^{-Dk^2 t}$ can be understood as an amplitude $A(k, t) = e^{-\omega_k t}$ that decays exponentially over a k -dependent inverse time scale $\omega_k = Dk^2$. The purpose of this question is to determine how well a numerical integration of the PDE can reproduce this result.

- (a) Devise a finite difference approximation to the PDE that leads to the discretized field $u_i^{(n)} = u(i \cdot \Delta x, n \cdot \Delta t)$ being updated according to

$$u_i^{(n+1)} := u_i^{(n)}(1 - 2s) + s(u_{i+1}^{(n)} + u_{i-1}^{(n)}).$$

What is the value of s ?

Make the approximations $\partial u(x, t)/\partial t \approx [u_i^{(n+1)} - u_i^{(n)}]/\Delta t$ and $\partial^2 u(x, t)/\partial x^2 \approx [u_{i-1}^{(n)} - 2u_i^{(n)} + u_{i+1}^{(n)}]/(\Delta x)^2$. The resulting difference equation can be rearranged to give

$$\begin{aligned} u_i^{(n+1)} &= u_i^{(n)} + \frac{D\Delta t}{(\Delta x)^2} (u_{i-1}^{(n)} - 2u_i^{(n)} + u_{i+1}^{(n)}) \\ &= u_i^{(n)}(1 - 2s) + s(u_{i-1}^{(n)} + u_{i+1}^{(n)}) \end{aligned}$$

where $s = D\Delta t/(\Delta x)^2$.

- (b) Draw the *stencil* for this scheme. Is it *implicit* or *explicit*?

The stencil is explicit and has this shape: \perp

- (c) Prove that the amplitude of a single wavevector mode obeys

$$A_k^{(n)} = A_k^{(0)} \left(1 - 4s \sin^2 \frac{k\Delta x}{2} \right)^n.$$

Hint: to start, substitute $u_i^{(n)} = A_k^{(n)} e^{ik(\Delta x \cdot i)}$.

With the recommended substitution,

$$u_i^{(n+1)} = u_i^{(n)}(1 - 2s) + s(u_{i-1}^{(n)} + u_{i+1}^{(n)})$$

becomes

$$\begin{aligned} A_k^{(n+1)} &= A_k^{(n)}(1 - 2s) + A_k^{(n)} s(e^{-ik\Delta x} + e^{ik\Delta x}) \\ &= A_k^{(n)}(1 - 2s + 2 \cos k\Delta x) \\ &= A_k^{(n)} \left(1 - 4s \sin^2 \frac{k\Delta x}{2} \right). \end{aligned}$$

By recursion,

$$A_k^{(n)} = A_k^{(0)} \left(1 - 4s \sin^2 \frac{k\Delta x}{2} \right)^n.$$

(d) Argue that the scheme is numerically unstable unless $\Delta t < (\Delta x)^2/2D$.

Since $A_k^{(n)}$ evolves geometrically, the amplitude will explode unless

$$\left| 1 - 4s \sin^2 \frac{k\Delta x}{2} \right| < 1.$$

That is, unless

$$-1 < 1 - 4s \sin^2 \frac{k\Delta x}{2} < 1 \quad \text{or} \quad 0 < 4s \sin^2 \frac{k\Delta x}{2} < 2.$$

Since the maximum value of sine is 1, we need to ensure that $s < 1/2$.

(e) Show that the amplitude $A_k^{(t/\Delta t)} = e^{-\tilde{\omega}_k t}$ varies on an inverse time scale $\tilde{\omega}_k$ that differs from the exact one as follows:

$$\tilde{\omega}_k = \omega_k \left(1 - \frac{(k\Delta x)^2}{12} + \frac{\omega_k \Delta t}{2} \right).$$

You may want to make use of the series expansions $\sin x = x - x^3/6 + O(x^5)$ and $\log(1+x) = x - x^2/2 + O(x^3)$.

We want to connect

$$A_k^{(n)} = \left(1 - \frac{4D\Delta t}{(\Delta x)^2} \sin^2 \frac{k\Delta x}{2} \right)^n$$

to its value at time $t = n \cdot \Delta t$. We start by writing

$$A_k^{(t/\Delta t)} = \left(1 - \frac{4D\Delta t}{(\Delta x)^2} \sin^2 \frac{k\Delta x}{2} \right)^{t/\Delta t} \equiv (1 - \Delta t \cdot \xi_k)^{t/\Delta t}$$

in terms of

$$\begin{aligned} \xi_k &= \frac{4D}{(\Delta x)^2} \sin^2 \frac{k\Delta x}{2} = \frac{4D}{(\Delta x)^2} \left[\frac{k\Delta x}{2} - \frac{1}{6} \left(\frac{k\Delta x}{2} \right)^3 + \dots \right]^2 \\ &= \frac{4D}{(\Delta x)^2} \left(\frac{k\Delta x}{2} - \frac{(k\Delta x)^2}{48} + \dots \right)^2 \\ &= Dk^2 \left(1 - \frac{(k\Delta x)^2}{24} + \dots \right)^2 \\ &= \omega_k \left(1 - \frac{(k\Delta x)^2}{12} + \dots \right), \end{aligned}$$

which we've expanded order by order in $k\Delta x$. Then

$$\begin{aligned} \tilde{\omega}_k t &= -\log A_k^{(t/\Delta t)} = -\frac{t}{\Delta t} \log(1 - \Delta t \cdot \xi_k) = -\frac{t}{\Delta t} \left[(-\xi_k \Delta t) - \frac{1}{2} (-\xi_k \Delta t)^2 + \dots \right] \\ &= \xi_k t \left(1 + \frac{\xi_k \Delta t}{2} + \dots \right) \\ &= \omega_k t \left(1 - \frac{(k\Delta x)^2}{12} + \dots \right) \left(1 + \frac{1}{2} \omega_k \Delta t + \dots \right) \\ &= \omega_k t \left(1 - \frac{(k\Delta x)^2}{12} + \frac{1}{2} \omega_k \Delta t + \dots \right) \end{aligned}$$

Programming (30 points)

This section of the exam is to be completed in the computer lab. You are free to organize your program in any way you choose so long as all the code resides in the program file `cluster.cpp` and can be compiled with the provided `makefile`.

To start, download the `Exam.tar.gz` archive from the exam start page:

http://www.ualberta.ca/~kbeach/phys420_580_exam.html

Unpack the archive and append your own last name to the `Exam` directory.

```
$ tar xzf Exam.tar.gz
$ mv Exam Exam_StudentLastName
$ cd Exam_StudentLastName
```

At the end of the exam, you should archive your work directory.

```
$ cd ..
$ tar czf Exam_StudentLastName.tar.gz Exam_StudentLastName
```

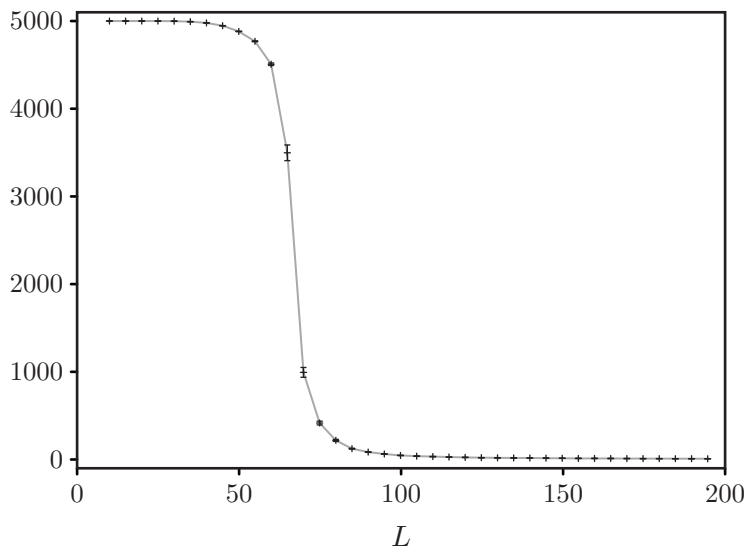
Email it to the instructor at `kbeach@ualberta.ca`.

N.B. Brute force solutions are fine. I don't expect elegance in an hour and a half.

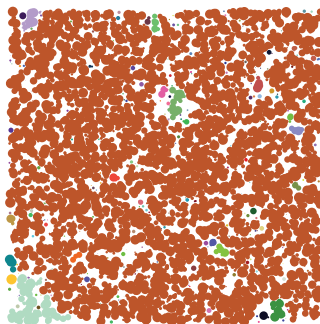
Imagine an experimental setup in which N tiny metallic spheres are sputtered onto an $L \times L$ surface. The spheres are variously sized; their radii are distributed according to $p(r) \sim r^{-2}$ to a maximum of $r = 1$ (in arbitrary units). Once they land on the surface, the spheres remain fixed in place.

The electrical properties of the resulting metallic film are dominated by the largest collection of spheres that are in direct contact with one another. Viewed in profile, the spheres are discs, and spheres that touch appear as overlapping discs. Thus, we want to develop an algorithm that can identify the largest cluster of overlapping discs.

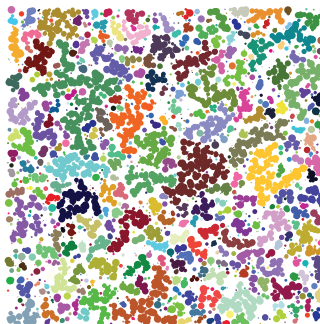
We restrict ourselves to $N = 5000$. What we'll find is that there are two distinct regimes (separated by a true phase transition in the $N \rightarrow \infty$ limit). At low density (N/L^2 small, L large), there are many isolated clusters consisting of a few discs each; at high density (N/L^2 large, L small) almost all the discs belong to one large cluster. The graph below shows the size of the largest cluster (averaged over many disorder realizations) as a function of L .



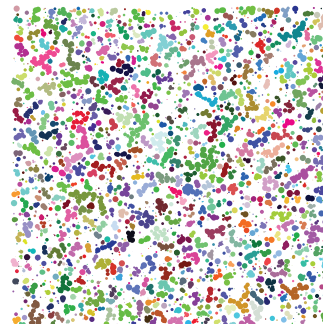
These are individual snapshots of particular disorder realizations for $L = 60, 80, 100$.



$L = 60$



$L = 80$



$L = 100$

13. (15 points) Read over the program file `cluster.cpp`. You'll find that it sets up some of the problem for you: the `discs` are initialized at random positions and stored in a `vector<disc>` named `sputter`; there is also code that saves a depiction of the configuration as an encapsulated postscript (EPS) file.

The `disc` class has an integer data member that will serve as a cluster label; you can access it via the `query_label` and `assign_label` member functions. By default, the label is set to `-1`. Your task is to assign it values `0, 1, 2, ...` such that

- (i) any two discs belonging to the same cluster have the same label,
- (ii) any two discs in different clusters have different labels, and
- (iii) the integers assigned run from `0` to $N_{cl} - 1$ with no gaps, where N_{cl} is the number of clusters

In the `print_config` routine, the discs are assigned fill colours based on their index. So, if you've done everything correctly, your program's EPS output should exactly reproduce the three figures shown on the previous page.

```
$ ./cluster 60 5000
WARNING: Vector 'cluster_number' is improperly assigned!
$ ./cluster 80 5000
WARNING: Vector 'cluster_number' is improperly assigned!
$ ./cluster 100 5000
WARNING: Vector 'cluster_number' is improperly assigned!
$ gv config-60-5000.eps
$ gv config-80-5000.eps
$ gv config-100-5000.eps
```

14. (10 points) Populate the containers `cluster_number` and `cluster_area` with the number of discs in each cluster and their total area, respectively. Use the cluster label as an index. The program output should now read as follows.

```
$ ./cluster 60 5000
Largest cluster by number = 4518
Largest cluster by area   = 4931.81
$ ./cluster 80 5000
Largest cluster by number = 199
Largest cluster by area   = 255.484
$ ./cluster 100 5000
Largest cluster by number = 41
Largest cluster by area   = 55.5979
```

15. (5 points) Create an (unnormalized) histogram of the sizes stored in `cluster_number` for each of $L = 60, 80, 100$. Output the histogram as a two-column text file with the naming convention `hist-L-N.dat`.