10-751 Assignment 4

Due: Thursday October 17

The following problems are concerned with random walks and the Dirichlet problem, as presented in class. This is discussed at length in Doyle and Snell's *Random Walks and Electric Networks*, from which the following problems are adapted. This book is available online from `http://front.math.ucdavis.edu/math.PR/0001057`.

Problem 1. Show that if f and g are harmonic functions so is $h = af + bg$ for constants a and b. (This is called the *superposition principle.*)

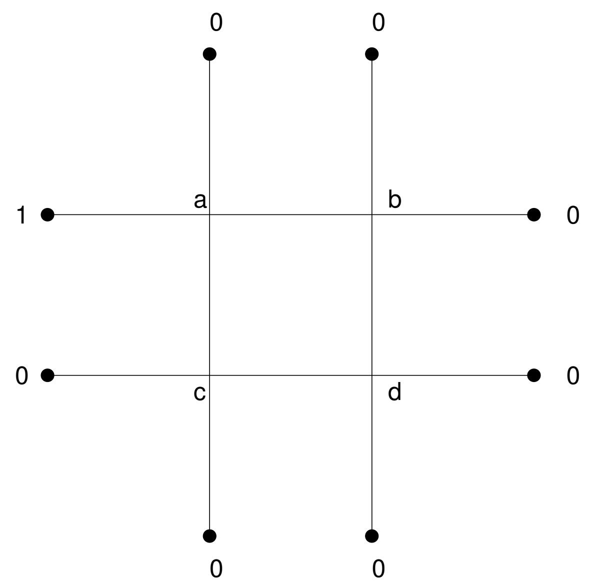
Solution. The combinatorial Laplacian Δ is linear, so if $\Delta f = \Delta g = 0$ then $\Delta(af + bg) = 0$.

Problem 2. Let $B_1, B_2, \ldots, B_n$ be the boundary points for a region S. Let $e_j(a, b)$ be a function that is harmonic in S and has boundary value 1 at B_j and 0 at the other boundary points. Show that if arbitrary boundary values $v_1, v_2, \ldots, v_n$ are assigned, we can find the harmonic function v with these values from the solutions $e_1, e_2, \ldots e_n$.

Solution. By the previous problem, the solution is

$$f = v_1 e_1 + v_2 e_2 + \ldots + v_n e_n$$

Problem 3. Consider a random walk on the graph below. Find the probability of reaching the point with a 1 before any of the points with 0's for each starting point a, b, c, and d.



Solution. The transition matrix P is given by

$$P = \begin{pmatrix} I & 0 \\ R & Q \end{pmatrix}$$

where I is the 8×8 identity matrix, Q is the matrix

$$Q = \begin{pmatrix} 0 & \frac{1}{4} & \frac{1}{4} & 0 \\ \frac{1}{4} & 0 & 0 & \frac{1}{4} \\ \frac{1}{4} & 0 & 0 & \frac{1}{4} \\ 0 & \frac{1}{4} & \frac{1}{4} & 0 \end{pmatrix}$$

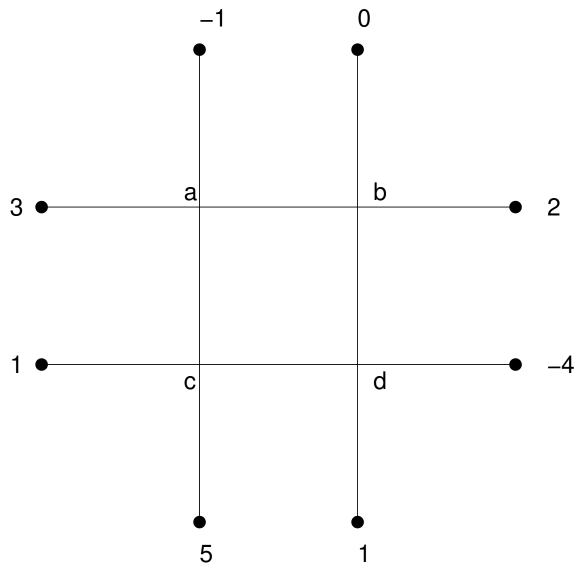
and R is the matrix

$$R = \begin{pmatrix} \frac{1}{4} & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{4} & \frac{1}{4} \\ 0 & 0 & 0 & 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 \end{pmatrix}$$

where we index the boundary points clockwise starting with the vertex labeled 1.

Let $A = (I - Q)^{-1}$. Then the probability of starting at $x \in \{a, b, c, d\}$ and hitting the vertex labeled 1 is given by $\frac{1}{4}A_{xa}$.

Problem 4. Solve the Dirichlet problem for the graph shown below.



Solution. Using the setup of the previous problem, and relying on the solution of Problem 2, the solution of the Dirichlet problem for this graph is given by

$$f_D(x) = \frac{1}{4}(3-1)A_{xa} + \frac{1}{4}(0+2)A_{xb} + \frac{1}{4}(-4+1)A_{xd} + \frac{1}{4}(5+1)A_{xc}$$

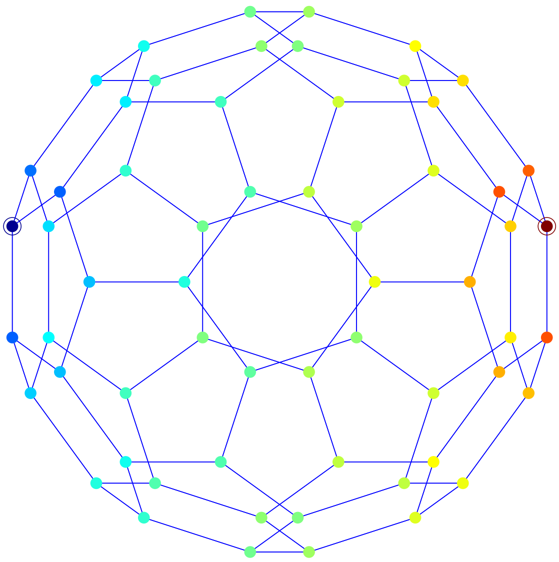
for $x \in \{a, b, c, d\}$.

Problem 5. In this problem, you will implement the random walk approach to the Dirichlet problem in Matlab.

Given a graph G and a set of distinguished boundary vertices with prescribed values f_B, when the random walk is at vertex x it steps to one of the neighbors of x (with uniform probability), until it hits one of the boundary vertices. The *payoff* is the value of the boundary vertex where the walk stops. The solution to the Dirichlet is the function $f_D : G \longrightarrow \mathbb{R}$ such that $f_D(x)$ is the expected payoff when the random walk starts at vertex x.

You are to write a Matlab function `dirichlet` that takes three arguments: the $n \times n$ weighted adjacency matrix `A` of the graph, an $n \times 2$ matrix `XY` of coordinates, and a list of boundary vertices and their values, in the form of a $k \times 2$ matrix `bndry`, where k is the number of boundary vertices. The matrix of coordinates `XY` is such that $(\texttt{XY}(j,1), \texttt{XY}(j,2))$ is the (x,y)-coordinate of vertex j when the graph is drawn in the plane. Your function should compute the solution f_D using the Markov chain method, and then plot the result, using the `gcplot` Matlab function which we have provided for you.

As an example of a graph, try setting `[A XY] = bucky` in Matlab and then evaluating `gplot(A, XY)`. Using this graph as input, an example call would be `dirichlet(A, XY, [[9,1];[53,0]])`, specifying that vertex 9 has value 1 and vertex 53 has value 0. The output of your program should then look something like the figure below, where the color of vertex x indicates the value of $f_D(x)$.



A call to `gcplot` looks like `gcplot(A, XY, bndry, f)` where `A` is the adjacency matrix, `XY` is the matrix of coordinates, `bndry` is a list of the boundary vertices (so that the graphing function can mark them with special symbols), and `f` is your computed solution to the Dirichlet problem.

The Matlab code `gcplot.m` and sample graphs are available off of the course Web page. Please email your Matlab code for the `dirichlet` function to Guy Lebanon, `lebanon@cs.cmu.edu`.

Solution. Here is a sample `dirichlet.m` function:

```
%  Solve the Dirichlet problem on a graph and display the results.
%  Usage:
%
%    dirichlet(G, XY, bndryValues)
%
%  runs on the graph G having (x,y) coordinates XY, with
%  boundary values bndryValues = [[n1,v1];[n2,v2];...;[nk,vk]]
%  which specify that node n1 has value v1, etc.

function dirichlet(G, XY, bndry)

     n = length(G);
     f = zeros(n,1);
     g = zeros(n,1);
     ind = zeros(n,1);
     pos = zeros(n,1);
     isboundary = zeros(n,1);
     p = zeros(n,n);
     d = size(bndry);
     labeled = d(1);
     unlabeled = n-labeled;

     % assign indices, with the labeled vertices first, and
     % fix the values on the boundary

     for i = 1:labeled
       x = bndry(i,1);
       v =  bndry(i,2);
       ind(x) = i;
       pos(i) = x;
       isboundary(x) = 1;
       f(i) = bndry(i,2);
     end;

     % now index the unlabeled data
     i = labeled+1;
     for x = 1:n
        if (not(isboundary(x)))
           ind(x) = i;
           pos(i) = x;
           i = i+1;
       end
     end
     if (not(i-1==n))
        error('Error indexing unlabeled data: %d != %d', i-1, n)
     end
```

```
for i = 1:labeled
   p(i,i) = 1;
end

% Now construct the transition matrix
for i = (labeled+1):n
   x = pos(i);
   numNeighbors = 0;
   for y = 1:n
     if (G(x,y))
        numNeighbors = numNeighbors+1;
     end
   end

   % normalize
   for y = 1:n
      if (G(x,y))
         j = ind(y);
         p(i,j) = 1/numNeighbors;
      end
   end
end

% Finally, do the algebra that computes the sol'n
% to the Dirichlet problem
R = p(labeled+1:n,1:labeled);
Q = p(labeled+1:n,labeled+1:n);
Id = eye(unlabeled,unlabeled);
B = inv(Id-Q)*R;
f(1+labeled:n) = B*f(1:labeled);

% Now map f to a vector of 'colors'
C = zeros(n,1);
for i = 1:n
  x = pos(i);
  C(x) = f(i);
end

% plot it!
gcplot(G, XY, bndry, C);
```