MA50177: Scientific Computing
Case Study

The Graph Drawing Problem

This is a problem in discrete mathematics which is closely related to other problems such as mesh partitioning and node ordering.

Suppose we are given a **graph** $\mathcal{G} := (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is a set of edges. Each edge connects two nodes. We assume that the graph is *undirected*, i.e. we do not distinguish between the edge (u, v) and the edge (v, u). We assume also that the graph is *connected*, i.e. every node is connected to every other node by a path of edges.

In many applications, spatial coordinates for the nodes are not given in the graph specification. For example the graph could represent a family tree, with nodes for people and edges representing relationships between them.

The graph drawing problem is to determine spatial coordinates for the nodes (in 2D or 3D space) in such a way that the layout of the subsequent visualisation of the graph represents, as faithfully as possible, the structure of the graph. Such an assignment of spatial coordinates is called a **drawing** of the graph. In this assignment we will only be concerned with 2D drawings of graphs.

In order to make the graph drawing problem more tractable, we shall assume that each edge $(u, v) \in \mathcal{E}$, has been given an associated weight $w_{(u,v)}$ (a real number). In the drawing, edges with large weights are required to be represented in the visualisation by relatively short lines, while edges with smaller weights should be represented by longer lines.

Let N denote the number of nodes in $\mathcal{V}$. The graph drawing problem in 2D then requires the computation of vectors $\boldsymbol{x},\ \boldsymbol{y} \in \mathbb{R}^N$, containing the $x-$ and $y-$ coordinates of suitable points where the nodes in $\mathcal{V}$ should be drawn (i.e. (x_u, y_u) will be the coordinates of node u). Subsequently the graph can be visualised.

Energy minimisation methods.

There is a huge literature on the graph drawing problem and in one of the original papers [1], Hall proposed that suitable vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ of x and y coordinates could be found by minimising, in some sense, the energy functional ψ defined, for any $\mathbf{z} \in \mathbb{R}^N$, by

$$\psi(\mathbf{z}) := \sum_{(u,v)\in\mathcal{E}} w_{(u,v)}(z_u - z_v)^2 \ . \tag{1}$$

In particular Hall suggested to determine the x coordinates of the nodes by requiring that $\boldsymbol{x}$ solves the minimisation problem:

$$\psi(\boldsymbol{x}) = \min\{\psi(\mathbf{z}) \ : \ \mathbf{z} \in \mathbb{R}^N \ , \ \|\mathbf{z}\|_2 = 1, \ \mathbf{z}^T\mathbf{e} = 0\} \ . \tag{2}$$

where $\mathbf{e} = (1, 1, \ldots, 1)^T \in \mathbb{R}^N$. Intuitively, if $\boldsymbol{x}$ minimises ψ, then edges with large weight $w_{(u,v)}$ will correspond to a small value of $(x_u - x_v)^2$. (Note that, although $\psi(\mathbf{e}) = 0$, the solution $\mathbf{e}$ has been ruled out as a solution to the minimisation problem.

This is because placement of all the nodes at $x = 1$ is clearly not a feasible drawing. Hence the minimisation problem (2) is carried out in the orthogonal complement of the "spurious solution" $\mathbf{e}$.)

If $\boldsymbol{x}$ does solve (2), then Hall proposed to determine the y coordinates of the nodes by solving the additional minimisation problem:

$$\phi(\boldsymbol{y}) = \min\{\psi(\mathbf{z}) \ : \ \mathbf{z} \in \mathbb{R}^N \ , \ \|\mathbf{z}\|_2 = 1, \ \mathbf{z}^T\mathbf{e} = 0 \ , \ \mathbf{z}^T\boldsymbol{x} = 0\} \ . \tag{3}$$

Finally we introduce two matrices which feature strongly in the solution of these minimisation problems. The $N \times N$ **weighted adjacency matrix** A is defined by

$$A_{u,v} = \begin{cases} w_{(u,v)} \ , & \text{if } (u,v) \in \mathcal{E} \\ 0, & \text{otherwise.} \end{cases}$$

For any $v \in \mathcal{V}$, set

$$d_v = \sum_{u \text{ s.t. } (u,v)\in\mathcal{E}} w_{(v,u)} \ , \quad \text{and} \quad D = \text{diag}\{d_v : v \in \mathcal{V}\}.$$

Then, the **weighted Laplacian** of the graph is defined by:

$$L = D - A.$$

The Assignment.

In the questions below, the notation [**length**, **n**] indicates the length of a typical handwritten answer and the number of marks out of 30 to be awarded for it.

1. Prove that the solution to (2) is an eigenvector of L corresponding to the smallest positive eigenvalue of L. Assuming that the smallest positive eigenvalue of L is simple, prove that the solution to (3) is an eigenvector of L corresponding to the second smallest positive eigenvalue of L. [**2 pages, 5**]

2. The algorithm below can be used to find the required eigenvectors $\boldsymbol{x}$ and $\boldsymbol{y}$. Two new notations appear in this algorithm. If Z denotes any $N \times 2$ matrix, then P denotes the operator:

$$P(Z) = Z - \frac{\mathbf{e}\ \mathbf{e}^T}{\mathbf{e}^T\mathbf{e}} Z \ .$$

   Also the notation $[Q, R] = qr(Z)$ means that an $n \times 2$ matrix Q with orthonormal columns, and a 2×2 upper triangular matrix R should be computed such that $Z = QR$. (Q and R can be found by taking appropriate submatrices of the matrices computed by the QR decomposition routine from `LAPACK` . You may assume that the diagonal entries of R are positive.)

   Algorithm 1.

   Initialisation phase:
   Choose a random $N \times 2$ matrix Y^0. Choose $\sigma < 0$, σ fairly near 0.

Project: $Y^0 \leftarrow P(Y^0)$.
Compute: $[Q, R] = qr(Y^0)$, then set $Z^1 = Q$.

Iteration:
do $k = 1, 2, \ldots$,
    Solve $(L - \sigma I)Y^k = Z^k$ for Y^k.   (*)
    Project $Y^k \leftarrow P(Y^k)$
    Compute: $[Q, R] = qr(Y^k)$, then set $Z^{k+1} = Q$.
end do

(The superscripts k denote the kth matrix in a sequence, not the kth power.)

Prove that as $k \to \infty$, the first column of Z^k will converge to a solution to (2). You may use any results given in lectures. [**1.5 pages, 4**]

You may now assume that the second column of Z^k will converge to a solution of (3).

3. Write a `fortran95` implementation of Algorithm1 to compute the x and y coordinates of a drawing for any given weighted graph. Use `real(kind=2)` arithmetic. Use `LAPACK` to provide the QR factorisation and to solve the linear systems in (*). It should be sufficient to employ the QR factorisation without pivoting. Make sure your program is correct and efficient in its use of storage. Program a suitable stopping criterion for the loop in Algorithm 1. Have your program write $\boldsymbol{x}$ and $\boldsymbol{y}$ to a file in a format which can be used by the given matlab visualisation tool `graphdraw.m`. Hand in your programs on a floppy disk. [**9**]

4. The given matlab program `graphmake.p` provides sample graphs which can be used to test the node placements determined Algorithm 1. `graphmake` depends on three parameters α, β and L. The parameters α and β should be real numbers in $(0, 1)$. They determine the shape of the graph, whereas the integer L determines the density of the graph. (Typical values for L are $2, 3, 4$.) The program `graphmake.p` depends on utilities `getedges.p` and `refine.p`. They all are parsed code. That is you can run them in matlab in the usual way but you cannot read them. The program `main.m` calls `graphmake.p` to assemble lists of edges and weights of the graph and then writes them to files `edges.txt` and `weights.txt`. The first entry of each list is the total number of edges (or weights). These files can be read from a `fortran95` program. . The parsed code will run in `matlab` on the MSc machines, but it cannot be guaranteed to run on any other matlab installation.

   Each student is assigned personal data which can be read from the assignment home page: `ma50177/assignments`

   For your personal data, compute the vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. In your handed-in assignment, give, to six figure accuracy, the coordinates of node 5 in your graph drawing. Provide also a hard copy of your visualised personal graph. [**4**]

You have now completed more that 70% of the assignment. I hope that you will complete the assignment by doing the last part, below. However, before you decide to do this, you should make sure that you are not neglecting your other courses by spending a disproportionately large amount of time on this assignment.

5. **For extra credit:** Ensure that your code performs as efficiently as possible, for example by (i) using BLAS to optimise the linear algebra operations; (ii) choosing σ carefully and possibly allowing σ to vary from one iteration to the next; (iii) employing an efficient linear solver for the solve step (*) or (iv) any other efficiency measures. Write a report explaining the measures you have taken to ensure efficiency. This may be illustrated with the results of numerical experiments.
[**3 pages, 8**]

References

[1] K.M. Hall, An r-dimensional quadratic placement algorithm, *Management Sci.* 17 (1970), pp 219-229. (This is available in the library)

[2] Y. Koren, L. Carmel and D. Harel, Drawing huge graphs by algebraic multigrid optimisation, *Multiscale Modeling and Simulation* 1 (2003), pp 645-673. (I will put a copy in the MSc room.)

[3] D. J. Higham and M. Kibble, A Unified View of Spectral Clustering, University of Strathclyde Mathematics Research Report 02 (2004), January 2004. (available from `http://www.maths.strath.ac.uk/~aas96106/reps.html`)