Algovision
Arithmetic Algorithms: Addition

This tour has just one scene that shows how to find the minimum edge cut of a unoriented (symmetric) regular graph (or at least its approximation) using a spectral heuristic. A similar method can be developed for general graphs, but it is much more complicated and it is not shown.

An unoriented graph is regular it all its vertices have the same degree (i.e., the number of neighbors).

An incidence matrix of a graph with n vertices is an $n \times n$ matrix A such that $A_{ij} = 1$ if the i-th vertex of the graph is connected by an edge with the j-th one, otherwise $A_{ij} = 0$.

A vector v is an eigenvector of a matrix A corresponding to an eigenvalue λ, if $Av = \lambda v$.

The incidence matrix of an unoriented graph is symmetric ($A_{ij} = A_{ji}$ for all i, j). Given a symmetric matrix A of the size $n \times n$, it is known that it is possible to find n linearly independent eigenvectors of A.

An incidence matrix of an unoriented regular graph with vertices of degree d has always an eigenvector $\mathbf{e} = (1, 1, \ldots, 1)$ corresponding to the eigenvalue d. The proof is easy: the i-the element of the vector $A\mathbf{e}$ is the sum of all elements in the i-th row of the matrix A. If A is an incidence matrix of a regular graph of degree d, then the sum is d for every row of the matrix, which means that the product $A\mathbf{e}$ is the vector $(d, d, \ldots, d)$. Note that in this case *any* vector of the form $(c, c, \ldots, c)$ for any constant c is an eigenvector of A corresponding to the eigenvector d.

It is possible to prove that d is the largest eigenvalue of the incidence matrix of an unoriented regular graph of the degree d. The proof is not difficult, but it is omitted.

The problem of the minimum cut of a graph with the vertex set V with even number of vertices is to split V into two disjoint subsets V_1 and V_2 of the same size $n/2$ so that the number of edges $u - v$ such that $u \in V_1$ and $v \in V_2$ is minimized. The problem is known to be NP-hard and therefore it is usually solved by heuristic algorithms that are fast, but do not guarantee to find the optimal solution. Spectral min-cut heuristics (i.e., heuristics using eigenvalues and eigenvectors) are supposed to be extremely good methods. The underlying idea of the is explained in the present tour.

When the tour starts, it shows a regular graph with 64 vertices of degree 4. The button New Input makes it possible to get another graph; using fields

Vertices and Degree it is possible to set its number of vertices and the degree to a different value.

The initial graph (and the graph obtained after clicking New Input) has the minimum cut of the size 0: its vertices can be separated into two disjoint parts that are not connected by any edge. If you click Change view, vertices of the graph move horizontally to show the partition. (Clicking Change view again restores the initial view).

A graph G with even number of vertices and such a separation of vertices has always another eigenvector corresponding to the eigenvalue d. Assume that vertices are ordered in such a way that the first half and the second half of the represent the 0 cut partition. Then the incidence matrix of the graph looks like that:

$$\begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix}$$

where A_1 and A_2 are incidence matrices of the subgraphs G_1 and G_2 of the graph G generated by the respective sets V_1 and V_2 (both of the size $(n/2) \times (n/2)$), and both symbols 0 represent matrices of the size $(n/2) \times (n/2)$ that are composed of zeros. Since no edges cross the boundary between V_1 and V_2, both G_1 and G_2 are unoriented regular graphs of degree d. In this case, if c_1 and c_2 are arbitrary constants, than the vector $(c_1, c_1, \ldots, c_1, c_2, \ldots, c_2)$ (the same number of c_1's and c_2's) is also an eigenvector of the incidence matrix of G corresponding to the eigenvalue d, because if we denote by v_i, $i = 1, 2$, the (transposition of the) vector $(c_i, \ldots, c_i)$ of the length $n/2$, then

$$\left(\begin{array}{cc} A_1 & 0 \\ 0 & A_2 \end{array} \right) \left(\begin{array}{c} v_1 \\ v_2 \end{array} \right) = d \left(\begin{array}{c} v_1 \\ v_2 \end{array} \right)$$

In the tour, we use the constants $c_1 = 1$, $c_2 = -1$, because in the case the eigenvectors $(1, \ldots, 1)$ and $(1, \ldots, 1, -1, \ldots, -1)$ are orthogonal (their scalar product is 0). Click the checkbox Show Spectrum to display the second (non-constant) eigenvector. Each element of the eigenvector corresponds to some vertex and it is drawn *below* the vertex. Click a bar that represents an element of the eigenvector - not only the bar, but also the corresponding vertex (and edges incident with it) are highlighted. Click a vertex - the corresponding eigenvector element is highlighted as well.

In the upper right corner of the window, the eigenvalue corresponding the the eigenvector $(1, \ldots, 1)$ (λ_1) and the eigenvalue corresponding the the eigenvector $(1, \ldots, 1, -1, \ldots, -1)$ (λ_2) are displayed; both are equal to the degree of vertices of the graph.

Thus, it is possible to find the (zero) minimum cut of the present graph by computing the second eigenvector and split the vertices by the sign of the corresponding eigenvector element.

Well, the partition in this case can be found by much easier methods, e.g., by a graph search. But click the button Step. The graph is randomly modified

to have the minimum cut equal to two:
an edge $u_1 - u_2$ is randomly selected in the left partition and an edge $v_1 - v_2$ is randomly selected in the right partition. The edges are removed from the graph, and the edges $u_1 - v_1$ and $u_2 - v_2$ are added. The graphs is still unoriented and regular, the partition of vertices to two parts remains (with extremely high probability) the minimum cut, but its size is 2 (two added edges being cut by the partition).

Look at the second eigenvalue (λ_2 in the upper right corner) and the corresponding eigenvector. The second eigenvalue is now already smaller than the degree of vertices. The eigenvector still has one half of elements positive (green) and the other half negative (black), and the partition of vertices is still equal to the partition by the signs of the eigenvector elements.

However, you can find two green bars and two black bars that are much shorter than the other one. Click at them - and you will see that they correspond exactly to four vertices that have one neighbor in the "other" partition. The bars of the other vertices do not have the same length. Click one that is among shorter (but not as short as the shortest four): it is very likely that such a vertex has one of the distinguished four as a neighbor.

Keep clicking Step for some time. Each time you click, the above described modification of the graph increases the number of edges crossing the partitions boundaries by two. Look first at the second eigenvalue $\lambda_@$ - it decreases by each click. This is known by mathematicians for a long time - the gap between the first and the second eigenvalue increases as the graph gets more "connected".

Now, look at the eigenvector and you will find that short elements of the eigenvector tend to correspond to vertices that have neighbors in the other partition. After some time you will perhaps find eigenvector elements that are "wrong" - they have the opposite sign than the other members of the partition (a black "negative" bar on the left or a green "positive" bar on the right). Click at such a bar - it is very likely that the vertex has the same number of neighbors in both partitions. This could mean that there is another cut of the minimal size, where the vertex is really in the other partition.

The heuristics to find a good cut works as follows: compute the second eigenvector, find the number s such that exactly one half of the eigenvector elements is greater than s, and split vertices by the relation of their corresponding eigenvector elements to s.

The only thing to be explained is why this method works so well - eigenvectors are often regarded as a kind of witchcraft or voodoo than the math and it is not easy to see how and why they help.

Let us explain the background of the method as follows:

View vertices of the graphs as citizens of two countries, West and East, that would represent the minimum (or a good) cut of the graph. We want to give a score to each vertex so that good citizens of the West have highly positive scores and good citizens of the East have highly negative scores (in the way you see in the window). Imagine that edges represent some kind of family relation. Since we want to get low number of edges connecting East and West, we would certainly want the score of a vertex similar to the scores of its relatives. Having this in mind, it is a good idea to to require that the score of a vertex be the average of the scores of its neighbors:

$$score(v) = \frac{1}{d} \sum_{w \in N(v)} score(w),$$

where $N(v)$ is the set of all neighbors (relatives) of the vertex v. After having consulted math textbooks, it turns out that a good score verifying the equation and separating citizens to West and East might not exist, and the equation should be amended by replacing d by another unknown λ that would hopefully be close to d. When the equation is modified in the way and both its sides multiplied by λ, we get

$$\lambda score(v) = \sum_{w \in N(v)} score(w).$$

But now, the left-hand side of the equation is the vector *score*, multiplied by λ, while the right-hand side of the equation is nothing else then the product of the incidence matrix of the graph with the vector *score*. This means that the last equation exactly says that the vector *score* is the eigenvector of the incidence matrix of the regular graph in question. The only question that is not clear is why solving the equation is so difficult.

Well, there is still one question unanswered: why we are using the eigenvector corresponding to the *second* eigenvalue. But the answer is easy: the eigenvector corresponding to the first (i.e., the largest) eigenvalue d is, as it has already been pointed out above, constant, which means that the score given to all citizens (vertices) would be the same. This is the best possible solution of the problem of finding a score so that a vertex has the score similar to the scores of its relatives (neighbors), but not a method for partition of vertices. It is known that the second eigenvector is (or can be) orthogonal to the first one (which is constant) which means that the sum of all elements of the eigenvector is zero. In such a case one could expect that about one half of the elements have positive score, and the partition by eigenvalue element signs gives a partition of the vertex set into two parts of approximately same size.