Graph-Theoretic Methods

Motivation and Introduction

One is often faced with analyzing large spatial or spatiotemporal datasets – say involving *N* nodes, or *N* time series. If one is only interested in the individual behavior of each node or time series, analysis remains tractable (or at least, as order *N*) – a "massively univariate" approach (for spatial datasets). However, this approach (a) ignores any possible interactions between the nodes or time series, and, implicitly, makes the assumption that the *N* nodes or time series directly represent the underlying variables of interest -- the more likely alternative being that the observables each represent a mixture of the underlying variables.

A natural approach is therefore to look at covariances (in the case of spatial datasets) or, in the case of time series, cross-spectra – the quantities $P_{X_i,X_j}(\omega)$ defined earlier. While this is comprehensive (at least for pairwise interactions), there's a scaling problem – the number of covariances (or cross-spectra) is $N(N-1)/2$. So even if *N* is relatively small (say, 25 to 50), N^2 can be sufficiently large so as to be difficult to visualize, and when *N* is large (e.g., imaging pixels), a comprehensive approach may be computationally impractical. So this motivates the development of methods that summarize the collective properties of a set of covariances, or a matrix of cross-spectra.

One kind of approach is to recognize that these covariances do form a matrix, and, as such, the standard ways of describing a matrix are applicable. A good example is the "global coherence" – this is the ratio of the largest eigenvalue of the matrix of cross-spectra to its trace, i.e., a measure of the extent to which the cross-spectral matrix can be approximated by a matrix of rank 1 (see Homework 3 for LSBB, 2012-2013). A rank-1 cross-spectral matrix implies that the observed covariances can all be accounted for by a single common source of noise, so the global coherence has an immediate interpretation.

Graph-theoretic methods represent another kind of approach. Here, the basic idea is also one of dimensional reduction, but the dimensional reduction occurs at an earlier stage: reducing the original pairwise measurements, which may be real or complex numbers, to something simpler – typically a matrix A whose elements a_{ij} are just 0's and 1's. Then, one looks at the properties of this array.

Overall, the advantage of this kind of approach is that it focuses on the qualitative nature of the interrelationships of the observed variables, and their network-like (or "topological") attributes. The main cautionary note is that the reduction from covariances or cross-spectra to binary quantities involves thresholding (and, in the case of cross-spectra, the choice of analysis freqeuency.) So to check on the robustness of the results, one might want to investigate how they depend on this choice – even if the choice is "objective" (as in, a level of statistical significance).

One should also note that graph-theoretic methods have in general been developed for application to domains in which this thresholding is not an issue – i.e., domains in which the

fundamental measurements are binary. A good example is the case of social networks: a matrix element $a_{ij} = 0$ means that individuals *i* and *j* are strangers, while $a_{ij} = 1$ means that they know each other.

Note also that in this case, the matrix *A* is necessarily symmetric. This symmetry typically is taken to be part of the definition of a graph. But many graph-theoretic measures are applicable far beyond the restricted domain of symmetric, binary graphs. Asymmetric but binary arrays correspond to "directed" graphs (for example, *i* has contacted *j*, or website *i* links to website *j*). Arrays that are not binary correspond to "weighted" graphs (some links are stronger than others). Depending on the data type, these extensions may be particularly natural. For example, diffusion tractography data provides a measure of the strength of connectivity, but not its directionality. In all of these cases (and in everything we consider), $a_{ij} \geq 0$. Relaxing this condition (i.e., allowing negative "strengths") changes the character of the problem greatly – for example, the graph Laplacian is no longer positive definite, so that "diffusion" can diverge. Taking this even further – allowing a_{ij} to be complex, with $a_{ji} = \overline{a_{ij}}$, essentially yields the picture of general linear dynamics at a single frequency, as the connectivities now can be considered to represent values of a transfer function.

There is also a connection to Markov chains. If every column of a graph matrix sums to 1 (the "Markov" condition), we can view a_{ij} as the probability than a particle at node *j* will, at the next time step, move to node *i*. Note also that we can take any graph matrix and column-normalize it (i.e,. replace a_{ij} by $a_{ij} / \sum_k a_{kj}$) to ensure that its columns sum to 1, so that we can always turn a graph matrix into a Markov matrix.

Elements

Two web-based sources for this material may be found in Radu Horaud, http://csustan.csustan.edu/~tom/Clustering/GraphLaplacian-tutorial.pdf and Sukanta Pati, http://www.lix.polytechnique.fr/~schwander/resources/mig/slides/pati.pdf But there are notational inconsistencies and some annoying typos.

General definitions

A *graph G* consists of a set of vertices *V* (a.k.a. nodes) and an adjacency matrix *A* whose rows and columns correspond to the vertices.

The vertices are typically labeled by the integers $\{1,...,N\}$ -- but typically, these are abstract labels, and their numerical values are irrelevant. All elements of the adjacency matrix *A* are assumed to be non-negative.

A *subgraph* of *G* is a subset of the vertices, along with the matrix formed from the corresponding rows and columns of the adjacency matrix *A*.

If $a_{ij} > 0$, we say that there is an edge connecting vertex *i* to vertex *j*, and write $i \sim j$.

The *degree* of a vertex is the number of vertices that it connects with. In the case of a directed graph, one needs to distinguish between the outgoing degree and the incoming degree.

The *distance* between two vertices is the minimum number of edges that must be traversed to pass between them. Each example of a minimum-length path is called a *geodesic*. The *diameter* of a graph is the length of its largest geodesic. It is the lowest power p for which A^p has no zero off-diagonal elements.

A *clique* is a set of vertices for which every pair is connected by an edge. Two vertices connected by an edge trivially form a clique. A three-clique is a set of vertices whose subgraph is a triangle.

On this general setup, we typically add one or more kinds of restrictions:

If we allow nonzero a_{ij}'s to be quantities other than 1, the graph is a "weighted" graph.

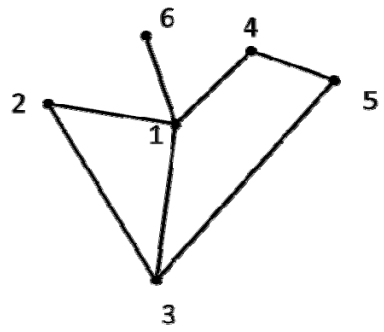
If we allow the matrix A to be asymmetric, the graph is a "directed" graph; a_{ij} is the strength of the connection from j to i (this convention makes the connection with Markov matrices most straightforward).

Typically, we will deal with "simple" graphs; three conditions apply: all a_{ij} are 0 or 1, they are symmetric, and the on-diagonal elements are 0. The last condition means that there are no self-loops.

Going from a simple graph to a weighted graph is often a straightforward generalization, in that a graph with integer weights can be thought of as a graph in which there are multiple connections between a single pair of nodes. So one can expect that this generalization will not affect approaches that depend on the algebraic structure of A (though it may affect approaches that depend on counting the number of paths).

In contrast, going from a simple graph to a directed graph means allowing A to be asymmetric. This will change its algebraic properties markedly, since A goes from being self-adjoint to non-self-adjoint. However, combinatorial approaches (e.g, approaches that rely on counting paths between nodes) may generalize readily. For example, each element of A^k counts the number of k-step paths from node j to node i.

A simple graph is just a set of points and lines that connect them:



For this graph, the adjacency matrix is given by $A = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$.

The cliques consist of the seven pairs of vertices connected by edges, and also the 3-clique $\{1,2,3\}$.

Some special kinds of graphs

The following are fairly common, and often useful:

A *connected* graph is a graph for which, for every pair of vertices, there is a sequence of edges that connects them. Equivalently, it is a graph for which a sufficiently high power of the adjacency matrix *A* contains no zero elements. Since the elements of *A* and its powers are non-negative, this is equivalent to the condition that $1 + zA + z^2A^2 + ... = \frac{1}{1 - zA}$ (series converges for sufficiently small *z*) has no zero elements. Equivalently, a connected graph is a graph whose diameter is finite.

A *cyclic* graph is a connected graph for which every vertex has degree 2.

A *forest* is a graph that has no subgraphs that are cycles.

A *tree* is a forest that is connected.

A *triangle-free* graph is a graph that has no subgraphs that are 3-cliques. All forests are triangle-free.

A *regular* graph is a graph for which all vertices have the same degree.

A *complete* graph is a graph for which all edges are present. Equivalently, the entire graph is a clique.) Equivalently, it is a graph of diameter 1.

A *star* graph is a graph in which one vertex is connected to all of the others, and there are no cycles. It has diameter 2.

A *bipartite* graph is a graph whose vertices can be partitioned into two disjoint subsets, and the vertices within each subset are not connected with each other. (Similarly for a *k-partite* graph.)

Some special kinds of graph architectures

The above terms apply to specific graphs, and they are fragile – the properties can be gained or lost by insertion or deletion of a single vertex or edge. The terms below are different – they are intended to apply to graph architectures (strategies of building graphs with an arbitrarily large number of nodes, as the number of nodes is large).

A *scale-free* graph is a graph for which the frequency of nodes with a given degree is a power-law function of the degree. That is, the probability that a node has degree between d and $d + \Delta d$ is given by $p(d) = d^{-\gamma}\Delta d$; typically γ is in the range of 2 to 3. Obviously this can only hold over a range of degrees, and must be approximate. But the basic idea is that there are a small number of vertices that have a large number of connections ("hubs"), a larger number of nodes that have somewhat fewer connections, etc.

A *small-world* graph (or network) is a graph in which the average distance between pairs of vertices is small in comparison to the total number of vertices. A complete graph is small-world (with N vertices, the ratio is $1/N$); so is a star graph (ratio is $1/N$). A cyclic graph is not small-world (with N vertices, the ratio is $N/2$). Typically, the term is reserved for situation that the average distance grows no faster than the logarithm of the number of vertices.

Scale-free graphs, with exponents γ in the range of 2 to 3, are small-world. But not all small-world graphs are scale-free. A classic example of this is the small-world architecture of Watts and Strogatz (Nature 1998). These graphs are made by (i) starting with a cyclic graph, and (ii) adding a small number of long-range cross-connections.

An Erdos-Renyi graph is a graph in which the probability of a connection between any two nodes is set to some value p. For $p = 1$, this yields a complete graph. For $p < 1$, the graph may not be connected. There are numerous results on the behavior of these graphs as the number of vertices grows – for example, the probability that the graph is connected, the probability that it has a single large component, etc. For large N, $\frac{\ln N}{N}$ determines connectedness: for $p > \frac{\ln N}{N}$, the graph is almost always connected, for $p < \frac{\ln N}{N}$, the graph is almost always not connected.

The graph Laplacian

The graph Laplacian is a matrix that captures many aspects of a graph, including how dynamical processes on the graph evolve, and how the graph can best be represented by a one-dimensional structure. It is also central to many algorithms ("spectral partitioning", "spectral clustering") that use graphs for many other purposes (image segmentation, VLSI layout).

We will define the Laplacian for a simple (unweighted, un-directed) graph. The extension to a weighted graph is straightforward: instead of a 1 or a 0 to indicate whether an edge is present, one can insert a positive real number to indicate its strength. (One can consider the intermediate case – the possibility of a multiplicity of equally-weighted edges between each pair of points – and this corresponds exactly to the case of integer weights.) However, the extension of the graph Laplacian to a directed graph is not, as the symmetry of the in-connections and out-connections from a node is an essential feature.

The starting point is to consider a function on the graph, i.e., an assignment x_i of values to each vertex on the graph. We can think of each value x_i to represent an amount of material (or heat) at the vertex *i*. Now allow this material to diffuse, according to the graph's connectivity. The amount at each vertex at time *t* is given by $x_i(t)$. At each time step, some of the material goes to each of the d_i vertices *j* that is connected to *i*, but also, some of the material at these neighboring vertices flows back to *i*. So,

$$\frac{d}{dt}x_i = K\left(-d_i x_i + \sum_{j\sim i} x_j\right),$$

where the first term on the right represents the loss to neighboring vertices, and the second term represents the influx. The constant *K* is irrelevant (it provides an overall temporal scale, and can be set to 1), so we can write

$$\frac{d}{dt}\vec{x} = -L\vec{x},$$

where $\vec{x}$ is the column matrix of the x_i's and *L* is the *graph Laplacian*. The graph Laplacian is given by $L = D - A$, the difference of the diagonal matrix of the degrees (*D*) and the adjacency matrix.

A justification for calling *L* a Laplacian is that the above resembles the standard heat (or diffusion) equation: for $x(s,t)$ is the temperature of a conducting bar at position *s* and time *t*, the evolution is $\frac{\partial}{\partial t}x = K\frac{\partial^2}{\partial s^2}x$. The graph Laplacian, like the ordinary Laplacian, is proportional to the difference between the value at a point, and its local average. But there is a (conventional) sign difference: the graph Laplacian approximates the *negative* of the second derivative, while the continuous-system Laplacian, $\frac{\partial^2}{\partial s^2}$ or ∇^2, is the (positive) second derivative.

We now would like to solve this equation, given some set of initial conditions $\vec{x}(0)$. If we have a complete set of eigenvectors $\vec{\varphi}_m$ (i.e, a set that spans the vector space of functions on the graph), along with their associated eigenvalues λ_m (with $L\vec{\varphi}_m = \lambda_m\vec{\varphi}_m$, the solution is immediate.

Each $\vec{\varphi}_m$ corresponds to a solution that dies down exponentially, with rate λ_m:

$$\frac{d}{dt}\vec{\varphi}_m e^{-\lambda_m t} = -\lambda_m\vec{\varphi}_m e^{-\lambda_m t} = -L\vec{\varphi}_m e^{-\lambda_m t}.$$

So we write $\vec{x}(0)$ in terms of the basis set, $\vec{x}(0) = \sum_{m=1}^{N} c_m \vec{\varphi}_m$, and have a solution:

$\vec{x}(t) = \sum_{m=1}^{N} c_m \vec{\varphi}_m e^{-\lambda_m t}$.

The bottom line is that the eigenfunctions of the Laplacian indicate the patterns that die out exponentially, and the eigenfunctions with lowest nonzero eigenvalues die out the slowest – and thus, characterize the dynamics of functions that evolve on G.

There are some useful variants of the Laplacian that differ in row- or column-normalization.

The "normalized" Laplacian is defined as $L_{norm} = D^{-1/2} L D^{-1/2}$. Since $L_{norm} = D^{-1/2} L D^{-1/2} = D^{-1/2}(D - A)D^{-1/2} = I - D^{-1/2} A D^{-1/2}$, it is equal to the identity on its diagonal. For a weighted graph with "weak" connections, $L_{norm} = I - \varepsilon D^{-1/2} A D^{-1/2}$. Since this is close to the identity, one can approximate repeated applications of L_{norm} by $(L_{norm})^r = \left(I - \varepsilon D^{-1/2} A D^{-1/2}\right)^r \approx \left(\exp^{-\varepsilon D^{-1/2} A D^{-1/2}}\right)^r = \exp^{-\varepsilon r D^{-1/2} A D^{-1/2}}$. L_{norm} is symmetric, and its eigenvalues can also be shown to be non-negative (see below).

The "transition matrix" Laplacian allows for a connection with Markov chains: $L_{transition} = D^{-1} L = I - D^{-1} A$. $L_{transition}$ differs from the standard graph Laplacian in that its rows are divided by the degree. This means that the matrix now represents particle diffusion in the following sense: a particle can choose to leave a node, and if it does so, it chooses any of the edges randomly to go to the next node. To see the difference between this and the graph Laplacian considered above, consider a "star" graph with $N+1$ vertices. For the standard graph Laplacian, the function that is constant on all nodes is an eigenvector of eigenvalue 0. For the transition-matrix Laplacian, the function that has value *N* on the central vertex and value 1 on the peripheral ones is an eigenvector: on every step, a particle at the central vertex moves to any one of the peripheral ones, but a particle at any peripheral node must move centrally.

Eigenstructure of the Laplacian: null space, positive-definiteness

We first determine the eigenvectors corresponding to a zero eigenvalue via elementary means, then show that the other eigenvectors are positive, and then use some group theory to determine them in some interesting special cases. We focus on the standard graph Laplacian, but all the arguments work for the other variants.

The null space

A function on the graph is mapped to zero by the Laplacian if its value at every vertex is equal to the average of its neighbors. So a function that is constant on the graph is mapped to zero. Conversely, if a function is not equal to the average value of its nearest neighbors, then it is not mapped to zero.

If a graph is not connected, the Laplacian consists of blocks, one for each connected component. This is because *D* is diagonal, and *A* is in blocks. A function that is constant on each component separately (but perhaps has a different value on each component) is therefore mapped to zero by the Laplacian.

So the null space of the Laplacian is the set of functions that are constant on each component separately.

Positive-definiteness, the incidence matrix, and representing the graph on a line

The above comments determine the eigenvectors of eigenvalue 0 for the graph Laplacian. With a bit more work, we can show that all of the eigenvalues of the graph Laplacian are non-negative. We do this by representing it as a sum of outer products, and this leads to some other applications and interpretations of the graph Laplacian.

But before demonstrating positive-definiteness in this fashion, it is worthwhile noting that this property is to be expected from the diffusion interpretation. Using the negative of the Laplacian as the right side of a differential equation moves the value at a node towards the local average. Since this cannot lead to divergence, no time-dependence can grow without bound, so the negative of the Laplacian cannot have any positive eigenvalues.

To represent the Laplacian as a sum of outer products, we define the incidence matrix. The *incidence matrix* (or *vertex incidence matrix*) of a graph is a matrix *Q* whose rows correspond to the edges of the graph, and whose columns correspond to its vertices. It is defined as follows: For each row (edge) *r*, which consists of a connection between vertex *b* and vertex *c*, set $q_{rb} = 1$ and $q_{rc} = -1$, and $q_{rj} = 0$ for $j \notin \{b, c\}$. Note that the assignment of +1 or -1 to *b* or *c* is arbitrary, so the adjacency matrix is defined only up to sign-flips of each row. However, we will only be concerned with properties of *Q* that are independent of such sign-flips, especially properties of $Q^T Q$.

In the above case, and ordering the edges in lexicographic order along the rows, the incidence matrix is

$$Q = \begin{pmatrix} +1 & -1 & 0 & 0 & 0 & 0 \\ +1 & 0 & -1 & 0 & 0 & 0 \\ +1 & 0 & 0 & -1 & 0 & 0 \\ +1 & 0 & 0 & 0 & 0 & -1 \\ 0 & +1 & -1 & 0 & 0 & 0 \\ 0 & 0 & +1 & 0 & -1 & 0 \\ 0 & 0 & 0 & +1 & -1 & 0 \end{pmatrix} \quad \text{with (arbitrary) row order} \quad \begin{bmatrix} \{1,2\} \\ \{1,3\} \\ \{1,4\} \\ \{1,6\} \\ \{2,3\} \\ \{3,5\} \\ \{4,5\} \end{bmatrix}.$$

The basic observation that we need to demonstrate is that $L = Q^T Q$. To see this, i.e., to see that $Q^T Q = A - D$: On the diagonal (say in position *i*), $Q^T Q$ has a contribution of $(\pm 1)^2$ for

every edge that begins or ends at vertex *i*, which is the degree of that vertex. Off the diagonal, Q^TQ is nonzero only if column i and column *j* have a nonzero entry in the same position (e.g., row *r*). This means that there is an edge (the edge corresponding to row *r*) that starts at *i* and ends at *j*, or vice-versa. So these entries in *Q* must be opposite in sign, and $\left(Q^TQ\right)_{ij} = -1$, as required by $Q^TQ = D - A$. Conversely, $\left(Q^TQ\right)_{ij} = 0$ means that no edge connects *i* to *j*, and that the adjacency matrix is zero at that location, also as required by $Q^TQ = D - A$.

Since each row of *Q* sums to 0, the vector $\vec{1}$ consisting of a column of 1's is in the null space of *Q*, and hence, an eigenvector of Q^TQ with eigenvalue 0. (We already knew this, from the above comment about what is in the null space of the Laplacian.)

The relationship of *L* to *Q* now shows that all eigenvalues of *L* are non-negative. For say λ is an eigenvalue. Then $\vec{x}^T L\vec{x} = \vec{x}^T Q^T Q\vec{x} = \left(Q\vec{x}\right)^T Q\vec{x} = \left|Q\vec{x}\right|^2$ but also $\vec{x}^T L\vec{x} = \vec{x}^T \lambda\vec{x} = \lambda\vec{x}^T\vec{x} = \lambda\left|\vec{x}\right|^2$, so $\lambda \geq 0$.

The same argument shows that the normalized Laplacian is also positive-definite: $L_{norm} = D^{-1/2}LD^{-1/2} = D^{-1/2}Q^TQD^{-1/2} = \left(QD^{-1/2}\right)^T QD^{-1/2}$. From this, it follows that the eigenvalues of the transition-matrix Laplacian (an asymmetric matrix), are also non-negative, since $L_{transition} = D^{-1}L = D^{-1/2}D^{-1/2}LD^{-1/2}D^{1/2} = D^{-1/2}L_{norm}D^{1/2}$. So the transition-matrix Laplacian and the normalized Laplacian are similar matrices, and their eigenvalues are the same.

The relationship $L = Q^TQ$ provides another application of the Laplacian. Writing out $\vec{x}^T L\vec{x} = \vec{x}^T Q^T Q\vec{x}$ in coordinates yields $\vec{x}^T L\vec{x} = \vec{x}^T Q^T Q\vec{x} = \sum_{i\sim j}(x_i - x_j)^2$. The eigenvector of lowest positive eigenvalue minimizes this quantity, subject to the constraint $\left|\vec{x}\right|^2 = 1$.

The interpretation is as follows: let's say that you wanted to arrange the graph vertices at points x_i along a line, so that vertices that connected map to points that are close together. Then you would want to minimize the above expression, $\vec{x}^T L\vec{x} = \sum_{i\sim j}(x_i - x_j)^2$. Obviously you could do this by setting all $x_i = 0$, but this wouldn't preserve the structure of the graph. So it makes sense to add an overall size constraint, $\left|\vec{x}\right|^2 = 1$. Another way to see this connection is that the distribution of heat on the graph that dies out the slowest (other than the trivial distribution of equal amounts of heat on every vertex) is one in which there is the least difference in heat on connected nodes – since the heat difference is what drives the equalization. So we want to minimize $\sum_{i\sim j}(x_i - x_j)^2$.

Viewing $\vec{x}^T L\vec{x}$ as an ellipsoid, our task is to find its smallest axis, avoiding the one dimension in which it is flat. This is equivalent to finding the smallest eigenvector of L. We will see this below in deriving PCA. Also, it is part of a general way of approaching constrained-

minimization problems: the eigenvectors of L are the solutions of the Lagrange multiplier problem $\frac{d}{dx}\left(\vec{x}^T L\vec{x} - \lambda\vec{x}^T\vec{x}\right) = 0$.

That is, the map from vertex *i* to x_i is the map from the graph to the line that best approximates the graph structure: it minimizes the total squared distance between the vertices that are connected, given a constraint on the total extent of the map.

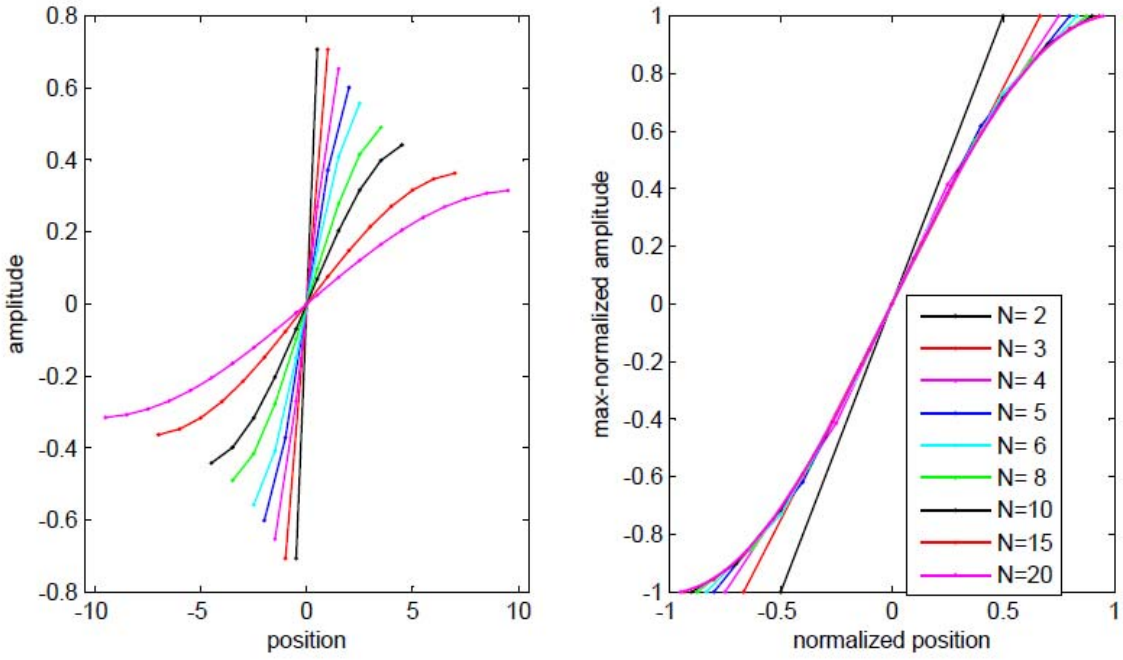
The idea extends: Consider a map *M* from each vertex *i* to the row vector $v_i = \left(\frac{1}{\sqrt{\lambda_p}}\left(\vec{\varphi}_p\right)_i, ..., \frac{1}{\sqrt{\lambda_q}}\left(\vec{\varphi}_q\right)_i\right)$ (where $\vec{\varphi}_p, ..., \vec{\varphi}_q$ are the eigenvectors of nonzero eigenvalue) is an embedding of the graph in a space of dimension $q - p + 1$. The first coordinate is proportional to the above mapping to the line; each subsequent coordinate is the best possible map to a line that is orthogonal to all previous mappings. A very hand-wavy argument as to why this set of coordinates is a good way to represent the graph: Let *C* be the matrix of coordinates of these vectors, i.e., $C_{i,r} = \left(v_i\right)_r = \frac{1}{\sqrt{\lambda_r}}\left(\vec{\varphi}_r\right)_i$, which consists of the eigenvectors $\vec{\varphi}_r$ in columns, with each column normalized by $\frac{1}{\sqrt{\lambda_r}}$. Then $C^T LC = C^T Q^T QC = C^T\left(\sum_m \lambda_m \varphi_m {\varphi_m}^T\right)C = I$. So in some sense, $C^T C$ is like the inverse of the Laplacian, which in turn indicates how easy it is to diffuse between two points. So when two vertices have coordinates (columns of *C*) with high covariance, they are close in the sense of diffusion on the graph.

The nonzero eigenvalues

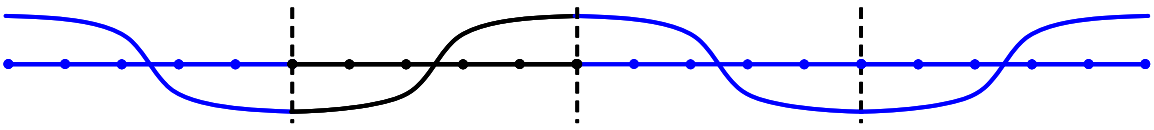
A simple calculation

One might think that for a graph whose topology is that of a line, the dominant eigenvector of the graph Laplacian is just a linear function on the line. But this can't actually be the case: where the value assigned to a node is equal to the average of its neighbors, the Laplacian goes to zero. So the Laplacian maps a linear function on the graph to a function that is zero everywhere, except at the endpoints. So unless $N = 2$ or $N = 3$, this cannot be an eigenvector.

So let's calculate the lowest eigenvector and eigenvalue of the Laplacian for a graph consisting of *N* points in a line. For *N*=2, the eigenvector (trivially) is [-1 1]; for *N*=3, almost as trivially, it is [-1 0 1]. But for $N \geq 4$, it is not a linear function of position on the graph, and, as the calculations show, it approaches a half cycle of a sinusoid, with a peak at one end of the graph, a zero-crossing in the middle, and a trough at the other end.



We can understand this asymptotic behavior as follows, via the "reflection" trick. For this graph, at every point except the endpoints, the graph Laplacian is a discrete approximation to the (negative of the) second derivative: the value at position x_j is replaced by $2x_j - (x_{j-1} + x_{j+1})$. At the endpoints, this is not the case – BUT – if we imagine adjoining reflections of the graph at each endpoint, the second-derivative behavior is recovered. More formally, we "guess" that the solution $\{x_1, ..., x_N\}$ is one section of an infinite line (black segment in figure below), where $x_{2N-j} = x_j$ (the mirror) and $x_{2N-2+j} = x_j$ (periodicity). So now a solution to the graph Laplacian heat equation is a segment of a solution to the standard Laplacian heat equation on the standard line, provided that this solution is mirror-symmetric at *N*, and periodic with period $2(N-1)$.



Since the heat equation on the line is translation-invariant, its solutions are the eigenfunctions of the translation operator – the sinusoids. For $x(s) = \exp(i\omega s)$, $Lx(s) = -\frac{\partial^2}{\partial s^2}\exp(i\omega s) = \omega^2 x(s)$. So the lowest nonzero eigenfunction is the one with the lowest frequency consistent with the mirror-symmetry condition, i.e,. a sinusoid with period $2(N-1)$.

This analysis properly suggests that group-theoretic considerations will help understand the solution of the heat equation on other graphs. If there is a permutation of the vertices that leaves the graph's connectivity unchanged, then it necessarily leaves the Laplacian unchanged as well.

So we can immediately calculate the eigenfunctions of the Laplacian on a cyclic graph with N elements – they are $\vec{x}_j^{[r]} = \exp(\frac{2\pi ijr}{N})$. To find the corresponding eigenvalues:

$$\left(L\vec{x}^{[r]}\right)_j = 2\left(\vec{x}^{[r]}\right)_j - \left(\vec{x}^{[r]}\right)_{j-1} - \left(\vec{x}^{[r]}\right)_{j+1} = e^{\frac{2\pi ijr}{N}}\left(2 - e^{-2\pi ir/N} - e^{+2\pi ir/N}\right) = \left(2 - e^{-2\pi ir/N} - e^{+2\pi ir/N}\right)\left(\vec{x}^{[r]}\right)_j$$

so $\lambda_r = \left(2 - e^{-2\pi ir/N} - e^{+2\pi ir/N}\right) = 2(1-\cos(\frac{2\pi r}{N})) = \sin^2(\frac{\pi r}{N})$. The smallest nonzero eigenvalue corresponds to $r = 1$.

Further implications of symmetry

Since we're dealing with graphs, there's the possibility for more "interesting" (i.e., larger, non-commutative) symmetry groups to be relevant. For the complete graph with N vertices, the full permutation group leaves the graph invariant. So now our task is to find out how the full permutation group acts in the space of function on N objects.

We recall that every finite group G has a specific, finite set of "irreducible representations", i.e., the distinct ways in which it can operate in a vector space V. These actions are defined by a sete of linear operators U_g in $Hom(V,V)$, which respect the group operation: $U_{gh} = U_g U_h$. For any nonzero vector v in that space, the set of images $U_g v$ must span the space. (If they did not span the space, then they would span a subspace of V preserved by G, and U would not be irreducible.

A main difference between commutative groups (for example, the cyclic group) and non-commutative groups (for example, the full permutation group with 3 or more elements) is that non-commutative groups contains irreducible representations of dimension 2 or more.

What are the consequences of this in the current set-up, when a group acts on a graph in a structure-preserving way? First, the action of the group on the graph induces an action of the group on functions of the graph. This means that there is a representation of the group (say, W) in the vector space X of functions on the graph. But also, since the group operation preserves the graph structure, it necessarily preserves the Laplacian. So for any group element g, $W_g L = L W_g$.

Let's say we have an eigenvector $\vec{\varphi}$ of L, with $L\vec{\varphi} = \lambda\vec{\varphi}$. Then for any group element g, $L\left(W_g\vec{\varphi}\right) = W_g L\vec{\varphi} = W_g\lambda\vec{\varphi} = \lambda\left(W_g\vec{\varphi}\right)$, so $W_g\vec{\varphi}$ is also an eigenvector of L, and has the same eigenvalue. Since there may be some irreducible subspaces of X that have dimension 2 or more, this means that two or more eigenvectors of L will be related by the group symmetry, and have the same eigenvalue. So this puts a premium on finding the irreducible components of how G acts in the space X of functions on the graph – for every irreducible component of dimension k, we will find k eigenvectors sharing the same eigenvalue.

Now consider the specific case of the complete graph of N points. The space of functions on the graph also has dimension N. The elements of the permutation group act on these functions via $N \times N$ permutation matrices. We can tell that this representation is *not* irreducible by a simple character calculation. Specifically, the number of copies of an irreducible representation M in a

given representation W is $\frac{1}{\#|G|}\sum_g \overline{\chi_M(g)}\chi_W(g)$, where $\chi_W(g) = \text{tr}(W_g)$. For permutation matrices, the trace is the sum of the number of items that are not relabeled. So for the representation W, $\chi_W(g) \geq 0$ everywhere, and, for some group elements, the inequality is strict. Now taking M to be the trivial representation E (that maps every group element to the number 1), it follows that there is at least one copy of E inside of W. So $W = E \oplus X$.

For representations of the full permutation group, there's a classic and general (but very involved) way to show that X is irreducible. Here we will use an elementary argument, by characterizing the matrices that commute with all elements of W. First, consider W_σ for a particular pair-swap permutation $\sigma = (pq)$. The corresponding W_σ is a matrix for which $(W_\sigma)_{p,q} = 1$, $(W_\sigma)_{q,p} = 1$, $(W_\sigma)_{j,j} = 1$ for $j \notin \{p,q\}$, and $(W_\sigma)_{j,k} = 0$ otherwise. If a matrix B commutes with W_σ, then it follows from $BW_\sigma = W_\sigma B$ that $B_{q,p} = B_{p,q}$. Therefore, a matrix that commutes with W_τ all pair-swap permutations must have all of its off-diagonal elements equal to the same value:

$B = \begin{pmatrix} b & c & \cdots & c \\ c & b & \ddots & \vdots \\ \vdots & \ddots & \ddots & c \\ c & \cdots & c & b \end{pmatrix}$. Finally, since the number of irreducible components of W is the number of linearly independent operators that commute with all W_g, the number of such components is 2 – one of dimension 1 (that corresponds to the trivial representation that maps all group elements to the identity) and one of dimension $N-1$, which consists of the "natural" representation in permutation matrices, but with the subspace corresponding to the identity removed. Since the Laplacian commutes with all of the matrices in this irreducible representation, it must act as a multiple of the identity – so the Laplacian has $N-1$ eigenvectors in this space, and they all have the same eigenvalue.

More explicitly: the eigenvector corresponding to the trivial representation is the vector $\vec{1}$ of all 1's, and it has eigenvalue 0. The other eigenvectors consist of any vector whose mean is zero, as this guarantees that they are orthogonal to $\vec{1}$. In this subspace, the Laplacian acts by replacing each element x_i by $(N-1)x_i - \sum_{j \neq i} x_j = Nx_i - \sum_j x_j = Nx_i$ (where we've used orthogonality, $\vec{x} \bullet \vec{1} = \sum_j x_j = 0$). So the eigenvalues in this space are N.

We can use a similar approach for other many other simple graphs. For example, in the star-topology, all of the peripheral vertices are equivalent under the action of the full symmetric group; the wagonwheel topology, the cube topology, and many others can be treated in a similar way.

Symmetry arguments are also useful when there is an approximate symmetry, as in this case, the exact eigenvalues and eigenvectors can be considered to be perturbations of the corresponding

quantities of the fully symmetric graph. So one can group the eigenvalues and eigenvectors in a meaningful way.

Embedding in a simpler context

It is useful to compare the calculation of the eigenvectors of the Laplacian with basic procedures for exploratory analysis of multidimensional data that arise in a simpler context: principal components analysis and multidimensional scaling. In principal components analysis (PCA), the data already have coordinates, and we look for a dimensionally-reduced set of coordinates that still represents the same data well. Multidimensional scaling is a bridge from PCA to Laplacian embedding – in MDS, one knows the distances between the datapoints but not their coordinates, and seeks a dimensionally-reduced representation. (For the graph Laplacian, one doesn't know distances, one just knows which vertices are connected.)

This material is adapted from MVAR1011.pdf

Principal Components Analysis

Principal components analysis can be thought of as finding a set of coordinates that do the best job of representing a high-dimensional dataset in a lower dimension.

The setup is that we have *k* observations, each of which yield a data point (or dataset) in a *n*-dimensional space. Each dataset is an *n*-element column vector $\vec{y}_j$, consisting of the observations $y_{1,j},\ldots,y_{n,j}$; we write a set of these column vectors together as a $n\times k$ matrix *Y*. We seek a representation in a space of *p* dimensions, where *p* is much smaller than *n* and *k*. That is, we seek a set of p column vectors $\vec{x}_1,\ldots,\vec{x}_p$, aiming to choose them so that linear combinations of these *p* column vectors are good approximations to the original datasets $\vec{y}_j$. To formalize this: the set of p column vectors $\vec{x}_1,\ldots,\vec{x}_p$ constitute an unknown $n\times p$ matrix, *X*. We want to choose them so that they explain as much of the data as possible, i.e., that we can find an associated set of coefficients *B* for which $Y-XB$ is as small as possible, for some $p\times k$, matrix *B*. If *p* is much smaller than both *n* and *k*, then we have found a concise reprentation of the dataset, since *Y* is $n\times k$ but *X* and *B* are $n\times p$ and $p\times k$. Geometrically, the columns of *X* define a *p*-dimensional subspace that accounts for the data. We can also think of the columns of *X* as "regressors", and that we seek regresors that simultaneously accout for the multivariate data.

Typically, one carries out PCA for a range of values of *p*, and then chooses a useful one based on considerations such as the fraction of the data explained, and the intended use (visualization, noise removal, etc.)

We note at the outset that we can't hope to determine *X* uniquely: alternative *X*'s whose columns span the same space will give an equivalent solution. Put another way, for any invertible $p\times p$ matrix *T*, $XB=XTT^{-1}B$, so $X'=XT$ and $B'=T^{-1}B$ can replace *X* and *B*.

One consequence is that we can always assume that the columns of *X* are orthonormal. The solution is still ambiguous (since in the above, *T* can still be taken to be a unitary matrix), but this turns out to be very helpful in finding the solutions.

Solution, several ways

Our goal is to minimize $R^2 = \mathrm{tr}\left((Y - Y^{fit})^*(Y - Y^{fit})\right)$, where $Y^{fit} = XB$ and *X* and *B* are both is unknown. If we knew *X*, then B could be found immediately – it is the projection of the datasets Y into the subspace spanned by *X*. We have already seen that this is $B = (X^*X)^{-1}X^*Y$.

Since we can assume that the columns of *X* are orthonormal, it follows that $X^*X = I$, and that $B = X^*Y$, and $Y^{fit} = XX^*Y$. Thus,

$$R^2 = \mathrm{tr}\left((Y - XX^*Y)^*(Y - XX^*Y)\right)$$

$$= \mathrm{tr}\left(Y^*Y - Y^*XX^*Y - Y^*XX^*Y + Y^*XX^*XX^*Y\right),$$

$$= \mathrm{tr}\left(Y^*Y - Y^*XX^*Y\right)$$

where we have again used $X^*X = I$ in the second equality. Thus, minimizing R^2 is equivalent to maximizing $\mathrm{tr}\left(Y^*XX^*Y\right)$, subject to the constraint that $X^*X = I$. Note that because $\mathrm{tr}(AB) = \mathrm{tr}(BA)$, this is equivalent to maximizing $\mathrm{tr}\left(YY^*XX^*\right)$, and also to maximizing $\mathrm{tr}\left(XX^*YY^*\right)$ and $\mathrm{tr}\left(X^*YY^*X\right)$.

At this point, we note that since YY^* is a symmetric matrix, it (typically) has a full set of eigenvectors and eigenvalues. These are the natural, data-driven coordinates for our problem. It therefore makes sense to write a potential solution for *X* in terms of these eigenvectors.

We will find that the columns of *X* must be the eigenvectors of YY^*. There are two ways to see this. The first is to write out a possible solution for *X* in terms of these eigenvectors. This leads to a coordinate-based calculation, which eventually yields the desired result (see pp. 10-12 of MVAR01-MVAR18, 2008-2009 notes for this strategy).

Alternatively, we could use the method of Lagrange Multipliers, which leads to the same conclusion, but in a much more systematic way.

In its simplest form, we can seek a solution for *p*=1: that is, find the direction that accounts for the largest fraction of the variance. With this direction corresponding to the unknown vector $\vec{x}$, this means maximizing $\mathrm{tr}\left(YY^*\vec{x}\vec{x}^*\right) = \mathrm{tr}\left(\vec{x}^TYY^*\vec{x}\right)$ subject to $|x|^2 = \vec{x}^T\vec{x} = 1$. In the Lagrange formalism, this is equivalent to maximizing $F = \vec{x}^TYY^*\vec{x} - \lambda\vec{x}^T\vec{x}$. Setting derivatives w.r.t the components of $\vec{x}$ to zero (ie., $\frac{\partial F}{\partial x_j} = 0$)yields the eigenvalue equation $YY^*\vec{x} = \lambda\vec{x}$. So the unknown vector $\vec{x}$ is an eigenvector. We could have already seen this, by expressing YY^* in its eigenbasis: $YY^* = \sum_{i=1}^{p}\lambda_i\vec{\varphi}_i\vec{\varphi}_i^{\,T}$.

If we try to find a solution for $p>1$, the algebra is identical. Here, our constraints ($X^*X=I$) can be thought of as a matrix of constraints, one for each element of X^*X . Thus, the Lagrange term (a sum of unknown coefficients multiplied by each constraint) can be compactly written as $\mathrm{tr}(\Lambda X^*X)$, for some $p\times p$ matrix Λ .

Thus, maximizing $\mathrm{tr}\left(YY^*XX^*\right)$ subject to $X^*X=I$ is equivalent to maximizing $F=\mathrm{tr}\left(YY^*XX^*\right)-\mathrm{tr}(\Lambda X^*X)$ without constraints on *X*, and choosing Λ so that $X^*X=I$ at the maximum. To do this, we calculate $\frac{\partial F}{\partial x_{j,m}}$, and put the resulting $n\times p$ equations, $\frac{\partial F}{\partial x_{j,m}}=0$, into a matrix. This yields (see p. 16 of MVAR01-MVAR18, 2008-2009 notes for details) $YY^*X=X\Lambda$.

Now it is obvious that if we choose Λ to be a diagonal matrix consisting of the eigenvalues $\lambda_1,\ldots\lambda_p$ of YY^*, then choosing the columns of X to be the associated eigenvectors (the $\vec{\varphi}_i$) satisfies both the minimization equation $YY^*X=X\Lambda$ and the constraints, because the eigenvectors of a symmetric matrix are orthogonal.

Which eigenvectors and eigenvalues to choose? Making use of the fact that X satisfies $YY^*X=X\Lambda$, it follows that

$$\mathrm{tr}\left(YY^*XX^*\right)=\mathrm{tr}\left(X\Lambda X^*\right)=\mathrm{tr}\left(X^*X\Lambda\right)=\mathrm{tr}\,\Lambda=\sum_{i=1}^{p}\lambda_i\,.$$

So if one is to choose *p* eigenvalues, one should choose the *p* largest ones of YY^*.

In sum, the best approximation (in the least-squares sense) of an $n\times k$ matrix *Y* by a product XB of an $n\times p$ matrix *X* and a $p\times k$ matrix *B* is to choose the columns of X to be the eigenvectors corresponding to the *p* largest eigenvalues of YY^*, and to choose $B=X^*Y$. The unexplained variance is the sum of the remaining eigenvalues of YY^*.

An important computational "note" is that this problem is symmetric in *n* and *k*, and this symmetry reflects the fact that the eigenvalues of YY^* are the same as those of Y^*Y . But when *n* and *k* differ greatly in size, one problem may be computationally much easier than the other.

This is implemented by matlab's 'princomp.m'.

Multidimensional scaling

In contrast to the above situation, the setup in multidimensional scaling is that we are given a set of dissimilarities d_{ij} as distances between points whose coordinates are as-yet unknown, and we are to determine the coordinates. For example, the d_{ij} could be the result of a survey of raters that are asked to compare stimuli *i* and *j*. But they also could be the vector-space distances

between measurements $\vec{y}_i$ and $\vec{y}_j$, i.e., $d_{ij} = \left|\vec{y}_i - \vec{y}_j\right|$ (but we are not given the vectors $\vec{y}_i$). Our problem is to find a representation of the d_{ij} as Euclidean distances. That is, we seek a set of vectors $\vec{x}_i = (x_{1,i},...,x_{R,i})$, for which

$$d_{ij}{}^2 = \left|\vec{x}_i - \vec{x}_j\right|^2 = \sum_{r=1}^{R} \left|x_{i,r} - x_{j,r}\right|^2 . \quad (1)$$

The embedding dimension *R* is not known, and we may want to choose a value of *R* for which eq. (1) is only approximately true. The distances and coordinates of the $\vec{x}_i$ are assumed to be real numbers. We are of course only interested in solutions in which the embedding dimension *R* is substantially less than the number of data points, *N*.

We use a trick (due I think to Kruskal) to turn this problem into an eigenvalue problem. The first observation is that the solution (1) is non-unique in two ways. First, as with most of the above problems, it is ambiguous up to rotation – for any rotation matrix *M*, $\left|M\vec{x}_i - M\vec{x}_j\right| = \left|\vec{x}_i - \vec{x}_j\right|$. But also, we can add an arbitrary vector to each of the $\vec{x}_i$: $\left|\left(\vec{x}_i + \vec{b}\right) - \left(\vec{x}_j + \vec{b}\right)\right| = \left|\vec{x}_i - \vec{x}_j\right|$. Because of this, we can restrict our search to a set of vectors $\vec{x}_i$ whose mean is zero.

We next note that if eq. (1) holds and also that vectors $\sum_i \vec{x}_i = 0$, we can write an equation for the inner products $\vec{x}_i{}^T \vec{x}_j$ in terms of the d_{ij}. Beginning with $d_{ij}{}^2 = \left|\vec{x}_i - \vec{x}_j\right|^2 = \left(\vec{x}_i - \vec{x}_j\right)^T \left(\vec{x}_i - \vec{x}_j\right) = \vec{x}_i{}^T \vec{x}_i + \vec{x}_j{}^T \vec{x}_j - 2\vec{x}_i{}^T \vec{x}_j$, we note that

$\frac{1}{N}\sum_{j=1}^{N} d_{ij}{}^2 = \frac{1}{N}\sum_{j=1}^{N}\left(\vec{x}_i{}^T \vec{x}_i + \vec{x}_j{}^T \vec{x}_j - 2\vec{x}_i{}^T \vec{x}_j\right) = \vec{x}_i{}^T \vec{x}_i + S$, where $S = \frac{1}{N}\sum_{k=1}^{N} \vec{x}_k{}^T \vec{x}_k$.

Similarly, $\frac{1}{N}\sum_{i=1}^{N} d_{ij}{}^2 = \frac{1}{N}\sum_{i=1}^{N}\left(\vec{x}_i{}^T \vec{x}_i + \vec{x}_j{}^T \vec{x}_j - 2\vec{x}_i{}^T \vec{x}_j\right) = \vec{x}_j{}^T \vec{x}_j + S$, and

$\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} d_{ij}{}^2 = \frac{1}{N}\left(\sum_{j=1}^{N}\left(\vec{x}_j{}^T \vec{x}_j + S\right)\right) = 2S$. So, if there are vectors $\vec{x}_i$ for which eq. (1) holds, then

$$\vec{x}_i{}^T \vec{x}_j = \frac{1}{2}\left(-d_{ij}{}^2 + \frac{1}{N}\sum_{i=1}^{N} d_{ij}{}^2 + \frac{1}{N}\sum_{j=1}^{N} d_{ij}{}^2 - \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} d_{ij}{}^2\right). \quad (2)$$

We therefore write $G_{ij} = \frac{1}{2}\left(-d_{ij}{}^2 + \frac{1}{N}\sum_{i=1}^{N} d_{ij}{}^2 + \frac{1}{N}\sum_{j=1}^{N} d_{ij}{}^2 - \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} d_{ij}{}^2\right)$, which is entirely determined by the given distances, and seek a set of vectors a set of vectors $\vec{x}_i = (x_{1,i},...,x_{R,i})$ for which $\vec{x}_i{}^T \vec{x}_j = G_{ij}$, i.e., $G_{ij} = \sum_{r=1}^{R} x_{r,i} x_{r,j}$. This is equivalent to the matrix equation $G = X^T X$, where each vector $\vec{x}_i$ forms a column of *X*.

This yields an immediate formal solution: we write *G* in terms of its normalized eigenvectors, $G = \sum_{i=1}^{N} \lambda_i \vec{v}_i^{\,*} \vec{v}_i$, and then take $\vec{x}_i = \sqrt{\lambda_i}\vec{v}_i$. The λ_i 's, which can be taken in descending order, indicate the importance of each coordinate in the representation (1).

The above allows for a parallel to the use of the graph Laplacians to characterize or embed a graph. The elements of *G* play the same role as the elements of the Laplacian: the rows and columns sum to zero, and the diagonals must be positive. The values off of the diagonal express distances, but here the correspondence is not so close: in MDS, a zero value indicates that distance is "typical", positive values connect points that are closer than typical. For the graph Laplacian, 0 values indicate points that are disconnected, and negative values indicate nodes that are linked.

Multidimensional scaling as described above works fine provided that all the eigenvalues are non-negative (since we want to find real coordinates). But there is no guarantee that this is the case for multidimensional scaling. This is one reason for using a Laplacian-like approach to deal with similarity data – in the graph-theoretic scenario, all eigenvalues must be positive – but the tradeoff is that one is no longer representing distances, just something like distances, as implied by the diffusion idea.

In standard multidimensional scaling, the presence of negative values of the eigenvectors indicate that no Euclidean representation is possible (i.e., no distance-preserving embedding in a Euclidean space is possible). Instead, the representation (1) must be generalized to

$$d_{ij}^{\,2} = \sum_{r=1}^{R} \varepsilon_r \left| x_{i,r} - x_{j,r} \right|^2 , \qquad (3)$$

where $\varepsilon_r = +1$ along the Euclidean dimensions ($\lambda_r > 0$, $\vec{x}_i = \sqrt{\lambda_i}\vec{v}_i$), and $\varepsilon_r = -1$ along the non-Euclidean dimensions ($\lambda_r < 0$, $\vec{x}_i = \sqrt{-\lambda_i}\vec{v}_i$). The non-Euclidean dimensions can be considered to describe an intrinsic aspect of the geometry of the original data. Alternatively, if all that is desired is a representation of the rank order of the distances, it is always possible to "cure" the non-Euclidean-ness by replacing the original distances d_{ij} by some power of them, $\left(d_{ij}\right)^a$. For a power *a* that is sufficiently close to 0, the non-Euclidean-ness goes away.

Combinatorial measures

We now return to graphs, and consider some examples of "combinatorial" measures – measures that emphasize counting connections, rather than the algebraic structure of *A*.

Global clustering coefficient

The *global clustering coefficient C* is a way of measuring whether the connectivity is mostly local (highly "clustered") or not. It is defined as three times the ratio of the number of triangles

to the number of connected triples, where a "connected triple" is a subset of nodes *i*, *j*, and *k* for which $i \sim j$ and $j \sim k$. The factor of 3 is because every triangle necessarily has three connected triples (each cyclic order). A complete graph has $C = 1$, a triangle-free graph has $C = 0$.

For a random graph (an Erdos-Renyi graph) in which nodes are connected with probability *p*, we can calculate that the expected number of connected triples is $\frac{N(N-1)(N-2)}{2} p^2$ and the expected number of triangles is $\frac{N(N-1)(N-2)}{6} p^3$, so the global clustering coefficient is *p*. More generally, the global clustering coefficient is not "normalized" for the overall connection density of the graph. It is also not straightforward to extend the idea of a global clustering coefficient to a graph with weighted edges.

Community structure

Community structure is an approach advanced over the last decade to analyze graphs that arise in many different contexts. The idea is to attempt to partition the vertices into "communities", in a way that most adjacency relationships are within communities, rather than between. See Newman (PNAS 2006, http://www.pnas.org/content/103/23/8577.full.pdf+html) for an application to social networks. For an application to fMRI, see Bassett… Porter… Grafton, http://www.pnas.org/content/early/2011/04/15/1018985108.full.pdf+html). We use Porter's normalization conventions but the algebraic approach of Newman.

A community structure consists of an assignment of each vertex *i* to a community c_i. The extent to which this assignment captures the graph structure is determined by a "quality function", typically
$Q = \sum_{i,j} \left(a_{ij} - p_{ij} \right) \delta(c_i, c_j)$.
The quantities a_{ij} are the entries in the adjacency matrix. The quantities p_{ij} correspond to the expected probability of a connection between *i* and *j*, subject to a null hypothesis. The natural null hypothesis is $p_{ij} = \frac{d_i d_j}{2m}$, where *m* is the total number of edges. This is the expected probability of a connection, given the total number of edges in the graph, and the number of edges present at *i* and at *j*, and no further organization. Put another way, if you create an adjacency matrix that is as random as possible given a specified set of degrees at each vertex, then its off-diagonal entries would be $p_{ij} = \frac{d_i d_j}{2m}$. Note that $m = \frac{1}{2} \sum_{i,j} a_{ij} = \frac{1}{2} \sum_q d_q$, since counting the degree of each vertex counts each connection twice.

Q is large if edges occur within a community in a way that is more than chance ($a_{ij} > p_{ij}$ if $\delta(c_i, c_j) = 1$). In the trivial case that all vertices are assigned to the same community,

$$Q = \sum_{i,j}\left(a_{ij} - p_{ij}\right) = \sum_{i,j}\left(a_{ij} - \frac{d_i d_j}{2m}\right) = \sum_{i,j} a_{ij} - 2m\sum_i \frac{d_j}{2m} = \sum_{i,j} a_{ij} - \sum_i d_i = \sum_{i,j} a_{ij} - 2m = 0.$$

Note that any permutation that respects the symmetry of the graph results in a reassignment of communities that leaves *Q* invariant.

One difficulty in applying this approach is that the general problem of determining the community structure that maximizes *Q* is intractable – specifically, it is an NP-hard problem, equivalent in difficulty to the traveling salesman problem. However, finding approximately optimal community structures (an informal concept) is much easier. Here we outline the Newman (2006) approach.

The approach is "greedy" – the first step is to find the best partitioning into two communities; at each subsequent step, the existing community structure is the starting point and the algorithm seeks to divide one of the existing communities into two. There is no guarantee that the optimal community structure can be identified this way, so this is the first sense in which the algorithm is inexact. ("Greedy" algorithms typically have this property.) At each stage, the algorithm can terminate – by indicating that no further subdivision improves *Q*. Newman's algorithm consists of an explicit eigenvector computation at each stage, but he also suggested ways to refine each stage by a local search, in which individual vertices are reassigned to the alternative community.

The key observation is that, for the special case of dividing a graph into two communities, we can rewrite *Q* as a quadratic form. Let the community structure be determined by $s_i = \pm 1$, where the two signs correspond to the two communities. Then, $\delta(c_i, c_j) = \frac{1}{2}(s_i s_j + 1)$. As noted above, $\sum_{i,j}\left(a_{ij} - p_{ij}\right) = 0$ (i.e., for the assignment of all vertices to the same community, $Q = 0$), so

$$Q = \frac{1}{2}\sum_{i,j}\left(a_{ij} - p_{ij}\right)s_i s_j .$$

Now using the s_i's to constitute a column vector $\vec{s}$, and *B* to be the matrix whose entries $b_{ij} = a_{ij} - p_{ij}$ express the difference between the true adjacency matrix and the "null hypothesis" matrix, we have

$$Q = \frac{1}{2}\vec{s}^T B\vec{s} .$$

The problem of finding the optimal community structure is now the problem of maximizing this quadratic form, subject to the constraint that all of the components of $\vec{s}$ are ± 1. If, instead, the constraint was that $|\vec{s}| = 1$, we would simply find the eigenvector of *B* that corresponds to its largest eigenvalue. So here is the second place at which the algorithm is heuristic rather than exact: the Newman procedure identifies this eigenvector $\vec{s}_{max}$, and then sets s_i to be ± 1

according to the sign of $\left(\vec{s}_{max}\right)_i$. This yields the vector $\vec{s}$ that is most closely aligned with $\vec{s}_{max}$ (in the dot-product sense), but not necessarily the one that yields the largest *Q*. Of course it makes sense that these are similar – and in specific examples, as checked by an exhaustive search, this is the case – but it is not guaranteed. The procedure to be followed when $s_i = 0$ is unspecified, but there are some obvious choices: choose randomly, or do an exhaustive search.

The numerical values of $\left(\vec{s}_{max}\right)_i$ have a meaning – they are the extent to which each vertex contributes to the community structure.

As is the case with the graph Laplacian, the matrix *B* is left invariant by any relabeling of the graph vertices that leaves the connectivity matrix invariant. So we can use group-theoretic tolls to find *B*'s eigenvalues in many simple cases (complete graph, cyclic graph, star graph, etc.).

Importantly, the matrix *B* could be negative-definite -- there may be no eigenvectors with positive eigenvalue. In this case, the Newman procedure simply terminates – as it can find no community structure that increases *Q* above its value for a single-community assignment. As an example, for the complete graph of size *N*, $a_{ij} = 1$ and $p_{ij} = \frac{(N-1)(N-1)}{N(N-1)/2} = \frac{2(N-1)}{N}$, so for $N \geq 3$, all off-diagonal $b_{ij} < 0$. In this case, the algorithm stops, as no eigenvector will increase the quality *Q* of the community structure.

But the situation is more complex: because the a true eigenvector of *B* is replaced by one whose entries are coordinates are ± 1, it is possible that *B* has a positive eigenvector, but none of the community assignments increase *Q*. In this case, the algorithm again terminates. An example of this is a graph with a "wagonwheel" connectivity and 6 vertices (5 peripheral): *B* has an eigenvalue of $\left(\sqrt{5}-1\right)/2 \approx 0.618$; this corresponds to an assignment of 3 vertices to each community and a decrement of *Q* by 0.2.

What may be even more of a concern – but this relates more to the definition of *Q* than to the algorithm -- is that *Q* may be large even for graphs in which the community structure does not seem relevant. For example, partitioning a cyclic graph into two connected halves increases *Q* by $N-4$. But there is no particular reason to choose any of the many ways that one can hemisect the graph; all lead to an equal increase in *Q*. are equally good. Put another way, the matrix *B* has several eigenvectors that are identical; they are guaranteed to be identical because of the symmetry of the graph. So this suggests a measure of robustness of the community assignment: if the next-largest eigenvalue of *B* is close to the largest one, the partitioning is not robust.

This also suggests an algorithmic tweak: when there is no gap, or only a small gap, between the largest eigenvalue of *B* and the next-largest eigenvalue, these top eigenvectors should be considered simultaneously. There are at least two ways of doing this: first, a binary the community assignment could be carried out by finding the vector containing ± 1's that is closest to the hyperplane that they span; second, one could look for a non-binary partition in that subspace (as in the way that the Fisher discriminant is used for more than binary partitioning).

It is interesting to note that the "community" approach becomes tractable (i.e., reduces to a set of eigenvalue problems) only if one ignores the combinatorial aspects of the problem – or at least, uses the heuristics that they can be approximated by an algebraic approach. It is also interesting to note that the above eigenvalue problem is very similar to the one that arises in the Fisher discriminant, where the problem is to find an axis that maximally separates two clusters. The difference is that in the Fisher problem, one is given coordinates of the points (and from this, one can compute their distances). Here, one is only given the distances between the points (effectively 1 if connected, large if not connected), and one needs to put them into a space first, so that one can find directions.

Analogous to the Fisher problem, one wonders whether a less-greedy approach , in which one allows for a multipart subdivision at a single stage if *B* has more than one positive eigenvalue – would be a useful generalization.

Linear discriminant analysis, a.k.a. Fisher discriminant

In discriminant analysis, rather than try to find the best coordinates to represent a dataset (as in MDS), we seek the best coordinates to distinguish one subset from another – hence the analogy with graph partitions. The idea extends readily to more than two subsets.

Let's say there are a total of *n* samples of multivariate data *X* , with the samples tagged as belonging to two subsets, n_1 in the first subset and n_2 in the second. Say the two subsets have means $\vec{\mu}^{[1]} = \frac{1}{n_1}\sum_{i=1}^{n_1}\vec{x}_i^{[1]}$, and $\vec{\mu}^{[2]} = \frac{1}{n_2}\sum_{i=1}^{n_2}\vec{x}_i^{[2]}$, and the global mean is $\vec{\mu} = \frac{n_1\vec{\mu}^{[1]} + n_2\vec{\mu}^{[2]}}{n_1 + n_2}$, all row-vectors. We want to find a linear function of the coordinates that does the best job of separating these two clouds of data. That is, we want to discriminate these subsets by their projections onto a (row) vector $\vec{v}$. That means, we want to maximize the difference of the projections of the means, while, simultaneously minimizing the scatter *within* the groups, as projected onto $\vec{v}$.

The setup extends to *C* classes. The variance between the group means, after projection onto $\vec{v}$, is $V_{between} = \sum_{c=1}^{C}\left|\vec{v}\bullet\left(\vec{\mu}^{[c]} - \vec{\mu}\right)\right|^2$. The variance within group *j* is $V_c = \frac{1}{n_c}\sum_{i=1}^{n_c}\left|\vec{v}\bullet\left(\vec{x}_i^{[c]} - \vec{\mu}^{[c]}\right)\right|^2$, so the total within-group variance is $V_{within} = \sum_{c=1}^{C}V_c$. We want to find directions that maximize the ratio of $V_{between}$ to $V_{within} = \sum_{c=1}^{C}V_c$. It doesn't make sense to simply maximize $V_{between}$; we could do this in an "empty" way just by magnifying $\vec{v}$. Simultaneously controlling V_{within} takes care of this, and ensures that we find we focus on the direction of $\vec{v}$, not its size.

To solve the problem, we could try a "brute-force" method of finding $\vec{v}$ that maximizes the ratio $V_{between}/V_{within}$. Or, we could attempt to maximize $V_{between}$ subject to the constraint that V_{within} is constant. (The specific constant doesn't matter, since it just multiplies $\vec{v}$ by a constant.)

The latter is more practical. Setting it up as a Lagrange Multiplier problem, our job is to minimize $V_{between} + \lambda V_{within}$. Each of the terms is quadratic in v, so derivatives will be linear. This leads to

$$\left(\sum_{c=1}^{C} \left(\vec{\mu}^{[c]} - \vec{\mu} \right)^* \left(\vec{\mu}^{[c]} - \vec{\mu} \right) \right) \vec{v}^* = \lambda \left(\sum_{c=1}^{C} \frac{1}{n_c} \sum_{i=1}^{n_c} \left(\vec{x}_i^{[c]} - \vec{\mu}^{[c]} \right)^* \left(\vec{x}_i^{[c]} - \vec{\mu}^{[c]} \right) \right) \vec{v}^* .$$

This is an equation of the form $Az = \lambda Bz$ (for $z = \vec{v}^*$), where A is the between-group covariance, and B is the within-group covariance. A has rank $C-1$, since $\sum_{c=1}^{C} n_c \left(\vec{\mu}^{[c]} - \vec{\mu} \right) = 0$ (i.e., the weighted mean of the within-group means is the global mean.)

For the two-group case, it is easy to solve. In this case, A has rank 1, so Bz must be within the one-dimensional subspace in the range of A, namely, $\left(\vec{\mu}^{[1]} - \vec{\mu} \right)^*$, which is necessarily a scalar multiple of $\left(\vec{\mu}^{[2]} - \vec{\mu} \right)^*$ and also $\left(\vec{\mu}^{[2]} - \vec{\mu}^{[1]} \right)^*$. Since Bz must be proportional to $\left(\vec{\mu}^{[2]} - \vec{\mu}^{[1]} \right)^*$, it follows that $\vec{v}^* = z = B^{-1} \left(\vec{\mu}^{[2]} - \vec{\mu}^{[1]} \right)^*$. More generally ($C > 2$), we seek eigenvectors of $B^{-1}A$, which are guaranteed to be in the span of the columns of A.

These solutions are known as “canonical variates”; they express the variables in which the classes are most cleanly discriminated.

Comparing community structures

Often one has several observations of data that one represents as a graph – e.g., measures of functional connectivity on several occasions, or on different tasks, or in different subjects. Given such data, one might want to compare the graphs, or, one might want to carry out a joint analysis to determine which set(s) of connections are robust across conditions.

It can’t be easier than sequence comparisons.