Semidefinite Embedding

Applied to Visualizing Folksonomies

CS6772 Project Proposal

Blake Shaw <bs2018@columbia.edu>

December 13th, 2005

1 Abstract

This paper investigates using Semidefinite Embedding (SDE) to visualize data collected from a folksonomy. The del.icio.us social bookmarking service is a perfect example of a folksonomy; a community of users label websites with descriptive tags. Each tag exists in a high-dimensional space corresponding to the frequency of use of that tag among all the users of the system. We are motivated by the following question: can we find a simple low-dimensional structure for these tags that captures the significant relationships inherent in the data? In this paper we explore Semidefinite Embedding, an algorithm for non-linear dimensionality reduction, and its application to visualizing folksonomic systems, focusing on the effects of specifying different levels of connectivity for the data and the heuristics which can be used to find the best parameters for the algorithm.

2 Introduction

2.1 Semidefinite Embedding

Semidefinite Embedding (also known as Maximum Variance Unfolding) is an algorithm for nonlinear dimensionality reduction. Given a Gram matrix consisting of the distances between N points, SDE finds a low-dimensional manifold that best approximates the high-dimensional space in which the data exists.

2.1.1 Steps of SDE

Given a Gram matrix $G_{i,j}$ for $i = \{1, 2, ...N\}$ and $j = \{1, 2, ...N\}$ such that $G \geq 0$

- Calculate a connectivity matrix C such that the $C_{i,j} = 1$ for points that are close to each other in G. This connectivity matrix specifies a neighborhood graph for the N points and is often created by connecting each point to its k-nearest neighbors.
- Formulate the problem to be solved as a semidefinite programming problem to find the Gram matrix K with the maximum variance that satisfies the constraints specified by the connectivity matrix and a centering constraint. The constraints from the connectivity matrix act to preserve local distances as best as possible.

$$\begin{aligned} &\text{Maximize } Tr(K) \text{ subject to } K \geq 0, \\ &\Sigma_{i,j} K_{i,j} = 0 \\ &\text{and } \forall\{i,j\} \text{ such that } C_{i,j} = 1 \\ &K_{ii} + K_{jj} - K_{ij} - K_{ji} = G_{ii} + G_{jj} - G_{ij} - G_{ji} \end{aligned}$$

- After K is found by the semidefinite programming package, calculate the low-dimensional coordinates through spectral embedding. By performing an eigenvalue decomposition on K, we can find the d eigenvectors which correspond to the highest eigenvalues and use these as the low-dimensional coordinates.

2.2 Del.icio.us Data

I propose using SDE to visualize the relationships between tags used to describe websites. Del.icio.us is a web-based social bookmarking system. Each user adds bookmarks and then labels them with tags. For example, a user might bookmark *apple.com* and tag it with *computer, tech, design.* Del.icio.us is a perfect example of a folksonomy (see figure 1). Users tag small parts of the web, and then when all of these tags are aggregated, a large comprehensive set

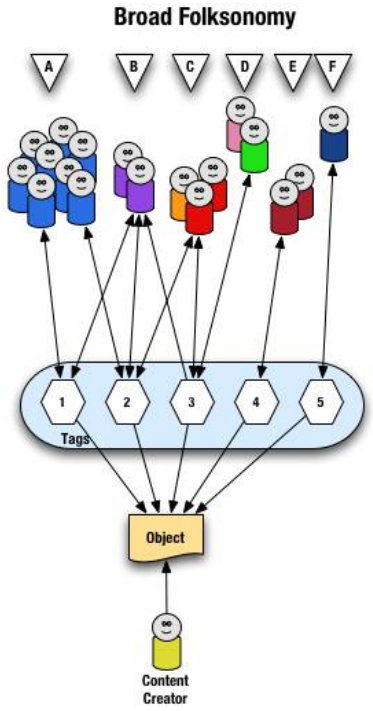

Figure 1: Structure of Del.icio.us. Users tag objects (in this case links) with a vocabulary of tags. [1]

of metadata is produced in the form of tags. However, because these tags are now contributed from many different individuals and aggregated, useful information comes not only from the tag itself but also from the information about who contributed to labeling the content with that tag. This information about how users apply tags can be analyzed to calculate the similarity between tags.

2.2.1 Details

Since del.icio.us is an open system, allowing anyone to view other users' bookmarks, I was able to collect a dataset which consists of 200 tags and their frequency of use by 1494 users. This data can be thought of as N points existing in D dimensions, where N is the number of tags, and each dimension represents a user. We can then calculate the distance between each of these N points using a distance metric such as Euclidean distance, KL-divergence, or an RBF kernel. For this paper I focus mainly on using KL-divergence although I experimented with other metrics as well. By applying the KL-divergence metric to each pair of tags represented as probability distributions p and q, we are able to calculate the initial Gram matrix G.

$$KL(p,q) = \sum_x p(x) * log\frac{p(x)}{q(x)} + q(x) * log\frac{q(x)}{p(x)}$$

3 Initial Results

Figure 4 shows 100 tags visualized in two dimensions. Distances are calculated using KL-divergence, and the connectivity scheme is 2 nearest neighbors on top of the minimum spanning tree. Details about how these parameters are chosen are presented in a later section. The size of each tag is proportional to its overall frequency of use and is related to a measure of generality as well. From examining this map of tags, we see many features that make sense (see figure 5). Related topics such as *apple, mac, osx, and ipod* appear close together. Furthermore, we can see regions emerge which correspond to general topics such as web-design, academic topics, and software. The pie chart of eigenvalues shows that this embedding captures 76% of the variance of the data. The connectivity matrix is sorted by general tag popularity and shows the existence of "hubs." These nodes are highly connected, representing tags that are related to many other tags. This feature is consistent with our expectation that a folksonomic system would act like a scale-free network.

4 Specifying the Connectivity Matrix

An important step in the algorithm is specifying the initial connectivity matrix C. As seen in figure 6, picking a level of connectivity that is too low results in a map lacking sufficient information about the relationships between tags, and picking a level of connectivity that is too high results in a map where the significant relationships are not dominant. In order to understand how to find the best level of connectivity for embedding this data, I experimented with a variety of different schemes for growing the connectivity matrix, expanding on the typical k-nearest neighbors approach, and investigated heuristics I could use to quantify the strength of the embedding.

4.1 Methods for Creating the Connectivity Matrix

- **k-nearest neighbors** – The standard method. It is easy to implement and works very well. However, for small values of k, it needs to be supplemented with a minimal spanning tree in order to ensure a connected graph.
- **k-nearest incremental** – This method is like k-nearest; however, it adds the links incrementally, allowing for more detailed plots.
- **Other methods** – I also experimented with other techniques such as simply adding the best links first, or varying how many links each node gets based on its popularity, as well as simply randomly adding links. These methods produced maps which did not appear to have clear advantages over a k-nearest neighbor approach, and they were not investigated in depth due to the difficulties of quantitatively comparing these different schemes.

4.2 Heuristics for Picking the Level of Connectivity

I explore how 3 different metrics can be used to evaluate the strength of an embedding. Figure 2 shows these values for N=50 and a range of k-values using k-nearest neighbors as the connectivity scheme.

- **Eigen-gap** – The gap between the eigenvalue corresponding to the last eigenvector used in the embedding and the eigenvalue corresponding to the first eigenvector to be discarded.
- **Eigen-sum** – The sum of the eigenvalues corresponding to the eigenvectors used in the embedding.
- **Objective function** – $Tr(K)$, the sum of the pairwise distances in K.

4.3 The Eigen-gap

The size of the eigen-gap is a well-known metric for evaluating how well high-dimensional data can be embedded in a lower dimensional space. As we see in figure 2, the eigengap has a maximum at 128 links, which corresponds to a k-value of 3. From a subjective point of view, this value of k does seem to correspond visually to the best embedding. Furthermore it is reassuring that this k-value provides an embedding which captures 78% of the variance of the data, as shown by the eigen-sum.

4.4 Weighting the Objective Function

At first glance, the objective function appears not to provide any help in picking the best k-value. As the number of links increases, the objective function falls asymptotically, approaching a minimum value. If we are trying to find the best embedding, corresponding to reducing the variance of K as much as possible, we can always add another link to get it a little lower, until we are finally specifying a full connectivity matrix.

However, it is wrong to increase k arbitrarily; k is directly related to how much long distances can be trusted, and therefore is related to how much non-linearity the algorithm can sufficiently capture. For $k = 1$, the algorithm trusts only very local distances. As $k \to N$, the algorithm trusts larger and larger distances, until finally the algorithm is reduced to simply linear spectral embedding, disregarding non-linearities.

I experimented with constructing a cost function by weighting the objective function by the percentage of links m used to generate constraints for the SDP problem. The intuition here is that adding each new link should reduce the objective function by $1/m$. As $m \to 1$, each new link should correspond to less of a change to the objective function. By finding the minimum of this cost function, we should be able to find the k-value that corresponds to the proper balance among the value of the objective function, the complexity added, and the reduction of non-linearity that comes from specifying more links.

Figure 3 shows the cost function (subplot 2), and compares it to the eigen-gap (subplot 3). We see that the cost function also identifies the best k-value to be at $k = 3$. Figure 7 offers an explanation of the 3 distinct regions of the cost function for a simpler dataset consisting of numbers arranged on a grid to better illustrate what is going on. We see that in region 1, the algorithm is incrementally building a consistent model; each additional link drastically reduces the objective-function as the nodes congeal into a somewhat stable state. Region 2 is a stable state where the penalty for adding another link is proportional to the objective function. In region 3,

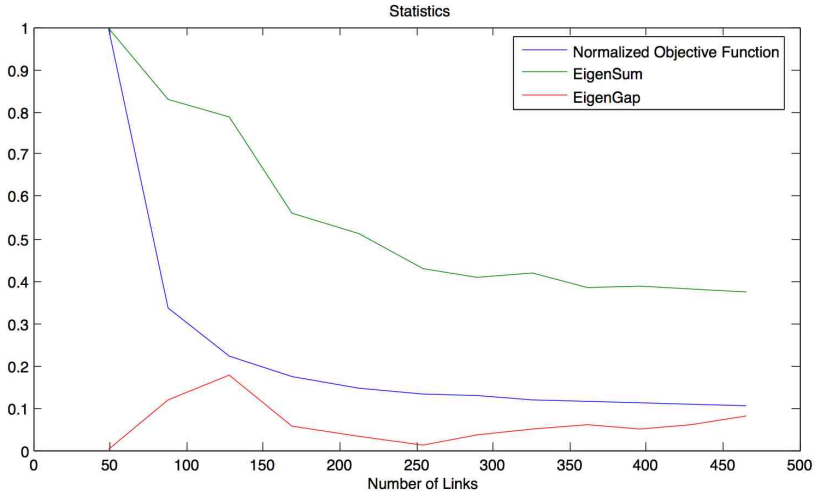

Figure 2: The eigen-gap, eigen-sum, and objective function for k ranging from 0 - 12 (using a minimum spanning tree initialization) for $N = 50$ tags. The level of connectivity is specified along the x-axis in terms of number of links.

adding another link does not affect the objective function as much as it affects the penalty for adding another link.

It is interesting that the minimum of the cost function corresponds to the maximum eigen-gap, since the cost function is evaluated before the spectral embedding and does not incorporate information about the dimension of the final embedding. Also, it is beneficial to avoid doing a time-intensive eigenvalue decomposition when trying to find optimal parameters. Although this cost function is not sufficiently mathematically grounded, it is interesting that it validates existing heuristics for choosing a connectivity level for a variety of datasets. Its validity needs to be explored further and is discussed more in the next section.

5 Future Directions

I am interested in 3 different future directions:

- The cost function $Tr(K) * m$ is presented intuitively as a way to balance the objective function and the number of links used in the connectivity matrix. Its relation to the eigen-gap is promising; however, this function needs a stronger mathematical grounding to be proven an effective heuristic for determining the best connectivity level.
- It is important to be able to compare different connectivity schemes and distance metrics. However, to quantitatively compare the effects of these techniques we need an objective measure of the strength of an embedding that is independent of these parameters.
- The problem of how to best grow a connectivity matrix given a Gram matrix is an interesting problem in its own right. It could be beneficial to experiment with algorithms for producing C that are independent of the embedding step, treating the selection of neighbors as a preprocessing step. For example, one could start with a full connectivity matrix and prune links that are either not necessary or can be proven to distort measurable quantities in the resulting graph.

6 Conclusion

From analyzing the results of using Semidefinite Embedding to visualize folksonomic data, we see that it is indeed possible to reduce the dimensionality of the data, provided we specify an appropriate level of connectivity. The resulting maps of tags preserve important relationships in the data and offer insight

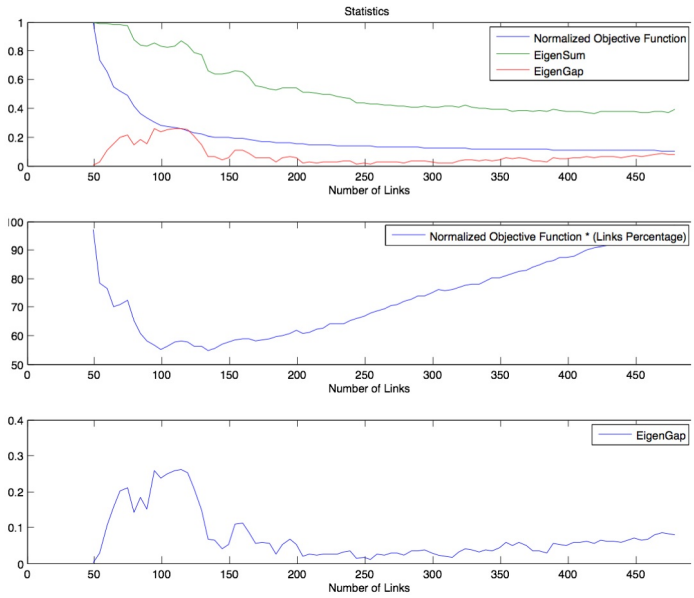

Figure 3: The top plot shows the original statistics from figure 2; however, in this plot the links are added incrementally to provide a finer level of detail. The middle plot shows the cost function $Tr(K) * m$. The bottom plot shows the eigen-gap. It is interesting that the the minimum of the cost function corresponds to the maximum of the eigen-gap.

into how these tags are related. Furthermore, we see the effects of specifying different levels of connectivity and offer two techniques, one well-tested, and one experimental, for choosing the appropriate level.

References

[1] Thomas Vander Wal. Explaining and showing broad and narrow folksonomies.

[2] K. Q. Weinberger, B. D. Packer, and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, Barbados, January 2005.

[3] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR-04)*, volume 2, pages 988–995, Washington D.C., 2004.

[4] K. Q. Weinberger, F. Sha, and L. K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the Twenty First International Conference on Machine Learning (ICML-04)*, pages 839–846, Banff, Canada, 2004.

[5] Wikipedia. Definition: Folksonomy.

[6] Wikipedia. Definition: Semidefinite embedding.

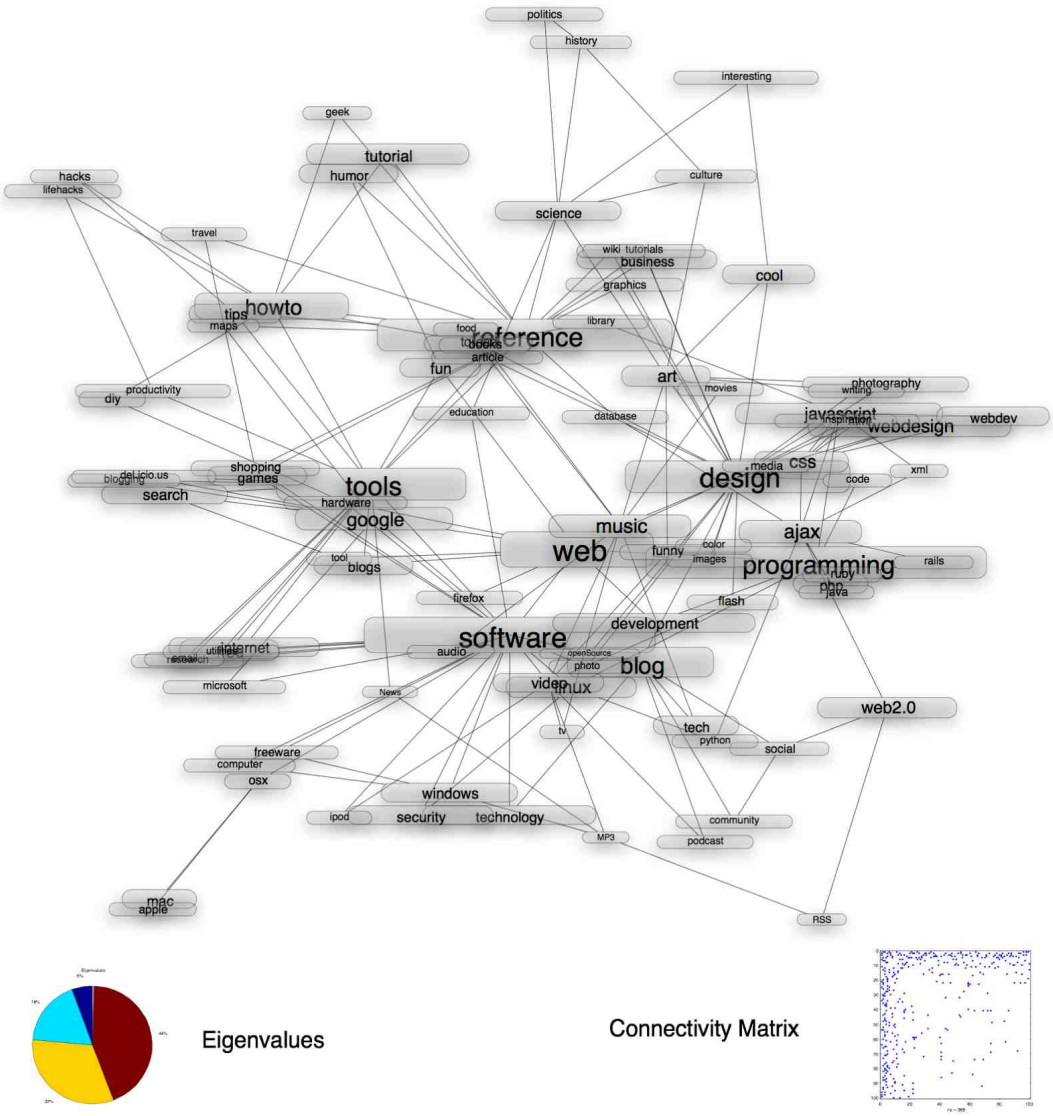

Figure 4: Map of 100 tags. K=2 + min span tree. Using KL-Divergence. Sizes represent popularity, and are also related to a measure of generality.

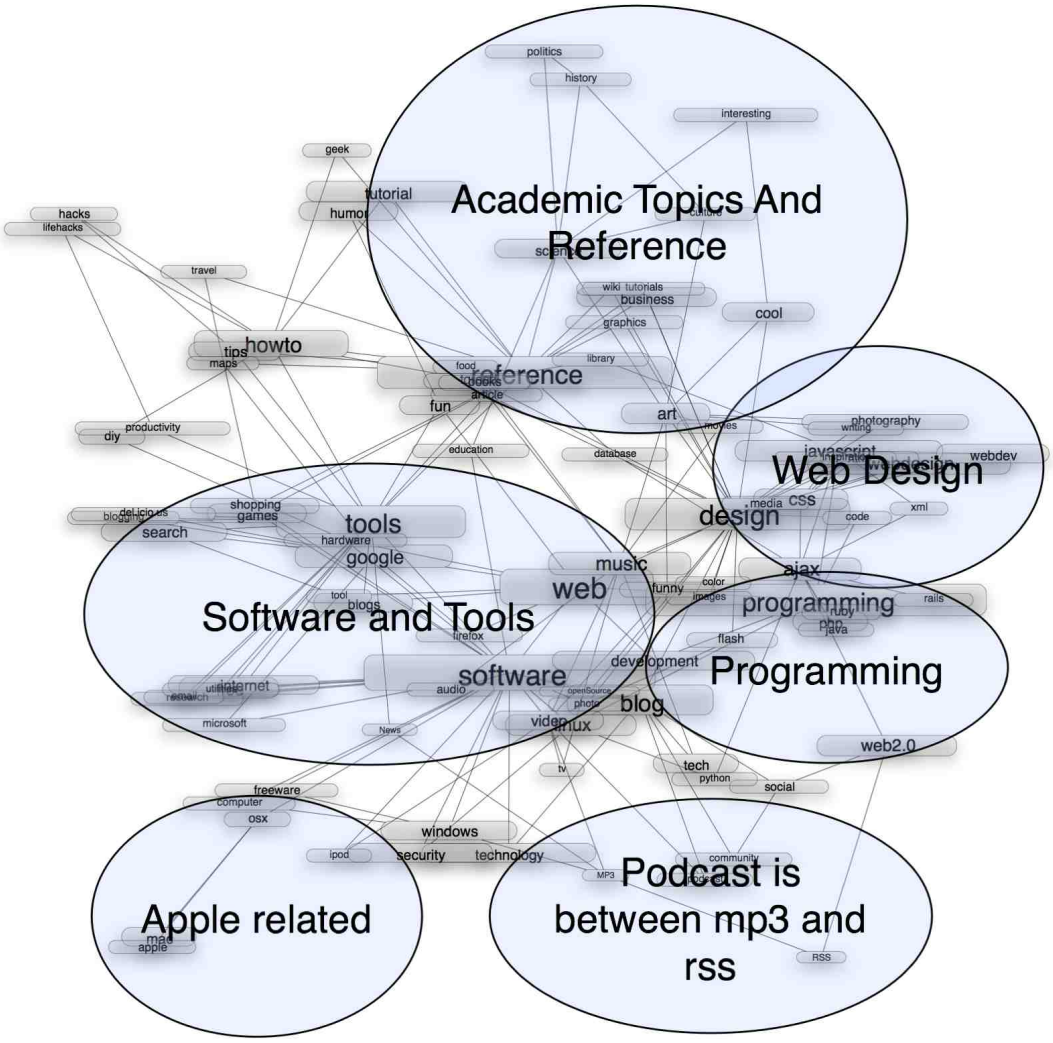

Figure 5: Map of 100 tags. $k = 2$ + min span tree. Features that make sense. General regions emerge and specific items are typically related to their neighbors.

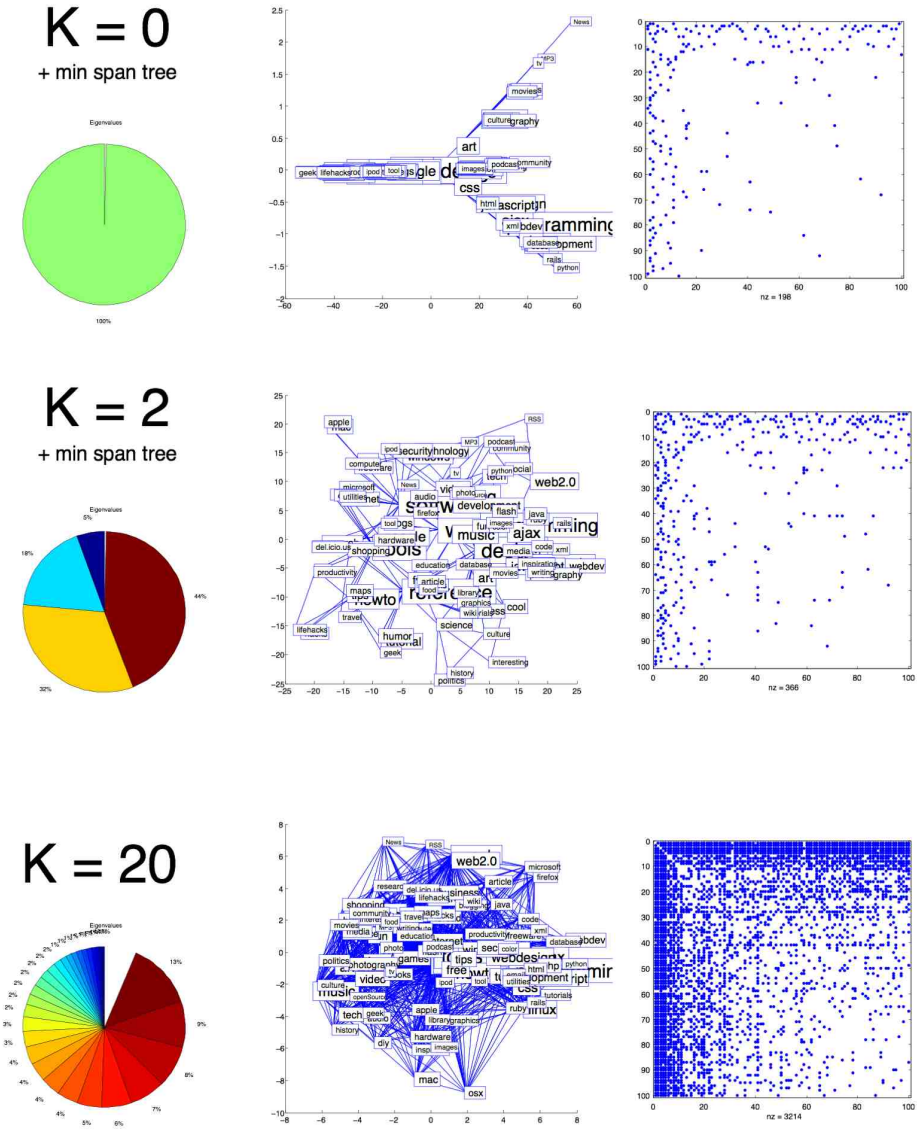

Figure 6: Eigenvalues, maps, and connectivity matrices for a range of k values.

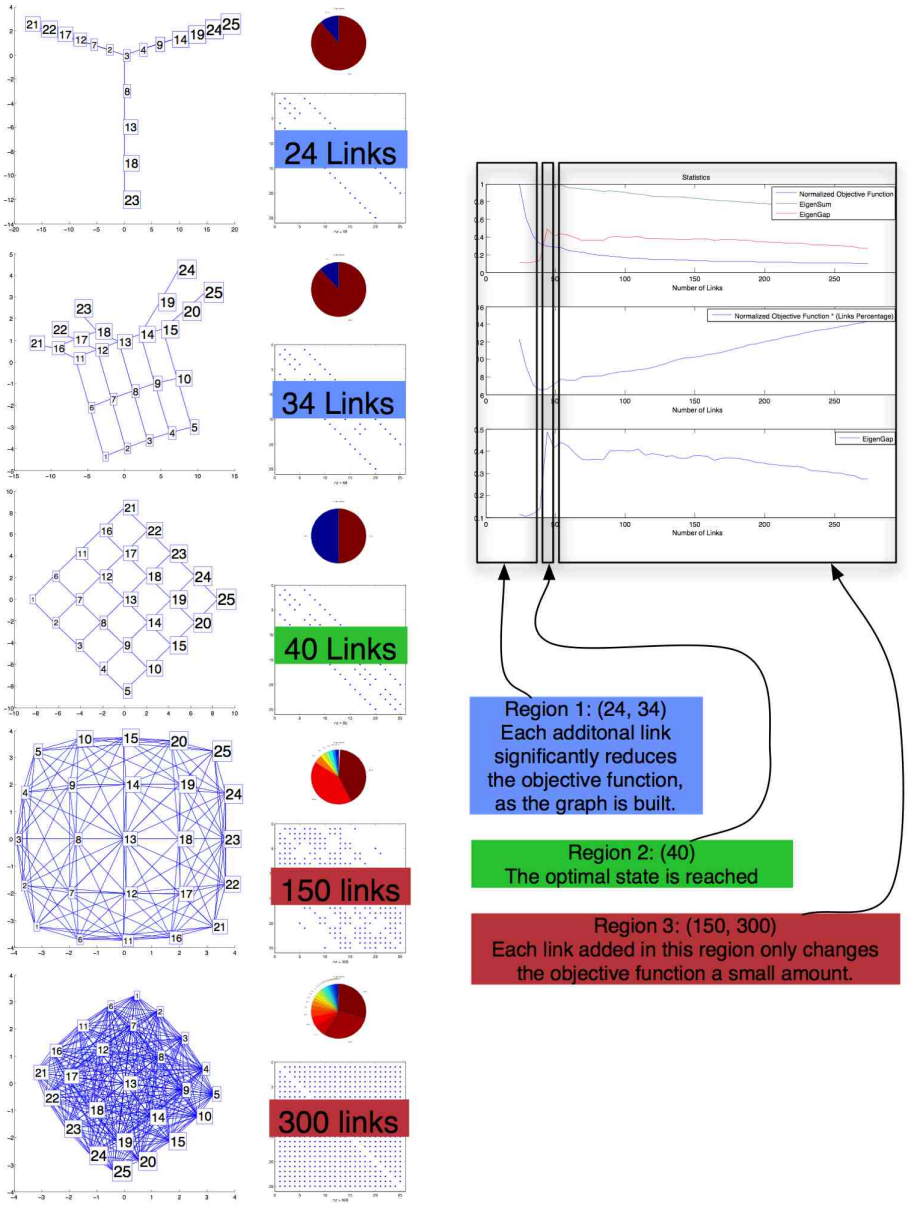

Figure 7: Three distinct regions of the cost function. Region 1 corresponds to a period of assembling the basic structure. Region 2 is the optimal state, corresponding to the minimum of the cost function, and region 3 consists of adding more links which do not significantly change the basic structure.