Unnormalized Spectral Clustering

This form of relaxed RatioCut = **Unnormalized Spectral Clustering**

$$\arg\min_{F \in \mathbb{R}^{N \times k}} \mathrm{Tr}(F^T \mathbf{L} F) \text{ such that } F^T F = \mathbb{I}$$

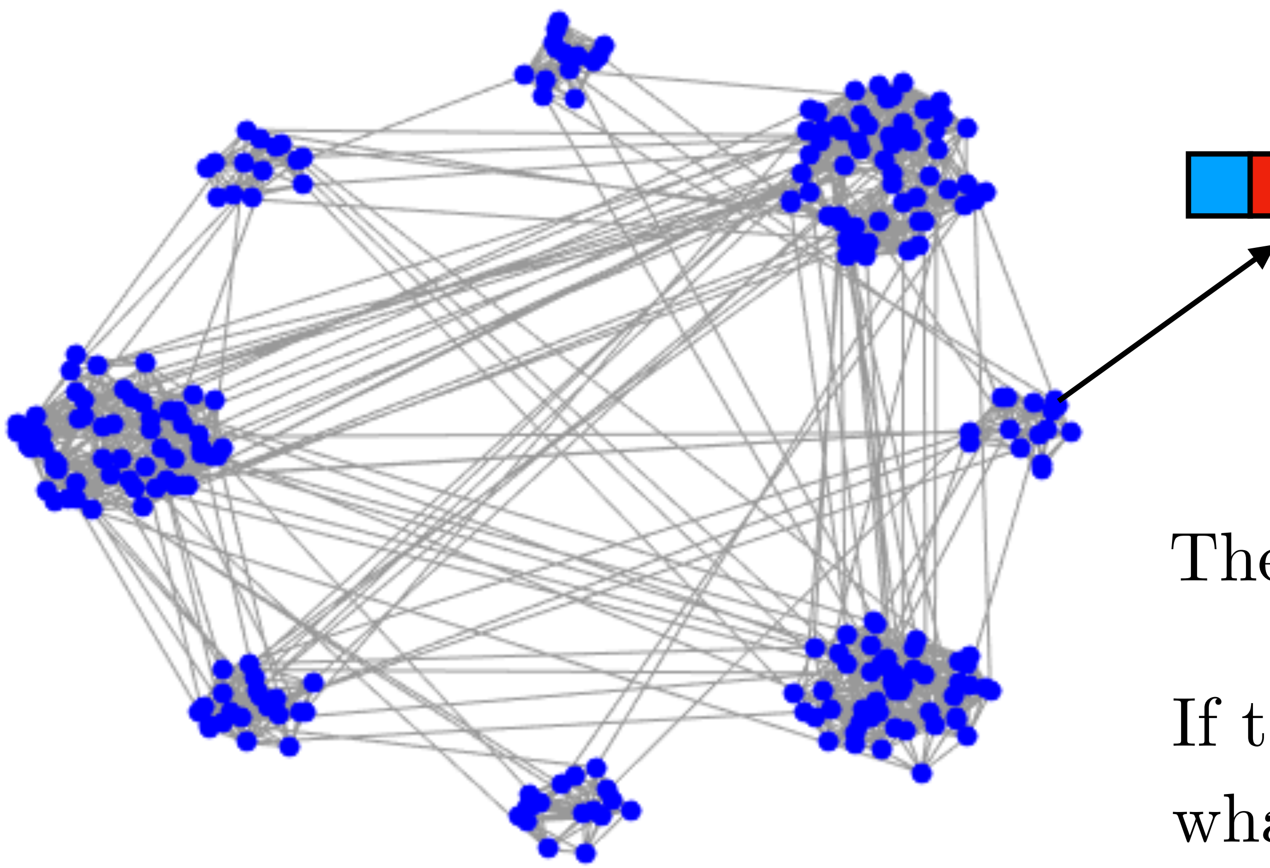
Algorithm: Unnormalized Spectral Clustering

Compute the matrix F of first k eigenvectors of $\mathbf{L}$

Apply k-means to rows of F to obtain cluster assignments

What is the algorithm doing - view 1

At each vertex the algorithm associates a feature vector that represents the fine and large scale structure of that vertex's neighbourhood in the graph



vertex $[i]$, feature $f_i \in \mathbb{R}^k$

$$f_i[\ell] = u_\ell[i], \quad \ell = 1, \ldots, k$$

These are the lines of the matrix F

If the graph has k disconnected components, what does F look like ?

k-means is then applied to these vectors to cluster into k clusters

In short, the algorithm classifies vertices into k clusters blindly

What is the algorithm doing - view 2

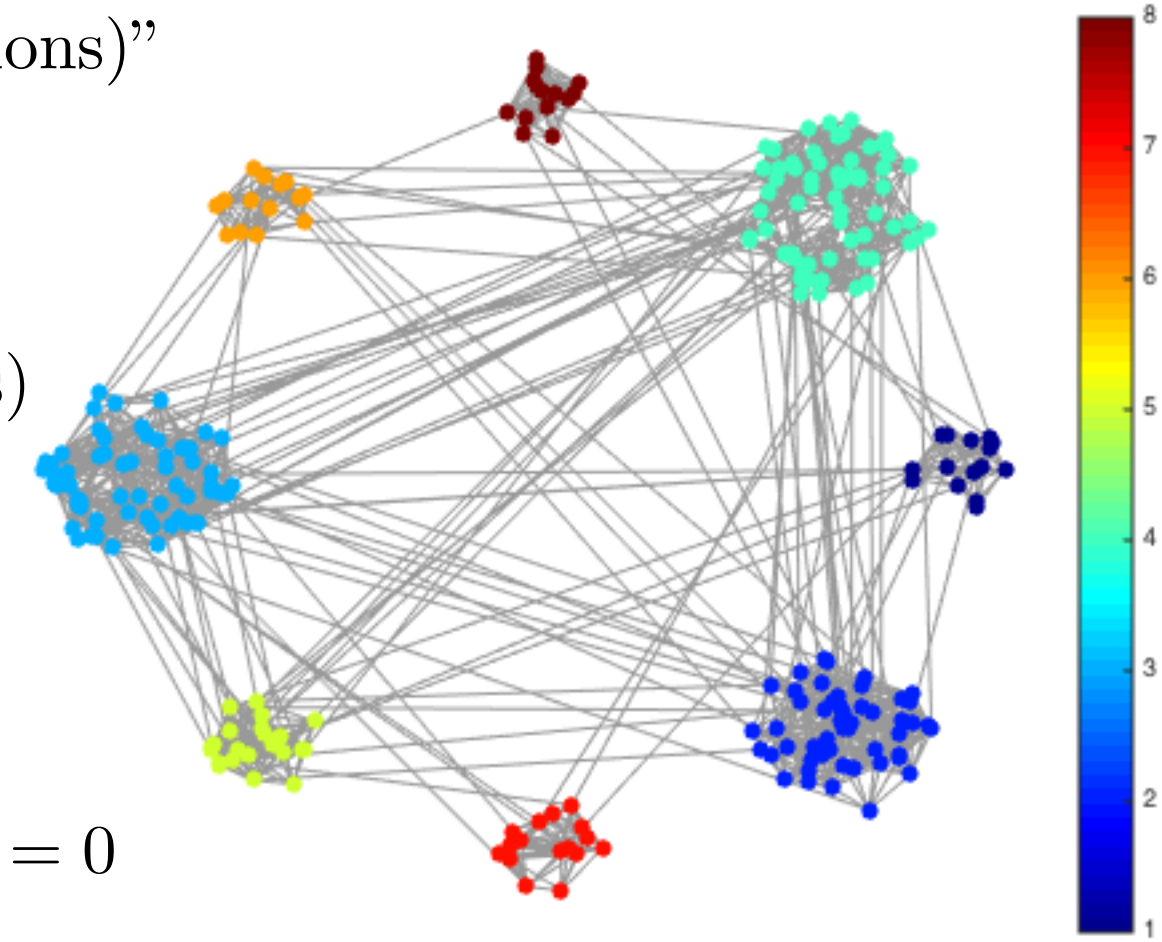
We are looking for k "partition signals (functions)"

$$f_\ell : V \mapsto \mathbb{R}$$

In the ideal case (k disconnected components)

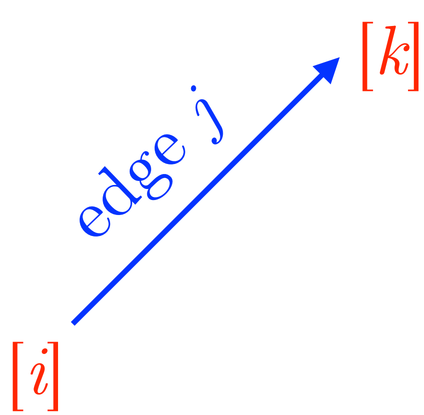
$$f_\ell[i] = \begin{cases} 1 & \text{if } i \in \text{cluster } \ell \\ 0 & \text{otherwise} \end{cases}$$

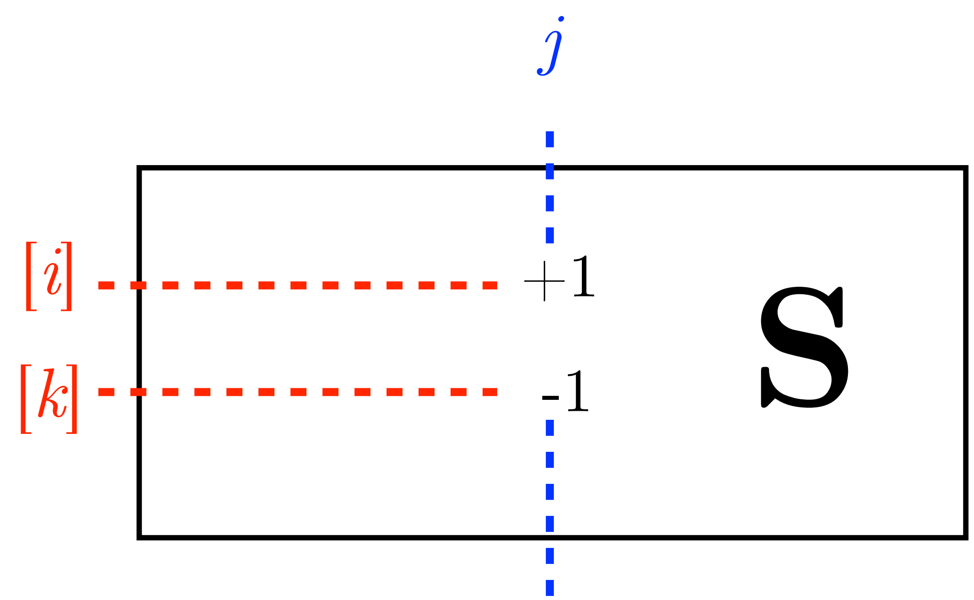
These are maximally **smooth** signals: $f_\ell^T \mathbf{L} f_\ell = 0$

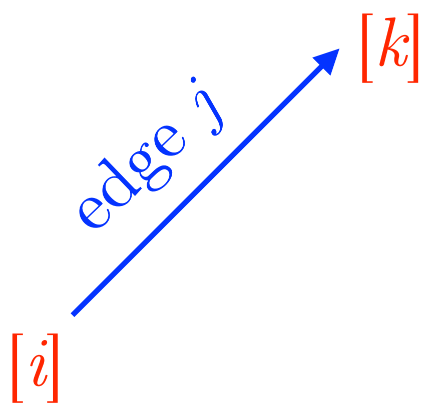


Incidence Matrix: $\mathbf{S} \in \mathbb{R}^{N \times M} \quad N = |V|, \quad M = |E|$

$$\mathbf{S}(i,j) = \begin{cases} +1 & \text{if } e_j = (v_i, v_k) \text{ for some } k \\ -1 & \text{if } e_j = (v_k, v_i) \text{ for some } k \\ 0 & \text{otherwise} \end{cases}$$







Signal (function) f defined on the vertices $f \in \mathbb{R}^N$

$(\mathbf{S}^T f)[j] = f[i] - f[k]$ derivative of f along edge j

$\mathbf{S}^T f \in \mathbb{R}^M$ gradient of f

$$\mathbf{L} = \mathbf{S}\mathbf{S}^T \qquad \begin{aligned} f^T \mathbf{L} f &= f^T \mathbf{S}\mathbf{S}^T f \\ &= \|\mathbf{S}^T f\|_2^2 \\ &= \sum_{i \sim k} (f[i] - f[k])^2 \end{aligned}$$

In general for a weighted graph: $f^T \mathbf{L} f = \sum_{i \sim k} \mathbf{W}(i,k)(f[i] - f[k])^2$

This quadratic (Dirichlet) form is a measure of how smooth the signal is

Laplacian Eigenmaps

Spectral Graph Embedding

Dataset is a large matrix $X \in \mathbb{R}^{N \times L}$

N is the number of data points

L is the dimension of each data points

Often $L >> 1$ and must be reduced (think images)

For computations

For visualisation, in which case we would like $L = 2, 3$

Q: can we reduce L in a way that resulting modified data stays faithful to the original one ?

Formulation

Find a mapping from the N high-dim data points to N low-dim points

$$x_1, \ldots, x_N \mapsto y_1, \ldots, y_N$$

$$x_i \in \mathbb{R}^L \qquad\qquad y_i \in \mathbb{R}^P$$

Assumption: we have a graph of similarities among original data points

Similarities are often constructed by either :

selecting k-nearest neighbours of each point with distance $d(x_i, x_j)$

OR

selecting all points in a neighbourhood $d(x_i, x_j) \leqslant \epsilon$

THEN weighting these edges ex: $\mathbf{W}(i,j) = e^{-d(x_i,x_j)^2/t}$

Formulation

$\mathbf{W}$ captures similarities among data points $x_i \in \mathbb{R}^L$

We want that similar points are embedded close to each other

Suppose we embed in 1 dimension (P=1)

$$\arg \min_{y_1,\ldots,y_N} \sum_{i \sim j} \mathbf{W}(i,j)(y_i - y_j)^2 \longrightarrow \arg \min_{y \in \mathbb{R}^N} y^T \mathbf{L} y$$

Add a constraint to avoid collapse y=0: $y^T \mathbf{D} y = 1$

Avoid trivial eigenvector: $y^T \mathbf{D1} = 0$

$$\longrightarrow \arg \min_{\substack{y \in \mathbb{R}^N \\ y^T \mathbf{D} y = 1 \\ y^T \mathbf{D1} = 0}} y^T \mathbf{L} y$$

Full problem

When we embed in P dimension ($P > 1$)

$$\arg \min_{y_1,\ldots,y_N} \sum_{i\sim j} \mathbf{W}(i,j)\|y_i - y_j\|_2^2$$

Algorithm: Laplacian Eigenmaps

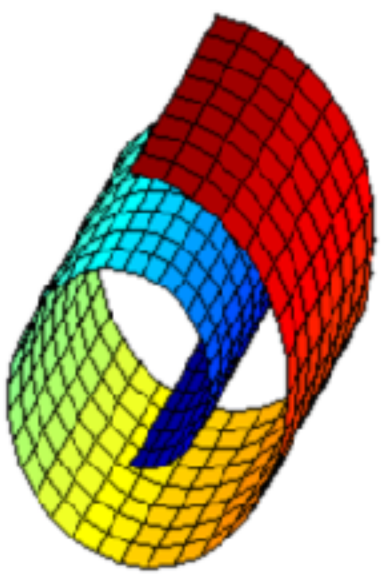
Collect the coordinates of embedded points as lines of matrix Y

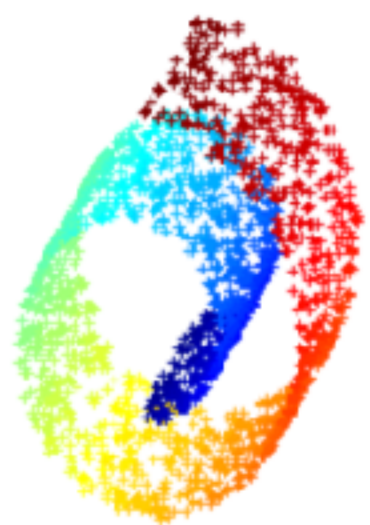
$$\arg \min_{\substack{Y \in \mathbb{R}^{N\times P} \\ Y^T\mathbf{D}Y = \mathbb{I}}} \operatorname{tr}(Y^T\mathbf{L}Y)$$

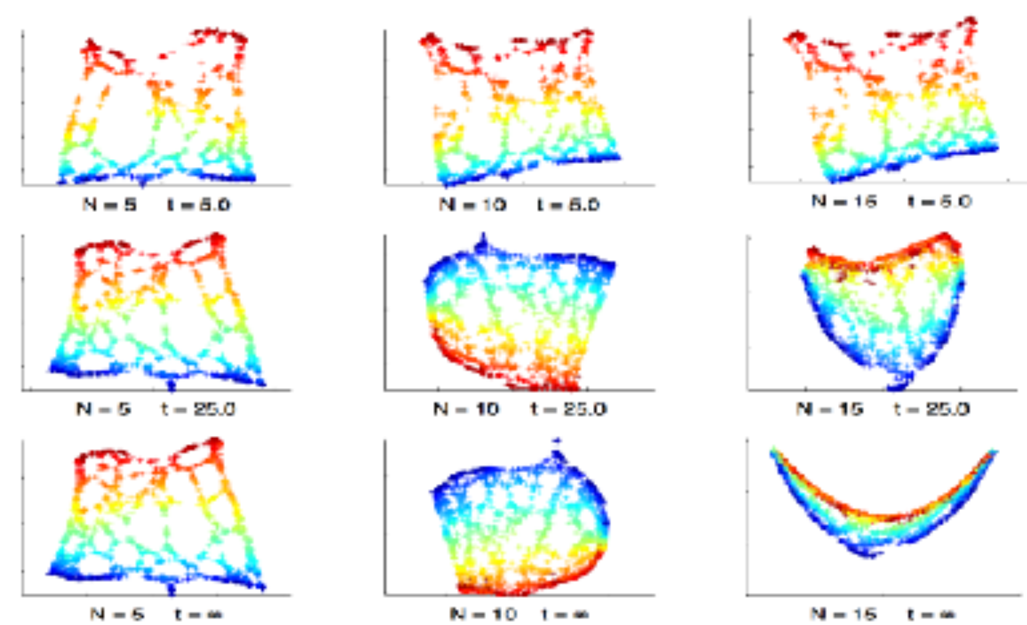
Laplacian Eigenmaps produces coordinate maps that are smooth functions over the original graph. Note similarity with clustering !

Examples: synthetic

M. Belkin and P. Niyogi, “Laplacian eigenmaps for dimensionality reduction and data representation,” *Neural Comput*, vol. 15, no. 6, pp. 1373–1396, 2003.

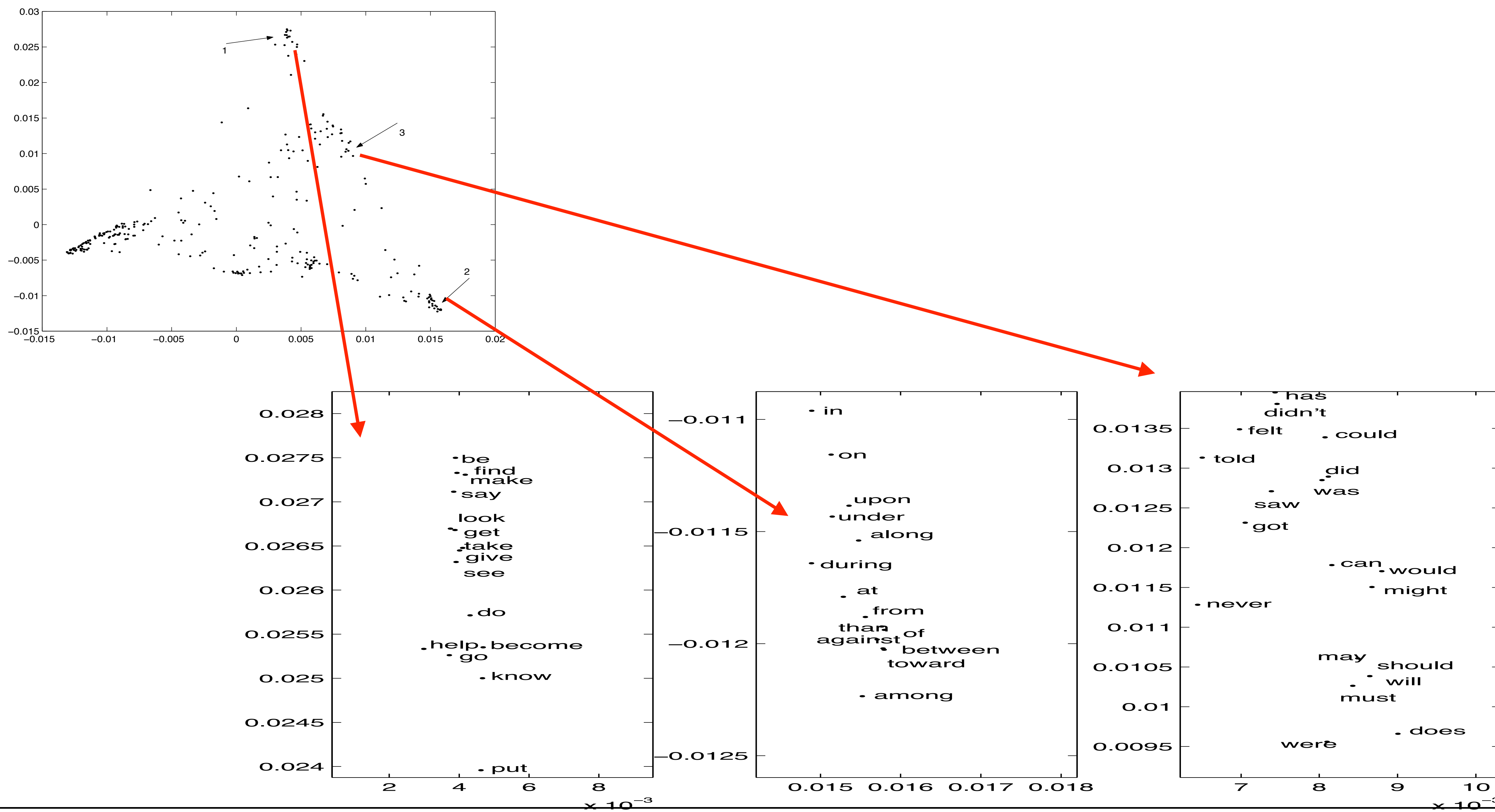






Examples: text

M. Belkin and P. Niyogi, “Laplacian eigenmaps for dimensionality reduction and data representation,” *Neural Comput*, vol. 15, no. 6, pp. 1373–1396, 2003.



Examples: speech