# Laplacian Eigenmaps for Dimensionality Reduction and Data Representation

Neural Computation, June 2003; 15 (6):1373-1396
Presentation for CSE291 sp07

M. Belkin[1] P. Niyogi[2]

[1]University of Chicago, Department of Mathematics
[2]University of Chicago, Departments of Computer Science and Statistics

5/10/07

# Outline

# Outline

# The problem

In AI, information retrieval and data mining we get
Intrinsically Low $m$-dimensional data lying in high $N$-dimensional space.
Recovering the initial dimensionality results

- Smaller data sets
  - Faster computations
  - Reduce the space needed
- More meaningful representations
- . . . much more

# Non Linearity

PCA, MDS and their variations create linear embeddings of the data
For non-linear structures their linear embeddings fail.
IsoMAP, LLE and Laplacian eigenmaps partially solve this problem.
The heart of these approaches is some kind of non-linear method

- IsoMAP, Adjacency Graph of "Neighbor nodes"-edges
- LLE, Each node is expressed by a limited number of neighbor nodes
- In Laplacian Eigenmaps, Adjacency Graphs of "Neighbor nodes"-edges

# Outline

# Step 1. Construct the Graph

We construct the Adjacency Graph $A$ putting $(i, j)$ edges if $x_i, x_j$ are "close"
Close may mean:

- $\epsilon$-neighborhoods $||x_i - x_j||^2 < \epsilon$
- $n$ nearest neighbors
- Combination of the above (at most $n$ nearest neighbors with $||x_i - x_j||^2 < \epsilon$)

# Step 2. Choose the Weights

The edges have weights that can be

- Simple-minded: 1 if connected 0 else
- Heat Kernel: $W_{ij} = e^{-\frac{||x_i - x_j||^2}{t}}$

We note the with $t = \infty$ we get the simple-minded approach
The second method includes more information in our model

# Step 3. The Laplacian eigenvalues

Let $A$ be the Adjacency matrix and $D$ the degree matrix of the graph We introduce the Laplacian matrix $L = A - D$ and we solve the generalized eigenvector problem

$$Lf = \lambda Df$$

We sort the eigenvectors according to their eigenvalues $0 = \lambda + 0 \leq \lambda_1 \leq \lambda_2 \ldots \lambda_{k-1}$ We leave out $f_0$ and use the first $m$ eigenvectors for embedding in $m$-dimensional Euclidian space.

$$x_i \rightarrow (f_1(i), \ldots, f_m(i))$$

# A quick overview

- There are several efficient approximate techniques for finding the nearest neighbors.
- The algorithm consists of a few local computations and one sparse eigenvalue problem.
- It is a member of the IsoMap-LLE family of algorithms.
- As LLE and IsoMap does not give good results for non-convex manifolds.

# Outline

# Our minimization problem for 1 Dimension

Let $y = (y_1, y_2, \ldots, y_n)^T$ be our mapping.
We would like to minimize

$$\sum_{ij} (y_i - y_j)^2 W_{ij}$$

under appropriate constraints

$\sum_{ij}(y_i - y_j)^2 W_{ij} = \sum_{ij}(y_i^2 + y_j^2 - 2y_i y_j) W_{ij} =$
$\sum_i y_i^2 D_{ii} + \sum_j y_j^2 D_{jj} - 2\sum_{ij} y_i y_j W_{ij} = 2y^T L y$
So we need to find

$$argmin_y y^T L y$$

# Our Constraints

The constraint

$$y^T Dy = 1$$

removes an arbitrary scaling factor in the embedding
$argmin_y y^T Ly, y^T Dy = 1$ is exactly the generalized eigenvalue problem $Ly = \lambda Dy$
The additional constraint $y^T D\mathbf{1} = 0$ the solution $\mathbf{1}$ with $\lambda_0 = 0$
So the final problem is:

$$argmin_y y^t Ly,\ y^T Dy = 1,\ y^T D\mathbf{1} = 0$$

The general problem for $m$ dimensions is similar and gives exactly the first $m$ smallest eigenvalues.

# Sum Up

So each eigenvector is a function from nodes to $\mathbb{R}$ in a way that "close by" points are assigned "close by" values.

The eigenvalue of each eigenfunction gives a measure of how "close by" are the values of close by points

By using the first $m$ eigenfunctions for determining our $m$-dimensions we have our solution.

# Outline

# Continuous Manifolds

When we are using continuous spaces the Graph Laplacian becomes the Laplace Beltrami Operator (More on this by Nakul.)

And our optimization problem is finding functions $f$ that map the manifold points to $\mathbb{R}$, in a way that $\int_{\mathcal{M}} || \bigtriangledown f(x)||^2$ is minimum. Intuitively minimizing the gradient minimizes the values assigned to close by points.

# Heat Kernels

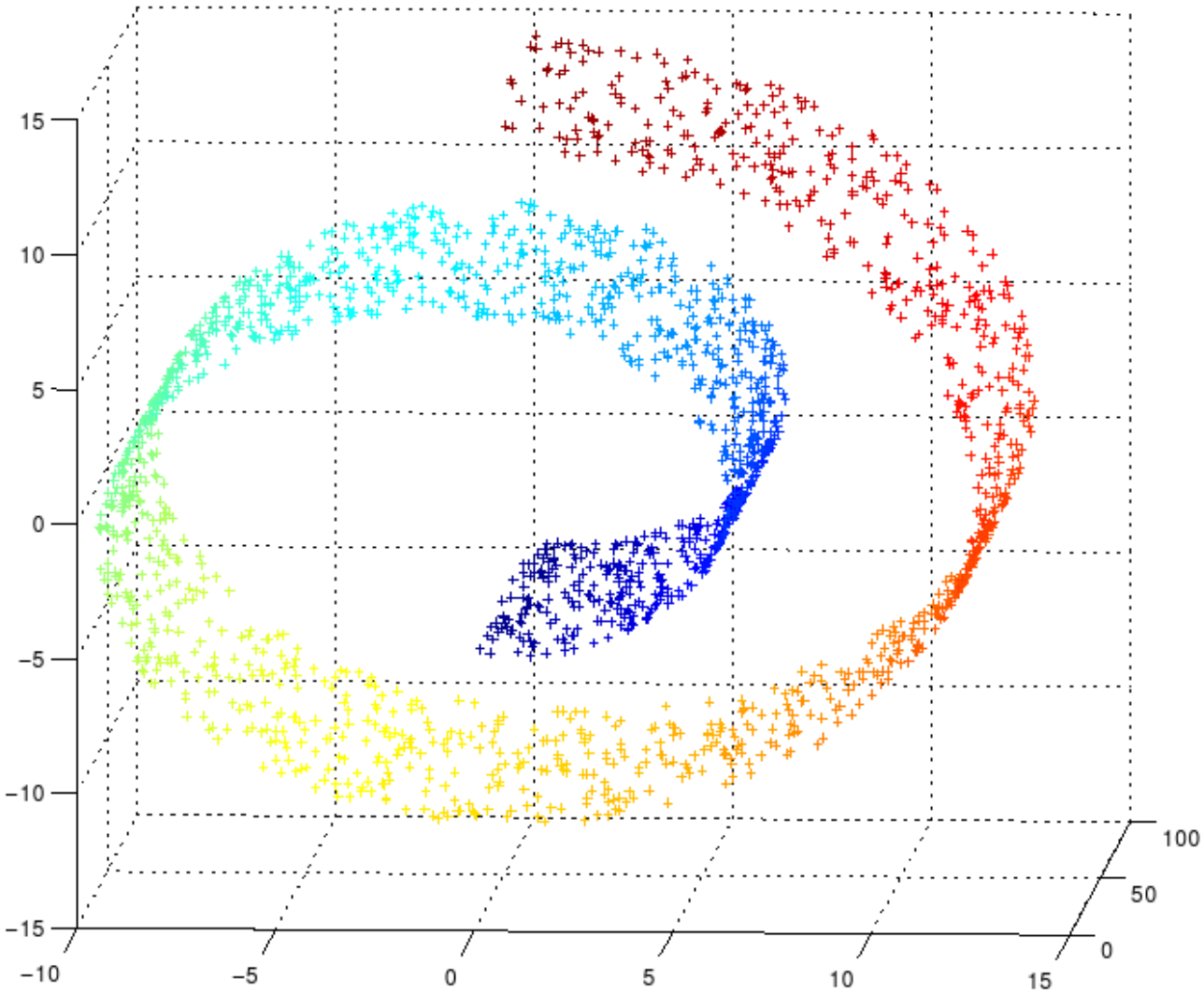
Continuous manifolds give the idea for the Heat Kernel

That is assigning weights $W_{ij} = e^{\frac{||x_i - x_j||^2}{t}}$

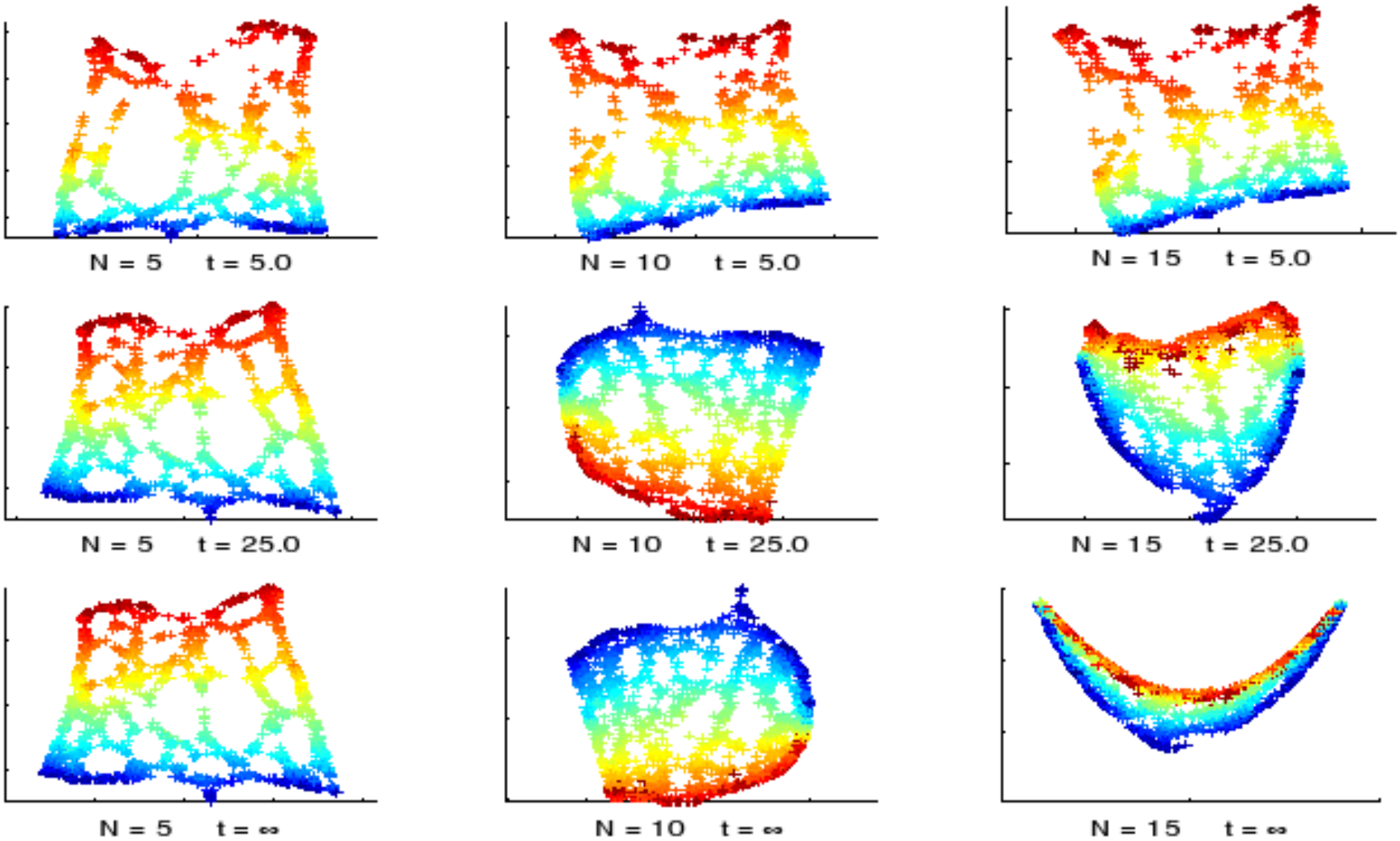
The heat function is a solution for the Laplace Beltrami Operator, and as a result by assigning the weights according to it we get better approximation of the "ideal" infinite points case.

# Outline

# One more Swiss roll!



# Unfolding the Swiss roll - the parameters t,N




- N=number of nearest neighbors, t the heat kernel parameter

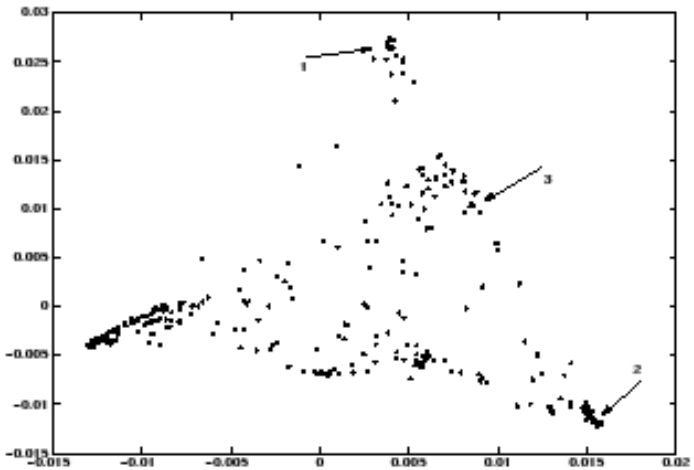
# Outline

# Understanding syntactic structure of words

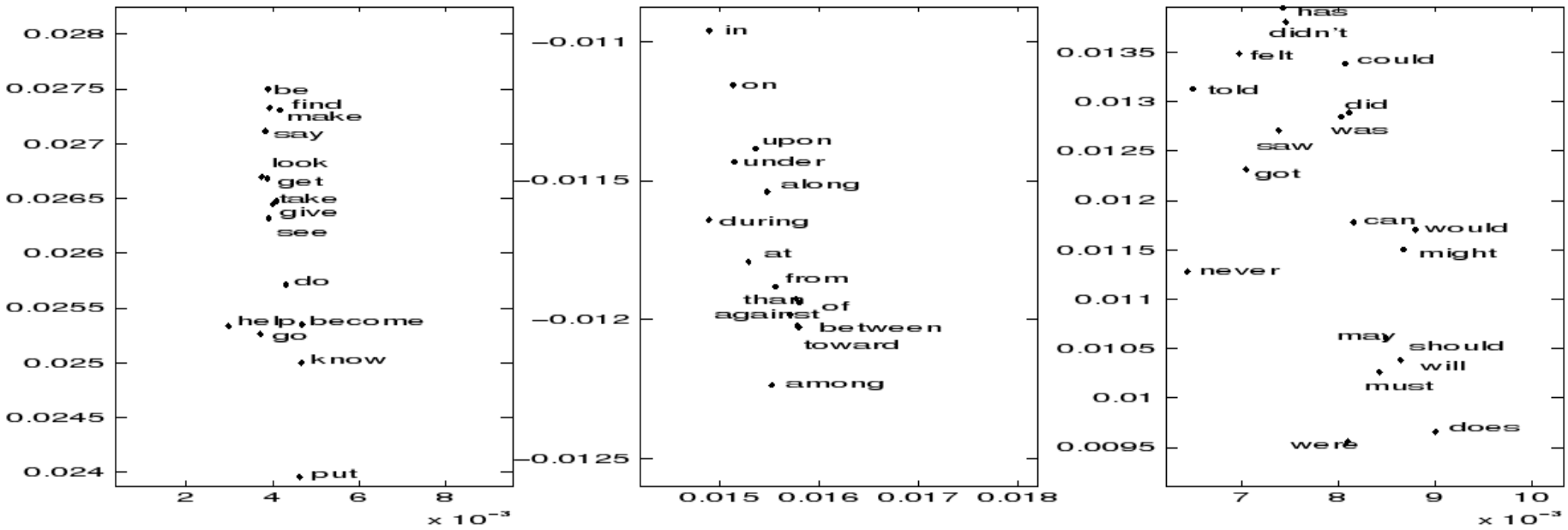
Input:

- 300 most frequent words of Brown corpus
- Information about the frequency of its left and right neighbors (600 Dimensional space.)

The algorithm run with $N = 14, t = \infty$



# Understanding syntactic structure of Words




- Three parts of the output. We can see verbs, prepositions, and auxiliary and modal verbs are grouped.

# Discussion

- Laplacian Eigenvalues is a part of the Isomap LLE family
- They actually prove that LLE minimizes approximately the same value
- Many times we need to skip eigenvectors, as eigenvectors with bigger eigenvalues give better results. However variations for doing so exist.
- Although all these algorithms give good results for convex non-linear structures, they do not give much better results in other cases.