Nonlinear Dimensionality Reduction

Chapter 5.3 + Appendixes D and E

Antti Sorjamaa and Yoan Miche

Time Series Prediction and ChemoInformatics Group
Adaptive Informatics Research Centre
Helsinki University of Technology

Outline

- Vector Quantization
- Graph Building

Chapter 5.3

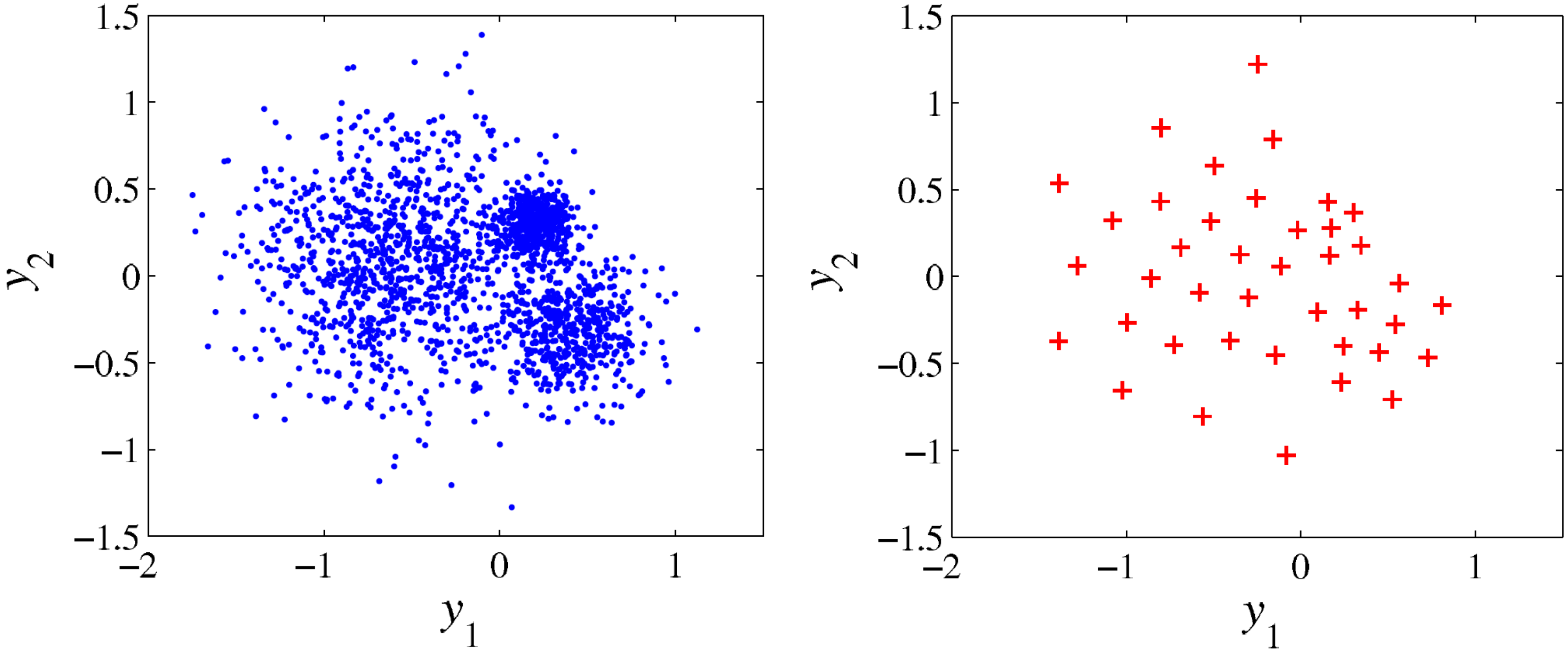
- Locally Linear Embedding (LLE)
- Laplacian Eigenmaps (LE)
- Isotop

Vector Quantization

- Aim is to decrease the amount of data
- Many methods use VQ as preprocessing method
- Used in many fields
  - Data analysis and compression
  - Clustering and classification
  - Telecommunications

Vector Quantization

- Aim is to decrease the amount of data



Vector Quantization (2)

- Quantization distortion

$$E_{\mathrm{VQ}} = \frac{1}{N} \sum_{i=1}^{N} \left\| \mathbf{y}(i) - \mathrm{dec}\left(\mathrm{cod}\left(\mathbf{y}(i)\right)\right) \right\|^2$$

  – <u>cod</u>ing denotes a function for finding the Best Matching Unit (BMU) of the data point
  – <u>dec</u>oding denotes a function for replacing the data point with the BMU

Vector Quantization (3)

- *K*-means

$$\mathbf{c}(j) \leftarrow \frac{1}{\left|V_j\right|} \sum_{\mathbf{y}(i) \in V_j} \mathbf{y}(i)$$

- Stochastic Gradient Descent

$$\text{¢}\ \ j) \leftarrow \text{¢}\ \ j) - \alpha\left(\text{¢}\ \ j) - \mathbf{x}_i\right)$$

Vector Quantization (4)

- Static → fixed number of prototypes
  - Classical techniques
    - *K*-means, LBG
  - Competitive Learning
    - Winner takes all (Stochastic Gradient Descent)
    - Winner takes most (SOM and Neural Gas)
- Incremental → increasing
- Dynamic → increasing and decreasing

Graph Building

Yoan presents...

Isotop

Close to SOM algorithm except

- Vector Quantization optional
- No predefined latent space
- Data points are not used in the learning phase, but instead a graph of them
- No online version
- More ”data-driven” methodology

Isotop Algorithm

- Optional Vector Quantization
- Build graph with pairwise distances
- Initialize all nodes to zero
- Calculate parameters with respect to q
- For each node
    I. Generate random point around the node
    II. Select closest node
    III. Update coordinates of all nodes according to neighborhood
- Increase q and goto step 4 unless converged

Isotop Algorithm (2)

1) Initialize all nodes to zero
   – Place Gaussian kernel of unit variance on each node, N(**x**(i),**I**)
- Calculate parameters with respect to *q*
   – Learning rate α
   – Neighborhood width λ

For each node from 1 to *N*

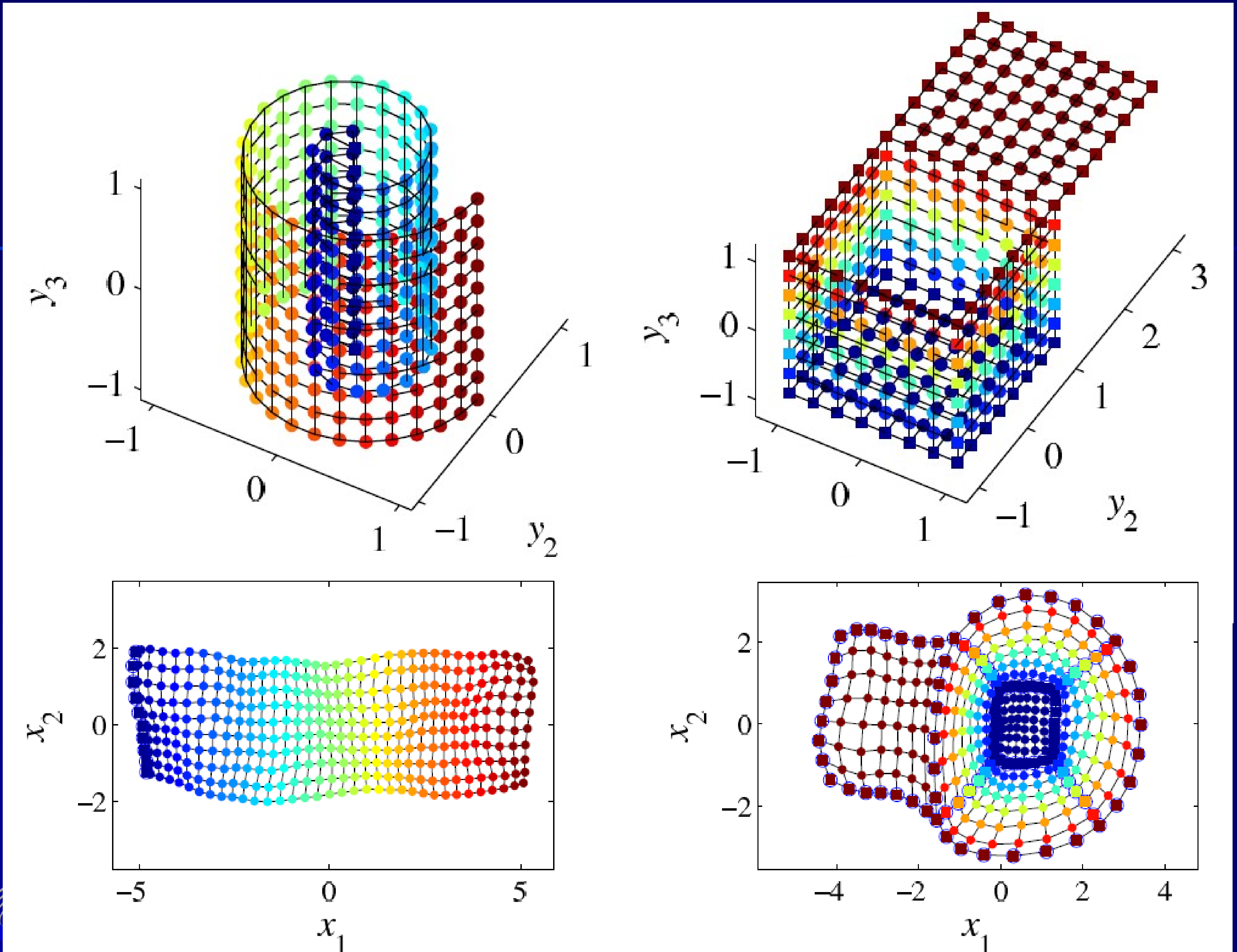
- Generate random point **r** from the distribution of the node N(**x**(i),**I**) → **r**
- Calculate the nearest node
- Update all nodes

$$\mathbf{x}(i) \leftarrow \mathbf{x}(i) - \alpha v_{\lambda}(i,j)\left(\mathbf{r} - \mathbf{x}(i)\right)$$

$$v_{\lambda}(i,j) = \exp\left(-\frac{1}{2}\frac{\delta_{\mathbf{y}}^{2}(i,j)}{\lambda^{2}\mu_{(v_h,v_j)\in E}^{2}\left(\delta_{\mathbf{y}}(h,j)\right)}\right)$$

Notes of Convergence

- ”Nodes won’t collapse”
  – Gaussian centers make boundary nodes expand the graph
  – Neighborhood size must be kept reasonable
- ”Nodes won’t disperse infinitely”
  – Update rule generates an attractive force
  – Neighborhood size larger than zero



y_3
y_2
x_2
x_1

Questions?

Antti.Sorjamaa@hut.fi

http://www.cis.hut.fi/projects/tsp