Dimensionality Reduction

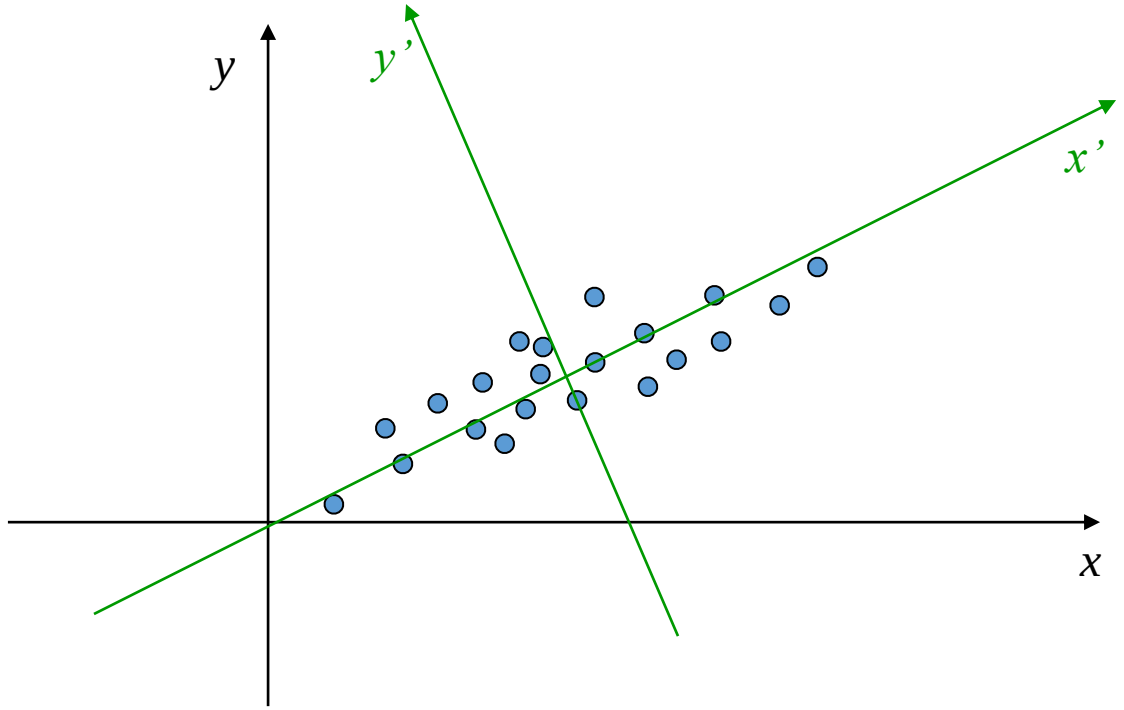
Yu-Shuen Wang, CS, NCTU

Principal component analysis

The slides are from Prof. Olga Sorkine, ETH Zurich.

PCA – the general idea

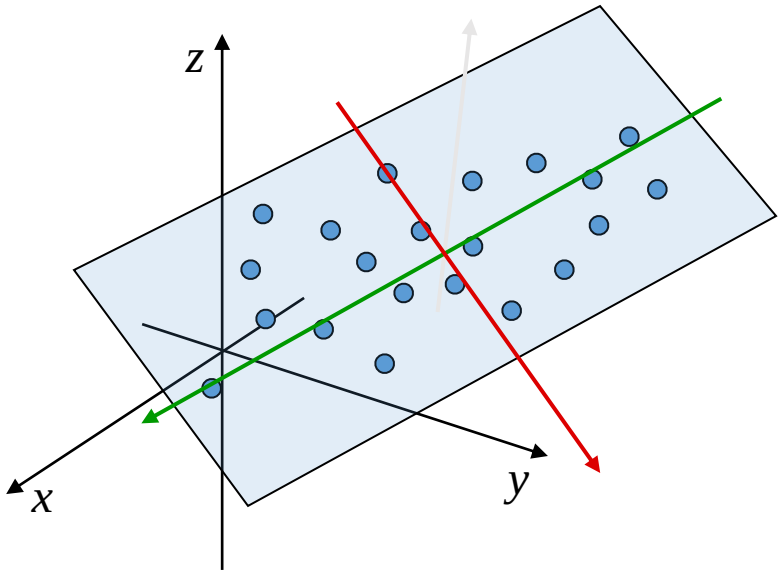
- PCA finds an orthogonal basis that best represents given data set.



- The sum of distances^2 from the x' axis is minimized.

PCA – the general idea

- PCA finds an orthogonal basis that best represents given data set.

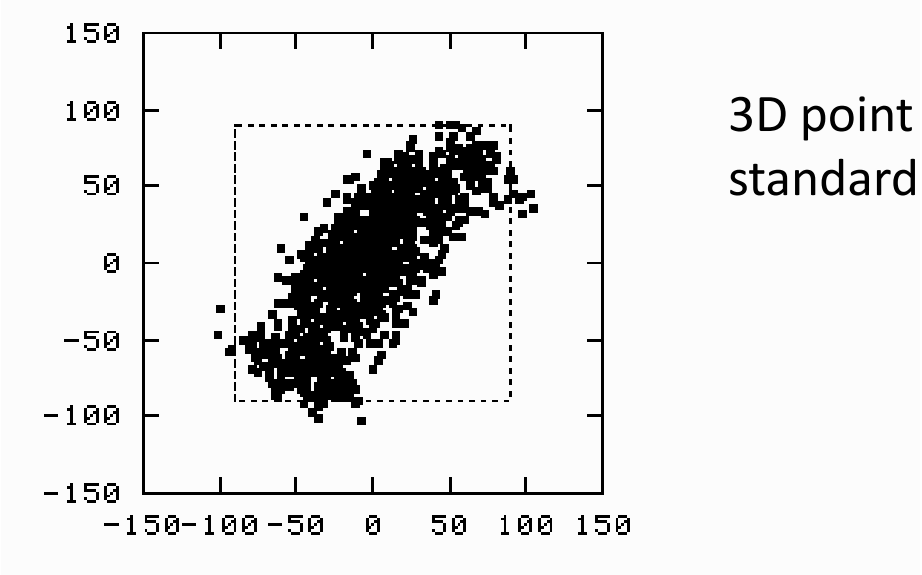


3D point set in standard basis

- PCA finds a best approximating plane (again, in terms of $\Sigma distances^2$)

PCA – the general idea

- PCA finds an orthogonal basis that best represents given data set.

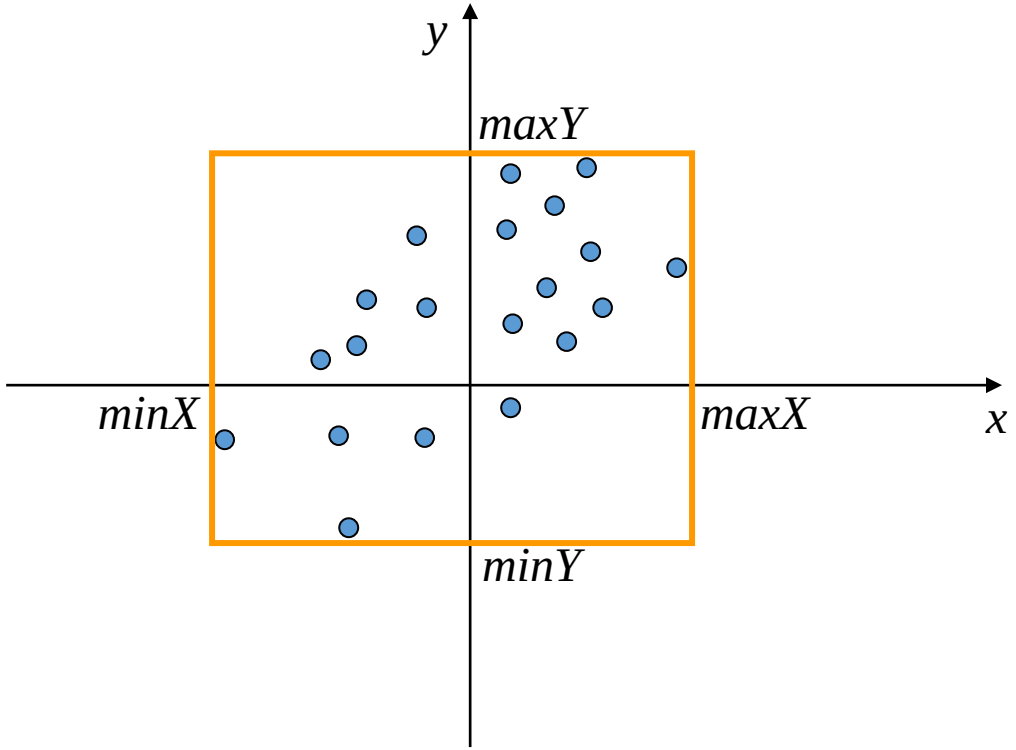


3D point set in standard basis

- PCA finds a best approximating plane (again, in terms of $\Sigma distances^2$)

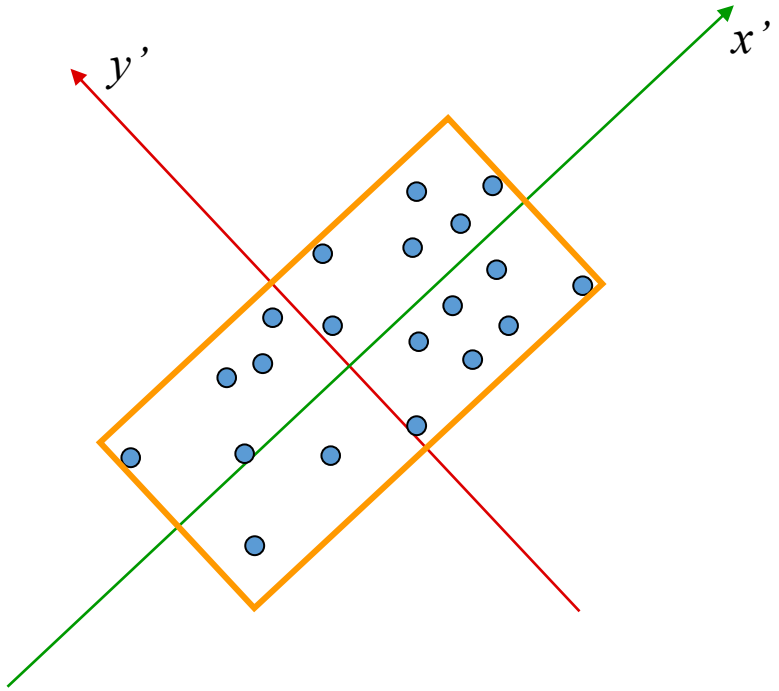
Application: finding tight bounding box

- An axis-aligned bounding box: agrees with the axes



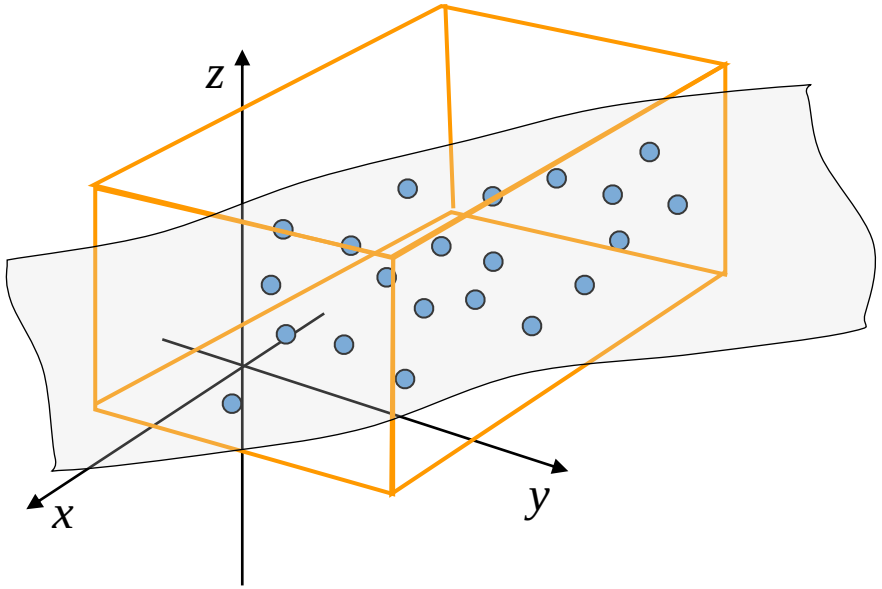
Application: finding tight bounding box

- Oriented bounding box: we find better axes!



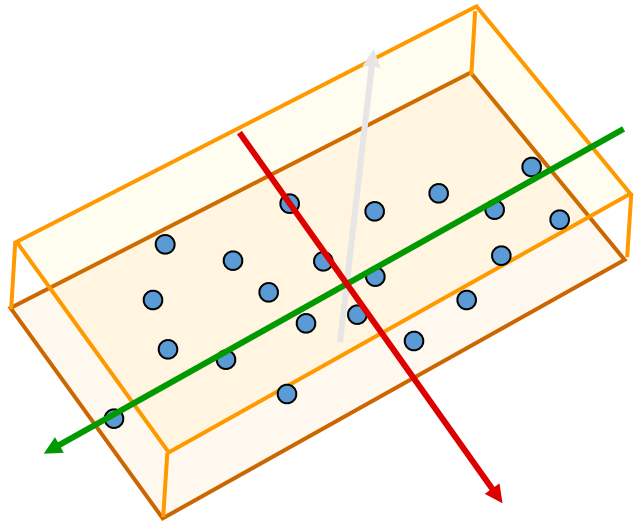
Application: finding tight bounding box

- This is not the optimal bounding box



Application: finding tight bounding box

- Oriented bounding box: we find better axes!



Notations

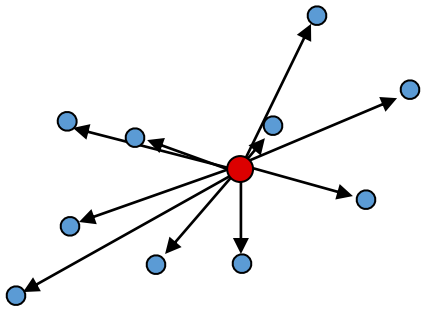
- Denote our data points by $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in R^d$

$$\mathbf{x_1} = \begin{pmatrix} x_1^1 \\ x_1^2 \\ \vdots \\ x_1^d \end{pmatrix}, \ \mathbf{x_2} = \begin{pmatrix} x_2^1 \\ x_2^2 \\ \vdots \\ x_2^d \end{pmatrix}, \ \ldots, \ \mathbf{x_n} = \begin{pmatrix} x_n^1 \\ x_n^2 \\ \vdots \\ x_n^d \end{pmatrix}$$

The origin of the new axes

- The origin is zero-order approximation of our data set (a point)
- It will be the center of mass:

$$\mathbf{m} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$$



- It can be shown that:

$$\mathbf{m} = \underset{\mathbf{x}}{\operatorname{argmin}} \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{x}\|^2$$

Scatter matrix

- Denote $\mathbf{y}_i = \mathbf{x}_i - \mathbf{m}, \quad i = 1, 2, \ldots, n$

$$\boxed{S = YY^T}$$

where Y is $d \times n$ matrix with $\mathbf{y}_k$ as columns ($k = 1, 2, \ldots, n$)

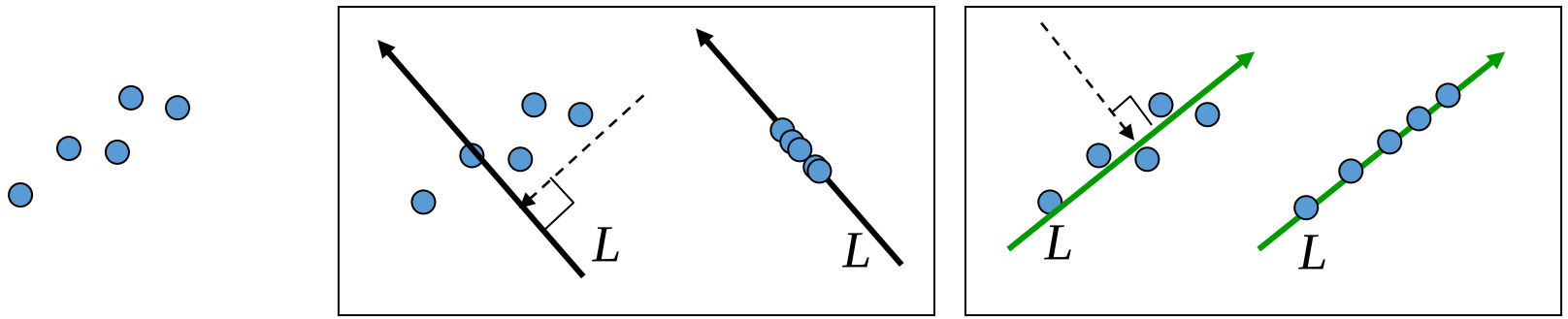
$$S = \begin{pmatrix} y_1^1 & y_2^1 & \cdots & y_n^1 \\ y_1^2 & y_2^2 & & y_n^2 \\ \vdots & \vdots & & \vdots \\ y_1^d & y_2^d & \cdots & y_n^d \end{pmatrix} \begin{pmatrix} y_1^1 & y_1^2 & \cdots & y_1^d \\ y_2^1 & y_2^2 & \cdots & y_2^d \\ \vdots & & & \vdots \\ y_n^1 & y_n^2 & \cdots & y_n^d \end{pmatrix}$$

$$\mathrm{Y} \qquad\qquad \mathrm{Y}^T$$

Variance of projected points

- In a way, S measures variance (= scatterness) of the data in different directions.
- Let's look at a line L through the center of mass $\mathbf{m}$, and project our points $\mathbf{x}_i$ onto it. The variance of the projected points $\mathbf{x'}_i$ is:

$$\text{var}(L) = \frac{1}{n}\sum_{i=1}^{n} \| \mathbf{x}'_i - \mathbf{m} \|^2$$

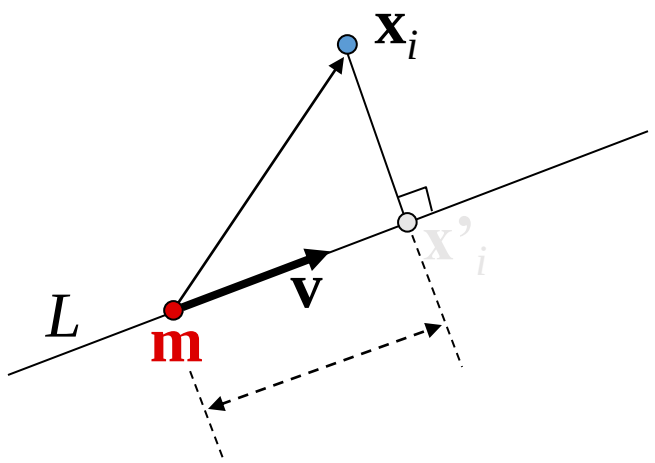


Original set | Small variance | Large variance

Variance of projected points

- Given a direction $\mathbf{v}$, $\|\mathbf{v}\| = 1$, the projection of $\mathbf{x}_i$ onto $L = \mathbf{m} + \mathbf{v}t$ is:

$$\boxed{\| \mathbf{x}_i' - \mathbf{m} \| = < \mathbf{v},\ \mathbf{x}_i - \mathbf{m} > / \| \mathbf{v} \| = < \mathbf{v},\ \mathbf{y}_i > = \mathbf{v}^T \mathbf{y}_i}$$



Variance of projected points

- So,

$$\text{var}(L) = \frac{1}{n}\sum_{i=1}^{n} \| \mathbf{x}_i' - \mathbf{m} \|^2 = \frac{1}{n}\sum_{i=1}^{n} (\mathbf{v}^{\mathbf{T}}\mathbf{y}_i)^2 = \frac{1}{n} \| \mathbf{v}^{\mathbf{T}} Y \|^2 =$$

$$= \frac{1}{n} \| Y^T \mathbf{v} \|^2 = \frac{1}{n} < Y^T \mathbf{v},\ Y^{\mathbf{T}}\mathbf{v} > = \frac{1}{n} \mathbf{v}^{\mathbf{T}} Y Y^T \mathbf{v} = \frac{1}{n} \mathbf{v}^{\mathbf{T}} S \mathbf{v} = \frac{1}{n} < S\mathbf{v},\ \mathbf{v} >$$

$$\sum_{i=1}^{n} (\mathbf{v}^T \mathbf{y_i})^2 = \sum_{i=1}^{n} \left(\begin{pmatrix} v^1 & v^2 & \cdots & v^d \end{pmatrix} \begin{pmatrix} y_i^1 \\ y_i^2 \\ \vdots \\ y_i^d \end{pmatrix} \right)^2 = \left\| \begin{pmatrix} v^1 & v^2 & \cdots & v^d \end{pmatrix} \begin{pmatrix} y_1^1 & y_2^1 & \cdots & y_n^1 \\ y_1^2 & y_2^2 & & y_n^2 \\ \vdots & \vdots & & \vdots \\ y_1^d & y_2^d & \cdots & y_n^d \end{pmatrix} \right\|^2 = \| \mathbf{v}^{\mathbf{T}} Y \|^2$$

Directions of maximal variance

- So, we have: $\mathrm{var}(L) = \langle S\mathbf{v}, \mathbf{v} \rangle$
- Theorem:

  Let $f : \{\mathbf{v} \in R^d \mid ||\mathbf{v}|| = 1\} \rightarrow R,$

  $f(\mathbf{v}) = \langle S\mathbf{v}, \mathbf{v} \rangle$ (and S is a symmetric matrix).

  Then, the extrema of f are attained at the eigenvectors of S.

- So, eigenvectors of S are directions of maximal/minimal variance!

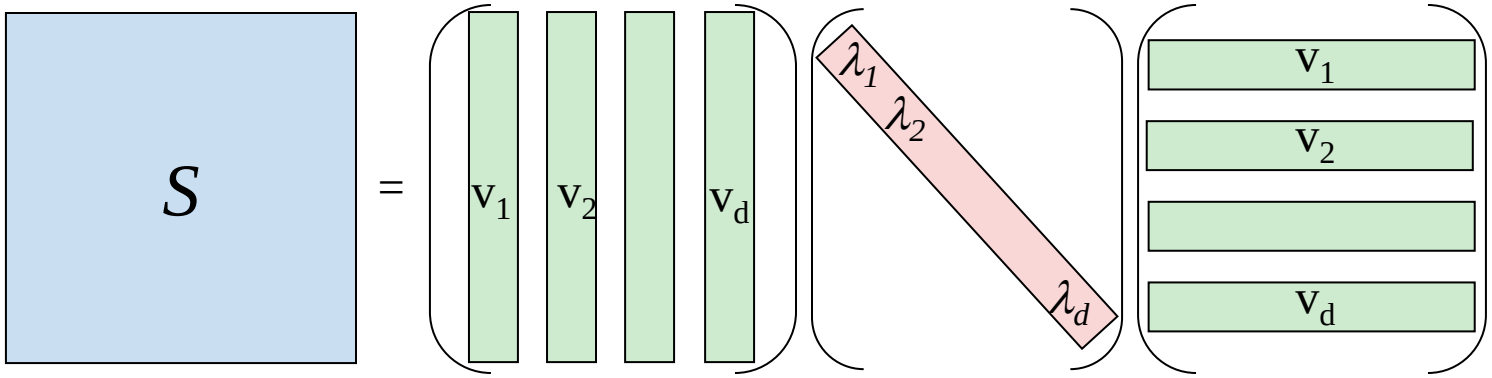
Summary so far

- We take the centered data vectors $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n \in R^d$
- Construct the scatter matrix $S = YY^T$
- S measures the variance of the data points
- Eigenvectors of S are directions of maximal variance.

Scatter matrix - eigendecomposition

- $\boldsymbol{S}$ is symmetric

$\Rightarrow$ $\boldsymbol{S}$ has eigendecomposition: $\boldsymbol{S} = V\Lambda V^{\mathrm{T}}$



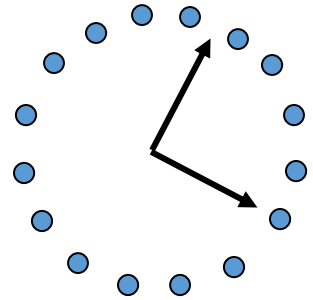
The eigenvectors form orthogonal basis

Principal components

- Eigenvectors that correspond to big eigenvalues are the directions in which the data has strong components (= large variance).
- If the eigenvalues are more or less the same – there is no preferable direction.

- Note: the eigenvalues are always non-negative. Think why…

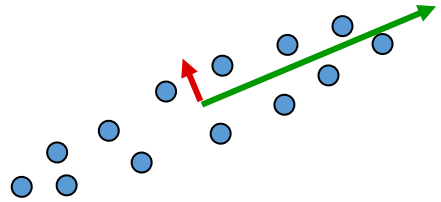
Principal components



- There's no preferable direction
- $\boldsymbol{S}$ looks like this:

$$\boldsymbol{V}\begin{pmatrix}\lambda & \\ & \lambda\end{pmatrix}\boldsymbol{V}^T$$

- Any vector is an eigenvector



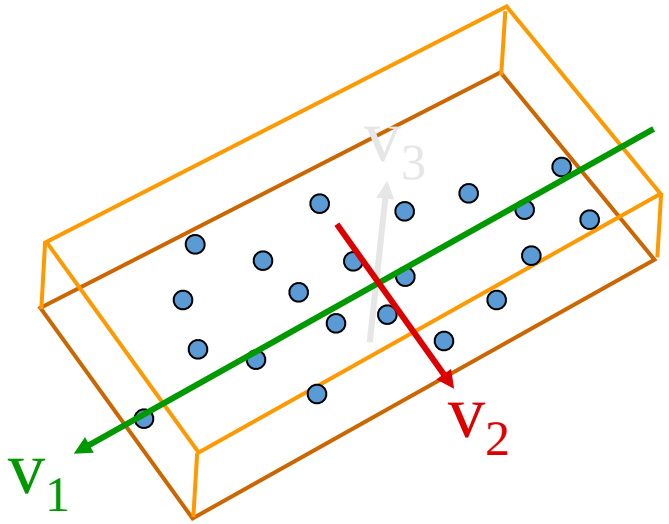
- There is a clear preferable direction
- $\boldsymbol{S}$ looks like this:

$$\boldsymbol{V}\begin{pmatrix}\lambda & \\ & \mu\end{pmatrix}\boldsymbol{V}^T$$

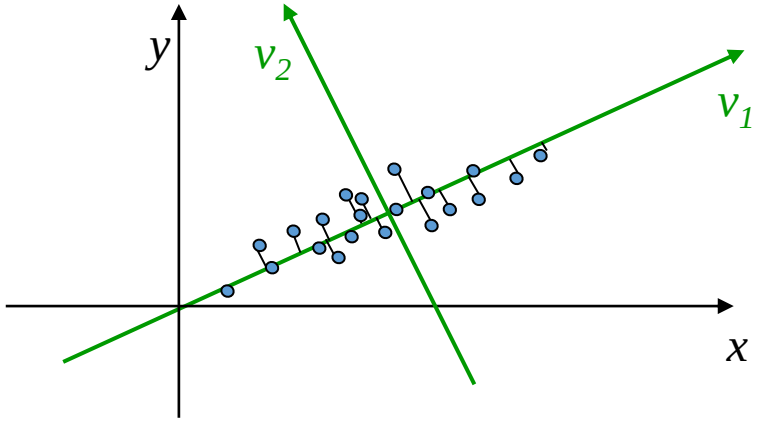
- μ is close to zero, much smaller than λ.

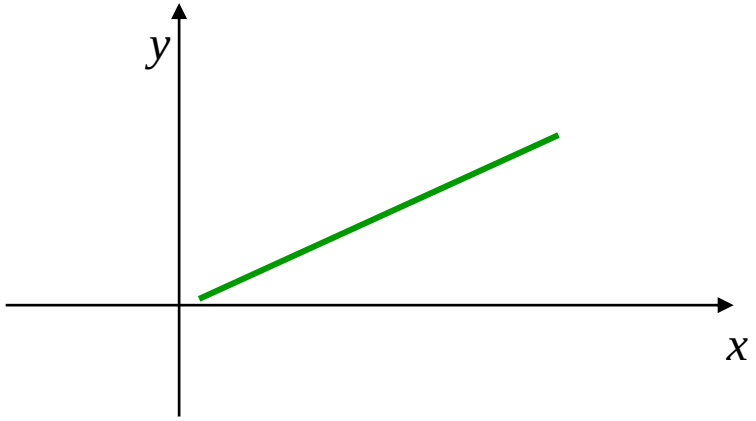
How to use what we got

- For finding oriented bounding box – we simply compute the bounding box with respect to the axes defined by the eigenvectors. The origin is at the mean point $\mathbf{m}$.

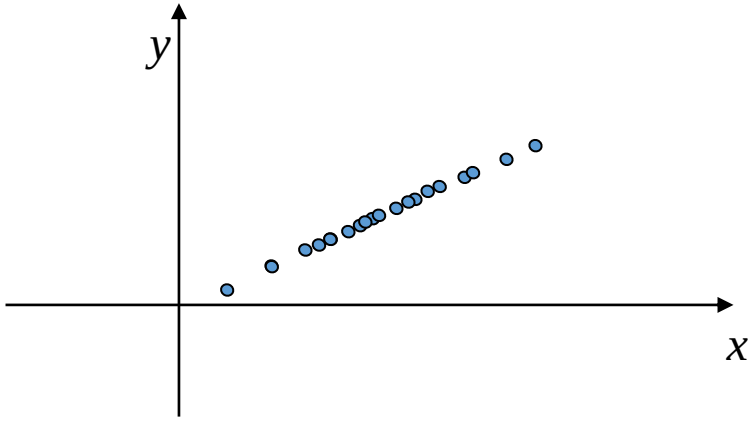


For approximation





This line segment approximates the original data set



The projected data set approximates the original data set

For approximation

- In general dimension d, the eigenvalues are sorted in descending order:

$$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$$

- The eigenvectors are sorted accordingly.
- To get an approximation of dimension $d' < d$, we take the d' first eigenvectors and look at the subspace they span ($d' = 1$ is a line, $d' = 2$ is a plane…)

For approximation

- To get an approximating set, we project the original data points onto the chosen subspace:

$$\mathbf{x}_i = \mathbf{m} + \alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2 + \ldots + \alpha_{d'} \mathbf{v}_{d'} + \ldots + \alpha_d \mathbf{v}_d$$

Projection:

$$\mathbf{x}_i' = \mathbf{m} + \underbrace{\alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2 + \ldots + \alpha_{d'} \mathbf{v}_{d'}} + 0 \cdot \mathbf{v}_{d'+1} + \ldots + 0 \cdot \mathbf{v}_d$$

Technical remarks:

- $\lambda_i \geq 0,\ i = 1,\ldots,d$ (such matrices are called positive semi-definite). So we can indeed sort by the magnitude of λ_i
- Theorem: $\lambda_i \geq 0 \quad \Leftrightarrow \quad <S\mathbf{v}, \mathbf{v}> \geq 0 \quad \forall \mathbf{v}$

Proof:

$$S = V\Lambda V^T \quad \Rightarrow \quad < S\boldsymbol{v}, \boldsymbol{v} > = \boldsymbol{v}^T S\boldsymbol{v} = \boldsymbol{v}^T V\Lambda V^T \boldsymbol{v}$$
$$= (V^T\boldsymbol{v})^T \Lambda (\boldsymbol{v}^T V) = \boldsymbol{u}^T \Lambda \boldsymbol{u} =< \Lambda\boldsymbol{u}, \boldsymbol{u} >$$

$$\boxed{< S\mathbf{v}, \mathbf{v} >= \lambda_1 \mathbf{u_1}^2 + \lambda_2 \mathbf{u_2}^2 + \ldots + \lambda_d \mathbf{u_d}^2}$$

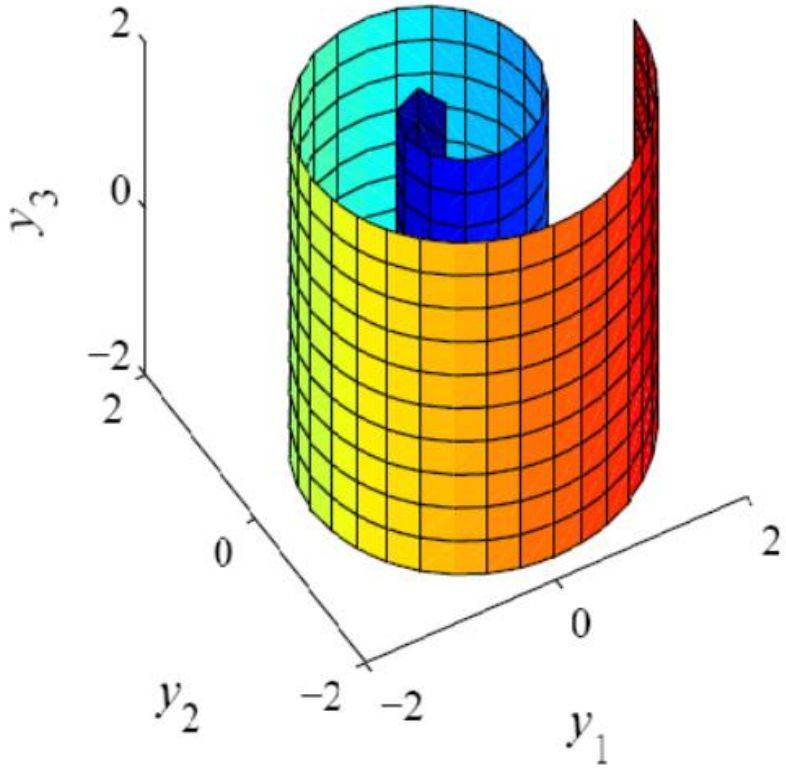
Therefore, $\lambda_i \geq 0 \Leftrightarrow <S\mathbf{v}, \mathbf{v}> \geq 0 \quad \forall \mathbf{v}$

Visualizing Data using t-SNE

The slides are from https://sites.uclouvain.be/ispgroup/uploads/ISPS/slides_120516.pdf

Dimensionality Reduction is a helpful tool for visualization

- Dimensionality reduction algorithms
  - Map high-dimensional data to a lower dimension
  - While preserving structure
- They are used for
  - Visualization
  - Performance
  - Curse of dimensionality
- A ton of algorithms exist
  - t-SNE is specialised for visualization and has gained a lot of popularity



Dimensionality Reduction techniques solve optimization problems

$$X = \{x_1, x_2, \dots, x_n \in R^h\} \Rightarrow Y = \{y_1, y_2, \dots, y_n \in R^l\}$$

$$\min_y C(X, Y)$$

- Three approaches for Dimensionality Reduction
  - Distance preserving
  - Topology preserving
  - Information preserving
- t-SNE is distance-based but tends to preserve topology

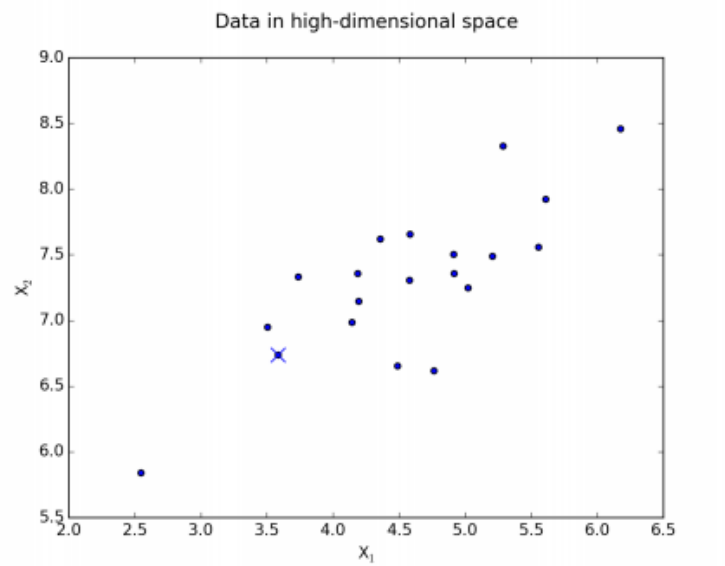
SNE computes pair-wise similarities

$$p_{j|i} = \frac{exp\left(-\left\|x_i - x_j\right\|^2 / 2\sigma_i^2\right)}{\sum_{k!=i} exp\left(-\|x_i - x_k\|^2 / 2\sigma_i^2\right)}, \quad q_{j|i} = \frac{exp\left(-\left\|y_i - y_j\right\|^2\right)}{\sum_{k!=i} exp\left(-\|y_i - y_k\|^2\right)}$$

$$p_{i|i} = 0, \quad q_{i|i} = 0$$

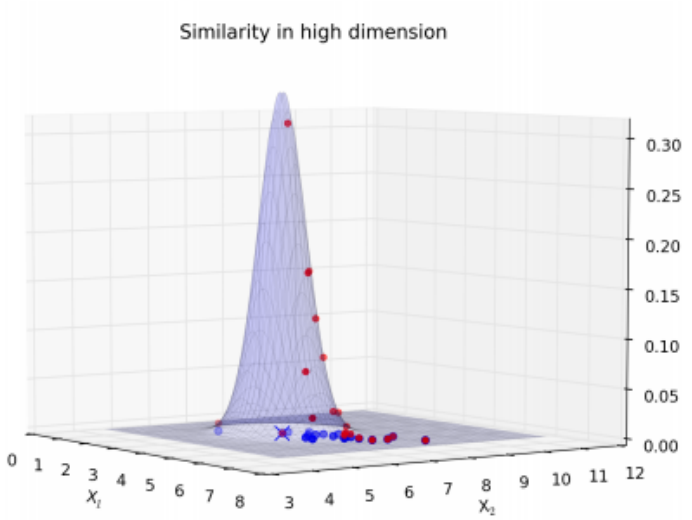
- SNE converts euclidean distances to similarities, that can be interpreted as probabilities

Pair-wise similarities should stay the same

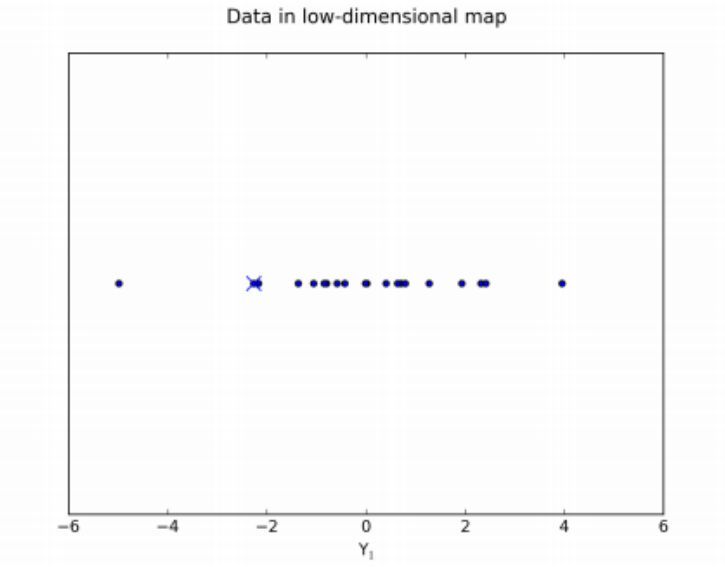


$p_{j|i}$

$\Leftrightarrow$

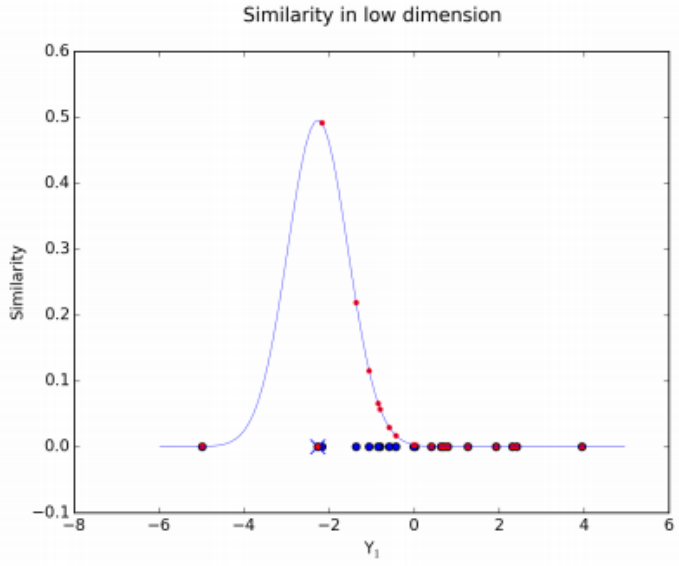


$\Downarrow$



$q_{j|i}$

$\Leftrightarrow$



Kullback-Leiber Divergence measures the faithfulness with wich $q_{j|i}$ models $p_{j|i}$

- $P_i = \{p_{1|i}, p_{2|i}, \dots, p_{n|i}\}$ and $Q_i = \{q_{1|i}, q_{2|i}, \dots, q_{n|i}\}$ are the distribution on the neighbors of point i.
- Kullback-Leiber Divergence (KL) compares two distributions.

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}}$$

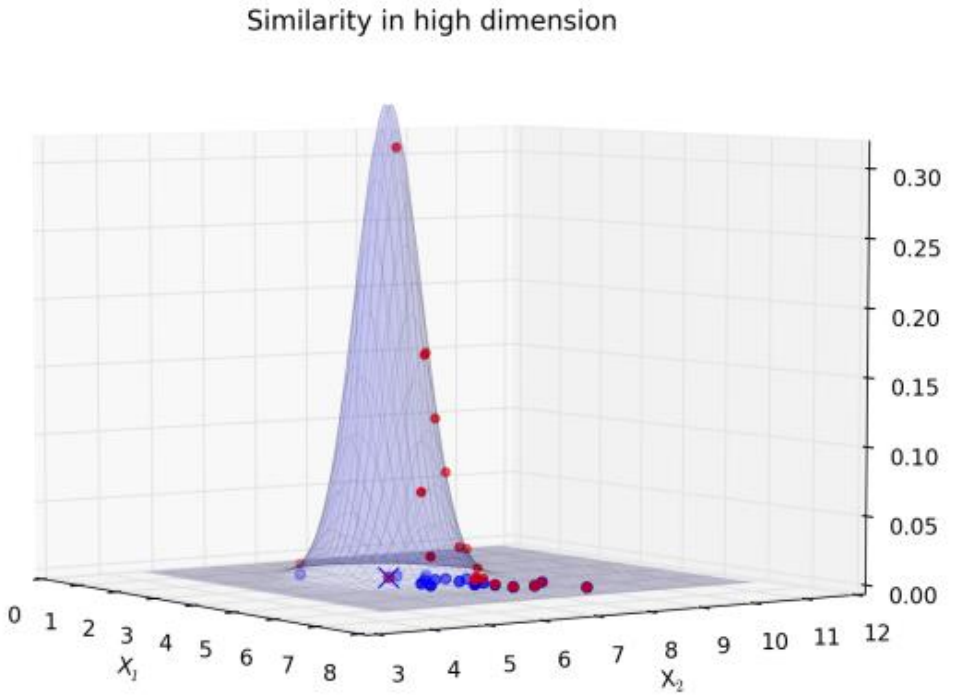
- KL divergence is asymmetric
- KL divergence is always positive.
- We have our minimization problem: $\min_y C(X, Y)$

Why radial basis function (exponential)?

$$p_{j|i} = \frac{exp\left(-\left\|x_i - x_j\right\|^2 / 2\sigma_i^2\right)}{\sum_{k!=i} exp\left(-\left\|x_i - x_k\right\|^2 / 2\sigma_i^2\right)}, \quad q_{j|i} = \frac{exp\left(-\left\|y_i - y_j\right\|^2\right)}{\sum_{k!=i} exp\left(-\left\|y_i - y_k\right\|^2\right)}$$

Focus on local geometry.

This is why t-SNE can be interpreted as topology-based

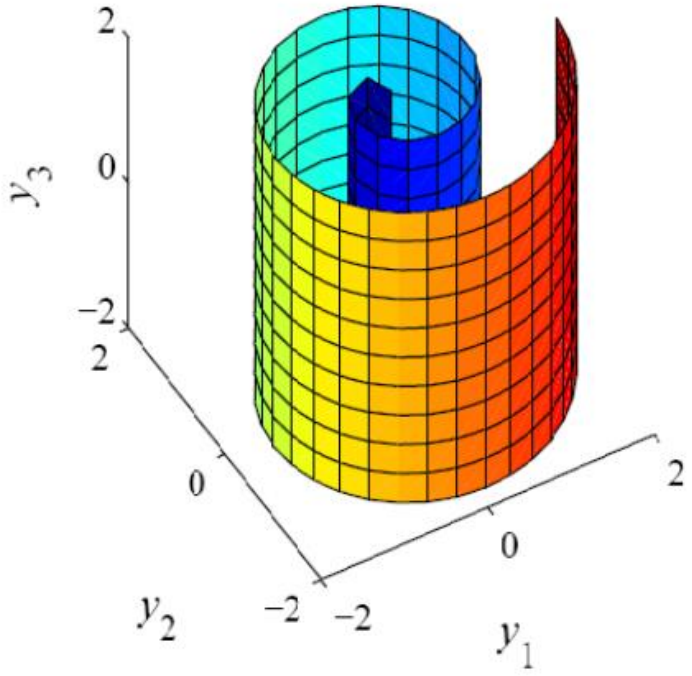


Why probabilities?

$$p_{j|i} = \frac{exp\left(-\|x_i - x_j\|^2 / 2\sigma_i^2\right)}{\sum_{k!=i} exp\left(-\|x_i - x_k\|^2 / 2\sigma_i^2\right)}, \quad q_{j|i} = \frac{exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k!=i} exp\left(-\|y_i - y_k\|^2\right)}$$

Small distance does not mean proximity on manifold.

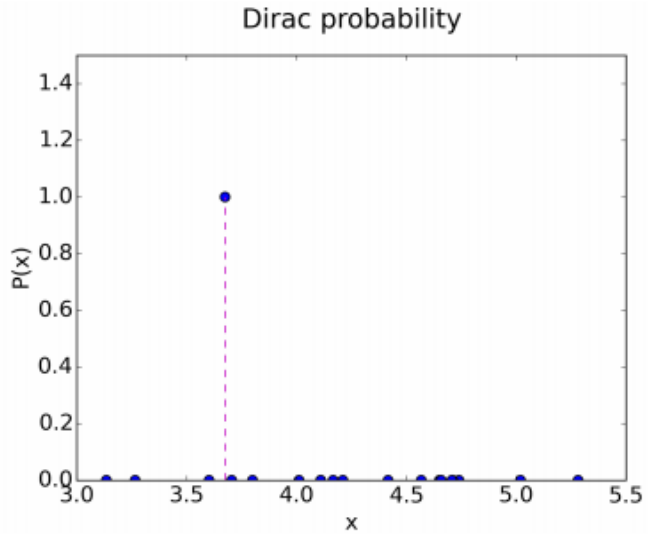
Probabilities are appropriate to model this uncertainty.

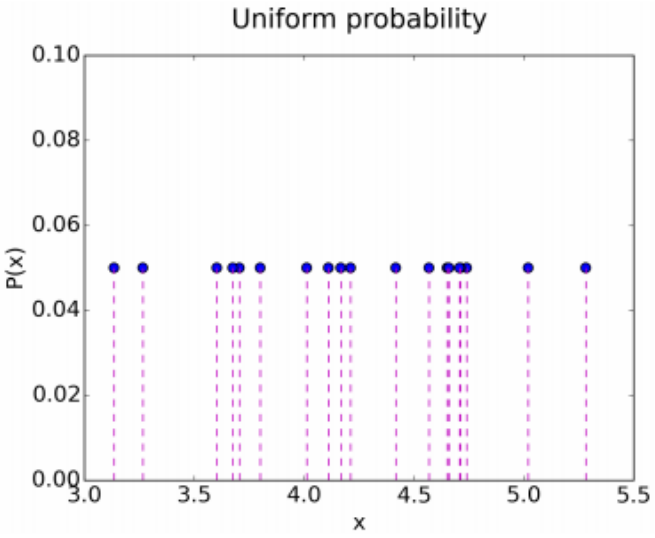


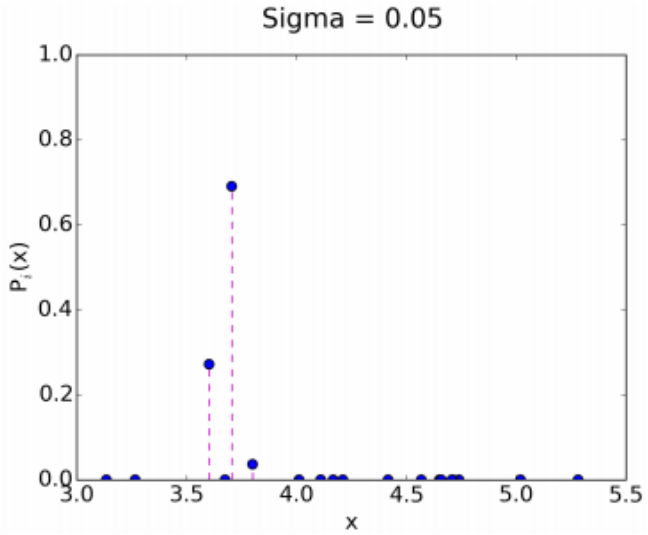
How do you choose σ_i ?

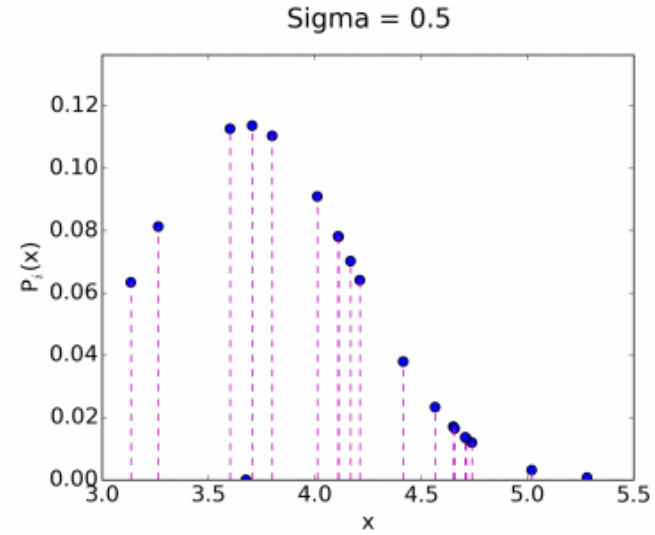
$$H(P) = -\sum_i p_i \log p_i$$

The entropy of P_i increases with σ_i

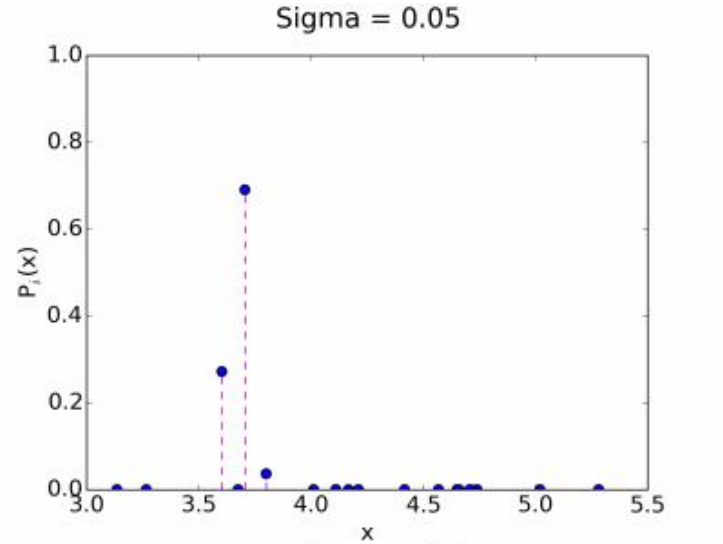




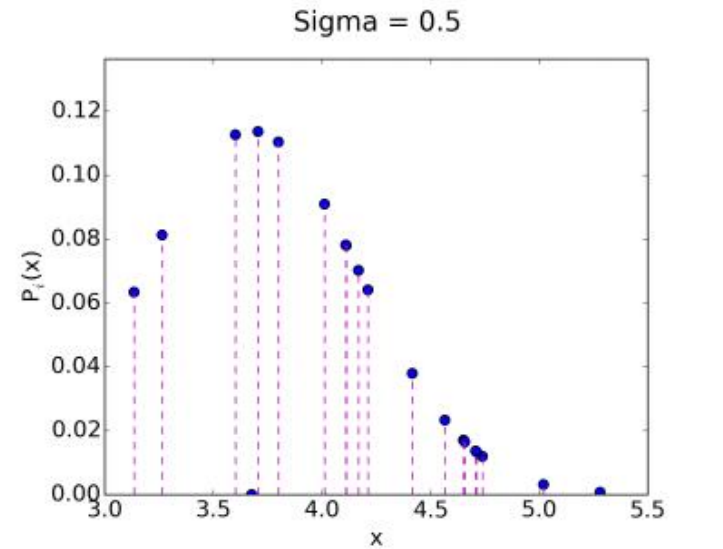




Perplexity, a smooth measure of the # of neighbors.



$\Rightarrow$ Entropy of 1.055
Perplexity of 2.078



$\Rightarrow$ Entropy of 3.800
Perplexity of 13.929

From SNE to t-SNE

SNE

Modelisation

$$p_{j|i} = \frac{exp\left(-\|x_i - x_j\|^2 / 2\sigma_i^2\right)}{\sum_{k!=i} exp(-\|x_i - x_k\|^2 / 2\sigma_i^2)},$$

$$q_{j|i} = \frac{exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k!=i} exp(-\|y_i - y_k\|^2)}$$

Cost Function

$$C = \sum_i KL(P_i||Q_i)$$

Derivatives

$$\frac{dC}{dy_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

Symmetric SNE

Modelisation

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n},$$

$$q_{ij} = \frac{exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k!=i} exp(-\|y_i - y_k\|^2)}$$

Cost Function

$$C = \sum KL(P||Q)$$

Derivatives

$$\frac{dC}{dy_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)$$

t-SNE

Modelisation

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n},$$

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k!=i} (1 + \|y_i - y_k\|^2)^{-1}}$$

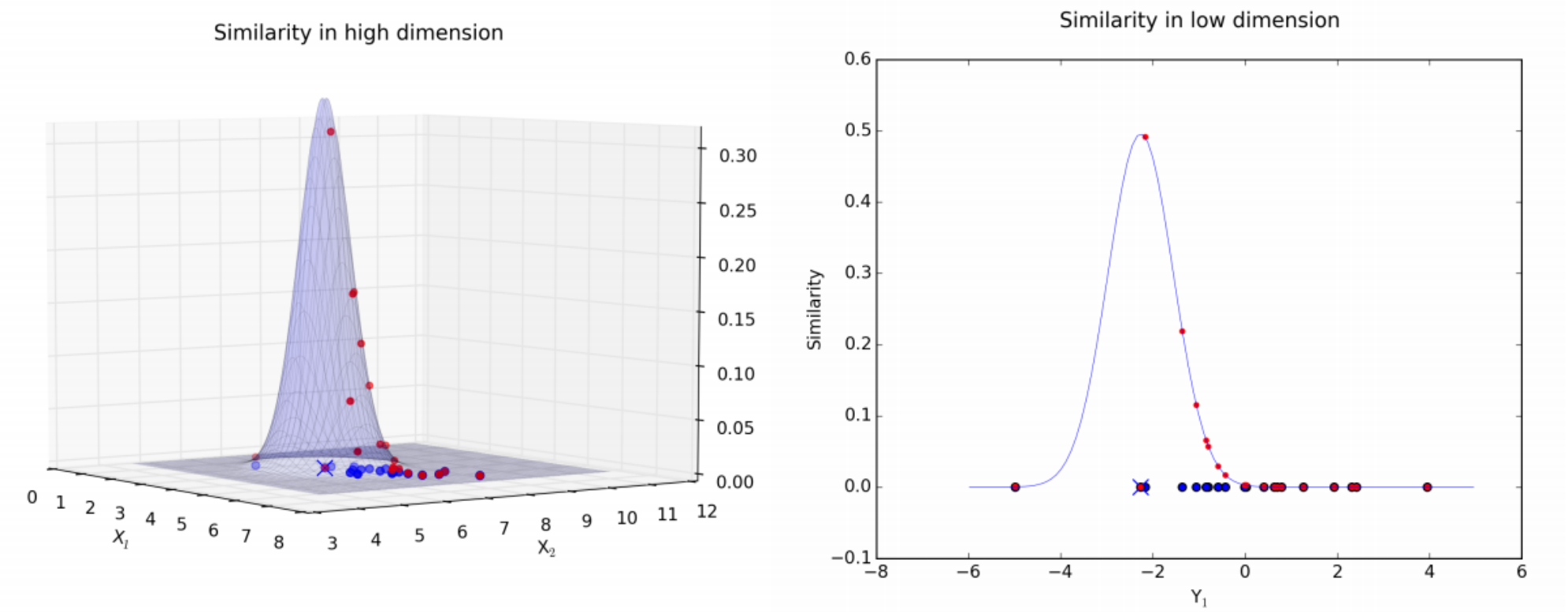
Cost Function

$$C = \sum KL(P||Q)$$

Derivatives

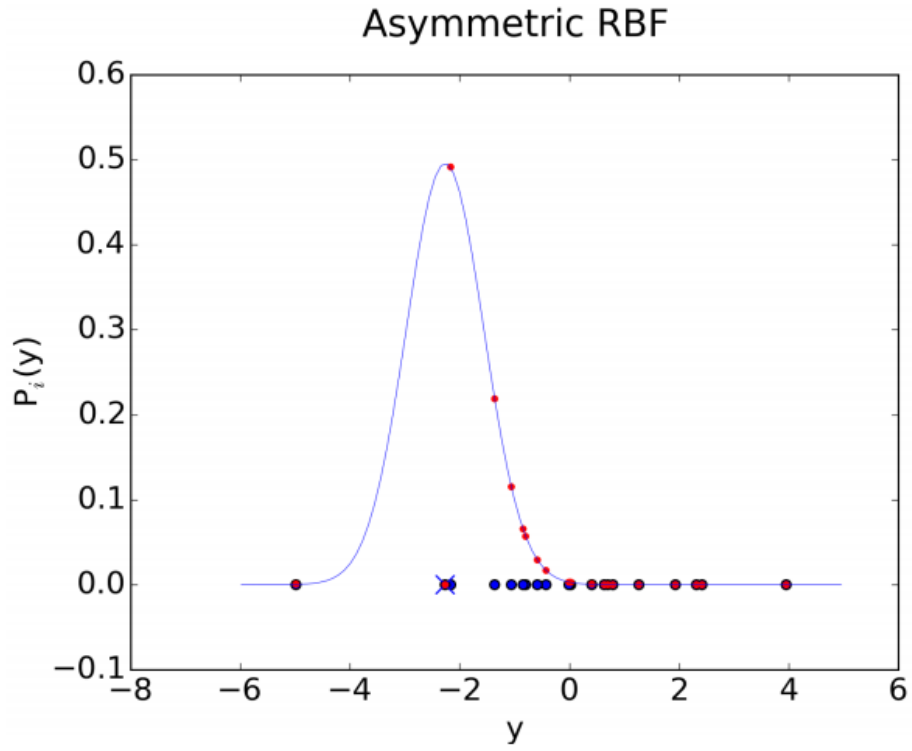
$$\frac{dC}{dy_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)\left(1 + \|y_i - y_j\|^2\right)^{-1}$$

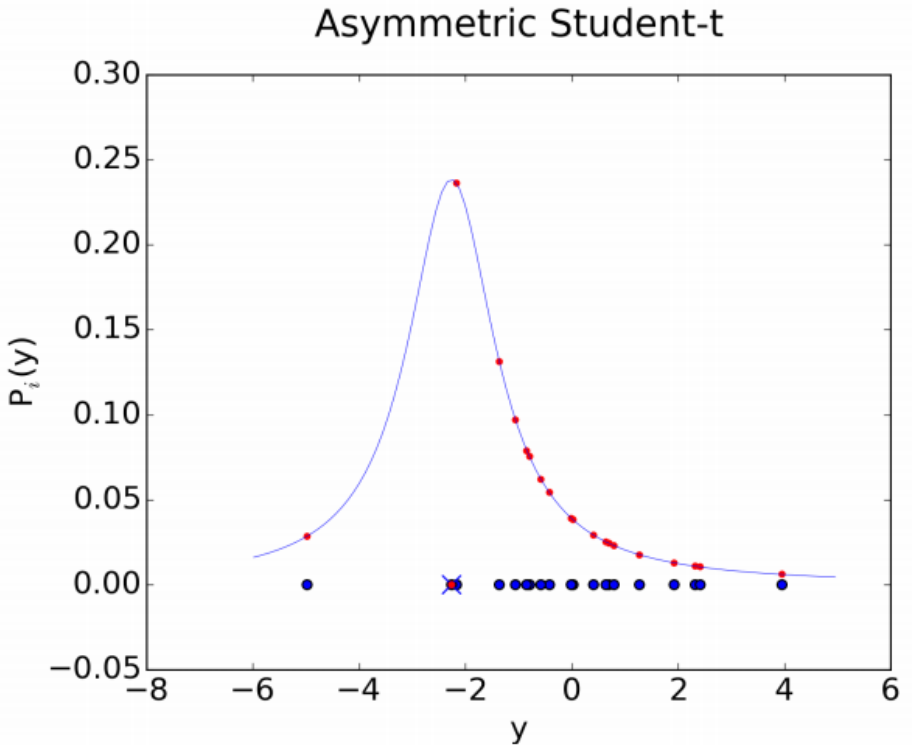
The ”Crowding problem”



- There is much more space in high dimensions.

Mismatched Tails can Compensate for Mismatched Dimensionalities





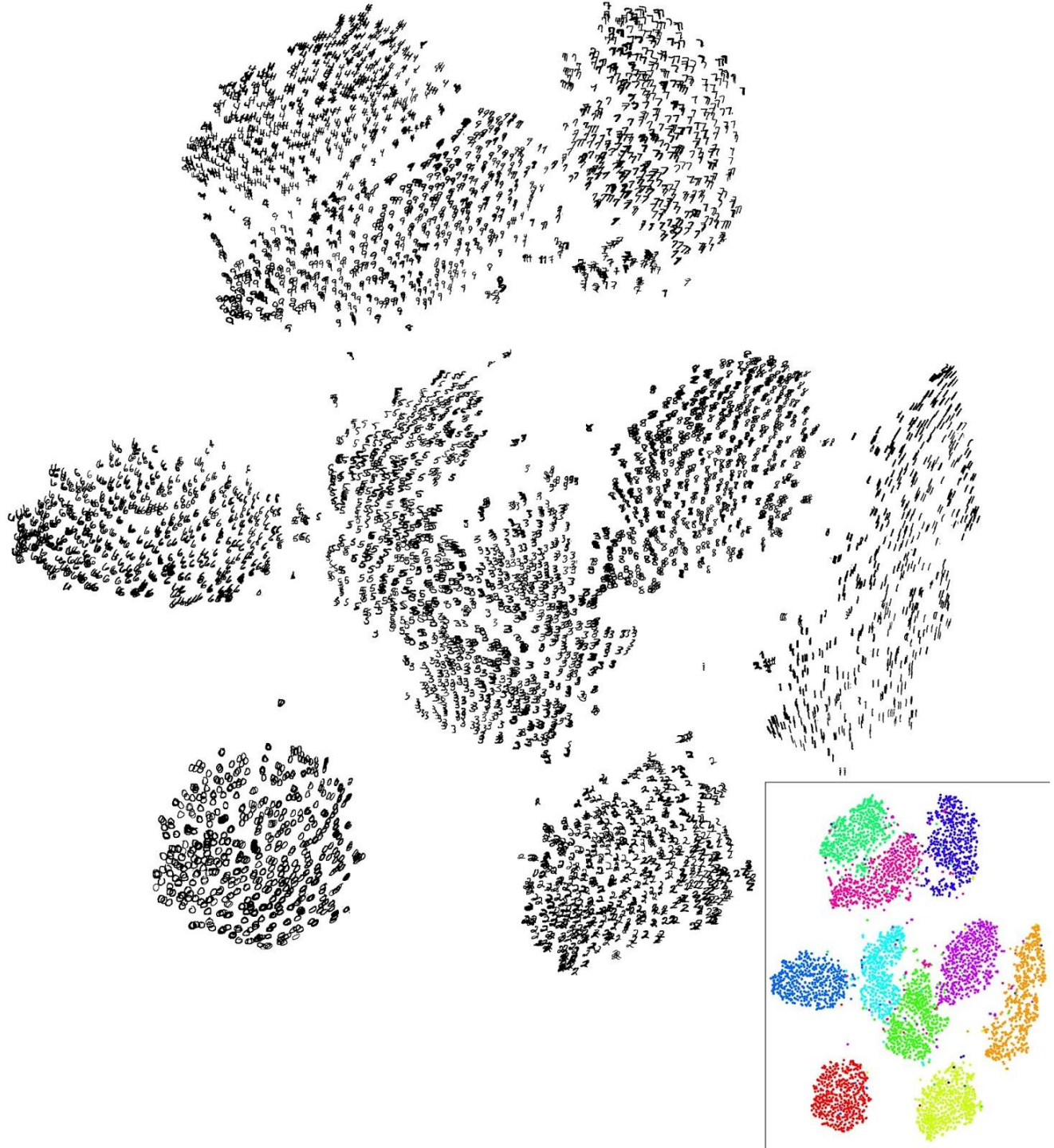
- Student-t distribution has heavier tails.

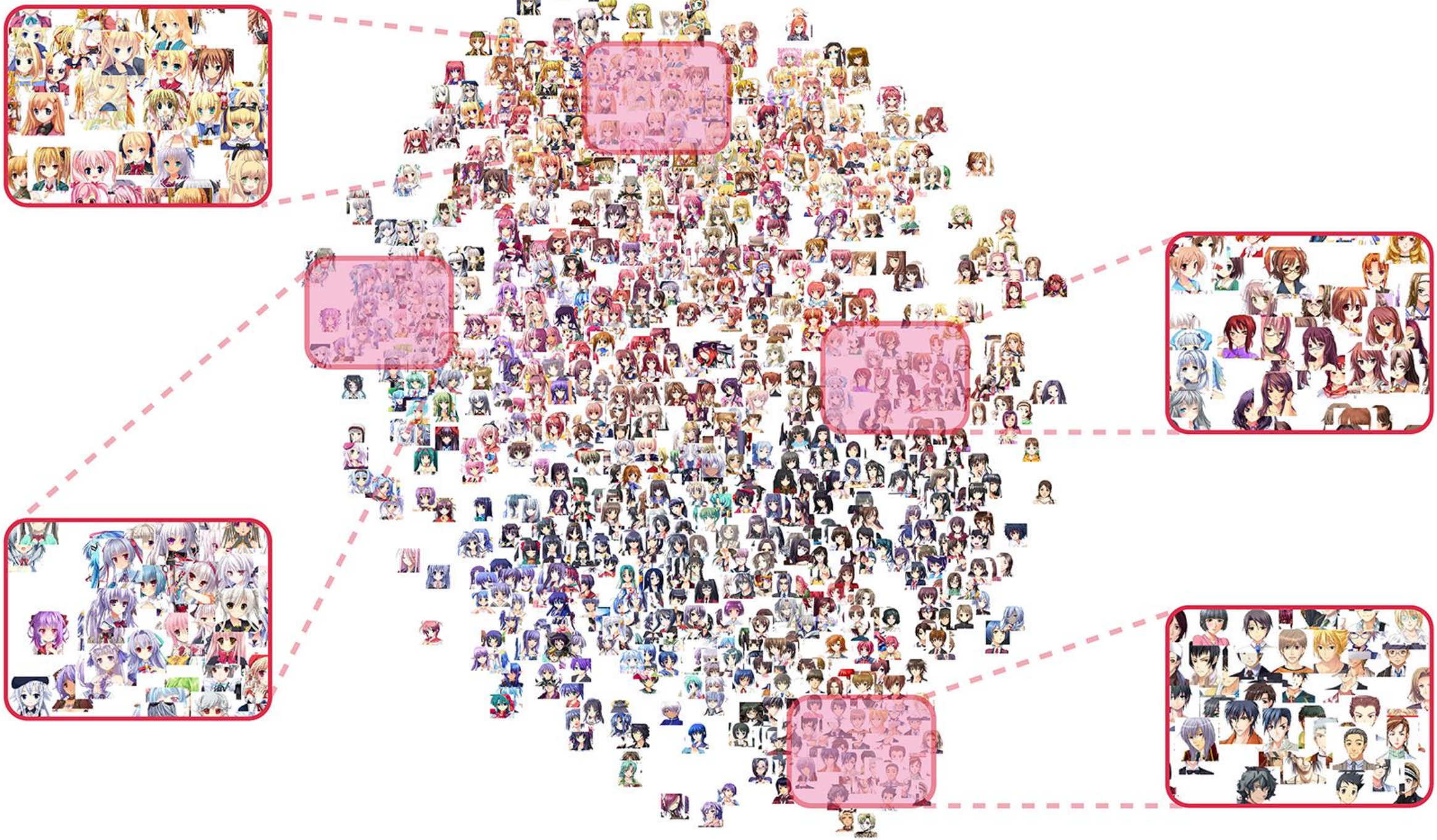
Last but not least: Optimization

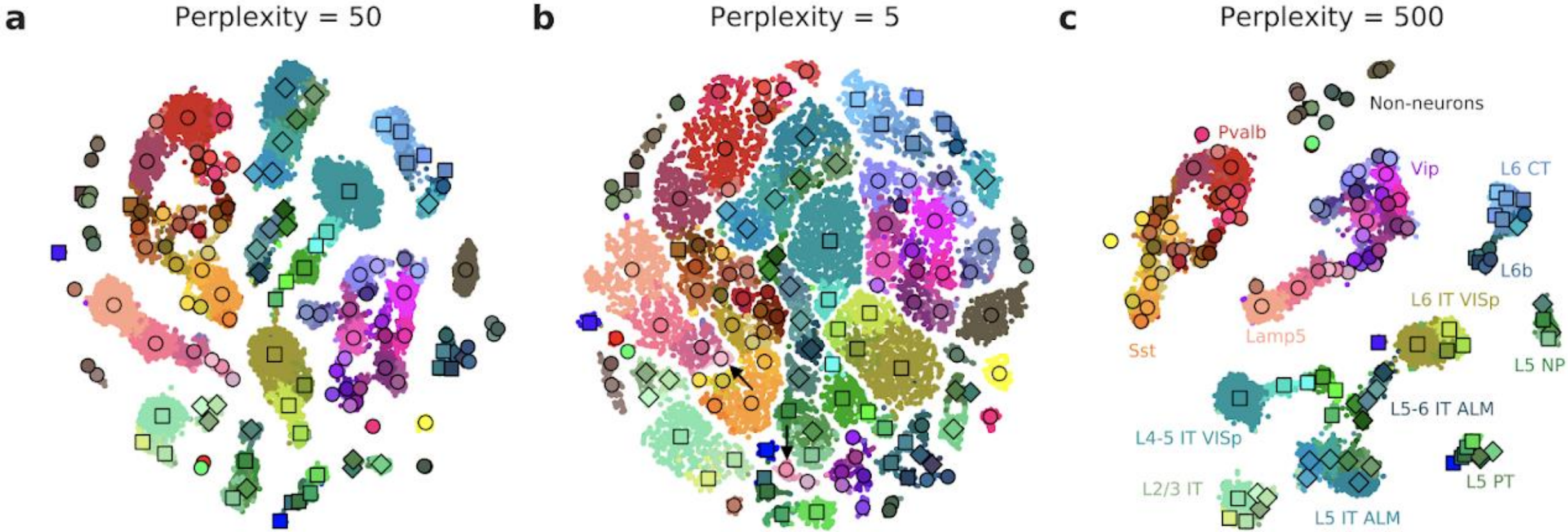
$$\min_{y} C(X, Y)$$

$$C = \sum_{i} KL(P_i||Q_i) = \sum_{i}\sum_{j} p_{j|i} log \frac{p_{j|i}}{q_{j|i}}$$

- Non-convex
- Gradient descent + Adaptive learning rate + Momentum
- $y^t = y^{t-1} + \eta \frac{\partial C}{\partial Y} + \alpha(y^{t-1} - y^{t-2})$






a
Perplexity = 50
b
Perplexity = 5
c
Perplexity = 500
Non-neurons
Pvalb
Vip
L6 CT
L6b
L6 IT VISp
Lamp5
Sst
L5 NP
L5-6 IT ALM
L4-5 IT VISp
L5 PT
L2/3 IT
L5 IT ALM