Comparing Nonlinear Dimensionality Reduction Methods with Large Real-World Dataset

CSci Numerical Linear Algebra in Data Exploration

Term Project Presentation

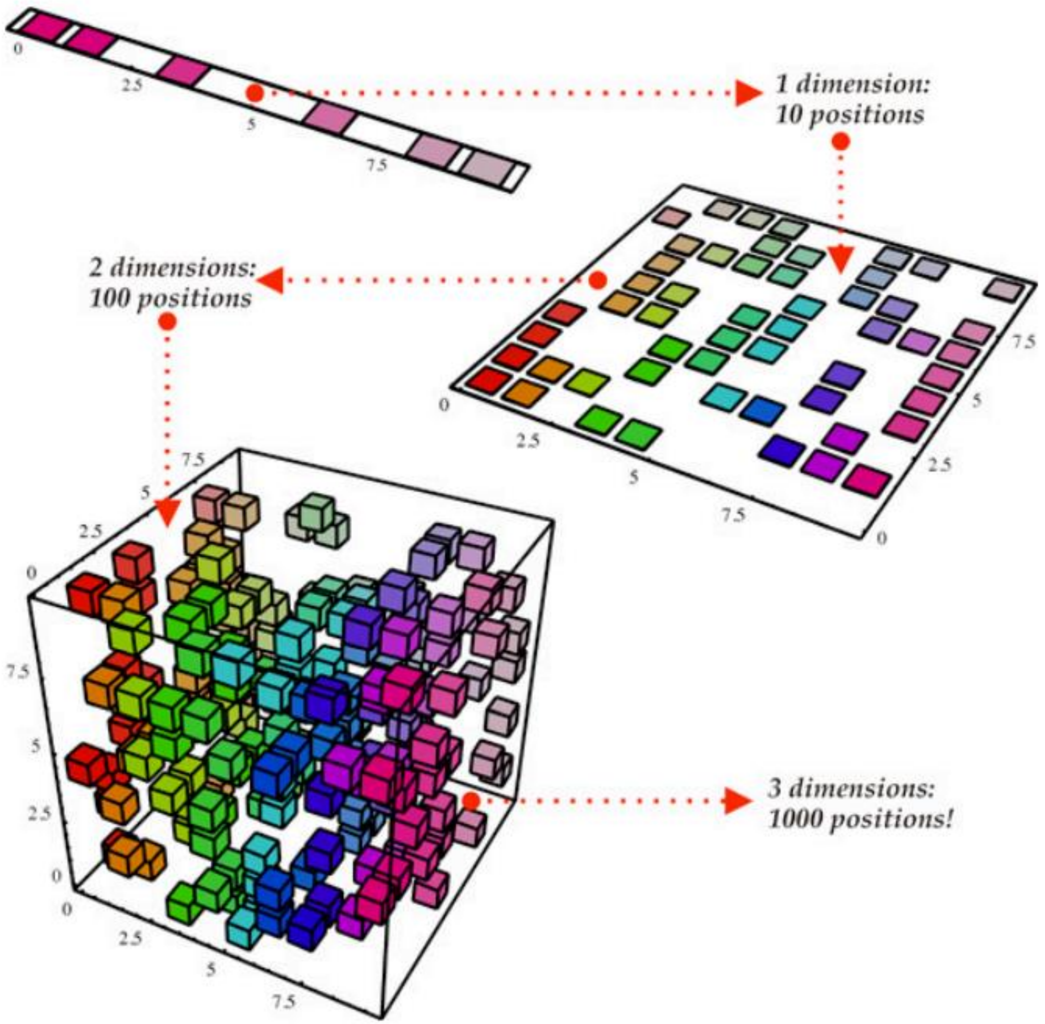
Qun Su

12-6-2017

University of Minnesota, Twin Cities

Motivation

- Real world data often nonlinear, with high dimensionality
  - A proper way to reduce their dimensionality with minimum loss of information is needed
- Wide range of methods are available
  - Proper choice and optimization
- In this work, three DR methods, LLE, k-PCA, and t-SNE will be compared with MNIST hand written digit dataset



Outline

- Introduction
  - Local linear embedding (LLE)
  - Kernel principal component analysis (K-PCA)
  - t-distributed stochastic neighbor embedding (t-SNE)
- Performance comparison on artificial data sets
- Experiment on MINIST data sets
- Conclusion

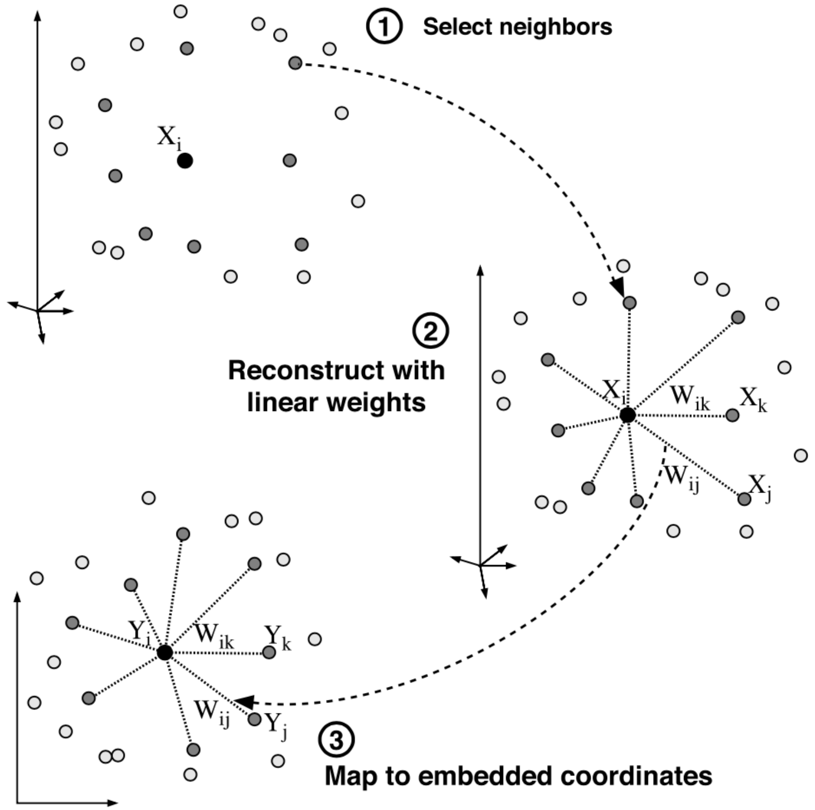
LLE

- Linear Local embedding reconstruct the dataset by representing each data point as a linear combination of its nearest neighbors [1].

$$\varepsilon(W) = \sum_i \left| \vec{X}_i - \Sigma_j W_{ij} \vec{X}_j \right|^2 \xrightarrow{\text{Solve for eigenvalue}} (I - W)^T (I - W)$$

Reconstruction error

- Solve the eigenvalue problem to minimize the reconstruction error
- Captures the local environment for each data point



S.T. Roweis et al, 2000

kernel PCA

- Select a kernel function ($\boldsymbol{\kappa}$) and compute the kernel matrix K of data points x_i.

$$k_{ij} = \kappa(x_i, x_j)$$

- Center $\phi(x_i)$, the transformation of data point x_i on the featured space at zero.
- Instead of explicitly compute ϕ's, we can simple solve for the eigenvalue problem

$$m\lambda\alpha = K\alpha$$

- Gaussian kernel: $K_{ij} = \exp(-\frac{||x_i - x_j||^2}{2t^2})$
- Polynomial kernel: $K_{ij} = (x_i \cdot x_j)^d$

t-distributed stochastic neighbor embedding

- Define a conditional probability of data point x_i having data point x_j as its neighbor using Gaussian probability density.
- This conditional probability can be computed in lower dimension, and should remain unchanged.

$$p_{j|i} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\|x_i - x_k\|^2 / 2\sigma_i^2\right)} \xrightarrow[\text{lower dimension}]{} q_{j|i} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k \neq i} \exp\left(-\|y_i - y_k\|^2\right)}$$

- Minimize the Kullback-Leibler divergence using gradient descent method. Cost function is expressed as

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

- Want to use varying σ_i for specific environment $\rightarrow$ search for σ_i according to a fixed perplexity, P_i

$$Perp(P_i) = 2^{H(P_i)} \qquad H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}.$$

- Perplexity is similar to the number of nearest neighbors.

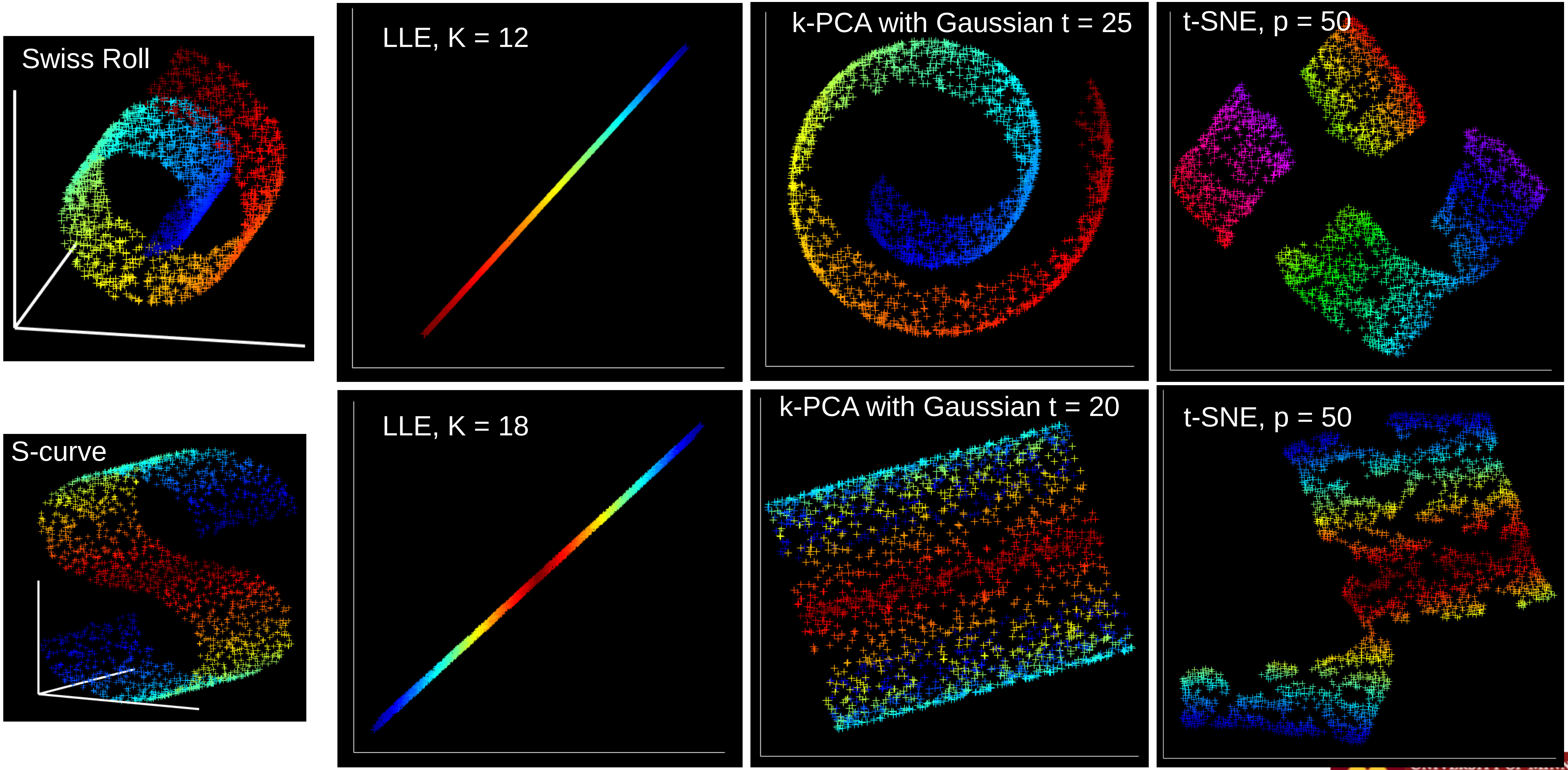
t-distributed stochastic neighbor embedding

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4} I)$
  for *t=1* **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)} \right)$ Gradient descent method
  end
end

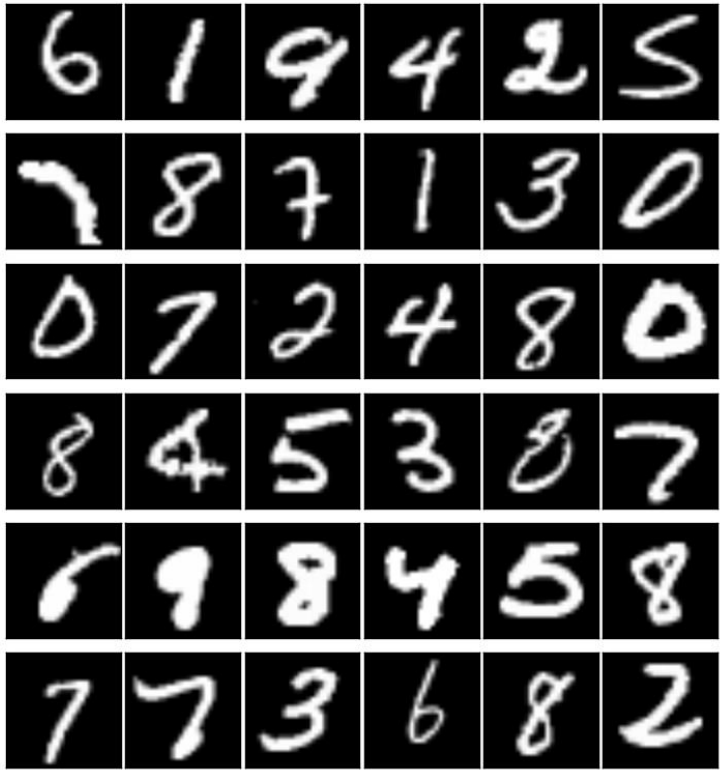
- In this work, t-SNE is done in 1000 iterations.

Examples with Artificial Data



Experiment setup

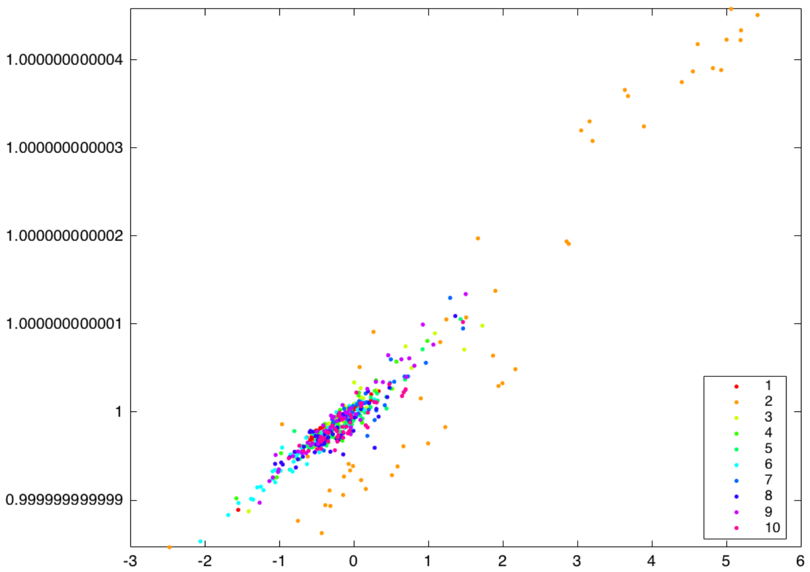
- MNIST hand-written digits (0~9) database
- Training set: 60000 28x28 gray scale images
- Testing set: 10000 28x28 gray scale images
- Randomly select from MNIST training set.

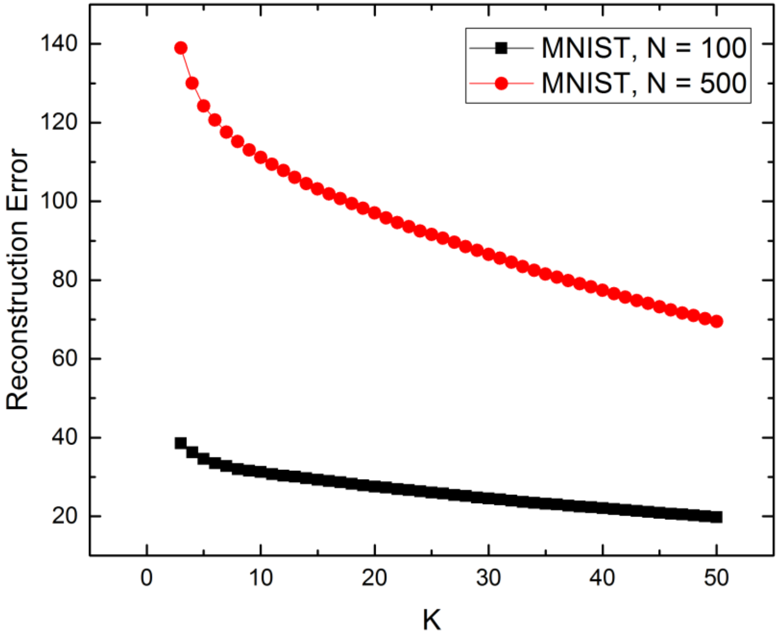


LLE on MNIST Data

- Reconstruction error with LLE is very large on MNIST, though on artificial data is much smaller.

N = 500, K = 25

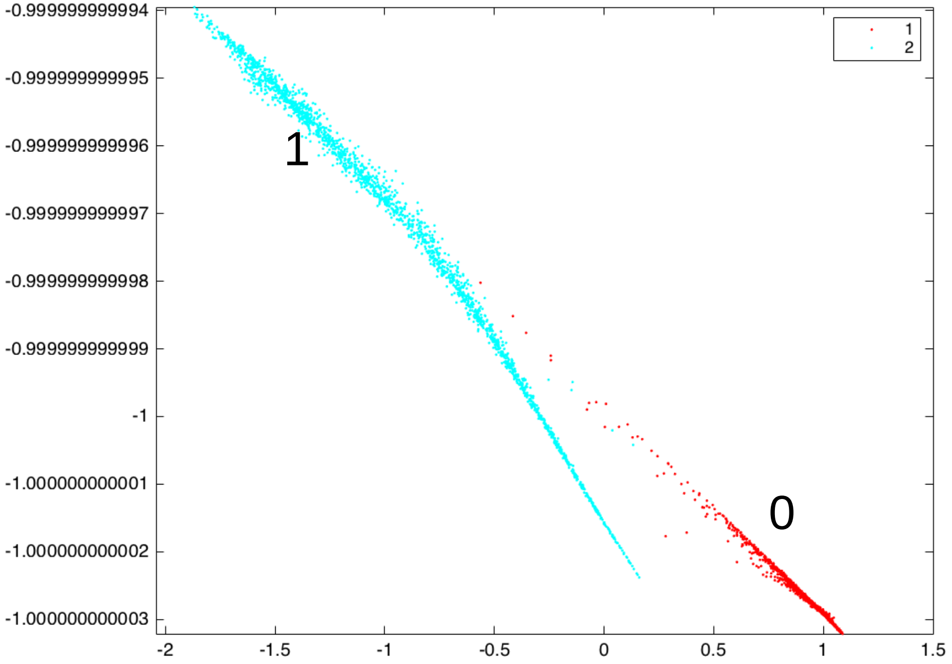




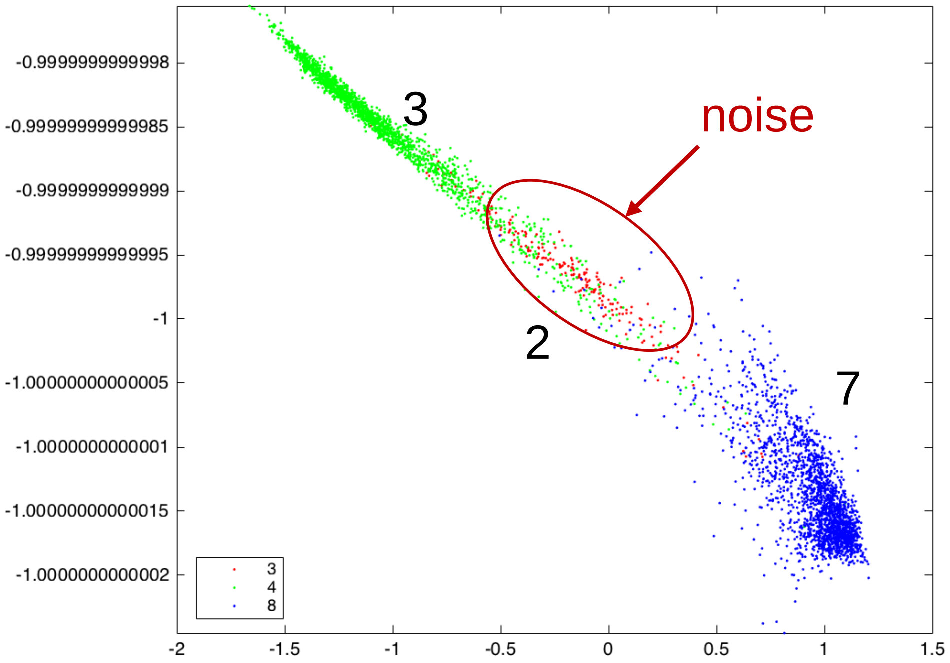
LLE with Fewer Digits

- If manipulate the input such that it contains only two digits, LLE is able to differentiate them
- Still suffers from noise.

N = 500+500, K = 18



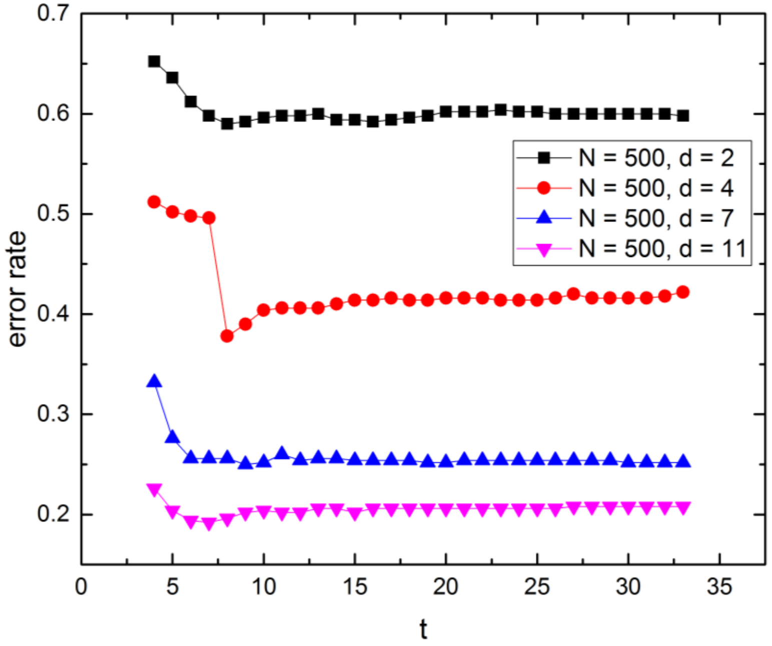
N = 2000+2000, K = 18



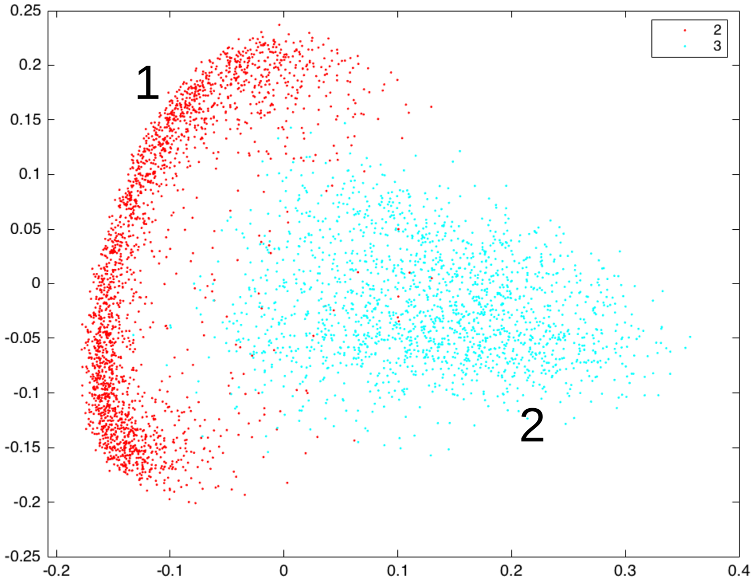
k-PCA on MNIST Data

- With Gaussian kernel, data mapped to 2D space shows large error rates. Input with less digits generates better results.
- Error rate can be lowered if the data is mapped to higher dimension.
- Overall, lower error rate if the Gaussian parameter increases.

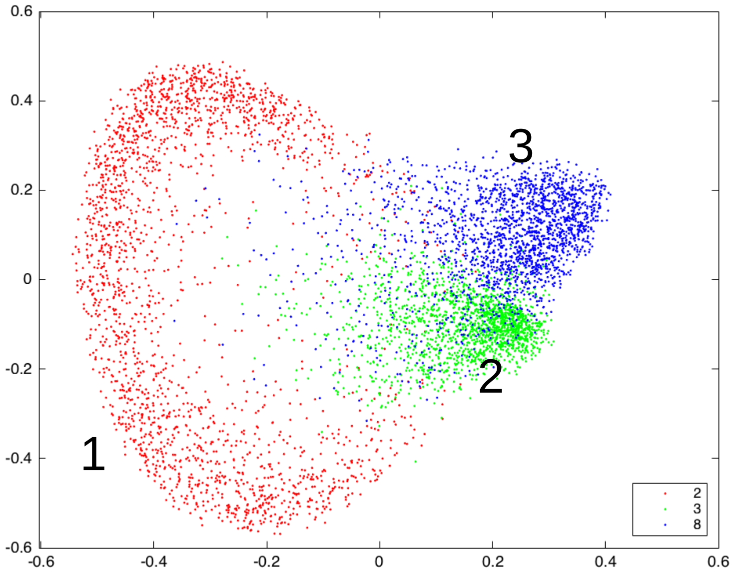
All 10 digits



N = 2000+2000, t = 20

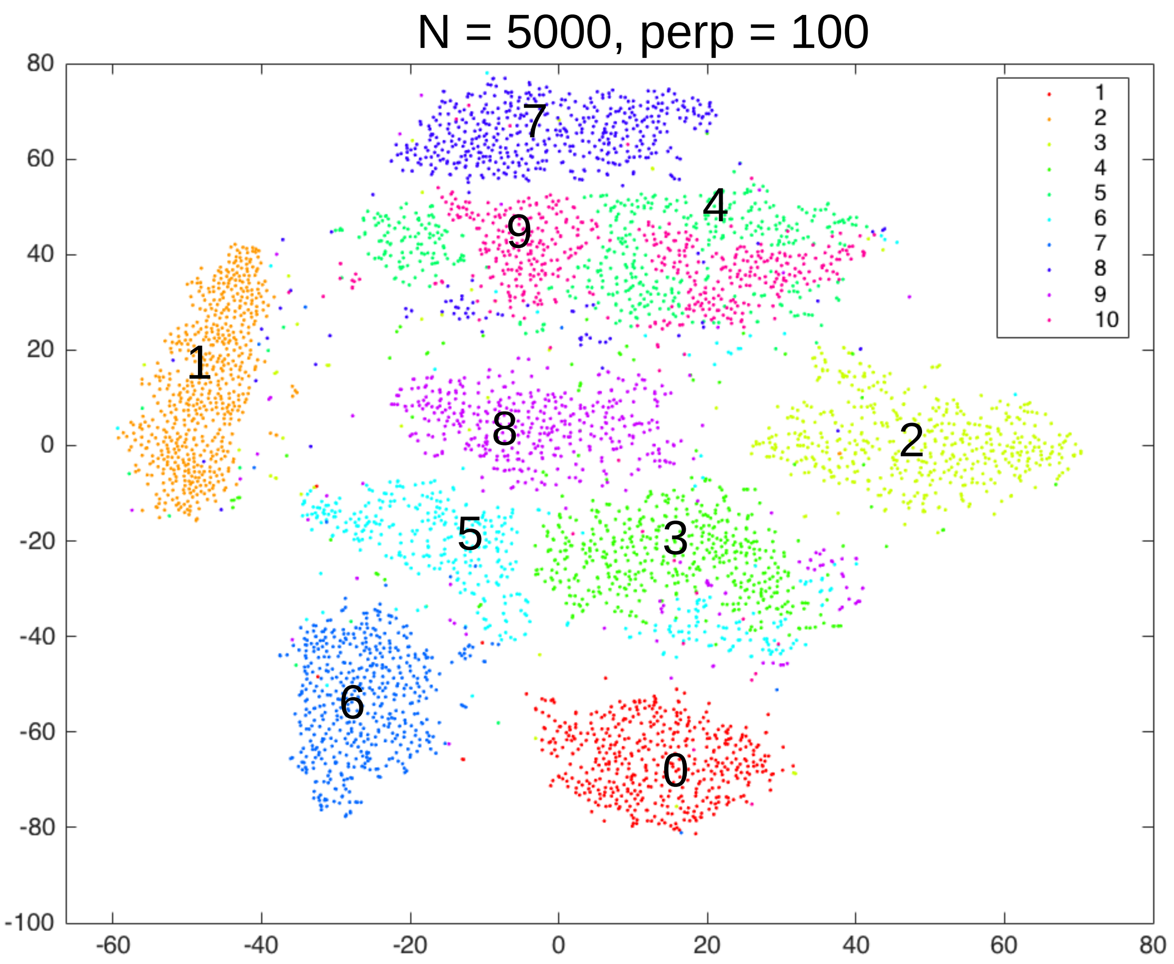


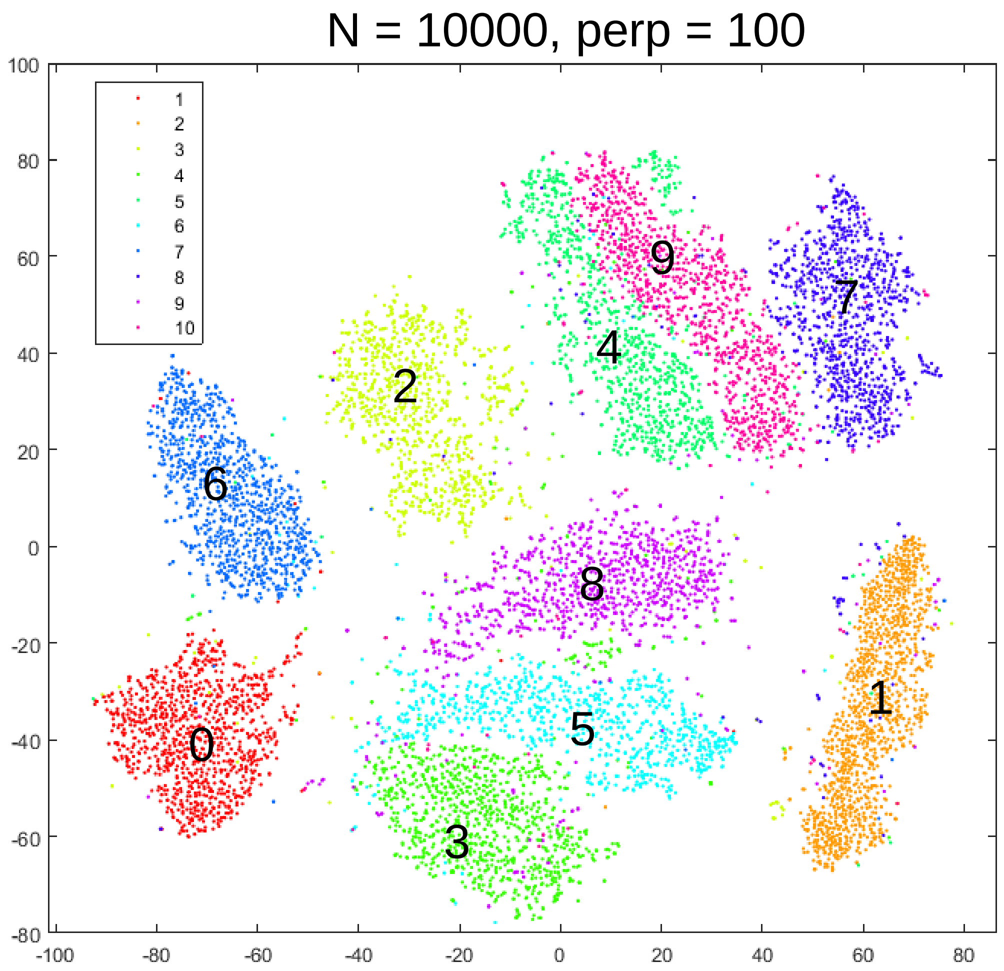
N = 2000+2000+2000, t = 5



No clear boundaries

t-SNE on MNIST Data

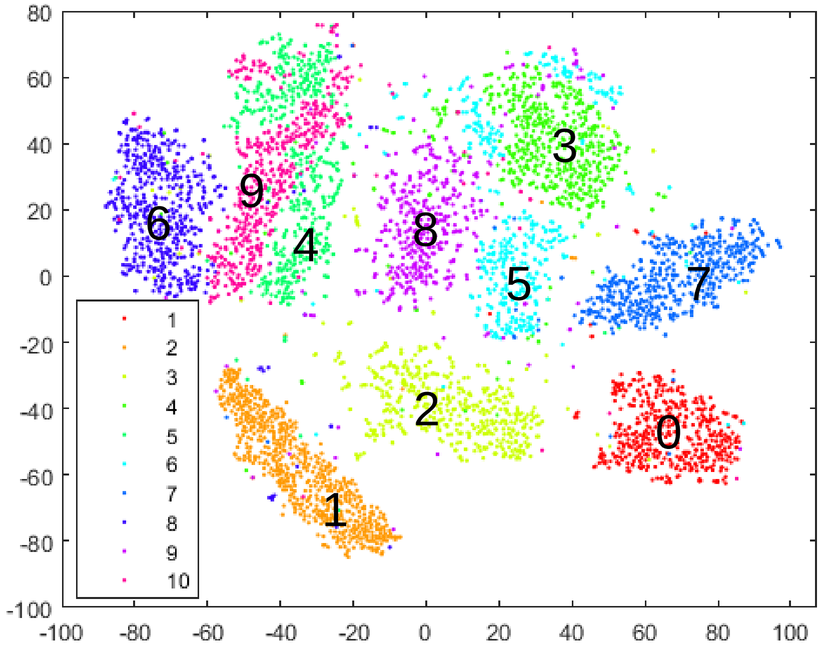


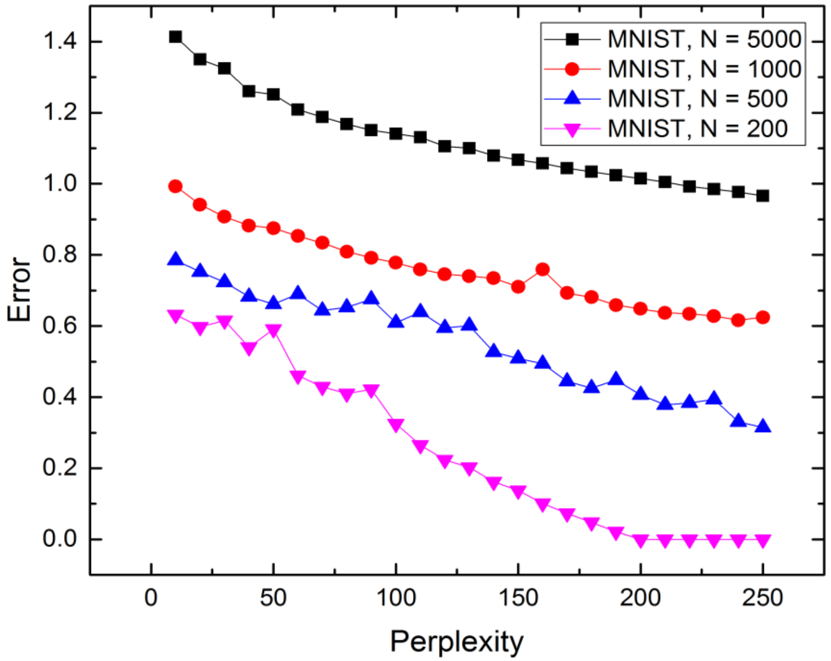


t-SNE on MNIST Data

- t-SNE is able differentiate the 10 digits well.
- Error can be reduced if large perplexity is selected. But no clear improvement in visualization.
- “4” and “9” are mixed up

N = 5000, perp = 250





Conclusions

- LLE produces large reconstruction error on MNIST dataset. It could work on binary input, while suffering from noise.
- Performance of kernel PCA mapping depends of the dimension of the target space and the Gaussian parameter. It could however distinguish the less digits are given.
- t-SNE work very well on large scale MINST data. The cost error decreases if higher perplexity was used, though it has limited improvement on data visualization.

Thanks!

Backup slides: LLE and t-SNE on Artificial Data