Nonlinear Dimension Reduction: Semi-Definite Embedding vs. Local Linear Embedding
Li Zhang and Lin Liao

Outline
- Nonlinear Dimension Reduction
- Semi-Definite Embedding
- Local Linear Embedding
- Experiments

Dimension Reduction
- To understand images in terms of their basic modes of variability.
- Unsupervised learning problem: Given $N$ high dimensional input $X_i \in \mathbb{R}^D$, find a faithful one-to-one mapping to $N$ low dimensional output $Y_i \in \mathbb{R}^d$ and $d < D$.
- Methods:
  - Linear methods (PCA, MDS): subspace
  - Nonlinear methods (SDE, LLE): manifold

Semi-Definite Embedding
- Given input $X=(X_1,...,X_N)$ and $k$
- Find the $k$ nearest neighbors for each input $X_i$
- Formulate and solve a corresponding semi-definite programming problem; find optimal Gram matrix of output $K=Y^TY$
- Extract approximately a low dimensional embedding $Y$ from the eigenvectors and eigenvalues of Gram matrix $K$

Semi-Definite Programming
Maximize $C \cdot X$
Subject to
$AX=b$
where $X$ is a vector with size $n^2$, and $A$ is a positive semi-definite $n \times n$ matrix reshaped from $X$

Semi-Definite Programming
Constraints:
- Maintain the distance between neighbors
$|Y_i-Y_j|^2=|X_i-X_j|^2$ for each pair of neighbor $(i,j)$
$K_i+K_j-K_{ij}=G_i+G_j-G_{ij}$ where $K=Y^TY, G=X^T X$
- Constrain the output centered on the origin
$\Sigma Y=0 \quad \Sigma K=0$
- $K$ is positive semidefinite
Semi-Definite Programming

- Objective function
  - Maximize the sum of pairwise squared distance between outputs
    \[ \sum_{ij} |Y_i - Y_j|^2 \quad \text{Tr}(K) \]

- Solve the best K using any SDP solver
  - CSDP (fast, stable)
  - SeDuMi (stable, slow)
  - SDPT3 (new, fastest, not well tested)

Locally Linear Embedding

1. Compute the neighbors of each data point, \( \mathcal{N} \).
2. Compute the weights \( W_{ij} \) that best reconstruct each data point \( X_i \) from its neighbors, minimizing the cost in Equation (1) by constrained linear fit.
3. Compute the vectors \( Y_j \) best reconstructed by the weights \( W_{ij} \), minimizing the quadratic form in Equation (2) by its bottom mzero eigenvectors.

Swiss Roll

LLE on Swiss Roll, varying K

N=800
LLE on Swiss Roll, varying K

Twos

Teapots

LLE on Teapot, varying N

Faces

SDE versus LLE

- Similar idea
  - First, compute neighborhoods in the input space
  - Second, construct a square matrix to characterize local relationship between input data.
  - Finally, compute low-dimension embedding using the eigenvectors of the matrix
SDE versus LLE

- Different performance
  - SDE: good quality, more robust to sparse samples, but optimization is slow and hard to scale to large data set
  - LLE: fast, scalable to large data set, but low quality when samples are sparse, due to locally linear assumption