# Machine Intelligence 2

## 4.4 Locally Linear Embedding

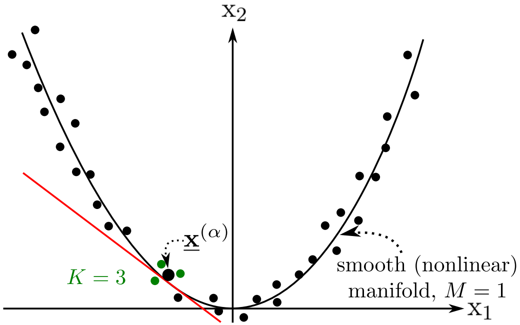
Prof. Dr. Klaus Obermayer

Fachgebiet Neuronale Informationsverarbeitung (NI)

SS 2018

# Locally Linear Embedding

Project the data (locally) into the tangential (linear) space of the data manifold.



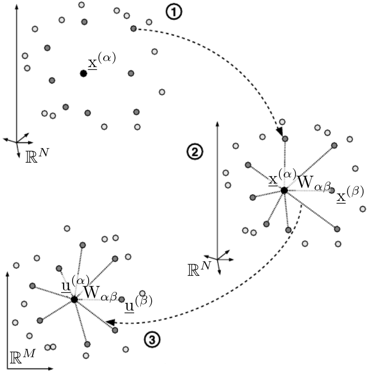

- data points $\underline{\mathbf{x}}^{(\alpha)} \in \mathbb{R}^N$
- embedded data points $\underline{\mathbf{u}}^{(\alpha)} \in \mathbb{R}^M, \quad M < N$

# Locally Linear Embedding

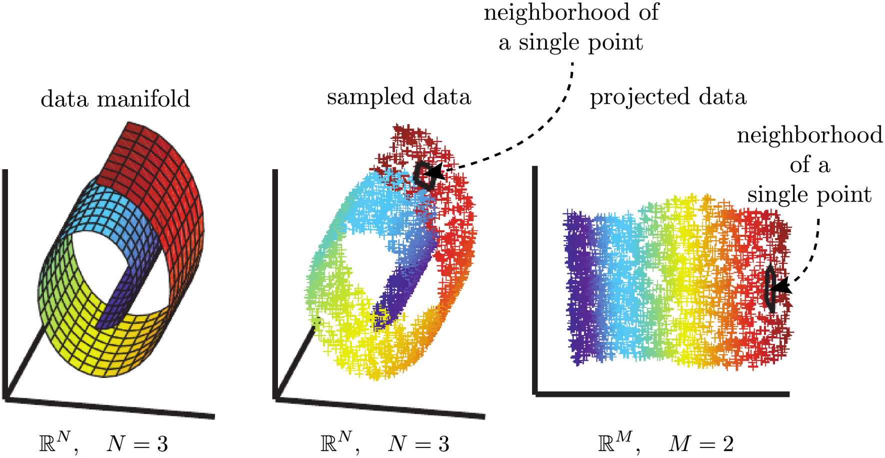
Project the data (locally) into the tangential (linear) space of the data manifold.

For each data point $\underline{\mathbf{x}}^{(\alpha)}$

1. find the $K$ nearest neighbors
2. calculate reconstruction weights $\underline{\underline{\mathbf{W}}}$
   s.t. $\underline{\mathbf{x}}^{(\alpha)} \approx \sum_{\beta \in \mathrm{KNN}(\underline{\mathbf{x}}^{(\alpha)})} \mathrm{W}_{\alpha\beta} \cdot \underline{\mathbf{x}}^{(\beta)}$
3. obtain embedding $\underline{\mathbf{u}}^{(\alpha)} \in \mathbb{R}^M$
   s.t. $\underline{\mathbf{u}}^{(\alpha)} \approx \sum_{\beta \in \mathrm{KNN}(\underline{\mathbf{x}}^{(\alpha)})} \mathrm{W}_{\alpha\beta} \cdot \underline{\mathbf{u}}^{(\beta)}$



# Locally Linear Embedding




*Source: Science; Roweis, Saul 2000, modified*

# Step 1: find $K$ nearest neighbors

choice: Euclidean distance

$$\begin{aligned}
\beta_1^{(\alpha)} &= \arg\min_{\beta} \left| \underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{x}}^{(\beta)} \right| \\
\beta_2^{(\alpha)} &= \arg\min_{\beta \neq \beta_1^{(\alpha)}} \left| \underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{x}}^{(\beta)} \right| \\
&\vdots \\
\beta_K^{(\alpha)} &= \arg\min_{\substack{\beta \neq \beta_k^{(\alpha)}, \\ k=1,\ldots,K-1}} \left| \underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{x}}^{(\beta)} \right| \\
\mathrm{KNN}(\underline{\mathbf{x}}^{(\alpha)}) &= \left\{ \beta_1^{(\alpha)}, \beta_2^{(\alpha)}, \ldots, \beta_K^{(\alpha)} \right\} \qquad \text{(not necessarily unique)}
\end{aligned}$$

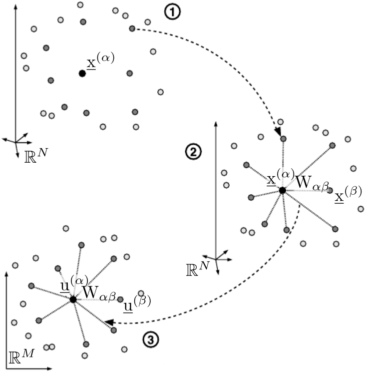
linear data structure (e.g. data matrix): $\mathcal{O}(Np^2)$
k-d tree (balanced search tree): $\mathcal{O}(Np \log p)$

# Locally Linear Embedding

Project the data (locally) into the tangential (linear) space of the data manifold.

For each data point $\underline{\mathbf{x}}^{(\alpha)}$

1. find the $K$ nearest neighbors
2. calculate reconstruction weights $\underline{\mathbf{W}}$ s.t. $\underline{\mathbf{x}}^{(\alpha)} \approx \sum_{\beta \in \text{KNN}(\underline{\mathbf{x}}^{(\alpha)})} \text{W}_{\alpha\beta} \cdot \underline{\mathbf{x}}^{(\beta)}$
3. obtain embedding $\underline{\mathbf{u}}^{(\alpha)} \in \mathbb{R}^M$ s.t. $\underline{\mathbf{u}}^{(\alpha)} \approx \sum_{\beta \in \text{KNN}(\underline{\mathbf{x}}^{(\alpha)})} \text{W}_{\alpha\beta} \cdot \underline{\mathbf{u}}^{(\beta)}$



# Step 2: calculate reconstruction weights

minimize cost function:

$$E(\underline{\mathbf{W}}) = \sum_{\alpha=1}^{p} \underbrace{\left| \underline{\mathbf{x}}^{(\alpha)} - \sum_{\beta=1}^{p} \mathrm{W}_{\alpha\beta} \underline{\mathbf{x}}^{(\beta)} \right|^2}_{\substack{\text{reconstruct } \underline{\mathbf{x}}^{(\alpha)} \text{ by its} \\ K \text{ nearest neighbors only}}} \stackrel{!}{=} \min_{\underline{\mathbf{W}}} \quad \text{s.t.} \quad \begin{array}{l} \mathrm{W}_{\alpha\beta} = 0 \text{ if } \beta \notin \mathrm{KNN}(\underline{\mathbf{x}}^{(\alpha)}), \\ \displaystyle\sum_{\beta=1}^{p} \mathrm{W}_{\alpha\beta} = 1 \end{array}$$

reconstruction weight matrix $\underline{\mathbf{W}} \in \mathbb{R}^{p,p}$:

- sparse: (up to) $K$ nonzero elements per row
- not symmetric: nearest neighbors of a data point can have closer neighbors

optimization problem:

- decomposes into $p$ non-interacting parts (one per data pt. reconstruction)
- equality constraint for invariances (see next slide)

# Step 2: calculate reconstruction weights

minimize cost function:

$$E(\underline{\mathbf{W}}) = \sum_{\alpha=1}^{p} \underbrace{\left| \underline{\mathbf{x}}^{(\alpha)} - \sum_{\beta=1}^{p} \mathrm{W}_{\alpha\beta} \underline{\mathbf{x}}^{(\beta)} \right|^2}_{\substack{\text{reconstruct } \underline{\mathbf{x}}^{(\alpha)} \text{ by its} \\ K \text{ nearest neighbors only}}} \overset{!}{=} \min_{\underline{\mathbf{W}}} \quad \text{s.t.} \quad \begin{array}{l} \mathrm{W}_{\alpha\beta} = 0 \text{ if } \beta \notin \mathrm{KNN}(\underline{\mathbf{x}}^{(\alpha)}), \\ \sum_{\beta=1}^{p} \mathrm{W}_{\alpha\beta} = 1 \end{array}$$

optimal weights are invariant to:

- rotation $\underline{\mathbf{R}}$ : $\overbrace{E}^{\text{cost function}} \left[ \underline{\mathbf{R}} \cdot \underline{\mathbf{x}}^{(1)}, \ldots, \underline{\mathbf{R}} \cdot \underline{\mathbf{x}}^{(p)} \right] \overset{\text{orthog. } \underline{\mathbf{R}}}{=} E \left[ \underline{\mathbf{x}}^{(1)}, \ldots, \underline{\mathbf{x}}^{(p)} \right]$
- scaling $\gamma > 0$ : $E \left[ \gamma \underline{\mathbf{x}}^{(1)}, \ldots, \gamma \underline{\mathbf{x}}^{(p)} \right] = \gamma^2 E \left[ \underline{\mathbf{x}}^{(1)}, \ldots, \underline{\mathbf{x}}^{(p)} \right]$
- translation $\Delta \underline{\mathbf{x}}$ : $E \left[ \underline{\mathbf{x}}^{(1)} + \Delta \underline{\mathbf{x}}, \ldots, \underline{\mathbf{x}}^{(p)} + \Delta \underline{\mathbf{x}} \right] \overset{\sum_\beta \mathrm{W}_{\alpha\beta} = 1}{=} E \left[ \underline{\mathbf{x}}^{(1)}, \ldots, \underline{\mathbf{x}}^{(p)} \right]$

# Step 2: calculate reconstruction weights

minimize cost function:

$$E(\underline{\mathbf{W}}) = \sum_{\alpha=1}^{p} \underbrace{\left| \underline{\mathbf{x}}^{(\alpha)} - \sum_{\beta=1}^{p} \mathrm{W}_{\alpha\beta} \underline{\mathbf{x}}^{(\beta)} \right|^2}_{\substack{\text{reconstruct } \underline{\mathbf{x}}^{(\alpha)} \text{ by its} \\ K \text{ nearest neighbors only}}} \stackrel{!}{=} \min_{\underline{\mathbf{W}}} \quad \text{s.t.} \quad \begin{array}{l} \mathrm{W}_{\alpha\beta} = 0 \text{ if } \beta \notin \mathrm{KNN}(\underline{\mathbf{x}}^{(\alpha)}), \\ \sum_{\beta=1}^{p} \mathrm{W}_{\alpha\beta} = 1 \end{array}$$

for each data point $\underline{\mathbf{x}}^{(\alpha)}$ (result from applying Lagrange multiplier method):

- local "covariance" matrix (symmetric & positive semidefinite) $\underline{\mathbf{C}}^{(\alpha)} \in \mathbb{R}^{K,K}$ :
$$\mathrm{C}_{jk}^{(\alpha)} = \left(\underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{x}}^{(\beta_j^{(\alpha)})}\right)^T \left(\underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{x}}^{(\beta_k^{(\alpha)})}\right)$$
- solve linear system $\underline{\mathbf{C}}^{(\alpha)} \underline{\widetilde{\mathbf{w}}}^{(\alpha)} = (1, \ldots, 1)^T$
- rescale weights: $\mathrm{W}_{\alpha\beta_j^{(\alpha)}} = \widetilde{\mathrm{w}}_j^{(\alpha)} / \sum_{k=1}^{K} \widetilde{\mathrm{w}}_k^{(\alpha)}$ to fulfill sum-to-one constraint

$\Rightarrow$ $\underline{\mathbf{W}}$ contains the optimal weights with $\mathrm{W}_{\alpha\beta} = 0$ for $\beta \notin \mathrm{KNN}(\underline{\mathbf{x}}^{(\alpha)})$

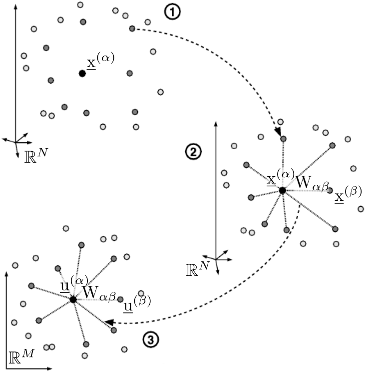
$\Rightarrow$ $p$ dense $K$-dim. linear systems to be constructed & solved: $\quad \mathcal{O}(pNK^2 + pK^3)$

# Locally Linear Embedding

Project the data (locally) into the tangential (linear) space of the data manifold.

For each data point $\underline{\mathbf{x}}^{(\alpha)}$

1. find the $K$ nearest neighbors
2. calculate reconstruction weights $\underline{\underline{\mathbf{W}}}$
   s.t. $\underline{\mathbf{x}}^{(\alpha)} \approx \sum_{\beta \in \mathrm{KNN}(\underline{\mathbf{x}}^{(\alpha)})} \mathrm{W}_{\alpha\beta} \cdot \underline{\mathbf{x}}^{(\beta)}$
3. obtain embedding $\underline{\mathbf{u}}^{(\alpha)} \in \mathbb{R}^M$
   s.t. $\underline{\mathbf{u}}^{(\alpha)} \approx \sum_{\beta \in \mathrm{KNN}(\underline{\mathbf{x}}^{(\alpha)})} \mathrm{W}_{\alpha\beta} \cdot \underline{\mathbf{u}}^{(\beta)}$



# Step 3: obtain embedding coordinates

For any $M$-dimensional manifold there exist linear mappings of each local "patch" onto $M$-dim. coordinates in a linear space (differential geometry)

- linear mapping: rotation, scaling, translation
- weights $\underline{\mathbf{W}}$ can be used to optimally reconstruct the data points in the lower-dimensional embedding space

idea:

- cut $N$-d manifold into small patches
- "glue" them together in $M$-d using only rotation, scaling, translation for each patch

$\rightsquigarrow$ use cost function from before, keep weights fixed, vary coordinates

# Step 3: obtain embedding coordinates

given $M \ll N$ and $\underline{\mathbf{W}}$ : find optimal coordinates $\underbrace{\underline{\mathbf{u}}^{(1)}, \ldots, \underline{\mathbf{u}}^{(p)}}_{=\underline{\mathbf{U}} \in \mathbb{R}^{M,p}} \in \mathbb{R}^M$

cost function (interactions between neighbored patches $\leadsto$ non-decomposable):

$$F(\underline{\mathbf{U}}) = \sum_{\alpha=1}^{p} \left| \underline{\mathbf{u}}^{(\alpha)} - \sum_{\beta=1}^{p} \mathrm{W}_{\alpha\beta} \underline{\mathbf{u}}^{(\beta)} \right|^2$$

equivalent quadratic form:

$$F(\underline{\mathbf{U}}) = \sum_{\alpha,\beta=1}^{p} g_{\alpha\beta} (\underline{\mathbf{u}}^{(\alpha)})^T \underline{\mathbf{u}}^{(\beta)}$$

where $g_{\alpha\beta}$ = **see blackboard** derivation here

$$= \delta_{\alpha\beta} - \mathrm{W}_{\alpha\beta} - \mathrm{W}_{\beta\alpha} + \sum_{\gamma=1}^{p} \mathrm{W}_{\gamma\alpha} \mathrm{W}_{\gamma\beta}$$

$\underline{\mathbf{G}} = \left(\underline{\mathbf{I}} - \underline{\mathbf{W}}^T\right)\left(\underline{\mathbf{I}} - \underline{\mathbf{W}}\right) = \left\{g_{\alpha\beta}\right\} \in \mathbb{R}^{p,p}$ is symmetric and positive semidefinite

# Step 3: obtain embedding coordinates

minimize cost function:

$$F(\underline{\mathbf{U}}) = \sum_{\alpha,\beta=1}^{p} g_{\alpha\beta} (\underline{\mathbf{u}}^{(\alpha)})^T \underline{\mathbf{u}}^{(\beta)}$$

s.t. $\sum_{\alpha=1}^{p} \underline{\mathbf{u}}^{(\alpha)} = 0,$ (remove translation freedom)

$\frac{1}{p} \sum_{\alpha=1}^{p} \underline{\mathbf{u}}^{(\alpha)} (\underline{\mathbf{u}}^{(\alpha)})^T = \underline{\mathbf{I}}$ (prevent trivial solution $\underline{\mathbf{u}}^{(\alpha)} = \underline{\mathbf{0}}$)

- $\leadsto$ w.l.o.g. as $F(\underline{\mathbf{U}})$ invariant to rotation, scaling, translation
- $\leadsto$ implies that reconstruction errors for different coordinates are measured on the same scale

# Step 3: obtain embedding coordinates

solution (via Lagrange multiplier method):

compute the $M+1$ eigenvectors of $\underline{\mathbf{G}}$ with the lowest eigenvalues but discard the eigenvector $\underline{\mathbf{e}}_p = \frac{1}{p}(1, \ldots, 1)^T$ with eigenvalue 0 (translation)

$$\underline{\mathbf{U}} = \begin{pmatrix} \underline{\mathbf{e}}_{p-M}^T \\ \vdots \\ \underline{\mathbf{e}}_{p-1}^T \end{pmatrix} = \left( \underline{\mathbf{u}}^{(1)}, \ldots, \underline{\mathbf{u}}^{(p)} \right) \quad \in \mathbb{R}^{M,p}$$

intuition [eigenvalue $\lambda_i$ and -vector $\underline{\mathbf{e}}_i = (u_i^{(1)}, \ldots, u_i^{(p)})^T = \underline{\mathbf{u}}_{(i)}$ of $\underline{\mathbf{G}}$]:

$$F(\underline{\mathbf{U}}) = \sum_{\alpha,\beta=1}^{p} g_{\alpha\beta} \underbrace{(\underline{\mathbf{u}}^{(\alpha)})^T \underline{\mathbf{u}}^{(\beta)}}_{=\sum_{i=1}^{M} u_i^{(\alpha)} u_i^{(\beta)}} = \sum_{i=1}^{M} \underbrace{\sum_{\alpha,\beta=1}^{p} g_{\alpha\beta} u_i^{(\alpha)} u_i^{(\beta)}}_{=\underline{\mathbf{u}}_{(i)}^T \underline{\mathbf{G}}\, \underline{\mathbf{u}}_{(i)}} = \sum_{i=1}^{M} \underline{\mathbf{u}}_{(i)}^T \underline{\mathbf{u}}_{(i)} \lambda_i$$

- implement $\underline{\mathbf{W}}$ as sparse matrix (at most $K \cdot p$ non-zero elements)
- sparse eigenvalue solver with: $\underline{\mathbf{v}} \mapsto \underline{\mathbf{G}} \cdot \underline{\mathbf{v}} = \left( \underline{\mathbf{I}} - \underline{\mathbf{W}}^T \right) \left[ \left( \underline{\mathbf{I}} - \underline{\mathbf{W}} \right) \underline{\mathbf{v}} \right]$

# Summary of the LLE algorithm

parameters: $K, M$

1. find $K$ nearest neighbors $\mathrm{KNN}(\underline{\mathbf{x}}^{(\alpha)}) = \left\{\beta_1^{(\alpha)}, \ldots, \beta_K^{(\alpha)}\right\} \; \forall \alpha = 1, \ldots, p$
2. calculate (locally invariant) reconstruction weights $\underline{\mathbf{W}} \in \mathbb{R}^{p,p}$ (sparse):

   $\underline{\mathbf{C}}^{(\alpha)} \underline{\widetilde{\mathbf{w}}}^{(\alpha)} = (1, \ldots, 1)^T, \quad \forall \alpha = 1, \ldots, p,$ with dense $\underline{\mathbf{C}}^{(\alpha)} \in \mathbb{R}^{K,K}$

   $$\mathrm{W}_{\alpha \beta_j^{(\alpha)}} = \frac{\widetilde{\mathrm{w}}_j^{(\alpha)}}{\sum_{k=1}^K \widetilde{\mathrm{w}}_k^{(\alpha)}}, \; \forall j = 1, \ldots, K, \qquad \mathrm{W}_{\alpha\beta} = 0, \; \forall \beta \notin \mathrm{KNN}(\underline{\mathbf{x}}^{(\alpha)})$$
3. calculate the embedding coordinates $\underline{\mathbf{U}}$:

   compute the $M+1$ eigenvectors $\left(\underline{\mathbf{e}}_p, \ldots, \underline{\mathbf{e}}_{p-M}\right)$ of (sparse) $\underline{\mathbf{G}} \in \mathbb{R}^{p,p}$ with the smallest eigenvalues and skip that of the smallest eigenval.

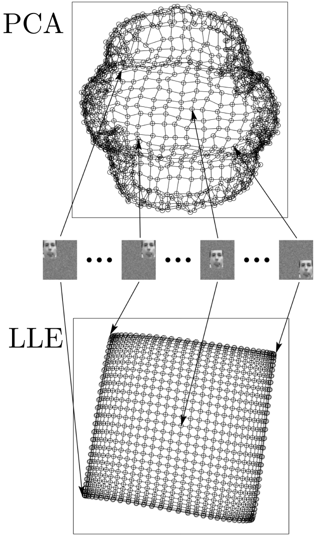
   $$g_{\alpha\beta} = \delta_{\alpha\beta} - \mathrm{W}_{\alpha\beta} - \mathrm{W}_{\beta\alpha} + \sum_{\gamma=1}^{p} \mathrm{W}_{\gamma\alpha} \mathrm{W}_{\gamma\beta}$$

   $$\underline{\mathbf{G}} \cdot \underline{\mathbf{e}}_j = \lambda_j \underline{\mathbf{e}}_j \qquad \underline{\mathbf{U}} = \begin{pmatrix} \underline{\mathbf{e}}_{p-M}^T \\ \vdots \\ \underline{\mathbf{e}}_{p-1}^T \end{pmatrix} = \left(\underline{\mathbf{u}}^{(1)}, \ldots, \underline{\mathbf{u}}^{(p)}\right)$$

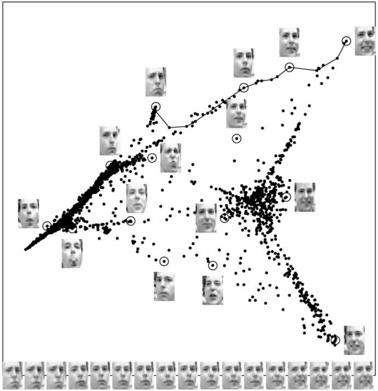
# Example 1

- Images of a single face translated across a two-dimensional background of noise
- PCA fails to preserve the neighborhood structure of nearby images
- LLE maps the images with corner faces to the corners of its two dimensional embedding ($M = 2$)

*Source: An Introduction to Locally Linear Embedding; Saul, Roweis 2001*



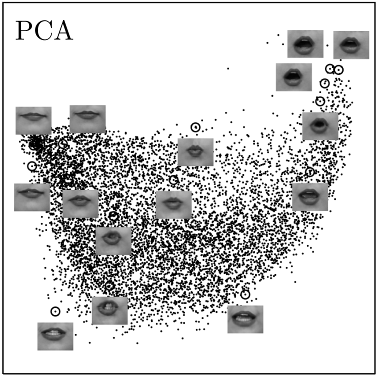
# Example 2

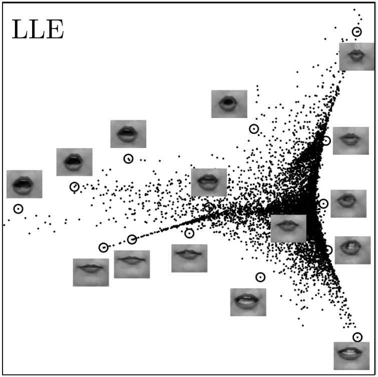


*Source: Science; Roweis, Saul 2000*

- Images of faces mapped into an embedding space with $M = 2$.
- Bottom: Images corresponding to points along the solid line shown on the top-right.

# Example 3




*Source: An Introduction to Locally Linear Embedding; Saul, Roweis 2001*

- Images of lips mapped into an embedding space with $M = 2$.
- Differences between the two embeddings indicate the presence of nonlinear structure in the data.

# Remarks

- efficient & robust algorithm
- parameters: number $K$ of neighbors, embedding dimension $M$
- convex optimization problem, standard (sparse) linear algebra methods
- for $K > N$ regularization is required (singular covariance matrix $\underline{\mathbf{C}}^{(\alpha)}$)

  $\underline{\mathbf{C}}^{(\alpha)} \leftarrow \underline{\mathbf{C}}^{(\alpha)} + \varepsilon \underline{\mathbf{I}}$ (alternative: *modified* LLE, which takes $\varepsilon \rightarrow 0$)
- extendible to (non-Euclidean) pairwise distances $d_{\alpha\alpha'}$ in $\underline{\mathbf{C}}^{(\alpha)}$
- LLE is designed for *one* manifold, multiple separate (w.r.t. KNN) manifolds yield unrelated embedding coordinates between non-connected data point subsets
- alternative methods available (e.g. Laplacian eigenmaps, t-stochastic neighbor embedding, isomap, Kernel PCA)

# Supplemental Material

Derivation of $g_{\alpha\beta}$ in cost function for finding optimal coordinates:

$$
\begin{aligned}
&\sum_{\alpha=1}^{p} \left( \underline{\mathbf{u}}^{(\alpha)} - \sum_{\beta=1}^{p} W_{\alpha\beta} \underline{\mathbf{u}}^{(\beta)} \right)^2 \\
&= \sum_{\alpha=1}^{p} \left[ (\underline{\mathbf{u}}^{(\alpha)})^2 - 2 \sum_{\beta=1}^{p} W_{\alpha\beta} (\underline{\mathbf{u}}^{(\alpha)})^T \underline{\mathbf{u}}^{(\beta)} + \sum_{\beta=1,\gamma=1}^{p} W_{\alpha\beta} W_{\alpha\gamma} (\underline{\mathbf{u}}^{(\beta)})^T \underline{\mathbf{u}}^{(\gamma)} \right] \\
&= \sum_{\alpha=1}^{p} \left[ (\underline{\mathbf{u}}^{(\alpha)})^2 - \sum_{\beta=1}^{p} W_{\alpha\beta} (\underline{\mathbf{u}}^{(\alpha)})^T \underline{\mathbf{u}}^{(\beta)} - \sum_{\beta=1}^{p} W_{\beta\alpha} (\underline{\mathbf{u}}^{(\beta)})^T \underline{\mathbf{u}}^{(\alpha)} \right. \\
&\quad \left. + \sum_{\beta=1}^{p} \sum_{\gamma=1}^{p} \left( W_{\gamma\alpha} W_{\gamma\beta} \right) (\underline{\mathbf{u}}^{(\alpha)})^T \underline{\mathbf{u}}^{(\beta)} \right] \\
&= \sum_{\alpha,\beta=1}^{p} \underbrace{\left\{ \delta_{\alpha\beta} - W_{\alpha\beta} - W_{\beta\alpha} + \sum_{\gamma=1}^{p} W_{\gamma\alpha} W_{\gamma\beta} \right\}}_{= g_{\alpha\beta}} (\underline{\mathbf{u}}^{(\alpha)})^T \underline{\mathbf{u}}^{(\beta)}
\end{aligned}
$$