MAT 280: Harmonic Analysis on Graphs & Networks
Lecture 14: Dimension Reduction via PCA, Laplacian Eigenmaps, & Diffusion Maps

Naoki Saito

Department of Mathematics
University of California, Davis

November 12, 2019

Outline

Outline

Dimensionality Reduction/Low-Dimensional Embedding

- Dimensionality reduction, *if properly done*, is quite useful and effective for many data analysis tasks.
- Many techniques, proposals, algorithms exist.
- In this lecture, we only discuss:
  - Classical Multidimensional Scaling (given data vectors) ≡ PCA
  - Laplacian Eigenmap
  - Diffusion Map
- CMDS/PCA is a linear technique whereas LE/DM are *nonlinear*.

Notation

- Let X be the training data matrix, $X := \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{\mathrm{tr}}}\right) \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$.
- Let $\widetilde{X} := X(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/N_{\mathrm{tr}})$, i.e., the *centered* data matrix (the mean of the column vectors $\overline{\boldsymbol{x}}$ is subtracted from each column vector).
- Let $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ be a low-dimensional embedding map.
- Let $\Psi(X) := \left(\Psi(\boldsymbol{x}_1), \ldots, \Psi(\boldsymbol{x}_{N_{\mathrm{tr}}})\right) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}$.

Dimensionality Reduction/Low-Dimensional Embedding

- Dimensionality reduction, *if properly done*, is quite useful and effective for many data analysis tasks.
- Many techniques, proposals, algorithms exist.
- In this lecture, we only discuss:
  - Classical Multidimensional Scaling (given data vectors) ≡ PCA
  - Laplacian Eigenmap
  - Diffusion Map
- CMDS/PCA is a linear technique whereas LE/DM are *nonlinear*.

Notation

- Let X be the training data matrix, $X := \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{\text{tr}}}\right) \in \mathbb{R}^{d \times N_{\text{tr}}}$.
- Let $\widetilde{X} := X(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/N_{\text{tr}})$, i.e., the *centered* data matrix (the mean of the column vectors $\overline{\boldsymbol{x}}$ is subtracted from each column vector).
- Let $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ be a low-dimensional embedding map.
- Let $\Psi(X) := \left(\Psi(\boldsymbol{x}_1), \ldots, \Psi(\boldsymbol{x}_{N_{\text{tr}}})\right) \in \mathbb{R}^{s \times N_{\text{tr}}}$.

Dimensionality Reduction/Low-Dimensional Embedding

- Dimensionality reduction, *if properly done*, is quite useful and effective for many data analysis tasks.
- Many techniques, proposals, algorithms exist.
- In this lecture, we only discuss:
    - Classical Multidimensional Scaling (given data vectors) ≡ PCA
    - Laplacian Eigenmap
    - Diffusion Map
- CMDS/PCA is a linear technique whereas LE/DM are *nonlinear*.

Notation

- Let X be the training data matrix, $X := \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{\mathrm{tr}}}\right) \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$.
- Let $\widetilde{X} := X(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/N_{\mathrm{tr}})$, i.e., the *centered* data matrix (the mean of the column vectors $\overline{\boldsymbol{x}}$ is subtracted from each column vector).
- Let $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ be a low-dimensional embedding map.
- Let $\Psi(X) := \left(\Psi(\boldsymbol{x}_1), \ldots, \Psi(\boldsymbol{x}_{N_{\mathrm{tr}}})\right) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}$.

Dimensionality Reduction/Low-Dimensional Embedding

- Dimensionality reduction, *if properly done*, is quite useful and effective for many data analysis tasks.
- Many techniques, proposals, algorithms exist.
- In this lecture, we only discuss:
  - Classical Multidimensional Scaling (given data vectors) ≡ PCA
  - Laplacian Eigenmap
  - Diffusion Map
- CMDS/PCA is a linear technique whereas LE/DM are *nonlinear*.

Notation

- Let X be the training data matrix, $X := \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{\text{tr}}}\right) \in \mathbb{R}^{d \times N_{\text{tr}}}$.
- Let $\widetilde{X} := X(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/N_{\text{tr}})$, i.e., the *centered* data matrix (the mean of the column vectors $\overline{\boldsymbol{x}}$ is subtracted from each column vector).
- Let $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ be a low-dimensional embedding map.
- Let $\Psi(X) := \left(\Psi(\boldsymbol{x}_1), \ldots, \Psi(\boldsymbol{x}_{N_{\text{tr}}})\right) \in \mathbb{R}^{s \times N_{\text{tr}}}$.

Dimensionality Reduction/Low-Dimensional Embedding

- Dimensionality reduction, *if properly done*, is quite useful and effective for many data analysis tasks.
- Many techniques, proposals, algorithms exist.
- In this lecture, we only discuss:
  - Classical Multidimensional Scaling (given data vectors) $\equiv$ PCA
  - Laplacian Eigenmap
  - Diffusion Map
- CMDS/PCA is a linear technique whereas LE/DM are *nonlinear*.

Notation

- Let X be the training data matrix, $X := \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{\mathrm{tr}}}\right) \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$.
- Let $\widetilde{X} := X(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/N_{\mathrm{tr}})$, i.e., the *centered* data matrix (the mean of the column vectors $\overline{\boldsymbol{x}}$ is subtracted from each column vector).
- Let $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ be a low-dimensional embedding map.
- Let $\Psi(X) := \left(\Psi(\boldsymbol{x}_1), \ldots, \Psi(\boldsymbol{x}_{N_{\mathrm{tr}}})\right) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}$.

Dimensionality Reduction/Low-Dimensional Embedding

- Dimensionality reduction, *if properly done*, is quite useful and effective for many data analysis tasks.
- Many techniques, proposals, algorithms exist.
- In this lecture, we only discuss:
  - Classical Multidimensional Scaling (given data vectors) ≡ PCA
  - Laplacian Eigenmap
  - Diffusion Map
- CMDS/PCA is a linear technique whereas LE/DM are *nonlinear*.

Notation

- Let X be the training data matrix, $X := \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{\mathrm{tr}}}\right) \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$.
- Let $\widetilde{X} := X(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/N_{\mathrm{tr}})$, i.e., the *centered* data matrix (the mean of the column vectors $\overline{\boldsymbol{x}}$ is subtracted from each column vector).
- Let $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ be a low-dimensional embedding map.
- Let $\Psi(X) := \left(\Psi(\boldsymbol{x}_1), \ldots, \Psi(\boldsymbol{x}_{N_{\mathrm{tr}}})\right) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}$.

Dimensionality Reduction/Low-Dimensional Embedding

- Dimensionality reduction, *if properly done*, is quite useful and effective for many data analysis tasks.
- Many techniques, proposals, algorithms exist.
- In this lecture, we only discuss:
  - Classical Multidimensional Scaling (given data vectors) ≡ PCA
  - Laplacian Eigenmap
  - Diffusion Map
- CMDS/PCA is a linear technique whereas LE/DM are *nonlinear*.

Notation

- Let X be the training data matrix, $X := \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{\mathrm{tr}}}\right) \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$.
- Let $\widetilde{X} := X(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/N_{\mathrm{tr}})$, i.e., the *centered* data matrix (the mean of the column vectors $\overline{\boldsymbol{x}}$ is subtracted from each column vector).
- Let $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ be a low-dimensional embedding map.
- Let $\Psi(X) := \left(\Psi(\boldsymbol{x}_1), \ldots, \Psi(\boldsymbol{x}_{N_{\mathrm{tr}}})\right) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}$.

Dimensionality Reduction/Low-Dimensional Embedding

- Dimensionality reduction, *if properly done*, is quite useful and effective for many data analysis tasks.
- Many techniques, proposals, algorithms exist.
- In this lecture, we only discuss:
  - Classical Multidimensional Scaling (given data vectors) ≡ PCA
  - Laplacian Eigenmap
  - Diffusion Map
- CMDS/PCA is a linear technique whereas LE/DM are *nonlinear*.

Notation

- Let X be the training data matrix, $X := \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{\mathrm{tr}}}\right) \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$.
- Let $\widetilde{X} := X(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/N_{\mathrm{tr}})$, i.e., the *centered* data matrix (the mean of the column vectors $\overline{\boldsymbol{x}}$ is subtracted from each column vector).
- Let $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ be a low-dimensional embedding map.
- Let $\Psi(X) := \left(\Psi(\boldsymbol{x}_1), \ldots, \Psi(\boldsymbol{x}_{N_{\mathrm{tr}}})\right) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}$.

Dimensionality Reduction/Low-Dimensional Embedding

- Dimensionality reduction, *if properly done*, is quite useful and effective for many data analysis tasks.
- Many techniques, proposals, algorithms exist.
- In this lecture, we only discuss:
  - Classical Multidimensional Scaling (given data vectors) ≡ PCA
  - Laplacian Eigenmap
  - Diffusion Map
- CMDS/PCA is a linear technique whereas LE/DM are *nonlinear*.

Notation

- Let X be the training data matrix, $X := \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{\mathrm{tr}}}\right) \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$.
- Let $\widetilde{X} := X(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/N_{\mathrm{tr}})$, i.e., the *centered* data matrix (the mean of the column vectors $\overline{\boldsymbol{x}}$ is subtracted from each column vector).
- Let $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ be a low-dimensional embedding map.
- Let $\Psi(X) := \left(\Psi(\boldsymbol{x}_1), \ldots, \Psi(\boldsymbol{x}_{N_{\mathrm{tr}}})\right) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}$.

Dimensionality Reduction/Low-Dimensional Embedding

- Dimensionality reduction, *if properly done*, is quite useful and effective for many data analysis tasks.
- Many techniques, proposals, algorithms exist.
- In this lecture, we only discuss:
  - Classical Multidimensional Scaling (given data vectors) ≡ PCA
  - Laplacian Eigenmap
  - Diffusion Map
- CMDS/PCA is a linear technique whereas LE/DM are *nonlinear*.

Notation

- Let X be the training data matrix, $X := \left(\boldsymbol{x}_1, \dots, \boldsymbol{x}_{N_{\mathrm{tr}}}\right) \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$.
- Let $\widetilde{X} := X(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/N_{\mathrm{tr}})$, i.e., the *centered* data matrix (the mean of the column vectors $\overline{\boldsymbol{x}}$ is subtracted from each column vector).
- Let $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ be a low-dimensional embedding map.
- Let $\Psi(X) := \left(\Psi(\boldsymbol{x}_1), \dots, \Psi(\boldsymbol{x}_{N_{\mathrm{tr}}})\right) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}$.

Dimensionality Reduction/Low-Dimensional Embedding

- Dimensionality reduction, *if properly done*, is quite useful and effective for many data analysis tasks.
- Many techniques, proposals, algorithms exist.
- In this lecture, we only discuss:
  - Classical Multidimensional Scaling (given data vectors) ≡ PCA
  - Laplacian Eigenmap
  - Diffusion Map
- CMDS/PCA is a linear technique whereas LE/DM are *nonlinear*.

Notation

- Let X be the training data matrix, $X := \left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{\mathrm{tr}}}\right) \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$.
- Let $\widetilde{X} := X(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/N_{\mathrm{tr}})$, i.e., the *centered* data matrix (the mean of the column vectors $\overline{\boldsymbol{x}}$ is subtracted from each column vector).
- Let $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ be a low-dimensional embedding map.
- Let $\Psi(X) := \left(\Psi(\boldsymbol{x}_1), \ldots, \Psi(\boldsymbol{x}_{N_{\mathrm{tr}}})\right) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}$.

Outline

Classical (Multidimensional) Scaling and PCA (Review)

- Define the *similarity* between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ by the centered correlation

$$\alpha(\boldsymbol{x}_i, \boldsymbol{x}_j) := (\boldsymbol{x}_i - \overline{\boldsymbol{x}})^{\mathsf{T}}(\boldsymbol{x}_j - \overline{\boldsymbol{x}}).$$

- Then, the classical scaling seeks the low-dimensional representation that preserves the pairwise similarities in X as well as possible by minimizing

$$J_{\mathrm{CS}}(\Psi) := \sum_{i,j}(\alpha(\boldsymbol{x}_i, \boldsymbol{x}_j) - \alpha(\Psi(\boldsymbol{x}_i), \Psi(\boldsymbol{x}_j)))^2 = \left\| \widetilde{X}^{\mathsf{T}}\widetilde{X} - \Psi(\widetilde{X})^{\mathsf{T}}\Psi(\widetilde{X}) \right\|_F^2.$$

- We can find this map using the *SVD* of $\widetilde{X} = U\Sigma V^{\mathsf{T}}$ as

$$\Psi(\widetilde{X}) = U_s^{\mathsf{T}}\widetilde{X} = \Sigma_s V_s^{\mathsf{T}},$$

  which is *exactly the same as* using the first s components of *PCA*!
- A drawback: too *global* and not incorporating *local* geometry

Classical (Multidimensional) Scaling and PCA (Review)

- Define the *similarity* between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ by the centered correlation

$$\alpha(\boldsymbol{x}_i, \boldsymbol{x}_j) := (\boldsymbol{x}_i - \overline{\boldsymbol{x}})^\top (\boldsymbol{x}_j - \overline{\boldsymbol{x}}).$$

- Then, the classical scaling seeks the low-dimensional representation that preserves the pairwise similarities in X as well as possible by minimizing

$$J_{\mathrm{CS}}(\Psi) := \sum_{i,j} (\alpha(\boldsymbol{x}_i, \boldsymbol{x}_j) - \alpha(\Psi(\boldsymbol{x}_i), \Psi(\boldsymbol{x}_j)))^2 = \left\| \widetilde{X}^\top \widetilde{X} - \Psi(\widetilde{X})^\top \Psi(\widetilde{X}) \right\|_F^2.$$

- We can find this map using the *SVD* of $\widetilde{X} = U\Sigma V^\top$ as

$$\Psi(\widetilde{X}) = U_s^\top \widetilde{X} = \Sigma_s V_s^\top,$$

which is *exactly the same as* using the first s components of *PCA*!
- A drawback: too *global* and not incorporating *local* geometry

Classical (Multidimensional) Scaling and PCA (Review)

- Define the *similarity* between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ by the centered correlation

$$\alpha(\boldsymbol{x}_i, \boldsymbol{x}_j) := (\boldsymbol{x}_i - \overline{\boldsymbol{x}})^\mathsf{T} (\boldsymbol{x}_j - \overline{\boldsymbol{x}}).$$

- Then, the classical scaling seeks the low-dimensional representation that preserves the pairwise similarities in X as well as possible by minimizing

$$J_{\mathrm{CS}}(\Psi) := \sum_{i,j} (\alpha(\boldsymbol{x}_i, \boldsymbol{x}_j) - \alpha(\Psi(\boldsymbol{x}_i), \Psi(\boldsymbol{x}_j)))^2 = \left\| \widetilde{X}^\mathsf{T} \widetilde{X} - \Psi(\widetilde{X})^\mathsf{T} \Psi(\widetilde{X}) \right\|_F^2 .$$

- We can find this map using the *SVD* of $\widetilde{X} = U \Sigma V^\mathsf{T}$ as

$$\Psi(\widetilde{X}) = U_s^\mathsf{T} \widetilde{X} = \Sigma_s V_s^\mathsf{T},$$

  which is *exactly the same as* using the first s components of *PCA*!

- A drawback: too *global* and not incorporating *local* geometry

Classical (Multidimensional) Scaling and PCA (Review)

- Define the *similarity* between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ by the centered correlation

$$\alpha(\boldsymbol{x}_i, \boldsymbol{x}_j) := (\boldsymbol{x}_i - \overline{\boldsymbol{x}})^{\mathsf{T}} (\boldsymbol{x}_j - \overline{\boldsymbol{x}}).$$

- Then, the classical scaling seeks the low-dimensional representation that preserves the pairwise similarities in X as well as possible by minimizing

$$J_{\mathrm{CS}}(\Psi) := \sum_{i,j} (\alpha(\boldsymbol{x}_i, \boldsymbol{x}_j) - \alpha(\Psi(\boldsymbol{x}_i), \Psi(\boldsymbol{x}_j)))^2 = \left\| \widetilde{X}^{\mathsf{T}} \widetilde{X} - \Psi(\widetilde{X})^{\mathsf{T}} \Psi(\widetilde{X}) \right\|_F^2 .$$

- We can find this map using the *SVD* of $\widetilde{X} = U \Sigma V^{\mathsf{T}}$ as

$$\Psi(\widetilde{X}) = U_s^{\mathsf{T}} \widetilde{X} = \Sigma_s V_s^{\mathsf{T}},$$

  which is *exactly the same as* using the first s components of *PCA*!
- A drawback: too *global* and not incorporating *local* geometry

Outline

Laplacian Eigenmaps (Belkin & Niyogi, 2001–3)

- Incorporating *local* geometric information in $\mathbb{R}^d$ for the embedding
- Define the *proximity* weight $w(\boldsymbol{x}_i, \boldsymbol{x}_j)$, e.g., $w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j) := e^{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/\epsilon^2}$.
- Now, seek $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ that minimizes the following

$$J_{\mathrm{LE}}(\Psi) := \sum_{i,j} \|\Psi(\boldsymbol{x}_i) - \Psi(\boldsymbol{x}_j)\|^2 w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

- This leads to the following optimization problem:

$$\min_{\Psi(X) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}} \operatorname{tr}\left(\Psi(X) L \Psi(X)^{\mathsf{T}}\right) \quad \text{subject to } \Psi(X) D \Psi(X)^{\mathsf{T}} = I,$$

where the matrices are defined as

$$A := \left(a_{ij} = w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j)\right), \quad D := \operatorname{diag}\left(\sum_j a_{1j}, \ldots, \sum_j a_{N_{\mathrm{tr}}j}\right).$$

The matrix $L := D - A$ is the (unnormalized) *graph Laplacian*, of course.

Laplacian Eigenmaps (Belkin & Niyogi, 2001–3)

- Incorporating *local* geometric information in $\mathbb{R}^d$ for the embedding
- Define the *proximity* weight $w(\boldsymbol{x}_i, \boldsymbol{x}_j)$, e.g., $w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j) := \mathrm{e}^{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/\epsilon^2}$.
- Now, seek $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ that minimizes the following

$$J_{\mathrm{LE}}(\Psi) := \sum_{i,j} \|\Psi(\boldsymbol{x}_i) - \Psi(\boldsymbol{x}_j)\|^2 w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

- This leads to the following optimization problem:

$$\min_{\Psi(X) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}} \operatorname{tr}\left(\Psi(X) L \Psi(X)^{\mathsf{T}}\right) \quad \text{subject to } \Psi(X) D \Psi(X)^{\mathsf{T}} = I,$$

where the matrices are defined as

$$A := \left(a_{ij} = w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j)\right), \quad D := \operatorname{diag}\left(\sum_j a_{1j}, \ldots, \sum_j a_{N_{\mathrm{tr}} j}\right).$$

The matrix $L := D - A$ is the (unnormalized) *graph Laplacian*, of course.

Laplacian Eigenmaps (Belkin & Niyogi, 2001–3)

- Incorporating *local* geometric information in $\mathbb{R}^d$ for the embedding
- Define the *proximity* weight $w(\boldsymbol{x}_i, \boldsymbol{x}_j)$, e.g., $w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j) := \mathrm{e}^{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/\epsilon^2}$.
- Now, seek $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ that minimizes the following

$$J_{\mathrm{LE}}(\Psi) := \sum_{i,j} \|\Psi(\boldsymbol{x}_i) - \Psi(\boldsymbol{x}_j)\|^2 w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

- This leads to the following optimization problem:

$$\min_{\Psi(X)\in\mathbb{R}^{s\times N_{\mathrm{tr}}}} \mathrm{tr}\left(\Psi(X) L \Psi(X)^{\mathsf{T}}\right) \quad \text{subject to } \Psi(X) D \Psi(X)^{\mathsf{T}} = I,$$

where the matrices are defined as

$$A := \left(a_{ij} = w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j)\right), \quad D := \mathrm{diag}\left(\sum_j a_{1j}, \ldots, \sum_j a_{N_{\mathrm{tr}} j}\right).$$

The matrix $L := D - A$ is the (unnormalized) *graph Laplacian*, of course.

Laplacian Eigenmaps (Belkin & Niyogi, 2001–3)

- Incorporating *local* geometric information in $\mathbb{R}^d$ for the embedding
- Define the *proximity* weight $w(\boldsymbol{x}_i, \boldsymbol{x}_j)$, e.g., $w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j) := \mathrm{e}^{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/\epsilon^2}$.
- Now, seek $\Psi : \mathbb{R}^d \to \mathbb{R}^s$ that minimizes the following

$$J_{\mathrm{LE}}(\Psi) := \sum_{i,j} \|\Psi(\boldsymbol{x}_i) - \Psi(\boldsymbol{x}_j)\|^2 w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

- This leads to the following optimization problem:

$$\min_{\Psi(X)\in\mathbb{R}^{s\times N_{\mathrm{tr}}}} \mathrm{tr}\left(\Psi(X) L \Psi(X)^{\mathsf{T}}\right) \quad \text{subject to } \Psi(X) D \Psi(X)^{\mathsf{T}} = I,$$

where the matrices are defined as

$$A := \left(a_{ij} = w_\epsilon(\boldsymbol{x}_i, \boldsymbol{x}_j)\right), \quad D := \mathrm{diag}\left(\sum_j a_{1j}, \ldots, \sum_j a_{N_{\mathrm{tr}}j}\right).$$

The matrix $L := D - A$ is the (unnormalized) *graph Laplacian*, of course.

Laplacian Eigenmaps . . .

- This leads to the following *generalized* eigenvalue problem:

$$L\Psi(X)^{\mathsf{T}} = D\Psi(X)^{\mathsf{T}}\Lambda; \quad L \in \mathbb{R}^{N_{\mathrm{tr}}\times N_{\mathrm{tr}}}, \Lambda \in \mathbb{R}^{s\times s},$$

$$\Updownarrow$$

$$L_{\mathrm{rw}}\Psi_{\mathrm{rw}}(X)^{\mathsf{T}} = \Psi_{\mathrm{rw}}(X)^{\mathsf{T}}\Lambda_{\mathrm{rw}}; \quad L_{\mathrm{rw}} := D^{-1}L = I - D^{-1}A.$$

- $\Psi_{\mathrm{rw}}(X) \in \mathbb{R}^{s\times N_{\mathrm{tr}}}$ is the *Laplacian Eigenmap* of X.
- Another possibility is:

$$L_{\mathrm{sym}}\Psi_{\mathrm{sym}}(X)^{\mathsf{T}} = \Psi(X)^{\mathsf{T}}_{\mathrm{sym}}\Lambda_{\mathrm{sym}}; \quad L_{\mathrm{sym}} := D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}.$$

- Both L_{rw} and L_{sym} are called the *normalized* graph Laplacians (rw = 'random walk'; sym = 'symmetric').

$$\Psi_{\mathrm{rw}}(X) = \Psi_{\mathrm{sym}}(X)D^{-\frac{1}{2}}, \quad \Lambda_{\mathrm{rw}} = \Lambda_{\mathrm{sym}}.$$

- Eigenvalues are sorted in *nondecreasing* order; $L_{\mathrm{rw}}\mathbf{1} = \mathbf{0}$.
- A drawback: *sensitive to sampling density on a manifold*.

Laplacian Eigenmaps . . .

- This leads to the following *generalized* eigenvalue problem:

$$L\Psi(X)^{\mathsf{T}} = D\Psi(X)^{\mathsf{T}}\Lambda; \quad L \in \mathbb{R}^{N_{\mathrm{tr}}\times N_{\mathrm{tr}}}, \Lambda \in \mathbb{R}^{s\times s},$$
$$\Updownarrow$$
$$L_{\mathrm{rw}}\Psi_{\mathrm{rw}}(X)^{\mathsf{T}} = \Psi_{\mathrm{rw}}(X)^{\mathsf{T}}\Lambda_{\mathrm{rw}}; \quad L_{\mathrm{rw}} := D^{-1}L = I - D^{-1}A.$$

- $\Psi_{\mathrm{rw}}(X) \in \mathbb{R}^{s\times N_{\mathrm{tr}}}$ is the *Laplacian Eigenmap* of X.
- Another possibility is:

$$L_{\mathrm{sym}}\Psi_{\mathrm{sym}}(X)^{\mathsf{T}} = \Psi(X)^{\mathsf{T}}_{\mathrm{sym}}\Lambda_{\mathrm{sym}}; \quad L_{\mathrm{sym}} := D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}.$$

- Both L_{rw} and L_{sym} are called the *normalized* graph Laplacians (rw = 'random walk'; sym = 'symmetric').

$$\Psi_{\mathrm{rw}}(X) = \Psi_{\mathrm{sym}}(X)D^{-\frac{1}{2}}, \quad \Lambda_{\mathrm{rw}} = \Lambda_{\mathrm{sym}}.$$

- Eigenvalues are sorted in *nondecreasing* order; $L_{\mathrm{rw}}\mathbf{1} = \mathbf{0}$.
- A drawback: *sensitive to sampling density on a manifold.*

Laplacian Eigenmaps . . .

- This leads to the following *generalized* eigenvalue problem:

$$L\Psi(X)^{\mathsf{T}} = D\Psi(X)^{\mathsf{T}}\Lambda; \quad L \in \mathbb{R}^{N_{\mathrm{tr}}\times N_{\mathrm{tr}}}, \Lambda \in \mathbb{R}^{s\times s},$$
$$\Updownarrow$$
$$L_{\mathrm{rw}}\Psi_{\mathrm{rw}}(X)^{\mathsf{T}} = \Psi_{\mathrm{rw}}(X)^{\mathsf{T}}\Lambda_{\mathrm{rw}}; \quad L_{\mathrm{rw}} := D^{-1}L = I - D^{-1}A.$$

- $\Psi_{\mathrm{rw}}(X) \in \mathbb{R}^{s\times N_{\mathrm{tr}}}$ is the *Laplacian Eigenmap* of X.
- Another possibility is:

$$L_{\mathrm{sym}}\Psi_{\mathrm{sym}}(X)^{\mathsf{T}} = \Psi(X)^{\mathsf{T}}_{\mathrm{sym}}\Lambda_{\mathrm{sym}}; \quad L_{\mathrm{sym}} := D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}.$$

- Both L_{rw} and L_{sym} are called the *normalized* graph Laplacians (rw = 'random walk'; sym = 'symmetric').

$$\Psi_{\mathrm{rw}}(X) = \Psi_{\mathrm{sym}}(X)D^{-\frac{1}{2}}, \quad \Lambda_{\mathrm{rw}} = \Lambda_{\mathrm{sym}}.$$

- Eigenvalues are sorted in *nondecreasing* order; $L_{\mathrm{rw}}\mathbf{1} = \mathbf{0}$.
- A drawback: *sensitive to sampling density on a manifold.*

Laplacian Eigenmaps . . .

- This leads to the following *generalized* eigenvalue problem:

$$L\Psi(X)^{\mathsf{T}} = D\Psi(X)^{\mathsf{T}}\Lambda; \quad L \in \mathbb{R}^{N_{\mathrm{tr}} \times N_{\mathrm{tr}}}, \Lambda \in \mathbb{R}^{s \times s},$$
$$\Updownarrow$$
$$L_{\mathrm{rw}}\Psi_{\mathrm{rw}}(X)^{\mathsf{T}} = \Psi_{\mathrm{rw}}(X)^{\mathsf{T}}\Lambda_{\mathrm{rw}}; \quad L_{\mathrm{rw}} := D^{-1}L = I - D^{-1}A.$$

- $\Psi_{\mathrm{rw}}(X) \in \mathbb{R}^{s \times N_{\mathrm{tr}}}$ is the *Laplacian Eigenmap* of X.
- Another possibility is:

$$L_{\mathrm{sym}}\Psi_{\mathrm{sym}}(X)^{\mathsf{T}} = \Psi(X)^{\mathsf{T}}_{\mathrm{sym}}\Lambda_{\mathrm{sym}}; \quad L_{\mathrm{sym}} := D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}.$$

- Both L_{rw} and L_{sym} are called the *normalized* graph Laplacians (rw = 'random walk'; sym = 'symmetric').

$$\Psi_{\mathrm{rw}}(X) = \Psi_{\mathrm{sym}}(X)D^{-\frac{1}{2}}, \quad \Lambda_{\mathrm{rw}} = \Lambda_{\mathrm{sym}}.$$

- Eigenvalues are sorted in *nondecreasing* order; $L_{\mathrm{rw}}\mathbf{1} = \mathbf{0}$.
- A drawback: *sensitive to sampling density on a manifold.*

Laplacian Eigenmaps ...

- This leads to the following *generalized* eigenvalue problem:

$$L\Psi(X)^{\mathsf{T}} = D\Psi(X)^{\mathsf{T}}\Lambda; \quad L \in \mathbb{R}^{N_{\mathrm{tr}}\times N_{\mathrm{tr}}}, \Lambda \in \mathbb{R}^{s\times s},$$
$$\Updownarrow$$
$$L_{\mathrm{rw}}\Psi_{\mathrm{rw}}(X)^{\mathsf{T}} = \Psi_{\mathrm{rw}}(X)^{\mathsf{T}}\Lambda_{\mathrm{rw}}; \quad L_{\mathrm{rw}} := D^{-1}L = I - D^{-1}A.$$

- $\Psi_{\mathrm{rw}}(X) \in \mathbb{R}^{s\times N_{\mathrm{tr}}}$ is the *Laplacian Eigenmap* of X.
- Another possibility is:

$$L_{\mathrm{sym}}\Psi_{\mathrm{sym}}(X)^{\mathsf{T}} = \Psi(X)^{\mathsf{T}}_{\mathrm{sym}}\Lambda_{\mathrm{sym}}; \quad L_{\mathrm{sym}} := D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}.$$

- Both L_{rw} and L_{sym} are called the *normalized* graph Laplacians (rw = 'random walk'; sym = 'symmetric').

$$\Psi_{\mathrm{rw}}(X) = \Psi_{\mathrm{sym}}(X)D^{-\frac{1}{2}}, \quad \Lambda_{\mathrm{rw}} = \Lambda_{\mathrm{sym}}.$$

- Eigenvalues are sorted in *nondecreasing* order; $L_{\mathrm{rw}}\mathbf{1} = \mathbf{0}$.
- A drawback: *sensitive to sampling density on a manifold.*

Laplacian Eigenmaps . . .

- This leads to the following *generalized* eigenvalue problem:

$$L\Psi(X)^{\mathsf{T}} = D\Psi(X)^{\mathsf{T}}\Lambda; \quad L \in \mathbb{R}^{N_{\mathrm{tr}}\times N_{\mathrm{tr}}}, \Lambda \in \mathbb{R}^{s\times s},$$

$$\Updownarrow$$

$$L_{\mathrm{rw}}\Psi_{\mathrm{rw}}(X)^{\mathsf{T}} = \Psi_{\mathrm{rw}}(X)^{\mathsf{T}}\Lambda_{\mathrm{rw}}; \quad L_{\mathrm{rw}} := D^{-1}L = I - D^{-1}A.$$

- $\Psi_{\mathrm{rw}}(X) \in \mathbb{R}^{s\times N_{\mathrm{tr}}}$ is the *Laplacian Eigenmap* of X.
- Another possibility is:

$$L_{\mathrm{sym}}\Psi_{\mathrm{sym}}(X)^{\mathsf{T}} = \Psi(X)^{\mathsf{T}}_{\mathrm{sym}}\Lambda_{\mathrm{sym}}; \quad L_{\mathrm{sym}} := D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}.$$

- Both L_{rw} and L_{sym} are called the *normalized* graph Laplacians (rw = 'random walk'; sym = 'symmetric').

$$\Psi_{\mathrm{rw}}(X) = \Psi_{\mathrm{sym}}(X)D^{-\frac{1}{2}}, \quad \Lambda_{\mathrm{rw}} = \Lambda_{\mathrm{sym}}.$$

- Eigenvalues are sorted in *nondecreasing* order; $L_{\mathrm{rw}}\mathbf{1} = \mathbf{0}$.
- A drawback: *sensitive to sampling density on a manifold*.

Outline

Diffusion Maps (Coifman & Lafon 2004–6)

- Focus on the normalized weighted adjacency matrix $P := D^{-1}A$.
- Interpret P as the *transition matrix* of a random walk on X or the *diffusion operator* on X. P^t = running the random walk t steps.
- Perform *density invariant normalization* on A, i.e., $\widetilde{A} := D^{-1}AD^{-1}$ first. Then, do the row-stochastic normalization, i.e., $\widetilde{P} := \widetilde{D}^{-1}\widetilde{A}$ where $\widetilde{D}$ is the degree matrix (diagonal) of $\widetilde{A}$.
- Finally perform the eigenanalysis:

$$\widetilde{P}\Psi_{\mathrm{DM}}(X)^{\mathsf{T}} = \Psi_{\mathrm{DM}}(X)^{\mathsf{T}}\Lambda_{\mathrm{DM}},$$

  where the eigenvalues are sorted in *nonincreasing* order; $\widetilde{P}\mathbf{1} = \mathbf{1}$.
- *Diffusion map* is defined as:

$$\Psi^t_{\mathrm{DM}}(X) := \Lambda^t_{\mathrm{DM}}\Psi_{\mathrm{DM}}(X).$$

- Relationship with the Laplacian eigenmap (if $L_{\mathrm{rw}} = I - \widetilde{P}$ is used instead of usual $L_{\mathrm{rw}} = I - P$):

$$\Psi^1_{\mathrm{DM}}(X) = \Psi_{\mathrm{rw}}(X); \quad \Lambda_{\mathrm{DM}} = I - \Lambda_{\mathrm{rw}}.$$

Diffusion Maps (Coifman & Lafon 2004–6)

- Focus on the normalized weighted adjacency matrix $P := D^{-1}A$.
- Interpret P as the *transition matrix* of a random walk on X or the *diffusion operator* on X. P^t = running the random walk t steps.
- Perform *density invariant normalization* on A, i.e., $\widetilde{A} := D^{-1}AD^{-1}$ first. Then, do the row-stochastic normalization, i.e., $\widetilde{P} := \widetilde{D}^{-1}\widetilde{A}$ where $\widetilde{D}$ is the degree matrix (diagonal) of $\widetilde{A}$.
- Finally perform the eigenanalysis:

$$\widetilde{P}\Psi_{\mathrm{DM}}(X)^{\top} = \Psi_{\mathrm{DM}}(X)^{\top}\Lambda_{\mathrm{DM}},$$

  where the eigenvalues are sorted in *nonincreasing* order; $\widetilde{P}\mathbf{1} = \mathbf{1}$.
- *Diffusion map* is defined as:

$$\Psi^t_{\mathrm{DM}}(X) := \Lambda^t_{\mathrm{DM}}\Psi_{\mathrm{DM}}(X).$$

- Relationship with the Laplacian eigenmap (if $L_{\mathrm{rw}} = I - \widetilde{P}$ is used instead of usual $L_{\mathrm{rw}} = I - P$):

$$\Psi^1_{\mathrm{DM}}(X) = \Psi_{\mathrm{rw}}(X); \quad \Lambda_{\mathrm{DM}} = I - \Lambda_{\mathrm{rw}}.$$

Diffusion Maps (Coifman & Lafon 2004–6)

- Focus on the normalized weighted adjacency matrix $P := D^{-1}A$.
- Interpret P as the *transition matrix* of a random walk on X or the *diffusion operator* on X. P^t = running the random walk t steps.
- Perform *density invariant normalization* on A, i.e., $\widetilde{A} := D^{-1}AD^{-1}$ first. Then, do the row-stochastic normalization, i.e., $\widetilde{P} := \widetilde{D}^{-1}\widetilde{A}$ where $\widetilde{D}$ is the degree matrix (diagonal) of $\widetilde{A}$.
- Finally perform the eigenanalysis:

$$\widetilde{P}\Psi_{\mathrm{DM}}(X)^{\mathsf{T}} = \Psi_{\mathrm{DM}}(X)^{\mathsf{T}}\Lambda_{\mathrm{DM}},$$

  where the eigenvalues are sorted in *nonincreasing* order; $\widetilde{P}\mathbf{1} = \mathbf{1}$.
- *Diffusion map* is defined as:

$$\Psi^t_{\mathrm{DM}}(X) := \Lambda^t_{\mathrm{DM}}\Psi_{\mathrm{DM}}(X).$$

- Relationship with the Laplacian eigenmap (if $L_{\mathrm{rw}} = I - \widetilde{P}$ is used instead of usual $L_{\mathrm{rw}} = I - P$):

$$\Psi^1_{\mathrm{DM}}(X) = \Psi_{\mathrm{rw}}(X); \quad \Lambda_{\mathrm{DM}} = I - \Lambda_{\mathrm{rw}}.$$

Diffusion Maps (Coifman & Lafon 2004–6)

- Focus on the normalized weighted adjacency matrix $P := D^{-1}A$.
- Interpret P as the *transition matrix* of a random walk on X or the *diffusion operator* on X. P^t = running the random walk t steps.
- Perform *density invariant normalization* on A, i.e., $\widetilde{A} := D^{-1}AD^{-1}$ first. Then, do the row-stochastic normalization, i.e., $\widetilde{P} := \widetilde{D}^{-1}\widetilde{A}$ where $\widetilde{D}$ is the degree matrix (diagonal) of $\widetilde{A}$.
- Finally perform the eigenanalysis:

$$\widetilde{P}\Psi_{\mathrm{DM}}(X)^{\mathsf{T}} = \Psi_{\mathrm{DM}}(X)^{\mathsf{T}}\Lambda_{\mathrm{DM}},$$

  where the eigenvalues are sorted in *nonincreasing* order; $\widetilde{P}\mathbf{1} = \mathbf{1}$.
- *Diffusion map* is defined as:

$$\Psi^t_{\mathrm{DM}}(X) := \Lambda^t_{\mathrm{DM}}\Psi_{\mathrm{DM}}(X).$$

- Relationship with the Laplacian eigenmap (if $L_{\mathrm{rw}} = I - \widetilde{P}$ is used instead of usual $L_{\mathrm{rw}} = I - P$):

$$\Psi^1_{\mathrm{DM}}(X) = \Psi_{\mathrm{rw}}(X); \quad \Lambda_{\mathrm{DM}} = I - \Lambda_{\mathrm{rw}}.$$

Diffusion Maps (Coifman & Lafon 2004–6)

- Focus on the normalized weighted adjacency matrix $P := D^{-1}A$.
- Interpret P as the *transition matrix* of a random walk on X or the *diffusion operator* on X. P^t = running the random walk t steps.
- Perform *density invariant normalization* on A, i.e., $\widetilde{A} := D^{-1}AD^{-1}$ first. Then, do the row-stochastic normalization, i.e., $\widetilde{P} := \widetilde{D}^{-1}\widetilde{A}$ where $\widetilde{D}$ is the degree matrix (diagonal) of $\widetilde{A}$.
- Finally perform the eigenanalysis:

$$\widetilde{P}\Psi_{\mathrm{DM}}(X)^{\mathsf{T}} = \Psi_{\mathrm{DM}}(X)^{\mathsf{T}}\Lambda_{\mathrm{DM}},$$

  where the eigenvalues are sorted in *nonincreasing* order; $\widetilde{P}\mathbf{1} = \mathbf{1}$.
- *Diffusion map* is defined as:

$$\Psi^t_{\mathrm{DM}}(X) := \Lambda^t_{\mathrm{DM}}\Psi_{\mathrm{DM}}(X).$$

- Relationship with the Laplacian eigenmap (if $L_{\mathrm{rw}} = I - \widetilde{P}$ is used instead of usual $L_{\mathrm{rw}} = I - P$):

$$\Psi^1_{\mathrm{DM}}(X) = \Psi_{\mathrm{rw}}(X); \quad \Lambda_{\mathrm{DM}} = I - \Lambda_{\mathrm{rw}}.$$

Diffusion Maps (Coifman & Lafon 2004–6)

- Focus on the normalized weighted adjacency matrix $P := D^{-1}A$.
- Interpret P as the *transition matrix* of a random walk on X or the *diffusion operator* on X. P^t = running the random walk t steps.
- Perform *density invariant normalization* on A, i.e., $\widetilde{A} := D^{-1}AD^{-1}$ first. Then, do the row-stochastic normalization, i.e., $\widetilde{P} := \widetilde{D}^{-1}\widetilde{A}$ where $\widetilde{D}$ is the degree matrix (diagonal) of $\widetilde{A}$.
- Finally perform the eigenanalysis:

$$\widetilde{P}\Psi_{\mathrm{DM}}(X)^{\mathsf{T}} = \Psi_{\mathrm{DM}}(X)^{\mathsf{T}}\Lambda_{\mathrm{DM}},$$

  where the eigenvalues are sorted in *nonincreasing* order; $\widetilde{P}\mathbf{1} = \mathbf{1}$.
- *Diffusion map* is defined as:

$$\Psi^t_{\mathrm{DM}}(X) := \Lambda^t_{\mathrm{DM}}\Psi_{\mathrm{DM}}(X).$$

- Relationship with the Laplacian eigenmap (if $L_{\mathrm{rw}} = I - \widetilde{P}$ is used instead of usual $L_{\mathrm{rw}} = I - P$):

$$\Psi^1_{\mathrm{DM}}(X) = \Psi_{\mathrm{rw}}(X); \quad \Lambda_{\mathrm{DM}} = I - \Lambda_{\mathrm{rw}}.$$

Density Invariant Normalization

is important for the mapping to be *less dependent on the sampling density* on a manifold in $\mathbb{R}^d$.

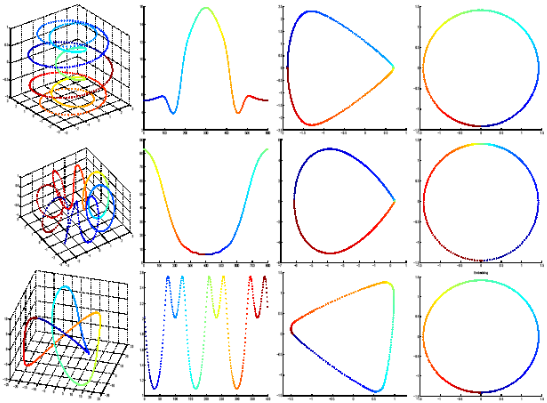

Figure: Courtesy: R. R. Coifman & S. Lafon. From left to right: 3D curves to be embedded into 2D; Sampling densities along curves; Embeddings by LE; Embeddings by DM.

Remarks

- The *rows* of $\Psi_{\mathrm{DM}}(X)$ (when they are transposed) are the *right* eigenvectors of $\widetilde{P}$.
- Can use SVD or symmetric eigenvalue solver for computing these embedding maps.
- Choosing a good scale parameter ϵ for both LE and DM is not easy:
  - $\epsilon =$ the mean of the k-nearest neighbor distances.
  - But how to choose k?

  $\Longrightarrow$ Cross validation, etc.
- For DM, choosing t or when to stop the diffusion is another subtle question, which is quite dependent on ϵ and the decay of the eigenvalues.
- Choosing an appropriate value of s is yet another problem $\Longrightarrow$ *Elongated* K-means algorithm:
  G. Sanguinetti, J. Laidler, and N. D. Lawrence, "Automatic determination of the number of clusters using spectral algorithms," *Proc. 15th IEEE Workshop on Machine Learning for Signal Processing*, pp. 55–60, 2005.

Remarks

- The *rows* of $\Psi_{\mathrm{DM}}(X)$ (when they are transposed) are the *right* eigenvectors of $\widetilde{P}$.
- Can use SVD or symmetric eigenvalue solver for computing these embedding maps.
- Choosing a good scale parameter ϵ for both LE and DM is not easy:
  - $\epsilon =$ the mean of the k-nearest neighbor distances.
  - But how to choose k?

  $\Longrightarrow$ Cross validation, etc.
- For DM, choosing t or when to stop the diffusion is another subtle question, which is quite dependent on ϵ and the decay of the eigenvalues.
- Choosing an appropriate value of s is yet another problem $\Longrightarrow$ *Elongated* K-means algorithm:
  G. Sanguinetti, J. Laidler, and N. D. Lawrence, "Automatic determination of the number of clusters using spectral algorithms," *Proc. 15th IEEE Workshop on Machine Learning for Signal Processing*, pp. 55–60, 2005.

Remarks

- The *rows* of $\Psi_{\mathrm{DM}}(X)$ (when they are transposed) are the *right* eigenvectors of $\widetilde{P}$.
- Can use SVD or symmetric eigenvalue solver for computing these embedding maps.
- Choosing a good scale parameter ϵ for both LE and DM is not easy:
  - $\epsilon =$ the mean of the k-nearest neighbor distances.
  - But how to choose k?

  $\Longrightarrow$ Cross validation, etc.
- For DM, choosing t or when to stop the diffusion is another subtle question, which is quite dependent on ϵ and the decay of the eigenvalues.
- Choosing an appropriate value of s is yet another problem $\Longrightarrow$ *Elongated* K-means algorithm:
  G. Sanguinetti, J. Laidler, and N. D. Lawrence, "Automatic determination of the number of clusters using spectral algorithms," *Proc. 15th IEEE Workshop on Machine Learning for Signal Processing*, pp. 55–60, 2005.

Remarks

- The *rows* of $\Psi_{\mathrm{DM}}(X)$ (when they are transposed) are the *right* eigenvectors of $\widetilde{P}$.
- Can use SVD or symmetric eigenvalue solver for computing these embedding maps.
- Choosing a good scale parameter ϵ for both LE and DM is not easy:
  - $\epsilon =$ the mean of the k-nearest neighbor distances.
  - But how to choose k?

  $\Longrightarrow$ Cross validation, etc.
- For DM, choosing t or when to stop the diffusion is another subtle question, which is quite dependent on ϵ and the decay of the eigenvalues.
- Choosing an appropriate value of s is yet another problem $\Longrightarrow$ *Elongated* K-means algorithm:
  G. Sanguinetti, J. Laidler, and N. D. Lawrence, "Automatic determination of the number of clusters using spectral algorithms," *Proc. 15th IEEE Workshop on Machine Learning for Signal Processing*, pp. 55–60, 2005.

Remarks

- The *rows* of $\Psi_{\mathrm{DM}}(X)$ (when they are transposed) are the *right* eigenvectors of $\widetilde{P}$.
- Can use SVD or symmetric eigenvalue solver for computing these embedding maps.
- Choosing a good scale parameter ϵ for both LE and DM is not easy:
  - $\epsilon =$ the mean of the k-nearest neighbor distances.
  - But how to choose k?

  $\Longrightarrow$ Cross validation, etc.
- For DM, choosing t or when to stop the diffusion is another subtle question, which is quite dependent on ϵ and the decay of the eigenvalues.
- Choosing an appropriate value of s is yet another problem $\Longrightarrow$ *Elongated* K-means algorithm:
  G. Sanguinetti, J. Laidler, and N. D. Lawrence, "Automatic determination of the number of clusters using spectral algorithms," *Proc. 15th IEEE Workshop on Machine Learning for Signal Processing*, pp. 55–60, 2005.

Remarks

- The *rows* of $\Psi_{\mathrm{DM}}(X)$ (when they are transposed) are the *right* eigenvectors of $\widetilde{P}$.
- Can use SVD or symmetric eigenvalue solver for computing these embedding maps.
- Choosing a good scale parameter ϵ for both LE and DM is not easy:
  - $\epsilon =$ the mean of the k-nearest neighbor distances.
  - But how to choose k?

  $\Longrightarrow$ Cross validation, etc.
- For DM, choosing t or when to stop the diffusion is another subtle question, which is quite dependent on ϵ and the decay of the eigenvalues.
- Choosing an appropriate value of s is yet another problem $\Longrightarrow$ *Elongated* K-means algorithm:
  G. Sanguinetti, J. Laidler, and N. D. Lawrence, "Automatic determination of the number of clusters using spectral algorithms," *Proc. 15th IEEE Workshop on Machine Learning for Signal Processing*, pp. 55–60, 2005.

Remarks

- The *rows* of $\Psi_{\mathrm{DM}}(X)$ (when they are transposed) are the *right* eigenvectors of $\widetilde{P}$.
- Can use SVD or symmetric eigenvalue solver for computing these embedding maps.
- Choosing a good scale parameter ϵ for both LE and DM is not easy:
  - $\epsilon =$ the mean of the k-nearest neighbor distances.
  - But how to choose k?

  $\Longrightarrow$ Cross validation, etc.
- For DM, choosing t or when to stop the diffusion is another subtle question, which is quite dependent on ϵ and the decay of the eigenvalues.
- Choosing an appropriate value of s is yet another problem $\Longrightarrow$ *Elongated* K-means algorithm:
  G. Sanguinetti, J. Laidler, and N. D. Lawrence, "Automatic determination of the number of clusters using spectral algorithms," *Proc. 15th IEEE Workshop on Machine Learning for Signal Processing*, pp. 55–60, 2005.

Remarks

- The *rows* of $\Psi_{\mathrm{DM}}(X)$ (when they are transposed) are the *right* eigenvectors of $\widetilde{P}$.
- Can use SVD or symmetric eigenvalue solver for computing these embedding maps.
- Choosing a good scale parameter ϵ for both LE and DM is not easy:
  - $\epsilon =$ the mean of the k-nearest neighbor distances.
  - But how to choose k?

  $\Longrightarrow$ Cross validation, etc.
- For DM, choosing t or when to stop the diffusion is another subtle question, which is quite dependent on ϵ and the decay of the eigenvalues.
- Choosing an appropriate value of s is yet another problem $\Longrightarrow$ *Elongated* K-means algorithm:
  G. Sanguinetti, J. Laidler, and N. D. Lawrence, "Automatic determination of the number of clusters using spectral algorithms," *Proc. 15th IEEE Workshop on Machine Learning for Signal Processing*, pp. 55–60, 2005.

Outline

Extension of Maps for Test Data

- Need such extensions because all the maps and embeddings are computed *only based on the training dataset* X; no one wants to recompute those maps from scratch using both the training dataset $X \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$ and the test dataset $Y \in \mathbb{R}^{d \times N_{\mathrm{te}}}$.
- For PCA, it is quite easy; simply the multiplication of U_s^{T} to Y.
- For LE/DM, it is more involved and one needs to use an extension algorithm something similar to the following *geometric harmonics multiscale extension* algorithm:
  S. Lafon, Y. Keller, R. R. Coifman, "Data fusion and multicue data matching by diffusion maps," *IEEE Trans. Pattern Anal. Machine Intell.*, vol.28, no.11, pp.1784–1797, 2006.

Extension of Maps for Test Data

- Need such extensions because all the maps and embeddings are computed *only based on the training dataset* X; no one wants to recompute those maps from scratch using both the training dataset $X \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$ and the test dataset $Y \in \mathbb{R}^{d \times N_{\mathrm{te}}}$.
- For PCA, it is quite easy; simply the multiplication of U_s^{T} to Y.
- For LE/DM, it is more involved and one needs to use an extension algorithm something similar to the following *geometric harmonics multiscale extension* algorithm:
  S. Lafon, Y. Keller, R. R. Coifman, "Data fusion and multicue data matching by diffusion maps," *IEEE Trans. Pattern Anal. Machine Intell.*, vol.28, no.11, pp.1784–1797, 2006.

Extension of Maps for Test Data

- Need such extensions because all the maps and embeddings are computed *only based on the training dataset* X; no one wants to recompute those maps from scratch using both the training dataset $X \in \mathbb{R}^{d \times N_{\mathrm{tr}}}$ and the test dataset $Y \in \mathbb{R}^{d \times N_{\mathrm{te}}}$.
- For PCA, it is quite easy; simply the multiplication of U_s^{T} to Y.
- For LE/DM, it is more involved and one needs to use an extension algorithm something similar to the following *geometric harmonics multiscale extension* algorithm:
  S. Lafon, Y. Keller, R. R. Coifman, "Data fusion and multicue data matching by diffusion maps," *IEEE Trans. Pattern Anal. Machine Intell.*, vol.28, no.11, pp.1784–1797, 2006.

Geometric Harmonics Multiscale Extension (GHME)

- Is an improvement of *the Nyström extension method* proposed by Fowlkes et al. (2004), and Bengio et al. (2004).
- First consider the Gaussian kernel matrix defined on X with the scale parameter $\sigma > 0$, which is *different* from ϵ used in the weight function in for constructing LE/DM, as follows:

$$W_\sigma(X) := \left(w_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j)\right) = \left(\mathrm{e}^{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/\sigma^2}\right) \in \mathbb{R}^{N_{\mathrm{tr}} \times N_{\mathrm{tr}}}.$$

- $W_\sigma(X)$ is positive semi-definite and its eigendecomposition is:

$$W_\sigma(X) = \Phi^{\mathsf{T}} M \Phi,\ \Phi^{\mathsf{T}} := \left[\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_{N_{\mathrm{tr}}}\right] \in \mathbb{R}^{N_{\mathrm{tr}} \times N_{\mathrm{tr}}},\ M := \operatorname{diag}(\mu_1, \ldots, \mu_{N_{\mathrm{tr}}}).$$

where $\mu_1 \geq \cdots \geq \mu_{N_{\mathrm{tr}}} \geq 0,\ \boldsymbol{\phi}_i := (\phi_i(\boldsymbol{x}_1), \ldots, \phi_i(\boldsymbol{x}_{N_{\mathrm{tr}}}))^{\mathsf{T}},\ i = 1, \ldots, N_{\mathrm{tr}}.$

Geometric Harmonics Multiscale Extension (GHME)

- Is an improvement of *the Nyström extension method* proposed by Fowlkes et al. (2004), and Bengio et al. (2004).
- First consider the Gaussian kernel matrix defined on X with the scale parameter $\sigma > 0$, which is *different* from ϵ used in the weight function in for constructing LE/DM, as follows:

$$W_\sigma(X) := \left(w_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j)\right) = \left(\mathrm{e}^{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/\sigma^2}\right) \in \mathbb{R}^{N_{\mathrm{tr}} \times N_{\mathrm{tr}}}.$$

- $W_\sigma(X)$ is positive semi-definite and its eigendecomposition is:

$$W_\sigma(X) = \Phi^\top M \Phi,\ \Phi^\top := \left[\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_{N_{\mathrm{tr}}}\right] \in \mathbb{R}^{N_{\mathrm{tr}} \times N_{\mathrm{tr}}},\ M := \mathrm{diag}(\mu_1, \ldots, \mu_{N_{\mathrm{tr}}}).$$

where $\mu_1 \geq \cdots \geq \mu_{N_{\mathrm{tr}}} \geq 0,\ \boldsymbol{\phi}_i := (\phi_i(\boldsymbol{x}_1), \ldots, \phi_i(\boldsymbol{x}_{N_{\mathrm{tr}}}))^\top,\ i = 1, \ldots, N_{\mathrm{tr}}.$

Geometric Harmonics Multiscale Extension (GHME)

- Is an improvement of *the Nyström extension method* proposed by Fowlkes et al. (2004), and Bengio et al. (2004).
- First consider the Gaussian kernel matrix defined on X with the scale parameter $\sigma > 0$, which is *different* from ϵ used in the weight function in for constructing LE/DM, as follows:

$$W_\sigma(X) := \left(w_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j)\right) = \left(\mathrm{e}^{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/\sigma^2}\right) \in \mathbb{R}^{N_{\mathrm{tr}} \times N_{\mathrm{tr}}}.$$

- $W_\sigma(X)$ is positive semi-definite and its eigendecomposition is:

$$W_\sigma(X) = \Phi^\mathsf{T} M \Phi, \Phi^\mathsf{T} := \left[\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_{N_{\mathrm{tr}}}\right] \in \mathbb{R}^{N_{\mathrm{tr}} \times N_{\mathrm{tr}}}, M := \mathrm{diag}(\mu_1, \ldots, \mu_{N_{\mathrm{tr}}}).$$

where $\mu_1 \geq \cdots \geq \mu_{N_{\mathrm{tr}}} \geq 0,\ \boldsymbol{\phi}_i := (\phi_i(\boldsymbol{x}_1), \ldots, \phi_i(\boldsymbol{x}_{N_{\mathrm{tr}}}))^\mathsf{T},\ i = 1, \ldots, N_{\mathrm{tr}}$.

Geometric Harmonics Multiscale Extension (GHME) ...

- Now, consider the kth eigenpair $(\mu_k, \boldsymbol{\phi}_k)$, i.e., $W_\sigma(X)\boldsymbol{\phi}_k = \mu_k \boldsymbol{\phi}_k$. The ith row of this equality gives us

$$\phi_k(\boldsymbol{x}_i) = \frac{1}{\mu_k} \sum_{j=1}^{N_{\mathrm{tr}}} w_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j)\phi_k(\boldsymbol{x}_j).$$

- The *Nyström extension* of $\boldsymbol{\phi}_k$ from X to $\boldsymbol{y} \in Y$ is defined as

$$\overline{\phi}_k(\boldsymbol{y}) := \frac{1}{\mu_k} \sum_{j=1}^{N_{\mathrm{tr}}} w_\sigma(\boldsymbol{y}, \boldsymbol{x}_j)\phi_k(\boldsymbol{x}_j).$$

- Since the eigenvectors $\{\boldsymbol{\phi}_k\}$ form an orthonormal basis for $\mathbb{R}^{N_{\mathrm{tr}}}$, any function $\boldsymbol{f} := (f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_{N_{\mathrm{tr}}}))^{\mathsf{T}} \in \mathbb{R}^{N_{\mathrm{tr}}}$ can be expanded as

$$\boldsymbol{f} = \sum_{k=1}^{N_{\mathrm{tr}}} \langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle \boldsymbol{\phi}_k, \quad \text{i.e., } f(\boldsymbol{x}_j) = \sum_{k=1}^{N_{\mathrm{tr}}} \langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle \phi_k(\boldsymbol{x}_j),\ j = 1, \ldots, N_{\mathrm{tr}}.$$

Geometric Harmonics Multiscale Extension (GHME) . . .

- Now, consider the kth eigenpair $(\mu_k, \boldsymbol{\phi}_k)$, i.e., $W_\sigma(X)\boldsymbol{\phi}_k = \mu_k \boldsymbol{\phi}_k$. The ith row of this equality gives us

$$\phi_k(\boldsymbol{x}_i) = \frac{1}{\mu_k} \sum_{j=1}^{N_{\mathrm{tr}}} w_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j)\phi_k(\boldsymbol{x}_j).$$

- The *Nyström extension* of $\boldsymbol{\phi}_k$ from X to $\boldsymbol{y} \in Y$ is defined as

$$\overline{\phi}_k(\boldsymbol{y}) := \frac{1}{\mu_k} \sum_{j=1}^{N_{\mathrm{tr}}} w_\sigma(\boldsymbol{y}, \boldsymbol{x}_j)\phi_k(\boldsymbol{x}_j).$$

- Since the eigenvectors $\{\boldsymbol{\phi}_k\}$ form an orthonormal basis for $\mathbb{R}^{N_{\mathrm{tr}}}$, any function $\boldsymbol{f} := (f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_{N_{\mathrm{tr}}}))^\mathsf{T} \in \mathbb{R}^{N_{\mathrm{tr}}}$ can be expanded as

$$\boldsymbol{f} = \sum_{k=1}^{N_{\mathrm{tr}}} \langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle \boldsymbol{\phi}_k, \quad \text{i.e., } f(\boldsymbol{x}_j) = \sum_{k=1}^{N_{\mathrm{tr}}} \langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle \phi_k(\boldsymbol{x}_j),\ j = 1, \ldots, N_{\mathrm{tr}}.$$

Geometric Harmonics Multiscale Extension (GHME) ...

- Now, consider the kth eigenpair $(\mu_k, \boldsymbol{\phi}_k)$, i.e., $W_\sigma(X)\boldsymbol{\phi}_k = \mu_k \boldsymbol{\phi}_k$. The ith row of this equality gives us

$$\phi_k(\boldsymbol{x}_i) = \frac{1}{\mu_k} \sum_{j=1}^{N_{\text{tr}}} w_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j) \phi_k(\boldsymbol{x}_j).$$

- The *Nyström extension* of $\boldsymbol{\phi}_k$ from X to $\boldsymbol{y} \in Y$ is defined as

$$\overline{\phi}_k(\boldsymbol{y}) := \frac{1}{\mu_k} \sum_{j=1}^{N_{\text{tr}}} w_\sigma(\boldsymbol{y}, \boldsymbol{x}_j) \phi_k(\boldsymbol{x}_j).$$

- Since the eigenvectors $\{\boldsymbol{\phi}_k\}$ form an orthonormal basis for $\mathbb{R}^{N_{\text{tr}}}$, any function $\boldsymbol{f} := (f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_{N_{\text{tr}}}))^\mathsf{T} \in \mathbb{R}^{N_{\text{tr}}}$ can be expanded as

$$\boldsymbol{f} = \sum_{k=1}^{N_{\text{tr}}} \langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle \boldsymbol{\phi}_k, \quad \text{i.e., } f(\boldsymbol{x}_j) = \sum_{k=1}^{N_{\text{tr}}} \langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle \phi_k(\boldsymbol{x}_j),\ j = 1, \ldots, N_{\text{tr}}.$$

Geometric Harmonics Multiscale Extension (GHME) ...

- Thus the *Nyström extension* of f from X to $\boldsymbol{y} \in Y$ can be defined as

$$\overline{f}(\boldsymbol{y}) := \sum_{k=1}^{N_{\mathrm{tr}}} \langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle \overline{\phi}_k(\boldsymbol{y}).$$

- In order to understand what we have done here, let us plug the Nyström extension formula of $\boldsymbol{\phi}_k$'s in the righthand side of the above equation.

$$\begin{aligned}
\overline{f}(\boldsymbol{y}) &= \sum_{k=1}^{N_{\mathrm{tr}}} \frac{\langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle}{\mu_k} \sum_{j=1}^{N_{\mathrm{tr}}} w_\sigma(\boldsymbol{y}, \boldsymbol{x}_j) \phi_k(\boldsymbol{x}_j) \\
&= \sum_{k=1}^{N_{\mathrm{tr}}} \frac{\boldsymbol{\phi}_k^{\top} \boldsymbol{f}}{\mu_k} w_\sigma(\boldsymbol{y}, :) \boldsymbol{\phi}_k \\
&= w_\sigma(\boldsymbol{y}, :) \Phi^{\top} M^{-1} \Phi \boldsymbol{f},
\end{aligned}$$

where $w_\sigma(\boldsymbol{y}, :) := [w_\sigma(\boldsymbol{y}, \boldsymbol{x}_1), \ldots, w_\sigma(\boldsymbol{y}, \boldsymbol{x}_{N_{\mathrm{tr}}})] \in \mathbb{R}^{1 \times N_{\mathrm{tr}}}$.

Geometric Harmonics Multiscale Extension (GHME) . . .

- Thus the *Nyström extension* of f from X to $\boldsymbol{y} \in Y$ can be defined as

$$\overline{f}(\boldsymbol{y}) := \sum_{k=1}^{N_{\mathrm{tr}}} \langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle \overline{\phi}_k(\boldsymbol{y}).$$

- In order to understand what we have done here, let us plug the Nyström extension formula of $\boldsymbol{\phi}_k$'s in the righthand side of the above equation.

$$\begin{aligned} \overline{f}(\boldsymbol{y}) &= \sum_{k=1}^{N_{\mathrm{tr}}} \frac{\langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle}{\mu_k} \sum_{j=1}^{N_{\mathrm{tr}}} w_\sigma(\boldsymbol{y}, \boldsymbol{x}_j) \phi_k(\boldsymbol{x}_j) \\ &= \sum_{k=1}^{N_{\mathrm{tr}}} \frac{\boldsymbol{\phi}_k^{\mathsf{T}} \boldsymbol{f}}{\mu_k} w_\sigma(\boldsymbol{y}, :) \boldsymbol{\phi}_k \\ &= w_\sigma(\boldsymbol{y}, :) \Phi^{\mathsf{T}} M^{-1} \Phi \boldsymbol{f}, \end{aligned}$$

where $w_\sigma(\boldsymbol{y}, :) := [w_\sigma(\boldsymbol{y}, \boldsymbol{x}_1), \ldots, w_\sigma(\boldsymbol{y}, \boldsymbol{x}_{N_{\mathrm{tr}}})] \in \mathbb{R}^{1 \times N_{\mathrm{tr}}}$.

Geometric Harmonics Multiscale Extension (GHME) . . .

- Observe that the range of the above extension is proportional to σ. If the ratio $\|\boldsymbol{y}-\boldsymbol{x}_j\|/\sigma$ is large for all $\boldsymbol{x}_j \in X$, then $\overline{\phi}_k(\boldsymbol{y})$ will be numerically small and hence may not be meaningful. Hence the extension scale σ should be as large as possible.
- On the other hand, for large enough σ, $W_\sigma(X)$ becomes *ill-conditioned*, i.e., μ_k tends to 0 more quickly compared to the case of small σ. Thus the above Nyström extension will blow up.
- Furthermore, it is well known that the extension range depends on the smoothness of the function to be extended: If f is fairly smooth, it can be extended far away from the training set while it has limited extension range if f varies wildly on X.
- In fact, if we can compute M^{-1} *without* blowing up, i.e., $\mu_{N_{\mathrm{tr}}} \gneqq 0$, then $\Phi^\top M^{-1}\Phi = W_\sigma(X)^{-1}$.
- Moreover, by setting $\boldsymbol{y} = \boldsymbol{x}_j \in X$ in the Nyström extension formula, we can recover $f(\boldsymbol{x}_j)$:

$$w_\sigma(\boldsymbol{x}_j,:)\Phi^\top M^{-1}\Phi \boldsymbol{f} = w_\sigma(\boldsymbol{x}_j,:)W_\sigma(X)^{-1}\boldsymbol{f} = \boldsymbol{e}_j^\top \boldsymbol{f} = f(\boldsymbol{x}_j).$$

Geometric Harmonics Multiscale Extension (GHME) . . .

- Observe that the range of the above extension is proportional to σ. If the ratio $\|\boldsymbol{y}-\boldsymbol{x}_j\|/\sigma$ is large for all $\boldsymbol{x}_j \in X$, then $\overline{\phi}_k(\boldsymbol{y})$ will be numerically small and hence may not be meaningful. Hence the extension scale σ should be as large as possible.
- On the other hand, for large enough σ, $W_\sigma(X)$ becomes *ill-conditioned*, i.e., μ_k tends to 0 more quickly compared to the case of small σ. Thus the above Nyström extension will blow up.
- Furthermore, it is well known that the extension range depends on the smoothness of the function to be extended: If f is fairly smooth, it can be extended far away from the training set while it has limited extension range if f varies wildly on X.
- In fact, if we can compute M^{-1} *without* blowing up, i.e., $\mu_{N_{\mathrm{tr}}} \gneqq 0$, then $\Phi^{\mathsf{T}} M^{-1} \Phi = W_\sigma(X)^{-1}$.
- Moreover, by setting $\boldsymbol{y} = \boldsymbol{x}_j \in X$ in the Nyström extension formula, we can recover $f(\boldsymbol{x}_j)$:

$$w_\sigma(\boldsymbol{x}_j,:)\Phi^{\mathsf{T}} M^{-1}\Phi \boldsymbol{f} = w_\sigma(\boldsymbol{x}_j,:)W_\sigma(X)^{-1}\boldsymbol{f} = \boldsymbol{e}_j^{\mathsf{T}}\boldsymbol{f} = f(\boldsymbol{x}_j).$$

Geometric Harmonics Multiscale Extension (GHME) . . .

- Observe that the range of the above extension is proportional to σ. If the ratio $\|\boldsymbol{y} - \boldsymbol{x}_j\|/\sigma$ is large for all $\boldsymbol{x}_j \in X$, then $\overline{\phi}_k(\boldsymbol{y})$ will be numerically small and hence may not be meaningful. Hence the extension scale σ should be as large as possible.
- On the other hand, for large enough σ, $W_\sigma(X)$ becomes *ill-conditioned*, i.e., μ_k tends to 0 more quickly compared to the case of small σ. Thus the above Nyström extension will blow up.
- Furthermore, it is well known that the extension range depends on the smoothness of the function to be extended: If f is fairly smooth, it can be extended far away from the training set while it has limited extension range if f varies wildly on X.
- In fact, if we can compute M^{-1} *without* blowing up, i.e., $\mu_{N_{\mathrm{tr}}} \gneqq 0$, then $\Phi^{\mathsf{T}} M^{-1} \Phi = W_\sigma(X)^{-1}$.
- Moreover, by setting $\boldsymbol{y} = \boldsymbol{x}_j \in X$ in the Nyström extension formula, we can recover $f(\boldsymbol{x}_j)$:

$$w_\sigma(\boldsymbol{x}_j,:)\Phi^{\mathsf{T}} M^{-1}\Phi \boldsymbol{f} = w_\sigma(\boldsymbol{x}_j,:)W_\sigma(X)^{-1}\boldsymbol{f} = \boldsymbol{e}_j^{\mathsf{T}}\boldsymbol{f} = f(\boldsymbol{x}_j).$$

Geometric Harmonics Multiscale Extension (GHME) . . .

- Observe that the range of the above extension is proportional to σ. If the ratio $\|\boldsymbol{y}-\boldsymbol{x}_j\|/\sigma$ is large for all $\boldsymbol{x}_j \in X$, then $\overline{\phi}_k(\boldsymbol{y})$ will be numerically small and hence may not be meaningful. Hence the extension scale σ should be as large as possible.
- On the other hand, for large enough σ, $W_\sigma(X)$ becomes *ill-conditioned*, i.e., μ_k tends to 0 more quickly compared to the case of small σ. Thus the above Nyström extension will blow up.
- Furthermore, it is well known that the extension range depends on the smoothness of the function to be extended: If f is fairly smooth, it can be extended far away from the training set while it has limited extension range if f varies wildly on X.
- In fact, if we can compute M^{-1} *without* blowing up, i.e., $\mu_{N_{\mathrm{tr}}} \gneqq 0$, then $\Phi^\mathsf{T} M^{-1} \Phi = W_\sigma(X)^{-1}$.
- Moreover, by setting $\boldsymbol{y} = \boldsymbol{x}_j \in X$ in the Nyström extension formula, we can recover $f(\boldsymbol{x}_j)$:

$$w_\sigma(\boldsymbol{x}_j,:)\Phi^\mathsf{T} M^{-1}\Phi \boldsymbol{f} = w_\sigma(\boldsymbol{x}_j,:)W_\sigma(X)^{-1}\boldsymbol{f} = \boldsymbol{e}_j^\mathsf{T}\boldsymbol{f} = f(\boldsymbol{x}_j).$$

Geometric Harmonics Multiscale Extension (GHME) . . .

- Observe that the range of the above extension is proportional to σ. If the ratio $\|\boldsymbol{y}-\boldsymbol{x}_j\|/\sigma$ is large for all $\boldsymbol{x}_j \in X$, then $\overline{\phi}_k(\boldsymbol{y})$ will be numerically small and hence may not be meaningful. Hence the extension scale σ should be as large as possible.
- On the other hand, for large enough σ, $W_\sigma(X)$ becomes *ill-conditioned*, i.e., μ_k tends to 0 more quickly compared to the case of small σ. Thus the above Nyström extension will blow up.
- Furthermore, it is well known that the extension range depends on the smoothness of the function to be extended: If f is fairly smooth, it can be extended far away from the training set while it has limited extension range if f varies wildly on X.
- In fact, if we can compute M^{-1} *without* blowing up, i.e., $\mu_{N_{\mathrm{tr}}} \gneqq 0$, then $\Phi^{\mathsf{T}} M^{-1} \Phi = W_\sigma(X)^{-1}$.
- Moreover, by setting $\boldsymbol{y} = \boldsymbol{x}_j \in X$ in the Nyström extension formula, we can recover $f(\boldsymbol{x}_j)$:

$$w_\sigma(\boldsymbol{x}_j,:)\Phi^{\mathsf{T}} M^{-1}\Phi \boldsymbol{f} = w_\sigma(\boldsymbol{x}_j,:)W_\sigma(X)^{-1}\boldsymbol{f} = \boldsymbol{e}_j^{\mathsf{T}}\boldsymbol{f} = f(\boldsymbol{x}_j).$$

Geometric Harmonics Multiscale Extension (GHME) ...

- In practice, however, $W_\sigma(X)$ is ill-conditioned, and we need to truncate M^{-1} to the first $p \times p$ submatrix and Φ to the first p rows where p, $1 \le p < N_{\mathrm{tr}}$, must be appropriately chosen.
- Let M_p^{-1}, Φ_p be these truncated matrices. Then, the Nyström extension of f can be approximated *without* blowup:

$$\overline{f}(\boldsymbol{y}) \approx w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1}\Phi_p \boldsymbol{f} = w_\sigma(\boldsymbol{y},:)W_{\sigma,p}(X)^{\dagger}\boldsymbol{f},$$

where $W_{\sigma,p}(X)^{\dagger} := \Phi_p^{\mathsf{T}} M_p^{-1}\Phi_p$ is the *pseudoinverse* of $W_\sigma(X)$ using the top p singular values and vectors of $W_\sigma(X)$.
- Hence, if we want to extend a low-dimensional embedding map $\Psi(X) = \left[\boldsymbol{\psi}_1 | \cdots | \boldsymbol{\psi}_s\right]^{\mathsf{T}}$, we have

$$\begin{aligned}
\overline{\Psi}(\boldsymbol{y}) = \left[\overline{\psi}_1(\boldsymbol{y}) | \cdots | \overline{\psi}_s(\boldsymbol{y})\right]^{\mathsf{T}} &\approx \left[w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1}\Phi_p[\boldsymbol{\psi}_1 | \cdots | \boldsymbol{\psi}_s]\right]^{\mathsf{T}} \\
&= \left[w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1}\Phi_p \Psi(X)^{\mathsf{T}}\right]^{\mathsf{T}} \\
&= \Psi(X)\Phi_p^{\mathsf{T}} M_p^{-1}\Phi_p w_\sigma(:,\boldsymbol{y}) \\
&= \Psi(X) W_{\sigma,p}(X)^{\dagger} w_\sigma(:,\boldsymbol{y}).
\end{aligned}$$

Geometric Harmonics Multiscale Extension (GHME) . . .

- In practice, however, $W_\sigma(X)$ is ill-conditioned, and we need to truncate M^{-1} to the first $p \times p$ submatrix and Φ to the first p rows where p, $1 \le p < N_{\mathrm{tr}}$, must be appropriately chosen.
- Let M_p^{-1}, Φ_p be these truncated matrices. Then, the Nyström extension of $\boldsymbol{f}$ can be approximated *without* blowup:

$$\overline{f}(\boldsymbol{y}) \approx w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1}\Phi_p \boldsymbol{f} = w_\sigma(\boldsymbol{y},:)W_{\sigma,p}(X)^{\dagger}\boldsymbol{f},$$

  where $W_{\sigma,p}(X)^{\dagger} := \Phi_p^{\mathsf{T}} M_p^{-1}\Phi_p$ is the *pseudoinverse* of $W_\sigma(X)$ using the top p singular values and vectors of $W_\sigma(X)$.
- Hence, if we want to extend a low-dimensional embedding map $\Psi(X) = \left[\boldsymbol{\psi}_1 \mid \cdots \mid \boldsymbol{\psi}_s\right]^{\mathsf{T}}$, we have

$$\begin{aligned}
\overline{\Psi}(\boldsymbol{y}) = \left[\overline{\psi}_1(\boldsymbol{y}) \mid \cdots \mid \overline{\psi}_s(\boldsymbol{y})\right]^{\mathsf{T}} &\approx \left[w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1}\Phi_p[\boldsymbol{\psi}_1 \mid \cdots \mid \boldsymbol{\psi}_s]\right]^{\mathsf{T}} \\
&= \left[w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1}\Phi_p \Psi(X)^{\mathsf{T}}\right]^{\mathsf{T}} \\
&= \Psi(X)\Phi_p^{\mathsf{T}} M_p^{-1}\Phi_p w_\sigma(:,\boldsymbol{y}) \\
&= \Psi(X) W_{\sigma,p}(X)^{\dagger} w_\sigma(:,\boldsymbol{y}).
\end{aligned}$$

Geometric Harmonics Multiscale Extension (GHME) . . .

- In practice, however, $W_\sigma(X)$ is ill-conditioned, and we need to truncate M^{-1} to the first $p \times p$ submatrix and Φ to the first p rows where p, $1 \le p < N_{\mathrm{tr}}$, must be appropriately chosen.
- Let M_p^{-1}, Φ_p be these truncated matrices. Then, the Nyström extension of $\boldsymbol{f}$ can be approximated *without* blowup:

$$\overline{f}(\boldsymbol{y}) \approx w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1} \Phi_p \boldsymbol{f} = w_\sigma(\boldsymbol{y},:) W_{\sigma,p}(X)^{\dagger} \boldsymbol{f},$$

  where $W_{\sigma,p}(X)^{\dagger} := \Phi_p^{\mathsf{T}} M_p^{-1} \Phi_p$ is the *pseudoinverse* of $W_\sigma(X)$ using the top p singular values and vectors of $W_\sigma(X)$.
- Hence, if we want to extend a low-dimensional embedding map $\Psi(X) = [\boldsymbol{\psi}_1 | \cdots | \boldsymbol{\psi}_s]^{\mathsf{T}}$, we have

$$\begin{aligned} \overline{\Psi}(\boldsymbol{y}) = \left[\overline{\psi}_1(\boldsymbol{y}) | \cdots | \overline{\psi}_s(\boldsymbol{y})\right]^{\mathsf{T}} &\approx \left[w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1} \Phi_p [\boldsymbol{\psi}_1 | \cdots | \boldsymbol{\psi}_s] \right]^{\mathsf{T}} \\ &= \left[w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1} \Phi_p \Psi(X)^{\mathsf{T}} \right]^{\mathsf{T}} \\ &= \Psi(X)\Phi_p^{\mathsf{T}} M_p^{-1} \Phi_p w_\sigma(:,\boldsymbol{y}) \\ &= \Psi(X) W_{\sigma,p}(X)^{\dagger} w_\sigma(:,\boldsymbol{y}). \end{aligned}$$

Geometric Harmonics Multiscale Extension (GHME) ...

- One idea to determine this rank p of the pseudoinverse:

$$p = \operatorname*{arg\,max}_{1 \le k \le N_{\mathrm{tr}}} \left\{ \frac{\mu_1}{\mu_k} \le \eta \right\}.$$

  where $\eta > 0$ is some fixed condition number. In other words, p is the largest possible stable rank of $W_\sigma(X)$ such that the condition number after truncation is bounded from above by η.
- Choice of η hence p is quite subtle and intertwined with the choice of σ: large η may lead to $p = N_{\mathrm{tr}}$, but $\bar{f}$ on X does not approximate f on X well unless σ is set so small that $W_\sigma(X)$ has a stable inverse.
- Such a case, however, is not of our interest because setting σ too small practically disconnects data points in X. In fact, $W_\sigma(X) \to I$ as $\sigma \downarrow 0$ (as long as $\boldsymbol{x}_i \neq \boldsymbol{x}_j$ for all $i \neq j$ in X).
- Yet observe that if σ decreases, $\mu_k \downarrow 0$ more slowly. This allows us to use larger p making $\bar{f}$ a better approximation of f on X.
- Hence the GHME iteratively searches for $\bar{f}$ that approximates f on X with an preset error tolerance $\varrho > 0$ by slowly decreasing σ.

Geometric Harmonics Multiscale Extension (GHME) . . .

- One idea to determine this rank p of the pseudoinverse:
$$p = \underset{1 \le k \le N_{\mathrm{tr}}}{\arg\max} \left\{ \frac{\mu_1}{\mu_k} \le \eta \right\}.$$
where $\eta > 0$ is some fixed condition number. In other words, p is the largest possible stable rank of $W_\sigma(X)$ such that the condition number after truncation is bounded from above by η.
- Choice of η hence p is quite subtle and intertwined with the choice of σ: large η may lead to $p = N_{\mathrm{tr}}$, but $\bar{f}$ on X does not approximate f on X well unless σ is set so small that $W_\sigma(X)$ has a stable inverse.
- Such a case, however, is not of our interest because setting σ too small practically disconnects data points in X. In fact, $W_\sigma(X) \to I$ as $\sigma \downarrow 0$ (as long as $\boldsymbol{x}_i \neq \boldsymbol{x}_j$ for all $i \neq j$ in X).
- Yet observe that if σ decreases, $\mu_k \downarrow 0$ more slowly. This allows us to use larger p making $\bar{f}$ a better approximation of f on X.
- Hence the GHME iteratively searches for $\bar{f}$ that approximates f on X with an preset error tolerance $\rho > 0$ by slowly decreasing σ.

Geometric Harmonics Multiscale Extension (GHME) . . .

- One idea to determine this rank p of the pseudoinverse:
$$p = \underset{1 \le k \le N_{\mathrm{tr}}}{\arg\max} \left\{ \frac{\mu_1}{\mu_k} \le \eta \right\}.$$
where $\eta > 0$ is some fixed condition number. In other words, p is the largest possible stable rank of $W_\sigma(X)$ such that the condition number after truncation is bounded from above by η.
- Choice of η hence p is quite subtle and intertwined with the choice of σ: large η may lead to $p = N_{\mathrm{tr}}$, but $\overline{f}$ on X does not approximate f on X well unless σ is set so small that $W_\sigma(X)$ has a stable inverse.
- Such a case, however, is not of our interest because setting σ too small practically disconnects data points in X. In fact, $W_\sigma(X) \to I$ as $\sigma \downarrow 0$ (as long as $\boldsymbol{x}_i \neq \boldsymbol{x}_j$ for all $i \neq j$ in X).
- Yet observe that if σ decreases, $\mu_k \downarrow 0$ more slowly. This allows us to use larger p making $\overline{f}$ a better approximation of f on X.
- Hence the GHME iteratively searches for $\overline{f}$ that approximates f on X with an preset error tolerance $\rho > 0$ by slowly decreasing σ.

Geometric Harmonics Multiscale Extension (GHME) . . .

- One idea to determine this rank p of the pseudoinverse:
$$p = \operatorname*{arg\,max}_{1\le k\le N_{\mathrm{tr}}} \left\{ \frac{\mu_1}{\mu_k} \le \eta \right\}.$$
where $\eta > 0$ is some fixed condition number. In other words, p is the largest possible stable rank of $W_\sigma(X)$ such that the condition number after truncation is bounded from above by η.
- Choice of η hence p is quite subtle and intertwined with the choice of σ: large η may lead to $p = N_{\mathrm{tr}}$, but $\overline{f}$ on X does not approximate f on X well unless σ is set so small that $W_\sigma(X)$ has a stable inverse.
- Such a case, however, is not of our interest because setting σ too small practically disconnects data points in X. In fact, $W_\sigma(X) \to I$ as $\sigma \downarrow 0$ (as long as $\boldsymbol{x}_i \ne \boldsymbol{x}_j$ for all $i \ne j$ in X).
- Yet observe that if σ decreases, $\mu_k \downarrow 0$ more slowly. This allows us to use larger p making $\overline{f}$ a better approximation of f on X.
- Hence the GHME iteratively searches for $\overline{f}$ that approximates f on X with an preset error tolerance $\varrho > 0$ by slowly decreasing σ.

Geometric Harmonics Multiscale Extension (GHME) . . .

- One idea to determine this rank p of the pseudoinverse:
$$p = \underset{1 \le k \le N_{\mathrm{tr}}}{\arg\max} \left\{ \frac{\mu_1}{\mu_k} \le \eta \right\}.$$
where $\eta > 0$ is some fixed condition number. In other words, p is the largest possible stable rank of $W_\sigma(X)$ such that the condition number after truncation is bounded from above by η.
- Choice of η hence p is quite subtle and intertwined with the choice of σ: large η may lead to $p = N_{\mathrm{tr}}$, but $\overline{f}$ on X does not approximate f on X well unless σ is set so small that $W_\sigma(X)$ has a stable inverse.
- Such a case, however, is not of our interest because setting σ too small practically disconnects data points in X. In fact, $W_\sigma(X) \to I$ as $\sigma \downarrow 0$ (as long as $\boldsymbol{x}_i \ne \boldsymbol{x}_j$ for all $i \ne j$ in X).
- Yet observe that if σ decreases, $\mu_k \downarrow 0$ more slowly. This allows us to use larger p making $\overline{f}$ a better approximation of f on X.
- Hence the GHME iteratively searches for $\overline{f}$ that approximates f on X with an preset error tolerance $\varrho > 0$ by slowly decreasing σ.

Algorithm (The GHME of Lafon, Keller, and Coifman (2006))

Suppose f is a function defined on the training set X and to be extended to a test set Y.

1. *Fix a condition number $\eta > 0$ and an error tolerance $\varrho > 0$. Set the extension scale $\sigma = \sigma_0$ for some large value σ_0.*
2. *Compute the eigendecomposition of $W_\sigma(X)$ and expand f (on the training set X) in this eigenbasis.*
3. *On the training set X, approximate f by $\overline{f}$ using the Nyström extension by finding $p = \arg\max_{1 \le k \le N_{\text{tr}}} \{\mu_1/\mu_k \le \eta\}$. Then compute the approximation error $Err := \left(\sum_{k>p} |\langle f, \phi_k \rangle|^2\right)^{1/2}$. If $Err > \varrho$, set $\sigma \leftarrow \frac{1}{2}\sigma$ and return to Step 2. Otherwise, continue.*
4. *Using the value of p obtained in Step 3, compute the final approximate extension for each $\boldsymbol{y} \in Y$:*

$$\overline{f}(\boldsymbol{y}) \approx w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1} \Phi_p \boldsymbol{f} = w_\sigma(\boldsymbol{y},:) W_{\sigma,p}(X)^{\dagger} \boldsymbol{f}.$$

Algorithm (The GHME of Lafon, Keller, and Coifman (2006))

Suppose f is a function defined on the training set X and to be extended to a test set Y.

1. *Fix a condition number $\eta > 0$ and an error tolerance $\varrho > 0$. Set the extension scale $\sigma = \sigma_0$ for some large value σ_0.*
2. *Compute the eigendecomposition of $W_\sigma(X)$ and expand f (on the training set X) in this eigenbasis.*
3. *On the training set X, approximate f by $\overline{f}$ using the Nyström extension by finding $p = \arg\max_{1 \le k \le N_{\mathrm{tr}}} \{\mu_1/\mu_k \le \eta\}$. Then compute the approximation error $Err := \left(\sum_{k>p} |\langle f, \phi_k \rangle|^2\right)^{1/2}$. If $Err > \varrho$, set $\sigma \leftarrow \frac{1}{2}\sigma$ and return to Step 2. Otherwise, continue.*
4. *Using the value of p obtained in Step 3, compute the final approximate extension for each $\boldsymbol{y} \in Y$:*

$$\overline{f}(\boldsymbol{y}) \approx w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1} \Phi_p \boldsymbol{f} = w_\sigma(\boldsymbol{y},:) W_{\sigma,p}(X)^{\dagger} \boldsymbol{f}.$$

Algorithm (The GHME of Lafon, Keller, and Coifman (2006))

Suppose f is a function defined on the training set X and to be extended to a test set Y.

1. *Fix a condition number $\eta > 0$ and an error tolerance $\varrho > 0$. Set the extension scale $\sigma = \sigma_0$ for some large value σ_0.*
2. *Compute the eigendecomposition of $W_\sigma(X)$ and expand f (on the training set X) in this eigenbasis.*
3. *On the training set X, approximate f by $\overline{f}$ using the Nyström extension by finding $p = \arg\max_{1 \le k \le N_{\mathrm{tr}}} \{\mu_1 / \mu_k \le \eta\}$. Then compute the approximation error $Err := \left(\sum_{k>p} |\langle f, \boldsymbol{\phi}_k \rangle|^2\right)^{1/2}$. If $Err > \varrho$, set $\sigma \leftarrow \frac{1}{2}\sigma$ and return to Step 2. Otherwise, continue.*
4. *Using the value of p obtained in Step 3, compute the final approximate extension for each $\boldsymbol{y} \in Y$:*

$$\overline{f}(\boldsymbol{y}) \approx w_\sigma(\boldsymbol{y}, :)\Phi_p^{\mathsf{T}} M_p^{-1} \Phi_p \boldsymbol{f} = w_\sigma(\boldsymbol{y}, :) W_{\sigma,p}(X)^{\dagger} \boldsymbol{f}.$$

Algorithm (The GHME of Lafon, Keller, and Coifman (2006))

Suppose f is a function defined on the training set X and to be extended to a test set Y.

1. *Fix a condition number $\eta > 0$ and an error tolerance $\varrho > 0$. Set the extension scale $\sigma = \sigma_0$ for some large value σ_0.*
2. *Compute the eigendecomposition of $W_\sigma(X)$ and expand f (on the training set X) in this eigenbasis.*
3. *On the training set X, approximate f by $\overline{f}$ using the Nyström extension by finding $p = \arg\max_{1 \le k \le N_{\mathrm{tr}}} \{\mu_1 / \mu_k \le \eta\}$. Then compute the approximation error $Err := \left(\sum_{k>p} |\langle f, \boldsymbol{\phi}_k \rangle|^2\right)^{1/2}$. If $Err > \varrho$, set $\sigma \leftarrow \frac{1}{2}\sigma$ and return to Step 2. Otherwise, continue.*
4. *Using the value of p obtained in Step 3, compute the final approximate extension for each $\boldsymbol{y} \in Y$:*

$$\overline{f}(\boldsymbol{y}) \approx w_\sigma(\boldsymbol{y},:)\Phi_p^{\mathsf{T}} M_p^{-1} \Phi_p \boldsymbol{f} = w_\sigma(\boldsymbol{y},:) W_{\sigma,p}(X)^{\dagger} \boldsymbol{f}.$$