Graphs in Machine Learning

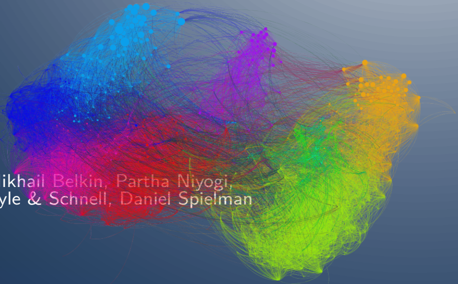
Manifold Learning

Laplacian Eigenmaps

Michal Valko

Inria & ENS Paris-Saclay, MVA

Partially based on material by: Ulrike von Luxburg, Mikhail Belkin, Partha Niyogi, Olivier Chapelle, Bernhard Schölkopf, Gary Miller, Doyle & Schnell, Daniel Spielman



Background: Manifold Learning

problem: definition reduction/manifold learning

Given $\{\mathbf{x}_i\}_{i=1}^{N}$ from $\mathbb{R}^d$ find $\{\mathbf{y}_i\}_{i=1}^{N}$ in $\mathbb{R}^m$, where $m \ll d$.

Background: Manifold Learning

problem: definition reduction/manifold learning

Given $\{\mathbf{x}_i\}_{i=1}^{N}$ from $\mathbb{R}^d$ find $\{\mathbf{y}_i\}_{i=1}^{N}$ in $\mathbb{R}^m$, where $m \ll d$.

- What do we know about the **dimensionality reduction**

Background: Manifold Learning

problem: definition reduction/manifold learning

Given $\{\mathbf{x}_i\}_{i=1}^{N}$ from $\mathbb{R}^d$ find $\{\mathbf{y}_i\}_{i=1}^{N}$ in $\mathbb{R}^m$, where $m \ll d$.

- What do we know about the **dimensionality reduction**
  - representation/visualization (2D or 3D)

Background: Manifold Learning

problem: definition reduction/manifold learning

Given $\{\mathbf{x}_i\}_{i=1}^N$ from $\mathbb{R}^d$ find $\{\mathbf{y}_i\}_{i=1}^N$ in $\mathbb{R}^m$, where $m \ll d$.

- What do we know about the **dimensionality reduction**
  - representation/visualization (2D or 3D)
  - an old example: globe to a map

Background: Manifold Learning

problem: definition reduction/manifold learning

Given $\{\mathbf{x}_i\}_{i=1}^{N}$ from $\mathbb{R}^d$ find $\{\mathbf{y}_i\}_{i=1}^{N}$ in $\mathbb{R}^m$, where $m \ll d$.

- What do we know about the **dimensionality reduction**
  - representation/visualization (2D or 3D)
  - an old example: globe to a map
  - often assuming $\mathcal{M} \subset \mathbb{R}^d$

Background: Manifold Learning

problem: definition reduction/manifold learning

Given $\{\mathbf{x}_i\}_{i=1}^{N}$ from $\mathbb{R}^d$ find $\{\mathbf{y}_i\}_{i=1}^{N}$ in $\mathbb{R}^m$, where $m \ll d$.

- What do we know about the **dimensionality reduction**
  - representation/visualization (2D or 3D)
  - an old example: globe to a map
  - often assuming $\mathcal{M} \subset \mathbb{R}^d$
  - feature extraction

Background: Manifold Learning

problem: definition reduction/manifold learning

Given $\{\mathbf{x}_i\}_{i=1}^N$ from $\mathbb{R}^d$ find $\{\mathbf{y}_i\}_{i=1}^N$ in $\mathbb{R}^m$, where $m \ll d$.

- What do we know about the **dimensionality reduction**
  - representation/visualization (2D or 3D)
  - an old example: globe to a map
  - often assuming $\mathcal{M} \subset \mathbb{R}^d$
  - feature extraction
  - linear vs. nonlinear dimensionality reduction

Background: Manifold Learning

problem: definition reduction/manifold learning

Given $\{\mathbf{x}_i\}_{i=1}^N$ from $\mathbb{R}^d$ find $\{\mathbf{y}_i\}_{i=1}^N$ in $\mathbb{R}^m$, where $m \ll d$.

- What do we know about the **dimensionality reduction**
  - representation/visualization (2D or 3D)
  - an old example: globe to a map
  - often assuming $\mathcal{M} \subset \mathbb{R}^d$
  - feature extraction
  - linear vs. nonlinear dimensionality reduction
- What do we know about linear vs. nonlinear methods?

Background: Manifold Learning

problem: definition reduction/manifold learning

Given $\{\mathbf{x}_i\}_{i=1}^{N}$ from $\mathbb{R}^d$ find $\{\mathbf{y}_i\}_{i=1}^{N}$ in $\mathbb{R}^m$, where $m \ll d$.

- What do we know about the **dimensionality reduction**
  - representation/visualization (2D or 3D)
  - an old example: globe to a map
  - often assuming $\mathcal{M} \subset \mathbb{R}^d$
  - feature extraction
  - linear vs. nonlinear dimensionality reduction
- What do we know about linear vs. nonlinear methods?
  - linear: ICA, PCA, SVD, ...

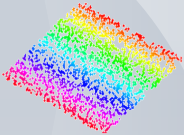
Background: Manifold Learning

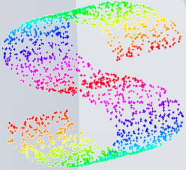
problem: definition reduction/manifold learning

Given $\{\mathbf{x}_i\}_{i=1}^N$ from $\mathbb{R}^d$ find $\{\mathbf{y}_i\}_{i=1}^N$ in $\mathbb{R}^m$, where $m \ll d$.

- What do we know about the **dimensionality reduction**
  - representation/visualization (2D or 3D)
  - an old example: globe to a map
  - often assuming $\mathcal{M} \subset \mathbb{R}^d$
  - feature extraction
  - linear vs. nonlinear dimensionality reduction
- What do we know about linear vs. nonlinear methods?
  - linear: ICA, PCA, SVD, ...
  - nonlinear often preserve only **local** distances

Manifold Learning: Linear vs. Non-linear

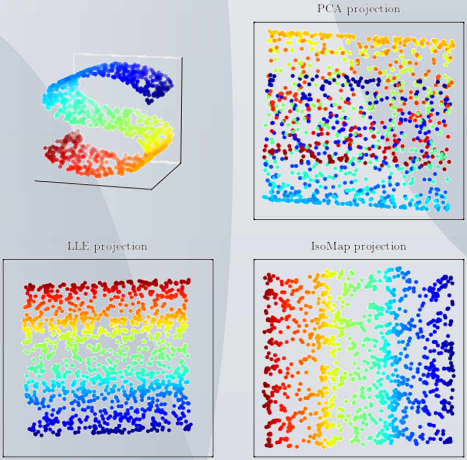




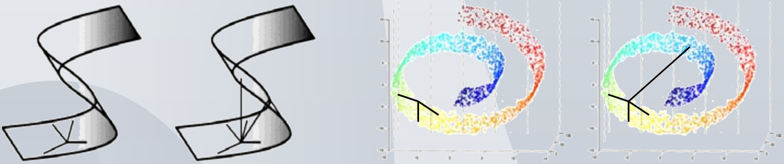
Source: Belkin & Niyogi (2003)

Manifold Learning: Linear vs. Non-linear (Alternative View)

What do we know about linear vs. nonlinear methods?

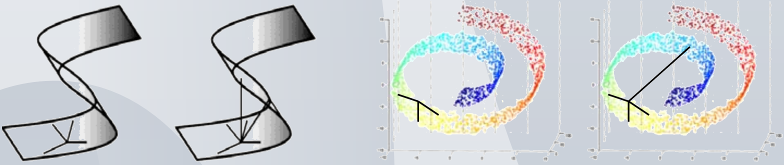


Manifold Learning: Preserving (just) local distances



Source: Belkin & Niyogi (2003)

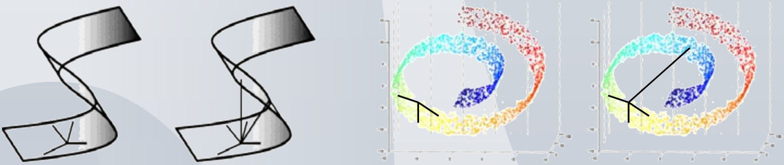
Manifold Learning: Preserving (just) local distances



Source: Belkin & Niyogi (2003)

$$d(\mathbf{y}_i, \mathbf{y}_j) = d(\mathbf{x}_i, \mathbf{x}_j) \quad \text{only if} \quad d(\mathbf{x}_i, \mathbf{x}_j) \quad \text{is small}$$

Manifold Learning: Preserving (just) local distances

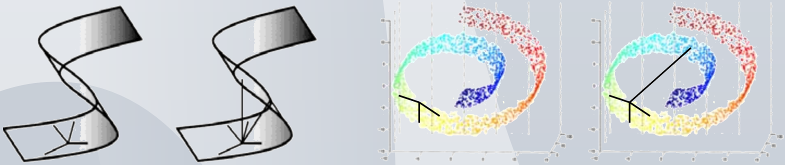


Source: Belkin & Niyogi (2003)

$$d(\mathbf{y}_i, \mathbf{y}_j) = d(\mathbf{x}_i, \mathbf{x}_j) \quad \text{only if} \quad d(\mathbf{x}_i, \mathbf{x}_j) \quad \text{is small}$$

1-D $$\min_y \sum_{ij} w_{ij}(y_i - y_j)^2$$

Manifold Learning: Preserving (just) local distances



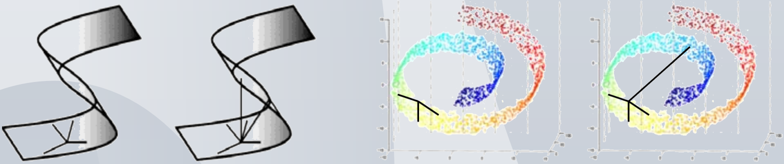
Source: Belkin & Niyogi (2003)

$$d(\mathbf{y}_i, \mathbf{y}_j) = d(\mathbf{x}_i, \mathbf{x}_j) \quad \text{only if} \quad d(\mathbf{x}_i, \mathbf{x}_j) \quad \text{is small}$$

1-D $$\min_{y} \sum_{ij} w_{ij}(y_i - y_j)^2$$

m-D $$\min_{\mathbf{y}} \sum_{ij} w_{ij}\|\mathbf{y}_i - \mathbf{y}_j\|^2$$

Manifold Learning: Preserving (just) local distances



Source: Belkin & Niyogi (2003)

$$d(\mathbf{y}_i, \mathbf{y}_j) = d(\mathbf{x}_i, \mathbf{x}_j) \quad \text{only if} \quad d(\mathbf{x}_i, \mathbf{x}_j) \quad \text{is small}$$

1-D $\qquad \min_{y} \sum_{ij} w_{ij}(y_i - y_j)^2$

m-D $\qquad \min_{\mathbf{y}} \sum_{ij} w_{ij}\|\mathbf{y}_i - \mathbf{y}_j\|^2$

Looks familiar?

Manifold Learning: Laplacian Eigenmaps

Step 1: Solve generalized eigenproblem:

$$\mathbf{Lf} = \lambda \mathbf{Df}$$

Manifold Learning: Laplacian Eigenmaps

Step 1: Solve generalized eigenproblem:

$$\mathbf{L}\mathbf{f} = \lambda \mathbf{D}\mathbf{f}$$

Step 2: Assign m new coordinates:

$$\mathbf{x}_i \mapsto \left(f_2\left(i\right), \ldots, f_{m+1}\left(i\right)\right) = \mathbf{y}_i$$

Manifold Learning: Laplacian Eigenmaps

Step 1: Solve generalized eigenproblem:

$$\mathbf{L}\mathbf{f} = \lambda\mathbf{D}\mathbf{f}$$

Step 2: Assign m new coordinates:

$$\mathbf{x}_i \mapsto \left(f_2\left(i\right), \ldots, f_{m+1}\left(i\right)\right) = \mathbf{y}_i$$

Note$_1$: we need to get $m+1$ smallest eigenvectors

Manifold Learning: Laplacian Eigenmaps

Step 1: Solve generalized eigenproblem:

$$\mathbf{Lf} = \lambda \mathbf{Df}$$

Step 2: Assign m new coordinates:

$$\mathbf{x}_i \mapsto \left(f_2\left(i\right), \ldots, f_{m+1}\left(i\right)\right) = \mathbf{y}_i$$

Note$_1$: we need to get $m + 1$ smallest eigenvectors
Note$_2$: $\mathbf{f}_1$ is useless

http://web.cse.ohio-state.edu/~mbelkin/papers/LEM_NC_03.pdf

Manifold Learning: Laplacian Eigenmaps to 1D

Laplacian Eigenmaps 1D objective

$$\min_{\mathbf{f}} \mathbf{f}^\mathsf{T}\mathbf{L}\mathbf{f} \quad \text{s.t.} \quad f_i \in \mathbb{R}, \quad \mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{1} = 0, \quad \mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{f} = \mathbf{1}$$

Manifold Learning: Laplacian Eigenmaps to 1D

Laplacian Eigenmaps 1D objective

$$\min_{\mathbf{f}} \mathbf{f}^\mathsf{T}\mathbf{L}\mathbf{f} \quad \text{s.t.} \quad f_i \in \mathbb{R}, \quad \mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{1} = 0, \quad \mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{f} = \mathbf{1}$$

The meaning of the constraints is similar as for spectral clustering:

$\mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{f} = \mathbf{1}$ is for

Manifold Learning: Laplacian Eigenmaps to 1D

Laplacian Eigenmaps 1D objective

$$\min_{\mathbf{f}} \mathbf{f}^\mathsf{T}\mathbf{L}\mathbf{f} \quad \text{s.t.} \quad f_i \in \mathbb{R}, \quad \mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{1} = 0, \quad \mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{f} = \mathbf{1}$$

The meaning of the constraints is similar as for spectral clustering:

$\mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{f} = \mathbf{1}$ is for scaling

Manifold Learning: Laplacian Eigenmaps to 1D

Laplacian Eigenmaps 1D objective

$$\min_{\mathbf{f}} \mathbf{f}^{\mathsf{T}}\mathbf{L}\mathbf{f} \quad \text{s.t.} \quad f_i \in \mathbb{R}, \quad \mathbf{f}^{\mathsf{T}}\mathbf{D}\mathbf{1} = 0, \quad \mathbf{f}^{\mathsf{T}}\mathbf{D}\mathbf{f} = \mathbf{1}$$

The meaning of the constraints is similar as for spectral clustering:

$\mathbf{f}^{\mathsf{T}}\mathbf{D}\mathbf{f} = \mathbf{1}$ is for scaling

$\mathbf{f}^{\mathsf{T}}\mathbf{D}\mathbf{1} = 0$ is to

Manifold Learning: Laplacian Eigenmaps to 1D

Laplacian Eigenmaps 1D objective

$$\min_{\mathbf{f}} \mathbf{f}^\mathsf{T}\mathbf{L}\mathbf{f} \quad \text{s.t.} \quad f_i \in \mathbb{R}, \quad \mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{1} = 0, \quad \mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{f} = \mathbf{1}$$

The meaning of the constraints is similar as for spectral clustering:

$\mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{f} = \mathbf{1}$ is for scaling

$\mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{1} = 0$ is to not get $\mathbf{v}_1$

Manifold Learning: Laplacian Eigenmaps to 1D

Laplacian Eigenmaps 1D objective

$$\min_{\mathbf{f}} \mathbf{f}^\mathsf{T}\mathbf{L}\mathbf{f} \quad \text{s.t.} \quad f_i \in \mathbb{R}, \quad \mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{1} = 0, \quad \mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{f} = \mathbf{1}$$

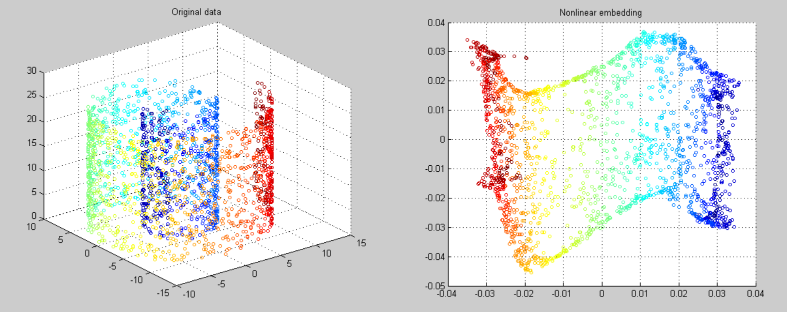
The meaning of the constraints is similar as for spectral clustering:

$\mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{f} = \mathbf{1}$ is for scaling

$\mathbf{f}^\mathsf{T}\mathbf{D}\mathbf{1} = 0$ is to not get $\mathbf{v}_1$

What is the solution?

Manifold Learning: Example



Source: Belkin & Niyogi (2003); MATLAB implementation: http://www.mathworks.com/matlabcentral/fileexchange/36141-laplacian-eigenmap-~-diffusion-map-~-manifold-learning

Michal Valko

michal.valko@inria.fr

Inria & ENS Paris-Saclay, MVA

https://misovalko.github.io/mva-ml-graphs.html