Spectral Dimensionality Reduction via Maximum Entropy

Neil D. Lawrence

Departments of Neuro- and Computer Science, University of Sheffield, U.K.
AISTATS 2011, Fort Lauderdale, FL

13th April 2011

Outline

Outline

Spectral Approaches

- Assume data is given in the form of interpoint squared distances.

$$d_{i,j} = \|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\|_2^2 = \mathbf{y}_{i,:}^\top \mathbf{y}_{i,:} - 2\mathbf{y}_{i,:}^\top \mathbf{y}_{j,:} + \mathbf{y}_{j,:}^\top \mathbf{y}_{j,:}.$$

- Classical MDS: find *linear* embedding which approximates distance matrix $\mathbf{D}$ (Mardia et al., 1979).
  - it provides a linear transformation between $\mathbf{X}$ (latent space) and $\mathbf{Y}$ (data space).
- Spectral approaches in machine learning give a *nonlinear* relationship between the data and the distances.
- This is done by not computing $\mathbf{D}$ directly in the space of $\mathbf{Y}$.
- Example: kernel PCA, where $\mathbf{D}$ is computed in a feature space derived from $\mathbf{Y}$,

$$d_{i,j} = k_{i,i} - 2k_{i,j} + k_{j,j}.$$

Spectral Approaches

- Assume data is given in the form of interpoint squared distances.

$$d_{i,j} = \|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\|_2^2 = \mathbf{y}_{i,:}^\top \mathbf{y}_{i,:} - 2\mathbf{y}_{i,:}^\top \mathbf{y}_{j,:} + \mathbf{y}_{j,:}^\top \mathbf{y}_{j,:}.$$

- Classical MDS: find *linear* embedding which approximates distance matrix **D** (Mardia et al., 1979).
  - it provides a linear transformation between **X** (latent space) and **Y** (data space).
- Spectral approaches in machine learning give a *nonlinear* relationship between the data and the distances.
- This is done by not computing **D** directly in the space of **Y**.
- Example: kernel PCA, where **D** is computed in a feature space derived from **Y**,

$$d_{i,j} = k_{i,i} - 2k_{i,j} + k_{j,j}.$$

Spectral Approaches

- Assume data is given in the form of interpoint squared distances.

$$d_{i,j} = \|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\|_2^2 = \mathbf{y}_{i,:}^\top \mathbf{y}_{i,:} - 2\mathbf{y}_{i,:}^\top \mathbf{y}_{j,:} + \mathbf{y}_{j,:}^\top \mathbf{y}_{j,:}.$$

- Classical MDS: find *linear* embedding which approximates distance matrix $\mathbf{D}$ (Mardia et al., 1979).
  - it provides a linear transformation between $\mathbf{X}$ (latent space) and $\mathbf{Y}$ (data space).
- Spectral approaches in machine learning give a *nonlinear* relationship between the data and the distances.
- This is done by not computing $\mathbf{D}$ directly in the space of $\mathbf{Y}$.
- Example: kernel PCA, where $\mathbf{D}$ is computed in a feature space derived from $\mathbf{Y}$,

$$d_{i,j} = k_{i,i} - 2k_{i,j} + k_{j,j}.$$

Spectral Approaches

- Assume data is given in the form of interpoint squared distances.

$$d_{i,j} = \|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\|_2^2 = \mathbf{y}_{i,:}^\top \mathbf{y}_{i,:} - 2\mathbf{y}_{i,:}^\top \mathbf{y}_{j,:} + \mathbf{y}_{j,:}^\top \mathbf{y}_{j,:}.$$

- Classical MDS: find *linear* embedding which approximates distance matrix $\mathbf{D}$ (Mardia et al., 1979).
  - it provides a linear transformation between $\mathbf{X}$ (latent space) and $\mathbf{Y}$ (data space).
- Spectral approaches in machine learning give a *nonlinear* relationship between the data and the distances.
- This is done by not computing $\mathbf{D}$ directly in the space of $\mathbf{Y}$.
- Example: kernel PCA, where $\mathbf{D}$ is computed in a feature space derived from $\mathbf{Y}$,

$$d_{i,j} = k_{i,i} - 2k_{i,j} + k_{j,j}.$$

Spectral Approaches

- Assume data is given in the form of interpoint squared distances.

$$d_{i,j} = \|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\|_2^2 = \mathbf{y}_{i,:}^\top \mathbf{y}_{i,:} - 2\mathbf{y}_{i,:}^\top \mathbf{y}_{j,:} + \mathbf{y}_{j,:}^\top \mathbf{y}_{j,:}.$$

- Classical MDS: find *linear* embedding which approximates distance matrix $\mathbf{D}$ (Mardia et al., 1979).
  - it provides a linear transformation between $\mathbf{X}$ (latent space) and $\mathbf{Y}$ (data space).
- Spectral approaches in machine learning give a *nonlinear* relationship between the data and the distances.
- This is done by not computing $\mathbf{D}$ directly in the space of $\mathbf{Y}$.
- Example: kernel PCA, where $\mathbf{D}$ is computed in a feature space derived from $\mathbf{Y}$,

$$d_{i,j} = k_{i,i} - 2k_{i,j} + k_{j,j}.$$

Classical MDS and KPCA

- CMDS procedure performs eigenvalue problem on centered kernel matrix.

$$\mathbf{B} = \mathbf{HKH}.$$

$$(\text{equivalently } \mathbf{B} = -\tfrac{1}{2}\mathbf{HDH})$$

- This matches the KPCA algorithm (Schölkopf et al., 1998).
- **However**, for the commonly used exponentiated quadratic kernel,

$$k(y_{i,:}, y_{j,:}) = \exp(-\gamma \left\|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right\|_2^2),$$

KPCA actually *expands* the feature space (Weinberger et al., 2004).

Classical MDS and KPCA

- CMDS procedure performs eigenvalue problem on centered kernel matrix.

$$\mathbf{B} = \mathbf{HKH}.$$

$$\text{(equivalently } \mathbf{B} = -\tfrac{1}{2}\mathbf{HDH}\text{)}$$

- This matches the KPCA algorithm (Schölkopf et al., 1998).
- **However**, for the commonly used exponentiated quadratic kernel,

$$k(y_{i,:}, y_{j,:}) = \exp(-\gamma \left\|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right\|_2^2),$$

KPCA actually *expands* the feature space (Weinberger et al., 2004).

Classical MDS and KPCA

- CMDS procedure performs eigenvalue problem on centered kernel matrix.
$$\mathbf{B} = \mathbf{HKH}.$$
$$(\text{equivalently } \mathbf{B} = -\tfrac{1}{2}\mathbf{HDH})$$
- This matches the KPCA algorithm (Schölkopf et al., 1998).
- **However**, for the commonly used exponentiated quadratic kernel,
$$k(y_{i,:}, y_{j,:}) = \exp(-\gamma \left\| \mathbf{y}_{i,:} - \mathbf{y}_{j,:} \right\|_2^2),$$
KPCA actually *expands* the feature space (Weinberger et al., 2004).

Maximum Variance Unfolding

Learn a "Kernel" for Dimensionality Reduction

- MVU (Weinberger et al., 2004): learn a "kernel matrix" that will allow for dimensionality reduction.
- Preserve only *local* proximity relationships in the data.
  - Take a set of neighbors.
  - Construct a kernel matrix where only distances between neighbors match data distances.

Maximum Variance Unfolding

Learn a "Kernel" for Dimensionality Reduction

- MVU (Weinberger et al., 2004): learn a "kernel matrix" that will allow for dimensionality reduction.
- Preserve only *local* proximity relationships in the data.
  - Take a set of neighbors.
  - Construct a kernel matrix where only distances between neighbors match data distances.

Maximum Variance Unfolding

Learn a "Kernel" for Dimensionality Reduction

- MVU (Weinberger et al., 2004): learn a "kernel matrix" that will allow for dimensionality reduction.
- Preserve only *local* proximity relationships in the data.
  - Take a set of neighbors.
  - Construct a kernel matrix where only distances between neighbors match data distances.

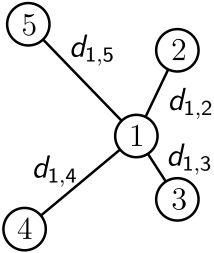
Maximum Variance Unfolding

Learn a "Kernel" for Dimensionality Reduction

- MVU (Weinberger et al., 2004): learn a "kernel matrix" that will allow for dimensionality reduction.
- Preserve only *local* proximity relationships in the data.
  - Take a set of neighbors.
  - Construct a kernel matrix where only distances between neighbors match data distances.

Maximum Variance Unfolding

- Optimize elements of $\mathbf{K}$ by maximizing $\mathrm{tr}(\mathbf{K})$ (total variance of data).



- Subject to distance constraints between neighbors

$$d_{i,j} = k_{i,i} - 2k_{i,j} + k_{j,j}$$

Maximum Entropy Unfolding

Our Contribution

- Maximize *entropy* instead of variance (Jaynes, 1986): MEU.
- Entropy and variance both measure uncertainty.
- Maximum entropy leads to a *probabilistic model*.
- The model relates several different spectral approaches.

Maximum Entropy Unfolding

Our Contribution

- Maximize *entropy* instead of variance (Jaynes, 1986): MEU.
- Entropy and variance both measure uncertainty.
- Maximum entropy leads to a *probabilistic model*.
- The model relates several different spectral approaches.

Maximum Entropy Unfolding

Our Contribution

- Maximize *entropy* instead of variance (Jaynes, 1986): MEU.
- Entropy and variance both measure uncertainty.
- Maximum entropy leads to a *probabilistic model*.
- The model relates several different spectral approaches.

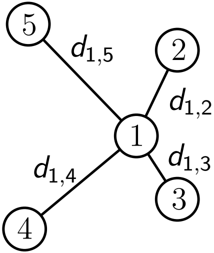
Maximum Entropy Unfolding

Our Contribution

- Maximize *entropy* instead of variance (Jaynes, 1986): MEU.
- Entropy and variance both measure uncertainty.
- Maximum entropy leads to a *probabilistic model*.
- The model relates several different spectral approaches.

Maximum Entropy Unfolding

- Find distribution with maximum entropy subject to constraints on *moments*.

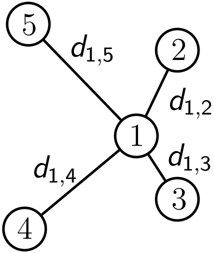


- MEU constraints are on expected distances between neighbors.

$$d_{i,j} = \left\langle \mathbf{y}_{i,:}^\top \mathbf{y}_{i,:} \right\rangle - 2 \left\langle \mathbf{y}_{i,:}^\top \mathbf{y}_{j,:} \right\rangle + \left\langle \mathbf{y}_{j,:}^\top \mathbf{y}_{j,:} \right\rangle$$

Maximum Entropy Unfolding

- Find distribution with maximum entropy subject to constraints on *moments*.



- MEU constraints are on expected distances between neighbors.

$$d_{i,j} = k_{i,i} - 2k_{i,j} + k_{j,j}$$

which can be written in terms of the covariance.

Gaussian Random Field

- The maximum entropy probability distribution is a *Gaussian random field*

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Covariance matrix is

$$\mathbf{K} = (\mathbf{L} + \gamma\mathbf{I})^{-1}.$$

- Where $\mathbf{L}$ is the *Laplacian* matrix associated with the neighborhood graph.
- Off diagonal elements of the Laplacian are Lagrange multipliers from moment constraints.
- On diagonal elements given by negative sum of off-diagonal ($\mathbf{L}\mathbf{1} = \mathbf{0}$).

Gaussian Random Field

- The maximum entropy probability distribution is a *Gaussian random field*

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Covariance matrix is

$$\mathbf{K} = (\mathbf{L} + \gamma\mathbf{I})^{-1}.$$

- Where $\mathbf{L}$ is the *Laplacian* matrix associated with the neighborhood graph.
- Off diagonal elements of the Laplacian are Lagrange multipliers from moment constraints.
- On diagonal elements given by negative sum of off-diagonal ($\mathbf{L1} = \mathbf{0}$).

Gaussian Random Field

- The maximum entropy probability distribution is a *Gaussian random field*

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Covariance matrix is

$$\mathbf{K} = (\mathbf{L} + \gamma\mathbf{I})^{-1}.$$

- Where **L** is the *Laplacian* matrix associated with the neighborhood graph.
- Off diagonal elements of the Laplacian are Lagrange multipliers from moment constraints.
- On diagonal elements given by negative sum of off-diagonal ($\mathbf{L1} = \mathbf{0}$).

Gaussian Random Field

- The maximum entropy probability distribution is a *Gaussian random field*

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Covariance matrix is

$$\mathbf{K} = (\mathbf{L} + \gamma\mathbf{I})^{-1}.$$

- Where $\mathbf{L}$ is the *Laplacian* matrix associated with the neighborhood graph.
- Off diagonal elements of the Laplacian are Lagrange multipliers from moment constraints.
- On diagonal elements given by negative sum of off-diagonal ($\mathbf{L1} = \mathbf{0}$).

Gaussian Random Field

- The maximum entropy probability distribution is a *Gaussian random field*

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^\top \mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Covariance matrix is

$$\mathbf{K} = (\mathbf{L} + \gamma\mathbf{I})^{-1}.$$

- Where $\mathbf{L}$ is the *Laplacian* matrix associated with the neighborhood graph.
- Off diagonal elements of the Laplacian are Lagrange multipliers from moment constraints.
- On diagonal elements given by negative sum of off-diagonal ($\mathbf{L1} = \mathbf{0}$).

Data Feature Independence

- The GRF specifying independence across data *features*.
- Most applications of Gaussian models are applied independently across data *points*.
  - Notable exceptions include Zhu et al. (2003); Lawrence (2004, 2005); Kemp and Tenenbaum (2008).
- Maximum likelihood in this model is equivalent maximizing entropy under distance constraints.

Blessing of Dimensionality

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Maximum likelihood is consistent: (see e.g. Wasserman, 2003, pg 126)
  - As we increase data points parameters become better determined.
  - **Not** in this model.
  - As we increase data features parameters become better determined.
- This turns the large p small n problem on its head.
- There is a "Blessing of Dimensionality" in this model.

Blessing of Dimensionality

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Maximum likelihood is consistent: (see e.g. Wasserman, 2003, pg 126)
  - As we increase data points parameters become better determined.
  - **Not** in this model.
  - As we increase data features parameters become better determined.
- This turns the large p small n problem on its head.
- There is a "Blessing of Dimensionality" in this model.

Blessing of Dimensionality

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Maximum likelihood is consistent: (see e.g. Wasserman, 2003, pg 126)
  - As we increase data points parameters become better determined.
  - **Not** in this model.
  - As we increase data features parameters become better determined.
- This turns the large p small n problem on its head.
- There is a "Blessing of Dimensionality" in this model.

Blessing of Dimensionality

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Maximum likelihood is consistent: (see e.g. Wasserman, 2003, pg 126)
  - As we increase data points parameters become better determined.
  - **Not** in this model.
  - As we increase data features parameters become better determined.
- This turns the large p small n problem on its head.
- There is a "Blessing of Dimensionality" in this model.

Blessing of Dimensionality

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Maximum likelihood is consistent: (see e.g. Wasserman, 2003, pg 126)
  - As we increase data points parameters become better determined.
  - **Not** in this model.
  - As we increase data features parameters become better determined.
- This turns the large p small n problem on its head.
- There is a "Blessing of Dimensionality" in this model.

Blessing of Dimensionality

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Maximum likelihood is consistent: (see e.g. Wasserman, 2003, pg 126)
  - As we increase data points parameters become better determined.
  - **Not** in this model.
  - As we increase data features parameters become better determined.
- This turns the large p small n problem on its head.
- There is a "Blessing of Dimensionality" in this model.

Blessing of Dimensionality

$$p(\mathbf{Y}) = \prod_{j=1}^{p} \frac{1}{|\mathbf{K}|^{\frac{1}{2}} (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{K}^{-1}\mathbf{y}_{:,j}\right),$$

- Maximum likelihood is consistent: (see e.g. Wasserman, 2003, pg 126)
  - As we increase data points parameters become better determined.
  - **Not** in this model.
  - As we increase data features parameters become better determined.
- This turns the large p small n problem on its head.
- There is a "Blessing of Dimensionality" in this model.

Outline

Relationship to Laplacian Eigenmaps

- Laplacian eigenmaps (Belkin and Niyogi, 2003): graph Laplacian is specified across the data points.
- Laplacian has exactly the same form as our matrix $\mathbf{L}$.
- Parameters of the Laplacian are set either as constant or according to the distance between two points.
- Smallest eigenvectors of this Laplacian are then used for visualizing the data (discarding constant eigenvector).
- This operation is equivalent to taking largest eigenvectors of $\mathbf{HKH}$.
- Laplacian eigenmaps do not preserve distances between neighbors.

Relationship to Laplacian Eigenmaps

- Laplacian eigenmaps (Belkin and Niyogi, 2003): graph Laplacian is specified across the data points.
- Laplacian has exactly the same form as our matrix $\mathbf{L}$.
- Parameters of the Laplacian are set either as constant or according to the distance between two points.
- Smallest eigenvectors of this Laplacian are then used for visualizing the data (discarding constant eigenvector).
- This operation is equivalent to taking largest eigenvectors of $\mathbf{HKH}$.
- Laplacian eigenmaps do not preserve distances between neighbors.

Relationship to Laplacian Eigenmaps

- Laplacian eigenmaps (Belkin and Niyogi, 2003): graph Laplacian is specified across the data points.
- Laplacian has exactly the same form as our matrix $\mathbf{L}$.
- Parameters of the Laplacian are set either as constant or according to the distance between two points.
- Smallest eigenvectors of this Laplacian are then used for visualizing the data (discarding constant eigenvector).
- This operation is equivalent to taking largest eigenvectors of $\mathbf{HKH}$.
- Laplacian eigenmaps do not preserve distances between neighbors.

Relationship to Laplacian Eigenmaps

- Laplacian eigenmaps (Belkin and Niyogi, 2003): graph Laplacian is specified across the data points.
- Laplacian has exactly the same form as our matrix $\mathbf{L}$.
- Parameters of the Laplacian are set either as constant or according to the distance between two points.
- Smallest eigenvectors of this Laplacian are then used for visualizing the data (discarding constant eigenvector).
- This operation is equivalent to taking largest eigenvectors of $\mathbf{HKH}$.
- Laplacian eigenmaps do not preserve distances between neighbors.

Relationship to Laplacian Eigenmaps

- Laplacian eigenmaps (Belkin and Niyogi, 2003): graph Laplacian is specified across the data points.
- Laplacian has exactly the same form as our matrix $\mathbf{L}$.
- Parameters of the Laplacian are set either as constant or according to the distance between two points.
- Smallest eigenvectors of this Laplacian are then used for visualizing the data (discarding constant eigenvector).
- This operation is equivalent to taking largest eigenvectors of $\mathbf{HKH}$.
- Laplacian eigenmaps do not preserve distances between neighbors.

Relationship to Laplacian Eigenmaps

- Laplacian eigenmaps (Belkin and Niyogi, 2003): graph Laplacian is specified across the data points.
- Laplacian has exactly the same form as our matrix $\mathbf{L}$.
- Parameters of the Laplacian are set either as constant or according to the distance between two points.
- Smallest eigenvectors of this Laplacian are then used for visualizing the data (discarding constant eigenvector).
- This operation is equivalent to taking largest eigenvectors of $\mathbf{HKH}$.
- Laplacian eigenmaps do not preserve distances between neighbors.

Locally Linear Embedding

- The Laplacian should be constrained positive definite.
- This constraint can be imposed by factorizing it as

$$\mathbf{L} = \mathbf{M}\mathbf{M}^\top$$

- To ensure it is a Laplacian, we can constrain $\mathbf{M}^\top\mathbf{1} = \mathbf{0}$ giving $\mathbf{L}\mathbf{1} = \mathbf{0}$.
  - i.e. $m_{i,i} = -\sum_{j\in\mathcal{N}(i)} m_{j,i}$
  - Set $m_{j,i} = 0$ if $j \notin \mathcal{N}(i)$.

Locally Linear Embedding

- The Laplacian should be constrained positive definite.
- This constraint can be imposed by factorizing it as

$$\mathbf{L} = \mathbf{M}\mathbf{M}^\top$$

- To ensure it is a Laplacian, we can constrain $\mathbf{M}^\top\mathbf{1} = \mathbf{0}$ giving $\mathbf{L}\mathbf{1} = \mathbf{0}$.
  - i.e. $m_{i,i} = -\sum_{j\in\mathcal{N}(i)} m_{j,i}$
  - Set $m_{j,i} = 0$ if $j \notin \mathcal{N}(i)$.

Locally Linear Embedding

- The Laplacian should be constrained positive definite.
- This constraint can be imposed by factorizing it as

$$\mathbf{L} = \mathbf{M}\mathbf{M}^\top$$

- To ensure it is a Laplacian, we can constrain $\mathbf{M}^\top\mathbf{1} = \mathbf{0}$ giving $\mathbf{L}\mathbf{1} = \mathbf{0}$.
  - i.e. $m_{i,i} = -\sum_{j\in\mathcal{N}(i)} m_{j,i}$
  - Set $m_{j,i} = 0$ if $j \notin \mathcal{N}(i)$.

Locally Linear Embedding

- The Laplacian should be constrained positive definite.
- This constraint can be imposed by factorizing it as

$$\mathbf{L} = \mathbf{M}\mathbf{M}^\top$$

- To ensure it is a Laplacian, we can constrain $\mathbf{M}^\top\mathbf{1} = \mathbf{0}$ giving $\mathbf{L}\mathbf{1} = \mathbf{0}$.
  - i.e. $m_{i,i} = -\sum_{j\in\mathcal{N}(i)} m_{j,i}$
  - Set $m_{j,i} = 0$ if $j \notin \mathcal{N}(i)$.

Locally Linear Embedding

- The Laplacian should be constrained positive definite.
- This constraint can be imposed by factorizing it as

$$\mathbf{L} = \mathbf{M}\mathbf{M}^\top$$

- To ensure it is a Laplacian, we can constrain $\mathbf{M}^\top \mathbf{1} = \mathbf{0}$ giving $\mathbf{L}\mathbf{1} = \mathbf{0}$.
  - i.e. $m_{i,i} = -\sum_{j \in \mathcal{N}(i)} m_{j,i}$
  - Set $m_{j,i} = 0$ if $j \notin \mathcal{N}(i)$.

Locally Linear Embedding

- Locally linear embeddings (Roweis and Saul, 2000) are then a specific case of MEU where
  1. The diagonal sums, $m_{i,i}$, are further constrained to unity.
  2. Model parameters found by maximizing *pseudolikelihood* of the data.

Locally Linear Embedding

- Locally linear embeddings (Roweis and Saul, 2000) are then a specific case of MEU where
  1. The diagonal sums, $m_{i,i}$, are further constrained to unity.
  2. Model parameters found by maximizing *pseudolikelihood* of the data.

Locally Linear Embedding

- Locally linear embeddings (Roweis and Saul, 2000) are then a specific case of MEU where
  1. The diagonal sums, $m_{i,i}$, are further constrained to unity.
  2. Model parameters found by maximizing *pseudolikelihood* of the data.

LLE Approximates MEU

- LLE is an approximation to maximum likelihood.
- Laplacian has factorized form.
- Pseudolikelihood also allows for relatively quick parameter estimation.
  - ignoring the partition function removes the need to invert to recover the covariance matrix.
  - LLE can be applied to larger data sets than MEU or MVU.

Note: The sparsity pattern in the Laplacian for LLE will not match that used in the Laplacian for the other algorithms due to the factorized representation.

LLE Approximates MEU

- LLE is an approximation to maximum likelihood.
- Laplacian has factorized form.
- Pseudolikelihood also allows for relatively quick parameter estimation.
  - ignoring the partition function removes the need to invert to recover the covariance matrix.
  - LLE can be applied to larger data sets than MEU or MVU.

Note: The sparsity pattern in the Laplacian for LLE will not match that used in the Laplacian for the other algorithms due to the factorized representation.

LLE Approximates MEU

- LLE is an approximation to maximum likelihood.
- Laplacian has factorized form.
- Pseudolikelihood also allows for relatively quick parameter estimation.
  - ignoring the partition function removes the need to invert to recover the covariance matrix.
  - LLE can be applied to larger data sets than MEU or MVU.

Note: The sparsity pattern in the Laplacian for LLE will not match that used in the Laplacian for the other algorithms due to the factorized representation.

LLE Approximates MEU

- LLE is an approximation to maximum likelihood.
- Laplacian has factorized form.
- Pseudolikelihood also allows for relatively quick parameter estimation.
  - ignoring the partition function removes the need to invert to recover the covariance matrix.
  - LLE can be applied to larger data sets than MEU or MVU.

Note: The sparsity pattern in the Laplacian for LLE will not match that used in the Laplacian for the other algorithms due to the factorized representation.

LLE Approximates MEU

- LLE is an approximation to maximum likelihood.
- Laplacian has factorized form.
- Pseudolikelihood also allows for relatively quick parameter estimation.
  - ignoring the partition function removes the need to invert to recover the covariance matrix.
  - LLE can be applied to larger data sets than MEU or MVU.

Note: The sparsity pattern in the Laplacian for LLE will not match that used in the Laplacian for the other algorithms due to the factorized representation.

LLE Approximates MEU

- LLE is an approximation to maximum likelihood.
- Laplacian has factorized form.
- Pseudolikelihood also allows for relatively quick parameter estimation.
  - ignoring the partition function removes the need to invert to recover the covariance matrix.
  - LLE can be applied to larger data sets than MEU or MVU.

Note: The sparsity pattern in the Laplacian for LLE will not match that used in the Laplacian for the other algorithms due to the factorized representation.

LLE and PCA

- LLE is motivated by considering local linear embeddings of the data.
- Interestingly, as we increase the neighborhood size to $K = n - 1$ we do not recover PCA.
- But PCA is the "optimal" linear embedding!!
- LLE is optimizing a pseudolikelihood: in contrast the MEU algorithm, which LLE approximates, does recover PCA when $K = n - 1$.

LLE and PCA

- LLE is motivated by considering local linear embeddings of the data.
- Interestingly, as we increase the neighborhood size to $K = n - 1$ we do not recover PCA.
- But PCA is the "optimal" linear embedding!!
- LLE is optimizing a pseudolikelihood: in contrast the MEU algorithm, which LLE approximates, does recover PCA when $K = n - 1$.

LLE and PCA

- LLE is motivated by considering local linear embeddings of the data.
- Interestingly, as we increase the neighborhood size to $K = n - 1$ we do not recover PCA.
- But PCA is the "optimal" linear embedding!!
- LLE is optimizing a pseudolikelihood: in contrast the MEU algorithm, which LLE approximates, does recover PCA when $K = n - 1$.

LLE and PCA

- LLE is motivated by considering local linear embeddings of the data.
- Interestingly, as we increase the neighborhood size to $K = n - 1$ we do not recover PCA.
- But PCA is the "optimal" linear embedding!!
- LLE is optimizing a pseudolikelihood: in contrast the MEU algorithm, which LLE approximates, does recover PCA when $K = n - 1$.

Isomap

- Isomap (Tenenbaum et al., 2000) follows the CMDS framework.
- Sparse graph of distances is created.
- Fill in graph for non-neighbors with a shortest path algorithm.
- MVU and MEU can be start with a sparse graph of (squared) distances.
- Fill in other distances by maximizing the total variance/entropy.

Isomap

- Isomap (Tenenbaum et al., 2000) follows the CMDS framework.
- Sparse graph of distances is created.
- Fill in graph for non-neighbors with a shortest path algorithm.
- MVU and MEU can be start with a sparse graph of (squared) distances.
- Fill in other distances by maximizing the total variance/entropy.

Isomap

- Isomap (Tenenbaum et al., 2000) follows the CMDS framework.
- Sparse graph of distances is created.
- Fill in graph for non-neighbors with a shortest path algorithm.
- MVU and MEU can be start with a sparse graph of (squared) distances.
- Fill in other distances by maximizing the total variance/entropy.

Isomap

- Isomap (Tenenbaum et al., 2000) follows the CMDS framework.
- Sparse graph of distances is created.
- Fill in graph for non-neighbors with a shortest path algorithm.
- MVU and MEU can be start with a sparse graph of (squared) distances.
- Fill in other distances by maximizing the total variance/entropy.

Isomap

- Isomap (Tenenbaum et al., 2000) follows the CMDS framework.
- Sparse graph of distances is created.
- Fill in graph for non-neighbors with a shortest path algorithm.
- MVU and MEU can be start with a sparse graph of (squared) distances.
- Fill in other distances by maximizing the total variance/entropy.

Relation to GP-LVM

- Both MEU and GP-LVM (Lawrence, 2004, 2005) specify a similar Gaussian density over the training data.
- Gauss Markov random field can easily be specified by a Gaussian process through an appropriate covariance.
- e.g. the O-U covariance in a 1-D latent space $k(x, x') = \exp(-\|x - x\prime\|_1)$ gives a sparse inverse with only neighbors connected.
- In GP-LVM neigborhood is optimized as part of the training procedure.

Relation to GP-LVM

- Both MEU and GP-LVM (Lawrence, 2004, 2005) specify a similar Gaussian density over the training data.
- Gauss Markov random field can easily be specified by a Gaussian process through an appropriate covariance.
- e.g. the O-U covariance in a 1-D latent space $k(x, x') = \exp(-\|x - x\prime\|_1)$ gives a sparse inverse with only neighbors connected.
- In GP-LVM neigborhood is optimized as part of the training procedure.

Relation to GP-LVM

- Both MEU and GP-LVM (Lawrence, 2004, 2005) specify a similar Gaussian density over the training data.
- Gauss Markov random field can easily be specified by a Gaussian process through an appropriate covariance.
- e.g. the O-U covariance in a 1-D latent space $k(x, x') = \exp(-\|x - x\prime\|_1)$ gives a sparse inverse with only neighbors connected.
- In GP-LVM neigborhood is optimized as part of the training procedure.

Relation to GP-LVM

- Both MEU and GP-LVM (Lawrence, 2004, 2005) specify a similar Gaussian density over the training data.
- Gauss Markov random field can easily be specified by a Gaussian process through an appropriate covariance.
- e.g. the O-U covariance in a 1-D latent space $k(x, x') = \exp(-\|x - x\prime\|_1)$ gives a sparse inverse with only neighbors connected.
- In GP-LVM neigborhood is optimized as part of the training procedure.

Outline

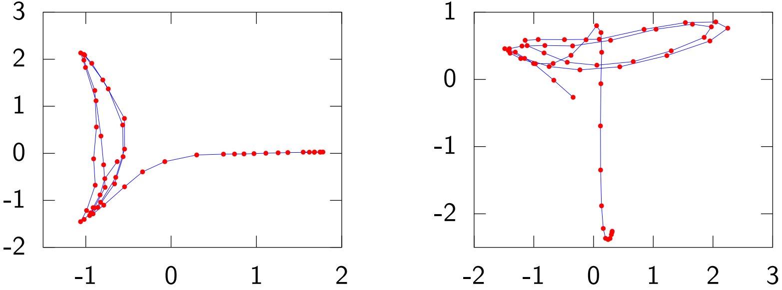
Simple Experiments

- Consider two real data sets.
- We apply each of the spectral methods we have reviewed.
- Apply the MEU framework.
- Follow the suggestion of Harmeling (Harmeling, 2007) and use the GPLVM likelihood (Lawrence, 2005) for embedding quality.
- The higher the likelihood the better the embedding.

Motion Capture Data

- Data consists of a 3-dimensional point cloud of the location of 34 points from a subject performing a run.
- 102 dimensional data set containing 55 frames of motion capture.
- Subject begins the motion from stationary and takes approximately three strides of run.
- Should see this structure in the visualization: a starting position followed by a series of loops.
- Data was made available by Ohio State University.
- The two dominant eigenvectors are visualized in following figures.

Laplacian Eigenmaps and LLE



(a) Laplacian Eigenmaps (b) Locally Linear Embedding

Figure: Models capture either the cyclic structure or the structure associated with the start of the run or both parts.

Isomap and MVU

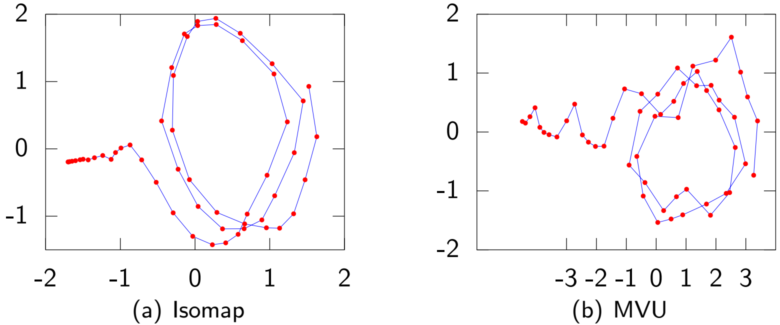
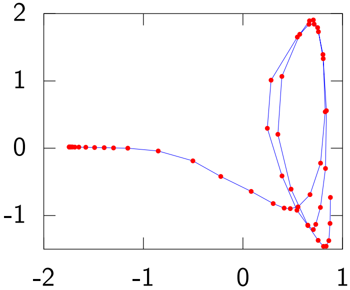


Figure: Models capture either the cyclic structure or the structure associated with the start of the run or both parts.

MEU



(a) MEU

Figure: Models capture either the cyclic structure or the structure associated with the start of the run or both parts.

Motion Capture: Model Scores

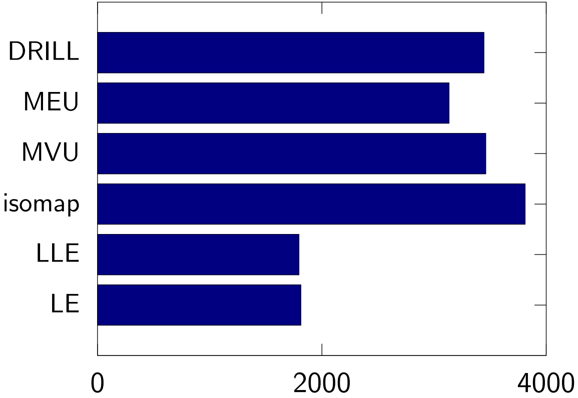
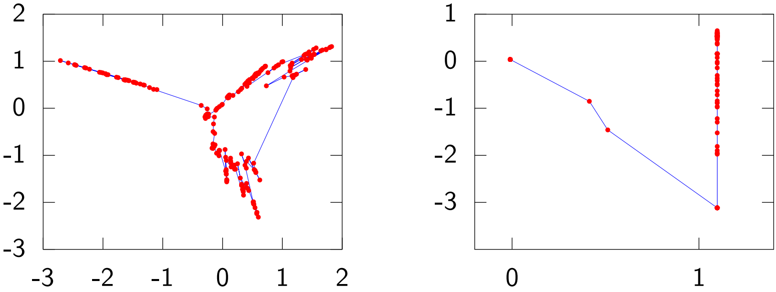


Figure: Model score for the different spectral approaches.

Robot Navigation Example

- Second data set: series of recordings from a robot as it traces a square path in a building.
- It records the strength of WiFi signals (see Ferris et al., 2007, for an application).
- Robot only in two dimensions, the inherent dimensionality of the data should be two.
- Robot completes a single circuit after entry: it is expected to exhibit "loop closure".
- Data consists of 215 frames of measurement of WiFi signal strength of 30 access points.

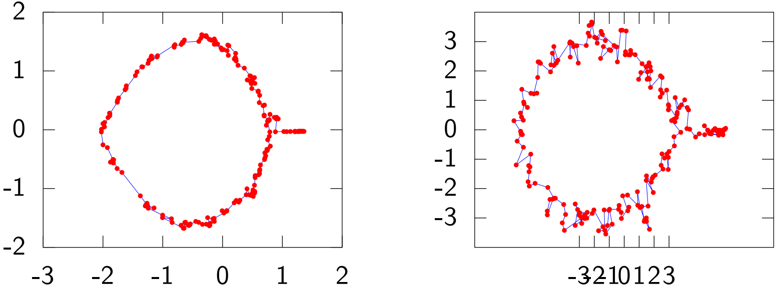
Laplacian Eigenmaps and LLE



(a) Laplacian Eigenmaps (b) Locally Linear Embedding

Figure: Models show loop closure but smooth the trace to different degrees.

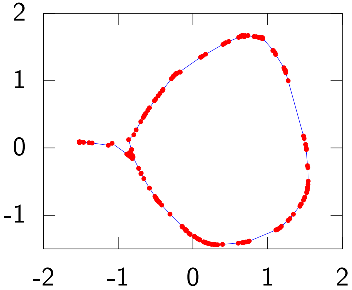
Isomap and MVU



(a) Isomap (b) MVU

Figure: Models show loop closure but smooth the trace to different degrees.

MEU



(a) MEU

Figure: Models show loop closure but smooth the trace to different degrees.

Robot Navigation: Model Scores

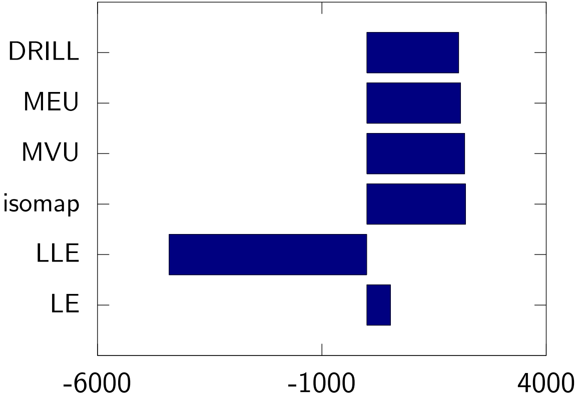


Figure: Model score for the different spectral approaches.

Outline

Discussion

- New perspective on dimensionality reduction algorithms based around maximum entropy.
- Start with MVU and end with GRFs.
- Hope that this perspective on dimensionality reduction will encourage new strands of research at the interface of these areas.

Discussion

- New perspective on dimensionality reduction algorithms based around maximum entropy.
- Start with MVU and end with GRFs.
- Hope that this perspective on dimensionality reduction will encourage new strands of research at the interface of these areas.

Discussion

- New perspective on dimensionality reduction algorithms based around maximum entropy.
- Start with MVU and end with GRFs.
- Hope that this perspective on dimensionality reduction will encourage new strands of research at the interface of these areas.

Stages of Spectral Dimensionality Reduction

- Our perspective shows there are three separate stages used in existing spectral dimensionality algorithms.
  1. A neighborhood between data points is selected. Normally k-nearest neighbors or similar algorithms are used.
  2. Interpoint distances between neighbors are fed to the algorithms which provide a similarity matrix. The way the entries in the similarity matrix are computed is the main difference between the different algorithms.
  3. The relationship between points in the similarity matrix is visualized using the eigenvectors of the similarity matrix.

Stages of Spectral Dimensionality Reduction

- Our perspective shows there are three separate stages used in existing spectral dimensionality algorithms.
  1. A neighborhood between data points is selected. Normally k-nearest neighbors or similar algorithms are used.
  2. Interpoint distances between neighbors are fed to the algorithms which provide a similarity matrix. The way the entries in the similarity matrix are computed is the main difference between the different algorithms.
  3. The relationship between points in the similarity matrix is visualized using the eigenvectors of the similarity matrix.

Stages of Spectral Dimensionality Reduction

- Our perspective shows there are three separate stages used in existing spectral dimensionality algorithms.
  1. A neighborhood between data points is selected. Normally k-nearest neighbors or similar algorithms are used.
  2. Interpoint distances between neighbors are fed to the algorithms which provide a similarity matrix. The way the entries in the similarity matrix are computed is the main difference between the different algorithms.
  3. The relationship between points in the similarity matrix is visualized using the eigenvectors of the similarity matrix.

Stages of Spectral Dimensionality Reduction

- Our perspective shows there are three separate stages used in existing spectral dimensionality algorithms.
  1. A neighborhood between data points is selected. Normally k-nearest neighbors or similar algorithms are used.
  2. Interpoint distances between neighbors are fed to the algorithms which provide a similarity matrix. The way the entries in the similarity matrix are computed is the main difference between the different algorithms.
  3. The relationship between points in the similarity matrix is visualized using the eigenvectors of the similarity matrix.

Stages of Spectral Dimensionality Reduction

- Our perspective shows there are three separate stages used in existing spectral dimensionality algorithms.
  1. A neighborhood between data points is selected. Normally k-nearest neighbors or similar algorithms are used.
  2. Interpoint distances between neighbors are fed to the algorithms which provide a similarity matrix. The way the entries in the similarity matrix are computed is the main difference between the different algorithms.
  3. The relationship between points in the similarity matrix is visualized using the eigenvectors of the similarity matrix.

Our Perspective

- Each step is somewhat orthogonal.
- Neighborhood relations need not come from nearest neighbors: can use structure learning.
- Main difference between approaches is how similarity matrix entries are determined.
- Final step attempts to visualize the similarity using eigenvectors. This is just one possible approach.
- There is an entire field of graph visualization proposing different approaches to visualizing such graphs.

Our Perspective

- Each step is somewhat orthogonal.
- Neighborhood relations need not come from nearest neighbors: can use structure learning.
- Main difference between approaches is how similarity matrix entries are determined.
- Final step attempts to visualize the similarity using eigenvectors. This is just one possible approach.
- There is an entire field of graph visualization proposing different approaches to visualizing such graphs.

Our Perspective

- Each step is somewhat orthogonal.
- Neighborhood relations need not come from nearest neighbors: can use structure learning.
- Main difference between approaches is how similarity matrix entries are determined.
- Final step attempts to visualize the similarity using eigenvectors. This is just one possible approach.
- There is an entire field of graph visualization proposing different approaches to visualizing such graphs.

Our Perspective

- Each step is somewhat orthogonal.
- Neighborhood relations need not come from nearest neighbors: can use structure learning.
- Main difference between approaches is how similarity matrix entries are determined.
- Final step attempts to visualize the similarity using eigenvectors. This is just one possible approach.
- There is an entire field of graph visualization proposing different approaches to visualizing such graphs.

Our Perspective

- Each step is somewhat orthogonal.
- Neighborhood relations need not come from nearest neighbors: can use structure learning.
- Main difference between approaches is how similarity matrix entries are determined.
- Final step attempts to visualize the similarity using eigenvectors. This is just one possible approach.
- There is an entire field of graph visualization proposing different approaches to visualizing such graphs.

Advantages of Existing Approaches

- Conflating the three steps allows faster complete algorithms.
- E.g. mixing 2nd & 3rd allows speed ups by never computing the similarity matrix.
- We still can understand the algorithm from the unifying perspective while exploiting the computational advantages offered by this neat shortcut.

Advantages of Existing Approaches

- Conflating the three steps allows faster complete algorithms.
- E.g. mixing 2nd & 3rd allows speed ups by never computing the similarity matrix.
- We still can understand the algorithm from the unifying perspective while exploiting the computational advantages offered by this neat shortcut.

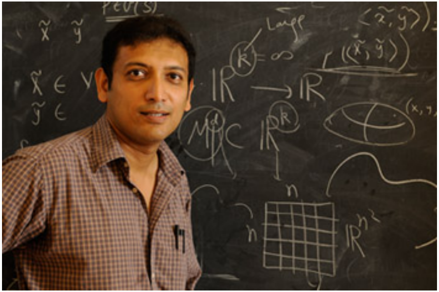
Advantages of Existing Approaches

- Conflating the three steps allows faster complete algorithms.
- E.g. mixing 2nd & 3rd allows speed ups by never computing the similarity matrix.
- We still can understand the algorithm from the unifying perspective while exploiting the computational advantages offered by this neat shortcut.

Acknowledgements

Conversations with John Kent, Chris Williams, Brenden Lake, Joshua Tenenbaum and John Lafferty have influenced the thinking in this work.

Partha and Sam



References I

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003. [DOI].

B. D. Ferris, D. Fox, and N. D. Lawrence. WiFi-SLAM using Gaussian process latent variable models. In M. M. Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 2480–2485, 2007. [PDF].

S. Harmeling. Exploring model selection techniques for nonlinear dimensionality reduction. Technical Report EDI-INF-RR-0960, University of Edinburgh,

E. T. Jaynes. Bayesian methods: General background. In J. H. Justice, editor, *Maximum Entropy and Bayesian Methods in Applied Statistics*, pages 1–25. Cambridge University Press, 1986.

C. Kemp and J. B. Tenenbaum. The discovery of structural form. *Proc. Natl. Acad. Sci. USA*, 105(31), 2008.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009. [Google Books] .

N. D. Lawrence. Gaussian process models for visualisation of high dimensional data. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, pages 329–336, Cambridge, MA, 2004. MIT Press.

N. D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 11 2005.

K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate analysis*. Academic Press, London, 1979. [Google Books] .

S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500): 2323–2326, 2000. [DOI].

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998. [DOI].

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. [DOI].

L. A. Wasserman. *All of Statistics*. Springer-Verlag, New York, 2003. [Google Books] .

K. Q. Weinberger, F. Sha, and L. K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In R. Greiner and D. Schuurmans, editors, *Proceedings of the International Conference in Machine Learning*, volume 21, pages 839–846. Omnipress, 2004.

X. Zhu, J. Lafferty, and Z. Ghahramani. Semi-supervised learning: From Gaussian fields to Gaussian processes. Technical Report CMU-CS-03-175, Carnegie Mellon University, [PDF].

LLE: Point One

- For unit diagonals we have $\mathbf{M} = \mathbf{I} - \mathbf{W}$.
- Here the off diagonal sparsity pattern of $\mathbf{W}$ matches $\mathbf{M}$.
- Thus
$$(\mathbf{I} - \mathbf{W})^\top \mathbf{1} = \mathbf{0}.$$
- LLE proscribes that the smallest eigenvectors of
$$(\mathbf{I} - \mathbf{W})(\mathbf{I} - \mathbf{W})^\top = \mathbf{M}\mathbf{M}^\top = \mathbf{L}$$
(like Laplacian Eigenmaps).
- Equivalent to CMDS on the GRF described by $\mathbf{L}$.

LLE: Point One

- For unit diagonals we have $\mathbf{M} = \mathbf{I} - \mathbf{W}$.
- Here the off diagonal sparsity pattern of $\mathbf{W}$ matches $\mathbf{M}$.
- Thus
$$(\mathbf{I} - \mathbf{W})^\top \mathbf{1} = \mathbf{0}.$$
- LLE proscribes that the smallest eigenvectors of
$$(\mathbf{I} - \mathbf{W})(\mathbf{I} - \mathbf{W})^\top = \mathbf{M}\mathbf{M}^\top = \mathbf{L}$$
(like Laplacian Eigenmaps).
- Equivalent to CMDS on the GRF described by $\mathbf{L}$.

LLE: Point One

- For unit diagonals we have $\mathbf{M} = \mathbf{I} - \mathbf{W}$.
- Here the off diagonal sparsity pattern of $\mathbf{W}$ matches $\mathbf{M}$.
- Thus

$$(\mathbf{I} - \mathbf{W})^\top \mathbf{1} = \mathbf{0}.$$

- LLE proscribes that the smallest eigenvectors of

$$(\mathbf{I} - \mathbf{W})(\mathbf{I} - \mathbf{W})^\top = \mathbf{M}\mathbf{M}^\top = \mathbf{L}$$

(like Laplacian Eigenmaps).
- Equivalent to CMDS on the GRF described by $\mathbf{L}$.

LLE: Point One

- For unit diagonals we have $\mathbf{M} = \mathbf{I} - \mathbf{W}$.
- Here the off diagonal sparsity pattern of $\mathbf{W}$ matches $\mathbf{M}$.
- Thus
$$(\mathbf{I} - \mathbf{W})^\top \mathbf{1} = \mathbf{0}.$$
- LLE proscribes that the smallest eigenvectors of
$$(\mathbf{I} - \mathbf{W})(\mathbf{I} - \mathbf{W})^\top = \mathbf{M}\mathbf{M}^\top = \mathbf{L}$$
(like Laplacian Eigenmaps).
- Equivalent to CMDS on the GRF described by $\mathbf{L}$.

LLE: Point One

- For unit diagonals we have $\mathbf{M} = \mathbf{I} - \mathbf{W}$.
- Here the off diagonal sparsity pattern of $\mathbf{W}$ matches $\mathbf{M}$.
- Thus
$$(\mathbf{I} - \mathbf{W})^\top \mathbf{1} = \mathbf{0}.$$

- LLE proscribes that the smallest eigenvectors of
$$(\mathbf{I} - \mathbf{W})(\mathbf{I} - \mathbf{W})^\top = \mathbf{M}\mathbf{M}^\top = \mathbf{L}$$
(like Laplacian Eigenmaps).
- Equivalent to CMDS on the GRF described by $\mathbf{L}$.

LLE: Second Point

- Pseudolikelihood approximation (see e.g. Koller and Friedman, 2009, pg 970): product of the conditional densities:

$$p(\mathbf{Y}) \approx \prod_{i=1}^{n} p(\mathbf{y}_{i,:}|\mathbf{Y}_{\backslash i}),$$

  $\mathbf{Y}_{\backslash i}$ represents data other than the ith point.
- True likelihood is proportional to this but requires renormalization.
- In pseudolikelihood normalization is ignored.

LLE: Second Point

- Pseudolikelihood approximation (see e.g. Koller and Friedman, 2009, pg 970): product of the conditional densities:

$$p(\mathbf{Y}) \approx \prod_{i=1}^{n} p(\mathbf{y}_{i,:}|\mathbf{Y}_{\backslash i}),$$

  $\mathbf{Y}_{\backslash i}$ represents data other than the ith point.
- True likelihood is proportional to this but requires renormalization.
- In pseudolikelihood normalization is ignored.

LLE: Second Point

- Pseudolikelihood approximation (see e.g. Koller and Friedman, 2009, pg 970): product of the conditional densities:

$$p(\mathbf{Y}) \approx \prod_{i=1}^{n} p(\mathbf{y}_{i,:}|\mathbf{Y}_{\backslash i}),$$

  $\mathbf{Y}_{\backslash i}$ represents data other than the ith point.
- True likelihood is proportional to this but requires renormalization.
- In pseudolikelihood normalization is ignored.

Consistency of Model

- Deeper lesson on interpretation of consistency:
  - for “sampled points” parameters better determined with increasing n
  - for “sampled features” parameters better determined with increasing p.
- In the large p small n domain, the “sampled features” formalism is attractive.
- For computing the likelihood of an out we need to estimate parameters associated with that point.

Outline

Final Experiment: Structure Learning

- Test the ability of L1 regularization of the random field to learn the neighborhood.
- Considered the motion capture data and used the DRILL with a neighborhood size of 20 and full connectivity.
- L1 regularization on the parameters: vary regularization size and seek a maximum under the GPLVM.

Structure Learning from Neighborhood of 20

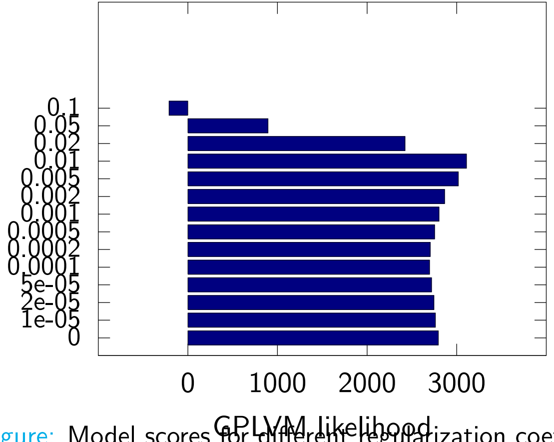


Figure: Model scores for different regularization coefficients.

Structure Learning from Neighborhood of 20

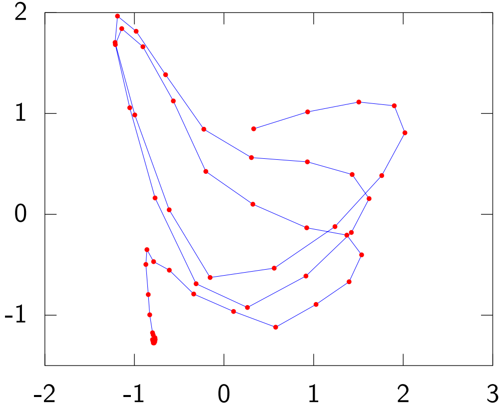


Figure: Visualization associated with highest model score.

Full Structure Learning

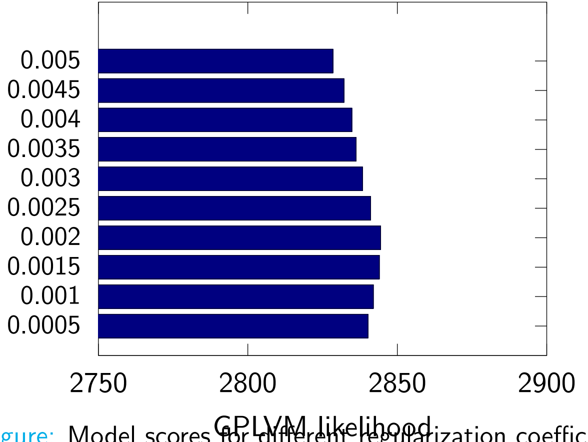


Figure: Model scores for different regularization coefficients.

Full Structure Learning

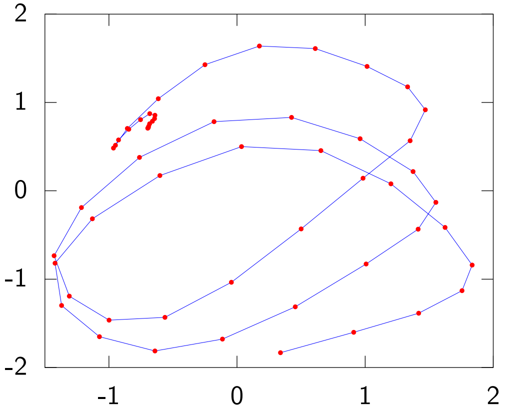


Figure: Visualization associated with highest model score.

Different Neighborhood Scores

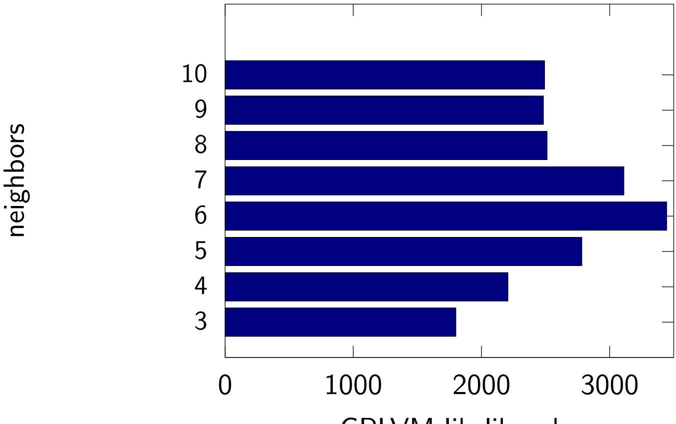


Figure: Model scores for different neighborhood sizes.

Different Neighborhood Scores

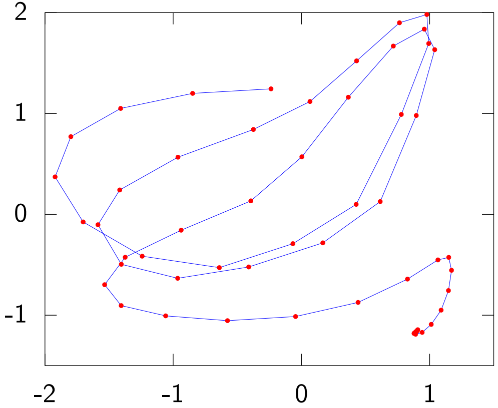


Figure: Visualization associated with highest model score.

Structure Learning from Neighborhood of 6

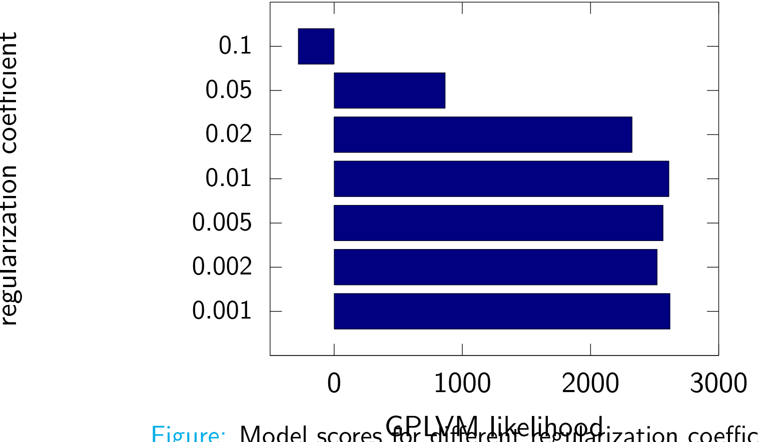


Figure: Model scores for different regularization coefficients.

Structure Learning from Neighborhood of 6

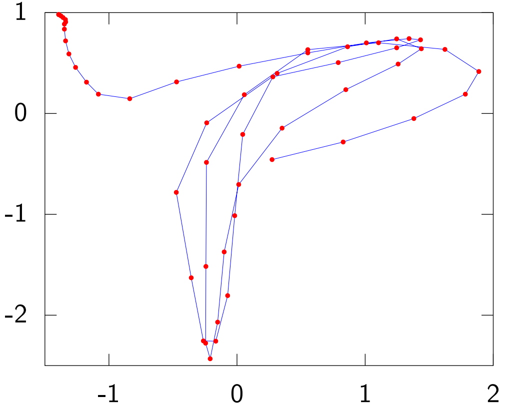


Figure: Visualization associated with highest model score.

Discussion of "Spectral Dimensionality Reduction via Maximum Entropy"

Laurens van der Maaten

TU Delft

Timeline

* First manifold learners were instantiations of Kernel PCA that use hand-crafted "kernels":
  * Examples: Isomap, Laplacian Eigenmaps, LLE, etc.
* Recently, interest shifted to learning a "good" kernel:
  * Maximize some rank-minimizing objective subject to linear constraints that preserve local structure
  * Examples: Maximum Variance Unfolding, Structure Preserving Embedding, Maximum Entropy Unfolding

(Tenenbaum et al., 2000; Roweis & Saul, 2000; Belkin & Niyogi, 2003; Weinberger & Saul, 2005; Shaw & Jebara, 2010)

Manifold learning vs. generative modeling

* Interesting connection between MEU and GPLVM:
  * Both model P(Y) as a GRF in which a data point is a node
  * Key difference is in how the GRF covariance is obtained
  * GPLVM: $\mathbf{K} = \mathbf{X}^T \mathbf{X}$
  * MEU: $\mathbf{K} = (\mathbf{L} + \gamma \mathbf{I})^{-1}$
* MEU unifies two seemingly very different approaches:
  * Manifold learning
  * Generative modeling

Manifold learning vs. generative modeling

* Manifold learning:
  * Smooth mapping *from data space to latent space*
  * Similar data points should be close together in the embedding: preserving local structure!
* Generative modeling:
  * Smooth mapping *from latent space to data space*
  * Dissimilar points may not be close together in the embedding: preserving global structure!

Manifold learning vs. generative modeling

* MEU is the first to combine the best of both worlds:
  * It preserves local data structure in the embedding
  * Probabilistic framework allows for natural extensions to missing data, hierarchical models, etc.
* Current formulation still has a peculiarity:
  * In MEU, the embedding does not appear as latent variable in the generative model
  * One could use any MDS technique to embed **K**

Rank minimization

* The rank of the kernel matrix controls what we do with dissimilar data points when preserving local data structure:
  * Maximum Variance Unfolding
    * Maximizes the sum of the kernel eigenvalues
  * Maximum Entropy Unfolding
    * Maximizes the sum of the log-eigenvalues

* Which one is better?

Rank minimization

* How you deal with dissimilar data makes a difference:
  * Stochastic Neighbor Embedding is Laplacian Eigenmaps with a different covariance constraint (Carreira-Perpinan, 2010)
* How to deal with dissimilar data may be more important than how to deal with similar data in manifold learning
* Recent successes push away dissimilar data as far as possible (Weinberger & Saul, 2005; van der Maaten & Hinton, 2008)
* Perhaps MEU can lead to new insights here?