Topic Modelling

Topic modelling: given a corpus of documents, find the "topics" they discuss.

Example: consider abstracts of papers PNAS.

> ***Global climate change and mammalian species diversity in U.S. national parks***
>
> *National parks and bioreserves are key conservation tools used to protect species and their habitats within the confines of fixed political boundaries. This inflexibility may be their "Achilles' heel" as conservation tools in the face of emerging global-scale environmental problems such as climate change. Global climate change, brought about by rising levels of greenhouse gases, threatens to alter the geographic distribution of many habitats and their component species....*
>
> ***The influence of large-scale wind power on global climate***
>
> *Large-scale use of wind power can alter local and global climate by extracting kinetic energy and altering turbulent transport in the atmospheric boundary layer. We report climate-model simulations that address the possible climatic impacts of wind power at regional to global scales by using two general circulation models and several parameterizations of the interaction of wind turbines with the boundary layer....*
>
> ***Twentieth century climate change: Evidence from small glaciers***
>
> *The relation between changes in modern glaciers, not including the ice sheets of Greenland and Antarctica, and their climatic environment is investigated to shed light on paleoglacier evidence of past climate change and for projecting the effects of future climate warming on cold regions of the world. Loss of glacier volume has been more or less continuous since the 19th century, but it is not a simple adjustment to the end of an "anomalous" Little Ice Age....*

Topic Modelling

Example topics discovered from PNAS abstracts (each topic represented in terms of the top 5 most common words in that topic).

217	274	126	63	200	209
INSECT	SPECIES	GENE	STRUCTURE	FOLDING	NUCLEAR
MYB	PHYLOGENETIC	VECTOR	ANGSTROM	NATIVE	NUCLEUS
PHEROMONE	EVOLUTION	VECTORS	CRYSTAL	PROTEIN	LOCALIZATION
LENS	EVOLUTIONARY	EXPRESSION	RESIDUES	STATE	CYTOPLASM
LARVAE	SEQUENCES	TRANSFER	STRUCTURES	ENERGY	EXPORT
42	**2**	**280**	**15**	**64**	**102**
NEURAL	SPECIES	SPECIES	CHROMOSOME	CELLS	TUMOR
DEVELOPMENT	GLOBAL	SELECTION	REGION	CELL	CANCER
DORSAL	CLIMATE	EVOLUTION	CHROMOSOMES	ANTIGEN	TUMORS
EMBRYOS	CO2	GENETIC	KB	LYMPHOCYTES	HUMAN
VENTRAL	WATER	POPULATIONS	MAP	CD4	CELLS
112	**210**	**201**	**165**	**142**	**222**
HOST	SYNAPTIC	RESISTANCE	CHANNEL	PLANTS	CORTEX
BACTERIAL	NEURONS	RESISTANT	CHANNELS	PLANT	BRAIN
BACTERIA	POSTSYNAPTIC	DRUG	VOLTAGE	ARABIDOPSIS	SUBJECTS
STRAINS	HIPPOCAMPAL	DRUGS	CURRENT	TOBACCO	TASK
SALMONELLA	SYNAPSES	SENSITIVE	CURRENTS	LEAVES	AREAS
39	**105**	**221**	**270**	**55**	**114**
THEORY	HAIR	LARGE	TIME	FORCE	POPULATION
TIME	MECHANICAL	SCALE	SPECTROSCOPY	SURFACE	POPULATIONS
SPACE	MB	DENSITY	NMR	MOLECULES	GENETIC
GIVEN	SENSORY	OBSERVED	SPECTRA	SOLUTION	DIVERSITY
PROBLEM	EAR	OBSERVATIONS	TRANSFER	SURFACES	ISOLATES
		109	**120**		
		RESEARCH	AGE		
		NEW	OLD		
		INFORMATION	AGING		
		UNDERSTANDING	LIFE		
		PAPER	YOUNG		

Recap: Beta Distributions

Recall the Bayesian coin toss example.

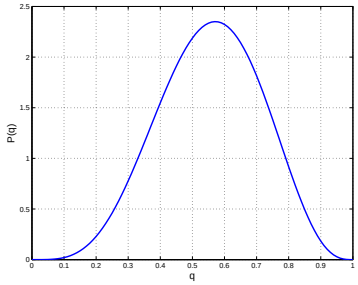
$$P(H|q) = q \qquad\qquad P(T|q) = 1 - q$$

The probability of a sequence of coin tosses is:

$$P(HHTT\cdots HT|q) = q^{\#\text{heads}}(1-q)^{\#\text{tails}}$$

A conjugate prior for q is the Beta distribution:

$$P(q) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} q^{a-1}(1-q)^{b-1} \qquad\qquad a, b \geq 0$$



Dirichlet Distributions

Imagine a Bayesian dice throwing example.

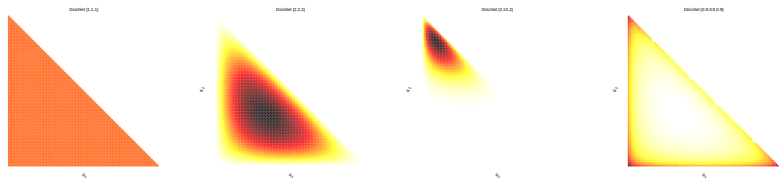
$$P(1|\mathbf{q}) = q_1 \quad P(2|\mathbf{q}) = q_2 \quad P(3|\mathbf{q}) = q_3 \quad P(4|\mathbf{q}) = q_4 \quad P(5|\mathbf{q}) = q_5 \quad P(6|\mathbf{q}) = q_6$$

with $q_i \geq 0, \sum_i q_i = 1$. The probability of a sequence of dice throws is:

$$P(34156\cdots 12|\boldsymbol{q}) = \prod_{i=1}^{6} q_i^{\#\text{ face } i}$$

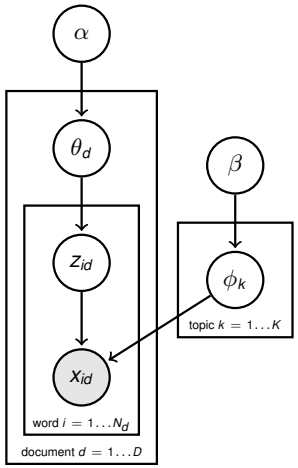
A conjugate prior for $\boldsymbol{q}$ is the Dirichlet distribution:

$$P(\boldsymbol{q}) = \frac{\Gamma(\sum_i a_i)}{\prod_i \Gamma(a_i)} \prod_i q_i^{a_i - 1} \qquad q_i \geq 0, \sum_i q_i = 1 \qquad a_i \geq 0$$



Latent Dirichlet Allocation

Each document is a sequence of words, we model it using a mixture model by ignoring the sequential nature—"bag-of-words" assumption.



- Draw topic distributions from a prior

$$\phi_k \sim \text{Dir}(\beta, \ldots, \beta)$$

- For each document:
  - draw a distribution over topics

$$\boldsymbol{\theta}_d \sim \text{Dir}(\alpha, \ldots, \alpha)$$

  - generate words iid:
    - draw topic from a document-specific dist:

$$z_{id} \sim \text{Discrete}(\boldsymbol{\theta}_d)$$

    - draw word from a topic-specific dist:

$$x_{id} \sim \text{Discrete}(\phi_{z_{id}})$$

Multiple mixtures of discrete distributions, sharing the same set of components (topics).

Latent Dirichlet Allocation as Matrix Decomposition

Let N_{dw} be the number of times word w appears in document d, and P_{dw} is the probability of word w appearing in document d.

$$p(N|P) = \prod_{dw} P_{dw}^{N_{dw}} \quad \text{likelihood term}$$

$$P_{dw} = \sum_k p(\text{pick topic } k) p(\text{pick word } w|k) = \sum_{k=1}^{K} \theta_{dk} \phi_{kw}$$

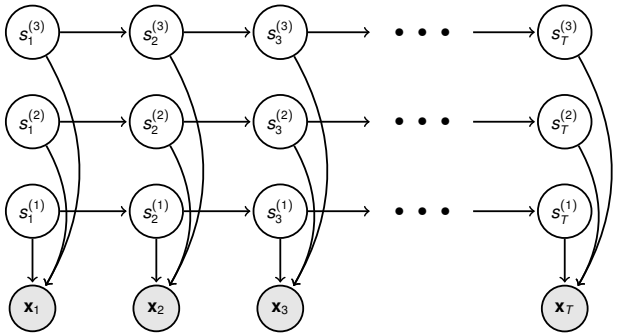
$$P_{dw} = \theta_{dk} \cdot \phi_{kw}$$

This decomposition is similar to PCA and factor analysis, but not Gaussian. Related to non-negative matrix factorisation (NMF).

Latent Dirichlet Allocation

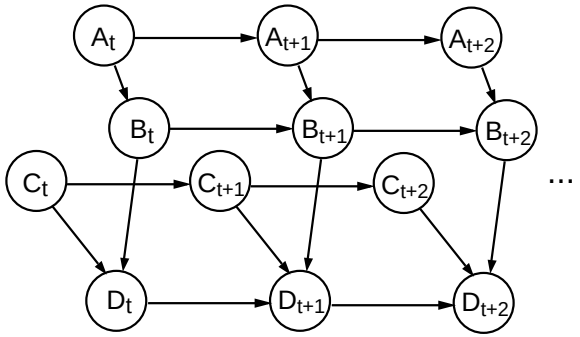
- Exact inference in latent Dirichlet allocation is intractable, and typically either variational or Markov chain Monte Carlo approximations are deployed.
- Latent Dirichlet allocation is an example of a mixed membership model from statistics.
- Latent Dirichlet allocation has also been applied to computer vision, social network modelling, natural language processing...
- Generalizations:
  - Relax the bag-of-words assumption (e.g. a Markov model).
  - Model changes in topics through time.
  - Model correlations among occurrences of topics.
  - Model authors, recipients, multiple corpora.
  - Cross modal interactions (images and tags).
  - Nonparametric generalisations.

Factorial Hidden Markov Models



- These are hidden Markov models with many state variables (i.e. a distributed representation of the state).
- Each state variable evolves independently.
- The state can capture many more bits of information about the sequence (linear in the number of state variables).
- E step is usally intractable (due to explaining away in latent states).

Dynamic Bayesian Networks



- Like factorial HMMs but with structured dependencies among latent states.

Nonlinear Dimensionality Reduction

We can see matrix factorisation methods as performing linear dimensionaliy reduction.

There are many ways to generalise PCA and FA to deal with data which lie on a nonlinear manifold:

- Nonlinear autoencoders
- Generative topographic mappings (GTM) and Kohonen self-organising maps (SOM)
- Multi-dimensional scaling (MDS)
- Kernel PCA (based on MDS representation)
- Isomap
- Locally linear embedding (LLE)
- Stochastic Neighbour Embedding
- Gaussian Process Latent Variable Models (GPLVM)

Another view of PCA: matching inner products

We have viewed PCA as providing a decomposition of the covariance or scatter matrix S. We obtain similar results if we approximate the Gram matrix:

$$\text{minimise} \quad \mathcal{E} = \sum_{ij} (G_{ij} - \mathbf{y}_i \cdot \mathbf{y}_j)^2$$

for $\mathbf{y} \in \mathbb{R}^k$.

That is, look for a k-dimensional embedding in which dot products (which depend on lengths, and angles) are preserved as well as possible.

We will see that this is also equivalent to preserving distances between points.

Another view of PCA: matching inner products

Consider the eigendecomposition of G:

$$G = U \Lambda U^{\mathsf{T}} \quad \text{arranged so} \quad \lambda_1 \geq \cdots \geq \lambda_m \geq 0$$

The best rank-k approximation $G \approx Y^{\mathsf{T}} Y$ is given by:

$$\begin{aligned} Y^{\mathsf{T}} &= [U]_{1:m,1:k} [\Lambda^{1/2}]_{1:k,1:k}; \\ &= [U \Lambda^{1/2}]_{1:m,1:k} \end{aligned}$$

$$Y = [\Lambda^{1/2} U^{\mathsf{T}}]_{1:k,1:m}$$

$$\begin{bmatrix} \sqrt{\lambda_1}\,\mathbf{u}_1^{\mathsf{T}} \\ \sqrt{\lambda_2}\,\mathbf{u}_2^{\mathsf{T}} \\ \mathbf{y}_1 \quad \mathbf{y}_2 \quad \cdots \quad \mathbf{y}_m \\ \sqrt{\lambda_k}\,\mathbf{u}_k^{\mathsf{T}} \\ \vdots \\ \sqrt{\lambda_m}\,\mathbf{u}_m^{\mathsf{T}} \end{bmatrix}$$

The same operations can be performed on the kernel Gram matrix $\Rightarrow$ Kernel PCA.

Multidimensional Scaling

Suppose all we were given were distances or symmetric "dissimilarities" Δ_{ij}.

$$\Delta = \begin{bmatrix} 0 & \Delta_{12} & \Delta_{13} & \Delta_{14} \\ \Delta_{12} & 0 & \Delta_{23} & \Delta_{24} \\ \Delta_{13} & \Delta_{23} & 0 & \Delta_{34} \\ \Delta_{14} & \Delta_{24} & \Delta_{34} & 0 \end{bmatrix}$$

Goal: Find vectors $\mathbf{y}_i$ such that $\|\mathbf{y}_i - \mathbf{y}_j\| \approx \Delta_{ij}$.

This is called **Multidimensional Scaling (MDS)**.

Metric MDS

Assume the dissimilarities represent Euclidean distances between points in some high-D space.

$$\Delta_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\| \text{ with } \sum_i \mathbf{x}_i = \mathbf{0}.$$

We have:

$$\begin{aligned} \Delta_{ij}^2 &= \|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 - 2\mathbf{x}_i \cdot \mathbf{x}_j \\ \sum_k \Delta_{ik}^2 &= m\|\mathbf{x}_i\|^2 + \sum_k \|\mathbf{x}_k\|^2 - \mathbf{0} \\ \sum_k \Delta_{kj}^2 &= \sum_k \|\mathbf{x}_k\|^2 + m\|\mathbf{x}_j\|^2 - \mathbf{0} \\ \sum_{kl} \Delta_{kl}^2 &= 2m \sum_k \|\mathbf{x}_k\|^2 \end{aligned}$$

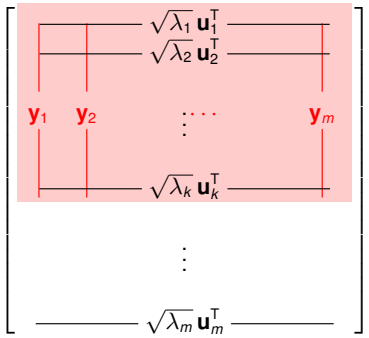
$$\Rightarrow G_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j = \frac{1}{2}\left(\frac{1}{m}\sum_k (\Delta_{ik}^2 + \Delta_{kj}^2) - \frac{1}{m^2}\sum_{kl} \Delta_{kl}^2 - \Delta_{ij}^2\right)$$

Metric MDS and eigenvalues

We will actually minimize the error in the dot products:

$$\mathcal{E} = \sum_{ij} (G_{ij} - \mathbf{y}_i \cdot \mathbf{y}_j)^2$$

As in PCA, this is given by the top slice of the eigenvector matrix.



Interpreting MDS

$$G = \frac{1}{2}\left(\frac{1}{m}(\Delta^2 1 + 1\Delta^2) - \Delta^2 - \frac{1}{m^2} 1^T \Delta^2 1\right)$$

$$G = U\Lambda U^T; \qquad Y = [\Lambda^{1/2} U^T]_{1:k,1:m}$$

(1 is a matrix of ones.)

- **Eigenvectors.** Ordered, scaled and truncated to yield low-dimensional embedded points $\mathbf{y}_i$.
- **Eigenvalues.** Measure how much each dimension contributes to dot products.
- **Estimated dimensionality.** Number of significant (nonnegative – negative possible if Δ_{ij} are not metric) eigenvalues.

MDS and PCA

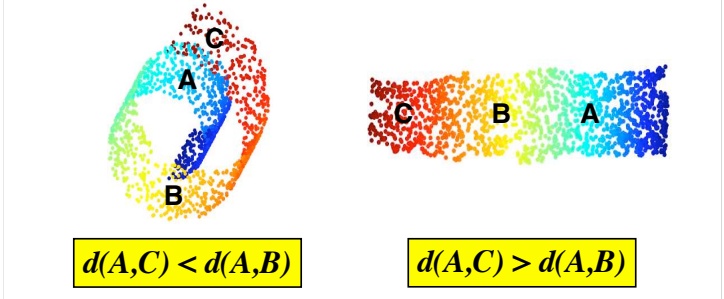
Dual matrices:

$S = \frac{1}{m} XX^\top$	scatter matrix	$(n \times n)$
$G = X^\top X$	Gram matrix	$(m \times m)$

- **Same eigenvalues** up to a constant factor.
- **Equivalent on metric data**, but MDS can run on non-metric dissimilarities.
- **Computational cost** is different.
  - PCA: $O((m+k)n^2)$
  - MDS: $O((n+k)m^2)$

But

Rank ordering of Euclidean distances is NOT preserved in "manifold learning".



Non-metric MDS

MDS can be generalised to permit a monotonic mapping:

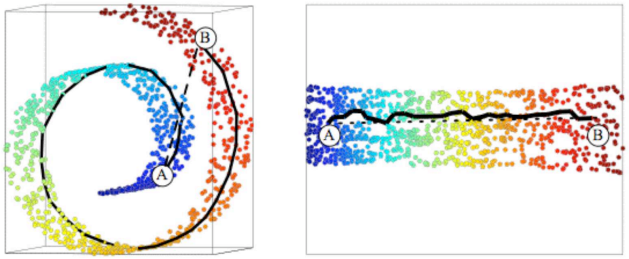
$$\Delta_{ij} \to g(\Delta_{ij}),$$

even if this violates metric rules (like the triangle inequality).

This can introduce a non-linear warping of the manifold.

Isomap

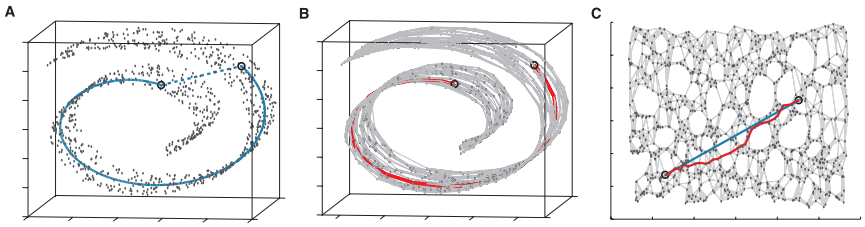
Idea: try to trace distance along the manifold. Use geodesic instead of (transformed) Euclidean distances in MDS.



- preserves local structure
- estimates "global" structure
- preserves information (MDS)

Stages of Isomap

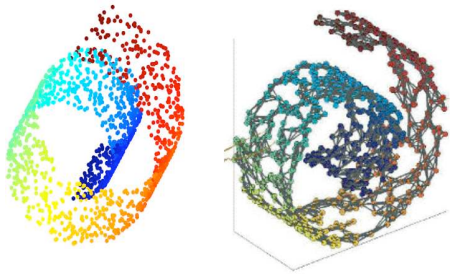
1. Identify neighbourhoods around each point (local points, assumed to be local on the manifold). Euclidean distances are preserved within a neighbourhood.
2. For points outside the neighbourhood, estimate distances by hopping between points within neighbourhoods.
3. Embed using MDS.



Step 1: Adjacency graph

First we construct a graph linking each point to its neighbours.

- vertices represent input points
- undirected edges connect neighbours (weight = Euclidean distance)



Forms a discretised approximation to the submanifold, assuming:

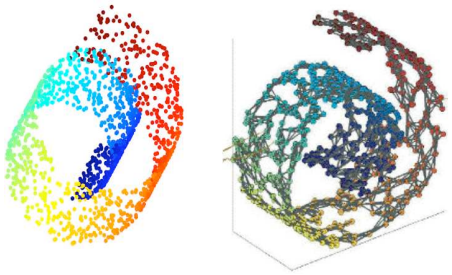
- Graph is singly-connected.
- Graph neighborhoods reflect manifold neighborhoods. No “short cuts”.

Defining the neighbourhood is critical: k-nearest neighbours, inputs within a ball of radius r, prior knowledge.

Step 2: Geodesics

Estimate distances by shortest path in graph.

$$\Delta_{ij} = \min_{\text{path}(\mathbf{x}_i,\mathbf{x}_j)} \left\{ \sum \delta_i \right\}$$

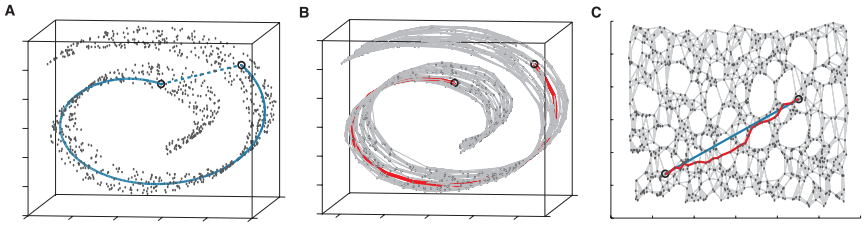


- Standard graph problem. Solved by Dijkstra’s algorithm (and others).
- Better estimates for denser sampling.
- Short cuts very dangerous (“average” path distance?) .

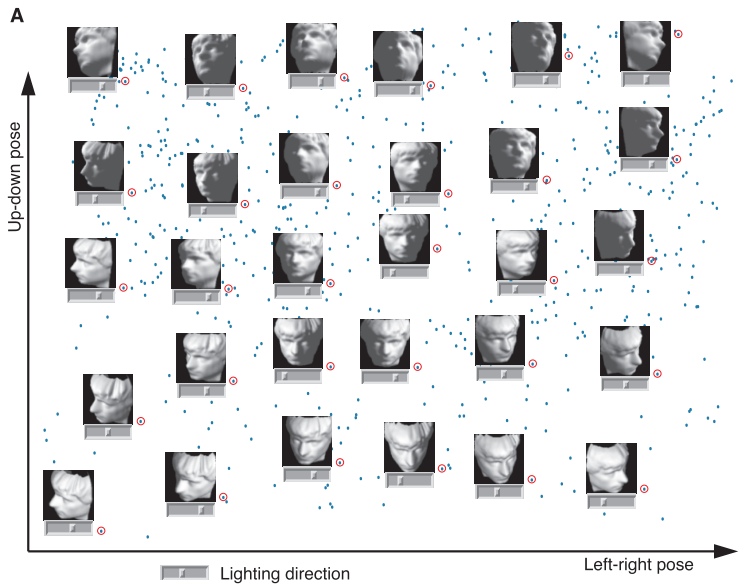
Step 3: Embed

Embed using metric MDS (path distances obey the triangle inequality)

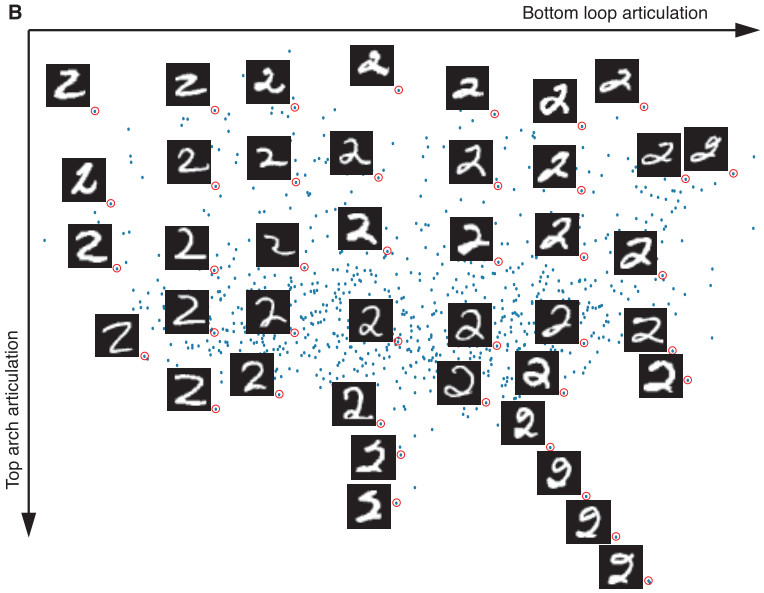
- Eigenvectors of Gram matrix yield low-dimensional embedding.
- Number of significant eigenvalues estimates dimensionality.



Isomap example 1

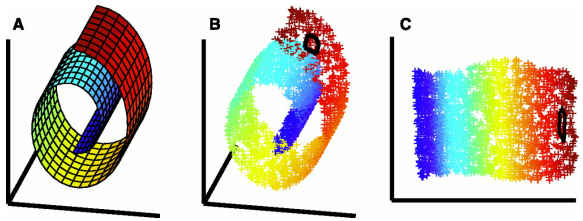


Isomap example 2



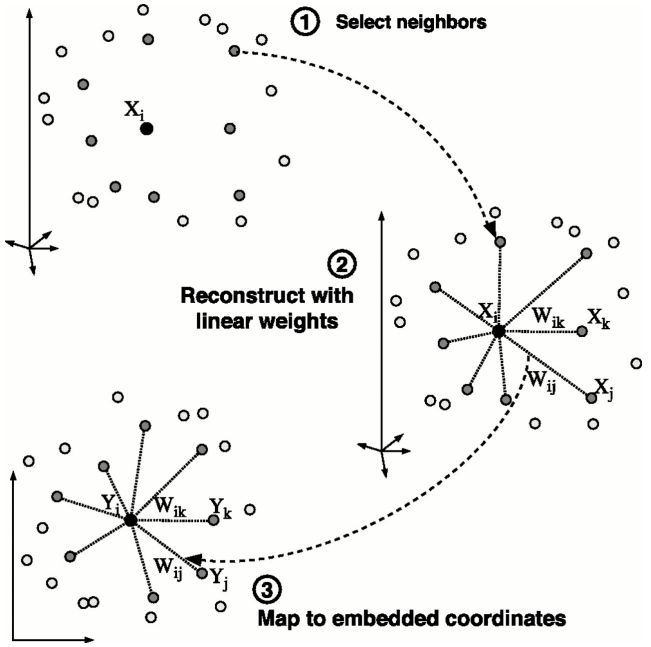
Locally Linear Embedding (LLE)

MDS and isomap preserve local and global (estimated, for isomap) distances. PCA preserves local and global structure.
Idea: estimate local (linear) structure of manifold. Preserve this as well as possible.



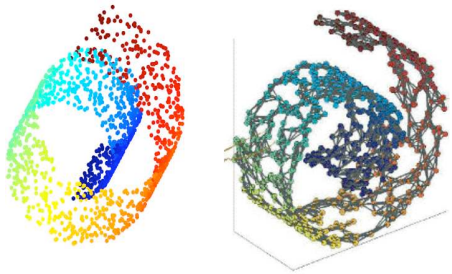
- preserves local structure (not just distance)
- not explicitly global
- preserves only local information

Stages of LLE



Step 1: Neighbourhoods

Just as in isomap, we first define neighbouring points for each input. Equivalent to the isomap graph, but we won't need the graph structure.



Forms a discretised approximation to the submanifold, assuming:

- Graph is singly-connected — although will "work" if not.
- Neighborhoods reflect manifold neighborhoods. No "short cuts".

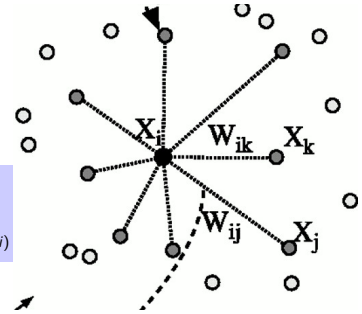
Defining the neighbourhood is critical: k-nearest neighbours, inputs within a ball of radius r, prior knowledge.

Step 2: Local weights

Estimate local weights to minimize error

$$\Phi(W) = \sum_i \left\| \mathbf{x}_i - \sum_{j \in \mathrm{Ne}(i)} \right\|$$

$$\sum_{j \in \mathrm{Ne}(i)} W_{ij} = 1$$



- Linear regression – under- or over-constrained depending on $|\mathrm{Ne}(i)|$.
- Local structure – optimal weights are invariant to rotation, translation and scaling.
- Short cuts less dangerous (one in many).

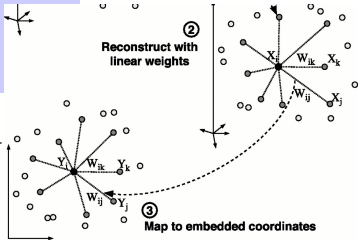
Step 3: Embed

Minimise reconstruction errors in **y**-space under the same weights:

$$\psi(Y) = \sum_i \left\| \mathbf{y}_i - \sum_{j \in \mathrm{Ne}(i)} \right\|$$

subject to:

$$\sum_i \mathbf{y}_i = \mathbf{0}; \qquad \sum_i \mathbf{y}_i \mathbf{y}_i^\mathsf{T}$$



We can re-write the cost function in quadratic form:

$$\psi(Y) = \sum_{ij} \Psi_{ij} [Y^\mathsf{T} Y]_{ij} \text{ with } \Psi = (I - W)^\mathsf{T} (I - W)$$

Minimise by setting Y to equal the bottom $2 \ldots k+1$ eigenvectors of Ψ. (Bottom eigenvector always **1** – discard due to centering constraint)

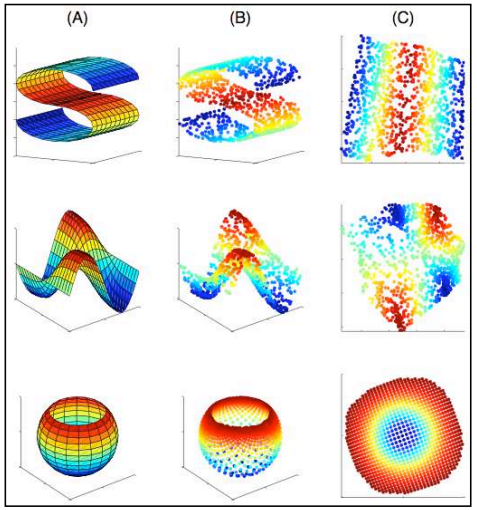
LLE example 1

Surfaces

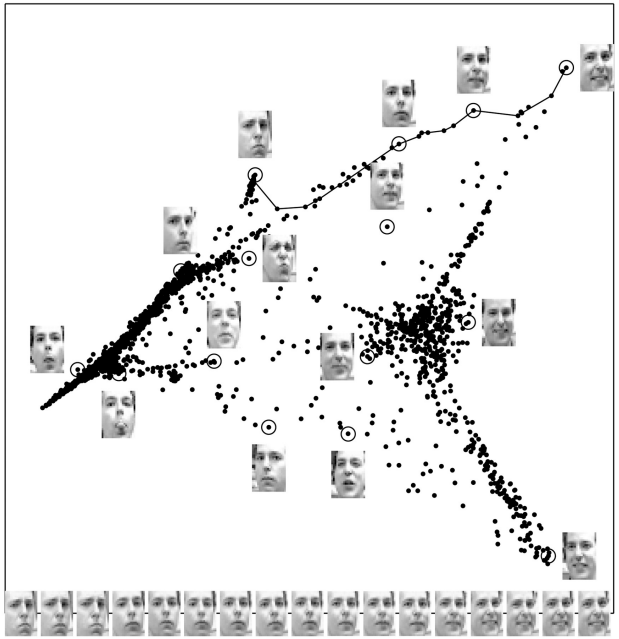
N=1000 inputs

k=8 nearest neighbors

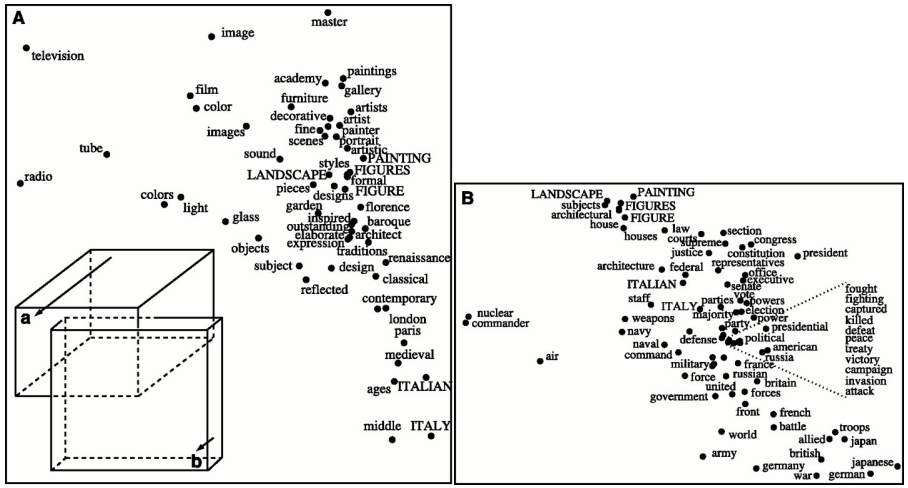
D=3 d=2 dimensions



LLE example 2



LLE example 3



LLE and Isomap

Many similarities

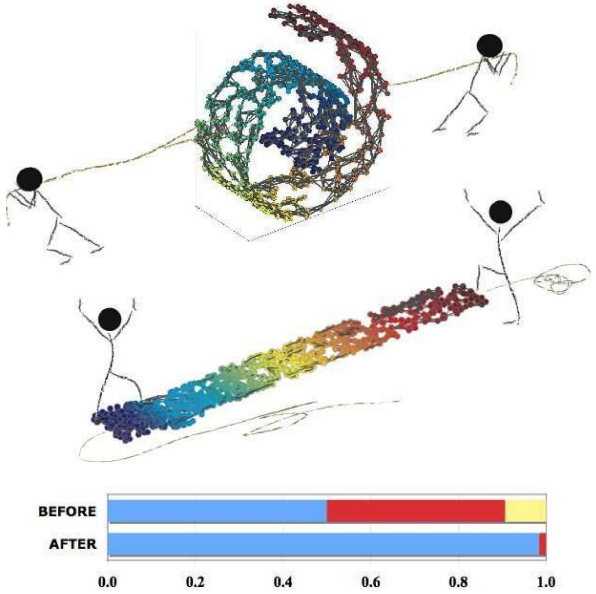
- Graph-based, spectral methods
- No local optima

Essential differences

- LLE does not estimate dimensionality
- Isomap can be shown to be consistent; no theoretical guarantees for LLE.
- LLE diagonalises a sparse matrix – more efficient than isomap.
- Local weights vs. local & global distances.

Maximum Variance Unfolding

Unfold neighbourhood graph preserving local structure.



Maximum Variance Unfolding

Unfold neighbourhood graph preserving local structure.

1. Build the neighbourhood graph.
2. Find $\{\mathbf{y}_i\} \subset \mathbb{R}^n$ (points in **high-D** space) with maximum variance, preserving local distances. Let $K_{ij} = \mathbf{y}_i^\top \mathbf{y}_j$. Then:

   Maximise $\mathrm{Tr}\,[K]$ subject to:

   $$\sum_{ij} K_{ij} = 0 \qquad \text{(centered)}$$
   $$K \succeq 0 \qquad \text{(positive definite)}$$
   $$\underbrace{K_{ii} - 2K_{ij} + K_{jj}}_{\|\mathbf{y}_i - \mathbf{y}_j\|^2} = \|\mathbf{x}_i - \mathbf{x}_j\|^2 \text{ for } j \in \mathrm{Ne}(i) \qquad \text{(locally metric)}$$

   This is a **semi-definite program**: convex optimisation with unique solution.
3. Embed $\mathbf{y}_i$ in $\mathbb{R}^k$ using linear methods (PCA/MDS).

Stochastic Neighbour Embedding

Softer "probabilistic" notions of neighbourhood and consistency.

High-D "transition" probabilities:

$$p_{j|i} = \frac{e^{-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2}}{\sum_{k \neq i} e^{-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_k\|^2/\sigma^2}} \quad \text{for } j \neq i, \qquad p_{i|i} = 0$$

Find $\{\mathbf{y}_i\} \subset \mathbb{R}^k$ to:

$$\text{minimise} \sum_{ij} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} \qquad \text{with } q_{j|i} = \frac{e^{-\frac{1}{2}\|\mathbf{y}_i - \mathbf{y}_j\|^2}}{\sum_{k \neq i} e^{-\frac{1}{2}\|\mathbf{y}_i - \mathbf{y}_k\|^2}}.$$

Nonconvex optimisation is initialisation dependent.

Scale σ plays a similar role to neighbourhood definition:

- Fixed σ: resembles a fixed-radius ball.
- Choose σ_i to maintain consistent entropy in $p_{j|i}$ of $\log_2 k$: similar to k-nearest neighbours.

SNE variants

- Symmetrise probabilities ($p_{ij} = p_{ji}$)

  $$p_{ij} = \frac{e^{-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2}}{\sum_{k \neq l} e^{-\frac{1}{2}\|\mathbf{x}_l - \mathbf{x}_k\|^2/\sigma^2}} \quad \text{for } j \neq i$$

- Gaussian Process Latent Variable Models. Lawrence. Advances in Neural Information Processing Systems, 2004.
  Define q_{ij} analagously, optimise joint KL.

- Heavy-tailed embedding distributions allow embedding to lower dimensions than true manifold:

  $$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}}$$

  Student-t distribution defines "t-SNE".

Focus is on visualisation, rather than manifold discovery.

Gaussian Process Latent Variable Models

Recap: probabilistic PCA

$$\mathbf{y}_i | \mathbf{x}_i, \Lambda \sim \mathcal{N}(\Lambda \mathbf{x}_i, \beta^{-1} I)$$
$$\mathbf{x}_i \sim \mathcal{N}(0, I)$$

Usually: compute posterior over $X = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top$, maximizing likelihood over Λ.

Suppose we know the values of the latent X, then we can integrate out Λ (c.f. linear regression), giving a conditional probability of $Y = [\mathbf{y}_1 \ldots \mathbf{y}_N]^\top$:

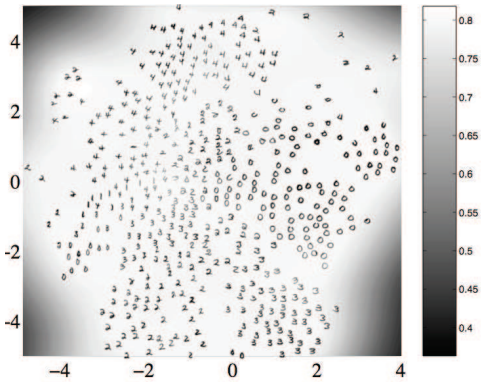
$$\Lambda \sim \mathcal{N}(0, \alpha^{-1} I)$$
$$p(Y|X) \sim |2\pi K|^{-\frac{D}{2}} \exp\left(-\frac{1}{2}\mathrm{Tr}[K^{-1} Y Y^\top]\right) \qquad K = \alpha X X^\top + \beta I$$

This is just D independent Gaussian processes, one for each dimension of Y! Each Gaussian process describes a mapping from latent space $\mathbf{x}$ to one dimension of $\mathbf{y}$.

Replacing the linear kernel with nonlinear kernels gives nonlinear mappings—nonlinear dimensionality reduction.

But now dependence on X is complicated—instead of computing a posterior over X we can only find point values that maximise the likelihood (jointly with the hyperparameters).

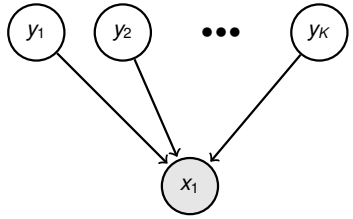
Gaussian Process Latent Variable Models



Intractability

For many probabilistic models of interest, exact inference is not computationally feasible. This occurs for three (main) reasons:

- Distributions may have complicated forms (e.g. non-linearities in generative model).
- "Explaining away" causes coupling from observations
  Observing the value of a child induces dependencies amongst its parents.



- Even with simple models, being Bayesian and computing the full posterior over both latent variables and parameters
  There is often strong coupling between latent variables and parameters.

We can still work with such models by using *approximate inference* techniques to estimate the latent variables.

Approximate Inference

- Linearisation: Approximate nonlinearities by Taylor series expansion about a point (e.g. the approximate mean or mode of the hidden variable distribution). Linear approximations are particularly useful since Gaussian distributions are closed under linear transformations (e.g., EKF). Also Laplace's approximation.
- Monte Carlo Sampling: Approximate posterior distribution over unobserved variables by a set of random samples. We often need Markov chain Monte carlo or sequential Monte Carlo methods to sample from difficult distributions.
- Variational Methods: Approximate the hidden variable posterior $p(H)$ with a tractable form $q(H)$, such that $\mathbf{KL}[q||p]$ is minimised. This gives a lower bound on the likelihood that can be maximised with respect to the parameters of $q(H)$.
- Local Message Passing Methods: Approximate the hidden variable posterior $p(H)$ with a tractable form $q(H)$ or with a set of locally consistent tractable forms by other means (loopy belief propagation, expectation propagation).
- Recognition Models: Approximate the hidden variable posterior distribution using an explicit *bottom-up* recognition model/network.

References

- Pattern Classification. Duda, Hart and Stork. Wiley, 2000.
- A Unifying Review of Linear Gaussian Models. Roweis and Ghaharamani. Neural Computation, 1999.
- Independent Component Analysis. Hyvarinen, Karhunan and Oja. John Wiley and Sons, 2001.
- Emergence of Simple-Cell Receptive Field Properties by Learning a Sparse Code for Natural Images. Olshausen & Field Nature, 1996.
- A Learning Algorithm for Boltzmann Machines. Ackley, Hinton and Sejnowski. Cognitive Science, 1985.
- Connectionist Learning of Belief Networks. Neal. Artificial Intelligence, 1992.
- Latent Dirichlet Allocation. Blei, Ng and Jordan. Journal of Machine Learning Research, 2003.
- Factorial Hidden Markov Models. Ghahramani and Jordan. Machine Learning, 1997.
- Dynamic Bayesian Networks: Representation, Inference and Learning. Kevin Murphy. PhD Thesis, 2002.

References

- Isomap. Tenenbaum, de Silva & Langford, Science, **290**(5500):2319–23 (2000).
- LLE. Roweis & Saul, Science, **290**(5500):2323–6 (2000).
- Laplacian Eigenmaps. Belkin & Niyogi, Neural Comput **23**(6):1373–96 (2003).
- Hessian LLE. Donoho & Grimes, PNAS **100**(10): 5591–6 (2003).
- Maximum variance unfolding. Weinberger & Saul, Int J Comput Vis **70**(1):77–90 (2006).
- Conformal eigenmaps. Sha & Saul ICML **22**:785–92 (2005).
- SNE Hinton & Roweis, NIPS, 2002; t-SNE van der Maaten & Hinton, JMLR, 9:2579–2605, 2008.
- Gaussian Process Latent Variable Models Lawrence. Advances in Neural Information Processing Systems, 2004.

More at: `http://www.gatsby.ucl.ac.uk/~maneesh/dimred/`