A Tour of Unsupervised Learning – Part I
Graphical models and dimension reduction

Marina Meilă
mmp@stat.washington.edu

Department of Statistics
University of Washington

MPS2016

Supervised vs. Unsupervised Learning

Supervised Learning

Prediction

Data pairs (x_i, y_i) from unknown distribution P_{XY}

Goal learn to predict output y from input x

in probabilistic terms, learn $P_{Y|X}$

Clearly defined objective: minimize cost of error in predicting y

Unsupervised Learning

Description / Analysis / Exploration / Modeling

Data observations x_i from unknown distribution P_X

Goal learn [something about] P_X

- Defining the objective is part of the problem
- Sometimes query or goal not known at the time of learning
- Objective can be defined in multiple ways

Example: Graphical models

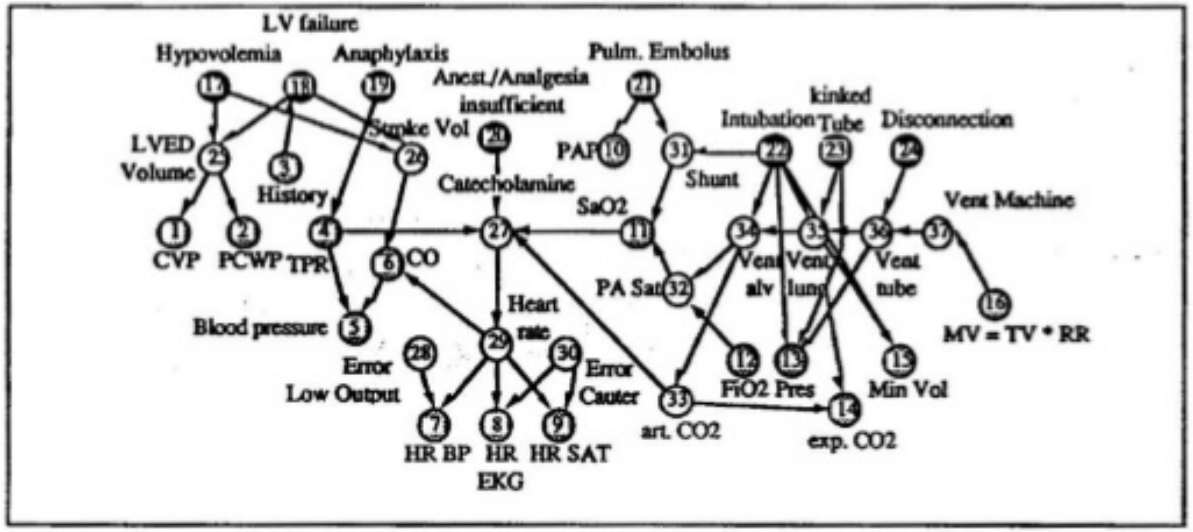
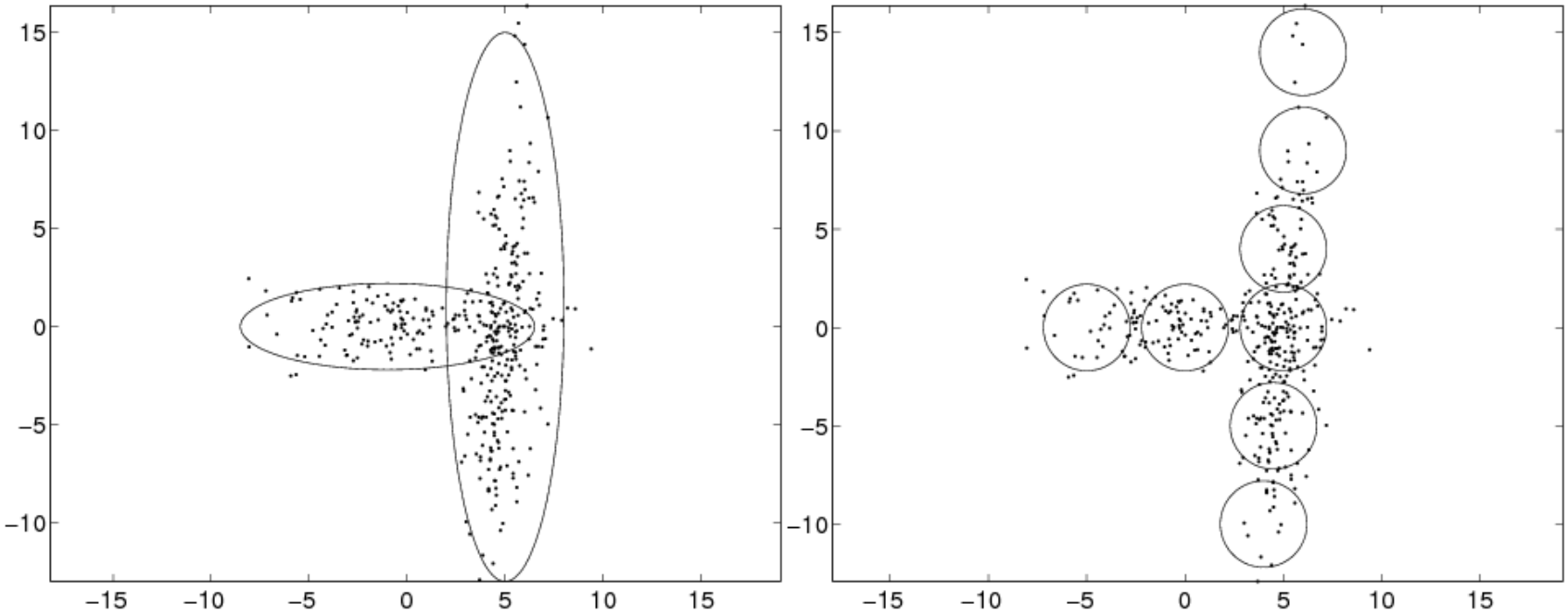


Fig. 1 The ALARM network representing causal relationships is shown with diagnostic (●), intermediate (○) and measurement (◎) nodes. CO: cardiac output, CVP: central venous pressure, LVED volume: left ventricular end-diastolic volume, LV failure: left ventricular failure, MV: minute ventilation, PA Sat: pulmonary artery oxygen saturation, PAP: pulmonary artery pressure, PCWP: pulmonary capillary wedge pressure, Pres: breathing pressure, RR: respiratory rate, TPR: total peripheral resistance, TV: tidal volume

I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper. The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks. In Proceedings of the 2nd European Conference on Artificial Intelligence in Medicine, pages 247-256. Springer-Verlag, 1989

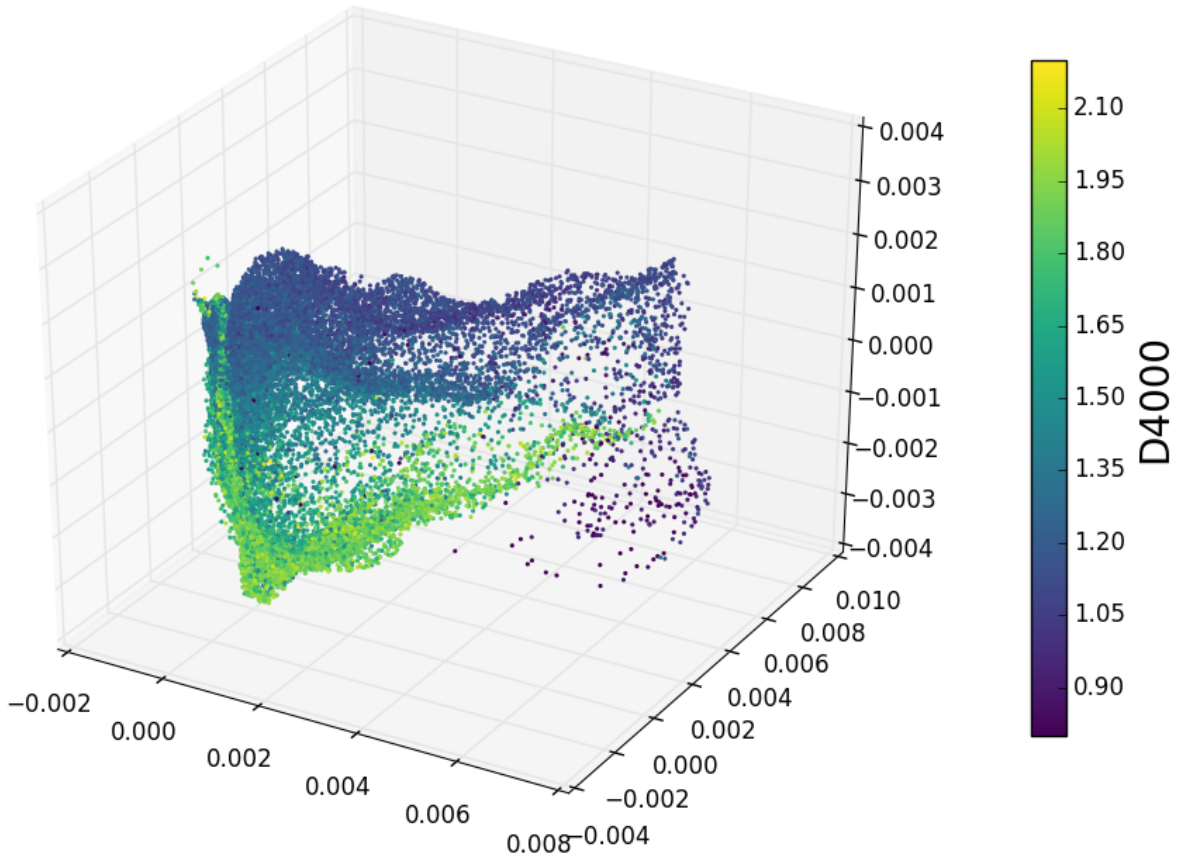
- Graphical model describes joint probabilistic dependence of all variables
- After it is learned, we can ask a variety of questions
  - What is the probability distribution of Pres is high if Kinked Tube is True?
  - What is the most probable failure if PA Sat is observed (and out of limits)?

Example: Clustering



- These data support multiple definitions for clusters)can be grouped in several ways)

Example: Dimension reduction



- Dimension depends on scale in many real data sets

Outline

Graphical probability models: modeling mediated dependencies between many variables

Dimension reduction: linear and non-linear

What is statistical inference and some other definitions

Notations:

$V = \{X_1, X_2, \ldots X_n\}$	the domain
$n = \lvert V \rvert$	the number of variables, or the dimension of the domain
$\Omega(X_i)$	the domain of variable X_i (sometimes denoted Ω_{X_i})
$r_i = \lvert\Omega(X_i)\rvert$	(we assume variables are discrete)
$P_{X_1,X_2,\ldots X_n}$	the joint distribution (sometimes denoted $P(X_1, X_2, \ldots X_n)$
$A, B \subseteq V$	disjoint subsets of variables, $C = V \setminus (AB)$

Example: The "Chest clinic" example - a domain with 4 discrete variables.

Smoker $\in \{Y, N\} = \Omega(S)$

Dyspnoea $\in \{Y, N\} = \Omega(D)$

Lung cancer $\in \{\text{no}, \text{incipient}, \text{advanced}\} = \Omega(L)$

Bronchitis $\in \{Y, N\} = \Omega(B)$

$V = \{S, D, L, B\}$

What is statistical inference and some other definitions

Notations:

$V = \{X_1, X_2, \ldots X_n\}$	the domain
$n = \lvert V \rvert$	the number of variables, or the dimension of the domain
$\Omega(X_i)$	the domain of variable X_i (sometimes denoted Ω_{X_i})
$r_i = \lvert\Omega(X_i)\rvert$	(we assume variables are discrete)
$P_{X_1,X_2,\ldots X_n}$	the joint distribution (sometimes denoted $P(X_1, X_2, \ldots X_n)$
$A, B \subseteq V$	disjoint subsets of variables, $C = V \setminus (AB)$

Example: The "Chest clinic" example - a domain with 4 discrete variables.

Smoker $\in \{Y, N\} = \Omega(S)$

Dyspnoea $\in \{Y, N\} = \Omega(D)$

Lung cancer $\in \{\text{no}, \text{incipient}, \text{advanced}\} = \Omega(L)$

Bronchitis $\in \{Y, N\} = \Omega(B)$

$V = \{S, D, L, B\}$

- $|\Omega_{S,D,L,B}| = 2 \times 2 \times 3 \times 2 = 24$ possible configurations.
- The **joint probability distribution** $P_{SDLB}(s, d, l, b)$ is a real valued function on $\Omega(\{S, D, L, B\}) = \Omega(S) \times \Omega(D) \times \Omega(L) \times \Omega(B)$.
  We sometimes call it a **multidimensional probability table**.

What is statistical inference and some other definitions

Notations:

$V = \{X_1, X_2, \ldots X_n\}$	the domain
$n = \lvert V \rvert$	the number of variables, or the dimension of the domain
$\Omega(X_i)$	the domain of variable X_i (sometimes denoted Ω_{X_i})
$r_i = \lvert \Omega(X_i) \rvert$	(we assume variables are discrete)
$P_{X_1, X_2, \ldots X_n}$	the joint distribution (sometimes denoted $P(X_1, X_2, \ldots X_n)$
$A, B \subseteq V$	disjoint subsets of variables, $C = V \setminus (AB)$

Example: The "Chest clinic" example - a domain with 4 discrete variables.

Smoker $\in \{Y, N\} = \Omega(S)$

Dyspnoea $\in \{Y, N\} = \Omega(D)$

Lung cancer $\in \{\text{no}, \text{incipient}, \text{advanced}\} = \Omega(L)$

Bronchitis $\in \{Y, N\} = \Omega(B)$

$V = \{S, D, L, B\}$

Examples of inferences

- The marginal distribution of S, L is
  $P_{SL}(s, l) = \sum_{d \in \Omega(D)} \sum_{b \in \Omega(B)} P_{SDLB}(s, d, l, b)$

What is statistical inference and some other definitions

Notations:

$V = \{X_1, X_2, \ldots X_n\}$	the domain
$n = \lvert V \rvert$	the number of variables, or the dimension of the domain
$\Omega(X_i)$	the domain of variable X_i (sometimes denoted Ω_{X_i})
$r_i = \lvert \Omega(X_i) \rvert$	(we assume variables are discrete)
$P_{X_1,X_2,\ldots X_n}$	the joint distribution (sometimes denoted $P(X_1, X_2, \ldots X_n)$
$A, B \subseteq V$	disjoint subsets of variables, $C = V \setminus (AB)$

Example: The "Chest clinic" example - a domain with 4 discrete variables.

Smoker $\in \{Y, N\} = \Omega(S)$

Dyspnoea $\in \{Y, N\} = \Omega(D)$

Lung cancer $\in \{\text{no}, \text{incipient}, \text{advanced}\} = \Omega(L)$

Bronchitis $\in \{Y, N\} = \Omega(B)$

$V = \{S, D, L, B\}$

Examples of inferences

- The marginal distribution of S, L is
  $P_{SL}(s, l) = \sum_{d \in \Omega(D)} \sum_{b \in \Omega(B)} P_{SDLB}(s, d, l, b)$
- The conditional distribution of Lung cancer given Smoking is
  $P_{L|S}(l|s) = \frac{P_{SL}(s,l)}{P_S(s)}$

What is statistical inference and some other definitions

Notations:

$V = \{X_1, X_2, \ldots X_n\}$	the domain
$n = \lvert V \rvert$	the number of variables, or the dimension of the domain
$\Omega(X_i)$	the domain of variable X_i (sometimes denoted Ω_{X_i})
$r_i = \lvert \Omega(X_i) \rvert$	(we assume variables are discrete)
$P_{X_1, X_2, \ldots X_n}$	the joint distribution (sometimes denoted $P(X_1, X_2, \ldots X_n)$
$A, B \subseteq V$	disjoint subsets of variables, $C = V \setminus (AB)$

Example: The "Chest clinic" example - a domain with 4 discrete variables.

Smoker $\in \{Y, N\} = \Omega(S)$

Dyspnoea $\in \{Y, N\} = \Omega(D)$

Lung cancer $\in \{\text{no}, \text{incipient}, \text{advanced}\} = \Omega(L)$

Bronchitis $\in \{Y, N\} = \Omega(B)$

$V = \{S, D, L, B\}$

Examples of inferences

- The marginal distribution of S, L is
  $P_{SL}(s, l) = \sum_{d \in \Omega(D)} \sum_{b \in \Omega(B)} P_{SDLB}(s, d, l, b)$
- The conditional distribution of Lung cancer given Smoking is
  $P_{L|S}(l|s) = \frac{P_{SL}(s,l)}{P_S(s)}$

Statistical inference (in the model P_{SDLB}) = computing the probabilities of some variables of interest (L) when we observe others (S) and we don't know anything about the rest (B, D)

The curse of dimensionality

Notations:

$V = \{X_1, X_2, \ldots X_n\}$	the domain
$n = \lvert V \rvert$	the number of variables, or the dimension of the domain
$\Omega(X_i)$	the domain of variable X_i (sometimes denoted Ω_{X_i})
$r_i = \lvert \Omega(X_i) \rvert$	(we assume variables are discrete)
$P_{X_1, X_2, \ldots X_n}$	the joint distribution (sometimes denoted $P(X_1, X_2, \ldots X_n)$
$A, B \subseteq V$	disjoint subsets of variables, $C = V \setminus (AB)$

Inferences in a domain with n (discrete) variables are exponential in n

- marginal distribution of S, L is $P_{SL}(s, l) = \sum_{d \in \Omega(D)} \sum_{b \in \Omega(B)} P_{SDLB}(s, d, l, b)$ involves summation over $\Omega(B) \times \Omega(D)$
- conditional distributions are ratios of two marginals: $P_{L|S}(l|s) = \frac{P_{SL}(s,l)}{P_S(s)}$
- if P_V is unnormalized, computing Z requires summation over $\Omega(V)$.

The curse of dimensionality

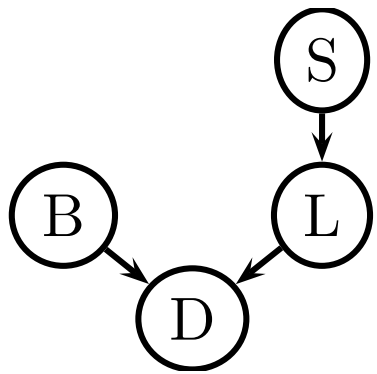
Notations:

$V = \{X_1, X_2, \ldots X_n\}$	the domain
$n = \lvert V \rvert$	the number of variables, or the dimension of the domain
$\Omega(X_i)$	the domain of variable X_i (sometimes denoted Ω_{X_i})
$r_i = \lvert\Omega(X_i)\rvert$	(we assume variables are discrete)
$P_{X_1,X_2,\ldots X_n}$	the joint distribution (sometimes denoted $P(X_1, X_2, \ldots X_n)$
$A, B \subseteq V$	disjoint subsets of variables, $C = V \setminus (AB)$

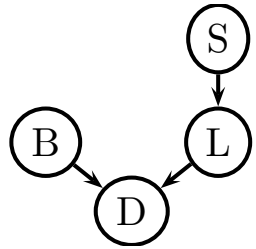
Inferences in a domain with n (discrete) variables are exponential in n

- marginal distribution of S, L is $P_{SL}(s, l) = \sum_{d \in \Omega(D)} \sum_{b \in \Omega(B)} P_{SDLB}(s, d, l, b)$ involves summation over $\Omega(B) \times \Omega(D)$
- conditional distributions are ratios of two marginals: $P_{L|S}(l|s) = \frac{P_{SL}(s,l)}{P_S(s)}$
- if P_V is unnormalized, computing Z requires summation over $\Omega(V)$.

Can we do better? Yes, sometimes



Graphical models



- Are a language for encoding dependence/independence relations between variables
- Are a language for representing classes of probability distributions that satisfy these independencies
- Support generic algorithms for inference.

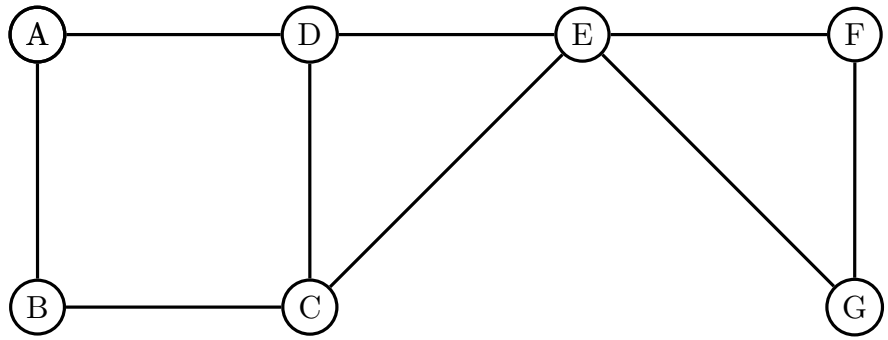
  Statistical inference = computing $P(X = x|Y = y)$
- The algorithms work by performing local computations
  - i.e "message passing/belief propagation"
  - involve only a node and its "neighbohood" in the graph
- Efficient and general algorithms for estimating the model parameters (by e.g Maximum Likelihood)

Moreover

- Powerful models (can represent a very rich class of distributions)
- Can represent distributions compactly
- Are intuitive

- Graph structure can be known (e.g from physics) or estimated from data

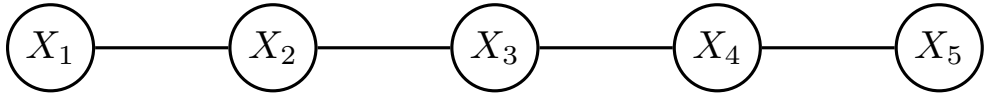
Example: Ising model



- Energy $E_{ABC\ldots G} = E_{AB} + E_{AD} + E_{BC} + \ldots E_{FG}$
- Joint distribution $P_{ABC\ldots G} = \frac{1}{Z(\beta)} \exp\left[\frac{1}{\beta}(E_{AB} + E_{AD} + E_{BC} + \ldots E_{FG})\right]$

  i.e. $P_{ABC\ldots G}$ factors over the edges of the graph

Example: Markov chain

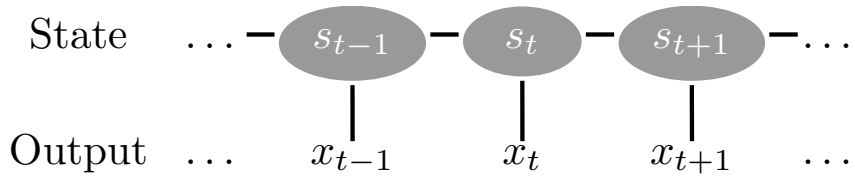


- The joint distribution

$$P_{X_1X_2X_3X_4X_5} = (P_{X_1}P_{X_2|X_1})P_{X_3|X_2}P_{X_4|X_3}P_{X_5|X_4}$$

- Parameters $P[X_{t+1} = i|X_t = j] = T_{ij}$ for $i, j \in \Omega(X)$
- $T = [T_{ij}]_{i,j\in\Omega}$ is the transition matrix of the Markov chain

Example: Hidden Markov Model

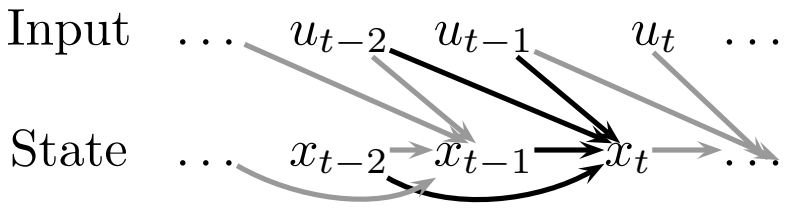


- The joint distribution

$$P_{X_1X_2X_3X_4X_5S_1S_2S_3S_4S_5} = (P_{S1}P_{S_2|S_1})P_{S_3|S_2}P_{S_4|S_3}P_{S_5|S_4}P_{X_1|S_1}P_{X_2|S_2}P_{X_3|S_3}P_{X_4|S_4}P_{X_5|S_5}$$

- Parameters
  - $T = [T_{ij}]_{i,j\in\Omega}$ the transition matrix of the Markov chain
  - Parameters of $P_{X_t|S_t}$ =output probabilities β
  - $S_1, S_2, \ldots$ are hidden variables, $X_1, X_2, \ldots$ are observed variables

Example: ARMA model

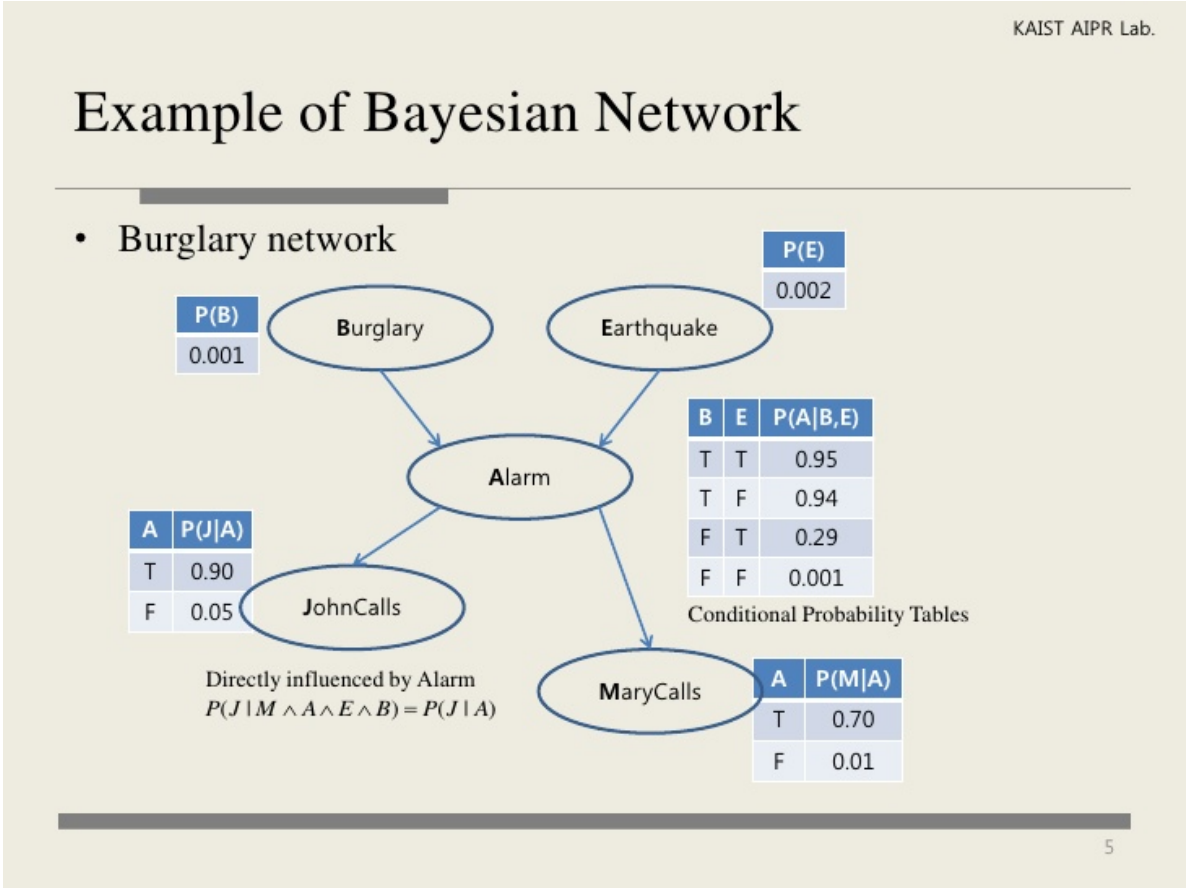


hi

- $x_t = \alpha_1 x_{t-1} + \alpha_2 x_{t-2} + \ldots + \beta_1 u_{t-1} + + \beta_2 u_{t-2} + \ldots + \epsilon_t$
- u_t are inputs, ϵ_t noise

- Directed graph: Bayes net

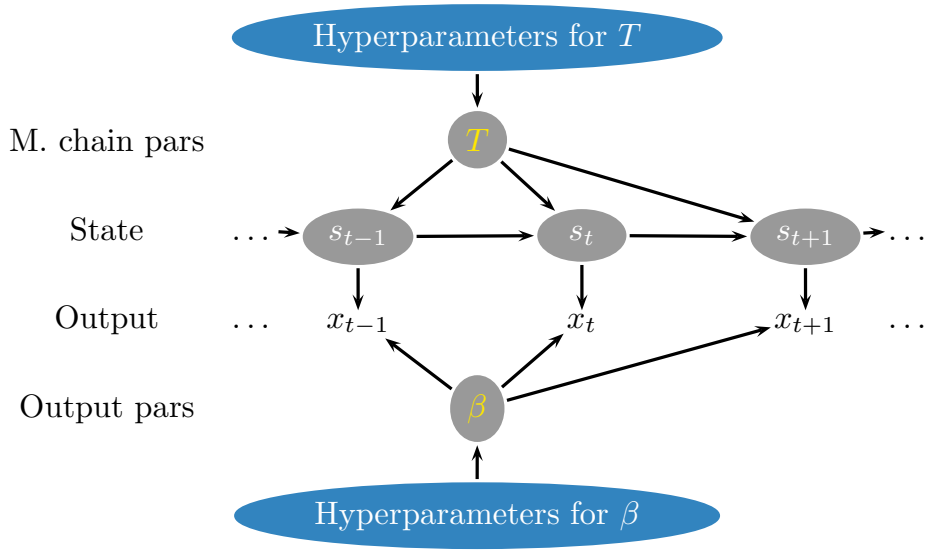
Example: Earthquake and the burglar alarm



(figure by Jung-Yeol Lee, KAIST)

▶

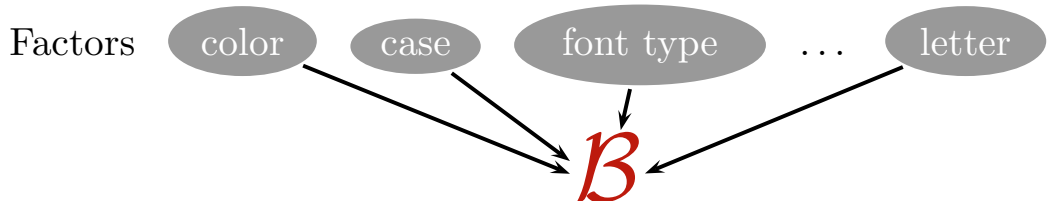
Bayesian Framework



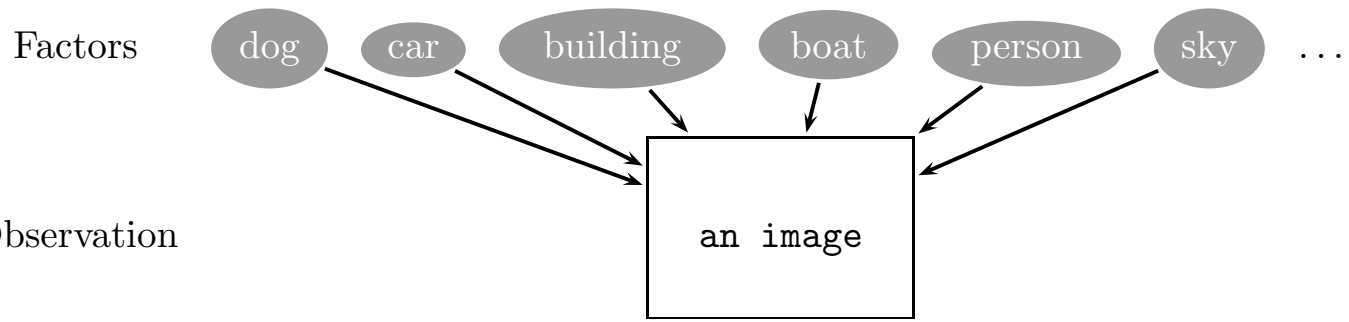
- We can assume that the model parameters are also random variables.
- We represent our prior knowledge by probability distributions over the parameters
  - If we have no knowledge, the prior distribution is uniform
  - The parameters of the prior distribution are called hyper-parameters
- Bayesian inference: calculate the posterior distribution of the unobserved variables given observed and hyperparameters
- uses Bayes's rule

More models with hidden variables

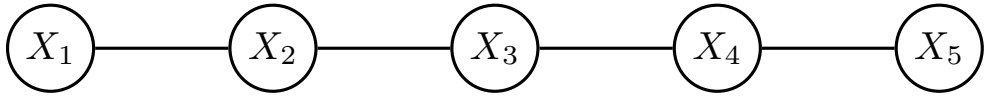
- Factorial model



- Sparse coding/population coding



Inference in graphical models – an example



- The joint distribution

$$P_{X_1 X_2 X_3 X_4 X_5} = P_{X_1} P_{X_2|X_1} P_{X_3|X_2} P_{X_4|X_3} P_{X_5|X_4}$$

- A query: What is $P_{X_4|X_1=\text{true}}$?
- Answer is recursive

$$P_{X_2} = TP_{X_1} \quad P_{X_3} = TP_{X_2} \quad P_{X_4} = TP_{X_3}$$

- each step involves only nodes that are neighbors

Message Passing

- Message from variable A to factor F

$$m_{A \to F}(a) = \prod_{F' \neq F} m_{F' \to A}(a)$$

  is a function on $\Omega(A)$.

- Message from factor F to variable A

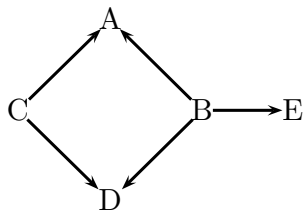
$$m_{F \to A}(a) = \sum_{\Omega(F):A=a} F(a, \ldots) \prod_{B \in F \setminus A} m_{B \to F}(b)$$

  is a function on $\Omega(A)$.

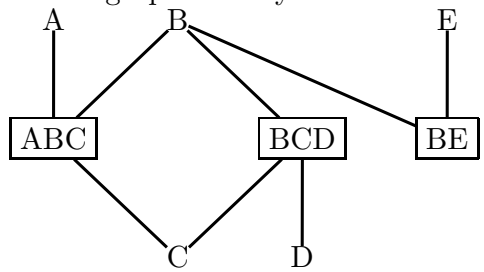
- Message = multiply all potentials covering A or F, then marginalize out all variables not in the target (Sum-Product Algorithm)
- Message $A \to F$ or $F \to A$ can be sent only after messages from all other neighbors are received

Graphical models as factor graphs

Bayes net

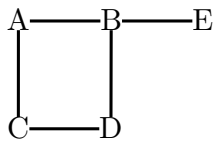


Factor graph for Bayes net

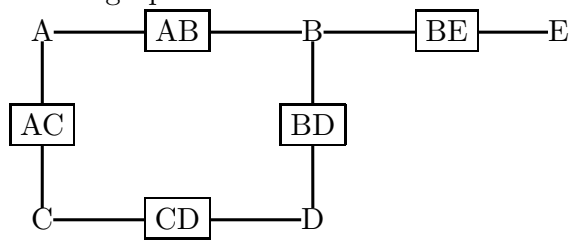


$$P_{ABCDE} = P_C P_B P_{E|B} P_{A|BC} P_{D|BC}$$

Markov field



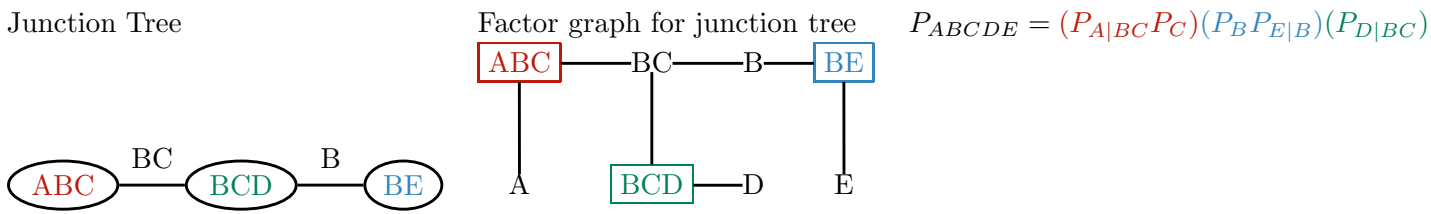
Factor graph for Markov Field



$$P_{ABCDE} = \phi_{AC}\phi_{AB}P_{BE}\phi_{BD}\phi_{CD}$$

- representation that reflects computation flow
- each box is a factor in the expression of the joint distribution
  - i.e a term in the energy
- factors involving a single variable are not shown
- factors involving the same set of variables are collapsed into one

Eliminating loops



- Group variables into "super-variables", so that graph over these has no cycles
  - process is called triangulation (edges are added if necessary)
  - super-variables are maximal cliques in the triangulated graph

The computational complexity of inference

- There exist automated algorithms to perform inference in graphical models of any structure
- However, the amount of computation (and storage) required depends on the graph structure
- In general, the computation required for inference grows exponentially with the number of variables that must be considered together at one step of belief propagation
- In general, the computation required for inference grows with the state space of variables that must be considered together at one step of belief propagation
- This is at least as large as the size of the largest factor
-

Approximate Inference

- Loopy Belief Propagation (Bethe, Kikuchi free energy)
- Variational Inference (mean-field approximation)
- Markov Chain Monte Carlo (MCMC) (Metropolis)
- Arithmetic circuits (exact or approximate)

Modern inference and arithmetic circuits

- Specifically for discrete (e.g Boolean) variables
- Associate to graphical model a multivariate polinomial $f(x_V, \theta)$, such that $P_V = f$
- Marginalizing and conditioning are equivalent to taking derivatives of f or $\ln f$, and setting the observed variables

$$P_V(X = x|E = e) = \frac{1}{f(e)} \frac{\partial f}{\partial x}$$

[Darwiche 03]

Outline

Graphical probability models: modeling mediated dependencies between many variables

Dimension reduction: linear and non-linear

Dimension reduction

- The problem:
  - Given data $\{x_1, x_2, \ldots x_n\} \in \mathbb{R}^D$
  - Output $\{y_1, y_2, \ldots y_n\} \in \mathbb{R}^s$ with $s \ll D$

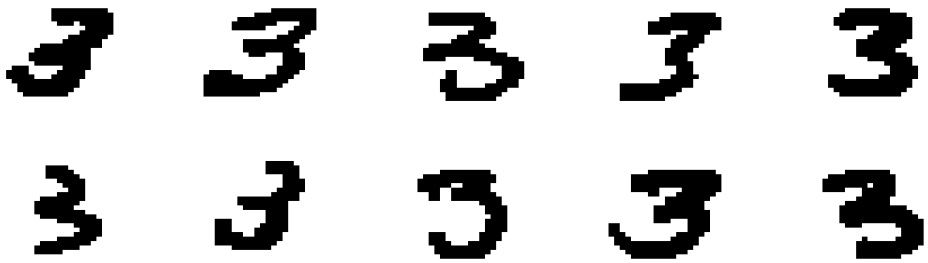
Dimension reduction

- The problem:
  - Given data $\{x_1, x_2, \ldots x_n\} \in \mathbb{R}^D$
  - Output $\{y_1, y_2, \ldots y_n\} \in \mathbb{R}^s$ with $s \ll D$

    so that $\{y_1, y_2, \ldots y_n\}$ preserve as much [topologic/geometric/ldots] information as possible about $\{x_1, x_2, \ldots x_n\}$

- Dimension reduction is a form of lossy compression
- embedding dimension s trades off compression vs. accuracy

When to do (non-linear) dimension reduction



- Very high dimensional data $x \in \mathbb{R}^D$
- Can be well described by a few parameters
- Usually, large sample size n

Why?

- To save space
- To understand the data better
- To use it afterwards in (prediction) tasks

Why non-linear?

- A celebrated method for dimension reduction

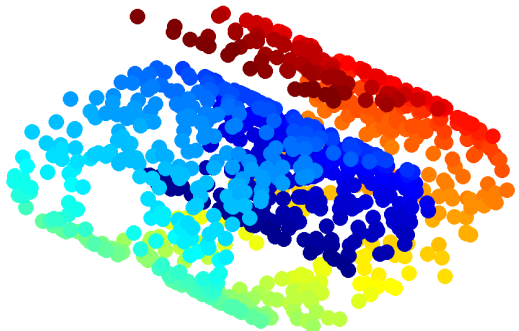
Why non-linear?

- A celebrated method for dimension reduction
  - Principal Component Analysis (PCA)
  - Computes covariance matrix Σ of data
  - Projects data on d principal subspace of Σ

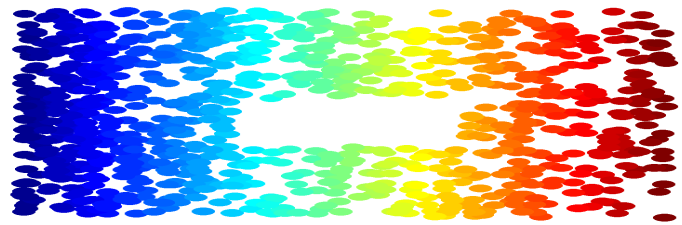
Why non-linear?

- A celebrated method for dimension reduction
  - Principal Component Analysis (PCA)
  - But: PCA is linear operation

points in $D \geq 3$ dimensions



same points reparametrized in $d = 2$

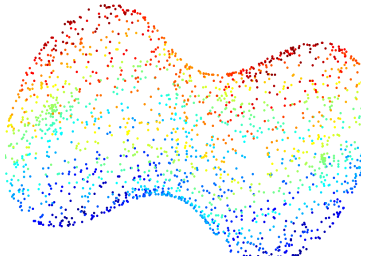


(the "Swiss Roll" with a hole)

- "Unfolding" the "swiss roll" is a problem in non-linear dimension reduction
- One way to solve it is by manifold learning

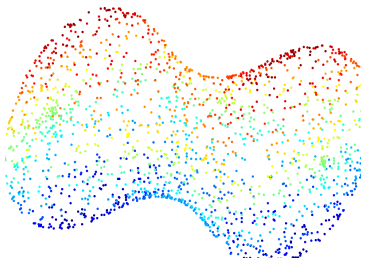
Blueprint to manifold learning algorithms

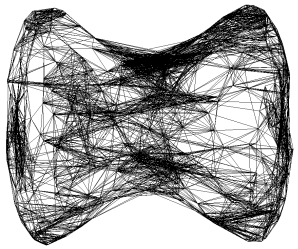
- **Input** Data $\mathcal{D} = \{p_1, \ldots p_n\} \subset \mathbb{R}^D$, embedding dimension s ($\epsilon =$ neighborhood scale parameter) (may use ϵ again)



Blueprint to manifold learning algorithms

- **Input** Data $\mathcal{D} = \{p_1, \ldots p_n\} \subset \mathbb{R}^D$, embedding dimension s
- Construct neighborhood graph p, p' neighbors iff $||p - p'||^2 \leq \epsilon$ ($\epsilon =$ neighborhood scale parameter) (may use ϵ again)

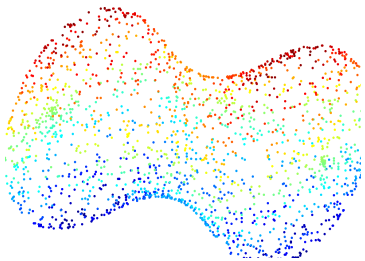
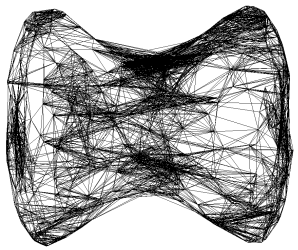
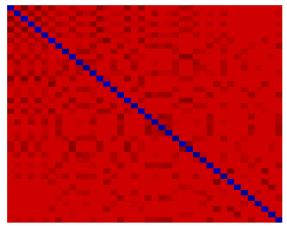




Blueprint to manifold learning algorithms

- **Input** Data $\mathcal{D} = \{p_1, \ldots p_n\} \subset \mathbb{R}^D$, embedding dimension s
- Construct neighborhood graph p, p' neighbors iff $||p - p'||^2 \leq \epsilon$ ($\epsilon =$ neighborhood scale parameter)
- Construct similarity matrix $S = [S_{pp'}]_{p,p' \in \mathcal{D}}$

  $S_{pp'} = e^{-\frac{1}{\epsilon}||p-p'||^2}$ iff p, p' neighbors (may use ϵ again)

Blueprint to manifold learning algorithms

- **Input** Data $\mathcal{D} = \{p_1, \dots p_n\} \subset \mathbb{R}^D$, embedding dimension s
- Construct neighborhood graph p, p' neighbors iff $||p - p'||^2 \leq \epsilon$ ($\epsilon =$ neighborhood scale parameter)
- Construct similarity matrix $S = [S_{pp'}]_{p,p' \in \mathcal{D}}$
  $S_{pp'} = e^{-\frac{1}{\epsilon}||p-p'||^2}$ iff p, p' neighbors (may use ϵ again)

Diffusion Maps/Laplacian Eigenmaps [Nadler et al. 05], [Belkin, Nyogi 02]

- Construct graph Laplacian matrix $L = I - T^{-1}S$ with $T = \mathrm{diag}(S\mathbf{1})$ (transform S)
- Calculate $\psi^{1\dots m}$ = eigenvectors of L (smallest eigenvalues)
- coordinates of $p \in \mathcal{D}$ are $(\psi^1(p), \dots \psi^m(p))$

Blueprint to manifold learning algorithms

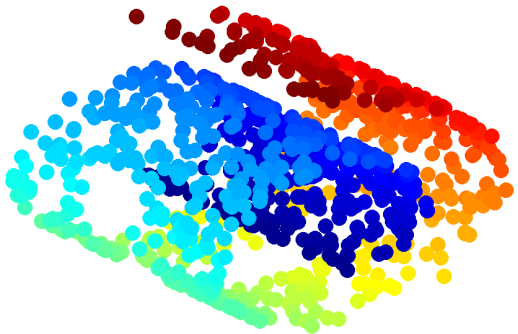
- **Input** Data $\mathcal{D} = \{p_1, \ldots p_n\} \subset \mathbb{R}^D$, embedding dimension s
- Construct neighborhood graph p, p' neighbors iff $||p - p'||^2 \leq \epsilon$ ($\epsilon =$ neighborhood scale parameter)
- Construct similarity matrix $S = [S_{pp'}]_{p,p' \in \mathcal{D}}$
  $S_{pp'} = e^{-\frac{1}{\epsilon}||p-p'||^2}$ iff p, p' neighbors (may use ϵ again)

ISOMAP [Tennenbaum, deSilva & Langford 00]

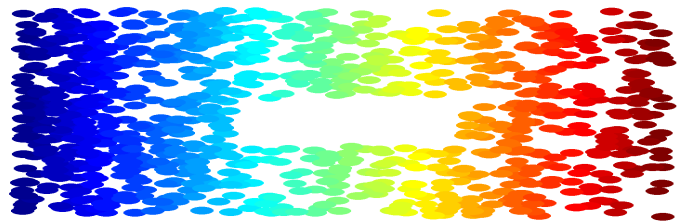
- Find all shortest paths in neighborhood graph $\Rightarrow$ $M = [d^2_{pp'}]$ (transform the distance matrix)
- use M and Multi-Dimensional Scaling (MDS) to obtain m dimensional coordinates for $p \in \mathcal{D}$
  (also uses principal eigenvectors of some matrix)

The "Swiss Roll" with a hole, again

points in $D \geq 3$ dimensions



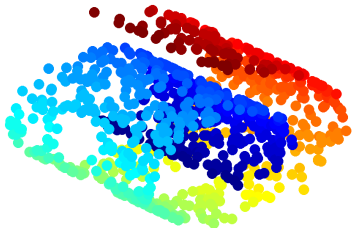
same points reparametrized in 2D



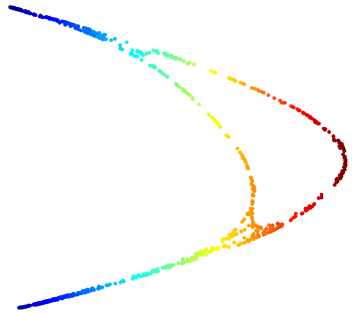
. . . and a problem. . .

Embedding in 2 dimensions by different manifold learning algorithms

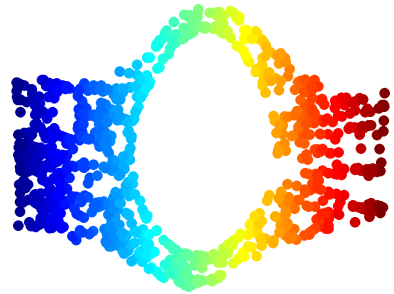
Original data
(Swiss Roll with hole)



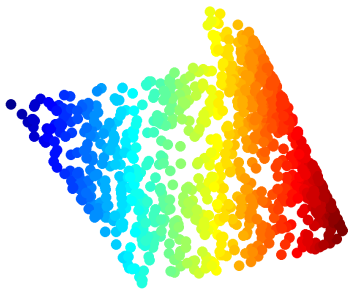
Laplacian Eigenmaps (LE)



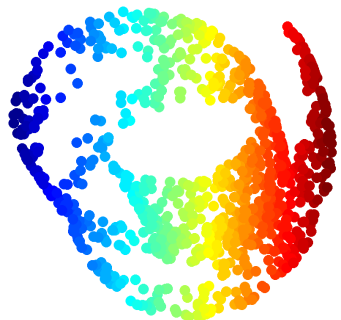
Isomap



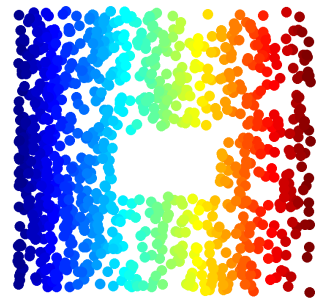
Hessian Eigenmaps (HE)



Local Linear Embedding
(LLE)



Local Tangent Space
Alignment (LTSA)



Preserving topology vs. preserving (intrinsic) geometry

- Algorithm maps data $p \in \mathbb{R}^D \longrightarrow \phi(p) = x \in \mathbb{R}^m$

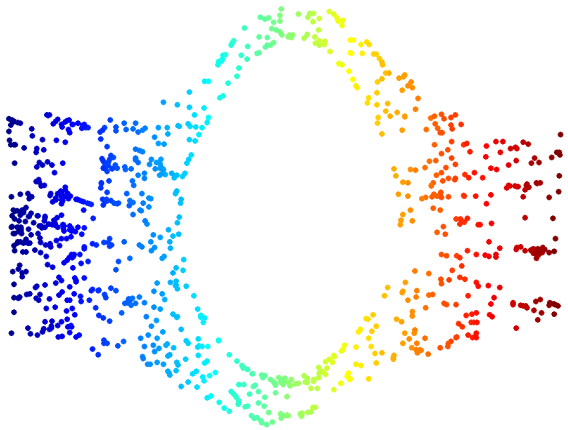
- Mapping $\mathcal{M} \longrightarrow \phi(\mathcal{M})$ is diffeomorphism
  - preserves topology
  - often satisfied by embedding algorithms
- Mapping ϕ preserves
  - distances along curves in $\mathcal{M}$
  - angles between curves in $\mathcal{M}$
  - areas, volumes

Preserving topology vs. preserving (intrinsic) geometry

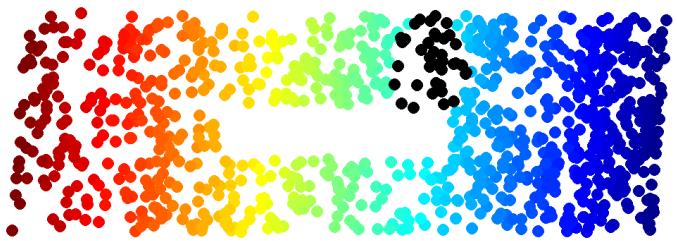
- Algorithm maps data $p \in \mathbb{R}^D \longrightarrow \phi(p) = x \in \mathbb{R}^m$

- Mapping $\mathcal{M} \longrightarrow \phi(\mathcal{M})$ is diffeomorphism
  preserves topology
  often satisfied by embedding algorithms
- Mapping ϕ preserves
  - distances along curves in $\mathcal{M}$
  - angles between curves in $\mathcal{M}$
  - areas, volumes
  ...i.e. ϕ is isometry
  For most algorithms, in most cases, ϕ is not isometry

Preserves topology



Preserves topology + intrinsic geometry



Algorithm to estimate Riemann metric g

Given dataset $\mathcal{D}$

1. Preprocessing (construct neighborhood graph, ...)
2. Obtain low dimensional embedding ϕ of $\mathcal{D}$ into $\mathbb{R}^m$ ✓

Algorithm to estimate Riemann metric g

Given dataset $\mathcal{D}$

1. Preprocessing (construct neighborhood graph, ...)
2. Obtain low dimensional embedding ϕ of $\mathcal{D}$ into $\mathbb{R}^m$ ✓
3. Estimate discretized Laplace-Beltrami operator L ✓

Algorithm to estimate Riemann metric g

Given dataset $\mathcal{D}$

1. Preprocessing (construct neighborhood graph, ...)
2. Obtain low dimensional embedding ϕ of $\mathcal{D}$ into $\mathbb{R}^m$ ✓
3. Estimate discretized Laplace-Beltrami operator L ✓
4. Estimate G_p for all p

   4.1 For $i, j = 1 : m$, (First estimate H_p the inverse of G_p)
   $$H_{ij} = \frac{1}{2}\left[L(\phi_i * \phi_j) - \phi_i * (L\phi_j) - \phi_j * (L\phi_i)\right]$$
   where $X * Y$ denotes elementwise product of two vectors $X, Y \in \mathbb{R}^N$

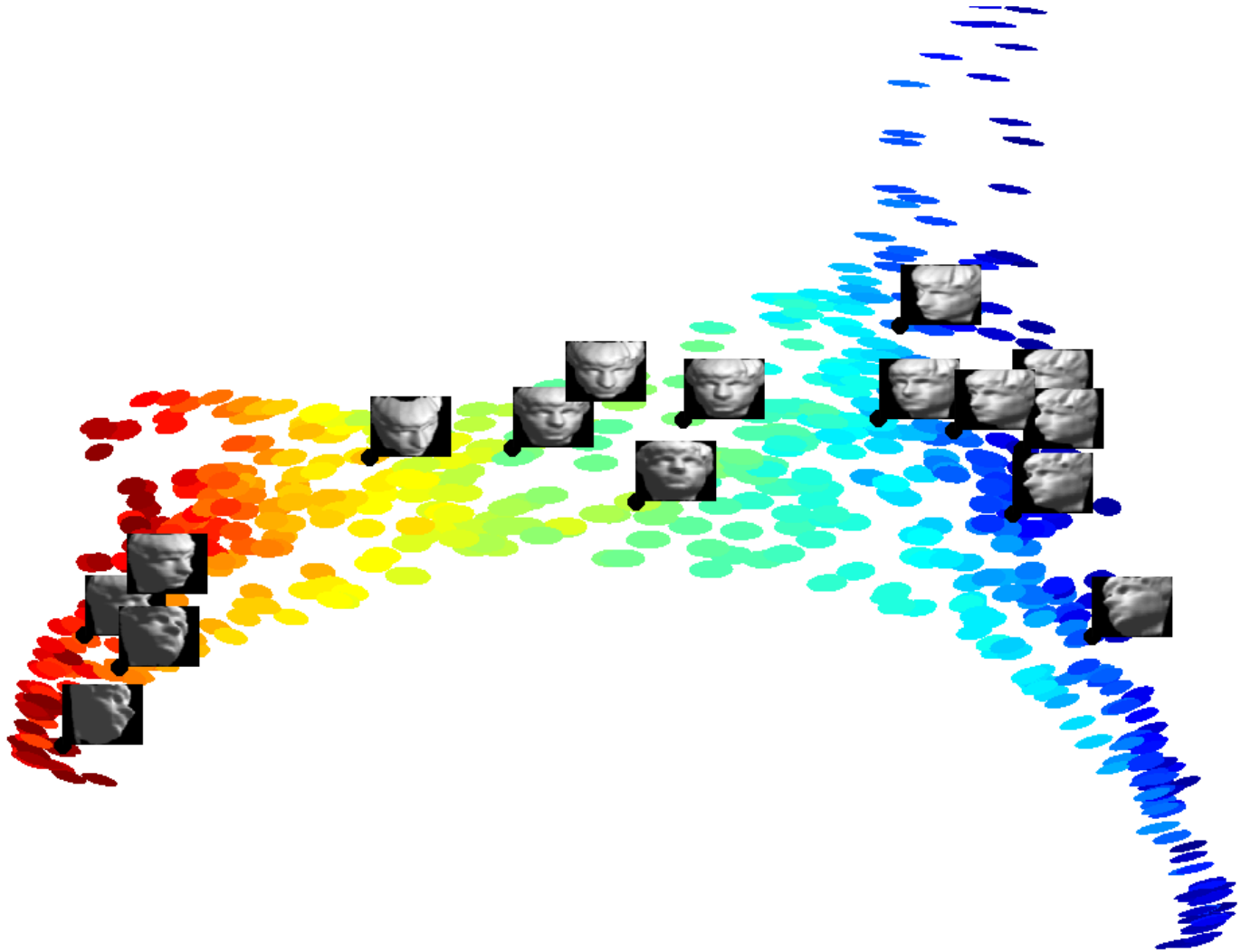
   4.2 For $p \in \mathcal{D}$, invert $H_p = [(H_{ij})_p]$ to obtain G_p

   (Note that this adds about nm^3 operations for each point)

Output (ϕ_p, G_p) for all p

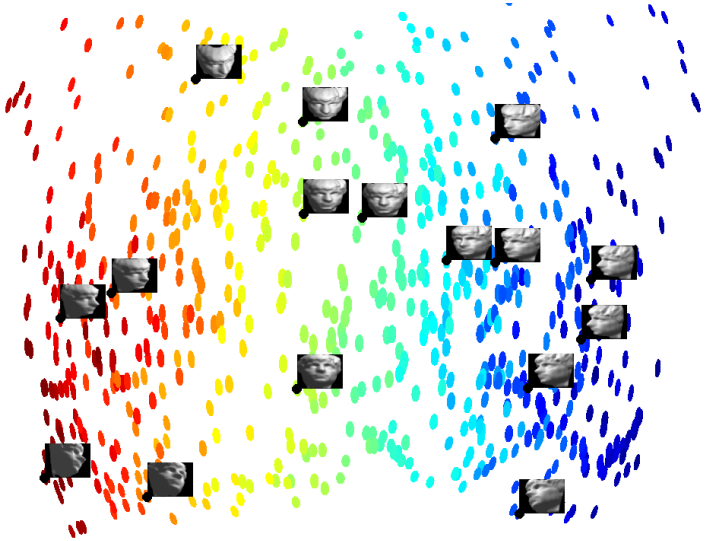
Sculpture Faces [Tenenbaum et al., 2006]

- $n = 698$ with 64×64 gray images of faces
  - head moves up/down and right/left
- With only two degrees of freedom, the faces define a 2D manifold in the space of all 64×64 gray images

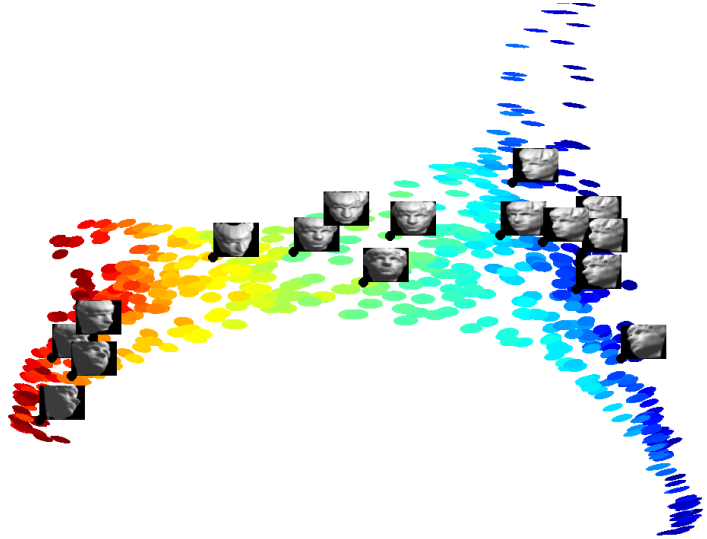


LTSA

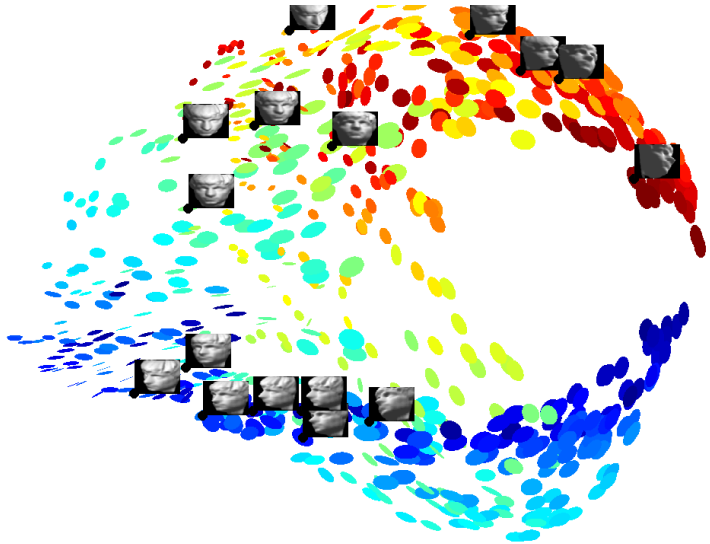
Scupture faces [Tenenbaum et al., 2006]



Isomap



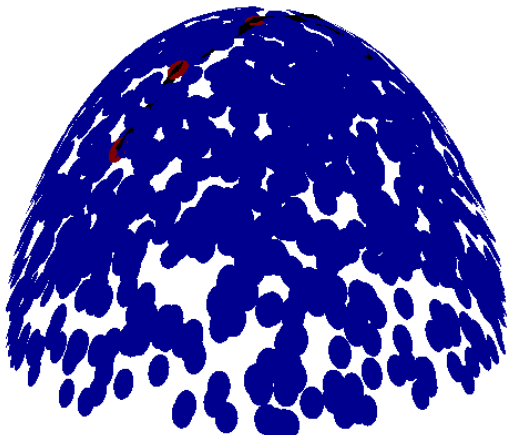
LTSA



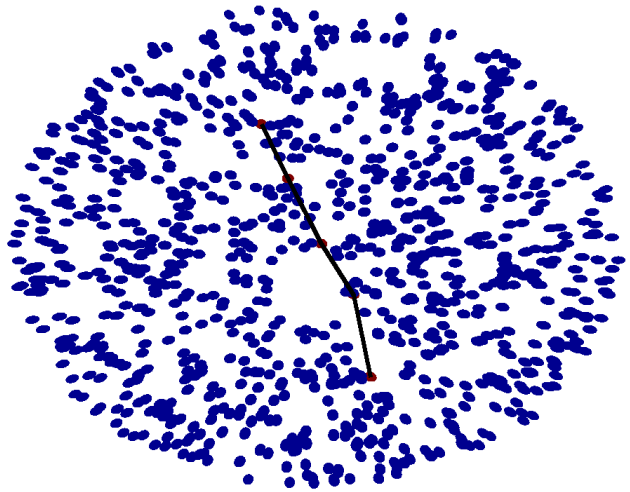
Laplacian Eigenmaps

Calculating distances in the manifold $\mathcal{M}$

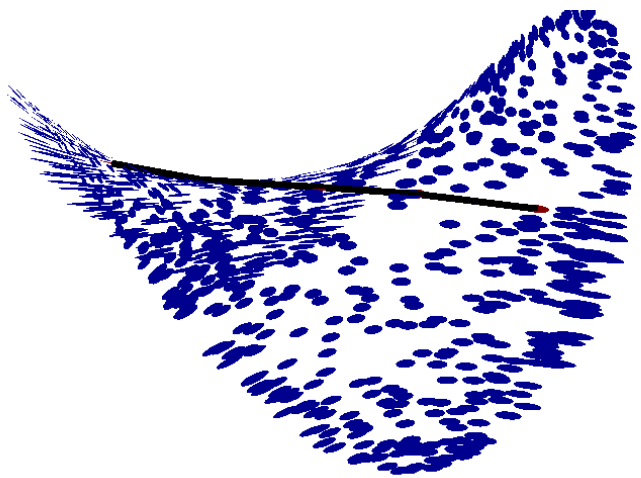
- Geodesic distance = shortest path on $\mathcal{M}$
- should be invariant to coordinate changes



Original



Isomap



Laplacian Eigenmaps

Calculating distances in the manifold $\mathcal{M}$

true distance $d = 1.57$

Embedding	$\|\|f(p) - f(p')\|\|$	Shortest Path $d_{\mathcal{G}}$	Metric $\hat{d}$	Rel. error
Original data	1.41	1.57	1.62	3.0%
Isomap $s = 2$	1.66	1.75	1.63	3.7%
LTSA $s = 2$	0.07	0.08	1.65	4.8%
LE $s = 3$	0.08	0.08	1.62	3.1%

Calculating Areas/Volumes in the manifold

(Results for Hourglass data)

true area $= 0.84$

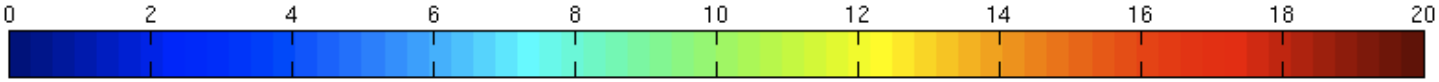
Embedding	Naive	Metric	Rel. err.
Original data	0.85 (0.03)	0.93 (0.03)	11.0%
Isomap	2.7	0.93 (0.03)	11.0%
LTSA	1e-03 (5e-5)	0.93 (0.03)	11.0%
LE	1e-05 (4e-4)	0.82 (0.03)	2.6%

An Application: Gaussian Processes on Manifolds

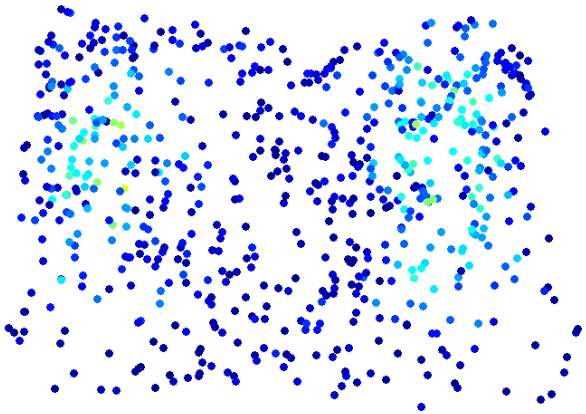
- Gaussian Processes (GP) can be extended to manifolds via SPDE's (Lindberg, Rue, and Lindstrom, 2011)
- Let $\left(\kappa^2 - \Delta_{\mathbb{R}^d}\right)^{\alpha/2} u(x) = \xi$ with ξ Gaussian with noise, then $u(x)$ is a Matérn GP
- Subsituting $\Delta_{\mathcal{M}}$ allows us to define a Matérn GP on $\mathcal{M}$
- Semi-supervised learning: unlabelled points can be learned by Kriging using the Covariance matrix Σ of $u(x)$

Semisupervised learning

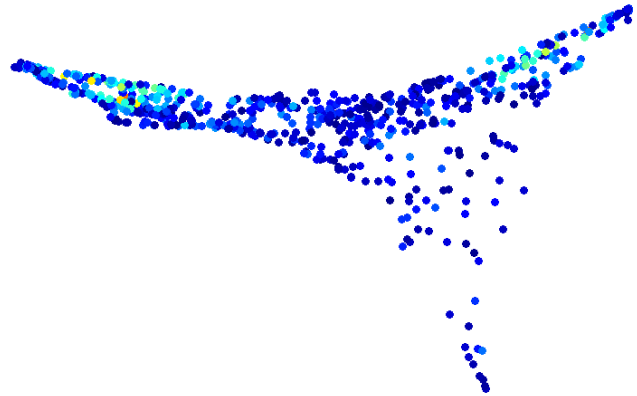
Sculpture Faces: Predicting Head Rotation



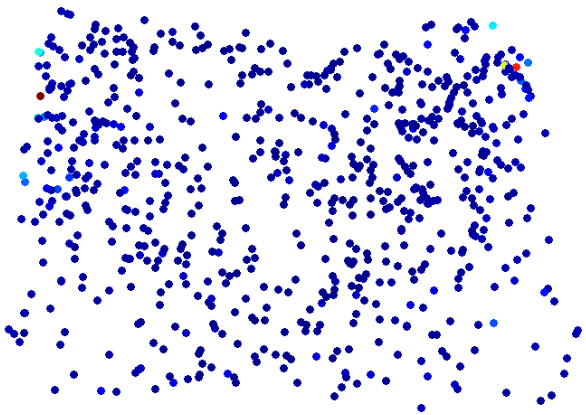
Absolute Error (AE) as percentage of range



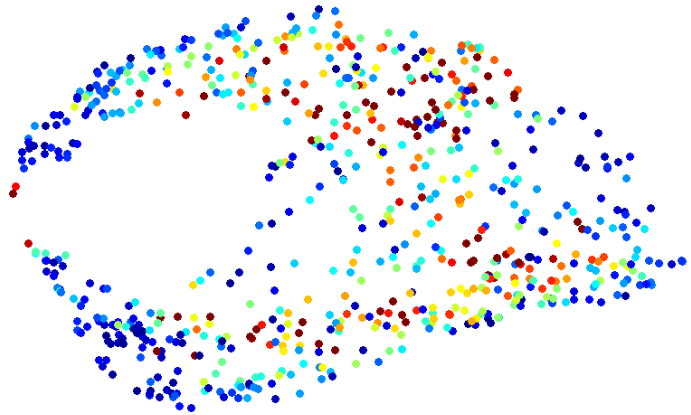
Isomap (Total AE = 3078)



LTSA (Total AE = 3020)



Metric (Total AE = 660)



Laplacian Eigenmaps (Total AE = 3078)

How to choose the neighborhood scale ϵ?

- This problem is related to the choice of s
- and an active area of research

- geometric method [Perrault-Joncas,Meila 14]
- topological methods using persistent homology and persistence diagrams

Self-consistent method of chosing ϵ

- Every manifold learning algorithm starts with a neighborhood graph
- Parameter $\sqrt{\epsilon}$
  - is neighborhood radius
  - and/or kernel banwidth

Self-consistent method of chosing ϵ

- Every manifold learning algorithm starts with a neighborhood graph
- Parameter $\sqrt{\epsilon}$
  - is neighborhood radius
  - and/or kernel banwidth

- For example,
  $S_{pp'} = e^{-\frac{||p-p'||^2}{\epsilon}}$ if $||p - p'||^2 \leq \epsilon$ and 0 otherwise

Self-consistent method of chosing ϵ

- Every manifold learning algorithm starts with a neighborhood graph
- Parameter $\sqrt{\epsilon}$
  - is neighborhood radius
  - and/or kernel banwidth

- For example,
  $S_{pp'} = e^{-\frac{||p-p'||^2}{\epsilon}}$ if $||p - p'||^2 \leq \epsilon$ and 0 otherwise

- Problem: how to choose ϵ?

Self-consistent method of chosing ϵ

- Every manifold learning algorithm starts with a neighborhood graph
- Parameter $\sqrt{\epsilon}$
  - is neighborhood radius
  - and/or kernel banwidth

- For example,
  $S_{pp'} = e^{-\frac{||p-p'||^2}{\epsilon}}$ if $||p - p'||^2 \leq \epsilon$ and 0 otherwise

- Problem: how to choose ϵ?
- Our idea for each ϵ, use Riemannian metric G to measure distortion. Choose the ϵ with minimum distortion.

Self-consistent method of chosing ϵ

- Every manifold learning algorithm starts with a neighborhood graph
- Parameter $\sqrt{\epsilon}$
  - is neighborhood radius
  - and/or kernel banwidth

- For example,
  $S_{pp'} = e^{-\frac{||p-p'||^2}{\epsilon}}$ if $||p - p'||^2 \leq \epsilon$ and 0 otherwise

- Problem: how to choose ϵ?
- Our idea for each ϵ, use Riemannian metric G to measure distortion. Choose the ϵ with minimum distortion.
- Interesting This can also be done independently of any particular embedding.

So, how well does it work?

Semisupervised learning benchmarks [Chapelle&al 08] Multiclass classification problems

Classification error (%)

		Method	
Dataset	**CV**	[Chen&Buja]	**Ours**
Digit1	3.32	2.16	2.11
USPS	5.18	4.83	3.89
COIL	7.02	8.03	8.81
g241c	13.31	23.93	12.77
g241d	8.67	18.39	8.76
	superv.	fully unsupervised	

How many dimensions s are needed?

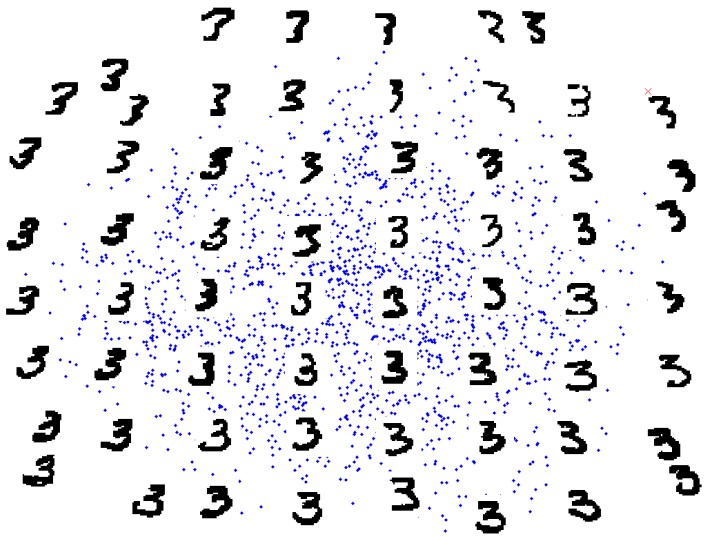
- PCA ("variance explained")
  - $Var(X) \geq Var(Y)$ always
  - $Var(Y)$ increases with s
  - choose s so that $Var(Y) \approx 0.95 Var(X)$

Non-linear

- Dimension estimation is an active area of research.
- A simple method: doubling dimension (next slide)
- More expensive methods
  - by Maximum likelihood [BickelLevina]
  - by Locally Weighted PCA [ChenLittleMaggioni]

How many dimensions are needed? A simple heuristic

Digit 3 in 2 dimensions

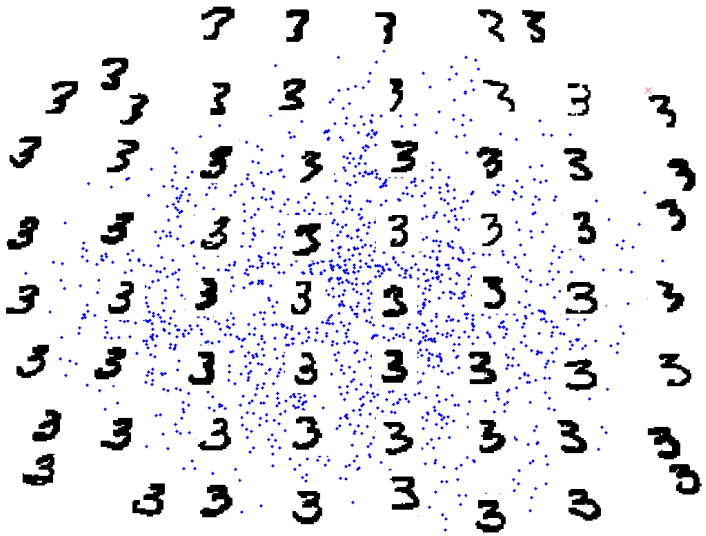


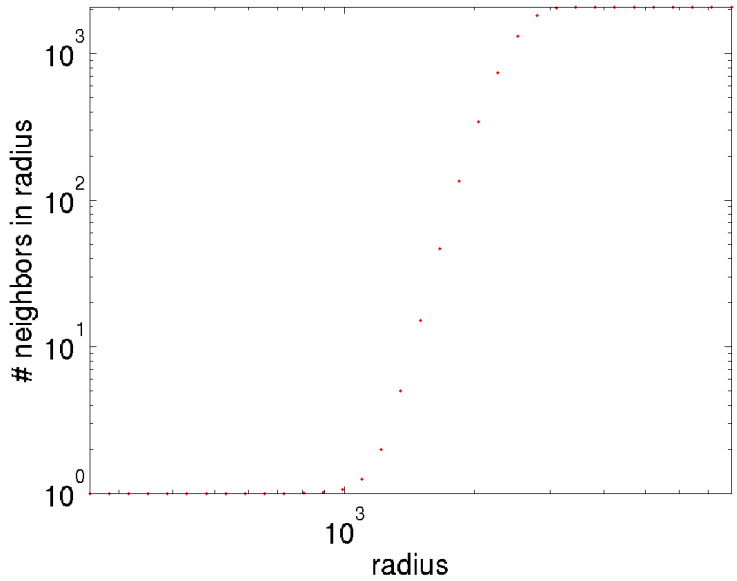
- How do we know dimennsion is 2?

also suffers from border effects

How many dimensions are needed? A simple heuristic

Digit 3 in 2 dimensions



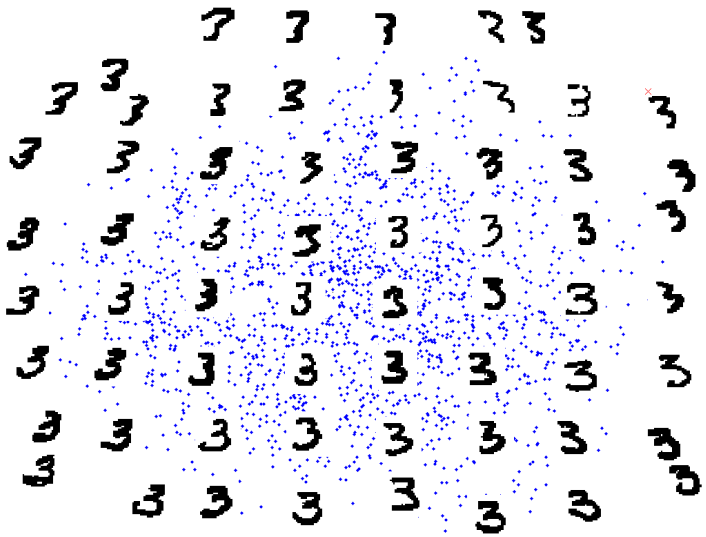


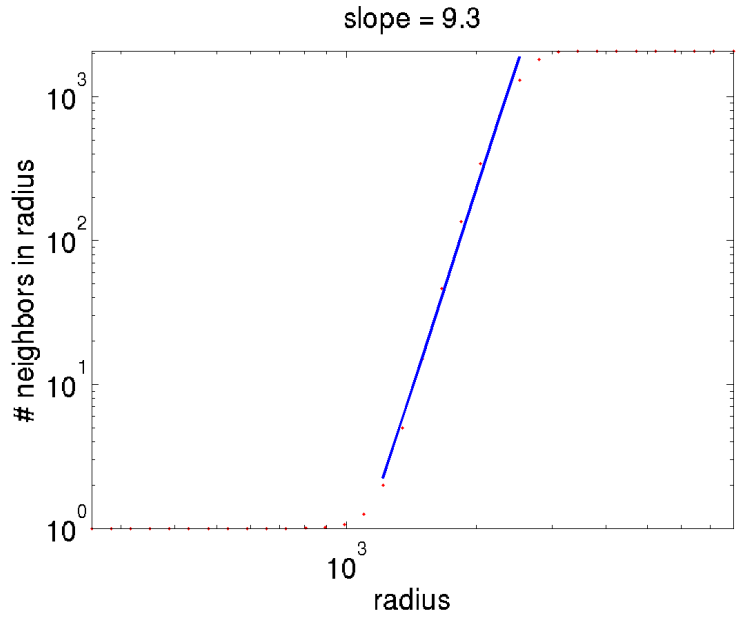
- How do we know dimennsion is 2?
- Volume of radius r ball in d dimenstions $\propto r^d$
- Hence, $\log(\#\text{ neighbors within } r) \propto d \log r$

also suffers from border effects

How many dimensions are needed? A simple heuristic

Digit 3 in 2 dimensions



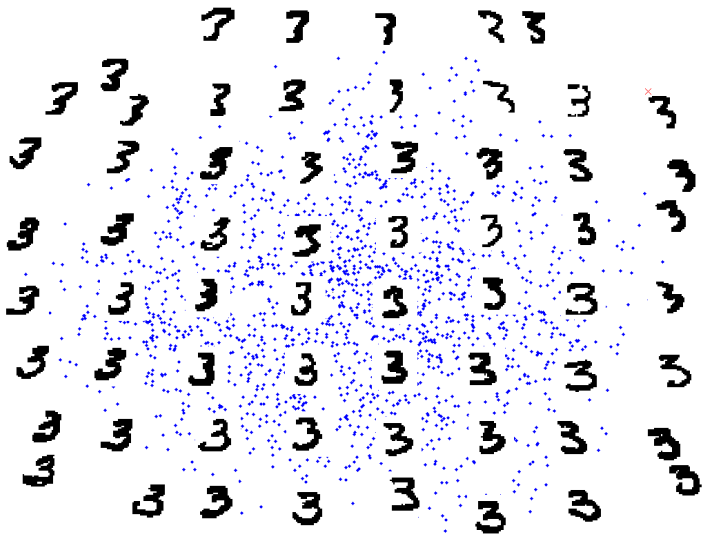


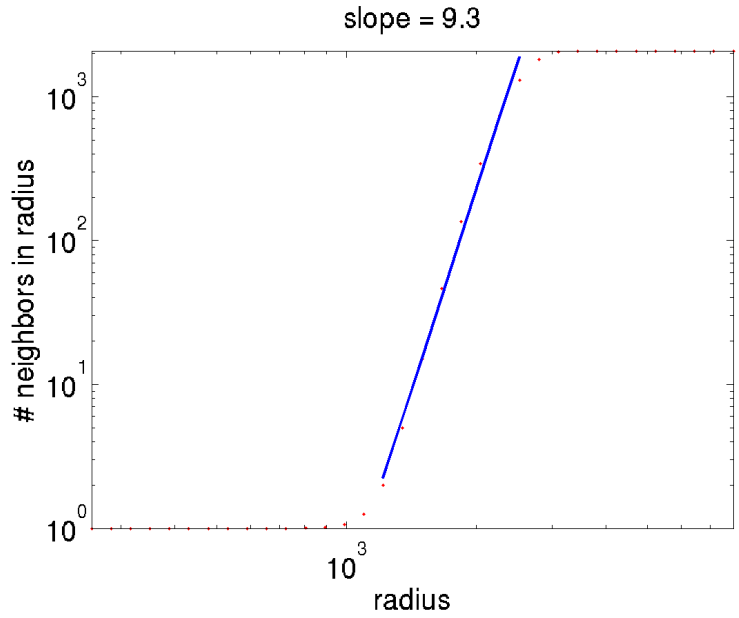
- How do we know dimennsion is 2?
- Volume of radius r ball in d dimenstions $\propto r^d$
- Hence, $\log(\#$ neighbors within $r) \propto d \log r$
- Slope of graph estimates dimension

also suffers from border effects

How many dimensions are needed? A simple heuristic

Digit 3 in 2 dimensions





- How do we know dimennsion is 2?
- Volume of radius r ball in d dimenstions $\propto r^d$
- Hence, $\log(\#$ neighbors within $r) \propto d \log r$
- But this method understimates

also suffers from border effects

Manifold learning and many particle systems

- Many particle systems are
  - high-dimensional
  - local interactions can be well described on physical grounds
  - large samples (often) available (e.g by simulation)
- Manifold learning/Non-linear dimension reduction
  - finds small number of state parameters – coarse graining
- Why now?
  - Scalable algorithms needed
  - Interpretability of coordinates needed (achieved partially by Metric ML)
  - Gaussian processes on manifolds allow for theoretically grounded prediction tools
  - Demands, grounding from practical problems will spur progress in ML itself