Learning the Density Structure of High-Dimensional Data

Yoshua Bengio

Work done with Martin Monperrus

Laboratoire d'Informatique
des Systèmes Adaptatifs
http://www.iro.umontreal.ca/~lisa

Université de Montréal

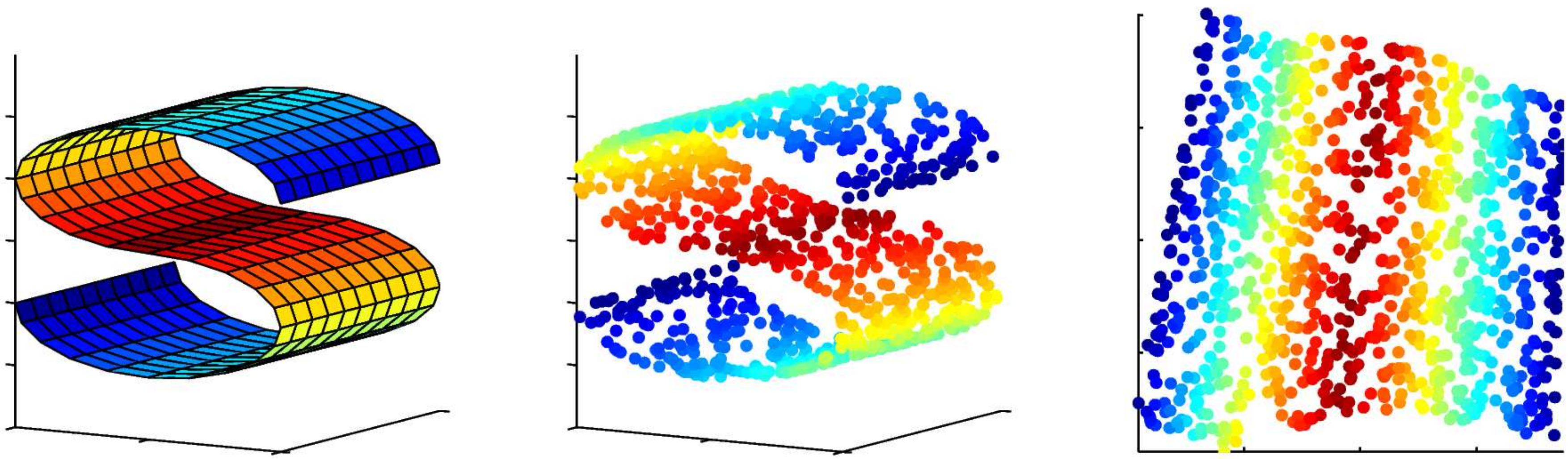
Real Goals of Statistical Learning

- Given a set D of l examples x_t coming from an unknown distribution or process.
- Discover structure in that distribution (= departures from uniformity and independence) so as to be able to make predictions about new combinations of values.
- Grossly: where are zones of high density vs low density?
- Generalization: inference must work on **new examples** from the same distribution.
- With **high-dimensional data**, new examples tend to be "far" for training data.

Spectral Embedding Algorithms

Algorithms for estimating a training set embedding on the presumed data manifold from the *principal eigenvectors of a Gram matrix* M with $M_{ij} = K_D(x_i, x_j)$ from data-dependent kernel K_D.

• Examples: **LLE** (Roweis & Saul 2000), **Isomap** (Tenenbaum et al 2000), **Laplacian Eigenmaps** (Belkin & Niyogi 2003), **spectral clustering** (Weiss 99), **kernel PCA** (Schölkopf et al 98). *Each corresponds to different* K_D. (fig. *Roweis & Saul*)



• Attractiveness: represent non-linear manifolds with analytic solution.

Out-of-Sample Embedding = Induction

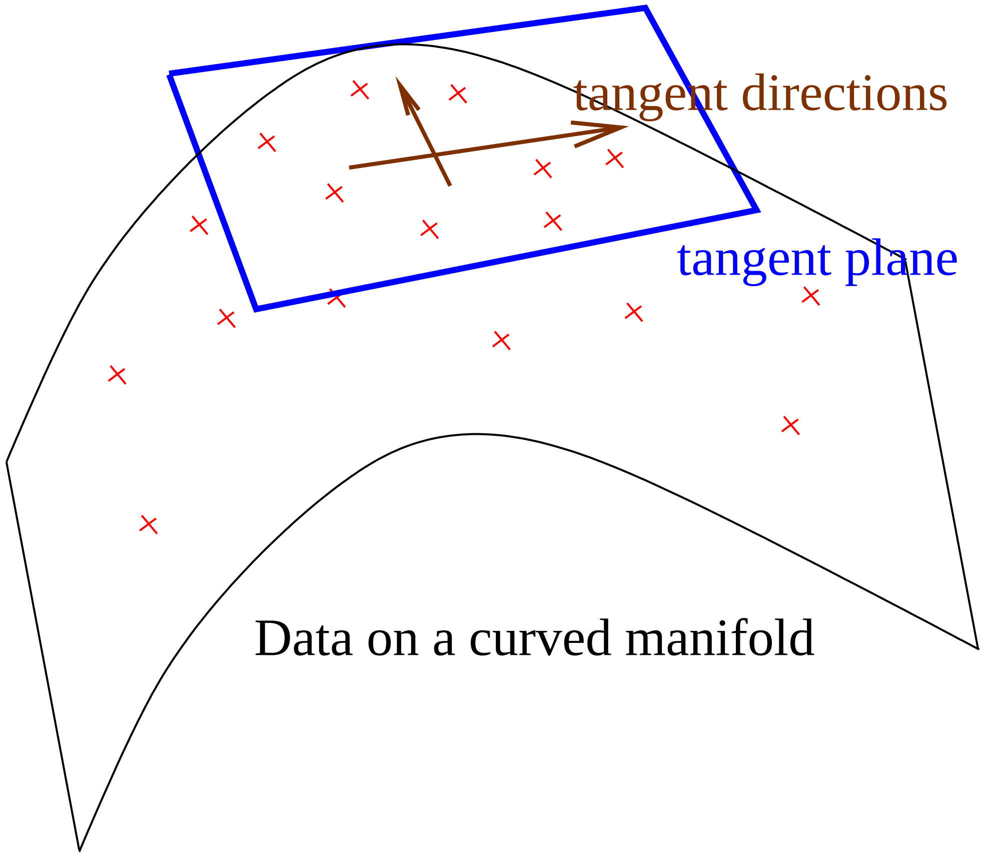
- How to generalize to new examples without recomputing eigenvectors?
- Are there corresponding **induction algorithms**?
- Out-of-sample generalization with the Nyström formula:

$$e_k(x) = \frac{1}{\lambda_k} \sum_{i=1}^{n} v_{ki} K_D(x, x_i) x$$

for k-th coordinate, with (λ_k, v_k) the k-th eigenpair of M.

- This is an estimator of the eigenfunctions of K_D as $|D| \to \infty$ (see upcoming Neural Comp. paper, on my web page).

Tangent Plane $\Longleftrightarrow$ Embedding Function



Important observation:

The tangent plane at x is simply the subspace spanned by the gradient vectors of the embedding function:

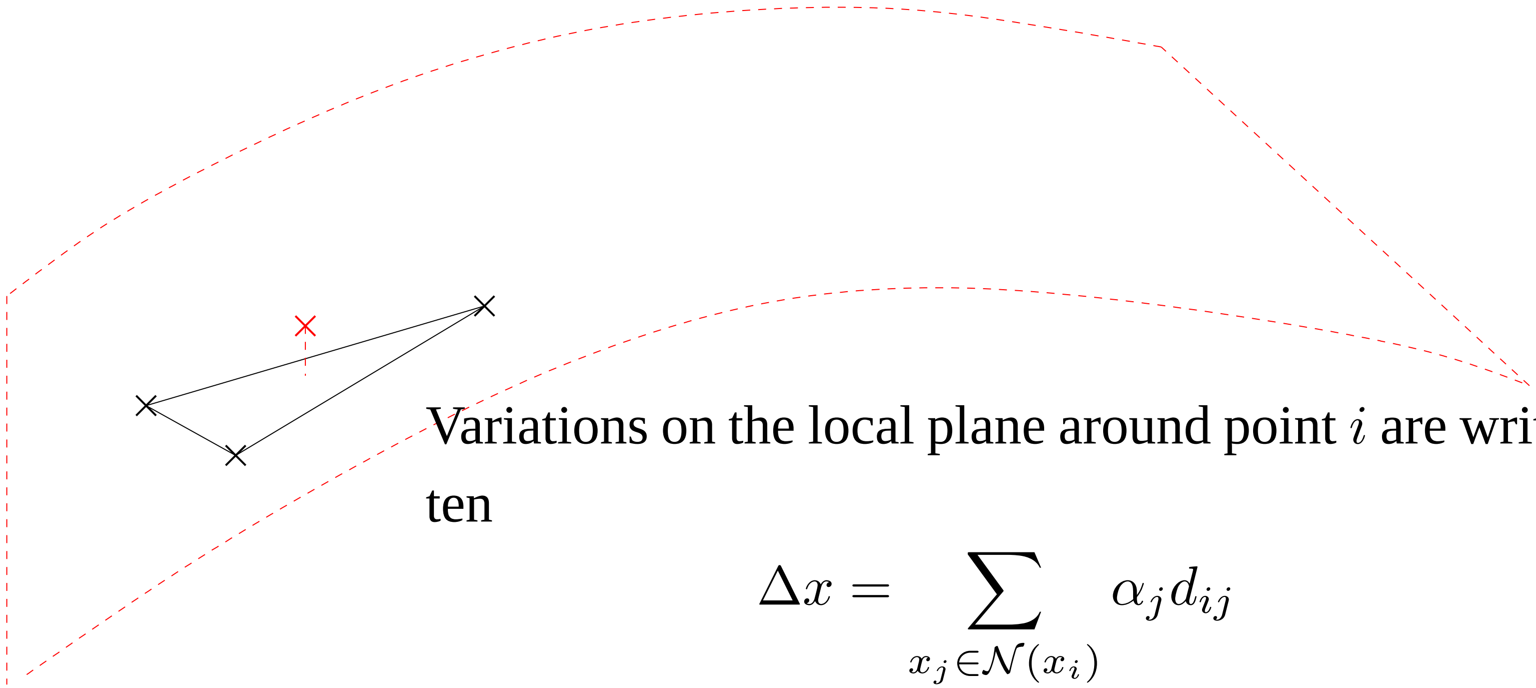
$$\frac{\partial e_k(x)}{\partial x}$$

Local Manifold Learning

- Local Manifold Learning Algorithms: derive information about the manifold structure near x using mostly the neighbors of x.
- For LLE, kernel PCA with Gaussian kernel, spectral clustering, Laplacian Eigenmaps $K_D(x, y) \rightarrow 0$ for x far from y, so $e_k(x)$ only depends on the neighbors of x.
- Therefore the tangent plane $\frac{\partial e_k(x)}{\partial x}$ also only depends on the neighbors of x.
- $\Rightarrow$ can't say anything about the manifold structure near a new example x that is "far" from training examples!

LLE: Local Affine Structure

The LLE algorithm estimates the local coordinates of each example in the basis of its nearest neighbors. Then looks for a low-dimensional coordinate system that has about the same expansion.

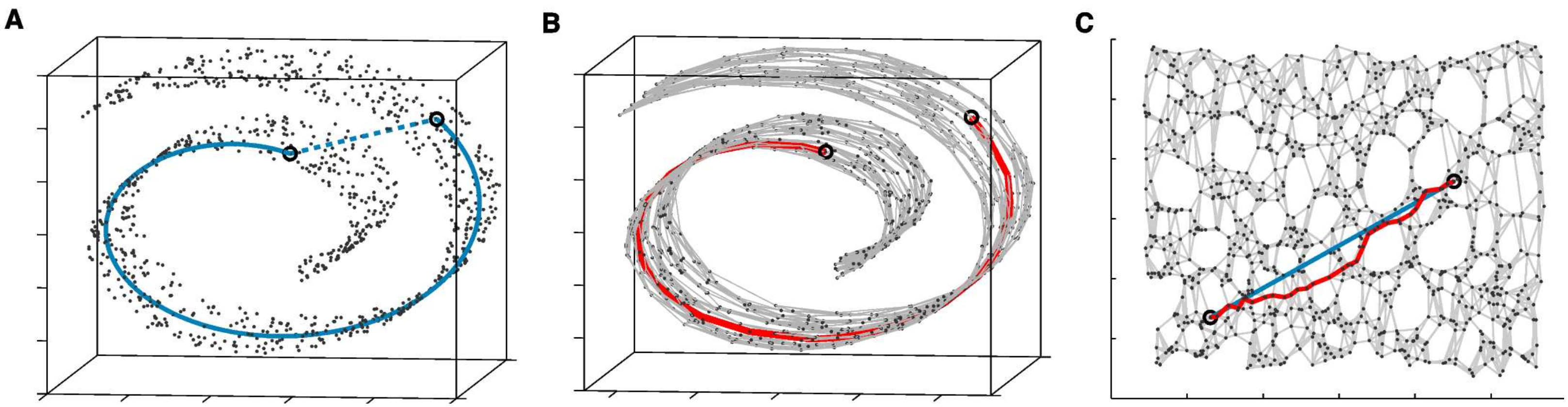


Variations on the local plane around point i are written

$$\Delta x = \sum_{x_j \in \mathcal{N}(x_i)} \alpha_j d_{ij}$$

where $d_{ij} = (x_i - x_j)$ are local "tangent directions" which are learned separately for each zone around a point x_i.

ISOMAP



Isomap estimates the **geodesic distance** along the manifold using the shortest path in the nearest neighbors graph.

It then looks for a low-dimensional representation that approximates those geodesic distances in the least square sense (MDS).

Lemma: the tangent plane at x of the manifold estimated by Isomap are included in the span of the vectors $x - x_j$ where x_j are training set neighbors of x (in the sense of being the first neighbor on the path from x to one of the training examples).

Isomap is also a local manifold learning algorithm!

Pancake Mixture Models

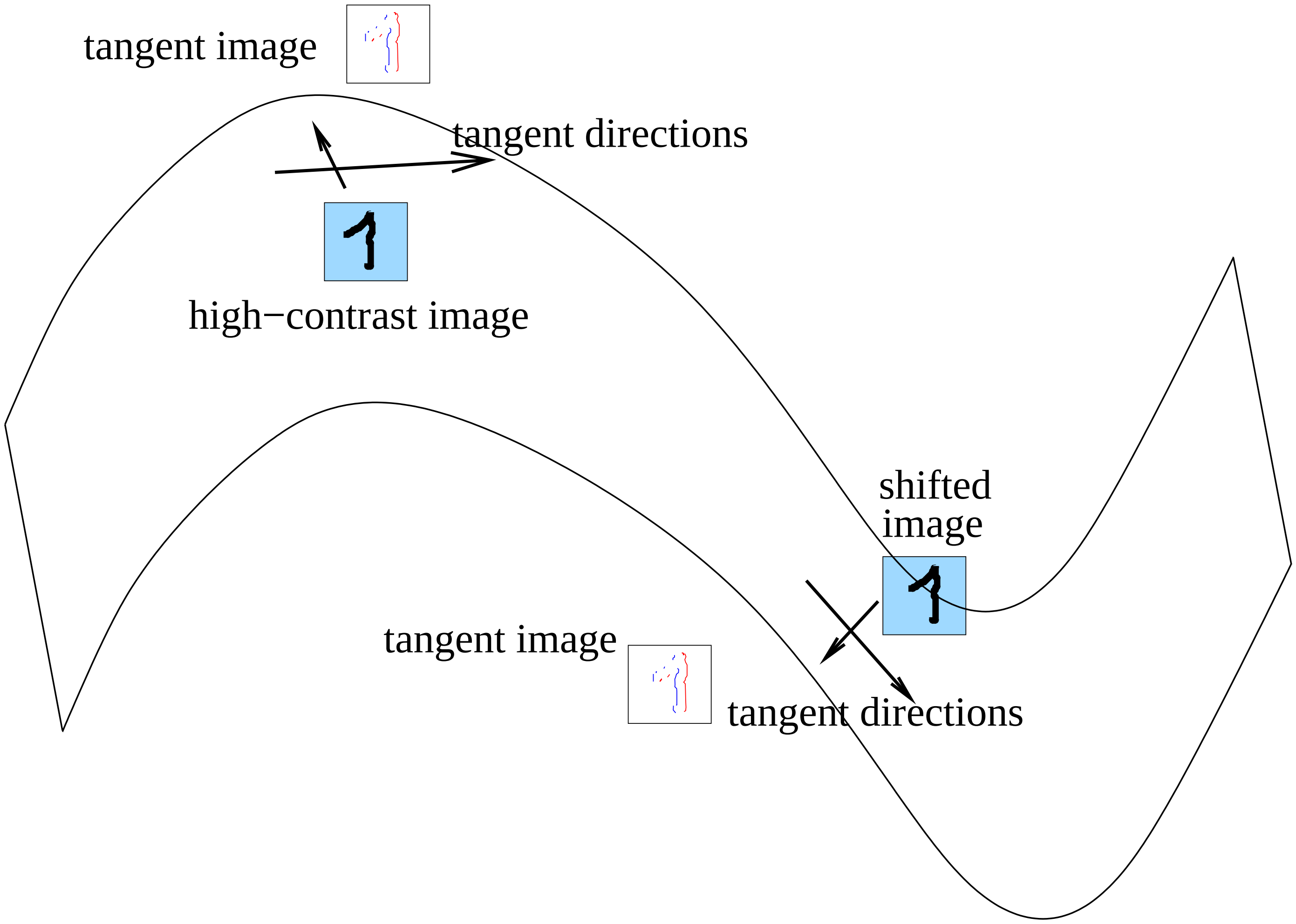
Other local manifold learning algorithms, density mixture models of flattened Gaussians:

- Mixtures of factor analyzers (Ghahramani & Hinton 96)
- Mixtures of probabilistic PCA (Tipping & Bishop 99)
- Manifold Parzen Windows (Vincent & Bengio 2003)
- Automatic Alignment of Local Representations (Teh & Roweis 2003)
- Manifold Charting (Brand 2003)

Some provide both density and embedding.

Local Manifold Learning: Local Linear Patches

Current manifold learning algorithms cannot handle highly curved manifolds because they are based on locally linear patches estimated locally.



Fundamental Problems with Local Manifold Learning

- **High Noise:** constraints not perfectly satisfied. Data not strictly on manifold. More noise $\longrightarrow$ more data needed per local patch.
- **High Curvature:** need more smaller patches O($(1/r)^d$) with $r =$ patch radius decreasing with curvature.
- **High Manifold Dimension:** $O((1/r)^d)$ patches are needed (curse of dimensionality), at least $O(d)$ examples per patch ($\propto$ noise).
- **Many manifolds:** e.g. images of transformed object instances = 1 manifold per instance or per object class. Local manifold learning can't take advantage of shared structure across multiple manifolds.

Non-Local Tangent Plane Predictors

Proposed approach: estimate tangent plane basis vectors as a function of position x in input space, with flexibly parametrized matrix-valued $d \times n$ function $F(x)$.

Train $F(x)$ to approximately span the differences between x and its neighbors.

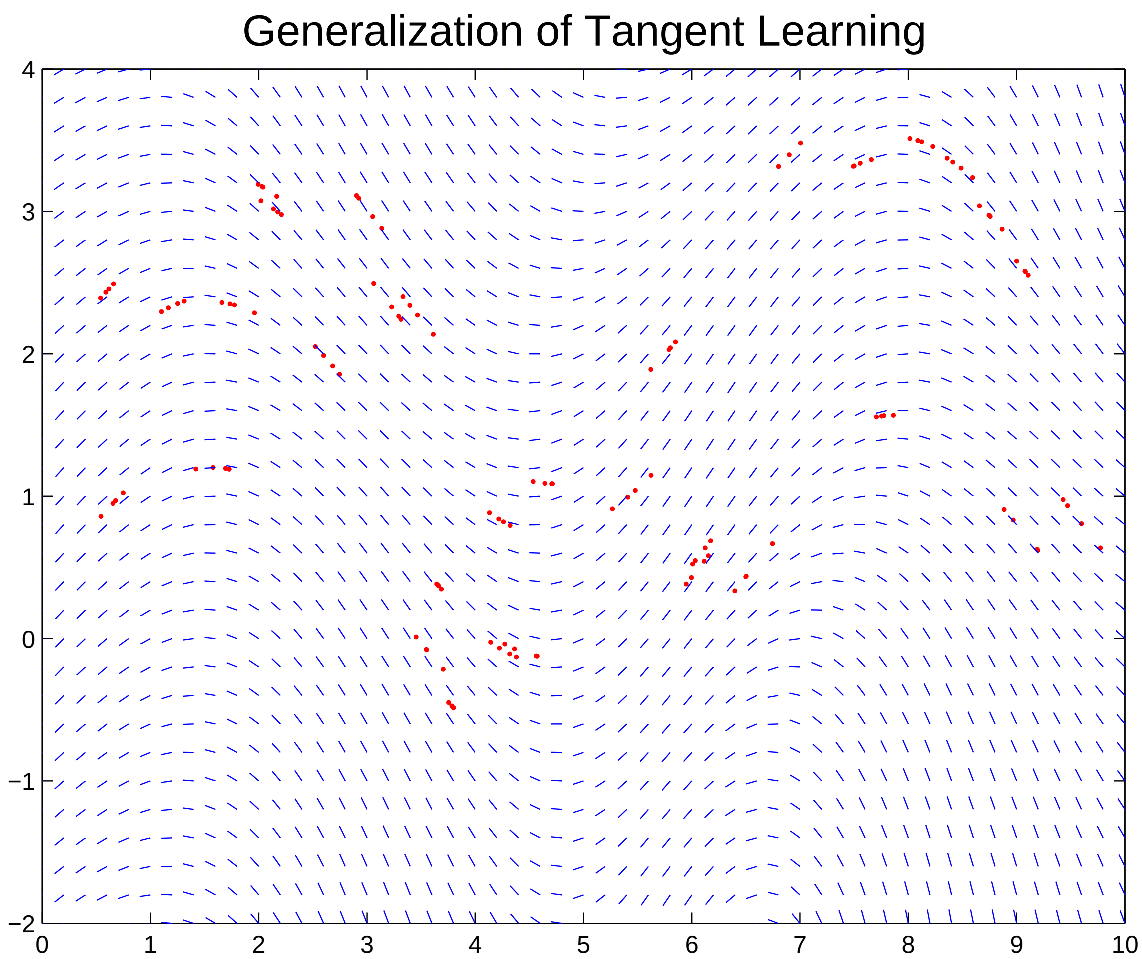
Experiments: estimate F with a simple neural network.

Training criterion = relative projection error at examples x_t and their neighbors x_i:

$$\min_{F,\{w_{tj}\}} \sum_t \sum_{j \in \mathcal{N}(x_t)} \frac{||F'(x_t)w_{tj} - (x_t - x_j)||^2}{||x_t - x_j||^2}$$

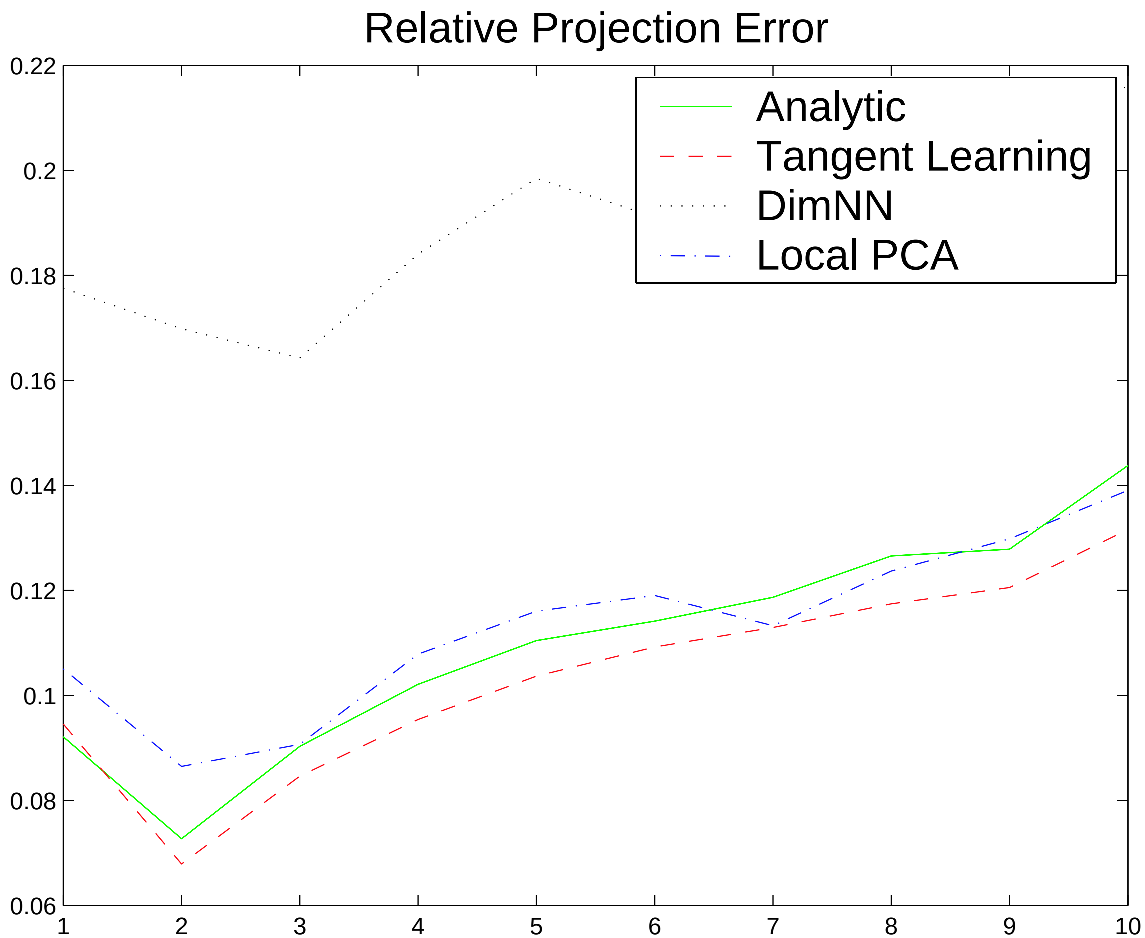
Double-optimization $\rightarrow$ Given F, analytic solution for each vector w_{tj}, can easily do stochastic gradient descent on F's parameters.

Results with Tangent Plane Predictors



Task 1: 2-D data with 1-D sinusoidal manifolds: the method indeed captures the tangent planes. Small blue segments are the estimated tangent planes. Red points are training examples.

Results with Tangent Plane Predictors



Task 2: 41-dimensional Gaussian curves $x(i) = e^{t_1 - (-2+i/10)^2/t_2}$ with two coordinates t_1 and t_2. Relative projection error for k-th nearest neighbor, w.r.t. k from 1 to 5, for the four compared methods.

Results with Tangent Plane Predictors

Task 3: 1000 digit images + image with rotation = 2 examples / manifold.

Images are 14×14 of 10 digits from MNIST database.

testing on MNIST digits	Average relative projection error
analytic tangent plane	0.27
tangent learning	0.43
Dim-NN or **Local PCA**	1.50

Tangent vector on test image of 8:

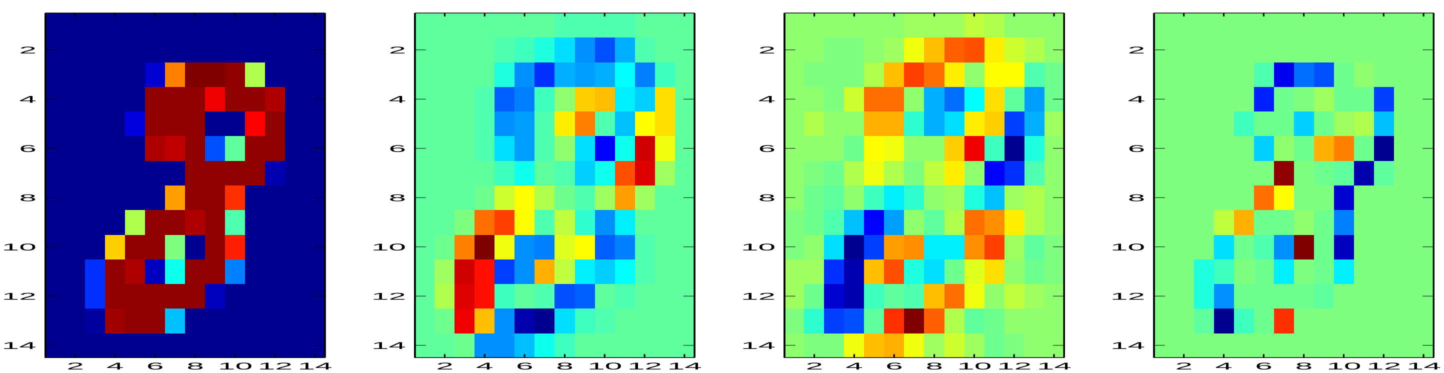
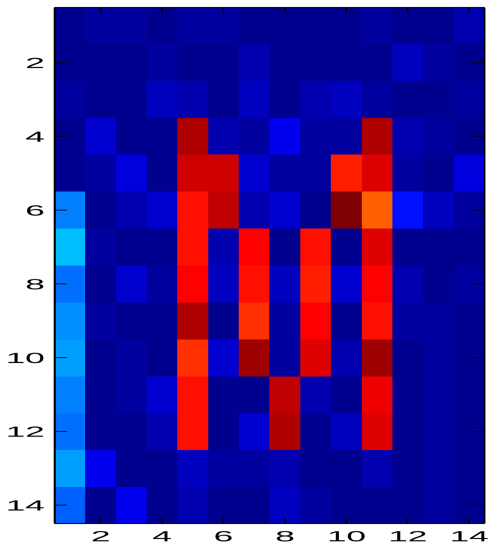


image analytic tan. learn. local PCA

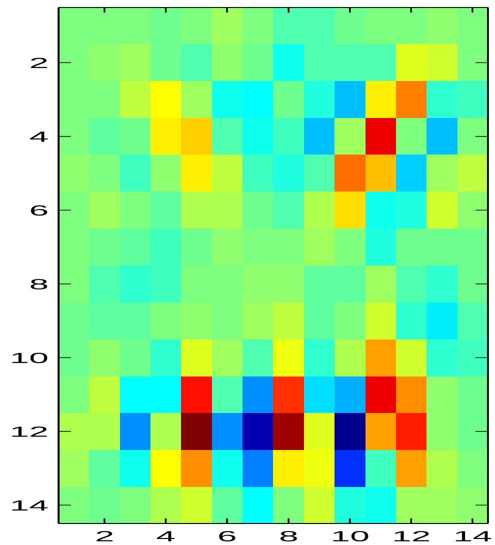
Truly Out-of-Sample Generalization

Model was trained on digits 0 to 9: test it on letter M

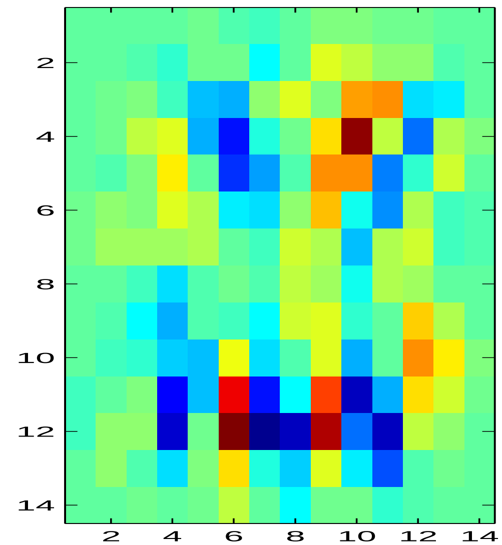
Compare predicted tangent vectors:



image



tan. learn.



local PCA

Not surprisingly, local manifold learning fails, whereas the globally estimated tangent plane predictor generalizes to very different image!

Conclusions

- Amazing progress in unsupervised learning the last few years: non-linear manifolds can be learned, with easy to optimize convex criteria.
- Can be extended to embedding function induction $\rightarrow$ generalization.
- Unfortunately they are estimating manifold tangents based on purely local information, which is very sensitive to four problems: **noise, curvature, dimensionality and multiple disjoint manifolds.**
- N.B. same problem with non-parametric semi-supervised learning!
- Proposed solution: learn a globally estimated tangent plane predictor function.
- Works superbly in all three experimental setups tested. **NOT CONVEX ANYMORE. BUT WORKS**.

Future Work

• *Proposed algorithm estimates principal directions of Gaussian covariance everywhere!*

• Using existing algorithms (Brand 2003;Teh & Roweis 2003), predicted Gaussian covariance at centers x_i can be converted into
(1) A Gaussian mixture density function (globally estimated!)
(2) A globally coherent embedding.

• **Exotic Extension**: uncountable Gaussian mixture. Follow random walk which moves x to $x + \Delta x$, with Δx sampled from $p(x + \Delta x|x)$ from local covariance at x. Density = normalized eigenfunction $p(x)$ solving

$$\int p(x)p(y|x)dx = p(y)$$

Can be estimated by solving finite linear system from data + random walk samples x_t, yielding a solution of the form $p(x) = \sum_i \alpha_t p(x|x_t)$.