Non-Local Manifold Parzen Windows

Yoshua Bengio
Hugo Larochelle

Summary

1. The *curse of dimensionality* in density estimation is due to the **locality** of the estimators: $f(x)$ mostly depends on the neighbors of x.

2. Previous work on density estimation taking advantage of manifold structure: Manifold Parzen Windows (Vincent & Bengio 2003). Each local Gaussian is flattened in the directions of the manifold. Still **local**.

3. Previous work on non-local manifold learning: Manifold Tangent Learning (Bengio and Monperrus, 2005). Learn to predict the tangent vectors of the manifold at x as a function of x and of globally estimated parameters θ. Yields to non-local generalization.

4. This work: combine the above two, i.e. predict the *covariance matrix* at i-th Gaussian x_i as a function of x_i and of globally estimated parameters θ.

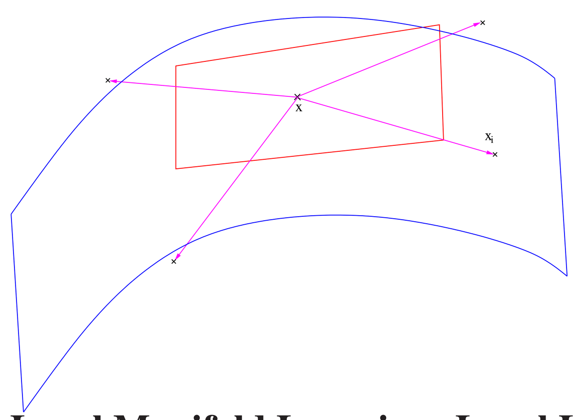
5. Results: *better density estimates* than using a local estimator (Parzen Windows or Manifold Parzen Windows or Gaussian Mixtures).

Curse of Dimensionality for Local Estimators

Curse of dimensionality for Parzen Windows

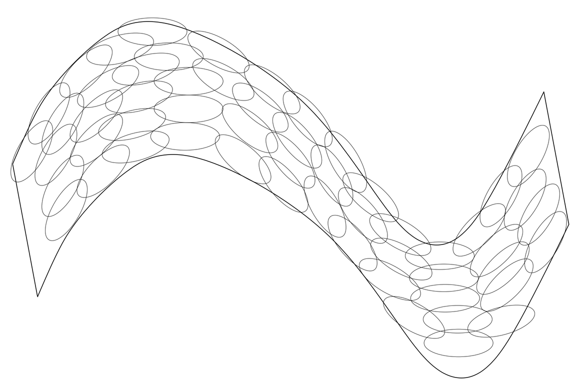
Number of required examples $\propto m^{(4+d)/5}$ where d is the intrinsic dimension of the data and m is the number of examples required to obtain given error level when $d=1$. See (Silverman, 1986; Hardle et al., 2004).

Tangent Plane Defined from Neighhbors



Spectral manifold learning algorithms (LLE, Isomap, etc...) define the local tangent plane at x mainly on the span of the vectors of neighbor differences $x_i - x$.

Local Manifold Learning: Local Linear Patches



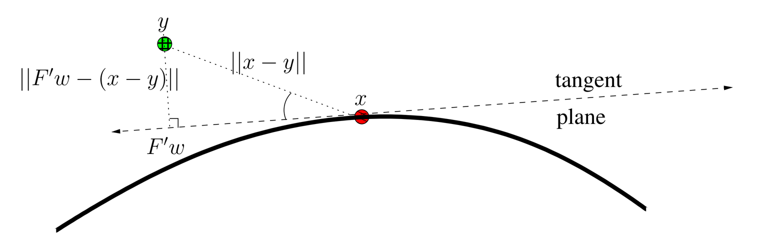
Current manifold learning algorithms cannot handle highly curved manifolds because they are based on locally linear patches estimated locally (possibly aligned globally).

Fundamental Problem with Local Manifold Learning: curse of dimensionality. Can't generalize "far" from training examples. $O((1/r)^d)$ patches needed, $> O(d)$ data/patch ($\propto$ noise).

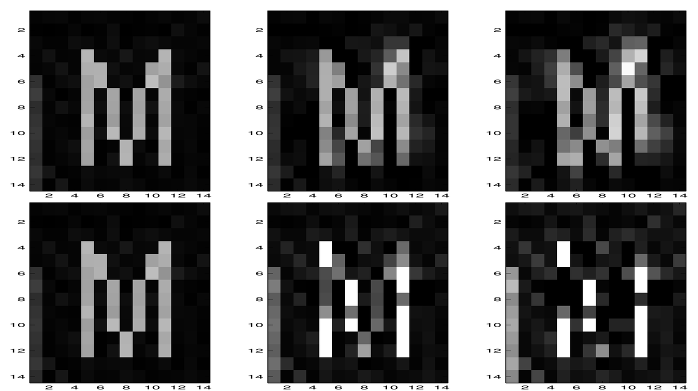
Non-Local Tangent Plane Predictors

Tangent plane basis vectors = learned function of x and global parameters θ, with flexibly parametrized matrix-valued $d \times n$ function $F(x)$.

Train $d \times n$ function $F(x)$ to approximately span the differences between x and its neighbors y.



Train on rotated DIGITS and generalize on LETTERS! Learn rotation manifold.



Top: using neural network DOES rotate. Bottom: using local predictor DOES NOT rotate.

Manifold Parzen Windows

Regularized Gaussian mixture, Gaussians centered near each training example x_i, covariance matrices flat in "principal directions":

$$p(y) = \frac{1}{n}\sum_{i=1}^{n} \text{Normal}(y; x_i + \mu(x_i), S(x_i))$$

where $x_i + \mu(x_i) =$ Gaussian center and $S(x_i) =$ covariance of i-th component, with

$$S(x_i) = \sigma^2_{noise}(x_i) I + \sum_{j=1}^{k} s_j(x_i) v_j(x_i) v_j(x_i)'$$

where $s_j(x_i)$ and $\sigma^2_{noise}(x_i)$ are scalars, and $v_j(x_i)$ denotes a "principal" direction with variance $s_j(x_i) + \sigma^2_{noise}(x_i)$, while $\sigma^2_{noise}(x_i)$ is the noise variance (the variance in all the other directions).

In (Vincent and Bengio, 2003), $\mu(x_i) = 0$, and $\sigma^2_{noise}(x_i) = \sigma^2$ is a global hyper-parameter, while $(\lambda_j(x_i), v_j(x_i)) = (s_j(x_i) + \sigma^2_{noise}(x_i), v_j(x_i))$ are leading (eigenvalue,eigenvector) of local empirical covariance.

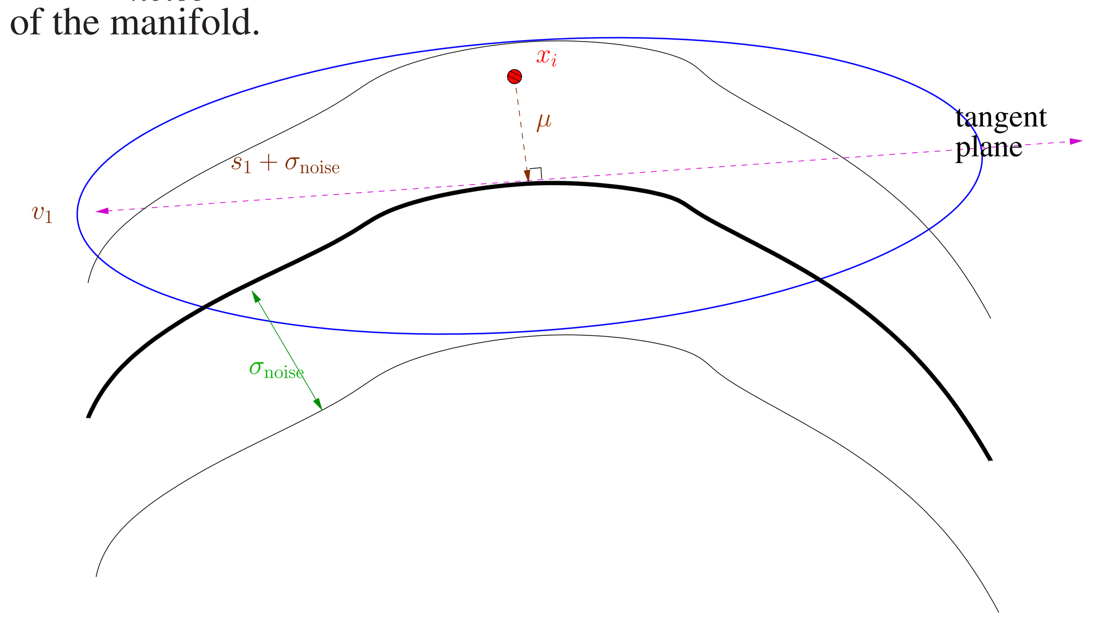
Hyper-parameters: k and σ^2_{noise}.

Non-Local Manifold Parzen Windows

Consider $\mu(x_i)$ and $S(x_i)$ as **functions** of x_i with *global parameters*: **share information about the density across different regions of space**.

Neural network with x_i in input to predict $\mu(x_i)$, $\sigma^2_{noise}(x_i)$, and the $s_j(x_i)$ and $v_j(x_i)$. The $v_j(x_i)$ do not need to be orthonormal.

- The Gaussian near x_i tells how neighbors of x_i are expected to differ from x_i.
- Its "principal" vectors $v_j(x_i)$ span the tangent of the manifold near x_i.
- $\mu(x_i)$ tells how x_i is located with respect to its projection on the manifold.
- $\sigma^2_{noise}(x_i)$ tells us how far from the manifold to expect neighbors.
- $s_j(x_i) + \sigma^2_{noise}(x_i)$ tell us how far to expect neighbors on the different local axes of the manifold.

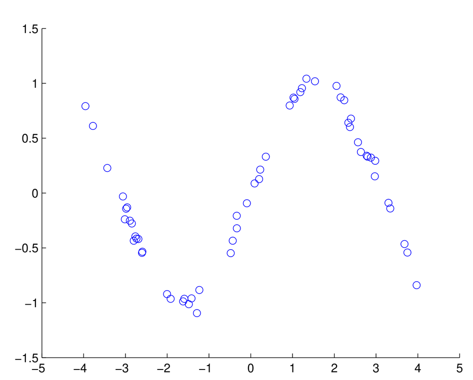


Semi-Spherical Variant

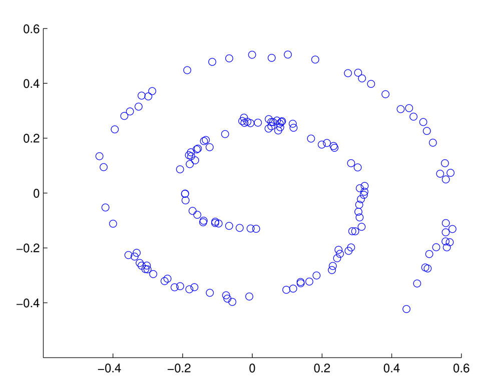
A simplified version of the above model has been implemented for the experiments described here. The simplification is that the *same variance is assumed for all the principal eigenvectors*, i.e. $s_j(x) = s_1(x)$.

In addition to the matrix $F(x_i)$ with rows $v_j(x_i)$, the outputs of the neural networks are the 'noise variance' $\sigma^2_{noise}(x_i)$, the excess 'on-manifold variance' $s_1(x_i)$, and the Gaussian mean displacement $\mu(x_i)$.

Toy Data



Samples used for training



Samples used for training

References

Bengio, Y. and Monperrus, M. (2005). Non-local manifold tangent learning. In Saul, L., Weiss, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems 17*. MIT Press.

Hardle, W., Muller, M., Sperlich, S., and Werwatz, A. (2004). *Nonparametric and Semiparametric Models*. Springer, http://www.xplore-stat.de/ebooks/ebooks.html.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, London.

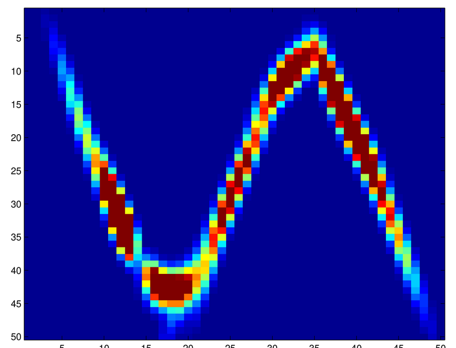
Vincent, P. and Bengio, Y. (2003). Manifold parzen windows. In Becker, S., Thrun, S., and Obermayer, K., editors, *Advances in Neural Information Processing Systems 15*, Cambridge, MA. MIT Press.

Experimental Results

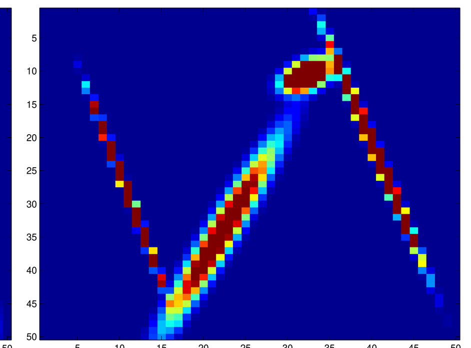
Hyper-parameters chosen by cross-validation
Out-of-sample Negative Log-Likelihood (NLL):

Algorithm	NLL on sinus	NLL on spiral
Non Local MP	1.409768	-1.35885
Gauss Mix Full	1.56653	-0.5747403
Gauss Mix Diag	2.593532	-0.6400698
Gauss Mix Spher	2.672263	-0.1716301
Manifold Parzen	1.697273	-0.6077337
Parzen Window	1.841497	-0.513347

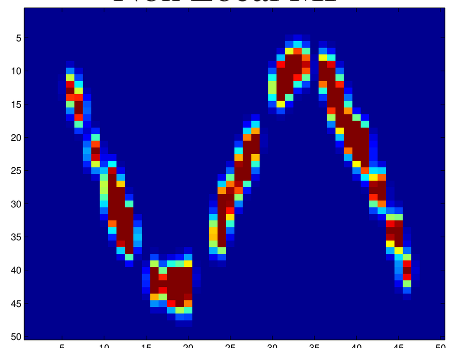
And the obtained distributions are:



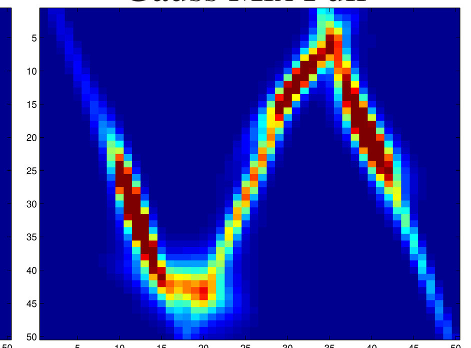
Non Local MP



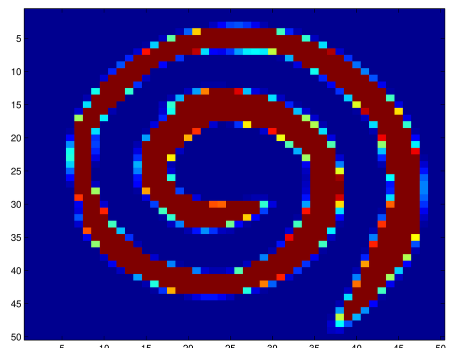
Gauss Mix Full



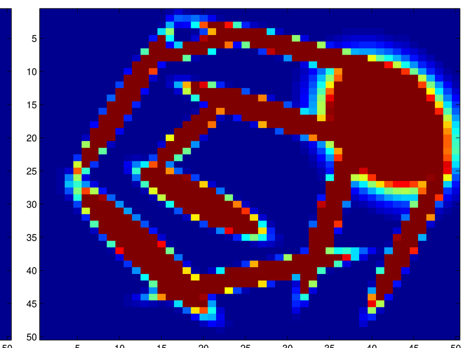
Parzen Window



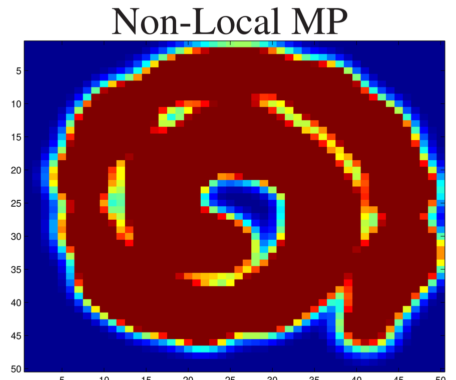
Manifold Parzen



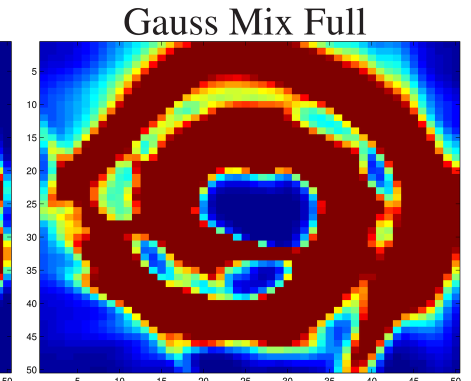
Non-Local MP



Gauss Mix Full

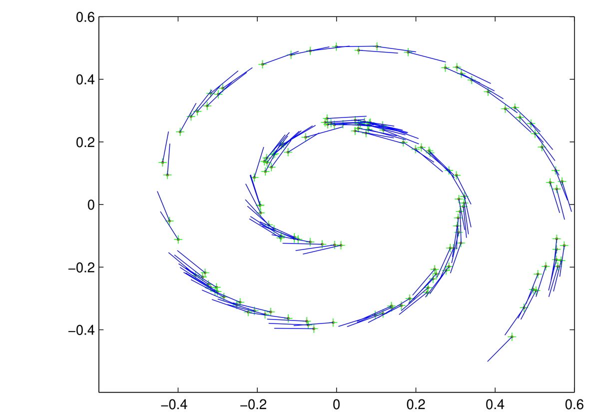


Parzen Window

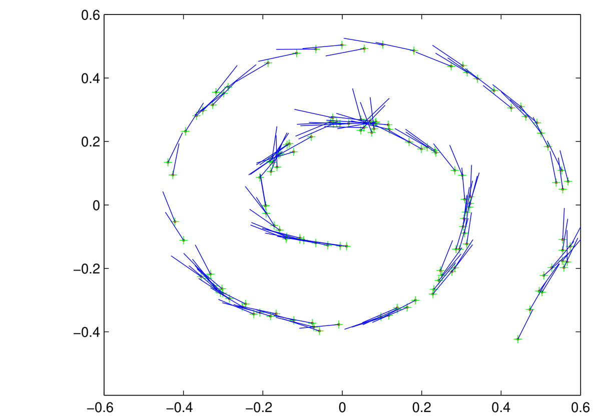


Manifold Parzen

The use of global parameters helps to find principal directions that are smoother and more consistent with the global structure of the data:



Principal directions learned with Non-Local Manifold Parzen



Principal directions learned with Manifold Parzen