Manifold Denoising As Preprocessing for Finding Natural Representations of Data

Matthias Hein **Markus Maier**
Max Planck Institute for Biological Cybernetics
Tübingen, Germany
{first.last}@tuebingen.mpg.de

Abstract

A natural representation of data is given by the parameters which generated the data. If the space of parameters is continuous, then we can regard it as a manifold. In practice, we usually do not know this manifold but we just have some representation of the data, often in a very high-dimensional feature space. Since the number of internal parameters does not change with the representation, the data will effectively lie on a low-dimensional submanifold in feature space. However, the data is usually corrupted by noise, which particularly in high-dimensional feature spaces makes it almost impossible to find the manifold structure. This paper reviews a method called *Manifold Denoising*, which projects the data onto the submanifold using a diffusion process on a graph generated by the data. We will demonstrate that the method is capable of dealing with non-trival high-dimensional noise. Moreover, we will show that using the denoising method as a preprocessing step, one can significantly improve the results of a semi-supervised learning algorithm.

Introduction

If one observes an image sequence of a moving head all the information of each image is contained in the parameters of the current head position. In this example, two parameters, the two angles which describe in which direction the head is pointing, are sufficient. Even though we have a high-dimensional representation of the data, the pixels of the image, the data can be effectively described by two parameters. Since the head moves continuously, the images corresponding to all possible values of the angles will form a two-dimensional submanifold in pixel space. It is obvious that using this *internal* parametrization should be helpful in learning problems. In recent years, several methods have been developed in the machine learning community which are based on the assumption that the data lies on a submanifold M in $\mathbb{R}^d$. They have been used in semi-supervised learning (Zhou *et al.* 2004), dimensionality reduction (Tenenbaum, de Silva, & Langford 2000; Belkin & Niyogi 2003) and clustering. However, there exists a certain gap between theory and practice. Namely, in practice the data lies almost never exactly on the submanifold but, due to noise, is scattered around it. Several of the existing algorithms, in particular graph based methods, are quite sensitive to noise. All of these methods basically use the same principle: The global structure of the data, the manifold the data lies on, is reconstructed by "gluing" together local structure in the data. In graph-based methods this principle is implemented by connecting only points which are close to each other. However, this concept breaks down in the presence of high-dimensional noise since the local neighborhoods are corrupted. The following Lemma illustrates this point.

Lemma 1 *Let $x, y \in \mathbb{R}^d$ and $\epsilon_1, \epsilon_2 \sim N(0, \sigma^2)$ and define $X = x + \epsilon_1$ and $Y = y + \epsilon_2$, then*

$$\mathbb{E}\,\|X - Y\|^2 = \|x - y\|^2 + 2\,d\,\sigma^2,$$
$$\mathrm{Var}\,\|X - Y\|^2 = 8\sigma^2\,\|x - y\|^2 + 8\,d\,\sigma^4.$$

We observe that the expected squared distance of two points x, y which are perturbed by high-dimensional Gaussian noise is dominated by the noise term if $d\,\sigma^2$ is sufficiently large. Thus, points which are initially close together can be far apart after the noise is added.

The process of denoising has the goal to project the data points, which are scattered around the manifold, back onto the submanifold. In the literature there already exist some methods which have related objectives, like principal curves (Hastie & Stuetzle 1989) and the generative topographic mapping (Bishop, Svensen, & Williams 1998). For both methods one has to know the intrinsic dimension of the submanifold as a parameter of the algorithm. However, in the presence of high-dimensional noise it is almost impossible to estimate the intrinsic dimension correctly. Moreover, usually problems arise if there is more than one submanifold. Our algorithm *Manifold Denoising* (Hein & Maier 2007) addresses these problems. It works well for low-dimensional submanifolds corrupted by high-dimensional noise and can deal with multiple submanifolds. The basic principle behind our denoising method has been introduced in the seminal work of (Taubin 1995) as a surface processing method in $\mathbb{R}^3$. We extend this method to general submanifolds in $\mathbb{R}^d$, where our emphasis lies on dealing with high-dimensional noise. Moreover, we provide a new interpretation of the denoising algorithm. This interpretation takes into account the probabilistic setting encountered in machine learning and differs from the one given in (Taubin 1995).

The noise model and problem statement

We assume that the data lies on an abstract m-dimensional manifold M, where the dimension m is the number of independent parameters in the data. This data is mapped via a smooth, regular embedding $i : M \to \mathbb{R}^d$ into the feature space $\mathbb{R}^d$. In the following we will not distinguish between M and $i(M) \subset \mathbb{R}^d$ since it should be clear from the context which case we are considering. The Euclidean distance in $\mathbb{R}^d$ then induces a metric on M. This metric depends on the embedding/representation (e.g. scaling) of the data in $\mathbb{R}^d$ but is at least continuous with respect to the intrinsic parameters. Furthermore, we assume that the manifold M is equipped with a probability measure P_M which is absolutely continuous with respect to the natural volume element[1] dV of M.

With these definitions the model of the noisy data-generating process in $\mathbb{R}^d$ has the following form:

$$X = i(\Theta) + \epsilon,$$

where $\Theta \sim P_M$ and $\epsilon \sim N(0, \sigma)$. Note that the probability measure of the noise ϵ has full support in $\mathbb{R}^d$. We consider here for convenience a Gaussian noise model but also any other reasonably concentrated isotropic noise should work. The density p_X of the noisy data X can be computed from the true data-generating probability measure P_M:

$$p_X(x) = (2\,\pi\,\sigma^2)^{-\frac{d}{2}} \int_M e^{-\frac{\|x-i(\theta)\|^2}{2\sigma^2}}\, p(\theta)\, dV(\theta). \qquad (1)$$

The Gaussian measure is equivalent to the heat kernel $p_t(x,y) = (4\pi t)^{-\frac{d}{2}} \exp\big(-\frac{\|x-y\|^2}{4t}\big)$ of the diffusion process on $\mathbb{R}^d$, see for example (Grigoryan 2006), if we make the identification $\sigma^2 = 2t$. Therefore an alternative point of view on p_X is to see p_X as the result of a diffusion of the density function $p(\theta)$ of P_M stopped at time $t = \frac{1}{2}\sigma^2$. The basic principle behind the denoising algorithm of this paper is to *reverse* this diffusion process.

The denoising algorithm

In practice we have only an i.i.d. sample $\{X_i\}_{i=1}^n$ of P_X. The ideal goal would be to find the corresponding set of points $\{i(\theta_i)\}_{i=1}^n$ on the submanifold M which generated the points X_i. However, due to the random nature of the noise this is in principle impossible. Instead the goal is to find points Z_i on the submanifold M which are close to the data points X_i. However, we are facing several problems. Since we are only given a finite sample, we do not know P_X or even P_M. Moreover, we have to discretize the reversal of the diffusion process which amounts to solving a PDE.

We do so by solving the diffusion process directly on a graph generated by the sample X_i. This can be motivated by recent results in (Hein, Audibert, & von Luxburg 2005) where it was shown that the generator of the diffusion process, the Laplacian $\Delta_{\mathbb{R}^d}$, can be approximated by the graph Laplacian of a random neighborhood graph.

[1] In local coordinates $\theta_1, \ldots, \theta_m$ the natural volume element dV is given as $dV = \sqrt{\det g}\, d\theta_1 \ldots d\theta_m$, where $\det g$ is the determinant of the metric tensor g.

The denoising algorithm As introduced in the previous section the denoising will be done using a backward diffusion process on the graph generated by the data points. We use a symmetric k-nearest neighbor graph (k-NN) with weights defined as

$$w(X_i, X_j) = \exp\Big(-\frac{\|X_i - X_j\|^2}{(\max\{h(X_i), h(X_j)\})^2}\Big),$$

if $\|X_i - X_j\| \leq \max\{h(X_i), h(X_j)\}$ and $w(X_i, X_j) = 0$ otherwise, where $h(X_i)$ is the distance of X_i to its k-nearest neighbor. Additionally, we set $w(X_i, X_i) = 0$, so that the graph has no loops. Furthermore, we denote by d the degree function $d(X_i) = \sum_{j=1}^n w(X_i, X_j)$ of the graph. Let D be the diagonal matrix with the degree function on the diagonal. Then the graph Laplacian in matrix form is given as $\Delta = \mathbf{1} - D^{-1}W$, see (Hein, Audibert, & von Luxburg 2005) for more details. Since the graph Laplacian is the generator of the diffusion process on the graph, we can formulate the algorithm by the following differential equation on the graph:

$$\partial_t X^\alpha = -\gamma\, \Delta X^\alpha, \quad \alpha = 1, \ldots, d \qquad (2)$$

where $\gamma > 0$ is the diffusion constant and X^α is the α-th component of a vector X. Since the points change with time, the whole graph is dynamic in our setting. In order to solve the differential equation (2), we choose an implicit Euler-scheme, that is

$$X^\alpha(t+1) - X^\alpha(t) = -\delta t\, \gamma\, \Delta X^\alpha(t+1), \qquad (3)$$

where δt is the time-step. Since the implicit Euler is unconditionally stable, we can choose the factor $\delta t\, \gamma$ arbitrarily. For the experiments we set $\gamma = 1$ and $\delta t = 0.5$. The solution of the implicit Euler scheme for one timestep in Equation 3 can then be computed as, $X^\alpha(t+1) = (\mathbf{1} + \delta t\, \Delta)^{-1} X^\alpha(t)$ for $\alpha = 1, \ldots, d$. After each timestep the point configuration has changed so that one has to recompute the weight matrix W of the graph. This procedure is continued until a predefined stopping criterion is satisfied, see (Hein & Maier 2007). The pseudo-code is given in Algorithm 1.

Algorithm 1 Manifold denoising

1: Choose δt, k
2: **while** Stopping criterion not satisfied **do**
3: Compute the k-NN distances $h(X_i)$, $i = 1, \ldots, n$,
4: Compute the weights $w(X_i, X_j)$ of the graph with $w(X_i, X_i) = 0$,
$w(X_i, X_j) = \exp\Big(-\frac{\|X_i - X_j\|^2}{(\max\{h(X_i), h(X_j)\})^2}\Big)$, if $\|X_i - X_j\| \leq \max\{h(X_i), h(X_j)\}$,
5: Compute the graph Laplacian Δ, $\Delta = \mathbf{1} - D^{-1}W$,
6: Solve $X^\alpha(t+1) - X^\alpha(t) = -\delta t\, \Delta X^\alpha(t+1) \quad \Rightarrow$ $X^\alpha(t+1) = (\mathbf{1} + \delta t\, \Delta)^{-1} X^\alpha(t)$ for $\alpha = 1, \ldots, d$.
7: **end while**

Large sample limit

Our qualitative theoretical analysis of the denoising algorithm is based on recent results on the limit of graph Laplacians (Hein, Audibert, & von Luxburg 2005) as the neighborhood size decreases and the sample size increases. We

use this result to study the continuous limit of the diffusion process. The following theorem about the limit of the graph Laplacian applies to h-neighborhood graphs (X_i and X_j are connected if $\|X_i - X_j\| \leq h$), whereas the denoising algorithm is based on a k-NN graph. Our conjecture is that the result carries over to k-NN graphs.

Theorem 1 *(Hein, Audibert, & von Luxburg 2005) Let $\{X_i\}_{i=1}^n$ be an i.i.d. sample of a probability measure P_M on a m-dimensional compact submanifold M of $\mathbb{R}^d$, where P_M has a density $p_M \in C^3(M)$. Let $f \in C^3(M)$ and $x \in M \backslash \partial M$, then if $h \to 0$ and $nh^{m+2}/\log n \to \infty$, we have almost surely*

$$\lim_{n\to\infty} \frac{1}{h^2}(\Delta f)(x) \sim -(\Delta_M f)(x) - \frac{2}{p}\langle \nabla f, \nabla p \rangle_{T_x M},$$

where Δ_M is the Laplace-Beltrami operator of M and $\sim$ means up to a constant.

We derive the continuum limit of our graph based diffusion process in the noise free case, for the noisy case see (Hein & Maier 2007). In that case $X = i(\Theta), \quad \Theta \sim P_M$. A change of the data points X is therefore equivalent to a change of the embedding i of the manifold M into $\mathbb{R}^d$. Thus, it is more appropriate to think of the diffusion process of the data points as a change of the embedding. In order to study the limit of the graph-based diffusion process, we make the usual argument for the transition from a difference equation on a grid to the corresponding differential equation. We rewrite the diffusion equation (3) on the graph with $\Delta \to \frac{1}{h^2}\Delta$ and $\gamma = \frac{h^2}{\delta t}$ as

$$\frac{i(t+1) - i(t)}{\delta t} = -\frac{h^2}{\delta t}\frac{1}{h^2}\Delta i$$

Performing the limit $h \to 0$ and $\delta t \to 0$ such that the diffusion constant $\gamma = \frac{h^2}{\delta t}$ stays finite and using Theorem 1 for the limit of $\frac{1}{h^2}\Delta$, we get the following differential equation,

$$\partial_t i = \gamma\,[\Delta_M i + \frac{2}{p}\langle \nabla p, \nabla i \rangle]. \tag{4}$$

Note that for the k-NN graph the neighborhood size h is a function of the density which implies that the diffusion constant γ also becomes a function of the density $\gamma = \gamma(p(x))$. It is well known that $\Delta_M i = m\,H$ where H is the mean curvature. Thus the differential equation (4) is equivalent to a generalized mean curvature flow.

$$\partial_t i = \gamma\,[m\,H + \frac{2}{p}\langle \nabla p, \nabla i \rangle], \tag{5}$$

The equivalence to the mean curvature flow $\partial_t i = m\,H$ is usually given in computer graphics as the reason for the denoising effect, see (Taubin 1995). However, as we have shown the diffusion already has an additional part if one has a non-uniform probability measure on M.

Experiments

In the experimental section we test the performance of the denoising algorithm on two datasets. Furthermore, we explore the possibility to use the denoising method as a preprocessing step for semi-supervised learning. Due to lack of space we have to omit other applications of denoising as preprocessing for clustering or dimensionality reduction.

Denoising The first experiment is done on a toy-dataset. The manifold M is given as $t \to [\sin(2\pi t),\ 2\pi t]$, t is sampled uniformly on $[0, 1]$. We embed M into $\mathbb{R}^{200}$ and put full isotropic Gaussian noise with $\sigma = 0.4$ on each datapoint resulting in the upper part of Figure . We verify the effect of the denoising algorithm by estimating continuously the dimension of the data for different scales using the correlation dimension estimator of (Grassberger & Procaccia 1983). Note that for a discrete set of points the estimate of the dimension depends on the scale at which one "examines" the data. The result of the denoising algorithm with $k = 25$ for the k-NN graph and 10 timesteps is given in the lower part of Figure . The dimension estimate and the histogram of distances show that the algorithm has reduced the noise significantly. One can also see two undesired effects. First as discussed in the last section the diffusion process has a component which moves the manifold in the direction of the mean curvature, which leads to a smoothing of the sinusoid. Second, at the boundary the sinusoid shrinks due to the missing counterparts in the local averaging done by the graph Laplacian, which results in an inward tangential component.

Next we apply the denoising method to the handwritten

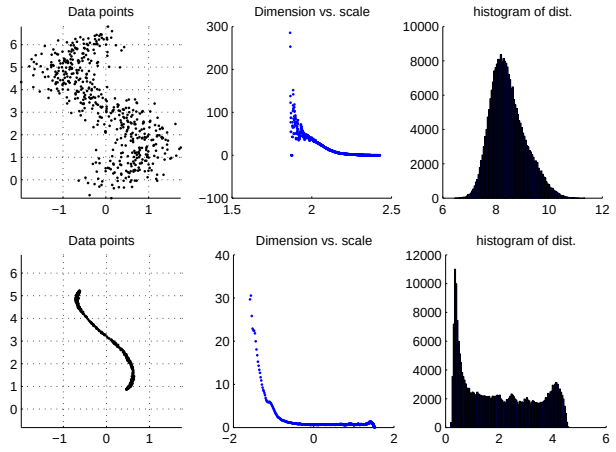

Figure 1: Top: 500 samples of the noisy sinusoid in $\mathbb{R}^{200}$ together with the dimension estimate plotted over a logarithmic scale and the histogram of distances, Bottom: Result after 10 steps of the denoising method with $k = 25$, note that the estimated dimension is much smaller and the scale has changed as can be seen from the histogram of distances.

digit dataset USPS. There the manifold corresponds to variations in writing styles. In order to check if the denoising method can also handle several manifolds at the same time, which would make the method useful for clustering and dimensionality reduction, we fed all the 10 digits into the algorithm. As distance measure we used the two-sided tangent distance in (Keysers *et al.* 2004) which provides a certain invariance against translation, scaling, rotation and line thickness. In Figure 2 a sample of the result across all digits is shown. Some digits are transformed wrongly. This happens since they are outliers with respect to their digit manifold

and lie closer to another digit component. An improved handling of invariances should resolve partially this problem.

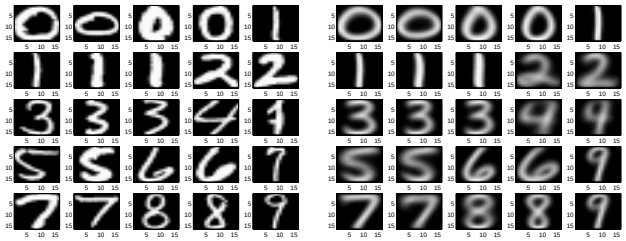

Figure 2: Left: Original images from USPS, right: after 15 iterations with $k = [9298/50]$.

Denoising as pre-processing for semi-supervised learning
In semi-supervised learning (SSL) only a small amount of the data is labeled and a huge amount is unlabeled. The goal in SSL is to use the unlabeled data as a kind of world knowledge in order to learn even quite complex decision boundaries with only a few labeled points. In this respect, it is the type of learning which is closest to human learning, where due to a huge domain knowledge a new concept can often be learned with little training examples.
Most semi-supervised learning (SSL) algorithms are based on the cluster assumption, that is the decision boundary should lie in a low-density region. The denoising algorithm is consistent with that assumption since it moves data points towards high-density regions. This is in particular helpful if the original clusters are distorted by high-dimensional noise. In this case the distance structure of the data becomes less discriminative, see Lemma 1, and the identification of the low density regions is quite difficult. Therefore manifold denoising as a pre-processing step should improve the performance of graph-based methods. However, the denoising algorithm does not take into account label information. In the case where the cluster assumption is not fulfilled the denoising algorithm might therefore decrease the performance. In order to avoid this, we add the number of iterations of the denoising process as an additional parameter in the SSL algorithm.
As SSL-algorithm we use a slight variation of the one by Zhou et al. (Zhou *et al.* 2004). It can be formulated as the following regularized least squares problem.

$$f^* = \operatorname{argmin}_{f \in \mathbb{R}^n} \sum_{i=1}^{n} d_i (y_i - f_i)^2 + \mu \sum_{i,j=1}^{n} w_{ij} (f_i - f_j)^2,$$

where y is the label vector with $y_i \in \{-1, +1\}$ for the labeled data and $y_i = 0$ for the unlabeled data. The solution is given as $f^* = (\mathbf{1} + \mu\Delta)^{-1} y$ with $\Delta = \mathbf{1} - D^{-1}W$. For the SSL-algorithm we used a symmetric k-NN graph with the weights: $w(X_i, X_j) = \exp(-\gamma \|X_i - X_j\|^2)$ if $\|X_i - X_j\| \leq \min\{h(X_i), h(X_j)\}$. The best parameters for the number of iterations as well as k and μ were found by cross-validation, see (Hein & Maier 2007) for the set of parameters. The Figure 3 shows the test error for the handwritten digit dataset MNIST. SSL with denoising outperforms SSL without denoising significantly. For 50 labeled points (on average only 5 training examples per class) the digit classification can be done already very accurately if denoising is used as a preprocessing step.

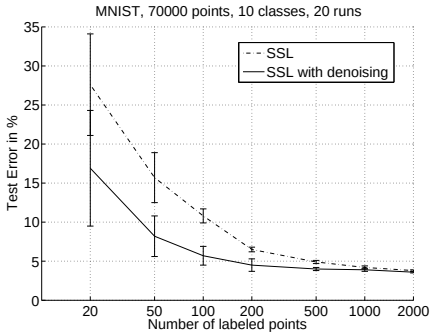

Figure 3: Test error on the MNIST dataset for varying number of labeled points. The test error is averaged over 20 random choices of the labeled points. SSL with denoising (solid line) is significantly better than SSL (dashed line).

References

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

Bishop, C. M.; Svensen, M.; and Williams, C. K. I. 1998. GTM: The generative topographic mapping. *Neural Computation* 10:215–234.

Grassberger, P., and Procaccia, I. 1983. Measuring the strangeness of strange attractors. *Physica D* 9:189–208.

Grigoryan, A. 2006. Heat kernels on weighted manifolds and applications. *Cont. Math.* 398:93–191.

Hastie, T., and Stuetzle, W. 1989. Principal curves. *J. Amer. Stat. Assoc.* 84:502–516.

Hein, M.; Audibert, J.-Y.; and von Luxburg, U. 2005. From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians. In *Proc. of the 18th Conf. on Learning Theory (COLT)*, 486–500.

Hein, M., and Maier, M. 2007. Manifold denoising. In *Adv. in Neural Inf. Proc. Syst. 19 (NIPS)*.

Keysers, D.; Macherey, W.; Ney, H.; and Dahmen, J. 2004. Adaptation in statistical pattern recognition using tangent vectors. *IEEE Trans. on Pattern Anal. and Machine Intel.* 26:269–274.

Taubin, G. 1995. A signal processing approach to fair surface design. In *Proc. of the 22nd annual conf. on Computer graphics and interactive techniques (Siggraph)*, 351–358.

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

Zhou, D.; Bousquet, O.; Lal, T. N.; Weston, J.; and Schölkopf, B. 2004. Learning with local and global consistency. In Thrun, S.; Saul, L.; and Schölkopf, B., eds., *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, volume 16, 321–328.