Manifold Blurring Mean Shift Algorithms for Manifold Denoising

Weiran Wang Miguel Á. Carreira-Perpiñán
Electrical Engineering and Computer Science, University of California, Merced
http://eecs.ucmerced.edu

Abstract

We propose a new family of algorithms for denoising data assumed to lie on a low-dimensional manifold. The algorithms are based on the blurring mean-shift update, which moves each data point towards its neighbors, but constrain the motion to be orthogonal to the manifold. The resulting algorithms are nonparametric, simple to implement and very effective at removing noise while preserving the curvature of the manifold and limiting shrinkage. They deal well with extreme outliers and with variations of density along the manifold. We apply them as preprocessing for dimensionality reduction; and for nearest-neighbor classification of MNIST digits, with consistent improvements up to 36% over the original data.

1. Introduction

Machine learning algorithms often take as starting point a high-dimensional dataset of N points $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_N) \subset \mathbb{R}^{N \times D}$, and then learn a model that is useful to infer information from this data, or from unseen data. Most algorithms, however, are more or less sensitive to the amount of noise and outliers in the data. For example, spectral dimensionality reduction methods such as Isomap [20] first estimate a neighborhood graph on the dataset $\mathbf{X}$ and then set up an eigenvalue problem to determine low-dimensional coordinates for each data point. Both steps are sensitive to noise and outliers, in particular building the neighborhood graph: it may be hard to find a good value (if it exists at all) for the number of neighbors k or the ball radius ϵ that will avoid disconnections or shortcuts. Other dimensionality reduction algorithms, such as latent variable models (e.g. mixtures of probabilistic PCAs [21]), try to learn a parametric model of the manifold and noise by maximum likelihood. However, these models are prone to bad local optima partly caused by noise and outliers. Although there are different ways of reducing the effects of noise and outliers (such as learning a graph in a more robust way [3] or using robust error functions), in this paper we concern ourselves with a different approach: to denoise the dataset $\mathbf{X}$ as a preprocessing step.

Data preprocessing is commonplace in machine learning. Consider, for example, the many simple but useful operations of subtracting the mean (possibly as a running average), low-pass filtering, standardizing the covariance, or removing outliers by trimming. Other operations are specific to certain types of data: deskewing or blurring for images, energy removal or cepstral normalization for speech. These operations help to achieve some invariance to unwanted transformations or to reduce noise and improve robustness. Here, we are interested in more sophisticated denoising techniques that adapt to the local manifold structure of high-dimensional data. We will assume that the dataset $\mathbf{X}$ comes from a manifold of dimension $L < D$ to which noise has been added. We will not make any assumptions about the nature of this noise—the form of its distribution (e.g. whether long-tailed), or whether it varies along the manifold. Denoising a manifold is also useful by itself, for example 3D mesh smoothing in computer graphics [19] or skeletonization of shapes such as digits. However, we will focus on denoising as a preprocessing step for supervised or unsupervised learning.

A good denoising algorithm should make as few assumptions about the data as possible, so nonparametric methods are preferable; and produce the same result for a given dataset, i.e., be deterministic. At the same time, it should have a small number of user parameters to control the algorithm's behavior (e.g. the amount of denoising). We propose an algorithm that fulfills these desiderata. It is based on two powerful ideas: the noise removal ability of locally averaging with a kernel of scale σ (implemented with the mean-shift algorithm); and the linear approximation of local manifold structure of dimension L (implemented with local PCA on the k nearest neighbors). We describe our algorithm in section 2, demonstrate it with unsupervised and supervised learning (dimensionality reduction and classification, resp.) in section 3, and discuss it in the context of related work in section 4.

2. The Manifold Blurring Mean Shift (MBMS) Algorithm

Our algorithm is based on the following ideas:

- *Local clustering with Gaussian blurring mean shift (GBMS)* (fig. 1): the blurring mean-shift update [10] with unit step size moves datapoints to the kernel average of their neighbors:

$$\mathbf{x}_n \leftarrow \sum_{m \in \mathcal{N}_n} \frac{G_\sigma(\mathbf{x}_n, \mathbf{x}_m)}{\sum_{m' \in \mathcal{N}_n} G_\sigma(\mathbf{x}_n, \mathbf{x}_{m'})} \mathbf{x}_m. \quad (1)$$

  The average is over $\mathcal{N}_n = \{1, \dots, N\}$ (full graph) or the k nearest neighbors of $\mathbf{x}_n$ (k-nn graph), and $G_\sigma(\mathbf{x}_n, \mathbf{x}_m) \propto \exp\left(-\frac{1}{2}(\|\mathbf{x}_n - \mathbf{x}_m\| / \sigma)^2\right)$. A single mean-shift step locally climbs up the kernel density estimate defined by the data points, and after one step all points are updated so the dataset shrinks over iterations. The process eventually converges to a state where all points coincide [4] but it can be reliably stopped to produce good clusterings that depend on σ [1, 2]. Its convergence is very fast, cubic with Gaussian clusters [1, 2].

- *Local tangent space estimation with PCA*: local PCA gives the best linear L-dimensional manifold in terms of reconstruction error (i.e., orthogonal projection on the manifold):

$$\min_{\boldsymbol{\mu}, \mathbf{U}} \sum_{m \in \mathcal{N}'_n} \left\| \mathbf{x}_m - (\mathbf{U}\mathbf{U}^T(\mathbf{x}_m - \boldsymbol{\mu}) + \boldsymbol{\mu}) \right\|^2 \quad (2)$$

  s.t. $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ with $\mathbf{U}_{D \times L},\ \boldsymbol{\mu}_{D \times 1}$, whose solution is $\boldsymbol{\mu} = \mathrm{E}_{\mathcal{N}'_n}\{\mathbf{x}\}$ and $\mathbf{U} =$ the leading L eigenvectors of $\mathrm{cov}_{\mathcal{N}_n}\{\mathbf{x}\}$. In general, $\mathcal{N}_n$ need not equal $\mathcal{N}'_n$.

Although GBMS by itself has strong denoising power (controlled by σ and the number of iterations), this denoising is directed not only orthogonally to the manifold but also tangentially. This causes motion along the manifold, which changes important properties of the data that are not noise (for example, for a handwritten digit, it may change its style). It also causes strong shrinkage, first at the manifold boundaries but also within the manifold (see the example of fig. 2). Thus, while very useful for clustering, its applicability to manifold denoising is limited.

Our **Manifold Blurring Mean Shift (MBMS) algorithm** combines these two steps. At each iteration and for every data point $\mathbf{x}_n$, a *predictor averaging step* is computed using one step of GBMS with width σ. We can use the full graph ($\mathcal{N}_n = \{1, \dots, N\}$) or the k-nn graph ($\mathcal{N}_n = k$ nearest neighbors of $\mathbf{x}_n$). This is responsible for local denoising. Then, a *corrector projective step* is computed using the local PCA of dimensionality L on the k nearest neighbors of $\mathbf{x}_n$. This is responsible for local manifold structure, and removes the tangential component of the motion. The two steps are iterated until sufficient denoising is achieved while avoiding shrinkage and distortions of the manifold (see later). The complete algorithm is in fig. 1. We will refer to the algorithm as MBMSf if using the full graph for the GBMS step, MBMSk if using the k-nn graph (same k for the GBMS and PCA steps), or simply as MBMS when the difference is not relevant.

Besides GBMS (MBMS for $L = 0$), another particular case of MBMS is of special interest, which we call

MBMS (L, k, σ) with full or k-nn graph: given $\mathbf{X}_{D \times N}$
repeat
  for $n = 1, \dots, N$
    $\mathcal{N}_n \leftarrow \{1, \dots, N\}$ (full graph) or
      k nearest neighbors of $\mathbf{x}_n$ (k-nn graph)
    $\partial \mathbf{x}_n \leftarrow -\mathbf{x}_n + \sum_{m \in \mathcal{N}_n} \frac{G_\sigma(\mathbf{x}_n, \mathbf{x}_m)}{\sum_{m' \in \mathcal{N}_n} G_\sigma(\mathbf{x}_n, \mathbf{x}_{m'})} \mathbf{x}_m$ mean-shift step
    $\mathcal{X}_n \leftarrow k$ nearest neighbors of $\mathbf{x}_n$
    $(\boldsymbol{\mu}_n, \mathbf{U}_n) \leftarrow \mathrm{PCA}_L(\mathcal{X}_n)$ estimate L-dim tangent space at $\mathbf{x}_n$
    $\partial \mathbf{x}_n \leftarrow (\mathbf{I} - \mathbf{U}_n \mathbf{U}_n^T) \partial \mathbf{x}_n$ subtract parallel motion
  end
  $\mathbf{X} \leftarrow \mathbf{X} + \partial \mathbf{X}$ move points
until stop
return $\mathbf{X}$

LTP (L, k) with k-nn graph: given $\mathbf{X}_{D \times N}$
repeat
  for $n = 1, \dots, N$
    $\mathcal{X}_n \leftarrow k$ nearest neighbors of $\mathbf{x}_n$
    $(\boldsymbol{\mu}_n, \mathbf{U}_n) \leftarrow \mathrm{PCA}_L(\mathcal{X}_n)$ estimate L-dim tangent space at $\mathbf{x}_n$
    $\partial \mathbf{x}_n \leftarrow (\mathbf{I} - \mathbf{U}_n \mathbf{U}_n^T)(\boldsymbol{\mu}_n - \mathbf{x}_n)$ project point onto tangent space
  end
  $\mathbf{X} \leftarrow \mathbf{X} + \partial \mathbf{X}$ move points
until stop
return $\mathbf{X}$

GBMS (k, σ) with full or k-nn graph: given $\mathbf{X}_{D \times N}$
repeat
  for $n = 1, \dots, N$
    $\mathcal{N}_n \leftarrow \{1, \dots, N\}$ (full graph) or
      k nearest neighbors of $\mathbf{x}_n$ (k-nn graph)
    $\partial \mathbf{x}_n \leftarrow -\mathbf{x}_n + \sum_{m \in \mathcal{N}_n} \frac{G_\sigma(\mathbf{x}_n, \mathbf{x}_m)}{\sum_{m' \in \mathcal{N}_n} G_\sigma(\mathbf{x}_n, \mathbf{x}_{m'})} \mathbf{x}_m$ mean-shift step
  end
  $\mathbf{X} \leftarrow \mathbf{X} + \partial \mathbf{X}$ move points
until stop
return $\mathbf{X}$

Figure 1. Manifold blurring mean shift algorithm (MBMS) and its particular cases Local Tangent Projection (LTP, k-nn graph, $\sigma = \infty$) and Gaussian Blurring Mean Shift (GBMS, $L = 0$). $\mathcal{N}_n$ contains all N points (full graph, MBMSf) or only $\mathbf{x}_n$'s nearest neighbors (k-nn graph, MBMSk).

Local Tangent Projection (LTP) algorithm (fig. 1): it is MBMSk with $\sigma = \infty$, or equivalently it replaces the GBMS step with the mean of the k nearest neighbors. Thus, each point projects onto its local tangent space, and the process is iterated. It is simpler (one parameter less) and almost as effective as MBMS. Finally, two other particular cases are PCA, for $\sigma = \infty$ and $k = N$, and no denoising (the dataset will not change), for $L = D$ or $\sigma = 0$.

Note the following remarks. (1) For given L, all versions of MBMS move points along the same direction (orthogonally) and only differ in the length of the motion. This length decreases monotonically with L because it is an orthogonal projection of the full-length motion (GBMS). The length increases with σ initially (more denoising) but may decrease for larger σ (as farther neighbors weigh in). (2) The GBMS coefficients in eq. 1 are updated at each iteration; not doing so is faster, but gives worse results. (3) All the algorithms admit online versions by moving points asynchronously, i.e., by placing the step "$\mathbf{x}_n \leftarrow \mathbf{x}_n + \partial\mathbf{x}_n$" inside the **for** loop.

How to set the parameters? If MBMS is embedded into another algorithm (e.g. classification), the most effective way to set the parameters is to cross-validate them with a test set, although this does add significant computation if other classifier parameters need to be cross-validated too; we do this in our MNIST experiments. Otherwise, the parameters have an intuitive meaning, and based on our experience it seems easy to find good regions for them:

- σ is related to the level of local noise outside the manifold. The larger σ is, the stronger the denoising effect; but too large σ can distort the manifold shape over iterations because of the effect of curvature and of different branches of the manifold. Using a smaller σ is safer but will need more iterations. Using a k-nn graph is even safer, as the motion is limited to near the k nearest neighbors and allows larger σ, in fact $\sigma = \infty$ yields the LTP method.
- k is the number of nearest neighbors that estimates the local tangent space; this is the easiest to set and we find MBMS quite robust to it. It typically grows sublinearly with N.
- L is the local intrinsic dimension; it could be estimated (e.g. using the correlation dimension) but here we fix it. If L is too small, it produces more local clustering and can distort the manifold; still, it can achieve pretty good results for good σ ($L = 0$ is GBMS, which can achieve some reasonable denoising, after all). If L is too large, points will move little ($L = D$: no motion).
- Number of iterations: in our experience, a *few* (1–3) iterations (with suitable σ) achieve most of the denoising; more iterations can refine this and achieve a better result, but eventually shrinkage arises.

We find MBMSf and MBMSk/LTP with a few iterations give the best results in low and high dimensions, resp., but using a k-nn graph (in partic. LTP) is generally a safer and faster option that achieves very similar results to MBMSf.

Convergence results We do not have a proof that MBMS converges for $L \geq 1$, but this is practically irrelevant since one would not run it for long anyway; best results (maximal denoising and minimal shrinkage) are usually obtained in 1–5 iterations. We do know the following (we omit detailed proofs for lack of space). (1) Any dataset that is contained in an L-dimensional linear manifold is a fixed point of MBMS (since the tangent space coincides with this manifold and tangential motion is removed). This holds no matter the geometry of the dataset (holes, etc.). Running MBMS for long does seem to converge to this. (2) A Gaussian distribution converges cubically to the linear manifold defined by its mean and L leading eigenvectors (from the proof technique of [1, 2], denoising proceeds independently along each principal axis but motion along the L leading eigenvectors is removed). Essentially, MBMS performs GBMS clustering orthogonal to the principal manifold.

Stopping criterion Because the denoising effect is strong, a practical indicator of whether we have achieved significant denoising while preventing shrinkage is the histogram over all data points of the orthogonal variance $\lambda_\perp$ (the sum of the trailing $k - L$ eigenvalues of $\mathbf{x}_n$'s local covariance). Its mean decreases drastically in the first few iterations (and would converge cubically to zero in the Gaussian case), while the mean of the histogram of the tangential variance $\lambda_\parallel$ decreases only slightly and stabilizes; see fig. 4. For curved manifolds, $\lambda_\perp$ tends to a positive value dependent on the local curvature.

Computational complexity Per iteration, this is $\mathcal{O}(N^2 D + N(D + k)\min(D, k)^2)$, where the first term is for finding nearest neighbors and for the mean-shift step, and the second for the local PCAs. If one uses the k-nn graph and does not update the neighbors at each iteration (which affects the result little) then the first term is negligible and the cost per iteration is linear on N; the one-off cost of computing the nearest neighbors is amortized if MBMS is followed by a spectral method for dimensionality reduction. Denoising a test point, as in our MNIST experiments, is $\mathcal{O}(ND)$.

3. Experiments

Noisy spiral with outliers Fig. 2 shows four versions of MBMS with a noisy spiral dataset ($N = 1\,000$ points with

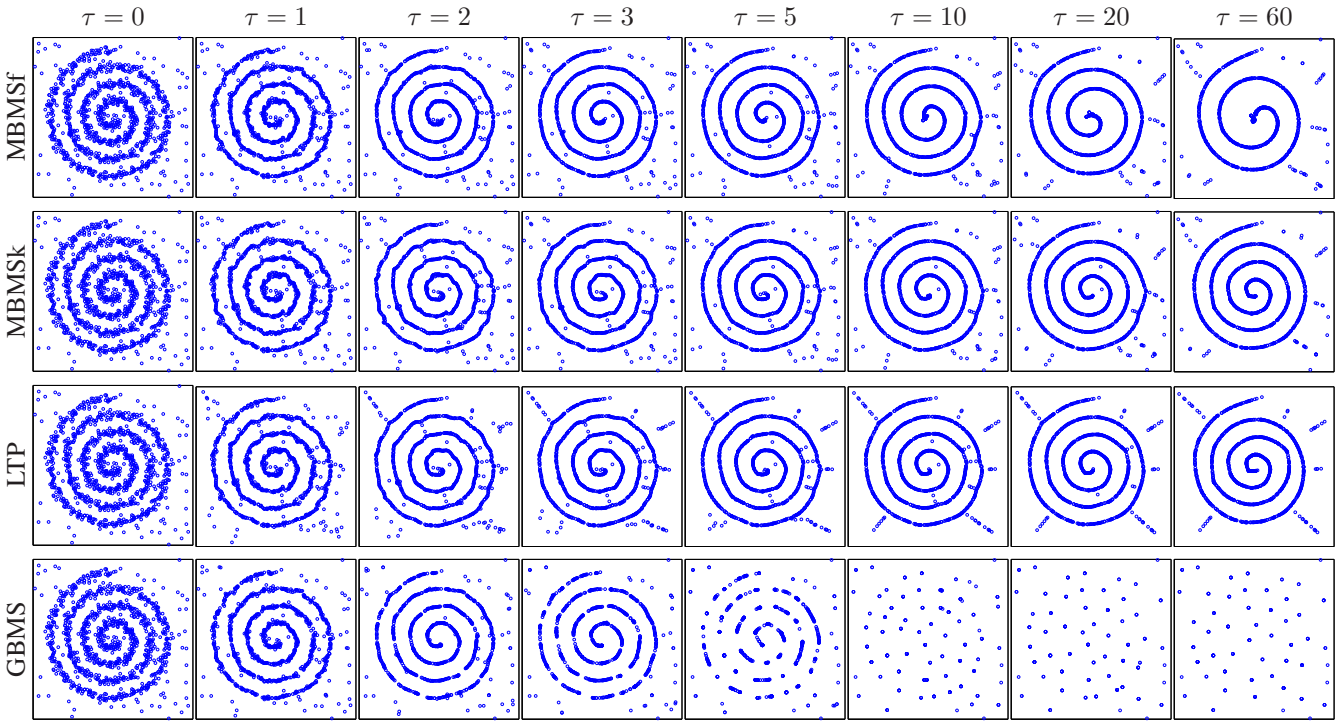


Figure 2. Denoising a spiral with outliers over iterations ($\tau = 0$ is the original dataset). Each box is the square $[-30, 30]^2$, where 100 outliers were uniformly added to an existing 1 000-point noisy spiral. Algorithms (L, k, σ): $(1, 10, 1.5)$ and full graph (MBMSf), $(1, 10, 1.5)$ and k-nn graph (MBMSk), $(1, 10, \infty)$ and k-nn graph (LTP), and $(0, \cdot, 1.5)$ and full graph (GBMS).

Gaussian noise) with 10% outliers added uniformly. GBMS ($L = 0$) clusters points locally and, while it denoises the manifold, it also visibly shrinks it tangentially, so already from the first iterations the boundaries shrink and points form multiple clusters along the manifold. When using $L = 1$ in MBMS to account for a curve, in-manifold movement is prevented and so these undesirable effects are reduced. The three versions with $L = 1$ behave very similarly for the first 5–10 iterations, achieving excellent denoising while being remarkably unaffected by outliers. Visually, the full graph (MBMSf) looks best, although it begins to be affected by shrinking much earlier than the k-nn graph versions (MBMSk and LTP); the inside of the spiral slowly winds in, and also the whole spiral shrinks radially. MBMSk and LTP preserve the spiral shape and size for far longer: after 200 iterations only a small radial shrinkage occurs. The reason is that the k-nn graph limits the influence on the mean-shift step of farther points (in regions with different curvature or even different branches); strong denoising (large σ) still occurs but is locally restricted. We have observed a similar behavior with other datasets.

After denoising for a few steps, outliers can be easily detected—the distance to their nearest neighbors is far larger than for non-outliers—and either removed, or projected on the tangent space of the k nearest neighbors on the manifold. The reason why they remain almost stationary and do not affect denoising of the mainstream points is simple. Points near the manifold (non-outliers) have no outliers as neighbors because the continuity of the manifold means all their neighbors will be near the manifold; neither the mean-shift step nor the tangent space estimation are affected, and these points move as if there were no outliers. Outliers have most neighbors somewhere near the manifold, and their tangent space is estimated as nearly orthogonal to the manifold at that point; they barely move, and remain isolated for many iterations (eventually they are denoised too, depending on how far they are from the manifold wrt k and σ). By this same reasoning, if MBMS is applied to disconnected manifolds, each will be denoised in isolation.

More complex shapes Fig. 3 shows a 1D manifold (two tangent ellipses) with a self-intersection, a gap, noise that varies depending on the manifold location, and a sharp density discontinuity. In spite of these varying conditions, MBMSf achieves very good denoising with a single (L, k, σ) value. Using the diffusion-map affinity normalization $\mathbf{D}^{-\alpha}\mathbf{W}\mathbf{D}^{-\alpha}$ of [6] with $\alpha = 1$ slightly improves the result, but with constant noise it has only negligible differences with our usual case ($\alpha = 0$).

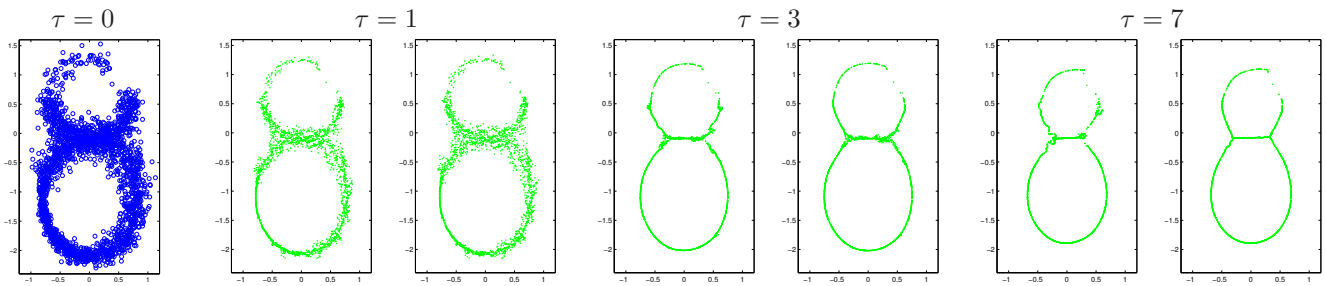


Figure 3. Denoising a complex shape with nonuniform density and noise with MBMSf $(1, 35, 0.2)$ using the usual affinity ($\alpha = 0$, left subplots) and the diffusion-map affinity normalization ($\alpha = 1$, right subplots). The upper partial ellipse has Gaussian noise of stdev 0.15 and the lower ellipse of stdev varying between 0 and to 0.2, with a sharp density discontinuity (top left).

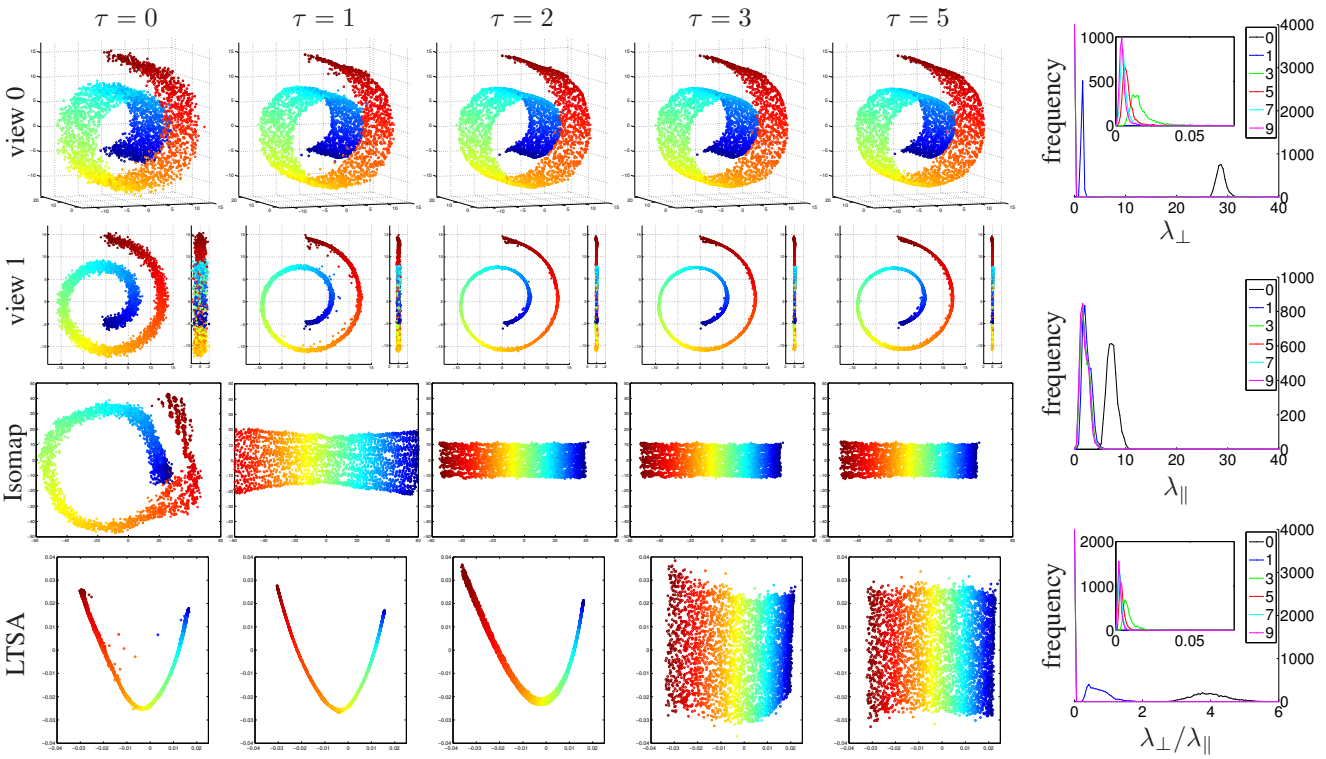


Figure 4. Dimensionality reduction with Isomap and LTSA for different iterations of MBMSk denoising (10–nearest-neighbor graph, $L = 2$, $k = 30$, $\sigma = 5$). $\tau = 0$ is the original Swiss roll dataset ($N = 4\,000$ points) lifted to 100 dimensions with additive Gaussian noise of stdev 0.6 in each dimension. Isomap/LTSA used a 10-nn graph. Isomap's residual variances [20] ($\tau = 0, 1, 2, 3, 5$): 0.3128, 0.0030, 0.0002, 0.0002, 0.0003. View 0 shows dimensions 1–3; view 1 shows dimensions 1, 2 (left subplot) and 2, 4 (right subplot). Right column: histograms over all data points of the normal, tangential, and normal/tangential ratio of the variances; the curves correspond to the iterations $\tau = 0, 1, 3, 5, 7, 9$, and the insets for $\lambda_\perp$ and $\lambda_\perp/\lambda_\parallel$ blow up the bins near zero (which contain all points for $\tau \geq 2$).

Dimensionality reduction Fig. 4 shows the k-nn-graph version (MBMSk) with a noisy Swiss roll in 100 dimensions (97 of which are noise). Isomap [20] and particularly LTSA [23] are sensitive to noise and to shortcuts in the neighborhood graph, but these are eliminated by MBMS. Excellent embeddings result for a wide range of iterations, and one can trade off a little more denoising with a little more shrinkage. In general, and depending on the level of noise, 2–3 iterations are often enough. The histograms show that the tangent space eigenvalues $\lambda_\parallel$ change little over iterations, i.e., there is little in-manifold motion. However, the normal space eigenvalues $\lambda_\perp$ drop drastically in the first 3 iterations (the histogram is a spike at almost zero) and then stay roughly constant (they do not become exactly zero because of the manifold curvature), indicating strong denoising orthogonal to the manifold, and signal-

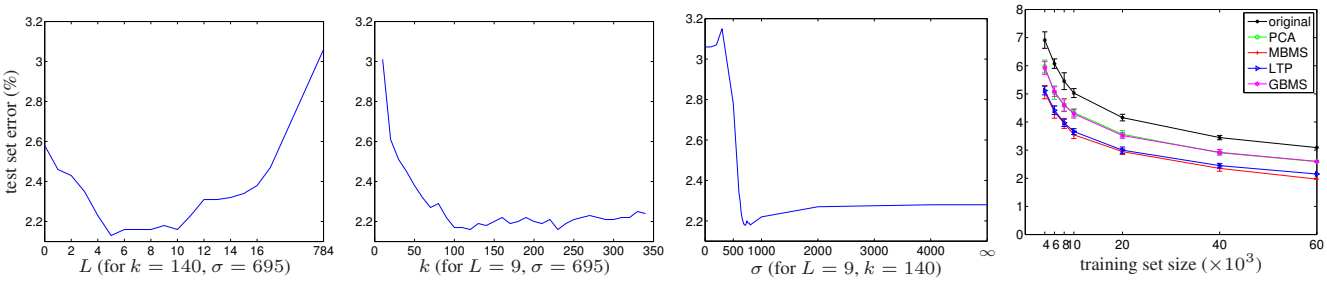
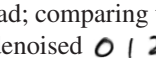
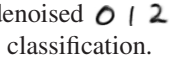


Figure 5. *Left 3 plots*: 5–fold cross-validation error (%) curves with a nearest-neighbor classifier on the entire MNIST training dataset (60k points, thus each fold trains on 48k and tests on 12k) using MBMSk; we selected $L = 9$, $k = 140$, $\sigma = 695$ as final values. *Right plot*: denoising and classification of the MNIST test set (10k points), by training on the entire training set (rightmost value) and also on smaller subsets of it (errorbars over 10 random subsets). Algorithms (L, k, σ), all using a k-nn graph: MBMSk $(9, 140, 695)$, LTP $(9, 140, \infty)$, GBMS $(0, 140, 600)$, and PCA $(L = 41)$.

ing a good stopping point. We repeated the experiment by adding 10% outliers within a box bounding the Swiss roll with essentially identical results (points near the manifold are denoised, outliers remain stationary), demonstrating the robustness of MBMS.

Classification of MNIST [13] digits It is reasonable to assume that much of the essential character (style, thickness, etc.) of a handwritten digit can be explained by a small number of degrees of freedom, so that MBMS denoising might preserve such a manifold while removing other types of noise; and that this may improve classification accuracy. Our setup is as follows. We use a nearest-neighbor classifier (like MBMS, a nonparametric method), which allows us to focus on the performance of MBMS without classifier-specific effects due to local optima, model selection, etc. As denoising methods we use PCA (i.e., projecting the data onto the L principal components' manifold) and 3 versions of MBMS using the k-nn graph and a single iteration: MBMSk, LTP and GBMS. We estimate (approximately) optimal parameters by 5–fold cross-validation by searching over a grid, denoising separately each class of the training fold ($N = 48\,000$ grayscale images of dimension $D = 784$, or 28×28 pixels) and measuring the classification error on the test fold (12 000 digits). For classification, the test points are fed directly (without denoising) to the nearest-neighbor classifier. Fig. 5 (left 3 plots) shows the MBMSk error curves over L, k and σ; notice how MBMSk improves the baseline error (no denoising, also achieved by $L = D = 784$ or $\sigma = 0$) of 3.06% over a very wide range of (L, k, σ). We chose $(9, 140, 695)$ and trained the models on the entire training set (60k points); fig. 5 (right plot) shows the test set classification error. MBMSk achieves 1.97% (a 36% relative decrease over the baseline of 3.09%); LTP $(9, 140, \infty)$ achieves a slightly larger error of 2.15% (30% relative decrease). GBMS and PCA also improve over the baseline but far less (2.59%, 14% decrease). These results are consistently confirmed over smaller training sets, even up to $N = 4\,000$ (right panel); we used the same parameters as for the entire set. The methods combining both clustering and manifold structure at the local level (MBMSk and LTP) are the clear winners. Judging from the trend of the curves, the relative error decrease would still grow with the training set size.

Other options also reduced the error, but less so (however, in all these cases we used the same parameters as above $(9, 140, 695)$, which are not optimal anymore). Denoising each test point (with one MBMSk iteration using the entire denoised training set): 2.23%. Denoising each test point but with the original training set: 2.42%. Denoising the entire training set without class information: 2.89%. The beneficial effect of MBMSk denoising in one way or another is clear.

Fig. 6 shows training images before and after denoising. The most obvious change is that the digits look smoother (as if they had been anti-aliased to reduce pixelation) and easier to read; comparing the original  vs the denoised  , one sees this would help classification. While this smoothing homogeneizes the digits somewhat, it preserves distinctive style aspects of each; excessive smoothing would turn each class into a single prototype image, and result in a Euclidean distance classifier (the method of [11] shows oversmoothing). MBMSk performs a sophisticated denoising (very different from simple averaging or filtering) by intelligently closing loops, removing or shortening spurious strokes, enlarging holes, removing speckle noise and, in general, subtly reshaping the digits while respecting their orientation, slant and thickness. We emphasize that we did not do any preprocessing of the data, and in particular no image-based preprocessing such as tangent distance, deskewing, or centering the images by bounding box (known to improve the nearest-neighbor classifier [13]). MBMS does not know that the data are images, and would give the same result if the pixels were reshuffled. Fig. 7 shows misclassified images.

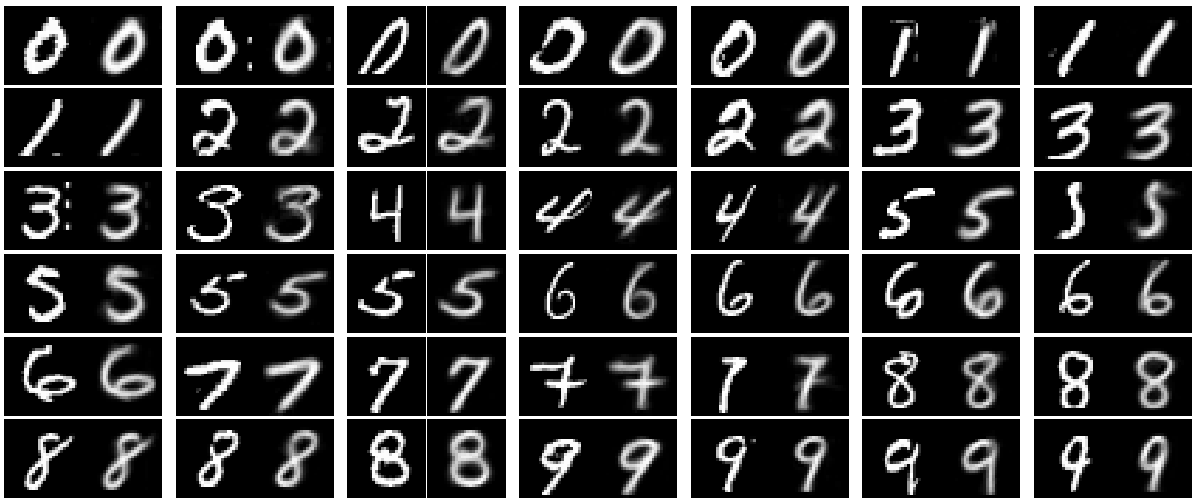


Figure 6. Sample pairs of (original,denoised) images from the training set. A few (2.62%) grayscale values outside the $[0, 255]$ training range have been clipped for visualization.

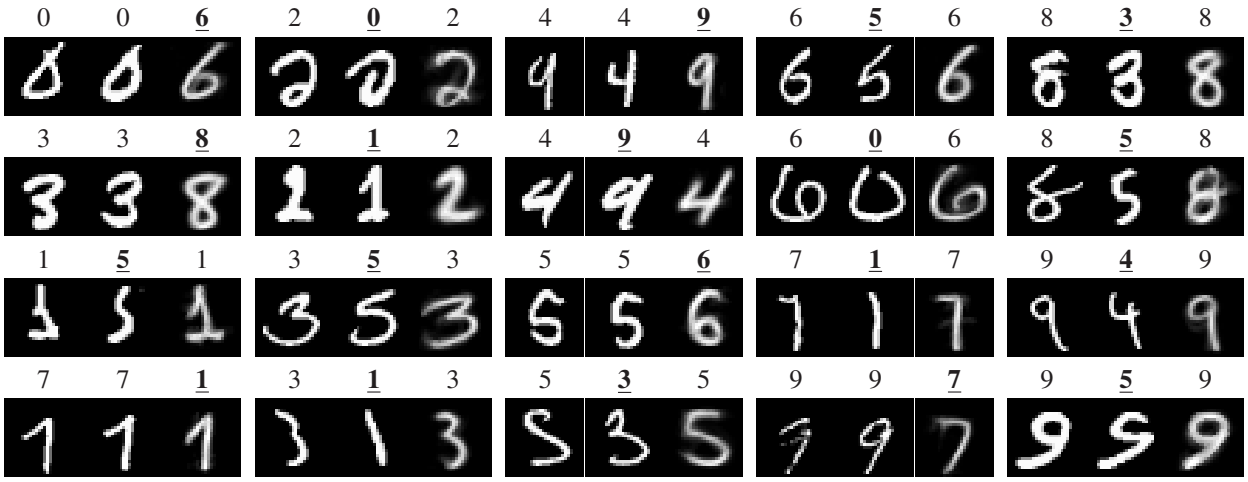


Figure 7. Some misclassified images. Each triplet is (test,original-nearest-neighbor,denoised-nearest-neighbor) and the corresponding label is above each image, with errors underlined. After denoising there are fewer errors, some of which are arguably wrong ground-truth labels.

4. Related work

MBMS can be seen as a hybrid between local clustering (blurring mean-shift) and local linear dimensionality reduction (PCA); in fact, it contains as particular cases GBMS (when manifold structure is lost: $L = 0$) and PCA (when locality is lost: $\sigma = \infty$ and $K = N$). In contrast, some previous denoising algorithms are really local clustering algorithms without a manifold constraint. The algorithm of [11] (see also [8]) is an implicit version of GBMS (it obtains $\mathbf{X}$ from an $N \times N$ linear system rather than a single matrix product, and is thus much slower) and suffers from significant shrinking within the manifold, as does GBMS. In [5], GBMS was used with a step size as preprocessing for density estimation. The method of [15] combines a local weighted PCA with several heuristics to deal with outliers, but has no local clustering component.

The computer graphics literature has considered a related denoising problem, that of 3D mesh smoothing (surface fairing), and several variations of GBMS have been used [19, 8, 12]. These methods, as well as computational geometry methods for curve and surface reconstruction [9], often rely on assumptions that may hold in 2D or 3D but not in the higher-dimensional problems typical of machine learning (e.g. availability of a polygonal mesh, or a Delaunay triangulation, with knowledge of normals, boundaries, junctions, etc.; dense sampling and very low noise).

Methods based on local regression first fit a (local) function to the data and then project each point onto the func-

tion [14, 16, 9, 22]. These methods are based on implicit functions and thus limited to manifolds of codimension 1 (the method of [22] uses explicit functions, but its computational cost grows quickly with the dimension). Also, these methods are not iterated, unlike MBMS. Essentially, these methods do local dimensionality reduction and then project the data (see below).

Tangent distance methods [17] perturb a data point according to known transformations to construct a linear approximation to the local space of ignorable distortions (for handwritten digits: translations, scaling, skewing, squeezing, rotation, line thickness variations). Jittering kernels are a similar idea for SVMs [7]. Mean-shift clustering has also been constrained to respect a known manifold structure, such as the matrix rotations group [18]. MBMS shares this spirit of eliminating undesirable degrees of freedom, but the transformations (off-manifold noise) are not known and are instead inferred locally. After denoising, the point distances better reflect their similarity wrt the manifold.

Any dimensionality reduction method that provides mappings from data to latent space and vice versa (such as PCA or an autoencoder) can be used for denoising. One first learns the parameters from the training set, and then maps each point to latent space, and then back to data space, thus projecting them on the model manifold. However, the point is that these methods (often sensitive to noise and/or local optima) may learn a better manifold if the training set is preprocessed to remove noise and outliers, and this is best done with a nonparametric method (such as MBMS) that imposes minimal model assumptions.

5. Conclusion

With adequate parameter values, the proposed MBMS algorithm is very effective at denoising in a handful of iterations a dataset with low-dimensional structure, even with extreme outliers, and causing very small shrinkage or manifold distortion. It is nonparametric and deterministic (no local optima); its only user parameters (L, k, σ) are intuitive and good regions for them seem easy to find. We also proposed LTP (local tangent projection), a particular, simple case of MBMS that has quasi-optimal performance and only needs L and k. We showed how preprocessing with MBMS improves the quality of algorithms for manifold learning and classification that are sensitive to noise or outliers, and expect this would apply to other settings with noisy data of intrinsic low dimensionality, such as density estimation, regression or semi-supervised learning.

Acknowledgments

Work supported by NSF CAREER award IIS–0754089.

References

[1] M. Á. Carreira-Perpiñán. Fast nonparametric clustering with Gaussian blurring mean-shift. *ICML*, 2006.

[2] M. Á. Carreira-Perpiñán. Generalised blurring mean-shift algorithms for nonparametric clustering. *CVPR*, 2008.

[3] M. Á. Carreira-Perpiñán and R. S. Zemel. Proximity graphs for clustering and manifold learning. *NIPS*, 2005.

[4] Y. Cheng. Mean shift, mode seeking, and clustering. *IEEE PAMI*, 17(8):790–799, Aug. 1995.

[5] E. Choi and P. Hall. Data sharpening as a prelude to density estimation. *Biometrika*, 86(4):941–947, Dec. 1999.

[6] R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proc. Natl. Acad. Sci. USA*, 102, 2005.

[7] D. DeCoste and B. Schölkopf. Training invariant support vector machines. *Machine Learning*, 46, 2002.

[8] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. *SIGGRAPH*, 1999.

[9] T. K. Dey. *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis*. Cambridge U. Press, 2007.

[10] K. Fukunaga and L. D. Hostetler. The estimation of the gradient of a density function, with application in pattern recognition. *IEEE Trans. Information Theory*, IT–21, 1975.

[11] M. Hein and M. Maier. Manifold denoising. *NIPS*, 2007.

[12] C. Lange and K. Polthier. Anisotropic smoothing of point sets. *Computer Aided Geometric Design*, 22, 2005.

[13] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, Nov. 1998.

[14] D. Levin. Mesh-independent surface interpolation. In G. Brunnett, B. Hamann, H. Müller, and L. Linsen, editors, *Geometric Modeling for Scientific Visualization*, pages 37–49. Springer-Verlag, 2003.

[15] J. Park, Z. Zhang, H. Zha, and R. Kasturi. Local smoothing for manifold learning. *CVPR*, 2004.

[16] M. Pauly, L. Kobbelt, and M. Gross. Point-based multiscale surface representation. *ACM Trans. Graphics*, 2006.

[17] P. Simard, Y. LeCun, , and J. Denker. Efficient pattern recognition using a new transformation distance. *NIPS*, 1993.

[18] R. Subbarao and P. Meer. Nonlinear mean shift over Riemannian manifolds. *Int. J. Computer Vision*, 84, 2009.

[19] G. Taubin. A signal processing approach to fair surface design. *SIGGRAPH*, 1995.

[20] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, Dec. 22 2000.

[21] M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, Feb. 1999.

[22] R. Unnikrishnan and M. Hebert. Denoising manifold and non-manifold point clouds. *British Machine Vision Conference (BMVC)*, 2007.

[23] Z. Zhang and H. Zha. Principal manifolds and nonlinear dimension reduction via local tangent space alignment. *SIAM J. Sci. Comput.*, 26(1):313–338, 2004.