AN EFFICIENT METHOD FOR FINDING INTERSECTIONS OF MANY MANIFOLDS WITH APPLICATION TO PATCH BASED IMAGE PROCESSING

Yevgen Matviychuk, Shannon M. Hughes
Department of Electrical, Computer, and Energy Engineering
University of Colorado at Boulder

ABSTRACT

Solving inverse problems in signal processing often involves making prior assumptions about the signal being reconstructed. Here the appropriateness of the chosen model greatly determines the quality of the final result. Recently it has been proposed to model images by representing them as sets of smaller patches arising from an underlying manifold. This model has been shown to be surprisingly effective in tasks such as denoising, inpainting, and superresolution. However, such a representation is fraught with the difficulty of finding the intersection of many presumably non-linear manifolds in high-dimensional space, which precludes much of its potential use. This paper proposes an efficient method of solving this problem using a kernel methods variant of the projection onto convex sets algorithm to quickly find the intersection of many manifolds while learning their non-linear structure. Indeed the final solution can even be expressed in closed form. We foresee our method allowing a patch-based regularization to be applied across a wide variety of inverse problems, including compressive sensing, inpainting, deconvolution, etc. Indeed, as a proof of concept of our approach, we show how it can be employed in the regularization of an image denoising problem. Here, it even outperforms a state-of-the-art denoising technique, non-local means.

Index Terms— Patch-based image processing, inverse problems, kernel methods, manifold models, projection algorithms.

1. INTRODUCTION

A typical (linear) inverse problem in signal processing is formulated as the problem of reconstructing a signal $\boldsymbol{y}^*$ given the vector of measurements $\boldsymbol{z}$, obtained by the transformation:

$$\boldsymbol{z} = \boldsymbol{A}\boldsymbol{y}^* + \boldsymbol{n}. \tag{1}$$

Usually the operator $\boldsymbol{A}$ has rank less then the number of dimensions of $\boldsymbol{y}^*$, which makes the problem underdetermined with infinitely many solutions. An example is reconstruction of a signal from its compressive sensing measurements obtained with the measurement matrix $\boldsymbol{A}$. Another common example is denoising, where $\boldsymbol{A} = \boldsymbol{I}$, the identity matrix. Here additive noise $\boldsymbol{n}$ also makes the inverse problem ill-posed. Other examples are image inpainting, deblurring, and superresolution. In each case, in order to reconstruct a plausible $\boldsymbol{y}^*$, additional assumptions about the sought signal (for example, those of sparsity in a wavelet basis or low total variation) should be made, which allow us to select the best estimate of the true solution from the reduced set of possible solutions.

In image processing, it is often particularly desirable to retain the sharp appearance of fine structural elements and patterns. To describe these features, models employing smaller pieces (patches) of the image are particularly effective. Exploiting similarity between patches was shown by Buydes et al. [1] to be very efficient denoising method (known as the non-local means algorithm). Moreover, Gaussian mixture models applied on image patches, which although requires sufficient computational resources on training stage, allowes to solve broader class of inverse problems [2, 3]. On the other hand, patch-based methods for structural image editing (inpainting, retargeting, reshuffling, etc.) have also been surprisingly effective [4, 5]. The observation that patches, even though sampled from relatively high dimensional space ($\mathbb{R}^{25}$ for 5×5 patches), can be parametrized by far fewer coordinates gives rise to the manifold model, which imposes mutual constraints on the pixel values of each patch. This was observed by Lee et al., who studied the statistical properties of image patches and corresponding manifolds in [6] suggesting their non-linearity and Carlsson et al., who showed in [7] that the patch manifold of the space of high-contrast images has the topology of a Klein bottle.

However, one of the obstacles to using the manifold model for patches is the problem of describing the whole image in terms of patches. For example, nonparametric Bayesian model used by Chen et al. [8] can be applied to describe the manifold corresponding to one particular patch but does not extend to the case of overlapping patches. Peyré in [9] uses the concept of manifold energy to regularize inverse problems, which leads to tracing the trajectory of reconstructed signal on the patch manifold. The main drawback of this method is the necessity of iterative explicit projections onto the densely sampled non-linear manifold, which creates a computational burden. We will show a fast and efficient way of solving this problem that allows sparse sampling of the manifold.

In our approach, instead of regarding an image as a trajectory on a single patch manifold, we consider the whole D-dimensional space of images, where D equals the number of pixels. For any $p \times q$-area of the image pixel grid there is a corresponding pq-dimensional subspace of $\mathbb{R}^D$. The manifold model allows us to assume that patches extracted from similar images lie on or close to a d-dimensional non-linear manifold (with $d < pq$) within this subspace. Therefore, the whole image lies on a $D - pq + d$-dimensional manifold of $\mathbb{R}^D$. Because there is one manifold constraint corresponding to each overlapping patch, the image lies at or close to the intersection of all these manifolds. The number of distinct manifolds (related to every possible patch position) is on the order of total number of pixels, which in addition to their non-linear geometry, makes the problem of finding intersections seemingly very hard.

The main idea of this paper is to use the kernel trick [10] to deal efficiently with non-linearity of manifolds. In kernel methods, points $\boldsymbol{x}$ are mapped by some non-linear transformation Φ to a higher-dimensional feature space, in which they can be analyzed by linear methods. Eventually, this yields a non-linear solution when mapped back to the original space. Efficiency is gained by the fact that the images $\Phi(\boldsymbol{x})$ do not need to be computed explicitly. Instead, the kernel function $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j) \rangle$ is defined to relate the inner products in the original and feature spaces. This

This material is based upon work supported by the National Science Foundation under Grant CCF-1117775.

allows us to obtain non-linear versions of popular linear machine learning algorithms, such as Support Vector Machines, Ridge Regression, and Principal Component Analysis (PCA), almost without increasing their computational complexity [10, 11].

In particular, the kernel PCA (KPCA) algorithm is one of the most powerful known methods for learning the structure of a manifold from its samples. Indeed, other popular manifold learning algorithms, such as Laplacian Eigenmaps [12] and Isomap [13], were shown in [14] to be special cases of it. The key idea of KPCA is that, in the induced feature space, the manifold becomes linear and can be parametrized by PCA. Its effectiveness has been proved in many signal processing settings. Besides direct application of KPCA for denoising [15], it was used in [16] to locally parametrize a patch manifold for the purpose of image deconvolution.

We will employ KPCA to linearize the manifolds intersection problem in feature space. This will allow us to compute projections onto the manifolds easily, iteratively find their intersection, and finally express the solution in closed form.

In Section 2, we will review the iterative projection method applied to the manifolds parametrized by their principal components. The third section is dedicated to expressing the solution entirely in terms of inner products and extending the algorithm to a higher-dimensional non-linear feature space. The details of a possible application of the proposed algorithm for image denoising are described in Section 4 and experimental results are presented in Section 5.

2. SUBSPACE INTERSECTION METHOD REVIEW

We base our model on the assumption that the solution to the inverse problem (1) lies on or close to the manifolds corresponding to different overlapping image patches, i.e. out of all possible solutions to the inverse problem, we want the one that minimizes:

$$\min_{\boldsymbol{y}} \sum_{m=1}^{M} d^2\left(\boldsymbol{y}, \mathcal{M}_m\right), \tag{2}$$

where $d\left(\boldsymbol{y}, \mathcal{M}_m\right) = \inf_{\boldsymbol{x} \in \mathcal{M}_m} d\left(\boldsymbol{y}, \boldsymbol{x}\right)$ is the Euclidean distance from point $\boldsymbol{y}$ to the m-th manifold.

Suppose first that each of the manifolds $\mathcal{M}_m$ is merely a d_m-dimensional affine subspace of $\mathbb{R}^D$ (we will generalize this in Section 3). Note that the manifolds here are permitted to have different dimensions. Then, any point $\boldsymbol{x}_i \in \mathcal{M}_m$ can be expressed by its subspace coordinates $\boldsymbol{x}'_{m,i}$ as:

$$\boldsymbol{x}_i = \boldsymbol{U}_m \boldsymbol{x}'_{m,i} + \boldsymbol{\mu}_m\,, \tag{3}$$

where $\boldsymbol{U}_m$ is the matrix whose columns form an orthonormal basis for the subspace $\mathcal{M}_m$ after eliminating the offset $\boldsymbol{\mu}_m$.

Problem (2) can then be solved by projecting $\boldsymbol{y}$ onto each subspace, and therefore reduces to finding the least squares solution of a (possibly underdetermined) system of linear equations:

$$(\boldsymbol{I} - \boldsymbol{U}_m \boldsymbol{U}_m^T)\boldsymbol{y}^* = \boldsymbol{\mu}_m,\ m = 1, \ldots, M, \tag{4}$$

where $\boldsymbol{\mu}_m$ is assumed to be orthogonal to the columns of $\boldsymbol{U}_m$. To solve it we can apply Cimmino's iterative method, which is related to the Landweber method and the projection onto convex sets algorithm [17]. Its idea is, starting with some initial point $\boldsymbol{y}^{(0)}$, to find projections of the current solution onto all manifolds and average them to get the next step approximation:

$$\boldsymbol{y}^{(k+1)} = \sum_{m=1}^{M} w_m \mathcal{P}_m\left(\boldsymbol{y}^{(k)}\right), \tag{5}$$

where w_m are non-negative weights that satisfy $\sum_{m=1}^{M} w_m = 1$. Proof of general convergence properties of the Landweber iteration can be found in [18]. Figure 1 shows a graphical interpretation of this algorithm for two subspaces.

Notice that this procedure results in a relevant solution regardless of whether the system of Eq. 4 is under- or overdetermined. In the first case, the answer, which is the intersection point closest to the initial guess, is not affected by the weights w_i. This is the behavior, one would want, for example, for a denoising purpose, assuming that the initial point is the noisy image. When the system of Eq. 4 has no solution (if it defines, for example, parallel hyperplanes or non-intersecting lines in $\mathbb{R}^3$), the iteration converges to the point that minimizes the sum of squared distances to all manifolds. In this case, non-equal weights w_i may be introduced to make the solution closer to one subspace than another.

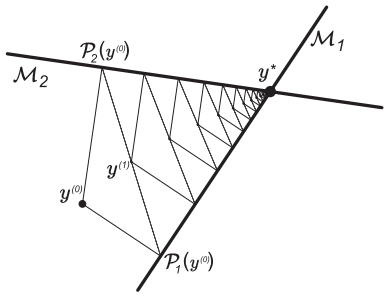


Fig. 1. Finding the intersection of two affine subspaces by the iterative projection algorithm given by Eq. 5.

3. KERNELIZING THE INTERSECTION ALGORITHM

We now show that the solution defined iteratively by Eq. 5 can be written entirely in terms of inner products, so we may use the kernel trick to extend it from linear subspaces to non-linear manifolds.

First we attempt to learn a description of each manifold $\mathcal{M}_m$ from its n_m samples (training points) $\boldsymbol{x}_{m,i}$. It can be specified by its principal components $\boldsymbol{u}_{m,k}$, $k = 1 \ldots d_m$ obtained using PCA and the offset vector. We consider the algorithm formulated in terms of the centered inner product matrix $\boldsymbol{K}_m$ with elements $(\boldsymbol{K}_m)_{i,j} = \langle \tilde{\boldsymbol{x}}_{m,i}, \tilde{\boldsymbol{x}}_{m,j} \rangle$, rather than the covariance matrix, to allow future extension to a higher-dimensional feature space by kernel PCA [11, 14]. Then:

$$\boldsymbol{u}_{m,k} = \sum_{i=1}^{n} \tilde{\boldsymbol{x}}_{m,i} \alpha_{m,i,k}\,, \tag{6}$$

where $\alpha_{m,:,k}$ denote values of the k-th eigenvector of $\boldsymbol{K}_m$ divided by the square root of the corresponding eigenvalue $\frac{1}{\sqrt{\lambda_{m,k}}}$ to achieve normalization in feature space. Here $\tilde{\boldsymbol{x}}_{m,i} = \boldsymbol{x}_{m,i} - \boldsymbol{\mu}_m$ denotes the centered manifold samples with $\boldsymbol{\mu}_m$ defined as the sample mean $\boldsymbol{\mu}_m = \frac{1}{n_m}\sum_{i=1}^{n_m} \boldsymbol{x}_{m,i} = \frac{1}{n_m}\boldsymbol{X}_m \boldsymbol{1}_{n_m \times 1}$. We observe that since all eigenvectors $\alpha_{m,:,k}$ are orthogonal to the vector of constant values $\boldsymbol{1}$, using non-centered samples $\boldsymbol{x}_{m,i}$ in Eq. 6 does not change the values of $\boldsymbol{u}_{m,k}$.

Principal components of the m-th manifold can then be written as $\boldsymbol{U}_m = \boldsymbol{X}_m \boldsymbol{\alpha}_m^T$, where $\boldsymbol{X}_m$ is a $D \times n_m$ matrix of manifold samples arranged in columns and $\boldsymbol{\alpha}_m$ is a $d_m \times n_m$ matrix of scaled eigenvectors of $\boldsymbol{K}_m$. Then, the projection of point $\boldsymbol{y}$ onto the m-th manifold in the original coordinate system is:

$$\mathcal{P}_m\left(\boldsymbol{y}\right) = \boldsymbol{U}_m \boldsymbol{U}_m^T \boldsymbol{y} + (\boldsymbol{I} - \boldsymbol{U}_m \boldsymbol{U}_m^T)\boldsymbol{\mu}_m\,. \tag{7}$$

By substituting Eq. 7 into Eq. 5 and observing that $\boldsymbol{y}^{(k+1)} = \boldsymbol{S}\boldsymbol{y}^{(k)} + \boldsymbol{M}$, it can be shown that the K-th step approximation of the solution takes the form:

$$\boldsymbol{y}^{(K)} = \boldsymbol{S}^K \boldsymbol{y}^{(0)} + \sum_{k=0}^{K-1} \boldsymbol{S}^k \boldsymbol{M}, \tag{8}$$

where $\boldsymbol{S} = \sum_{m=1}^{M} w_m \boldsymbol{U}_m \boldsymbol{U}_m^T = \sum_{m=1}^{M} w_m \boldsymbol{X}_m \boldsymbol{\alpha}_m^T \boldsymbol{\alpha}_m \boldsymbol{X}_m^T$,

$$\begin{aligned} \boldsymbol{M} &= \sum_{m=1}^{M} w_m \left(\boldsymbol{I} - \boldsymbol{U}_m \boldsymbol{U}_m^T\right) \boldsymbol{\mu}_m \\ &= \sum_{m=1}^{M} w_m \boldsymbol{X}_m \boldsymbol{c}_m\,, \end{aligned} \tag{9}$$

with $\boldsymbol{c}_m = \left(\boldsymbol{I} - \boldsymbol{\alpha}_m^T \boldsymbol{\alpha}_m \boldsymbol{X}_m^T \boldsymbol{X}_m\right) \frac{1}{n_m} \boldsymbol{1}$, a $n_m \times 1$ vector, computed using only inner products.

Let $\boldsymbol{V}_m = \sqrt{w_m}\boldsymbol{U}_m = \sqrt{w_m}\boldsymbol{X}_m\boldsymbol{\alpha}_m^T$ to simplify the notation. Then, for $k > 0$, $\boldsymbol{S}^k = \left(\sum_{m=1}^M \boldsymbol{V}_m\boldsymbol{V}_m^T\right)^k$ can be represented as a product of block vectors and matrices with $i, j = 1, \ldots, M$:

$$\boldsymbol{S}^k = \sum_{m_1=1}^M \cdots \sum_{m_k=1}^M \boldsymbol{V}_{m_1}\boldsymbol{V}_{m_1}^T\boldsymbol{V}_{m_2}\boldsymbol{V}_{m_2}^T \ldots \boldsymbol{V}_{m_k}\boldsymbol{V}_{m_k}^T$$

$$= \begin{bmatrix} \cdots & \boldsymbol{V}_i & \cdots \end{bmatrix} \begin{bmatrix} & \vdots & \\ \cdots & \boldsymbol{V}_i^T\boldsymbol{V}_j & \cdots \\ & \vdots & \end{bmatrix}^{k-1} \begin{bmatrix} \vdots \\ \boldsymbol{V}_j^T \\ \vdots \end{bmatrix}.$$

Let us then define the following block matrices, each with entries expressed entirely in terms of inner products:

- the $\sum d_m \times 1$-dimensional vector $\boldsymbol{h}$ with M block entries $\boldsymbol{h}_{[i]} = \boldsymbol{V}_i^T\boldsymbol{y}^{(0)} = \sqrt{w_i}\boldsymbol{\alpha}_i\boldsymbol{X}_i^T\boldsymbol{y}^{(0)}$ arranged vertically;
- the $\sum d_m \times 1$-dimensional vector $\boldsymbol{g}$ with M block entries $\boldsymbol{g}_{[i]} = \boldsymbol{V}_i^T\boldsymbol{M} = \sqrt{w_i}\sum_{m=1}^M w_m\boldsymbol{\alpha}_i\boldsymbol{X}_i^T\boldsymbol{X}_m\boldsymbol{c}_m$ arranged vertically;
- the $\sum d_m \times \sum d_m$-dimensional matrix $\boldsymbol{H}$ with block entries $\boldsymbol{H}_{[i,j]} = \boldsymbol{V}_i^T\boldsymbol{V}_j = \sqrt{w_i}\boldsymbol{\alpha}_i\boldsymbol{X}_i^T\boldsymbol{X}_j\boldsymbol{\alpha}_j^T\sqrt{w_j}$.

Using this notation, $\boldsymbol{S}^k = \sum_{i=1}^M\sum_{j=1}^M \boldsymbol{V}_i\boldsymbol{H}_{[i,j]}^{k-1}\boldsymbol{V}_j^T$, and also $\boldsymbol{S}^k\boldsymbol{y}^{(0)} = \sum_{m=1}^M \boldsymbol{V}_m\boldsymbol{H}_{[m,:]}^{k-1}\boldsymbol{h}$ and $\boldsymbol{S}^k\boldsymbol{M} = \sum_{m=1}^M \boldsymbol{V}_m\boldsymbol{H}_{[m,:]}^{k-1}\boldsymbol{g}$, where $\boldsymbol{H}_{[m,:]}^{k-1} = \left[\boldsymbol{H}_{[m,1]}^{k-1}, \boldsymbol{H}_{[m,2]}^{k-1}, \cdots, \boldsymbol{H}_{[m,M]}^{k-1}\right]$.

Finally, let $\boldsymbol{s} = \boldsymbol{H}^{K-1}\boldsymbol{h} + \sum_{k=1}^{K-1}\boldsymbol{H}^{k-1}\boldsymbol{g}$, which can be split into blocks of lengths d_m as $\boldsymbol{s}_{[m]} = \boldsymbol{s}_{\sum_{i=1}^{m-1} d_i + 1 \ldots \sum_{i=1}^m d_i}$.

Then rewriting Eq. 8 for the K-th step approximation of the solution we get:

$$\begin{aligned} \boldsymbol{y}^{(K)} &= \boldsymbol{S}^K\boldsymbol{y}^{(0)} + \sum_{k=1}^{K-1}\boldsymbol{S}^k\boldsymbol{M} + \boldsymbol{M} \\ &= \sum_{m=1}^M \boldsymbol{V}_m\boldsymbol{H}_{[m,:]}^{K-1}\boldsymbol{h} + \sum_{k=1}^{K-1}\sum_{m=1}^M \boldsymbol{V}_m\boldsymbol{H}_{[m,:]}^{k-1}\boldsymbol{g} + \boldsymbol{M} \\ &= \sum_{m=1}^M \sqrt{w_m}\boldsymbol{X}_m\boldsymbol{\alpha}_m^T\boldsymbol{s}_{[m]} + \sum_{m=1}^M w_m\boldsymbol{X}_m\boldsymbol{c}_m \\ &= \sum_{m=1}^M \boldsymbol{X}_m\boldsymbol{\gamma}_m\,, \end{aligned} \tag{10}$$

where $\boldsymbol{\gamma}_m = \sqrt{w_m}\boldsymbol{\alpha}_m^T\boldsymbol{s}_{[m]} + w_m\boldsymbol{c}_m$.

Since computing $\boldsymbol{\gamma}_m$ involves only evaluation of inner products, this algorithm can be easily extended to the non-linear case by substituting the entries of inner product matrices $\langle \boldsymbol{x}_i, \boldsymbol{x}_j\rangle$ with corresponding values of the kernel function $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Therefore, the intersection of subspaces is sought in an implicitly induced higher-dimensional feature space, which corresponds to finding the intersection of non-linear manifolds in the original space. The preimage $\boldsymbol{y}^*$ of the solution $\Phi(\boldsymbol{y}^{(K)}) = \sum_{m=1}^M\sum_{i=1}^{n_m}\Phi(\boldsymbol{x}_i^{(m)})\boldsymbol{\gamma}_i^{(m)}$ can thus be found by minimizing the distance $\left\|\Phi(\boldsymbol{y}^*) - \Phi(\boldsymbol{y}^{(K)})\right\|^2$ in feature space. The form of Eq. 10 allows use of any of the preimage methods described in [10, 19, 20]. The entire algorithm is summarized in Table 1.

Moreover, we notice that the number of samples in training sets $\boldsymbol{X}_m$, which were used to learn parameters of manifolds, grows rapidly with the size of the patches. Due to the form of the final solution (Eq. 10), this directly affects the running time. However, sparse (in terms of training data) approximation of the principal components in feature space can be found, for example, by the greedy method described in [21]. This allows us to compute the kernel matrices for only very small subsets of the training points $\boldsymbol{x}_i^{(m)}$. This approach reduces the running time by several orders of magnitude and was crucial to produce the results in Section 5.

Table 1. Manifolds intersection algorithm.

Inputs:

- training samples $\boldsymbol{x}_i^{(m)}$, $i = 1 \ldots n_m$ of M manifolds;
- initial approximation to the solution $\boldsymbol{y}^{(0)}$;
- type and parameters of kernel function $\kappa(\cdot,\cdot)$;
- number of iterations K.

Output: point $\boldsymbol{y}^*$ in original space that optimizes the criterion in Eq. 2.

for each manifold m

Create kernel matrices with entries
$\tilde{\boldsymbol{K}}_{i,j}^{(m,m)} \leftarrow \kappa\left(\boldsymbol{x}_i^{(m)}, \boldsymbol{x}_j^{(m)}\right), i, j = 1, \ldots, n_m,$
$\boldsymbol{K}_i^{(m,y)} \leftarrow \kappa\left(\boldsymbol{x}_i^{(m)}, \boldsymbol{y}^{(0)}\right), i, j = 1, \ldots, n_m;$
Center the data in feature space by computing [11]:
$\boldsymbol{K}^{(m,m)} \leftarrow \left(\boldsymbol{I} - \frac{1}{n_m}\boldsymbol{1}\boldsymbol{1}^T\right)\tilde{\boldsymbol{K}}_{i,j}^{(m,m)}\left(\boldsymbol{I} - \frac{1}{n_m}\boldsymbol{1}\boldsymbol{1}^T\right);$
Find the eigendecomposition of
$\boldsymbol{K}^{(m,m)} = \boldsymbol{A}_m\boldsymbol{\Lambda}_m\boldsymbol{A}_m^T;$
Choose d_m leading eigenvectors and scale them:
$\left[\boldsymbol{\alpha}_{:,i}^{(m)}\right]^T = \boldsymbol{A}_{:,i}^{(m)}\frac{1}{\sqrt{\boldsymbol{\Lambda}_{i,i}^{(m)}}}, \; i = 1 \ldots d_m;$
$\hat{\boldsymbol{h}}_m \leftarrow \sqrt{w_m}\boldsymbol{\alpha}_m\boldsymbol{K}^{(m,y)}; \hat{\boldsymbol{g}}_m \leftarrow \boldsymbol{0}_{d_m};$
for $l = 1 \ldots M$
$\boldsymbol{K}_{i,j}^{(m,l)} \leftarrow \kappa\left(\boldsymbol{x}_i^{(m)}, \boldsymbol{x}_j^{(l)}\right), \quad \begin{matrix} i = 1 \ldots n_m, \\ j = 1 \ldots n_l; \end{matrix}$
$\boldsymbol{c}_l \leftarrow \left(\boldsymbol{I} - \boldsymbol{\alpha}_l^T\boldsymbol{\alpha}_l\boldsymbol{K}^{(l,l)}\right)\frac{1}{n_l}\boldsymbol{1}_{n_l};$
$\hat{\boldsymbol{g}}_m \leftarrow \hat{\boldsymbol{g}}_m + \sqrt{w_m}w_l\boldsymbol{\alpha}_m\boldsymbol{K}^{(m,l)}\boldsymbol{c}_l;$
$\check{\boldsymbol{H}}_{m,l} \leftarrow \sqrt{w_m}\boldsymbol{\alpha}_m\boldsymbol{K}^{(m,l)}\boldsymbol{\alpha}_l^T\sqrt{w_l};$
endfor
$\hat{\boldsymbol{H}}_m \leftarrow concatenate_{rows}(\check{\boldsymbol{H}}_{m,1}, \check{\boldsymbol{H}}_{m,2}, \ldots, \check{\boldsymbol{H}}_{m,M});$
endfor
$\boldsymbol{h} \leftarrow concatenate_{columns}(\hat{\boldsymbol{h}}_1, \hat{\boldsymbol{h}}_2, \ldots, \hat{\boldsymbol{h}}_M);$
$\boldsymbol{g} \leftarrow concatenate_{columns}(\hat{\boldsymbol{g}}_1, \hat{\boldsymbol{g}}_2, \ldots, \hat{\boldsymbol{g}}_M);$
$\boldsymbol{H} \leftarrow concatenate_{columns}(\hat{\boldsymbol{H}}_1, \hat{\boldsymbol{H}}_2, \ldots, \hat{\boldsymbol{H}}_M);$
$\boldsymbol{s} \leftarrow \boldsymbol{H}^{K-1}\boldsymbol{h} + \sum_{k=1}^{K-1}\boldsymbol{H}^{k-1}\boldsymbol{g};$
for $m = 1 \ldots M$
$\hat{\boldsymbol{s}}_m \leftarrow \boldsymbol{s}_{\sum_{i=1}^{m-1} d_i + 1 \ldots \sum_{i=1}^m d_i};$
$\boldsymbol{\gamma}_m \leftarrow \sqrt{w_m}\boldsymbol{\alpha}_m^T\hat{\boldsymbol{s}}_m + w_m\boldsymbol{c}_m;$
endfor
Solve the preimage problem [10, 20, 19]:
$\hat{\boldsymbol{y}}^* = \underset{\boldsymbol{y}}{argmin}\left\|\Phi(\boldsymbol{y}) - \sum_{m=1}^M\sum_{i=1}^{n_m}\Phi(\boldsymbol{x}_i^{(m)})\boldsymbol{\gamma}_i^{(m)}\right\|^2.$

Note: In this table, indexes m, l refer to entire matrices or vectors, and i, j denote scalar entries of a particular matrix.

Finally, we note that our algorithm can be run with several initial conditions simultaneously by replacing the vector $\boldsymbol{y}^{(0)}$ with a matrix $\boldsymbol{Y}^{(0)}$ with initial conditions arranged columnwise, which can further reduce computational burden.

4. APPLICATION TO IMAGE DENOISING

To demonstrate the use of our algorithm in a practical patch-based image processing application, we describe how it can be employed for solving denoising problems. While we use denoising as a proof of concept here, we emphasize that the main strength of our method lies not in its application to patch-based image denoising, for which there are already several other efficient algorithms such as [1, 22, 15, 23], but in its potential broad applicability across the spectrum

of possible inverse problems. These denoising algorithms rely crucially on the fact that local neighborhoods are roughly preserved as they pass through the operator $\boldsymbol{A}$ in Eq. 1, so one can solve the denoising problem locally, e.g. by working on patches of $\boldsymbol{z}$. This makes these sorts of algorithms unsuitable for more general inverse problems such as compressive sensing in which $\boldsymbol{A}$ does not preserve locality, and the entries of $\boldsymbol{z}$ reflect only global behaviors of the image $\boldsymbol{y}^*$. By contrast, since our method creates a tractable model for the whole image (as the union of many overlapping patches), it can be applied also to more general inverse problems such as compressive sensing. This will be explored further in future work.

For a denoising problem, given a library of training images of a particular class (for example, images with high contrast smooth edges), we form the set of all $p \times q$ patches extracted from them. Then, for a D-pixel noisy image $\boldsymbol{I}_\sigma$, instead of finding the intersection of patch manifolds in $\mathbb{R}^D$ directly, we consider a set of its $P \times Q$ regions ($P < 2p$, $Q < 2q$). We tile each region $\boldsymbol{R}_j$ with $(P-p+1)(Q-q+1)$ (the maximum number) of patches such that they all overlap in the middle $(2p-P) \times (2q-Q)$-pixel area. Now to find the intersection of manifolds $\boldsymbol{R}_j^* \in \mathbb{R}^{PQ}$ for each $\mathcal{M}_m$ we define the set of training samples $\boldsymbol{X}_m$, $m = 1 \ldots (P-p+1)(Q-q+1)$ by augmenting the training set of patches (that are vectors in $\mathbb{R}^{pq}$) with $PQ-pq$ lacking pixels of constant value (for example, gray pixels). Only center pixels of each region $\boldsymbol{R}_j^*$, computed by the algorithm described in Section 3, are retained. Proceeding in the same way for the rest of pixels in $\boldsymbol{I}_\sigma$ we find the denoised image $\boldsymbol{I}_\sigma^*$.

5. EXPERIMENTAL RESULTS AND DISCUSSION

First, we will run our algorithm on a small synthetic example to demonstrate that it finds the intersection of smooth surfaces. Then we will show its application to regularizing the denoising problem. In all considered examples we use the Gaussian kernel $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{1}{\sigma^2}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\right)$ and an MDS-based approach for finding the preimage (see [20] for details).

Figure 2 demonstrates the result of projecting a cloud of random points onto the intersection of two non-linear smooth surfaces in $\mathbb{R}^3$. For each projected point, the found solution lies on or close to the real intersection curve, and is also close to the initialization point $\boldsymbol{y}^{(0)}$ as desired.

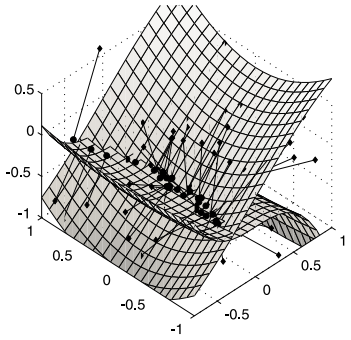
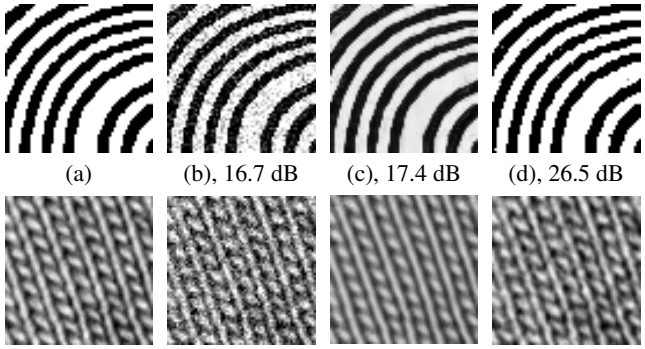


Fig. 2. Result of projecting a cloud of random points onto the intersection of two surfaces in $\mathbb{R}^3$. Notice how the projected points trace the intersection curve while remaining close to the initial points.

To assess the effectiveness of the proposed algorithm for the regularization of inverse problems using an underlying image patch model, we consider denoising of a synthetic high contrast image as well as a natural texture pattern using the procedure described in Section 4, with training sets of $p \times q = 5 \times 5$ patches, used to simultaneously denoise one pixel of $P \times Q = 9 \times 9$ image regions corrupted by additive zero-mean Gaussian noise. The results are shown in Figure 3.



(a) (b), 16.7 dB (c), 17.4 dB (d), 26.5 dB

(e) (f), 16.9 dB (g), 21.6 dB (h), 22.0 dB

Fig. 3. Results of image denoising: original images (a, e), noisy images (b, f), denoised using NL-means (c, g), denoised using proposed method (d, h). Numbers represent corresponding PSNR. Our algorithm preserves sharp high contrast edges of smooth curves as well as details of texture pattern; notice their blurring by NL-means.

Notice that the local manifold patch model has retained sharp contrast edges without blurring them and has also reconstructed the pattern details with good quality. To quantify the denoising performance we use peak signal-to-noise ratio (PSNR), defined as $PSNR = 10 \log \frac{\max I_{i,j}^2}{\frac{1}{N}\sum\left(I_{i,j} - J_{i,j}\right)^2}$, where I and J are the original and denoised N-pixel images respectively. For comparison, our method beats the near state-of-the-art non-local means denoising algorithm, which is based on weighted averaging of similar image patches [1]. Moreover, the running times of our algorithm are on the order of 5-10 seconds for an entire 64×64 image, which is quite fast and also comparable to the NL-means algorithm. In fact, given a training set of patches, subject-independent terms that define coefficients $\boldsymbol{\gamma}_m$ in Eq. 10 (mainly the matrices $\boldsymbol{M}$, $\boldsymbol{H}$ and $\boldsymbol{g}$) can be precomputed beforehand and used for processing of any image of a particular class, which makes the implementation even faster.

6. CONCLUSION

In this paper we proposed a new approach to regularizing inverse problems in image processing based on the model of overlapping patches each drawn from a manifold, resulting in an intersecting manifolds model for the entire image. This representation preserves the local features of images, such as sharp edges of smooth curves or details of textures. The kernel trick was used to describe presumably non-linear manifolds as linear subspaces in higher-dimensional feature space and to find their intersection with a simple iterative projection algorithm.

Moreover, as a proof of concept of our approach, a computationally efficient method of image denoising based on the derived algorithm, which shows results slightly better than those obtained with the state-of-the-art approaches was described. However, we want to emphasize that the capabilities of this model are not limited to denoising, but can also be extended to the regularization of other inverse problems. For example, simply imposing an additional set of linear constraints at each step of iterative projections makes this setting suitable for broad variety of applications, such as solving compressive sensing reconstruction, superresolution, or inpainting problems. We hope to investigate the impact of our computationally efficient patch-based image processing strategy on these problems in future works.

7. REFERENCES

[1] A. Buades, B. Coll, and J. M. Morel, "A Review of Image Denoising Algorithms, with a New One," *Multiscale Modeling & Simulation*, vol. 4, no. 2, pp. 490–530, 2005.

[2] Guoshen Yu, Guillermo Sapiro, and Stéphane Mallat, "Solving inverse problems with piecewise linear estimators: From gaussian mixture models to structured sparsity," *IEEE Transactions on Image Processing*, vol. 21, no. 5, pp. 2481–2499, 2012.

[3] Daniel Zoran and Yair Weiss, "From learning models of natural image patches to whole image restoration," in *Proceedings of the 2011 International Conference on Computer Vision*, Washington, DC, USA, 2011, ICCV '11, pp. 479–486, IEEE Computer Society.

[4] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman, "Patchmatch: a randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 24:1–24:11, July 2009.

[5] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani, "Summarizing visual data using bidirectional similarity," in *CVPR 2008. IEEE Conference on Computer Vision and Pattern Recognition, 2008.*, June 2008, pp. 1–8.

[6] Ann B. Lee, Kim S. Pedersen, and David Mumford, "The Nonlinear Statistics of High-Contrast Patches in Natural Images," *Int. J. Comput. Vision*, vol. 54, no. 1-3, pp. 83–103, Aug. 2003.

[7] Gunnar Carlsson, Tigran Ishkhanov, Vin Silva, and Afra Zomorodian, "On the local behavior of spaces of natural images," *Int. J. Comput. Vision*, vol. 76, no. 1, pp. 1–12, Jan. 2008.

[8] Minhua Chen, J. Silva, J. Paisley, Chunping Wang, D. Dunson, and L. Carin, "Compressive sensing on manifolds using a nonparametric mixture of factor analyzers: Algorithm and performance bounds," *Signal Processing, IEEE Transactions on*, vol. 58, no. 12, pp. 6140 –6155, Dec. 2010.

[9] G. Peyré, "Manifold models for signals and images," *Computer Vision and Image Understanding*, vol. 113, no. 2, pp. 249–260, Feb. 2009.

[10] Bernhard Schölkopf and Alexander J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, Cambridge, MA, USA, 2001.

[11] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998.

[12] Mikhail Belkin and Partha Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, June 2003.

[13] J. B. Tenenbaum, V. Silva, and J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[14] Jihun Ham, Daniel D. Lee, Sebastian Mika, and Bernhard Schölkopf, "A kernel view of the dimensionality reduction of manifolds," in *Proceedings of the twenty-first international conference on machine learning*, New York, NY, USA, 2004, ICML'04, pp. 47–54, ACM.

[15] Sebastian Mika, Bernhard Schölkopf, Alex Smola, Klaus-Robert Müller, Matthias Scholz, and Gunnar Rätsch, "Kernel PCA and De-Noising in Feature Spaces," in *Proceedings of the 1998 conference on advances in neural information processing systems II*, Cambridge, MA, USA, 1999, pp. 536–542, MIT Press.

[16] Jie Ni, P. Turaga, V.M. Patel, and R. Chellappa, "Example-Driven Manifold Priors for Image Deconvolution," Nov. 2011, vol. 20, pp. 3086–3096.

[17] H.J. Trussel and M.R. Civanlar, "The Landweber iteration and projection onto convex sets," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 33, pp. 1632–1634, 1985.

[18] O. Strand, "Theory and methods related to the singular-function expansion and landweber's iteration for integral equations of the first kind," *SIAM Journal on Numerical Analysis*, vol. 11, no. 4, pp. 798–825, 1974.

[19] Paul Honeine and Cédric Richard, "A Closed-form Solution for the Pre-image Problem in Kernel-based Machines," *J. Signal Process. Syst.*, vol. 65, no. 3, pp. 289–299, Dec. 2011.

[20] J.T.-Y. Kwok and I.W.-H. Tsang, "The Pre-Image Problem in Kernel Methods," *Neural Networks, IEEE Transactions on*, vol. 15, no. 6, pp. 1517–1525, Nov. 2004.

[21] M. Ouimet and Y. Bengio, "Greedy spectral embedding," in *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*. Omni Press Madison, WI, Jan. 2005, pp. 253–260.

[22] François G. Meyer and Xilin Shen, "Perturbation of the eigenvectors of the graph laplacian: Application to image denoising," *CoRR*, vol. abs/1202.6666, 2012.

[23] Michael Elad and Michal Aharon, "Image denoising via learned dictionaries and sparse representation," in *In CVPR*, 2006, pp. 17–22.