Joint Data Alignment Up To (Lossy) Transformations

Andrea Vedaldi Gregorio Guidi Stefano Soatto
Computer Science Department
University of California at Los Angeles, 90095, USA
{vedaldi,soatto}@cs.ucla.edu
http://vision.ucla.edu/

Abstract

Joint data alignment is often regarded as a data simplification process. This idea is powerful and general, but raises two delicate issues. First, one must make sure that the useful information about the data is preserved by the alignment process. This is especially important when data are affected by non-invertible transformations, such as those originating from continuous domain deformations in a discrete image lattice. We propose a formulation that explicitly avoids this pitfall. Second, one must choose an appropriate measure of data complexity. We show that standard concepts such as entropy might not be optimal for the task, and we propose alternative measures that reflect the regularity of the codebook space. We also propose a novel and efficient algorithm that allows joint alignment of a large number of samples (tens of thousands of image patches), and does not rely on the assumption that pixels are independent. This is done for the case where the data is postulated to live in an affine subspaces of the embedding space of the raw data. We apply our scheme to learn sparse bases for natural images that discount domain deformations and hence significantly decrease the complexity of codebooks while maintaining the same generative power.

1. Introduction

Alignment is a preprocessing element of many decision as well as compression procedures involving complex data. It serves to remove nuisance transformations in the data that are either irrelevant to the decision, or that can be represented explicitly in a generative model. Usually alignment is performed with respect to group (invertible) transformations as a pre-processing step. For instance, the image range (values) can be normalized for affine transformations (contrast and scaling) to gain insensitivity to illumination changes in image classification, while the image domain can be normalized with respect to transformations (e.g. translations or affine warps) that are applied directly by the compression algorithm (e.g. in MPEG). Here we study in particular the problem of *joint data alignment*, where the goal is to align simultaneously a large collection of data.

A popular approach to joint alignment is to transform the data in order to simplify their ensemble. In doing so, however, one may remove from the data not only irrelevant variability, but also useful information. Take the example of a scaling of the image domain. While diffeomorphic domain deformations, and in particular scalings, form a group in the continuum, invertibility is lost once we consider the discrete nature of digital images: The cascade of lattice-interpolation, domain deformation and resampling is in general *not* invertible. This problem is present in any image alignment problem, and has been sometimes neglected in the literature, where the algorithms are often illustrated on simple transformations such as translation by integer pixel values [15]. We explicitly address this issue and formulate the problem of joint data alignment in the presence of non-information-preserving transformations.

A second problem with this view of joint alignment is the choice of an appropriate measure of complexity of the data ensemble. It is tempting to use off-the-shelf measures such as entropy and mutual information, and to regard joint alignment as a compression problem (Sect. 1.1 and [9, 15]). However, complexity in Shannon's sense (as captured for instance by vector quantization (VQ)) essentially reflects the number of prototypes, or *codewords*, required to represent the data with a given accuracy. The structure of the codewords themselves is disregarded. So, while multiple data may be aligned to a given codeword, and hence one another, there is *a-priori* no reason for codewords to be globally aligned.

We propose a novel approach that measures complexity relative to the regularity of the space where the codebook lives, hence derives a codebook that is optimal relative to the postulated structure of the data space. We also propose a new cost functional for alignment, and point out the re-

lationship to VQ, image congealing [9] (IC), transformed component analysis [5] (TCA), as well as other alignment schemes recently proposed in the literature and classical rate-distortion theory. We also propose a novel and efficient algorithm that allows joint alignment of a large number of samples (tens of thousands of image patches), and does not rely on the assumption – implicit in many other approaches – that pixels are independent. This is done for the case where the data is postulated to live in affine subspace of the embedding space of the raw data (Sect. 1.2). The fact that we work with real-valued (as opposed to discrete) data enables us to use gradient-based optimization that is fast and efficient.

Finally, in Sect. 2 and 3 we show how our approach can be applied to learn "transformation-hypercolumns," sparse bases for natural images that discount domain deformations and hence significantly decrease the complexity of codebooks for natural images while maintaining the same generative power.

1.1. Joint alignment as compression

In this section we first review a few concepts from the theory of lossy compression and then show how they can be adapted to the task of joint data alignment.

Lossy compression, clustering, and joint alignment. Let $x \in \mathbb{R}^n$ be a continuous random variable (*datum*). In lossy compression we search for another (continuous or discrete) r.v. $y \in \mathbb{R}^n$ (*code*) with conditional density $p(y|x)$ that represents x concisely and accurately. Formally, we look for $p(y|x)$ that minimizes

$$E(y) = D(x, y) + \lambda \mathcal{C}(x, y). \tag{1}$$

The *distortion* $D(x, y)$ reflects the average error of the approximation, the *complexity* $\mathcal{C}(x, y)$ is the average number of symbols required to encode y, and the parameter $\lambda \geq 0$ trades off the two terms. This general idea has been pursued in various forms in the literature. In VQ [10], y is restricted to $K < \infty$ *codewords*, all encoded with the same number of symbols. Thus $C(x, y) \propto \log K$ and VQ trades off the number of codewords for the average distance of the code y to the datum x. Entropy-Constrained Vector Quantization (ECVQ) [3, 2] generalizes this idea and represents codewords with a variable number of symbols. The optimal variable-length code uses an average number symbols equal to the entropy $H(y)$, so that $\mathcal{C}(x, y) = H(y)$. There also exist relaxations of VQ and ECVQ based on deterministic annealing (DA) [13] that use a regularized complexity term which depends jointly on x and y. Rate-Distortion (RD) [14] is a further generalization to the problem of compressing long sequences of data. Shannon [14] reduced this problem to the one of encoding a single instance x by a code y (which in this case can be a continuous r.v.) and complexity measure $\mathcal{C}(x, y) = I(x, y)$. This complexity gives also the *rate* (i.e. the average number of symbols per component of the sequence).

Notice that VQ and ECVQ are useful not only for compression, but also for clustering. Next, we look for choices of the distortion and complexity measures that are useful for the problem of joint alignment. In particular, let G be a set of transformations $g : \mathbb{R}^m \to \mathbb{R}^n$. Our goal is to remove the effects of G on the datum x. We do so by searching for a code that represents x "up to the action of G" and is "as simple as possible". We do so by expanding upon our previous work [15].

Distortion for alignment. We choose a distortion measure for which x is represented by y up to the action of the transformations G. Starting from an arbitrary point-wise distortion measure $d_0 : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^+$, we simply consider the *expected invariant distortion*

$$D(x, y) = E[d(x, y)], \qquad d(x, y) = \inf_{g \in G} d_0(x, gy). \tag{2}$$

Notice that the transformations $g \in G$ need not have a special structure and, in particular, they are not requires to form a group.

Complexity for alignment. In VQ, ECVQ, and RD complexity essentially reflects the average number of symbols needed to index codewords. Unfortunately, this is not well suited for alignment because it is insensitive to the actual *values* of the codewords and cannot capture any of their "structural" regularities. In particular, there is no natural way of encouraging the mutual alignment of the codewords.

In order to do this, we restrict our attention to indexing mechanisms (and corresponding complexity measures) that are efficient precisely when y exhibits the desired regularity. For example, in IC the code y is a (random) binary image whose complexity $\mathcal{C}(x, y)$ is defined as the average entropy of its pixels, regarded as independent random variables. This gives the number of symbols required to index the codewords y if we *disregard the dependencies* among pixels. This description is efficient if the pixels of y are mostly constant (across different codewords y). Thus minimizing this complexity encourages the mutual alignment of the codewords.

The method we propose follows this philosophy: We use as complexity measure the (properly normalized) entropy of a Gaussian distribution fitting the code y. This corresponds to describing the code y by exploiting only to the linear dependencies between its components. Such a description is concise only if y spans a low dimensional subspace of $\mathbb{R}^n$ (Sect. 1.2).

Comparison to IC and TCA. As noted in [15], IC minimizes the complexity of the code y but does not directly enforce the requirement that y is still a good approximation to the original data. This fact requires restricting *a-priori*

the class of transformations G to a set that is guaranteed not too loose too much information (i.e., "not too lossy"). Our approach naturally trades off compression and reconstruction accuracy. When applied to continuous data, moreover, approaches such as IC and [15] may be affected by an additional degeneracy which we discuss in Sect. 1.2.

TCA [6, 5] is an approach to joint alignment different from IC and to the method we propose here. TCA aligns images by fitting a generative model that captures foreground and background appearance and transformation parameters. While TCA can handle non-invertible transformations as we do, differently from it (and from IC) we enforce a specific property of the aligned data (low dimensionality) and have an explicit approximation error (distortion). Moreover we do not need a prior distribution on the class of transformations G (we only need to know the class), nor to make assumptions on the distribution of the data (we fit a Gaussian model, but this is only used to extract the dimensionality of the data, regardless of their actual distribution). Finally, we can deal with relatively complicated transformations, while TCA on large data is practically limited to handle translations [8]. An advantage of TCA over the proposed method and IC is the ability to automatically segment the images into foreground (to which transformations are applied) and background (modeled as noise).

1.2. Structural complexity: The linear case

In this section we introduce a complexity term $\mathcal{C}(x, y)$ that characterizes the *linear dimensionality* (the number of dimensions of the linear subspace spanned by y) of the code $y \in \mathbb{R}^n$. We do so by constructing a description of y which is efficient when y spans a low-dimensional affine subspace of $\mathbb{R}^n$. To do this, first we approximate the density $p(y)$ of the code y with a Gaussian density $g(y)$, which captures the linear statistics of $p(y)$. Then, as if y had density $g(y)$, we use standard tools from rate-distortion theory to devise the optimal description of y and estimate its length (rate). The Gaussian density $g \in \mathcal{N}(\mu_g, \Sigma_g)$ which is closer to $p(y)$ in Kullback-Leibler (KL) divergence $\mathrm{kl}(p||g) = E_p[-\log g(y)] - h(p)$ is the one that matches the mean and the variance of y, i.e. $\mu_g = \mu_p = E_p[y]$, and $\Sigma_g = \Sigma_p = E_p[yy^\top] - \mu_p\mu_p^\top$. This Gaussian g yields an upper bound on the *rate* $R(\epsilon; p)$ (number of bits per symbol) required to describe y with some accuracy[1] ϵ: $R(\epsilon; p) \leq R(\epsilon; g)$. The rate-distortion function of a Gaussian source is known analytically [4], but in general its calculation requires computing the eigenvalues of Σ_g. If, however, we add to y a small Gaussian noise of variance ϵ^2 [16], the formula is simply $R(\epsilon; g) = \frac{1}{2}\log\det\frac{\epsilon^2 I + \Sigma_g}{\epsilon^2}$ and we obtain

$$\mathcal{C}(x, y) = \frac{1}{2}\log\det\left(I + \frac{\Sigma_p}{\epsilon^2}\right). \tag{3}$$

Degenerate solutions and normalization. It is easy to see that (3) decreases not only with the dimensionality of y, but also with its variance. Thus, if the transformations $g \in G$ enable reducing the variance of y without increasing the distortion of the reconstruction gy, then minimizing (3) yields a degenerate code (see also Fig. 1).

We remark that the same problem affects all similar formulations in which the code y is a continuous r.v. (for instance, it applies to some versions of IC [9]). The reason is that the mere fact of *measuring* the complexity of the continuous r.v. y requires approximating it, as reflected by the error term ϵ in (3). Thus ϵ is an *additional distortion* which is not accounted for in (2). While it is possible to map the error ϵ back to the distortion $d(x, gy)$ through the transformation $g \in G$, doing so is cumbersome. Fortunately, there is a simple shortcut that works well in practice. The idea is to tune ϵ adaptively as a fraction of the average variance $E[\|y - \mu\|^2] = \operatorname{tr}\Sigma_p$ of the code itself. This yields the corrected complexity term

$$\mathcal{C}'(x, y) = \frac{1}{2}\log\det\left(I + \frac{\Sigma_p}{\epsilon^2 \operatorname{tr}\Sigma_p}\right). \tag{4}$$

Notice that (3) can still be used in place of (4) when the particular problem prevents the degenerate solution to be found (for instance, (3) works well for aligning images).

A first example: Removing planar transformations. Consider $g : \mathbb{R}^2 \to \mathbb{R}^2$ acting on a set of K 2-D points $x_1, \ldots, x_K \in \mathbb{R}^2$ (Fig. 1). We compute the transformations $g_1, \ldots, g_K \in G$ and codes $y_1, \ldots, y_K \in \mathbb{R}^2$ by minimizing the cost function

$$\begin{aligned} &E(\{g_k, y_k\}) = D(x, y) + \lambda\mathcal{C}'(x, y) \\ &= \frac{1}{K}\sum_{k=1}^{K}\|x_k - g_k y_k\|^2 + \frac{\lambda}{2}\log\det\left(I + \frac{YY^\top}{\epsilon^2 \operatorname{tr} YY^\top}\right) \end{aligned} \tag{5}$$

where $Y = \begin{bmatrix} y_1 - \mu & \ldots & y_K - \mu \end{bmatrix}$ is the matrix of the centered codes and $\mu = \sum_{k=1}^{K} y_k/K$ is the sample average. This can be done by gradient descent. In Fig. 1 we use this method to remove rotations around the origin $G_1 = SE(2)$ and scalings $G_2 = \mathbb{R}$ respectively. The latter case clearly illustrates the importance of the normalization in (4), lest all points collapse to the origin.

2. Joint alignment of images

The main application of our method is the joint alignment of large collections of images. In this section we specialize (3) to solve this problem.

[1] The slack between $R(\epsilon; p)$ and $R(\epsilon; g)$ is irrelevant, as our goal is to characterize the *linear* dimensionality of the code y.

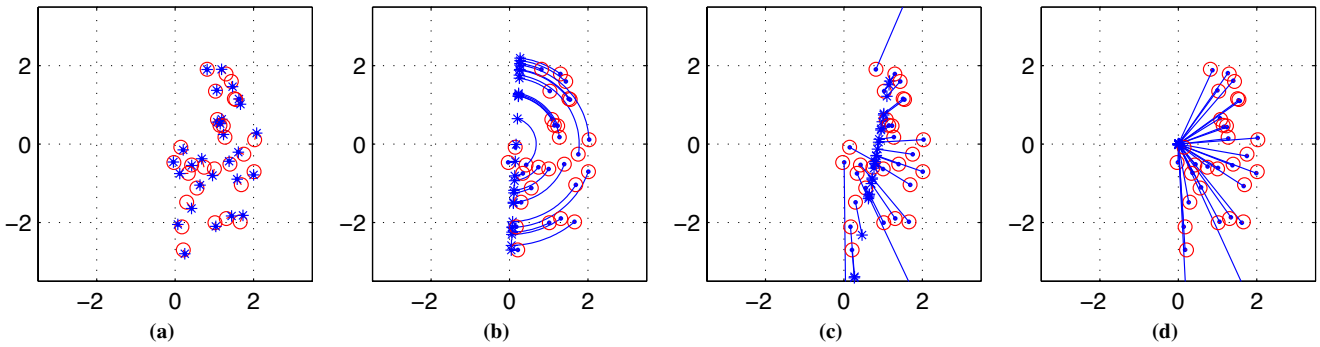


Figure 1: Aligning 2-D points (see text). (a) the data points $x_1, \dots, x_K$ (circles), the initial codes $y_1, \dots, y_K$ (stars — they are obtained by adding a small noise to the data) and reconstructions (dots — they coincide with the codes as initially $g_1 = \cdots = g_K = 1$); (b) removing the group G_1 of rotations from the data by mapping them to an affine subspace (line) — the curves show the trajectories mapping back the codes to the data; (c) removing the group G_2 of scalings from the data; (d) same as (c), except that the un-normalized complexity term (3) is used, which causes the solution to collapse on the origin.

We start by specifying the nature of the data, of the codes and of the transformations. The datum $x(u,v) \in \mathbb{R}$ is a (random) discrete image defined on the two dimensional lattice $\Omega = \{-r, -r+1, \dots, r\}^2$ where r is a non-negative integer. Similarly, the code $y(u,v)$ is a (random) discrete image defined on a lattice $\Omega' = \{-r', \dots, r'\}$. The image x will also be identified with a matrix in $\mathbb{R}^{(2r+1)\times(2r+1)}$ or a vector in $\mathbb{R}^{(2r+1)^2}$, and similarly for the image y. Our goal is to remove from the random image x transformations $g : \mathbb{R}^2 \to \mathbb{R}^2$ of the real plane. For simplicity, we consider only affine transformations g_α where $\alpha = \begin{bmatrix} A & T \end{bmatrix}$, $T \in \mathbb{R}^2$, $A \in GL(2)$ and $g_\alpha(u,v) = A^{-1} \cdot (u,v) - A^{-1} \cdot T$ (notice that α parametrizes the inverse warp, as this simplifies the equations below). Applying the transformation g_α to an image $y(u',v')$ yields the image $(g_\alpha y)(u,v) = y(g_\alpha^{-1}(u,v))$, where the value $y(u',v')$ at the fractional point $(u',v') = g_\alpha^{-1}(u,v)$ is obtained by extending y to the real plane by bilinear interpolation and zero padding.

Given samples $\{x_1, \dots, x_K\}$ of the random image x, the problem is then to find transformations $\{\alpha_1, \dots, \alpha_K\}$ and codes $\{y_1, \dots, y_K\}$ that minimize

$$E(\{\alpha_k, y_k\}) = \frac{1}{K} \sum_{k=1}^{K} \|x_k - g_{\alpha_k} y_k\|^2 + \lambda \frac{1}{2} \log\det\left(I + \frac{YY^\top}{K\epsilon^2}\right), \quad (6)$$

where $Y = \begin{bmatrix} y_1 - \mu & \dots & y_k - \mu \end{bmatrix}$ is the matrix of centered codes $y_k - \mu$, and $\mu = \sum_{k=1}^K y_k / K$ is the sample average. Notice that (6) is formally identical to (5), except that the complexity term (3) is used as the normalization is not necessary (because warps cannot decrease the variance of the code without affecting the reconstruction accuracy).

Dealing with image boundaries. According to (6), only the portion of the code y which is mapped back within the bounding box of the image x is actually constrained by the distortion term $\|x - g_\alpha y\|^2$ (see Fig. 2). The other portion of the code y is determined uniquely by minimizing the complexity $\mathcal{C}(x,y)$. In some cases this introduces a discontinuity in the estimated code y which makes the optimization of (6) tricky. This could be alleviated for example by delimiting the domain of x by a Gaussian window rather than by a bounding box. If, however, the image x can be extended beyond its bounding box in a natural way, then that information can be used to "fill the hole". We will get back to this issue in Sect. 3.

Experiments. We explore the effect of minimizing the cost functional (6) on the NIST handwritten digits dataset. A simple gradient descent method was used to find the optimal set of codewords and transformation parameters $\{y_k, \alpha_k\}$. For each digit, a set of 500 samples was extracted and aligned. The result is shown in Fig. 3 for different values of the trade-off parameter λ. Note that increasing λ the structure of the codewords converges to a low-dimensional space, eventually collapsing to a zero-dimensional space (a single template). Once the algorithm has found the optimal codeword and transformation g_α for each sample, we can apply the reverse transformations g_α^{-1} on the original digits to obtain the aligned dataset, as shown in Fig. 3-(c,d). Figure 4 shows the mean of all the digits aligned in this way, compared to the mean before alignment. The result is qualitatively similar to IC.

3. An efficient variant for decimated affine transformations

In this section we derive a variant of the model (6) which is computationally more attractive. The key idea is that, in-

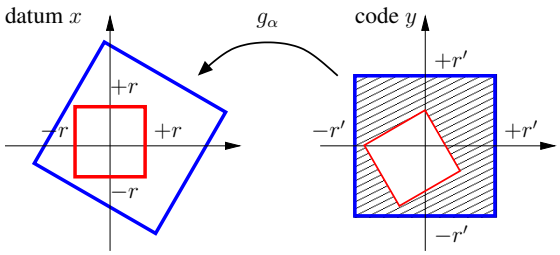
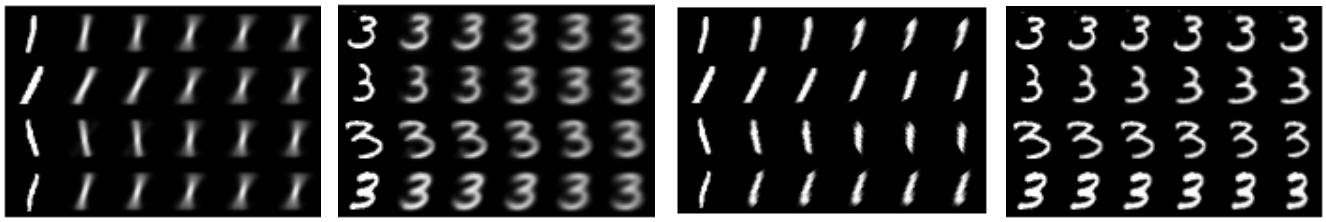


Figure 2: Boundaries: the code y is mapped back to the datum x by a transformation g_α. The portion of the code which is clipped by this operation (dashed area) needs not match x. If, however, x is extracted from a larger image or can otherwise be extended to the real plane, then the context of x can be used to fill the dashed area.



(a) codewords **(b)** codewords **(c)** aligned **(d)** aligned

Figure 3: Aligning NIST digits: (a), (b) show examples of original digits (first column) and of the codewords found by minimizing (6) for different values of λ. From second column to last: $\lambda = 1.5, 2, 2.5, 3, 3.5$. (c), (d) show the aligned versions of the same digits ($g_\alpha^{-1}x$) obtained by backprojection according to the optimal transformation.

stead of explicitly estimating the codes y_k, one could simply let $y_k = g_\alpha^{-1}x_k$ and avoid estimating the codes altogether. Unfortunately doing so requires in general to extend the image x beyond its bounding box (see Fig. 2 and Sect. 2), so this method can be used only if there is a reasonable way of doing so. For instance, in Fig. 5 small patches x are naturally extended by their context in the larger image and in Fig. 6 the hand-written digits are naturally extended by zero-padding.

By letting $y_k = g_\alpha^{-1}x_k$ the distortion term becomes $\|x - g_\alpha y\|^2 = \|x - g_\alpha(g_\alpha^{-1}x)\|^2$, which in general is *not* identically zero since the action of g_α on a discrete image is not necessarily invertible. In particular, $y = g_\alpha^{-1}x$ could be used to decimate the data by mapping the data x to a constant code $g_\alpha^{-1}x$ which in turn would trivially decrease the complexity $\mathcal{C}(x, y)$. So the term $\|x - g_\alpha(g_\alpha^{-1}x)\|^2$ forces the code $y = g_\alpha^{-1}x$ to preserve information about the datum x.

Notice that IC uses implicit codes too [9]. However, in place of the distortion term $\|x - g_\alpha(g_\alpha^{-1}x)\|^2$, IC simply penalizes transformations g_α that differ from the identity. This method has the advantage of speed and simplicity. Motivated by this observation, we experimented with a few surrogates of the distortion term and found that the simple function $\beta(x)/|\det(A)|$ approximates well $\|x - g_\alpha(g_\alpha^{-1}x)\|^2$. Here $\beta(x)$ is a constant which depends only on the datum x and can be estimated easily during preprocessing. Of course, this approximation is valid as long as the bounding box of the image x is mapped within the bounding box of the image y (Fig. 2), which can be enforced as a set of sixteen linear constraints $M\alpha + b \preceq 0$. It is convenient to incorporate these additional constraints into the energy function as a logarithmic barrier [1], yielding to the formulation

$$E(\{\alpha_k\}) = \frac{1}{K}\sum_{k=1}^{K}\left(\frac{\beta_k}{\det A_k} - \frac{1}{\gamma}\sum_{l=1}^{16}\log(-e_l^\top(M\alpha_k + b))\right) + \lambda\frac{1}{2}\log\det\left(I + \frac{YY^\top}{K\epsilon^2}\right), \quad (7)$$

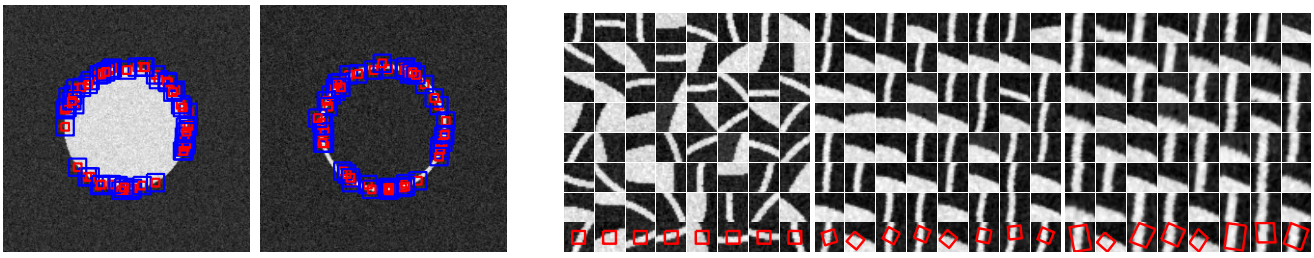
where $\alpha_k = \begin{bmatrix} A_k & T_k \end{bmatrix}$, $Y = [g_{\alpha_1}^{-1}x_1 - \mu,\ \ldots,\ g_{\alpha_K}^{-1}x_K - \mu]$ is the matrix of the centered implicit codes and γ is the slope of the logarithmic barrier (we use a large value of γ so that the barrier has an effect only at the boundaries of the feasible region).

Optimization. We optimize (7) one image at a time, looping over the entire dataset $x_1, \ldots, x_K$ several times. By doing this we can derive an efficient update rule for the transformations α_k. We start by noting that in (7) the only term that couples the different data is the entropic term through the covariance $C = I + YY^\top/K\epsilon^2$. Now fix the attention on a particular code y_k. As we vary y_k while keeping the other variables fixed, the matrix C becomes

$$C = C - \frac{(y_k - \mu)(y_k - \mu)^\top}{K\epsilon^2} + \frac{(y_k - \mu)(y_k - \mu)^\top}{K\epsilon^2} = \tilde{C} + \frac{(y_k - \mu)(y_k - \mu)^\top}{K\epsilon^2}. \quad (8)$$

We can expand the entropic term to second-order around

Figure 4: Aligning NIST digits: Per-digit average of the original data (above) and of the aligned data (below). It can be seen that the average appears much sharper after the alignment process despite only affine transformations being removed.



(a) context **(b)** original **(c)** direct **(d)** gradient

Figure 5: Aligning bars and wedges by the efficient formulation of Sect. 3: (a) two images from which a number of patches are sampled (in red we show the actual patch, and in blue the context used to complete the code $y = g_\alpha^{-1} x$); (b) a few such patches; (c) alignment based on direct search of rotation and translation; (d) refinement based on gradient descent on the full six parameters of the affine transformation. Note that in (c), (d) all bars are aligned and so are edges; the algorithm found two "codewords" to be sufficient to represent the data with prescribed accuracy.

$y_k = \mu$, obtaining

$$\frac{1}{2}\log\det\left(\tilde{C} + \frac{(y_k-\mu)(y_k-\mu)^\top}{K\epsilon^2}\right) \approx \frac{1}{2}(y_k-\mu)^\top \frac{\tilde{C}^{-1}}{K\epsilon^2}(y_k-\mu) + \text{const.}$$

which is a good approximation if $\|y_k - \mu\|/\epsilon\sqrt{K}$ is small, i.e. when K is large. Moreover, for a large K we have $\tilde{C} \approx C$. Adding the other terms of (7) back, we get that, as long as only one image is changed and K is sufficiently large, (7) can be approximated well by

$$E(\alpha_k) \approx \frac{1}{K}\left(\frac{\beta_k}{\det A_k} - \frac{1}{\gamma}\sum_{l=1}^{16}\log(-e_l^\top(M\alpha_k + b))\right) + \frac{\lambda}{2\epsilon^2 K}(g_{\alpha_k}^{-1}x - \mu)C^{-1}(g_{\alpha_k}^{-1}x - \mu) + \text{const.} \quad (9)$$

which depends only on α_k. We use two algorithms to optimize (9). The first, dubbed *direct search*, simply tries a number of values of each parameter of the transformation α_k (this is basically the same strategy of IC). The second, dubbed *gradient search*, uses the efficient Gauss-Newton quadratic approximation of (9). In Fig. 6 we use the efficient formulation (9) and the two algorithms to align NIST digits, and we get alignment results analogous to the one obtained from the formulation (6).

4. Aligning natural image patches

Starting from [12], there has been an emerging interest in studying sparse representations of natural images. Results from [12] show that, when a collection of natural patches are projected onto a linear basis whose coefficients have sparse statistics, structures such as bars, wedges and dots emerge, which resemble receptive fields of the human brain cortical areas V1, V2. Formally, given a collection $Y = [y_1, \ldots, y_K]$ of such natural patches, the sparse decomposition could be obtained by minimizing

$$E(A,B) = \|Y - BA\|_F^2 + \eta\sum_{qk}\log(1 + a_{qk}^2), \quad \text{subject to } \|b_q\|^2 = \beta > 0 \text{ for } q = 1,\ldots,Q \quad (10)$$

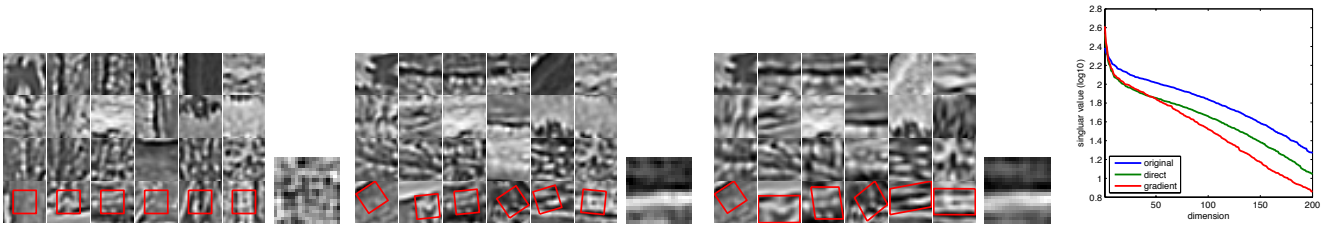
where N is the number of pixels of each image y_k, $B = [b_1 \quad \ldots \quad b_Q] \in \mathbb{R}^{N\times Q}$ is the matrix of basis elements b_q and $A = [a_{qk}] \in \mathbb{R}^{Q\times K}$ is the matrix of coefficients a_{qk} and η a parameter controlling the sparsity of the solution. For this procedure to work well, natural images must be appropriately whitened and contrast normalized [12]. The result of minimizing (10) over $Q = 128$ basis elements on a collection of 5000 natural image patches extracted in such a way is shown in Fig. 8-(a).

From Fig. 8-(a) and the analogous results obtained by many other authors, it is evident that many of the structures found are similar, differing only by geometric parameters such as position, orientation and scale. Recently it has been argued by [7, 11] that these systematic transformations could be estimated and removed, obtaining a more compact representations which would also be invariant to such kind of geometric distortions.

To this end [7, 11] extend the generative model (10) to

(a) original **(b)** direct **(c)** gradient **(d)** PCA

Figure 6: Aligning NIST digits: (a),(b) and (c) have been obtained as in Fig. 5; (d) shows the singular values of the three datasets (a),(b) and (c): notice the progressive reduction in the linear dimensionality of the data.



(a) original **(b)** direct **(c)** gradient **(d)** PCA

Figure 7: Aligning natural image patches: We use the same conventions of Fig. 5. The alignment results may not be evident from the patches alone, but the mean of the data (small images on the right) reveals the structure found by the algorithm. (d) PCA analysis of (a), (b) and (c) reveals the decreasing linear complexity.

account for geometric transformations and solve for basis, coefficients and geometric parameters (this is not dissimilar from [5], except for the sparsity prior). Unfortunately this results in a large computation which is unstable. Moreover, we explicitly address the problem of boundaries, which are an important factor even when dealing with simple transformation such as translations.

Since our alignment algorithm is capable of decreasing the dimensionality of the linear embedding spanned by the data (no matter whether its statistic is sparse or Gaussian) it may be appropriate as a pre-processing step to align the collection of natural image patches before the sparse dimensionality reduction (10). The result of such alignment is shown in Fig. 7 and Fig. 8-(b) illustrates the result of applying the very same algorithm of Fig. 8-(a) to the aligned data. Several observations can be made. First, a few dominant horizontal structures emerge, which evidently subsume many of the other structures found in Fig. 8-(a) at different orientations and translations. Second, such structures are found multiple times, with exactly the same appearance *and* position and orientation. To quantify this phenomenon, in Fig. 8-(a) we collapse similar basis elements until the reconstruction error in (10) increases less than 1% (we do this by iteratively collapsing the pair of most similar basis elements). This shows quantitatively that indeed several of the basis elements are redundant copies, created by the local optimization procedure used to minimize (10). Third, a number of relatively unstructured basis elements remain, which may indicate that the variety of strong structures has been significantly reduced by aligning the data.

5. Conclusions

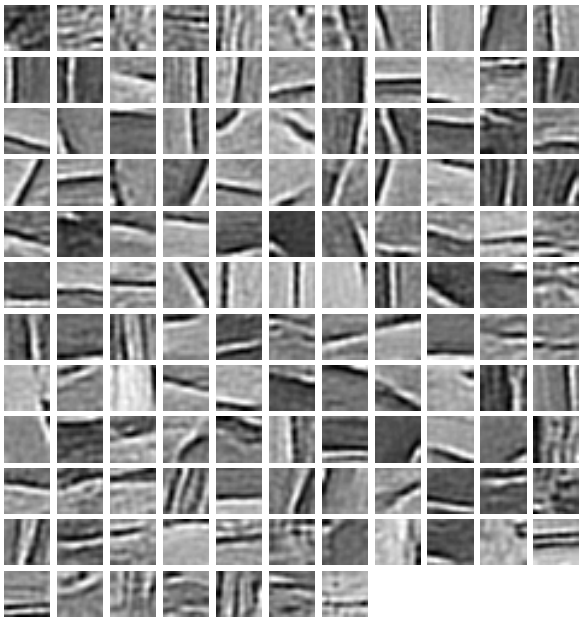
We have presented a novel approach to perform alignment with respect to transformations of the data that are not invertible. We show that a measure of complexity can be defined that is tailored to the postulated structure of the space where the codebook lives, and in particular we explore the case of affine subspaces of the embedding space of the raw data. We have presented efficient alignment algorithms that allow aligning large collections of handwritten digits and natural image patches, and more general real valued data.

Acknowledgments. Research supported by ONR 67F-1080868 and AFOSR FA9550-06-1-0138.

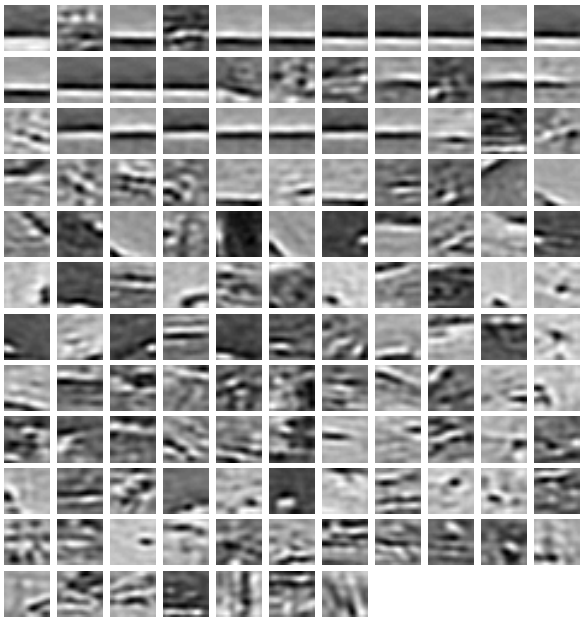
References

[1] S. Boyd and L. Vanderberghe. *Convex Optimization*. Cambridge University Press, 2004.

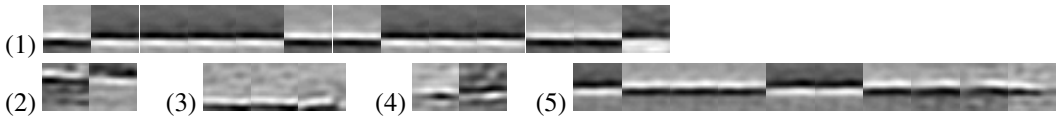
[2] J. Buhmann and H. Kühnel. Vector quantization with complexity costs. *IEEE Trans. on Information Theory*, 39, 1993.



(a) original



(b) aligned



(c) clusters

Figure 8: Sparse decomposition of natural image patches: (a) decomposition of 5000 image patches (basis elements are ordered by decreasing Kurtosis of their coefficients). (b) decomposition of the same 5000 patches after alignment. (c) the duplicate basis function detected in (b) form five clusters.

[3] P. A. Chou, T. Lookabaugh, and R. M. Gray. Entropy-constrained vector quantization. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 37(1), 1989.

[4] T. M. Cover and J. A. Thomson. *Elements of Information Theory*. Wiley, 2006.

[5] B. J. Frey and N. Jojic. Transformed component analysis: Joint estimation of spatial transformations and image components. In *Proc. ICCV*, 1999.

[6] B. J. Frey and N. Jojic. Transformation-invariant clustering using the EM algorithm. *PAMI*, 25(1), 2003.

[7] D. B. Grimes and R. P. N. Rao. Bilinear sparse coding for invariant vision. *Neural Computation*, 17, 2005.

[8] A. Kannan, N. Jojic, and B. J. Frey. Fast transformation-invariant component analysis. *IJCV*, 77(1–3), 2007.

[9] E. G. Learned-Miller. Data driven image models through continuous joint alignment. *PAMI*, 28(2), 2006.

[10] Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Trans. on Communications*, 1980.

[11] B. A. Olshausen, C. Cadieu, J. Culpepper, and D. K. Warland. Bilinear models of natural images. In *Proc. SPIE*, volume 6492, 2007.

[12] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 1996.

[13] K. Rose and D. Miller. A deterministic annealing algorithm for entropy-constrained vector quantizer design. In *Proc. of IEEE Conf. on Signals, Systems and Computers*, 1993.

[14] C. Shannon. A mathematical theory of communications. *The Bell System Tech. Journal*, 27, 1948.

[15] A. Vedaldi and S. Soatto. A complexity-distortion approach to joint pattern alignment. *NIPS*, 2007.

[16] A. Yang, J. Wright, S. S. Sastry, and Y. Ma. Unsupervised segmentatiton of natural images via lossy data compression. *CVIU (submitted)*, 2007.