Journal of Information Processing Vol.24 No.3 565–572 (May 2016)

[DOI: 10.2197/ipsjjip.24.565]

Regular Paper

Dictionary learning by Normalized Bilateral Projection

Taro Tezuka[1,a)]

Received: September 27, 2015, Accepted: January 7, 2016

Abstract: Dictionary learning is an unsupervised learning task that finds a set of template vectors that expresses input signals by sparse linear combinations. There are currently several methods for dictionary learning, for example K-SVD and MOD. In this paper, a new dictionary learning method, namely K-normalized bilateral projections (K-NBP), is proposed, which uses faster low rank approximation. Experiments showed that the method was fast and when the number of iterations was limited, it outperforms K-SVD. This indicated that the method was particularly suited to large data sets with high dimension, where each iteration takes a long time. K-NBP was applied to an image reconstruction task where images corrupted by noise were recovered using a dictionary learned from other images.

Keywords: dictionary learning, sparse coding, bilateral projections, image reconstruction

1. Introduction

Sparse coding aims to express the input vector as a linear combination of a small number of template vectors [2], [9]. It is now intensively being studied in various fields in computer science, including signal processing, machine learning, and pattern recognition. In fact, sparse coding generalizes some of the most widely used methods in data processing. For example, in Fourier expansion and wavelet analysis, sinusoidal functions and wavelets can be considered as template vectors, so when the observed data is approximated using few sinusoidal functions or wavelets with relatively large coefficients, they are actually being sparsely coded [14], [21]. Finding template vectors is in fact an effective way for data mining and data aggregation.

Sparse coding mathematically corresponds to finding sparse vector x which fulfills an underdetermined system of linear equations $y = Dx$, where y is the input vector. A sparse vector is a vector having mostly zero entries and only a few non-zero entries. Matrix D is usually called a *dictionary*, since each column of D represents a template vector. It is also sometimes called a codebook. D must sparsify not just one input vector y but many others as well. Therefore a matrix Y, whose column vectors are input vectors, and an equation $Y = DX$ were used. The goal is to find a matrix X that is sparse. Hereafter Y will be called the input matrix, and X the source matrix.

Whether input matrix Y can be transformed into sparse X or not depends on dictionary D. The goodness of D is problem specific, i.e., it depends on Y. In dictionary learning, D is optimized using Y. Thus far, there have been several proposals for dictionary learning, most notably K-SVD (K-singular value decomposition) [1]. K-SVD, however, requires long computation time when the dimension of the input vector increases, due to singular value decomposition (SVD) that is computed in each step of the main iteration in K-SVD. To overcome the problem, this paper proposes K-normalized bilateral projections (K-NBP), which replaces SVD in K-SVD with bilateral random projection of normalized vectors. The method was described earlier by the author in Ref. [22]. This paper improved the method by introducing statistical standardization, and also applied it to an image reconstruction task.

The rest of the paper is organized as follows. Section 2 discusses the background to this research. Section 3 proposes the method, and Section 4 presents the results obtained from experiments. Section 5 discusses related work and Section 6 concludes the paper.

2. Background

This section provides the background to the method proposed in this paper. First, K-SVD, which is one of the most commonly used methods of dictionary learning, will be described. Then bilateral random projections, which is a newly proposed method for low rank approximation, will be introduced.

2.1 Dictionary Learning

In what follows, $A_{i:}$ denotes the i-th row vector of matrix A and $A_{:j}$ denotes the j-th column vector of matrix A. Given input matrix $Y \in R^{m\times n}$, dictionary learning seeks a dictionary $D \in R^{m\times k}$ and a sparse matrix $X \in R^{k\times n}$ which fulfills $Y = DX$. The columns of D are called *atoms*. They are usually normalized using ℓ_2-norm, so that $\|D_{:h}\| = 1$ for all h.

One of the most widely used dictionary learning methods, K-SVD, is outlined in Algorithm 1. It starts with a randomly generated dictionary, and iteratively applies two stages, i.e., (1) sparse coding and (2) a dictionary update, for a set number of iterations. Dictionary D stays fixed in the sparse coding stage, and the optimal X is sought for. Any method of sparse coding can be used, e.g., orthogonal matching pursuit (OMP) or basis pursuit (BP) [8]. Both D and X are revised in the dictionary update stage. The main goal is to minimize the norm of the error matrix, $E = Y - DX$. Its Frobenius norm, $\|E\|_F$, defined as the square root

1 University of Tsukuba, Tsukuba, Ibaraki 305–0006, Japan

a) tezuka@slis.tsukuba.ac.jp

Algorithm 1 K-SVD

```
Input: Input matrix $Y \in R^{m\times n}$
Output: Estimated dictionary $D \in R^{m\times k}$ and estimated source matrix $X \in R^{k\times n}$
  Set $D$ randomly
  for $t = 1$ to $\tau$
    Sparse coding stage:
      Get sparse $X$ that fulfills $Y = DX$
    Dictionary update stage:
      for $h = 1$ to $k$
        $E_h \longleftarrow Y - \sum_{j\neq h} D_{:j}X_{j:}$
        Obtain $\Omega_h$ using $X_{h:}$
        $\tilde{E}_h \longleftarrow E_h\Omega_h$
        By applying SVD to $\tilde{E}_h$, get $U_{:1}$, $V_{:1}$, and $\lambda_1$
        $D_{:h} \longleftarrow U_{:1}$
        Revise the non-zero entries of $X_{h:}$ by $\lambda_1 V_{:1}^T$
      end
  end
```

of the sum of the squares of its entries, is used in K-SVD.

$$\begin{aligned}\|E\|_F &= \|Y - DX\|_F \qquad (1)\\ &= \|Y - \sum_{j=1}^{k} D_{:j}X_{j:}\|_F\\ &= \left\|\left(Y - \sum_{j\neq h} D_{:j}X_{j:}\right) - D_{:h}X_{h:}\right\|_F\\ &= \|\left(Y - \sum_{j\neq h} D_{:j}X_{j:}\right) - D_{:h}X_{h:}\|_F\\ &= \|E_h - D_{:h}X_{h:}\|_F\end{aligned}$$

Here, E_h is defined as $Y - \sum_{j\neq h} D_{:j}X_{j:}$. This reduces the problem to find the rank-1 approximation to E_h for each h. When this approximation is conducted, $X_{h:}$ must stay sparse. In order to do this, a smaller matrix $\tilde{E}_h$ defined below is introduced, so that the zero entries of $X_{h:}$ are not revised. Set ω_h of integers is introduced for this purpose. ω_h is often called "support".

$$\omega_h = \left(q | 1 \leq q \leq n, X_{hq} \neq 0\right) \qquad (2)$$

ω_h lists the indices of the non-zero entries of $X_{h:}$. Using this, projection matrix Ω_h is defined as follows. Let $|\omega_h|$ indicate the number of elements in $|\omega_h|$. For $i = 1, \ldots, |\omega_h|$, the $(w_h(i), i)$-th entries of Ω_h are 1, and all other entries are 0. Finally, define $\tilde{E}_h = E_h\Omega_h$, and $\tilde{E}_h$ is then approximated by a product of a column vector and a row vector. In other words, $\tilde{E}_h$ is approximated by a rank 1 matrix. In K-SVD, singular value decomposition (SVD) is used for this low rank approximation. Let $\tilde{E}_h = U\Lambda V^T$ be SVD. Then, $E_h \simeq \lambda_1 U_{:1}V_{:1}^T$. Finally, $U_{:1}$ is substituted into $D_{:h}$, while $\lambda_1 V_{:1}^T$ is substituted into the non-zero entries of $X_{h:}$.

The use of SVD constrains K-SVD in terms of computation time. Since $\tilde{E}_h \in R^{m\times|\omega_h|}$ matrix, the complexity of SVD is $O(\min\{m'^2n', m'n'^2\})$. It means that the computation time rapidly increases as the dimensions of the input vectors, m, or the number of input vectors n, increases. The low rank approximation part of K-SVD is therefore replaced with a method that uses less computation time than SVD.

Algorithm 2 Bilateral random projections

```
Input: Matrix $M \in R^{m\times n}$
Output: Low rank approximation $L \in R^{m\times n}$
  Generate random matrices $A_1 \in R^{n\times r}$ and $A_2 \in R^{m\times r}$
  for $s = 1$ to $\rho$
    $W_1 \longleftarrow MA_1$
    $W_2 \longleftarrow M^TA_2$
    $A_1 \longleftarrow W_2$
    $A_2 \longleftarrow W_1$
  end
  $L \longleftarrow W_1(A_2^TW_1)^{-1}W_2^T$
```

2.2 Bilateral Random Projections

Bilateral random projections (BRP) is a method of low rank approximation that was recently proposed by Zhou and Tao [23], [24]. Their motivation was to implement a system that decomposed the input matrix into the sum of three matrices, namely a sparse matrix, a low rank matrix, and a noise matrix.

The low rank approximation of matrix $M \in R^{m\times n}$ in BRP is obtained using bilateral projections starting from random matrices. Let r be the required rank. The overall algorithm is indicated in Algorithm 2. First, two random matrices, $A_1 \in R^{n\times r}$ and $A_2 \in R^{m\times r}$, are generated. In practice, random values sampled from a Gaussian distribution can be assigned to each entry of the matrices. These matrices are used to iteratively project M to r-dimensional subspaces, by $W_1 = MA_1$ and $W_2 = M^TA_2$. Finally, low rank approximation is obtained by $L = W_1(A_2^TW_1)^{-1}W_2^T$. When $r = 1$, this is the power iteration for computing eigenvectors, since the left and right singular vectors of M correspond to eigenvectors of MM^T and M^TM. It can also be considered as a form of alternating projections [5], where A_1 and A_2 are alternatingly projected using M and M^T.

3. Method

In this paper, a new dictionary learning method, k-normalized bilateral projections (K-NBP), is proposed. The name comes from its principal feature that not only atoms $D_{:h}$ but also source row vectors $X_{h:}$ are normalized.

3.1 Update Rule

Instead of using computationally costly SVD, K-NBP conducts low rank approximation by bilateral random projections. Projected vectors are normalized and substituted to $D_{:h}$ and $X_{h:}$. For the normalization step to be justified, the ℓ_2-norm of source vector $X_{h:}$ must equal to 1. The assumption becomes approximately true when input matrix Y is normalized and standardized. Let $\overline{X}_{h:}$ be the standardized source vector, i.e., each of its entries follows a standard Gaussian distribution. The expected ℓ_2-norm of this vector is $\sqrt{n}$. By dividing $\overline{X}_{h:}$ by $\sqrt{n}$, a vector whose expected ℓ_2-norm is 1 can be obtained. Algorithm 3 carries out both standardization and normalization for the input matrix together.

3.2 Standardization and Normalization

In order for source vector X_h to have norm 1, input matrix Y is standardized and normalized in K-NBP before the main loop starts. First, the entries of Y are standardized using the sample

Algorithm 3 Standardization and normalization stage for K-NBP

```
Input: Input matrix $Y \in R^{m \times n}$
Output: Standardized and normalized input matrix $\tilde{Y} \in R^{m \times n}$
  Calculate mean $\mu$ and std.dev. $S$ of all entries of $Y$
  for $i = 1$ to $m$
    for $j = 1$ to $n$
      $\tilde{Y}_{i,j} \longleftarrow (Y_{i,j} - \mu)\alpha / S\sqrt{n}$
    end
  end
```

Algorithm 4 Dictionary update stage of K-NBP

```
Input: Standardized and normalized input matrix $\tilde{Y} \in R^{m \times n}$, estimated dictionary $D \in R^{m \times k}$, and estimated source matrix $X \in R^{k \times n}$
Output: Revised $D \in R^{m \times k}$ and $X \in R^{k \times n}$
  for $h = 1$ to $k$
    $E_h \longleftarrow \tilde{Y} - \sum_{j \neq h} D_{:j} X_{j:}$
    $\tilde{E}_h \longleftarrow E_h \Omega_h$
    Obtain $\Omega_h$ using $X_{h:}$
    Generate random vector $A_1 \in R^{|\omega_h|}$ and $A_2 \in R^m$
    $W_1 \longleftarrow \tilde{E}_h A_1$
    $W_2 \longleftarrow \tilde{E}_h^T A_2$
    for $p = 1$ to $\rho$
      $W_1 \longleftarrow \tilde{E}_h W_2$
      $W_2 \longleftarrow \tilde{E}_h^T W_1$
    end
    if $\|W_1\| \neq 0$
      $D_{:h} \longleftarrow W_1 / \|W_1\|$
    else
      $D_{:h} \longleftarrow 0$
    end
    if $\|W_2\| \neq 0$
      Revise non-zero entries of $X_{h:}$ by $W_2^T / \|W_2\|$
    end
  end
```

mean and sample standard deviation. Here, an approximation assuming all the entries were sampled from i.i.d. Gaussian distributions was used. A more sophisticated way of standardization must be carried out when this cannot be assumed. When the dimensions of the input matrix are large enough, however, standardization using the sample mean and sample standard deviation is assumed to be an acceptable approximation.

Input matrix Y is multiplied by a coefficient α in the preprocessing stage to cope with strong stochastic dependencies in observations. For an ideal case where the entries follow independent Gaussian distributions, $\alpha = 1$ can be used. The actual value of α is optimized using the given data set.

For normalization, each entry of input matrix Y is divided by $\sqrt{n}$, where n is the number of columns of Y. Since $\|Y\|_{\ell_2}$ scales by $\sqrt{n}$ with respect to n, it is divided by $\sqrt{n}$.

3.3 Dictionary Update

Algorithm 4 describes the dictionary update stage of K-NBP. As in bilateral random projections, W_1 and W_2 are obtained by projecting with $\tilde{E}_h$ and $\tilde{E}_h^T$, respectively. After repeating the projections for a set number of times ρ, $W_1/\|W_1\|$ is substituted into $D_{:h}$, and $W_2^T/\|W_2\|$ is substituted into the non-zero entries of $X_{h:}$. Note that the approximation of SVD proposed by Rubinstein et al. uses $W_2^T/\|W_1\|$ instead of $W_2^T/\|W_2\|$ [18]. Note also that this normalization is distinct from the normalization commonly used in the power iteration to obtain eigenvectors, since the proposed method normalizes the vector right before it is substituted into the source matrix, and is therefore part of the dictionary update stage but not of power iteration. When $\tilde{E}_h$ is a $m' \times n'$ matrix on average, the complexity of the whole dictionary update stage is $O(km'n'\rho)$.

Although K-NBP contains an internal loop where bilateral projections are repeated, the overall computation time for high dimensional signals is expected to be less than that K-SVD, since K-NBP does not contain costly computation of SVD, which is repeatedly conducted in K-SVD.

4. Evaluation

K-NBP and K-SVD were compared using both synthetic and real data. Parameters that had to be adjusted were τ, which is the number of iterations in the main loop and ρ, which is the number of iterations in the low rank approximation. The optimal values for these parameters were sought for in the experiments.

4.1 Synthetic Data

4.1.1 Experimental Setup

The following experiments were conducted based on the scheme used by Aharon et al. for evaluating K-SVD [1]. Input matrix Y is generated in the following way. First, dictionary $D \in R^{20 \times 50}$ is randomly generated, by sampling each entry from the uniform distribution over $[-1, 1]$, and then normalizing each column using the ℓ_2-norm, i.e., set all h to $\|D_{:h}\| = 1$. Sparse source matrix $X \in R^{50 \times 1500}$ is generated by randomly assigning 3 non-zero entries to each of its columns. The value is assigned by sampling from the standard Gaussian distribution, i.e., $N(0, 1)$ for each of these non-zero entries. This is different from the original setting by Aharon et al. where they used a uniform distribution. This was to make the standardization of the input matrix simpler and more reliable. K-NBP and K-SVD were compared by using this input matrix Y.

OMP was used for both K-NBP and K-SVD in the sparse coding stage. For K-NBP, each entry of random matrices A_1 and A_2 is sampled from the standard Gaussian distribution. Aharon et al. carried out experiments by setting the number of iterations in the main loop to 80, i.e., they used $\tau = 80$ [1]. Elad and Aharon proposed to iterate it to 180 times [8] in another work where K-SVD was applied to image processing. This indicates that the appropriate value of τ depends on the types of applications and the size of input vectors.

The method was evaluated based on the number of successfully estimated atoms [1]. This was defined as the number of cases that an estimated atom came closer to an atom than the threshold in the original dictionary. The level of closeness was measured using the inner product. Let $\hat{D}$ be the estimated dictionary and D be the original (correct) dictionary. When there is $D_{:h'}$ such that $|\langle \hat{D}_{:h}, D_{:h'} \rangle| \geq \theta$, atom $\hat{D}_{:h}$ was judged to be successfully estimated. Aharon et al. used $\theta = 0.99$, but here the case where $\theta = 0.95$ was also considered. α was set to 1 in this experiment.

4.1.2 Results

In the first experiment, the value of ρ (the number of itera-

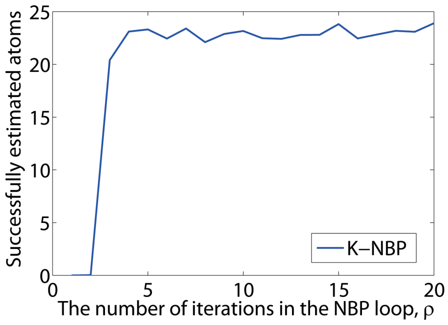

Fig. 1 Successfully estimated atoms for $\theta = 0.99$ as number of iterations in low rank approximation was changed.

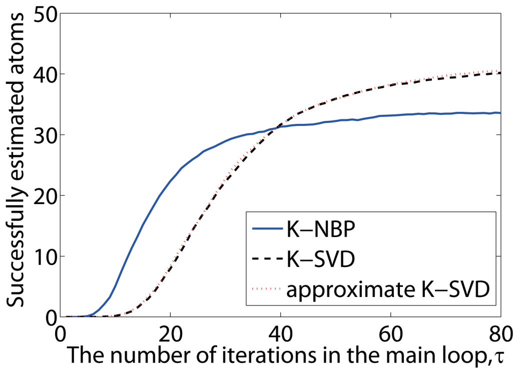

Fig. 2 Successfully estimated atoms for $\theta = 0.99$ as number of iterations in main loop was changed.

tions in low rank approximation) was changed to see how large it should be in order to obtain sufficiently accurate results. The number of iterations in the main loop was set to $\tau = 20$, while ρ was changed from 0 to 20. The threshold was set to $\theta = 0.99$. **Figure 1** plots the average number of successfully estimated atoms as ρ was changed. Each average is for 100 trials. The result indicates that the number of successfully estimated atoms has not changed much for $\rho \geq 5$.

In the second experiment, the value of ρ was set to 5 while τ, the number of iterations in the main loop, was changed from 0 to 80. The number of iterations in low rank approximation was set to $\rho = 5$. The threshold was set to $\theta = 0.99$. **Figure 2** plots the average number of successfully estimated atoms for 100 trials. The solid blue line is for K-NBP, and the dashed black line is for K-SVD. The red dotted line, which runs very close to K-SVD (dashed black line), plots the results for the approximate K-SVD proposed by Rubinstein et al. [18]. The number of iterations in the low rank approximation of approximate K-SVD (this value is denoted by υ) was set to 3, since they suggested that a single iteration performed satisfactorily. Note that the number of atoms in this experiment was set to 50. The results indicate that although K-NBP does not reach the level of accuracy obtained by K-SVD, it nevertheless reaches a certain level of accuracy much faster. This means that when either the sparse coding stage or the dictionary update stage takes a very long time, K-NBP has an advantage over K-SVD.

Figure 3 plots average performance when the threshold was set to $\theta = 0.95$. K-NBP converged to a value very close to that of K-SVD and the approximate K-SVD in this case, but was much quicker. Therefore, K-NBP can be considered to be a good choice

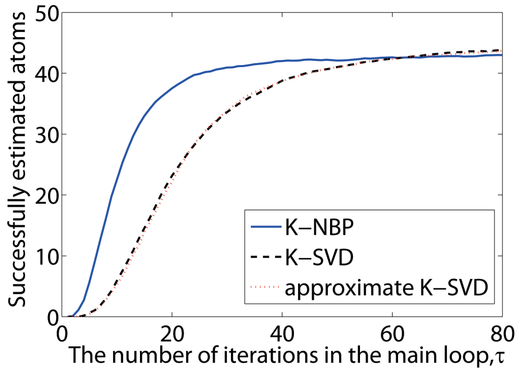

Fig. 3 Successfully estimated atoms for $\theta = 0.95$ as number of iterations in main loop was changed.

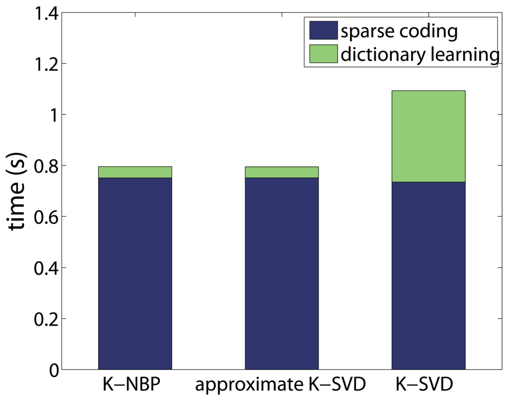

Fig. 4 Average computation time for different methods and stages.

Table 1 Average computation time for different methods and stages for a single step of the main loop (in seconds).

Stage	K-NBP	approx. K-SVD	K-SVD
Sparse coding	0.7510	0.7515	0.7350
Dictionary update	0.0447	0.0433	0.3583

when either only rough approximation is needed or it takes too much time to run K-SVD. K-NBP achieves 80 percent average recall after 26 iterations, while K-SVD and approximate K-SVD require 45 iterations. If 80 percent recall is the goal, K-NBP improves the efficiency by 1.7 compared to K-SVD.

Figure 4 and **Table 1** compare the computation times for the sparse coding stage and dictionary update stage for K-SVD, K-NBP, and the approximate K-SVD by Rubinstein et al. [18]. The time is the total number of k iterations in the dictionary update stage. The experiments were carried out using an Intel Core i7 2.90 GHz processor. The time indicated here is the CPU time for running MATLAB programs. For computation, Intel MKL Basic Linear Algebra Subroutines (BLAS) and Linear Algebra Package (LAPACK) libraries were used.

The number of trials was 100, with $\tau = 80$, $\rho = 5$, and $\upsilon = 3$. Therefore, the value was the average of $80 \times 100 = 8,000$ CPU time recordings. The blue part at the bottom of the figure indicates the average time for one execution of the sparse coding stage. The light green part at the top indicates the average time for one execution of the dictionary update stage. The results indicate that K-NBP was much faster than the original K-SVD, and was comparable to the approximate K-SVD. In addition, it shows that the sparse coding stage (OMP) took much longer than the dictionary update stage of any of the methods compared here. This feature was also reported by Rubinstein et al. [18], and suggests that if the computation time of the sparse coding stage cannot be reduced further, it is important to reduce the number of iterations in the main loop in order to reduce the total computation time. K-NBP is more suited than K-SVD or the approximate K-SVD to achieve this purpose. It should also be noted that the computation times for OMP and SVD rapidly increases as functions of the data size. It means that K-NBP is more effective with large amounts of data.

4.2 Real Data

K-NBP was compared to K-SVD using real data, specifically images. The experiment was designed following the scheme used by Aharon et al. when evaluating K-SVD [1]. Fifteen images of cameras contained in Caltech-256 [12] were broken down into 11,000 local image blocks, each of which had 8×8 pixels. These training data were of the same size as those used in the evaluation of K-SVD [1]. Each block B corresponded to a column of input matrix Y.

Dictionary $D \in R^{64\times441}$, consisting of 441 atoms, was trained for input matrix $Y \in R^{64\times11000}$ using K-NBP, K-SVD, and the approximate K-SVD. The first column (atom) of D in all three methods was always set to be a normalized constant vector, i.e., having the same value for all components. The maximum number of non-zero coefficients was set to 10 for the OMP used with dictionary learning. The maximum tolerated error in OMP was $\|(0.02)1\|$, where 1 indicates a vector with all components equal to 1. This means that an error of ±0.02 was tolerated for each pixel. As OMP was conducted within the subspace of non-corrupted pixels, each atom was normalized in this space.

The trained dictionaries were tested in the following way. An image that was not contained in the training set was used as a test image. These images were corrupted and then the trained dictionaries were evaluated based on how well the original image could be recovered. The test image consisted of 1,369 blocks in the experiment. Randomly selected pixels were corrupted for each block, for ratio r of all the pixels. r was set to 0.5 in the experiment. This means that half of all the pixels were corrupted.

The corrupted pixels were reconstructed with dictionary D. The non-corrupted pixels remained as-is, i.e., they were not revised. Note that the system recognized which pixels were corrupted in this experiment. Such information is not always available in many applications, but this is the evaluation scheme used in Ref. [1], and for the sake of comparing the proposed method with K-SVD, this assumption was considered to be valid.

The system conducts OMP for each block B in the subspace spanned by the non-corrupted pixels in B, and sparse vector x_B is obtained. The non-zero components of this vector correspond to a small set of atoms that are used to approximate block B. The maximum number of non-zero components of x_B is set to 10 in OMP. Finally, the corrupted pixels of B are revised by the corresponding components of Dx_B. The resulting reconstructed block is expressed by $\tilde{B}$.

As in [1], RMSE (root mean squared error) was used for measuring the reconstruction error. RMSE is defined by

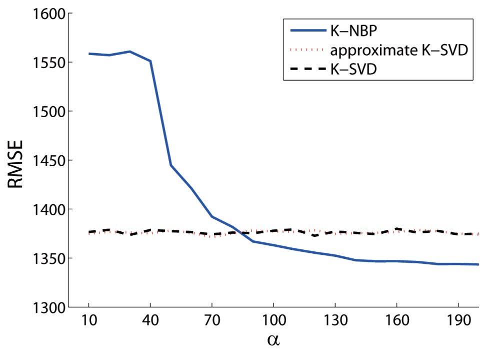

Fig. 5 RMSE for $\alpha \in [10, 200]$.

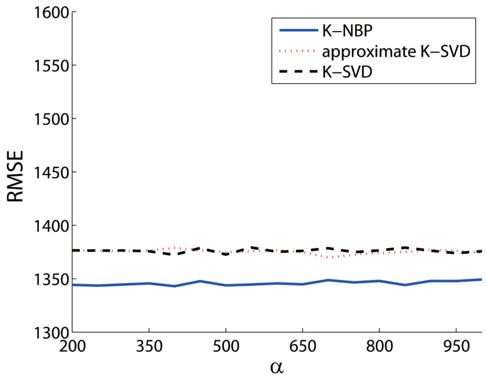

Fig. 6 RMSE for $\alpha \in [200, 1000]$.

$$\sqrt{\|B - \tilde{B}\|_F^2/64} \tag{3}$$

where $\|-\|_F$ indicates the Frobenius norm and 64 is the number of pixels in a block to evaluate how successful image reconstruction had been. The lower the value of RMSE, the better the image had been reconstructed.

The value of α used in Algorithm 3 was optimized before actual evaluation. α is a parameter by which the input matrix was multiplied with in the preprocessing stage. Since it depends on the statistical properties of data, a feasible value must be found for each type of data. **Figures 5** and **6** plot RMSE as the value of α was changed. The number of repetitions in the main loop, i.e., τ, was set to 5. The results indicate that K-NBP performs well in a wide range of α. α was set to 200 in the experiments that followed.

In order to see if this setting is applicable for other sets of images as well, the same test was carried out using another set of images, namely those of dragonflies, available at Caltech-256 [12]. The result is indicated in **Figs. 7** and **8**. Shapes of the graphs closely resemble those in prior figures, except for the scale of the y-axis. Here also, setting $\alpha = 200$ is an appropriate option.

Figure 9 shows the atoms obtained by K-NBP on 11000 blocks of local images, each with 8×8 pixels, at $\tau = 10$ and $\rho = 5$. The atoms in this figure are presented visually. The values of each of their components are represented by the intensities of corresponding pixels. Such images are often called basis images. They capture local features of images at different frequencies and orientations, and resemble receptive fields of neurons in vision [15], [16].

The test images were reconstructed with OMP by using these atoms. **Figure 10** has examples of the original, corrupted, and re-

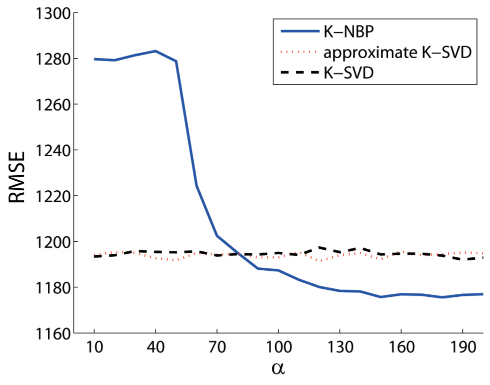

Fig. 7 RMSE for $\alpha \in [10, 200]$ for dragonfly images.

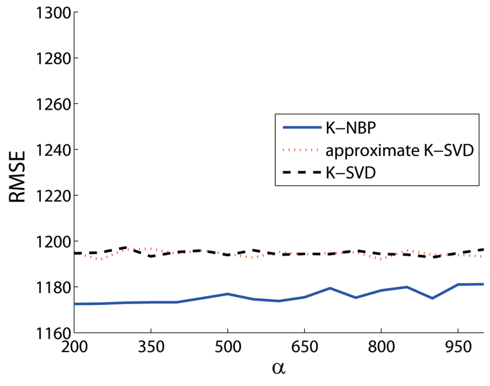

Fig. 8 RMSE for $\alpha \in [200, 1000]$ for dragonfly images.

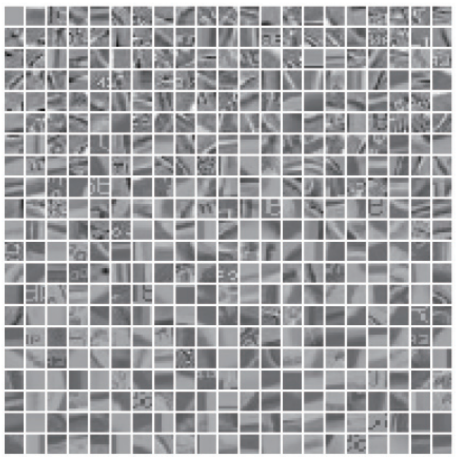

Fig. 9 Atoms (basis images) obtained from set of 8×8 image blocks.

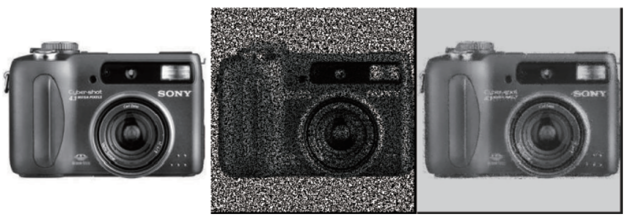

Fig. 10 Original, corrupted and reconstructed images using K-NBP (from left to right).

constructed images. Although high frequency components such as edges are blurred, overall reconstruction is successful.

Figure 11 plots RMSE as the number of repetitions in the main loop in the dictionary learning stage is changed. Ten test images were used. When there were fewer repetitions, K-NBP provided

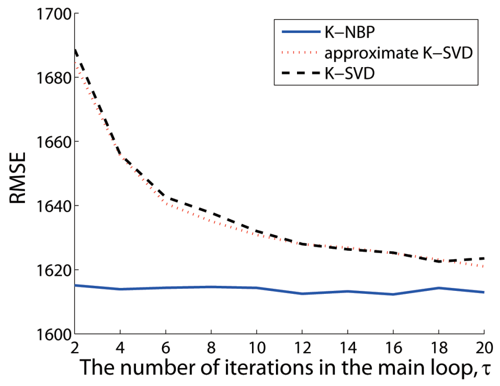

Fig. 11 RMSE when number of main iterations was changed.

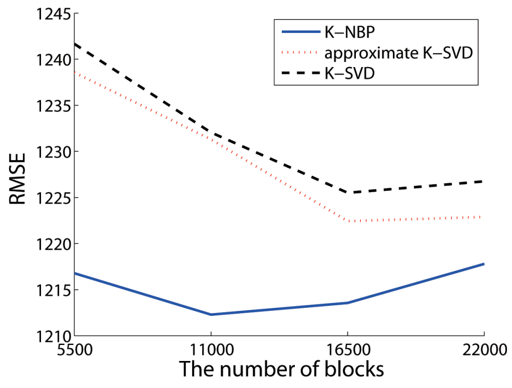

Fig. 12 RMSE when numbers of blocks were changed.

better results than K-SVD and the approximate K-SVD. This is roughly the same as the results obtained for the synthetic data.

Finally, the value of α for image data did not need to be changed for different sizes of input matrix Y. It amounted to changing the size and number of blocks in images. This was a very useful property since once α was determined from one data set, it could also be applied to other data sets of the same type. In other words, α needed to be optimized just once for each type of data set.

Figure 12 plots RMSE as the number of blocks was changed. This corresponds to changing the number of columns in Y. α was set to 200 and $\tau = 10$ and $\rho = 5$. Ten test images were used. The graph indicates that although the values of RMSE differ for different numbers of blocks, these are consistent with the changes in RMSE of K-SVD and the approximate K-SVD. Note that these two methods do not use α. K-NBP with $\alpha = 200$ outperforms K-SVD and the approximate K-SVD for a wide range of block sizes.

Figure 13 plots RMSE as the size of blocks were changed. The x-axis is the width of a block. For example, if the value is 4, the block size is 4×4. Changing this value corresponds to changing the number of rows in Y. Similar to when the numbers of blocks were changed, the change in the value of RMSE is consistent with the change in the RMSE of K-SVD and its approximation, which means α does not need to be changed as much.

These results indicate that the optimal value of α depends more on the statistical properties of the input matrix, rather than its size. Theoretical analysis on how optimal α is determined based on the statistical properties of data, and a systematic way of finding a

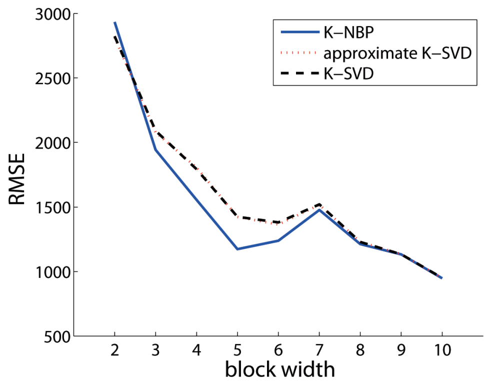

Fig. 13 RMSE when the size of blocks were changed.

feasible value is part of future work.

5. Related Work

Sparsity has recently been extensively discussed in the literature as one of the fundamental concepts in data processing [14], [21]. Sparse coding is also widely being used in various applications including image denoising [6], [8], blind source separation [11], and sensing [7].

There has been some work that has modified K-SVD to make it faster and more scalable. Peng and Hwang proposed the use of the proximal method to improve it [17]. Rubinstein et al. used bilateral projections to approximate SVD [18]. Their methods are different from the one proposed in this paper, since normalization of the source vector is not involved.

Chang et al. proposed a single-pass algorithm for K-SVD [3]. It makes K-SVD applicable to a large data set by dissolving the need to repeatedly access data. In contrast, this paper proposes a method of reducing reconstruction error using a geometric property of the data distribution. The two techniques can be combined to increase the performance in both ways. Developing a single-pass version of K-NBP would be an interesting topic to explore.

Mairal et al. proposed task-driven dictionary learning, where knowledge about the task was exploited. They have developed algorithms for online dictionary learning based on stochastic gradient descent [13]. Dai et al. proposed simultaneous codeword optimization (SimCo) that generalized two commonly used dictionary learning methods, i.e., the method of optimal directions (MOD) and K-SVD, and that also allowed simultaneous updates of atoms and source vectors [4]. Spielman et al. developed an algorithm called Exact Recovery of Sparsely-Used Dictionaries (ER-SpUD) that stochastically recovered the dictionary and source matrix [20]. Sadeghi et al. have recently proposed a fast approximation to dictionary learning based on convexifying the dictionary learning problem [19].

6. Conclusion

In this paper, K-NBP, a new method of dictionary learning was proposed, which replaces SVD in K-SVD with bilateral projections. The experiments showed that the use of bilateral projections produces good results-SVD while reducing the total computation time. K-NBP is a promising option to satisfy the currently increasing demands to apply dictionary learning to large scale data.

K-NBP was tested using the image reconstruction task. K-NBP, however, is a very general method that could also be applied to other types of data. In future work the method will be tested for other tasks in data analysis.

One interesting generalization of K-NBP is to use a structured projection or structured random projections instead of random projections. If there is some background knowledge about the distribution of data, it could be incorporated into the projection step. This could be very effective for certain application domains.

K-NBP does not improve the efficiency in terms of space, i.e. memory usage. One way to overcome this limitation in K-NBP is to develop a single-pass algorithm [3], which can potentially increase the number of images that could be processed.

Also, faster convergence may be achieved by modifying the bilateral projection step so that is more suited to the purpose of dictionary learning. For example, it may be effective to introduce a learning rate to control the randomness inherent in the algorithms. The adequacy of the normalization stage, namely dividing the input matrix by $\sqrt{n}$, still requires further theoretical investigations. For example, the dimension of $\tilde{E}_h$ is a random variable and must be analyzed using a stochastic framework.

Acknowledgments This work was supported in part by the Japan Society for the Promotion of Science (JSPS) KAKENHI: Grant Numbers 21700121, 25280110, and 25540159.

References

[1] Aharon, M., Elad, M. and Bruckstein, A.: K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation, *IEEE Trans. Signal Processing*, Vol.54, No.11, pp.4311–4322 (2006).
[2] Bruckstein, A., Donoho, D.L. and Elad, M.: From sparse solutions of systems of equations to sparse modeling of signals and images, *SIAM Review*, Vol.51, Issue 1, pp.34–81 (2009).
[3] Chang, K., Lin, C., Chen, C. and Hung, Y.: Single-Pass K-SVD for Efficient Dictionary Learning, *Circuits, Systems, and Signal Processing*, Vol.33, pp.309–320 (2014).
[4] Dai, W., Xu, T. and Wang, W.: Simultaneous codeword optimization (SimCO) for dictionary update and learning, *IEEE Trans. Signal Processing*, Vol.60, No.12, pp.6340–6353 (2012).
[5] Deutsch, F.: *Best approximation in inner product spaces*, Springer, Berlin (2011).
[6] Dong, W., Li, X., Zhang, L. and Shi, G.: Sparsity-based image denoising via dictionary learning and structural clustering, *2011 IEEE Conf. on Computer Vision and Pattern Recognition*, pp.457–464 (2011).
[7] Donoho, D.L.: Compressed sensing, *IEEE Trans. Information Theory*, Vol.52, No.4, pp.1289–1306 (2006).
[8] Elad, M. and Aharon, M.: Image denoising via sparse and redundant representations over learned dictionaries, *IEEE Trans. Image Processing*, Vol.15, No.12, pp.3736–3745 (2006).
[9] Elad, M.: *Sparse and Redundant Representations*, Springer, Berlin (2010).
[10] Field, D.J.: What is the goal of sensory coding?, *Neural Computation*, Vol.6, No.4, pp.559–601 (1994).
[11] Gribonval, R. and Zibulevsky, M.: Sparse component analysis, Comon, P. and Jutten, C. (ed.), *Handbook of Blind Source Separation: Independent Component Analysis and Applications*, Elsevier, Oxford, pp.367–420 (2010).
[12] Griffin, G., Holub, A. and Perona, P.: Caltech-256 object category dataset, Caltech Technical Report, No.CNS-TR-2007-001 (2007).
[13] Mairal, J., Bach, F. and Ponce, J.: Task-driven dictionary learning, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.34, No.4, pp.791–804 (2012).
[14] Mallat, S. and Wavelet, A.: *Tour of Signal Processing—The Sparse Way*, Academic Press, Oxford (2009).
[15] Olshausen, B.A. and Field, D.J.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images, *Nature*, Vol.381, pp.607–609 (1996).
[16] Olshausen, B.A. and Field, D.J.: Sparse coding of sensory inputs, *Cur-*

rent Opinion in Neurobiology, Vol.14, pp.481–487 (2004).

[17] Peng, G. and Hwang, W.: A proximal method for the K-SVD dictionary learning, *Proc. 23rd IEEE International Workshop on Machine Learning for Signal Processing* (2013).

[18] Rubinstein, R., Zibulevsky, M. and Elad, M.: Efficient implementation of the K-SVD algorithm using batch orthogonal matching pursuit, *Technical Report - CS Technion* (2008).

[19] Sadeghi, M., Babaie-Zadeh, M. and Jutten, C.: Dictionary learning for sparse representation: A novel approach, *Proc. 39th International Conf. Acoustic, Speech and Signal Processing*, pp.3601–3604 (2014).

[20] Spielman, D.A., Wang, H. and Wright, J.: Exact recovery of sparsely-used dictionaries, *Proc. 25th Conf. Learning Theory*, pp.37.1–37.18 (2012).

[21] Starck, J., Murtagh, F. and Fadili, J.M.: *Sparse Image and Signal Processing*, Cambridge University Press, Cambridge (2010).

[22] Tezuka, T.: A dictionary learning algorithm for sparse coding by the normalized bilateral projections, *Proc. 24th IEEE International Workshop on Machine Learning for Signal Processing* (*MLSP2014*), Reims, France (2014).

[23] Zhou, T. and Zhao, D.: GoDec: Randomized low-rank and sparse matrix decomposition in noisy case, *Proc. 28th International Conf. Machine Learning*, pp.33–40 (2011).

[24] Zhou, T. and Tao, D.: Bilateral random projections, *Proc. 2012 IEEE International Symposium on Information Theory*, pp.1286–1290 (2012).

Taro Tezuka is an associate professor at the Faculty of Library, Information and Media Science in the University of Tsukuba. He got his Ph.D. in informatics from Kyoto University in 2005. He worked in Ritsumeikan University prior to joining the University of Tsukuba in 2011. He was a visiting researcher at the Naval Academy Research Institute in France from 2014 to 2015. He is currently working on sparse coding and dictionary learning for image processing.