REVISIT COMPLETE DICTIONARY LEARNING VIA L_p NORM AND ITS APPLICATION ON DENOISING

Yufan Feng, Hangfei Zhang
School of Information Science and Technology
Shanghai Tech University
{fengyf2, zhanghf1}@shanghaitech.edu.cn

ABSTRACT

Dictionary learning, also known as sparse coding, is a representing method aiming to find a sparse representation of input data. In this paper, we review some related works of dictionary learning especially under the complete settings, compare the algorithms and performances, and also apply dictionary learning on image denoising.

1 INTRODUCTION

Traditional machine learning learns on huge database, and usually extract high-dimension of features, which means dense matrices to process, resulting in huge amount of calculations and low degree of interpretability. What dictionary learning wants to do is to sieve out useless feature information and find the sparse representation of input data. Specifically, for the input database with n samples

$$\mathbf{Y} = [\mathbf{y_1}, \ldots, \mathbf{y_n}] \in \mathbb{R}^{d \times n}$$

we want to find dictionary $\mathbf{D} \in \mathbb{R}^{d \times d}$ and a sparse representation $\mathbf{X} \in \mathbb{R}^{d \times n}$ such that

$$\mathbf{Y} = \mathbf{DX} \text{ or } \mathbf{y}_i = \mathbf{D}\mathbf{x}_i, \text{ where } i = 1, \cdots, p$$

Note that both $\mathbf{D}$ and $\mathbf{x}$ are unknown, and we wish to find them by learning on a sufficiently large dataset $\mathbf{Y}$. The dictionary $\mathbf{D}$ is undercomplete if $n < d$, and is overcomplete if $n > d$. For undercomplete $\mathbf{D}$, the atoms of $\mathbf{D}$ need to be orthogonal, while researchers usually consider the complete and overcomplete $\mathbf{D}$ for more flexible dictionaries and richer data representations. Such a problem can be modeled as an optimization problem, where the first term asks for the accuracy of the model, and the second term asks for the sparsity of X. We include $\lambda > 0$ to control the trade off:

$$\arg\min_{D, X_i} \sum_{i=1}^{N} \|y_i - D \cdot x_i\|_2^2 + \lambda \sum_{i=1}^{N} \|x_i\|_0. \tag{1}$$

Minimizing equation 1 is difficult, as L_0-norm is a non-convex, non-smooth, discontinuity, global non-differentiable function. So, minimization of the second term of the L_0-norm is a challenging NP-Hard problem. Thus, L_1-norm was used in some previous works due to it can also promote sparsity. However, since only one single row can be solved at a time, those methods often suffered from high computation complexity.

Various methods have been proposed to tackle this problem, e.g. LLC (locality-constrained linear coding) algorithm, LASSO and online learning methods. The K-SVD method (Aharon et al. (2006)) utilizes sparse coding algorithms as OMP (Orthogonal Matching Pursuit) to encode $\mathbf{x}$, then update each atom of the dictionary $\mathbf{D}$ by SVD decomposition. It alternatively freeze one term of $\mathbf{D}$ and $\mathbf{x}$, then renew another. This approach is employed in many different applications and is regarded as a classical method of dictionary learning. However, it also has flaws due to its high computation complexity, and it is likely to be trapped in a local minima.

Intuitively, finding a associated sparse representing is a compression problem. However, this abstract mathematical model can be applied in numerous practical situations with $\mathbf{Y}$ being various signals (e.g. images, audios, videos, etc.). Image is a unique two-dimensional signal. High-frequency information in images is visually reflected as structural features such as corners and edges. As a result, dictionary can discover latent structures in natural images and tackle with a series of downstream image processing tasks. For example, as $\mathbf{Y}$ refers to images with noise in image denoising scheme, we can solve $\mathbf{x}$ to find original images. And in signal classification tasks, we can construct highly-condensed feature $\mathbf{x}$ from original signals $\mathbf{Y}$.

2 LITERATURE REVIEW

2.1 L_p-NORM BASED DICTIONARY LEARNING

From the conception of dictionary learning, the simplest problem formulation was proposed as

$$\min_{X,D} \|X\|_0, \text{ subject to } Y = DX, D \in O(n; R), \tag{2}$$

directly translated from the idea that representing Y in form of a dictionary, where L_0-norm denotes the sparsity of X, minimizing the number of non-zero items in X. However, such formulation remains more modifications to become more resolvable. Considering such optimal problem is non-convex for both L_0-norm objective function and sparsity constraint condition, we can apply what usually used in optimization problem, One of the methods is to connect two optimal objectives, sparsity of X and correctness of representation DX, by a hyper-parameter λ, which is what equation 1 demonstrates. We can deflate the problem to L_1-norm, in which we can achieve optimal solution but the convergence rate is too slow due to the non-smoothness, as shown in Fig.1. L_1-norm presents better properties among convexity and smoothness, and it can also promote sparsity, giving more achievable problem formulation 3 and can be solved by heuristic algorithms, but unfortunately heuristic algorithm lacks rigorous proof of convergence (Shao et al. (2014)).

$$\min_{X,D} \|X\|_1, \text{ subject to } Y = DX, D \in O(n; R), \tag{3}$$

Here we want to introduce another method widely used in representation learning, via L_p-norm maximization. Sun et al. (2015) proposed an executable method to work out one column of X,

$$\min_{d \in \mathbb{R}^n} \|d^* Y\|_1, \text{ subject to } \|d\|_2 = 1. \tag{4}$$

Problem 4 is with spherical constraint, and in L_p function, we can see from Fig.1 that L_p functions with different p share something similar especially when we are seeking for the sparsity of X. When $p = 1$, sparse solution leads to the coordinates in $\mathbb{R}^2$, as well as we solve a L_1-norm minimization problem. When $p \geq 2$, if we want to obtain the same solution on coordinates, AKA the sparse solution, on the contrary we need to solve a L_p-norm ($p \geq 2$) maximization problem. Rewrite $X = D^* Y, A = D^*$, therefore, we intuitively transfer problem 3 to an L_p-norm maximization problem as follows.

$$\max_{d \in \mathbb{R}^n} \|AY\|_p (p \geq 2), \text{ subject to } A \in O(n; R). \tag{5}$$

Then we think about the equivalence and convergence of problem 5 in dictionary learning. Luckily lots of literature discussed about it. Shen et al. (2020) gave short illustration that $L_p (p \geq 2)$ problem can get very close to the true dictionary in high probability under some condition, and Zhai et al. (2020) proved completely that such 2 can be reformulated to an L_4 norm problem in perfect equivalence, both the correctness of global optima and convergence rate. Though Zhai et al. (2020) only proved case of L_4-norm, viewing the similarity of L_p-norm($p \geq 2$), we can abstractly think that similar conclusion holds for L_p-norm($p \geq 2$) maximization problem in dictionary learning.

To solve L_p-norm($p \geq 2$) maximization problem, there are many contributions have been proposed, such as GPM algorithm in Journée et al. (2010), applying linear approximation and then gradient along the linear direction.

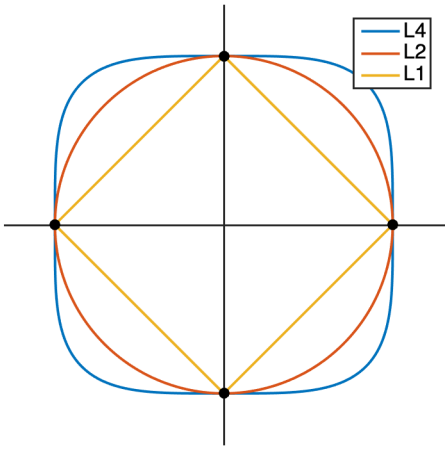


Figure 1: Unit Circle of L_p-norm in $\mathbb{R}^2$, where $p = 1, 2, 4$.

2.2 Dictionary learning algorithm

To maximize a convex function $f(\cdot)$, Journée et al. (2010) proposed the generalized power method (GPM) algorithm. (see Algorithm 4) This algorithm has great theoretical convergence properties that it achieves linear or super-linear convergence rates while maintaining a cheap per-iteration cost.

Algorithm 1: GPM

1 initialize $\mathbf{A}_0 \in D$;
2 **for** $t = 0, 1, \cdots, T-1$ **do**
3 $\quad \mathbf{A}_{t+1} = \underset{\boldsymbol{s} \in D}{\operatorname{argmax}} \langle \boldsymbol{s}, f'(\mathbf{A}_t) \rangle$
4 **end**
Output: $\mathbf{A}_t$

From the GPM algorithm, Zhai et al. (2020) derived an efficient algorithm MSP (see Algorithm 6) based on L_4-based dictionary learning. This algorithm, refers to *matching, stretching, and projection*, was shown to maintain the efficiency of GPM in practice as it performs a projected gradient ascent with a indefinite step size for each iteration. It starts with a randomly initialized $\mathbf{A}$ on $\mathrm{O}(n; \mathbb{R})$.

Algorithm 2: MSP

1 initialize $\mathbf{A}_0 \in \mathrm{O}(n, \mathbb{R})$;
2 **for** $t = 0, 1, \cdots, T-1$ **do**
3 $\quad \partial \mathbf{A}_t = 4(\mathbf{A}_t \mathbf{D}_o)^{\circ 3} \mathbf{D}_o^*$;
4 $\quad \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* = \mathit{SVD}(\partial \mathbf{A}_t)$;
5 $\quad \mathbf{A}_{t+1} = \mathbf{U}\mathbf{V}^*$;
6 **end**
Output: $\mathbf{A}_T$

In Line 3 of 6, $\mathbf{A}_t$ first *matches* with the true dictionary $\mathbf{D}_O$. Then $(\cdot)^{\circ 3}$ ($\circ$ implies the Hadamard product) is used to *stretch* all entries of $\mathbf{A}_t \mathbf{D}_O$ with enlarge the large ones and minimize the small

ones. Also, $4(\mathbf{A}_t\mathbf{D}_O)^{\circ 3}\mathbf{D}_o^*$ can be treated as the gradient of the objective function.

$$\nabla_{\boldsymbol{A}} g\left(\boldsymbol{A}\boldsymbol{D}_o\right) = \nabla_{\boldsymbol{A}} \left\|\boldsymbol{A}\boldsymbol{D}_o\right\|_4^4 = 4\left(\boldsymbol{A}\boldsymbol{D}_o\right)^{\circ 3} \boldsymbol{D}_o^* \tag{6}$$

Then, line 4 and 5 *projects* $\mathbf{A}_t$. This projection is the orthogonal matrix which is closest to $\mathbf{A}$ in Frobenius distance, as a lemma was shown in Zhai et al. (2020):

$$\mathcal{P}_{\mathrm{O}(n;\mathbb{R})}(\boldsymbol{A}) = \underset{\boldsymbol{M}\in \mathrm{O}(n;\mathbb{R})}{\arg\min} \|\boldsymbol{A}-\boldsymbol{M}\|_F^2 = \boldsymbol{U}\boldsymbol{V}^*, \text{ where } \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^* = \mathit{SVD}(A) \tag{7}$$

Through this process, sparse and spiky solutions to $\mathbf{A}_t\mathbf{D}_O$ are obtained. At last, it will converge towards a signed permutation matrix. Since the expectation of $\nabla_{\boldsymbol{A}}\|\boldsymbol{A}\boldsymbol{Y}\|_4^4$ approximately refers to $3p\theta$, the final output is normalized by $\|\mathbf{A}_T\mathbf{Y}\|_4^4/3np\theta$ in the scope of dictionary learning.

Although same properties for all $L_p(p > 2)$ have not been proved yet, Shen et al. (2020) also provides a practical algorithm (see Algorithm 6) based on GPM algorithm. We can see that the derivative part is adapted to L_p scheme.

Algorithm 3: The GPM for L_p based dictionary learning

1 initialize $\mathbf{A}_0^* \in \mathrm{St}(n, m)$;
2 **for** $t = 0, 1, \cdots, T-1$ **do**
3 $\quad \partial\mathbf{A}_t = \left(|\mathbf{A}_t\mathbf{Y}^{\circ(p-1)}| \circ \mathrm{sign}(\mathbf{A}_t\mathbf{Y})\right)\mathbf{Y}^*$;
4 $\quad \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^* = \mathit{SVD}(\partial\mathbf{A}_t)$;
5 $\quad \mathbf{A}_{t+1} = \mathbf{U}\mathbf{V}^*$;
6 **end**
Output: $\mathbf{A}_T$

2.3 Image denoising based on dictionary learning

Since there is no restrictions on the characteristics of the input matrix Y for the representation $Y = DX$, being represented as product of dictionary D and sparse matrix X.

Currently, some previous work tackle image denoising task with K-SVD method(Aharon et al. (2006)). Here K-SVD algorithm is an alternative algorithm, in which we first find the sparse coding of X, and then alternatively fix X and update the dictionary. There are multiple ways of calculating X and updating dictionary, leading to different performances, for example some pursuit algorithms for sparse coding, like OMP, LARS.

The most typical method was proposed by Elad & Aharon (2006). First we separate the origin image into sets of patches and train the dictionary on the patches, which can be seen as an update of atom. Questions occurs if we try to find a enough universal dictionary that how we can recover the image and by which dictionary. The dictionary is trained by pre-given image sets, but we cannot ensure that all the test images share the same dictionary as the training sets, and it is hard to design a training sets robust for all verification sets. After training dictionary on origin image, we train dictionary on corrupted images, in other words apply denoising on image by dictionary learning. Similarly, we take sparse coding and training on patches of the corrupted image, and then averaging for the results.

The reason why the K-SVD method could be more easily adapted to image denoising scheme is because it has a sparse representation stage. As mentioned above, during the training process, it would try to minimize the noise in the OMP stage.

Another related way to denoise images is based on PCA. In Zhang et al. (2010), the authors proposed a two-stage algorithm LPG-PCA. LPG, local pixel grouping, models a pixel and its nearest neighbors as a vector, better preserving local structures when training. More specific on denoising, when apply recover algorithm, LPG holds more local detail, since training sample sets are selected from the local window. LPG-PCA performs better on Gaussian Noise, whereas K-SVD denoising results look like still exists a layer of Gaussian noise. Results given in Zhang et al. (2010) are shown in Fig.2. PCA is most used on data dimension reduction, and for denoising, consider image as large dimension of matrix, enhancement of sparsity shares similar motivation with dimension reduction.

Figure 2: The denoising results of Lena by (a) Noiseless (d) K-SVD (f) LPG-PCA

3 Criticism of the Existing Work

Dictionary learning is a sparse coding method in representation learning, aiming to extract main features of signals. Naturally we think of Singular Value Decomposition for such show well-pleasing performance of recovery. That is what K-SVD algorithm realizes, keeping k singular values as a term of representation, which theoretically makes a role but in actual experiments, the execution time and convergence rate seems acting not well so much.

General optimization method dealing with concave objective functions can also be used in this problem scenario. Before we have transfer problem 2 to formulation 4 with spherical constraint, and then we can apply Generalized Power Method (GPM) algorithm. Journée et al. (2010) proposed a general gradient algorithm for spherical constraint, stating pleasing property of convergence, robustness, initialization, and global optimization. GPM will converge with high probability to the true optimization for enough size of database in polynomial time complexity, shows robustness to not such large noise, and without special initialization.

In our problem, L_p-norm$(p \geq 2)$ maximization with spherical constraint, GPM demands for the gradient of $\|AY\|_p^p$. Shen et al. (2020) states the gradient as line 3 of Algorithm 6, and then we can plug it in GPM and obtain Algorithm as in 6. Without loss of generality, we can assume that all L_P-norm in sphere have good properties, but actually there still exists many saddle points that may affect our results. Luckily, Zhai et al. (2020) prove that when $p = 4$ it can achieve much better properties. More specifically, all saddle points have negative curvatures Zhai et al. (2020), which means we can get to the global optimal point Instead of thinking about considering how to escape the saddle points. Equation 6 is what Zhai et al. (2020) proposed as gradient.

There is some trade-off among L_p-norm algorithms. Experiments by Shen et al. (2020) verify what we just discussed about, including the convergence rate of L_p-norm problem over the sphere constraint – the convergence involves two stages, where the first stage only takes a few iterations and the second enjoys a linear convergence rate. They also conclude from experiments that, without introducing projection or other methods, L_3-norm exhibits the most balanced properties among lowest sample complexity, robustness, and time-efficiency.

4 Our Contributions

From our perspective, L_p-norm based dictionary learning is kind of similar to PCA, but it could better represent the information according to the experiments. In this work, we tried to perform L_p-norm based dictionary learning on image denoising.

Specifically, we noticed that if we look into the norm of the output sparse representation, those columns with larger norm contains the main information of the images. A simple way to remove the noise is to keep the columns with large norms and delete the others. Follow the previous works, we also segment the image into N patches with shape $p \times p$, and then reshape those patches as $Y \in \mathbb{R}^{N \times p^2}$. The details of our experiments will be included in the next section.

5 NUMERICAL RESULTS

We choose PSNR and SSIM as our evaluation metrics. PSNR, refers to Peak Signal-to-Noise Ratio, measures the quality of reconstruction quality. Usually, the higher PSNR often indicates the closer to the original image. SSIM refers to Structual Similarity Index Measure. It evaluates luminance, contrast, and structure of the image, and would be more in line with the intuitive feeling of the human eyes. The range of SSIM is $[-1, 1]$, and it equals to 1 when two images are completely the same.

To simply examine our ideas of keeping the pivots with the most information, we firstly test the algorithm on different remaining columns number. We add a random Gaussian noise with sigma $= 25$ (pixel values range from 0 to 255) on the image. The results are shown in Figure 3 and Table 1.

(a) Original (b) Noisy (c) top 5

(d) top 20 (e) top 50 (f) top 80

Figure 3: Image denoising using LpDL with top K remaining columns

topK	Noisy	0	5	10	20	30
PSNR	20.16	20.18	26.49	28.12	**29.09**	28.89
SSIM	0.2973	0.2969	0.7462	**0.7784**	0.7560	0.7068
topK	40	50	80	100	150	200
PSNR	28.71	28.55	27.62	26.75	24.41	22.24
SSIM	0.6878	0.6739	0.6161	0.5685	0.4576	0.3695

Table 1: test on remaining columns

We also evaluated the effects brought by the hyperparameters.

- Max iteration time
  Max iteration time refers to T in MSP Algorithm. Surprisingly, we can see from Table 2

that MSP algorithm could converge in only a few iterations.

max iteration number	1	2	5	8	10
PSNR	23.38	28.90	**29.16**	29.05	28.96
SSIM	0.7365	**0.7734**	0.7437	0.7303	0.7178
max iteration number	20	40	80	100	150
PSNR	28.86	28.87	28.85	28.89	28.87
SSIM	0.7064	0.7076	0.7049	0.7075	0.7074

Table 2: test on max iteration number

- Max training size
  As all patches can be treated as one piece of sample in the image denoising scheme. Therefore, we believe that even if we do not use all the patches, we can still learn a general dictionary from some randomly sampled patches. Table 3 shows that we need a certain number of patches, but we indeed do not need all the patches.

Train size	100	500	1000	5000
PSNR	27.37	27.95	28.03	28.82
SSIM	0.7690	0.7761	0.7777	**0.7798**
Train size	10000	50000	100000	all
PSNR	**29.15**	29.12	29.14	29.15
SSIM	0.7670	0.7538	0.7545	0.7564

Table 3: test on max training size

- Patch size
  We also adapt the patch size of original patches. From Table 4, basically larger patches would lead to better effects. However, when the patch becomes too large, as our dictionary size would also be expanded and it would be the power of patch size, time consumption would be exploded.

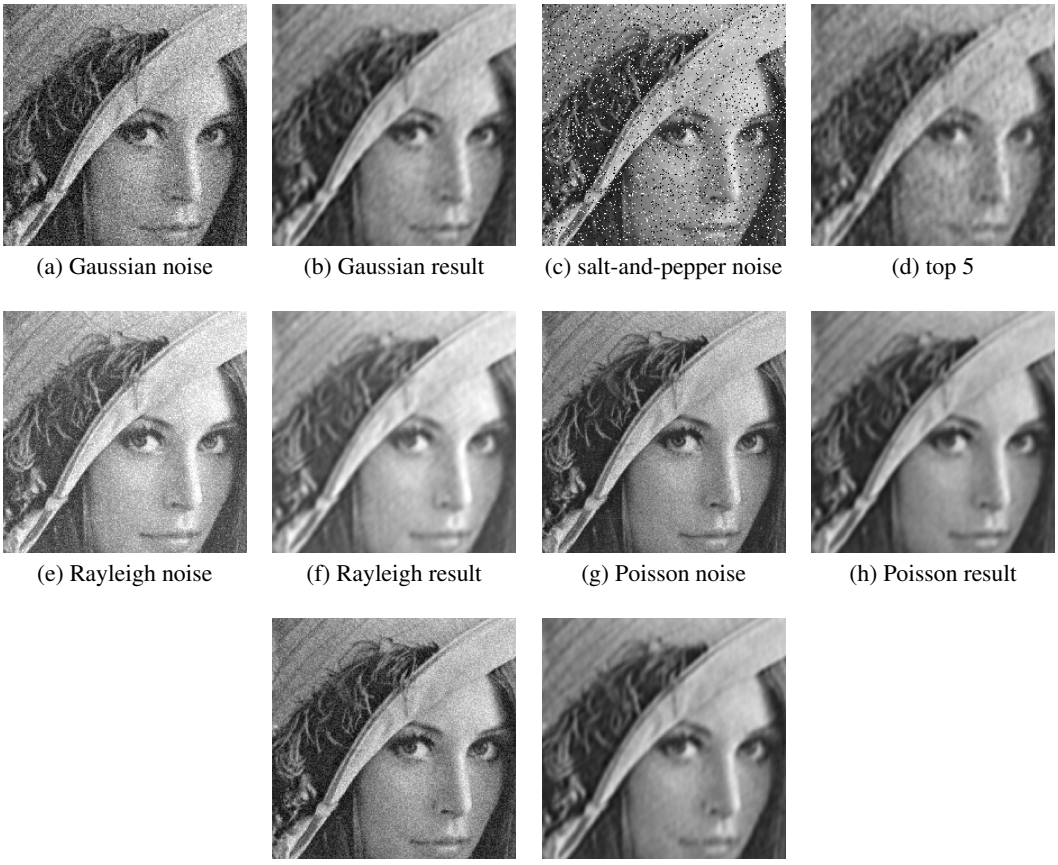
Patch size	4	8	12	16	32
PSNR	21.26	25.66	28.55	**28.82**	27.52
SSIM	0.3339	0.5148	0.6719	0.7026	**0.7636**
time (s)	1.26	5.61	13.20	23.59	136.55

Table 4: test on patch size

Furthermore, we applied various type of noises on the original image and got the results of Figure 4. Surprisingly, we find that this method is effective to almost all types of noises except for the salt-and-pepper noise. Perhaps this is because L_p-norm maximization is easy to be affected by the peak values.

6 CONCLUSIONS

We review the modification of the problem formula, from L_0 minimization, L_1 minimization, to $L_p(p = 2, 3, 4)$ maximization, and also compare the theoretical performance among them. We also discuss the similarity among K-SVD, LPG-PCA, and Dictionary Learning applied in the field of image denoising. We apply L_p norm maximization on image denoising, adding different types of noise. Finally we can conclude that the experimental results acts very well.



(a) Gaussian noise (b) Gaussian result (c) salt-and-pepper noise (d) top 5

(e) Rayleigh noise (f) Rayleigh result (g) Poisson noise (h) Poisson result

(i) Uniform noise (j) Uniform result

Figure 4: Image denoising on different type of noises

REFERENCES

Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11): 4311–4322, 2006.

Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006.

Michel Journée, Yurii Nesterov, Peter Richtárik, and Rodolphe Sepulchre. Generalized power method for sparse principal component analysis. *Journal of Machine Learning Research*, 11 (2), 2010.

Ling Shao, Ruomei Yan, Xuelong Li, and Yan Liu. From heuristic optimization to dictionary learning: A review and comprehensive comparison of image denoising algorithms. *IEEE Transactions on Cybernetics*, 44(7):1001–1013, 2014. doi: 10.1109/TCYB.2013.2278548.

Yifei Shen, Ye Xue, Jun Zhang, Khaled Ben Letaief, and Vincent K. N. Lau. Complete dictionary learning via L_p-norm maximization. *CoRR*, abs/2002.10043, 2020. URL `https://arxiv.org/abs/2002.10043`.

Ju Sun, Qing Qu, and John Wright. Complete dictionary recovery over the sphere. In *2015 International Conference on Sampling Theory and Applications (SampTA)*, pp. 407–410, 2015. doi: 10.1109/SAMPTA.2015.7148922.

Yuexiang Zhai, Zitong Yang, Zhenyu Liao, John Wright, and Yi Ma. Complete dictionary learning via l4-norm maximization over the orthogonal group. *J. Mach. Learn. Res.*, 21(165):1–68, 2020.

Lei Zhang, Weisheng Dong, David Zhang, and Guangming Shi. Two-stage image denoising by principal component analysis with local pixel grouping. *Pattern recognition*, 43(4):1531–1549, 2010.