NAGSC: NESTEROV'S ACCELERATED GRADIENT METHODS FOR SPARSE CODING

Liang Liu
Chongqing University of Posts and Telecommunications
Chongqing, 400065
China
liuliang@cqupt.edu.cn

Ling Zhang
Chongqing Industry Polytechnic College
Chongqing, 401120
China
466695057@qq.com

Xiangguang Dai
Key Laboratory of Intelligent Information Processing and Control of Chongqing Municipal
Institutions of Higher Education
Chongqing Three Gorges University
Wanzhou, Chongqing, 404100
China

Yuming Feng[*]
Chongqing Engineering Research Center of Internet of Things and Intelligent Control Technology
Chongqing Three Gorges University
Wanzhou, Chongqing, 404100
Chinayumingfeng25928@163.com

Abstract. This paper proposes efficient algorithms for Sparse Coding. Firstly, Sparse Coding is divided into two sub-convex problems including L1 and L2 problems. Secondly, we transform the nonsmooth L1 problem into two smooth sub-problems, and alternatively optimize them by Nesterov's Accelerated Gradient methods (NAG). Thirdly, we apply NAG to optimize L2 problem. Finally, L1 and L2 problems are iteratively solved until convergence. Experiments show that our proposed algorithms are effective to optimize L1, L2 and learn over-complete bases.
Keywords: sparse coding, nonsmooth, nonconvex, accelerated gradient.

1. Introduction

Sparse Coding (SC) can be viewed as an unsupervised method to find representations for the original data. Given the unlabeled image data, SC learns bases to capture the intrinsic features, and the learned bases resemble the receptive

∗. Corresponding author

fields of visual neurons [1, 2]. If the number of bases is smaller than the dimension of the original data, SC can be applied to dimensional reduction such as Principal Component Analysis [3], Locality Preserving Projections [4] and Linear Discriminant Analysis [5]; otherwise, SC can be used to learn a set of over-complete bases. Based on these properties, SC is widely applied to pattern recognition [6, 7], clustering [8, 9] and signal processing [10, 11].

SC is to learn sets of basis vectors so that an input vector can be represented by the combination of these basis vectors. Given an input vector $\overrightarrow{\xi} \in R^m$, SC can find the basis vectors $\overrightarrow{b_1}, \cdots, \overrightarrow{b_r} \in R^m$ and the coefficient vector $\overrightarrow{s} \in R^r$ such that the $\overrightarrow{\xi}$ can be represented by

$$\overrightarrow{\xi} \approx \sum_j \overrightarrow{b_j} s_j.$$

Generally, SC hopes to learn over-complete sets of basis vectors $\overrightarrow{b_1}, \cdots, \overrightarrow{b_r} \in R^m$ to represent a input vector $\overrightarrow{\xi} \in R^m$ (i.e. $r > m$). Given n input vectors $\overrightarrow{\xi^1}, \cdots, \overrightarrow{\xi^n}$ and their corresponding coefficient vectors $\overrightarrow{s^1}, \cdots, \overrightarrow{s^n}$, the SC model can be mathematically defined by

$$\begin{aligned} \min_{b_j, s^i} \quad & \sum_{i=1}^{n} \frac{1}{2} \parallel \overrightarrow{\xi}^i - \sum_{j=1}^{r} \overrightarrow{b}_j s_j^i \parallel^2 + \beta \sum_{i=1}^{n} \sum_{j=1}^{r} \parallel s_j^i \parallel_1 \\ \text{subject to} \quad & \parallel \overrightarrow{b_j} \parallel^2 \leq c, \forall j = 1, \cdots, r. \end{aligned} \tag{1}$$

Suppose that $V = [\overrightarrow{\xi_1}, \cdots, \overrightarrow{\xi_n}] \in R^{m \times n}$ is an input matrix, $W = [\overrightarrow{b_1}, \cdots, \overrightarrow{b_r}] \in R^{m \times r}$ is a basis matrix and $H = [\overrightarrow{s^1}, \cdots, \overrightarrow{s^n}] \in R^{r \times n}$ is a coefficient matrix, then problem (1) can be transformed into the following matrix form:

$$\begin{aligned} \min_{W,H} \quad & F(W,H) = \frac{1}{2} \|V - WH\|_F^2 + \beta \sum_{i=1}^{n} \parallel H_i \parallel_1 \\ \text{subject to} \quad & \sum_i W_{i,j}^2 \leq c, \forall j = 1, \cdots, r. \end{aligned} \tag{2}$$

In this paper, we propose efficient algorithms to optimize SC. Firstly, problem (2) is divided into two sub-problems including L1 and L2 problems. Secondly, we transform the nonsmooth L1 problem into two smooth sub-problems, and alternatively optimize them by Nesterov's Accelerated Gradient methods (NAG) [12]. Thirdly, we apply NAG to optimize L2 problem. Finally, L1 and L2 problems are alternatively solved until convergence. According to the block-coordinate-descent method [13], we divide problem (2) into two convex problems as follows:

$$\begin{aligned} \min_{W} \quad & F(W) = \frac{1}{2} \|V - WH\|_F^2 \\ \text{subject to} \quad & \sum_i W_{i,j}^2 \leq c, \forall j = 1, \cdots, r \end{aligned} \tag{3}$$

and

$$\min_{H} F(H) = \frac{1}{2}\|V - WH\|_F^2 + \beta \sum_{i=1}^{n} \| H_i \|_1 . \tag{4}$$

For each sub-problem, we construct three sequences and update them recursively. These sequences can accelerate the optimization of each sub-problem. Therefore, this scheme leads to the convergence rate at $O(\frac{1}{k^2})$.

2. L1 optimization

Suppose that $P = [p_{ij}] = \max(H, 0)$, $Q = [q_{ij}] = \max(-H, 0)$, then $H = P - Q$. Problem (4) can be transformed as follows:

$$\begin{aligned} \min_{P,Q} \quad & F(P,Q) = \frac{1}{2}\|V - W(P-Q)\|_F^2 + \beta \sum_{i,j} p_{ij} + \beta \sum_{i,j} q_{ij} \\ \text{subject to} \quad & P \geq 0, Q \geq 0. \end{aligned} \tag{5}$$

To minimize (5), the common approach is the block-coordinate-descent method, where P and Q are alternatively minimized until convergence. Hence, we divide (5) into two the following optimization problems:

$$\begin{aligned} \min_{P} \quad & F(P) = \frac{1}{2}\|V + WQ - WP\|_F^2 + \beta \sum_{i,j} p_{ij} \\ \text{subject to} \quad & P \geq 0 \end{aligned} \tag{6}$$

and

$$\begin{aligned} \min_{Q} \quad & F(Q) = \frac{1}{2}\|V - WP - (-W)(Q)\|_F^2 + \beta \sum_{i,j} q_{ij} \\ \text{subject to} \quad & Q \geq 0. \end{aligned} \tag{7}$$

According to (6) and (7), it is clear that both of them have the same matrix form. In the following, we only consider how to optimize (6), then (7) can be solved accordingly. Based on Lemma 1 and Lemma 2, NAG can solve (6) efficiently.

Lemma 1 ([14])**.** The objective function $F(P)$ is convex.

Lemma 2 ([14])**.** The gradient $\nabla F(P)$ is Lipschitz continuous, and the Lipschitz constant is $\| W^T W \|$.

Recent researches show that NAG are suitable for convex optimization problems and can obtain the convergence rate at $O(\frac{1}{k^2})$. To use NAG effectively, the optimization problem should be convex and its gradient is Lipschitz continuous.

In particular, three sequences are constructed by NAG, and they are alternatively updated in each iteration. At the iteration number $k \geq 1$, the updating rules for optimizing (6) are given as follows:

$$
\begin{aligned}
&(8) \quad Y_k = \arg\min_{Y \geq 0} \{F(P_k) + < \nabla F(P_k), Y - P_k > + \frac{1}{2} L \parallel Y - P_k \parallel_F^2\}, \\
&(9) \quad Z_k = \arg\min_{Z \geq 0} (\frac{L}{2} \parallel Z - Z_{k-1} \parallel_F^2 + \tau_k (< \nabla F(P_k), Z >)), \\
&(10) \quad P_{k+1} = \alpha_k Z_k + (1 - \alpha_k) Y_k,
\end{aligned}
$$

where $\alpha_k = \frac{2}{k+3}$, $\tau_k = \frac{k+1}{2}$ and $< \cdot, \cdot >$ is the sum of the element-wise multiplication of two matrices. In the following, we apply KKT conditions to solve constrained optimization problems (8) and (9). By the Lagrange multiplier method, (8) is re-written to the following form:

$$
(11) \quad L(Y, \lambda) = F(P_k) + < \nabla F(P_k), Y - P_k > + \frac{1}{2} L \parallel Y - P_k \parallel_F^2 - < \lambda, Y >,
$$

where $\lambda \in R^{r \times n}$. Let (Y_k, λ_k) be the optimal solution of (11). The KKT conditions are summarized as

$$
(12) \quad Y_k \geq 0, \lambda_k \geq 0, < \lambda_k, Y_k >= 0,
$$

$$
(13) \quad \nabla F(P_k) + L(Y - P_k) - \lambda_k = 0.
$$

However, λ_k is unknown. According to (13), we have

$$
(14) \quad Y_k = P_k - \frac{1}{L} \nabla F(P_k) + \frac{1}{L} \lambda_k.
$$

It is clear that only $\lambda_k > 0$ and $Y_k = 0$ can satisfy (12). Hence, we obtain

$$
(15) \quad \lambda_k = \max(\nabla F(P) - L P_k, 0).
$$

Next, we substitute (15) into (14). The optimal solution of Y_k can be simplified by the following term.

$$
(16) \quad Y_k = \max(P - \frac{1}{L} \nabla F(P_k), 0).
$$

Similarly, the optimal solution of Z_k can be obtained as follows:

$$
(17) \quad Z_k = \max(Z_k - \frac{\tau_k}{L} \nabla F(P_k), 0).
$$

Above all, the final updating rules for (6) can be summarized as the following sequences:

$$
\begin{aligned}
&(18) \quad Y_k = \max(P_k - \frac{1}{L} \nabla F(P_k), 0), \\
&(19) \quad Z_k = \max(Z_k - \frac{k+1}{2L} \nabla F(P_k), 0), \\
&(20) \quad P_{k+1} = \alpha_k Z_k + (1 - \alpha_k) Y_k,
\end{aligned}
$$

where $L = \| W^T W \|$ and $\nabla F(P_k) = W^T W P_k - (W^T V + W^T W Q) + \beta E$, wherein all the elements in matrix $E \in R^{r \times n}$ are 1. Similarly, the final updating rules for (7) can be summarized as the following sequences:

$$Y_k = \max(Q_k - \frac{1}{L} \nabla F(Q_k), 0), \tag{21}$$

$$Z_k = \max(Z_k - \frac{k+1}{2L} \nabla F(Q_k), 0), \tag{22}$$

$$Q_{k+1} = \alpha_k Z_k + (1 - \alpha_k) Y_k, \tag{23}$$

where $\nabla F(Q_k) = W^T W Q_k - (-W^T V + W^T W P) + \beta E$. We summarize above updating rules (18) to (23) in Algorithm 1.

Algorithm 1 NAG for L1 (NAGL1)

Input: V, W, H, β, *outer*
Output: H
Initialization: *inner*, $L = \frac{1}{\|W^T W\|}$, $k \leftarrow 0$, $i \leftarrow 0$
for $i = 0$ to *outer* **do**
1. $P = max(H, 0)$
2. $Q = max(-H, 0)$
3. $Z = P$
4. $WTV = W^T V + W^T W Q$

for $k = 0$ to *inner* **do**
5. $F(P) = W^T W P - WTV + \beta$
6. $Y = \max(P - L \nabla F(P), 0)$
7. $Z = \max(Z - \frac{k+1}{2} L \nabla F(P), 0)$
8. $P = \frac{2}{k+3} Z + (1 - \frac{2}{k+3}) Y$

end for
9. $Z = Q$
10. $WTV = -W^T V + W^T W P$

for $k = 0$ to *inner* **do**
11. $F(Q) = W^T W Q - WTV + \beta$
12. $Y = \max(Q - L \nabla F(Q), 0)$
13. $Z = \max(Z - \frac{k+1}{2} L \nabla F(Q), 0)$
14. $Q = \frac{2}{k+3} Z + (1 - \frac{2}{k+3}) Y$
15. $k = k + 1$

end for
16. $H = P - Q$
17. $i = i + 1$

end for

Algorithm 1 accepts input V, W, H, β and *outer* and outputs H. W and H can be obtained from BCD. *inner* is the iteration number of each sub-problem and *outer* is the iteration number of Algorithm 1. In the following Theorem 1 and Theorem 2, Algorithm 1 can be demonstrated to achieve the convergence rate at $O(\frac{1}{k^2})$.

Theorem 1. Supposed that the two sequences $\{Y_k\}_{k=0}^{\infty}$ and $\{P_k\}_{k=0}^{\infty}$ are generated by (19) and (20), then we have

$$F(Y_k) - F(P^*) \leq \frac{2L \parallel P^* - P_k \parallel_F^2}{(k+1)(k+2)},$$

where P^* is an optimal solution to (6).

Proof. According to Proposition 2.1 in [15] and Lemma 1 in [12], we obtain

$$F(Y_k) - F(P^*) \leq \frac{1}{A_k}(\phi_0(P) - F(P^*)) \tag{24}$$

and

$$A_k F(Y_k) \leq \min_X \{\frac{1}{2} L \parallel X - P_k \parallel_F^2 + \sum_{i=0}^{k} \tau_i < \nabla F(P_i), X - P_i >\}, \tag{25}$$

where $A_k = \sum_0^k \tau_i = \frac{(k+1)(k+2)}{4}$ and $\phi_k(X) = \frac{1}{2}L \parallel X - P_k \parallel_F^2 + \sum_{i=0}^k \tau_i < \nabla F(P_i), X - P_i >$. We get

$$\begin{aligned} & F(Y_k) - (1 - \frac{1}{A_k})F(P^*) \leq \frac{1}{A_k}\phi_0(P^*) \\ & = \frac{1}{A_k}\{L \parallel P^* - P_k \parallel_F^2 + \tau_0[F(P_0) + < \nabla F(P_0), P^* - P_0 >]\} \\ & \leq \frac{1}{A_k}\frac{L}{2} \parallel P^* - P_k \parallel_F^2 + \frac{1}{A_k}\tau_0 F(P^*) \\ & \leq \frac{1}{A_k}\frac{L}{2} \parallel P^* - P_k \parallel_F^2 + \frac{1}{A_k}F(P^*). \end{aligned}$$

According to simple algebra, $F(Y_k) - F(P^*) \leq \frac{2L\|P^* - P_k\|_F^2}{(k+1)(k+2)}$.

Theorem 2. Supposed that the two sequences $\{Y_k\}_{k=0}^{\infty}$ and $\{Q_k\}_{k=0}^{\infty}$ are generated by (22) and (23), then we have

$$F(Y_k) - F(Q^*) \leq \frac{2L \parallel Q^* - Q_k \parallel_F^2}{(k+1)(k+2)},$$

where Q^* is an optimal solution to (7).

3. Constrained quadratic programming problem optimization

At the iteration number $k \geq 1$, the updating rules for optimizing (3) are

$$Y_k = \arg\min_{\|Y_{(i)}\|_2^2 \leq c} \{F(W_k) + < \nabla F(W_k), Y - W_k > + \frac{1}{2} L \parallel Y - W_k \parallel_F^2\}, \tag{26}$$

$$Z_k = \arg\min_{\|Z_{(i)}\|_2^2 \leq c} (\frac{L}{2} \parallel Z - Z_{k-1} \parallel_F^2) + \tau_k(< \nabla F(W_k), Z >)), \tag{27}$$

$$W_{k+1} = \alpha_k Z_k + (1 - \alpha_k) Y_k, \tag{28}$$

where $i = 1, 2, \cdots, r$, $Y_{(i)}$ is the i-th column of Y and $Z_{(i)}$ is the i-th column of Z. By the Lagrange multiplier method, (26) can be transformed as follows:

$$\begin{aligned} L(Y, \lambda) = & F(W_k) + < \nabla F(W_k), Y - W_k > + \frac{1}{2} L \parallel Y - W_k \parallel_F^2 \\ & - \frac{1}{2} \sum_i^r \lambda_i (c - \parallel Y_{(i)}, \parallel_2^2), \end{aligned} \tag{29}$$

where $\lambda = [\lambda_1, \cdots, \lambda_r]^T \in R^r$. Let (Y, λ) be the optimal solution of (26). The KKT conditions are as follows:

$$\parallel Y_{(i)} \parallel_2^2 \leq c, \lambda_i \geq 0, \lambda_i(\parallel Y_{(i)} \parallel_2^2 - c) = 0, \tag{30}$$

$$\nabla F(W_k)_{(i)} + L(Y_{(i)} - W_{k(i)}) + \sum_i^r \lambda_i Y_{(i)} = 0, \tag{31}$$

where $\nabla F(W_k)_{(i)}$ is the i-th column of $\nabla F(W_k)_{(i)}$ and $W_{k(i)}$ is the i-th of W_k. According to (31), we have

$$Y_{(i)} = \frac{L W_{k(i)} - \nabla F(W_k)_{(i)}}{L + \lambda_i}. \tag{32}$$

However, λ_i is an unknown variable. We find that only $\lambda_i > 0$ and $\parallel Y_{(i)} \parallel_2^2 - c = 0$ can satisfy (30). We substitute (32) into $\parallel Y_{(i)} \parallel_2^2 - c = 0$, and obtain

$$\lambda_i = \max(\frac{\parallel L W_{k(i)} - \nabla F(W_k)_{(i)} \parallel_2}{\sqrt{c}} - L, 0). \tag{33}$$

Similarly to (27), we get

$$Z_{(i)} = \frac{L Z_{k(i)} - \tau_k \nabla F(W_k)_{(i)}}{L + \gamma_i} \tag{34}$$

and

$$\gamma_i = \max(\frac{\parallel L Z_{k(i)} - \tau_k \nabla F(W_k)_{(i)} \parallel_2^2}{\sqrt{c}} - L, 0). \tag{35}$$

According to above analysis, the final updating rules for (3) can be summarized as follows:

$$\lambda_i = \max(\frac{\| LW_{k(i)} - \nabla F(W_k)_{(i)} \|_2^2}{\sqrt{c}} - L, 0), \tag{36}$$

$$Y_{k(i)} = \frac{LW_{k(i)} - \nabla F(W_k)_{(i)}}{L + \lambda_i}, \tag{37}$$

$$\gamma_i = \max(\frac{\| LZ_{k(i)} - \tau_k \nabla F(W_k)_{(i)} \|_2^2}{\sqrt{c}} - L, 0), \tag{38}$$

$$Z_{k(i)} = \frac{LZ_{k(i)} - \tau_k \nabla F(W_k)_{(i)}}{L + \gamma_i}, \tag{39}$$

$$W_{k+1} = \frac{2}{k+3} Z_k + (1 - \frac{2}{k+3}) Y_k, \tag{40}$$

where $L = \| HH^T \|$ and $\nabla F(W) = WHH^T - VH^T$. We summarize above updating rules in Algorithm 2.

Algorithm 2 NAG for CQP (NAGCQP)

Input: V, W, H, c, *outer*
Output: W
Initialization: $Z \leftarrow W$, $L = \| HH^T \|$, $c = \sqrt{c}$
for $k = 0$ to *outer* **do**
1. $\nabla F(W) = WHH^T - VH^T$

for $i = 0$ to r **do**
2. $\lambda = \max(\frac{\|LW_{(i)} - \nabla F(W)_{(i)}\|_2}{c} - L, 0)$
3. $Y_{(i)} = \frac{LW_{(i)} - \nabla F(W)_{(i)}}{L + \lambda}$
4. $\gamma = \max(\frac{\|LZ_{(i)} - \frac{k+1}{2} \nabla F(W)_{(i)}\|_2}{c} - L, 0)$
5. $Z_{(i)} = \frac{LZ_{(i)} - \frac{k+1}{2} \nabla F(W)_{(i)}}{L + \gamma}$

end for
6. $W = \frac{2}{k+3} Z + (1 - \frac{2}{k+3}) Y$.

end for $W = W^{k+1}$

Algorithm 2 accepts input V, W, c and *outer* and outputs W. W and H can be obtained from BCD. *outer* is the iteration number of Algorithm 2. To save space, we do not prove that Algorithm 2 has the convergence rate of $O(\frac{1}{k^2})$.

4. Experiments

In this section, three experiments are presented to evaluate the performances of our proposed algorithms for the COIL20 dataset. The dataset is from Columbia University which can be downloaded on the web site: `http://www1.cs.columbia.edu/CAVE/software/softlib/coil-20.php`. There are 1440 16×16 gray scale images in total, and this dataset includes 20 different classes.

Firstly, Algorithm 1 is evaluated to optimize the L1 function. We compared NAGL1, Feature-sign [16] and MexLasso [17] in the running time, the sparse degree and the objective value. Let $inner = 20$, $outer = 100$ and $\beta = 1$. Table 1 shows the results of different dimensions r by different algorithms. When the number of dimensions is smaller, MexLasso achieves the shorter times, the sparser values and the smaller objective values. With the increase of dimensions, NAGL1 achieves the best performance. However, Feature-sign needs more time to converge than NAGL1 and MexLasso.

Table 1. The running time, the sparseness and the objective value in a time limit of 1500 seconds.

	Time			Sparseness			Objective value		
	r=100	r=500	r=1000	r=100	r=500	r=1000	r=100	r=500	r=1000
NAGL1	4.679	77.937	246.046	0.111	0.127	0.160	129931	96544	57259
MexLasso	1.238	54.228	524.050	0.111	0.127	0.160	129931	96544	57259
Feature-sign	25.797	1024.693	-	0.155	0.088	-	130102	96546	-

Secondly, Algorithm 2 is evaluated to optimize the constrained quadratic programming function. We compared NAGCQP and LagDual [16] in the running time and the objective value. Let $outer = 100$ and $c = 10$. Table 1 shows the results of different dimensions r by different algorithms. When the number of dimensions is smaller, NAGCQP performs as well as LagDual. With the increase of dimensions, LagDual performs better than NAGCQP.

Table 2. The running time and the objective value in a time limit of 1500 seconds.

	Time			Objective value		
	r=100	r=500	r=1000	r=100	r=500	r=1000
NAGCQP	0.596	4.259	10.962	125890	7057	4234
LagDual	0.041	0.487	2.387	125890	7057	2186

Thirdly, our proposed algorithms NAGL1 and NAGCQP have a fast convergence speed. However, it doesn't mean that our proposed algorithm NAGSC (details in Algorithm 3) can learn over-complete bases. NAGSC and ESC [16] are compared to learn over-complete bases. We run each algorithm until the relative error is less than 10^{-6} (i.e., $|F_{new} - F_{old}|/F_{old} < 10^{-6}$). Let $m = 196$, $r = 256$, $n = 10000$, $c = 1$, $\beta = 0.4$, $inner = 10$, $outer = 100$ and $iter = 50$. Figure 1 shows the learned over-complete bases of natural images by each algorithm. NAGSC and ESC take 22.4518 minutes and 55.7725 minutes in learning 256 bases, respectively.

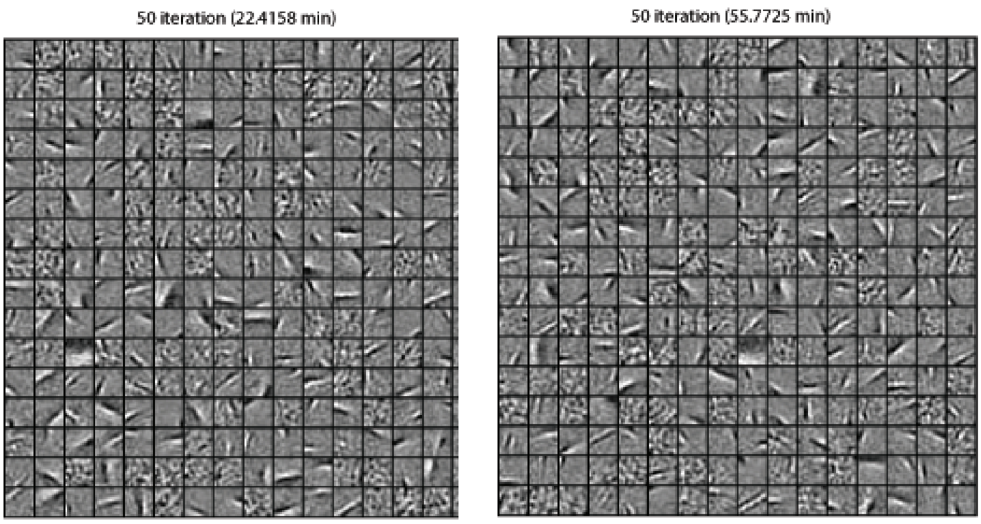
5. Conclusion and future work

In this paper, Sparse Coding is formulated by L1 and L2 problems and we present algorithms for each sub-problem: NAG for solving L1 and L2 problems.

Algorithm 3 NAGSC

Input: V, W, H, c, β, *outer*, *iter*
Output: W, H
for $k = 0$ to *iter* **do**
1. W=NAGLCQP(V, W, H, c, *outer*);
2. H=NAGL1(V, W, H, β, *outer*);

end for



NAGSC ESC

Figure 1. Learned over-complete natural image bases

Experiments show that our proposed algorithms learn over-complete bases more quickly.

Two topics should be discussed in future work:

- There are many different sparse coding models, hence, the corresponding NAG should be discussed to optimize them.
- The Lipshcitz constant L in algorithm 1 and 2 is fixed, therefore, a variable Lipshcitz constant L should be discussed.

Acknowledgments

This work is supported by the Research Foundation of Chongqing Municipal Education Commission (KJ1710253), Foundation of Chongqing Municipal Key Laboratory of Institutions of Higher Education ([2017]3), Foundation of Chongqing Development and Reform Commission (2017[1007]), Foundation of

Wanzhou Development and Reform Commission ([2017]32), and Foundation of Chongqing Three Gorges University.

References

[1] B. A. Olshausen, D. J. Field, *Emergence of simple-cell receptive field properties by learning a sparse code for natural images*, Nature, 381 (1996), 607-609.

[2] B. A. Olshausen, D. J. Field, *Sparse coding with an overcomplete basis set: a strategy employed by v1?*, Vision Research, 37 (1997), 3311-3325.

[3] M. Turk, A. P. Pentland, *Face recognition using eigenface*, IEEE Conf. Computer Vision and Pattern Recognition, 118 (1991), 586-591.

[4] X. He, P. Niyogi, *Locality preserving projections*, Advances in Neural Information Processing Systems, 16 (2003), 186-197.

[5] P. N. Belhumeur, J. P. Hespanha, D. J. Kriegman, *Eigenfaces vs. fisherfaces: recognition using class specific linear projection*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 19 (2002), 711-720.

[6] K. Labusch, E. Barth, T. Martinetz, *Simple method for high-performance digit recognition based on sparse coding*, IEEE Trans Neural Netw, 19 (2008), 1985-1989.

[7] B. He, D. Xu, N. Rui, M. V. Heeswijk, Q. Yu, Y. Miche, A. Lendasse, *Fast face recognition via sparse coding and extreme learning machine*, Cognitive Computation, 6 (2014), 264-277.

[8] M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu, D. Cai, *Graph regularized sparse coding for image representation*, IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society, 20 (2011), 1327-1336.

[9] Z. Shu, C. Zhao, P. Huang, *Constrained sparse concept coding algorithm with application to image representation*, Ksii Transactions on Internet and Information Systems, 8 (2014), 3211-3230.

[10] L. Andrs, P. Zsolt, S. Gbor, *Efficient sparse coding in early sensory processing: Lessons from signal recovery*, Plos Computational Biology, 8 (2012), e1002372.

[11] Y. Zhao, Z. Liu, Y. Wang, H. Wu, S. Ding, *Sparse coding algorithm with negentropy and weighted 1-norm for signal reconstruction*, Entropy, 19 (2017), 599.

[12] Y. Nesterov, *Smooth minimization of non-smooth functions*, Mathematical Programming, 103 (2005), 127-152.

[13] D. P. Bertsekas, *Nonlinear programming*, Athena scientific Belmont, 1999.

[14] N. Guan, D. Tao, Z. Luo, B. Yuan, *Nenmf: an optimal gradient method for nonnegative matrix factorization*, IEEE Transactions on Signal Processing, 60 (2012), 2882-2898.

[15] M. Baes, *Estimate sequence methods: extensions and approximations*, Institute for Operations Research, 2009.

[16] H. Lee, A. Battle, R. Raina, A. Y. Ng, *Efficient sparse coding algorithms. nips*, Proc of Nips, 19 (2007), 801-808.

[17] J. Mairal, F. Bach, J. Ponce, G. Sapiro, *Online learning for matrix factorization and sparse coding*, Journal of Machine Learning Research, 11 (2009), 19-60.

Accepted: 28.05.2018