DALM-SVD: ACCELERATED SPARSE CODING THROUGH SINGULAR VALUE DECOMPOSITION OF THE DICTIONARY

Hugo Gonçalves[1,2] *Miguel Correia*[2] *Xin Li*[1] *Aswin Sankaranarayanan*[1] *Vitor Tavares*[2]

Carnegie Mellon University[1]
Faculdade de Engenharia da Universidade do Porto[2]

ABSTRACT

Sparse coding techniques have seen an increasing range of applications in recent years, especially in the area of image processing. In particular, sparse coding using ℓ_1-regularization has been efficiently solved with the Augmented Lagrangian (AL) applied to its dual formulation (DALM). This paper proposes the decomposition of the dictionary matrix in its Singular Value/Vector form in order to simplify and speed-up the implementation of the DALM algorithm. Furthermore, we propose an update rule for the penalty parameter used in AL methods that improves the convergence rate. The SVD of the dictionary matrix is done as a pre-processing step prior to the sparse coding, and thus the method is better suited for applications where the same dictionary is reused for several sparse recovery steps, such as block image processing.

1. INTRODUCTION

Sparse coding techniques have seen an increasing range of applications in recent years, especially in the area of image processing. Many image processing problems, such as denoising [1], deblurring [2], inpainting [3], classification [4] and others [5,6], have been approached successfully with this formulation. This technique aims to build a signal $\mathbf{y}$, such as an image or a block of an image, as a transformation of a sparse signal $\mathbf{x}$, of which most coefficients are zero, by a dictionary matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$. The fidelity of the representation is measured by the reconstruction error $\|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2$, while its sparsity is measured by a function $\Omega(\mathbf{x})$. The problem can be stated as:

$$\min_{\mathbf{x}} f(\mathbf{x}) = \frac{1}{2}\|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2^2 + \Omega(\mathbf{x}). \quad (1)$$

The regularizing function $\Omega(\mathbf{x})$ often makes the solution unique when the dictionary is overcomplete ($\mathbf{A} \in \mathbb{R}^{m\times n}$, $m < n$). A common form of regularization is the ℓ_1 norm, restating the original problem as

$$\min_{\mathbf{x}} f_{\ell_1}(\mathbf{x}) = \frac{1}{2}\|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1, \quad (2)$$

where λ is a parameter that defines the balance between the reconstruction error and the sparsity of $\mathbf{x}$. Although it does not strictly measures sparsity (number of non-zero elements), the ℓ_1 norm is the convex relaxation of the sparsity of the vector and is a widely used replacement.

Many methods have been developed to solve (2) [7–13], among which the Dual Augmented Lagrangian (DAL) method [14] is currently one of the best in terms of processing time [15, 16]. The DAL method solves the dual formulation of problem (2) by applying the Augmented Lagrangian (AL) method to it (described in section 2). The AL step involves updating more than one variable by solving simpler optimization problems. Two ways in the literature have been used to solve the AL iterations: a) *per iteration*, apply several Newton steps to update the dual variable until a criterion of convergence is met (the calculation of the inverse of the Hessian is required for each inner iteration) [14]; or b) following the Alternating Direction algorithm, use values from previous iterations to approximate to quadratic the expression that updates the dual variable (this option avoids calculation of the inverse of the Hessian at each iteration at the cost of a fixed penalty parameter and more iterations to converge) [17]. This paper proposes an adaptation of option b) based on Singular Value Decomposition (SVD) of the dictionary matrix that makes the Hessian diagonal and consequently, allows changes of the penalty parameter. An update rule of the penalty parameter is suggested that performs better than any constant value. Given that the SVD precomputation is done only once, the overall processing time is reduced.

2. BACKGROUND

The DAL methods were proposed in [14, 17] and thoroughly evaluated in [15, 16]. This section reviews both versions. The method follows from solving the dual problem instead of the primal. Duality theory applied to (2) w.r.t. variable $\mathbf{A}\mathbf{x}$ results in the following dual problem

$$\max_{\alpha} f^{\star}_{\ell_1}(\alpha) = -\frac{1}{2}\|\alpha\|_2^2 + \alpha^T\mathbf{y} \quad \text{s.t. } \left\|\mathbf{A}^T\alpha\right\|_\infty \le \lambda. \quad (3)$$

Adding an auxiliary variable ν such that $\nu = \mathbf{A}^T\alpha$, the constraint above is replaced by $\|\nu\|_\infty \le \lambda$, and the problem can be solved by the Augmented Lagrangian method:

$$\max_{\alpha,\nu} L_\eta(\alpha,\nu,\mathbf{x}) = \max_{\alpha}\left\{-\frac{1}{2}\|\alpha\|_2^2 + \alpha^T\mathbf{y} \right. \quad (4)$$

$$\left. + \quad \max_{\nu}\left\{\mathbf{x}^T\left(\nu - \mathbf{A}^T\alpha\right) - \frac{\eta}{2}\left\|\nu - \mathbf{A}^T\alpha\right\|_2^2 - \delta_\lambda(\nu)\right\}\right\}$$

where η is a penalty parameter and $\delta_\lambda(\nu)$ is $+\infty$ if any element of ν is greater than λ and 0 otherwise. The solution is found by iteratively updating the three variables: $\nu, \alpha, \mathbf{x}$.

ν is updated by equating to zero the derivate of the Lagrangian w.r.t. ν, resulting in:

$$\begin{aligned} \nabla_\nu L_\eta(\alpha,\nu,\mathbf{x}) &= \mathbf{x} - \eta\left(\nu - \mathbf{A}^T\alpha\right) - \delta_\lambda(\nu) = 0 \\ \nu(\alpha) &= P_\lambda^\infty\left(\frac{\mathbf{x}}{\eta} + \mathbf{A}^T\alpha\right), \end{aligned} \quad (5)$$

where $P_\lambda^\infty(x)$ is an element-wise projection to the range $[-\lambda,\lambda]$.

In the same way, α is updated by equating to zero the derivate of the Lagrangian w.r.t. α, resulting in:

$$\begin{aligned} & \nabla_\alpha L_\eta(\alpha,\nu,\mathbf{x}) \quad (6) \\ =\; & -\alpha + \mathbf{y} - \mathbf{A}\mathbf{x} + \eta\mathbf{A}\left(\nu(\alpha) - \mathbf{A}^T\alpha\right) \\ =\; & -\alpha + \mathbf{y} - \mathbf{A}\left(\mathbf{x} + \eta\mathbf{A}^T\alpha - P_{\lambda\eta}^\infty\left(\mathbf{x} + \eta\mathbf{A}^T\alpha\right)\right) \\ =\; & -\alpha + \mathbf{y} - \mathbf{A}\cdot \text{shrink}_{\eta\lambda}(\mathbf{x} + \eta\mathbf{A}^T\alpha) = 0, \end{aligned}$$

where $\text{shrink}_\lambda(x) = \max(|x| - \lambda, 0)\cdot\text{sign}(x)$ is an element-wise operator. Here we can take two approaches: either a) keep ν as function of α [14] or b) take the last value of α to calculate ν and consider it constant when updating α [17].

Option a) is the original version of the DAL method and updates α through a Newton step with Hessian:

$$\nabla_\alpha^2 L_\eta(\alpha,\nu,\mathbf{x}) = -\left(\mathbf{I} + \eta\mathbf{A}_\Omega\mathbf{A}_\Omega{}^T\right), \quad (7)$$

where $\mathbf{A}_\Omega$ is the subset of columns i of $\mathbf{A}$ where $x_i \neq 0$. This approach requires very few iterations, however, for every update of α, the Hessian and its inverse needs to be recomputed.

Option b) makes an approximation of ν using α of the previous iteration and (6) solved for α leads to

$$\alpha \;=\; \left(\mathbf{I} + \eta\mathbf{A}\mathbf{A}^T\right)^{-1}\left(\mathbf{y} - \mathbf{A}\left(\mathbf{x} - \eta\nu\right)\right). \quad (8)$$

If η is kept fixed and the inverse of the Hessian is precomputed, each iteration of this version amounts to two matrix-vector multiplications, although it runs more of them.

The dual variable of the dual problem (which is the primal variable) is updated according to the AL method

$$\mathbf{x}_t = \text{shrink}_{\lambda\eta}\left(\mathbf{x}_{t-1} + \eta\mathbf{A}^T\alpha\right) \quad (9)$$

In both versions, updating the penalty parameter η wisely can improve the convergence rate. Recalling the α update, we have option a) that allows η to be updated, but needs to find the inverse of the Hessian at every iteration, or option b) to precompute the Hessian, but keep η fixed as a result. The main point of this paper is to suggest an alternative that not only allows to free the penalty parameter, but also makes each iteration faster than option b). A precomputation based on Singular Value Decomposition is required. Thankfully, in many applications, the dictionary is reused and this computation needs to be *done only once*. In the sequel, we only consider option (b) of the DAL method.

3. PROPOSED METHOD

The dictionary matrix can be decomposed in its singular values/vectors, as in $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, where the economic version of the SVD has $\mathbf{U} \in \mathbb{R}^{m\times m}$, $\mathbf{\Sigma} \in \mathbb{R}^{m\times m}$, $\mathbf{V}^T \in \mathbb{R}^{m\times n}$. Here, $\mathbf{U}\mathbf{U}^T = \mathbf{U}^T\mathbf{U} = \mathbf{I}$ and $\mathbf{V}^T\mathbf{V} = \mathbf{I}$, however $\mathbf{V}\mathbf{V}^T \neq \mathbf{I}$. By applying SVD decomposition to $\mathbf{A}$ in (8), we obtain

$$\mathbf{U}\mathbf{\Sigma}\left(-\left(\mathbf{\Sigma}^{-1} + \eta\mathbf{\Sigma}\right)\mathbf{U}^T\alpha + \underbrace{\mathbf{\Sigma}^{-1}\mathbf{U}^T\mathbf{y}}_{\mathbf{y}'} - \underbrace{\mathbf{V}^T}_{\mathbf{A}'}\left(\mathbf{x} - \eta\nu\right)\right) = \mathbf{0} \quad (10)$$

and recognizing that the null spaces of $\mathbf{U}$ and Σ are both $\{0\}$, we can drop the premultiplication by $\mathbf{U}\mathbf{\Sigma}$. Then, from (10):

$$\underbrace{\mathbf{\Sigma}\mathbf{U}^T\alpha}_{\alpha'} = \left(\mathbf{\Sigma}^{-2} + \eta\mathbf{I}\right)^{-1}\left(\mathbf{y}' - \mathbf{A}'\left(\mathbf{x} - \eta\nu\right)\right). \quad (11)$$

Finally, note that α is only used in each iteration by the operator $\mathbf{A}^T$. Therefore:

$$\mathbf{A}^T\alpha = \mathbf{V}\mathbf{\Sigma}\mathbf{U}^T\alpha = \mathbf{V}\alpha' = \mathbf{A}'^T\alpha'.$$

Concerning the updates of the remaining variables, the only change is the equivalence between $\mathbf{A}^T\alpha$ and $\mathbf{A}'^T\alpha'$.

The decomposition of $\mathbf{A}$ is done prior to the AL method. The advantage of working with the decomposed matrix is that the term to be inverted during the α update becomes diagonal. This has two benefits: the inverse is easily computed and multiplied and, as a consequence, the penalty parameter can be updated at each iteration.

3.1. Update Rule of Penalty Parameter

Experiments done with fixed penalty parameters show that a smaller value slows down convergence at the beginning but speeds up in latter iterations, while a larger value produces fast converging initial iterations but it tends to slow down after some point. Depending on the tolerance of the stopping criterion, the slowing down in the end may or may not be a problem.

Therefore, given the newly obtained freedom of the penalty parameter, it seems fitting to change η proportional to a term that tends to decrease throughout iterations. The proposed update rule in this paper is:

$$\eta = \frac{\|\mathbf{y}' - \mathbf{A}'\mathbf{x}\|_2}{\lambda}. \tag{12}$$

3.2. Practical Implementation

In practice, the algorithm has some room for mathematical manipulation. Note that, as is, the term $\mathbf{A}'\mathbf{x}$ needs to be computed only for the update rule, since the term $\mathbf{A}'(\mathbf{x} - \eta\nu)$ is calculated with a single matrix-vector multiplication. If we form a variable $\mathbf{z}$

$$\begin{aligned} \mathbf{z} &= \text{shrink}_{\eta\lambda}(\mathbf{x} + \eta\mathbf{A}'^T\alpha'_{t-1}) \\ &= \left(\mathbf{x} + \eta\mathbf{A}'^T\alpha'_{t-1}\right) - P^{\infty}_{\lambda\eta}\left(\mathbf{x} + \eta\mathbf{A}'^T\alpha'_{t-1}\right) \\ &= \mathbf{x} - \eta\nu + \eta\mathbf{A}'^T\alpha'_{t-1} \end{aligned} \tag{13}$$

and replace $\mathbf{x} - \eta\nu = \mathbf{z} - \eta\mathbf{A}'^T\alpha'_{t-1}$ in (11), the α' update becomes

$$\begin{aligned} \alpha' &= \left(\mathbf{\Sigma}^{-2} + \eta\mathbf{I}\right)^{-1}\left(\mathbf{y}' - \mathbf{A}'\left(\mathbf{z} - \eta\mathbf{A}'^T\alpha'_{t-1}\right)\right) \\ &= \left(\mathbf{\Sigma}^{-2} + \eta\mathbf{I}\right)^{-1}\left(\eta\alpha'_{t-1} + (\mathbf{y}' - \mathbf{A}'\mathbf{z})\right). \end{aligned} \tag{14}$$

Since $\mathbf{z}$ is a prediction of $\mathbf{x}$, the term $\mathbf{y}' - \mathbf{A}'\mathbf{x}$ can be replaced by $\mathbf{y}' - \mathbf{A}'\mathbf{z}$ in (12) to avoid the operation $\mathbf{A}'\mathbf{x}$. In fact, $\mathbf{x}$ and $\mathbf{z}$ tend to the same value as $\mathbf{x}$ tends to the true solution.

4. RESULTS

At the optimal solution, the duality gap, the difference between the primal and dual problems, is zero. Therefore, in practice, the algorithm is deemed finished when the relative duality gap is below a tolerance ϵ:

$$\frac{f(\mathbf{x}) - f^\star(\alpha)}{f(\mathbf{x})} < \epsilon. \tag{15}$$

The following results are shown for $\epsilon = 10^{-3}$, $\lambda = 10^{-4}$, $\|\mathbf{x}\|_0 = 50$, $m = 256$, $n = 512$, elements of $\mathbf{A}$ and $\mathbf{x}$ extracted from a Normal distribution and the atoms of $\mathbf{A}$ are unit norm columns.

Fig. 1 shows the result of an experiment where problem (2) was solved with the introduced DAL methods, using adapted versions of the original codes, as well as DALM SVD using the update rule (12). The experiment shows that option a) requires a very few number of outer iterations, but each iteration is expensive. Regarding option b), a fixed η either slows down in the beginning or the end of the minimization problem. However, the proposed update rule is able to avoid the disadvantage seen for fixed penalty parameters. Similar plots persist for different levels of sparsity and values of λ.

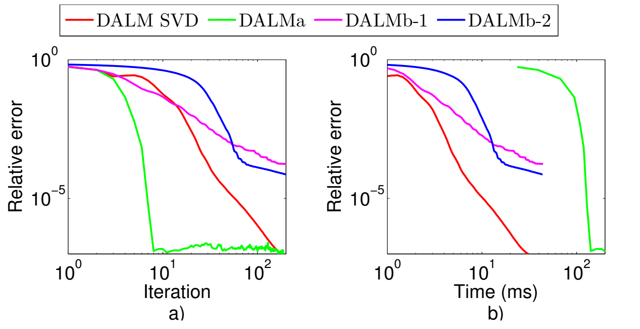


Fig. 1: Relative error $\|\hat{\mathbf{x}} - \mathbf{x}_{opt}\|_2 / \|\mathbf{x}_{opt}\|_2$, $\mathbf{y} = \mathbf{A}\mathbf{x}$, $\mathbf{x}_{opt}$ the optimal solution, averaged over 50 trials a) per iteration and b) against time, for problem (2) solved by different DAL methods. Each algorithm is run for 200 iterations, resulting in variable processing time. DALMa starts with $\eta = 0.1/\lambda$ and doubles it at each outer iteration (as in [15]), DALMb has a fixed η value (as in [16]) of $\eta = \|\mathbf{y}\|_1/m/\lambda$ for DALMb-1 and $\eta = 0.1\|\mathbf{y}\|_1/m/\lambda$ for DALMb-2, and DALM SVD updates η according to (12). Elements of $\mathbf{A}$ and $\mathbf{x}$ extracted from Normal random distribution.

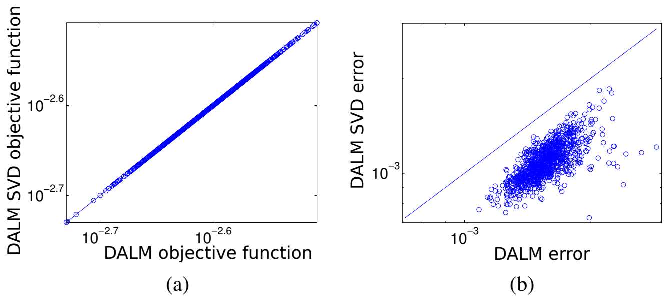
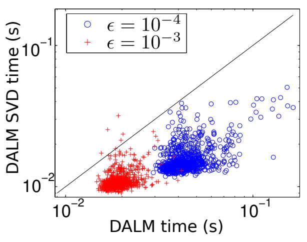


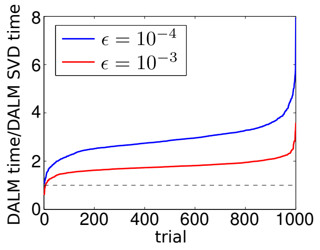
Fig. 2: a) Minimum objective function value and b) solution relative error of the DAL SVD method against the DAL method, for 1000 trials.

To confirm that the proposed method solves (2) faster, the tests evaluate the performance of the method not only per iteration, but also accounting for the setup time. The first test validates that both DALM and DALM SVD converge to the same solution. Fig. 2 shows that both algorithms converge to similar objective function values and solution errors for 1000 trials. The second test measures the time each algorithm takes to compute each solution. In this case, time is a fair comparison since both algorithms are run in Matlab and have very similar codes. As shown in Fig. 3a, most trials are in the lower-right triangle, which means the DAL SVD method is generally faster. As explained above, the slow convergence near the true solution for a fixed penalty parameter becomes a problem when tolerances are low. For $\epsilon = 10^{-4}$, the speedup of DALM SVD is larger compared to $\epsilon = 10^{-3}$, which can be seen in Fig. 3b.

DALM needs to precompute the inverse of the Hessian,



(a)



(b)

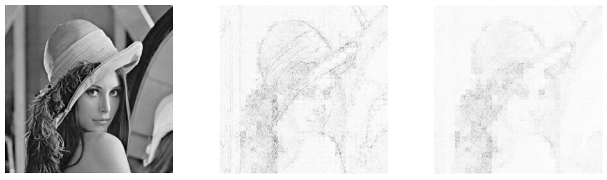
Fig. 3: Two ways of representing the time each algorithm takes to solve problem (2). a) shows the DALM SVD time against the DALM time and b) shows the speedup ratio with DALM SVD for each trial, in ascending order. Red crosses are results for $\epsilon = 10^{-3}$ and blue circles for $\epsilon = 10^{-4}$. The slow convergence near the true solution for a fixed penalty parameter can become a problem when tolerances are low.

Times	DALM	DALM SVD
Setup Time (s)	0.0079	0.0555
Processing Time (s)	0.0449	0.0158
Overall time (s)	46	16.25

Table 1: Average of setup time, processing time (single minimization problem) and Overall time (setup time + 1024 minimization problems) for $\epsilon = 10^{-4}$, $m = 256$, $n = 512$.

while the DALM SVD has to precompute the Singular Value Decomposition. Both have $\mathcal{O}(n^3)$ complexity, but SVD has a higher multiplicative factor. Assuming that the setup is done once, but the number of problems to solve (such as an image divided in patches) increases, the overall time of DALM SVD soon becomes shorter than the standard approach, even with a higher setup time. As an application example, consider the sparse representation of a 512×512 image sliced in 1024 patches of size 16×16 with a dictionary with 512 atoms and $\epsilon = 10^{-4}$. The setup is done only once while the sparse coding step is performed 1024 times. The resulting overall time for DALM is 46 s and for DALM SVD is 16.25 s. Table 1 reports the decomposition of the overall time in setup time and processing time (of each minimization problem).

Real-time applications have a time budget that cannot be exceeded. Therefore, a different example would be to evaluate the accuracy of the solution having a limited time to compute it. As referred in the Introduction, sparse recovery as proven itself a useful technique in the image processing area. Therefore, the next example is used for the purpose of comparing performance between DALM implementations. It is a compressive sensing application, where an image is divided in 16×16 blocks and each block is sensed by the same matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$ with random gaussian entries, $n = 256$ and $m = 128$. An estimate $\hat{\mathbf{y}}$ of each block $\mathbf{y}$ is then found by solving problem:



(a) (b) (c)

Fig. 4: a) Original Lena image. Normalized (black = largest pixel error, white = no pixel error) reconstruction error of 16×16 blocks with a 4 ms time budget per block done with b) DALMb with $\eta = \|\mathbf{y}\|_1 / m / \lambda$ and c) DALM SVD.

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{My} - \mathbf{MDx}\|_2^2 + \lambda \|\mathbf{x}\|_1 \quad (16)$$

$$\hat{\mathbf{y}} = \mathbf{D}\hat{\mathbf{x}} \quad (17)$$

knowing that each block $\mathbf{y}$ has a sparse representation in the dictionary $\mathbf{D}$. We use a $n \times n$ DCT matrix as the dictionary $\mathbf{D}$. In practice, the reconstruction will be free of patterns not matched by the dictionary, such as noise (denoising) or missing pixels (inpainting). The comparison is done between DALM and DALM SVD with automatic update, $\lambda = 0.1$ and a time budget of 4 ms per block, thus forcing the runtime of both methods to be equal. Fig. 4 shows the resulting images with the two methods. The PSNR[1] of the DALM reconstruction is 44.76 dB and of the DALM SVD is 47.95 dB.

5. CONCLUSION

The DAL method has been improved by decomposing the dictionary in its singular value/vector matrices. The decomposition allows the inverse involved in the update of the dual variable to be easily computed, since the matrix to be inverted is diagonal. As a consequence, the penalty parameter of the Augmented Lagrangian method, which is part of the inverse, can be updated in each iteration. Depending on the tolerance of the duality gap for the stopping criterion, the speed up can be significant. Furthermore, values close to the minimum are reached in less iterations, which makes it more suitable for real-time applications where a time budget exists for each coding step.

Acknowledgments

This work was funded by Fundação para a Ciência e Tecnologia scholarship SFRH / BD / 51435 / 2011 and the Carnegie Mellon Portugal Program .

[1]PSNR defined as $20 \log_{10} \left(\max(I) / \sqrt{\sum_{i=1}^{N} (I_i - J_i)^2 / N} \right)$, where I and J are the vectorized original and reconstructed images respectively and N is the number of pixels.

6. REFERENCES

[1] M. Elad and M. Aharon, "Image Denoising Via Learned Dictionaries and Sparse representation.," in *CVPR (1)*, 2006, pp. 895–900.

[2] F. Couzinie-Devy, J. Mairal, F. Bach, and J. Ponce, "Dictionary Learning for Deblurring and Digital Zoom," *CoRR*, vol. abs/1110.0957, 2011.

[3] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online Learning for Matrix Factorization and Sparse Coding.," *Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

[4] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Discriminative learned dictionaries for local image analysis.," in *CVPR*, 2008.

[5] J. Mairal, M. Leordeanu, F. Bach, M. Hebert, and J. Ponce, "Discriminative Sparse Image Models for Class-Specific Edge Detection and Image Interpretation.," in *ECCV (3)*, 2008, pp. 43–56.

[6] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution as sparse representation of raw image patches.," in *CVPR*, 2008.

[7] M. R. Osborne, B. Presnell, and B. A. Turlach, "A new approach to variable selection in least squares problems," *IMA journal of numerical analysis*, vol. 20, no. 3, pp. 389, 2000.

[8] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least Angle Regression," *The Annals of Statistics*, vol. 32, no. 2, pp. 407–451, 2004.

[9] H. Lee, A. Battle, R. Raina, and A. Y Ng, "Efficient sparse coding algorithms," *Advances in neural information processing systems*, vol. 19, pp. 801, 2007.

[10] S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo, "Sparse reconstruction by separable approximation.," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2479–2493, 2009.

[11] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[12] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proceedings of the National Academy of Sciences*, vol. 106, no. 45, pp. 18914–18919, 2009.

[13] S. Becker, J. Bobin, and E. J. Candès, "NESTA: A Fast and Accurate First-Order Method for Sparse Recovery.," *SIAM J. Imaging Sciences*, vol. 4, no. 1, pp. 1–39, 2011.

[14] R. Tomioka and M. Sugiyama, "Dual-augmented lagrangian method for efficient sparse reconstruction," *IEEE Signal Processing Letters*, vol. 16, no. 12, pp. 1067–1070, 2009.

[15] R. Tomioka, T. Suzuki, and M. Sugiyama, "Super-Linear Convergence of Dual Augmented Lagrangian Algorithm for Sparsity Regularized Estimation.," *Journal of Machine Learning Research*, vol. 12, pp. 1537–1586, 2011.

[16] A. Y. Yang, Z. Zhou, A. G. Balasubramanian, S. S. Sastry, and Y. Ma, "Fast l1-minimization algorithms for robust face recognition," *IEEE Transactions on Image Processing*, vol. 22, no. 8, pp. 3234–3246, 2013.

[17] J. Yang and Y. Zhang, "Alternating direction algorithms for l1-problems in compressive sensing," *SIAM J. Sci. Comput.*, vol. 33, no. 1, pp. 250–278, 2011.