

Article

A Fast Sparse Recovery Algorithm for Compressed Sensing Using Approximate l_0 Norm and Modified Newton Method

Dingfei Jin ¹, Yue Yang ^{1,2,*}, Tao Ge ³ and Daole Wu ²

¹ School of Traffic and Transportation Engineering, Central South University, Changsha 410075, China; dr.kin@csu.edu.cn

² The State Key Laboratory of Heavy Duty AC Drive Electric Locomotive Systems Integration, CRRC Zhuzhou Locomotive Co., Ltd., Zhuzhou 412001, China; zdhwdl0722@163.com

³ China Mobile (Suzhou) Software Technology Co., Ltd., Suzhou 215004, China; getao@cmss.chinamobile.com

* Correspondence: yangyue@csu.edu.cn

Received: 11 March 2019; Accepted: 11 April 2019; Published: 15 April 2019



Abstract: In this paper, we propose a fast sparse recovery algorithm based on the approximate l_0 norm (FAL0), which is helpful in improving the practicability of the compressed sensing theory. We adopt a simple function that is continuous and differentiable to approximate the l_0 norm. With the aim of minimizing the l_0 norm, we derive a sparse recovery algorithm using the modified Newton method. In addition, we neglect the zero elements in the process of computing, which greatly reduces the amount of computation. In a computer simulation experiment, we test the image denoising and signal recovery performance of the different sparse recovery algorithms. The results show that the convergence rate of this method is faster, and it achieves nearly the same accuracy as other algorithms, improving the signal recovery efficiency under the same conditions.

Keywords: sparse recovery; compressed sensing; approximate l_0 norm; modified Newton method

1. Introduction

Compressed sensing (CS) [1–4] is a theory proposed in recent years. Based on this theory, the original signal can be recovered with sampling values much lower than the Nyquist sampling rate. Assuming that a $m \times 1$, signal x will be recovered from $y = Ax$, where $x \in R^n$, $y \in R^m$, and $A \in R^{m \times n}$. If $m < n$, recovery of x from $y = Ax$ is ill-posed. However, if the x is K ($K \leq m$) sparse, that is, only K elements in x are not 0, then x can be calculated. A , y , and K are called measurement matrix, measurement vector, and sparsity level, respectively.

The CS theory consists of two parts. The first is the sparse representation. Because a natural signal is not sparse in the usual case, we need to first make it sparse by mathematical transformation. The second is sparse recovery, which means to recover the sparse signal. Among the two parts, the sparse recovery method directly determines the quality of signal recovery. In this paper, we focus on proposing a sparse recovery algorithm for CS. In the following sections, signal x is set to be a sparse signal vector.

Generally, all sparse recovery methods are devoted to solve Equation (1) [1,3,5], where $\|x\|_0$ denotes the number of nonzero elements in x . Equation (1) is aimed at seeking the minimal $\|x\|_0$ solution of the underdetermined linear system of equations $y = Ax$ with known A and y .

$$\begin{cases} \min \|x\|_0 \\ \text{s.t. } y = Ax \end{cases} \quad (1)$$

Clearly, the direct l_0 norm minimization problem is NP-hard [6–8], so there are many proposed algorithms to approximate its solution. In general, these methods can be classified into three categories: (1) convex relaxation method (CR) [9,10], (2) greedy algorithm (GA) [11,12], and (3) approximate l_0 norm algorithm (AL0) [13,14].

The CR methods, also called l_p norm minimization, use the l_p norm to approximate the l_0 norm, which can be shown as Equation (2):

$$\begin{cases} \min \|x\|_p^p \\ \text{s.t. } y = Ax \end{cases} \quad (2)$$

where $\|x\|_p = (\sum_{i=1}^N |x_i|^p)^{1/p}$, and x_i is the element in x . In particular, when $p = 1$, this method is called basis pursuit (BP) algorithm. The performance of this algorithm is stable. However, because it is solved by linear programming without solving the linear equations directly, its computational cost is very high [15].

The GA algorithms, such as matching pursuit [12] (MP) and orthogonal matching pursuit [11] (OMP), do not always guarantee a global optimal solution [5].

The AL0 adopts a smoothed function with parameters to approximate the l_0 norm of x . Thus, the l_0 norm minimization problem can be converted into the smoothed function minimization problem, which is called the recovery algorithm using the approximate l_0 norm. For instance, Mohimani [14] adopted the following equation to approximate the l_0 norm:

$$f(x_i) = \exp\left(-\frac{x_i^2}{2\delta^2}\right) \quad (3)$$

where δ is a positive number approaching 0. Then, the minimization problem can be converted to the following equation:

$$\begin{cases} \min \sum_{i=1}^N \exp\left(-\frac{x_i^2}{2\delta^2}\right) \\ \text{s.t. } y = Ax \end{cases} \quad (4)$$

Algorithms of this category directly solve the linear equations in the process of iteration and do not need to use linear programming. Therefore, this algorithm has fast convergence speed and guarantees global optimum. This algorithm solves the defects of CR and GA to a certain degree and has gradually become a popular sparse recovery algorithm, but it cannot avoid the jaggies in the iterative process. That is to say, the convergence direction does not always follow the descending direction of the gradient.

Based in the idea of AL0, we adopt a simple fractional function to approximate the l_0 norm, and, to avoid jaggies in the iterative process, we propose a modified Newton method, which can make the algorithm converge faster. In addition, we neglect the zero elements in the process of computing, which further reduces the amount of computation. Finally, we compare the performance of the proposed algorithm with several typical sparse recovery algorithms in the field of signal recovery and image processing to prove the advantages of FAL0.

2. Fast Sparse Recovery Algorithm Using Approximate l_0 Norm

In this section, we first adopt a simple function to approximate the l_0 norm and then derive the process of solving this problem in detail. In addition, we summarize the procedure of the algorithm and analyze its computational complexity.

2.1. Implementation of the Algorithm

We adopt Equation (5) to substitute $\|x\|_0$

$$f_\delta(x_i) = \frac{\delta}{\delta + x_i} \quad (5)$$

where δ is a positive parameter and, obviously,

$$\lim_{\delta \rightarrow 0} f_\delta(x_i) = \begin{cases} 1, & x_i = 0 \\ 0, & x_i \neq 0 \end{cases} \quad (6)$$

Then, we consider

$$\begin{cases} \max \sum_{i=1}^N f_\delta(x_i) \\ \text{s.t. } \mathbf{y} = \mathbf{A}\mathbf{x} \end{cases} \quad (7)$$

When δ approaches 0, the optimal solutions of Equation (7) and Equation (1) are same. This problem is an optimization problem with an equality constraint. The Lagrange multiplier method can transform such problems into unconstrained optimization problems, thus avoiding solving linear programming and greatly reducing the computation cost. The detailed process of this method is as follows:

$$\max L(\mathbf{x}) = \|\mathbf{Ax} - \mathbf{y}\|_2^2 + \lambda \sum_{i=1}^N f_\delta(x_i) \quad (8)$$

where λ is the regularization parameter, which is used to adjust the weight of the two terms in Equation (8) and can be chosen by the method of generalized cross-validation (GCV) [16]. Here, we adopt the Newton method to solve Equation (8), and the convergence direction (Newton direction) \mathbf{d} is shown by Equation (9).

$$\mathbf{d} = -[\nabla^2 L(\mathbf{x})]^{-1} \nabla L(\mathbf{x}) \quad (9)$$

where $[\nabla^2 L(\mathbf{x})]^{-1}$ is the inverse matrix of $[\nabla^2 L(\mathbf{x})]$. In Equation (9)

$$\nabla L(\mathbf{x}) = \frac{\partial L}{\partial \mathbf{x}} = 2\mathbf{A}^T(\mathbf{Ax} - \mathbf{y}) + \lambda \left(-\frac{\partial f}{\partial x_1}, -\frac{\partial f}{\partial x_2}, \dots, -\frac{\partial f}{\partial x_N} \right)^T \quad (10)$$

$$\left(-\frac{\partial f}{\partial x_1}, -\frac{\partial f}{\partial x_2}, \dots, -\frac{\partial f}{\partial x_N} \right)^T = \left(-\frac{\delta}{(\delta + x_1)^2}, -\frac{\delta}{(\delta + x_2)^2}, \dots, -\frac{\delta}{(\delta + x_N)^2} \right)^T \quad (11)$$

$$\begin{aligned} \nabla^2 L(\mathbf{x}) &= \frac{\partial(\frac{\partial L}{\partial \mathbf{x}})}{\partial \mathbf{x}} = \frac{\partial^2 L}{\partial \mathbf{x}^2} = 2\mathbf{A}^T \mathbf{A} + \lambda \text{diag}\left(\frac{\partial^2 f}{\partial x_i^2}\right) \\ &= 2\mathbf{A}^T \mathbf{A} + \lambda \text{diag}\left(\frac{2\delta}{(\delta + x_i)^3}\right) \end{aligned} \quad (12)$$

where \mathbf{A}^T is the transpose of \mathbf{A} .

Clearly, the Hessian matrix $\nabla^2 L(\mathbf{x})$ is not always positive definite, which often leads to “jaggies” in the iterative process. This means that the Newton direction is not always gradient descending. A common means to deal with such a problem is to find an approximate matrix that is positive definite to replace the Hessian matrix. The basic idea of this method is to ensure that the objective function converges to the optimal direction regardless of the speed of convergence. This method is called modified Newton method, and the details of it are shown as follows.

We construct a new matrix

$$\mathbf{E} = \nabla^2 L(\mathbf{x}) + \lambda \boldsymbol{\varepsilon} \mathbf{I} \quad (13)$$

where $\boldsymbol{\varepsilon}$ is a set of appropriate elements, and \mathbf{I} is an identity matrix, which can form the elements on the principal diagonal of \mathbf{E} .

$$\varepsilon_i = \frac{2x_i}{(\delta + x_i)^3} \quad (14)$$

where ε_i are the elements in ε . Then, we get a positive definite matrix

$$E = 2A^T A + \lambda \text{diag} \left(\frac{2}{(\delta + x_i)^2} \right) \quad (15)$$

thus d is updated to

$$d = -E^{-1} \nabla L(x) \quad (16)$$

and the recurrence formula of FAL0 is

$$x_{k+1} = h d + x_k \quad (17)$$

where h is the step length of the iteration, which can be chosen by the line search method [17].

Because x is a sparse vector and most of its elements are 0, we neglect the zero elements to simplify the calculation. Here, we consider

$$S = \{i : |x_i| > 0\} \quad (18)$$

$$x^S = (x_i), i \in S \quad (19)$$

$$A^S = (a_i), i \in S \quad (20)$$

where a_i is the i th column of A . Thus, Equation (17) is updated to

$$\begin{cases} x_{k+1}^S = x_k^S + h d \\ x^{\bar{S}} = 0 \end{cases} \quad (21)$$

where \bar{S} is the complement set of S . In the iterative process, δ decreases gradually, and its update formula is $\delta_{k+1} = 0.7\delta_k$.

2.2. Algorithm Description

The pseudo code of FAL0 is shown in Table 1, where x_0 and δ_0 are the initial value of x and δ , respectively. $A^T(AA^T)y$ is the least square solution of $y = Ax$. A , y , and K depend on the specific sparse problems, and λ and h can be chosen by GCV and line search method, respectively. The usage of $\delta_{k+1} = 0.7\delta_k$ makes δ smaller and smaller, which makes the search scope clearer.

Table 1. The pseudo code of FAL0.

Algorithm: FAL0

Input: A , y , λ , h , K , and the termination condition: $\delta_{\min} = 10^{-3}\delta_0$;

Output: The recovery signal x^* ;

Initialization: $x_0 = A^T(AA^T)y$, $\delta_0 = 1$;

Step 1:

$$S = \{i : |x_i| > 0\}$$

$$x^S = (x_i), i \in S$$

$$A^S = (a_i), i \in S$$

Step 2: Update d according to Equation (16);

Step 3: Iterate,

$$\begin{cases} x_{k+1}^S = x_k^S + h d \\ x^{\bar{S}} = 0 \end{cases};$$

Step 4: Update δ , $\delta_{k+1} = 0.7\delta_k$;

Step 5: If $\delta < \delta_{\min}$, output to x^* , or else go to **Step 1** and continue iterating.

2.3. Computational Complexity Analysis of FAL0

The FAL0 algorithm is mainly based on the modified Newton method, which is quadratic convergent [18]. The main computational complexity of FAL0 lies in the computation of the initial solution and the iteration, which involves the product of the matrix and vector. The computational burden of $A^T y$ does not exceed $O(n^2)$, and because we neglected 0 in x in the calculation process, the computational burden will be further reduced.

3. Computer Simulation Experiments and Analysis

In this section, we give several computer simulation experiments on the recovery of both random sparse signals and noisy image tasks to prove the performance of FAL0. The performances are compared with BP [9], OMP [11], MP [12], AL0 [14], and AL0-L [13], and these comparison algorithms all refer to the source code given by the author in the supplement. All of the experiments are implemented in MATLAB 2012a on a personal computer with a 2.59 GHz Intel Pentium dual-core processor and 2 GB memory running the Microsoft Windows XP operating system (Redmond, WA, USA).

3.1. Simulation Experiments on Random Sparse Signal Recovery

In these simulation experiments, the matrix A is a Gaussian random matrix of size $m \times n$, where $m < n$. The vector x is a sparse signal of n dimensions. The K nonzero elements are randomly distributed in x , and their values obey the standardized normal distribution. The vector y is given by $y = Ax$. When y and A are known, we use the above six algorithms to recover x , and the result is x^* .

Parameters λ and h can be chosen by GCV [16] and line search [17] method, respectively. When the equation set changes, λ and h will also change.

The sparse recovery algorithm is usually evaluated by precision and convergence speed, and we adopt exact recovery rate and maximal time consumption to indicate these two metrics.

We assume that if x^* satisfies $\|x - x^*\|_2 \leq 10^{-4} \|x\|_2$, x is exactly recovered. Thus, the exact recovery rate can be defined as the ratio of the number of the exactly recovered trails to the total number of the trails under the condition of the fixed parameters. All curves in the figures are obtained by averaging the results of 500 independent trails. In addition, the maximal time consumption is defined as the maximal runtime of the 500 trails under the condition of the fixed parameters.

In the first experiment, we set $K = 300$ and $n = 1500$, and m varies from 400 to 1000. Every time m changes, we calculate the exact recovery rate of the six algorithms by averaging the results of the 500 trails. In addition, we show maximal time consumption of six algorithms. The results are shown in Figure 1. The number of measurements means the number of rows of matrix A and also means the number of elements in vector y .

In the second experiment, we set $m = 700$ and $n = 1500$, and K varies from 100 to 700. Every time K changes, we calculate the exact recovery rate of the six algorithms by averaging the results of the 500 trails. In addition, we show maximal time consumption of six algorithms. The results are shown in Figure 2.

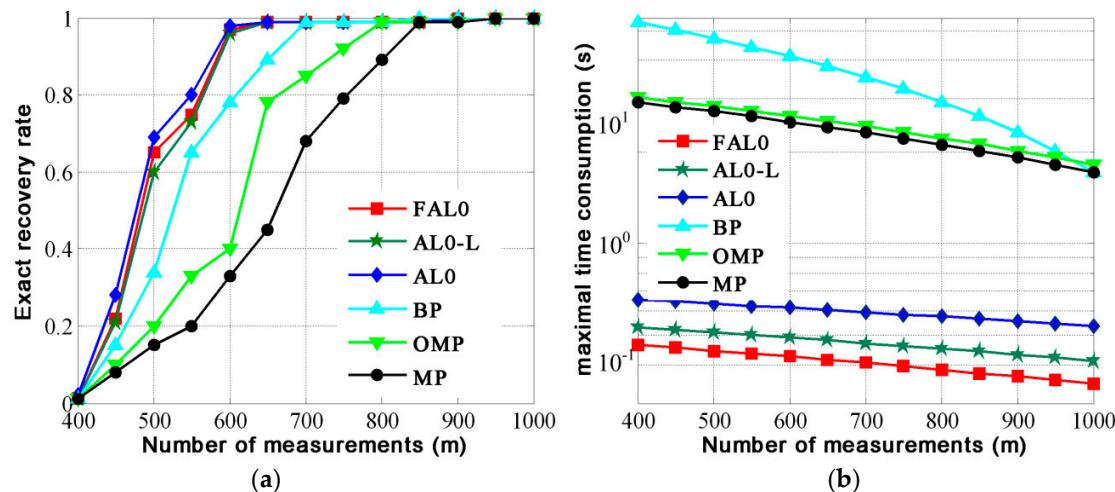


Figure 1. Performance of the six algorithms with different measurements. (a) Exact recovery rate; (b) maximal time consumption.

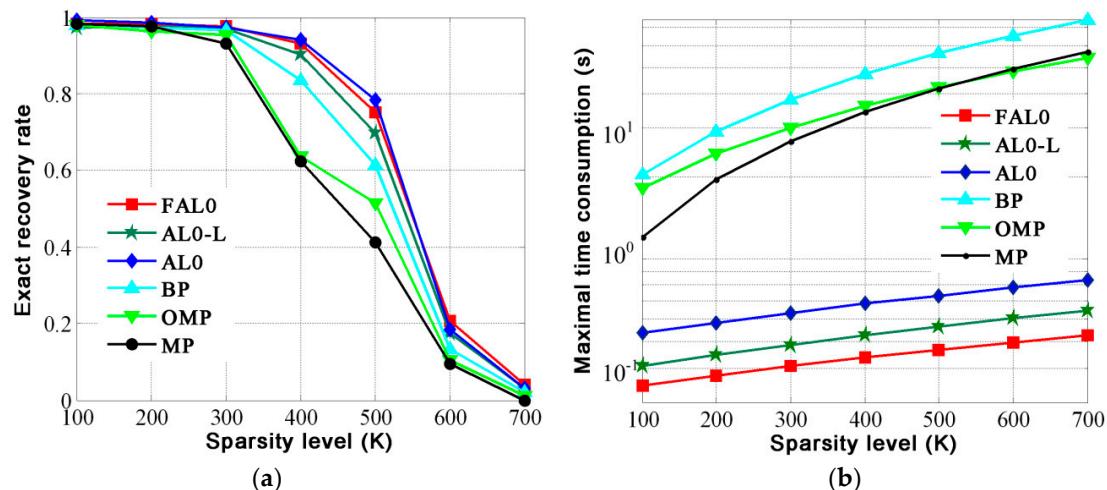


Figure 2. Performance of the six algorithms with different sparsity levels. (a) Exact recovery rate; (b) maximal runtime time consumption.

From Figure 1a, we can see that the exact recovery rate of FAL0 is better than that of BP, MP, and OMP and similar to that of AL0 and AL0-L in the case of the same measurement value. From Figure 1b, we find that the maximal time consumption of FAL0 is much less than that of the other five algorithms under the same conditions.

From Figure 2a, we can see that the exact recovery rate of FAL0 is better than that of BP, MP, and OMP and quite close to that of AL0 and AL0-L in the case of the same sparsity level. From Figure 2b, we can see that maximal time consumption of FAL0 is much less than that of the other five algorithms under the same sparsity level.

These experiments prove that FAL0 performs well in the aspect of algorithm's precision and convergence speed, which indicates that the proposed algorithm is suitable for the fast recovery of sparse signals.

3.2. Simulation Experiments on Image Denoising

Image denoising is also a common application of sparse recovery algorithm [19]. The detailed process of the CS-based image denoising method can be found in the literature [20], which includes the image's sparse representation, recovery of the sparse image, and inverse transformation of the sparse image.

In this section, we compare the performance of these six algorithms in the CS-based image denoising method and adopt signal noise ratio (SNR, Equation (22)), time consumption, and memory usage to evaluate the performance of these six algorithms. In the experimental process, we keep the other conditions fixed and only change the sparse recovery algorithm.

$$NR = 10 \cdot \lg \frac{\sum_i |x_i|^2}{\sum_i |x_i^* - x_i|^2} \quad (22)$$

In Equation (22), x_i and x_i^* are the elements in \mathbf{x} and \mathbf{x}^* , respectively. The higher the SNR, the better the image quality is.

We adopt biomedical imaging, including a computer tomography (CT) image and a fundus image, and the typical Lena image as the test objects. Figures 3–5 show the denoising effect of these six algorithms intuitively. We can see that the denoising performance of FAL0 is close to that of AL0 and AL0-L but better than that of MP, OMP, and BP.

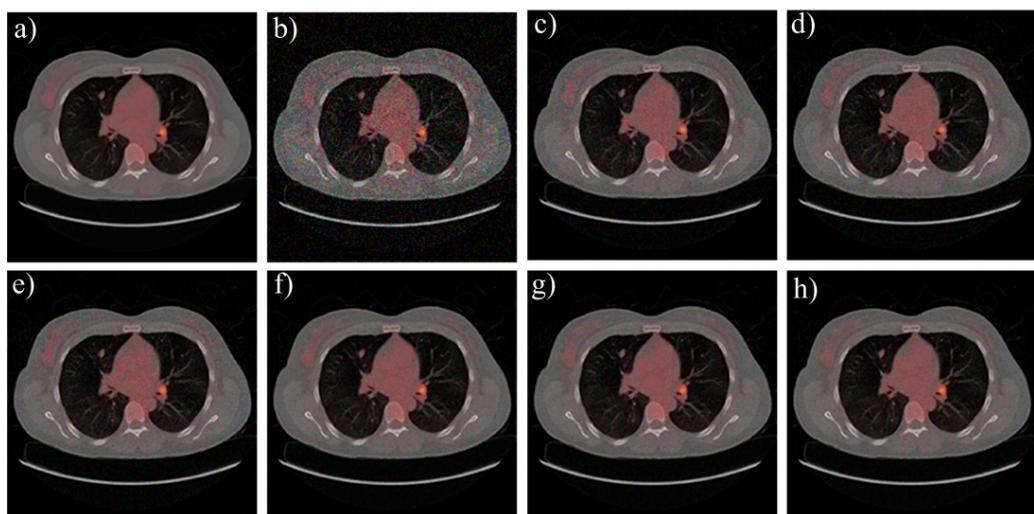


Figure 3. Denoising performance of the six algorithms to the CT image. (a) Original image; (b) noisy image; (c) MP; (d) OMP; (e) BP; (f) FAL0; (g) AL0; (h) AL0-L.

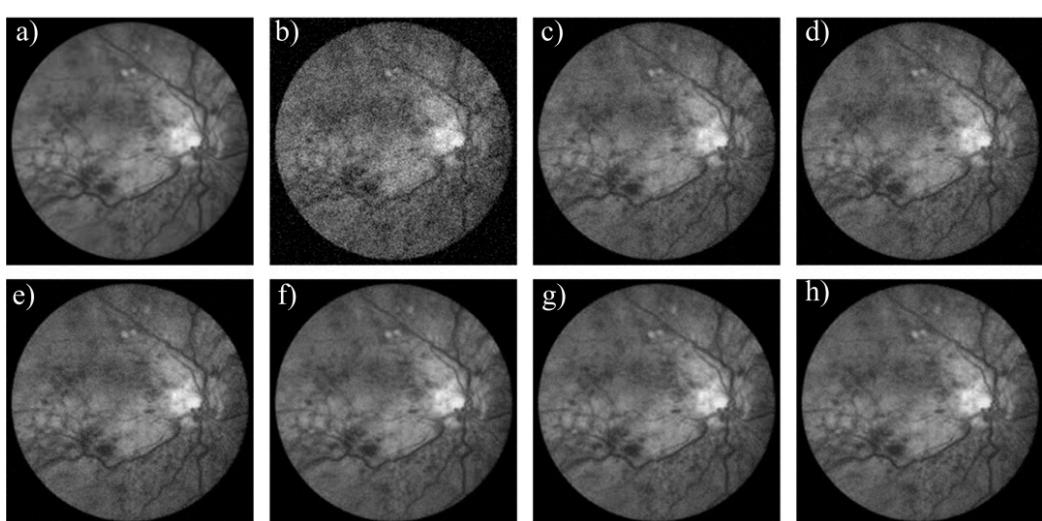


Figure 4. Denoising performance of the six algorithms to the fundus image. (a) Original image; (b) noisy image; (c) MP; (d) OMP; (e) BP; (f) FAL0; (g) AL0; (h) AL0-L.



Figure 5. Denoising performance of the six algorithms to the Lena image. (a) Original image; (b) noisy image; (c) MP; (d) OMP; (e) BP; (f) FAL0; (g) AL0; (h) AL0-L.

Their SNR and time consumption are shown in Table 2. From Table 2, we can see that the performance in the application of image denoising of FAL0 is better than that of BP, MP, and OMP and quite close to that of AL0 and AL0-L. Moreover, FAL0 takes the least time, which proves that FAL0 is a fast and effective sparse recovery algorithm.

Table 2. Denoising performance of the six algorithms.

Image	Algorithm	SNR (dB)	Time (s)	Memory Usage (MB)
CT	MP	26.30	46.58	65.76
	OMP	26.19	43.72	60.58
	BP	28.32	78.33	101.75
	FAL0	33.19	10.79	22.37
	AL0	34.01	22.24	36.68
	AL0-L	32.17	15.76	33.52
Fundus	MP	27.52	44.57	68.32
	OMP	26.88	42.16	61.07
	BP	29.11	81.15	112.59
	FAL0	33.12	11.70	20.45
	AL0	33.45	21.12	37.95
	AL0-L	32.77	15.69	36.58
Lena	MP	28.22	49.02	68.70
	OMP	27.73	46.33	59.42
	BP	29.45	85.19	127.34
	FAL0	33.20	11.15	26.82
	AL0	31.54	23.56	38.96
	AL0-L	32.54	18.77	35.40

4. Conclusions

Based on the previous study of the approximate l_0 norm, we present a fast and effective sparse recovery algorithm for compressed sensing. We adopt a simple fractional function to approximate the l_0 norm and use the modified Newton method to implement the algorithm, which combines the advantages of fast convergence of AL0 and high accuracy of the Newton method. The results of computer simulation experiments indicate that FAL0 is fast and effective in the application of signal recovery and image denoising.

Author Contributions: Conceptualization and writing, D.J.; project administration, Y.Y.; software, T.G.; funding acquisition, D.W.

Funding: Open fund of the State Key Laboratory of Heavy Duty AC Drive Electric Locomotive Systems Integration (No. 2017ZJKF09).

Acknowledgments: This study was supported by the State Key Laboratory of Heavy Duty AC Drive Electric Locomotive Systems Integration.

Conflicts of Interest: The authors declare that there are no conflicts of interests regarding the publication of this paper.

References

1. Simon, F.; Holger, R. A mathematical introduction to compressive sensing. In *Applied and Numerical Harmonic Analysis*; Springer: New York, NY, USA, 2013.
2. Ji, S.; Xue, Y.; Carin, L. Bayesian compressive sensing. *IEEE Trans. Signal Process.* **2008**, *56*, 2346–2356. [[CrossRef](#)]
3. Jin, D.; Yang, Y.; Luo, Y.; Liu, S. Lens distortion correction method based on cross-ratio invariability and compressed sensing. *Opt. Eng.* **2018**, *57*, 054103. [[CrossRef](#)]
4. Babacan, S.D.; Molina, R.; Katsaggelos, A.K. Bayesian compressive sensing using Laplace priors. *J. IEEE Trans. Image Process.* **2010**, *19*, 53–63. [[CrossRef](#)] [[PubMed](#)]
5. Du, X.; Cheng, L.; Chen, D. A simulated annealing algorithm for sparse recovery by l_0 minimization. *Neurocomputing* **2014**, *131*, 98–104. [[CrossRef](#)]
6. Mashud, H.; Mahata, K. An approximate L0 norm minimization algorithm for compressed sensing. In Proceedings of the IEEE International Conference on Acoustics, Taipei, Taiwan, 19–24 April 2009.
7. Candes, E.; Tao, T. Decoding by linear programming. *IEEE Trans. Inf. Theory* **2005**, *51*, 4203–4215. [[CrossRef](#)]
8. Davis, G. Adaptive Greedy Approximations. *Constr. Approx.* **1997**, *13*, 57–98. [[CrossRef](#)]
9. Chen, S.S.; Donoho, D.L.; Saunders, M.A. Atomic Decomposition by Basis Pursuit. *SIAM Rev.* **2001**, *43*, 129–159. [[CrossRef](#)]
10. Gorodnitsky, I.F.; Rao, B.D. Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm. *IEEE Trans. Signal Process.* **2002**, *45*, 600–616. [[CrossRef](#)]
11. Tropp, J.A.; Gilbert, A.C. Signal Recovery from Random Measurements Via Orthogonal Matching Pursuit. *IEEE Trans. Inf. Theory* **2007**, *53*, 4655–4666. [[CrossRef](#)]
12. McClure, M.R.; Carin, L. Matching pursuits with a wave-based dictionary. *IEEE Trans. Signal Process.* **1997**, *45*, 2912–2927. [[CrossRef](#)]
13. Zhang, C.; Song, S.; Wen, X.; Yao, L.; Long, Z. Improved sparse decomposition based on a smoothed L0 norm using a Laplacian kernel to select features from MRI data. *J. Neurosci. Methods* **2015**, *245*, 15–24. [[CrossRef](#)] [[PubMed](#)]
14. Mohimani, H.; Babaie-Zadeh, M.; Jutten, C. A Fast Approach for Overcomplete Sparse Decomposition Based on Smoothed l_0 Norm. *IEEE Trans. Signal Process.* **2009**, *57*, 289–301. [[CrossRef](#)]
15. Elad, M. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*; Springer: New York, NY, USA, 2010.
16. Lukas, M.A. Asymptotic optimality of generalized cross-validation for choosing the regularization parameter. *Numer. Math.* **1993**, *66*, 41–66. [[CrossRef](#)]
17. Nocedal, J.; Stephen, W. *Numerical Optimization*; Springer Science & Business Media: New York, NY, USA, 2006.
18. Altman, T.; Boulos, P.F. Convergence of Newton method in nonlinear network analysis. *Math. Comput. Model.* **1995**, *21*, 35–41. [[CrossRef](#)]
19. Jin, D.; Yang, Y. Sensitivity analysis of the error factors in the binocular vision measurement system. *Opt. Eng.* **2018**, *57*, 104109. [[CrossRef](#)]

20. Li, S.; Qi, H. A Douglas-Rachford Splitting Approach to Compressed Sensing Image Recovery Using Low-Rank Regularization. *IEEE Trans. Image Process.* **2015**, *24*, 4240–4249. [CrossRef] [PubMed]



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).