Data-driven diagnosis for compressed sensing algorithms

Y Nakanishi-Ohno\textsuperscript{1,2} and K Hukushima\textsuperscript{1}

\textsuperscript{1}Graduate School of Arts and Sciences, The University of Tokyo, 3-8-1, Komaba, Meguro, Tokyo, 153-8902, Japan
\textsuperscript{2}PRESTO, Japan Science and Technology Agency, 4-1-8, Honcho, Kawaguchi, Saitama, 332-0012, Japan
E-mail: nakanishi@phys.c.u-tokyo.ac.jp

Abstract. A data-driven diagnosis approach is developed for compressed sensing algorithms to objectively detect the failure of each algorithm and the insufficiency of data. Compressed sensing enables us to reduce the amount of data to a threshold; however, the problem is that no one can tell where the threshold is in practice. The approach is based on the statistical technique of cross-validation, in which part of the data is set aside as a validation dataset to assess the results of the analysis of the remaining data with an objective algorithm. The relation between cross-validation error and the size ratio of the whole dataset and validation dataset is evaluated and compared with a power-law relation to determine whether data analysis was successfully conducted. After summarizing our recent theoretical work on the first application of the approach to the basis pursuit method, it is shown by numerical experiments that the approach is universally applicable to several other algorithms.

1. Introduction
Data-driven science is growing as a tree trunk of science, which stands on the foundation of information science to support the many branches of natural science. Every branch commonly suffers from high-dimensional data but studying data handling exclusively within itself can sometimes go astray. In data-driven science, scientists from various disciplines collaborate on data-related issues, such as data acquisition, data analysis, data interpretation, and data re-acquisition, in order to create a driving force for sustaining and accelerating the whole cycle of science. In this paper, “data-driven diagnosis” is discussed as a kind of data interpretation. It is interesting to consider what happens if you come across a situation where the results of data analysis with different algorithms are vague or ambiguous. One can speculate that some algorithms are going the wrong way, while others may doubt whether there is enough data to be interpreted as something scientifically significant in the first place. Then, it is essential to objectively detect the failure of each algorithm and the insufficiency of data. What is important here is that the procedure of data-driven diagnosis should not involve any hypothesis to be tested. Researchers from information science will probably develop algorithms based on unconfirmed facts about natural phenomena; in contrast, those from natural science will misuse algorithms beyond the scope of their applications if they work separately. Therefore, we cannot make too much of the culture of data-driven science.
Here, we develop a data-driven diagnosis framework for compressed sensing. The purpose of compressed sensing is to improve the efficiency and availability of experiments [1, 2]. A key technology is sparse modeling, where one assumes that an object is described by a parsimoniously small number of explanatory factors. Owing to the affinity of sparse modeling with natural science, there are many applications in various fields: exploration geophysics [3, 4], magnetic resonance imaging [5, 6], ghost imaging [7, 8], nuclear magnetic resonance spectroscopy [9, 10], astronomical radio interferometry [11, 12], and scanning tunneling microscopy/spectroscopy [13]. On the other hand, in information science, there are various compressed sensing algorithms with sparse modeling, which include:

- orthogonal least squares (OLS) [14]
- orthogonal matching pursuit (OMP) [15]
- homotopy method for basis pursuit (Homotopy-BP) [16]
- least angle regression (LARS) [17]
- iterative hard thresholding (IHT) [18, 19]
- compressive sampling matching pursuit (CoSaMP) [20]
- subspace pursuit (SP) [21]
- hard thresholding pursuit (HTP) [22]

The algorithm that should be used remains confusing to users although many studies have evaluated and compared these algorithms in terms of factors such as the necessary and sufficient conditions for success, computational complexity, number of tuning parameters, etc. It makes sense that a tractable algorithm should be chosen and more importantly that a reliable algorithm is used. However, the conditions for success are often described using the sparsity of what one wants to know as pointed out in [23], and this is against the rule that one should not employ a hypothesis. This is nothing but a data-driven diagnosis problem.

In our most recent work [24], we developed a cross-validation approach to data-driven diagnosis and theoretically verified that it works well for compressed sensing with the basis pursuit method. Cross-validation is a classical statistical technique for parameter tuning and model selection [25–27], but we will attempt to use it in an unconventional way. We have found that when cross-validation error is plotted against the size ratio of the training data and whole data, it behaves in a power-law fashion at the phase boundary of the success and failure of compressed sensing; this suggests that a critical factor for data-driven diagnosis can be obtained by monitoring the relation. In the current work, further progress in data-driven diagnosis will be made by investigating the applicability of our approach to the algorithms listed above.

The rest of the paper is organized as follows: Section 2 formulates the compressed sensing problem and the cross-validation procedure. Section 3 summarizes our recent theoretical work on the basis pursuit method. Section 4 discusses the results of numerical experiments to examine the applicability of our approach to several algorithms, and concludes the paper.

2. Formulation

2.1. Compressed sensing

The general problem of sensing, not restricted to compressed sensing, is formulated as finding the representation of an object, $x = (x_i)_{i \in [N]}$, from a set of data $y = (y_{\mu})_{\mu \in [M]}$, which is given by

$$y = Ax^0. \quad (1)$$

Matrix $A = (a_{\mu i})_{\mu \in [M], i \in [N]}$ represents a linear measurement process and each item of data $y_{\mu}$ is acquired by a measurement denoted by a corresponding row vector $a_{\mu} = (a_{\mu i})_{i \in [N]}$. The object
has a true representation $x^0$. It is noted that $[N]$ is the set of all positive integers not exceeding $N$, namely, $[N] = \{1, 2, \ldots, N\}$. For convenience, we introduce a parameter $\alpha = M/N$ to denote the amount of data. If the dimension of $y$ is greater than or equal to that of $x$, that is, $\alpha \geq 1$, one can reconstruct $x^0$ from $y$ as follows:

$$x^0 = A^+y, \tag{2}$$

Matrix $A^+$ is the Moore–Penrose pseudoinverse of $A$, and it coincides with the inverse $A^{-1}$ when $\alpha = 1$. In other words, $x^0$ is obtained by using the method of least squares as follows:

$$x^0 = \arg \min_{x} \frac{1}{2M} \sum_{\mu}(a_\mu x - y_\mu)^2. \tag{3}$$

Another problem with compressed sensing is solving an underdetermined equation system due to $\alpha < 1$. In the underdetermined case, there are many solution candidates satisfying $Ax = y$. If you multiply the pseudoinverse of $A$ to $y$ in the same way as the case of $\alpha \geq 1$, you are supposed to calculate

$$A^+y = \arg \min_{x} \|x\|_2 \quad \text{subject to} \quad Ax = y \tag{4}$$

as derived from the definition of pseudoinverse in the case of $\alpha < 1$, where the $l_2$-norm $\|\cdot\|_2$ is defined as $\|x\|_2 = \sqrt{\sum_i x_i^2}$. However, the $l_2$-norm minimization technique does not give the true representation $x^0$ in the underdetermined case [35]. An effective approach to this problem is sparse modeling, which assumes that $x^0$ has a sparse representation composed of a small number of non-zero elements. The number of non-zeros that $x$ has is denoted by the $l_0$-norm $\|x\|_0$, which is used to measure sparsity. The efficacy of sparse modeling is intuitively understood as follows: If the assumption is true, then the number of unknowns can be sufficiently reduced so that $M > \|x\|_0$, and the ill-posed problem can be solved.

To simplify the discussion, we will consider the following case throughout the study: Each element of $x^0$ is generated from a mixture distribution $p(x^0_i) = \rho_0 \mathcal{N}(0, 1) + (1 - \rho_0) \delta(x^0_i)$ in an independent and identically distributed (iid) way. The distribution $\mathcal{N}(\mu, \sigma^2)$ is a Gaussian distribution with mean $\mu$ and variance $\sigma^2$. The function $\delta(\cdot)$ is the Dirac delta function. The parameter $\rho_0$ represents the sparsity of the true representation since

$$\rho_0 = \int dx^0 \left( \prod_i p(x^0_i) \right) \frac{\|x^0\|_0}{N} = \left( \frac{\|x^0\|_0}{N} \right)_{x^0}. \tag{5}$$

The angle brackets $\langle \cdot \rangle$ denote the expectation with respect to the subscript in this way. Each element of $A$ is generated from $\mathcal{N}(0, N^{-1})$ in an iid way. In this setting, each element of $y$ is acquired without any noise according to equation (1); in other words, generated from $p(y_\mu) = \delta(y_\mu - a_\mu x^0)$.

The remaining question in compressed sensing is which algorithm successfully finds the support set of $x^0$, namely, $S_0 = \{i | x^0_i \neq 0\}$. Although an exhaustive search should be conducted, in the high-dimensional case, it is intractable owing to its exponentially increasing computational complexity. As listed in the Introduction, there are various kinds of algorithms, but one often becomes confused about which algorithm to use for data analysis. Therefore, a data-driven diagnosis framework is required.
2.2. Cross-validation

Cross-validation is a general but simple statistical technique for evaluating generalization error, which measures the predictability of an algorithm on unseen data [25–27]. This error is called cross-validation error in the context of cross-validation. Cross-validation error is a frequently used criterion for parameter tuning and model selection in various fields of science. The procedure of calculating cross-validation error has three main steps. First, the available dataset \( y \) is divided into two non-empty subsets: a training dataset \( (y_\mu)_{\mu \in T} \) and validation dataset \( (y_\mu)_{\mu \in V} \), where \( V = [M] \setminus T \). For convenience, a parameter \( k = M/|V| \) \( (k > 1) \), where \( |V| \) is the number of elements in \( V \), is introduced to represent the size ratio of the validation dataset and whole dataset, and it will also play an important role later in data-driven diagnosis. Second, an estimation algorithm is applied to the training dataset to obtain an outcome \( \hat{x}_T \). Finally, how accurately the outcome \( \hat{x}_T \) describes the validation dataset \( (y_\mu)_{\mu \in V} \) is measured by cross-validation error introduced as follows:

\[
CVE = \frac{1}{2|V|} \sum_{\mu \in V} (a_\mu \hat{x}_T - y_\mu)^2. \tag{6}
\]

If cross-validation error has a small value close to zero, it means that the unseen data \( (y_\mu)_{\mu \in V} \) acquired from future measurements \( (a_\mu)_{\mu \in V} \) are well predicted by the outcome \( \hat{x}_T \) estimated from the already seen data \( (y_\mu)_{\mu \in T} \) acquired from past measurements \( (a_\mu)_{\mu \in T} \) with an algorithm of interest. The cross-validation error fluctuates according to how the data is divided. Then, the calculation of cross-validation error is often repeated with many different partitions of the same dataset to obtain a typical value of cross-validation error. There are various ways of partitioning data, such as leave-one-out cross-validation, \( k \)-fold cross-validation, and Monte Carlo cross-validation. In the repetitive calculations, every item of the data plays a role of training and validation data by turns. What is used as training data in a calculation is used as validation data in another calculation, and vice versa. This is why the statistical technique is called cross-validation.

Cross-validation is a key ingredient in our data-driven diagnosis approach because of two points. One is that the calculation of cross-validation does not involve a true representation of \( x_0 \), its support set \( S_0 \), or its sparsity \( \rho_0 \) at all. The other point is that cross-validation is applicable to whatever algorithm is used for estimation in compressed sensing, since the procedure does not assume the use of a specific algorithm to obtain \( \hat{x}_T \). Therefore, a cross-validation approach meets the need for objectivity and is expected to be universally applicable.

3. Application to basis pursuit

This section summarizes our previous work on the initial application of our approach to compressed sensing with the basis pursuit method [24].

Basis pursuit is formulated as the following optimization problem [28]:

\[
\hat{x} = \arg \min_{x} \|x\|_1 \quad \text{subject to} \quad Ax = y. \tag{7}
\]

The \( l_1 \)-norm \( \| \cdot \|_1 \) is defined as \( \|x\|_1 = \sum_i |x_i| \). As from a linear programming algorithm [29, 30], efficient algorithms such as Homotopy-BP [16], a Bregman iterative algorithm [31], an approximate message-passing algorithm [32], and a primal-dual algorithm [33] have been developed for carrying out basis pursuit. The performance of basis pursuit in compressed sensing has been well studied and representative results are shown in figures 1(a) and 1(b) [34–36]. Figure 1(a) is a phase diagram of basis pursuit and the yellow line divides the two phases of success and failure. The success phase of basis pursuit is broader than that of the pseudoinverse and this fact has expanded the possibilities of compressed sensing. However, the result of the
Figure 1. (a) Phase diagram on the success and failure in compressed sensing. Horizontal and vertical axes represent true sparsity $\rho_0$ and data amount $\alpha$, respectively. The yellow line is the phase boundary of basis pursuit. The dashed line is the phase boundary of the pseudoinverse. For each method, the top-left region and bottom-right regions are the success phase and failure phase, respectively. (b) Plots of mean squared error against $\rho_0$ with respect to basis pursuit. $\alpha = 0.3$. (c), (d) Plots of cross-validation error against $k$ with respect to basis pursuit. $\alpha = 0.3$. $\rho_0 = 0.0822$ (blue), 0.0872 (yellow), 0.0922 (red). These points of $\rho_0$ are indicated by corresponding colored circles in (b), and the yellow one is at the phase boundary. (a) and (b) are derived from calculations in [35]. (c) and (d) are derived from calculations in [24].

The phase diagram has a critical defect in practice. For example, consider the case of $\alpha = 0.3$ as shown in figure 1(b). Figure 1(b) plots the typical value of the following mean squared error,

$$\text{MSE} = \frac{1}{2N} \sum_i (\hat{x}_i - x^0_i)^2,$$

against $\rho_0$. In the left success phase, the mean squared error is equal to zero, while in the right failure phase, it takes a non-zero value. However, without the true sparsity $\rho_0$, no one knows in which phase the outcome of basis pursuit is.

To discuss data-driven diagnosis in [24], we analyzed the typical cross-validation error values of basis pursuit, defined as

$$\text{CVE}_{\text{BP}} = \left\langle \left\langle \frac{1}{2|\mathcal{V}|} \sum_{\mu \in \mathcal{V}} (a_{\mu}x - y_{\mu})^2 \right\rangle \right\rangle_{x \cdot A \cdot x^0},$$
where the use of basis pursuit with a training dataset is reflected in the probability
\[ p(x) \propto e^{-\beta \|x\|_1} \prod_{\mu \in T} \delta(a_\mu x - y_\mu). \quad (10) \]

Comparison with equation (7) shows that the Boltzmann factor \( e^{-\beta \|x\|_1} \) in the limit \( \beta \to +\infty \) corresponds to the \( l_1 \)-norm minimization and the delta functions represent the equality constraint. We applied the replica method, which was developed in the field of spin-glass theory [37], to obtain the results for a replica-symmetric ansatz. In our theoretical analysis, the limit \( N \to +\infty \) was dealt with because the use of the saddle-point method was justified.

Figures 1(c) and 1(d) plot the typical values of cross-validation error against \( k \), the size ratio of the validation dataset and whole dataset, when \( \alpha = 0.3 \). The blue line in figure 1(c) represents a success case of \( \rho_0 = 0.0822 \), while the red line in figure 1(d) represents a failure case of \( \rho_0 = 0.0922 \). For reference, the yellow lines in both figures represent the case of \( \rho_0 = 0.0872 \) at the phase boundary.

Surprisingly, a glance at figures 1(c) and 1(d) shows that there is a qualitative difference between the success and failure phases. In the success phase, the cross-validation error decreases in an exponential fashion as \( k \) increases; in the failure phase, it decreases to a finite value for large \( k \). At the phase boundary between them, a power-law relation is seen in the asymptotic behavior of the cross-validation error. Our intuitive understanding of this is as follows: Imagine that your dataset is actually in the success phase in figure 1(a) although you do not know it, and that you try setting aside a part of the data as validation data. If the amount of validation data is small, your training dataset remains in the success phase and the cross-validation error disappears for large \( k \), as shown in figure 1(c). In the opposite case, if your whole dataset is in the failure phase, your training dataset is also within the failure phase and the cross-validation error takes a finite value for any \( k \), as shown in figure 1(d). This may be the reason why the behavior of cross-validation error distinguishes the success or failure of basis pursuit.

Next, we have to consider what should be done if a failure of basis pursuit is successfully detected. Although data re-acquisition may be an option, it is also promising to perform data analysis with another algorithm. According to the intuitive explanation above, our approach is expected to be universally applicable to other algorithms. In the following section, this applicability is examined.

4. Application to algorithms
In this section, we compare various aspects of well-known compressed sensing algorithms, including data-driven diagnosis by numerical experiments. We examined the eight algorithms, which are listed in Introduction and are explained in detail in Appendix A. We set the parameter \( \epsilon \) to stop the iteration of each algorithm at \( 1 \times 10^{-9} \). There are some variants of these algorithms with respect to initialization, stopping criterion, and so forth, each of which is an interesting target for data-driven diagnosis as well.

4.1. Conventional work
We examined the performance of the algorithms in terms of mean squared error and success rate among 100 instances of dataset \( y \), measurement matrix \( A \), and true representation of an object \( x^0 \). Figures 2(a) and 2(b) show the results for mean squared error and success rate, respectively. The success rate is defined as the number of success instances divided by the total number of instances. The parameters are as follows: \( N = 1000, \rho_0 = 0.05, 0.06, \ldots, 0.15, \alpha = 0.3 \). We can see that numerical mean squared error of Homotopy-BP is consistent with the theoretical results of basis pursuit shown in figure 1(b), which confirms our theoretical work. Although Homotopy-BP estimates a representation with relatively low mean squared error, it also has
the lowest success rate. Thus, the basis pursuit method is considered to be unsuitable for the correct estimation of the support set. Among the algorithms studied, SP achieved the highest performance in success rate regardless of $\rho_0$, and all but LARS were superior to the basis pursuit method. From the viewpoint of data-driven diagnosis, it is worthwhile to mention that every algorithm as well as basis pursuit exhibited a phase transition of success and failure, and our approach is expected to be applicable to the algorithms.

We also examined the computation time of the algorithms among 100 instances. Figures 2(c) and 2(d) show the results. The parameters are as follows: (c) is the same as (a) and (b), that is, $N = 1000$, $\rho_0 = 0.05, 0.06, \ldots, 0.15$, $\alpha = 0.3$. (d) $N = 100, 200, \ldots, 1000, 2000, \ldots, 10000$, $\rho_0 = 0.05$, $\alpha = 0.3$. We find that OLS had the longest computation time, while OMP had the shortest time, regardless of $N$ and $\rho_0$. Unexpectedly, in addition to its good performance in terms of success rate, the computation time of SP was the fourth shortest when it correctly estimated the true representation. As shown in figure 2(c), computation time increases as $\rho_0$ increases because many iterations are needed to find the true representation composed of many non-zeros. Besides, it may be noted that the relation between sparsity $\rho_0$ and computation time in the failure phase is different from that in the success phase. Except for Homotopy-BP and LARS, computation time greatly increased around the transition point of $\rho_0$. Since it can be measured without knowing $x^0$ or $\rho_0$, computation time is perhaps a promising quantity for future work on data-driven diagnosis.

Can conventional work answer the question of what algorithm should be used? Indeed one
can recommend the use of SP because it achieved the best success rate at the fourth shortest computation time. However, it is not guaranteed that every result of SP will be correct unless its success rate is exactly one in all cases. Thus, what is really important is data-driven diagnosis to interpret whether the algorithm actually succeeds or fails in data analysis.

4.2. Data-driven diagnosis

Our approach to data-driven diagnosis is based on cross-validation, which can be conducted without the true representation $x^0$ or the true sparsity $\rho_0$. In our approach, one only has to monitor the relation between the cross-validation error and parameter $k = M/|V|$, which represents the size ratio of the whole dataset and validation dataset. If our approach works well, cross-validation error decreases asymptotically in an exponential fashion in the case of success, and remains a finite value for large $k$ in the case of failure.

Figure 3 shows the results of data-driven diagnosis for the algorithms. The parameters are as follows: $N = 300$, $\rho_0 = 0.1$, $\alpha = 0.3$. Cross-validation is plotted at various points of $k = 2, 3, \ldots, 90$. The maximum $k$ that can be examined is equal to the size of dataset $M = \alpha N$ because at least one item of data has to be set aside as validation data. Each plotted value of cross-validation error is the median among 100 trials of cross-validation with random partitions of data into training and validation data. In each panel of the algorithm, the results of 10 instances of $y, A, x^0$ are shown. The blue lines represent success instances and red lines represent failure instances in light of the mean squared error.

On the whole, every red instance has a finite value at $k = 90$, and in most blue instances, cross-validation error rapidly decreased to zero. The relation between the cross-validation error and parameter $k$ can successfully distinguish the success or failure in a data-driven way. We note that some blue instances remained a finite value even at maximum $k$. This indicates that the training data does not give a correct representation, unlike the whole data. The reason for this is because the whole amount of data is barely above the phase boundary between success and failure, and it enters the failure phase by setting aside a small amount of validation data. Then, our approach sometimes gives conservative results. However, it is better to temporarily delay the interpretation of the results of data analysis rather than cause confusion with an inaccurate analysis.

As shown in figure 3, the cross-validation error value fluctuates, which could lead to a mistake in data-driven diagnosis in more serious situations. In our numerical experiments, only 100 trials of cross-validation were conducted to obtain a typical value, but the number of possible ways of data partition is a binomial coefficient $\binom{M}{k}$ for $k$. A precise and efficient method of calculating a typical value of cross-validation error needs to be developed. Originally, cross-validation was a time-consuming process, but the use of parallel computing, along with the invention of approximation methods [38], is worth discussing.

Consequently, we claim that our data-driven diagnosis approach performs well not only for the basis pursuit method but also for various compressed sensing algorithms. Our data-driven diagnosis approach provides a novel criterion to determine what algorithm should be used. Conventional works just recommend a specific algorithm according to its typical performance, but the present work enables us to select an algorithm that provides reliable results for specific data, or tentatively withhold an interpretation owing to insufficient data. There are many potential applications of data-driven diagnosis in natural science. Compressed sensing with noisy measurements is a practically important one. Moreover, the scope of our approach is not restricted to the formulation of linear regression, and can be expanded to include other formulations such as matrix completion [39], affine rank minimization [40], and demixing problems [41]. This is because cross-validation, on which our approach is based, is a versatile statistical tool.

In conclusion, we developed a cross-validation approach to data-driven diagnosis and
Figure 3. Relation between cross-validation error and parameter $k$ for 10 instances of $(y, A, x^0)$. $N = 300$, $\rho_0 = 0.1$, $\alpha = 0.3$. The blue lines represent success instances and red lines represent failure instances.
demonstrated that the approach works well for various compressed sensing algorithms. Our approach is useful for identifying an algorithm that provides reliable information and that definitely states when there is insufficient data. Data-driven diagnosis should have a significant impact on data-driven science because it solves the general problem of what algorithm should be used for data analysis in natural science among a variety of methods in information science.

Acknowledgments
This work was supported by JSPS KAKENHI Grant Numbers JP17K12749 (YN-O) and JP25120010 (KH), and JST PRESTO Grant Number JPMJPR1773 (YN-O).

Appendix A. Algorithms of compressed sensing
The appendix explains algorithms examined in this study.

Notation

- $A_S$: The submatrix of a matrix $A$ with columns indexed by a support $S$.
- $x_S$: The subvector of a vector $x$ with elements indexed by a support $S$.
- $A^\dagger$: The adjoint matrix of a matrix $A$.
- $\min\{\cdot\}$: The minimum taken only over positive arguments.
- $\text{sign}(\cdot)$: Sign function. It is defined element-wise when the argument is a vector.
- $L_s(\cdot)$: An index set of $s$ largest vector elements in the sense of absolute value.
- $H_s(\cdot)$: Hard thresholding operator. It sets all but $s$ largest vector elements in the sense of absolute value to zero.

Appendix A.1. OLS
The OLS algorithm for estimating $x$, given a vector $y$, a matrix $A$, and a small positive scalar $\epsilon$, is described as follows:

(i) $S = \emptyset$, $r = \frac{\|y\|_2^2}{2M}$.
(ii) repeat the following (1) and (2) while $r > \epsilon$:

1. $[r, j_{\text{add}}] = \min_{j \notin S} \frac{1}{2} \left( \|y - A_{S \cup \{j\}}(A_{S \cup \{j\}})^+ y \|_2 \right)^2$.
2. $S = S \cup \{j_{\text{add}}\}$.

(iii) $x = 0, x_S = (A_S)^+ y$.

OLS takes an index $j_{\text{add}}$ in the support $S$ at each iteration of (ii), and the additional index $j_{\text{add}}$ is determined by using the method of least squares at (ii-1). Be careful not to confuse OLS with the method of ordinary least squares.

Appendix A.2. OMP
The OMP algorithm for estimating $x$, given a vector $y$, a matrix $A$, and a small positive scalar $\epsilon$, is described as follows:

(i) $S = \emptyset$, $x = 0$, $r = y$, $r = \frac{\|y\|_2^2}{2M}$.
(ii) repeat the following (1)–(5) while $r > \epsilon$:

1. $j_{\text{add}} = \arg \max_{j \notin S} |(A_{\{j\}})^+ r|$.
2. $S = S \cup \{j_{\text{add}}\}$.
3. $x_S = (A_S)^+ y$.
\( r = y - Ax. \)
\( r = \frac{||r||_2^2}{2M}. \)

OMP as well as OLS takes an index \( j_{\text{add}} \) in the support \( S \) at each iteration of (ii), but the additional index \( j_{\text{add}} \) is determined by minimizing the absolute inner product between each column vector in \( A \) and the residual \( r \) at (ii-1). Owing to the replacement for the method of least squares, one can reduce computation time since the calculation of the pseudoinverse needs to be done only once per iteration at (ii-3).

**Appendix A.3. Homotopy-BP**

The Homotopy-BP algorithm for estimating \( x \), given a vector \( y \), a matrix \( A \), and a small positive scalar \( \epsilon \), is described as follows:

(i) \( x = 0, r = y, r = \frac{||y||_2^2}{2M}. \)

(ii) \([\gamma_{\text{add}}, j_{\text{add}}] = [\max_j \arg \max_j |(A_j)^\dagger r|]. \)

(iii) \( S = \{j_{\text{add}}\}. \)

(iv) repeat the following (1)–(9) while \( r > \epsilon \):

1. \( d = 0, d_S = ((A_S)^\dagger A_S)^{-1}\text{sign}((A_S)^\dagger r). \)

2. \([\gamma_{\text{add}}, j_{\text{add}}] = [\min_j \arg \min_j \{ \frac{\lambda - (A_j)^\dagger r}{1 - (A_j)^\dagger A_S d_S}, \frac{\lambda + (A_j)^\dagger r}{1 + (A_j)^\dagger A_S d_S} \}]. \)

3. \([\gamma_{\text{rem}}, j_{\text{rem}}] = [\min_j \arg \min_j \{ -\frac{x_{(j)}^2}{d_{(j)}} \}]. \)

4. \( \gamma = \min\{\gamma_{\text{add}}, \gamma_{\text{rem}}\}. \)

5. \( x = x + \gamma d. \)

6. \( r = y - Ax. \)

7. \( r = \frac{||r||_2^2}{2M}. \)

8. \( \lambda = \lambda - \gamma. \)

9. if \( \gamma = \gamma_{\text{add}} \), \( S = S \cup \{j_{\text{add}}\} \); or if \( \gamma = \gamma_{\text{rem}} \), \( S = S \setminus \{j_{\text{rem}}\}. \)

Homotopy-BP is a solver of basis pursuit with a homotopy method. The homotopy method solves a convergent series of optimization problems for an original problem and takes the limit of solutions with respect to the parameter of the series as follows:

\[ \hat{x} = \lim_{\lambda \to +0} \hat{x}_\lambda, \]  

(A.1)

where in the case of basis pursuit \( \hat{x}_\lambda \) is given by

\[ \hat{x}_\lambda = \arg \min_x \left\{ \frac{1}{2} ||y - Ax||_2^2 + \lambda ||x||_1 \right\}. \]  

(A.2)

The optimization problem in equation (A.2) is called basis pursuit denoising [42] and equivalently, lasso (least absolute shrinkage and selection operator) [43]. The Homotopy-BP algorithm follows the solution \( \hat{x}_\lambda \) from \( 0 \) to \( \hat{x} \), and the solution path is piecewise linear as shown in (iv-5). Then, it is enough to find the ends of the linear pieces in the process of (iv-2)–(iv-4).

**Appendix A.4. LARS**

The LARS algorithm for estimating \( x \), given a vector \( y \), a matrix \( A \), and a small positive scalar \( \epsilon \), is described as follows:

(i) \( x = 0, r = y, r = \frac{||y||_2^2}{2M}. \)
(ii) $[\lambda, j_{\text{add}}] = [\max, \arg \max] [(A_{(j)})^T r]$.  

(iii) $S = \{j_{\text{add}}\}$.  

(iv) repeat the following (1)-(7) while $r > \epsilon$:

1. $d = 0$, $d_S = ((A_S)^T A_S)^{-1} \text{sign}((A_S)^T r)$.  
2. $[\gamma, j_{\text{add}}] = [\max, \arg \max] \left\{ \lambda + (A_{(j)})^T r \right\}$.  
3. $x = x + \gamma d$.  
4. $r = y - Ax$.  
5. $r = \frac{\|r\|_2^2}{2M}$.  
6. $\lambda = \lambda - \gamma$.  
7. $S = S \cup \{j_{\text{add}}\}$.  

LARS takes an index $j_{\text{add}}$ in the support $S$ at each iteration of (iv-7) when the $j_{\text{add}}$-th column vector in $A$ gets as close to the residual vector $r$ in angle as the closest vector in the span of $A_S$ is. LARS is the same as Homotopy-BP except that the former does not remove an index once it is selected.

Appendix A.5. IHT

The IHT algorithm for estimating $x$, given a vector $y$, a matrix $A$, and a small positive scalar $\epsilon$, is described as follows:

(i) $s = 0$, $S = \emptyset$, $x = 0$, $r = y$, $r = \frac{\|y\|_2^2}{2M}$.  

(ii) repeat the following (1)-(5) while $r > \epsilon$:

1. $s = s + 1$, $S_{\text{old}} = S^0(\neq \emptyset)$.  
2. repeat the following (a)-(d) while $S \neq S_{\text{old}}$:
   
   a. $S_{\text{old}} = S$.  
   b. $x = H_s(x + A^T r)$.  
   c. $S = \{j | x_j \neq 0\}$.  
   d. $r = y - Ax$.  
3. $x = 0$, $x_S = (A_S)^T y$.  
4. $r = y - Ax$.  
5. $r = \frac{\|r\|_2^2}{2M}$.  

IHT is based on gradient descent for the method of least squares, which is modified by using a hard thresholding operator, as shown in (ii-2-b).

Appendix A.6. CoSaMP

The CoSaMP algorithm for estimating $x$, given a vector $y$, a matrix $A$, and a small positive scalar $\epsilon$, is described as follows:

(i) $s = 0$, $S = \emptyset$, $x = 0$, $r = y$, $r = \frac{\|y\|_2^2}{2M}$.  

(ii) repeat the following (1)-(6) while $r > \epsilon$:

1. $s = s + 1$, $\Delta r = \Delta r^0(> 0)$.  
2. repeat the following (a)-(g) while $\Delta r > 0$:
   
   a. $S_{\text{old}} = S$, $r_{\text{old}} = r$.  
   b. $S = S_{\text{old}} \cup L_{2A}(A^T r)$.  
   c. $x = 0$, $x_S = H_s((A_S)^T y)$.  
   d. $S = \{j | x_j \neq 0\}$.  
   e. $r = y - Ax$.  

\[\]
\( r = \frac{(\|r\|_2)^2}{2M} \).

(3) \( S = S_{\text{old}} \).

(4) \( x = 0, x_S = (A_S)^+y \).

(5) \( r = y - Ax \).

(6) \( r = \frac{(\|r\|_2)^2}{2M} \).

CoSaMP merges an index set of 2s largest elements in absolute value into the current support at (ii-2-b), and prunes some of them by using the method of pseudoinverse and hard thresholding operator at (ii-2-c).

**Appendix A.7. SP**

The SP algorithm for estimating \( x \), given a vector \( y \), a matrix \( A \), and a small positive scalar \( \epsilon \), is described as follows:

(i) \( s = 0, S = \emptyset, x = 0, r = y, r = \frac{(\|y\|_2)^2}{2M} \).

(ii) repeat the following (1)–(6) while \( r > \epsilon \):

(1) \( s = s + 1, \Delta r = \Delta r^0(> 0) \).

(2) repeat the following (a)–(g) while \( \Delta r > 0 \):

(a) \( S_{\text{old}} = S, r_{\text{old}} = r \).

(b) \( S = S_{\text{old}} \cup L_s(A^1r) \).

(c) \( x = 0, x_S = H_s((A_S)^+y) \).

(d) \( S = \{ j | x_j \neq 0 \} \).

(e) \( r = y - Ax \).

(f) \( r = \frac{(\|r\|_2)^2}{2M} \).

(3) \( S = S_{\text{old}} \).

(4) \( x = 0, x_S = (A_S)^+y \).

(5) \( r = y - Ax \).

(6) \( r = \frac{(\|r\|_2)^2}{2M} \).

SP can be regarded as a variant of CoSaMP. A slight difference between them is the size of an index set merged at (ii-2-b).

**Appendix A.8. HTP**

The HTP algorithm for estimating \( x \), given a vector \( y \), a matrix \( A \), and a small positive scalar \( \epsilon \), is described as follows:

(i) \( s = 0, S = \emptyset, x = 0, r = y, r = \frac{(\|y\|_2)^2}{2M} \).

(ii) repeat the following (1) and (6) while \( r > \epsilon \):

(1) \( s = s + 1, \Delta r = \Delta r^0(> 0) \).

(2) repeat the following (a)–(g) while \( \Delta r > 0 \):

(a) \( S_{\text{old}} = S, r_{\text{old}} = r \).

(b) \( S = L_s(x + A^1r) \).

(c) \( x = 0, x_S = (A_S)^+y \).

(d) \( S = \{ j | x_j \neq 0 \} \).

(e) \( r = y - Ax \).

(f) \( r = \frac{(\|r\|_2)^2}{2M} \).

(3) \( S = S_{\text{old}} \).
HTP can be regarded as a variant of IHT. A slight difference between them is that the pseudoinverse is used at (ii-2-c) on a support which is selected by using the hard thresholding operator.

References
[1] Donoho D L 2006 IEEE Trans Inf Theory 52 1289–1206.
[2] Candès E J and Wakin M B 2008 IEEE Signal Process Mag 25 21–30.
[3] Claerbout J F and Muir P 1973 Geophysics 38 826–844.
[4] Taylor H L, Banks S C, and McCoy J F 1979 Geophysics 44 39–52.
[5] Lustig M, Donoho D, and Pauly J M 2007 Magn Reson Med 58 1182–1195.
[6] Lustig M, Donoho D L, Santos J M, and Pauly J M 2008 IEEE Signal Process Mag 25 72–82.
[7] Katz O, Bromberg Y, and Sibberberg Y 2009 Appl. Phys. Lett. 95 131110.
[8] Zhao C, Gong W, Chen M, Li E, Wang H, Zu W, Han S 2012 Appl. Phys. Lett., 101 141123.
[9] Kazimierczuk K and Orekhov V Y 2011 Angew Chem Int Ed 50 5556–5559.
[10] Holland D J, Bostock M J, Gladden L F, and Nietlispach D 2011 Angew Chem Int Ed 50 6548–6681.
[11] Honma M, Akiyama K, Uemura M, and Ikeda S 2014 Publ Astron Soc Jpn 66 95.
[12] Ikeda S, Tazaki F, Akiyama K, Hada K, and Honma M 2016 Publ Astron Soc Jpn 68 45.
[13] Nakashish-Ohno Y, Haze M, Yoshida Y, Hukushima K, Hasegawa Y, and Okada M 2016 J Phys Soc Jpn 85 093702.
[14] Chen S, Billings S A, and Luo W 1989, Int J Control 50 1873–1896.
[15] Pati Y C, Rezaifar R, and Krishnaprasad P S 1993 in Proceedings of 27th Asilomar Conference on Signals, Systems and Computers 40–44.
[16] Osborne M R, Presnell B, and Turlach B A 2000 IMA J Numer Anal 20 389–403.
[17] Efron B, Hastie T, Johnstone I, and Tibshirani R 2004 Ann Stat 32 407–499.
[18] Blumensath T and Davies M E 2008 J Fourier Anal Appl 14 629–654.
[19] Blumensath T and Davies M E 2009 Appl Comput Harmon Anal 27 265–274.
[20] Needell D and Tropp J A 2009 Appl Comput Harmon Anal 26 301–321.
[21] Dai W and Milenkovic O 2009 IEEE Trans Inf Theory 55 2230–2249.
[22] Foucart S 2011 SIAM J Numer Anal 49 2543–2563.
[23] Lopes M E 2013 in Proceedings of Machine Learning Research 28 217–225.
[24] Nakashish-Ohno Y and Hukushima K in preparation.
[25] Mosier C I 1951 Educ Psychol Meas 11 5–11; Cureton E E Ibid 12–15; Katzell R A Ibid 16–22; Wherry R J Ibid 23–28.
[26] Efron B and Gong G 1983 Am Stat 37 36–48.
[27] Kohavi R 1995 in Proceedings of 14th International Joint Conference on Artificial Intelligence 2 1137–1143.
[28] Chen S and Donoho D 1994 in Proceedings of 28th Asilomar Conference on Signals, Systems and Computers 41–44.
[29] Bloomfield P and Steiger W 1983 Least Absolute Deviations: Theory, Applications, and Algorithms Basel Birkhäuser.