

Iterative Projection Algorithms for Solving Inverse Problems

R. P. Millane

Department of Electrical and Computer Engineering
University of Canterbury
Private Bag 4800, Christchurch, New Zealand
Email: rick@elec.canterbury.ac.nz

Abstract—Inverse problems abound in the ocean sciences, as they do in almost every area of science and technology. Solutions to inverse problems are notoriously difficult, computationally expensive, and noise sensitive. Iterative projection algorithms have proved to be particularly useful for solving some image reconstruction problems with incomplete data. These techniques are outlined with a view to promoting their application in the ocean sciences.

I. INTRODUCTION

One usually collects data with the aim of obtaining meaningful and useful information on a physical system. Often however, the quantities that we wish to determine are different from the data that we are able to measure. The useful information on the physical system is described by a set of parameters that are often a function of position (often in two or more dimensions) and it is these parameters that one wishes to determine from the data. The parameters are often referred to as the “object,” or often we say that we wish to construct an “image” of the object. The words “object” and “image” are then often used interchangeably.

In practically all cases a mathematical–physical model is available that allows one to calculate (at least in principle) the data that would be measured from a given object (or image). This is often referred to as the “forward problem.” However, the problem at hand is usually to calculate the object from the data, which is referred to as the “inverse problem,” or the “image reconstruction problem.” Inverse problems abound in science and technology, and the ocean sciences are no exception. Determination of sound–speed profiles, or temperature profiles, from travel times; determination of wave–heights, or ocean surface winds, from radar backscatter; and determination of phytoplankton distributions from optical measurements, are a few of many examples.

Inverse problems present a number of difficulties. A mathematical–physical model is generally not available (in all except the simplest cases) that allows one to explicitly calculate the image from the data. The measured data are often *incomplete*, in the sense that a multitude of images could have produced the measured data, i.e. the solution is not unique. The problem is often *ill-posed*, in that small errors in the data (which are inevitable) can lead to large errors in the solution. These are all practical difficulties that have to be dealt with when solving an inverse problem.

There are a wide variety of methods used for solving inverse problems [1]. Suitable methods depend on the particular problem at hand, although all involve regularization to deal with missing data and noise sensitivity. An approach referred to as “iterative projection algorithms” has proved useful in a number of image reconstruction problems from incomplete data, particularly phase retrieval, due to their computational efficiency and their ability to escape from local minima [2]. In this paper I briefly review iterative projection algorithms for solving image reconstruction problems, with the goal of encouraging their application to inverse problems in the ocean sciences.

II. INVERSE PROBLEMS

Consider an object, or image, characterized by a parameter $f(\mathbf{y})$ which is a function of (say) the position vector $\mathbf{y}$. Let the measured data be $d(\mathbf{z})$ where $\mathbf{z}$ represents the position (say) at which the different measurements are made. The data are related to the image by a (generally nonlinear) forward operator $K[\cdot]$ such that

$$d(\mathbf{z}) = K[f(\mathbf{y})](\mathbf{z}) + n(\mathbf{z}) \quad (2.1)$$

where $n(\mathbf{z})$ represents the noise in the data.

In practice, the data are measured at a finite number M of discrete points $\mathbf{z}_i$, and the image is discretized onto a finite grid $\mathbf{y}_i$ of N points. It is convenient to represent the image by a vector $\mathbf{x}$ in an N –dimensional vector space S such that the coordinates correspond to the values at the *pixels* $\mathbf{y}_i$, i.e.

$$\mathbf{x} = (x_1, x_2, \dots, x_N) = (f(\mathbf{y}_1), f(\mathbf{y}_2), \dots, f(\mathbf{y}_N)). \quad (2.2)$$

The discrete data are denoted by the M –dimensional vector $\mathbf{d}$, i.e.

$$\mathbf{d} = (d(\mathbf{z}_1), d(\mathbf{z}_2), \dots, d(\mathbf{z}_M)). \quad (2.3)$$

The forward operator K is appropriately discretized such that (2.1) can be written in the discrete form

$$\mathbf{d} = K \mathbf{x} + \mathbf{n}. \quad (2.4)$$

The vector space formalism is useful for describing the method of iterative projections.

In general, the data $\mathbf{d}$ are incomplete so that additional *a priori* information are needed in order to obtain a unique (or usefully unique) solution, or estimate $\hat{\mathbf{x}}$ of $\mathbf{x}$. The *a priori*

information are constraints that one knows the image must satisfy. For example, a sound speed must be positive at all points, or further, in a particular situation, a feasible sound speed may have known upper and lower bounds. Such constraints limit the set of possible solutions.

Since only some of all possible images $\mathbf{x} \in S$ will satisfy a particular constraint A , the constraint can be represented by a subset A of S . We refer to this subset as the *constraint set* for the constraint A . Note that the data $\mathbf{d}$ also induce a constraint set, since only a subset of all images $\mathbf{x}$ will satisfy (2.4) within the noise level of the data. With sufficient (or sufficiently strong) constraints, the non-uniqueness due to incomplete data can be replaced by uniqueness. Assuming that sufficient *a priori* information is available, the inverse problem, or the image reconstruction problem, is then to find an image that satisfies all the constraints, i.e. to find an image that is in the intersection of all the constraints sets. Since the dimensionality of the space S may be very large (say $N = 10^6$), finding a point in the intersection is non-trivial. Iterative projection algorithms are a methods for seeking to find feasible solutions to image reconstruction problems by finding such a point.

III. CONSTRAINTS AND PROJECTIONS

A. Convex Constraints

A special role is played by constraints that are *convex* [2]. Convex constraints are represented by convex constraint sets. A convex set is one for which the line joining any two points in the set is totally within the set (Fig. 1a). An image with a given *support* is an example of a convex constraint. The support of an image is the region where it is nonzero. In many cases an image is known to be zero (or to not exist) outside a certain known region of space, and this information represents a support constraint. This constraint is characterized by certain pixels in the image being zero, i.e. certain coordinates in S are zero. The constraint set is therefore a hyperplane in S , which has the property of convexity (Fig. 1b). Another example of convex constraint is an image whose energy is bounded above; the constraint set is then a hypersphere in S that is centered on the origin and with radius equal to the square-root of the maximum energy.

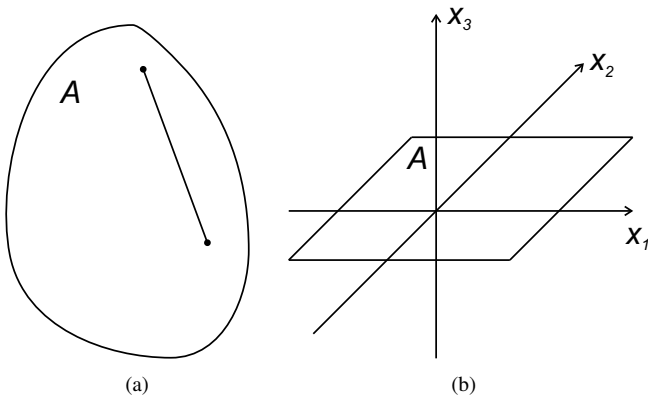


Fig. 1. Convex sets A : (a) general geometry, and (b) the convex set for a support constraint for which $x_3 = 0$.

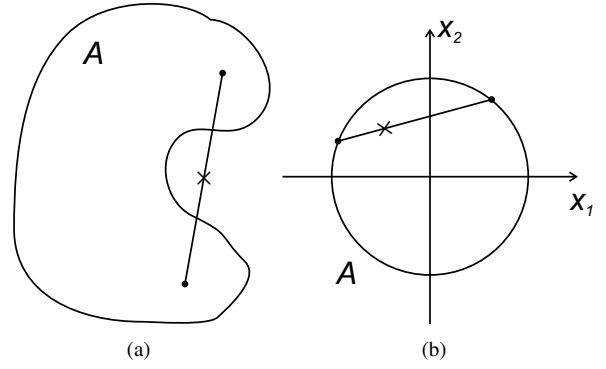


Fig. 2. Nonconvex sets A : (a) general geometry, and (b) the nonconvex set for a specified image energy. (b) Also describes the Fourier magnitude constraint (see text).

B. Nonconvex Constraints

Nonconvex constraints are obviously those that are not convex (Fig. 2a). An image that has a given energy is a nonconvex constraint since it is represented by the surface of a hypersphere in S (Fig. 2b).

C. Projections

We take the vector space S to be a Euclidean space which allows us to define a “distance” between two images $f_1(\mathbf{y})$ and $f_2(\mathbf{y})$ as being the Euclidean distance $\|\mathbf{x}_1 - \mathbf{x}_2\|$ between the corresponding points $\mathbf{x}_1$ and $\mathbf{x}_2$ in S , where

$$\|\mathbf{x}_1 - \mathbf{x}_2\|^2 = \sum_{i=1}^N (x_{1i} - x_{2i})^2. \quad (3.1)$$

A *projection* $\mathbf{x}_2 = P_A \mathbf{x}_1$ is an operation on an image $\mathbf{x}_1$ and a constraint set A that produces an image $\mathbf{x}_2$ such that

$$\|\mathbf{x}_2 - \mathbf{x}_1\| = \min_{\mathbf{x}_2 \in A} \|\mathbf{x}_2 - \mathbf{x}_1\|, \quad (3.2)$$

i.e. the projection of $\mathbf{x}_1$ onto A is the point $\mathbf{x}_2$ in A that is closest to $\mathbf{x}_1$ (Fig. 3a) [2]. The usefulness of a projection in the context of an inverse problem is that if we have an estimate $\hat{\mathbf{x}}$ of the solution, then $P_A \hat{\mathbf{x}}$ may be a better estimate since it is close to $\hat{\mathbf{x}}$ and satisfies the constraint A . An important property of convex sets is that the projection $P_A \mathbf{x}$ is unique for any

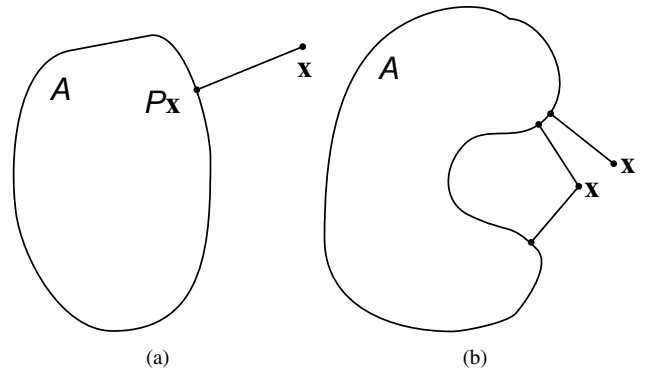


Fig. 3. Projections onto (a) convex and (b) nonconvex sets.

$\mathbf{x}$ [2]. For nonconvex sets, the projection is not necessarily unique for any $\mathbf{x}$, although in many cases it is unique for most points $\mathbf{x}$. Projections onto convex and nonconvex sets are illustrated in Fig. 3.

IV. THE PHASE RETRIEVAL PROBLEM

Phase retrieval is an image reconstruction problem that occurs in a variety of imaging applications [3]. This paper is not concerned with phase retrieval, however phase retrieval is an area where iterative projection algorithms have been applied with considerable success. The phase retrieval problem has a rather simple setting and is therefore a useful example with which to illustrate some of the following discussion. It is outlined briefly here.

A phase retrieval problem occurs where one measures the amplitude $d(\mathbf{z})$ of the Fourier transform of an image $f(\mathbf{y})$ and it is desired to reconstruct $f(\mathbf{y})$ from $d(\mathbf{z})$. If the phase of the Fourier transform were measured, in addition to the amplitude, then the image could be straightforwardly reconstructed by inverse Fourier transformation. The phase is therefore the missing data, and if the data were complete an explicit inversion formula is available. Additional information is therefore needed to solve the phase problem. A variety of kinds of *a priori* information are utilised, depending on the particular problem at hand. We consider a single constraint, a support constraint, to simplify the discussion.

The problem has two constraints which we denote A and B . Constraint set A contains all images that have the specified support and, as described above, this is a convex constraint. Constraint set B contains all images whose Fourier transform has the specified magnitude function (the data). Since the Fourier transform is distance preserving, the geometry of the constraint set can be viewed in Fourier space. For a particular data sample, the real and imaginary parts form two coordinates and the magnitude constraint corresponds to a circle in the plane formed by these two coordinate axes (see Fig. 2b). The constraint set B is therefore nonconvex. The projection onto A is given by

$$P_A \mathbf{x} = \mathbf{x} * \mathbf{a} \quad (4.1)$$

where the vector $\mathbf{a}$ has elements equal to 1 corresponding to pixels within the support region and zero elsewhere, and $*$ denotes element-by-element multiplication. The projection onto B is given by

$$P_B \mathbf{x} = F^{-1}[\mathbf{d} \exp(\phi[F[\mathbf{x}]])] \quad (4.2)$$

where $F[\cdot]$ and $F^{-1}[\cdot]$ denote Fourier transformation and inverse Fourier transformation, respectively, $\phi[\cdot]$ denotes the phase, and $\mathbf{d}$ is the Fourier magnitude data. It is easily verified that (4.1) and (4.2) give the projections as defined in section III.

It is worth noting that the phase problem has a rather simple and desirable structure. This is due to an explicit inversion formula (F^{-1}) being available if the data are complete, which results in an explicit expression being available for the projection onto the data (4.2). We will assume this desirable

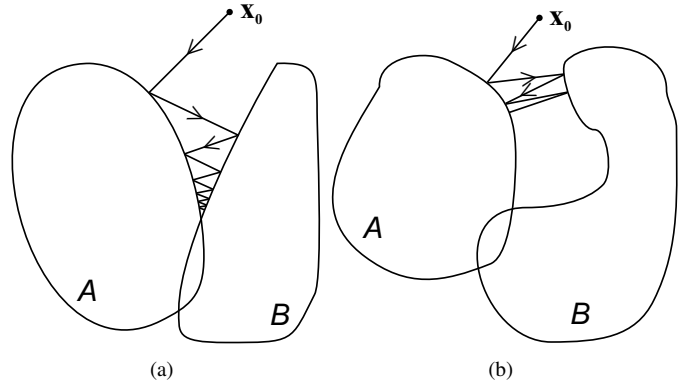


Fig. 4. Geometric illustrations of (a) the projection onto convex sets (POCS) algorithm, and (b) the generalized projection algorithm (GPA) converging to a limit cycle.

situation in the following, and cases where this is not the case will be discussed in section VI.

V. ITERATIVE PROJECTION ALGORITHMS

Solving the inverse amounts to finding a point in the intersection of all the constraint sets. The idea of iterative projection algorithms is to repeatedly project onto the different constraint sets in an attempt to converge towards a point in their intersection. If there are two constraints A and B , then the algorithm takes the form [2]

$$\hat{\mathbf{x}} = \cdots P_B P_A P_B P_A \mathbf{x}_0 \quad (5.1)$$

where $\mathbf{x}_0$ is some chosen starting image. The iterations are stopped when the estimate $\hat{\mathbf{x}}$ satisfies all the constraints, or in practice when the constraints are satisfied within an error that is acceptable given the errors in the data, or when further iterations do not lead to any better fit to the constraints. The properties of the iteration (5.1) depend on the convexity of the constraint sets as described below.

A. Projection onto Convex Sets Algorithm

If the constraint sets A and B are both convex, then the algorithm (5.1) is referred to the *projection onto convex sets* (POCS) algorithm [4]. The POCS algorithm has the property that it converges to a point $\mathbf{x}^*$ in the intersection of A and B [2], i.e.

$$\hat{\mathbf{x}} \rightarrow \mathbf{x}^* \in A \cap B. \quad (5.2)$$

This is obviously a desirable property, although it says nothing about the rate of convergence, i.e. the number of iterations that are necessary before $\hat{\mathbf{x}}$ is usefully close to $\mathbf{x}^*$. In some cases converge is quite rapid and in other cases it is unacceptably slow. Convergence of the POCS algorithm is illustrated geometrically in Fig. 4a. This convergence property of POCS is the most important aspect of convex constraints and is why they are singled out for special attention.

The convergence property of the POCS algorithm is retained if there are more than two (but a finite number of) constraints $A, B, \dots, Q$, i.e. the algorithm

$$\hat{\mathbf{x}} = \cdots P_Q \cdots P_B P_A P_Q \cdots P_B P_A \mathbf{x}_0 \quad (5.3)$$

converges to a point $\mathbf{x}^* \in A \cap B \cap \dots \cap Q$.

B. Relaxed Projection onto Convex Sets Algorithm

A relaxed projection T_A is a generalization of the projection defined above and is given by [2]

$$T_A \mathbf{x} = \lambda P_A \mathbf{x} + (1 - \lambda) \mathbf{x} \quad (5.4)$$

where λ is a constant called the relaxation parameter such that $0 < \lambda < 2$. Note that the point $T_A \mathbf{x}$ is not necessarily in A . A geometric interpretation of a relaxed projection is shown in Fig. 5. Note that the relaxed projection is the same as the projection if $\lambda = 1$. The convergence property described above for the POCS algorithm also applies to a relaxed projection algorithm in which the projections are replaced by relaxed projections (although still onto convex sets), i.e. the algorithm

$$\hat{\mathbf{x}} = \dots T_Q \dots T_B T_A T_Q \dots T_B T_A \mathbf{x}_0 \quad (5.5)$$

converges to $\mathbf{x}^* \in A \cap B \cap \dots \cap Q$ if $A, B, \dots, Q$ are convex sets. The advantage of using relaxed projections is that often the rate of convergence can be improved by using $\lambda \neq 1$. Although there is much mathematical theory concerned with the choice of optimal values for λ , in practice one has to experiment with different values. In many cases values $\lambda > 1$ improves the rate of convergence, with $\lambda = 1.5 - 1.9$ often being a good choice. Values too close to 2 can give oscillatory behaviour and can slow convergence. In some cases, although less often, values $\lambda < 1$ can be useful.

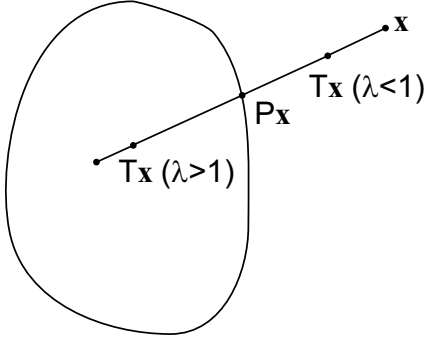


Fig. 5. Geometric interpretation of a relaxed projection.

C. Generalized Projection Algorithm

Although the POCS algorithm has desirable convergence properties, many practical problems are subject to nonconvex constraints. This presents two difficulties: (1) The projections are not necessarily unique, and (2) an iterative projection algorithm of the kind (5.3) or (5.5) may not converge to a point in the intersection of the constraint sets.

The first difficulty is generally not too serious. In practice, although some points will not project uniquely onto a nonconvex set, many points will. For those that do not, a unique projection can usually be obtained by using some other physically-motivated requirement to select one of the multiple projections. When this is done, the single projection that is chosen is referred to as a *generalized projection*. Applying

generalized projections, or relaxed generalized projections, when at least one of the constraint sets is nonconvex, in an iterative algorithm as in (5.3) or (5.4), is referred to as a *generalized projection algorithm* (GPA), and is clearly the analog of POCS when not all constraints are convex [5].

The second problem is more serious and is illustrated in Fig. 4b. As a result of the nonconvexity of at least one constraint, the GPA may not converge to a point in the intersection [2]. In particular, the convergence behaviour depends on the starting point $\mathbf{x}_0$. The algorithm may converge to a point in the intersection, converge to a point not in the intersection, converge to a limit cycle, or diverge, depending on the particular constraints and the starting point. Convergence to a limit cycle that is not close to the solution, often referred to as *stagnation*, is a problem that plagues the GPA.

For the case of two constraint sets, the relaxed GPA has the property that the sum of the squared distances of $\hat{\mathbf{x}}$ to each of the constraint sets is nonincreasing [2]. This is a desirable property in that the overall agreement with the constraints improves with more iterations. However, this may not correspond to movement towards a point in the intersection of the constraint sets, so is not particularly useful in practice. This property does not hold for more than two constraint sets. However, any number of constraint sets can be combined into a single constraint set (by set intersection) so that, in principle, the number of constraint sets can be reduced to two and the above property then holds. However, the sets become more complex, as do the projection operators, when sets are combined. (Note that combining convex constraints generally gives a nonconvex constraint).

Inverse problems with convex constraints fall into the field of convex optimization for which there is substantial mathematical theory [6]. However, for nonconvex optimization rigorous results are scarce. Despite the poor theoretical understanding of the nonconvex problem, variants of the GPA have been applied to various problems, sometimes with considerable success. Some of the variants of the GPA that are quite successful in avoiding stagnation problems are described in the following sections.

D. Fienup's Hybrid-Input-Output Algorithm

One of the first applications of iterative projection algorithms to image reconstruction was the Gerchberg-Saxton algorithm for the phase retrieval problem with support and positivity constraints on the image as well as the Fourier modulus constraint [7]. This corresponds to an (unrelaxed) GPA since, as noted above, the Fourier modulus constraint is nonconvex. Although this algorithm is sometimes successful, it often suffers from stagnation.

In an important breakthrough occurred in 1982 when Fienup described the *hybrid input-output* (HIO) algorithm for phase retrieval, based on ideas from control theory [8]. Although the basis for the HIO algorithm was unclear, it showed an uncanny ability to escape from stagnation, and has become an important practical algorithm for solving phase retrieval problems [9]. One iteration of the HIO algorithm can be put

in the form

$$\mathbf{x}^{(n+1)} = P_A P_B \mathbf{x}^{(n)} + (\mathbf{x}^{(n)} - \beta P_B \mathbf{x}^{(n)}) * \bar{\mathbf{a}} \quad (5.6)$$

where P_A and P_B are the support and Fourier modulus projections, respectively, as described in section IV, the superscript (n) denotes the n -th iteration, $\bar{\mathbf{a}}$ is the complement of $\mathbf{a}$ (i.e. has elements equal to 1 outside the support region and 0 inside the support region), and β is a parameter between 0 and 1 ($\beta \approx 0.7 - 1.0$ is often most effective). It has recently been shown that the HIO algorithm for $\beta = 1$ is identical to the Douglas–Rachford algorithm of classical convex optimization [6]. Positivity was also incorporated into the HIO algorithm but this is not discussed here. Note that removal of the second term in (5.6) reduces the HIO algorithm to the GPA. The HIO algorithm has been found to be particularly effective in providing good convergence in solving the phase retrieval problem, particularly if a number of iterations (say 20–50) of HIO are alternated with some iterations (say 10) of the (unrelaxed) GPA [8, 9]. Since the output of HIO iteration is not necessarily a good estimate of the solution, the algorithm is always terminated with a few cycles of the GPA.

E. Millane and Stroud’s Generalized HIO Algorithm

The HIO algorithm as originally formulated applies strictly only to a support constraint (with positivity), together with a data constraint (Fourier modulus in this case). The theme of the HIO algorithm can be easily extended to the case of general constraints [10] and can be put in the form [11]

$$\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} + [(1 + \beta)P_A P_B - P_A - \beta P_M] \mathbf{x}^{(n)} \quad (5.7)$$

which we refer to as the *generalized HIO* (GHIO) algorithm. Millane and Stroud [10] applied this algorithm to a phase retrieval problem in which the Fourier magnitude data is undersampled, and this further loss of data is compensated for by symmetry information in the image in addition to support information. A significant feature of this application is that the symmetry constraint, and therefore the corresponding projection, depends on the current estimate of the image; see Ref. 10 for details.

Results using simulated, undersampled Fourier magnitude data for an icosahedral virus are shown in Fig. 6 [10]. The plots show the error in the recovered Fourier phase as a function of iteration number for the GPA (top) and the GHIO algorithm (bottom). The striking ability of the GHIO algorithm to break the stagnation of the GPA is evident.

F. The Hybrid Projection–Reflection Algorithm

Recently, Bauschke, Combettes and Luke [12] have described a phase retrieval algorithm they call the *hybrid projection–reflection* (HPR) algorithm. This algorithm results from noting a key difference between Fienup’s HIO algorithm with $\beta = 1$ and the Douglas–Rachford algorithm of convex optimization if the constraint is nonlinear (i.e. Px_i is a nonlinear equation in x_i). Since positivity is a nonlinear constraint, the HPR algorithm is a slightly different variant of the HIO algorithm in the way that positivity is incorporated; see Ref. 12

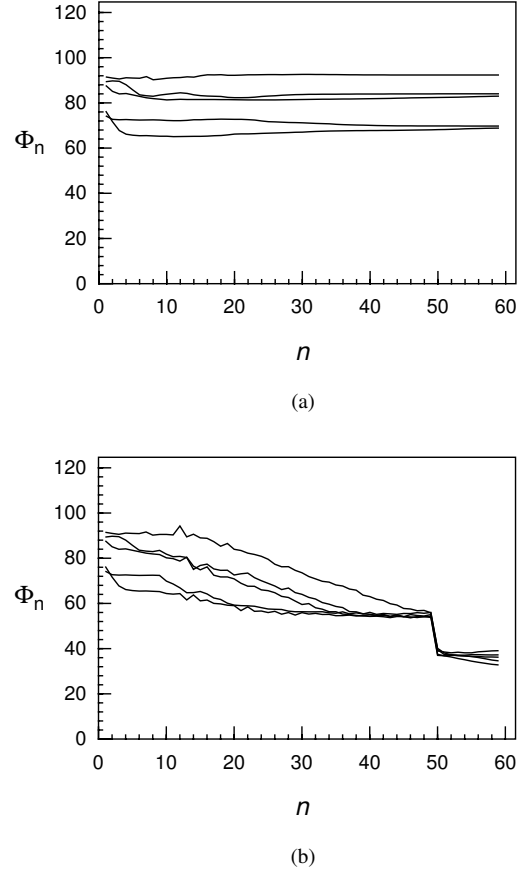


Fig. 6. Weighted phase error for reconstruction of an icosahedral virus incorporating 5-fold symmetry constraints using (a) the GPA, and (b) the GHIO algorithm (10 iterations of GPA, 40 iteration of GHIO and 10 iterations of GPA). Results are shown for five different starting images. See Ref. 10 for details.

for details. The HPR algorithm reduces to the HIO algorithm for a support constraint only, and to the GHIO algorithm for a general linear constraint.

Simulations show that the HPR algorithm converges to a smaller error than the HIO algorithm (with no GPA iterations) for noise-free data, although the errors are similar for noisy data [12]. A possible advantage of the HPR algorithm is that there is no need to alternate it with GPA iterations, leading to an algorithm that requires less user intervention.

G. Elser’s Difference Map Algorithm

Recent work by Elser using a somewhat different approach has led to an exciting new iterative projection algorithm that he calls the *difference map* (DM) algorithm [13]. The DM algorithm takes the form

$$\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} + \beta(P_A[(1 + \beta^{-1})P_B \mathbf{x}^{(n)} - \beta^{-1} \mathbf{x}^{(n)}] - P_B[(1 - \beta^{-1})P_A \mathbf{x}^{(n)} + \beta^{-1} \mathbf{x}^{(n)}]), \quad (5.8)$$

where $\beta \neq 0$ is a parameter. Once the algorithm has converged to the point $\mathbf{x}'$, the solution is found by projecting $\mathbf{x}'$ onto P_A or P_B . The DM algorithm with $\beta = 1$ in (5.8) reduces to the HIO algorithm (with $\beta = 1$ in (5.6)) for a support

constraint [13], and reduces to the GHIO algorithm (with $\beta = 1$ in (5.7)) for a general linear constraint A [11]. As with the HPR algorithm, the DM algorithm treats nonlinear constraints differently than do the HIO and GHIO algorithms.

Elser has applied the DM algorithm to a variety of phase retrieval, and other, problems with considerable success [13]. Although further study of this algorithm is needed, it appears to have considerable potential.

VI. DISCUSSION

The primary advantages of iterative projection algorithms for solving inverse problems are: (1) they are computationally efficient compared to many other optimization methods (the projection operators are fast to compute), (2) they (the projection operators) are relatively simple to implement, and (3) the more advanced algorithms are quite resistant to becoming trapped in local minima (stagnation). The first two characteristics are important for large problems, and of course the third characteristic is critically important for nonconvex problems.

As implied in the previous sections, although projection algorithms have been applied to a number of problems in signal and image recovery, most studies of the more advanced techniques have been in phase retrieval. The techniques are more widely applicable however, and may provide useful solutions to other inverse problems, including some in the ocean sciences.

Rigorous theory and understanding of the algorithms is still sorely lacking. Among issues worthy of attention are: (1) how should multiple (more than two) constraint sets be incorporated into some of these algorithm in an optimum way, (2) how should noise in the data be accounted for, and (3) what is the best error metric with which to measure convergence (a thorny problem for the nonconvex case that has not been addressed here)?

In the case of phase retrieval an explicit expression for the projection onto the data (4.2) is possible because of a closed form expression for the inverse operator *if* the data are complete. Although such cases occur in numerous other inverse problems, in many cases the inverse operator is not so easily formulated. The usefulness of these techniques when this is not the case is really an open question. In many cases a local approximation to the inverse operator may be obtainable and used to formulate the projection onto the data. In such a case the algorithm might be expected to be more sensitive to the starting point. Computing such an approximation would increase the computational cost, although the potential of projection algorithms to escape local minima could still make the approach worthwhile.

In conclusion, iterative projection algorithms have been very successful in phase retrieval and other image reconstruction problems, and emerging algorithms have the potential to solve other difficult inverse problems. This potential should be explored by practitioners in other fields.

REFERENCES

- [1] M. Betero and P. Boccacci, *Introduction to inverse problems in imaging*. Institute of Physics Publishing, London, 1998.
- [2] H. Stark and Y. Yang, *Vector space projections: A numerical approach to signal and image processing, neural nets, and optics*. John Wiley & Sons, New York, 1998.
- [3] R. P. Millane, "Phase retrieval in crystallography and optics," *J. Opt. Soc. Am. A*, vol. 7, pp. 394–411, March 1990.
- [4] D. C. Youla and H. Webb, "Image restoration by the method of convex projections: Part I—Theory," *IEEE Trans. Med. Imaging*, vol. MI-1, pp. 81–94, 1982.
- [5] A. Levi and H. Stark, "Image restoration by the method of generalized projections with application to restoration from magnitude," *J. Opt. Soc. Am. A*, vol. 1, pp. 932–943, 1984.
- [6] H. H. Bauschke, P. L. Combettes, and D. R. Luke, "Phase retrieval, error reduction algorithm, and Fienup variants: a view from convex optimization," *J. Opt. Soc. Am. A*, vol. 19, pp. 1334–1345, July 2002.
- [7] R. W. Gerchberg and W. O. Saxton, "A practical algorithm for the determination of phase from image and diffraction plane pictures," *Optik*, vol. 35, pp. 237–246, 1972.
- [8] J. R. Fienup, "Phase retrieval algorithms: a comparison," *Appl. Opt.*, vol. 21, pp. 2758–2769, 1982.
- [9] J. C. Dainty and J. R. Fienup, "Phase retrieval and image reconstruction for astronomy," in *Image Recovery: Theory and Application*, H. Stark (Ed.), pp. 231–275, Academic Press, London, 1987.
- [10] R. P. Millane and W. J. Stroud, "Reconstructing symmetric images from their undersampled Fourier intensities," *J. Opt. Soc. Am. A*, vol. 14, pp. 568–579, March 1997.
- [11] R. P. Millane, in preparation.
- [12] H. H. Bauschke, P. L. Combettes, and D. R. Luke, "Hybrid projection–reflection method for phase," *J. Opt. Soc. Am. A*, vol. 20, pp. 1025–1034, June 2003.
- [13] V. Elser, "Phase retrieval by iterated projections," *J. Opt. Soc. Am. A*, vol. 20, pp. 40–55, January 2003.