Phase-retrieval stagnation problems and solutions

J. R. Fienup and C. C. Wackerman

Environmental Research Institute of Michigan, P.O. Box 8618, Ann Arbor, Michigan 48107

Received November 25, 1985; accepted July 3, 1986

The iterative Fourier-transform algorithm has been demonstrated to be a practical method for reconstructing an object from the modulus of its Fourier transform (i.e., solving the problem of recovering phase from a single intensity measurement). In some circumstances the algorithm may stagnate. New methods are described that allow the algorithm to overcome three different modes of stagnation: those characterized by (1) twin images, (2) stripes, and (3) truncation of the image by the support constraint. Curious properties of Fourier transforms of images are also described: the zero reversal for the striped images and the relationship between the zero lines of the real and imaginary parts of the Fourier transform. A detailed description of the reconstruction method is given to aid those employing the iterative transform algorithm.

1. INTRODUCTION

In a number of different disciplines, including astronomy, wave-front sensing, x-ray crystallography, and holography, one encounters the phase-retrieval problem: Given the modulus $|F(u, v)|$ of the Fourier transform

$$F(u, v) = |F(u, v)|\exp[i\psi(u, v)] = \mathcal{F}[f(x, y)]$$
$$= \iint_{-\infty}^{\infty} f(x, y)\exp[-i2\pi(ux + vy)]\mathrm{d}x\mathrm{d}y \qquad (1)$$

of an object $f(x, y)$, reconstruct the object $f(x, y)$ or, equivalently, reconstruct the Fourier phase $\psi(u, v)$. (Here and throughout this paper functions represented by upper-case letters are the Fourier transforms of the functions represented by the corresponding lower-case letters.) Because the autocorrelation of the object is given by $\mathcal{F}^{-1}[|F(u, v)|^2]$, this is equivalent to reconstructing an object from its autocorrelation. Many solutions to this problem have been proposed.[1-12] The method of solution that we believe is most practical from the point of view of minimum computational complexity and minimum sensitivity to noise and applicability under the most general assumptions is the iterative Fourier-transform algorithm.[1-3]

The iterative transform algorithm, a descendant of the Gerchberg–Saxton algorithm,[13-15] bounces back and forth between the object domain, where *a priori* knowledge about the object such as nonnegativity or its support is applied (the support is the set of points over which the object is nonzero), and the Fourier domain, where the measured Fourier modulus data is applied. The algorithm is reviewed in Section 2. Although the algorithm works well for many cases of interest, there is no guarantee that it will converge to a solution.

For certain types of objects the iterative algorithm can stagnate on images that are not fully reconstructed. Stagnation of the algorithm means that the output image changes very little after many further iterations while not at a solution. A solution is any Fourier-transform pair that satisfies the measured data and constraints in both domains with an error metric (defined later) no greater than the expected root-mean-squared (rms) error of the measured data. This paper will describe three such conditions of stagnation and algorithms that we have developed to jump each of these stagnation hurdles, allowing the algorithm to move on toward a solution. The three modes of stagnation are those characterized by (1) simultaneous twin images, (2) stripes superimposed upon the image, and (3) unintentional truncation by the support constraint.

The first stagnation problem results from the fact that an object $f(x, y)$ and its twin $f^*(-x, -y)$ (the complex conjugated object rotated by 180 deg) have the same Fourier modulus and, for cases in which the support of the object is symmetric with respect to this rotation, have the same support. If the iterative algorithm starts from an initial guess of random numbers, then there is an equal probability that it will reconstruct either of these two objects. The problem arises when, during the initial stages of reconstruction, features of both objects are reconstructed simultaneously. If this situation continues and features of both objects become equally strong (in the sense that applying the constraints does not favor one over the other), then the iterative algorithm may stagnate. Unable to suppress one twin image and converge to the other, the algorithm tries to reconstruct both together and goes nowhere.

Because the Fourier transform of the linear combination $tf(x, y) + (1 - t)f^*(-x, -y)$ does not have modulus $|F(u, v)|$ except for $t = 1$ or 0 [assuming that $f^*(-x, -y) \neq f(x, y)$], an image output by the algorithm in this condition is not a solution that is consistent with the data. This is recognized by the algorithm from the fact that the error metrics (defined in Section 2) are nonzero (or, in the presence of noise, greater than the expected rms error of the data) in this circumstance. This mode of stagnation often occurs if the support of the object is centrosymmetric. An example of twin-image stagnation is displayed in Fig. 2(b) of Ref. 16. A method for overcoming this stagnation problem is described in Section 3.

The second stagnation problem is characterized by an output image that looks much like the true object but with a pattern of stripes superimposed. The pattern of stripes is approximately sinusoidal in one direction and constant in the orthogonal direction. The stripes are usually of low contrast and therefore are not objectionable, but they occa-

0740-3232/86/111897-11$02.00

sionally are of sufficiently high contrast to be disturbing. They are stronger over the support of the object and weaker outside the support. This problem frequently occurs to varying degrees. Examples of its occurrence are Figs. 3(f), 3(i), 4(a), and 4(b) of Ref. 1, Figs. 9(a)–9(d) of Ref. 17, and, to a lesser extent, Fig. 2(b) of Ref. 16. The error metric is nonzero when the stripes are present since they extend (although with lower contrast) outside the known support of the object, and so the striped images are not a solution and do not represent a uniqueness problem. Methods for overcoming this mode of stagnation are given in Section 4. During the study of the stripes phenomenon, some interesting properties of the Fourier transforms were discovered and are described in Section 4.

The third stagnation problem arises when the support constraint is used in a manner that is inconsistent with the partially reconstructed image output by the algorithm. If the partially reconstructed image is in a position that is translated relative to the position of the mask array defining the support constraint, then the object-domain step of the algorithm will inadvertently tend to truncate (cut off spatially) part of the image. This usually results in stagnation of the algorithm. A previously reported method of reducing the likelihood of encountering this problem[3] is described in Section 6. A new method for overcoming this stagnation problem is introduced in Section 5.

As an aid to the practical implementation of the iterative transform algorithm, Section 6 discusses a number of helpful hints that make it converge more reliably. This description should help those employing the iterative transform algorithm to achieve greater success and to convert some of the "black art" of the iterative approach into a more automatic algorithm. Section 7 contains a summary and conclusions.

2. REVIEW OF THE ITERATIVE TRANSFORM ALGORITHM

When working with sampled data on a digital computer, one employs the discrete Fourier transform (DFT)

$$F(u) = \sum_{x=0}^{N-1} f(x)\exp(-i2\pi ux/N) \tag{2}$$

and its inverse

$$f(x) = N^{-2} \sum_{u=0}^{N-1} F(u)\exp(i2\pi ux/N), \tag{3}$$

which can be computed by using the fast-Fourier-transform (FFT) method. Here we employ u and x as two-dimensional vectors, $x = (x_1, x_2)$, $u = (u_1, u_2)$, where u_1, u_2, x_1, and $x_2 = 0, 1, 2, \ldots, N-1$ (square arrays are assumed for simplicity). In order to avoid aliasing in the computation of $|F(u)|^2$, we restrict $f(x)$ to be zero for $x_1 \geq N/2$ and for $x_2 \geq N/2$. Therefore we are considering only problems for which the object has finite support. For problems in astronomy $f(x)$ is a real, nonnegative function, but for other problems $f(x)$ may be complex valued. This paper assumes the case of real, nonnegative objects (particularly in the discussion of stripes), although much of the discussion can also be applied to the more general case of complex-valued images.

The simplest version of the iterative transform algorithm, known as the error-reduction algorithm,[1] follows the philosophy of the Gerchberg–Saxton algorithm. It can be viewed in a number of different ways: in terms of the method of successive approximations,[18] as a form of steepest-descent gradient search,[3] or as a projection onto sets in a Hilbert space (the Fourier modulus constraint being onto a nonconvex set, however, so convergence is not assured).[19]

For the most general problem, the error-reduction algorithm consists of the following four steps (for the kth iteration): (1) Fourier transform $g_k(x)$, an estimate of $f(x)$, yielding $G_k(u)$; (2) make the minimum changes in $G_k(u)$ that allow it to satisfy the Fourier-domain constraints to form $G_k'(u)$, an estimate of $F(u)$; (3) inverse Fourier transform $G_k'(u)$, yielding $g_k'(x)$, the corresponding image; and (4) make the minimum changes in $g_k'(x)$ that allow it to satisfy the object-domain constraints to form $g_{k+1}(x)$, a new estimate of the object. For phase retrieval from a single intensity measurement, in which the Fourier modulus $|F(u)|$ is the square root of the intensity, these four steps are

$$G_k(u) = |G_k(u)|\exp[i\phi_k(u)] = \mathcal{F}[g_k(x)], \tag{4}$$

$$G_k'(u) = |F(u)|\exp[i\phi_k(u)], \tag{5}$$

$$g_k'(x) = \mathcal{F}^{-1}[G_k'(u)], \tag{6}$$

$$g_{k+1}(x) = \begin{cases} g_k'(x), & x \notin \gamma \\ 0, & x \in \gamma \end{cases}, \tag{7}$$

where γ is the set of points at which $g_k'(x)$ violates the object-domain constraints and where g_k, G_k', and ϕ_k are estimates of f, F, and the phase ψ of F, respectively. The algorithm is typically started by using an array of random numbers for $g_0(x)$ or for $\phi_0(u)$. Figure 1 shows a block diagram of the iterative transform algorithm.

For the astronomy problem, the object-domain constraints are the object's nonnegativity and a (usually loose) support constraint. The diameter of the object can be computed since it is just half the diameter of the autocorrelation; however, the exact support of the object in general cannot be determined uniquely from the support of the autocorrelation,[20] and so the support constraint cannot be applied tightly. For other problems, one may not have a nonnegativity constraint but have *a priori* knowledge of a tighter support constraint.[21]

For the problem of phase retrieval from two intensity measurements, $g_k'(x) = |g_k'(x)|\exp[i\theta_k'(x)]$ is complex valued, and Step (4) becomes

$$g_{k+1}(x) = |f(x)|\exp[i\theta_{k+1}(x)] = |f(x)|\exp[i\theta_k'(x)], \tag{8}$$

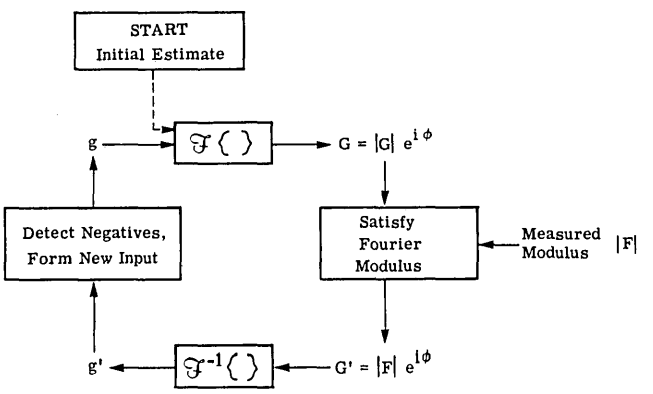

Fig. 1. Block diagram of the iterative transform algorithm.

where $|f(x)|$ is the known modulus of the complex-valued object and θ_k is an estimate of the phase of the object. With this modulus constraint in the object domain, the error-reduction algorithm is precisely the Gerchberg–Saxton algorithm. In this paper we consider only the problem of phase retrieval from a single intensity measurement.

A measure of the convergence of the algorithm to a solution (a Fourier-transform pair satisfying all the constraints in both domains) is the squared-error metric in the Fourier domain,

$$E_F^2 = N^{-2} \sum_u [|G(u)| - |F(u)|]^2, \tag{9}$$

or in the object domain,

$$E_0^2 = \sum_{x \in \gamma} |g_k'(x)|^2, \tag{10}$$

where γ is defined as in Eq. (7). Values of the error metrics mentioned later are the square roots of these expressions divided by $\Sigma_x |g_k'(x)|^2$, i.e., normalized root-mean-squared (nrms) errors. It can be shown that the error-reduction algorithm converges in the sense that the squared error cannot increase with an increasing number of iterations.[3]

Although it works well for the problem of phase retrieval from two intensity measurements, the error-reduction algorithm usually converges slowly for the problem of phase retrieval from a single intensity measurement being considered here.[3] Several modifications of the iterative transform algorithm were made and tested, and most of them converged faster than the error-reduction algorithm.[3] To date, the most successful version is the hybrid input–output algorithm, which replaces Step (4) of the algorithm by[1,3]

$$g_{k+1}(x) = \begin{cases} g_k'(x), & x \notin \gamma, \\ g_k(x) - \beta g_k'(x), & x \in \gamma, \end{cases} \tag{11}$$

where β is a constant feedback parameter. Values of β between 0.5 and 1.0 work well. When the hybrid input–output algorithm is used, $g_k(x)$ is no longer an estimate of $f(x)$; it is, instead, the input function used to drive the output $g_k'(x)$ [which is an estimate of $f(x)$] to satisfy the constraints. Hence only the object-domain error E_0 is meaningful.[3] When the hybrid input–output algorithm is used, even E_0 does not always correlate with image quality as well as one would like. For this reason one may prefer to perform a number of cycles of iterations, in which one cycle consists of, say, 20 to 50 iterations of the hybrid input–output algorithm followed by 5 to 10 iterations of the error-reduction algorithm, and note E_0 only at the end of a cycle.[3]

See Ref. 3 for a more complete description of the iterative algorithm. Additional details concerning the implementation of the algorithm are given in Section 6. A description of the algorithm as it applies to a number of different problems is given in Ref. 18.

3. METHOD FOR OVERCOMING SIMULTANEOUS TWIN IMAGES

Figure 2(A) shows a real, nonnegative object, $f(x)$, which has centrosymmetric (square) support, and Fig. 2(B) shows the conjugate or twin image, $f^*(-x)$, which has the same Fourier modulus, $|F(u)|$. Since the object is real valued, $f^*(-x) = f(-x)$. Figure 2(C) shows the output image of the iterative transform algorithm after a few hundred iterations using both a nonnegativity constraint and a support constraint consisting of the actual (assumed to be known *a priori*) square support of the object. On close inspection of Fig. 2(C), it is seen that features of both $f(x)$ and $f^*(-x)$ are present (to see it, it helps to turn the page upside down). Often a few additional cycles of iterations are all that is needed to converge to one or the other of the twin images. An example of this is the sequence of output images shown in Figs. 6(c)–6(f) of Ref. 2. However, in the case of Fig. 2(C) (although with a very large number of further iterations it may be possible to move away from this output having both twin images), this output image represents a fairly stable condition of stagnation. Like the fabled donkey that starved to death standing between two bales of hay because it was unable to decide which of the two to eat, the algorithm is not readily able to move farther from the features of either of the twin images, and so it is also prevented from moving closer to one rather than the other.

We have devised a method for getting beyond this condition: the reduced-area support constraint method, which consists of the following steps:

(1) Replace the current correct mask defining the support constraint with a temporary one that (a) covers only a subset of the correct support including at least one of its edges and (b) has no 180° rotational symmetry (is not centrosymmetric).
(2) Perform a few iterations with the temporary mask.
(3) Replace the temporary mask with the correct one and continue with the iterations.

The reduced-area support constraint method is illustrated by the example shown in Fig. 3. Figure 3(A) shows the stagnated output image of Fig. 2(C) used as the input image to the algorithm. It had an error metric $E_0 = 0.027$. The correct support is a square. Figure 3(B) shows the reduced-area temporary support mask used for 10 error-reduction iterations. Figure 3(C) shows the output image after the 10 iterations. The correct square support constraint was then reinstated. Figure 3(D) shows the output image after 10 more iterations of the error-reduction algorithm ($E_0 = 0.060$); Fig. 3(E) shows the output after an additional 60 iterations of the hybrid input–output algorithm plus five iterations of error reduction ($E_0 = 0.027$); and Fig. 3(F) shows the output after an additional three cycles of 40 hybrid input–output plus five error-reduction iterations each ($E_0 = 0.018$).

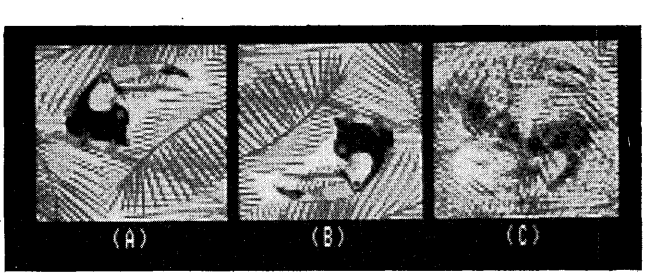

Fig. 2. Simultaneous twin-images problem. (A) Object $f(x)$. (B) Twin image $f^*(-x)$. (C) Output image from the iterative transform algorithm that has stagnated with features of both.

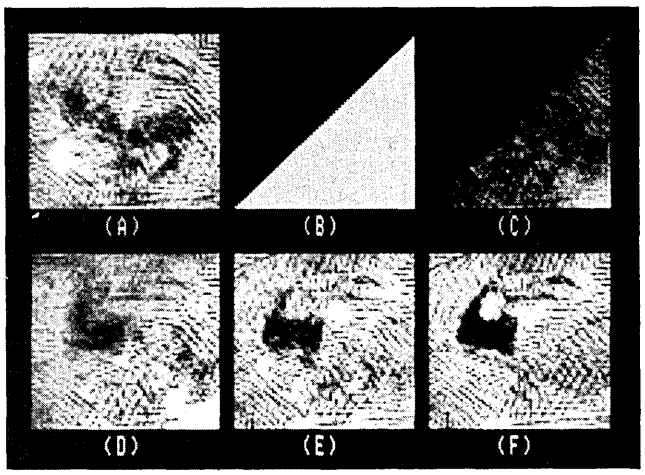

Fig. 3. Reduced-area support constraint method for overcoming the problem of simultaneous twin images. (A) Stagnated output image. (B) Mask defining the temporary reduced-area support constraint. (C) Output image after 10 iterations using temporary support. (D)–(F) Output image after further iterations using the correct support.

The reduced-area support constraint tends to favor some of the features of either $f(x)$ or $f^*(-x)$ over the other. Since it employs an incorrect support constraint, it cannot converge to a solution. However, when the correct support constraint is reinstated, one of the two twin images may have a sufficiently large advantage over the other that the algorithm can then converge toward that image.

In the small number of trials in which it was tested, the reduced-area support constraint method worked in the majority of the cases tried, but it is by no means guaranteed to work. If an application of the method does not relieve the problem of stagnation with both twin images present, then one might try another application of the method, using a different reduced-area temporary support constraint mask. The method as it stands has not yet been optimized as to the form of the temporary mask or the number or type of iterations that should be performed with the temporary mask. Nevertheless, the method has been shown to be promising as a solution to the problem of stagnation with features of both twin images present.

For the example shown, knowledge that the simultaneous twin-image mode of stagnation was present was obtained by visual inspection of the output image. The decision could also be automated by measuring the degree of symmetry of the image. An example of such a measure would be the ratio of the peak of the cross correlation of $g'(x)$ with $g'^*(-x)$ to the peak of the autocorrelation of $g'(x)$.

4. METHODS OF OVERCOMING STRIPES

Several methods of overcoming the problem of stagnation associated with stripes across the image were attempted before successful methods were developed. Before we describe the successful ones we will mention two of the unsuccessful methods because they illustrate features of the problem.

A. Some Features of Stripes

The error metric, E_0 or E_F, can be considered as a function of an N^2-dimensional parameter space spanned by the values of $g(x)$ or of $\phi(u)$. Stagnation with stripes can be thought of as being stuck at a local minimum of the error metric. This local minimum is not very far from the global [$E_0 = E_F = 0$ for $g(x) = f(x)$] minimum, since the output image usually closely resembles the original object except for the presence of the stripes. It was thought that if the input image $g(x)$ were sufficiently perturbed, then the estimate $g'(x)$ would be moved out the that local minimum and perhaps fall into the sought-after global minimum. Experimental tests were made in which increasing amounts of random noise were added to $g(x)$, and from these starting points more iterations were performed. It was found that even with large amounts of added noise, in very few iterations the output image reverted back to the same point of stagnation having the same stripes as before. These experiments are an indication that the local minima characterized by stripes are very strong local minima.

A second unsuccessful method utilized the fact that since $g(x)$ is real valued, and so $\phi(-u) = -\phi(u)$, the sinusoidal stripes must come from a conjugate pair of localized areas in the Fourier domain. In addition, because the iterative algorithm forces the output image to have the correct Fourier modulus at the sampled points, the error must also be a pure-phase error at the sampled points (see Subsection 4.D for a discussion of the values between the samples). The spatial frequency of the stripes was measured to determine what area of the Fourier domain was in error. Constant phase terms were added to the Fourier transforms of the striped images in these areas in a conjugate-symmetric way, but the stripes remained in the image despite the use of a variety of constant phases and a variety of sizes of such areas. This was true despite the fact that, when similar constant phase errors were added to the Fourier transform of the object itself, the corresponding image had stripes that looked very much like the type of stripe that was produced by the mode of stagnation of the iterative algorithm; these stripes went away after only one or two iterations, whereas the stripes produced by the stagnating iterative algorithm were stable. This experience pointed out the spatial complexity of the phase error that caused the stripes.

B. Voting Method

The key to solving the problem with the stripes is the fact that if the iterative algorithm is applied multiple times, each time with a different random starting guess, then the stripes of the various reconstructions will usually have different orientations and frequencies. This behavior was noted in Fig. 9 of Ref. 17. This implies that the errors will occur in different areas of the Fourier domain. Because the sinusoidal patterns usually become well defined, the areas of the phase errors are localized reasonably well. These features of the phase errors suggested a voting method. The idea is that if two of three Fourier phases are similar but a third is different, then the dissimilar phase is usually incorrect and should be discarded.

The voting method consists of the following steps:

(1) Generate three output images with different stripes by running the iterative transform algorithm three times, each with different random numbers for the initial input (if one of the output images is without stripes, then the problem is, of course, solved).

(2) Cross correlate the second and third images and their

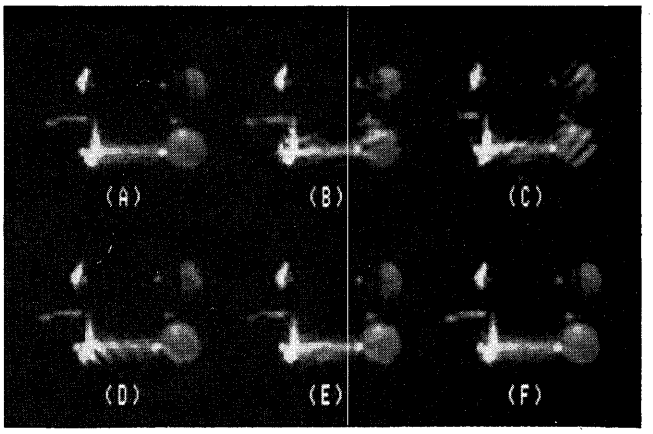

Fig. 4. Voting method for eliminating stripes in the output image. (A) The object. (B)–(D) Output images from the iterative transform algorithm, each with different stripes. (E) Output of voting method. (F) Output image after further iterations.

twins (the images rotated 180°) with the first image to determine their relative translations (to within a small fraction of a pixel, which can be accomplished by oversampling the cross-correlation peak) and orientations.

(3) Fourier transform all three images.

(4) Subtract appropriate linear phase terms from the phases of the Fourier transforms of the second and third images and conjugate if the orientation is opposite, to give the translations and orientations in the image domain that would cause the three images to be in registration. This removes unwanted relative linear phase terms in the Fourier transforms (which could have been accomplished by translating the second and third images before Fourier transformation, but fractional-pixel translation is more easily accomplished in the Fourier domain).

(5) At each point u in the Fourier domain, compute the modulus of the difference between each pair of complex transforms. Average the two complex numbers that are the closest (discarding the third) and replace the complex value at that point with this average. Optionally: replace the modulus of the average with the measured modulus. Alternatively, one can take the phase midway between the two closest phases of the three, if proper attention is paid to modulo-2π questions.

(6) Inverse Fourier transform to yield the corresponding image.

The output of the voting method is used as the input to further iterations of the iterative transform algorithm. If the method fails because two of the Fourier transforms have errors in the same location, then it should be repeated using different random numbers for the initial inputs to the iterative transform algorithm.

An example of the use of the voting method is shown in Fig. 4. Figure 4(A) shows a diffraction-limited image of a satellite that is our object. It was formed from a digitized picture of a satellite within a 64×64 array embedded in a 128×128 array. The digitized picture was low-pass filtered using the incoherent transfer function of a circular aperture of diameter 62 pixels (the Fourier transform of the object was multiplied by the autocorrelation of the circular aperture) to produce the object (a diffraction-limited image) shown in Fig. 4(A). The sidelobes of the impulse response due to the circular aperture cause the diffraction-limited image to have a small amount of energy well outside the support of the object, making the error metric $E_0 = 0.0026$ for this object. Because of this inherent slight inconsistency between the Fourier modulus data (which corresponds to the diffraction-limited image) and the support constraint, E_0 can never be driven to zero. Figures 4(B), 4(C), and 4(D) show three output images from the iterative transform algorithm, each generated by using different random numbers for the starting input. The support constraint used was the 64×64 square support. Stripes of different spatial frequencies are clearly seen in each of the images. E_0 for the three output images is 0.0155, 0.0316, and 0.0038, respectively. For the output image shown in Fig. 4(D), detection of the existence of the stripes is more difficult because of the low value of E_0, but inspection of an overexposed version of it clearly reveals stripes in the area outside the support of the object. Figure 4(E) shows the output of the voting method, for which $E_0 = 0.0680$, and Fig. 4(F) shows the result of further iterations of the iterative transform algorithm, for which $E_0 = 0.0035$ and the stripes are successfully removed.

An advantage of the voting method is that one need not understand the nature of the phase error except that it is localized in different areas of the Fourier domain for different output images. The voting method may therefore be useful for other types of phase errors in addition to those characterized by stripes in the image.

C. Patching Method

The patching method, like the voting method, utilizes the fact that output images coming from different starting inputs usually have phase errors localized in different areas of the Fourier domain. The patching method uses an additional piece of information: Since the stripes extend beyond the known support of the object (although they are dimmer there), they can be isolated and analyzed to determine approximately what area in the Fourier domain contains the localized phase errors. With this information one can patch together a Fourier transform having fewer errors from two Fourier transforms that have these localized phase errors.

The patching method consists of the following steps:

(1)–(4) Perform Steps (1)–(4) of the voting algorithm but use only two output images rather than three (if one of the output images is without stripes, then the problem is, of course, solved).

(5) For each of the two images, zero out the image in its support region. This isolates the stripes. Use a smooth apodization to avoid sidelobe problems in the Fourier domain.

(6) Fourier transform the isolated stripes from each image.

(7) Smooth and threshold the Fourier modulus (after zeroing out a region about the origin to eliminate an undesired dc component) to generate a Fourier mask for each of the two images. These masks define the areas in the Fourier domain that have the phase errors.

(8) If the two masks overlap, repeat Step (7) using a larger threshold value or a smaller smoothing kernel, or redo Steps (1)–(7) using another random input to start the iterative transform algorithm.

(9) Form a new Fourier transform having the phase of the Fourier transform of the first image except within its

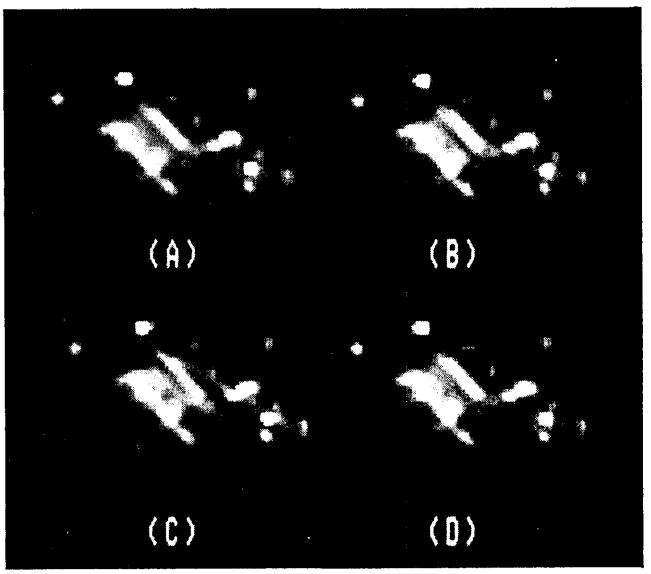

Fig. 5. Patching method for eliminating stripes in the output image. (A) The object. (B), (C) Output images from the iterative transform algorithm, each with different stripes. (D) Output of patching method.

Fourier mask, where the phase of the Fourier transform of the second image is substituted.

(10) Inverse Fourier transform to yield the corresponding image.

The output of the patching method is used as the input to further iterations of the iterative transform algorithm.

An example of the use of the patching method is shown in Figs. 5–7. Figure 5(A) shows the object, a diffraction-limited image of a satellite (E_0 = 0.0024), and Figs. 5(B) (E_0 = 0.0268) and 5(C) (E_0 = 0.0503) show two output images from the iterative transform algorithm, each generated by using different random numbers for the starting input. Upon close inspection, stripes of different spatial frequencies can be seen in each of the output images. Figure 6 shows the same thing as Fig. 5, only heavily overexposed so that the stripes over the object and beyond the support of the object can be seen more readily. Figure 7(A) shows the apodized mask used in the image domain that, when multiplied with the striped image, isolates the stripes. The resulting isolated stripes are shown in Fig. 7(B). (A bias was added in the display of this result, making the most negative value black, zero value gray, and the largest value white.) Figure 7(C) shows the modulus of the Fourier transform of the isolated stripes, and Fig. 7(D) shows the Fourier mask obtained by thresholding that Fourier modulus at a value 0.9 times the peak and smoothing with a 16 × 16 kernel. Figures 7(E)–7(G) show the same things as Figs. 7(B)–7(D) but for the second striped image. The output of the patching method is shown in Figs. 5(D) and 6(D)—the stripes in the two images were eliminated. Its error metric is E_0 = 0.00576, which is much lower than that for the striped images.

The voting and patching methods are both completely automated once it is decided that the iterative transform algorithm is stagnating on an image that has stripes. For the examples shown, knowledge that the striped-image mode of stagnation is present was obtained by visual inspection of the output image, from which it is quite obvious. This decision could also be automated, for example, by performing for a given single output image Steps (5) and (6) of the patching method and detecting the presence of especially bright areas in the Fourier domain.

D. Zero Reversal of the Fourier Transform

Comparison of the Fourier phase of the striped image with that of the original object yielded interesting insights into the properties of Fourier transforms.

Figure 8(A) shows the phase of the Fourier transform of the original object [shown in Fig. 4(A)], and Fig. 8(B) shows the upsampled phase of the area in Fig. 8(A) outlined by the white square. Figures 8(C) and 8(D) show the same thing for the striped image [shown in Fig. 4(B)]. To reduce the linear phase component, the centroid of the object was translated to the origin before Fourier transformation and the striped image was translated to be in register with the object. The large circular pattern in Fig. 8(A) is due to the simulation of the effects of diffraction by the circular aperture mentioned earlier. Outside the circle the Fourier transform has small nonzero values due to round-off error in the computer. Note that to upsample the phase, one must compute the phase of an upsampled complex Fourier transform that can in turn be computed by Fourier transformation of the object (or image) embedded in a larger array padded with zeros.

Of particular interest is the phase within the four small squares drawn on Figs. 8(B) and 8(D). In Fig. 8(B), the phase within the upper-right square "wraps around" one point, u_0, in the Fourier domain. That is, if one starts at a point u near u_0, as one progresses full circle around u_0 the phase slips by 2π rad. It is easily shown that this branch cut in the phase indicates that the Fourier modulus goes through a zero at u_0.[22] Second-order zeros [where $F(u_0) = 0$ has zero first partial derivatives as well] might not exhibit phase wraparound, but they are rare compared with first-order zeros. The existence of zeros in $F(u)$ implies an inherent $2\pi n$ (where n is an integer) ambiguity in the phase. Self-consistent phase unwrapping cannot logically be performed

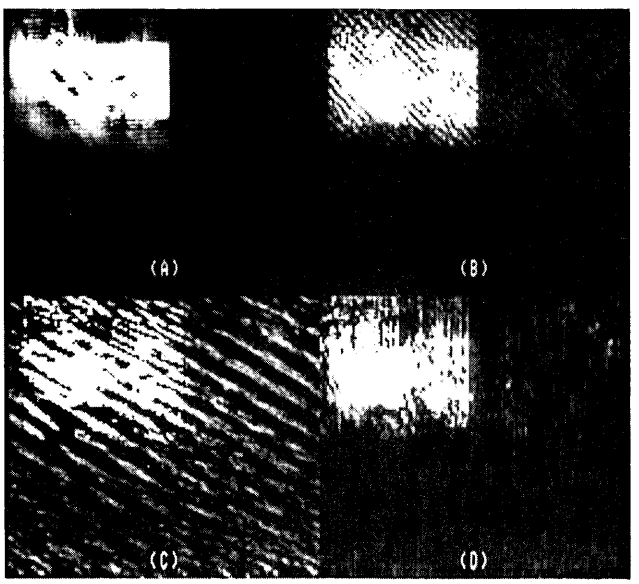

Fig. 6. Same as Fig. 5 but overexposed to emphasize the stripes, and full 128 × 128 arrays are shown.

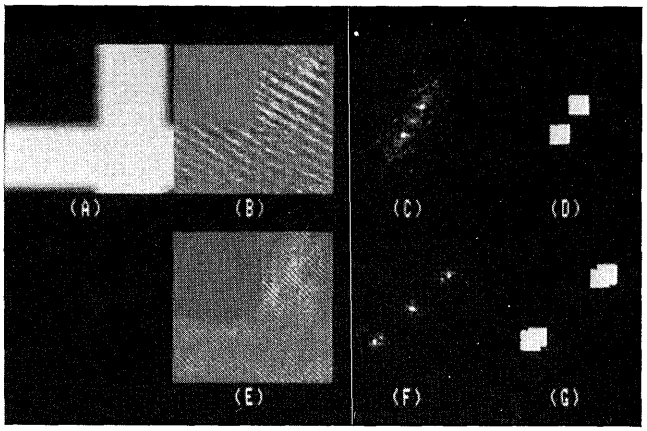

Fig. 7. Details of the patching method. (A) Mask used to isolate the stripes of the output images. (B) Stripes isolated from the first image. (C) Modulus of the Fourier transform of the isolated stripes from the first image. (D) Fourier mask obtained by thresholding and smoothing (C). (E)–(G) Same as (B)–(D) but for the second image.

in such cases. This is the usual case for Fourier transforms of images. From the presence or absence of phase wraparound, it is evident that the Fourier transform goes through first-order zeros both within the upper-right and lower-left squares but not within the two squares in the middle of Fig. 8(B). On the other hand, for the case of the striped image the presence or absence of first-order zeros is just the opposite, as can be seen in Fig. 8(D). Where there are two first-order zeros for the original object there are not any for the striped image, and vice versa. That is, the zeros are "reversed." (This zero reversal should not be confused with the flipping of complex zeros that is encountered in the analysis of uniqueness.) Also, by inspecting upsampled versions of the Fourier transforms we found that the first-order zeros did not become higher-order zeros (which could cause the disappearance of the phase wraparound) but truly became nonzero. The difference between having and not having first-order zeros is extremely important: with a first-order zero the phase wraps around and varies very rapidly, whereas otherwise the phase is relatively smooth. Note that the transitions from white (π phase) to dark ($-\pi$ phase) in Fig. 8 are not jumps in phase *per se*; they are just an artifact of our ability to compute and display phase only modulo 2π.

If one draws a quadrilateral having vertices at the four points at which the zeros are reversed, one finds that the Fourier phase of the striped image differs from that of the object only (approximately) within the quadrilateral. Note that the phases outside the quadrilateral are practically the same in Fig. 8(D) as in Fig. 8(B). That is, the Fourier phase error for the striped image is localized in the area between the reversed zeros.

It is not accidental that the reversed zeros come in pairs. In order for the phase to be consistent in the surrounding area, a continuous path around the entire area of the localized phase error for the striped image must contain the same number of first-order zeros as for the Fourier transform of the object.

Initially it seems contradictory that the zeros of the Fourier transform of the striped output image could be different from those of the object's Fourier transform since the stripped image and the object have exactly the same Fourier modulus at the sampled points. However, this possibility arises because we are dealing with sampled data in the computer. The zeros only rarely fall on the sampling lattice: they usually fall some distance between the samples. In the presence of even the slightest error, including round-off error due to the finite word length used by the computer, it becomes difficult to see, even from a heavily oversampled Fourier modulus, whether it goes through zero or merely comes close to it. More important, though, is that since the striped image has energy throughout image space, rather than being confined to the support of the object, its Fourier modulus is aliased and differs from the Fourier modulus of the object for points off the sampling lattice. Hence its Fourier transform can truly have zeros where the object's Fourier transform does not, and vice versa, despite their having the same Fourier modulus at the sampled points.

E. Lines of Real and Imaginary Zeros

Figure 9 shows the lines where the real and imaginary parts of the Fourier transform of the object (in this case translated to be in one quadrant of the array) are zero, for the same area of the Fourier domain shown in Fig. 8(B). We will refer to these lines as the lines of real zeros and lines of imaginary zeros. (Note that we are referring here to the zeros in the two-dimensional real plane, not to the zeros in the complex plane, which are frequently discussed in regard to the uniqueness of phase retrieval.) The lines of real zeros were computed by scanning across each line and each column of an oversampled version of the real part of the Fourier transform and noting where it changed sign. The lines of imaginary zeros were found in a similar manner. In Fig. 9 the real zeros are denoted by dark lines and the imaginary zeros by light lines on a gray background. The complex Fourier transform goes through a zero wherever both the real part

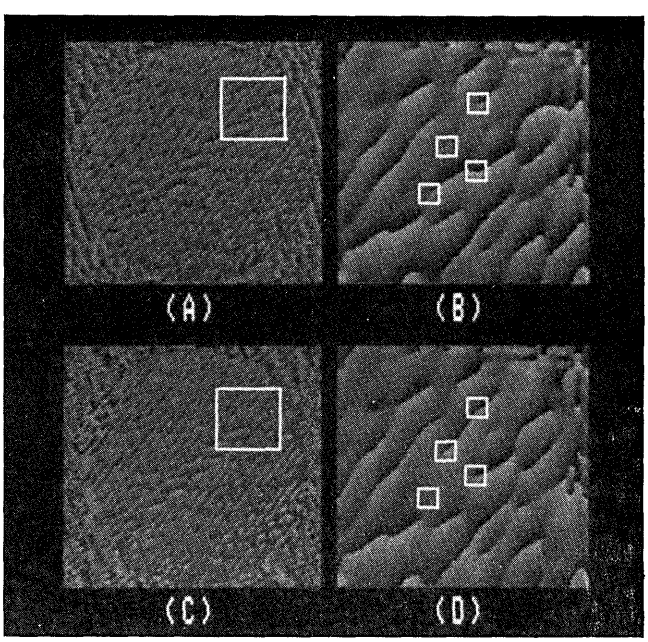

Fig. 8. Fourier phases. (A) Fourier phase of the object. (B) Upsamples phase from the area in (A) outlined by the square. (C) Fourier phase of the striped output image. (D) Upsampled phase from the area in (C) outlined by the square. The (u, v) zeros of the complex Fourier transforms are reversed in the areas enclosed in squares in (B) and (D).

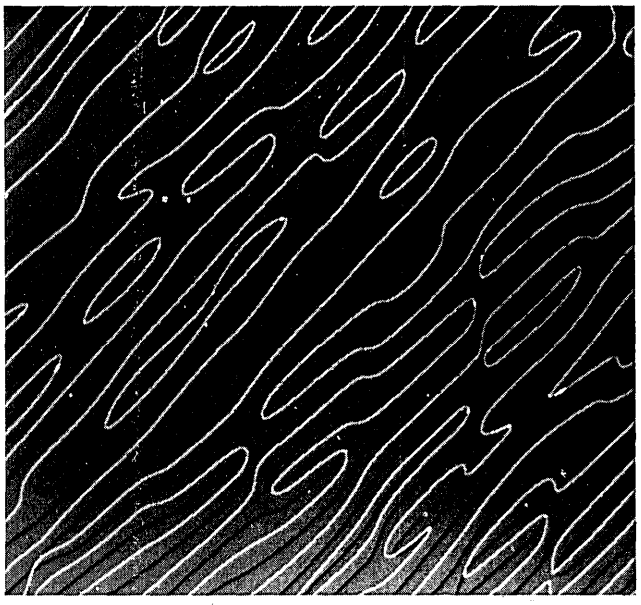

Fig. 9. Locations of the zeros of the real part (dark lines) and the imaginary part (white lines) of the Fourier transform of the object. The object was translated to be causal, and the area of its Fourier transform shown here is that shown in Fig. 8(B).

and the imaginary part are zero, that is, where the dark lines and light lines intersect. The entire discussion regarding the zeros of the Fourier transform of the striped image versus those of the object can be explained in terms of Fig. 9 and a similar picture for the striped image case as well as by using Fig. 8.

Except for special cases, in Fig. 9 the lines of real zeros and imaginary zeros cross at single points rather than being tangent to one another over extended intervals; hence the zeros tend to occur at discrete points.

In addition, notice that the lines of imaginary zeros have a strong tendency to be halfway between two lines of real zeros, and vice versa. This can be understood as follows. Halfway between two neighboring lines of real zeros (think of them as a single-level topographic map of the real part), one would expect to find a line of local maxima or minima, like ridges or gullies, respectively. These ridges (or gullies) have the property that they are local maxima (minima) along all directions except along the length of the ridge (gully). Because the object $f(x, y)$ was translated to the upper-left quadrant of the plane [i.e., $f(x, y) = 0$ for $x < 0$ and for $y < 0$, making it causal], then $F(u, v)$ satisfies the Hilbert transform relationships (see Appendix A)

$$F_I(u, v) = \frac{1}{\pi} P \int \frac{F_R(u, v')}{v' - v} \, dv' \tag{12}$$

$$= \frac{1}{\pi} P \int \frac{F_R(u', v)}{u' - u} \, du', \tag{13}$$

where $F = F_R + iF_I$, F_R, and F_I are real valued and where P denotes the Cauchy principal value. For clarity in this discussion and in Appendix A, we use (x, y) as the two-dimensional coordinates in object space and (u, v) in Fourier space rather than the vector notation used elsewhere in this paper. Because $1/(v' - v)$ is much larger near $v' = v$ than elsewhere, one would expect the integrand near $v' = v$ to dominate the integral of Eq. (12). If the point (u_0, v_0) is at a ridge (or a gully) of $F_R(u, v)$, then one would expect $F_R(u_0, v')$ to have a local maximum (minimum) at $v' = v_0$ and be closely approximated by a quadratic in a small region of v' centered about v_0, since $F_R(u_0, v')$ is a band-limited function of v' (it is the Fourier transform of a function of finite extent). Therefore, since the numerator $F_R(u_0, v')$ is even about $v' = v_0$ and the denominator $(v' - v_0)$ is odd about $v' = v_0$, the integrand is odd about $v' = v_0$ and the contribution to the integral from the neighborhood about v_0 is near zero. Since that neighborhood is the part of the integral that usually dominates, it is easily seen why $F_I(u, v)$ tends to be zero near the ridges and gullies of $F_R(u, v)$. The same can be seen from Eq. (13). The same argument can be used to show why $F_R(u, v)$ tends to be zero near the ridges and gullies of $F_I(u, v)$.

5. METHOD FOR OVERCOMING TRANSLATED SUPPORT

Because $f(x - x_0)$ has the same Fourier modulus as $f(x)$, the location of the object's support is arbitrary. Frequently the image partially reconstructed by the algorithm will not be in perfect registration with the support constraint. Then enforcing the support constraint causes an inadvertent truncation of part of the desired image, causing the algorithm to stagnate. In addition to the enlarging support method described in Section 6, a method of combating this stagnation problem is to translate dynamically either the support constraint or the image.

The amount of translation to be used can be determined as follows. Compute the total energy of the output image, $g_k'(x)$, (i.e., square and sum) over the area of support constraint for the current position of the support constraint and for the support constraint translated by one or two pixels in every direction. The support constraint should be translated to the position for which the energy is maximized. This can be done occasionally or at every iteration. Alternatively, compute the cross correlation of the support mask with $g_k'(x)$ or with $|g_k'(x)|^2$ and translate according to the peak of the cross correlation. This method would be particularly effective if, just before it were performed, a support constraint larger than the usual support were used for a few iterations; this would give the truncated part of the image a chance to establish itself.

6. ITERATIVE TRANSFORM ALGORITHM DETAILS

Some researchers have had varying success in applying the iterative transform algorithm to phase retrieval from a single intensity measurement. In this section, a number of additional aspects of making the iterative algorithm work are given as an aid to the practical implementation of the algorithm.

Recall from Section 2 that the heart of the algorithm consists of several cycles of iterations, where one cycle consists of K_1 iterations of the hybrid input–output algorithm [Eqs. (4)–(6) and (11)] followed by K_2 iterations of the error-reduction algorithm [Eqs. (4)–(7)]. Our experience has shown that values of K_1 from 20 to 100, of K_2 from 5 to 10, and of the feedback parameter β from 0.5 to 1.0 (use, say, 0.7) work well.

The DFT's are computed by using the FFT algorithm. The sampling in the Fourier domain should be fine enough to ensure that the object domain array size be at least twice the width and height of the object itself, which is equivalent to achieving the Nyquist sampling rate for $|F(u)|^2$.

A straightforward method to evaluate Eq. (5) is to compute the phase from the real and imaginary parts of $G_k(u)$, then combine it with $|F(u)|$ to form $G_k'(u)$, and finally compute the real and imaginary parts of $G_k'(u)$ (which are required by the FFT) from its modulus and phase. Alternatively, one can employ

$$G_k'(u) = G_k(u)|F(u)|/[|G_k(u)| + \delta], \tag{14}$$

where δ is a very small number used to prevent overflow problems in the rare event that $G_k(u) = 0$. (For some computers one can use $\delta = 0$ with no ill effects.)

The datum that one must have available is an estimate, $|\hat{F}(u)|$, of the modulus, $|F(u)|$, of the Fourier transform of the object. Although the iterative transform reconstruction algorithm is not hypersensitive to noise, care must be taken to obtain the best possible estimate of the Fourier modulus, which may involve considerable compensation of the raw data,[23] depending on how it is collected. In many circumstances one can estimate the expected value of the nrms errors of the data:

$$E_{|\hat{F}|} = \left[\frac{\sum_u [|\hat{F}(u)| - |F(u)|]^2}{\sum_u |F(u)|^2}\right]^{1/2}. \tag{15}$$

This is useful for deciding when one is close enough to a solution.

For the astronomy problem one has a nonnegativity constraint in the object domain. Furthermore, one can compute upper bounds on the support of the object in any of several ways.[20] The simplest way is to use a rectangle that is half the size, in each dimension, of the smallest rectangle that encloses the autocorrelation, which is given by the inverse Fourier transform of $|F(u)|^2$. If the actual support of the object is known *a priori*, then that should of course be used. Any other types of *a priori* information should be used during the iterations if available. The support constraint can in general be defined by a mask that is unity within the support and zero outside. If the support constraint and the set of points are defined as binary mask arrays, then the computations of Eqs. (7), (10), and (11) can be performed arithmetically without the use of logic, which is advantageous when using array processors.

There are many ways to pick an initial input to the algorithm. Although claims have been made that a certain crude estimation of the phase offers a superior starting point,[24] others have found that random numbers do as well or better.[25] Having an initial input close to the true solution reduces the number of iterations required and might help to avoid some of the stagnation problems. If another reconstruction method (Knox–Thompson[26] for astronomical speckle interferometry, for example) has yielded an image, then that image would be the appropriate starting input. One can either view the other reconstruction method as a means of supplying starting inputs for the iterative transform algorithm or view the iterative transform algorithm as a means of "cleaning up" images reconstructed by the other method. If no other initial estimate for the object is available, then one should use random numbers in the object domain or for the Fourier phase, giving an unbiased start to the algorithm. In the object domain, a convenient starting guess, $g_0(x)$, can be formed by filling the support mask with random numbers. Another method[3] is to threshold the autocorrelation (at, say, 0.005 its maximum value), demagnify that by a factor of 2 in each dimension by discarding every other row and every other column, and finally fill the resulting shape with random numbers. (Note that this shape does not necessarily contain the support of the object.[20])

The algorithm can be made to converge faster and avoid a stagnation problem (see Section 5) if the support mask is chosen to be somewhat smaller than the correct support for the first cycle or two of iterations. Since it is the incorrect support, the smaller support mask is inconsistent with Fourier modulus, and stagnation will eventually occur when it is used. Nevertheless, the smaller support mask helps to force most the energy of the output, $g'(x)$, into a confined region in fewer iterations. After this has happened the support mask should be enlarged to the correct support constraint for the object. This enlargement of the mask could be done in more than one step if desired. When the algorithm has nearly finished reconstructing the object, it is often beneficial to make the support mask even larger than the correct support for the object. This helps to ensure that no parts of the object are being inadvertently truncated by the support constraint. The progression from a smaller support mask to a large one also helps to avoid having edges of the output image biased toward falling right at the edges of the support mask. Use of the method described in Section 5 does this as well and is recommended for use whenever truncation is suspected (or, to be safe, at the end of each cycle or even after every iteration).

As the iterations progress, the nrms error in the object domain,

$$E_0 = \left[\frac{\sum_{x\in\gamma} |g_k'(x)|^2}{\sum_x |g_k'(x)|^2}\right]^{1/2} \tag{16}$$

[compare with Eq. (10)], should be computed. The nrms error is a measure of how close the current Fourier transform pair is to a solution. Note that the denominator of the above error metric is a constant that need be computed only once. Note also that at the end of a cycle $E_0 \simeq E_F$, the nrms error in the Fourier domain.[3] When E_0 goes significantly below $E_{|\hat{F}|}$ of Eq. (15), one has a solution consistent with the measured data and constraints to within the limits of the error in the given data. It is unlikely that E_0 will ever go to zero because noise in $|\hat{F}(u)|$ almost always results in a Fourier modulus that is inconsistent with either the nonnegativity constraint or any reasonable support constraint or both. This can be seen from the fact that an autocorrelation computed from a noisy $|\hat{F}(u)|$ will ordinarily have negative values at some points (which could only arise from an object having negative values) and will ordinarily have (possibly small) nonzero values far beyond the extent of the true autocorrelation of

the object. This problem can be alleviated by setting equal to zero the values of the autocorrelation that are negative or lie beyond some assumed autocorrelation support; but even then noise remains and there will be no nonnegative image that is completely consistent with the Fourier modulus estimate. On the other hand, in the presence of noise there will ordinarily exist an output image $g'(x)$ that is in better agreement with the noisy data than the true image is. Consequently, for the case of noisy data, a "solution" is not found until E_0 is decreased to a level somewhat less than $E_{|F|}$. An exception to this is for the very-low-noise case in which the dominant error is sidelobe energy that spills outside the object's support due to diffraction effects, as for the examples shown in Section 4.

If all goes well, the iterative transform algorithm will converge to a solution after a small number of cycles of iterations. If there are multiple solutions, the iterative transform algorithm is capable of finding any one of them, depending on the starting input.[27,28] Confidence that the solution is the one and only true solution can be increased by performing two or more trials of the algorithm, each time using different random numbers for the initial input.

In some cases the iterative transform algorithm will stagnate before reaching a solution. The algorithm can be considered to have stagnated if the error E_0 has failed to decrease after three additional cycles. While some objects can be reconstructed very easily, requiring only one or two cycles, other more difficult objects can require many cycles comprising well over a thousand iterations. Consequently, one should not jump too readily to the conclusion that the algorithm has stagnated. It often occurs that very slow progress is made for many iterations, but then the algorithm suddenly finds its way and rapid progress is made in just a few iterations.

If the iterative transform algorithm does stagnate, then one can start over with a different set of random numbers for the initial input; alternatively, several methods for getting out of the stagnated condition are possible. Sometimes changing the support constraint (enlarging it or translating it as described in Section 5) is what is required. The need for doing this can be established by following the steps suggested at the end of Section 5. Altering the feedback parameter, β, sometimes helps. Temporarily using a larger value for β, say, 1.2, causes larger changes to be made and may move the output away from the condition of stagnation; however, this should not be carried on for too many iterations because it causes the algorithm to become unstable. If the support mask is centrosymmetric or nearly so, then the simultaneous twin images can be present. This condition can be detected visually (comparing the output image with a second version of it rotated 180° helps) or by the method suggested at the end of Section 3. If this condition is suspected, then use the method of overcoming the problem of simultaneous twin images described in Section 3. Even if the twin-image problem is not present, the method might move the output image out of the condition of stagnation. The mode of stagnation characterized by stripes is easily detected by looking for stripes outside the support region in a picture of $g_k'(x)$ that is heavily overexposed; alternatively one could use the method described at the end of Subsection 4.C. If stripes are present, the iterations should be continued until stagnation occurs, because (1) the stripes may go away naturally and (2) further iterations cause the stripes to become more nearly sinusoidal, which is equivalent to the phase errors' being confined to a smaller, more distinct area of the Fourier domain, which makes them easier to overcome by the methods described in Section 4.

Other tricks may be helpful or necessary for certain special cases. For example, if the object consists of some interesting details superimposed on a diffuse background, then the defogging method can make the reconstruction of the object easier.[24]

7. SUMMARY AND CONCLUSIONS

In many cases of interest, the problem of phase retrieval from a single intensity measurement can be solved by a straightforward application of a few cycles of the iterative transform algorithm. For some cases, the algorithm stagnates before reaching a solution that is consistent with the data and constraints. Three different modes or conditions of stagnation have been identified: simultaneous twin images, stripes superimposed upon the image, and unintentional truncation by the support constraint. Methods for overcoming each of these modes of stagnation have been devised and have been demonstrated to be effective for particular examples. The use of these methods in conjunction with the iterative transform algorithm greatly enlarges the class of objects that can be reconstructed successfully. This has also helped to provide further empirical evidence of the uniqueness of the solution for two-dimensional objects. Some previous doubts of uniqueness, tied to an inability of the algorithm to converge in some instances, have been removed.[28] In particular, we have definitively shown that the striped images represent a local minimum rather than a true ambiguity.

In the course of investigating the stripes phenomenon, insight was gained into some of the properties of the Fourier transform of images. The Fourier transform of a striped image has a phase that differs from that of the Fourier transform of the object in a fairly well-defined region that is determined by the locations of zeros of the Fourier transform that are reversed, i.e., where first-order zeros appear or disappear. First-order zeros are common in the Fourier transforms of images. Attempts at phase unwrapping, as required by the Knox–Thompson method, utilizing multiple paths of integration will fail unless proper attention is paid to the branch cuts associated with first-order zeros. If an image is causal, then the lines of real and imaginary zeros of its Fourier transform follow along the ridges and gullies of the imaginary part and the real part, respectively, which can be understood from the Hilbert transform relationship.

APPENDIX A: HILBERT TRANSFORMS FOR TWO DIMENSIONS

Let $f(x, y)$ be zero for $x < 0$ and for $y < 0$ (let all integrations be understood to be from $-\infty$ to $+\infty$).

$$F(u, v) = F_R(u, v) + iF_I(u, v) \tag{A1}$$

$$= \iint f(x, y)\exp[-i2\pi(ux + vy)]\mathrm{d}x\mathrm{d}y$$

$$= \int \left[\int f(x, y) \exp(-i2\pi ux) \mathrm{d}x \right] \exp(-i2\pi vy) \mathrm{d}y$$

$$= \int \tilde{f}(u, y) \exp(-i2\pi vy) \mathrm{d}y, \tag{A2}$$

where $\tilde{f}(u, y)$ is zero for $y < 0$. Fixing u to be a constant for the moment, $F(u, v)$ is the one-dimensional Fourier transform of $\tilde{f}(u, y)$, which is zero for $y < 0$ (i.e., it is causal in y), in which case we have the Hilbert transform relationships[29]

$$F_I(u, v) = \frac{1}{\pi} P \int \frac{F_R(u, v')}{v' - v} \mathrm{d}v' \tag{A3}$$

and

$$F_R(u, v) = -\frac{1}{\pi} P \int \frac{F_I(u, v')}{v' - v} \mathrm{d}v', \tag{A4}$$

where P denotes the Cauchy principal value. This is true for all values of (u, v). By a similar argument we have

$$F_I(u, v) = \frac{1}{\pi} P \int \frac{F_R(u', v)}{u' - u} \mathrm{d}u' \tag{A5}$$

and

$$F_R(u, v) = -\frac{1}{\pi} P \int \frac{F_I(u', v)}{u' - u} \mathrm{d}u'. \tag{A6}$$

ACKNOWLEDGMENTS

This research was supported by the U.S. Air Force Office of Scientific Research under contract F49620-82-K-0018.

Portions of this paper were presented at the 1984 Annual Meeting of the Optical Society of America in San Diego, California, October 29–November 2.[30]

REFERENCES

1. J. R. Fienup, "Reconstruction of an object from the modulus of its Fourier transform," Opt. Lett. **3,** 27–29 (1978).
2. J. R. Fienup, "Space object imaging through the turbulent atmosphere," Opt. Eng. **18,** 529–534 (1979).
3. J. R. Fienup, "Phase retrieval algorithms: a comparison," Appl. Opt. **21,** 2758–2769 (1982).
4. G. H. Stout and L. H. Jenson, *X-Ray Structure Determination* (Macmillan, London, 1968).
5. C. Y. C. Liu and A. W. Lohmann, "High resolution image formation through the turbulent atmosphere," Opt. Commun. **8,** 372–377 (1973).
6. P. J. Napier and R. H. T. Bates, "Inferring phase information from modulus information in two-dimensional aperture synthesis," Astron. Astrophys. Suppl. **15,** 427–430 (1974).
7. B. R. Frieden and D. G. Currie, "On unfolding the autocorrelation function," J. Opt. Soc. Am. **66,** 1111 (A) (1976).
8. J. E. Baldwin and P. J. Warner, "Phaseless aperture synthesis," Mon. Not. R. Astron. Soc. **182,** 411–422 (1978).
9. R. H. T. Bates, "Fourier phase problems are uniquely solvable in more than one dimension. I: underlying theory," Optik **61,** 247–262 (1982); K. L. Garden and R. H. T. Bates, "II: One-dimensional considerations," Optik **62,** 131–142 (1982); W. R. Fright and R. H. T. Bates, "III: Computational examples for two dimension," Optik **62,** 219–230 (1982).
10. H. H. Arsenault and K. Chalasinska-Macukow, "The solution to the phase retrieval problem using the sampling theorem," Opt. Commun. **47,** 380–386 (1983); K. Chalasinska-Macukow and H. H. Arsenault, "Fast iterative solution to exact equations for the two-dimensional phase-retrieval problem," J. Opt. Soc. Am. A **2,** 46–50 (1985).
11. A. Levi and H. Stark, "Image restoration by the method of generalized projections with application to restoration from magnitude," J. Opt. Soc. Am. A **1,** 932–943 (1984).
12. H. V. Deighton, M. S. Scivier, and M. A. Fiddy, "Solution of the two-dimensional phase retrieval problem," Opt. Lett. **10,** 250–251 (1985).
13. R. W. Gerchberg and W. O. Saxton, "A practical algorithm for the determination of phase from image and diffraction plane pictures," Optik **35,** 237–246 (1972).
14. R. W. Gerchberg, "Super-resolution through error energy reduction," Opt. Acta **21,** 709–720 (1974).
15. W. O. Saxton, *Computer Techniques for Image Processing in Electron Microscopy* (Academic, New York, 1978).
16. M. A. Fiddy, B. J. Brames, and J. C. Dainty, "Enforcing irreducibility for phase retrieval in two dimensions," Opt. Lett. **8,** 96–98 (1983).
17. G. B. Feldkamp and J. R. Fienup, "Noise properties of images reconstructed from Fourier modulus," in *1980 International Optical Computing Conference*, W. T. Rhodes, ed., Proc. Soc. Photo-Opt. Instrum. Eng. **231,** 84–93 (1980).
18. J. R. Fienup, "Reconstruction and synthesis applications of an iterative algorithm," in *Transformations in Optical Signal Processing*, W. T. Rhodes, J. R. Fienup, and B. E. A. Saleh, eds., Proc. Soc. Photo-Opt. Instrum. Eng. **373,** 147–160 (1981).
19. D. C. Youla, "Generalized image restoration by method of alternating orthogonal projections," IEEE Trans. Circuits Syst. **CAS-25,** 694–702 (1978).
20. J. R. Fienup, T. R. Crimmins, and W. Holsztynski, "Reconstruction of the support of an object from the support of its autocorrelation," J. Opt. Soc. Am. **72,** 610–624 (1982).
21. J. R. Fienup, "Phase retrieval from a single intensity distribution," in *Optics in Modern Science and Technology*, proceedings of the International Commission for Optics-13, August 20–24, 1984, Sapporo, Japan (Optics-13 Conference Committee, Sapporo, Japan, 1984).
22. M. S. Scivier and M. A. Fiddy, "Phase ambiguities and the zeros of multi-dimensional band-limited functions," J. Opt. Soc. Am. A **2,** 693–697 (1985).
23. J. R. Fienup and G. B. Feldkamp, "Astronomical imaging by processing stellar speckle interferometry data," in *Applications of Speckle Phenomena*, W. H. Carter, ed., Proc. Soc. Photo-Opt. Instrum. Eng. **243,** 95–102 (1980).
24. R. H. T. Bates and W. R. Fright, "Composite two-dimensional phase-restoration procedure," J. Opt. Soc. Am. **73,** 358–365 (1983).
25. R. J. Sault, "Two procedures for phase estimation from visibility magnitudes," Aust. J. Phys. **37,** 209–229 (1984).
26. K. T. Knox and B. J. Thompson, "Recovery of images from atmospherically degraded short-exposure photographs," Astrophys. J. Lett. **193,** L45–L48 (1974).
27. P. vanToorn, A. H. Greenaway, A. M. J. Huiser, "Phaseless object reconstruction," Opt. Acta **7,** 767–774 (1984).
28. J. R. Fienup, "Experimental evidence of the uniqueness of phase retrieval from intensity data," in *Indirect Imaging, Proceedings of URSI/IAU Symposium* (Cambridge U., Cambridge, 1984), pp. 99–109.
29. R. N. Bracewell, *The Fourier Transform and Its Applications*, 2nd ed. (McGraw-Hill, New York, 1978), pp. 271–272.
30. J. R. Fienup and C. C. Wackerman, "Improved phase retrieval algorithm," J. Opt. Soc. Am. A **1,** 1320 (A) (1984).