Robust two-dimensional weighted and unweighted phase unwrapping that uses fast transforms and iterative methods

Dennis C. Ghiglia and Louis A. Romero

Sandia National Laboratories, Albuquerque, New Mexico 87185

Received March 1, 1993; revised manuscript received May 10, 1993; accepted June 22, 1993

Two-dimensional (2D) phase unwrapping continues to find applications in a wide variety of scientific and engineering areas including optical and microwave interferometry, adaptive optics, compensated imaging, and synthetic-aperture-radar phase correction, and image processing. We have developed a robust method (not based on any path-following scheme) for unwrapping 2D phase principal values (in a least-squares sense) by using fast cosine transforms. If the 2D phase values are associated with a 2D weighting, the fast transforms can still be used in iterative methods for solving the weighted unwrapping problem. Weighted unwrapping can be used to isolate inconsistent regions (i.e., phase shear) in an elegant fashion.

1. INTRODUCTION

Researchers grappling with wave-front-reconstruction problems in adaptive optics and speckle imaging developed the mathematical formalism for reconstructing a two-dimensional (2D) phase surface from noisy phase-difference measurements.[1-3] Since these phase differences are usually obtained in two orthogonal directions (i.e., x and y) from wave-front tilt sensors, the problem is to reconstruct the phase, in a least-squares sense, from these noisy measurements. The papers by Fried[1] and Hudgin[2] provide a thorough background.

Hunt[4] placed the 2D phase-reconstruction problem in the solid mathematical framework of matrix algebra, tied this formulation to that of Fried and Hudgin, and discussed some direct and iterative methods for the least-squares solution of the resulting linear equations. Much of the previous research has been applied to reconstruction of the phase on a relatively small grid [i.e., $O(64 \times 64)$ or less] on which simple iterative schemes such as those of Jacobi, Gauss–Seidel, or successive overrelaxation may be feasible. Solutions on larger grids usually require a more-direct approach, including fast sparse-matrix software and Fourier-transform methods.[5,6]

Recent research, driven in part by synthetic-aperture-radar signal processing,[7] has shown that the least-squares solution to the phase unwrapping problem is mathematically identical to the solution of Poisson's equation on a rectangular grid with Neumann boundary conditions.[8] Therefore fast, direct methods with specialized elliptic partial differential equation solvers can be applied in many cases.[9] Further experimentation (since the publication of Ref. 8) has shown that even library-quality subroutines can have numerical problems on 32-bit machines for grids that exceed 512×512.

Some synthetic-aperture-radar processing applications force the requirement for phase unwrapping on very large grids [i.e., $O(16\text{K} \times 16\text{K})$], efficient use of available memory, and 32-bit precision, as is commonly available on modern workstations. With these requirements in mind, we reexamined the phase unwrapping problem and determined that all the above requirements could be met, without numerical difficulty, by solution of Poisson's equation with a specific form of the fast cosine transform. This approach is numerically stable and robust, exactly solves Poisson's equation, automatically imposes the proper boundary conditions, can be posed as a separable process (computationally efficient), and can be performed in place with regard to available memory.

It should be noted that all prior research on least-squares 2D phase unwrapping has dealt directly with the resulting normal equations[4] without regard to the notion of specific data integrity. If certain wrapped phase values are known to be less reliable than others because of regionally varying noise, aliasing (i.e., undersampling a complex function before computing its argument), phase inconsistencies (i.e., shear; see Ref. 10 for discussion and examples), or measurement errors, the phase unwrapping should be posed in the weighted sense. To the best of our knowledge, no one else has considered the weighted least-squares solution of the 2D phase unwrapping problem on very large rectangular grids.

It should be made clear that our least-squares formulation is based solely on attacking path-integral inconsistencies in a robust manner. Inconsistencies are present when multiples of 2π rad cannot be added to each wrapped phase sample over a 2D grid to eliminate all adjacent phase differences greater than π rad in magnitude.

The algorithms were not developed (nor do we know how to develop a method) for obtaining a least-squares unwrapped phase estimate, in the presence of noise, where there are no inconsistencies (i.e., the algorithms do not smooth out the effects of noise unless the noise causes inconsistencies). No 2D phase unwrapping method that we know of does that. In other words, if the original wrapped phase values do not have any path inconsistencies, the output of our algorithms are exactly equivalent to that obtained with a path-following integration method. How-

0740-3232/94/010107-11$06.00

ever, when path inconsistencies are present (as is the case in virtually all practical 2D problems) the least-squares formulation handles these robustly without need for interjection of branch cuts or other heuristically derived methods to resolve path integral discrepancies.[11,12]

We show, in the sections that follow, the derivation of the technique for 2D unweighted least-squares phase unwrapping with use of fast cosine transforms and how this method forms the heart of efficient iterative techniques for solving the weighted least-squares phase unwrapping problem. Weighted 2D phase unwrapping elegantly solves the problems of regional inconsistencies by permitting arbitrary region partitioning or exclusion through assignment of corresponding zero-valued weights. These new algorithms obviate the need for any unwrapping method based on path following.

2. MATHEMATICAL DEVELOPMENT

A. Two-Dimensional Unweighted Phase Unwrapping

Let us assume that we know the phase, ϕ modulo 2π, of a function on a discrete grid of points:

$$\begin{aligned} &\psi_{i,j} = \phi_{i,j} + 2\pi k, \qquad k \text{ an integer}, \\ &-\pi < \psi_{i,j} \le \pi, \qquad i = 0 \ldots M-1, \qquad j = 0 \ldots N-1. \end{aligned} \tag{1}$$

Given the wrapped phase values $\psi_{i,j}$, we wish to determine the unwrapped phase values $\phi_{i,j}$ at the same grid locations, with the requirement that the phase differences of the $\phi_{i,j}$ agree with those of the $\psi_{i,j}$ in the least-squares sense. To see how this is done, let us first define a wrapping operator W that wraps all values of its argument into the range $(-\pi, \pi)$ by adding or subtracting an integral number of 2π rad from its argument. Therefore, for example, $W\{\phi_{i,j}\} = \psi_{i,j}$.

Next we compute two sets of phase differences: those differences with respect to the i index and those with respect to the j index. Specifically, from our known values of the wrapped phase $\psi_{i,j}$, we compute the following wrapped phase differences:

$$\begin{aligned} \Delta^x_{i,j} &= W\{\psi_{i+1,j} - \psi_{i,j}\} \\ &\quad i = 0 \ldots M-2, \qquad j = 0 \ldots N-1, \\ \Delta^x_{i,j} &= 0, \qquad \text{otherwise}; \end{aligned} \tag{2}$$

$$\begin{aligned} \Delta^y_{i,j} &= W\{\psi_{i,j+1} - \psi_{i,j}\}, \\ &\quad i = 0 \ldots M-1, \qquad j = 0 \ldots N-2, \\ \Delta^y_{i,j} &= 0, \qquad \text{otherwise}, \end{aligned} \tag{3}$$

where the x and y superscripts refer to differences in the i and j indices, respectively. The solution, $\phi_{i,j}$, that minimizes

$$\sum_{i=0}^{M-2}\sum_{j=0}^{N-1} (\phi_{i+1,j} - \phi_{i,j} - \Delta^x_{i,j})^2 + \sum_{i=0}^{M-1}\sum_{j=0}^{N-2} (\phi_{i,j+1} - \phi_{i,j} - \Delta^y_{i,j})^2$$

is the least-squares solution.

Hunt's matrix formulation shows that the normal equations leading to the least-squares phase unwrapping solution can be summarized by the following simple equation[4]:

$$\phi_{i+1,j} + \phi_{i-1,j} + \phi_{i,j+1} + \phi_{i,j-1} - 4\phi_{i,j} = \Delta^x_{i,j} - \Delta^x_{i-1,j} + \Delta^y_{i,j} - \Delta^y_{i,j-1}. \tag{4}$$

Equation (4) gives the relationship between the wrapped phase differences [available from the original wrapped phases by means of Eqs. (2) and (3)] and the unwrapped phase values $\phi_{i,j}$, in the least-squares-error sense. A simple manipulation of Eq. (4) yields

$$(\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}) + (\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}) = \rho_{i,j}, \tag{5}$$

where

$$\rho_{i,j} = (\Delta^x_{i,j} - \Delta^x_{i-1,j}) + (\Delta^y_{i,j} - \Delta^y_{i,j-1}). \tag{6}$$

It is now easy to see that Eq. (5) is a discretization of Poisson's equation on a rectangular $M \times N$ grid,

$$\frac{\partial^2}{\partial x^2}\phi(x,y) + \frac{\partial^2}{\partial y^2}\phi(x,y) = \rho(x,y). \tag{7}$$

It is important to note that Eq. (5) is valid for all indices on the rectangular grid, $i = 0 \ldots M-1, j = 0 \ldots N-1$, and that the appropriate phase differences that are used to compute $\rho_{i,j}$ in Eq. (6) are nonzero only if they come from phases entirely within the rectangular grid [i.e., Eqs. (2) and (3)]. This requirement results directly from the least-squares formulation and forms the discrete equivalent of the imposition of Neumann boundary conditions on Poisson's equation. Specifically [as stated in Eqs. (2) and (3)], we require that

$$\begin{aligned} \Delta^x_{-1,j} &= 0, \\ \Delta^x_{M-1,j} &= 0, \qquad j = 0 \ldots N-1, \end{aligned} \tag{8}$$

$$\begin{aligned} \Delta^y_{i,-1} &= 0, \\ \Delta^y_{i,N-1} &= 0, \qquad i = 0 \ldots M-1. \end{aligned} \tag{9}$$

It is also worth noting that the wrapped phase differences given by Eqs. (2) and (3) are the equivalent of the measured phase differences in the formulation of Hunt[4] (and others). Since 2D phase unwrapping had its roots in adaptive optics, phase differences obtained from wavefront tilt sensors were necessarily wrapped values. Therefore, since we are formulating the 2D unwrapping from measured or computed values of wrapped *phase*, wrapped phase differences, not ordinary phase differences, must be used in the formation of the driving term (or right-hand side) $\rho_{i,j}$ of the resulting partial differential equation [Eq. (5)].

We now show how to solve Eq. (4) [or Eq. (5)] by using a specific form of a cosine expansion that leads to a fast discrete cosine transform (DCT) implementation. A specific form of the 2D discrete cosine transform pair is as follows[13]:

Forward 2D DCT:

$$C_{m,n} = \begin{cases} \displaystyle\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} 4x_{i,j} \cos\left[\frac{\pi}{2M}m(2i+1)\right]\cos\left[\frac{\pi}{2N}n(2j+1)\right]. \\ \qquad 0 \le m \le M-1; \qquad 0 \le n \le N-1 \\ 0 \qquad \text{otherwise} \end{cases} \tag{10}$$

Inverse 2D DCT:

$$x_{i,j} = \begin{cases} \dfrac{1}{MN}\displaystyle\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} w_1(m)w_2(n)C_{m,n}\cos\left[\frac{\pi}{2M}m(2i+1)\right]\cos\left[\frac{\pi}{2N}n(2j+1)\right], \\ \qquad 0 \le i \le M-1, \quad 0 \le j \le N-1 \\ 0 \qquad \text{otherwise} \end{cases},$$

$$\begin{aligned} w_1(m) &= 1/2, \quad m = 0, \\ w_1(m) &= 1, \quad 1 \le m \le M-1, \\ w_2(m) &= 1/2, \quad n = 0, \\ w_2(m) &= 1, \quad 1 \le n \le N-1. \end{aligned} \tag{11}$$

It is important to note that the above cosine expansion imposes the Neumann boundary conditions, $\nabla\phi \cdot \mathbf{n} = 0$, automatically and leads to the exact solution of Eq. (5), as is shown below.

Let us expand the desired solution $\phi_{i,j}$ in the form of Eq. (11):

$$\phi_{i,j} = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} w_1(m)w_2(n)\hat{\phi}_{m,n} \times \cos\left[\frac{\pi}{2M}m(2i+1)\right]\cos\left[\frac{\pi}{2N}n(2j+1)\right] \tag{12}$$

and substitute Eq. (12) into Eq. (5). Performing a similar expansion and substitution for the right-hand side of Eq. (5) and rearranging the result yields the following exact solution in the DCT domain:

$$\hat{\phi}_{i,j} = \frac{\hat{\rho}_{i,j}}{2\left((\cos\frac{\pi i}{M} + \cos\frac{\pi j}{N} - 2\right)}. \tag{13}$$

The unwrapped phase $\phi_{i,j}$ is now easily obtained by the inverse DCT of Eq. (13).

It is easily shown by simple substitution that the expansion given by Eq. (12) automatically imposes the discrete equivalent of the required Neumann boundary conditions:

$$\begin{aligned} \phi_{0,j} - \phi_{-1,j} &= 0, \\ \phi_{M,j} - \phi_{M-1,j} &= 0, \quad j = 0 \ldots N-1 \\ \phi_{i,0} - \phi_{i,-1} &= 0, \\ \phi_{i,N} - \phi_{i,N-1} &= 0, \quad i = 0 \ldots M-1. \end{aligned} \tag{14}$$

The 2D unweighted phase unwrapping algorithm is summarized as follows:

Algorithm 1

1. Perform the 2D forward DCT [Eq. (10)] of the array of values, $\rho_{i,j}$, computed by Eq. (6), to yield the 2D DCT values $\hat{\rho}_{i,j}$.
2. Modify the values $\hat{\rho}_{i,j}$ according to Eq. (13) to obtain $\hat{\phi}_{i,j}$.
3. Perform the 2D inverse DCT [Eq. (11)] of $\hat{\phi}_{i,j}$ to obtain the least-squares unwrapped phase values $\phi_{i,j}$.

It is important to note that we cannot evaluate Eq. (13) for $i = 0$ and $j = 0$ simultaneously, because the denominator becomes zero. This means that $\hat{\phi}_{0,0}$ is indeterminate, because Poisson's equation cannot be solved for a constant bias component. In practice, we usually set $\hat{\phi}_{0,0} = \hat{\rho}_{0,0}$ to leave the bias unchanged.

B. Two-Dimensional Weighted Phase Unwrapping

In Subsection 2.A we showed how to solve the overdetermined set of linear equations (of Hunt[4]),

$$\mathbf{Ax} = \mathbf{b}, \tag{15}$$

in a least-squares sense with cosine transforms. The least-squares solution is, of course, the solution to the normal equations,

$$\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b}, \tag{16}$$

where $\mathbf{x}$ is a length MN solution vector of phase values, $\mathbf{b}$ is a vector of length $N(M-1) + M(N-1)$ containing the wrapped phase differences, and $\mathbf{T}$ refers to matrix transposition. We can rewrite Eq. (16) as

$$\mathbf{P}\phi = \rho, \tag{17}$$

where $\mathbf{P} = \mathbf{A}^T\mathbf{A}$ is a matrix that performs the discrete Laplacian operation on the vector ϕ as illustrated in Eq. (5) and $\rho = \mathbf{A}^T\mathbf{b}$ is a vector containing the discrete Laplacian operation on the wrapped phase differences as illustrated in Eq. (6). Therefore we now know how to solve Eq. (17) by using algorithm 1.

When certain wrapped phase values are known to be corrupted by regionally varying noise, measurement errors, aliasing, or other degrading influences, one should weight those values appropriately to obtain a more robust unwrapping. In addition, *a priori* information based on the physics of the underlying problem may dictate that certain regions of the unwrapped phase be isolated from one another. For example, phase surfaces that contain regional shearing or isolated regions of no phase information whatsoever can be weighted so that the unwrapped phase solution is not required to meet continuity constraints across any shearing or defined regional boundary.

We now turn our attention to solving the weighted least-squares phase unwrapping problem. Given that we know how to solve Eq. (15) in the least-squares sense, we now wish to solve the weighted least-squares problem,

$$\mathbf{WAx} = \mathbf{Wb}. \tag{18}$$

The resulting normal equations for the weighted least-squares problem are

$$\mathbf{A}^T\mathbf{W}^T\mathbf{WAx} = \mathbf{A}^T\mathbf{W}^T\mathbf{Wb}, \tag{19}$$

where $\mathbf{W}$ is a matrix of weighting values.

Now let $\mathbf{Q} = \mathbf{A}^T\mathbf{W}^T\mathbf{WA}$, $\bar{\mathbf{b}} = \mathbf{W}^T\mathbf{Wb}$, and substitute these into Eq. (19) to obtain

$$\mathbf{Qx} = \mathbf{A}^T\bar{\mathbf{b}}. \tag{20}$$

With the final substitution,

$$\mathbf{c} = \mathbf{A}^T\bar{\mathbf{b}}, \tag{21}$$

and associating the desired solution ϕ with $\mathbf{x}$, we obtain

$$\mathbf{Q}\phi = \mathbf{c}. \tag{22}$$

It is important to note that $\bar{\mathbf{b}}$ is simply a vector consisting of appropriately *weighted* phase differences and that Eq. (21) indicates that $\mathbf{c}$ is nothing more than a vector formed from the modified discrete Laplacian (to be defined below) of the *weighted* wrapped phase differences. The formation of $\mathbf{c}$ is entirely analogous to what we did to form the right-hand side of Eq. (17). Therefore Eq. (22) is the matrix equation defining the 2D weighted least-squares phase unwrapping problem. We now provide two algorithms for solving Eq. (22), using our ability to solve Eq. (17).

C. Picard Iteration Method for Two-Dimensional Weighted Least-Squares Phase Unwrapping

Incorporation of the weighting matrix makes it impossible for us to solve Eq. (22) by using the discrete cosine transform expansion directly as we did for the unweighted case. However, we can incorporate our ability to solve the unweighted problem into a simple interative scheme that exactly solves the weighted problem. The technique is based on a simple Picard iteration.

Let us split the matrix $\mathbf{Q}$ into the sum of the matrix $\mathbf{P}$ [from Eq. (17)] and a difference matrix, $\mathbf{D}$, to obtain

$$\mathbf{Q} = \mathbf{P} + \mathbf{D}. \tag{23}$$

Substitution of Eq. (23) int Eq. (22) yields

$$(\mathbf{P} + \mathbf{D})\phi = \mathbf{c} \tag{24}$$

or

$$\mathbf{P}\phi = \mathbf{c} - \mathbf{D}\phi. \tag{25}$$

Equation (25) is now solved iteratively,

$$\mathbf{P}\phi_{k+1} = \mathbf{c} - \mathbf{D}\phi_k, \tag{26}$$

where k refers to the iteration number.

Recall that

$$\mathbf{D}\phi_k = (\mathbf{Q} - \mathbf{P})\phi_k, \tag{27}$$

where $\mathbf{Q} = \mathbf{A}^T\mathbf{W}^T\mathbf{W}\mathbf{A}$ performs the modified discrete Laplacian operation on the weighted phase differences and $\mathbf{P} = \mathbf{A}^T\mathbf{A}$ performs the ordinary discrete Laplacian operation on the unweighted phase differences. Therefore $\mathbf{D}\phi_k$ is nothing more than a matrix difference operator operating on ϕ_k to produce a vector containing the modified Laplacian of the weighted phase values at iteration k minus the ordinary Laplacian of the unweighted phase values at iteration k. More-specific information will be given in Section 3 on exactly how the weighting is used in computing the modified discrete Laplacian.

Equation (26) is the basis for a simple iterative scheme for solving the 2D weighted least-squares unwrapping problem by incorporating the efficient DCT expansion solution method for the unweighted case. Since fast DCT algorithms exist (and are discussed later in the paper), efficient solution of Eq. (26) is summarized in the following algorithm.

Algorithm 2

1. Compute the vector $\mathbf{c}$ containing the modified discrete Laplacian formed from the original weighted and wrapped phase differences, and save it.
2. Specify the maximum number of iterations, $k_{\max}$.
3. Set the iteration counter, $k = 0$ and $\phi_k = 0$ (or some initial guess).
4. Compute the vector $\rho_k = \mathbf{c} - \mathbf{D}\phi_k$.
5. Solve $\mathbf{P}\phi_{k+1} = \rho_k$ [Eq. (17), with $\rho = \rho_k$], using algorithm 1 to obtain ϕ_{k+1}.
6. If $k < k_{\max}$, continue, otherwise stop and set the final solution to ϕ_{k+1}.
7. Update the iteration counter, $k = k + 1$.
8. Go to step 4 above.

Algorithm 2 has the advantages that (a) it is easy to implement, (b) it requires minimal additional storage since most operations can be done in place, and (c) if it converges (which is not guaranteed), it converges to the correct solution of the weighted least-squares problem. It is also possible for one to monitor convergence by computing the norm of the residual, $\|\mathbf{D}\phi_k\|$, at each iteration.

Potential disadvantages include the lack of guaranteed convergence and that an excessive number of iterations may be required for the application at hand. A faster-converging algorithm will now be developed on the basis of the method of preconditioned conjugate gradients (PCG's).[14] The PCG technique allows us again to solve Eq. (22) interatively, utilize our DCT method of algorithm 1 in a fashion similar to algorithm 2, and obtain faster (and guaranteed) convergence. The disadvantage lies in the fact that PCG may require excessive storage and does not permit in-place computation. More will be said about implementation details of all the algorithms in a later section.

D. Preconditioned Conjugate Gradient for Two-Dimensional Weighted Least-Squares Phase Unwrapping

Conjugate gradient (CG) methods are a means of iteratively solving a set of sparse linear equations, of which the discretized version of Poisson's equation [Eq. (5)] is one. The CG technique poses the solution of the linear equations as a minimization problem and obtains much faster convergence than do simpler minimization schemes such as steepest descent.[14] The CG algorithm has robust convergence properties with well-defined termination conditions. If there were no roundoff error, CG would converge in exactly N iterations for an $N \times N$ problem. However, for large N the number of iterations may still be excessive, and exact convergence is not guaranteed because of numerical roundoff error. In practice, CG is used as a true iterative algorithm for obtaining an approximate solution in much less than N iterations.

For large problems, the number of iterations required may still be excessive. The actual number of iterations required for obtaining an accurate solution depends on the condition number of the underlying matrix. If the underlying matrix is near the identity matrix, CG converges quickly. Preconditioned conjugate gradient methods apply a preconditioning step that effectively trans-

forms the underlying matrix to one very near the identity matrix; hence rapid convergence is obtained.

To be useful in PCG, the preconditioner must both be close to the matrix of interest and represent an easily solvable linear system. In our case we obtain the preconditioning by solving an approximate problem (i.e., the *unweighted* phase unwrapping problem) and then updating the solution at each iteration. The approximate problem is solved with our previously derived algorithm 1. The solution of this preconditioning step is then used to obtain an estimate for the true solution to the exact problem. The PCG method of 2D weighted least-squares phase unwrapping is summarized in the following algorithm.

Algorithm 3

1. $k = 0, \quad \phi_0 = 0, \quad \mathbf{r}_0 = \mathbf{c}.$
2. While ($\mathbf{r}_k \neq 0$), solve $\mathbf{P}\mathbf{z}_k = \mathbf{r}_k$, using algorithm 1.
3. $k = k + 1.$
4. If $k = 1$, $\mathbf{p}_1 = \mathbf{z}_0.$
5. If $k > 1$, then

$$\beta_k = \mathbf{r}_{k-1}{}^T\mathbf{z}_{k-1}/\mathbf{r}_{k-2}{}^T\mathbf{z}_{k-2},$$
$$\mathbf{p}_k = \mathbf{z}_{k-1} + \beta_k\mathbf{p}_{k-1}.$$

6. Perform one scalar and two vector updates,

$$\alpha_k = \mathbf{r}_{k-1}{}^T\mathbf{z}_{k-1}/\mathbf{p}_k{}^T\mathbf{Q}\mathbf{p}_k,$$
$$\phi_k = \phi_{k-1} + \alpha_k\mathbf{p}_k,$$
$$\mathbf{r}_k = \mathbf{r}_{k-1} - \alpha_k\mathbf{Q}\mathbf{p}_k.$$

7. If $k \geq k_{\max}$, or $\|\mathbf{r}_k\| < \varepsilon\|\mathbf{r}_0\|$, end; otherwise go to step 2 above.

The beauty of this algorithm is that it allows us to incorporate the DCT method efficiently within the robust structure of CG's to solve the weighted least-squares problem iteratively. The reader is encouraged to consult Ref. 14 for a thorough treatment of CG methods.

3. IMPLEMENTATION ISSUES

The forward and the inverse 2D DCT's given in Eqs. (10) and (11), respectively, can be computed by fast algorithms when M and N are powers of 2. In addition, the DCT is separable as is the fast Fourier transform, thus permitting independent row and column transforms, and all transforms can be accomplished in place with regard to available memory.

We have chosen to implement the 2D DCT as a separable process and to utilize readily available FORTRAN subroutines for computing the fast summation expansions. For example, let us write the one-dimensional versions of the DCT as shown below:

$$C_m = \begin{cases} \sum_{i=0}^{M-1} 2x_i \cos\left[\frac{\pi}{2M} m(2i+1)\right] & 0 \leq m \leq M-1 \\ 0 & \text{otherwise} \end{cases}, \tag{28}$$

$$x_i = \begin{cases} \frac{1}{M}\sum_{m=0}^{M-1} w_1(m)C_m\left[\cos\frac{\pi}{2M} m(2i+1)\right] & 0 \leq i \leq M-1 \\ 0 & \text{otherwise} \end{cases},$$

$$w_1(m) = 1/2, \qquad m = 0,$$
$$w_1(m) = 1, \qquad 1 \leq m \leq M-1. \tag{29}$$

Expanding the cosine term in Eqs. (28) and (29) and rearranging yields

$$C_m = 2\cos\frac{\pi m}{2M}\sum_{i=0}^{M-1} x_i \cos\frac{\pi m}{M} i - 2\sin\frac{\pi m}{2M}\sum_{i=0}^{M-1} x_i \sin\frac{\pi m}{M} i, \tag{30}$$

$$x_i = \frac{1}{M}\sum_{m=0}^{M-1} {}^1\hat{C}_m \cos\frac{\pi m}{M} i - \frac{1}{M}\sum_{i=0}^{M-1} {}^2\hat{C}_m \sin\frac{\pi m}{M} i, \tag{31}$$

where

$${}^1\hat{C}_m = w_1(m)C_m \cos\frac{\pi m}{2M}, \tag{32}$$

$${}^2\hat{C}_m = w_1(m)C_m \sin\frac{\pi m}{2M}. \tag{33}$$

The summation terms in Eqs. (30) and (31) containing the cosine and sine expansions are easily computed as fast algorithms by means of the FORTRAN subroutines COSFT[15] and SINFT, respectively, of Ref. 16.

It should also be noted that one must compute the denominator of Eq. (13) in double precision and then convert it to single precision to avoid numerical zeros for low values of the i and j indices other than zero, especially for large M and N. This is the only requirement for double-precision computation that we have found in all the unwrapping experiments that we have performed to date.

It is worth a few words at this point to clarify exactly how the weighting array is used in the weighted least-squares problem solved by algorithms 2 and 3. In practice, we would have a weighting array whose elements $0 \leq w_{i,j} \leq 1$ [not to be confused with the vectors $w_1(m)$ and $w_2(m)$ in the DCT equations (10), (11), (28), and (29)] are in one-to-one correspondence with the original wrapped phase values $\psi_{i,j}$. In order to compute the modified Laplacian, one must apply proper weighting to the phase difference, not to the phase values. For any phase difference that we wish to compute, we select the minimum of the two weights corresponding to the two phases as the proper weight to apply to that phase difference.

For example, the elements of the vector $\mathbf{c}$ from Eq. (21), when written out in 2D-array notation, become the modified discrete Laplacian of the original wrapped phases:

$$c_{i,j} = \min(w^2_{i+1,j}, w^2_{i,j})\Delta^x_{i,j} - \min(w^2_{i,j}, w^2_{i-1,j})\Delta^x_{i-1,j} + \min(w^2_{i,j+1}, w^2_{i,j})\Delta^y_{i,j} - \min(w^2_{i,j}, w^2_{i,j-1})\Delta^y_{i,j-1}. \tag{34}$$

Similarly, the right-hand side of Eq. (26) (step 7 of algorithm 2) when written out in 2D-array notation becomes

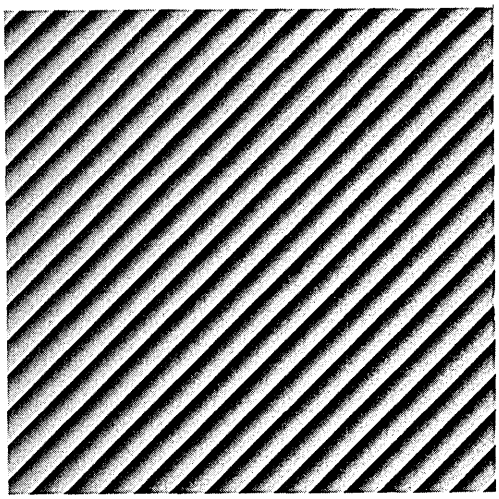


Fig. 1. Wrapped values of a 512 × 512-pixel 2D phase plane (scaled for display).

$$\begin{aligned}\rho_{i,j} = c_{i,j} - [&(w_{x1}^2 - 1)(\phi_{i+1,j} - \phi_{i,j}) \\ &- (w_{x2}^2 - 1)(\phi_{i,j} - \phi_{i-1,j}) \\ &+ (w_{y1}^2 - 1)(\phi_{i,j+1} - \phi_{i,j}) \\ &- (w_{y2}^2 - 1)(\phi_{i,j} - \phi_{i,j-1})], \end{aligned} \tag{35}$$

where

$$\begin{aligned} w_{x1}^2 &= \min(w_{i+1,j}^2, w_{i,j}^2), \\ w_{x2}^2 &= \min(w_{i,j}^2, w_{i-1,j}^2), \\ w_{y1}^2 &= \min(w_{i,j+1}^2, w_{i,j}^2), \\ w_{y2}^2 &= \min(w_{i,j}^2, w_{i,j-1}^2), \end{aligned} \tag{36}$$

The weighting elements are squared because of the $\mathbf{W}^T\mathbf{W}$ operation of the weighted least-squares formulation.

4. EXAMPLES OF TWO-DIMENSIONAL PHASE UNWRAPPING

Let us begin with the wrapped values of a 512 × 512 pixel 2D phase-plane array as shown in Fig. 1. All images depicting wrapped phases in the range $(-\pi, \pi)$ are scaled between black and white (i.e., 0–255) for display. Unwrapped phase images are also scaled between black and white to capture the full dynamic range.

It can be easily seen that this wrapped phase plane represents phase values that increase linearly from the upper left-hand corner to the lower right-hand corner. Unwrapping these values with algorithm 1 yields the phase plane shown in Fig. 2.

Because the dynamic range of the unwrapped result is large, visual display is not always convincing that the unwrapping is correct. Therefore we have chosen to rewrap the unwrapped values to permit a direct comparison with the wrapped input. This rewrapping will be visually convincing that the unwrapping is qualitatively correct and will also easily show how the least-squares formulation manifests itself on inconsistent data. The rewrapped values depicted in Fig. 2 are shown in Fig. 3 and are qualitatively consistent (with the possible exception of an inconsequential constant bias) with the original input depicted in Fig. 1. Because the original input was noise free and totally consistent, the least-squares unwrapping was perfect.

We now illustrate, with a few simple examples, how the least-squares unwrapping accommodates inconsistent data and how the unwrapped values are locally influenced. We modified the perfect data depicted in Fig. 1 by replacing the phase values in a rectangular region with uniform noise, as shown in Fig. 4.

The phase values are inconsistent in the rectangular region and around its boundary. These inconsistencies are accommodated by the least-squares formulation of algorithm 1 and influence the unwrapping regionally. Figure 5 shows the rewrapped phase after unwrapping of the data depicted in Fig. 4. It is easy to see how the least-squares approach accommodates noise and how the unwrapped phase is influenced in the vicinity of inconsistent data with diminishing effects further away.

We show the unwrapped values in Fig. 6 (without rewrapping) to illustrate the apparent quality of the least-squares product. The 2D planar aspect is qualitatively captured, and the influence of the noisy region is minimal. Figure 6 also illustrates why it is important to rewrap the

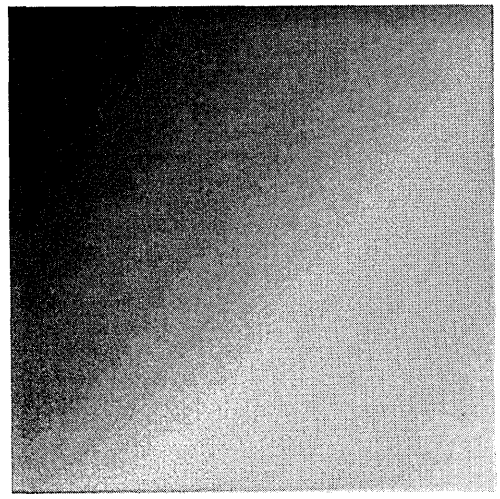


Fig. 2. Unwrapped phase from values depicted in Fig. 1.

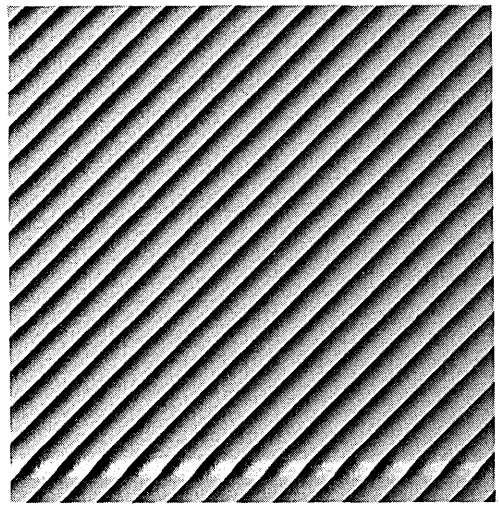


Fig. 3. Rewrapped values from Fig. 2 for direct comparison with Fig. 1. In this case the least-squares unwrapping of totally consistent data yielded perfect results.

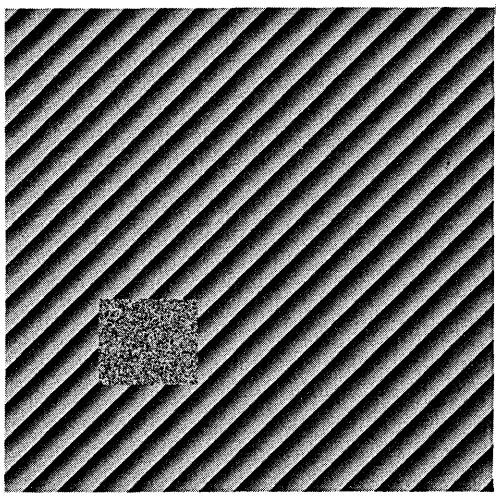


Fig. 4. Wrapped 2D planar phase values with a small rectangular region containing uniformly distributed noise. The phase is inconsistent within the rectangular region and around its boundary.

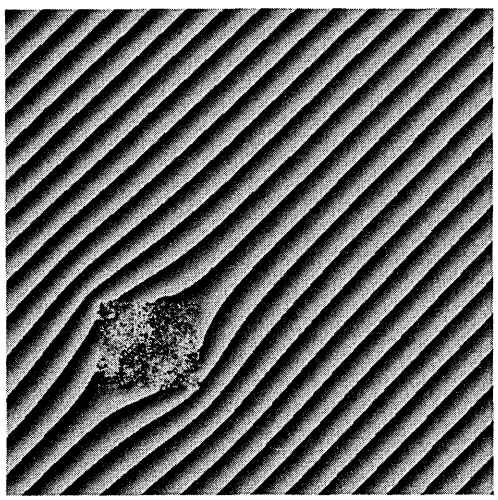


Fig. 5. Rewrapped values from algorithm 1 applied to Fig. 4. Inconsistent data influences the unwrapped results locally, with diminishing effects further away.

values for display so that subtle deviations can be clearly seen (as in Fig. 5).

This is a logical point for bringing up the prospect of weighted phase unwrapping. The last example showed that if good phase values are corrupted or destroyed in a region because of noise, aliasing, etc., the unwrapped results are influenced mostly in the vicinity of the bad values, but they also influence the result globally. Assuming that we have some additional information that allows us to define a corresponding weighting array, we can prevent the noisy phase values from having any influence on the result simply by assigning a weight of zero to the inconsistent data. Figure 7 depicts the corresponding weighting array to accompany the phase values of Fig. 4. The black rectangular region corresponds to zero-valued weights, and the remainder of the 512 × 512 array is set to unity weight and is shown as white. The one-pixel-wide dark outside border is superimposed just to locate the proper array size in the figure.

Figure 8 depicts the result of applying the Picard iteration method (algorithm 2) to the weighted least-squares phase unwrapping problem. In this case, algorithm 2 has relatively fast convergence, requiring approximately 10 iterations to produce a perfect solution. Again, the rewrapped results are shown to highlight subtle behavior. It is interesting that the unwrapped phase is continuous across the noisy region even though the weights were zero. Recall that algorithm 2 is not guaranteed to converge, but if it does converge it provides the correct solution. Algorithm 3, on the other hand, has guaranteed convergence properties and can be used with confidence in all weighted least-squares unwrapping problems, regardless of the relative percentage of the phase values affected by weighting.

The weighting values need not be binary, of course, but can reflect the certainty or uncertainty of the underlying data integrity according to the particular problem at hand. One can have full confidence that the converged solution will be correct in the true weighted least-squares sense.

Another application illustrating the power and robustness of weighted phase unwrapping is shown in the following example. Figure 9 depicts the wrapped (and

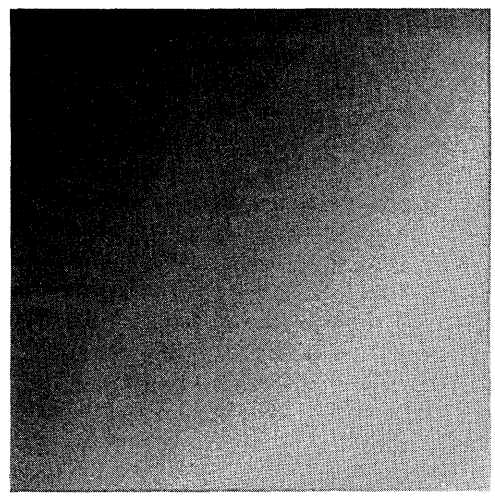


Fig. 6. Unwrapped values of Fig. 4 from algorithm 1 showing minimal influence of noisy region on the final result.

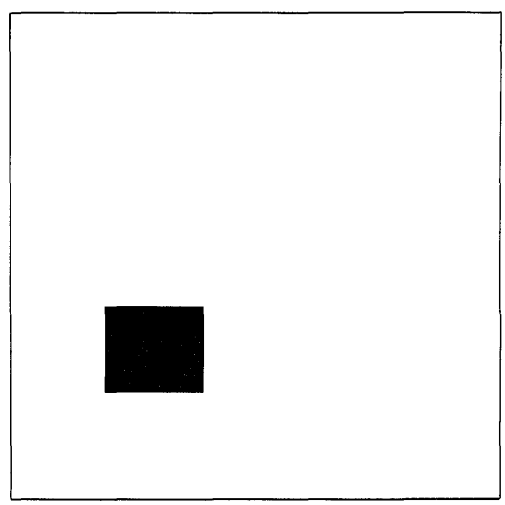


Fig. 7. Two-dimensional weighting array to accompany the wrapped phase data depicted in Fig. 4. Black corresponds to zero-valued weights, and white corresponds to unity weight. The dark border defines the array boundary.

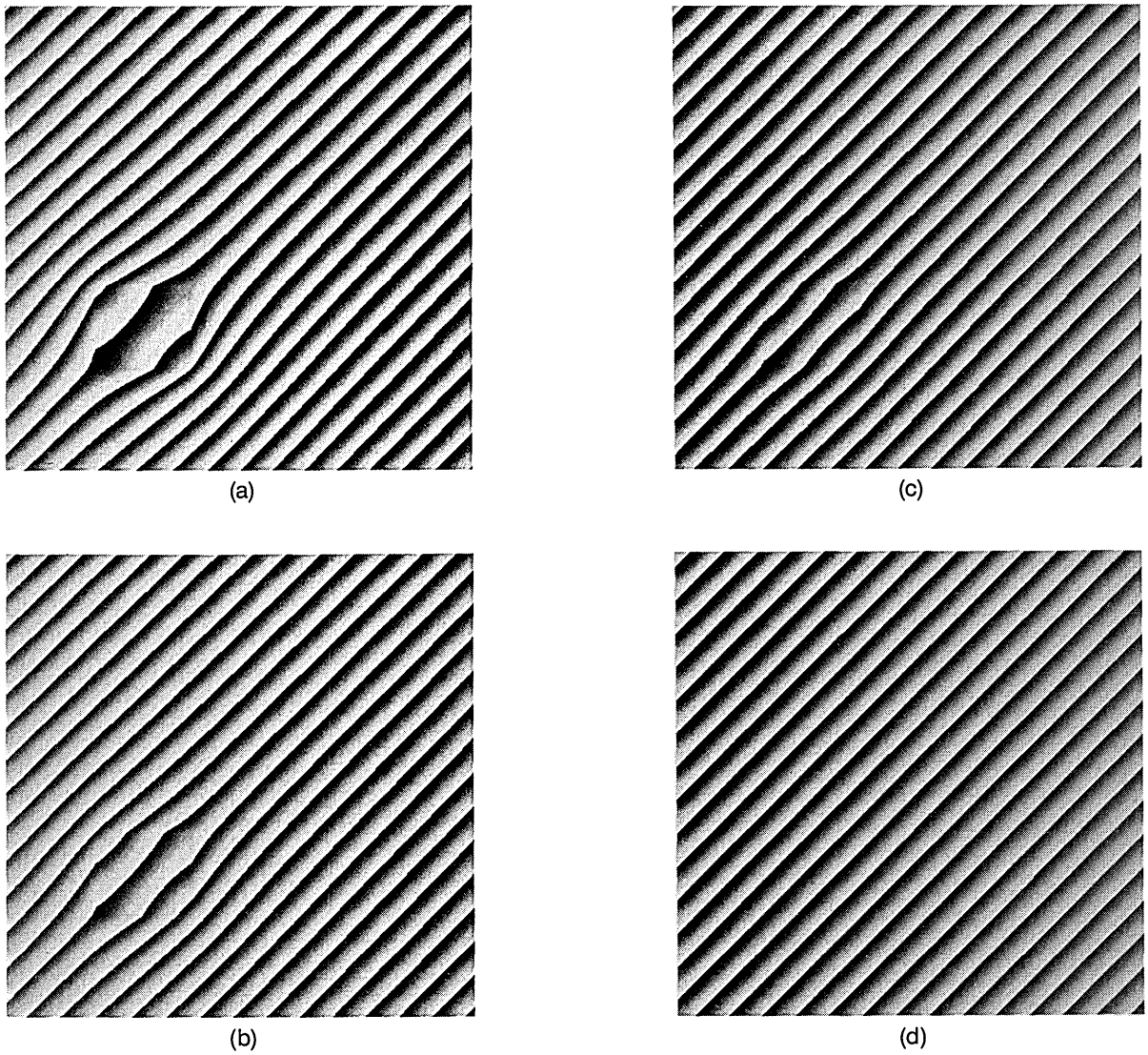


Fig. 8. Sequence of Picard iteration weighted phase unwrapping (algorithm 2) applied to Fig. 4 with the corresponding weighting array of Fig. 7. (a) One iteration, (b) two iterations, (c) three iterations, (d) ten iterations. Note correct solution in region corresponding to unity weights and continuity across region corresponding to zero-valued weights. We show rewrapped phases to highlight subtle features.

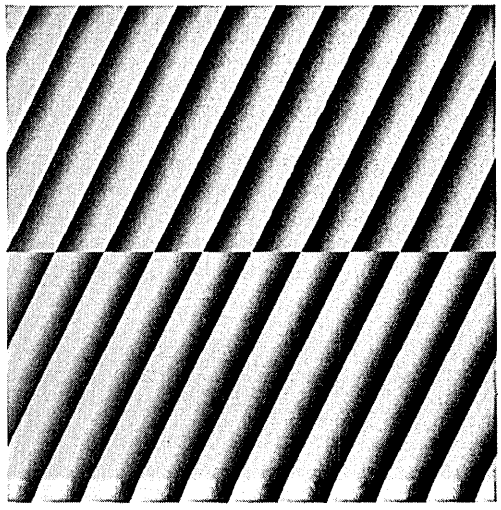


Fig. 9. Wrapped noise-free phase values depicting a phase surface with a shear along the horizontal line midway between the top and the bottom of the figure.

noise-free) phase values of a phase surface with shear. The shear line is horizontal midway between the top and the bottom of the image. Above the shear line the phases represent a planar surface increasing in value from the top-left corner toward the right and down. Below the shear line the phases are sloping linearly downward to the right and down.

If these phase values actually do represent a phase with shear (from the physics of the problem), it is obvious that a global 2D unweighted phase unwrapping produced with use of algorithm 1 cannot reconcile the phase differences everywhere across the shear boundary and yield the true solution. For example, the solution obtained from algorithm 1 is shown in Fig. 10.

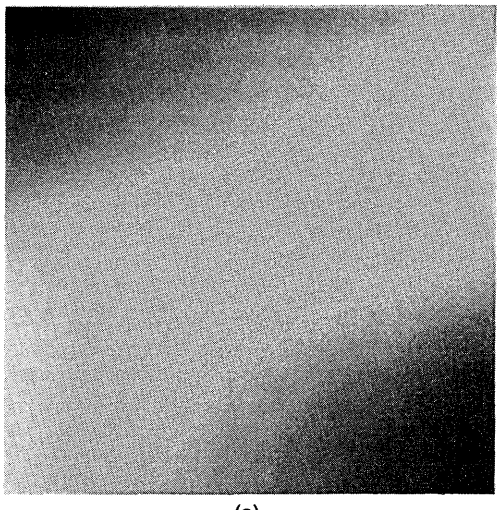
If one has no prior knowledge of shear, Fig. 10(a) is the best that one can hope for. The solution nicely seams together the phases across the horizontal shear line and tends to capture the true phase away from this line. However, if one knows that a shear line exists, it is a simple matter to define a corresponding weighting array contain-

ing a one-sample-wide line of zero-valued weights along the shear line.

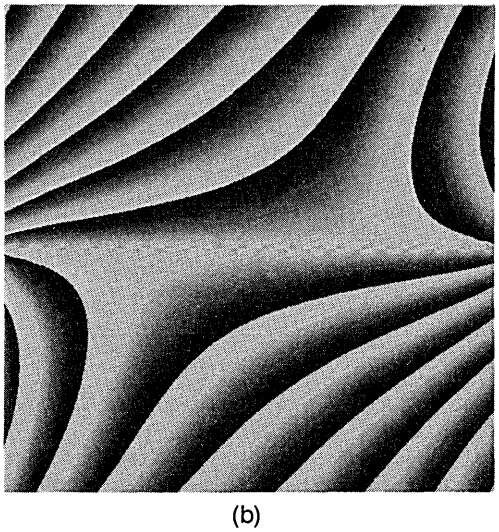
Since the zero-valued weighting line extends across the entire phase array, it is impossible for phases on opposite sides of the line to influence each other. Consequently, the top and bottom halves of the phase array unwrap independently and correctly (since the data were noise free), as is shown in Fig. 11. It was found that algorithm 2 (Picard iteration) did not converge very quickly on this weighted unwrapping (although it did converge after approximately 500 iterations); therefore algorithm 3 (PCG) was used with great success. The results from algorithm 3 are shown in Fig. 11.

Figure 12 depicts a few of the rewrapped results of the Picard iteration method (algorithm 2) for comparison of convergence rates with those of algorithm 3. Final convergence occurred at approximately 500 iterations.

All the examples shown so far were simple in structure (for easy visualization and qualitative assessment) but contained exactly the kinds of phase inconsistency en-

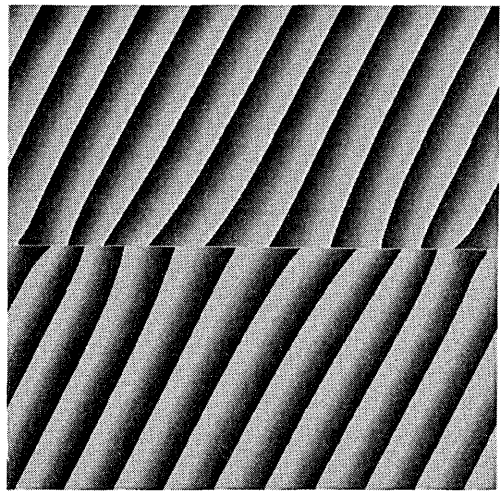


(a)

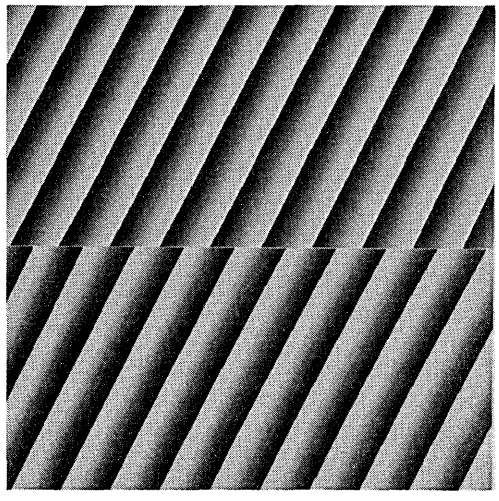


(b)

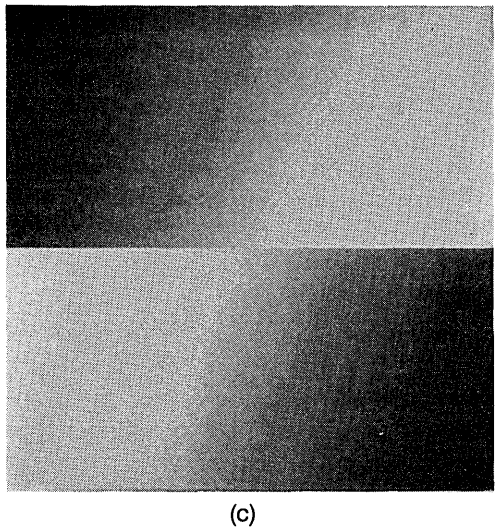
Fig. 10. Unweighted 2D least-squares unwrap of phases that depict a phase shear. (a) Unwrapped result, (b) rewrapped values shown to illustrate solution subtleties and least-squares seaming across the horizontal shear line.



(a)



(b)



(c)

Fig. 11. Weighted unwrap (algorithm 3) of phase data with shear. (a) Rewrapped output after 10 iterations, (b) rewrapped output after 20 iterations, and (c) unwrapped output after 20 iterations. Note that top and bottom halves unwrapped independently, with no influence across the shear line.

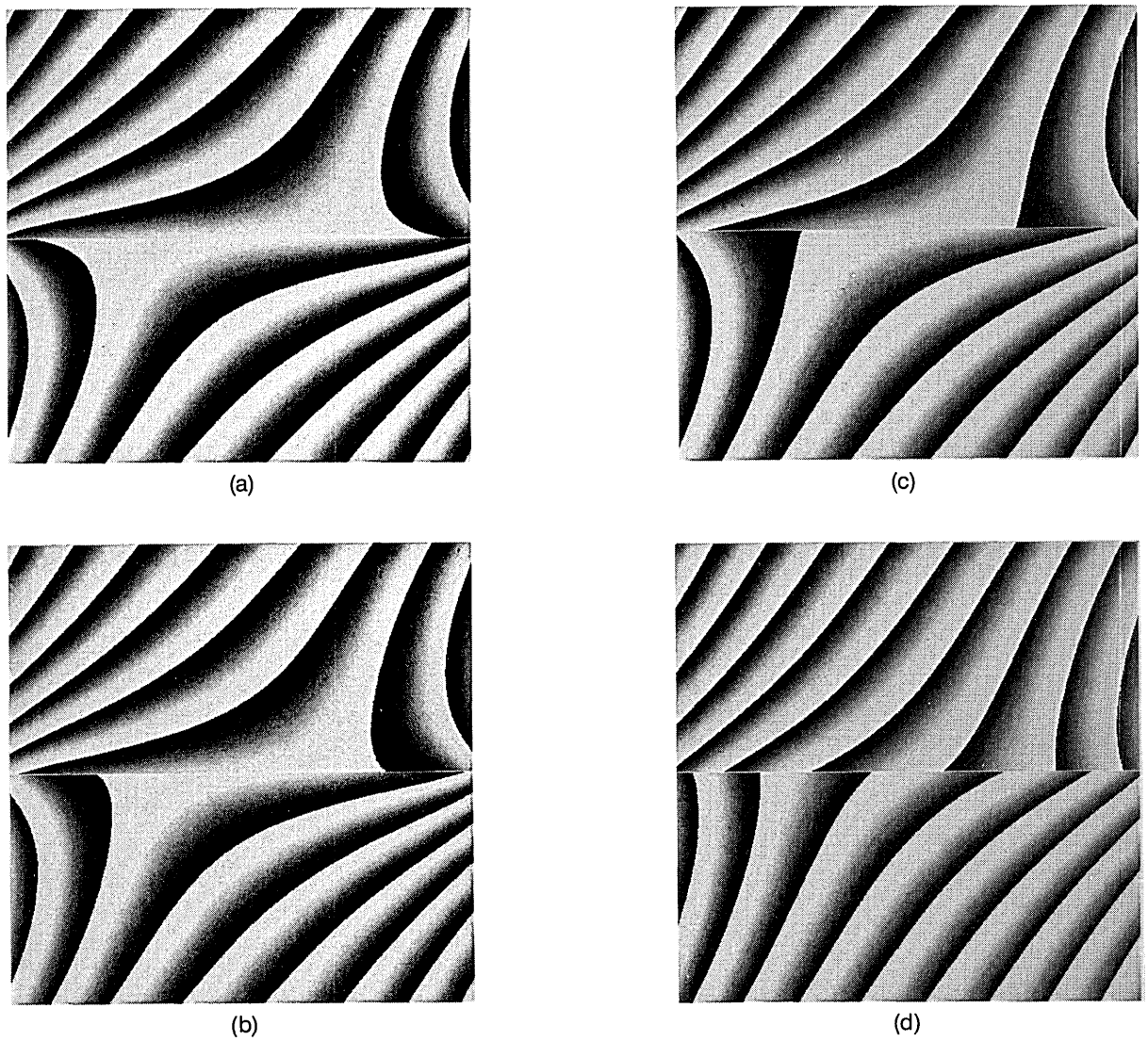


Fig. 12. Some rewrapped results of algorithm 2 applied to Fig. 9 for comparison with the convergence rate of algorithm 3. (a) 10 iterations, (b) 20 iterations, (c) 30 iterations, and (d) 100 iterations. Final convergence was achieved at approximately 500 iterations, in contrast to the convergence at 20 iterations for algorithm 3.

countered in virtually all practical problems. More-complicated wrapped phase data would not provide a more-thorough test or illuminate subtle performance comparisons of the algorithms. One also can use weighted unwrapping to unwrap phases in non–power-of-two-sized rectangular regions, using the fast DCT method by defining unity weighting over the valid region of support and defining zero out to the power-of-two-array boundaries otherwise. Arbitrary regions can be isolated and unwrapped independently and simultaneously by definition of an appropriate weighting array.

If the underlying wrapped phases are totally consistent, any of the weighted unwrapping techniques will converge to the correct result, independently of the weighting values. For example, the data of Fig. 1 are everywhere consistent; therefore the correct unwrapped result would be obtained from algorithm 2 or 3, independently of the weighting values. This result should be expected from any robust algorithm.

The computation required for implementing algorithm 1 is based on fast transforms, as noted earlier in the paper, and is very efficient, requiring $O(N^2 \log_2 N)$ operations for the 2D DCT of an $N \times N$ array. The time required for performing a 2D unweighted unwrapping on a 512×512 sample file of wrapped phase values is of the order of 10 s for a SUN Sparc 10 workstation. Since algorithm 1 is embedded in the iteration loop for both algorithm 2 and algorithm 3 and forms the bulk of the computation, the time required for solving a weighted 2D phase unwrapping is of the order of $k_{\max}$ times the time for an unweighted unwrap, where $k_{\max}$ is the maximum number of iterations.

For input data files much larger than available physical memory, efficient disk-to-disk matrix transpose routines exist[17] for facilitating implementation of the 2D DCT. We have successfully unwrapped files exceeding $8K \times 8K$ samples, in single-precision arithmetic, on Sparc workstations in approximately 2 h (including input–output) with use of algorithm 1. Because of the iterative nature of algorithms 2 and 3, we typically use smaller data sets for weighted unwrapping problems. Although there are

no fundamental restrictions on data-set sizes, it is much more efficient to fit the problem entirely within available memory when one is using the iterative techniques.

5. CONCLUSION

We have developed three algorithms that, we believe, span a rather large range of requirements for 2D phase unwrapping of sampled data. Phase noise, data inconsistencies, and other degradations are automatically accommodated by the least-squares formulation without operator intervention (unlike path-following unwrapping algorithms).

When additional information is available, the weighted least-squares algorithms offer great potential. For example, the incorporation of a weighting array permits exact unwrapping of phase data with shears, deemphasis of suspect phase values, elegant elimination of totally corrupted regions, arbitrarily shaped region unwrapping, and simultaneous unwrapping of multiple isolated arbitrary regions.

All the algorithms use an efficient implementation of the fast 2D DCT as the basis for least-squares unwrapping. Separable and in-place computation, simple algorithmic structures, and efficient input–output capabilities allow these algorithms to be implemented easily on modern workstations. Very large 2D phase unwrapping problems have been successfully performed, in single-precision (32-bit floating point) arithmetic, on these workstations in a few seconds to a couple of hours, depending on data-set size and problem requirements. We hope that these new capabilities will find application in some exciting new areas of signal and image processing.

ACKNOWLEDGMENTS

We thank our colleagues (in alphabetical order) T. M. Calloway, P. H. Eichel, C. V. Jakowatz, Jr., G. A. Mastin, P. A. Thompson, and D. E. Wahl for the many hours of stimulating technical discussions revolving around this and other research. We also thank the reviewers for their helpful comments. This research was performed at Sandia National Laboratories, supported by the U.S. Department of Energy under contract DE-AC04-76DP00789.

REFERENCES AND NOTES

1. D. L. Fried, "Least-squares fitting a wave-front distortion estimate to an array of phase-difference measurements," J. Opt. Soc. Am. **67,** 370–375 (1977).
2. R. H. Hudgin, "Wave-front reconstruction for compensated imaging," J. Opt. Soc. Am. **67,** 375–378 (1977).
3. R. J. Noll, "Phase estimates from slope-type wave-front sensors," J. Opt. Soc. Am. **68,** 139–140 (1978).
4. B. R. Hunt, "Matrix formulation of the reconstruction of phase values from phase differences," J. Opt. Soc. Am. **69,** 393–399 (1979).
5. H. Takajo and T. Takahashi, "Least-squares phase estimation from phase differences," J. Opt. Soc. Am. A **5,** 416–425 (1988).
6. H. T. Takajo and T. Takahashi, "Noniterative method for obtaining the exact solution for the normal equation in least-squares phase estimation from the phase difference," J. Opt. Soc. Am. A **5,** 1818–1827 (1988).
7. D. C. Ghiglia and G. A. Mastin, "Two-dimensional phase correction of synthetic-aperture-radar imagery," Opt. Lett. **14,** 1104–1106 (1989).
8. D. C. Ghiglia and L. A. Romero, "Direct phase estimation from phase differences using fast elliptic partial differential equation solvers," Opt. Lett. **14,** 1107–1109 (1989).
9. B. L. Busbee, G. H. Gollub, and C. W. Nielson, "On direct methods for solving Poisson's equations," SIAM J. Numer. Anal. **7,** 627–656 (1970).
10. D. C. Ghiglia, G. A. Mastin, and L. A. Romero, "Cellular automata method for phase unwrapping," J. Opt. Soc. Am. A **4,** 267–280 (1987).
11. R. M. Goldstein, H. A. Zebker, and C. L. Werner, "Satellite radar interferometry: Two-dimensional phase unwrapping," Radio Sci. **23,** 713–720 (1988).
12. N. H. Ching, D. Rosenfeld, and M. Braun, "Two-dimensional phase unwrapping using a minimum spanning tree algorithm," IEEE Trans. Image Process. **1,** 355–365 (1992).
13. J. S. Lim, "The discrete cosine transform," in *Two-Dimensional Signal and Image Processing* (Prentice-Hall, Englewood Cliffs, N.J., 1990), pp. 148–157.
14. G. H. Gollub and C. F. Van Loan, "Iterative methods for linear systems," in *Matrix Computations,* 2nd ed. (Johns Hopkins U. Press, Baltimore, Md., 1990), pp. 516–538.
15. COSFT must be used in the forward mode in all cases. In addition, the technique mentioned on p. 649 of Ref. 16 for solving Poisson's equation with Neumann boundary conditions by using COSFT alone is not correct and does not work. COSFT does not specify the boundary conditions exactly [as defined by Eq. (14)] and therefore does not solve the least-squares problem.
16. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes: The Art of Scientific Computing* (Cambridge U. Press, Cambridge, 1986).
17. G. A Mastin and D. C. Ghiglia, "A research-oriented spotlight synthetic aperture radar polar reformatter," Publ. SAND90-1793 (Sandia National Laboratories, Albuquerque, N.M., 1990), pp. 23–27.