STATISTICAL-COST NETWORK-FLOW APPROACHES TO TWO-DIMENSIONAL PHASE UNWRAPPING FOR RADAR INTERFEROMETRY

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Curtis W. Chen
June 2001

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Howard A. Zebker
(Principal Adviser)

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

G. Leonard Tyler

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Dwight G. Nishimura

Approved for the University Committee on Graduate Studies:

Abstract

Two-dimensional phase unwrapping is a key step, and often the most significant error source, in the analysis of synthetic-aperture-radar interferograms. In the interferometric technique, very accurate measurements of the Earth's topography or its surface deformation are derived from radar-image phase data. Phase, however, is defined only modulo 2π rad, so a resulting 2-D array of measurements is wrapped with respect to some modulus or ambiguity. These data must be unwrapped to provide meaningful information. For this purpose, we introduce a new, nonlinear constrained-optimization approach in which i) defined cost functions map particular unwrapped solutions to scalar costs and ii) a solver routine computes minimum-cost solutions. Previous efforts have focused mainly on simple cost functions that have yielded efficient—but not necessarily accurate—algorithms. These inaccuracies seriously degrade the effectiveness of the interferometric technique and can preclude useful geophysical interpretation of the data. We propose a new set of nonconvex, statistically based cost functions through which we treat phase unwrapping as a maximum *a posteriori* probability estimation problem. That is, we derive approximate, application-specific statistical models for the problem variables. Based on these models, we cast phase unwrapping as an optimization problem whose objective is to find the most physically probable unwrapped solution given the observable quantities: wrapped phase, image intensity, and interferogram coherence. We prove that the resulting problem is *NP*-hard, and we develop nonlinear network-flow solver techniques for approximating solutions to this problem. Extending our statistical framework and network methods, we also present a tiling heuristic for applying our algorithm to large data sets. Performance tests on topographic and deformation data acquired by the ERS-1 and ERS-2 satellites suggest that our algorithm yields superior accuracy and competitive efficiency as compared to other existing algorithms.

Acknowledgements

Many people have helped make this dissertation what it is; I'd like to express my deepest and most sincere thanks to all who have had a hand in the graduate study concluded by the following pages.

I am especially grateful to the Stanford faculty who supervised my academic research. Professor Howard Zebker served as my principal adviser in this effort, and his mentorship and support have been foundational to the work summarized here. Professor Len Tyler acted as my associate adviser, and I have learned much from him as well. Likewise, Professors Dwight Nishimura and Dawson Engler offered me fresh insights and new perspectives on a number of issues while serving on my reading and oral-defense committees. Professor Umran Inan provided me with invaluable guidance during my undergraduate and early graduate years.

My gratitude goes also to the students and postdocs with whom I have had the good fortune of working: Falk Amelung, Margrit Gelautz, Ramon Hanssen, Leif Harcke, Weber Hoen, Jörn Hoffmann, Sigurjón Jónsson, Jack Yi Liang, Mamta Sinha, and Haibin Xu. These individuals have, among other things, shared with me their insights in technical discussions, shared with me their insights in not-so-technical discussions, assisted me in logistical matters, allowed me the use of their data, helped me with data processing, and tutored me in the black art of running ROI. Moses Charikar gave me helpful feedback on computational-complexity issues, and Bevan Baas aided with typesetting. I am grateful to all of these people, and to the rest STARLab, for contributing to an environment which I have found both enjoyable and rewarding.

I am further grateful for the data, software, and funding I have received. The European Space Agency provided the radar data of the Death Valley and Hector Mine test areas. The data and DEM for the Alaska test areas were provided by the Alaska SAR Facility; Rick Guritz, Mark Ayers, Tom Logan, and Rudi Gens of ASF helped with data processing. The

software implementations of the MST and DCC algorithms draw upon ideas from the SPLIB shortest-path codes written by Boris Cherkassky, Andrew Goldberg, and Tomasz Radzik. The MCF algorithm implementation is based on the CS2 solver written by Goldberg and Cherkassky and copyrighted by IG Systems, Inc. The residue-cut algorithm implementation was written by Richard Goldstein, Howard Zebker, and Charles Werner; the PCG and LPN algorithm implementations were written by Dennis Ghiglia and Mark Pritt and are copyrighted by John Wiley and Sons, Inc. This research was funded by NASA and the University of Alaska Geophysical Institute.

Issues of funding, software, and data would have been irrelevant had it not been for the encouragement and support of those close to me, however. I can consequently do no less than extend my heartfelt thanks to Niroshana, for her warmth, cheer, and love; to Adam, for his understanding and his sharp wit; to Peter, for his spirit and optimism; and, finally, to my mother and father, for everything.

Contents

List of Tables

List of Figures

List of Symbols

1-D	one dimensional.
2-D	two dimensional.
3-D	three dimensional.
A	ground surface area of imaged pixel (m^2).
a	arbitrary arc.
$B_\perp$	perpendicular component of interferometer baseline (m).
C	arbitrary constant.
c	arc residual cost.
d	network distance.
d_r	range component of surface displacement (m).
DCC	dynamic-cost cycle canceling.
DEM	digital elevation model.
$E[\cdot]$	expected value operator.
ERS-1	European Space Agency Earth Remote Sensing satellite 1.
ERS-2	European Space Agency Earth Remote Sensing satellite 2.
$f(\cdot)$	probability density or mass function.
$G(\cdot)$	optimization objective function.
$g(\cdot)$	phase-difference cost function.
g_d	cost of shear shelf in deformation cost function.
g_{lay}	cost of layover shelf in topographic cost function.
$g_{\text{lay}}^{(a)}$	cost of layover shelf in topographic azimuth cost function.
$g_{\text{lay}}^{(r)}$	cost of layover shelf in topographic range cost function.
I	SAR image intensity.
i	arbitrary integer.
j	$\sqrt{-1}$; arbitrary integer.

K	arbitrary constant integer.
k	arbitrary integer.
k_{ds}	ratio of diffuse to specular scattering.
L^p	optimization norm.
l_0	length of optimal L^0 flow.
l_R	length of subset of optimal L^0 flow.
l_{RST}	length of optimal RST.
LPN	minimum L^p-norm algorithm.
M	number of rows in a 2-D array.
m	arbitrary integer.
MAP	maximum *a posteriori* probability.
MCF	minimum cost flow.
MR	magnetic resonance.
MST	minimum spanning tree.
N	number of columns in a 2-D array.
n	scattering model parameter; arbitrary integer.
N_c	number of complex looks.
N_i	equivalent number of independent looks.
NP-complete	problem complexity class.
NP-hard	problem complexity class.
p	L^p-norm exponent parameter; arbitrary network node.
PDF	probability density function.
q	arbitrary network node.
R	subset of optimal L^0 flow.
r	range (m).
R_r	slant range resolution (m).
RST	rectilinear Steiner tree.
S	set of network nodes.
s	complex pixel value of SAR image.
SAR	synthetic aperture radar.
SNAPHU	statistical-cost, network-flow algorithm for phase unwrapping.
T	network tree.
u	arbitrary network node.

USGS	United States Geological Survey.
w	phase difference weight.
x	across-track ground coordinate (m).
y	along-track ground coordinate (m).
$y_{\max}$	greatest coordinate in y of arcs a_y.
$y_{\min}$	least coordinate in y of arcs a_y.
Z	set of all real integers.
z	elevation (m).
δ	flow increment.
Δr	SAR sample spacing in range (m).
Δx	ground projection of SAR range sample spacing (m).
Δy	ground projection of SAR azimuth sample spacing (m).
Δz	local topographic slope.
$\Delta z^{(a)}$	azimuth component of local topographic slope.
$\Delta z^{(r)}$	range component of local topographic slope.
$\Delta z_0^{(r)}$	range slope for shadow condition.
$\Delta z_I^{(r)}$	range slope estimated from intensity.
Δz_{lay}	maximum slope expected from layover given intensity.
$\Delta\Phi$	array or vector of unwrapped phase differences (rad).
$\Delta\phi$	unwrapped phase difference between neighboring samples (rad).
$\Delta\phi^{(a)}$	unwrapped phase difference in azimuth (rad).
$\Delta\phi^{(r)}$	unwrapped phase difference in range (rad).
$\Delta\phi_0^{(r)}$	topographic unwrapped phase gradient for shadow condition (rad).
$\Delta\phi_{\text{back}}$	topographic unwrapped gradient of layover back slope (rad).
$\Delta\phi_{\text{crit}}$	unwrapped gradient at discontinuity regime transition point (rad).
$\Delta\phi_{\text{defo}}$	deformation component of unwrapped phase difference (rad).
$\Delta\phi_I$	topographic unwrapped gradient estimated from intensity (rad).
$\Delta\phi_{\text{lay}}$	layover cutoff of topographic unwrapped gradient PDF (rad).
$\Delta\phi_{\max}$	upper cutoff of topographic unwrapped gradient PDF (rad).
$\Delta\phi_{\text{noise}}$	noise component of unwrapped phase difference (rad).
$\Delta\phi_{\text{signal}}$	signal component of unwrapped phase difference (rad).
$\Delta\phi_{\text{topo}}$	topography component of unwrapped phase difference (rad).
$\Delta\phi_{\text{topo}}^{(a)}$	topographic part of unwrapped gradient in azimuth (rad).

$\Delta\phi^{(r)}_{\text{topo}}$	topographic part of unwrapped gradient in range (rad).
$\Delta\phi_\rho$	correlation cutoff of topographic unwrapped gradient PDF (rad).
$\Delta\Psi$	array or vector of wrapped phase differences (rad).
$\Delta\psi$	wrapped phase difference between neighboring samples (rad).
Θ	arbitrary cycle in network.
θ	SAR look angle with respect to nadir (rad).
θ_i	local incidence angle (rad).
Λ	network path.
λ	SAR signal wavelength (m).
π	ratio of circumference of circle to its diameter.
π_p	linear network potential of node p.
π^{in}	inward potential of node on nonlinear network.
π^{out}	outward potential of node on nonlinear network.
ρ	interferogram complex correlation coefficient.
$\hat{\rho}$	estimate of interferogram complex correlation coefficient.
ρ_s	spatial decorrelation factor.
ρ_{min}	correlation threshold for detecting deformation discontinuity.
ρ_{other}	factor accounting for nonspatial sources of decorrelation.
σ^0	normalized radar cross section.
σ_{lay}	standard deviation of phase-parameter uncertainty from layover (rad).
σ_{meas}	standard deviation of phase-parameter measurement error (rad).
$\sigma_{\Delta\phi_{\text{noise}}}$	standard deviation of unwrapped phase-difference noise (rad).
$\sigma_{\psi_{\text{noise}}}$	standard deviation of wrapped phase noise (rad).
Υ	arbitrary scalar potential.
Φ	2-D unwrapped phase field or array (rad).
$\hat{\Phi}$	estimate of 2-D unwrapped phase field or array (rad).
ϕ	unwrapped phase value (rad).
ϕ_{defo}	deformation component of unwrapped phase value (rad).
ϕ_{noise}	noise component of unwrapped phase value (rad).
ϕ_{signal}	signal component of unwrapped phase value (rad).
ϕ_{topo}	topography component of unwrapped phase value (rad).
χ	arc flow.
Ψ	2-D wrapped phase field or array (rad).

ψ	wrapped phase value (rad).
ψ_{noise}	noise component of wrapped phase (rad).

Chapter 1

Introduction

Two-dimensional (2-D) phase unwrapping is a key data-processing step in many applications of synthetic aperture radar (SAR) interferometry. As interferometric radar applications have found widespread use in recent years, the phase unwrapping problem has emerged as a research topic of great importance for the remote sensing community. Despite no small amount of effort, however, the problem remains unresolved.

Driven by these motivations, we examine in this dissertation both theoretical aspects of the phase unwrapping problem as well as practical algorithms for its solution. Focusing our attention on the unique needs of SAR interferometry, we anchor our investigation in the specific context of this application. We thus begin with a definition of the problem and a brief explanation of the circumstances in which it arises.

1.1 Problem Definition: 2-D Phase Unwrapping

Two-dimensional phase unwrapping is the process of recovering unambiguous phase values from a 2-D array of phase values known only modulo 2π rad. This problem arises when, as in many applications, phase is used as a proxy indicator of a physical quantity, such as time delay, from which other information can be inferred. Since phase is observable only on a circular, repeating space, measurements derived from phase data are wrapped with respect to some modulus or ambiguity that is the physical equivalent of 2π rad. The observed data must therefore be unwrapped, or mapped back into the full range of real numbers, if meaningful results are to be obtained.

1.2 Motivation: Synthetic Aperture Radar Interferometry

Current interest in 2-D phase unwrapping has been motivated largely by the advent of SAR interferometry. In this technique, multiple coherent radar images of the earth are combined to form interferograms, and, depending on how the data are processed, the interferometric phase can be used to make extremely fine measurements of surface topography, deformation, or velocity, as explained below. There are many geophysical applications for such data, but the data must usually be unwrapped before they can be made useful. In fact, incorrect phase unwrapping is often the most significant source of error in interferometric SAR measurements. For this reason, the main goal of our research is the development of an accurate, efficient, and robust phase unwrapping algorithm that is applicable to the specifics of SAR interferometry.

SAR interferometry is an extension of classical radar techniques that date back to the first half of the twentieth century. In such radar techniques, the range to a target is determined through a measurement of the time delay between the transmission of a signal and the reception of the signal's echo from the target (see the text by *Skolnik* [1980], for example). A SAR system is an instrument, usually aboard an aircraft or a spacecraft, that forms high-resolution images of the earth by examining a series of radar echoes from the surface. Typically, the SAR system is designed with a side-looking geometry so that as the platform moves, the area scanned by the radar's antenna beam sweeps out a swath on the ground (see Fig. 1.1). This swath comprises the imaged area. The brightness of each pixel in the image is related to the radar reflectivity of the ground surface at that location. Resolution in the direction perpendicular to the flight track is obtained through knowledge of the time delay, and hence the range, of every sample of each pulse return. Resolution in the along-track direction is obtained through aperture synthesis techniques whereby echoes from multiple locations along the flight track are linearly combined in order to form, effectively, a large antenna aperture. A radar image is thus rendered with respect to coordinates in range and azimuth, the across-track and along-track directions, respectively; surface features appearing in the image are projected into this coordinate system. Note that as a consequence of this projection, radar images often exhibit distortion phenomena known as foreshortening and layover in areas of significant surface topography. More detail about these and other topics related to SAR imaging can be found in texts such as the one by *Curlander and McDonough* [1991].

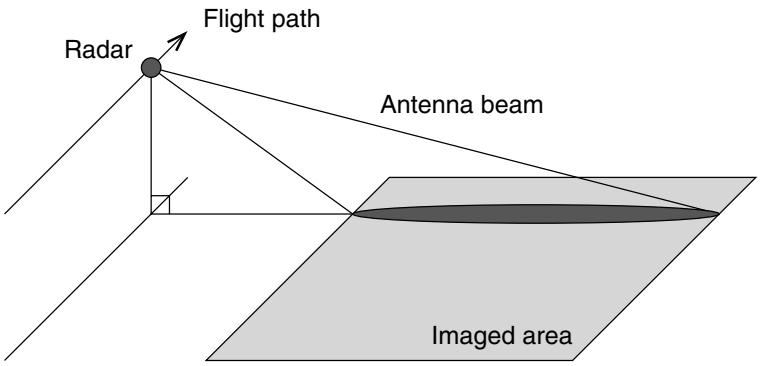

Figure 1.1 SAR imaging geometry. SAR systems typically employ a side-looking geometry as shown here. Features on the ground are projected into a range-azimuth coordinate system in the radar image.

Since radar signals are coherent and narrowband, the amplitude and phase of each image pixel derive from the random complex superposition of the echoes from many individual ground scatterers within the resolution element. Therefore, the amplitude values of a SAR image are usually characterized by speckle, and the phase values are usually unrelated to one another. However, the relative phase of a pixel in one SAR image may be related to the relative phase of the corresponding pixel in another SAR image of the same area. Thus, if the two images are acquired from appropriate viewing geometries, the differences in phase between corresponding pixels may serve as a very precise indicator of range variations or signal path-length differences between the two image acquisitions.

Consider the special case in which two individual SAR images are obtained with identical viewing geometries, and the ground surface remains completely unchanged between the times that the images are acquired. The two images will be the same, thermal noise notwithstanding, so the differences in phase between corresponding pixels of the two images will be zero. Suppose now that the viewing geometries are slightly different, so that the images are acquired from locations separated by a known, nonzero baseline as shown in Fig. 1.2. Since the observed phase differences are related to the path length differences $r_1 - r_2$, the phase differences can be used to measure the surface topography very precisely [*Zebker and Goldstein,* 1986; *Zebker et al.,* 1994a]. Consider next the case in which the viewing geometries of the two images are identical, but the surface deforms during the time between image acquisitions. Instead of topography, the observed phase differences can be

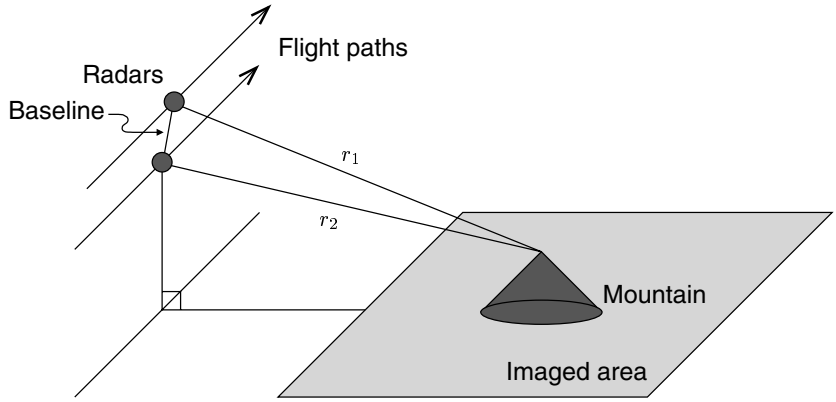

Figure 1.2 Viewing geometry for SAR interferometry. Phase differences between the two complex images are related to the range differences $r_1 - r_2$. If the viewing locations have a known, nonzero baseline as shown, the interferometric phase signature contains variations due to the relative surface elevation. The phase signature may also contain variations due to relative line-of-sight surface deformation during the time between image acquisitions.

used to measure the line-of-sight components of relative surface displacement [*Gabriel et al.,* 1989; *Massonnet et al.,* 1993; *Zebker et al.,* 1994b] or velocity [*Goldstein and Zebker,* 1987; *Goldstein et al.,* 1993].

These types of measurements—ones derived from phase differences between images—are the subject of SAR interferometry. An interferogram formed from two complex 2-D images has phase values that are equal to the differences in phase between corresponding pixels of the individual images:

$$s_1 s_2^* = |s_1||s_2| \exp\left(j\left(\phi_1 - \phi_2\right)\right). \tag{1.1}$$

Here, s_1 and s_2 are the complex values of corresponding pixels from the two images, "*" denotes complex conjugation, $j = \sqrt{-1}$, and ϕ_1 and ϕ_2 are the phases of s_1 and s_2. Thus, by definition, the phase of the interferogram can only be observed modulo 2π rad. An example topographic interferogram of the San Francisco Bay Area is shown in Fig. 1.3. As is evident from the wrapping of color fringes in the phase image, the phase data must be unwrapped before useful topographic information can be obtained.

For interferometry to be effective, the relative positions of individual scatterers within a resolution element cannot change much between image acquisitions. If they do, as from

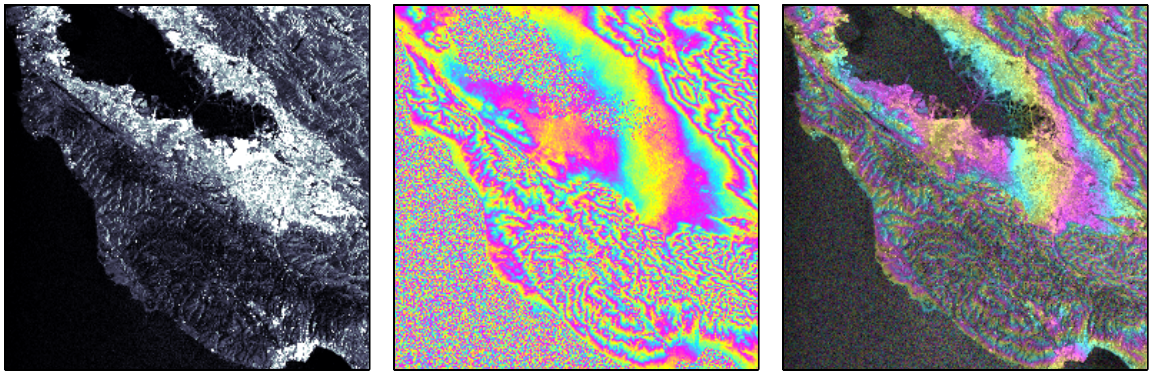

Figure 1.3 Topographic interferogram of the San Francisco Bay Area: (a) the interferogram magnitude, which is related to the radar reflectivity of the earth surface; (b) the interferogram phase, which in this case represents surface topography, wrapped; (c) a combined image showing the magnitude in gray-scale brightness and the wrapped phase in color. The phase is meaningless in areas covered by water because of temporal decorrelation effects.

geometric or temporal effects for example, the phases of the individual images decorrelate, making the interferogram more noisy [*Zebker and Villasenor,* 1992]. Such phase-noise effects can be reduced in some cases through spatial averaging, a process also called 'taking looks,' but in locations where the images are completely decorrelated, the interferometric phase simply does not contain meaningful information. Together with spatial undersampling of the true signals under observation, decorrelation makes phase unwrapping nontrivial, and sometimes exceedingly difficult, for most SAR interferograms of any interest.

1.3 Other Applications

SAR interferometry is not the only application in which phase unwrapping is required. Some of the earliest work on 2-D phase unwrapping was carried out in the context of wavefront distortion estimation for adaptive-optics (compensated-imaging) systems [*Fried,* 1977; *Hudgin,* 1977]. In this application, phase measurements from a 2-D sensor array are unwrapped to provide estimates of atmospheric turbulence effects on an optical imaging system. These atmospheric distortions are then removed through the use of a deformable focusing mirror.

Phase unwrapping can be important in magnetic resonance (MR) imaging as well. Depending on the application and how the data are acquired, phase measurements from 2-D or 3-D MR images can be used for such purposes as estimating blood flow rates [*Herment et al.,* 2000] or separating water and fat signals [*Glover and Schneider,* 1991; *Hedley and Rosenfeld,* 1992].

In each of these cases, the phase unwrapping problem is similar on a fundamental level to the phase unwrapping problem posed by SAR interferometry. The latter calls for markedly different unwrapping algorithms, however. This is because the phase characteristics of SAR interferograms are usually distinct from those observed in other applications (see Chapter 4). Consequently, we focus our attention here on phase unwrapping algorithms for applications of SAR interferometry. By thus narrowing the scope of the problem, we are able to attain more accurate results. Of course, the methodology of our approach may well apply elsewhere, but we leave the exploration of that topic for future work.

1.4 Synopsis of Research

In Chapter 2, we provide a brief overview of existing phase unwrapping methods and ideas. Of these ideas, one of the most important is the notion that phase unwrapping can be treated as a constrained optimization problem. In Chapter 3, we introduce the two main issues raised by the optimization approach: setting up the problem and solving it. We examine these issues in Chapters 4 and 5. In Chapter 4, we pose phase unwrapping as a maximum *a posteriori* probability (MAP) estimation problem, and we derive models for approximating the joint statistics of interferometric SAR data. We then incorporate these statistics into nonlinear cost functions, thereby setting up the optimization problem. To solve the problem, albeit only approximately, we develop in Chapter 5 efficient nonlinear optimization routines based on network-flow techniques. Our algorithm, SNAPHU, results from the ideas of those two chapters. In Chapter 6, we develop a tiling heuristic for efficiently applying the algorithm to large data sets. We demonstrate the algorithm's performance on interferometric data sets in Chapter 7. We then conclude this work in Chapter 8 with comments regarding the implications of our research and suggestions for future work.

1.5 Contributions

The contributions of this research are summarized as follows:

1. Proof of the intractability of the general phase unwrapping optimization problem. The proof is based on the ideas of *NP*-hardness.

2. Suggestion of a MAP framework that provides a physical basis for the phase unwrapping optimization problem. The framework generalizes existing L^p-norm approaches.

3. Derivation of models for approximating the joint statistics of interferometric SAR data, as required by the MAP framework, for the cases of both topography and deformation measurement. The models are quantified through the use of nonlinear cost functions.

4. Design and implementation of efficient, nonlinear network-flow solvers for use with the statistical cost functions described above.

5. Development of a tiling heuristic for efficiently applying the above algorithm to large data sets.

6. Evaluation of the performance of the above approach, as compared to existing techniques, on interferometric SAR data.

Chapter 2

Background

Many varied approaches to 2-D phase unwrapping have been proposed over the past several years, but only a limited number are currently in common use. In this chapter, we review the most popular of existing phase unwrapping algorithms, and by examining how they are related to one another, we identify several fundamental concepts upon which the following chapters rely.

2.1 Assumptions and General Approach

Strictly, phase unwrapping is an impossible problem because an unwrapped phase array necessarily contains information that is not available in the corresponding wrapped array. That is, given only an ambiguous wrapped phase array, there is no definitive way to determine which of the many possible unambiguous unwrapped solutions is correct. All algorithms therefore rely on at least some assumptions, the most basic and most common of which is that the Nyquist criterion is met throughout most—but not necessarily all—of the scene. That is, the spatial sampling rate is assumed to be high enough in most parts of the complex interferogram that aliasing is avoided. Thus, the true unwrapped phase values of neighboring pixels may be assumed to lie within one-half cycle (π rad) of each other almost everywhere.

Consider the simple one-dimensional (1-D) wrapped phase example of Fig. 2.1(a), in which the numbers represent phase values in cycles. Based on the regularity of the data, we might reasonably surmise that the wrapped phase values are derived from the smooth ramp of unwrapped values in Fig. 2.1(b). Of course, any other solution obtained through the

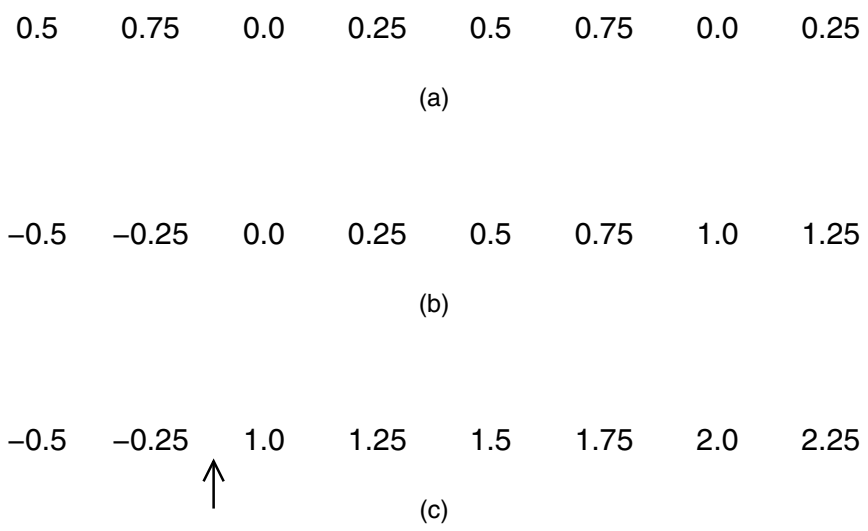

Figure 2.1 One-dimensional phase unwrapping example: (a) wrapped phase values modulo one cycle; (b) unwrapped values obtained under the assumption of adequate (Nyquist) sampling; (c) a possible unwrapped solution that violates the assumption of adequate sampling. The unwrapped phase difference at the location indicated by the arrow is outside the $[-0.5, 0.5)$ cycle interval.

addition of a constant, integer-cycle phase offset to these values is equally plausible, but the determination of the exact offset is usually not regarded as part of the phase unwrapping problem; phase unwrapping is the estimation of unwrapped phase values relative only to each other.

Our unwrapped solution of Fig. 2.1(b) is inferred through the implicit assumption of adequate spatial sampling. For example, we assume that the true unwrapped phase difference, or discrete derivative, between the second and third wrapped values (0.75 and 0.0) is actually $+0.25$ rather than -0.75 cycles. Similarly, we assume that all true neighboring-pixel phase differences are between -0.5 and $+0.5$ cycles, or between $-\pi$ and π rad. By integrating these estimated phase differences (derivatives) from pixel to pixel, we obtain the unwrapped estimate.

The key to phase unwrapping thus lies not in directly estimating the unwrapped phase values themselves, but in estimating the unwrapped phase *differences* between them. This illustrates the fundamental premise of phase unwrapping—that under appropriate conditions, an accurate estimate of the unwrapped solution can be determined from the relationships between neighboring phase samples.

Let us define a wrapped phase difference as the difference in phase, wrapped into the interval $[-\pi, \pi)$, between two neighboring samples in the data. The notion of adequate sampling therefore reduces to the assumption that, locally, the true unwrapped phase differences are equal to the observed wrapped phase differences.

When this assumption fails, however, the 1-D unwrapping strategy above results in serious errors. Suppose, for example, that the true unwrapped phase difference between the second and third samples of Fig. 2.1 is actually 1.25 cycles, so that the true unwrapped solution is as shown in Fig. 2.1(c). The assumption of adequate sampling would then be violated, and the unwrapped estimate of Fig. 2.1(b) would be incorrect. The error, moreover, would not be locally confined since the solution relies on the integration of phase differences; all phase values on either side of the underestimated difference would be incorrect by one cycle relative to the values on the other side. Thus, phase unwrapping errors often manifest themselves as incorrect, integer-cycle relative phase offsets between different sections or patches of the solution.

With 1-D data, little can be done to avoid these undersampling errors. For 2-D data, however, neighboring-pixel phase differences in the row and column directions are not independent, and their relationships can be exploited to avoid certain types of unwrapping errors. In keeping with the literature, we use the term "gradient" to describe such neighboring-pixel phase differences in the 2-D case as well as the 1-D case. That is, we will sometimes use the term "gradient" to describe scalar, discrete directional derivatives in the 2-D case, even though the term more often refers to a vector quantity. The use of this terminology should be clear in context, however.

Most 2-D phase unwrapping approaches involve the assumption that the data are sampled adequately in most, but not necessarily all, parts of the scene. Where the data are indeed sampled adequately, the unwrapped phase gradients equal the observed wrapped gradients, and an unwrapped estimate may be readily obtained through the integration of these estimated gradients. The difficulty of phase unwrapping, of course, stems from the fact that most interesting interferograms contain gradients that exceed one-half cycle in magnitude, and their incorrect integration leads to global errors. Such gradients arise because of both decorrelation (*i.e.*, noise) and true spatial variations of the signals under observation. In either case, we call such gradients—those outside of the $[-\pi, \pi)$ interval—discontinuities. The task of a 2-D phase unwrapping algorithm can be reduced to locating and accommodating these discontinuities.

2.2 Residue-Cut Algorithm

Following the 1-D example in the previous section, a very simple 2-D phase unwrapping strategy would be to integrate the estimated unwrapped-phase gradients along some path through the 2-D array. The choice of integration paths requires particular attention, however.

For most interferometric SAR applications, the unwrapped phase represents a well-behaved physical quantity, such as topographic height or surface displacement, that can be expressed mathematically as a 2-D potential or scalar function. Such functions follow the vector identity [*Thomas and Finney,* 1988]

$$\nabla \times \nabla \Upsilon \equiv 0. \tag{2.1}$$

Equation (2.1) states that the curl of the vector gradient of any scalar potential Υ is identically zero. The vector gradient of an unwrapped phase field Φ is therefore a conservative vector field, through which any closed loop integral is necessarily zero. An unwrapped phase field is thus completely specified, except for an additive constant, by its unwrapped gradients; the gradients, if they are known, can be integrated along an arbitrary path to recover the unwrapped phase field.

When only wrapped data are available, however, gradients that are truly outside the $[-\pi, \pi)$ interval are observable only after having been wrapped into it. Therefore, as pointed out by *Goldstein et al.* [1988], closed loop integrals of wrapped gradients can give nonzero results, and wrapped gradient fields are consequently nonconservative. Thus, if unwrapped phase values are estimated by integrating the wrapped rather than the unwrapped gradients, the resulting phase-field estimate does depend upon the choice of integration paths in the 2-D plane. In other words, some sets of observed wrapped gradients may not be self consistent given the assumption that the underlying phase field represents a physical surface. This is illustrated by the 2×2 set of wrapped phase values in Fig. 2.2. Because the data are not conservative, different unwrapped solutions are obtained for different integration paths. Moreover, the closed loop integral shown gives a nonzero result called a 'residue,' and the data are hence analogous to the physically unrealizable staircase of the sketch by M. C. Escher shown in Fig. 2.3.

In order to avoid such inconsistencies, *Goldstein et al.* [1988] proposed a method for estimating a conservative set of unwrapped gradients from the observed wrapped gradients.

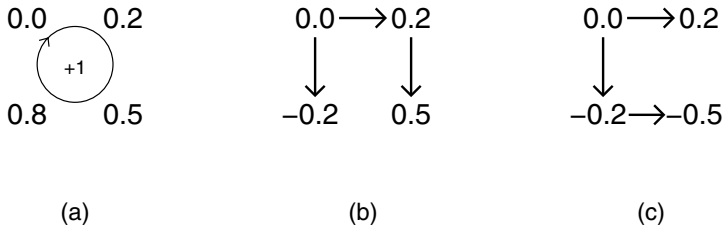

Figure 2.2 A 2×2 example array of wrapped phase values that is inconsistent with the assumption of adequate sampling. The clockwise loop integral of wrapped gradients indicated in (a) has a nonzero (+1) result. The unwrapped solutions in (b) and (c) from simple wrapped-gradient integration depend on the choice of integration path (shown as arrows). The lower-right values of the two unwrapped solutions differ.

In this approach, a clockwise loop integral of wrapped phase gradients is calculated for each 2×2 square of neighboring pixels in the input array (see Fig. 2.4). Because the wrapped gradients are usually equal to the unwrapped gradients, these loop integrals evaluate to zero throughout most of the array. In some places, however, residues of $+1$ or -1 cycle arise. At such locations, the data are known to be inadequately sampled. Whether because of decorrelation or high-spatial-frequency variations in the true unwrapped signal, nonzero residues thus indicate local inconsistencies with the assumption that the unwrapped gradients equal their wrapped counterparts. *Goldstein et al.* [1988] pointed out that any closed loop integral of wrapped gradients has a nonzero result if and only if it encloses unequal numbers of positive and negative residues. Therefore, for a given wrapped phase array, integration paths that do not enclose unbalanced residues give self-consistent solutions.

The residue-cut algorithm therefore entails the computation of an unwrapped solution from some integration path that does not encircle unbalanced residues. To find such a path, 'cuts' are designated between residues. These cuts are defined as barriers over which the phase gradients may not be integrated. They are arranged in tree-like structures such that each tree spans equal numbers of positive and negative residues (*i.e.*, there are the same number of positive and negative residues on each tree). Thus, because integration paths are disallowed from crossing any cuts, no integration path can encircle unbalanced residues. Wrapped gradients along allowable paths may then be safely assumed to lie within the $[-\pi, \pi)$ interval. In other words, phase gradients are allowed to exceed one-half cycle in magnitude only at locations of cuts; by avoiding cuts, the remaining phase gradients can

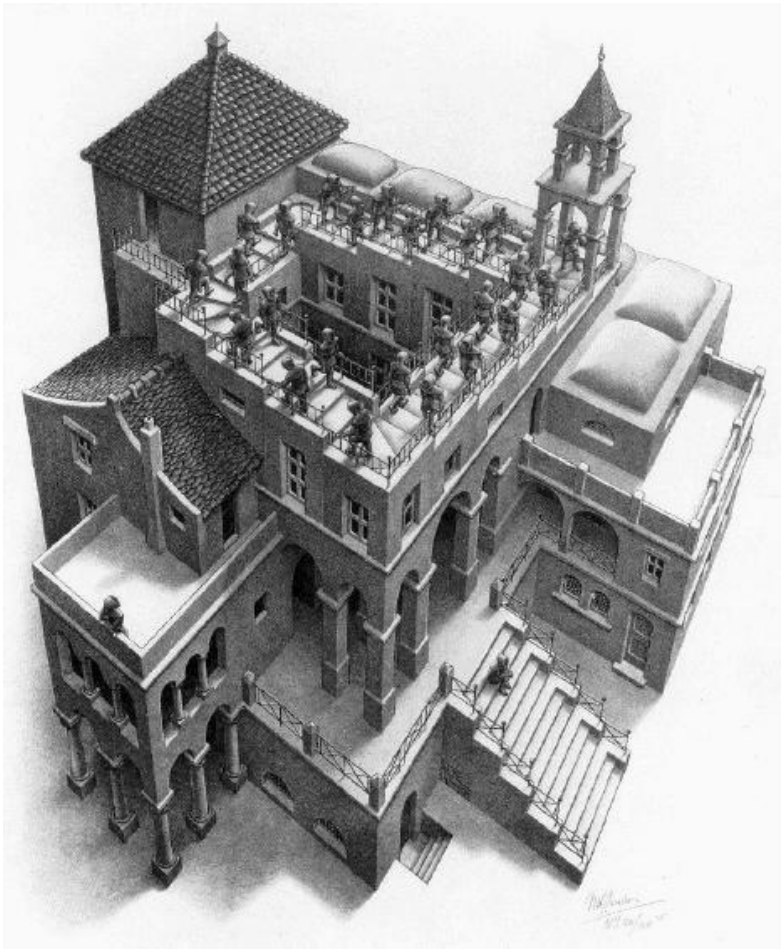

Figure 2.3 "Ascending and Descending," by M. C. Escher. The physically unrealizable staircase represents a nonconservative gradient field.

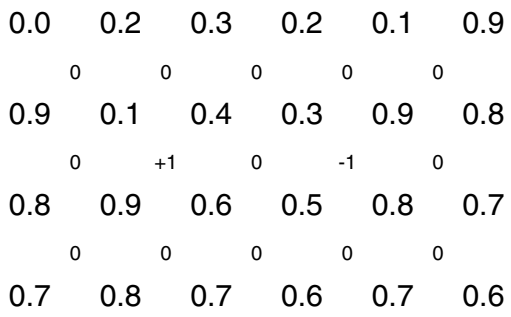

Figure 2.4 Residues resulting from an example wrapped phase array. The large numbers represent wrapped phase values, in cycles, and the smaller numbers are the results of the clockwise wrapped-gradient loop integrals.

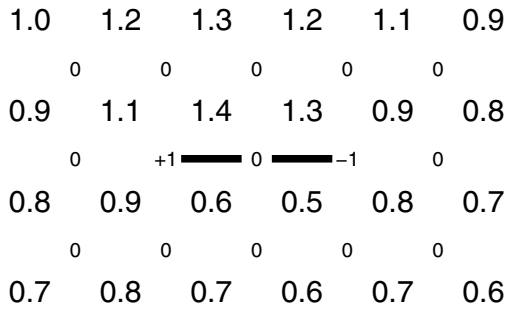

Figure 2.5 An example unwrapped solution from the residue cut algorithm for the wrapped data of Fig. 2.4. The large numbers represent unwrapped phase values, in cycles, while the heavy black lines denote cuts connecting the positive and negative residues. Unwrapped phase gradients that are not crossed by cuts are assumed to be in $[-\pi, \pi)$.

be integrated to determine a self-consistent unwrapped solution (see Fig. 2.5).

Because cuts represent possible locations of phase discontinuities, the residue-cut algorithm involves an attempt at minimizing the total length of cuts in the scene. Although the exact minimum generally cannot be found, as discussed in Chapter 3, the residue-cut algorithm is fast and usually quite accurate. A major drawback of the algorithm, however, is that it does not always provide a complete solution. Since the algorithm produces cuts that sometimes close on themselves in low-coherence, residue-dense parts of the input interferogram, large areas of the 2-D array may be closed off from other areas with no allowable integration paths connecting them. An unwrapped solution may thus contain holes.

Cusak et al. [1995] and *Buckland et al.* [1995] have both proposed variations on the residue-cut algorithm that involve different ways of selecting the arrangements of cuts. Collectively, these approaches have in the literature been variously called residue-cut, branch-cut, cut-line, and ghost-line algorithms. Here, we treat them together as versions of the same basic residue-cut approach.

Note that this approach assumes that corresponding unwrapped and wrapped phase values may differ only by integer numbers of cycles. This property, called congruence, is an *a priori* assumption in the algorithm's formulation of the phase unwrapping problem. Not all algorithms assume congruence, however.

2.3 Least-Squares Algorithms

Like the residue-cut algorithm, least-squares algorithms are based on the assumption that the observed wrapped phase field is adequately sampled nearly everywhere. Rather than integrate gradient estimates along explicit paths, however, algorithms in the least-squares family compute unwrapped solutions that minimize the total squared departure of the estimated unwrapped gradients from their wrapped counterparts. Thus, given a wrapped phase field Ψ, a least-squares algorithm produces an unwrapped estimate $\hat{\Phi}$ subject to the following optimization objective:

$$\text{minimize} \left\{ \sum_{i,j} w_{i,j}^{(x)} \left(\Delta\phi_{i,j}^{(x)} - \Delta\psi_{i,j}^{(x)} \right)^2 + \sum_{i,j} w_{i,j}^{(y)} \left(\Delta\phi_{i,j}^{(y)} - \Delta\psi_{i,j}^{(y)} \right)^2 \right\}. \tag{2.2}$$

Here, $\Delta\phi^{(x)}$ and $\Delta\psi^{(x)}$ are the x components of the unwrapped and wrapped phase gradients, respectively, and $\Delta\phi^{(y)}$ and $\Delta\psi^{(y)}$ are their y direction counterparts. User-defined

weights w can be assigned to each gradient, and the summations include all appropriate rows i and columns j. When uniform weights are applied to the data (*i.e.*, $w = 1$ for all gradients), the problem is said to be unweighted.

Drawing on work done by *Fried* [1977], *Hudgin* [1977], and *Hunt* [1979], *Ghiglia and Romero* [1994] recognized that the least-squares estimate $\hat{\Phi}$ for a wrapped phase field Ψ can be found by solving a version of Poisson's equation:

$$\nabla^2 \hat{\Phi} = \hat{\nabla}^2 \Psi. \tag{2.3}$$

The $\hat{\nabla}^2$ operator on the right-hand side represents the differentiation of the *wrapped* gradients $\Delta\psi$, which are by definition between $-\pi$ and π rad. That is, for the wrapped data, a wrapped version of the Laplacian operator is used. On the other hand, for the unwrapped phase estimate $\hat{\Phi}$ defined on a discrete rectangular grid, the (ordinary) Laplacian operation denoted by ∇^2 is equivalent to convolution with the kernel

$$\begin{array}{ccc} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{array} .$$

Since the wrapped phase Ψ is known (observed), the right-hand side of Eq. (2.3) can be calculated directly. Then, with Fourier or cosine transforms, Eq. (2.3) can be solved for the unwrapped estimate $\hat{\Phi}$. That is, using the convolution theorem [*Bracewell,* 1986], the kernel above can be deconvolved from $\hat{\nabla}^2 \Psi$ to obtain $\hat{\Phi}$ through division in the spatial-frequency domain. With the use of fast transform routines, the unweighted least-squares technique can thus achieve great computational efficiency in implementation [*Ghiglia and Romero,* 1994].

If nonuniform weights are applied, however, the least-squares problem can no longer be represented by the simple form of Poisson's equation above. For this case, *Ghiglia and Romero* [1994] proposed a preconditioned conjugate gradient (PCG) technique that computes weighted least-squares solutions through iterative use of the unweighted least-squares solver algorithm. The PCG approach is therefore much less efficient than the unweighted approach. It can achieve greater accuracy,[1] though, because weights based on *a priori* information can be used to allow greater disparities between the unwrapped and wrapped

[1] We use the term "accuracy" here somewhat subjectively. The issue of how best to quantify the accuracy of an unwrapped solution is discussed in Chapters 3 and 4.

gradients at locations where the data are believed to be less reliable. Advantageous weighting schemes may be based on such quantities as interferogram coherence or magnitude, and much attention has been given to this topic [*Pritt,* 1996; *Ghiglia and Pritt,* 1998; *Zebker and Lu,* 1998; *Chen and Zebker,* 2000].

Subsequent to the work of *Ghiglia and Romero* [1994], many alternate methods have been proposed for solving the weighted and unweighted least-squares phase unwrapping problems. These proposals include algorithms based on multigrid techniques [*Pritt,* 1996], multiresolution techniques [*Davidson and Bamler,* 1999], Green's functions [*Fornaro et al.,* 1996; *Lyuboshenko and Maître,* 1999], Fourier-domain vector projections [*Costantini et al.,* 1999], and region growing [*Fornaro and Sansosti,* 1999]. In essence, though, these algorithms all use the same least-squares formulation given in Eq. (2.2) for posing the phase unwrapping problem.

In the least-squares problem formulation, no gradients are explicitly disregarded as cuts are in the residue-cut algorithm, so a least-squares unwrapped estimate is usually very smooth spatially, and it is generally not congruent to the wrapped input. That is, the unwrapped phase values produced by most least-squares algorithms are unrestricted in the values they may take; they need not be at integer-cycle offsets from the wrapped phase values. Congruence may be enforced in a processing step performed after optimization [*Ghiglia and Pritt,* 1998], but only at the expense of least-squares optimality.

In either case, when congruence is not explicitly enforced before optimization, the least-squares formulation involves the use of a linear estimator with variables (*i.e.*, the wrapped phase gradients) defined on a wrapping, circular space. *Bamler et al.* [1998] showed that such least-squares approaches consequently produce biased results, underestimating the true unwrapped phase gradients. Empirical observations have confirmed this behavior [*Zebker and Lu,* 1998; *Ghiglia and Pritt,* 1998; *Chen and Zebker,* 2000]. Thus, despite their reasonable efficiency and mathematical elegance, most least-squares algorithms often give disappointing results in practice. They do, however, always generate complete solutions.

2.4 Minimum L^p-norm Framework

The residue-cut and least-squares approaches differ greatly in character with respect to both problem formulation and empirical results. *Ghiglia and Romero* [1996], however, introduced a unifying framework that neatly captures many of the theoretical relationships between

the two. Under this framework, phase unwrapping is treated as an optimization problem; an objective function maps possible unwrapped solutions to scalar values, and the goal is to find the solution that minimizes the value of the objective function. *Ghiglia and Romero* [1996] suggested for the phase unwrapping problem an L^p-norm objective function of the form

$$\text{minimize} \left\{ \sum_{i,j} w_{i,j}^{(x)} \left| \Delta\phi_{i,j}^{(x)} - \Delta\psi_{i,j}^{(x)} \right|^p + \sum_{i,j} w_{i,j}^{(y)} \left| \Delta\phi_{i,j}^{(y)} - \Delta\psi_{i,j}^{(y)} \right|^p \right\}. \tag{2.4}$$

This function is least when the unwrapped and wrapped gradients $\Delta\phi$ and $\Delta\psi$ agree as much as possible, though as described above, they usually cannot be equal everywhere if the unwrapped gradient field is to be self consistent. The parameter p thus determines how differences between $\Delta\phi$ and $\Delta\psi$ are penalized, and user-defined weights w can be assigned to individual row or column gradients. The summations include all appropriate rows i and columns j.

When $p = 2$, Eq. (2.4) reduces to Eq. (2.2). That is, the least-squares problem described in the previous section is simply a special case of the more general minimum L^p-norm problem, or for brevity of notation, the L^p problem. On the other hand, in the limit as p approaches zero (henceforth $p = 0$ or L^0), the objective becomes minimization of the total weighted number of unwrapped gradients that differ from their wrapped counterparts. Thus, the residue-cut algorithm's objective reduces to minimizing the unweighted L^0 norm, and the algorithm is categorized as an L^0 algorithm. This is so even though the residue-cut algorithm does not guarantee the exact L^0 optimality of its solutions.

In proposing the L^p framework, *Ghiglia and Romero* [1996] also developed an algorithm for use with arbitrary L^p norms. However, this algorithm, based on the calculus of variations, finds only local rather than global minima for $p < 1$. The algorithm also requires iterative use of a weighted least-squares solver, and hence doubly iterative use of an unweighted least-squares solver, so it is very intensive computationally. Furthermore, because the algorithm does not assume congruence in its problem formulation, it is prone to the slope-underestimation effects described in Section 2.3.

More important than any actual solver implementation, though, is the L^p framework's ability to link theoretically the objectives of the residue-cut, least-squares, and other algorithms. By expressly casting the phase unwrapping problem as an optimization problem, a fundamental question is brought to light: What is the optimization objective whose use

leads to the most accurate unwrapped solutions? We address this question in Chapters 3 and 4.

2.5 Network Flow Formulation

The L^p framework is very useful for describing and analyzing optimization objectives, but the framework itself does not prescribe specific methods by which these objectives are to be minimized. For that purpose, network theory[2] provides considerable insight into both the minimization problem and algorithms for its solution. Network theory is studied in a wide variety of fields because of its generality and applicability to different types of problems. With its application to the phase unwrapping problem, much is gained from the work done on other topics in which the same ideas apply.

Throughout the history of SAR interferometry, many network ideas have been used in the context of phase unwrapping, either implicitly or explicitly [*Goldstein et al.,* 1988; *Ching et al.,* 1992; *Buckland et al.,* 1995; *Flynn,* 1997]. *Costantini* [1998], however, was the first to propose an explicit network model for the phase unwrapping problem itself. In his network model, the wrapped gradient loop integral around each 2×2 square of pixels (see Section 2.2) is represented by a node. Nodes corresponding to positive and negative residues are assigned single units of surplus and demand of some imagined commodity. Flow[3] of this commodity is allowed on arcs which connect neighboring nodes. The resulting grid-like network (see Fig. 2.6) therefore contains one arc for each phase gradient. As was pointed out in Section 2.1, it is these phase gradients that must be estimated to obtain an unwrapped solution. Therefore, the network model's arcs, and the quantities of flow on them, play a critical role in relating wrapped and unwrapped phase fields.

The network problem entails the arrangement of flow on some subset of the arcs, from surplus nodes, through intermediate nodes, to demand nodes. Flow must be conserved at all nodes: The total flow out of any node must equal the total flow in, plus or minus the node's surplus or demand. When this constraint is met, the solution is said to be feasible.

Physically, the amount of flow on an arc represents the difference $\Delta\phi - \Delta\psi$ between the

[2]We adopt the notation of networks containing nodes and arcs rather than that of graphs containing vertices and edges. We use the former to describe both, neglecting their subtle differences for simplicity sake.

[3]Note that the term "flow" can be used to describe either a flow on an arc or the arrangement of flow throughout a network.

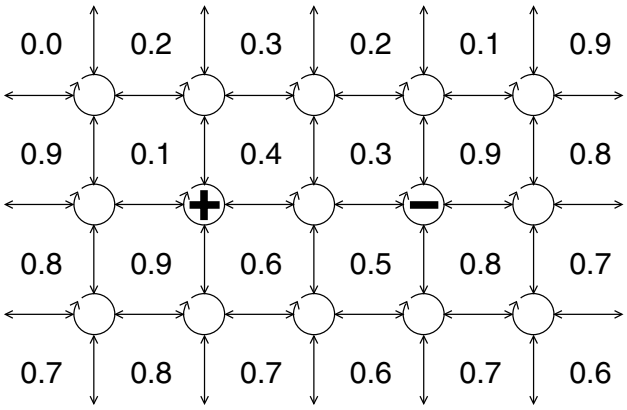

Figure 2.6 An example network equivalent of the phase unwrapping problem. The numbers represent a 2-D array of phase samples (normalized to one cycle). Each 2×2 clockwise loop integral of wrapped phase gradients is a node in the network, and positive and negative residues result in supply and demand nodes. Neighboring nodes are connected by arcs, or possible flow paths.

unwrapped and wrapped gradients associated with that arc, where the direction of the flow determines the sign of the difference. The importance of the network model lies in the fact that any feasible flow arrangement corresponds uniquely to a valid, residue-free unwrapped solution. That is, if flow is conserved at a node in the network model, the loop integral of the corresponding unwrapped gradients is zero. Thus, a wrapped phase field defines the locations of surplus and demand nodes in a network, and feasible-flow solutions for this network problem determine self-consistent sets of unwrapped gradient estimates that can be integrated along arbitrary paths to produce unwrapped solutions.

L^p objectives can still be used to determine which feasible flows—that is, which unwrapped phase fields—are most favorable. *Costantini* [1998] pointed out that because arc flows correspond to the quantity $\Delta\phi - \Delta\psi$, the network problem is linear in the flow variables when the L^1 metric is used [see Eq. (2.4)]. Therefore, fast, existing minimum-cost-flow (MCF) solver routines can be used to find L^1-optimal unwrapped phase fields. These solvers, designed for generic network applications, associate optionally weighted arc flows with scalar costs and compute solutions that minimize the total cost. By taking advantage of the special network structure of the problem, these routines attain much greater efficiency than can be had from general-purpose linear optimization algorithms (see the text by *Ahuja et al.*, [1993]). Such speed is especially important for phase unwrapping applications, which typically contain millions of variables.

It should be noted that *Flynn* [1997] used MCF ideas to propose an L^1 algorithm that predates Costantini's. Flynn did not explicitly adopt a network model, however, and he did not explore the relationships between network theory and phase unwrapping. We hence treat Flynn's algorithm as a specific implementation of the general MCF approach whose theoretical significance was first explored by *Costantini* [1998].

The applicability of the network model can also be extended beyond the L^1 MCF objective. Since the network model is independent of the criteria chosen for optimization, algorithms other than those using the L^1 norm can be viewed from a network perspective as well. For example, cuts in the residue-cut algorithm represent locations of possible phase discontinuities, so cuts are related to arcs that have nonzero flow in the network model. For least-squares algorithms, in which unwrapped and wrapped phase gradients are not necessarily congruent, flows on the equivalent network might be noninteger. In either case, the network model remains completely general.

Note that an even more general form of the network model includes upper and lower bounds on the arc flows. We assume here, however, that the arcs are unconstrained in the amounts of flow they may carry, so long as flow is conserved. We also assume bidirectional arcs, meaning that flow may go in either direction on any arc; the sign of the flow denotes its direction. In some contexts, bidirectional arcs are equivalently represented by pairs of oppositely directed unidirectional arcs for notational reasons [*Ahuja et al.,* 1993].

2.6 Other Approaches

The concepts described above overview the most popular of existing phase unwrapping approaches. Other more novel approaches are possible, too, though. For example, *Collaro et al.* [1998] proposed a genetic algorithm that implements a survival-of-the-fittest approach for choosing wrapped-gradient integration paths. *Xu and Cumming* [1999] proposed a region-growing approach that examines not only horizontal and vertical phase differences, but diagonal ones as well. Algorithms such as these remain unproven, but their diversity illustrates the open-ended nature of the phase unwrapping problem.

Chapter 3

Optimization Objectives and Complexity

As described in the previous chapter, phase unwrapping is often cast as a constrained optimization problem in which the objective is to minimize the value of some function that maps possible unwrapped solutions to scalar costs. This approach follows naturally from our want of the best possible—optimal in some sense—unwrapped solutions to the problems with which we are faced. Consequently, in this and the following chapters, we use an optimization framework for our development of a new phase unwrapping algorithm.

In constructing an optimization algorithm for our purpose, we face two main tasks: (1) setting up the problem, and (2) solving the problem. That is, we must first choose criteria for evaluating possible phase-unwrapped solutions and embody these criteria in an objective function that leads to accurate solutions when optimized. We must then develop means by which optimal or approximately optimal solutions in terms of this objective function may be practically and efficiently computed. In this chapter, we examine the relationships between these two tasks, laying the theoretical groundwork upon which we will build in Chapters 4 and 5.

The empirical performance records of existing L^p-norm algorithms elucidate many facets of the phase unwrapping problem, so we open this chapter with a discussion of the different L^p objectives, analyzing both their accuracy and ease of solution. Because we wish to go beyond the restrictive limits of the L^p-norm framework, however, we next generalize the problem by formulating objective functions composed of independent, arbitrarily shaped cost functions which afford us great flexibility in setting up an optimization problem. Solving

the arbitrary-cost-function problem is very difficult, though. We show in Appendix A that this problem is *NP*-hard, indicating that for all practical purposes, the problem is impossible to solve in an exact sense. We thus conclude this chapter by justifying the use of approximations in the development of real-world algorithms.

3.1 Minimum L^p-norm Approaches

Algorithms described by the minimum L^p-norm framework [*Ghiglia and Romero,* 1996] constitute only a subset of general optimization approaches, but their well-established performance records [*Ghiglia and Pritt,* 1998] provide instructive case studies of different optimization strategies. As described in Chapter 2 and repeated here for convenience, the L^p optimization objective can be written

$$\text{minimize} \left\{ \sum_{i,j} w_{i,j}^{(x)} \left| \Delta\phi_{i,j}^{(x)} - \Delta\psi_{i,j}^{(x)} \right|^p + \sum_{i,j} w_{i,j}^{(y)} \left| \Delta\phi_{i,j}^{(y)} - \Delta\psi_{i,j}^{(y)} \right|^p \right\} \tag{3.1}$$

where $\Delta\phi^{(x)}$ and $\Delta\psi^{(x)}$ are the x components of the unwrapped and wrapped phase gradients, respectively, and $\Delta\phi^{(y)}$ and $\Delta\psi^{(y)}$ are their y direction counterparts. Wrapped gradients always assume values between $-\pi$ and π rad. User-defined weights w are assigned to all neighboring-pixel phase differences (gradients), and the summations include all appropriate rows i and columns j.

It is important to note that we treat L^p norms and other objective functions as generic optimization criteria, independent of the methods used to minimize them and distinct from any specific algorithm implementations. Different algorithms may use the same objective function for guiding the unwrapping process, but if they use different solver routines, they may arrive at strikingly different results.

With this in mind, we treat congruence as an additional constraint on the optimization problem. When this constraint is enforced, corresponding unwrapped and wrapped phase values may differ only by integer numbers of cycles. Congruence therefore has the effect of making the solution space for unwrapped gradients discrete rather than continuous. Nevertheless, any objective function can still be used to compare different allowable solutions. Moreover, for the L^0 objective with integer weights, there always exists an optimal solution that is congruent (see Appendix A). The same is true for the L^1 objective [*Ahuja et al.,* 1993]. With the L^2 objective, though, a noncongruent optimum is generally better (in the

L^2 sense) than one computed under the assumption of congruence.[1] Note, however, that the former solution loses L^2 optimality—even over the set of strictly congruent solutions—if it is forced into congruence *after* optimization as described by *Ghiglia and Pritt* [1998].

3.1.1 L^p Objectives

It has been repeatedly suggested that the goal of phase unwrapping should be to minimize the total weighted length of discontinuities in the unwrapped phase, as in the L^0 metric [*Goldstein et al.,* 1988; *Buckland et al.,* 1995; *Ghiglia and Romero,* 1996; *Ghiglia and Pritt,* 1998]. This idea has some intuitive appeal, and in topographic applications of SAR interferometry, for example, it corresponds to minimizing the length of physical discontinuities in a real surface. Empirically, L^0 and L^1 algorithms have also tended to be more accurate than L^2 algorithms [*Ghiglia and Pritt,* 1998; *Zebker and Lu,* 1998; *Chen and Zebker,* 2000]. If it is in fact more desirable to minimize the L^0 objective than other L^p criteria, the accuracy of L^1 algorithms might be explained by the observation that L^0- and L^1-optimal solutions are often very similar, and are identical when the gradients of the optimal unwrapped phase fields differ from their wrapped counterparts by exactly one cycle or not at all [*i.e.*, $(\Delta\phi - \Delta\psi) \in \{-1, 0, 1\}$ cycles]. This may be the case in noisy, low-relief areas of an elevation-mapping interferogram, where residues exist mainly in positive-negative pairs. However, if the true topographic phase signature in the interferogram contains large discontinuities, as from layover for example, corresponding unwrapped and wrapped phase gradients may differ by many cycles. In such cases, the L^1 objective assigns high costs to these multiple-cycle discontinuities and might instead favor an incorrect set of single-cycle discontinuities. A simplified example of such a situation is depicted in Fig. 3.1; the L^0-optimal flow correctly reflects the multiple-cycle discontinuities introduced by layover, but the L^1-optimal flow results in a global error.

Still, L^1 solutions are appealing in that they can be calculated exactly and quite efficiently (see Chapter 2), and our ability to *find* good L^1 solutions may compensate, to some degree, for deficiencies in the objective itself. On the other hand, finding or computing exact L^0 solutions is very difficult. The residue-cut algorithm approximates L^0 behavior reasonably well in areas of good coherence, but very poorly in other areas [*Goldstein et al.,* 1988]. The *Ghiglia and Romero* [1996] minimum L^p-norm (LPN) algorithm converges on

[1] This can be shown by simply evaluating the L^2 norms of solutions from the respective algorithms for the same input data.

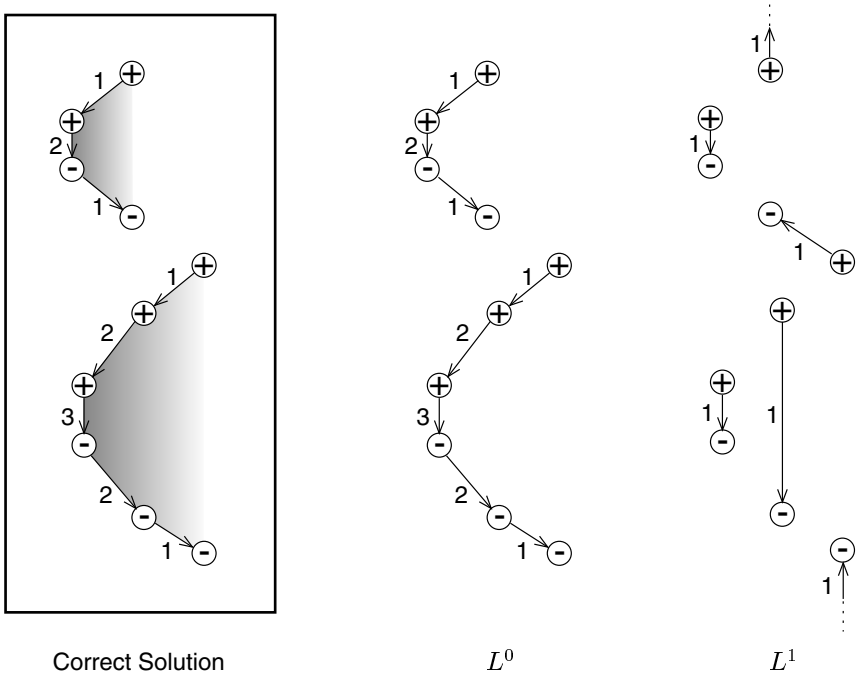

Figure 3.1 Topographic phase residue arrangement and L^0- and L^1-optimal flows resulting from the presence of layover. Radar illumination is from the left, and hence range increases towards the right. The brightness of the left panel represents the true unwrapped phase of a scene containing two features in layover. Discontinuities along the left edges of these features result in the residue arrangement shown. The arrows represent flow in the directions indicated, with magnitudes denoted by the numbers. The L^0-optimal flow (middle) correctly matches the true flow arrangement (left), while the L^1-optimal flow (right) is incorrect.

only local rather than global minima when $p = 0$. *Buckland et al.* [1995] proposed a minimum-cost matching algorithm and claimed that it finds "the global minimum of total cut length," but they assume that cuts are allowed only between positive-negative residue pairs (matchings), whereas a global L^0 minimum generally contains extended discontinuities (trees) with many residues on them. *Carballo* [2000] also asserted the solvability of the L^0 problem, describing linearization methods that can be used to solve nonlinear problems under certain conditions. Those conditions, however, reduce to the well-known requirement of convexity [*Ahuja et al.,* 1993; *Nash and Sofer,* 1996], while the L^0 objective function is far from convex in optimization space.

No known algorithm can efficiently compute an exact, globally minimal L^0 solution, and we describe below and prove in Appendix A that in all likelihood, none will *ever* be able to do so in the general case. Nevertheless, an objective function's computational difficulty does not necessarily detract from its appeal; L^0 algorithms relying on approximations and heuristics are therefore still very useful in practice even though they cannot compute exact L^0-optimal solutions.

3.1.2 Example L^p Results

We now examine a topographic test interferogram in order to illustrate the characteristics of the L^p algorithms described above and in Chapter 2. The interferogram, shown in Fig. 3.2(a) with magnitude in gray-scale brightness and phase in color, depicts a desert region north of Death Valley, California. The interferogram is formed from SAR images acquired 105 days apart by the European Space Agency ERS-1 satellite. This 1250×830 pixel interferogram, comprising five looks in azimuth and a single look in range, is shown with range increasing towards the right. The projected pixel spacing on the ground is approximately 20 m in both dimensions. Figure 3.2(b) shows the estimated interferogram coherence, calculated from twenty looks in azimuth and four looks in range. Here, the coherence is shown in color while the interferogram magnitude is again shown in gray-scale brightness. For topographic reference, we use the 30 m posting U.S. Geological Survey (USGS) digital elevation model (DEM) shown in Fig. 3.2(c), where the elevation is represented in color and a shaded-relief image generated from this DEM is shown in gray-scale brightness. The DEM's rms accuracy of 7.5 m is sufficient for identifying unwrapping errors since the interferogram ambiguity height is approximately 80 m.

The variety of topographic features in this interferogram allows us to analyze several

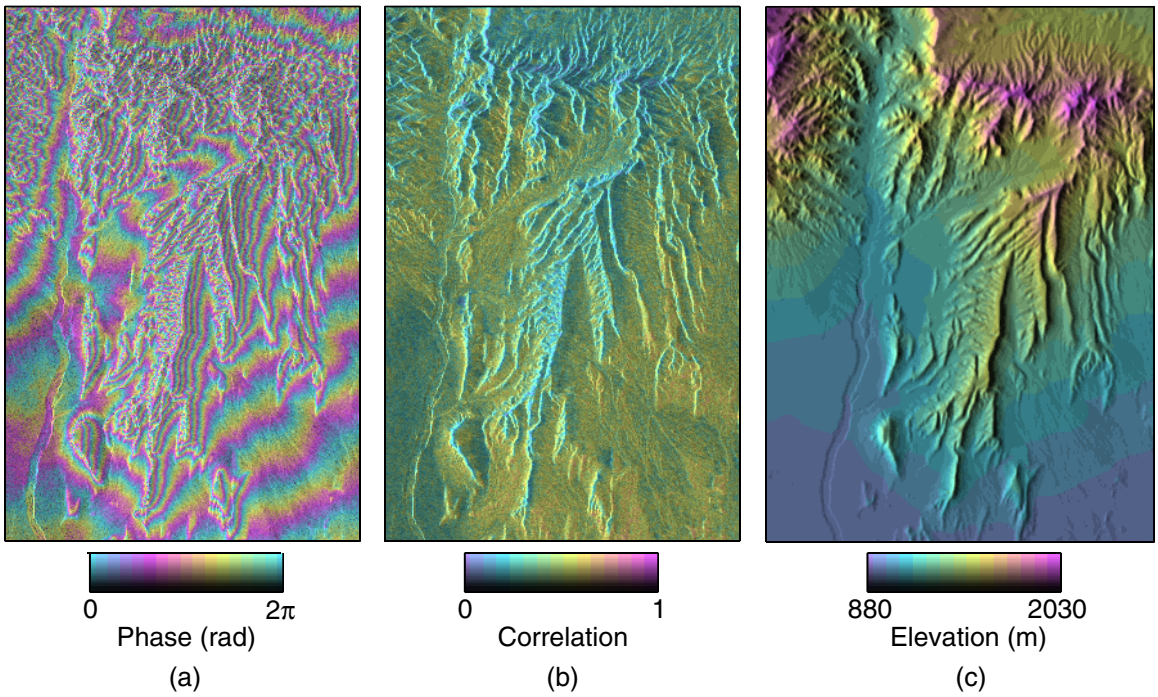

Figure 3.2 Death Valley topographic test data: (a) interferogram with wrapped phase in color and magnitude in gray-scale brightness; (b) measured coherence in color with interferogram magnitude in gray-scale brightness; (c) reference DEM with elevation in color and shaded relief in gray-scale brightness.

Algorithm	Execution Time (s)	Memory Usage (MB)
Least-squares	220	50
MCF	350	400
LPN	1960	70
Residue-cut	12	10

Table 3.1 Algorithm execution times and memory requirements for Death Valley test data on a Hewlett-Packard C-180 workstation.

problems often encountered in phase unwrapping. Running in azimuth in the middle of the image are long discontinuities resulting from layover, while the top of the image contains areas of rough topography. The bottom is relatively smooth, although it is not without areas of low correlation.

Algorithm performance results are shown in Fig. 3.3 for the (a) L^2 least-squares, (b) L^1 MCF, (c) L^0 LPN, and (d) L^0 residue-cut algorithms. The least-squares algorithm is implemented using a preconditioned conjugate-gradient (PCG) technique with congruence applied after optimization [*Ghiglia and Romero,* 1994; *Ghiglia and Pritt,* 1998]. The other algorithms are implemented as described in Chapter 2. Weightings for the results in (a)–(c) are based on the thresholded interferogram coherence magnitude, while the results in (d) are unweighted since the residue-cut algorithm does not accept weights. For all panels, the color represents relative unwrapped phase error, calculated by subtracting the DEM-derived unambiguous phase from the algorithm solutions. The gray-scale brightness again shows the interferogram magnitude. For the residue-cut result, areas in black indicate that no solution was produced. Algorithm execution times are summarized in Table 3.1.

Since these algorithms all produce congruent solutions, phase errors due to incorrect unwrapping can be easily identified as areas differing from their surroundings by integer numbers of cycles. Other errors of less than one cycle may be due to atmospheric effects [*Zebker et al.,* 1997], inaccuracies in the DEM, noise in the interferogram, or artifacts from the transformation and registration of the DEM to radar coordinates. In any case, it is easy to distinguish the patch-like, integer-cycle unwrapping errors from other errors. Of course, there is always an unknown constant offset in interferogram phase, so we can determine error only in a relative sense. Color differences between patches thus indicate relative errors, and the quality of an unwrapped solution can be evaluated by its degree of color homogeneity.

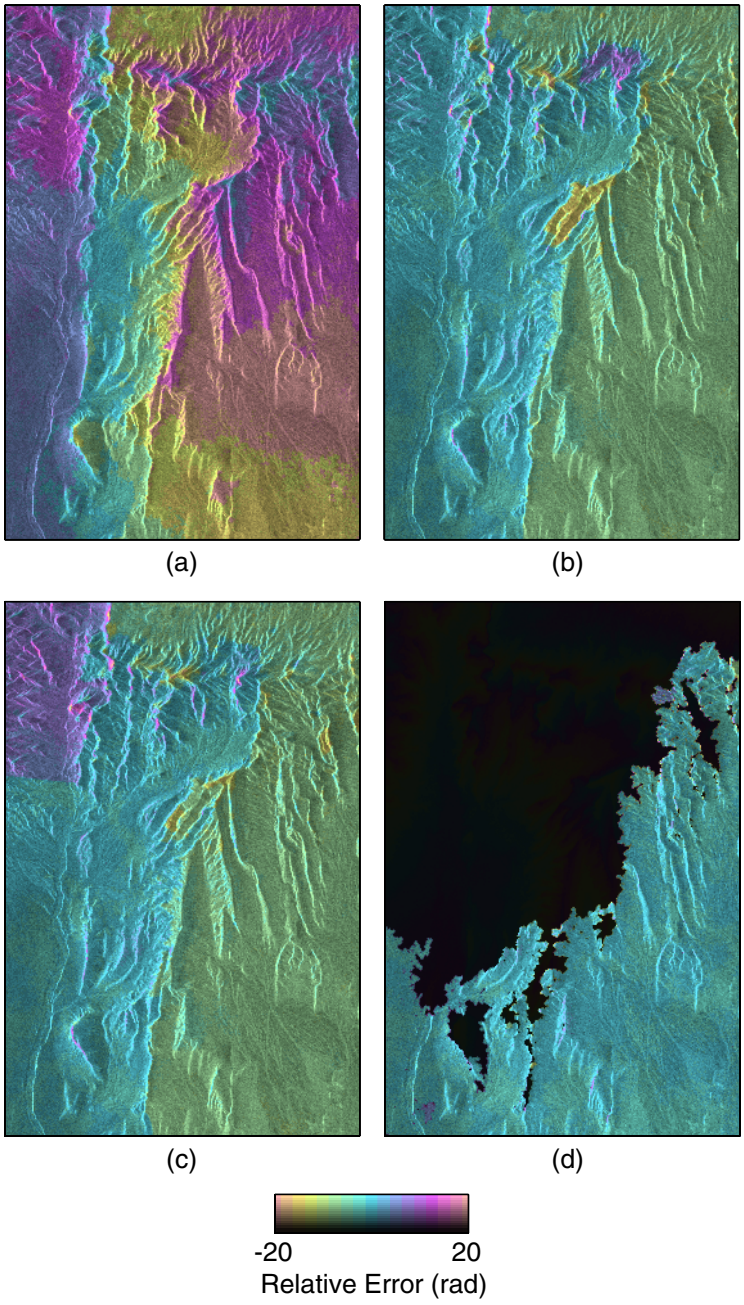

Figure 3.3 Relative unwrapped-phase errors for the Death Valley test data from different L^p algorithms: (a) least-squares (L^2); (b) MCF (L^1); (c) LPN (L^0); (d) residue-cut (L^0). The interferogram magnitude is shown in gray-scale brightness. Black areas in (d) indicate that no solution was produced.

As is evident by the massive color segmentation in (a), the least-squares algorithm is highly inaccurate when congruence is applied only after optimization. The poor performance of this algorithm is an example of the idea that unless an algorithm's optimization criteria are reasonable, its solutions will be unreliable, regardless of how elegantly or efficiently these solutions are computed.

The MCF and LPN algorithms perform better, but they both still segment the scene into large pieces with relative elevation errors on the order of the 80 m ambiguity height. The residue-cut algorithm is very accurate where it unwraps in the areas of gentle terrain, but it fails to produce a solution for nearly half the interferogram. Notably, the LPN and residue-cut algorithms generate very different solutions from the same (L^0) objective function. Furthermore, despite our expectation that the L^0 norm should lead to more accurate results than the L^1 norm, the L^1 MCF solution is qualitatively comparable to if not better than the L^0 LPN and residue-cut solutions. In fact, the solution that is best in the coherence-weighted L^0 sense is produced by the MCF algorithm, even though this algorithm optimizes an L^1 objective. The quality of an algorithm's unwrapped solution is thus determined not only by its objective function, but also to a great extent by its method of solution.

All of the solutions shown in Fig. 3.3 are characterized by significant error. While this lack of accuracy clearly motivates the approach put forth in subsequent chapters, the results also have another, more subtle implication: In order to achieve the best possible algorithm performance, attention must be paid to both setting up and solving the phase unwrapping optimization problem.

3.2 Setting up the Problem: Generalized Cost Functions

Minimum L^p-norm methods are popular and have led to reasonable success in many phase unwrapping applications, but they are by no means the only optimization approaches available. In this section, we generalize the optimization framework so that we may more flexibly set up a phase unwrapping optimization problem. In the most general case, an optimization objective may involve minimizing any function of the set of all phase values:

$$\text{minimize}\left\{G_{\Phi,\Psi}(\Phi,\Psi)\right\}. \tag{3.2}$$

Here, $G(\cdot)$ is the objective function to be minimized, and Φ and Ψ are the sets of all unwrapped and wrapped phase values ϕ and ψ, respectively (the capital letters denote arrays or vectors). Because unwrapped and wrapped phase fields are completely specified by their unwrapped and wrapped gradients, except for an additive constant not of interest here, the general optimization objective can be expressed equivalently as

$$\text{minimize}\left\{G_{\Delta\Phi,\Delta\Psi}(\Delta\Phi,\Delta\Psi)\right\} \tag{3.3}$$

where $\Delta\Phi$ and $\Delta\Psi$ are the sets of all unwrapped and wrapped phase gradients $\Delta\phi$ and $\Delta\psi$, in both the row-wise and column-wise directions.

In the interests of computational efficiency, however, the objective function is commonly assumed to be separable so that the objective can be written

$$\text{minimize}\left\{\sum_{i=1}^{M}\sum_{j=1}^{N-1} g_{i,j}^{(x)}\left(\Delta\phi_{i,j}^{(x)},\Delta\psi_{i,j}^{(x)}\right)+\sum_{i=1}^{M-1}\sum_{j=1}^{N} g_{i,j}^{(y)}\left(\Delta\phi_{i,j}^{(y)},\Delta\psi_{i,j}^{(y)}\right)\right\} \tag{3.4}$$

where $\Delta\phi^{(x)}$ and $\Delta\psi^{(x)}$ are the x-direction unwrapped and wrapped phase gradients, respectively, and $\Delta\phi^{(y)}$ and $\Delta\psi^{(y)}$ are their y-direction counterparts for a 2-D $M\times N$ array. In the range-azimuth coordinate system of a side-looking imaging radar, we define $\Delta\phi^{(r)}=\Delta\phi^{(x)}$, $\Delta\psi^{(r)}=\Delta\psi^{(x)}$, $\Delta\phi^{(a)}=\Delta\phi^{(y)}$, and $\Delta\phi^{(a)}=\Delta\phi^{(y)}$. The functions $g(\cdot)$ are called cost functions, and they are each unrestricted in form. Transforming the 2-D arrays of Eq. (3.4) into 1-D vectors, we can collapse the above expression into a simpler but equivalent form:

$$\text{minimize}\left\{\sum_{k} g_k\left(\Delta\phi_k,\Delta\psi_k\right)\right\}. \tag{3.5}$$

With these cost functions defining the optimization objective, our task in setting up the problem is to formulate appropriate expressions for $g_k(\Delta\phi_k,\Delta\psi_k)$, for all k.

Under the minimum L^p-norm framework described by Eq. 3.1, the cost functions are defined by

$$g_k(\Delta\phi_k,\Delta\psi_k)=w_k\left|\Delta\phi_k-\Delta\psi_k\right|^p. \tag{3.6}$$

Example plots of these functions are shown in Fig. 3.4. Although L^p cost functions have performed reasonably well empirically as described above, their simple geometric shapes

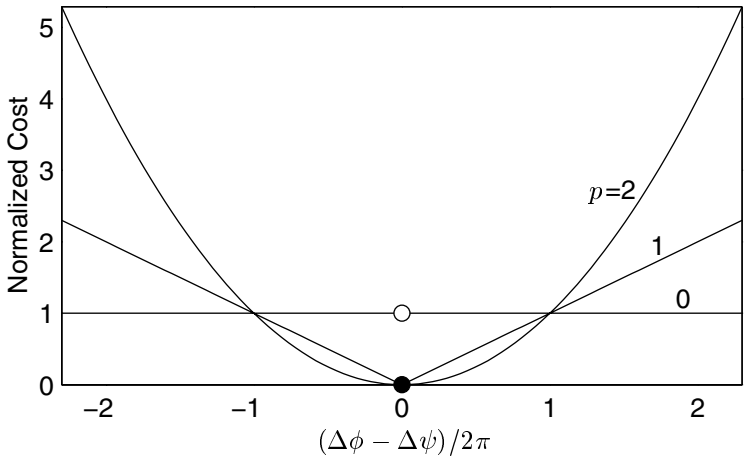

Figure 3.4 Cost functions for the L^p family of objective functions. The abscissa is the normalized flow, or the difference in cycles between the unwrapped and wrapped gradients. Normalized costs are plotted for the unweighted case.

have little physical meaning. Indeed, there is no theoretical reason that exactly optimal L^p solutions should be correct with respect to the true unwrapped phase: L^p objectives are just abstract mathematical quantities which often lead to workable solutions.

Solutions from L^p objectives are also highly dependent on the weights used to individually scale the cost functions within a scene. The subject of advantageous weights has been widely examined (see, for example, *Zebker and Lu* [1998], *Ghiglia and Pritt* [1998], and *Chen and Zebker* [2000]), but because all cost functions for a given L^p objective necessarily have the same principal shape throughout an interferogram, most investigations have treated cost function shape and scaling as two distinct issues. The generalized cost functions of Eq. (3.5), on the other hand, may differ in both shape and scaling. Because the cost function for each neighboring-pixel phase difference in the interferogram has its own individual form—arbitrary and independent of all others—generalized cost functions provide us with great freedom in the formation of optimization objectives.

The generalized framework does not prescribe specific objectives, though. We address that challenge in Chapter 4. We point out for now, however, that on a theoretical level, assumptions are unavoidable in setting up the optimization problem. This is because, as stated in Chapter 2, an unwrapped phase field necessarily contains information unavailable in the wrapped data alone. As a result, no single objective can lead to perfectly correct

solutions in all circumstances.

3.3 Solving the Problem: *NP*-hardness

Solving the phase unwrapping optimization problem is no less important than setting it up, as demonstrated by the results of Fig. 3.3. Solving the problem, however, often involves a new level of intractability—the difficulty of optimizing the L^0 objective is a case in point. Algorithms have been developed for exactly minimizing the L^1 and L^2 norms [*Flynn,* 1997; *Costantini,* 1998; *Ghiglia and Romero,* 1994], but these objectives are only special, highly simplified cases of the generalized problem expressed by Eq. (3.5). Without such simplifications, the problem is *NP*-hard, or in other words, is one of the most difficult problems known to complexity theory; it is believed that such problems cannot be solved exactly by efficient algorithms (the exact meaning of *NP*-hardness is described below). In this section, we describe our method for proving the *NP*-hardness of the phase unwrapping optimization problem. The proof itself is given in Appendix A. We briefly review the idea of *NP*-hardness here, but the reader is directed to references such as the text by *Garey and Johnson* [1979] for more detail on this subject.

Complexity theory holds that in all likelihood, exact solutions to *NP*-hard problems cannot be computed in polynomial time. A polynomial-time algorithm is one whose worst-case number of operations or worst-case running time is bounded by some polynomial in the size of the problem instance. Suppose, for example, that the maximum number of operations performed by some phase unwrapping algorithm is n^2, where n is the number of pixels in the input interferogram. Because n^2 is a polynomial in the problem size, the algorithm is said to run in polynomial time. On the other hand, if the number of operations performed by the algorithm can be limited only by a term such as 2^n or $n!$, the algorithm's computational complexity cannot be bounded by any polynomial in n as n gets large. Clearly, an algorithm whose execution time increases exponentially or factorially with the number of input variables quickly becomes impracticable. Of course, an algorithm's average-case complexity may be much less than its worst-case complexity, but as explained below, we nevertheless have little hope of efficiently computing exact solutions to *NP*-hard problems.

NP-hard problems derive their difficulty from a more restrictive set of problems known as class *NP*-complete. *NP*-complete problems are posed as questions with yes-or-no answers, and among other interesting characteristics, any *NP*-complete problem can be transformed

into any other *NP*-complete problem in polynomial time. Therefore, if some polynomial-time algorithm could solve any one *NP*-complete problem, it could also be used to solve all others in polynomial time as well. Such an algorithm has never been found to exist, however, and the prevailing assumption in the field is that no such algorithm can exist. Problems in class *NP*-hard are those that, if solvable exactly, could also yield solutions to *NP*-complete problems through polynomial-time transformations.[2]

Thus, to show that the general phase unwrapping optimization problem of Eq. (3.5) is *NP*-hard, we show that some *NP*-complete problem can be polynomially transformed into it. More specifically, we show that if a hypothetical black box could solve the L^0 optimization problem, it could also be used to solve any instance of an *NP*-complete problem called the Rectilinear Steiner Tree (RST) problem [*Garey and Johnson,* 1977]. We address the details and subtleties of this transformation in Appendix A, proving the *NP*-hardness of the L^0 problem. Since any algorithm that solves the general phase unwrapping problem also solves the L^0 problem, the general problem is *NP*-hard as well. Therefore, without simplifying assumptions such as those that restrict cost function shapes to those of the L^1 or L^2 norms, phase unwrapping is at least as difficult as any *NP*-complete problem and is among the most difficult known to complexity theory.

3.4 Implications of Intractability

Despite the intractability of the problem, there remains a very real need for phase unwrapping. The difficulty of the problem therefore suggests not that we should abandon our efforts, but merely that we should direct them towards the design of practical, approximate[3] algorithms rather than necessarily exact or mathematically elegant ones.

The lack of complete information in a wrapped phase array and the *NP*-hardness of the general optimization problem imply intractability on multiple levels, however. We must therefore balance our approximations between setting up and solving the problem if we are to pursue simultaneously the dual goals of formulating an accurate objective function and developing a practical optimization routine for it. That is, the problem should neither be set up with an objective function that is easily solvable but prone to give inaccurate results,

[2]The letters "*NP*" arise because problems in a class called 'class *NP*' are accepted by a hypothetical nondeterministic Turing machine in polynomial time. See the text by *Garey and Johnson* [1979].

[3]While the term "approximate" implies algorithm performance guarantees and/or complexity bounds in some contexts, its meaning in this dissertation is colloquial and simply denotes inexactness.

nor should it be set up with an objective function that is theoretically ideal but impossible to optimize robustly.

The need for balance between theoretical rigor and computational manageability does not imply that the two always trade off, however. In the following two chapters, we propose methods for enhancing both. Chapter 4 describes generalized, statistical cost functions wherein phase unwrapping is viewed through the lens of maximum *a posteriori* probability (MAP) estimation. Chapter 5 describes network-flow techniques for computing approximate solutions to the nonlinear optimization problem implied by Chapter 4. The simplifications and approximations in either chapter may make aspects of the algorithm nonideal individually, but our intent is to design a working algorithm that provides the best possible results overall.

Chapter 4

Statistical Cost Functions

We have thus far shown that phase unwrapping can be cast as a nonlinear optimization problem; we must now decide what to optimize. Our optimization criteria, expressed mathematically as objective functions, will be used to guide the solver routines of Chapter 5. Thus, our task for this chapter is to derive objective functions that lead to accurate solutions when minimized.

Aiming to strengthen the physical foundations of these optimization objectives, we pose phase unwrapping as a maximum *a posteriori* probability (MAP) estimation problem which we set up using generalized, nonlinear cost functions. Because the problem statistics vary for different phase unwrapping applications, however, the MAP framework implies the need for application-specific optimization objectives. Such objectives should yield greater accuracy than can be obtained from more generic approaches. Here, we examine topographic and deformation-mapping applications of SAR interferometry, deriving MAP cost functions for each.

Of course, the accuracy of the MAP approach depends greatly on our ability to model the elaborate probability relationships of the problem. Such relationships are very difficult to formulate exactly, however, and realistic solver routines cannot guarantee exact solutions to the subsequent *NP*-hard minimization problem in any event. Consequently, our approach here is pragmatically geared towards the design of a more modest algorithm that—even if inexact—provides significantly better results in an applied, practical sense than other existing algorithms. Following this philosophy, we approximate the problem statistics with simple models whose refinement we leave for later work. We present here the general methodology and practical application of our approach, focusing on physical insight rather

than mathematical detail.

4.1 MAP Cost Functions

The generalized cost functions of Eq. (3.5) allow the phase unwrapping problem to be set up with great flexibility, and we now use this flexibility to design an optimization objective based on MAP estimation. That is, given a 2-D wrapped phase field Ψ, we develop an objective function such that its minimization results in an estimate $\hat{\Phi}$ of the true unwrapped phase field Φ, where $\hat{\Phi}$ approximately maximizes the conditional probability density function (PDF) $f(\Phi|\Psi)$.

In our notation, the capital letters denote arrays, while lowercase letters denote individual entries in those arrays. That is, $\Delta\Phi$ is the set of all unwrapped gradients $\Delta\phi$ in an interferogram, where $\Delta\phi$ is any particular row-wise or column-wise gradient (uppercase and lowercase letters are not random variables and their specific instances). We use the function $f(\cdot)$ to signify both probability mass and density functions; unless otherwise noted by a subscript, $f(\cdot)$ describes the random variable or variables in its argument. Thus, for example, $f(\Phi|\Psi)$ is equivalent to $f_{\Phi|\Psi}(\Phi|\Psi)$.

We begin our formulation of the MAP objective by changing its variables, without loss of generality, as in Eq. (3.3). Instead of dealing with the phase values themselves, our goal now is to estimate the set of all unwrapped gradients $\Delta\Phi$ given the set of all wrapped gradients $\Delta\Psi$. We next assume that the PDF $f(\Delta\Phi|\Delta\Psi)$ is separable, implying that the individual unwrapped phase gradients $\Delta\phi$ are statistically independent given their wrapped counterparts $\Delta\psi$ and given the knowledge that the resulting unwrapped phase field Φ is a residue-free (irrotational) surface. This latter condition is enforced by our solver routine (see Chapter 5), which ignores invalid, nonfeasible solutions. Of course, our assumption of independence is not strictly correct, but its viability is born out by empirically verified results (see Chapter 7), and it is in any case required for computational tractability. Thus,

$$f(\Delta\Phi|\Delta\Psi) = \prod_k f(\Delta\phi_k|\Delta\psi_k). \tag{4.1}$$

The product with index k in this expression is taken over all rows and columns for the sets of both row-wise and column-wise (range and azimuth) gradients. Using logarithms, we transform the maximization of this product into the minimization of a sum, and the

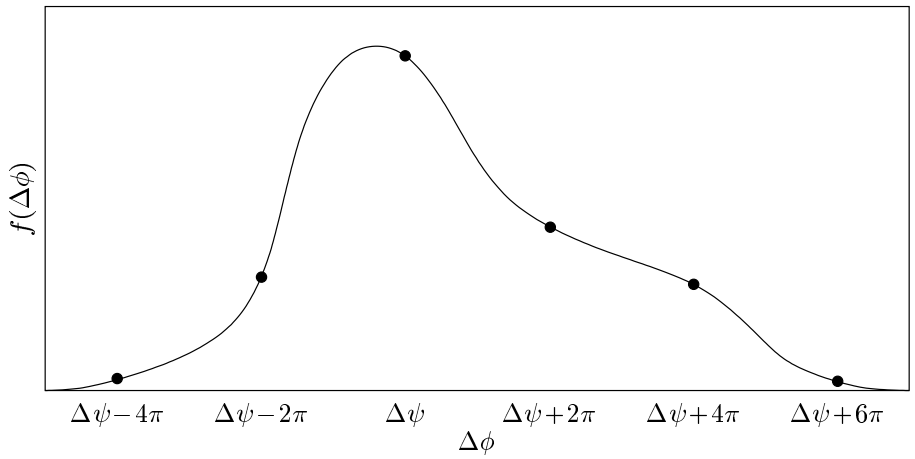

Figure 4.1 Example unwrapped gradient PDF $f(\Delta\phi)$, marked at the discrete values of $\Delta\phi$ that are allowed under the assumption of congruence. Congruence limits $\Delta\phi$ to values at integer-cycle offsets from $\Delta\psi$ (indicated by dots). The relative probabilities of these values of $\Delta\phi$ are determined by the nonconditional PDF $f(\Delta\phi)$. Other values of $\Delta\phi$ have zero probability given $\Delta\psi$. Note that the shape of the PDF shown here is for illustration only and does not necessarily depict our model for the statistics of real data.

objective becomes

$$\text{minimize}\left\{-\sum_k \log\left(f(\Delta\phi_k|\Delta\psi_k)\right)\right\}. \tag{4.2}$$

Treating Eq. (4.2) as a special form of Eq. (3.5), we thus define our MAP cost functions as the negative logarithms of the unwrapped-gradient PDFs:

$$g_k(\Delta\phi_k, \Delta\psi_k) = -\log\left(f(\Delta\phi_k|\Delta\psi_k)\right). \tag{4.3}$$

These cost functions may be further simplified through our assumption of congruence, which limits an unwrapped gradient to the discrete set of values at integer-cycle offsets from the wrapped gradient (see Fig. 4.1). Specific instances of a particular unwrapped gradient then have probabilities that may be expressed in terms of the gradient's nonconditional PDF as

$$f(\Delta\phi|\Delta\psi) = \begin{cases} \dfrac{f_{\Delta\phi}(\Delta\phi)}{\sum_{m=-\infty}^{\infty} f_{\Delta\phi}(\Delta\psi + m2\pi)} & \text{if } \Delta\phi = \Delta\psi + n2\pi, \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

where n and m are integers. Note that $f(\Delta\phi|\Delta\psi)$ represents a probability mass, not density, function. The denominator of the fraction does not depend on $\Delta\phi$ and thus has no effect on the minimization problem of Eq. (4.2). Consequently, we need model only the individual unwrapped-gradient distributions $f(\Delta\phi)$ and evaluate them at integer-cycle offsets from the wrapped phase. Our MAP cost functions then become

$$g_k(\Delta\phi_k, \Delta\psi_k) = -\log\left(f(\Delta\phi_k)\right) \text{ for } \Delta\phi_k \equiv \Delta\psi_k + n2\pi. \tag{4.5}$$

Subject to our assumptions, the minimization of these cost functions maximizes the conditional probability of the unwrapped solution given the wrapped data.

As with any MAP estimate, however, the unwrapped solution benefits from the inclusion of additional information. Intensity and coherence information are present in all SAR interferograms, so we explicitly rewrite $f(\Delta\phi)$ as a conditional PDF, recasting Eq. (4.5) as

$$g_k(\Delta\phi_k, \Delta\psi_k) = -\log\left(f(\Delta\phi_k|I,\rho)\right) \text{ for } \Delta\phi_k \equiv \Delta\psi_k + n_k2\pi \tag{4.6}$$

where I is the average of the intensities of the SAR images forming the interferogram, and ρ is the magnitude of the interferogram complex correlation coefficient [*Zebker and Villasenor,* 1992]. The interferogram phase statistics may be represented very compactly in this form by the term $f(\Delta\phi|I,\rho)$, but this conditional PDF actually embodies untold complexity. The shape of the PDF also varies throughout the interferogram, so different unwrapped gradients have different cost functions based on the local values of I and ρ. Moreover, different applications have different statistics, so we must treat each application separately. In the following sections, we model the phase noise statistics common to different applications then derive specific cost functions for topographic and deformation-mapping applications of SAR interferometry.

4.2 Phase-Noise Statistics

For geophysical applications, unwrapped phase values ϕ represent measurements of some physical quantity in the presence of noise. We now model the statistics of the noise, which we treat as an additive quantity corrupting the true unwrapped phase signal. Thus, for each pixel,

$$\phi = \phi_{\text{signal}} + \phi_{\text{noise}} \tag{4.7}$$

and, for each unwrapped gradient,

$$\Delta\phi = \Delta\phi_{\text{signal}} + \Delta\phi_{\text{noise}}. \tag{4.8}$$

Note that ϕ_{noise} represents phase noise, not the phase of the complex noise.

Given the coherence magnitude, the phase noise is unrelated to the intensity and independent of the measured signal. That is, any dependence of $\Delta\phi_{\text{noise}}$ on I or $\Delta\phi_{\text{signal}}$ is subsumed by knowledge of ρ. Thus, the conditional unwrapped-gradient PDF is the convolution of the probability densities corresponding to Eq. (4.8):

$$f(\Delta\phi|I,\rho) = f(\Delta\phi_{\text{signal}}|I,\rho) * f(\Delta\phi_{\text{noise}}|\rho). \tag{4.9}$$

Lee et al. [1994] derived analytical expressions for wrapped, multilook interferometric phase-noise PDFs $f(\psi_{\text{noise}}|\rho)$, and in areas of high correlation, these PDFs can be approximated by zero-mean normal distributions with variance $\sigma^2_{\psi_{\text{noise}}}$. Since the correlation varies slowly across most parts of the interferogram, neighboring phase-noise terms can usually be treated as identically distributed as well as independent. In areas of high correlation, their difference $\Delta\phi_{\text{noise}}$, the second term on the right-hand side of Eq. (4.8), can then be modeled by a zero-mean Gaussian random variable whose variance $\sigma^2_{\Delta\phi_{\text{noise}}}$ is twice that of the individual wrapped phase-noise variances:

$$\sigma^2_{\Delta\phi_{\text{noise}}} = 2\sigma^2_{\psi_{\text{noise}}}. \tag{4.10}$$

We thus calculate $\sigma^2_{\Delta\phi_{\text{noise}}}$ from a model for the single-pixel phase-noise standard deviation $\sigma_{\psi_{\text{noise}}}$. Our model, shown in Fig. 4.2 as a function of ρ for different numbers of independent looks N_i, is based on similar plots given by *Li and Goldstein* [1990], *Rodriguez and Martin*

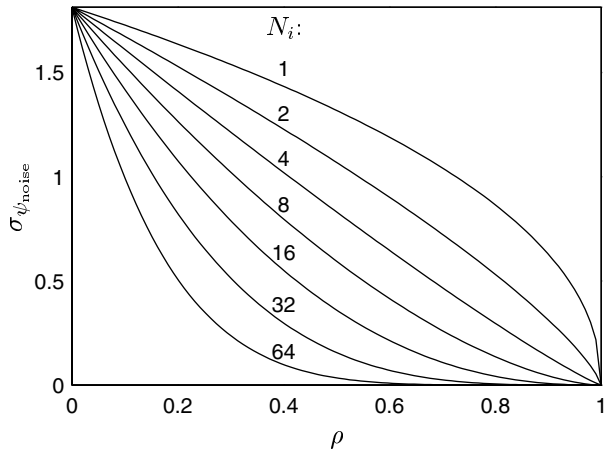

Figure 4.2 Model interferometric phase-noise standard deviation $\sigma_{\psi_{\text{noise}}}$ as a function of interferogram coherence ρ for different numbers of independent looks N_i.

[1992], and *Lee et al.* [1994]. The number of independent looks N_i is generally not equal to the actual number of complex looks N_c since the size of a resolution element is usually larger than the size of a pixel. If the correlation changes rapidly from one pixel to the next, Eq. (4.10) can be modified so that $\sigma^2_{\Delta\phi_{\text{noise}}}$ is calculated from the two different values of $\sigma^2_{\psi_{\text{noise}}}$ involved in the gradient.

For phase gradients located where the correlation is low, our normal approximation for $\Delta\phi_{\text{noise}}$ is less valid. For example, if the data are completely decorrelated, the PDF $f(\Delta\phi_{\text{noise}}|\rho = 0)$ is triangular, resulting from the convolution of two uniform $(-\pi, \pi)$ distributions. Rather than take on the computation burden of iterative hypergeometric-function evaluations as required by the exact PDF expression, however, we maintain the normal approximation in our algorithm even for low-coherence areas.

Several curves representing our model phase noise PDFs $f(\Delta\phi_{\text{noise}}|\rho)$ for different values of ρ are plotted in Fig. 4.3. The Gaussian functions shown reasonably approximate the true PDFs even though the latter should be nonzero only for $|\Delta\phi_{\text{noise}}| < 2\pi$.

In calculating $\sigma^2_{\Delta\phi_{\text{noise}}}$ from the observed coherence, we must also be aware of the bias

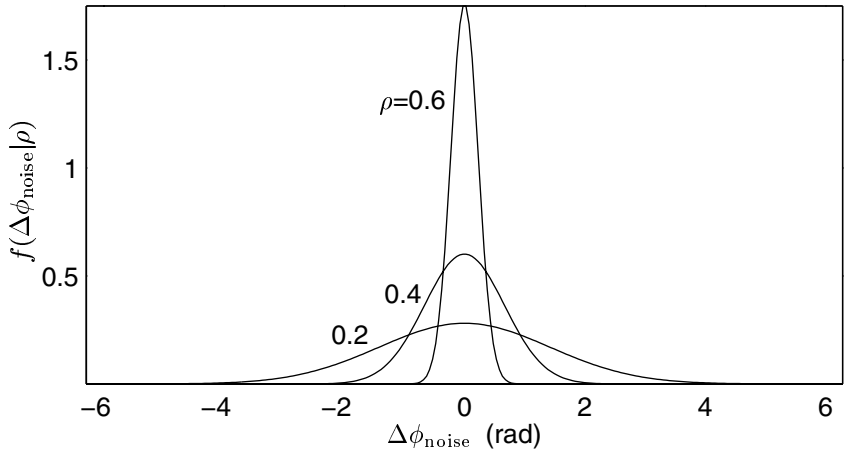

Figure 4.3 Model interferometric phase-noise PDFs $f(\Delta\phi_{\text{noise}}|\rho)$ for different values of ρ and twenty independent looks.

introduced by the common coherence estimator

$$\hat{\rho} = \left| \frac{\sum_{k=1}^{N_c} s_{1k} s_{2k}^*}{\sqrt{\sum_{k=1}^{N_c} |s_{1k}|^2 \sum_{k=1}^{N_c} |s_{2k}|^2}} \right| . \tag{4.11}$$

Here, $\hat{\rho}$ is the biased estimate of the true coherence magnitude ρ, s_1 and s_2 are the signals forming the interferogram, "*" denotes complex conjugation as before, and N_c is the number of complex pixels (looks) used for the estimate. *Touzi et al.* [1999] found an exact expression—involving another hypergeometric function—for the expected value of $\hat{\rho}$ in terms of ρ and N_i. We use a piecewise-linear model based on these results to estimate ρ from $\hat{\rho}$.

Independently, *Carballo and Fieguth* [2000] proposed phase-noise weights for an L^1 minimization problem using an approach similar, but not identical, to the one described here. *Lyuboshenko and Maître* [1999] used phase-noise statistics in an attempt to correct for the biases inherent in least-squares unwrapped phase estimates. Neither approach gave much consideration to the statistics of the signal part of Eq. (4.9), though. They instead assumed that unwrapped and wrapped gradients virtually never differ by more than one

cycle. Because of physical discontinuities in the measured signal, however, multiple-cycle phase gradients are in fact quite common and very important in many phase unwrapping applications. We model their statistics below.

4.3 Topography Measurements

If we are to model the statistics of our interferometric data, we must consider, in addition to noise, the signal components of our unwrapped gradients. These are represented by the first term on the right-hand side of Eq. (4.9). However, because the signal statistics are determined by the physical nature of the interferometric measurements, we must model these statistics separately for different phase unwrapping applications. We do so in this section for topographic SAR interferometry, defining $\phi_{\text{topo}} = \phi_{\text{signal}}$ and $\Delta\phi_{\text{topo}} = \Delta\phi_{\text{signal}}$.

If the topography-independent (flat-earth) phase signature is removed from the interferometric data, the true unwrapped topographic phase at a particular pixel is approximately related to the relative local surface elevation z by

$$\phi_{\text{topo}} = \frac{-4\pi B_{\perp}}{\lambda r \sin\theta} z \tag{4.12}$$

where $B_{\perp}$ is the component of the interferometer baseline perpendicular to the radar signal, λ is the signal wavelength, r is the range, and θ is the look angle with respect to nadir (see Fig. 1.2) [*Zebker and Goldstein,* 1986]. An unwrapped topographic gradient $\Delta\phi_{\text{topo}}$ thus depends nearly linearly on the physical surface slope Δz, although care must be taken to properly interpret measurements in the SAR range-azimuth coordinate system.

Separating the intensity and correlation dependencies of the unwrapped topographic gradient, we apply Bayes' rule to rewrite the gradient's conditional PDF as

$$f(\Delta\phi_{\text{topo}}|I,\rho) = \frac{f(\Delta\phi_{\text{topo}}|I)f(\rho|\Delta\phi_{\text{topo}},I)}{f(\rho|I)}. \tag{4.13}$$

The denominator on the right-hand side does not depend on $\Delta\phi_{\text{topo}}$, so we need model only the numerator for our optimization problem. We assume that the correlation ρ is independent of the intensity I given the actual topography $\Delta\phi_{\text{topo}}$, so

$$f(\Delta\phi_{\text{topo}}|I,\rho) \propto f(\Delta\phi_{\text{topo}}|I)f(\rho|\Delta\phi_{\text{topo}}). \tag{4.14}$$

We now treat the two terms on the right-hand side of this expression in turn.

4.3.1 Topography and Intensity

The first PDF on the right-hand side of Eq. (4.14) describes the relationship between topography and intensity; to quantify this relationship, we model the dependence of the radar image brightness on the local surface slope. The close correspondence between the two is evident upon inspection of any SAR image, and efforts have been devoted to both the recovery of topographic information from SAR intensity alone [*Guindon,* 1990] as well as the correction of radiometric measurements for topography-induced intensity effects [*Goering et al.,* 1995].

The coherent nature of SAR images—the very quality that permits interferometry—also causes speckle in the intensity, however. Because speckle complicates the inference of topographic information from image brightness, we use an adaptive speckle-removal filter based on ideas from *Lopes et al.* [1993]. Our filter computes the mean intensity $E[I]$ of a variably oriented rectangular window, selecting the window orientation that maximizes contrast with the local background. While the subjects of speckle statistics and speckle removal have been examined extensively (see the text by *Oliver and Quegan* [1998]), we assume perfect speckle removal to avoid unduly complicating our model PDFs. We then normalize $E[I]$, dividing by a coarse moving-window average.

We relate the normalized, speckle-removed image intensity to the surface topography using

$$E[I] = C\sigma^0 A \tag{4.15}$$

where σ^0 is the local normalized radar cross section of the illuminated surface, A is the area of the surface contributing to the measurement, and C is a scaling factor which may be ignored if the intensity data are normalized. Topography enters this equation as both σ^0 and A are functions of the local signal incidence angle θ_i. We obtain expressions for these quantities by using a facet model to represent the ground surface demarcated by a single range-azimuth pixel (see Fig. 4.4). In the model's right-handed (x, y, z) coordinate system, earth curvature is neglected, and x and z are aligned with increasing ground range and elevation. The ground-range and azimuth pixel spacings are denoted by Δx and Δy, respectively, and their corresponding components of local elevation change are denoted by

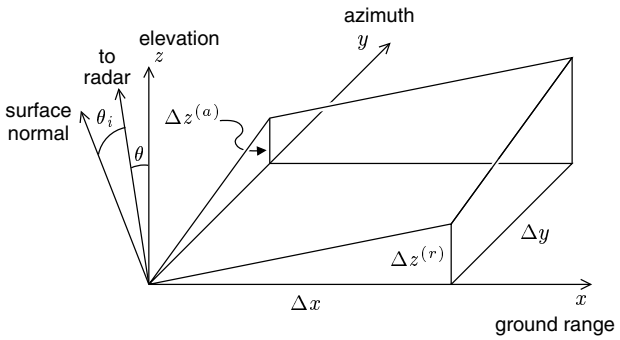

Figure 4.4 Facet model used to relate topography to the brightness of a single SAR range-azimuth pixel. Note that Δx depends on both Δr and $\Delta z^{(r)}$.

$\Delta z^{(r)}$ and $\Delta z^{(a)}$. Note that through Eq. (4.12), $\Delta z^{(r)}$ and $\Delta z^{(a)}$ are nearly proportional to the topographic parts of the unwrapped phase gradients. Note also that while the slant-range bin spacing Δr is constant, the ground-range spacing depends on the local slope through

$$\Delta x = \frac{\Delta r}{\sin\theta} + \frac{\Delta z^{(r)}}{\tan\theta}. \tag{4.16}$$

Assuming that the SAR viewing direction is exactly normal to the sensor velocity vector (*i.e.*, the squint angle is zero), we thus obtain the following two equations:

$$\cos\theta_i = \frac{\left(\Delta z^{(r)}/\Delta x\right)\sin\theta + \cos\theta}{\sqrt{\left(\Delta z^{(r)}/\Delta x\right)^2 + \left(\Delta z^{(a)}/\Delta y\right)^2 + 1}} \tag{4.17}$$

$$A = \sqrt{(\Delta y \Delta z^{(r)})^2 + (\Delta x \Delta z^{(a)})^2 + (\Delta x \Delta y)^2}. \tag{4.18}$$

Similar expressions have also been derived elsewhere [*Ulander,* 1996].

Guided by Eq. (4.15), we next examine the normalized cross section σ^0, for which a variety of models exist (see the texts by *Elachi* [1987] and *Ulaby et al.* [1981]). We adopt

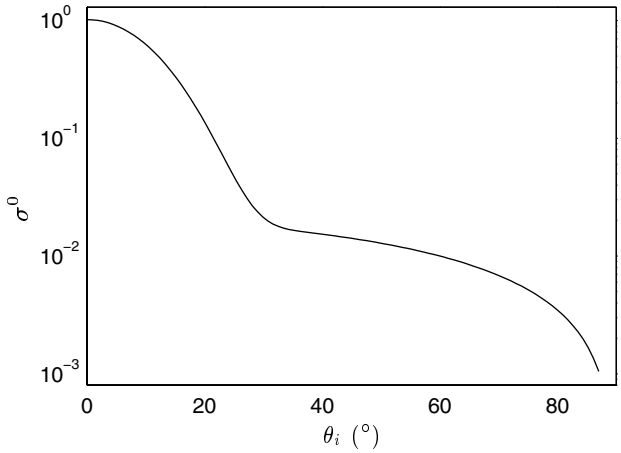

Figure 4.5 Dependence of model normalized cross section σ^0 on local signal incidence angle θ_i for $k_{ds} = 0.02$ and $n = 8$.

the model used by *Goering et al.* [1995] because of its ease of evaluation given Eq. (4.17):

$$\sigma^0 = \begin{cases} k_{ds} \cos^2 \theta_i + \cos^n 2\theta_i \cos \theta_i & \text{if } \cos 2\theta_i > 0, \\ k_{ds} \cos^2 \theta_i & \text{otherwise} \end{cases} . \tag{4.19}$$

Here, k_{ds} is a model parameter that determines the ratio of the two terms on the top line of the right side of Eq. (4.19); these terms can be thought of as diffuse and specular components of backscatter from the surface. The parameter n determines the sharpness of the specular peak with incidence angle. We have dropped a constant scaling factor from the model since we deal only with normalized intensity data as described above. An example curve of σ^0 from the model is plotted in Fig. 4.5.

Incorporating the scattering model of Eq. (4.19) and the viewing geometry relations of Eqs. (4.17) and (4.18) into Eq. (4.15), we arrive at a model for the SAR intensity as a function of topography. For the scattering model parameters, we use values of $k_{ds} = 0.02$ and $n = 8$, finding them to give good agreement between the real and simulated SAR intensity images shown in Fig. 4.6. The simulated image is generated from a digital elevation model (DEM), projected into radar coordinates, with only range components of slope used in our brightness model as explained below. Chapter 3 contains more detail about both the SAR data and the DEM. As evidenced by the agreement between the two images, the observed

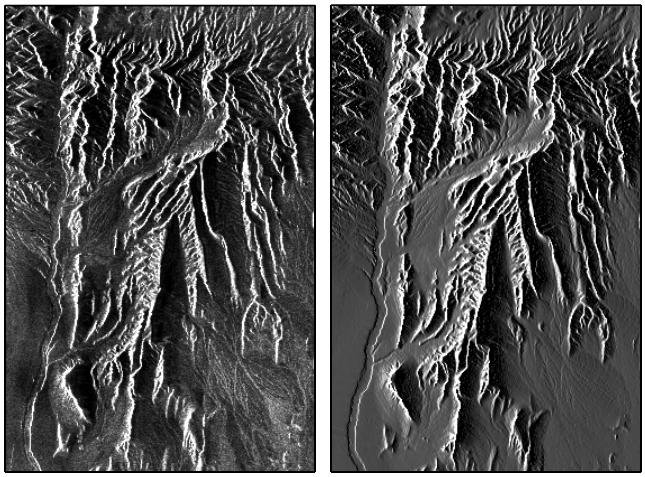

Figure 4.6 Comparison of (a) actual, normalized SAR image intensity with (b) simulated intensity from a DEM and scattering model. The similarity of the two images suggests that observed intensity information can be used to guide the topographic phase unwrapping process.

intensity is a valuable source of information about the surface topography and is fairly well reproduced with our model. The scattering model parameters may require adjustment for other terrain types, though.

In Fig. 4.7, we plot the expected image intensity as a function of the range slope $\Delta z^{(r)}$ for the case of zero slope in azimuth ($\Delta z^{(a)} = 0$). The curve is normalized so that a horizontal surface ($\Delta z^{(r)} = 0$) has unit expected intensity. The curve is symmetric about its zero for which the local signal incidence angle is 90°. This zero occurs when $\Delta z^{(r)} = \Delta z_0^{(r)}$ (see Fig. 4.8), where

$$\Delta z_0^{(r)} = -\Delta r \cos\theta. \tag{4.20}$$

Negative values of $\Delta z^{(r)}$ greater than $\Delta z_0^{(r)}$ correspond to surfaces sloping away from the radar, while positive values of $\Delta z^{(r)}$ correspond to surfaces sloping towards the radar. The intensity increases without bound as $\Delta z^{(r)}$ becomes large and the local incidence angle approaches zero ($\Delta z^{(r)} \to \infty$, $\Delta z^{(r)}/\Delta x \to \tan\theta$, $\cos\theta_i \to 1$) because a single range bin then encloses infinite area (*i.e.*, the facet model becomes invalid). A surface for which $\Delta z^{(r)} < \Delta z_0^{(r)}$ may be in either layover or shadow. Surfaces in shadow usually have negligible

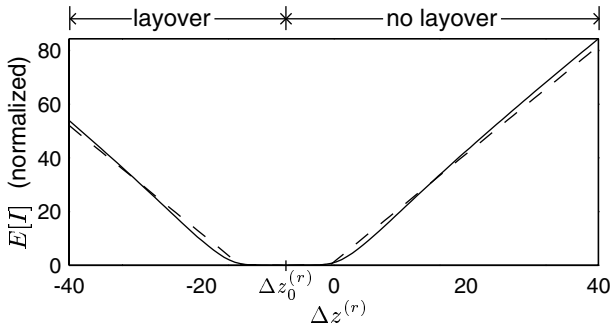

Figure 4.7 Model intensity as a function of slant-range elevation change for zero azimuth slope. The solid line represents the expected intensity $E[I]$ from Eq. (4.15); the dashed line is a piecewise linear approximation to the solid line.

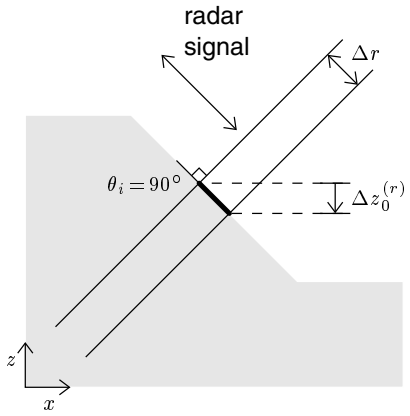

Figure 4.8 Normal incidence on the ground surface. We define $\Delta z_0^{(r)}$ to be the elevation change from one range bin to the next when the local incidence angle is 90°.

brightness, so, for the purpose of deducing topography from intensity, we assume that the surface is in layover when $\Delta z^{(r)} < \Delta z_0^{(r)}$. Thus, for example, if $\Delta z^{(r)} = -\Delta r / \cos\theta$, the surface is vertical, like the face of a cliff. Such a surface is physically sloped towards the radar, but the true elevation decreases as the slant range increases (see Fig. 4.9). As $\Delta z^{(r)}$ becomes infinitely negative, θ_i again approaches zero and $E[I]$ again becomes infinite.

Because variations in image intensity are much more dependent upon slopes in range than in azimuth, as described by *Guindon* [1990], we deal with range and azimuth gradients separately, beginning with the former. Qualitatively speaking, foreshortening effects from the SAR imaging geometry tend to make the range components of significant slopes much greater than the azimuth components. When the azimuth components are more significant than the range components, the total slope is often relatively low and presents little difficulty in phase unwrapping. Hence, at each pixel of the input image, with $\Delta z^{(a)} = 0$, we can invert the relation illustrated in Fig. 4.7 to find an expected slant-range slope from our computed value of $E[I]$. To facilitate this inversion, we further simplify our brightness model by using a piecewise-linear approximation to $E[I]$ (dashed line in Fig. 4.7).

If there is no layover, then $\Delta z^{(r)} > \Delta z_0^{(r)}$ and the observed intensity $E[I]$ corresponds uniquely to some expected local range slope $\Delta z_I^{(r)}$. This slope, the elevation change in range from one pixel to the next, can be related to the unwrapped gradient $\Delta\phi$ through Eq. (4.12). We therefore expect a peak in the PDF $f(\Delta\phi_{\text{topo}}^{(r)}|I)$ at $\Delta\phi_I$, the unwrapped gradient corresponding to the layover-absent expected slope $\Delta z_I^{(r)}$.

The PDF must also account for the possibility of layover, though, especially because layover often causes significant phase unwrapping errors. Where there is layover, however, the observed pixel intensity is not directly related to the desired slope. This is because multiple parts of the imaged surface fall into the same range bin, as illustrated in Fig. 4.10. A large phase discontinuity arises from the elevation jump between range bins r_0 and r_1, but the brightness of range bin r_1 does not indicate the magnitude of this discontinuity. Furthermore, range bin r_2 appears bright because it contains part of the front face of the mountain, yet its unwrapped phase is usually assumed to represent the elevation of the mountain's back face.

In order to model the effects of topography on intensity where there is layover, we note that the elevation change from range bin r_0 to r_1 in Fig. 4.10 is related to the full height of the illustrated mountain. We note also that the height of the mountain is equal to the integral of the slope along one of its faces. Therefore, because slope is related to brightness

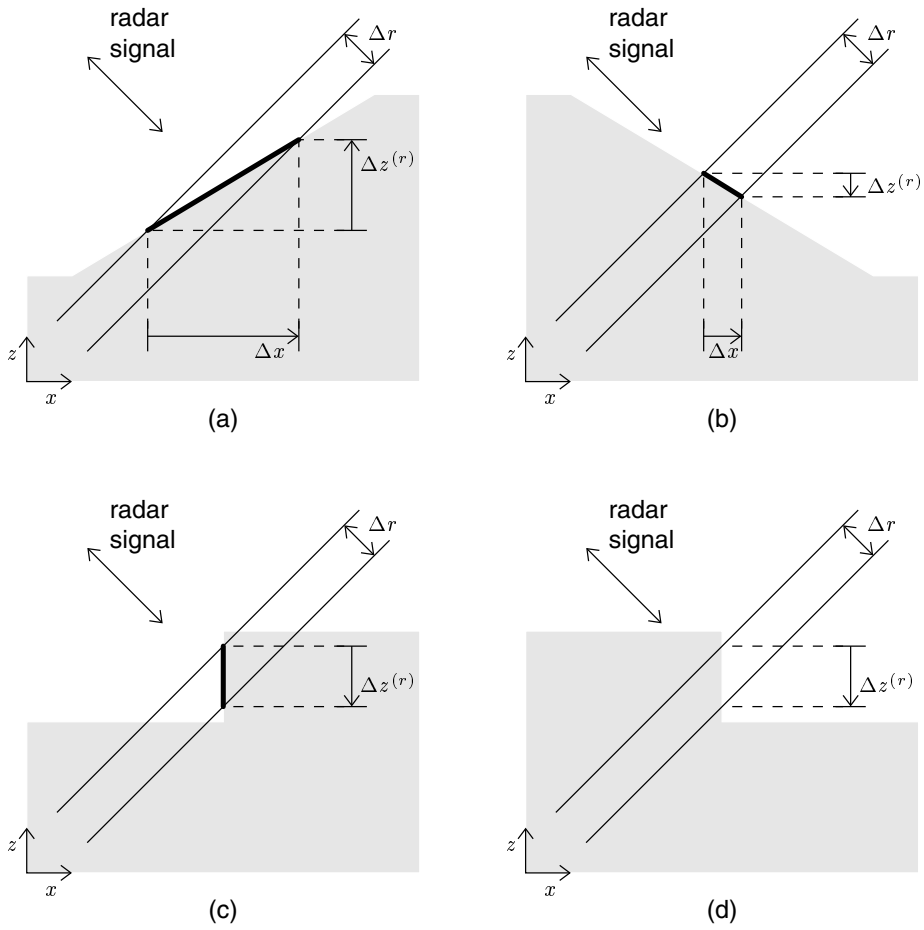

Figure 4.9 Regimes of the slant-range slope $\Delta z^{(r)}$: (a) the ground surface slopes towards the radar when $\Delta z^{(r)} > 0$; (b) the surface slopes away from the radar when $\Delta z_0^{(r)} < \Delta z^{(r)} < 0$; (c) the surface can be in layover when $\Delta z^{(r)} < \Delta z_0^{(r)}$; (d) the surface can also be in shadow when $\Delta z^{(r)} < \Delta z_0^{(r)}$. The part of the ground surface marked by the heavy line segment is mapped into the indicated range bin. The brightness model of Fig. 4.7 describes the contribution of this part of the surface to the range bin's expected intensity. In (c), other parts of the surface also map into the same range bin because of layover. In (d), the relevant part of the surface is in shadow and is hence not illuminated. Note that $\Delta z^{(r)}$ is identical for (c) and (d); we assume case (c) for this value of $\Delta z^{(r)}$.

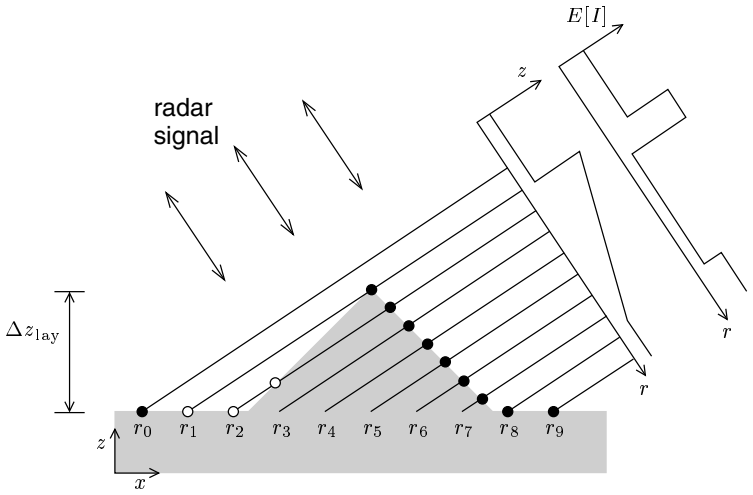

Figure 4.10 Profile of a mountain in layover. The range bins r_0–r_9 represent contours of constant range from the radar. The elevation z and mean intensity $E[I]$ are plotted as they map into slant range for this profile. Because of layover, multiple parts of a surface may map into the same range bin; the closed dots and open circles represent intersections of the ground surface with the range contours. Unwrapped phase values are usually assumed to represent elevations at the closed dots, but echoes from the locations of the open circles complicate the topography-intensity relationship where layover exists.

through Eq. (4.15), we can estimate the severity a layover discontinuity by examining the intensities of all range bins containing part of the face in layover (*e.g.*, range bins r_1 and r_2 in Fig. 4.10). We therefore compute Δz_{lay}, defined as the expected maximum elevation change of an unwrapped gradient straddling a layover discontinuity, for each pixel as follows. First, we compare the local intensity to a predetermined threshold value. If the intensity is below this threshold, layover is deemed unlikely. Otherwise, we convert the intensity to a layover-face range slope using the relation of Fig. 4.7 over the layover regime ($\Delta z^{(r)} < \Delta z_0^{(r)}$). We then integrate the similarly calculated, intensity-derived range slopes for the next several pixels in increasing range to obtain Δz_{lay}. The true elevation change may be less than Δz_{lay}, though, since some of the bright pixels contributing to the integrated value may actually correspond to slopes that are not in layover ($\Delta z^{(r)} > \Delta z_0^{(r)}$). Layover thus forces us to incorporate more features into our model for the unwrapped gradient PDF $f(\Delta\phi_{\text{topo}}^{(r)}|I)$. In addition to the peak at $\Delta\phi_I$, we expect the PDF to have a wide, platform-like section with an upper cutoff at $\Delta\phi_{\text{lay}}$, the unwrapped gradient corresponding to Δz_{lay}. This section reflects the probability that the local phase gradient $\Delta\phi_{\text{topo}}^{(r)}$ indeed straddles a layover discontinuity.

Layover causes further complications, however. A pixel whose unwrapped phase corresponds to a mild, negatively sloped back face, such as range bin r_2 in Fig. 4.10, may appear bright because it contains part of the face in layover. The true unwrapped gradient would then be unrelated to the computed values for $\Delta z_I^{(r)}$ and Δz_{lay}. Since a gradient straddling a layover discontinuity cannot be easily distinguished from the gradients in the bright area beyond the layover discontinuity, we must account for the probability of the latter in $f(\Delta\phi_{\text{topo}}^{(r)}|I)$. We therefore include in the PDF a peak at a slightly negative value near zero, denoted $\Delta\phi_{\text{back}}$ (the exact location of this peak is not important for now). The resulting model is shown qualitatively in Fig. 4.11(a). It is asymmetric because as range increases, an upwards step in elevation due to layover is usually much more likely than a downwards one.

We have thus far considered PDFs only for unwrapped gradients in range, yet gradients in azimuth are equally important for phase unwrapping though they may have little effect on intensity and correlation. The azimuth PDFs $f(\Delta\phi_{\text{topo}}^{(a)}|I)$ are shaped somewhat like the range PDFs $f(\Delta\phi_{\text{topo}}^{(r)}|I)$ since the physical mechanisms behind the two are similar. As illustrated in Fig. 4.11(b), however, the azimuth PDFs are two-sided and symmetric due to the azimuthal symmetry of the SAR viewing geometry. Where layover is not expected,

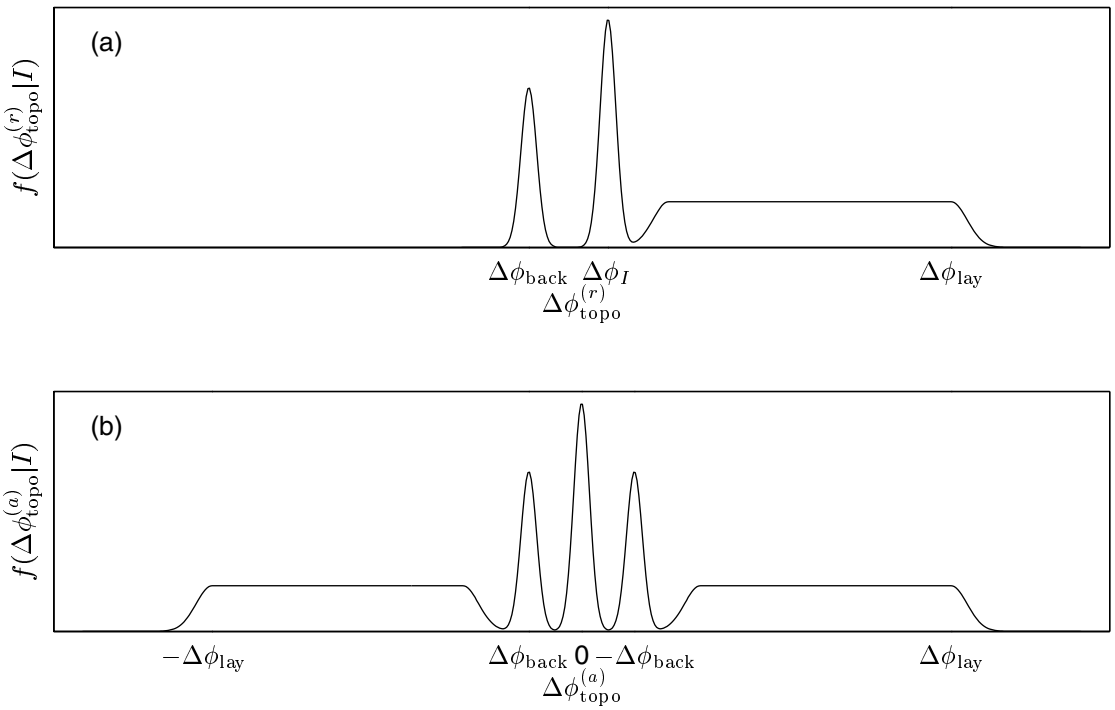

Figure 4.11 Model conditional PDFs $f(\Delta\phi_{\text{topo}}|I)$ for (a) unwrapped topographic range gradients $\Delta\phi_{\text{topo}}^{(r)}$ and (b) azimuth gradients $\Delta\phi_{\text{topo}}^{(a)}$.

the azimuth gradient PDF, like the range gradient PDF, consists only of a single peak. We assume that this peak is located at zero rather than at $\Delta\phi_I$, though, since the value of $\Delta\phi_I$ is determined mainly by the range component of the local slope (see above). Where layover is possible, the azimuth PDF contains both shelf-like regions and peaks at $\pm\Delta\phi_{\text{back}}$. These features, similar to those of the range PDF, arise because layover-induced discontinuities often contain azimuth as well as range components. The layover shelf cutoff $\Delta\phi_{\text{lay}}$ thus has the same value for azimuth as it does for range.

While such a PDF does suggest that some azimuth gradients may indeed be very large, it does not necessarily invalidate our assumption above of small azimuth gradients. This assumption holds as long as local range gradients are more significant than their azimuth counterparts. Moreover, as explained above, gradients that straddle layover discontinuities do not affect the intensity in proportion to their magnitude.

4.3.2 Correlation and Topography

Having a qualitative model for $f(\Delta\phi_{\text{topo}}|I)$, we now return to Eq. (4.14) and consider $f(\rho|\Delta\phi_{\text{topo}})$, the conditional PDF of the local coherence given the unwrapped gradient. While we have already described the relationship between the coherence and the phase noise (see Fig. 4.2), ρ is also related to the topographic part of the unwrapped gradient through spatial or baseline decorrelation. *Zebker and Villasenor* [1992] obtained an expression for this decorrelation factor, denoted ρ_s:

$$\rho_s = \max\left\{0, 1 - \frac{2\,|B_\perp|\,R_r}{\lambda r|\tan\theta_i|}\right\}. \tag{4.21}$$

Here, r is the slant range, R_r is the slant-range resolution, and we again assume that range slopes are much more significant than azimuth slopes. We also assume that the data have not been spectrally filtered in range [*Gatelli et al.,* 1994]. Thus, as the range slope increases and the local incidence angle decreases, the correlation decreases as well.

In the presence of layover, however, several different parts of the surface map into the same pixel, and the multiplicity of local incidence angles makes Eq. (4.21) inapplicable. Correlation measures for areas in layover are generally very low, though, because of actual decorrelation as well as the random complex superposition of the different signals contributing to the interferometric phase. We assume that the observed correlation will be zero in these areas. Similar assumptions are made implicitly in phase unwrapping schemes that

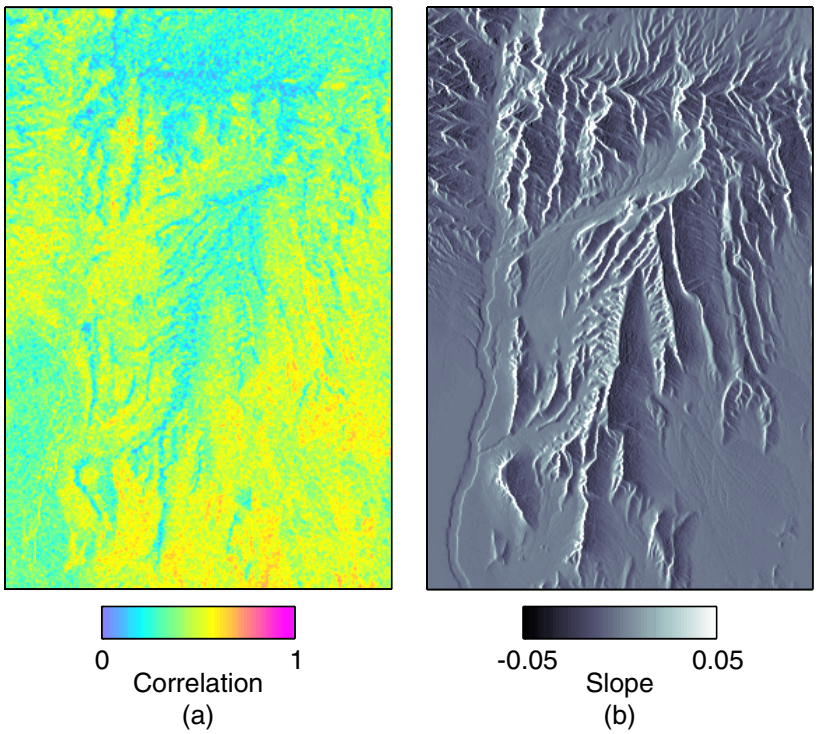

Figure 4.12 Comparison of (a) an interferogram correlation image with (b) a DEM-derived range-slope image.

use correlation information alone for weighting L^p cost functions [*Pritt,* 1996; *Ghiglia and Pritt,* 1998].

Figure 4.12 depicts the biased correlation image and a DEM-derived topographic slope map for the Death Valley data shown in Chapter 3. The slope map, in radar coordinates, shows only the components of slope in the range direction. Clearly, areas of layover and steep positive slopes are characterized by significant decorrelation, illustrating the relationship between ρ and $\Delta z^{(r)}$. Decorrelation in other parts of the interferogram are likely due to topography-independent temporal effects and thermal noise [*Zebker and Villasenor,* 1992].

Combining sources of decorrelation other than spatial effects into a single variable ρ_{other},

$$\rho = \rho_s \rho_{\text{other}}. \tag{4.22}$$

By definition, ρ, ρ_s, and ρ_{other} are all between zero and one, so ρ_s is an upper bound on

the expected statistical correlation ρ. Thus, for a fixed, observed value of ρ, the function $f(\rho|\Delta\phi^{(r)}_{\text{topo}})$ in terms of $\Delta\phi^{(r)}_{\text{topo}}$ has an upper cutoff at $\Delta\phi^{(r)}_{\text{topo}} = \Delta\phi_\rho$, the unwrapped gradient corresponding to a local range slope for which $\rho = \rho_s$. In other words, the observed correlation places an upper limit on the expected range slope; as the correlation becomes larger, the maximum expected slope becomes smaller. Quantitative values for $\Delta\phi_\rho$ may be computed by numerically inverting Eq. (4.21), with the range slope determined from θ_i through Eq. (4.17).

The correlation-induced lower limit of the local range slope is $\Delta z^{(r)}_0$ if $\rho > 0$. Slopes any more negative would be in shadow and would therefore imply noise-only data with zero correlation. We define $\Delta\phi^{(r)}_0$ to be the unwrapped range gradient corresponding to a slope of $\Delta z^{(r)}_0$.

Without more specific knowledge of other decorrelation sources, we assume that for a fixed, nonzero value of ρ,

$$f(\rho|\Delta\phi^{(r)}_{\text{topo}}) = \begin{cases} C^{(r)} & \text{if } \Delta\phi^{(r)}_0 < \Delta\phi^{(r)}_{\text{topo}} \leq \Delta\phi_\rho \\ 0 & \text{otherwise} \end{cases} \tag{4.23}$$

for some constant $C^{(r)}$. At locations in the interferogram where $\rho = 0$, we assume that $f(\rho|\Delta\phi^{(r)}_{\text{topo}})$ is constant for all possible $\Delta\phi^{(r)}_{\text{topo}}$.

Under the assumption that azimuth gradients are much smaller than range gradients, the upper cutoff $\Delta\phi_\rho$, as calculated for the local range gradient, also serves as an upper cutoff for the azimuth expression $f(\rho|\Delta\phi^{(a)}_{\text{topo}})$ as a function of $\Delta\phi^{(a)}_{\text{topo}}$. Because of azimuthal symmetry, $-\Delta\phi_\rho$ is this function's lower cutoff. Thus, for a fixed, nonzero value of ρ,

$$f(\rho|\Delta\phi^{(a)}_{\text{topo}}) = \begin{cases} C^{(a)} & \text{if } -\Delta\phi_\rho \leq \Delta\phi_{\text{topo}} \leq \Delta\phi_\rho \\ 0 & \text{otherwise} \end{cases} \tag{4.24}$$

where $C^{(a)}$ is another constant. Both $f(\rho|\Delta\phi^{(r)}_{\text{topo}})$ and $f(\rho|\Delta\phi^{(a)}_{\text{topo}})$ are plotted in Fig. 4.13.

4.3.3 Topographic PDFs and Cost Functions

Guided by Eq. (4.14), we may now combine the two PDFs $f(\Delta\phi_{\text{topo}}|I)$ and $f(\rho|\Delta\phi_{\text{topo}})$ relating the local unwrapped gradient to the observed intensity and correlation. The resulting

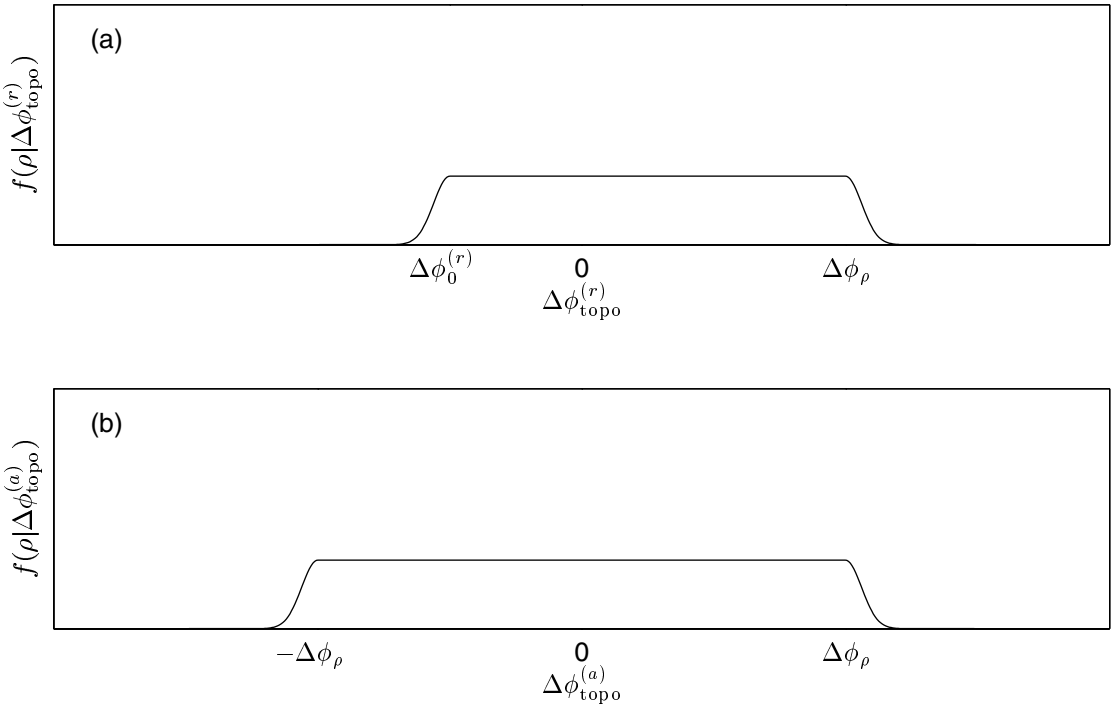

Figure 4.13 Model conditional PDFs $f(\rho|\Delta\phi_{\text{topo}})$ for fixed, nonzero ρ as functions of (a) the unwrapped topographic range gradients $\Delta\phi_{\text{topo}}^{(r)}$ and (b) azimuth gradients $\Delta\phi_{\text{topo}}^{(a)}$. Note that the curves shown, because they are functions of $\Delta\phi_{\text{topo}}$ rather than ρ, are not PDFs themselves.

PDF $f(\Delta\phi_{\text{topo}}|I,\rho)$ describes the topographic signal component of a particular unwrapped gradient. Together with the shape of the corresponding noise PDF $f(\Delta\phi_{\text{noise}}|\rho)$, the shape of $f(\Delta\phi_{\text{topo}}|I,\rho)$ determines the shape of the full PDF $f(\Delta\phi|I,\rho)$. From models for these PDFs, we can proceed to form our MAP cost functions through Eq. (4.6).

Equation (4.14) states that the combined, conditional topographic PDF $f(\Delta\phi_{\text{topo}}|I,\rho)$ is proportional to the product of $f(\Delta\phi_{\text{topo}}|I)$ and $f(\rho|\Delta\phi_{\text{topo}})$. As shown in Fig. 4.13, $f(\rho|\Delta\phi_{\text{topo}})$ is, for the most part, either zero or a constant, so our model for $f(\Delta\phi_{\text{topo}}|I,\rho)$ has a shape similar to that of $f(\Delta\phi_{\text{topo}}|I)$. However, the shelf cutoff $\Delta\phi_{\max}$ of $f(\Delta\phi_{\text{topo}}|I,\rho)$ is determined by the lesser of the intensity-derived cutoff $\Delta\phi_{\text{lay}}$ and the correlation-derived cutoff $\Delta\phi_{\rho}$:

$$\Delta\phi_{\max} = \min\left\{\Delta\phi_{\text{lay}}, \Delta\phi_{\rho}\right\}. \tag{4.25}$$

Figures 4.14 and. 4.15 illustrate $f(\Delta\phi_{\text{topo}}|I,\rho)$ for range gradients and azimuth gradients.

With the topographic PDF modeled, we now form our full conditional PDFs and MAP cost functions through Eqs. (4.9) and (4.6). We begin by noting that the sharp peaks of $f(\Delta\phi_{\text{topo}}|I,\rho)$ illustrated in Figs. 4.14(b) and 4.15(b) are likely much narrower and much closer together than the width of the Gaussian noise PDF with which they are convolved. Consequently, the full PDF $f(\Delta\phi|I,\rho)$ is characterized mainly by the width of the noise PDF and the critical unwrapped-gradient values $\Delta\phi_{\text{back}}$, $\Delta\phi_I$ and $\Delta\phi_{\max}$. We hence simplify our model by assuming that after convolution with the noise PDF, the narrow peaks at 0, $\Delta\phi_{\text{back}}$, and $\Delta\phi_I$ can be modeled by a single, wide hump, as shown in Figs. 4.14(c) and 4.15(c). That is, the convolution removes much of the apparent structure from $f(\Delta\phi_{\text{topo}}|I,\rho)$, resulting in the simpler PDF $f(\Delta\phi|I,\rho)$ which has fewer parameters. This simplification is important in implementation because an independent set of parameters, computed from the local observables, must be maintained in computer memory for each phase difference in the interferogram.

Through Eq. (4.6), the negative logarithm of $f(\Delta\phi|I,\rho)$ is the continuous-valued cost function $g(\Delta\phi)$, which we can evaluate at integer-cycle offsets from $\Delta\psi$ to obtain discrete gradient costs (*i.e.*, flow costs) for our topographic phase unwrapping problem. We now model these cost functions, considering range gradients $\Delta\phi^{(r)}$ first.

When $\Delta\phi_{\max}$ is small compared to the phase-noise variance, layover is unlikely. Our model PDF $f(\Delta\phi^{(r)}|I,\rho)$ is then Gaussian with mean $\Delta\phi_I$, the phase value suggested by

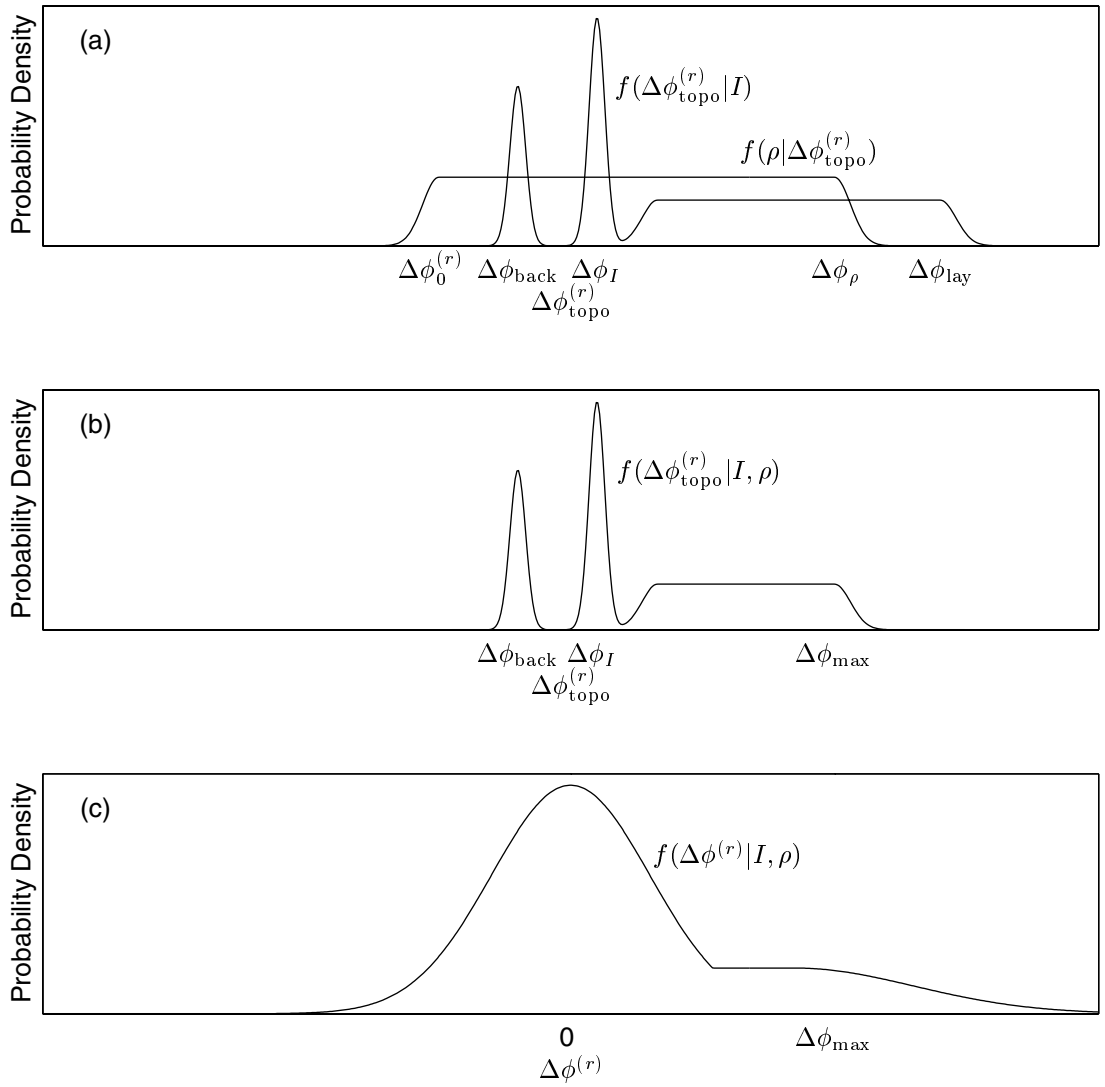

Figure 4.14 Model PDFs for unwrapped range gradients in topographic applications. The PDFs $f(\Delta\phi^{(r)}_{\text{topo}}|I)$ and $f(\rho|\Delta\phi^{(r)}_{\text{topo}})$ are shown overlaid in (a); their product leads to the combined topographic-term PDF $f(\Delta\phi^{(r)}_{\text{topo}}|I,\rho)$ in (b). The final PDF $f(\Delta\phi^{(r)}|I,\rho)$ in (c), which contains both signal and noise terms, results from the convolution of the topographic-term PDF with a Gaussian noise PDF. The labeled model parameters depend on the local observables and vary throughout the interferogram.

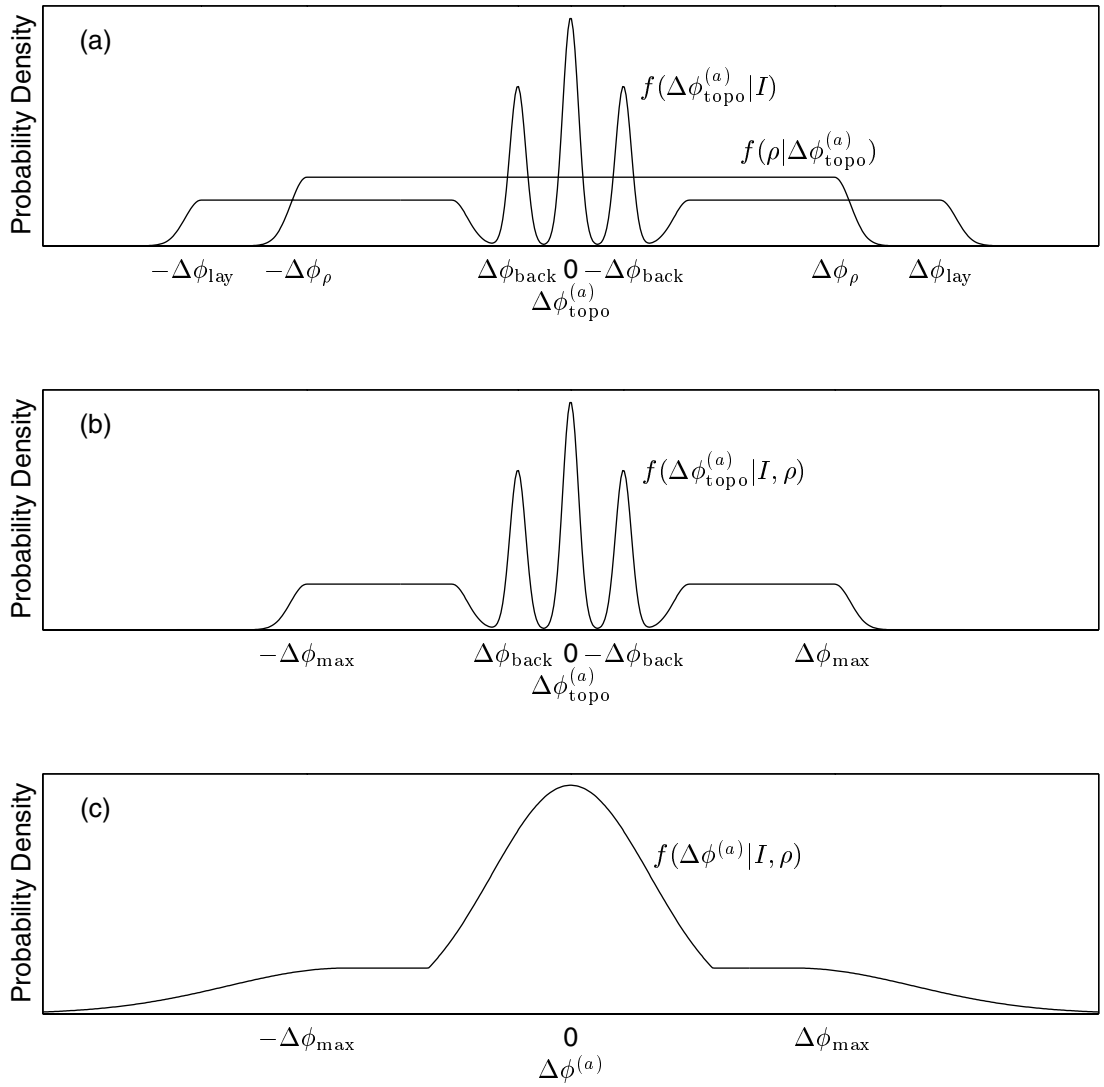

Figure 4.15 Model PDFs for unwrapped azimuth gradients in topographic applications. The PDFs $f(\Delta\phi^{(a)}_{\text{topo}}|I)$ and $f(\rho|\Delta\phi^{(a)}_{\text{topo}})$ are shown overlaid in (a); their product leads to the combined topographic-term PDF $f(\Delta\phi^{(a)}_{\text{topo}}|I,\rho)$ in (b). The final PDF $f(\Delta\phi^{(a)}|I,\rho)$ in (c), which contains both signal and noise terms, results from the convolution of the topographic-term PDF with a Gaussian noise PDF. The labeled model parameters depend on the local observables and vary throughout the interferogram.

the single-pixel intensity. The cost function $g^{(r)}(\Delta\phi)$ is therefore a parabola centered on $\Delta\phi_I$. The width of this parabola is the PDF standard deviation $\sigma_{\Delta\phi}$. Since variances add,

$$\sigma^2_{\Delta\phi} = \sigma^2_{\Delta\phi_{\text{noise}}} + \sigma^2_{\text{meas}}. \tag{4.26}$$

The first term on the right-hand side represents the phase-noise variance calculated from the measured correlation (Fig. 4.2). We implement the second term as a constant representing uncertainty in our estimates of $E[I]$ and ρ.

In parts of the interferogram where the upper cutoff $\Delta\phi_{\max}$ is large compared to the phase-noise variance, the cost function must include a shelf-like region that accounts for the probability of layover. This region extends out to $\Delta\phi_{\max}$ for positive gradient values, and its cost is based on the conditional probability of layover. In our implementation, we assume a constant, empirically derived layover cost $g^{(r)}_{\text{lay}}$. Because backface slopes are also likely near layover, the central parabola of our layover cost function is centered on $\Delta\phi = 0$ instead of $\Delta\phi = \Delta\phi_I$, and $\sigma^2_{\Delta\phi}$ includes an extra term σ^2_{lay} :

$$\sigma^2_{\Delta\phi} = \sigma^2_{\Delta\phi_{\text{noise}}} + \sigma^2_{\text{meas}} + \sigma^2_{\text{lay}}. \tag{4.27}$$

The new term represents our uncertainty in the location of the true peak because of layover. Beyond $\Delta\phi_{\max}$, the cost function increases quadratically at a rate related to $\sigma_{\Delta\phi}$.

Thus, the cost function for a particular range gradient is illustrated in Fig. 4.16 and described by parameters computed from the local observables as summarized in Table 4.1. We quantify these cost functions with the following parameterized model. Let

$$\Delta\phi^{(r)}_{\text{crit}} = \sqrt{g^{(r)}_{\text{lay}}\sigma^2_{\Delta\phi}} \tag{4.28}$$

with $\sigma^2_{\Delta\phi}$ calculated from Eq. (4.27). If $\Delta\phi_{\max} \geq \Delta\phi^{(r)}_{\text{crit}}$,

$$g^{(r)}(\Delta\phi) = \begin{cases} \frac{\Delta\phi^2}{\sigma^2_{\Delta\phi}} & \text{if } \Delta\phi \leq \Delta\phi^{(r)}_{\text{crit}}, \\ g^{(r)}_{\text{lay}} & \text{if } \Delta\phi^{(r)}_{\text{crit}} < \Delta\phi \leq \Delta\phi_{\max}, \\ \frac{(\Delta\phi - \Delta\phi_{\max})^2}{C_{\text{tail}}\sigma^2_{\Delta\phi}} + g^{(r)}_{\text{lay}} & \text{if } \Delta\phi > \Delta\phi_{\max} \end{cases} \tag{4.29}$$

where C_{tail} is an empirically derived constant. Conversely, if $\Delta\phi_{\max} < \Delta\phi^{(r)}_{\text{crit}}$, we define

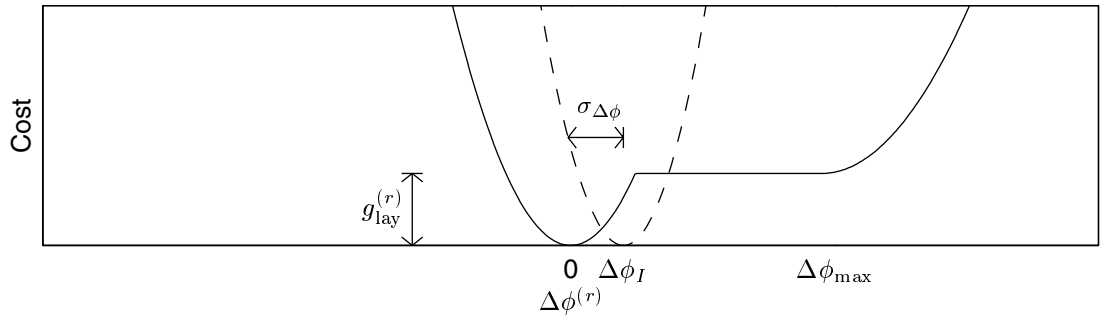

Figure 4.16 Example cost functions for topographic range gradients. In parts of the interferogram where $\Delta\phi_{\max}$ is large, layover is expected and the cost functions have shelf-like regions (solid line). Where $\Delta\phi_{\max}$ is small, the cost functions consist only of parabolas (dashed line).

Parameter	Predominant Physical Source
$\sigma_{\Delta\phi}$	ρ
$\Delta\phi_{\max}$	integrated $E[I]$, ρ
$\Delta\phi_I$	single-pixel $E[I]$
g_{lay}	conditional layover probability

Table 4.1 Predominant physical sources of the topographic cost-function parameters shown in Figs. 4.16 and 4.17.

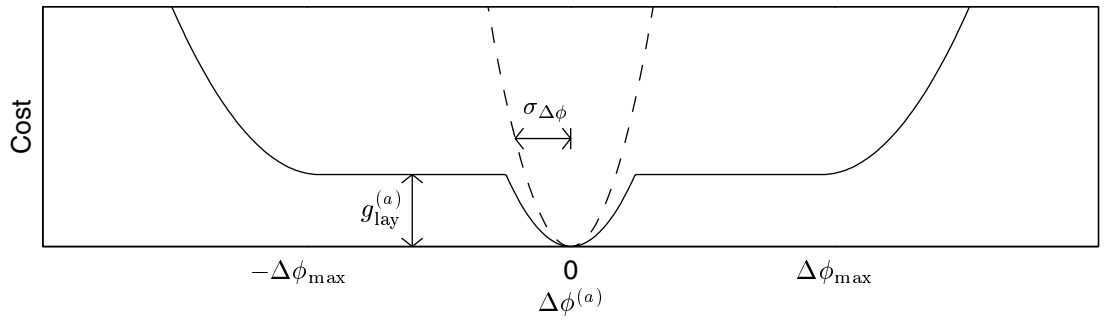

Figure 4.17 Example cost functions for topographic azimuth gradients. In parts of the interferogram where $\Delta\phi_{\max}$ is large, layover is expected and the cost functions have shelf-like regions (solid line). Where $\Delta\phi_{\max}$ is small, the cost functions consist only of parabolas (dashed line). Unlike the range cost functions of Fig. 4.16, the azimuth cost functions are symmetric about $\Delta\phi^{(a)} = 0$.

$\sigma_{\Delta\phi}^2$ according to Eq. (4.26) and assume

$$g^{(r)}(\Delta\phi) = \frac{(\Delta\phi - \Delta\phi_I)^2}{\sigma_{\Delta\phi}^2}. \tag{4.30}$$

Our azimuth cost functions may be derived in an analogous manner and are also described by the parameters listed in Table 4.1. Because both range and azimuth cost functions are determined by the same physical mechanisms, their shapes are very similar. The parameter $\sigma_{\Delta\phi}$ is the same for both because the same noise processes affect both. The parameter $\Delta\phi_{\max}$ is also the same since the parameters $\Delta\phi_{\text{lay}}$ and $\Delta\phi_\rho$ are common to both, as described above. The layover-discontinuity shelf cost $g_{\text{lay}}^{(a)}$ for azimuth may be greater than $g_{\text{lay}}^{(r)}$, though, as discontinuities are more likely across range gradients.

The most important difference between range and azimuth cost functions, however, is the two-sided nature of the azimuth cost functions (see Fig. 4.17). This symmetry follows from the symmetry of the azimuth PDFs illustrated in Fig. 4.15. We thus quantify the azimuth cost functions by letting

$$\Delta\phi_{\text{crit}}^{(a)} = \sqrt{g_{\text{lay}}^{(a)}\sigma_{\Delta\phi}^2} \tag{4.31}$$

with $\sigma^2_{\Delta\phi}$ calculated from Eq. (4.27). If $\Delta\phi_{\max} \geq \Delta\phi^{(a)}_{\text{crit}}$,

$$g^{(a)}(\Delta\phi) = \begin{cases} \frac{\Delta\phi^2}{\sigma^2_{\Delta\phi}} & \text{if } |\Delta\phi| \leq \Delta\phi^{(a)}_{\text{crit}}, \\ g^{(a)}_{\text{lay}} & \text{if } \Delta\phi^{(a)}_{\text{crit}} < |\Delta\phi| \leq \Delta\phi_{\max}, \\ \frac{(|\Delta\phi| - \Delta\phi_{\max})^2}{C_{\text{tail}}\sigma^2_{\Delta\phi}} + g^{(a)}_{\text{lay}} & \text{if } |\Delta\phi| > \Delta\phi_{\max}. \end{cases} \tag{4.32}$$

If $\Delta\phi_{\max} < \Delta\phi^{(a)}_{\text{crit}}$, layover-induced discontinuities are unlikely and

$$g^{(a)}(\Delta\phi) = \frac{\Delta\phi^2}{\sigma^2_{\Delta\phi}} \tag{4.33}$$

where $\sigma^2_{\Delta\phi}$ is calculated from Eq. (4.26).

These MAP cost functions provide a physical framework for the topographic phase unwrapping optimization problem, but the framework is built upon application-specific assumptions. Consequently, in aiming to increase unwrapping accuracy for one particular type of interferometric measurement, we sacrifice general applicability to others. We must therefore develop additional cost functions matched to the other specific applications in which we are interested.

4.4 Deformation Measurements

SAR interferometry for deformation studies is closely related to the topographic application described in the previous section, but the two entail different phase statistics and therefore require different sets of cost functions. In deformation-mapping applications, the unwrapped interferometric phase measures earth surface displacement that may be due to phenomena such as earthquakes, volcanism, or glacial flow. Deformation interferograms often contain fewer abrupt changes in unwrapped phase than topographic interferograms, but very large discontinuities are still possible because of surface fracture and shear. Moreover, while low fringe rates may sometimes make deformation interferograms relatively easy to unwrap away from these discontinuities, the areas near discontinuities are often of most interest.

Our derivation of deformation cost functions again begins from Eq. (4.9); we must now model $f(\Delta\phi_{\text{signal}}|I,\rho)$ with the interferometric signal representing a measurement of surface displacement. That is, we define $\phi_{\text{defo}} = \phi_{\text{signal}}$ and $\Delta\phi_{\text{defo}} = \Delta\phi_{\text{signal}}$. The deformation

phase signature is given by

$$\phi_{\text{defo}} = \frac{-4\pi}{\lambda} d_r \tag{4.34}$$

where d_r is the surface displacement parallel to the radar line of sight—other components of displacement are not measurable with this technique—and λ is again the SAR wavelength [*Gabriel et al.,* 1989].

Unwrapped deformation gradients $\Delta\phi_{\text{defo}}$ do not depend on intensity in any obvious way. Although there is sometimes correspondence between topographic features and areas of large deformation changes, this is not always the case. Moreover, while deformation-induced decorrelation may lessen an interferogram's magnitude, it has no effect on the intensities of the individual SAR images. Consequently, without further insight into the expected deformation pattern, we assume that the deformation signal is independent of the intensity:

$$f(\Delta\phi_{\text{defo}}|I,\rho) = f(\Delta\phi_{\text{defo}}|\rho). \tag{4.35}$$

On the other hand, the relationship between deformation and correlation is much closer, as demonstrated by the interferogram and correlation map of Fig. 4.18 (see Chapter 7 for more details about this interferogram). The interferogram phase measures surface displacement along an earthquake fault, near which large phase gradients arise. Because the surface moves in opposite directions on either side of the fault, the fault itself coincides with a severe phase discontinuity. Thus, as suggested by the plain visibility of the fault in the correlation image, correlation can play an important role in unwrapping deformation interferograms.

Large displacement changes imply distortions of the surface, so they are likely accompanied by significant decorrelation through such mechanisms as temporal scattering changes [*Zebker and Villasenor,* 1992] and local misregistration [*Just and Bamler,* 1994]. We therefore account for the probability of a discontinuity where the correlation is low by using the following simple model. In parts of the interferogram where the correlation falls below a preselected threshold, the local PDFs $f(\Delta\phi_{\text{defo}}|\rho)$ contain shelf-like regions, similar to those of our topographic azimuth PDFs (see Fig. 4.15). Where the correlation is high, the PDFs consist only of single narrow peaks.

Combining the phase-noise PDFs and the unwrapped-gradient PDFs as in Eq. (4.9), the deformation cost functions are thus shaped as illustrated in Fig. 4.19, where now both

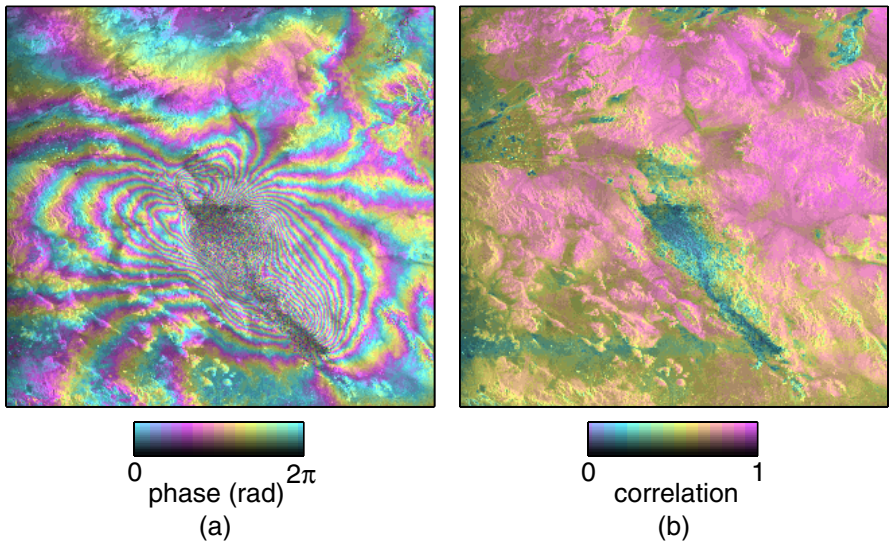

Figure 4.18 Comparison of (a) a deformation interferogram with (b) its correlation image. Areas close to the fault trace are associated with low correlation. Both panels show the interferogram magnitude in gray-scale brightness.

range and azimuth cost functions have identical shapes.

These parameterized deformation cost functions are quantified analogously to our topographic azimuth cost functions of Eqs. (4.32) and (4.33), although different physical quantities are used for the deformation model parameters. Specifically, if the local correlation exceeds some threshold $\rho_{\min}$, the cost function $g(\cdot)$ at that location consists only of a parabola—the negative logarithm of the Gaussian noise PDF—centered at zero:

$$g(\Delta\phi) = \frac{\Delta\phi^2}{\sigma^2_{\Delta\phi}}. \tag{4.36}$$

Parameter	Predominant Physical Source
$\sigma_{\Delta\phi}$	ρ
$\Delta\phi_{\max}$	assumed maximum discontinuity
g_d	conditional discontinuity probability

Table 4.2 Predominant physical sources of the deformation cost-function parameters shown in Fig. 4.19.

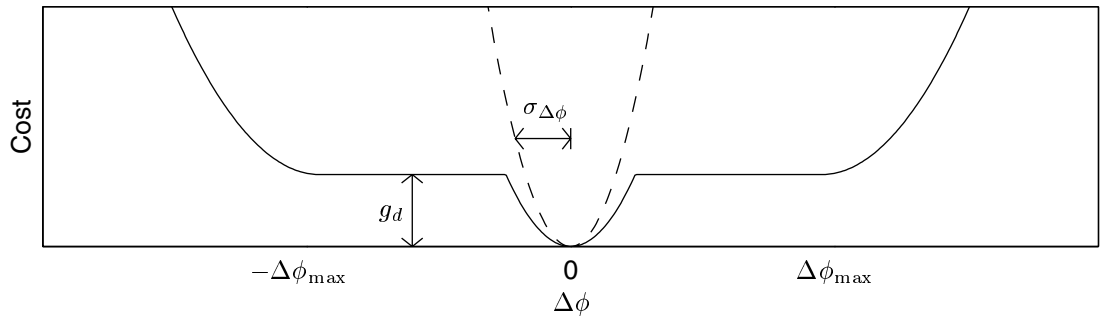

Figure 4.19 Example cost functions for deformation phase gradients. Where $\rho < \rho_{\min}$, the cost functions have shelf-like regions (solid line). Otherwise, the cost functions consist only of parabolas (dashed line).

Where the correlation falls below $\rho_{\min}$, the cost function also includes shelf-like regions representing the probability of a discontinuity in the unwrapped phase. The heights of these shelves, denoted g_d, are related to the local conditional probability of a phase discontinuity. The maximum probable magnitude of this discontinuity determines $\Delta\phi_{\max}$, the upper shelf cutoff. Thus, for $\rho < \rho_{\min}$,

$$g(\Delta\phi) = \begin{cases} \frac{\Delta\phi^2}{\sigma^2_{\Delta\phi}} & \text{if } |\Delta\phi| \leq \Delta\phi_{\text{crit}}, \\ g_d & \text{if } \Delta\phi_{\text{crit}} < |\Delta\phi| \leq \Delta\phi_{\max}, \\ \frac{(|\Delta\phi| - \Delta\phi_{\max})^2}{C_{\text{tail}}\sigma^2_{\Delta\phi}} + g_d & \text{if } |\Delta\phi| > \Delta\phi_{\max} \end{cases} \tag{4.37}$$

where

$$\Delta\phi_{\text{crit}} = \sqrt{g_d \sigma^2_{\Delta\phi}} \tag{4.38}$$

and C_{tail} is again a constant. In all cases, $\sigma^2_{\Delta\phi}$ is calculated from Eq. (4.26). Without more specific information, we use empirically or experimentally derived constants for $\rho_{\min}$, g_d, and $\Delta\phi_{\max}$. For the results in Chapter 7, these parameters were initially estimated through visual inspection of the interferogram, then refined with further applications of the algorithm.

4.5 Statistical Cost Functions and L^p Objectives

In parts of the interferogram where discontinuities are not expected, our MAP cost functions consist simply of parabolas whose widths are determined by the local correlation (see Figs. 4.16, 4.17, and 4.19). The cost functions are thus sometimes similar in shape to weighted least-squares (L^2) cost functions. On the other hand, where discontinuities are possible (*i.e.*, where $\Delta\phi_{\max} > \Delta\phi_{\text{crit}}$ for topography and $\rho < \rho_{\min}$ for deformation), our cost functions are somewhat similar to L^0 cost functions. Our MAP approach therefore receives some validation from both the successes of existing least-squares algorithms on noisy interferograms that lack discontinuities as well as the successes of L^0 algorithms on high-coherence interferograms that contain significant physical discontinuities.

Note, however, that while L^p cost functions are always minimal when $\Delta\phi = \Delta\psi$, our cost functions impose no such restriction and need not be centered on the wrapped gradients. Recall also that we apply the constraint of congruence during rather than after optimization. Thus, having modeled the statistics of a particular unwrapped gradient, we might expect the likelihood of an extra cycle of phase to be much higher in one direction than the other. Consider the case of a wrapped topographic gradient $\Delta\psi = 0.4$ cycles. If our observations suggest that the mean topographic slope is near zero ($\Delta\phi_{\text{topo}} \approx 0$), the true unwrapped gradient $\Delta\phi$ is much more likely to have been wrapped from -0.6 than from 1.4 cycles. Moreover, although above we use cost functions centered at either $\Delta\phi = \Delta\phi_I$ or $\Delta\phi = 0$, more elaborate models for the cost-function minima are possible; they might, for example, be determined by averaging wrapped gradients over local neighborhoods.

Our generalized cost functions thus treat unwrapped phase gradients as random draws from predetermined probability distributions. As the observed data vary from pixel to pixel throughout the interferogram, the shapes of these distributions vary as well. Hence, our cost functions, unlike L^p cost functions, are not just scaled, they have entirely different forms throughout the interferogram.

In this light, L^p cost functions can be viewed as coarse approximations to more-specific MAP objectives. The former might consequently benefit from weighting schemes based on the statistical models described in this chapter. As explained in Chapter 3, particular solver requirements might motivate the use of such statistically weighted L^p cost functions in certain applications. If our nonlinear MAP cost functions are to be minimized directly,

however, more powerful solver techniques are needed. We describe solver techniques based on network-flow ideas in the next chapter.

Chapter 5

Network-Flow Optimization Techniques

As discussed in Chapter 3, a phase unwrapping optimization algorithm entails not only setting up the problem, but solving it as well. While the statistical cost functions of Chapter 4 provide a means of evaluating and comparing different unwrapped solutions—setting up the problem—they do not suggest procedures for computing high-quality solutions—solving the problem. We describe solver techniques based on network optimization in this chapter. We briefly review relevant ideas from network theory as necessary, but the reader is directed to texts such as the one by *Ahuja et al.* [1993] for more detail on this subject.

Network concepts are used in a wide variety of fields because of their generality and applicability to different types of problems. As the phase unwrapping problem is elegantly described by a network model (see Chapter 2), we draw upon the rich and well-developed ideas of network theory here to derive a new nonlinear solver algorithm. In order to introduce this algorithm more easily, however, we first describe two new minimum L^0-norm solvers which shed light upon the underlying network structure of the phase unwrapping problem. Next, using these L^0 methods as stepping stones, we propose a simple but slow algorithm for the generalized optimization problem of Eq. (3.4), wherein cost functions may have arbitrary shapes. Then, combining our preliminary algorithms with several speed-enhancing techniques, we derive finally an efficient algorithm whose use allows us to solve, approximately, the MAP estimation problem posed in the previous chapter.

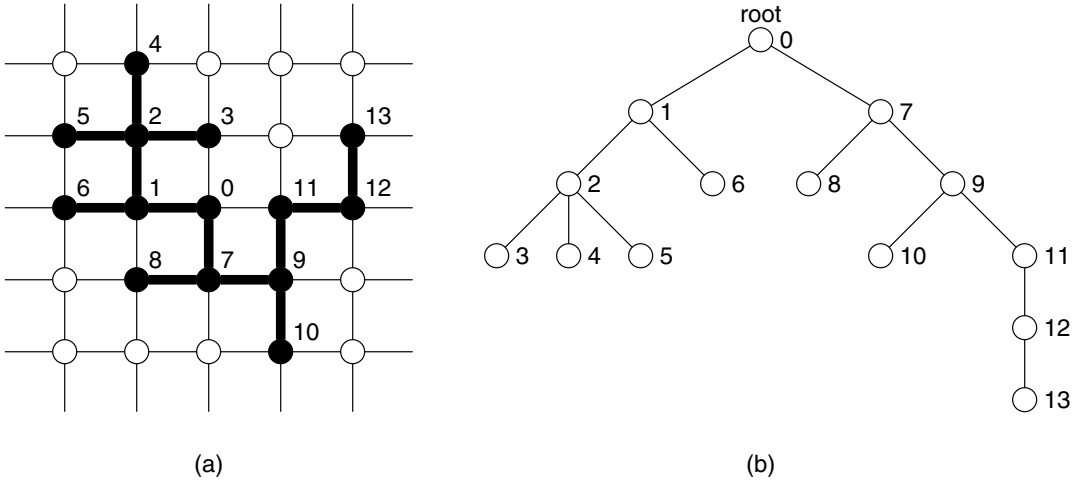

Figure 5.1 Representations of trees: (a) a tree in a grid network; (b) its schematic layout. Nodes and arcs in heavy black in (a) are on the tree, while other nodes and arcs are not. Node 0 is assumed to be the tree root in (b), although other tree representations are possible as well. Node 0 is the parent of nodes 1 and 7, node 1 is the parent of nodes 2 and 6, and so on.

5.1 The Minimum Spanning Tree Algorithm

Among existing phase unwrapping techniques, the residue-cut algorithm (see Chapter 2) is quite popular because of its speed and relative accuracy. It is severely limited, however, by its frequent inability to generate complete solutions. In this section, we use the principles of network theory to develop an improved version of the residue-cut algorithm that guarantees complete coverage and also allows user-defined weights for greater accuracy.

Goldstein et al. [1988] did not describe the original residue-cut algorithm explicitly in the language of networks, but one concept central to the algorithm's function is that of the tree. Trees, which are also fundamental to the other algorithms in this chapter, are connected sets of nodes and arcs which do not contain cycles, or closed loops of arcs. Thus, with the exception of a root node from which the tree can be thought to hang, all nodes on the tree have a parent or predecessor node that can be reached through a unique arc on the tree (see Fig. 5.1). The root node is therefore eventually reached if all of a node's predecessors are followed back up the tree. Furthermore, tree traversal yields a unique path between any two nodes on the tree, although a tree need not contain all nodes in the network. We define a tree branch or a subtree as any connected subset of the tree.

The original residue-cut algorithm begins with the identification of some nonzero residue, or charge, in the wrapped data. A cut is then drawn from this charge, through intermediate nodes, to the next nearest charge (regardless of sign), forming a tree. If the tree is not neutral, that is, if the tree contains unequal numbers of positive and negative residues, a cut is drawn to the next nearest charge, expanding the tree. The process continues until the tree becomes neutral, at which point some other unbalanced charge becomes the root of a new tree. When all charges are on neutral trees, the phase is integrated in a flood-fill fashion along some path such that no cuts (tree branches) are integrated over, and hence, no unbalanced charges are encircled. This approach is generally accurate in areas of high interferogram coherence, but it has problems in areas of poor coherence where densely placed cuts close on themselves, preventing integration into whole regions of the interferogram.

Since cuts serve as possible locations of discontinuities, or phase gradients greater than one-half cycle, the total cut length is an upper bound on the total discontinuity length (*i.e.*, the L^0 norm). Minimizing the total tree length is thus a reasonable objective, but finding the exact minimum involves solving an *NP*-hard problem called the minimum rectilinear Steiner tree (RST) problem (see Appendix A). Increasing the tree length does not necessarily increase the total discontinuity length, however, as not all tree branches represent discontinuities.

We therefore make the following modification to the residue-cut algorithm. When a tree becomes neutral, we start the next tree at the next nearest charge to the existing tree, and we actually draw a cut to the new tree. In other words, we build a single tree that contains all the charges. If this tree were of exactly minimal length, it would be a minimum Steiner tree. The tree-building process described does not generally result in a minimum Steiner tree, but it does result in what is called a minimum spanning tree (MST), which reasonably approximates a minimum Steiner tree. On an unweighted rectilinear network, for instance, *Hwang* [1976] showed that an MST can be no longer than 1.5 times the length of an exactly optimal Steiner tree.

An MST can be defined as a tree, of the shortest possible total length, that contains all nodes of some set S, given the important restriction that only nodes in S may be parents of multiple nodes. That is, tree branches of an MST are allowed to split only at nodes in S, whereas in a minimum Steiner tree, branches may split anywhere (see Fig. 5.2). With S as the set of all charges, the tree-growth process described above is known as Prim's algorithm [*Prim,* 1957]. Stated more succinctly, one can build an exact MST by initially designating

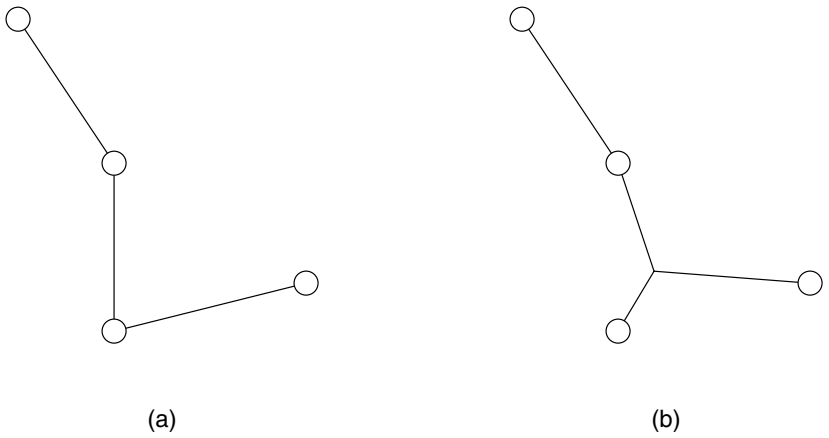

Figure 5.2 Comparison of (a) a minimum spanning tree and (b) a minimum Steiner tree. Only nodes in the set S are shown. The branches of the former tree may split only at these nodes, whereas the branches of the latter tree may split at any node in the network (not shown explicitly).

an arbitrary node in S as the tree root, then adding the shortest path from any node in S on the tree to any node in S not on the tree, and continuing this way until all nodes in S are on the tree (see Fig. 5.3). Note that in this case, our MST is meant to span only nodes of nonzero phase residue (the nodes in S), not all nodes of the network. That is, the tree spans all nodes of an implied network consisting of the set of all charges and the shortest paths between these charges. MSTs have been used previously for phase unwrapping in a different way to connect reliable areas of a phase field derived from magnetic resonance imaging [*Ching et al.,* 1992].

The reader may suspect that our inclusion of additional tree branches would tend to make even more cuts close on themselves than in the original residue-cut algorithm, leaving even more holes in the unwrapped solution. However, a true tree, by definition, cannot contain cycles and should never close on itself. Indeed, an examination of the source code of the original residue-cut implementation reveals that cuts are allowed to close on themselves only because of their internal computer representation: Cuts are associated with particular phase *values*, as in Fig. 5.4. A cut, however, physically represents a phase *difference* between two adjacent phase values that may be greater than one-half cycle. Since phase values are associated with both row and column differences (*i.e.*, the components of the vector gradient), cut locations cannot be stored faithfully in association with scalar phase values alone. Thus, in our algorithm, we associate cuts with phase differences as in the network

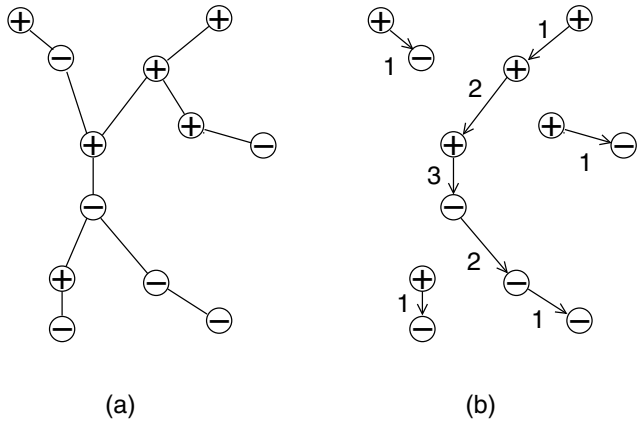

Figure 5.3 A minimum spanning tree and the locations of flow. A minimum spanning tree connects all residues (a). The tree branches are possible locations of discontinuities, which occur where flow on the tree is nonzero (b). Numbers in (b) represent flow magnitudes.

model, and our tree never closes on itself.

Because of further idiosyncratic behavior in the original residue-cut implementation, two charges both a distance d away from a tree might both be added through d distance cuts, even if the addition of one allows the addition of the other through a shorter cut. Some of the algorithm's cuts are thus longer than necessary, making closed cuts more likely. We avoid this behavior by following Prim's algorithm more closely. Thus, incorporating network ideas into our revised algorithm, we can overcome the most significant drawback of the residue-cut algorithm, the closing of cuts.

Further improvement to the algorithm is found in another network idea that allows the specification of weights on the input data. Instead of searching for the closest charge to a tree by examining boxes of increasing size, we can use Dijkstra's shortest-path algorithm [*Dijkstra,* 1959; *Ahuja et al.,* 1993]. This well-known algorithm is used to find the shortest paths from a given source node to all other nodes in some network whose arcs have nonnegative integer lengths or weights. By using Dijkstra's algorithm to find the nearest charge to the tree in each step of Prim's algorithm, we can define weights on individual phase gradients, thus guiding the placement of tree branches. Such weights might provide for greater flexibility than allowed by the neutral-charge residues, or neutrons, which are used for similar purposes in some variants of the residue-cut algorithm [*Madsen,* personal communication, 2001].

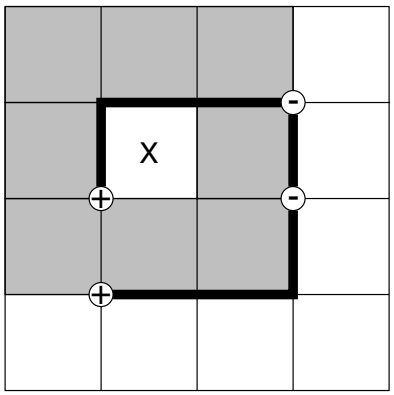

Figure 5.4 Cut representations. Residues and cuts are associated with phase *values* (gray boxes) in the original implementation of the residue-cut algorithm. This allows the cuts to close on themselves so that pixel X cannot be unwrapped. When discontinuities are associated with phase *differences* (heavy lines), as in the network model, no pixels are isolated. Note that the tree defined by the cuts indicated is a spanning tree, but not a minimum spanning tree.

Significantly, the collection of shortest paths found by Dijkstra's algorithm constitutes another tree which spans all nodes in the network and has the source as its root. The shortest path from any node to the root is simply the path between those two nodes along this tree. Note that the shortest-path tree produced by Dijkstra's algorithm spans all nodes of the network, whereas the discontinuity-cut tree for which the MST algorithm is named need span only nodes of nonzero residues. Nevertheless, the close relationship between the two indicates the ubiquity of spanning-tree ideas in network theory. As we will revisit the shortest-path tree in the following sections, we simply point out its existence for now; all subsequent references to trees in this section refer to the discontinuity-cut tree.

Bringing together the above ideas, our MST algorithm is summarized as follows. We begin by selecting one arbitrary charge as the root of our tree. Using the shortest-path algorithm, we find the nearest charge to the tree in the positively weighted network, and we add this new charge to the tree by zeroing the weights of arcs on the path to it. Since all nodes on the tree now have a distance of zero between them, we can locate the next nontree charge nearest to any node on the tree. We successively add nearest nontree charges to the tree until the tree spans all charges, resulting in an approximate minimum Steiner tree on the weighted network. Note that we do not follow Prim's algorithm exactly in that we allow new tree branches to connect not only to charges, but to any nodes in the existing tree.

```
algorithm MST
begin
    choose one charge as the root of the tree T
    while T does not contain all charges do
    begin
        find the shortest path Λ from any node on T to a charge not on T
        incorporate Λ into T, and set the arc distances in Λ to zero
    end
    integrate the phase
end
```

Figure 5.5 MST algorithm pseudocode.

This relaxation produces a solution whose length is closer to that of the optimal Steiner tree. Pseudocode is given in Fig. 5.5.

In implementation, efficiency can be greatly increased by using special data structures [*Dial,* 1969; *Ahuja et al.,* 1993] and by recomputing at each stage only the quantities affected by the inclusion of a new tree branch. The edges of the interferogram can be handled, as *Costantini* [1998] proposed, by connecting all boundary nodes to a single 'ground' node that is assigned whatever charge necessary to make the network neutral. This is equivalent to the *Goldstein et al.* [1988] technique of using a 'superconducting' boundary.

To find the unwrapped phase associated with the tree formed, we can use the flood-fill integration method of the residue-cut algorithm since our tree does not close on itself. Alternately, we can integrate the phase using our knowledge of the network flow as is done in the MCF algorithm (see Chapter 2). Flow, in our case, is restricted to arcs on the tree, so a unique flow is defined by the structure of the tree and the arrangement of charges on it. We can easily determine this flow by recursively descending the tree from the root and integrating charge from the tree leaves back up (as in a depth-first search). Doing so verifies that many tree branches do not carry flow and that by combining all of the trees of the residue-cut algorithm into one large tree, we do not necessarily add to the overall L^0 discontinuity length.

Since the MST algorithm is derived from the residue-cut algorithm, we would expect it to be just as accurate in the unweighted case, with the extra advantage that a complete solution is always produced. Moreover, if the user specifies weights based on ancillary information, the algorithm can favor cuts in locations where discontinuities are more likely,

resulting in even greater solution accuracy. Surprisingly, there is no trade-off in run time for this increase in capability. In fact, our MST implementation runs even more quickly than our residue-cut implementation, probably because the latter is burdened by some redundant searching.

5.2 Dynamic-Cost Cycle Canceling

The MST algorithm is fast because it approximates a global L^0 objective by evaluating only local quantities at every step. Such a strategy may be unreliable where residues become dense in the input interferogram, however. These situations call for a more complex, iterative algorithm. We develop such an algorithm in this section by extending the network ideas of the linear MCF problem to the concave L^0 case. The technique we use is called cycle canceling, and it is similar to the one used by *Flynn* [1997] in his L^1 minimum-weighted-discontinuity algorithm. Here, we adapt the cycle-canceling technique so that it makes explicit improvements in the L^0 sense to any unwrapped initialization. The result is an approximate L^0 solution.

Linear cycle canceling is discussed at length in the text by *Ahuja et al.* [1993], so here we describe the general technique only briefly. We focus instead on the ways in which nonlinearities affect the cycle-canceling approach in the context of phase unwrapping. We first point out, though, that our convention for representing arcs is different than the one used by *Ahuja et al.* [1993]. Because phase gradients may be either positive or negative, arcs in our network model are bidirectional, or able to carry flow in either direction. We therefore assume that the sign of a flow value represents its direction. In references that primarily address linear problems, however, bidirectional arcs are typically represented by pairs of oppositely directed unidirectional arcs. This convention allows a network with L^1 cost functions, which are only piecewise linear, to be represented entirely by arcs with strict linear costs over their ranges of allowable flow. Thus, at the expense of an increased number of arcs and flow constraints, most L^1 algorithms make use of the special characteristics of linear cost functions. The consequent inapplicability of these algorithms to nonlinear problems is the motivation for this chapter.

The specifics of the phase unwrapping problem also allow one further simplification to the general network model. Since there are no hard upper or lower bounds on the values of the phase gradients, we assume that all arcs are uncapacitated, and are hence unconstrained

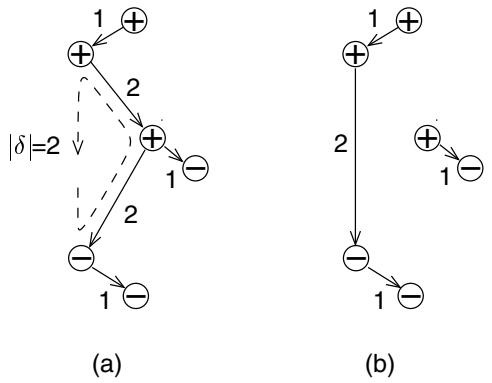

Figure 5.6 A cycle-canceling improvement. A nonoptimal flow in the L^0 sense (a) may be improved upon by augmenting flow on a closed, directed cycle (dashed line) that has a net negative residual cost. The result (b) is a feasible solution with a smaller total cost. The numbers indicate flow magnitudes. The cycle shown has a negative cost with respect to the L^0 metric for $|\delta| = 2$, but not for $|\delta| \neq 2$.

in the amounts of flow they carry. This difference is minor, as an uncapacitated network can always be viewed as one in which the arc capacities are infinite.

As might be imagined, the key concept of cycle canceling is that of the cycle. Observe that if we have a solution for which flow is conserved, flow conservation is preserved with the addition of flow on a closed, directed cycle (*i.e.*, a loop). That is, since the net flow out of each node remains unchanged, the superposition of a directed flow cycle onto an existing feasible flow results in a new feasible flow. The idea of cycle canceling is to augment flow on directed cycles that result in new solutions with smaller total costs. An example of such a cycle is shown in Fig. 5.6.

In order to determine how a cycle affects the total solution cost, we examine the residual network, a collection of nodes and arcs that maintains the state of an iterative algorithm.[1] In our case, the residual network has the same arrangement of nodes and arcs as the original network. The cost of a residual arc, however, reflects the incremental cost of sending additional flow on the original arc given its existing flow in the current iteration.

As an example, suppose the original network contains an arc (p, q) going from node p to node q. With L^0 cost functions, the cost of arc (p, q) will be zero if its flow χ is zero, and some constant cost C otherwise. As shown in Fig. 5.7, the residual network will contain an

[1] The term "residual" here arises from notational overlap and is unrelated to phase residues.

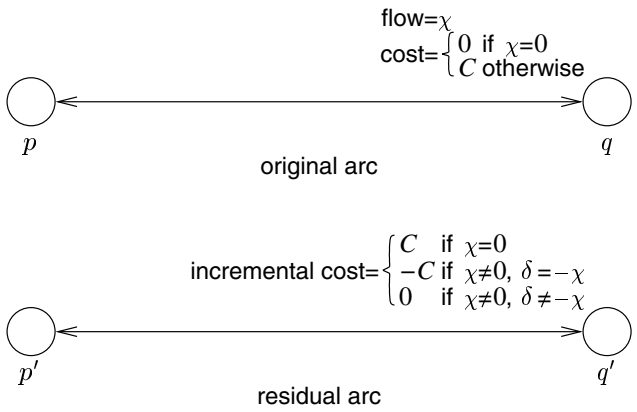

Figure 5.7 Original and residual arcs for the L^0 network problem. The nonlinearity of the L^0 objective makes arc costs in the residual network dependent upon the flow in the current solution as well as the flow increment.

arc (p', q') which reflects the incremental cost of sending some nonzero amount of flow δ on arc (p, q), given its existing flow χ. If arc (p, q) carries no flow (*i.e.*, $\chi = 0$), the cost of the residual arc (p', q') is simply C. That is, adding flow to an arc that does not already have flow incurs an additional cost equal to the cost of the arc. If $\chi \neq 0$ and $\delta = -\chi$, however, the residual cost is $-C$; the flow increment δ, when augmented on arc (p, q), would cancel the existing flow on the arc since $\chi + \delta = 0$. The cost of the arc would thus go from C to zero for an incremental or residual cost of $-C$. Alternately, if $\chi \neq 0$ and $\delta \neq -\chi$, the residual cost is zero; the augmentation of δ units of flow on (p, q) would simply make χ go from one nonzero value another, so the arc cost would remain unchanged.

Since the costs of flows on distinct arcs are evaluated independently then summed to obtain the total solution cost, as in Eq. (3.4), the total incremental cost of sending flow on a cycle is simply the sum of the residual costs of arcs on the cycle. Hence, directed cycles that decrease the total solution cost are those with net negative costs in the residual network. Cycle canceling thus involves searching for cycles with negative residual costs and augmenting flow on them to improve the current[2] solution (see Fig. 5.6).

We can find these negative cycles by using a variant of Dijkstra's algorithm. Minor modifications are required to account for negative arc lengths, but the modified algorithm

[2]Throughout this dissertation, we use the word "current" in reference to the present state of iteration, not to the movement of charge as in an electrical current.

still attempts to find paths with lengths closest to $-\infty$ from some source or root to all other nodes in the network [*Ahuja et al.,* 1993]. Recall that in the absence of negative cycles this collection of shortest paths forms a tree spanning all nodes in the network (see Section 5.1). This tree is generated in a typical shortest-path algorithm as follows. Each node in the network is assigned a distance label which represents the node's current distance from the source along some intermediate version of the tree. A node that is not on the tree has an infinitely large distance label. As the algorithm runs, the distance labels are used to suggest ways in which the tree structure can be modified such that nodes can be reached through shorter paths. In turn, modifications to the tree structure result in new distance labels, which allow further tree modifications, and so on, until the tree becomes an optimal shortest-path tree. Thus, with iterative updates to the node distance labels, the tree structure is reworked until it spans all nodes and no node can be reached through a path shorter than its current distance label.

If a negative cycle exists in the network, however, a node's path length can be made arbitrarily negative by repeatedly looping around the cycle. Consequently, by taking an arc's length to be its residual cost, we can use a shortest-path algorithm to identify negative-residual-cost cycles. The cycle-canceling strategy thus involves the following steps: (1) initialize the network with any feasible flow; (2) use the shortest-path algorithm to find some negative cycle; (3) augment flow on this cycle; (4) update the residual costs to reflect the change in flow; (5) continue from step (2), canceling negative cycles until no more can be found. As all network quantities are restricted to be integers, each canceled negative cycle decreases the L^0 objective value by an integer amount, so this value is monotonically nonincreasing and we need not worry about instabilities or convergence failures. When the nonconvex cost functions of the L^0 norm are used, however, the absence of negative cycles does not guarantee exact optimality. This is expected given the problem's *NP*-hardness.

In further contrast to the linear case, arcs with nonlinear cost functions have residual costs that depends on both the flow increment δ and the current flow. For example, the cycle in Fig. 5.6(a) has a negative L^0 cost for $|\delta| = 2$, but not for $|\delta| \neq 2$, where the local sign of δ depends on the relative orientations of the arc and cycle directions. For the generalized case, we therefore search for directed negative cycles with respect to specific flow increments $|\delta|$, and once a cycle is identified and canceled, we recalculate the arc residual costs to reflect the new flow. Because of the need for these continual cost updates, we call this section's technique dynamic-cost cycle canceling (DCC).

```
algorithm DCC
begin
    find an initial feasible flow
    |δ| := 1
    while negative cycles exist in the network do
    begin
        while negative cycles exist for the current flow increment |δ| do
        begin
            identify a negative-residual-cost cycle Θ using a shortest-path algorithm
            augment |δ| units of flow on Θ
            recalculate residual costs of arcs on Θ
        end
        |δ| := |δ| + 1 (modulo the maximum network flow)
    end
    integrate the phase
end
```

Figure 5.8 DCC algorithm pseudocode.

In our DCC algorithm, we thus calculate residual arc costs corresponding to specific flow increments $\pm\delta$, and we search for negative cycles in terms of these costs. When we find a negative cycle, we augment exactly $|\delta|$ units of flow around it, then recalculate any residual costs altered by the new flow. When no more negative cycles can be found, we reset the flow increment $|\delta|$ to equal $|\delta| + 1$, recalculate costs in the residual network, and continue, iterating on $|\delta|$. Since augmenting $|\delta_1|$ units of flow on a negative cycle may create other cycles that have negative costs for $|\delta_2| \neq |\delta_1|$, we continue looping over $|\delta|$ until no more negative cycles can be found for any value of $|\delta|$. Pseudocode is given in Fig. 5.8.

One more subtlety must be addressed before the DCC approach can be implemented for nonconvex cost functions. Since a residual cost depends on the direction of the flow increment, a trivial negative cycle appears in the residual network whenever an arc (p, q) has a nonzero flow χ. That is, we can send an additional $\delta = \chi$ units of flow on the arc for a residual cost $c_{pq} = 0$, and then send $\delta = -\chi$ units of flow back in the opposite direction for a residual cost $c_{qp} = -C$. Physically, this cycle does nothing to the solution since we simply send flow on an arc and immediately cancel it; the cycle's incremental cost is not actually negative since flow in one direction changes the residual cost of flow in the other. Such cycles seem to arise, however, because our cycle-finding routine, having been adapted from the shortest-path algorithm, is designed for direction-independent arc distances and

therefore cannot handle residual costs that do depend on direction. We address this issue further in Sect. 5.4. For now, though, concentrating on the insights of the DCC technique, we sidestep the problem by simply modifying the cycle-finding routine so that it does not backtrack as described. We thereby avoid the problem of trivial negative cycles, although other, nontrivial negative cycles may be overlooked as well.

Any phase unwrapping algorithm can be used to initialize the DCC algorithm, as the cycle-canceling process can proceed from any feasible flow. Empirically, good results are obtained both qualitatively and in the L^0 sense when approximate L^0 initializations from the MST algorithm are used. On the other hand, in our experiments with L^1 initializations, the final solutions obtained are almost the same as their initializations. Presumably, L^1 solutions are often very near local L^0 minima, so L^0 cycle canceling has little effect on these initializations.

5.3 Cycle Canceling with Arbitrarily Shaped Cost Functions

Though the above explanation of the DCC algorithm is given in the context of L^0 cost functions, the algorithm itself does not require any specific assumptions about cost-function shapes. Thus, the cycle-canceling strategy can be used to make iterative improvements with respect to arbitrary cost functions if those cost functions are used in the algorithm's residual cost calculations. That is, for an arc (p, q) with an arbitrarily shaped cost function $g_{pq}(\cdot)$, the residual cost c_{pq} is simply the incremental cost

$$c_{pq} = g_{pq}(\chi + |\delta|) - g_{pq}(\chi) \tag{5.1}$$

where χ is the existing flow on the arc and $|\delta|$ is the flow increment. Similarly, for incremental flow on the arc in the opposite direction,

$$c_{qp} = g_{pq}(\chi - |\delta|) - g_{pq}(\chi). \tag{5.2}$$

Note that χ is defined to be positive if flow goes from p to q and that $g_{pq}(\chi) = g_{qp}(-\chi)$.

Since the cost functions are evaluated independently and then summed, the functions $g_{pq}(\cdot)$ may take independent, arbitrary shapes. A shortest-path algorithm can still be used to find negative cycles in the residual network, so with this minor modification, the DCC

algorithm can be used to solve, approximately, separable instances of the general phase-unwrapping optimization problem. While this algorithm is fairly easy to explain and implement, however, it is relatively inefficient. Moreover, its shortest-path cycle-finding routine does not account for the dependence of nonlinear residual arc costs on the direction of the flow increment δ.

5.4 The Pivot-and-Grow Algorithm

Simple cycle canceling is seldom used in practice for linear applications because better, more efficient algorithms have been developed. We now borrow from some of these approaches and apply them to the case of nonlinear, arbitrarily shaped cost functions. Specifically, we combine our generalized DCC algorithm with ideas from the network-simplex algorithm [*Kennington and Helgason,* 1980], Pallottino's double-queue shortest-path algorithm [*Pallottino,* 1984], and Dial's implementation of Dijkstra's algorithm [*Dial,* 1969; *Ahuja et al.,* 1993].

While these algorithms are meant for different purposes, they all share the common idea of the spanning tree: Upon termination, each algorithm represents its solution by a tree that connects all of the nodes in the network. We explained above that shortest-path solutions can be represented as spanning trees, and since shortest-path algorithms can be used to find negative cycles, it should not be surprising that spanning trees are closely related to our DCC algorithm and to linear MCF algorithms.

Consider some intermediate iteration of a typical shortest-path algorithm. Nodes are arranged on some tree whose root is the source, and each of these nodes is labeled by its current distance from the root along the tree (see Section 5.2). The same kinds of ideas are used in the network-simplex algorithm to find and cancel negative cycles in linear MCF applications [*Kennington and Helgason,* 1980]. At each iteration, the algorithm maintains a spanning tree in which every node is labeled by a quantity called its potential. Analogous to the shortest-path distance label, the potential of a node p represents the incremental cost of sending δ units of flow from the root to p along the tree. Equivalently, the potential can represent the cost of sending the same flow in the opposite direction since residual arc costs are independent of δ when the cost functions are linear. A node p therefore has a potential π_p that equals the sum of the residual costs of all tree arcs between the p and the root. Just as distance labels suggest ways in which the tree structure can be improved in a shortest-path

algorithm, node potentials in the network-simplex algorithm are examined and updated to guide the improvement of the residual-cost tree. In other words, as the residual-cost tree is brought towards optimality in terms of its node potentials, negative residual-cost cycles can be identified just as they are identified by the shortest-path algorithm described above. The node potentials of a spanning tree are thus used by the network-simplex algorithm to find and eliminate negative cycles.

Since a network-simplex tree spans all nodes, a cycle is formed with the addition of any nontree arc to the tree; the removal of some other arc from the cycle results in a new spanning tree (see Fig. 5.9). This action, moving from one spanning tree to another, is called a pivot by analogy to the simplex method for linear programming (see the text by *Nash and Sofer* [1996]). Because a pivot results in modifications to the tree structure, the node potentials consequently change as well. A network-simplex solver thus selects pivots that reduce node potentials, identifying and eliminating negative cycles in the process. In this manner, the solver steps from spanning tree to spanning tree, improving the solution as it goes. When the tree becomes optimal in terms of its node potentials, no negative cycles can exist and the optimal flow has been found. Finite termination is ensured by the fact that on every iteration, an integer improvement is made to either the node potentials or the total solution cost [*Ahuja et al.,* 1993; *Kennington and Helgason,* 1980].

We adapt the idea of pivots for use with arbitrarily shaped cost functions in our hybrid algorithm. That is, we calculate residual costs in terms of the current arc flows and a specific flow increment $|\delta|$, as in the generalized DCC algorithm. Because each arc (p, q) in the nonlinear case has two different residual costs, c_{pq} and c_{qp}, one for either direction of the flow increment, we assign each node p two potentials rather than just one: an outwards potential $\pi_p^{(\text{out})}$ that represents the cost of sending $|\delta|$ units of flow along the tree from the root to p, and an inwards potential $\pi_p^{(\text{in})}$ that represents the cost of sending the same flow in the opposite direction. Note that if node u is an ancestor of node p, the cost of sending $|\delta|$ units of flow from u to p is $\pi_p^{(\text{out})} - \pi_u^{(\text{out})}$, and the cost of sending $|\delta|$ units of flow from p to u is $\pi_p^{(\text{in})} - \pi_u^{(\text{in})}$. We additionally associate each nontree arc with an apex node, the node closest to the root on the cycle formed by adding the arc to the current spanning tree. If arc (p, q) is not on the tree, its apex is thus the first common ancestor of nodes p and q.

Let u be the apex of the cycle formed by adding nontree arc (p, q) to the current spanning

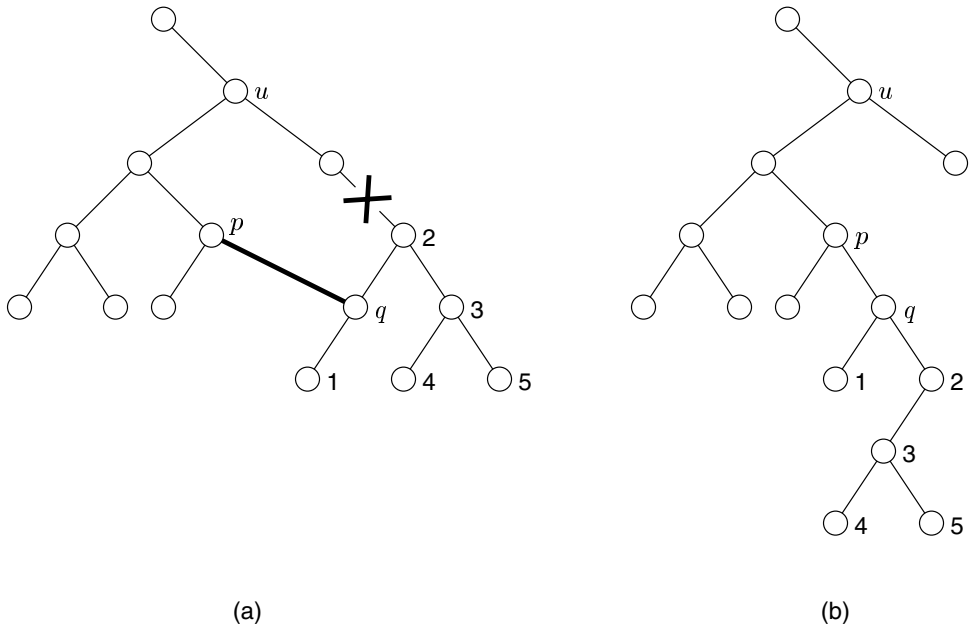

Figure 5.9 An example pivot. Arc (p, q), shown in heavy black, is added to the tree and some other arc is removed from the resulting cycle (a). The pivot results in a new tree structure (b).

tree. The inclusion of the arc results in a negative cycle (see Fig. 5.10) if

$$\left(\pi_p^{(\text{out})} - \pi_u^{(\text{out})}\right) + \left(\pi_q^{(\text{in})} - \pi_u^{(\text{in})}\right) + c_{pq} < 0 \tag{5.3}$$

where the residual cost c_{pq} is given by Eq. (5.1). The augmentation of flow on this cycle and the subsequent removal of one arc from the cycle results in a new spanning tree with a smaller total cost.

Alternately, the addition of arc (p, q) improves (reduces) the outwards potential of node q, as in a shortest-path algorithm, if

$$\pi_p^{(\text{out})} + c_{pq} < \pi_q^{(\text{out})}. \tag{5.4}$$

Flow is not necessarily augmented on the resulting cycle, but such pivots (ones that only improve node potentials) are necessary steps in the process of locating negative cycles. In fact, such pivots often constitute most of the algorithm's work, for it is the improvement of node potentials, like the reduction of distance labels in a shortest-path algorithm, that

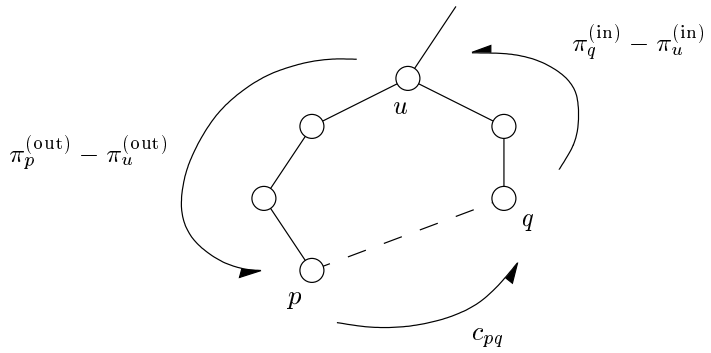

Figure 5.10 A cycle's residual cost in terms of node potentials. Inwards and outwards node potentials can be used to compute the residual cost of the cycle formed by adding arc (p, q) to the spanning tree. The cycle's apex is u.

guides the algorithm towards tree structures in which negative cycles are apparent. For the nonlinear problem, however, pivots that improve outwards potentials might make inwards potentials worse. Our algorithm thus makes no attempt to improve inwards potentials directly; it relies solely on improvements of outwards potentials for finding negative cycles.

After each pivot, augmenting or not, the potentials and tree structure must be updated. *Kennington and Helgason* [1980] describe a number of methods and related data structures for performing these tasks efficiently in the network-simplex algorithm. For the most part, these routines can be used in the nonlinear case as well. However, it is important to note that node potentials will need more frequent update in the nonlinear case because of their dependence on the current flow and the flow increment as well as the tree structure. Moreover, our apex pointers, which have no linear-case counterparts, must also be updated once the tree structure is modified. Nevertheless, by maintaining a few extra data arrays and iterating on the flow increment $|\delta|$, we can adapt the network-simplex algorithm to solve nonlinear problems, albeit only approximately.

Empirically, we have found that further speed improvements result from the adoption of a tree-growth strategy similar to the one suggested by *Pallottino* [1984] for the shortest-path problem. Observe that the network-simplex algorithm maintains a complete spanning tree at every iteration. On the other hand, Pallottino's shortest-path algorithm grows a spanning tree as it progresses, fanning out from the root. Specifically, Pallottino's algorithm deals with a tree T that spans some, but not necessarily all, nodes in the network. At each iteration, the algorithm adds a new node to T, then optimizes T in terms of the nodes

it does span. This approach is advantageous computationally because much of the effort put into tree optimization is done while the tree is still small, so fewer total operations are required. Applying this idea to our algorithm, we start with an arbitrary root node and grow a spanning tree T, maintaining its optimality over each major algorithm iteration. Each of these major iterations is divided into two stages. In the first stage, a predetermined number of new nodes is added to the tree in the order prescribed by the shortest-path algorithm [*Dijkstra,* 1959; *Ahuja et al.,* 1993]. These nodes, prior to their inclusion in the tree, are stored in bucket data structures similar to those suggested by *Dial* [1969]. In the second stage of a major algorithm iteration, T is reoptimized in terms of all the nodes it now spans, using pivots as in the network-simplex algorithm. After the final iteration, T spans all nodes in the network, and the network flows represent an approximately optimal solution with respect to the generalized cost functions defined for the problem. Pseudocode is given in Fig. 5.11.

Since this algorithm solves the general optimization problem, it can also be used with L^p cost functions. If the cost functions are convex, the algorithm's solutions will be exactly optimal over the sets of all congruent solutions. For example, when $p = 2$, the algorithm provides a solution optimal over all congruent least-squares solutions. However, a transform-based least-squares algorithm might produce a noncongruent solution better in the L^2 sense. If a post-optimization congruence operation [*Ghiglia and Pritt,* 1998] is applied to the latter solution, though, the solution loses optimality even over the set of strictly congruent solutions (see Chapter 3). For L^1 cost functions with integer weights, an optimal integer flow always exists [*Ahuja et al.,* 1993], so congruent L^1 solutions generated by our algorithm are exactly optimal.

Together with the statistical cost functions of Chapter 4, we call our implementation of this solver algorithm SNAPHU, an acronym for statistical-cost, network-flow algorithm for phase unwrapping.

```
algorithm Pivot and Grow
begin
    find an initial feasible flow
    |δ| := 1
    while negative cycles exist in the network do
    begin
        choose some node to be the root of a tree T
        while not all nodes in the network are on T do
        begin
            for 1 ≤ i ≤ n
            begin
                add to T the nearest adjacent node to it
                set the inwards and outwards potentials of the new node
            end
            while T has nonoptimal node potentials for the nodes it spans do
            begin
                find an arc (p, q) whose addition to T results in an improvement
                add (p, q) to T
                if a negative cycle is formed
                    augment |δ| units of flow on the cycle
                endif
                drop some arc on the cycle from the tree
                update the tree structure
                update the inwards and outwards potentials of nodes affected by the pivot
                update the apex pointers affected by the change to the tree's structure
            end
        end
        |δ| := |δ| + 1 (modulo the maximum network flow)
    end
    integrate the phase
end
```

Figure 5.11 Pivot-and-grow algorithm pseudocode.

Chapter 6

Tiling Strategies for Large Data Sets

The phase unwrapping algorithm developed in the previous two chapters addresses many of the difficulties inherent in unwrapping interferometric SAR data. New difficulties arise, however, when the dimensions of the input data are so large that the algorithm requires more computer memory than one has available. Relatedly, higher computational throughput may be required for applications involving voluminous quantities of data. In this chapter, we examine the practical and theoretical challenges unique to large data sets.

We use the term "large" to describe interferograms approaching or exceeding the limits of typical computer resources available for unwrapping them. An obvious strategy for circumventing these resource limitations is to adopt an approach in which a large interferogram is treated as a mosaic of smaller interferograms called tiles. The tiles can be unwrapped in sequence to reduce the required memory, or they can be unwrapped in parallel to reduce the required execution time [*Costantini,* 1998; *Carballo,* 2000]. Unfortunately, as explained below, boundary effects often make smaller interferograms more difficult than larger ones to unwrap correctly. Moreover, combining individually unwrapped tiles is not a straightforward task. Consequently, a mosaic of tile solutions may be more error prone than a solution obtained by unwrapping the entire interferogram as a single piece. To avoid such errors as much as possible, we propose here a tiling scheme based on the statistical-cost, network-flow framework described in the previous chapters.

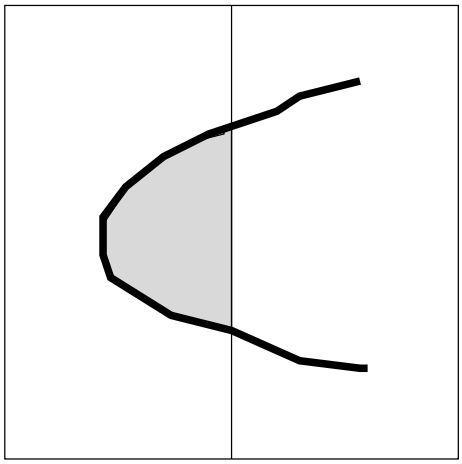

Figure 6.1 Representation of an unwrapping artifact due to tiling. The heavy black line represents a phase discontinuity following the decorrelated front face of a mountain in layover. Range is assumed to increase towards the right. If the interferogram is split into left and right halves as shown, the shaded area is isolated from the rest of the left half by the decorrelated region.

6.1 Basic Tiling Strategies

For interferograms with long discontinuities, unwrapping accuracy often degrades with the degree of interferogram subdivision. Consider a topographic interferogram such as the one illustrated schematically in Fig. 6.1, where range is assumed to increase towards the right. The heavy black line represents a phase discontinuity associated with the decorrelated front face of a mountain in layover. If the interferogram is unwrapped as a single piece, all parts of the interferogram can be connected by reliable integration paths despite the presence of the discontinuity (see Chapter 2). On the other hand, if the scene is partitioned into two smaller tiles as shown, the discontinuity isolates the shaded area from the rest of the left tile, and an unwrapping error likely results. Such tiling artifacts are even more likely to occur in interferograms containing longer or more sinuous discontinuities. Thus, the adoption of a tiling strategy requires great care. In this section, we examine a relatively simple tiling strategy, and in so doing, we review the basic issues raised by interferogram subdivision.

Breaking up an interferogram into separate tiles is easy; putting those tiles back together is the difficult step. *Carballo* [2000], however, noted that the problem of mosaicing individually unwrapped tiles is similar to the original (single-piece) phase unwrapping problem in

that integer-cycle offsets must be added to each tile in order to align the phase values of adjacent tiles. That is, adjacent tiles must be brought to within one-half cycle of one another throughout most of the array of tiles. Tiles in the tile-reassembly problem are hence analogous to pixels in the original phase unwrapping problem. For brevity of notation, we refer here to the pixel-level unwrapping problem and the subsequent tile-reassembly problem as the primary and secondary problems, respectively.

For the secondary problem, the degree of alignment between adjacent tiles can be quantified in various ways. We use the following procedure. For each pair of pixels straddling the tile boundary, we compute the unwrapped phase difference and round the result to the nearest integer cycle. We then find the most common value of the rounded pixel differences and call this value the 'tile difference' between the two tiles. In virtually all real-world situations, the true tile difference between any two correctly unwrapped tiles of reasonable coherence is zero. This is because most pixel differences along the boundary are less than one-half cycle by assumption (see Chapter 2), even if some are not.

Thus, if all individual tiles are unwrapped correctly, there must exist an arrangement of tile offsets such that all tile differences are zero. These offsets can be found by aligning adjacent tiles following some path from tile to tile, just as pixel-to-pixel phase integration results in a self-consistent primary solution for an interferogram in which there are no residues.

If many of the pixels along a tile boundary are unwrapped in error, however, the affected tile difference may no longer be zero in the desired secondary solution. This case is illustrated in Fig. 6.2. Tile reassembly thus becomes a problem similar to phase unwrapping in the presence of residues: Tiles must be arranged so that tile differences are zero *most* of the time—but tile differences affected by primary unwrapping errors may need to be nonzero for self consistency in the secondary solution.

Realizing this, *Carballo* [2000] showed that the secondary problem of arranging phase offsets between tiles can be described by a network model nearly identical to the network model used in the previous chapters for the primary problem. The secondary network for the tile set of Fig. 6.2 is shown in Fig. 6.3. Nodes exist at the tile corners, and, with arcs connecting neighboring nodes, there is a tile difference for each arc in the network. Therefore, as in the primary network, flow on a secondary arc represents the additional number of cycles that should be included in the corresponding tile difference. Residues arise in Fig. 6.3 because of the unwrapping error in the primary solution for tile B. That

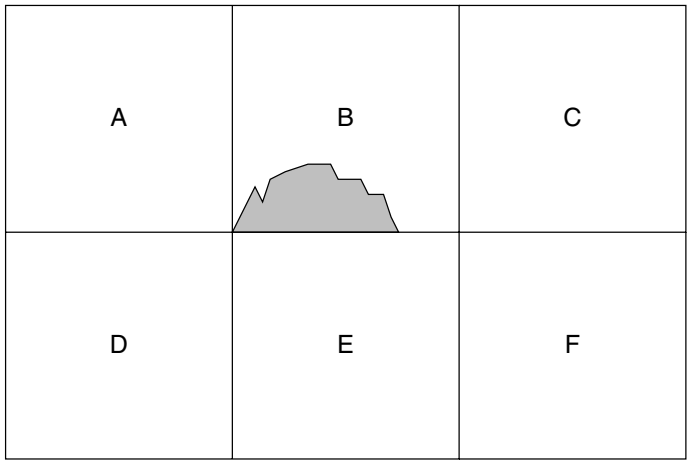

Figure 6.2 Effects of an unwrapping error on tile differences. The shaded area in tile B represents an unwrapping error which is offset from the rest of the tile by an incorrect integer number of cycles. The tile difference (*i.e.*, phase difference) between tiles B and E must be nonzero to account for this error if the tile differences between the rest of the tiles are assumed to be zero.

is, the sums of tile differences around the closed loops {A,B,E,D,A} and {B,C,F,E,B} are nonzero. Equivalently, there is nonzero net secondary flow out of the nodes indicated when the tile solutions are arranged with arbitrary offsets and the secondary arc flows are calculated from the resulting unwrapped tile differences.

If a superconducting boundary (see Chapter 2) is desired for the secondary network, secondary arcs along the edges of the scene can be assigned zero cost. In this case, all secondary nodes on the scene edge behave effectively as a single ground node which can be assigned whatever charge necessary to make the network neutral. Cost functions for secondary arcs not on the scene edge are discussed in Section 6.3 below.

With the network defined in this way, network-flow solvers (see Chapters 2 and 5) can be used to arrange the tile offsets in some optimal way. This simple tiling strategy may be acceptable for interferograms whose tiles are easy to unwrap individually, but for interferograms characterized by long phase discontinuities, tiling artifacts may become much more problematic. This is because relative unwrapping errors that exist in the primary solutions of individual tiles will also exist as errors in any final tile-reassembled solution; no arrangement of tile offsets can correct for unwrapping errors internal to individual tiles. The likelihood of tiling artifacts can be lessened by specifying larger tiles, but doing so also

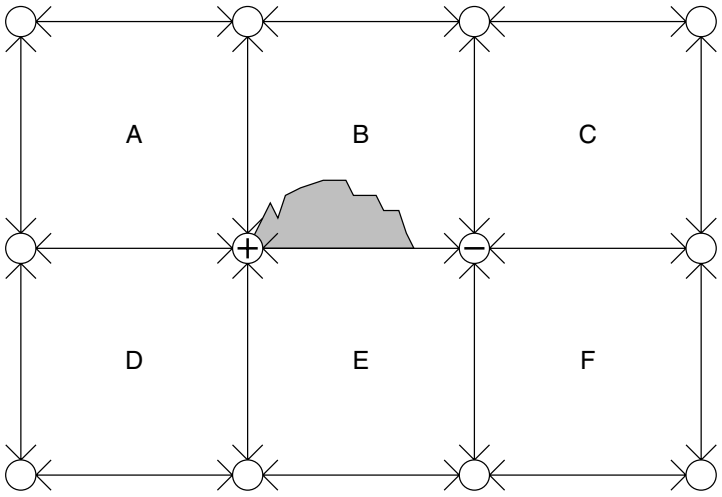

Figure 6.3 Secondary network for the tile set of Fig. 6.2. Nodes exist at tile corners, and arcs between nodes correspond to tile differences. Residues exist because the unwrapping error in tile B affects the tile difference between tiles B and E.

mitigates the computational gains motivating the technique in the first place. The same applies to the use of widely overlapping tiles.

6.2 Reliable Regions and Tile Subdivision

To avoid the artifacts associated with the simple tiling approach described in the previous section, *Carballo* [2000] insightfully proposed the following technique. Individual tiles are partitioned into independent, arbitrarily shaped regions that are believed to be free of relative internal unwrapping errors. The secondary network problem is then carried out between these reliable regions rather than between whole tiles. Thus, with the offsets of individual regions independently adjustable, major unwrapping errors within primary tile solutions can be corrected as the secondary problem is solved.

For defining reliable regions, *Carballo* [2000] used a scheme in which multiple primary unwrapped solutions from different interferogram partitionings are compared to one another and to a coherence map. The use of redundant primary solutions imposes a significant additional computational burden, however. In this section, we propose an alternative method for defining reliable regions. Our method is very efficient and is justified theoretically by the statistical-cost framework described in Chapter 4. We discuss techniques for the problem

of assembling these reliable regions in the next section.

The reliability of a region can be quantified by the probability that the primary unwrapped gradients within the region are estimated correctly. Fittingly, our statistical cost functions are, by design, measures of these very probabilities. For example, if a primary arc has high incremental (residual) costs given the current primary unwrapped solution, there is a low probability that flow should be added to or removed from the arc, so there is a low probability that the corresponding phase gradient is estimated in error. Thus, to form reliable regions, we employ a statistical-cost region-growing approach. Region-growing techniques have been used in several contexts for phase unwrapping previously [*Goldstein et al.,* 1988; *Xu and Cumming,* 1999; *Fornaro and Sansosti,* 1999; *Carballo,* 2000].

Our approach is described as follows. For each primary phase difference in a tile, we compute a scalar cost by taking the lesser of the incremental statistical costs for positive and negative unit flow increments given the unwrapped tile solution. We next smooth these scalar costs using convolutional methods. Then, we select a seed pixel as the beginning of a region, and we successively add adjacent nonregion pixels to the region if they can be reached without crossing phase differences whose scalar costs fall below a certain threshold. When no more pixels can be added to the region, a new seed is chosen from the set of nonregion pixels, and another region is grown. This process continues until all pixels in the tile are contained in a region.

To implement this region-growing procedure in an efficient and theoretically rigorous way, we apply the network ideas described in Chapter 5. We can even reuse much of our source code since the region-growing network model is topologically identical to that of the primary phase unwrapping problem (see Fig. 2.6). The region-growing network is set up differently, however; its nodes correspond to pixels rather than residues. Arcs between neighboring nodes still correspond to phase differences, but they now have a different physical interpretation: Arcs connect rather than divide neighboring pixels. We thus favor the use of high-cost arcs in our region-growing algorithm, whereas we favor the use of low-cost arcs in our primary phase unwrapping algorithms. Figure 6.4 shows an example region-growing network.

With this network, we can adapt the tree-growth routine of the MST algorithm (see Chapter 5) for region growing. We first assign each arc a shortest-path distance measure equal to the negative of the arc's scalar phase-difference cost. We next make the region seed the root of the MST algorithm's shortest-path tree, and we take nodes on the tree to

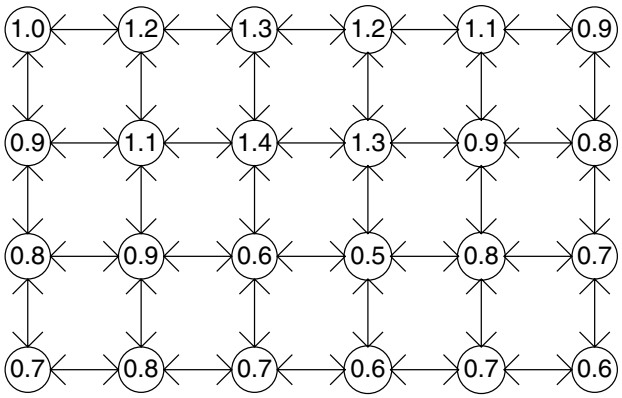

Figure 6.4 Example region-growing network model for a primary unwrapped-phase solution to the wrapped array of Fig. 2.6. The numbers represent values of unwrapped-phase pixels. They are shown here for illustrative purposes only and are not used in the network model. Arcs correspond to paths between pixels across primary unwrapped phase differences.

be part of the current region. As nodes are added to the tree, we reset the costs of arcs between tree nodes to zero so that the tree (*i.e.*, the region) behaves like a single equivalent node. Continuing, we add nodes until no more nodes meet the minimum phase-difference cost requirement.

We next enforce a lower limit on the size of each region so as to keep the number of different regions, and hence the size of the secondary problem, manageable. That is, we identify regions containing fewer than a preset number of pixels and merge these regions with their closest neighbors. Thus, where reliable regions might otherwise be very small (*e.g.*, decorrelated areas), regions are clumped together through arcs with the shortest possible distances. Phase unwrapping errors might accumulate within merged regions, but such areas often provide little useful information in any event. Our intention is to prevent such errors from spreading into more reliable areas.

6.3 Secondary Network Construction

Once reliable regions have been defined within the individually unwrapped tiles, our task is to solve the secondary problem of arranging the phase offsets between different regions. Since regions are arbitrarily shaped, however, the implied secondary network no longer has the same regular, grid-like topological structure as the primary network. Nevertheless, a

network model can still be used for the secondary problem.

Carballo [2000] suggested that the secondary network be constructed from a Delaunay triangulation of control points representing the different regions. This scheme has several disadvantages, though. Because secondary nodes correspond to triangles, these nodes are all of degree three or less. This forces the secondary network to contain extraneous nodes and arcs which degrade computational efficiency and obscure the relationship between the secondary network and the region-assembly problem. The computation of the Delaunay triangulation itself also introduces additional complexity [*Guibas and Stolfi,* 1985; *Fortune,* 1987]. Moreover, region offsets are defined in reference only to control points consisting of single pixels—one for each region. Consequently, the scheme may not be robust to small, localized errors within regions.

We propose a secondary network model that avoids such shortcomings. With this model, the secondary network problem is exactly equivalent to the region-offset problem at hand: Any possible solution to the region-offset problem corresponds uniquely to a feasible flow solution on the secondary network. This secondary network is formed from the region data directly, so triangulation is unnecessary. Furthermore, region offsets are computed with respect to all pixels on the region boundaries, not just single-pixel control points.

Recall that in the primary network model, arcs correspond to phase differences between pixels. We thus assume analogously that secondary arcs correspond to phase differences between regions, and we include one secondary arc in the network for each boundary between two regions. We then place secondary nodes at tile corners and at other locations where more than two regions touch. Nodes thus occur where flow paths split, as illustrated in Fig. 6.5. Since the phase offset between any two neighboring regions is described by a unique secondary arc, any possible arrangement of offsets can be represented by a feasible flow.

We construct the secondary network from the region data using an algorithm similar to a tree search. That is, beginning from a primary node known to correspond to a secondary node (*e.g.*, a node at a tile corner), we trace out secondary arcs by following sequences of primary arcs that have different regions on either side of them or that lie on the edges of the interferogram. A secondary arc terminates when it enters a primary node that touches more than two regions. At these locations, we create secondary nodes if such nodes do not yet exist. We then continue, tracing unvisited secondary arcs that depart from these secondary nodes, until all secondary arcs have been followed. The structure and connectivity of the

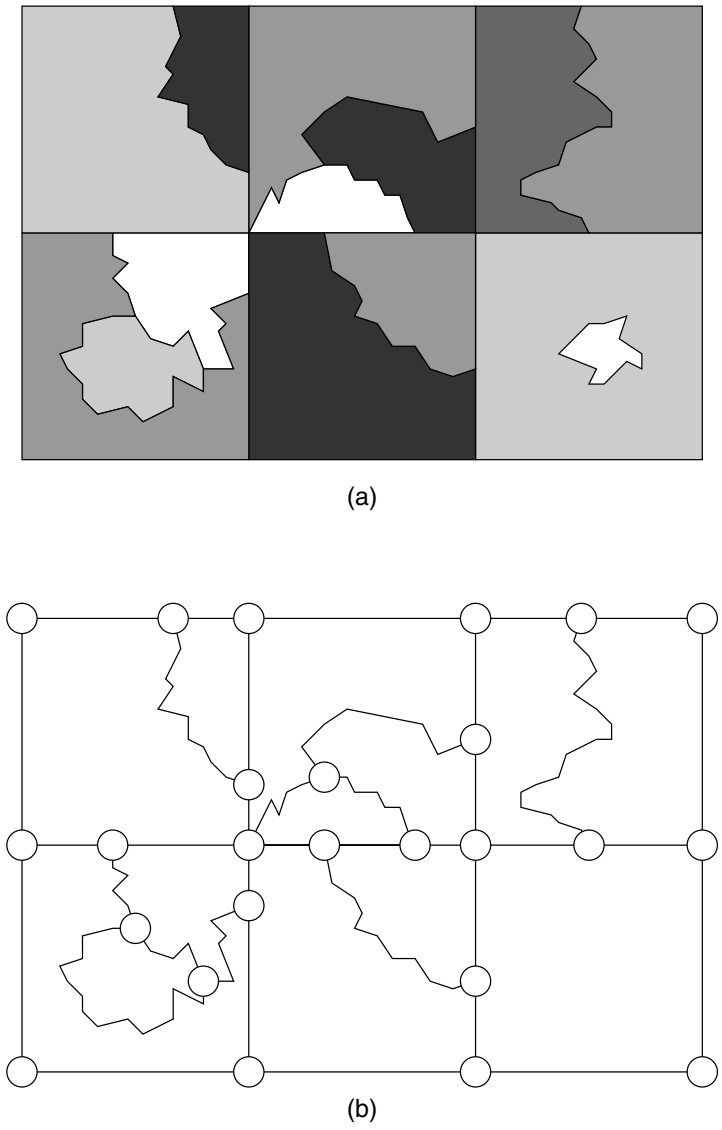

Figure 6.5 Example region-based secondary network: (a) region-set; (b) corresponding secondary network. Shaded areas in (a) represent reliable regions, indicated with arbitrary gray levels, for the set of six square tiles shown. Circles and lines in (b) represent secondary nodes and arcs. Arrowheads on the arcs are omitted for clarity, but all arcs are assumed to be bidirectional, as elsewhere in this dissertation. Arcs correspond to boundaries between regions, and nodes occur where arcs split. The region in the center of the lower-right tile is not represented in the network since this region cannot be reached from any tile edge.

secondary network are thus parsed from the region map.

Note that the secondary network does not contain nodes or arcs for regions that are unaffected by the inclusion of adjacent tiles (*e.g.*, the small, central region in the lower-right tile of Fig. 6.5). The phase offsets of such regions are set relative to their surroundings in the primary problem; these offsets should not change in the secondary problem.

The feasibility of a secondary flow solution depends in principle on the distribution of surplus and demand on the secondary nodes. We need not explicitly compute residues for the secondary problem, though. Instead, since the unwrapped primary solutions provide estimates of the relative offsets between regions within the same tile, we simply initialize the tile offsets and find feasible flow values for the secondary arcs. That is, we choose an initial, temporary set of tile offsets by aligning neighboring tiles across the top row then down each column, defining tile differences as above. Such an initialization is analogous to a trivial path-integration unwrapping method for the primary problem. With the relative region offsets thus selected, we compute initial values for the secondary arc flows. That is, based on the unwrapped primary pixel values and the relative region offsets, we compute the flow for each primary arc that is part of a secondary arc. We then initialize the secondary flows to the most common values of the primary flows along the respective secondary arcs. If desired, residues may be computed now by summing the net flow out of each secondary node. Doing so is unnecessary, though; we can make iterative improvements to the secondary flow initialization, as through cycle canceling, while maintaining the feasibility of the current solution.

In order to make improvements, however, we must define costs for the secondary arcs. Here, we again depart from the scheme proposed by *Carballo* [2000]. We instead propose an approach that is more physically meaningful and theoretically justifiable.

As in Section 6.1, we form an effective ground node on the scene boundary by assigning incremental costs of zero to secondary arcs on the edges of the interferogram. For the other secondary arcs, we derive costs based on our primary statistical cost functions. Since secondary arcs are simply sequences of primary arcs, we define the incremental cost of a secondary arc to be the sum of the incremental costs of the primary arcs traced by the secondary arc, given the current primary flows. In other words, adding δ units of flow to a secondary arc is equivalent to adding δ units of flow (in the appropriate direction) to every primary arc traced by the secondary arc; we therefore equate the incremental cost of the former to that of the latter. Hence, just as in the primary problem, secondary incremental

costs depend on the current secondary solution and the flow increment δ. Nevertheless, we can easily build a look-up table of secondary absolute costs from secondary incremental costs computed for likely values of the flow increment δ.

With secondary costs thus defined, we use the pivot-and-grow solver of Chapter 5 to find a feasible secondary flow that approximately maximizes the *a posteriori* probability of the region-offset solution. Note that the pivot-and-grow solver accepts arbitrary network topologies, so the secondary network's lack of grid-like regularity poses no extra difficulty in solving the secondary optimization problem. Once the secondary flow solution is found, we adjust the primary flows to reflect the new secondary flows and integrate the phase at the pixel level to obtain the full unwrapped interferogram solution.

The result we obtain is an approximation to the solution that would have been obtained had the interferogram been unwrapped as a single piece. That is, our tiling approach produces a solution that approximately minimizes the MAP objective function defined for the *full-sized* interferogram. To see this, imagine that the interferogram is unwrapped as a single piece, starting from an initialization that consists of an arbitrary mosaic of the primary tile solutions. Improvements to this full-sized initialization will be necessary because of tiling artifacts, but many locally correct areas will not need adjustment; flow will be placed only on a small subset of the arcs in the scene—it is these arcs whose connectivity and costs are represented by our secondary network. The other, more reliable parts of the scene need not be examined further. Hence, by using our statistical cost functions to identify reliable regions that are unlikely to need improvement, we can ignore most of the variables of the full-sized problem. We can represent the remaining variables through the secondary network constructions described above. Thus, our tiling methodology is not just an *ad hoc* collection of procedures, it is a cohesive extension of the MAP optimization framework.

6.4 Tile Size Considerations

The tiling scheme described in the previous sections provides a theoretically rigorous approach to interferogram subdivision. While this approach is meant to mimic the results of unwrapping a large interferogram as a single piece, however, the single-piece approach is probably still more accurate. Though expensive—perhaps prohibitive—computationally, the single-piece method avoids uncertainties related to how reliable regions are defined. That is, the region-growing method described here relies only on local arc costs, so some

regions might enclose paths that require flow updates. Because such updates would not be possible in the secondary problem, the tiling approach would contain errors that might be avoided by an iterative algorithm whose scope is the full interferogram. Consequently, there may remain a slight tradeoff between unwrapping accuracy and computational efficiency or capability.

As a result, tile size is an important consideration in the practical application of our approach. Clearly, memory limitations impose strict upper bounds on tile sizes. Moreover, since most phase unwrapping algorithms including SNAPHU have execution times that are worse than linear in the size of their inputs, the aggregate execution time should decrease with tile size as long as overhead associated with region reassembly remains small. For interferograms that are easy to unwrap, there may be no noticeable penalty in accuracy if the scene is divided into many small tiles. All other things being equal, however, we might expect the likelihood of errors to increase as tile sizes decrease. When possible, it would be prudent to keep the tiles larger than the expected characteristic length of phase discontinuities in the input interferogram. Such lengths may be related, for example, to the characteristic physical size of terrain features in a topographic interferogram. Moreover, if phase discontinuities are known to favor a particular orientation (*e.g.*, they tend to run in azimuth more so than range), tile aspect ratios can be adjusted accordingly. Empirical tests may be needed for determining ideal tile dimensions.

Questions of accuracy and efficiency cannot be addressed in isolation, however; both are affected by other factors. As discussed in Chapter 3, different algorithms may be more appropriate for different situations. Fortunately, the secondary network model described in this chapter is completely general, so our tiling approach can be used with other optimization frameworks and solvers as well.

Finally, we note that in some cases, the need for tiling might be obviated with the adoption of parallelization schemes based on shared memory rather than interferogram subdivision. We leave this as a topic for future work.

Chapter 7

Results

In this chapter, we examine the performance of our phase unwrapping algorithm. In the previous chapters our intent has been to maintain a balance between setting up and solving the phase unwrapping problem so that our algorithm can be as accurate as practical given the needs of real-world applications. We have thus considered our MAP cost functions and nonlinear network-flow solvers—along with their underlying assumptions and approximations—together, as equally important parts of a unified phase unwrapping strategy.

Because our goal remains the development of a useful working algorithm, we test the algorithm on real rather than simulated interferometric SAR data. We examine a variety of data examples in this chapter: a topographic interferogram containing areas of low correlation, rough topography, and layover; a deformation interferogram measuring surface displacement from a major earthquake; and a set of topographic interferograms that are part of a large-scale mapping project. In comparisons with existing algorithms, our algorithm yields superior solution accuracy and demonstrates competitive computational efficiency. The results therefore suggest significant promise for our approach.

7.1 Topographic Data: Death Valley

In Chapter 3, we examined a topographic test interferogram of an area near Death Valley, California, and we showed the results of applying several existing L^p algorithms to it. We now examine the results obtained from SNAPHU and our MST algorithm.

The test data are reproduced here in Fig. 7.1 for completeness. Figure 7.1(a) depicts

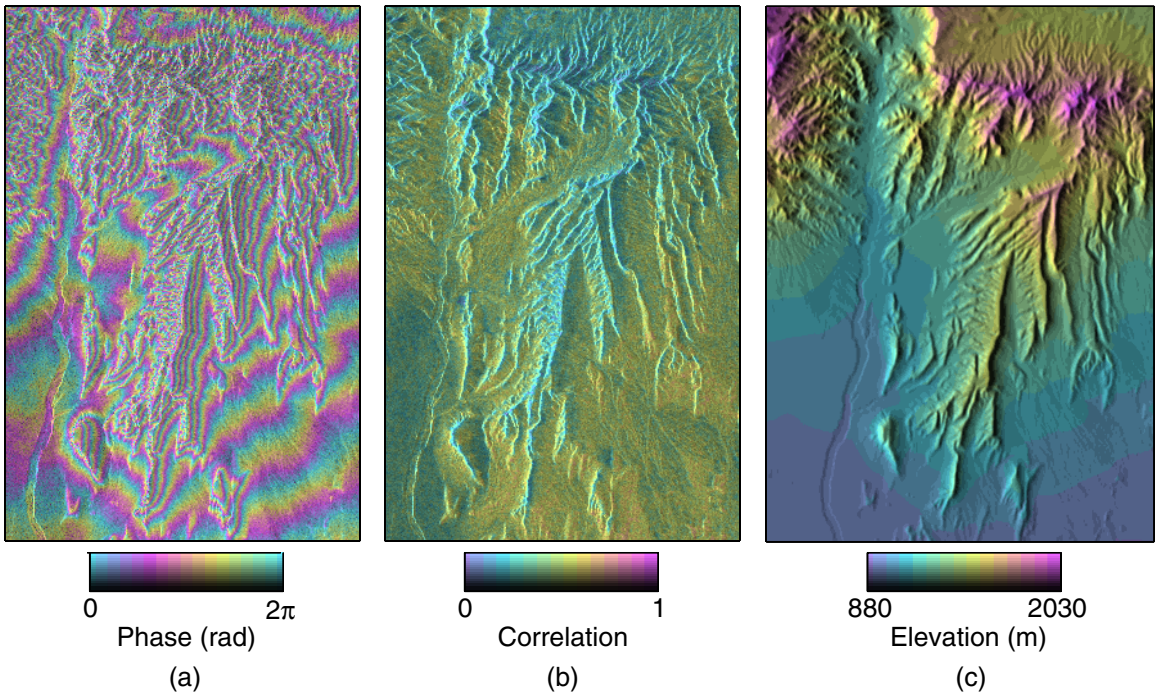

Figure 7.1 Death Valley topographic test data: (a) interferogram with wrapped phase in color and magnitude in gray-scale brightness; (b) measured coherence in color with interferogram magnitude in gray-scale brightness; (c) reference DEM with elevation in color and shaded relief in gray-scale brightness.

the 1250×830 pixel interferogram with magnitude displayed in gray-scale brightness and wrapped phase in color. The biased interferogram coherence estimate $\hat{\rho}$, as calculated from Eq. (4.11), is shown in Fig. 7.1(b). Figure 7.1(c) shows the USGS reference DEM. More detail about both the interferogram and DEM are found in Chapter 3.

The topography in this interferogram presents several difficulties for phase unwrapping. Running in azimuth in the middle of the image are long phase discontinuities resulting from layover, and at the top of the image there are areas of rough topography. The bottom is relatively smooth, although it is not without areas of low correlation.

Figure 7.2 depicts the result of applying SNAPHU to this data set. As in Fig. 3.3, the color represents relative unwrapped phase error, calculated by subtracting the DEM-derived unambiguous phase from the algorithm solution. The gray-scale brightness again shows the interferogram magnitude. Since the algorithm produces a congruent solution, unwrapping

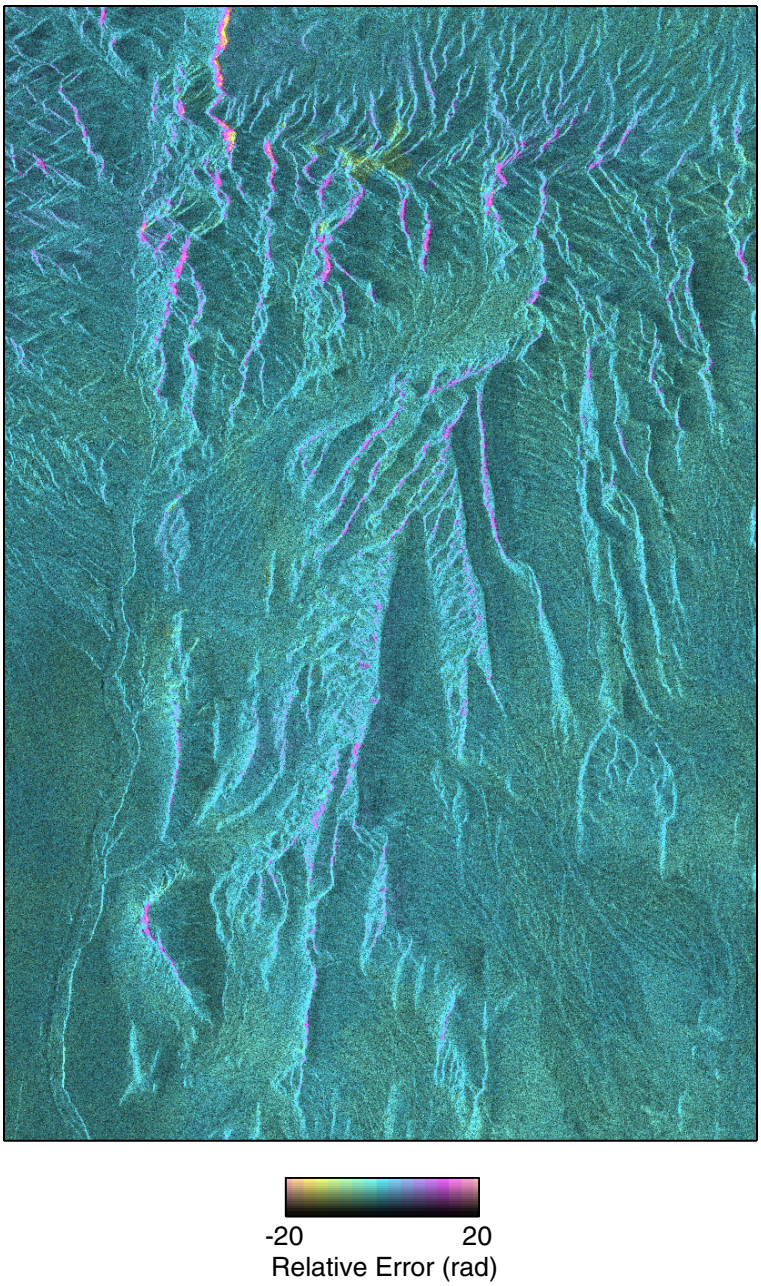

Figure 7.2 SNAPHU results with Death Valley test data. Relative phase error with respect to the reference DEM is shown in color, and the interferogram magnitude is shown in gray-scale brightness. There are very few significant phase unwrapping errors, in contrast to the results of Fig. 3.3.

errors are easily identifiable as patches differing from their surrounding areas by integer numbers of cycles. Other differences of less than one cycle may be due to atmospheric effects [*Zebker et al.,* 1997], inaccuracies in the DEM, noise in the interferogram, or artifacts from transforming and registering the DEM to radar coordinates. Interferometric height measurements may also be slightly different than those derived from other sources since the observed radar echoes may have scattering centers that lie above or below the ground surface. These effects are probably minor for the desert region shown here, but they may be more significant in areas that are covered by vegetation canopies or characterized by significant microwave ground penetration (*e.g.*, icy surfaces).

As is evident from the general homogeneity of color in Fig. 7.2, our algorithm produces an accurate unwrapped solution, with errors predominantly confined to areas of layover. Overall, 94% of the pixels in the solution are within π rad of the reference phase, and the rms error is 1.92 rad (0.31 cycles). Since few 2π jumps are apparent in the image, most of these errors can be attributed not to the phase unwrapping process, but to the error sources described above.

Figure 7.3 shows the relative phase error resulting from the use of our MST algorithm. In our implementation of the algorithm, we calculate the scalar, integer weight for each arc from its corresponding MAP cost function as follows. We assume that the arc does not have any existing flow on it, and we compute the incremental costs of adding single units of flow to the arc in either direction. We then set the MST arc weight to the lesser of these two costs, or to unity if weight would otherwise be nonpositive. As illustrated by Fig. 7.3, the MST algorithm performs fairly well with such weights. There is a significant unwrapping error at the top of the interferogram in the area of rough topography, but the algorithm handles the layover discontinuities in the middle of the image quite well. Overall, 89% of the pixels in the solution are within π rad of the reference phase, and the rms error is 2.67 rad (0.43 cycles).

The quality of the MST algorithm's solution is noteworthy because this algorithm is used as the initialization routine in our implementation of SNAPHU. That is, SNAPHU uses the MST algorithm to compute an initial feasible solution for the pivot-and-grow solver. The latter solver then makes iterative improvements to this solution in terms of the full MAP cost functions. A comparison of Figs. 7.2 and 7.3 thus demonstrates that the pivot-and-grow solver corrects many of the errors in the MST initialization. On other test interferograms (not shown), the pivot-and-grow solver corrects more significant errors in the initialization

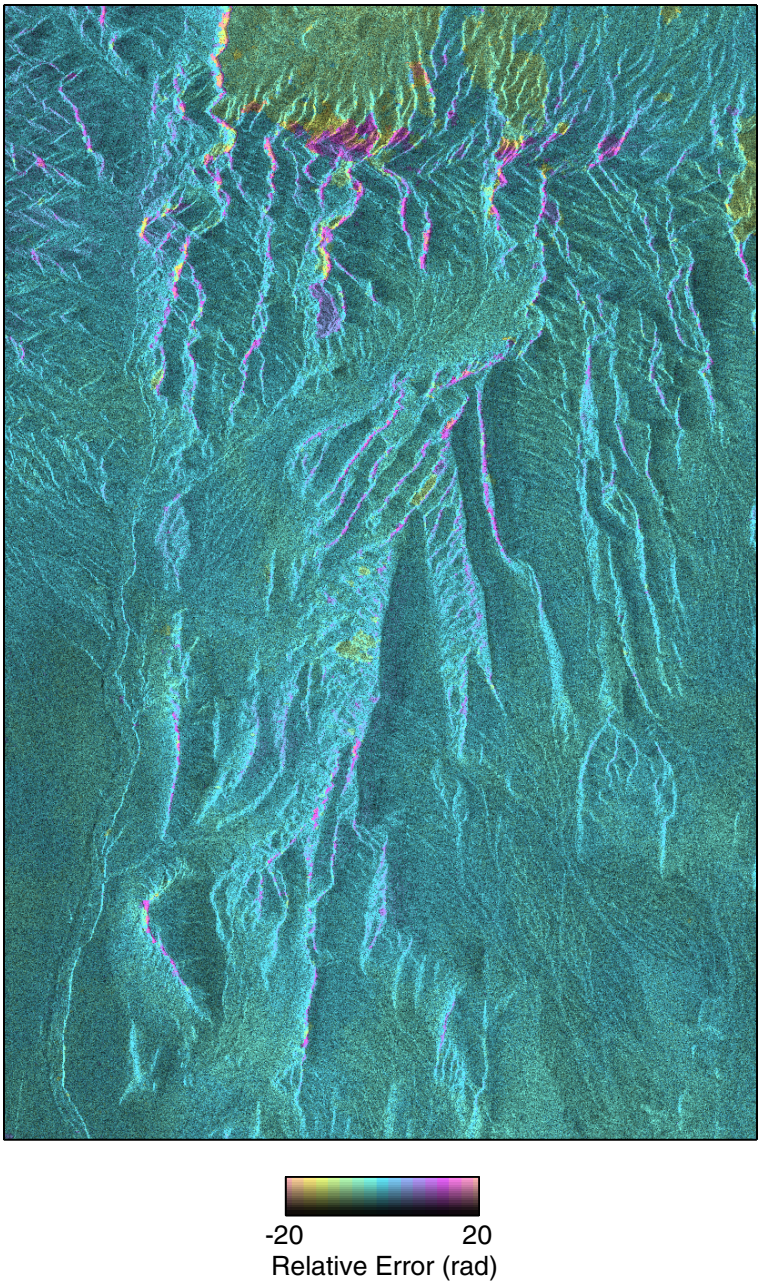

Figure 7.3 MST results with Death Valley test data. Relative phase error with respect to the reference DEM is shown in color, and the interferogram magnitude is shown in gray-scale brightness. A significant phase unwrapping error is visible in the area of rough topography at the top of the image, but the rest of the interferogram is unwrapped fairly well.

Algorithm	Execution Time (s)	Memory Usage (MB)
Residue-cut	12	10
MST	50	60
SNAPHU	210	90
Least-squares	220	50
MCF	350	400
LPN	1960	70

Table 7.1 Algorithm execution times and computer memory requirements for Death Valley test data on a Hewlett-Packard C-180 workstation with a 180 MHz PA8000 processor.

as well, although its execution time increases as the quality of the initialization degrades.

For comparison with existing L^p algorithms, Fig. 7.4 shows SNAPHU and MST results side by side with results from Chapter 3 of the least-squares, MCF, LPN, and residue-cut algorithms. Details about the specific implementations of those algorithms are given in Chapter 3. We must note, however, that the least-squares, MCF, and LPN algorithms are used here with simple correlation weights, and the residue-cut algorithm does not accept weights at all. It might therefore be argued that the comparison of Fig. 7.4 is not quite fair since different weighting schemes are used for the different algorithms. Indeed, some L^p algorithms do perform better with other weighting schemes. In particular, weighting schemes based on our statistical models, as used with the MST algorithm above, lead to more accurate solutions than those shown. The real issue, though, is that most current L^p phase unwrapping algorithms are solver routines exclusively; they make no specification of how the problem is to be set up. The dependence of such algorithms' results on the choice of weights simply underscores the importance of setting up the problem as well as solving it. Moreover, since the correlation weights used in Fig. 7.4 are a reflection of the current state of the art [*Ghiglia and Pritt,* 1998], the comparison emphasizes the contributions of this research.

SNAPHU clearly yields the most accurate solution for this data set. Moreover, the data suggest that the MST algorithm may be useful by itself as a quick, fairly accurate alternative for easy-to-unwrap interferograms.

Table 7.1 summarizes the approximate execution times and storage requirements of the six algorithms compared in Fig. 7.4. Note that most of the MST algorithm's execution

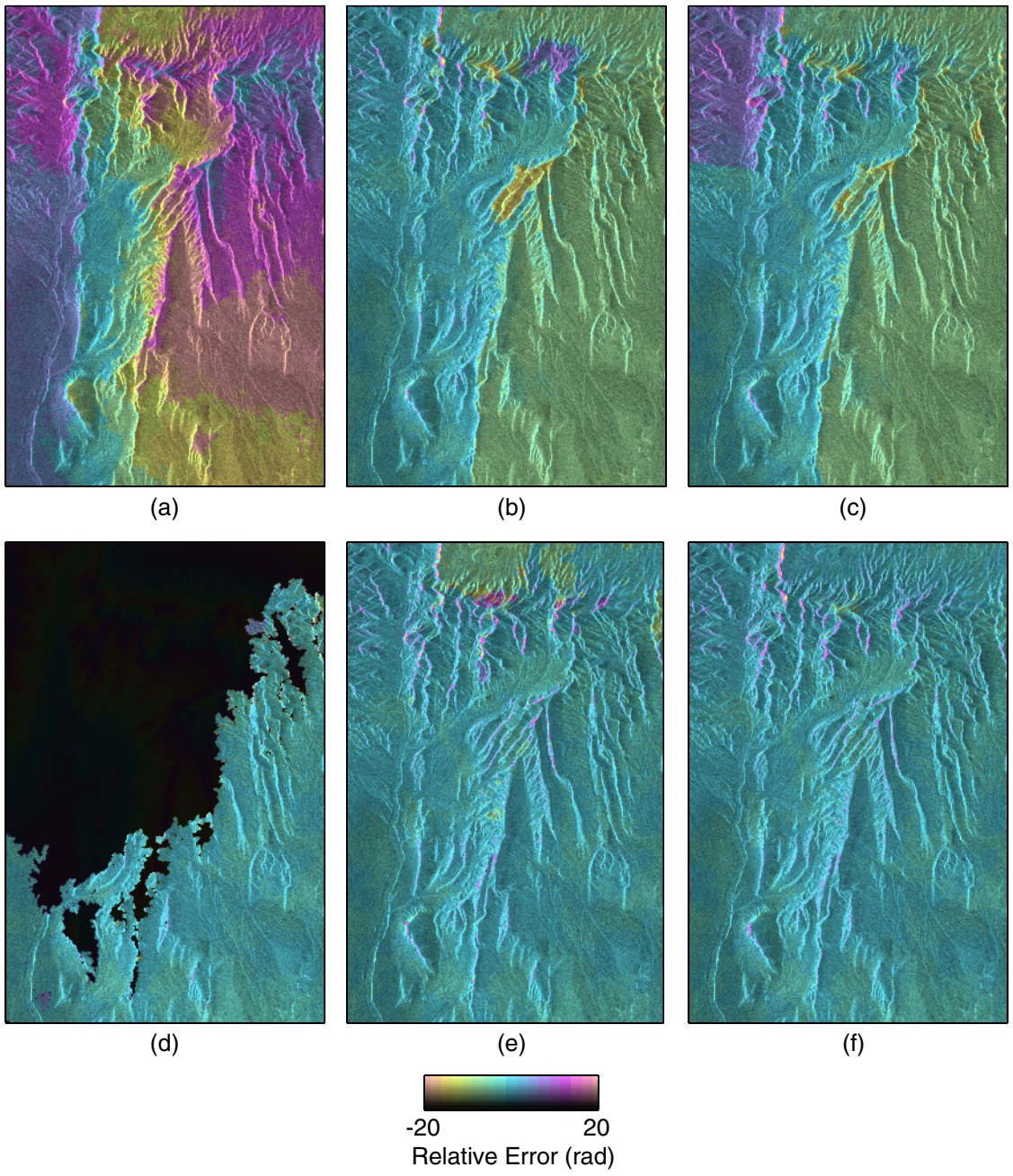

Figure 7.4 Relative unwrapped-phase errors for the Death Valley test data from different algorithms: (a) least-squares; (b) MCF; (c) LPN; (d) residue-cut; (e) MST; (f) SNAPHU. The interferogram magnitude is shown in gray-scale brightness. Black areas in (d) indicate that no solution was produced.

time is spent computing its statistical weights. With uniform weights, the MST algorithm runs slightly faster than the residue-cut algorithm. That said, we must point out that these measures of computational efficiency are telling only for the specific algorithm implementations used here; other implementations may behave quite differently. Furthermore, the least-squares and LPN algorithms are intensive in floating-point computations, while SNAPHU and the MST, MCF, and residue-cut algorithms are generally more intensive in integer operations. Their relative speeds consequently depend upon the platform on which they are run.

7.2 Deformation Data: The Hector Mine Earthquake

While DEMs provide reliable ground truth for topography data, reference data are more difficult to come by for deformation data. However, the subtleties and nuances of real data are also more difficult to simulate faithfully in the latter case, so we again use a real interferogram for testing SNAPHU. We evaluate the unwrapped solution subjectively by examining its geophysical plausibility.

The 2380×2548 pixel interferogram shown in Fig. 7.5(a) depicts the deformation signature resulting from the M7.1 Hector Mine, California earthquake of October 16, 1999. The data were acquired by ERS-2 during orbits 23027 and 23528. Topographic phase variations arising from this pair's -23 m perpendicular baseline have been removed as much as possible with the use of ERS-1/ERS-2 tandem data from orbits 24664 and 4991. As elsewhere in this chapter, the interferogram magnitude is indicated by gray-scale brightness and the phase by color. Through Eq. (4.34), each color cycle represents 2.8 cm of relative line-of-sight surface displacement. Note that the interferogram fringe rates are in the same direction on either side of the earthquake fault, so across the fault the phase changes by tens of cycles over a narrow spatial strip. The peak deformation is greater than 20 cycles, or 56 cm. While some areas along the fault generally appear darker in the interferogram magnitude, such magnitude variations can be attributed to correlation effects. The individual SAR intensity images (not shown) do not suggest any easily discernible relationship between deformation and brightness. Because areas along the fault are completely decorrelated, however, the fault is plainly visible in the coherence image of Fig. 7.5(b). This coherence estimate is formed from twenty looks in azimuth and four looks in range, while ten looks in azimuth and two looks in range are incorporated into the interferogram. Range is shown increasing

towards the left.

The unwrapped solution from SNAPHU is shown in Fig. 7.5(c). The boxed part of this image is shown enlarged in Fig. 7.5(d) without the interferogram magnitude and re-wrapped modulo 40 rad (6.37 cycles) for display purposes. Away from the fault where fringe rates are low, the interferogram is easy to unwrap and virtually any algorithm would be expected to perform well. Nearer the fault, the fringe rates increase and coherence decreases, but the solution still does not present any apparent errors. The relatively smooth unwrapped solution is consistent with the intuition that shear or fracture in the surface should be limited to few places, mostly along the fault. At those locations, the algorithm allows large phase gradients and discontinuities, so it also captures the high-spatial-frequency information of the interferogram. SNAPHU thus avoids the distortions characteristic of algorithms prone to excessive smoothing or slope underestimation [*Bamler et al.,* 1998; *Zebker and Lu,* 1998]. Where coherence is lost completely in the interferogram, the algorithm produces a (reasonable) guess at the unwrapped phase field based on the surrounding valid areas. The overall accuracy of these deformation measurements thus seems to be limited more by the quality of the interferogram than by the accuracy of the unwrapping algorithm.

The interferogram was unwrapped using one of the two 550 MHz Intel Pentium III processors of a Dell Precision 610 workstation. The execution time and memory requirement were approximately 1500 s and 490 MB.

7.3 A Large-Scale Data Example: The Alaska DEM Project

We now examine SNAPHU's performance on three large topographic interferograms from the Alaska SAR Facility's Alaska DEM project. This project entails the use of ERS-1/ERS-2 tandem data for mapping the state's topography, and because Alaska covers a large geographical area with a great deal of rugged terrain, the project poses a difficult challenge for phase unwrapping. The data thus provide a thorough test of our algorithm's feasibility for such applications.

For each of these three interferograms, we compare our algorithm's unwrapped results to an unambiguous simulated unwrapped phase field generated from a reference DEM (also obtained from the Alaska SAR Facility). Because the 60 m posting of this DEM is relatively coarse, systematic disparities between the interferometric data and the reference data are

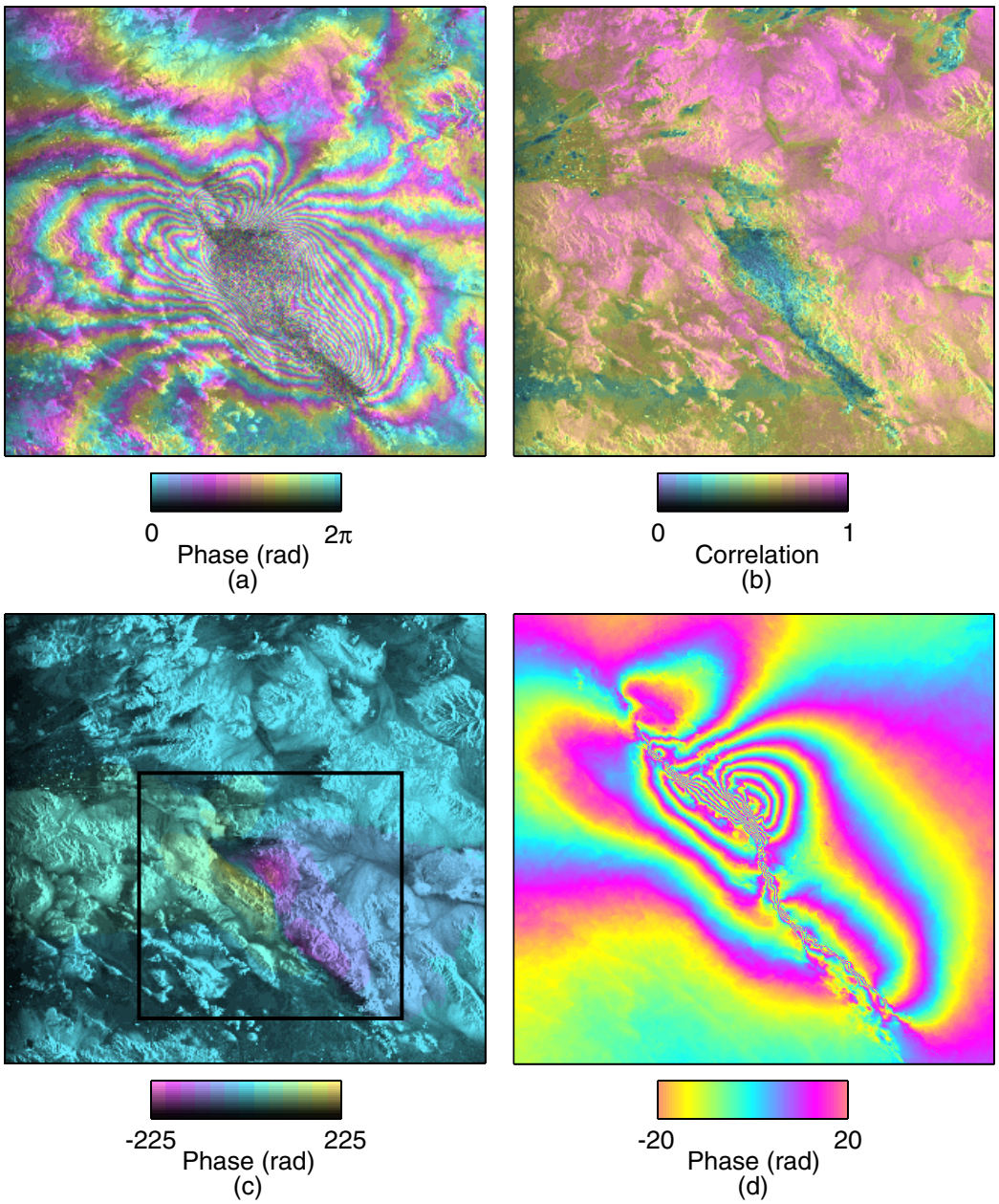

Figure 7.5 Deformation test data and SNAPHU results: (a) interferogram with wrapped phase in color and magnitude in gray-scale brightness; (b) biased coherence magnitude in color with magnitude in gray-scale brightness; (c) unwrapped solution from our algorithm, with unwrapped phase in color and magnitude in gray-scale brightness; (d) unwrapped phase of the area indicated in (c), shown wrapped modulo 40 rad (6.37 cycles) for display.

more significant than in the Death Valley example above. Nevertheless, the DEM is adequate for the identification of most significant phase unwrapping errors which again appear as integer-cycle jumps.

Because many of the images shown in this section are quite large, we have greatly reduced their resolutions for reproduction purposes. Note, however, that all processing was done on the large, high-resolution data files. All interferograms consist of five looks in azimuth and one look in range, and all coherence maps are computed from twenty looks in azimuth and four looks in range. Additionally, all images are shown in radar coordinates (slant range and azimuth). The images are consequently rotated and slightly distorted with respect to their true geographical orientations.

Our first test interferogram, shown in Fig. 7.6, depicts an area surrounding the city of Fairbanks. This 7304×4800 pixel interferogram is formed from data taken during ERS-1/ERS-2 orbits 23942 and 4269, an ascending pair with a perpendicular baseline of approximately 139 m. Range is shown increasing towards the right. As shown in Fig. 7.7, the interferogram coherence is reasonably high in flatter areas, but decreases substantially in areas of rough topography.

For unwrapping, the interferogram was divided into 70 separate tiles, comprising a 10×7 array, with neighboring tiles overlapping by 20 pixels in either dimension. After SNAPHU was applied to each tile, reliable regions were identified and assembled as described in Chapter 6. Results are shown in Fig. 7.8. Color in this image represents relative unwrapped phase error with respect to the reference DEM. A number of slowly varying differences between the unwrapped solution and the DEM-derived reference phase are evident in the error image, but the lack of integer-cycle jumps suggests that these differences are due to the sources described in Section 7.1 rather than to phase unwrapping errors. Unwrapping errors are found only in areas of layover and extreme foreshortening, and there are no apparent tiling artifacts. Our approach thus performs quite well overall on this test case.

Our second test interferogram is shown in Fig. 7.9, and its coherence map is shown in Fig. 7.10. Range is shown increasing towards the left. These images consist of data acquired during ERS-1/ERS-2 orbits 24222 and 4549, a descending pair with a perpendicular baseline of approximately 98 m. The interferogram again depicts the Fairbanks area, but because it is formed from a descending rather than an ascending image pair, the viewing direction is different than that of the previous interferogram. The scene thus appears at a different orientation in radar coordinates, making the data of Figs. 7.6 and 7.9 complementary.

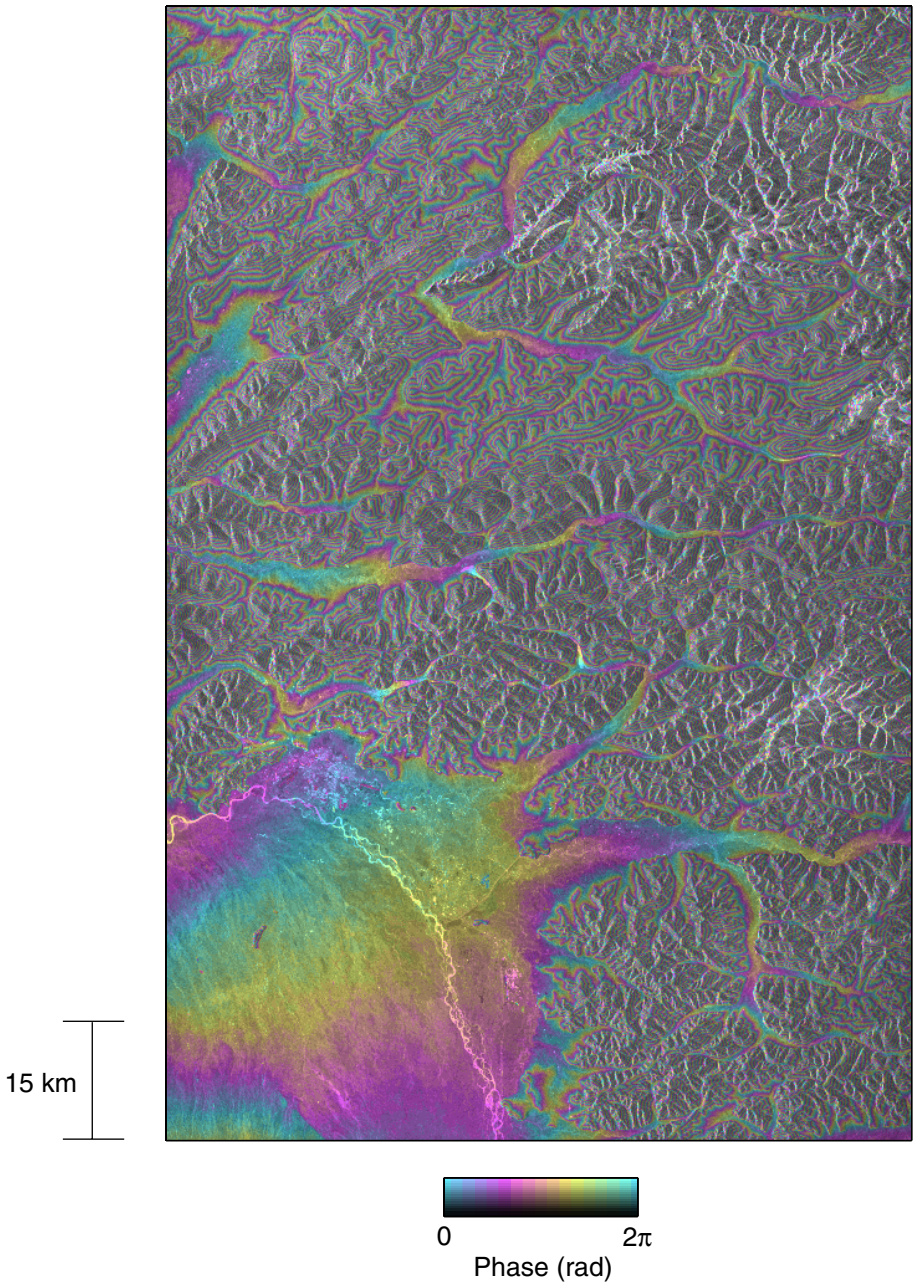

Figure 7.6 Alaska test interferogram from ERS pair 23942-4269. The interferogram phase is shown in color and the magnitude is shown in gray-scale brightness. Range increases towards the right.

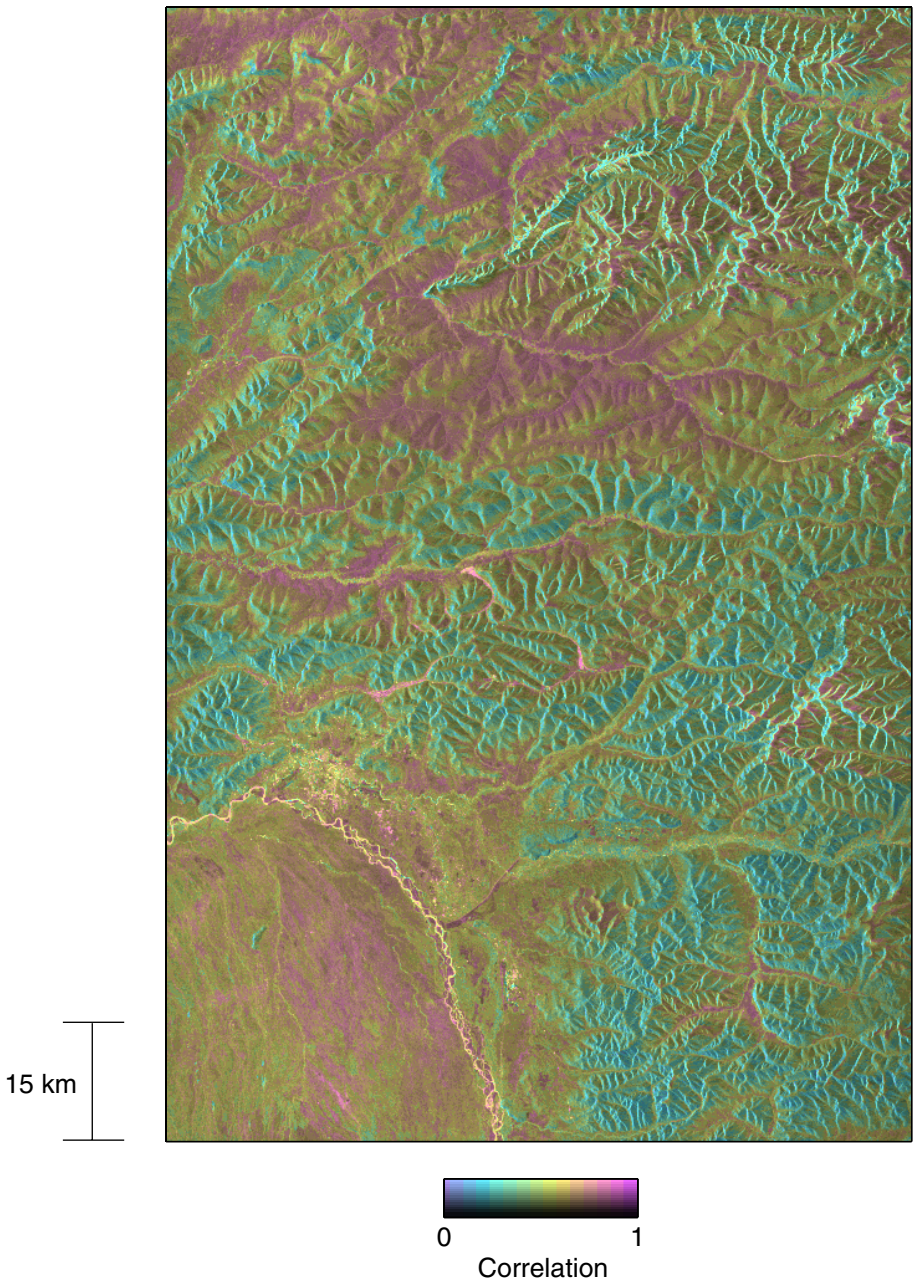

Figure 7.7 Coherence map for the 23942-4269 interferogram. The estimated coherence is shown in color and the interferogram magnitude is shown in gray-scale brightness.

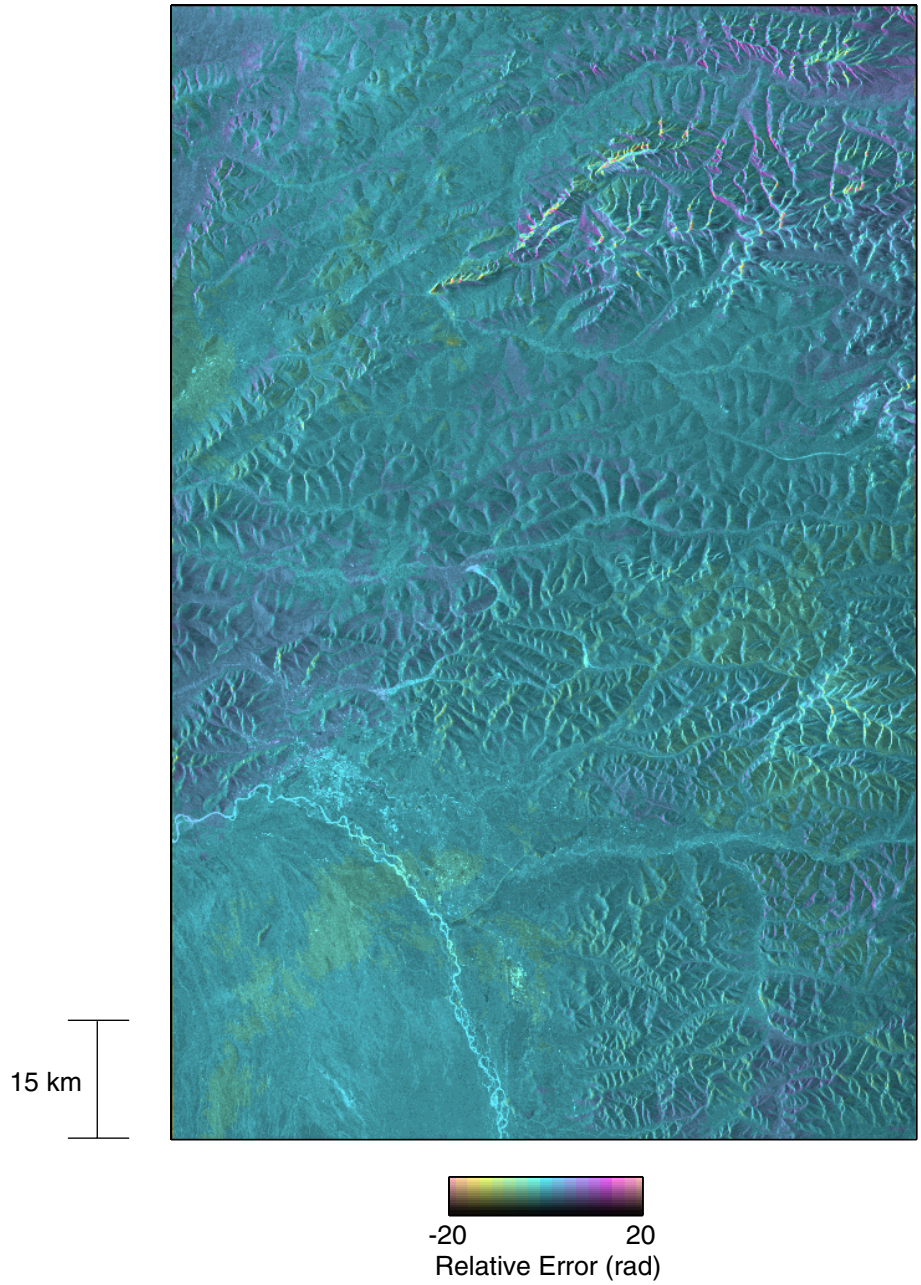

Figure 7.8 Relative unwrapped phase error for the 23942-4269 interferogram. Some minor unwrapping errors can be found in areas of layover and extreme foreshortening near the upper right part of the image, though most visible color variations are likely not due to phase unwrapping. The interferogram magnitude is shown in gray-scale brightness.

That is, at geographic locations where layover or extreme foreshortening make the data in one interferogram unusable, good data are often available in the other interferogram. Since we are concerned mainly with phase unwrapping, however, we do not make a direct comparison of the two images' unwrapped solutions here. Doing so would require that the data be geocoded and might consequently make the algorithm results more difficult to evaluate. We do compare the interferograms to the same reference DEM (projected into the interferograms' respective coordinate systems), so we can still verify that the algorithm produces consistent results.

The interferogram of Fig. 7.9 is the same size as that of Fig. 7.6 (7304×4800 pixels), so the same tiling parameters were used to unwrap it. Results are shown in Fig. 7.11. Again, phase unwrapping errors are confined mainly to areas of layover, extreme foreshortening, or low correlation, and no tiling artifacts are visible.

Although the two interferogram frames examined in this section so far have been fairly large, we now examine a swath-mode interferogram that is larger still. The 23240×4800 interferogram of Fig. 7.12(a) is formed from data acquired during ERS-1/ERS-2 orbits 22210 and 2537. Range is shown increasing towards the right. With its swath covering an area approximately 460 km long and 100 km wide, the interferogram contains a variety of terrain types, from well-correlated flatlands to rugged mountains in layover. Owing to its large size and perpendicular baseline of approximately -136 m, however, fringes in some areas are so finely spaced that they are not visible at the resolutions used for reproduction. The interferometric correlation is shown in Fig. 7.12(b).

The interferogram was unwrapped as a 32×7 array of overlapping tiles to obtain the results shown in Fig. 7.12(c). Most of the interferogram is correctly unwrapped. There are no large-scale errors, though some localized errors are apparent near the rugged terrain in the middle of the image. A few tiling artifacts are also evident upon close inspection of this area; they appear as straight, abrupt, integer-cycle jumps at tile boundaries. Overall, however, our region-assembly approach performs well given the difficulty of the interferogram. Of the visible unwrapping errors, some are likely due to phase unwrapping, while others are likely due to poor correlation and other sources as described above. Note that the area near the upper left corner of Fig. 7.12(c) is masked out because reference data there were missing. SNAPHU always produces a complete solution.

Figures 7.13–7.15 show enlarged the indicated parts of the interferogram, coherence map, and error image of Fig. 7.12. This 7304×4800 part of the swath contains a section

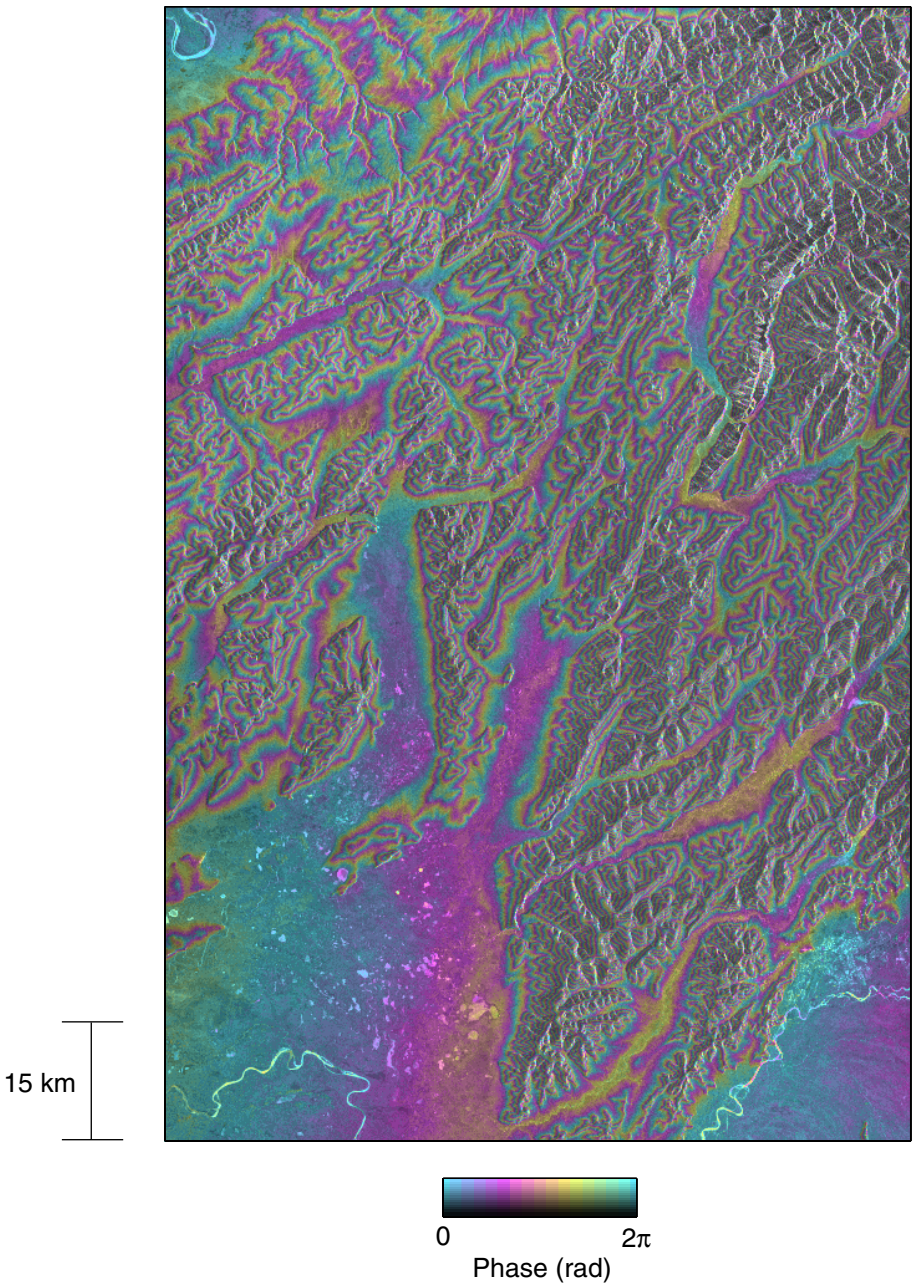

Figure 7.9 Alaska test interferogram from ERS pair 24222-4549. The interferogram phase is shown in color and the magnitude is shown in gray-scale brightness. Range increases towards the left.

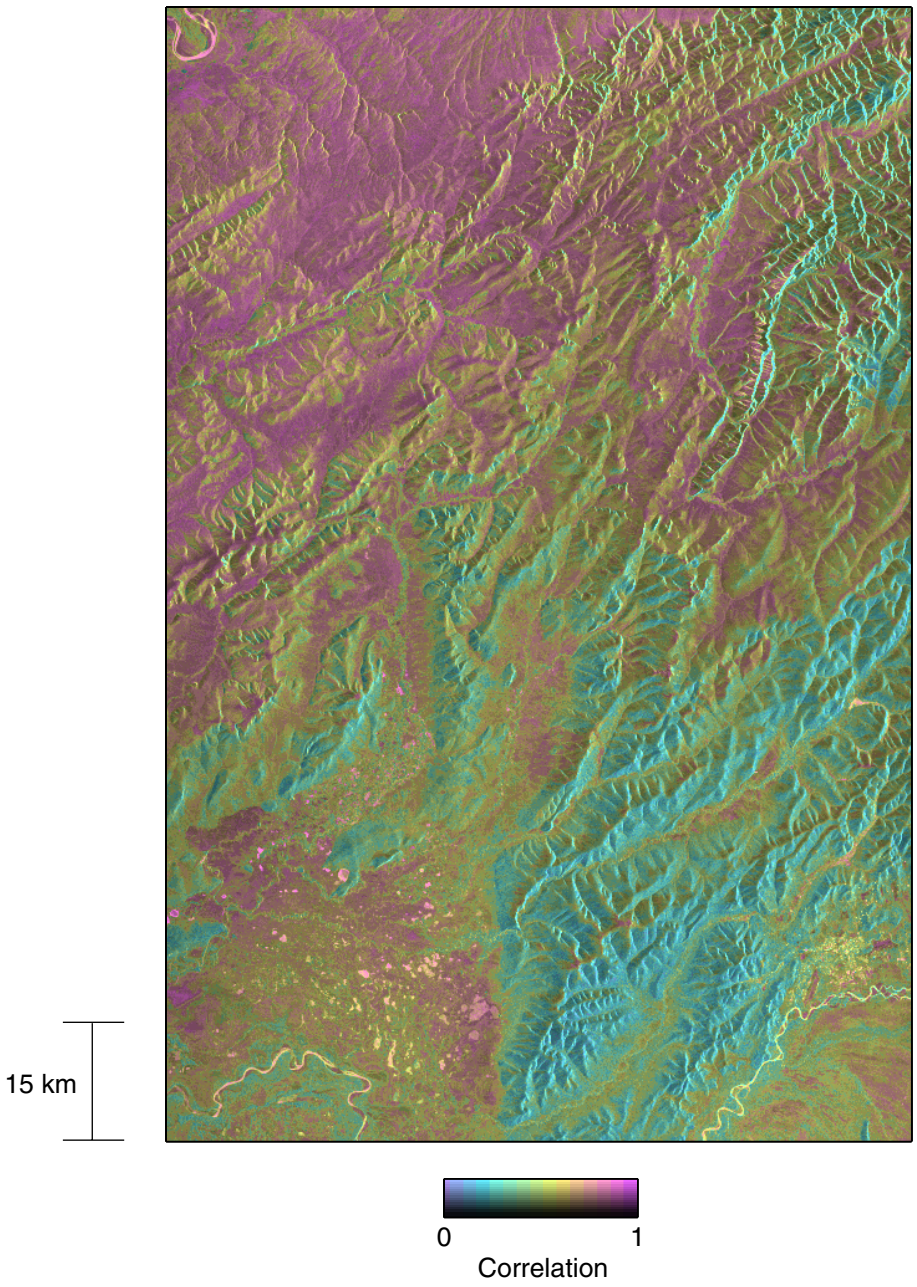

Figure 7.10 Coherence map for the 24222-4549 interferogram. The estimated coherence is shown in color and the interferogram magnitude is shown in gray-scale brightness.

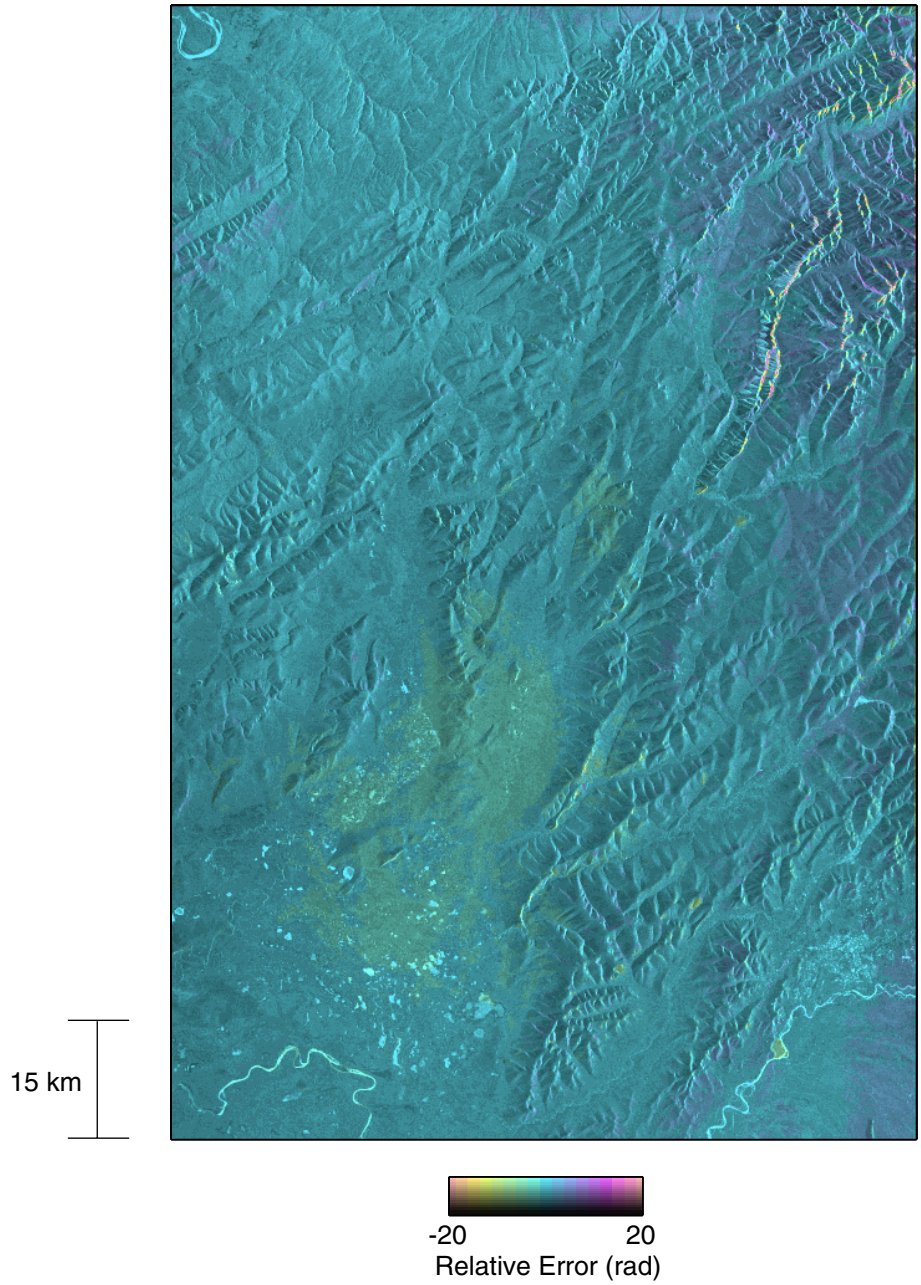

Figure 7.11 Relative unwrapped phase error for the 24222-4549 interferogram. Some minor unwrapping errors can be found in areas of layover and extreme foreshortening near the upper right part of the image, though most visible color variations are likely not due to phase unwrapping. The interferogram magnitude is shown in gray-scale brightness.

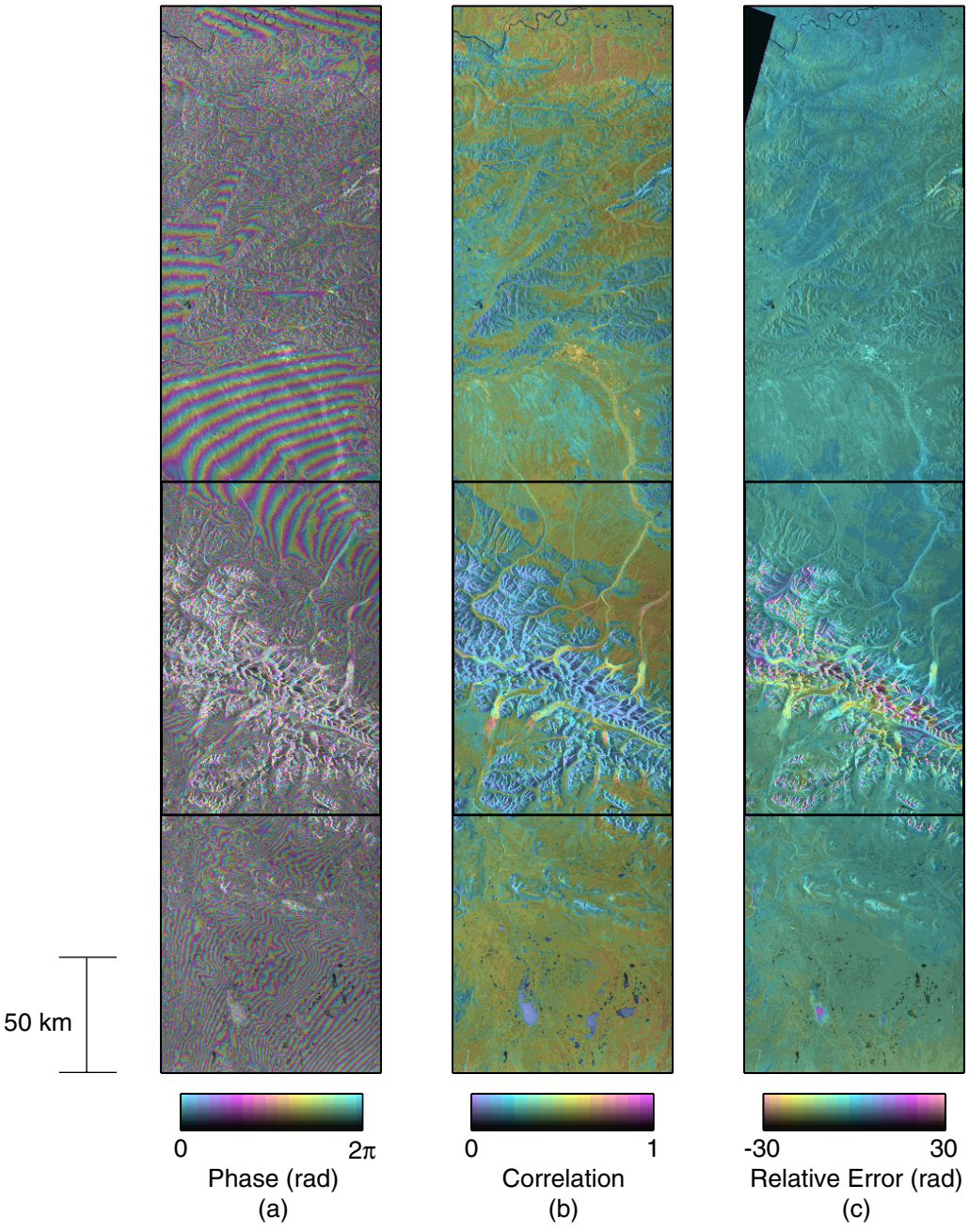

Figure 7.12 Test data and results for the 22210-2537 pair: (a) wrapped phase; (b) coherence estimate; (c) relative unwrapped phase error. An area at the upper left of the error image is masked out because reference data there were missing. All panels depict the interferogram magnitude in gray-scale brightness. The boxed areas are shown enlarged in Figs. 7.13–7.15.

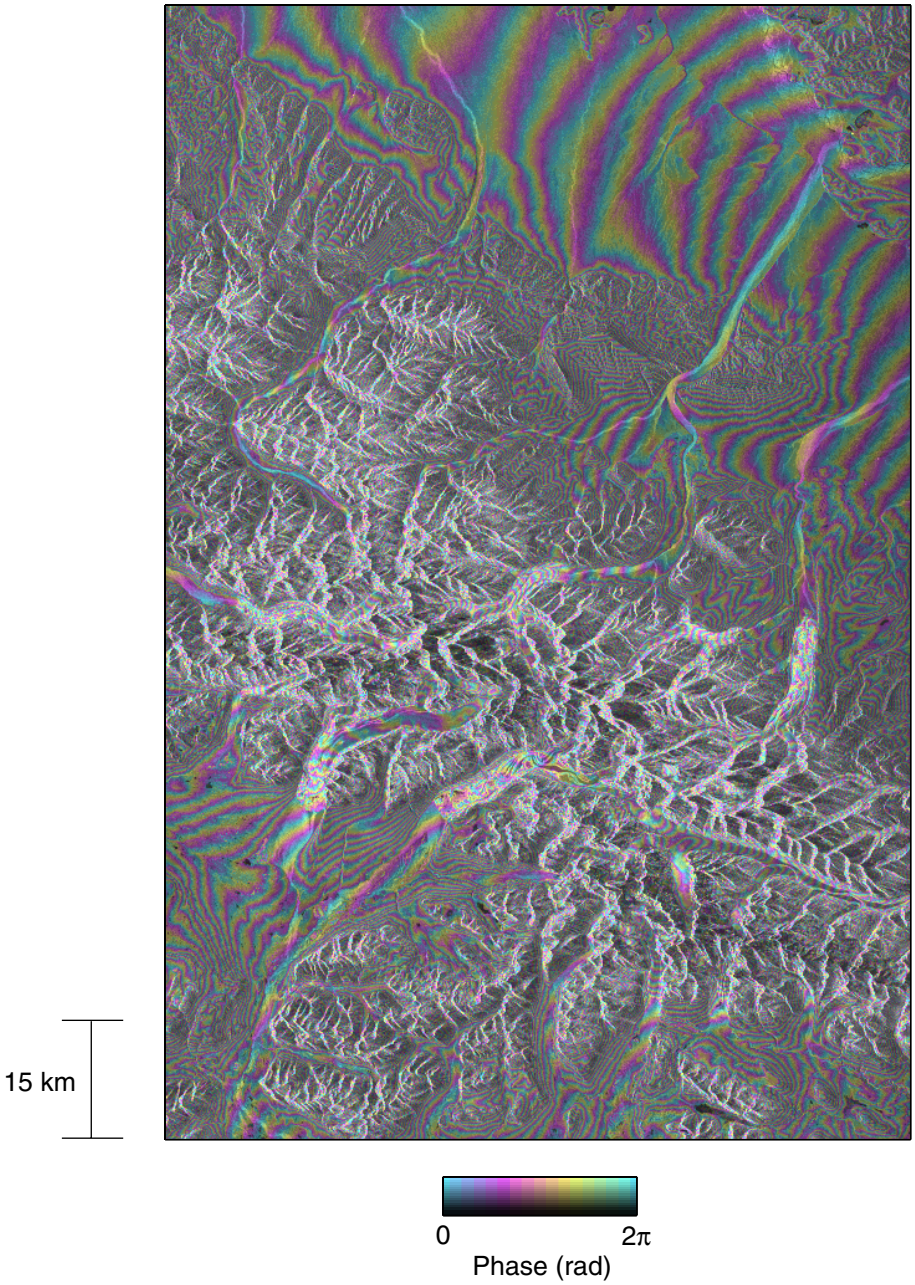

Figure 7.13 Enlargement of the boxed part of the 22210-2537 interferogram shown in Fig. 7.12(a).

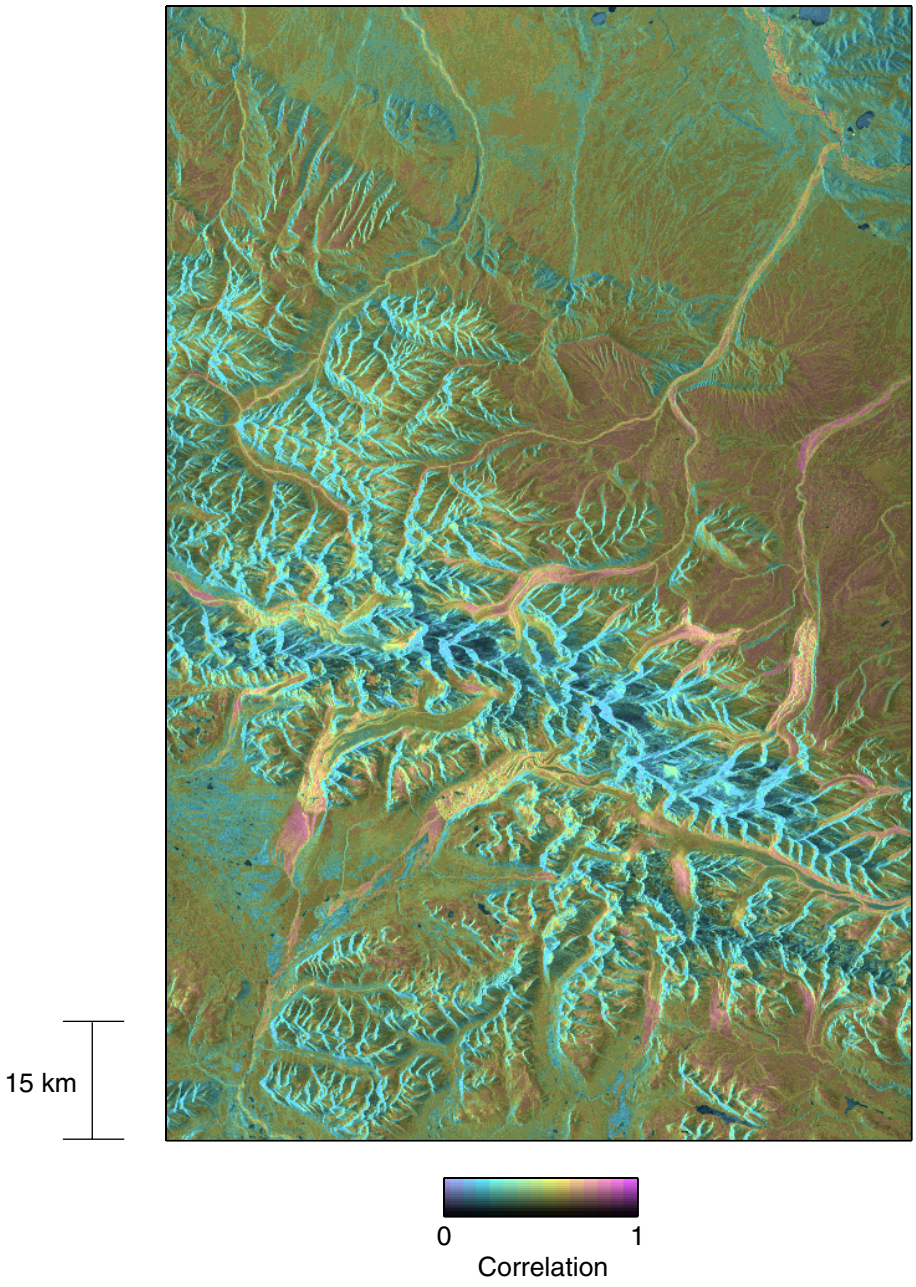

Figure 7.14 Enlargement of the boxed part of the 22210-2537 coherence map shown in Fig. 7.12(b).

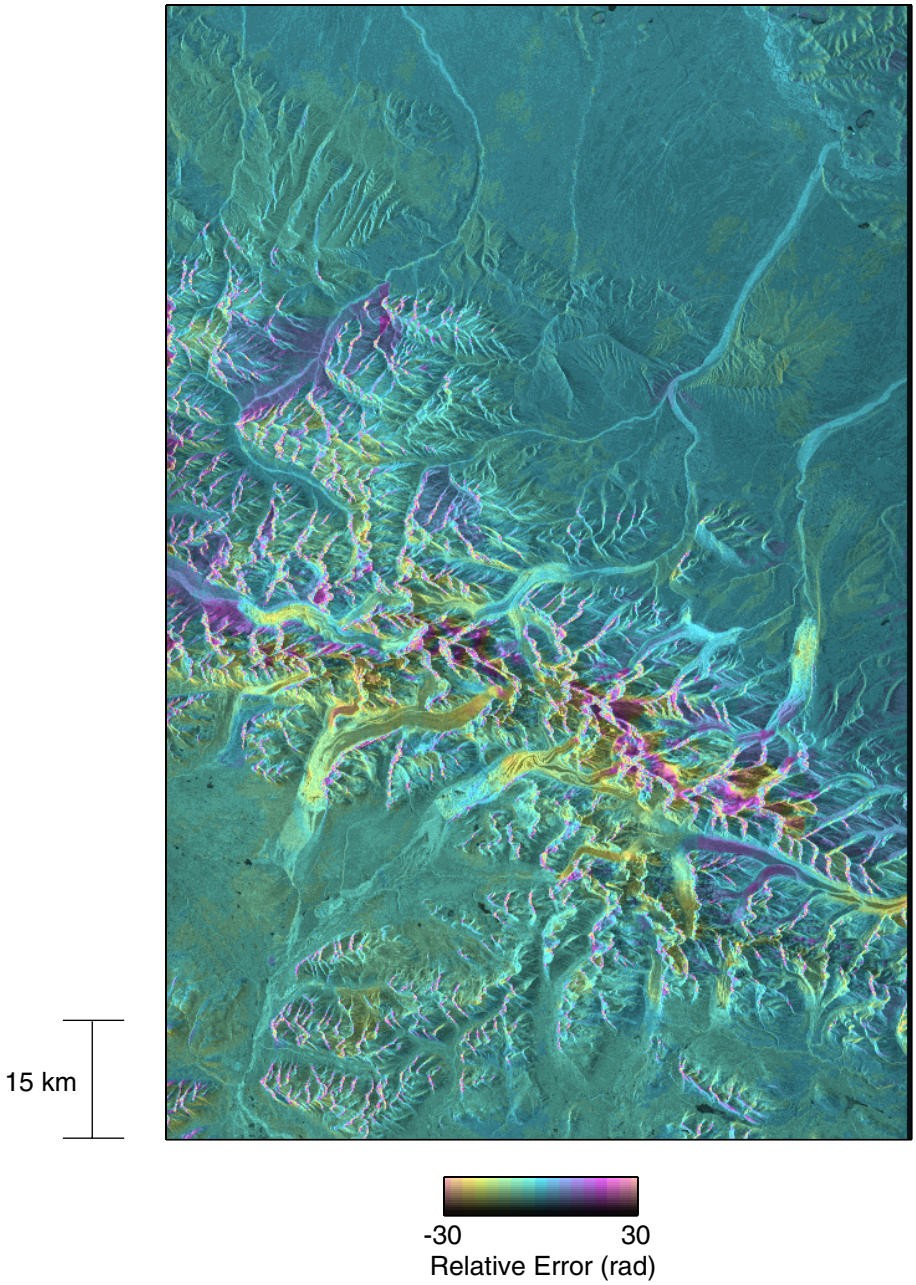

Figure 7.15 Enlargement of the boxed part of the 22210-2537 error image shown in Fig. 7.12(c).

Orbit Pair	Interferogram Size	Elapsed Real Time	Total CPU Time
23942-4269	7304×4800	0:16:07	0:52:10
24222-4549	7304×4800	0:16:13	0:50:24
22210-2537	23240×4800	2:13:33	8:22:14

Table 7.2 Execution times for the Alaska test data. The third and fourth columns list the real (wall-clock) time and the aggregate processor time spent unwrapping the interferogram with parallel use of four 500 MHz Intel Pentium III processors on a Dell PowerEdge 6300 server.

of the mountainous Alaska Range, so the topography in the region is characterized by low correlation, layover, and extreme foreshortening. Moreover, many of the mountain peaks are decorrelated on their back slopes as well as their front slopes, probably because of temporal effects such as changes in snow cover. Together, these traits make phase unwrapping very difficult, so the errors of Fig. 7.15 are not altogether unexpected. In fact, this area accounts for almost all of the unwrapping errors visible in Fig. 7.12(c).

Such errors might indeed lead to severe height inaccuracies in an interferometrically derived DEM, but the utility of our unwrapping algorithm should not be dismissed. The absence of more extensive errors is telling, and it is not clear that any alternative algorithms would perform better on such data. Furthermore, a number of additional processing techniques might enhance the algorithm's performance, as described in the next section.

All interferograms in this section were unwrapped on a Dell PowerEdge 6300 server. Table 7.2 lists each interferogram's required execution time with parallel use of the server's four 500 MHz Intel Pentium III processors. In all cases, each tile required approximately 52 MB of memory.

The algorithm's relatively short execution times for the first two interferograms suggest that it deals well with large data sets of moderate unwrapping difficulty. The swath-mode interferogram requires somewhat more time, even accounting for its larger size, but it is also much harder to unwrap. The greatest potential gains in solution accuracy are thus also associated with such data. Furthermore, a disproportionately large amount of this interferogram's total processor time is spent on the tiles containing the rugged Alaska Range. Given the relationship between phase unwrapping difficulty and required execution time, the accuracy-enhancement techniques described below might thus improve overall efficiency as well.

7.4 Preprocessing Techniques for Topographic Data

We now describe a few preprocessing techniques that might enhance SNAPHU's phase-unwrapping accuracy for topographic interferograms such as those of the previous section. One such technique is phase filtering. *Goldstein and Werner* [1998] proposed a nonlinear, adaptive filter for the express purpose of making phase unwrapping easier. Its effects on SNAPHU's statistical models can be examined in future research.

DEM flattening might also make topographic interferograms easier to unwrap. In this approach, the simulated unwrapped phase field from an existing low-resolution DEM is subtracted from the wrapped interferometric phase before unwrapping then reinserted afterwards. This procedure reduces fringe rates and residue densities in the wrapped phase, simplifying the problem. Since the subtraction of a constant from a random variable simply translates the random variable's PDF, DEM flattening has the effect of shifting the statistical cost functions of Chapter 4 along the $\Delta\phi$ axis.

The cost functions might also be made to better reflect the statistics of the data with more accurate estimation of the interferometric baseline. Slight baseline errors can cause phase tilts or height overestimation in the data, making fringe rates and residue densities in some areas higher than expected. Thus, although it is sometimes easier to correct for baseline errors after unwrapping, slightly better results might be obtained if baseline adjustments are made beforehand. These and other techniques might help SNAPHU avoid errors like those found in Fig. 7.12(c).

Chapter 8

Conclusions

Despite considerable progress, 2-D phase unwrapping remains a formidable challenge. As the problem is attacked in new and different ways, however, important insights are gained into both the nature of the problem and algorithms for its solution. Our work here is meant to further this process of innovation and improvement.

8.1 Contributions

In this dissertation, we describe a new approach to 2-D phase unwrapping for SAR interferometry. This approach and the ideas behind it comprise the key contributions of our research:

1. Proof of the intractability of the general phase unwrapping optimization problem. The problem is shown to be *NP*-hard, suggesting that efforts towards solving it are best spent on approximate rather than exact algorithms.

2. Suggestion of a MAP framework that provides a physical basis for the phase unwrapping optimization problem. Phase unwrapping is posed as a statistical estimation problem in which the goal is to find the most probable unwrapped solution given the observed data. The statistical framework generalizes many existing approaches since L^p objectives can be treated as coarse approximations to MAP objectives.

3. Derivation of models for approximating the joint statistics of interferometric SAR data, as required by the MAP framework, for the cases of both topography and deformation measurement. The models, quantified through the use of nonlinear cost

functions, reflect the theoretical relationships between unwrapped phase fields and observable quantities such as wrapped phase, image intensity, and interferogram coherence.

4. Design and implementation of efficient, nonlinear network-flow solvers for use with the statistical cost functions described above. Although the MAP cost functions cast phase unwrapping as an *NP*-hard optimization problem, the solver routines give good, albeit only approximate, results.

5. Development of a tiling heuristic for efficiently applying the above algorithm to large data sets. The heuristic extends the MAP framework, allowing large interferograms to be unwrapped in a theoretically justified and accuracy-preserving manner.

6. Evaluation of the performance of the above approach, as compared to existing techniques, on interferometric SAR data. The approach yields accurate results with respect to topographic and deformation-mapping test data. It is also competitive with existing algorithms in terms of computational efficiency.

8.2 Perspective

Phase unwrapping is an attempt to solve a problem we know to be unsolvable in any rigorous sense. Attempts at solving this problem are more than just academic exercises, however, as the difficulty of the problem does not diminish the need for reliable, real-world solutions. With this in mind, we have proposed in this dissertation a phase unwrapping approach, specific to SAR interferometry, that is based on MAP estimation and nonlinear network-flow techniques. Performance tests suggest great promise with this approach. Thus, while room indeed remains in the algorithm for both theoretical refinement and empirical improvement, the algorithm's accuracy and reasonable efficiency make it well worth considering for specific unwrapping applications.

The MAP framework implies, however, that no single algorithm will be best for all applications without modification. Since different physical quantities are involved for different applications, each application should have its own statistical models. Thus, while we might wish for the simplicity of a single, grand superalgorithm that is universally applicable, realistic algorithms must make specific assumptions about their inputs in order to achieve the greatest possible accuracy.

Assumptions involved in setting up the problem cannot be made independently of the method of solution, though. An optimization objective that is unsolvable is no more useful than one that gives unreliable results when solved. In the previous chapters, this duality guided the balance between theoretical rigor and computational manageability. The balance might need readjustment in other contexts, however. For example, statistically weighted L^p cost functions can serve as easily solvable, coarse approximations to more specific MAP criteria; their use may be appropriate in applications for which efficiency or solver simplicity are paramount. Conversely, the approximations used in our derivations may be refined in other cases to enhance analytical precision at the possible expense of solver complexity.

Different phase unwrapping algorithms—and perhaps even different implementations of the same algorithm—often have unique behavioral traits as well. Such characteristics, though sometimes subtle, might make some approaches better suited than others to certain classes of data. Thus, of the many approaches that are ultimately possible, it is still up to the user to know how and when to use each.

8.3 Future Directions

Many avenues for future work are possible. In this dissertation, we have used a MAP framework to propose both a new algorithm and a new paradigm for phase unwrapping. Improvements are possible in either. We identify in this section a few areas towards which the direction of future efforts might prove particularly beneficial.

An obvious starting point for further work lies in the refinement of the assumptions upon which the derivations of our statistical cost functions are based. Of course, increased theoretical rigor might make the resulting optimization problem harder to solve, but as we have noted above, improved unwrapping accuracy might justify additional complexity. Experimentation is needed to resolve this question.

Entirely new cost functions will also be needed if SNAPHU is to be used in other contexts. Specific statistical models will have to be derived and appropriate cost functions formed for each of the applications in which our technique is used. This is true whether the input data arise from other forms of SAR interferometry or from other measurement methods altogether.

New or better solver algorithms can be anticipated as well. Efforts at improving the network-flow routines described here may be warranted if greater flexibility in setting up

the problem or greater efficiency in solving it is desired. Nonlinear optimization routines based on concepts other than network flow are also possible.

Finally, a logical step beyond simple phase unwrapping might be the generalization of the MAP model itself. In this dissertation, we have used MAP ideas for the task of unwrapping a single interferogram; a much more powerful strategy might be to formulate a MAP framework that relates some desired quantity to *all* available input data. Phase unwrapping might then comprise only a small part of a larger, big-picture estimation problem. Such a problem might require the fusion of data from such disparate sources as multibaseline and multitemporal SAR interferometry, hyperspectral imagery, stereo photogrammetry, and global-positioning system ground measurements. Because the MAP framework would ensure that such data sets are amalgamated in a theoretically sound and physically meaningful way, the generalized MAP strategy might prove immensely beneficial. The complexity of such an approach might make its successful implementation very difficult, but the idea should be considered as a long-term goal.

Appendix A

NP-hardness of the L^0 Problem

In this appendix, we prove the *NP*-hardness of the L^0 phase unwrapping problem. We first show that any instance of an *NP*-complete called the rectilinear Steiner tree (RST) problem can be solved by an algorithm solving the network-equivalent L^0 problem. Then, we show that any instance of this network problem can be solved by a general L^0 phase unwrapping algorithm. We begin by formally stating the RST problem [*Garey and Johnson,* 1977; *Garey and Johnson,* 1979]: Let $S \subseteq Z \times Z$ be a set of points that have integer coordinates on a Euclidean plane, and let K be a positive integer; define an RST as a connected tree, composed solely of horizontal and vertical line segments, that contains all points in S; does there exist an RST whose total length is no larger than K? We show here that with the use of a hypothetical L^0 algorithm, we will always be able to find an RST of minimum length for any S. Clearly, we can then answer the question posed by the RST problem for any S and K.

The network equivalent of the L^0 phase unwrapping problem is to find a feasible flow, given some grid network (see Fig. 2.6), such that the total length of arcs carrying flow is minimized. Suppose for now that all arcs have unit length. Since the nodes of the network lie on a rectilinear grid, we can assume that they are located at integer points of a Cartesian coordinate system. Let n be the number of points in S. We construct an instance of the network problem such that there are n positive residues at nodes corresponding to the points in S, as in Fig. A.1. Suppose the rightmost positive residue is at (x_0, y_0). We line up n negative residues at $(x_0 + 1, y_0)$, $(x_0 + 2, y_0)$, $\ldots$, $(x_0 + n, y_0)$.

Now, we prove that the optimal RST length l_{RST} for the points in S is equal to $l_0 - n$ where l_0 is the total length of the L^0-optimal flow. We first show that the L^0 solution will

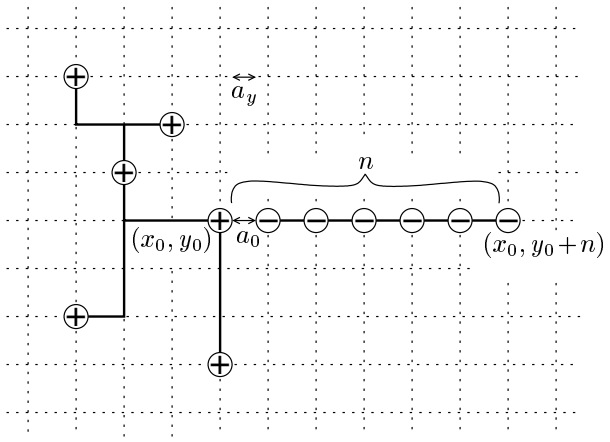

Figure A.1 An L^0 network formulation of an RST problem instance. Positive residues are located at the points in S, and negative residues are lined up to the right of the rightmost positive residue. The L^0-optimal solution must be a single minimum Steiner tree, whose length is n units longer than the minimum length for the original RST problem.

be a single tree. Consider the set of all arcs a_y going from (x_0, y) to $(x_0 + 1, y)$. Suppose there are m disconnected trees in the L^0 solution. Each tree must be neutral, so it must contain at least one distinct arc a_y in order to establish a connection between positive and negative residues. Let $y_{\max}$ be the y-coordinate of the uppermost of these arcs if any such arcs have $y > y_0$, or zero otherwise. Define $y_{\min}$ similarly for $y < y_0$. Define R as the set of all arcs which are part of the L^0-optimal solution on or to the right of the line $x = x_0$. A lower bound on the total length l_R of these arcs is expressed

$$l_R \geq n + (y_{\max} - y_0) + (y_0 - y_{\min}) + (m - 1). \tag{A.1}$$

The term n on the right side of this equation stems from the fact that at least one horizontal path must connect some positive residue to the rightmost negative residue. Flow must also traverse the vertical distances from $y_{\max}$ and $y_{\min}$ to the negative residues at y_0, explaining the next two terms. The last term includes one unit of length for each of the m arcs a_y used, excluding the one included in the first term. Now, observe that we can draw a single tree R' having length $l_{R'}$ and consisting of all arcs between (x_0, y_0) and $(x_0 + n, y_0)$ and all arcs between $(x_0, y_{\max})$ and $(x_0, y_{\min})$. R' contains all of the residues R contains, so if we substitute R' for R, we still have a feasible flow. Because we assume that the original

solution is optimal, l_R can be no greater than $l_{R'}$:

$$l_R \leq l_{R'} = n + (y_{\max} - y_0) + (y_0 - y_{\min}). \tag{A.2}$$

Together, the two inequalities imply that $m = 1$, so the L^0-optimal solution is a single tree. This tree must consequently be a minimum-length RST with respect to the set of all points at which there is either a positive or a negative residue. Furthermore, all positive residues are on a connected subtree and all negative residues are on another connected subtree, both of which are minimum Steiner trees for their respective residues [*Hanan,* 1966]. Clearly, the length of arcs from (x_0, y_0) to $(x_0 + n, y_0)$ is n, so the remaining length $l_0 - n$ of the L^0 solution is the length of the minimum rectilinear Steiner tree for the points in S. Knowledge of this length provides us with the answer to the given RST problem instance.

So far, we have shown that an L^0 network algorithm can solve the RST problem, so we must now show that any instance of the network problem has a phase unwrapping equivalent. That is, we must be able to generate a wrapped phase field for any instance of the network problem above. However, some arrangements of residues do not correspond to realizable phase fields (*e.g.*, when many residues of the same sign are all located at adjacent nodes). This calls for a slight modification to our formulation of the network RST equivalent. Suppose that the nodes of the network problem are now located at multiples of $1/n$ instead of 1 in the Cartesian coordinate system, while residues still lie at their integer coordinates. The rectilinear distance metric implies that the number of arcs in the L^0-optimal network solution simply increases by a factor of n [*Hanan,* 1966]. Therefore, we lose no generality in our ability to solve the RST problem. Now, however, any two residues are separated by at least n nodes, so we can easily (polynomially) generate a wrapped phase field for the given arrangement of residues. We might do so, for example, by wrapping some nonoptimal unwrapped surface into which we have introduced discontinuities of appropriate sign and magnitude along arbitrary paths between positive and negative residues. The resulting wrapped phase field has an L^0-optimal unwrapped-phase solution that corresponds exactly to the solution of the network problem. Therefore, an L^0 phase unwrapping algorithm can be used to solve any instance of the *NP*-complete RST problem.

This nearly completes the proof, but one more subtlety must be addressed. The number of nodes in our constructed network problem is related to the spacings of the points in the RST problem instance we are given. As this spacing grows, the number of nodes in

the network problem may become very large even though the number points in the RST problem may be relatively small. The number of nodes in the L^0 problem thus cannot be polynomially bounded by the number of points in the RST problem. However, the RST problem was shown to be *NP*-complete through a transformation from another *NP*-complete problem called the node-cover problem for planar graphs [*Garey and Johnson,* 1977]. We will not discuss that problem here, but merely note that any instance of it may be posed as an RST problem with a *constant* bound on point spacing. That is, the RST problem is *NP*-complete in the strong sense [*Garey and Johnson,* 1979]. Consequently, any instance of the node cover problem for planar graphs may be solved by an L^0 phase unwrapping algorithm through a polynomial-time transformation. Therefore, the L^0 phase unwrapping problem is *NP*-hard.

Above, we assume that all arcs are uniformly weighted, but we can easily generalize the proof to the weighted case as well since an algorithm solving the weighted L^0 problem can always be assigned uniform weights to solve the unweighted problem.

We also assume above that in obtaining a solution, our hypothetical L^0 algorithm adds only integer numbers of cycles to the wrapped phase. Again, we can generalize the proof by noting that an L^0 optimum will never require the addition of noninteger numbers of cycles. That is, an L^0-optimal unwrapped phase field will always be congruent to its wrapped input. To see this, recall that the network model remains completely general even when noninteger flows are present. Now, suppose there is a noninteger flow on arc a in some feasible L^0 solution. Because flow is conserved and all residues are integers, there must be at least one other arc with a noninteger flow connected to each endpoint of a. Extending this analysis further, a must be part of a closed (but not necessarily directed) loop of noninteger flows as shown in Fig. A.2. We can consequently reduce the total L^0 length of the solution by deleting one arc from the loop and reallocating the flows, which then become integers, on the remaining arcs. The new solution still contains the same residues, so it must still be feasible. Since any solution containing noninteger flows can be improved upon in the L^0 sense, an optimal L^0 solution will never contain noninteger flows. Our hypothetical L^0 algorithm can thus ignore all noncongruent unwrapped solutions.

It should be noted in passing that the Steiner tree problem with a Euclidean distance metric is also *NP*-complete [*Garey et al.,* 1977]. Furthermore, the general network-flow optimization problem with concave cost functions has been shown to be *NP*-hard [*Guisewite and Pardalos,* 1990]. We have shown here that the L^0 problem is *NP*-hard even given its

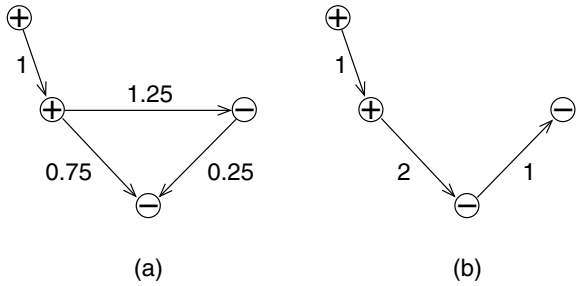

Figure A.2 Part of an L^0 solution, where the numbers indicate flow magnitudes. Noninteger flows must be arranged in closed loops if the net flow out of a node is equal to the node's integer charge (a). The loop can always be broken and the flows reallocated to form integer flows that comprise a better feasible L^0 solution (b).

special network structure and simple cost functions.

Bibliography

[*Ahuja et al.,* 1993] Ahuja, R. K., T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, (Prentice-Hall, Englewood Cliffs, 1993).

[*Bamler et al.,* 1998] Bamler, R., N. Adam, G. W. Davidson, and D. Just, "Noise-Induced Slope Distortion in 2-D Phase Unwrapping by Linear Estimators with Application to SAR Interferometry," *IEEE Transactions on Geoscience and Remote Sensing*, **36**, 913–921 (1998).

[*Bracewell,* 1986] Bracewell, R. N., *The Fourier Transform and Its Applications*, (McGraw-Hill, New York, 1986).

[*Buckland et al.,* 1995] Buckland, J. R., J. M. Huntley, and S. R. E. Turner, "Unwrapping Noisy Phase Maps by Use of a Minimum-Cost-Matching Algorithm," *Applied Optics*, **34**, 5100–5108 (1995).

[*Carballo,* 2000] Carballo, G., *Statistically-Based Multiresolution Network Flow Phase Unwrapping for SAR Interferometry*, Ph. D. dissertation, (Royal Institute of Technology, Stockholm, Sweden, 2000).

[*Carballo and Fieguth,* 2000] Carballo, G. F., and P. W. Fieguth, "Probabilistic Cost Functions for Network Flow Phase Unwrapping," *IEEE Transactions on Geoscience and Remote Sensing*, **38**, 2192–2201 (2000).

[*Chen and Zebker,* 2000] Chen, C. W., and H. A. Zebker, "Network Approaches to Two-Dimensional Phase Unwrapping: Intractability and Two New Algorithms," *Journal of the Optical Society of America A*, **17**, 401–414 (2000).

[*Chen and Zebker,* 2001] Chen, C. W., and H. A. Zebker, "Two-Dimensional Phase Unwrapping with Use of Statistical Models for Cost Functions in Nonlinear Optimization," *Journal of the Optical Society of America A*, **18**, 338–351 (2001).

[*Ching et al.,* 1992] Ching, N. H., D. Rosenfeld, and M. Braun, "Two-Dimensional Phase Unwrapping Using a Minimum Spanning Tree Algorithm," *IEEE Transactions on Image Processing* , **1**, 355–365 (1992).

[*Collaro et al.,* 1998] Collaro, A., G. Franceschetti, F. Palmieri, and M. S. Ferreiro, "Phase Unwrapping by Means of Genetic Algorithms," *Journal of the Optical Society of America A*, **15**, 407–418 (1998).

[*Costantini,* 1998] Costantini, M., "A Novel Phase Unwrapping Method Based on Network Programming," *IEEE Transactions on Geoscience and Remote Sensing*, **36**, 813–821 (1998).

[*Costantini et al.,* 1999] Costantini, M., A. Farina, and F. Zirilli, "A Fast Phase Unwrapping Algorithm for SAR Interferometry," *IEEE Transactions on Geoscience and Remote Sensing*, **37**, 452–460 (1999).

[*Curlander and McDonough,* 1991] Curlander, J. C., and R. N. McDonough, *Synthetic Aperture Radar: Systems and Signal Processing*, (Wiley, New York, 1991).

[*Cusak et al.,* 1995] Cusak, R., J. M. Huntley, and H. T. Goldrein, "Improved Noise-Immune Phase-Unwrapping Algorithm," *Applied Optics*, **34**, 781–789 (1995).

[*Davidson and Bamler,* 1999] Davidson, G. W., and R. Bamler, "Multiresolution Phase Unwrapping for SAR Interferometry," *IEEE Transactions on Geoscience and Remote Sensing*, **37**, 163–174 (1999).

[*Dial,* 1969] Dial, R. B., "Shortest Path Forest with Topological Ordering," *Communications of the ACM*, **12**, 632–633 (1969).

[*Dijkstra,* 1959] Dijkstra, E. W., "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik*, **1**, 269–271 (1959).

[*Elachi,* 1987] Elachi, C., *Introduction to the Physics and Techniques of Remote Sensing*, (Wiley, New York, 1987).

[*Fjørtoft et al.,* 1998] Fjørtoft, R., A. Lopès, P. Marathon, and E. Cubero-Castan, "An Optimal Multiedge Detector for SAR Image Segmentation," *IEEE Transactions on Geoscience and Remote Sensing*, **36**, 793–802 (1998).

[*Flynn,* 1997] Flynn, T. J., "Two-Dimensional Phase Unwrapping with Minimum Weighted Discontinuity," *Journal of the Optical Society of America A*, **14**, 2692–2701 (1997).

[*Fornaro et al.,* 1996] Fornaro, G., G. Franceschetti, R. Lanari, and E. Sansosti, "Robust Phase Unwrapping Techniques: A Comparison," *Journal of the Optical Society of America A*, **13**, 2355–2366 (1996).

[*Fornaro and Sansosti,* 1999] Fornaro, G., and E. Sansosti, "A Two-Dimensional Region Growing Least Squares Phase Unwrapping Algorithm for Interferometric SAR Processing," *IEEE Transactions on Geoscience and Remote Sensing*, **37**, 2215–2226 (1999).

[*Fortune,* 1987] Fortune, S., "A Sweepline Algorithm for Voronoi Diagrams," *Algorithmica*, **2**, 153–174 (1987).

[*Fried,* 1977] Fried, D. L., "Least-Square Fitting a Wave-Front Distortion Estimate to an Array of Phase-Difference Measurements," *Journal of the Optical Society of America*, **67**, 370–375 (1977).

[*Gabriel et al.,* 1989] Gabriel, A. K., R. M. Goldstein, and H. A. Zebker, "Mapping Small Elevation Changes over Large Areas: Differential Radar Interferometry," *Journal of Geophysical Research*, **94**, 9183–9191 (1989).

[*Garey et al.,* 1977] Garey, M. R., R. L. Graham, and D. S. Johnson, "The Complexity of Computing Steiner Minimal Trees," *SIAM Journal on Applied Mathematics*, **32**, 835–859 (1977).

[*Garey and Johnson,* 1979] Garey, M. R., and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, (Freeman, San Francisco, 1979).

[*Garey and Johnson,* 1977] Garey, M. R., and D. S. Johnson, "The Rectilinear Steiner Tree Problem is *NP*-complete," *SIAM Journal on Applied Mathematics*, **32**, 826–834 (1977).

[*Gatelli et al.,* 1994] Gatelli, F., A. M. Guarnieri, F. Parizzi, P. Pasquali, C. Prati, and F. Rocca, "The Wavenumber Shift in SAR Interferometry," *IEEE Transactions on Geoscience and Remote Sensing*, **32**, 855–865 (1994).

[*Ghiglia and Pritt,* 1998] Ghiglia, D. C., and M. D. Pritt, *Two-Dimensional Phase Unwrapping: Theory, Algorithms, and Software* (Wiley, New York, 1998).

[*Ghiglia and Romero,* 1996] Ghiglia, D. C., and L. A. Romero, "Minimum L^p-norm Two-Dimensional Phase Unwrapping," *Journal of the Optical Society of America A*, **13**, 1999–2013 (1996).

[*Ghiglia and Romero,* 1994] Ghiglia, D. C., and L. A. Romero, "Robust Two-Dimensional Weighted and Unweighted Phase Unwrapping that Uses Fast Transforms and Iterative Methods," *Journal of the Optical Society of America A*, **11**, 107–117 (1994).

[*Glover and Schneider,* 1991] Glover, G. H., and E. Schneider, "Three-Point Dixon Technique for True Water/Fat Decomposition with B_0 Inhomogeneity Correction," *Magnetic Resonance in Medicine*, **18**, 371–383 (1991).

[*Goering et al.,* 1995] Goering, D. J., H. Chen, L. D. Hinzman, and D. L. Kane, "Removal of Terrain Effects from SAR Satellite Imagery of Arctic Tundra," *IEEE Transactions on Geoscience and Remote Sensing*, **33**, 185–194 (1995).

[*Goldstein et al.,* 1993] Goldstein, R. M., H. Englehardt, B. Kamb, and R. M. Frolich, "Satellite Radar Interferometry for Monitoring Ice Sheet Motion: Application to an Antarctic Ice Stream," *Science*, **262**, 1525–1530 (1993).

[*Goldstein and Werner,* 1998] Goldstein, R. M., and C. L. Werner, "Radar Interferogram Phase Filtering for Geophysical Applications," *Geophysical Research Letters*, **25**, 4035–4038 (1998).

[*Goldstein and Zebker,* 1987] Goldstein, R M., and H. A. Zebker, "Interferometric Radar Measurements of Ocean Surface Currents," *Nature* (London), **328**, 707–709 (1987).

[*Goldstein et al.,* 1988] Goldstein, R. M., H. A. Zebker, and C. L. Werner, "Satellite Radar Interferometry: Two-Dimensional Phase Unwrapping," *Radio Science*, **23**, 713–720 (1988).

[*Guibas and Stolfi,* 1985] Guibas, L., and J. Stolfi, "Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi Diagrams," *ACM Transactions on Graphics*, **4**, 74–123 (1985).

[*Guindon,* 1990] Guindon, B., "Development of a Shape-from-Shading Technique for the Extraction of Topographic Models from Individual Spaceborne SAR Images," *IEEE Transactions on Geoscience and Remote Sensing*, **28**, 654–661 (1990).

[*Guisewite and Pardalos,* 1990] Guisewite, G. M., and P. M. Pardalos, "Minimum Concave-Cost Network Flow Problems: Applications, Complexity and Algorithms," *Annals of Operations Research*, **25**, 75–100 (1990).

[*Hanan,* 1966] Hanan, M., "On Steiner's Problem with Rectilinear Distance," *Journal SIAM Applied Mathematics*, **2**, 255–265 (1966).

[*Hedley and Rosenfeld,* 1992] Hedley, M., and D. Rosenfeld, "A New Two-Dimensional Phase Uwnrapping Algorithm for MRI Images," *Magnetic Resonance in Medicine*, **24**, 177–181 (1992).

[*Herment et al.,* 2000] Herment, A., E. Mousseaux, O. Jolivet, A. DeCesare, F. Frouin, A. Todd-Pokropek, and J. Bittoun, "Improved Estimation of Velocity and Flow Rate Using Regularized Three-Point Phase-Contrast Velocimetry," *Magnetic Resonance in Medicine*, **44**, 122–128 (2000).

[*Hudgin,* 1977] Hudgin, R. H., "Wave-Front Reconstruction for Compensated Imaging," *Journal of the Optical Society of America*, **67**, 375–378 (1977).

[*Hunt,* 1979] Hunt, B. R., "Matrix Formulation of the Reconstruction of Phase Values from Phase Differences," *Journal of the Optical Society of America*, **69**, 393–399 (1979).

[*Hwang,* 1976] Hwang, F. K., "On Steiner Minimal Trees with Rectilinear Distance," *SIAM Journal on Applied Mathematics*, **30**, 104–114 (1976).

[*Just and Bamler,* 1994] Just, D., and R. Bamler, "Phase Statistics of Interferograms with Applications to Synthetic Aperture Radar," *Applied Optics*, **33**, 4361–4368 (1994).

[*Kennington and Helgason,* 1980] Kennington, J. L., and R. V. Helgason, *Algorithms for Network Programming*, (Wiley, New York, 1980).

[*Lee et al.,* 1994] Lee, J. S., K. W. Hoppel, S. A. Mango, A. R. Miller, "Intensity and Phase Statistics of Multilook Polarimetric and Interferometric SAR Imagery," *IEEE Transactions on Geoscience and Remote Sensing*, **32**, 1017–1028 (1994).

[*Li and Goldstein,* 1990] Li, F. K., and R. M. Goldstein, "Studies of Multibaseline Spaceborne Interferometric Synthetic Aperture Radars," *IEEE Transactions on Geoscience and Remote Sensing*, **28**, 88–97 (1990).

[*Lopes et al.,* 1993] Lopes, A., E. Nezry, R. Touzi, and H. Laur, "Structure Detection and Statistical Adaptive Speckle Filtering in SAR Images," *International Journal of Remote Sensing*, **14**, 1735–1758 (1993).

[*Lyuboshenko and Maître,* 1999] Lyuboshenko, I., and H. Maître, "Phase Unwrapping for Interferometric Synthetic Aperture Radar by Use of Helmholtz Equation Eigenfunctions and the First Green's Identity," *Journal of the Optical Society of America A*, **16**, 378–395 (1999).

[*Madsen,* personal communication, 2001] Madsen, S. N, personal communication (2001).

[*Massonnet et al.,* 1993] Massonnet, D., M. Rossi, C. Carmona, F. Adragna, G. Peltzer, K. Feigl, and T. Rabaute, "The Displacement Field of the Landers Earthquake Mapped by Radar Interferometry," *Nature* (London), **364**, 138–142 (1993).

[*Nash and Sofer,* 1996] Nash, S. G., and A. Sofer, *Linear and Nonlinear Programming*, (McGraw-Hill, New York, 1996).

[*Oliver and Quegan,* 1998] Oliver, C. J., and S. Quegan, *Understanding Synthetic Aperture Radar Images*, (Artech House, Boston, 1998).

[*Pallottino,* 1984] Pallottino, S., "Shortest-Path Methods: Complexity, Interrelations and New Propositions," *Networks*, **14**, 257–267 (1984).

[*Prim,* 1957] Prim, R. C., "Shortest Connection Networks and Some Generalizations," *The Bell Systems Technical Journal*, **36**, 1389–1401 (1957).

[*Pritt,* 1996] Pritt, M. D., "Phase Unwrapping by Means of Multigrid Techniques for Interferometric SAR," *IEEE Transactions on Geoscience and Remote Sensing*, **34**, 728–738 (1996).

[*Rodriguez and Martin,* 1992] Rodriguez, E., and J. M. Martin, "Theory and Design of Interferometric Synthetic Aperture Radars," *IEE Proceedings-F*, **139**, 147–159 (1992).

[*Skolnik,* 1980] Skolnik, M. I., *Introduction to Radar Systems*, (McGraw-Hill, New York, 1980).

[*Thomas and Finney,* 1988] Thomas, G. B., and R. L. Finney, *Calculus and Analytic Geometry*, (Addison-Wesley, Reading, 1988).

[*Touzi et al.,* 1999] Touzi, R., A. Lopes, J. Bruniquel, and P. W. Vachon, "Coherence Estimation for SAR Imagery," *IEEE Transactions on Geoscience and Remote Sensing*, **37**, 135–149 (1999).

[*Ulaby et al.,* 1981] Ulaby, F. T., R. K. Moore, and A. K. Fung, *Microwave Remote Sensing, Active and Passive*, (Addison-Wesley, London, 1981).

[*Ulander,* 1996] Ulander, L. M. H., "Radiometric Slope Correction of Synthetic Aperture Radar Images," *IEEE Transactions on Geoscience and Remote Sensing*, **34**, 1115–1122 (1996).

[*Xu and Cumming,* 1999] Xu, W., and I. Cumming, "A Region-Growing Algorithm for InSAR Phase Unwrapping," *IEEE Transactions on Geoscience and Remote Sensing*, **37**, 124–134 (1999).

[*Zebker et al.,* 1994a] Zebker, H. A., T. G. Farr, R. P. Salazar, and T. H. Dixon, "Mapping the World's Topography Using Radar Interferometry: The TOPSAT Mission," *Proceedings of the IEEE*, **82**, 1774–1786 (1994).

[*Zebker et al.,* 1994b] Zebker, H. A., P. A. Rosen, R. M. Goldstein, A. Gabriel, and C. Werner, "On the Derivation of Coseismic Displacement Fields Using Differential Radar Interferometry: The Landers Earthquake," *Journal of Geophysical Research*, **99**, 19617–19634 (1994).

[*Zebker and Goldstein,* 1986] Zebker, H., and R. Goldstein, "Topographic Mapping from Interferometric SAR Observations," *Journal of Geophysical Research*, **91**, 4993–4999 (1986).

[*Zebker and Lu,* 1998] Zebker, H. A., and Y. Lu, "Phase Unwrapping Algorithms for Radar Interferometry: Residue-Cut, Least-Squares, and Synthesis Algorithms," *Journal of the Optical Society of America A*, **15**, 586–598 (1998).

[*Zebker et al.,* 1997] Zebker, H. A., P. A. Rosen, and S. Hensley, "Atmospheric Effects in Interferometric Synthetic Aperture Radar Surface Deformation and Topography Maps," *Journal of Geophysical Research*, **102**, 7547–7563 (1997).

[*Zebker and Villasenor,* 1992] Zebker, H. A., and J. Villasenor, "Decorrelation in Interferometric Radar Echoes," *IEEE Transactions on Geoscience and Remote Sensing*, **30**, 950–959 (1992).