# Windowed Factorized Backprojection for Pulsed and LFM-CW Stripmap SAR

Kyra M. Moon

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

David G. Long, Chair
Richard W. Christiansen
Dah Jye Lee

Department of Electrical Engineering

Brigham Young University

August 2012

# ABSTRACT

## Windowed Factorized Backprojection for Pulsed and LFM-CW Stripmap SAR

Kyra M. Moon
Department of Electrical Engineering
Master of Science

Factorized backprojection is a processing algorithm for reconstructing images from data collected by synthetic aperture radar (SAR) systems. Factorized backprojection requires less computation than conventional time-domain backprojection with little loss in accuracy for straight-line motion. However, its implementation is not as straightforward as direct backprojection. Further, implementing an azimuth window has been difficult in previous versions of factorized backprojection. This thesis provides a new, easily parallelizable formulation of factorized backprojection designed for both pulsed and linearly frequency modulated continuous wave (LFM-CW) stripmap SAR data. A method of easily implementing an azimuth window as part of the factorized backprojection algorithm is introduced. The approximations made in factorized backprojection are investigated and a detailed analysis of the corresponding errors is provided. We compare the performance of windowed factorized backprojection to direct backprojection for simulated and actual SAR data.

Keywords: synthetic aperture radar (SAR), image reconstruction, factorized backprojection, backprojection

# ACKNOWLEDGMENTS

Dr. David Long, thank you for your guidance and direction throughout my years of research at BYU. Thank you for offering me flexibility and freedom in my research and for seeing potential in me. I have learned so much by working with you.

Thank you to the Rocky Mountain NASA Space Grant Consortium and the Space Dynamics Laboratory for financially supporting me in my research.

To my wonderful husband Kevin, thank you for supporting and encouraging me in my research. I have appreciated being able to bounce ideas off of you. Thank you for being there for both my frustrations and successes.

# Table of Contents

# List of Tables

## List of Figures

# Chapter 1

# Introduction

Synthetic aperture radar (SAR) can generate high-resolution images using a short antenna and a large bandwidth [1] [2]. SAR creates images by transmitting and receiving electromagnetic waves and differentiating objects based on the radar echoes returned from them. Images can be created day or night and in inclement weather since radar does not depend on light to create images.

A common method for collecting data with SAR is to attach a short antenna to an aircraft. This antenna sends out electromagnetic pulses as the aircraft moves, enabling synthesis of a long linear array. Since a longer antenna provides finer resolution than a short antenna, this linear array provides finer resolution than a single antenna position. If the antenna is kept orthogonal to the motion of the aircraft for the duration of the flight, then the SAR operating mode is denoted as *stripmap*.

Several algorithms have been proposed for stripmap image reconstruction of SAR data in both the time domain and frequency domain [3]. A particular time-domain algorithm known as *backprojection* is able to reconstruct well-focused images, even with non-ideal motion such as when the aircraft does not fly on a straight track. Unfortunately, the computational complexity of backprojection is $O(N^3)$, where $N$ is the number of pixels in a column of the imaging grid and is typically in the hundreds or thousands. This can quickly become computationally expensive.

Because of this computational cost, *factorized backprojection* was developed. This algorithm divides the process of backprojection into recursive steps to achieve complexity of $O(N^2 \log N)$. Multiple variations on factorized backprojection have been developed [1] [4] [5] [6] [7] [8]. Although these algorithms successfully generate images from SAR data, none include an implementation of an azimuth window. Such a window can reduce sidelobes and aliasing at a tradeoff of some loss of resolution. Additionally, these algorithms are not easily parallelizable.

This thesis presents a new, easily parallelizable formulation of factorized backprojection designed for both pulsed and LFM-CW stripmap SAR which facilitates addition of an azimuth window. I first demonstrate how to implement the algorithm without the azimuth window for pulsed and LFM-CW SAR data. I then demonstrate how to implement an azimuth window as part of the factorized backprojection algorithm. The performance of the algorithm is analyzed, including an error analysis and evaluation of the computational requirements. I compare performance of the windowed factorized backprojection algorithms with factorized and conventional time-domain backprojection.

## 1.1 Contributions and Outline

The primary contribution of this thesis is the factorized backprojection algorithm I have developed. Like other factorized backprojection algorithms, this algorithm has computational complexity of $O(N^2 \log N)$, an $N/\log N$ improvement over conventional backprojection. However, this new algorithm has the advantage of performing the factorization in columns parallel to the flight track. Since each column can be reconstructed independent of the others, this allows for easy parallelization of the algorithm and allows for the easy implementation of an azimuth window. The algorithm is valid for both pulsed and LFM-CW SAR with only slight differences between the two.

Additionally, this thesis provides a performance analysis. As with other factorized backprojection algorithms, this algorithm has a direct tradeoff between computational complexity and accuracy. The thesis provides an error analysis which allows for the bounds on the overall error to be predicted prior to implementing the algorithm.

This thesis is organized as follows. In Chapter 2, the background necessary for understanding the remainder of the thesis is given. This includes the basics of synthetic aperture radar operation for pulsed and LFM-CW operating modes, an explanation of various SAR image reconstruction algorithms, and an overview of past factorized backprojection algorithms.

Chapter 3 describes the basics of the factorized backprojection algorithm for both pulsed and LFM-CW SAR. It is shown how to incorporate an azimuth window into the factorized backprojection algorithm. The computational and memory requirements are discussed.

In Chapter 4, the performance of factorized backprojection is analyzed. The errors inherent to factorized backprojection are discussed. Because pulsed SAR and LFM-CW SAR have slightly different formulations, the errors in each are discussed separately. A brief discussion of errors caused by range migration is included. Plots demonstrating the efficacy of factorized backprojection are shown for simulated and real data.

In Chapter 5, some miscellaneous results pertaining to factorized backprojection are presented which provide further insight into the algorithm. First, an alternate time-domain image reconstruction algorithm is discussed which uses several similar principles as factorized backprojection. Then, a matrix formulation of factorized backprojection is introduced.

In Chapter 6, conclusions are made and possibilities for future work are discussed. In Appendix A, pseudo-code is provided. In Appendix B, I provide tables of parameters for the various data sets used to demonstrate the performance of factorized backprojection.

# Chapter 2

# Background

This chapter presents the fundamentals of stripmap synthetic aperture radar processing and provides an introduction to time-domain SAR image reconstruction. An overview is given of two SAR modes, namely pulsed and continuous wave SAR. Time-domain image reconstruction is then discussed. Finally, an overview of past factorized backprojection algorithms is presented.

## 2.1 Introduction to SAR

Synthetic aperture radar, or SAR, is an active microwave remote sensing technique that produces high-resolution images of wide areas [9]. This is accomplished by attaching an antenna to a platform and sending out a series of electromagnetic pulses as the platform moves. The series of radar echoes from the target area are then received and synthesized to form the image.

SAR imagery differs from optical imagery in a number of ways [9]. Unlike optical imagery, SAR imagery can be collected day or night since it uses its own electromagnetic waves as illumination to form images rather than reflected light from the sun as in photography. Furthermore, SAR data can be collected even in inclement weather since electromagnetic waves can penetrate rain, snow, or clouds with some attenuation. Additionally, SAR imagery is based on the backscattered reflectivity. Hence, SAR highlights objects which reflect energy back to the radar antenna such as buildings and vehicles, whereas optical imagery is brightest for objects which are bright in the visible spectrum. Another difference is that SAR resolution is independent of the height of the radar and only depends on the length of the antenna. However, unlike optical imagery which can be instantaneous, SAR systems require platform motion and thence time to collect enough data to achieve fine resolution. Thus, SAR imagery and optical imagery are appropriate for different situations.

## 2.2 Basics of SAR

The basic premise of SAR is that a short antenna provides coarse resolution while a long antenna provides fine resolution. By moving a short antenna along a linear series of positions, a longer antenna can be synthesized (hence the name synthetic aperture radar).

There are several different SAR imaging modes. The mode discussed in this thesis is stripmap SAR, which occurs when a fixed antenna points orthogonally to the motion of the platform [3]. Other modes of operation include ScanSAR, Spotlight SAR, and Circular SAR [2] [9].

### 2.2.1 General SAR Model

Figure 2.1 shows a simple geometric model of a stripmap SAR system. The terms used to describe the SAR geometry are defined as follows.

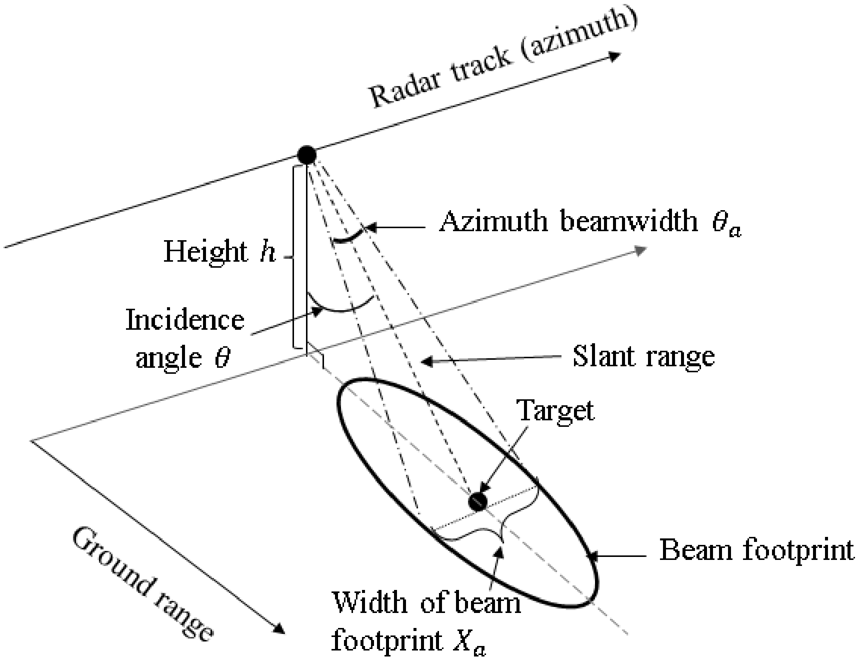


**Figure 2.1:** Radar data acquisition geometry. Adapted from [3]

**Point Target:** This is a hypothetical point on the surface that the SAR system is imaging. Although the SAR system actually images an area, to develop the SAR equations, a single point target on the ground is considered.

**Incidence Angle:** The incidence angle $\theta$ is the angle between the antenna pointing straight down and the angle at which a pulse is transmitted.

**Azimuth Beamwidth:** The *half-power beamwidth*, or simply *beamwidth* $\theta_a$, is the range of angles for which the radiation strength is within 3 dB of the maximum strength. In azimuth, the beamwidth is approximately the wavelength divided by the antenna length in this direction.

**Beam Footprint:** During the transmission of a particular pulse, the radar antenna projects a beam onto an area called the *beam footprint*. The width of the beam footprint in the azimuth direction is $X_a = h\theta_a / \cos\theta$, where $h$ is the height of the radar platform [2].

**Range:** The ground range direction, commonly referred to as *range*, lies perpendicular to the radar track.

**Azimuth:** The azimuth direction lies parallel to the radar track.

**Slant Range:** The slant range is the distance from a given antenna position (pulse) to a given target on the ground.

### 2.2.2 SAR Resolution

#### Range Resolution

The range resolution corresponds to the minimum distance between two points along the same range line which can be distinguished from each other. If two points are separated by a distance $X_r$, then their respective echoes are separated by a time of flight difference $\Delta t$ where [2]

$$\Delta t = \frac{2X_r}{c_0} \sin\theta \tag{2.1}$$

where $\theta$ is the incidence angle and $c_0$ is the speed of light. As discussed in [2], the smallest discriminable time difference is $1/B$ where $B$ is the bandwidth. Hence, the range resolution is given by

$$\begin{aligned} \frac{2X_r}{c_0}\sin\theta &= 1/B \\ \Rightarrow X_r &= \frac{c_0}{2B}\sin\theta. \end{aligned} \tag{2.2}$$

### Azimuth Resolution

The azimuth resolution corresponds to the minimum distance between two points along the same azimuth line which can be distinguished from each other. For a real aperture (single antenna), this resolution is equal to the width of the antenna footprint because all the points along a line spanning that width are returned at the same time because they are equidistant from the antenna [2]. Thus the azimuth resolution of a real aperture is equal to

$$X_a = h\theta_a/\cos\theta = h\lambda/L\cos\theta \tag{2.3}$$

where $\theta_a$ is the antenna beamwidth in azimuth, $h$ is the height of the aircraft above the imaging surface, $L$ is the length of the antenna, and $\lambda$ is the wavelength of the transmit frequency.

Recall that the length of an antenna is inversely proportional to its beamwidth. Thus, using a longer antenna decreases the beamwidth, which decreases the width of the footprint and thus improves the resolution. However, it is frequently costly or impractical to construct a long antenna. An alternative is to synthesize a long antenna by moving an antenna with length $L$ along a linear array of positions [2]. The width of the antenna footprint for the single antenna is

$$\mathscr{L} = \frac{h\lambda}{L\cos\theta} = \frac{2\lambda h}{L}. \tag{2.4}$$

If the antenna is moved along an array of length $\mathscr{L}$, then the synthesized array has a beamwidth $\theta_s$ of

$$\theta_s \approx \lambda/\mathscr{L} \approx L/2h \tag{2.5}$$

and the resulting footprint has width

$$X_a = h\theta_s = L/2. \tag{2.6}$$

Thus, the azimuth resolution of the synthesized antenna depends only upon the length of the physical antenna, not on the height of the aircraft or the wavelength. This allows for very fine azimuth resolution that is independent of range, which is one of the chief advantages of SAR.

## 2.3 Signal Transmission

A frequently used signal sent out by the radar is called a chirp, which can be expressed as

$$s(t) = A(t)\exp\left(j(2\pi f_0 t + \pi k_r t^2 + \phi)\right) \tag{2.7}$$

where $A(t)$ is the signal amplitude, $f_0$ is the initial frequency of the chirp, $k_r$ is the chirp rate, and $\phi$ is the starting phase, which is usually neglected [2] [3] [10]. Note that the chirp is linearly frequency modulated, or LFM. This signal propagates to the target area, and some of it is reflected back to the radar. The signal received by the radar can be expressed as

$$s_r(t) = A'(t)\exp\left(j(2\pi f_0(t-\tau) + \pi k_r (t-\tau)^2 + \phi)\right) \tag{2.8}$$

where $A'(t)$ is an attenuated version of $A(t)$ and $\tau$ is the two-way time of flight to the target at range $R$ [11],

$$\tau = \frac{2R}{c_0}. \tag{2.9}$$

Two modes of SAR which use this type of signal are pulsed SAR and LFM-CW SAR, discussed in Section 2.3.1.

### 2.3.1 Schemes

This thesis discusses two types of SAR systems, namely pulsed SAR and LFM-CW SAR. These are delineated primarily by the length of the chirp.

If the chirp is short enough that a chirp can be transmitted and received before the next chirp is sent out, then the operating scheme is called *pulsed SAR*. This allows for a monostatic system since the same antenna can be used for both transmit and receive. It also permits the start-stop assumption, which is that the antenna is stationary between transmit and receive [3].

If the chirp is long enough that the echo is received before it is entirely transmitted, then the operating scheme is called *linearly frequency modulated-continuous wave*, or LFM-CW. This scheme requires two separate antennas for transmit and receive but maximizes the pulse length, thus requiring less power to achieve the same effective SNR. This allows for smaller and thus cheaper SAR systems [12].

A complication for LFM-CW signals comes due to the movement of the platform between transmit and receive. Due to the constant motion of the platform, $\tau$, the two-way travel time to a target, can be difficult to calculate because the signal travels a different distance to and from the target. However, for most applications this difference in distances can be neglected [12].

## 2.4 SAR Image Formation

In modern digital SAR processing, there are two primary steps to reconstruct a well-focused image from the received signal: range compression and azimuth compression [12]. These are discussed in the following sections.

### 2.4.1 Range Compression

Once the radar echoes have been received, the data must be range compressed. Range compression is performed by matched filtering the received signal with the expected signal, a time-delayed copy of the chirp that was transmitted [12]. This is typically accomplished via an FFT, complex multiply, and IFFT.

### Range Compression for Pulsed SAR

For pulsed SAR, recall that the received signal is of the form

$$s_r(t) = A'(t)\exp(j(2\pi f_0(t-\tau) + \pi k_r(t-\tau)^2 + \phi)). \tag{2.10}$$

The Fourier transform of $s_r(t)$ can be expressed as [12]

$$\begin{aligned} S_r(f_r) &= \int_{\tau}^{\tau+t_p} s_r(t)\cdot\exp(-j2\pi f_r t)dt \\ &= B(f_r)\cdot\exp\left(-j\frac{2\pi\tau k_r f_0 + 2\pi\tau k_r f_r + \pi f_r^2}{k_r}\right) \end{aligned} \tag{2.11}$$

where $t_p$ is the pulse length and $B(f_r)$ is a complex function which can be approximated by a rect function.

The Fourier transform of the time-reversed conjugate of the chirp is

$$\begin{aligned} H_{rc}(f_r) &= \int_{0}^{t_p} \exp(-j\pi k_r(-t)^2)\cdot\exp(-j2\pi f_r t) \\ &= B'(f_r)\cdot\exp\left(j\frac{\pi f_r^2}{k_r}\right) \end{aligned} \tag{2.12}$$

where $B'(f_r)$ is approximately the same as $B(f_r)$.

$S_r$ and $H_{rc}$ are multiplied together to yield

$$S_h(f_r) = S_r(f_r)\cdot H_{rc}(f_r) = B(f_r)B'(f_r)\cdot\exp(-j2\pi\tau(f_0+f_r)) \tag{2.13}$$

and the inverse Fourier transform is

$$\begin{aligned} s_{rc}(t) &= \int_{0}^{t_p\cdot k_r} S_h(f_r)\cdot\exp(j2\pi f_r t)df_r \\ &= \frac{\sin(t_p\pi k_r(t-\tau))\exp(j\pi k_r t_p(t-\tau) - j2\pi f_0\tau)}{\pi k_r(t-\tau)} \\ &= t_p\text{sinc}(t_p\pi k_r(t-\tau))\cdot\exp(j\pi k_r t_p(t-\tau) - j2\pi f_0\tau) \\ &\approx t_p\cdot\text{sinc}(t_p\pi k_r(t-\tau))\cdot\exp(-j2\pi f_0\tau). \end{aligned} \tag{2.14}$$

The peak of the sinc function occurs at $t = \tau = 2R/c_0$, giving

$$s_{rc}(t = 2R/c_0) = \exp\left\{-j\frac{4\pi R}{\lambda}\right\}. \tag{2.15}$$

The range-compressed signal $s_{rc}$ must then be azimuth compressed.

### Range Compression for LFM-CW SAR

Typically, a small LFM-CW SAR may be limited to a small sampling bandwidth due to hardware constraints [12]. This small bandwidth may limit the imaging swath. To overcome this limitation, a delayed dechirp may be used to shift more ranges into the sampling bandwidth. This allows the bandwidth to be maximized since no bandwidth is wasted on any range nearer to the platform than a set minimum range.

However, the delayed dechirp requires additional computations prior to range compression. Recall that the received signal is of the form

$$s_r(t) = \exp\left[j(\phi + 2\pi f_0(t-\tau) + \pi k_r(t-\tau)^2)\right]. \tag{2.16}$$

The received signal is then mixed with a copy of the transmit signal delayed by **d**

$$s_{td}(t) = \exp\left[j(\phi + 2\pi f_0(t-\mathbf{d}) + \pi k_r(t-\mathbf{d})^2)\right]. \tag{2.17}$$

The dechirped signal is given by multiplying Eq. (2.16) by the complex conjugate of Eq. (2.17), resulting in [11]

$$s_{dc}(t) = \exp\left\{j\left[2\pi k_r t(\tau-\mathbf{d}) + 2\pi f_0(\tau-\mathbf{d}) - \pi k_r(\tau^2-\mathbf{d}^2)\right]\right\}. \tag{2.18}$$

As before, the range Fourier transform of the signal is calculated, yielding

$$\begin{aligned} S_R(f_r) &= \int_{\tau}^{\tau+t_p} s_{dc}(t)\exp(-j2\pi f_r t)dt \\ &= t_p \mathrm{sinc}[t_p(f_r + k_r\mathbf{d} - k_r\tau)]\exp(j\pi\Phi_{rc}) \end{aligned} \tag{2.19}$$

where $t_p$ is the pulse length and

$$\Phi_{rc} = k_r t_p(\tau-\mathbf{d}) - f_r t_p + \frac{4k_r R(\tau-\mathbf{d})}{c_0} + 2f_0(\tau-\mathbf{d}) - k_r(\tau^2-\mathbf{d}^2) - \frac{4f_r R}{c_0}. \tag{2.20}$$

$S_R$ is the range-compressed signal which must then be azimuth compressed.

### 2.4.2 Azimuth Compression

Once the data has been range compressed, it must be compressed in the azimuth direction to be fully focused. Azimuth compression is necessary because the data has been collected over a series of short antenna positions which must be consolidated into one long synthetic antenna. This can be accomplished via a matched filter, i.e., multiplying the range-compressed signal by the conjugate of the expected phase in the azimuth direction.

#### Frequency-Domain Azimuth Compression

The traditional approach to azimuth compression is called the Range-Doppler Algorithm, or RDA [3]. This algorithm performs azimuth compression in the frequency domain. Basic RDA consists of an azimuth FFT, range-cell migration correction, azimuth matched filtering, and an azimuth IFFT.

Recall that the range-compressed signal from a pulsed SAR has the form

$$s_{rc}(t) = t_p \cdot \mathrm{sinc}\left[t_p \pi k_r (t - 2R/c_0)\right] \cdot \exp\left(-j4\pi R/\lambda\right). \tag{2.21}$$

The range $R$ can be considered a function of $\eta$, where $\eta$ is the slow-time (corresponding to pulse position). Then

$$R(\eta) = \sqrt{R_0^2 + V_r^2 \eta^2} \approx R_0 + \frac{V_r^2 \eta^2}{2R_0} \tag{2.22}$$

where $R_0$ is the range of closest approach (the minimum distance from a target to the flight track) and $V_r$ is the velocity of the flight track. The range-compressed signal can be rewritten as

$$s_{rc}(t,\eta) \approx t_p \cdot \mathrm{sinc}(t_p \pi k_r (t - 2R(\eta)/c_0)) w_a(\eta - \eta_c) \cdot \exp(-j4\pi/\lambda R_0) \exp(-j2\pi/\lambda V_r^2 \eta^2) \tag{2.23}$$

where $w_a(\eta - \eta_c)$ is an azimuth envelope similar to a sinc-squared function and $\eta_c$ is the Doppler centroid frequency [3]. Since the phase is a function of $\eta^2$, the signal has linear FM characteristics, with the linear FM rate being

$$K_a = \frac{2V_r^2}{\lambda R_0}. \tag{2.24}$$

An azimuth FFT is then performed to transform the data into the range Doppler domain. The relationship between azimuth frequency and time is

$$f_\eta = -K_a \eta. \tag{2.25}$$

Thus, the azimuth FFT can be expressed as

$$\begin{aligned} S_1(t, f_\eta) &= \mathrm{FFT}\left\{s_{rc}(\tau, \eta)\right\} \\ &= \mathrm{rect}\left[t - \frac{2R_{rd}(f_\eta)}{c_0}\right] W_a(f_\eta - f_{\eta_c}) \exp\left(-j\frac{4\pi R_0}{\lambda}\right) \exp\left(j\pi\frac{f_\eta^2}{K_a}\right) \end{aligned} \tag{2.26}$$

where

$$R_{rd}(f_\eta) \approx R_0 + \frac{V_r^2}{2R_0}\left(\frac{f_\eta}{K_a}\right)^2 = R_0 + \frac{\lambda^2 R_0 f_\eta^2}{8V_r^2}. \tag{2.27}$$

The next step in RDA is range migration correction. *Range migration* is caused when all the returns from a target are not contained within the same range bin (range resolution element). If an antenna has a narrow beamwidth, then all of the pulses which contain the target in their footprint are roughly the same distance from the target, so little to no range migration occurs. However, an antenna with a wider beamwidth is more likely to cause range migration because a greater range of pulses include the target in their beamwidth.

RDA operates under the assumption that a target stays in the same range bin across the length of the synthetic aperture. However, if the antenna has a wide enough beamwidth, then this assumption is no longer valid. In order to correct for range migration, all of the radar returns corresponding to a given target must be assigned to the same range bin, namely the range bin corresponding to $R_0$ for the target. This can be accomplished by performing a range interpolation operation in the range Doppler domain [3]. The amount of range migration to correct is given by

$$\Delta R(f_\eta) = \frac{\lambda^2 R_0 f_\eta^2}{8V_r^2}. \tag{2.28}$$

The range-migration-corrected signal is

$$S_2(t, f_\eta) = \text{rect}\left(t - \frac{2R_0}{c_0}\right) W_a(f_\eta - f_{\eta_c}) \exp\left(-j\frac{4\pi R_0}{\lambda}\right) \exp\left(j\pi\frac{f_\eta^2}{K_a}\right). \tag{2.29}$$

Following range-cell migration correction, a matched filter is applied to focus the data in the azimuth direction. Since the data is already in the range Doppler domain, it is convenient to perform the matched filtering in this domain as a function of slant range and azimuth frequency. The matched filter $H_{az}$ is the complex conjugate of $\exp(j\pi f_\eta^2/K_a)$,

$$H_{az}(f_\eta) = \exp\left(-j\pi\frac{f_\eta^2}{K_a}\right). \tag{2.30}$$

$S_2$ is multiplied by $H_{az}$ to yield

$$S(t, f_\eta) = \text{rect}(t - 2R_0/c_0) W(f_\eta - f_{\eta_c}) \exp\left(-j\frac{4\pi R_0}{\lambda}\right). \tag{2.31}$$

Finally, the range Doppler domain data $S(t, f_\eta)$ is transformed back to the time domain via an azimuth IFFT. This yields

$$\begin{aligned} s_{\text{ac}}(t,\eta) &= \text{IFFT}\left\{S(t, f_\eta)\right\} \\ &= \text{sinc}\left[t_p \pi k_r (t-\tau)\right] \cdot p_a(\eta) \exp\left(-j\frac{4\pi f_0 R}{c_0}\right) \exp\left(j2\pi f_{\eta_c}\eta\right) \end{aligned} \tag{2.32}$$

where $p_a$ is the amplitude of the azimuth impulse response (a sinc-like function) [3] . The range and azimuth envelopes indicate that the target is positioned at $t = 2R/c$ and $\eta = 0$. This process is repeated for each range bin.

Although RDA is efficient and fairly accurate, it can require a large computational load to accurately correct for range migration. Other frequency-domain algorithms have been developed to handle the range migration such as the chirp-scaling algorithm and the $\omega - K$ algorithm [3] [10].

### Pulsed SAR Backprojection

To eliminate the complications caused by range cell migration, azimuth compression can be done in the time domain. Time-domain image reconstruction can be exact, handle range migration, and handle a variety of flight tracks [13]. The disadvantage of the time-domain technique is the high computational requirement.

A common time-domain matched filtering technique is called backprojection. This process coherently integrates the radar data over each antenna position to form the image. Given a pixel at location $p$, the backprojected image $A(p)$ is given by [5] [14]

$$A(p) = \int_{-\infty}^{\infty} R(d(\eta, p)) \exp(j4\pi d(\eta, p)/\lambda) d\eta \tag{2.33}$$

where $A(p)$ is the complex pixel value, $\lambda$ is the wavelength of the transmit frequency, $d(\eta, p)$ is the distance between the pixel $p$ and the along-track position at slow-time index $\eta$, and $R(d(\eta, p))$ is the baseband range-compressed echo data interpolated to the distance $d(\eta, p)$. Backprojection normally operates on digitized echo data. We obtain such data by first digitizing the range-compressed data $s(t, \eta)$ in discrete pulses where each pulse number $n$ corresponds to the along-track position at time $\eta$, the slow-time index. We then interpolate this digitized signal to provide an indexed value for each range $d[n, p]$ from a pulse $n$ to the center some pixel $p$ on the image grid. This interpolated data is denoted as $R(d[n, p])$. This equation can be then represented in the time domain as

$$A(p) = \sum_{n} R(d[n, p]) \exp(j4\pi d[n, p]/\lambda). \tag{2.34}$$

Note that the expected phase of $R(d[n, p])$ is $\exp(-j4\pi d[n, p]/\lambda)$, so the summation acts as the azimuth matched filter.

### LFM-CW SAR Backprojection

Similar to backprojection for pulsed SAR, backprojection for LFM-CW SAR consists of coherently summing the radar data to generate the image. The equation for LFM-CW SAR backprojection for a pixel $p$ can be written as [11] [15]

$$A(p) = \sum_n R(d[n,p] - Kx[n,p]/d[n,p]) \exp\{j\Phi_e(d[n,p])\} \tag{2.35}$$

where

$$\Phi_e(d[n,p]) = \frac{4\pi k_r d[n,p]^2}{c_0^2} - \frac{4\pi d[n,p]}{\lambda} - \pi k_r \mathbf{d^2} + 2\pi f_0 \mathbf{d} \tag{2.36}$$

where $d[n,p]$ is the distance from the $n^{th}$ pulse to a pixel $p$, $R(d[n,p] - Kx[n,p]/d[n,p])$ is the motion-corrected range-compressed SAR data interpolated to slant range $d[n,p]$ (see [12]), $\mathbf{d}$ is the dechirp delay, $k_r$ is the chirp rate, $f_0$ is the transmit frequency, and $\lambda$ is the wavelength of the transmit frequency.

Although backprojection is straightforward to implement and can handle a variety of flight tracks, it can be computationally expensive. To obtain an image with $M \times N$ pixels from $L$ equally spaced antenna pulse positions, a total of $L \times M \times N$ square root calculations and transcendental computations must be performed, corresponding to a computational complexity of $O(N^3)$. This can become costly as $L, M$, and $N$ become large.

## 2.5 History of Factorized Backprojection

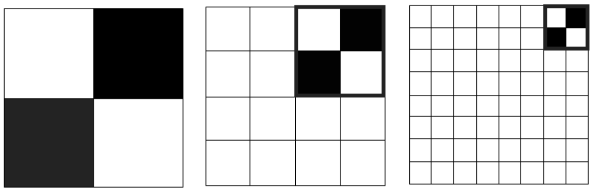
An alternative to backprojection is factorized backprojection, a time-domain algorithm which takes advantage of the redundancy of the SAR data to achieve complexity of $O(N^2 \log N)$. This redundancy is created because single small antennas correspond to wide beamwidth or coarse resolution, which allows for data reuse within the same range bin.

Historically, there have been two general approaches to factorized backprojection, namely the quadtree approach and the polar approach, or a combination of the two. The basics of the two major approaches are discussed in the following sections.

### 2.5.1 Quadtree Approach to Factorized Backprojection

One formulation of factorized backprojection introduced by Rofheart and McCorkle [4] performs the factorization in the context of a quadtree. Several variations on the quadtree have been developed [5] [6].

The quadtree approach divides the image reconstruction into a series of stages. In the first stage, all of the pulses are used to reconstruct the image with coarse resolution. In subsequent stages, the resolution of the image improves by a factor of four as an image is partitioned into square subimages until the final stage where a subimage is the size of a high-resolution pixel (see Fig. 2.2).



(a) Step 1 (b) Step 2 (c) Step 3

**Figure 2.2:** Illustration of the quadtree-based factorized backprojection algorithm. In the first step, short antenna arrays are used to reconstruct images with coarse resolution. As the length of the antenna arrays increases, the resolution becomes finer until the antenna array is the entire SAR array and the image has the desired resolution.

The algorithm is as follows. First, the distance from each pulse to the center of the image is calculated, and the corresponding range-compressed data is stored. For the next step, adjacent pulses are combined to form longer subapertures, and the image is split into four subimages. Then, the distance from the center of each subaperture to the center of each subimage is found. The corresponding range-compressed data corresponding to each subimage/subaperture pair is formed by recursively combining the parent data stored from the previous step. As long as the parent data corresponds to the same range bin as the child data, the parent data can be reused without error. Note that as the algorithm progresses, each subaperture increases in length, corresponding to narrower beamwidth. Simultaneously, the subimage becomes smaller, so it is still possible for the parent data and child data to correspond to the same smaller range bin.

This process continues until a subaperture consists of the entire length of the antenna array and a subimage is the size of a high-resolution pixel. The child data is backprojected, and the image is reconstructed.

The quadtree approach achieves its computational gain because as the number of subimages increases by a factor of four in each step, the number of subapertures decreases by the same factor. Thus, the total computational complexity of each step is $O(N^2)$. The total number of steps is $x$, where $4^x = N =$ number of high-resolution pixels. Solving for $x$, $x = \log_4 N$, so the algorithm has complexity $O(N^2 \log N)$.

Despite its computational gains, quadtree backprojection has several disadvantages. Because of the assumption that parent data corresponds to the same range bin as the child data, there can be high errors when the parent data is sparsely sampled over the entire imaging grid. Thus, the algorithm must be complemented with a mechanism of controlling the error to prevent image quality degradation [1]. Additionally, the algorithm is not easily parallelizable and does not include an implementation of an azimuth window to reduce sidelobes.

### 2.5.2 Polar Approach to Factorized Backprojection

An alternate approach to factorized backprojection is to represent images in local polar coordinates to reduce the number of operations [1] [7] [16]. As shown in Fig. 2.3, adjacent aperture positions have essentially the same circular pattern within a triangle shaped subimage. Hence, data corresponding to one aperture can be reused in an adjacent aperture with little loss in accuracy.

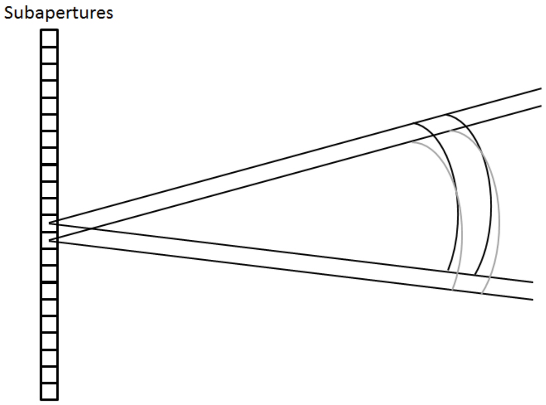


**Figure 2.3:** Subaperture beam formation. Adapted from [1].

The algorithm is similar to the quadtree algorithm in that it is divided into a series of steps wherein the resolution increases as the subaperture increases in length. However, unlike the quadtree algorithm, the data is kept in polar form where the coordinates of the polar grid correspond to the center of a given subaperture. This allows for efficient computation of data within the beamwidth of the given aperture.

On the first step, a subaperture is simply an antenna position, corresponding to wide beamwidth. For each range bin within the given beamwidth of the subaperture, a single data point is computed, corresponding to coarse resolution.

On the next step, two adjacent subapertures are combined to create a longer subaperture with narrower beamwidth. The narrower beamwidth allows for finer angular resolution which is obtained by combining coarse resolution beams from the parent subapertures. The new backprojection data is then computed by interpolation in range and angle of data from two parent sets of beams corresponding to the parent subapertures. Note that the number of operations stays constant over each step because the number of subapertures decreases at the same rate as the resolution increases. This process of increasing the angular resolution while decreasing the number of subapertures continues until the beamwidth of each subaperture is narrow enough to achieve the desired resolution. The computed beam points are then located on a Cartesian grid. Since each of the $\log N$ processing stage has the same number of operations $N^2$, the computational complexity is $O(N^2 \log N)$.

Although the polar factorized backprojection algorithm achieves low computational complexity, there are several shortcomings. Because of the polar nature of the algorithm, it is better suited for spotlight SAR than stripmap SAR and for ultrawideband signals rather than bandpass signals. Thus, it can be difficult to implement polar factorized backprojection for stripmap SAR. Furthermore, the interpolation of the polar data onto a Cartesian grid can be computationally intense.

Additionally, the polar factorized backprojection algorithm has high memory requirements in order to store the intermediate results. To overcome these requirements, factorization into quadtrees is required, with a penalty in computational gain. Finally, as with the quadtree algorithm, this algorithm is not easily parallelizable and includes no implementation of an azimuth window.

### 2.5.3 Factorized Backprojection for Stripmap SAR

Although most of the past work for factorized backprojection has been done for spotlight SAR, there has been some research on efficient algorithms for stripmap SAR [17]. These algorithms achieve their efficiency by reusing data from adjacent pulses because two adjacent pulses have similar antenna footprints. To achieve fine resolution, the data is upsampled and then interpolated.

Though the factorized backprojection algorithms developed up to this point have been computationally efficient, they have all had their drawbacks for stripmap SAR. Many algorithms both for spotlight and stripmap SAR have required upsampling and interpolation, which can be computationally inefficient. Additionally, no previous algorithm included the implementation of an azimuth window. The factorized backprojection algorithm introduced in the following chapter overcomes these limitations while providing a relatively straightforward implementation of a factorized backprojection algorithm.

# Chapter 3

# Windowed Factorized Backprojection for Stripmap SAR

This chapter introduces a new formulation of a factorized backprojection algorithm. Unlike past algorithms, this algorithm is designed specifically for pulsed and LFM-CW stripmap SAR and takes advantage of stripmap geometry to achieve lower computational complexity. It is also shown in this chapter how to utilize the stripmap geometry to implement an azimuth window as part of the factorized backprojection algorithm. Finally, the computational and memory requirements of the algorithms are discussed.

## 3.1 Factorized Backprojection for Pulsed SAR

In factorized backprojection, the image reconstruction is divided into a series of steps in which the resolution of the image becomes finer as the length of a synthetic subaperture increases. The geometry of the SAR array allows the interpolated radar data associated with the subapertures of the previous step to be used in subsequent steps, reducing the required computation at a tradeoff of some loss of accuracy.

Although the formulation of factorized backprojection presented here uses the same recursive principles as the previous algorithms, there are some notable differences. First, this particular implementation is designed for stripmap SAR and assumes that the flight track is straight. Second, rather than divide the image into square subimages or use polar coordinates, we split the image into columns, which are defined as a region of the image one pixel wide in the range direction (see Fig. 3.1). By splitting the image into columns, both the explanation and the implementation of the algorithm are simplified. Additionally, the algorithm can be easily parallelized since each column can be computed independent of the others. A high-level flow diagram highlighting the major steps of this factorized backproejction algorithm is shown in Fig. 3.2.

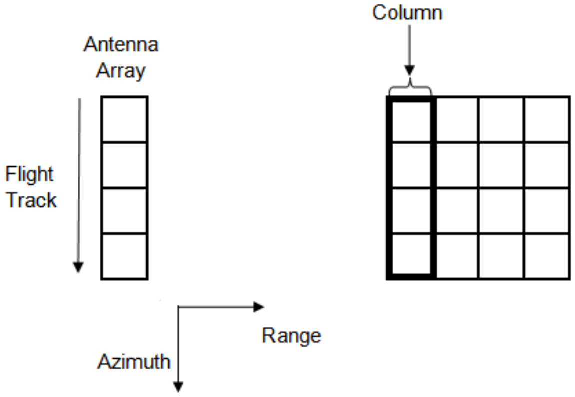


**Figure 3.1:** Left: Notional antenna phase center positions. Each position corresponds to the antenna location for a transmit/receive pulse. Right: Imaging grid with a single column highlighted.

We now describe this factorized backprojection algorithm in detail. Suppose there are $L$ collected pulses with which we wish to image an area comprised of $M \times N$ pixels. Then, the number of stages is $\min\{\log_2 L, \log_2 M\}$, in addition to a preliminary stage. For this explanation, we assume $L = M = N = 4$ and that the pulses and pixels are equally spaced. In practice, however, $L$, $M$, and $N$ do not need to be equal, nor do the pulses and pixels need to be equally spaced. We note that a pixel must lie in the beamwidth of the real aperture to be fully reconstructed. For pixels on the edge of an image, reconstruction requires antenna positions that extend beyond the imaging grid.

Initially, each subaperture corresponds to the actual antenna positions for each collected pulse, but in later steps it corresponds to the combination of two or more adjacent antenna positions. We divide the image into subimages, or sections of columns. Initially, a subimage consists of a single large area covering the entire column, but by the final stage, each of the multiple subimages is a single pixel of the column. (To reduce error, a subimage may initially consist of a portion of a column rather than the entire column, but this increases the total number of computations despite decreasing the number of steps.) Because the same algorithm is applied for each column independent of the other columns, we concentrate on a single column in this explanation.

Since the central positions of both subimages and subapertures change for each step of the factorization, we introduce some notation to aid in the explanation. Let $n_i^{(s)}$ index the center of the

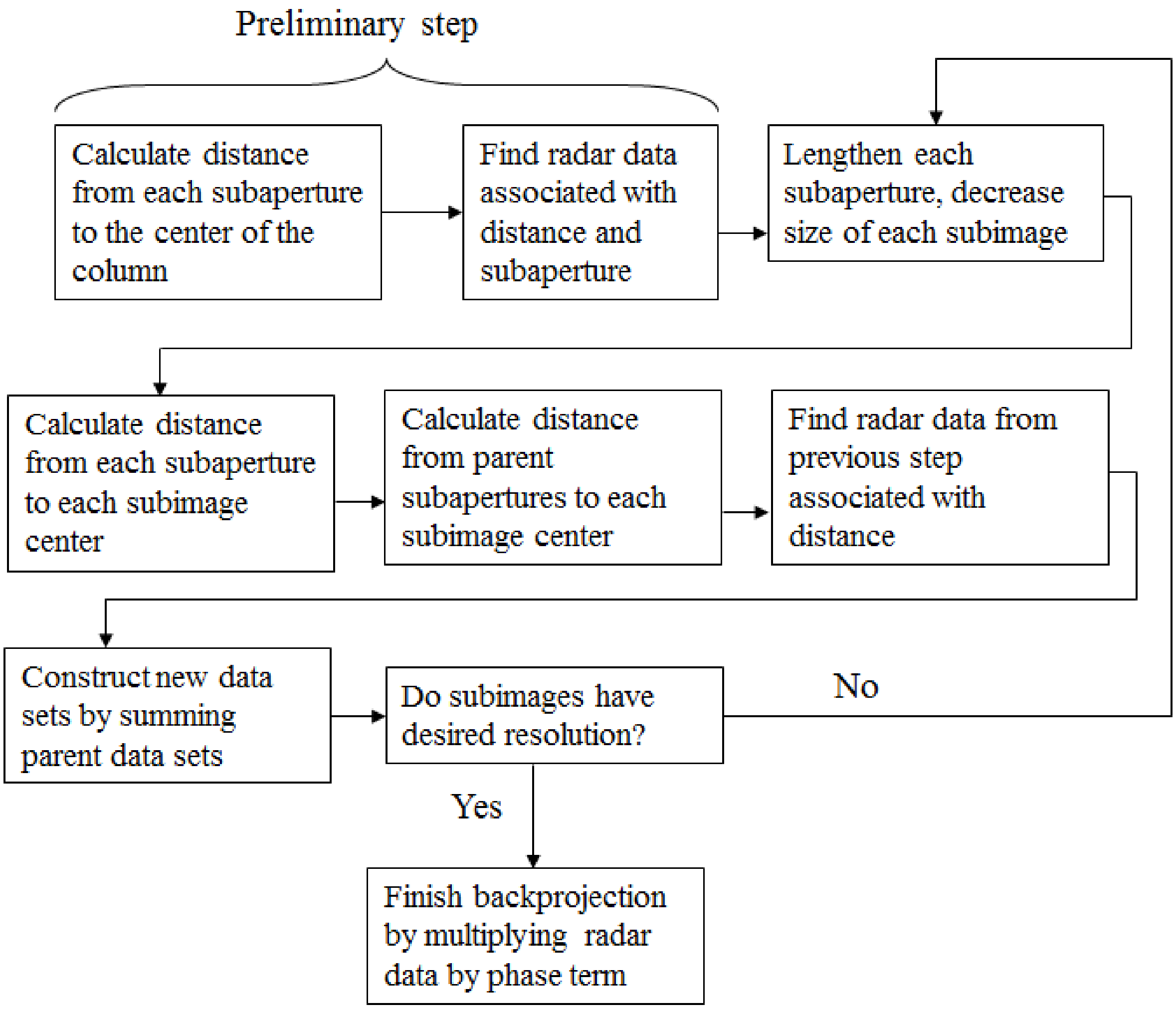

Preliminary step
Calculate distance from each subaperture to the center of the column
Find radar data associated with distance and subaperture
Lengthen each subaperture, decrease size of each subimage
Calculate distance from each subaperture to each subimage center
Calculate distance from parent subapertures to each subimage center
Find radar data from previous step associated with distance
Construct new data sets by summing parent data sets
Do subimages have desired resolution?
No
Yes
Finish backprojection by multiplying radar data by phase term

**Figure 3.2:** Flow diagram for factorized backprojection.

$i^{th}$ subaperture on the $s^{th}$ step. Let $p_k^{(s)}$ index the center of the $k^{th}$ subimage on the $s^{th}$ step in the along track direction. The distance from the $i^{th}$ subaperture center to the $k^{th}$ subimage is denoted $d[n_i^{(s)}, p_k^{(s)}]$ (see Fig. 3.3) and the interpolated range-compressed complex SAR data set associated with this subaperture-subimage pair is denoted $R(d[n_i^{(s)}, p_k^{(s)}])$. In the preliminary step, the data set is the range-compressed SAR data interpolated to slant range, but in subsequent steps the data set is formed from combinations of elements from the parent data set.

In the preliminary step of the algorithm, the distance from each subaperture center (pulse) to a subimage center is calculated. Since our example involves four pulses and one initial subimage, this step requires four distance calculations. In Fig. 3.4(a), which shows the preliminary step of the algorithm, the central pixel is denoted $p_0^{(0)}$, and each pulse is denoted as $n_i^{(0)}, i = 0...3$. Once each distance $[n_i^{(0)}, p_0^{(0)}]$ has been calculated, the radar echo data $R(d[n_i^{(0)}, p_0^{(0)}])$ is found from the range-compressed SAR data.

For the first factorization step, the number of subapertures is decreased by a factor of two by combining the parent subapertures into longer child subapertures. Because the resulting subapertures are longer than the parent subapertures, the corresponding beamwidth is narrower. In addition, the subimage is divided in half so that there are two pixels per column rather than one (see Fig. 3.4(b)).

The distance from each subaperture center $n_i^{(1)}$ to each subimage center $p_k^{(1)}$ is calculated, where $n_i^{(1)}$ has coordinates $(x_i, y_i, z_i)$ and $p_k^{(1)}$ has coordinates $(x_k, y_k, z_k)$. Then, the distance from each parent subaperture center $n_j^{(0)}$ to each subimage center $p_k^{(1)}$ is calculated or approximated. Given a parent subaperture $n_j^{(0)}$ with coordinates $(x_j, y_j, z_j)$, the distance from $n_j^{(0)}$ to the $k^{th}$ subimage center is given by

$$d[n_j^{(0)}, p_k^{(1)}] = \sqrt{(x_j - x_k)^2 + (y_j - y_k)^2 + (z_j - z_k)^2}. \tag{3.1}$$

If the flight track is ideal (i.e., parallel to the image column) and the imaging area is flat, then the distance can be approximated using the first terms of a Taylor series:

$$d[n_j^{(0)}, p_k^{(1)}] \approx d[n_i^{(1)}, p_k^{(1)}] + \Delta r \tag{3.2}$$

where

$$\Delta r = \frac{2(y_i - y_j)(y_j - y_k) + (y_j - y_i)^2}{2d[n_i^{(1)}, p_k^{(1)}]} \tag{3.3}$$

(see Fig. 3.4(c)). Note that for our column-based algorithm where the area to be imaged is a flat surface, $x_j = x_i$ and $z_j = z_i$.

Because the child subapertures are longer than the original subapertures, there is no previously interpolated radar data corresponding exactly to these new subapertures. However, we

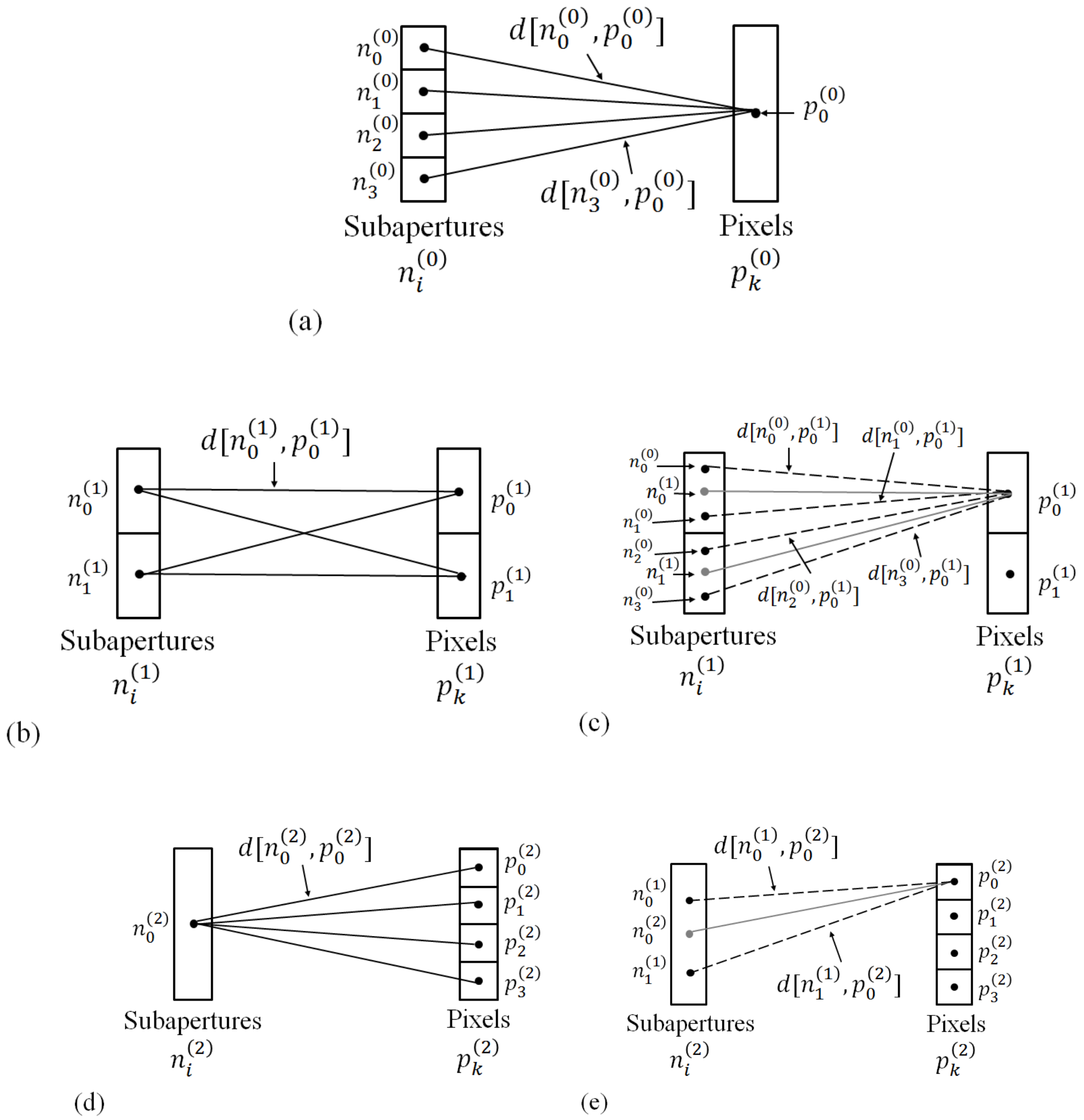


**Figure 3.3:** Illustration of distance calculations for factorized backprojection algorithm. (a) Distance from current subaperture centers to current subimage centers for preliminary step; (b) distance from current subaperture centers to current subimage centers for first step; (c) distance from parent subaperture centers to one of two current subimage centers for first step; (d) distance from current subaperture centers to current subimage centers for second step; (e) distance from parent subaperture centers to one of four current subimage centers for second step.

can construct data sets $R(d[n_i^{(s)}, p_k^{(s)}])$ corresponding to these longer subapertures by combining the data sets from parent subapertures and multiplying by a phase factor to compensate for the difference in distances:

$$R(d[n_i^{(s)}, p_k^{(s)}]) = \sum_{n_j \ni n_i} R(d[n_j^{(s-1)}, p_k^{(s)}]) \exp(j4\pi/\lambda \Delta r_j) \tag{3.4}$$

where

$$\Delta r_j = d[n_j^{(s-1)}, p_k^{(s)}] - d[n_j^{(s)}, p_k^{(s)}] \tag{3.5}$$

or if the prior distances are calculated with a Taylor series approximation,

$$\Delta r_j = \frac{2(y_i - y_j)(y_j - y_k) + (y_j - y_i)^2}{2d[n_i^{(s)}, p_k^{(s)}]}. \tag{3.6}$$

Rather than directly calculating $R(d[n_j^{(s-1)}, p_k^{(s)}])$, we approximate it by data sets formed in the previous step because these parent data sets include the phase factor as shown in Eq. (3.4). $R(d[n_j^{(s-1)}, p_k^{(s)}])$ is then given by

$$R(d[n_j^{(s-1)}, p_k^{(s)}]) \approx R(d[n_j^{(s-1)}, p_{\lfloor k/2 \rfloor}^{(s-1)}]). \tag{3.7}$$

If $d[n_j^{(s-1)}, p_k^{(s)}] = d[n_j^{(s-1)}, p_{\lfloor k/2 \rfloor}^{(s-1)}]$, then the approximation is exact since both values correspond to the same range bin. However, if the distances are not equal, the approximate data set may not correspond to the same range bin as the correct data set, so there may be quantization error. Additionally, if the distances are not equal, the incorrect phase may be computed in Eq. (3.4). We discuss these errors more in Section 4.1.

For the remaining iterations, the process of lengthening subapertures and decreasing subimage size continues until a subimage is a single pixel and there is only one subaperture covering the full length with center $n_c$ (see Fig. 3.4(d) and 3.4(e)). The backprojected image for a pixel $p_k$ is given by

$$A(p_k) = R(d[n_c, p_k]) \exp(j4\pi/\lambda d[n_c, p_k]). \tag{3.8}$$

Since $R(d[n_c, p_k])$ has been formed from parent data sets each corresponding to smaller subapertures, if we consider $R(d[n_c, p_k])$ in terms of its parent data sets we find

$$\begin{aligned} A(p_k) &= \sum_n R(d[n, p_k]) \exp\{j4\pi/\lambda\, (d[n, p_k] - d[n_c, p_k])\} \exp\{j4\pi/\lambda d[n_c, p_k]\} \\ &= \sum_n R(d[n, p_k]) \exp\{j4\pi/\lambda d[n, p_k]\} \end{aligned} \tag{3.9}$$

which is the original backprojection equation shown in Eq. (2.34). Due to some quantization error, the expansion is not exact, so the image may have some loss of accuracy. This loss of accuracy is discussed more in Chapter 4.

### 3.2 Factorized Backprojection for LFM-CW SAR

In this section, we discuss how to implement factorized backprojection for LFM-CW SAR. There are few differences between factorized backprojection for pulsed and LFM-CW SAR, with the most notable difference in the two algorithms being the formulation of the phase. This is due to the different formulation of the backprojection equation, which is given by

$$A(p) = \sum_n R(d[n, p] - Kx[n, p]/d[n, p]) \exp\{j\Phi_e(d[n, p])\} \tag{3.10}$$

where

$$\Phi_e(d[n, p]) = \frac{4\pi k_r d[n, p]^2}{c_0^2} - \frac{4\pi d[n, p]}{\lambda} - \pi k_r \mathbf{d^2} + 2\pi f_0 \mathbf{d} \tag{3.11}$$

where $d[n, p]$ is the distance from the $n^{th}$ pulse to a pixel $p$, $R(d[n, p] - Kx[n, p]/d[n, p])$ is the motion-corrected range-compressed SAR data interpolated to slant range $d[n, p]$ (see [12]), $\mathbf{d}$ is the dechirp delay, $k_r$ is the chirp rate, $f_0$ is the transmit frequency, and $\lambda$ is the wavelength of the transmit frequency. For the sake of simplicity we choose to neglect $Kx[n, p]/d[n, p]$ to make the factorization easier, though this causes a slight defocusing of the image. $\Phi_e(d[n, p])$ is split into separate terms as

$$\Phi_e(d[n, p]) = \rho(d[n, p]) + \delta \tag{3.12}$$

where

$$\rho(d[n, p]) = \frac{4\pi k_r d[n, p]^2}{c_0^2} - \frac{4\pi d[n, p]}{\lambda}$$

and

$$\delta = -\pi k_r \mathbf{d}^2 + 2\pi f_0 \mathbf{d}.$$

The backprojection equation serves as the basis for the LFM-CW factorized backprojection algorithm.

As with pulsed SAR, in the preliminary step of the algorithm, the distance from each subaperture center (pulse) to a subimage center is calculated. Once each distance $[n_i^{(0)}, p_0^{(0)}]$ has been calculated, the radar echo data $R(d[n_i^{(0)}, p_0^{(0)}])$ is found from the range-compressed SAR data. For the first factorization step, the number of subapertures is decreased by a factor of two by combining the parent subapertures into longer child subapertures. Because the resulting subapertures are longer than the parent subapertures, the corresponding beamwidth is narrower. In addition, the subimage is divided in half so that there are two pixels per column rather than one (see Fig. 3.4(b))

The distance from each subaperture center $n_i^{(1)}$ to each subimage center $p_k^{(1)}$ is then calculated, where $n_i^{(1)}$ has coordinates $(x_i, y_i, z_i)$ and $p_k^{(1)}$ has coordinates $(x_k, y_k, z_k)$. Then, the distance from each parent subaperture center $n_j^{(0)}$ to each subimage center $p_k^{(1)}$ is calculated or approximated.

Because the child subapertures are longer than the original subapertures, there is no previously interpolated radar data corresponding exactly to these new subapertures. However, as with pulsed SAR, we can construct intermediate data sets $R(d[n_i^{(s)}, p_k^{(s)}])$ corresponding to these longer subapertures by combining the data sets from parent subapertures and multiplying by a phase factor to compensate for the difference in distances:

$$R(d[n_i^{(s)}, p_k^{(s)}]) = \sum_{n_j \ni n_i} R(d[n_j^{(s-1)}, p_k^{(s)}]) \exp(j\Delta\rho_j) \tag{3.13}$$

where

$$\Delta\rho_j = \rho(d[n_j^{(s-1)}, p_k^{(s)}]) - \rho(d[n_j^{(s)}, p_k^{(s)}]). \tag{3.14}$$

Note that the phase factor $\Delta\rho_j$ is not the same as the phase factor $\Delta r_j$ given for pulsed SAR.

Rather than directly calculating $R(d[n_j^{(s-1)}, p_k^{(s)}])$, we approximate it by data sets formed in the previous step, i.e.,

$$R(d[n_j^{(s-1)}, p_k^{(s)}]) \approx R(d[n_j^{(s-1)}, p_{\lfloor k/2 \rfloor}^{(s-1)}]). \tag{3.15}$$

The errors caused by this approximation are discussed in Section 4.1.

For the remaining iterations, the process of lengthening subapertures and decreasing subimage size continues until a subimage is a single pixel and there is only one subaperture covering the full length with center $n_c$ (see Fig. 3.4(d) and 3.4(e)). The backprojected image for a pixel $p_k$ is given by

$$A(p_k) = R(d[n_c, p_k]) \exp\{j(\rho(d[n_c, p_k]) + \delta)\}. \tag{3.16}$$

Since $R(d[n_c, p_k])$ has been formed from parent data sets each corresponding to smaller subapertures, if we consider $R(d[n_c, p_k])$ in terms of its parent data sets we find

$$\begin{aligned} A(p_k) &= \sum_n R(d[n, p_k]) \exp\{j[\rho(d[n, p_k]) - \rho(d[n_c, p_k])]\} \exp\{j[\rho(d[n_c, p_k]) + \delta]\} \\ &= \sum_n R(d[n, p_k]) \exp\{j(\rho(d[n_c, p_k]) + \delta)\} \\ &= \sum_n R(d[n, p_k]) \exp\{j\Phi_e(d[n, p_k])\} \end{aligned} \tag{3.17}$$

which is similar to the original backprojection equation shown in Eq. (3.10). Due to some quantization error, the expansion is not exact, so the image may have some loss of accuracy.

## 3.3 Windowed Factorized Backprojection

In SAR image processing, an azimuth window is often applied to minimize azimuth aliasing and suppress sidelobes at a cost of some loss in azimuth resolution. In this section, we show that an azimuth window can also be incorporated into both pulsed and LFM-CW factorized backprojection with little additional computation.

For direct backprojection, if an azimuth window is desired for some pixel $p_k$, one approach is to apply a weighting function to the backprojection equation:

$$A(p_k) = \sum_{n_i} W(n_i, p_k) R(d[n_i, p_k]) \exp(j\phi(d[n_i, p_k])) \tag{3.18}$$

where $W(n_i, p_k)$ is a weighting function expressed in terms of the pulse number $n_i$ and specified pixel $p_k$ and $\phi$ is the phase specific to either pulsed or LFM-CW SAR. In this thesis we consider weighting functions of the form

$$W(n_i, p_k) = \exp(-(n_{iy} - p_{ky})^2/a) \tag{3.19}$$

where $n_{iy}$ is the $y$-coordinate of $n_i$, $p_{ky}$ is the $y$-coordinate of $p_k$, $a$ is some constant, and the azimuth direction is in $y$. The output of the weighting function for a given pixel $p$ is a Gaussian curve, thus creating a window for the given pixel. We call this the *direct window*.

In factorized backprojection, implementing an azimuth window is more complex because the algorithm is divided into a series of steps. Since there is no single equation that depends on both an individual pulse $n_i$ and an individual pixel $p_k$, there is no equation where the weighting term $W(n_i, p_k)$ used in direct backprojection can be logically inserted. However, an alternative approach is to include intermediate weighting functions in the formation of the data sets for each step to create windowed data sets $R'(d[n^{(s)}_{\lfloor i/2^s \rfloor}, p^{(s)}_{\lfloor k/2^{S-s} \rfloor}])$. Then, in the final step of windowed factorized backprojection, the equation for a pixel $p_k$ takes the form

$$A(p_k) = R'(d[n_c, p_k]) \exp\left(j\phi(d[n_c, p_k])\right). \tag{3.20}$$

If this expression is expanded as in Eq. (3.9), then

$$A(p_k) = \sum_{n_i} R'(d[n_i, p_k]) \exp\left(j\phi(d[n_i, p_k])\right) \tag{3.21}$$

where

$$R'(d[n_i, p_k]) = R(d[n_i, p_k]) W_{\text{eff}}(n_i, p_k) \tag{3.22}$$

where $W_{\text{eff}}(n_i, p_k)$ is the effective weighting function formed in the steps of the algorithm corresponding to a pulse $n_i$ and a pixel $p_k$. We call the output of this weighting function the *factorized window*. Due to the factorization, the factorized window is not identical to the direct window. However, by the proper choice of intermediate weighting functions, the factorized window can be similar to the direct window.

We now discuss an intermediate weighting function that is easy to implement and which creates a factorized window that is similar to the direct window. Consider an intermediate subaperture center $n_i^{(s)}$ with parent subaperture center $n_j^{(s-1)}$ with coordinates $(n_{jx}, n_{jy})$ and an intermediate subimage center $p_k^{(s)}$ with coordinates $(p_{kx}, p_{ky})$. We define an intermediate weighting function $W(n_j^{(s-1)}, p_k)$ to weight the corresponding data set as

$$R'(d[n_i^{(s)}, p_k^{(s)}]) = \sum_{n_j \ni n_i} W(n_j^{(s-1)}, p_k) R'(d[n_j^{(s-1)}, p_k^{(s)}]) \exp\left\{ j\phi(d[n_j^{(s-1)}, p_k^{(s)}] - d[n_i^{(s)}, p_k^{(s)}]) \right\} \tag{3.23}$$

where

$$W(n_j, p_k) = \exp(-|n_{jy} - p_{ky}|/a) \tag{3.24}$$

with $a$ determined as a function of the beamwidth. Given a pulse $n_i$ and a pixel $p_k$, the resulting effective weighting function corresponding to $n_i$ and $p_k$ is

$$W_{\text{eff}}(n_i, p_k) = \exp(-|n_{iy} - p_{\lfloor k/2^S \rfloor y}|/a) \prod_{s=2}^{S} \exp(-|n_{\lfloor i/2^s \rfloor y}^{(s-1)} - p_{\lfloor k/2^{S-s} \rfloor y}^{(s)}|/a). \tag{3.25}$$

Figure 3.4 shows plots of the factorized window and direct window for given pixels located in various locations of an imaging grid. Note that the shape of the factorized window is similar to the shape of the direct window for each pixel. However, while the direct window has the same shape regardless of the pixel, the factorized window changes shape slightly for different pixels. This discrepancy is expected due to the creation of the window over a series of steps.

### 3.4 Computational and Memory Requirements

In this section, we discuss the computational and memory requirements associated with the factorized backprojection algorithms introduced in this chapter. Because the difference between factorized and windowed factorized backprojection involves only a few multiplies per step, we assume the computational burden is nearly identical.

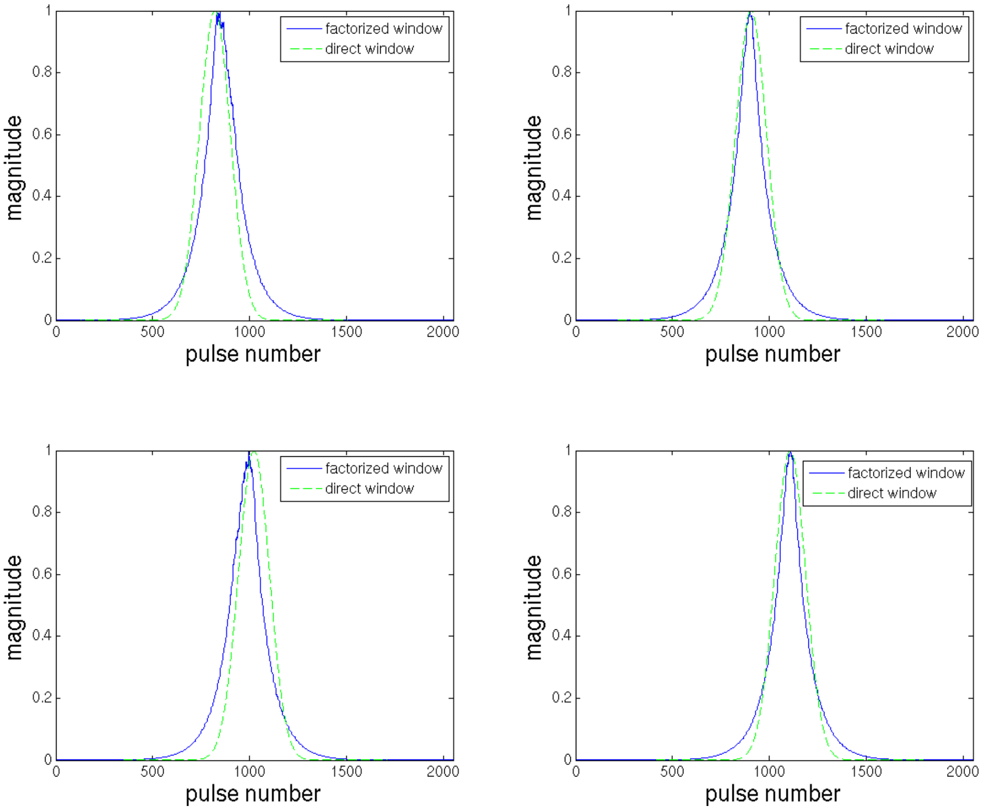


**Figure 3.4:** Effective factorized and actual weighting functions for various pixels in a column of 64 pixels. Upper left: pixel 1; upper right: pixel 14; lower left: pixel 32; lower right: pixel 45.

### 3.4.1 Computational Complexity

We now show that the windowed factorized backprojection algorithm has complexity $O(N^2 \log N)$. For simplicity's sake, we assume that there are $N$ antenna positions and the imaging pixel grid has $N \times N$ pixels.

There are a total of $\log_2 N$ steps. For each step $s$, there are $2^s$ pixels per column, $N$ columns, and $N/2^s$ subapertures. Thus the total number of operations per step is proportional to

$$\begin{aligned} \text{OPS} &= 2^s \cdot N \cdot \frac{N}{2^s} \\ &= N^2 \end{aligned}$$

so the total number of operations is proportional to

$$\text{OPS} = N^2 \log_2 N. \tag{3.26}$$

This is an $\frac{N}{\log N}$ improvement over the direct approach which has $O(N^3)$ operations as discussed in Section 2.4.2. Note that adding the window adds a few computations per step due to the exponential multiply but does not substantially increase the time. The cost for this lower computational complexity is a less straightforward algorithm and some error due to approximation. Interpolation and windowing decrease the error but slightly increase the time.

Note also that each column can be processed independently of the other columns. Thus the factorized backprojection algorithm can be parallelized efficiently.

### 3.4.2 Memory Requirements

We now consider the memory requirements of the factorized backprojection algorithm with $N$ pulses and an $L \times M$ grid. If the system can be parallelized, then there is less memory required than if the system cannot be parallelized. We examine both cases.

Suppose first that the system can be parallelized so that each column of the image can be processed independently. Since the original data collected by the radar is used only in the first step, the $m \times N$ range-compressed data matrix is necessary only for the first step, where $m$ is the number of samples corresponding to a given pulse (typically in the thousands). Within this first step, only the data corresponding to one sample per pulse is necessary for a parallelized system, so the total memory requirement for the original radar data is bounded by $N$. After this first step, the range-compressed data is no longer necessary and can be removed from memory.

For each subsequent step, both a parent data set and a child data set must be recorded. Each data set corresponds to $N/2^s$ pulses and $k2^s$ pixels in a column, where $k$ is the number of initial subimages, so the size remains constant for each step. Since the parent and child data sets are required, a total of $2kN$ memory locations are required. On the final step, there are $M$ memory locations for the $M$ pixels in each column, but this is generally less than $2kN$. Hence, the total memory required per parallel structure is bounded by $2kN$.

If the system is not parallelized, then the $L$ columns of the grid must be considered simultaneously in order to delete the range-compressed data from memory after the first step. Since the system is not parallelized, the entire range-compressed data matrix must be stored for the first step, corresponding to $mN$ memory locations. It is still possible to construct data sets with constant size $kN$, but $2L$ data sets at a time are required rather than the two required for a parallelized structure. On the final step, $LM$ memory locations are required for the $LM$ pixels on the image grid. If $L < 2kN$, the total memory required is bounded by $L \cdot 2kN$.

## 3.5 Conclusion

This chapter discussed the formulation of the factorized backprojection algorithm for pulsed and LFM-CW SAR. This algorithm achieves its computational gain by reusing radar data within a column and factoring the phase over a series of steps. It was also shown how to implement an azimuth window. A more thorough performance analysis of the algorithms is given in Chapter 4.

# Chapter 4

# Performance Analysis

This chapter discusses the performance of the windowed factorized backprojection algorithm. Sources of errors are discussed and equations are provided which give bounds on the expected error. Then, example imagery is provided to illustrate the performance of windowed factorized backprojection algorithm for pulsed and LFM-CW SAR compared to direct backprojection.

## 4.1 Errors in the Factorized Backprojection Algorithm

There are two types of errors associated with factorized backprojection: those caused by using incorrect distances for phase calculations and those caused by errors in the creation of data sets from the range interpolated data. We first discuss the phase error for pulsed and LFM-CW SAR separately. We then discuss the error associated with the creation of data sets (frequently referred to as range bin error), along with a possible way to minimize range bin error.

### 4.1.1 Phase Errors in Pulsed SAR

One type of error in factorized backprojection is the phase error caused by not directly calculating $\exp\{j4\pi d[n_i,p_k]/\lambda\}$ for each pulse $n_i$ and pixel $p_k$ and instead using an approximation formed over a series of steps. The effective phase term for a given pulse $n_i$ and pixel $p_k$ is of the form $\exp\{j4\pi\tilde{d}[n_i,p_k]/\lambda\}$ where

$$\tilde{d}[n_i,p_k] = \sum_{s=1}^{S}\left(d[n^{(s-1)}_{\lfloor i/2^s\rfloor}, p^{(s)}_{\lfloor k/2^{S-s}\rfloor}] - d[n^{(s)}_{\lfloor i/2^{s+1}\rfloor}, p^{(s)}_{\lfloor k/2^{S-s}\rfloor}]\right) + d[n^{(S)}_{\lfloor i/2^S\rfloor}, p_k] \qquad (4.1)$$

where $S$ is the number of steps in the algorithm. We refer to $\tilde{d}[n_i,p_k]$ as the *factorized distance*. Ideally, the actual distance $d[n_i,p_k]$ equals the factorized distance. However, in practice, this is not generally true. We can obtain an upper bound on the error by setting a single pixel and pulse as reference points and then defining the coordinates of the parent subimages and child subapertures in terms of these reference points.

Let a pixel $p_k$ have coordinates $(x_k, y_k, z_k)$ and let a pulse $n_i$ have coordinates $(x_i, y_i, z_i)$, where the azimuth direction is along the $y$-axis. Let $L_I$ be the length of the imaging grid, $P$ be the number of pixels in the imaging grid, $L_A$ be the length of the antenna array, and $N$ be the number of pulses. Let $R_0$ be the minimum distance from the SAR array to the column. Let $S_P = \log_2 P$, $S_N = \log_2 N$, and $S = \min\{S_P, S_N\}$. Then, a child subimage center $p^{(s)}_{\lfloor k/2^{S_P-s}\rfloor}$ has coordinates $(x_k, y^{(s)}_{\lfloor k/2^{S_P-s}\rfloor}, z_k)$, where

$$y^{(s)}_{\lfloor k/2^{S_P-s}\rfloor} = y^{(s-1)}_{\lfloor k/2^{S_P-s+1}\rfloor} + (-1)^{\lfloor k/2^{S_P-s}\rfloor} \frac{P}{2^{s+1}} \frac{L_I}{P-1}. \tag{4.2}$$

Similarly, a child subaperture center $n^{(s)}_{\lfloor i/2^s\rfloor}$ has coordinates $(x_i, y^{(s)}_{\lfloor i/2^s\rfloor}, z_i)$, where

$$y^{(s)}_{\lfloor i/2^s\rfloor} = y^{(s-1)}_{\lfloor i/2^{s-1}\rfloor} + (-1)^{\lfloor i/2^{s-1}\rfloor} \frac{N}{2^{S_N-s+2}} \frac{L_A}{N-1}. \tag{4.3}$$

Let $\Delta^{(s)}_k = y^{(s)}_{\lfloor k/2^{S_P-s}\rfloor} - y_k$ and $\Delta^{(s)}_i = y^{(s)}_{\lfloor i/2^s\rfloor} - y_i$. Using these relationships, the error $\varepsilon$ between the actual distance and the factorized distance from a pulse $n_i$ and a pixel $p_k$ can be written as

$$\begin{aligned}
\varepsilon =& d[n_i, p_k] - \left\{ \sum_{s=1}^{S} \left[ d[n^{(s-1)}_{\lfloor i/2^s\rfloor}, p^{(s)}_{\lfloor k/2^{S-s}\rfloor}] - d[n^{(s)}_{\lfloor i/2^{s+1}\rfloor}, p^{(s)}_{\lfloor k/2^{S-s+1}\rfloor}] \right] + d[n^{(S)}_{\lfloor i/2^S\rfloor}, p_k] \right\} \\
=& \sqrt{R_0^2 + (y_i - y_k)^2} - \left\{ \sum_{s=1}^{S} \left[ \sqrt{R_0^2 + (y_i + \Delta^{(s-1)}_i - y_k - \Delta^{(s)}_k)^2} - \sqrt{R_0^2 + (y_i + \Delta^{(s)}_i - y_k - \Delta^{(s)}_k)^2} \right] \right. \\
& \left. + \sqrt{R_0^2 + (y_i + \Delta^S_i - y_k)^2} \right\}.
\end{aligned} \tag{4.4}$$

We can approximate $\varepsilon$ by $\tilde{\varepsilon}$, where $\tilde{\varepsilon}$ is the Taylor series approximation given by

$$\begin{aligned}
\tilde{\varepsilon} =& R_0 + \frac{1}{2R_0}(y_i - y_k)^2 - \left\{ \sum_{s=1}^{S} \left[ R_0 + \frac{1}{2R_0}(y_i + \Delta^{(s-1)}_i - y_k - \Delta^{(s)}_k)^2 - R_0 \right. \right. \\
& \left. \left. - \frac{1}{2R_0}(y_i + \Delta^{(s)}_i - y_k - \Delta^{(s)}_k)^2 \right] + R_0 + \frac{1}{2R_0}(y_i + \Delta^{(S)}_i - y_k)^2 \right\} \\
=& \frac{1}{2R_0} \left\{ (y_i - y_k)^2 - \sum_{s=1}^{S} \left[ (y_i + \Delta^{(s-1)}_i - y_k - \Delta^{(s)}_k)^2 - (y_i + \Delta^{(s)}_i - y_k - \Delta^{(s)}_k)^2 \right] - (y_i + \Delta^{(S)}_i - y_k)^2 \right\}.
\end{aligned} \tag{4.5}$$

By canceling and rearranging terms and noting that $\Delta_i^{(0)} = 0$, this equation can be further simplified as

$$\begin{aligned}\tilde{\varepsilon} =& \frac{1}{2R_0}\left[2\sum_{s=2}^{S}\left(\Delta_i^{(s-1)}\Delta_k^{(s)} - \Delta_i^{(s-1)}\Delta_k^{(s-1)}\right) - 2\Delta_i^{(S)}\Delta_k^{(S)}\right] \\ =& \frac{1}{2R_0}\left[2\sum_{s=2}^{S}\left(\Delta_i^{(s-1)}[\Delta_k^{(s)} - \Delta_k^{(s-1)}]\right) - 2\Delta_i^{(S)}\Delta_k^{(S)}\right]. \end{aligned} \tag{4.6}$$

We note that

$$\begin{aligned}\Delta_k^{(s)} - \Delta_k^{(s-1)} =& (y^{(s)}_{\lfloor k/2^{S_P - s}\rfloor} - y_k) - (y^{(s-1)}_{\lfloor k/2^{S_P - s+1}\rfloor} - y_k) \\ =& y^{(s)}_{\lfloor k/2^{S_P - s}\rfloor} - y^{(s-1)}_{\lfloor k/2^{S_P - s+1}\rfloor} \\ =& (-1)^{\lfloor k/2^{S_P - s}\rfloor}\frac{P}{2^{s+1}}\frac{L_I}{P-1} \\ \leq& \frac{P}{2^{s+1}}\frac{L_I}{P-1}. \end{aligned} \tag{4.7}$$

Thus,

$$\tilde{\varepsilon} \leq \frac{1}{2R_0}\left[2\sum_{s=2}^{S}\left(\Delta_i^{(s-1)}\frac{P}{2^{s+1}}\frac{L_I}{P-1}\right) - 2\Delta_i^{(S)}\Delta_k^{(S)}\right]. \tag{4.8}$$

Using the triangle inequality, we can further bound $\tilde{\varepsilon}$ by

$$\tilde{\varepsilon} \leq \frac{1}{2R_0}\left[2\sum_{s=2}^{S}\left|\Delta_i^{(s-1)}\frac{P}{2^{s+1}}\frac{L_I}{P-1}\right| + 2|\Delta_i^{(S)}\Delta_k^{(S)}|\right]. \tag{4.9}$$

Since for any given pulse $n_i$,

$$\Delta_i^{(s)} \leq \frac{N}{2^{S_N - s+1}}\frac{L_A}{N-1} \approx \frac{L_A}{2^{S_N - s+1}}$$

and for any given pixel $p_k$,

$$\Delta_k^{(s)} \leq \frac{P}{2^s}\frac{L_I}{P-1} \approx \frac{L_I}{2^s}$$

we can further simplify the bound in Eq. (4.9) as

$$\begin{aligned}\tilde{\varepsilon} \leq & \frac{1}{R_0}\left[\sum_{s=2}^{S}\left(\frac{L_A}{2^{S_N-s+2}}\frac{L_I}{2^{s+1}}\right)+\frac{L_A}{2^{S_N-s+1}}\frac{L_I}{2^S}\right] \\ = & \frac{1}{R_0}\left[\sum_{s=2}^{S}\left(\frac{L_I L_A}{2^{S_N+3}}\right)+\frac{L_I L_A}{2^{S_N+2}}\right] \\ = & \frac{1}{R_0}\left[(S-1)\frac{L_I L_A}{2^{S_N+3}}+\frac{2L_I L_A}{2^{S_N+3}}\right] \\ = & \frac{1}{8R_0}\frac{(S+1)L_I L_A}{2^{S_N}}. \end{aligned} \tag{4.10}$$

Note the similarity of this error bound to that given by [1]. From this equation, we see that the distance error can be reduced by decreasing the length of the image to be reconstructed. Similarly, by initially dividing a column into several subimages rather than performing factorized backprojection for the entire column, the error is reduced because each subimage is shorter. However, this requires more computation. Figure 4.1 shows the distance error for simulated data for a given pixel and varying numbers of initial subimages.

Recall that $\varepsilon$ is the difference between the actual distance and factorized distance for a given pulse and pixel. We may assume that a phase error of $\exp\{j\pi/8\}$ is acceptable, that is, there is negligible error in the image if

$$(4\pi/\lambda)\,|\varepsilon| \leq \pi/8 \tag{4.11}$$

which implies

$$|\varepsilon| \leq \lambda/32. \tag{4.12}$$

For the simulation described in Section 4.2.1 whose error plot is shown in Fig. 4.1, the wavelength of the transmit frequency is 0.0292 m, so $\lambda/32$ = $9.1250 \times 10^{-4}$. In Fig. 4.1, the bound on the magnitude of the distance error is less than this value for each initial subimage number.

### 4.1.2 Phase Errors in LFM-CW SAR

Recall that

$$\rho(d[n_i,p_k]) = \frac{4\pi k_r d[n_i,p_k]^2}{c_0^2} - \frac{4\pi d[n_i,p_k]}{\lambda}.$$

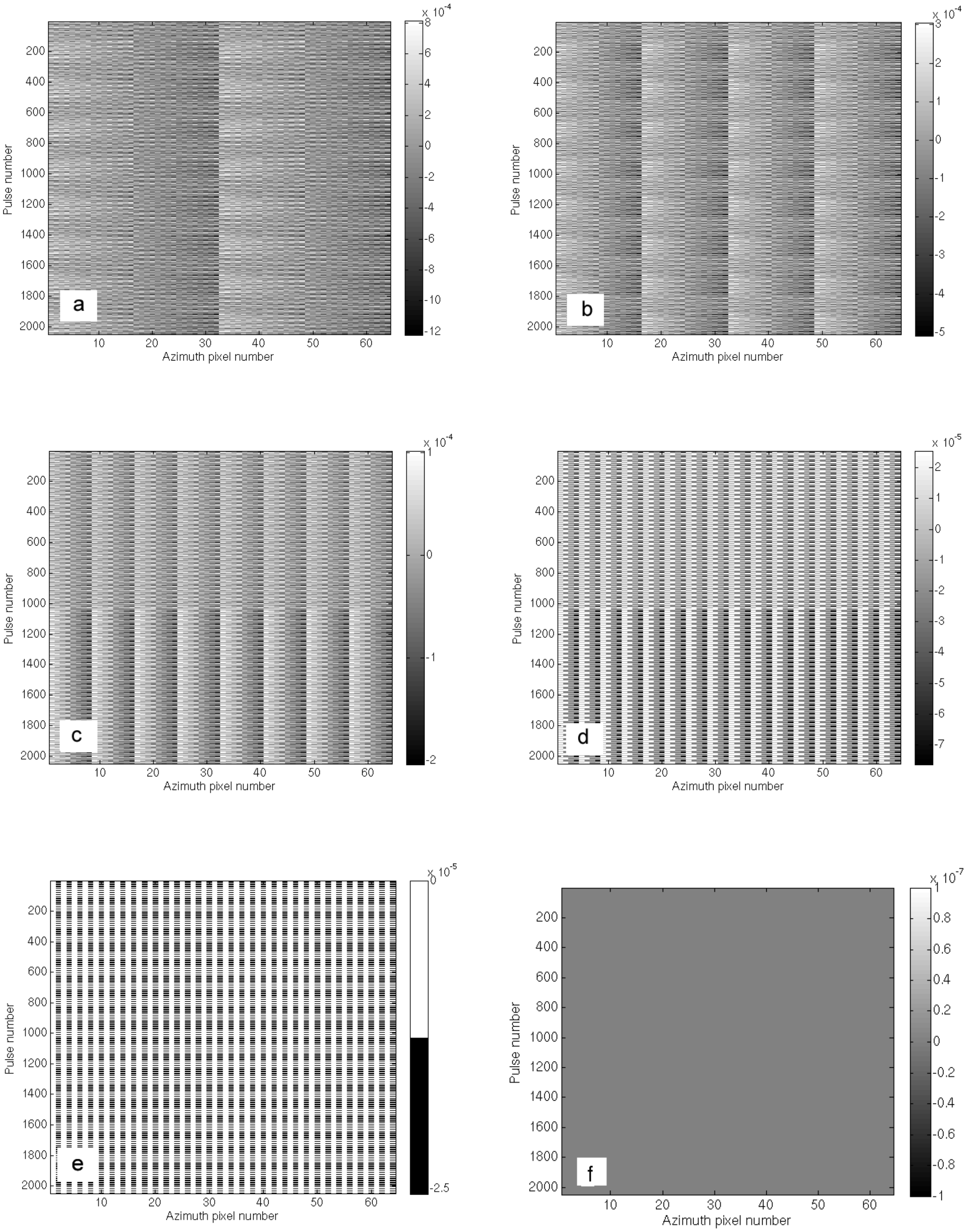


**Figure 4.1:** Difference between actual and factorized distances for each pixel within a column and each pulse in the antenna array for the parameters in Table B.1. (a) error with one initial subimage; (b) error with two initial subimages; (c) error with four initial subimages; (d) error with eight initial subimages; (e) error with sixteen initial subimages; (f) error with thirty-two initial subimages (that is, there is zero phase error because each distance is calculated correctly). Note that if more inital subimages are used, the magnitude of the error is smaller.

The effective phase term for a given pulse $n_i$ and pixel $p_k$ is of the form $\exp\{j\rho(\tilde{d}[n_i, p_k])\}$ where

$$\tilde{d}[n_i, p_k] = \sum_{s=1}^{S} \left( d[n^{(s-1)}_{\lfloor i/2^s \rfloor}, p^{(s)}_{\lfloor k/2^{S-s} \rfloor}] - d[n^{(s)}_{\lfloor i/2^{s+1} \rfloor}, p^{(s)}_{\lfloor k/2^{S-s} \rfloor}] \right) + d[n^{(S)}_{\lfloor i/2^S \rfloor}, p_k] \tag{4.13}$$

where $S$ is the number of steps in the algorithm. We refer to $\tilde{d}[n_i, p_k]$ as the *factorized distance*. Ideally, the actual distance $d[n_i, p_k]$ equals the factorized distance. However, in practice, this is not generally true since the factorized distance is formed by computing the distance between subaperture and subimage centers on each step rather than the distance between the actual pulse and pixel. This creates a phase error (in radians) of

$$\begin{aligned} \varepsilon =& \rho(d[n_i, p_k]) - \rho(\tilde{d}[n_i, p_k]) \\ =& \frac{4\pi}{\lambda}\varepsilon_1 + \frac{4\pi k_r}{c_0^2}\varepsilon_2 \end{aligned} \tag{4.14}$$

where

$$\varepsilon_1 = d[n_i, p_k] - \tilde{d}[n_i, p_k]$$

and

$$\varepsilon_2 = d[n_i, p_k]^2 - \tilde{d}[n_i, p_k]^2.$$

We can obtain an upper bound on the error by setting a single pixel and pulse as reference points and then defining the coordinates of the parent subimages and child subapertures in terms of these reference points.

Let a pixel $p_k$ have coordinates $(x_k, y_k, z_k)$ and let a pulse $n_i$ have coordinates $(x_i, y_i, z_i)$, where the azimuth direction is along the $y$-axis. Let $L_I$ be the length of the imaging grid, $P$ be the number of pixels in the imaging grid, $L_A$ be the length of the antenna array, and $N$ be the number of pulses. Let $R_0$ be the minimum distance from the SAR array to the column. Let $S_P = \log_2 P$, $S_N = \log_2 N$, and $S = \min\{S_P, S_N\}$. Then, a child subimage center $p^{(s)}_{\lfloor k/2^{S_P-s} \rfloor}$ has coordinates $(x_k, y^{(s)}_{\lfloor k/2^{S_P-s} \rfloor}, z_k)$, where

$$y^{(s)}_{\lfloor k/2^{S_P-s} \rfloor} = y^{(s-1)}_{\lfloor k/2^{S_P-s+1} \rfloor} + (-1)^{\lfloor k/2^{S_P-s} \rfloor} \frac{P}{2^{s+1}} \frac{L_I}{P-1}. \tag{4.15}$$

Similarly, a child subaperture center $n^{(s)}_{\lfloor i/2^s \rfloor}$ has coordinates $(x_i, y^{(s)}_{\lfloor i/2^s \rfloor}, z_i)$, where

$$y^{(s)}_{\lfloor i/2^s \rfloor} = y^{(s-1)}_{\lfloor i/2^{s-1} \rfloor} + (-1)^{\lfloor i/2^{s-1} \rfloor} \frac{N}{2^{S_N - s+2}} \frac{L_A}{N-1}. \tag{4.16}$$

Let $\Delta^{(s)}_k = y^{(s)}_{\lfloor k/2^{S_P - s} \rfloor} - y_k$ and $\Delta^{(s)}_i = y^{(s)}_{\lfloor i/2^s \rfloor} - y_i$. Using these relationships, the error $\varepsilon_1$ between the actual distance and the factorized distance from a pulse $n_i$ and a pixel $p_k$ can be written as

$$\begin{aligned}
\varepsilon_1 =& d[n_i, p_k] - \left\{ \sum_{s=1}^{S} \left[ d[n^{(s-1)}_{\lfloor i/2^s \rfloor}, p^{(s)}_{\lfloor k/2^{S-s} \rfloor}] - d[n^{(s)}_{\lfloor i/2^{s+1} \rfloor}, p^{(s)}_{\lfloor k/2^{S-s+1} \rfloor}] \right] + d[n^{(S)}_{\lfloor i/2^S \rfloor}, p_k] \right\} \\
=& \sqrt{R_0^2 + (y_i - y_k)^2} - \left\{ \sum_{s=1}^{S} \left[ \sqrt{R_0^2 + (y_i + \Delta^{(s-1)}_i - y_k - \Delta^{(s)}_k)^2} - \sqrt{R_0^2 + (y_i + \Delta^{(s)}_i - y_k - \Delta^{(s)}_k)^2} \right] \right. \\
& \left. + \sqrt{R_0^2 + (y_i + \Delta^{S}_i - y_k)^2} \right\}.
\end{aligned} \tag{4.17}$$

We can approximate $\varepsilon_1$ by $\tilde{\varepsilon_1}$, where $\tilde{\varepsilon_1}$ is the Taylor series approximation given by

$$\begin{aligned}
\tilde{\varepsilon_1} =& R_0 + \frac{1}{2R_0}(y_i - y_k)^2 - \left\{ \sum_{s=1}^{S} \left[ R_0 + \frac{1}{2R_0}(y_i + \Delta^{(s-1)}_i - y_k - \Delta^{(s)}_k)^2 - R_0 \right. \right. \\
& \left. \left. - \frac{1}{2R_0}(y_i + \Delta^{(s)}_i - y_k - \Delta^{(s)}_k)^2 \right] + R_0 + \frac{1}{2R_0}(y_i + \Delta^{(S)}_i - y_k)^2 \right\} \\
=& \frac{1}{2R_0} \left\{ (y_i - y_k)^2 - \sum_{s=1}^{S} \left[ (y_i + \Delta^{(s-1)}_i - y_k - \Delta^{(s)}_k)^2 - (y_i + \Delta^{(s)}_i - y_k - \Delta^{(s)}_k)^2 \right] - (y_i + \Delta^{(S)}_i - y_k)^2 \right\}.
\end{aligned} \tag{4.18}$$

By canceling and rearranging terms, this equation can be further simplified as

$$\begin{aligned}
\tilde{\varepsilon_1} =& \frac{1}{2R_0} \left[ 2 \sum_{s=2}^{S} \left( \Delta^{(s-1)}_i \Delta^{(s)}_k - \Delta^{(s-1)}_i \Delta^{(s-1)}_k \right) - 2\Delta^{(S)}_i \Delta^{(S)}_k \right] \\
=& \frac{1}{2R_0} \left[ 2 \sum_{s=2}^{S} \left( \Delta^{(s-1)}_i [\Delta^{(s)}_k - \Delta^{(s-1)}_k] \right) - 2\Delta^{(S)}_i \Delta^{(S)}_k \right].
\end{aligned} \tag{4.19}$$

We note that

$$\begin{aligned}\Delta_k^{(s)} - \Delta_k^{(s-1)} =& (y^{(s)}_{\lfloor k/2^{S_P-s}\rfloor} - y_k) - (y^{(s-1)}_{\lfloor k/2^{S_P-s+1}\rfloor} - y_k)\\ =& y^{(s)}_{\lfloor k/2^{S_P-s}\rfloor} - y^{(s-1)}_{\lfloor k/2^{S_P-s+1}\rfloor}\\ =& (-1)^{\lfloor k/2^{S_P-s}\rfloor}\frac{P}{2^{s+1}}\frac{L_I}{P-1}\\ \leq& \frac{P}{2^{s+1}}\frac{L_I}{P-1}. \end{aligned} \tag{4.20}$$

Thus,

$$\tilde{\varepsilon}_1 \leq \frac{1}{2R_0}\left[2\sum_{s=2}^{S}\left(\Delta_i^{(s-1)}\frac{P}{2^{s+1}}\frac{L_I}{P-1}\right) - 2\Delta_i^{(S)}\Delta_k^{(S)}\right]. \tag{4.21}$$

Using the triangle inequality, we can further bound $\tilde{\varepsilon}_1$ by

$$\tilde{\varepsilon}_1 \leq \frac{1}{2R_0}\left[2\sum_{s=2}^{S}\left(\left|\Delta_i^{(s-1)}\frac{P}{2^{s+1}}\frac{L_I}{P-1}\right|\right) + 2|\Delta_i^{(S)}\Delta_k^{(S)}|\right]. \tag{4.22}$$

Since for any given pulse $n_i$,

$$\Delta_i^{(s)} \leq \frac{N}{2^{S_N-s+1}}\frac{L_A}{N-1} \approx \frac{L_A}{2^{S_N-s+1}}$$

and for any given pixel $p_k$,

$$\Delta_k^{(s)} \leq \frac{P}{2^s}\frac{L_I}{P-1} \approx \frac{L_I}{2^s}$$

we can further simplify the bound in Eq. (4.22) as

$$\begin{aligned}\tilde{\varepsilon}_1 \leq& \frac{1}{R_0}\left[\sum_{s=2}^{S}\left(\frac{L_A}{2^{S_N-s+2}}\frac{L_I}{2^{s+1}}\right) + \frac{L_A}{2^{S_N-s+1}}\frac{L_I}{2^S}\right]\\ =& \frac{1}{R_0}\left[\sum_{s=2}^{S}\left(\frac{L_I L_A}{2^{S_N+3}}\right) + \frac{L_I L_A}{2^{S_N+2}}\right]\\ =& \frac{1}{R_0}\left[(S-1)\frac{L_I L_A}{2^{S_N+3}} + \frac{2L_I L_A}{2^{S_N+3}}\right]\\ =& \frac{1}{8R_0}\frac{(S+1)L_I L_A}{2^{S_N}}. \end{aligned} \tag{4.23}$$

To obtain $\varepsilon_2$, we follow a similar procedure. We note that

$$\begin{aligned}\varepsilon_2 =& d[n_i,p_k]^2 - \tilde{d}[n_i,p_k]^2 \\ =& (d[n_i,p_k] - \tilde{d}[n_i,p_k])(d[n_i,p_k] + \tilde{d}[n_i,p_k]) \\ =& \varepsilon_1(d[n_i,p_k] + \tilde{d}[n_i,p_k]) \\ \approx& \varepsilon_1(2R_0) \end{aligned} \tag{4.24}$$

to find

$$\varepsilon_2 \leq \frac{1}{4}\frac{(S+1)L_I L_A}{2^{S_N}}. \tag{4.25}$$

Hence,

$$\begin{aligned}\varepsilon \leq & \left|\frac{4\pi}{\lambda}\varepsilon_1\right| + \left|\frac{4\pi k_r}{c^2}\varepsilon_2\right| \\ \approx & \frac{4\pi}{\lambda}\frac{1}{8R_0}\frac{(S+1)L_I L_A}{2^{S_N}} + \frac{4\pi|k_r|}{c_0^2}\frac{1}{4}\frac{(S+1)L_I L_A}{2^{S_N}}. \end{aligned} \tag{4.26}$$

From this equation, we see that the distance error can be reduced by decreasing the length of the image to be reconstructed. Similarly, by initially dividing a column into several subimages rather than performing factorized backprojection for the entire column, the error is reduced because each subimage is shorter. However, this requires more computation.

### 4.1.3 Range Bin Error for Pulsed and LFM-CW SAR

Recall that in the creation of the data set $R(d[n_i^{(s)}, p_k^{(s)}])$, we make the approximation

$$R(d[n_j^{(s-1)}, p_k^{(s)}]) \approx R(d[n_j^{(s-1)}, p_{\lfloor k/2 \rfloor}^{(s-1)}]). \tag{4.27}$$

That is, we assume that the radar data associated with a given subaperture and subimage is the same as the radar data associated with the subaperture and the parent subimage. Since data is considered constant over a range bin, this assumption is true so long as both subimages lie within the same range bin. However, if both subimages do not lie in the same range bin, then the data corresponding to the child subimage is assigned to wrong range bin, causing errors. This range bin error is caused when the antenna has either an extremely narrow beamwidth or a moderately wide beamwidth.

When the antenna has a narrow beamwidth, a given pulse may contain the center of the column in its beamwidth but not the edge of the column (see Fig. 4.2). However, since the data assigned to the center of the column is also assigned to the edge of the column, factorized back-projection introduces spurious data to the edge of the column. The window discussed in Section 3.3 minimizes these errors.

When the antenna has a moderately wide beamwidth such that the entire column is contained within the beamwidth, the edge of the column and the center of the column may not lie within the same range bin depending on the curvature of the footprint (see Fig. 4.3) Thus, the assumption that data at the center of a column is the same data at the edge of a column is incorrect. Additionally, the assumption that the center and edge of the column are roughly the same distance from the pulse can be incorrect, causing further errors.

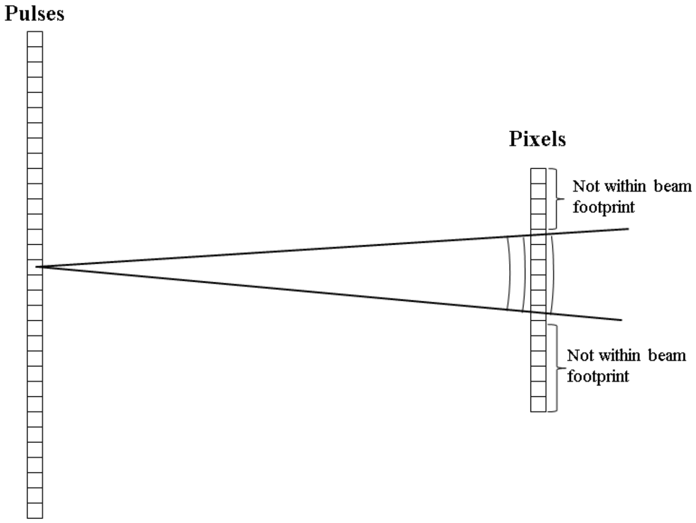


**Figure 4.2:** Illustration of antenna array and column where single antenna has narrow beamwidth. Note that the indicated antenna footprint does not cover pixels on the edges of the column. Hence, assigning data corresponding to the central pixel to pixels on the edge causes errors.

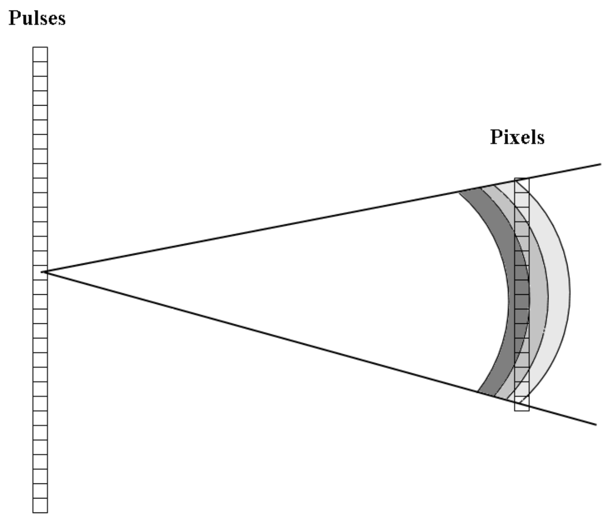


**Figure 4.3:** Illustration of antenna array and column where single antenna has moderately wide beamwidth. Note that the entire column is covered by the antenna footprint, but three different range bins from a single pulse (indicated in various shades of gray) correspond to the column. Hence, assigning data corresponding to the central pixel to pixels on the edge causes errors.

In either case, if it does not appear that the range bins corresponding to each pulse and the column align, one solution is to partition columns into subimages, referred to for the remainder of the section as subcolumns. Each subcolumn has the property that the center of the subcolumn and the edges of the subcolumn correspond to the same range bin for each pulse which contains the center of the subcolumn in its footprint.

A potential algorithm is as follows. Begin with the first pixel $p$ in the column. Consider the footprint of the pulse position $n$ which is directly perpendicular to the $p$ (that is, at the range of closest approach to the $p$). Determine which pixels in the column fall in the same range bin of $n$ as the first pixel. The first pixel $p_1$ that does not fall into the correct range bin marks the beginning of the next subcolumn (see Fig. 4.4). Consider the footprint of the pulse position $n_1$ at the range of closest approach from $p_1$. Determine which pixels below $p_1$ lie in the same range bin of $n_1$ as $p_1$. The pixel $p_2$ that does not fall into the correct range bin marks the beginning of the next subcolumn, and the process continues until all pixels are assigned to some subcolumn. As an added precaution, the algorithm can then be performed from bottom to top, further partitioning subcolumns as deemed necessary.

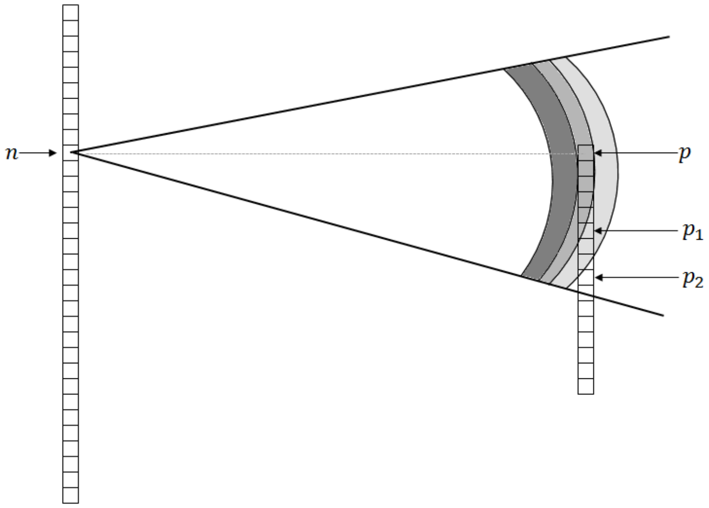


**Figure 4.4:** Illustration of partitioning of subcolumns

## 4.2 Performance Evaluation and Example Imagery

In this section we display images formed by factorized and windowed factorized backprojection for pulsed and LFM-CW SAR and compare them to images formed with direct backprojection.

### 4.2.1 Results for an Ideal Track for Pulsed SAR

We first assume that the flight track is ideal, that is, straight and level, with uniform spacing. Figure 4.5 shows the impulse response (IPR) of a point target created with noise-free simulated data acquired from an L-band pulsed SAR (parameters given in Table B.1) which was reconstructed with direct backprojection. Figure 4.6 shows the IPR of the same point target reconstructed with factorized backprojection. Note that both images have notable azimuth sidelobes.

When a window is added to the direct backprojection image, the image quality improves, although the resolution is slightly degraded as evidenced by the wider target main lobe (see Fig. 4.7). When the window is applied to the factorized backprojection image, the image has reduced sidelobes and similar resolution loss. Figure 4.8 shows the windowed factorized backprojection image where each pixel has been normalized by the area of the effective window on the pixel. Note that the width of the main lobe in the azimuth direction for both windowed images is slightly wider,

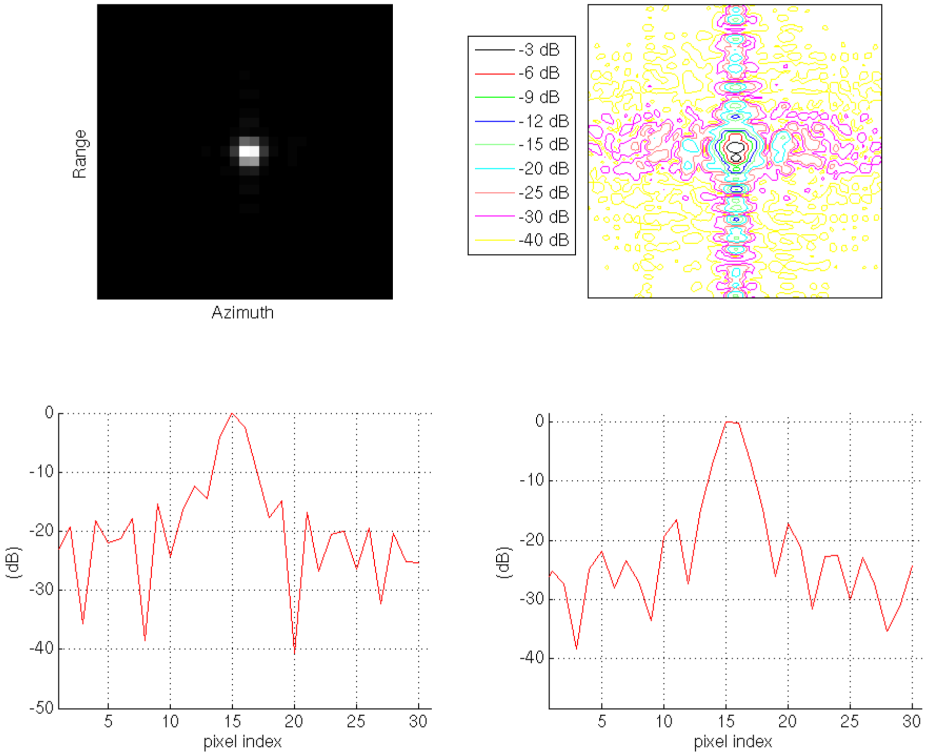


**Figure 4.5:** IPR of point target generated from simulated SAR data collected from an ideal track with parameters given in Table B.1 using direct backprojection. Upper left: power image (linear scale); upper right: contour plot; lower left: range slice through peak; lower right: azimuth slice through peak.

resulting in slightly coarser resolution. However, the sidelobes in the azimuth direction have been reduced considerably from Fig. 4.6 to Fig. 4.8.

### 4.2.2 Results on a Non-Ideal Track for Pulsed SAR

If the flight track is non-ideal, then factorized backprojection becomes less accurate because the range bins corresponding to a child subaperture may differ from the range bins corresponding to a parent subaperture (see [1] for a more complete analysis). To illustrate this, we simulate a non-ideal flight track with a sinusoidal movement at an amplitude of 1 m (which spans more than one range bin). In Figs. 4.9, 4.10, 4.11, and 4.12, the IPR is shown when the flight track is non-ideal for an image reconstructed with direct, windowed direct, factorized, and windowed factorized backprojection, respectively. As shown in Fig. 4.11, factorized backprojection alone

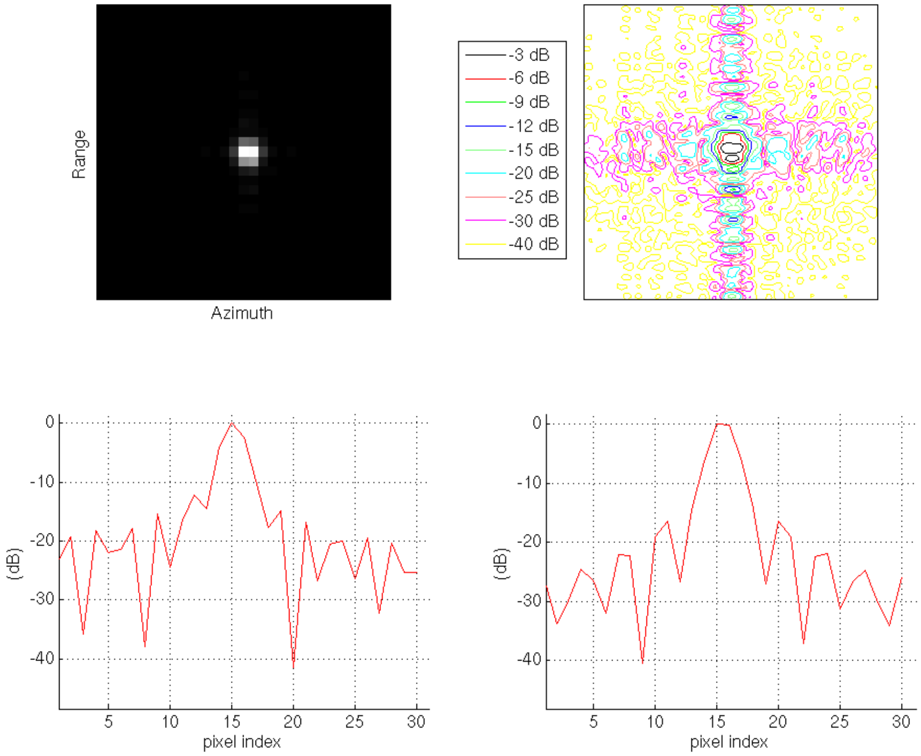


**Figure 4.6:** IPR of point target generated from simulated SAR data collected on an ideal track with parameters given in Table B.1 using factorized backprojection. See caption for Fig. 4.5.

can be unsuitable for dealing with non-ideal tracks. However, windowed factorized backprojection improves the image quality to an extent.

### 4.2.3 Results with Real Data for Pulsed SAR

Figure 4.13 shows various images generated from real pulsed SAR data of a uniform scene with a trihedral corner reflector (parameters given in Table B.2). There are 4096 aperture positions and an image grid of $1024 \times 1024$ pixels, with each pixel 0.5m by 0.3m. Figure 4.14(a) shows the results of direct backprojection. Figure 4.14(c) shows the same image reconstructed using factorized backprojection. Note that the corner reflector appears more smeared in the factorized backprojection image than in the direct backprojection image, mostly due to non-ideal motion. Figure 4.14(e) shows the image reconstructed with windowed factorized backprojection. Note that the sidelobes have been compressed slightly and the corner reflector appears less smeared than it

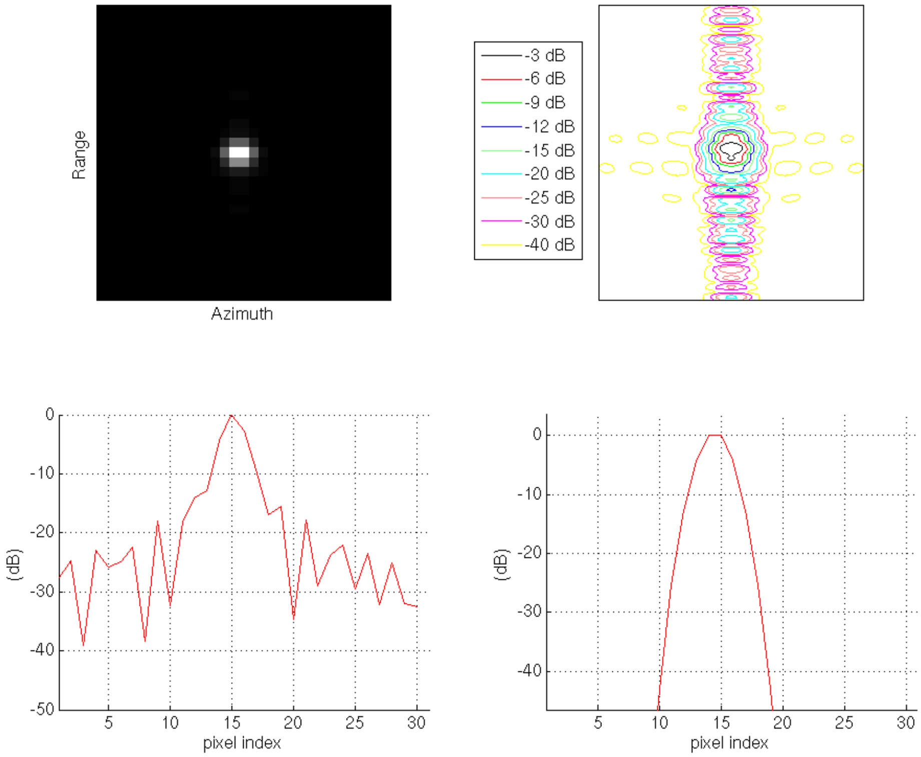


**Figure 4.7:** IPR of point target generated from simulated SAR data collected on an ideal track with parameters given in Table B.1 using direct backprojection with a Gaussian window. See caption for Fig. 4.5.

did in Fig. 4.14(c), although the overall resolution is somewhat coarser. The IPR of each image is also shown.

### 4.2.4 Results for Simulated Data for LFM-CW SAR

Figure 4.15(a) displays the IPR response of a point target created with noise-free simulated data acquired from an LFM-CW SAR (parameters given in Table B.3) which was reconstructed with direct backprojection. Figure 4.15(b) shows the IPR of the same point target reconstructed with windowed factorized backprojection. Although the range and azimuth slices and power image look similar, the contour plots differ in shape. This is due to the quantized nature of factorized backprojection. Since adjacent pixels use similar range data, the dropoff is more discrete than continuous in nature.

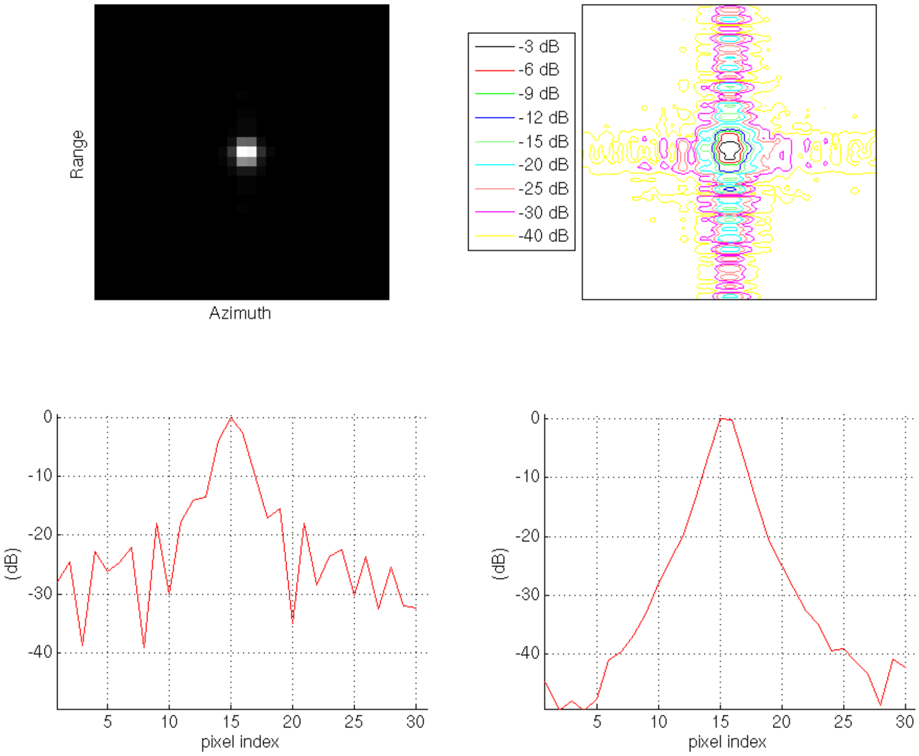


**Figure 4.8:** IPR of point target generated from simulated SAR data collected on an ideal track with parameters given in Table B.1 using factorized backprojection with a factorized window. See caption for Fig. 4.5.

### 4.2.5 Results for Real Data for LFM-CW SAR

Figure 4.15 shows images generated from real SAR data collected by the BYU/Artemis microASAR system as flown as part of the Characterization of Arctic Sea Ice Experiment 2009 (CASIE-09) [18]. The parameters are given in Table B.4. Figure 4.16(a) shows the results of direct backprojection. Figure 4.16(b) shows the same image reconstructed using windowed factorized backprojection with 11 initial subimages per column. Note that the two images are similar in that the major features are visible in both. However, the image reconstructed with windowed factorized backprojection is somewhat degraded compared to the image constructed with direct backprojection in several ways. Some details have been lost in the image reconstructed by windowed factorized backprojection, and there is some aliasing in the windowed factorized backprojection

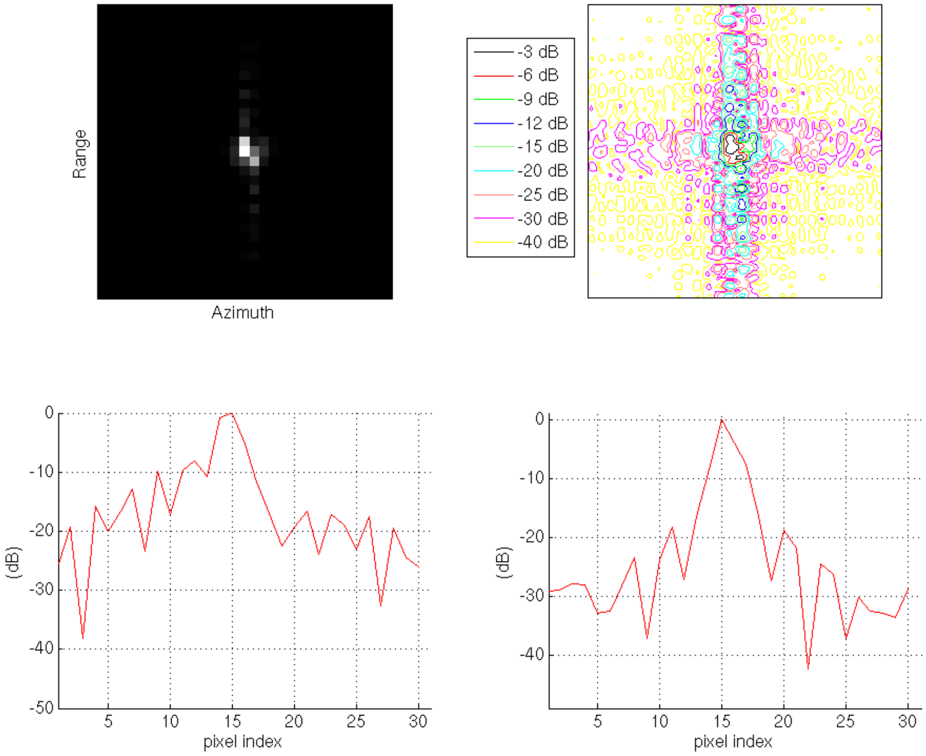


**Figure 4.9:** IPR of point target generated from simulated SAR data collected on a non-ideal track with parameters given in Table B.1 using direct backprojection. See caption for Fig. 4.5.

image. The image degradation is due to the non-ideal motion of the radar as well as the implicit phase error of factorized backprojection.

Although there was no attempt at optimizing the code, windowed factorized backprojection offered a savings of approximately a factor of 5 in computational time, i.e. 30 minutes compared to 146 minutes. Though this is not as high as the theoretical bound, it does demonstrate the improved computational complexity of factorized backprojection algorithms even using code which has not been optimized.

## 4.3 Conclusion

This chapter provided a performance analysis of factorized backprojection. The phase error was discussed and an upper bound on the phase error was given. The effects of range bin error were discussed, and an algorithm was provided to mitigate its effects. Example imagery of SAR

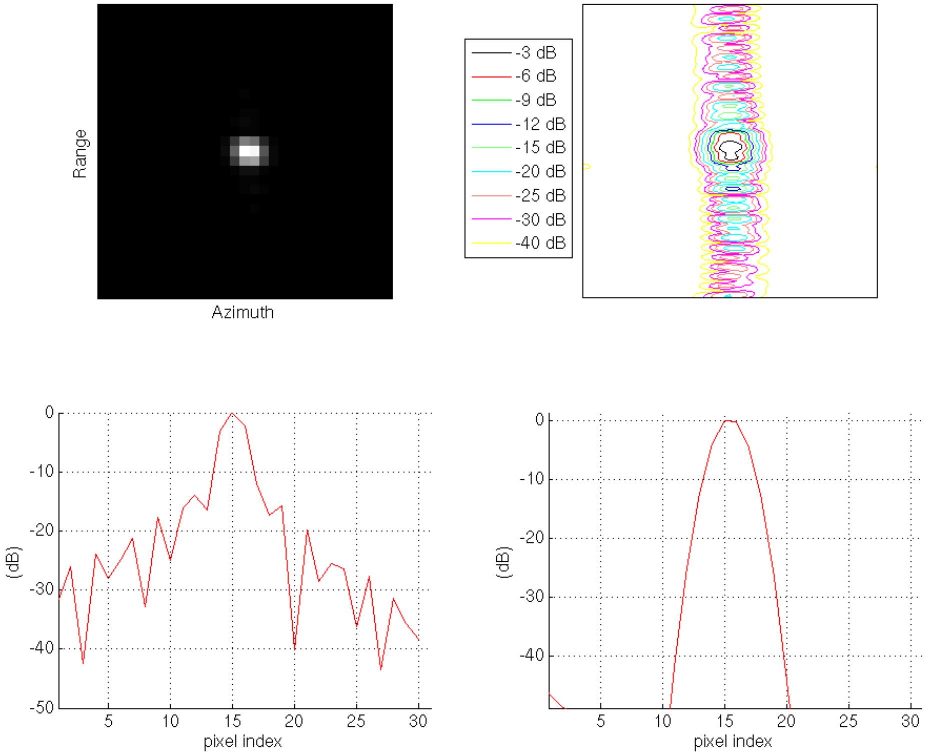


**Figure 4.10:** IPR of point target generated from simulated SAR data collected on a non-ideal track described in the text with parameters given in Table B.1 using direct backprojection with a Gaussian window. See caption for Fig. 4.5.

data reconstructed windowed factorized backprojection was displayed. Based on the error analysis and example imagery, it is shown that windowed factorized backprojection approaches the quality of factorized backprojection although there is inherent error in the algorithm.

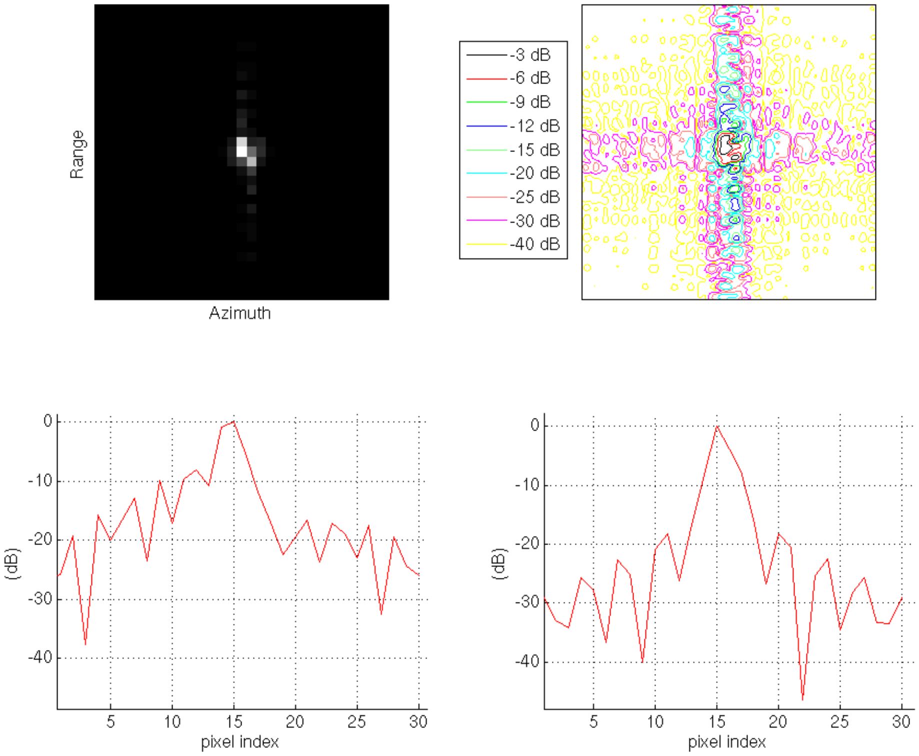


**Figure 4.11:** IPR of point target generated from simulated SAR data collected on a non-ideal track described in the text with parameters given in Table B.1 using factorized backprojection. See caption for Fig. 4.5.

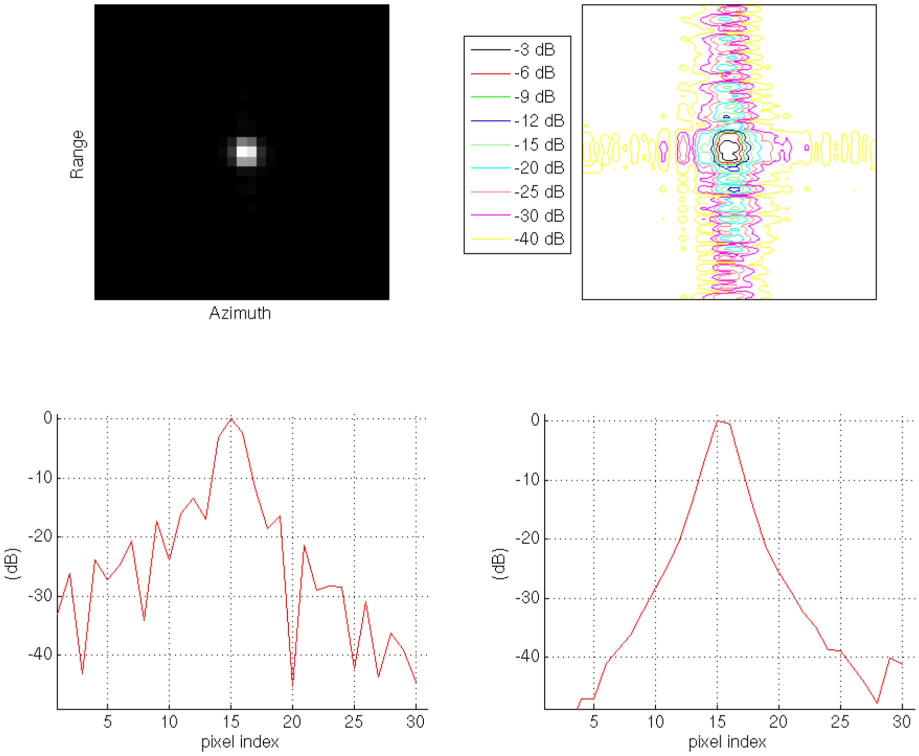
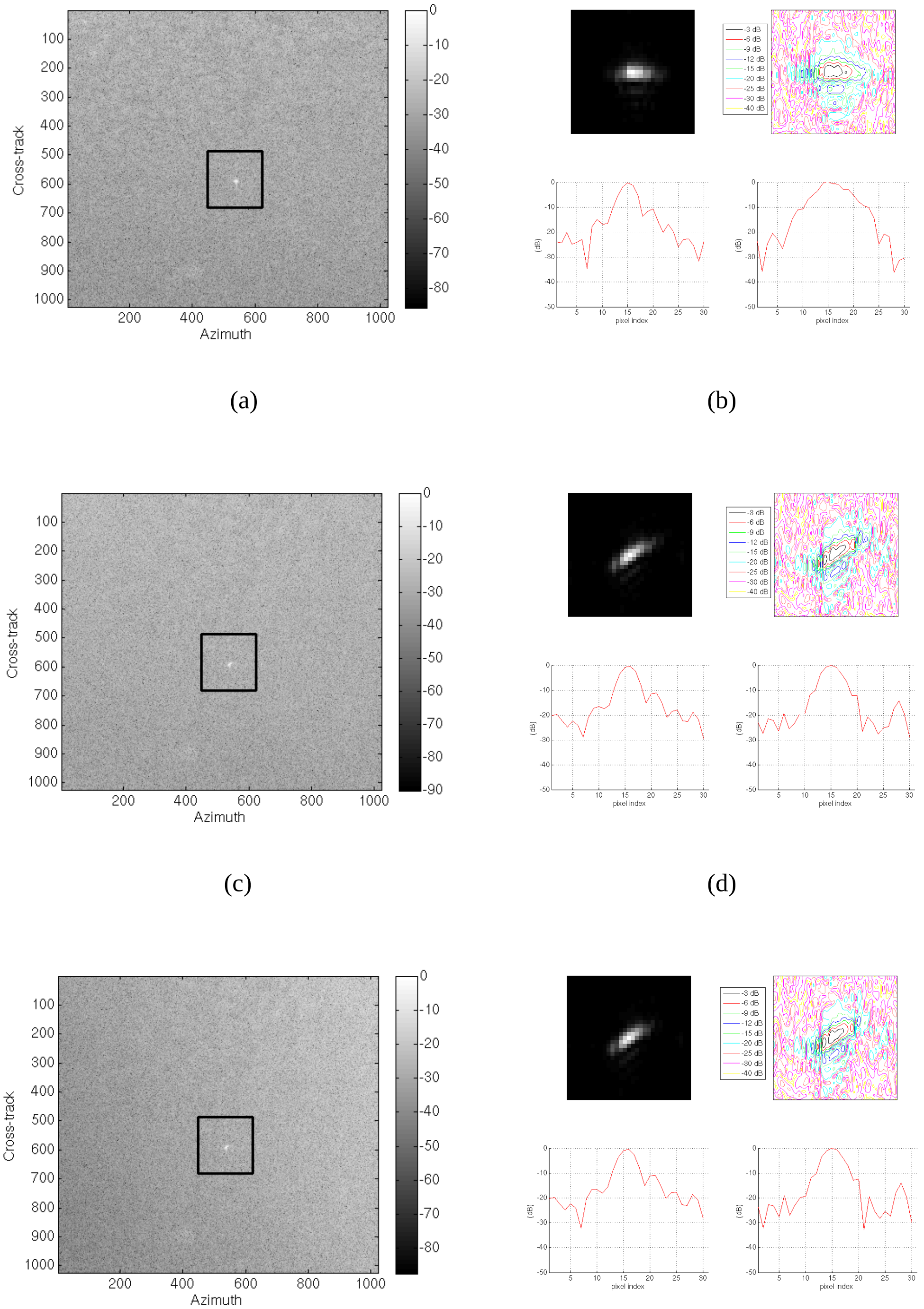


**Figure 4.12:** IPR of point target generated from simulated SAR data collected on a non-ideal track described in the text with parameters given in Table B.1 using factorized backprojection with a factorized window. See caption for Fig. 4.5.



(a)

(b)

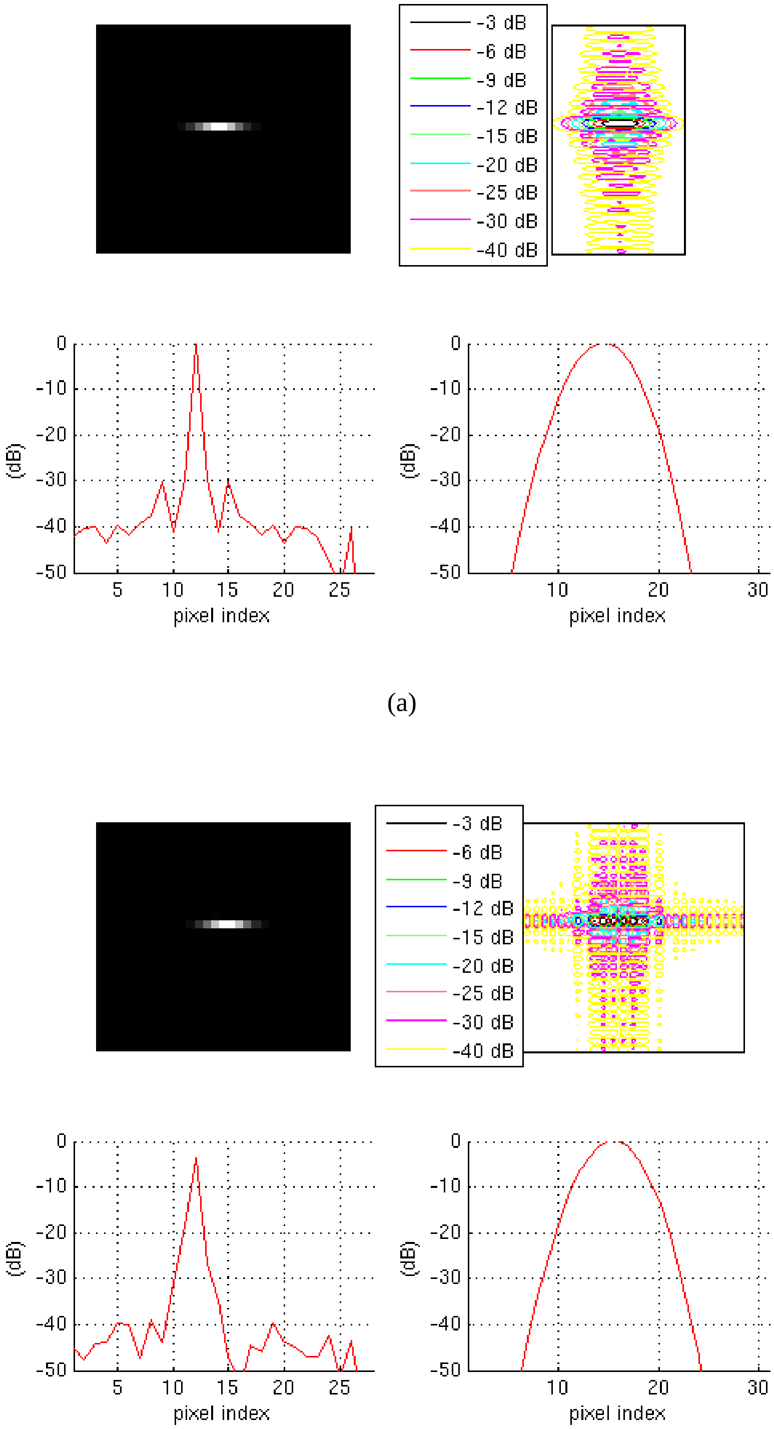
(c)

(d)

(e)

(f)

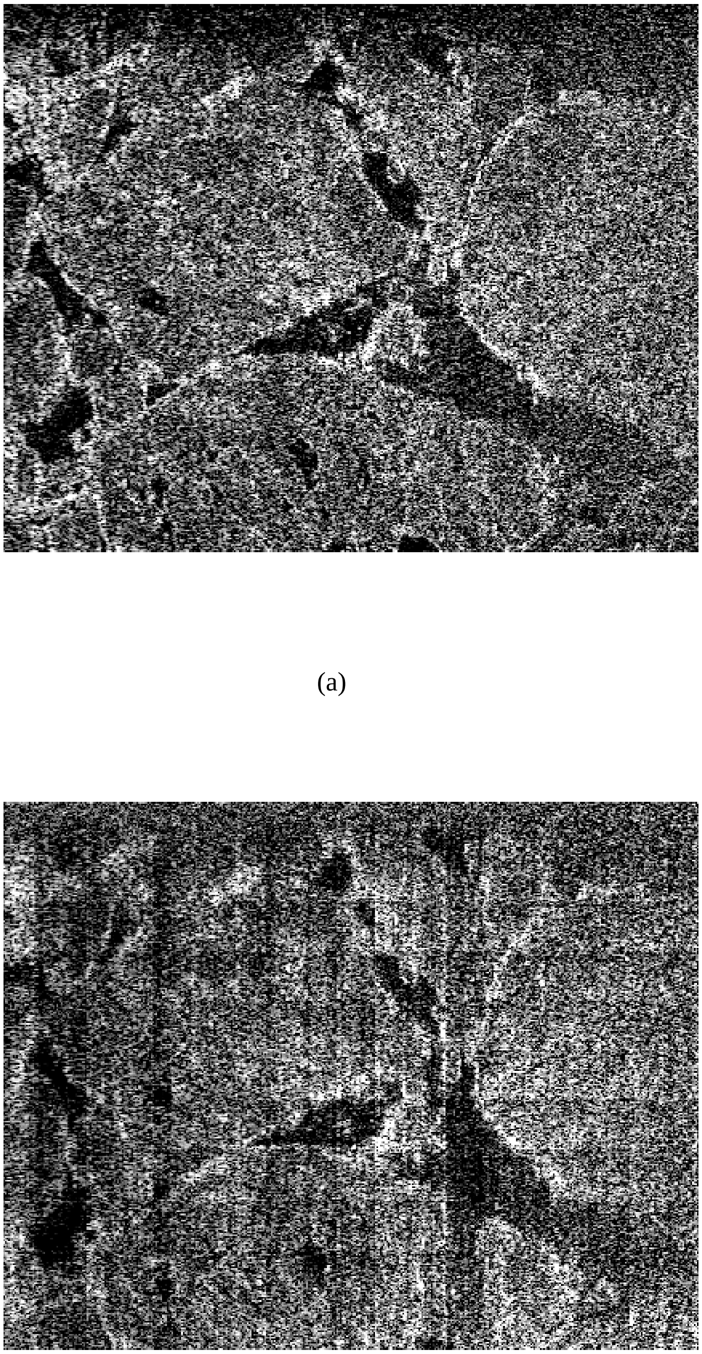
**Figure 4.13:** Images generated from real SAR data of uniform scene with a trihedral corner reflector. Parameters given in Table B.2. (a): direct backprojection (in dB); (b): IPR of area outlined by black rectangle in direct backprojection image; (c): factorized backprojection (in dB); (d): IPR of area outlined by black rectangle in factorized backprojection image; (e): windowed factorized backprojection; (f): IPR for area outlined by black rectangle in windowed factorized backprojection image. See caption of Fig. 4.5 for labels of IPR. Note that the reconstructed point target is smeared due to the real (and hence non-ideal) motion of the SAR.



(a)

(b)

**Figure 4.14:** IPR of a point target generated from simulated SAR data with parameters given in Table B.3. (a) IPR for direct backprojection image (upper left: power, upper right: contour plot, lower left: azimuth slice, lower right: range slice); (b) IPR for windowed factorized backprojection



(a)

(b)

**Figure 4.15:** Images generated from real SAR data collected as part of CASIE-09. Parameters given in Table B.4. (a) LFM-CW direct backprojection (in dB); (b) LFM-CW windowed factorized backprojection (in dB).

# Chapter 5

# Variations on Factorized Backprojection

This chapter discusses variations on factorized backprojection for stripmap SAR. These algorithms use similar principles as the factorized backprojection algorithm but have different implementations.

The first algorithm discussed is called *Correct Phase Factorized Backprojection*. If the projected phase error given in Eq. (4.10) is expected to be too high to produce an acceptable image, then the algorithm presented in the following section may be appropriate.

The second algorithm discussed is a matrix formulation of factorized and windowed factorized backprojection. This matrix formulation provides a concise and consolidated view of factorized backprojection

## 5.1 Correct Phase Factorized Backprojection

Recall the approximation made on each step of factorized backprojection,

$$R(d[n_j^{(s-1)}, p_k^{(s)}]) \approx R(d[n_j^{(s-1)}, p_{\lfloor k/2 \rfloor}^{(s-1)}]). \tag{5.1}$$

Since this approximation is made on each step of the algorithm, we find that

$$R(d[n_j^{(s-1)}, p_k^{(s)}]) \approx R(d[n_j^{(0)}, p_0^{(0)}]). \tag{5.2}$$

That is, the range data associated with a given pulse $n_j^{(0)}$ and the central pixel $p_0^{(0)}$ is reused for all pixels $p_k$ in the column and the given pulse $n_j^{(0)}$. (The errors discussed with this assumption are discussed in Section 4.1.3.)

Thus, rather than performing factorized backprojection, an alternative is to calculate the correct distance to each pixel from each pulse but reuse the range data corresponding to a single pixel in the column. Let $p$ be a pixel in the column and $p_0$ be the central pixel in the column. Then backprojection equation for $p$ is

$$A(p) = \sum_n R(d[n, p_0]) \exp\left(j\phi(d[n, p])\right) \tag{5.3}$$

where $\phi(d[n, p])$ is the phase specific to either pulsed or LFM-CW SAR.

If an azimuth window is desired for some pixel $p$, a weighting $W(n, p)$ function can easily be implemented:

$$A(p) = \sum_n W(n, p) R(d[n, p_0]) \exp\left(j\phi(d[n, p])\right) \tag{5.4}$$

where $W(n, p)$ is a weighting function expressed in terms of the pulse number $n$ and specified pixel $p$. The direct window introduced in Section 3.3 can be used,

$$W(n, p) = \exp(-(n_y - p_y)^2/a) \tag{5.5}$$

where $n_y$ is the $y$-coordinate of $n$, $p_y$ is the $y$-coordinate of $p$, $a$ is some constant, and the azimuth direction is in $y$.

This formulation has some advantages over both direct and factorized backprojection. The memory requirements for this algorithm are less than those required for factorized backprojection since there are no intermediate data sets which must be stored, thus requiring memory for only the aperture matrix and image.

In addition to requiring less memory, this algorithm tends to create higher quality images than factorized backprojection. This is because factorized backprojection images tend to have degraded quality primarily due to the phase error discussed in Section 4.1. Since this new formulation uses the correct phases, its only errors are caused by assuming data is in a different range bin (see Section 4.1.3).

Despite these advantages, there are some drawbacks to this algorithm. The computational complexity is $O(N^3)$ since distances are calculated exactly. Although this may require fewer computations than direct backprojection since range data is only calculated once per pulse per column, the computational gains are minimal.

## 5.2 A Matrix Formulation of Factorized Backprojection

This section demonstrates how the factorized backprojection process can be expressed in terms of matrix multiplication. We first show the matrices for the first factorization step and then show how the same structure can be extended to subsequent steps. In this section, we denote $L$ as the number of pulses, $M$ as the number of full-resolution pixels in a column, and $S = \min\{\log_2 L, \log_2 M\}$ as the number of steps (not including a preliminary step $s = 0$).

On the first (non-preliminary) step of factorized backprojection, assuming that we start with one initial subimage per column, there are two low-resolution pixels per column. Recall that the distance from each child subaperture to each pixel is calculated. Then, the distance from each parent subaperture to each pixel is calculated. The intermediate data sets are constructed as

$$R(d[n_i^{(s)}, p_k^{(s)}]) = \sum_{n_j \ni n_i} R(d[n_j^{(s-1)}, p_k^{(s)}]) \exp(j4\pi/\lambda \Delta r_{j,k}) \tag{5.6}$$

where

$$\Delta r_{j,k} = d[n_j^{(s-1)}, p_k^{(s)}] - d[n_{\lfloor j/2 \rfloor}^{(s)}, p_k^{(s)}]. \tag{5.7}$$

Define $\Delta\phi_{j,k}$ as

$$\Delta\phi_{j,k} = \exp(j4\pi/\lambda \Delta r_{j,k}). \tag{5.8}$$

Note that when $j$ is not in a subscript it refers to $\sqrt{-1}$.

Since we use the approximation

$$R(d[n_j^{(s-1)}, p_k^{(s)}]) \approx R(d[n_j^{(s-1)}, p_{\lfloor k/2 \rfloor}^{(s-1)}]), \tag{5.9}$$

and since $\lfloor k/2 \rfloor = \lfloor (k+1)/2 \rfloor$ when $k$ is even, both $R(d[n_i^{(s)}, p_k^{(s)}])$ and $R(d[n_i^{(s)}, p_{k+1}^{(s)}])$ both depend on $R(d[n_j^{(s-1)}, p_{\lfloor k/2 \rfloor}^{(s-1)}])$ (where $k$ is even), and they only differ in $\Delta\phi_{j,k}$. Hence, the computations for the intermediate data sets $R(d[n_i^{(s)}, p_k^{(s)}])$ and $R(d[n_i^{(s)}, p_{k+1}^{(s)}])$ can be written as

$$\begin{bmatrix} R(d[n_i^{(s)}, p_k^{(s)}]) & R(d[n_i^{(s)}, p_{k+1}^{(s)}]) \end{bmatrix} = \begin{bmatrix} R(d[n_{2\tilde{i}}^{(s-1)}, p_{\tilde{k}}^{(s-1)}]) & R(d[n_{2\tilde{i}+1}^{(s-1)}, p_{\tilde{k}}^{(s-1)}]) \end{bmatrix} \begin{bmatrix} \Delta\phi_{2\tilde{i},k} & \Delta\phi_{2\tilde{i},k+1} \\ \Delta\phi_{2\tilde{i}+1,k} & \Delta\phi_{2\tilde{i}+1,k+1} \end{bmatrix} \tag{5.10}$$

where $\tilde{x} = \lfloor \frac{x}{2} \rfloor$.

All of the intermediate data sets in step $s = 1$ can be computed via the matrix multiplication

$$\begin{aligned}
&\Big[R(d[n_0^{(s)}, p_0^{(s)}]) \quad R(d[n_0^{(s)}, p_1^{(s)}]) \quad \cdots \quad R(d[n_{L/2-1}^{(s)}, p_0^{(s)}]) \quad R(d[n_{L/2-1}^{(s)}, p_1^{(s)}])\Big] = \\
&\Big[R(d[n_0^{(s-1)}, p_0^{(s-1)}]) \quad \cdots \quad R(d[n_{L-1}^{(s-1)}, p_0^{(s-1)}])\Big] \\
&\times \begin{bmatrix}
\Delta\phi_{0,0} & \Delta\phi_{0,1} & 0 & 0 & \cdots & 0 & 0 \\
\Delta\phi_{1,0} & \Delta\phi_{1,1} & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & \Delta\phi_{2,0} & \Delta\phi_{2,1} & \cdots & 0 & 0 \\
0 & 0 & \Delta\phi_{3,0} & \Delta\phi_{3,1} & \cdots & \vdots & \vdots \\
\vdots & \vdots & 0 & 0 & \ddots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & \Delta\phi_{L-2,0} & \Delta\phi_{L-2,1} \\
0 & 0 & 0 & \cdots & 0 & \Delta\phi_{L-1,0} & \Delta\phi_{L-1,1}
\end{bmatrix}
\end{aligned} \tag{5.11}$$

or in general,

$$Rp^{(s)} = R^{(s-1)}E^{(s)} \tag{5.12}$$

where

$$Rp^{(s)} = \Big[R(d[n_0^{(s)}, p_0^{(s)}]) \cdots R(d[n_0^{(s)}, p_{2^s-1}^{(s)}]) \cdots R(d[n_{2^{S-s}-1}^{(s)}, p_0^{(s)}]) \cdots R(d[n_{2^{S-s}-1}^{(s)}, p_{2^s-1}^{(s)}])\Big], \tag{5.13}$$

$$R^{(s-1)} = \Big[R(d[n_0^{(s-1)}, p_0^{(s-1)}]) \cdots R(d[n_{D-1}^{(s-1)}, p_0^{(s-1)}]) \cdots R(d[n_0^{(s-1)}, p_{D-1}^{(s-1)}]) \cdots R(d[n_{D-1}^{(s-1)}, p_{D-1}^{(s-1)}])\Big], \tag{5.14}$$

and

$$E^{(s)} = \begin{bmatrix}
\Delta\phi_{0,0} & \Delta\phi_{0,1} & 0 & \cdots & 0 \\
\Delta\phi_{1,0} & \Delta\phi_{1,1} & 0 & \cdots & 0 \\
0 & 0 & \ddots & 0 & 0 \\
0 & \cdots & 0 & \Delta\phi_{D-2,2^s-2} & \Delta\phi_{D-2,2^s-1} \\
0 & \cdots & 0 & \Delta\phi_{D-1,2^s-2} & \Delta\phi_{D-1,2^s-1}
\end{bmatrix} \tag{5.15}$$

with $D = 2^{S-(s-1)}$. Note that for each step $s$, $Rp^{(s)}$ has dimensions $1 \times LM/2^S$, $R^{(s-1)}$ has dimensions $1 \times LM/2^S$, and $E^{(s)}$ has dimensions $LM/2^S \times LM/2^S$.

To use this matrix formulation to compute all of the intermediate data sets in step $s+1$ in a similar fashion, $Rp^{(s)}$ must be permuted. This is because $Rp^{(s)}$ lists the intermediate data sets in terms of increasing pixel index and then increasing pulse index, while $R^{(s-1)}$ lists the intermediate data sets in terms of increasing subaperture index and then increasing subimage index. In order to reorder $Rp^{(s)}$ to obtain $R^{(s)}$, the permutation scheme $P^{(s)}$ is used, where given an index $n$,

$$P^{(s)}(n) = A \cdot n - (AB-1) \left\lfloor \frac{n}{B} \right\rfloor \tag{5.16}$$

where $A = M/2^{S-s}$ corresponds to the number of subimages in the step and $B = L/2^s$ corresponds to the number of subapertures in the step. Using this permutation scheme, $R^{(s)}$ is obtained by

$$R^{(s)} = Rp^{(s)} \mathscr{P}^{(s)} \tag{5.17}$$

where $\mathscr{P}^{(s)}$ is the permutation matrix whose rows $i$ are reordered by $P^{(s)}(i)$.

On the next step, $Rp^{(s+1)}$ can be obtained with the equation

$$\begin{aligned} Rp^{(s+1)} &= R^{(s)} E^{(s+1)} \\ &= Rp^{(s)} P^{(s)} E^{(s+1)} \\ &= R^{(s-1)} E^{(s)} P^{(s)} E^{(s+1)}. \end{aligned} \tag{5.18}$$

Note that each $R^{(s)}$ depends on $R^{(s-1)}$, which means that each $R^{(s)}$ depends on $R^{(0)}$ where

$$R^{(0)} = \left[ R(d[n_0^{(0)}, p_0^{(s)}]) \quad R(d[n_1^{0)}, p_0^{(s)}]) \quad \cdots \quad R(d[n_{L-1}^{0}, p_0^{(s)}]) \right]. \tag{5.19}$$

On the final step $S$,

$$Rp^{(S)} = R^{(0)} \left( \Pi_{i=1}^{S-1} E^{(i)} P^{(i)} \right) E^{(S)}. \tag{5.20}$$

The vector of backprojection pixels corresponding to this column is computed as

$$A = Rp^{(S)} \Phi^{(S)} \tag{5.21}$$

where

$$\Phi^{(S)} = \Big[\exp(j4\pi/\lambda d[n_0^{(S)}, p_0^{(S)}]) \quad \exp(j4\pi/\lambda d[n_0^{(S)}, p_1^{(S)}]) \quad \cdots \quad \exp(j4\pi/\lambda d[n_0^{(S)}, p_{M-1}^{(S)}])\Big]^T . \tag{5.22}$$

### 5.2.1 Incorporation of an Azimuth Window

To incorporate the azimuth window discussed in Section 3.3 into the matrix formulation, recall that a window can be implemented using Eqs. (3.23) and (3.24) with

$$R'(d[n_i^{(s)}, p_k^{(s)}]) = \sum_{n_j \ni n_i} W(n_j^{(s-1)}, p_k) R'(d[n_j^{(s-1)}, p_k^{(s)}]) \Delta\phi_{j,k} \tag{5.23}$$

where

$$W(n_j, p_k) = \exp(-|n_{jy} - p_{ky}|/a). \tag{5.24}$$

To include the $W(n_j, p_k)$ term into the matrix formulation, we incorporate it into the matrix $E$ to create a new matrix $E_W$ where

$$E_W^{(s)} = \begin{bmatrix} \Delta\phi_{0,0}W(n_0,p_0) & \Delta\phi_{0,1}W(n_0,p_1) & 0 & \cdots & 0 \\ \Delta\phi_{1,0}W(n_1,p_0) & \Delta\phi_{1,1}W(n_1,p_1) & 0 & \cdots & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & \cdots & \Delta\phi_{D-2,2^s-2}W(n_{D-2},p_{2^s-2)} & \Delta\phi_{D-2,2^s-1}W(n_{D-2},p_{2^s-1}) \\ 0 & 0 & \cdots & \Delta\phi_{D-1,2^s-2}W(n_{D-1},p_{2^s-2}) & \Delta\phi_{D-1,2^s-1}W(n_{D-1},p_{2^s-1}) \end{bmatrix} . \tag{5.25}$$

Thus, an azimuth window can be implemented into the matrix formulation with little added computation.

## 5.3 Conclusion

This chapter discusses alternatives to the factorized backprojection algorithm introduced in Chapter 3. Correct phase factorized backprojection uses the correct phase but assumes one range bin per column similar to factorized backprojection. Although it achieves little computational gain compared to direct backprojection, it offers additional insight behind the principles which allow factorized backprojection to work.

This chapter also discusses a matrix formulation of factorized backprojection. The formulation uses a permutation of the data from step to step to perform the factorization. I also demonstrate how a window can be implemented as part of the matrix formulation.

# Chapter 6

# Conclusion

This thesis contributes to the theory of synthetic aperture radar image processing by introducing a new formulation of factorized backprojection for stripmap SAR. This formulation is easily parallelizable and allows for the easy implementation of a Gaussian azimuth window.

In stripmap SAR, an antenna with a wide beamwidth is moved along an array to generate high-resolution images. These images can be reconstructed using backprojection, a time-domain algorithm. Although backprojection is an exact algorithm, it can be computationally expensive. Unlike backprojection, factorized backprojection takes advantage of the redundancy of the SAR data caused by using an antenna with a wide beamwidth to achieve a more computationally efficient algorithm.

This thesis explains how to implement factorized backprojection for both pulsed and LFM-CW SAR. Then, it is shown how to implement an azimuth window with shape similar to a Gaussian window. The computational and memory requirements are discussed, and it is shown that factorized backprojection achieves $N/\log N$ improvement over backprojection with only slightly higher memory requirements.

There are several assumptions that factorized backprojection operates on which can cause loss of image quality. An expression for the phase error has been developed, and it is shown that the phase error is dependent on the length of the image, the length of the antenna array, and the distance from the flight track to the region of interest. A discussion of errors due to range migration is then provided.

## 6.1 Contributions

The contributions of this thesis include the following:

- I have introduced a new factorized backprojection algorithm that can be used to reconstruct images from both pulsed and LFM-CW stripmap SAR data. This factorized backprojection algorithm has computational complexity $O(N^2 \log N)$.
- I have demonstrated how to implement an azimuth window into the algorithm to reduce sidelobes and prevent aliasing.
- I have presented an error analysis of factorized backprojection. In particular, I have provided upper bounds for the phase error.
- I have demonstrated how factorized backprojection can be performed via matrix multiplication.

## 6.2 Future Work

The work of this thesis can be applied and extended to a variety of research topics. A few examples are listed below.

1. The formulations of factorized backprojection were based on the assumption that the flight track was linear. Although it was demonstrated empirically that factorized backprojection is suitable for some nonlinear flight tracks (see Fig. 4.12), the research can be extended to include a more thorough analysis of nonlinear flight tracks in general.
2. A small squint angle has been assumed for this analysis. The research can be extended to a higher squint angle.
3. An algorithm for handling range migration was introduced in Section 4.1.3, but the research can be extended to find a more computationally efficient algorithm for handling range migration in factorized backprojection.
4. The azimuth window was chosen to be a Gaussian window. Future work could involve implementing windows other than the Gaussian window in factorized backprojection.

5. Some approximations were made for the derivation of factorized backprojection for LFM-CW factorized backprojection. The research can be extended to implement LFM-CW factorized backprojection without these approximations.

6. It has been shown that factorized backprojection can be implemented, but no implementation has necessarily been optimal. Future work could include optimizing the code which implements factorized backprojection.

# Bibliography

[1] L. Ulander, H. Hellsen, and G. Stenstrom, “Synthetic aperture radar processing using fast factorized back-projection,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 3, pp. 760–776, 2003. viii, 1, 18, 38, 47

[2] C. Elachi, *Spaceborne Radar Remote Sensing: Applications and Techniques*. New York: IEEE Press, 1988, pp. 72–77. 1, 5, 6, 7, 8

[3] I. Cumming and F. Wong, *Digital Processing of Synthetic Aperture Radar Data*. Norwood, MA: Artech House, 2005. 1, 5, 8, 12, 13, 14

[4] M. Rofheart and J. McCorkle, “An order $N^2 \log N$ backprojection algorithm for focusing wide angle wide bandwidth arbitrary-motion synthetic aperture radar,” in *SPIE Radar Sensor Technology Conference Proceedings*, 1996, pp. 25–36. 1, 16

[5] A. Hunter, M. Hayes, and P. Gough, “A comparison of fast factorised back-projection and wavenumber algorithms,” in *Fifth World Congress on Ultrasonics*, Paris, France, September 2003, pp. 527–530. 1, 15, 16

[6] S. Oh and J. McClellan, “Multiresolution imaging with quadtree backprojection,” in *Thirty-Fifth Asilomar Conference on Signals, Systems and Computers*, 2001, pp. 105–109. 1, 16

[7] H. Callow and R. Hansen, “Fast factorized back projection for synthetic aperture imaging and wide-beam motion compensation,” in *Proceedings of the Institute of Acoustics*, vol. 28, September 2006, pp. 191–200. 1, 18

[8] S. Basu and Y. Bresler, “$o(n^2 \log_2 n)$ filtered backprojection reconstruction algorithm for tomography,” *IEEE Transactions on Image Processing*, vol. 9, no. 10, pp. 1760–1773, 2000. 1

[9] G. A. Showman, *An Overview of Radar Imaging*. Raleigh, North Carolina: Scitech Publishing, 2010, ch. 21. 4, 5

[10] E. Zaugg and D. G. Long, “Generalized frequency-domain SAR processing,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 11, pp. 3761–3773, November 2009. 8, 14

[11] E. C. Zaugg, “Generalized Image Formation for Pulsed and LFM-CW Synthetic Aperture Radar,” Ph.D. dissertation, Brigham Young University, April 2010. 8, 11, 16

[12] E. Zaugg and D. G. Long, “Generalized LFM-CW SAR Processing,” in review. 9, 10, 11, 16, 27

[13] M. D. Desai and W. K. Jenkins, “Convolution backprojection image reconstruction for spotlight mode synthetic aperture radar,” *IEEE Transactions on Image Processing*, vol. 1, no. 4, pp. 505–516, October 1992. 15

[14] J. N. Ash, “An autofocus method for backprojection imagery in synthetic aperture radar,” *IEEE Geoscience and Remote Sensing Letters*, vol. 9, no. 1, pp. 104–108, January 2012. 15

[15] A. Ribalta, “Time-domain reconstruction algorithms for FMCW-SAR,” *IEEE Geoscience and Remote Sensing Letters*, vol. 8, no. 3, pp. 396–400, May 2011. 16

[16] B. Rigling and R. Moses, “Polar format algorithm for bistatic SAR,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, pp. 1147–1158, October 2004. 18

[17] J. Xiong, J. Chen, Y. Huang, J. Yang, Y. Fan, and Y. Pi, “Analysis and improvement of a fast backprojection algorithm for stripmap bistatic SAR imaging,” in *Synthetic Aperture Radar (EUSAR), 2008 7th European Conference on*, June 2008, pp. 1–4. 20

[18] D. G. Long and C. Stringham, “The Sample BYU CASIE-09 MicroASAR Dataset,” Brigham Young University, Tech. Rep., October 2011. [Online]. Available: http://www.mers.byu.edu/microASAR/CASIE_sample/casie_sample2.pdf 50

# Appendix A

## Pseudo Code

In this section we provide pseudo-code to illustrate how the factorized backprojection algorithm can be used to form a column of an image from SAR data. The pseudo-code, shown in Fig. A.2, is based on the flow diagram given in Fig. 3.2, repeated for convenience in Fig. A.1. We assume that the data has already been range compressed. In other words, the data is in the form required for direct backprojection. There are $N$ pulses and an $M \times M$ imaging grid. The input is `echoData`, an $N \times M$ `matrix`. The output is `image`, an $M \times 1$ image.

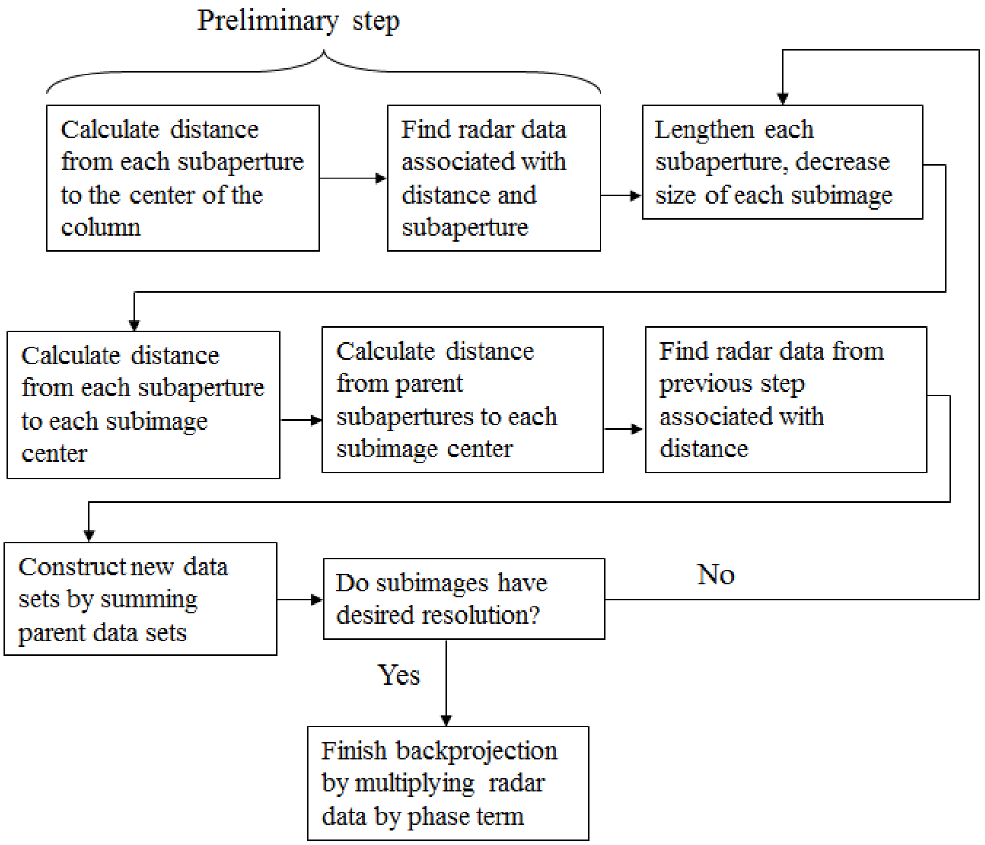

Preliminary step
Calculate distance from each subaperture to the center of the column
Find radar data associated with distance and subaperture
Lengthen each subaperture, decrease size of each subimage
Calculate distance from each subaperture to each subimage center
Calculate distance from parent subapertures to each subimage center
Find radar data from previous step associated with distance
Construct new data sets by summing parent data sets
Do subimages have desired resolution?
No
Yes
Finish backprojection by multiplying radar data by phase term

**Figure A.1:** Flow diagram for factorized backprojection.

```
/*Create the first data set using all the original echo data*/
for az = 1:numPulses
       pulse = rangeInterpolate(echo data associated with az$^{th}$ pulse);
       xCent = column center;
       azPosition = az$^{th}$ pulse in SAR array;

/*Compute the Euclidean distance between the
pulse position and the column center position*/
       distance = dist(azPosition,xCent);

/*convert slant range to A/D sample number */
       index = round(distance*dscale + delay)

/*form data set from echo data*/
       dataSet = pulse(index);
end

/*Form the image in a series of steps*/
numSTEPS = log2(M)

/*Run through all of the steps*/
for step = 1:numSteps

oldDataSet = dataSet;

oldSubimageCenters = subimageCenters;
/*Run through the new subaperture centers*/
       for az = subapertureCenters

/*Run through each subimage center*/
                  for xCent = subimageCenters
                            distance = dist(az,xCent);

/*Find the distance from the parent
subaperture center to the current subimage center*/
                            parentDistance1 = dist(azParent1,xCent)
                            parentDistance2 = dist(azParent2,xCent)
                            $\Delta r_1$= parentDistance1-distance
                            $\Delta r_2$= parentDistance2-distance

/*Find the index of the parent subimage in oldSubimageCenters
                            index = index(xCentParent1);

/*Determine the weight of the window applied on the data set*/
                            xCentParent1$_y$=the $y$-coordinate of xCentParent1
                            xCentParent2$_y$=the $y$-coordinate of xCentParent2
                            weight1=exp(-|az$_y$ $-$ xCentParent1$_y$|/a)
                            weight2=exp(-|az$_y$ $-$ xCentParent2$_y$|/a)
/*Create the new dataSet*/
                            dataSet(az,xCent)= oldDataSet(index)*weight1*$\exp(j4\pi/\lambda\Delta r_1)$
                                             + oldDataSet(index)*weight2*$\exp(j4\pi/\lambda\Delta r_2)$;

/*on the final step, backproject the dataSet to form the image*/
                            image(x)= image(x)+ dataSet(az,x)*$\exp(j4\pi/\lambda *$distance$)$;

                  end
       end
end
```

**Figure A.2:** Pseudo code

# Appendix B

## SAR Parameters

This section contains tables with the processing parameters for the simulated and real SAR data used in Section 4.2. The parameters for simulated pulsed SAR data are shown in Table B.1, real pulsed SAR data are shown in Table B.2, simulated LFM-CW SAR data are shown in Table B.3, and real LFM-CW SAR data are shown in Table B.4.

**Table B.1:** Summary of simulation processing parameters for Figs. 4.5–4.12.

| | |
|---|---|
| Chirp Bandwidth (MHz) | 500 |
| Center Frequency (GHz) | 1.75 |
| Azimuth Beamwidth | 30° |
| Pulse Repetition Frequency (Hz) | 1500 |
| Sample Rate (MHz) | 500 |

**Table B.2:** Summary of processing parameters for Fig. 4.13.

| | |
|---|---|
| Chirp Bandwidth (MHz) | 210 |
| Center Frequency (GHz) | 1.605 |
| Azimuth Beamwidth | 15° |
| Chirp Length ($\mu s$) | 5 |
| Sample Rate (MHz) | 500 |
| Range to Target (km) | 2.20 |
| Antenna Height (km) | 1.48 |

**Table B.3:** Summary of processing parameters for Fig. 4.14.

| | |
|---|---|
| Chirp Bandwidth (MHz) | 200 |
| Center Frequency (GHz) | 10 |
| Azimuth Beamwidth | 15° |
| Pulse Repetition Frequency (Hz) | 2000 |
| Sample Rate (MHz) | 500 |

**Table B.4:** Summary of processing parameters for Fig. 4.15.

| | |
|---|---|
| Chirp Bandwidth (MHz) | 170 |
| Center Frequency (GHz) | 5.429 |
| Azimuth Beamwidth | 10.9° |
| Sample Rate (MHz) | 24.49 |
| Chirp Rate (THz) | 1.59724 |
| Pulse Repetition Frequency (Hz) | 307 |