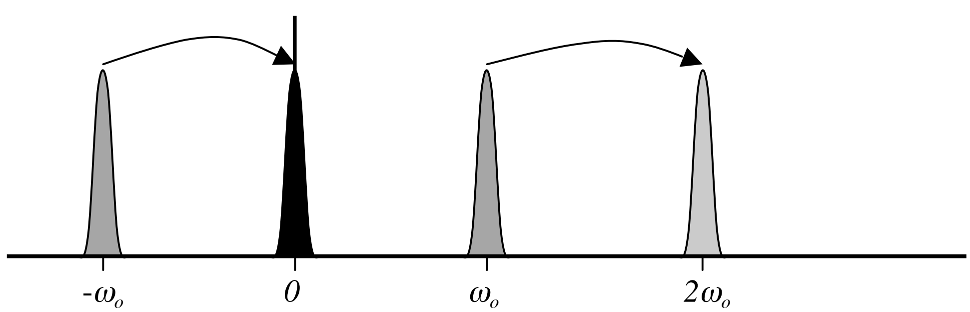
SAR IMAGE FORMATION:
ERS SAR PROCESSOR CODED IN MATLAB

The Range-Doppler Algorithm (Curlander and McDonough, *Synthetic Aperture Radar: Systems & Signal Processing*, Chapter 4, John Wiley & Sons, New York, 1991.)

The basic SAR theory is conceptually simple yet when one looks into the inner workings of the SAR processor it appears quite complicated. The most difficult part of the problem is related to the coupling between the azimuthal compression and the orbital parameters. In the old days when the satellite orbits were not known very accurately, one would use techniques such as *clutterlock* and *autofocus* to derive the orbital parameters from the data. This is no longer necessary with the high accuracy orbits available today. A standard processing sequence follows where the first two steps are done onboard the satellite while the remaining 4 steps are done by the user with a digital SAR processor.

Processing Onboard the Satellite

Demodulate - The electromagnetic wave consists of a chirp of bandwidth B (~15 MHz) superimposed on carrier frequency ω_o (~5 GHz). To record the return signal, one could simply digitize the amplitude of the electric field $E(t)$ at a sampling rate of 10 GHz using 1-byte encoding. However, this would generate 10 Gbytes of data every second! We are really only interested in how the chirp has been delayed and distorted by the reflection off the Earth' surface. The trick here is to use the shift theorem [*Bracewell*, 1978, p. 108] to isolate the 15 MHz part of the spectrum containing the chirp. Suppose we multiply the signal $E(t)$ by $e^{i2\pi\omega_o t}$. This will shift the fourier transform of the signal as shown in the diagram below.



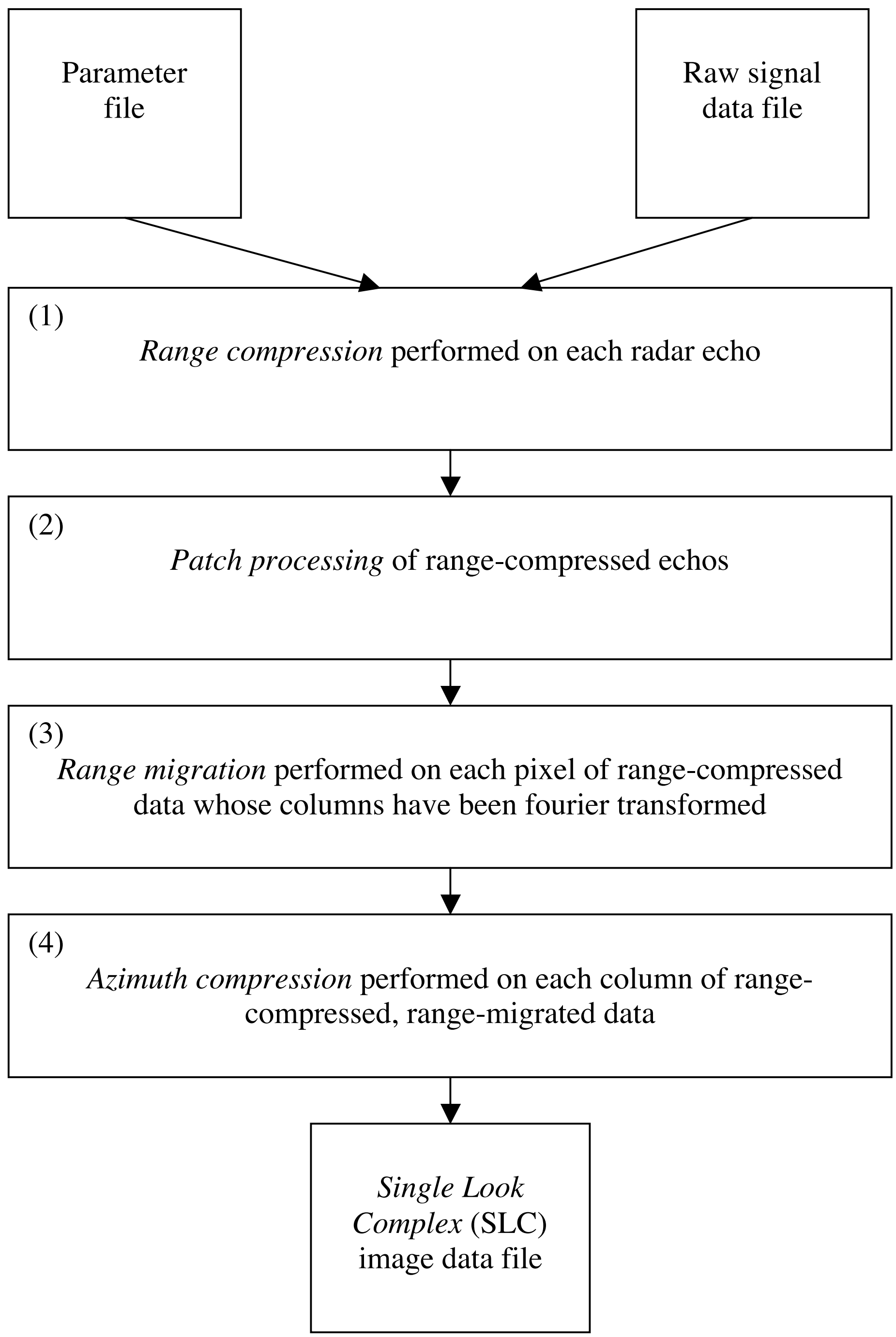
The function $E(t)$ is a real-valued function so we know that its fourier transform has Hermitian symmetry about the origin, that is $E(-\omega)=E^*(\omega)$. Because of this symmetry, all of the information in the time series is contained within the bandwidth of a single spectral peak. Next we low-pass filter the signal to isolate the spectral peak centered around the zero frequency. The original signal $E(t)$ was a real-valued function but after this shift/filter operation the output complex. No information is lost in this process and now the output can be digitized at the much lower rate of twice the bandwidth of the chirp (~30 MHz).

Digitize – The complex signal is digitized at 5 bits per pixel so the numbers range from 0 to 31; the mean value is about 15.5. Digitizing the data at 8 bits per pixel would seem more convenient but sending these extra 3 bits to the ground receiving exceeds the bandwidth of the telemetry so every effort is made to compress the data before transmission. Once the data are on the ground they are expanded to 8 bits (one byte) for programming convenience. Engineers refer to the real and imaginary parts of the signal as the in-phase (I) and quadrature (Q) components. The raw signal files contain rows of (11644 bytes) that represent a single radar echo. The first 412 bytes are timing and other information while the remaining 11232 bytes contain the 5616 complex numbers of raw signal data.

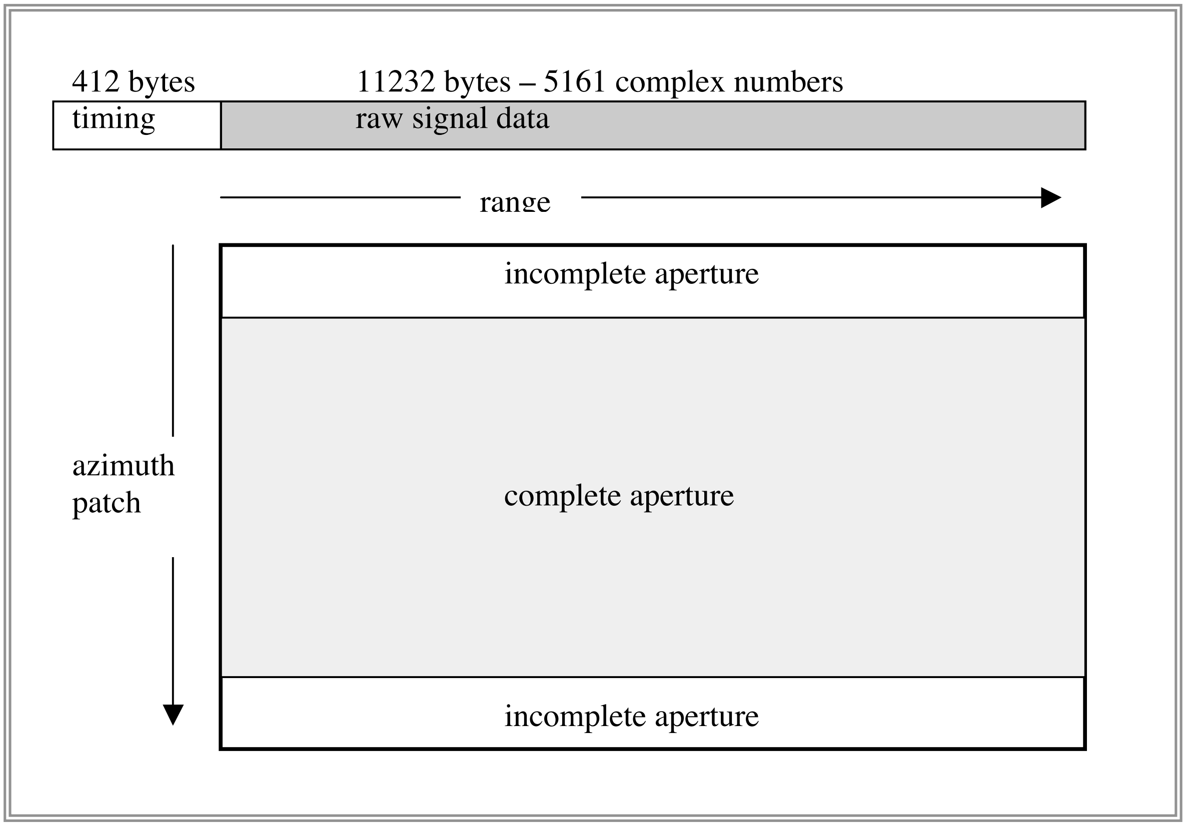
Digital SAR Processing

The digital SAR processor is a computer program that converts the raw signal data into a single-look complex (SLC) image. An overview is provided in the diagram below this is followed by a detailed description of each step. With this information one can write a basic SAR processor using just a few lines of code in MATLAB. An example program is provided in the Appendix.

Digital SAR Processing Overview



Raw Signal Data



Parameter File (e2_10001_2925.PRM)

file information/format			
input_file	e2_10001_2925.fix	name of raw data file	
SC_identity	2	(1) – ERS-1 or (2) – ERS-2	
bytes_per_line	11644	number of bytes in row of raw data	
first_sample	206	412 bytes of timing information to skip over	
fd1	248.115	Doppler centroid estimated from data	f_{Dc}
I_mean	15.504000	mean value of real numbers	
Q_mean	15.549000	mean value of imaginary numbers	
icu_start	2576039268.848	spacecraft clock for first sample	
SC_clock_start	1997078.767881851	UTC for first sample YYYDDD.DDDDDDDD	
SC_clock_stop	1997078.768074699	UTC for last sample YYYDDD.DDDDDDDD	
radar characteristics			
PRF	1679.902394	pulse repitition frequency	PRF
rng_samp_rate	1.89625e+07	range sampling rate	f_s
chirp_slope	4.17788e+11	chirp slope	k
pulse_dur	3.712e-05	pulse duration	τ_p
radar_wavelength	0.056666	radar wavelength	λ
orbital information			
near_range	829924.365777	distance to first range bin	R_{near}
earth_radius	6371746.4379	earth radius 1/2 way into image	R_e
SC_height	787955.52	spacecraft height 1/2 way into image	H
SC_vel	7125.0330	ground-track velocity	V
processing information			
num_valid_az	2800	size of patch to process	
num_patches	10	number of patches to process	
first_line	1		
deskew	n	deskew (yes or no)	
st_rng_bin	1	start processing at row number	
num_rng_bins	6144	number of range bins in output file	
chirp_ext	614	extend the chirp to process outside swath	
Flip_iq	n	exchange real and imaginary numbers (yes or no)	
nlooks	1	number of azimuth echoes to average	
image alignment			
rshift	15.1	range shift to align image to master	
ashift	483.2	azimuth shift to align image to master	
stretch_r	.0014569	range stretch versus range	
stretch_a	-.0019436	azimuth stretch versus range	
a_stretch_r	0.0	range stretch versus azimuth	
a_stretch_a	0.0	azimuth stretch versus azimuth	

(1) *Range Compression* – A sharp radar pulse is recovered by deconvolution of the chirp. This is done with fast fourier transform. There are 5616 points in the ERS-1 signal data and the chirp is 703 points long. Both are zero-padded to a length of 8192 prior to deconvolution to take advantage of the speed radix-2 fft algorithms. (Later we keep 6144 points, which enables one to extract the phase beyond the edges of the original swath. Since the chirp really did interact with the ground outside the digitized swath, this is not magic. The number 6144 has small prime factors that may be optimal for later 2-D fft phase unwrapping algorithms.)

(2) *Patch Processing* – The next step is to focus the image in the along-track or *azimuth* direction. This is also done by fft, but now we need to process columns rather than rows. For the ERS radar, the synthetic aperture is 1296 points long so we need to read at least that many rows into the memory of the computer. Actually 4096 rows is better number since it is a power of 2. A typical ERS raw data file has 28,000 rows so we will need to process many of these *patches*. One problem with this approach is that the synthetic aperture is only complete for the inner 2800 points of the 4096 data patch so the patches must have a 1296-point overlap. A 4096 x 5616 patch requires at least 184 Mbytes of computer memory for single precision real numbers. In the old days before this large memory space was available, each patch was transposed - *corner turning* – using high-speed disks so the data could be processed on a column-by-column basis. Nowadays a multiprocessor computer could process many patches in parallel and assemble the output files in the proper order since the overlapping patches are independent.

(3) *Range Migration* – As we will see below, a point target will appear as a hyperbolic-shaped reflection as it moves through the synthetic aperture. In addition, there could be a pronounced linear drift due to an elliptical orbit and earth rotation. In other words, the target will migrate in range cell as a linear trend plus a hyperbola. The shape of this migration path is calculated from the precise orbital information. Prior to focusing the image along a single column, these signals must be migrated back to a constant range cell. This is called *range migration* and the fastest way to do this is by fourier transforming the columns first. Each fourier component corresponds to a unique Doppler shift and also a unique value of range migration; For an ideal radar, the first component will have zero Doppler and will correspond to the point on the earth that is perpendicular to the spacecraft velocity vector. The second component will have a small positive Doppler shift and a small migration of the range cells will be needed - and so on all the way throught the positive and negative Doppler spectrum.

(4) *Azimuth Compression* – The final step in the processing is to focus the data in azimuth by accounting for the phase shift of the target as it moves through the aperture. In the lecture on diffraction, we calculated the illumination pattern on a screen due to the propagation of a coherent wave from the aperture to the screen. The azimuth compression relies on the same theory although we do the opposite calculation – given the illumination pattern, calculate the shape of the aperture (reflectivity of the target). This is done by generating a second frequency-modulated chirp where the chirp parameters depend on the velocity of the spacecraft, the pulse repetition frequency (*PRF*), and the absolute range. The chirp is fourier transformed into Doppler space and

multiplied by each column of range-migrated data. The product is inverse fourier transformed to provide the focused image.

Single-Look Complex (SLC) Image – As described above, only the inner 2800 rows use the complete synthetic aperture so these are written to an output file. The file-reading pointer is moved back 1296 rows to start the processing of the next patch. Because complete apertures are used for each patch, they will abut seamlessly.

Range Compression

To reduce the peak power of the radar transmitter associated with a short pulse, a long frequency-modulated chirp is emitted by the radar. This chirp propagates to the ground where it reflects from a swath typically 100 km wide. When it returns to the radar, the raw signal data consists of the complex reflectivity of the surface convolved with the chirp. Our objective is to recover the complex reflectivity by deconvolution of the chirp. As discussed above, this is the first step in the digital SAR processor. For the ERS-radar, the frequency-modulated chirp is

$$s(t) = e^{i\pi k t^2} \qquad |t| < \tau_p / 2.$$

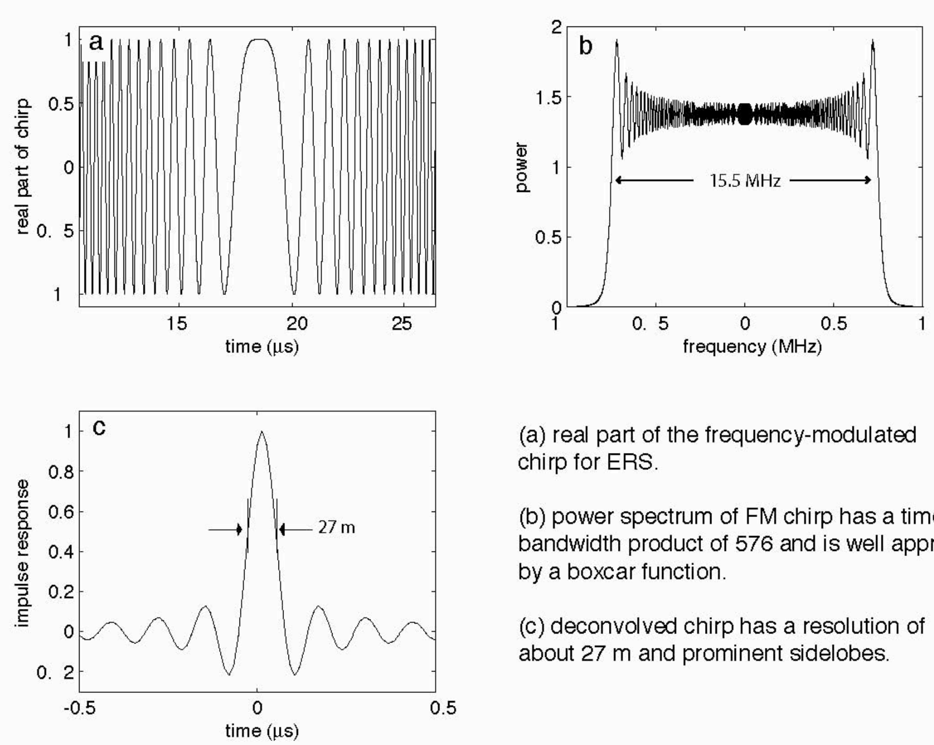
where

k - chirp slope (4.17788×10^{11} s^{-2})

τ_p - pulse duration (3.712×10^{-5} s or ~11 km long)

f_s - range sampling rate (1.89625×10^{7} s^{-1}).

An example of a portion of the chirp for the ERS- radar as well as its power spectrum and impulse response is shown below.



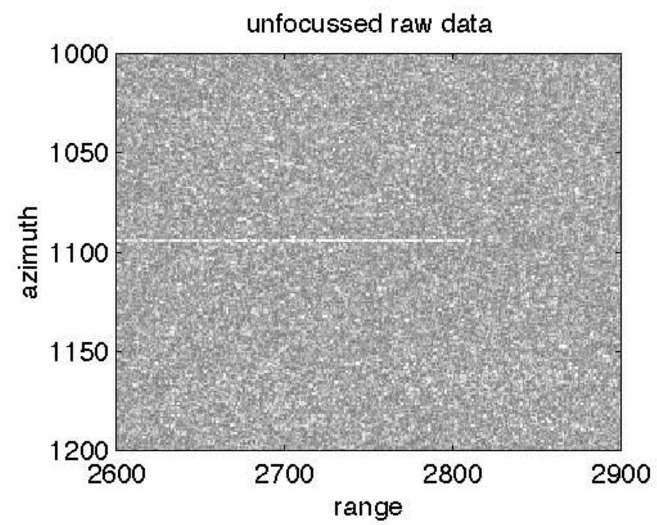
(a) real part of the frequency-modulated chirp for ERS.

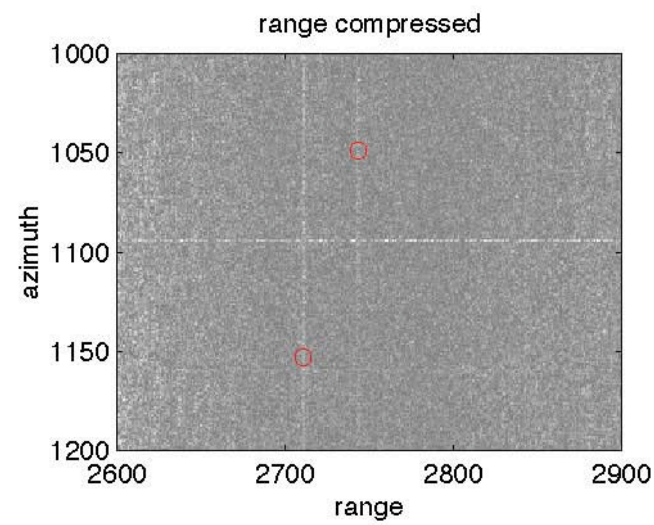
(b) power spectrum of FM chirp has a time-bandwidth product of 576 and is well approximated by a boxcar function.

(c) deconvolved chirp has a resolution of about 27 m and prominent sidelobes.

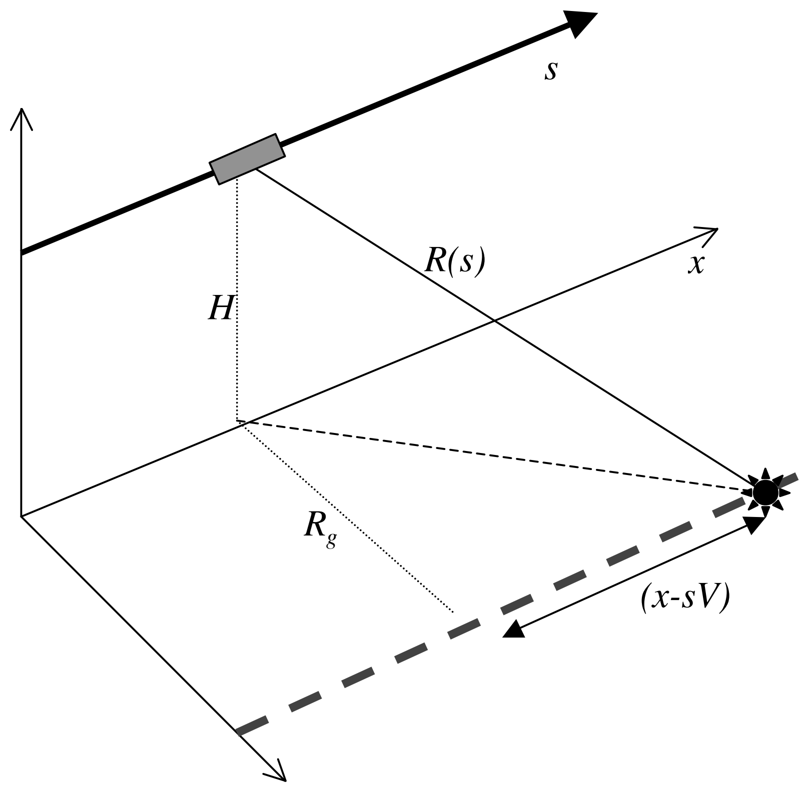
The bandwidth of the chirp, $B = k\tau_p$, has a value of 15.5 MHz for the ERS radar. A matched filter is used to deconvolve the chirp from the data. In this case, the matched filter is simply the complex conjugate of the chirp or $s^*(t) = e^{-i\pi kt^2}$. The convolution of $s^*(t)$ and $s(t)$ is shown in the figure above. The effective range resolution of the radar is about 27 m.

Using MATLAB code (Appendix) and the radar parmaters provided above, one can deconvolve the ERS chirp. The figure below is a small patch of ERS-2 data for an area around Pinyon Flat, California (track 127, frame 2925, orbit 10001). This area contains two radar reflectors (see figure below) spaced about 600 m apart that were installed in late 1996. The amplitude of the raw signal data appears as noise with a single horizontal streak due to missing data. The range-compressed data (right image) shows two vertical streaks due to the high reflectivity of the reflectors (red circles) as they migrate through the synthetic aperture.





Azimuth Compression
Azimuth compression or azimuth focusing involves generation of a frequency-modulated chirp in azimuth based on the knowledge of the spacecraft orbit. The geometry of the strip-mode acquisition is shown below.



s	– slow time along the satellite track
x	– ground-track position
V	– ground track velocity
s_o	– time when the target is in the center of the radar illumination pattern
H	– spacecraft height
R_g	– ground range
$R(s)$	– range from spacecraft to target
R_o	$= \sqrt{H^2 + R_g^2} = R_{near} + n*(C/fs)$ minimum range from the spacacraft to the target

The complex phase of the return echo is

$$C(s) = \exp\left[i\frac{4\pi}{\lambda}R(s)\right]$$

where the range is

$$R^2(s) = H^2 + R_g^2 + \left(x - sV\right)^2.$$

This is a hyperbola that we can approximate using a parabola

$$R(s) = R_o + \dot{R}_o(s - s_o) + \frac{\ddot{R}_o}{2}(s - s_o)^2 + ...$$

where the dot indicates derivative with respect to slow time, *s*. *Curlander and McDonough* [1991] discuss the accuracy of this polynomial approximation. Is is good enough for strip-mode SAR but may be inadequate for the much longer apertures associated with spotlight-mode SAR.

Now we need to calculate $\dot{R}$ and $\ddot{R}$ in terms of spacecraft parameters. Lets start with $\dot{R}$ by taking the derivative of R^2 with respect to *s*.

$$\frac{\partial R^2}{\partial s} = 2R\dot{R} \qquad \Rightarrow \qquad \dot{R} = \frac{1}{2R}\frac{\partial R^2}{\partial s}$$

We note that $\dot{R}_o$ is the velocity of the spacecraft in the range direction at the time s_o and thus a measure of the Doppler shift when the target is in the center of the radar illumination pattern.

$$\dot{R}_o = -V\frac{\left(x - s_o V\right)}{R_o}$$

The range acceleration $\ddot{R}$ is the derivative of the range rate $\dot{R}$.

$$\ddot{R} = \frac{\partial}{\partial s}\left(\frac{1}{2R}\frac{\partial R^2}{\partial s}\right) = \frac{1}{2}\left[\frac{1}{R}\left(\frac{\partial^2 R^2}{\partial s^2}\right) + \frac{-1}{R^2}\left(\frac{\partial R}{\partial s}\right)\left(\frac{\partial R^2}{\partial s}\right)\right]$$

This can be written as

$$\ddot{R} = \frac{V^2}{R} + \frac{-V^2\left(x - sV\right)^2}{R^3} = \frac{V^2}{R}\left[1 - \frac{\left(x - sV\right)^2}{R^2}\right].$$

For the orbital characteristics of the ERS spacecraft, the second term is small so we have

$$\ddot{R}_o \cong \frac{V^2}{R_o}$$

Now we can write the phase of the return signal as a function of geometry and speed

$$C(s) = \exp\left\{ i\frac{4\pi}{\lambda}\left[R_o^2 + \dot{R}_o(s - s_o) + \ddot{R}_o(s - s_o)^2/2\right]\right\}$$

or

$$C(s) = \exp\left\{ i\frac{4\pi R_o^2}{\lambda}\right\}\exp\left\{ i2\pi\left[\boxed{\frac{-2V}{\lambda}\frac{(x - s_oV)}{R_o}}(s - s_o) + \boxed{\frac{2V^2}{\lambda R_o}}(s - s_o)^2/2\right]\right\}$$

Doppler centroid   Doppler frequency rate

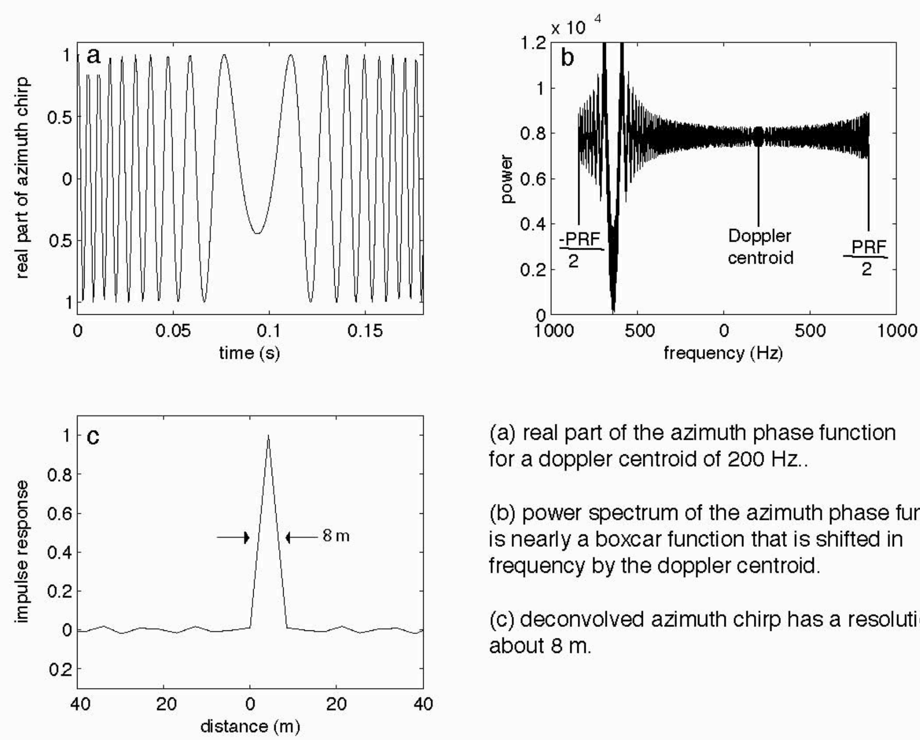
Note that this function is another frequency-modulated chirp where and there are two important parameters the *Doppler centroid*

$$f_{DC} = \frac{-2V}{\lambda}\frac{(x - s_oV)}{R_o}$$

and the *Doppler frequency rate*

$$f_R = \frac{2V^2}{\lambda R_o}.$$

An example of this azimuthal chirp function for the ERS orbit/radar as well as its power spectrum and impulse response are shown below.



(a) real part of the azimuth phase function for a doppler centroid of 200 Hz..

(b) power spectrum of the azimuth phase function is nearly a boxcar function that is shifted in frequency by the doppler centroid.

(c) deconvolved azimuth chirp has a resolution of about 8 m.

The bandwidth of the azimuthal chirp is equal to the pulse repition frequency (PRF = 1680 Hz). In this case the matched filter is simply the complex conjugate of the azimuthal phase function or *C*(s)*.

Azimuthal Compression Parameters and ERS Orbit

The Doppler centroid and Doppler frequency rate depend on the ground velocity of the spacecraft V, the wavelength of the radar λ, the minimum distance between the spacecraft and the target R_o, and a factor $\frac{x - s_o V}{R_o}$ which is the *squint angle*.

Ground velocity - First consider the ground velocity of the spacecraft. Precise orbital information is used to compute the velocity of the spacecraft at satellite altitude V_s. As an exercise, show that the ground velocity is

$$V = \frac{V_s}{\left(1 + \frac{H}{R_e}\right)^{1/2}}$$

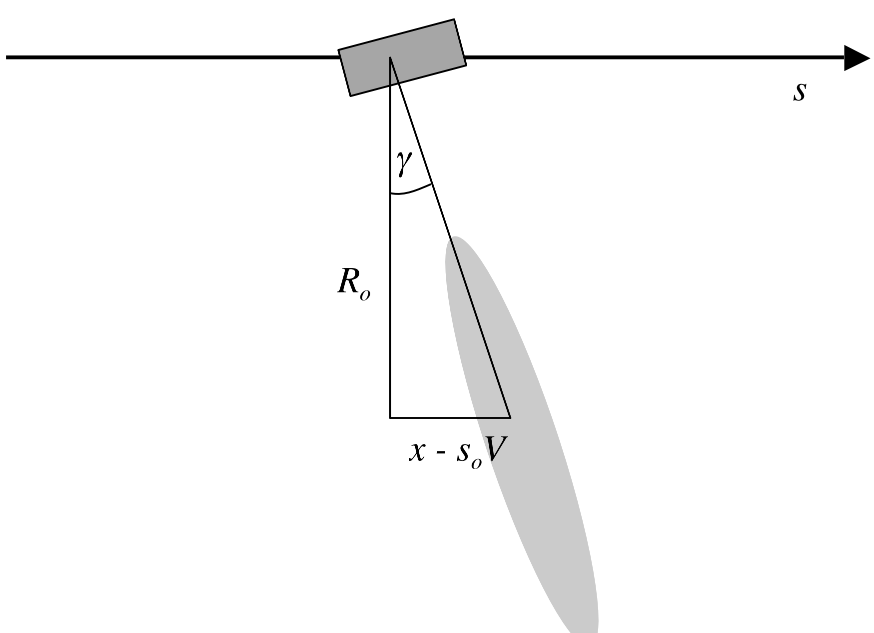
where H is the local spacecraft height and R_e is the local earth radius.

Range - The quantity R_o is the range to a particular column in the raw signal data file.

$$R_o = R_{near} + n(c/2f_s)$$

where R_{near} is the *near range* to the first column of the raw signal data file, n is the column number (0-5615), c is the speed of light and f_s is the range sampling rate given in the previous section on range compression.

Squint angle - Finally lets examine the quantity, $\frac{x - s_o V}{R_o}$. This is related to something called the squint angle, γ, as shown in the following diagram.



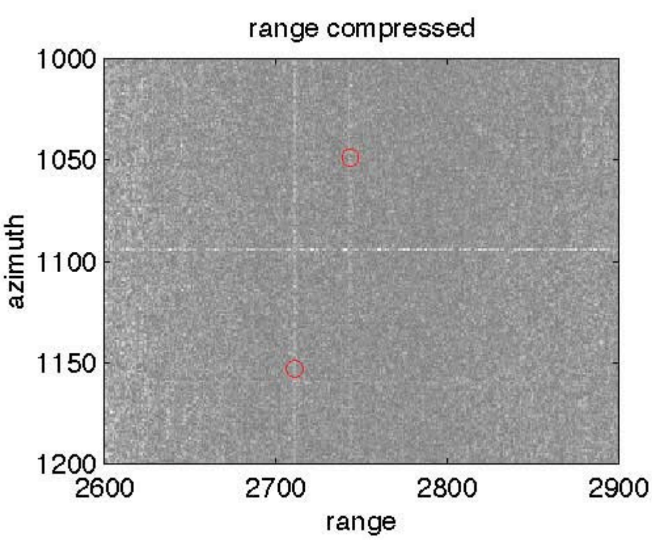
The squint angle is simply

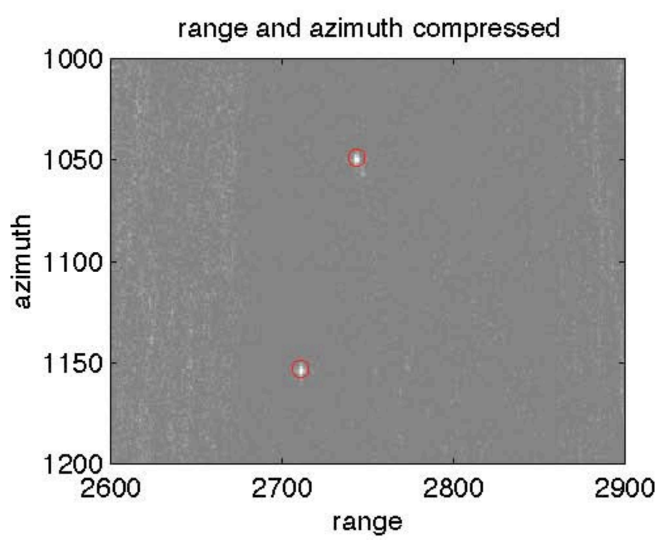
$$\gamma = \tan^{-1}\left(\frac{x - s_o V}{R_o}\right).$$

For the ERS satellites, the squint angle is adjusted so the zero Doppler occurs roughly in the center of the antenna beam pattern. If the satellite orbit was circular and the earth was a non-rotating sphere, then the zero Doppler would correspond to a zero squint angle.

For interferometry it is important that the beam patterns of the reference and repeat orbits have a large overlap at a given along-track spacecraft position. The requirement is that the Doppler centroid of the reference and repeat passes agree to about 1/2 of the pulse repetition frequency (*PRF*). In late 1999, the gyroscopes on ERS-2 failed so it became difficult to control the squint angle of the spacecraft. Data acquired after this date may a Doppler centroid outside of the acceptable range and thus may be useless for interferometry.

Using MATLAB code (Appendix) and the azimuth compression provided above, one canfocus the image in the azimuth direction. The figures below the same small patch of ERS-2 data for an area of Pinyon Flat, California. The range-compressed data (left image) shows two vertical streaks due to the high reflectivity of the reflectors (red circles) as they migrate through the synthetic aperture. The fully focused image (right) shows the high point-like reflectivity associated with the two radar reflectors.





Appendix – MATLAB Code for Focusing ERS Signal Data

```
%****************************************************
function [cref,fcref]=rng_ref(nfft,fs,pulsedur,slope)
%
%  routine to compute ERS chirp and its fourier
transform
%
%  input
%   fs       - sampling frequency, ts=1./fs
%   pulsedur - pulse duration
%   slope    - chirp slope
%
%   set the constants and make npts be odd
%
  npts=floor(fs*pulsedur);
  ts=1./fs;
  if(mod(npts,2.0) ==  0.0)
        npts=npts+1;
  end
%
%  compute the reference function
%
  npt2=floor(npts/2.);
  t=ts*(-npt2:npt2);
  phase=pi*slope*t.*t;
  cref1=exp(i*phase);
%
%  pad the reference function to nfft
%
  cref=[cref1,zeros(1,nfft-npts)]';
%
%  compute the fourier transform
%
  fcref=fft(cref)/nfft;

%****************************************************
  function [cazi,fcazi]=azi_ref(nazi,PRF,fdc,fr)
%
% routine to compute ERS azimuthal chirp and its
fourier transform
%
%  input
%   nazi     -  number of points in azimuth
%   PRF      -  pulse repitition frequency, ts=1./fs
%   fdc      -  doppler centriod frequency
%   fr       -  doppler frequency rate
%
%   set the constants and make npts be odd
%
  npts=min(nazi-1,1296);
  ts=1./PRF;
  if(mod(npts,2.0) ==  0.0)
        npts=npts+1;
  end
%
%  compute the azimuth chirp function
%
  npt2=floor(npts/2.);
  t=ts*(-npt2:npt2);
  phase=-2.*pi*fdc*t+pi*fr*t.*t;
  cazi1=exp(i*phase);
%
%  pad the function to nfft
%
  cazi=[cazi1(npt2:npts),zeros(1,nazi-
npts),cazi1(1:npt2-1)]';
%
%  compute the fourier transform
%
  fcazi=fft(cazi)/nazi;


%****************************************************
function [csar,nrow,ncol]=read_rawsar(sar_file)
%
%  routine to read and unpack ERS SAR data in
DPAF format
%
  fid=fopen(sar_file,'r');
%
%  read the bytes
%
  [sar,nsar]=fread(fid,'uchar');
  sar=reshape(sar,11644,nsar/11644);
  nrow=nsar/11644;
  ncol=5616;
  st=fclose(fid);
%
%  extract the real and imaginary parts
%  and remove the mean value
%
  csar=complex(sar(413:2:11643,:)-
15.5,sar(414:2:11644,:)-15.5);
%

%****************************************************
%  matlab script to focus ERS-2 signal data
%
%  set some constants for e2_10001_2925
%
%  range parameters
%
 rng_samp_rate = 1.896e+07;
 pulse_dur = 3.71e-05;
 chirp_slope = 4.1779e+11;
%
% azimuth parameters
%
 PRF=1679.902394;
 radar_wavelength=0.0566666;
 SC_vel=7125.;
%
%   compute the range to the radar reflectors
%
 near_range=829924.366;
```

```
 dr=3.e08/(2.*rng_samp_rate);
 range=near_range+2700*dr;
%
%  use the doppler centroid estimated from the data
and the
%  doppler rate from the spacecraft velocity and
range
%
 fdc=284;
 fr=2*SC_vel*SC_vel/(range*radar_wavelength);
%
%  get some sar data
%
 [cdata,nrow,ncol] = read_rawsar('rawsar.raw');
%
% make a colormap
%
  map=ones(21,3);
  for k=1:21;
    level=0.05*(k+8);
    level=min(level,1);
    map(k,:)=map(k,:).*level;
  end
  colormap(map);
%
%  image the raw data
%
  figure(1)
  subplot(2,2,1),imagesc(abs(cdata'));
  xlabel('range')
  ylabel('azimuth')
  title('unfocussed raw data')
  axis([2600,2900,1000,1200])
%
%  generate the range reference function
%

[cref,fcref]=rng_ref(ncol,rng_samp_rate,pulse_dur,chir
p_slope);
%
%  take the fft of the SAR data
%
  fcdata=fft(cdata);
%
%  multiply by the range reference function
%
  cout=0.*fcdata;
  for k=1:nrow;
  ctmp=fcdata(:,k);
  ctmp=fcref.*ctmp;
  cout(:,k)=ctmp;
  end
  clear cdata
%
%  now take the inverse fft
%
  odata=ifft(cout);
  clear cout
%
% plot the image and the reflector locations
%
  x0=[2653.5,2621];
  x0=x0+90;
  y0=[20122,20226];
  y0=y0-19500+427;
  figure(1)
  hold
  subplot(2,2,2),imagesc(abs(odata'));
  plot(x0,y0,'o')
  xlabel('range')
  ylabel('azimuth')
  title('range compressed')
  axis([2600,2900,1000,1200])
%
%  use this for figure 2 as well
%
  figure(2)
  colormap(map);
  subplot(2,2,1),imagesc(abs(odata'));
  hold on
  plot(x0,y0,'o')
  xlabel('range')
  ylabel('azimuth')
  title('range compressed')
  axis([2600,2900,1000,1200])
%
%  generate the azimuth reference function
%
  [cazi,fcazi]=azi_ref(nrow,PRF,fdc,fr);
%
%  take the column-wise fft of the range-compressed
data
%
  fcdata=fft(odata');
%
%  multiply by the azimuth reference function
%
  cout=0.*fcdata;
  for k=1:ncol;
  ctmp=fcdata(:,k);
  ctmp=fcazi.*ctmp;
  cout(:,k)=ctmp;
  end
%
%  now take the inverse fft and plot the data
%
  odata=ifft(cout);
  figure(2)
  subplot(2,2,2),imagesc(abs(odata));
  hold on
  plot(x0,y0,'o')
  xlabel('range')
  ylabel('azimuth')
  title('range and azimuth compressed')
  axis([2600,2900,1000,1200])
```