# The Keystone Transformation for Correcting Range Migration in Range-Doppler Processing

Mark A. Richards

March 2014; Revised February 2020

## 1 Acknowledgement

Thanks to Dr. Gregory Showman of the Georgia Tech Research Institute for helping me to understand this topic with his usual lucid explanations; and to Stuart Rogers of the University of Minnesota for identifying and correcting a discrepancy between this text and the associated code, and for discussions regarding the commutativity of the keystone and ambiguity corrections. The keystone transformation for compensating range migration is discussed in a number of papers, e.g. [1],[2].

## 2 The Problem

Conventional range-Doppler (RD) processing collects a coherent processing interval (CPI) of fast-time/slow-time data and performs a slow-time discrete Fourier transform (DFT) on all range bins to convert it to an RD matrix. Implicit is the assumption that the target velocity $v$, CPI duration (*aperture time*) $T_a$, and range bin spacing $\delta R$ are such that the target's range change within the CPI is less than one range bin, $vT_a < \delta R$. In other words, the target stays in the same range bin over the duration of the CPI. If this is the case, the target signature will be concentrated in the same range bin on each pulse and a 1D slow-time DFT of the data in that range bin will result in a well-formed, full-resolution Doppler spectrum. For a constant-velocity target without slow-time windowing, this spectrum will be just an asinc[1] function in the Doppler coordinate with a Rayleigh width of $1/T_a$ Hz (windowing for sidelobe control will increase this). Figure 1a is a diagram of a notional data matrix in which the target stays in the same (gray-shaded) range bin over the CPI.

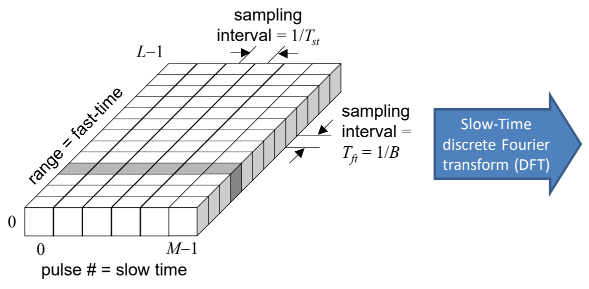
A simulated range-Doppler matrix was generated consistent with the following operating conditions:

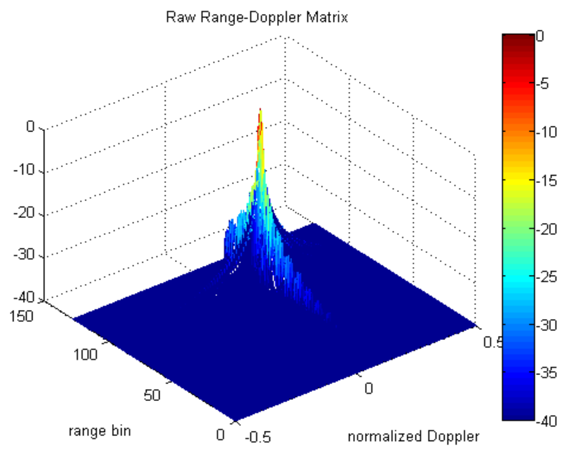
1. The radar frequency (RF) is $F_0$ = 10 GHz.
2. The waveform bandwidth is $B$ = 200 MHz.
3. Matched filtering is used, and the range response of the radar at the matched filter output is an unwindowed sinc function with a Rayleigh resolution of $\Delta R = c/2B$ =0.75 m. This would be consistent, for example, with use of a linear frequency modulated (LFM) pulse waveform.

[1] "aliased sinc" function, also sometimes called a "digital sinc" (dsinc) or Dirichlet function [3].

4. The fast-time axis is oversampled by a factor of 2.3×, so the spacing of the range bins is $\delta R$ = 0.3261 m.
5. The moving target is located in range bin #100 and approaches the radar at 20 m/s.
6. The pulse repetition frequency (PRF) is 10 kHz, giving a pulse repetition interval (PRI, denoted $T_{st}$) of 100 $\mu$s.
7. The CPI is $M$ = 101 pulses long, giving a a CPI duration of $T_a = M \cdot T_{st}$ = 10.1 ms. In this time, the target moves 0.202 m or 0.62 range bins.

Under these conditions, the target velocity corresponds to a Doppler shift of $F_D = 2vF_0/c$ = 1.33 kHz and an unaliased normalized Doppler shift of $f_D = F_D/PRF$ = +0.133 cycles/sample. The range-Doppler spectrum is shown in Figure 1b. Its peak does indeed occur at range bin $l$ = 100 and normalized frequency $f_D$ = +0.133.


sampling interval = $1/T_{st}$
$L-1$
range = fast-time
sampling interval = $T_{ft} = 1/B$
0
0
$M-1$
pulse # = slow time
Slow-Time discrete Fourier transform (DFT)



(a) (b)

**Figure 1. Range-Doppler matrix for a single target without range migration. (a) Arrangement of fast-time/slow-time data. (b) Resulting range-Doppler matrix. See text for parameters.**

When the target does not remain within a single range bin over the CPI, *range migration* is said to occur. The target Doppler signature will smear in both range and Doppler. It smears in range because portions of the target signature appear in more than one range bin. It smears in Doppler because any one range bin contains the signature for only a portion of the CPI. Since Doppler resolution (width of the asinc mainlobe) in a given range bin is inversely proportional to signal duration in that range bin, the reduced duration degrades the Doppler resolution (broadens the mainlobe). Range migration is obviously more severe for fast-moving targets. However, it is also made worse in wide-bandwidth systems. Such systems have fine range resolution and therefore smaller spacing between range bins, so that a given amount of motion crosses more bins.

Consider an $L \times M$ fast-time/slow-time matrix $y_{rd}[l,m]$. [2] Suppose a radar transmits a series of $M = 2M_a+1$ pulses which reflect from a target. The slow-time sampling interval (the radar PRI) is $T_{st}$ seconds. The pulses are indexed as $-M_a \le m \le M_a$. Assume the slow-time span of the CPI is from $-T_a/2 = -M_a T_{st}$ to $+T_a/2 = +M_a T_{st}$, so that the center of the CPI is at time $t = 0$ and pulse $m = 0$. Note that the unambiguous Doppler range is $\pm 1/2T_{st}$ Hz.

If the fast-time sampling interval is $T_{ft}$ seconds, the range bin spacing is $\delta R = cT_{ft}/2$ meters. It is usually the case that $T_{ft} \approx 1/B$, where $B$ is the instantaneous bandwidth of the radar waveform; but we do not require this. Assume the range corresponding to the first range bin ($l$ = 0) is $R_0$. Express the target range $R_{ref}$ at the center of the CPI as $R_{ref} = R_0 + R_{rel}$, i.e. $R_{rel}$ is the reference range relative to $R_0$. Let the range bin corresponding to $R_{rel}$ be $l_{rel}$. Then $l_{rel} = (2R_{rel}/c)/T_{ft} = R_{rel}/\delta R$.

The target radial velocity relative to the radar is $v$ m/s, with positive $v$ representing approaching targets. The target range on the $m$th pulse () will be $R_{ref} - vT_{st}m = R_0 + R_{rel} - vT_{st}m$, corresponding to range bin $l_{rel} - 2vT_{st}m/cT_{ft} = l_{rel} - vT_{st}m/\delta R$. The range bin number is rounded to the nearest integer.

The radar transmits a pulse of the form $\bar{x}(t) = x(t)\exp(j2\pi F_0 t)$, where $x(t)$ is the baseband waveform (for example, a simple pulse or LFM chirp, etc.). After taking out the delay of $mT_{st}$ to the beginning of that pulse's transmission and demodulation to baseband by multiplication with the function $\exp(-j2\pi F_0 t)$ to remove the carrier, the received fast-time signal for the $m$th pulse will be of the form [3]

$$\begin{aligned} y_m(t) &= x\left[t - \frac{2}{c}\left(R_{ref} - vT_{st}m\right)\right]\exp\left[-j\frac{4\pi F_0}{c}\left(R_{ref} - vT_{st}m\right)\right] \\ &= \exp\left(-j\frac{4\pi F_0}{c}R_{ref}\right)x\left[t - \frac{2}{c}\left(R_{ref} - vT_{st}m\right)\right]\exp\left(+j\frac{4\pi F_0}{c}vT_{st}m\right) \end{aligned} \tag{1}$$

Amplitude factors are of no concern here and so will be ignored.

Now assume this baseband signal is passed through the matched filter for the envelope $x(t)$. Regardless of the particular waveform, the matched filter fast-time output, after removing the matched filter delay, can be modeled as consisting of a dominant peak of Rayleigh width approximately $1/B$ seconds at the appropriate time delay, surrounded by low-amplitude sidelobes. For this analysis, a sinc function in fast time with a zero spacing of $1/B$ seconds, i.e. $x(t) = \sin(\pi Bt)/(\pi Bt) \equiv \operatorname{sinc}(Bt)$, has been selected to represent a typical matched filter output.[3] The demodulated and matched-filtered output becomes

[2] A lower case $r$ or $d$ in the subscript to a signal $y$ or $Y$ indicates a signal in the time domain in the corresponding dimension. An upper case $R$ or $D$ indicates that the signal is in the frequency domain in that dimension. An upper case $Y$ is used when at least one of the dimensions is in the frequency domain. Thus $y_{rd}$ is raw fast-time/slow-time data, while $Y_{rD}$ is a function of fast time and Doppler frequency, i.e. the range-Doppler matrix. Square brackets indicate discrete variables, parentheses indicate continuous variables.

[3] The definition $\operatorname{sinc}(x) \equiv \sin(\pi x)/\pi x$ is consistent with that used in both MATLAB® and [3].

$$y_m(t) \approx \exp\left(-j\frac{4\pi F_0}{c}R_{ref}\right)\operatorname{sinc}\left(B\left[t-\frac{2}{c}\left(R_{ref}-vT_{st}m\right)\right]\right)\exp\left(+j\frac{4\pi F_0}{c}vT_{st}m\right) \tag{2}$$

This analog data is converted to discrete fast-time data by sampling at the range bin interval of $T_{ft}$ seconds. Sampling begins at the time delay corresponding to $R_0$, so samples are taken at times $t = 2R_0/c + l \cdot T_{ft}$, $l = 0,\ldots,L-1$. The resulting fast-time vector is

$$\begin{aligned} y_m[l] &\approx \exp\left(-j\frac{4\pi F_0}{c}R_{ref}\right)\operatorname{sinc}\left(B\left[T_{ft}l-\frac{2}{c}\left(R_{rel}-vT_{st}m\right)\right]\right)\exp\left(+j\frac{4\pi F_0}{c}vT_{st}m\right) \\ &= \exp\left(-j\frac{4\pi F_0}{c}R_{ref}\right)\operatorname{sinc}\left[BT_{ft}\left(l-\frac{2}{cT_{ft}}R_{rel}+\frac{2vT_{st}}{cT_{ft}}m\right)\right]\exp\left(+j\frac{4\pi F_0}{c}vT_{st}m\right) \\ &= \exp\left(-j\frac{4\pi F_0}{c}R_{ref}\right)\operatorname{sinc}\left[BT_{ft}\left(l-l_{rel}+\frac{vT_{st}}{\delta R}m\right)\right]\exp\left(+j\frac{4\pi F_0}{c}vT_{st}m\right) \end{aligned} \tag{3}$$

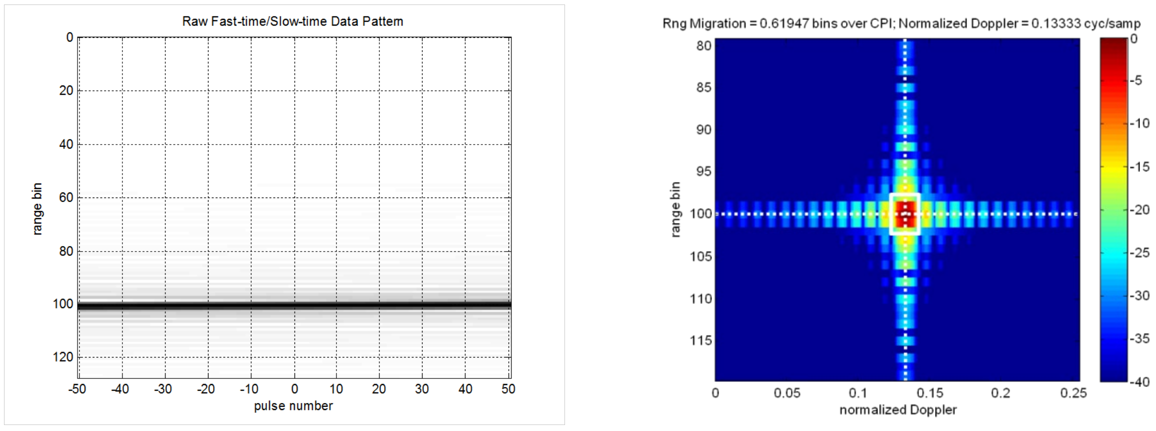
The returns from each of the $M$ pulses in the CPI are assembled into the raw fast-time/slow-time CPI data matrix $y_{rd}[l,m]$:

$$y_{rd}[l,m] \approx \exp\left(-j\frac{4\pi F_0}{c}R_{ref}\right)\operatorname{sinc}\left[BT_{ft}\left(l-l_{rel}+l_m\right)\right]\exp\left(+j\frac{4\pi F_0}{c}vT_{st}m\right) \tag{4}$$

where $l_m = vT_{st}m/\delta R$. Note that $l_m$ is not integer in general.

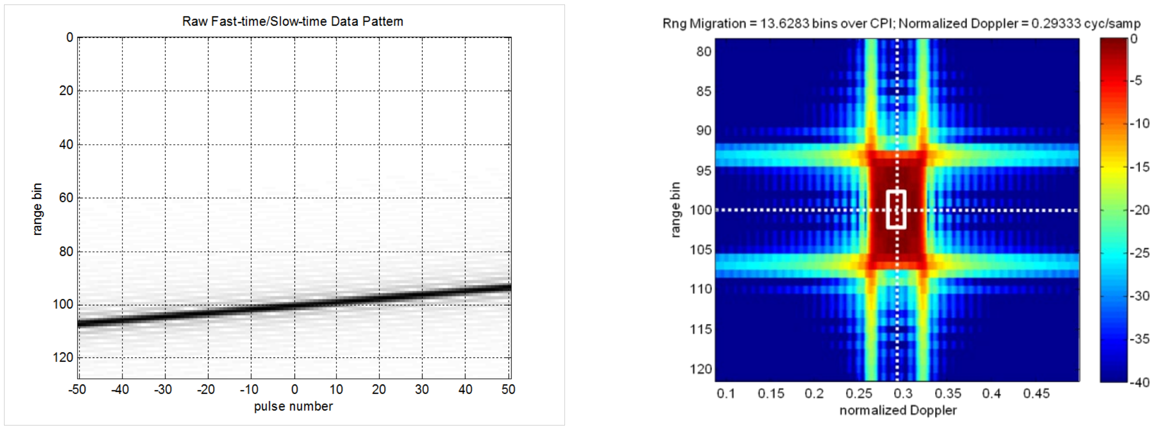
Figure 2a illustrates the pattern of data corresponding to Eq. (4) for the same example used to obtain Fig. 1. Again, the simulation was carried out for $L$ = 128 range bins and $M$ = 101 pulses. The normalized Doppler shift is $f_D = F_D T_{st} = (2vF_0/c)T_{st}$ = 0.133 cycles/sample. There is no significant range migration (0.62 range bins over the CPI) from the reference range bin of $l_{rel}$ = 100 at the center of the CPI. Figure 2a shows the magnitude of the range bin vs. pulse number (fast-time/slow-time) data, illustrating that the signal peak stays within range bin #100. The oversampling factor of 2.3× in fast time makes it possible to see some of the sidelobe structure detail of the sinc response in fast time. Figure 2b shows the central portion of the resulting range-Doppler spectrum. In this and subsequent similar figures, the white dotted lines mark the reference range bin at the middle of the CPI ($l_{rel}$ = 100) and normalized Doppler shift ($f_D$ = 0.133) of the spectrum, and thus the expected spectrum peak location. The white rectangle shows the expected Rayleigh resolution extent (expected location of first null) in each dimension. In the absence of range migration, the spectrum is a clean two-dimensional sinc/asinc with its peak at the correct location and full resolution in both range and Doppler (that is, the peak is entirely contained within the white rectangle).

Figure 3 shows a similar example, with two changes. The target velocity is now 440 m/s, giving a significant range migration of 13.63 bins over the CPI that is readily visible in Figure 3a. The RF frequency has been reduced to $F_0$ = 1 GHz, making the Doppler shift $F_D$ = 2.933 kHz and the normalized Doppler frequency now $f_D$ = 0.2933 cycles/sample. The reduction in $F_0$ allows for an unaliased Doppler at this velocity; the aliased case is considered in Section 7. Figure 3b shows that the range-Doppler signature is now spread over a little more than 13 range bins. Its mainlobe in the Doppler dimension, while correctly centered, has also been severely broadened.



(a) (b)

**Figure 2. Range-Doppler matrix with no range migration. (a) Range vs. pulse number. (b) Zoom into the central portion of the resulting range-Doppler matrix.**



(a) (b)

**Figure 3. Range-Doppler matrix with range migration. Compare to Fig. 2. (a) Range vs. pulse number. (b) Zoom into non-zero portion of the resulting range-Doppler matrix.**

## 3 Compensation by Range Shifting

The goal of the keystone transformation is to develop a process that will compensate for the range migration so that the range-Doppler spectrum of the data in Fig. 3a looks like Fig. 2b instead of Fig. 3b. To begin, consider compensating for range migration in the not-very-useful case of a single target with known radial velocity. The target moves $vT_{st}/\delta R$ range bins closer to the radar on each successive pulse. This is easily corrected by shifting each successive fast-time vector by $-vT_{st}/\delta R$ bins with respect to the previous pulse data. The reference range to which all echoes will be shifted can be chosen arbitrarily. One obvious choice is the range $R_{ref}$ at the middle of the CPI.

The range can be shifted using properties of the discrete-time Fourier transform (DTFT). Recall that $y_m[l]$ is the fast-time data for the $m$th pulse given in Eq. (3), and let $Y_m(\omega_l)$ be its DTFT. Then standard DTFT properties state that

$$\mathbf{F}^{-1}\left\{\exp\left(-j\omega_l l_m\right)Y_m\left(\omega_l\right)\right\} = y_m\left[l-l_m\right] \equiv y'_m\left[l\right] \tag{5}$$

where $\mathbf{F}^{-1}\{\cdot\}$ represents the inverse DTFT (IDTFT). Because the desired shift $l_m = vT_{st}m/\delta R$ is not integer in general, the quantity $y_m\left[l-l_m\right]$ should be interpreted as meaning the corresponding shifted analog signal $y_m\left(t-\left(vT_{st}/\delta R\right)m\right)$ sampled at the times $T_{ft}l$.

The shift is applied in practice by first computing the $K_l$-point DFT $Y_m[k_l]$ of the fast time data, then performing an "FFT shift" operation to circularly shift the resulting vector of $K_l$ DFT samples so that the sample corresponding to $k_l = 0$ (and thus to $\omega_l = 0$) is in the center of the vector. The DFT size $K_l$ must be greater than or equal to the number of range bins, $L$. The shifted DFT is then multiplied by the phase function from Eq. (5). In DFT form, this is $\exp\left[-j\left(2\pi k/K_l\right)\left(vT_{st}/\delta R\right)m\right]$, where $k$ is the DFT index. The modified DFT is circularly rotated back to its original order,[4] and finally an inverse DFT is computed to obtain $y'_m\left[l\right]$. The inverse DFT returns a $K_l$-point fast-time sequence; only the first $L$ points are retained.

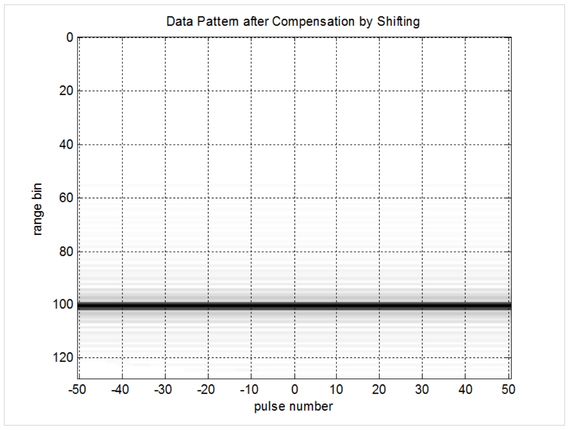
Figure 4 illustrates the result of applying this compensation procedure to the data of Fig. 3a. The results are excellent: the target signature data is re-centered into range bin 100 for all pulses, resulting in a range-Doppler spectrum with the full expected resolution in both dimensions. The spectrum shape is nearly indistinguishable in shape from that of the negligible-migration case of Fig. 2, except for being centered at a different Doppler value in accordance with the change in velocity and RF.

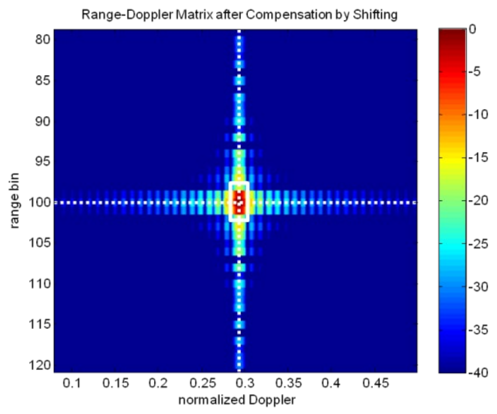
While the results in Fig. 4 are very good, they require two assumptions, both problematic:

1. The target velocity $v$ is known. This is required to compute the DFT phase modification function in Eq. (5). Knowledge of $v$ implies that the target is already under track or that other sensors or information sources provide this information. Alternatively, the data can be processed with a series of different trial values of $v$ in an attempt to identify the velocity by finding the value that best "focuses" the spectrum, usually interpreted as that value that provides the largest peak. If the target velocity might be aliased, it may also be necessary to estimate both the aliased velocity and the ambiguity number (number of foldovers); see Section 7.
2. There is only one target, or all targets have the same radial velocity. Otherwise, there is more than one value of $v$ to be corrected, so there is no single DFT phase modification function that can be used.

[4] The MATLAB® functions `fftshift` and `ifftshift` implement the required rotations.

If all targets share a common known velocity, the range migration could be compensated even more simply by adjusting the start of the fast-time sampling times for each successive pulse so that each target remained at a constant delay relative to the transmission time for the current pulse. Even if target velocity is known only approximately, this transmit time adjustment technique should probably be used as a first-order data correction in order to reduce the amount of range migration that must be compensated by signal processing methods of phase shifting or keystone transformation.





(a) (b)

**Figure 4. Range-Doppler matrix with range migration and compensation using DFT phase multiplies. Compare to the spectrum shape in Figs. 2 and 3. (a) Range vs. pulse number. (b) Zoom into non-zero portion of the resulting range-Doppler matrix.**

## 4 The Keystone Transformation

A more robust range migration compensation method can be developed as follows. First let us establish the goal, which is to obtain the range-Doppler spectrum of Figure 4b corresponding to a target at relative range $R_{rel}$ and normalized Doppler shift $f_D$, with full resolution in both dimensions expected for the waveform bandwidth $B$ and CPI duration $T_a$. It is convenient to work with continuous fast time $t$ instead of sampled fast time $l$. In slow time we will use a mixture of continuous slow time $\tau$ and sampled slow time $m = \tau/T_{st}$. In the fast time (range) dimension the desired response is, to within a complex constant, $\operatorname{sinc}\left[B\left(t-2R_{rel}/c\right)\right]$.[5] The fast-time Fourier transform of this function would have unit magnitude over the fast-time baseband frequency interval $F\in\left[-B/2, B/2\right]$ and zero elsewhere, and a phase of $\exp\left[-j2\pi F\left(2R_{rel}/c\right)\right] = \exp\left(-j4\pi\, R_{rel}F/c\right)$ within that interval. This would be the same for each pulse (value of $m$) because there would be no range migration. In the slow-time dimension, the ideal Doppler spectrum is the DTFT of a constant-frequency discrete-time sinusoid at a

[5] The slow-time data is only available in sampled form, one sample per range bin per pulse. By "continuous slow-time data", we mean the continuous function obtained by bandlimited interpolation of the slow-time sample sequence, based on the slow time sampling interval of $T_{st}$ seconds.

Doppler frequency $2vF_0/c$, sampled at the interval $T_{st}$. The corresponding slow-time phase progression is the sequence $\exp\left[j2\pi\left(2vF_0/c\right)T_{st}m\right] = \exp\left[j4\pi\left(F_0/c\right)vT_{st}m\right]$ with continuous slow time equivalent $\exp\left[j4\pi\left(F_0/c\right)v\tau\right]$. Combining these desired fast- and slow-time data patterns gives the ideal two-dimensional continuous fast time/slow time data function

$$y_{rd\_ideal}(t,\tau) = \exp\left(-j\frac{4\pi}{c}F_0R_{ref}\right)\text{sinc}\left[B\left(t-\frac{2}{c}R_{ref}\right)\right]\exp\left(+j\frac{4\pi}{c}F_0v\tau\right), \quad \begin{array}{c}-\infty < t < +\infty,\\ -M_aT_{st} < \tau < +M_aT_{st}\end{array} \tag{6}$$

Sampling in slow time at intervals of $T_{st}$ gives

$$y_{rd\_ideal}(t,m] = \exp\left(-j\frac{4\pi}{c}F_0R_{ref}\right)\text{sinc}\left[B\left(t-\frac{2}{c}R_{ref}\right)\right]\exp\left(+j\frac{4\pi}{c}F_0vT_{st}m\right), \quad \begin{array}{c}-\infty < t < +\infty,\\ -M_a < m < +M_a\end{array} \tag{7}$$

Now form the two-dimensional function $Y_{Rd\_ideal}(F,\tau)$ by computing the FT of $y_{rd\_ideal}(t,\tau)$ in fast time. The result is

$$Y_{Rd\_ideal}(F,\tau) = \begin{cases}\exp\left[-j\frac{4\pi}{c}(F+F_0)R_{ref}\right]\exp\left(+j\frac{4\pi}{c}F_0v\tau\right), & \begin{array}{l}-B/2 < F < +B/2,\\ -M_aT_{st} < \tau < +M_aT_{st}\end{array}\\ 0, & \text{otherwise}\end{cases} \tag{8}$$

Sampling this function in slow time at intervals of $T_{st}$ gives

$$Y_{Rd\_ideal}(F,m] = \begin{cases}\exp\left[-j\frac{4\pi}{c}(F+F_0)R_{ref}\right]\exp\left(+j\frac{4\pi}{c}F_0vT_{st}m\right), & \begin{array}{l}-B/2 < F < +B/2,\\ -M_a < m < +M_a\end{array}\\ 0, & \text{otherwise}\end{cases} \tag{9}$$

Notice that the fast-time baseband frequency $F$ and the velocity $v$ are uncoupled, that is, they appear in separable terms. Consequently, the slow-time data pattern is the same at all frequencies in the fast-time frequency range. If the fast-time Fourier transform of the actual data $y_{rd}(t,m]$ can be manipulated to be in the form of Eq. (8), the corresponding range-Doppler spectrum will be correctly centered and well-focused in both dimensions.

To that end, now consider the Fourier transform of the actual fast-time signal of Eq. (2). Using arguments similar to those leading to Eq. (9) gives

$$\begin{aligned}\mathbf{F}\{y_m(t)\} &\approx \exp\left(-j\frac{4\pi F_0}{c}R_{ref}\right)\exp\left[-j2\pi F\left(\frac{2}{c}\left(R_{ref}-vT_{st}m\right)\right)\right]\exp\left(+j\frac{4\pi F_0}{c}vT_{st}m\right)\\ &= \exp\left[-j\frac{4\pi}{c}(F_0+F)R_{ref}\right]\exp\left[+j\frac{4\pi}{c}(F_0+F)vT_{st}m\right], \quad \begin{matrix}-B/2<F<+B/2,\\ -M_a<m<+M_a\end{matrix} \\ &\equiv Y_{Rd}(F,m]\end{aligned} \tag{10}$$

We see that the fast-time frequency and velocity are *not* separable in this expression, but instead are coupled in the slow-time phase term. At frequencies greater than $F_0$ ($F > 0$) the slow-time phase progression rate is faster than the desired value of $F_0 vT_{st}m$, while at frequencies lower than $F_0$ ($F < 0$) it is slower than desired.

Equation (10) is the model for the structure of the fast-time FT $Y_{Rd}(F,m]$ of the available measured data matrix $y_{rd}(t,m]$. We now seek an operation on $Y_{Rd}(F,m]$ that will transform it to the desired form $Y_{Rd\_ideal}(F,m]$ of Eq. (9). An obvious approach is to multiply $Y_{Rd}(F,m]$ by the slow-time phase function $\exp\left[-j(4\pi/c)FvT_{st}m\right]$. However, this technique has the same flaws (and in fact is identical to) the range-shifting method of the previous section: one would have to know $v$ and the process would only work for one target velocity.

A clever way to remove the frequency-velocity coupling proceeds as follows. Concentrate on the slow-time phase term in Eq. (10), since that is where the coupling occurs. Create a new function $Y_{Rd\_key}(F,m]$ by resampling the slow-time axis so that the slow-time sample-to-sample phase increment is the same for each fast-time frequency bin, rather than varying with $F$ as it does in Eq (10). This can be done by sampling in slow time at intervals at times $\tau = \left[F_0/(F_0+F)\right]T_{st}m$. This resampling in turn implies the need to interpolate to obtain new values of the slow-time phase function between the existing values at $\tau = T_{st}m$. An appropriate processing approach is bandlimited interpolation using sinc-based interpolation kernels [4], though simpler methods such as spline interpolation might be adequate in some cases.

Bandlimited interpolation of the existing samples will give the slow-time function

$$\exp\left[+j\frac{4\pi}{c}(F_0+F)v\tau\right], \quad -M_aT_{st}<\tau<+M_aT_{st} \tag{11}$$

Sampling this function at $\tau = \left[F_0/(F_0+F)\right]T_{st}m$ gives the new slow-time function

$$\exp\left(+j\frac{4\pi}{c}F_0vT_{st}m\right), \quad -M_a<m<+M_a \tag{12}$$

which has the desired form of the slow-time portion of Eq. (9). Note that this result no longer varies with the fast-time frequency $F$. In terms of the fast-time-frequency/slow-time spectrum, the resampling operation forms a new spectrum $Y_{Rd\_key}(F,m]$ given by

$$Y_{Rd\_key}(F,m] = Y_{Rd}\left(F,\left(\frac{F_0}{F_0+F}\right)m\right] = Y_{Rd\_ideal}(F,m] \tag{13}$$

Equation (13) summarizes the keystone transformation operation and result. A fast-time FT and a fast-time-frequency-variant slow-time resampling of the acquired data will result in a new function that is the fast-time FT/slow-time inverse FT of the desired range-Doppler spectrum. An inverse FT in the range dimension and a forward FT in the pulse number dimension will result in the focused range-Doppler image with no degradation in either range or Doppler due to range migration.

Figure 5 illustrates the results of applying the keystone transformation to the same scenario used for Fig. 3. A Hamming-windowed bandlimited sinc interpolating filter was used to interpolate $Y_{Rd}(F,m]$ in slow time. The filter length was 11 times the maximum of the slow-time sample spacing before and after interpolation;[6] the specific value varies with fast-time frequency because of the variation in the interpolation factor $F_0/(F_0+F)$. Note in Fig. 5a that the range migration has been removed. The fading and fast-time smearing of the fast-time/slow-time signature at the beginning and end of the CPI is due to end effects of the interpolation: the first and last few slow-time samples cannot be fully interpolated because the interpolation filter impulse response extends beyond the ends of the available data. See Section 5 for more information. This will result in a slight loss of Doppler resolution that will become more severe for longer interpolation filters. Figure 5b shows the resulting range-Doppler spectrum. The peak is correctly centered and obtains very nearly full resolution in both dimensions. Some modification of the sidelobe structure is evident in the "X-shaped" Doppler sidelobes, but this is of little consequence and can be reduced by windowing.

A major advantage of the keystone transformation for range migration correction over the shifting process discussed earlier is that it correctly handles multiple targets at different velocities. Notice that the resampling of Eq. (13) does not depend on target velocity $v$. Consequently, target velocity need not be known, and this process will simultaneously correct range migration for multiple targets of various velocities. Figure 6 illustrates this with another example using three targets having (reference range bin,velocity (m/s)) pairs of (30,−200), (60,0), and (65,650). All radar parameters are unchanged from those used to obtain Figure 3. The resulting range migrations over the CPI are −6.1947, 0, and +20.1327 bins, while the normalized Doppler shifts are −0.1333, 0, and +0.4333 cycles/sample. Figure 6 illustrates the results. Part (a) of the figure shows the fast-time/slow-time data pattern. Note that two of the targets cross during the CPI. Figure 6b shows the range-Doppler spectrum before the keystone transformation. The zero-velocity (non-migrating) target is well-focused; the other two are not, with the degree of defocus increasing at higher velocities (more range migration).

[6] See the MATLAB® code for the function `sinc_interp` (included at the end of this memo) for details.

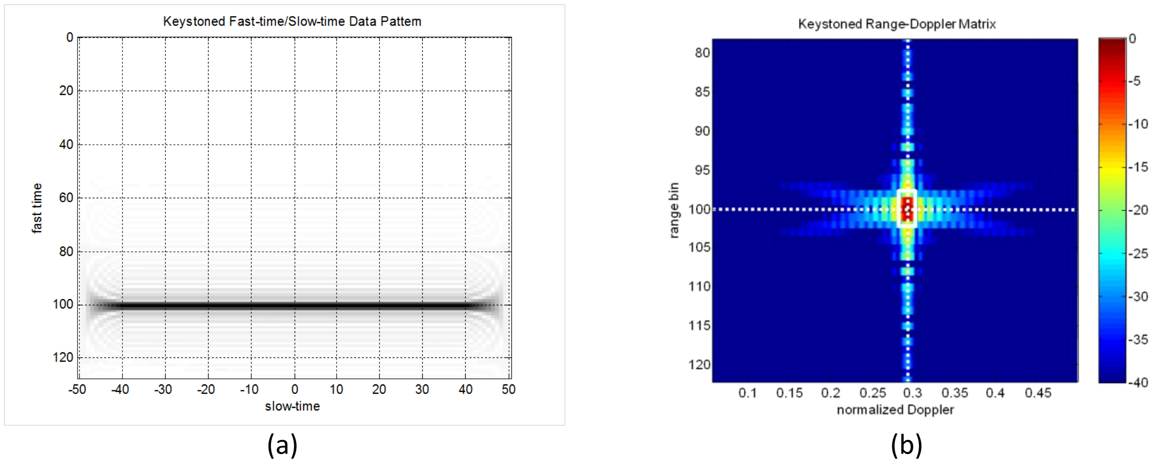


**Figure 5. Range-Doppler matrix with range migration and compensation using the keystone transformation. Compare to Figs. 3 and 4. (a) Range vs. pulse number. (b) Zoom into non-zero portion of the resulting range-Doppler matrix.**

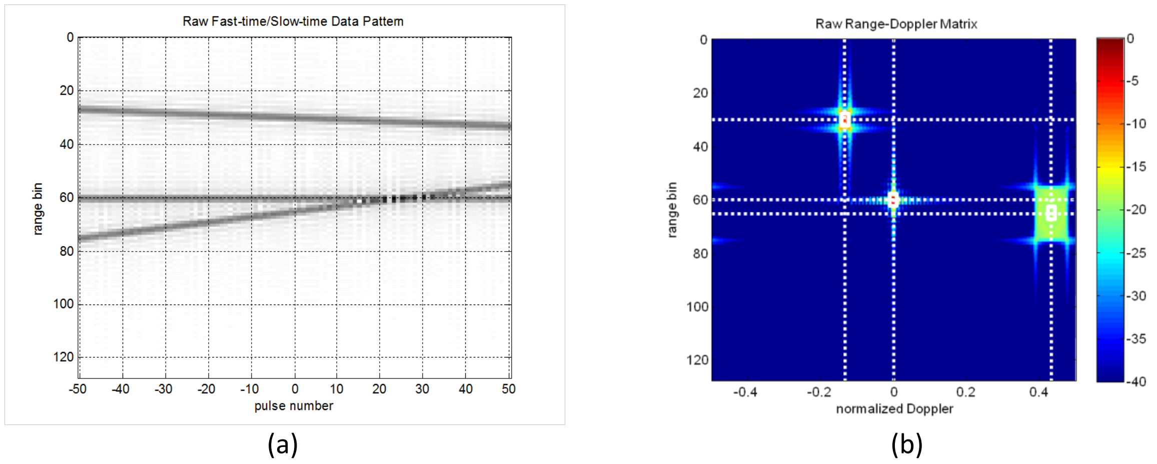


**Figure 6. Range-Doppler matrix with multiple targets at different velocities. (a) Range vs. pulse number. (b) Zoom into non-zero portion of the resulting range-Doppler matrix.**

Figure 7 shows the results of applying the keystone transformation to this example. Figure 7a shows that each target's signature has been realigned to a single range bin. Figure 7b shows the resulting range-Doppler spectrum, now having nearly full resolution in both dimensions on all three targets, despite their differing velocities.

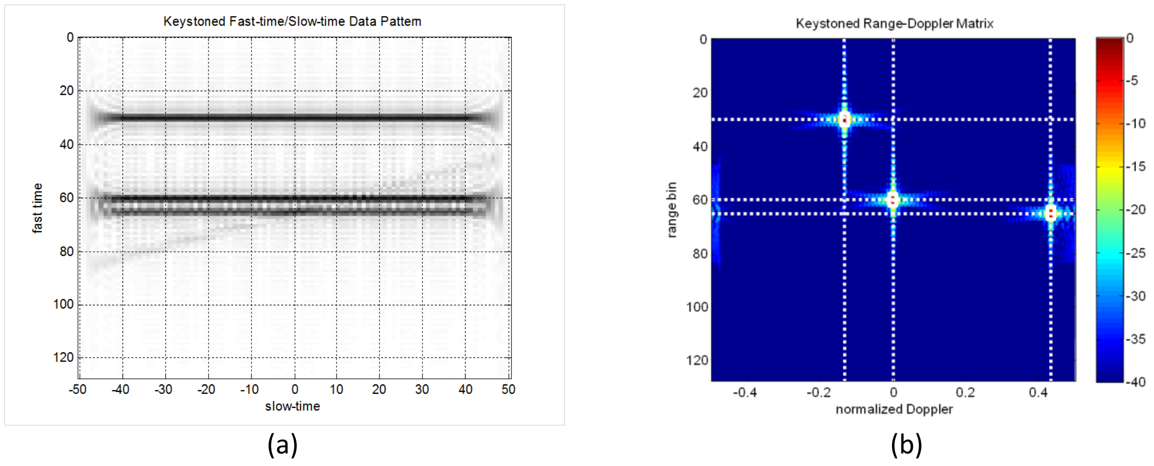


**Figure 7. Result of applying the keystone transformation to example of Fig. 6. (a) Range vs. pulse number. (b) Zoom into non-zero portion of the resulting range-Doppler matrix.**

## 5 Region of Support

The region of support (ROS) of $Y_{Rd\_key}(F,m]$ in fast-time frequency is the same as that of the original data $Y_{Rd}(F,m]$, namely $(-B/2,+B/2)$. The ROS in slow time varies with fast-time frequency. Specifically, the resampling operation linearly stretches or compresses the slow-time data sequence in each fast-time range bin by the factor $\left(F_0/(F_0+F)\right)$. At the nominal radar frequency $F_0$ $(F=0)$ the slow-time axis is unchanged; $Y_{Rd\_key}(F,m]$ represents samples of $Y_{Rd}(F,\tau)$ taken at the locations $T_{st}m$. For $F>0$, $Y_{Rd\_key}(F,m]$ represents samples of $Y_{Rd}(F,\tau)$ taken at the more closely-spaced locations $\left(F_0/(F_0+F)\right)m$; these data samples are then interpreted as slow-time samples at intervals of $T_{st}$. in $Y_{Rd\_key}(F,m]$. This has the effect of stretching the slow-time axis for $F>0$. Similarly, the slow-time axis is compressed for $F<0$.

In terms of the continuous slow-time variable $\tau$, the slow-time ROS of the original data is $\pm M_a T_{st}$. Because the interpolated sample spacing varies from $T_{st}$, the maximum integer value of $m$ in the interpolated data that will not exceed the support in $\tau$ of the original data varies with fast time frequency $F$ and is given by

$$\left\lceil -\left(\frac{F_0+F}{F_0}\right)M_a \right\rceil \le m \le \left\lfloor +\left(\frac{F_0+F}{F_0}\right)M_a \right\rfloor \tag{14}$$

The floor and ceiling functions are needed to restrict $m$ to an integer range. For $F>0$ Eq. (14) states that this range will be greater than the original full range of $\pm M_a$, because the samples in the $\tau$ dimension are closer together. For $F<0$ the range will be less than $\pm M_a$.

In addition, the range of $m$ may be reduced by edge effects of the interpolation process, depending on the specific technique used. If the bandlimited interpolation is implemented using an interpolation kernel of odd length $N_s = 2N_0+1$, as would typically be the case, the maximum range of $m$ for which the interpolation filter will not run off the end of the original data is

$$-\left\lceil \frac{M_a}{\left(\left(F_0+F\right)/F_0\right)} - N_s \right\rceil \le m \le +\left\lfloor \frac{M_a}{\left(\left(F_0+F\right)/F_0\right)} - N_s \right\rfloor \tag{15}$$

Thus, the keystone interpolation process may entail a small loss of slow-time support and therefore a small loss of Doppler resolution that increases with interpolating filter size and must therefore be traded off against the interpolation quality.

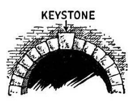
## 6 Why "Keystone"?

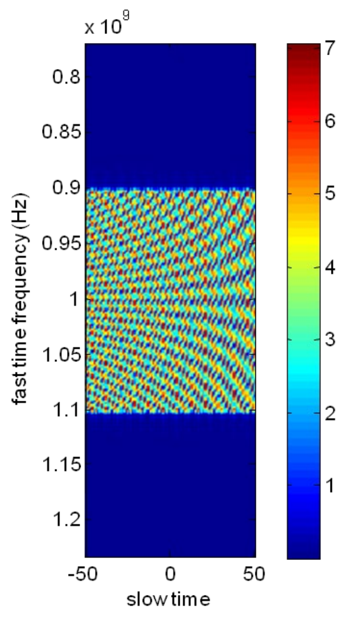
Why is the process of resampling $Y_{Rd}\left(F,m\right]$ to obtain $Y_{Rd\_key}\left(F,m\right]$ called a "keystone" transformation?[7] The reason is the frequency-variant expansion and contraction of the slow-tie axis discussed above. This is illustrated in Fig. 8. Part (a) of the figure shows $\left|Y_{Rd}\left(F,m\right]\right|$, the magnitude of the range frequency/slow time spectrum obtained by taking the DFT of the raw fast-time/slow-time data in the fast-time dimension only. The parameters are those of the three-target example used above. Notice that the spectrum support in fast time is 200 MHz wide, centered on the 1 GHz RF. Part (b) of the figure shows the magnitude of the modified spectrum $\left|Y_{Rd\_key}\left(F,m\right]\right|$ resulting from the slow-time interpolation process. Obviously, the spectral support region is no longer a rectangle but is now a keystone shape.

## 7 Effect of Doppler Ambiguity

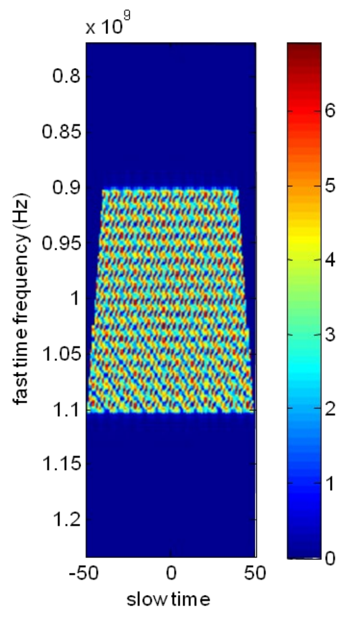
In all of the examples considered here, the Doppler frequencies of all targets were within the unambiguous Doppler interval for data sampled in slow time at an interval of $T_{st}$ seconds, namely $1/T_{st}$ Hz. Figure 9 shows a single-target case where this is not true. All parameters are the same as in the single-target case used earlier, except that the RF is increased by 10× to 10 GHz, thus increasing the Doppler frequency by 10× to $F_D$ = 29.33 kHz. The normalized Doppler shift at 10 GHz is then $f_D$ = 2.933 cycles/sample, which aliases at the 10 kHz PRF to $f_{Dn}$ = −0.0667 cycles/sample = $f_D - 3$. The number of "wraps" of the Doppler (three in this case) is called the Doppler *ambiguity number* $N_{amb}$ and is given by $N_{amb} = f_D - f_{Dn}$. $N_{amb}$ can be positive or negative. The raw fast-time/slow-time data and its range-Doppler spectrum are essentially identical to those of Fig. 3, except that the range-Doppler response is now centered at −0.0667 cycles/sample. Figure 9 shows that the keystone transformation fails to correct the range migration in this case.

[7] A keystone is the stone at the top of a masonry arch that locks all of the arch stones in place, allowing the arch to be self supporting and bear weight.



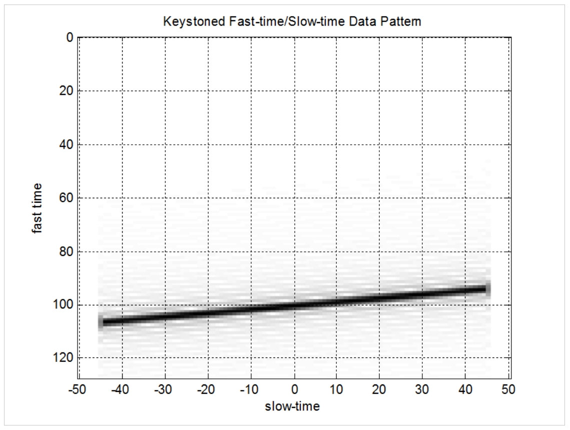


(a)

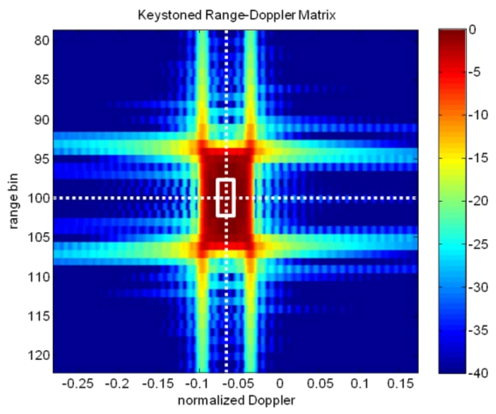


(b)

**Figure 8. Why it's called a keystone transformation. (a) Magnitude of the range frequency/slow-time spectrum of the raw data. (b) Magnitude of the range frequency/slow-time spectrum after keystone interpolation.**



(a)



(b)

**Figure 9. Result of applying the keystone transformation to Doppler-aliased target. See text for parameters. (a) Range vs. pulse number. (b) Zoom into non-zero portion of the resulting range-Doppler matrix.**

To understand and correct this phenomenon, begin by recalling the form of the slow-time data from Eq. (10):

$$\exp\left[+j\frac{4\pi}{c}\left(F_0+F\right)vT_{st}m\right], \qquad -M_a < m < +M_a \tag{16}$$

We assume $v$ is such that the corresponding Doppler shift will be aliased at the PRI $T_{st}$. Consequently, the bandlimited interpolation process will not produce the slow-time function $\exp\left[+j\frac{4\pi}{c}\left(F_0+F\right)v\tau\right]$, but rather a function with a value of velocity, call it $v_F$, that will produce the aliased Doppler shift. That value is

$$v_F = v - \frac{cN_{amb}}{2\left(F_0+F\right)T_{st}} \tag{17}$$

A key modeling detail is that $v_F$ is computed using the actual fast-time frequency in each frequency bin, $F_0+F$, rather than just $F_0$. As a result, $v_F$ is different in each fast-time frequency bin; hence the "$F$" subscript.[8]

The continuous slow-time spectrum reconstructed by the interpolation will therefore be of the form

$$\exp\left[+j\frac{4\pi}{c}\left(F_0+F\right)\left(v-\frac{cN_{amb}}{2\left(F_0+F\right)T_{st}}\right)\tau\right] \tag{18}$$

Applying the keystone transformation by sampling this function at $\tau = \left[F_0/\left(F_0+F\right)\right]T_{st}m$ gives

$$\begin{aligned}&\exp\left(+j\frac{4\pi}{c}F_0vT_{st}m\right)\exp\left[-j2\pi\left(\frac{F_0}{F_0+F}\right)N_{amb}m\right]\\&=\exp\left(+j\frac{4\pi}{c}F_0v_0T_{st}m\right)\exp\left[-j2\pi\left(\frac{F_0}{F_0+F}\right)N_{amb}m\right]\end{aligned} \tag{19}$$

The substitution of $v_0$ ($v_F$ at $F$ = 0) for $v$ in the second line occurs because that term is a pure sinusoid at frequency $F_0$ sampled at intervals $T_{st}$; the differential frequency $F$ has been removed. Consequently, $v$ will alias to $v_0$ in this term.

The first term in the second line of Eq. (19) is the desired result. The second term can be removed by multiplying the data by the slow-time phase compensation term $\exp\left[+j2\pi\left(F_0/\left(F_0+F\right)\right)N_{amb}m\right]$ after the interpolation. The keystone transformation when there is Doppler ambiguity therefore becomes

[8] Another modeling detail is that we assume that $N_{amb}$ is the same in all fast-time frequency bins for simplicity. A target whose nominal Doppler shift at the band center $F_0$ is at the edge of the Doppler spectrum (very close to $\pm PRF/2$) could have different values of $N_{amb}$ for positive and negative differential frequency $F$. We do not deal with this case here.

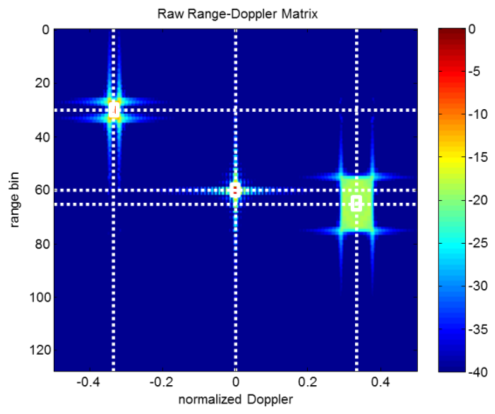
$$\begin{aligned}\hat{Y}_{Rd\_key}(F,m) &= \exp\left[+j2\pi\left(\frac{F_0}{F_0+F}\right)N_{amb}m\right]Y_{Rd\_key}(F,m) \\ &= \exp\left[+j2\pi\left(\frac{F_0}{F_0+F}\right)N_{amb}m\right]Y_{Rd}\left(F,\left(\frac{F_0}{F_0+F}\right)m\right)\end{aligned} \tag{20}$$

The ambiguity correction requires that $N_{amb}$ be known, perhaps from tracking data or other sensors, and that it be the same for all targets. If there are multiple targets and the ambiguity numbers are not the same for all targets, only the target(s) having an ambiguity number that matches that used when applying Eq. (20) will be focused.

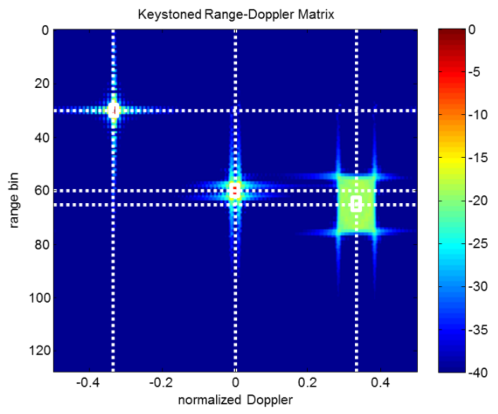
Figure 10 illustrates the effect of the ambiguous Doppler correction process. The same multitarget example used previously was repeated, but with the RF increased to 10 GHz. The resulting target ambiguity numbers are −1, 0, and +4. Figure 10a is the range-Doppler plot of the data before the keystone transformation and ambiguity compensation. Figure 10b shows the range-Doppler spectrum after the keystone transformation and the compensation of Eq. (20) using $N_{amb}$ = −1. The negative-Doppler target is well-focused. The zero-velocity target is slightly defocused because of processing with the incorrect ambiguity number, although that effect is not very visible in this image. The smearing of the positive-Doppler target is also slightly worsened. Figure 10c repeats the processing with $N_{amb}$ = +4, appropriate for the positive-Doppler target. That target is now well-focused, while the other two are severely smeared.

The varying amplitude of the target responses in Fig. 10 also shows that the amplitude of the range-Doppler signature of a target varies with the difference between the assumed and correct ambiguity numbers. If the ambiguity number is not known for a given target, it can be estimated by forming the corrected range-Doppler spectrum using several different values and choosing the one that produces the highest-amplitude peaks. Note that the interpolation process need only be carried out once; different ambiguity number phase corrections can then be tried on the same interpolated data.

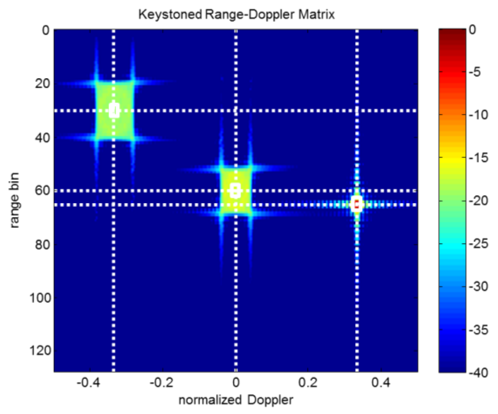
Figure 11 shows the peak amplitude in decibels of the positive-Doppler target as a function of $N_{amb}$. The maximum response is obtained when the correct value of $N_{amb}$ = 4 is used. Thus, processing the data with several different values of $N_{amb}$ can identify the correct value and thus also the true velocity.



(a)



(b)



(c)

**Figure 10. Result of applying keystone processing to Doppler-aliased target. See text for parameters. (a) Before keystone transformation. (b) After keystone transformation with $N_{amb}$ = −1. (c) After keystone transformation with $N_{amb}$ = +4.**

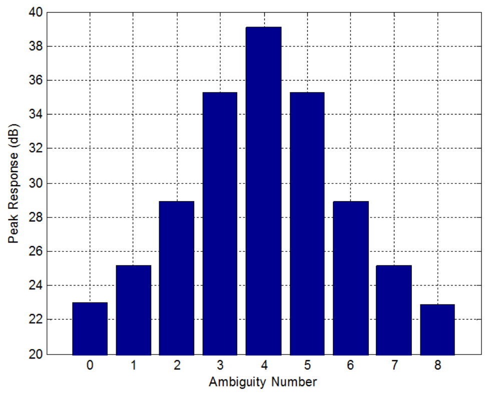


**Figure 11. Peak magnitude of the positive-Doppler target response from the example of Fig. 10 as a function of the ambiguity number used in the keystone transformation. The correct ambiguity number of $N_{amb}$ = 4 produces the largest peak response.**

This ambiguity compensation technique is discussed in both [1] and [2]. In [1], the phase compensation factor is of the same form (after adjusting for notation and a difference in sign in the definition of velocity) given in Eq. (20). In [2], it is given in our notation in the form $\exp\left[-j2\pi N_{amb}\left(F/F_0\right)m\right]$. The two forms can be reconciled by noting that, for small fractional bandwidths $(F \ll F_0)$, $F_0/(F+F_0)$ = $1/\left(1+\left(F/F_0\right)\right) \approx 1 - F/F_0$ . Substituting this approximation into Eq. (20) gives

$$\begin{aligned}\hat{Y}_{Rd\_key}(F,m] &= Y_{Rd\_key}(F,m]\exp\left(+j2\pi N_{amb}\left[1-\left(\frac{F}{F_0}\right)\right]m\right)\\ &= Y_{Rd\_key}(F,m]\exp\left(-j2\pi N_{amb}\left(\frac{F}{F_0}\right)m\right)\end{aligned} \tag{21}$$

The form of Eq. (20) is preferred because the fractional bandwidth may not be particularly small in low-RF, fine-resolution systems.

## 8 Commutativity of Keystone and Ambiguity Corrections

In the discussion above, the keystone migration correction was applied to the data first, followed by a phase compensation for ambiguity correction. It is remarked in [2] that this order can be reversed. Here, we briefly show how this can be done. We will find that process uses the same interpolation for range migration correction as before, but that the form of the phase compensation must be different, so that the reversal is not strictly a commutation of exactly the same two operations.

We again start with the slow-time phase term from Eq. (10), repeated here for convenience:

$$\exp\left[+j\frac{4\pi}{c}(F_0+F)vT_{st}m\right] \tag{22}$$

We assume the velocity is ambiguous with ambiguity number $N_{amb}$. Form the new phase-compensated slow-time function

$$\begin{aligned}&\exp\left[+j\frac{4\pi}{c}(F_0+F)vT_{st}m\right]\exp\left[-j2\pi N_{amb}\left(\frac{F_0+F}{F_0}\right)m\right]\\ &\quad=\exp\left[+j\frac{4\pi}{c}(F_0+F)vT_{st}m\right]\exp\left[-j\frac{4\pi}{c}(F_0+F)\left(\frac{cN_{amb}}{2F_0T_{st}}\right)T_{st}m\right]\\ &\quad=\exp\left[+j\frac{4\pi}{c}(F_0+F)\left(v-\frac{cN_{amb}}{2F_0T_{st}}\right)T_{st}m\right]\end{aligned} \tag{23}$$

Note that the term involving $N_{amb}$ has only $F_0$ in its denominator, not $(F_0 + F)$ as earlier. The term in parentheses is therefore $v_0$ specifically, not the more general $v_F$. Now applying the keystone transformation will effectively replace $T_{st}m$ with $\left(F_0/(F_0+F)\right)T_{st}m$, producing the slow-time function

$$\begin{aligned}\exp\left[+j\frac{4\pi}{c}(F_0+F)\left(v-\frac{cN_{amb}}{2F_0T_{st}}\right)T_{st}\left(\frac{F_0}{F_0+F}\right)m\right] &= \exp\left[+j\frac{4\pi}{c}F_0\left(v-\frac{cN_{amb}}{2F_0T_{st}}\right)T_{st}m\right] \\ &= \exp\left[+j\frac{4\pi}{c}F_0v_0T_{st}m\right]\end{aligned} \qquad (24)$$

This is the same result obtained with the interpolation-first ordering.

If a search over $N_{amb}$ is required, the interpolation would have to be repeated for each value of $N_{amb}$ considered, since the phase compensation comes first. For this reason, the interpolation-first order is preferred.

## 9 References

[1] Yang Li, Tao Zeng, Teng Long, and Zheng Wang, "Range Migration Compensation and Doppler Ambiguity Resolution by Keystone Transform", *Proceedings Intl. Conf. on Radar 2006* (CIE 2006), pp. 1 – 4, 2006.

[2] R. P. Perry, R. C. DiPietro, and R. Fante, "Coherent Integration With Range Migration Using Keystone Formatting", *Proceedings 2007 IEEE Radar Conference*, pp. 863 – 868, 2007.

[3] M. A. Richards, *Fundamentals of Radar Signal Processing*, second edition. McGraw-Hill, 2014.

[4] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, third edition. Prentice-Hall, 2009.

## 10 Demonstration MATLAB® Code

Following are the two scripts used to generate the examples in this memo. The script `keystone.m` was used to generate the single target examples; it can perform the keystoning and ambiguity correction operations in either order, demonstrating the commutativity discussed in Section 8. The order is controlled by a user setting at the beginning of the code. The examples in this memo were done with keystoning performed first, although the differences when the reverse order is used are indistinguishable to the eye. The script `keystone_multitarget` was used for the three-target example. Also included is the script `sinc_interp.m` used for the keystone interpolation. The scripts `hline.m` and `vline.m` used for some plot markings are available from the MATLAB® File Exchange at http://www.mathworks.com/matlabcentral/fileexchange/.

## keystone.m

```
% keystone
%
% Demo of keystone formatting for correcting range-Doppler
% measurements for range migration.
%
% This code closely follows the equations in the tech memo "The Keystone
% Transformation for Correcting Range Migration in Range-Doppler
% Processing" by Mark A. Richards, Mar. 2014, available at www.radarsp.com.
%
% Mark Richards
%
% March 2014
% Revised January 2020

clear all
close all

c = 3e8;  % speed of light

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  USER INPUT SECTION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
L = 128; % fast time dimension, samples
M = 101;  % slow-time dimension, samples; keep it odd.

% K_L and K_M must be even to avoid labeling problems later
K_L = 2^(nextpow2(128)+1); % fast-time DFT size for interpolation and shifting
K_M = 2^(nextpow2(512)+1); % slow-time DFT size

% Lref is the range bin # of the target, on a 0:L-1 scale, at the center of
% the CPI (middle pulse)
% Lref = round(L/2); % puts target at middle range bin on the middle pulse
Lref = 100;

F0 = 10000e6; % RF (Hz)
B = 200e6; % waveform bandwidth (Hz)

v = 440; % velocity in m/s towards the radar

% sampling intervals and rates
Fsft = 2.3*B;

PRF = 10e3;

% Order of sinc interpolating filter
Nsinc = 11;

% Order of interpolation and ambiguity correction ooperations. Set
% 'interp_first' to 0 to do ambiguity correction first, 1 to do keystoning
% first.
interp_first = 0;
if (interp_first)
    disp('Performing keystoning first, then ambiguity correction ...')
else
    disp('Performing ambiguity correction first, then keystoning ...')
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  END USER INPUT SECTION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Some derived parameters
m_end = (M-1)/2;
ms = (-m_end:m_end);  % slow time index labels
Fd = 2*v*F0/c;  % Doppler shift, Hz
Tft = 1/Fsft;  % fast time sampling interval
dr = c*Tft/2; % range bin spacing
Tst = 1/PRF;  % slow-time sampling interval (PRI)
Dfd = 1/M; % Rayleigh Doppler resolution in cycles/sample
Drb = (1/B)/Tft; % Rayleigh fast-time resolution in range bins

if (PRF < Fd/2)
    fprintf('\nWarning: PRF < Fd/2. PRF = %g, Fd = %g.\n',PRF,Fd)
end

% Compute and report total range migration over the CPI in range bins
RM = v*Tst/dr; % amount of range migration per pulse in range bins
RMtot = M*RM;  % total range migration over the dwell, in range bins
fprintf('\nTotal range migration = %g m = %g range bins.\n',RMtot*dr,RMtot)

if ( (floor(Lref-RMtot/2) < 0) || (ceil(Lref+RMtot/2) > L-1) )
    fprintf(['\nWarning: Target will migrate out of range.', ...
        'RMtot = %g range bins, Lref = %g, L = %g range bins.\n'],RMtot,Lref,L)
end

% Compute normalized Doppler frequency and wrap it
fd = Fd*Tst;
fdn = mod(fd + 0.5,1) - 0.5;  % alias it back into [-0.5,+0.5]
amb_num = round(fd - fdn); % number of Doppler wraps

% Define corners of a box centered on the expected target coordinates, and
% one Rayleigh width wide in each direction and each dimension (i.e., a
% null-to-null resolution box for a well-formed sinc spectrum). Will use
% this to draw such a resolution box on some of the figures.
L1 = Lref - Drb;
L2 = Lref + Drb;
fd1 = fdn - Dfd;
fd2 = fdn + Dfd;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create synthetic data. First compute pulse-to-pulse phase shift. Then set
% up a matrix.  Loop over pulses, computing current range. The range
% profile of the compressed data for a single pulse is assumed to be a sinc
% function with a zero spacing equal to 1/B seconds, and a phase shift of
% the usual -(4*pi*F0/c)*R, where R = Rref - v*Tst*m and m = pulse number.
% Don't worry about amplitude.  Also don't bother with the
% -(4*pi*F0/c)*Rref phase term, it is the same for all pulses.

del_phi = -4*pi*(F0/c)*v*Tst;  % pulse-to-pulse phase increment due to range change
y = zeros(L,M);

for m = 1:M
    mm = ms(m);  % counts from -(M-1/2) to +(M-1)/2
    y(:,m) = exp(-1i*del_phi*mm)*sinc( B*Tft*((0:L-1)-Lref+v*Tst*mm/dr) );
end

% Now examine the data.   Then compute the range-Doppler matrix with a
% slow-time DFT and look at that.
```

```
figure
imagesc(ms,0:L-1,abs(y))
grid
colormap(flipud(gray))
xlabel('pulse number')
ylabel('range bin')
title('Raw Fast-time/Slow-time Data Pattern')

Y_rD = fft(y,K_M,2);
Y_rD = fftshift(Y_rD,2);
Y_rD_dB = db(abs(Y_rD),'voltage');
Y_rD_dB = Y_rD_dB - max(Y_rD_dB(:)); % normalize to 0 dB max
Y_rD_dB(:) = max(-40,Y_rD_dB(:)); % limit dynamic range for plot purposes

fD = (-K_M/2:K_M/2-1)/K_M;  % this only works correctly if K_M is even

% figure
% mesh(fD,0:L-1,Y_rD_dB)
% xlabel('normalized Doppler')
% ylabel('range bin')
% title('Raw Range-Doppler Matrix')

figure
imagesc(fD,0:L-1,Y_rD_dB)
hline(Lref,':w'); vline(fdn,':w')  % mark the correct spectrum center
line([fd1 fd1 fd2 fd2 fd1],[L1 L2 L2 L1 L1],'Color','w','LineWidth',2) % resolution
box
title(['Rng Migration = ',num2str(RMtot),...
    ' bins over CPI; Normalized Doppler = ',num2str(fdn),' cyc/samp'])
xlabel('normalized Doppler')
ylabel('range bin')
colorbar
shg

% It is convenient to look at the fast time DFT of the raw data as well. We
% will need this product as the starting point for keystoning a little
% further down. Apply a fast-time DFT. fftshift in fast-time freq dimension
% to center the origin. This will be the frequency corresponding to F0.
% Also work out axis label in Hz.
Y_Rd = fftshift(fft(y,K_L,1),1);

Fl = (-K_L/2:K_L/2-1)/K_L*Fsft;

figure
subplot(1,2,1)
imagesc(ms,F0+Fl,abs(Y_Rd))
xlabel('slow time')
ylabel('fast time frequency (Hz)')
title('Magnitude after fast-time FT')
colorbar

subplot(1,2,2)
imagesc(ms,F0+Fl,angle(Y_Rd))
xlabel('slow time')
ylabel('fast time frequency (Hz)')
title('Unwrapped phase after fast-time FT')
colorbar

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Now do compensation by explicit shifting and interpolation in range for
% each pulse to see that that does indeed work (for a single target with
% known velocity) and verify formulas
%
yc = zeros(size(y));  % this will be the compensated fast-time/slow-time data
wl = 2*pi*(-K_L/2:K_L/2-1)'/K_L;  % normalized fast time freq in radians/sample

for m = 1:M  % loop over pulses
    mm = ms(m);  % counts from -(M-1/2) to +(M-1)/2
    Lm = v*Tst*mm/dr;  % # of range bins of shift needed
    Y_shift = fftshift(fft(y(:,m),K_L)).*exp(-1i*wl*Lm);
    y_shift_temp = ifft(ifftshift(Y_shift));
    y_shift(:,m) = y_shift_temp(1:L);
end  % of loop over pulses

figure
imagesc(ms,0:L-1,abs(y_shift))
colormap(flipud(gray))
grid
xlabel('pulse number')
ylabel('range bin')
title('Data Pattern after Compensation by Shifting')

Y_shift = fft(y_shift,K_M,2);
Y_shift = fftshift(Y_shift,2);
Y_shift_dB = db(abs(Y_shift),'voltage');
Y_shift_dB = Y_shift_dB - max(Y_shift_dB(:)); % normalize to 0 dB max
Y_shift_dB(:) = max(-40,Y_shift_dB(:)); % limit to 40 dB range

% figure
% mesh(fD,0:L-1,Y_shift_dB)
% xlabel('normalized Doppler')
% ylabel('range bin')
% title('Range-Doppler Matrix after Compensation by Shifting')

figure
imagesc(fD,0:L-1,Y_shift_dB)
hline(Lref,':w'); vline(fdn,':w')  % mark the correct spectrum center
line([fd1 fd1 fd2 fd2 fd1],[L1 L2 L2 L1 L1],'Color','w','LineWidth',2) % resolution
box
xlabel('normalized Doppler')
ylabel('range bin')
title('Range-Doppler Matrix after Compensation by Shifting')
shg
colorbar

% Let's also look at the fast-time DFT of this data.  It will show what
% we're trying to get to with the keystoning in the next section.

Y_Rd_shift = fftshift(fft(y_shift,K_L,1),1);

figure
subplot(1,2,1)
imagesc(ms,F0+Fl,abs(Y_Rd_shift))
xlabel('slow time')
ylabel('fast time frequency (Hz)')
title('Magnitude after fast-time FT')
colorbar

subplot(1,2,2)
imagesc(ms,F0+Fl,angle(Y_Rd_shift))
xlabel('slow time')
ylabel('fast time frequency (Hz)')
title('Unwrapped phase after fast-time FT')
```

```
colorbar

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Now start over and correct by keystoning. Begin with Y_Rd, i.e. data
% DFT'ed in fast time but not in slow time.  We have two operations to do:
% keystoning proper, and ambiguity correction.  They can be done in either
% order, depending on the setting of the 'interp_first' variable in the
% user input section.

if (interp_first)  % Do keystoning first, then ambiguity correction

    % If we chose to do interpolation first, then for each fast-time frequency
    % bin, compute a new, interpolated slow-time sequence. Use the existing
    % sinc_interp function for bandlimited, Hamming-weighted interpolation to
    % do the work.

    Y_Rd_key = zeros(size(Y_Rd));

    for k = 1:K_L
        [y_temp,mm_i] = sinc_interp(Y_Rd(k,:),ms,(F0/(F0+Fl(k)))*ms,Nsinc,1);
        %     y_temp = interp1(ms,Y_Rd(k,:),(F0/(F0+Fl(k)))*ms,'spline');
        % Mi will always be odd the way I'm setting up the problem. Also, Mi <= M.
        Mi = length(y_temp);
        dM = M - Mi;  % dM will be even so long as M and Mi are odd
        Y_Rd_key(k,1+dM/2:1+dM/2+Mi-1) = y_temp; % center the interpolated data in
slow time
    end

    % Now correct Y_Rd_key for the ambiguity number of the Doppler. This
    % code uses the ambiguity number of the first target. So if the other
    % targets have a different ambiguity number it won't be correct for
    % them. The first version of the correction corresponds to the Li et al
    % paper and is consistent with my memo. The second (commented out)
    % corresponds to the Perry et al paper and can be obtained from the
    % first using a binomial expansion approximation to (F0/(F0+Fl)).
    for mp = 1:M
        for k = 1:K_L
            mmp = ms(mp);  % counts from -(M-1/2) to +(M-1)/2
            Y_Rd_key(k,mp) = Y_Rd_key(k,mp)*exp(1i*2*pi*amb_num*mmp*(F0/(F0+Fl(k))));
%             Y_Rd_key(k,mp) = Y_Rd_key(k,mp)*exp(-1i*2*pi*amb_num*mmp*(Fl(k)/F0));
        end
    end   % end of interpolation-first implementation

else % Do ambiguity correction first, then keystoning

    % First correct the modified spectrum for the ambiguity number of the
    % Doppler. Note that the phase function used to correct ambiguity is
    % different if ambiguity correction is performed first. This code uses the
    % ambiguity number of the first target. So if the other targets have a
    % different ambiguity number it won't be correct for them. This version of
    % the correction corresponds to the approach of the Li et al paper and is
    % consistent with my memo.

    for mp = 1:M
        for k = 1:K_L
            mmp = ms(mp);  % counts from -(M-1/2) to +(M-1)/2
            Y_Rd(k,mp) = Y_Rd(k,mp)*exp(-1i*2*pi*amb_num*mmp*((F0+Fl(k))/F0));
%             Y_Rd(k,mp) = Y_Rd(k,mp)*exp(-1i*2*pi*amb_num*mmp*(1+(Fl(k)/F0)));
        end
    end

    % Y_Rd is DFT'ed in fast time but not in slow time. For each fast-time
    % frequency bin, compute a new, interpolated slow-time sequence. Use the
    % existing sinc_interp function for bandlimited, Hamming-weighted
    % interpolation to do the work.

    Y_Rd_key = zeros(size(Y_Rd));

    for k = 1:K_L
        [y_temp,mm_i] = sinc_interp(Y_Rd(k,:),ms,(F0/(F0+Fl(k)))*ms,Nsinc,1);
        %      y_temp = interp1(ms,Y_Rd(k,:),(F0/(F0+Fl(k)))*ms,'spline');
        % Mi will always be odd the way I'm setting up the problem. Also, Mi <= M.
        Mi = length(y_temp);
        dM = M - Mi;  % dM will be even so long as M and Mi are odd
        Y_Rd_key(k,1+dM/2:1+dM/2+Mi-1) = y_temp; % center the interpolated data in
slow time

    end  % end of ambiguity-correction-first implementation

end  % end of processing, both keystoning and ambiguity correction

% Now IDFT in fast-time and DFT in slow time to get range-Doppler matrix
y_temp_key = ifft( ifftshift(Y_Rd_key,1),K_L,1 );
y_rd_key = y_temp_key(1:L,:);
Y_rD_key = fftshift( fft(y_rd_key,K_M,2),2 );

Y_rD_key_dB = db(abs(Y_rD_key),'voltage');
Y_rD_key_dB = Y_rD_key_dB - max(Y_rD_key_dB(:)); % normalize to 0 dB max
Y_rD_key_dB(:) = max(-40,Y_rD_key_dB(:)); % limit to 40 dB range

figure
imagesc(ms,0:L-1,abs(y_rd_key))
grid
colormap(flipud(gray))
xlabel('slow-time')
ylabel('fast time')
title('Keystoned Fast-time/Slow-time Data Pattern')

% figure
% mesh(fD,0:L-1,Y_rD_key_dB)
% xlabel('normalized Doppler')
% ylabel('fast time')
% title('Keystoned Range-Doppler Matrix')

figure
imagesc(fD,0:L-1,Y_rD_key_dB)
hline(Lref,':w'); vline(fdn,':w')  % mark the correct spectrum center
line([fd1 fd1 fd2 fd2 fd1],[L1 L2 L2 L1 L1],'Color','w','LineWidth',2) % resolution
box
xlabel('normalized Doppler')
ylabel('range bin')
title('Keystoned Range-Doppler Matrix')
shg
colorbar

figure
subplot(1,2,1)
imagesc(ms,F0+Fl,abs(Y_Rd_key))
xlabel('slow time')
ylabel('fast time frequency (Hz)')
title('Magnitude after interpolation')
colorbar

subplot(1,2,2)
```

```
imagesc(ms,F0+Fl,angle(Y_Rd_key))
xlabel('slow time')
ylabel('fast time frequency (Hz)')
title('Unwrapped phase after interpolation')
colorbar
```

## keystone_multitarget.m

```
% keystone_multitarget
%
% Demo of keystone formatting for correcting range-Doppler
% measurements for range migration, with multiple targets at different
% speeds.
%
% This code closely follows the equations in the tech memo "The Keystone
% Transformation for Correcting Range Migration in Range-Doppler
% Processing" by Mark A. Richards, Mar. 2014, available at www.radarsp.com.
%
% Mark Richards
%
% March 2014

clear all
close all

c = 3e8;  % speed of light

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  USER INPUT SECTION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
L = 128; % fast time dimension, samples
M = 101;  % slow-time dimension, samples; keep it odd.

% K_L and K_M must be even to avoid labeling problems later
K_L = 2^(nextpow2(128)+1); % fast-time DFT size for interpolation and shifting
K_M = 2^(nextpow2(512)+1); % slow-time DFT size

Ntgt = 3;
v = [-200,0,650]; % velocities in m/s towards the radar
% Lref is the range bin # of the targets, on a 0:L-1 scale, at the center of
% the CPI (middle pulse)
% Lref = round(L/2); % puts target at middle range bin on the middle pulse
Lref = [30,60,65];

F0 = 10000e6; % RF (Hz)
B = 200e6; % waveform bandwidth (Hz)

% sampling intervals and rates
Fsft = 2.3*B;

PRF = 10e3;

% Order of sinc interpolating filter
Nsinc = 11;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  END USER INPUT SECTION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Some derived parameters
```

```
m_end = (M-1)/2;
ms = (-m_end:m_end);  % slow time index labels
Fd = 2*v*F0/c;  % Doppler shifts, Hz
Tft = 1/Fsft;  % fast time sampling interval
dr = c*Tft/2; % range bin spacing
Tst = 1/PRF;  % slow-time sampling interval (PRI)
Dfd = 1/M; % Rayleigh Doppler resolution in cycles/sample
Drb = (1/B)/Tft; % Rayleigh fast-time resolution in range bins

if (PRF < Fd/2)
    fprintf('\nWarning: PRF < Fd/2. PRF = %g, Fd = %g.\n',PRF,Fd)
end

% Compute and report total range migration over the CPI in range bins
RM = v*Tst/dr; % amount of range migration per pulse in range bins
RMtot = M*RM  % total range migration over the dwell, in range bins

% Compute normalized Doppler frequencies and wrap them
fd = Fd*Tst
fdn = mod(fd + 0.5,1) - 0.5  % alias back into [-0.5,+0.5]
amb_num = round(fd - fdn); % number of Doppler wraps

% Define corners of a box centered on the expected target coordinates, and
% one Rayleigh width wide in each direction and each dimension (i.e., a
% null-to-null resolution box for a well-formed sinc spectrum). Will use
% this to draw such a resolution box on some of the figures.
L1 = Lref - Drb;
L2 = Lref + Drb;
fd1 = fdn - Dfd;
fd2 = fdn + Dfd;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create synthetic data. First compute pulse-to-pulse phase shift. Then set
% up a matrix.  Loop over pulses, computing current range. The range
% profile of the compressed data for a single pulse is assumed to be a sinc
% function with a zero spacing equal to 1/B seconds, and a phase shift of
% the usual -(4*pi*F0/c)*R, where R = Rref - v*Tst*m and m = pulse number.
% Don't worry about amplitude.  Also don't bother with the
% -(4*pi*F0/c)*Rref phase term, it is the same for all pulses.

y = zeros(L,M);

for t = 1:Ntgt
    del_phi = -4*pi*(F0/c)*v(t)*Tst;  % pulse-to-pulse phase increment due to range
change

    for m = 1:M
        mm = ms(m);  % counts from -(M-1/2) to +(M-1)/2
        y(:,m) = y(:,m) + exp(-1i*del_phi*mm)*sinc( B*Tft*((0:L-1)'-
Lref(t)+v(t)*Tst*mm/dr) );
    end

end % of loop over targets

% Now examine the data.   Then compute the range-Doppler matrix with a
% slow-time DFT and look at that.

figure
imagesc(ms,0:L-1,abs(y))
grid
colormap(flipud(gray))
xlabel('pulse number')
```

```
ylabel('range bin')
title('Raw Fast-time/Slow-time Data Pattern')

Y_rD = fft(y,K_M,2);
Y_rD = fftshift(Y_rD,2);
Y_rD_dB = db(abs(Y_rD),'voltage');
Y_rD_dB = Y_rD_dB - max(Y_rD_dB(:)); % normalize to 0 dB max
Y_rD_dB(:) = max(-40,Y_rD_dB(:)); % limit dynamic range for plot purposes

fD = (-K_M/2:K_M/2-1)/K_M;  % this only works correctly if K_M is even

% figure
% mesh(fD,0:L-1,Y_rD_dB)
% xlabel('normalized Doppler')
% ylabel('range bin')
% title('Raw Range-Doppler Matrix')

figure
imagesc(fD,0:L-1,Y_rD_dB)
title('Raw Range-Doppler Matrix')
xlabel('normalized Doppler')
ylabel('range bin')
for t = 1:Ntgt
    hline(Lref(t),':w'); vline(fdn(t),':w')  % mark the correct spectrum center
    line([fd1(t) fd1(t) fd2(t) fd2(t) fd1(t)], ...
        [L1(t) L2(t) L2(t) L1(t) L1(t)],'Color','w','LineWidth',2) % resolution box
end
colorbar
shg

% It is convenient to look at the fast time DFT of the raw data as well. We
% will need this product as the starting point for keystoning a little
% further down. Apply a fast-time DFT. fftshift in fast-time freq dimension
% to center the origin. This will be the frequency corresponding to F0.
% Also work out axis label in Hz.
Y_Rd = fftshift(fft(y,K_L,1),1);

Fl = (-K_L/2:K_L/2-1)/K_L*Fsft;

figure
subplot(1,2,1)
imagesc(ms,F0+Fl,abs(Y_Rd))
xlabel('slow time')
ylabel('fast time frequency (Hz)')
title('Magnitude after fast-time FT')
colorbar

subplot(1,2,2)
imagesc(ms,F0+Fl,angle(Y_Rd))
xlabel('slow time')
ylabel('fast time frequency (Hz)')
title('Unwrapped phase after fast-time FT')
colorbar

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Now start over and correct by keystoning. Begin with Y_Rd, i.e. data
% DFT'ed in fast time but not in slow time. For each fast-time frequency
% bin, compute a new, interpolated slow-time sequence. Use the existing
% sinc_interp function for bandlimited, Hamming-weighted interpolation to
% do the work.

Y_Rd_key = zeros(size(Y_Rd));
```

```
for k = 1:K_L
    [y_temp,mm_i] = sinc_interp(Y_Rd(k,:),ms,(F0/(F0+Fl(k)))*ms,Nsinc,1);
    %     y_temp = interp1(ms,Y_Rd(k,:),(F0/(F0+Fl(k)))*ms,'spline');
    % Mi will always be odd the way I'm setting up the problem. Also, Mi <= M.
    Mi = length(y_temp);
    dM = M - Mi;  % dM will be even so long as M and Mi are odd
    Y_Rd_key(k,1+dM/2:1+dM/2+Mi-1) = y_temp; % center the interpolated data in slow
time
end

% Now correct the modified spectrum for the ambiguity number of the
% Doppler. This code uses the ambiguity number of the first target. So if
% the other targets have a different ambiguity number it won't be correct
% for them. The first version of the correction corresponds to the Li et al
% paper and is consistent with my memo. The second (commented out)
% corresponds to the Perry et al paper and can be obtained from the first
% using a binomial expansion approximation to (F0/(F0+Fl)). Either one
% works if done either after the keystone correction, as is done here, or
% before.
for mp = 1:M
    for k = 1:K_L
        mmp = ms(mp);  % counts from -(M-1/2) to +(M-1)/2
        Y_Rd_key(k,mp) = Y_Rd_key(k,mp)*exp(1i*2*pi*amb_num(1)*mmp*(F0/(F0+Fl(k))));
%         Y_Rd_key(k,mp) = Y_Rd_key(k,mp)*exp(-1i*2*pi*amb_num(1)*mmp*(Fl(k)/F0));
    end
end

% Now IDFT in fast-time and DFT in slow time to get range-Doppler matrix
y_temp_key = ifft( ifftshift(Y_Rd_key,1),K_L,1 );
y_rd_key = y_temp_key(1:L,:);
Y_rD_key = fftshift( fft(y_rd_key,K_M,2),2 );

Y_rD_key_dB = db(abs(Y_rD_key),'voltage');
Y_rD_key_dB = Y_rD_key_dB - max(Y_rD_key_dB(:)); % normalize to 0 dB max
Y_rD_key_dB(:) = max(-40,Y_rD_key_dB(:)); % limit to 40 dB range

figure
imagesc(ms,0:L-1,abs(y_rd_key))
grid
colormap(flipud(gray))
xlabel('slow-time')
ylabel('fast time')
title('Keystoned Fast-time/Slow-time Data Pattern')

% figure
% mesh(fD,0:L-1,Y_rD_key_dB)
% xlabel('normalized Doppler')
% ylabel('fast time')
% title('Keystoned Range-Doppler Matrix')

figure
imagesc(fD,0:L-1,Y_rD_key_dB)
xlabel('normalized Doppler')
ylabel('range bin')
title('Keystoned Range-Doppler Matrix')
for t = 1:Ntgt
    hline(Lref(t),':w'); vline(fdn(t),':w')  % mark the correct spectrum center
    line([fd1(t) fd1(t) fd2(t) fd2(t) fd1(t)],[L1(t) L2(t) L2(t) L1(t)
L1(t)],'Color','w','LineWidth',2) % resolution box
end
colorbar
```

```
shg

figure
subplot(1,2,1)
imagesc(ms,F0+Fl,abs(Y_Rd_key))
xlabel('slow time')
ylabel('fast time frequency (Hz)')
title('Magnitude after interpolation')
colorbar

subplot(1,2,2)
imagesc(ms,F0+Fl,angle(Y_Rd_key))
xlabel('slow time')
ylabel('fast time frequency (Hz)')
title('Unwrapped phase after interpolation')
colorbar
```

## sinc_interp.m

```
function  [out,x_out] = sinc_interp(in,x_in,x_new,N,win)
%
% sinc_interp
%
% Sinc-based (band-limited)interpolation.
%
% INPUTS
%    in = data sequence to be interpolated
%  x_in = vector of sample locations corresponding to data samples of 'in'.
%         Must be uniformly spaced at some interval dx_in.  Must be same
%         length as 'in'.
% x_new = vector of desired sample locations.  Must be uniformly spaced at
%         some interval dx_out.
%     N = order of interpolating sinc, in units of max(dx_in,dx_out).
%         Must be odd.
%   win = 1 if Hamming window applied to interpolation kernel, otherwise no
%         window.  (win=1 is recommended.)
%
% OUTPUTS
%   out = interpolated data sequence corresponding to sample locations in
%         x_out.
% x_out = sample locations of output vector.  This will be a subset of the
%         locations in x_new; relative span of x_new and x_in, and filter
%         end effects, may limit x_out to not include some of the values in
%         x_new.
%
% Mark A. Richards
% February 2007
%
if (mod(N,2) ~= 1)
    disp(' ')
    disp(' ** Error: sinc_interp : filter order not odd.')
    disp([' ** Filter order input = ',int2str(N)])
    disp(' ')
    return
end
Nhalf = (N-1)/2;

d_in = x_in(2) - x_in(1);
d_out = x_new(2) - x_new(1);
```

```
% Now figure out interpolating filter impulse response in continuous time.
% This will be a sinc function, possibly windowed. Bandwidth of the sinc
% LPF frequency response is based on the larger sampling interval of the
% two grids.  Specifically, the unwindowed impulse response is h(x) =
% sin(pi*x/del)/(pi*x) and dt = max(dt1,dt2).  To add to this, we specify
% how many sample increments we will go out on the tails, where 1 increment
% is dt; and then we also apply a hamming window of the same length.  The
% Hamming formula in continuous time is w(t) = 0.54 + 0.46*cos(pi*t/dt).
del = max(d_in,d_out);


% find the values within x_new that can be successfully interpolated from the
% values available in x_in, i.e. where end effects won't kill us.
% x_in
% x_new
% Nhalf
% x_new(1)-Nhalf*del
% x_in(1)
% x_new(end)+Nhalf*del
% x_in(end)

index = find( (x_new-Nhalf*del >= x_in(1) ) & ...
    (x_new+Nhalf*del <= x_in(end)) );
if (isempty(index))
    disp(' ')
    disp(' ** Error: sinc_interp : Requested output samples cannot be interpolated')
    disp(' ')
    return
end
x_out = x_new(index);
out = zeros(size(x_out));

% step through the output samples one at a time, interpolating a value for
% each one from the input samples.
for k = 1:length(x_out)
    % first find the span of the interpolating filter on the x axis
    x_current = x_out(k);
    x_low = x_current - Nhalf*del;
    x_high = x_current + Nhalf*del;
    % compute the *relative* position of each input sample within this span
    % compared to the current output sample location; these will be the
    % values at which the interpolating kernel filter response will be
    % needed.
    index_rel = find( (x_in >= x_low ) & ...
        (x_in <= x_high) );
    x_rel = x_in(index_rel) - x_current;

    % Now compute and apply the sinc weights.  First fix any spots where
    % x_rel = 0; these will cause the sinc function to be undefined.  Then
    % add in the window, if used, and apply to the data to compute the
    % output point.
    trouble = find(x_rel==0);
    if (~isempty(trouble))
        x_rel(trouble) = x_rel(trouble)+eps; % this will prevent division by zero
    end
    h = (sin(pi*x_rel/del)/pi./x_rel);
    w = ones(size(h));
    if (win == 1)
        w = 0.54 + 0.46*cos(pi*x_rel/del/(Nhalf+1));
        h = h.*w;
    end
    h = h/sum(h);
    out(k) = sum( in(index_rel).*h);
end

% % Diagnostic figures showing a sample of sinc kernel and Hamming weights,
% % original and interpolated waveforms, and spectra of same.
% figure
% stem(x_rel,[h;w]')
% figure
% plot(x_in,real(in));
% title('Continuous Sinc Interpolation')
% hold on
% plot(x_out,real(out),'r');
% hold off
%
% % plot before-and-after spectra for a quality check.  Note that different
% % sampling rates mean I need to use different frequency scales, and they
% % can't be normalized frequency.
% % figure
% Nfft = 2^(ceil(log2(max(length(x_out),length(x_in))))+2);
% X_in = d_in*fft(in,Nfft);
% f_in = (1/d_in)*((0:Nfft-1)/Nfft-0.5);
% X_out = d_out*fft(out,Nfft);
% f_out = (1/d_out)*((0:Nfft-1)/Nfft-0.5);
% % plot(f_in,abs(fftshift(X_in)))
% % title('Continuous Sinc Interpolation')
% % hold on
% % plot(f_out,abs(fftshift(X_out)),'r')
% % grid
% % hold off
% figure
% plot(f_in,db(abs(fftshift(X_in)),'voltage'))
% title('Continuous Sinc Interpolation')
% hold on
% plot(f_out,db(abs(fftshift(X_out)),'voltage'),'r')
% grid
% hold off
% pause
```