Some MATLAB Tutorials

Dr. Mark A. Richards
School of Electrical & Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia  30332-0250
mark.richards@ece.gatech.edu
LICENSE

• Permission to use, copy, modify and distribute, including the right to grant others rights to distribute at any tier, this software and its documentation for any purpose and without fee or royalty is hereby granted, provided that the following copyright and attribution and the disclaimer (see next two pages) are retained in ALL copies and derivative works of the software and documentation, including modifications that you make for internal use or for distribution.
COPYRIGHT AND ATTRIBUTION

• Except as otherwise noted, this software was developed by Dr. Mark A. Richards and/or Gregory Heim of the Georgia Institute of Technology, and was provided by the Georgia Institute of Technology as part of the professional education course “Fundamentals of Radar Signal Processing”, 2006 and subsequent offerings, or by Dr. Mark A. Richards to accompany the book Fundamentals of Radar Signal Processing (McGraw-Hill, New York, 2005).

• Copyright 2006-2009, Mark A. Richards and/or Gregory Heim. All Rights Reserved.
DISCLAIMER

• All software is provided "as is". It is intended for tutorial and demonstration use only and is provided as a convenience and courtesy to the user. No user support is available.

• The developer, the Georgia Institute of Technology, and the Distance Learning and Professional Education division of the Georgia Institute of Technology make absolutely no warranty of merchantability or fitness for any use for any particular purpose, or that the use of the software will not infringe on any third party patents, copyrights, trademarks, or other rights.
Software is available …
- On FRSP textbook support web site at www.radarsp.com
- On CD-ROM for FRSP professional education course ("short course")

Software is provided as a WinZip file titled FRSP_Demos.zip

Assumes MATLAB 6 and Signal Processing Toolbox
- mainly for hamming, db functions, but may be others
- Used with version 7.8 (R2009a)
Software contains the following GUI-based MATLAB demonstrations:
- RCS of Complex Targets
- LFM Pulse Compression
- Multi-PRF Blind Zone Calculation

... and the following non-GUI-based demonstrations
- Pulse Doppler Processing
- Detection Calculator
- Doppler Beam Sharpening
- Adaptive Beamformer

... and the following student project assignments
- Pulse Doppler Processing (similar to above, in project form)
- RCS (similar to above, in project form)
- LFM waveform and matched filtering (similar to above, in project form)
- Threshold detection
Software Installation

- Unzip FRSP_Demos.zip in the directory of your choice
- This will create a new directory called FRSP_Demos
- Start MATLAB and choose the Set Path command under the File menu
- Navigate to the FRSP_Demos directory and click the Add with Subfolders button
- Click the Save button, and then Close
- All of the FRSP demo software should now be in your MATLAB path
GUI-Based Demos
RCS: GUI-Based RCS of Complex Targets

- Angle variation of RCS of “complex target” composed of “many” scatterers
  - With or without “dominant scatterer”
- Reproduces this experiment:
  - Can vary # of scatterers, box size, presence/absence of dominant scatterer and relative strength
  - Compare to various pdfs
  - Compute autocorrelation lengths

- > RCS
  - Select/change target characteristics, then click on ‘New Scatterers’
  - Select/change plot options, then click on ‘Compute RCS and Plot Results’
LFM: GUI-Based LFM Waveform Characteristics

• Generates and analyzes LFM waveforms and corresponding matched filter outputs
  – Can vary pulse duration, bandwidth, sampling rate
  – Effects of windows
  – Two-scatterer resolution

• **LFM**
  – Select/change waveform, window, plot characteristics
    • Plots should update automatically
  – Click on ‘View Additional Plots’ for two-scatterer and ambiguity plots
    • Caution: ambiguity function can be VERY slow and/or exhaust memory!
BlindZone: GUI-Based Multi-PRF Blind Zone Calculator

- Computes range-Doppler blind zones for multiple PRIs and “M-of-N” detection
  - Can vary # and value of PRIs, and the threshold \( M \) for M-of-N detection
  - Allows for clutter spectrum width, near-in clutter eclipsing

- >> BlindZone
  - Enter PRIs and detection threshold
  - Other parameters as desired
  - Click “Plot Blind Zones”
Non-GUI-Based Demos

After Range Compression, Before Cross-Range Compression

RANGE-DOPPLER PLOT OF UNPROCESSED DATA

Adapted Pattern, Post-DFT Beamformer

RANGE-DOPPLER PLOT OF UNPROCESSED DATA
Detection Calculator \texttt{Pd\_calc}

- Detection calculator that computes $P_d$ for non-coherent square-law integration of Swerling target models in Gaussian I/Q noise with a square-law detector
  - Computes the threshold and probability of detection for the four standard Swerling cases and the nonfluctuating case
  - Also included is the Albersheim approximation to the nonfluctuating case.
- To use: write a main program to call \texttt{Pd.m}
  - See examples on next two slides
- Based on \texttt{meyerfun.m}, version 1.2 (by Douglas Dougherty, NSWC DD Code T45, 4/14/99)
  - \texttt{meyerfun.m} and a controlling GUI, \texttt{meyer.m}, available on MathWorks’ MATLAB Central file exchange, 
    \url{http://www.mathworks.com/matlabcentral/fileexchange/1569}
- Modified significantly
  - Additional cases, vectorization, numerical issues
Example Calling Program #1: 
\texttt{Pd\_as\_a\_function\_of\_N.m}

\% M file for computing a figure to illustrate the effect of 
\% noncoherent integration on Pd vs. SNR for Swerling 0, 
\% for Pfa = 1e-8, SNR from -2 to +15 dB, and N=1 to 100. 
\% 
\% Mark Richards 
\% July 2002 

SNR\_dB = linspace(0,15); 
Pfa = 1e-8*ones(size(SNR\_dB));

\% Step through the cases: 

Pd0 = Pd(1*ones(size(SNR\_dB)),Pfa,SNR\_dB,0); 
Pd1 = Pd(2*ones(size(SNR\_dB)),Pfa,SNR\_dB,0); 
Pd2 = Pd(5*ones(size(SNR\_dB)),Pfa,SNR\_dB,0); 
Pd3 = Pd(10*ones(size(SNR\_dB)),Pfa,SNR\_dB,0); 
Pd4 = Pd(20*ones(size(SNR\_dB)),Pfa,SNR\_dB,0); 

\% OK, now draw the results 

plot(SNR\_dB,[Pd0; Pd1; Pd2; Pd3; Pd4]) 
axis([0,15,0,1]); 
xlabel('SNR (dB)'); ylabel('Pd'); grid; 
legend('N=1','N=2','N=5','N=10','N=20','Location','SouthEast');
Example Calling Program #1: Swerling_compare.m

% M file for computing a figure to compare the 5 swerling cases + Albersheim
% for N=10 pulses, Pfa = 1e-8, and SNR from -2 to +15 dB
%
% Mark Richards
% July 2002

SNR_dB = linspace(-2,15);
Pfa = 1e-8*ones(size(SNR_dB));
N = 10*ones(size(SNR_dB));

% Step through the cases:
Pd0 = Pd(N,Pfa,SNR_dB,0); % nonfluctuating; also called Swerling 0 or 5 in some cases
Pd1 = Pd(N,Pfa,SNR_dB,1);
Pd2 = Pd(N,Pfa,SNR_dB,2);
Pd3 = Pd(N,Pfa,SNR_dB,3);
Pd4 = Pd(N,Pfa,SNR_dB,4);
Pd6 = Pd(N,Pfa,SNR_dB,6); % Albersheim's equation

% OK, now draw the results
plot(SNR_dB,[Pd0; Pd1; Pd2; Pd3; Pd4; Pd6])
axis([-2,15,0,1]);
xlabel("SNR (dB)"); ylabel("Pd"); grid;
legend('Nonfluctuating','Swerling 1','Swerling 2','Swerling 3', 'Swerling 4','Albersheim','Location','Southeast');
Pulse Doppler Processing Demonstration

• Formation of a fast-time/slow-time (range/pulse #) data matrix for moving targets in noise and clutter
  – LFM chirp waveform

• Pulse Doppler processing for target detection; and range, velocity, and relative RCS estimation
  – Formation of range-Doppler matrix
  – Pulse compression
  – MTI filter compensation
  – \( R^4 \) correction
  – Threshold detection
  – Peak interpolation
Pulse Doppler Processing Procedure

• Two stages
  – makePDdata creates a fast-time/slow-time data matrix that will support the desired scenario
  – procPDdata performs the processing

• Stage 1: Data Creation
  – >> edit makePDdata
    • Set all simulation parameters by editing input section of makePDdata.m
  – >> makePDdata
    • To create data set for processing
    • Output in file.mat
      – Where file is the “root file name” you specify
      – Parameters logged in file.lis

• Stage 2: Processing
  – >> procPDdata
Pulse Doppler Processing Inputs

• `makePDdata` user input section:

```matlab
% User Input Section #####################################################################
% #####################################################################

file = input('Enter root file name for data and listing files: ','s');
T = 10e-6;  % pulse length, seconds
W = 10e6;   % chirp bandwidth, Hz
fs = 12e6;  % chirp sampling rate, Hz; oversample by a little

Np = 20;    % # of pulses
jkI = 0:(Np-1);  % pulse index array
PRF = 25.0e3; % PRF in Hz
PRI = (1/PRF); % PRI in sec
T_0 = PRI*jkI; % relative start times of pulses, in sec
g = ones(1,Np); % gains of pulses
T_out = [12 38]*1e-6; % start and end times of range window in sec
T_ref = 0;  % system reference time in usec
fc = 10e9;  % RF frequency in Hz; 10 GHz is X-band

% Compute unambiguous Doppler interval in m/sec
% Compute unambiguous range interval in meters
vua = 3e8*PRF/(2*fc);
rmmin = 3e8*T_out(1)/2;
rmax = 3e8*T_out(2)/2;
rua = rmax-rmin;

% Define number of targets, then range, amplitude, and radial velocity of each
Ntargets = 4;
del_R = (3e8/2)*(1/fs)/1e3;  % in km
ranges = [2.3 2.7 3.1 3.5]*1e3;  % in km
SNR = [10 20 15 20];  % dB
vels = [-0.3 -0.15 +0.1 +0.25]*vua;  % in m/sec

% End User Input Section #####################################################################
% #####################################################################

Fundamentals of Radar Signal Processing
© 2009 Mark A. Richards, All Rights Reserved
Pulse Doppler Processing Outputs - 1

Noncoherently Integrated Range Trace

Range-Doppler Plot of Unprocessed Data

Range-Doppler Contour Plot of Unprocessed Data
Pulse Doppler Processing Outputs - 2
Pulse Doppler Processing Outputs - 3
Doppler Beam Sharpening Imaging - 1

• Closely related to the pulse Doppler demonstration and project
• Imaging of a user-specified array of point scatterers
• Two stages
  – `makeSARdata_DBS` creates a fast-time/slow-time data matrix that will support the desired scenario
  – `procSARdata_DBS` performs the DBS imaging algorithm

• Stage 1: Data Creation
  • `>> edit makeSARdata_DBS`
    – Set all simulation parameters by editing input section of `makeSARdata_DBS.m`
  • `>> makeSARdata_DBS`
    – To create data set for processing
    – Output in `file.mat`
      • Where `file` is the “root file name” you specify
      • Parameters logged in `file.lis`
Doppler Beam Sharpening Imaging - 2

- **Stage 2: Image Formation**

- `>> edit procSARdata_DBS`
  - Set image formation option switches by editing input section of `procSARdata_DBS.m`

- `>> procSARdata_DBS` to generate DBS image
  - Range-compressed only in Fig. 1 window
  - Fully-formed image in Fig. 2 window
  - If geometric corrections selected, then
    - Cross-range resampled image in Fig. 3 window
    - Range curvature-corrected image in Fig. 4 window
% User input section

% need a file name to store data, e.g. 'myfile' or 'temp' or 'sardata'.
% Data will then be in 'myfile.dat', etc.
file=input('Enter root file name for data and listing files: ','s');

DCR = 20;           % cross-range resolution, m
DR = 20;             % range resolution, m
Rcrp = 40000;       % range to swath center
v = 150;             % platform velocity, m/s
fc = 10e9;           % RF frequency in Hz
lambda = c/fc;       % wavelength, m
tau = 5e-6;          % pulse length, seconds (bandwidth set by resolution)
Daz = 0.2;           % antenna azimuth size, m
thetaaz = lambda/Daz; % azimuth beamwidth, radians
BWdopp = 2*v*lambda*thetaaz; % slow time Doppler bandwidth, Hz
Ls = 3000;           % swath depth, m
oversample_st = 2;   % slow time oversample factor; higher makes prettier pictures but larger data sets
oversample_ft = 2;   % fast time oversample factor, similar to slow time

% Define target locations, one row of (x,R) coordinates per target
% Ranges are relative to the CRP range (Rcrp) above.
coords = [0,0];       % a single point target at the CRP
coords = ...          % a grid of 9 point targets
  [-1000,-1000;       %
  -1000,0;            %
  -1000,+1000;        %
  0,-1000;            %
  0,0;                %
  0,+1000;            %
  +1000,-1000;        %
  +1000,0;            %
  +1000,+1000];
Sample Makesardata_DBG Output
DBS Demonstration: procsardata_DBSe

- %#%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
- % User input section %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

- % algorithm control parameters
  - dechirp = false; % use azimuth dechirp step or not
  - oversample_freq = 2; % oversampling in Doppler; bigger makes better
  - % picture but needs more memory and time
  - fix_geometry = false; % perform geometric corrections or not

- % Get data file name
  - file=input('Enter root file name for data file: ','s');

- % End user input section %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
- %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Sample Makesardata_DBS Output, No Geometric Corrections
Sample Makesardata_DBS Output with Geometric Corrections

Resampled and Range-Shifted Image

range relative to CRP (km)
cross-range (km)
Sample procSARdata_DBS Images, with and without Azimuth Dechirp

- Azimuth dechirp not needed if $\Delta CR \geq \sqrt{\frac{\lambda R}{2}}$
- Consider $R = 40$ km, $\lambda = 3$ cm, $\Delta CR = \text{rm}$
  - Violates limit, which is about 17 m in this case
- Generate new data with $\Delta CR = 5$ m, etc.
  - Process with frequency oversample = 4 for good mainlobe definition

No Azimuth Dechirp: Cross-Range Smearing

With Azimuth Dechirp: Cross-Range Resolution Goal Met
Adaptive Beamformer

• Simple demonstration of four different beamformer patterns in an environment with two jammers
  – Non-adaptive (fixed) beamformer
  – fully-adaptive beamformer using weights \( h = S^{-1}_I t^* \)
  – “distortionless” beamformer
  – Post-DFT (beamspace) beamformer

• >> edit beamform
  – Set all parameters by editing input section of beamform.m
  – RF, jammer AOAs, powers
  – Array parameters
  – Use -30 dB Taylor weighting (or not)

• >> beamform
  – Four different patterns in four figure windows
Adaptive Beamformer beamform

lambda = 0.03; % wavelength
d = lambda/2; % element spacing
dl = d/lambda;
N = 16; % # of array elements
aoa_max = asin(1/2/dl); % maximum "real space" AOA (radians)
window_on = false; % true or false

Nangle = 1000; % # of angles for evaluating beam pattern
Nm1 = Nangle-1;

t_aoa = pi/180*(0); % target AOA (radians)
j_aoa1 = pi/180*(18); % jammer #1 AOA (radians)
j_aoa2 = pi/180*(-33); % jammer #2 AOA (radians)

SNR = 0; % signal to noise ratio (dB)
JSR1 = +50; % jammer #1 to noise ratio (dB)
JSR2 = +30; % jammer #2 to noise ratio (dB)

% End user input section
Outputs from beamform, Both Jammers in the Sidelobes, No Weighting
Outputs from beamform, One Jammer in the Mainlobe, No Weighting
Student Projects
Student Projects - 1

• “Student projects” are simulation projects intended to illustrate radar signal behavior and/or teach students to implement simple signal processing algorithms

• Included are four project topics
  – RCS statistics
  – Linear FM waveform properties and matched filtering
  – Pulse Doppler processing and detection
  – CFAR detection

• For each project, the following are provided:
  – Example problem assignment in Microsoft Word format
  – Example problem solution consisting of
    • Sample MATLAB code
    • Microsoft Word document describing the solution
Student Projects - 2

• **Document note for all projects:** Equations in the problem assignment and solution documents were created in MathType™. MathType is an upgrade to Microsoft’s Equation Editor. It can edit equations created in Equation Editor, however, the converse is not true. In addition, MathType may use some fonts or characters not available on machines on which it is not installed. Consequently, it is recommended that the user install MathType if it is desired to work with the student project assignment and solution documents. MathType is available at [www.dessci.com/en/products/mathtype/](http://www.dessci.com/en/products/mathtype/).
Student Projects - 3

• All projects are self-contained and self-explanatory using the problem assignment document, except for the pulse Doppler project.

• The pulse Doppler project requires that data be generated by the instructor, to be analyzed by the students.

• The following charts provide some additional detail on creating data for the pulse Doppler project and then using that data in the sample solution.
MTI + Pulse Doppler Processing

- Place all files in the Pulse Doppler directory in the MATLAB work directory
  - Or anywhere on the MATLAB path
- >> edit makedata
  - Set all parameters by editing input section of makedata.m
  - Includes noise, clutter, moving targets, and $R^4$
    - Radar parameters: RF, PRF, range window
    - Target SNR, ranges, velocities, #of targets
    - CNR
- >> makedata
  - To create data set for processing
  - Output data in file.mat
    - Where file is the “root file name” you specify
    - Parameters logged in file.lis
- >> procdata
  - To perform MTI + pulse Doppler processing, generate displays
    - Input is in same file.mat specified in makedata
    - Program pauses at every graph; hit any key to continue
Create PD Data: makedata

% User Input Section  ###################################################################
% # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
% Get root file name for saving results
file=input('Enter root file name for data and listing files: ','s');

T = 10e-6;     % pulse length, seconds
W = 10e6;      % chirp bandwidth, Hz
fs = 12e6;     % chirp sampling rate, Hz; oversample by a little

Np = 20;        % # of pulses
jkl = 0:(Np-1);  % pulse index array
PRF = 25.0e3;   % PRF in Hz
PRI = (1/PRF);  % PRI in sec
T_0 = PRI*jkl;  % relative start times of pulses, in sec
g = ones(1,Np); % gains of pulses
T_out = [12 38]*1e-6; % start and end times of range window in sec
T_ref = 0;       % system reference time in usec
fc = 10e9;       % RF frequency in Hz; 10 GHz is X-band

vua = 3e8*PRF/(2*fc);

% Compute unambiguous Doppler interval in m/sec
T_out = [12 38]*1e-6; % start and end times of range window in sec
T_ref = 0; % system reference time in usec
fc = 10e9; % RF frequency in Hz; 10 GHz is X-band

% Define number of targets, then range, amplitude, and radial velocity of each
Ntargets = 4;
del_R = (3e8/2)*(1/fs)/1e3; % in km
ranges = [2.3 2.7 3.1 3.5]*1e3; % in km
SNR = [10 20 15 20]; % dB
vels = [-0.3 -0.15 +0.1 +0.25]*vua; % in m/sec

% End User Input Section  ###################################################################
% # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
Final Output from \texttt{makedata}
Process PD Data: procdata

- No parameters to set
- Some sample figures:
Missing Pieces?

- Contact Mark Richards by e-mail, mark.richards@ece.gatech.edu