Data Analysis

Matlab Tutorial

Cameron Lewis, GRA

NATIONAL SCIENCE FOUNDATION :: KANSAS TECHNOLOGY ENTERPRISE CORPORATION :: NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

The University of Kansas | The Ohio State University | Pennsylvania State University
The University of Maine | Elizabeth City State University | Haskell Indian Nations University

Centre for Polar Observation and Modelling | University of Copenhagen
Technical University of Denmark | Antarctic Climate & Ecosystems CRC

CReSIS

Agenda

Zwally et al., 2002, Science

- Motivation
- Background
- Analysis Methods
  - Statistics
  - Interpolation/Extrapolation
  - Gridded Data
  - Selection
  - Visualization
- Examples

Motivation

Importance of polar regions

- Ice Sheets and Glaciers
  - Melting contributes to sea level rise
- Importance of understanding mass balance
  - Greenland loses 100 Gigatons annually (100 km^3)
  - 360 Gigatons=1mm global sea level
  - Snowfall accumulation is between 10cm-2m per year
- Importance of including polar regions in climate models

CReSIS

Definitions

- Data: measurements or observations of a variable
- Analysis: act of transforming data with the aim of extracting useful information and facilitating calculations

Background

- Data Analysis help the knowledge process come full-circle:
    1. Science defines questions and hypotheses
    2. Technology is developed based on this science
    3. Measurements/observations are taken
    4. Data analysis performed on measurements/observations
    5. Conclusions drawn, added to science
    6. New science used to drive new questions and hypotheses

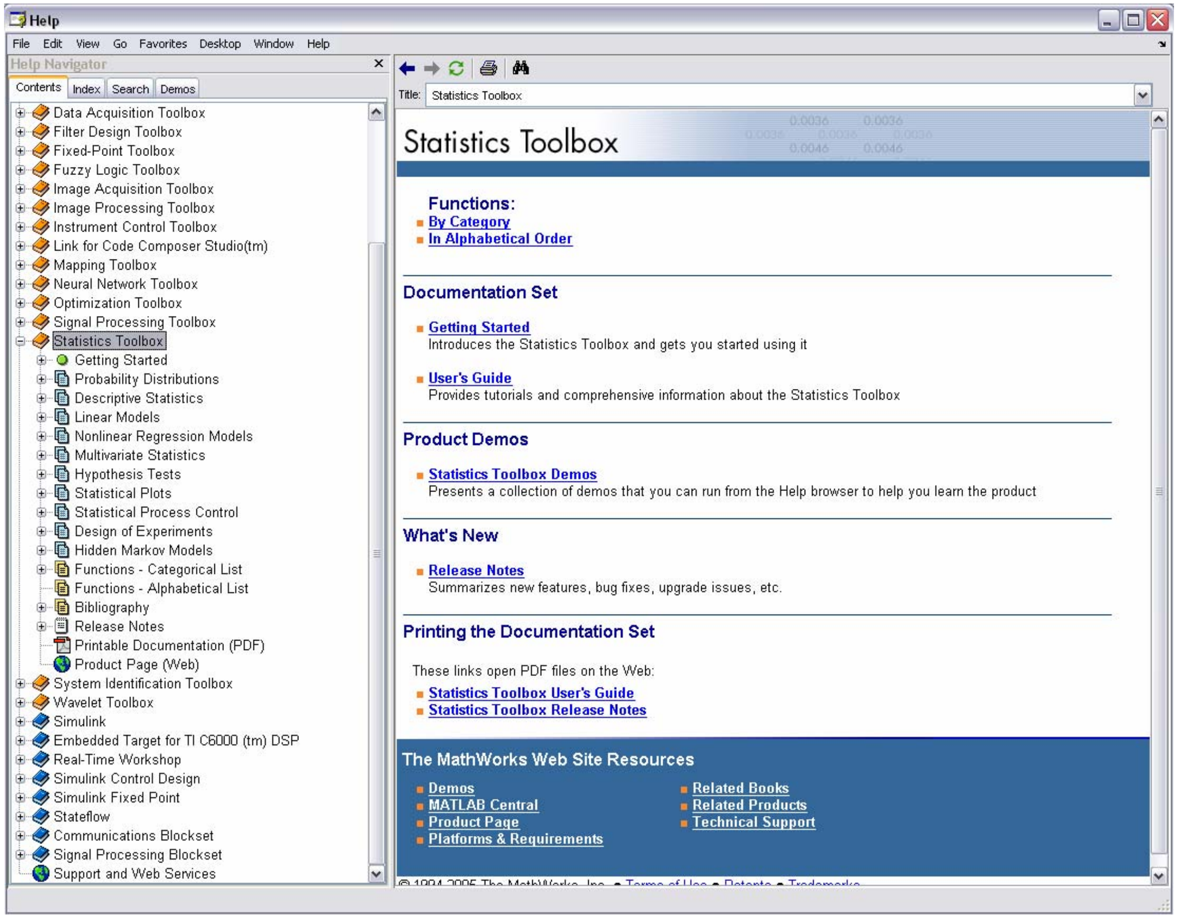
Analysis Methods

- Statistics and curve fitting (regression)
- Interpolation/Extrapolation (modeling)
- Gridded Data (modeling)
- Selecting/Discarding subsets of data based on criteria
- Visualization (explorative analysis)

Statistics

- Myriad of statistical functions built into Matlab
  - Probability distributions
  - Descriptive statistics
  - Linear/Non-Linear regression
  - Plotting

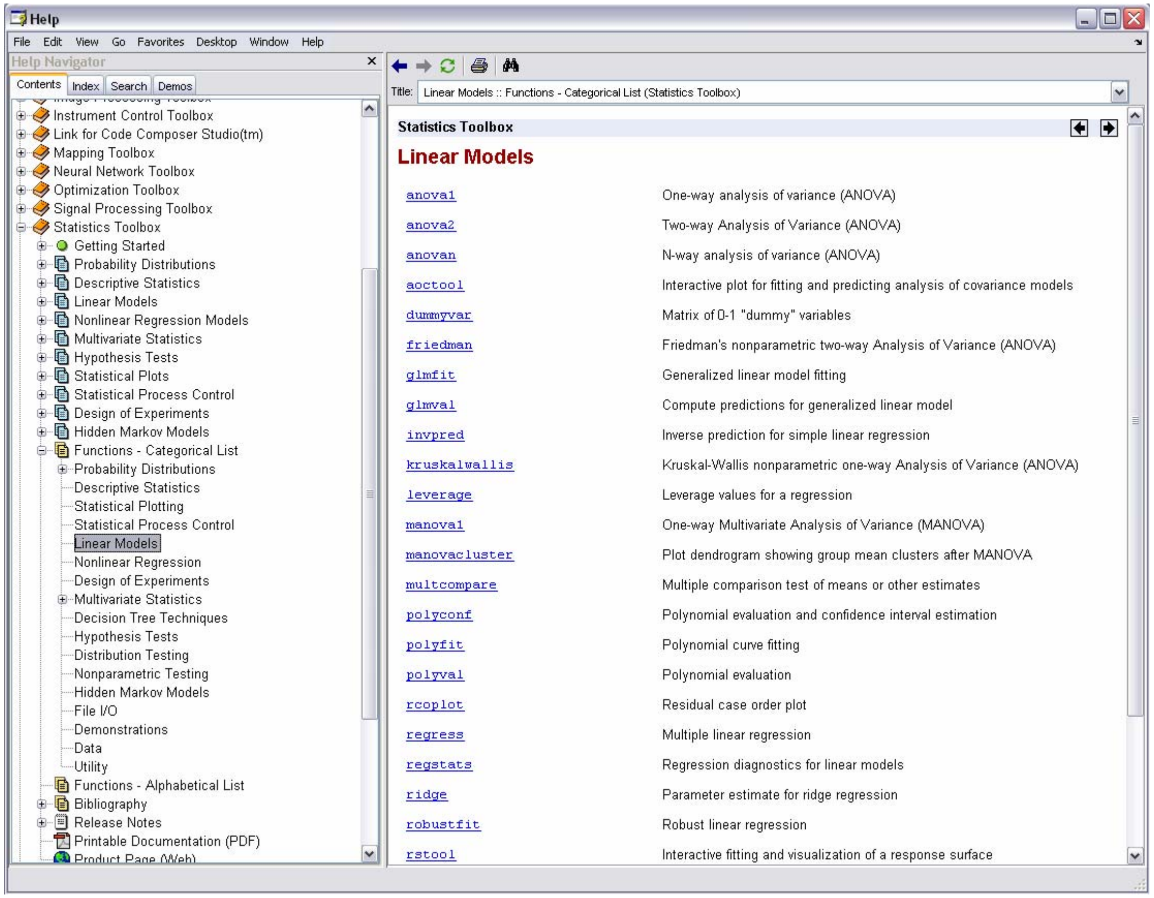
Statistics


Help
File Edit View Go Favorites Desktop Window Help
Help Navigator
Contents Index Search Demos
Data Acquisition Toolbox
Filter Design Toolbox
Fixed-Point Toolbox
Fuzzy Logic Toolbox
Image Acquisition Toolbox
Image Processing Toolbox
Instrument Control Toolbox
Link for Code Composer Studio(tm)
Mapping Toolbox
Neural Network Toolbox
Optimization Toolbox
Signal Processing Toolbox
Statistics Toolbox
Getting Started
Probability Distributions
Descriptive Statistics
Linear Models
Nonlinear Regression Models
Multivariate Statistics
Hypothesis Tests
Statistical Plots
Statistical Process Control
Design of Experiments
Hidden Markov Models
Functions - Categorical List
Functions - Alphabetical List
Bibliography
Release Notes
Printable Documentation (PDF)
Product Page (Web)
System Identification Toolbox
Wavelet Toolbox
Simulink
Embedded Target for TI C6000 (tm) DSP
Real-Time Workshop
Simulink Control Design
Simulink Fixed Point
Stateflow
Communications Blockset
Signal Processing Blockset
Support and Web Services
Title: Statistics Toolbox
Statistics Toolbox
Functions:
By Category
In Alphabetical Order
Documentation Set
Getting Started
Introduces the Statistics Toolbox and gets you started using it
User's Guide
Provides tutorials and comprehensive information about the Statistics Toolbox
Product Demos
Statistics Toolbox Demos
Presents a collection of demos that you can run from the Help browser to help you learn the product
What's New
Release Notes
Summarizes new features, bug fixes, upgrade issues, etc.
Printing the Documentation Set
These links open PDF files on the Web:
Statistics Toolbox User's Guide
Statistics Toolbox Release Notes
The MathWorks Web Site Resources
Demos
MATLAB Central
Product Page
Platforms & Requirements
Related Books
Related Products
Technical Support

Statistics

- Commonly used functions:
  - mean, geomean, range
  - unifit, normfit, lognfit, poissfit, expfit
  - dfittool, disttool, polytool
  - boxplot, gscatter, normplot, pareto

Statistics


Help
File Edit View Go Favorites Desktop Window Help
Help Navigator
Contents
Index
Search
Demos
Instrument Control Toolbox
Link for Code Composer Studio(tm)
Mapping Toolbox
Neural Network Toolbox
Optimization Toolbox
Signal Processing Toolbox
Statistics Toolbox
Getting Started
Probability Distributions
Descriptive Statistics
Linear Models
Nonlinear Regression Models
Multivariate Statistics
Hypothesis Tests
Statistical Plots
Statistical Process Control
Design of Experiments
Hidden Markov Models
Functions - Categorical List
Probability Distributions
Descriptive Statistics
Statistical Plotting
Statistical Process Control
Linear Models
Nonlinear Regression
Design of Experiments
Multivariate Statistics
Decision Tree Techniques
Hypothesis Tests
Distribution Testing
Nonparametric Testing
Hidden Markov Models
File I/O
Demonstrations
Data
Utility
Functions - Alphabetical List
Bibliography
Release Notes
Printable Documentation (PDF)
Title: Linear Models :: Functions - Categorical List (Statistics Toolbox)
Statistics Toolbox
Linear Models
anova1 One-way analysis of variance (ANOVA)
anova2 Two-way Analysis of Variance (ANOVA)
anovan N-way analysis of variance (ANOVA)
aoctool Interactive plot for fitting and predicting analysis of covariance models
dummyvar Matrix of 0-1 "dummy" variables
friedman Friedman's nonparametric two-way Analysis of Variance (ANOVA)
glmfit Generalized linear model fitting
glmval Compute predictions for generalized linear model
invpred Inverse prediction for simple linear regression
kruskalwallis Kruskal-Wallis nonparametric one-way Analysis of Variance (ANOVA)
leverage Leverage values for a regression
manova1 One-way Multivariate Analysis of Variance (MANOVA)
manovacluster Plot dendrogram showing group mean clusters after MANOVA
multcompare Multiple comparison test of means or other estimates
polyconf Polynomial evaluation and confidence interval estimation
polyfit Polynomial curve fitting
polyval Polynomial evaluation
rcoplot Residual case order plot
regress Multiple linear regression
regstats Regression diagnostics for linear models
ridge Parameter estimate for ridge regression
robustfit Robust linear regression
rstool Interactive fitting and visualization of a response surface

Curve Fitting

- Form of regression
- Linear regression is the simplest
- Built-in Matlab functions
  - polyfit
  - polyval

polyfit

p = polyfit(x,y,n)

[p,S] = polyfit(x,y,n)

[p,S,mu] = polyfit(x,y,n)

x and **y** define points of measured data

n defines order of desired regression polynomial

p array defining polynomial coefficients

S structure defining error function parameters

mu two-element vector specifying first two moments

polyfit

```
% FOR TESTING - Reduce to first 10000 rows
iout = iout(1:10000,:);

fprintf('Fitting data to straight line\n');
% East-North data set
[north, east, height] = eastNorth(old_lat(1), old_lon(1), old_elev(1), old_lat, old_lon, old_elev);
% find slope of polyfit line
p = polyfit(east,north,1);
path_slope = p(1,1);

% fit data to new coordinate
ortho_norm_vector = ([1 path_slope]/(1+(path_slope.^2)));
x_distance = ([east(1) north(1)] * ortho_norm_vector.');
for ENindex = 2:length(east)
    x_dist_TEMP = ([east(ENindex) north(ENindex)] * ortho_norm_vector.');
    x_distance = [x_distance, x_dist_TEMP];
end
```

- Find linear regression (n=1)
- Fit data points to that regression
- Decompose fitted points into east and north components
- Example: used in FK migration process so that DFT can be performed

polyval

Y = polyval(p,X)

[Y, DELTA] = polyval(p,X,S)

p array of polynomial coefficients defined by polyfit

X values for which Y values are defined

Y desired values of **p** function as desired by **X**

DELTA error estimates → **Y ± DELTA**

Interpolation

- Interpolation and Extrapolation are both handled via the **interp** functions
  - interp1 (one dimension)
  - interp2 (two dimension)
  - interp3 (three dimension)
  - interpn (n dimension)
  - interpft (one dimension interpolation using the FFT method)

interp1

yi = interp1(x,Y,xi)

yi = interp1(x,Y,xi,method)

yi = interp1(x,Y,xi,method,'extrap')

yi newly interpolated y values based on xi positions

x original data x vector

Y original data y vector

xi new x position vector, used to define interpolation points

method defines the interpolation method (i.e. 'linear', 'spline', 'cubic')

interp1

```
if (ispc)
  acid = load('P:\prism\radar\radar_simulator\profiles\gisp2_dep_20030914.txt');
else
  acid = load('/projects/prism/radar/radar_simulator/profiles/gisp2_dep_20030914.txt');
end
eV = 0.22;
acid_interp = 1e-6 * interp1(acid(:,1),acid(:,3),depthInt,'linear','extrap').';
condInterp = 1e-6 * interp1(acid(:,1),acid(:,2),depthInt,'linear','extrap').';
if (plotFlag)
  plot(depthInt,1e6*acid_interp,'k-');
  old = axis; axis([0 3047.9 old(3:4)]);
  xlabel('Depth (meters)');
  ylabel('Acidity (micromolarity)');
  fprintf('Mean acidity = %f micromolarity\n', mean(acid_interp)/1e-6);
  pause;
  plot(depthInt,1e6*condInterp,'k-');
  old = axis; axis([0 3047.9 old(3:4)]);
  xlabel('Depth (meters)');
  ylabel('Conductivity (uS/m)');
  fprintf('Mean conductivity = %f uS/m\n', mean(condInterp)/1e-6);
  pause;
end
```

- Interpolate ice acid content data to the predefined depth array
- Linear interpolation, with extrapolation of the acid content data for the extra points at the bottom of the ice sheet

Gridded Data

- Measured data is often random in both space and time
- In order for this data to be useful for selection/visualization/modeling, it must be fit to a grid that is evenly divided in both space and time
- Requires methods of interpolation (and sometimes extrapolation)
- Matlab provides functions for this:
  - meshgrid
  - griddata

meshgrid

[X,Y] = meshgrid(x,y)

[X,Y,Z] = meshgrid(x,y,z)

- Used to define a 2D or 3D grid **[X,Y]** based on **x** and **y**
- **x** and **y** must be monotonically increasing vectors
- meshgrid is required in order to define interpolation (and extrapolation) points for griddata

NSF NASA KTEC CReSIS

griddata

Zi = griddata(x,y,z,Xi,Yi,*method*,*options*)

Zi is the interpolated (extrapolated) **z** values by remapping them from the original **x,y** system to the defined **Xi,Yi** system created with meshgrid

method defines the interpolation method

- 'linear'
- 'cubic'
- 'nearest'

options typically not used

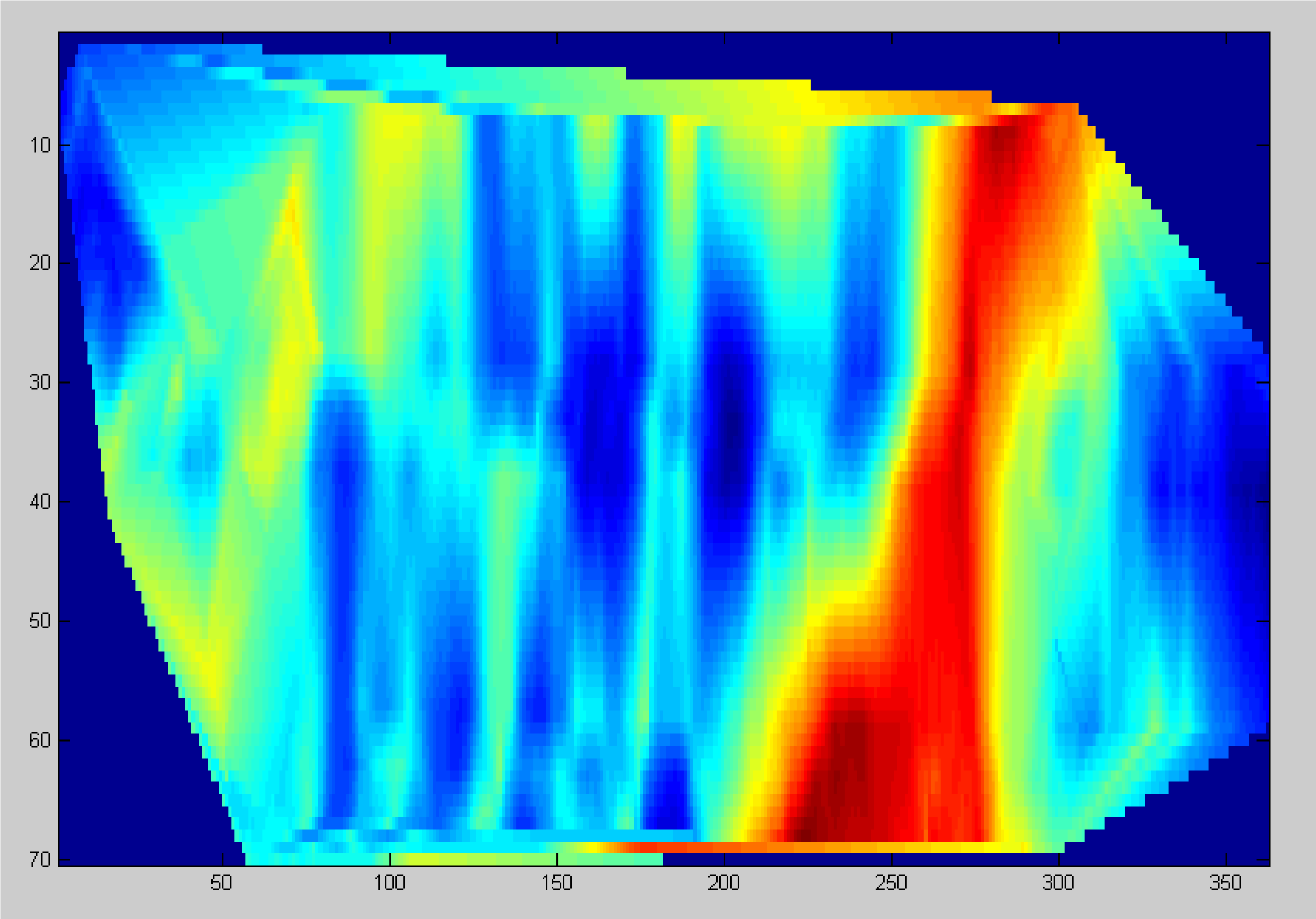
Gridded Data

```
% create_grid_data
% Cameron Lewis
% This program will create gridded data out of complete_bedrock
% Requires: complete_bedrock, complete_lat, complete_lon, and complete_elev

path(path,genpath('/ps3/matlab/support/geometry'));

file = load('/ps3/insar/results/complete_bedrock');

[north, east, height] = eastNorth(1.26672971, -0.671246297, 3252.3602, file.complete_lat, ...
        file.complete_lon, file.complete_elev);
north_min = min(north);
north_max = max(north);
east_min = min(east);
east_max = max(east);
[X,Y] = meshgrid(east_min:100:east_max, north_min:100:north_max);
Z = griddata(east, north, file.complete_bedrock, X, Y);
~
~
                                                                        1,1           All
```

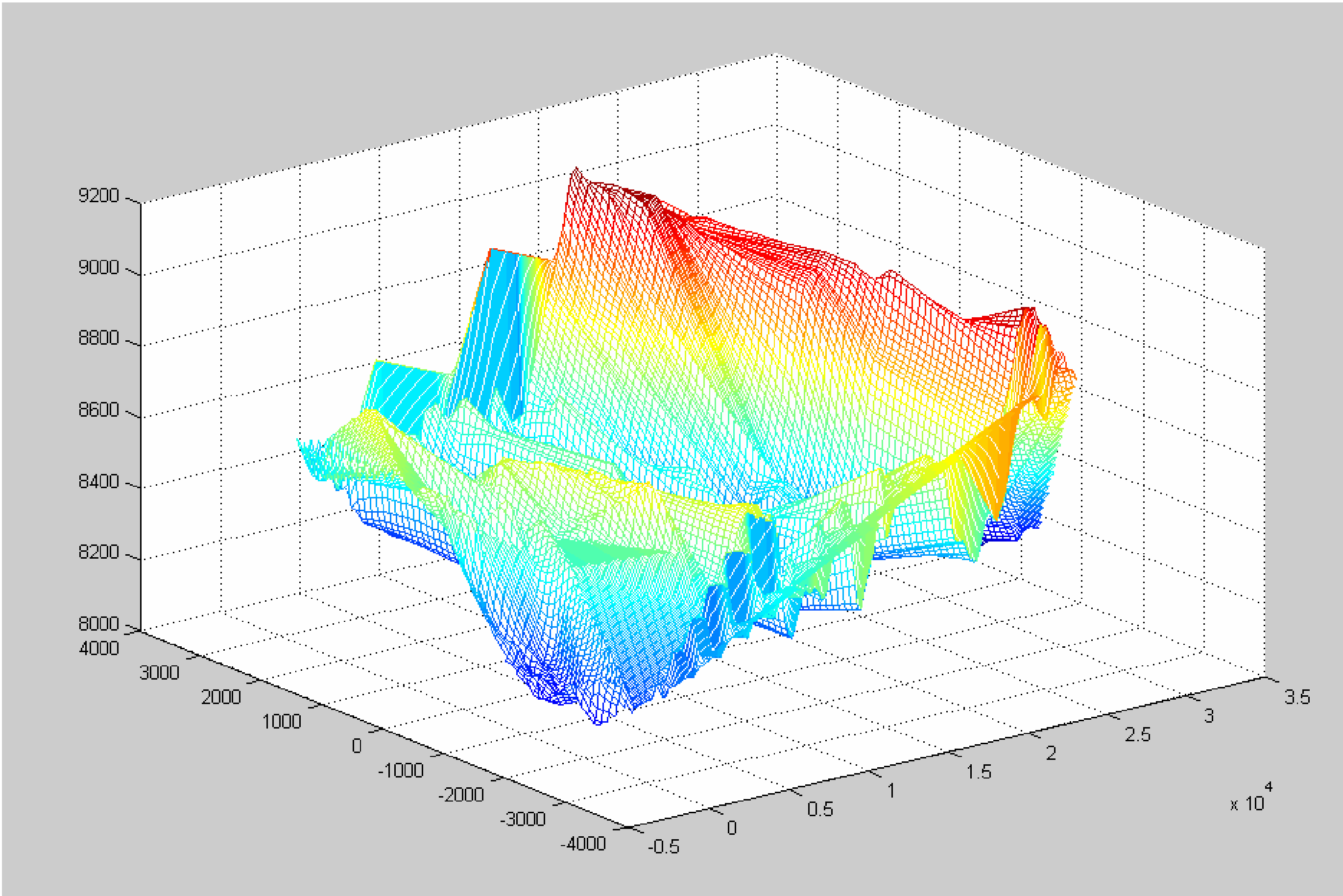
- Create gridded data of bedrock depth by remapping from track lines to grid
- Use **imagesc**, **mesh**, **surf**, etc to plot the results

CReSIS

imagesc



mesh

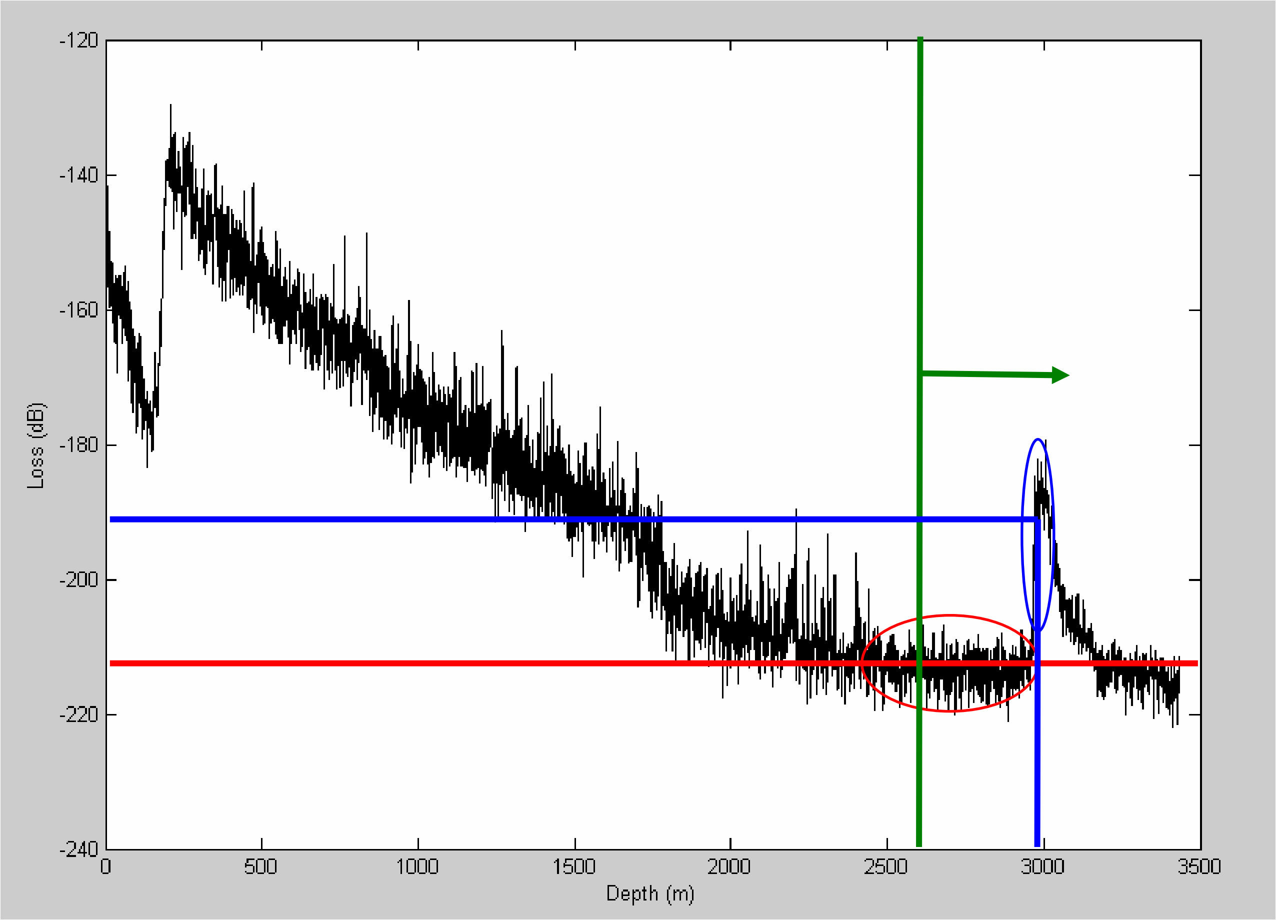


9200
9000
8800
8600
8400
8200
8000
4000
3000
2000
1000
0
-1000
-2000
-3000
-4000
-0.5
0
0.5
1
1.5
2
2.5
3
3.5
x 10^4

Selection

- Selection or deletion of a subset of data based on a criteria
- Example: find the bedrock depth by finding the first point, below 2600m, that is 20dB above the noise floor

AScope



-120
-140
-160
-180
-200
-220
-240
Loss (dB)
0
500
1000
1500
2000
2500
3000
3500
Depth (m)

Bedrock Locator

```
% Attempt 2: Threshold
% Look for first point that is 20dB above noise floor below point 7500

noise_floor = -124;
bedrock_sig = (noise_floor + 20);
start = 7500;
max_rows = size(rds_data,1);
memory_spread = 40;
bedrock_loc = zeros(1,size(rds_data,2));

for column = 1:size(rds_data,2);
    for row = start:max_rows
        value = rds_data(row, column);
        if((column == 1) && (memory_in == 0))
            if(value > bedrock_sig)
                bedrock_loc(column) = row;
                break;
            end
        elseif((column == 1) && (memory_in ~= 0))
            bottom = (memory_in - memory_spread);
            top = (memory_in + memory_spread);
            if((value > bedrock_sig) && (row > bottom) && (row < top))
                bedrock_loc(column) = row;
                break;
            end
        elseif((column > 1) && (bedrock_loc(column-1) ~= 0))
            bottom = (bedrock_loc(column-1) - memory_spread);
            top = (bedrock_loc(column-1) + memory_spread);
            if((value > bedrock_sig) && (row > bottom) && (row < top))
                bedrock_loc(column) = row;
                break;
            end
        elseif((column > 1) && (bedrock_loc(column-1) == 0))
            if(value > bedrock_sig)
                bedrock_loc(column) = row;
                break;
            end
        end
    end
    if(bedrock_loc(column) == 0)
        warning = sprintf('Warning: Lost bedrock at column %d', column);
        disp(warning);
    end
end
memory_out = bedrock_loc(column);
```

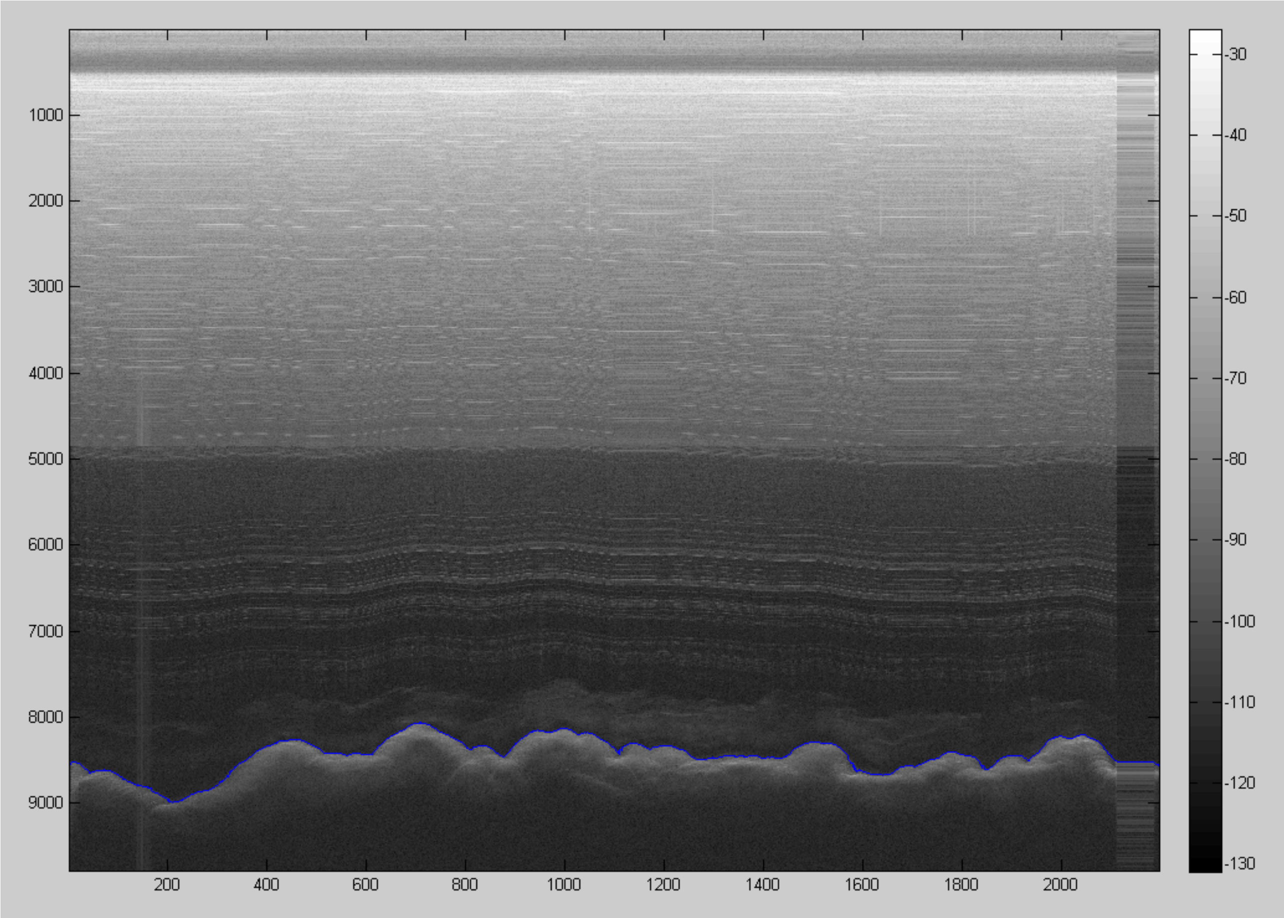
- Program acquires bedrock location in first column
- This is used as a starting point in the next column
- If bedrock is lost, program returns to acquisition step

CReSIS

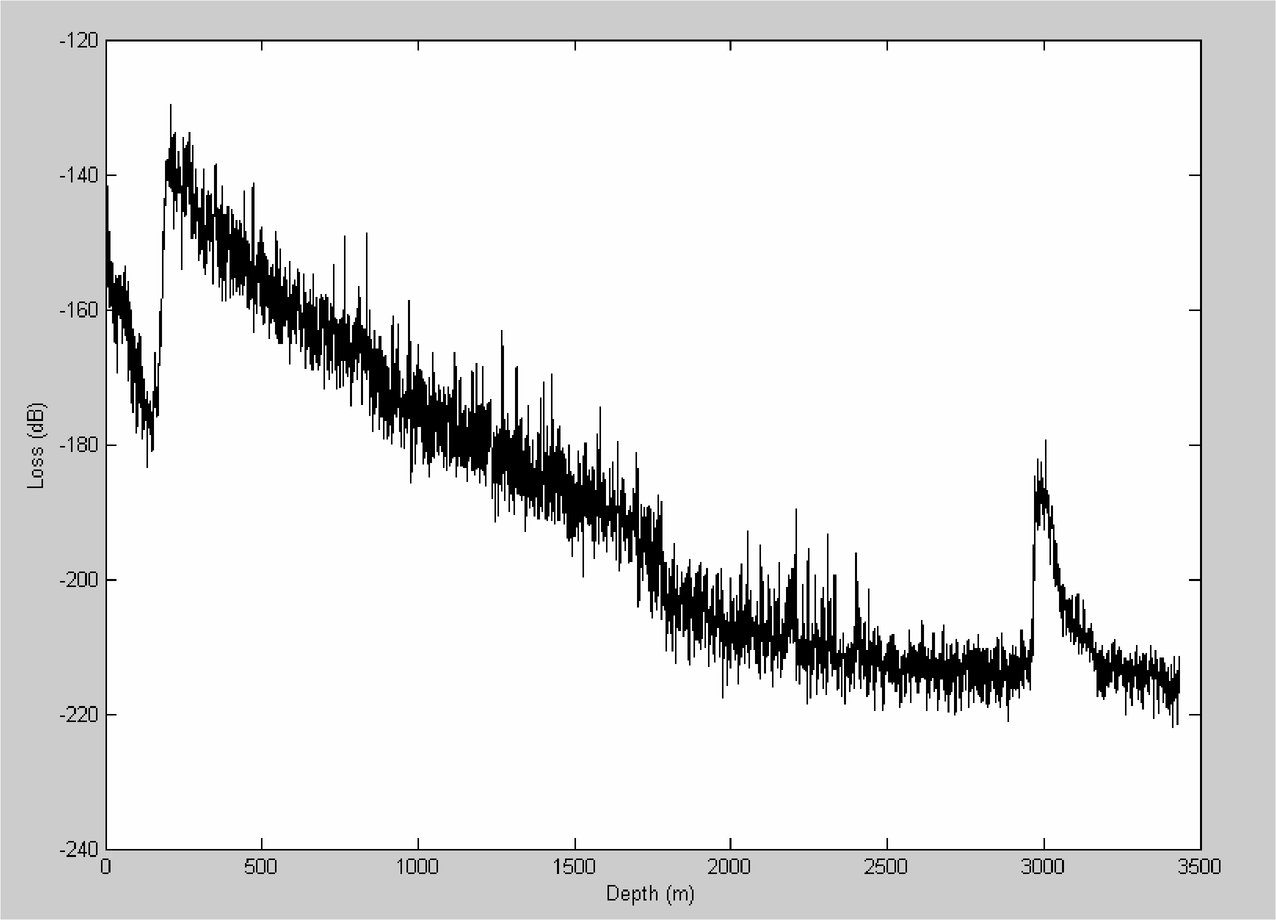
Visualization

- Analysis of data through sight
- An important part of the analysis process
- Particularly used during explorative data analysis, where the analysis is driven by the data itself, as opposed to a hypothesis

Cross Sections

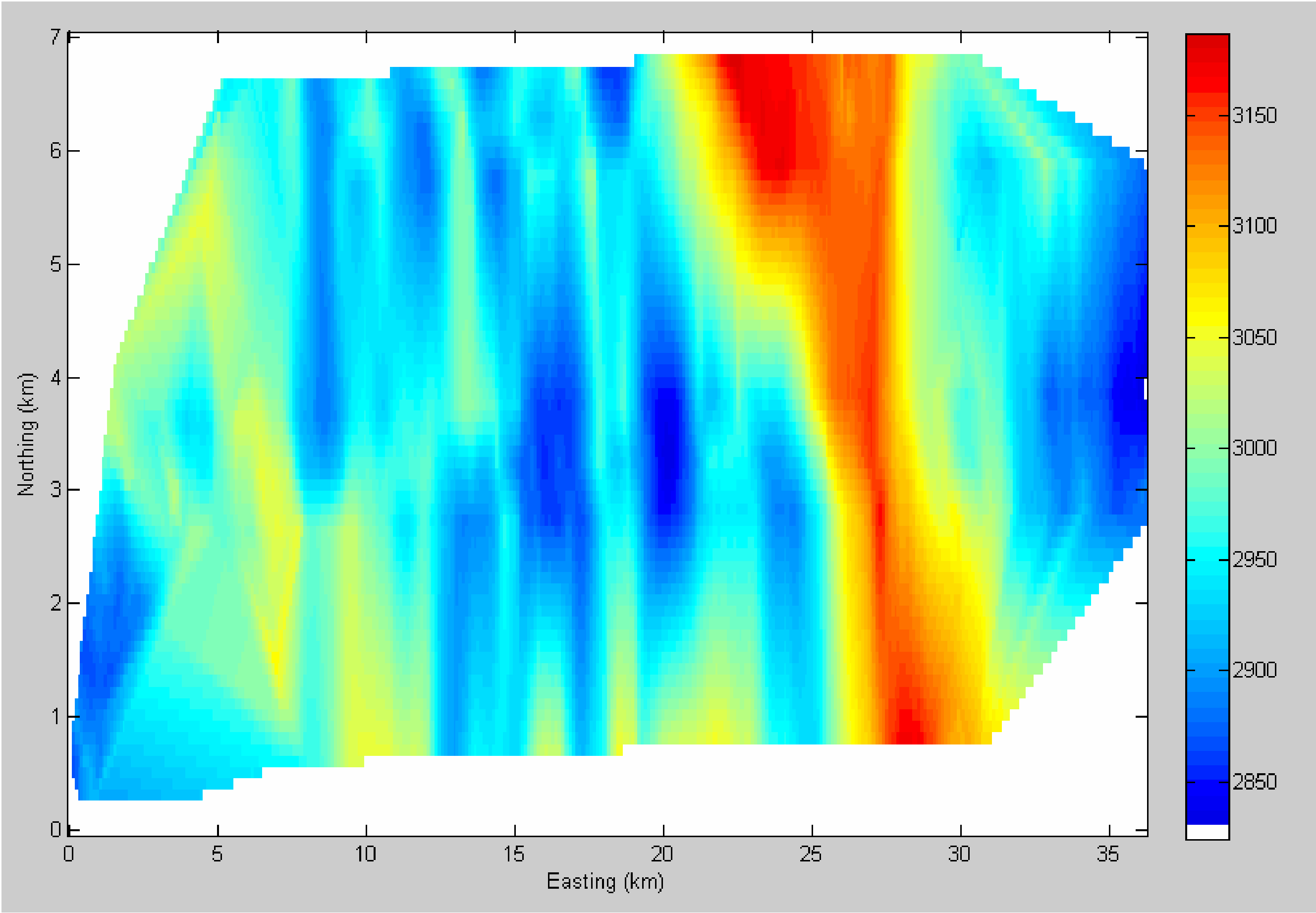


AScope



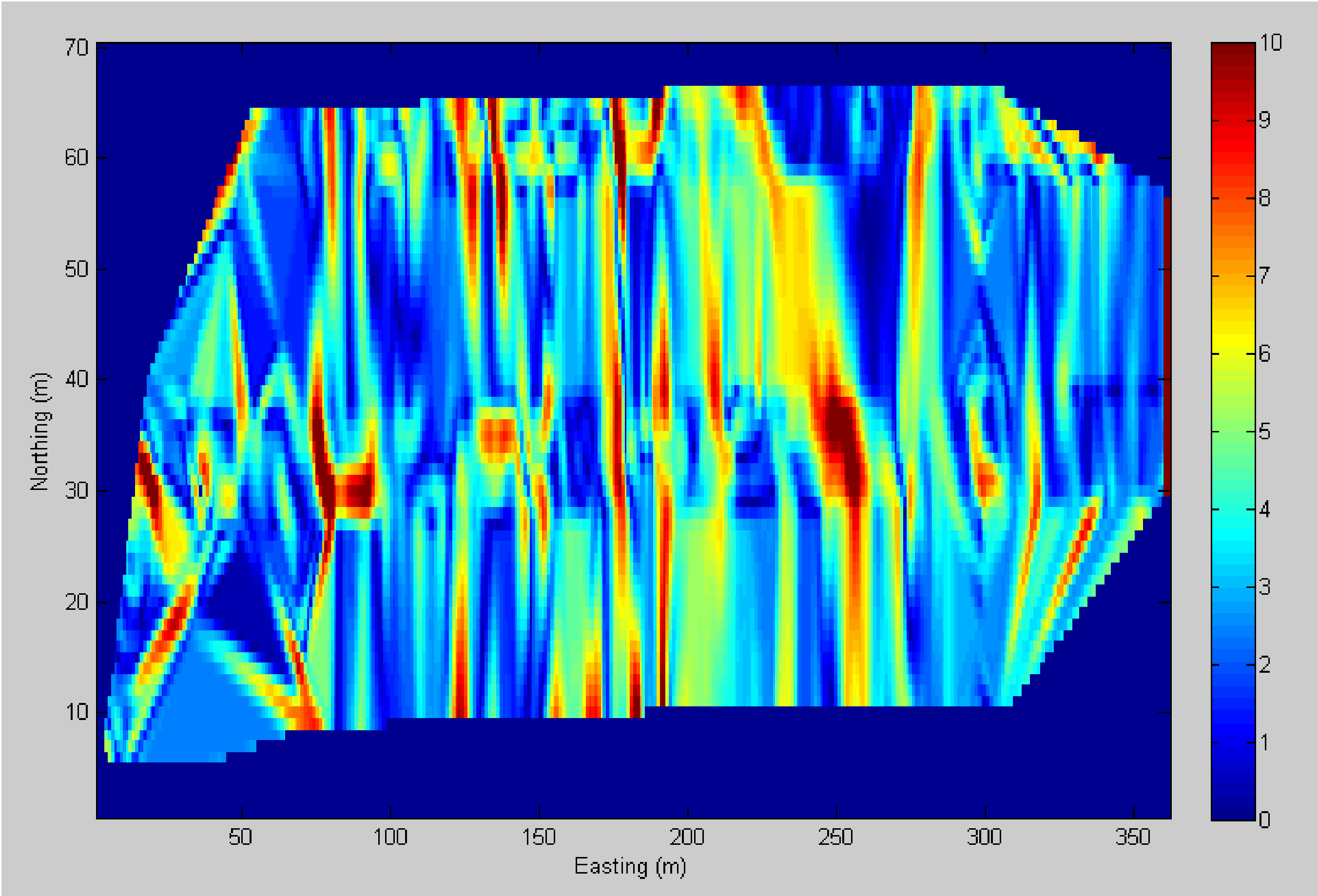
CReSIS

Thickness Chart



CReSIS

Slope Chart



Northing (m)
Easting (m)
70
60
50
40
30
20
10
50
100
150
200
250
300
350
10
9
8
7
6
5
4
3
2
1
0

Questions?

- Office: 315 Nichols
- Email: clewis@cresis.ku.edu

National Science Foundation
WHERE DISCOVERIES BEGIN

KTEC
KANSAS TECHNOLOGY
ENTERPRISE CORPORATION

KU THE UNIVERSITY OF KANSAS

DTU

cpom

MAINE

THE OHIO STATE UNIVERSITY

PENNSTATE 1855

ELIZABETH CITY ECSU

ace CRC

NSF NASA KTEC

CReSIS