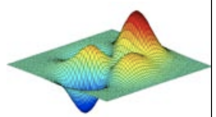
Applied Geoscience Data Analysis using Matlab



07 Nov 2013

(Lecture 20)

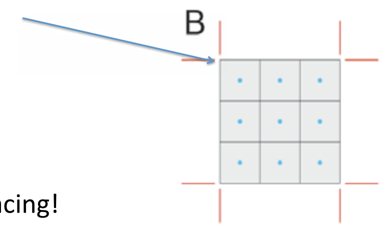
Referencing Matrix, Profiling & Terrain Analysis

1

Referencing Vector (from last lecture)

R= [cells/angular unit, north-latitude, west-longitude]

Where north-latitude and west-longitude refer to the edges of the upper-left corner.



Cell- or pixel-registration

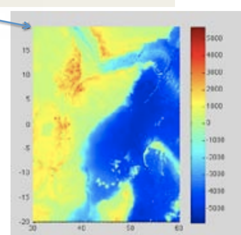
Requires equal x-y spacing!

Example from last lecture

For ETOPO2 (2 minute or 2/60 = 1/30 degree spacing)
(i.e., a dataset with ½ the resolution of ETOPO1)
R = [30, 20+1/60, 30-1/60];

Limits of the node registered file, plus ½ grid space to the north and west of north-west most node.

R = 30.0   20.0167   29.9833



Referencing Matrix (R)
- new/expanded in version 3.0 of the mapping toolbox
- Allows for a different spacing in the x and y dimensions of the grid

R is a 3-by-2 matrix that transforms pixel subscripts (row, column) to map coordinates (x,y) according to:

[x y] = [row col 1] * R

Or transforms pixel subscripts to/from geographic coordinates according to

[lon lat] = [row col 1] * R.

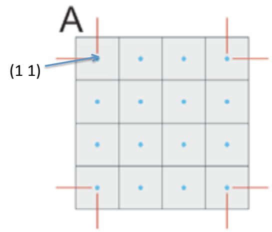
R = makerefmat(X_{11}, Y_{11}, DX, DY)

- DX and DY are scalars giving the spacing between pixels.

X_{11} and Y_{11} are scalars that specify the location of the center of the first (1,1) entry in the NODE registered sense.

DX is the difference in X (or longitude) between pixels in successive columns

DY is the difference in Y (or latitude) between pixels in successive rows.



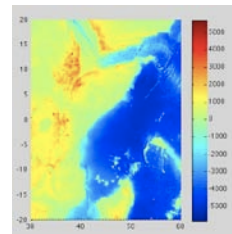
Grid- or node-registration

Referencing Matrix (example for ETOPO2 dataset)

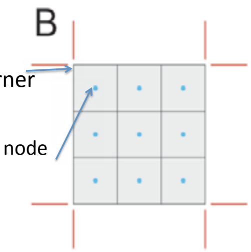
R = makerefmat(X_{11}, Y_{11}, DX, DY)
R = makerefmat(30,20,1/60,-1/60)   % note the negative DY

R =[0 -0.01667
0.01667 0
29.9833 20.01666]

North-west corner



Upper-left node
20 N, 30 E



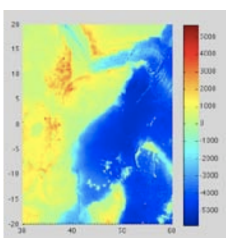
Cell- or pixel-registration

R =[0 -0.01667
0.01667 0
29.9833 20.01666]

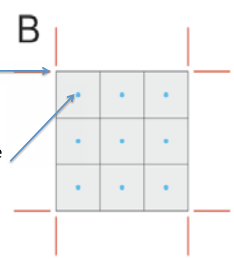
[lon lat] = [row col 1] * R.

[1 1 1]*R   Did this work?
= 30 20

20.016 N 29.9833 E



Upper-left node
20 N, 30 E



Cell- or pixel-registration

Extracting values z from the geo-referenced gridded data

VAL = ltln2val(Z, R, LAT, LON) interpolates a regular data grid Z with referencing vector R at the points specified by vectors of latitude and longitude, LAT and LON. R is either a 1-by-3 vector containing elements:

[cells/degree northern_latitude_limit western_longitude_limit]

or a 3-by-2 referencing matrix that transforms raster row and column
indices to/from geographic coordinates according to:

[lon lat] = [row col 1] * R.

If R is a referencing matrix.

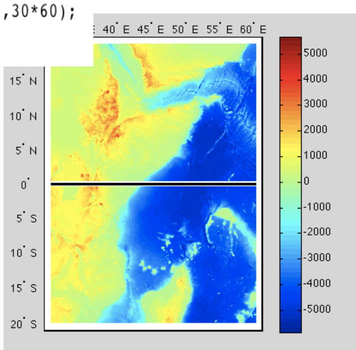
Profiling: 1st define points to sample

```
TRACK2  Geographic tracks from starting and ending points

   [lat,lon] = TRACK2(lat1,lon1,lat2,lon2) computes great circle
   tracks on a sphere starting at the point lat1, lon1 and ending at
   lat2, lon2.  The inputs can be scalar or column vectors.
```

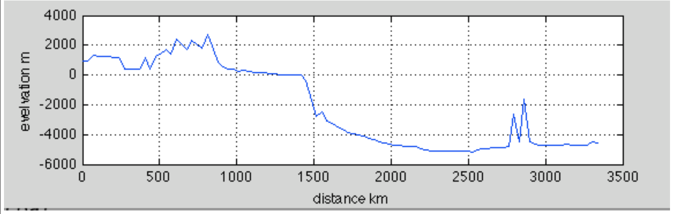
```
[tr_lat,tr_lon]=track2(0,30,0,60,30*60);
h=linem(tr_lat,tr_lon,'k');
set(h,'LineWidth',3)
```

Also See
track1
trackg



Then extract the values at these points

```
trz=ltln2val(ETOPO1,R,tr_lat,tr_lon);
trd=distance(tr_lat(1),tr_lon(1),tr_lat,tr_lon);
figure; plot(deg2km(trd),trz); grid on
xlabel('distance km'); ylabel('evelvation m');
```



An alternative to ltln2val.

[zi,rng] = mapprofile(Z,R,lat,lon) accepts as input a regular data grid and waypoint vectors. No displayed grid is required. Sets of waypoints may be separated by NaNs into line sequences. The output ranges are measured from the first waypoint within a sequence. R is either a 1-by-3 vector containing elements:

[cells/degree northern_latitude_limit western_longitude_limit]

or a 3-by-2 referencing matrix that transforms raster row and column indices to/from geographic coordinates according to:

[lon lat] = [row col 1] * R.

If R is a referencing matrix,

Many of the mapping toolbox function default to a spherical earth model; however, you can also specify the reference ellipsoid.

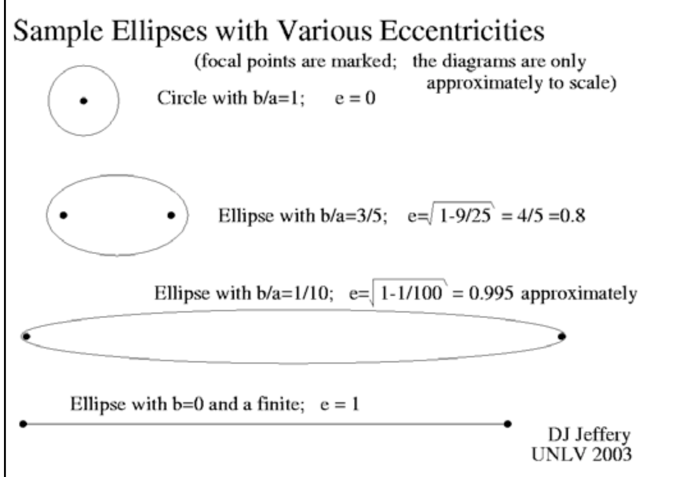
For example.

[zi,rng] = mapprofile(Z,R,lat,lon,ellipsoid)

Or

[arclen,az] = distance(lat1,lon1,lat2,lon2,ellipsoid)

- Where ellipsoid is a vector of the form: [semimajor_axis, eccentricity].
- The output, arclen, is expressed in the same length units as the semimajor axis of the ellipsoid.



We can get specific ellipsoid models using almanac

almanac R2012b

Parameters for Earth, planets, Sun, and Moon

Syntax

```
almanac
almanac(body)
data = almanac(body,parameter)
data = almanac(body,parameter,units)
data = almanac(parameter,units,referencebody)
```

Description

almanac is not recommended. Use `earthRadius`, `referenceEllipsoid`, `referenceSphere`, or `wgs84Ellipsoid` instead.

Almanac (perhaps) being phased out in 2012 version, but still functioning

'clarke80'	1880 Clarke ellipsoid
'international'	1924 International ellipsoid
'krasovsky'	1940 Krasovsky ellipsoid
'wgs60'	1960 World Geodetic System ellipsoid
'iau65'	1965 International Astronomical Union ellipsoid
'wgs66'	1966 World Geodetic System ellipsoid
'iau68'	1968 International Astronomical Union ellipsoid
'wgs72'	1972 World Geodetic System ellipsoid
'grs80'	1980 Geodetic Reference System ellipsoid
'wgs84'	1984 World Geodetic System ellipsoid

>> distance(35,-80,36,-81) % spherical earth

ans = 1.2895degrees

>> distance(35,-80,36,-81,almanac('earth',’wgs84','degrees'))

ans = 1.2889 degrees

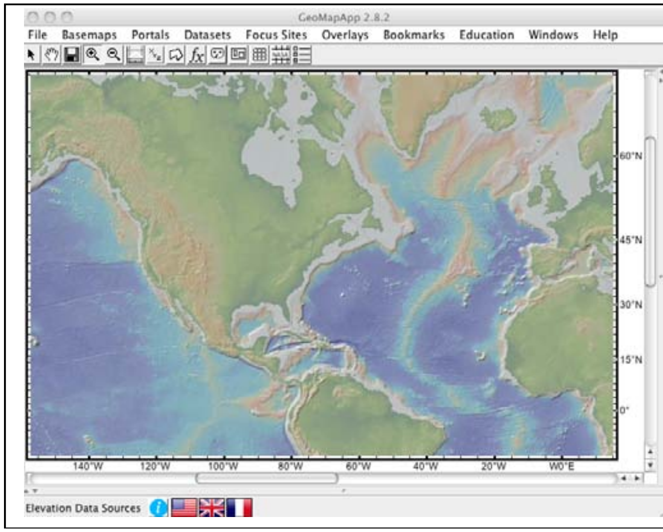
>> distance(35,-80,36,-81,almanac('earth',’wgs84','meters'))

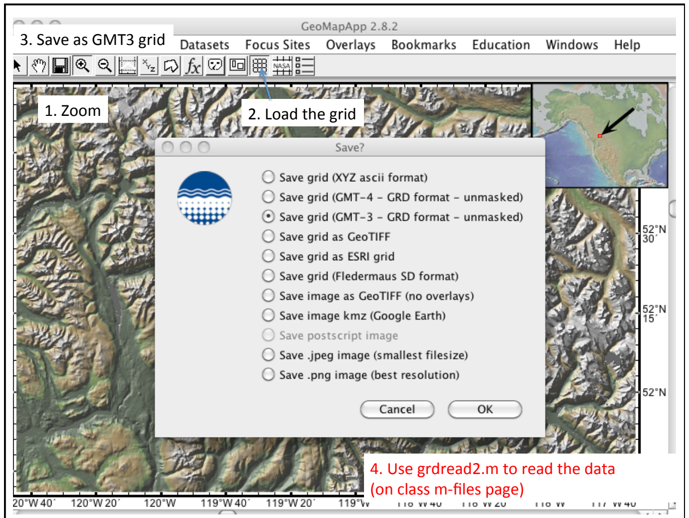
ans = 1.4332e+05 meters

>> distance(35,-80,36,-81,almanac('earth',’wgs84','km'))

ans = 143.3216 km

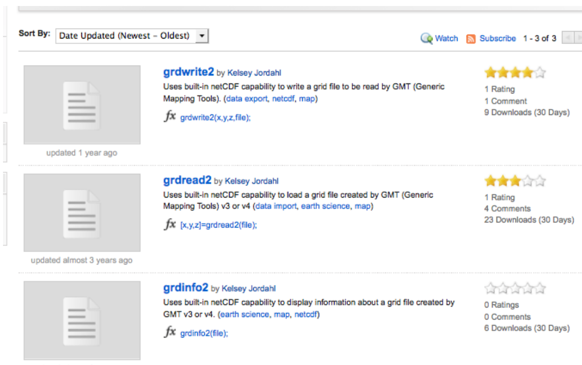
A couple other ways to get topography & bathymetry: GeoMapApp





http://www.marine-geo.org/portals/gmrt/

Download from the mathworks central page



grdread2 read in as NODE registered GMT data

```
>> [X,Y,Z]=grdread2('canrockies_gmt3.grd');
>> whos
  Name         Size                Bytes  Class

  X            1x723                5784  double
  Y            1x475                3800  double
  Z          475x723             1373700  single
```

dx=mode(diff(X)); % = 0.00439
dy=mode(diff(Y)); %= 0.0026

R=makerefmat(X(1),Y(1),dx,dy)
R =[0 0.00269
0.00439 0
-120.744 51.680

Actual Header File Info:
grdinfo2('canrockies_gmt3.grd')
Gridline node registration used

x_min: -120.739746094
x_max: -117.566894531
x_inc: 0.00439453125
name: Longitude nx: 723

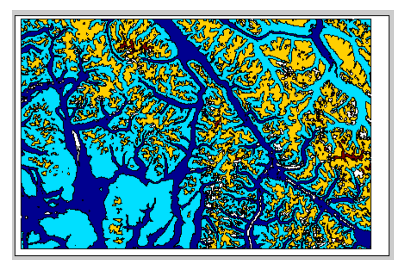
y_min: 51.6834549965
y_max: 52.9565806756
y_inc: 0.00268591915434
name: Latitude ny: 475

We can check that our R matrix return the correct result

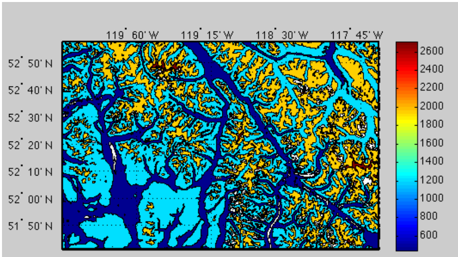
[1 1 1]*R
= -120.739746094
51.6834549965

[475 7231]*R
= -117.566894531
52.95658067561

```
[X,Y,Z]=grdread2('canrockies_gmt3.grd');
dx=mode(diff(X));
dy=mode(diff(Y));
Z=double(Z); % make into a floating point value
R=makerefmat(X(1),Y(1),dx,dy);
figure; axesm('MapProjection','mercator');
contourfm(Z,R,3,'k')
```



```
setm(gca, 'MapLatLimit',[min(Y) max(Y)],'MapLonLimit',[min(X) max(X)])
setm(gca,'ParallelLabel','on','MeridianLabel','on','Grid','on','LabelUnits','dm')
setm(gca,'MLabelLocation',45/60,'PLabelLocation',10/60)
setm(gca,'MlineLocation',45/60,'PLineLocation',10/60)
tightmap; colorbar('vert');
```



Terrain Analysis:
Digital estimation of the slope, aspect, and curvatures of terrain data.

Leading to: terrain classification, flow path analysis, catchment delineation, solar radiation, channel lines, line of sight calculation.

Mapping toolbox function: gradientm

[aspect, slope, gradN, gradE] = gradientm(Z, R)

Computes the slope, aspect and north and east components of the gradient for a regular data grid Z with 1x 3 referencing vector or 3 x 2 reference matrix (if x and y grid spacing are different).

If the grid contains elevations in meters, the resulting aspect and slope are in units of **degrees clockwise from north and up from the horizontal**.

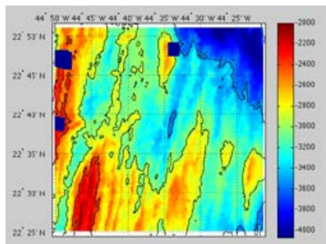
The north and east gradient components are the **change in the map variable per meter of distance in the north and east directions**.

[...] = gradientm(lat, lon, Z)

lat and lon are the latitudes and longitudes of the geo-located points, and are in degrees.

```
[X,Y,Z]=grdread2('grid3_rs.grd');
Z=double(Z); dx=mode(diff(X));  dy=mode(diff(Y));
R=makerefmat(X(1),Y(1),dx,dy) % spacing not equal
figure; axesm('MapProjection','mercator');
meshm(Z,R); view(0,90); caxis([min(Z(:)),max(Z(:))]);
contourm(Z,R,10,'k')
setm(gca, 'MapLatLimit',[min(Y) max(Y)],'MapLonLimit',[min(X) max(X)])
setm(gca,'ParallelLabel','on','MeridianLabel','on','Grid','on','LabelUnits','dm')
setm(gca,'MLabelLocation',5/60,'PLabelLocation',5/60)
setm(gca,'MlineLocation',5/60,'PLineLocation',5/60)
tightmap; colorbar('vert');
```

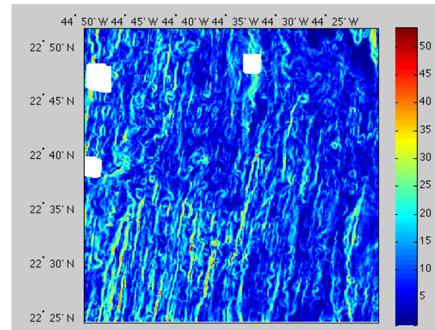
Terrain Analysis Example
Flanks of Mid Atlantic Ridge



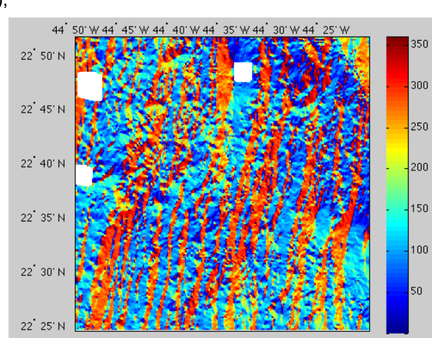
```
[aspect,slope,gradN,gradE] = gradientm(Z,R);  % does all the calculations

[LAT, LON] = meshgrat(Z, R);  % need for surfm
```

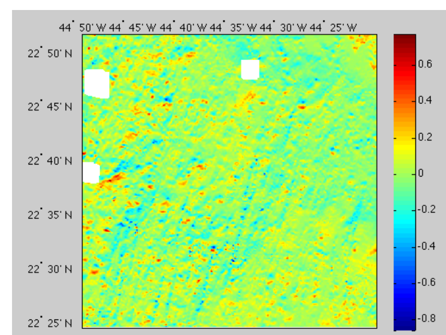
```
figure; axesm('MapProjection','mercator');
surfm(LAT,LON,slope);
setm(gca, 'MapLatLimit',[min(Y) max(Y)],'MapLonLimit',[min(X) max(X)])
setm (gca,'ParallelLabel','on','MeridianLabel','on','LabelUnits','dm')
setm(gca,'MLabelLocation',5/60,'PLabelLocation',5/60)
setm(gca,'MlineLocation',5/60,'PLineLocation',5/60)
tightmap; colorbar('vert');
```




```
figure; axesm('MapProjection','mercator');
surfm(LAT,LON,aspect);
setm(gca, 'MapLatLimit',[min(Y) max(Y)],'MapLonLimit',[min(X) max(X)])
setm (gca,'ParallelLabel','on','MeridianLabel','on','LabelUnits','dm')
setm(gca,'MLabelLocation',5/60,'PLabelLocation',5/60)
setm(gca,'MlineLocation',5/60,'PLineLocation',5/60)
tightmap; colorbar('vert');
```




```
figure; axesm('MapProjection','mercator');
surfm(LAT,LON,gradN);
setm(gca, 'MapLatLimit',[min(Y) max(Y)],'MapLonLimit',[min(X) max(X)])
setm (gca,'ParallelLabel','on','MeridianLabel','on','LabelUnits','dm')
setm(gca,'MLabelLocation',5/60,'PLabelLocation',5/60)
setm(gca,'MlineLocation',5/60,'PLineLocation',5/60)
tightmap; colorbar('vert');
```



```
figure;
v=2:4:54;
c=isnan(slope);
hist(slope(~c),v);  xlim([0,50])
xlabel('slope'); ylabel('number of points')
```