速習 GMT　日本語版

Generic Mapping Tool (GMT)

2017+

OKINO, Kyoko
沖野郷子　東大大海研
okino@aori.u-tokyo.ac.jp

2017.3.8

目次

独習に必要な環境

UNIX もしくは UNIX-like の環境
GMT （このテキストは ver 5.1 準拠）
テキストエディタなにか
gv, GSView などの post script viewer

関連ウェブサイト

GMT Project Home　http://gmt.soest.hawaii.edu
GMT latest document　http://gmt.soest.hawaii.edu/doc/latest/index.html

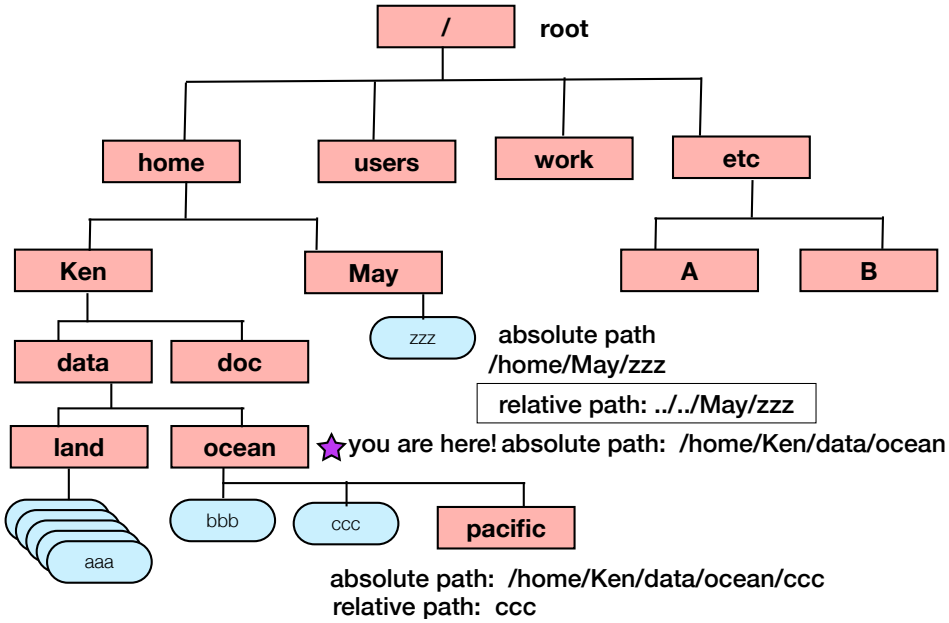
Global topography
https://www.ngdc.noaa.gov/mgg/global/global.html
(ETOPO1)

沖野による海底地形図作成講座
http://ofgs.aori.u-tokyo.ac.jp/~okino/gmtscripts/index.html

日本の GMT ユーザグループのサイト
http://www2.kobe-u.ac.jp/~kakehi/gmt-users-jp/

1　はじめに
　1.1　GMT の歴史
　1.2　GMT　とはどのようなものか
　1.3　　講習の目標

2　UNIXとawkの基礎の基礎
　2.1　UNIX　とは
　2.2　UNIXにおけるファイルシステム



2.3　最重要のコマンド

まず、UNIX 環境でターミナルウィンドウを開いて、コマンドがテキスト入力できる状態にしましょう。
まず、あなたが UNIX ファイルシステムのどこにいるか知るために、

　pwd↵

と入力。すると今いる場所(present working directory)が絶対パスで表示されます。次に

　ls↵

と入力すると、今いるディレクトリ下にあるファイルの一覧が示されるでしょう。
もし、各ファイルの名前だけでなく容量などの詳しい情報が知りたければ、オプション-l をつけます。

　ls　-l ↵

それでは、新しいディレクトリをつくって、確認してみましょう。

```
mkdir testdir↵
ls ↵
```

testdir というディレクトリができていますね？Mac-OS や Linux の通常のファイルブラウザでも確認してみるといいです。それでは、ファイルシステム上を移動してみましょう。今つくったディレクトリに移動するには

```
cd testdir↵
pwd↵
```

今いる場所のひとつ上（親）のディレクトリに移動する時は便利な..を使います。

```
cd ..↵
pwd↵
```

練習用に置いてあるファイルの中身をみてみましょう。

```
cat testmap.bash↵
```

このファイルのコピーをつくるには、以下のようにします。

```
cp testmap.bash test-copy.bash↵
ls -l↵
cat test-copy.bash↵
```

同じファイルができていますね？ファイル名を変更したいときは、mv コマンドを使います。これはファイルを移動する時にも使います。

```
mv test-copy.bash test-move.bash↵
ls -l↵
```

ファイルを削除したい時には

```
rm test-move.bash↵
rm test-copy.bash↵
ls -l↵
```

ここで非常に大事なこと。これまで使ったコマンドは、出力はすべて画面に出てきました。UNIX の世界ではこれを「標準出力が画面になっている」と言いますが、出力内容を画面ではなくて別のファイルに書き出すこともできます。これは出力先を画面からファイルに方向転換するので、リダイレクトと読んで、以下のような不等号マークで指定します。

```
cat testmap.bash > testout.bash↵
cat testout.bash↵
```

リダイレクトできていますか？ちょうどコピーしたのと同じになっていますね。

これまでやってきた作業の履歴を見るには

```
history ↵
```

とタイプしてみてください。番号と自分がタイプしたコマンド一覧を見ることができます。ここで↑キーをいくつか打ってみてください。↓も。現在のプロンプトに

過去のコマンドが表示されますね。これを使うと何度も同じコマンドを繰り返しタイプしなくても、↑だけで繰り返し実行ができます。ほぼ同じコマンドでちょっとだけ数値や文字を入れ替えたい、という時も、よく似たコマンドを呼び出して、今度は←キーと delete キーで一部だけ書き直せばよいのです。

ちなみに、もうひとつだけ Tips を。ほとんどの UNIX 環境では、ファイル名（ときどきすごく長い場合がある）の自動補完機能があります。ファイルの最初の何文字かを打って、esc キーをたたくと、うまくいけば残りのファイル名が補完されます。この２つでかなり作業がスピードアップかつ入力ミスがなくなります。

OK, それでは準備完了で lesson ディレクトリに移動して GMT を練習しましょう。

```
cd lesson ↵
```

2.4 ちょっとだけ*awk*を使う

AWK はテキストを扱うプログラミング言語で、UNIX 環境では標準的に装備されています。あるデータファイルから特定のデータを抜き出したり、書式を変えたりという作業には非常に便利です。たくさんの参考書やウェブサイトで AWK の多彩な機能を学びことができますが、ここでは例題ファイルを使って最低限の使い方を覚えましょう

ファイル“sample1.dat”がホームディレクトリにあるので中身を確認してください。

cat sample1.dat↵

ファイルは 10 行からなり、各行は９つの値（年/月/日/時/分/秒/経度/緯度/水深）が空欄（ブランク）を区切りとして入っています.
以下の通りタイプして結果を見てください。

awk ‘{print $7,$8}’ sample1.dat↵

全行の経度と緯度だけが出力されました。変数$X は各行の X 番目の列を指します。

演習 1: 時間 (時/分/秒)と水深を抜き出して新しいファイルを作りなさい。
(リダイレクト “>”を使う)
演習 2: 変数の数値演算(+, -, *, / など) もできます. 秒を使わずに (時、分=分+秒/60, 水深)を出力してみよう。

{ }で囲まれた部分が AWK のプログラムです（プログラムの実態はコマンド 1 行だけ)。もう少し複雑な作業をしたい場合には、いくつかのコマンド行をひとつのファイルの保存してプログラムファイルとして扱うこともできます。ホームディレクトリにある awk プログラム例“pos_min.awk”を見て下さい。

cat pos_min.awk↵

以下の内容になっていることがわかるでしょう;

```
{
lon_deg=int($7);
lon_min=($7-lon_deg)*60;
lat_deg=int($8);
lat_min=($8-lat_deg)*60;
print $1,$2,$3,$4,$5,$6,$lon_deg,$lon_min,$lat_deg,$lat_min;
}
```

int($X)：小数点以下の切り捨てを行う関数

この awk プログラムを走らせてみましょう。

awk –f pos_min.awk sample1.dat ↵

–f *filename*:　プログラムは　*filename* という名前のファイルに保存

AWK プログラムでは、C や Fortran 同様に “if else”, “do”などの判断や制御の文が使えるほか、正規表現によるテキスト操作ができます。例えば上記のプログラムをより一般的な状況で使えるようにするには、１）南緯や西経がでてきたらどうするか、2) 出力の桁数を制限したい場合にはどうするか、などを考えて、もっと高度な awk の機能を学ぶ必要があります。

3　GMT 最初の一歩：白地図を描く

3.1　地図の枠を描く　*psbasemap*　　-R　-J　-B

最初の地図（図の枠）として日本周辺図を描きましょう。以下の行を”正確に”コマンドウィンドウにタイプしてください。

```
gmt psbasemap -R135/145/30/40 –JM12 –Bf10ma1g30m –V > basemap1.ps↵
```

新しいファイル（出力ファイル）　***basemap1.ps*** が同じディレクトリにできていることを***ls –l*** コマンドで確認してください。出力はpostscript形式の図のファイルなので、gvコマンドを使って表示してみましょう。

```
gv basemap1.ps↵
```

“gv”ウィンドウが開いて地図の枠組みが表示されましたか？. “***gv***” はpostscriptファイルを画面に表示するプログラムです。GSViewなどの他のpostscript表示プログラムを利用しても構いません。GMTの“***psbamsemap***” コマンドはオプションの指示に従って地図枠を描きます。上記の例では、４つのオプションを指定しています。

-R　図の範囲　西端/東端/南端/北端　単位は度か度:分:秒
-J　地図投影法と縮尺　　アルファベットは投影法　Mはメルカトル図法
　　M*length*　でlengthはできあがりの図の横幅実寸
　　m*scale*　でscaleはいわゆる縮尺　例えば1:50000で5万分の１
-B　枠の目盛りなどの指定　単位なしだと度、分ならmつける
　　f: 目盛り間隔　a: 文字間隔　g: 経緯線間隔
-V　verbose（饒舌）オプション　メッセージをたくさん画面に表示する。つけなくても結果には変わりないが、エラーを起こした時などに原因がわかりやすいので普通は付けたほうがいい。

オプションや使い方はこまめにGMTの公式サイトにあたるのが一番。
　http://gmt.soest.hawaii.edu/doc/latest/index.html
このサイトでマニュアルやCookBook（詳しい説明や例題が出ている）などが公開されています。*CookBook 13 GMT Map Projections*のリンクをたどると、利用できる地図投影法のすべてが記載されています。

それでは以下の演習に挑戦。

演習 1: 図の西端を125°E に、北端を45°Nに変えましょう。
演習 2: 縮尺を1:10000000にしましょう。
演習 3: 図法をランベルト正積図法にしましょう。
演習 4: 図枠を以下の仕様に変えてみよう。
目盛りは1°, 文字間隔は5°, 経緯線間隔は 1°

3.2 海岸線を描く *pscoast* -D -W -G

それでは海岸線を描いてみましょう。”pscoast”が海岸線描画コマンドです。

```
gmt pscoast -R125/145/30/45 –JM12 –Bf1a5g5 –Di –Ggray
 –Wthin,black –V > coastmap1.ps↵
```

*gv*を使って*coastmap1.ps*を確認してください。

-D 海岸線の分解能 (f)ine h(igh) (i)ntermediate (l)ow
-G 陸域の色もしくはパターンの指定
-W 海岸線を描くペンの太さと色の指定
ペンの太さは以下の通り、色指定については *CookBook 22 Color Space*を見ましょう。

faint	0	thicker	1.5p
default	0.25p	thickest	2p
thinnest	0.25p	fat	3p
thinner	0.50p	fatter	6p
thin	0.75p	fattest	12p
thick	1.0p	obese	18p

演習 1:図の分解能を確認しましょう。Lowにするとどうなるか？
演習 2:陸域を緑にしてみましょう。
演習 3: 海岸線を太めの青線に変えましょう。
演習 4: GMT のオンラインマニュアルで ***pscoast*** の項目を読んで、国境線を入れて見ましょう。
演習 5: –B optionを削除してみましょう。何が起こるかな？

3.3 シェルスクリプトを使う　-K –O

GMTのコマンドを複数組み合わせて使いたい場合、コマンドを順番に実行していく必要があります。試してみましょう。まず以下を入力。

```
gmt pscoast -R125/145/30/45 –JM15 –Di –Ggray
 –Wthin,black –V –K > coastbasemap1.ps↵
```

ls –l コマンドで *coastbasemap1.ps* ができていることを確認したら、*gv* を使って図を見てみましょう。図の枠がありませんね？これは-Bオプションをつけなかったからです。それでは続けて図の枠だけを描かせます。

```
gmt psbasemap -R125/145/30/45 –JM15 –Bf1a5g5 –V –O >>
coastbasemap1.ps↵
```

!! ここで注意 !! 出力ファイル名は前と同じで、リダイレクトは”>”ではなく“>>”です。これは、前のファイルに追加書きするという意味です。>にすると、前のファイルは上書きされて内容は消されてしまいます。

それではもう一度*ls –l*で*coastbasemap1.ps*を確認してください。ファイルサイズが大きくなっていますね? ふたつのコマンドが続けて実行されて、その結果がひとつのファイルに続けて書き込まれたのです。*gv*で図を確認してください。

重要（かつ間違えやすいNo.1）は 以下の２つのオプションです。
Postscriptファイルは通常は「これから始まるよ」と「ここでおしまいよ」という記述がファイルの最初と最後に付きます。

-K　「ここでおしまい」を削除する＝このコマンドの後に別の出力が付きますよ、のサイン

-O　「これから始まるよ」を削除＝このコマンドの前には別の出力がありましたよ、のサイン

つまり、コマンド行が複数あるときは、最初のコマンドには-K, 最後のコマンドには-O, 間にはさまれたコマンドには-K –Oをつける必要があるのです。

シェルスクリプト

今のようにコマンド行をいちいち順番に打ち込むと、とかく間違い易く、途中で間違うと修正できないので、最初からやり直しになってしまいます。そこで、シェルスクリプトを使うのがスマート。シェルスクリプトとは、実行するコマンドなどが

まとめて書いてあるファイルです。ホームディレクトリにあるシェルスクリプト例として*plot_coast.bash* をテキストエディタで開いてください。ここでは *gedit* を使いますが、他のテキストエディタでも構いません。

```
gedit plot_coast.bash↵
```

```
#! /bin/bash

# parameter setting
region=125/145/30/45
proj=M15
frame=f1a5g5
psfile=coastbasemap2.ps
#
gmt pscoast –R$region –J$proj –Di –Ggray –Wthin,black –K –V > $psfile
gmt psbasemap –R$region –J$proj –B$frame –O –V >> $psfile
#
```

シェルスクリプトの中身は前章で打ち込んだコマンドと同じですね。ただし、オプションで指定する種々のパラメタを、前半で変数として自分で定義しています。定義した変数をコマンドで使うときは$をつけます。このファイルを実行するには以下のようにタイプするだけ。

```
./plot_coast.bash ↵
```

coastbasemap2.ps がうまくできたか、図を表示させて確認してください。

シェルというのは、OSと対話するためのインターフェイスで、ある種の作業環境だと思ってください。シェルにも種類があり、多くのLinuxマシンではbashと呼ばれるシェルを採用していますが、Cシェル（csh）, sh, ksh, tcshなどのシェルも使われています。シェルが違うと例えば上のスクリプトの変数指定の方法などが若干違う（方言みたいなものだ）ことがあります。マシンに複数のシェル環境が用意されていることもあるので、指定したい場合は、上記のスクリプトの1行目のように、特別なサイン#!を使って、シェルを明示的に指定します。このへんはシステムによって異なるので、シェルスクリプトがGMTのエラー以外でうまく動かなかったら、マシンの管理者などの詳しい人に相談してください。

それでは演習してみましょう。

演習:　最初の例題シェルスクリプト *plot_coast.bash* をコピーしてあなたのシェルをつくりましょう（好きな名前をつけてよいですが、拡張子.bash をつけることを薦めます）。地図の範囲、図法や縮尺、ペンの色や枠の仕様などを自由に変えて、好きな白地図をつくってみてください。

4. GMT 第２段階: 地形/水深図を描く

4.1. グリッドデータを使う　*grdinfo*

GMTはアスキーやバイナリのリストのほか、格子状（グリッド）データを扱うことができます。グリッドデータの書式としては、標準ではnetCDFと呼ばれる書式が使われています。グリッドデータを使って地形図を描いてみましょう。入力データとしてjapan_etopo2.grd がディレクトリ内にあるはずです。netCDFのファイルはバイナリなので、*cat* コマンドやテキストエディタで内容をチェックすることはできません。かわりにGMTのgrdinfoコマンドを使って内容を知ることができます。

```
gmt grdinfo japan_etopo2.grd ↵
```

グリッドの詳細仕様（グリッドの範囲、グリッドセルの大きさ、値の最大最小値などなど）が表示されます。

4.2. 等高線（等深線）を描く　　*grdcontour*　-C

グリッドデータと*grdcontour* コマンドを使って等高線（等深線）を描いてみましょう。もういちど全章で使ったシェルスクリプト*plot_coast.bash* をコピーして新しいシェルスクリプトを作りましょう。作ったシェルスクリプトをテキストエディタで開いて、次のように修正・追加をします。赤字の部分が修正・追加したところです。

```
#! /bin/bash
# parameter setting
region=125/145/30/45
proj=M15
frame=f1a5g5
grdfile=japan_etopo2.grd
psfile=contour.ps
#
gmt grdcontour $grdfile –R$region –J$proj –C200 –A1000 –K –V > $psfile
gmt pscoast –R$region –J$proj –Di –Ggray –Wthin,black –K –V –O >>
$psfile
gmt psbasemap –R$region –J$proj –B$frame –O –V >> $psfile
#
```

スクリプトを走らせてみて、結果を表示して確かめましょう。 *grdcontour*　では -C オプションは等値線の間隔を指定し、-Aオプションは等値線に文字（数字）を入れる間隔をしています。前章で覚えた-Kと－Oオプションを忘れずに。出力ファイル名は好きなようにつければよいのですが、入力グリッドの名前を組み合わせるなど、内容がわかる名前を選ぶべきですね。

演習: 等値線の間隔を500mにしてみましょう。

4.3. 彩色図を描く　*makecpt　grdimage　psscale*

次に ***grdimage*** コマンドを使って、色で値（この場合は標高/水深）を示す図を描きましょう。ここでは、どの値が何色になるかを指定したカラーパレットが必要になります。カラーパレットを作るコマンドは***makecpt*** で、GMT世界では***.cptという拡張子をつける慣習です。GMTには多くの標準的なカラーパレットが装備されていて、普通はその中から好きなパレットを選んで、入力データの範囲に合わせて最大最小値を指定して描画用パレットをつくります。
さきほどのシェルスクリプトを開いて（もしくはコピーして別スクリプトを作って）、***grdcontour*** コマンドの行頭に#を追加し、次に***grdimage*** コマンドを追加します。#をつけることを"コメントアウト"と言います。シェルスクリプトでは行頭に#があると、その行は単なるコメントでコマンドではないと解釈されます。
grdcontourを使わない時は、もちろん行をまるごと削除してもよいのですが、先頭に#を付け加えることによってコメント行だと思わせて実行をとばすことができます。このほうが後で等値線を描きたくなった時に便利です。でもあまり#を多用すると字面がごちゃごちゃでわかりにくくなることもあります。スクリプトの説明や作成日付などもコメント行として記録しておくと便利ですね。

```
#! /bin/bash
# parameter setting
region=125/145/30/45
proj=M15
frame=f1g5a5
grdfile=japan_etopo2.grd
psfile=colorfill.ps
cptfile=topocol.cpt
#
# making color table
gmt makecpt –Ctopo –T-8000/4000/1000 –Z > $cptfile
#
# drawing color-fill map
gmt grdimage $grdfile –R$region –J$proj –C$cptfile –K –V > $psfile
gmt psscale –D16/3/10/0.5 –C$cptfile –K –O >> $psfile
# gmt grdcontour $grdfile –R$region –J$proj –C100 –A1000 –K –V >
$psfile
gmt pscoast –R$region –J$proj –Di –Ggray –Wthin,black –K –V –O >>
$psfile
gmt psbasemap –R$region –J$proj –B$frame –O –V >> $ofile
```

ファイルの書き換えが終わったら、スクリプトを実行して、出力ファイルを描画して確認してください。***makecpt*** コマンドでは、 -C オプションは標準のカラーパレット名を示し（ここでは.cptはつかない）, -Tオプションでそのカラーパレットの両端と間の指定間隔を-T min/max/intervalで指定します。. *CookBook 26 Of colors and Color Legends* を参照すると、どのような標準カラーパレットがあるかわかります。makecptの出力は自分のデータにあわせたカラーパレット名で、自由に命名して構いませんが、標準パレットと全く同じ名前にするとトラブルの元です。
grdimage コマンドでは、-C オプションは彩色に使いたいカラーパレットのファイル名の指定です。コマンド***psscale*** は地図とあわせて色の凡例をプロットします。ここでーDオプションは凡例のカラーバーの位置と大きさを指定しています。
この例題スクリプトでできたファイルを表示させると、陸域がグレーになっていると思います。もし陸域にも海域同様の彩色をしたい場合は、***pscoast***の-Gコマンドを消してください。GMTはコマンドを実行した順番に描画の指示を出力していくので、後から実行されたコマンドが上書きされていきます。元のスクリプトでは、grdimageを実行した段階では、海域陸域ともに彩色されていたのですが、その次にpscoastコマンドで陸域をgrayに指定してしまったので、上から塗りつぶしているのです。-Gコマンドを消すと、pscoastは陸域の塗色をしませんから、元の彩色が表に出てくるわけです。

演習 1：オンラインマニュアルを参照して、makecptコマンドの-Zオプションの意味を調べましょう。そして-Zオプションを消すとどうなるか試してみましょう。
演習 2: カラーパレットを変えてみましょう。違う標準パレットを選んでごらんなさい。標準パレットは名前と簡単な説明であれば、コマンド画面でmakecptとタイプすると表示されますし、CookBookにはカラーの凡例が出ています。
演習 3: 等深線と彩色図を重ねて描くにはどうしますか？grdcontourのコメント#を消して、それから何が必要でしょうか？

4.4. 陰影図を描く *grdgradient*

地形に光をあてた時にできる影を描いて、3次元的に地形を表示させてみましょう。GMTでは、いわゆる3次元図(鳥瞰図)をgrdviewコマンドで描くこともできますが、ここではgrdimageの-Iオプションを使った陰影図を描きましょう。陰影図の場合、視点が通常の地図同様に真上ですから、死角はなく、位置や距離・方角を直感的に理解することができます。まず、grdimageコマンドを実行する前に、grdgradientを使って入力データの勾配（この場合は地形の傾斜）を計算して、陰影（英語ではillumination=照明）のデータを作る必要があります。前に使ったフシェルスクリプトを開いて、***grdcontour***はコメントアウトし， ***grdgradient*** コマンド行を挿入、そして***grdimage*** コマンドに作成した陰影ファイルを指定する-Iコマンドを追加し

ます。もし、カラーパレットは以前作成したものを使うならば、makecptはコメントアウトしたほうがいいですね。

```
#! /bin/bash
# parameter setting
region=125/145/30/45
proj=M15
frame=f1g5a5
grdfile=japan_etopo2.grd
intfile=japan_etopo2_90.int
psfile=colshade.ps
cptfile=topocol.cpt
#
# making color table
# makecpt –Ctopo –T-8000/4000/1000 –Z > $cptfile
#
# making illumination/shade intensity grid
gmt grdgradient $grdfile –A90 –Ne0.6 -G$intfile
#
#drawing color-fill map
gmt grdimage $grdfile –I$intfile –R$region –J$proj –C$cptfile –K –V >
$psfile
gmt psscale –D16/3/10/0.5 –C$cptfile –K –O >> $psfile
# gmt grdcontour $grdfile –R$region –J$proj –C100 –A1000 –K –V >
$psfile
gmt pscoast –R$region –J$proj –Dh –Wthin,black –K –V –O >> $psfile
gmt psbasemap –R$region –J$proj –B$frame –O –V >> $psfile
#
```

実行して、図を確認してください。***grdgradient*** コマンドでは、 -A オプションは光源の方位（北から時計回りに角度で示す）を指定します。つまり–A90は東から光をあてた場合の陰影であり、東向きの斜面は明るく、西向きの斜面は暗くなります。その結果として、南北の構造（崖など）が強調される効果をもたらします。これは地質構造の解釈に非常に役にたつ（たとえばある走向の断層を抽出するとか）のですが、時として光と平行な方向の構造を見落とすことにもなるので注意が必要です。-Nオプションは正規化のファクターで詳細はマニュアルを見て勉強しましょう。一般的な地形図の場合は-Ne0.6が試しにやってみるには良い選択です。

演習 1： 試しに-Ne1.0で何が起こるかやってみましょう。
演習 2: 光源の方向を変えてみましょう。
演習 3: 等値線も重ねて描かせてみましょう。

5. GMT 第３段階：シンボルや線を描く

5.1. シンボルや線を描く *psxy*

GMTはアスキーやバイナリのリストを読み込んで、任意の場所にシンボルを描いたり、線を引くことができます。白地図に試料採取点と測線を描く練習をしてみましょう。練習用のデータとして、**sample_loc.dat** と **line_loc.dat** が用意されています。これらはアスキーデータなので、***cat*** コマンドで内容を確認してください。経度と緯度の羅列だということがわかります。

いつもの最初のスクリプト ***plot_coast.bash*** をコピーして新しいシェルスクリプトをつくりましょう。新しいスクリプトにはpsxyコマンドを追加して、星印と線を描いてみます。

```
#! /bin/bash

# parameter setting
region=125/145/30/45
proj=M15
frame=f1a5g5
pointfile=sample_loc.dat
linefile=line_loc.dat
psfile=locationmap.ps
#
gmt pscoast –R$region –J$proj –Dh –Ggray –Wthin,black –K –V > $psfile
gmt psxy $pointfile –R$region –J$proj –Sa0.2 –Gred –K –O –V >> $psfile
gmt psxy $linefile –R$region –J$proj –Wthick,blue –K –O –V >> $psfile
gmt psbasemap –R$region –J$proj –B$frame –O –V >> $psfile
#
```

スクリプトを実行して、図を確認しましょう。 ***psxy*** コマンドでは、-Sオプションはシンボルの種類（アルファベット）と大きさ（数字）を指定します。-S オプションがない場合は、GMTは入力位置データをつないだ直線を引きます。 色を指定するには-G (シンボルを塗る)オプションや-W (ペンの色や太さ)オプションが必要になります。多彩なシンボルの詳細についてはオンラインマニュアルの***psxy*** を調べましょう。

演習 1: シンボルの種類や大きさを変えてみましょう。
演習 2: シンボルや線の色を変えてみましょう。シンボルを描くときにシンボルの外枠も-Wオプションで指定してみましょう。

6 GMT 第4段階: 時系列データを表示する

6.1 簡単なXY座標のグラフを描く *psxy –Jx,X gmtinfo*

まず最初に、簡単なXY座標のグラフとして時系列データをプロットしてみましょう。ではいわゆる地図の代わりに種々のグラフを描画することができます。 ディレクトリにある練習用時系列データ***sample_grav.fa*** を使います。このファイルはアスキー形式で、海域で観測した重力異常などが入っています。***cat*** か ***more*** コマンドを使ってファイルの内容を確認してみましょう。

more sample_grav.fa↵

ファイルの各列の内容は以下の通りです。
year, month, day, hour, min, sec, latitude, longitude, absolute gravity, depth, Etovos-correction, normal-gravity, free-air anomaly, heading, record number,

ここではまず“depth（水深）”をpsxyコマンドを使ってプロットしましょう。縦軸は水深、横軸はレコード番号とします。シェルスクリプト例の***plot_timeseries_rec.bash***を走らせ、出力されるgrav_plot_num.psを表示してください。表示を見ながら、シェルスクリプトを見ていきましょう。 Awkコマンドの使い方を覚えていますか？

```
#   plot time-series data
#
region=0/6000/-50/50
proj=X18/5
xframe=f100a500g500
yframe=f10a20g20
infile=sample_grav.fa
tmpfile=sample_num_fa.dat
psfile=grav_plot_num.ps
#
# extract record number and free-air anomaly using awk
#
awk '{print $15,$13}' $infile > $tmpfile
#
gmt psxy $tmpfile -R$region -J$proj -Sp   -Gblue -K -V > $psfile
gmt psbasemap -R$region -J$proj -Bx$xframe -By$yframe -O -V >>
$psfile
#
```

各軸の範囲は適当ですか？実際にはシェルスクリプトを書く際に、入力データの範囲を確認しておいたほうがよいです。一般的な（グリッド出ないアスキーの）データを確認するには、*gmtinfo* コマンドが使えます。

```
gmt gmtinfo sample_grav.fa↵
```

各列の<最小値/最大値>が表示されるはずです。

演習 1: heading（船の方位）をプロットしましょう。.
演習 2: 最初に行った水深プロットでは、値の大きい（深い）ほうがグラフの上にくるので、直感的に裏返しの地形になっています。直感的に理解しやすいように、縦軸をひっくり返す方法を考えてください。

6.2 横軸を”時刻”表示する

ここまでは横軸としてレコード番号を使っていました。時系列プロットとして横軸を時間（時刻）にするほうが、欠測などもわかるのでよいのですが、60進法が入ってくるのでちょいと面倒です。時系列データをプロットする方法として、例えば観測開始時（データ開始時）からの通秒を計算して横軸にするという方法があります。これは簡単に計算できるので使うこともありますが、目盛りを振っても直感的に何時何分なのかがわかりません。そこで、少々面倒ですが、きちんと日付や時刻が横軸に表示される方法を試してみましょう。

シェルスクリプト例は*plot_timeseries_hour.bash* です。このスクリプトは下のように、２つのパートに分かれています。１）入力の（year,mon,day,hour,min,sec）データを、awkを使ってGMTの標準時刻フォーマット（yyyy-mm-ddThh:mi:ss）に変更する, 2）psxyの図枠指定オプションで−Btとして時刻軸だと認識させる。最初の段階では、fa2ts.awkというawkのスクリプトを使っています。.bashと.awkそれぞれのスクリプトを良く読んで、何をしているか理解しましょう。

fa2ts.awk

```
{
printf("%4d-%02d-%02dT%02d:%02d:%02d %7.1f\n",$1,$2,$3,$4,$5,$6
,$13);
}
```

```
#   plot time-series data
#
infile=sample_grav.fa
tsfile=sample_grav.ts
start_ts=2016-01-23T00:00:00
end_ts=2016-01-24T00:00:00
region=$start_ts/$end_ts/-100/100
proj=X18/5
xframe=f1ha4hg4h
yframe=f10a20
psfile=grav_plot_ts.ps
#
# reformat input file using awk program "fa2ts.awk"
#
awk -f fa2ts.awk $infile > $tsfile
#
gmt psxy $tsfile -R$region -J$proj -Sp -Gblue -K -V > $psfile
gmt psbasemap -R$region -J$proj -Bx$xframe -By$yframe -O -V >>
$psfile
```

演習 1:　データの質を入念にチェックするために、グラフの一部を拡大してみましょう。例えば12:00 -16:00 までのデータだけを表示させ、細かく目盛りや文字を入れて見ましょう。時刻軸の目盛りの振り方については以下のサイトを見ること。
http://gmt.soest.hawaii.edu/doc/5.1.0/gmt.html#the-common-gmt-options

7　GMT 第5段階: 簡単な演算とフィルター処理

7.1　２つのグリッドデータの差を調べる　*grdmath*

グリッドデータ同士の簡単な演算（加減乗除等）はgrdmathコマンドで行うことができます。ディレクトリにある２つのファイルtarama_mb.grd and tarama_etopo.grdを例に見てみましょう。まず、２つのグリッドファイルの仕様をgrdinfoで確認してください。

```
gmt grdinfo tarama_mb.grd↵
gmt grdinfo tarama_etopo.grd↵
```

グリッドの範囲とグリッドの間隔が共通ですね？この２つが共通ならば、ファイル同士の演算ができます。ちなみに、tarama_mb ファイルはソナーの観測で実際に得られた水深、tarma_etopo file は人工衛星の海面高度計から得られる重力異常から推定した水深が入っています。その差をgrdmathで見てみましょう。

```
gmt grdmath tarama_mb.grd tarama_etopo.grd SUB =
tarama_diff.grd↵
```

演習 1:　元の２つのファイルと差のファイルを描画しましょう。差はプラスマイナスがありますから、カラーパレットは何を選ぶのが適切でしょうか。
演習 2:　２つのグリッドの違いを、単純な差ではなく、片方の（.etopo）ファイルの値の百分率（%）で計算して図化してみましょう。

7.2　1次元データにフィルタをかける　*filter1d*

GMT にはグリッドデータや通常のデータに種々のフィルタを施すコマンドがあります。ここでは、最も簡単な1次元データにフィルタをかけるコマンド***filter1d***を使ってみましょう。前章で作った **sample_num_fa.dat** を練習用ｈに使います。シェルスクリプト例***test_filt.bash*** を実行してウィンドウ長50のBoxcar フィルタ（移動平均）をかけてみます。
描画スクリプト ***plot_timeseries_rec.bash*** をコピーして新しいシェルスクリプトを作成し、下の赤字の部分を修正・追加してください。そのあと実行して、結果を確認しましょう。

```
#   plot time-series data
#
region=0/6000/-50/50
proj=X18/5
xframe=f100a500g500
yframe=f10a20g20
infile=sample_grav.fa
tmpfile=sample_num_fa.dat
psfile=grav_plot_num.ps
#
# extract record number and free-air anomaly using awk#
#
# awk '{print $15,$13}' $infile > $tmpfile
#
gmt psxy $tmpfile -R$region -J$proj -Sp   -Gblue -K -V > $psfile
gmt psbasemap -R$region -J$proj -Bx$xframe -By$yframe -K -O -V >>
$psfile
#
#   box car filtering
#
filtfile=sample_num_fa_b50.dat
gmt filter1d $tmpfile -Fb50 -N0 -V > $filtfile
gmt psxy $filtfile -R$region -J$proj -Sp   -Gred -O -V >> $psfile
```

演習:　フィルタの種類と長さをいくつか変えて試してみましょう。フィルタについての詳細はオンラインマニュアルを調べましょう。

8. GMT 第 6 段階: 2 種類のデータを重ねて表示する

時には2種類の異なるデータを同時に表示して比較してみたい場合があるかもしれません。ここでは、重力異常分布と地形の関係を調べることを考えてみましょう。使うのは、**japan_etopo2.grd** （地形）と **japan_altgrv.grd** （重力）のファイルです。

8.1 重力異常を彩色で示し、地形の等値線を重ねる

演習 1: 重力異常の彩色図（陰影なし）をつくりなさい。前に作成した地形の彩色図をつくるスクリプトをコピーして、重力異常図用に書き換えましょう。カラーパレットは正と負が区別できるものがよいかもしれません。値の範囲を知るために、スクリプトを書く前にgrdinfoコマンドで異常値の範囲を知っておく必要があります。

演習 2: 上記の図に地形の等値線を書き足します。**grdcontour**コマンドを **grdimage** コマンドの後に追加します。適切な等値線間隔 (—Cオプション) を選びましょう。

演習 3: さらに、もう1行grdconroutを追加して、重力異常の等値線も別の色で重ねてみましょう。

8.2 3次元の地形に重力異常に対応した色を重ねる

先に陰影図を作成した時には, ***grdimage*** コマンドは２つのグリッドデータを入力ファイルとして必要としました。ひとつは彩色用のグリッド（地形データ）、もうひとつは陰影のグリッド（grdgradientで作成）です。ここで、彩色用のグリッドとして地形のかわりに重力異常ファイルを指定してみましょう。陰影は地形から計算した陰影のままです。こうすると、ちょうど地形の凹凸の上に重力異常をかぶせたような表示になります。

演習: 陰影図作成のシェルスクリプトを改造して、上述のような重力異常を地形の凹凸に重ねた図を描きましょう。適切なカラーパレットを指定すること。

9 GMT 次の一歩: グリッドデータをつくる

これまでの演習では、地形や重力のグリッドデータは公開されているデータセット（の一部）を利用してきました。さまざまな解析をするにあたって、等間隔のデータ（2次元に分布する場合はグリッドデータ）が前提となることが多くあります。この演習の最後に、一般的な観測データ（離散値）からグリッドデータをつくる方法を学びましょう。

9.1 グリッドを設計する

自分でグリッドデータを作成する場合、まず最初に必要なことはグリッドの範囲と格子間隔を適切に決めることです。格子間隔をどのようにとるか、一般的な規則があるわけではありません。はじめて扱うタイプのデータの場合は、まず*gmtinfo*などを使ってデータの最大最小値を確認して仮の範囲を定め、次に*psxy* を使ってデータをプロットしてみる（適当な小さなシンボルもしくはドットを使うとよいです）ことを薦めます。これによって、元のデータの分布がわかります。なるべく密に格子をつくりたい（格子間隔を小さくしたい）場合は、分布を見て、格子１つに元データがひとつ入る程度に設計するという考え方ができます。ただし、この場合は元データの値のばらつきが強くグリッドデータに反映されます。もっと大きな格子を設定すると、格子点の値（グリッドデータの値）は、範囲内の加重平均になるので、より安定したなめらかなグリッドデータができます。格子間隔をどのように選ぶかは、何をやりたいか、どのような解析を目指すのか、によるのです。

練習用データ ship.xyzを使って演習します（このデータはGMTのTutorialセットに入っているものです）。このファイルには、カリフォルニア沖の水深データが入っています。データは不均質にばらついています。データの書式と最大最小値を以下のように確認してみましょう。

```
head ship.xyz ↵

gmt gmtinfo ship.xyz↵
```

シェルスクリプト例 *gridmaking.bash*を開いて、そのまま実行してみてください。このスクリプトの後半はコメントアウトされているので、前半の*psxy*によるプロットだけが実行されます。出力ファイル point_plot.psを確認してみましょう。.

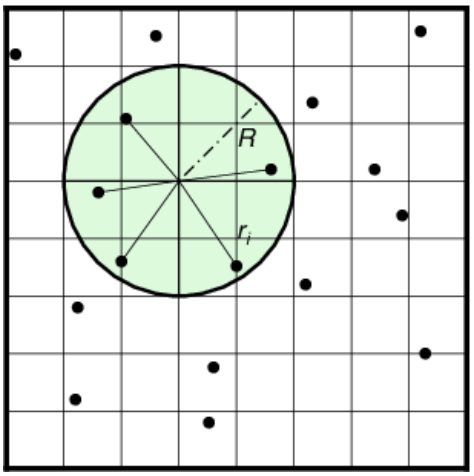
```
# parameter setting
region=245/255/20/30
proj=M15
frame=f10ma30mg1
xyzfile=ship.xyz
grdfile=ship_5m.grd
psfile=ship_cont.ps
#
# first step: just plot
gmt psxy $xyzfile –R$region –J$proj –Sp –K –V > $psfile
gmt pscoast –R$region –J$proj –Dh –Ggray –Wthick,black –K –V –O >>
$psfile
gmt psbasemap –R$region –J$proj –B$frame –O –V >> $psfile

#
# second step: gridding
#
#grdint=5m
#sradius=40k
#
#gmt nearneighbor $xyzfile –R$region –I$grdint –S$sradius –G$grdfile –V
#gmt grdcontour $grdfile –R$region –J$proj –C200 –A1000 –K –V >
$psfile
#gmt pscoast –R$region –J$proj –Dh –Ggray –Wthick,black –K –V –O >>
$psfile
#gmt psbasemap –R$region –J$proj –B$frame –O –V >> $psfile
#
```

9.2 グリッドを設計する　*nearneighbor*

それでは、前節で表示した**ship.xyz**の離散データから、格子サイズが5' x 5' (角度の分) のグリッドデータをつくってみます。前章で作った***gridmaking.bash*** を開いて、前半をコメントアウト（行頭に#をつける）し、後半のsecond stepとある部分の#を削除しましょう。これで後半だけが実行されるようになります。ここで使われている ***nearneighbor*** コマンドは、任意の分布のデータ(x,y,z) を使ってnearest neighbor algorithmというアルゴリズムで格子点値を決定します。ここでは、ある格子点の値として、その格子点を中心とするある範囲内に存在するデータ群の加重平均を採用するものです。下図のように、格子点から設定範囲（R）の円を描き（緑の円）その円をいくつかに分割します。デフォルトでは90°ずつの4象

限に分けます。円内の各象限の中で、一番中心に近い点(nearest point)を選び、それらの加重平均を格子点値とします。平均の重みは格子点とnearest pointの距離r_iの関数とします。



(from GMT cookbook)

それではスクリプトを走らせて、グリッドデータと等深線図を作成してみましょう。

グリッドデータを作成するコマンドは***nearneighbor***だけではありません。多用されるもうひとつのコマンドは ***surface*** です。このコマンドはスプライン関数を用いて格子点値をつくるものです。イメージとしては元データの場所にデータ値の高さの柱が立っていて、その柱群の上に布をかぶせて張力を適当にかけて曲面をつくり、格子点における曲面の高さを格子点値とする感じです。***nearneighbor***コマンドでは、平均する範囲内に元データが存在しない時には格子点値はNaN（計算機上でデータがないことを示す）となりますが、***surface***の場合は布をかぶせるので範囲内にいちおうすべて値が入ります。従って、なめらかで、かつ欠損のないグリッドをつくるには***surface***が適しており、実際には重力や磁力などのポテンシャル場の解析ではこちらがよく使われます。