速習 GMT　日本語版

Generic Mapping Tool (GMT)

2021 Modern Mode

GMT は 30 年以上の歴史があり, ver.5 までの 30 年間は構造としては大きな変更をせずに, 主に新しい機能やオプションの追加による機能拡張が行われてきました．2020 年に発表された ver.6 では，これまでと異なる新しいスタイルとして modern mode という形式（記法？）が導入されています．Modern mode ではこれまで初心者にとって間違いやすく面倒だった点が解消されており，今から新しく学ぶのであれば modern mode での利用がお勧めです．このマニュアルは modern mode に即したマニュアルです．一方, 従来の記法（classic mode とされる）のスクリプトも，ver.6 の環境で問題なく動きますので，これまでの資産を利用したい方はそのまま以前のスクリプトが使えます．

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

2021.7

目次

9　GMT もう一歩：グリッドデータをつくる

9.1　グリッドを設計する

9.2　グリッドデータを作成する　*nearneighbor*

独習に必要な環境

UNIX もしくは UNIX-like の環境（MacOS，windows-Cygwin）

GMT ver.6

テキストエディタなにか

関連ウェブサイト

GMT Project Home　https://www.generic-mapping-tools.org

GMT Module list　https://docs.generic-mapping-tools.org/6.2/modules.html

GMT Cookbook.　https://docs.generic-mapping-tools.org/6.0/cookbook.html

沖野による海底地形図作成講座(2021.7　この夏更新予定)　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　（あまり更新されてない）
http://www2.kobe-u.ac.jp/~kakehi/gmt-users-jp/

1　はじめに

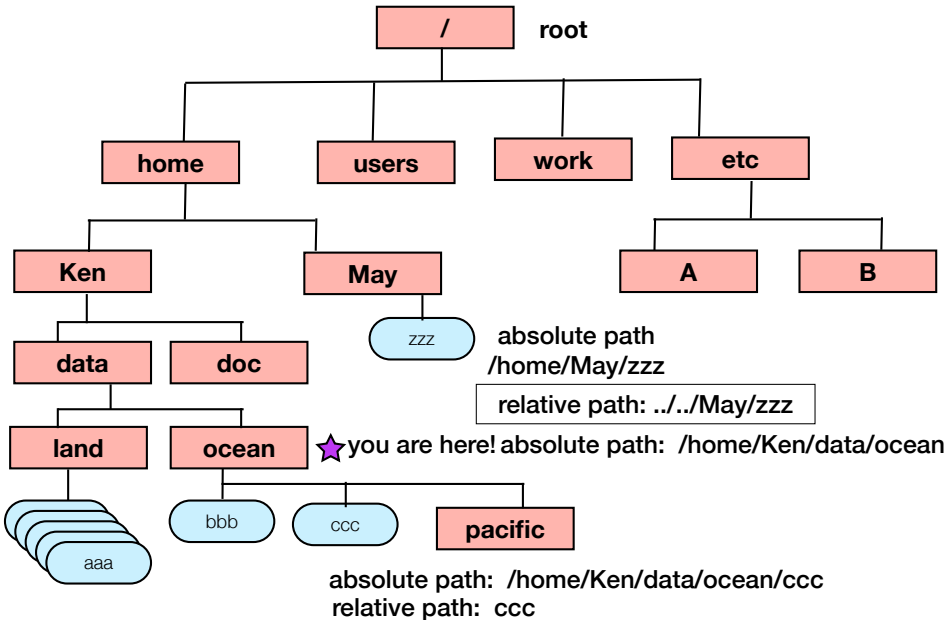
GMT はハワイ大学のグループにより 1980 年代末に開発されたプログラム群です．Generic Mapping Tool の名前の通り，主に地図上に情報をプロットすることを目的としてスタートしました．UNIX 上の C 言語で書かれており，当初からソースコードで公開されていました．その後，30 年以上にわたって世界の地球科学者によって使い続けられ，内容も進化拡張してきています．基本は，アスキーデータもしくは netCDF 形式のグリッドデータを読み込んで，図のファイルを出力する，という形ですが，図化の手前のデータ処理のためのプログラム（モジュール）も多数含まれます．使い方としては，ターミナルからテキストで命令を打ち込む形をとります．

現在(2021 夏)のバージョンは ver.6.2 ですが，ver6 になった時に比較的大きな変更がありました．Ver.5 以前の使い方もそのまま可能（classic mode）ですが，このマニュアルは ver.6 以後の新しい使い方（modern mode）に沿ったものとなっています．

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 lesson↵
ls ↵
```

lesson 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，自分のホームディレクトリの下に lesson ができていることを確認してください. 次に，練習用データをホームページからダウンロードしで，この lesson ディレクトリの下に置いてください.

それでは，ターミナル上で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 lesson↵
pwd↵
ls↵
```

ダウンロードした練習用ファイルのなかに，sample1.dat がありますね.

このファイルの中身をみてみましょう.

```
cat sample1.dat↵
```

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 sample1.dat sample2.dat↵
ls -l↵
cat sample2.dat↵
```

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 sample2.dat sample3.dat↲
ls -l↲
```

ファイルを削除したい時には

```
rm rample3.dat↲
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，リダイレクトと呼ばれ，以下のような不等号マークで指定します．

```
cat sample1.dat   > sample_out.dat↲
cat sample_out.dat↲
```

リダイレクトできていますか？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 キーとか tab 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，それではもうひとつ事前準備をしましょう．

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

もういちどファイル “sample1.dat”の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　練習用データのなかの “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．これは，緯度経度を度から度，分に表示を変えるという作業ですね．

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．

それではいよいよ GMT を使って地図を書きましょう．

3　GMT 最初の一歩：白地図を描く

3.1　スクリプトの構成

GMTには，さまざまなデータ処理をするモジュール（コマンド）もありますが，主となるモジュールは「図を描く」モジュールです．私たちがつくりたい図は，普通はさまざまな構成要素（図の枠，海岸線，等高線，とか）から成り立っています．GMTのモジュールは基本的にはそれぞれの構成要素を描くモジュールになっていて，複数のモジュールを組み合わせて複雑な（描きたい）図を作成していくことになります．まず非常に単純なケースから見ていきましょう．ダウンロードした練習用データのなかの，plot_basemap.bashファイルを見てみます．cat コマンドでもよいし，Macのテキストエディタ，WindowsのNotePad, linuxのエディタなどで開いても構いません．中身はこんな感じですね？

```
gmt begin basemap1
        gmt basemap -R135/145/25/45 -JM12 -B -V
gmt end show
```

GMTのモジュールはすべてgmt XXXXの形で使います．最初の行は，beginモジュールということですね．基本の構成としては，***begin***で図の作成の開始を宣言していて，***end***（この場合は3行目）で図の作成終了を宣言しています．このbeginとendに挟まれた行が，図の構成要素を描くモジュールです．この場合は後述するように要素はひとつ．

beginモジュールには少なくともひとつの引数（モジュール名の後に書くべき文字とか数字）が必要で，それが作成される図のファイル名になります．この例の場合は，図のファイル名がbasemap1.pdfとなります．また，endモジュールの後にshowという引数がある場合は，図のファイルが作成された後に，画面にその図を表示されます．では，beginとendに挟まれた本体はなんでしょう？

3.2　地図の枠を描く　*basemap*　　-R　-J　-B

2行目の***basemap***は図枠を描くモジュールです．やってみたほうが早いので，このファイルplot_basemap.bashを走らせてみましょう．このファイルに書かれたスクリプトを実行するにはいくつか方法がありますが，ここでは以下の通りターミナルでタイプしてみてください．

```
bash plot_basemap.bash↵
```

これは，bash環境でplot_basemap.bashの内容を実行してください，といった意味です．bash環境については，別途勉強してね．
地図の枠が表示されましたか（紙の縦置き，横置きがあるので，横向きになっているかもしれませんが）？
それでは，***basemap***モジュールのオプションを見ていきましょう．GMTの各モジュールのオプションは一記号からはじまります．ここでは，4つだけオプションを指定していますね．

-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．このファイルでは横幅12cmの図をつくっている．
m*scale*　でscaleはいわゆる縮尺指定も可　例えば1:50000で5万分の１
-B　枠の目盛りをつける．自分で目盛り間隔を指定したい場合は例えば
-Bf1a2g2などと-Bの後ろに詳細を記述する．
f: 目盛り間隔　a: 文字間隔　g: 経緯線間隔
＊ 上の例は，1度ごとの目盛り，2度ごとに文字入れ，1度のグリッド線
ファイルの例は，詳細設定をしていないので，システムが適切と思われる　目盛りを勝手につけてくる
-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．

各モジュールの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．
https://docs.generic-mapping-tools.org/6.2/modules.html
絶対必要で，かつ一番難しいのが-Jの地図図法と縮尺の選択です．これは，GMT CookBook 6　*GMT Map Projections*に例とともに詳しく書かれています．

演習 1: 図の西端を125°E 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角円錐図法（Lambert conic conformal）にしてみましょう．投影中心と標準緯線を2本設定する必要があります．
演習 4: 図枠を以下の仕様に変えてみましょう
目盛りは1°，文字間隔は10°，経緯線間隔は 5°

3.3 海岸線を描く　*coast*　-D -W -G

それでは次に海岸線を描いてみましょう. ***coast***が海岸線描画コマンドです. さきほどのplot_basemap.bashをテキストエディタで開いて，1行加えてみましょう. 出力図の名前もbasemap2に変えてみます.

```
gmt begin basemap2
        gmt basemap -R125/145/25/45 -JL135/35/30/40/12 -Bf1a10g5 -V
        gmt coast –Di -Ggray -Wthin,black
gmt end show
```

加えたら，ファイルを保存して（これ忘れがち），そのあとターミナルで実行します.

```
bash plot_basemap.bash↵
```

海岸線，出ましたね？

オプションは

-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表を，色の名前は以下のgmtcolorsページをみましょう

https://docs.generic-mapping-tools.org/6.0/gmtcolors.html

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(l)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 のモジュールマニュアルで ***coast*** の項目を読んで，国境線を入れて見ましょう.

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　*grdcut*　*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．グリッドデータには，標準ではnetCDFと呼ばれる汎用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．
グリッドファイルは，自分でつくるとか，人からもらうとか，公開されているデータを取得するとか，買うとか，さまざまな場合が考えられます．とりあえず，GMTのパッケージの一環として使える公開地形データの一部を練習用に使いましょう．

まず，ターミナルで以下のようにタイプしてください．

```
gmt grdcut @earth_relief_02m -R125/145/25/45 -Gjapan_02m.grd ↵
```

grdcutは既存のグリッドデータの一部を切り出して，新しいグリッドデータファイルを作るモジュールです．引数として，元の既存グリッドファイル名を指定し，切り出す範囲を-Rオプションで，切り出した新しいファイル名を-Gオプションで指定します．ファイル名に@がついているのは，GMTのパッケージとして含まれているというサインで，02mは分解能が2分（緯度1分＝1nautical mile=1852m）という意味です．
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***モジュールを使って内容を知ることができます．切り出したファイルの内容を確認しましょう．

```
gmt grdinfo japan_02m.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）が表示されました．それでは，このグリッドファイルを使って地形図をかきましょう．

4.2. 等高線（等深線）を描く　　*grdcontour*　-C

グリッドデータと*grdcontour* モジュールを使って等高線（等深線）を描きます．前章でつくった*plot_basemap.bash* をコピーして新しいスクリプトファイルを作りましょう．

cp plot_basemap.bash plot_contour.bash ↵

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．

```
gmt begin contourmap1
        gmt basemap -R125/145/30/40 -JM12 -B -V
        gmt grdcontour japan_01m.grd -C200 -A2000
        gmt coast –Di -Ggray -Wthin,black
gmt end show
```

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．
bash plot_contour.bash ↵

オプションは
- -C　等値（深）線の間隔
- -A　等値（深）線に文字を入れる間隔

coast同様に，-Wオプションで線の太さや色も指定できますし，スムージングを掛けるなどのオプションもあります．

演習1: 等値線の間隔を500m，文字入れは5000mにしてみましょう．
演習2：オプション-Qは，ごく小さい閉曲線は描かない（図がすっきりする）指定です．モジュールマニュアルを見て，-Q10などを指定してみましょう．

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

次に ***grdimage*** モジュールを中心に，いくつかのコマンドを使って，色で値（この場合は標高/水深）を示す図を描きましょう．まず，前節のスクリプトをコピーして新しいファイルをつくります．

cp plot_contour.bash plot_colmap.bash

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．

```
gmt begin colormap1
        gmt basemap -R125/145/ 25/45 -JL135/35/30/40/12 -Bf1a10g5 -V
        gmt makecpt -Chaxby -T-8000/0/1000
        gmt grdimage japan_02m.grd
        gmt colorbar -DJBC -Bxa2000 -By+lm
#       gmt contour japan_01m.grd -C200 -A2000
        gmt coast –Di -Ggray -Wthin,black
gmt end show
```

まず，このスクリプトを実行して図を出し，各行を見ていきましょう．
bash plot_colormap.bash ↵

1行目で，出力ファイルの名前を変えます．
3行目の***makecpt*** がカラーパレットを作成するモジュールです．カラーパレットは，どの数字に対して何色をあてるかを並べて書いたアスキーファイルです． 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．
ここで使っている***makecpt***のオプションは

-C　標準となるカラーパレットの名前．どんなパレットがあるかは以下に．
https://docs.generic-mapping-tools.org/6.0/cookbook/cpts.html

-T　-Cで指定したカラーを，どの値に割り付けるか．-T最小値/最大値（/分割間隔）．

　このスクリプトでは，つくられたカラーパレットはファイルには記録されずに，４行目以下のモジュールで使われます．自分の作成したカラーパレットを記録したい，他のデータにも同じパレットを利用したい，という場合は，

　gmt makecpt -Chaxby -T-8000/0/1000　> mycolor_haxby.cpt

のように，ファイル名をつけて（.cptをつけるのが慣習）リダイレクトするといいです．

4行目の***grdimage***が彩色のモジュールで，グリッドファイルを読み込んで，前の行で作成したカラーパレットに対応する色を塗ります．以前につくったカラーパレットファイルがある場合は，

　-Cmycolor_haxby.cpt

といった形でファイル指定をすることができます．

5行目の***colorbar***は色の凡例を描くコマンド．オプションは

　-J　凡例の位置とサイズの指定．細かく指定もできるが，ここでは簡易指定でBC(Bottom Center)として，図の下の中央に凡例を描かせている．

　-B　凡例の目盛り間隔やラベルの指定．ここでは，x軸（水平の凡例なので）方向に2000ごとに文字を刻み，y軸側に単位である”m”をつけている（+lがplus labelの意味）

6行目の頭に #．　#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実行する行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ファイルの最初に記録しておくと便利ですね．

演習 1：カラーパレットを変えてみましょう．
演習 2： makecptに-Zコマンドや-Iコマンドをつけると何が起こるか試してみましょう．
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#を消してみましょう．
演習 4: GMTは行の順番に図を上書きしています．coastを含む7行目をgrdimageの行より前に移動させるとどうなりますか？

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，地形が3次元的に浮かび上がるように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，前節のスクリプトを開いて，いくつか書き加えましょう．赤字の部分が修正・追加したところです．

```
gmt begin colorshade1
        gmt basemap -R125/145/ 25/45 -JL135/35/30/40/12 -Bf1a10g5 -V
        gmt makecpt -Chaxby -T-8000/0/1000 -Z
        gmt grdgradient japan_02m.grd -A90 -Ne0.6 -Gjapan_02m_90.int
        gmt grdimage japan_02m.grd -Ijapan_02m_90.int
        gmt colorbar -DJBC -Bxa2000 -By+lm
#       gmt contour japan_01m.grd -C200 -A2000
        gmt coast –Di -Ggray -Wthin,black
gmt end show
```

まず，このスクリプトを実行して図を出し，各行を見ていきましょう．

bash plot_colormap.bash ↵

影のついた立体感のある図ができましたか？

1行目で，出力ファイルの名前を変えます．

4行目が，*grdgradient*のモジュールです．これは，入力グリッドデータに対して，勾配（この場合は地形の傾斜）計算するモジュールです．このモジュールでは，最大勾配の方向を出したり，x,y方向の勾配を出したりするオプションもありますが，ここではある方向から光をあてた際の陰影が，光の到来方向に相対する傾斜の大きさによることを使って，陰影用ファイルを作成することに使っています．オプションとして指定しているのは

-A　光をあてる方向．北を0として時計回りに度で測る．90なので東から光をあてた場合を想定．

-N　単にデータ勾配に比例して影をつけると値の分布が大きすぎて見にくい図になるので，計算した勾配を正規化します．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．この数字を小さくしていくとだんだん起伏の強調が弱くなります．

-G　出力する勾配（陰影）ファイル名．拡張子はなんでもよい．

-A オプション変化させると図の印象がかなり変わるだけでなく，読み取れる情報も変わってしまいます．–A90は東から光をあてた場合の陰影で，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．一方，東西の構造は目立たなくなり，見逃してしまう危険があります．陰影図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，光源の方向を変えてみることも重要です．また，主たる方向が2方向あるとか，全体としてあまり1方向を強調したくないが立体的に見せたい，という場合は，-Aオプションで2方向（-A90/160とか）を指定するとよいです．

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　*plot*

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．GMTの場合，デフォルトでは，1列目は緯度ではなく経度です（x座標だから）．

最初のスクリプト plot_basemap.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．

cp plot_basemap.bash plot_symbol.bash↵

スクリプトを開いて，いくつか書き加えましょう．赤字の部分が修正・追加したところです．

```
gmt begin symbol1
    gmt basemap -R125/145/30/45 -JM12 -B -V
    gmt plot sample_loc.dat -Sa0.3 -Gred -Wthinner,black -V
    gmt plot line_loc.dat -Wthick,blue -V
    gmt coast -Di -Ggray -Wthin,black
gmt end show
```

スクリプトを実行して，図を確認しましょう．

bash plot_symbol.bash ↵

日本海に赤い星印が，本州南岸に青い線がひけていますね．入力ファイルのsample_loc.datをcatコマンドで開いてみて，星印がファイルに記載された位置に描かれていることを確認しましょう．

plotコマンドは線やシンボルを描くモジュールです．オプションは

-S　シンボルの種類と大きさ．この場合は星(stAr)で直径が0.2cm．モジュールマニュアルを見て，どのようなシンボルが描けるか確認しましょう．このオプションがないと，4行目のようにデータ点をつなぐ線が描かれます．

-G　シンボルの塗りの色指定

-W 線やシンボルの枠線の太さと色指定

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-Wオプションをなくすとどうなりますか．

GMTのtextというモジュールでは，指定した位置に文字を書き込むこともできます．ただ，これは指定の方法がかなり面倒．多数の観測点の番号をリスト通りに間違いなく振るような場合はこのモジュールを使いますが，テキストがそれほど多くない場合は，基本の図だけGMTで作成して，他のお絵かきソフトやプレゼンソフトで書き込むほうが楽です．これまでのスクリプトでは，出力ファイルはbeginモジュールで指定した名前でpdfファイルがつくられていますが，他のソフトウェアに取り込んだり，文書ファイルに図を挿入する時用に，jpg, pngなどの画像ファイルでの出力指定もできます．例えば，以下のようにすると，.pdf, .jpgの2種類のファイルが出力されます．

```
gmt  begin  symbol1  pdf  jpg
```

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　*plot –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．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plotモジュールを使ってプロットしましょう．縦軸はフリーエア重力異常，横軸はレコード番号とします．練習用データフォルダにあるシェルスクリプトplot_timeseries_rec.bashを走らせまましょう．

bash plot_timeseries_rec.bash↵

表示される図を見ながら，シェルスクリプトを見ていきましょう．

```
#   plot timeseries data
#
#   extract record number & free-air anomaly using awk
awk '{print $15, $13}' sample_grav.fa > tmpfile
#
#   plot graph
gmt begin fagrav_timeseries
        gmt basemap -R0/6000/-60/60 -JX18/5 -B -Bx+lrec_number -V
        gmt plot tmpfile -Sp -Gblue   -V
gmt end show
```

最初に，awkを使って入力ファイルsample_grav.faから，レコード番号（15列目）と水深（13列目）を抜き出して，tmpfileというファイルに入れています．これが，作図をする時の入力ファイルとなります．
作図は，前章で試した**basemap**と**plot**モジュールです．basemapで図法を指定していますが，今回はいわゆる地図図法でなく，普通の平面XY座標です．-JX18/5というのは，平面座標で，できあがり寸法がX18cm,Y5cmにしますよ，ということです．最初の-Bオプションで何も指定していないの，軸の目盛りはGMTが適当につけてくれています．2番目の-Bオプションでは，x軸方向のラベルを追加しています．plotモジュールでは，-Sp -GBlueで青いポイント（単に点を打つイメージ）を指定しています．
データは図の範囲にちゃんと入っていますか？ここでは，-Rの範囲指定は，x軸（レコード番号）が0~6000, y軸（フリーエア重力以上）が-50から50としています．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．

ここでは，中間ファイルとしてtmpfileが作成されます．awkがちゃんと働いているかを確認する時には，catコマンドでtmpfileをチェックするといいですね．

```
more tmpfile↵
```

このやり方だと中間ファイルがディスクを圧迫して嫌だ，という場合は
・スクリプトの最終行に rm tmpfileを加えて削除してしまう
・unixのパイプ（｜）を使う
といった方法もあります．パイプというのは，あるコマンドの出力を，続いて実行するコマンドの入力に流す方法で，例えばスクリプトのawkの行は削除して，gmt plotの行を
awk '{print $15, $13}' sample_grav.fa | gmt plot -Sp -Gblue　-V
とすることができます．これはスマート．

演習 1: 水深をプロットしましょう. 色も変えてみよう.
演習 2: ｙ軸にラベルdepthをつけよう.
演習 3: 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探しましょう. awkで入力を変える方法と, GMTのオプションで上下反転する方法があります.

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, 欠測時間などがわかりやすいですね. 60進法が入ってくるのでちょいと面倒ですが, 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１）awkを使ってGMTの標準時刻フォーマット（yyyy-mm-ddThh:mi:ss）とフリーエア重力異常の組み合わせファイルをつくる, 2）plotするになります. 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);
}
```

この awk は出力の書式も指定しています. 書式指定は C 言語に近く, 最初の%4dで 4 桁整数で$1 を書く…といったことになります.

plot_timeseries_hour.bash

```
#   plot timeseries data   x-axis=time
#
#   extract record number & depth using awk
awk -f fa2ts.awk sample_grav.fa > tmpfile
#
#   plot graph
gmt begin fagrav_timeseries_hour
        gmt basemap -R2016-01-23T00:00:00/2016-01-24T00:00:00/-100/100 -
JX18/-5 -Bpxf1ha4Hg4h -Bsxa1D -Byf10a50g50+lfa -V
      gmt plot tmpfile -Sp -Gblue   -V
gmt end show
```

このスクリプトを走らせてみましょう.

bash plot_timeseries_hour.bash↵

横軸が時間になった図ができたと思います. ここで-Bpx -Bsxとなっているのは, 軸を二重に（ラベルが二重になっていますね？）する場合の, primary axisとsecondary axisの意味です. ここでは, まず, 1時間刻みを入れ, 4時間ごとに文字と罫線を入れます. そして2重にして, そこには日付の文字だけを入れる, という指示になっています.

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の4.3.5を見ること. https://docs.generic-mapping-tools.org/6.0/cookbook/options.html#cartesian-axis2

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．前章で作った時系列データプロットのスクリプトplot_timeseries_rec.bashに追加してウィンドウ長50のBoxcar フィルタ（移動平均）をかけてみます．
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ます．このスクリプトを開いて，下の赤字の部分を修正・追加してください．

```
cp plot_timeseries_rec.bash plot_timeseries_filt.bash
```

```
#   plot timeseries data and filtered data
#
#   extract record number & free-air anomaly using awk then filtered
awk '{print $15, $13}' sample_grav.fa > tmpfile
gmt filter1d tmpfile -Fb50 -N0 -V > grav_filt.fa
#
#   plot graph
gmt begin fa_filt_timeseries
    gmt basemap -R0/6000/-50/80 -JX18/5 -B -Bx+lrec_number -V
    gmt plot tmpfile -Sp -Gblue   -V
    gmt plot grav_filt.fa -Wred   -V
gmt end show
```

修正したら走らせてみましょう.

bash plot_timeseries_filt.bash↵

まず，前回同様にレコード番号とフリーエア異常を抜き出してtmpfileに格納し，そのtmpfileを読み込んでfilter1dで1次元フィルターをかけています.

ここで使っている*filter1d*のオプションは

-F　フィルターの種類とウィンドウ長．この場合はウィンドウ長50のboxcarフィルタをかけて移動平均をとって平滑化している.

-N　何列目が独立変数（時間にあたるところ）か．0が一番左の列を表す.

図化の段階で，元のデータを青のポイントで，フィルター後のデータは赤の連続線で描いています.

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でfilter1dモジュールを調べましょう.

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，磁気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_02m.grd** （地形, 2章で切り出した）と **japan_02m_EMAG.grd** （磁気異常）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grdcontour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．

スクリプト例はこんな感じ．

```
gmt begin colormap_mag+depcont
    gmt basemap -R125/145/25/45 -JM12 -Bf1a5g5 -V
    gmt makecpt -Cpolar -T-500/500/100 -Z
    gmt grdimage japan_02m_EMAG.grd
    gmt grdcontour   japan_02m.grd -C1000 -Wthinner,black -Q10
    gmt grdcontour   japan_02m_EMAG.grd -C500   -Wthinner,gray -Q10
    gmt coast -Di -Ggray -Wthin,black
gmt end show
```

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, 上述のような地磁気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.

スクリプト例はこんな感じ.

```
gmt begin colorshade_mag+depcont
    gmt basemap -R125/145/25/45 -JM12 -B -V
    gmt makecpt -Cpolar -T-500/500/100 -Z
    gmt grdgradient japan_02m.grd -A90 -Ne0.6 -Gjapan_02m_90.int
    gmt grdimage japan_02m_EMAG.grd -Ijapan_02m_90.int
    gmt coast -Dl -Ggray -Wthin,balck
gmt end show
```

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，地形や磁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*plot* 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 tut_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，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．データの範囲を*gmtinfo*を使って確認してみましょう．

```
gmt gmtinfo @tut_ship.xyz↵
```

以下のような表示が出ると思います（@はGMTに含まれているファイル）．

```
.gmt/cache/tut_ship.xyz: N = 82970 <245/254.705>
<20/29.99131>   <-7708/-9>
```

ここから，82970個のデータがあって，経度が245~254°位，緯度が20~30°位の範囲で，水深は8000mくらいまである，ということがわかります．

まず，これまでの復習として，データのある位置を点で示した図をつくってみましょう．plot_symbol.bashを書き換えるとよいですね．

スクリプト例はこんな感じ

```
gmt begin ship_datappoint
    gmt basemap -R245/255/20/30 -JM12 -V
    gmt plot @tut_ship.xyz -Sp -Gred -V
    gmt coast -Di -Ggray -Wthin,blacks -Bf1a10g5
gmt end show
```

船の航跡上にだけデータポイントがあることがわかります．密にデータがあるところもあり，緯度 1 度近くデータのないところもあります．中間をとって，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ることにしましょう．

9.2 グリッドデータを作成する *nearneighbor*

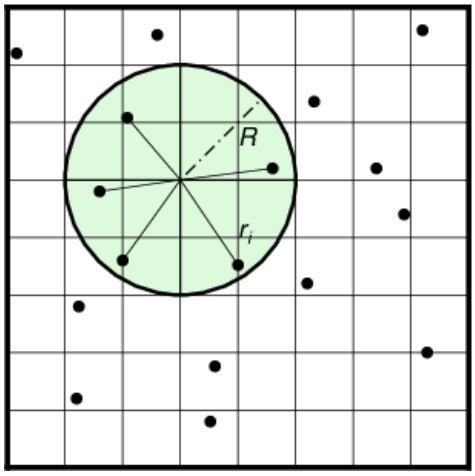
それでは，前節で表示した**ship.xyz**の離散データから，格子サイズが5’ x 5’ (角度の分) のグリッドデータをつくります．練習用フォルダにあるmkgrid+plot.bash を開いて，内容を確認します．

```
#     make grid file from xyz data, using near neighbor
gmt nearneighbor @tut_ship.xyz -R245/255/20/30 -I5m -S40k -Gtut_ship_5m.grd -V
#
#     plot grid data
gmt begin color_california
      gmt basemap -R245/255/20/30 -JM12
      gmt makecpt -Ctopo -T-8000/0/1000 -Z
      gmt grdgradient tut_ship_5m.grd -A90 -Ne0.6 -Gtut_ship_5m.int
      gmt grdimage tut_ship_5m.grd -Itut_ship_5m.int
      gmt colorbar -DJBC -Bxf1000a2000 -By+lm
      gmt grdcontour tut_ship_5m.grd -C200 -A1000 -Q10
      gmt coast -Di -Gwhite -Wthin,balck -Bf1a10g5
gmt end show
```

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．オプションは

-I　格子の間隔，mがついていると分(メートルではない).

-S　探す(Search)の範囲．下の図のR．kがついていると単位はkm



(from GMT cookbook)

それではスクリプトを走らせて，グリッドデータと図を作成してみましょう．

Bash mkgrid+plot.bash ↵

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，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．