第3回:海洋数値モデルの構築

高速計算の方法:なぜ並列計算なのか

並列海洋数値モデル　sbPOM

どんな問題を解きたいか: 2011-2012年瀬戸内海シミュレーション

瀬戸内海3km格子モデルを構築する

計算入出力データの取り扱い: NetCDF

入力データの作成:地形、気候値、境界条件

モデルを走らせる

後処理、計算結果の可視化

数値計算のためのコンピュータ言語

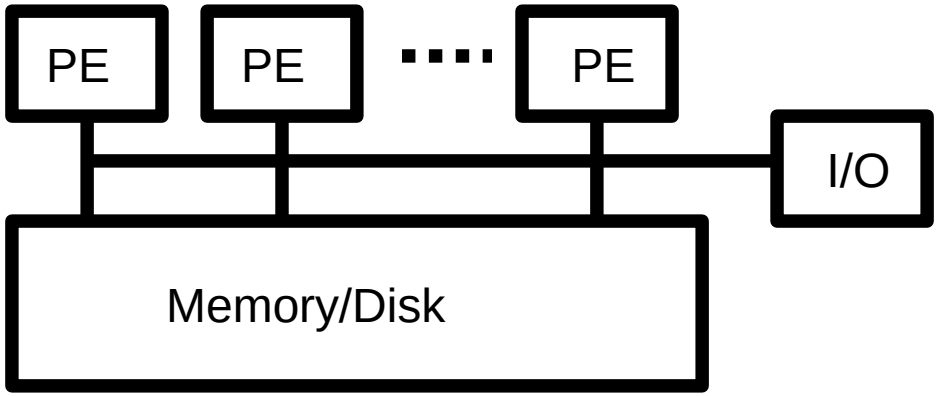
Fortran 77, Fortran90/95

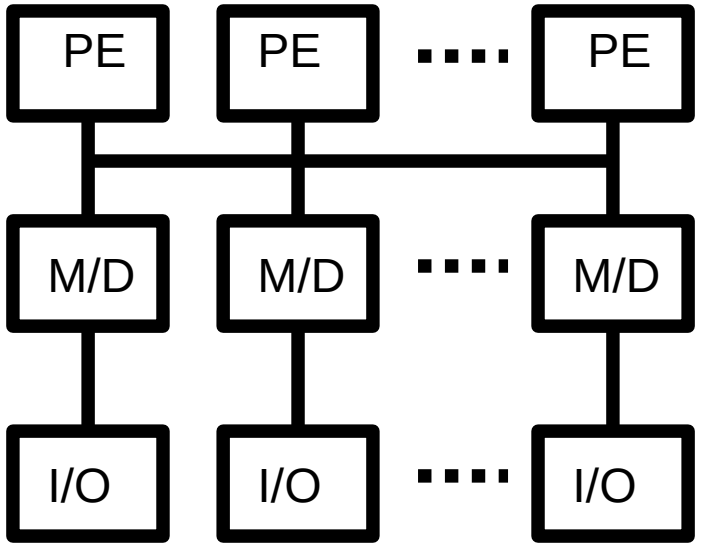
sbPOMはFortran77によって記述

```
do k=2,kbm1
      do j=2,jmm1
        do i=2,imm1
          zflux(i,j,k)=.5e0*(f(i,j,k-1)+f(i,j,k))*w(i,j,k)*art(i,j)
        end do
      end do
end do
```

DOループの塊

高速計算の2つの方法: ベクトルかクラスターか





ベクトル計算
　DOループの並列化
　自動並列化

メモリーは共有

並列計算
　計算プログラムそのものの並列化

メモリーは別々（分散）

なぜ並列計算なのか

<1個で計算>

講師のノートPC
(Intel core i7 2.13Ghz, 2010年)
2シミュレーション年: 7.6日

JAMSTECクラスター計算機　SGI ICE
(Intel Xeon E5-2670 2.6 GHz, 2012年)
2シミュレーション年: 1.9日

<8個で計算>

JAMSTECクラスター計算機　SGI ICE　16-core x 432 nodes
(Intel Xeon E5-2670 2.6 GHz, 2012年)
2シミュレーション年: 8.5時間

最新型ワークステーション　16-core x 2 nodes
(Intel Xeon E5-2667 v2 3.3 GHz, 2013年)
2シミュレーション年: 5.7時間

クラスター計算機のもう一つの利点

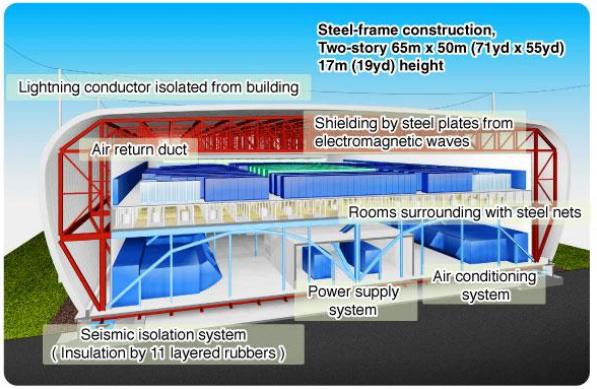
<JAMSTEC大型計算機システム>

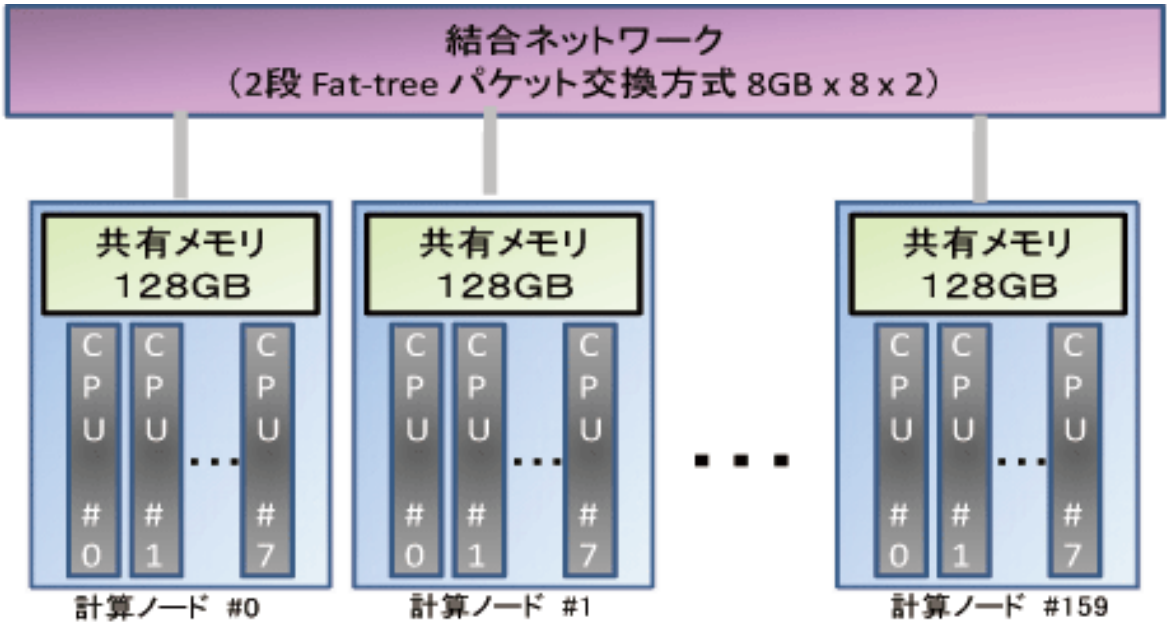
ベクトル型: 32CPU
クラスター: 6912core, 1024core
→ たくさんの感度実験をいちどに実行できる

愛媛大学　郭新宇研究室　SGIクラスター計算機 116-core

JAMSTEC地球シミュレータ

ベクトルなのに大規模並列可能！　実効性能が早い！
そのかわり高価で、電気代もかかる
8-core x 160 nodes (1280-core)

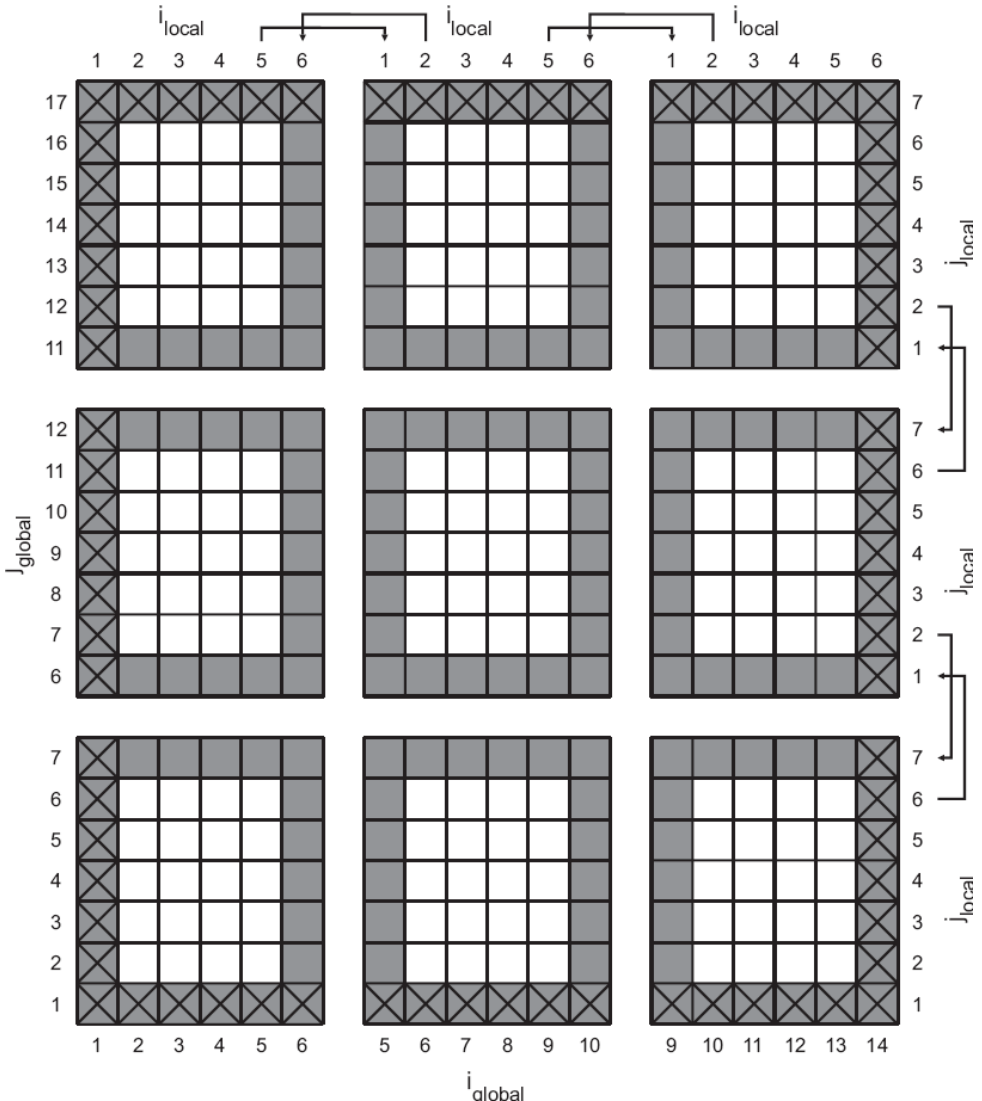




並列海洋モデルsbPOM

Jordi and Wang (2012) ‘sbPOM: A parallel implementation of Princenton Ocean Model’, Environmental Modelling & Software

$$n_{proc} = \frac{im_{global} - 2}{im_{local} - 2} \times \frac{jm_{global} - 2}{jm_{local} - 2}$$



並列プログラミング:MPI

```
do j=2,jmm1
      do i=2,imm1
        elf(i,j)=elb(i,j)
   $             +dte2*(-(fluxua(i+1,j)-fluxua(i,j)
   $                    +fluxva(i,j+1)-fluxva(i,j))/art(i,j)
   $                    -vfluxf(i,j))
      end do
 end do

 call bcond(1)

 call exchange2d_mpi(elf,im,jm)
```

Arrays of model variables are defined on local range (im_local, jm_local)

imm1 (im-1) and jmm1(jm-1) do not indicate global range but local range calculated by each processor .

Communications of model variables along the edge of local regions are required.

並列化効率

$$Eff_{n/n_0} = \frac{n_0 \cdot ElapseTime_{n_0}}{n \cdot ElapseTime_n}$$

If completely effectively parallelized, we expect reduction of elapse time depending on increase of number of processors. But communication of variables along edges of localized regions need some time. Then elapsed time is not much reduced with ineffective parallelization configuration.

If completely parallelized, Eff = 1. But in the ineffective cases, Eff < 1.

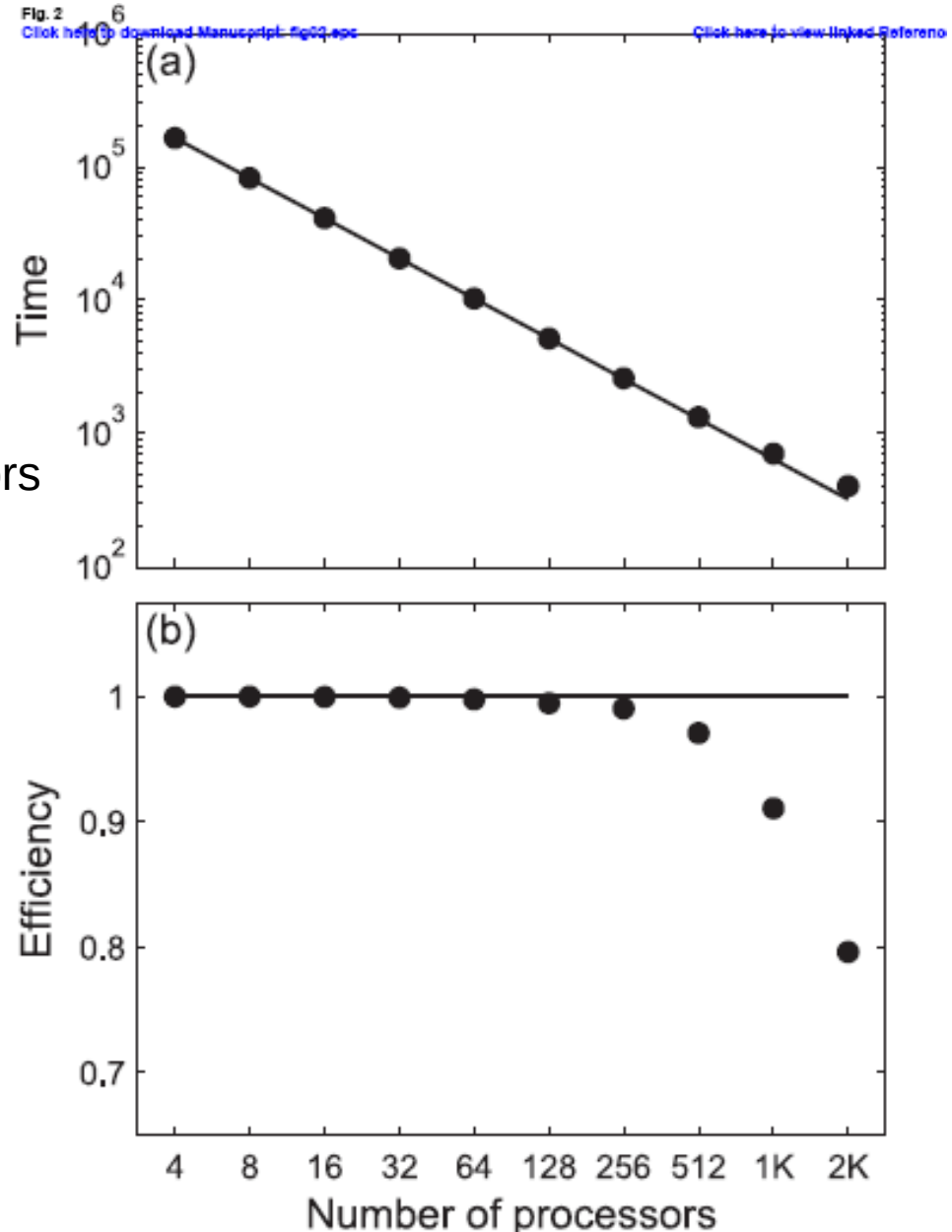
sbPOMの並列化効率

1026x770x31 grids
IBM Blue Gene / L
at Stony Brook University

sbPOM is able to maintain
a very good performance
for large number of processors

The efficiency relative to performance on four processors *is* very high even for 2K processors (almost 0.8).

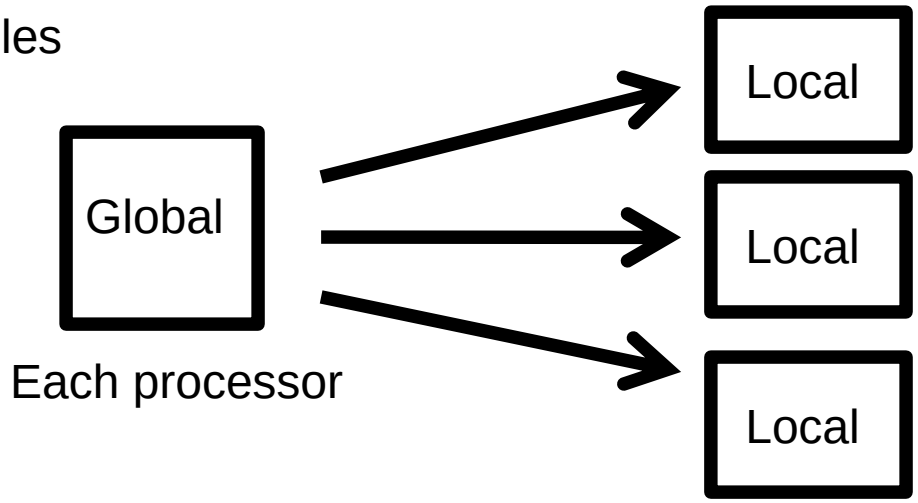
(Jordi and Wang, 2010)



並列モデルのデータ入出力

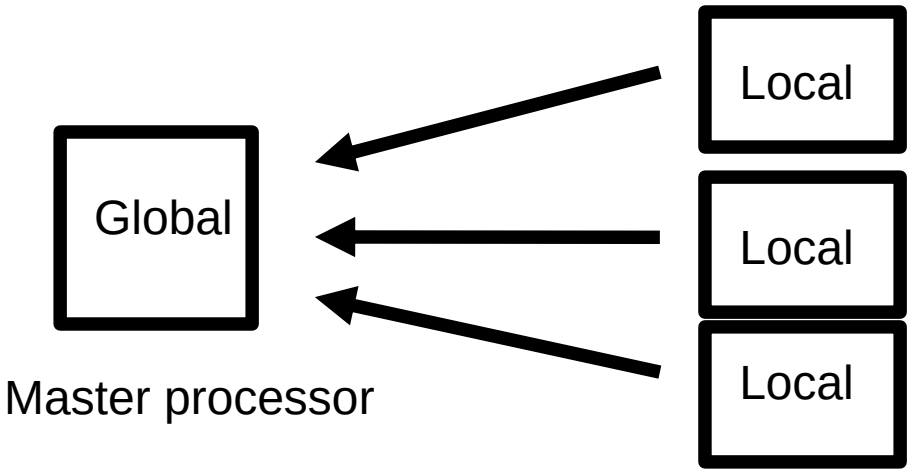
Initial condition of temperature T_global(1:i_global,1:j_global,1:kb) should be scattered to those of parallel processors: T(1:i_local,1:j_local,1:kb)

Reading files



Calculated temperature T_local(1:i_local,1:j_local,1:kb) should be merged into the global variable: T_global(1:i_global,1:j_global,1:kb)

Writing files



sbPOMにおけるデータ入出力

‘Parallel’ NetCDF is used for parallel I/O in the original version of sbPOM.

But our Earth Simulator-2 do not allow use of the parallel NetCDF owing to some technical problems.

Then I have modified the SBPOM to use usual NetCDF.

```
real glb3d(i_global,j_global,kb), t(i_local,j_local,kb)
```

Reading variables

```
status=nf_get_var_real(ncid,t_varid,glb3d)
call handle_error_netcdf('nf_get_var_real‘, status,nf_noerr)
call scatter3d(t,glb3d)
```

Writing variables

```
call merge3d(glb3d,t)
if(my_task.eq.master_task) then
  status=nf_put_var_real(ncid,t_varid,glb3d)
  call handle_error_netcdf('nf_put_var_real: t',status,nf_noerr)
end if
```

sbPOMのデータ入出力

‘Parallel’ NetCDF is used for parallel I/O in the original version of sbPOM.

But our Earth Simulator-2 do not allow use of the parallel NetCDF owing to some technical problems.

Then I have modified the SBPOM to use usual NetCDF.

```
real glb3d(i_global,j_global,kb), t(i_local,j_local,kb)
```

Reading variables

```
status=nf_get_var_real(ncid,t_varid,glb3d)
call handle_error_netcdf('nf_get_var_real‘, status,nf_noerr)
call scatter3d(t,glb3d)
```

Writing variables

```
call merge3d(glb3d,t)
if(my_task.eq.master_task) then
  status=nf_put_var_real(ncid,t_varid,glb3d)
  call handle_error_netcdf('nf_put_var_real: t',status,nf_noerr)
end if
```

sbPOMの中身

src/

pom.h: common block

pom.nml: name list

pom/pom.f: main program
- /initilize_relax.f: initialization
- /advance_relax.f: time integration of the model
- /solver.f: dynamical core of Princeton Ocean Model
- /fluxlib_rh_rt.f: bulk formulae for surface flux calculation
- /timelib.f: calendar functions for hindcast simulations
- /parallel_mpi.f: parallelization
- /io_netcdf.F, meger_scatter.f: parallel I/O using NetCDF

sbPOM メインプログラム: pom.f

```
! initialize model
      call initialize

! main loop
      do iint=1,iend
        call advance ! advance model
        if(error_status.ne.0) then
          if(my_task.eq.master_task) write(*,‘(/a)‘)   'POM terminated
with error'
          call finalize_mpi
          stop
        end if
      end do

! finalize mpi
      call finalize_mpi
```

sbPOMの並列数設定 ：pom.h

parameter(
$ im_global=164 ,
$ jm_global=110 ,
$ kb=21 ,
$ im_local=83 ,
$ jm_local=29 ,
$ n_proc=8)

$$n_{proc} = \frac{im_{global} - 2}{im_{local} - 2} \times \frac{jm_{global} - 2}{jm_{local} - 2}$$

$ pwd
$ /home/miyazawa/seto2/src/
$ make clean
$ make
$ ls –l pom.seto2.exe

$ mpirun –np 8 pom.seto2.exe

どんな問題を解きたいか

2011年　瀬戸内海　異常潮位

2011-2012年の2年間をシミュレーションし、2011年と2012年の結果を比較する

モデル設計
計算量と解像度の調整

3km格子：1/36度格子

潮汐計算には粗すぎるが外洋の影響を調べるには十分？

瀬戸内海3km格子モデルを構築する

Unix型のオペレーティングシステムを前提
クラスター計算機: 代表的なUnixであるLinuxが標準
Macintosh: Mac OS XはUnixであり実行可能（未確認）
Windows: 非Unix → Vmplayer で仮想マシンを構築しLinux導入
(Cygwin では実行失敗)

瀬戸内海3km格子モデルの作成: コードネーム seto2
標準ケース test1

ディレクトリ構造

seto2/ - prep/	前処理	入力データの作成
- in/	入力データ	入力データを格納
- src/	計算プログラム	sbPOM本体
- run_test1	計算結果	出力データを格納
- post_test1	後処理	可視化データ作成

必要なソフトウェア

フォートラン言語:
  Intel Fortran 有料だが速い
  Gfortran 無料

NetCDF: 無料　http://www.unidata.ucar.edu/software/netcdf/

MPI:
 クラスター計算機にはたいてい実装済
 OpenMPI 無料

講師のノートPC(Windows)では、
  Vmplayer(仮想マシン), CentOS(Linux), Gfortran
  NetCDF, OpenMPI　の組み合わせ

入力データの作成

地形データ作成 (バイナリーデータ)

水平鉛直格子データ作成 (NetCDF)

気候値水温塩分データ作成 (NetCDF)

側面境界条件データ作成 (NetCDF)

初期値データ作成 (NetCDF)

外洋同化データ作成 (Net CDF)

海表面気象データ作成 (NetCDF)

バイナリーデータの取り扱い

アスキーデータ （テキストデータ）:
「眼で見て内容を確かめることができるが容量が大きく読み書きが遅い」

バイナリーデータ:
「簡単に眼で見ることができないが容量が小さく読み書きが早い」

しかし、両者の問題は、データが何を「意味」しているかわからないところである。

NetCDF: 自己記述型のバイナリーファイル形式

NetCDFの中身を見る

ヘッダーファイル

$ ncdump –h seto2.grid.nc

```
netcdf seto2.grid {
dimensions:
        x = 164 ;
        y = 110 ;
        z = 21 ;
variables:
        double z(z) ;
                z:long_name = "sigma of cell face" ;
                z:units = "sigma_level" ;
                z:standard_name = "ocean_sigma_coordinate" ;
                z:formula_terms = "sigma: z eta: elb depth: h" ;
```

σレベル

NetCDFの中身を見る

各変数
$ ncdump –v z seto2.grid.nc

data:
z = 0, -0.0078125, -0.015625, -0.03125, -0.0625, -0.125, -0.1875, -0.25, -0.3125, -0.375, -0.4375, -0.5, -0.5625, -0.625, -0.6875, -0.75, -0.8125, -0.875, -0.9375, -0.96875, -1 ;
}

σレベル

NetCDFの作成法

ファイルの内容を記述するヘッダーファイルを編集する
- 格子：seto2.grid.hdr
- 水温塩分気候値：seto2.tsclim.hdr
- 側面境界条件：seto2.lbc.hdr
- 初期条件：seto2.ic.hdr
- 外洋同化データ：seto2.tsdata.hdr
- 海面気象データ：seto2.atm.hdr

NetCDFコマンドで、NetCDFデータ本体を作成する

```
$ ncgen –o seto2.grid.nc seto2.grid.hdr
```

地形データの作成

もっとも重要な工程

学習用として、3km格子地形データを用意

地形調整済

陸岸地形の設定済

海底地形の平滑化済

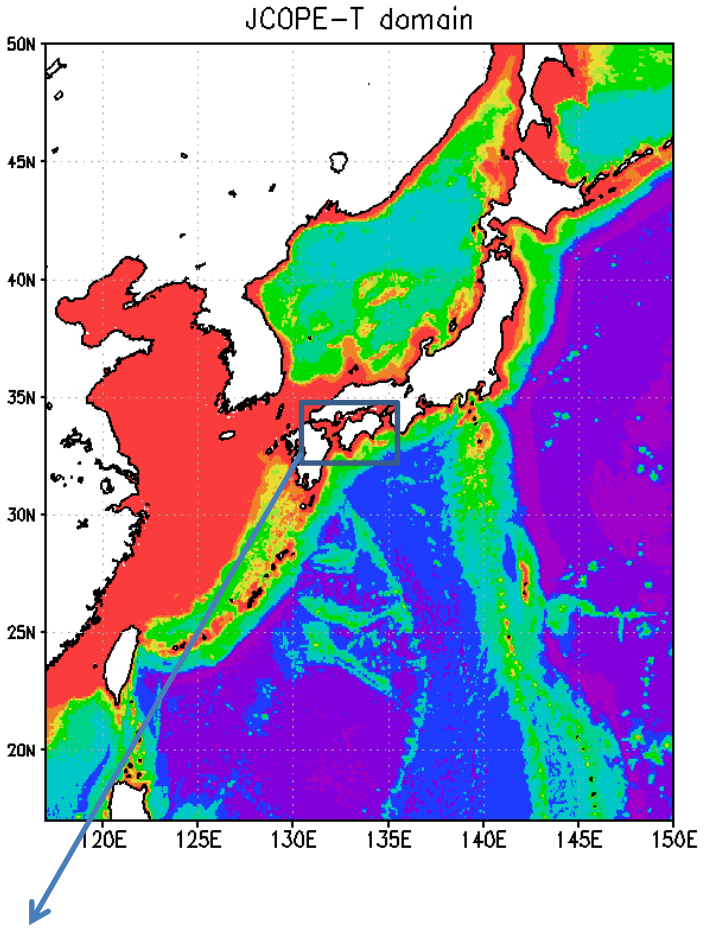
全領域の計算は重すぎる

バイナリーデータ

SETO2_DEP.dat

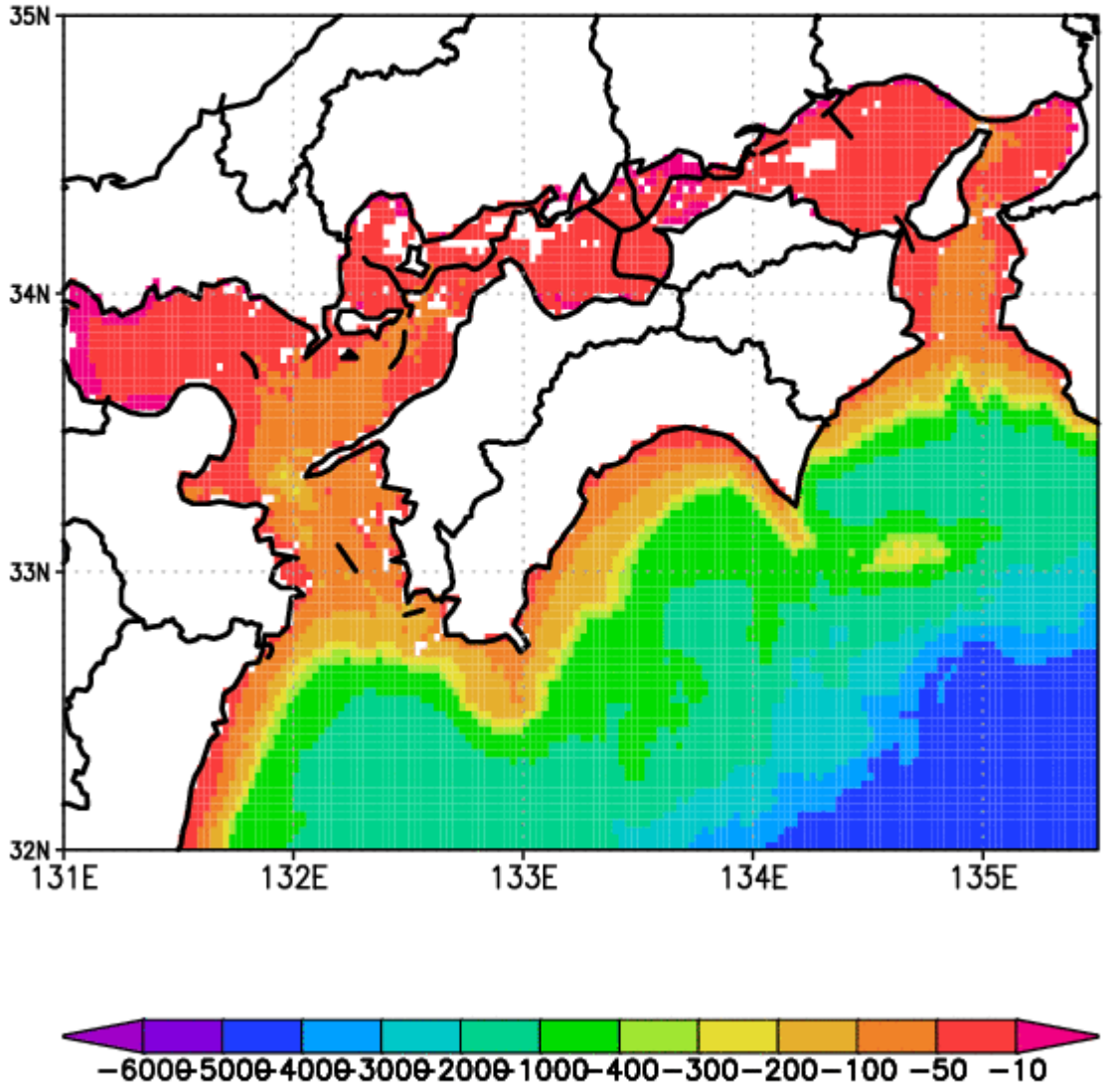
read(10) h(1:164,1:110), iflg (=123456)

1190 x 1190



今回切り出す領域： 164 x 110

seto2 計算領域と水深

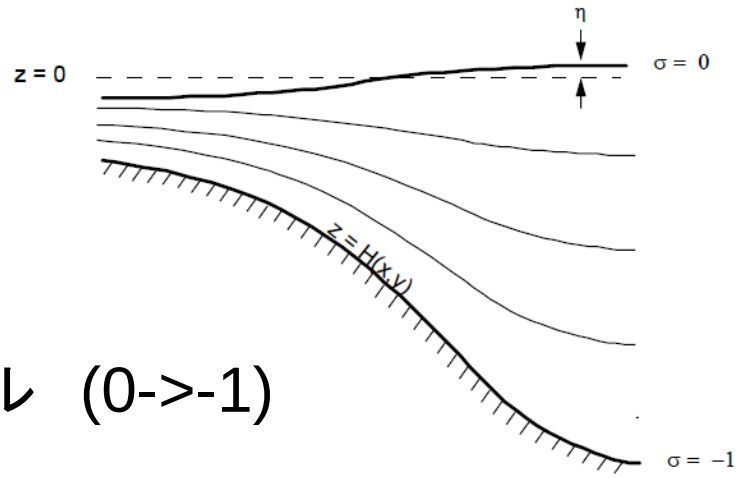


水平鉛直格子作成

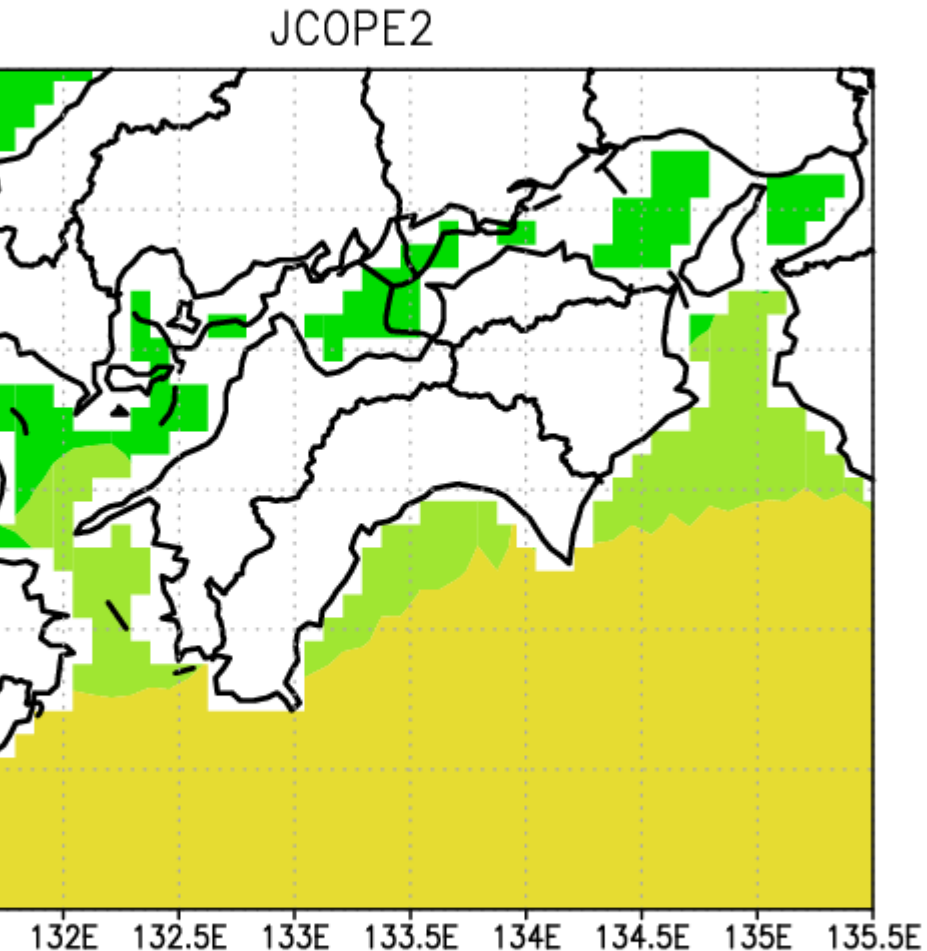
配列サイズ　im x jmx kb　(164 x 110 x21)

球面座標系 (経度、緯度)
水平格子
東西方向間隔:
　　dx(i,j) = R cos(latitude) dlon(i,j)　R: 地球半径
南北方向間隔:
　　dy(i,j) = R dlat(I,j)

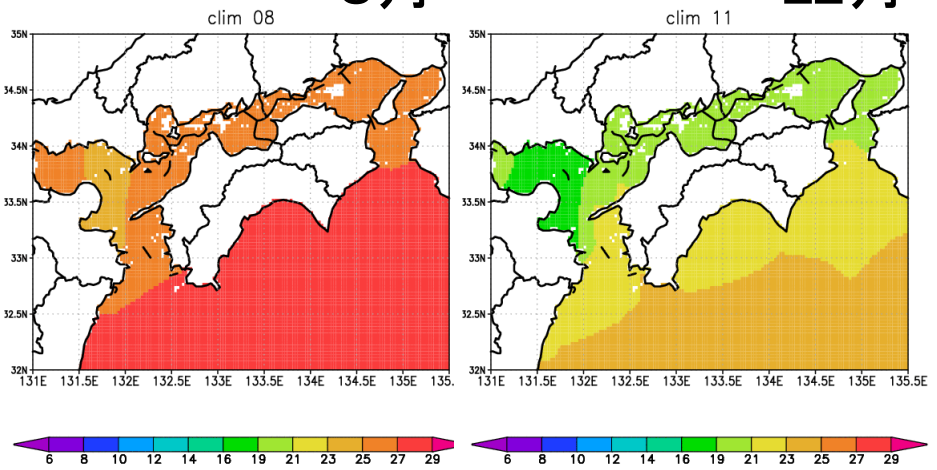
鉛直格子　格子間隔:
　　層厚　H(I,j) x dz(k)　H(i,j): 最大水深
　　水深　H(I,j) x z(k)　z(k): シグマレベル　(0->-1)



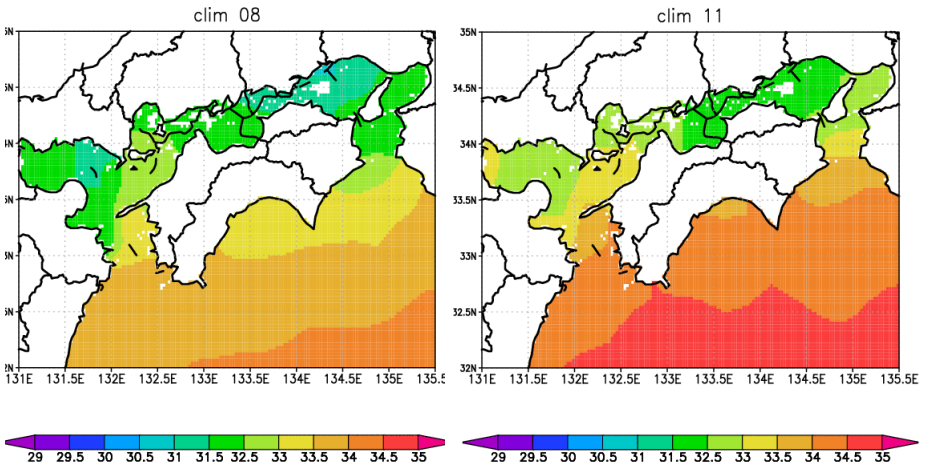
気候値水温・塩分データ作成



8月 11月

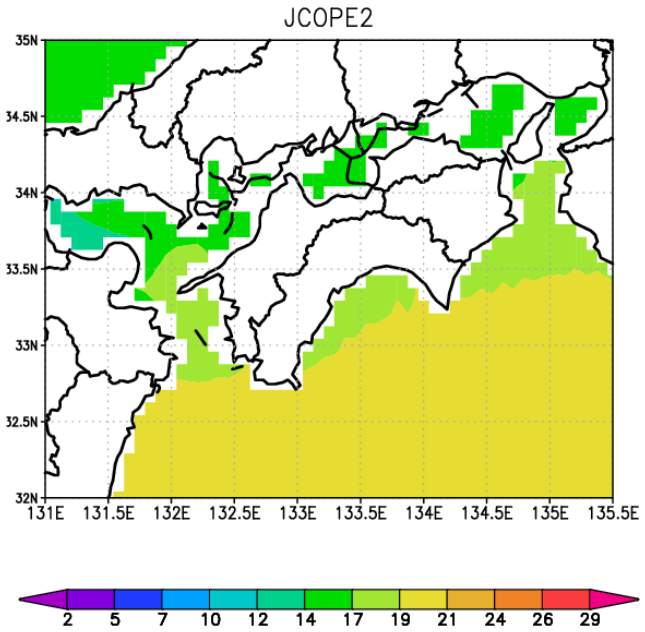


8月 11月

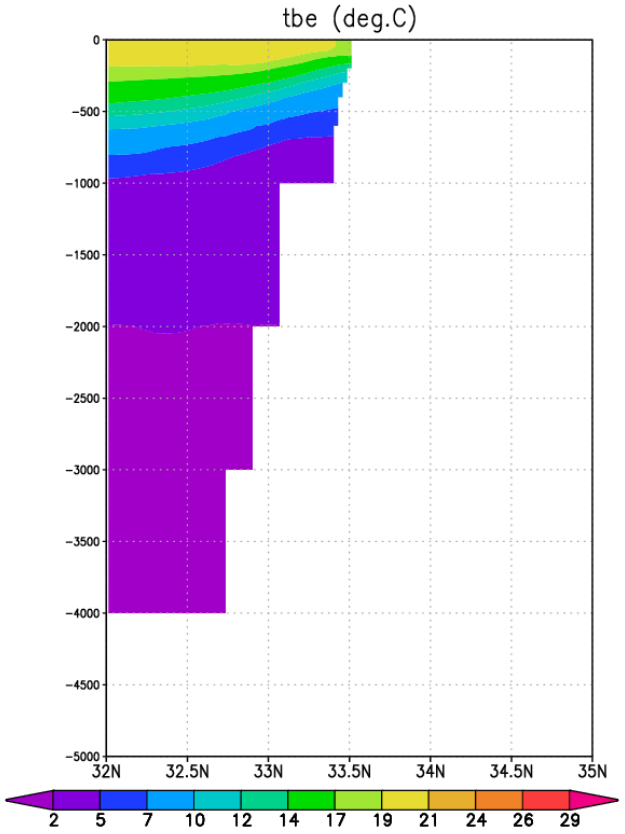


側面境界条件作成

2011/01/01 1/12度再解析データ

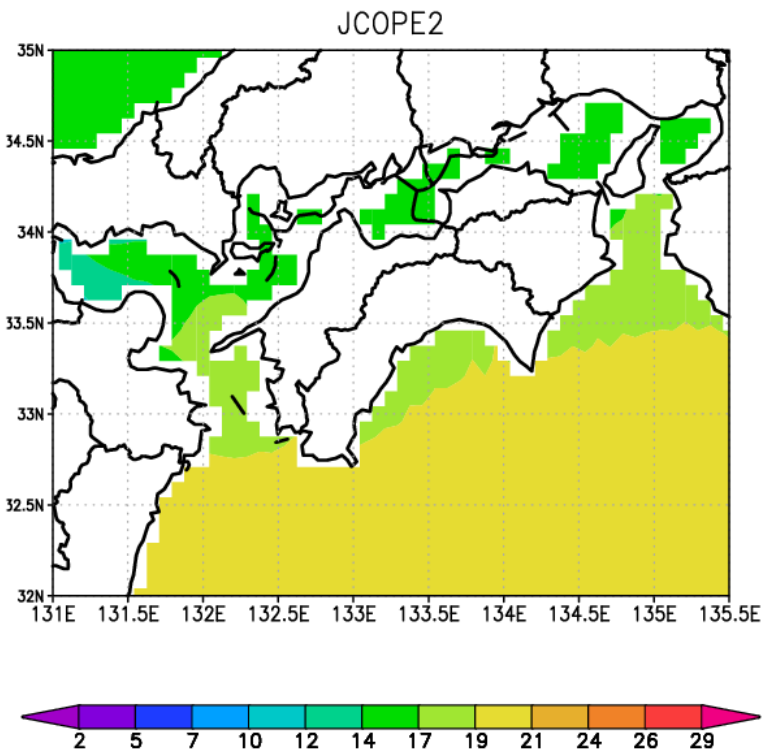


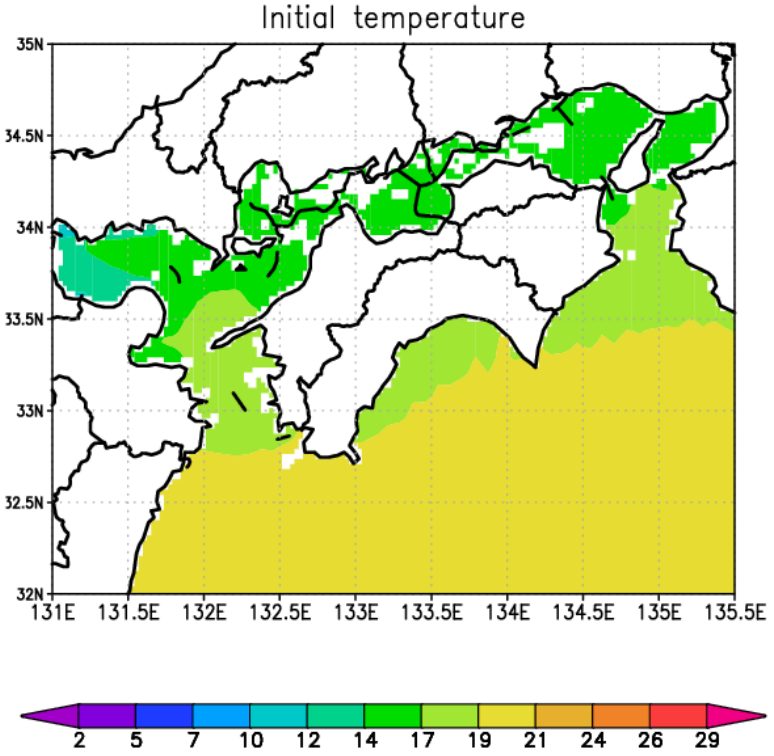
東側
ube(j,k) j=1,jm, k =1,kb
vbe(j,k)
tbe(j,k)
sbe(j,k)



南側
ubs(i,k) i=1,im, k =1,kb
vbs(i,k)
tbs(i,k)
sbs(i,k)

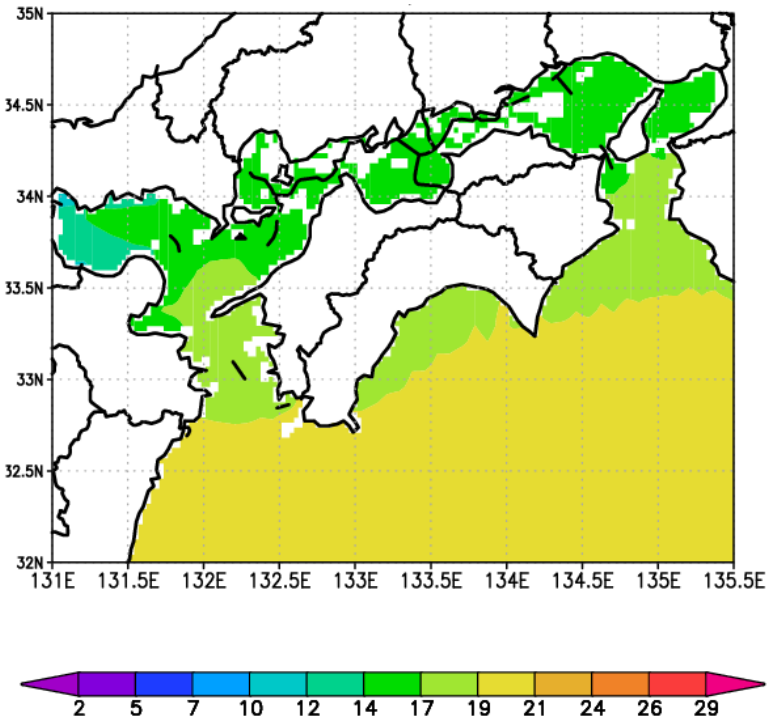
初期値データ作成


JCOPE2
35N
34.5N
34N
33.5N
33N
32.5N
32N
131E
131.5E
132E
132.5E
133E
133.5E
134E
134.5E
135E
135.5E
2
5
7
10
12
14
17
19
21
24
26
29


Initial temperature
35N
34.5N
34N
33.5N
33N
32.5N
32N
131E
131.5E
132E
132.5E
133E
133.5E
134E
134.5E
135E
135.5E
2
5
7
10
12
14
17
19
21
24
26
29

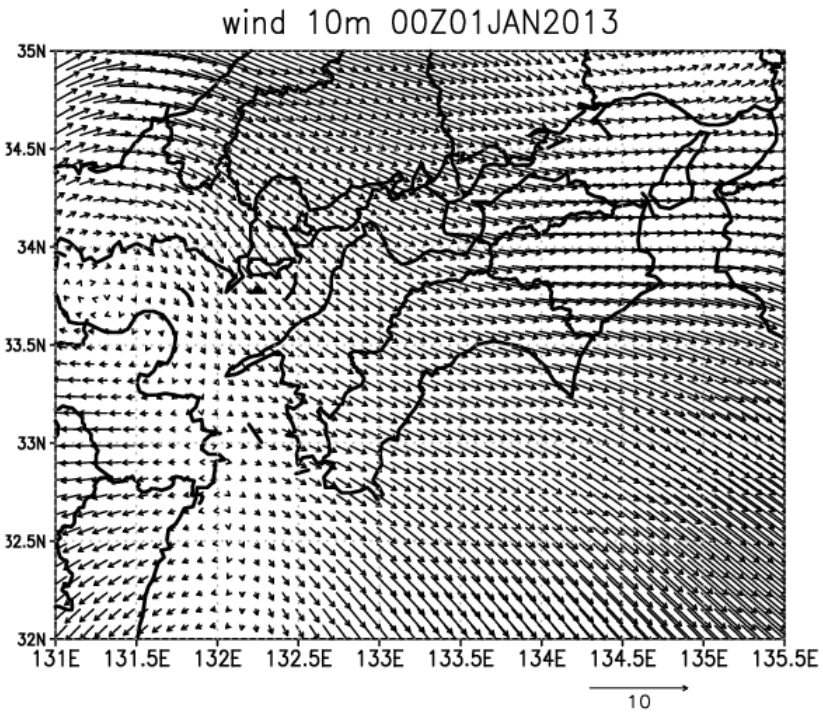
外洋同化データ作成

水温･塩分格子データ　日平均

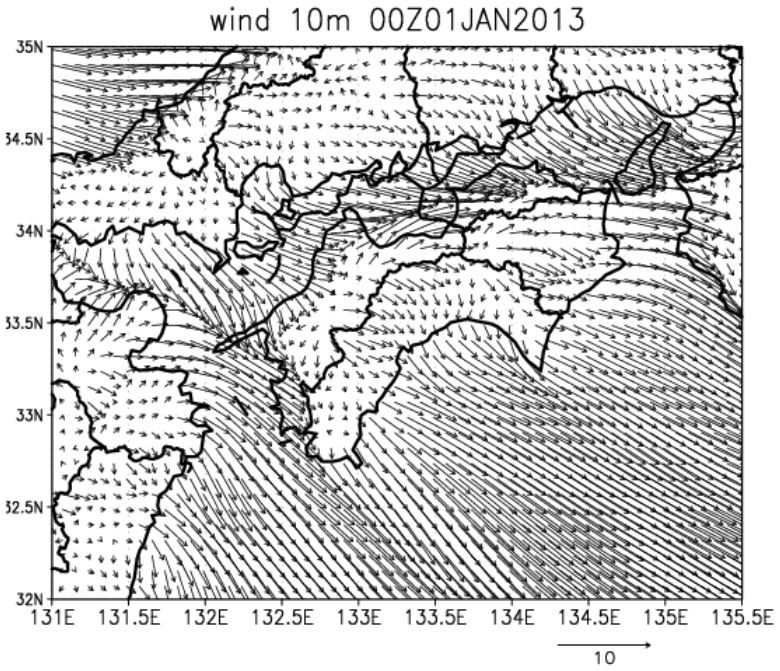


海面気象データ

NCEP GFS



JMA MSM



瀬戸内海3km格子モデルを走らせる

seto2/ - src/　makefile　ソースコードのコンパイル手順を記述
　　pom/　　sbPOMのソースコード
　　pom.h　　sbPOMの変数を記述　(コモンブロック)
　　sbPOMの並列度を記述
　　run.seto2.sh 実行シェルスクリプト
　　pom.seto2.exe コンパイル済実行可能ファイル

コマンド実行
seto2/src/ に移動
> make (pom.seto2.exe)
> run.seto2.sh >& log &

後処理

計算結果はNetCDFファイルとして出力される

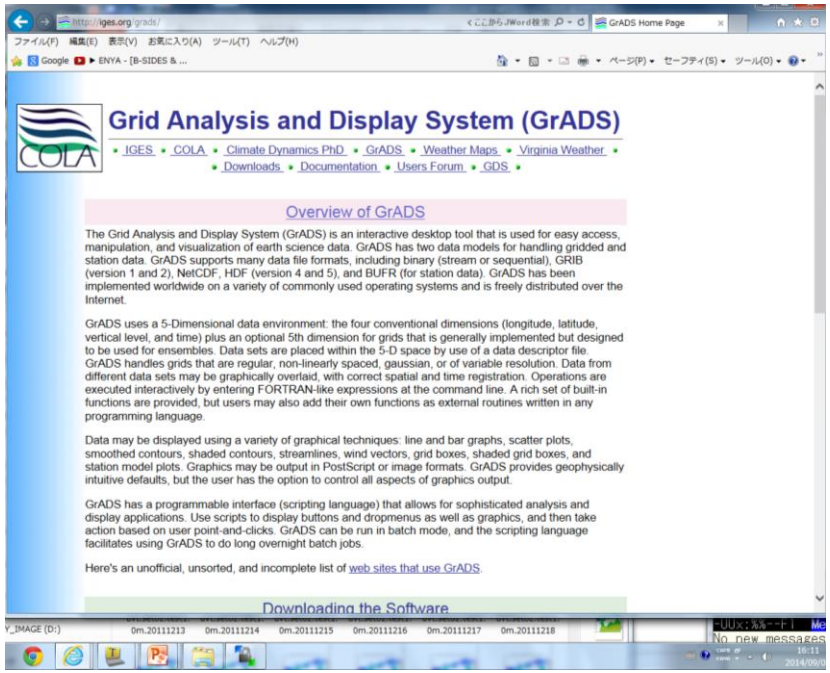
seto2/ - run_test1/ - out/
　1日毎　2年分　seto2.00001.nc
　　…
　　seto2.00730.nc

seto2/ - post_test1/ - makefile
　　- src/　　後処理ソフトウェア

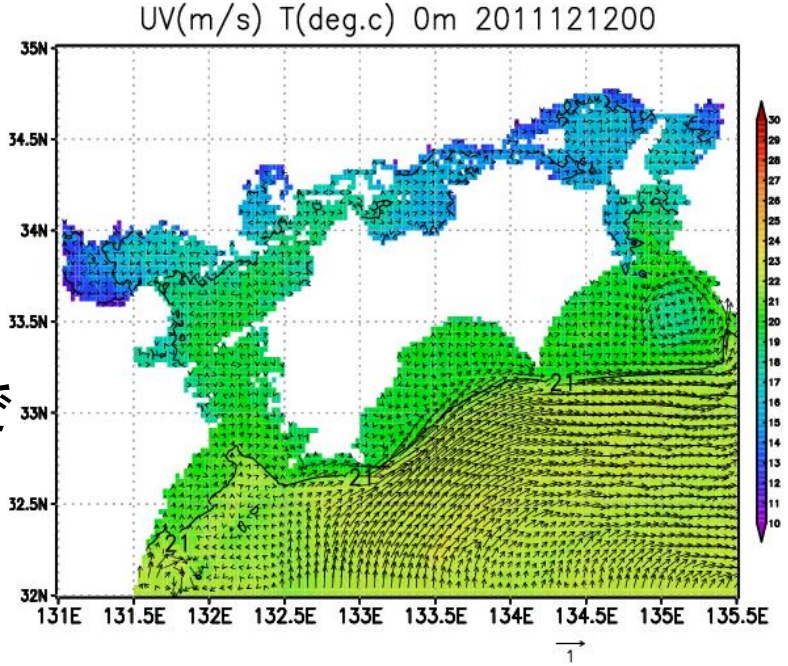
seto2/post_test1/ を作成、そこに移動
> make (nc2grd.x)
> nc2grd.x 1 730 ../run_test1/out/
変数毎にバイナリーデータに変換
u.2011010200.dat, v.2011010200.dat,
t.2011010200.dat, s.2011010200.dat
e.2011010200.dat …

可視化ソフトウェアの例

GrADS: 無料
http://iges.org/grads/



> grads -p
ga-> open t.2011041200.ctl
Scanning description file: t.2011041200.ctl
Data file t.2011041200.dat is open as file 1
LON set to 130.986 135.513
LAT set to 31.9861 35.014
LEV set to -1000 -1000
Time values set: 2011:4:12:0 2011:4:12:0
E set to 1 1
ga->



カタログファイルとの組み合わせで
バイナリーデータを
直接読むことができる