Pyroms – Python for ROMS

Kate Hedstrom
Frederic Castruccio
Bob Torgerson

Outline

- **Pyroms description**
- **Setup**
- **Grid generation**
- **Plotting**
- **Interpolation**

Functionality

- **Grid generation**
- **Bathymetry**
- **Interpolation**
- **Plotting**

Grid Generation

- **Based on Pavel Sakov's gridgen program**
- **Supports convex (beta=+1) and concave (beta=-1) corners**
- **Interactive or not, you decide**
  - Also interactive mask editing

Bathymetry

- **Comes with etopo2, can load any other on standard lat,lon grid**
- **Clip and smooth with smoothing options:**
  - Martinho and Batteen
  - Mellor, Ezer and Oey
  - Shapiro filter
  - Linear programming

Prerequisites

- Python 2.4-2.6, not 3.0 yet
- numpy and scipy
- netCDF4
- matplotlib
- basemap
- Fortran compiler
- ipython (optional)
- cmake

Installing Python Packages

- **If root, unpack package and in that directory:**
  - sudo python setup.py install
- **If not root, unpack package and in that directory:**
  - python setpy.py install –prefix=<pypath>
  - Add <pypath> to your PYTHONPATH environment variable

Download Pyroms

- **This one is git only:**

```
git clone https://github.com/
kshedstrom/pyroms.git
```

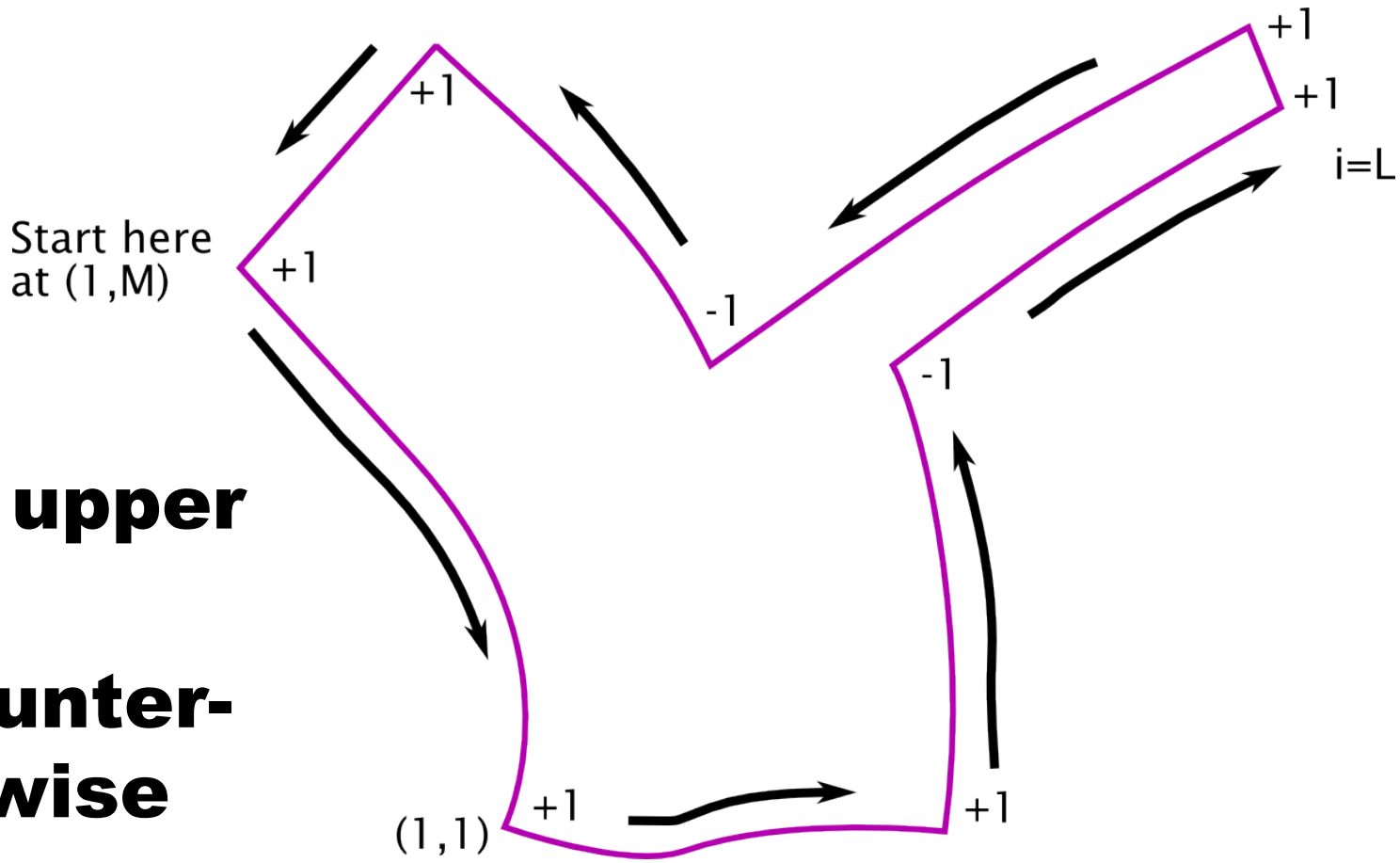
Setting up pyroms

- **Needs work, but right now we are using cmake**
- **Can't just do the usual setup.py because we need to compile external C/Fortran codes**
- **Read the INSTALL.pdf file**
  - Let's go through it…
- **Need to update because pyroms changed (yikes)**

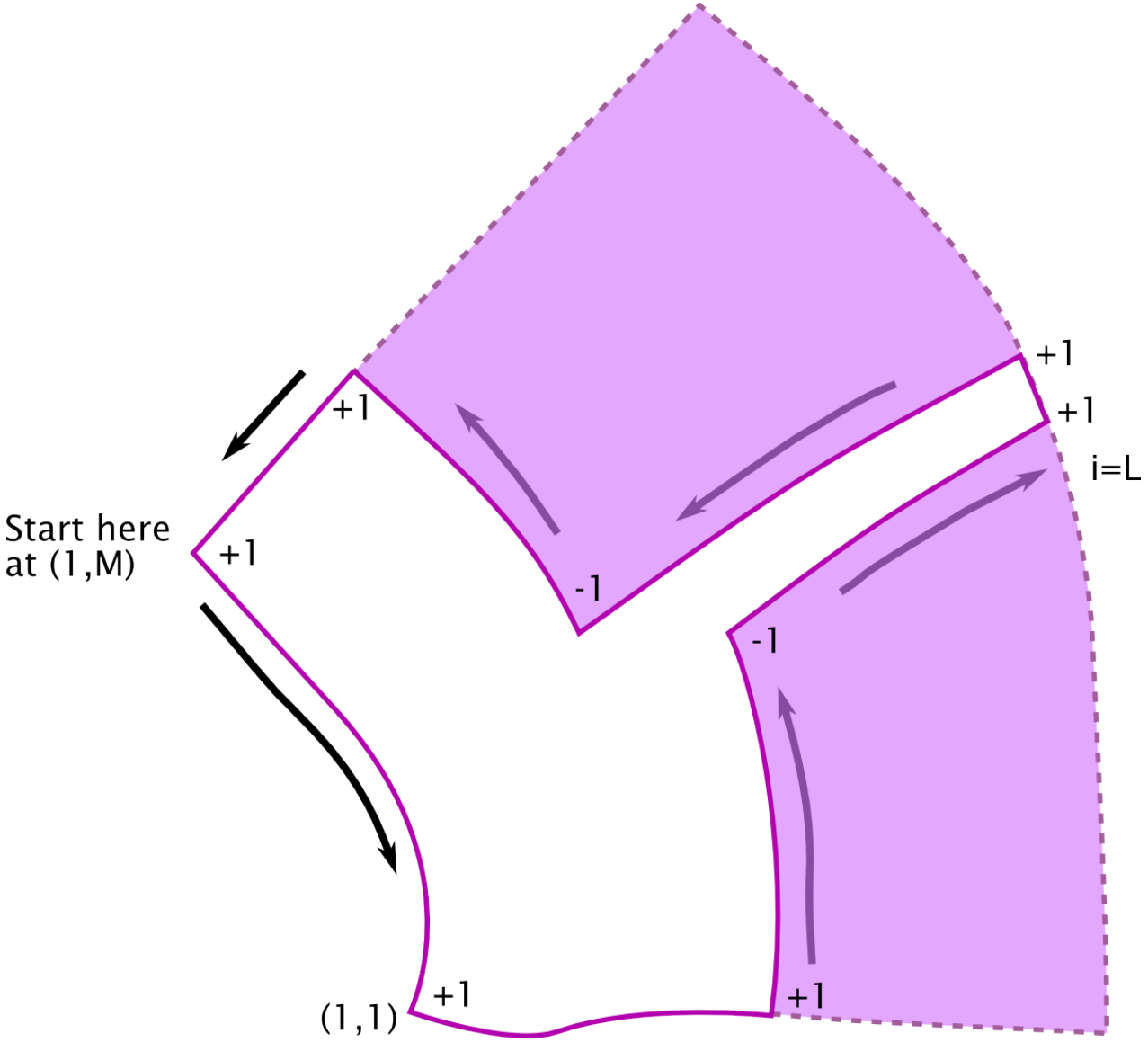
Grid Generation

- **Run interactively or in a script**
- **Fred sent me code from which to cut and paste**
- **http://www.arsc.edu/~kate/ROMS/HK/make_grid.py**
- **Let's give it a whirl…**

Boundary Selection



- **Begin upper left**
- **Go counter-clockwise**

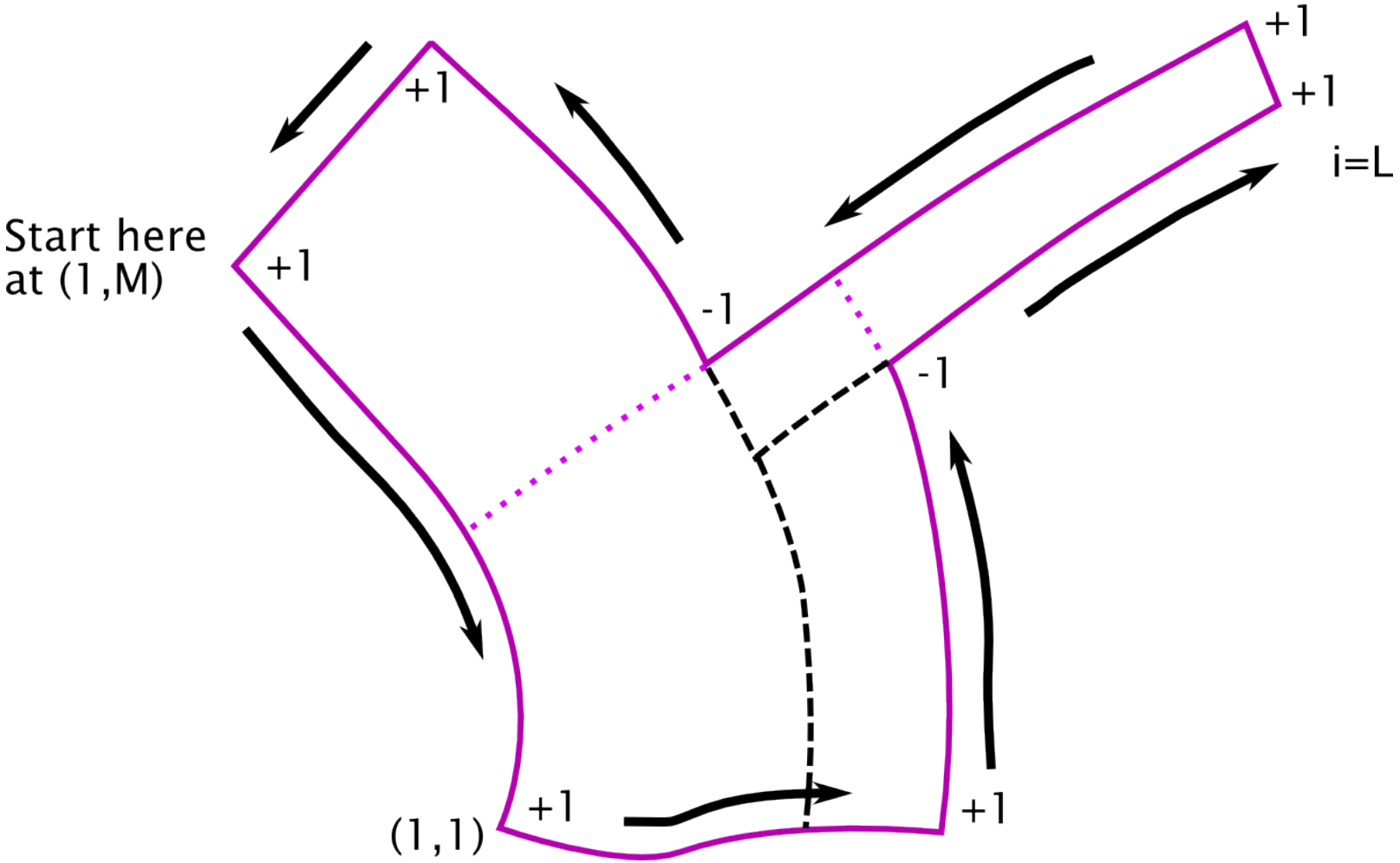

Start here
at (1,M)
+1
+1
+1
+1
i=L
-1
-1
(1,1)
+1
+1

Fill in the Pink Areas

- **Python code using grd object (needs work):**

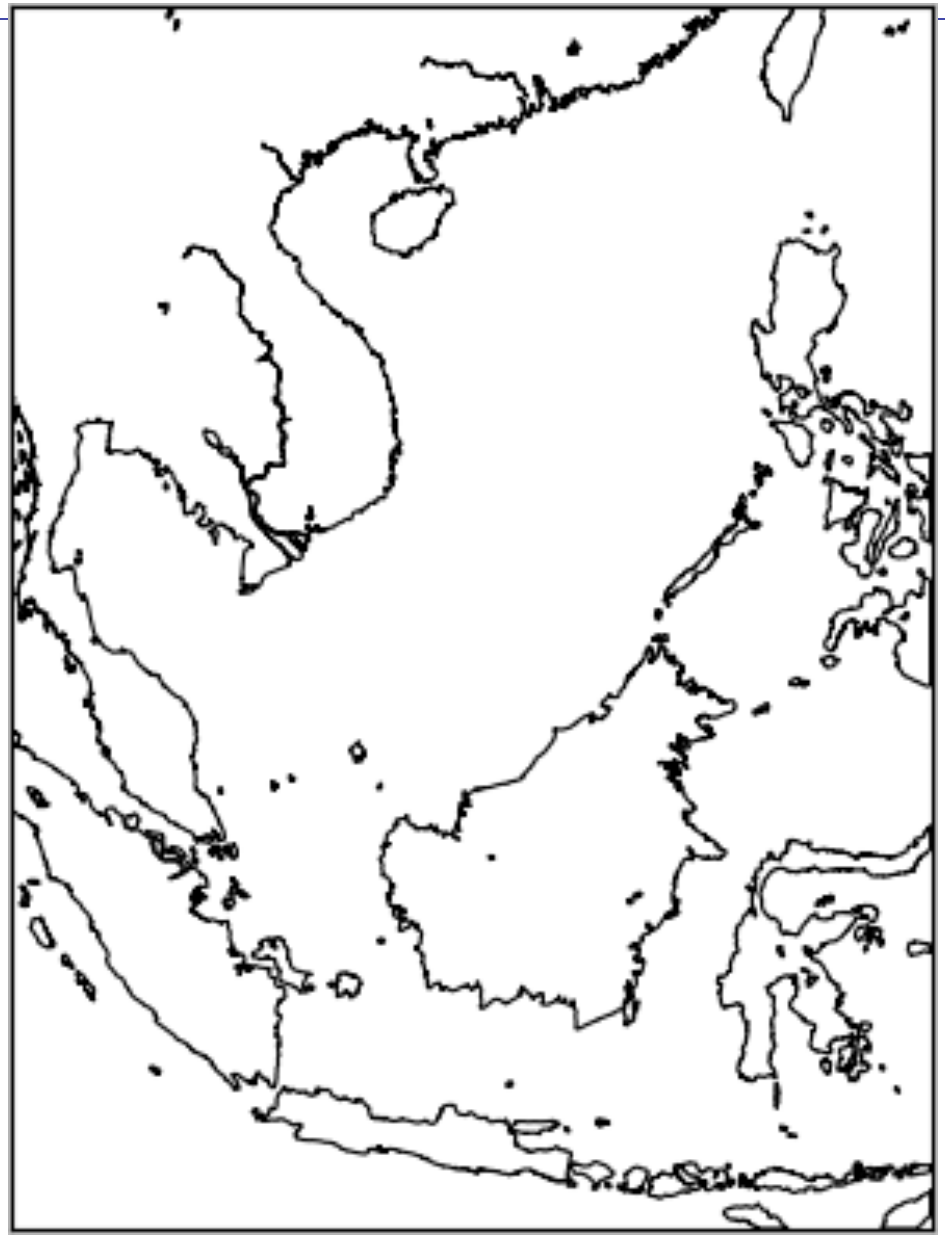
```
grd.dx = grd.dx.filled( grd.dx.mean() )
grd.dy = grd.dy.filled( grd.dy.mean() )
grd.dndx = grd.dndx.filled( grd.dndx.mean() )
grd.dmde = grd.dmde.filled( grd.dmde.mean() )
grd.angle = grd.angle.filled( grd.angle.mean() )
```

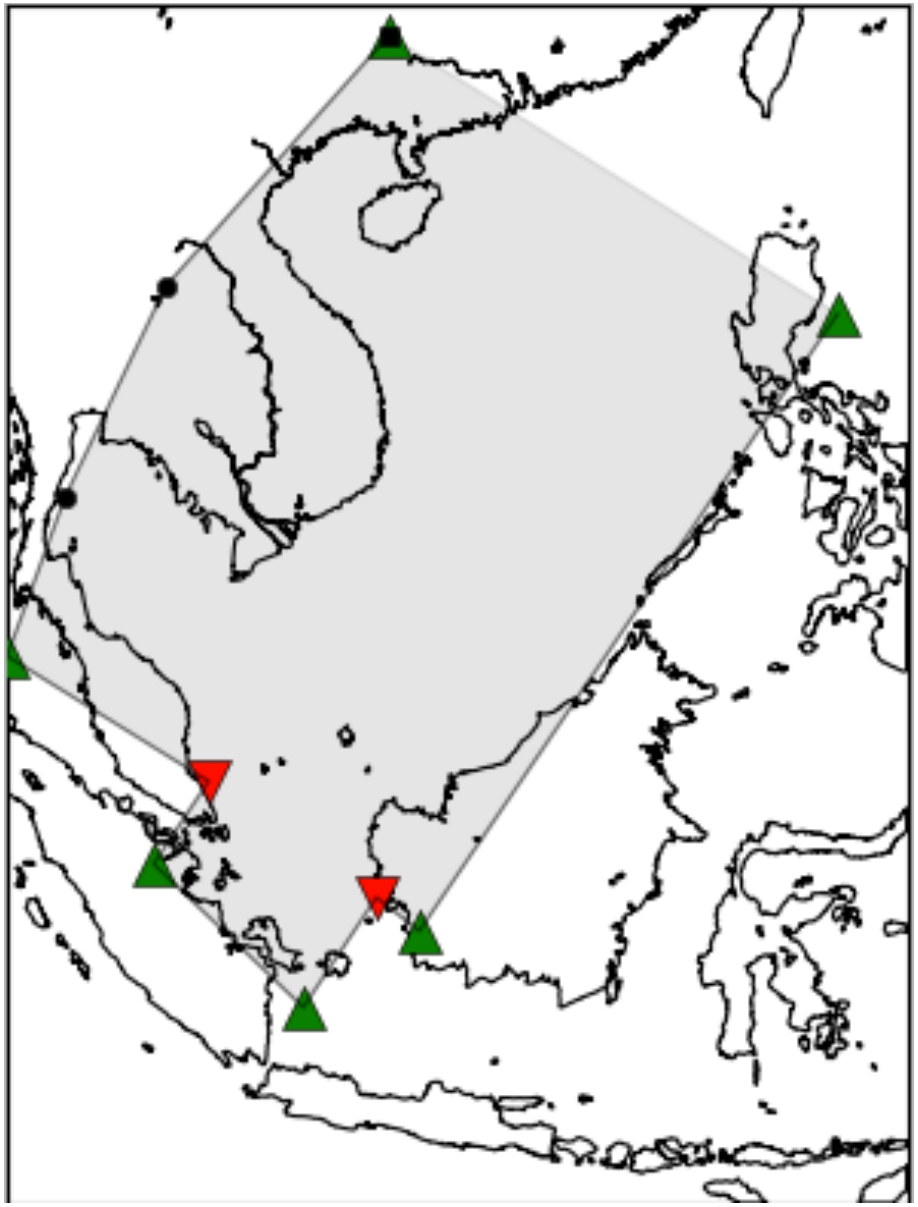
ROMS 4

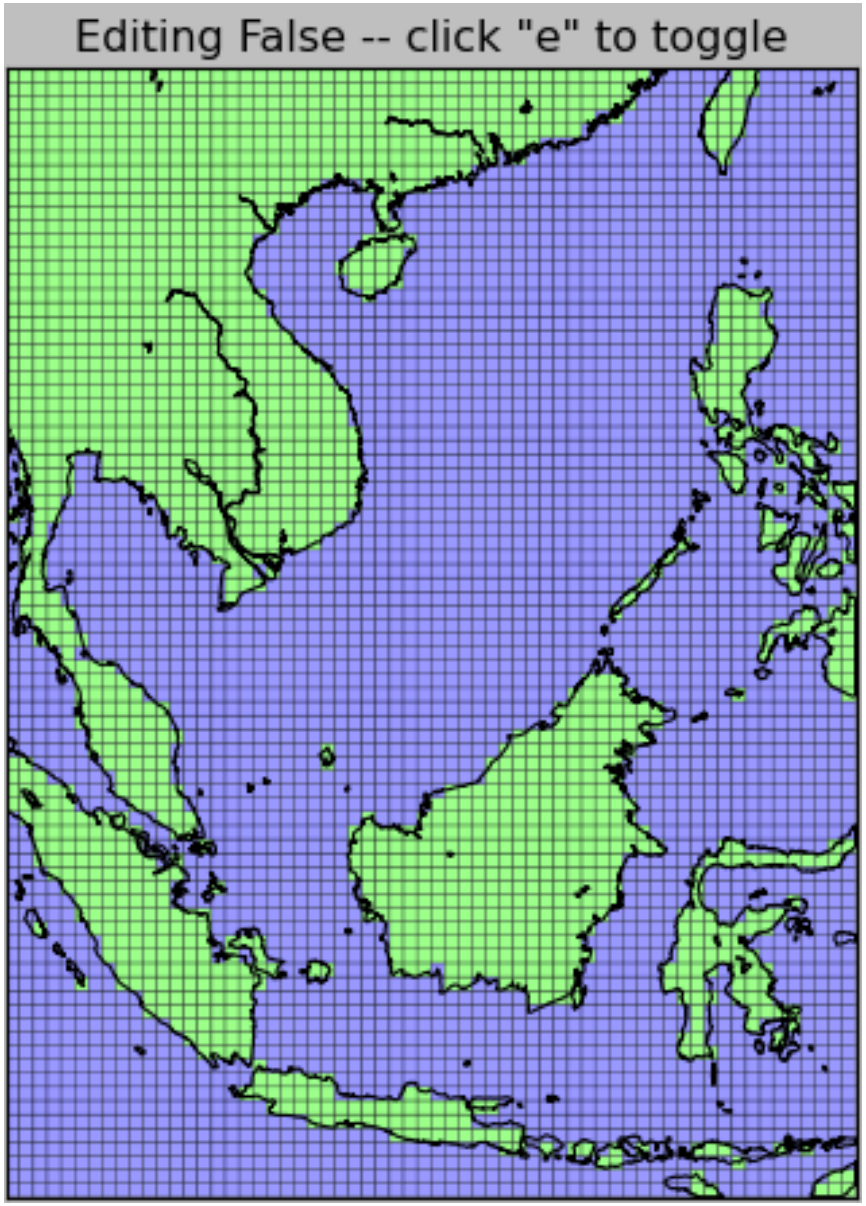


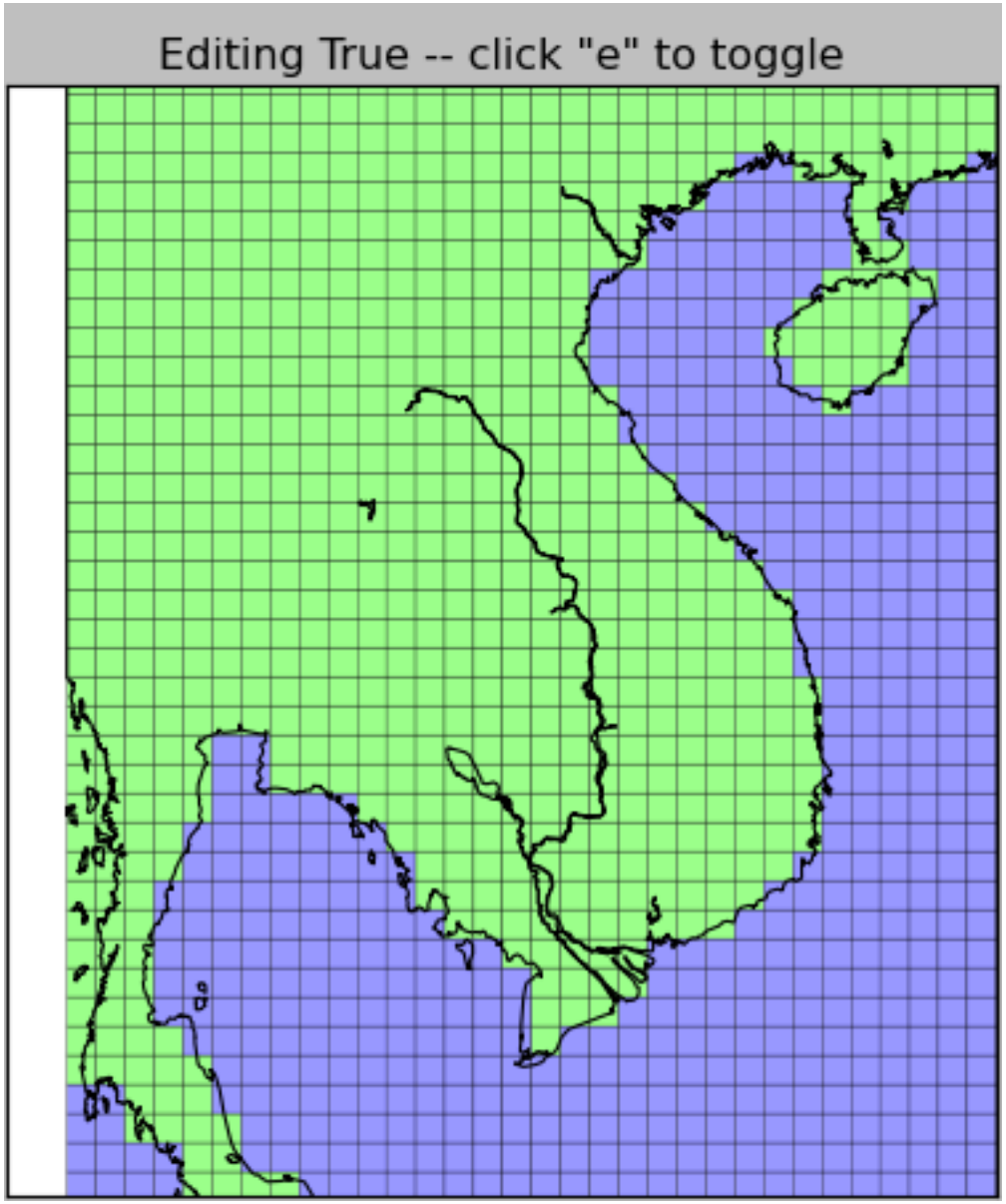
Interactive Commands

- i – new vertex
- d – delete a vertex
- p – set vertex as beta=1 (CCW)
- m – set vertex as beta=-1 (CW)
- G – generate grid
- Sum of betas must be 4






Editing False -- click "e" to toggle


Editing True -- click "e" to toggle

Cartesian Grids

- if you omit proj=map, gridgen will generate a Cartesian grid with x_rho, y_rho, x_u, y_u, … in meters for example
- See circle and box examples
- Reminder: ipython –pylab or else you need “from numpy import *”

Plotting

- **Knows about full ROMS geometry**
- **Set up info about your domain in an ascii file**
  - This info is used by the interpolations as well
- **Uses matplotlib for plotting, with all its warts (looks like Matlab plots)**

Gridid.txt

- **Pointed to by environment variable PYROMS_GRIDID_FILE**
- **Contains a chunk for each grid:**

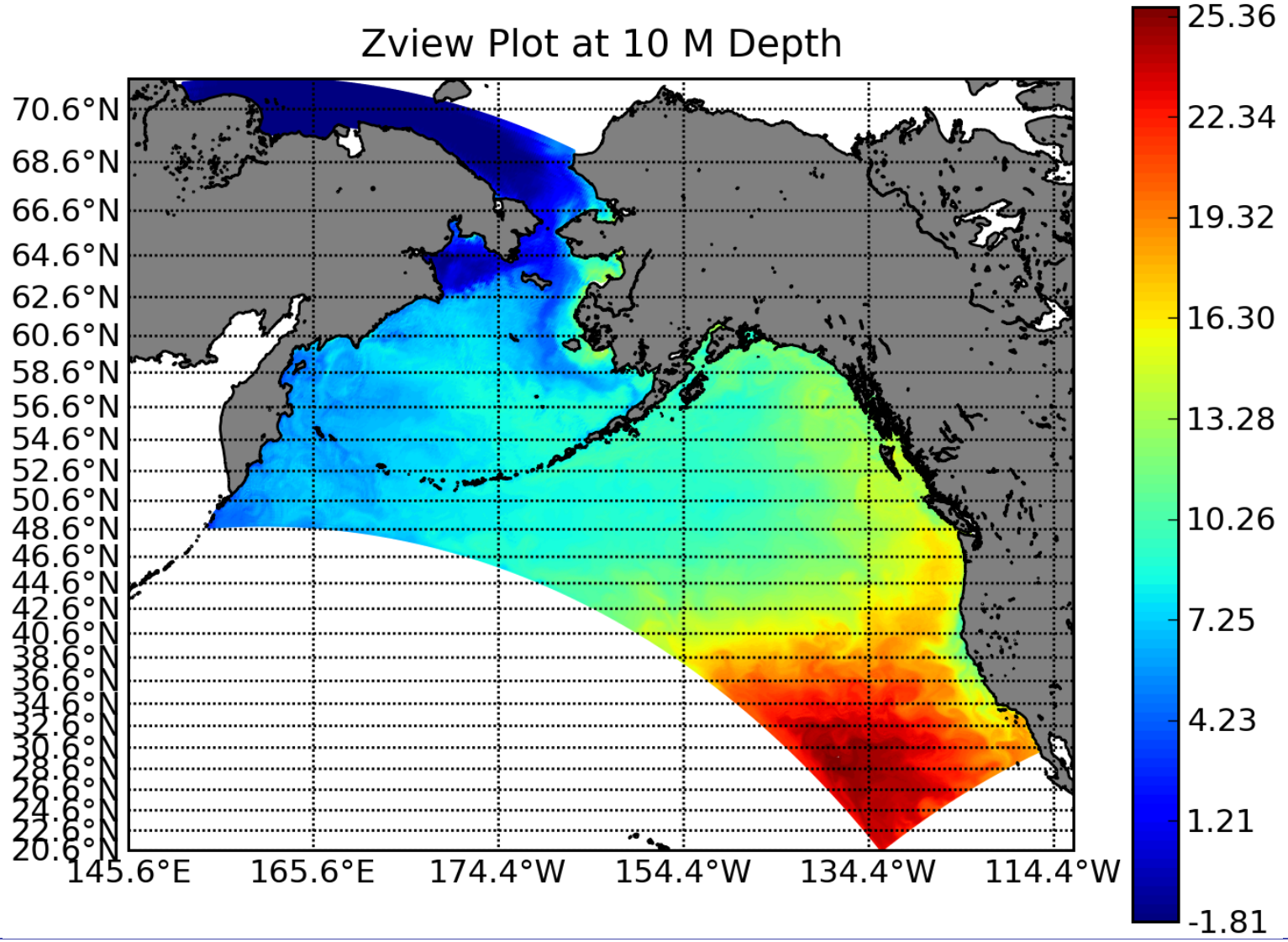
```
id      = BERING
name    = BERING
grdfile = /archive/u1/uaf/kate/gridpak/Bering/
   Bering_grid_4.nc
N       = 60
grdtype = roms
Vtrans = 1
theta_s = 5
theta_b = 0.4
Tcline  = 10
```

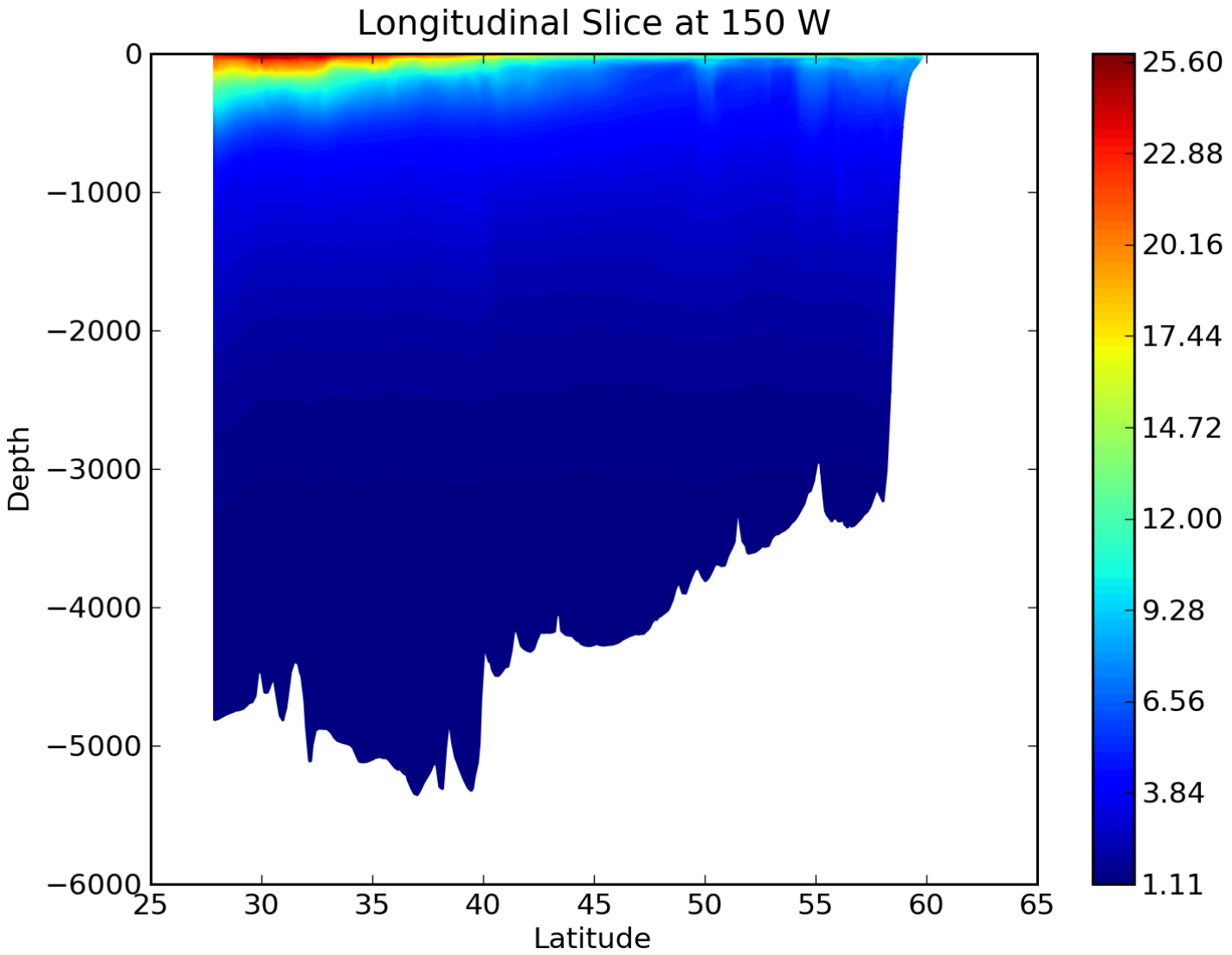
- **Grdtype can also be “z” for interpolating from MOM/POP**
- **Then need a list of depths:**

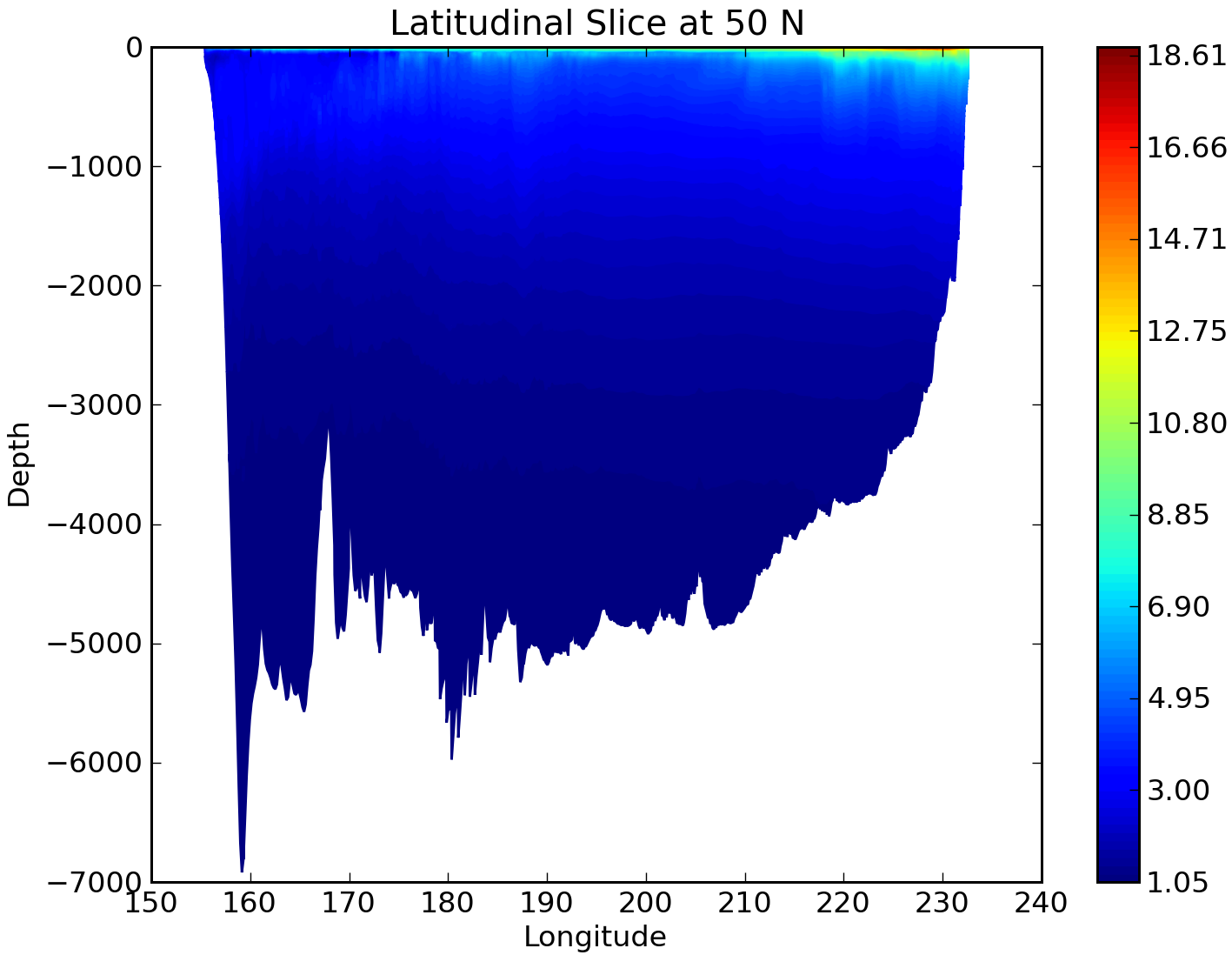
```
id       = ESPRESSO_Z
name     = ESPRESSO_Z
grdfile = /home/frederic/ROMS_projects/espresso/…
N        = 42
grdtype = z
depth    = [  -4500.  -4000.  -3500.  -3000.  -2500.
-2000.  -1750.  -1500.  -1250.  -1000. \
              -900.  -800.  -700.  -600.  -500.
-400.  -300.  -250.  -200.  -175.  -150.  -125. \
              -100.  -90.  -80.  -70.  -60.  -50.
-45.  -40.  -35.  -30.  -25.  -20.  -17.5  -15. \
              -12.5  -10.  -7.5  -5.  -2.5  0.  ]
```

Plotting code is in pyroms_toolbox

- Zview – constant z surface plots
- Sview – constant s surface plots
- Latview – constant latitude vertical slice
- Lonview, iview, jview – like above


Zview Plot at 10 M Depth
70.6°N
68.6°N
66.6°N
64.6°N
62.6°N
60.6°N
58.6°N
56.6°N
54.6°N
52.6°N
50.6°N
48.6°N
46.6°N
44.6°N
42.6°N
40.6°N
145.6°E
165.6°E
174.4°W
154.4°W
134.4°W
114.4°W
25.36
22.34
19.32
16.30
13.28
10.26
7.25
4.23
1.21
-1.81


Longitudinal Slice at 150 W
Depth
0
−1000
−2000
−3000
−4000
−5000
−6000
25
30
35
40
45
50
55
60
65
Latitude
25.60
22.88
20.16
17.44
14.72
12.00
9.28
6.56
3.84
1.11


Latitudinal Slice at 50 N
Depth
0
−1000
−2000
−3000
−4000
−5000
−6000
−7000
150
160
170
180
190
200
210
220
230
240
Longitude
18.61
16.66
14.71
12.75
10.80
8.85
6.90
4.95
3.00
1.05

Interpolation

- **For initial and boundary conditions from another run**
  - Either ROMS or POP (SODA)
- **Uses scrip and has to find scrip.so**
- **Scrip is a three-phase process:**
  - Generate the grid NetCDF files into the scrip input format
  - Generate the remapping weights
  - Do the interpolation

Boundary Conditions

- **Make one weights file for each**
  - Side of the grid you want BCs for
  - U, V, rho point on the grid
  - Could have 12 weights files!
- **Scrip is faster than rnt in Matlab, but BCs can still take time to generate**
- **Want to gather it all up into one BC file for ROMS at the end**