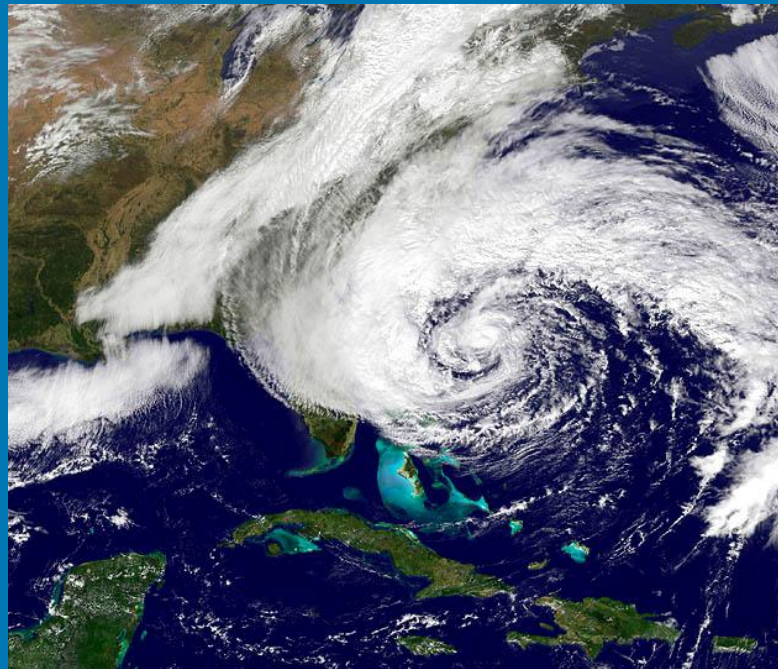


ROMS Application Tutorial



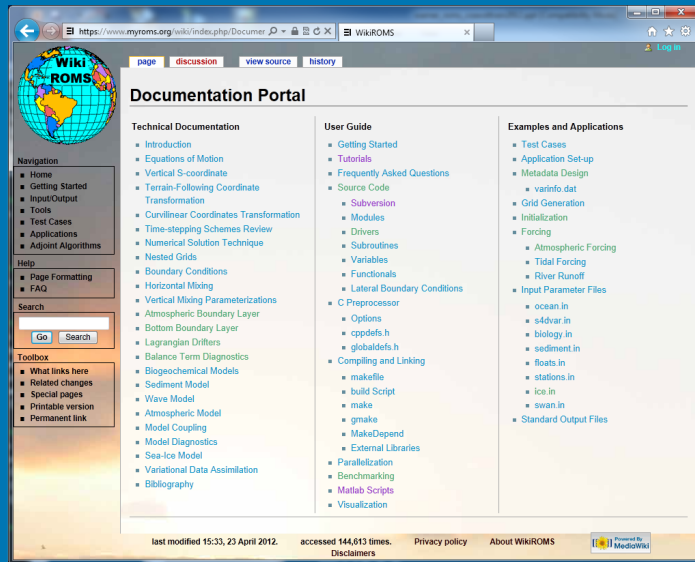
Projects/Sandy- Hurricane Sandy example



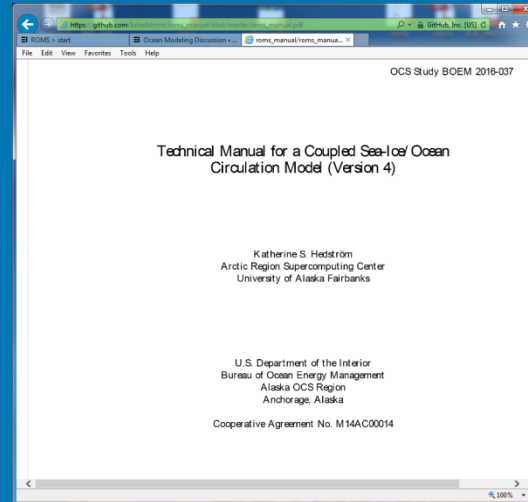
Regional Ocean Modeling System

- Free surface, hydrostatic ocean model
- Finite-difference 3D Reynolds-averaged Navier-Stokes equations
- Horizontal orthogonal curvilinear Arakawa C grid
- Vertical stretched terrain-following Sigma coordinates
- Wide range of advection schemes: (e.g. 3rd-order upstream-biased, 4th-order)
- Wide range of open boundary conditions: (e.g. Radiation, clamped, nudged)
- CF-compliant NetCDF I/O
- Wide range of vertical mixing schemes (k-epsilon, k-omega, MY2.5, KPP, GLS)
- Ice model
- Biological modules
- Model adjoint for data assimilation
- Fortran 90; Runs on Unix, Mac, and Windows
- Parallel code in MPI and OpenMP

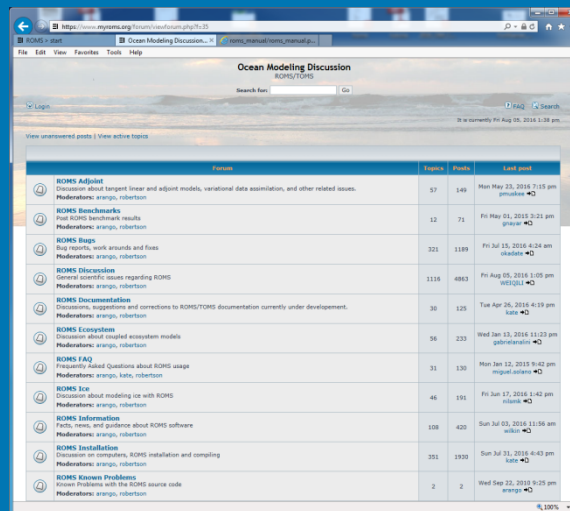
ROMS information



<https://www.myroms.org/wiki>



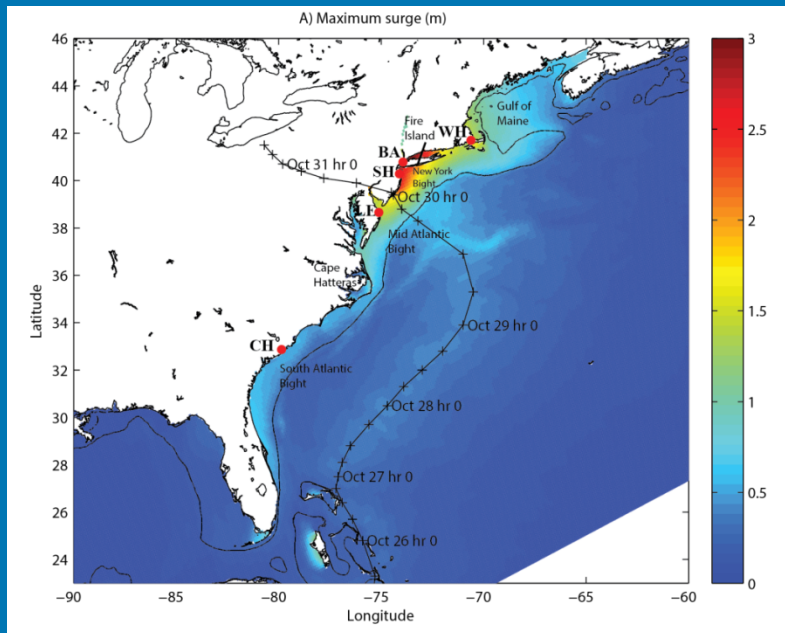
https://github.com/kshedstrom/roms_manual/blob/master/roms_manual.pdf



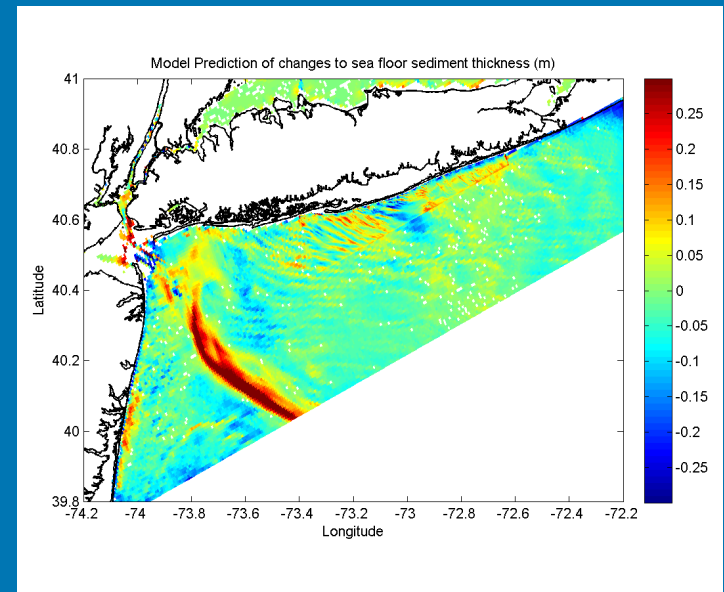
<https://www.myroms.org/forum/index.php>



Wide Range of realistic Applications



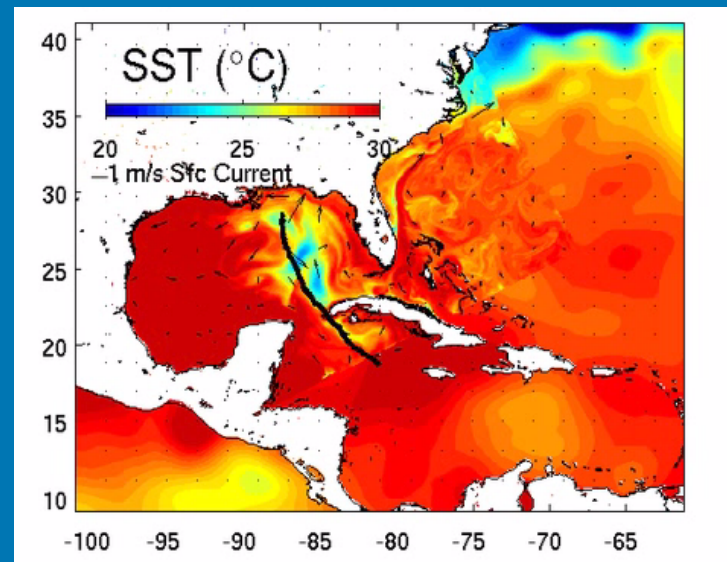
Storm surge (H. Sandy)



seafloor bed thickness change (H. Sandy)

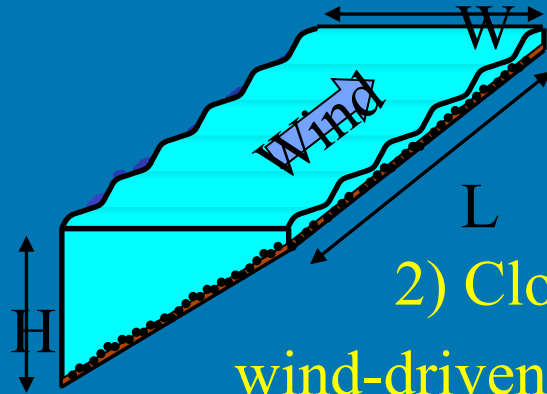
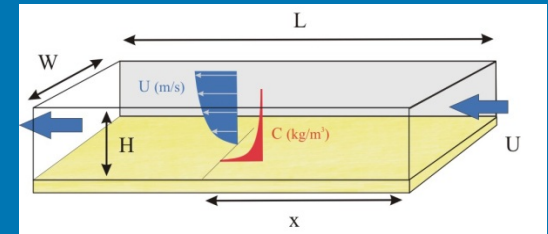


SST (H. Ivan)

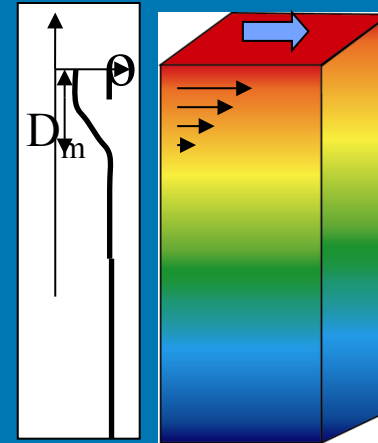


Test Cases

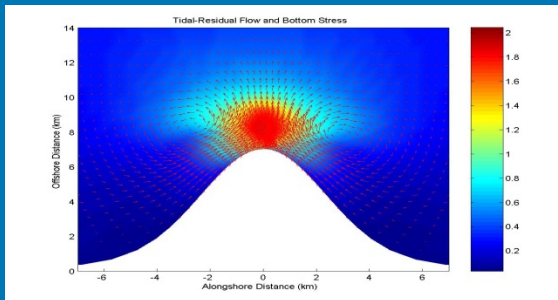
1) Open channel flow



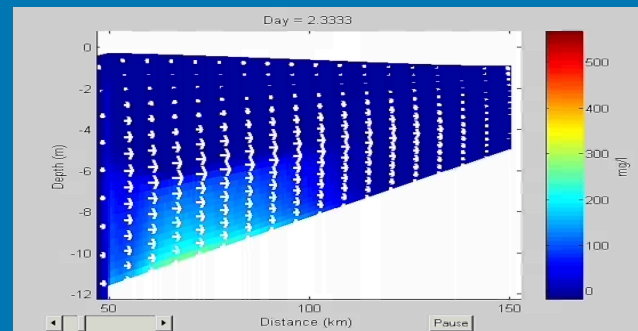
2) Closed basin, wind-driven circulation



3) Mixed layer deepening



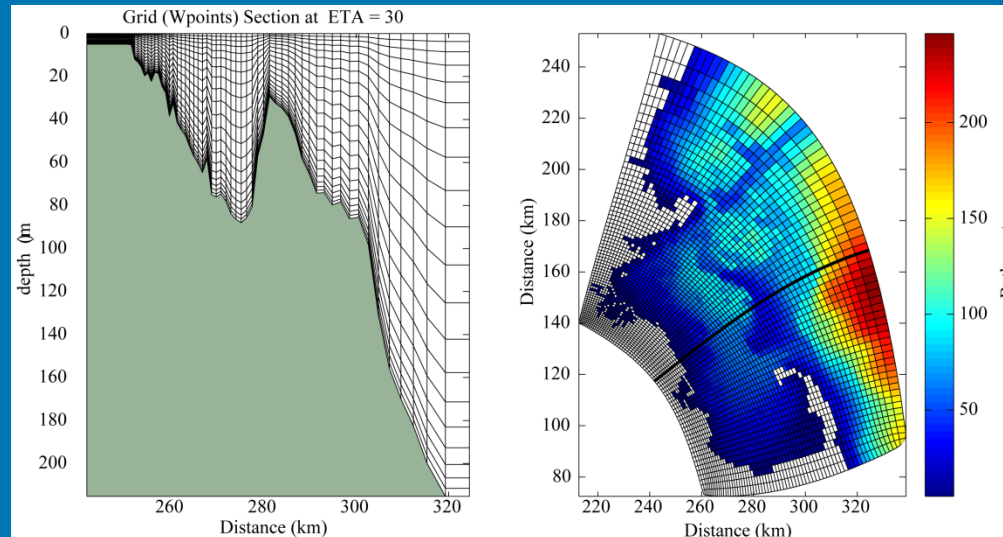
4) Tidal flow around a headland



5) Estuarine circulation

ROMS Grid

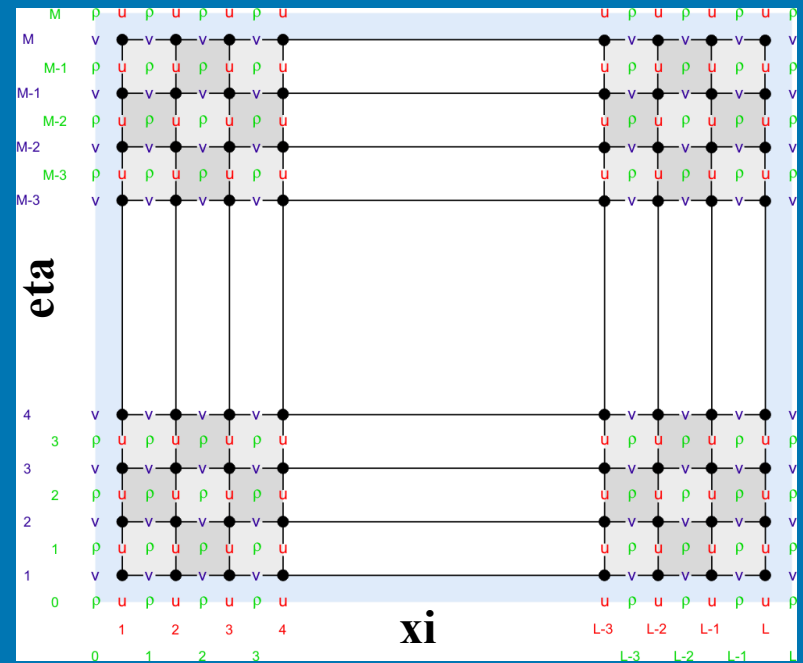
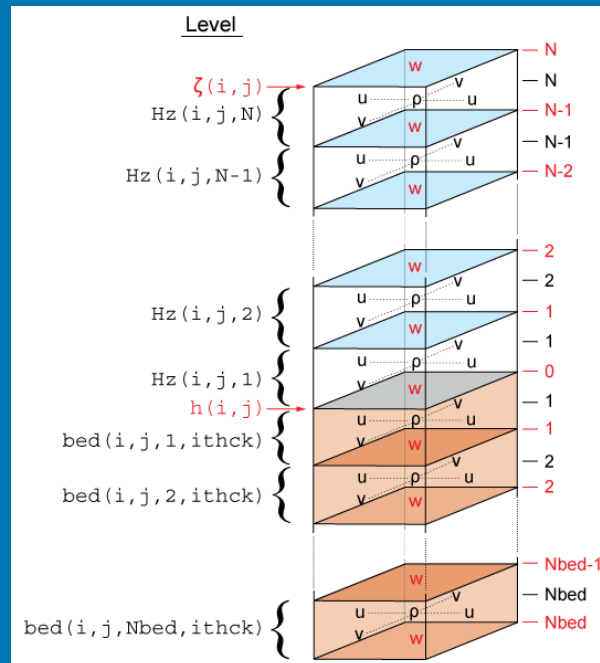
terrain following
coordinates



- masking
- curvature
- stretching

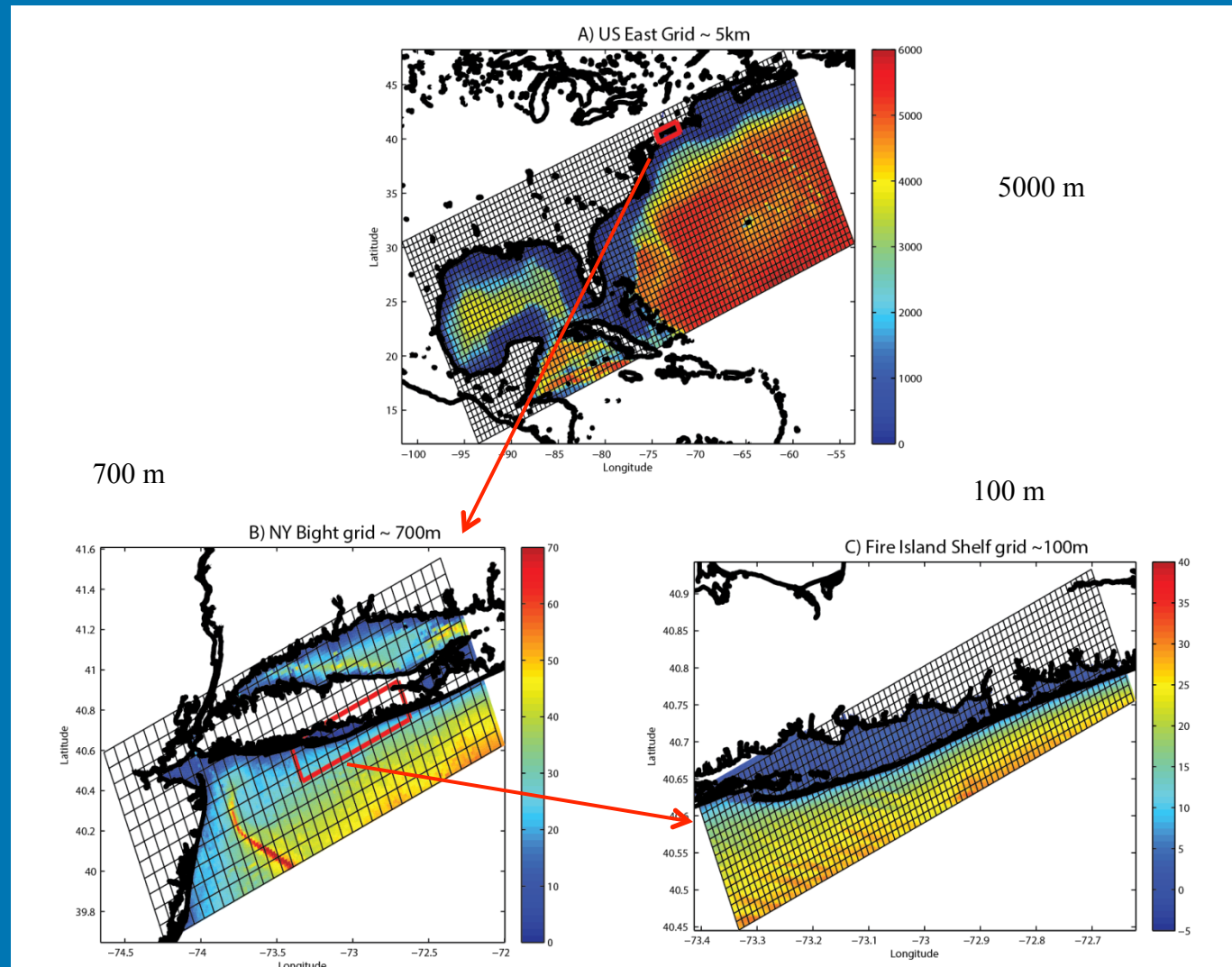
Horizontal
Arakawa "C" grid

Vertical



Grid resolution – horizontal static nests

For structured grids, use grid refinement to increase resolution.



Terrain following transformations

(Vtransform + Vstretching)

Vtransform

Transformation Equations

The following vertical coordinate transformations are available:

$$z(x, y, \sigma, t) = S(x, y, \sigma) + \zeta(x, y, t) \left[1 + \frac{S(x, y, \sigma)}{h(x, y)} \right], \quad (1)$$

$$S(x, y, \sigma) = h_c \sigma + [h(x, y) - h_c] C(\sigma)$$

or

$$z(x, y, \sigma, t) = \zeta(x, y, t) + [\zeta(x, y, t) + h(x, y)] S(x, y, \sigma),$$
$$S(x, y, \sigma) = \frac{h_c \sigma + h(x, y) C(\sigma)}{h_c + h(x, y)} \quad (2)$$

where $S(x, y, \sigma)$ is a nonlinear vertical transformation functional, $\zeta(x, y, t)$ is the time-varying free-surface, $h(x, y)$ is the unperturbed water column thickness and $z = -h(x, y)$ corresponds to the ocean bottom, σ is a fractional vertical stretching coordinate ranging from $-1 \leq \sigma \leq 0$, $C(\sigma)$ is a nondimensional, monotonic, vertical stretching function ranging from $-1 \leq C(\sigma) \leq 0$, and h_c is a positive thickness controlling the stretching. In sediment applications, $h = h(x, y, t)$ is changed at every time-step since it is affected by erosion and deposition processes.

1

★ 2

Vstretch (1 of 4)

Available Stretching Functions:

1. [Song and Haidvogel \(1994\)](#) function available in ROMS since its beginning, `Vstretching = 1`. $C(\sigma)$ is defined as:

$$C(\sigma) = (1 - \theta_B) \frac{\sinh(\theta_S \sigma)}{\sinh \theta_S} + \theta_B \left[\frac{\tanh[\theta_S(\sigma + \frac{1}{2})]}{2 \tanh(\frac{1}{2}\theta_S)} - \frac{1}{2} \right] \quad (5)$$

where θ_S and θ_B are the surface and bottom control parameters. Their ranges are $0 < \theta_S \leq 20$ and $0 \leq \theta_B \leq 1$, respectively. This function has the following features:

- It is infinitely differentiable in σ .
- The larger the value of θ_S , the more resolution is kept above h_c .
- For $\theta_B = 0$, the resolution all goes to the surface as θ_S is increased.
- For $\theta_B = 1$, the resolution goes to both the surface and the bottom equally as θ_S is increased.
- For $\theta_S \neq 0$, there is a subtle mismatch in the discretization of the model equations, for instance in the horizontal viscosity term. We recommend that you stick with *reasonable* values of θ_S , say $\theta_S \leq 8$.
- Some applications turn out to be sensitive to the value of θ_S used.

Vstretch (2 of 4)

2. A. Shchepetkin (2005) UCLA-ROMS deprecated function, [Vstretching = 2](#). $C(\sigma)$ is defined as a piecewise function:

$$C(\sigma) = \mu C_{sur}(\sigma) + (1 - \mu) C_{bot}(\sigma),$$

$$C_{sur}(\sigma) = \frac{1 - \cosh(\theta_S \sigma)}{\cosh(\theta_S) - 1}, \quad \text{for } \theta_S > 0, \quad C_{bot}(\sigma) = \frac{\sinh[\theta_B (\sigma + 1)]}{\sinh(\theta_B)} - 1,$$

$$\mu = (\sigma + 1)^\alpha \left[1 + \frac{\alpha}{\beta} (1 - (\sigma + 1)^\beta) \right]$$

This function is similar in meaning to the [Song and Haidvogel \(1994\)](#), but note that hyperbolic function in C_{sur} is $\cosh$ instead of $\sinh$ and

$$\frac{\partial C}{\partial \sigma} \rightarrow 0 \quad \text{as } \sigma \rightarrow 0.$$

Vstretch (3 of 4)

3. R. Geyer function for high bottom boundary layer resolution in relatively shallow applications, [Vstretching](#) =

3. $C(\sigma)$ is defined as a piecewise function:

$$C(\sigma) = \mu C_{bot}(\sigma) + (1 - \mu) C_{sur}(\sigma),$$

$$C_{sur}(\sigma) = -\frac{\log[\cosh(\gamma \text{abs}(\sigma)^{\theta_S})]}{\log[\cosh(\gamma)]}, \quad C_{bot}(\sigma) = \frac{\log[\cosh(\gamma (\sigma + 1)^{\theta_B})]}{\log[\cosh(\gamma)]} - 1,$$

$$\mu = \frac{1}{2} \left[1 - \tanh \left(\gamma \left(\sigma + \frac{1}{2} \right) \right) \right]$$

where the power exponents θ_S and θ_B are the surface and bottom control parameters specified in standard input file [ocean.in](#), as before. Here, γ is a scale factor for all hyperbolic functions. Currently, $\gamma = 3$. Typical values for the control parameters are:

$\theta_S = 0.65$	minimal increase of surface resolution
$\theta_S = 1.0$	significant surface amplification
$\theta_B = 0.58$	no bottom amplification
$\theta_B = 1.0$	significant bottom amplification

Vstretch (4 of 4)

4. A. Shchepetkin (2010) UCLA-ROMS current function, $Vstretching = 4$. $C(\sigma)$ is defined as a continuous, double stretching function:

Surface refinement function:

$$C(\sigma) = \frac{1 - \cosh(\theta_S \sigma)}{\cosh(\theta_S) - 1}, \quad \text{for } \theta_S > 0, \quad C(\sigma) = -\sigma^2, \quad \text{for } \theta_S \leq 0$$

Bottom refinement function:

$$C(\sigma) = \frac{\exp(\theta_B C(\sigma)) - 1}{1 - \exp(-\theta_B)}, \quad \text{for } \theta_B > 0 \quad (9)$$

Notice that the bottom function (9) is the second stretching of an already stretched transform (8). The resulting stretching function is continuous with respect to θ_S and θ_B as their values approach zero. The range of meaningful values for θ_S and θ_B are:

$$0 \leq \theta_S \leq 10 \quad \text{and} \quad 0 \leq \theta_B \leq 4$$

However, users need to pay attention to extreme r-factor (rx1) values near the bottom.

💡 Due to its functionality and properties $Vtransform = 2$ and $Vstretching = 4$ are now the default values for ROMS.

Equations in Mass Flux form

$$(\mathbf{u}^l, \omega^l) = (\mathbf{u}, \omega) + (\mathbf{u}^{St}, \omega^{St})$$

Continuity

$$\frac{\partial}{\partial t} \left(\frac{H_z}{mn} \right) + \frac{\partial}{\partial \xi} \left(\frac{H_z u^l}{n} \right) + \frac{\partial}{\partial \eta} \left(\frac{H_z v^l}{m} \right) + \frac{\partial}{\partial s} \left(\frac{\omega_s^l}{mn} \right) = 0$$

xi-direction Momentum Balance

$$\begin{aligned} & \underbrace{\frac{\partial}{\partial t} \left(\frac{H_z}{mn} u \right)}_{\text{ACC}} + \underbrace{\frac{\partial}{\partial \xi} \left(\frac{H_z u}{n} u \right) + \frac{\partial}{\partial \eta} \left(\frac{H_z v}{m} u \right) + u \frac{\partial}{\partial \xi} \left(\frac{H_z u^{St}}{n} \right) + u \frac{\partial}{\partial \eta} \left(\frac{H_z v^{St}}{m} \right)}_{\text{HA}} \\ & + \underbrace{\frac{\partial}{\partial s} \left(\frac{\omega_s}{mn} u \right) + u \frac{\partial}{\partial s} \left(\frac{\omega_s^{St}}{mn} \right)}_{\text{VA}} - \underbrace{H_z \left(\frac{fv}{mn} \right)}_{\text{COR}} - \underbrace{H_z \left(\frac{fv^{St}}{mn} \right)}_{\text{StCOR}} = - \underbrace{\frac{H_z}{n} \frac{\partial \varphi^c}{\partial \xi}}_{\text{PG}} + \underbrace{H_z v^{St} \left(\frac{1}{n} \frac{\partial v}{\partial \xi} - \frac{1}{m} \frac{\partial u}{\partial \eta} \right) - \omega_s^{St} \frac{\partial}{\partial s} \left(\frac{u}{mn} \right)}_{\text{HVF}} \\ & + \underbrace{\frac{H_z F^\xi}{mn}}_{\text{BF}} + \underbrace{\frac{H_z F^{w\xi}}{mn}}_{\text{BA+RA+BtSt+SuSt}} + \underbrace{\frac{H_z D^\xi}{mn}}_{\text{HM}} - \underbrace{\frac{\partial}{\partial s} \left(\overline{u'w'} - \frac{v}{H_z} \frac{\partial u}{\partial s} \right)}_{\text{VM}} + \underbrace{\hat{F}^u}_{\text{FCurv}} \end{aligned}$$

u, v, ω_s	= Eulerian Velocity
u^l, v^l, ω_s^l	= Lagrangian Velocity
$u^{St}, v^{St}, \omega_s^{St}$	= Stokes Velocity
f	= Coriolis parameter
φ	= Dynamic pressure
H_z	= Grid cell thickness
$\mathcal{F} \uparrow \eta; \mathcal{F} \uparrow \xi$	= Non wave body force
$D \uparrow \eta; D \uparrow \xi$	= Momentum mixing terms
$\mathcal{F} \uparrow w \eta; \mathcal{F} \uparrow w \xi$	= Non-conservative wave force

eta-Direction Momentum Balance

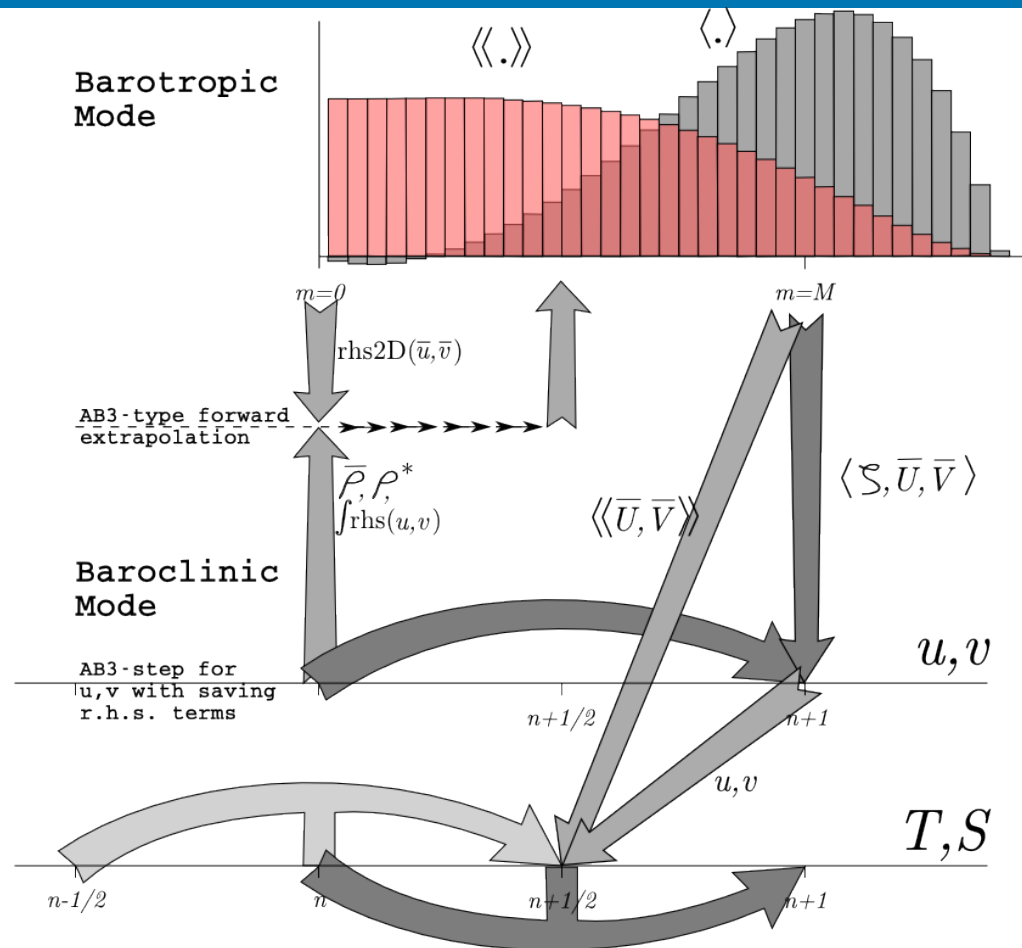
$$\begin{aligned}
 & \underbrace{\frac{\partial}{\partial t} \left(\frac{H_z}{mn} v \right)}_{\text{ACC}} + \underbrace{\frac{\partial}{\partial \xi} \left(\frac{H_z u}{n} v \right) + \frac{\partial}{\partial \eta} \left(\frac{H_z v}{m} v \right) + v \frac{\partial}{\partial \xi} \left(\frac{H_z u^{st}}{n} \right) + v \frac{\partial}{\partial \eta} \left(\frac{H_z v^{st}}{m} \right)}_{\text{HA}} \\
 & + \underbrace{\frac{\partial}{\partial s} \left(\frac{\omega_s}{mn} v \right) + v \frac{\partial}{\partial s} \left(\frac{\omega_s^{st}}{mn} \right)}_{\text{VA}} + \underbrace{H_z \left(\frac{fu}{mn} \right)}_{\text{COR}} + \underbrace{H_z \left(\frac{fu^{st}}{mn} \right)}_{\text{StCOR}} = \underbrace{- \frac{H_z}{m} \frac{\partial \varphi^c}{\partial \eta}}_{\text{PG}} - \underbrace{H_z u^{st} \left(\frac{1}{n} \frac{\partial v}{\partial \xi} - \frac{1}{m} \frac{\partial u}{\partial \eta} \right) - \omega_s^{st} \frac{\partial}{\partial s} \left(\frac{v}{mn} \right)}_{\text{HVF}} \\
 & + \underbrace{\frac{H_z F^\eta}{mn}}_{\text{BF}} + \underbrace{\frac{H_z F^{w\eta}}{mn}}_{\text{BA+RA+BtSt+SuSt}} + \underbrace{\frac{H_z D^\eta}{mn}}_{\text{HM}} - \underbrace{\frac{\partial}{\partial s} \left(\frac{v}{H_z} \frac{\partial v}{\partial s} \right)}_{\text{VM}} + \underbrace{\hat{F}^v}_{\text{FCurv}}
 \end{aligned}$$

Description of Terms

ACC	= Local Acceleration
HA	= Horizontal Advection
VA	= Vertical Advection
COR	= Coriolis Force
StCOR	= Stokes-Coriolis Force
PG	= Pressure Gradient
HVF	= Horizontal Vortex Force
BF	= Body Force
BA+RA+BuSt+SuSt	= Breaking Acceleration+ Roller Acceleration+ Bottom Streaming+ Surface Streaming
HM	= Horizontal Mixing
VM	= Vertical Mixing
Fcurv	= Curvilinear terms

Kumar, N., Voulgaris, G., Warner, J.C., and M., Olabarrieta (2012). Implementation of a vortex force formalism in a coupled modeling system for inner-shelf and surf-zone applications. *Ocean Modelling*, 47, 65-95.

Solution techniques- mode splitting



Shchepetkin, A. F., and J. C. McWilliams, 2008: Computational kernel algorithms for fine-scale, multi-process, long-time oceanic simulations. In: *Handbook of Numerical Analysis: Computational Methods for the Ocean and the Atmosphere*, eds. R. Temam & J. Tribbia, Elsevier Science, ISBN-10: 0444518932, ISBN-13: 978-0444518934.

Numerical algorithms

$$\frac{\partial}{\partial t} \frac{H_z C}{mn} = -\frac{\partial}{\partial \xi} F^\xi - \frac{\partial}{\partial \eta} F^\eta - \frac{\partial}{\partial \sigma} F^\sigma$$

Advection schemes

- 2nd order centered
- 4th order centered
- 4th order Akima
- TSU3 3rd order
- MPDATA
- HSIMT



$$\begin{aligned} F^\xi &= \frac{\overline{H_z^\xi} u \overline{C}^\xi}{\overline{m}^\xi} \\ F^\eta &= \frac{\overline{H_z^\eta} v \overline{C}^\eta}{\overline{m}^\eta} \\ F^\sigma &= \frac{\overline{H_z^\sigma} \Omega \overline{C}^\sigma}{mn} \end{aligned}$$

many choices, see manual for details.

How do we select different schemes

- c pre-processor definitions (list them in your project *.h “header” file)
- during compilation, .F → .f90s
- compiles f90's into objects
- compiles objects to libs
- ar the libs to make 1 exe
- for coupling, it makes wrf, roms, and swan as libs, then pull them together for coupling and only produces one exe → coawstM [.exe]

Build errors

- During the build, the error “can not make wrf.exe” or ‘undefined reference to MAIN’ is ok.

libdev/frtl/src/libfor/for_main.c:(.text+0x2a): undefined reference to `MAIN__'

make[2]: [em_wrf] Error 1 (ignored)

This “Error” is ok.

If you get other errors, do this:

scrip build.txt

./coawst.bash

exit

and that will create a file build.txt

Edit that file and search for the word “error”

COAWST specific cpp options

You need to use at least 1 model of the 3:

- `#define ROMS_MODEL` if you want to use the ROMS model
- `#define SWAN_MODEL` if you want to use the SWAN model
- `#define WRF_MODEL` if you want to use the WRF model

- `#define MCT_LIB` if you have more than one model selected

Use these 3 INTERP calls if the models are on different grids.

- `#define MCT_INTERP_WV2AT` allows grid interpolation between the wave and atmosphere models
- `#define MCT_INTERP_OC2AT` allows grid interpolation between the ocean and atmosphere models
- `#define MCT_INTERP_OC2WV` allows grid interpolation between the ocean and wave models

- `#define NESTING` allows grid refinement in roms or in swan
For now, I suggest if you couple ROMS+SWAN and use NESTING, then use the same grids for R+S.

COAWST specific cpp options

If you activate ROMS + WRF, then pick one of these:

- `#define ATM2OCN_FLUXES` provide consistent fluxes between atm and ocn.
WRF send USTRESS,VSTRESS, LH, HFX, GSW, GLW
- OR ---
- `#define BULK_FLUXES` If you don't use this, then
this will send Uwind, Vwind, Patm, RH, Tair,
cloud, GSW, GLW

Related:

- `#define EMINUSP` send EVAP and RAIN from WRF to ROMS
- `#define CLOUDS` send cloud fraction from WRF to ROMS
- `#define ATM_PRESS` send MSLP (Patm) from WRF to ROMS

These 3 options use SWAN wave data for computation of ocean surface stress in ROMS bulk_fluxes and in WRF myjsfc and mynn surface layer schemes :

- `#define COARE_TAYLOR_YELLAND` wave enhanced roughness (swan to roms or wrf)
- `#define COARE_OOST` wave enhanced roughness (swan to roms or wrf)
- `#define DRENNAN` wave enhanced roughness (swan to roms or wrf)
- `#define DRAGLIM_DAVIS` feature added to WRF and SWAN to limit the ocean roughness drag to be a maximum of 2.85E-3

COAWST specific cpp options

If you couple ROMS + WRF then you can select this for testing:

- `#define SST_CONST` do not allow SST from ROMS to affect WRF

Wave Options

- `#define UV_CONST` send vel = 0 from the ocn to wave model
- `#define ZETA_CONST` send zeta = 0 from the ocn to wave model
- `#define UV_KIRBY` compute "depth-avg" current based on Hwave to be sent from the ocn to the wav model for coupling
- `#define WEC_MELLOR` radiation stress terms from Mellor 08
- `#define WEC_VF` wave-current stresses from Uchiyama et al.
- `#define WDISS_THORGUZA` wave dissipation from Thorton/Guza
- `#define WDISS_CHURTHOR` wave dissipation from Church/Thorton
- `#define WDISS_WAVEMOD` wave dissipation from a wave model
- `#define WDISS_INWAVE` wave dissipation from a InWave model
- `#define ROLLER_SVENDSEN` wave roller based on Svendsen
- `#define ROLLER_MONO` wave roller for monochromatic waves
- `#define ROLLER_RENIERS` wave roller based on Reniers
- `#define BOTTOM_STREAMING` wave enhanced bottom streaming
- `#define SURFACE_STREAMING` wave enhanced surface streaming

COAWST specific cpp options

Wave Options

- `#define CHARNOK` Charnok surface roughness from wind stress
- `#define CRAIG_BANNER` Craig and Banner wave breaking surface flux
- `---` or `---`
- `#define ZOS_HSIG` surface roughness from wave amplitude
- `#define TKE_WAVEDISS` wave breaking surface flux from wave amplitude

Vegetation Options

- `# define VEGETATION` Activate vegetation module
- `# define VEG_DRAG` Drag terms Luhar M. et.al (2011)
- `# define VEG_FLEX` Flexible vegetation terms
- `# define VEG_TURB` Turbulence terms, Uittenbogaard R. (2003)
- `# define VEG_SWAN_COUPLING` Exchange of VEG data btwn. ROMS and SWAN
- `# define VEG_STREAMING` Wave streaming effects
- `# define MARSH_WAVE_THRUST` Wave thrust on marshes, Tonelli, M. et al. (2010)

Tracer Advection Option

- `# define TS_HSIMT` Positive definite tracer advection (Wu and Zhu, OM 2010)



+ ROMS Specific CPP options

ROMS/Include/cppdefs.h

See cppdefs for a complete list of ROMS options

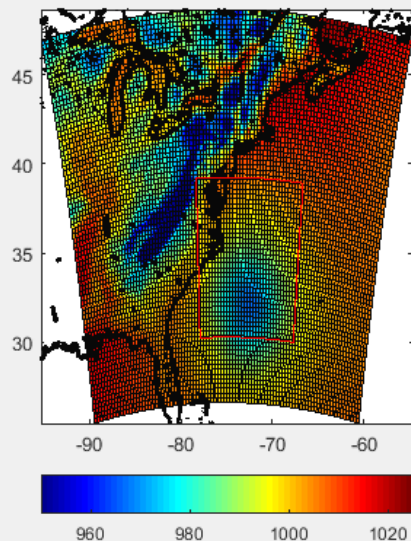
```
TextPad - C:\work\models\COAWST\ROMS\Include\cppdefs.h
File Edit Search View Tools Macros Configure Window Help
Find incrementally

cppdefs.h x
**
** The following is short description of all available CPP options.
**
** OPTIONS associated with momentum equations:
**
** The default horizontal advection is 3rd-order upstream bias for
** 3D momentum and 4th-order centered for 2D momentum. The default
** vertical advection is 4th-order centered for 3D momentum. If this
** is the case, no flags for momentum advection need to be activated.
**
** The 3rd-order upstream split advection (UV_U3ADV_SPLIT) can be used
** to correct for the spurious mixing of the advection operator in
** terrain-following coordinates. If this is the case, the advection
** operator is split in advective and viscosity components and several
** internal flags are activated in "globaldefs.h". Notice that
** horizontal and vertical advection of momentum is 4th-order centered
** plus biharmonic viscosity to correct for spurious mixing. The total
** time-dependent horizontal mixing coefficient are computed in
** "hmixing.F".
**
** WARNING: Use the splines vertical advection option (UV_SADVECTION)
** only in idealized, high vertical resolution applications.
**
** UV_ADV          use to turn ON or OFF advection terms
** UV_COR          use to turn ON or OFF Coriolis term
** UV_U3ADV_SPLIT  use if 3rd-order upstream split momentum advection
** UV_C2ADVECTION  use to turn ON or OFF 2nd-order centered advection
** UV_C4ADVECTION  use to turn ON or OFF 4th-order centered advection
** UV_SADVECTION   use to turn ON or OFF splines vertical advection
** UV_VIS2         use to turn ON or OFF harmonic horizontal mixing
** UV_VIS4         use to turn ON or OFF biharmonic horizontal mixing
** UV_SMAGORINSKY  use to turn ON or OFF Smagorinsky-like viscosity
** UV_DRAG_GRID    use if spatially varying bottom friction parameters
** UV_LOGDRAG      use to turn ON or OFF logarithmic bottom friction
** UV_LDRAG        use to turn ON or OFF linear bottom friction
** UV_QDRAG        use to turn ON or OFF quadratic bottom friction
** UV_WAVEDRAG     use to turn ON or OFF extra linear bottom wave drag
** SPLINES_VVISC   use if splines reconstruction of vertical viscosity
**
Tool Output
Search Results Tool Output
11 | 1 | Read | Ovr | Block | Sync | Rec | Caps
```

This list goes on
for many many
more pages

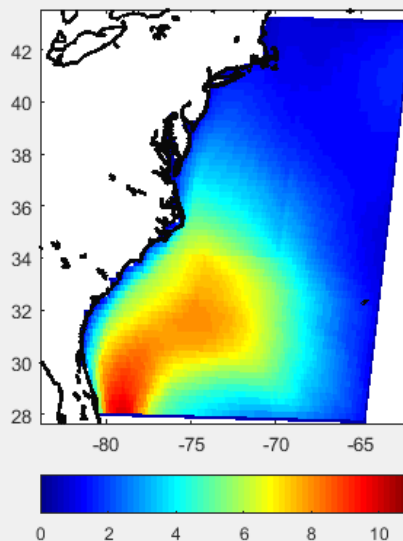
Projects/Sandy Application

WRF



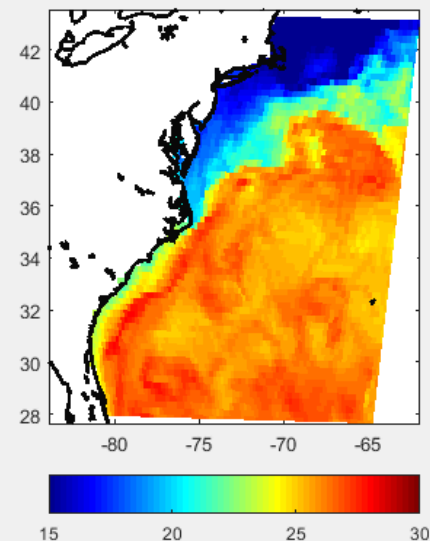
PSFC (mb)

SWAN



Hsig (m)

ROMS



SST (C)

Steps to create ROMS application

- 1) parent grid
- 2) masking
- 3) bathymetry
- 4) child grid
- 5) 3D: BC's (u,v,temp,salt), init, and climatology
- 6) 2D: BC's (ubar, vbar, zeta) = tides
- 7) Surface forcing (heat and momentum fluxes)
- 8) sandy.h and ocean_sandy.in
- 9) coawst.bash
- 10) run it

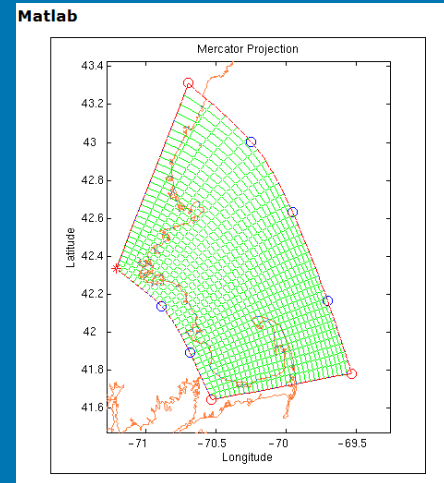
- Classroom tutorial will follow

[Projects/Sandy/create_sandy_application.m](#)

1) Grid generation tools

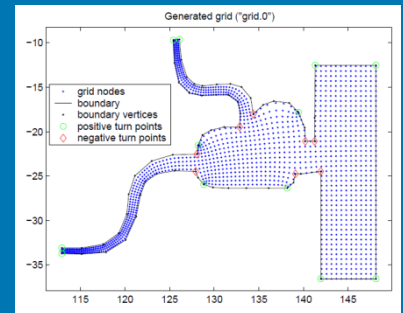
■ Seagrid - matlab

<http://woodshole.er.usgs.gov/operations/modeling/seagrid/>
(needs unsupported netcdf interface)

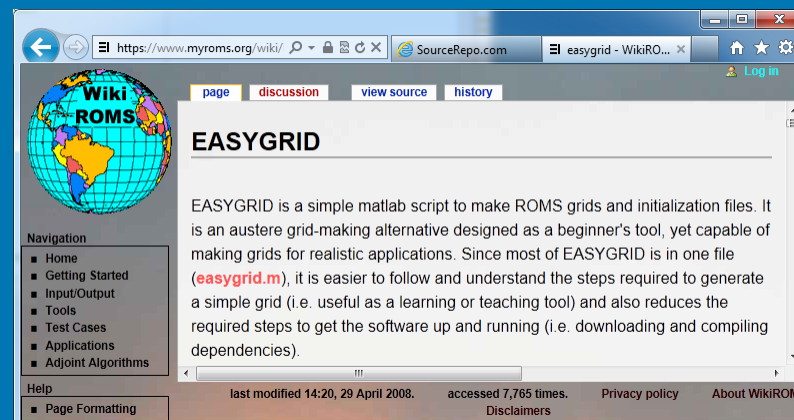


■ gridgen - command line

<http://code.google.com/p/gridgen-c/>



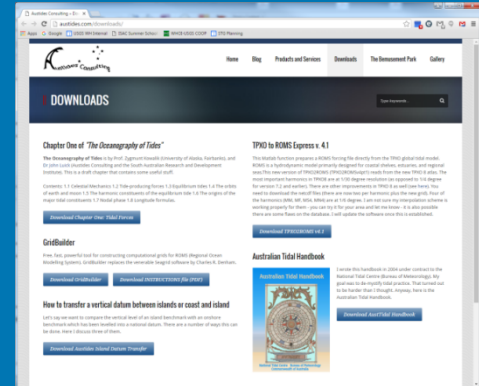
■ EASYGRID



1) Grid generation tools

- Austides

<http://austides.com/downloads/>



- COAWST/Tools/mfiles/mtools/wrf2roms _mw.m

`function wrf2roms_mw(theWRFFile, theROMSFile)`

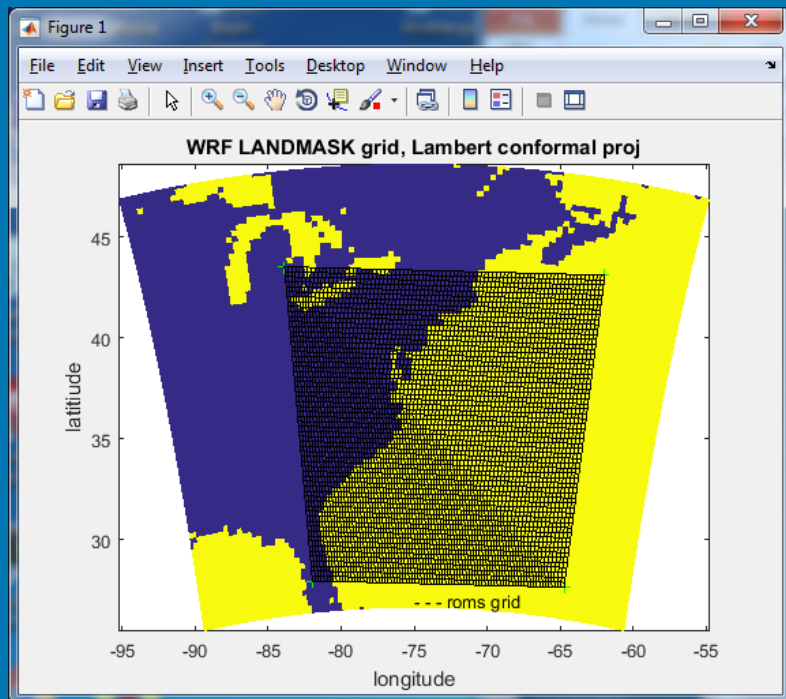
Generates a ROMS grid from a WRF grid.

- COAWST/Tools/mfiles/mtools/create_roms_xygrid.m

intended for simple rectilinear grids

- or any other method that you know

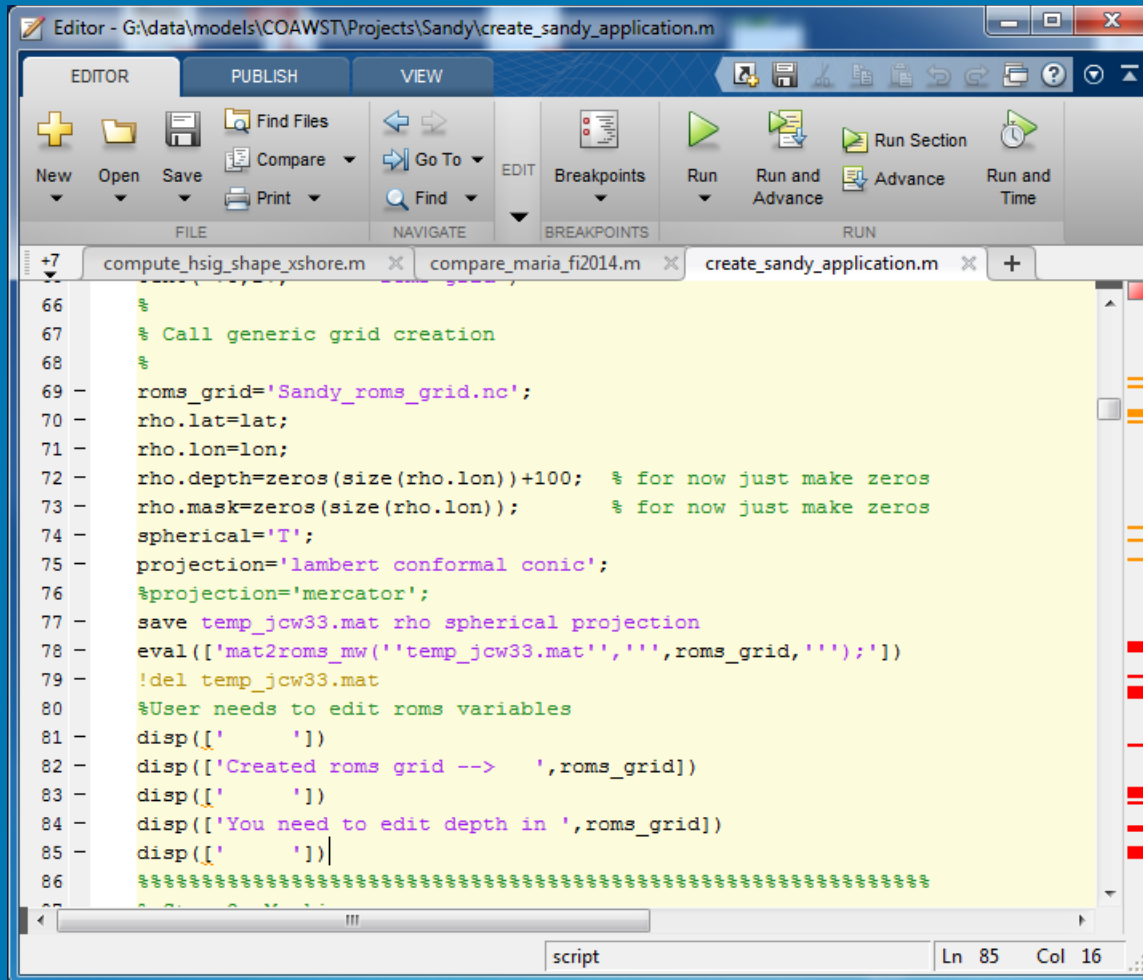
1) Grid generation tools – m file



select 4 corners

```
Editor - G:\data\models\COAWST\Projects\Sandy\create_sandy_application.m
EDITOR PUBLISH VIEW
New Open Save Find Files Find Go To Comment % fx Breakpoints Run Run and Advance Run Section Advance Run and Time
FILE NAVIGATE EDIT BREAKPOINTS RUN
+6 compute_hsig_shape_xshore.m compare_maria_fi2014.m create_sandy_application.m roms_master_climatology_sandy.m
25 % Step 1: ROMS grid
26 %
27 % As a first cut, you can use the wrf grid.
28 % Load the wrf grid to get coastline data
29 netcdf_load('wrfinput_d01')
30 figure
31 pcolorjw(XLONG,XLAT,double(1-LANDMASK))
32 hold on
33 title('WRF LANDMASK grid, Lambert conformal proj')
34 xlabel('longitude'); ylabel('latitude')
35 %
36 % pick 4 corners for roms parent grid
37 Istr=27; Iend=84;
38 Jstr=7; Jend=64;
39 plot(XLONG(Istr,Jstr),XLAT(Istr,Jstr),'g+')
40 plot(XLONG(Iend,Jstr),XLAT(Iend,Jstr),'g+')
41 plot(XLONG(Istr,Jend),XLAT(Istr,Jend),'g+')
42 plot(XLONG(Iend,Jend),XLAT(Iend,Jend),'g+')
43 %
44 % pick number of points in the grid
45 numx=84;
46 numy=64;
47 %
48 % Make a matrix of the lons and lats
49 % You should not need to change this.
50 % start in lower left and go clockwise
51 corner_lon=double([XLONG(Istr,Jend) XLONG(Iend,Jend) XLONG(Iend,Jstr) ]);
52 corner_lat=double([ XLAT(Istr,Jend) XLAT(Istr,Jend) XLAT(Iend,Jend) XLAT(Iend,Jstr) ]);
53 x=[1 numx; 1 numx]; y=[1 1; numy numy];
54 z=[corner_lon(1) corner_lon(2) corner_lon(4) corner_lon(3)];
55 F = TriScatteredInterp(x(:),y(:),z(:));
56 [X,Y]=meshgrid(1:numx,1:numy);
57 lon=F(X,Y).';
58 %
59 x=[1 numx; 1 numx]; y=[1 1; numy numy];
60 z=[corner_lat(1) corner_lat(2) corner_lat(4) corner_lat(3)];
61 F = TriScatteredInterp(x(:),y(:),z(:));
62 lat=F(X,Y).';
63 plot(lon,lat,'k-')
64 plot(lon',lat', 'k-')
65 text(-75,27,'- - - roms grid')
66 %
67 % Call generic grid creation
68 %
```

1) Grid generation tools

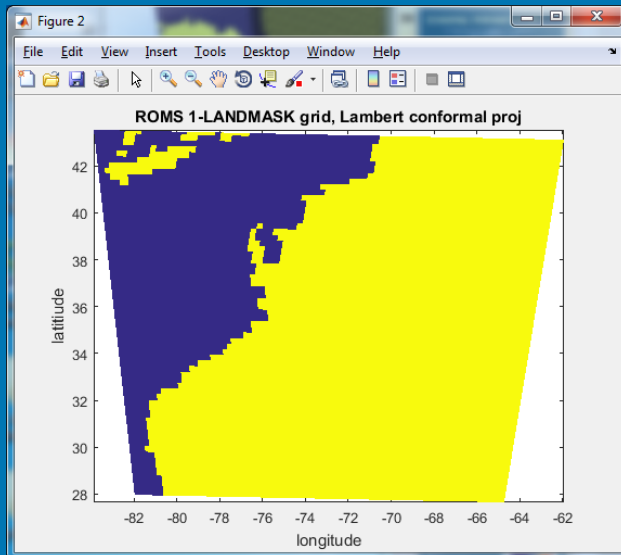


The screenshot shows a MATLAB editor window titled "Editor - G:\data\models\COAWST\Projects\Sandy\create_sandy_application.m". The window has a menu bar with "EDITOR", "PUBLISH", and "VIEW". Below the menu bar is a toolbar with icons for "New", "Open", "Save", "Find Files", "Compare", "Go To", "Find", "Breakpoints", "Run", "Run and Advance", "Run Section", "Advance", and "Run and Time". The main editing area shows a script with the following code:

```
66 %  
67 % Call generic grid creation  
68 %  
69 roms_grid='Sandy_roms_grid.nc';  
70 rho.lat=lat;  
71 rho.lon=lon;  
72 rho.depth=zeros(size(rho.lon))+100; % for now just make zeros  
73 rho.mask=zeros(size(rho.lon)); % for now just make zeros  
74 spherical='T';  
75 projection='lambert conformal conic';  
76 %projection='mercator';  
77 save temp_jcw33.mat rho spherical projection  
78 eval(['mat2roms_mw('temp_jcw33.mat','',roms_grid,'');'])  
79 !del temp_jcw33.mat  
80 %User needs to edit roms variables  
81 disp([''])  
82 disp(['Created roms grid --> ',roms_grid])  
83 disp([''])  
84 disp(['You need to edit depth in ',roms_grid])  
85 disp([''])  
86 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

The status bar at the bottom indicates "script" and "Ln 85 Col 16".

2) masking – first base it on WRF

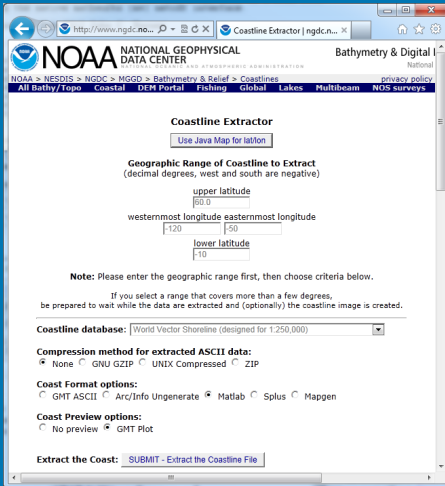


```
Editor - G:\data\models\COAWST\Projects\Sandy\create_sandy_application.m
EDITOR PUBLISH VIEW
New Open Save Compare Find Files Go To Comment % % % Breakpoints Run Run and Run and
Print Find Indent Breakpoints Run and Advance Advance Run and Time
FILE NAVIGATE EDIT BREAKPOINTS RUN
+6 plot_sfcrr_wave_refract.m compute_hsig_shape_xshore.m compare_maria_fi2014.m create_sandy_application.m +
87 % Step 2: Masking
88 %
89 % base this on the WRF mask.
90 F = TriScatteredInterp(double(XLONG(:)),double(XLAT(:)), ...
91 double(1-LANDMASK(:)), 'nearest');
92 roms_mask=F(lon,lat);
93 figure
94 pcolorjw(lon,lat,roms_mask)
95 title('ROMS 1-LANDMASK grid, Lambert conformal proj')
96 xlabel('longitude'); ylabel('latitude')
97 %
98 % compute mask on rho, u, v, and psi points.
99 water = double(roms_mask);
100 u_mask = water(1:end-1,:) & water(2:end,:);
101 v_mask = water(:,1:end-1) & water(:,2:end);
102 psi_mask= water(1:end-1,1:end-1) & water(1:end-1,2:end) & water(2:end,1:end-1) & water(2:end,2:end);
103 ncwrite('Sandy_roms_grid.nc','mask_rho',roms_mask);
104 ncwrite('Sandy_roms_grid.nc','mask_u',double(u_mask));
105 ncwrite('Sandy_roms_grid.nc','mask_v',double(v_mask));
106 ncwrite('Sandy_roms_grid.nc','mask_psi',double(psi_mask));
107 %
108 % use coastline tool to get coast:
109 % use GEODAS software http://www.ngdc.noaa.gov/mgg/geodas/geodas.html
110 % select lat bounds of 45 / 25 lon of -84 / -60
111 %
112 lon=coastline(:,1);
113 lat=coastline(:,2);
114 save coastline.mat lon lat
115 %
116 % To create better mask, use editmask with the
117 % grid file Sandy_roms_grid.nc and
118 % the coastline file coastline.mat
119 %
120 editmask
121 %
script Ln 99 Col 27
```

2) Masking – get coastline

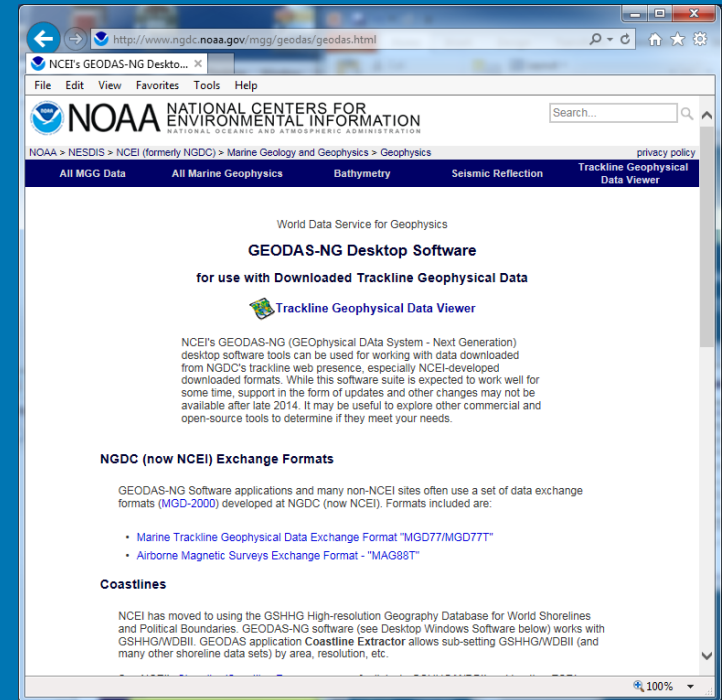
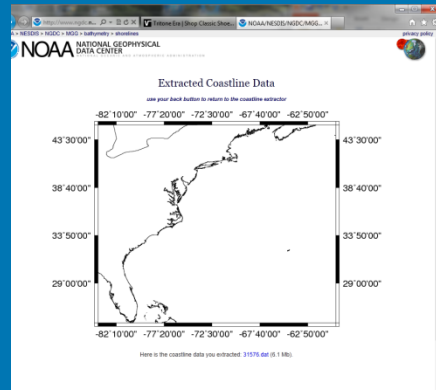
May also need a coastline, can obtain this here:

<http://www.ngdc.noaa.gov/mgg/coast/>



45
-84 -60
25

No longer active.



<http://www.ngdc.noaa.gov/mgg/geodas/geodas.html>

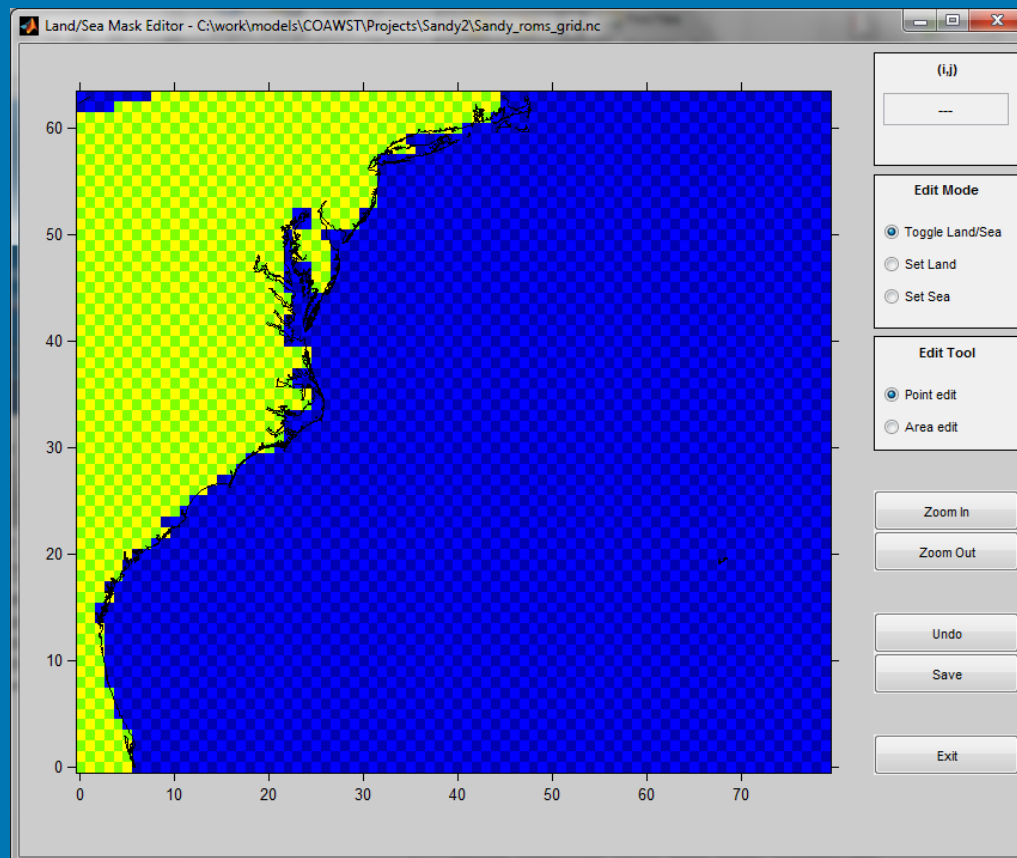


```
lon=coastline(:,1);  
lat=coastline(:,2);  
save coastline.mat lon lat
```

2) Masking

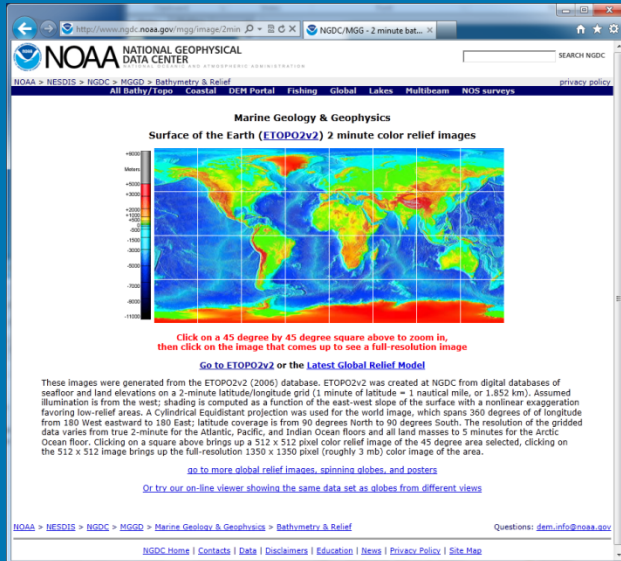
use COAWST/Tools/mfiles/mtools/editmask m file

(from Rutgers, but I changed it to use native matlab netcdf)
`editmask('USeast_grd.nc','coastline.mat')`

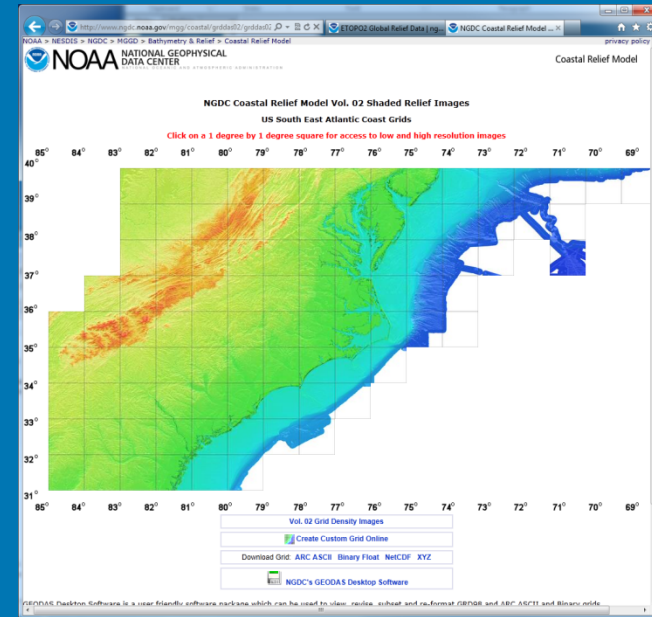


3) bathymetry

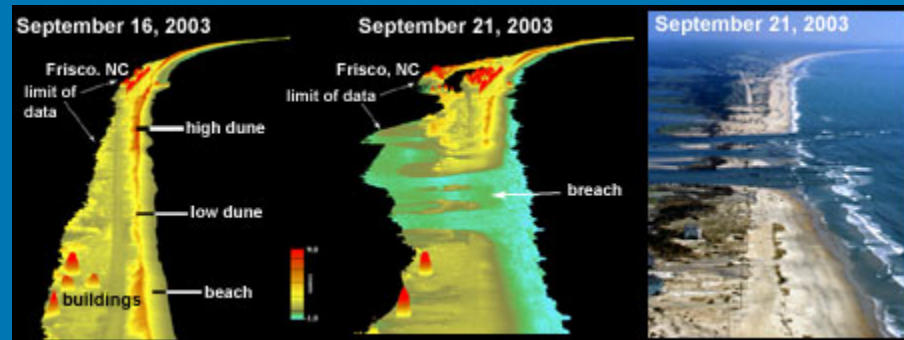
many sources



ETOPO2



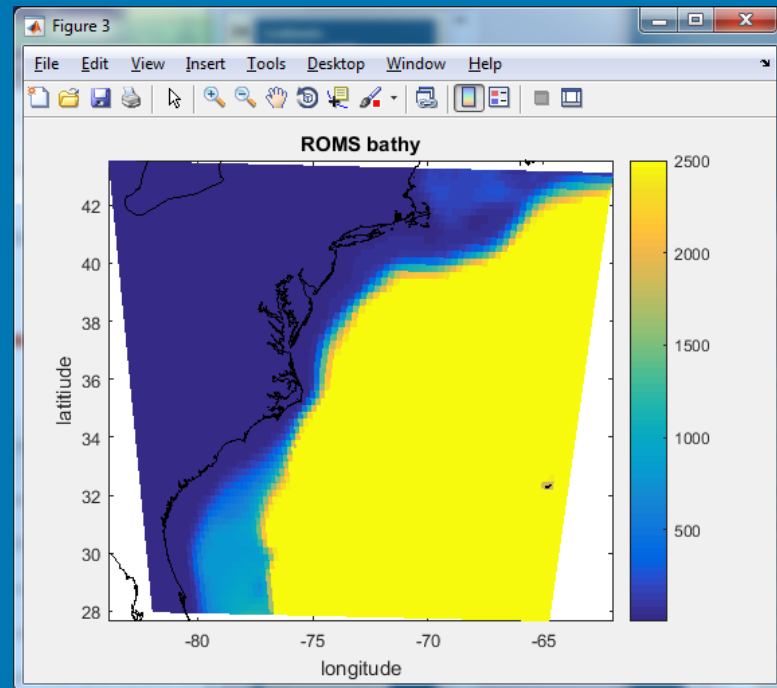
Coastal Relief Model



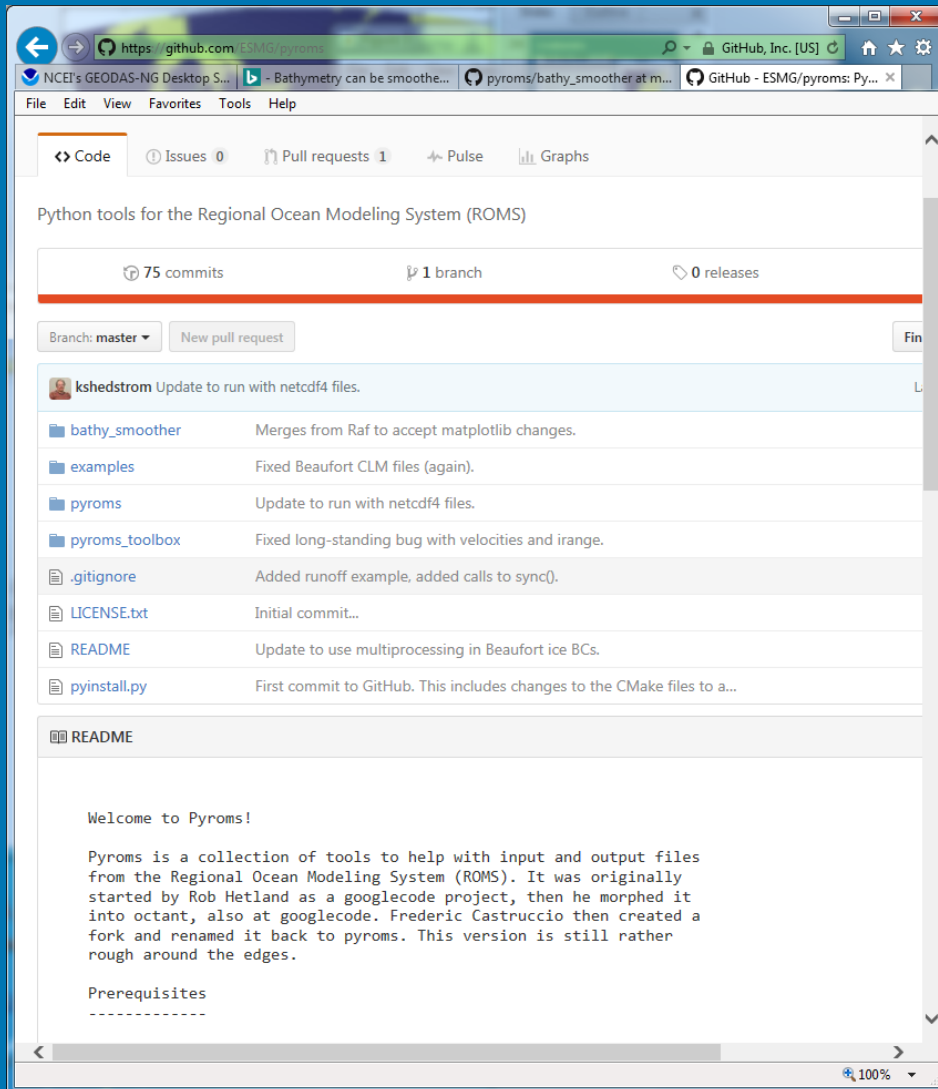
LIDAR

3) bathymetry

```
Editor - G:\data\models\COAWST\Projects\Sandy\create_sandy_application.m
EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Comment % Breakpoints Run Run and Advance Run Section Run and Time
FILE NAVIGATE EDIT BREAKPOINTS RUN
+6 plot_sfr_wave_refract.m x compute_hsig_shape_xshore.m x compare_maria_fi2014.m x create_sandy_application.m x
121 %
122 % Step 3: Bathymetry
123 %
124 % You can use a source like this:
125 % http://www.ngdc.noaa.gov/mgg/global/global.html
126 % I am using data from a local file
127 load USEast_bathy.mat
128 netcdf_load(roms_grid)
129 h=griddata(h_lon,h_lat,h_USEast,lon_rho,lat_rho);
130 h(isnan(h))=5;
131 %smooth h a little
132 h(2:end-1,2:end-1)=0.2*(h(1:end-2,2:end-1)+h(2:end-1,2:end-1)+h(3:end,2:end-1)+...
133 h(2:end-1,1:end-2)+h(2:end-1,3:end));
134 % Bathymetry can be smoothed using
135 % http://drobilica.irb.hr/~mathieu/Bathymetry/index.html
136 figure
137 pcolorjw(lon_rho,lat_rho,h)
138 hold on
139 load coastline.mat
140 plot(lon,lat,'k')
141 caxis([5 2500]); colorbar
142 title('ROMS bathy')
143 xlabel('longitude'); ylabel('latitude')
144 ncwrite(roms_grid,'h',h);
145 %
146 %
script Ln 131 Col 15
```



3) bathymetry



Bathy smoothing:

<https://github.com/ESMG/pyroms>

Grid Parameters

Beckman & Haidvogel number (1993)

$$r_{xo} = \max \left(\frac{\Delta h}{2\bar{h}} \right) = \max \left(\frac{|h_i - h_{i-1}|}{h_i + h_{i-1}} \right)$$

should be < 0.2 but can be fine up to ~ 0.4

determined only by smoothing

Haney number (1991)

$$r_{x1} = \max \left(\frac{z_{i,j,k} - z_{i-1,j,k} + z_{i,j,k-1} - z_{i-1,j,k-1}}{z_{i,j,k} + z_{i-1,j,k} - z_{i,j,k-1} - z_{i-1,j,k-1}} \right)$$

should be < 9 but can be fine up to ~ 16 in some cases

determined by smoothing AND vertical coordinate functions

If these numbers are too large, you will get large pressure gradient errors and Courant number violations and the model will typically blow up right away

4) Child grid

```

Editor - G:\data\models\COAWST\Projects\Sandy\create_sandy_application.m
EDITOR PUBLISH VIEW
+ New Open Save Find Files Compare Go To Insert fx Breakpoints Run Run and Advance Run Section Run and Time
FILE NAVIGATE EDIT BREAKPOINTS RUN
compute_hsig_shape_xshore.m compare_maria_fi2014.m create_sandy_application.m
146 %
147 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
148 % Step 4: roms child grid
149 %
150 % plot wrf d01 and d02 grids
151 netcdf_load('wrfinput_d01')
152 figure
153 pcolorjw(XLONG,XLAT,double(1-LANDMASK))
154 hold on
155 netcdf_load('wrfinput_d02')
156 pcolorjw(XLONG,XLAT,double(1-LANDMASK))
157 plot(XLONG(1,:),XLAT(1,:), 'r'); plot(XLONG(end,:),XLAT(end,:), 'r')
158 plot(XLONG(:,1),XLAT(:,1), 'r'); plot(XLONG(:,end),XLAT(:,end), 'r')
159 % plot roms parent grid
160 plot(lon_rho(1,:),lat_rho(1,:), 'k'); plot(lon_rho(end,:),lat_rho(end,:), 'k')
161 plot(lon_rho(:,1),lat_rho(:,1), 'k'); plot(lon_rho(:,end),lat_rho(:,end), 'k')
162 text(-75,29,'roms parent grid')
163 text(-77,27,'wrf parent grid')
164 text(-77.2,34,'wrf child grid')
165 title('LANDMASKS')
166 xlabel('longitude'); ylabel('latitude')
167 %
168 % Select child indices
169 Istr=22; Iend=60; Jstr=26; Jend=54;
170 %now make some plots and call the coarse-> fine
171 plot(lon_rho(Istr,Jstr),lat_rho(Istr,Jstr), 'm+')
172 plot(lon_rho(Istr,Jend),lat_rho(Istr,Jend), 'm+')
173 plot(lon_rho(Iend,Jstr),lat_rho(Iend,Jstr), 'm+')
174 plot(lon_rho(Iend,Jend),lat_rho(Iend,Jend), 'm+')
175 ref_ratio=3;
176 roms_child_grid='Sandy_roms_grid_ref3.nc';
177 F=coarse2fine('Sandy_roms_grid.nc','Sandy_roms_grid_ref3.nc', ...
178             ref_ratio,Istr,Iend,Jstr,Jend);
179 Gnames={'Sandy_roms_grid.nc','Sandy_roms_grid_ref3.nc'};
180 [S,G]=contact(Gnames,'Sandy_roms_contact.nc');
181 %
script Ln 177 Col 66

```

```

ref_ratio=3;
roms_child_grid='Sandy_roms_grid_ref3.nc';

```

```

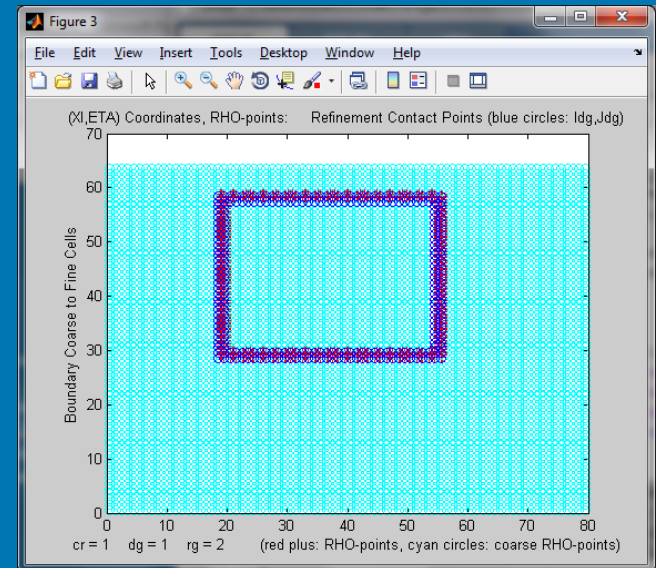
F=coarse2fine('Sandy_roms_grid.nc','Sandy_roms_grid_ref3.nc', ...
             ref_ratio,Istr,Iend,Jstr,Jend);

```

```

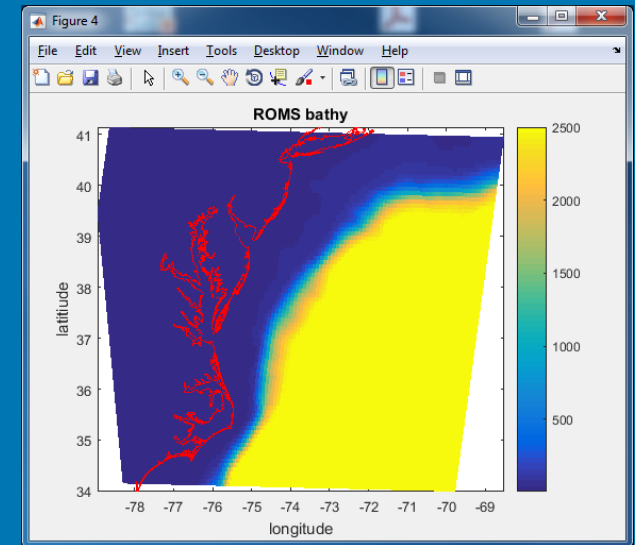
Gnames={'Sandy_roms_grid.nc','Sandy_roms_grid_ref3.nc'};
[S,G]=contact(Gnames,'Sandy_roms_contact.nc');

```

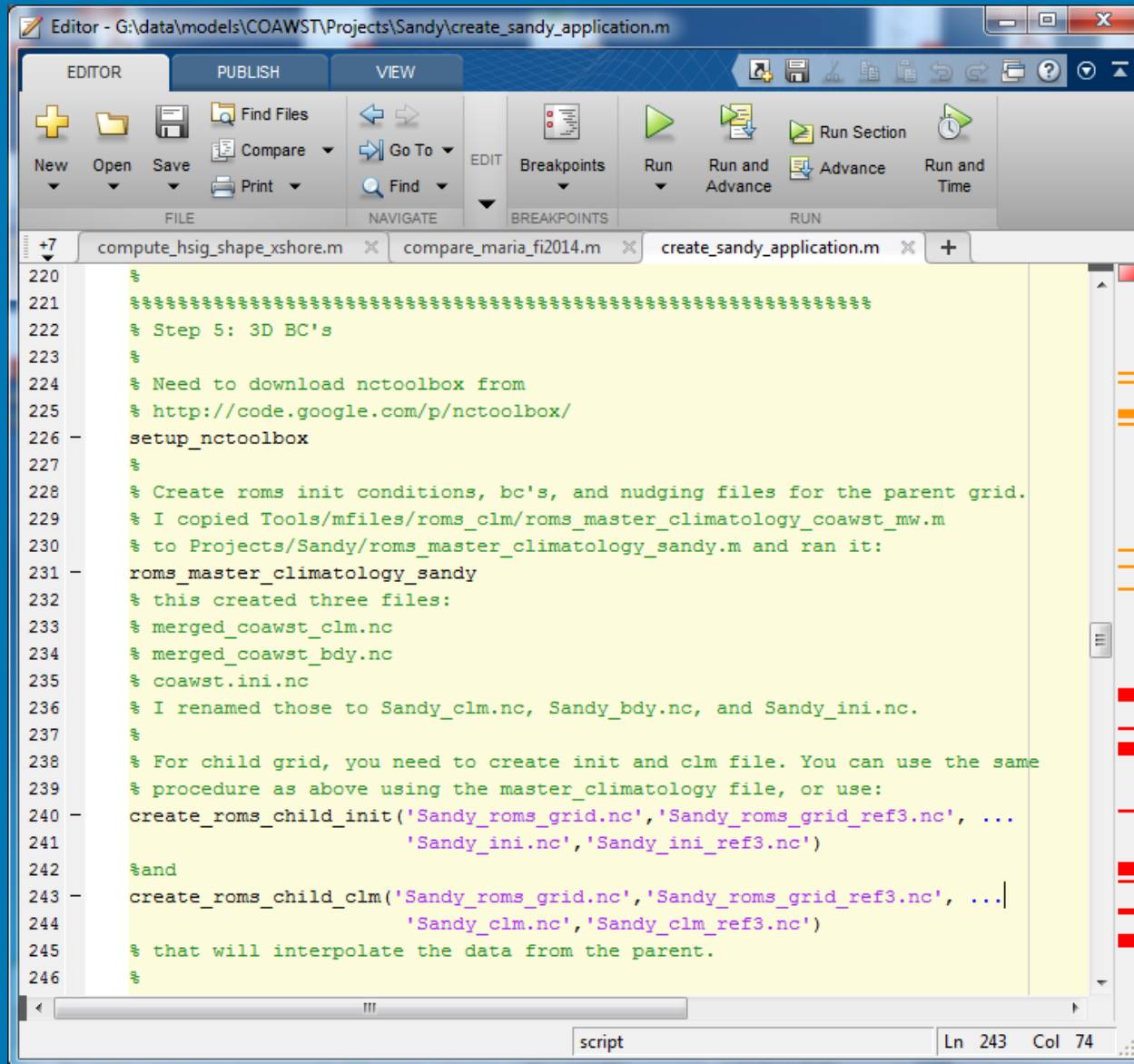


4) Child grid- bathy + masking

```
Editor - G:\data\models\COAWST\Projects\Sandy\create_sandy_application.m
EDITOR PUBLISH VIEW
+ New Open Save Find Files Compare Go To Comment % % Breakpoints Run Run and Advance Run Section Advance Run Time
FILE NAVIGATE EDIT BREAKPOINTS RUN
+6 plot_sfc_wave_refract.m x compute_hsig_shape_xshore.m x compare_maria_fi2014.m x create_sandy_application.m x
182 % Recompute h.
183 %
184 netcdf_load('Sandy_roms_grid_ref3.nc')
185 load USeast_bathy.mat
186 h=griddata(h_lon,h_lat,h_USeast,lon_rho,lat_rho);
187 h(isnan(h))=5;
188 h=h;
189 h(2:end-1,2:end-1)=0.2*(h(1:end-2,2:end-1)+h(2:end-1,2:end-1)+h(3:end,2:end-1)+ ...
190 h(2:end-1,1:end-2)+h(2:end-1,3:end));
191 figure
192 pcolorjw(lon_rho,lat_rho,h)
193 hold on
194 plot(lon,lat,'r')
195 caxis([5 2500]); colorbar
196 title('ROMS bathy')
197 xlabel('longitude'); ylabel('latitude')
198 ncwrite('Sandy_roms_grid_ref3.nc','h',h);
199 %
200 % recompute child mask based on WRF mask
201 %
202 netcdf_load('wrfinput_d02');
203 F = TriScatteredInterp(double(XLONG(:)),double(XLAT(:)), ...
204 double(1-LANDMASK(:)), 'nearest');
205 roms_mask=F(lon_rho,lat_rho);
206 figure
207 pcolorjw(lon_rho,lat_rho,roms_mask)
208 title('ROMS child mask')
209 xlabel('longitude'); ylabel('latitude')
210 hold on
211 plot(lon,lat,'r')
212 water = double(roms_mask);
213 u_mask = water(1:end-1,:) & water(2:end,:);
214 v_mask= water(:,1:end-1) & water(:,2:end);
215 psi_mask= water(1:end-1,1:end-1) & water(1:end-1,2:end) & water(2:end,1:end-1) & water(2:end,2:end);
216 ncwrite('Sandy_roms_grid_ref3.nc','mask_rho',double(roms_mask));
217 ncwrite('Sandy_roms_grid_ref3.nc','mask_u',double(u_mask));
218 ncwrite('Sandy_roms_grid_ref3.nc','mask_v',double(v_mask));
219 ncwrite('Sandy_roms_grid_ref3.nc','mask_psi',double(psi_mask));
220 %
script Ln 201 Col 1
```



5) 3D BC's (u,v,temp,salt), init, and clima.



```
Editor - G:\data\models\COAWST\Projects\Sandy\create_sandy_application.m

EDITOR PUBLISH VIEW
+ New Open Save Find Files Compare Go To Find Breakpoints Run Run and Advance Run Section Run and Time
FILE NAVIGATE BREAKPOINTS RUN

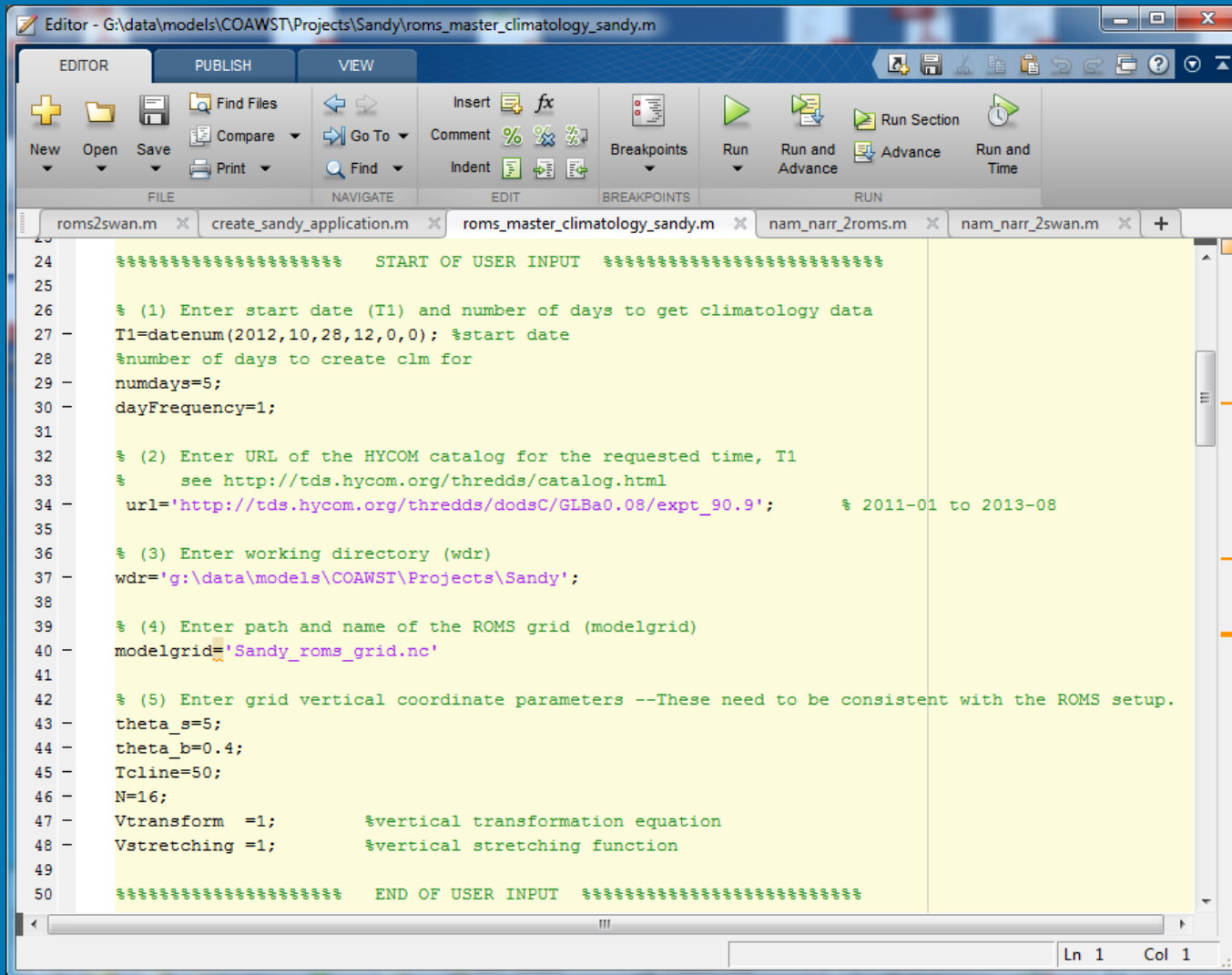
+7 compute_hsig_shape_xshore.m compare_maria_fi2014.m create_sandy_application.m +
220 %
221 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
222 % Step 5: 3D BC's
223 %
224 % Need to download nctoolbox from
225 % http://code.google.com/p/nctoolbox/
226 - setup_nctoolbox
227 %
228 % Create roms init conditions, bc's, and nudging files for the parent grid.
229 % I copied Tools/mfiles/roms_clm/roms_master_climatology_coawst_mw.m
230 % to Projects/Sandy/roms_master_climatology_sandy.m and ran it:
231 - roms_master_climatology_sandy
232 % this created three files:
233 % merged_coawst_clm.nc
234 % merged_coawst_bdy.nc
235 % coawst.ini.nc
236 % I renamed those to Sandy_clm.nc, Sandy_bdy.nc, and Sandy_ini.nc.
237 %
238 % For child grid, you need to create init and clm file. You can use the same
239 % procedure as above using the master_climatology file, or use:
240 - create_roms_child_init('Sandy_roms_grid.nc','Sandy_roms_grid_ref3.nc', ...
241 % 'Sandy_ini.nc','Sandy_ini_ref3.nc')
242 %and
243 - create_roms_child_clm('Sandy_roms_grid.nc','Sandy_roms_grid_ref3.nc', ...
244 % 'Sandy_clm.nc','Sandy_clm_ref3.nc')
245 % that will interpolate the data from the parent.
246 %
```

Needs nctoolbox to
access data via
thredds server:

<https://github.com/nctoolbox/nctoolbox>

5) 3D BC's (u,v,temp,salt), init, and clima.

Projects/Sandy/roms_master_climatology_sandy.m



```
24 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% START OF USER INPUT %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
25
26 % (1) Enter start date (T1) and number of days to get climatology data
27 T1=datetime(2012,10,28,12,0,0); %start date
28 %number of days to create clm for
29 numdays=5;
30 dayFrequency=1;
31
32 % (2) Enter URL of the HYCOM catalog for the requested time, T1
33 % see http://tds.hycom.org/thredds/catalog.html
34 url='http://tds.hycom.org/thredds/dodsC/GLBa0.08/expt_90.9'; % 2011-01 to 2013-08
35
36 % (3) Enter working directory (wdr)
37 wdr='g:\data\models\COAWST\Projects\Sandy';
38
39 % (4) Enter path and name of the ROMS grid (modelgrid)
40 modelgrid='Sandy_roms_grid.nc'
41
42 % (5) Enter grid vertical coordinate parameters --These need to be consistent with the ROMS setup.
43 theta_s=5;
44 theta_b=0.4;
45 Tcline=50;
46 N=16;
47 Vtransform =1; %vertical transformation equation
48 Vstretching =1; %vertical stretching function
49
50 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% END OF USER INPUT %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

This works for multiple days.

5) 3D BC's (u,v,temp,salt), init, and clima.

% I copied Tools/mfiles/roms_clm/roms_master_climatology_coawst_mw.m
% to Projects/Sandy/roms_master_climatology_sandy.m and ran it:

roms_master_climatology_sandy

% this created three files:

% merged_coawst_clm.nc

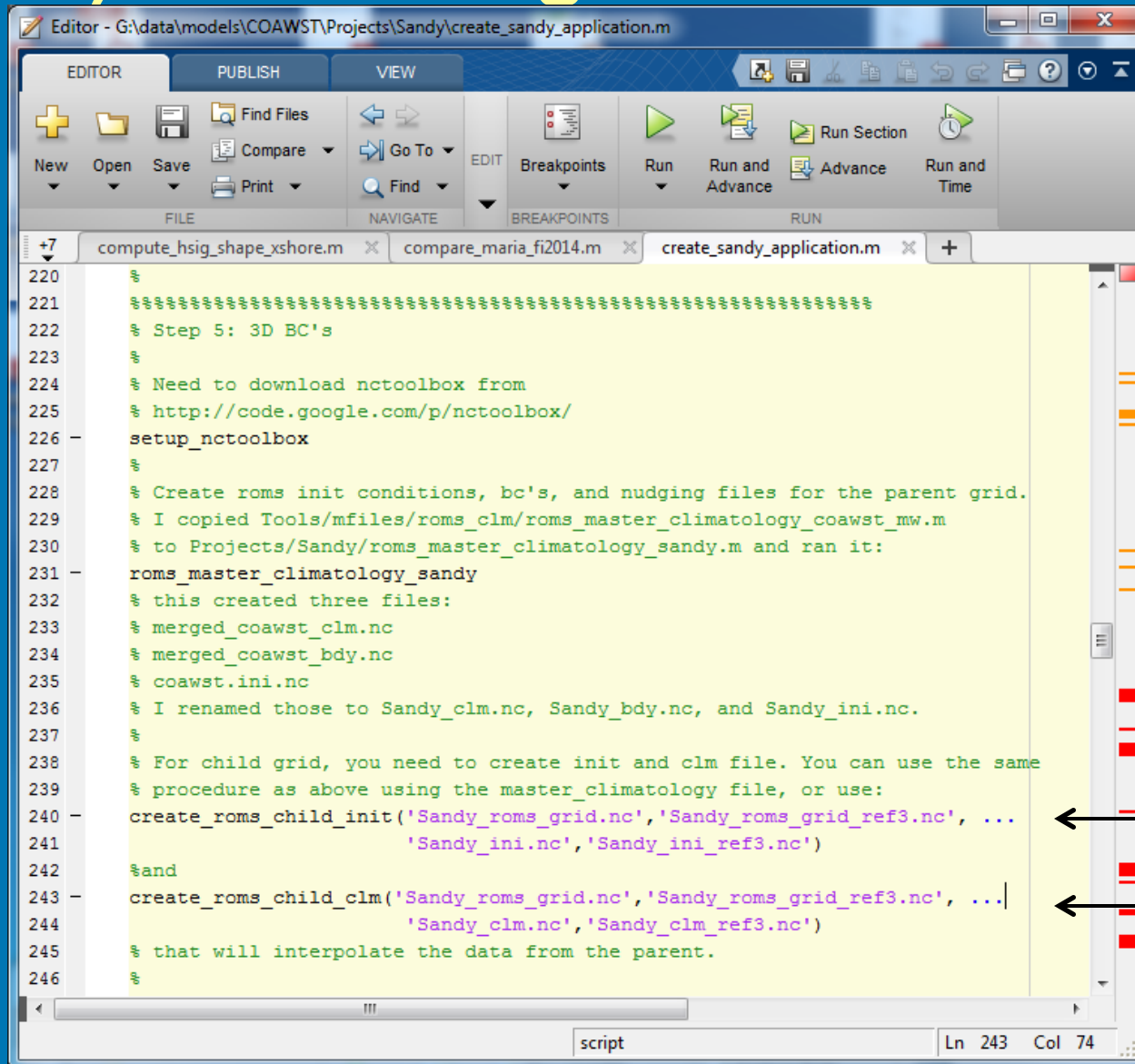
% merged_coawst_bdy.nc

% coawst.ini.nc

% I renamed those to Sandy_clm.nc, Sandy_bdy.nc, and Sandy_ini.nc.

%

5) For child grid need init and clim.



```
Editor - G:\data\models\COAWST\Projects\Sandy\create_sandy_application.m

EDITOR PUBLISH VIEW
+ New Open Save Find Files Compare Go To Find EDIT Breakpoints Run Run and Advance Run Section Advance Run and Time
FILE NAVIGATE BREAKPOINTS RUN

+7 compute_hsig_shape_xshore.m compare_maria_fi2014.m create_sandy_application.m +
220 %
221 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
222 % Step 5: 3D BC's
223 %
224 % Need to download nctoolbox from
225 % http://code.google.com/p/nctoolbox/
226 - setup_nctoolbox
227 %
228 % Create roms init conditions, bc's, and nudging files for the parent grid.
229 % I copied Tools/mfiles/roms_clm/roms_master_climatology_coawst_mw.m
230 % to Projects/Sandy/roms_master_climatology_sandy.m and ran it:
231 - roms_master_climatology_sandy
232 % this created three files:
233 % merged_coawst_clm.nc
234 % merged_coawst_bdy.nc
235 % coawst.ini.nc
236 % I renamed those to Sandy_clm.nc, Sandy_bdy.nc, and Sandy_ini.nc.
237 %
238 % For child grid, you need to create init and clim file. You can use the same
239 % procedure as above using the master_climatology file, or use:
240 - create_roms_child_init('Sandy_roms_grid.nc','Sandy_roms_grid_ref3.nc', ...
241 % 'Sandy_ini.nc','Sandy_ini_ref3.nc')
242 %and
243 - create_roms_child_clm('Sandy_roms_grid.nc','Sandy_roms_grid_ref3.nc', ...
244 % 'Sandy_clm.nc','Sandy_clm_ref3.nc')
245 % that will interpolate the data from the parent.
246 %
```

script Ln 243 Col 74

6) 2D: BC's (ubar, vbar, zeta) = tides



The screenshot shows a web browser window displaying the WikiROMS website. The page title is "Description of Tidal Forcing in ROMS". The URL in the address bar is https://www.myroms.org/wiki/index.php/Tidal_Forcing. The page has a navigation sidebar on the left with links like Home, Getting Started, Input/Output, Tools, Test Cases, Applications, and Adjoint Algorithms. The main content area includes a "Contents" table of contents with links to "1 Tidal Forcing Files in ROMS", "2 OSU Tidal Prediction Software Example (Matlab)", "3 ADCIRC Tidal Database Example (Matlab)", and "4 Comments on the 18.6 year tides". Below this is the section "Tidal Forcing Files in ROMS", which contains a paragraph explaining the process and a numbered list of six steps: 1. Download the desired tidal constituent database and associated software. 2. Interpolate the tidal constituent database to the desired ROMS grid. 3. Verify all open boundary grid cells contain valid data. 4. The phase of the u/v/zeta components of the tidal constituents should be shifted to the appropriate reference time (t=to). 5. Convert the amplitude/phase information to tidal ellipse parameters, if necessary. 6. Export tidal ellipse parameters to a ROMS netcdf forcing file. The page also includes a warning about Matlab compatibility and a section titled "OSU Tidal Prediction Software Example (Matlab)".

WikiROMS

Navigation

- Home
- Getting Started
- Input/Output
- Tools
- Test Cases
- Applications
- Adjoint Algorithms

Help

- Page Formatting
- FAQ

Search

Go Search

Toolbox

- What links here
- Related changes
- Special pages
- Printable version
- Permanent link

Description of Tidal Forcing in ROMS

Contents [hide]

- 1 Tidal Forcing Files in ROMS
- 2 OSU Tidal Prediction Software Example (Matlab)
- 3 ADCIRC Tidal Database Example (Matlab)
- 4 Comments on the 18.6 year tides

Tidal Forcing Files in ROMS

Once the appropriate [CPP options](#) have been set (e.g. [SSH_TIDES](#), [UV_TIDES](#), [RAMP_TIDES](#)), a netcdf file of tidal constituents must be generated.

- Download the desired tidal constituent database and associated software.** There are many tidal constituent databases available for download and the choice of databases depends on the desired constituents and location of the ROMS grid. Two possibilities which have broad geographic range and generally the dominant constituents are the [OSU Tidal Data Inversion Software](#) (OTIS) and the [ADCIRC tidal database](#). Remember to download the associated software with each data base as there are typically routines which facilitate the extraction and interpolation of the database to the ROMS grid.
- Interpolate the tidal constituent database to the desired ROMS grid.** See the above comment. While it is possible to write code (e.g. MATLAB, FORTRAN) to perform this task, it is typically easier to use the provided packages.
- Verify all open boundary grid cells contain valid data.** During the interpolation process, depending on the land mask for the ROMS grid and the tidal database grid, it is possible to have grid cells near land points which the ROMS grid may define as water, but the tidal grid defines as land. Should this occur a 180 degree phase shift along the open boundary near land points is possible. As ROMS is tidally forced on the open boundary, this could be problematic.
- The phase of the u/v/zeta components of the tidal constituents should be shifted to the appropriate reference time (t=to).** It is typical for tidal constituent databases to be stored with the phase shifted by the equilibrium tidal argument. Consequently the reference time for the tide is not the desired time, as set in [ocean.in](#) using the variable [TIDE_START](#). In addition, nodal corrections need to be made due to long period tides. The equilibrium argument and nodal corrections can be calculated using the tidal database software or Rich Pawlowicz's [T_TIDE](#) MATLAB package. Also, see below for more thoughts on the 18.6 year business.
- Convert the amplitude/phase information to tidal ellipse parameters, if necessary.** ROMS requires tidal information to be stored as ellipse parameters for use. One tidal ellipse package is [ap2ep.m](#) from Zhigang Xu. MATLAB tidal ellipse code is available.
- Export tidal ellipse parameters to a ROMS netcdf forcing file.**

Two examples of this process are described below.

⚠ Newer Matlab versions are incompatible with older versions of `t_tide` and will cause errors. In particular, the "finite" function has been replaced with "isfinite". Rich Pawlowicz has provided an updated version of `t_tide` on his website http://www.eos.ubc.ca/~rich/#T_Tide. I have also provided a link here [1].

OSU Tidal Prediction Software Example (Matlab)

The processing in this example was carried out in MATLAB using routines found in <http://marine.rutgers.edu/~hunter/roms/tides/otps/>. It has been

last modified 14:48, 1 August 2011. accessed 12,558 times. Privacy policy About WikiROMS

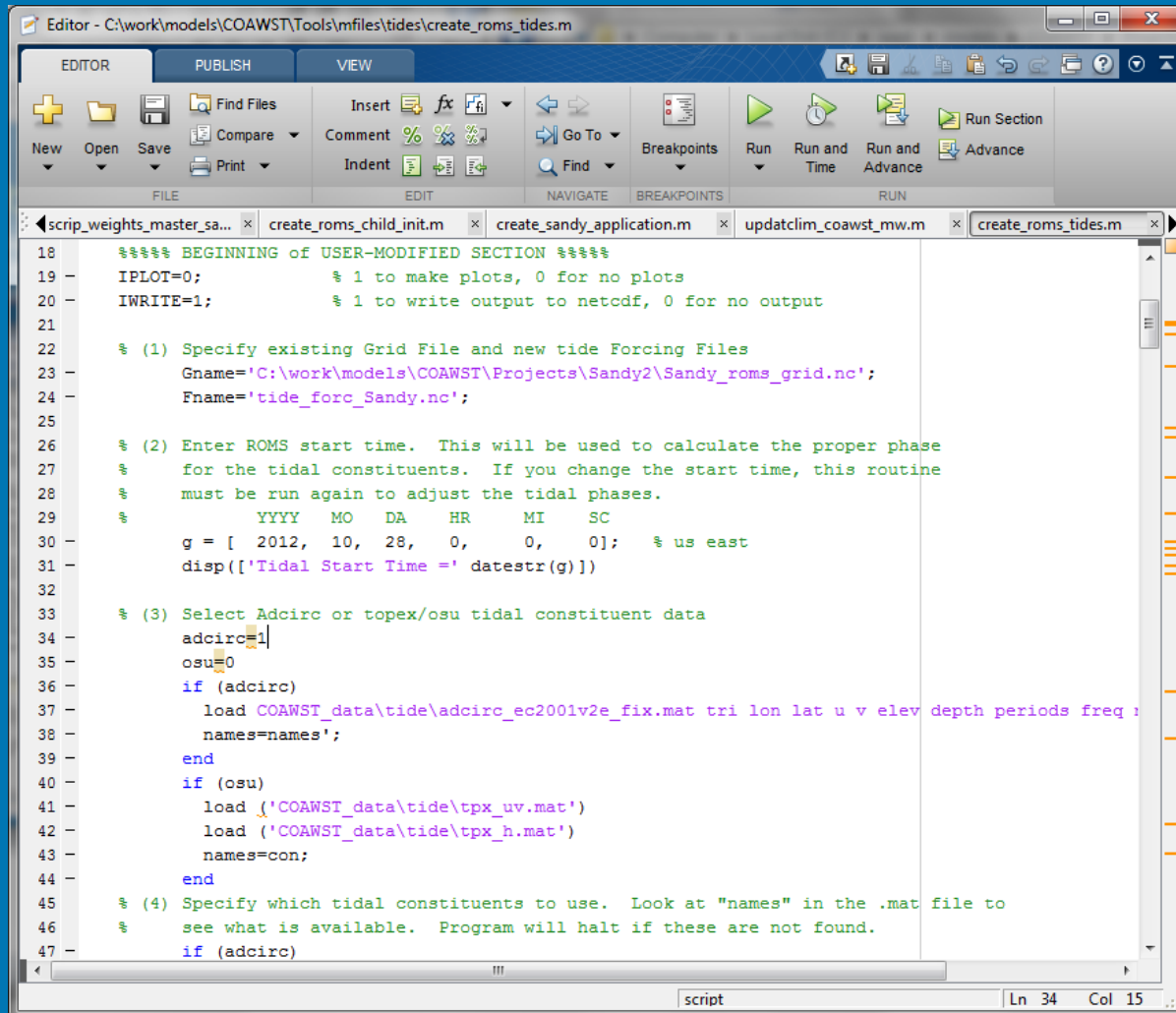
Powered By MediaWiki

6) 2D: BC's (ub_{bar}, v_{bar}, zeta) = tides

1) Get the tidal data at

svn checkout <https://coawstmodel.sourcerepo.com/coawstmodel/data> .

2) edit Tools/mfiles/tides/create_roms_tides



```
Editor - C:\work\models\COAWST\Tools\mfiles\tides\create_roms_tides.m

EDITOR PUBLISH VIEW
New Open Save Find Files Compare Print Insert Comment Indent Go To Breakpoints Run Run and Time Run and Advance Run Section Advance

FILE EDIT NAVIGATE BREAKPOINTS RUN

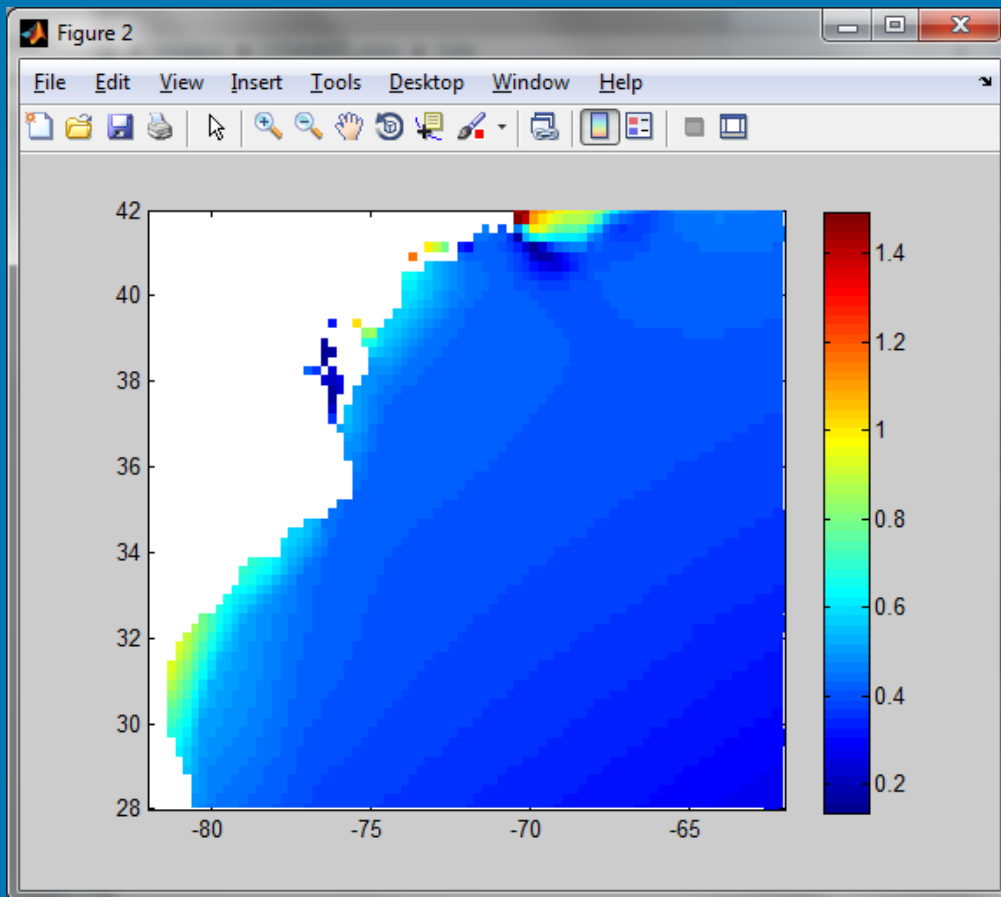
scrip_weights_master_sa... create_roms_child_init.m create_sandy_application.m updatclim_coawst_mw.m create_roms_tides.m

18 %%%% BEGINNING of USER-MODIFIED SECTION %%%%
19 IXPLOT=0; % 1 to make plots, 0 for no plots
20 IWRITE=1; % 1 to write output to netcdf, 0 for no output
21
22 % (1) Specify existing Grid File and new tide Forcing Files
23 Gname='C:\work\models\COAWST\Projects\Sandy2\Sandy_roms_grid.nc';
24 Fname='tide_forc_Sandy.nc';
25
26 % (2) Enter ROMS start time. This will be used to calculate the proper phase
27 % for the tidal constituents. If you change the start time, this routine
28 % must be run again to adjust the tidal phases.
29 % YYYY MO DA HR MI SC
30 g = [ 2012, 10, 28, 0, 0, 0]; % us east
31 disp(['Tidal Start Time = ' datestr(g)])
32
33 % (3) Select Adcirc or topex/osu tidal constituent data
34 adcirc=1;
35 osu=0;
36 if (adcirc)
37 load COAWST_data\tide\adcirc_ec2001v2e_fix.mat tri lon lat u v elev depth periods freq ;
38 names=names';
39 end
40 if (osu)
41 load ('COAWST_data\tide\tpx_uv.mat')
42 load ('COAWST_data\tide\tpx_h.mat')
43 names=con;
44 end
45 % (4) Specify which tidal constituents to use. Look at "names" in the .mat file to
46 % see what is available. Program will halt if these are not found.
47 if (adcirc)
```

script Ln 34 Col 15

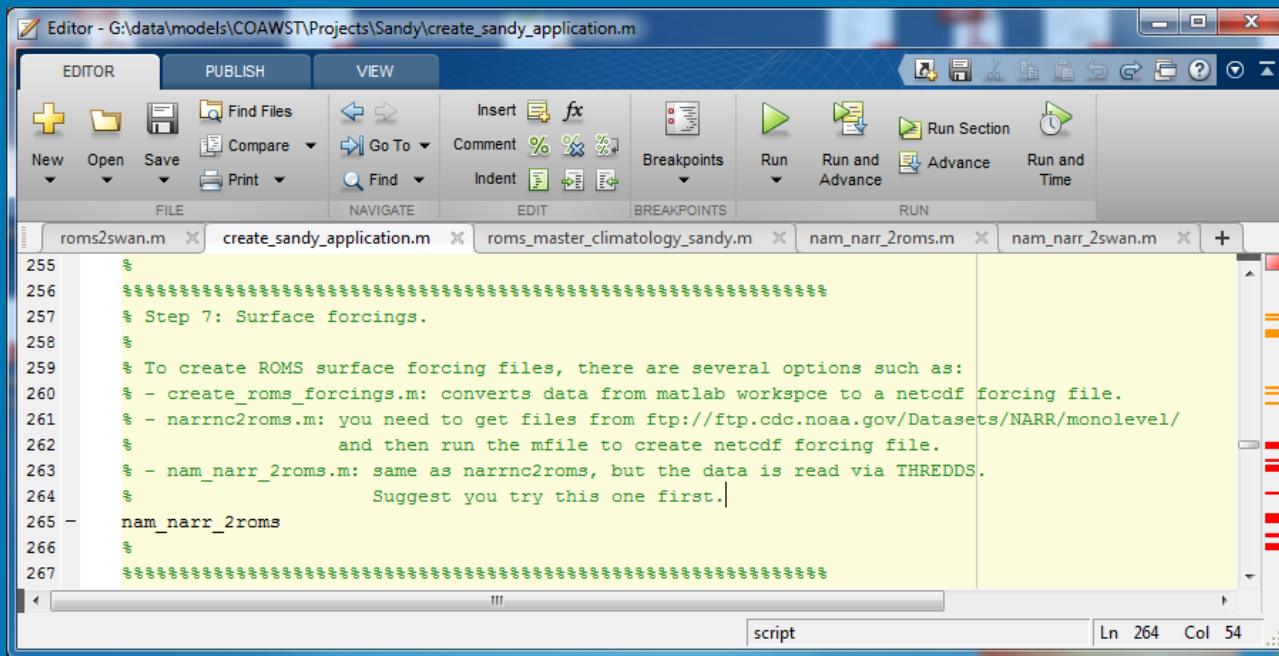
6) 2D: BC's (ubar, vbar, zeta) = tides

```
netcdf_load('Sandy_roms_grid.nc')  
pcolorjw(lon_rho,lat_rho,squeeze(tide_Eamp(:,:,1)))
```



M2 tidal amplitude

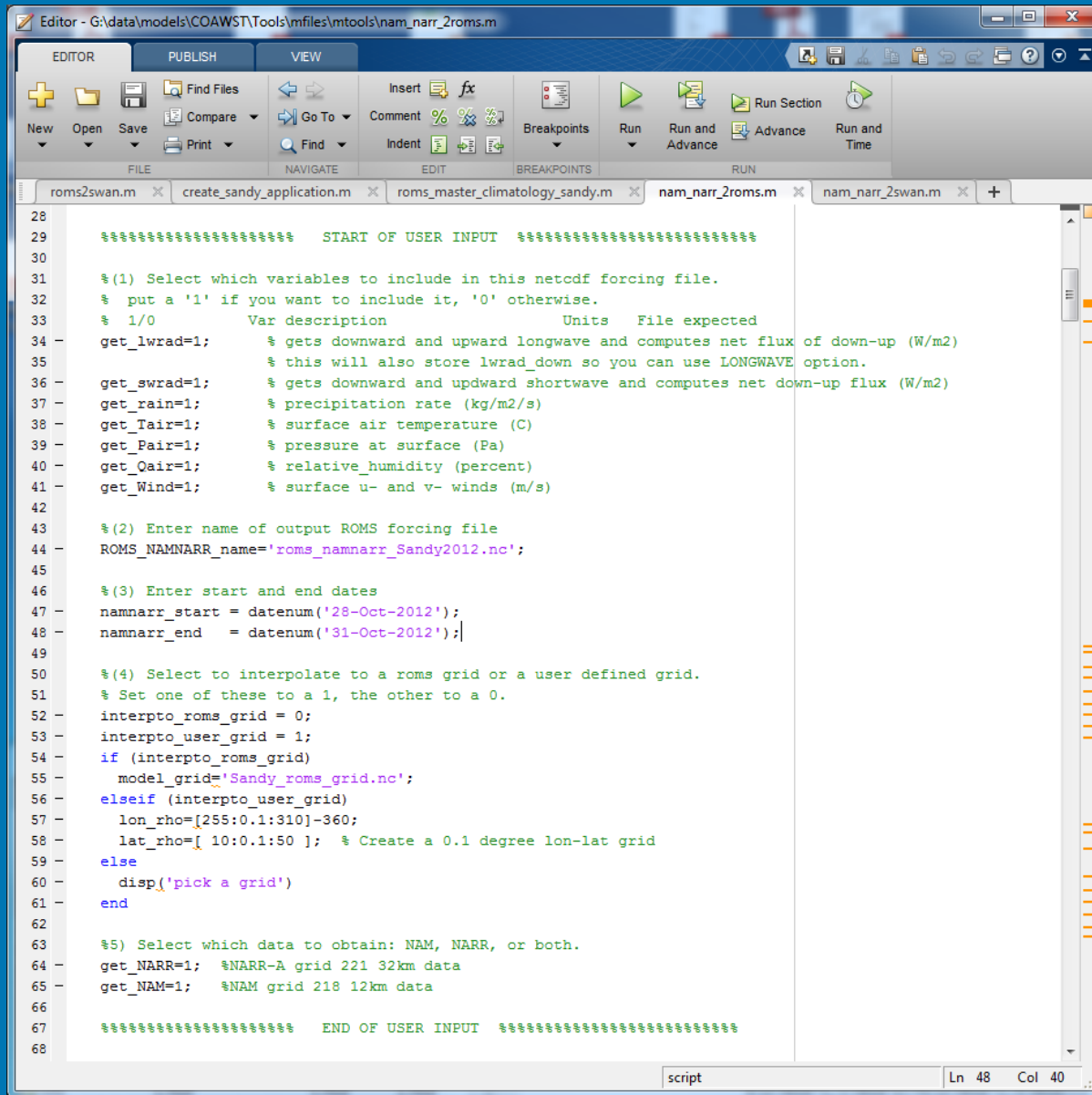
7) Surface forcings



```
255 %  
256 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
257 % Step 7: Surface forcings.  
258 %  
259 % To create ROMS surface forcing files, there are several options such as:  
260 % - create_roms_forcings.m: converts data from matlab workspace to a netcdf forcing file.  
261 % - narrnc2roms.m: you need to get files from ftp://ftp.cdc.noaa.gov/Datasets/NARR/monolevel/  
262 %   and then run the mfile to create netcdf forcing file.  
263 % - nam_narr_2roms.m: same as narrnc2roms, but the data is read via THREDDS.  
264 %   Suggest you try this one first.  
265 nam_narr_2roms  
266 %  
267 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Many options. We will use `nam_narr_2roms.m`

7) Surface forcings - nam_narr_2roms.m

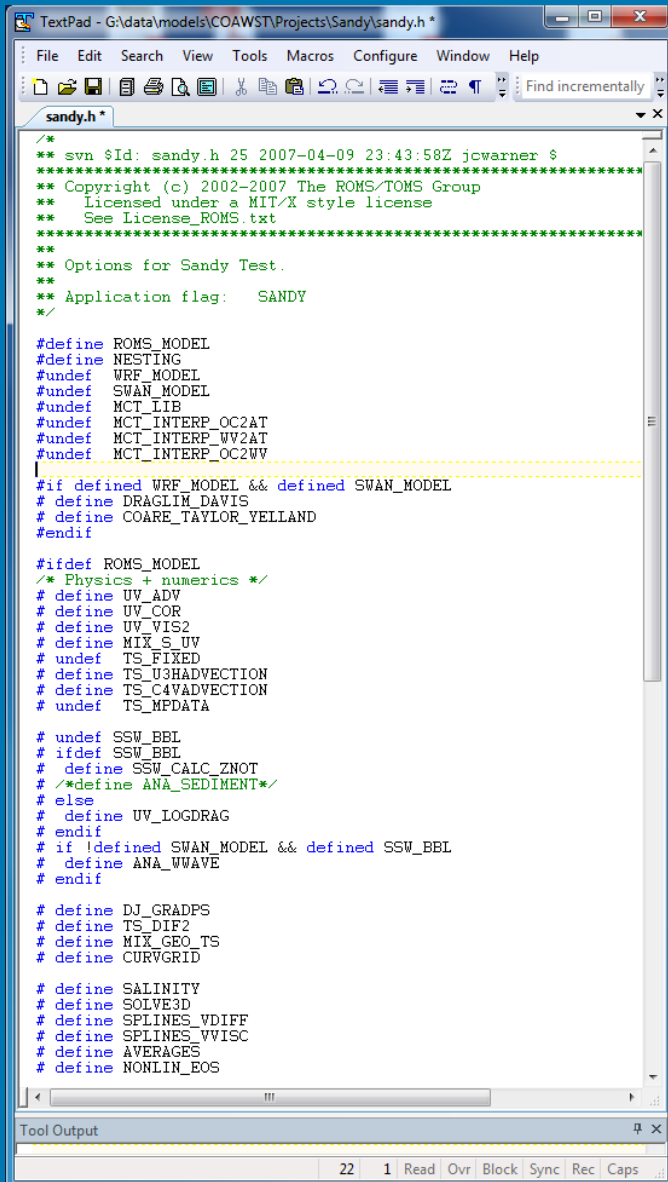


```
28
29 ***** START OF USER INPUT *****
30
31 % (1) Select which variables to include in this netcdf forcing file.
32 % put a '1' if you want to include it, '0' otherwise.
33 % 1/0      Var description      Units      File expected
34 get_lwrad=1; % gets downward and upward longwave and computes net flux of down-up (W/m2)
35 % this will also store lwrad_down so you can use LONGWAVE option.
36 get_swrad=1; % gets downward and upward shortwave and computes net down-up flux (W/m2)
37 get_rain=1;  % precipitation rate (kg/m2/s)
38 get_Tair=1;  % surface air temperature (C)
39 get_Pair=1;  % pressure at surface (Pa)
40 get_Qair=1;  % relative humidity (percent)
41 get_Wind=1;  % surface u- and v- winds (m/s)
42
43 % (2) Enter name of output ROMS forcing file
44 ROMS_NAMNARR_name='roms_namnarr_Sandy2012.nc';
45
46 % (3) Enter start and end dates
47 namnarr_start = datenum('28-Oct-2012');
48 namnarr_end   = datenum('31-Oct-2012');
49
50 % (4) Select to interpolate to a roms grid or a user defined grid.
51 % Set one of these to a 1, the other to a 0.
52 interpto_roms_grid = 0;
53 interpto_user_grid = 1;
54 if (interpto_roms_grid)
55     model_grid='Sandy_roms_grid.nc';
56 elseif (interpto_user_grid)
57     lon_rho=[255:0.1:310]-360;
58     lat_rho=[ 10:0.1:50 ]; % Create a 0.1 degree lon-lat grid
59 else
60     disp('pick a grid')
61 end
62
63 % (5) Select which data to obtain: NAM, NARR, or both.
64 get_NARR=1; %NARR-A grid 221 32km data
65 get_NAM=1;  %NAM grid 218 12km data
66
67 ***** END OF USER INPUT *****
68
```

COAWST/Tools/mfiles/mtools/
nam_narr_2roms
creates netcdf forcing files for
ROMS.

Many others tools available.

8) sandy.h



```
TextPad - G:\data\models\COAWST\Projects\Sandy\sandy.h *
File Edit Search View Tools Macros Configure Window Help
Find incrementally

sandy.h
/*
** svn $Id: sandy.h 25 2007-04-09 23:43:58Z jowarner $
** Copyright (c) 2002-2007 The ROMS/TOMS Group
** Licensed under a MIT/X style license
** See License_ROMS.txt
**
** Options for Sandy Test.
** Application flag: SANDY
**
#define ROMS_MODEL
#define NESTING
#undef WRF_MODEL
#undef SWAN_MODEL
#undef MCT_LIB
#undef MCT_INTERP_OC2AT
#undef MCT_INTERP_WV2AT
#undef MCT_INTERP_OC2WV

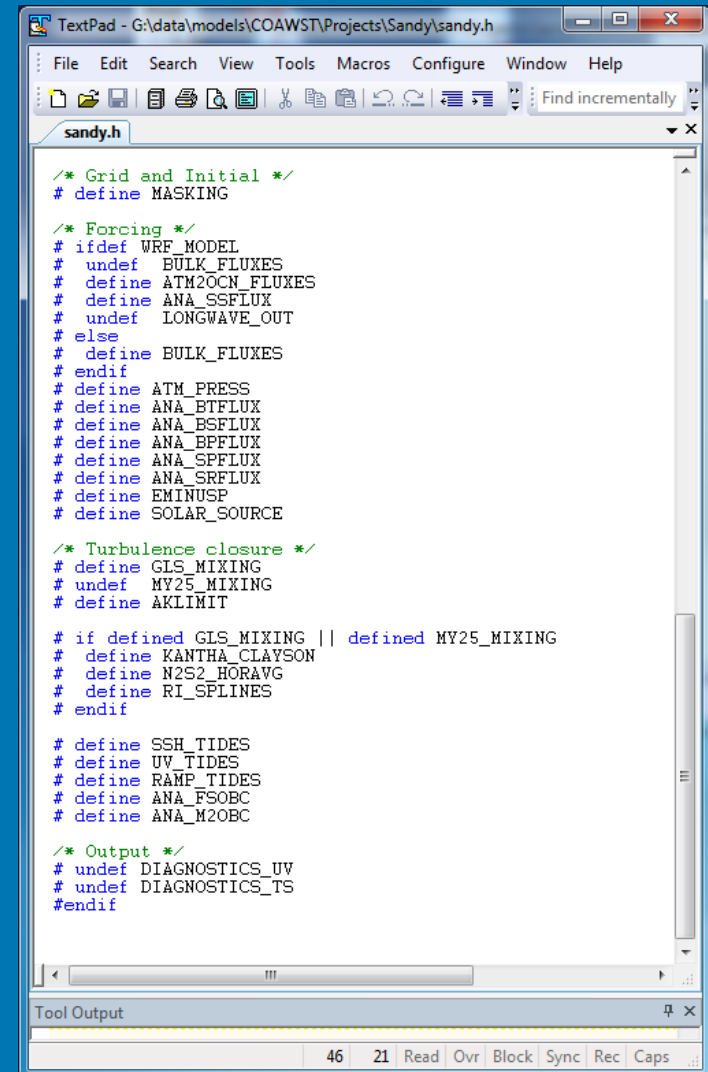
#if defined WRF_MODEL && defined SWAN_MODEL
# define DRAGLIM_DAVIS
# define COARE_TAYLOR_YELLAND
#endif

#ifdef ROMS_MODEL
/* Physics + numerics */
# define UV_ADV
# define UV_COR
# define UV_VIS2
# define MIX_S_UV
# undef TS_FIXED
# define TS_U3HADVECTION
# define TS_C4VADVECTION
# undef TS_MPDATA

# undef SSW_BBL
# ifdef SSW_BBL
# define SSW_CALC_ZNOT
/*#define ANA_SEDIMENT*/
# else
# define UV_LOGDRAG
# endif
# if !defined SWAN_MODEL && defined SSW_BBL
# define ANA_WWAVE
# endif

# define DJ_GRADPS
# define TS_DIF2
# define MIX_GEO_TS
# define CURVGRID

# define SALINITY
# define SOLVE3D
# define SPLINES_VDIFF
# define SPLINES_VVISC
# define AVERAGES
# define NONLIN_EOS
```



```
TextPad - G:\data\models\COAWST\Projects\Sandy\sandy.h
File Edit Search View Tools Macros Configure Window Help
Find incrementally

sandy.h

/* Grid and Initial */
# define MASKING

/* Forcing */
# ifdef WRF_MODEL
# undef BULK_FLUXES
# define ATM2OCN_FLUXES
# define ANA_SSFLUX
# undef LONGWAVE_OUT
# else
# define BULK_FLUXES
# endif
# define ATM_PRESS
# define ANA_BTFLUX
# define ANA_BSFLUX
# define ANA_BPFLUX
# define ANA_SPFLUX
# define ANA_SRFLUX
# define EMINUSP
# define SOLAR_SOURCE

/* Turbulence closure */
# define GLS_MIXING
# undef MY25_MIXING
# define AKLIMIT

# if defined GLS_MIXING || defined MY25_MIXING
# define KANTHA_CLAYSON
# define N2S2_HORAVG
# define RI_SPLINES
# endif

# define SSH_TIDES
# define UV_TIDES
# define RAMP_TIDES
# define ANA_FSOBC
# define ANA_M2OBC

/* Output */
# undef DIAGNOSTICS_UV
# undef DIAGNOSTICS_TS
#endif

Tool Output
46 21 Read Ovr Block Sync Rec Caps
```

8) sandy_ocean.in

```
! Application title.
    TITLE = Hurricane Sandy

! C-preprocessing Flag.
    MyAppCPP = SANDY

! Input variable information file name. This file needs to be processed
! first so all information arrays can be initialized properly.
    VARNAME = ROMS/External/varinfo.dat

! Number of nested grids.
    Ngrids = 2

! Number of grid nesting layers. This parameter is used to allow refinement
! and composite grid combinations.
    NestLayers = 2

! Number of grids in each nesting layer [1:NestLayers].
GridsInLayer = 1 1

! Grid dimension parameters. See notes below in the Glossary for how to set
! these parameters correctly.
    Im == 82 114      ! Number of I-direction INTERIOR RHO-points
    Mm == 62 84       ! Number of J-direction INTERIOR RHO-points
    N == 16 16        ! Number of vertical levels
    ND == 0           ! Number of wave directional bins

    Nbed = 0          ! Number of sediment bed layers

    NAT = 2           ! Number of active tracers (usually, 2)
    NPT = 0           ! Number of inactive passive tracers
    NCS = 0           ! Number of cohesive (mud) sediment tracers
    NNS = 0           ! Number of non-cohesive (sand) sediment tracers

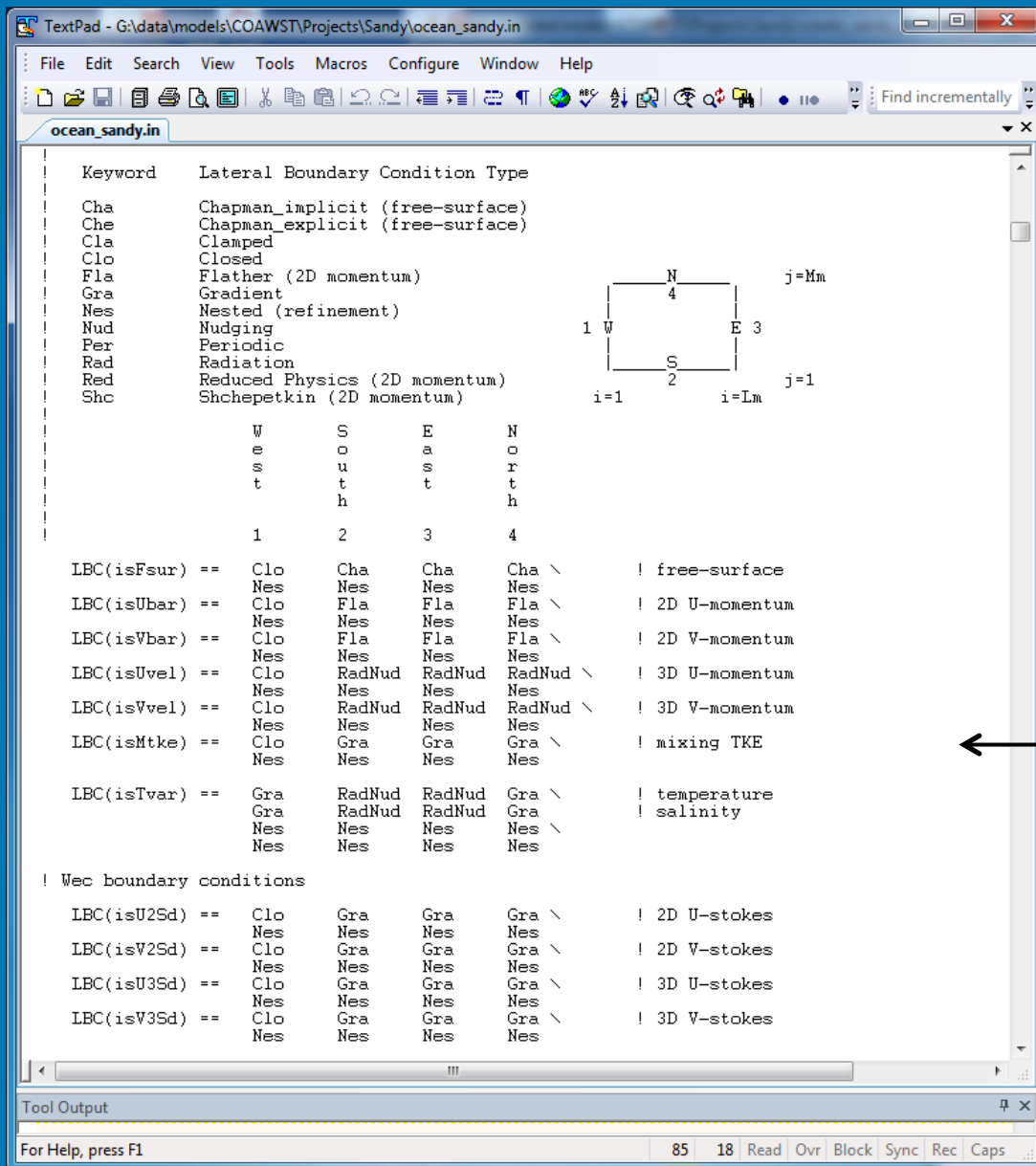
! Domain decomposition parameters for serial, distributed-memory or
! shared-memory configurations used to determine tile horizontal range
! indices (Istr,Iend) and (Jstr,Jend). [1:Ngrids].
-----
    NtileI == 1       ! I-direction partition
    NtileJ == 1       ! J-direction partition
-----
```

Nesting params

Grid sizes

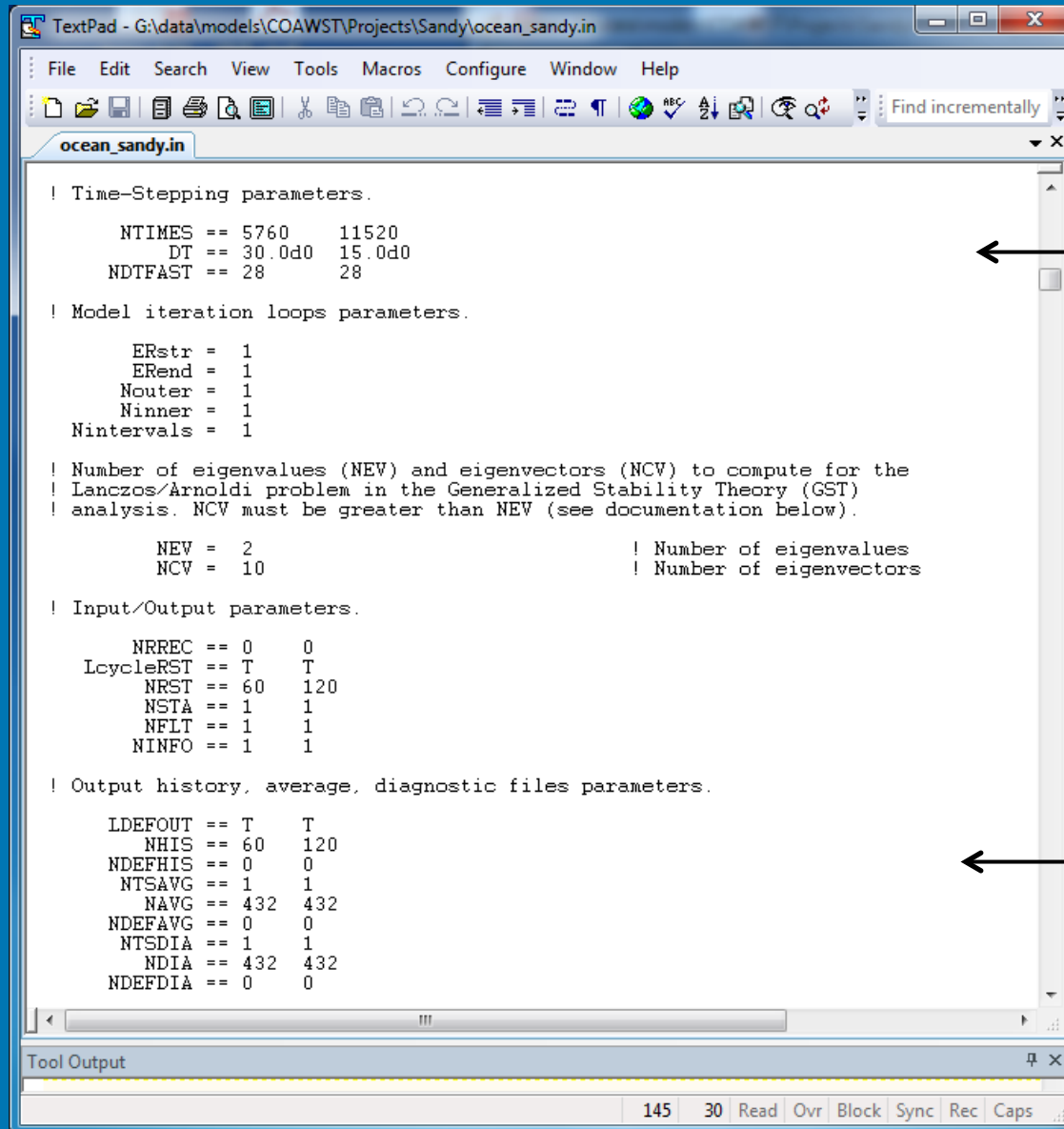
NtileI NtileJ

8) sandy_ocean.in



BC's

8) sandy_ocean.in



```
! Time-Stepping parameters.

      NTIMES == 5760      11520
        DT == 30.0d0    15.0d0
      NDTFAST == 28      28

! Model iteration loops parameters.

      ERstr = 1
      ERend = 1
      Nouter = 1
      Ninner = 1
      Nintervals = 1

! Number of eigenvalues (NEV) and eigenvectors (NCV) to compute for the
! Lanczos/Arnoldi problem in the Generalized Stability Theory (GST)
! analysis. NCV must be greater than NEV (see documentation below).

      NEV = 2              ! Number of eigenvalues
      NCV = 10             ! Number of eigenvectors

! Input/Output parameters.

      NRREC == 0      0
      LcycleRST == T   T
      NRST == 60      120
      NSTA == 1       1
      NFLT == 1       1
      NINFO == 1      1

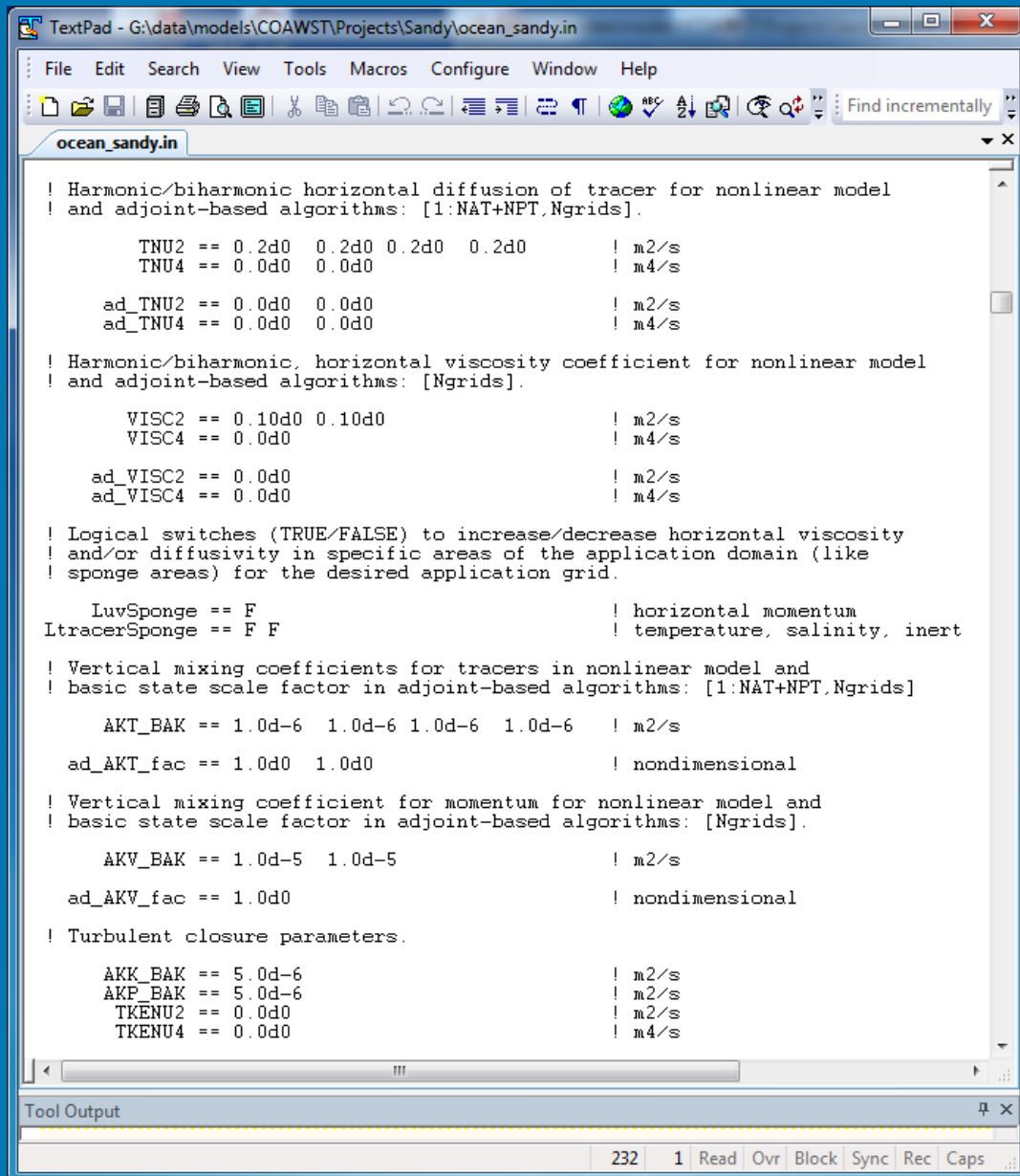
! Output history, average, diagnostic files parameters.

      LDEFOUT == T      T
      NHIS == 60      120
      NDEFHIS == 0      0
      NTSavg == 1       1
      NAVG == 432      432
      NDEFavg == 0      0
      NTSDIA == 1       1
      NDIA == 432      432
      NDEFDIA == 0      0
```

Time stepping

nhis, navg, etc

8) sandy_ocean.in



```
! Harmonic/biharmonic horizontal diffusion of tracer for nonlinear model
! and adjoint-based algorithms: [1:NAT+NPT,Ngrids].

      TNU2 == 0.2d0  0.2d0 0.2d0  0.2d0      ! m2/s
      TNU4 == 0.0d0  0.0d0      ! m4/s

      ad_TNU2 == 0.0d0  0.0d0      ! m2/s
      ad_TNU4 == 0.0d0  0.0d0      ! m4/s

! Harmonic/biharmonic, horizontal viscosity coefficient for nonlinear model
! and adjoint-based algorithms: [Ngrids].

      VISC2 == 0.10d0 0.10d0      ! m2/s
      VISC4 == 0.0d0      ! m4/s

      ad_VISC2 == 0.0d0      ! m2/s
      ad_VISC4 == 0.0d0      ! m4/s

! Logical switches (TRUE/FALSE) to increase/decrease horizontal viscosity
! and/or diffusivity in specific areas of the application domain (like
! sponge areas) for the desired application grid.

      LuvSponge == F      ! horizontal momentum
      ltracerSponge == F F      ! temperature, salinity, inert

! Vertical mixing coefficients for tracers in nonlinear model and
! basic state scale factor in adjoint-based algorithms: [1:NAT+NPT,Ngrids]

      AKT_BAK == 1.0d-6  1.0d-6 1.0d-6  1.0d-6      ! m2/s

      ad_AKT_fac == 1.0d0  1.0d0      ! nondimensional

! Vertical mixing coefficient for momentum for nonlinear model and
! basic state scale factor in adjoint-based algorithms: [Ngrids].

      AKV_BAK == 1.0d-5  1.0d-5      ! m2/s

      ad_AKV_fac == 1.0d0      ! nondimensional

! Turbulent closure parameters.

      AKK_BAK == 5.0d-6      ! m2/s
      AKP_BAK == 5.0d-6      ! m2/s
      TKENU2 == 0.0d0      ! m2/s
      TKENU4 == 0.0d0      ! m4/s
```

8) sandy_ocean.in

```
TextPad - G:\data\models\COAWST\Projects\Sandy\ocean_sandy.in
File Edit Search View Tools Macros Configure Window Help
ocean_sandy.in
! Generic length-scale turbulence closure parameters.
    GLS_P == 3.0d0          ! K-epsilon
    GLS_M == 1.5d0
    GLS_N == -1.0d0
    GLS_Kmin == 7.6d-6
    GLS_Pmin == 1.0d-12
    GLS_CMU0 == 0.5477d0
    GLS_C1 == 1.44d0
    GLS_C2 == 1.92d0
    GLS_C3M == -0.4d0
    GLS_C3P == 1.0d0
    GLS_SIGK == 1.0d0
    GLS_SIGP == 1.30d0
! Constants used in surface turbulent kinetic energy flux computation.
    CHARNOK_ALPHA == 1400.0d0      ! Charnok surface roughness
    ZOS_HSIG_ALPHA == 0.5d0        ! roughness from wave amplitude
    SZ_ALPHA == 0.25d0             ! roughness from wave dissipation
    CRGBAN_CW == 100.0d0           ! Craig and Banner wave breaking
    WEC_ALPHA == 0.0d0             ! 0: all wave dissip goes to break
                                   ! 1: all wave dissip goes to roller
! Constants used in momentum stress computation.
    RDRG == 3.0d-04               ! m/s
    RDRG2 == 0.025d0              ! nondimensional
    Zob == 0.02d0                 ! m
    Zos == 0.02d0                 ! m
! Height (m) of atmospheric measurements for Bulk fluxes parameterization.
    BLK_ZQ == 2.0d0               ! air humidity
    BLK_ZT == 2.0d0               ! air temperature
    BLK_ZW == 10.0d0              ! winds
! Minimum depth for wetting and drying.
    DCRIT == 0.10d0               ! m
! Various parameters.
    WTYPE == 1
    LEVSFRC == 15
    LEVBFR == 1
! Set vertical, terrain-following coordinates transformation equation and
! stretching function (see below for details). [1:Ngrids].
    Vtransform == 1                ! transformation equation
    Vstretching == 1              ! stretching function
! Vertical S-coordinates parameters (see below for details). [1:Ngrids].
    THETA_S == 5.0d0              ! surface stretching parameter
    THETA_B == 0.4d0              ! bottom stretching parameter
    TCLINE == 50.0d0              ! critical depth (m)
```

← GLS params

← Wave breaking params

← Bottom roughness

← Bulk flux heights

← Grid stretching params

8) sandy_ocean.in

```
TextPad - G:\data\models\COAWST\Projects\Sandy\ocean_sandy.in
File Edit Search View Tools Macros Configure Window Help
ocean_sandy.in Sort
! Mean Density and Brunt-Vaisala frequency.
  RHO0 = 1025.0d0 ! kg/m3
  BVF_BAK = 1.0d-5 ! 1/s2
! Time-stamp assigned for model initialization, reference time
! origin for tidal forcing, and model reference time for output
! NetCDF units attribute.
  DSTART = 56228.5d0 56228.5d0 ! days
  TIDE_START = 52866.0d0 ! days
  TIME_REF = 18581117.0d0 ! yyyyymmdd.dd
! Nudging/relaxation time scales, inverse scales will be computed
! internally, [1:Ngrids].
  TNUDG == 1.0d0 1.0d0 ! days
  ZNUDG == 0.0d0 ! days
  M2NUDG == 0.0d0 ! days
  M3NUDG == 1.0d0 ! days
! Factor between passive (outflow) and active (inflow) open boundary
! conditions, [1:Ngrids]. If OBCFAC > 1, nudging on inflow is stronger
! than on outflow (recommended).
  OBCFAC == 0.0d0 ! nondimensional
! Linear equation of State parameters:
  R0 == 1027.0d0 ! kg/m3
  T0 == 10.0d0 ! Celsius
  S0 == 30.0d0 ! nondimensional
  TCOEF == 1.7d-4 ! 1/Celsius
  SCOEF == 7.6d-4 ! nondimensional
! Slipperiness parameter: 1.0 (free slip) or -1.0 (no slip)
  GAMMA2 == 1.0d0
! Logical switches (TRUE/FALSE) to activate horizontal momentum transport
! point Sources/Sinks (like river runoff transport) and mass point
! Sources/Sinks (like volume vertical influx), [1:Ngrids].
  LuvSrc == F ! horizontal momentum transport
  LvSrc == F ! volume vertical influx
! Logical switches (TRUE/FALSE) to activate tracers point Sources/Sinks
! (like river runoff) and to specify which tracer variables to consider:
! [1:NAT+NPT,Ngrids]. See glossary below for details.
  ItracerSrc == F F ! temperature, salinity, inert
! Logical switches (TRUE/FALSE) to read and process climatology fields.
! See glossary below for details.
  LsshCLM == F ! sea-surface height
  Lm2CLM == F ! 2D momentum
  Lm3CLM == F ! 3D momentum
  ItracerCLM == F F ! temperature, salinity, inert
! Logical switches (TRUE/FALSE) to nudge the desired climatology field(s).
! If not analytical climatology fields, users need to turn ON the logical
! switches above to process the fields from the climatology NetCDF file
! that are needed for nudging. See glossary below for details.
  LnudgeM2CLM == F ! 2D momentum
  LnudgeM3CLM == F ! 3D momentum
  LnudgeTCLM == F F ! temperature, salinity, inert
```

← Time starts, tide start

← BC tnudge

8) sandy_ocean.in

```
TextPad - G:\data\models\COAWST\Projects\Sandy\ocean_sandy.in *
File Edit Search View Tools Macros Configure Window Help
ocean_sandy.in *
! Logical switches (TRUE/FALSE) to activate writing of fields into
! HISTORY output file.

Hout(idUvel) == T T      | u          3D U-velocity
Hout(idVvel) == T T      | v          3D V-velocity
Hout(idu3dE) == F F      | u_eastward 3D U-eastward at RHO-points
Hout(idv3dN) == F F      | v_northward 3D V-northward at RHO-points
Hout(idWvel) == T T      | w          3D W-velocity
Hout(idOvel) == T T      | omega       omega vertical velocity
Hout(idUbar) == T T      | ubar        2D U-velocity
Hout(idVbar) == T T      | vbar        2D V-velocity
Hout(idu2dE) == F F      | ubar_eastward 2D U-eastward at RHO-points
Hout(idv2dN) == F F      | vbar_northward 2D V-northward at RHO-points
Hout(idFsur) == T T      | zeta        free-surface
Hout(idBath) == T T      | bath        time-dependent bathymetry

Hout(idTvar) == T T T T | temp, salt  temperature and salinity

Hout(idpthR) == F      | z_rho       time-varying depths of RHO-points
Hout(idpthU) == F      | z_u         time-varying depths of U-points
Hout(idpthV) == F      | z_v         time-varying depths of V-points
Hout(idpthW) == F      | z_w         time-varying depths of W-points

.....

! Logical switches (TRUE/FALSE) to activate writing of time-averaged
! fields into AVERAGE output file.

Aout(idUvel) == F      | u          3D U-velocity
Aout(idVvel) == F      | v          3D V-velocity
Aout(idu3dE) == F      | u_eastward 3D U-eastward at RHO-points
Aout(idv3dN) == F      | v_northward 3D V-northward at RHO-points
Aout(idWvel) == F      | w          3D W-velocity
Aout(idOvel) == F      | omega       omega vertical velocity
Aout(idUbar) == F      | ubar        2D U-velocity
Aout(idVbar) == F      | vbar        2D V-velocity
Aout(idu2dE) == F      | ubar_eastward 2D U-eastward at RHO-points
Aout(idv2dN) == F      | vbar_northward 2D V-northward at RHO-points
Aout(idFsur) == F      | zeta        free-surface
Aout(idBath) == F      | bath        time-dependent bathymetry

.....

! Logical switches (TRUE/FALSE) to activate writing of time-averaged,
! 2D momentum (ubar,vbar) diagnostic terms into DIAGNOSTIC output file.

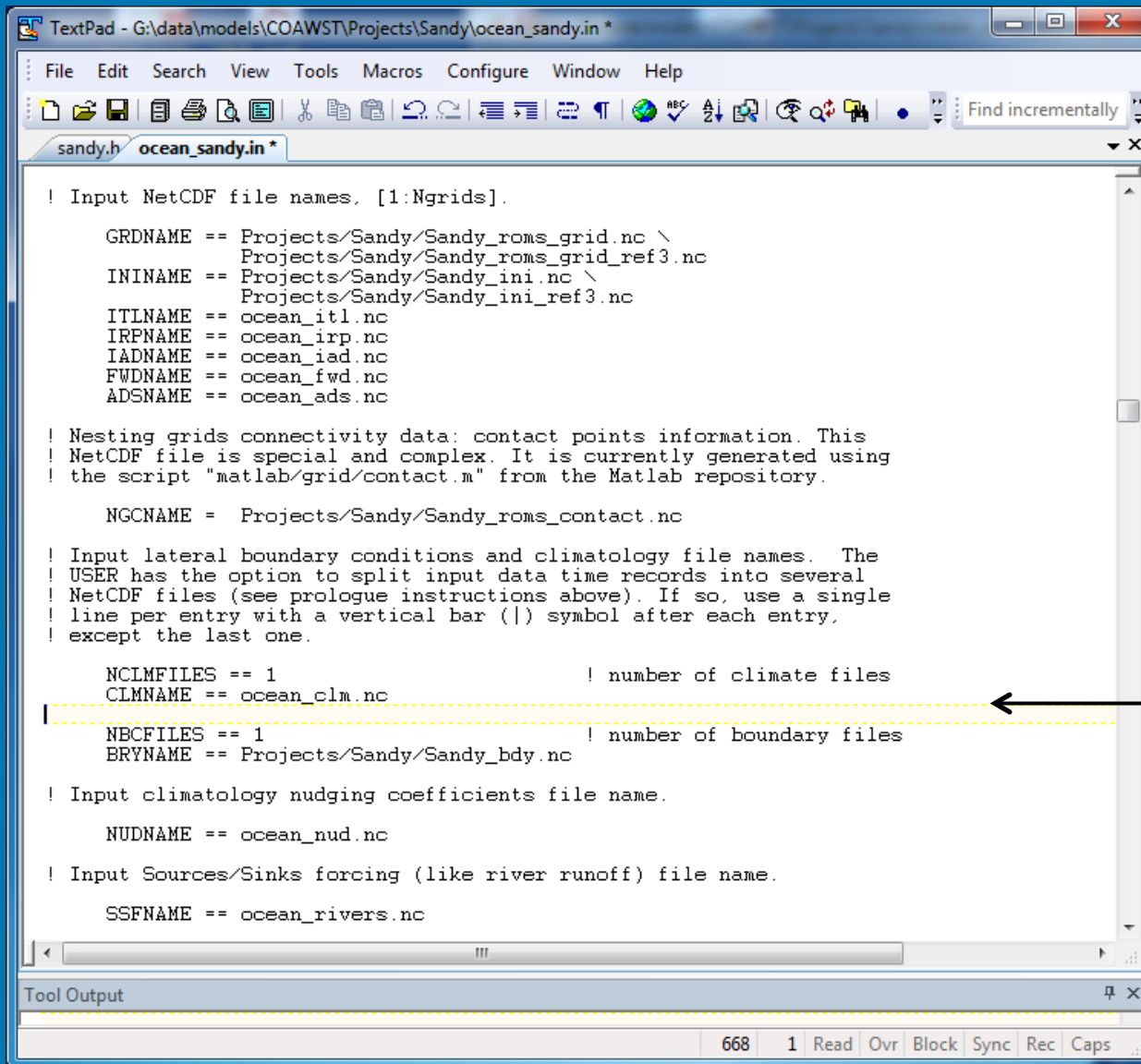
Dout(M2rate) == F      | ubar_accel, ... acceleration
Dout(M2pgrd) == F      | ubar_prgd, ... pressure gradient
Dout(M2fcor) == F      | ubar_cor, ... Coriolis force
Dout(M2hadv) == F      | ubar_hadv, ... horizontal total advection
Dout(M2xadv) == F      | ubar_xadv, ... horizontal XI-advection
Dout(M2yadv) == F      | ubar_yadv, ... horizontal ETA-advection
Dout(M2hrad) == F      | ubar_hrad, ... horizontal total vec_mellor radiation stress
Dout(M2hvis) == F      | ubar_hvisc, ... horizontal total viscosity
Dout(M2xvis) == F      | ubar_xvisc, ... horizontal XI-viscosity
Dout(M2yvis) == F      | ubar_yvisc, ... horizontal ETA-viscosity
Dout(M2sstr) == F      | ubar_sstr, ... surface stress
Dout(M2bstr) == F      | ubar_bstr, ... bottom stress
Dout(M2hvjf) == F      | 2D vec_vf horizontal J vortex force
Dout(M2kvrf) == F      | 2D vec_vf K vortex force
Dout(M2fsco) == F      | 2D vec_vf coriolis-stokes
```

History

Averages

Diagnostics

8) sandy_ocean.in



```
! Input NetCDF file names, [1:Ngrids].

GRDNAME == Projects/Sandy/Sandy_roms_grid.nc \
Projects/Sandy/Sandy_roms_grid_ref3.nc
ININAME == Projects/Sandy/Sandy_ini.nc \
Projects/Sandy/Sandy_ini_ref3.nc
ITLNAME == ocean_itl.nc
IRPNAME == ocean_irp.nc
IADNAME == ocean_iad.nc
FWDNAME == ocean_fwd.nc
ADSNAME == ocean_ads.nc

! Nesting grids connectivity data: contact points information. This
! NetCDF file is special and complex. It is currently generated using
! the script "matlab/grid/contact.m" from the Matlab repository.

NGCNAME = Projects/Sandy/Sandy_roms_contact.nc

! Input lateral boundary conditions and climatology file names. The
! USER has the option to split input data time records into several
! NetCDF files (see prologue instructions above). If so, use a single
! line per entry with a vertical bar (|) symbol after each entry,
! except the last one.

NCLMFILES == 1 ! number of climate files
CLMNAME == ocean_clm.nc
|
NBCFILES == 1 ! number of boundary files
BRYNAME == Projects/Sandy/Sandy_bdy.nc

! Input climatology nudging coefficients file name.

NUDNAME == ocean_nud.nc

! Input Sources/Sinks forcing (like river runoff) file name.

SSFNAME == ocean_rivers.nc
```

Different in COAWST

8) sandy_ocean.in

```
! Input forcing NetCDF file name(s). The USER has the option to enter
! several file names for each nested grid. For example, the USER may
! have different files for wind products, heat fluxes, tides, etc.
! The model will scan the file list and will read the needed data from
! the first file in the list containing the forcing field. Therefore,
! the order of the file names is very important. If using multiple forcing
! files per grid, first enter all the file names for grid 1, then grid 2,
! and so on. It is also possible to split input data time records into
! several NetCDF files (see prologue instructions above). Use a single line
! per entry with a continuation (\) or vertical bar (|) symbol after each
! entry, except the last one.

NFILES == 2 1                      ! number of unique forcing files

FRCNAME == Projects/Sandy/roms_namnarr_Sandy2012.nc \
           Projects/Sandy/tide_forc_Sandy.nc \
           Projects/Sandy/roms_namnarr_Sandy2012.nc

! Output NetCDF file names, [1:Ngrids].

DAIName == ocean_dai.nc
GSTName == ocean_gst.nc
RSTName == Sandy_ocean_rst.nc \
           Sandy_ocean_ref3_rst.nc
HISName == Sandy_ocean_his.nc \
           Sandy_ocean_ref3_his.nc
TLMName == ocean_tlm.nc
TLFName == ocean_tlf.nc
ADJName == ocean_adj.nc
AVGName == Sandy_ocean_avg.nc \
           Sandy_ocean_ref3_avg.nc
DIAName == ocean_dia.nc
STAName == ocean_sta.nc
FLTName == oceanflt.nc

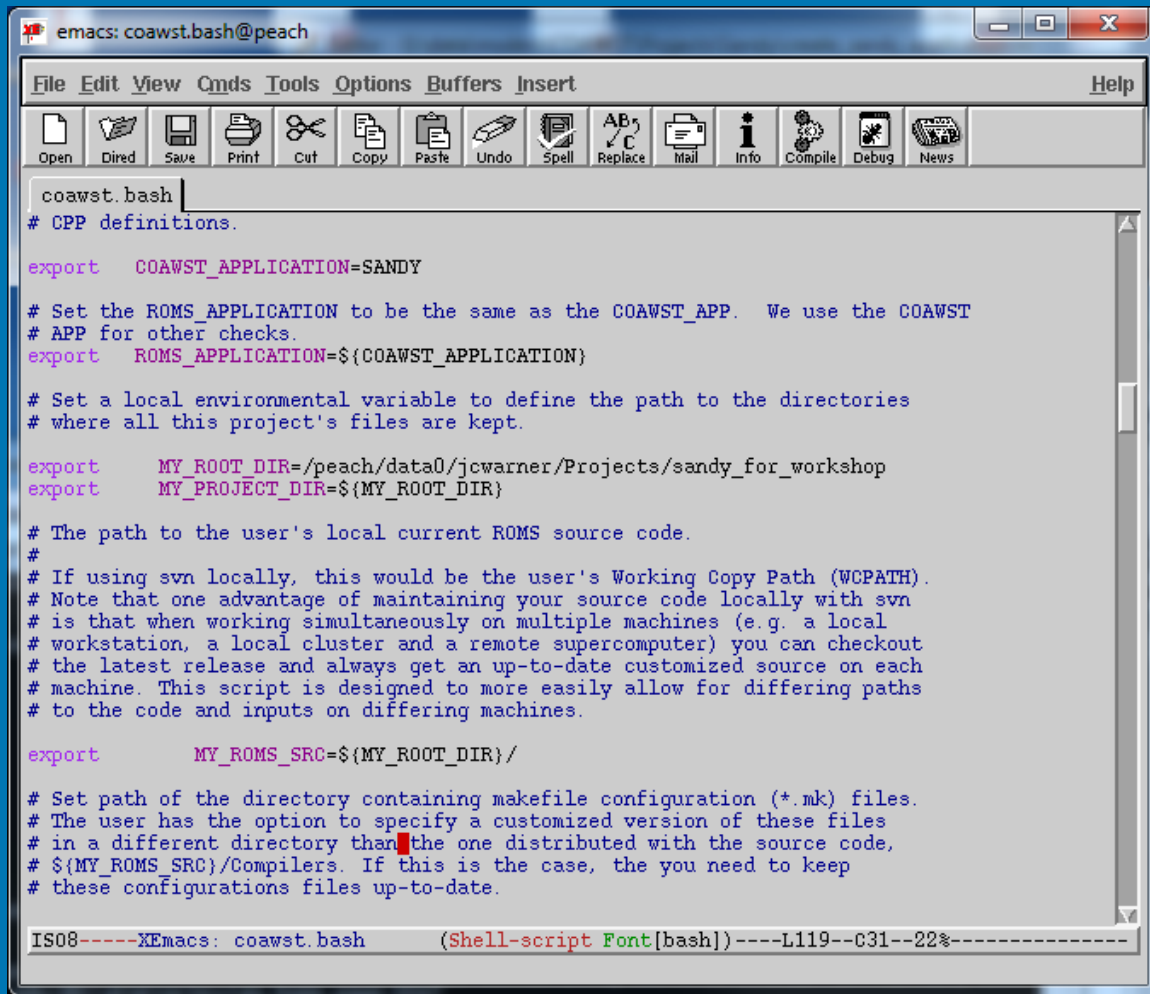
! Input ASCII parameter filenames.

APARNAM = ROMS/External/s4dvar.in
SPOSNAM = ROMS/External/stations.in
FPOSNAM = ROMS/External/floats.in
BPARNAM = ROMS/External/bioFasham.in
SPARNAM = ROMS/External/sediment.in
USRNAME = ROMS/External/MyFile.dat
```

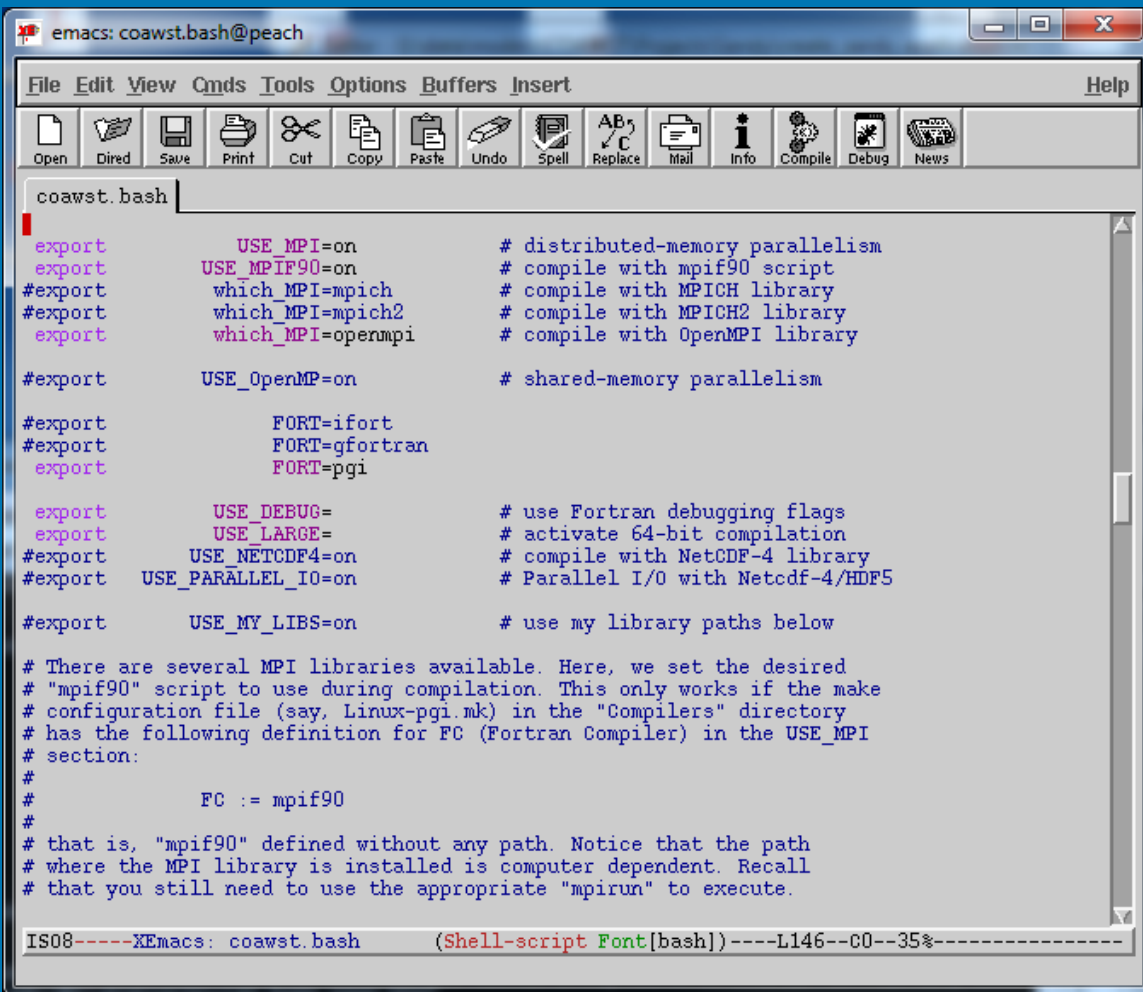
← Only list tide forc for parent

← his and avg names, etc

9) coawst.bash

A screenshot of an Emacs window titled 'emacs: coawst.bash@peach'. The window has a menu bar with 'File', 'Edit', 'View', 'Cmds', 'Tools', 'Options', 'Buffers', 'Insert', and 'Help'. Below the menu bar is a toolbar with icons for Open, Dired, Save, Print, Cut, Copy, Paste, Undo, Spell, Replace, Mail, Info, Compile, Debug, and News. The main text area shows the content of the 'coawst.bash' script. The script starts with '# CPP definitions.' and includes several 'export' statements for environment variables like 'COAWST_APPLICATION', 'ROMS_APPLICATION', 'MY_ROOT_DIR', 'MY_PROJECT_DIR', and 'MY_ROMS_SRC'. It also contains several comments explaining the script's purpose and usage. The status bar at the bottom shows 'ISO8-----XEmacs: coawst.bash (Shell-script Font[bash])-----L119--C31--22%-----'.

9) coawst.bash



```
emacs: coawst.bash@peach
File Edit View Cmds Tools Options Buffers Insert Help
Open Dired Save Print Cut Copy Paste Undo Spell Replace Mail Info Compile Debug News

coawst.bash

export      USE_MPI=on           # distributed-memory parallelism
export      USE_MPIF90=on        # compile with mpif90 script
#export     which_MPI=mpich      # compile with MPICH library
#export     which_MPI=mpich2     # compile with MPICH2 library
export      which_MPI=openmpi    # compile with OpenMPI library

#export     USE_OpenMP=on        # shared-memory parallelism

#export     FORT=ifort
#export     FORT=gfortran
export      FORT=pgi

export      USE_DEBUG=           # use Fortran debugging flags
export      USE_LARGE=           # activate 64-bit compilation
#export     USE_NETCDF4=on       # compile with NetCDF-4 library
#export     USE_PARALLEL_IO=on   # Parallel I/O with Netcdf-4/HDF5

#export     USE_MY_LIBS=on       # use my library paths below

# There are several MPI libraries available. Here, we set the desired
# "mpif90" script to use during compilation. This only works if the make
# configuration file (say, Linux-pgi.mk) in the "Compilers" directory
# has the following definition for FC (Fortran Compiler) in the USE_MPI
# section:
#
#         FC := mpif90
#
# that is, "mpif90" defined without any path. Notice that the path
# where the MPI library is installed is computer dependent. Recall
# that you still need to use the appropriate "mpirun" to execute.

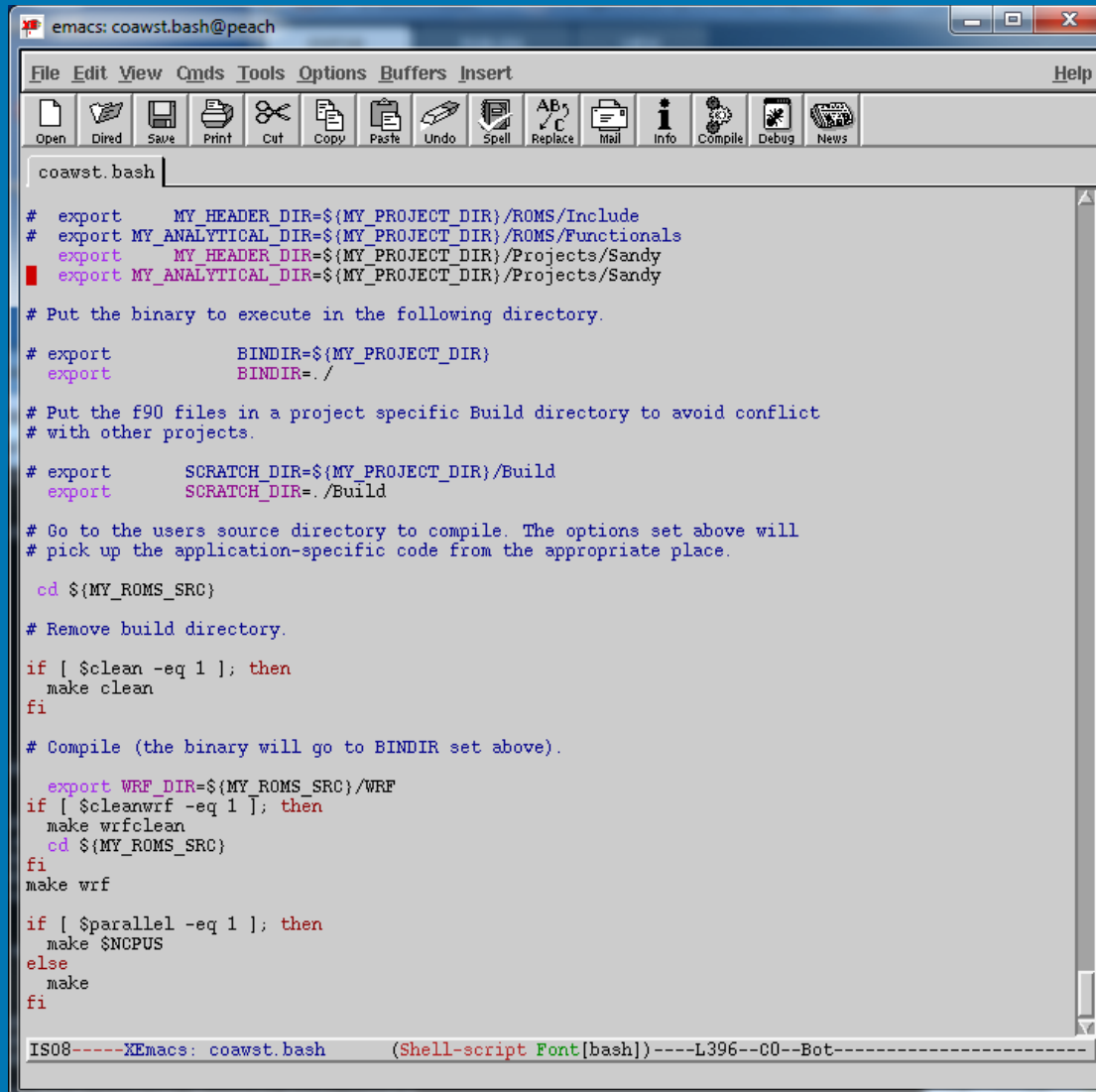
IS08-----XEmacs: coawst.bash      (Shell-script Font[bash])-----L146--C0--35%-----
```

Makefile determines the
Compilers file from:

- 1) `uname -s`
- 2) the FORT value

For example, here it will be
Compilers/Linux-pgi.mk

9) coawst.bash

A screenshot of an Emacs window titled 'emacs: coawst.bash@peach'. The window displays the contents of the 'coawst.bash' script. The script is a shell script that sets various environment variables for project directories, defines build and scratch directories, and contains logic for cleaning, compiling, and running the code. The script uses 'cd' to change directories and 'make' to build the code. The Emacs interface includes a menu bar with 'File', 'Edit', 'View', 'Cmds', 'Tools', 'Options', 'Buffers', 'Insert', and 'Help'. Below the menu bar is a toolbar with icons for common actions like Open, Save, Print, Cut, Copy, Paste, Undo, Spell, Replace, Mail, Info, Compile, Debug, and News. The script text is as follows:

```
coawst.bash

# export MY_HEADER_DIR=${MY_PROJECT_DIR}/ROMS/Include
# export MY_ANALYTICAL_DIR=${MY_PROJECT_DIR}/ROMS/Functionals
# export MY_HEADER_DIR=${MY_PROJECT_DIR}/Projects/Sandy
# export MY_ANALYTICAL_DIR=${MY_PROJECT_DIR}/Projects/Sandy

# Put the binary to execute in the following directory.

# export BINDIR=${MY_PROJECT_DIR}
# export BINDIR=./

# Put the f90 files in a project specific Build directory to avoid conflict
# with other projects.

# export SCRATCH_DIR=${MY_PROJECT_DIR}/Build
# export SCRATCH_DIR=./Build

# Go to the users source directory to compile. The options set above will
# pick up the application-specific code from the appropriate place.

cd ${MY_ROMS_SRC}

# Remove build directory.
if [ $clean -eq 1 ]; then
    make clean
fi

# Compile (the binary will go to BINDIR set above).

export WRF_DIR=${MY_ROMS_SRC}/WRF
if [ $cleanwrf -eq 1 ]; then
    make wrfclean
cd ${MY_ROMS_SRC}
fi
make wrf

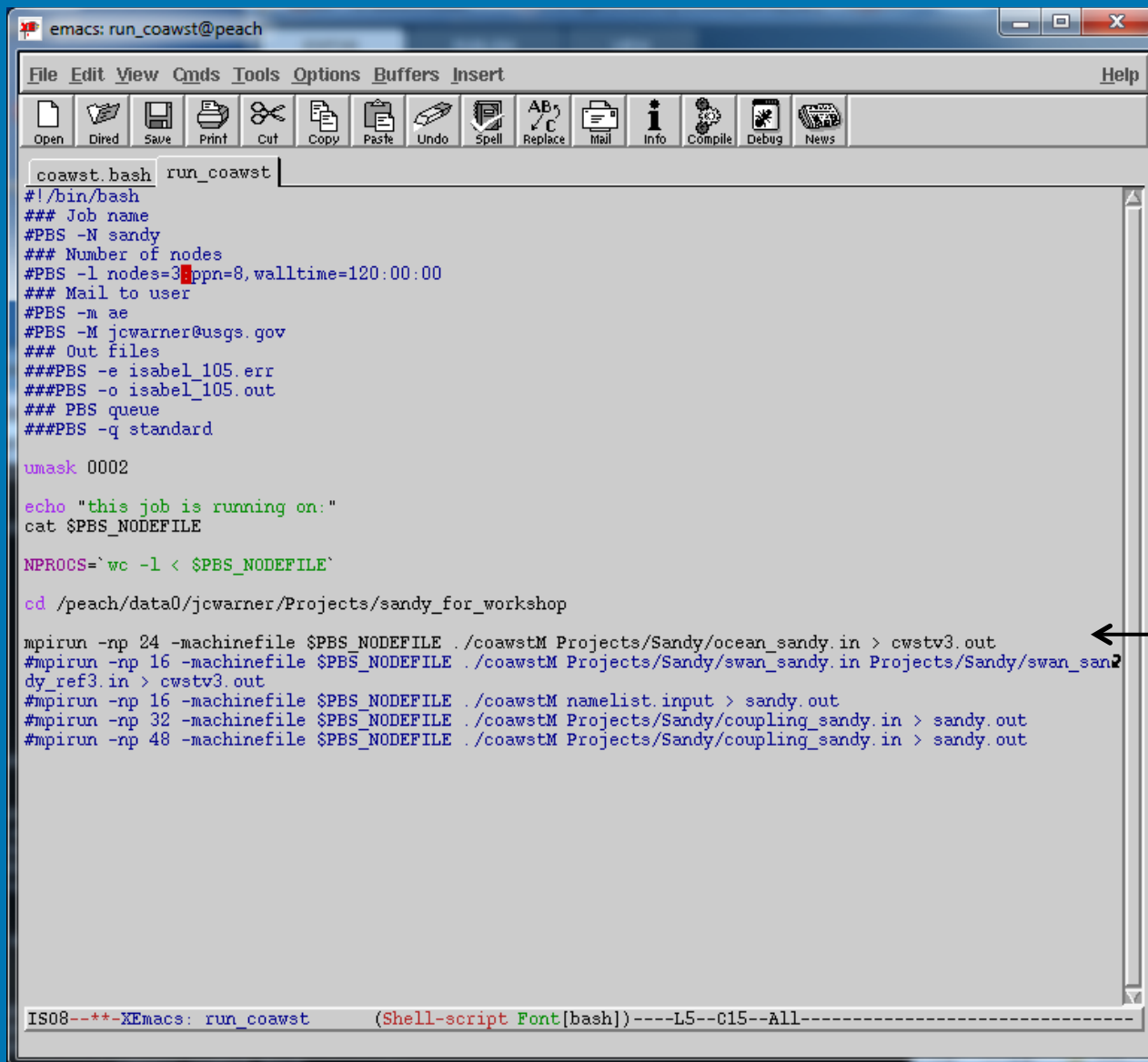
if [ $parallel -eq 1 ]; then
    make $NCPUS
else
    make
fi

IS08-----XEmacs: coawst.bash (Shell-script Font[bash])-----L396--C0--Bot-----
```

To build the code use
./coawst.bash -j

this should make the
coawstM

10) run_file



```
emacs: run_coawst@peach
File Edit View Cmds Tools Options Buffers Insert Help
Open Dired Save Print Cut Copy Paste Undo Spell Replace Mail Info Compile Debug News

coawst.bash run_coawst
#!/bin/bash
### Job name
#PBS -N sandy
### Number of nodes
#PBS -l nodes=38ppn=8,walltime=120:00:00
### Mail to user
#PBS -m ae
#PBS -M jcwarner@usgs.gov
### Out files
###PBS -e isabel_105.err
###PBS -o isabel_105.out
### PBS queue
###PBS -q standard

umask 0002

echo "this job is running on:"
cat $PBS_NODEFILE

NPROCS=`wc -l < $PBS_NODEFILE`

cd /peach/data0/jcwarner/Projects/sandy_for_workshop

mpirun -np 24 -machinefile $PBS_NODEFILE ./coawstM Projects/Sandy/ocean_sandy.in > cwstv3.out
#mpirun -np 16 -machinefile $PBS_NODEFILE ./coawstM Projects/Sandy/swan_sandy.in Projects/Sandy/swan_sand
dy_ref3.in > cwstv3.out
#mpirun -np 16 -machinefile $PBS_NODEFILE ./coawstM namelist.input > sandy.out
#mpirun -np 32 -machinefile $PBS_NODEFILE ./coawstM Projects/Sandy/coupling_sandy.in > sandy.out
#mpirun -np 48 -machinefile $PBS_NODEFILE ./coawstM Projects/Sandy/coupling_sandy.in > sandy.out

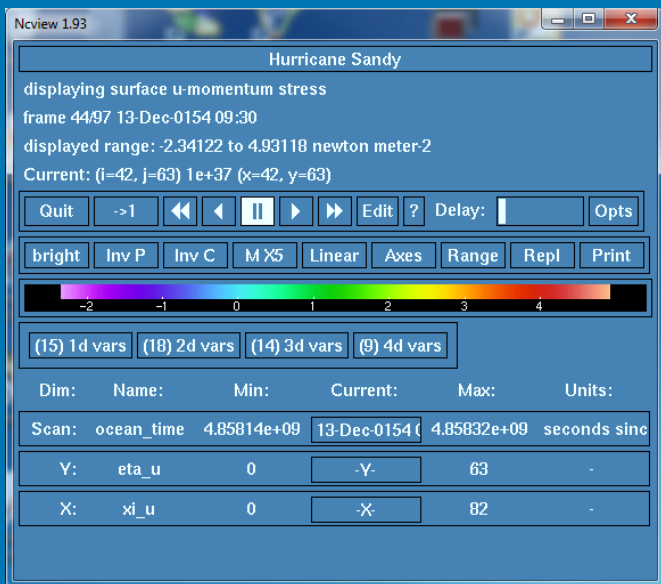
ISO8-***-XEmacs: run_coawst (Shell-script Font(bash))----L5--C15--A11-----
```

Make sure the `-np X` number of procs is equal to the `NtileI * NtileJ` in the ocean.in file.

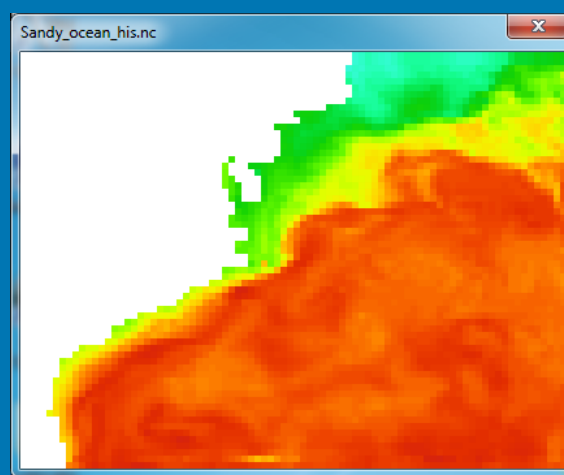
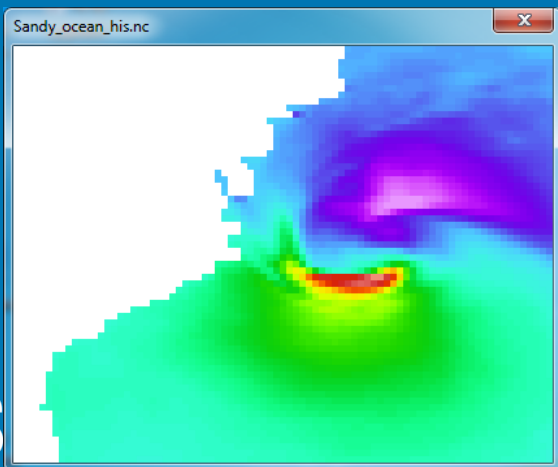
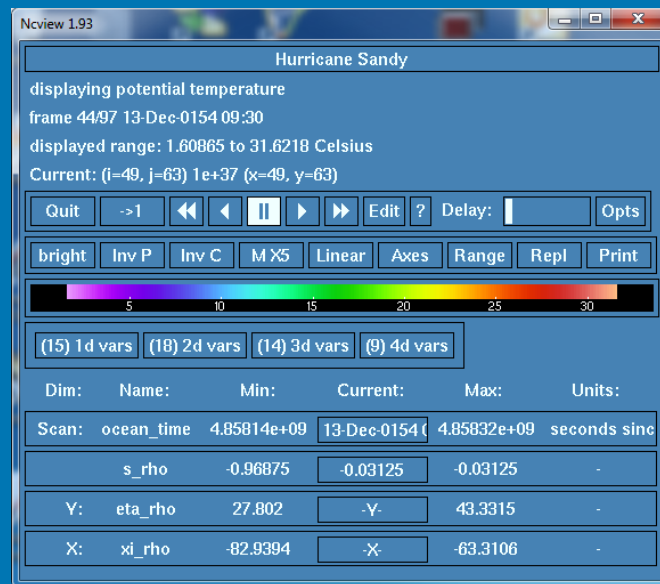
To run just ROMS, the mpirun should point to the ocean.in

Output

sustr



temp (surface)



It's that easy. I don't understand why people have problems.