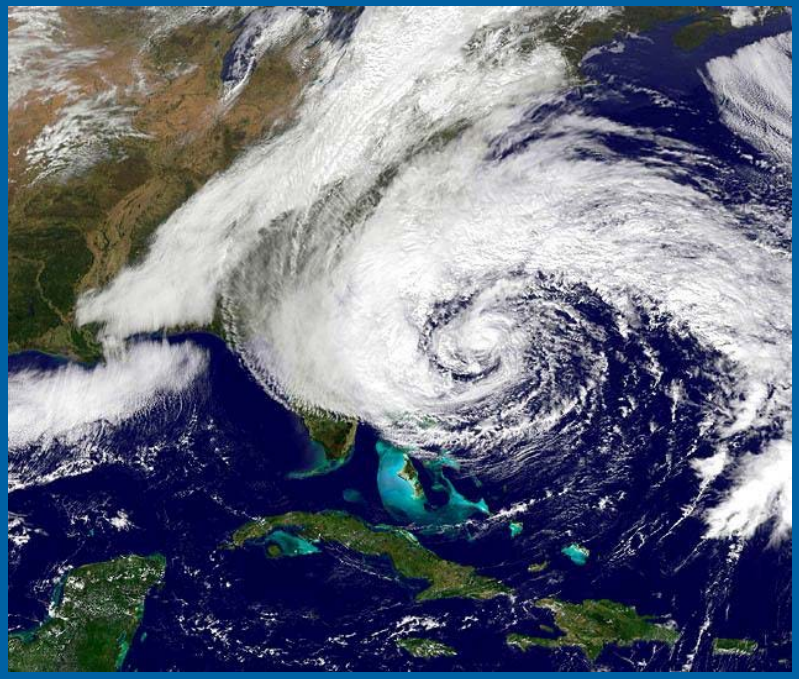
USGS
science for a changing world

Coupled model applications Tutorial

Tuesday afternoon
1:46:00 ~



Projects/Sandy- Hurricane Sandy example

U.S. Department of the Interior
U.S. Geological Survey

- How does the coupled modeling system work (MCT)

- SCRIP interpolation

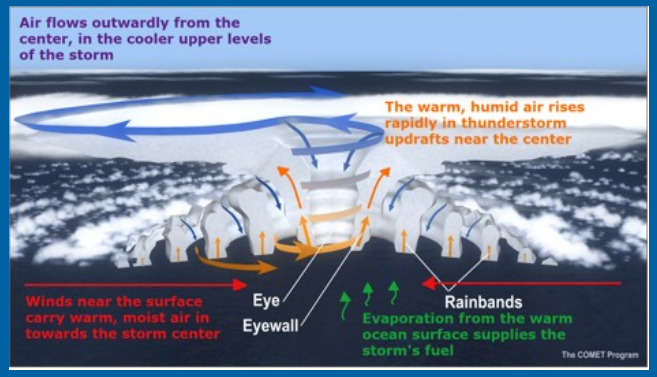
- Setting up a coupled application

USGS

Why do we couple models?

Usually treated independently but actually occur together

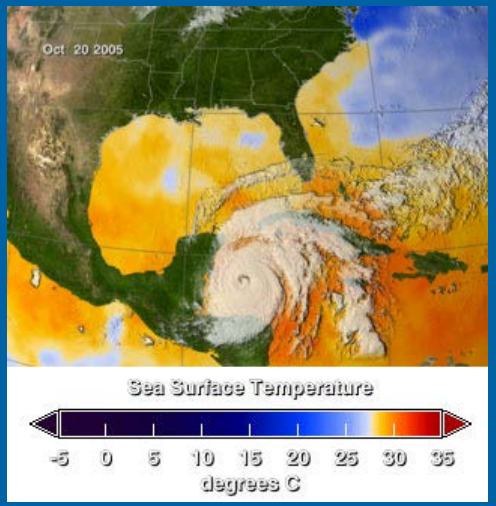
Atmosphere



Coastal geomorphology

Wave

Ocean

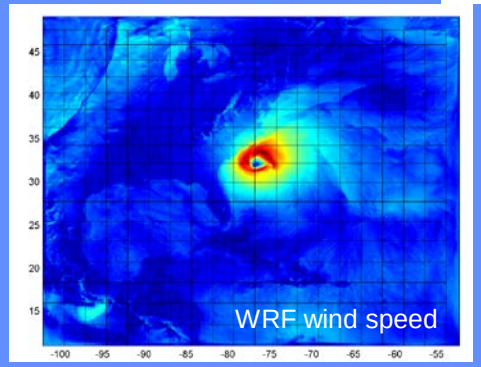


USGS

Model setup

ATMOSPHERE

WRF



Uwind, Vwind, Patm, RH, Tair, cloud, rain, evap, SWrad, Lwrad LH, HFX, Ustress, Vstress

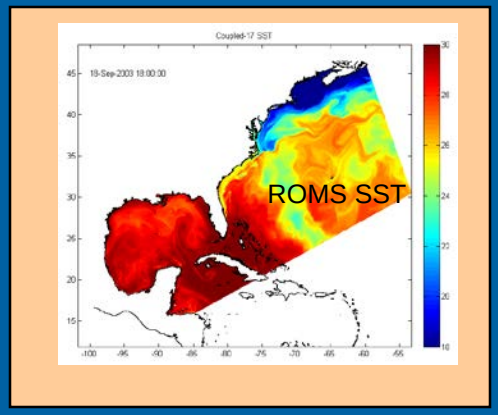
MCT

SST

H_{wave}, L_{pwave}, T_{psurf},

MCT

U_{wind}, V_{wind}



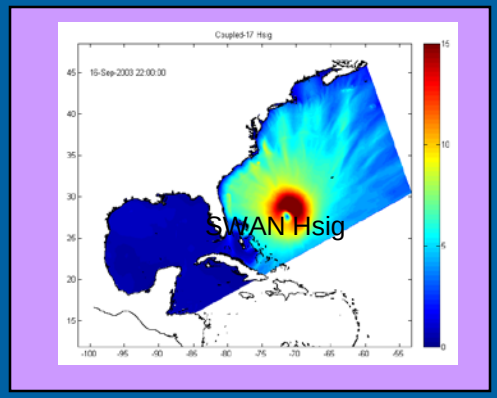
OCEAN

ROMS

H_{wave}, L_{mwave}, L_{pwave}, D_{wave}, D_{wavep}, T_{psurf}, T_{mbott}, Q_b, $Diss_{bot}$, $Diss_{surf}$, $Diss_{wcap}$, U_{bot}

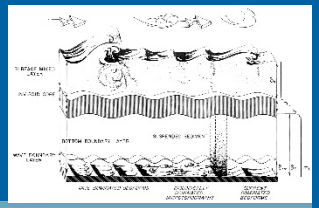
MCT

u_s, v_s, η, bath, Z_0



WAVE

SWAN



SEDIMENT

CSTMS
Community Sediment Transport Modeling System

USGS

Ice
Biology
Vegetation

OCN interactions

Use consistent stress
roms+wrf
#define
ATM2OCN_FLUXES

or

Use wrf vars in
COARE
#define
BULK_FLUXES

Ustress, Vstress,
Swrad, Lwrad
LH, HFX

Uwind-ucur,
Vwind-vcur,
Swrad, Lwrad,
RH, Tair,
cloud

Salt flux

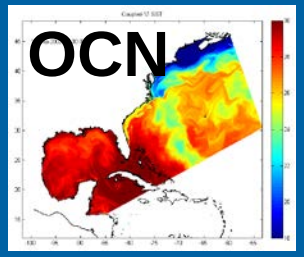
rain,
evap

Atm press

Patm

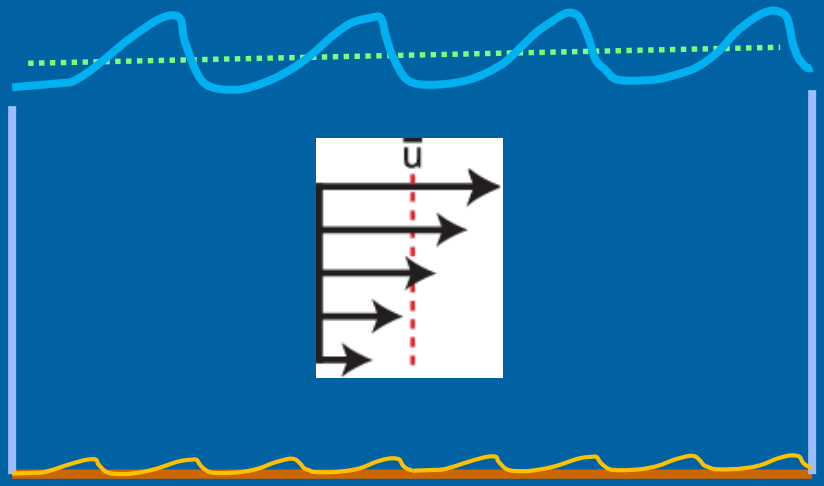
ATM

Uwind, Vwind, Patm, RH, Tair,
cloud, rain, evap, SWrad, Lwrad
LH, HFX, Ustress, Vstress



H_{wave}, L_{mwave}, L_{pwave}, D_{wave},
T_{psurf}, T_{mbott}, Q_b,
$Diss_{bot}$, $Diss_{surf}$, $Diss_{wcap}$,
U_{bot}

WAVE



Water column

vortex force

Stokes + VF

Surface

$\tau_{s} = f(Z_{os})$

roughness

Bottom

Zoa

$\tau_{b} = f(Z_{ob})$

H_{wave}, L_{mwave}, D_{wave},
T_{psurf}, Q_b,
$Diss_{bot}$, $Diss_{surf}$,
$Diss_{wcap}$,

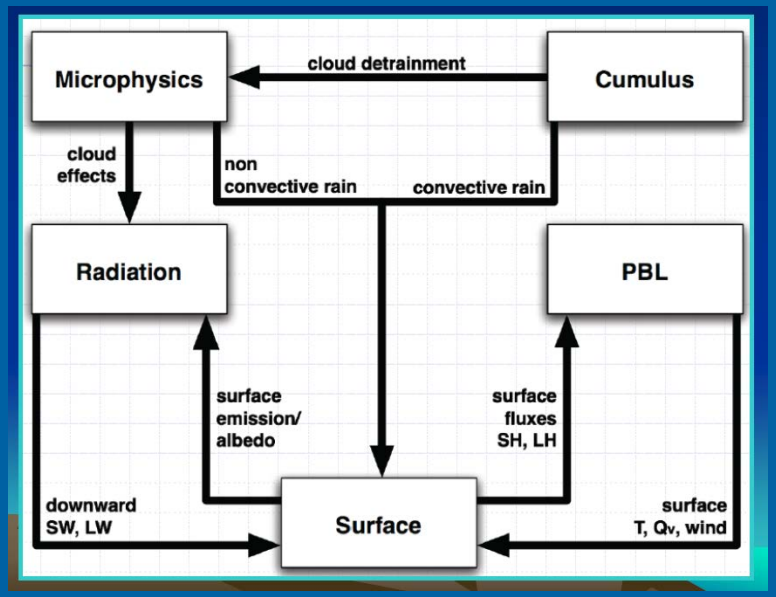
H_{wave}, L_{pwave},
D_{wave}, T_{psurf},

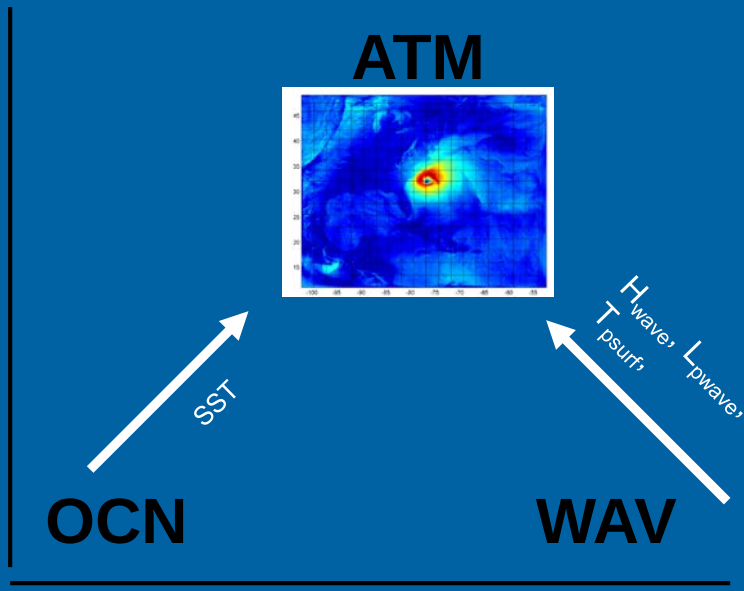
H_{wave}, L_{mwave},
D_{wave},
T_{mbott}, U_{bot}

USGS

Integration and Application Network (ian.umces.edu/symbols),
University of Maryland Center for Environmental Science.

ATM interactions





SST **OCN**

Surface fluxes

Momentum

Heat

Moisture

$$F_m = C_m |\vec{V}_{SL}|^2 ,$$
$$F_h = \rho_1 c_p C_{hq} (\theta_{sk} - \theta_1) ,$$
$$F_q = \rho_1 L C_{hq} M (q_{vsk} - q_{v1}) .$$

$$|\vec{V}_{SL}|^2 = u^2 + v^2$$

C_m is the exchange coefficient for momentum and is expressed as

$$C_m = \frac{u_*^2}{|\vec{V}_{SL}|^2}.$$

C_{hq} is the exchange coefficient valid for both heat and water vapor as

$$C_{hq} = u_* k \left[\psi_h \left(\frac{z}{L_{MO}} \right) - \psi_h \left(\frac{z_{0T}}{L_{MO}} \right) + \ln \left(\frac{z}{z_{0T}} \right) \right]^{-1},$$

u_* is the friction velocity and is expressed as

$$u_* = \kappa |\vec{V}_{SL}| \left[\psi_m \left(\frac{z}{L_{MO}} \right) - \psi_m \left(\frac{z_{0m}}{L_{MO}} \right) + \ln \left(\frac{z}{z_{0m}} \right) \right]^{-1}$$

z_{0m} = f (H_{wave}, L_{pwave}, T_{psurf})

WAV

OCEAN SURFACE ROUGHNESS CLOSURE MODELS

H_s = significant wave height

z_0 = ocean surface roughness

u_* = wind friction velocity

C_p = peak wave celerity

L_p = peak wave length

$\frac{u_*}{C_p}$ = wave age

CHARNOCK 1955

$$z_{0m} = \frac{0.011(u_*)^2}{g}$$

TAYLOR & YELLAND 2001: TY2001

$$\frac{z_{0m}}{H_s} = 1200\left(H_s/L_p\right)^{4.5}$$

- *Wave steepness based parameterization.*
- *Based on three datasets representing sea-state conditions ranging from strongly forced to shoaling.*

DRENNAN 2003: DGQH

$$\frac{z_{0m}}{H_s} = 3.35\left(u_*/C_p\right)^{3.4}$$

- *Wave age based formula to characterize the ocean roughness.*
- *They combined data from many field experiments representing a variety of condition and grouped the data as a function of the wind friction velocity.*

OOST 2002: OOST

$$\frac{z_{0m}}{L_p} = \frac{25.0}{\pi}\left(u_*/C_p\right)^{4.5}$$

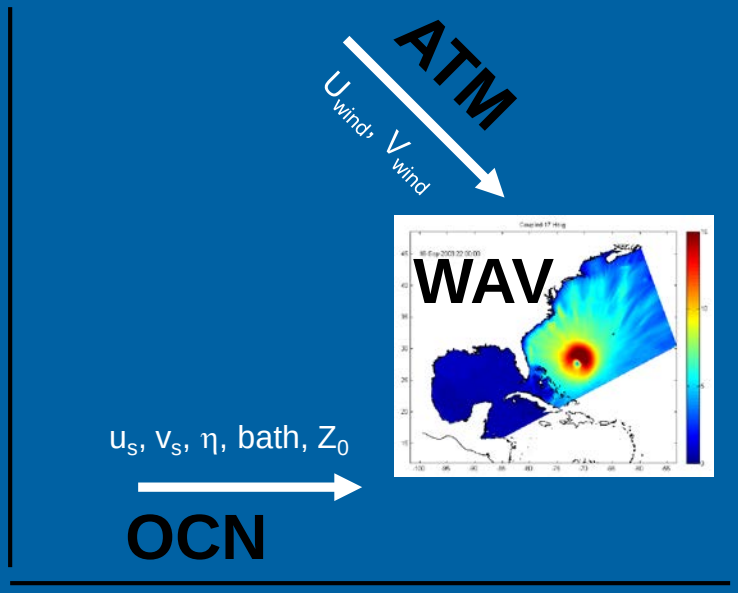
- *Wave age dependent formula but it also considers the effect of the wave steepness.*

WAV interactions

$$\frac{\partial N}{\partial t}+\frac{\partial c_x N}{\partial x}+\frac{\partial c_y N}{\partial y}+\frac{\partial c_\sigma N}{\partial \sigma}+\frac{\partial c_\theta N}{\partial \theta}=\frac{S_w}{\sigma}$$

1) **Generation** – wind speed forcing is modified by ocean currents:

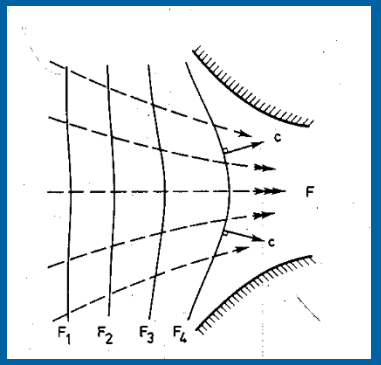
$S(w) = f(U_{wind} - u_s \; ; V_{wind} - v_s \;)$



2) **Propagation**

– wave celerity in geographic space is modified by ocean currents

$c_x = c_{gx} + u_s \; ; c_y = c_{gy} + v_s$



– change of wave direction (refraction) due to η, bathy, and currents:

$$C_{g,\theta}=\frac{\sigma}{\sinh(2kh)}\left(\frac{\partial h}{\partial x}\sin\theta-\frac{\partial h}{\partial y}\cos\theta\right)+\cos\theta\left(\frac{\partial U}{\partial x}\sin\theta-\frac{\partial U}{\partial y}\cos\theta\right)+\sin\theta\left(\frac{\partial V}{\partial x}\sin\theta-\frac{\partial V}{\partial y}\cos\theta\right)$$

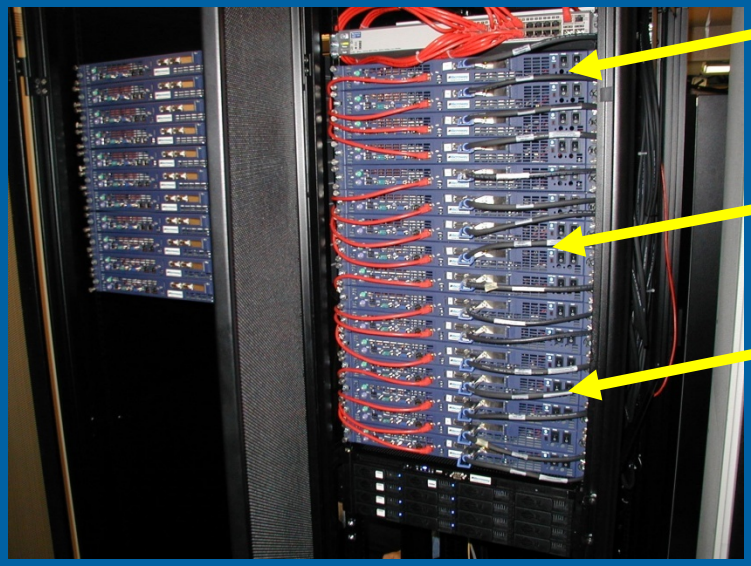
Coupled Modeling System

MPI が必要
WRF ビルドの前に MCT のインストールは完了しているはず

Model Coupling Toolkit

Mathematics and Computer Science Division Argonne National Laboratory
http://www-unix.mcs.anl.gov/mct/

MCT is an open-source package that provides MPI based communications between all nodes of a distributed memory modeling component system. Download and compile as libraries that are linked to.



Model A running on M nodes.

Model B running on N nodes.

Model C ……

……

MCT provides communications between all models.

(it also works here)

USGS

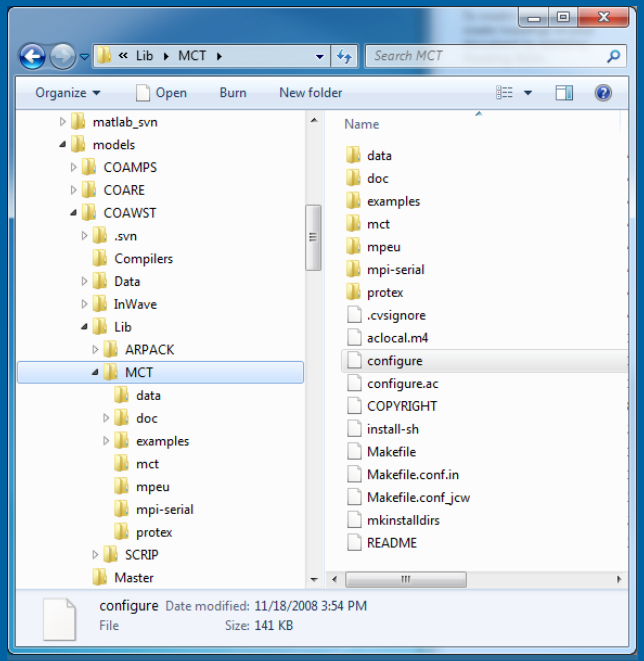
Warner, J.C., Perlin, N., and Skyllingstad, E. (2008). Using the Model Coupling Toolkit to couple earth system models. Environmental Modeling and Software

1:55:20

Libraries

/Lib/MCT/

- **MCT - v2.60 or higher (distributed)**



1) cd to the MCT dir

2) ./configure
This makes Makefile.conf. you can edit this file.

3) make

4) make install

5) set environment vars
setenv MCT_INCDIR COAWST/Lib/MCT/include
setenv MCT_LIBDIR COAWST/Lib/MCT/lib
(or where ever you installed them, see last slide)

USGS

Compilers dir (side note)

emacs: Linux-pgi.mk

File Edit View Cmds Tools Options Buffers Makefile Help

Open Dired Save Print Cut Copy Paste Undo Spell Replace Mail Info Compile Debug News

Linux-pgi.mk

```
#
# Library locations, can be overridden by environment variables.
#

ifdef USE_NETCDF4
    NETCDF_INCDIR ?= /opt/pgisoft/netcdf4/include
    NETCDF_LIBDIR ?= /opt/pgisoft/netcdf4/lib
      HDF5_LIBDIR ?= /opt/pgisoft/hdf5/lib
else
    NETCDF_INCDIR ?= /opt/pgisoft/netcdf/include
    NETCDF_LIBDIR ?= /opt/pgisoft/netcdf/lib
endif
             LIBS := -L$(NETCDF_LIBDIR) -lnetcdf -lnetcdff -L/opt/mx/lib64 -lcurl -lgssapi_krb5
             LIBS += $(shell nc-config --libs)
ifdef USE_NETCDF4
             LIBS += -L$(HDF5_LIBDIR) -lhdf5_hl -lhdf5 -lz
endif

ifdef USE_MCT
       MCT_INCDIR ?= /opt/pgisoft/mct/include
       MCT_LIBDIR ?= /opt/pgisoft/mct/lib
           FFLAGS += -I$(MCT_INCDIR)
             LIBS += -L$(MCT_LIBDIR) -lmct -lmpeu
endif

ifdef USE_ARPACK
 ifdef USE_MPI
   PARPACK_LIBDIR ?= /opt/pgisoft/PARPACK
             LIBS += -L$(PARPACK_LIBDIR) -lparpack
 endif
    ARPACK_LIBDIR ?= /opt/pgisoft/PARPACK
             LIBS += -L$(ARPACK_LIBDIR) -larpack
endif

ifdef USE_MPI
          CPPFLAGS += -DMPI
 ifdef USE_MPIF90
               FC := mpif90
 else
             LIBS += -Bdynamic -lfmpi-pgi -lmpi-pgi -Bstatic
 endif
endif
```

:= は定義

+= は追加

?= は環境変数に定義されていない場合に用いる定義

ISO8--**-XEmacs: Linux-pgi.mk (Makefile)----L74--C0--27%----------

?= means that it checks your environment first

Model organization

mpi_init

init_file (# procs/model)

SWAN { init, run, finalize }

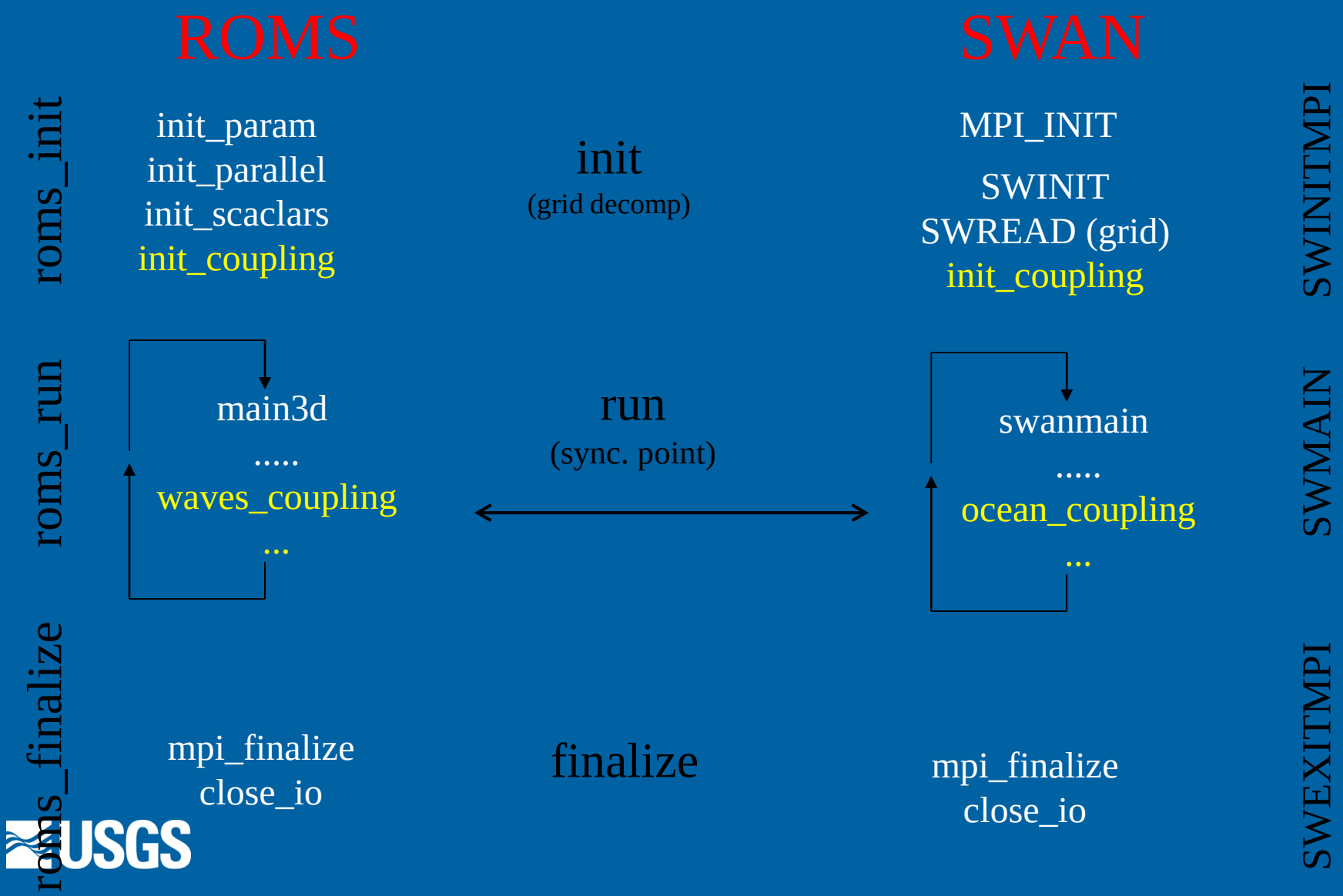
ROMS { init, run, finalize }

master.F

```
master.F - XE
File  Edit  View  C                                                    Help
Open  Dired  Save  Print  Cut  Copy  Paste  Undo  Spell  Replace  Mail  Info  Compile  Debug  News
waves coupler.F  master.F

!
!  Initialize MPI execution environment.                    coawst/Master/master.F 中身なし
!
      CALL mpi_init (MyError)                               このfile はどこにある？
!                                                                  ↓
!  Get rank of the local process in the group associated with the
!  communicator.                                            coawst/Master/mct_driver.h
!
      CALL mpi_comm_size (MPI_COMM_WORLD, nprocs, MyError)
      CALL mpi_comm_rank (MPI_COMM_WORLD, MyRank, MyError)
!
!  Read in waves-ocean coupling parameters from standard input.
!  Set temporarily the ocean communicator to current handle before
!  splitting so the input coupling script name can be broadcasted to
!  all the nodes.
!
      OCN_COMM_WORLD=MPI_COMM_WORLD

      CALL read_CouplePar (iNLM)
!
!  Split the communicator into SWAN and ROMS subgroups based on color
!  and key.
!
      MyKey=0
      IF (MyRank.le.peWAV_last) THEN
        MyColor=1                                                ! 1=SWAN
        MyString="COMPONENT_ID=1,COMPONENT_NAME=swan"
      ELSE
        MyColor=2                                                ! 2=ROMS
        MyString="COMPONENT_ID=2,COMPONENT_NAME=roms"
      END IF
      CALL mpi_comm_split (MPI_COMM_WORLD, MyColor, MyKey, MyCOMM,      &
     &                     MyError)
!
!-----------------------------------------------------------------------
!  Run either SWAN or ROMS according to the processor rank.
!-----------------------------------------------------------------------
!
      IF (MyRank.le.peWAV_last) THEN
        CALL SWINITMPI (MyCOMM)
        CALL SWMAIN (nWAV_steps)
        CALL SWEXITMPI
      ELSE
        first=.TRUE.
        Nrun=1
        IF (exit_flag.eq.NoError) THEN
          CALL roms_init (first, MyCOMM)
        END IF
        IF (exit_flag.eq.NoError) THEN
          CALL roms_run
        END IF
          CALL roms_finalize
      END IF
!
!-----------------------------------------------------------------------
!  Terminates all the MPI processing.
!-----------------------------------------------------------------------
!
      CALL mpi_finalize (MyError)

      END PROGRAM waves_ocean
Raw-----XEmacs: master.F      (Fortran Font)----L84--C0--Bot----------
Font Courier New:Regular:10::Western
```

USGS

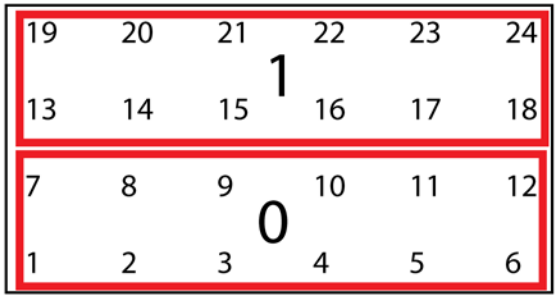
init, run, and finalize



USGS

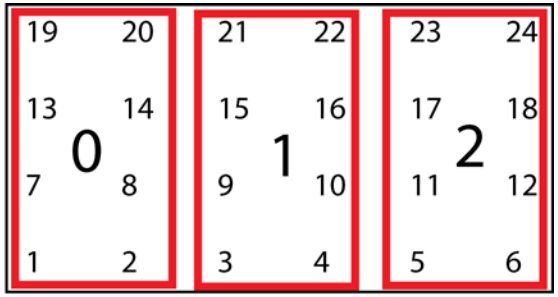
Grid decomposition (during initialization)

ROMS



ROMS : example grid tiling
NTILEI = 1, NTILEJ = 2
tile0: start(1) = 1, length(1) = 6
start(2) = 7, length(2) = 6
tile1: start(1) = 13, length(1) = 6
start(2) = 19 length(2) = 6

SWAN



SWAN : example grid tiling
NTILE = 3
tile0: start(1) = 1, length(1) = 2
start(2) = 7, length(2) = 2
start(3) = 13, length(3) = 2
start(4) = 19, length(4) = 2
tile1: start(1) = 3, length(1) = 2
start(2) = 9, length(2) = 2
start(3) = 15, length(3) = 2
start(4) = 21, length(4) = 2
tile0: start(1) = 5, length(1) = 2
start(2) = 11, length(2) = 2
start(3) = 17, length(3) = 2
start(4) = 23, length(4) = 2

Figure 1. Example of how SWAN and ROMS decompose the same grid. SWAN decomposition is determined by grid orientation whereas ROMS decompomposition is user specifed.

-Each tile is on a separate processor.

-Each tile registers with MCT.

Determine what information to exchange

ROMS- init_coupling

processed by each ROMS tile

waves_coupler.F - XEmacs

File Edit View Cmds Tools Options Buffers Help

waves_coupler.F | master.F

```
!
! Determine start and lengths for domain decomposition.
!
      Jsize=JendR-JstrR+1
      allocate(start(Jsize))
      allocate(length(Jsize))
      indx=0
      DO j=JstrR,JendR
        indx=indx+1
        length(indx)=(IendR-IstrR+1)
        start(indx)=j*(Lm(ng)+2)+IstrR+1
      END DO

1     CALL GlobalSegMap_init(GSMapROMS,start,length,0,OCN_COMM_WORLD,    &
     &                       OcnId)
!
! Initialize Attribute vectors.
!
      ! size is the number of grid point on this processor
      gsmsize = MCT_GSMap_lsize(GSMapROMS,OCN_COMM_WORLD)

2     CALL MCT_AtrVt_init(FromWavesAV,rList=                             &
     &"DISSIP:FORCEX:FORCEY:HSIGN:RTP:SETUP:TMBOT:UBOT:DIR:WLEN:TM01:XP"&
     &":YP:TM01",lsize=gsmsize)

      ! initialize oceanAv with one real attribute for now.
      call MCT_AtrVt_init(ToWavesAV,rList="DEPTH:WLEV:VELX:VELY",        &
     &                    lsize=gsmsize)
!
      itialize a router to the Waves component.

3     call MCT_Router_init(WavId,GSMapROMS,OCN_COMM_WORLD,RoutROMS)
```

Raw-----XEmacs: waves_coupler.F (Fortran Font)----L165--C0--21%----------

Font Courier New:Regular:16::Western

SWAN- init_coupling

processed by each SWAN tile

swancplr.ftn - XEmacs

File Edit View Cmds Tools Options Buffers Help

swancplr.ftn

```
!
      DO j = 1,Jsize
        length(j) = ISIZE
        IF (MXCGL.gt.MYCGL) THEN
          IF (MyRank.eq.0) THEN
            start(j) = MXF + (j-1)*MXCGL
          ELSE
            start(j) = MXF + (j-1)*MXCGL + IHALOX
          ENDIF
        ELSE
          IF (MyRank.eq.0) THEN
            start(j) = MYF + (j-1)*MXCGL
          ELSE
            start(j) = (MYF + IHALOY - 1)*MXCGL + 1 + (j-1)*MXCGL
          ENDIF
        ENDIF
      END DO
!
1     gsmsize = Isize*Jsize

      CALL GlobalSegMap_init(GSMapSWAN,start,length,0,                   &
     &                       WAV_COMM_WORLD,WavId)
!
! Initialize Attribute Vector ToOceanAv to hold the data sent to ROMS.
!
2     CALL AttrVect_init(ToOceanAV,rlist=                                &
     &"DISSIP:FORCEX:FORCEY:HSIGN:RTP:SETUP:TMBOT:UBOT:DIR:WLEN:TM01:XP"&
     &":YP:TM01",lsize=gsmsize)
! Initialize Attribute Vector FromOceanAV that will have ROMS data in it.
!
      CALL AttrVect_init(FromOceanAV,rList="DEPTH:WLEV:VELX:VELY",       &
     &                   lsize=gsmsize)
!
! Initialize a router to the Waves component.

3     call Router_init(OcnId,GSMapSWAN,WAV_COMM_WORLD,RoutSWAN)
```

Raw-----XEmacs: swancplr.ftn (Fundamental)----L141--C0--21%----------

Wrote C:\data\presentations\2006_04_24_WISE\swancplr.ftn

USGS

Synchronization (run phase)

ROMS- ocean_coupling

waves_coupler.F - XEmacs

File Edit View Cmds Tools Options Buffers Help

waves_coupler.F | master.F

```
!
!-----------------------------------------------------------------------
!  Receive data from SWAN.
!-----------------------------------------------------------------------
!
      CALL MCT_Recv(FromWavesAV, RoutROMS, MyError)
!
! Fill local variables with received data.
!
      indx=0
      DO j=JstrR,JendR
        DO i=IstrR,IendR
          indx=indx+1
          Wave_dissip(i,j)  = FromWavesAV%rAttr(2,indx)
          Hwave(i,j)        = FromWavesAV%rAttr(5,indx)
.................( + other vars)

# if defined EW_PERIODIC || defined NS_PERIODIC || defined DISTRIBUTE
        CALL exchange_r2d_tile (ng, iNLM, Istr, Iend, Jstr, Jend,        &
     &                          LBi, UBi, LBj, UBj,                      &
     &                          NghostPoints,                            &
     &                          Wave_dissip(:,:))
# endif
!-----------------------------------------------------------------------
!  Send depth, water level, and velocity data to SWAN.
!-----------------------------------------------------------------------

!
! Fill attribute vector.
!
      indx=0
      DO j=JstrR,JendR
        DO i=IstrR,IendR
          indx=indx+1
          testarray(indx)=hwater(i,j)
        END DO
      END DO
      call MCT_AtrVt_importRA(ToWavesAV,"DEPTH",testarray)
.................. (+ other vars)
!
! Send the data
!
      CALL MCT_SEND(ToWavesAV, RoutROMS, MyError)

      write(*,*) '++ ROMS sent ocean fields and Myerror??= ', MyError
```

Raw-----XEmacs: waves_coupler.F (Fortran Font)----L725--C0--Bot--------

processed by each ROMS tile

MCT

MCT

SWAN- waves_coupling

swancplr.ftn - XEmacs

File Edit View Cmds Tools Options Buffers Help

swancplr.ftn

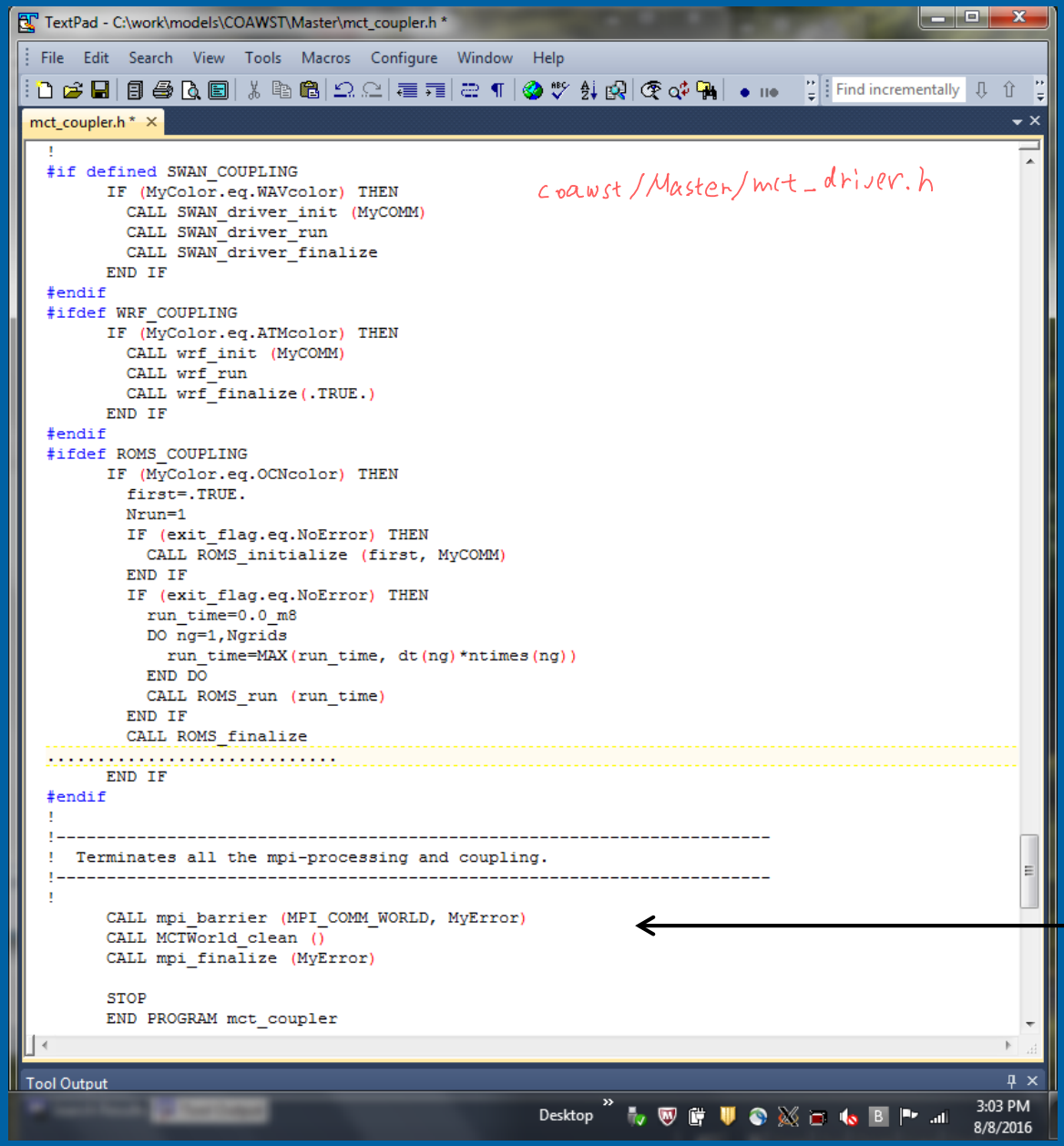
```
!-----------------------------------------------------------------------
!  Fill attribute vectors.
!-----------------------------------------------------------------------

! Get output variable id.
      IVTYPE = IVTYP(1)
!ask for points in this tile
      CALL GlobalSegMap_Ordpnts(GSMapSWAN,MyRank,points)
      DO IP=1,gsmsize
        avdata(IP) = VOQ(points(IP),VOQR(IVTYPE))
      END DO
      IF (IVTYPE.eq.4) THEN
        CALL AttrVect_importRAttr(ToOceanAV,"DEPTH",avdata)
      END IF
      IF (IVTYPE.eq.7) THEN
        CALL AttrVect_importRAttr(ToOceanAV,"DISSIP",avdata)
      END IF
.................( + other vars)

!-----------------------------------------------------------------------
!  Send wave parameters to ROMS.
!-----------------------------------------------------------------------
!
      CALL MCT_SEND(ToOceanAV, RoutSWAN, MyError)
!
!-----------------------------------------------------------------------
!  Receive from ROMS: Water Level, Depth, VELX, and VELY.
!-----------------------------------------------------------------------
!
      CALL MCT_Recv(FromOceanAV, RoutSWAN, MyError)

      IP=0
      DO IY = Jstr, Jend
        DO IX = Istr, Iend
          IP=IP+1
          INDX = KGRPNT(IX,IY)
          IF (INDX.GT.1) THEN
!
! insert water levels from ROMS into SWAN wlev array.
!
            COMPDA(INDX,JWLV2)=FromOceanAV%rAttr(2,IP)
!
! insert velx and vely from ROMS into SWAN velocity arrays.
!
            COMPDA(INDX,JVX2)=FromOceanAV%rAttr(3,IP)
            COMPDA(INDX,JVY2)=FromOceanAV%rAttr(4,IP)
          END IF
```

Raw-----XEmacs: swancplr.ftn (Fundamental)----L560--C23--92%---------

Font Courier New:Regular:14::Western

processed by each SWAN tile

End phase

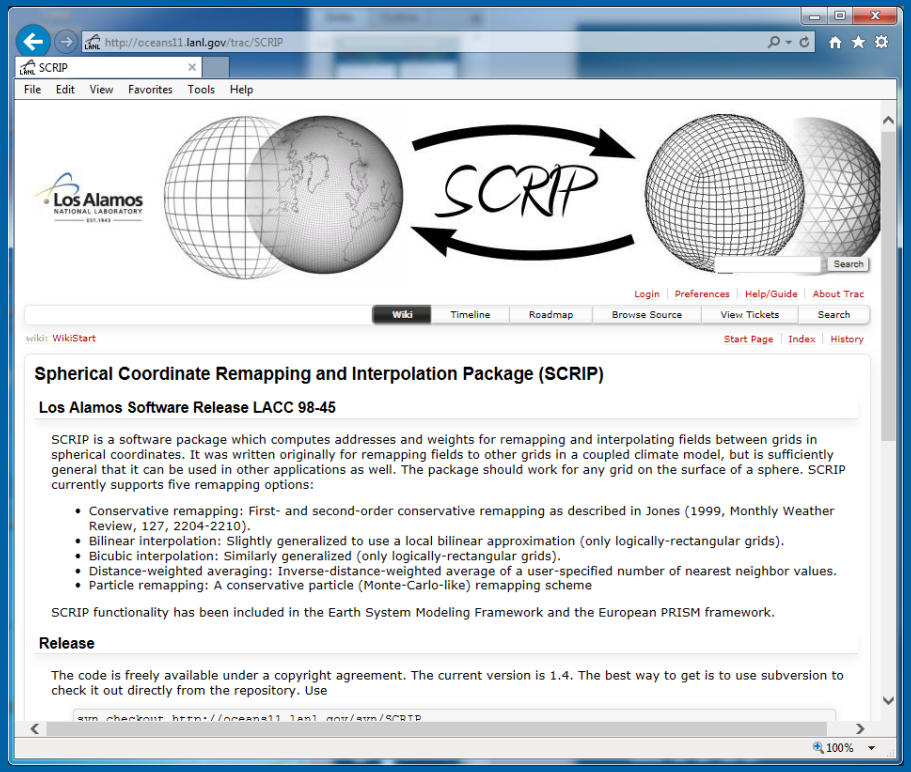


~~"master.F"~~

MPI barrier prevents program complete exit until all models arrive at this point.

Sometimes one model may die and other model is waiting.

SCRIP grid interpolation

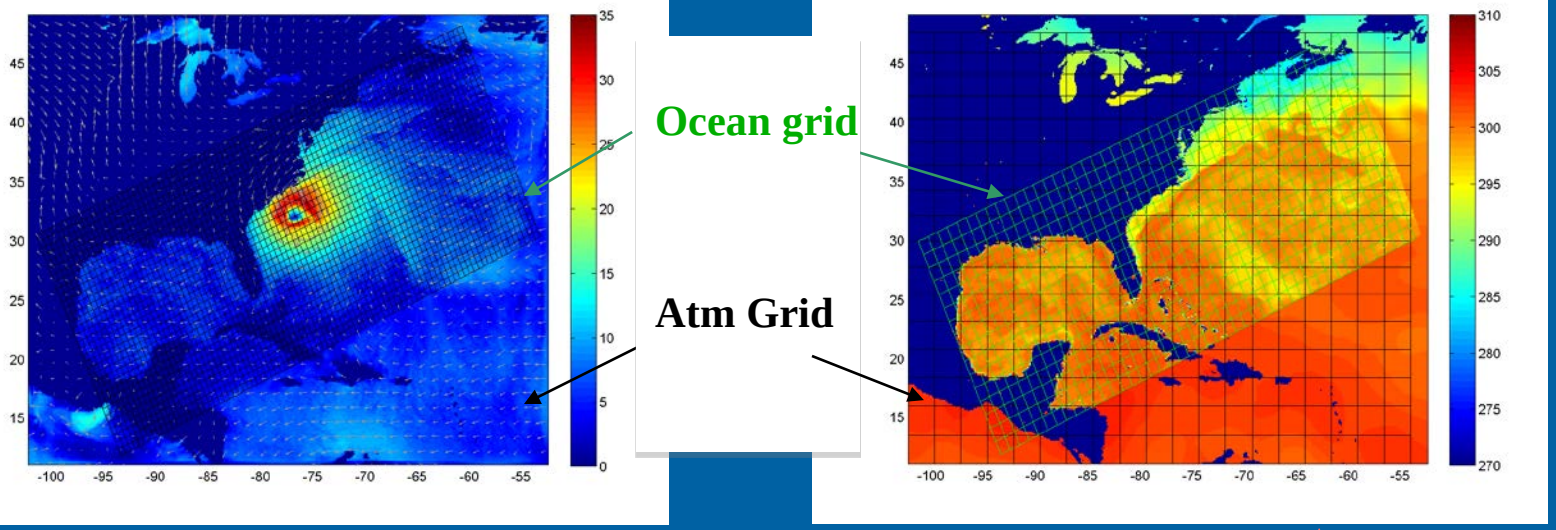


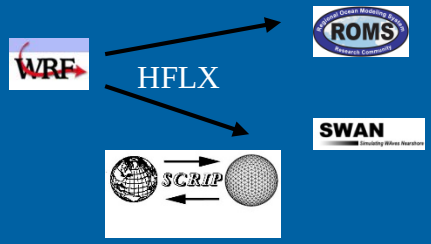
http://oceans11.lanl.gov/trac/SCRIP

- Needed when you run more than 1 model and the models are on different grids.
- We started with SCRIP from LANL.
- We adapted the code to read model grids, and write out one NetCDF file with all the interpolation weights.
- We use the ‘group’ capability in NetCDF4.

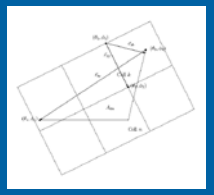
USGS

3) SCRIP - grid interpolation

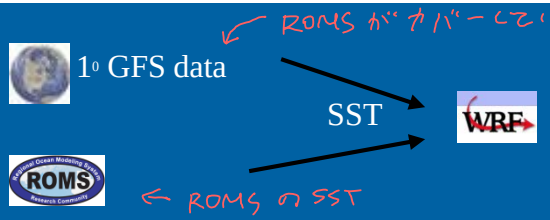




Atmosphere model provides heat flux to cover entire ocean grid. SCRIP interpolations weights needed to remap data fields.



Flux conservative remapping scheme



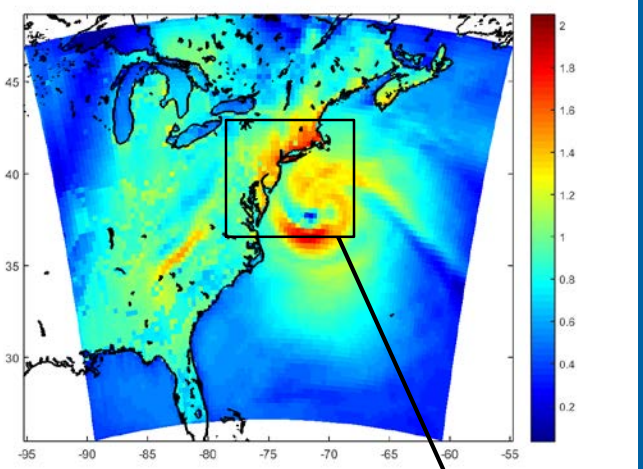
Ocean model provides higher resolution and coupled response of SST to atmosphere. But the ocean grid(s) are limited in spatial coverage so atmosphere model must combine data from different sources.

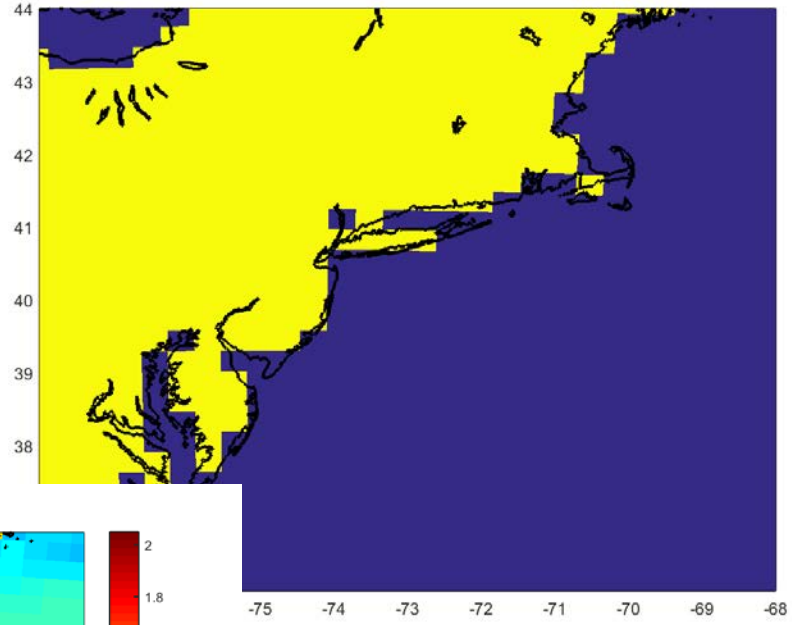
USGS

Two main issues:

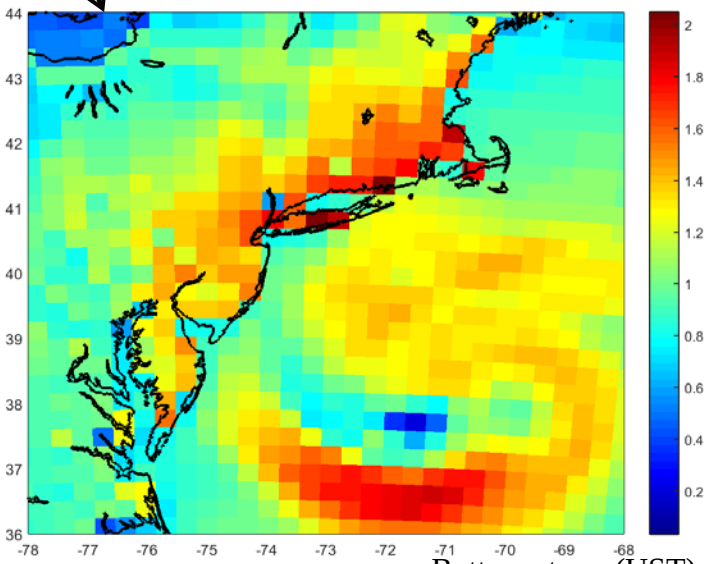
1) **Interpolate data between different grids.**
2) **Combine data from different sources.**

Land masking effects – example of grid interpolation





LANDMASK



Bottom stress (UST)

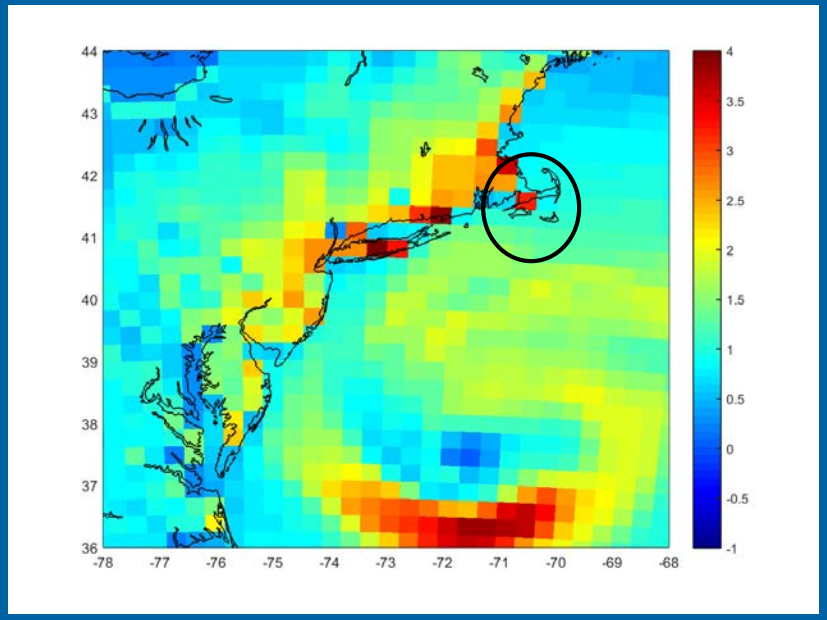
This is a coarse example to demonstrate the effects of land/sea masking on atmospheric processes.

Large bottom stress over land will cause problems on an ocean grid with different masking.

USGS

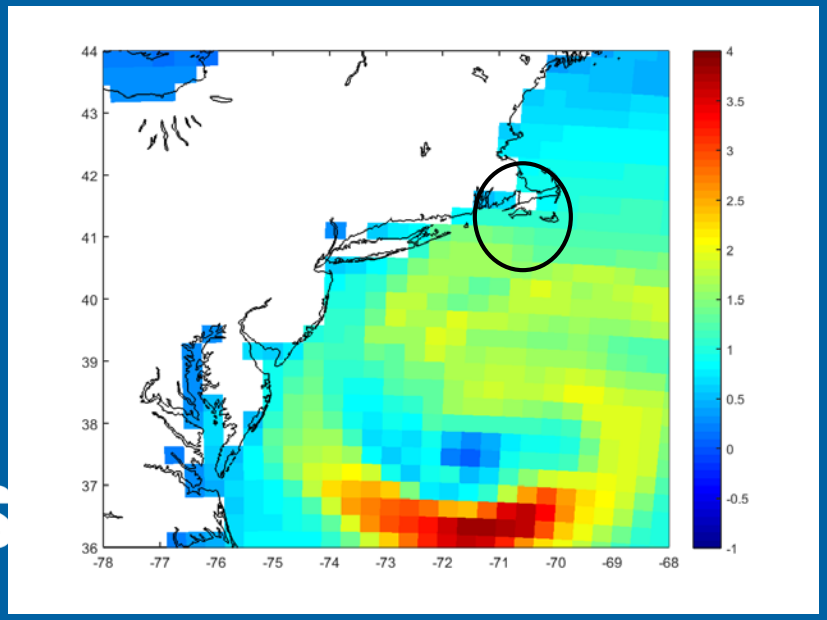
Grid interpolation

WRF
UST^2 over
land + ocean

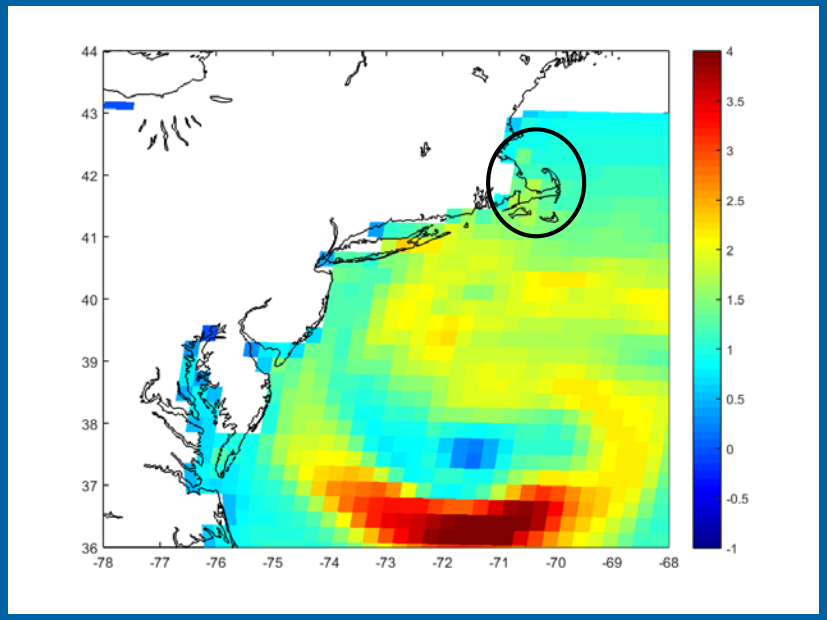


ATM land mask areas do not effect ocean.

WRF
UST^2 over
ocean only

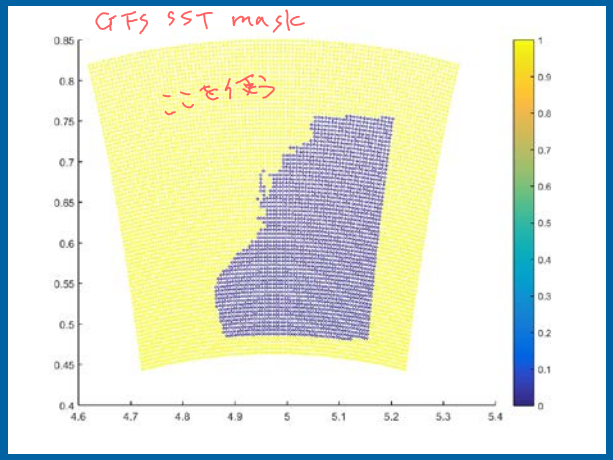


ROMS
su/vstr over
ocean only

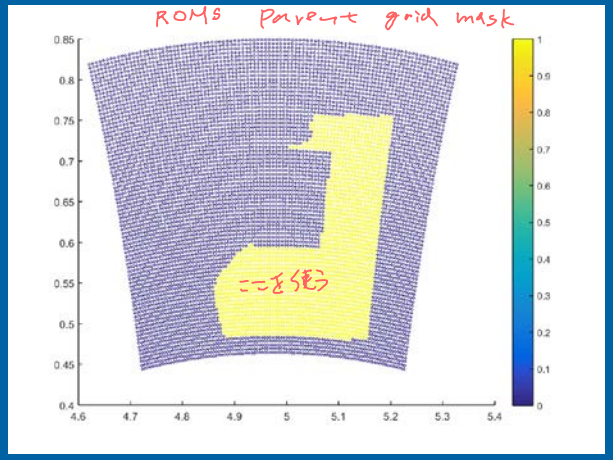


Combining multiple sources

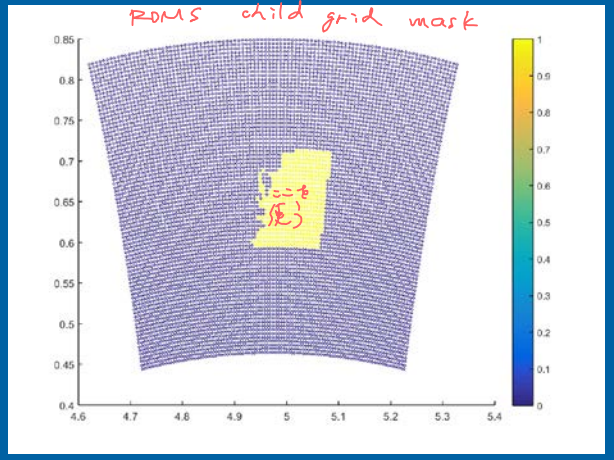
Sum of all 3 masks = 1



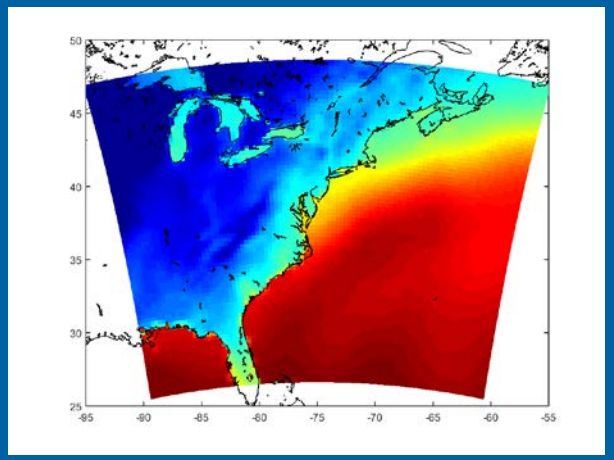
Background GFS SST mask



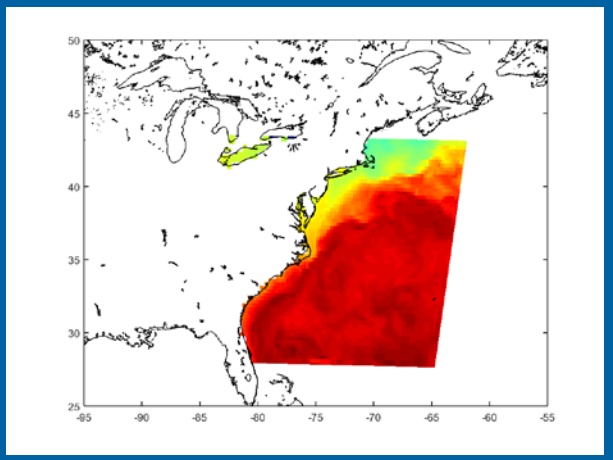
ocn 1 to atm 1 mask



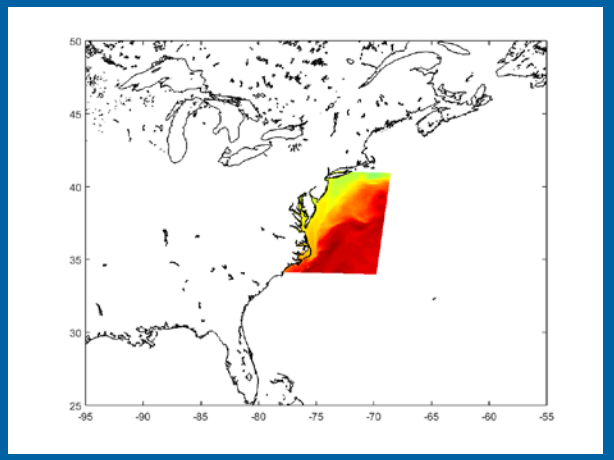
ocn 2 to atm 1 mask



Background GFS SST



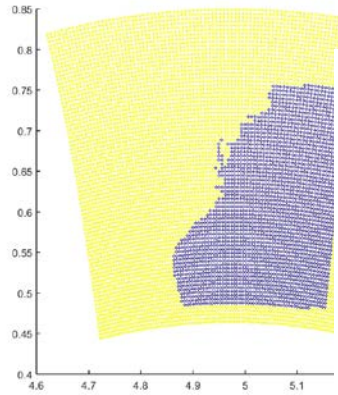
ocn 1 SST



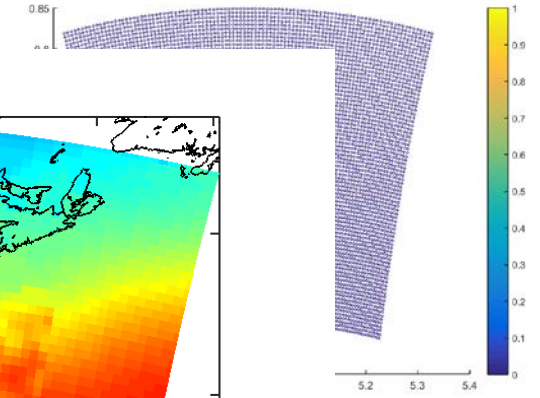
ocn 2 SST

Combining multiple sources

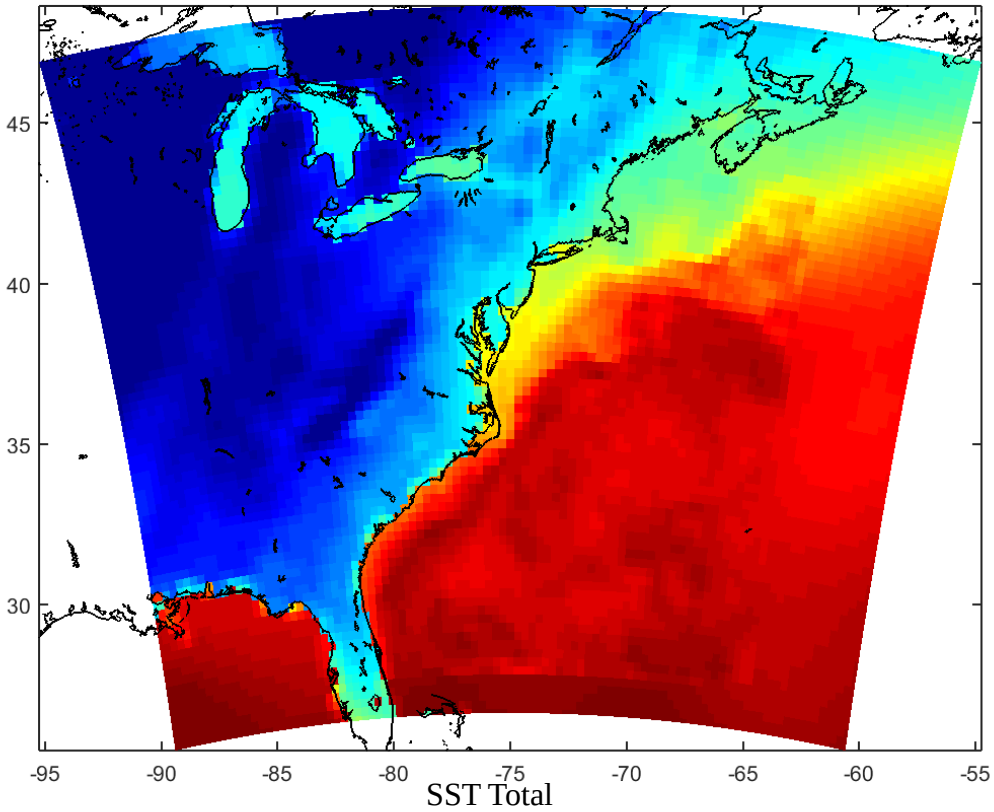
Sum of all 3 masks = 1



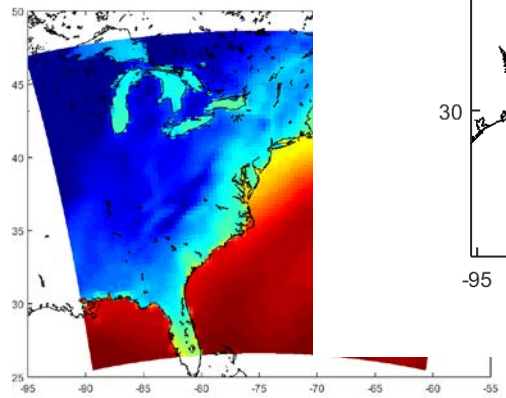
Background GFS



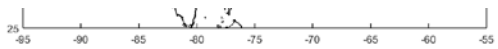
1 mask



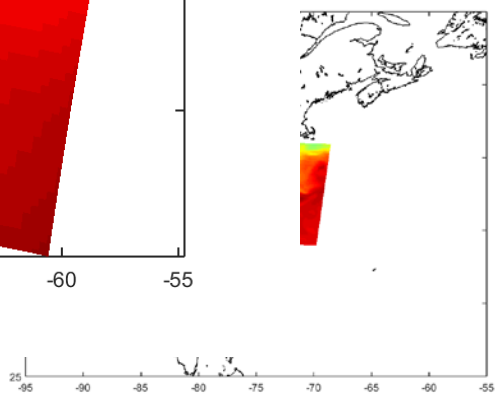
SST Total



Background GFS SST



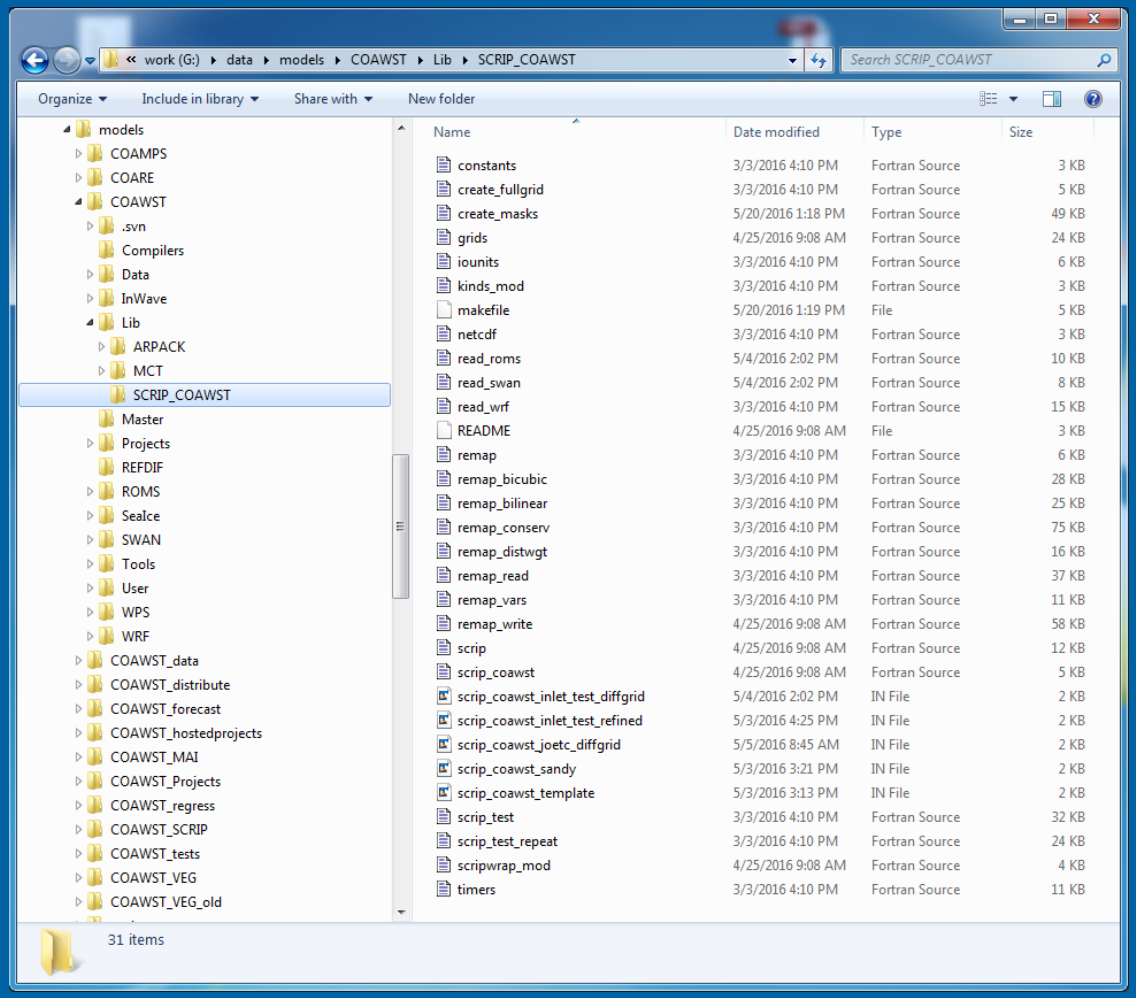
ocn 1 SST



ocn 2 SST

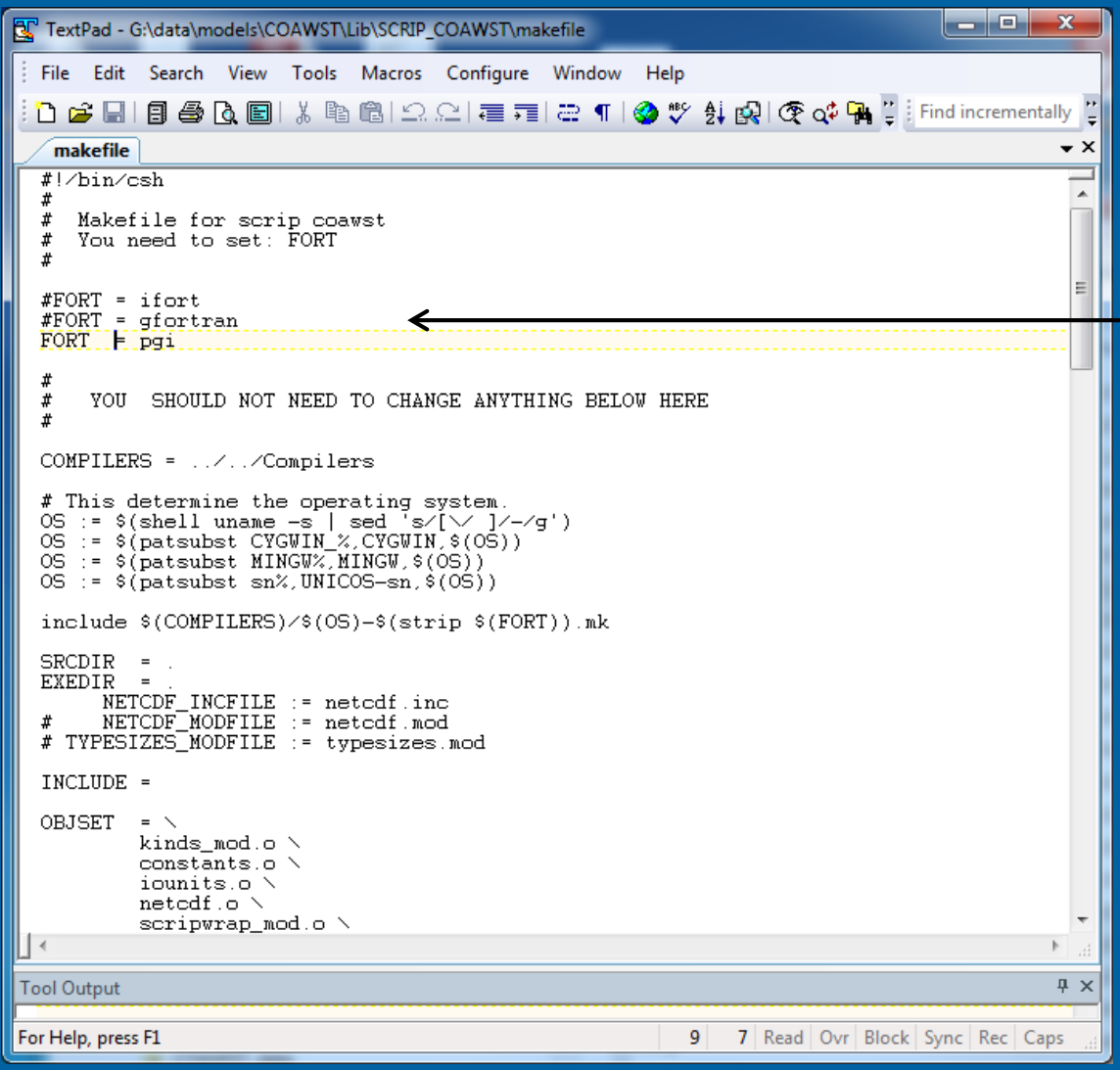
SCRIP_COAWST

– based on SCRIP v 1.6 and modified to read scrip_coawst.in



SCRIP_COAWST

1) cd to COAWST/Lib/SCRIP_COAWST
2) edit makefile
3) make



Set your compiler here

This will build the program "scrip_coawst"

SCRIP_COAWST – input file

scrip_coawst_template.in

```
$INPUTS
!
!  Input file for scrip_coawst.
!  The $INPUTS line is required at the top of this file.
!  Edit this file to enter the correct information below.
!  Then run this program as "scrip_coawst scrip_coawst.in"
!
! 1) Enter name of output netcdf4 file
!
OUTPUT_NCFILE='scrip_file.nc'

! 2) Enter total number of ROMS, SWAN, and WRF (max_dom) grids:
!
NGRIDS_ROMS=2,
NGRIDS_SWAN=2,
NGRIDS_WRF=2,

! 3) Enter name of the ROMS grid file(s):
!
ROMS_GRIDS(1)='roms_grid1.nc',
ROMS_GRIDS(2)='roms_grid2.nc',

! 4) Enter SWAN information:
!    -the name(s) of the SWAN grid file(s) for coords and bathy.
!    -the size of the SWAN grids, and
!    -if the swan grids are Spherical(set cartesian=0) or
!                           Cartesian(set cartesian=1).
!
SWAN_COORD(1)='swan_coord1.grd',
SWAN_COORD(2)='swan_coord2.grd',
SWAN_BATH(1)='swan_bathy1.bot',
SWAN_BATH(2)='swan_bathy2.bot',
SWAN_NUMX(1)=84,
SWAN_NUMX(2)=116,
SWAN_NUMY(1)=64,
SWAN_NUMY(2)=86,
CARTESIAN(1)=0,
CARTESIAN(2)=0,
```

Output file name

Number of grids for each model

ROMS grid names

SWAN grid names

SCRIP_COAWST – input file

```
! 4) Enter SWAN information:
!    -the name(s) of the SWAN grid file(s) for coords and bathy.
!    -the size of the SWAN grids, and
!    -if the swan grids are Spherical(set cartesian=0) or
!                           Cartesian(set cartesian=1).
!
SWAN_COORD(1)='swan_coord1.grd',
SWAN_COORD(2)='swan_coord2.grd',
SWAN_BATH(1)='swan_bathy1.bot',
SWAN_BATH(2)='swan_bathy2.bot',
SWAN_NUMX(1)=84,
SWAN_NUMX(2)=116,
SWAN_NUMY(1)=64,
SWAN_NUMY(2)=86,
CARTESIAN(1)=0,
CARTESIAN(2)=0,

! 5) Enter the name of the WRF input grid(s). If the grid is a
!    moving child nest then enter that grid name as 'moving'.
!    Also provide the grid ratio, this is used for a moving nest.
!
WRF_GRIDS(1)='wrfinput_d01',
WRF_GRIDS(2)='moving',
PARENT_GRID_RATIO(1)=1,
PARENT_GRID_RATIO(2)=3,
PARENT_ID(1)=0
PARENT_ID(2)=1

!
!  The $END statement below is required
!
$END
```

SWAN grid names

WRF grid names

Enter ‘moving’ for a moving nest.
Only the last WRF grid can be moving.

SCRIP file(s)

Run the scrip coawst program using
./scrip_coawst scrip_coawst.in

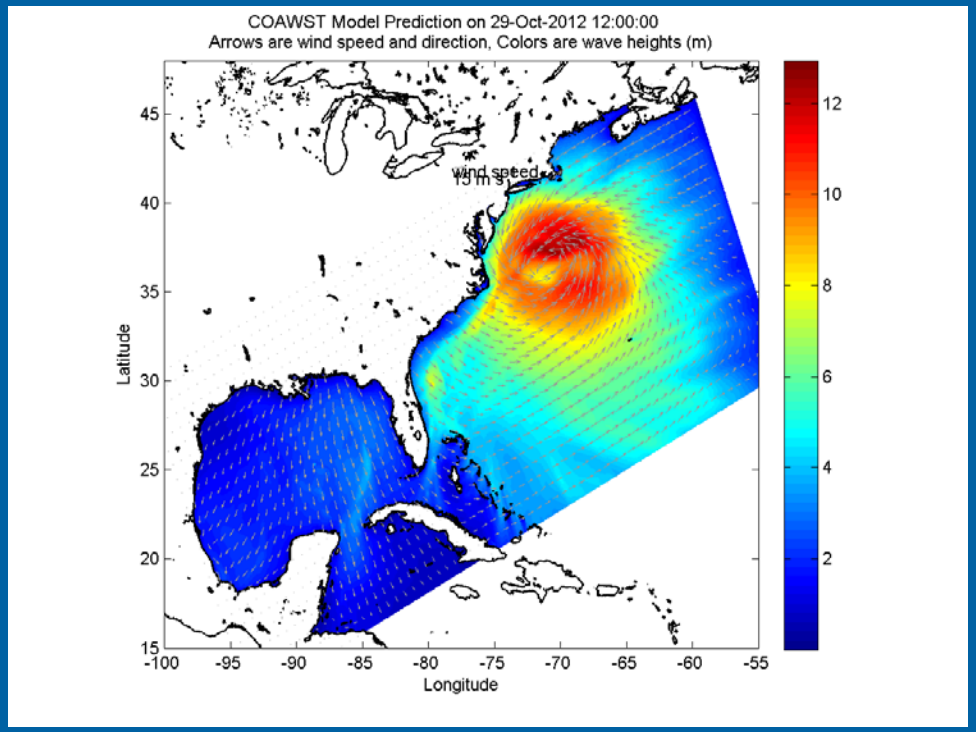
We use NetCDF 4 functionality to create only 1 netcdf file using the ‘group’ functionality.

To read the data in matlab you can use (for example)
dst_grid2=ncread('scrip_sandy_static.nc','/ocn2_to_atm1_weights.nc/dst_grid_imask');

The old way: created a weight file between each grid.
So if you have 1 WRF grid, and 2 ROMS grids, it will produce
atm1_to_ocn1_weights.nc
atm1_to_ocn2_weights.nc
ocn1_to_atm1_weights.nc
ocn2_to_atm1_weights.nc

USGS

Setting up a coupled application: Projects/Sandy



USGS

How to create coupled application

- 1) Create all input, BC, init, forcing, etc files for each model as if running separately. I recommend that you run each model separately first.
- 2) SCRIP (if different grids or grid refinement)
- 3) modify cppdefs in your header file.
- 4) ocean.in, namelist.in, and coupling.in
  ↑ROMS ↑WRF
- 5) coawst.bash
- 6) run it as coawstM

USGS

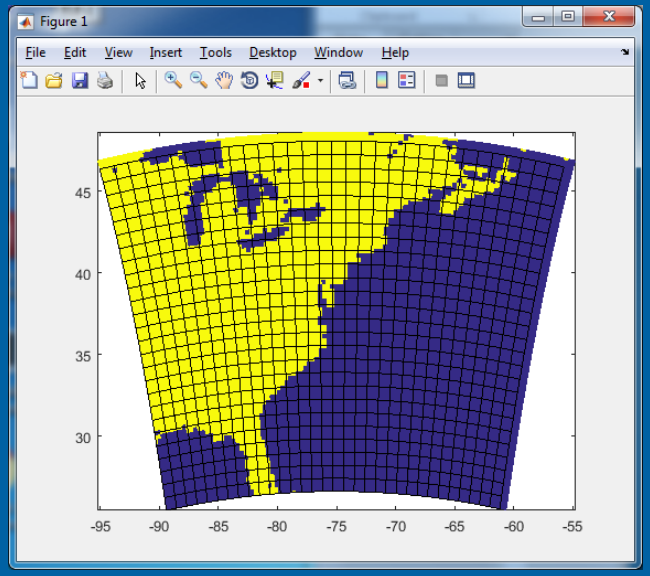
1) Use each model separately

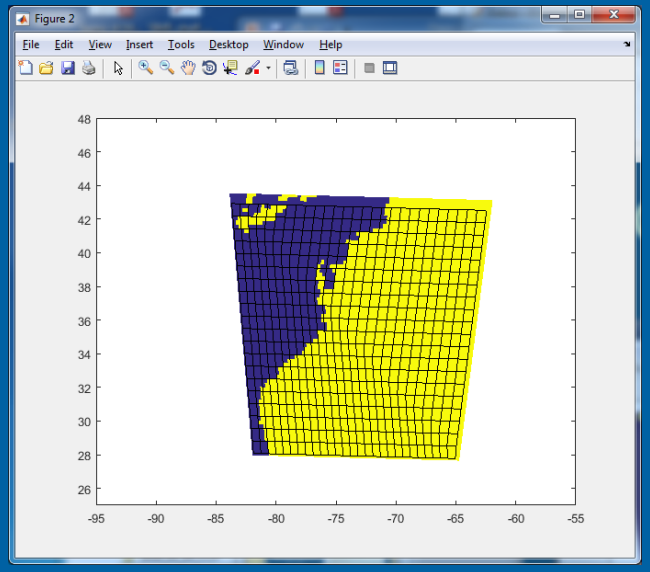
WRF

- 27 vertical levels
- dt 36 s
- Physics
  - Lin microphysics
  - RRTM longwave, Dudhia shortwave
  - Mellor-Yamada-Janjic (MYJ) PBL
  - .....

ROMS

- 16 vertical levels
- dt 240, 48
- Physics
  - GLS turbulence closure
  - COARE bulk fluxes
  - BC's from HYCOM
  - .....





2) SCRIP

Edit the SCRIP input file:
COAWST/Lib/SCRIP_COAWST/scrip_coawst_sandy.in

TextPad - F:\data\models\COAWST\Lib\SCRIP_COAWST\scrip_coawst_sandy.in *

File Edit Search View Tools Macros Configure Window Help

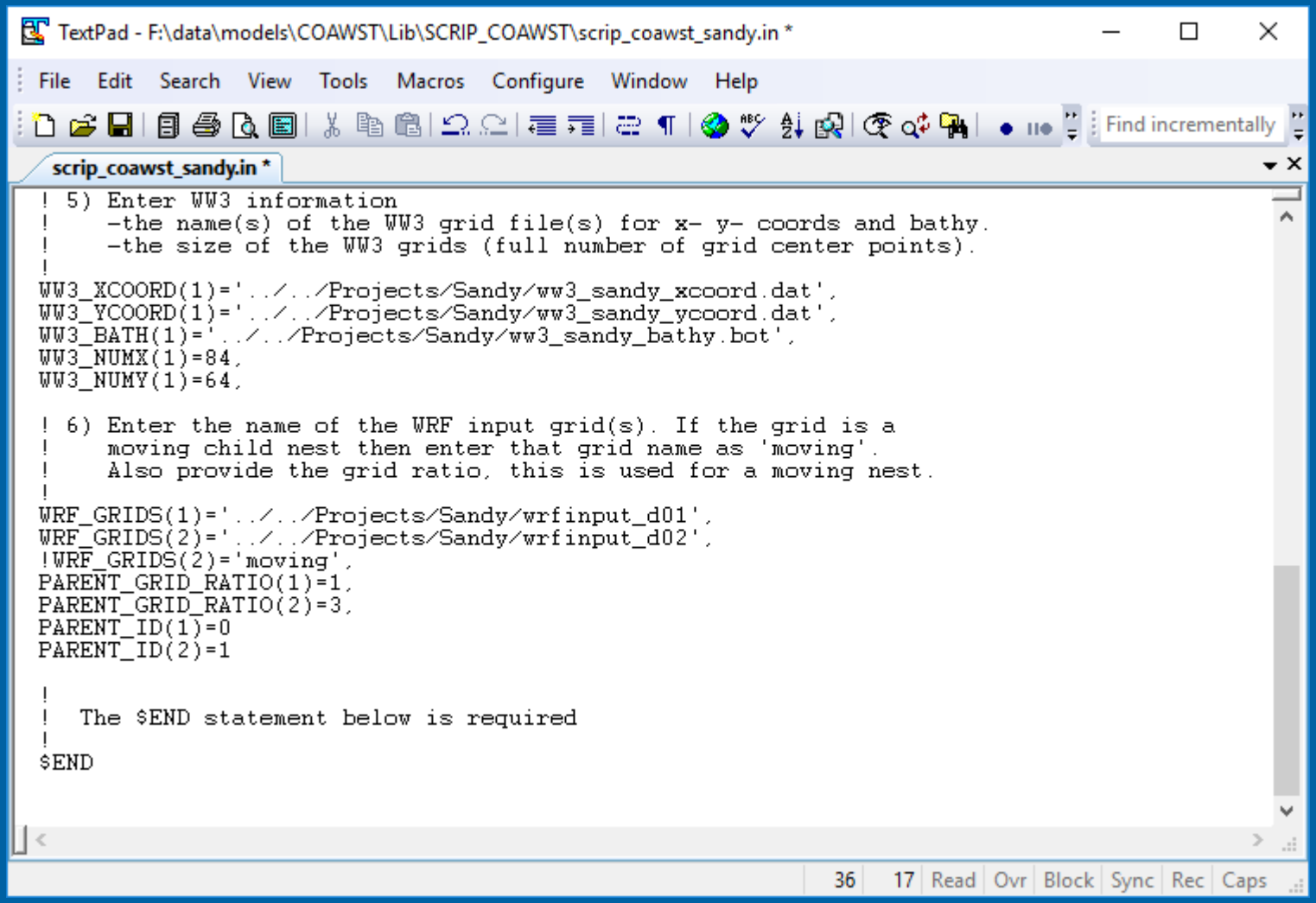
scrip_coawst_sandy.in *

```
$INPUTS
!
!  Input file for scrip_coawst.
!  The $INPUTS line is required at the top of this file.
!  Edit this file to enter the correct information below.
!  Then run this program as "scrip_coawst scrip_coawst_sandy.in"
!
! 1) Enter name of output netcdf4 file
!
!OUTPUT_NCFILE='scrip_sandy_moving.nc'
OUTPUT_NCFILE='scrip_sandy_static.nc'
!OUTPUT_NCFILE='scrip_sandy_nowavenest.nc'

! 2) Enter total number of ROMS, SWAN, WW3, and WRF grids:
!
NGRIDS_ROMS=2,
NGRIDS_SWAN=0,
NGRIDS_WW3=0,
NGRIDS_WRF=2,

! 3) Enter name of the ROMS grid file(s):
!
ROMS_GRIDS(1)='../../Projects/Sandy/Sandy_roms_grid.nc',
ROMS_GRIDS(2)='../../Projects/Sandy/Sandy_roms_grid_ref3.nc',

! 4) Enter SWAN information:
!    -the name(s) of the SWAN grid file(s) for coords and bathy.
!    -the size of the SWAN grids (full number of center points), and
!    -if the swan grids are Spherical(set cartesian=0) or
!                           Cartesian(set cartesian=1).
!
SWAN_COORD(1)='../../Projects/Sandy/Sandy_swan_coord.grd',
SWAN_COORD(2)='../../Projects/Sandy/Sandy_swan_coord_ref3.grd',
SWAN_BATH(1)='../../Projects/Sandy/Sandy_swan_bathy.bot',
SWAN_BATH(2)='../../Projects/Sandy/Sandy_swan_bathy_ref3.bot',
SWAN_NUMX(1)=87,
SWAN_NUMX(2)=116,
SWAN_NUMY(1)=65,
SWAN_NUMY(2)=86,
CARTESIAN(1)=0,
CARTESIAN(2)=0,
```

18 12 Read Ovr Block Sync Rec Caps

2) SCRIP

Edit the SCRIP input file:
COAWST/Lib/SCRIP_COAWST/scrip_coawst_sandy.in



2) SCRIP

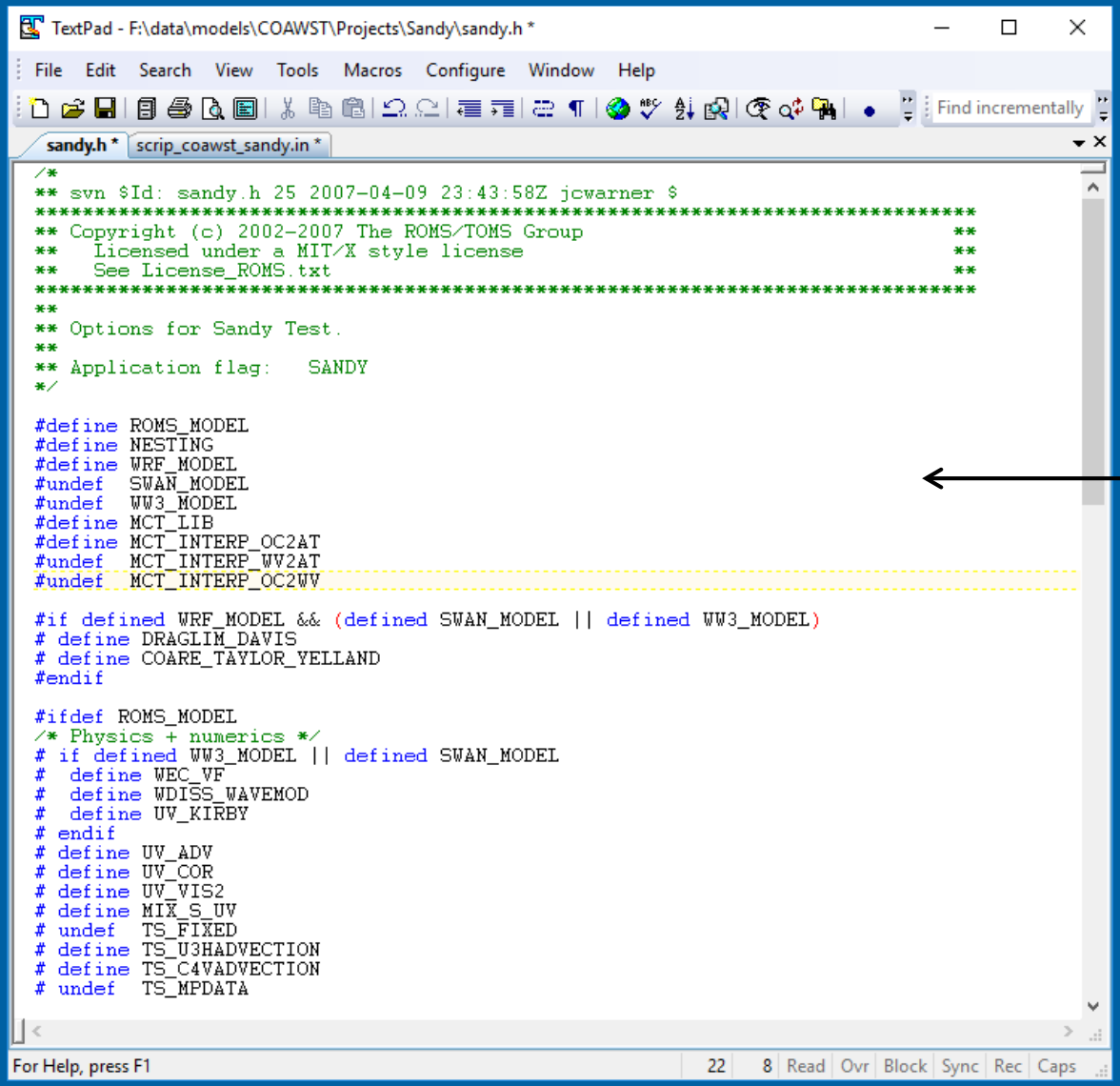
Run this in COAWST/Lib/SCRIP_COAWST
./scrip_coawst scrip_coawst_sandy.in

This will create one file called
OUTPUT_NCFILE='scrip_sandy_static.nc'

WSL-2 Ubuntu ではこれで動いた
ITO-A では エラーとなり，
mpiexe.hydra を用いバッチ処理したら
動いた。
しかし両者でなぜか netcdf の ファイルサイズ
が異なる。 ncview で比較したが、
よく分からなかった。
xscrip を2倍残し，数値を見たが、差はなさそう。
とりあえずうまく行ったと判断

USGS

3) sandy.h



ROMS + WRF application

Projects/Sandy/coupling_sandy.in

4) coupling.in (this is a ROMS+WRF app)

FILE NAVIGATE EDIT BREAKPOINTS

roms_master_climatology_sandy.m | nam_narr_2roms.m | nam_narr_2swan.m | coupling_sandy.in

```
37  ! Number of parallel nodes assigned to each model in the coupled system.
38  ! Their sum must be equal to the total number of processors.
39
40     NnodesATM =  1                    ! atmospheric model
41     NnodesWAV =  1                    ! wave model
42     NnodesOCN =  1                    ! ocean model
43
44  ! Time interval (seconds) between coupling of models.
45
46     TI_ATM2WAV =    1800.0d0          ! atmosphere to wave coupling interval
47     TI_ATM2OCN =    1800.0d0          ! atmosphere to ocean coupling interval
48     TI_WAV2ATM =    1800.0d0          ! wave to atmosphere coupling interval
49     TI_WAV2OCN =    1800.0d0          ! wave to ocean coupling interval
50     TI_OCN2WAV =    1800.0d0          ! ocean to wave coupling interval
51     TI_OCN2ATM =    1800.0d0          ! ocean to atmosphere coupling interval
52
53  ! Enter names of Atm, Wav, and Ocn input files.
54  ! The Wav program needs multiple input files, one for each grid.
55
56     ATM_name = namelist.input                          ! atmospheric model
57     WAV_name = Projects/Sandy/swan_sandy.in \
58                Projects/Sandy/swan_sandy_ref3.in       ! wave model
59     OCN_name = Projects/Sandy/ocean_sandy.in           ! ocean model
60
61  ! Sparse matrix interpolation weights files. You have 2 options:
62  ! Enter "1" for option 1, or "2" for option 2, and then list the
63  ! weight file(s) for that option.
64
65     SCRIP_WEIGHT_OPTION = 1
66  !
67  ! Option 1: IF you set "SCRIP_WEIGHT_OPTION = 1", then enter name
68  !           of the single netcdf file containing all the exchange
```

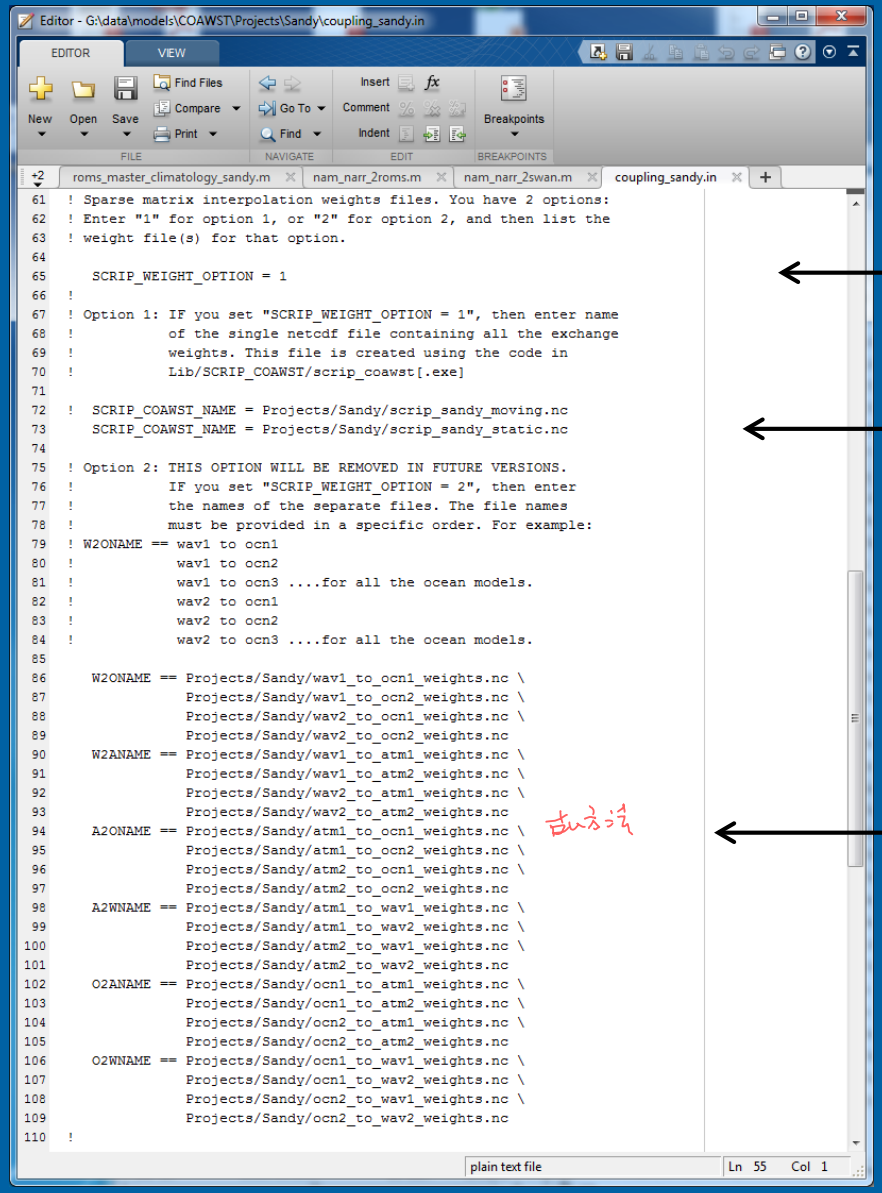
plain text file Ln 17 Col 52

set # procs for each model

set coupling interval. Can be different for each direction.

input file names. only 1 for WRF, 1 for ROMS, multiple for SWAN

4) coupling.in (this is a ROMS+WRF app)




```
! Sparse matrix interpolation weights files. You have 2 options:
! Enter "1" for option 1, or "2" for option 2, and then list the
! weight file(s) for that option.

   SCRIP_WEIGHT_OPTION = 1
!
! Option 1: IF you set "SCRIP_WEIGHT_OPTION = 1", then enter name
!           of the single netcdf file containing all the exchange
!           weights. This file is created using the code in
!           Lib/SCRIP_COAWST/scrip_coawst[.exe]

!  SCRIP_COAWST_NAME = Projects/Sandy/scrip_sandy_moving.nc
   SCRIP_COAWST_NAME = Projects/Sandy/scrip_sandy_static.nc

! Option 2: THIS OPTION WILL BE REMOVED IN FUTURE VERSIONS.
!           IF you set "SCRIP_WEIGHT_OPTION = 2", then enter
!           the names of the separate files. The file names
!           must be provided in a specific order. For example:
! W2ONAME == wav1 to ocn1
!            wav1 to ocn2
!            wav1 to ocn3 ....for all the ocean models.
!            wav2 to ocn1
!            wav2 to ocn2
!            wav2 to ocn3 ....for all the ocean models.

   W2ONAME == Projects/Sandy/wav1_to_ocn1_weights.nc \
              Projects/Sandy/wav1_to_ocn2_weights.nc \
              Projects/Sandy/wav2_to_ocn1_weights.nc \
              Projects/Sandy/wav2_to_ocn2_weights.nc
   W2ANAME == Projects/Sandy/wav1_to_atm1_weights.nc \
              Projects/Sandy/wav1_to_atm2_weights.nc \
              Projects/Sandy/wav2_to_atm1_weights.nc \
              Projects/Sandy/wav2_to_atm2_weights.nc
   A2ONAME == Projects/Sandy/atm1_to_ocn1_weights.nc \
              Projects/Sandy/atm1_to_ocn2_weights.nc \
              Projects/Sandy/atm2_to_ocn1_weights.nc \
              Projects/Sandy/atm2_to_ocn2_weights.nc
   A2WNAME == Projects/Sandy/atm1_to_wav1_weights.nc \
              Projects/Sandy/atm1_to_wav2_weights.nc \
              Projects/Sandy/atm2_to_wav1_weights.nc \
              Projects/Sandy/atm2_to_wav2_weights.nc
   O2ANAME == Projects/Sandy/ocn1_to_atm1_weights.nc \
              Projects/Sandy/ocn1_to_atm2_weights.nc \
              Projects/Sandy/ocn2_to_atm1_weights.nc \
              Projects/Sandy/ocn2_to_atm2_weights.nc
   O2WNAME == Projects/Sandy/ocn1_to_wav1_weights.nc \
              Projects/Sandy/ocn1_to_wav2_weights.nc \
              Projects/Sandy/ocn2_to_wav1_weights.nc \
              Projects/Sandy/ocn2_to_wav2_weights.nc
!
```

SCRIP weights OPTION 1 or 2

SCRIP weights files name listed here for Option 1 (single file)

NetCDF file uses “groups” feature so we need netcdf 4.

SCRIP weights files names are listed here for Option 2 (multiple files, this option to be phased out.)

4) ocean.in

```
          Lm == 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       プロセス数              ! I-direction partition
      NtileJ == 1                               ! J-direction partition

! Set lateral boundary conditions keyword. Notice that a value is expected
! for each boundary segment per nested grid for each state variable.
!
! Each tracer variable requires [1:4,1:NAT+NPT,Ngrids] values. Otherwise,
! [1:4,1:Ngrids] values are expected for other variables. The boundary
! order is: 1=west, 2=south, 3=east, and 4=north. That is, anticlockwise
! starting at the western boundary.
!
! The keyword is case insensitive and usually has three characters. However,
! it is possible to have compound keywords, if applicable. For example, the
! keyword "RadNud" implies radiation boundary condition with nudging. This
! combination is usually used in active/passive radiation conditions.
```

set # procs NtileI NtileJ

```
! Time-Stepping parameters.

      NTIMES == 5760     11520      coupling interval の分数
          DT == 30.0d0   15.0d0     である必要がある
      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.
```

dt needs to divide evenly into the coupling interval.

5) namelist.input

TextPad - F:\data\models\COAWST\Projects\Sandy\namelist.input

File Edit Search View Tools Macros Configure Window Help

namelist.input | coupling_sandy.in | sandy.h * | scrip_coawst_sandy.in *

```
 start_year                          = 2012, 2012, 2012,
 start_month                         = 10,   10,   10,
 start_day                           = 28,   28,   28,
 start_hour                          = 12,   12,   12,
 start_minute                        = 00,   00,   00,
 start_second                        = 00,   00,   00,
 end_year                            = 2012, 2012, 2012,
 end_month                           = 10,   10,   10,
 end_day                             = 30,   30,   30,
 end_hour                            = 12,   12,   12,
 end_minute                          = 00,   00,   00,
 end_second                          = 00,   00,   00,
 interval_seconds                    = 21600
 input_from_file                     = .true.,.true.,.false.,
 history_interval                    = 30,   30,   60,
 frames_per_outfile                  = 1000, 1000, 1000,
 restart                             = .false.,
 restart_interval                    = 5000,
 io_form_history                     = 2
 io_form_restart                     = 2
 io_form_input                       = 2
 io_form_boundary                    = 2
 debug_level                         = 0
 auxinput4_inname                    = "wrflowinp_d<domain>"
 auxinput4_interval                  = 360, 360, 360,
 io_form_auxinput4                   = 2
 /

 &domains
 time_step                           = 180,
 time_step_fract_num                 = 0,
 time_step_fract_den                 = 1,
 max_dom                             = 2,
 e_we                                = 85,    100,   94,
 e_sn                                = 82,    100,   91,
 e_vert                              = 48,    48,    48,
 p_top_requested                     = 5000,
 num_metgrid_levels                  = 40,
 num_metgrid_soil_levels             = 4,
 dx                                  = 30000, 10000,  3333.33,
 dy                                  = 30000, 10000,  3333.33,
 grid_id                             = 1,     2,     3,
 parent_id                           = 0,     1,     2,
 i_parent_start                      = 1,     33,    30,
 j_parent_start                      = 1,     8,     30,
 parent_grid_ratio                   = 1,     3,     3,
 parent_time_step_ratio              = 1,     3,     3,
 feedback                            = 1,
 smooth_option                       = 0
 corral_dist                         = 2,     2,     2,
 nproc_x                             = 1
 nproc_y                             = 1
 /
```

WRF

プロセス数

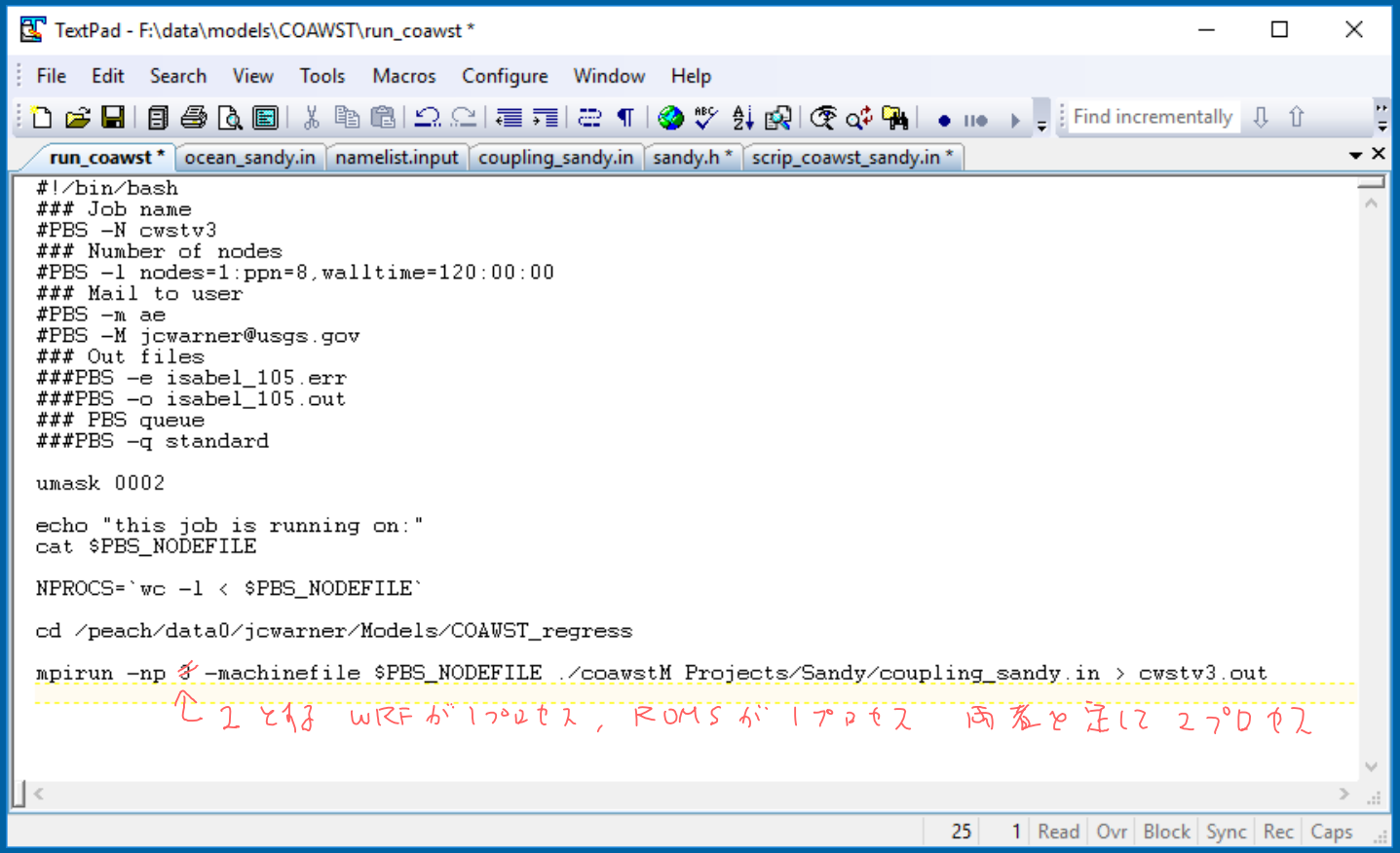
For Help, press F1 | 1 | 1 | Read | Ovr | Block | Sync | Rec | Caps

need dt of 180 to divide evenly into coupling interval of 1800 sec.

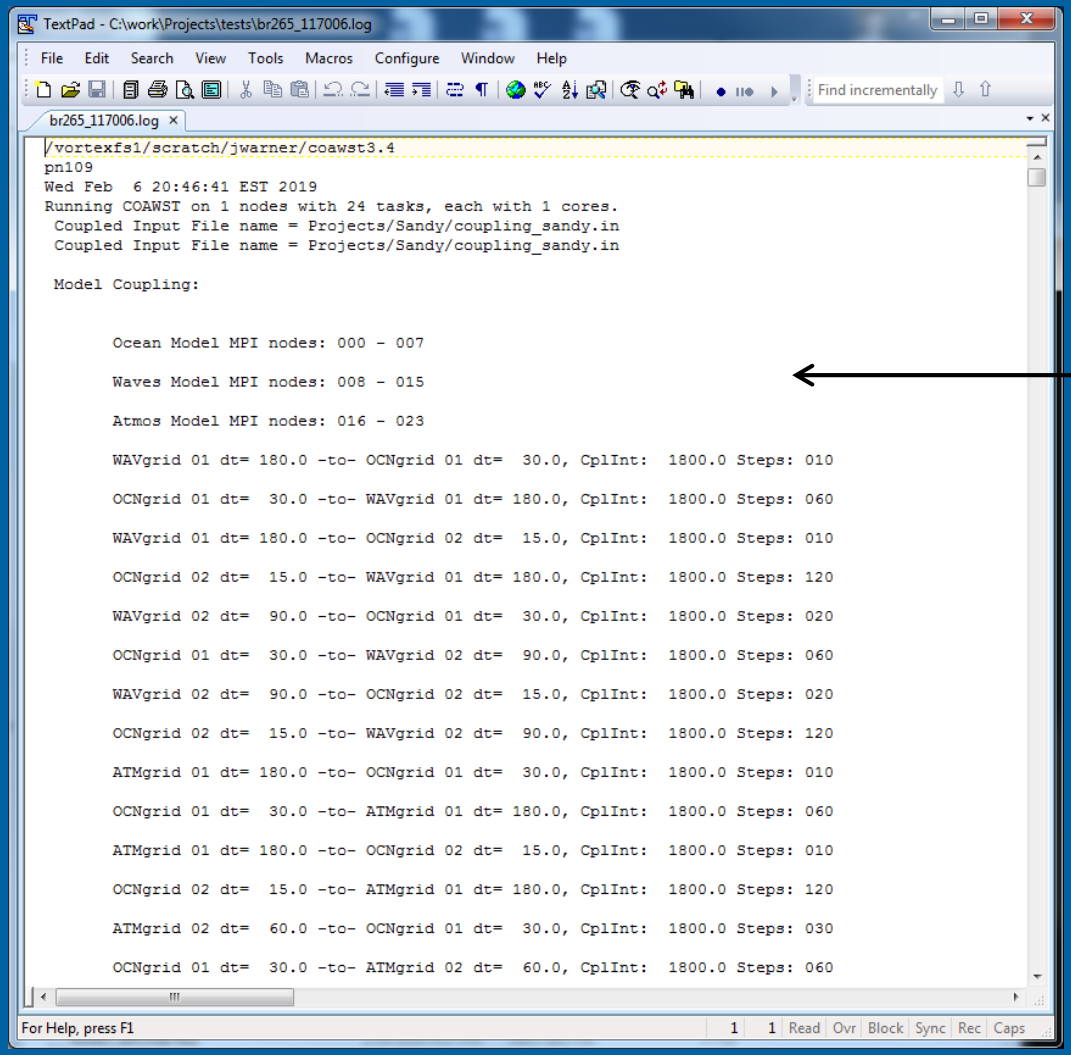
set # procs for atm model

6) run it as coawstM

- use total number of procs from coupling.in
- only 1 executable



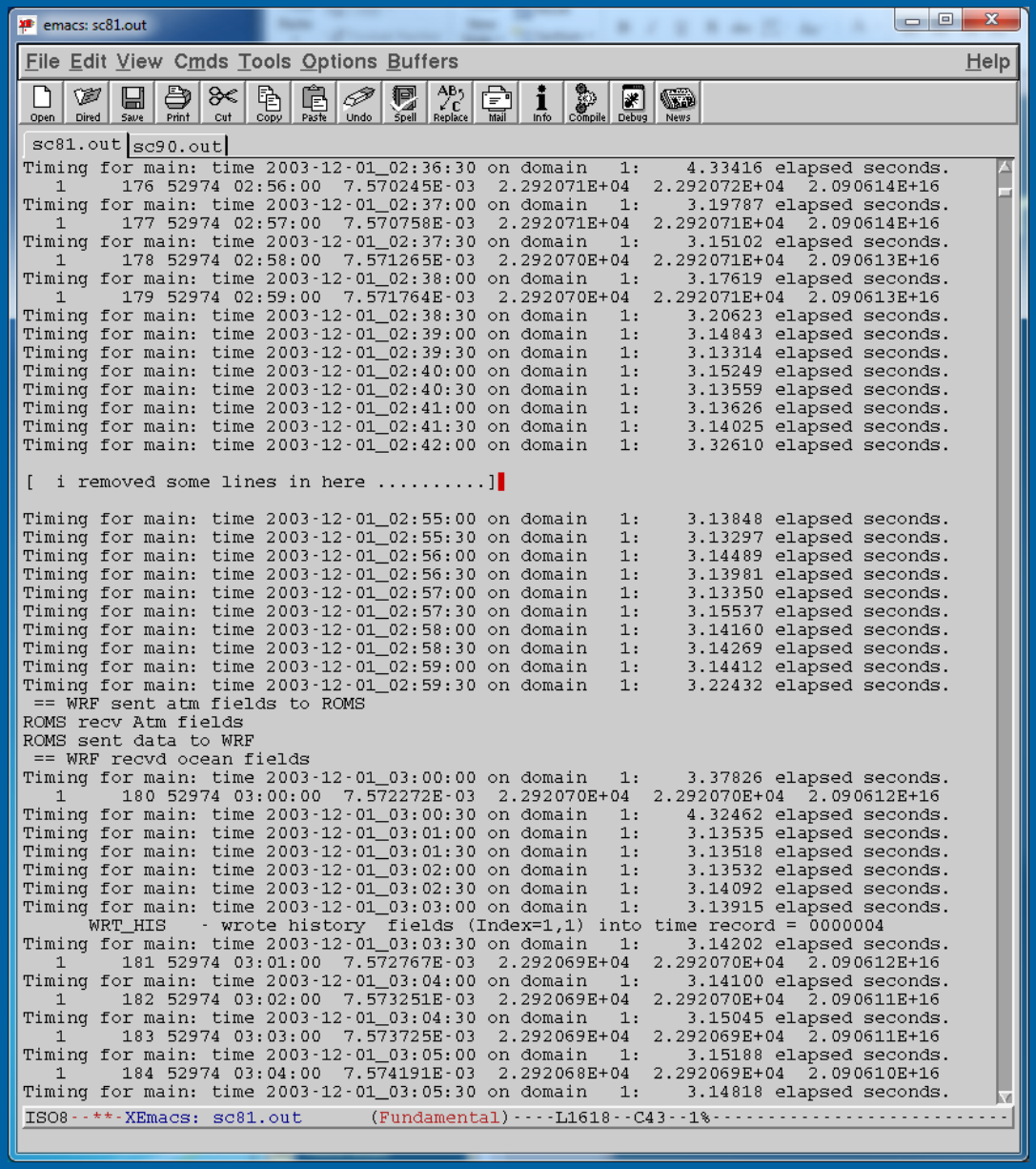
Processor allocation



stdout reports processor allocation
and
Number of steps between synchronization.

This looks like from a different run,
but you get the idea

Processor allocation



"Timing for …." = WRF

"1 179 52974 02:59:00 " = ROMS

Here is where the model coupling synchronization occurs.
so probably could re-allocate more nodes to WRF

Data field exchange min/max values

jwarner@poseidon-l1:/vortexfs1/scratch/jwarner/coawst3.4_test10

```
Timing for main: time 2012-10-28_13:30:00 on domain   2:    0.19166 elapsed seconds
 ## ROMS grid  1 recv data from WRF grid  1
WRFtoROMS Min/Max  GSW      (Wm-2):     0.000000E+00  4.476423E+02
WRFtoROMS Min/Max  GLW      (Wm-2):     0.000000E+00  4.406325E+02
WRFtoROMS Min/Max  LH       (Wm-2):     0.000000E+00  8.354071E+02
WRFtoROMS Min/Max  HFX      (Wm-2):    -1.985328E+01  2.440330E+02
WRFtoROMS Min/Max  USTRESS  (Nm-2):    -9.503103E-01  1.176217E+00
WRFtoROMS Min/Max  VSTRESS  (Nm-2):    -8.051951E-01  7.339598E-01
WRFtoROMS Min/Max  MSLP     (mb):       0.000000E+00  1.020593E+03
WRFtoROMS Min/Max  RELH     (-):        0.000000E+00  9.506385E-01
WRFtoROMS Min/Max  T2       (C):        0.000000E+00  2.687080E+01
WRFtoROMS Min/Max  RAIN   (kgm-2s-1):   0.000000E+00  1.318768E-03
WRFtoROMS Min/Max  EVAP   (kgm-2s-1):   0.000000E+00  3.425169E-04
  ## WRF grid  1 sent data to ROMS grid  1
  ## WRF grid  2 sent data to ROMS grid  1
 ## ROMS grid  1 recv data from WRF grid  2
WRFtoROMS Min/Max  GSW      (Wm-2):     0.000000E+00  4.179347E+02
WRFtoROMS Min/Max  GLW      (Wm-2):     0.000000E+00  4.444828E+02
WRFtoROMS Min/Max  LH       (Wm-2):     0.000000E+00  8.468719E+02
WRFtoROMS Min/Max  HFX      (Wm-2):    -1.604046E+01  2.516284E+02
WRFtoROMS Min/Max  USTRESS  (Nm-2):    -1.651402E+00  1.887024E+00
WRFtoROMS Min/Max  VSTRESS  (Nm-2):    -1.810099E+00  1.265032E+00
WRFtoROMS Min/Max  MSLP     (mb):       0.000000E+00  1.003535E+03
WRFtoROMS Min/Max  RELH     (-):        0.000000E+00  9.839446E-01
WRFtoROMS Min/Max  T2       (C):        0.000000E+00  2.655765E+01
WRFtoROMS Min/Max  RAIN   (kgm-2s-1):   0.000000E+00  1.059673E-02
WRFtoROMS Min/Max  EVAP   (kgm-2s-1):   0.000000E+00  3.472175E-04
 ## ROMS grid  2 recv data from WRF grid  1
WRFtoROMS Min/Max  GSW      (Wm-2):     0.000000E+00  1.476153E+02
WRFtoROMS Min/Max  GLW      (Wm-2):     0.000000E+00  4.187577E+02
WRFtoROMS Min/Max  LH       (Wm-2):     0.000000E+00  4.920473E+02
WRFtoROMS Min/Max  HFX      (Wm-2):    -2.440168E+01  9.009011E+01
WRFtoROMS Min/Max  USTRESS  (Nm-2):    -9.670363E-01  0.000000E+00
WRFtoROMS Min/Max  VSTRESS  (Nm-2):    -6.882157E-01  0.000000E+00
WRFtoROMS Min/Max  MSLP     (mb):       0.000000E+00  1.010489E+03
WRFtoROMS Min/Max  RELH     (-):        0.000000E+00  9.404117E-01
WRFtoROMS Min/Max  T2       (C):        0.000000E+00  2.336388E+01
WRFtoROMS Min/Max  RAIN   (kgm-2s-1):   0.000000E+00  1.126228E-03
WRFtoROMS Min/Max  EVAP   (kgm-2s-1):   0.000000E+00  2.017394E-04
 ## ROMS grid  2 recv data from WRF grid  2
WRFtoROMS Min/Max  GSW      (Wm-2):     0.000000E+00  3.141597E+02
WRFtoROMS Min/Max  GLW      (Wm-2):     0.000000E+00  4.400405E+02
WRFtoROMS Min/Max  LH       (Wm-2):     0.000000E+00  7.637908E+02
WRFtoROMS Min/Max  HFX      (Wm-2):    -2.377396E+01  2.101876E+02
WRFtoROMS Min/Max  USTRESS  (Nm-2):    -1.668581E+00  2.084062E-01
WRFtoROMS Min/Max  VSTRESS  (Nm-2):    -1.696622E+00  2.368454E-01
WRFtoROMS Min/Max  MSLP     (mb):       0.000000E+00  1.003199E+03
WRFtoROMS Min/Max  RELH     (-):        0.000000E+00  9.717784E-01
WRFtoROMS Min/Max  T2       (C):        0.000000E+00  2.558522E+01
WRFtoROMS Min/Max  RAIN   (kgm-2s-1):   0.000000E+00  5.698533E-03
WRFtoROMS Min/Max  EVAP   (kgm-2s-1):   0.000000E+00  3.131542E-04
 ## ROMS grid  1  sent data to WRF grid  1
 ## ROMS grid  1  sent data to WRF grid  2
 ## ROMS grid  2  sent data to WRF grid  1
 ## ROMS grid  2  sent data to WRF grid  2
  ## WRF grid  1 sent data to ROMS grid  2
```

WRF 1 to ROMS 1

WRF 2 to ROMS 1

WRF 1 to ROMS 2

WRF 2 to ROMS 2