Developing a high-resolution coupled regional climate model for the tropical Atlantic region

Raffaele Montuoro
Supercomputing Facility, Texas A&M University

Ping Chang, R. Saravanan
Depts. of Oceanography & Atmospheric Sciences, Texas A&M University

L. Ruby Leung
Pacific Northwest National Laboratory

U.S. DEPARTMENT OF ENERGY

TEXAS A&M UNIVERSITY

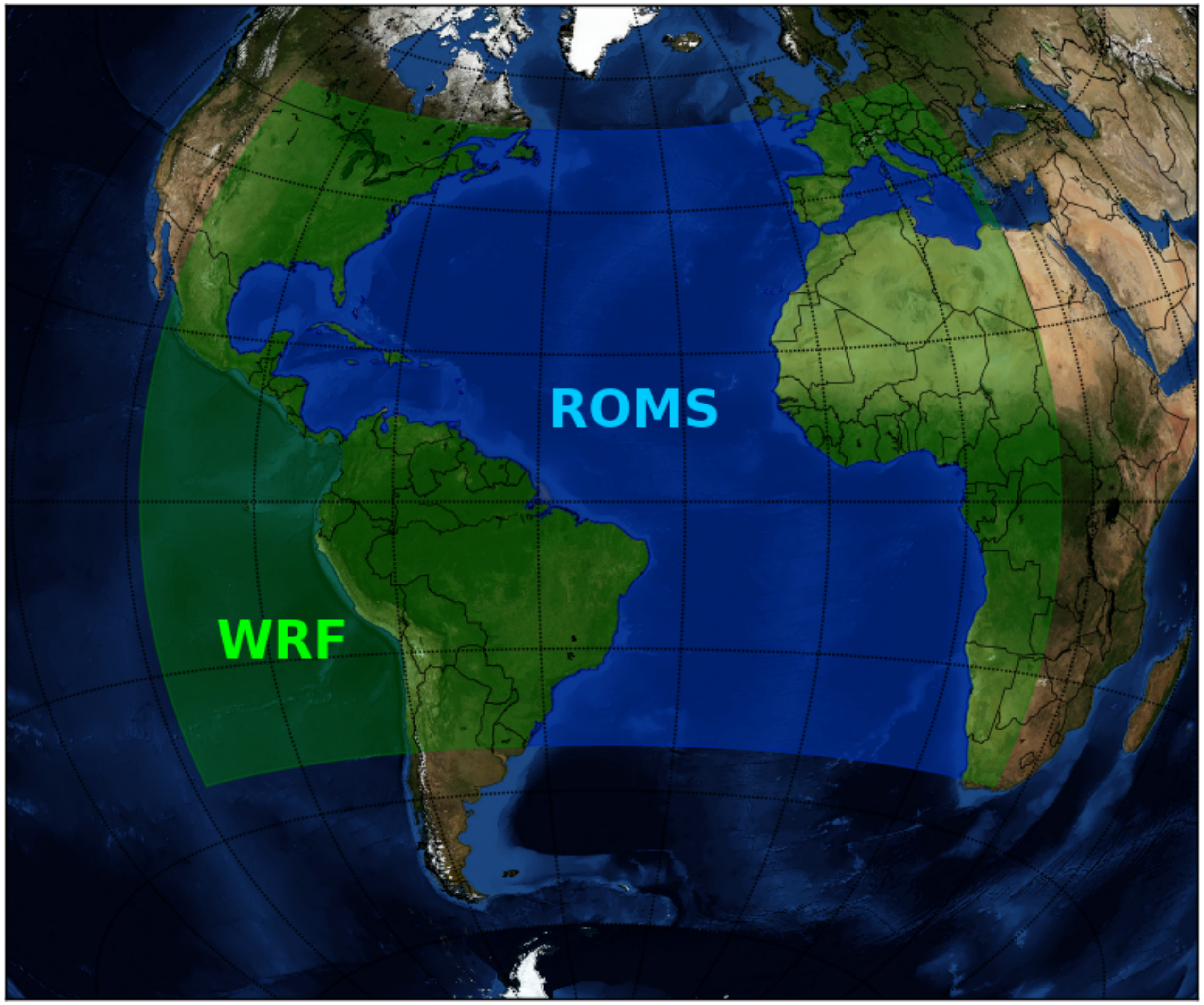
Pacific Northwest NATIONAL LABORATORY

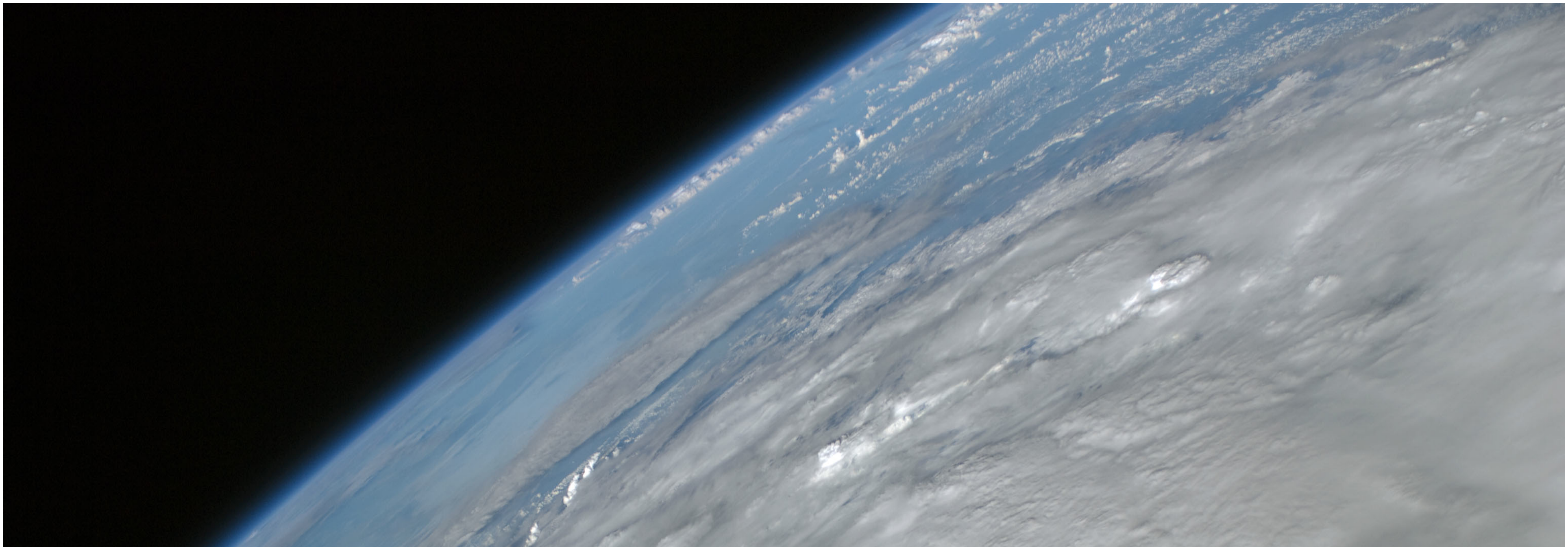
UCAR SEA Conference 2012, Boulder, CO – February 22, 2012

A Coupled Regional Climate Model (CRCM)

- Extreme climate events affecting the US—hurricanes, severe precipitation, drought—are influenced by the conditions in the tropical Atlantic region
- Ability to simulate accurately the climate mean and variability in the tropical Atlantic becomes crucial
- Results from current climate models show biases in the tropical Atlantic climate. Too coarse resolutions unable to resolve some of the processes responsible for such biases.
- A high-resolution coupled climate model may be effective in addressing the tropical bias. Grid resolution as low as 1km.

Ocean/Atmosphere domains in CRCM

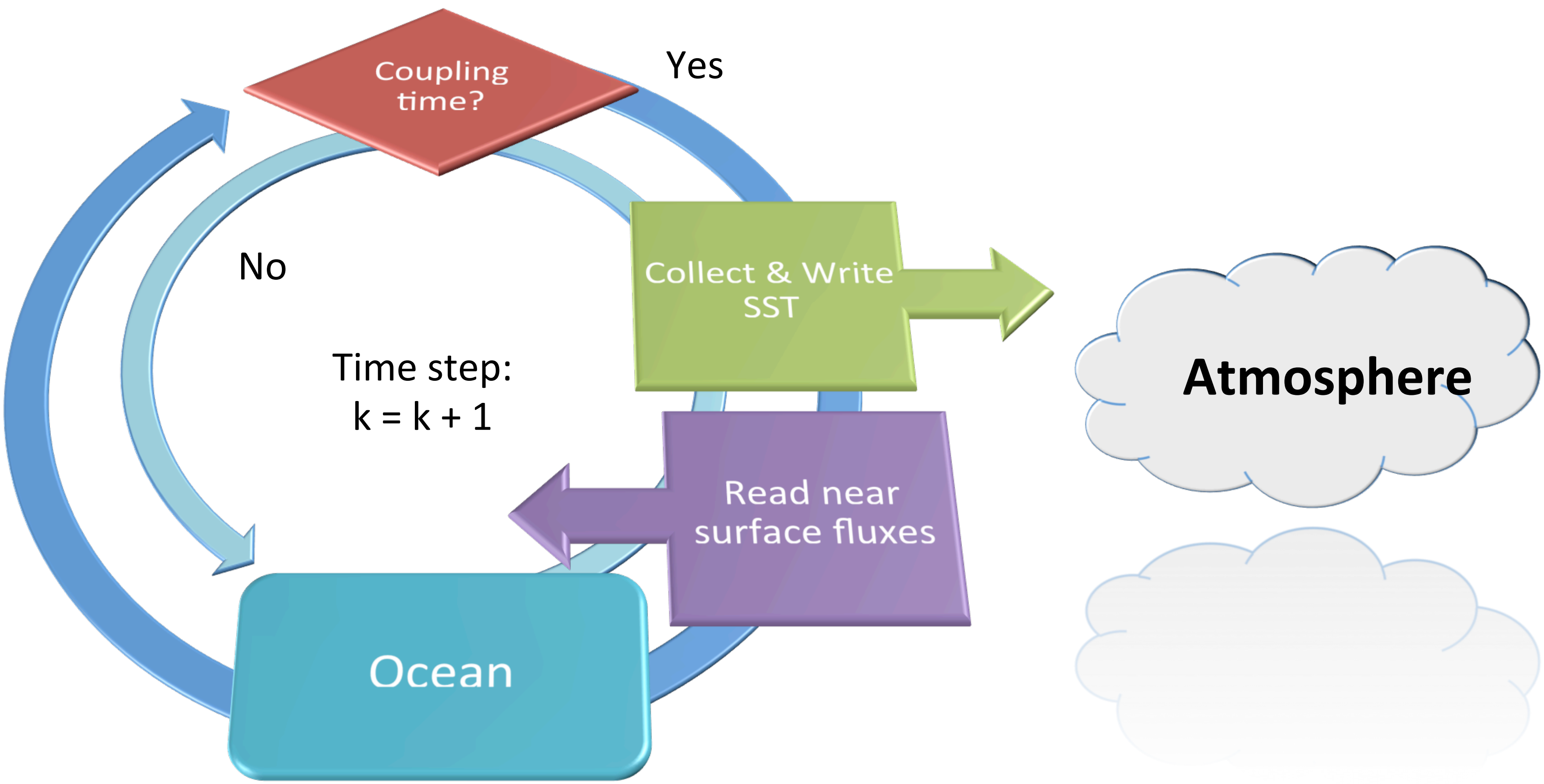




Mechanics of ocean/atmosphere coupling

Defining exchange process and quantities

The coupling process



Coupling equations: *surface Reynold's stress*

U_{ij}^* : velocity scale in similarity theory

$u_{10,ij}, v_{10,ij}$: u and v at 10m

Atmosphere

$$\rho_{ij} = \sqrt{u_{10,ij}^2 + v_{10,ij}^2}$$

$$\exists U_{ij}^* > 0: \qquad \tau_{x,ij} = \begin{cases} \dfrac{T_{ij} u_{10,ij}}{\rho_{ij}} & \rho_{ij} > 0 \\ 0 & \rho_{ij} = 0 \end{cases} \qquad T_{ij} = 1.293 \left| U_{ij}^* \right|^2$$

otherwise:

$$\tau_{x,ij} = T_{ij} \rho_{ij} u_{10,ij}$$

$$\tau_{y,ij} = T_{ij} \rho_{ij} v_{10,ij}$$

$$T_{ij} = \begin{cases} 1.2 \times 10^{-3} \left(\dfrac{0.4}{14.56 - \ln \rho_{ij}} \right)^2 & \rho_{ij} > 0 \\ 0 & \rho_{ij} = 0 \end{cases}$$

Coupling equations: *short wave & heat flux*

$$\sigma_{ij}^{rf,\mathrm{ROMS}} \leftarrow \sigma_{ij}^{rf,\mathrm{WRF}}$$

net short wave flux at ground surface

$$H_{ij} = \sigma_{ij}^{rf} - h_{ij}^{s} - \varphi_{ij}^{up} + \varepsilon_{ij}^{s}\left(\lambda_{ij}^{down} - 5.67051\times10^{-8}\, s_{ij}^{4}\right)$$

heat flux at ground surface

h_{ij}^{s}	latent heat flux at the surface
φ_{ij}^{up}	upward heat flux at the surface
ε_{ij}^{s}	surface emissivity
λ_{ij}^{down}	downward long wave flux at ground surface
s_{ij}	sea surface temperature

Atmosphere

Coupling equations: *water balance*

$$Q_{ij} = \frac{\Xi_{ij}(k) - \Xi_{ij}(k-1) + \Gamma_{ij}(k) - \Gamma_{ij}(k-1)}{t(k) - t(k-1)} - q_{ij}^{up}$$

$\Xi_{ij}(k)$	accumulated total cumulus precipitation at time step k	*Atmosphere*
$\Gamma_{ij}(k)$	accumulated total grid scale precipitation at time step k	*Atmosphere*
$t(k)$	model time at time step k	*Atmosphere*
q_{ij}^{up}	upward moisture flux at the surface	*Atmosphere*

Implementation overview

WRF/ROMS coupled code

Critical steps in model coupling

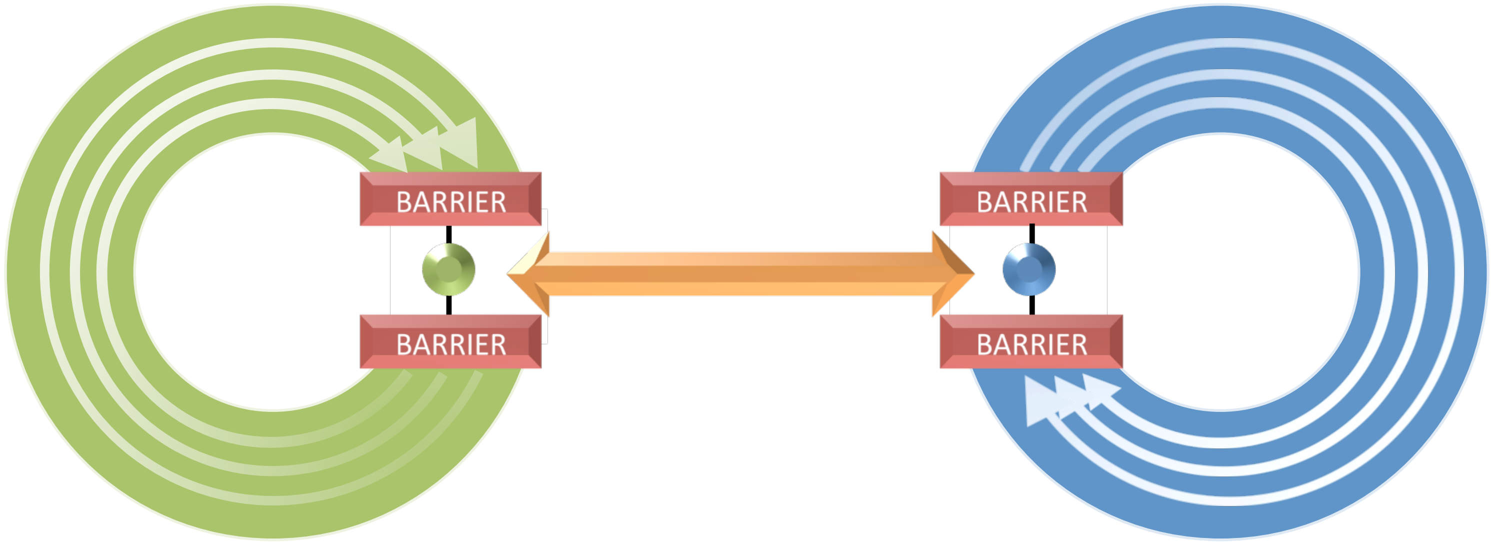
- **Simultaneous execution** of parallel (MPI) independent codes

Critical steps in model coupling

- **Simultaneous execution** of parallel (MPI) independent codes
- **Data exchange** between models—several different ways possible. I/O Interfaces.

Critical steps in model coupling

- **Simultaneous execution** of parallel (MPI) independent codes
- **Data exchange** between models—several different ways possible. I/O Interfaces.
- **Synchronization**
  - *Intra-model* – pre/post coupling data input/output
  - *Inter-model* – sync time loops, exchange data at right time



Implementation strategies

- **Minimal modifications** to original model codes to improve portability
- **Modularity:** enclose coupling code into software modules
- Coupling code added through **preprocessor directives**

  Advantages:

  - Implementation is straightforward
  - Complete code after preprocessing—helps debugging
  - May be distributed using standard, open-source tools—e.g. GNU autoconf.

  Disadvantages:

  - Source code may become hard to follow
  - No "hot" pluggable

The starting point …

- **Regional Ocean Modeling System (ROMS) version 3.4**

  ROMS/TOMS Framework: June 3, 2011

  Snapshot:

  svn: $HeadURL: https://www.myroms.org/svn/src/trunk/ROMS/Version $

  svn: $LastChangedBy: arango $

  svn: $LastChangedRevision: 563 $

  svn: $LastChangedDate: 2011-06-03 16:26:31 -0500 (Fri, 03 Jun 2011) $

- **Weather Research and Forecast (WRF) Model Version 3.3.1**

  September 16, 2011

  http://wrf-model.org/users/users.php

- Only pure distributed-memory (MPI) versions used. No OpenMP/hybrid.

Approach to inter-model communication

- Communication and data exchange between models achieved through I/O interfaces.
- Software I/O interfaces of various complexity are available:
  - Disk I/O – generic files, NetCDF
  - Message Passing Interface (MPI)
  - Model Coupling Toolkit (MCT)
  - CPL7
- WRF provides framework to handle generic I/O streams (default: NetCDF)
- WRF's I/O APIs allow to add custom I/O interfaces—e.g.:
  `external/io_coupler/`
- ROMS only provides a low-level interface to NetCDF calls.

Synchronization

- **Intra-model** synchronization is achieved through (MPI) barriers:

  Input barrier: receive -> distribute

  Output barrier: collect -> send

- Depending upon the I/O interface, barriers can be *implicit* or *explicit*

- All WRF synchronization barriers are **implicit** if using I/O streams

- Time loops and data exchange synchronization (**inter-model**) is obtained through matching timestamps for the exchanged data, waiting if no (suitable) data is available.

- Use of a single (external) time loop is planned.

Coupled code: ROMS/WRF files

1. Compilers/make_macros.h
2. ROMS/Drivers/nl_ocean.h
3. ROMS/Include/cplatl.h
4. ROMS/Include/tamu.h
5. ROMS/Nonlinear/diag.F
6. ROMS/Nonlinear/main3d.F
7. ROMS/TAMU/mod_tamu_coupling.F
8. ROMS/Utility/inp_par.F

1. share/input_wrf.F
2. share/mediation_integrate.F
3. share/module_check_a_mundo.F

Coupled code: ROMS files

1. **Compilers/make_macros.h**
2. ROMS/Drivers/nl_ocean.h
3. ROMS/Include/tamu.h
4. ROMS/Nonlinear/diag.F
5. ROMS/Nonlinear/main3d.F
6. ROMS/TAMU/mod_tamu_coupling.F
7. ROMS/Utility/inp_par.F

Define `USE_TAMU`, used in `makefile` to include new TAMU module:

```
#ifdef TAMU_COUPLER
  USE_TAMU := on
#else
  USE_TAMU :=
#endif
```

Coupled code: ROMS files

1. `Compilers/make_macros.h`
2. **`ROMS/Drivers/nl_ocean.h`**
3. `ROMS/Include/tamu.h`
4. `ROMS/Nonlinear/diag.F`
5. `ROMS/Nonlinear/main3d.F`
6. `ROMS/TAMU/mod_tamu_coupling.F`
7. `ROMS/Utility/inp_par.F`

a. Read atmosphere (WRF) grid

b. Initialize coupling variables

Coupled code: ROMS files

1. `Compilers/make_macros.h`
2. `ROMS/Drivers/nl_ocean.h`
3. **`ROMS/Include/tamu.h`**
4. `ROMS/Nonlinear/diag.F`
5. `ROMS/Nonlinear/main3d.F`
6. `ROMS/TAMU/mod_tamu_coupling.F`
7. `ROMS/Utility/inp_par.F`

New option set for coupling.

Use in input:

```
MyAppCPP = TAMU
```

Coupled code: ROMS files

1. `Compilers/make_macros.h`
2. `ROMS/Drivers/nl_ocean.h`
3. `ROMS/Include/tamu.h`
4. **`ROMS/Nonlinear/diag.F`**
5. `ROMS/Nonlinear/main3d.F`
6. `ROMS/TAMU/mod_tamu_coupling.F`
7. `ROMS/Utility/inp_par.F`

Modifies diag output

– Introduces calendar date & time

Coupled code: ROMS files

1. Compilers/make_macros.h
2. ROMS/Drivers/nl_ocean.h
3. ROMS/Include/tamu.h
4. ROMS/Nonlinear/diag.F
5. **ROMS/Nonlinear/main3d.F**
6. ROMS/TAMU/mod_tamu_coupling.F
7. ROMS/Utility/inp_par.F

Introduce ocean to atmosphere coupling after ocean to wave:

```
DO ng=1,Ngrids
  IF (DoTAMUCoupling(ng,iic(ng))) THEN
    coupling_count = coupling_count + 1
!$OMP PARALLEL DO PRIVATE(thread,subs,tile)
SHARED(ng,numthreads)
    DO thread=0,numthreads-1
     subs=NtileX(ng)*NtileE(ng)/numthreads
     DO tile=subs*(thread+1)-1,subs*thread,-1
       CALL tamu_ocn2atm_coupling()
     END DO
    END DO
!$OMP END PARALLEL DO
  END IF
END DO
```

Coupled code: ROMS files

1. `Compilers/make_macros.h`
2. `ROMS/Drivers/nl_ocean.h`
3. `ROMS/Include/tamu.h`
4. `ROMS/Nonlinear/diag.F`
5. `ROMS/Nonlinear/main3d.F`
6. **`ROMS/TAMU/mod_tamu_coupling.F`**
7. `ROMS/Utility/inp_par.F`

Main module containing TAMU coupling code

Coupled code: ROMS files

1. `Compilers/make_macros.h`
2. `ROMS/Drivers/nl_ocean.h`
3. `ROMS/Include/tamu.h`
4. `ROMS/Nonlinear/diag.F`
5. `ROMS/Nonlinear/main3d.F`
6. `ROMS/TAMU/mod_tamu_coupling.F`
7. **`ROMS/Utility/inp_par.F`**

Include new input parameters for coupling

Coupled code: WRF files

1. **`share/input_wrf.F`**
2. `share/mediation_integrate.F`
3. `share/module_check_a_mundo.F`

Modifies return behavior for stream `auxinput4`

– No fatal error is generated when reading

Coupled code: WRF files

1. share/input_wrf.F
2. **share/mediation_integrate.F**
3. share/module_check_a_mundo.F

Driver for coupling – reads ROMS SSTs:

```
#ifdef TAMU_COUPLER

ELSE IF( ialarm .EQ. AUXINPUT4_ALARM ) THEN
 IF(WRFU_AlarmIsRinging(grid%alarms(ialarm),rc=rc)) THEN
  IF ( grid%coupling_import ) THEN
    CALL med_read_roms_sst ( grid , config_flags )
  ELSE
    CALL wrf_debug(0,' Skipping SST')
  ENDIF
  CALL WRFU_AlarmRingerOff(grid%alarms(ialarm),rc=rc)
 ENDIF

#endif
```

Coupled code: WRF files

1. `share/input_wrf.F`
2. `share/mediation_integrate.F`
3. **`share/module_check_a_mundo.F`**

Checks coupling parameters in input

Subroutine call sequence: WRF

```
PROGRAM wrf                         main/wrf.F
```

Subroutine call sequence: WRF

`PROGRAM wrf`	`main/wrf.F`
`↳ call wrf_run`	`main/module_wrf_top.F`

Subroutine call sequence: WRF

`PROGRAM wrf`	`main/wrf.F`
↳ `call wrf_run`	`main/module_wrf_top.F`
↳ `call integrate(grid)`	`frame/module_integrate.F`

Driver-level recursive subroutine for integration over domains & subdomains

Subroutine call sequence: WRF

`PROGRAM wrf > call wrf_run > call integrate(grid)`

```
IF ( .NOT. domain_clockisstoptime( grid ) ) THEN
  ... load configuration information for grid ...
  DO WHILE ( .NOT. domain_clockisstopsubtime(grid) )
    CALL med_setup_step ( grid , config_flags )
    ... initialize nests ...
    ... accumulate DFI ...
    CALL med_before_solve_io ( grid , config_flags )
    grid_ptr => grid
    DO WHILE ( ASSOCIATED( grid_ptr ) )
      CALL set_current_grid_ptr( grid_ptr )
      CALL solve_interface ( grid_ptr )
      CALL domain_clockadvance ( grid_ptr )
      CALL domain_time_test( grid_ptr, 'domain_clockadvance' )
      grid_ptr => grid_ptr%sibling
    END DO
    CALL set_current_grid_ptr( grid )
    CALL med_calc_model_time ( grid , config_flags )
    CALL med_after_solve_io ( grid , config_flags )
    grid_ptr => grid
    ... recursive: advance nests to this time level (call integrate) ...
    IF (domain ending) CALL med_last_solve_io ( grid , config_flags)
  END DO
END IF
```

Check if done with domain (IF (.NOT. domain_clockisstoptime …))

Iterate forward until subtime (DO WHILE (.NOT. domain_clockisstopsubtime …))

input/output streams history & restart data written (CALL med_before_solve_io)

Solver: advance domain by dt (CALL solve_interface)

Stub (compute time series) (CALL med_after_solve_io)

Final history & restart output (CALL med_last_solve_io)

Subroutine call sequence: WRF

```
PROGRAM wrf                                                    main/wrf.F
 └→ call wrf_run                                               main/module_wrf_top.F
     └→ call integrate(grid)                                   frame/module_integrate.F
         └→ call med_before_solve_io( grid , config_flags )
                                                               share/mediation_integrate.F
```

```
DO ialarm = first_auxhist, last_auxhist
   ... write history data ...
END DO
DO ialarm = first_auxinput, last_auxinput
   ...
#ifdef TAMU_COUPLER
! - Get ROMS SST
   ELSE IF( ialarm .EQ. AUXINPUT4_ALARM ) THEN
      IF( WRFU_AlarmIsRinging( grid%alarms( ialarm ), rc=rc ) ) THEN
         IF ( grid%coupling_import ) THEN
            CALL med_read_roms_sst ( grid , config_flags )
         ELSE
            CALL wrf_debug(0,' Skipping SSTs')
         ENDIF
         CALL WRFU_AlarmRingerOff( grid%alarms( ialarm ), rc=rc )
      ENDIF
#endif
   ELSE IF ( ... ) THEN
   ...
END DO
... write restart data ...
... look for boundary data ...
```

Input SSTs

Subroutine call sequence: WRF

```
PROGRAM wrf                                                    main/wrf.F
 └→ call wrf_run                                               main/module_wrf_top.F
     └→ call integrate(grid)                                   frame/module_integrate.F
         └→ call med_before_solve_io( grid , config_flags )
                                                               share/mediation_integrate.F
```

```
DO ialarm = first_auxhist, last_auxhist
   ... write history data ...
END DO
DO ialarm = first_auxinput, last_auxinput
   ...
#ifdef TAMU_COUPLER
! - Get ROMS SST
   ELSE IF( ialarm .EQ. AUXINPUT4_ALARM ) THEN
      IF( WRFU_AlarmIsRinging( grid%alarms( ialarm ), rc=rc ) ) THEN
         IF ( grid%coupling_import ) THEN
            CALL med_read_roms_sst ( grid , config_flags )
         ELSE
            CALL wrf_debug(0,' Skipping SSTs')
         ENDIF
         CALL WRFU_AlarmRingerOff( grid%alarms( ialarm ), rc=rc )
      ENDIF
#endif
   ELSE IF ( ... ) THEN
   ...
END DO
... write restart data ...
... look for boundary data ...
```

Output surface fluxes data (→ write history data)

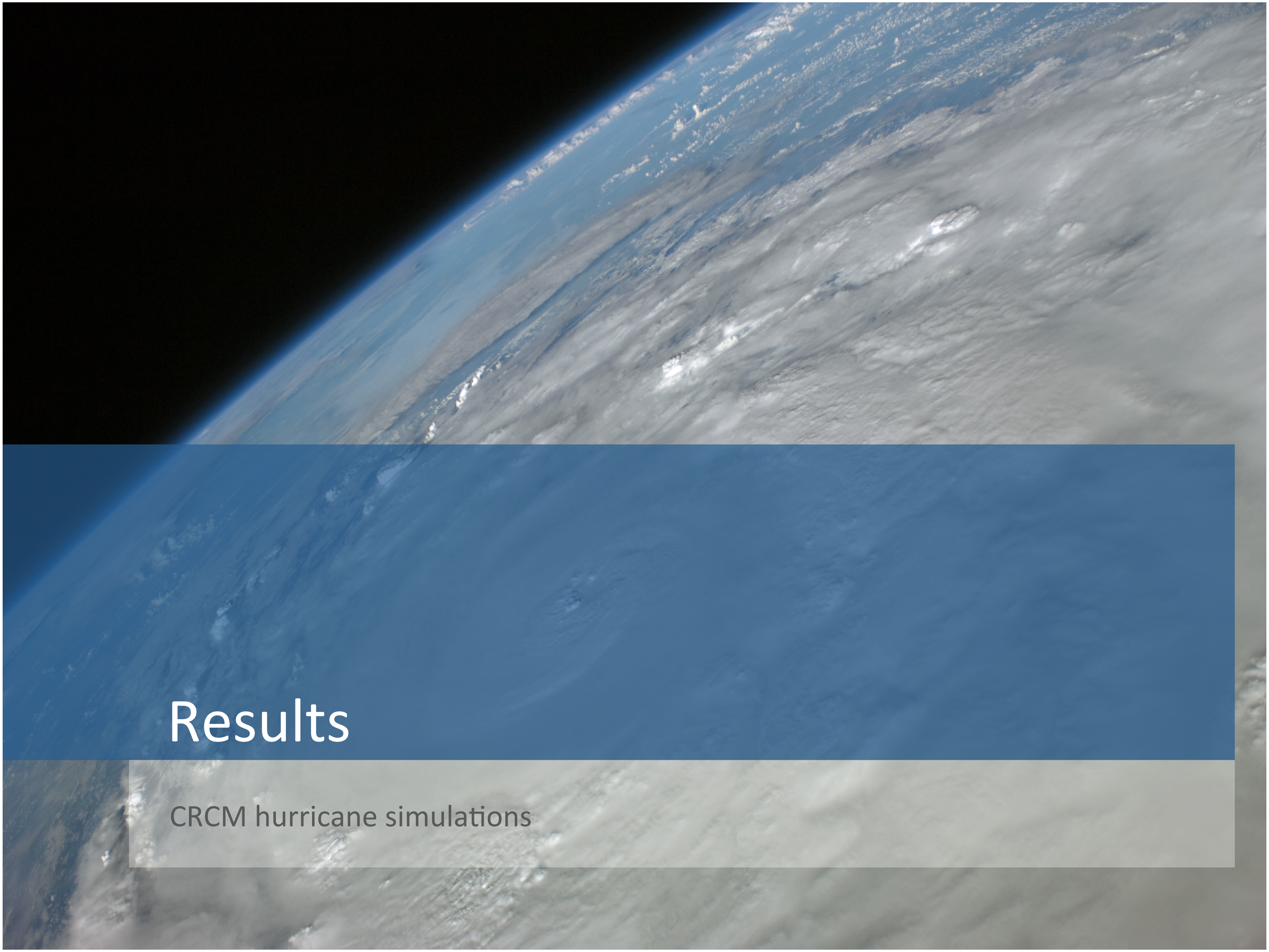
Input SSTs (→ CALL med_read_roms_sst)

Input: ROMS

```
MyAppCPP = TAMU

TAMU_start_year   = 1981
TAMU_start_month  = 05
TAMU_start_day    = 01
TAMU_start_hour   = 00
TAMU_start_minute = 00
TAMU_start_second = 00

TAMU_coupling_interval = 60
TAMU_coupling_delay = 0
TAMU_coupling_import = T
TAMU_coupling_export = T

TAMU_grid_input  = wrfinput_d01

TAMU_input  = cpl_flx.nc
TAMU_output = cpl_sst.nc
```

Input: WRF

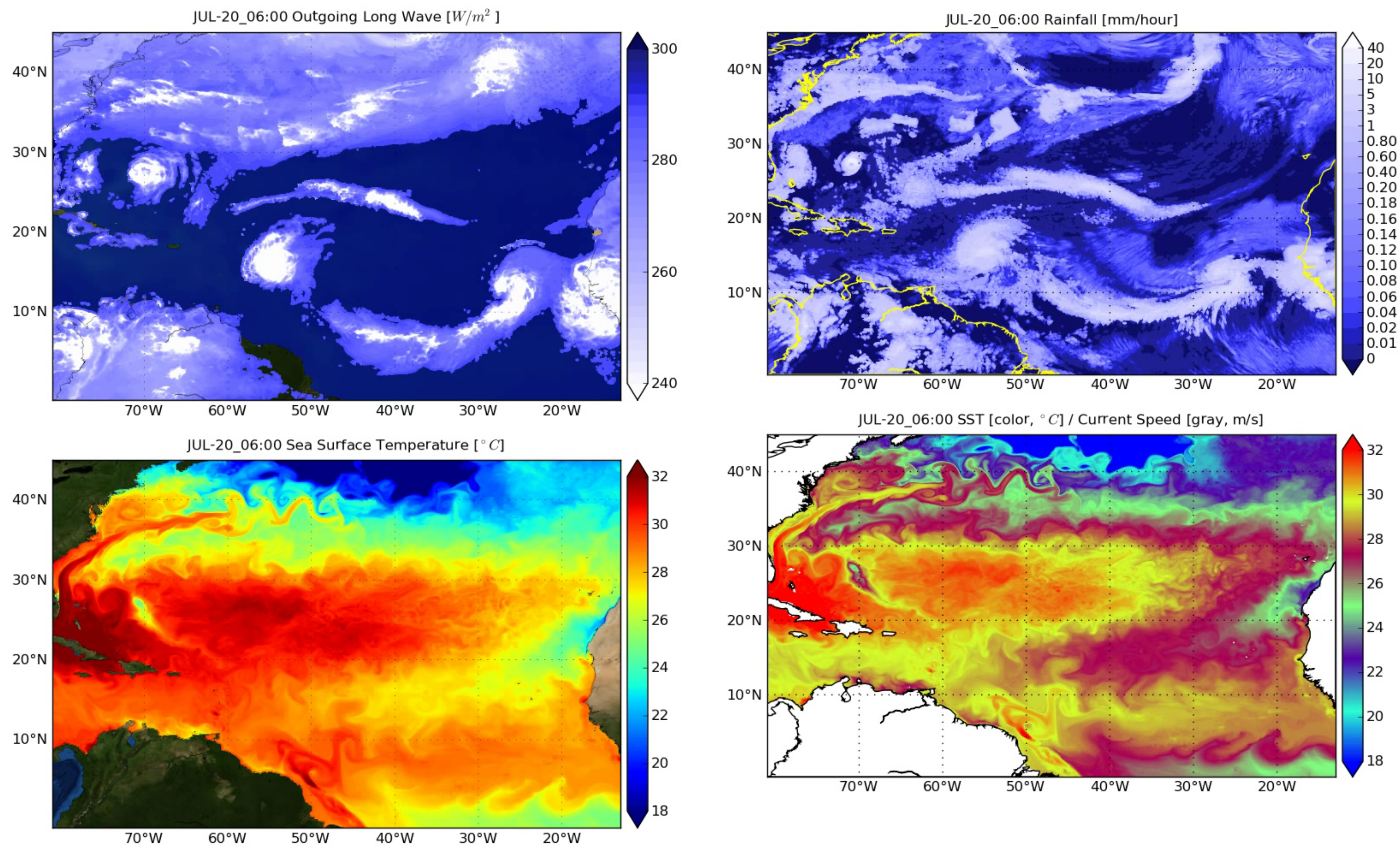
```
&time_control
...
 io_form_auxinput4 = 2
 io_form_auxhist5  = 2
 auxinput4_inname  = "cpl_sst.nc"
 auxhist5_outname  = "cpl_flx.nc"
/

&roms_coupling
 coupling_interval   = 60,
 coupling_import     = .true.,
 coupling_export     = .true.,
 import_max_attempts = 100,
/
```



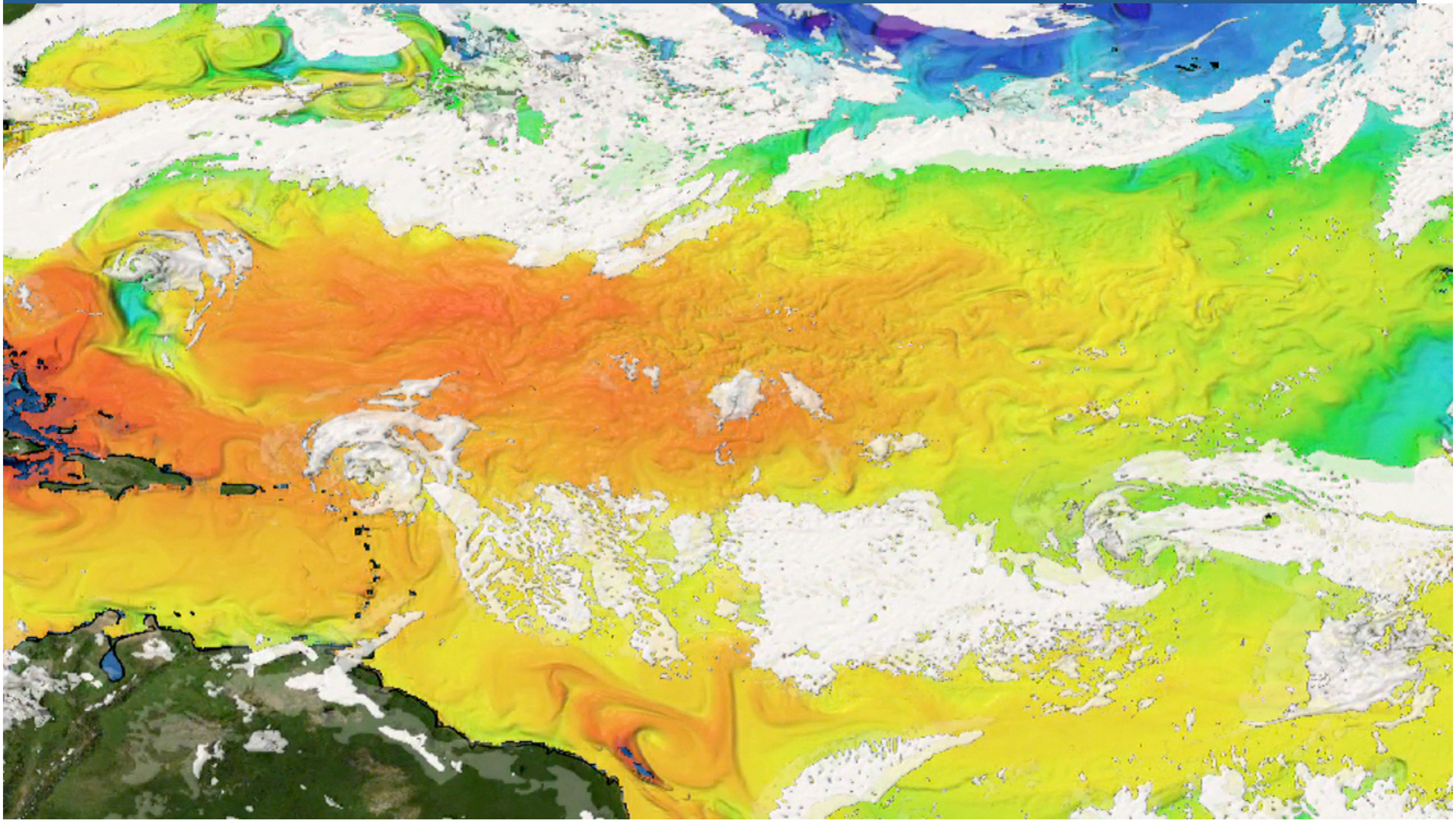
Results

CRCM hurricane simulations

Ocean/Atmosphere interaction along the path of a hurricane at 9km resolution



Ocean/Atmosphere interaction during hurricane season (9km)




Thank you!
UCAR SEA Conference 2012