Design and Use of **RegESM**: Regional Earth System Model

Ufuk Utku Turuncoglu\textsuperscript{1,2}

(1) Istanbul Technical University, Informatics Institute
(2) ICTP, ESP Section
RegESM Design

- Model components merged with ESMF/NUOPC

Two different land surface model

- Atmosphere (RegCM)
- Land (CLM)
- Ocean (ROMS / MITgcm)
- Sea ice (unnamed)
- Driver
- Initial and boundary conditions
- Natural emissions
- Anthropogenic emissions
- Initial conditions

Two different ocean model

- Ocean (ROMS / MITgcm)
- Wave (WAM)
- River routing (HD)
- Driver
- Initial conditions

ATM:
ICTP’s RegCM 4.4 / 4.5

OCN:
Rutgers Univ.
ROMS (r737)
MITgcm (63s / 64s)

WAV:
ECMWF’s WAM
4.5.3 MPI

RTM:
Max Planck’s HD
(1.0.2 modified)
Special thanks to Prof. Stefan Hagemann

# Following combination of model components can be used:
2 component: ATM-OCN, ATM-WAV,
3 component: ATM-OCN-RTM, 4 component: ATM-OCN-WAV-RTM
Driver

- It is a translator between different earth system models
- Assumed as a separated model component without any physical code
- Allows minimal code change in the model components
- It basically responsible
  - exchanging data among the model components (i.e. atmosphere, ocean, land)
  - synchronization of model components (run order)
  - performing interpolation to the data (basically exchange fields) from one component to others (different interpolation techniques can be supported)
  - applying unit conversion to exchange fields (i.e. from Kelvin to degree Celsius)
  - applying rotation of wind components (curvilinear grid)
  - applying flux correction / adjustment
Earth System Modeling Framework

• It is high-performance, flexible software infrastructure for building and coupling weather, climate and related Earth science applications.

• It defines an architecture for composing complex, coupled modeling systems and includes data structures and utilities for developing individual models.

• The basic idea behind ESMF is that complicated application should be broken up into coherent pieces or components with standardized calling interfaces (initialize, run and finalize)

• It includes toolkits for building components and applications such as regridding software (bilinear; nearest-neighbour, patch recovery and conservative), calendar management, parallel communications etc.

https://www.earthsystemcog.org/projects/esmf/
Installation

• Requirements
  • C/C++ and Fortran compiler (GNU, Intel etc.)
  • MPI (OpenMPI, Intel MPI etc.)
  • Zlib
  • HDF
  • NetCDF - Fortran/C/C++ interfaces
  • Xerces for ESMF (XML library)
  • ESMF > 7.0

• Model components
  • Every model component must support coupling
  • RegCM 4.5 already supports coupling (out-of-box)
  • ROMS, MITgcm, HD and WAM are modified to work with RegESM driver (contact with RegESM developer)

• install-deps.sh might help to install dependencies (still testing)
The installation of the coupled model can be little bit confusing at the beginning. To that end, it is better to follow the conventions based on experience.

Suggested directory structure of RegESM installation:
Bit-to-bit reproducibility

- It is very important concept and hard to reach especially for the parallel codes (MPI and/or OpenMP)

- Think about following situations:
  - reveal the added value of the model coupling, you might need to compare the standalone and coupled model results
  - having sensitivity test to get insight about the effect of the coupling interval

- The model might give slightly different results even if you keep initial and boundary conditions same due to the floating point arithmetic

- The order of the numbers in reduction operators (+, -, *, and /) might affect the result in the computer. Think about MPI_Reduce?

- The modern compilers could handle this issue by providing mechanisms to achieve bit-to-bit reproducible results
Bit-to-bit reproducibility ...

- The modern compilers have flags to overcome this issue
  - Intel: `-fp-model precise` and `-fp-model source`
  - Gnu: `-fno-fast-math`

- So, the user must compile all model components (including driver) and also ESMF library with these options

- This options is not default in ESMF and also RegCM < 4.5

- For ESMF, user need to modify following file
  `build_config/Linux.intel.default/build_rules.mk`
  it depends on used architecture and compiler (Linux+Intel)

```c
ifeq ($(ESMF_ABISTRING),x86_64_small)
ESMF_CXXCOMPILEOPTS  += -m64 -mcmmodel=small  -fp-model precise
ESMF_CXXLINKOPTS     += -m64 -mcmmodel=small  -fp-model precise
ESMF_F90COMPILEOPTS  += -m64 -mcmmodel=small  -fp-model precise
ESMF_F90LINKOPTS     += -m64 -mcmmodel=small  -fp-model precise
endif...
```
Installation …

• … model components and required libraries are already installed

• The Git repository - https://github.com/uturuncoglu/RegESM

• You might need to install git package

• To install driver:

  git clone https://github.com/uturuncoglu/RegESM.git
  cd RegESM
  ./bootstrap.sh (it will create the configuration script)
  ./configure --prefix=[HOME] --with-atm=[ATM_HOME]
    --with-ocn=[OCN_HOME]
    --with-rtm=[RTM_HOME]
    --with-wav=[WAV_HOME]
  make
  make install
Usage

- **Suggestions**
  - Setup and tune individual model components in standalone mode to find their best configuration
  - Try to add new components one by one and test the performance of the modeling system
  - You might need to do extra fine tuning of the coupled model at the end due to the complex two-way interaction between model components
  - The coupling might put extra complexity to the overall system
  - Sometimes, it might be hard to identify the added value of the coupling
  - The debug options (writing grid information in VTK format etc.) can be used to identify problems
  - Check stdout and PET* log files to find the error. In general, problem can be identified easily by looking those files
  - If you really think that there is a bug, send a mail to me 🌟
Driver Configuration File (namelist.rc)

• It is used to control driver (RegESM)
  • Resource distribution among model components (PET: Persistent Execution Thread - CPUs)
  • PET layout (sequential vs. concurrent)
  • Coupling type (explicit vs. semi-implicit)
  • Extrapolation support (unaligned land-sea mask)
  • Debugging
  • High level definition of time related configuration of simulation (start, stop, restart time and calendar)
  • Coupling time-step to exchange data among the model components (fast vs. slow)
  • Representation of rivers (source vs. surface boundary condition)
  • List of active rivers and their configurations (locations, effective radius etc.)
Driver Configuration File (namelist.rc)

- Explicit vs. Semi-implicit (leapfrog) coupling type

It allows different interaction mechanisms between ATM and OCN model components.

- The fully implicit type coupling also exists (not supported!)
- The `CouplingType` namelist parameter is used for this purpose.
Driver Configuration File (namelist.rc)

- Fast vs. slow coupling time step

It is used to define slow coupling time step for RTM component (daily)

The **DividerForTStep** namelist parameter is used
Driver Configuration File (namelist.rc)

• Sequential vs. Concurrent

- In sequential mode, model components shares same resources and run in an order
- In concurrent model, each model has its own resource and run in parallel (load balance become important!)
- The PETLayoutOption namelist parameter is used
Exchange Field Table (exfield.tbl)

- It basically designed to simplify the definition of exchanged fields among the model components
- The flexible design allows to create different coupled modeling applications without any code change in driver and model components
- The pool of exchange fields are defined but new variables might be added w/o major code development
- The ASCII formatted exchange field table includes
- Structure:
  - It contains a separate section for each coupling direction (or interaction) such as ATM-OCN, OCN-ATM or ATM-WAV
  - Each section contains the list of fields (along with unit conversion parameters, conservation option etc.) that will be transferred between model component
Exchange Field Table (exfield.tbl)

- Four component (ATM-OCN-RTM-WAV) example:

<table>
<thead>
<tr>
<th>number of exchange field</th>
<th>coupling direction</th>
<th>support for extrapolation</th>
</tr>
</thead>
<tbody>
<tr>
<td>variable name</td>
<td>standard name</td>
<td>interpolation type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>type of source stencil</td>
</tr>
<tr>
<td></td>
<td></td>
<td>type of destination stencil</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale factor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>add offset</td>
</tr>
<tr>
<td></td>
<td></td>
<td>support for conservative interpolation</td>
</tr>
</tbody>
</table>

```plaintext
6 atm2ocn T

taux: eastward_10m_wind_stress: bilinear: cross: u:N/m2:m2/s2:cf3:0.0:F
tauy: northward_10m_wind_stress: bilinear: cross: v:N/m2:m2/s2:cf3:0.0:F
psfc: surface_air_pressure: bilinear: cross: cross: cm:mb:1.0:0.0:F
swrd: shortwave_radiation: bilinear: cross: cross: W/m^2:Cm/s:cf2:0.0:T
sflx: water_flux_into_sea_water: bilinear: cross: cross: kg/m^2:s:m/s:0.001:0.0:T
nflx: surface_heat_flux: bilinear: cross: cross: W/m^2:Cm/s:cf2:0.0:T

1 ocn2atm T
2 atm2rtm F
rnof: surface_runoff: bilinear: cross: cross: mm/s:m/s:0.001:0.0:F
snof: subsurface_runoff: bilinear: cross: cross: mm/s:m/s:0.001:0.0:F
1 rtm2ocn F
rdis: river_discharge: nearestd: cross: cross: m^3:m^3:1.0:0:F
2 atm2wav T
wdir: wind_direction: bilinear: cross: cross: m/s:m/s:1.0:0:F
ustr: friction_velocity: bilinear: cross: cross: m/s:m/s:1.0:0:F
2 wav2atm T
zo: roughness_length: bilinear: cross: cross: m:1.0:0.0:F
```
Extrapolation Support

• It is used to overcome unaligned land-sea mask problem
• Interpolation from 50 km atmosphere to 7 km ocean model

**Step 1** / Bilinear interpolation from ATM to OCN only over sea

**Step 2** / Nearest-neighbour (NN) type interpolation from OCN to OCN (It uses result of previous step)

**Step 3** / Merging step 1 and 2 to have a complete exchange field
Extrapolation Error

- The simplest type extrapolation (NN) is used

\[ f = 2 + \sin^{16}(2\theta) \cos(16\phi) \]

\( \theta \), latitude

\( \phi \), longitude

Pseudo Spherical Harmonics
SCRIP: \( L=32 \) and \( M=16 \)

Relative Error:
\([\text{Model}]/[\text{Analytic}]) - 1

LRES:
\( \text{min: } -0.207 \) / \( \text{max: } 0.071 \)

HRES:
\( \text{min: } -0.025 \) / \( \text{max: } 0.014 \)
Conservative Interpolation

- Conservative type interpolation is suitable for area-integrated fields such as water or heat fluxes

- Global conservation:

\[
\Delta = \int \text{destination field} - \int \text{source field}
\]

\[
\text{destination field} = \text{destination field} - \frac{\Delta}{A_{ocn}}
\]

bilinear + conservation
Project Home and Repository

• The open source RegESM modeling system (only driver) is hosted by GitHub

• We are currently testing Travis-ci as automatic build mechanism to have a Continuous Integration (CI) with GitHub

• Project Home: https://github.com/uturuncoglu/RegESM

• User documentation is ready for use (version 1.0e)
  • Distributed along with the source code (under docs/ directory)
  • Basic design of the modeling system
  • Requirements (libraries etc.) and their installation
  • Detailed information about the configurations files
  • Known bugs and limitations
  • Bugs:
    • Issue tracking is also handled by GitHub
    • Please report any bug or new feature request by GitHub
Questions !!!

ufuk.turuncoglu@itu.edu.tr
http://faculty.itu.edu.tr/turuncoglu/