Community Coupler C-Coupler1 User's Guide

Edition 2

Li Liu, Ruizhe Li, Cheng Zhang, Guangwen Yang, Bin Wang

liuli-cess@tsinghua.edu.cn
lrz04@mails.tsinghua.edu.cn
zhang-cheng09@mails.tsinghua.edu.cn
Ministry of Education Key Laboratory for Earth System Modeling,
Center for Earth System Science (CESS),
Tsinghua University, Beijing, China

21st, January, 2015

Content

1 Introduction

1.1 Brief introduction to coupler

Climate system models (CSMs) or earth system models (ESMs) are fundamental tools for global climate change study. There are always coupled models consisting of several separate interoperable component models, which are coupled together with a coupler, to simultaneously simulate the variation of the atmosphere, oceans, land surface, sea ice, etc. With the fast development of science and technology, there are more and more coupled models as well as component models in the world. For example, in the Coupled Model Intercomparison Project Phase 5 (CMIP5), there were more than 50 coupled model versions.

A coupler (Valcke et al., 2012) is a key component in a coupled model. It achieves interactions and parallel computing among multiple component models, and controls the integration of the whole coupled model. It also provides a platform to enable scientists and engineers to cooperate together. Most of state-of-the-art CSMs and ESMs are constructed with couplers. With more and more component models (e.g., land ice model, chemistry model, and biology model) to be added into ESMs, couplers become more and more important to the development of ESMs.

There are several existing couplers before Coupler1, e.g., OASIS (Redler et al., 2010; Valcke, 2013), MCT (Larson et al., 2005; Jacob et al., 2005), ESMF coupler (Hill et al., 2004), FMS coupler (Balaji et al., 2006), Scup (Yoshimura and Yukimoto, 2008), CPL6 (Craig et al., 2005), CPL7 (Craig et al., 2012), etc.

1.2 C-Coupler project

C-Coupler is an open source community coupler developed in China. The C-Coupler project was initiated in 2010. There was no coupler developed for ESMs in China before, and CPL6, MCT and OASIS are often used for coupled model development in China. For example, in CMIP5, almost all the coupled models

developed by Chinese institutions use CPL6.

C-Coupler is targeted to provide various coupling functions for constructing coupled models and targeted to be able to integrate various models on the same model platform for sharing. It therefore contains a library (called the C-Coupler library) with functions for coupling a number of component models together and a uniform runtime environment (called the C-Coupler platform) with scripts and configuration files for creating, configuring, compiling and running model simulations. In detail, the targeting functions and features of C-Coupler include:

1) Flux calculation. C-Coupler can integrate flux algorithms, such as the algorithms for calculating the fluxes between atmosphere and oceans.
2) 3-D coupling. Besides traditional 2-D coupling, C-Coupler will provide 3-D data communication and 3-D data interpolation to support the coupling of 3-D fields.
3) Model nesting. C-Coupler will support one-way even two-way model nesting to facilitate the work of nesting regional models.
4) Ensemble. C-Coupler will enable multiple ensemble members in a model simulation to be run simultaneously.
5) Efficient parallelization. C-Coupler itself will be parallelized. It will further provide parallel I/O for the models. With the development of models as well as C-Coupler, we will continuously improve the parallel efficiency, especially when the resolutions get higher.
6) Automatic error detection. C-Coupler will provide various kinds of automatic error detection to help users to detect and fix bugs when constructing a coupled model.
7) Bitwise identical reproducibility of model simulation results. The C-Coupler platform will provide the function of managing model simulations. It will formulate the rules for bitwise identically reproducing simulation results. The information for reproducing simulation results will be automatically generated by the C-Coupler platform.
8) User friendliness. C-Coupler will provide various coupling functions in a

user friendly way. It will facilitate the works of integrating component models, constructing coupled models, integrating external algorithms and operating model simulations.

9) Modularity and extendibility. C-Coupler will have a modular and extendible software framework for coupling various component models and for integrating various algorithms, such as flux algorithms and remapping algorithms.
10) Standardization and sharing. C-Coupler will provide uniform standards for integrating component models and external algorithms, to facilitate the sharing of component models and algorithms among different coupled models.
11) Reliability. An increasing number of test cases will be preserved for the development of C-Coupler. Before releasing a new version of C-Coupler, it must pass all test cases.
12) Free and good service. C-Coupler will be constantly free for non-commercial usage. The C-Coupler team will try the best to provide good service for users. Any new requirements are possibly achieved in the future versions of C-Coupler.

1.3 C-Coupler1

C-Coupler1 is the first version of C-Coupler for public use. It is mainly programmed by C++, with more than 30000 lines of source code mainly contributed by the C-Coupler team. Guided by the targeting functions and features of C-Coupler, C-Coupler1 makes the following achievements:

1) Flux calculation. C-Coupler1 can integrate existing flux algorithms used in coupled models or in other couplers. For example, the flux algorithms used in CPL6 (Craig et al., 2005) have been integrated into C-Coupler1.
2) 3-D coupling. Besides 2-D coupling, 3-D data communication and 3-D data interpolation have implemented for 3-D coupling.

3) User friendliness. C-Coupler1 has not achieved all targets of C-Coupler in the aspect of user friendliness. However, the C-Coupler platform can facilitate the work of operating model simulations.
4) Modularity and extendibility. C-Coupler1 contains a modularized library with object-oriented code structure. It provides Fortran interfaces for coupling component models and interfaces for integrating external algorithms. External algorithms can be either in Fortran or C++.
5) Parallelization. The C-Coupler1 library has been parallelized using the Message Passing Interfaces (MPI) library. This parallelization achieves the same (bitwise identical) results when using any number of processor cores. The C-Coupler1 library supports concurrent execution of component models, and also supports the parallel execution of each component model, where 1-D and 2-D parallel decompositions on horizontal grids are supported. It uses multiple executables for coupled models. It enables direct coupling (without a separate coupler component) between component models for better parallel performance, where C-Coupler does not take unique executable and processes/processor cores.
6) Automatic error detection. C-Coupler1 provides several kinds of automatic error detection. It can check the order of calling C-Coupler interfaces in component models, check the consistency of grids between component models and remapping weights files, and check whether component models provide sufficient coupling fields, etc.
7) Reliability. C-Coupler1 has passed various test cases with more than 800 diagnostic statements intra its source code. There are several standards for testing C-Coupler, e.g.: C-Coupler must achieve the same (bitwise identical) result no matter how many processor cores are used; C-Coupler must achieve the same (bitwise identical) result in restart run; and a coupled model can achieve the same (bitwise identical) result after replacing the original coupler with C-Coupler.
8) Bitwise identical reproducibility of model simulation results. The C-Coupler

platform provides the functionality of managing model simulations. It formulates the rules and automatically records the information for bitwise identically reproducing simulation results.

9) Free and good service. After the public release of C-Coupler1, the C-Coupler team will provide good service for constructing various coupled models with C-Coupler1.

1.4 General terms for C-Coupler

There are several general terms for C-Coupler, listed out as follows.

1) **Component model**. A component model typically simulates a component or a process of the Earth system. Component models are classified into several categories, e.g., the atmosphere model, land model, ocean model, sea ice model, wave model, etc.
2) **Coupled model**. A coupled model consists of several component models.
3) **Experiment model**. An experiment model is a version of model which is ready for simulations. Generally, it can be any kind of models, such as single-column models, stand-alone component models, regionally coupled models, air-sea coupled models, nested models, CSMs, ESMs, etc. On the C-Coupler platform, we use a "*compset*" to stand for an experiment model. Experiment models are always built from existing component models or existing coupled model versions, such as the model versions in CMIP5. An experiment model can also be a new version of coupled model with several component models which have never been coupled together before.
4) **Model simulation**. A model simulation is a simulation run based on certain setups of an experiment model for scientific research purposes. A new model simulation can be evolved from existing ones.

1.5 About this user's guide

This user's guide is mainly about how to operate model simulations on the

C-Coupler platform. More details about the design and implementation of C-Coupler1 can be found in Liu et al. (2014b).

1.6 How to use the C-Coupler platform

Figure 1 shows the flowchart of operating model simulations on the C-Coupler platform. It generally takes the following major steps for a new simulation:

1) Set the environment variables of the C-Coupler platform (Section 4.1)
2) Create a simulation from a default setting (Section 4.2.1) or from an existing simulation (Section 4.2.2).
3) Check and modify simulation setting under the working directory of the simulation (Section 6).
4) Configure the simulation and then compile the source code (Section 4). Before the compilation, users can remove the binary files or executable files produced by the compilation (Section 4.4). A simulation setting package will be generated when configuring a model simulation. Such a package can be used for creating a simulation.
5) Run the new simulation (Section 4). If users want further simulations after analyzing the simulation results, they can improve simulation setting for another new simulation.

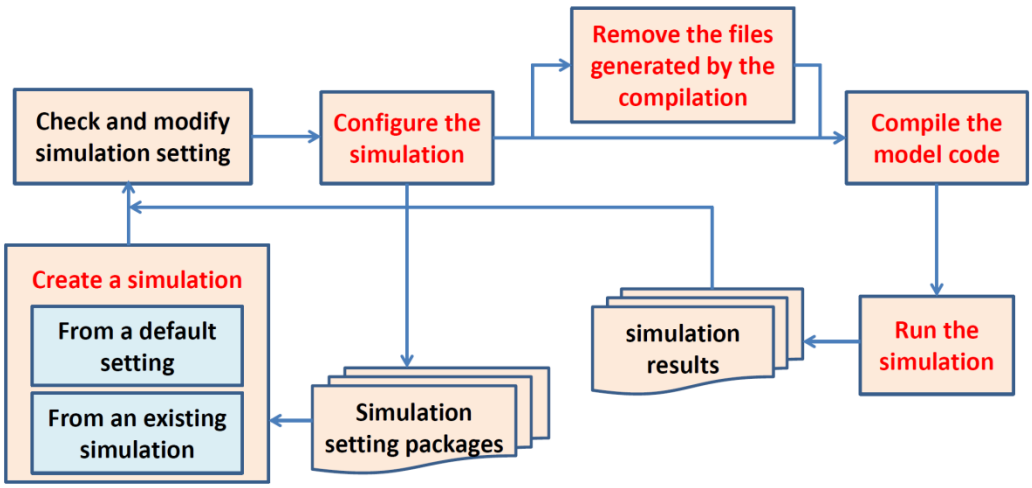

Figure 1 Flowchart of operating model simulations on the C-Coupler platform

2 Models on the C-Coupler platform

Models on the C-Coupler platform include component models, coupled models and experiment models. Any kinds of component models and coupled models can be integrated onto the C-Coupler platform. Experiment models are always built from the component models and coupled models on the C-Coupler platform.

2.1 Component models

2.1.1 Atmosphere models

2.1.1.1 GAMIL2

GAMIL is an AGCM developed mainly in the National Key Laboratory for Numerical Modeling of Atmospheric Sciences and Geophysical Fluid Dynamics (LASG), Institute of Atmospheric Physics (IAP), Chinese Academy of Sciences (CAS), which has taken part in various international model intercomparison projects and has been widely used for various science problems studies.

GAMIL is based on the Eulerian finite-difference dynamical core. Its discrete grid includes a hybrid horizontal grid with the Gaussian grid (for low resolution) or uniform grid (for high resolution) in the low and middle latitudes region and a weighted even area grid in the high latitudes and polar region, and 26-σ vertical levels (pressure normalized by surface pressure) with the model top at 2.194hPa. Its dynamical core includes a finite-difference scheme that conserves mass and effective energy (Wang et al., 2004), and a two-step shape-preserving advection scheme for the moisture equation (Yu, 1994).

GAMIL2 is the second version of GAMIL. It is the atmospheric component of the CSM FGOALS-g2. More details of GAMIL2 can be found in Li et al., (2013a).

2.1.1.2 WRF

The Weather Research and Forecasting (WRF) Model (Michalakes et al., 2004) is a next-generation mesoscale numerical weather prediction system designed to serve

both atmospheric research and operational forecasting needs. It has been widely used for weather forecasting and scientific researches. The detailed version of WRF on the C-Coupler platform is WRF3.6.

2.1.2 Ocean models

2.1.2.1 LICOM2

LICOM is the fourth generation of LASG OGCM (Liu et al. 2004a, 2004b) since the first generation developed in 1989 (Zhang and Liang, 1989). The version 2 of LICOM (LICOM2) is the ocean component of FGOALS-g2. Comparing with LICOM1.0 and LICOM1.1, LICOM2 increases the horizontal resolutions (1°x1° horizontal resolution with 0.5° meridional resolution in the tropics) and adjusts vertical resolution (10m each layer in the upper 150m) firstly (Wu et al., 2005). Secondly, the advection scheme is introduced (Xiao et al., 2006). Thirdly, the physical processes are updated or improved (Liu et al., 2012) including the mixing schemes (Canuto et al. 2001, 2002; Liu et al. 2012), solar penetration scheme (Lin et al. 2007; Lin et al. 2011), etc (Liu et al. 2012). Several updated schemes are not adopted in the latest versions of FGOALS (Lin et al. 2012). The details can be found in Liu et al (2012) and Lin et al. (2012).

2.1.2.2 POM

The Princeton ocean model (POM) is a community general numerical model for ocean circulation that can be used to simulate and predict oceanic currents, temperatures, salinities and other water properties. The detailed code version of POM on the C-Coupler platform is a parallel version developed by Wang et al. (2010).

2.1.3 Land surface models

2.1.3.1 CoLM and CLM3

CoLM (Dai et al., 2008) and CLM3 (Oleson et al., 2004) are two main branches of the early Common Land Model (Dai et al., 2003, 2004) for further development of land surface models. The Common Land Model is a third-generation land surface

model based on the Biosphere-Atmosphere Transfer Scheme (BATS), the Institute of Atmospheric Physics Land Surface Model (IAP94) and the Land Surface Model (LSM). Due to its outstanding simulation performance, it was selected to be the land surface model in the CCSM (Community Climate System Model) model from the National Center for Atmospheric Research (NCAR), and was further developed by NCAR. At the same time, the Common Land Model was further developed by the cooperation between Beijing Normal University and University of Texas at Austin.

2.1.4 Sea ice models

2.1.4.1 CICE4-LASG

The CICE4-LASG is an improved version of CICE4 (Los Alamos sea ice model version 4.0 (http://climate.lanl.gov/Models/CICE)). It is the sea ice component in the FGOALS-g2 model. The main difference between CICE4-LASG and CICE4 is the simulation of sea ice salinity: CICE4 uses an unchanged vertical salinity profile while CICE4-LASG formulates salinity variations and designs a simple parameterization accordingly. More information about CICE4-LASG could be found in Wang et al. (2009) and Liu (2010).

2.1.5 Wave models

2.1.5.1 MASNUM

MASNUM (Laboratory of ***MA***rine ***S***ciences and ***Nu***merical ***M***odeling, State Oceanic Administration (SOA)) is a third-generation wave model (Yang et al, 2005) which has already been used in forecasting systems. It is also the wave component of FIO-ESM.

2.2 Coupled models

2.2.1 FGOALS-g2

FGOALS-g2 (Flexible Global Ocean-Atmosphere-Land System Model, Grid-point version2) (Li et al., 2013b) is a coupled CSM developed mainly in the

LASG IAP. It consists of GAMIL2, LICOM2, CICE4_LASG, and CLM3, using NCAR coupler CPL6 for model coupling. Recently, FGOALS-g2 participated in several model intercomparison projects, e.g., CMIP5 and PMIP3 (Paleoclimate Modelling Intercomparison Project, Phase 3), and showed good performance on ENSO (Bellenger, et al., 2013) and other aspects.

2.2.2 CESM

CESM (Community Earth System Model) is a fully-coupled, community, global climate model that provides state-of-the-art computer simulations of the Earth's past, present, and future climate states (Hurrell et al., 2013). It is a CMIP5 model that has been widely used for scientific researches. The detailed version of CESM on the C-Coupler platform is CESM1.2.1.

2.3 Experiment models

2.3.1 FGOALS-gc

FGOALS-gc is a version of FGOALS-g2, by replacing the coupler CPL6 with C-Coupler1. In FGOALS-gc, C-Coupler1 integrates the scientific algorithms in CPL6 (such as flux algorithms) and works as a component that takes physical processor cores. FGOALS-gc achieves the same (bitwise identical) simulation result no matter the number of processor cores for each component. In the original version, FGOALS-gc achieves the same (bitwise identical) simulation result with FGOALS-g2. In the latest version of FGOALS-gc, GAMIL2 is an updated version with 2-D parallel decomposition (Liu et al., 2014b) and with the upgrade of TSPAS advection implementation.

2.3.2 GAMIL2-sole

GAMIL2-sole is a version of GAMIL2 for sole component run. GAMIL2-sole shares the same version of GAMIL2 with FGOALS-gc. GAMIL2-sole achieves the same (bitwise identical) simulation result no matter how many processor cores are

used.

2.3.3 GAMIL2-CLM3

GAMIL2-CLM3 is a coupled model consisting of GAMIL2 and CLM3. In GAMIL2-CLM3, direct coupling (without a separate coupler component) is used. GAMIL2-CLM3 achieves the same (bitwise identical) simulation result no matter how many processor cores are used for each component.

2.3.4 CESM1_2_1-sole

CESM1_2_1-sole is a “standalone” version of CESM1.2.1 without the coupling with other component models. The C-Coupler platform can be utilized for operating the simulations of CESM1.2.1. Less than 10 lines of code are added to CESM code for building CESM1_2_1-sole. For the specific details for operating CESM1_2_1-sole, please refer to Section 9.

3 Directory structures of C-Coupler platform

3.1 Main directories of C-Coupler platform

The C-Coupler platform consists of two packages: the *C-Coupler model platform* and *input data*. The *C-Coupler model platform* contains three main directories: *models*, *scripts*, and *config*. *models* is used to store the source code of each component model and each library. *scripts* is used to store the scripts for operating the simulations, including creating, configuring, compiling, running and downloading a model simulation, and differing two simulations. *config* is used to store the configuration information for component models, experiment models, machines, etc. After creating a new model simulation, its working directory is created. **Please do not change the directory structure of the *C-Coupler model platform* and *input data* as possible.**

3.2 Directory structure of *input data*

input data includes several sub directories, e.g., *atm*, *ocn*, *lnd*, *sice*, *cpl* and *grids*, to respectively store the input data for atmosphere models, ocean models, land surface models, sea ice models, couplers and model grids. Each sub directory stores the input data files of several component models. There are two scripts *register_inputdata.sh* and *register_inputdata.csh* under *input data*, which are used for setting environment variables of the C-Coupler platform (Section 4.1).

3.3 Directory structure of *models*

Similar to *input data*, *models* includes several sub directories, e.g., *atm*, *ocn*, *lnd*, *sice*, *cpl* and *libs*, to respectively store the source code of atmosphere models, ocean models, land surface models, sea ice models, couplers and libraries, etc. Each sub directory stores the source code of several component models or libraries.

The code of the C-Coupler library is stored under the directory

models/libs/c_coupler/.

3.4 Directory structure of *scripts*

Under *scripts*, there are several scripts for operating model simulations: *create_newcase*, *configure*, *compile*, *runcase*, *clean*, *checkout_experiment*, *differ_experiments*, *register_platform.csh*, *register_platform.sh*. The former four scripts are used for creating, configuring, compiling and running a model simulation. *Clean* is used for deleting the compilation generated files. *checkout_experiment* is used for downloading the whole simulation environment (including input data files, model codes, input parameters and computing environment) of a model simulation from a set of simulation resource servers. *differ_experiments* is used for obtaining the difference between the simulation environments of two model simulations. *register_platform.csh* and *register_platform.sh* are used for registering environment variables for the *C-Coupler model platform*. Besides these scripts, there is a sub directory *utils* which stores the common tools for these scripts.

3.5 Directory structure of *config*

config includes several sub directories, e.g., *atm*, *ocn*, *lnd*, *sice*, *cpl*, *lib*, *compset* and *common*, which stores default configuration information for the atmosphere models, ocean models, land surface models, sea ice models, couplers, libraries, experiment models and common tools. Each sub directory stores the configuration information of several models or libraries.

3.6 Structure of working directory of a model simulation

Each model simulation has its own working directory. Users should go into the working directory to configure, compile and run the simulation. After configuring, under the working directory, there are four scripts, e.g., *configure*, *compile*, *clean* and *runcase*, and several sub directories, e.g., *config*, *configure_history*, *job_logs* and *run*.

These four scripts are originally copied from the *scripts* directory of the *C-Coupler model platform*. *config* stores the configuration information related to the current model simulation. *configure_history* stores the simulation setting packages after every configuration by users, for reproducing model simulation results. *job_logs* stores the log of every model run. *run* stores the files related to model compiling and model run, for each component model and library, including namelist, input data, output data, the source code for the compilation, and the source code produced by the compilation. Given a component model with the name *model_name* and the type (e.g., atm, ocn, lnd, sice) *model_type*, the namelist, input data as well as output data are under the directory *run/model_type/model_name/data/*, the files produced by the compilation are under the directory *run/model_type/model_name/obj/*, the executable file is under the directory *run/model_type/model_name/exe/*, the log files of compilation are under the directory *run/model_type/model_name/build_logs/*, and the log files of component model run are under the directory *run/model_type/model_name/run_logs/*.

4 Create, configure, compile and run a model simulation

As shown in Section 1.6, users can create, configure, compile and run a model simulation.

4.1 Environment variables

Table 1 lists out the environment variables for a model simulation. *COMPSET* and *MACH* are specific to each model simulation. *DATAROOT*, *CODEROOT*, *CONFIGROOT* and *SCRIPTSROOT* are common environment variables that can be shared by multiple model simulations. Users should use command such as "source *register_platform.sh*" or "source *register_platform.csh*" to set *CODEROOT*, *CONFIGROOT* and *SCRIPTSROOT*, and use command such as "source *register_inputdata.sh*" or "source *register_inputdata.csh*" to set *DATAROOT*.

Table 1 Environment variables for a model simulation

Environment variables	Descriptions
COMPSET=*compset_name*	*COMPSET* is the name of an experiment model, which specifies the experiment model for the model simulation
MACH=*machine_name*	*MACH* specifies the machine to run the model simulation
DATAROOT=*data_dir*	*DATAROOT* specifies the root directory of the input data, which is always the directory *input data* of the C-Coupler platform
CODEROOT=*code_dir*	*CODEROOT* specifies the directory of *models* in the *C-Coupler model platform*
CONFIGROOT=*config_dir*	*CODEROOT* specifies the directory of *config* in the *C-Coupler model platform*
SCRIPTSROOT=*scripts_dir*	*SCRIPTSROOT* specifies the directory of *scripts* in the *C-Coupler model platform*

4.2 Create a model simulation

There are two approaches to creating a model simulation: from a default setting or from an existing simulation. We encourage users to use the second approach, to minimize the modifications of configuration information for a new simulation.

4.2.1 New model simulation from default setup

To create a new model simulation from a default setting, users should run the command "***./create_newcase simulation_env***" under the directory *scripts* of the *C-Coupler model platform*, where *create_newcase* is the script for creating the simulation, and *simulation_env* is a parameter file with environment variables of the simulation, including *COMPSET*, *CASEROOT* and *MACH*, where *CASEROOT* specifies the location of the working directory of the new simulation.

For example, with the environment variables in Table 2, users can create a simulation of model FGOALS-gc under the directory */home/liuli/model_simulations/FGOALS-gc.RCP26* on the machine Tansuo100.

Table 2 An example of environment variables for creating a new simulation

```
COMPSET=FGOALS-gc
CASEROOT="/home/liuli/model_simulations/FGOALS-gc.RCP26"
MACH=Tansuo100
```

4.2.2 New model simulation from an existing simulation

The C-Coupler platform provides the utility of creating a new simulation from an existing simulation. Given an existing simulation *old_sim* and its simulation setting package *old_sim.setup.tar*, users can create a new simulation *new_sim* under the following steps:

1) Source *register_platform.sh* (or *register_platform.csh*) and *register_inputdata.sh* (or *register_inputdata.csh*) to set the environment variables (Section 4.1).

2) Create the working directory of *new_sim*.
3) Go to the working directory of *new_sim* and then untar *old_sim.setup.tar*.
4) Modify environment variable *MACH* in configuration file *config/common/env* if necessary.
5) Modify configuration information of the new simulation (Section 6) if necessary.

4.3 Configure a model simulation

To configure an simulation, users should run command "*./configure*" under the working directory of a model simulation. All information of the whole simulation environment is automatically packed into a *.tar* file (called a simulation setting package) under sub directory *configure_history*, where the time (in the format of YYYYMMDD-HHMMSS) of configuring is recorded in the file name as a keyword. Before configuring, users should view and modify the configuration information (Section 6).

4.4 Compile a model simulation

To compile an experiment model, users should run the command "*./compile [-print_on_screen]*" under the simulation working directory of the model. The log files of the compilation are stored under the sub directory *run* of the simulation working directory. When *-print_on_screen* is specified, the log of the compilation will be printed to the screen. Users can run the command "*./clean para*" to remove the files produced by the compilation, where *para* means the parameter for *clean*. The parameter can be the name of a component model or a library, or *all*. For command "*./clean all*", all files produced by compilation will be removed.

4.5 Run a model simulation

To run a model simulation, users should run the command "*./runcase*" under the

simulation working directory. Note that, "*./runcase*" is common to any kind of simulations on any kind of hardware platforms.

5 Configuration information of models

5.1 Configuration information of component models

The configuration information of a component model is under the directory *config/model_type/model_name/* of the *C-Coupler model platform*, where *model_type* and *model_name* are the type and name of the component model respectively. There are six files, including *field_buf_register.cfg*, *private_field_attribute.cfg*, *config.sh*, *form_src.sh*, *compiler.cfg* and *build.sh*. File *field_buf_register.cfg* records the fields that should be registered by the component model. File *private_field_attribute.cfg* records the information of private model fields (not coupling fields) of the component model. File *config.sh* is a script specifically for configuring the component model. It can generate namelist files. File *form_src.sh* is a script that specifies source code directories of the component model. File *compiler.cfg* is a configuration file that specifies private compiling options of the component model. File *build.sh* is a script for compiling the component model, which can generate some header code files.

5.2 Configuration information of experiment models

The configuration information of an experiment model is under the directory *config/compset/model_name/* of the *C-Coupler model platform*, where *model_name* is the experiment model. It contains a subdirectory *coupler* and a file *compset.settings*. The subdirectory *coupler* stores the configuration files for driving the C-Coupler library. There are two sections in the file *compset.settings*: *common* and *model*. Section *common* records the overall information of the experiment model, including component models, libraries and a template of namelist. Section *model* records the information of each component model, including *realname* (real name of the component model), *grid* (label of model grid), and *type* (type of component model, including *atm*, *ocn*, *lnd*, *sice*, *wave*, *cpl*, etc.), etc.

6 Configuration Information of a Model Simulation

We suggest users go to the private working directory of a model simulation to check and modify the configuration information. Please **do not** modify the shared directories *config* and *scripts* of the *C-Coupler model platform*. Please **do not** modify any file under the subdirectory *run* of simulation working directory, such as *namelist* files, because they are temporal and generated by the scripts *configure* and *compile* according to the configuration files under the directory *config* of the simulation working directory. Note that, the experiment model (*compset*) for an existing simulation cannot be modified. If users want to use another experiment model, please create a new simulation. **For the reproducibility of model simulation results, if the configuration information of a simulation is modified, users should reconfigure the simulation (run the command "*./configure*") before running it.** Then configuration information of the simulation will be stored automatically to facilitate the reproduction of the model simulation results.

6.1 Name and working directory of a model simulation

The name of a model simulation is the same as the name of the working directory. When users want to modify the simulation name or migrate the simulation to another directory, please take the following two steps:

1) Move or copy the simulation to a new working directory.
2) Configure the simulation under the new working directory with the command "./configure". Then, users can run the simulation after the compilation.

6.2 Initial run, restart run or continue run

There are four choices to run a simulation: *initial* run, *restart* run, *continue* run and *hybrid* run. For *initial* run, please set variable *run_type* in Section *common* of file *config/common/case.conf* to "*initial*" and then configure the simulation. For *continue* run, please set variable *run_type* in Section *common* of file *config/common/case.conf*

to “*continue*” and then configure the simulation. For the *restart* run or *hybrid* run from the original simulation, please take the following steps:

1) Set variable *run_type* in Section *common* of file *config/common/case.conf* to “*restart*” or “*hybrid*”.
2) Set variables *original_case_name*, *original_config_time*, *run_restart_date* and *run_restart_second* in Section *common* of file *config/common/case.conf*. *Original_case_name* and *original_config_time* specify the name and configuration time of the original simulation. *run_restart_date* and *run_restart_second* specify the simulation time restarted from. The format of variable *run_restart_date* is “YYYY-MM-DD”. When these variables are different from that in the restart data files, the simulation cannot be restarted.
3) Configure the simulation.

The difference between the *restart* run and *hybrid* run is that: the *restart* run will continue the original simulation while the *hybrid* run start a new simulation based on the original simulation.

6.3 Start time and stop time

The variables *run_start_date* and *run_start_second* in the section *common* of the file *config/common/case.conf* specify the start time of the model simulation, while the *run_stop_date* and *run_stop_second* specify the stop time. The format of variables *run_start_date* and *run_stop_date* is “YYYY-MM-DD”. Please configure the model simulation after the modification.

6.4 Frequency of writing restart file

Variables *rest_freq_unit* and *rest_freq_count* in the section *common* of the file *config/common/case.conf* specify the frequency of writing restart files. *rest_freq_unit* is the unit of frequency that can be “*seconds*”, “*days*”, “*months*” or “*years*”. For example, given *rest_freq_unit=days* and *rest_freq_count*=5, the frequency of writing

restart files is once per 5 days. Please configure the model simulation after the modification.

6.5 Nonleap year and leap year

Variable *leap_year* in the section *common* of file *config/common/case.conf* specifies whether leap year is on. If it is “true”, leap year is on. Otherwise, leap year is off. Please configure the model simulation after the modification.

6.6 Simulation description

Variable *case_desc* in the section *common* of file *config/common/case.conf* specifies the description of a model simulation. This description can be written into the output data files through a C-Coupler API (application programming interface). Please configure the model simulation after its modification.

Simulation description is important for comparing, evaluating and reproducing simulation results. We propose the authors of a model simulation leave their names (even emails) into the description.

6.7 Parallel settings

Variables *num_thread*, *num_x_proc*, *num_y_proc* and *num_total_proc* in the section *model* of file *config/common/case.conf* specify the parallel setting of the corresponding component model. *num_thread* is the number of threads (e.g., OpenMP threads). *num_x_proc* and *num_y_proc* are the numbers of processes on the two directions, which are useful for 2-D parallel decomposition. They must be set or unset at the same time. *num_total_proc* is the number of the total processes. If *num_x_proc* and *num_y_proc* are set, *num_total_proc* must be the multiple of *um_x_proc* and *num_y_proc*.

Please configure the model simulation after modifying the parallel setting. For a component model that uses macro definitions to control the parallel setting, please

remove the compilation generated files and then recompile the model code.

6.8 Compilation options

The compilation options of a component model or library consist of the local compilation options specifically for the component model and common compilation options for the whole experiment model. Given a component model *comp_name* with type *comp_type*, its local compilation options are specified in file *config/comp_type/comp_name/compiler.cfg*. Given a library with name *lib_name*, its local compilation optimizations are specified in file *config/lib/lib_name/compiler.cfg*. The common compilation options depend on the machine (computer platform) that the model simulation runs on. Given a machine with name *mach_name*, the common compilation options are specified in the file *config/common/machine/mach_name/common_compiler.mach_name.cfg*. Please clean the corresponding compilation generated files after modifying the compilation options. The following table shows the meaning of main variables in the *common_compiler.mach_name.cfg* file.

Table 3 Descriptions of the main variables in the file *common_compiler.mach_name.cfg*

Variable	Meaning
FC	Fortran compiler
CC	C compiler
CXX	C++ compiler
CPP	Tool for precompiling according to the macro definitions
AR	Tool for building a library
LD	Linker of all objective files for generating an executable file
CFLAGS	Compiling options for CC (C compiler)
FFLAGS	Compiling options for FC (Fortran compiler)
CXXFLAGS	Compiling options for CXX (C++ compiler)

CPPFLAGS	Compiling options for CPP (precompiling)
LDFLAGS	Compiling options for LD (linker)
ULIBS	Compiling options of linking libraries for LD (linker)
NETCDFINC	Directory of NetCDF header files
NETCDFLIB	Directory of NetCDF library files
MPIINC	Directory of MPI compiler header files
MPILIB	Directory of MPI compiler library files

6.9 Directories of source code

On the C-Coupler platform, each component model or library has its own directories of source code. Given a component model *comp_name* of type *comp_type*, its directories of source code are specified in the file *config/comp_type/comp_name/form_src.sh*. Given a library with name *lib_name*, its directories of source code are specified in the file *config/lib/lib_name/ form_src.sh*. Please clean (use the command "./clean *comp_name*") the related compilation generated files after modifying the source code directories.

6.10 Namelist of component model

Given a component model *comp_name* of type *comp_type*, its namelist is generated by a script file *config/comp_type/comp_name/config.sh*. Please modify *config.sh* to modify the namelist. Please do not modify the namelist file under the *run* directory of the model simulation.

7 Computer systems

7.1 Recommended system setups

We recommend users to use Linux operating system for the model simulations on the C-Coupler platform. We propose users to use the Intel compiler (including Fortran, C and C++) with the version 11 or higher and use the Intel MPI library with the version 3 or higher for compiling the source code. For NETCDF library, we propose to use the version around 4.

7.2 High-performance computers

Currently, the C-Coupler platform successfully runs on several high-performance computers. Under the directory *config/common/machine* of the *C-Coupler model platform*, there are 5 machines corresponding to the environment variable *MACH*, including *generic_linux*, *Tansuo100*, *IAP_avatar*, *FIO_HPC* and *Tianhe1*. *generic_linux* stands for common single-node server with multiple cores. *Tansuo100* stands for the high-performance computer Tansuo100 in Tsinghua University. *IAP_avatar* stand for the high-performance computer avatar in Institute of Atmospheric Physics (IAP), Chinese Academy of Sciences (CAS). *FIO_HPC* stands for a high-performance computer in the First Institution of Oceanography (FIO), State Oceanic Administration (SOA). *Tianhe1* stands for Tianhe 1 high-performance computer.

If users want to use the C-Coupler platform on a new high-performance computer, we'd like to help the migration to that computer and integrate the corresponding machine configuration information into the C-Coupler platform.

8 Operations for bitwise identical reproducibility

The C-Coupler platform provides specific supports for achieving the bitwise identical reproducibility of model simulation results. First, when scientists configure a model simulation, a simulation setting package that records the information of the whole simulation environment is generated. Second, using a simulation setting package, scientists can download the corresponding model simulation and then reproduce it. Third, given two simulation setting packages, scientists can get the differences between the corresponding two model simulations.

8.1 Generation of a simulation setting package for reproducibility

In order to achieve the bitwise identical reproducibility of simulation results, please use the command "*./configure -checksum*" to configure a model simulation. This command is generally time-consuming because it scans each input data file to calculate the checksum and then stores the checksum into the simulation setting package. **"*-checksum*" is important to the reproducibility because it guarantees that users can obtain exactly the same input data files used in the original simulation in a reproduction. Note that the command "*./configure*" does not calculate and store the checksums.**

8.2 Downloading a model simulation

Users can download the *C-Coupler model platform*, model codes and input data files from a set of simulation resource servers, for recreating the whole simulation environment of an existing model simulation. Given a simulation setting package *exp_setup.tar*, users can download the corresponding simulation environment using the command "*./checkout_experiment [-bypass_inputdata] [-bypass_platform] exp_setup.tar local_env*", where *checkout_experiment* is the script for the downloading (Section 3.4), and *local_env* is a file with the environment variables described in Table 4. When *-bypass_inputdata* is specified in the command, all input

data files will not be downloaded. When *-bypass_platform* is specified, the *C-Coupler model platform* and all model codes will not be downloaded.

Figure 2 shows the flowchart of downloading the code of a component model, a library or the *C-Coupler model platform*. Figure 3 shows the flowchart of downloading an input data file. The process of downloading may require the interactions with users, such as importing username and password, and specifying another simulation resource server path or a local path.

Table 4 Environment variables for downloading a simulation

Environment variables	Necessary or optional	Description
DATAROOT=*dataroot_dir*	Necessary	Local directory for storing the downloaded input data files of the simulation
PLATFORMROOT=platformroot_dir	Necessary	Local directory for storing the downloaded *C-Coupler model platform*
REQUIRE_CHECKSUM=*"no"* or *"yes"*	Optional. Default value is "*yes*"	When the value is "*no*", the checksum of the downloaded each input data file will not be checked, so as that bitwise identical reproducibility is not guaranteed. Any other values but not "*no*" equals to "*yes*".
EXTRA_DATASRC_LOW_PRIORITY=*path_file1*	Optional	*path_file1* is an external file. Each line in it specifies a local path or a remote server path for downloading the input data files. The format of specifying a path is " *[http://[username[:password]@]host[:port]/]path[*reversion]* ".
EXTRA_DATASRC_HIGH_PRIORITY=*path_file2*	Optional	*path_file2* is similar to *path_file1*. The paths specified in *path_file2* have higher priority than the *path_file1*.

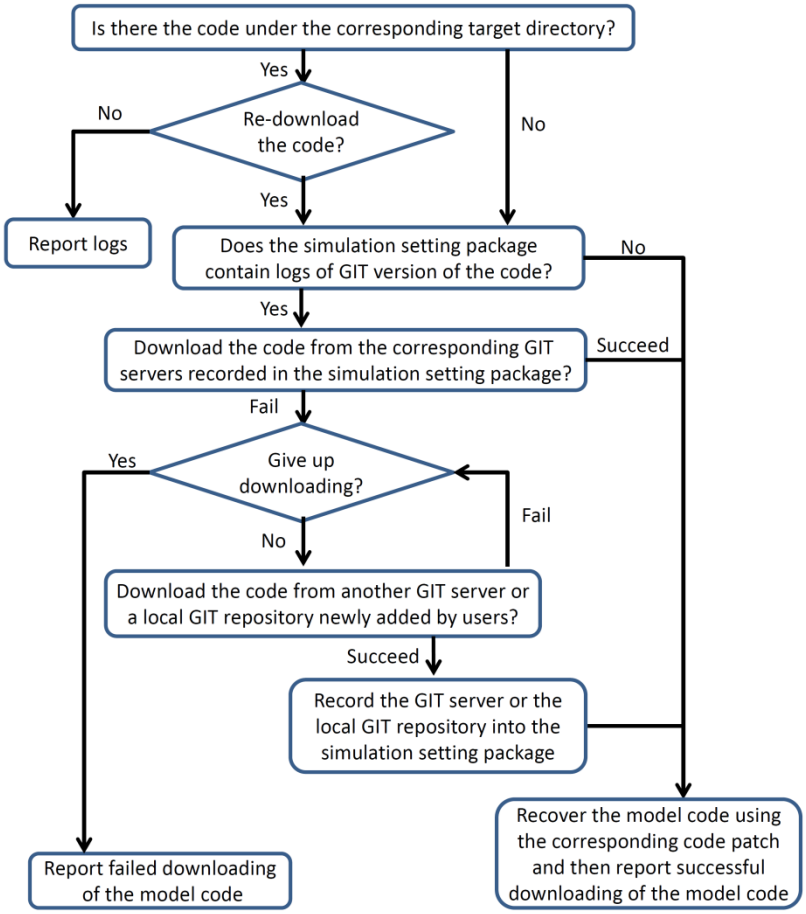

Figure 2 Flowchart of downloading the code of a component model, a library or *the C-Coupler model platform*.

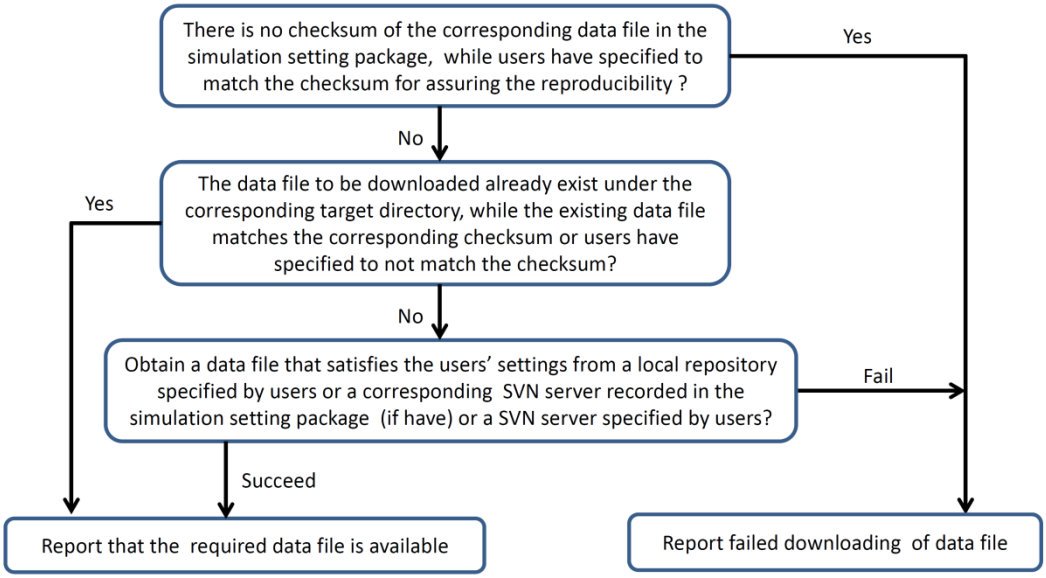

Figure 3 Flowchart of downloading an input data file.

8.3 Differing two model simulations

Given two simulation setting packages *exp_setup1.tar* and *exp_setup2.tar*, users can obtain the differences between the corresponding two model simulations using the command "*./differ_experiments [-output dir] exp_setup1.tar exp_setup2.tar*", where *differ_experiments* is the script of differing (Section 3.4). When *-output dir* is not specified, a default directory will be created for storing the differences. *differ_experiments* will implicitly invoke the script *checkout_experiment* for downloading the codes of component models, libraries and the *C-Coupler model platform*. Therefore interactions with users may be required.

9 CESM1_2_1-sole on the C-Coupler platform

The type of CESM1_2_1 is marked as “cesm”. In the working directory of a CESM simulation, the configuration files of CESM are under the directory “config/cesm/cesm/”, the directory specific for CESM under “*run/*” is “*run/cesm/cesm*”. Users can use almost all common operations on the C-Coupler platform to operate the CESM1_2_1-sole simulations, except the default creation of a CESM simulation case, modification of parallel settings and input parameters, and location of log files of compilation and execution.

9.1 Default creation of a simulation case

When using command “***./create_newcase simulation_env***” under the directory *scripts* of the *C-Coupler model platform* to create a new model simulation of CESM1_2_1-sole, please set “*COMPSET*” to “*CESM1_2_1-sole*” and also set other two parameters “*CESM_RES*” and “*CESM_COMPSET*”, for example, as shown in Table 5. “*CESM_RES*” means the resolution of CESM, which corresponds to the parameter “*-res*” of the CESM script “*create_newcase*”. “*CESM_COMPSET*” means the set of components for a CESM simulation, which corresponds to the parameter “*-compset*” of the CESM script “*create_newcase*”.

Table 5 An example of environment variables for creating a new simulation

COMPSET=CESM1_2_1-sole
CASEROOT="/home/liuli/model_simulations/CESM.RCP26"
MACH=Tansuo100
CESM_RES=1.9x2.5_gx1v6
CESM_COMPSET=B_RCP2.6_CN

9.2 Modification of parallel settings and input parameters

The C-Coupler platform does not change the way of modifying the parallel settings and input parameters of CESM. In the working directory of a CESM

simulation, please modify the CESM own script files under the directory "*config/cesm/cesm/cesm_case_scripts/*" to change the parallel settings and private input parameters of CESM. Please still set common parameters of a simulation (such as start and stop time, restart frequency, etc.) in the section *common* in the file "*config/common/case.conf/*". Please configure the simulation after the modification.

9.3 Location of log files of compilation and execution

In the working directory of a CESM simulation, the log files of the compilation are under "*run/cesm/cesm/cesm_bld/*" and the log files of the execution are under "*run/cesm/cesm/data/*".

10 Frequently Asked Questions

1) Cannot find MPI and NETCDF header files when compiling

Solution: Please check the path of compiler (use commands such as “which mpiifort”), and then modify compiling option variables *MPIINC*, *MPILIB*, *NETCDFINC* and *NETCDFLIB* accordingly (please refer Section 6.8 for details).

2) Error happens when linking NETCDF library

Solution: Please check the version of NETCDF to verify the option of linking NETCDF library, such as *-lnetcdf* and *-lnetcdff*, and then modify compiling option variable *NETCDFLIB* accordingly (please refer Section 6.8 for details).

11 Copyright

C-Coupler and the C-Coupler platform are protected by copyrights and patents. They are free and open-source software for non-commercial usage. We encourage users to use them for model coupling, simulation, etc. Models and simulations on the C-Coupler platform will keep their own copyrights. C-Coupler and the C-Coupler platform will not infringe these copyrights.

References

Balaji, V., J. Anderson, I. Held, M. Winton, J. Durachta, S. Malyshev, and R. J. Stouffer, 2006: The Exchange Grid: a mechanism for data exchange between Earth System components on independent grids. In: Proceedings of the 2005 International Conference on Parallel Computational Fluid Dynamics, College Park, MD, USA, Elsevier.

Bellenger, H., Guilyardi, E., Leloup, J., Lengaigne, M., Vialard, J., 2013: ENSO representation in climate models: from CMIP3 to CMIP5. Climate Dyn., DOI 10.1007/s00382-013-1783-z

Canuto, V. M., A. Howard, Y. Cheng, and M. S. Dubovikov, 2001: Ocean Turbulence. Part I: One-Point Closure Model–Momentum and Heat Vertical Diffusivities. J. Phys. Oceanogr., 31, 1413-1426.

Canuto, V. M., A. Howard, Y. Cheng, and M. S. Dubovikov, 2002: Ocean Turbulence. Part II: Vertical Diffusivities of Momentum, Heat, Salt, Mass, and Passive Scalars. J. Phys. Oceanogr., 32, 240-264.

Craig, A. P., R. L. Jacob, B. Kauffman, T. Bettge, J. W. Larson, E. T. Ong, C. H. Q. Ding, Y. He, 2005: CPL6: The New Extensible, High Performance Parallel Coupler for the Community Climate System Model. *International Journal of High Performance Computing Applications*, 19(3): 309-327.

Craig, A. P., M. Vertenstein, and R. Jacob, 2012: A New Flexible Coupler for Earth System Modeling developed for CCSM4 and CESM1. *Int. J. High Perform. C*, 26-1, 31–42, doi:10.1177/1094342011428141.

Dai, Y., et al., 2003: The Common Land Model (CLM), Bull. Am. Meteorol. Soc., 84, 1,013–1,023, doi:10.1175/BAMS-84-8-1013.

Dai, Y., R. E. Dickinson, and Y.-P. Wang, 2004: A two-big-leaf model for can- opy temperature, photosynthesis, and stomatal conductance, J. Clim., 17, 2,281–

2,299, doi:10.1175/1520-0442(2004)017<2281:ATMFCT>2.0.CO;2.

Dai, Y., DY Ji, 2008: The Common Land Model (CoLM) Technical Guide. [at http://globalchange.bnu.edu.cn/download/doc/CoLM/CoLM_Technical_Guide.pdf]

Hill, C., C. DeLuca, V. Balaji, M. Suarez, and A. da Silva, 2004: Architecture of the Earth System Modeling Framework. *Comput. Sci. Eng.*, 6, 18–28.

Hurrell, J. W., M. M. Holland, P. R. Gent, S. Ghan, Jennifer E. Kay, P. J. Kushner, J.-F. Lamarque, W. G. Large, D. Lawrence, K. Lindsay, W. H. Lipscomb, M. C. Long, N. Mahowald, D. R. Marsh, R. B. Neale, P. Rasch, S. Vavrus, M. Vertenstein, D. Bader, W. D. Collins, J. J. Hack, J. Kiehl, and S. Marshall, 2013: The Community Earth System Model: A Framework for Collaborative Research. Bull. Amer. Meteor. Soc., 94, 1339–1360.

Jacob, R., J. Larson, and E. Ong, 2005: M x N Communication and Parallel Interpolation in Community Climate System Model Version 3 Using the Model Coupling Toolkit, Int. J. High. Perform. C, 19, 293–307.

Larson, J., R. Jacob, and E. Ong, 2005: The Model Coupling Toolkit: A New Fortran90 Toolkit for Building Multiphysics Parallel Coupled Models. *Int. J. High Perform*, C, 19, 277–292.

Li L., B. Wang, L. Dong, L. Liu, S. Shen, N. Hu, W. Sun, Y. Wang, W. Huang, X. Shi, Y. Pu, G. Yang, 2013a: Evaluation of Grid-point Atmospheric Model of IAP LASG version 2 (GAMIL2). Advances in Atmospheric Sciences, 30(3), 855-867.

Li, L. J., P. F. Lin, Y. Q. Yu, B. Wang, T. J. Zhou, L. Liu, J. P. Liu, Q. Bao, S. M. Xu, W. Y. Huang, K. Xia, Y. Pu, L. Dong, S. Shen, Y. M. Liu, N. Hu, M. M. Liu, W. Q. Sun, X. J. Shi, W. P. Zheng, B. Wu, M.-R. Song, H. L. Liu, X. H. Zhang, G. X. Wu, W. Xue, X. M. Huang, G. W. Yang, Z. Y. Song, and F. L. Qiao, 2013b: The Flexible Global Ocean-Atmosphere-Land System Model: Grid-point Version 2: FGOALS-g2. Adv. Atmos. Sci., 30(3), 543-560.

Lin, P. F., and Coauthors, 2013: Long-term Stability and Oceanic Mean State Simulated by the Coupled Model FGOALS-s2. Adv. Atmos. Sci., 30(1), 175-192.

Liu, H. L., Zhang, X.H., Li, W., Yu, Y.Q., Yu, R.C., 2004a: A eddy-permitting oceanic general circulation model and its preliminary evaluations, Adv. Atmos. Sci. 21, 675–690.

Liu, H. L., Y. Q. Yu, W. Li, and X. H. Zhang, 2004b: Manual for LASG/IAP Climate System Ocean Model (LICOM1.0). Science Press, Beijing, 1–128. (in Chinese).

Liu, H. L., P. F. Lin, Y. Q. Yu, and X. H. Zhang, 2012: The baseline evaluation of LASG/IAP Climate system Ocean Model (LICOM) version 2.0. Acta Meteorologica Sinica.

Liu, J., 2010: Sensitivity of sea ice and ocean simulations to sea ice salinity in a coupled global climate model, Science in China Series D: Earth Sciences, 53(6), 911-916.

Liu L., R. Li, G. Yang, B. Wang, L. Li, Y. Pu, 2014a: Efficient Parallelization Strategies for the Finite-Difference AGCM on Modern High-Performance Computers. Submited to Journal of Atmospheric and Oceanic Technology, minor revision.

Liu, L., Yang, G., Wang, B., Zhang, C., Li, R., Zhang, Z., Ji, Y., and Wang, L., 2014b: C-Coupler1: a Chinese community coupler for Earth System Modeling, Geosci. Model Dev., 7, 2281-2302, doi:10.5194/gmd-7-2281-2014.

Michalakes, J., J. Dudhia, D. Gill, T. Henderson, J. Klemp, W. Skamarock, and W. Wang, 2004: "The Weather Reseach and Forecast Model: Software Architecture and Performance,". In proceedings of the 11th ECMWF Workshop on the Use of High Performance Computing In Meteorology, 25-29 October 2004, Reading U.K. Ed. George Mozdzynski.

Oleson, K. W., Y. Dai, and Coauthors, 2004: Technical description of the Community

Land Model (CLM), NTIS #PB2004-105836.

Redler, R., S. Valcke, and H. Ritzdorf, 2010: OASIS4 - a coupling software for next generation earth system modeling. *Geosci. Model Dev.*, 3, 87–104, doi:10.5194/gmd-3-87-2010.

Valcke, S., V. Balaji, A. Craig, C. DeLuca, R. Dunlap, R. W. Ford, R. Jacob, J. Larson, R. O'Kuinghttons, G. D. Riley, and M. Vertenstein, 2012: Coupling technologies for Earth System Modelling. *Geosci. Model Dev.*, 5, 1589-1596

Valcke, S., 2013: The OASIS3 coupler: A European climate modelling community software. *Geosci. Model Dev.*, 6, 373–388, doi:10.5194/gmd-6–373-2013

Wang, B., H. Wan, Z. Z. Ji, X. Zhang, R. C. Yu, Y. Q. Yu, and H. L. Liu, 2004: Design of a new dynamical core for global atmospheric models based on some efficient numerical methods. Science in China (A), 47, 4--21.

Wang, G., F. Qiao, and C. Xia, 2010: Parallelization of a coupled wave-circulation model and its application, Ocean Dynamics, 60(2), 331-339.

Wang, X.C., J.P. Liu, Y.Q. Yu, H.L. Liu, and L.J. Li, 2009: Numerical simulation of polar climate with FGOALS− g1.1. Acta Meteorologica Sinica, 67, 961– 972. (in Chinese)

Wu, F., H. Liu, W. Li, and X. Zhang, 2005: Effect of adjusting vertical resolution on the eastern equatorial Pacific cold tongue. Acta. Meteor. Sinica, 24, 1-12.

Xiao, C., 2006: Adoption of a two-step shape-preserving advection scheme in an OGCM and its coupled experiment. M.S. thesis, Institute of Atmospheric Physics, Chinese Academy of Sciences, 89pp. (in Chinese)

Yang, Y., F. Qiao, W. Zhao, Y. Teng and Y. Yuan, 2005: MASNUM ocean wave numerical model in spherical coordinates and its application, Acta Oceanologica Sinica, 2005, 27(2): 1-7.

Yoshimura, H. and S. Yukimoto, 2008: Development of a Simple Coupler (Scup) for

Earth System Modeling. *Meteorology and Geophysics*, 59, 19-29.

Yu, R., 1994: A test for numerical weather prediction of real-time for china flood season precipitation in 1993 by a regional eta-coordinate model. Scientia Atmospherica Sinica, 18, 284-292.

Zhang, X., and X. Liang, 1989: A numerical world ocean general circulation model. Adv. Atmos. Sci., 6, 43-61.