

# Hands on exercises with ORCHIDEE OFF-LINE

Revised for training session 2021-01-12 - 2021-01-13  
Josefine Ghattas/IPSL, Fabienne Maignan/LSCE,  
Bertrand Guenet/LG-ENS

## Contents

1	Before starting	2
1.1	First time at Jean-Zay/IDRIS with your own login	2
1.2	First time at irene/TGCC	2
1.3	First time at obelix/LSCE	2
1.4	First time at ciclad	2
1.5	Short guide to use editor vi	3
2	Install and compile	4
2.1	Install ORCHIDEE trunk for offline use	4
2.2	Optional: Install a branch of ORCHIDEE	5
3	Test simulations	6
3.1	First regional run	6
3.2	Relaunch in the same directory	6
3.3	Continue a simulation using restart files	7
3.4	Visualization with ferret	7
3.5	Change output levels	7
3.6	Add a new output variable in ORCHIDEE	8
4	Compile and run with debug options	9
6	Exercises to learn SVN	11
7	Test in parallel run mode	13
7.1	Running ORCHIDEE with XIOS in attached mode	13
7.2	Check reproducibility of results	13
7.3	Optional: How to rebuild output if running multiple file mode	14
8	Simulations using libIGCM	15
8.1	Some parameters are changed by the comp.driver	15
8.2	SPINUP ANALYTIC experiment	16
8.3	OOL SEC STO experiment	17
A	Appendix: More details	18
A.1	Description of xml files	18
A.2	Example of parallel job files	19
A.3	Copy/link input files for simple test case according to the wiki	21

## 1 Before starting

The goal of these exercises is to learn how to install, compile and launch basic test cases with ORCHIDEE in off-line mode. All exercises can be done at Jean-Zay(IDRIS), irene(TGCC), obelix(LSCE) or ciclad(IPSL ESPRI-mesocenter). All commands needed for the basic exercises are listed in the text.

During the training session, you'll install the model on the \$SCRATCH. When you work on your proper login and your proper version, it is most of the time better to install the model in the work directory. Remember that the scratch directory at most machines is temporary and can be cleaned automatically after 30days (depending on the machine).

### 1.1 First time at Jean-Zay/IDRIS with your own login

During the online training session you will work on your own account. When working on jean-zay using your own login you need to install the IPSL-cmc environnement the first time, do as follow:

```
cd $HOME
cp /gpfswork/rech/psl/commun/MachineEnvironment/jeanzay/bash_login .bash_login
rm .bash_profile
vi .bash_login
```

Disconnect and reconnect, the .bash login will now be sourced by default each time you connect to jean-zay. Modules needed to compile and run the model are loaded via this file. libIGCM will load the same modules by sourcing the file /gpfswork/rech/psl/commun/MachineEnvironment/jeanzay/env jeanzay which is also sourced from .bash-login.

### 1.2 First time at irene/TGCC

During the online training session you will work on your own account. If you'll start working on irene, then install the IPSL-cmc environnement before the first installation of the model. Do as follow:

```
cp /ccc/cont003/home/igcmg/igcmg/MachineEnvironment/irene/bashrc $HOME/.bashrc
cp /ccc/cont003/home/igcmg/igcmg/MachineEnvironment/irene/bashrc_irene $HOME/.bashrc_irene
```

Disconnect and reconnect, the .bashrc irene will now be sourced by default. Modules needed to compile and run the model are loaded by this file. The modules are specified in the file /ccc/cont003/home/igcmg/igcmg/MachineEnvironment/irene/env irene. When running the model, the same modules need to be loaded. This is done automatically by libIGCM. When you work without libIGCM, you must source the file env irene from your job, see job example in the ap- pendix.

### 1.3 First time at obelix/LSCE

During the online training session you will work on your own account. At obelix, you need to load the following netcdf module before compiling (you can add it to your login environnement, for example in .cshrc depending on the shell used):

```
module load netcdf/4p
```

If you run without libIGCM you need to load the same module in your run script or in the terminal if running interactively. When running with libIGCM the following file, which contains the above module, is sourced:

```
/home/orchideeshare/igcmg/MachineEnvironment/obelix/env_atlas_obelix
```

You can work in a folder in /home/scratch01/yourlogin for these exercises. Note that /home/scratch01 might be cleaned after 30 days.

### 1.4 First time at ciclad

During the online training session you will work on your own account. At Ciclad, add

following in your `.bashrc` file (create the file if it doesn't exist):

```
# User specific aliases and functions
. /home/igcmg/.atlas_env_ciclad_ksh
```

Disconnect and reconnect, the `.bashrc` will now be sourced by default. Modules needed to compile and run the model are loaded by this file. When running with libIGCM, the same file

`.atlas_env_ciclad_ksh` will be sourced. When you work without libIGCM, you must source this file from your job, see job example in the appendix.

Install the model in your space `/data/login`.

## 1.5 Short guide to use editor vi

In these exercises when it says “**vi filename**” it means open the file with an editor of your choice. You can use for example **vi** or **emacs**. **vi** is a text editor program which can be used in a terminal window to open and edit ascii files. Here are some very basic commands to use vi:

```
vi filename      # open the file
/toto            # search for toto in the file. Use n for next or
                # ? for previous occurrence of the word.
i               # open insert mode. You can now edit the file
escap           # close insert mode
:w              # save the file
:q              # close the file
:q!             # close the file without saving anything you did since last :w
:wq            # save and close
:syntax off     # to deactivate colors if needed
nG             # go to line n in the file
```

**emacs** is another text editor. It can be opened in a separate window and it has a menu board which might be easier to use. Open as follow:

```
emacs filename &
```

## 2 Install and compile

### 2.1 Install ORCHIDEE trunk for offline use

We are first going to download **modipsl** and explore what is inside. **modipsl** contains some tools in the directory **util**. In **util**, scripts are found for extraction (*model*, *mod.def*) and creation of makefiles (*ins make*, *AA make.gdef*). **modipsl** is also an empty file tree that will receive the models and tools when downloading a configuration.

Start this exercise by extracting **modipsl** in a new directory:

```
cd $SCRATCH
mkdir TESTOFFLINE; cd TESTOFFLINE
mkdir modipsl
svn co modipsl cd modipsl/util
ls
```

The script **model** is used to download a specific predefined configuration with the model sources and tools needed. The script uses the file **mod.def** that contains specifications for each configuration predefined. Use **./model -h** to see all existing configurations and **./model -h config name** for information about a specific configuration. Same information can be found in the file **mod.def**.

For these exercises you will use the configuration ORCHIDEE trunk which is an offline set up using the latest revision of the trunk ORCHIDEE. Open **mod.def** file and look at lines beginning with ORCHIDEE trunk, close the file without any changes and then extract and compile ORCHIDEE as follow (please note that the compilation script changed recently and if you work on another version you may need to do it differently. All information are on the wiki):

```
./model ORCHIDEE_trunk
cd ../config/ORCHIDEE_OL
./compile_orchidee_ol.sh
```

While the compilation is on going, open a new terminal, connect to **jean-zay**(**irene**, **obelix** or **ciclad**) as in the beginning and continue the exercises (no need to copy the bash login file again).

The makefile was created automatically in the end of the script **model**, done by the script **ins make**. **ins make** will detect on which machine you are working and create adapted make- files. By default **ins make** recognizes the following machines: **irene** at TGCC, **jean-zay** at IDRIS, **obelix** at LSCE and **ciclad** and **climserv** at ESPRI mesocenter at IPSL. **ins make** can also be re-launched manually. For example, this is needed if you move the **modipsl** directory or if you want to create makefiles for another target machine. The main makefile is found in the **modipsl/config/ORCHIDEE\_OL** directory.

When the compilation is finished you will find the executables **orchidee\_ol** and **xios server.exe** in **modipsl/bin**.

Now explore the directories in **modipsl**. You will find all source code for ORCHIDEE in directory **modipsl/modeles**. You also find the directory **IOIPSL** and **XIOS** which are fortran and C libraries linked to ORCHIDEE for input and output issues. In directory **modipsl/config/ORCHIDEE\_OL** you find scripts to run ORCHIDEE using **libIGCM**. **libIGCM** is a tool developed at IPSL to run coupled and off-line simulations. Specific training sessions about **libIGCM** are given by the **Plateforme groupe** at IPSL.

Go into each directory (using **cd** and **ls**) and check which versions have been extracted. You're supposed to find the same information as you can see in **mod.def**. Use **svn** to know the version and the revision number.

```
cd ../modeles/ORCHIDEE
svn info
cd ../IOIPSL/src
svn info
cd ../../../../libIGCM
svn info
```

If the compilation is still ongoing, you can take the waiting time to do **Exercises to learn SVN**.

## 2.2 Optional: Install a branch of ORCHIDEE

Do not this exercise during the training session. Do as follow if you would like to install a specific branch instead of the trunk of ORCHIDEE. Note you can only do this for branches where you have read acces. Install first modipsl as before. Then open the file util/mod.def and look for the section ORCHIDEE trunk. Change the line containing ORCHIDEE/trunk into the name for the branch you will use.

For exemple, to extract ORCHIDEE-MICT, do the following:

```
cd $SCRATCH; mkdir TESTMICT ; cd TESTMICT ; mkdir modipsl
svn co http://forge.ipsl.jussieu.fr/igcmg/svn/modipsl/trunk modipsl
cd modipsl/util
```

In mod.def, change the line

```
#-C- ORCHIDEE_trunk trunk/ORCHIDEE HEAD 14 ORCHIDEE modeles
```

into

```
#-C- ORCHIDEE_trunk branches/ORCHIDEE-MICT/ORCHIDEE HEAD 14 ORCHIDEE modeles
```

Finally extract as before:

```
./model ORCHIDEE_trunk
```

Note that *branches/ORCHIDEE-MICT/ORCHIDEE* can be changed to another path on the svn repository, for example *branches/ORCHIDEE-CN-CAN/ORCHIDEE* or *branches/ORCHIDEE-SOM/ORCHIDEE*. It can also be the path to a personal version.

You can also set a specific revision number by changing HEAD into a revision number.

### 3 Test simulations

We will now do some test simulations using the ORCHIDEE trunk offline installation. This will first be done interactively which means to launch directly in the terminal without passing through the batch system. In other words, launch directly “./orchidee\_ol”. libIGCM is not used here because we want you first to understand how the model works and what is needed as input files. You can also run using a job and launch on the batch system, this will be done later in the exercises for parallel computing. The main advantage with a job is that you reserve the computing resources in advance and that you can run in parallel.

#### 3.1 First regional run

Set up a simple test case to run in sequential mode without libIGCM. First create a new directory outside modipsl. You already compiled the executable so it can simply be linked in the new folder.

```
cd TESTOFFLINE; mkdir RUN1; cd RUN1
ln -s ../modipsl/bin/orchidee_ol .
```

Follow the ORCHIDEE wiki documentation to copy input netcdf files, copy and modify xml files and create the run.def parameter file:

**<http://forge.ipsl.jussieu.fr/orchidee/wiki/Documentation/UserGuide/TestCase1>**

In complement to the wiki documentation, the input files are found on the following shared repositories:

```
# At Jean-Zay: R_IN=/gpfswork/rech/psl/commun/IGCM
# At Irene:   R_IN=/ccc/work/cont003/igcmg/igcmg/IGCM
# At Obelix:  R_IN=/home/orchideeshare/igcmg/IGCM
# At Ciclad/ClimServ: R_IN=/prodigfs/ipslfs/igcmg/IGCM
```

You can use for example year 2000 for the input files. If you feel that it takes too long time to type and copy each file, you can find in the appendix the lines ready to be copied.

When you have set up the run directory according to the wiki, you can use ncdump to see what is in the netcdf files. For example:

```
ncdump -h forcing_file.nc
```

Now launch the model:

```
./orchidee_ol # or ./orchidee_ol > out_exec
```

When the execution is completed correctly, following log message is found in the output text file out\_orchidee 0000:

```
END of dim2_driver
```

#### 3.2 Relaunch in the same directory

If you want to relaunch the model in the same directory, then you need to delete restart and output files previously created by the model. Do this now and re-run the model:

```
rm driver_rest_out.nc sechiba_rest_out.nc stomate_rest_out.nc
rm sechiba_history.nc sechiba_out_2.nc sechiba_history_4dim.nc
rm stomate_history.nc stomate_ipcc_history.nc
rm out_*
./orchidee_ol
```

### 3.3 Continue a simulation using restart files

The model writes restart files at the end of each execution period, after a time set by `TIME LENGTH` in `run.def` parameter file. The restart files contain all state variables needed to continue a simulation without losing information. To continue the simulation, you need to rename the restart files produced in previous run and activate reading of these files in `run.def` with parameters `SECHIBA restart in`, `STOMATE RESTART FILEIN` and `RESTART FILEIN`. Save the output files `sechiba history.nc` and `stomate history.nc` for later analyses.

Add in `run.def`:

```
SECHIBA_restart_in=sechiba_rest_in.nc
STOMATE_RESTART_FILEIN=stomate_rest_in.nc
RESTART_FILEIN=driver_rest_in.nc
```

Rename restart files:

```
mv driver_rest_out.nc driver_rest_in.nc
mv sechiba_rest_out.nc sechiba_rest_in.nc
mv stomate_rest_out.nc stomate_rest_in.nc
```

Save output:

```
mv sechiba_history.nc sechiba_history_month01.nc
mv stomate_history.nc stomate_history_month01.nc
```

Relaunch the model:

```
./orchidee_ol > out_exec
```

Note that for longer simulations **libIGCM** is used to chain the executions without manual copy or move of files. **libIGCM** is a powerful tool but it is important to know how the model works.

### 3.4 Visualization with ferret

Here is a small example to visualize `sechiba history.nc` using `ferret`.

```
> module load ferret
> ferret
use sechiba_history.nc # read file
sh d                   # list content in file
shade CONTFRAC         # 2D plot of a variable
go land                # add contour of continents
shade TEMP_SOL[l=1]    # 2D plot of TEMP_SOL for first time step
shade TEMP_SOL[l=@ave] # 2D plot of TEMP_SOL average over all time steps
shade SWDOWN[i=@ave]   # zonal plot
plot SWDOWN[i=@ave,j=@ave] # plot mean value over time
quit
```

### 3.5 Change output levels

Do different small runs where you change output frequency, number of variables in the files, activate or deactivate files, change name of the variables in the output file, etc. Do these kinds of changes in file `def orchidee.xml`. Read more about xml files in the appendix.

### 3.6 Add a new output variable in ORCHIDEE

Create a diagnostic output variable for the variable `resp_hetero_litter` calculated in `stomate_litter.f90` using XIOS. To do this, add in `src/stomate/stomate_litter.f90`:

```
! Load XIOS in the beginning of the module
use xios_orchidee
...
! Add send to XIOS in the end of the subroutine littercalc
CALL xios_orchidee_send_field("resp_hetero_litter", resp_hetero_litter)
```

Compile in `modipsl/config/ORCHIDEE_OL`:

```
cd config/ORCHIDEE_OL
./compile_orchidee_ol.sh
```

Create a new run directory as before (or copy the previous one and remove everything created during run time). The variable has dimension `DIMENSION(npts,nvm)` in the subroutine. The corresponding grid to be used in the xml files is `grid_ref="grid_nvm"` in `field_def_orchidee.xml` and `grid_ref="grid_nvm_out"` in `file_def_orchidee.xml`. Add in `field_def_orchidee.xml` the definition of the new variable:

```
<fieldid="resp_hetero_litter" name="RESP_H_LITT" unit="gC/m^2/s" grid_ref="grid_nvm"/>
```

Add in `file_def_orchidee.xml` in the section for the file where you want to add the variable:

```
<field field_ref="resp_hetero_litter" grid_ref="grid_nvm_out" level="1"/>
```

Use `svn diff` in `modeles/ORCHIDEE` to see your modifications in the code. Launch as before. Verify that the variable is in the output file.

It is recommended to modify the `field_def_orchidee.xml` in the model directory `ORCHIDEE/src/xml` as well as the file used in the run directory.

Launch the model and use `ferret` to visualize the new output variable.



## 4 Compile and run with debug options

In this exercise you'll work on a revision on the model which contains an error. This revision of the model can run without any direct errors when compiled in default production mode. But when compiling with additional debug options, the model will crash due to errors in the code. The errors are always there even when using the default compilation but they are less visible and the model continues running instead of crashing.

Install a new modipsl, extract the ORCHIDEE trunk configuration and update to use the revision 6445. Do as follow (see also the exercises on SVN):

```
cd $SCRATCH; mkdir TESTDEBUG; cd TESTDEBUG; mkdir modipsl
svn co http://forge.ipsl.jussieu.fr/igcmg/svn/modipsl/trunk modipsl
cd modipsl/util/
./model ORCHIDEE_trunk
cd ../modeles/ORCHIDEE
svn info # => check that you have the trunk version
svn update -r 6445
```

Now modify the Makefile to compile ORCHIDEE and XIOS in mode debug instead of mode prod. To do so, change all locations of -prod into -debug in the Makefile:

```
cd ../../config/ORCHIDEE_OL
./compile_orchidee_ol.sh -debug
```

Continue with the other exercises will waiting for the compilation to finish.

Create a run directory as in the first exercises and launch it. You can also launch using libIGCM and the experience\_OOL\_SEC STO FG2. What happens ?

Answer following questions:

1. What is the difference in the text output?
2. Do you expect the same run time? Note that in this case, the model crashes so you can not measure the run time.
3. Give an example of a warning message. What does it say?
4. Find the error message. In which file and on which line does the model crash? You can find answers and explanations in the appendix B.

## 5 Read parameters from run.def file using getin

### Description of some parameters in run.def

The file run.def contains parameters to run the model. A line beginning with a # is a comment. Default values for each parameter are set in the ORCHIDEE model fortran code and they will be used for all parameters not set in run.def. You can find the list of all parameters and their default values in modipsl/modeles/ORCHIDEE/orchidee.default. The variables in run.def are read from the model using a call to getin\_p which is an interface taking different types of variables. See here some example of parameters:

- **TIME\_LENGTH** gives the simulation length for each execution. In this test case **TIME\_LENGTH** is 31 days. It is possible to run one year by putting **TIME\_LENGTH=1Y**. It is not possible to run less than one day. In global or regional simulations, we do not advice to run more than 1 year per execution but for site simulations using FLUXNET forcing it is recommended to run the full forcing file length in one execution.
- **LIMIT EAST**, **LIMIT WEST**, **LIMIT NORTH** and **LIMIT SOUTH** are borders (in degrees) for the horizontal domain to be modeled. The default values correspond to the domain of the forcing file. The model will stop if the domain does not cover any land points with error message:

```
FATAL ERROR FROM ROUTINE dim2_driver
--> number of land points error.
--> is zero !
--> stop driver
```

- **RIVER\_ROUTING** parameter activates the river routing, default value is **TRUE**.
- **NVM** number of PFTs used in the model. Default value is 15. This number must correspond to the number of PFTs in the vegetation map (PFTmap.nc). For example, the default set up for ORCHIDEE 1.9.5 which is used for the CMIP5 simulations uses **NVM=13**.
- **VEGET\_UPDATE** frequency for updating the vegetation map. By default, **VEGET\_UPDATE=0Y** which means that the vegetation map will not be updated. Using **VEGET\_UPDATE=1Y**, the map will be updated each 1st of January. The PFTmap.nc for the current year must be available.

### Exercises

- Search in the model to find where the default value for the parameter **RIVER\_ROUTING** is set. In which fortran file is the parameter read?
- Open ORCHIDEE/src sechiba/hydrol.f90 and see how the variable **froz\_frac\_corr** is modified. Which is the default value and how can you change it without recompiling?
- Which parameter should be used to change default value to prescribe the atmospheric **CO2 concentration** and which is the default value? Does the default value set in orchidee.default correspond to the value set in the code?
- In thermosoil.f90, the variables **THKICE**, **THKQTZ** and **THKW** are set in the code and cannot be changed from run.def. Make this possible! The default values should be the same as before. Add a write command to see the values used during run time. Change the code, recompile and run with default and modified values. Make sure that the modified values are taken into account. For this exercise you can choose to run using a simple run directory as in the beginning of the exercises or using a libGCM experiment such as OOL SEC STO FG2.

## 6 Exercises to learn SVN

This exercise aims to learn the basic use of svn. The setup for the exercise you must download a previous version. Type the commands listed below and answer questions.

### Determine version and revision

1. Which version and which revision of ORCHIDEE is installed in directory ORCHIDEE.2?

```
cd $SCRATCH; mkdir TESTORCHIDEE_2_2; cd TEST ORCHIDEE_2_2; mkdir modipsl
svn co http://forge.ipsl.jussieu.fr/igcmg/svn/modipsl/trunk/modipsl
cd modipsl/util/

./model ORCHIDEE_2_2

svn info
```

2. Are there any local modifications done after download? Using `svn stat`, the local so called working version is compared to the version and revision that was extracted (see information in `svn info`).

```
svn stat
svn diff
```

3. Are there any modifications done on the server more recent than your version? What is the difference when you add `-u`? Use `svn help` to know more about the commands `svn`, for example `svn help stat`.

```
svn -u stat
```

4. Answer the same question for the version ORCHIDEE.3: Which version and which revision is installed here? Are there any local modifications? Modifications done later on the server?

```
cd $SCRATCH; mkdir TESTORCHIDEE_3; cd TEST ORCHIDEE_3; mkdir modipsl
svn co http://forge.ipsl.jussieu.fr/igcmg/svn/modipsl/trunk/modipsl
cd modipsl/util/

./model ORCHIDEE_3_head
```

Note that the use of `svn` is based on `.svn` folders stored in each directory and each sub-directory. If these directories are not there, it is not possible to get information about the version used. Therefore, when you copy a local version of ORCHIDEE, always copie all sub-folder using `cp -r`.

### Clean, update and resolve conflicts

5. Now copy the version ORCHIDEE.3 into your workdir. Is this version modified? Is it up to date or has it been changed on the server?

```
cd $SCRATCH
cp -r yourpath/ORCHIDEE_3 myORCH.3
cd myORCH.3
```

6. You will now clean this version before updating it. Remove modifications, which are not needed by using `revert` and check which files have been updated on the server. In this exercises, do not revert `stomate.f90` and `stomate_vmax.f90` because there modified versions will be used in next exercises.

```
svn stat; svn diff
svn revert file_where_modifications_are_not_needed_to_be_saved
svn -u stat
```

7. You will now update the full ORCHIDEE to the latest revision on the current version (path on svn). Always do this command from the base directory ORCHIDEE. Use `svn -u stat`. All

files marked with \* (star) will be updated. Files with \* and M, might make conflicts.

```
svn update      # Answer p if conflict is detected
svn stat
```

**8.** If a file is marked C, it means conflict. You have to resolve the conflict manually. Open the file and try to resolve the conflict. When this is done, tell svn that the conflict is resolved by using svn resolved.

```
vi file_with_conflict      # look for sections with <<<<<< and correct them
svn diff
svn resolved file_with_conflict  # tell svn conflict is resolved
svn diff file_with_conflict
```

**9.** What happen to the file stomata\_vmax.f90? Where there any conflicts detected? Open the file and check the lines with the local modifications in your working version. Are the modifications still accurate?

## 7 Test in parallel run mode

The default parallelization mode for ORCHIDEE offline configuration is MPI. The model can then be run on 1 or several MPI cores. Coupled to LMDZ, the default parallelization mode is hybrid mode with mixed MPI and OpenMP.

Output diagnostic files are written using XIOS by default. It is also possible to use IOIPSL but this method is no longer fully maintained. XIOS makes it possible to run either in attached mode or in server mode. When running in attached mode, only the executable `orchidee_ol` is used. XIOS is used as a library in the model. When running in server mode, the executable `xios server.exe` is launched together with the executable `orchidee_ol`. This is a more complex way to launch the model but it is more efficient when simulating on a big region or with high resolution.

### 7.1 Running ORCHIDEE with XIOS in attached mode

You will now launch a global test run using 20 cores (only use 8 at obelix) in attached mode. When running in attached mode it is possible to set XIOS to write one global output file (mode one file) or to write partial output files, one per local MPI domain (mode multiple file). When running in multiple file mode, the output files need to be post-processed to contain the full domain. This is done using the tool **rebuild**. We will here use the one file mode. Running in attached mode is recommended only for debugging or development purpose or when running on 1 pixel or a small domain. For longer simulations, the server mode is used.

Prepare a new run directory as in the first exercise but do not put any regional limits in `run.def` (remove the `LIMIT` parameters). Open the xml files and see that one file mode is activated in file `def_orchidee.xml`. It is not possible to use compression of output when running in attached one file mode. Therefore modify to `compression_level=0` in the same file.

```
# in file def_orchidee.xml
<file_definition type="one_file" ... compression_level="0">
```

In earlier version of XIOS it was necessary to specify in `iodef.xml` if running in attached or in server mode. This is not longer needed as XIOS detects it according to how the model is launched.

Write a file `Job_orchidee` as in the appendix to launch the model in the batch system instead of interactively. The job file will be different for different machines, choose the job for `jean-zay` as in the appendix.

### 7.2 Check reproducibility of results

Changing the number of cores used to run the model should not change the final results. Create a new directory and make the same test but on half of the processors. Check that restart files and output files are the same. You can use `cdo` to check that netcdf files are identical.

```
> cdo -diffv file1.nc file2.nc
```

Check also the difference in time due to the change of number of processors. In the ideal case, if ORCHIDEE would be perfectly scalable the job should run 4 times faster on 4 processes than on

1. This is not the case but at least you should notice a significant gain in time while increasing the number of processes.

Note that in the some version of the code, when running on 1 core compared to several cores, the model doesn't give exactly the same results whereas running using 2 or more cores give the same results. In tag ORCHIDEE 2.0 the model always give the same results.

### 7.3 Optional: How to rebuild output if running multiple file mode

Parallel jobs will write to several text files, one per process, out orchidee 0000, out orchidee 0001... and the output netcdf files will be written in local domain sechiba history 000.nc, sechiba history 0001.nc etc. A reconstruction of the output netcdf files to the total domain is necessary. This is done with

the *rebuild* tool developed at IPSL as an extension of IOIPSL.

rebuild is installed at the different machines here:

```
# at Jean-Zay:
/gpfswork/rech/psl/commun/Tools/rebuild/modips1_IOIPSL_PLUS_v2_2_4/bin
# at Irene:
/ccc/cont003/home/igcmg/igcmg/Tools/irene/rebuild/modips1_IOIPSL_PLUS_v2_2_4/bin
# at Obelix:
/home/orchideeshare/igcmg/IOIPSL_PLUS/modips1.tagv2_2_4.28082019/bin
# at Ciclad:
/home/igcmg/rebuild/src_X64_CICLAD/modips1_v2_2_3_netcdf4.2/bin
# at ClimServ (same as Ciclad but the path is different):
/ciclad-home/igcmg/rebuild/src_X64_CICLAD/modips1_v2_2_3_netcdf4.2/bin
```

If you've installed the IPSL environment (.bash login at Jean-Zay) you already have rebuild in your path. Try typing *which rebuild* to see the version of rebuild in use.

Do the rebuild as follow:

```
rebuild -o sechiba_history.nc sechiba_history_00*
```

## 8 Simulations using libIGCM

The following exercises will use the first installation you did of ORCHIDEE trunk configuration with libIGCM, in directory TESTOFFLINE.

There are some differences between ORCHIDEE trunk configuration and the coupled configurations such as LMDZOR v6. In the configuration ORCHIDEE trunk it is not needed to create the submit directory. Instead different predefined experiment directories already exist. They can be copied and used directly. These directories are OOL SEC STO FG\*, OOL SEC and SPINUP ANALYTIC FG1. They follow the standard rules described in the training and documentation for libIGCM. The DRIVER directory does not exist but the “comp.driver” files are found in the COMP directory. The directory FORCESOIL also follows the same structure but this experiments is currently under development. SPINUP and ENSEMBLE directories contain experiences that are more complex and are not taught in the course.

We will here work with the OOL SEC STO FG2 and SPINUP ANALYTIC FG1 experiments.

### Specific system option during the training course

To help the post-treatment job to start run more easily, you can set the time limit in the beginning of the job to a lower value for short exercises like we do here. Change in libIGCM/AA create ts in the section jeanzay to have:

```
#-Q- jeanzay #SBATCH --time=1:00:00
```

If the create\_ts.job already exists, then remove it. It will be recreated when you run ins job.

### 8.1 Some parameters are changed by the comp.driver

According to the choices made in config.card and the comp.card (comp stands for the components: orchidee ol.card, sechiba.card, stomate.card), some parameters will be changed in the run.def and in the file def orchidee.xml. The modifications are done by the comp.driver (orchidee ol.driver, sechiba.driver, stomate.driver). The parameters who might be modified are always marked as AUTO or AUTOBLOCKER

- **AUTO:** These parameters can be changed using options in comp.card or config.card. You can also change them directly in the PARAM/run.def or modeles/ORCHIDEE/src xml/file def orchidee.xml, in that case the drivers will not change them again.
- **AUTOBLOCKER:** The job will stop if you modify these parameters. They are set by the comp.driver using the information from config.card.

For example, in PARAM/run.def:

```
STOMATE_RESTART_FILEIN = _AUTOBLOCKER_  
VEGET_UPDATE = _AUTO_
```

### Exercise

Go into OOL SEC STO FG2 experiment directory, open PARAM/run.def and search for variables marked AUTO and AUTOBLOCKER. Try to find out which are the variables and how they can be changed from the config.card or comp.card using the appropriate options.

## 8.2 SPINUP ANALYTIC experiment

You'll now set up a spinup simulation using libIGCM. Start by coping the SPINUP\_ANALYTIC\_FG1 directory:

```
cd modips1/config/ORCHIDEE_OL
cp -r SPINUP_ANALYTIC_FG1 MyTestSpinup
cd MyTestSpinup
```

Look into the config.card. The variables CyclicBegin and CyclicEnd describe the years to loop over. Setting these 2 variables in config.card makes the variable CyclicYear available. CyclicYear is used in the orchidee.ol.card to copy the forcing file. For this exercise loop over years 1901-1910. Check CyclicBegin and CyclicEnd and modify if necessary. Limit the region to a grid-cell. Set in PARAM/run.def:

```
LIMIT_WEST = -60.
LIMIT_EAST = -58.
LIMIT_NORTH = -8.
LIMIT_SOUTH = -10.
```

Set up the simulation to run over 4 forcing periods (40 years in total). In config.card, set DateEnd=1940-12-31. Set SpaceName=TEST to deactivate pack post treatment if running at Jean-Zay or Irene. Set TimeSeriesFrequency=5Y and SeasonalFrequency=NONE. It is important to adjust the number of cores used (number of MPI and OMP) to the domain which is used. Here we run on 1 grid-cell and therefore set 1 MPI on the line for the executable to run in sequential mode. You also need to deactivate XIOS server by setting IOS=(",") in the Executable section.

Create the job using ins job in libIGCM. Increase PeriodNb before launching the job. Launch the job and answer following questions:

1. Why should you deactivate XIOS in server mode in this example?
2. How do you calculate PeriodNb?
3. Which forcing file is used and where is it stored? Where is the shared repository IGCM?
4. Where is the output stored? How can you change the place for the ARCHIVE directory? Note that this is recommended only at obelix.
5. In orchidee.ol.card you can use the variable **year** or **CyclicYear**. Which are the differences?
6. The variable SPINUP PERIOD is calculated by the stomate.driver and set in run.def. Where can you see the run.def file that was used during simulation? What is the SPINUP PERIOD?

When the time-series have been done, you can have a look at the evolution of the carbon pools. Go to the output directory

```
cd .../IGCM_OUT/.. /JobName/SBG/Analyse/TS_YE
ferret
use JobName_19010101_19401231_1Y_SOIL_ACTIVE.nc
use JobName_19010101_19401231_1Y_SOIL_SLOW.nc
use JobName_19010101_19401231_1Y_SOIL_PASSIVE.nc

set v ul; plot SOIL_ACTIVE[k=@ave,i=@ave,j=@ave,d=1]
set v ur; plot SOIL_SLOW[k=@ave,i=@ave,j=@ave,d=2]
set v ll; plot SOIL_PASSIVE[k=@ave,i=@ave,j=@ave,d=3]
```



### 8.3 OOL SEC STO experiment

Currently 4 different OOL SEC STO experiments exist. OOL SEC STO FG2 and FG3 are both historical set up with land use change activated. The difference are the forcing file and the resolution: for FG2 CRU-NCEP is used and for FG3 WFDEI is used. OOL SEC STO FG3nd is the same as FG3 but it uses the driver *orchideedriver* which is still under development. FG1trans is set up for a transient simulation between the SPINUP ANALYCTIC FG1 and historical OOL SEC STO FG2.

Set up an experiment with sechiba and stomate using the experiment directory OOL SEC STO FG2.

- Set up the simulation by period of 1 month for a total simulation length of 3 months.
- Activate the option for floodplains in run.def. Look in the file orchidee.default in the source directory to have the exact name for the parameter activating this option.
- Add the input file floodplains.nc if needed. This file is only needed for the first period. The information will be stored in the restart file. Search in the shared repository IGCM (directory R IN) to find the input file. In which section in sechiba.card should the file be added and why?
- Output the variable floodplains in 2 files by frequency 1 month and 1 day. Look into sechiba.card and read comments to know how to set output frequencies. What is the file id used in the xml files for the variable floodplains?

Prepare the rest of the job as usual. Launch the test and analyze the results.

## A Appendix: More details

### A.1 Description of xml files

The xml files are used to configure the output files when using XIOS. The xml files are stored in ORCHIDEE/src xml directory. When running the model using libIGCM, the file def orchidee.xml is changed for all occurrences of the keyword AUTO . The following 5 files are needed for ORCHIDEE:

- iodef.xml: this file is the first file read by XIOS.
- context orchidee.def: containing axis and grid information
- context input orchidee.def: containing information about input files and related grids. This file is mandatory in the trunk since revision 5565 (since 09/11/2018) but not yet present in all branches.
- field def orchidee.xml: contains one line per output variable sent from the model. This file is only changed if new output are added in the model. A variable is output from the model with a call to subroutine xios orchidee send field.
- file def orchidee.xml: contains specifications about the output files and contents. This is the file to be changed for all modifications in the output settings. This file is modified by orchidee ol.driver when running with libIGCM. It is only modified where the keyword AUTO is set. You can change the AUTO as you wish and make other changes according to your-needs, they will never be overwritten. When running without libIGCM you must change all AUTO , read comments in the begining of the file.

## A.2 Example of parallel job files

### A.2.1 Job using XIOS in attached mode at Jean-Zay/IDRIS

Follow the instruction in *Example of a job to start an executable in a Parallel environment* for MPI in the on-line documentation in chapter Computing Centers:

[http://forge.ipsl.jussieu.fr/igcmg\\_doc/wiki/Doc/ComputingCenters/IDRIS/JeanZay](http://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/ComputingCenters/IDRIS/JeanZay)

Don't forget to modify the project id. For ORCHIDEE offline global run, modify to ntask=20 in the header.

### A.2.2 Job using XIOS in attached mode at obelix/LSCE

This is a job file example for running orchidee with XIOS in attached mode at obelix. Create Job orchidee with following lines:

```
#PBS -N test
#PBS -m a
#PBS -j oe
#PBS -o Script_Output
#PBS -S /bin/ksh
#PBS -q shortp
#PBS -l nodes=1:ppn=8
#
# Source module
. /home/orchideeshare/igcmg/MachineEnvironment/obelix/env_atlas_obelix
# Go to current folder and execute
cd $PBS_O_WORKDIR
time mpirun ./orchidee_ol
# End of file
```

Submit the job to the queue with the command qsub. Check with the qstat and use the command qcat to see the progress of the job. Do following

```
> qsub Job_orchidee      # submit the job
> qstat -u login        # check the job's running status
> qcat job_id | more    # check the progress of the job. job_id is given by qstat
```

### A.2.3 Job using XIOS in attached mode at ciclad/IPSL

This is a job file example for running orchidee with XIOS in attached mode at ciclad. Create Job orchidee with following lines:

```
#####
## CICLAD IPSL ##
#####
#PBS -N test
#PBS -m a
#PBS -j oe
###PBS -q h12 # Queue for 12 hours at ciclad only
#PBS -o Script_Output
#PBS -S /bin/ksh
#PBS -v BATCH_NUM_PROC_TOT=32
#PBS -l nodes=1:ppn=32
#PBS -l mem=12gb
#PBS -l vmem=40gb
#
```

```

# Access to module command
. /usr/share/Modules/init/ksh
# Source module
. /home/igcmg/.atlas_env_ciclad_ksh
# Go to current folder and execute
cd $PBS_O_WORKDIR
time mpirun ./orchidee_ol
# End of file

```

Submit the job to the queue with the command `qsub`. Check with the `qstat` and use the command `qcat` to see the progress of the job. Do following

```

> qsub Job_orchidee      # submit the job
> qstat -u login        # check the job's running status
> qcat job_id | more    # check the progress of the job. job_id is given by qstat

```

Note that at ciclad, by default compilation is done with netcdf sequential library, using the argument `-netcdf lib netcdf4 seq` while compiling XIOS (see Makefile). This means that when you run in attached mode, the one file mode will not work. XIOS will switch automatically to multiple file mode and you'll see that the output files will be splitted in several files. Rebuild is needed to see the full domain after the run. But when running in server mode using 1 server, the server will output on the full domain no rebuild is needed. Use libGCM default mode with 1 server for further simulations.

#### A.2.4 Job using XIOS in attached mode at irene/TGCC

This is a job file example for running orchidee with XIOS in attached mode at irene/TGCC. Create Job\_orchidee with following lines:

```

#!/bin/ksh
#MSUB -r test           # name of the job
#MSUB -o Script_Output # name of output file for standard messages
#MSUB -e Script_Output # name of output file for error messages
#MSUB -eo
#MSUB -n 32            # Request nombre of cores
#MSUB -T 1800          # Time limit in seconds
#MSUB -q skylake
#MSUB -m store,work,scratch
#MSUB -A genXXX        # Set your project id. Most people working with
#                      # ORCHIDEE at LSCE belong to project gen6328
# Source modules
. /ccc/cont003/home/igcmg/igcmg/MachineEnvironment/irene/env_irene
# Go to current folder and execute the model
cd ${BRIDGE_MSUB_PWD}
/usr/bin/time ccc_mprun -n 32 ./orchidee_ol
# End of file

```

Submit the job to the queue with the command `ccc msub`. Check with the `ccc mstat`. Do following

```

> ccc_msub Job_orchidee      # submit the job
> ccc_mstat -u login        # check the job's running status

```

### A.3 Copy/link input files for simple test case according to the wiki

This test case is adapted for ORCHIDEE trunk revisions from 5639 and later. The path R\_IN corresponds to Jean-Zay. Change this path for other platforms. If your terminal shell is csh or tesh, use set instead of export (this might be the case at obelix).

```
export R_IN=/gpfswork/rech/psl/commun/IGCM
# At Jean-Zay: R_IN=/gpfswork/rech/psl/commun/IGCM
# At Irene:
R_IN=/ccc/work/cont003/igcmg/igcmg/IGC
M # At Obelix:
R_IN=/home/orchideeshare/igcmg/IGC
M
# At Ciclad/ClimServ: R_IN=/prodigfs/ipslfs/igcmg/IGCM

export year=2000

# Link the executable
ln -s ../modipsl/bin/orchidee_ol .

# Link input netcdf files
ln -s ${R_IN}/SRF/METEO/CRU-NCEP/v5.3.2/twodeg/cruncep_twodeg_${year}.nc forcing_file.nc
ln -s ${R_IN}/SRF/PFTMAPS/CMIP5/PFTmap_1850to2005_AR5_LUHa.rc2/PFTmap_IPCC_${year}.nc
PFTmap.nc ln -s ${R_IN}/SRF/WOODHARVEST/LUH2v2/historical4/woodharvest_${year}.nc
woodharvest.nc
ln -s ${R_IN}/SRF/SOIL/soils_param.nc soils_param.nc
ln -s ${R_IN}/SRF/SOIL/soil_bulk_and_ph.nc
soil_bulk_and_ph.nc ln -s
${R_IN}/SRF/cartepente2d_15min.nc cartepente2d_15min.nc
ln -s ${R_IN}/SRF/reftemp.nc reftemp.nc
ln -s ${R_IN}/SRF/albedo/alb_bg_modisopt_2D_ESA_v3.nc
alb_bg.nc ln -s ${R_IN}/SRF/ROUTING/routing.nc
routing.nc
ln -s ${R_IN}/SRF/NITROGEN/N_FERTILISATION/NMIP/synthetic/historical/Nfer_pasture_${year}.nc
nfert_pasture.nc ln -s
${R_IN}/SRF/NITROGEN/N_FERTILISATION/NMIP/synthetic/historical/Nfer_cropland_${year}.nc nfert_cropland.nc
ln -s ${R_IN}/SRF/NITROGEN/N_FERTILISATION/NMIP/manure/historical/Nmanure_pasture_${year}.nc
nmanure_pasture.nc
ln -s ${R_IN}/SRF/NITROGEN/N_FERTILISATION/NMIP/manure/historical/Nmanure_cropland_${year}.nc
nmanure_cropland.nc ln -s
${R_IN}/SRF/NITROGEN/N_DEPOSITION/CCMI_ndep/historical/CCMI_ndep_nhx_${year}.nc ndep_nhx.nc
ln -s ${R_IN}/SRF/NITROGEN/N_DEPOSITION/CCMI_ndep/historical/CCMI_ndep_noy_${year}.nc
ndep_noy.nc ln -s ${R_IN}/SRF/NITROGEN/BNF/bnf_1850.nc bnf.nc

# Copy xml files from the model. Don't forget to adapte them by changing _AUTO_ into true
values cp ../modipsl/modeles/ORCHIDEE/src_xml/* .

# Don't forget to create the run.def as well
```

### A.4 Example of a run.def parameter file for simple test case of the trunk

Create the parameter file by pasting following lines into a file with name run.def:

```
# Simulation length
TIME_LENGTH=31D

# Parameters for regional run :
LIMIT_WEST = -10.
LIMIT_EAST = 0.
LIMIT_NORTH = 10.
LIMIT_SOUTH = 0.

# Parameters for global run :
#LIMIT_WEST = -180.
#LIMIT_EAST = 180.
#LIMIT_NORTH = 90.
#LIMIT_SOUTH = -90.

# Set RIVER_ROUTING=n to deactivate the ROUTING module for regional domain
only RIVER_ROUTING=n

# File and variable name for nitrogen input files
*****
* Nammonium_FILE = ndep_nhx.nc
Nammonium_VAR = nhx
```

Nnitrate\_FILE = ndep\_noy.nc  
Nnitrate\_VAR = noy

Nfert\_FILE = NONE  
Nfert\_VAR = nfer

Nmanure\_FILE = NONE  
Nmanure\_VAR = Nmanure

Nfert\_cropland\_FILE = nfert\_cropland.nc  
Nfert\_cropland\_VAR = nfer

Nmanure\_cropland\_FILE = nmanure\_cropland.nc  
Nmanure\_cropland\_VAR = Nmanure

Nfert\_pasture\_FILE = nfert\_pasture.nc  
Nfert\_pasture\_VAR = Nfer

Nmanure\_pasture\_FILE = nmanure\_pasture.nc  
Nmanure\_pasture\_VAR = Nmanure

Nbnf\_FILE= bnf.nc  
Nbnf\_VAR= BNF\_MGN\_PERM2\_PERYR

## A.5 Set up for test simulation using ORCHIDEE 2.0

In this section you'll see how to set up a test case to run the ORCHIDEE 2.0 tagged version. For this version of the model, you need at least the following files in the run directory (you'll see further below how to get them):

- orchidee\_ol : ORCHIDEE executable
- run.def : parameter text file
- forcing.file.nc : climate forcing variables
- PFTmap.nc : vegetation map
- woodharvest.nc : woodharvest map
- soils\_param.nc : initialization of soil parameters
- alb\_bg.nc : background albedo from Modis
- iodef.xml, context\_orchidee.xml, context\_input\_orchidee.xml, field\_def\_orchidee.xml, file\_def\_orchidee.xml: parameter files for output and input settings using XIOS
- Optional: routing.nc, floodplains.nc, soils\_param\_usda.nc

Install the model using target ORCHIDEE\_2\_0 as the following:

```
svn co http://forge.ipsl.jussieu.fr/igcmg/svn/modipsl/trunk/modipsl
cd modipsl/util
./model ORCHIDEE_2_0
cd ../config/ORCHIDEE_OL
gmake
```

In this exercise you will set up a simple test case and run in sequential mode without using libIGCM. Create now a new directory outside modipsl to run the model and copy or link the ORCHIDEE executable:

```
cd TESTOFFLINE; mkdir RUN1; cd RUN1
ln -s ../modipsl/bin/orchidee_ol .
```

Create the parameter file by saving the following lines into a file named run.def:

```
TIME_LENGTH=31D
NVM=15
PFT_TO_MTC=1,2,3,4,5,6,7,8,9,10,11,12,13,10,10
LIMIT_WEST = -10.
LIMIT_EAST = 30.
LIMIT_NORTH = 70.
LIMIT_SOUTH = 30.
```

Copy xml files from models/ORCHIDEE/src\_xml directory into the run directory.

```
cp ../modipsl/modeles/ORCHIDEE/src_xml/* .
```

file\_def\_orchidee.xml describes the output files, frequencies and variable contents. Change all occurrences of AUTO according to the comments in the beginning of the file. Using libIGCM, these variables marked with AUTO are modified by the drivers. We suggest for these exercises to set daily output frequency. You can activate all files. Read more about the xml files in the appendix.

Copy or link the netcdf files from the shared repository IGCM into your run directory. The location of the shared repository IGCM depends on the machine but the content is synchronized between the different repositories. You can use export if your shell is bash (for tcsh shell: replace export by set):

```

# At Jean-Zay:
export R_IN=/gpfswork/rech/psl/commun/IGCM
# At irene:
export R_IN=/ccc/work/cont003/igcmg/igcmg/IGCM
# At obelix:
set R_IN=/home/orchideeshare/igcmg/IGCM
# At ciclad:
export R_IN=/prodigfs/ipslfs/igcmg/IGCM

ln -s $R_IN/SRF/METEO/CRU-NCEP/v5.3.2/twodeg/cruncep_twodeg_1901.nc forcing_file.nc
ln -s $R_IN/SRF/PFTMAPS/CMIP6/ESA-LUH2v2/historical/15PFT.v1/PFTmap_1901.nc PFTmap.nc
ln -s $R_IN/SRF/WOODHARVEST/LUH2v2/historical/woodharvest_1901.nc woodharvest.nc
ln -s $R_IN/SRF/cartepente2d_15min.nc .
ln -s $R_IN/SRF/SOIL/soils_param.nc .
ln -s $R_IN/SRF/albedo/alb_bg_modisopt_2D_ESA_v2.nc alb_bg.nc
ln -s $R_IN/SRF/reftemp.nc .
ln -s $R_IN/SRF/ROUTING/routing.nc .

```

You can use `ncdump` to see what is in the netcdf files. For example:

```
ncdump -h forcing_file.nc
```

Now launch the model:

```
./orchidee_ol      # or ./orchidee_ol > out_exec
```

When the execution is completed correctly, following log message is found in the output text file `out_orchidee 0000`:

```
END of dim2_driver
```



## B Appendix: Answers to questions

### B.1 Compile and run with debug options

**1. Difference in text output:** The executable writes more warning messages due to the compiler options that were used. If you want the model to output even more text output, you can set PRINTLEV=3 or higher in run.def parameter file. PRINTLEV can be used also for default compilation in prod mode.

**2. Difference in run time:** The model will run much slower when compiled with debug options.

#### 3. Example of a warning message:

forrtl: warning (406): fort: (1): In call to WOOD\_TO\_QMHEIGHT, an array temporary was created for argument #2

Image	PC	Routine	Line	Source
orchidee_ol	00000000031D0E36	Unknown	Unknown	Unknown
orchidee_ol	0000000002134EDA	stomate_growth_f u	803	stomate_growth_fun_all.f 90
orchidee_ol	0000000000E63400	stomate_lpj_mp_st	740	stomate_lpj.f90
orchidee_ol	0000000000BA1214	stomate_mp_stoma t	1637	stomate.f90
orchidee_ol	0000000000A231CF	slowproc_mp_slow p	740	slowproc.f90
orchidee_ol	00000000009516A1	sechiba_mp_sechib	839	sechiba.f90
orchidee_ol	0000000000592D09	intersurf_mp_int e	582	intersurf.f90
orchidee_ol	00000000004EDFF2	MAIN_	1285	dim2_driver.f90
orchidee_ol	000000000041CC62	Unknown	Unknown	Unknown
libc-2.17.so	00007FFFEAD53495	_libc_start_main	Unknown	Unknown
orchidee_ol	000000000041CB69	Unknown	Unknown	Unknown

The warning refers to the line 803 in the file stomate\_growth\_fun\_all.f90. Open the preprocessed file to see the corresponding line. The preprocessed files are found in the folder build/ppsrc/ which has been created during compilation in ORCHIDEE. In file ORCHIDEE/build/ppsrc/stomate/stomate\_growth\_fun\_all.f90, the line 803:

```
qm_height(ipts,j) = wood_to_qmheight(circ_class_biomass(ipts,j,1,;,icarbon), &  
    circ_class_n(ipts,j,;, j))
```

This warning only tells that a temporary variable has been created instead of circ\_class\_biomass(ipts,j,1,;,icarbon) which do not represent the full variable. The coding is here correct but in some cases it could be

costfull, it could take longer time to execute and that's why the warning is printed out.

#### 4. Find the error message:

forrtl: error (182): floating invalid - possible uninitialized real/complex variable.

Image	PC	Routine	Line	Source
orchidee_ol	00000000031D5054	Unknown	Unknown	Unknown
libpthread-2.17.so	00007FFFE40F5D00	Unknown	Unknown	Unknown
orchidee_ol	00000000022AE4A4	stomate_growth_fu	4555	stomate_growth_fun_all.f90
orchidee_ol	0000000000E63400	stomate_lpj_mp_st	740	stomate_lpj.f90 orchidee_ol
orchidee_ol	0000000000BA1214	stomate_mp_stomat	1637	stomate.f90
orchidee_ol	0000000000A231CF	slowproc_mp_slowp	740	slowproc.f90
orchidee_ol	00000000009516A1	sechiba_mp_sechib	839	sechiba.f90
orchidee_ol	0000000000592D09	intersurf_mp_inte	582	intersurf.f90
orchidee_ol	00000000004EDFF2	MAIN_	1285	dim2_driver.f90
orchidee_ol	000000000041CC62	Unknown	Unknown	Unknown
libc-2.17.so	00007FFFEAD53495	_libc_start_main	Unknown	Unknown
orchidee_ol	000000000041CB69	Unknown	Unknown	Unknown

The error is situated on the line 4555 in the file stomate\_growth\_fun\_all.f90. Open the file and look at the line, use the file in the build/ppsrc/stomate folder as indicated above. The line is the following:

```
update_sugar_load = (labile_pool+reserve_pool) / (biomass(ipts,j,ilabile,icarbon) + &  
    biomass(ipts,j,icarbres,icarbon))
```

We now know that one of the variables on this line present an error. It might be that the variable is not initialize, does not have any valid value or that it has the wrong dimension. Some compilers might also say which of the variables is bad. For example, the same test case at obelix, gives the following message:

```
forrtl: severe (193): Run-Time Check Failure. The variable
'stomate_growth_fun_all_mp_growth_fun_all_$LABILE_POOL' is being used without being defined
Image          PC          Routine          Line          Source
orchidee_ol    00000000038DE8B  Unknown         Unknown       Unknown
orchidee_ol    E
orchidee_ol    00000000038DD356  Unknown         Unknown       Unknown
orchidee_ol    000000000389090  Unknown         Unknown       Unknown
orchidee_ol    2
orchidee_ol    000000000385174B  Unknown         Unknown       Unknown
orchidee_ol    0000000002FA809D  stomate_growth_f  4555         stomate_growth_fun_all.f
orchidee_ol    u
orchidee_ol    00000000011699F1  stomate_lpj_mp_st  740          stomate_lpj.f90
orchidee_ol    0000000000D894EA  stomate_mp_stoma  1637         stomate.f90
orchidee_ol    t
orchidee_ol    0000000000B76128  slowproc_mp_slow  740          slowproc.f90
orchidee_ol    p
orchidee_ol    0000000000A70A70  sechiba_mp_sechib  839          sechiba.f90
orchidee_ol    000000000060FBFC  intersurf_mp_int  582          intersurf.f90
orchidee_ol    e
orchidee_ol    000000000052E2ED  MAIN_            1285         dim2_driver.f90
orchidee_ol    000000000041AC4C  Unknown         Unknown       Unknown
libc.so.6     00002AF39B51B50  Unknown         Unknown       Unknown
orchidee_ol    5
orchidee_ol    000000000041AB4F  Unknown         Unknown       Unknown
```

We here get the information that it is LABILE POOL which is wrong.

## B.2 SPINUP ANALYTIC

See here all commands needed for exercise SPINUP ANALYTIC with libIGCM:

```
cd modipsl/config/ORCHIDEE_OL
cp -r SPINUP_ANALYTIC_FG1 MyTestSpinup
cd MyTestSpinup

# Modify following variables in the config.card
JobName=MySpin
SpaceName=TEST
DateEnd=1940-12-31
OOL= (orchidee_ol, orchidee_ol, 1MPI)
IOS= ("", "")
TimeSeriesFrequency=5Y

# Add following in PARAM/run.def
LIMIT_WEST = -60.
LIMIT_EAST = -58.
LIMIT_NORTH = -8.
LIMIT_SOUTH = -10.

# Create the job
../../../../libIGCM/ins_job

# Set in the job Job_MyTestSpinup
PeriodNb=30

# Launch
sbatch Job_MySpin      # at Jean-Zay
ccc_msub Job_MySpin   # at irene
qsub Job_MySpin       # at obelix or ciclad
```

Answers to questions in exercise 4.1:

- 1. Why should you deactivate XIOS in server mode in this example?*  
The XIOS server is used to optimize the calculation time when running in parallel. We then use several cores for the ORCHIDEE model and 1 or more cores for the server. The server will take care of the reconstruction and transformation of the output variables while the ORCHIDEE executable continues calculating. In this example you only run a very small domain(1 grid-cell) using 1 core only. There is no need for reconstruction of output and therefor no need for the server. For this case it would take longer time to use server mode.
- 2. How do you calculate PeriodNb?*  
Find information in the plateforme documentation here:  
<http://forge.ipsl.jussieu.fr/igcmg/doc/wiki/Doc/Setup#ChoosingPeriodNb>
- 3. Where is the output stored?*  
The output is first written in the scratch directory at Irene or Jean-Zay, in a sub folder in IGCM OUT. If SpaceName=PROD or DEVT the output will be moved by the pack post- treatment to IGCM OUT on the store directory. If SpaceName=TEST, then the output will be left on the scratch.  
At Ciclad, the output will be written in IGCM OUT in /data/yourlogin directory and at Obelix in /home/scratch01/yourlogin. The pack post-treatment is not implemented at Ci- clad or at Obelix.  
*How can you change the place for the ARCHIVE directory? Note that this is recommended*

*only at obelix.*

At obelix, the output is stored in

/home/scratch01/yourlogin/IGCM

OUT/TagName/SpaceName/ExperimentName/JobName. Change this in config.card by adding in the UserChoices section: ARCHIVE=/home/yourdisc/yourlogin. It is only the permanent archive that will be changed. This means that if SpaceName=TEST, the output will still be at scratch01. The archive can also be changed directly in libIGCM/libIGCM sys/libIGCM sys obelix.ksh\_ - -

4. *In orchidee.ol.card you can use the variable year or CyclicYear. Which are the differces?*  
The variable year is incremented from DateBegin to DateEnd whears CyclicYear loops from CyclicBegin to CyclicEnd over and over again.
5. *The variable SPINUP PERIOD is calculated by the stomate.driver and set in run.def. Where can you see the run.def file that was used during simulation? What is the SPINUP PERIOD?*  
.../yourlogin/IGCM  
OUT/OL2/TEST/ExperimentName/JobName/OOL/Debug/\*run.def SPINUP  
PERIOD= 10