Hands on exercises with ORCHIDEE OFF-LINE

Revised for training session 2020-01-16 - 2020-01-17

Josefine Ghattas/IPSL, Fabienne Maignan/LSCE

Contents

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.

1.1 At Jean-Zay during the training session

You can change the keyboard language on the training computer by touching **shift + alt** at the same time. You can also change keyboard with spare ones to be more comfortable during the training sessions.

During the training session everybody works on temporary training accounts at IDRIS. These acounts have specific priority in the queue to avoid too much waiting time. First connect to **ipcours** using a login **cfor0XX** and the password **Climat0120**, then open a terminal and connect to jean-zay via ssh as follow:

```
ssh -Y jean-zay4
```

Note: we advice to change the colors in the terminal window to more easily read when using vi. To do so, use the menuboard and choose Edit/Preferences.

Install the IPSL-cmc environnement using the commands below, on your account during the first connexion to jean-zay and add export SBATCH_RESERVATION. The SBATCH_RESERVATION variable is a specific reservation with computing resources available only during the training session and it changes each day. It can not be used later. Do following using the SBATCH_RESERVATION as written below (identifier given by the command **scontrol show reservation**):

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

Now open the file .bash_login and add one of the following lines, depending on the day, at the end of the file. For example use vi or emacs to open the file.

```
vi .bash_login
export SBATCH_RESERVATION=for@cpu_94      # Thursday 16/01/2020
export SBATCH_RESERVATION=for@cpu_95      # Friday 17/01/2020
```

After editing .bash_login, source the file or deconnect and reconnect to jean-zay to take into account the changes.

During the training session, you'll install the model on the $SCRATCH. This is done to have enough space to compile the model because the training accounts are more limited in quota. When you work on your proper login, 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).

During the training session this year, you need to update the revision of libIGCM. This should not be done when you work on your own login. Do as follow as soon you installed a new modpsl with ORCHIDEE:

```
cd modipsl/libIGCM
svn update
```

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

Do not this exercise during the training session. 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
```

Deconnect 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.

1.3 First time at irene/TGCC

Do not this exercise during the training session. 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
```

Deconnect 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 appendix.

1.4 First time at obelix/LSCE

Do not this exercise during the training session. 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 interactivly. 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.

1.5 First time at Ciclad or ClimServ / IPSL ESPRI-mesocenter

Do not this exercise during the training session. At Ciclad, add following in your .bashrc file(create the file if it doesn't exist):

```
# User specific aliases and functions
source /home/igcmg/MachineEnvironment/ciclad/atlas_env_ciclad
```

Or if you work at ClimServ, add following in your .bashrc file(create the file if it doesn't exist):

```
# User specific aliases and functions
source /ciclad-home/igcmg/MachineEnvironment/climserv/atlas_env_climserv
```

Deconnect 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 at Ciclad and in /homedata/login at ClimServ.

1.6 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 preivous occurence 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 desactivate colors if needed
nG             # go to line n in the file
```

emacs is another text editor. It can be openend in a separate window and it has a menu bord 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
svn co http://forge.ipsl.jussieu.fr/igcmg/svn/modipsl/trunk 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 begining with ORCHIDEE_trunk, close the file without any changes and then extract and compile ORCHIDEE as follow:

```
./model ORCHIDEE_trunk
cd ../config/ORCHIDEE_OL
gmake
```

While the compilation is on going, open a new terminal, connect to jean-zay(irene, obelix or ciclad) as in the beginning and contine 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 makefiles. 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
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
```

Finaly 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 trough 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 loosing 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
gmake
```

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:

```
<field id="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 recommanded 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 additionnal 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
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
vi Makefile # => change -prod into -debug
gmake
```

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 messure 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 explainations in the appendix B.

5 Exercises to learn SVN

This exercise aims to learn the basic use of svn. The setup for the exercise is only available at Jean-Zay during the training session. 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 /gpfswork/rech/psl/commun/TRAINING/ORCHIDEE_20200116/modipsl/modeles/ORCHIDEE.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 /gpfswork/rech/psl/commun/TRAINING/ORCHIDEE_20200116/modipsl/modeles/ORCHIDEE.3
```

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 /gpfswork/rech/psl/commun/TRAINING/ORCHIDEE_20200116/modipsl/modeles/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_max.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 resolve 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 happend to the file stomate_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?

6 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 lenght 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 recommanded 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 modelized. 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 13. This number must correspond to the number of PFTs in the vegetation map (PFTmap.nc). For example, the default set up for ORCHIDEE_2_0 which is used for the CMIP6 simulations uses NVM=15.
- **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 can not be changed from run.def. Make this possible by adding call to getin_p in thermosoil_initialize. The default values should be the same as before. Add also 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 libIGCM experiment such as OOL_SEC_STO_FG2.

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 defaut 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 recomanded only for debugging or developement 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 detectes 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 interactivly. The job file will be different for different machines, choose the job for jean-zay as in the appendix.

7.2 Check reproductibility 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 current version of the trunk, 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, on 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 developped at IPSL as an extension of IOIPSL.

rebuild is installed at the different machines here:

```
# at Jean-Zay:
/gpfswork/rech/psl/commun/Tools/rebuild/modipsl_IOIPSL_PLUS_v2_2_4/bin
# at Irene:
/ccc/cont003/home/igcmg/igcmg/Tools/irene/rebuild/modipsl_IOIPSL_PLUS_v2_2_4/bin
# at Obelix:
/home/orchideeshare/igcmg/IOIPSL_PLUS/modipsl.tagv2_2_4.28082019/bin
# at Ciclad:
/home/igcmg/rebuild/src_X64_CICLAD/modipsl_v2_2_3_netcdf4.2/bin
# at ClimServ (same as Ciclad but the path is different):
/ciclad-home/igcmg/rebuild/src_X64_CICLAD/modipsl_v2_2_3_netcdf4.2/bin
```

If you've installed the IPSL environnment (.bash_login at Jean-Zay) you already have rebuild in your path. Try tygping *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 follow the same structure but this experiments is currently under developement. 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 alredy 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 modipsl/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 periodes (40 years in total). In config.card, set DateEnd=1940-12-31. Set SpaceName=TEST to deactivate pack post treatement if running at Jean-Zay or Irene. Set TimeSeriesFrequency=5Y and SeasonalFrequency=NONE.
It is important to adjust the numer of cores used (number of MPI and OMP) to the domain which is used. Here we run on 1 grid-cell and therefor 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 diffent 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 developpement. 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 lenght 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 shoudl the file be added and why ?
- Output the variable floodplains in 2 files by frequence 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 analyse 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 environnement* for MPI in the on-line documentation in chapter Computing Centers:
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 domaine after the run. But when running in server mode using 1 server, the server will output on the full domaine no rebuild is needed. Use libIGCM 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 numbre 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 correponds to Jean-Zay. Change this path for other platforms. If your terminal shell is csh or tcsh, 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/IGCM
# At Obelix:   R_IN=/home/orchideeshare/igcmg/IGCM
# 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 occurences of _AUTO_ according to the comments in the beginning of the file. Using libIGCM, these variables marqued 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_fu         803  stomate_growth_fun_all.f90
orchidee_ol        0000000000E63400  stomate_lpj_mp_st         740  stomate_lpj.f90
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 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.s  00007FFFEB40F5D0  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        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
Aborted (core dumped)
```

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        00000000038DE8BE  Unknown               Unknown  Unknown
orchidee_ol        00000000038DD356  Unknown               Unknown  Unknown
orchidee_ol        0000000003890902  Unknown               Unknown  Unknown
orchidee_ol        000000000385174B  Unknown               Unknown  Unknown
orchidee_ol        0000000002FA809D  stomate_growth_fu        4555  stomate_growth_fun_all.f90
orchidee_ol        00000000011699F1  stomate_lpj_mp_st         740  stomate_lpj.f90
orchidee_ol        0000000000D894EA  stomate_mp_stomat        1637  stomate.f90
orchidee_ol        0000000000B76128  slowproc_mp_slowp         740  slowproc.f90
orchidee_ol        0000000000A70A70  sechiba_mp_sechib         839  sechiba.f90
orchidee_ol        000000000060FBFC  intersurf_mp_inte         582  intersurf.f90
orchidee_ol        000000000052E2ED  MAIN__                   1285  dim2_driver.f90
orchidee_ol        000000000041AC4C  Unknown               Unknown  Unknown
libc.so.6          00002AF39B51B505  Unknown               Unknown  Unknown
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-treatement is not implemented at Ciclad 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