Getting started with the IPSL tools: modipsl and libIGCM

Exercises for Training course

Revised for training session January 2020

L. Falletti, N. Lebas

A. Caubel, A. Cozic, C. Ethé, L. Fairhead, S. Flavoni, J. Ghattas, R. Pennel, R. Person, J. Servonnat

---

!!! Please read this first introduction carefully !!!

The aim of this document is to give you all the information to know about how to install, compile and launch simulations with reference configurations using ***modipsl*** and ***libIGCM*** environment.
During the exercises, we show you step by step how to handle these tools and simulations but you will have to search in the IGCMG documentation for all of the details : http://forge.ipsl.jussieu.fr/igcmg_doc. It's all part of the training!

The present document contains an introduction section (0.) following by 13 sections with exercises (see the table of contents thereafter). Depending on your knowledge of modipsl and libIGCM, we advise you to use this document as follows:

- ***For beginners*** (if you never used the tools or just a little bit), first you have to focus on sections 1 and 2 which detail how to *install, compile and launch a basic simulation*. Note that subsection 2.6 is only useful for LMDZ users (LMDZ and LMDZOR).
  If you have time, you can then continue with sections 3 to 7.
  If you finish all of them, you can then choose some other exercises from section 8 to 13, depending on your future use of the tools.
- ***For more advanced users***, we advice to still start with sections 1 and 2 as you will need the *basic simulation* for other sections. But you should not spend too much time on these two sections.
  Then continue with sections 3 to 7 to learn about *debugging, post-processing and monitoring*.
  If you finish all of them, you can then choose some other exercises from section 8 to 13, depending on your future use of the tools.

Note on environment variables:

In this document, we mainly use the disk spaces' environment variables for IDRIS (`$WORK`…) as for today's training we work on an IDRIS' machine.
Pay attention that they are not the same for other computing center (for instance it's `$CCCWORKDIR` on irene). You can read more details on the IGCMG documentation.

Here for IDRIS files systems
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/ComputingCenters/IDRIS#Thingstoknowaboutfilesystems

Here for TGCC files systems
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/ComputingCenters/TGCC#Aboutfilesystems

Table of content

0. Introduction **5**
0.1 Essentials notes on today's training 5
0.2 Subscribe to plateform-users mailing list 6
0.3 How to correctly install your environment? 6

1. Install and compile **7**
1.0 Install modipsl 7
1.1 Extract LMDZOR_v6 configuration 8
1.2 Compile with the resolution 144x142x79 10

2. Basic simulations **13**
2.1 Create first experiment directory 13
2.2 Define and launch your first simulation of 1 day 15
2.3 Continue the simulation 4 more days 22
2.4 Create another simulation with pack 23
2.5 Use differents forcing files 24
2.6 CREATE_clim and CREATE_amip : Experiments to create initial state files and boundary conditions for LMDZ 25
2.7 Summary on how to extract, compile and launch a simulation 26

3. Debug **28**
3.0 How can you analyze the Job Output : Script_Output ? 28
3.1 Debug : setup error 28
3.2 Debug : error during the simulation 29
3.3 Compilation in debug mode 30

4. Create time series **34**
4.1 Launch 10 years with default time series 34
4.2 Use supervisor during run time 35
4.3 Add variables to time series and relaunch with the TimeSeriesChecker.job 38

5. Monitoring and Inter-monitoring **40**
5.1 Monitoring 40
5.2 Inter-monitoring 40
5.2.1 from supervisor interface 40
5.2.2 from web interface tool "webservices" 40

6. Modify output using XIOS **42**
6.1 Create a new output file for ORCHIDEE 42
6.2 Enable a new output file in LMDZ 44
6.3 XIOS in other models 47

7. Check your quota **48**
7.1 For login at IDRIS 48

7.2 For login at TGCC 48
7.3 For login at LSCE/obelix 49

8. Install and run NEMO-PISCES 50

9. REDO 56
9.1 Launch a 3 days simulation of LMDZOR experiment 56
9.2 Remove daily output file for ATM component of the day 2 (i.e 1980-01-02) 57
9.3 Apply the method to redo day 2 of the simulation (to recover missing output file) 57

10. Output files manipulations 60
10.0 Protocol and environment 60
10.0.1 Protocol 60
10.0.1 Environment 60
10.1 Network Common Data Form (NetCDF) format 61
10.2 NetCDF Operator (NCO) 61
10.3 Climate Data Operators (CDO) 62
10.3 NetCDF Visual browser (NCView) 63
10.4 Ferret 64
10.5 NCAR Command Language (NCL) 65
10.6 Python 67
10.6.1 NetCDF4 / Numpy 67
10.6.1 XArray 68

11. CliMAF and the C-ESM-EP 71

12. Ensembles 73

13. Coupled model 76

0. Introduction

0.1 Essentials notes on today's training

All exercises can be done at ***Irene/TGCC*** or at ***Jean-Zay/IDRIS*** and most of them at ***obelix/LSCE***, read specifications in the text.
Note that for this training session we will work only on IDRIS temporary accounts. There are a few specific commands that you will not need when you will work on other machines and they are marked as "Today on Jean-Zay".

All commands needed for the basic exercises are listed in the text. Exercise using NEMO configuration is proposed as a complement.

Today on Jean-Zay: use training account

During the training session, specific training accounts on Jean-Zay will be used. They have login `cforXXX` with password ****. Connect first to the machine **ipcours** and then use your temporary login.

If you need to switch between qwerty and azerty you can use the command **alt+shift**.

To access Jean-Zay, open a terminal and type the command:

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

For your first connexion to Jean-Zay, you need to install the IPSL environment. Do the following:

```
cp
$WORK/../../../rech/psl/commun/MachineEnvironment/jeanzay/bash_log
in $HOME/.bash_login # warning it's rps"L" and not rps"one"
rm $HOME/.bash_profile
source $HOME/.bash_login
```

Set the following environment variable (complete with the return of `ReservationName` lists by the command `scontrol show reservation`) :
You have to modify $HOME/.bash_login as follows :

```
vi $HOME/.bash_login
# Add export SBATCH_RESERVATION=*****
# Replace ***** by for@cpu_92 for the 14th and for@cpu_93 for the
# 15th.
source $HOME/.bash_login
```

0.2 Subscribe to plateform-users mailing list

Before working with **modipsl/libIGCM** and IPSL's models, you need to subscribe to the **plateform-users mailing list**. Do this by following the link :

https://listes.ipsl.fr/sympa/info/platform-users

Exercises proposed in this training session are using LMDZOR_v6 (LMDZ + ORCHIDEE) and ORCHIDEE_trunk (ORCHIDEE offline) configurations. But everything you will learn will be usable with all models configuration (IPSLCM6, LMDZORINCA, etc.)

0.3 How to correctly install your environment?

Before working with modipsl/libIGCM on IDRIS or TGCC you need to install your environment. For this you will find all necessary information here:

https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/ComputingCenters

1. Install and compile

In this section, you will learn how to install the tools and compile the configuration.

Start with creating a new directory in your `$WORK`. (Warning : it's `$CCCWORKDIR` on irene)

```
mkdir $WORK/MYFIRSTTEST ; cd $WORK/MYFIRSTTEST
```

1.0 Install modipsl

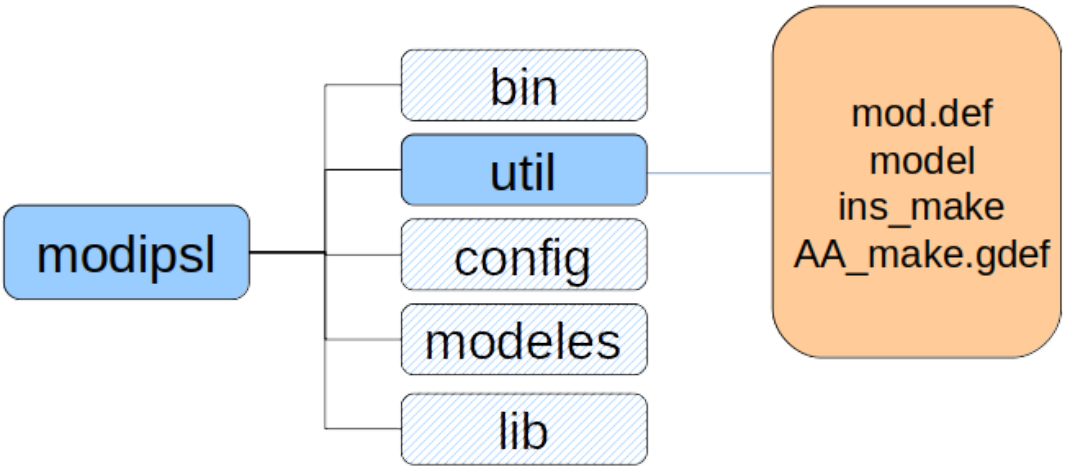
Download modipsl:

```
svn co http://forge.ipsl.jussieu.fr/igcmg/svn/modipsl/trunk modipsl
```

Other method: as you have installed IPSL environment, you can use the command ***`svn_ano`*** instead of previous one to download modipsl.

```
mkdir $WORK/MYTESTALIAS ; cd $WORK/MYTESTALIAS ; svn_ano
```

Explore **`modipsl/`** directory. You can see that some directories are empty. To download one models configuration and create its makefile, you will use script store in `modipsl/util/` directory.
Compare your `modipsl/` tree and the following diagram.



You can find the description of all these directories here :
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Install#Themodipsldirectories

Scripts stored in `util/` directory can be used to :

- Choose the models configuration to download (`mod.def`)
- Download this configuration (`model`)
- Create a makefile adapt to this configuration and the supercomputer where you are working (`ins_make`, `AA_make.gdef`)

Explore `util/` directory:

```
cd $WORK/MYFIRSTTEST/modipsl/util
ls
```

1.1 Extract LMDZOR_v6 configuration

Description : The script ***`model`*** is used to download a specific predefined configuration with the model source codes and tools needed. The script uses the file ***`mod.def`*** that contains specifications for each predefined configuration. Use the command ***`./model -h`*** to see all existing configurations and ***`./model -h config_name`*** for information of a specific configuration. Same information can be found by reading *`mod.def`* file. You can find information on how you can read *`mod.def`* file on this page :
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Install#Syntaxinmod.def

Question 1a Using ***`./model -h`*** command, find which version of LMDZ, ORCHIDEE and libIGCM are currently defined in the configuration LMDZOR_v6.1.5 ? Note the SVN revision number and SVN branch or tag name. Verify that you can find the same information in mod.def file.

Note on Subversion (SVN) - a version control software :

IPSL models are saved via svn, this allows to keep track of changes done over the time, backup and store all previous versions, centralize all existing developments done on each model.

Each modification on svn will match with a revision number and a save path (with prefix trunk, tag or branches). To know them, you should use the command *svn info*.

```
cd $WORK/MYFIRSTTEST/modipsl/util
./model -h
./model -h LMDZOR_v6.1.10
vi mod.def
```

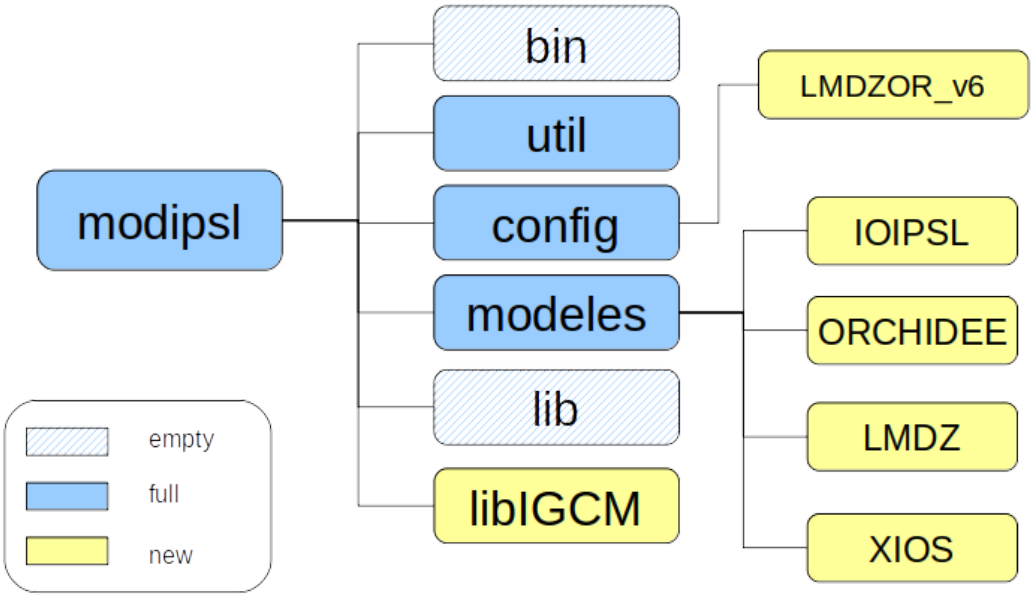
Now download the configuration LMDZOR_v6.1.10 by using the script **model**. Note : for the first extraction the password for ORCHIDEE is needed.

```
./model LMDZOR_v6.1.10
```

When prompt for password :

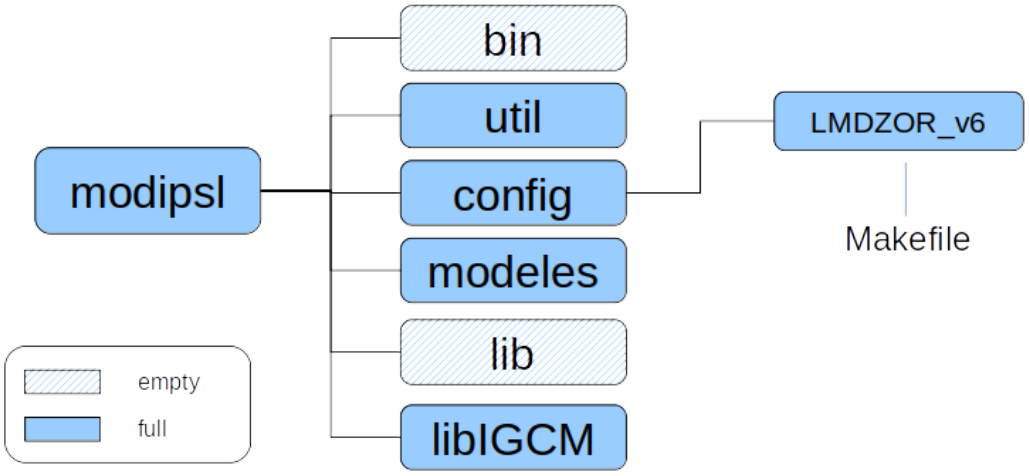
hit enter and then use ORCHIDEE login credentials written on the room's blackboard.

Now explore the directories in modipsl. You can see in *modipsl/modeles* that you have one directory per model. You also find the directory *modipsl/config/LMDZOR_v6* and the directory *modipsl/libIGCM*. Type *svn info* in each model directory to get information about the extracted version and compare them with your answer to the question 1.a.



1.2 Compile with the resolution 144x142x79

The makefiles were automatically created by the script ***`ins_make`*** that was launched at the end of the script ***`model`***. `ins_make` will detect your environment and will create adapted makefiles. By default `ins_make` recognizes the following environnements : irene at TGCC, jean-zay at IDRIS, obelix at LSCE and ciclad at IPSL.
`ins_make` can also be launched manually. For example this is needed if you move the modipsl directory or if you create makefiles for another target machine.
The main `makefile` is found in `config/LMDZOR_v6` directory.



Question 1b Open the main Makefile and try to find all resolutions available for the compilation. Find which resolution is the default one, then launch compilation for the resolution 144x142x79. You can use these page
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Compile#Themainmakefile
to help you to understand the Makefile syntax.

We will use the LMDZOR default resolution 144x142x79 for the atmosphere. To compile you will use gmake command and in option the chosen resolution:

```
cd $WORK/MYFIRSTTEST/modipsl/config/LMDZOR_v6
gmake LMD144142-L79
```

Today on Jean-Zay :

Launch the compilation as explained above.

In case that the compilation duration is too long, you can copy the executable in your bin directory:

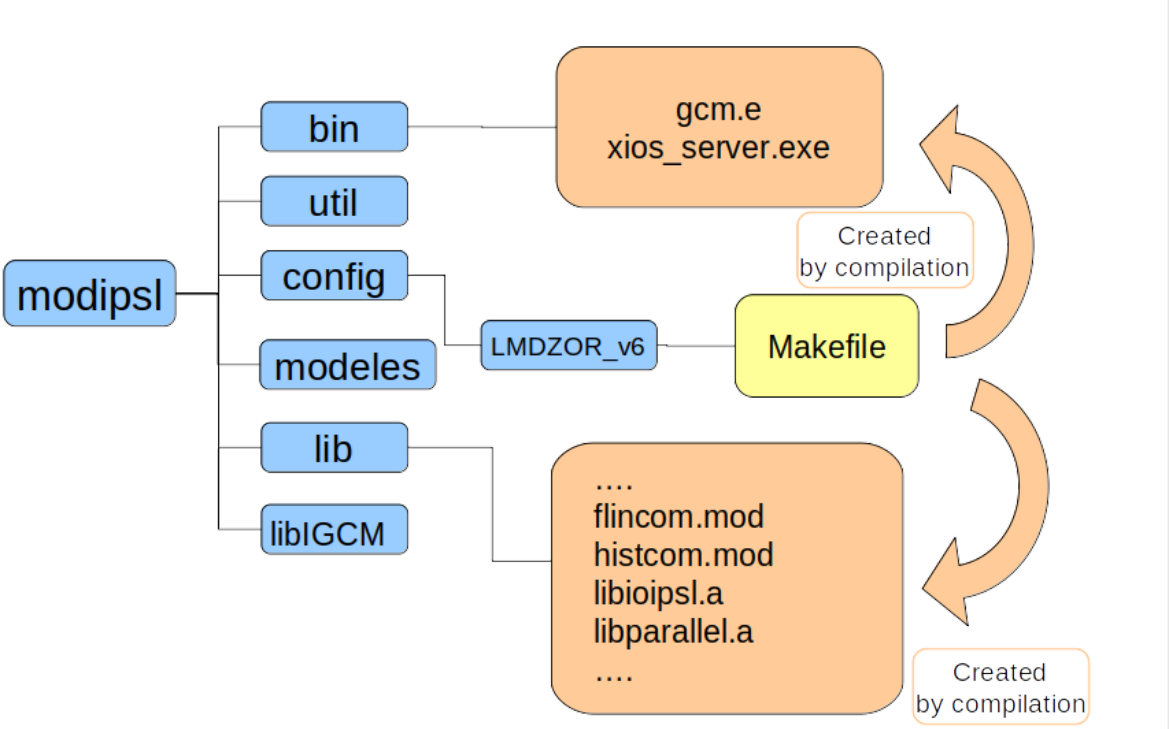
```
cp
$WORK/../../../rech/psl/commun/TRAINING/MODIPSL_HandsOn_20200114/
bin/* $WORK/MYFIRSTTEST/modipsl/bin/.

And create the postcompilation file .resol (it will be use by the
simulation to find which model is compiled and at which
resolution)

cd modipsl/config/LMDZOR
vi .resol
>> write these 2 lines
noORCAxLMD144142-L79
RESOL_ATM_3D=144x142x79
```

Comments on compilation

The compilation creates executables which are necessary for the launch of the simulation. Note that the executables are done for the specific configuration of models that you have downloaded (see 1.1 section).

When the compilation is over you will find executables in the directory *modipsl/bin* and a file *.resol* is created in *modipsl/config/LMDZOR_v6*. The compilation takes between 30 and 60 minutes depending on the platform. After the compilation, if you run *gmake* again, only modified files and files depending on them will be compiled.

Remember to verify that the executables are present in the directory ***`modipsl/bin`*** !



Question 1c How can you do if you want to recompile the whole code? Open the Makefile and check the different targets.

Specific installation of LMDZ at obelix/LSCE:

Read more about using LMDZ at obelix here:
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/ComputingCenters/LSCE

2. Basic simulations

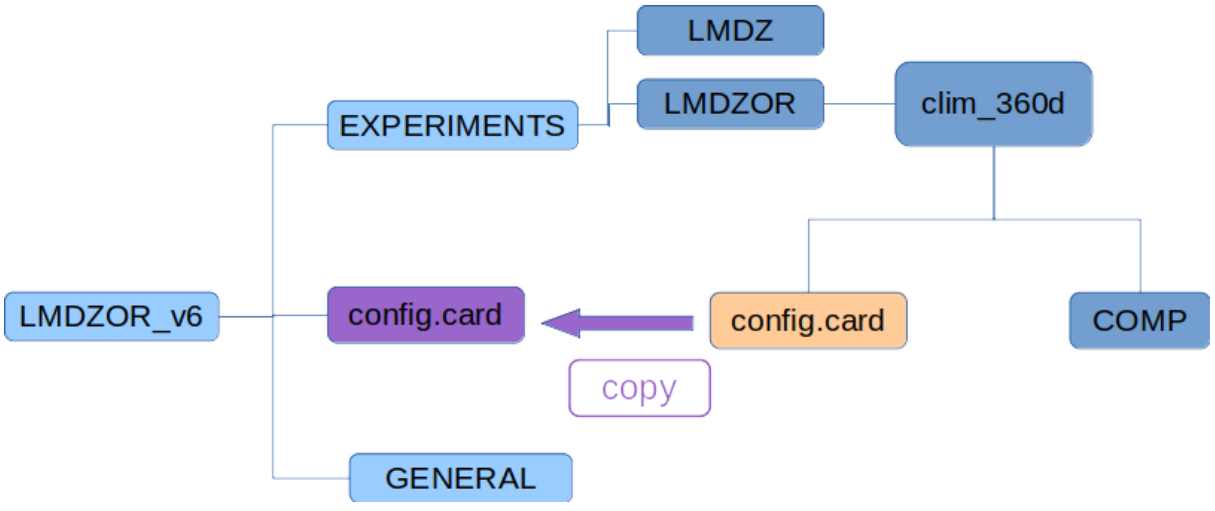
In a configuration with a same executable we can choose between several types of experiments. All experiments available are stored in `EXPERIMENTS/` directory.
For example, with LMDZORINCAREPR configuration (lmdz + orchidee + inca + reprobus) you can launch a lmdz simulation, or lmdzor, or lmdzorinca, or lmdzrepr, or lmdzorincarepr and all of them shared the same executable.

So once you choose with which models configuration you want to work, you have to download it and make the compilation, now you can choose which type of experiment you want to use.

2.1 Create first experiment directory

In `EXPERIMENTS` directory you can find different predefined experiments which you can possibly run using the configuration you extracted. For the LMDZOR_v6 case, you can choose between LMDZOR and LMDZ type of experiments.

For this exercise we will create an experiment from `LMDZOR/clim_360d`. To do this we copy the `config.card` found in `EXPERIMENTS/LMDZOR/clim_360d` to the directory `config/LMDZOR_v6/`.

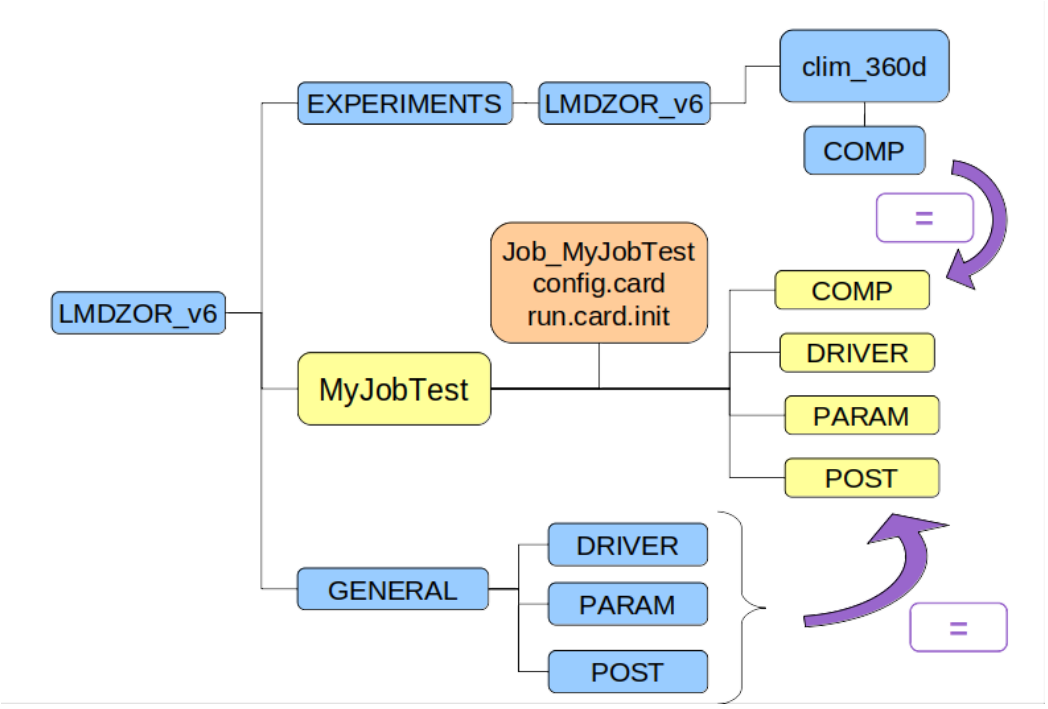


The simulation directory will be created with information which are found by libIGCM in the file `config.card`. Before creating this directory, we need at least to indicate the simulation name.
The script `modipsl/libIGCM/ins_job` will be used to create the simulation directory.

At obelix, change in config.card the number of MPI processes to use 7, OMP threads to use 1, and XIOS to use 1MPI.
At Irene change nothing.
At Jean Zay, change in config.card the number of MPI processes to use 71MPI, and OMP threads to use 5 (for training day) and 10 (for other days)

```
cd $WORK/MYFIRSTTEST/modipsl/config/LMDZOR_v6
cp EXPERIMENTS/LMDZOR/clim_360d/config.card .

vi config.card
      # Modify JobName=MyJobTest
      # Modify Executable part for the parallelization
[Executable]
ATM= (gcm.e, lmdz.x, 71MPI, 8OMP)
SRF= ("", "")
SBG= ("", "")
IOS= (xios_server.exe, xios.x, 1MPI)

       # At obelix only, change to 7 MPI and 1 OMP in
       # At Irene, change nothing for parallelization
       # At JeanZay, change 8 OMP by 5 or 10
       # At JeanZay training day change to 5 OMP

../../libIGCM/ins_job   # At JeanZay enter your project ID
                         #  At  Irene  enter  your  project  ID  and  default
answer for other questions

cd MyJobTest
```

The submission directory has been created with the same name as the `JobName`. Explore this directory and compare to the following diagram.



2.2 Define and launch your first simulation of 1 day

In this subsection, you will prepare and launch your first test simulation.

Generally, before any important experiment, it is good practice to check the good behaviour of the workflow with a test simulation. In particular, we need to check that pre and post processing stages do not induce any errors and that the simulation meet our expectations.

How to define a simulation?

To define a simulation, you need to answer the following questions :

1. **Which date to start and finish the simulation ?**
2. **Is your simulation a TEST, DEVT or PROD ? This choice define where your simulation output will be store**
   https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Running#Theoutputfiles
3. **Which calendar will you use ?**
   https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Setup#config.card
4. **Which initial states files ?**
5. **Which boundaries files ?**
6. **Which output variables ? With which frequency ?**
7. **Which post-processing ?**

If the simulation is a TEST (like in this exercise) we can not answer the last question (number 7). For this training session, we will use default arguments in `config.card` for questions 3, 4, 5 and 6. The 7th question will be seen in a later exercise.

Setup the `config.card`

You must be in the directory specially created for your simulation (`MyJobTest/`).

Now setup the `config.card` to do a short 1 day simulation. (`DateEnd` = last day of simulation):

```
DateBegin=1980-01-01
DateEnd=1980-01-01
```

This is a first test simulation so keep `SpaceName=TEST`. This option will deactivate pack functions and no archiving will be done. Output will therefore be found on `$SCRATCHDIR` (Irene) or `$SCRATCH` (JeanZay).

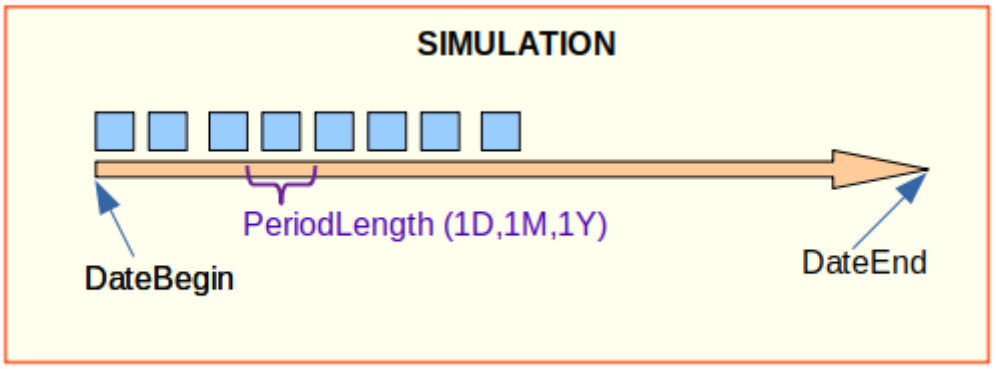
```
JobName=MyJobTest
#----- Short Name of Experiment
ExperimentName=clim
#----- DEVT TEST PROD
SpaceName=TEST
LongName="LMDZOR configuration"
TagName=LMDZOR
#D- Choice of experiment in EXPERIMENTS directory
ExpType=LMDZOR/clim_360d
```

For a 1 day simulation you will indicate `PeriodLength=1D`:

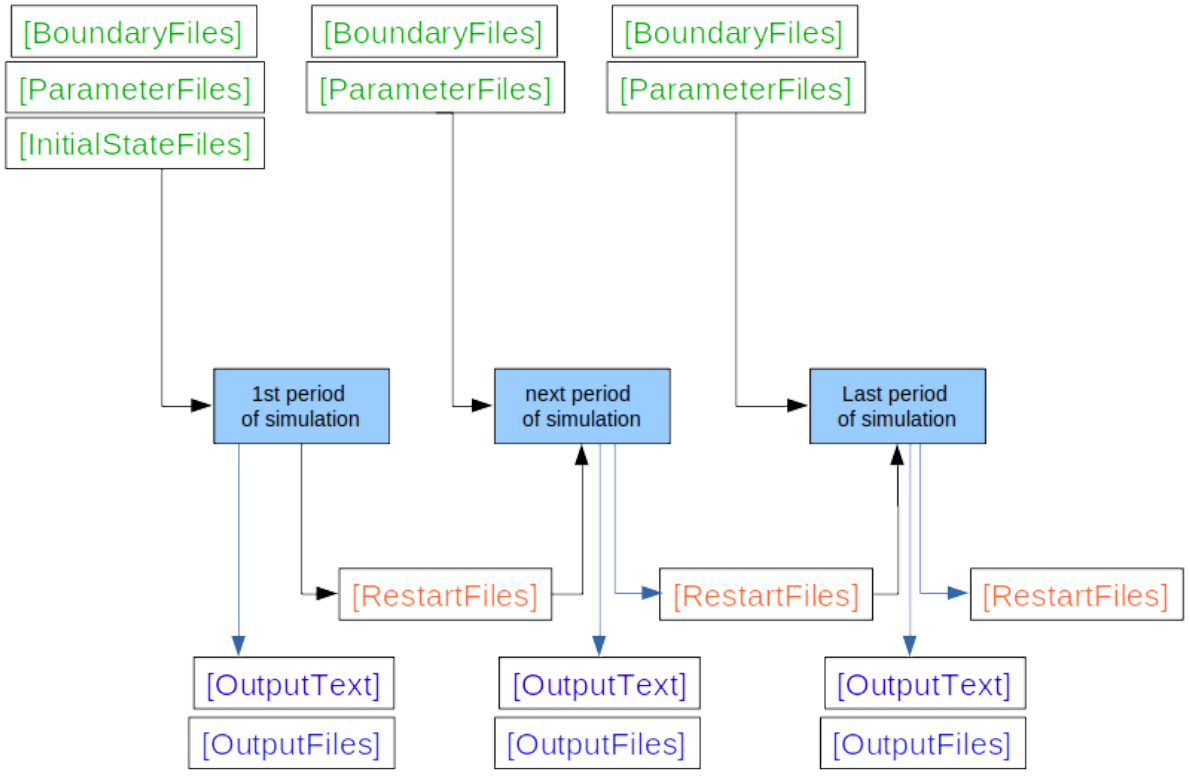
```
PeriodLength=1D
```

What is a period?

A simulation is a succession of **periods**.



At the end of each of them the simulation create outputs files use for some of them as inputs files for the next period.



Post-processing in `config.card`

We will deactivate all post-processing in `config.card` (you will see how to use them in sections 2.4 and 4.):

```
#D-- Post -
[Post]
#D- PackFrequency determines the frequency of pack submission
PackFrequency=1Y
```

```
#D- TimeSeriesFreqency determines the frequency of post-processing submission
#D- Set NONE to deactivate the creation of all time series
TimeSeriesFrequency=NONE
#D- SeasonalFrequency determines the length for each seasonal average
#D- Set NONE to deactivate the creation of all seasonal avereage
SeasonalFrequency=NONE
#D- Offset for seasonal average first start dates ; same unit as SeasonalFrequency
#D- Usefull if you do not want to consider the first X simulation's years
SeasonalFrequencyOffset=0
#D- If you want to produce compute PCMDI metrics from seasonal average
#D- Set TRUE or FALSE to activate/deactivate the metrics computation.
MetricsPCMDI=FALSE
```

The main job `Job_MyJobTest`

This file is the one which is used by the job scheduler to launch the simulation. It needs some information in the header which are specific to the machine that you are using.

Now you have to verify the header in the main job `Job_MyJobTest` and then you can submit the job.
You can find here https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Setup#Jobheaders a documentation on job headers syntax for Irene and Jean Zay.

To launch a test (on Jean Zay or Irene) you need to modify the CPU time and indicate that you will use the queue test.

- **Header for Jean Zay:**

```
#######################
## JEANZAY    IDRIS ##
#######################
#SBATCH --job-name=MyJobTest    # Job Name
#SBATCH --output=Script_Output_MyJobTest.000001       # standard output
#SBATCH --error=Script_Output_MyJobTest.000001        # error output
#SBATCH --nodes=9
#SBATCH --exclusive
#SBATCH --ntasks=72   # Number of MPI tasks
#SBATCH --hint=nomultithread        # 1 processus MPI par par physical core (no
hyperthreading)
#SBATCH --time=00:30:00             # Wall clock limit (seconds)
#SBATCH --account for@cpu
#SBATCH --qos=qos_cpu-dev                          # Queue test
```

- **Header for Irene:**

```
#######################
## IRENE   TGCC/CEA ##
#######################
#MSUB -r MyJobTest                              # Job name
#MSUB -o Script_Output_MyJobTest.000001 # Standard output
#MSUB -e Script_Output_MyJobTest.000001 # Error output
#MSUB -eo
#MSUB -n 976                                    # Number of MPI tasks allocated
#MSUB -x                                        # Node exclusivity
#MSUB -T 1800                                   # Wall clock limit (seconds)
#MSUB -Q test                                   # Test queue (max: 1800 seconds)
#MSUB -A dekcmip6                               # Project allocation
#MSUB -q skylake                                # Partition used
#MSUB -m store,work,scratch                     # Visible spaces
```

(for Irene, the wall clock limit for test queue is 1800 seconds maximum, if you are not running on test you can ask for 86400 seconds max).

For the training day on jeanzay with cfor account :
We need to define some libIGCM variables for the storage and run directories of the simulation.

In `config.card` add **`ARCHIVE=$STORE`**

```
#============================
JobName=MyJobTest
ARCHIVE=$STORE
#----- Short Name of Experiment
ExperimentName=clim
```

In the `main Job` discomment **`RUN_DIR_PATH`** variable, and add **`R_OUT`** and **`R_BUF`** like this

```
#D- Define running directory
#D- Default=${TMPDIR} ie temporary batch directory
#D-
RUN_DIR_PATH=$SCRATCH/RUN_DIR
R_OUT=$SCRATCH
```

```
R_BUF=$SCRATCH
#
```

Launch the job

Now, use one of these commands (depending on your machine) to launch the job:

`sbatch` (IDRIS) / `ccc_msub` (TGCC) / `qsub` (Obelix)

```
cd $WORK/MYFIRSTTEST/modipsl/config/LMDZOR_v6/MyJobTest/

JeanZay: sbatch Job_MyJobTest
Irene: ccc_msub Job_MyJobTest
Obelix: qsub Job_MyJobTest
```

The file `run.card`: to follow the status of your simulation

To know the status of your simulation a file `run.card` is created. Please read the pages https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Running#Statusoftherunningsimulation and https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Running#Endofthesimulation for more information.

Then, use this file to check if your simulation is well finished.

You can also use the following commands to check the job queue and check if your simulation is waiting/is still running/has finished:

`squeue` (IDRIS) / `ccc_mpp` (TGCC) / `qstat` (Obelix)

To see only yours jobs you can add the option `-u $user`.

```
JeanZay : squeue -u $user
Irene: ccc_mpp -u $user
```

How to delete a job?

`scancel` (IDRIS) / `ccc_mdel` (TGCC) / `qdel` (Obelix) followed by your job ID:

```
On JeanZay
squeue -u $user
>> JOBID PARTITION    NAME  USER ST    TIME  NODES NODELIST(REASON)
>>  389685 cpu_p1 LMDZOR02  rpsl592  R     11:05 15 r5i7n[9-23]
```

```
scancel 389685
```

On Irene

```
ccc_mpp -u $user
>> USER          ACCOUNT    BATCHID  NCPU   QUEUE      (...)
>> p24cozic      aercmip6   3351314  624    skylake    (...)
ccc_mdel 3351314
```

Explore the ***Script_Output_*.0001*** file and ***run.card*** in the submit directory.
Explore the output directories.

Question 2a Which files are produced and where are they stored ? You did not find any files in the archive directory at `$STORE` (Jean Zay) or `$CCCSTOREDIR` (Irene)? Why not?
Help with this documentation to answer this question
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Running#Theoutputfiles

If needed, how to clean up and relaunch

If an error occurred and you need to relaunch the whole experiment, **you need to erase all output created during previous submission**, stored in the different `IGCM_OUT/LMDZ/JobName` directories:

- IDRIS: *`$WORK`*, `$SCRATCH` and `$STORE`,
- TGCC: `$CCCSTOREDIR`, `$CCCSCRATCHDIR` and `$CCCWORKDIR`,
- Obelix: within `/home/scratch01/login`.

In the submit directory you also have to remove `run.card`.

To ease the cleaning, the script `clean_PeriodLength.job` in ***`libIGCM`*** can be used. This script will clean up everything related to the last period that failed. Note that this script does not work if the `run.card` is missing or if you have `PeriodState=Completed` in `run.card`.
Note also that this script does not work on the `CREATE_clim_360d` experiment because this simulation saves output files on a specific format using suffix *`_clim`* instead of *`_${PeriodDateEnd}`*.

To use this script, stay in the submit directory `modipsl/config/LMDZOR_v6/MyJobTest`:

```
../../../libIGCM/clean_PeriodLenght.job  # Read questions and
answer yes to erase files.
```

2.3 Continue the simulation 4 more days

Now you want to continue your simulation for more days. For this you need to change in `config.card` the `DateEnd`.
NB: Do not change `DateBegin`.

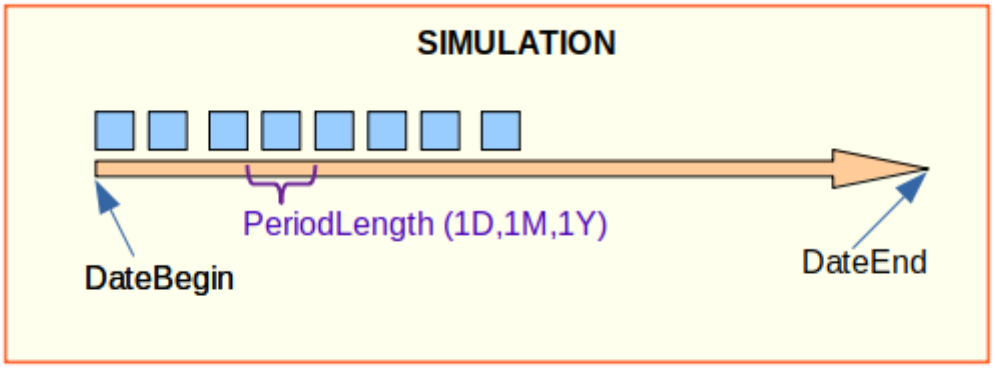
You also need to indicate to `run.card` that you will re-launch the simulation by changing `PeriodState=Completed` into `PeriodState=OnQueue`.

Do this for 4 more days:

```
vi config.card # →  Modify DateEnd
vi run.card # → modify PeriodState
sbatch Job_MyJobTest / ccc_msub Job_MyJobTest / qsub Job_MyJobTest
```

Question 2b How many times did the job go into the queue?

Your simulation will be submitted 4 times, because it's a succession of 4 simulations of 1 day. At the end of each **period** the simulation is submitted one more time to launch the next **period.**



To avoid all these submissions, you will modify the parameter **`PeriodNb`** in the main Job. PeriodNb will be the number of Period that can be launch in the CPUtime.

Question 2c : create a new simulation of 5 days, always with `PeriodLength=1D`, but with a different `PeriodNb` parameter to launch the job only one time on queue.

Question 2d Look in your first simulation run.card. How long did one day take? Did all days take the same time?

Once done the test simulation, we need to be sure that we have all the wanted output files and that they store all the variables required to analyse to simulation.

To know where are stored your output files you can read this page https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Running#Theoutputfiles

Question 2e Look in your scratchdir to check all directories and files created by your simulation.

In the next exercise you will done a simulation in PROD mode. Like this you will see the difference between these two modes.

2.4 Create another simulation with pack

This exercise can not be done on obelix because the pack function is not activated on obelix.

Create a new experience of type `LMDZOR/clim_360d`. This time we will also activate the archiving Pack functionality. The pack is activated when ***SpaceName=PROD*** or ***DEVT***. In this example, put ***SpaceName=DEVT***.

To test the pack functionality, set `PackFrequency=2M` in this exercise.
Launch 4 months with 1 month period length.

```
cp EXPERIMENTS/LMDZOR/clim_360d/config.card .

vi config.card
  # Modify : JobName
  # Modify : DateEnd=1980-04-30
  # Modify : PeriodLength=1M
   # Modify number of OMP threads if you are running on Obelix or JeanZay
(as before)
  # Activate pack : SpaceName=DEVT, PackFrequency=2M
  # Desactivate TimeSeries and Seasonnal average as before

../../libIGCM/ins_job

cd MyJobTest3

vi Job_MyJobTest3
  # for information : one month on JeanZay take between 550 and 650s CPU
Time. Define the CPU Time and the queue in function of this.

  { For training day on cfor accounts don't forget to define ARCHIVE,
RUN_DIR_PATH, R_OUT, and R_BUF }

sbatch Job_MyJob / ccc_msub Job_MyJob
```

Continue with next exercises while this job is running.

Check how it is proceeding in the queue every now and then.

Question : explore output directories, can you understand what was done ?
Read this page to check what you understood correctly and what it's really done
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Running#ConcatenationofPACKoutputs

2.5 Use differents forcing files

Forcings Files are divided in two categories : Initial States Files and Boundary files. There are defined in ***COMP/model.card*** (`COMP/lmdz.card`, `COMP/orchidee.card` etc.) files.

Initial State Files : these files give information on the state (atmospheric concentrations, temperatures etc.) of your domain at the beginning of the simulation. To start a new simulation you can choose to use default file given by modeles, or to start from the state of a previous simulation, or use the atmosphere state from one, and surface from another…
Read this documentation to learn how you can do these 3 choices
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Setup#Setupinitialstateforthesimulation

Boundary Files : There are two kinds of boundaries files, those depending on time and those that will not change during the whole simulation.
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Setup#TheBoundaryFilessection

Exercise :
Do a new simulation of 5 days using as an initial state file the restart created at the end of your simulation *MyJobTest*.
NB: It's not a problem if the date of the restart is not the date previous the beginning of your simulation. By coherence it's better, but it's not mandatory

```
vi config.card

#D-- Restarts -
[Restarts]
OverRule=y
#D- Last day of the experience used as restart for all components
RestartDate=1980-01-05
#D- Define restart simulation name for all components
RestartJobName=MyJobTest
#D- Path Server Group Login
RestartPath=$SCRATCH/IGCM_OUT/LMDZOR/TEST/clim
```

Question : which files start.nc, startphy.nc, sechiba_rest_in.nc are used ?

Exercise : modify `COMP/orchidee.card` to use the PFTmap of the current year of simulation, by using the variable `${year}`

Question : Verify in Script_output file you use the file you want.

2.6 CREATE_clim and CREATE_amip : Experiments to create initial state files and boundary conditions for LMDZ

`EXPERIMENT/LMDZ/CREATE_clim_360d` and `EXPERIMENTS/LMDZ/CREATE_amip` are two experiments set-up that launch the program ***`create_etat0_limit.e`***, a program based on LMDZ. This program is used to create initial state files (***`start.nc`*** and `startphy.nc`) and boundary condition files (`limit.nc`, `climoz_LMDZ.nc`) needed by LMDZ. The normal use of the LMDZOR_v6 configuration is to first run the experiment CREATE_clim_360d or CREATE_amip and then the experiment clim or amip. The CREATE_clim_360d/_amip experiment needs to be done only one time per resolution.
For use of the default resolution it is also possible to do as in exercise 2.2 and change to copy files from IGCM shared repository.

You will now launch the CREATE_clim_360d experiment. Note that for a standard use of CREATE_clim_360d you don't need to change anything. CREATE_clim_360d is set up for a 360 days/year calendar and CREATE_amip is set up for a noleap calendar (always 365 days/year). The same thing applies for experiments clim_360d (360 days/year) and amip (365days/year).

Now install the submit directory for CREATE_clim:

```
cd modipsl/config/LMDZOR_v6
cp EXPERIMENTS/LMDZ/CREATE_clim_3660d/config.card .

../../libIGCM/ins_job

cd ELC-144x142x79
```

The directory ELC-144x142x79 was created and the config.card was moved inside. The resolution in the JobName was taken from the ***`.resol`*** file created during compilation.

This experiment will launch the executable ***`create_etat0_limit.e`***. It is possible to use a test class because the run will not take more than a few minutes. You can set the test class in the beginning of the Job_ELC-144x142x79.

Submit the job as before:

```
sbatch Job_ELC-144x142x79  ccc_msub Job_ELC-144x142x79 /
    qsub Job_ELC-144x142x79
```

Output files are found in the directory ***IGCM_OUT/LMDZ/ELC-144x142x79*** on the $STORE at IDRIS, in $CCCSTOREDIR at TGCC or at /home/scratch01/login at obelix.

Explore the script output text file in the submit directory and the files in the output directory ELC-144x142x79.

Question 2e Where can you find the output? Which files are produced and where are they stored?

Question 2f What type of calendar is used? How many days contains a year? Check also the number of time step in the output file limit.nc. Do you know how you can change the calendar that has been used?

Question 2g Now create a new experiment clim_360d using boundaries files created by `ELC-144x142x79`. For this in COMP/lmdz.card you will modify the path for start.nc, startphy.nc, limit.nc and climoz_LMDZ.nc files.

2.7 Summary on how to extract, compile and launch a simulation

1. **Download modipsl**

```
mkdir $WORK/MYFIRSTTEST ; cd $WORK/MYFIRSTTEST
svn co http://forge.ipsl.jussieu.fr/igcmg/svn/modipsl/trunk modipsl
```

2. **Extract a configuration** (ex: LMDZOR_v6)

```
cd $WORK/MYFIRSTTEST/modipsl/util
./model LMDZOR_v6.1.10
```

3. **Compil**

```
cd $WORK/MYFIRSTTEST/modipsl/config/LMDZOR_v6
gmake LMD144142-L79
```

4. **Create experiment directory**

```
cd $WORK/MYFIRSTTEST/modipsl/config/LMDZOR_v6

cp EXPERIMENTS/LMDZOR/clim_360d/config.card .

vi config.card   ### Modify at least JobName=MyJobTest & // options

../../libIGCM/ins_job   # At JeanZay enter your project ID
                          # At Irene enter your project ID and default
                          answer for other questions
```

5. **Launch simulation**

```
cd $WORK/MYFIRSTTEST/modipsl/config/LMDZOR_v6/MyJobTest/

sbatch Job_MyJobTest / ccc_msub Job_MyJobTest /
 qsub Job_MyJobTest
```

3. Debug

We will now work on three small exercises for debugging. For these exercises we will use files prepared and stored :

- At irene, TGCC:
$CCCWORKDIR/../../igcmg/igcmg/TRAINING/MODIPSL_HandsOn_20200114/LMDZOR_v6

- At jean-zay, IDRIS :
$WORK/../../../rech/psl/commun/TRAINING/MODIPSL_HandsOn_20200114/LMDZOR_v6

- At obelix:
/home/orchideeshare/igcmg/TRAINING/MODIPSL_HandsOn_20200114

3.0 How can you analyze the Job Output : Script_Output ?

If your simulation has a problem the first thing to do is to read and analyse the file Script_Output. It will give you first important information on your simulation.

https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/CheckDebug#AnalyzingtheJoboutput:Script_Output

3.1 Debug : setup error

Copy the file lmdz.card_1 from the directory above into the lmdz.card file in the COMP/ in your submit directory. You can choose one of the submit directory from the previous exercises or create a new one.

Now launch the simulation and debug it. Don't forget to clean up as done in exercise 2 before re-launching the simulation. Use clean_PeriodLength.job to do this.

Question 3b What was the error?

Copy the lmdz.card_2 and debug again. **Question 3c** What was the error?
Copy the lmdz.card_3 and debug again. **Question 3d** What was the error? If you don’t find the solution you can try to find the difference between your actual lmdz.card file and the last one that was working.

3.2 Debug : error during the simulation

If you add a “print” directive in a model you can check during the simulation the output in the temporary directory RUN_DIR/.
Try to add a “print” in LMDZ or ORCHIDEE model

```
cd modipsl/modeles/LMDZ/libf/phylm/
vi physiq_mod.90
Look for line
   IF (iflag_pbl/=0) THEN
And add just before
write(lunout,*) ‘debug LMDZ - iflag_pbl = ’, iflag_pbl

OR

cd modipsl/modeles/ORCHIDEE/src_sechiba
vi sechiba.F90
Look for line
   IF ( river_routing .AND. nbp_glo .GT. 1) THEN
And add just before
WRITE (numout,*) ‘debug ORCHIDEE - river_routing = ’, river_routing
```

Note : The unit use by the `WRITE` instruction will be different from one model to another one.

Re-compile your models and launch a test of 1 month. Now don’t wait the of the simulation, check your simulation id and go on the RUN_DIR directory (on the scratchdir),

```
cd $SCRATCH/RUN_DIR/Id_job/****/   (JeanZay)
cd $CCCSCRATCHDIR/RUN_DIR/Id_job/****/   (Irène)
ls
```

To look values of your previous print you need to open for LMDZ files `out_lmdz.e.out_***`, or for ORCHIDEE files `out_orchidee_****`.
In each case you can notice that there are several output files, there is one by OMP threads (if you are running a parallel simulation). In each of them you will find the output text print for this specific threads or proc.

If you have a problem during a simulation, you can try to debug by adding print in yours models.

3.3 Compilation in debug mode

Can not be done on training day because we cannot compile (du to quota problems)

If you don't have any clue to solve your bug you can try to compile the model in "debug", for this open the main Makefile and replace "prod" by "debug" every where you find it (one by compilation line). After you need to recompile and launch one more time your simulation. Like this you will have more information on your bug and on the moment your simulation crash.
For this exercise we will create 2 bugs in LDMZ model :

1. A buffer overflow
2. A division by zero

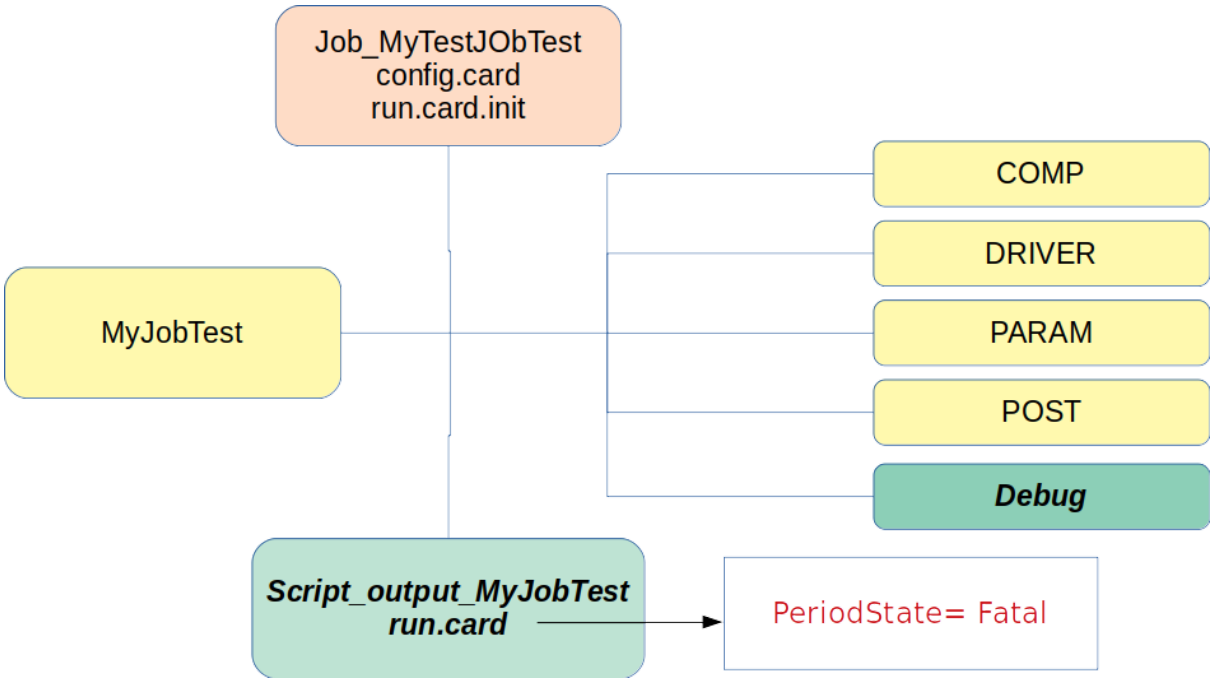
For this, copy the file `TRAINING/MODIPSL_HandsOn_20200114/LMDZOR_v6/physiq_mod.F90` in your model

```
cd modipsl/modeles/LMDZ/libf/phylmd
cp path/TRAINING/MODIPSL_HandsOn_20200114/LMDZOR_v6/physiq_mod.F90  .
```

Now recompile your code and launch a simulation of 1 day. Check the run.card file at the end of this simulation. You can notice that the run bug.

```
# State of Job "Start", "Running", "OnQueue", "Completed"
PeriodState= Fatal
```

The simulation create a new directory call `Debug` in your experiment directory.



You can read the description of this directory here
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/CheckDebug#TheDebugdirectory

In our case we know that the modification was made in LMDZ, so we will start our investigation in its debug file. For this, open the file `Debug/***_out_lmdz.x.err`.
In this file you will find information for each proc mpi. You can read that there is a bug but there is no more information on the localisation of this bug. It's because to compile we use permissives options that will not track precisely bug.

```
0forrtl: severe (174): SIGSEGV, segmentation fault occurred
0forrtl: severe (174): SIGSEGV, segmentation fault occurred
0forrtl: severe (174): SIGSEGV, segmentation fault occurred
0forrtl: severe (174): SIGSEGV, segmentation fault occurred
0forrtl: severe (174): SIGSEGV, segmentation fault occurred
```

To obtain more clues on this bug we need to recompile in debug. For this you will replace all the words "prod" by "debug" in the main Makefile, and relaunch the compilation

```
cd modipsl/config/LMDZOR_v6
vi Makefile
→ look for "prod" and replace by "debug"
```

```
gmake
```

Create a new simulation. This new simulation will crash again, but now you will find more information in the file `Debug/***_out_lmdz.x.err`.

```
9forrtl: severe (174): SIGSEGV, segmentation fault occurred
9Image          PC                Routine            Line     Source
9lmdz.x         0000000005146D79  Unknown               Unknown  Unknown
9libpthread-2.17.s  00002AAAB10C45D0  Unknown          Unknown  Unknown
9lmdz.x         0000000000AA8C70  physiq_mod_mp_phy       1619  physiq_mod.f90
9lmdz.x         00000000009872B2  callphysiq_mod_mp       81
callphysiq_mod.f90
9lmdz.x         0000000000979F74  calfis_loc_             729  calfis_loc.f
9lmdz.x         0000000000687DAA  call_calfis_mod_m       214
call_calfis_mod.f90
9lmdz.x         00000000004AF7AE  leapfrog_loc_           807  leapfrog_loc.f
9lmdz.x         0000000000427DCA  MAIN__                  454  gcm.f90
9libiomp5.so            00002AAAB3DE0ED3  __kmp_invoke_micr Unknown  Unknown
9libiomp5.so            00002AAAB3DA3726  Unknown           Unknown  Unknown
9libiomp5.so            00002AAAB3DA50FD  __kmp_fork_call   Unknown  Unknown
9libiomp5.so            00002AAAB3D66020  __kmpc_fork_call  Unknown  Unknown
9lmdz.x         0000000000424A19  MAIN__                  445  gcm.f90
9lmdz.x         000000000041EC62  Unknown               Unknown  Unknown
9libc-2.17.so   00002AAAB4105495  __libc_start_main     Unknown  Unknown
9lmdz.x         000000000041EB69  Unknown               Unknown  Unknown
```

To find these lines, you can read all the files or look for the key word “gcm” (the name of LMDZ main program).
It’s telling you that there is a problem at line 1619 on physiq_mod.f90, call by callphysiq_mod.f90 at line 81, call by calfis_loc.f at line 729, call by call_calfis_mod.f90 at line 214, call by leapfroc_loc.f at line 807, call by gcm.f90 at line 454.
Warning : all lines numbers don’t refer to the code sources, but to pre-compile sources

```
In LMDZ : modeles/LMDZ/tmp_src
In ORCHIDEE: modeles/ORCHIDEE/build/ppsrc/
In INCA : modeles/INCA/build/ppsrc/
In NEMO/PISCES :
modeles/NEMOGCM/CONFIG/ORCA1_LIM3_PISCES/BLD/ppsrc
```

Open the file `modeles/LMDZ/tmp_src/phys/physiq_mod.f90` and look for the line indicated by your debug file `out_lmdz.x.err`.

Question 3e Try to understand the problem on this specific line.

Now we will create a division by zero

```
In LMDZ/libf/phylmd/physiq_mod.F90 modify
 temporary = rugoro(99999999)

By
 temporary = 0.
```

Question 3f Compile a new time. And make the analyze of the `Debug/***out_lmdz.x.err`.

4. Create time series

4.1 Launch 10 years with default time series

This exercise is done to understand how to control the creation of time series. It is also an opportunity to test the supervisor.. We will use here an ORCHIDEE offline configuration with a small horizontal domaine just to have a model that runs quickly. The principle is the same for all configurations.

Install a new modipsl, download the configuration ORCHIDEE trunk and compile.

```
mkdir MYPOSTTEST; cd MYPOSTTEST
svn    co    http://forge.ipsl.jussieu.fr/igcmg/svn/modipsl/trunk
modipsl
cd modipsl/util
./model ORCHIDEE_trunk
cd ../config/ORCHIDEE_OL
gmake   #WARNING for the training day do not compile - copy in
modipsl/bin  executables stored in
$WORK/../../../rech/psl/commun/TRAINING/ORCHIDEE_20200116/modipsl/
bin
```

In this configuration it is not needed to create the experiment directory. Instead different experiment directories already exist : OOL_SEC_STO_FG**, OOL_SEC, FORCESOIL and SPINUP_ANALYTIC_FG1 are experiences that follow the standard rules described in this tutorial.

The DRIVER directory do not exist but “drivers" are found in the COMP directory.

SPINUP and ENSEMBLE are experiments that are more complicated and are not taught in the course. We will here work with the OOL_SEC_STO_FG2 experiment which is a full ORCHIDEE offline setup with sechiba and stomate components.

Copy the OOL_SEC_STO_FG2 directory into a new one, modify config.card and create the job.

For Orchidee offline configurations it is best to run with PeriodLength=1Y. Use a regional domain by setting LIMIT parameters in run.def. Because we change to a smaller domain, no

need to run on many processors. For this case, change to 3MPI for orchidee_ol in config.card.

```
cp -r OOL_SEC_STO_FG2 MyPostExp
cd MyPostExp

vi config.card # => Change JobName, SpaceName=DEVT
               # PeriodLength=1Y, DateEnd=1910-12-31,
               # PackFrequency=5Y, TimeSeriesFrequency=5Y,
               # SeasonalFrequency=5Y
                # OOL= (orchidee_ol, orchidee_ol, 3MPI)
               # IOS= (xios_server.exe, xios.x, 1MPI)

vi PARAM/run.def
# Add these lines
LIMIT_WEST = -10.
LIMIT_EAST = 20.
LIMIT_NORTH = 30.
LIMIT_SOUTH = 0.

../../../libIGCM/ins_job
```

Set `PeriodNb=10` in the main job and submit using `sbatch`, `ccc_msub` or `qsub` depending on the platform.

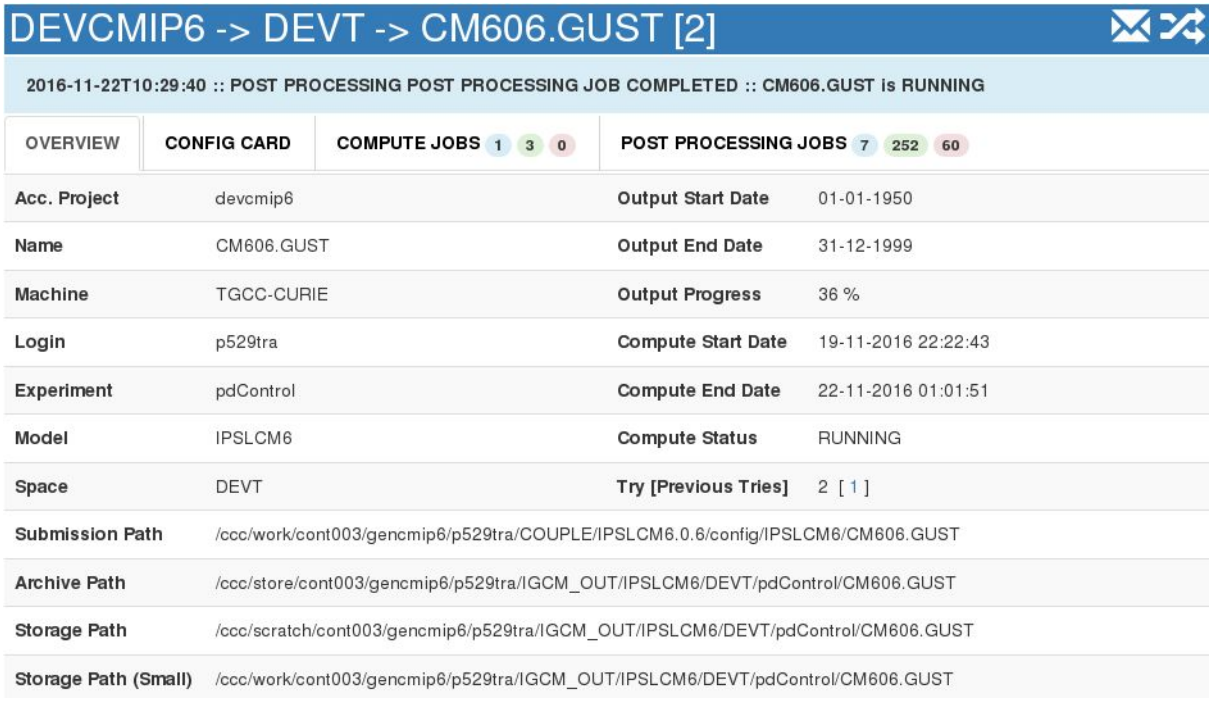
4.2 Use supervisor during run time

After about 40 min the simulation and the post processing are expected to be finished. (if it's still not done, go back to the question of part 2.5 during this simulation).

Check the supervision web interface to follow the rebuild, pack and time series status.

Go to the following web interface to see how the simulation is going. Try to understand what kind of information are gathered at that page : https://hermes.ipsl.upmc.fr

Find the summary of a simulation of interest.



DEVCMIP6 -> DEVT -> CM606.GUST [2]

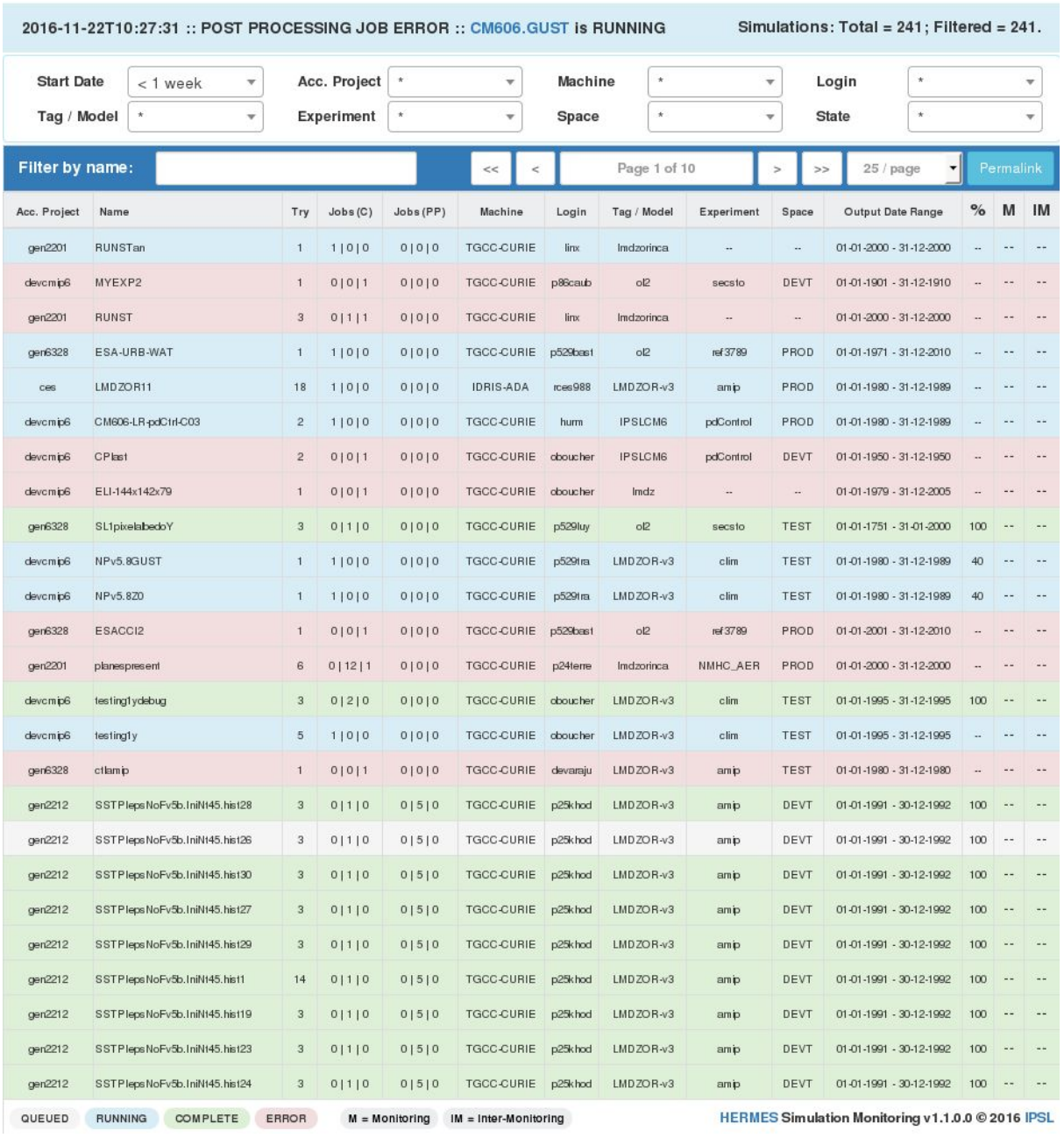
2016-11-22T10:29:40 :: POST PROCESSING POST PROCESSING JOB COMPLETED :: CM606.GUST is RUNNING

OVERVIEW | CONFIG CARD | COMPUTE JOBS 1 3 0 | POST PROCESSING JOBS 7 252 60

Acc. Project	devcmip6	Output Start Date	01-01-1950
Name	CM606.GUST	Output End Date	31-12-1999
Machine	TGCC-CURIE	Output Progress	36 %
Login	p529tra	Compute Start Date	19-11-2016 22:22:43
Experiment	pdControl	Compute End Date	22-11-2016 01:01:51
Model	IPSLCM6	Compute Status	RUNNING
Space	DEVT	Try [Previous Tries]	2 [1]
Submission Path	/ccc/work/cont003/gencmip6/p529tra/COUPLE/IPSLCM6.0.6/config/IPSLCM6/CM606.GUST		
Archive Path	/ccc/store/cont003/gencmip6/p529tra/IGCM_OUT/IPSLCM6/DEVT/pdControl/CM606.GUST		
Storage Path	/ccc/scratch/cont003/gencmip6/p529tra/IGCM_OUT/IPSLCM6/DEVT/pdControl/CM606.GUST		
Storage Path (Small)	/ccc/work/cont003/gencmip6/p529tra/IGCM_OUT/IPSLCM6/DEVT/pdControl/CM606.GUST		

Try the different search options.

Explore links and displays that are available on the page. What information are they trying to communicate?



2016-11-22T10:27:31 :: POST PROCESSING JOB ERROR :: CM606.GUST is RUNNING — Simulations: Total = 241; Filtered = 241.

Start Date: < 1 week | Acc. Project: * | Machine: * | Login: *
Tag / Model: * | Experiment: * | Space: * | State: *

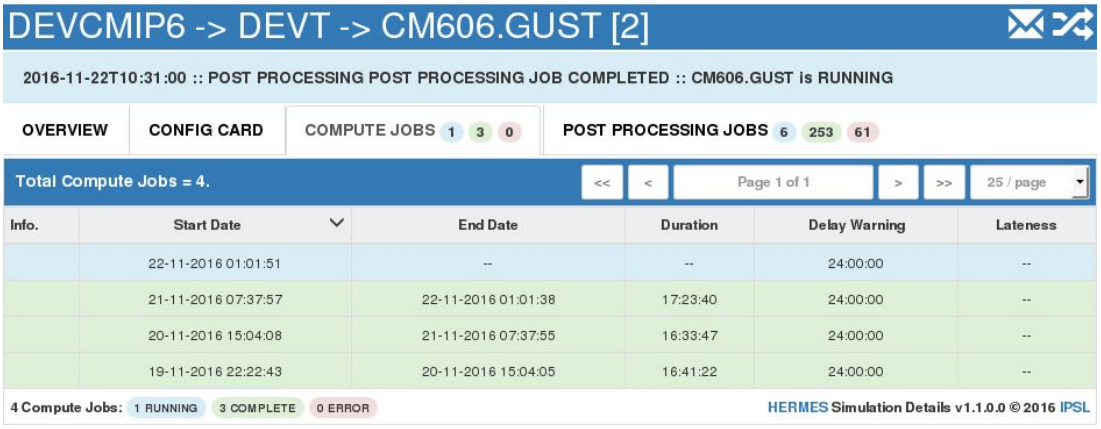
Filter by name: | << | < | Page 1 of 10 | > | >> | 25 / page | Permalink

Acc. Project	Name	Try	Jobs (C)	Jobs (PP)	Machine	Login	Tag / Model	Experiment	Space	Output Date Range	%	M	IM
gen2201	RUNSTan	1	1 | 0 | 0	0 | 0 | 0	TGCC-CURIE	linx	lmdzorinca	--	--	01-01-2000 - 31-12-2000	--	--	--
devcmip6	MYEXP2	1	0 | 0 | 1	0 | 0 | 0	TGCC-CURIE	p86caub	ol2	secsto	DEVT	01-01-1901 - 31-12-1910	--	--	--
gen2201	RUNST	3	0 | 1 | 1	0 | 0 | 0	TGCC-CURIE	linx	lmdzorinca	--	--	01-01-2000 - 31-12-2000	--	--	--
gen6328	ESA-URB-WAT	1	1 | 0 | 0	0 | 0 | 0	TGCC-CURIE	p529baat	ol2	ref3789	PROD	01-01-1971 - 31-12-2010	--	--	--
ces	LMDZOR11	18	1 | 0 | 0	0 | 0 | 0	IDRIS-ADA	rces988	LMDZOR-v3	amip	PROD	01-01-1980 - 31-12-1989	--	--	--
devcmip6	CM606-LR-pdCtrl-C03	2	1 | 0 | 0	0 | 0 | 0	TGCC-CURIE	hurm	IPSLCM6	pdControl	PROD	01-01-1980 - 31-12-1989	--	--	--
devcmip6	CPlast	2	0 | 0 | 1	0 | 0 | 0	TGCC-CURIE	oboucher	IPSLCM6	pdControl	DEVT	01-01-1950 - 31-12-1950	--	--	--
devcmip6	ELI-144x142x79	1	0 | 0 | 1	0 | 0 | 0	TGCC-CURIE	oboucher	lmdz	--	--	01-01-1979 - 31-12-2005	--	--	--
gen6328	SL1pixelabedoY	3	0 | 1 | 0	0 | 0 | 0	TGCC-CURIE	p529luy	ol2	secsto	TEST	01-01-1751 - 31-01-2000	100	--	--
devcmip6	NPv5.8GUST	1	1 | 0 | 0	0 | 0 | 0	TGCC-CURIE	p529tra	LMDZOR-v3	clim	TEST	01-01-1980 - 31-12-1989	40	--	--
devcmip6	NPv5.8Z0	1	1 | 0 | 0	0 | 0 | 0	TGCC-CURIE	p529tra	LMDZOR-v3	clim	TEST	01-01-1980 - 31-12-1989	40	--	--
gen6328	ESACCI2	1	0 | 0 | 1	0 | 0 | 0	TGCC-CURIE	p529baat	ol2	ref3789	PROD	01-01-2001 - 31-12-2010	--	--	--
gen2201	planespresent	6	0 | 12 | 1	0 | 0 | 0	TGCC-CURIE	p24terre	lmdzorinca	NMHC_AER	PROD	01-01-2000 - 31-12-2000	--	--	--
devcmip6	testing1ydebug	3	0 | 2 | 0	0 | 0 | 0	TGCC-CURIE	oboucher	LMDZOR-v3	clim	TEST	01-01-1995 - 31-12-1995	100	--	--
devcmip6	testing1y	5	1 | 0 | 0	0 | 0 | 0	TGCC-CURIE	oboucher	LMDZOR-v3	clim	TEST	01-01-1995 - 31-12-1995	--	--	--
gen6328	ctlamip	1	0 | 0 | 1	0 | 0 | 0	TGCC-CURIE	devaraju	LMDZOR-v3	amip	TEST	01-01-1980 - 31-12-1980	--	--	--
gen2212	SSTPlepsNoFv5b.IniNt45.hist28	3	0 | 1 | 0	0 | 5 | 0	TGCC-CURIE	p25khod	LMDZOR-v3	amip	DEVT	01-01-1991 - 30-12-1992	100	--	--
gen2212	SSTPlepsNoFv5b.IniNt45.hist26	3	0 | 1 | 0	0 | 5 | 0	TGCC-CURIE	p25khod	LMDZOR-v3	amip	DEVT	01-01-1991 - 30-12-1992	100	--	--
gen2212	SSTPlepsNoFv5b.IniNt45.hist30	3	0 | 1 | 0	0 | 5 | 0	TGCC-CURIE	p25khod	LMDZOR-v3	amip	DEVT	01-01-1991 - 30-12-1992	100	--	--
gen2212	SSTPlepsNoFv5b.IniNt45.hist27	3	0 | 1 | 0	0 | 5 | 0	TGCC-CURIE	p25khod	LMDZOR-v3	amip	DEVT	01-01-1991 - 30-12-1992	100	--	--
gen2212	SSTPlepsNoFv5b.IniNt45.hist29	3	0 | 1 | 0	0 | 5 | 0	TGCC-CURIE	p25khod	LMDZOR-v3	amip	DEVT	01-01-1991 - 30-12-1992	100	--	--
gen2212	SSTPlepsNoFv5b.IniNt45.hist1	14	0 | 1 | 0	0 | 5 | 0	TGCC-CURIE	p25khod	LMDZOR-v3	amip	DEVT	01-01-1991 - 30-12-1992	100	--	--
gen2212	SSTPlepsNoFv5b.IniNt45.hist19	3	0 | 1 | 0	0 | 5 | 0	TGCC-CURIE	p25khod	LMDZOR-v3	amip	DEVT	01-01-1991 - 30-12-1992	100	--	--
gen2212	SSTPlepsNoFv5b.IniNt45.hist23	3	0 | 1 | 0	0 | 5 | 0	TGCC-CURIE	p25khod	LMDZOR-v3	amip	DEVT	01-01-1991 - 30-12-1992	100	--	--
gen2212	SSTPlepsNoFv5b.IniNt45.hist24	3	0 | 1 | 0	0 | 5 | 0	TGCC-CURIE	p25khod	LMDZOR-v3	amip	DEVT	01-01-1991 - 30-12-1992	100	--	--

QUEUED | RUNNING | COMPLETE | ERROR | M = Monitoring | IM = Inter-Monitoring

HERMES Simulation Monitoring v1.1.0.0 © 2016 IPSL

What is the status of your compute job? Find the link to the graphical monitoring.



DEVCMIP6 -> DEVT -> CM606.GUST [2]

2016-11-22T10:31:00 :: POST PROCESSING POST PROCESSING JOB COMPLETED :: CM606.GUST is RUNNING

OVERVIEW | CONFIG CARD | COMPUTE JOBS 1 3 0 | POST PROCESSING JOBS 6 253 61

Total Compute Jobs = 4. | << | < | Page 1 of 1 | > | >> | 25 / page

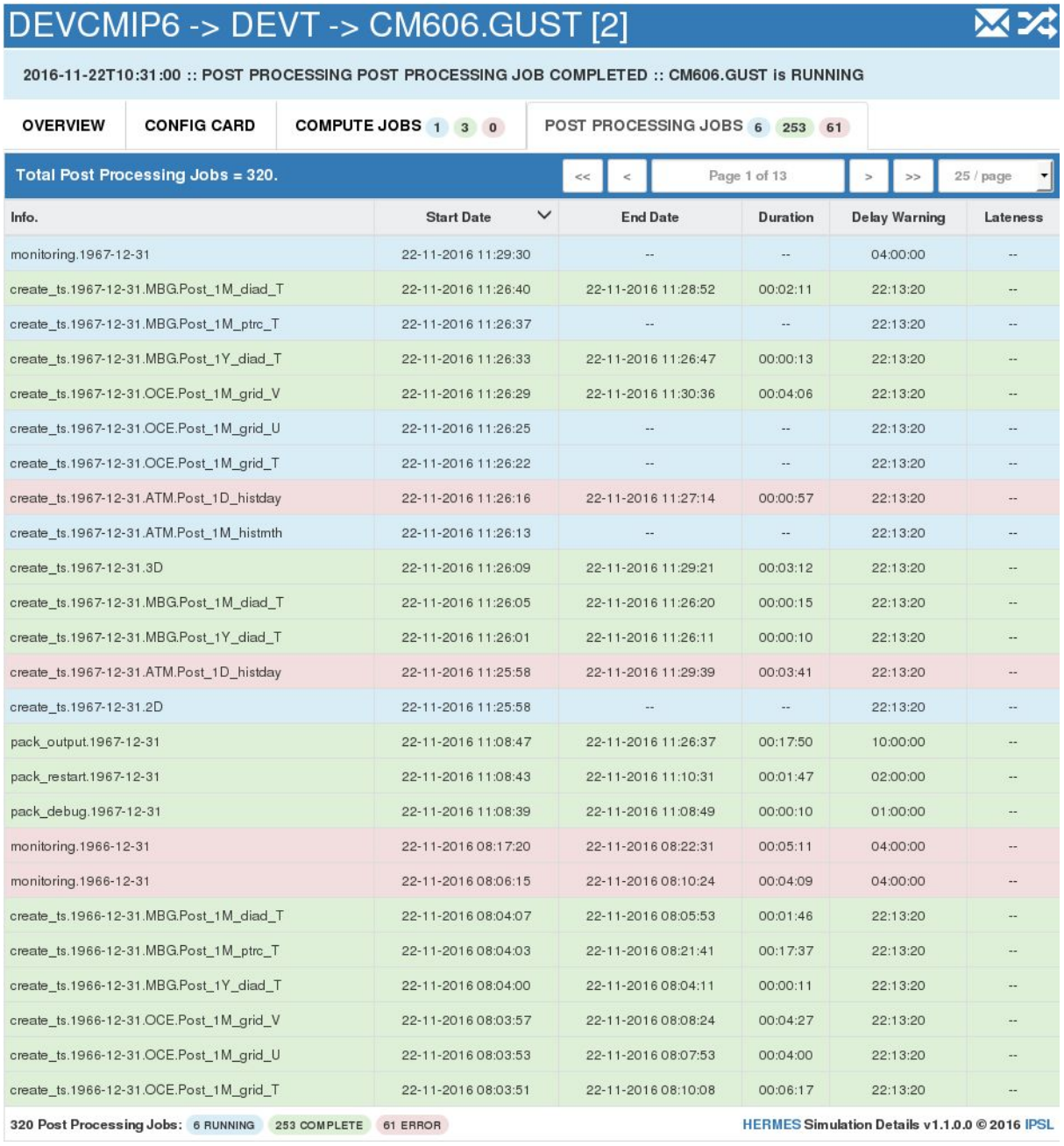
Info.	Start Date	End Date	Duration	Delay Warning	Lateness
	22-11-2016 01:01:51	--	--	24:00:00	--
	21-11-2016 07:37:57	22-11-2016 01:01:38	17:23:40	24:00:00	--
	20-11-2016 15:04:08	21-11-2016 07:37:55	16:33:47	24:00:00	--
	19-11-2016 22:22:43	20-11-2016 15:04:05	16:41:22	24:00:00	--

4 Compute Jobs: 1 RUNNING | 3 COMPLETE | 0 ERROR

HERMES Simulation Details v1.1.0.0 © 2016 IPSL

What is the status of your post-processing job? In case of error find the root cause of it.

Find the list of messages that has been sent by libIGCM.



DEVCMIP6 -> DEVT -> CM606.GUST [2]

2016-11-22T10:31:00 :: POST PROCESSING POST PROCESSING JOB COMPLETED :: CM606.GUST is RUNNING

OVERVIEW | CONFIG CARD | COMPUTE JOBS 1 3 0 | POST PROCESSING JOBS 6 253 61

Total Post Processing Jobs = 320. << < Page 1 of 13 > >> 25 / page

Info.	Start Date	End Date	Duration	Delay Warning	Lateness
monitoring.1967-12-31	22-11-2016 11:29:30	--	--	04:00:00	--
create_ts.1967-12-31.MBG.Post_1M_diad_T	22-11-2016 11:26:40	22-11-2016 11:28:52	00:02:11	22:13:20	--
create_ts.1967-12-31.MBG.Post_1M_ptrc_T	22-11-2016 11:26:37	--	--	22:13:20	--
create_ts.1967-12-31.MBG.Post_1Y_diad_T	22-11-2016 11:26:33	22-11-2016 11:26:47	00:00:13	22:13:20	--
create_ts.1967-12-31.OCE.Post_1M_grid_V	22-11-2016 11:26:29	22-11-2016 11:30:36	00:04:06	22:13:20	--
create_ts.1967-12-31.OCE.Post_1M_grid_U	22-11-2016 11:26:25	--	--	22:13:20	--
create_ts.1967-12-31.OCE.Post_1M_grid_T	22-11-2016 11:26:22	--	--	22:13:20	--
create_ts.1967-12-31.ATM.Post_1D_histday	22-11-2016 11:26:16	22-11-2016 11:27:14	00:00:57	22:13:20	--
create_ts.1967-12-31.ATM.Post_1M_histmth	22-11-2016 11:26:13	--	--	22:13:20	--
create_ts.1967-12-31.3D	22-11-2016 11:26:09	22-11-2016 11:29:21	00:03:12	22:13:20	--
create_ts.1967-12-31.MBG.Post_1M_diad_T	22-11-2016 11:26:05	22-11-2016 11:26:20	00:00:15	22:13:20	--
create_ts.1967-12-31.MBG.Post_1Y_diad_T	22-11-2016 11:26:01	22-11-2016 11:26:11	00:00:10	22:13:20	--
create_ts.1967-12-31.ATM.Post_1D_histday	22-11-2016 11:25:58	22-11-2016 11:29:39	00:03:41	22:13:20	--
create_ts.1967-12-31.2D	22-11-2016 11:25:58	--	--	22:13:20	--
pack_output.1967-12-31	22-11-2016 11:08:47	22-11-2016 11:26:37	00:17:50	10:00:00	--
pack_restart.1967-12-31	22-11-2016 11:08:43	22-11-2016 11:10:31	00:01:47	02:00:00	--
pack_debug.1967-12-31	22-11-2016 11:08:39	22-11-2016 11:08:49	00:00:10	01:00:00	--
monitoring.1966-12-31	22-11-2016 08:17:20	22-11-2016 08:22:31	00:05:11	04:00:00	--
monitoring.1966-12-31	22-11-2016 08:06:15	22-11-2016 08:10:24	00:04:09	04:00:00	--
create_ts.1966-12-31.MBG.Post_1M_diad_T	22-11-2016 08:04:07	22-11-2016 08:05:53	00:01:46	22:13:20	--
create_ts.1966-12-31.MBG.Post_1M_ptrc_T	22-11-2016 08:04:03	22-11-2016 08:21:41	00:17:37	22:13:20	--
create_ts.1966-12-31.MBG.Post_1Y_diad_T	22-11-2016 08:04:00	22-11-2016 08:04:11	00:00:11	22:13:20	--
create_ts.1966-12-31.OCE.Post_1M_grid_V	22-11-2016 08:03:57	22-11-2016 08:08:24	00:04:27	22:13:20	--
create_ts.1966-12-31.OCE.Post_1M_grid_U	22-11-2016 08:03:53	22-11-2016 08:07:53	00:04:00	22:13:20	--
create_ts.1966-12-31.OCE.Post_1M_grid_T	22-11-2016 08:03:51	22-11-2016 08:10:08	00:06:17	22:13:20	--

320 Post Processing Jobs: 6 RUNNING 253 COMPLETE 61 ERROR

HERMES Simulation Details v1.1.0.0 © 2016 IPSL

4.3 Add variables to time series and relaunch with the TimeSeriesChecker.job

All variables in the Output files can be used to create time series. A selection of variables are done by default.
Now add the creation of time series for the variables "`z0h`" and "`z0m`". First be sure that they are produced and exist in `sechiba_history.nc` file (in directory `IGCM_OUT/…./JobName/SRF/Output/MO/`). Then add in `sechiba.card`:

```
[Post_1M_sechiba_history]
Patches = ()
GatherWithInternal= (lon, lat, veget, time_counter, time_counter_bnds, Areas, Contfrac)
TimeSeriesVars2D = (riverflow, coastalflow, nobiofrac, ....
```

Find the documentation about the script `TimeSeries_Checker.job` and launch it to create missing and new time series.
For remind the web documentation is available here : https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc

Question : why when you launch the `TimeSeries_Checker.job` do you have a message like this one?

```
---------Debug3--> Missing time series from 1M_stomate_history in
/***/IGCM_OUT/OL2/DEVT/secsto/MyPostExp/SBG/Analyse/TS_MO :

----------Debug3--> MyPostExp_19010101_19101231_1M_AGE.nc
----------Debug3--> MyPostExp_19010101_19101231_1M_HEIGHT.nc
----------Debug3--> MyPostExp_19010101_19101231_1M_ADAPTATION.nc
----------Debug3--> MyPostExp_19010101_19101231_1M_REGENERATION.nc
----------Debug3--> MyPostExp_19010101_19101231_1M_CARBON_ACTIVE.nc
----------Debug3--> MyPostExp_19010101_19101231_1M_CARBON_SLOW.nc
(...)

-----Debug2--> 47% files OK. for period 19010101-19101231
```

Question : verify that Time Series for `z0h` and `z0m` was created.

5. Monitoring and Inter-monitoring

The monitoring is a web-interface tool that visualizes the global mean over time for a setup of key variables. Inter-monitoring web-interface allows to simultaneously monitor various simulations. More details can be found in:
http://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Running#Monitoringandintermonitoring

5.1 Monitoring

Visualize for example the monitoring on the web for **CM61-LR-pi-03 simulation** (IPSLCM6-CMIP6 piControl simulation performed on Curie-TGCC)

https://vesg.ipsl.upmc.fr/thredds/fileServer/work/p86maf/IPSLCM6/PROD/piControl/CM61-LR-pi-03/MONITORING/index.html

5.2 Inter-monitoring

5.2.1 from supervisor interface

It is possible to use supervisor interface to superpose simulations referenced in supervisor database.

Step 1 : https://hermes.ipsl.upmc.fr

Step 2 : Find different members of simulations CM61-LR-scen-ssp126 by using "Filter by name" and chosen " * " for StartDate

Step 3: Tick "IM" for both simulations and right click on IM (top of the column) select "Open Inter-Monitoring"

5.2.2 from web interface tool "webservices"

Now you will use the web interface tool "inter-monitoring" to superpose several simulations. The default inter-monitoring is found at address :
http://webservices2017.ipsl.fr/interMonitoring/.
For this exercise choose following 2 simulations : **CM61-LR-pi-03** (IPSLCM6-CMIP6 piControl simulation performed on Curie-TGCC) and **CM61-pi-valid.02.JZ** (IPSLCM6 piControl simulation performed on JeanZay-IDRIS). These simulations have been used to valide porting on JeanZay.
To do the inter-monitoring comparaison, set the corresponding paths :

- CM61-LR-pi-03:

http://vesg.ipsl.upmc.fr/thredds/catalog/work/p86maf/IPSLCM6/PROD/piControl

- CM61-pi-valid.02.JZ:

http://vesg.ipsl.upmc.fr/thredds/catalog/work/p86caub/IPSLCM6/DEVT/piControl

And follow the following Mini how to use the inter-monitoring :
Go to http://webservices2017.ipsl.fr/interMonitoring/

- **Step 1:** Enter the first path and click on the button List Directories.
- **Step 2:** You'll see a list of all simulations at this path. Go back to step 1.
- **Step 1 bis:** Go back to step 1, enter the second path ***and click on Append Directories.***
- **Step 2 bis:** You'll now see all simulations on the 2 paths. Choose the two simulations with the corresponding names. (use the mouse and type ctrl to select only 2 simulations). Click on Search files.
- **Step 3:** Select one variable and click on Validate.
- **Step 4:** Choose default setting for “plot01:Time series" and click on Validate. Then click on the button below called “Prepare and run the ferret script”.
- Now a ferret script will appear on the screen and one image. Click on the button “Run this script on the server” below on the page. The inter-monitoring for all variables will now appear on the screen.

Note : CM61-pi-valid.02.JZ simulation is shorter than CM61-LR-pi-03. Back to the Step 4 to select only the part 1850-1900 (using “Dates range” cursor) which is the common period between both simulations then click again on “Prepare and run the ferret script”.

6. Modify output using XIOS

6.1 Create a new output file for ORCHIDEE

The different output files and their contents in ORCHIDEE are defined in the file ***modeles/ORCHIDEE/src_xml/file_def_orchidee.xml***
This file can be modified to contain specific output if needed. The key words _AUTO_ can be changed directly in the file or using the variables in `orchidee.card`, `sechiba.card` and `stomate.card` (section `[UserChoices]`). To save a variable, the file must also be listed in `orchidee`/`sechiba`/`stomate.card` (section `[OutputFiles]`). The same method is used working coupled to LMDZ or using ORCHIDEE in offline mode. The only difference is the name of the comp.card: `orchidee.card` for coupled to LMDZ and `sechiba.card` when running in offline mode. For this exercise, use a test in offline mode because it is faster to run.

In this exercise you should create a new output file from ORCHIDEE containing only rain and snow fall on daily average. The variables are already output from the model using `xios_send_field` and they are declared in the `field_def_orchidee.xml` with the id `precip_rain` and `precip_snow`. If you want to see where in the model they are written, search for `precip_` and `xios` in `ORCHIDEE/src*/*` using

```
grep precip_ src_*/* | grep xios
```

in `modipsl/modeles/ORCHIDEE/` folder.

Set up the file with following specifications:

- The file should be named `myoutput_orch.nc`
- The name of the variables in the output file should be "`rainfall`" and "`snowfall`"
- Keep the default unit, mm/s
- File output frequency should be daily average. You have to set the file attribute ***output_freq="1d"***
- File attribute ***enabled=.TRUE.***

Do the following:

1. Continue in the same modipsl where you installed ORCHIDEE offline in exercise 4
2. Add a section in `file_def_orchidee.xml` with the specifications as above. Take example on how the first file `sechiba_history` is defined and do in similar way just below or above:

```
<file id="sechiba0" name="myoutput_orch" output_level="1" output_freq="1d"
enabled="true">

<field_group group_ref="remap_1d" grid_ref="grid_landpoints_out" >
  <field field_ref="precip_rain" level="0" name="rainfall" level="1"/>
  <field field_ref="precip_snow" level="0" name="snowfall" level="1"/>
</field_group>

</file>
```

3. Create a new experiment called "`MyPostExp2`" similar to `MyPostExp` used in 4.1. You can start from a copy of `MyPostExp as follows`:

```
cp -r MyPostExp MyPostExp2
cd MyPostExp2

vi config.card    # Change JobName, Set DateEnd=1902-12-31

# Remove files related to MyPostExp
rm Job_MyPostExp run.card Script_Output_MyPostExp.000001

# Create a new job
../../../libIGCM/ins_job
```

Note : you don't need to recompile because you didn't make modification in the code. The xml files are read directly during the execution.

4. Add the new file to be stored in ***`COMP/sechiba.card`*** (see example of `1M_sechiba_history.nc`)
   In `[OutputFiles]` section :

```
(myoutput_orch.nc, {R_OUT_SRF_O_D}/${PREFIX}_1M_myoutput_orch.nc,
Post_1D_myoutput_orch), \
```

Also define the new Post section "`Post_1D_myoutput_orch`" and add the two new variables to be produced as TimeSeries.

```
[Post_1D_myoutput_orch]
Patches = ()
GatherWithInternal = (lon, lat, time_counter, time_centered,
time_centered_bounds)
TimeSeriesVars2D = (rainfall, snowfall)
ChunckJob2D = 200Y
TimeSeriesVars3D = ()
ChunckJob3D = NONE
Seasonal = ON
```

Submit using `sbatch`, `ccc_msub` or `qsub` depending on the platform.

Question : Verify that this new file is created and has all post-processing.

6.2 Enable a new output file in LMDZ

Similarly to the ORCHIDEE mechanism described above, the different output files and their contents in LMDZ are defined in the files
`modeles/LMDZ/DefLists/file_def_*_lmdz.xml`
You can see that there are quite a few of these files. Each one describes the contents of one possible output file for LMDZ. These files may differ by the time averaging used to output variables (monthly means or instantaneous values for example) or may come from different parts of the LMDZ model (the *COSP* ones for example are output by the COSP simulator embedded in LMDZ).
As for the ORCHIDEE example above, the files can be modified to contain specific output if needed. The key words _AUTO_ can be changed directly in the file or using the variables in `lmdz` (section `[UserChoices]`). To save a variable, the file must also be listed in `lmdz.card` (section `[OutputFiles]`) but you will see that most of the files are mentioned (and saved) in the default `lmdz.card`.

In this exercise, you will enable a new output file from LMDZ containing high frequency hourly average values for a small list of variables adding sea-level pressure to that list. Sea-level pressure is already output from the model using `xios_send_field` and is declared in the `field_def_lmdz.xml` with the id `slp`. If you want to see where in the model they are written, all LMDZ output variables are defined and written in the LMDZ routine `phys_output_write_mod.F90` which can be found in the `modipsl/modeles/LMDZ/libf/phylmd/` folder.

If you looked at the files mentioned above, you will have noticed that there already exists a file ***`modeles/LMDZ/DefLists/file_def_histhf_lmdz.xml`*** containing specifications

to output average values every 3 hours of a long list of variables in a file called histhf. We will modify this file to output the desired file and variable.

Set up the file with following specifications:

- The file should be named `myoutput_lmdz.nc`
- The level of the variable `slp` should be set to 5. If you look at the header of the file, you will see that the `output_level` is set to 5; that means that only variables with a `level` less than or equal to 5 will be written out to the file. As, by default, `slp` has a level of 10, it won't be written if you leave it as such.
- File output frequency should be hourly average. You have to set the file attribute ***output_freq="1h"***
- File attribute ***enabled=TRUE***
  There is actually another way to enable writing out this file: it is planned that high frequency outputs can be controlled from the `config.card` file. To do this, you need to add the keyword **`HF`** to the `WriteFrequency` variable of the `ATM` section in `config.card` so that it would read `WriteFrequency="1M HF"`
  But we won't do it this way here.

Do the following:

1. Continue in the same modipsl where you installed LMDZOR in exercise 2.1
2. Modify `LMDZ/DefLists/file_def_histhf_lmdz.xml` with specifications given above:

```
<file id="histhf" name="myoutput_lmdz" output_freq="1h" output_level="5"
enabled="true" compression_level="4">
…
   <field field_ref="slp" level="5" />
…

</file>
```

3. Create a new experiment called "`MyJobTestLMDZ`" similar to `MyJobTest` used in 2.1. You can start from a copy of `MyJobTest as follows`:

```
cp -r MyJobTest MyJobTestLMDZ
cd MyJobTestLMDZ

vi   config.card          #   Change   JobName,   Set   Date=1980-01-05,   Set
PeriodLength=5D

# Remove files related to MyPostExp
rm Job_MyPostExp run.card Script_Output_MyPostExp.000001

# Create a new job
../../../libIGCM/ins_job
```

```
# Make sure the following two lines are in the header of your job file
# for jean-zay
#SBATCH --cpus-per-task=4
#SBATCH --qos=qos_cpu-dev
```

Note : you don't need to recompile because you didn't make modification in the code. The xml files are read directly during the execution.

4. Add the new file to be stored in ***COMP/lmdz.card*** (see example of `histhf.nc`)
   In `[OutputFiles]` section :

```
(myoutput(myoutput_lmdz.nc, ${R_OUT_ATM_O_H}/${PREFIX}_HF_myoutput_LMDZ.nc,
Post_HF_myoutput_LMDZ), \
```

Also define the new Post section "`Post_1D_myoutput_LMDZ`".

```
[Post_HF_myoutput_LMDZ]
Patches= ()
GatherWithInternal = (lon, lat, presnivs, time_counter, time_centered,
time_centered_bounds)
TimeSeriesVars2D = (cldt, psol, q2m, slp, precip, pluc, plul, t2m, tsol,
u10m, v10m)
ChunckJob2D = 50Y
TimeSeriesVars3D = (temp, theta, ovap, vitu, vitv)
ChunckJob3D = OFF
Seasonal=OFF
```

Submit using `sbatch`, `ccc_msub` or `qsub` depending on the platform.

Question : Verify that this new file is created and that it contains the `slp` variable.

After you have finished this example, you should disable writing out the high frequency file again by editing the header of `LMDZ/DefLists/file_def_histhf_lmdz.xml` thus:

```
<file id="histhf" name="histhf" output_freq="3h" output_level="5"
enabled="_AUTO_" compression_level="4">
```

If you don't, all subsequent exercises using LMDZ will try to output a high frequency file that will slow execution of the model.

6.3 XIOS in other models

NEMO, REPROBUS, and INCA models also use XIOS to manage output files.

Where can you find the xml files for these models ?

```
NEMO : modipsl/config/***/GENERAL/PARAM/ (note that directory
will be copy in your simulation directory)
REPROBUS : modipsl/modeles/REPROBUS/XML
INCA : modipsl/modeles/INCA/src/INCA_XML
```

These 3 models use Xios by the same way than LMDZ and ORCHIDEE.
You can find here a documentation for XIOS in Inca model
https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/Models/INCA#ManageoutputusingXIOS
And here for XIOS in Nemo model https://zenodo.org/record/3248739#.XhhOAOEo8ax on page 229.

7. Check your quota

Do the exercises below on the computing center where you have a login.
Remember that you will find the questions' answers in the presentation of the first day and in the IGCMG doc.

7.1 For login at IDRIS

Use the command `idrquota -m` to check the HOME quota, `idrquota -w` for WORK quota and `idrquota -s` for STORE quota.

Question 7a

- Is the quota individual or per project?
- What happens to the other users if you exceed the quota?
- Which type of files do you store in your HOME?

7.2 For login at TGCC

At TGCC space and number of inodes (files and directories) are limited. Use the command `ccc_quota` to show your current quota and the limits on all file systems. Analyse what you see on the screen.

Question 7b

- Is the quota individual?
    - What happens to the other users if you exceed the quota?
- What is your global score?
- What means by "non_files"?
- Which file systems have a limit on the number of inodes?
- What is the size of the files that you are supposed to store at the STOREDIR?

To facilitate the clean you can use the command "find" to list all small files at STOREDIR.

```
cd $CCCSTOREDIR
find . -type f -size -32M
```

7.3 For login at LSCE/obelix

At the LSCE cluster there is an individual quota only at your home, at `/home/users/login`. Use the **`quota`** command to check the quota at your home. At the other disks there are no quota control but they can saturate. Use **`df -h`** to see the occupation of the disks.

Question 7c

- To which disk do you have write permission?
- What happens to the other users if you saturate a disk?

Note that the default base directory for the archive of output files is defined in libIGCM to `/home/scratch01/yourlogin` for obelix. This scratch directory might be purged and therefore you have to change to save your important simulations on another disk. You can change archive by setting the variable “`ARCHIVE`" directly in the config.card or change it in `modipsl/libIGCM/libIGCM_sys_obelix.ksh`.

8. Install and run NEMO-PISCES

This exercise is separated in 2 parts. The first part presents the basic steps to run and install NEMO-PISCES and the second part allows to get much deeper in the use of a configuration of NEMO-PISCES.

First part: In this exercise, we will first perform a 1 month simulation of the coupled ocean-biogeochemical model NEMO-PISCES, using 32 MPI processes for NEMO and 1 MPI process for XIOS. Note that for this configuration some specific commands need to be done. If it is your first time to download NEMO, you first need to register and choose a login/passwd : www.nemo-ocean.eu. (This exercise can not be done on obelix.)

Download modipsl as before and then install the NEMO_v6 configuration :

```
mkdir $WORK/NEMO_STD ; cd $WORK/NEMO_STD
svn co http://forge.ipsl.jussieu.fr/igcmg/svn/modipsl/trunk modipsl

cd modipsl/util
./model NEMO_v6_OMIP
```

Compile the ORCA2_LIM3_PISCES configuration:

```
cd ../config/NEMO_v6   ;  gmake ORCA2LIM3PISCES
```

Create your first job for NEMO:

```
cp EXPERIMENTS/ORCA2_LIM3_PISCES/core/clim/config.card .
```

Now set up the `config.card` to do the simulation. You can see that for the configuration ORCA2_LIM3_PISCES. There are 3 components : OCE for ocean, ICE for Sea-Ice and MBG for PISCES.

Modify in `config.card` the following:

```
vi config.card
JobName=OR2L3P1 ; SpaceName=TEST ; DateEnd=1850-01-31 ; PeriodLength=1M
```

Create the job as usual :

```
../../libIGCM/ins_job
```

Question8a : Explore the `COMP/opa9.card` (`COMP/pisces.card`) to see the inputs files needed for OPA and PISCES

Question8b Explore in `PARAM/NAMELIST/ORCA2` the namelists (`namelist_core_clim_cfg`) to see some parameters for the run

Question8c Explore in `PARAM/XML/file_def_nemo*` files where the output fields are managed for OPA/LIM/PISCES resp.

Submit the job as usual:

```
cd OR2L3P1
sbatch Job_OR2L3P1 / ccc_msub Job_OR2L3P1
```

Question8d Explore the `Script_Output` file and `run.card` in the submit directory

Question8e Explore the output directories where the output files are stored : `OCE/Output` ; `ICE/Output`; `MBG/Output`

Continue the simulation for one more month.

Second part: in this 2nd exercise, we will perform a 1 year long simulation of the coupled ocean-biogeochemical model NEMO-PISCES in an offline mode (ORCA2_OFF_PISCES), using 32 MPI processes for NEMO and 1 MPI process for XIOS. Here, only the biogeochemical fields are computed, NEMO outputs are used to force the dynamical state of the ocean. This allow to explore specific biogeochemical features with lower computational costs. We will see how to create a 5 days outputs file and also the good practice to modify the pisces parameters if needed.

Compile the ORCA2_OFF_PISCES configuration:

```
cd $WORK/NEMO_STD/modipsl/config/NEMO_v6/   ;  gmake ORCA2OFFPISCES
```

Create the job for NEMO-PISCES offline:

```
cp EXPERIMENTS/ORCA2_OFF_PISCES/clim/config.card .
```

Set up the config.card to do the simulation. You can see that for the configuration ORCA2_OFF_PISCES, there is only 1 component : MBG for PISCES.

Modify in config.card the following lines:

```
vi config.card
JobName=OR2OFFPIS ; SpaceName=TEST ; DateEnd=0001-12-31
```

Create the job :

```
../../libIGCM/ins_job
```

Question8f : Explore the `COMP/pisces.card` to see the inputs files from NEMO needed for PISCES

Question8g Explore in `PARAM/NAMELIST/ORCA2` the `namelist_offline_clim_cfg` to see the parameters for the run

Submit the job as usual:

```
cd OR2OFFPIS
sbatch Job_OR2OFFPIS / ccc_msub Job_OR2OFFPIS
```

Question8h Explore the output directories where the output files are stored : `MBG/Output`.

We will now create a new NEMO-PISCES offline configuration. We will modify the `config.card`, the `pisces.card`, and the `file_def_nemo-pisces_offline.xml` to get output of some fields at a frequency of 5 days. We will also see how to modify the parameters in the `namelist_pisces_cfg` file.

Create a new NEMO-PISCES offline configuration

```
cd $WORK/NEMO_STD/modipsl/config/NEMO_v6/
cp EXPERIMENTS/ORCA2_OFF_PISCES/clim/config.card .
```

Modify in `config.card` the following:

```
vi config.card
JobName=OR2OFFPIS2 ; SpaceName=TEST ; DateEnd=0001-12-31 ;
[MBG]
WriteFrequency="5D 1M 1Y"
```

Create the job :

```
../../libIGCM/ins_job
```

Edit the `pisces.card` to add 5 days outputs for `*.ptrcT` file :

```
cd OR2OFFPIS2/
vi COMP/pisces.card
```

Add the following line in the `[OutputFiles]` list of the `pisces.card` file:

```
...
(${config_UserChoices_JobName}_5d_ptrc_T.nc,${R_OUT_MBG_O_D}/${PREFIX}_5D_
ptrc_T.nc   , NONE   ) , \
...
```

Add in the `PARAM/XML/file_def_nemo-pisces_offline.xml` the variables NO3, PO4, Si, Fer, DCHL, NCHL in the specific group of 5d files.

```
vi PARAM/XML/file_def_nemo-pisces_offline.xml
```

Replace the `<!-- 5d files -->` line below:

```
<file_group     id="5d_pis"     output_freq="5d"          output_level="10"
enabled="_AUTO_"/>  <!-- 5d files -->
```

by the following lines in the `file_def_nemo-pisces_offline.xml`

```
<file_group       id="5d_pis"       output_freq="5d"       output_level="10"
enabled="_AUTO_">  <!-- 5d files -->
<file id="file35" name_suffix="_ptrc_T" description="pisces sms variables"
>
<field field_ref="PO4"         name="PO4"   />
<field field_ref="NO3"         name="NO3"   />
<field field_ref="Si"          name="Si"    />
<field field_ref="NCHL"        name="NCHL"  />
<field field_ref="DCHL"        name="DCHL"  />
</file>
</file_group>
```

We have finished to set up the configuration to get biogeochemical fields at an output frequency of 5 days for the `*ptrc_T` file.

Now we will see how to modify the parameters of the namelist of pisces. For instance, we will remove the sediment source of Fe and will explore the impacts for surface Fe, chlorophyll, nitrate, and Si, particularly in coastal regions.

open the `pisces.card`

```
vi COMP/pisces.card
```

question8j: Find where the reference namelist of pisces is stored. Open the the file.

```
vi ../../../modeles/NEMOGCM/CONFIG/SHARED/namelist_pisces_ref
```

All the parameters of pisces are listed here. This file should not be modified if you want/need to change some pisces parameters

question8k: Explore the `namelist_pisces_ref`

Copy the parameter for inputs deposition from the `namelist_pisces_ref` in the `namelist_pisces_cfg` of your configuration.

Copy the line below from the `namelist_pisces_ref`

```
   ln_ironsed  =  .true.   ! boolean for Fe input from sediments
```

Paste the copied line in the `namelist_pisces_cfg` in the section of nampissbc:

```
vi PARAM/NAMELIST/namelist_pisces_cfg
```

change the boolean value to switch off the Fe input from sediment in the `namelist_pisces` cfg

```
   ln_ironsed  =  .false.
```

submit the job:

```
cd OR2OFFPIS
sbatch Job_OR2OFFPIS2 / ccc_msub Job_OR2OFFPIS2
```

question8l: Explore the output directories where the output files are stored to check whether the 5d `*ptrc_T` file has been created: `MBG/Output`

question8m: Compare the annual output files of the 2 offline configurations (OR2OFFPIS, OR2OFFPIS2) and explore the differences on surface Fe,CHL, NO3, and Si.

9. REDO

Sometimes, because of machine problems (or other unknown reasons), output files are missing. Here is how to recover missing output files. The general method is explained on FAQ of the documentation:

http://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/FAQ#HowdoIrestartasimulationtorecovermissingoutputfiles

As an example, we suggest you to :

- launch a 3 days simulation of LMDZOR experiment
- remove output files for 1 day of the simulation
- apply the method to recover missing output files

9.1 Launch a 3 days simulation of LMDZOR experiment

```
cd modipsl/config/LMDZOR_v6
cp EXPERIMENTS/LMDZOR/clim_360d/config.card .

vi config.card
    # Modify JobName=MyJobTest-3D
    # SpaceName=DEVT
    # Note : REDO method does not work with TEST as SpaceName
    # DateBegin=1980-01-01
    # DateEnd=1980-01-03
    # PeriodLength=1D
    # PackFrequency=NONE
    # Modify Executable part for the parallelization
[Executable]
ATM= (gcm.e, lmdz.x, 71MPI, 8OMP)
SRF= ("", "")
SBG= ("", "")
IOS= (xios_server.exe, xios.x, 1MPI)

     # At obelix only, change to 7 MPI and 1 OMP in
     # At Irene, change nothing for parallelization
     # At JeanZay, change 8 OMP by 5 or 10
     # At JeanZay training day change to 5 OMP
```

```
../../libIGCM/ins_job  # At JeanZay enter your project ID
                       # At Irene enter your project ID and default answer for other
questions

cd MyJobTest-3D
vi Job_MyJobTest-3D
   # Modify
#SBATCH --time=00:30:00                          # Wall clock limit (seconds)
#SBATCH --qos=qos_cpu-dev
PeriodNb=3

sbatch Job_MyJobTest-3D
```

9.2 Remove daily output file for ATM component of the day 2 (i.e 1980-01-02)

```
Check
$STORE/IGCM_OUT/LMDZOR/DEVT/clim/MyJobTest-3D/ATM/Output/DA/MyJob
Test-3D_19800102_19800102_1D_histday.nc exists...then remove it
rm -f
$STORE/IGCM_OUT/LMDZOR/DEVT/clim/MyJobTest-3D/ATM/Output/DA/MyJob
Test-3D_19800102_19800102_1D_histday.nc
```

9.3 Apply the method to redo day 2 of the simulation (to recover missing output file)

```
# Handling of the restart files of the new simulation

mkdir -p $STORE/IGCM_OUT/LMDZOR/REDO/clim/MyJobTest-3D
cd $STORE/IGCM_OUT/LMDZOR/REDO/clim/MyJobTest-3D
mkdir -p ATM/Restart SRF/Restart SBG/Restart
cp
../../../DEVT/clim/MyJobTest-3D/ATM/Restart/MyJobTest-3D_19800101
_restart.nc ATM/Restart/.
```

```
cp
../../../DEVT/clim/MyJobTest-3D/ATM/Restart/MyJobTest-3D_19800101
_restartphy.nc ATM/Restart/.
cp
../../../DEVT/clim/MyJobTest-3D/SRF/Restart/MyJobTest-3D_19800101
_sechiba_rest.nc SRF/Restart/.
cp
../../../DEVT/clim/MyJobTest-3D/SBG/Restart/MyJobTest-3D_19800101
_stomate_rest.nc SBG/Restart/.

# Set up of the new simulation

cd modipsl/config/LMDZOR_v6
cp -pr MyJobTest-3D MyJobTest-3D-REDO
cd MyJobTest-3D-REDO

# In this new directory, change the run.card and config.card file and set the following
parameters to:
vi run.card
   # PeriodDateBegin= 1980-01-02
   # PeriodDateEnd= 1980-01-02
   # CumulPeriod= 2 # Specify the same period in the run.card of initial simulation
   # PeriodState= OnQueue
   # SubmitPath= ...modipsl/config/LMDZOR_v6/MyJobTest-3D-REDO
vi config.card
# you don't need to change the name of the simulation
    # SpaceName=REDO
    # DateEnd= 1980-01-02

sbatch Job_MyJobTest-3D
```

Once the job is finished you can have a look on $STORE/IGCM_OUT/LMDZOR/REDO/clim/MyJobTest-3D/ATM/Output/DA/MyJobTest-3D_19800102_19800102_1D_histday.nc

Once validated the new run (same results as the previous one : comparaison of restart files at the end of the day 2), you can copy the new file in the initial directory :

```
cp
$STORE/IGCM_OUT/LMDZOR/REDO/clim/MyJobTest-3D/ATM/Output/DA/MyJob
Test-3D_19800102_19800102_1D_histday.nc
$STORE/IGCM_OUT/LMDZOR/DEVT/clim/MyJobTest-3D/ATM/Output/DA/.
```

10. Output files manipulations

This section will propose some exercises to present you common tools used in climate/meteo community to manipulate data. This is not an exhaustive list of tools and the idea is to perform the same basic output manipulations and let you see which one seems the most suitable for you. Be careful however only one simple use case, and some tools could appear complicated compared to others whereas it could be different for complex analysis ; that's why there is a quick conclusion paragraph where we bring additional information and a point a view of the best usage. This is only a point and everybody has to discuss with people, read docs and test them to conclude.

Note that we won't speak about Climaf in this section which get its own practical in next section.

10.0 Protocol and environment

10.0.1 Protocol

In the following sections you will use several tools/languages to load the daily atmospherical output file produced by LMDZ. Extract the "2m-temperature" field (`t2m`) and save it as a timeserie file. Then we propose to compute a zonal and global weighted mean (using latitude cosine) and finally plot them.

Note that you could use another variable or output instead.

10.0.1 Environment

Before starting you need to check that the following modules are available: `module list`

```
1) netcdf/4.7.2-mpi       2) nco/4.8.1
3) ferret/7.2             4) netcdf/4.7.2-mpi
5) ncview/2.1.7-mpi       6) cdo/1.9.7.1
7) ncl/6.6.2-mpi          8) python/3.7.5
```

Otherwise you could load them using `module load` command as follow:

```
module load netcdf/4.7.2-mpi
module load nco/4.8.1
module load cdo/1.9.7.1
module load ncview/2.1.7-mpi
module load ferret/7.2
module load ncl/6.6.2-mpi
module load python/3.7.5
```

10.1 Network Common Data Form (NetCDF) format

In the IPSL models the output format is NetCDF. NetCDF is "*self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data"* (from Wikipedia). This is a binary format and need some tools and/or a particular library to use them.

When the NetCDF library is installed on a computer, some basic manipulation tools are supplied. This is the case of the `ncdump` command which allow you to see the content of a netCDF file.

Use it with the option `-h` to get header information only, no data:

```
cd $SCRATCH/IGCM_OUT/LMDZOR/TEST/clim/MyJobTest/ATM/Output/DA/
ncdump -h MyJobTest_19800101_19800130_1D_histday.nc
```

Question: Look at the file structure, how is composed ? Explore other variables or components (SBG, MBG, OCE, ICE…). Are they structured in the same way ?

Informations: https://www.unidata.ucar.edu/software/netcdf/

10.2 NetCDF Operator (NCO)

We're going to use the atmospherical output file produced in the "basic exercise" (section 1).

The general atmospheric file `MyJobTest_19800101_19800130_1D_histday.nc` content all variables set in `lmdz.card`. To avoid to manipulate this big file, we'll first create our own timeseries file for the 2D temperature *t2m* (this is the same process done during a simulation).

To extract this variable use `ncks` as follow:

```
cd $STORE/MyJobTest/ATM/Output/DA
ncks -v t2m MyJobTest_19800101_19800130_1D_histday.nc t2m_TS.nc
```

Question: Check the output file content using `ncdump -h`
Now, to calculate an area-averaged index, you first need to add the *latitude weights* to the file with `ncap2` before computing average with `ncwa` (-O option is to overwrite file):

```
# Add cos(latitude) to balance all grid point contribution
ncap2 -h -O -s "weights=cos(lat*3.1415/180)" t2m_TS.nc t2m_TS.nc
# Global average
ncwa -h -O -w weights -a lat,lon t2m_TS.nc t2m_glob_mean.nc
# Zonal average
ncwa -h -O -a lon t2m_TS.nc t2m_zon_mean.nc
```

Question: add the keyword `time` before each command and note the time elapsed to compare performances with CDO presented in the following section.

Conclusion: NCO is a very common set of several tools used through a terminal. It is generally installed on computation centers and frequently updated. As shown in the exercise before it creates a lot of intermediary files if you need to perform a complex analysis but it is optimized to perform quickly some complex analysis and use large files. There is no visualisation in NCO.

Informations: http://nco.sourceforge.net

10.3 Climate Data Operators (CDO)

CDO is a set of tools very useful to manipulate climate data. Its usage is close to NCO (see previous section) with its own operators. The CDO syntax is the following :

`cdo <operator>,<option> input.nc output.nc`

Let's start with variable extraction with the `selvar` operator:

```
cd $STORE/MyJobTest/ATM/Output/DA
cdo selvar,t2m MyJobTest_19800101_19800130_1D_histday.nc t2m_TS_CDO.nc
```

Question: Check the output file content using `ncdump -h`. You could print information and simple statistics for each field of a dataset using `cdo info t2m_TS_CDO.nc` (mean is computed without the area weights).

Now perform the same analysis than before: global weighted average with `fldmean` (use directly grid info to find area weights) and zonal one using `zonmean`:

```
# Global average
cdo fldmean t2m_TS_CDO.nc t2m_glob_mean_CDO.nc
# Zonal average
cdo zonmean t2m_TS_CDO.nc t2m_zon_mean_CDO.nc
```

Question: add the keyword `time` before each command and note the time elapsed to compare performances with NCO presented in previous section.

Conclusion: CDO is a set of tools, developed by the Max Planck institute, similar to NCO. The syntax is a bit different but it allows to perform almost the same things. Sometime it is easier to perform some analysis with CDO, sometimes with NCO. Both could be used and chained. However the memory optimisation seems better with NCO. It also create temporary files to clean after and doesn't propose visualisation. The documentation is not so easy to find on the internet.

Informations: https://code.mpimet.mpg.de/projects/cdo

10.3 NetCDF Visual browser (NCView)

NCView is a very basic NCDF file visual browser. We propose to use it to show outputs from previous exercises and let you play with its basic interface (need to select the *t2m* variable):

```
# plot global mean
ncview t2m_glob_mean.nc
# show zonal average
ncview t2m_glob_mean.nc
```

Conclusion: It could be useful to check quickly file content and show data (with >> you could play data along an axis) ; but it is still very basic and doesn't allow to perform analysis.

Informations: http://meteora.ucsd.edu/~pierce/ncview_home_page.html

10.4 Ferret

Open ferret and load the *t2m timeserie* file (created with NCO in 10.2) or the global daily one *histday* otherwise:

```
cd $STORE/MyJobTest/ATM/Output/DA
ferret # go into ferret app

> USE "MyJobTest_19800101_19800130_1D_histday.nc”
> SAVE/FILE="t2m_TS_FERRET.nc" t2m[d=1]
> use "t2m_TS_FERRET.nc"
> show data/f 2 ! show all info in dataset 2 (ie t2m TS)
```

Note: Ferret is not case sensitive so it ignores lower and upper case for commands and variable names.

Now you will compute the zonal mean using `@ave` command to do it (Ferret automatically weighted average using grid properties) and show it with `shade`:

```
shade t2m[y=@ave, d=2]
```

And then plot the global average:

```
plot t2m[x=@ave,y=@ave, d=2]
```

Conclusion: It is a very good tool for quick sanity checks. Very easy to load/save data, basic data manipulation (averages, sums) and plot timeseries and 2D view.
Otherwise syntax is not very friendly (there is no variables but aliases saved), generate bad image quality (need to use PyFerret to solve this), only few doc and not very active developments.

Informations: https://ferret.pmel.noaa.gov/Ferret/

10.5 NCAR Command Language (NCL)

NCL is an environment developed by NCAR people. It was very popular in the weather and climate community, particularly for the large panel of visualisation proposed.

First, you'll start the NCL environment using `ncl` command line (use `ctrl+d` to exit):

```
ncl
```

Then you'll create a `t2m` timeserie from model output file ; using `addfile()` function to load model output, then select the variable to finally create a new output with `"c"` option and write it:

```
df  = addfile("MyJobTest_19800101_19800130_1D_histday.nc","r")
temp = df->t2m                    ; store t2m in a variable
fout=addfile("t2m_TS_NCL.nc","c") ; create out file
fout->t2m=temp                    ; write temp in t2m variable
```

Note: You could show quick information about a variable using `printVarSummary` command. For example to look at the temperature info: `printVarSummary(temp)`

Question: Quit ncl via `ctrl+d` and look inside the new created file using `ncdump -h`

Now continue loading the t2m file just created to compute the weighted global using `wgt_areaave_Wrap` (`Wrap` is to keep metadata) function and then plot it into a "*ave.png*" file (don't forget to start the ncl program first!):

```
df  = addfile("t2m_TS_NCL.nc","r"); read t2m TS
temp = df->t2m                    ; store t2m in a variable
lat  = df->lat                    ; store lat in a variable
rad  = 4.0*atan(1.0)/180.0
clat = cos(lat*rad)               ; lat cosine
globav = wgt_areaave_Wrap(temp, clat, 1.0, 0) ; global average

; *** create graphic into ave.png file ***
wks  =  gsn_open_wks("png","globave")      ;  send  graphics  to  PNG
file
res = True
res@tiYAxisString= globav@long_name + " (" + globav@units + ")"
res@tiXAxisString= "Time Steps"
res@tiMainString = "Global Weighted Average"
```

```
x = ispan(0,dimsizes(globav)-1,1)      ; create x-axis
plot = gsn_csm_xy(wks,x,globav,res)    ; create plot
```

Question: you could have a look at the output in the *globave.png* file using for example `display` command such as `display globave.png`

Now proceed to the zonal mean using `dim_avg_n_Wrap` which averaged the rightmost dimension (so you need to permute them if it is not the `lon`):

```
df  = addfile("t2m_TS_NCL.nc","r"); read t2m TS
temp = df->t2m                    ; store t2m in a variable
zave = dim_avg_n_Wrap(temp,2)     ; zonal average (=dim 2)

; *** create graphic into ave.png file ***
wks  = gsn_open_wks("png","zonal") ; send graphics to PNG file
res                  = True                ; plot mods desired
res@tiMainString     = "Hovmoller"         ; title
res@tmXBLabelStride   = 2                      ; tick mark label
stride
res@tiYAxisString    = "Time"              ; y axis title
res@tiXAxisString    = "Lat"               ; x axis title

res@cnFillOn         = True                ; color on
res@lbLabelStride    = 2                   ; every other label
res@lbOrientation    = "Vertical"          ; vertical label bar
res@cnLinesOn          = False                 ; turn off contour
lines
res@cnFillPalette    = "gui_default"    ; set color map
res@cnLevelSpacingF  = 1                   ; contour spacing

plot = gsn_csm_time_lat(wks, zave, res )   ; plot zonal ave
```

Question: you could have a look at the output in the *zonal.png* file using for example `display` command such as `display zonal.png`

Conclusion: NCL is a very powerful tool with a good documentation and community. For about 1 year, the developers announced that the environment won't be updated but all the functionalities will become a Python library PyNIO and PyNGL for the graphical part. The project is called the Geosciences Community Analysis Toolkit (GeoCAT), and now get a specific website. So we advise you to directly use the Python version.

Informations: http://www.ncl.ucar.edu and https://geocat.ucar.edu (Python version)

10.6 Python

First you will probably to load python module: `module load python/3.7.5 and then start ipython3`

10.6.1 NetCDF4 / Numpy

Read NetCDF file, extract *t2m* variable and write its timeserie:

```
from netCDF4 import Dataset, num2date, default_fillvals

import numpy as np
import matplotlib.pyplot as plt

# load dataset
fnc=Dataset("MyJobTest_19800101_19800130_1D_histday.nc",
mode='r')
# extract t2m and dimension variables
temp = fnc.variables['t2m']
time = fnc.variables['time_counter']
lati = fnc.variables['lat']
long = fnc.variables['lon']

# Create output file
fout = Dataset("t2m_TS_NC.nc", mode='w')
# create dimensions
fout.createDimension('time_counter', None)
fout_tdim   =   fout.createVariable('time_counter',   time.dtype,
('time_counter',))
fout.variables['time_counter'][:] = time[:]
for ncattr in time.ncattrs(): # copy metadata
     fout_tdim.setncattr(ncattr, time.getncattr(ncattr))

fout.createDimension('lat', len(lati))
fout_latdim = fout.createVariable('lat', lati.dtype, ('lat',))
fout.variables['lat'][:] = lati[:]
for ncattr in lati.ncattrs():
     fout_latdim.setncattr(ncattr, lati.getncattr(ncattr))

fout.createDimension('lon', len(long))
fout_londim = fout.createVariable('lon', long.dtype, ('lon',))
fout.variables['lon'][:] = long[:]
for ncattr in long.ncattrs():
```

```
     fout_londim.setncattr(ncattr, long.getncattr(ncattr))

# create variables
temp_var      =      fout.createVariable('t2m',      temp.dtype,
('time_counter', 'lat', 'lon'),fill_value=True)
for ncattr in temp.ncattrs():
     # patch for some version of python
     if(ncattr == '_FillValue'):
          continue
     temp_var.setncattr(ncattr, temp.getncattr(ncattr))
fout.variables['t2m'][:] = temp[:]

fout.close() # close file
```

Now load the timeserie file and compute zonal and global averages using `numpy`:

```
ftemp=Dataset("t2m_TS_NC.nc", mode='r')
temp = ftemp.variables['t2m']
lat = ftemp.variables['lat']
wgt = np.cos(np.deg2rad(lat))    # lat cosine
zave= np.average(temp, axis = 2) # zonal average
gave= np.average(zave, axis = 1, weights = wgt) # global weighted
```

And plot results using `matplotlib` library:

```
plt.show(block=False)  # let you continue to write
plt.plot(gave)
plt.figure()           # create new figure
plt.contourf(zave, cmap=plt.cm.YlOrBr)
plt.colorbar()         # show colorbar
```

10.6.1 XArray

XArray library is a Python package that makes working with labelled multi-dimensional arrays simple. It is based on Numpy and Pandas and use Matplotlib by default to plot data. Let's start with the t2m TS file creation in python (don't forget to start Python using `python` command):

```
import xarray as xr
```

```
import numpy as np
import matplotlib.pyplot as plt
ds = xr.open_dataset("MyJobTest_19800101_19800130_1D_histday.nc")
temp = ds.t2m                    # store t2m in a variable
temp.to_netcdf("t2m_TS_XR.nc") # write temp in t2m variable
```

Now compute zonal and global averages (need to first compute zonal):

```
dst = xr.open_dataset("t2m_TS_XR.nc")
temp=dst.t2m
wgt = np.cos(np.deg2rad(dst.lat)) # lat cosine
zave= temp.mean(dim="lon") # zonal average
gave=(zave*wgt).sum(dim=('lat'))/wgt.sum(dim=('lat'))#glob
weighted
```

And plot results using `matplotlib` library:

```
plt.show(block=False)  # let you continue to write
plt.plot(gave)
plt.figure()           # create new figure
plt.contourf(zave, cmap=plt.cm.YlOrBr)
plt.colorbar()         # show colorbar
```

Note: when computing averages with XArray internal functions, the metadata will be kept. You could see it if you try to print variables `print(zave)`.

Conclusion:

- NetCDF is the basic library which allow you to work at a very low level in the same way that other environments based on it. It is powerful but need to explicit a lot of things (in particular create dimensions and metadata) that could afraid users.
- XArray in another way, adds a lot of very confortable simplicity to manipulate netCDF files and to manage metadata. It is powerful too and allow to convert data in other numpy types to use other libraries but need a bit of learning.

In a general way, Python seems to become the reference language for data analysis in climate or other field through the impressive amount of libraries available (maybe too much) and each user get its favourite's ones.

Informations: NetCDF4 - https://unidata.github.io/netcdf4-python/netCDF4/index.html
XArray - http://xarray.pydata.org

11. CliMAF and the C-ESM-EP

CliMAF (for Climate Model Assessment Framework, https://climaf.readthedocs.io/en/master/) is a python library developed in collaboration between IPSL and CNRM (ANR Convergence project) for easier analysis of climate model outputs. It works very efficiently on IPSL climate model outputs (as well as on CMIP5 or CMIP6 outputs, and any CF-compliant netcdf file).
The C-ESM-EP (CliMAF Earth System Model Evaluation Platform, https://github.com/jservonnat/C-ESM-EP/wiki) is an evaluation package built with CliMAF, also in collaboration with CNRM and CERFACS. It produces sets of model evaluation diagnostics but can also be used to build your own set of diagnostics.

CliMAF and the C-ESM-EP have been used intensively during the development of the IPSL-CM6A-LR to produce the numerous evaluation diagnostics to follow the evolution of the climatology of the model due to the new developments.

In brief, CliMAF should help you:

- Do all the basic data treatments you do every day, like selection of a variable, period, geographical domain, finding you data
- Avoid re-computation of a diagnostic: CliMAF has a smart-cache system that stores all your results in a way that avoids recomputing an existing result
- Do nice plots
- Work with ensembles
- Build an html page with your results
- Or build your own set of diagnostics that you will use routinely

We invite you to look at the CliMAF documentation (https://climaf.readthedocs.io/en/master/) and particularly at the several notebooks (https://climaf.readthedocs.io/en/master/#can-climaf-make-my-scientific-life-easier and https://climaf.readthedocs.io/en/master/#cmip6-cmip5-climeri-convergence-training-session-november-19-2018) available.

While CliMAF is a python library that you can use interactively in a python prompt or a jupyter notebook, the C-ESM-EP is a package that submits jobs. If you ask yourself whether you should use basic CliMAF or the C-ESM-EP to do what you need, the idea is: if you are not interested specifically by the scientific content of the C-ESM-EP (https://github.com/jservonnat/C-ESM-EP/wiki/Scientific-content-of-the-C-ESM-EP), you better start by manipulating CliMAF and getting used to the core functionalities. Once you know how to do your diagnostic with CliMAF, it will be very easy to add it to the C-ESM-EP (https://github.com/jservonnat/C-ESM-EP/wiki/Add-your-own-diagnostic-to-the-C-ESM-EP_v2) or simply your own html page (https://climaf.readthedocs.io/en/master/_downloads/83abb8e0dd9cf2ac147089d92ae8bb5b/Gathering_my_results_in_an_html_page.html).

We organize weekly hands-on sessions (called “bocal”), mainly in Jussieu (but we can do it also at LSCE if we know that some people are interested). If you want to know receive the informations about those “bocal” sessions, subscribe to the CliMAF users mailing list: https://climaf.readthedocs.io/en/master/community.html

Instructions for the practical on Ciclad:

- Set your environment: https://climaf.readthedocs.io/en/master/installing.html#using-climaf-at-cnrm-on-ciclad-or-climserv-fast-track
- Create the directory for the practicals and get the examples notebooks :
  mkdir TP_CLIMAF
  cd TP_CLIMAF
  cp $CLIMAF/examples/*.ipynb .
- run the jupyter notebook:
  - Give your ciclad username to Jerome Servonnat (jerome.servonnat@lsce.ipsl.fr)
  - And follow the instructions on this page: https://climaf.readthedocs.io/en/master/installing.html#using-climaf-at-cnrm-on-ciclad-or-climserv-fast-track
- Pick up a notebook and start playing! In order:
  - CliMAF in a nutshell
  - Basis of CliMAF data access at CLIMERI
  - Main operators and how to plug your own script
  - Getting started with plot
  - Working with ensembles

12. Ensembles

To configure an ensemble of simulations with slightly different perturbed initial conditions it is possible to use “ins_job -e” option.
To use this option ensemble.card file is needed.
We give here an example of config.card and ensemble.card to generate 2 members, starting date 1851. A white noise (of 0.1) is applied to the SST of a restart of a historical simulation.

There are two types of Ensemble :
[Ens_PERTURB] : configures a set of period (annual) simulations from a Start date to an End date, with a defined number of members
[Ens_DATE] : configures a set of simulations using several restart dates

NOTE: for ensemble **JobName** in config.card IS TO BE THE SAME that **NAME** in ensemble.card

Important:
verify that in your config.card there is a Section “[Ensemble]”:

```
#============================
[Ensemble]
#D- Ensemble run ? 'y' or 'n'
#D- If 'y', fill in ensemble.card !!
EnsembleRun=y
EnsembleName=
EnsembleDate=
EnsembleType=
```

```
vi $WORK/MYFIRSTTEST/modipsl/libIGCM/ins_job

Change line:
RUN_DIR="${CCCWORKDIR}/ENSEMBLE_TMP"
in:
RUN_DIR="${WORK}/ENSEMBLE_TMP"
```

```
cd modipsl/config/IPSLCM
cp EXPERIMENTS/IPSLCM/decadal/config.card .
cp EXPERIMENTS/IPSLCM/decadal/ensemble.card .
```

An example of [Ens_PERTURB] :

```
vi config.card  # Modifiy using these following lines
    JobName=ENS
    SpaceName=TEST
    DateBegin=1851-01-01
    DateEnd=1851-12-31
    PeriodLength=1Y

vi ensemble.card # write this following lines
    [Ens_PERTURB]
    active=y
    NAME=ENS
    MEMBER=2
    LENGTH=1Y

   BEGIN_INIT=18510101
   END_INIT=18511231
   PERIODICITY=1Y

   PERTURB_BIN=(AddNoise, CPL, sstoc, O_SSTSST, 0.1)

   INITFROM=CM61-pi-valid.02
   INITPATH=$STORE/../../../rech/psl/commun/IGCM_OUT/IPSLCM6/DEVT/piControl

../../libIGCM/ins_job -e  # At JeanZay enter your project ID
                          # At Irene enter your project ID and default answer for other questions

cd ENS
vi Qsub.ENS1851.sh
```

This example generates 2 members of simulation starting in 1851, from restart simulation on Jean Zay (IDRIS) :

`$STORE/../../../rech/psl/commun/IGCM_OUT/IPSLCM6/DEVT/piControl/CM61-pi-valid.02`

White Noise is applied to sst; you can verify perturbed variables here:

$WORK/IGCM_IN/IPSLCM6/`JobNameYEAR/JobNameYEAR-0$member/`CPL/Restart

In this example JobNameYEAR is “ENS1851”, and subdirectories are ENS1851-01 and ENS1851-02.

The submission directory has been created with the same name as the `JobNameYEAR`. In this directory there are as many directories as number of members. Look at `JobNameYEAR` directory and explore subdirectories.
In `JobNameYEAR` there is a shell script that can be launched (chmod 755 `Qsub.ENS1851.sh; sh Qsub.ENS1851.sh`) . With this script all members of all years will be launched.

For more information see documentation :

https://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc#Ensemblesetup

13. Coupled model

The aim of this part is to apply ***what you have learnt in part 2*** : performing extraction, compilation and run of the whole coupled (ocean-atmosphere) model configuration IPSLCM6.1.10-LR. So you have to :

- Extract modipsl
- Extract IPSLCM6.1.10-LR configuration
- Compile
- Set up a 5 days piControl experiment (piControl_TEST experiment)
- Launch the simulation
- Check output files of the simulation

```
For training day do not compile (gmake command) but copy in modipsl/bin
executables stored in

$WORK/../../../rech/psl/commun/TRAINING/MODIPSL_HandsOn_20200114/bin_IPSL
CM/

And create these 2 files :
modipsl/config/IPSLCM6/.resol with commands
        echo "ORCA1LIM3xLMD144142-L79" >.resol
       echo "RESOL_ATM_3D=144x142x79" >>.resol
modipsl/config/IPSLCM6/.libmpi with commands
       echo "MPI1" >.libmpi
```