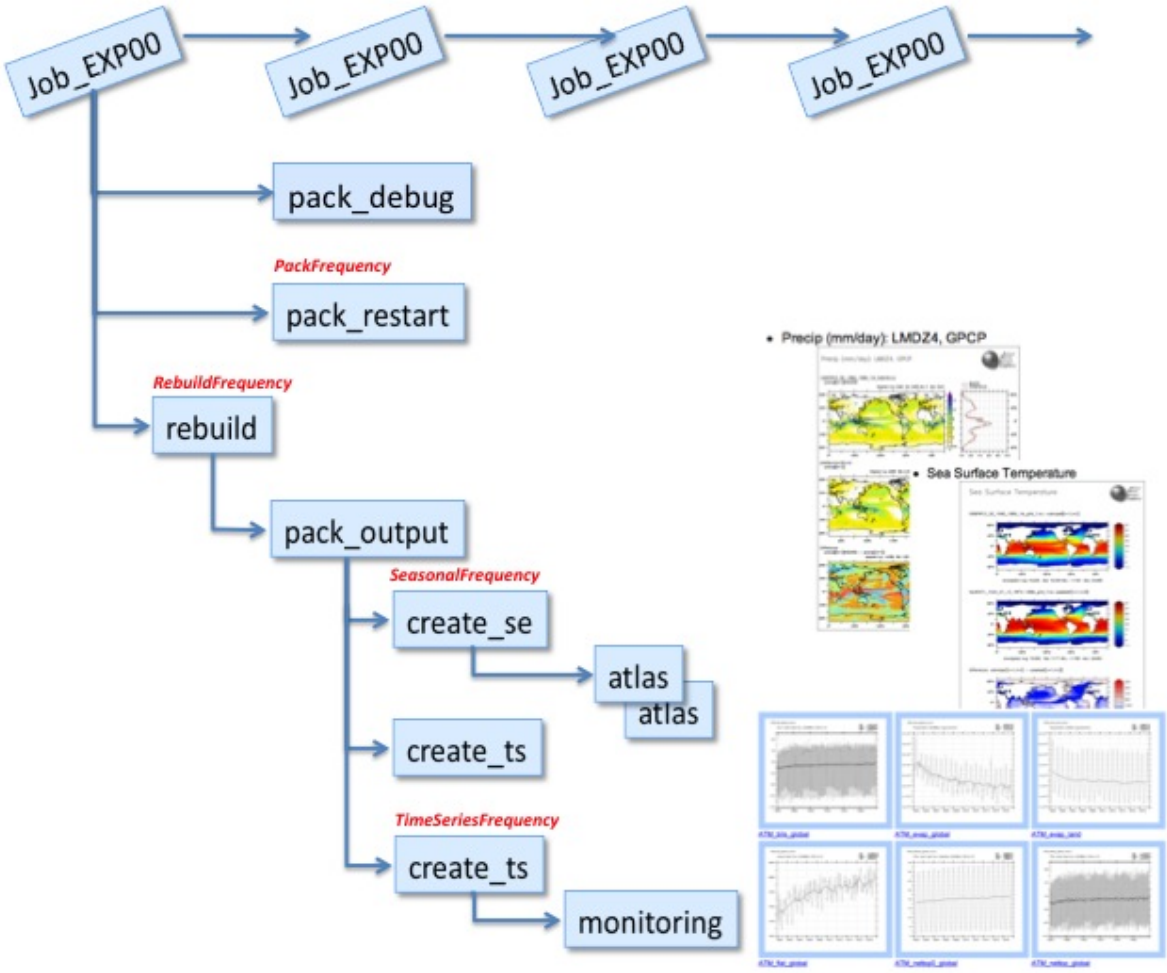
Running simulation and post-processing

In this chapter you will learn about how to start a simulation and how to use the IPSL models and tools, from the beginning of the simulation to the post processing and basic visualization of the outputs.

Table of Content

1. Overview of IPSL running environment workflow

The main computing job automatically runs post processing jobs (at different frequencies) during the simulation. Here is a diagram describing the job sequence:



2. Simulation - Computing part

2.1. Submitting your simulation

Once you have defined and setup your simulation you can submit it. The run commands are:

- **`ccc_msub`** at TGCC
- **`sbatch`** at IDRIS

```
irene > ccc_msub Job_MYJOBNAME
jean-zay   > sbatch Job_MYJOBNAME
```

These commands return a job number that can be used with the machine specificities to manage your job. Please refer to the Computing Center page of your machine.

Before starting a simulation it is very important to double check that it was properly setup. We strongly encourage you to perform a short test before starting a long simulation.

The job you just submitted is the first element of a sequence of jobs. These jobs include the computing job itself, post processing jobs like: *pack*, *create_ts*, *create_se* and visualization jobs like *monitoring* and *atlas* which are started at given frequencies.

If you recompile the modele during a simulation, the new executable will be used in the next period of the running job.

2.1.1. Launch a simulation test

On config.card :

```
PeriodLength=1D ou 5D
SpaceName=TEST
TimeSeriesFrequency=NONE
SeasonalFrequency=NONE
```

Run on test queue:
modify beginning of main job :

- irene :

```
#MSUB -Q test
#MSUB -T 1800
```

- jean-zay :

```
#SBATCH --time=00:30:00             # Wall clock limit (seconds)
#SBATCH --qos=qos_cpu-dev
```

2.2. Status of the running simulation

2.2.1. run.card during the simulation

A *run.card* file is created as soon as your simulation starts. It contains information about your simulation, in particular the PeriodState parameter which is:

- `Start` or `OnQueue` if your simulation is queued
- `Running` if your simulation is being executed
- `Completed` if your simulation was successfully completed
- `Fatal` if your simulation was aborted due to a fatal error

2.2.2. Execution directory

At TGCC your simulation is performed in a $CCCSCRATCHDIR/RUN_DIR/job_number directory. At IDRIS it's on a $SCRATCH/RUN_DIR/job_number directory. You can check the status of your simulation in this directory.

2.2.3. Accounting mail

You receive a mail « Simulation Accounting » that indicates the simulation starts fine, how many NbPeriodsPerJob you can use to be efficient and how many computing hours the simulation will consume. For example :

```
Dear Jessica,

this mail will be sent once for the simulation CURFEV13 you recently submitted

The whole simulation will consume around 10074.036500 hours. To be compared with your project allocation.

The recommended NbPeriodsPerJob for a 24 hours job seems to be around 38.117600. To be compare with the current setting (J

Greetings!
```

2.3. End of the simulation

2.3.1. messages received

Example of message for a successfully completed simulation

```
From : no-reply.tgcc@cea.fr
Object : CURFEV13 completed

Dear Jessica,

Simulation CURFEV13 is completed on supercomputer irene3820.
Job started : 20000101
Job ended   : 20001231
Output files are available in /ccc/store/.../IGCM_OUT/IPSLCM5A/DEVT/pdControl/CURFEV13
Files to be rebuild are temporarily available in /ccc/scratch/.../IGCM_OUT/IPSLCM5A/DEVT/pdControl/CURFEV13/REBUILD
Pre-packed files are temporarily available in /ccc/scratch/.../IGCM_OUT/IPSLCM5A/DEVT/pdControl/CURFEV13
Script files, Script Outputs and Debug files (if necessary) are available in /ccc/work/.../modipsl/config/IPSLCM5_v5/CURFE
```

Example of message when the simulation failed

```
From : no-reply.tgcc@cea.fr
Object : CURFEV13 failed

Dear Jessica,

Simulation CURFEV13 is failed on supercomputer irene3424.
Job started : 20000101
Job ended   : 20001231
Output files are available in /ccc/store/.../IGCM_OUT/IPSLCM5A/DEVT/pdControl/CURFEV13
Files to be rebuild are temporarily available in /ccc/scratch/.../IGCM_OUT/IPSLCM5A/DEVT/pdControl/CURFEV13/REBUILD
Pre-packed files are temporarily available in /ccc/scratch/.../IGCM_OUT/IPSLCM5A/DEVT/pdControl/CURFEV13
Script files, Script Outputs and Debug files (if necessary) are available in /ccc/work/.../modipsl/config/IPSLCM5_v5/CURFE
```

2.3.2. run.card at the end of a simulation

At the end of your simulation, the PeriodState parameter of the *run.card* files indicates if the simulation has been **completed** or was aborted due to a **Fatal** error.
This file contains the following sections :

- Configuration : allows you to find out how many integration steps were simulated and what would be the next integration step if the experiment would be continued.

```
[Configuration]
#lastPREFIX
OldPrefix=        # ---> Prefix of the last created files during the simulation = JobName + date of the last period. Use
#Warning : OldPrefix not used anymore from libIGCM_v2.5.
#Compute date of loop
PeriodDateBegin=   #  --->start date of the next period to be simulated
PeriodDateEnd=     # ---> end date of the next period to be simulated
CumulPeriod=       # ---> number of already simulated periods
# State of Job "Start", "Running", "OnQueue", "Completed"
PeriodState="Completed"

SubmitPath=   # ---> Submission directory
```

- PostProcessing : returns information about the post processing status

```
[PostProcessing]
TimeSeriesRunning=n   # ---> indicates if the timeSeries are running
TimeSeriesCompleted=20091231   # ---> indicates the date of the last TimeSerie produced by the post processing
```

- Log : returns technical (run-time) information such as the size of your executable and the execution time of each integration step.

```
[Log]
# Executables Size
LastExeSize=()

#---------------------------------
# CumulPeriod | PeriodDateBegin |   PeriodDateEnd |        RunDateBegin |          RunDateEnd |     RealCpuTime |     Us
#           1 |        20000101 |        20000131 | 2013-02-15T16:14:15 | 2013-02-15T16:27:34 |       798.33000 |
#           2 |        20000201 |        20000228 | 2013-02-15T16:27:46 | 2013-02-15T16:39:44 |       718.16000 |
```

If the run.card file indicates a problem at the end of the simulation, you can check your Script_Output file for more details. See more details here.

2.3.3. Script_Output_JobName

A Script_Output_JobName file is created for each job executed. It contains the simulation job output log (list of the executed scripts, management of the I/O scripts).
This file contains mainly three parts :

- copying and handling of input and parameters files
- running the model
- copying of outputs files and launching of post processing steps (rebuild if necessary and pack)

These three parts are defined as below :

```
########################################
#       ANOTHER GREAT SIMULATION       #
########################################

1st part (copying and handling of the input and parameter files)

########################################
#      DIR BEFORE RUN EXECUTION        #
########################################

2nd part (running the model)

########################################
#       DIR AFTER RUN EXECUTION        #
########################################

3rd part (copying of outputs files and launching of post processing steps (rebuild and pack))
```

2.4. The output files

The output files are stored on file servers. Their names follow a standardized nomenclature: IGCM_OUT/TagName/[SpaceName]/[ExperimentName]/JobName/ in different subdirectories for each "Output" and "Analyse" component (e.g. ATM/Output, ATM/Analyse), DEBUG, RESTART, ATLAS and MONITORING. File server where outputs will be stored depends on the SpaceName choose for the simulation. Remember : TEST mode is a specific case which deactivate pack. Prior to the packs execution, this directory structure is stored

- on the $CCCSCRATCHDIR at TGCC
- on the $SCRATCH at IDRIS

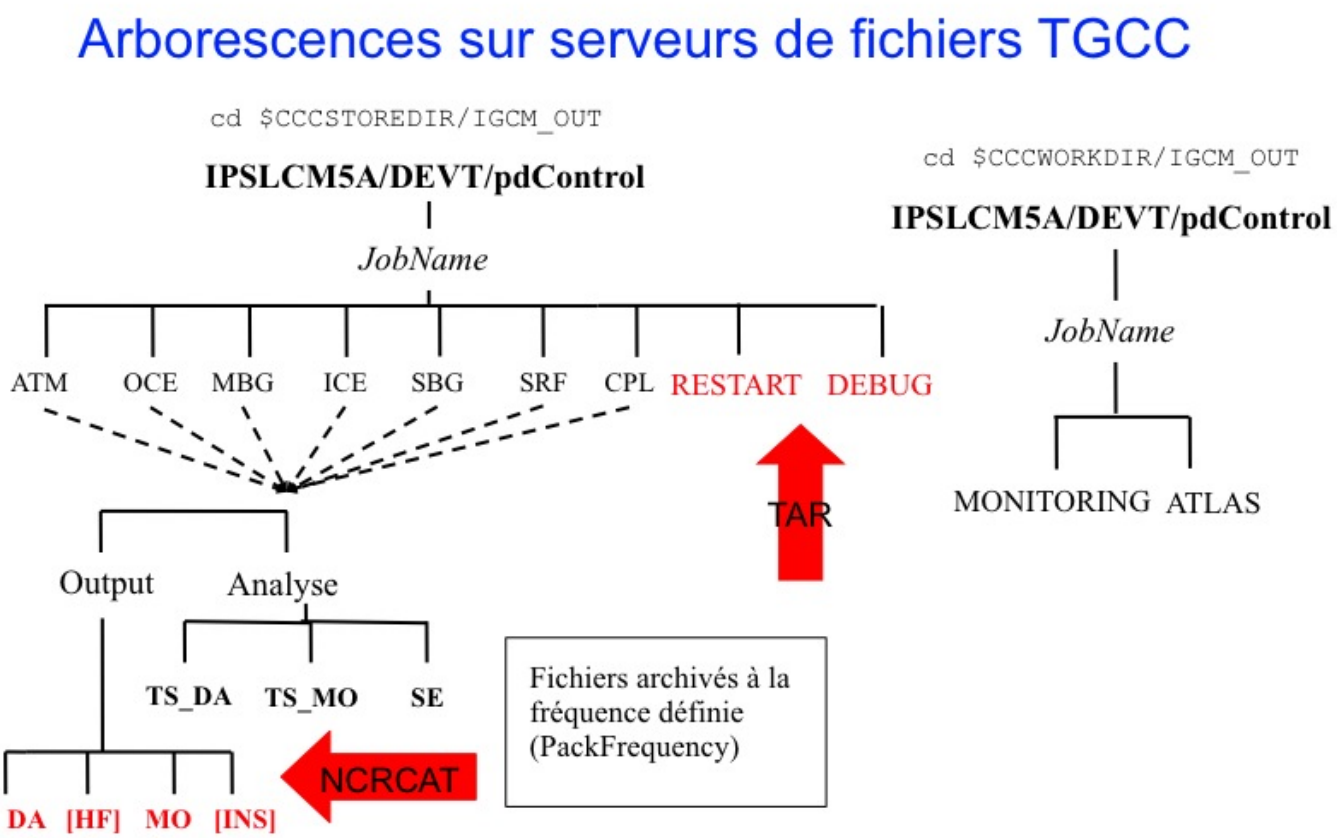
After the packs execution (see diagram below), this directory structure is stored

- on the $CCCSTOREDIR at TGCC
- on the $STORE at IDRIS

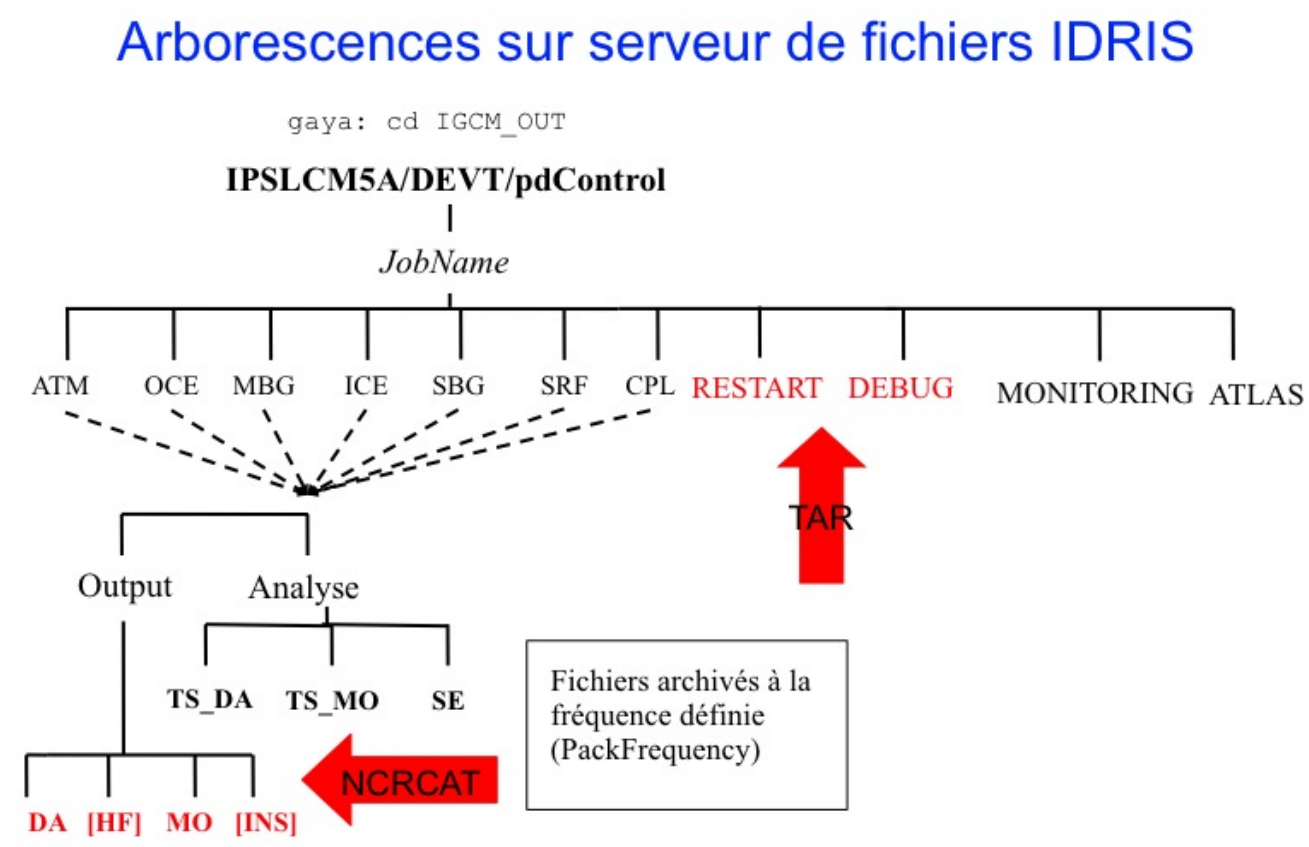
To summarize:

SpaceName	Computing Center	File Server
TEST	TGCC	$CCCSCRATCHDIR
TEST	IDRIS	$SCRATCH
DEVT / PROD	TGCC	$CCCSTOREDIR
DEVT / PROD	IDRIS	$STORE

2.4.1. Here is the storage directory structure of the output files produced at TGCC



2.4.2. Here is the storage directory structure of the output files produced at IDRIS



2.5. Debug/ directory

A Debug/ directory is created if the simulation crashed. This directory contains text files from each of the model components to help you finding reasons for the crash. See also the chapter on monitoring and debugging.

2.6. How to continue or restart a simulation?

1. If you want to continue an existing and finished simulation
   - change the simulation end date in the `config.card` file. Do not change the simulation start date.
   - launch ../../../libIGCM/clean_or_continue.job from the experment folder. This script will update parmeters in run.def file such as PeriodState. The script will also change the name of the Script file in the main job Script_Output_JobName_.0000X to correspond to the new period.
   - launch the main job as before
2. If your simulation has stopped in the middle of an execution and you want to restart it, some cleaning must be done using the script clean_or_continue.job:
   - launch ../../../libIGCM/clean_or_continue.job from the experment folder
   - launch the main job as before

Note: **clean_or_continue.job** was previously named **clean_PeriodLength.job**. If in your version of libIGCM, there is no clean_or_continue.job, you can use clean_PeriodLength.job in the same way.

---

3. Simulation - Post processing

3.1. Post processing in config.card

You must specify in *config.card* the kind and frequency of the post processing.

```
#============================================================================
#D-- Post -
[Post]
#D- Do we rebuild parallel output, this flag determines
#D- frequency of rebuild submission (use NONE for DRYRUN=3)
RebuildFrequency=NONE
#D- frequency of pack post-treatment : DEBUG, RESTART, Output
PackFrequency=1Y
#D- If you want to produce time series, this flag determines
#D- frequency of post-processing submission (NONE if you don't want)
TimeSeriesFrequency=10Y
#D- If you want to produce seasonal average, this flag determines
#D- the period of this average (NONE if you don't want)
SeasonalFrequency=10Y
#D- Offset for seasonal average first start dates ; same unit as SeasonalFrequency
#D- Useful if you do not want to consider the first X simulation's years
SeasonalFrequencyOffset=0
#============================================================================
```

If no post processing is desired you must specify **NONE** for the TimeSeriesFrequency and SeasonalFrequency frequencies.

3.2. Rebuild

For almost all configurations, the rebuild phase is not needed. Single files are directly written in parallel by XIOS

- `rebuild` is a tool which allows you to combine several files created by a parallel program (sub domains) to a single file. Note that if you use XIOS as output library (XIOS is used in v6 configurations), the rebuild step could not be needed : it depends on the writing mode (parallel or not, server or not) you have activated.
- `rebuild` is available with IOIPSL package. See ■http://forge.ipsl.jussieu.fr/igcmg/browser/IOIPSL/trunk/tools (it can therefore be distributed via modipsl)
- `rebuild` is installed on the IDRIS and TGCC front-end machines.

  `rebuild` is only needed and launched in following cases :
  - IOIPSL is used as output writing library and you run in parallel mode.
  - XIOS is used as output writing library, you run in parallel mode, you run in XIOS attached mode (or server mode with several servers) and you activate multiple_file XIOS mode.
- If needed, `rebuild` is automatically called at the RebuildFrequency frequency and it is usually the very first step of post processing. Specifying NONE for RebuildFrequency will start the file combining on the computing machine instead of doing it on the post processing machine. This is strongly discouraged. RebuildFrequency=1Y indicates the frequency of running REBUILD. The files to be combined by `rebuild` are stored on a buffer space $CCCSCRATCHDIR/IGCM_OUT/../JobName/REBUILD/ at TGCC and $WORK/IGCM_OUT/.../JobName/REBUILD at IDRIS. Note: if JobType=DEV the parameter is forced to have the PeriodLength value.
- RebuildFromArchive=NONE is the option to be used on all machines. The REBUILD job first looks for the files to be assembled on the buffer space. Then it assembles them (*rebuild*), applies requested Patchs and stores them in the usual COMP/Output/MO or COMP/Output/DA directories for monthly or daily files of the COMP component (OCE, ICE, ATM, SRF, ...). Note: REBUILD does the ordering of other post processing jobs ran by the create_ts.job and create_se.job jobs.

3.3. Concatenation of "PACK" outputs

The model outputs are concatenated before being stored on archive servers. The concatenation frequency is set by the **PackFrequency** parameter (NONE means no concatenation, not recommended). If this parameter is not set the rebuild frequency RebuildFrequency is used. This packing step is performed by the PACKRESTART, PACKDEBUG(started by the main job) and PACKOUTPUT (started by the rebuild job or the main job) jobs.

3.3.1. How are the different kinds of output files treated ?

All files listed below are archived or concatenated at the same frequency (PackFrequency)

- **Debug** : those files are archived and grouped in a single file with the `tar` command. They are then stored in the IGCM_OUT/TagName/.../JobName/DEBUG/ directory.

- **Restart** : those files are archived and grouped in a single file with the `tar` command. They are then stored in the IGCM_OUT/TagName/.../JobName/RESTART/ directory.
- **Output** : those files are concatenated by type (histmth, histday ...) with the `ncrcat` command in the IGCM_OUT/TagName/.../JobName/_comp_/Output/ directories.

3.3.2. Specification of output pack frequency per file (available from libIGCM rev 1603)

In order to reduce the number of inodes, it is possible to specify the frequency of output packing per file. The syntax to do that is in the 4th column of OutputFiles section of the component.card, for example as follows in lmdz.card :

```
[!OutputFiles]
List=   (histmth.nc,       ${R_OUT_ATM_O_M}/${PREFIX}_1M_histmth.nc, Post_1M_histmth, 100Y),       \
       (histday.nc,       ${R_OUT_ATM_O_D}/${PREFIX}_1D_histday.nc, Post_1D_histday, 10Y),                \

...
```

In this example, histmth files will be packed every 100 years and histday files will be packed every 10 years.

The pack frequency you defined in config.card is the frequency of pack by default, that means if a specific frequency of pack is specified for a file in a component.card, this file will be packed at the specific frequency whereas all other files will be packed at global pack frequency (specified in config.card) and in this case, the frequency pack (from the config.card) is the frequency the pack_output job will be launched at.

There is a constraint to use this fonctionality : the Packfrequency you defined in config.card must be greater or equal to the pack frequencies you specified for each type of file in component.card, otherwise the computing job will be stopped (with an explicit error message).

3.3.3. Surpack functionality (available from libIGCM rev 1603)

A surpack mode functionality is available through the use of pack_output.job in order to (sur) pack output file which have been already packed to generate bigger files. To enable this functionality, you have to put "surpack_mode=y" (default value is n). The way to use is similar to restart post-processing pack_output jobs, as indicated here : ■http://forge.ipsl.jussieu.fr/igcmg_doc/wiki/Doc/CheckDebug#RestartPack_output. You can either use a global pack frequency in config.card or specific pack frequency per file, as explained above.

3.4. Time Series

A Time Series is a file which contains a single variable over the whole simulation period (ChunckJob2D = NONE) or for a shorter period for 2D (ChunckJob2D = 100Y) or 3D (ChunckJob3D = 50Y) variables.

- The write frequency is defined in the *config.card* file: TimeSeriesFrequency=10Y indicates that the time series will be written every 10 years and for 10-year periods.
- The Time Series are set in the COMP/*.card files by the TimeSeriesVars2D and TimeSeriesVars3D options.

Example for lmdz :

```
45  [OutputFiles]
46  List=   (histmth.nc,      ${R_OUT_ATM_O_M}/${PREFIX}_1M_histmth.nc,      Post_1M_histmth), \
...
53  [Post_1M_histmth]
54  Patches= ()
55  GatherWithInternal = (lon, lat, presnivs, time_counter, aire)
56  TimeSeriesVars2D = (bils, cldh, ... )
57  ChunckJob2D = NONE
58  TimeSeriesVars3D = ()
59  ChunckJob3D = NONE
```

- Each output file (section [OutputFiles]) is related to a post processing job: Post_1M_histmth in the example.
- Post_1M_histmth is a section (starting by "[Post_1M_histmth]")
- This section contains the variables : Patches= , GatherWithInternal = , TimeSeriesVars2D = , ChunckJob2D , TimeSeriesVars3D and ChunckJob3D.
  - Patches= (), this functionality is not used anymore. Before, the patch was applied to the output file. The old patches can be found here: ■libIGCM_post Different Patches can be applied consecutively.
  - GatherWithInternal = (lon, lat, presnivs, time_counter, aire) These are the variables to be extracted from the initial file and to be stored with the Time Series variable.

- TimeSeriesVars2D/3D = those are variable lists of time series to create.
- ChunckJob2D/3D = if the simulation is too long you can cut the time series into x-year long chunks (ChunckJob2D=50Y for example).
- ChunckJob2D = OFF (or ChunckJob3D = OFF) means no Times Series.

The Time Series coming from monthly (or daily) output files are stored on the archive server in the IGCM_OUT/TagName/[SpaceName]/[ExperimentName]/JobName/Composante/Analyse/TS_MO and TS_DA directories.

You can add or remove variables to the TimeSeries lists according to your needs.

There are as many time series jobs as there are ChunckJob3D values. This can result in a number of create_ts jobs (automatically started by the computing sequence).

3.5. Monitoring and intermonitoring

The monitoring is a web-interface tool that visualizes the global mean over time for a set up of key variables. Access the monitoring using the esgf/thredds address for your machine ending with yourlogin/TagName/SpaceName/JobName/MONITORING. If you have a new account, you might need to contact the assistant team at the computer center to activate your write access to esgf/thredds.

- esgf/thredds at TGCC:
  - ■https://thredds-su.ipsl.fr/thredds/catalog/tgcc_thredds/work/catalog.html
- esgf/thredds at IDRIS:
  - ■https://thredds-su.ipsl.fr/thredds/catalog/idris_thredds/work/catalog.html

The key variables plotted in the monitoring are computed using Time Series values. The monitoring is updated at the TimeSerieFrequency set in *config.card* if the time series were successfully done. This allows you to monitor a simulation. By monitoring your simulations you can detect anomalies and evaluate the impact of changes you have made. We suggest to create a tab in your browser allowing you to frequently monitor your simulation. If a few key variables start looking suspicious you might want to stop your simulation. By doing so, you will save computing time.

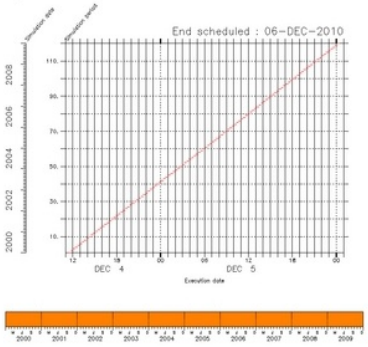
Here is an example for the IPSLCM5A coupled model and a 10-year period. Once you are in yourlogin/TagName/SpaceName/JobName/MONITORING, you have to click on index.html. The first tab called **Analysis Cards** gives a summary of dates and execution times obtained from the *config.card* and *run.card* files. The second tab called **Monitoring Board** presents a monitoring table for the key variables (selecting one or more model components is optional).

BAL1210 monitoring

at 2010-12-06 05:38:15

Cards Analysis | Monitoring Board | About

- Progress of the simulation



- run.card
  - Open at original format
  - Open as sortable table
- config.card
  - Open at original format
- Directories access
  - files
  - images
  - progress

- Simulation date summary

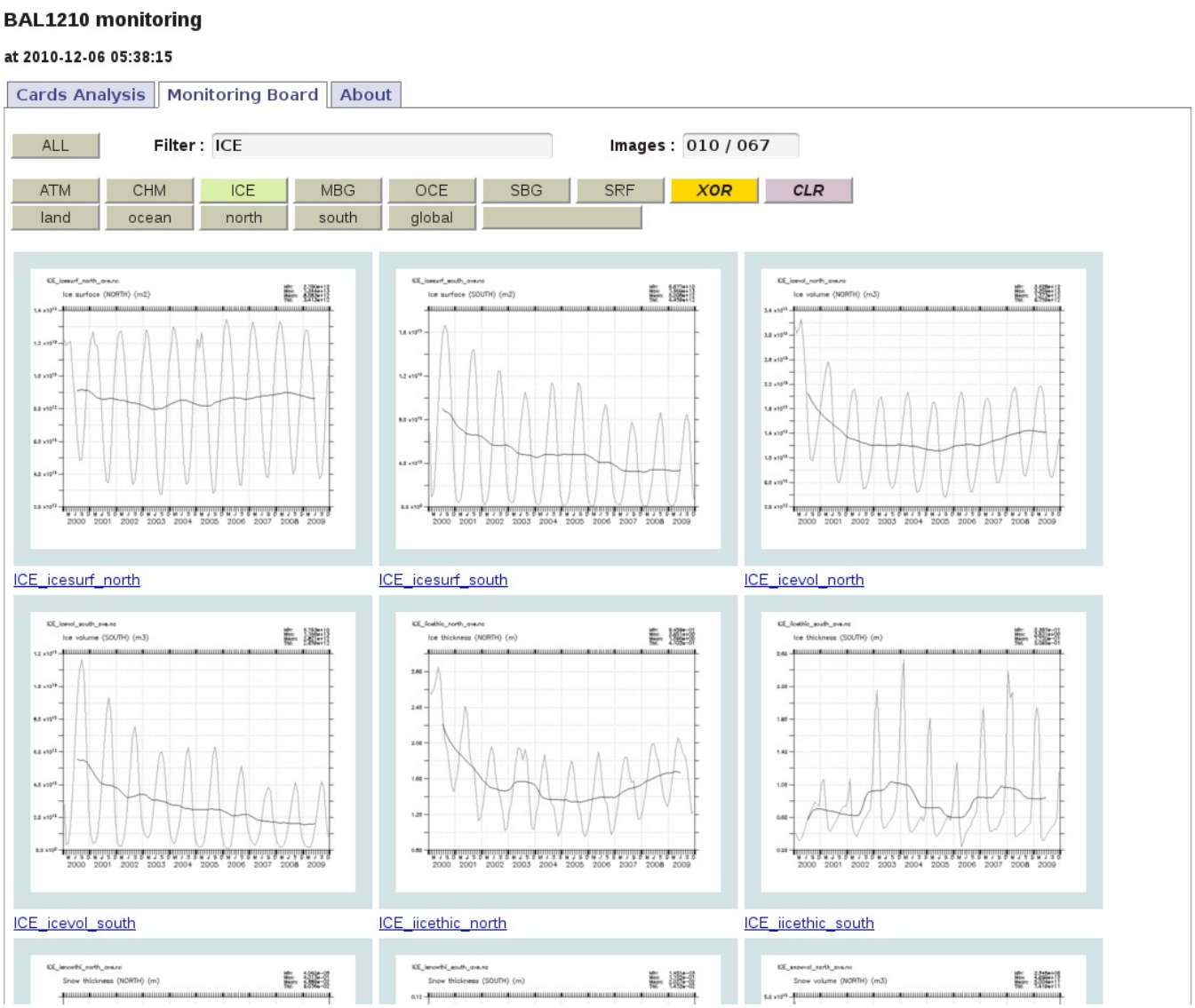
CalendarType	DateBegin	DateEnd	DateCurrent	PeriodLength	CumulPeriod
noleap	2000-01-01	2009-12-31	2010-01-31	1M	121

- Real Cpu time summary

min	max	average
967.67	1086.38	1050.64

- User Cpu time summary

min	max	average
0.94	3.28	1.20



- The diagnostics of each experiment are stored in the MONITORING directory following the IGCM_OUT/TagName/SpaceName/ExperimentName/MONITORING nomenclature (on the $CCCWORKDIR at TGCC and on $WORK at IDRIS).
- The diagnostics starts automatically after the Time Series are created. See the diagram on the computing sequence.

3.5.1. Adding a variable to the monitoring

You can add or change the variables to be monitored by editing the configuration files of the monitoring. Those files are defined by default for each component.

By default, the monitoring is defined here: `~compte_commun/atlas` For example for LMDZ : `monitoring01_lmdz_LMD9695.cfg`

You can change the monitoring by creating a `POST` directory which is part of your configuration. Copy a `.cfg` file and change it the way you want. You will find examples in ■special post processing

Be careful : to calculate a variable from two variables you must define it within parenthesis :

```
#-------------------------------------------------------------------------------------------------------------------
#  field | files patterns | files additionnal | operations | title | units | calcul of area
#-------------------------------------------------------------------------------------------------------------------
nettop_global | "tops topl"                 | "" | "(tops[d=1]-topl[d=2])" | "TOA. total heat flux (GLOBAL)"        | "W
tops_global   | "tops"                      | "" | "tops[d=1]"  | "tops Global " | "W/m²" | "aire[d=1]"
```

- field can not have the same name than the variable used for the monitoring (--> it creates an error) (tops_global / tops in the example)

- units need to be complete (if you have "" for units it creates an error --> you need to indicate one character)
- if "aire" or "area" is not defined in the TS file, you can add an additional file to find this variable (and in this case it changes the number of files for the last parameter)

```
#-------------------------------------------------------------------------------------------------------------------
#  field | files patterns | files additionnal | operations | title | units | calcul of area
#-------------------------------------------------------------------------------------------------------------------
nettop_global | "tops topl"                  | LMDZ4.0_9695_grid.nc | "(tops[d=1]-topl[d=2])" | "TOA. total heat flux (
```

by default monitoring is based on TS_MO files, but it is possible to change the frequency by adding **FreqTS** in the configuration file monitoring*.cfg

```
FreqTS=DA
#-------------------------------------------------------------------------------------------------------------------
#  field | files patterns | files additionnal | operations | title | units | calcul of area
#-------------------------------------------------------------------------------------------------------------------
nettop_global | "tops topl"                  | LMDZ4.0_9695_grid.nc | "(tops[d=1]-topl[d=2])" | "TOA. total heat flux (
```

3.5.2. Inter Monitoring

- To simultaneously monitor various simulations, a web application intermonitoring has been created to monitor the evolution of different variables at once.

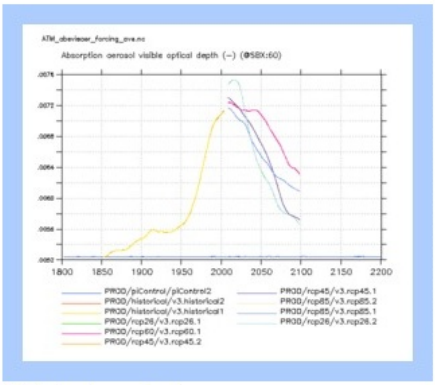
Monitoring comparison: piControl2 vs v3.historical2 vs v3.historical1 vs v3.rcp26.1 vs v3.rcp60.1 vs v3.rcp45.2 vs v3.rcp45.1 vs v3.rcp85.2 vs v3.rcp85.1 vs v3.rcp26.2

at 2011-05-29 21:49:27

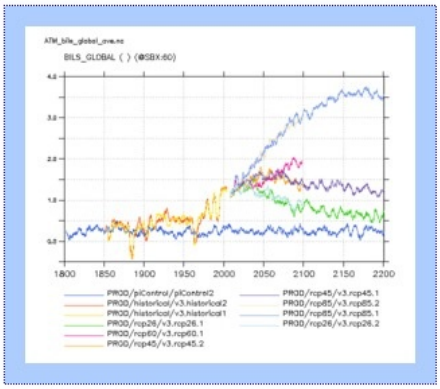
ALL **Filter :** .* **Images :** 074 / 074

ATM | CHM | ICE | MBG | OCE | SBG | SRF | XOR | CLR

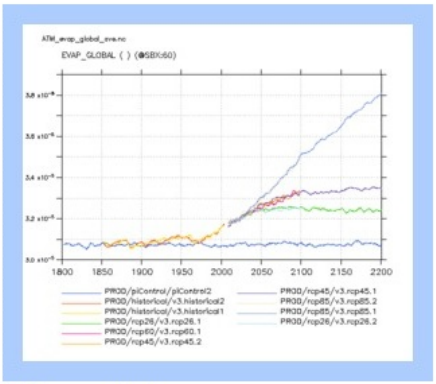
land | ocean | north | south | global | forcing



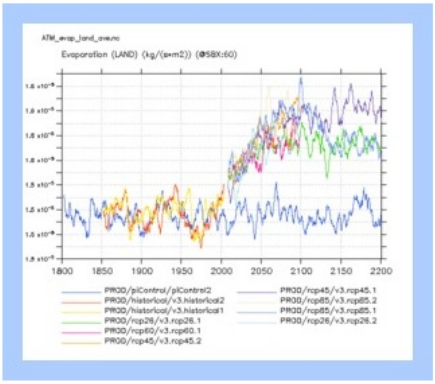
ATM_absvisaer_forcing_ave



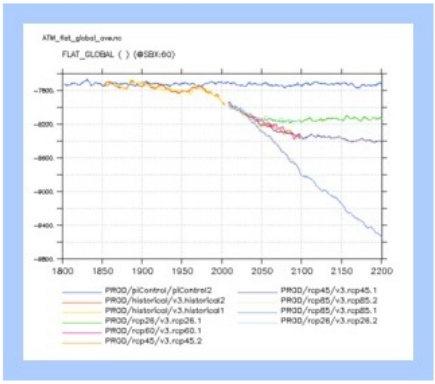
ATM_bils_global_ave



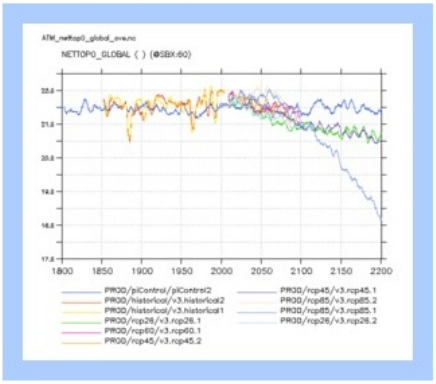
ATM_evap_global_ave



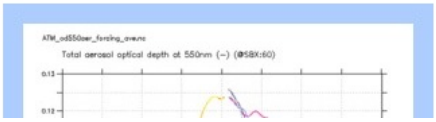
ATM_evap_land_ave

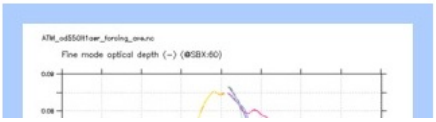


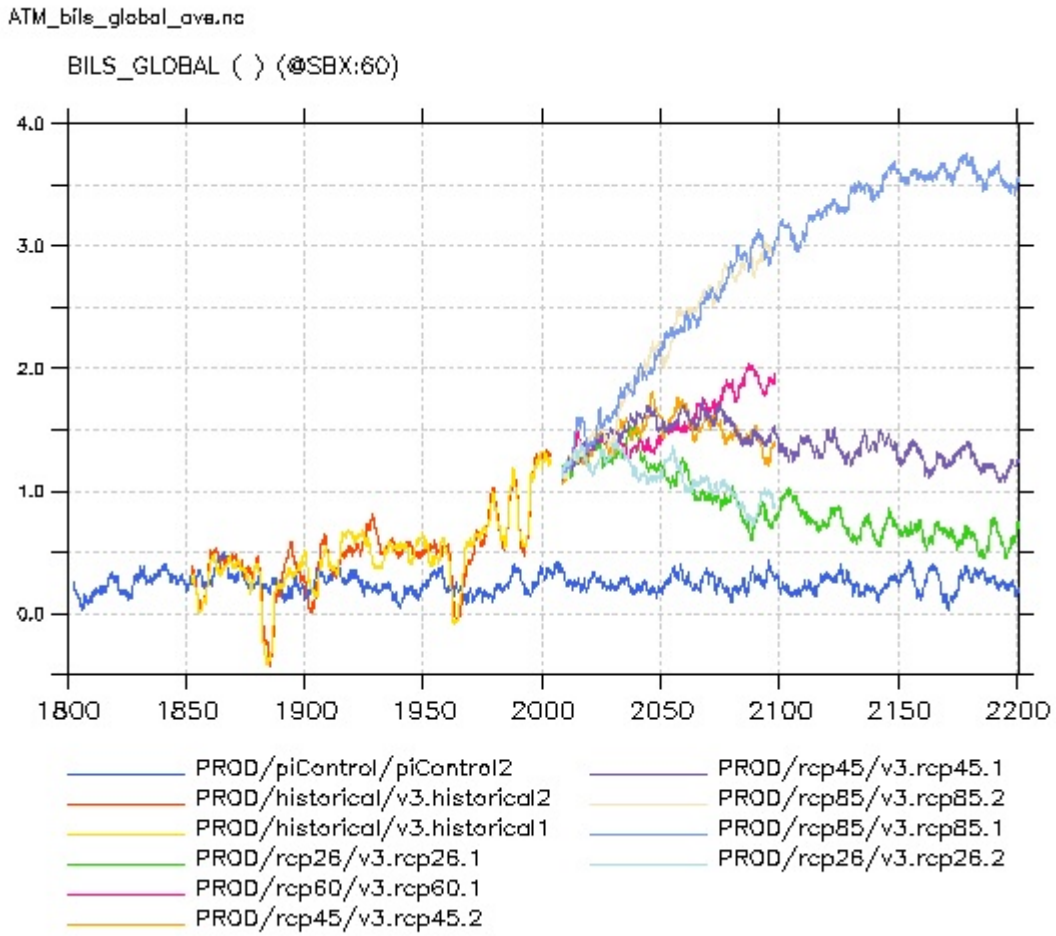
ATM_flat_global_ave



ATM_nettop0_global_ave







- This application is available on ■ICMC Web Applications / intermonitoring.
- A video tutorial is available ■here.

  In the **Step 1** tab, specify URLs which give access to the MONITORING directory
  - see examples given by the application.

3.5.3. Mini how to use the intermonitoring

Go to ■http://webservices.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: 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 or more simulations (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 "plot01:Time series" and click on Validate. Then click on the button below called Prepare and run the ferret script.
- Step 5: 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.

3.5.4. How to save the intermonitoring permanently

The plots done by the intermonitoring will be kept 30 days. During these days you can visualize using the same link the plots done. To keep them permanently, do as follow:

- Create the intermonitoring using the webservices interface (see mini howto or audio guide above)
- Save the .jnl script and the .bash script created by the webservices to your computer, together in the same directory.

- Edit the .bash and modify as follows :
  - source of ferret configuration files. Here are examples to uncomment if needed :

```
#  # IPSL (webservices)
#  # IDRIS (jeanzay)
#. /home/users/brock/.atlas_env_asterix_bash      # LSCE (asterix)

# add for TGCC (irene)
module unload ferret
module load pyferret
module load imagemagick
. ~igcmg/MachineEnvironment/irene/env_atlas_irene


# add for ciclad:
#====================================
module unload ferret
module unload python
module unload pyferret
export PATH="/home/brocksce/anaconda3/bin:$PATH"
module load ferret/7.4.3
. /home/brocksce/.env_fast_atlas.sh
```

  - define the name and the path where you saved the .jnl script.

```
scriptname=./intermonit_CM6.jnl
```

  - uncomment and adapt the last line in the script with copy to esgf/thredds. For example at ciclad :

```
# for ciclad
cp -fR ${scriptname%%.jnl}_prod /prodigfs/ipslfs/http/dods/web/html/login/INTERMONITORING

# or for irene:
cp -fR ${scriptname%%.jnl}_prod $CCCWORKDIR/../../thredds/login/INTERMONITORING
```

- On Ciclad you will have an error on **-batch** option of pyferret but it will still work
- For irene you need to edit the .jnl script as follows:
  - change the path of the netCDF file to plot, inserting the SpaceName, ExperimentName, JobName related to your run:

```
use "$CCCWORKDIR/../../thredds/login/TagName/SpaceName/ExperimentName/JobName/MONITORING/files/($FILE)"
```

- change permissions on .bash

```
chmod 755 *.bash
```

- Run the .bash script. A new directory will appear on your computer. This is the directory that is also copied to esgf/thredds.
- The intermonitoring is now on esgf/thredds and you can keep the link permanently. For example for irene, you can find intermonitoring following the link below, where XXXXX is the name of the .jnl file (without .jnl) :

```
http://vesg.ipsl.upmc.fr/thredds/fileServer/work/login/INTERMONITORING/XXXXX_prod/index.html
```

3.6. Seasonal means

The SE (seasonal means) files contain averages for each month of the year (jan, feb,...) for a frequency defined in the *config.card* files

- SeasonalFrequency=10Y The seasonal means will be computed every 10 years.
- SeasonalFrequencyOffset=0 The number of years to be skipped for calculating seasonal means.

- The SE files will automatically start at the SeasonalFrequency=10Y frequency (pay attention to SeasonalFrequencyOffset=0) when the last parameter in the file of the '[OutputFiles]' section is not NONE.
- All files with a requested Post are then averaged within the ncra script before being stored in the directory: IGCM_OUT/IPSLCM5A/DEVT/pdControl/MyExp/ATM/Analyse/SE. There is one file per SeasonalFrequency=10Y

3.7. Atlas

- The atlas is a result of the automatic post processing, which will creates a collection of plots presented as a web tree. Each plot is available as image and pdf file. The plots are made with the software ferret and the FAST and ATLAS libraries. More information on ferret and on those libraries can be found here ■http://wiki.ipsl.jussieu.fr/IGCMG/Outils/ferret/Atlas
- By default, automatic atlas are disabled. To activate automatic atlas, you have to specify `config_Post_AtlasIPSL=TRUE` in Post part of *config.card*
- Here is an example of atlas for the coupled IPSLCM5A available on esgf/thredds : ■ATM
- There are at least 8 directories with Atlas for the coupled model. They are based on atlas_composante.cfg files. You can look at them on the file servers in the IGCM_OUT/IPSLCM5A/DEVT/pdControl/MyExp/ATLAS directories.
- The script libraries fast/atlas are installed in shared directories on every machine.
- Note that it is also possible to generate CLIMAF atlas : you have to launch it on your own. See ■documentation

3.8. Storing files like ATLAS, MONITORING and ANALYSE

The files produced by ATLAS, MONITORING, Time series and Seasonal means are stored in the following directories:

- **ANALYSE**: copied to JobName/_comp_/Analyse on the file server **store** or **ergon** (to be modified)
- **MONITORING**: copied to JobName/MONITORING on the file server **work** or **ergon** (to be modified)
- **ATLAS**: copied to JobName/Atlas on the file server **work** or **ergon** (to be modified)

They are available through esgf/thredds server at IDRIS and at TGCC.

- at TGCC with the new one called esgf/thredds:
  - ■https://vesg.ipsl.upmc.fr/thredds/catalog/catalog.html click on work_thredds, click on your login and... for ATLAS and MONITORING
  - ■https://vesg.ipsl.upmc.fr/thredds/catalog/catalog.html click on store_thredds, click on your login and... for Analyse files (TS or SE)
- at IDRIS: ■https://prodn.idris.fr/thredds

3.9. How to check that the post processing was successful

The post processing output log files are :

- on JeanZay: `$WORK/IGCM_OUT/TagName/.../JobName/Out`
- on Irene: `$CCCSCRATCHDIR/IGCM_OUT/TagName/.../JobName/Out`

In these directories, you find the job output files for following jobs: `rebuild`, `pack*`, `ts`, `se`, `atlas`, `monitoring` .

Scripts to transfer data on esgf/thredds are run at the end of the monitoring job or at the end of each atlas job.

If you need to run post-processing jobs while the main job is finished, see Start or restart post processing jobs page for details.