Manual PlaSim

Marte Hofsteenge, Claudia Wieners, Michiel Baatsen, Jasper de Jong, Tim Hermans

March 18, 2024

Contents

Useful links

- Plasim User Guide
- PlaSim Reference Manual
- Description Hamburg LSG ocean circulation model
- Manual for the LSG ocean model

1 Working on the Gemini Server

1.1 Logging in on Gemini

During this project you will run the PlaSim model and analyze its output on the Gemini high-performance cluster[1]. On this cluster, which is essentially a collection of computers, you will be able to run the model in parallel on different computer cores, which can shorten the simulation time significantly from running the model on your own laptop. You can access this high-performance cluster with your personal solis-id as follows:

Mac users
Open the Terminal, an application through which you can give commands to your computer. With the following command, you canlogin on Gemini:

```
ssh -Y <your solis-id>@gemini.science.uu.nl
```

replacing `<your solis-id>` with your solis-id. Next, enter your solis password to access your home directory on Gemini.

Windows users
If you are working on a Windows computer, we recommend giving your commands through the MobaXterm programme, which provides several features to conveniently work on a remote server like Gemini and to easily upload and download data from the server. Install and open MobaXterm. Then, navigate to MobaXterm → Go to Sessions → New session → Fill in as remote host: **gemini.science.uu.nl** → fill in your solis-id as username and click OK. Give your solis password and you have access to your home directory on Gemini.

Troubleshooting: If `ssh -Y` does not work, try using `ssh -X`.

1.2 Working in the terminal

Once logged in on Gemini, you can use linux commands to do things like navigating to directories and editing, copying and moving files to other directories. This is very similar to what you are doing when you are using Finder or Windows Explorer to view your documents. Table 1 provides a list of linux commands that will be useful when using PlaSim. To get started, try out the following commands from Table 1:

- Use the command `pwd` to view the path to your current working directory. This is your home directory: the directory you are located in directly after logging in. Your home directory is called `/nethome/<your solis-id>`.
- Use the command `mkdir new_directory` to make a new directory called `new_directory` within your home directory. Next, give the commands `mkdir new_directory/subdirectory_a` and `mkdir new_directory/subdirectory_b`. This will create two new directories within the directory `new_directory` that you just made.
- To check what has happened, take a look at the content of your home directory using `ls`. Can you find the new directory that you made? Enter it using `cd new_directory`.
- Use `ls` again to view the contents of the directory `new_directory`. Then, use `cd` to navigate to `subdirectory_a`. What is the path to this subdirectory?

[1] see `https://www.hpc.iastate.edu/guides/introduction-to-hpc-clusters/what-is-an-hpc-cluster` for some background information on high-performance clusters.

- Next, navigate to `subdirectory_b`: to move one directory up, use `cd ..`, and to enter `subdirectory_b`, use `cd subdirectory_b`. You can also combine these commands to go directly from `subdirectory_a` to `subdirectory_b`, using: `cd ../subdirectory_b`.
- Finally, let's return to the home directory using `cd`. To move one directory up, use `cd ..`, to move two directories up, use `cd ../..`, et cetera. You can also use `cd` without specifying a path or use `cd ~` to go to your home directory directly.

Command	
`cd <directory>/`	change directory
`cd ..`	move one directory up
`cd` (or: `cd ~`)	move to your home directory
`pwd`	give path of current working directory
`ls`	list contents of a directory
`ll`	list contents of a directory plus file details
`cp <file1> <directory>/<file2>`	copy a file (file1) from current to another directory, where it gets the name file2. (If you omit the directory path, the new file is in the same folder; if you omit the new name, the file will have the same name in the new directory.)
`cp -r <directory1> <directory2>`	copy a directory to another directory (similar to cp, but if you apply it to a directory you need the `-r`)
`mv <file1> <directory>/<file2>`	move file to another directory with new name (see also cp)
`mkdir <directory>`	create a new directory
`rm <file>`	remove a file
`rm -r <directory>/`	remove a directory
`scp <host>:<dirhost/file> <dir>`	copy file from a remote server to your PC
`nedit <textfile>`	open and edit a text file
`more <textfile>`	show contents of a text file
`df -h`	show available data storage
`du -sh`	show amount of data in current folder
`du -sh <directory>`	show amount of data in specified directory
`ncdump -h <file.nc>`	show header of data in netcdf file
`display <figure.png>`	display an image
`module list`	check which modules are loaded to your account
`module avail`	check which modules are available on Gemini
`module load <name>`	load a module (replace `<name>` by true name)
`module purge`	unload all modules

Table 1: Some relevant linux commands.

With commands like `cd` and `cp` you often do not need to fill in the whole name of the directory or file: You can use a wildcard (type *) in order to "leave a gap" in the name of the directory or file or press the tab button to complete a name automatically. For example, if you want to delete both `subdirectory_a` and `subdirectory_b`, and your current working directory is `new_directory`, you can do so by typing: `rm -r subd*` (make sure you have no other subdirectories also starting with "subd" which you want to keep!).

The `scp` command is used from a terminal on your computer that is not logged in to Gemini. The command to copy a file from Gemini to your computer could look like the following:
`scp <your solis-id>@gemini.science.uu.nl:/nethome/<your solis-id>/file destination_directory/`
where `destination_directory` is the path of the directory in your computer that you want to copy

the file to. If your current working directory is also the `destination_directory`, you can simply replace `destination_directory/` by a dot).

1.3 Shared project space

In this project, you will use (subdirectories in) your home directory to install PlaSim and set up simulations. However, running simulations and storing the output will be done in a shared directory outside your home directory:

```
/science/projects/ESM
```

This directory offers more storage space and enables you to work together with other students. To go to this directory, use the command `cd /science/projects/ESM`. Like before, to get back to your home directory, use `cd /nethome/<your solis-id>` , `cd` or `cd ~` .

1.4 Connecting to Gemini using FileZilla

Working on Gemini in the terminal is a little bit less intuitive than using Finder or Windows Explorer on your own computer, because you cannot directly see how your directories and documents are organized. However, with computer programmes like FileZilla you can visualize your directories on Gemini too. To do that, install FileZilla from `https://filezilla-project.org/`. When you open FileZilla you will see the directories on your own computer on the left side of the window.
To connect to Gemini, open the Site Manager by pressing the top-left icon above 'Host'. Click on 'New site', and fill in the form as in the example in Figure 1: under 'Protocol', choose 'SFTP - SSH File Transfer Protocol' and under 'Host' fill in gemini.science.uu.nl. Choose the 'Normal' Logon Type, and fill in your solis-id under 'User' and your solis password under 'Password' Finally, click 'Connect'. Press 'OK' if you are asked to trust the host and carry on connecting.
After connecting to Gemini, you will now see your home directory on Gemini on the right hand side of the FileZilla window. You will see that it contains the directories that you previously created, as well as some hidden directories starting with a dot. To also view the shared projected space (see Section 1.3), click on the path next to 'Remote Site', fill in `/science/projects/ESM`, and press enter.
Like the terminal and MobaXTerm, you can use FileZilla to navigate through directories or make new directories. Moreover, you can **easily transfer directories or files from Gemini to your own computer** (as with the `scp` command) by simply dragging them from the right side of the window to the left side of the window, and vice versa.
While FileZilla is a nice tool to visualize your directories and transfer files between Gemini and your computer, you will still need to use the terminal for more complex commands.

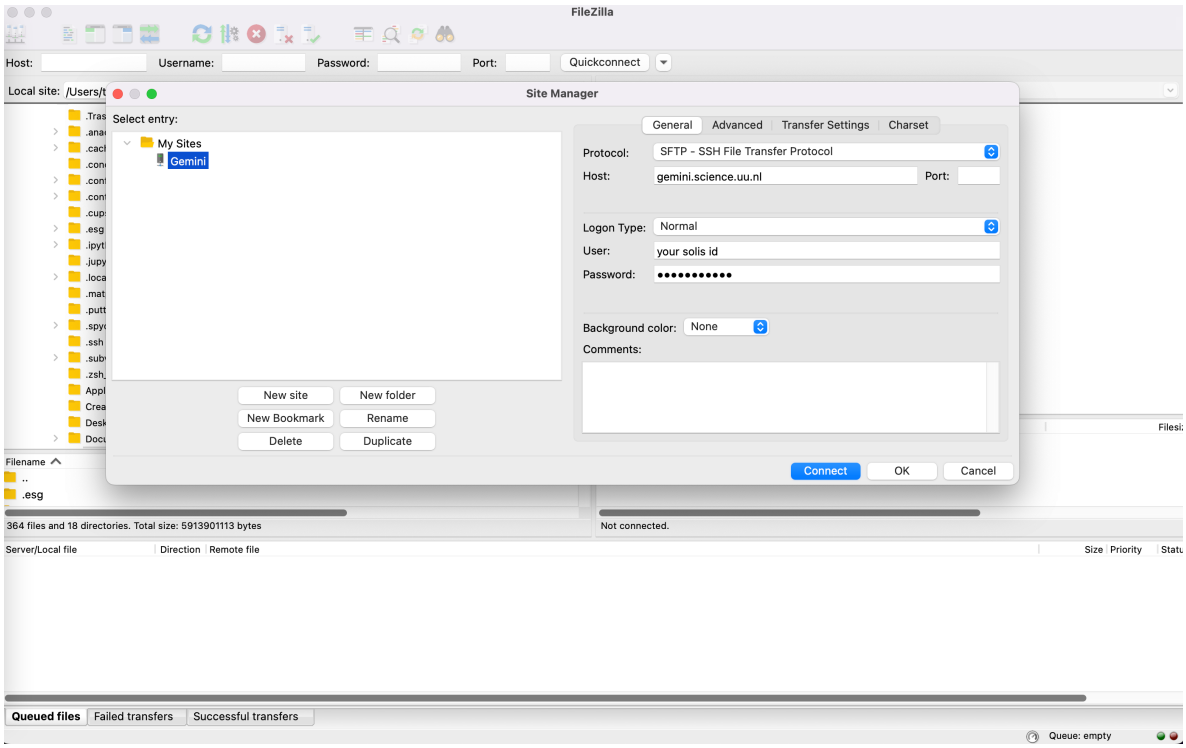


Figure 1: Connecting to Gemini in FileZilla.

2 Installing and running Plasim

In this section you will learn how to install Plasim and how to do the first simulations. Later on you will see how to do more advanced things, such as running PlaSim with an active ocean component (as opposed to using a constant sea surface temperature).

2.1 Installing PlaSim

Before you can install and use the PlaSim model, you need to load some software to your Gemini account that is required to run the model. You can do so using the *module* system of the Gemini server (see `https://hpc-wiki.info/hpc/Modules` for additional explanation). Use the following commands to load two modules:.

```
module load mpi/openmpi-x86_64
module load sge/2011
```

To avoid having to do this again every time you log in on your Gemini account, you can edit the `.bash_profile` file in your home directory by using `nedit ~/.bash_profile` (or: `vi ~/.bash_profile`). This file is executed automatically when you log in on Gemini. If no such file exists, create it (the command `vi`[2] will automatically make an empty file if a file with the requested name does not yet exist). Once the file is opened/created, add the same two commands as above to the file and save it. Note that the changes in your bash profile will not become activated until you log in again, so either log out and in again or load the modules once more manually in the terminal.

[2] vi is a very robust editor because it runs directly in the terminal. You do need some basic commands to run it, see `https://www.cs.colostate.edu/helpdocs/vi.html`. i = insert text; esc = stop inserting; :q = quit file without saving, :q! = quite without saving even though changes were made; :x save and exit.

Now you have loaded the required software, it is time to download the PlaSim model to your home directory on Gemini. Download PlaSim by 'cloning' (copying) the PlaSim software from the Plasim repository on GitHub, an online platform for software development:

```
cd /nethome/<your-solisid>
git clone https://github.com/HartmutBorth/PLASIM.git
```

Go to the PLASIM directory that you just created and take a look at its contents using `ls`. Next, execute the configuration file of PlaSim to install the model and create the Model Starter (MoSt):

```
cd ~/PlASIM/
./configure.sh #command to execute the file configure.sh
```

This will go wrong if the mpi and sge modules are not loaded!
If you think that something went wrong and you want to re-install PlaSim, then first (!!) clean your PlaSim directory by moving into the PLASIM directory (`cd ~/PLASIM`) and then typing `./cleanplasim`. Then you can remove the entire PLASIM folder (see the commands in Table 1) and re-install.

2.2 Running your first simulation with the GUI

Now you are ready to do a first test run with the PlaSim model in the PLASIM directory within your home directory. To use the MOdel STarter MoSt to start a simulation, execute the most.x file that has just been created

```
./most.x
```

which launches the graphical user interface (GUI) (Figure 2). [3] [4]

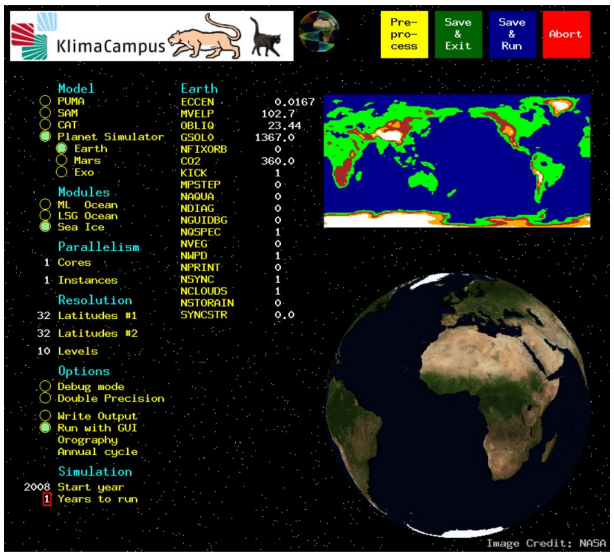


Figure 2: Screenshot of Model Starter (MoSt)

[3] If you get the following error on a Mac: "Cannot connect to X server", then it may be that you need to install Xquartz (`https://www.xquartz.org` on your computer). This is a programme that allows your terminal to open windows.

[4] If you get an error saying there is no rule to make target '.f90', needed for '.o', try excecuting ./configure.sh again and then restart the most.x executable. If this was not configured properly (e.g. after reloading or installing the model or a specific component), the GUI will not show any default number of cores/instances, telling you that this is most likely the issue.

In the interface, select **Planet Simulator** as the model to run , and select the option **Run with GUI**. In this window, you can also choose how many **Years to run**the model for and change some other settings. Choose 1 year of simulation time for this first test, and set the **Start year** to 2000. When you press **Save & Run** the model will be built with the selected settings and the the model will be launched. Another interface will appear in your window where you can see the output of the model changing over time. This gives you a first idea of what output you will have available to analyze. In that interface, you can also play around with parameters such as CO2 concentration and incoming solar radiation.

The GUI makes it easy to run the model for the first time, however using this interface you can only change a limited number of settings. Besides, you have a lot more freedom in analyzing your experiments when you visualize the model output data yourself. Therefore, from now on, we will no longer run the model directly from the graphical interface.

2.3 Building your simulation to run in parallel

For your project you will not run PlaSim with the GUI, but instead you will run it on the compute nodes of the Gemini server. By dividing the tasks of your simulation over multiple computer cores (= running your simulation in parallel), your simulations will be done much faster.

To prepare running PlaSim in parallel, execute the model starter again with `./most.x`. This time, in the GUI:

- Increase the number of cores that are used to run the model under **Parallelism**, for example to 4 or 8 cores (possible values: 1,2,4,8). Set Instances to 1.
- Adjust, if you like, the spatial resolution of the simulation, by adjusting the number of latitudes. Make sure that the number of latitudes is dividable by the number of cores, as the calculations over the latitudes will be split over different computer cores! You can choose 32, 48 or 64 latitudes.
- De-select **Run with GUI** and instead select **Write Output**
- Choose the number of years you would like to run the model for by changing **Years to run**
- Press **Save & Exit** (instead of **Save & Run**!) to build the model

Troubleshooting: If PlaSim throws a compilation error while building the model with several cores, the modules `sge/2011` and `mpi/openmpi-x86_64` may not have been loaded during configuration. Check with `module list` which modules are loaded and try reconfiguring or reinstalling PlaSim. If these moodules have been loaded, but the error nonetheless occurs, it can help to un-load the modules with `module purge` and loading the two modules again.

2.4 Setting up your simulations on the shared project space

As explained in Section 1.3, you will run and save the output of your simulations on the shared project space. Therefore, you now need to copy the settings of the simulation that you just configured in your home directory to your directory on the shared project space. To go to this directory, use:

```
cd /science/projects/ESM/users/<your solis-id>
```

NOTE: you must run simulations in your own directory on the shared project space because only there you have the right to write. Others can still read your data there, or copy it to their own directory.

Next, copy the files from the directory in which you built PlaSim to your personal directory on the shared project space. For an example, see Figure 3. First, create a folder with a logical name for this run, for example `testrun` or `defaultrun`. Organizing your different simulations well is crucial for when you analyze your different simulations later.

```
mkdir <name_of_directory>
cd <name_of_directory>
cp ~/PLASIM/plasim/run/* .  (note! the dot in the end means you will copy the
                                files to the current directory)
```

```
(esm) -bash-4.2$ cd /science/projects/ESM/users/herma040
(esm) -bash-4.2$ mkdir testrun
(esm) -bash-4.2$ cd testrun
(esm) -bash-4.2$ pwd
/science/projects/ESM/users/herma040/testrun
(esm) -bash-4.2$ cp ~/PLASIM/plasim/run/* .
(esm) -bash-4.2$ ls
GUI.cfg             N048_surf_0200.sra  N048_surf_1741.sra  kleiauf           most_plasim_run           salflu
N032_surf_0172.sra  N048_surf_0210.sra  beklas              kleiin1           most_plasim_t31_l10_p1.x  seaice
N048_surf_0129.sra  N048_surf_0212.sra  cpl_namelist        kleiswi           oceanmod_namelist         seamod_namelist
N048_surf_0169.sra  N048_surf_0229.sra  flukofile.txt       landmod_namelist  planet_namelist           surfmod_namelist
N048_surf_0172.sra  N048_surf_0232.sra  fluxmod_namelist    map.bmp           plasim_namelist           topogr
N048_surf_0173.sra  N048_surf_1730.sra  glacold             miscmod_namelist  radmod_namelist           vegmod_namelist
N048_surf_0174.sra  N048_surf_1731.sra  icemod_namelist     monsal            rainmod_namelist          wistrx
N048_surf_0199.sra  N048_surf_1740.sra  input               montem            runoffmap.txt             wistry
(esm) -bash-4.2$
```

Figure 3: Copying the run directory from your home directory to your personal directory in the shared project space.

Now that you copied the configuration of your simulation from the run directory in your home directory (`~/PLASIM/plasim/run/`) to your directory on the shared project space, it is useful to clean the run directory in your home directory before you build a new simulation. This avoids mixing up files of the different simulations. So, move to the PLASIM directory `cd ~/PLASIM` and execute the plasim cleaner `./cleanplasim`. This will empty the `~/PLASIM/plasim/run` directory. Once this is empty, you can set up a new simulation using the ModelStarter if you want to, by repeating the steps in Section 2.3.

Trouble shooting: If you get "permission denied" errors, this probably means that one of the course supervisors needs to add you to the project group!

2.5 Changing additional input parameters

While in the model starter some parameter settings can be changed (Section 2.3), PlaSim has many more model settings that can be changed by changing the 'namelist' files. For example, the file `plasim_namelist` (in the folder `/nethome/<your solis-id>/PLASIM/plasim/run`) contains all general simulations settings that you set in the model starter in Section 2.3. In other namelists you can control which submodules and parameter settings are used. For example, in the `seamod_namelist` you can switch on a sea-ice model and alter the sea ice albedo as a parameter. The PlaSim User Guide gives a full description of the submodules and settings that can be set in the namelist files. You can open and edit files (like the namelist) using

```
nedit <filename>
```

which will open a window wherein you can make and save your changes. If you want to change the namelists, you need to do so in your personal directory on the shared project space, since that is where you will run your simulations.

Caveat: When you run most.x, it might overwrite the namelist files in /PLASIM/plasim/run. Also, when you run ./cleanplasim, this will empty the folder /PLASIM/plasim/run. Therefore, it is good practice to always check your namelist files, especially changes you have made, after recompiling the model!

2.6 Submitting your simulation

After you have set all model settings, you are ready to submit the run to the Gemini compute nodes. For this we use the script `most_plasim_run`, in which you basically tell Gemini what to do. Copy the `most_plasim_run` file and edit it:

```
cp most_plasim_run my_most_plasim_run
nedit my_most_plasim_run
```

This script needs to be adjusted in order to run the model in parallel on multiple computer cores of Gemini. Typically, everything below the line `EXP=MOST # Name your experiment here` is already properly set up by MoSt, but the first few lines (above `EXP=MOST...`) need to be changed. In the end, the script should look roughly like the example below. A few attention points:

- You can set the name of the job (the task to do the simulation) that you will submit to Gemini with `#$ -N plasim`
- Set the number of cores you will use, e.g., 8, with `#$ -pe mpich 8`. This number needs to agree with the number of cores you chose in Section 2.3.
- Set the time you need for a job. `#$ -l h_rt=01:00:00`. With this setting the model will stop running after 1 hour, even if the simulation is not finished yet. Note to not use spaces before/after the '=', this might give errors.
- Adjust `EXP=MOST` if you want to use a different experiment name
- Change `YEARS=1` according to the number of years that you selected in Section 2.3.
- Adjust `mpiexec -np $NSLOTS most_plasim_t21_l10_p8.x` when you have changed the resolution of the model. For example if you set in ./most.x to run the model with the highest resolution (64 latitudes) on 4 cores: `mpirun -np $NSLOTS most_plasim_t42_l10_p4.x`. In this case `$NSLOTS` = 4, instead of 8 in the example above. Note that the name of your simulation file `most_plasim_t42_l10_p4.x` ends with p4, not p8!
- Hopefully, we will be given a special "queue" to submit simulations during the phase where we generate our output. If so, you need to add a special line in the first part (above the `EXP=MOST` statement), namely: `#$ -q soac.q` (where we assume that the queue name is soaq.q). You will be informed about the presence of a special queue.

```
#!/bin/bash
#$ -S /bin/bash      # name of the used shell
#$ -V                # copies your current (main node) settings to compute nodes
#$ -N plasim         # name of the job
#$ -pe mpich 8       # number of cores to run on (< 8 when mpich is used)
#$ -l h_rt=01:00:00  # reserved time on the nodes in hh:mm:ss
#$ -cwd              # execute job in current working directory


EXP=MOST    # Name your experiment here
[ $# == 1 ] && cd $1
```

```
rm -f plasim_restart # Comment out this line when restarting the model
rm -f Abort_Message
YEAR=0      # year number to start from
YEARS=1     # year number to end with
while [ $YEAR -lt $YEARS ]
do
   YEAR=`expr $YEAR + 1`
   DATANAME=`printf '%s.%03d' $EXP $YEAR`
   DIAGNAME=`printf '%s_DIAG.%03d' $EXP $YEAR`
   RESTNAME=`printf '%s_REST.%03d' $EXP $YEAR`
   mpirun -np $NSLOTS most_plasim_t21_l10_p8.x
   [ -e Abort_Message ] && exit 1
   [ -e plasim_output ] && mv plasim_output $DATANAME
   [ -e plasim_diag ] && mv plasim_diag $DIAGNAME
   [ -e plasim_status ] && cp plasim_status plasim_restart
   [ -e plasim_status ] && mv plasim_status $RESTNAME
done
```

Now you can submit the simulation. First check how busy the Gemini compute nodes are by typing the command `check_queue all`. This command will show how many cores are in use at the moment, for example that 12 out of 48 cores are used.

Trouble shooting: If you get the "argument missing" error, try adding a 0 (to see pending jobs) or a 1 (running jobs) behind the command `check_queue all`.

Then, submit the model with the command:

```
qsub my_most_plasim_run
```

and check the status of your simulation with:

```
qstat
```

The system will tell you the status of your simulation: qw = in the queue, waiting; r = running. If you get an empty response, your simulation is finished (either succesfully or crashed).
In case you want to kill the job, use the command `qdel`:

```
qdel <jobid>
```

The jobid can be retrieved with the command `qstat`. To check which dates have already been simulated by the model during the simulation, check the output in the diagnostic file `plasim_diag` with an editor, for example vi[5]:

```
vi plasim_diag
```

Scroll down to see, for example, the last day the model has computed. You can scroll up and down more quickly using `cntrl+b` and `cntrl+f`. The diagnostics file may be deleted once the simulation is finished.
In order to choose the right amount of cores, consider the occupation of the server by checking `check_queue all` and consider the resolution and amount of years you want to simulate. We give a table here with the approximate time that it takes to simulate 1 year, per resolution per amount of cores. This is to give you an indication and help you make choices on which resolution you need, for how long you need to run the model and how many cores and wall time you need to request. Exact duration can differ for different model settings.

[5] see footnote section 2.1

	1 core	2 cores	4 cores	8 cores
32 lat (T21)	500 sec	300 sec	190 sec	100 sec
48 lat (T31)	35 min	14 min	7.5 min	4 min
64 lat (T42)			18 min	10 min

2.7 Restarting the model

Sometimes your simulation might stop when you run out of the time that was requested. In order to not have to re-run all years, the model writes a restart file after every year. The restart will contain all information needed to start the simulation from where you left it, i.e., model settings and the current state of all relevant variables. Older restart files are removed, i.e., only the most recent year is kept. To continue the simulation, you can submit a new simulation that will initialize from the last restart file, `plasim_restart`.

In order to do such a restart simulation, comment out the line `rm -f plasim_restart` from the file `my_most_plasim_run` (use # before the line, e.g. `#rm -f plasim_restart`). When this line is turned into a comment, the model will look for `plasim_restart` and continue the simulations from this point. If you continue your run after 30 years of simulation, remember to fill in the end year as startyear now (change YEAR=0 to YEAR=30). This will only affect the names of the new output files. The actual time stamp in the data is determined by the latest restart file, or the `N_START_YEAR` parameter in `plasim_namelist` in case there is none. Keeping your filenames in ascending order ensures easy postprocessing, though.

Using restart-files as initial conditions is also a way to avoid computational time until the model reaches equilibrium after a spin-up period. For example, if you want to do simulations with the full ocean model that start from pre-industrial conditions, you can use the restart files in the `/science/projects/ESM/restart_files/` directory. Copy the restart file for the right resolution to the run directory in your directory on the shared project space, for example as follows:

```
cp /science/projects/ESM/restart_files/restart_LSG_T21/* .
```

Here, T21 stands for the model resolution, with alternatives: T31, T42 for higher resolution. Spin-ups are only available for the model setup with a full ocean (LSG), because these require the longest spinup. For the slab (i.e. mixed-layer, ML) ocean, the spinup time is at most 30 years.
As described in the README file in the restart folder, a few MOST-files (MOST.201 etc) are provided for each simulation as comparison material. These will be overwritten if you start a new experiment, unless you rename the files or move them to a different (sub)folder, or choose a different name for your new experiment. Also as described in the local README file, the file `plasim_restart_200` is the actual restart file after 200 years of spinup. To start from it, rename it into the standard name for restart files, i.e. `plasim_restart`.
When running from a restart, you generally have to make sure that the model setups are consistent, i.e. the same model components and parameters (except if making changes, e.g. in CO2 concentrations, is part of your experiment).
In the `/science/projects/ESM/restart_files` folder, the namelists and runscripts (`most_plasim_....`) are provided. This means you can restart by simply adjusting the `most_plasim_run`, e.g. adjusting start and end years, experiment name and runtime. You can therefore restart from the material in the folder, without having to run `./most.x` again. This will automatically ensure you use the same setup as the spinup. If desired, you can manually adjust (for example) CO2 concentrations in `radmod_namelist`.
If you want check settings used, consider the `MOST_DIAG.xxx` files provided in the restart folders. In particular, in our restarts `oceanmod_namelist` contains a parameter NOCEAN which must be set to 1 when running either the LSG or the ML model (as explained in the section on ocean models), and NLSG=1 in case you use the LSG model. Also, in `icemod_namelist`, NICE must be =1.

If for example you wish to run a mixed-layer (ML) simulation which is "the same as a particular LSG simulation except for the ocean model", you will need to set up this simulation through `most.x`, but you can check, and if needed adjust, namelist settings in the corresponding LSG run after running `most.x`.

2.8 Pumaburn post-processor

The raw output of the PlaSim model is written in so-called packed binary values, in files called MOST.001, MOST.002, et cetera. Luckily, PlaSim has a post-processor **Pumaburn** that helps you to select what output you want to use and to write out the raw model output in a convenient format to use for data visualisation, such as GRIB or NetCDF.

Return to your home directory and compile (= turn the code of the postprocessor into instructions that Gemini understands) the pumaburn post-processor by running the following commands:

```
cd ~/PLASIM/postprocessor/
c++ -O2 -o burn7.a burn7.cpp -I/opt/local/include -L/opt/local/lib -lm -lnetcdf_c++
```

Troubleshooting By adjusting the namelist `example.nl` you can select which simulated variables you want to write to an output file. This is done via the variable "code" (see Figure 5 for an example). Each variable has a code number and a short name (e.g., 130 and ta for the air temperature); code=ta and code=130 are equivalent. For now, write the following codes in `example.nl`: `code=ta,spd,ts,sic,rst,rlut,prc,prl,prsn`. You will need these variables to use the python script we provide for visualisation (see Section 3).

The command `./burn7.a -c` shows all variables that are possible to write to output (see also fig. 4. Choose which variables are relevant to analyze for your study and if you need daily output or monthly output, for which vertical levels, etc. See Figure 5 below for the options you can specify in the namelist. For now, use the options `multi=1,hpa=500,netcdf=1,htype=g,mean=0`. Then run the post-processor burn7 with this namelist and write the output to a netcdf file in the shared project directory:

```
./burn7.a /science/projects/ESM/users/<group>/<run>/MOST.001 /science/projects/
ESM/users/<group>/<run>/output.nc < example.nl
```

To postprocess multiple years (e.g., 3), make sure to set `multi=3` in the `example.nl` file. In our example, the post-processor will then start from the MOST.001 file and process this and the next two years (total = 3). To start with a later year (e.g. year 5), replace "MOST.001" by "MOST.005" in the command given above.

Additional options are explained in the PlaSim User Guide, chapter 5.

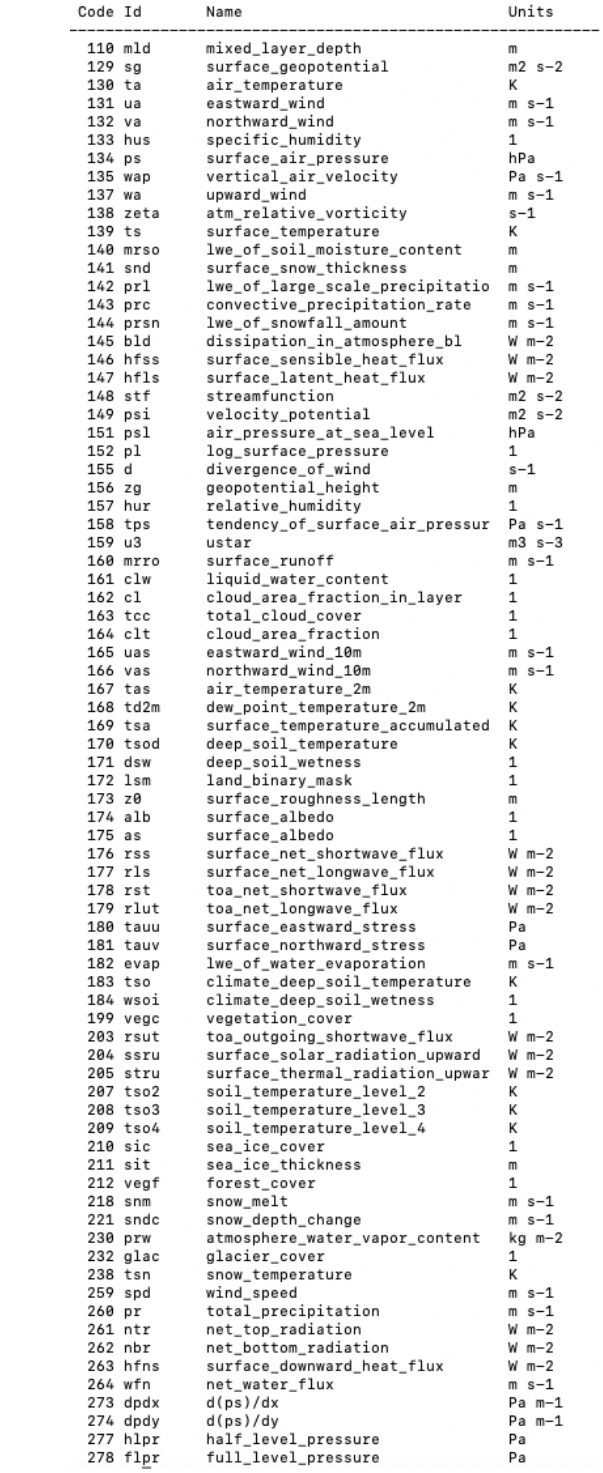
Trouble shooting: If `./burn7.a -c` doesn't work, open the file burn7.cpp and navigate to the passage around line 5500 for a list of variables with their codes. Alternatively, look at Appendix B of the PlaSim User Guide.

To check which data was indeed written to your just created netcdf file, use:

```
ncdump -h output.nc
```

to show the variables and their dimensions of the output file.

Trouble shooting: If you get the error "error while loading shared libraries: libmpi.so.12: cannot open shared object file: No such file or directory", try `module load mpi/openmpi-x86_64`



Code	Id	Name	Units
110	mld	mixed_layer_depth	m
129	sg	surface_geopotential	m2 s-2
130	ta	air_temperature	K
131	ua	eastward_wind	m s-1
132	va	northward_wind	m s-1
133	hus	specific_humidity	1
134	ps	surface_air_pressure	hPa
135	wap	vertical_air_velocity	Pa s-1
137	wa	upward_wind	m s-1
138	zeta	atm_relative_vorticity	s-1
139	ts	surface_temperature	K
140	mrso	lwe_of_soil_moisture_content	m
141	snd	surface_snow_thickness	m
142	prl	lwe_of_large_scale_precipitatio	m s-1
143	prc	convective_precipitation_rate	m s-1
144	prsn	lwe_of_snowfall_amount	m s-1
145	bld	dissipation_in_atmosphere_bl	W m-2
146	hfss	surface_sensible_heat_flux	W m-2
147	hfls	surface_latent_heat_flux	W m-2
148	stf	streamfunction	m2 s-2
149	psi	velocity_potential	m2 s-2
151	psl	air_pressure_at_sea_level	hPa
152	pl	log_surface_pressure	1
155	d	divergence_of_wind	s-1
156	zg	geopotential_height	m
157	hur	relative_humidity	1
158	tps	tendency_of_surface_air_pressur	Pa s-1
159	u3	ustar	m3 s-3
160	mrro	surface_runoff	m s-1
161	clw	liquid_water_content	1
162	cl	cloud_area_fraction_in_layer	1
163	tcc	total_cloud_cover	1
164	clt	cloud_area_fraction	1
165	uas	eastward_wind_10m	m s-1
166	vas	northward_wind_10m	m s-1
167	tas	air_temperature_2m	K
168	td2m	dew_point_temperature_2m	K
169	tsa	surface_temperature_accumulated	K
170	tsod	deep_soil_temperature	K
171	dsw	deep_soil_wetness	1
172	lsm	land_binary_mask	1
173	z0	surface_roughness_length	m
174	alb	surface_albedo	1
175	as	surface_albedo	1
176	rss	surface_net_shortwave_flux	W m-2
177	rls	surface_net_longwave_flux	W m-2
178	rst	toa_net_shortwave_flux	W m-2
179	rlut	toa_net_longwave_flux	W m-2
180	tauu	surface_eastward_stress	Pa
181	tauv	surface_northward_stress	Pa
182	evap	lwe_of_water_evaporation	m s-1
183	tso	climate_deep_soil_temperature	K
184	wsoi	climate_deep_soil_wetness	1
199	vegc	vegetation_cover	1
203	rsut	toa_outgoing_shortwave_flux	W m-2
204	ssru	surface_solar_radiation_upward	W m-2
205	stru	surface_thermal_radiation_upwar	W m-2
207	tso2	soil_temperature_level_2	K
208	tso3	soil_temperature_level_3	K
209	tso4	soil_temperature_level_4	K
210	sic	sea_ice_cover	1
211	sit	sea_ice_thickness	m
212	vegf	forest_cover	1
218	snm	snow_melt	m s-1
221	sndc	snow_depth_change	m s-1
230	prw	atmosphere_water_vapor_content	kg m-2
232	glac	glacier_cover	1
238	tsn	snow_temperature	K
259	spd	wind_speed	m s-1
260	pr	total_precipitation	m s-1
261	ntr	net_top_radiation	W m-2
262	nbr	net_bottom_radiation	W m-2
263	hfns	surface_downward_heat_flux	W m-2
264	wfn	net_water_flux	m s-1
273	dpdx	d(ps)/dx	Pa m-1
274	dpdy	d(ps)/dy	Pa m-1
277	hlpr	half_level_pressure	Pa
278	flpr	full_level_pressure	Pa

Figure 4: List with all possible output variables from the atmosphere

2.8.1 Running postprocessing in the queue

Postprocessing can take some time. You can run it in the queue with the script below. The script must be submitted within in the folder /PLASIM/postprocessor. You will need to adjust folder names and potentially the names of the MOST files and the output file.

```
#!/bin/bash
#$ -S /bin/bash # name of the used shell
#$ -V           # copies your current (main node) settings to compute nodes
#$ -N postproc
#$ -pe mpich 1
#$ -l h_rt=04:20:00  # reserved time on the nodes in hh:mm:ss
#$ -cwd              # execute job in current working directory
./burn7.a /science/projects/ESM/simulation_data/sim_XX/MOST.001
/science/projects/ESM/simulation_data/sim_XX/output.nc < example.nl
```

Tip 1: You can clone your postprocessing folder (e.g. `cp postprocessor postprocesor2`) and run one postprocessing activity in each folder. For example, for long simulations you can process the first and second half separately (do make sure the output files get different names, otherwise you overwrite your results!).
Tip 2: Postprocessing a T42 simulation takes 2-3 minutes per model year, handy to know for choosing your runtime.

Name	Def.	Type	Description	Example
HTYPE	S	char	Horizontal type	HTYPE=G
VTYPE	S	char	Vertical type	VTYPE=P
MODLEV	0	int	Model levels	MODLEV=2,3,4
hPa	0	real	Pressure levels	hPa=500,1000
CODE	0	int	ECMWF field code	CODE=130,152
GRIB	0	int	GRIB output selector	GRIB=1
NETCDF	0	int	NetCDF output selector	NETCDF=1
MEAN	1	int	Compute monthly means	MEAN=0
HHMM	1	int	Time format in Service format	HHMM=0
HEAD7	0	int	User parameter	HEAD7=0815
MARS	0	int	Use constants for planet Mars	MARS=1
MULTI	0	int	Process multiple input files	MULTI=12

Figure 5: Namelist settings to write PlaSim output using the post-processor pumaburn

3 Visualizing and analysing PlaSim output

3.1 For the first glance: ncdump and ncview

`ncview` is a software, installed on Gemini, which allows you to generate simple plots and movies of NetCDF (.nc) files. For example,

```
ncview output.nc
```

opens a window where you can click on the variable you want to inspect. By default, two-dimensional variable fiedswill be plotted as a colour map in the x-y-plane (i.e., horizontal direction), and you can choose the time and (if applicable) height level to show.

3.2 Visualizing/analysing with Python

To analyze and visualize the results for your project, using python gives more freedom in your analysis. You can download the output files on your local computer, or run python on Gemini to avoid issues with storage space.
A special conda environment is available that contains most important packages needed for analyzing the results. To activate it, enter:

```
source /import/gemini/usr/local/miniconda3/etc/profile.d/conda.sh
conda env create -f /science/projects/ESM/example_scripts/environment.yml
conda activate esm_course
```

Now you should see `(esm_course)` in front of your cursor, indicating this environment is active. If you wish to automatically activate this environment on every login, simply add the first and last of these commands to your `~/.bash_profile`. If you want to return to your default (base) environment, type `conda deactivate`.
Alternatively, you can create a new environment yourself:

```
module load miniconda/3
conda create -n esm_course python=3.6.8
```

When during the installation you are asked to proceed, answer 'y' to continue.
Activate the conda environment in the following way:

```
source /import/gemini/usr/local/miniconda3/etc/profile.d/conda.sh
conda activate esm_course
```

Then, through conda, you can install the python packages that are needed for your scripts, such as `cartopy` to plot the output netcdf data on a map. Install the packages in your conda environment as follows (make sure your conda environment activated!):

```
conda install -c conda-forge cartopy
conda install -c conda-forge scipy
conda install netcdf4
conda install xarray
conda install cftime
conda install dask
```

If the installation of a package does not succeed, you can try `conda install` instead of `conda install -c conda-forge` and vice versa. After installing the packages once, next times you only have to activate the conda environment to run python with these packages. You can also do this automatically every time you login to Gemini by adding the following lines to your `.bash_profile` file (as in Section 2.1):

```
source /import/gemini/usr/local/miniconda3/etc/profile.d/conda.sh
conda activate esm_course
```

An example output file and python script are given in `/science/projects/ESM/example_scripts/`. To use the example script for your own analysis you can copy it to your personal directory on the shared project space:

```
cd /science/projects/ESM/users/<your solis-id>
mkdir analyses
cd analyses
cp /science/projects/ESM/example_scripts/example_plotting.py .
cp example_plotting.py myplot.py
python myplot.py
analyses
```

Have a look at the figures that this script produces.
Troubleshooting: If your python script gives an error of the type "No module named 'xarray' ", try installing the missing package (in this example, xarray) using "conda install", as was explained above.

4 Running Plasim with an Ocean model

The standard version of Plasim has no ocean model, but instead prescribes the state of the ocean, in particular Sea Surface Temperature (SST). There is an influence of the ocean on the atmosphere, e.g., there can be heat fluxes and/or moisture fluxes between them. These depend on atmospheric condition (a 20 degree ocean surface will heat the overlying atmosphere more if the atmosphere is 0 degrees warm than when it is 18 degrees warm). However, these fluxes do not influence the ocean (in our example, the ocean does not cool even after releasing heat to the atmosphere).
You can, however, add an ocean model in two ways:

- A mixed-layer or slab ocean: The ocean is basically a layer of water that has a heat capacity, so it can absorb or release heat, depending on the atmospheric conditions, and thus provides s simple feedback. Ocean dynamic such as currents are not resolved. Their effect on the ocean's temperature (heat transport) very crudely mimicked by assuming heat diffusion.
  To switch on the mixed-layer ocean, you must select ML and deselect in the most.x, and set `nocean=1` and `nlsg=0` in the file `oceanmod_namelist` (in PLASIM/plasim/run). As explained below, you also need to fix certain parameters.

- A simple ocean model with some dynamics, the Large Scale Geostrophic (LSG) model.
  To switch on the LSG ocean, you must select LSG and deselect ML in the most.x, and set `nocean=1` and `nlsg=1` in the file `oceanmod_namelist` (in PLASIM/plasim/run)[6].

- When using an ocean model, you typically want a sea ice model too. To switch it on, you must select Sea ice in most.x and put NICE=1 in `icemod_namelist`(in PLASIM/plasim/run).

Running Plasim with an ocean model comes with some technical caveats outlined below.

4.1 Tuning the mixed-layer ocean model

In ./most.x you can select how you want to treat the ocean in your simulations. One option is to use prescribed fields of sea surface temperature, as we have done in Section 2. When you don't use any ocean model and only these prescribed fields, the parameter `nocean=0` in the `oceanmod_namelist` (which can be found in `~/PLASIM/plasim/run`, or in `/science/projects/ESM/users/<your solis-id>...` after you copied the settings). However, if you want to simulate the ocean in a simple way, you can use the mixed-layer ocean model (`nocean=1`), which is a simple slab ocean model.

If you do a simulation with a current CO2 level using this mixed-layer model, you will notice the world gets too cold. It gets too cold because there is no oceanic heat transport in this case. To get a more realistic climate, you need to add the oceanic heat transport. This can be done by using horizontal diffusion in the mixed layer. In `oceanmod_namelist`, in addition to `NOCEAN=1`, you need to set `NHDIFF=1` and `HDIFFK` (the diffusion coefficient) to some number. Note: you need to add additional lines to the namelist file to add these settings. For T21 resolution, HDIFFK=4.E4 gives a reasonable global mean surface T (about 15C). For the T42 resolution you have to slightly increase this parameter to tune the model, to around 4.5E4.[7]

[6] most.x sets nocean=0 when selecting LSG, and indeed LSG is run with that setting. It seems however that nocean=1 is needed ensure the atmosphere reats to the ocean model's SST rather than reading out the fixed SST.

[7] Even better results might be achieved with HDIFFK taking different values on each hemisphere (`https://gmd.copernicus.org/preprints/gmd-2020-245/` fig.1). However, for our purposes, this is not needed.

Figure 6: Simulated global surface temperature in a pre-industrial spin-up simulation using the T21, T31 and T42 resolutions with standard settings and tuned with ocean diffusivity

4.2 Running the LSG ocean model on T31 or T42 resolution

If you want to actively simulate ocean transport, you can run PlaSim coupled to the LSG ocean model. This can also be chosen in the `most.x`. Again, you need to manually change NOCEAN in the file `ocenamod_namelist` to 1 (as described above for the ML model). This simulation will give a more realistic climate since there is ocean transport between the different latitudes. You will get an error when you try to use the LSG ocean model with the higher resolutions for the atmosphere (T31, T42). To fix this problem, you will have to make little adjustment in the model code. Follow the following steps. First, we edit the fortran script `cpl.f90` that couples the ocean and atmosphere models:

```
cd ~/PLASIM
./cleanplasim           (clean directory to remove files of old model versions)
cd ~/PLASIM/plasim/src/     (location of the plasim source code)
nedit cpl.f90
```

When you open the file, you see as one of the first lines `parameter(nxa=64,nya=32)`. This line specifies the number of latitudes and longitudes in the atmospheric model, and will give a problem when you use the higher resolution settings. Comment this line out (place a `!` before the line, so you still know the original code) and add a new line: `parameter(nxa=96,nya=48)` for the T31 resolution or `parameter(nxa=128,nya=64)` for the T42 resolution.

Even after this step, when you will select the LSG module and T31 resolution in `most.x`, it will jump back to the T21 resolution and create a model executable file which is configured for the standard T21 resolution. To avoid this, you will have to edit the `most.c` file before building the simulation.

```
cd ~/PLASIM/
nedit most.c
```

and comment out the following lines from line number 2351 - 2355 (use CTRL+L to search for this line number), which automatically set the resolution to T21 when using the LSG ocean model. Since `most.c` is a script to create the model starter in C, use `//` at the start of each line to comment out the section:

```
// if (Lsg)
// {
//     Resolution = RES_T21;
//     Latitudes = ResLat[RES_T21];
// }
```

Now, configure the model starter to set-up the simulation with the changes you made in the most.c code.

```
./configure.sh
```

Next, run MoSt (`./most.x`) and set the resolution to T31 (or T42) and select the LSG model. Again, copy all files in the `~/PLASIM/plasim/run/` directory to your run directory in the ESM project directory. Check if `N_DAYS_PER_YEAR = 360` in the file `plasim_namelist`. Now, your simulation with the LSG ocean model is ready to go! As before, create a `my_most_plasim_run` script and submit the model to run on the Gemini compute nodes. Don't forget to undo the changes that you made if you wish to use the original model version.

4.3 LSG model output

The LSG model output is in SERVICE format. You can use the cdo[8] software to convert SERVICE data into NetCDF data, i.e., the same format also used for the atmospheric simulations.
In our specific case, you need the following command:

```
cdo -f nc -setgrid,LSG_grid <filenameIN> <filenameOUT.nc>
```

Here, `<filenameIN>` is the file wherein your ocean data is saved (probably called `lsg_output`), and `<filenameOUT.nc>` is the name you want to give the resulting NetCDF file. `LSG_grid` is a file that you need to copy to the directory in which you work with your LSG data. You can copy this file from `/science/projects/ESM/lsg_files/` to your current working directory by typing `cp /science/projects/ESM/lsg_files/LSG_grid ..`
Unfortunately, the resulting .nc file has unclear variable names such as "var18". Two (incomplete) lists of the names to which these codes correspond can be found on page 34 and 23 of the LSG manual[9].
With the above command, cdo will select all variables to move them into the .nc file, and data is written out every 24 days.

[8] Some cdo commands can be found here: `https://code.mpimet.mpg.de/projects/cdo/wiki/Tutorial`. However, you will probably not need much of this in the current project.

[9] `https://cera-www.dkrz.de/WDCC/ui/cerasearch/entry?acronym=DKRZ_Report_No02`

5 Running RCP scenarios

To simulate future climates with PlaSim you can force the simulations with CO2 concentrations that belong to different RCP scenarios (scenarios of how greenhouse gas concentrations will evolve in the future). CO2 is a fixed parameter in the `radmod_namelist`, however with a python script we can change this value every simulation year to simulate one of the RCP scenarios. Figure 7 shows the radiative forcing associated with the RCP8.5, RCP6, RCP4.5 and RCP3-PD scenarios (Meinshausen et al., 2011). The greenhouse-gas equivalent CO2 concentrations are stored in a .csv file on the shared project space for each of the four scenarios.

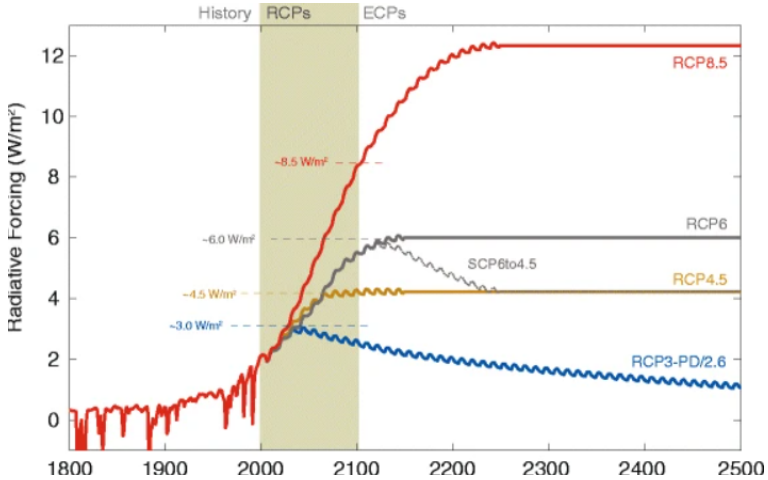


Figure 7: Radiative forcing of different RCP scenario's (Meinshausen et al., 2011).

Set up your simulations as follows to run a RCP scenario:

- `cd ~/PLASIM/`
- `./cleanplasim`
- `./most.x`
- Select the resolution and modules you wish. Copy the run files to your run directory on the shared project space as explained in Section 2.4.
- To initialize the model from pre-industrial conditions, copy the appropriate restart file to your run directory (see also the explanation in Section 2.7). For example, if you use the T21 resolution with mixed layer use the following restart file:

  `/science/projects/ESM/restart_files/preindustrial_T21_ML/plasim_restart`
- Copy `/science/projects/ESM/example_scripts/CO2_scenarios.csv` to your run directory
- Copy `/science/projects/ESM/example_scripts/CO2_scenario.py` to your run directory.
- Set in `CO2_scenario.py` the year you wish to start your scenario with, and which of the 4 scenarios to use.
- Copy `/science/projects/ESM/example_scripts/RCP_scenario_plasim_run` to your run directory. Note that you may need to copy or adjust some of the namelist files (e.g., radmod for CO_2) as well if you deviate from the original model setup.

- Change the `RCP_scenario_plasim_run` file according to your simulation as explained in Section 2.6. Set YEAR=0 and for YEARS the amount of years chosen in most.xto run.
- Submit `RCP_scenario_plasim_run` to run the model.

6 Adjusting surface fields

For the PlaSim simulations a set of surface data is provided as input for the three different resolutions. These files are located in the `~/PLASIM/plasim/dat/` directory. The Reference Manual gives a description of all the variables that are used as input fields.

For some model experiments you might want to adjust these input fields. We give an arbitrary example of how to do a simulation where the Sahara is fully covered with forest vegetation. Follow these steps to adjust the surface input files and submit a simulation with the adjusted surface forest cover as input:

- Set-up a simulation as usual with MoSt, selecting your preferred resolution and model settings, and build the model as explained in Section 2.3. This will copy the standard surface input fields for the chosen resolution to the run directory.
- Copy the run files to a new run directory in your personal directory on the shared project space, for example:
  `/science/projects/ESM/users/<your solis-id>/GreenSahara/` and set your current directory to this directory
- Copy the example python script `adjust_sra.py`, which you can use to adjust the surface fields, to this directory:
  `cp /science/projects/ESM/example_scripts/adjust_sra.py .`
- Adjust the python script as you like to select the region you want to adjust, which variable to adjust and to which new value. In this case we select the Sahara region and set the forest cover to 1.
- Run the python script with `python adjust_sra.py`. This will overwrite the old surface field with a new version with the adjustments.
- Submit the simulations as usual. This time the adjusted surface field will be used. Postprocess the data and specify as output the variable you adjusted in the post-processor namelist, this allows you to check if the changes you made were indeed incorporated in your simulations.

Note: in case you prefer to work with Fortran, an example script in Fortran made by a student is available here: `/science/projects/ESM/example_scripts/212_adjust.f90`

7 References

Meinshausen, M., S. J. Smith, K. V. Calvin, J. S. Daniel, M. L. T. Kainuma, J.-F. Lamarque, K. Matsumoto, S. A. Montzka, S. C. B. Raper, K. Riahi, A. M. Thomson, G. J. M. Velders and D. van Vuuren (2011). "The RCP Greenhouse Gas Concentrations and their Extension from 1765 to 2300." *Climatic Change (Special Issue)*, DOI: 10.1007/s10584-011-0156-z

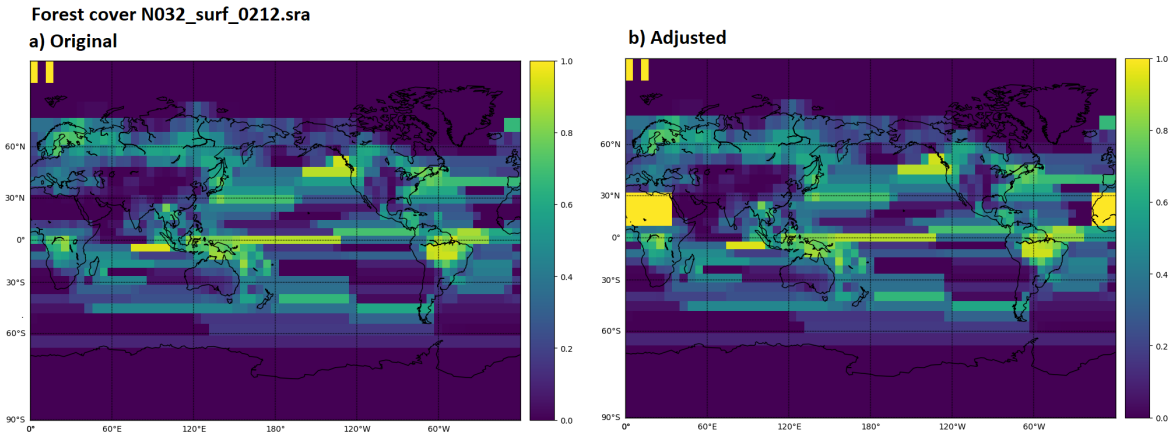


Figure 8: Plot of input forest cover, based on original file (a) and after adjustment with python script where Sahara region is vegetated