Hands-on Training with the NEMO Ocean Model

("*Introduction to Numerical Modelling in Oceanography*", Spring Semester - 1st Year)

Please feel free to contact John Karagiorgos for any question: jkaragiorgos@phys.uoa.gr

HELLENIC REPUBLIC
National and Kapodistrian
University of Athens
EST. 1837

Graduate Program in *"Oceanography and Marine Management"*
Ocean Physics & Modelling Group, Department of Physics

Part I: Beginner's Guide to Linux Command Line and Bash Scripting

Introduction:

This tutorial is designed to help beginners get started with Linux commands and bash scripting. Linux commands provide a powerful way to interact with the operating system, while bash scripting enables automation and execution of multiple commands. By the end of this guide, you will have a basic understanding of commonly used Linux commands and be able to create simple bash scripts with practical examples.

Linux Command Line Basics:

1. **Opening the Terminal:** To access the command line interface, open the terminal by pressing `Ctrl + Alt + T` or searching for "Terminal" in the applications menu.
2. **Basic Command Structure:** A command typically consists of the command name, options, and arguments. The general format is:

```
command [option(s)] [argument(s)]
```

3. **Navigating the File System:**

- `pwd` - Print the current working directory:

```
$ pwd
```

Example output: `/home/user/Documents`

- `ls` - List files and directories in the current directory:

```
$ ls            ## displays content of current directory
$ ls -al        ## list in alphabetical order
$ ls -lrt       ## list from oldest to most recent
$ ls -1         ## list containing file/directory names only
$ ls dir1       ## displays content of directory dir1
$ ls *.m        ## displays all files with a name ending as ".m"
$ ls toto*      ## displays all files with a name starting as "toto"
$ ls [a-f]*     ## displays all files with a name starting with a letter between a and f
```

- `cd` - Change directory:

```
$ cd /home/bob/dir1       ## go into directory /home/bob/dir1
$ cd ..                   ## go into parent directory
$ cd -                    ## go into previous directory
$ cd                      ## go into base directory (i.e. your home directory)
$ cd ~/dir1               ## go into dir1 that is located in your base directory
```

- `mkdir` - Create a new directory:

```
$ mkdir new_directory
```

- `rmdir` - Remove an empty dir:

```
$ rmdir new_directory
```

- `rm` - Remove files and directories:

```
$ rm file1.txt
```

- `cp` - Copy files and directories:

```
$ cp file1.txt file2.txt
```

- `mv` - Move or rename files and directories:

```
$ mv file1.txt new_directory/file.txt
```

4. **Working with Files:**
   - To edit an ASCII file (e.g. any fortran, python, matlab script) there are several options (see online documentations), e.g. :

```
$ nano file.txt
$ gedit file.txt &
$ vi file.txt
```

- To redirect the results of a command line into an ASCII file called toto.log:

```
$ ls -al > toto.log    ## toto.log is created (or overwritten if it existed)
                     ## and contains the result of the ls command.
$ ls -al >> toto.log   ## Same as above, except that the result is written
                     ## at the end of toto.log if it already exists.
```

- `cat` - Display the contents of a file:

```
$ cat file.txt
```

- `less` - View the contents of a file interactively:

```
$ less file.txt
```

- `head` - Display the first few lines of a file:

```
$ head file.txt
```

- `tail` - Display the last few lines of a file:

```
$ tail file.txt
```

- `grep` - Search for a pattern in a file:

```
$ grep "keyword" file.txt
```

- `wc` - Count lines, words, and characters in a file:

```
$ wc file.txt
```

- Modify read (r), write (w) and execute (x) properties of a file for you (u), the rest of the group (g), others (o) or all (a):

```
$ chmod a+r file    ## gives reading right to everybody
$ chmod go-w file   ## removes writing rights to users from the group and other
users
$ chmod +x file     ## gives executing rights to everybody
$ chmod u+x file    ## you obtain executing rights for this file
```

5. **System Information:**
   - `uname` - Print system information:

```
$ uname -a
```

- `whoami` - Print the current username:

```
$ whoami
```

- `date` - Display the current date and time:

```
$ date
```

- To check space usage and file sizes:

```
$ du -hs file1   ## displays size of file file1
```

```
$ du -hs dir1    ## displays total size of directory dir1 (including files
therein)
$ df .          ## displays usage and available space for current disk
$ df -h         ## displays usage and available space for every mounted disk
```

- `free` - Display memory usage:

```
$ free -h
```

- To obtain the history of previous command lines:

```
$ history            ## to get the full history
$ history | grep cp  ## to get the list of cp commands in the history
$ history | tail -5  ## to get the last 5 command lines
```

Introduction to Bash Scripting:

What is Bash? Bash (Bourne Again SHell) is a popular Unix shell and command language. It provides a scripting environment to automate tasks and execute sequences of commands.

1. **Creating and Running a Bash Script:**

- Create a new file with the `.sh` extension, e.g., `script.sh`.
- Open the file in a text editor (e.g. nano script.sh) and add the following lines:

```
#!/bin/bash
echo "Hello, World!"
```

- Save the file and exit the text editor.
- Set the script file as executable:

```
$ chmod +x script.sh
```

- Run the script:

```
$ ./script.sh
```

Output: `Hello, World!`

2. Variables and User Input:

- Assign a value to a variable. Type in terminal:

```
$ name="John"
```

- Access the value of a variable:

```
$ echo "My name is $name"
```

Output: `My name is John`

3. Prompt the user for input:

```
#!/bin/bash
echo "What is your age?"
read age
echo "Wow, you look younger than $age years old"
```

4. Control Structures:
- `if` statement:

```
#!/bin/bash
number=10
if [ $number -gt 5 ]; then
    echo "Number is greater than 5"
fi
```

- `for` loop:

```
#!/bin/bash
for i in 1 2 3; do
    echo "Number: $i"
done
```

- `while` loop:

```
#!/bin/bash
counter=1
while [ $counter -le 3 ]; do
    echo "Counter: $counter"
    counter=$((counter + 1))
done
```

5. Advanced Bash Scripting:
- Functions:

Define a function:

```
#!/bin/bash
greet() {
    echo "Hello, $1!"
}

# Call the function
greet "Alice"
```

Output: `Hello, Alice!`

- Read input from a file:

```
#!/bin/bash
while read line; do
    echo "Line: $line"
done < input.txt
```

- Command Line Arguments:

Access command line arguments:

```
$0  # Name of the script
$1, $2, ...  # First, second, etc., command line arguments
$@  # All command line arguments as an array
```

Simple Bash Scripts: Here are a few examples of simple Bash scripts to get you started:

- Script to calculate the sum of two numbers:

```
#!/bin/bash

echo "Enter the first number:"
read num1
echo "Enter the second number:"
read num2

sum=$((num1 + num2))
echo "The sum is: $sum"
```

- Check if a Number is Even or Odd

```
#!/bin/bash
```

```
read -p "Enter a number and I will check if its odd or even " mynumber
if [ $((mynumber%2)) -eq 0 ]
then
echo "Your number is even"
else
echo "Your number is odd."
fi
```

- Generate factorial of a number

The factorial of a number is the result of all positive descending integers. For example, the factorial of 5 would be 120: 5! = 5*4*3*2*1 = 120

```
#!/bin/bash
echo Enter the number you want to get factorial for
read mynumber
factorial=1
for ((i=1;i<=mynumber;i++))
do
factorial=$(($factorial*$i))
done
echo $factorial
```

4. Additional Resources:

- Bash scripting tutorials and examples (https://linuxconfig.org/bash-scripting-tutorial)
- https://linuxhint.com/30_bash_script_examples/

Part II: Compiling and Running the NEMO Ocean Model on Linux operating System

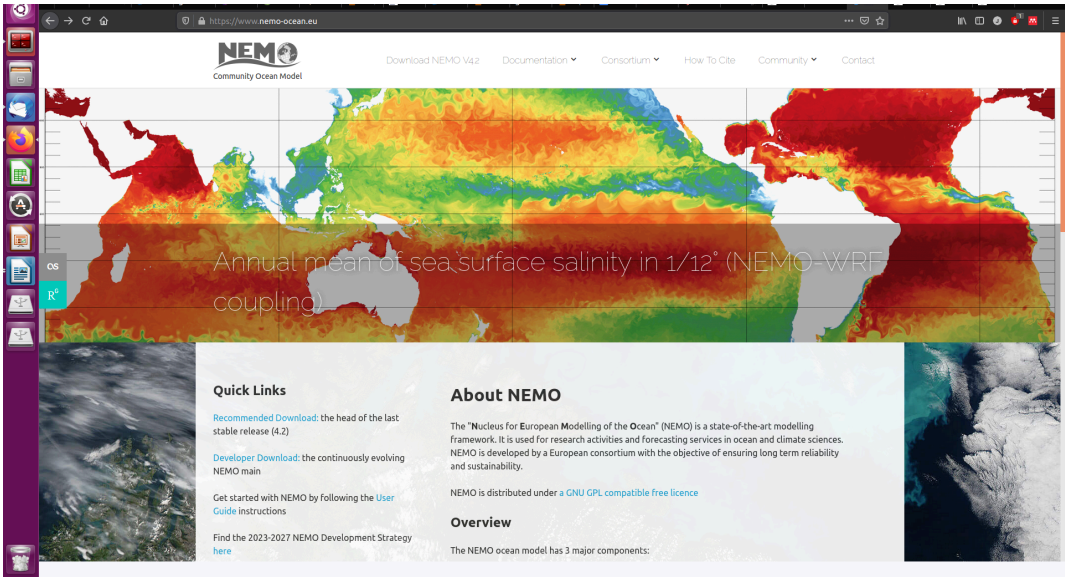
Introduction:

This tutorial aims to guide beginners through the process of compiling the NEMO (Nucleus for European Modelling of the Ocean) ocean model and launch simulations with the GYRE_PISCES benchmark configuration. By the end of this guide, you will have a basic understanding of the steps involved in compiling and running the NEMO ocean model.

About NEMO ocean model

The ocean circulation model NEMO (stands for *Nucleus for European Modelling of the Ocean)* is a state-of-the-art modelling framework for research activities and forecasting services in ocean and climate sciences. The NEMO platform is one of the leading ocean simulation frameworks and is used by a number of monitoring forecasting services worldwide. It solves the primitive equations in spherical coordinates, realistically representing the ocean dynamics and the interaction with the other components of the Earth system. The NEMO architecture includes 5 prognostic variables relative to the ocean circulation: the three-dimensional velocities, a non-linear sea surface height, the conservative temperature and the absolute salinity, projected onto an Arakawa C-type grid.

Further information on NEMO and its varied capabilities can be found at:
https://www.nemo-ocean.eu/



Operating procedures

The practical sessions will take place on a remote server. The access ID of each participant is:
User1 [**tp01**], User2 [**tp02**], User3 [**tp03**], User4 [**tp04**], User5 [**tp05**],

1. **Log-on onto the Linux machine**
   - Open a terminal
   - Connect to remote host

```
$ ssh -X -p 65000 tp[01-05]@88.197.81.176
```

Password for each login tp[01-05] : **####**

2. **Show your default home directory**

```
$ pwd
```

3. **Extract and install NEMOv4 source code**
   - Create a new directory in your $HOME

```
$ mkdir -p $HOME/Training2023/NEMO
$ cd $HOME/Training2023/NEMO
```

   - Get the NEMO v4 source code and navigate to the corresponding directory

```
$ cp /data/inputs/nemo/Training2023/Code/NEMOGCM.tgz .
$ tar zxvf NEMOGCM.tgz ; cd NEMOGCM
```

Description of 1st level tree structure

`arch`	Compilation settings
`cfgs`	Reference configurations (https://forge.ipsl.jussieu.fr/nemo/chrome/site/doc/NEMO/guide/html/cfgs.html)
`doc`	Documentation
`ext`	Dependencies of external packages included (`AGRIF`, `FCM` & `IOIPSL`)
`mk`	Compilation scripts
`src`	Modelling routines
`tests`	Test cases

`tools`	Utilities to {pre,post}process data (https://forge.ipsl.jussieu.fr/nemo/chrome/site/doc/NEMO/guide/html/tools.html)

4. Compile and create NEMO executable

The NEMO source code is written in ***Fortran 95*** and some of its prerequisite tools and libraries are already included in the download. Also, the necessary software are already installed on the system, including Fortran compilers with MPI implementation (e.g. Intel Fortran and C compilers, OpenMPI), NetCDF & HDF5 libraries and dependencies required by NEMO such as XIOS I/O library (https://forge.ipsl.jussieu.fr/ioserver/).

4.1 Setup your configuration architecture file

All compiler options in NEMO are controlled using files in ./arch/arch-'my_arch'.fcm where `my_arch` is the name of the computing architecture. In our case, the machine specific architecture file is under **"./arch/arch-ifort.fcm"**. Open the configuration file typing in terminal:

```
$ nano ./arch/arch-ifort.fcm
```

and explore the settings for the required environmental variables: `%NCDF_HOME`; `%HDF5_HOME` and `%XIOS_HOME` which refers to the installation directories.

4.2 Create and compile GYRE_PISCES configuration case.

`GYRE_PISCES` is an idealized configuration of NEMO representing a Northern hemisphere double gyres system, in the Beta-plane approximation with a regular 1° horizontal resolution and 31 vertical levels, with PISCES biogeochemical model. Analytical forcing for heat, freshwater and wind-stress fields are applied.

Run the 'makenemo' script to compile the code for the GYRE_PISCES configuration:

```
$ ./makenemo -m ifort -r GYRE_PISCES -n 'MY_GYRE_PISCES'
```

`makenemo` has several other options that can control which source files are selected and the operation of the build process itself. To output all the usage options type: `makenemo -h`

-m Computing architecture (./arch), FCM file describing the compilation settings

-r Reference configuration (./cfgs), proven with long-term support

-n Name for new configuration

Monitor the Compilation Process:

- The compilation process may take some time. Observe the output in the terminal for any errors or warnings.
- If errors occur, review the error messages and consult the NEMO documentation or community forums for troubleshooting assistance (https://nemo-ocean.discourse.group/).

After a successful configuration compilation, navigate to the appropriate directory ./cfgs/MY_GYRE_PISCES directory that will have the following structure:

Directory	Purpose
`BLD`	BuiLD folder: target executable, headers, libs, preprocessed routines, …
`EXP00`	Run folder: link to executable called `nemo`, namelists, `*.xml` and IOs
`EXPREF`	Files under version control only for official configurations
`MY_SRC`	New routines or modified copies of NEMO sources
`WORK`	Links to all raw routines from `./src` considered

5. **Run the model**

Once makenemo has run successfully, a symbolic link to the `nemo` executable is available in the run directory at ./cfgs/MY_GYRE_PISCES/EXP00. For the reference configurations, the EXP00 folder also contains the initial input files (namelists, `*.xml` files for the IOs, …). If the configuration needs other input files, they have to be placed here.

- **Understanding NEMO namelists**

  The namelist allows input variables (character, logical, real and integer) into the code without the need to recompile the model. For example you can define the model timestep and integration time, the boundary and initial condition, the physics schemes used for the simulation etc. There is one namelist file for each component of NEMO (dynamics, sea-ice, biogeochemistry...) containing all the FOTRAN namelists needed. The implementation in NEMO uses a two step process. For each FORTRAN namelist, two files are read:

  - A reference namelist (**namelist_ref**) is read first. This file contains all the namelist variables which are initialised to default values.
  - A configuration namelist (**namelist_cfg**) is read afterwards. This file contains only the namelist variables which are changed from default values, and overwrites those.

Explore the run directory and observe the GYRE_PISCES-specific configuration settings in the namelist_cfg file. The file (namelist_cfg) is in sections. E.g. the parameters of the run and the model timestep:

```
!-----------------------------------------------------------------------
&namrun         !   parameters of the run
!-----------------------------------------------------------------------
   cn_exp       =  "GYRE"   !  experience name
   nn_it000     =       1   !  first time step
   nn_itend     =       4320   !  last  time step
   nn_leapy     =       0   !  Leap year calendar (1) or not (0)
   nn_istate    =       0   !  output the initial state (1) or not (0)
/
!-----------------------------------------------------------------------
&namusr_def     !   GYRE user defined namelist
!-----------------------------------------------------------------------
   nn_GYRE      =       1       !  GYRE resolution [1/degrees]
   jpkglo       =       31      !  number of model levels
/
!-----------------------------------------------------------------------
&namdom         !   time and space domain
!-----------------------------------------------------------------------
   rn_rdt       = 7200. !  time step for the dynamics
/
```

In the default case, the simulation runs for 4320 timsteps * 7200 sec/timestep = 31104000 sec = 360 days.

Tracers

```
!!======================================================================
!!                     Tracer (T & S) namelists                 !!
!!                                                                    !!
!!   nameos     equation of state                                (default: NO selection)
!!   namtra_adv advection scheme                                 (default: NO selection)
!!   namtra_ldf lateral diffusion scheme                         (default: NO selection)
!!   namtra_mle mixed layer eddy param. (Fox-Kemper param.)      (default: OFF)
!!   namtra_eiv eddy induced velocity param.                     (default: OFF)
!!   namtra_dmp T & S newtonian damping                          (default: OFF)
!!======================================================================
!
!-----------------------------------------------------------------------
&nameos         !   ocean Equation Of Seawater                          (default: NO selection)
!-----------------------------------------------------------------------
   ln_eos80     = .true.        !  = Use EOS80 equation of state
/
!-----------------------------------------------------------------------
&namtra_adv     !   advection scheme for tracer                         (default: NO selection)
!-----------------------------------------------------------------------
   ln_traadv_fct =  .true.   !  FCT scheme
        nn_fct_h   =  2                 !  =2/4, horizontal 2nd / 4th order
        nn_fct_v   =  2                 !  =2/4, vertical   2nd / COMPACT 4th order
/
!-----------------------------------------------------------------------
&namtra_ldf     !   lateral diffusion scheme for tracers                (default: NO selection)
!-----------------------------------------------------------------------
```

```
   ln_traldf_lap   =  .true.    ! laplacian operator
   ln_traldf_iso   =  .true.    !  iso-neutral (standard operator)
   nn_aht_ijk_t  = 0     !  = 0   constant = 1/2  Ud*Ld   (lap case)
        rn_Ud           = 0.02          !  lateral diffusive velocity [m/s]
        rn_Ld           = 100.e+3       !  lateral diffusive length   [m]
/
```

Dynamics

```
!!======================================================================
!!                     ***  Dynamics namelists  ***                    !!
!!                                                                     !!
!!   nam_vvl    vertical coordinate options                    (default: z-star)
!!   namdyn_adv formulation of the momentum advection          (default: NO selection)
!!   namdyn_vor advection scheme                               (default: NO selection)
!!   namdyn_hpg hydrostatic pressure gradient                  (default: NO selection)
!!   namdyn_spg surface pressure gradient                      (default: NO selection)
!!   namdyn_ldf lateral diffusion scheme                       (default: NO selection)
!!   namdta_dyn offline TOP: dynamics read in files            (OFF_SRC only)
!!======================================================================
!
!-----------------------------------------------------------------------
&namdyn_adv     !   formulation of the momentum advection               (default: NO selection)
!-----------------------------------------------------------------------
   ln_dynadv_vec = .true.  !  vector form - 2nd centered scheme
        nn_dynkeg       = 0     ! grad(KE) scheme: =0   C2  ;  =1   Hollingsworth correction
/
!-----------------------------------------------------------------------
&namdyn_vor     !   Vorticity / Coriolis scheme                         (default: NO selection)
!-----------------------------------------------------------------------
   ln_dynvor_ene = .true.  !  energy conserving scheme
/
!-----------------------------------------------------------------------
&namdyn_hpg     !   Hydrostatic pressure gradient option                (default: NO selection)
!-----------------------------------------------------------------------
   ln_hpg_zco  = .true.  !  z-coordinate - full steps
/
!-----------------------------------------------------------------------
&namdyn_spg     !   surface pressure gradient                           (default: NO selection)
!-----------------------------------------------------------------------
   ln_dynspg_ts  = .true.  !  split-explicit free surface
/
!-----------------------------------------------------------------------
&namdyn_ldf     !   lateral diffusion on momentum                       (default: NO selection)
!-----------------------------------------------------------------------
   ln_dynldf_lap =  .true.       !       laplacian operator
   ln_dynldf_lev =  .true.       !  iso-level
   nn_ahm_ijk_t  = 0             !  = 0  constant = 1/2  Uv*Lv   (lap case)
        rn_Uv   = 2.0            !  lateral viscous velocity [m/s]
        rn_Lv   = 100.e+3        !  lateral viscous length   [m]
/
```

Vertical Physics schemes

```
!!======================================================================
!!                     vertical physics namelists                     !!
```

```
!!                                                                     !!
!!      namzdf          vertical physics manager                       (default: NO selection)
!!      namzdf_ric      richardson number vertical mixing              (ln_zdfric=T)
!!      namzdf_tke      TKE vertical mixing                            (ln_zdftke=T)
!!      namzdf_gls      GLS vertical mixing                            (ln_zdfgls=T)
!!      namzdf_osm      OSM vertical diffusion                         (ln_zdfosm=T)
!!      namzdf_iwm      tidal mixing parameterization                  (ln_zdfiwm=T)
!!======================================================================
!
!-----------------------------------------------------------------------
&namzdf         !   vertical physics                                   (default: NO selection)
!-----------------------------------------------------------------------
   ln_zdftke   = .true.          !  Turbulent Kinetic Energy closure     (T =>   fill namzdf_tke)
   ln_zdfevd   = .true.          !  enhanced vertical diffusion
        nn_evdm =       1        ! apply on tracer (=0) or on tracer and momentum (=1)
        rn_evd  =  100.          ! mixing coefficient [m2/s]
   !                    ! coefficients
   rn_avm0      =   1.2e-4       !  vertical eddy viscosity   [m2/s]  (background Kz if ln_zdfcst=F)
   rn_avt0      =   1.2e-5       !  vertical eddy diffusivity [m2/s]  (background Kz if ln_zdfcst=F)
   nn_avb       =       0        !  profile for background avt & avm (=1) or not (=0)
   nn_havtb     =       0        !  horizontal shape for avtb (=1) or not (=0)
/
!-----------------------------------------------------------------------
&namzdf_tke     !   turbulent eddy kinetic dependent vertical diffusion  (ln_zdftke =T)
!-----------------------------------------------------------------------
   nn_etau      =   0  !  penetration of tke below the mixed layer (ML) due to internal & intertial
waves
/
```

Run the model by typing:

```
$ cd cfgs/MY_GYRE_PISCES/EXP00
$ ./nemo &
```

Monitor and Analyze the Simulation:

Observe the simulation output and monitor its progress as it runs. The model will generate output files in NetCDF format (*.nc) containing various variables and diagnostic information.

```
# print model time-step
$ cat time.step

# print runtime messages warnings/errors etc
$ tail ocean.output

# check for running errors or abnormal behavior e.g. wrong model settings
$ grep "E R R O R" ocean.output

# list all model netcdf outputs
```

```
$ ls GYRE_5d*
```

6. **Quick and easy view of the netcdf model outputs:**

```
# visual browser of the netcdf files
$ ncview GYRE_5d*T*

# print in terminal the netcdf file information, e.g. variables,
coordinates, grid size etc
$ ncdump -h <filename>
```

7. **Plot the results with python script**

```
# Create directory to save your figures
$ mkdir plots
$ cd plots
# Create the python tplot script in your run directory
$ cp /data/inputs/nemo/Training2023/Plot/plot_GYRE.py .
# Generate plots
$ python plot_GYRE.py
# Merge all figures to one pdf file
$ convert $(ls -v *.png) NEMO_GYRE_plot_all.pdf
```

Output is NEMO_GYRE_plot_all.pdf, you can open the pdf file from the pdf viewer (evince NEMO_GYRE_plot_all.pdf) or download to your laptop.

8. **Additional Resources:**

- NEMO official website: https://www.nemo-ocean.eu
- NEMO documentation: https://zenodo.org/record/6334656
- NEMO discussion forum: https://nemo-ocean.discourse.group/

Part III: Configuration of Experiments in GYRE_PISCES

This GYRE configuration can be used as a benchmark since it is easy to increase resolution and obtain results without much computational cost. In the following exercises, various configuration possibilities in NEMO will be explored. For each exercise run the model and back up the outputs.

Exercise 1: Change the integration time of the simulation.

Launch a simulation for a period of 10 years (3600 days). **Hint:** modify the `nn_itend` value in the namelist_cfg. The run takes about 10 minutes.

Exercise 2: Start the simulation from a restart file and continue running for another 10 years.

In NEMO code is also possible to restart from a previous computation, by using a restart file. The restart files are created at the timestep frequency defined in the namelist at `&namrun` block (nn_stock), after a simulation run without errors.

In the `namelist_cfg` add in `&namrun` block:

```
ln_rstart   = .true.   !  start from rest (F) or from a restart file (T)
```

Then create a symlink of the restart file you want to continue from, e.g. GYRE_00043200_restart.nc by typing in the terminal:

```
$ ln -sf GYRE_00043200_restart.nc restart.nc
```

`GYRE_00043200_restart.nc` is the restart file created from exercise 1 at timestep 43200 (e.g. after 10 years of simulation).

Hint: remember to modify accordingly the `nn_it000` and `nn_itend` timestep in the namelist to continue the run from year 10 to year 20.

Exercise 3: Understand XIOS and create model outputs at different frequencies

The model's Input/Output management is performed by the XIOS external dedicated library (XML-I/O Server; http://forge.ipsl.jussieu.fr/ioserver) handling the output specifications e.g. output frequencies. XIOS allows for an efficient management of diagnostic outputs as the time averaging, file writing and even some simple arithmetic or regridding are carried out in parallel to the NEMO model run.

XIOS is controlled by means of XML input files (*.xml):

- `field_def_nemo-oce.xml` defines all the variables that NEMO can export.
- `file_def_nemo.xml` defines the variables we want to export to our simulation.

The user is allowed to modify only the `file_def_nemo.xml`. For example, the following block defines the output of the sea surface temperature with monthly time-averaging (**1mo**).

```
<file_group id="1m" output_freq="1mo"  output_level="10" enabled=".TRUE.">  <!-- 1mo files
-->
  <file id="file7" name_suffix="_grid_T" description="ocean T grid variables" >
     <field field_ref="sst"   name="tos" long_name="sea_surface_temperature" />
  </file>
</file_group>
```

Add the above block in line 96 of `file_def_nemo.xml` and run the model. What do you observe in the outputs?

Exercise 4: Change the model horizontal resolution.

GYRE grid resolution can be increased at runtime by setting a different value of nn_GYRE (integer multiplier scaling factor).

Try to increase the resolution to 1/2° horizontal resolution by setting in the namelist_cfg `nn_GYRE=2`.

```
!-----------------------------------------------------------------------
&namusr_def     !   GYRE user defined namelist
!-----------------------------------------------------------------------
   nn_GYRE      =       2       !  GYRE resolution [1/degrees]
/
```

Then, save your previous results in a folder:

```
$ mkdir outputs      # create a new folder
$ mv GYRE_* outputs  # move file to folder
```

Run again the model for 1 year (nn_it000=1, nn_itend=4320, ln_rstart=.false.). What do you observe? Is the model running successfully? **Hint:** Check `ocean.output` file for errors.

In case of errors, try to decrease the model time step by 50% and re-run the model.

```
!-----------------------------------------------------------------------
&namdom         !   time and space domain
!-----------------------------------------------------------------------
   rn_Dt        = 7200. !  time step for the dynamics
/
```

Hint: The model numerical stability depends on the Courant-Friedrichs-Lewy ratio, or simply CFL. The CFL condition is derived from the requirement that **the information should not propagate faster than the physical wave speed in the simulation**.

We define the Courant number (Cou) as "C δ t / δ x" where

- C is the wave speed,
- δ t is the time step,
- δ x is the spatial grid spacing

Thus, the physical meaning of the "C δ t" term is how far a fluid element can travel during a single time step δ t, relative to the grid spacing δ x. Mathematically, the CFL condition is expressed as:

Cou = C × δ t / δ x ≤ CFL_max, where CFL_max is the maximum allowable Courant number. The value of CFL_max depends on the numerical scheme used to calculate it in the simulation and the stability criterion of the PDE being solved.

Part IV: Visualization of the Results and Ocean Metrics

In this part, you will find the main steps to open, visualize and manipulate NEMO outputs in standard **NetCDF** file format, within **Python** and **Jupyter Notebook**.

1. **Remote visualization with Jupyter notebooks on remote server**

Open a new terminal and do the following setting your <PORT> to 88[01-04]. For example, if you are the user **tp01**, set <PORT> to **8801**:

```
$ nohup ssh -p 65000 -N -f -L localhost:<PORT>:localhost:<PORT>
tp[01-05]@88.197.81.176
```

Ensure you are asking to enter your password! Then login to the remote server:

```
$ ssh -p 65000 tp[01-05]@88.197.81.176
```

After the login, launch Jupyter Notebook from remote server, setting your <PORT>:

```
$ jupyter-notebook --no-browser --port=<PORT>
```

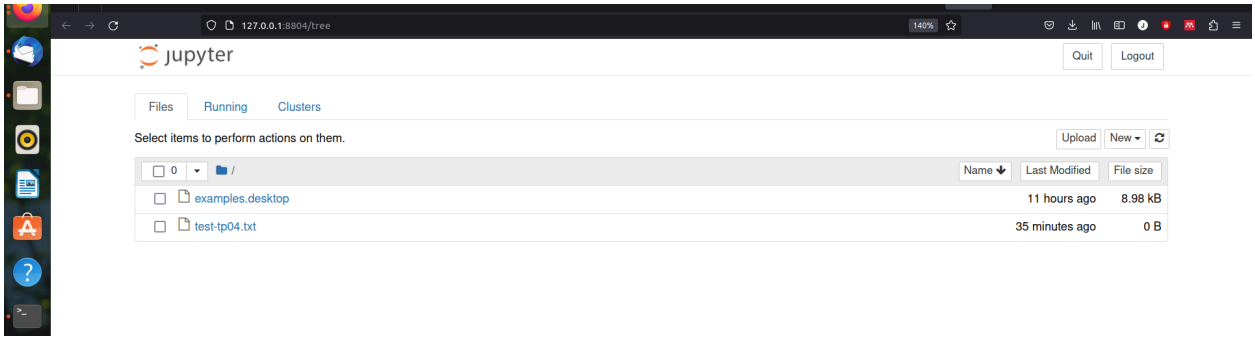
Example output:

```
tp03@ionio:~$ jupyter-notebook --no-browser --port=8892
[I 21:01:28.402 NotebookApp] Serving notebooks from local directory: /home/tp03
[I 21:01:28.402 NotebookApp] The Jupyter Notebook is running at:
[I 21:01:28.402 NotebookApp]
http://localhost:8892/?token=d03d07d09ca69ec0885ae035879a1408ad04240f5b897f38
[I 21:01:28.402 NotebookApp]  or
http://127.0.0.1:8892/?token=d03d07d09ca69ec0885ae035879a1408ad04240f5b897f38
[I 21:01:28.402 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip
confirmation).
[C 21:01:28.404 NotebookApp]

        To access the notebook, open this file in a browser:
        file:///home/tp03/.local/share/jupyter/runtime/nbserver-12659-open.html
        Or copy and paste one of these URLs:
        http://localhost:8892/?token=d03d07d09ca69ec0885ae035879a1408ad04240f5b897f38
        or http://127.0.0.1:8892/?token=d03d07d09ca69ec0885ae035879a1408ad04240f5b897f38
```

Open your local browser, and copy the corresponding link as in the highlighted text above.

The following page should appear:



2. **Open `GYRE_Visualization.ipynb` in your home directory, and follow the instructions.**