WW3 Tutorial: Run model with different input/output

Purpose

In this tutorial exercise we will go through the steps of running the model with different kinds of input, from the idealized forcing field to a realistic initial condition. Then we will go through the different outputs possible, from the gridded field to the spectral data at a point or along a track. All the input and output files will use the NetCDF format.

Input files

Copy the **TUTORIAL_INOUT** folder on your home directory:

```
open a new terminal on your local machine:
datarmor
cp -r  $TUTORIALS/TUTORIAL_INOUT  $HOME/
cp -r $DATA/INOUT/*     $HOME/TUTORIAL_INOUT/data/
or (if you are working remotely)
wget -mnH --cut-dirs=4 ftp://ftp.ifremer.fr/ifremer/ww3/TRAINING/TUTORIALS/TUTORIAL_INOUT/
cd TUTORIAL_INOUT/data
wget -mnH --cut-dirs=5 ftp://ftp.ifremer.fr/ifremer/ww3/TRAINING/DATA/INOUT/
```

Let's go now in ~/***TUTORIAL_INOUT***
At the start of this tutorial exercise, you will find there the following files :

in **data** folder :

iro1k.bot	**bathy file for Iroise sea 1km regular grid**
iro1k.mask	**mask file Iroise sea 1km regular grid**
ww3_*.inp	**input files corresponding to ww3 programs**
ww3_*.nml	**namelist files corresponding to ww3 programs**
job_shel.pbs	**script to run ww3_shel using mpirun**
SPECTRA_NC	**folder containing the spectral boundaries**
wind.nc	**wind input forcing field**

1. Preparing the run

Copy the executables already compiled on DATARMOR:

```
cd  $HOME/TUTORIAL_INOUT
mkdir  work && cd  work
cp $TRAINING/BIN/bin_Ifremer2_intel/ww3_*  .
```

If you are **remotely**, you need to recompile the code:

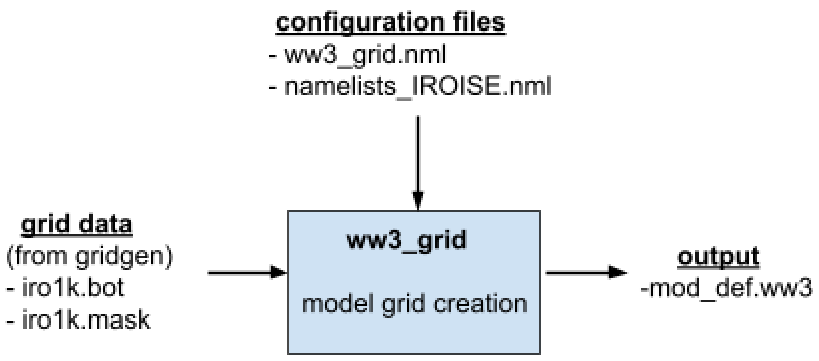
```
cd WW3
rm -rf build && mkdir build && cd build
cmake .. -DSWITCH=Ifremer2 -DCMAKE_INSTALL_PREFIX=install > build.out
make -j8 > make.out
make install > install.out
```

2. Defining the grid

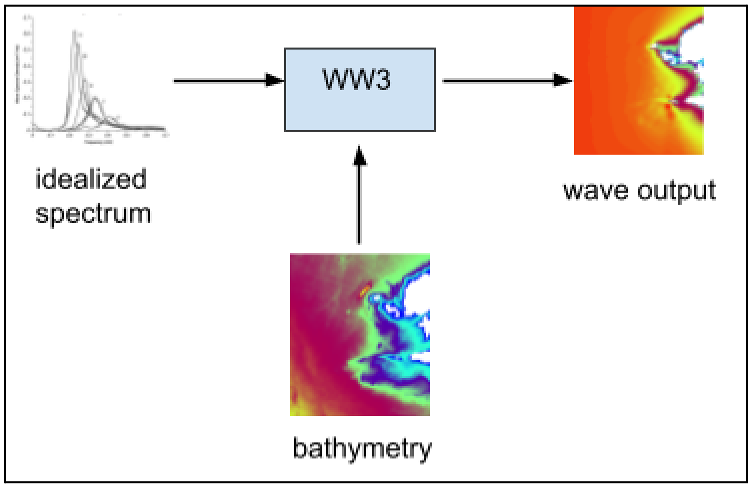
Using a bathy file **iro1k.bot** and a mask file **iro1k.mask** for the study area (see tutorial gridgen) and a namelist file **ww3_grid.nml** to define the grid settings and **namelists_IROISE.nml** to define the model parameterization (based on activated switches), you will create the model definition file **mod_def.ww3** readable by all the following ww3 programs. This is done by running the program **ww3_grid** :

```
cp ../data/iro1k.bot  .
cp ../data/iro1k.mask  .
cp ../data/ww3_grid.nml .
cp ../data/namelists_IROISE.nml .
./ww3_grid | tee ww3_grid.out
```

When it's done, list on your folder the last files created (**ls -lrt**), you should have a **mod_def.ww3**. This binary file contains the spectral and spatial definition of your model grid. It is needed by all the ww3 programs.



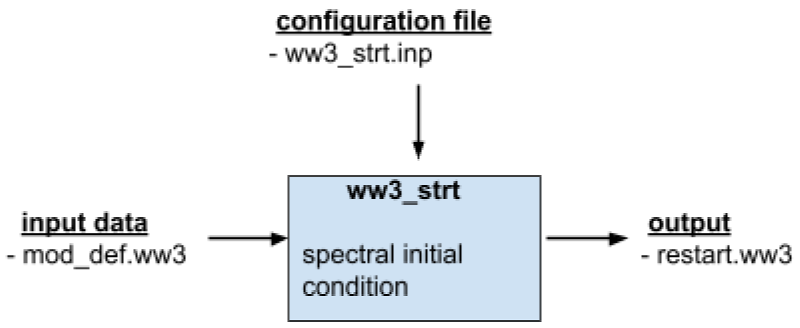
3. Running the model with an idealized spectral initial conditions



The easiest way to define an initial condition all over the grid is to define an idealized wave spectrum using **ww3_strt**. The program will only read a parameterization defined in the input file **ww3_strt.inp**, it can be a Gaussian in frequency and space or JONSWAP spectrum. For more details on the JONSWAP spectrum, see https://wikiwaves.org/Ocean-Wave_Spectra. It will create a file **restart.ww3** which contains the full spectrum information over the whole model grid.

```
cp ../data/ww3_strt.inp .
./ww3_strt | tee ww3_strt.out
```

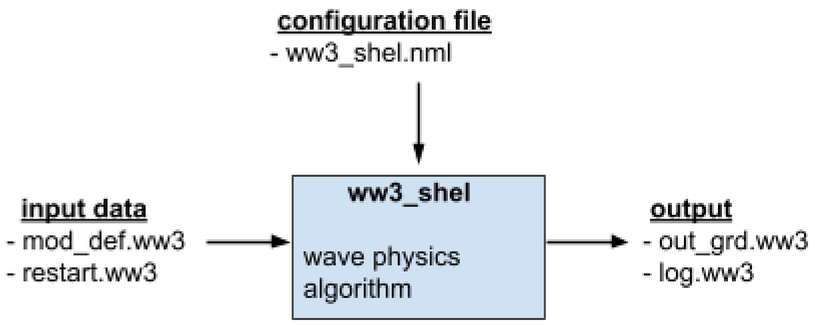
When it's done, list on your folder the last files created (**ls -lrt**), you should have a **restart.ww3**.



Now you can run the model, to do so, copy the namelist file **ww3_shel.nml** corresponding to **ww3_shel** and the MPI submission job **job_shel.pbs** :

```
cp ../data/job_shel.pbs  .
cp ../data/ww3_shel.nml .
qsub job_shel.pbs
qstat -u $USER   (to check if your job is still running :  job_shel.pbs $USER  00:03:08 R mpi_1)
```

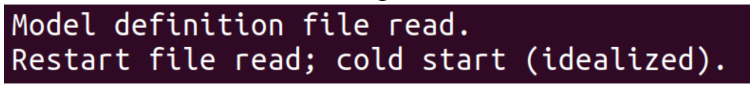
The result of this process is a binary file **out_grd.ww3** which contains all the mean wave parameters calculated on the grid for each output time step. In this case, no input wind is given in the model, it's only the initial idealized condition done by **ww3_strt**. When there is a **restart.ww3** in the current path, it will be automatically read by the model.



Always read the summary of ww3_shel program, which is **log.ww3** to know if everything went well before going further :

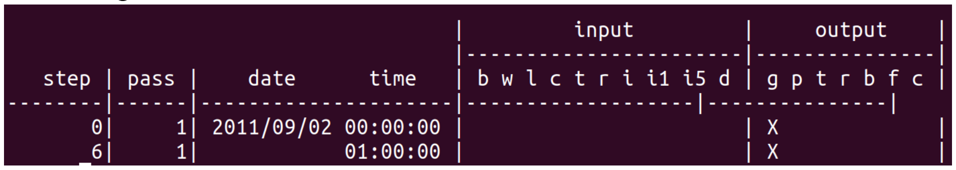
```
gedit log.ww3
```

At the top of the file, you'll find notifications to know if the mod_def.ww3 and restart.ww3 has been taken into account by the model with this kind message : .



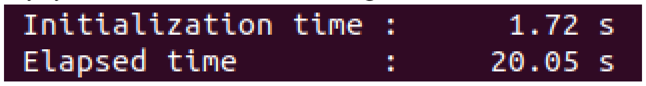
```
Model definition file read.
Restart file read; cold start (idealized).
```

In the output timeline, you can see the I/O updates shown by "X" symbols (or "L" for last update) in the column "g" which is "gridded field".



```
                                      |         input         |     output    |
                                      |-----------------------|---------------|
    step | pass |     date      time  | b w l c t r i i1 i5 d | g p t r b f c |
 --------|------|---------------------|-------------------|---------------|
        0|     1| 2011/09/02 00:00:00 |                       | X             |
        6|     1|            01:00:00 |                       | X             |
```

If it has finished successfully, you will have the following lines at the end of the log file :

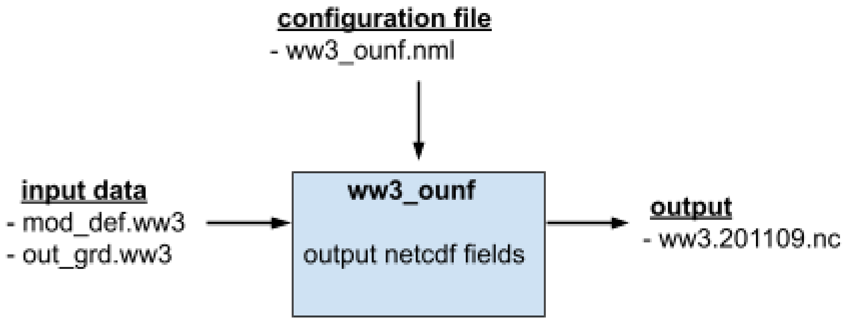


```
Initialization time :       1.72 s
Elapsed time        :      20.05 s
```

You can do the post-processing to extract wave parameters from the binary file out_grd.ww3. To do so, copy the namelist file **ww3_ounf.nml** corresponding to **ww3_ounf**, run it and visualize the results :

```
cp ../data/ww3_ounf.nml .
./ww3_ounf | tee ww3_ounf.out
ncview ww3.201109.nc
```

When it's done, list on your folder the last files created (**ls -lrt**), you should have a netCDF file **ww3.201109.nc** which contains all the requested wave fields.



You can quickly visualize the results with ncview :

```
ncview ww3.201109.nc
```

Looking at the MAPSTA variable, which is the map status of the grid, sea points are set to 1, land points are set to 0 and active boundaries points are set to 2. To know more about the map status, see Appendix in tutorial gridgen.

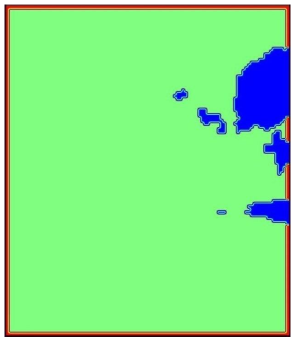


Figure 1 : MAPSTA variable

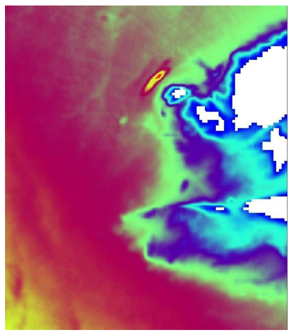


Figure 2 : depth variable

At the first time step, all the domain is initialized with a 2.5m swell from west, then on the next time steps it's only wave dissipation without any source terms.

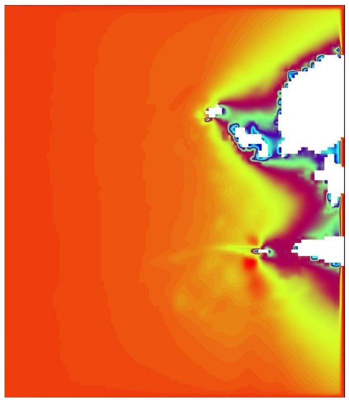
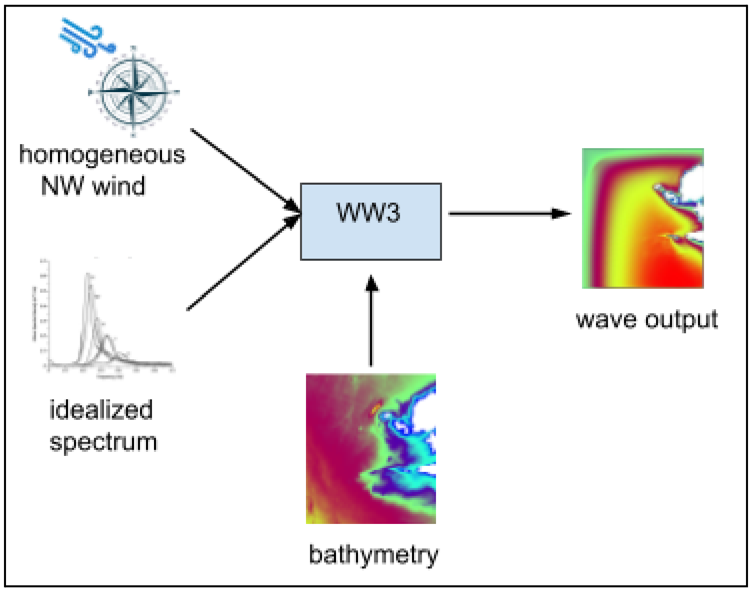


Figure 3 : Hs variable at last time step

Pay attention to the boundary points which are not updated in time. Since they are set to 2 in the map status, those points must be updated by a wave spectrum at every time step, here it’s only updated at the first time step by the restart.ww3 file.

4. Running the model with homogeneous input field



An homogeneous input field is a forcing field (wind, current, level, ice) that can be defined at a given value for a given date on the whole grid. Many homogeneous fields can be added to make it vary in time, otherwise it will keep its value for all the time steps. To define it, open the namelist file **ww3_shel.nml** to set up the input wind field. In the namelist INPUT_NML, set the forcing flag for the wind input to H for "homogeneous".

```
&INPUT_NML
  INPUT%FORCING%WINDS        = 'H'
/
```

To define the value of the homogeneous wind field, it will be set in the last section of the **ww3_shel.nml**. First, set the number of homogeneous wind field to use in the namelist HOMOG_COUNT_NML

```
&HOMOG_COUNT_NML
 HOMOG_COUNT%N_WND   =   1
/
```

Then define the date, wind speed, direction and air-sea difference in HOMOG_INPUT_NML

```
&HOMOG_INPUT_NML
 HOMOG_INPUT(1)%NAME       = 'WND'
 HOMOG_INPUT(1)%DATE       = '20110902 000000'
 HOMOG_INPUT(1)%VALUE1     = 20.
 HOMOG_INPUT(1)%VALUE2     = 315.
 HOMOG_INPUT(1)%VALUE3     = 2.0
/
```

or in one line :

```
&HOMOG_INPUT_NML
 HOMOG_INPUT(1)  = 'WND'  '20110902 000000'   20.   315.   2.0
/
```

Now you can rerun the model :

```
qsub job_shel.pbs
qstat -u $USER   (to check if your job is still running)
gedit log.ww3
```

In this case, the wind has only been updated at a given time, then it keeps constant. You can read the file **log.ww3** to verify if the run has terminated successfully. In the input timeline, you can see the first I/O update shown by the "F" symbol in the column "w" for "wind".

You can post-process and visualize the results :

```
./ww3_ounf | tee ww3_ounf.out
ncview ww3.201109.nc
```

At the first time step, all the domain is initialized with a 2.5m swell from west, then a constant 20 m/s wind from North-West blows on all the grid at every time step which produces a wave growth coming from wind direction.

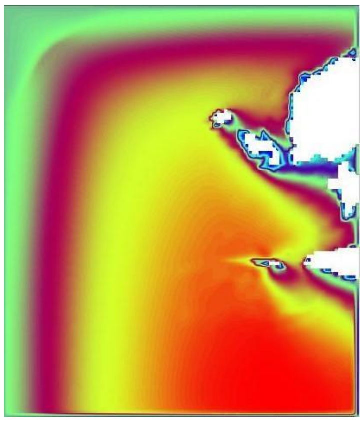
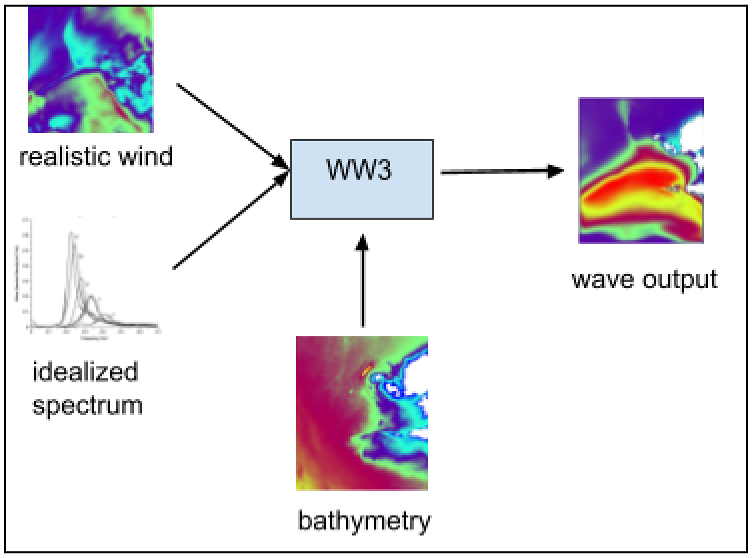


Figure 4 : Hs variable at last time step

5. Running the model with real input forcing field

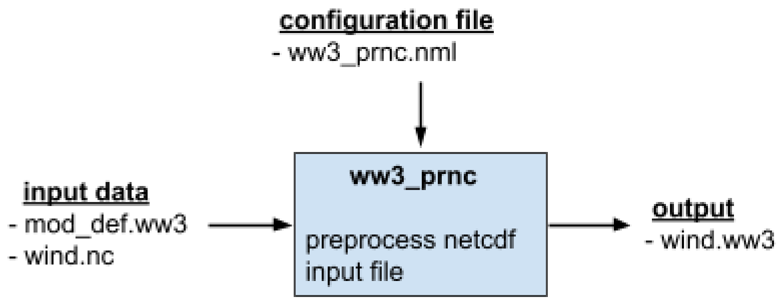


The main global wind reanalysis has been done by NCEP and ECMWF institutes.
-From the NCEP, the historical one is CFSR from 1979 to 2010 : http://rda.ucar.edu/datasets/ds093.1/
and the latest one is CFSRR from 2011 to now : http://rda.ucar.edu/datasets/ds094.1/
-From ECMWF, the latest one is ERA-5 : https://climate.copernicus.eu/climate-reanalysis

For the tutorial, you just have to copy the wind file from the data folder. Since it's a netcdf file, it must be converted into a binary format readable by WW3, this is done with the preprocessing program **ww3_prnc**. Another important point is that the grid used for the wind has not the same size and resolution as the one of the model, so it must be interpolated over the model grid, this is done by default by ww3_prnc. The corresponding namelist file **ww3_prnc.nml** contains all the settings needed to convert it :

```
cp ../data/wind.nc  .
cp ../data/ww3_prnc.nml .
./ww3_prnc | tee ww3_prnc.out
```

When it's done, list on your folder the last files created (**ls -lrt**), you should have a **wind.ww3**.



To define the external input field, you have to edit in **ww3_shel.nml**. In the namelist INPUT_NML, set the forcing flag for the wind input to T for "external forcing file".

```
&INPUT_NML
  INPUT%FORCING%WINDS       = T
/
```

Remove or comment the content of the homogeneous namelists. You should have something like :

```
&HOMOG_COUNT_NML
/

&HOMOG_INPUT_NML
/
```

Now you can run the model :

```
qsub job_shel.pbs
qstat -u $USER   (to check if your job is still running)
gedit log.ww3
```

In this case, the input wind is updated every hour in the model using the wind.ww3. You can read the **log.ww3** to know at which frequency the forcing fields are updated. In the input timeline, you can see the I/O updates shown by "F" then "X" symbol in the column "w" for "wind".

You can post-process and visualize the results :

```
./ww3_ounf | tee ww3_ounf.out
ncview ww3.201109.nc
```

At the first time step, all the grid is initialized with a 2.5m swell from the west, then a realistic wind blows on all the grid and is updated every hour.

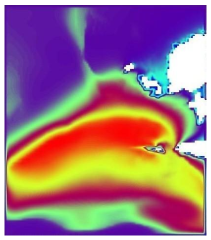
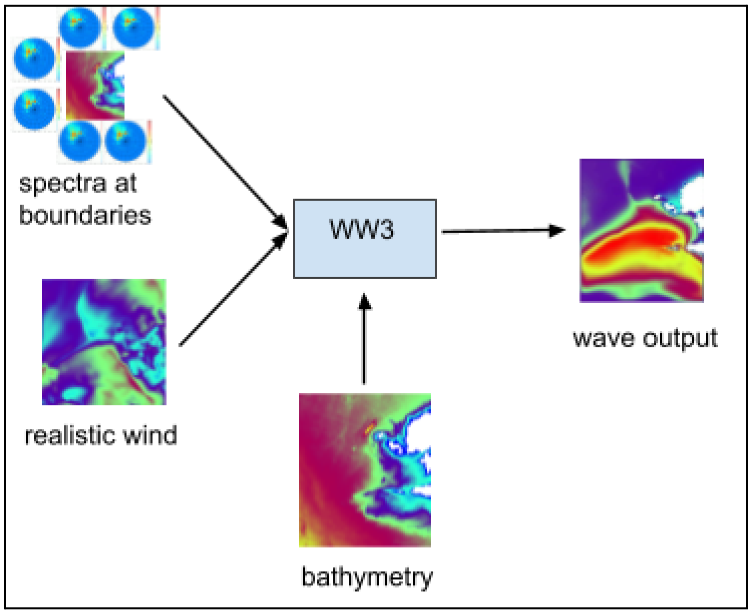


Figure 5 : Hs variable at last time step

6. Running the model with real spectral boundary conditions



Because most of your grid configurations are not covering the globe or a closed sea, you have to take into account sea states conditions at the surroundings of your grid to propagate them into your domain. To achieve this, you must add realistic wave spectra at the open boundaries of the grid.

One way to do it is to run a multi-grid model to communicate spectral boundaries between the global and the regional grid (see TUTORIAL_MULTI). Another way to do it is to first run a global grid model and output wave spectra at the boundary points needed for your regional grid. For that purpose we did a global hindcast over almost 30 years and we provided 2D spectral points every 0.5 degree all along the global coastline.

The location of 2D spectral output points for our hindcasts can be seen with google earth :

```
cp -r $DATA/KMZ  $HOME/
chromium-browser https://earth.google.com/web/
```

In a web browser, go to the section “Project”, then “Open” and browser in the folder “KMZ” to open the file **IOWAGA_WWATCH_MULTI_output_points.kmz**

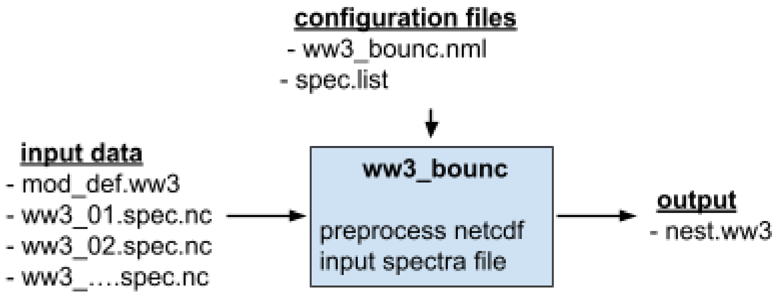
To automatically find the list of spectra needed for your grid, see Appendix F in the tutorial GRIDGEN.

For example, 2020 for North Hemisphere between 19W and 19E :
ftp://ftp.ifremer.fr/ifremer/dataref/ww3/GLOBMULTI_ERA5_GLOBCUR_01/POINTS/2020/SPEC_NW19toE19/

For the tutorial, you just have to copy the spectra files from the data folder. The active boundaries of your grid are defined in the mask file by values at 2 as shown on *Figure 1*. You don’t need to provide a spectral point at each active boundary point. The preprocessing program **ww3_bounc** will read all the provided netcdf files and interpolate them to all the active boundaries and create a unique binary file **nest.ww3**. All the settings are defined in **ww3_bounc.nml** and the netcdf files to read are listed in **spec.list** :

```
cp -r ../data/SPECTRA_NC .
cp ../data/spec.list .
cp ../data/ww3_bounc.nml .
./ww3_bounc | tee ww3_bounc.out
```

When it's done, list on your folder the last files created (**ls -lrt**), you should have a **nest.ww3**.



Now you can run the model :

```
rm -f restart.ww3    (remove restart created from an idealized spectrum)
qsub job_shel.pbs
qstat -u $USER       (to check if your job is still running)
gedit log.ww3
```

In this case, the boundary conditions from a global domain are updated 3-hourly (at the next global time step). When there is a **nest.ww3** in the current path, it will be automatically read by the model. You can read the **log.ww3** to see that the input boundary has been updated at the first time of the run. In the input timeline, you can see the I/O updates shown by "F" for the restart.ww3 then "X" symbol in the column "b" for "boundary".

You can post-process and visualize the results :

```
./ww3_ounf | tee ww3_ounf.out
ncview ww3.201109.nc
```

Similar results as before but now with calm sea conditions at the initialization, then wave energy propagating a 1m swell from the active boundaries into the domain.

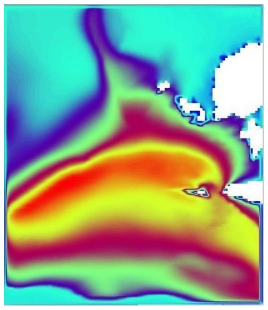


Figure 6 : Hs variable at last time step

Depending on your domain size, a spin up time is needed to consider the sea states conditions realistics on the whole domain. For a global grid, 15 days are required but almost a few days for a regional grid.

If you want to avoid running the model a few days before your required start date, you have to save the wave spectra at each point at the last time step of a previous run a **restart.ww3** file. How to output it is explained in the next chapter of this tutorial.

7. Running the model with full output options

There are also different kinds of output data that ww3 can handle. It will be defined in **ww3_shel.nml** in the namelist OUTPUT_TYPE_NML and OUTPUT_DATE_NML with 7 types of outputs defined as fields of mean wave parameters, spectral point output, spectral point output along a track, restart file, boundary data, separated wave fields data and the coupling fields.

For each output data, you have to set the start and stop data which can be different to the start and stop dates of the run itself. The time stride of each output data must be greater or equal to the global time step, ideally it should be a multiple of it. To disable an output type, you have to set its time stride to 0, which is the default value.

The first output data is the "**fields of mean wave parameters**". You must specify in OUTPUT_TYPE_NML the output fields you want to request : (it should already be done)

```
&OUTPUT_TYPE_NML
 TYPE%FIELD%LIST = 'HS FP DIR DP CHA UST DPT CUR WND'
/
```

Then define the start date, the time stride and the stop date in OUTPUT_DATE_NML :

```
&OUTPUT_DATE_NML
  DATE%FIELD%START    = '20110902 000000'
  DATE%FIELD%STRIDE   = '3600'
  DATE%FIELD%STOP     = '20110902 060000'
/
```

or in one line :

```
&OUTPUT_DATE_NML
  DATE%FIELD  = '20110902 000000'  '3600'  '20110902 060000'
/
```

The second output data is the "**point output**". The positions of spectral points to output are listed in a text file **points.list**. This file contains the list of points defined by longitude, latitude and name.

Copy the spectral output file :

```
cp ../data/points.list .
```

The content of points.list should be :

```
-4.97 48.29  '62069'
-6.00 48.29  'Boundary'
```

Then define the start date, the time stride and the stop date in OUTPUT_DATE_NML :

```
&OUTPUT_DATE_NML
  DATE%FIELD  = '20110902 000000'  '3600'  '20110902 060000'
  DATE%POINT  = '20110902 000000'  '3600'  '20110902 060000'
/
```

The third output data is the "**output along track**". The positions of spectral points to output are listed in a text file **track_i.ww3** (after the header line). This file contains the list of points defined by time, longitude, latitude and name. The model will output 2D spectra at the closest grid points for the closest output dates.

Copy the track input file :

```
cp ../data/track_i.ww3 .
```

The content of track_i.ww3 should be :

```
WAVEWATCH III TRACK LOCATIONS DATA
20110902 020000     -5.0    48.0   S1B
20110902 040000     -4.2    48.5   S1C
```

Then define the start date, the time stride and the stop date in OUTPUT_DATE_NML :

```
&OUTPUT_DATE_NML
  DATE%FIELD    =  '20110902 000000'  '3600'  '20110902 060000'
  DATE%POINT    =  '20110902 000000'  '3600'  '20110902 060000'
  DATE%TRACK    =  '20110902 000000'  '3600'  '20110902 060000'
/
```

The fourth output data is the "**restart file**". To generate only one restart file for example at the end of the run, you have to set the same dates for the start and stop date and set a stride value greater than 0. Then start date, the time stride and the stop date are defined in OUTPUT_DATE_NML. In the tutorial, you will output one restart file at the end of the run :

```
&OUTPUT_DATE_NML
  DATE%FIELD    =  '20110902 000000'  '3600'  '20110902 060000'
  DATE%POINT    =  '20110902 000000'  '3600'  '20110902 060000'
  DATE%TRACK    =  '20110902 000000'  '3600'  '20110902 060000'
  DATE%RESTART    =  '20110902 060000'  '1'  '20110902 060000'
/
```

The fifth output data is the "**boundary data**". The output boundary points must be previously defined in the **ww3_grid.nml** to output spectral conditions at the boundaries of an inner grid. It is not used often, the preferred way to do it is to output spectral points that can be used as spectral boundaries for an inner by running ww3_bounc as you did for this run in the previous exercise. It will not be used in this tutorial, by default the time stride is set to 0 so it is disabled.

The sixth output data is the "**separated wave field data**". This output is to obtain spatially and temporally consistent fields and time series of all the component wave systems in the computational domain; there is a dedicated tutorial for this (see TUTORIAL_WAVETRACK). It will not be used in this tutorial, by default the time stride is set to 0 so it is disabled.

The seventh output data is the "**coupling**". This output is to exchange fields between the wave model and other coupled models, there is also a dedicated tutorial for this (see TUTORIAL_COUPLING). It will not be used in this tutorial, by default the time stride is set to 0 so it is disabled.

Now you can run the model again to output all the model results.

```
qsub job_shel.pbs
qstat -u $USER   (to check if your job is still running)
gedit log.ww3
```

In the output timeline, you can see the I/O updates shown by "X" symbol in the column "g" for "gridded field" and also in the column "p" for "output points". A "L" symbol appears at the last time step in the column "r" for "restart file".

When it's done, list on your folder the last files created (**ls -lrt**), you should have this :

log.ww3	log file of the input/output
output.mpi	log file of the time steps
out_grd.ww3	binary output file of the gridded data
out_pnt.ww3	binary output file of the spectral data
track_o.ww3	binary output file of the track data
restart001.ww3	binary output file of the restart data

The **out_grd.ww3** file will be used by **ww3_ounf** to generate the gridded fields data.
The **out_pnt.ww3** file will be used by **ww3_ounp** to create the output spectra point.
The **track_o.ww3** file will be used by **ww3_trnc** to create the output track point.
The **restart001.ww3** file must be renamed as **restart.ww3** to be used as initial condition for the next run.

Now you can post-process the binary results to …

… netcdf gridded fields :

```
cp ../data/ww3_ounf.nml .
./ww3_ounf | tee ww3_ounf.out
ncview ww3.201109.nc
```

… netcdf spectra point :

```
cp ../data/ww3_ounp.nml .
./ww3_ounp | tee ww3_ounp.out
idlww3
open ww3.201109_spec.nc  as done in TUTORIAL_VISU_SPEC
 ( modelled time series → modelled → f-theta spec → POLAR_POLY → scroll the time steps  )
```

… netcdf spectra points along track :

```
cp ../data/ww3_trnc.nml .
./ww3_trnc | tee ww3_trnc.out
```

note : to visualize the results, you must extract the data from a given location and time, then use idl

Conclusion

In this tutorial, we have studied how to run the model from theoretical to realistic forcing fields, using netcdf files in input and output with the different ww3 programs to pre-process and post-process the data. Now you should be able to redo those steps on your own model grid with the forcing fields of your choice. Even if it is tempting to directly make a fully realistic run over a newly designed grid, it is important to do it step by step like in this tutorial to be sure your model is behaving correctly.

More information:
Mickaël Accensi (Mickael.Accensi-at-ifremer.fr)