Bathymaker Formation 2026

1-Small description of the program

Here we are describing the method to create a grid by using a small program which has been developed using routines existing inside SYMPHONIE sources.

This program needs a fortran compiler and a netcdf-fortran (and netcdf-c) to be installed on your system.

Under the *~/Path-To/BathyMaker* directory you will find two sub-directories :

- data_bathy : which contains an extraction of the global gebco bathymetry database and several other data files.
- bathy_maker : which contains the code and the scripts used to create a grid and interpolate the bathymetry on it.

Go to the ***bathy_maker*** directory

And list the content of the directory. (command ls)

In the bathy_maker directory you will find several files and in particular:

- compile main1.F90 module_bathymaker.F90 module_grid.F90 module_netcdf.F90 module_ww3.F90 : those files are fortran routines of the code and compile is a script that has to be adjusted in order to work on your machine
- input_file_list.txt, notebook_grid.f : are the input files used by the code to create the grid and the bathymetry
- Do_Image_Bathy.jnl and gnuplot_commandline.txt : are script and command line used to visualize the results.

2-First step : compiling the code

In order to compile you will have to edit the file : ***compile***.

In this executable file, there is a line starting by ifort ${inclink} etc... or gfortran ${inclink} etc…

You will have to change the compiler if you do not have the intel fortran compiler « ifort » installed on your machine, you can use gfortran instead for example. If the include and lib variables (inclink and/or liblink) are not working then you will have to give the right path to the include and the libraries for the NetCDF associated with **your** fortran compiler.

When you give the right information, you will just have to execute the script « compile » by using the command :

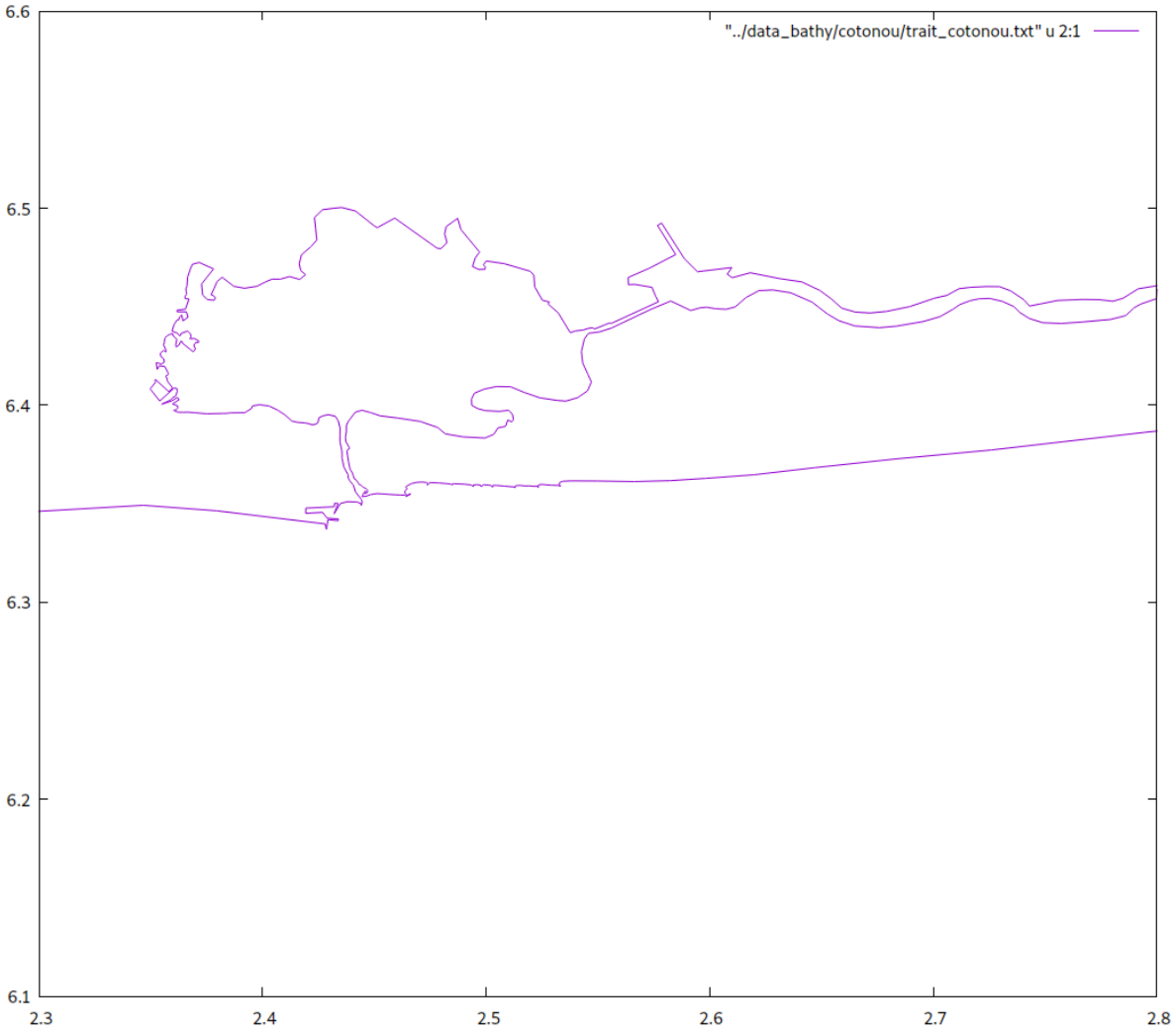
./compile

This should create a new executable file : ***main.exe***

3-Making a grid over Nokoue lake

The idea of the training course is to produce a regular grid over the Nokoue lake with a **250m** resolution. And then comparing it with a (similar in grid points) bipolar grid to see the difference.

3.1 Regular grid case :



Making the Grid :

The file ***notebook_grid.f*** will be adjusted in order to set a regular grid over the lake. In this file you will find several parameters to adjust.

- Edit this file using the editor of your choice (vi, gedit, kate ,etc...)

You will find inside several information ; we recall here only the ones you should look at for the exercise :

In the notebook_grid.f file these parameters are given by :

iglb=*200* *! dimension along the Oi axis (global grid)*
jglb=*200* *! dimension along the Oj axis (global grid)*
kmax=*1* *! dimension along the Ok vertical axis*

iglb and jglb are describing the number of grid points in the i and j direction, kmax is the number of vertical levels and has to be set at 1 to create the grid.

You will find parameter that will help to fix the grid size in X and Y direction.

dxb=*250.* *! Along Oi axis cellboxe size (m)*
dyb=*250.* *! Along Oj axis cellboxe size (m)*

dxb and dyb are the size in meters along the x and y direction (along grid axes respectively i and j) at the reference grid point determined below by the indexes (grid_i0,grid_j0). Depending on the type of the grid those parameters will have a different signification. We will see that in the following examples.

The position of the North and South Pole are given by the parameters :

northpole_lon=*0.* *! longitude (° decimal) of the grid nortth pole (antipode=9999.)*
northpole_lat=*90.* *! latitude (° decimal) of the grid nortth pole (antipode=9999.)*
southpole_lon=*9999.* *! longitude (° decimal) of the grid south pole (antipode=9999.)*
southpole_lat=*9999.* *! latitude (° decimal) of the grid south pole (antipode=9999.)*

Those northpole_lon, northpole_lat, southpole_lon and southpole_lat are the positions of the new north and south poles in the old referential. For monopolar grid and regular grid, you will only have to set the value of the south pole coordinates to 9999.

Then in reference to these new pole locations, the grid constructed around a reference grid point (i0,j0). This point is located in the grid space (or matrix point of view) by the grid index grid_i0 and grid_j0 (also described as i0,j0 sometime).

grid_i0=*100* *! i0=i(longi0,latit0)*
grid_j0=*100* *! j0=j(longi0,latit0)*

grid_i0 and grid_j0 are the indexes of the reference grid point.

NOTE : There is no need for this point to be inside the grid mesh. As you can see here we chose to put it just outside the final grid mesh. But in practice we often take that point to be at the grid point (1,jglb) for a monopolar grid because this point fixes the highest resolution of the grid, we want to have the best control on it. And it is easier to do it that way. For a bipolar grid, as a first approach of construction of the grid we suggest you to locate this point at the point (1,jglb/2) please look at the bipolar description to get more information about this.

The location of this point is given in the **new referential,** linked to the **new north and south poles**, by the latitude and longitude :

phi0=*6.400* *! Reference latitude for the Mercator projection*
latit0=*6.400* *! latitude of grid point (I0,J0) relative to north pole*
longi0=*2.500* *! longitude of grid point (I0,J0) relative to north pole*

latit0 and longi0 are the latitude and longitude of the reference grid point in the new referential associated to the new north pole.

Try with those parameters by using launch the bathymaker : **./main.exe**

First the program will ask you if you want to perform a grid/bathy creation manually or by using a file :

```
This program will ask you questions.
Do you want to answer them directly (type 1)
or use an answer file (type 2)?
```

The first use of the program you have to type **1**.

Afetr a first use of the bathymaker, the program had generated a text file : ***answer_file_history****.*

*Then if you want to remake the exact same actions, for exemple you made a little adjustment into the notebook_grid.f (let's say you only change the size of the grid iglb/iglb) then you can copy this file (****answer_file_history)*** *into an* ***answer_file****.*

*Then you will be able to chose the second choice "****2****", and the bathymaker will redo the exact same actions.*

If you had type 1 (or 2):

Then, you will see several information like the minimum and maximum grid size :

```
S grid latmin,latmax=    6.17286958909306        6.62927676160380
S grid lonmin,lonmax=    2.27149541769164        2.73076700391537
Verification de la transformation inverse
i,j,i=fi(lon,lat),j=fj(lon,lat):
          1           1  0.999999999999645       0.999999999996746
min dx dy=    249.886923786438        249.886923723349
max dx dy=    250.108136561574        250.108136497956
```

You have also a question to answer :

```
!--------------------------------------!
 Do you want to generate the bathymetry?
           type yes or no
!--------------------------------------!
```

At this stage you only generate the grid. You can type **no** and check the result of your computation.

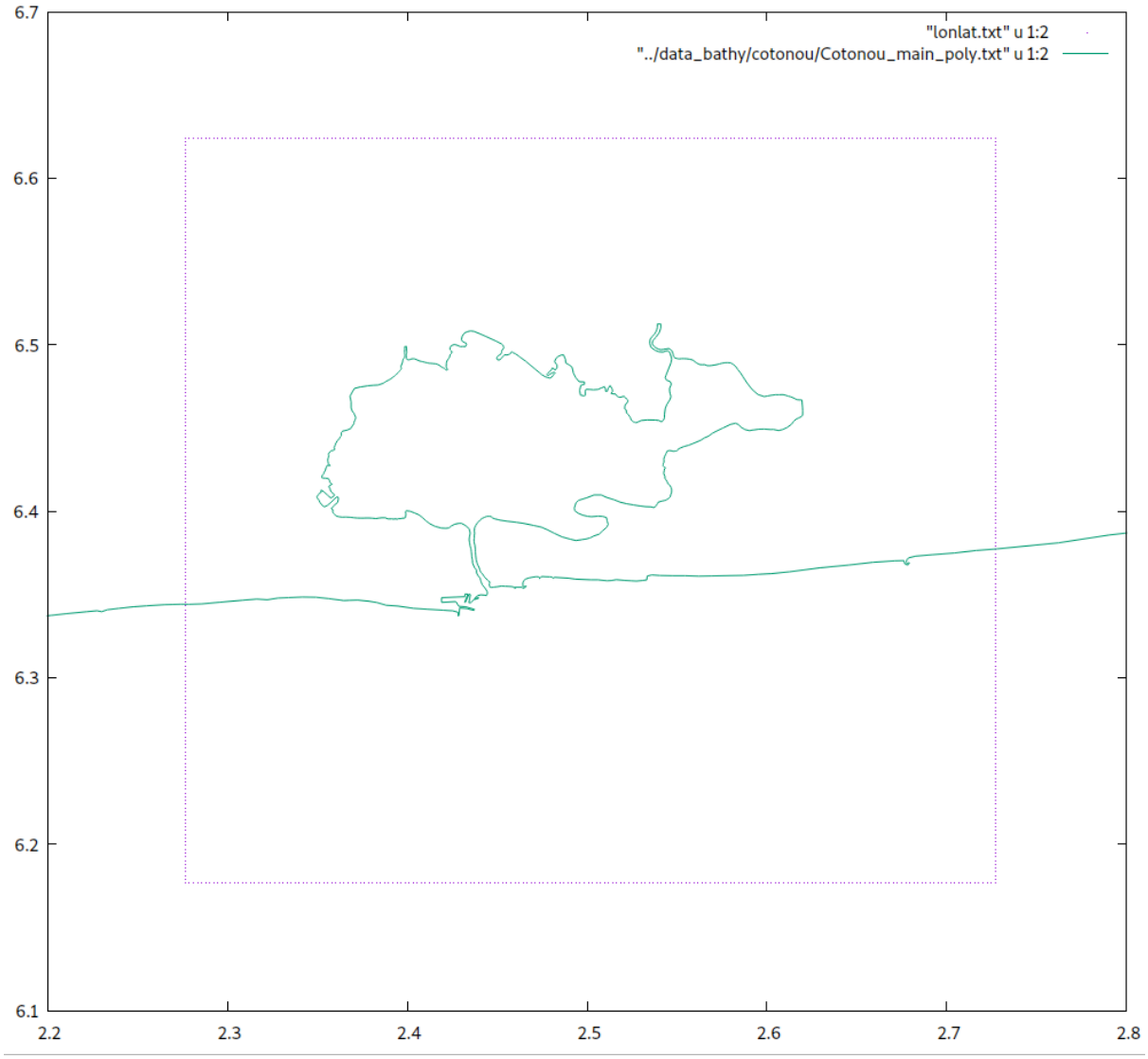
```
no
the end
```

The program produces several files, in particular : **lonlat.txt** that contains the longitude and latitude of the outline of the grid you just created.

To visualize the grid we propose you to use the gnuplot software by simply using the command : gnuplot

Then you can use the gnuplot line command :

plot [2.2:2.8] [6.1:6.7] "lonlat.txt" u 1:2 w d, "../data_bathy/cotonou/Cotonou_main_poly.txt" u 1:2 w l



HELP : you can find this command in the file **gnuplot_commandline.txt**.

With this graphic tool you can see the coastline in green and the outline of your grid in purple.

Type q for quitting gnuplot.

Now, you can make some adjustments on the parameters : grid_i0 , grid_j0 and also the iglb, jglb . And re-run ./main.exe to generate the changes.

After some tries, we propose you to use some preset values set in the file notebook_grid_reg250x250m.f , in order to generate the bathymetry with the bathymaker.

cp notebook_grid_reg250x250m.f notebook_grid.f

Generating the bathymetry :

Rerun : **./main.exe** and after the first part answer “yes” to the question :

```
!---------------------------------------!
 Do you want to generate the bathymetry?
            type yes or no
!---------------------------------------!
```

Type **yes** and enter to proceed

Then the next question will allow you to adjust the level for the land/sea mask with respect with a level of topography you can set by answering the question :

```
!---------------------------------------!
 Land/Sea mask will be computed from
 bathymetry zero meter level
 Do you want to change the reference level?
            type yes or no
!---------------------------------------!
```

type **no** for not changing the default value.

*If you type **yes** the programme will ask to enter a new value for the reflevel :*

```
!-------------------------------------!
 Enter a new reference level (meters):
 convention: H>0 in the sea
        if H>Hreference mask=1 (sea)
        if H<Hreference mask=0 (land)
 No file answer available: you have to answer now
-10
```

For example we type -10 here, the landsea mask will be set using this new bathymetry level. Everything “below” -10m will be set as a masked point. You have to remember that the bathymetry is definitely positive downward for SYMPHONIE when we are proceeding for the grid/bathy creation or reading. So using a negative value will set the mask to a potentially flooded area that SYMPHONIE will then see as a “sea” grid point. Useful for small littoral simulation for marine submersion cases for example.

At this step, the program is reading the bathymetric data given in ***input_file_list.txt*** . We wil see the content later.

```
Input file:
../data_bathy/cotonou/river_bathy_Nokoue_20170901.txt


---------------------------------
num_file_=1
file name=../data_bathy/cotonou/river_bathy_Nokoue_20170901.txt
trous autorisés
```

```
no smoothborder
Input file:
../data_bathy/cotonou/Data_bathy_nokoue

-----------------------------------
num_file_=2
file name=../data_bathy/cotonou/Data_bathy_nokoue
trous autorisés
no smoothborder
Input file:
../data_bathy/cotonou/Data_Cotonou_Mod-GEBCO.txt

-----------------------------------
num_file_=3
file name=../data_bathy/cotonou/Data_Cotonou_Mod-GEBCO.txt
trous autorisés
no smoothborder
Masque terre mer
```

Now the code will propose to use a list of polygons to make the landsea mask over the grid. If you want to use polygones to set the landsea mask give the name of the file **`Polygon_Cotonou.txt`**.

```
!--------------------------------------!
 Coastline using polygon files?
 If no type "no"
 If yes give the name of the polygon file list:
Polygon_Cotonou.txt

Polygon file ../data_bathy/cotonou/Cotonou_main_poly.txt
Polygon file ../data_bathy/cotonou/cotonou_ile1_poly.txt
Polygon file ../data_bathy/cotonou/cotonou_ile2_poly.txt
Polygon file ../data_bathy/cotonou/cotonou_ile3_poly.txt
Polygon file ../data_bathy/cotonou/cotonou_ile4_poly.txt
Polygon file ../data_bathy/cotonou/cotonou_ile5_poly.txt
Polygon file ../data_bathy/cotonou/cotonou_ile6_poly.txt
!--------------------------------------!
```

Then after that answer no to all the following questions :

```
!--------------------------------------!
 Do you want to remove secondary bassins
 and apply mask open boundary conditions?
            type yes or no
!--------------------------------------!
```

no

data_lonmin,data_lonmax 2.11195187707125 3.19256953572647

data_latmin,data_latmax 5.52478572404861 6.77896984761218

create gnuplot_s26_seanode file

create gnuplot_s26_landnode file

!---------------------------------------!

Do you want to create a WW3 grid with

the SYMPHONIE grid?

type yes or no

!---------------------------------------!

no

!---------------------------------------!

Default grid size is imax= 200 jmax= 200

Do you want to resize the grid dimensions?

type yes or no

!---------------------------------------!

no

!---------------------------------------!

Do you want a periodic domain along the Oi axis?

type yes or no

!---------------------------------------!

no

Ecriture du fichier ascii bathycote_in.dat

Ecriture du fichier ascii bathycote_in.ijh

Ecriture du fichier ascii lonlat_4col.txt

Ecriture du fichier ascii obc.lonlat

Ecriture du fichier bathycote_in.nc

nf_enddef ok

longitude_t ok

latitude_t ok

mask_t ok

h_w ok

hm_w ok

dhdx ok

dhdy ok

!---------------------------------------!

Do you want to make a low resolution file: yes or no

no

!---!

Do you want to create mpi distribution files

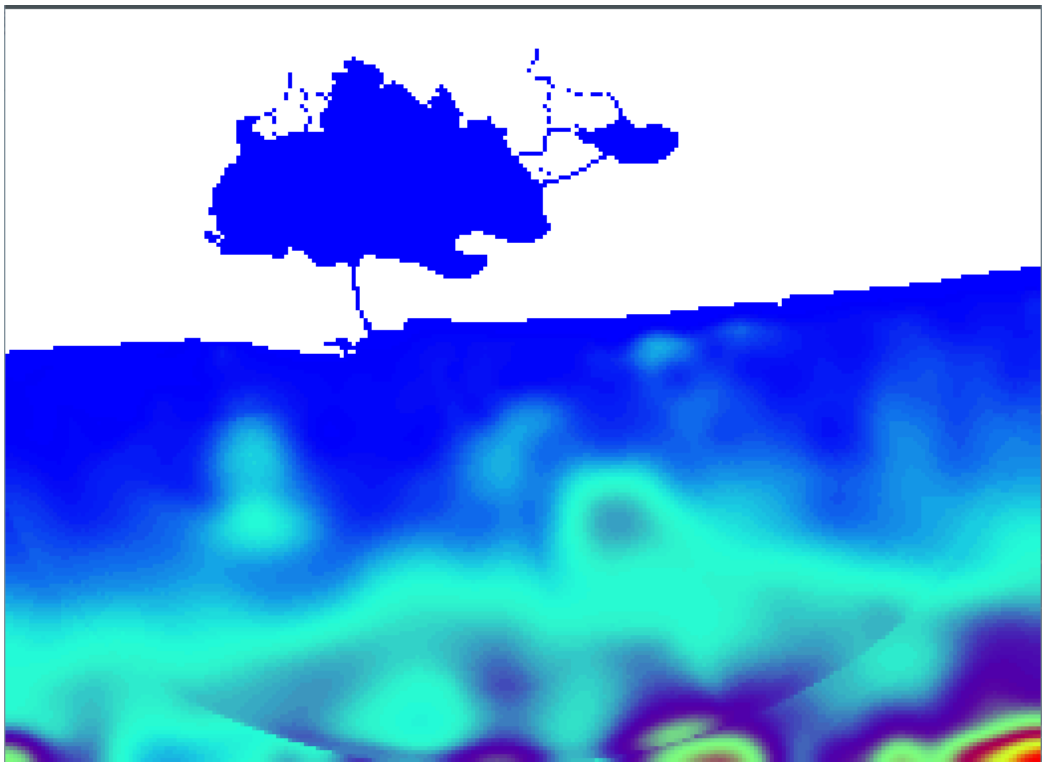
description_domaine.next and description_trous.txt?

type yes or no

```
!-------------------------------------------------!
no
Total points:                          40000
Nb of sea points:                          0
Nb of sea point with dx/735<2:             0
m**2 of sea points:                0.000000000000000E+000
m**2 of sea point with dx/735<2:   0.000000000000000E+000
fin
```

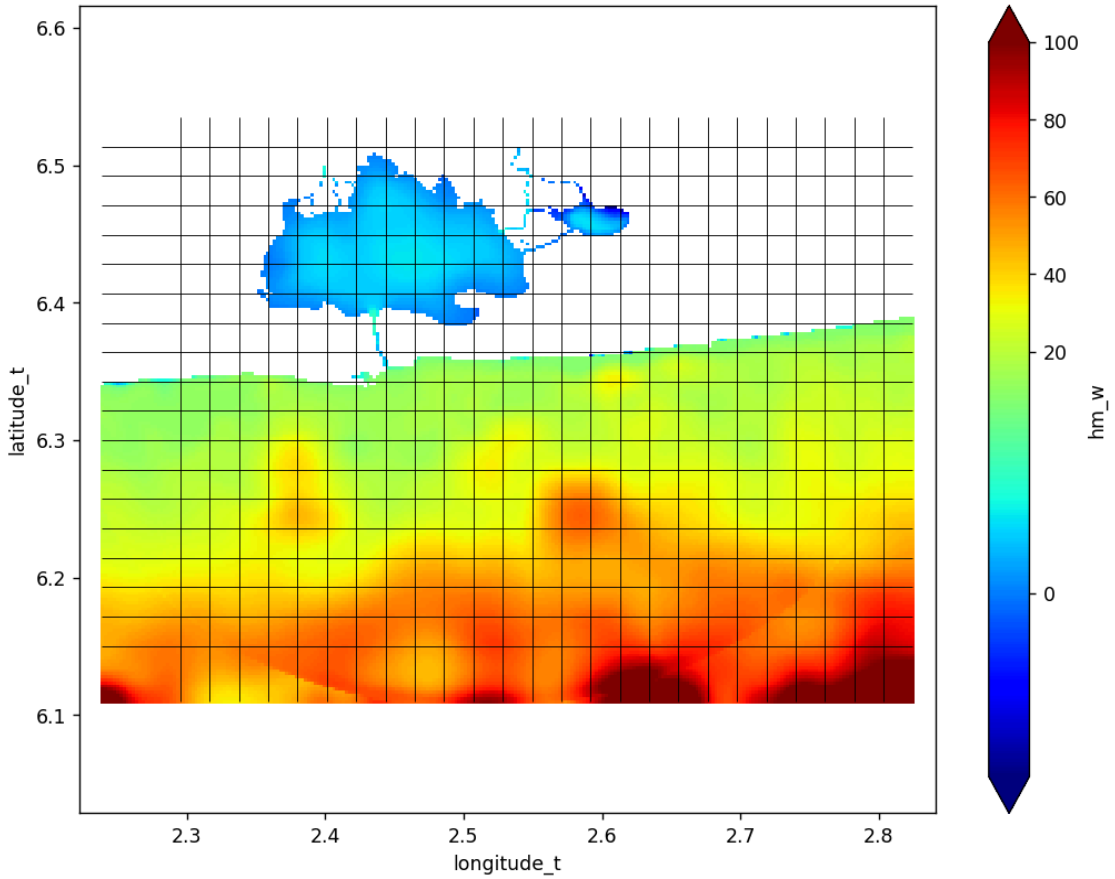
At this state the code have generate several files, the one that you can look at is bathycote_in.nc : use ncview :

ncview bathycote_in.nc



Or python : **Bathy_visu.py** script (with the Spytools_Env conda environment) with a name of experience : for example “regular”. Then the output file will be grid_regular.png .

python Bathy_visu.py regular



Description of the input_file_list.txt and Polygon list file:

input_file_list.txt

In the file ***input_file_list.txt*** , you will find one (or more) bloc of information before the word “FIN” that end the input read of the code:

```
.....................................
../data_bathy/cotonou/river_bathy_Nokoue_20170901.txt
lat lon h
50.         ! Si resol n'est pas fournie donner une valeur
2.          ! facteur applique a la resol
1           ! 1=h>0 -1=h<0
1.0         ! Indice de confiance
0.          ! H offset (en metres)
0.          ! -0.0043609 ! Longitude offset (degres)
0.          ! -0.0056123 ! Latitude  offset (degres)
trous autorisés
0 smoothborder
.....................................
../data_bathy/cotonou/Data_bathy_nokoue
lon lat h resol
```

```
-999.       ! Si resol n'est pas fournie donner une valeur
1.2         ! facteur applique a la resol
1           ! 1=h>0 -1=h<0
0.1         ! Indice de confiance
0.          ! H offset (en metres)
0.          ! -0.0043609 ! Longitude offset (degres)
0.          ! -0.0056123 ! Latitude  offset (degres)
trous autorisés
0 smoothborder
.....................................
../data_bathy/cotonou/Data_Cotonou_Mod-GEBCO.txt
lon lat h resol
-999.       ! Si resol n'est pas fournie donner une valeur
1.2         ! facteur applique a la resol
1           ! 1=h>0 -1=h<0
0.001       ! Indice de confiance
0.          ! H offset (en metres)
0.          ! -0.0043609 ! Longitude offset (degres)
0.          ! -0.0056123 ! Latitude  offset (degres)
trous autorisés
0 smoothborder
.....................................
FIN
```

1st line is just a separation line
2nd give the dataset name including path
3rd is the format of the data file, there is "**lon lat h resol**" or "**lat lon h**" format recognized
4th if there is no resolution given in the file you have to fix it here.
5th you can apply a factor to increase the resolution of the data (bigger resolution will smooth the gradient to bathymetry a little)
6th vertical coordinate convention : if depth is given positively downward put "1" if it is upward the put "-1"
7th Give the trust index in you data file, we will explain further.
8th you can add an offset on your data set depth
9th and 10th you can also add horizontal offset to the data set.
11th you can choose to authorize or not holes in your data set for interpolation. If you allows holes, the code will fill them with interpolation routines.
Finally if you use several datasets you can smooth the change from one to the others.
It is important that the word : **FIN** is set into the file. The code will check about it.

About the trust index, line 7, give the bigger number (closer to 1) to the dataset you trust the more.

Polygon_file.txt

In the Polygon_Cotonou.txt, you will found several lignes :

```
0
1 ../data_bathy/cotonou/Cotonou_main_poly.txt
0 ../data_bathy/cotonou/cotonou_ile1_poly.txt
0 ../data_bathy/cotonou/cotonou_ile2_poly.txt
0 ../data_bathy/cotonou/cotonou_ile3_poly.txt
0 ../data_bathy/cotonou/cotonou_ile4_poly.txt
0 ../data_bathy/cotonou/cotonou_ile5_poly.txt
0 ../data_bathy/cotonou/cotonou_ile6_poly.txt
```

The first line is 0 or 1 : 0 the landsea mask is reset to land everywhere, 1 the landsea mask calculated from the interpolated bathymetry is kept.
Here the idea is to reconstruct the complete landsea mask so we choose 0 to start over.
Next lines are working the same way : 0 or 1 and a path to a polygon file (text format too, we will give an explanation below : Format expected for polygon file). The first value will be used to set the value of each grid point found **inside** the related polygon.
So in the example, the mask is set to 0 everywhere with respect to the first line parameter, then a coastline polygon will set the "ocean" part of the grid domain by setting the mask to 1 inside the polygon ("../Relative/path/to/polygon_coastline.txt"). Then for the following 3 lines, for each island polygon given, the mask will be set to 0.

3.2 Bipolar grid case :

For the Bipolar case, we propose to go straight to the good parameter given in the notebook_grid_bip.f . Simply copy this file into the notebook_grid.f :

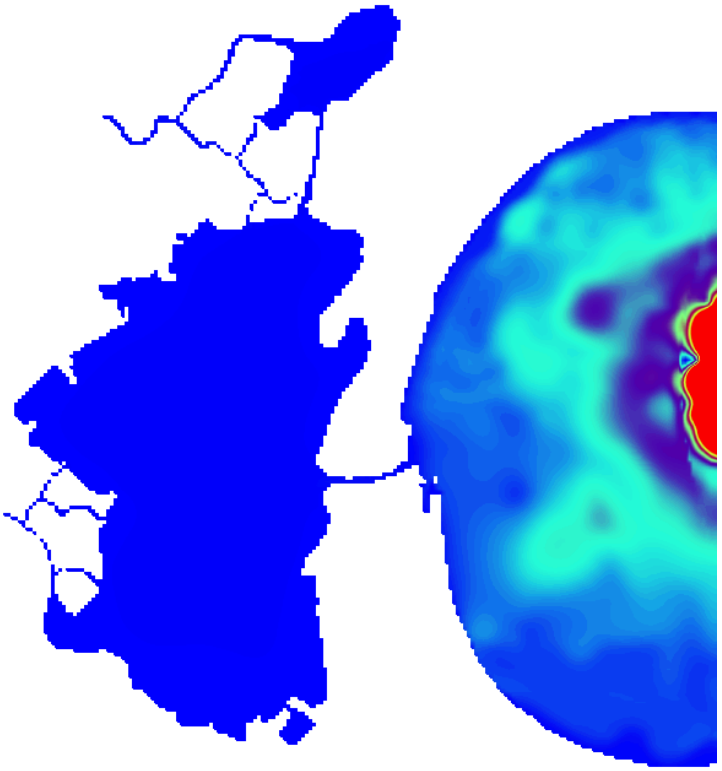
cp notebook_grid_bip.f notebook_grid.f

For more explanation you can refer later to the 4th section of this document dealing in details with the grid type description and how to start the right way.

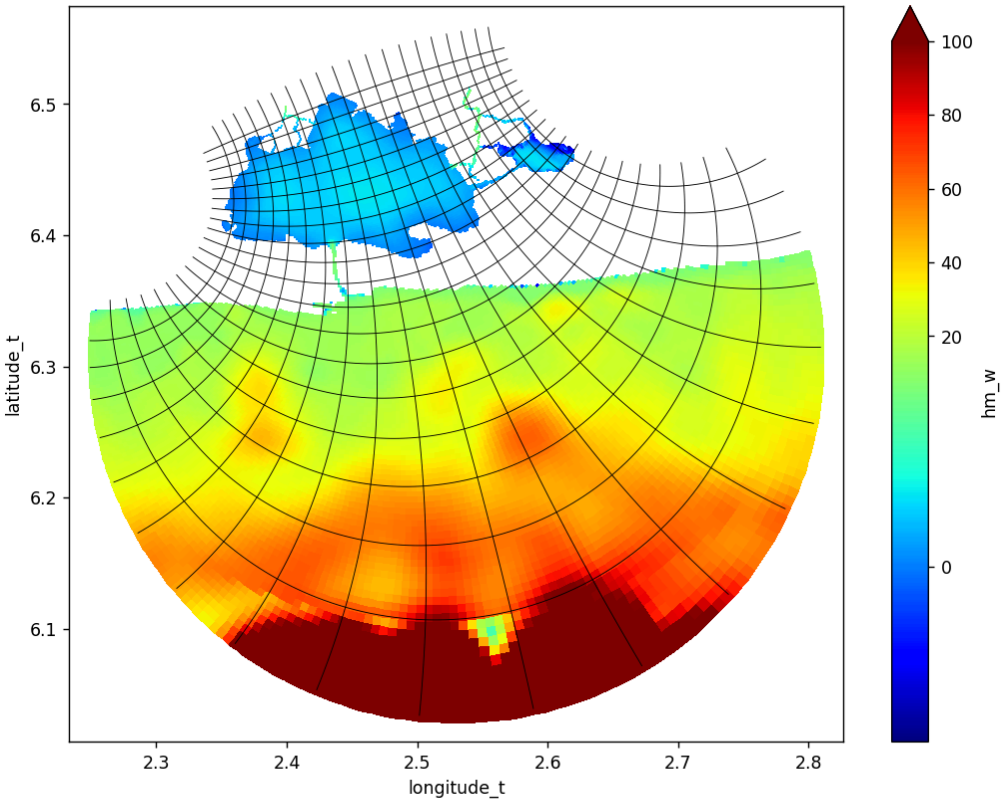
Then re-run the bathymaker command : **./main.exe** .
You can use the same input_file_list.txt and Polygon_Cotonou.txt inputs.

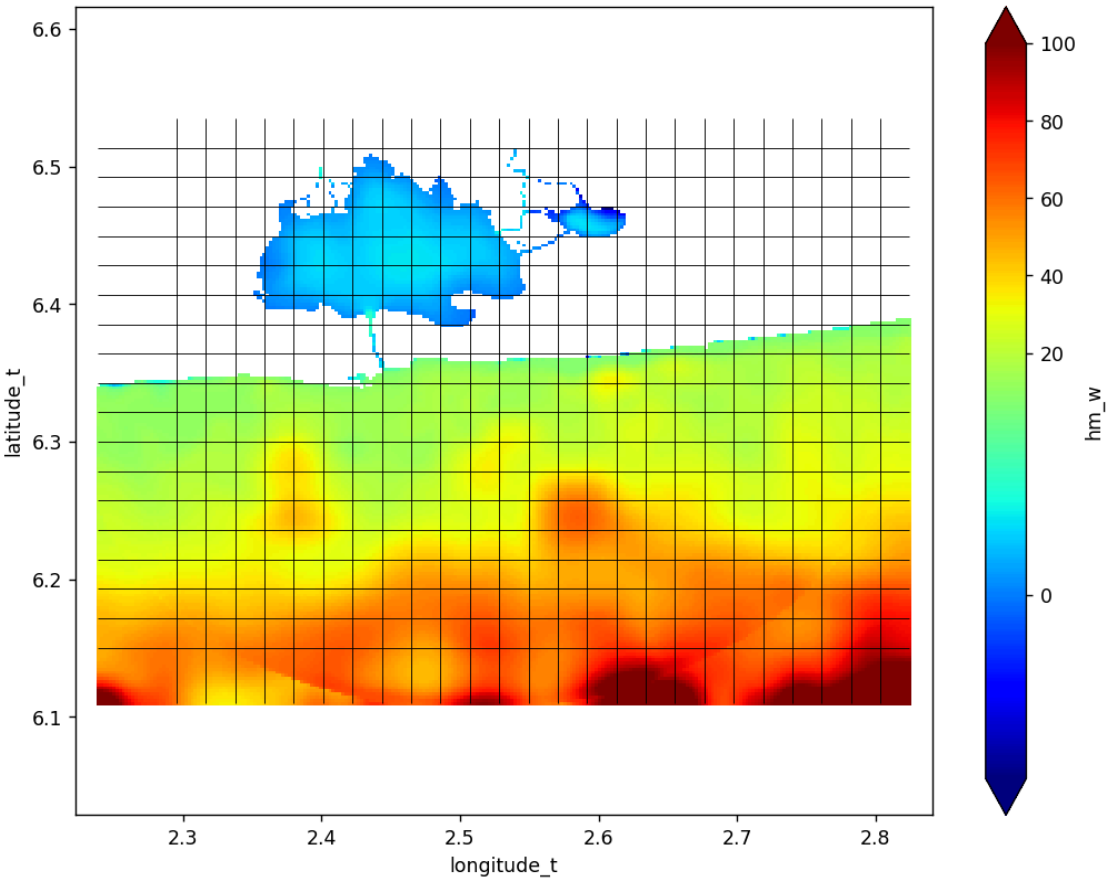
Then you can visualize the results with ncview and python :

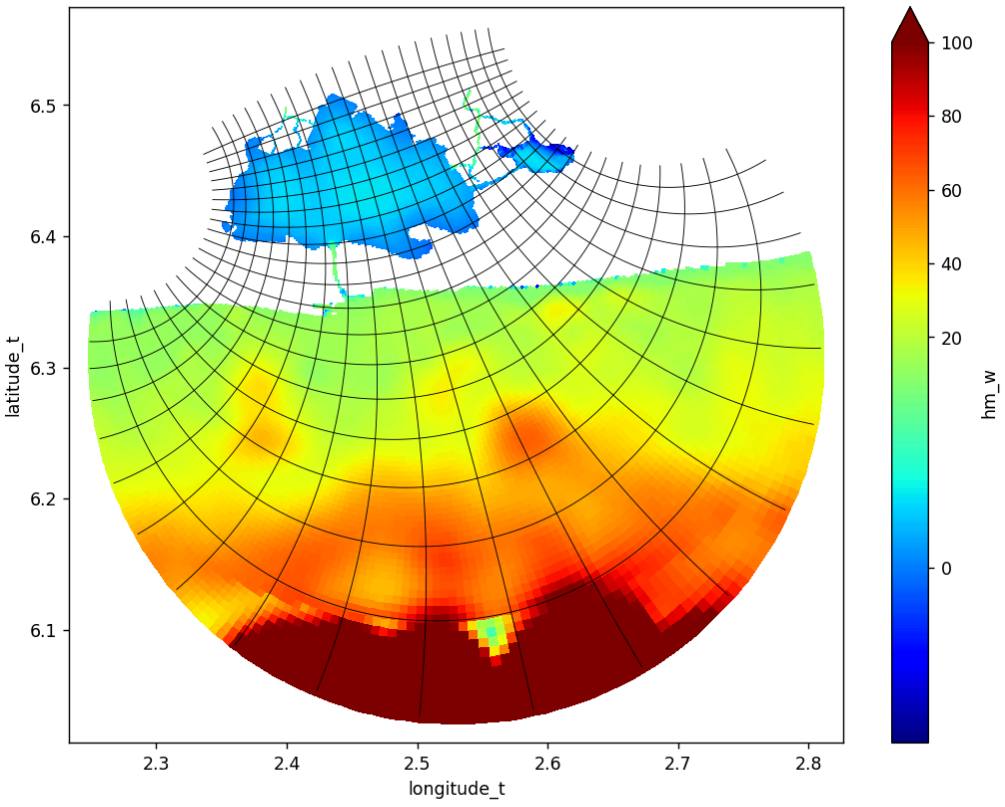
ncview bathycote_in.nc



python Bathy_visu.py Bipolar




6.6
6.5
6.4
6.3
6.2
6.1
latitude_t
2.3
2.4
2.5
2.6
2.7
2.8
longitude_t
100
80
60
40
20
0
hm_w


6.5
6.4
6.3
6.2
6.1
latitude_t
2.3
2.4
2.5
2.6
2.7
2.8
longitude_t
100
80
60
40
20
0
hm_w

4-Grid description

4-1 Standard Monopolar grid

For a standard monopolar grid, after you choose your new north pole location, you will have to set the position of the reference point in the grid (I0,J0) and in the new geographical referential (link to your new north pole of course). Then, the value of dxb and dyb will set the value of the grid resolution at this reference point. Because of the nature of the polar grid, the resolution will vary in function of the latitude of the grid point in the new referential.

In Fig 1 and 2, you will notice more clearly that the longitude of the reference grid point is counted from the northpole_lon parameter, and the latitude of the reference grid point is given also relatively to the new referential.
There is an important observation you should keep in mind: Phi0 is a parameter to perform the Mercator projection. The **best use is to set it to the same value as latit0** in order to ensure that the size of the reference grid point mesh is effectively set at the dxb and dyb parameters.

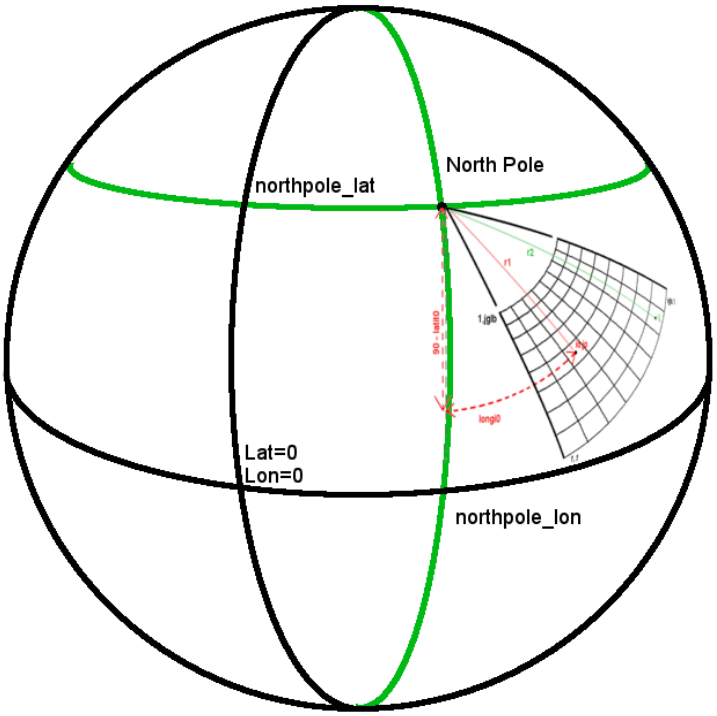


Figure 1 : North pole position on the sphere

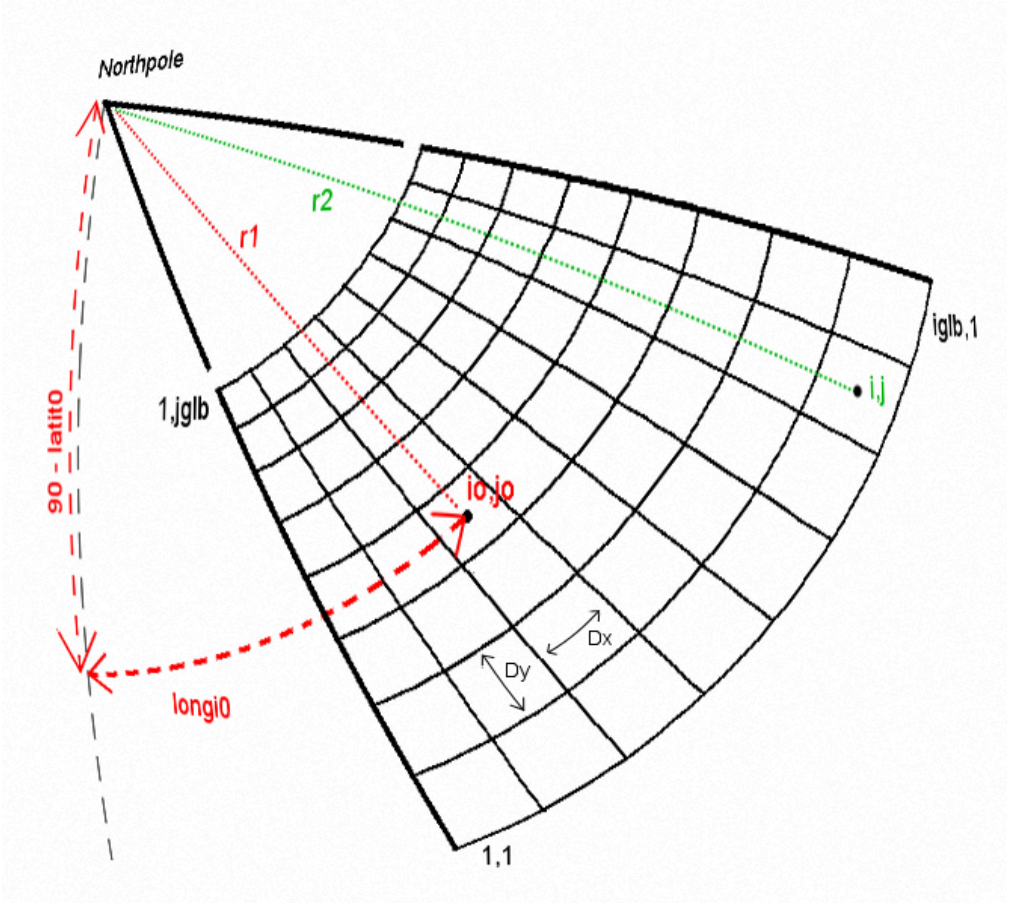


Figure 2 : Grid description for SYMPHONIE

Because of its nature, the single pole grid sees its resolution varying. On Fig. 2 you can see that the mesh grids are bigger for the small value of j index. The increase of the grid mesh is linked to the distance from the pole. Under the small angle approximation (i.e. for a small angle theta we can approximate the sine of theta by theta itself at the first order), the increasing rate of the mesh size between two grid points, for example i0,j0 and i,j (in green), is given by : dx(i,j)=dx(i0,j0)*(r2/r1) and dy(i,j)=dy(i0,j0)*(r2/r1). Keep in mind that this explanation is valid only when you remain close to the pole.
The variation of the grid is extremely handy when you want high resolution in interest region and a large grid coverage without using a huge grid which will be very costly in computing time.

4-2 Standard “Regular” grid

To create a “régular” grid, the work is similar to the monopolar grid at the particularity that the north pole of the grid will be the real geographical north pole so with the value 90 for the latitude and 0 for the longitude :

northpole_lon=0 *! longitude (° decimal) of the grid north pole (antipode=9999.)*
northpole_lat=90. *! latitude (° decimal) of the grid north pole (antipode=9999.)*

Then the rest of the explanation for the monopolar grid is still valid.

4-3 Bipolar grid

More challenging, the bipolar grid is a little more complicated to understand at first but it is not that complicated.

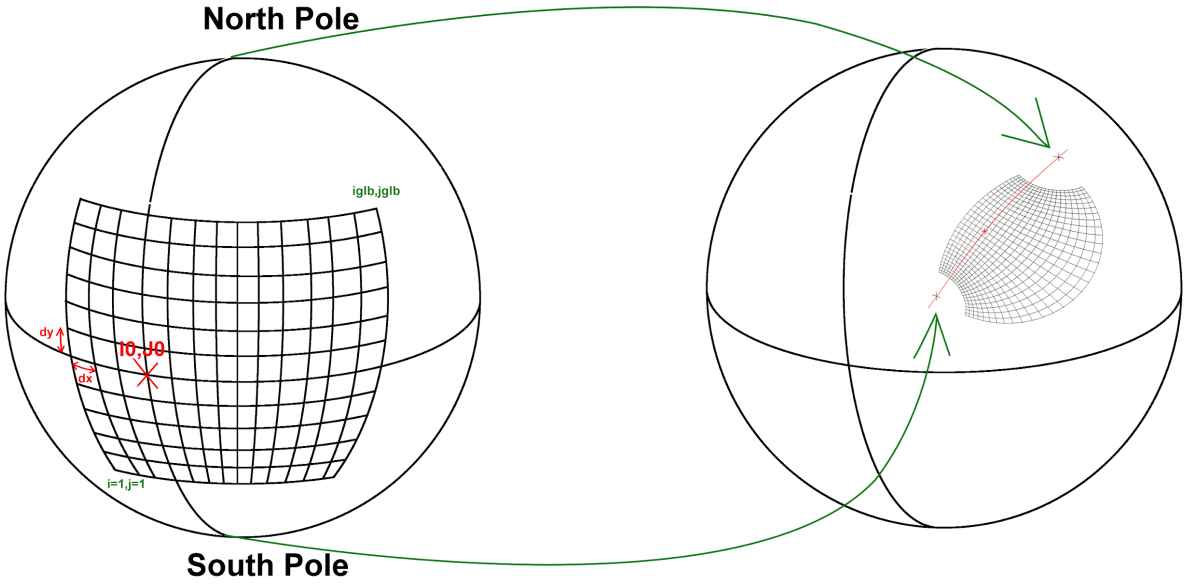


Figure 3 : North pole / South Pole positions and transformation on the sphere

In the case of the monopolar grid a good way to start is to set the north pole then the reference point (i0,j0) in the location where you want the resolution you fix with the dxb and dyb values. Here the easy way is to first imagine the grid on the “original” sphere and set the reference point in between the two poles , set i0 =0 and j0=jglb/2. In the news referential this point is on the equator, so the value if lati0=0. and loni0=0. You will have to change the coordinates of the news poles so the original grid (left

picture on fig. 3) will be transformed into the news grid (right picture on fig. 3). The tricky part here is to set the value of dxb and byb. In monopolar grid it would fix the value of the grid mesh of the final grid (the one transform), but in bipolar those parameters are giving the value of the mesh on the original grid.

So if you set a grid of 200 points in the J direction and you want the final grid to look like fig4. So with the side of the grid at -/+50° on both sides of the center of the grid. So on the original grid the 100th point from the middle should be at 50° in latitude. On the earth this latitude represent approximately 50*110km ~ 5500km so for 100 points you need to set at least 55000m for dxb and dyb as a first guess.

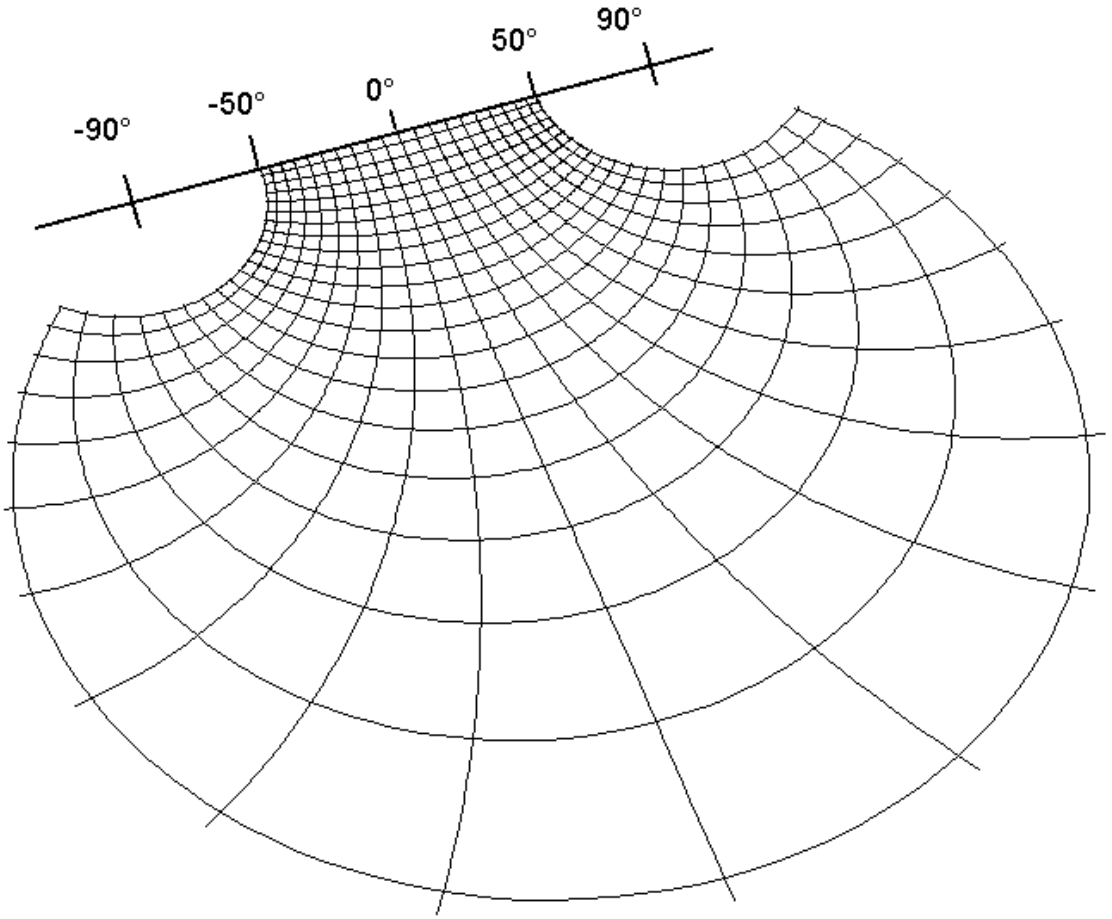


Figure 4 : Example of Bipolar grid.

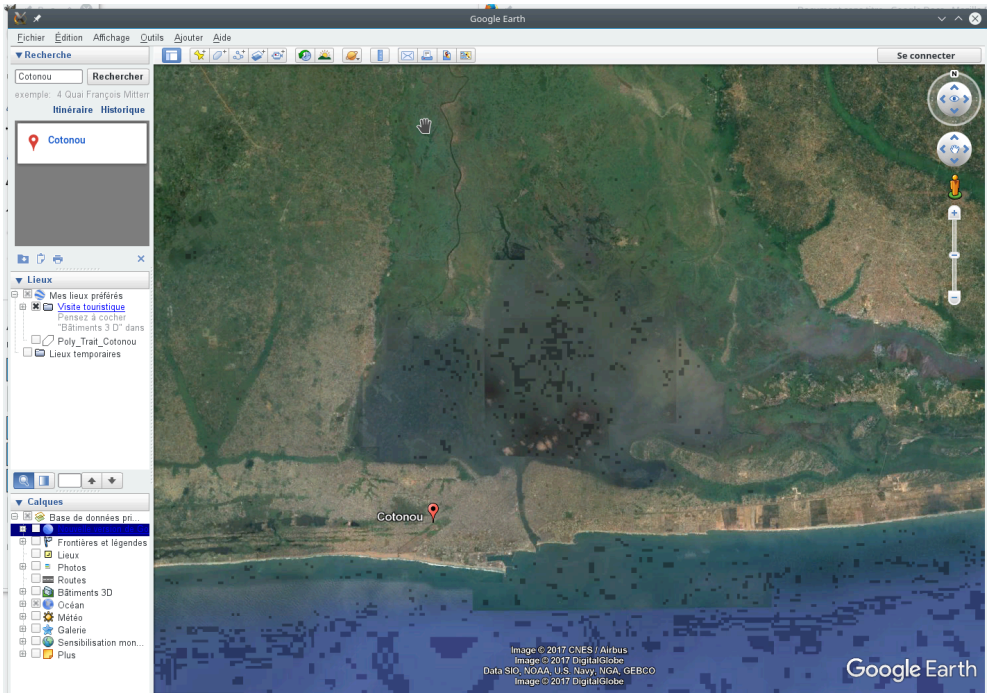
In fast you will need to put a bigger dxb and dyb because this is going to set the value of the grid at the reference point. But because the grid in getting smaller when you reach the poles, it will cover a small distance. So you will have to compensate by using a bigger dxb and dyb. Let's said 80000 instead.

After that it is a question of trial and you will get the perfect grid you want.

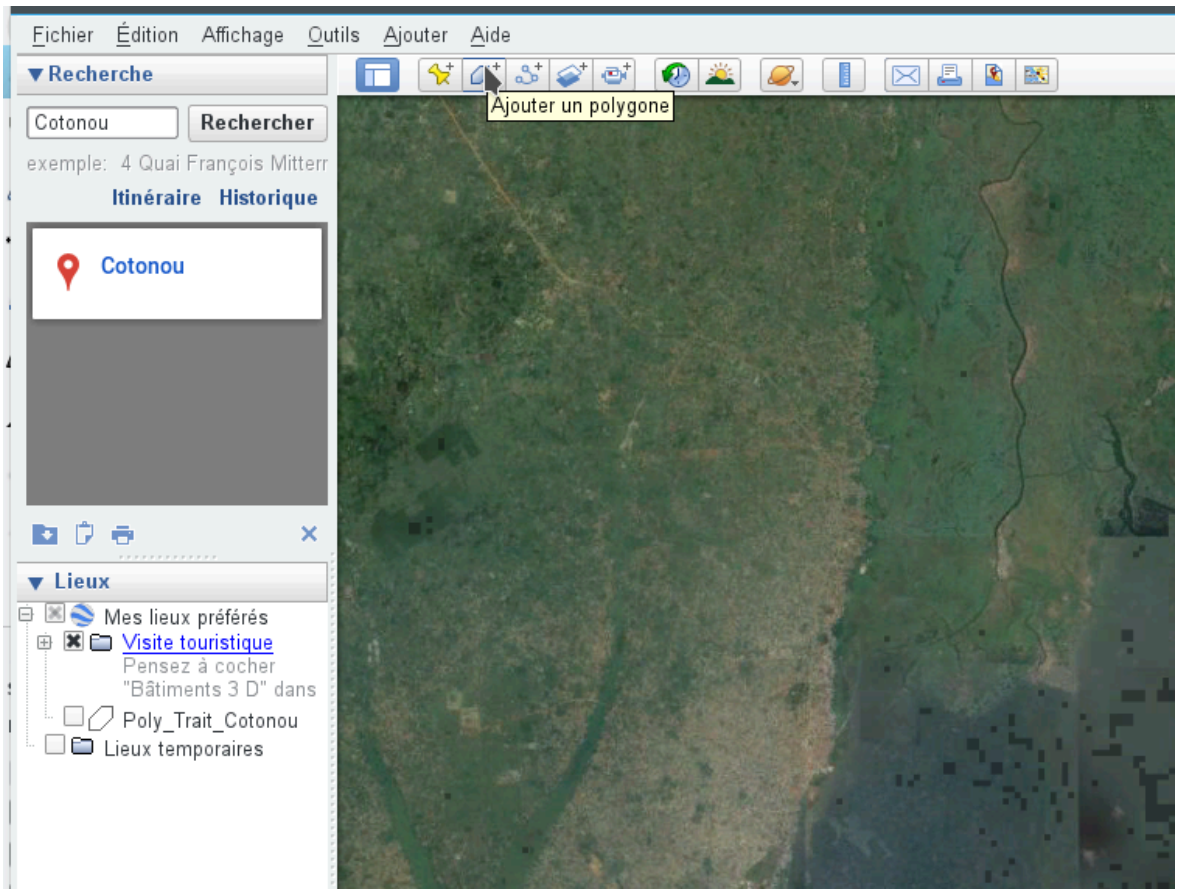
Annex :

How to create a polygon from Google-Earth :

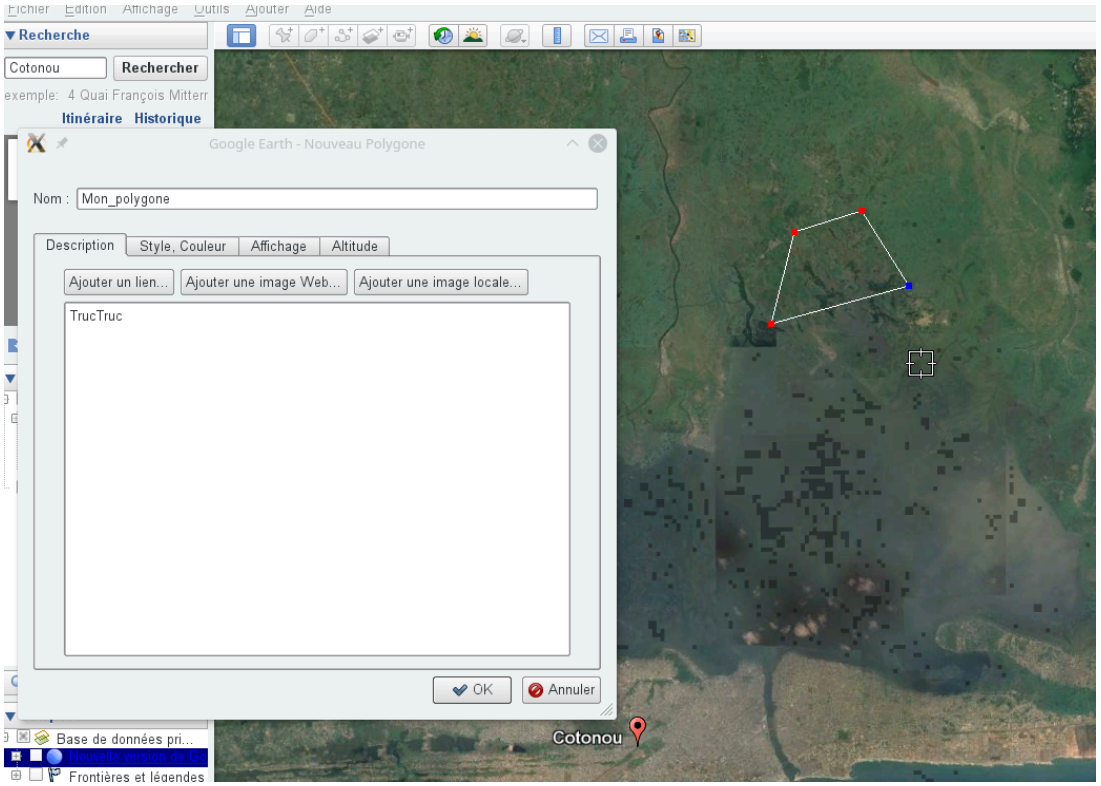
First Run Google-Earth and zoom on the region you are interested. For example : Cotonou!



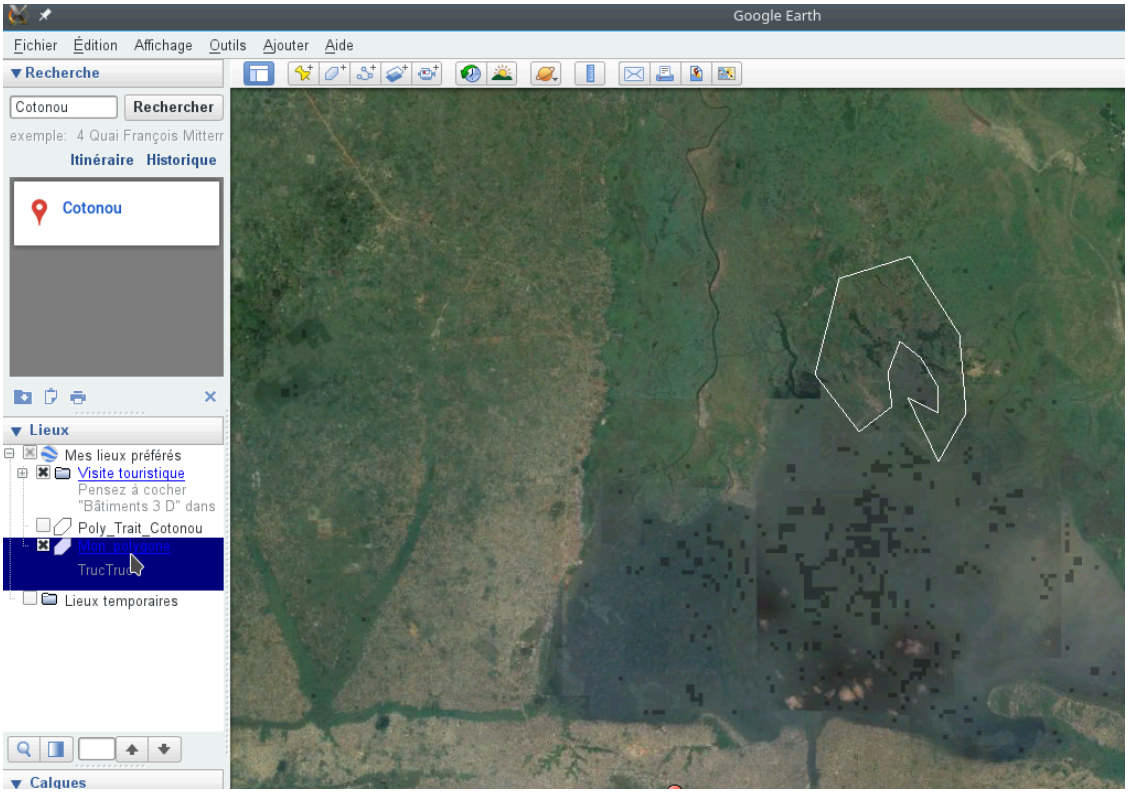
Select the polygon tool



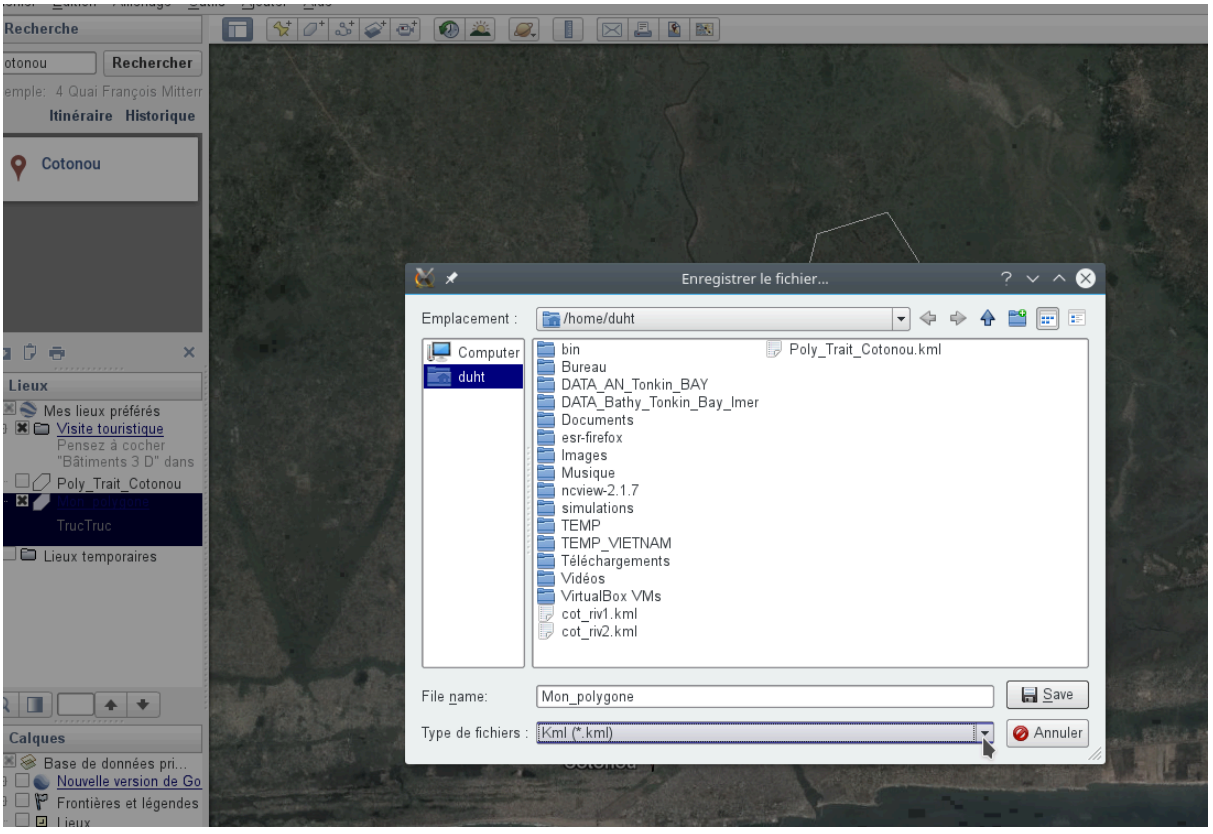
A windows will pop, you can give a name to your polygon and a description, but here it doesn't matter. **Without click on "OK"** start drawing your polygon by clicking using the cross. The blue dot is the last you created or move. You can fix the polygon during its creation but not after Clicking "OK".



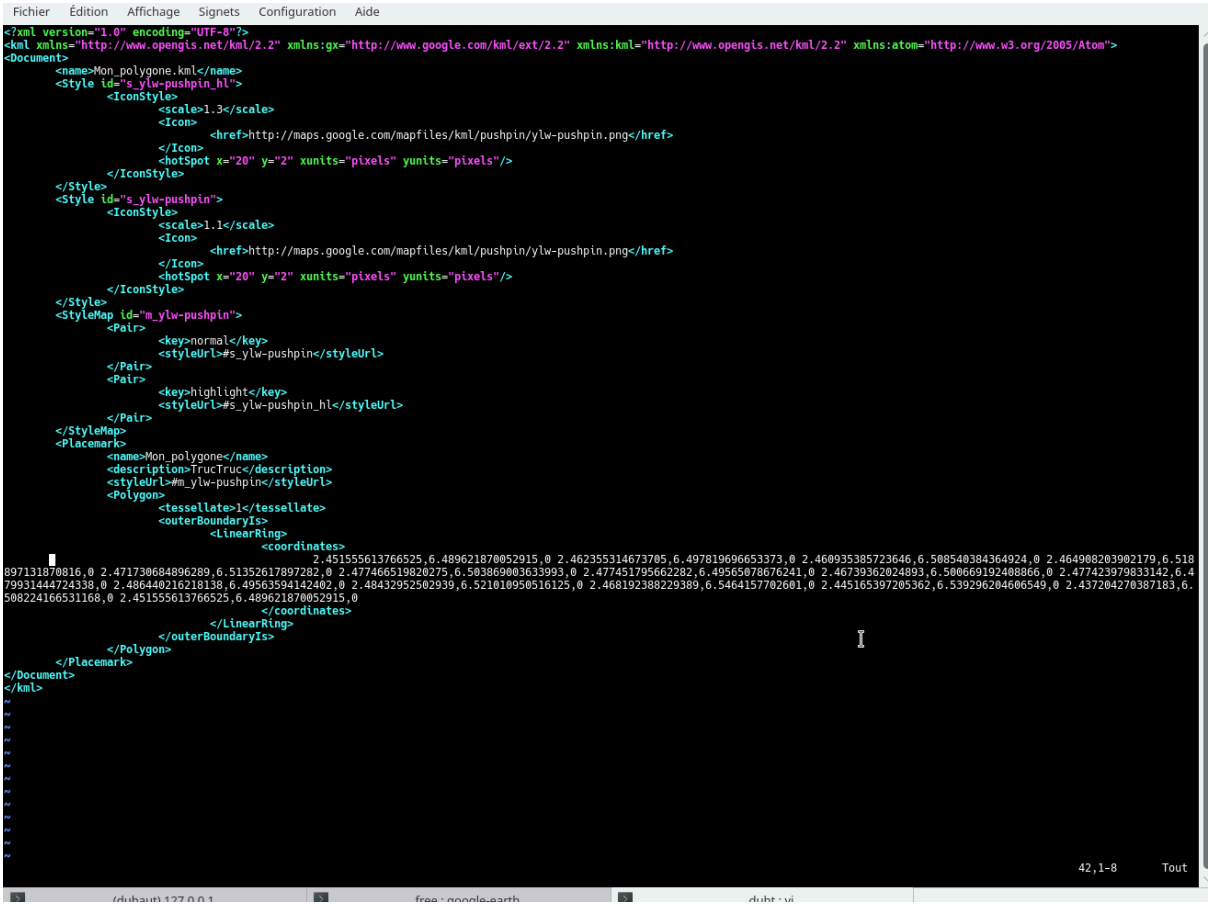
Then you can go for whatever you want in size and number of points (there is probably a limit…) Click on “OK” this time, your polygon is set into the left panel.



Right-Click on it and save it in a kml format :



If you open the kml file you will get something like that :



The information we want are set on the line 42 in my example. It is all the polygon point coordinates given in “lon,lat,0 “ format. So I design a little script in bash commands to change this file into a readable polygon format for code to create mangrove mask.

```
#!/bin/bash

# ce bash aide a découper les fichiers kml issue de google-earth pour les polygones
# A partir d'une liste de fichier *.kml il les découpe et les transformes en fichier ascii
# qui seront utiliser pour lire dans l'outil de réalisation de masque de trait de trait de côte ou
# de mangrove.
# Attention Il faut donner être sur que les fichiers kml respectent le même format :
# ici la ligne 42 est celle qui nous interesse

ligne=42

#for file in *.kml ; do
for file in Mon_polygone.kml ; do

  echo $file
  head -n $ligne $file | tail -n 1 > toto

  # On retire les tabulations et on cree la balise de découpe X
  sed -i -e 's/\t//g' toto
  sed -i -e 's/0 /X/g' toto

  tata=`cat toto`

  # On récupère le nombre de point du polygone
  NB=`grep -o X toto | wc -l`
  echo $NB > toto

  #Sur une chaine de charactère $tata, la découper en morçeau suivant le charactère "X" et mise ligne par ligne dans un fichier toto
  for i in `echo $tata | tr "X" " "`; do echo $i; done >> toto
  # On remplace les "," par des " "
  sed -i -e 's/,/ /g' toto

  # ecriture du fichier de sortie
  file_out=`basename $file .kml`"_poly.txt"
  mv toto $file_out

done
```

This code will set the polygon “Mon_polygone.kml” into a “Mon_polygone.txt” ascii file formatted for the “mangrove mask maker” code.

Format expected for polygon file :

Each polygon file must be ascii (text) format with :

- First line : the number of nodes of the polygon
- the following lines : longitude latitude of each nodes (one node per line)
- The last node must be the same node as the first given in order to close the polygon.