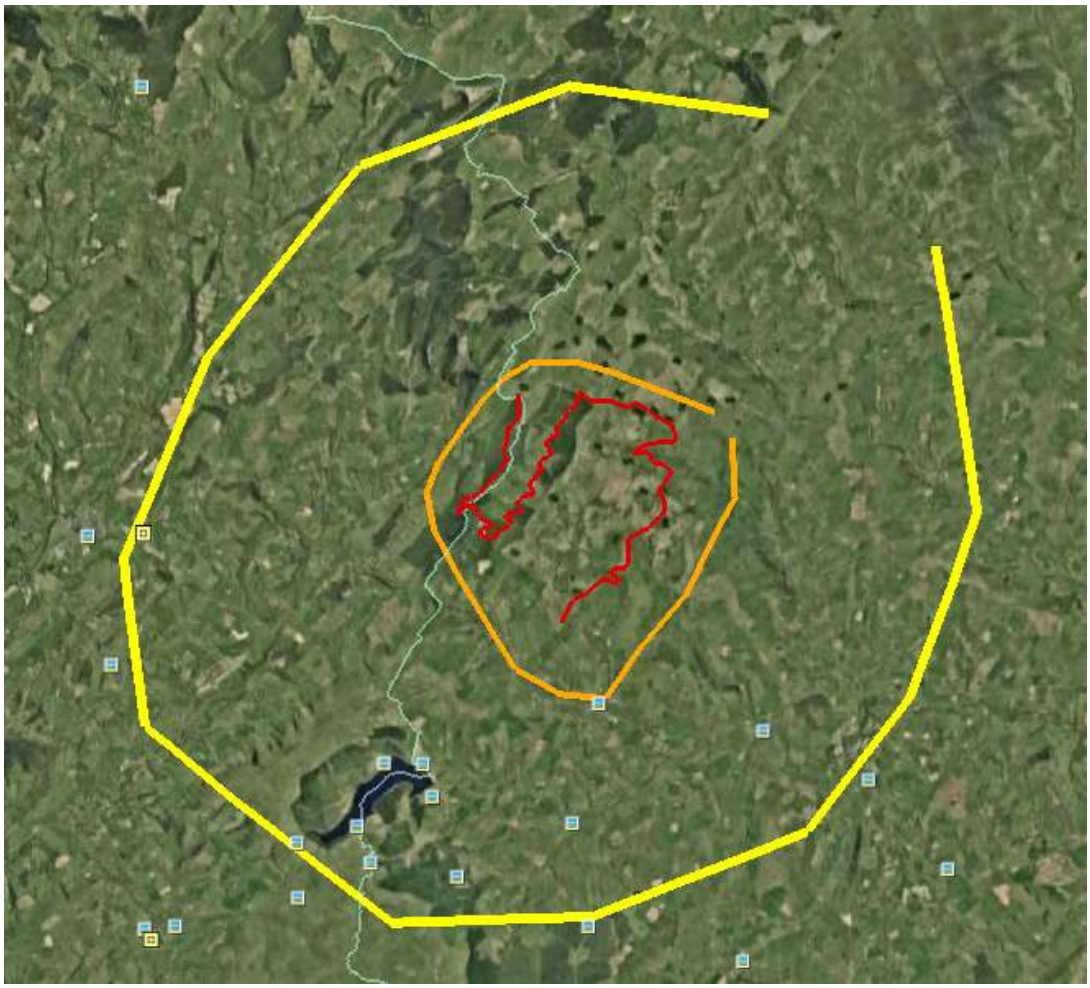
Starting a project with “start” command

First of all I have to create the kml of the route, with a lot of detail, so the shape is well defined.

I save it as **"road.kml" inside s0_import** (red route in the figure)

Then I create 2 more kmls:

1) "**limits.kml**" (orange route) delimiting the terrain for the main route plus all the detours. I save it inside **s2_elevation**
2) "**limits_b.kml**" (yellow route) delimiting the terrain for all the project. I save it inside **s2_elevation_b**



When I am ready, I start Octave

```
> cd c:\project
> addpath('c:\project\scripts')
> start
```

start processes road.kml, limits.kml and limits_b.kml, calling importakml and make_grid scripts.

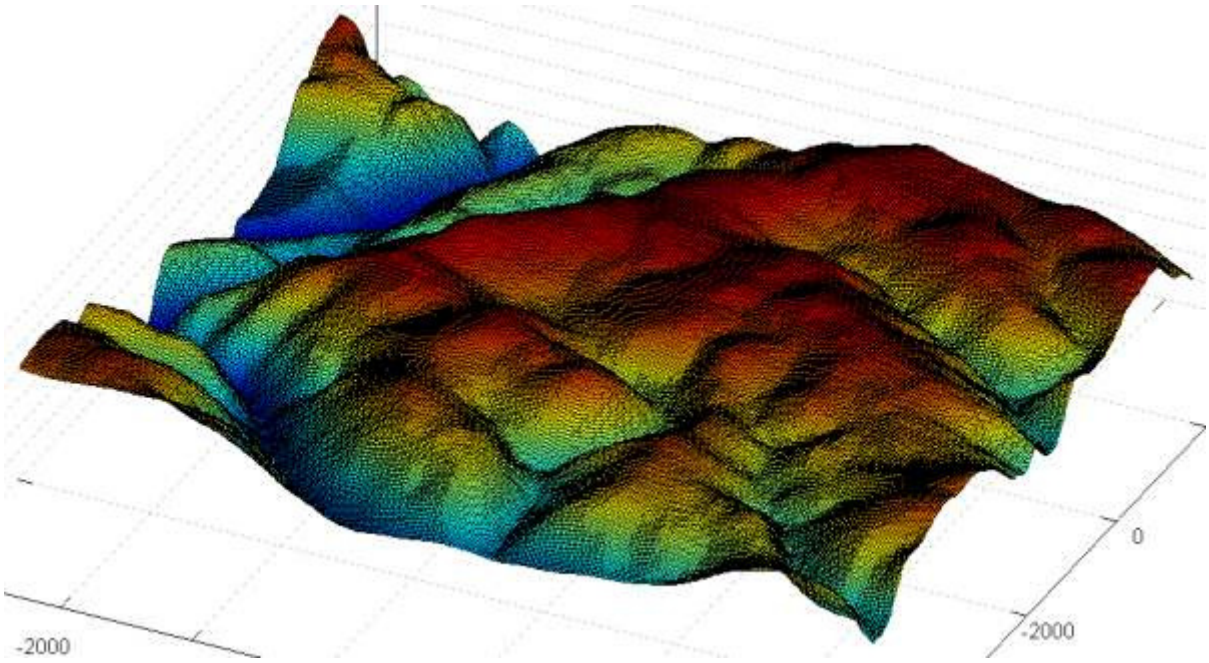
Now the gridXXX.kml are waiting inside s2_elevation\salida and s2_elevation_b\salida to be raised with BTBLofty or raise_kml. By default each kml is limited to 5000 points,and 25 and 75m are used as step for the grids. If you want to use other values, you can use them as parameters for "start" (i.e. **start(25,75,5000)**)

I use BTBLofty or raise_kml to raise the kmls, saving gridXXX_relleno.kml files.

> cd ..\s2_elevation
> **read_grid**

I want to see the graph:

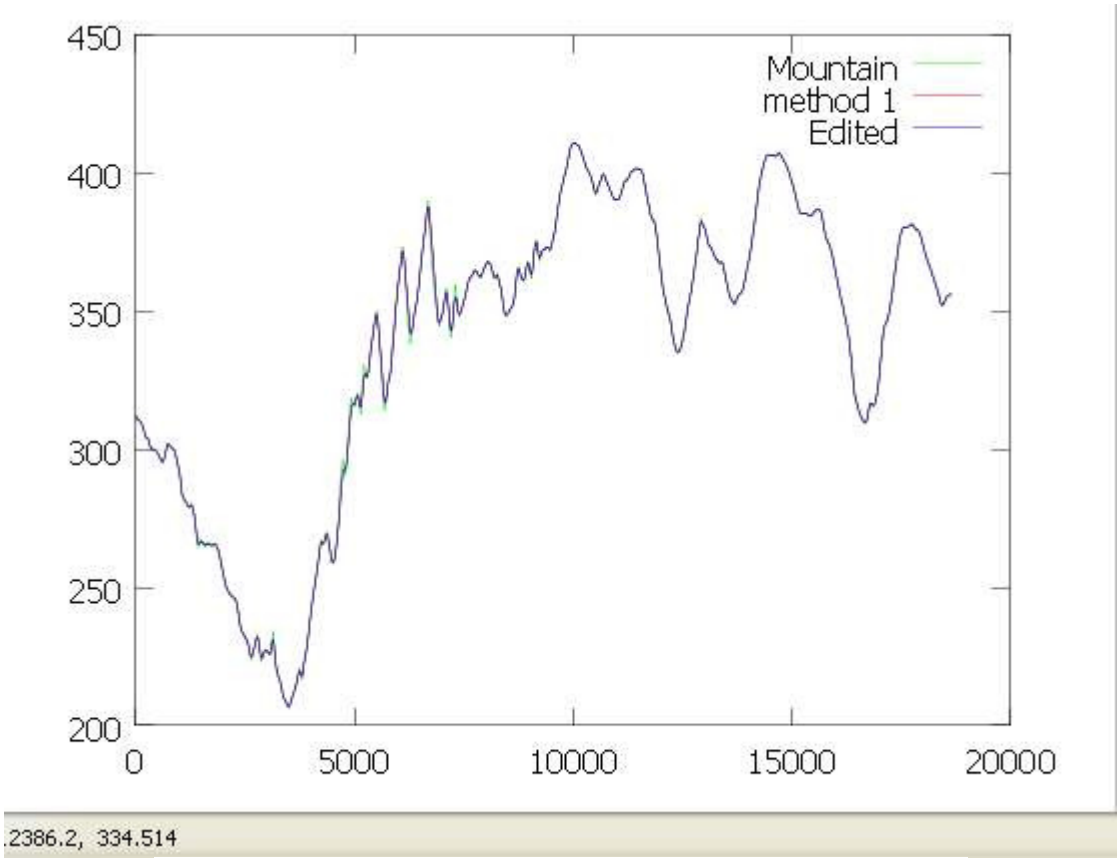
> **plot_lamalla**

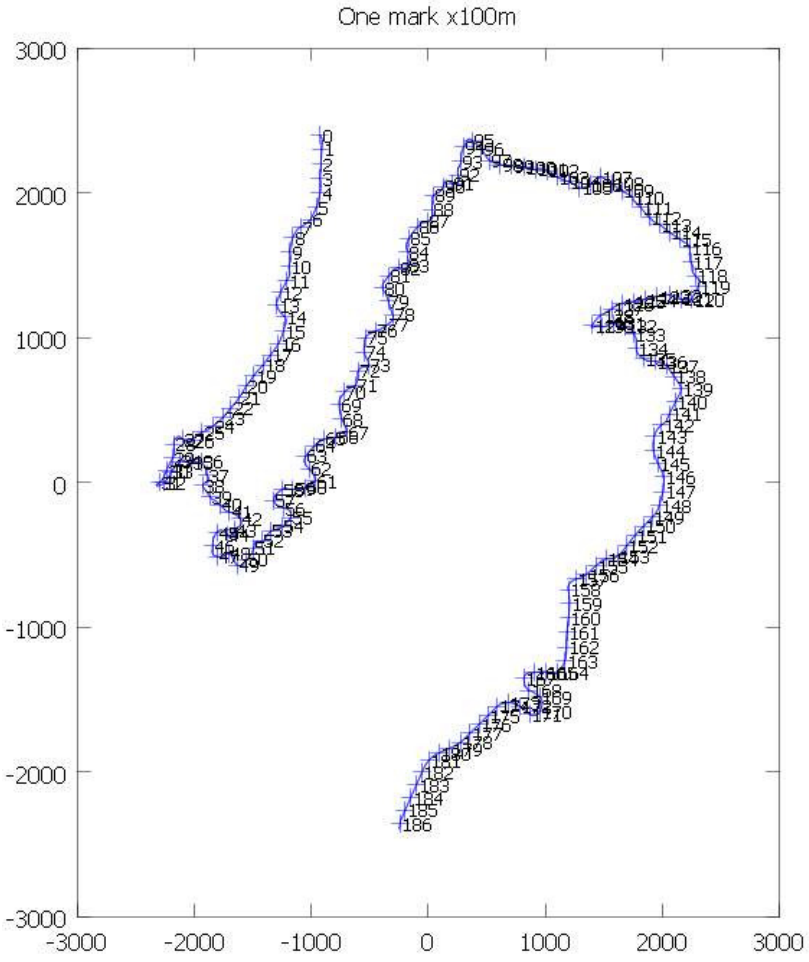


Later I will process the data inside s2_elevation_b, now I want to see the elevation profile of the route. I follow the messages on screen:

> gv
> btb(5,1)
> gs3
> coge
> ct1
> dar_altura(15,0.2,-0.2)

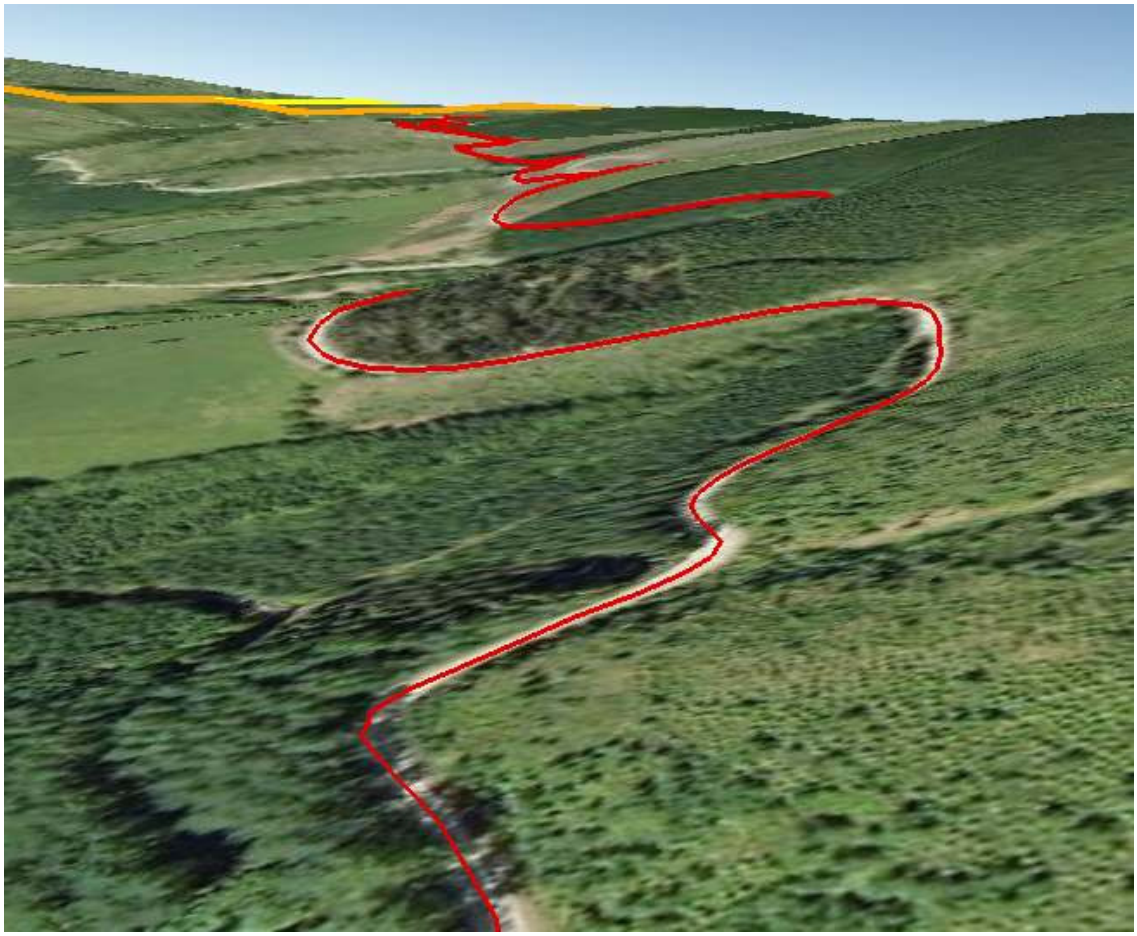
The first figure is the elevation profile (elevation vs distance) and the second figure tells me the distance each 100m on the route.


Mountain
method 1
Edited
450
400
350
300
250
200
0
5000
10000
15000
20000
.2386.2, 334.514


One mark x100m
3000
2000
1000
0
-1000
-2000
-3000
-3000
-2000
-1000
0
1000
2000
3000

The elevation profile is a disaster. Some of the peaks and notchs are real but a lot of them are due to bad elevation data.

For example, all the elevation changes between 5 and 10Km are unreal, as can be seen on Google Earth: bad elevation data makes the road go up when we turn to the right and go down when we turn to the left (or viceversa).



But for example, the peak on Km 12.9 is real because we were following a road upwards and then we took another road that goes downwards. There is a clear direction change.

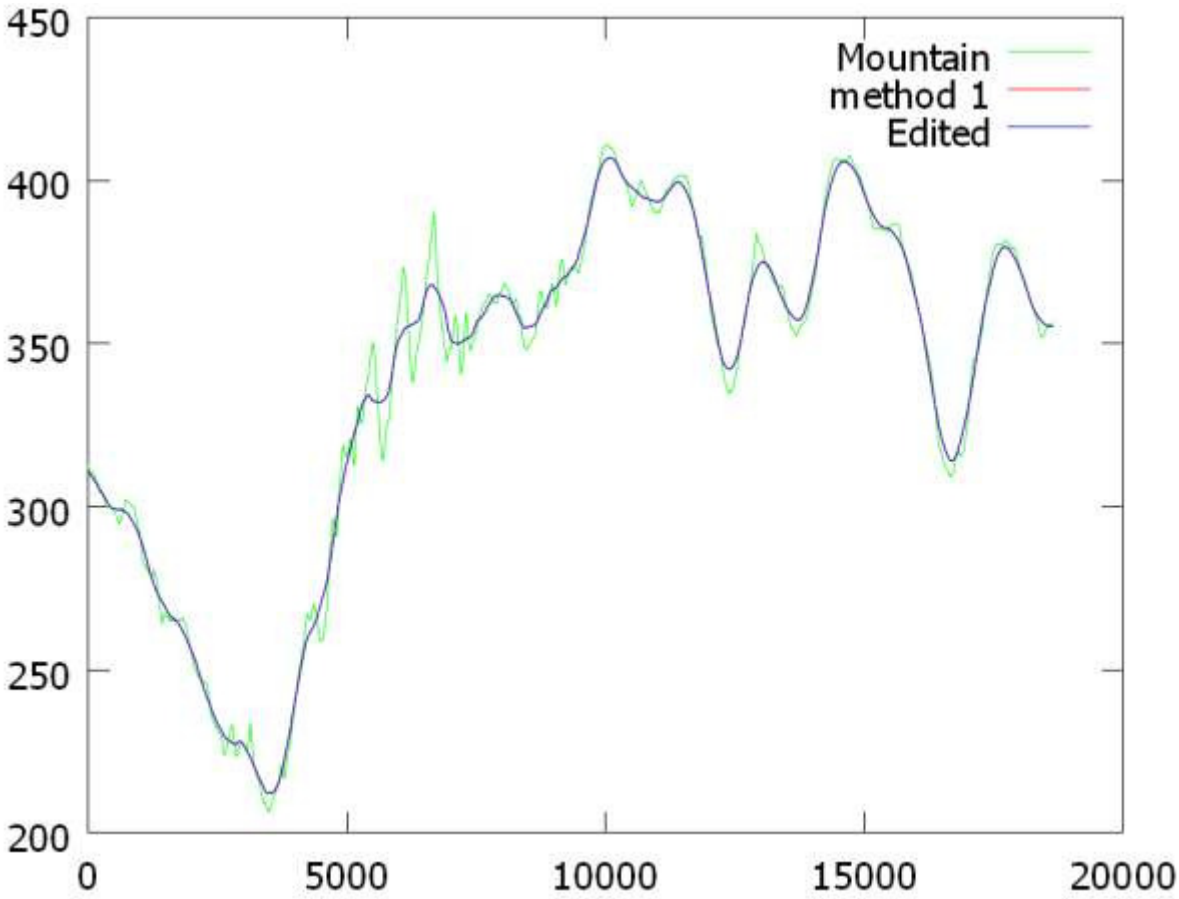
So, playing with dar_altura parameters will not work, as I can get a) a smooth road that doesn't fit bad elevation data, or b) a crazy/unreal road that does a good fit to bad elevation data, going up and down in a nonsense way.

Inside s3_road\salida I have a Venue.xml I can test, but I won't get good information from it.

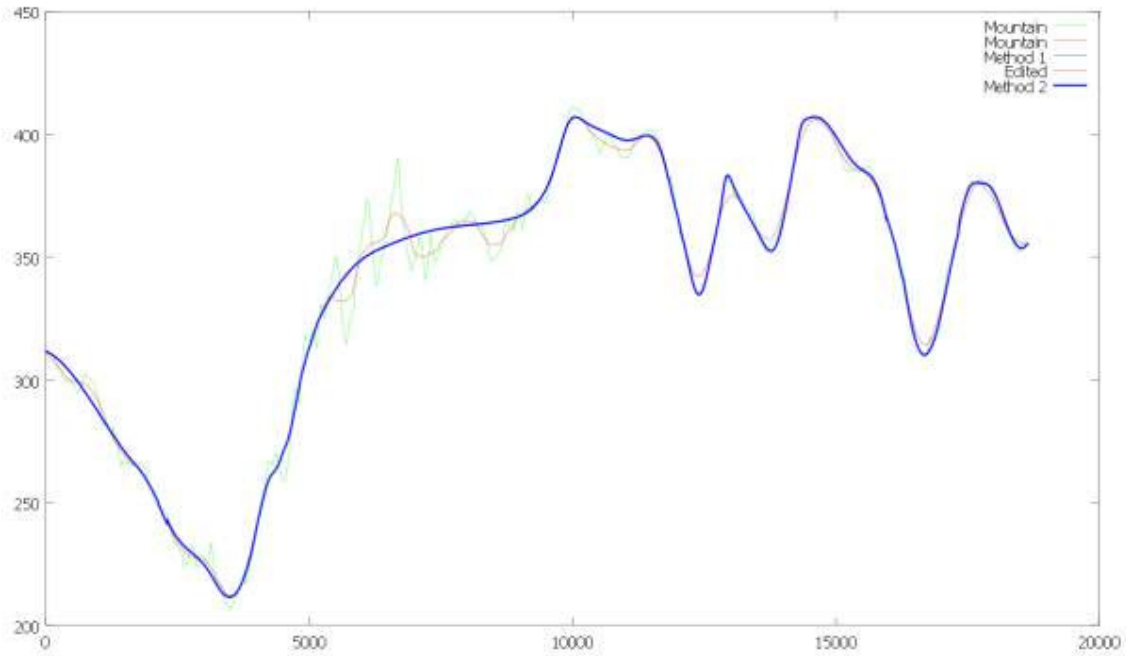
I need to do more steps before testing the road.

So, I start asking dar_altura for a smoother version of the road:

> dar_altura(103,1,-1,10)



The elevation profile is smoother, but it is not what I want, so I change it using the mouse and dar_altura capabilities:

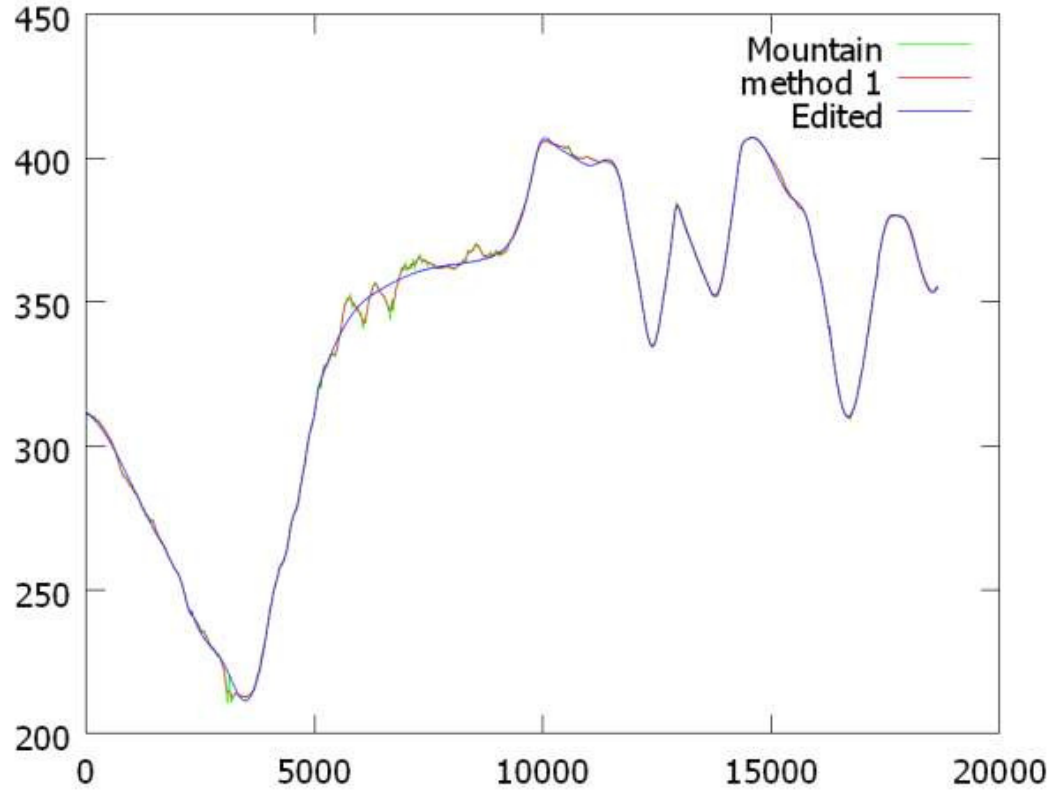


Ok, I accept this profile for the road. Now I want to change terrain elevation data to fit this road. I wouldn't do this step if the elevation profile created by dar_altura was good.

> **corregir**

And I run again

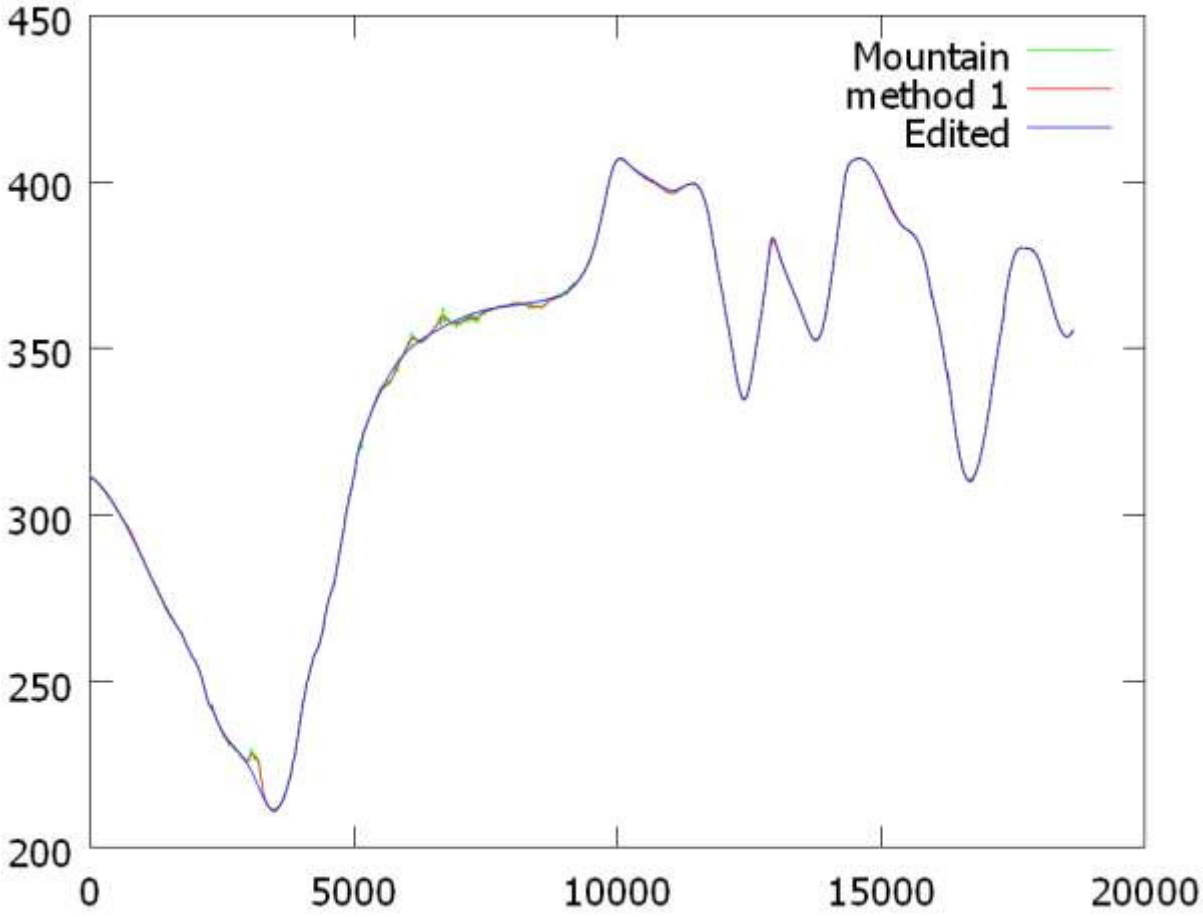
> dar_altura(19,0.25,-0.25)



The blue line is still conditioned by the changes I did to the profile using the mouse, so I **delete s3_road\retoques.txt**, but as we can see the elevation data is still bad, so I run corregir again

> **corregir**
> dar_altura(31,0.25,-0.25)

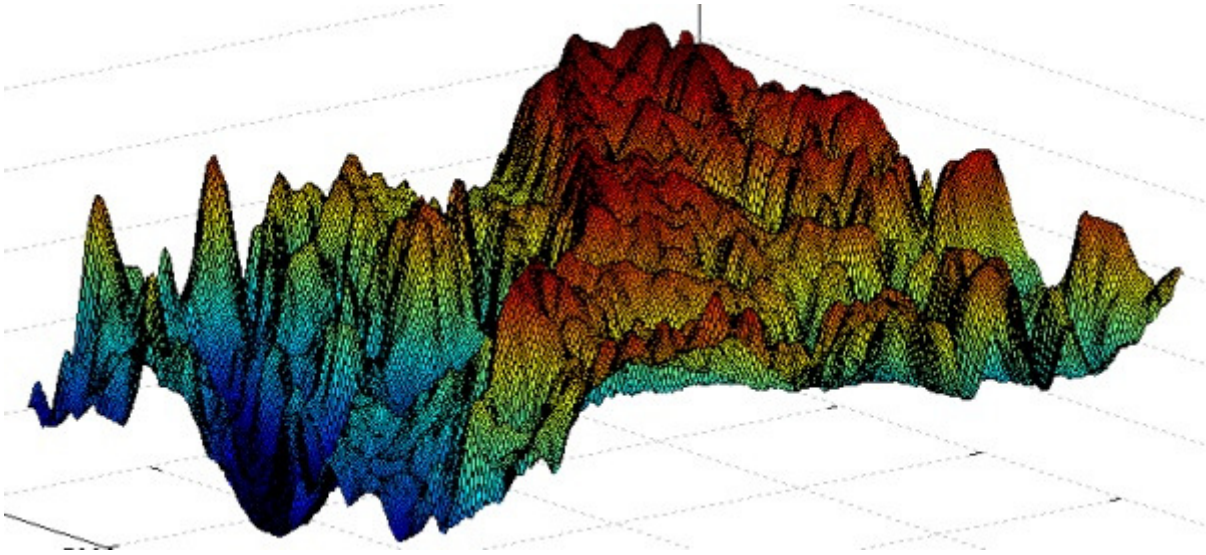
Now the result is better, so I just have to use dar_altura capabilities to change the profile a little bit in two zones of the graph. I accept the mountain on those zones will have to be fixed by hand using BTB, if needed.



So, now I can test s3_road\salida\Venue.xml

I am going to create a quick version of the terrain so I can check how road and terrain fit together (and if I like the track). In the meanwhile I have processed data inside s2_elevation_b (it must be processed before working on s4_terrain folder)

> cd ..\s2_elevation_b
> **read_grid**
> plot_lamalla

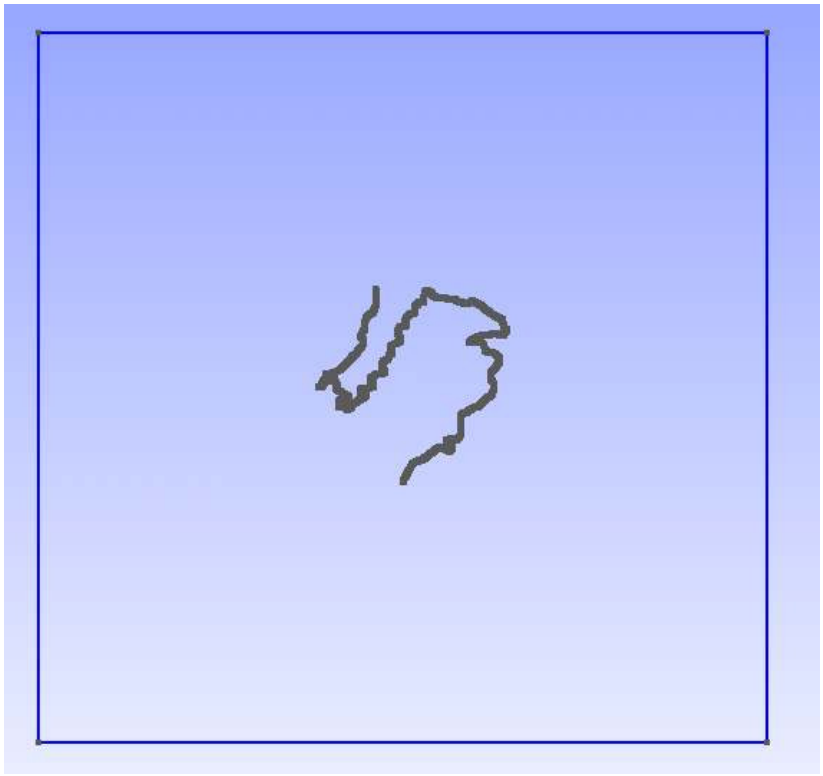


So I follow the suggested steps:

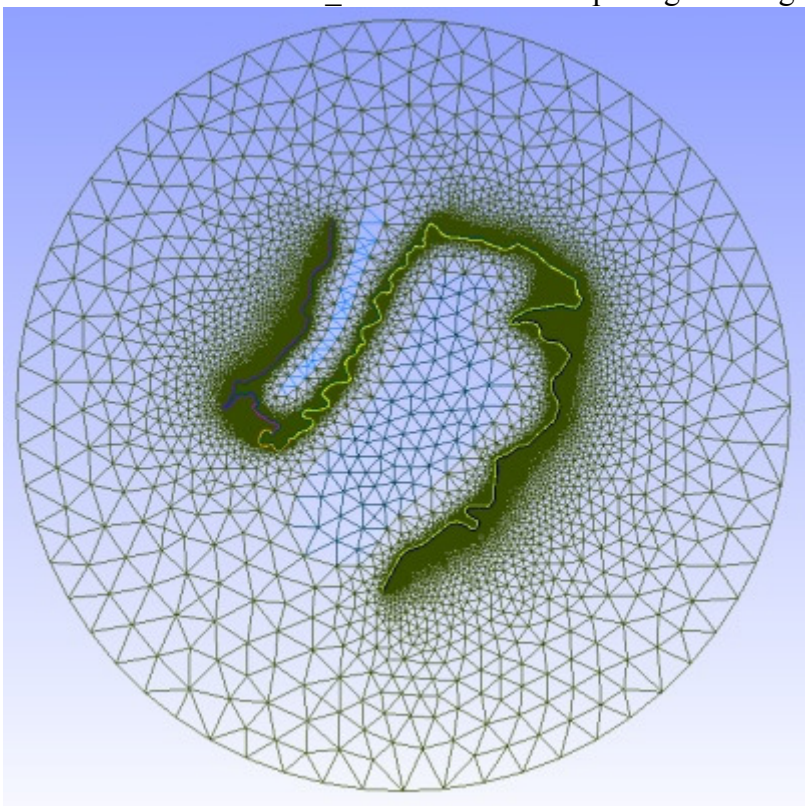
> gv
>**btb06(5)**
>gs1
>**mallado_regular(12,3)**

Then I use addgrid to view the limits of the available elevation data.
> **addgrid(1,1)**
(I select option 3, adding the limits to anchors_carretera.geo)

When I open anchors_carretera.geo with gmsh I see the track and the limits of available elevation data:



I use gmsh to create the Physical Surfaces for the driveable and non-driveable zones. I save the mesh as anchors_carretera.msh and opening it with gmsh I see this:



I am not pretending to create a final version, so I don't pay too much attention to the mesh.

Then I process anchors_carretera.msh
> **t_m**

I want to have satellite photos as background images so I run create_hlg:

> **create_hlg**

and I open s1_mesh\salida\grid.hlg with SASPlanet. I go on with the process while SASPlanet works.

> gs4
> coge
> **p_n**
> **simp**

I process the s4_terrain\salida*.ply files with MeshaLab and I join the result using juntar_mallas:

> **j_m**

SASPlanet has fnished its task and I save the images inside images folder. I make a 10x10 split and I make sure .dat files are created. Files are named sat_*.jpg

While juntar_mallas works I process the .jpg files, converting them to .dds (I use Irfanview to batch resize them as 2 power size, and DDSConvert to batch convert them to .dds).

I add the invisible walls:

```
> gs7
> coge
> p_m
```

I split the track into 10 segments and create the terrain in BTB format:

```
> gs10
> coge
> split_track(10)
> p_t
> procesar_elementstxt_mt(10,10,1)
```

BTB terrain will have also 10x10 zones and they will have a texture blending with the background images (the third parameter set to 1 forces the blending).

While procesar_elementstxt_mt works, I open another octave to create the list of images (list_bi.txt)

```
> cd c:\project\s1_mesh
> add_dat_to_geo('..\images')
```

I select option 1 (create grid.geo) because for this example I am not interested in changing .geo files to fit the images boundaries.

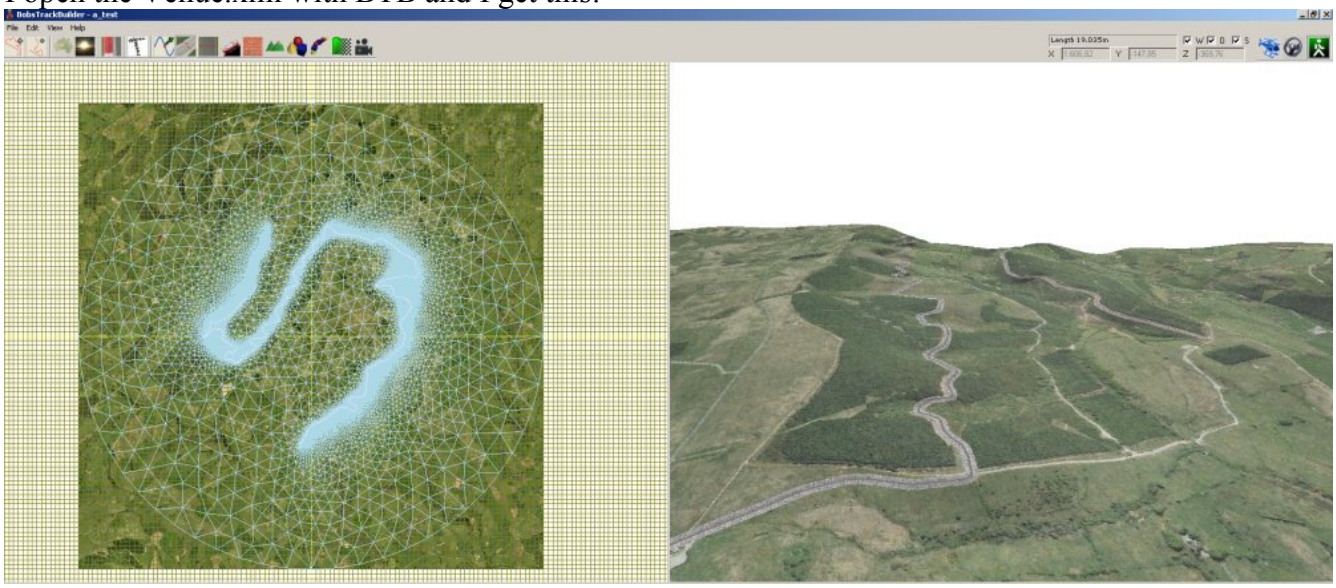
procesar_elementstxt_mt has finished. I create the pacenotes and I finish the process:

```
> cd ..\pacenotes
> coge
> pacenotes_a
> pacenotes2_a(0.01,10)

> gs9
> j_a
```

Now, before trying to open the Venue.xml I must insert s1_mesh\list_bi.txt in the Venue.xml and the images must be inside *My Project*\XPacks\Common\Textures

I open the Venue.xml with BTB and I get this:



The track used is **Halfway, one of the stages of the WRC Wales Rally GB**

Download all the files used with the scripts:
http://www.mediafire.com/file/ge8tayq2foii42b/halfway_example.7z
Download RXRSBR file to test the result (tarmac):
http://www.divshare.com/download/12178954-232

NOTE: running “start” is the same as running

> cd s0_import
> importakml(‘road.kml’)
> cd ..\s2_elevation
> make_grid(‘limits.kml’,25)
> cd ..\s2_elevation_b
> make_grid(‘limits_b.kml’,75)