TUTORIAL

LSCorridors Package

Authors

John Wesley Ribeiro & Milton Cezar Ribeiro

Organization

Bernardo B. S. Niebuhr

Juliana Silveira dos Santos

Felipe Martello

Pavel Dodonov

May 2016.

Summary

1. Introduction

The LSCorridors is a free and open source package developed in the Python library and Graphical User Interface (GUI) that simulates multiples functional ecological corridors. This tutorial aims to show the basic functions of the LSCorridors with the a demonstration data set. The use of this software and tutorial requires: a) the GRASS software installed; b) a folder with the LSCorridors files and c) a folder with a database to run the app (BD_demo).

Note: 1.) For this tutorial, we put the LSCorridors and BD_demo folders in the "C:\Users\Juliana\Documents\RUN_LS_Corridors" directory, and all instructions in this tutorial use this address as default. If you want to set these folders in directory, remember to change the path; 2) the LSCorridors and the BD_demo folders are compacted for download and have to be unzipped before use.

The LSCorridors and BD_demo files are available in the LSCorridors github (https://github.com/LEEClab/LS_CORRIDORS).

2. Installing GRASS Software

The LSCorridors was developed in the GRASS software - 7.0.x version. The installation files for this version are available at https://GRASS.osgeo.org/download/. This tutorial uses the 7.0.0 GRASS version, but our tests demonstrated that LSCorridors runs in all 7 GRASS version.

After download the GRASS software - version 7.0.0, execute the installation file "WinGRASS-7.0.0-1-Setup.exe". Press NEXT at the first window, select the installation folder and press NEXT until the software begins extracting files (Figure 1, 2, 3, 4, 5, 6, 7 and 8). After the extraction, press the NEXT and FINISH buttons respectively.

Figure 1 – Installing GRASS GIS.

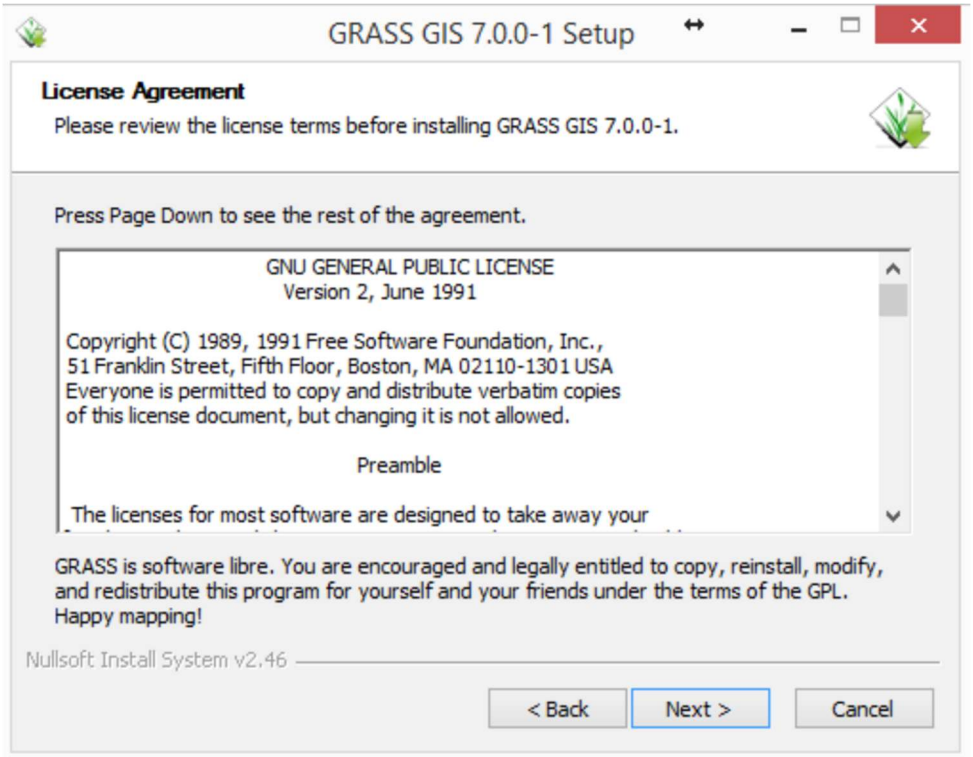


Figure 2 – Installing GRASS GIS.

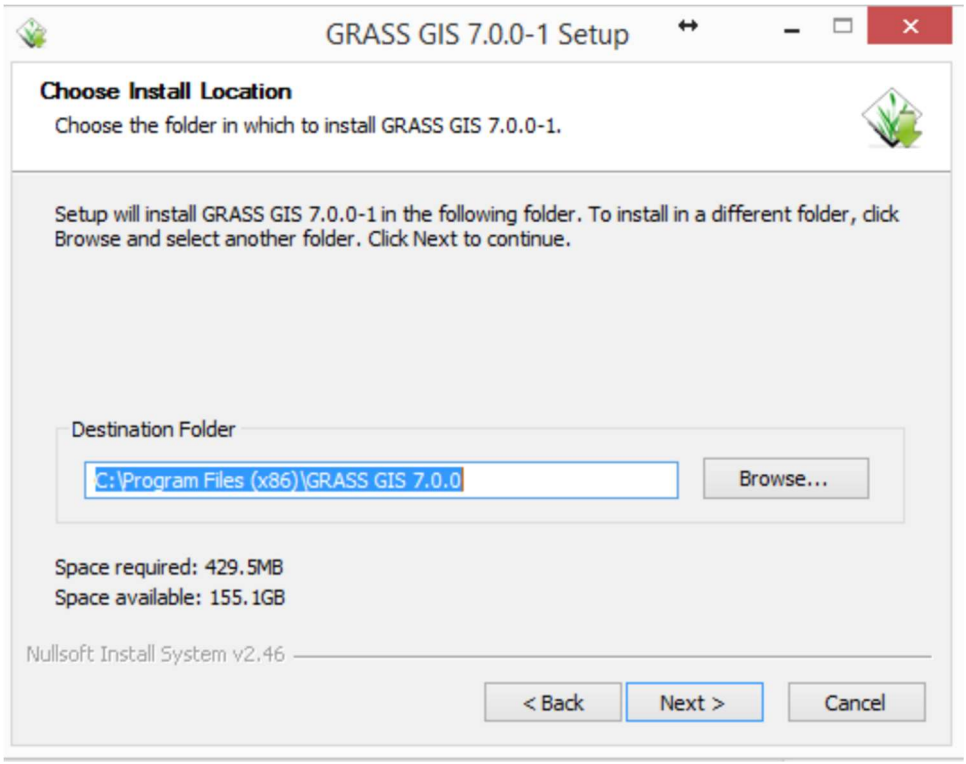


Figure 3 – Installing GRASS GIS.

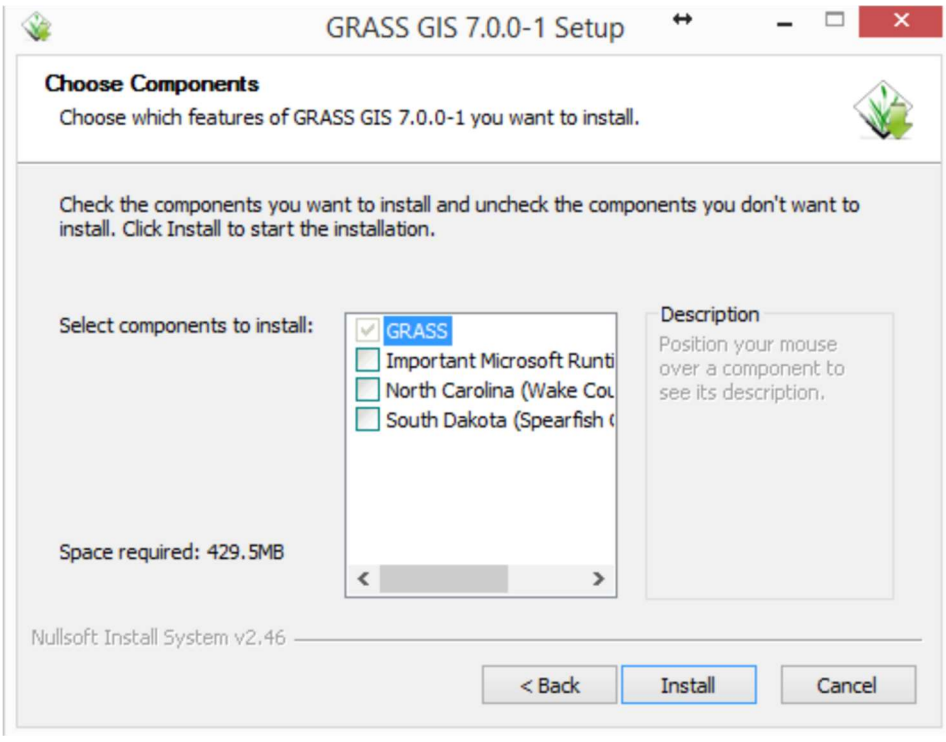


Figure 4 – Installing GRASS GIS.

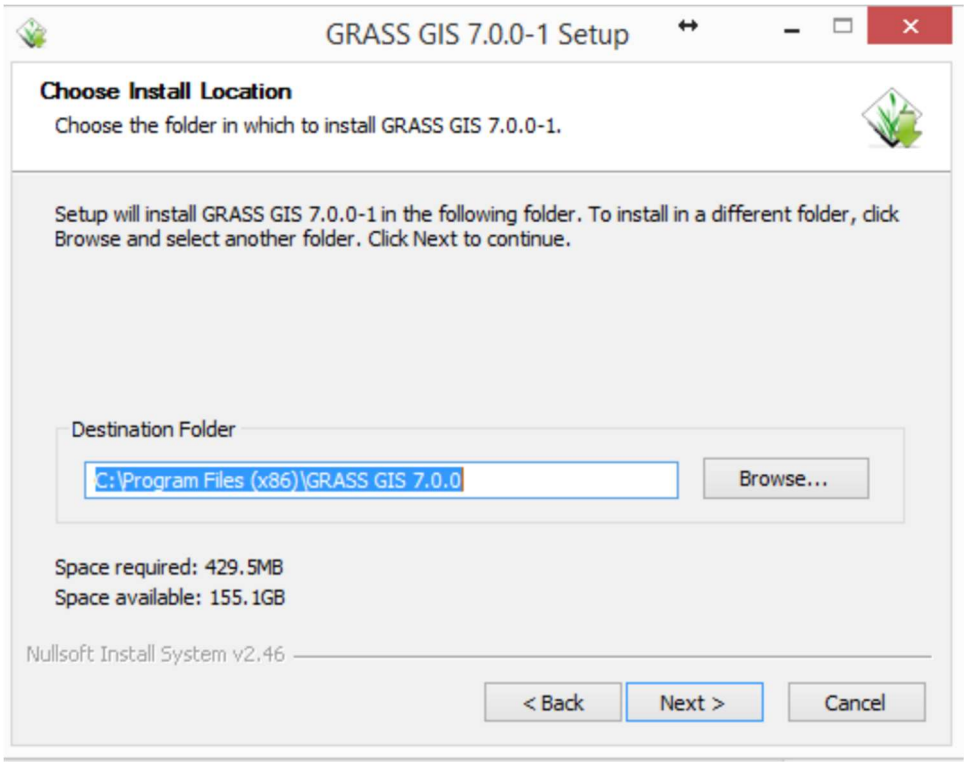


Figure 5 – Installing GRASS GIS.

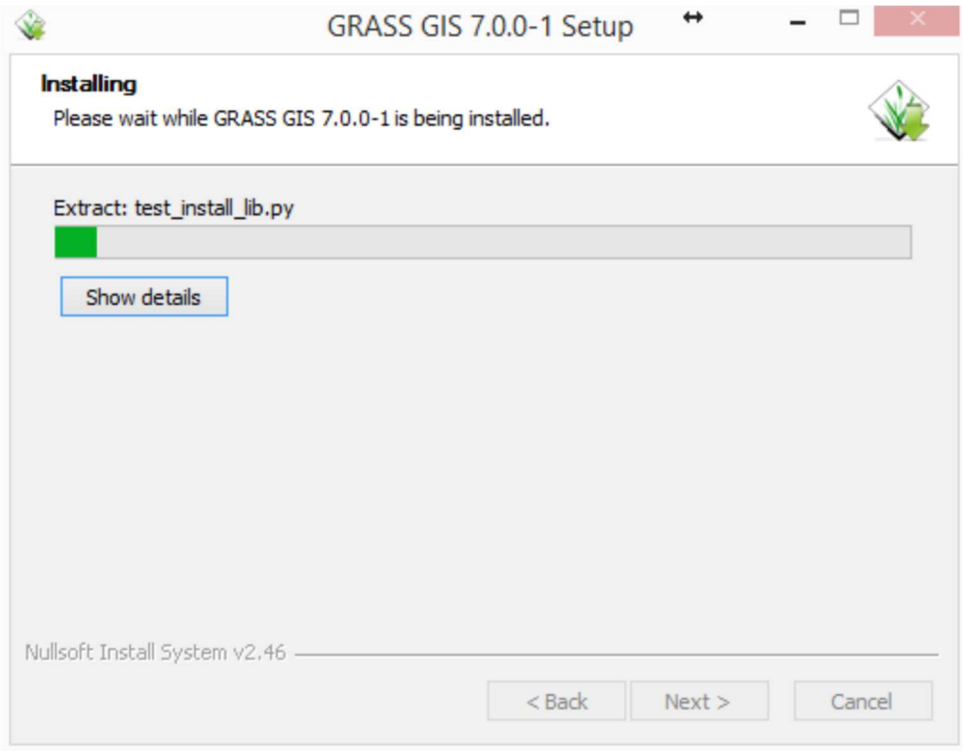


Figure 6 – Installing GRASS GIS.

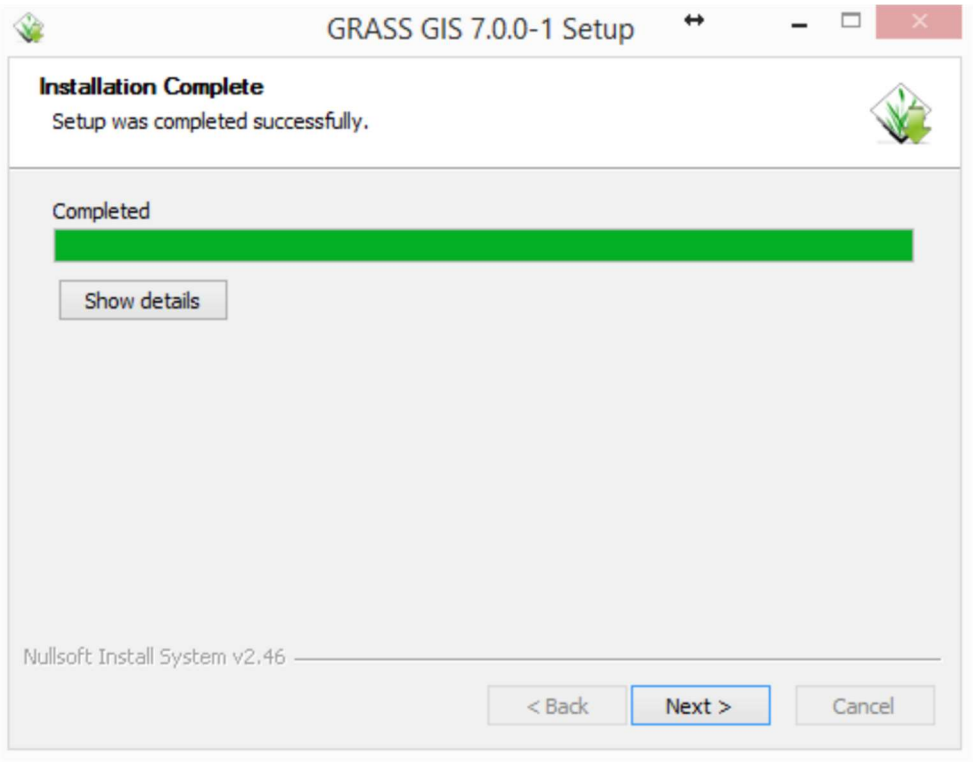


Figure 7– Installing GRASS GIS.

Figure 8 – Installing GRASS GIS.

To run GRASS GIS, perhaps will be necessary to install some ".dll" files and to do the upgrade of Python version. Please, check the GRASS warning messages.

3. Starting GRASS

Execute GRASS GIS 7.0.0 with GUI (Figure 9).

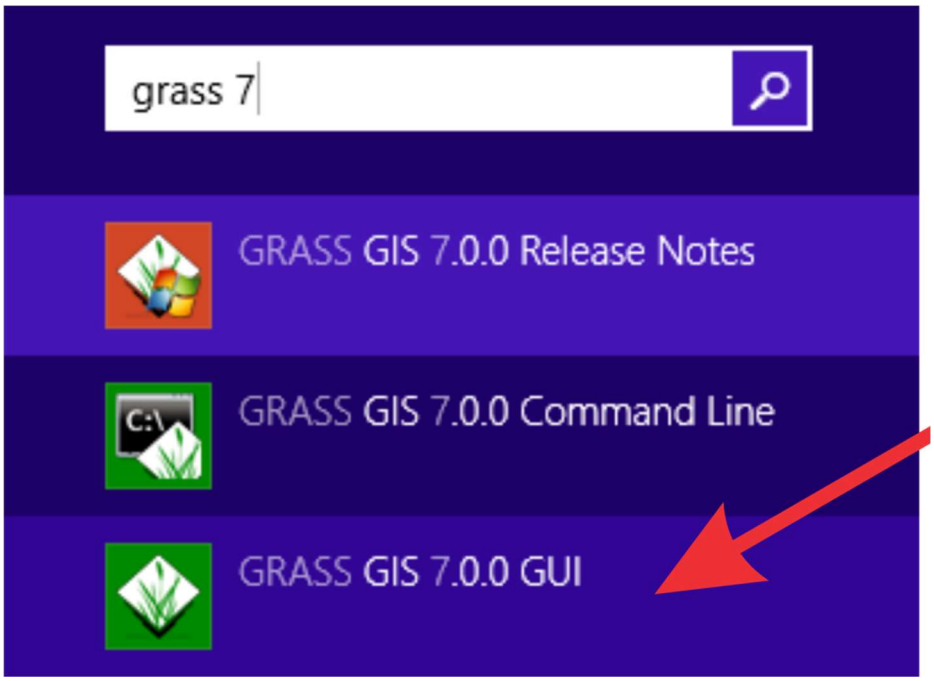


Figure 9 - GRASS icon.

A prompt command window will open, and you just have to press ENTER (Figure 10).

Figure 10 - Initializing GRASS.

A Welcome window will then open asking in which folder the data will be stored. In our example we will create a new database folder (newLocation folder). Click in Location wizard button (Figure 11).

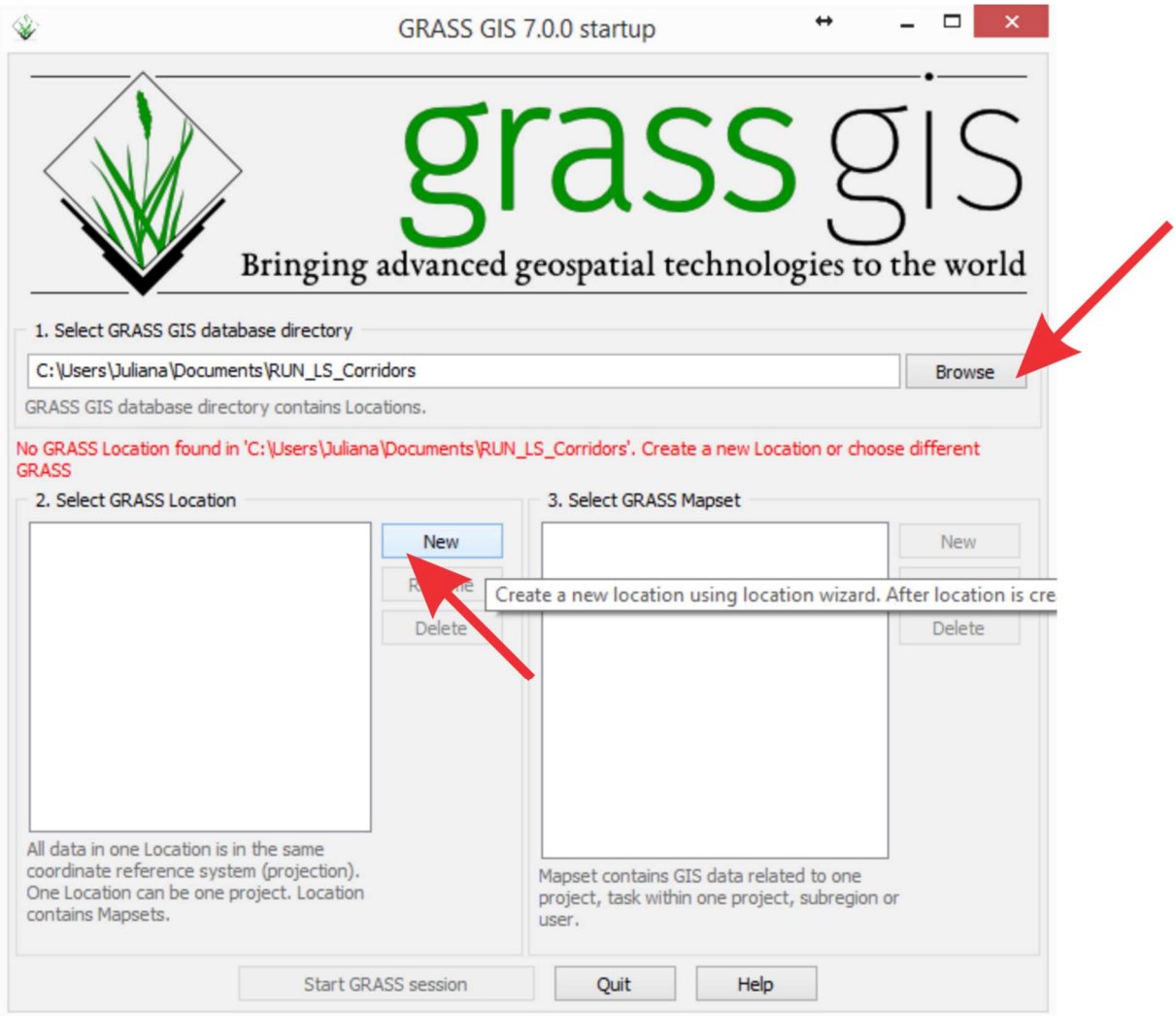


Figure 11 – Defining the directory to GRASS database.

On the Define new GRASS location select the location where the folder will be created, in our example "*C:\Users\Juliana\Documents\RUN_LS_Corridors*" and click the NEXT button (Figure 12).

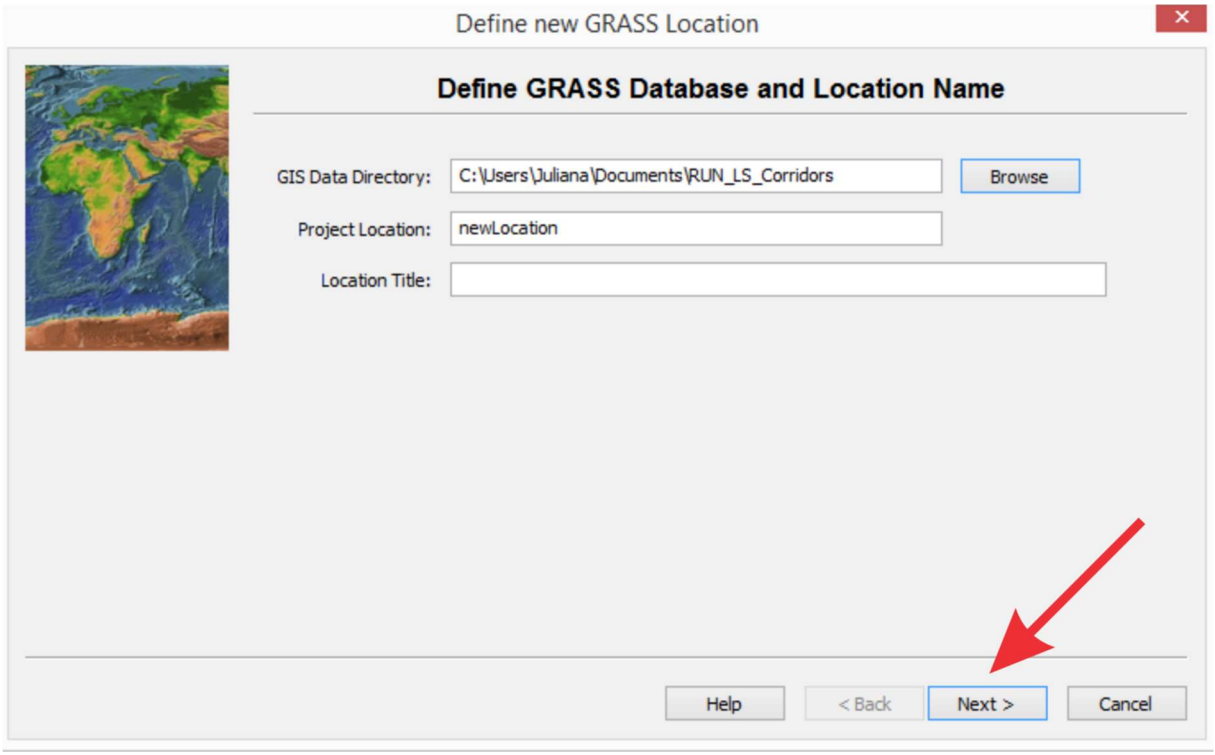


Figure 12 - Creating new location.

GRASS will open a window asking about the spatial reference system to be used for the newLocation files (Figure 13, 14, 15, 16, 17 and 18). Select the third option, Read projection and datum terms from a georeferenced data file, and click the NEXT button. Select the file “Resistance_map1.tif” as reference. Click the NEXT and then Conclude buttons.

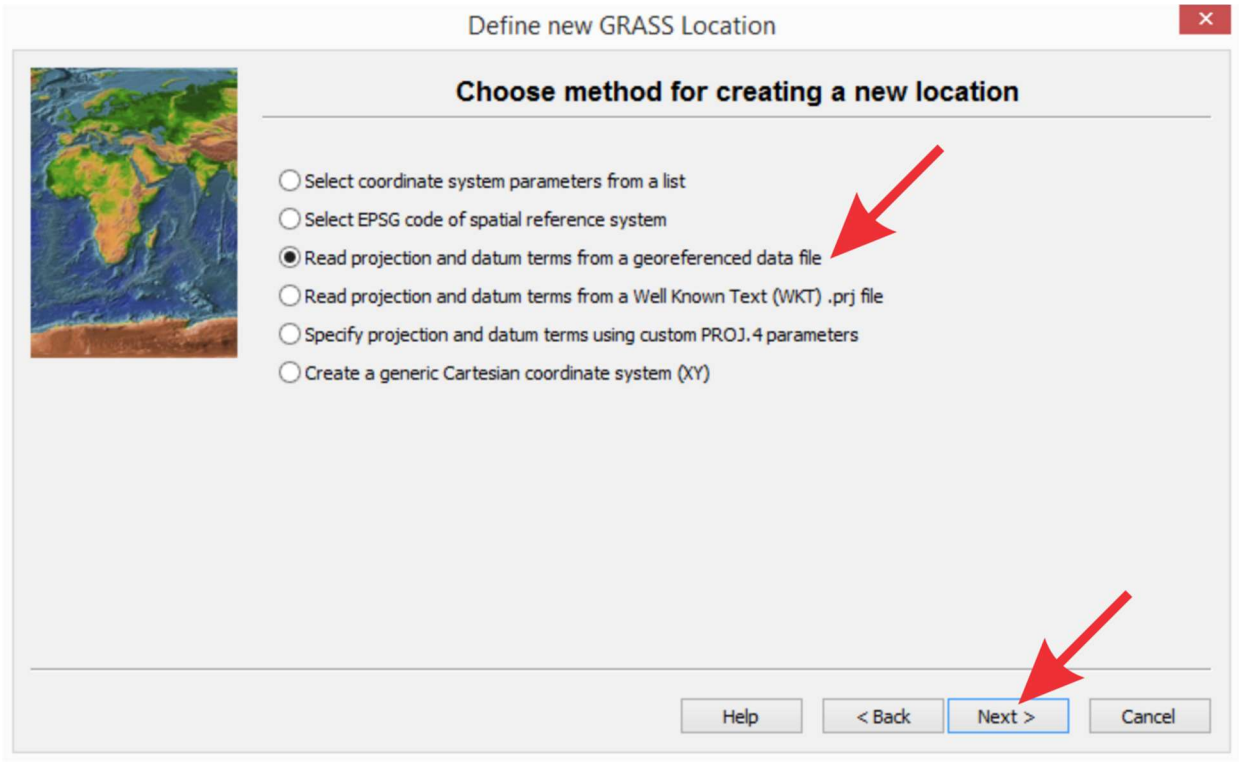


Figure 13 - Defining the new location projection system.

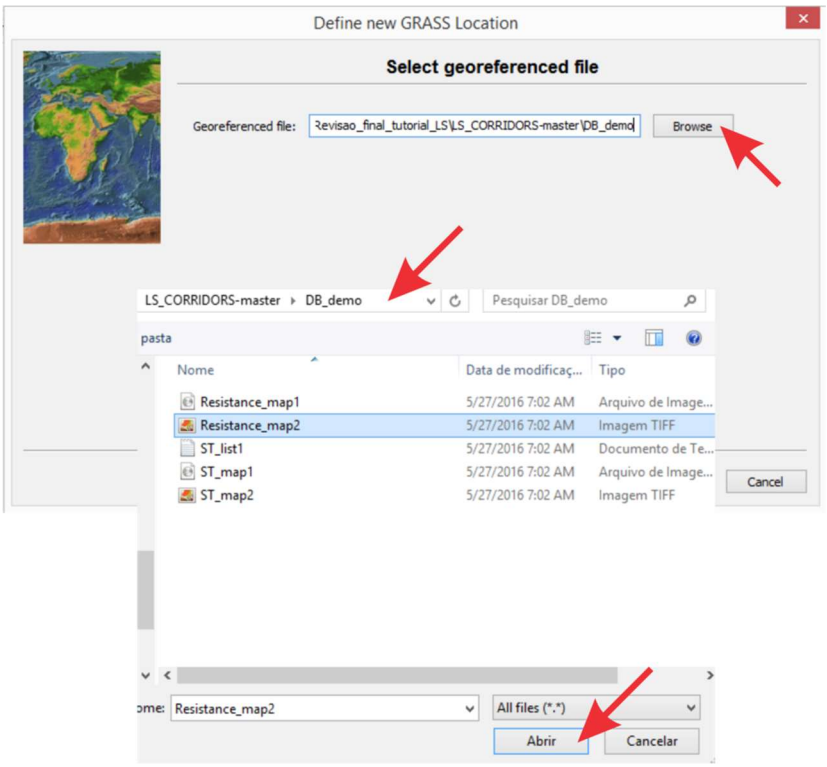


Figure 14 – Reading a file with the specific projection system.

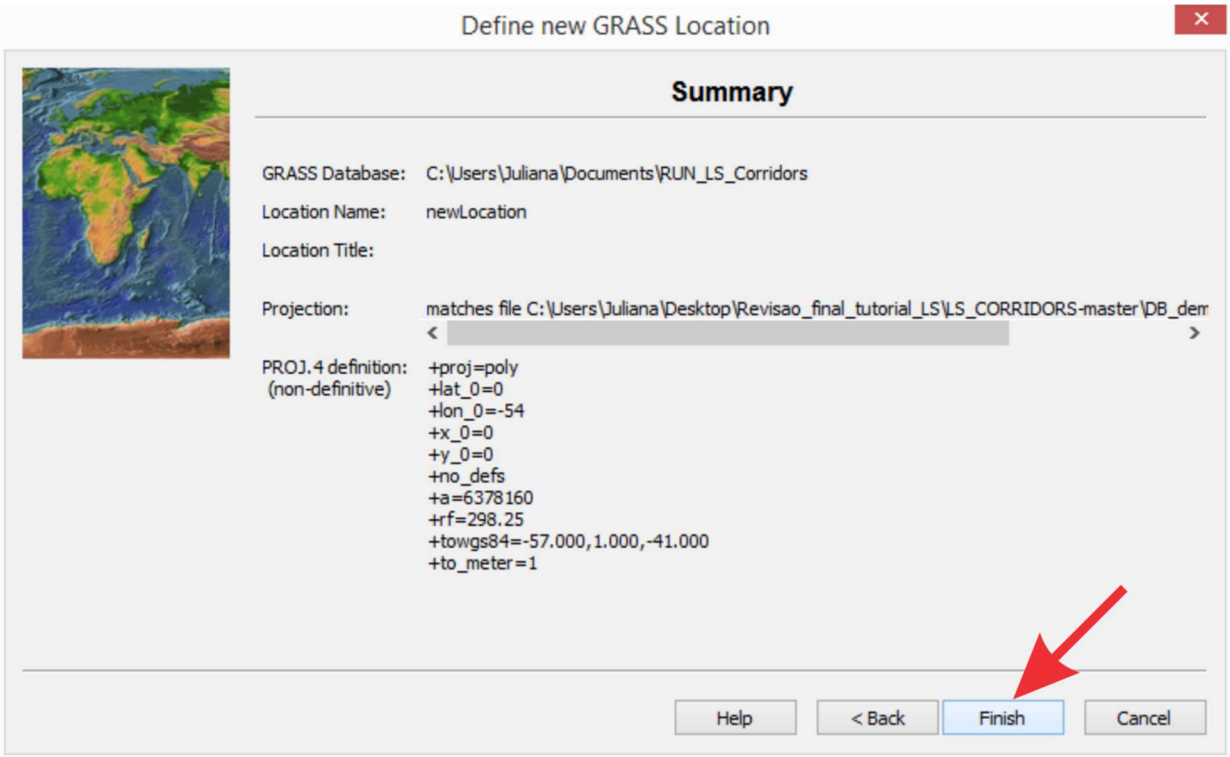


Figure 15 – Checking the projection system defined.

GRASS will open a window to alert you that a new folder will be created. You just need to click the OK button (Figure 16).

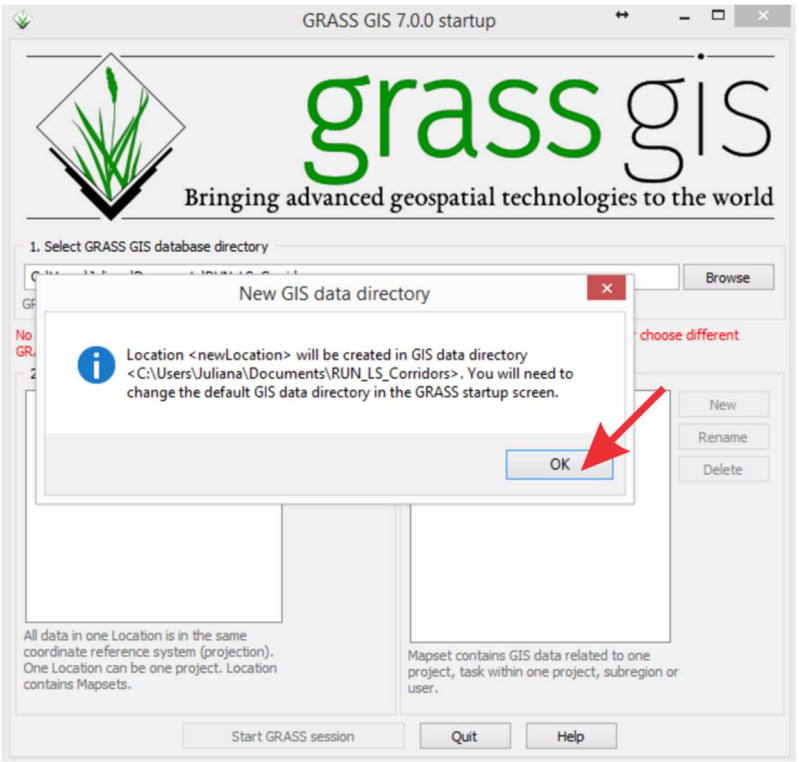


Figure 16 – Checking the new location.

GRASS will ask if you want to import the files now. We recommend to click **No** and import the files after (Figure 17).

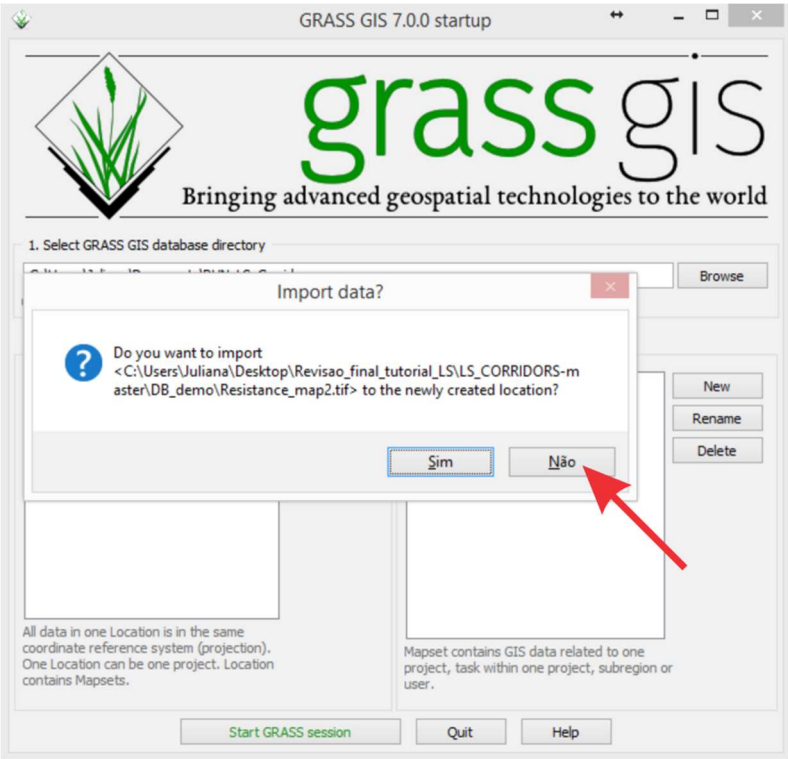


Figure 17 – Setting the configurations of the new location.

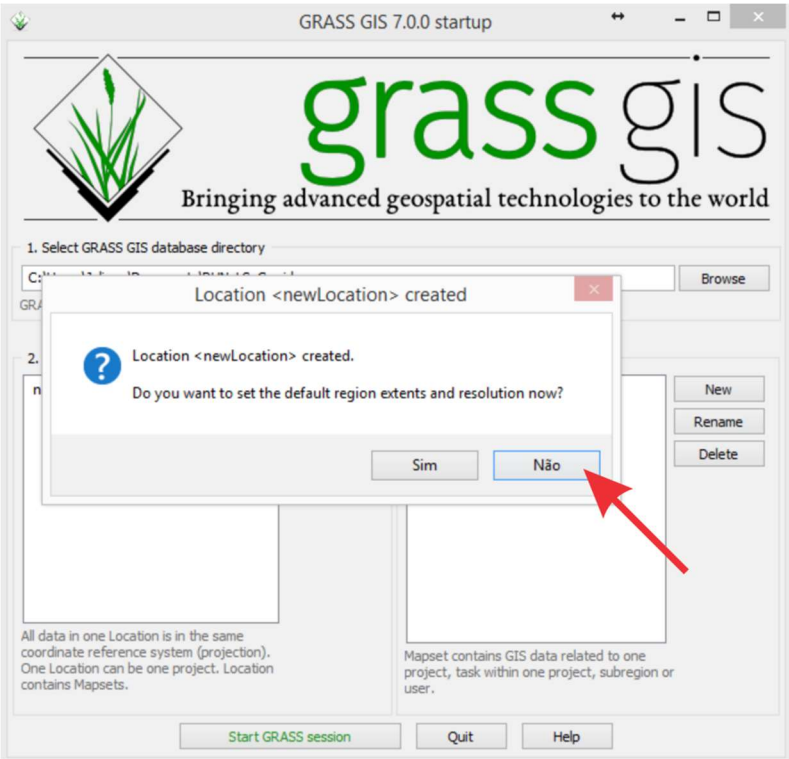


Figure 18 – Setting the configurations of the new location.

The next step is to create a new name for the new location.

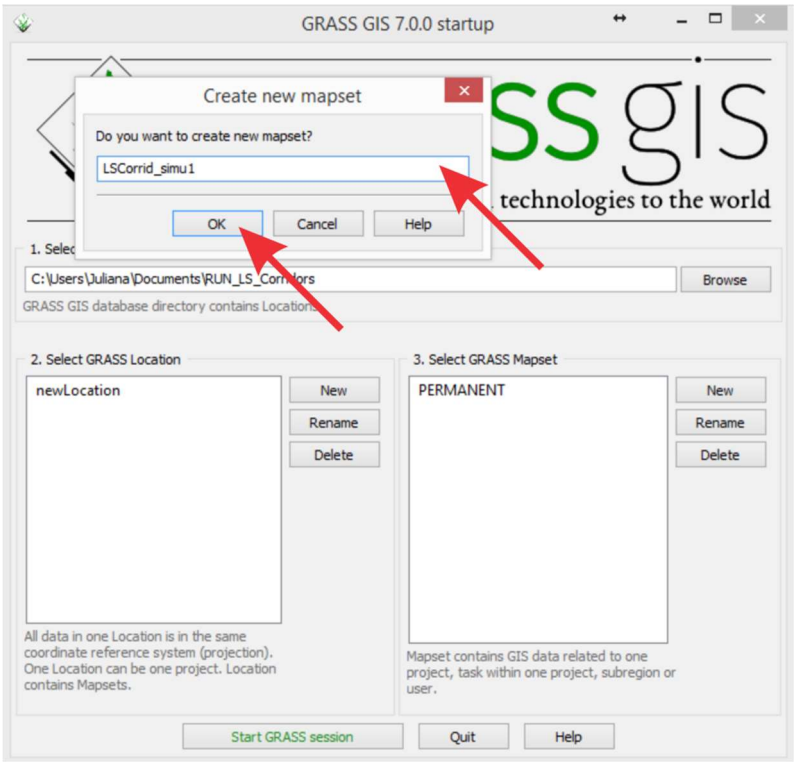


Figure 19 – Defining the project name.

Figure 20 – Starting GRASS GIS.

If everything works out, GRASS will open three windows: Layer manager, Map Display and sh (command prompt) (Figure 21).

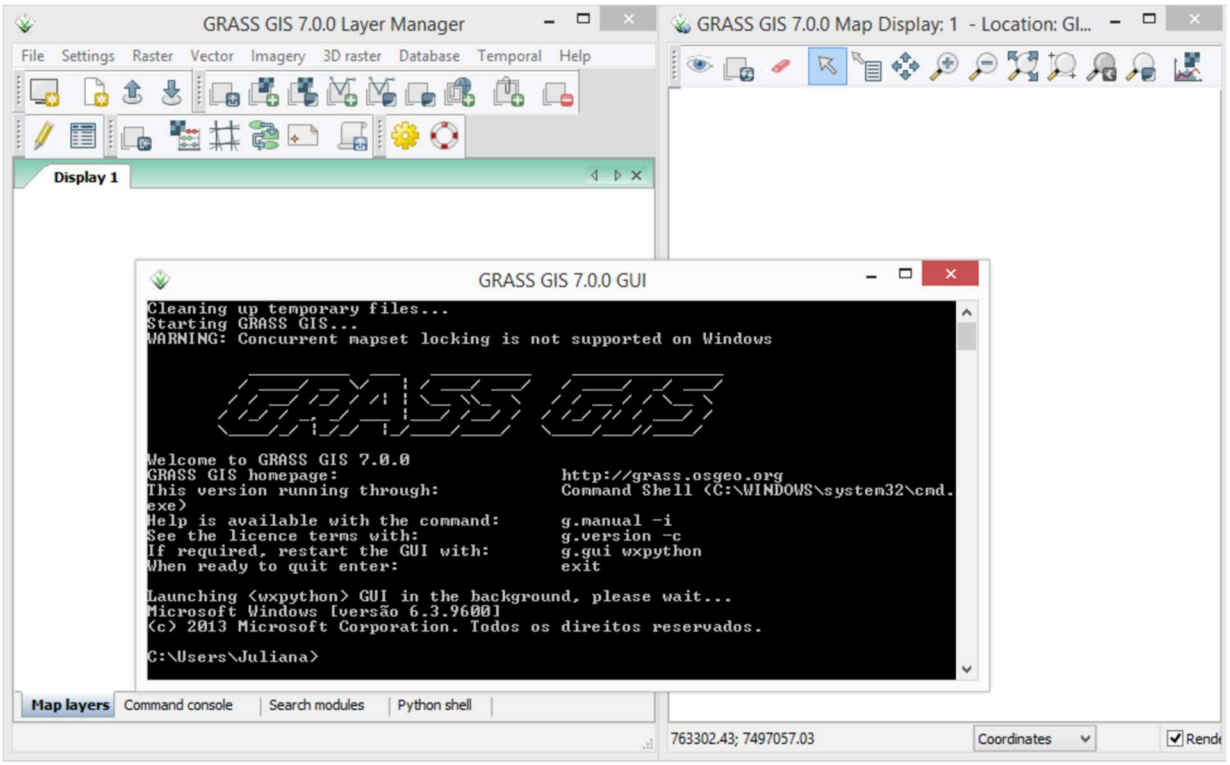
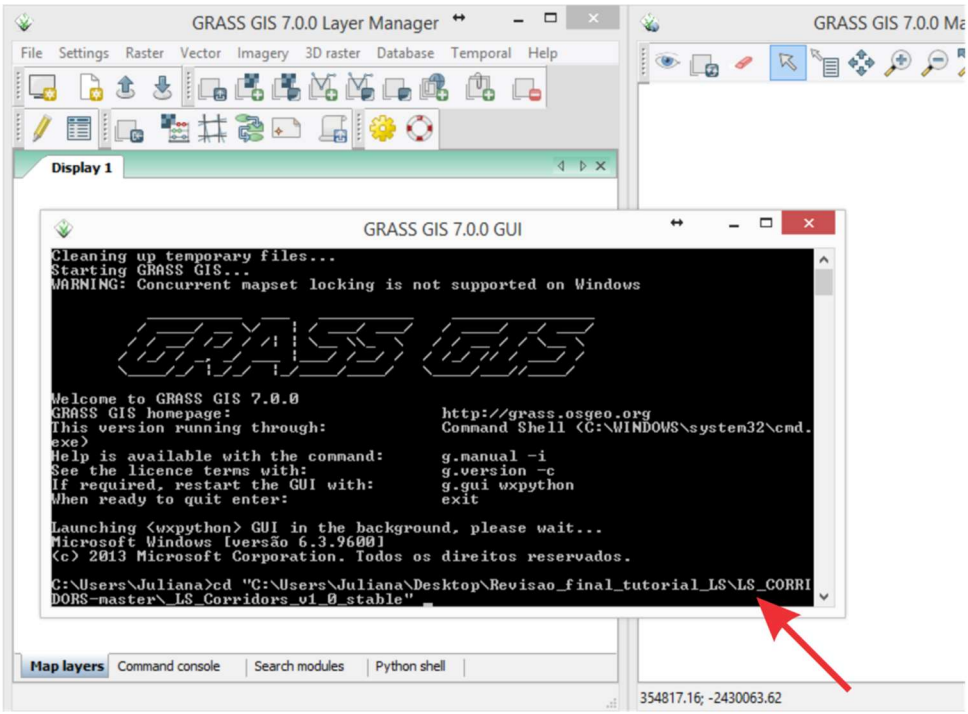


Figure 21 - GRASS layout with 3 windows.

4. Opening LSCorridors package

In the sh window (command prompt) set the LSCorridors folder by typing *cd "C:\Users\Juliana\Documents\RUN_LS_Corridors\LS_CORRIDORS-master_LS_Corridors_v1_0_stable"* and pressing ENTER (Figure 22).



cd "C:\Users\Juliana\Desktop\Revisao_final_tutorial_LS\LS_CORRIDORS-master_LS_Corridors_v1_0_stable"

Figure 22 - cmd_LSCorridors.

After that, type "*python LS*" and press TAB. The TAB key will complete the name of LSCorridors file (Figure 23). Finally, press ENTER.

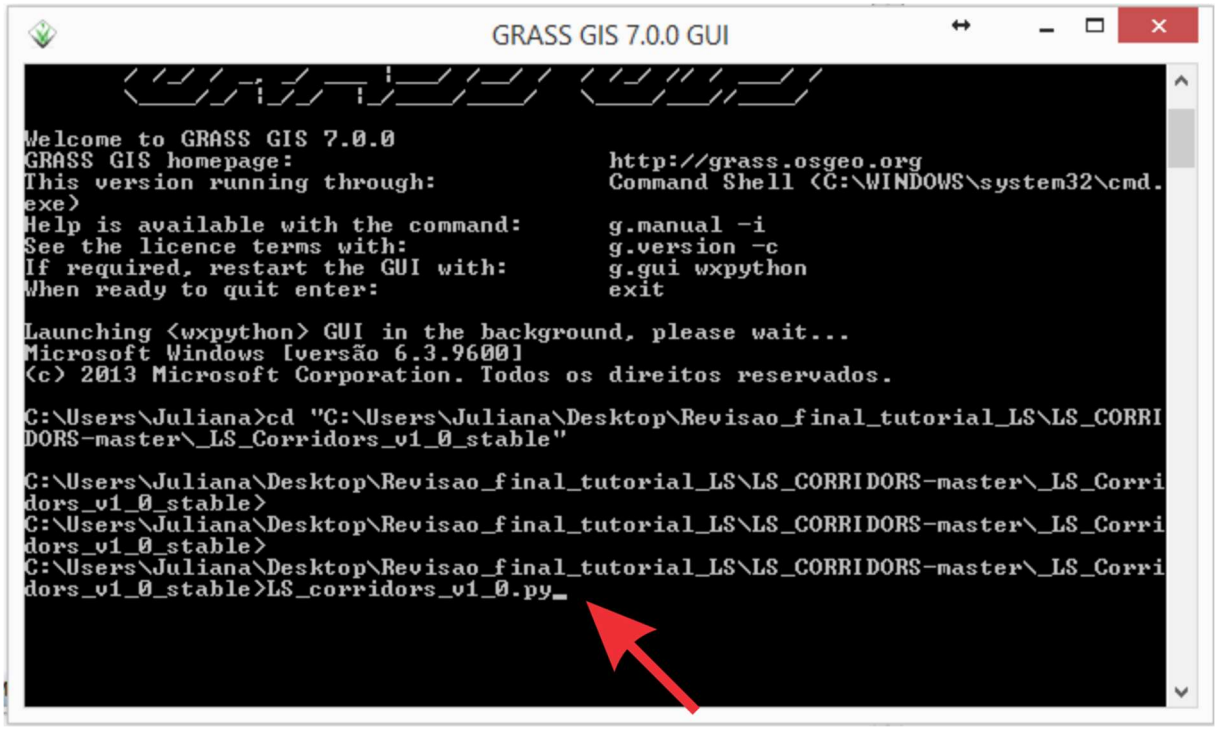


Figure 23 - Opening LSCorridors in GRASS GIS.

If everything works out the LSCorridors window will be opened (Figure 24).

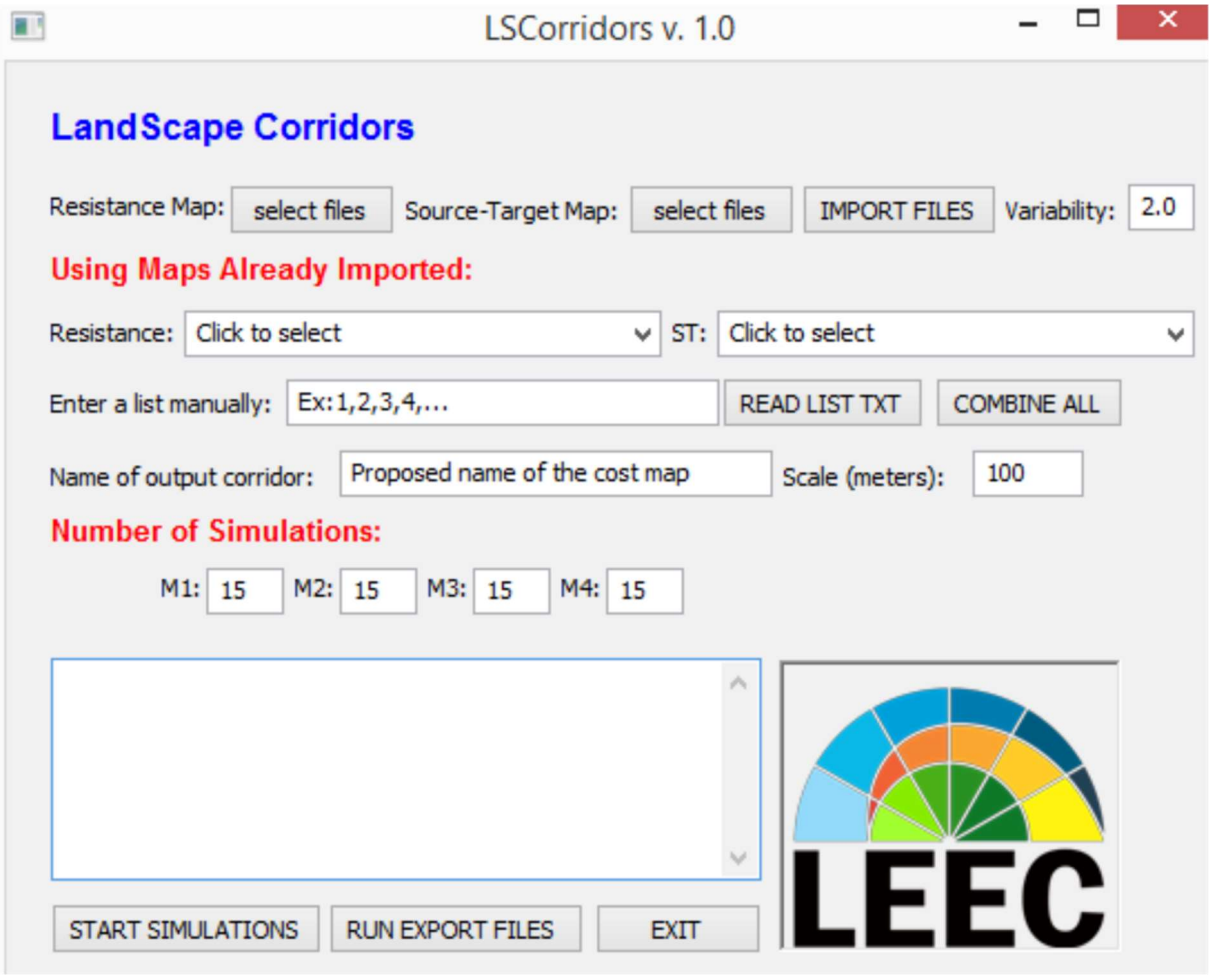


Figure 24 - LSCorridors layout.

5. Inserting the database and setting LSCorridors

The LSCorridors needs two raster maps as input files: a resistance map and a map of patches to be connected (Source-Target map). In this second map, pixels corresponding to each patch must be identified with a number unique to this patch (so the pixels within each patch will have the same number and pixels in different patches will have different numbers). Pixels not belonging to any patch are assigned the NULL value. Although not necessary, another file that maybe useful (and we will use it on our example) is a list with the identification numbers (ID) of the patches that you want to connect, i.e. between which the corridors will be simulated. The three demonstration files that we will use in this tutorial are in the BD_demo folder.

5.1 Inserting the resistance surface map

There are two ways of inserting the cost map: 1) selecting it in a specific folder in "Resistance Map" button, or 2) if you had already imported the map to the GRASS database you can select it in "Resistance". Here, we will use the first option. Click in the "select files" button beside the "Resistance Map" and inside the folder "BD_demo" select the file "Resistance_map1.tif" (Figure 25).

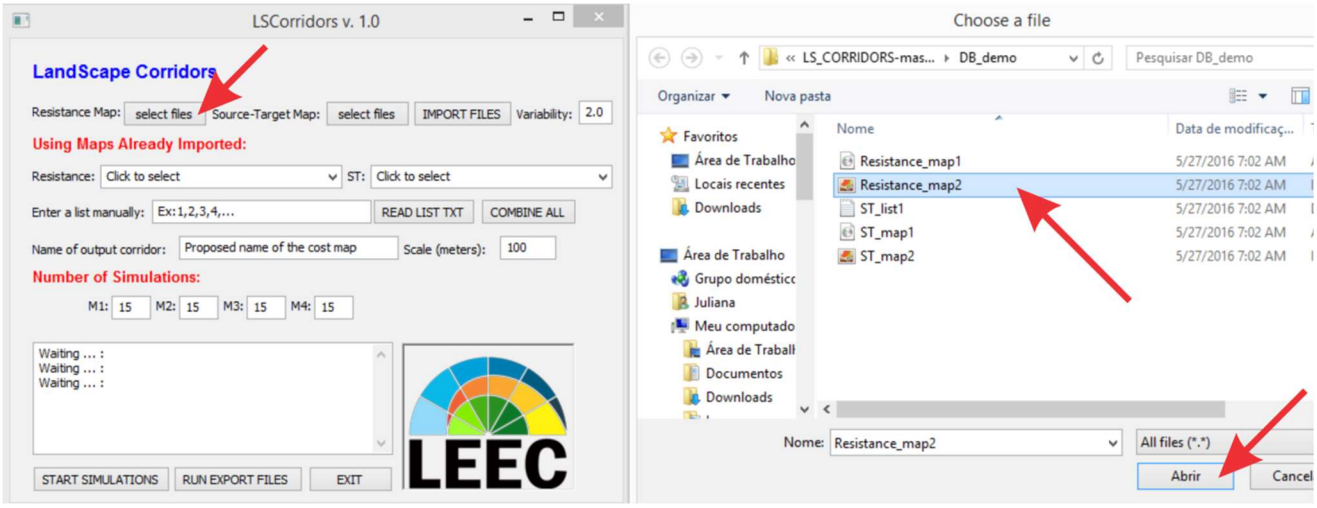


Figure 25 – Setting resistance map.

5.2 Inserting the habitat patches map

Similar to the resistance map, there are two ways of inserting the Source-Target map: 1) selecting a file in a specific folder with the button "Source-Target Map", or 2) or if you had already imported the map to the GRASS database you can select it in "ST" (Figure 26).

In our example we will use the first option. Click the "select files" button beside the "Source-Target map", and select the "ST_map1.tif" in "C:\BD_demo" folder. The selected map will also appear in the LSCorridors display.

After selecting both maps, click the IMPORT button: the LSCorridors display will show the message "importing rasters…" (Figure 27).

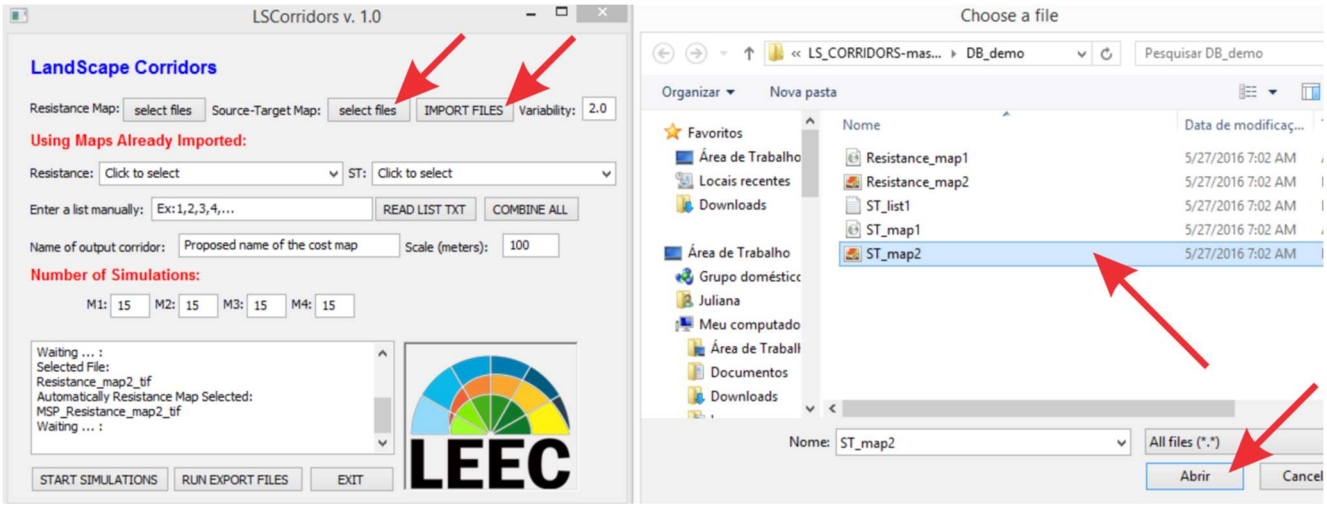


Figure 26 – Setting Source- target map and the importing files.

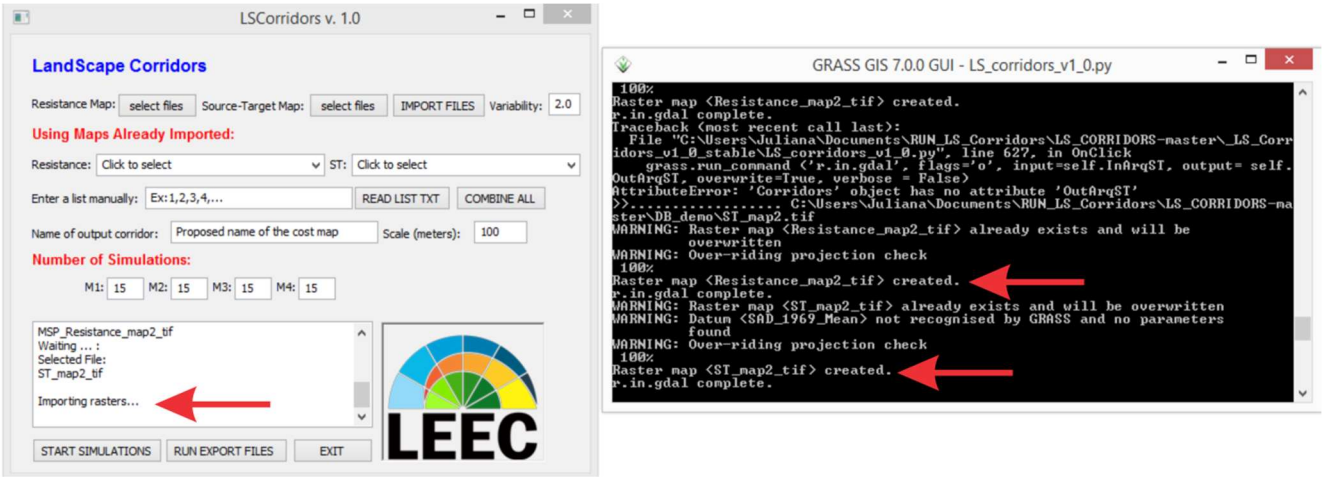


Figure 27 – Checking if the maps were imported.

5.3 Listing Source-Target patches

There also two ways to insert the list of Source-Target patches that will be connected (ST): 1) manually, by inserting the ID of the patches in the "Enter a list manually" command, or 2) by inserting a .txt file with the list of the patch IDs in the READ LIST TXT button (Figure 28). In both cases, the patches to be connected must be inserted as pairs, with commas separating the patches within each pair and the different pairs. For example, "1,2,1,3,1,4" means that patch 1 will be connected to patch 2, patch 1 will be connected to patch 3 and patch 1 will be connected to patch 4, whereas "1,2,1,4,2,3" means that patch 1 will be connected to patch 2, patch 1 will be connected to patch 4 and patch 2 will be connected to patch 3.

Here we will use the second option: click the READ LIST TXT button and select the "st_list.txt" file in the "C:\BD_demo" folder (Figure 28).

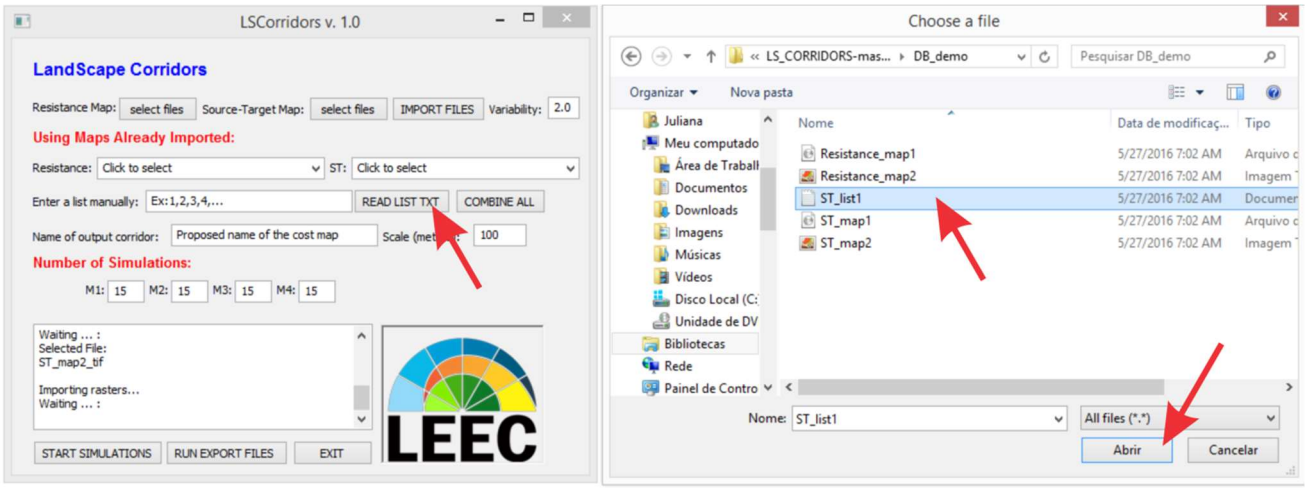


Figure 28 – Reading txt file with the Source-Target combinations.

If you want to generate corridors among all the fragments listed, click the COMBINE ALL button, but note that if your ST list is too extensive it may cause an excessive delay in the processing. The LSCorridors display will show the message TXT Combinations with the patch IDs (Figure 29 and 30).

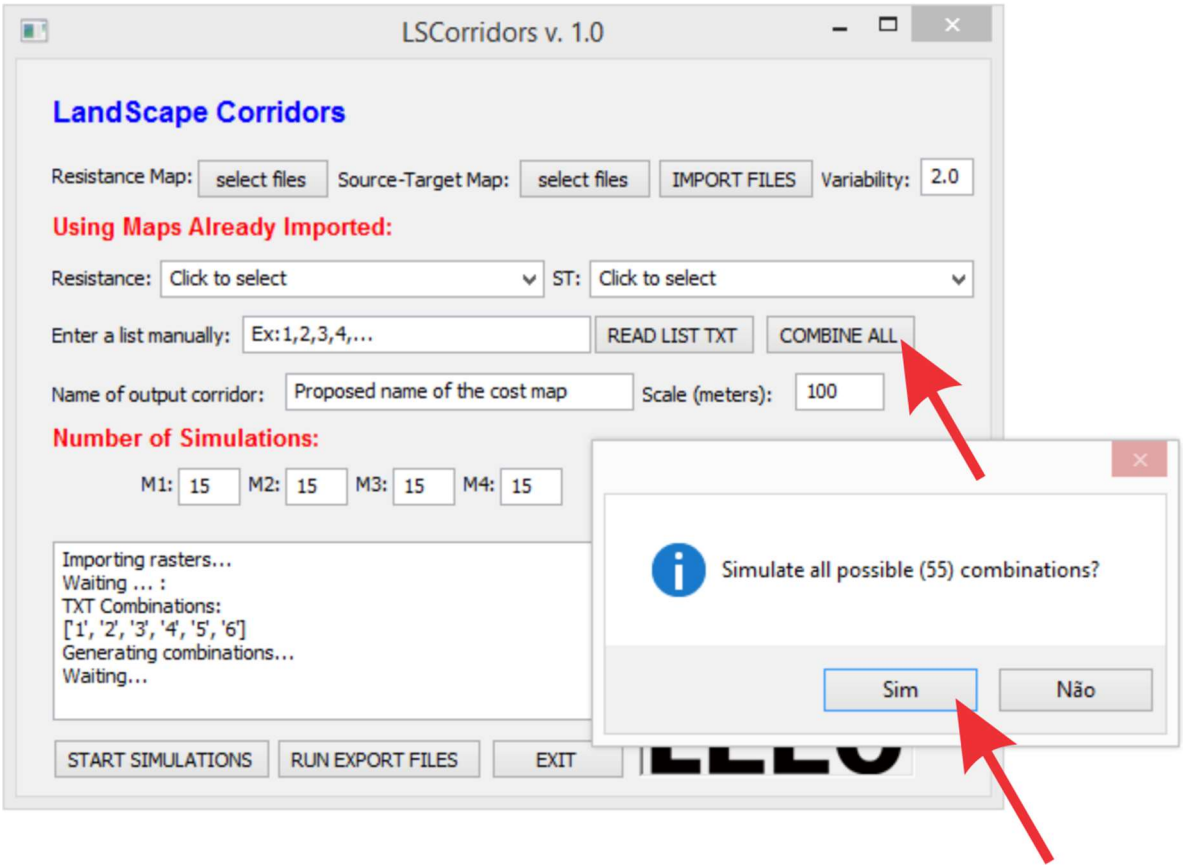


Figure 29 – Defining the number of simulations.

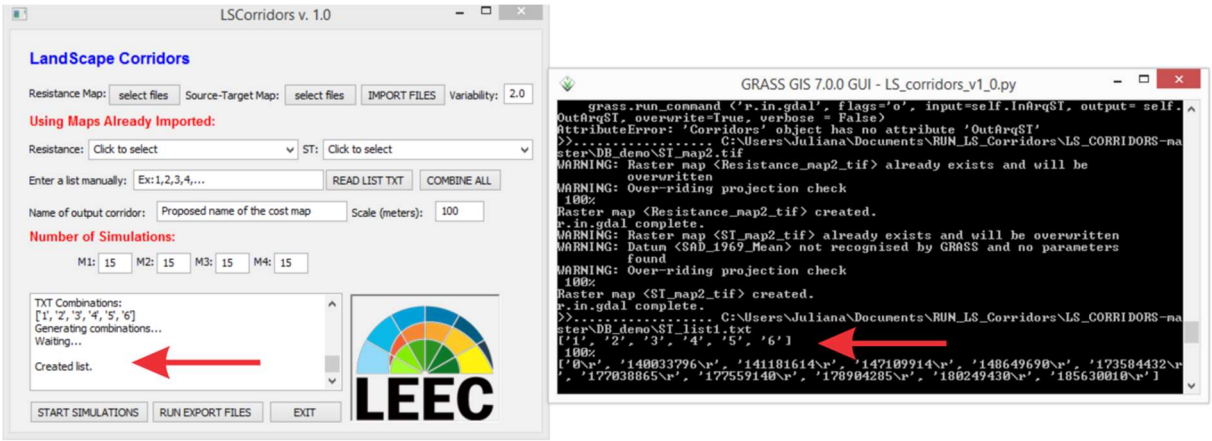


Figure 30 – Checking if the list of combinations was created.

5.4 LSCorridors parameters

There is a set of parameters that you must configure in LSCorridors before starting the simulations:

a) Variability: use this command to choose the spatial variability among corridor replicates. The default is 2, which we will use in our example.

b) Scale (meters): use this command to define the scale of species movement capacity. This parameter is used in the simulation methods M2, M3 and M4. The default is 100, we will use this value in our example.

c) Number of simulations: commands “M1”, “M2”, “M3” and “M4” represent each type of simulation method. The differences between the methods are in the way each pixel of the resistance map is used. Whereas M1 used the resistance value of each pixel, the methods M2, M3 and M4 use, respectively, the mode, maximum, and average resistance values within a window centered on each pixel, with the size of the window being defined by the scale parameter above. The value that we put on it refers to the numbers of replicates that LSCorridors will simulate for each Source-Target pair, in each method. The default is 15 in each method, and we again will use these values in our example, meaning that for each Source-Target pair we will simulate 60 corridors (15 by each method).

d) Name output corridor: you can name the output files inserting an export name here. This name will be used in all the output files (see session 7). If you don’t use this option the package will name the export files with the resistance map name.

6. Starting the simulations and setting the output folder

After setting the parameters you can finally click the START SIMULATION button. A window will open for you to set the output folder: select “C:\Users\Juliana\Documents\RUN_LS_Corridors\results” (Figure 31 and 32).

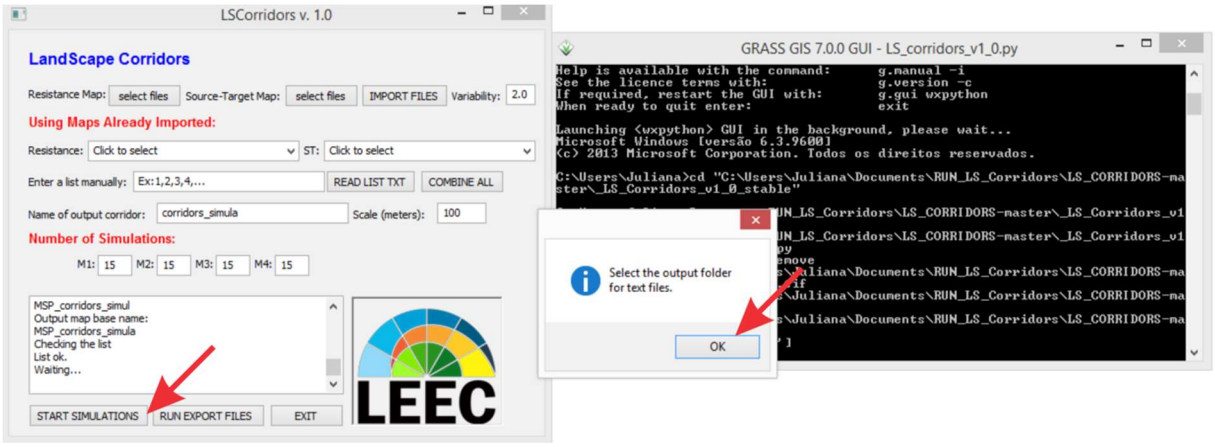


Figure 31 – Defining the output folder to results.

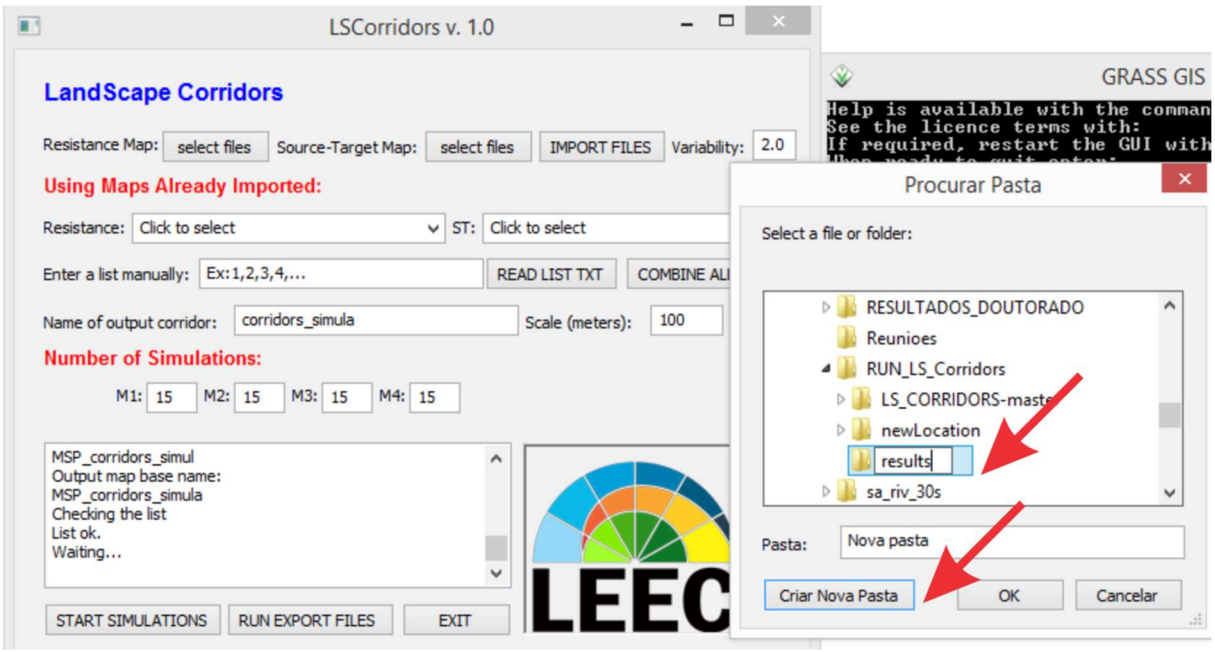


Figure 32 – Defining the output folder to results.

LSCorridors will start the simulation. A message will appear on the display indicating that the simulation has started, and you can follow the corridors simulation in the sh window (Figure 33 and 34). Another message will appear when the simulation have finished (Figure 35). LSCorridors will then automatically export raster files to output folder defined in above. After that click in “EXIT” button.

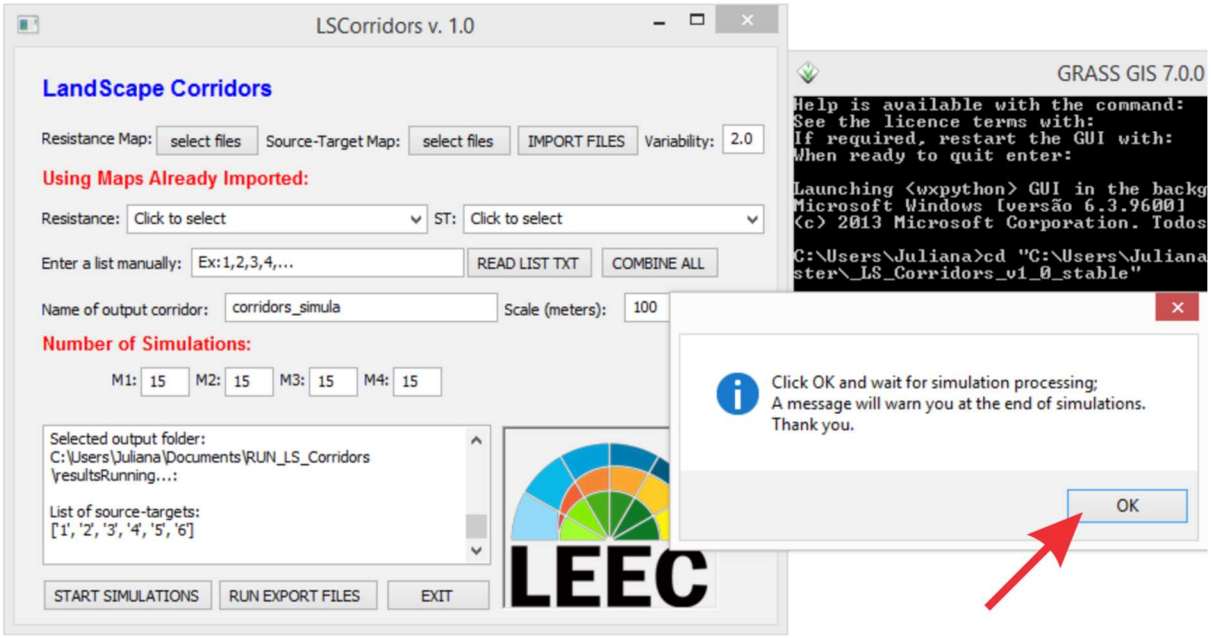


Figure 33 – Starting the simulations.

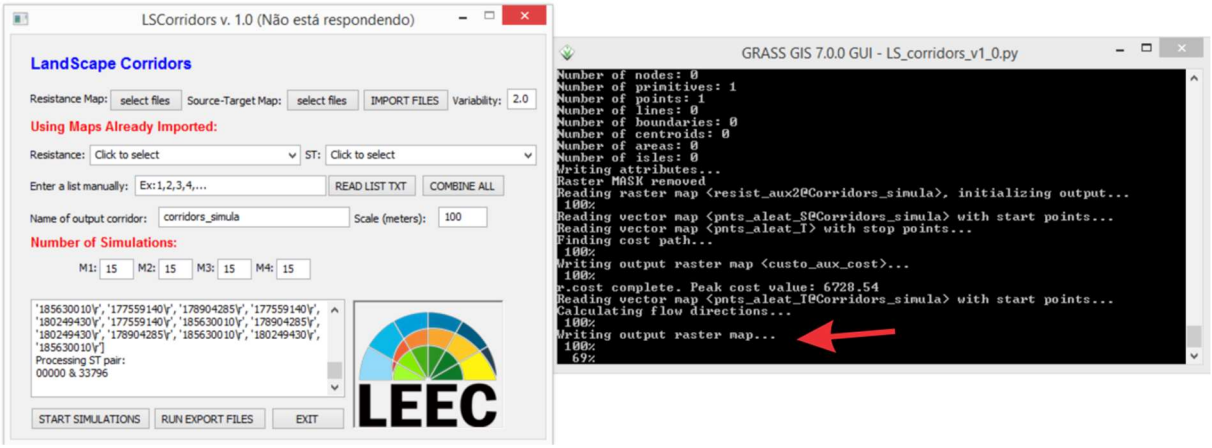


Figure 34 – Starting the simulations.

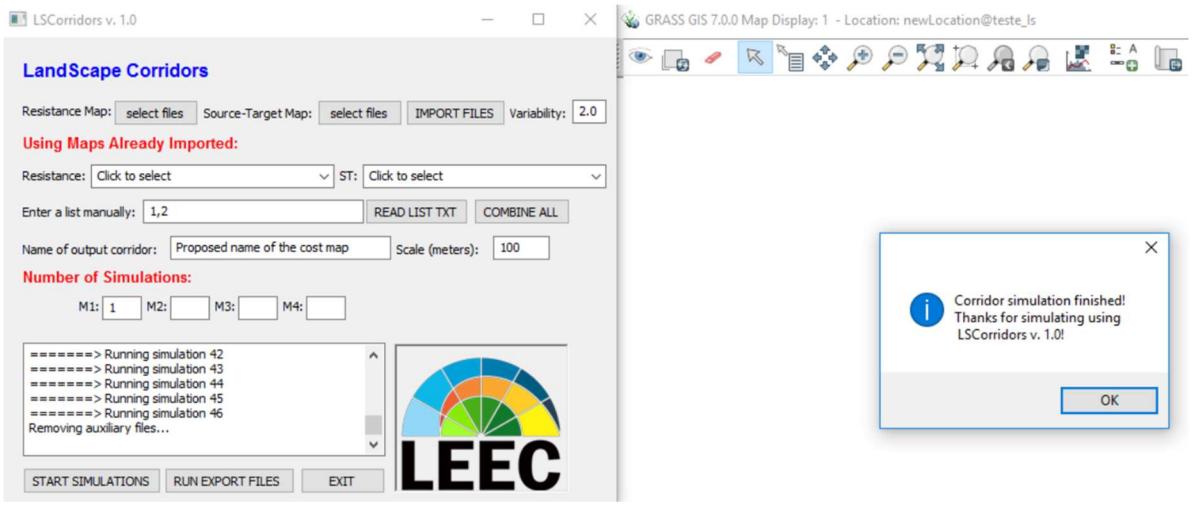


Figure 35 - End of simulations.

7. Output files

LSCorridors creates a set of output files per source-target pair: it generates in the output folder two .txt files and one folder for each source-target pair of patches (ST). The first file showed the parameters used in the simulations and the simulation's time.

Each folder contains a set of four GIS files for each simulated corridor (extensions ".shp", ".dbf", ".prj" and ".shx"), which enables you to open each simulated corridor as a shapefile file in a GIS software.

The end of the names of these files and folders indicate the source-target patches between which the corridors were simulated and the previously defined name of the output corridor (Figure 36).

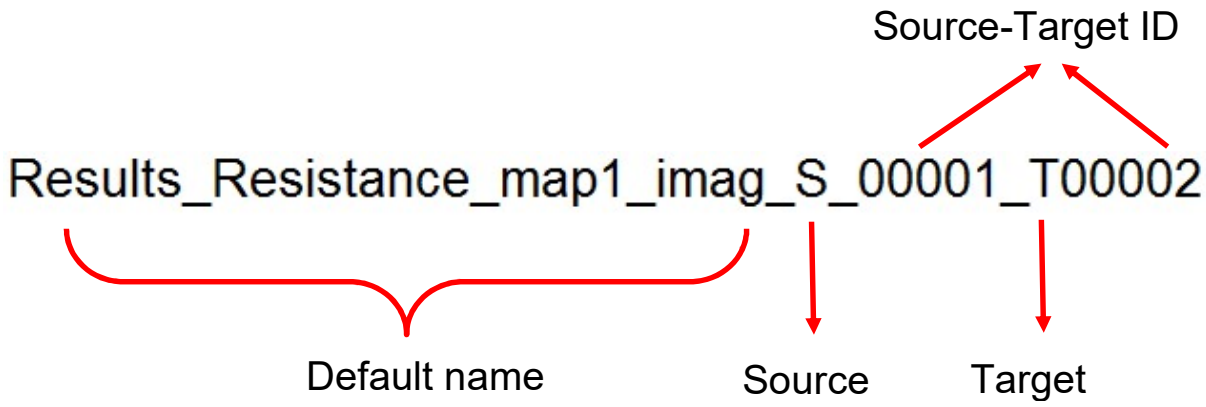


Figure 36 - Output file name

7.1 Descriptive data files

LSCorridors available ".tif" files correspondents to each pair simulated. The file has the identification of each method used in the simulation, considering each pair. The package also available a RSFI (Route Selection Frequency Index) map to each method. This map is the sum of the all maps simulated, and the final map shows how many of the corridor simulations passed through a given route. High RSFI values indicate areas (i.e. pixels) that have a high probability of being suitable as corridors between a given source-target pair. To analyse the corridors we recommended the use of the RSFI map.

In the output folder there are .txt files. These files are descriptive data of each pair of ST patches, with each row representing one simulation. The columns represent, respectively:

a. the name of the resistance map used;
b. the simulation mode (M1, M2 etc);
c. the number of the simulation;
d. the length of the corridor in meters;
e. the cost of the corridor (sum of resistance cost values of each pixel occupied by the simulated corridor);
f. the longitude coordinate of the point of the source patch;
g. the latitude coordinate of the point of the source patch;
h. the longitude coordinate of the point of the target patch;
i. the latitude coordinate of the point of the source patch;
j. the euclidean distance between the points of the source and target patches.

Although LSCorridors generates one descriptive data file for each ST pair of patches, it is possible to aggregate all files in a single one. We prepared a script for this in the R software (the R file is available at: https://github.com/LEEClab/LS_CORRIDORS.

```
install.packages("stringr")
library(stringr)

setwd("C:\\BD_demo\\output")
all.corr<-matrix(,,11)
corr.files<-list.files(pattern="txt")

for (i in 1: length(corr.files)){
  corr.temp<-read.table(corr.files[i], header=T, sep=",")

  #creating a character with Source-Target_ID
  sour.track<-str_locate(corr.files[i],"S_")
  tar.track<-str_locate(corr.files[i],"T_")
  end.track<-str_locate(corr.files[i],".txt")
  sour<-substr(corr.files[i],sour.track[,1],(tar.track[,1]-2))
  tar<-substr(corr.files[i],tar.track[,1],(end.track[,1]-1))
  ST_ID<-paste(sour,tar,sep="_")

  #inserting Source-Target_ID in dataframe
  data.temp<-cbind(ST_ID,corr.temp)
  colnames(all.corr)<-colnames(data.temp)
  all.corr<-rbind(all.corr, data.temp)
  all.corr<-na.omit(all.corr)
}

write.table(all.corr,file=paste(substr(corr.files[i],1,sour.track-2),"_allCorridors",".txt",sep=""), sep=",",
row.names=F)
```

You need an internet connection when using this script for the first time in order to download the “stringr” package.