# Appendix C - Manual for LSMetrics v. 1.0.0

Bernardo Niebuhr, Renata Muylaert, Felipe Martello

# Contents

## Welcome to the LSMetrics wiki manual!

LandScape Metrics (LSMetrics) is a free and open-source package that calculates landscape connectivity and other ecologically-scaled landscape metrics. It is intended to be used as a tool for environmental research as well as for landscape management, conservation, and restoration. It uses land use or binary class (e.g., habitat/non-habitat) maps to calculate metrics of structural and functional connectivity, edge-based landscape metrics, and landscape diversity indices. It incorporates edge depth and the capacity of organisms on crossing gaps between habitat patches to re-classify and calculate metrics considering the landscape at the perspective of different species and functional groups. Besides, metrics are spatially explicit and may be assessed at multiple scales.

LS Metrics was developed in Python 2.7 and runs within a GRASS GIS environment (currently, GRASS 7.0.x or newer). Here you will find a brief description of the program functionalities so you can quickly use it. We wish to make sure you can easily install GRASS, import your input maps, run LSMetrics and export or use its output in various ways.

If you have suggestions on how to improve this tutorial, feel free to contact us or open an issue.

### Authors and contributors

Authors:

- Bernardo B. S. Niebuhr <bernardo_brandaum@yahoo.com.br>
- Felipe Martello <felipemartello@gmail.com>
- John Wesley Ribeiro <jw.ribeiro.rc@gmail.com>
- Mauricio Humberto Vancine <mauricio.vancine@gmail.com>
- Milton Cezar Ribeiro <mcr@rc.unesp.br>

Contributors:

- Juliana Silveira dos Santos <juliana.silveiradossantos@gmail.com>
- Vinicius Rodrigues Tonetti <vrtonetti@gmail.com>
- Renata L. Muylaert <renatamuy@gmail.com>
- Vanessa Ellen Weidt Campos <vanessaellen@hotmail.com>

## Citation

Please cite LSMetrics when using the package in your work:

Niebuhr, B. B. S.; Martello, F.; Ribeiro, J. W.; Vancine, M. H.; Muylaert, R. L.; Campos, V. E. W.; Santos, J. S.; Tonetti, V. R.; Ribeiro, M. C. Landscape Metrics (LSMetrics): a spatially explicit tool for calculating connectivity and other ecologically-scaled landscape metrics. *In preparation*.

## LSMetrics Copyright and License Information



If you want to know more about free and open source software, look this and that.

## Brief description

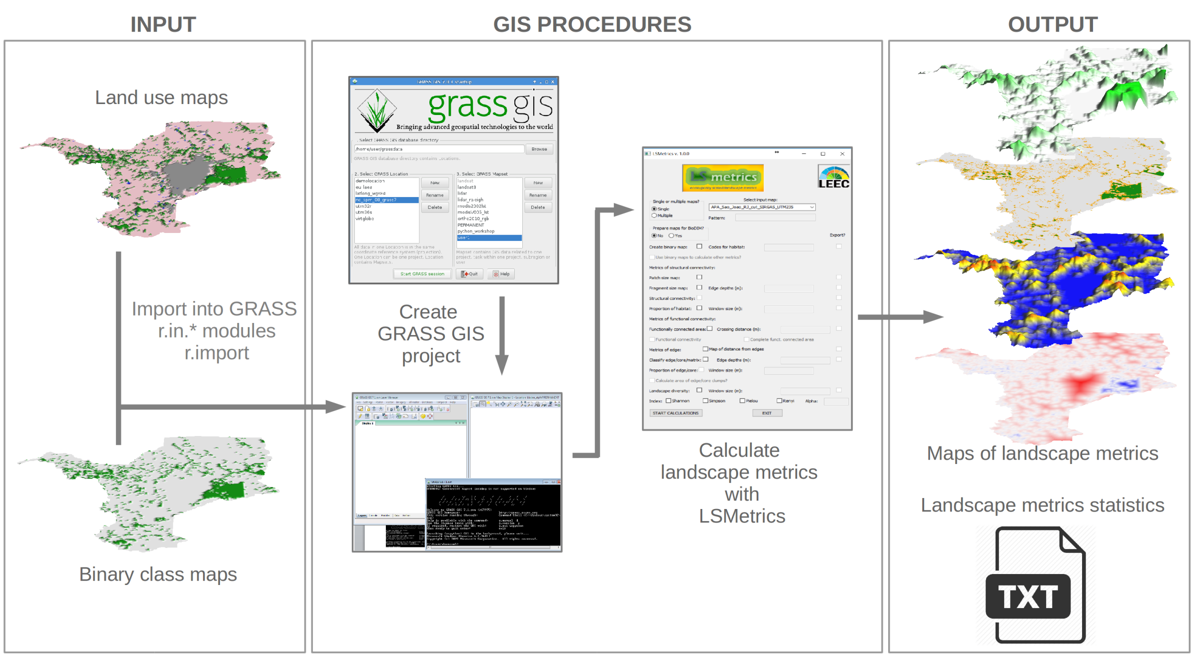
### The package

LandScape metrics (LSMetrics) is a free and open source package developed calculates multiple landscape metrics for raster data.

The package uses input raster maps with integer values only, in which each cell represents an area considered to be homogeneous, such as a land use or vegetation type. The maps can be either binary (1 = habitat, 0 = non-habitat) or multi-class (e.g. land use and land cover maps). The majority of landscape metrics are calculated using binary class raster maps, except for the landscape diversity indices, which only make sense for multi-class raster maps. LSMetrics may also transform multiple class maps into binary ones before the calculation of metrics.

Once a GRASS project is created and raster maps are imported into it, using r.import or r.in.gdal (or other r.in.*) modules, for instance, LSMetrics may be run in two ways (see the Figure below). The first is calling a python application and opening the GUI; the second is building a Python script (or opening a Python shell inside GRASS GIS prompt) and calling each landscape metric as a Python function. Both methods allow the users to run multiple metrics with various parameters and scales, for multiple maps, in a single run.

The output maps consist in raster maps for each chosen landscape metric, as well as text files with statistics of some of the metrics at the patch or class level, to be analyzed with any statistical software.




### Metrics

LSMetrics currently perform the following calculations:

Preparation of inputs:

- Transformation of land use maps in binary class maps

Metrics of *structural connectivity*:

- Patch size

- Fragment size
- Structural connectivity
- Proportion of habitat

Metrics of *functional connectivity*:

- Functionally connected area
- Functional connectivity
- Complete functionally connected area

Edge-based metrics:

- Classification in core/edge/matrix
- Classification in landscape elements: edge/core/stepping stones/corridors/branches/matrix
- Binary maps: edge/non-edge
- Binary maps: core/non-core
- Proportion of edge area
- Proportion of core area
- Area of clumps of edge and core areas

Landscape diversity (through the r.diversity GRASS addon):

- Shannon
- Simpson
- Pielou
- Rényi

For more information on the metrics calculated and details on implementation, look at the publication:

Niebuhr, B. B. S.; Martello, F.; Ribeiro, J. W.; Vancine, M. H.; Muylaert, R. L.; Campos, V. E. W.; Santos, J. S.; Tonetti, V. R.; Ribeiro, M. C. Landscape Metrics (LSMetrics): a spatially explicit tool for calculating connectivity and other ecologically-scaled landscape metrics. *In preparation*.

## The repository

The LSMetrics repository is organized in 7 folders:

- `_LSMetrics_v1_0_0`: Here the main pieces of the LSMetrics code are located:
    1. `LSMetrics_v1_0_0.py`: main script.

2. `test_LSMetrics.py`: a Python script with the list of functions of LSMetrics and their usage as Python functions (outside GUI).
3. `r_diversity.py`: The r.diversity GRASS addon as python code, used to calculate landscape diversity indices in LSMetrics.

- `previous_versions`: Old versions of the code.
- `grassdb_test`: raster maps for testing. This includes:

1. `APA_Sao_Joao_RJ_cut_SIRGAS_UTM23S.tif`: A land use map in Rio de Janeiro state, Brazil, inside the Golden Lion Tamarin occurrence region. The map was classified based on LANDSAT 7 satellite images.
2. `SP_RioClaro_use_raster.tif`: A land use raster of the municipality of Rio Claro, State of São Paulo, Brazil.

# Using LSMetrics

Here we describe how to install and use LSMetrics step-by-step. If you have questions or suggestions to make it more didatic, please contact us!

## 1. Downloading LSMetrics

First download the .zip package or clone it into your local enviroment using GitHub Desktop or `git`:

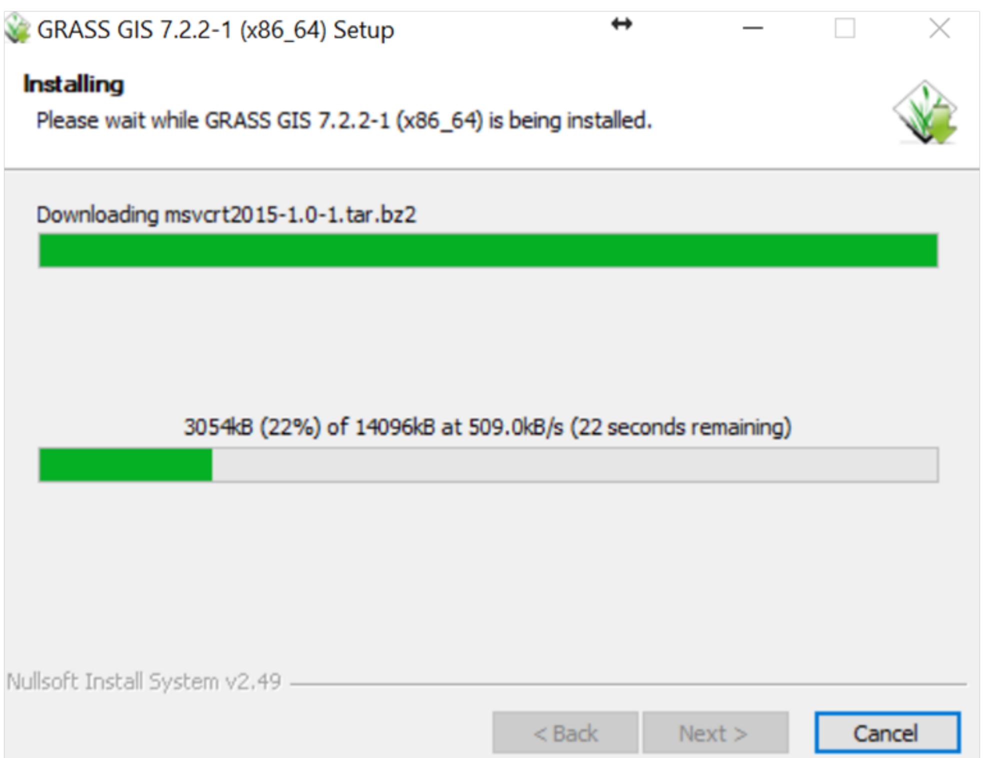
```
# Choose the directory where to clone LSMetrics
cd path/to/LSMetrics_dir

# Clone it
git clone "https://github.com/LEEClab/LS_METRICS.git"
```

## 2. Have GRASS GIS installed

LSMetrics runs within GRASS GIS and uses many of its modules and functionalities, so you first need to have GRASS installed in your computer. We recommend you install GRASS version 7.2.2. Follow the instructions.

### Installing GRASS GIS

LSMetrics was developed and runs inside GRASS GIS environment, version 7.2.x. Thus, we strongly recommend the use of one of these GRASS versions. It is possible to download this software at the GRASS GIS website.



### MS Windows

GRASS GIS is available for different versions of Windows. Download the `.exe` file and install it in your computer. Alternativelly, you may download the OSGeo4W package, which includes QGIS, GDAL, GRASS and other GIS tools. More information on GRASS installation on Windows can be found here.

### Ubuntu Linux

The installation of GRASS GIS in Ubuntu is simple. In the terminal, it is necessary to enter all following lines:

```
sudo add-apt-repository ppa:ubuntugis/ubuntugis-unstable
sudo apt-get update
sudo apt-get install grass
```

For more information or previous versions of GRASS, click here.

GRASS GIS is also available for other Linux distributions, but we have not tested LSMetrics on them. If you wish to test it, please informs us either by e-mail or by an issue, so that we can include information about it here (and thanks in advance!).

### MacOS

For information on GRASS GIS installation on MacOS, take a look here.

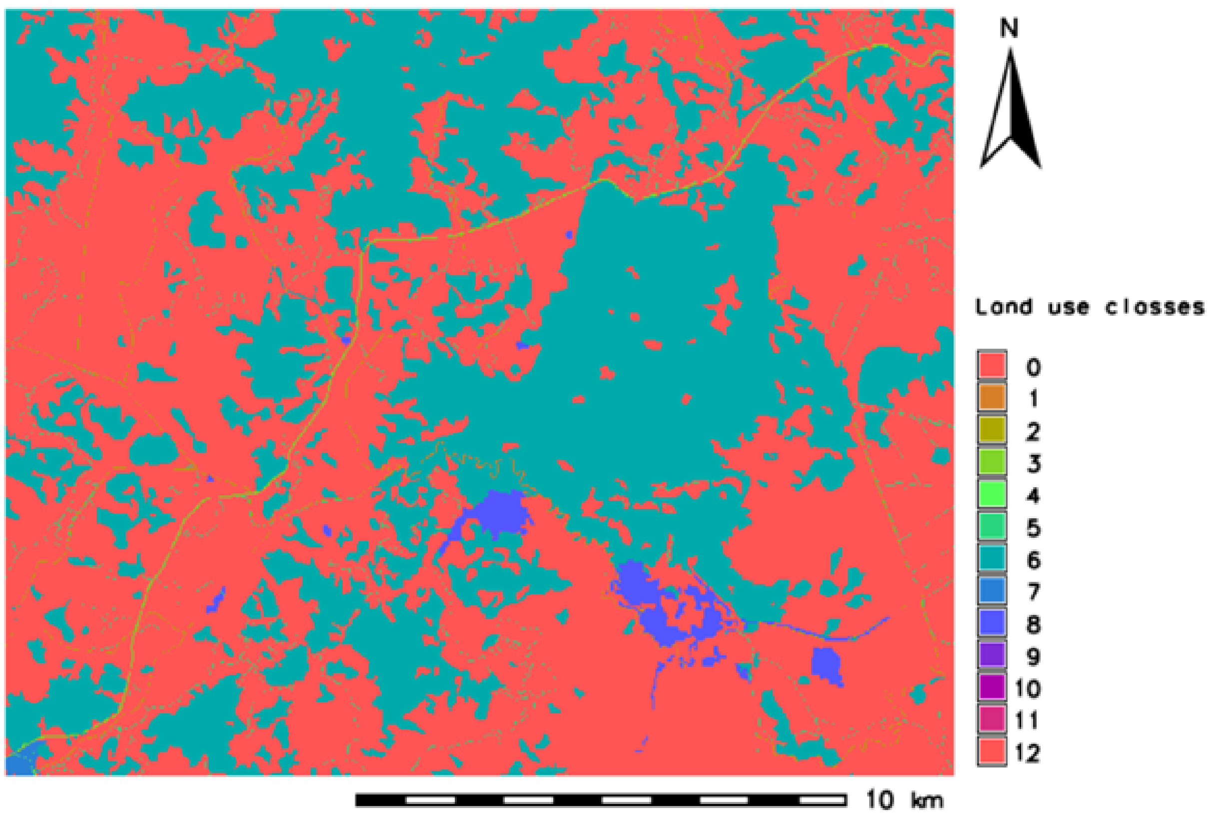
LSMetrics was not tested through the Graphical User Interface (GUI) in MacOS yet. If you want to do it, please tell us so that we can adapt the GUI slightly to it. However, LSMetrics package is expected to run in any MacOS platform through Python command line.

# 3. Preparation of inputs

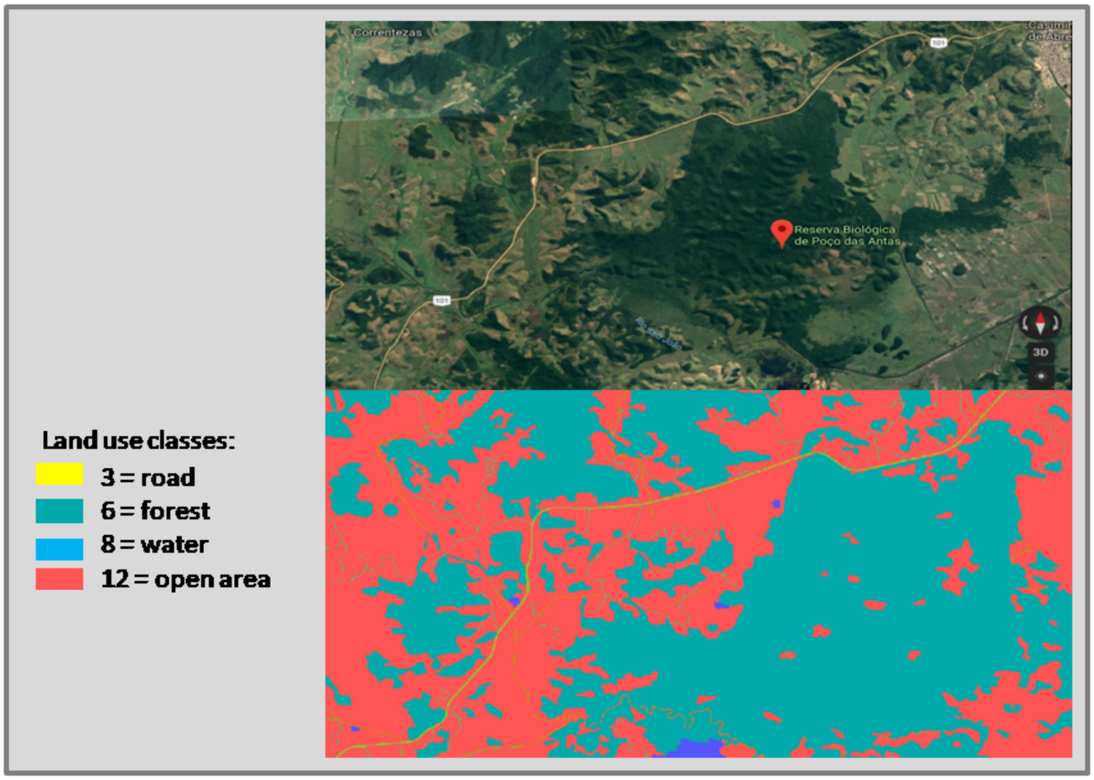
## Prepare your input maps outside GRASS GIS

LSMetrics input maps can be binary maps or non-binary land use maps. Binary maps contain “1” and “0” values; “1” represent a kind of habitat or land use of interest, while “0” represents all other land use classes. Non-binary maps can include different land use classes, such as different types of forests and matrices (pasture, plantations, highways). Each class is then assigned a integer code. For example, in the map shown below forest pixels have code “6”, pastures are “12”, water equals “8”, and roads equals “3”. The user must to define the codes of each class of the land use map using GRASS GIS or any other GIS software, so that they can select the codes of their interest in LSMetrics.

See in the figure below a land use raster representation with several classes, using as example a location in Rio de Janeiro, near the Poço das Antas Biological Reserve.




Note that the codes are showed as integers. See the main codes meaning below, in comparison with a satellite image for the same area:




The LS metrics repository also contains a map with land use classes in the municipality of Rio Claro, for testing purposes. We used the Rio Claro map to run the examples of metrics in the Output links.

Satellite images can be classified by image interpretation or automatic classification algorithms (supervised or unsupervised). If you still need to map your study area, consider the following auxiliary readings and video:

- QGIS tutorial by LEEC team
- Supervised and unsupervised classification
- Tutorial 1: Your First Land Cover Classification

There are many alternatives of land use maps and binary maps for different geographic regions. Take a look at some examples:

- MapBiomas
- Global Forest Watch
- Land use and land cover in Brazil

## Reference system

As LSMetrics uses meters and hectares the default units for calculating distances and areas, raster input maps must be exported in a metric coordinate reference system (CRS), such as Albers or UTM coordinate systems. Please, check the next sessions on creating a GRASS Location and be sure about your working EPSG code.

# 4. Starting

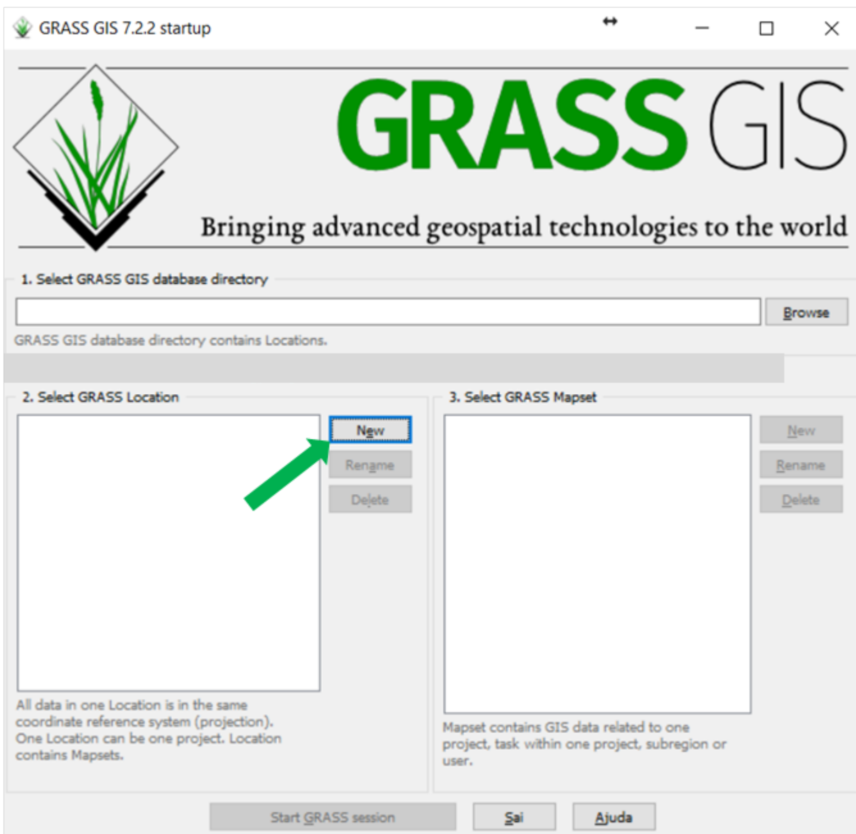
To start calculating metrics, first you need to create or open a GRASS GIS project (called location) and then open the LSMetrics GUI through command line.

## Open or load a GRASS project

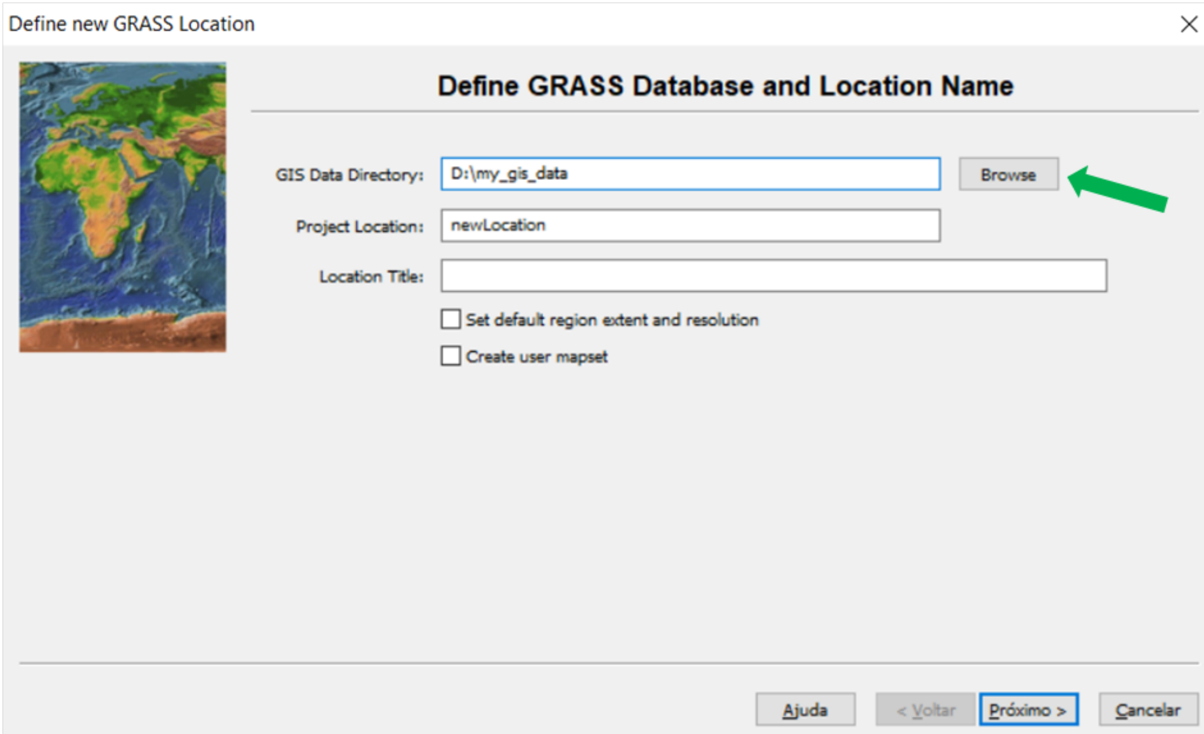
Open GRASS GIS and load or create your project location. If you do not know how to configure a project in GRASS, take a look here. Here we show one way of creating a GRASS location.

A GRASS location is a project that keeps all maps and spatial data cohesive under the same Datum and Projection. To work with your own GIS data, you must create a location with an **EPSG** (a code that represent a combination of datum and projection) that is similar to your data's. For example, if your rasters are in UTM 23 South projection with a SIRGAS 2000 datum, your GRASS location should be exactly on this projection. Follow the next steps to create a projection and import a raster map, so you can run LSMetrics with your own data.

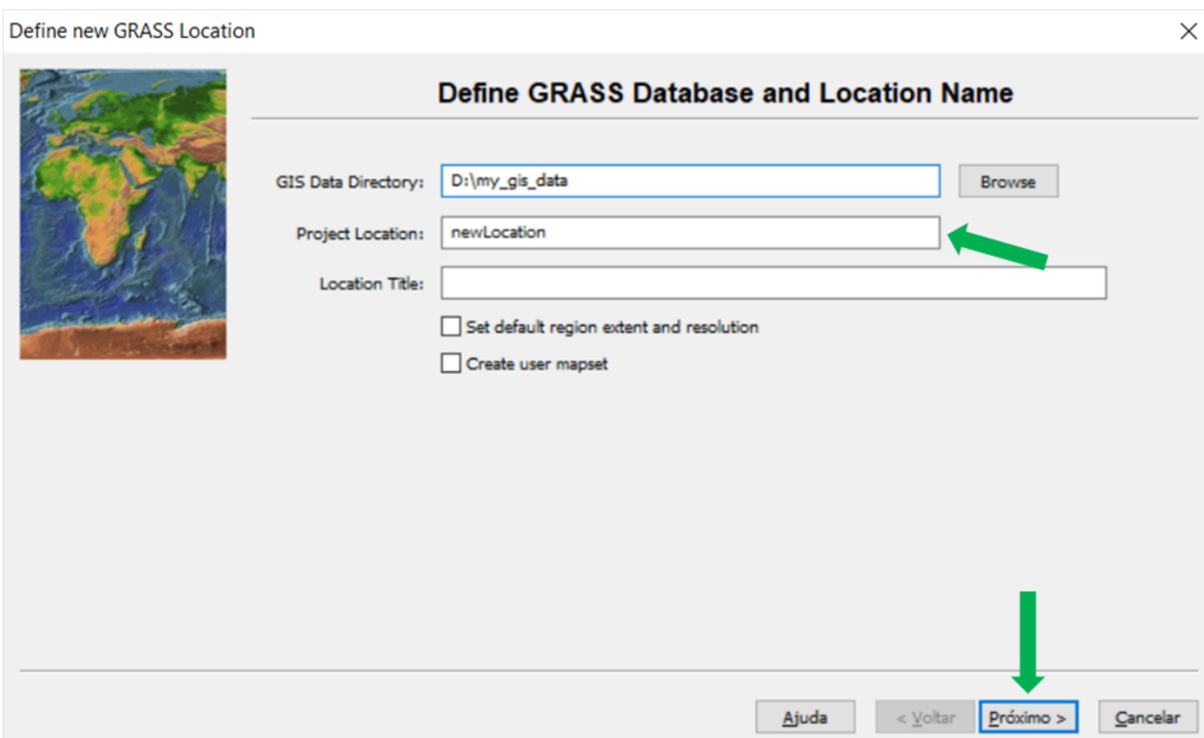
Click on New Location wizard as follows:



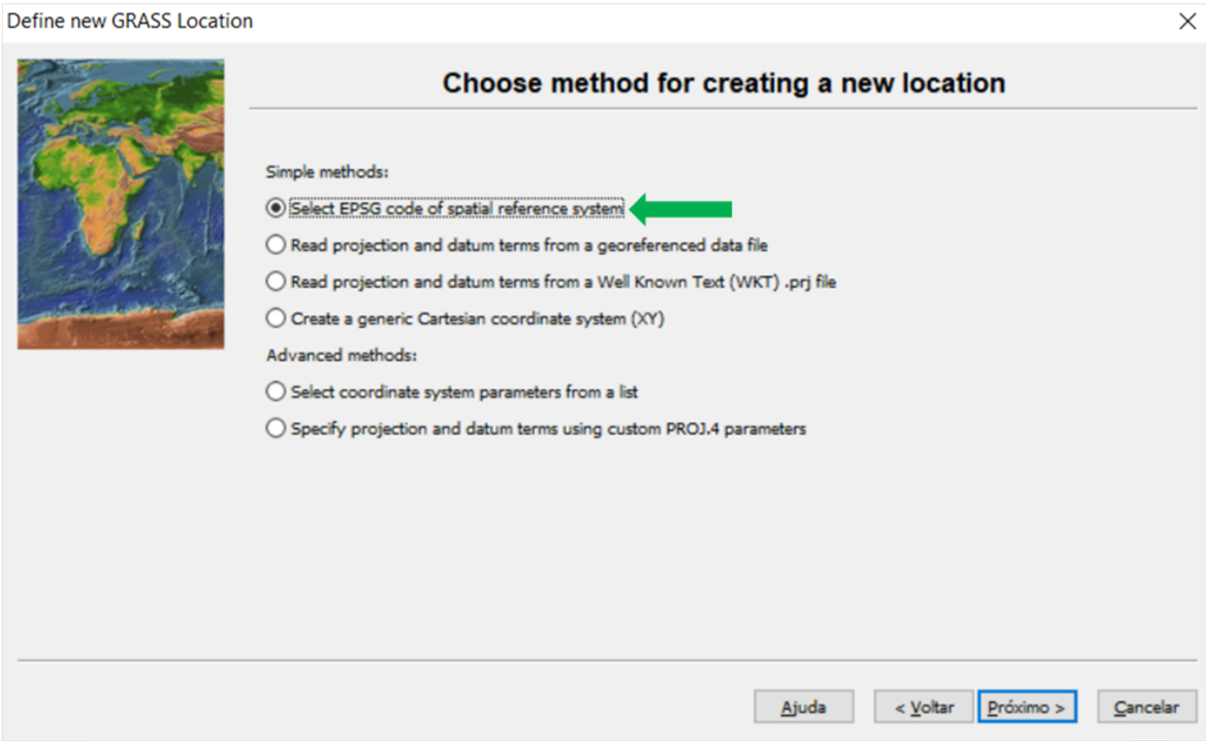

Browse the file containing the rasters of interest:

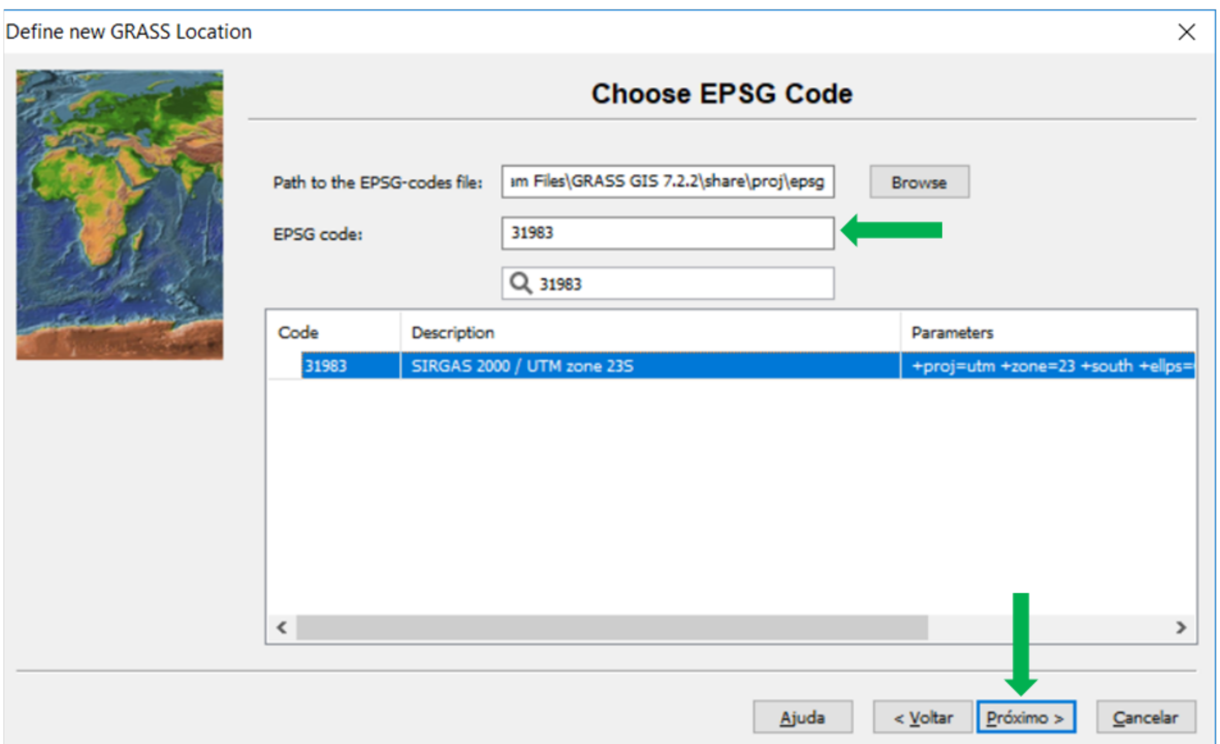



Choose a simple name for your Location (Here we keep as the default **newLocation**):
**(Please avoid using spaces and special characters on the name of the location!!!)**

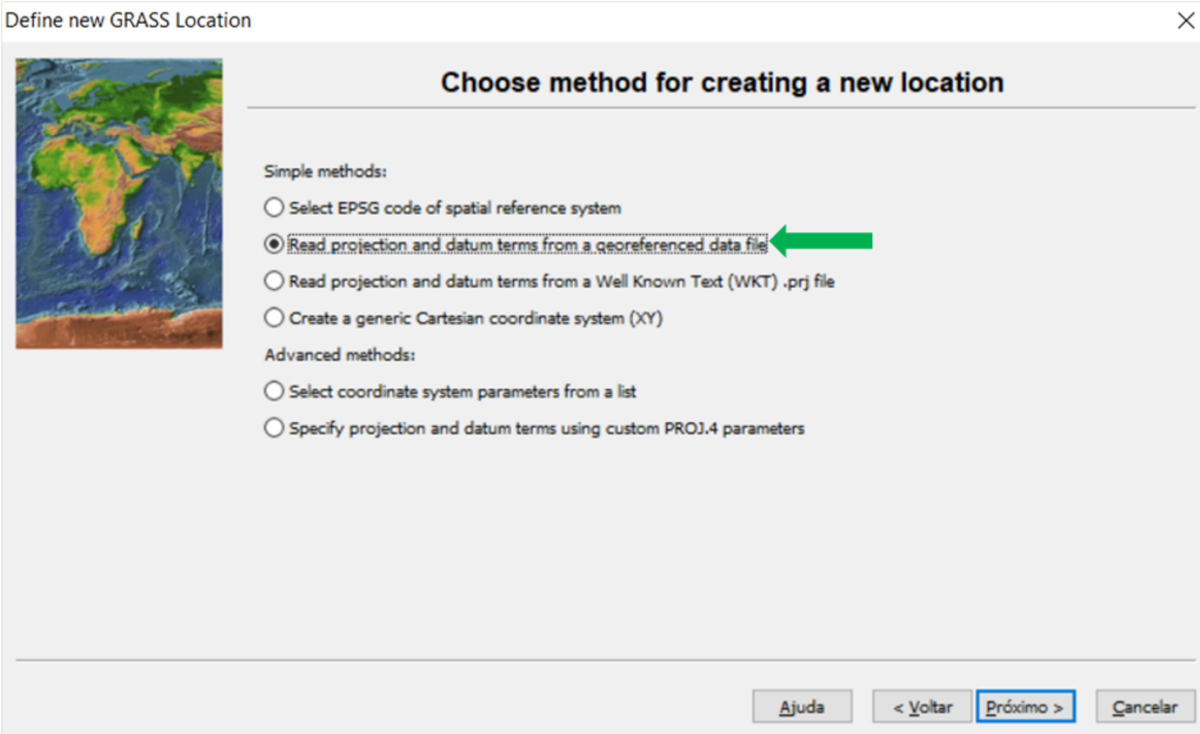



Now it is time to select your working EPSG, which is basically the spatial reference system of your data. If you don't know which one is it, figure it out here.

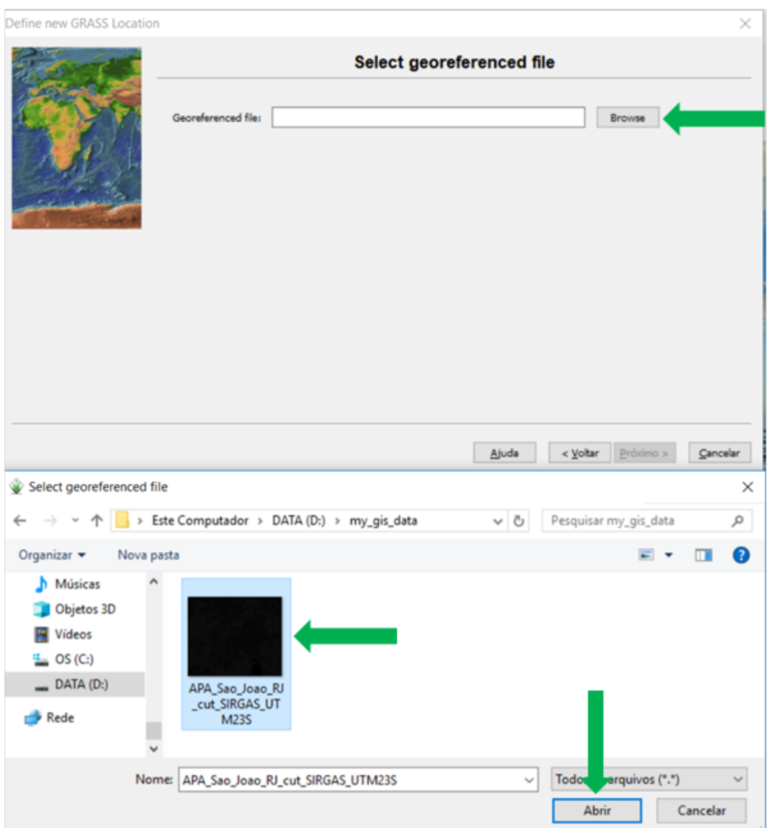


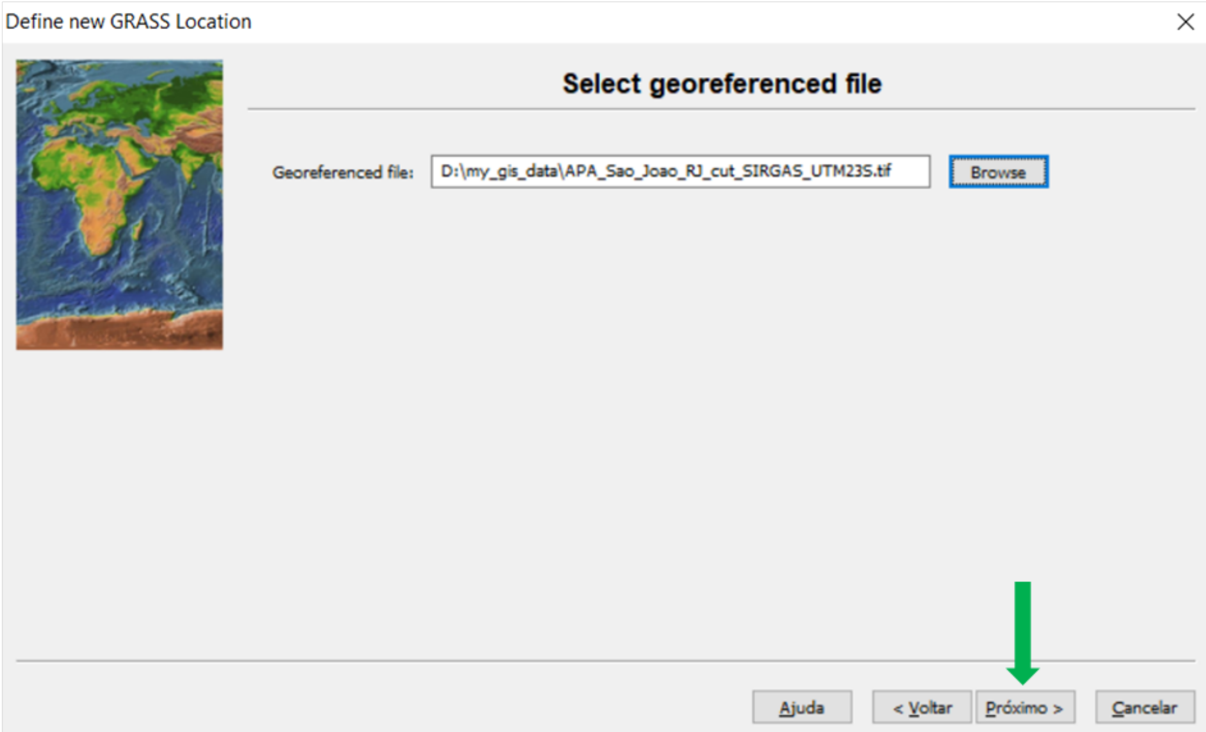



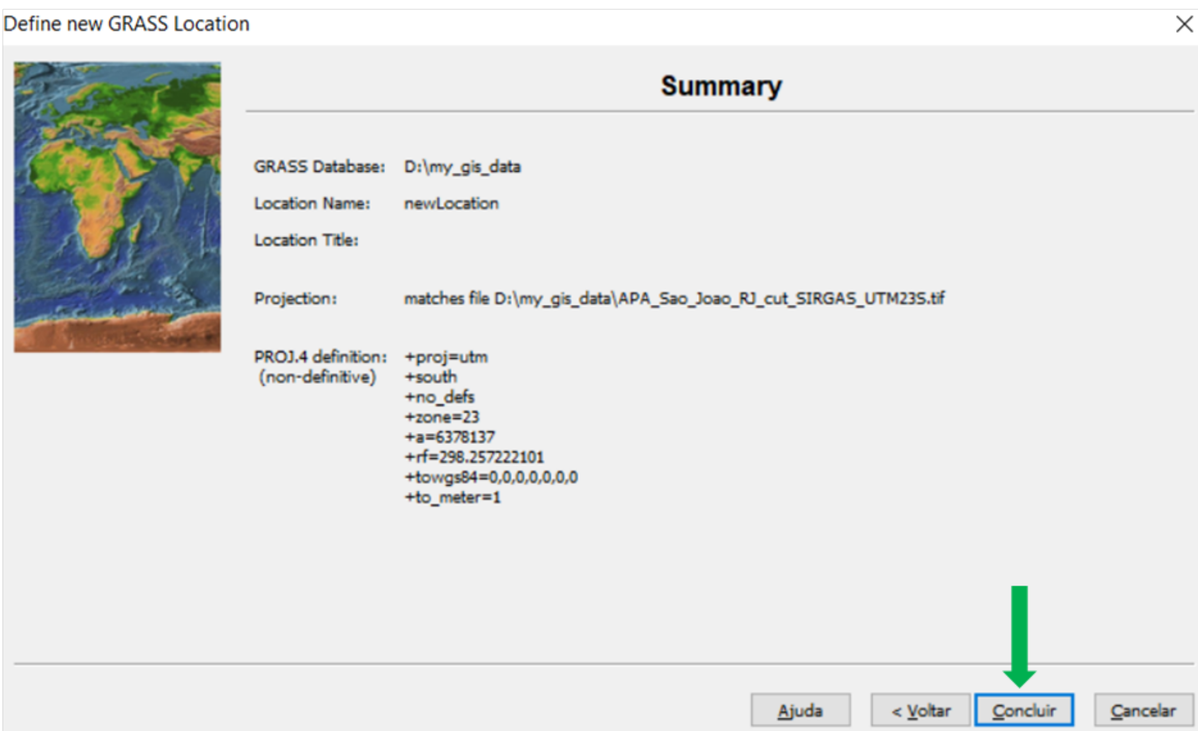

You also can easily read the right spatial references system from a georeferenced file.




Just find the file in you computer and GRASS will get the EPSG code from it. Follow the steps:





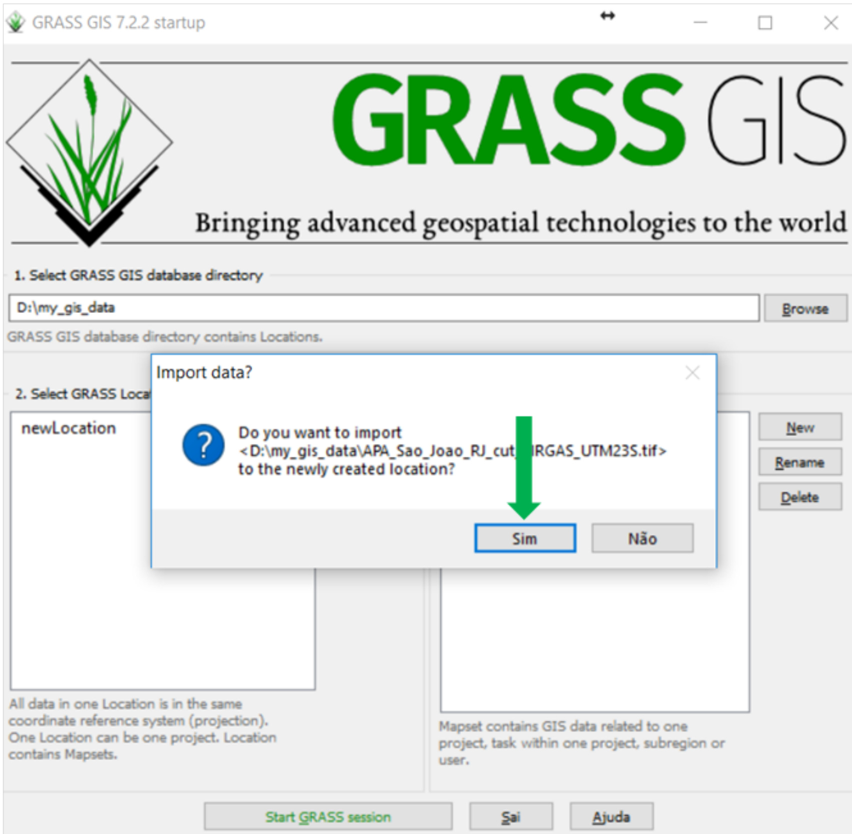




Done! Now you created a GRASS GIS location!

## Import your maps into GRASS (your own SIG data)

### Import maps right after creating the location

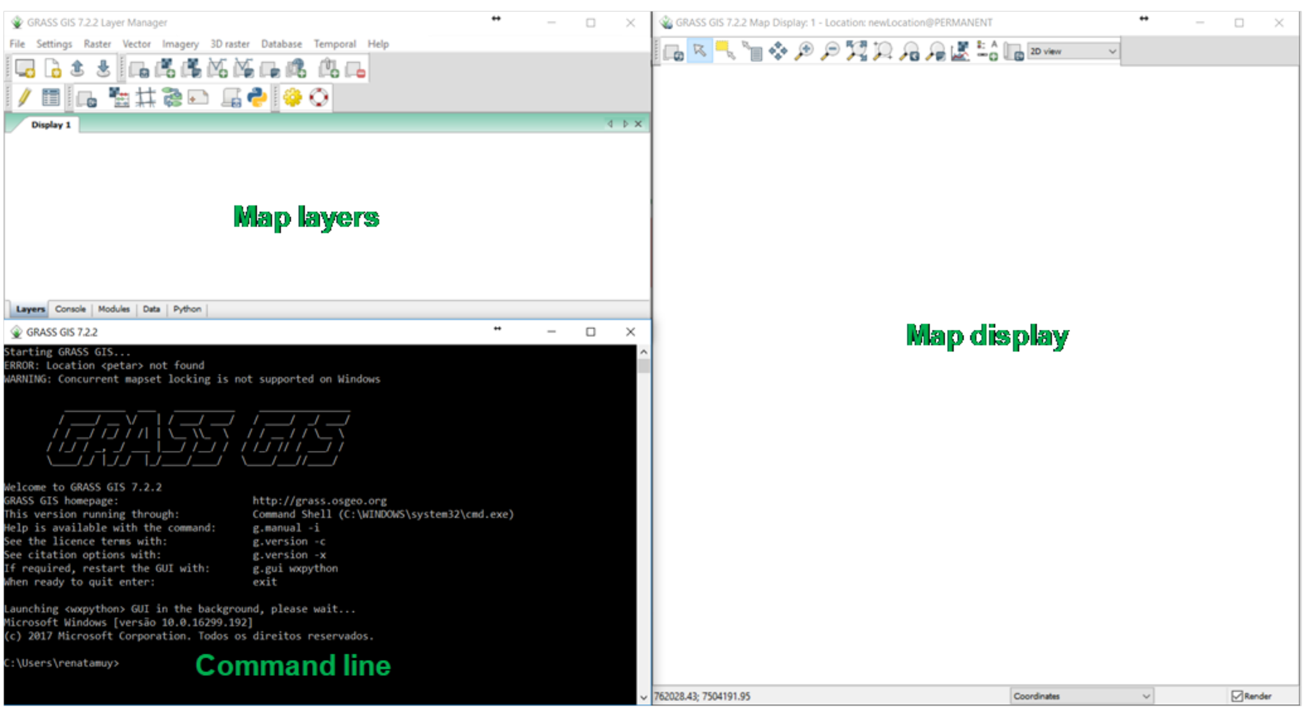
After creating a GRASS project, you can readily import the raster used for selecting the EPSG to GRASS environment. If you click "No", then you can also import your rasters easily - see below.



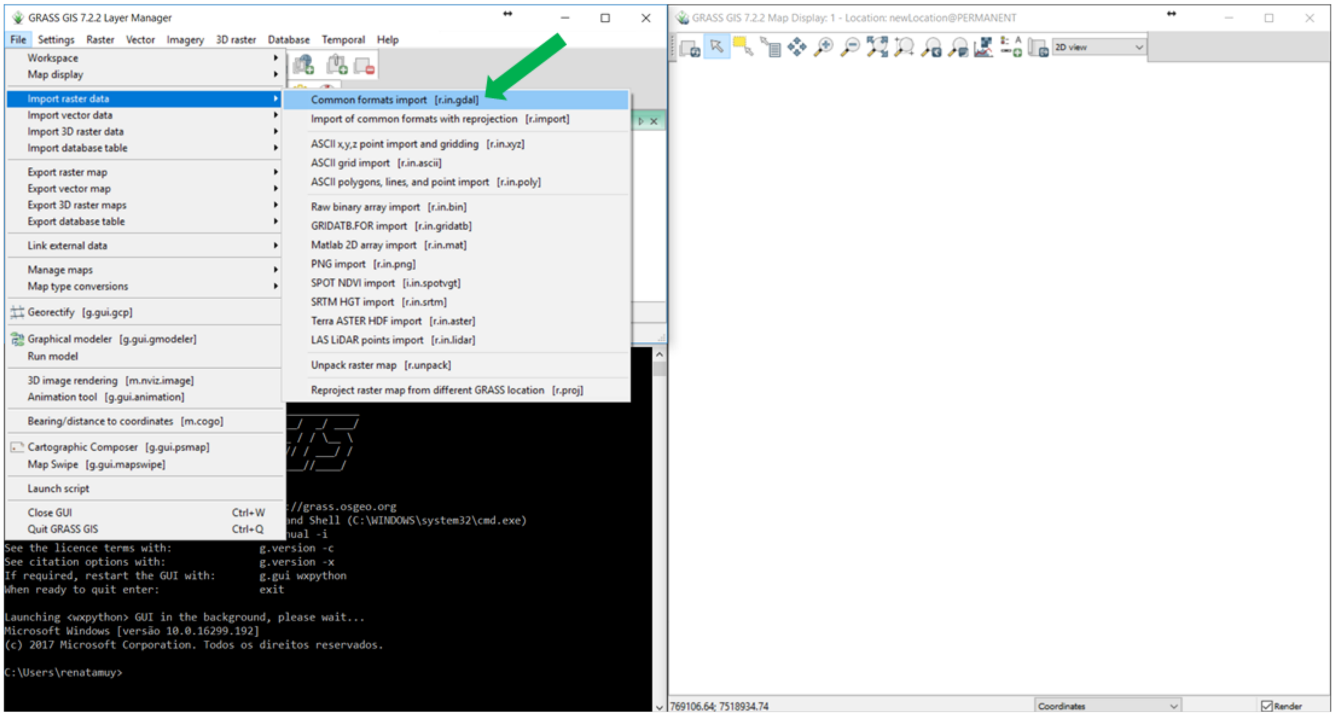

### Importing data within a GRASS session

First you can start your GRASS session by pressing the START GRASS button.

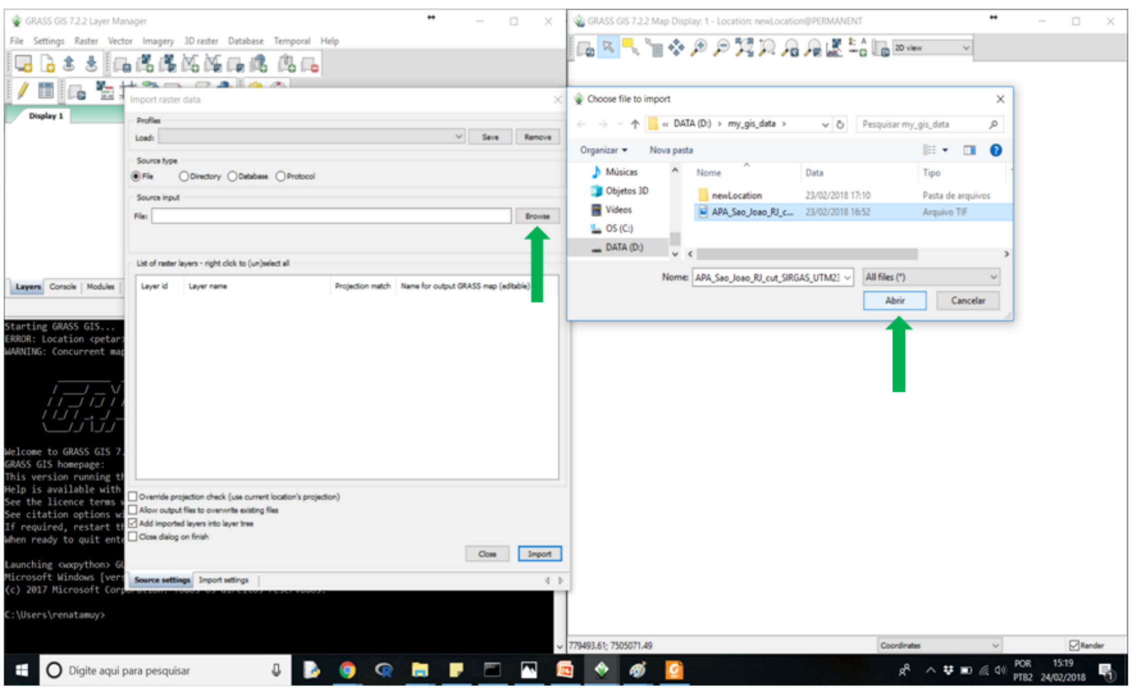

Now that you opened a grass session, see how GRASS-GIS looks like. Basically you have thee windows - a shell terminal, in which you can run GRASS procedures through command line; and two Graphical User Interface windows:



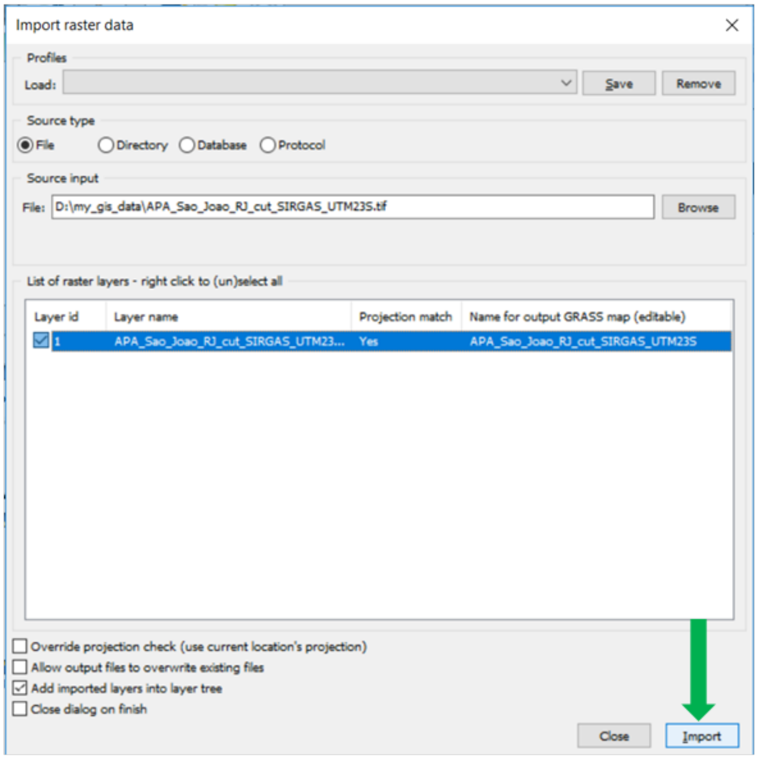

Let´s import your raster data using the function r.in.gdal, but clicking:




Where is your raster data? Browse to import:




And finally, click on "Import":




Follow the next session to visualize your imported map(s) in GRASS display with our example data.

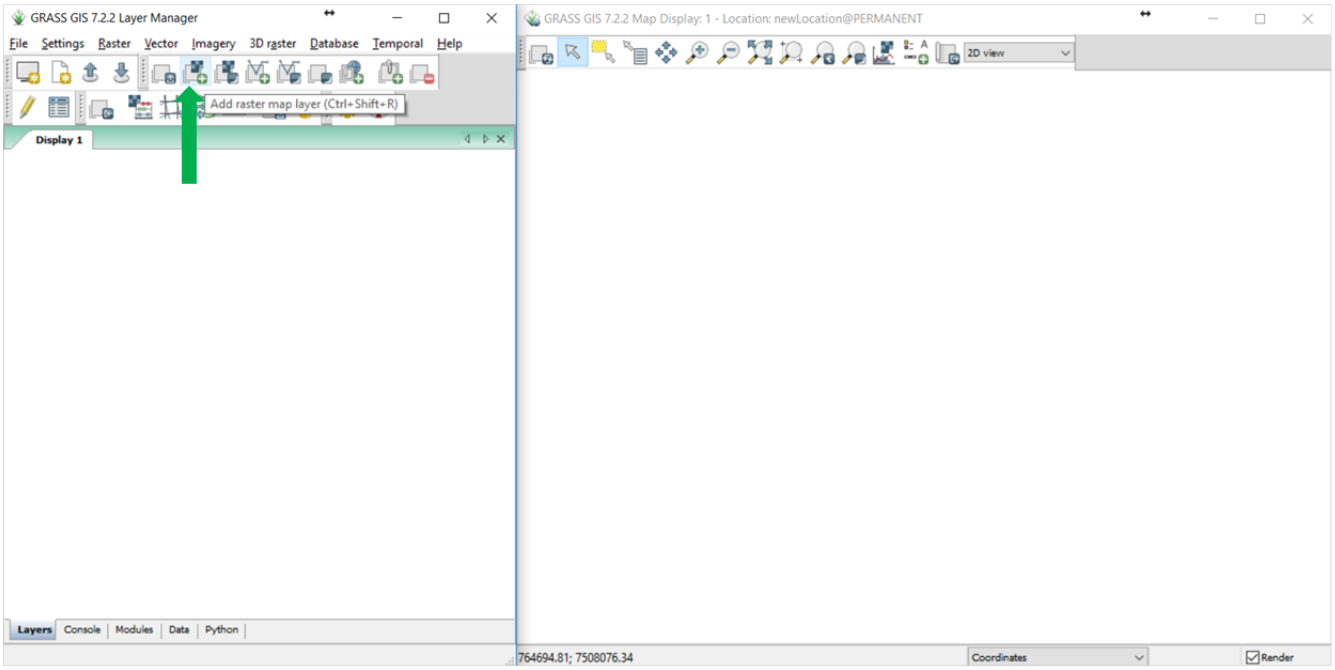
If you want more information on importing maps into GRASS, have a look at this and this.

**Observation: Make sure that your imported maps are in a projected coordinate reference system (CRS), in which distances are measured in meters, so that areas and distances are calculated in a meaningful way by LSMetrics.**

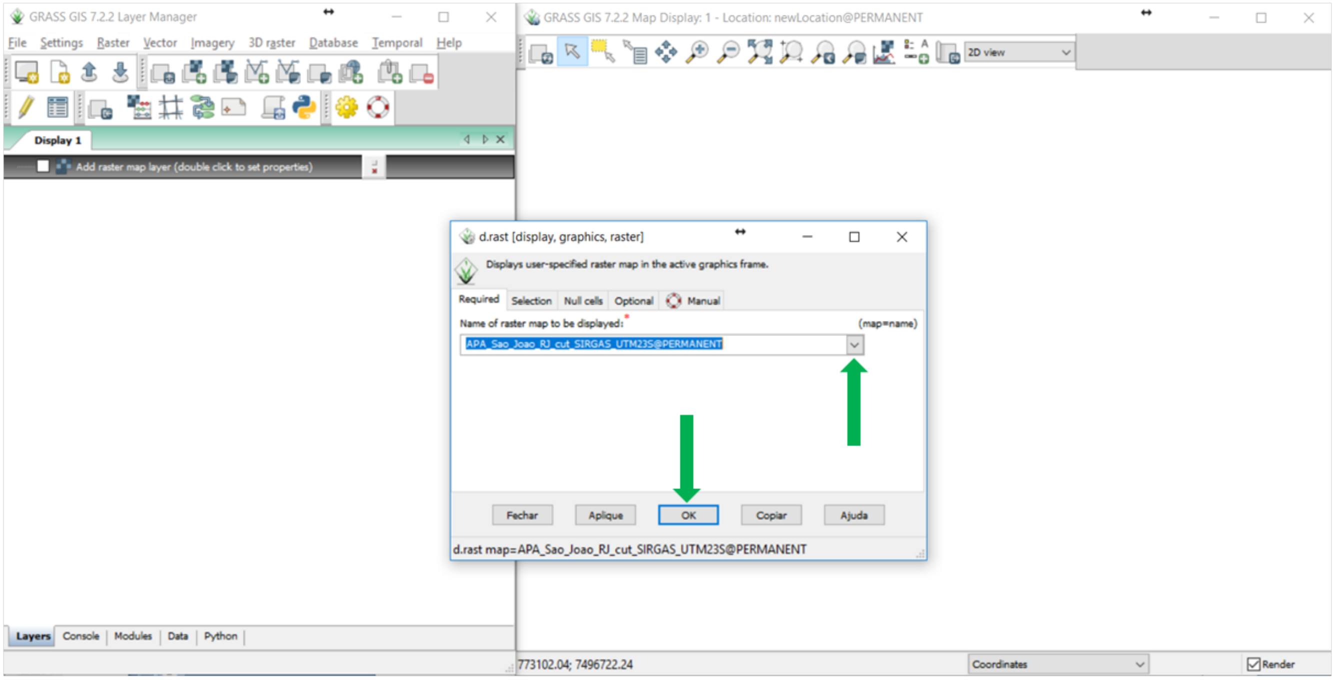
## GRASS database for testing

To make your life easier, we make available a GRASS database with two raster images, for testing. Select the file grassdb_test and **Start GRASS session** as follows:

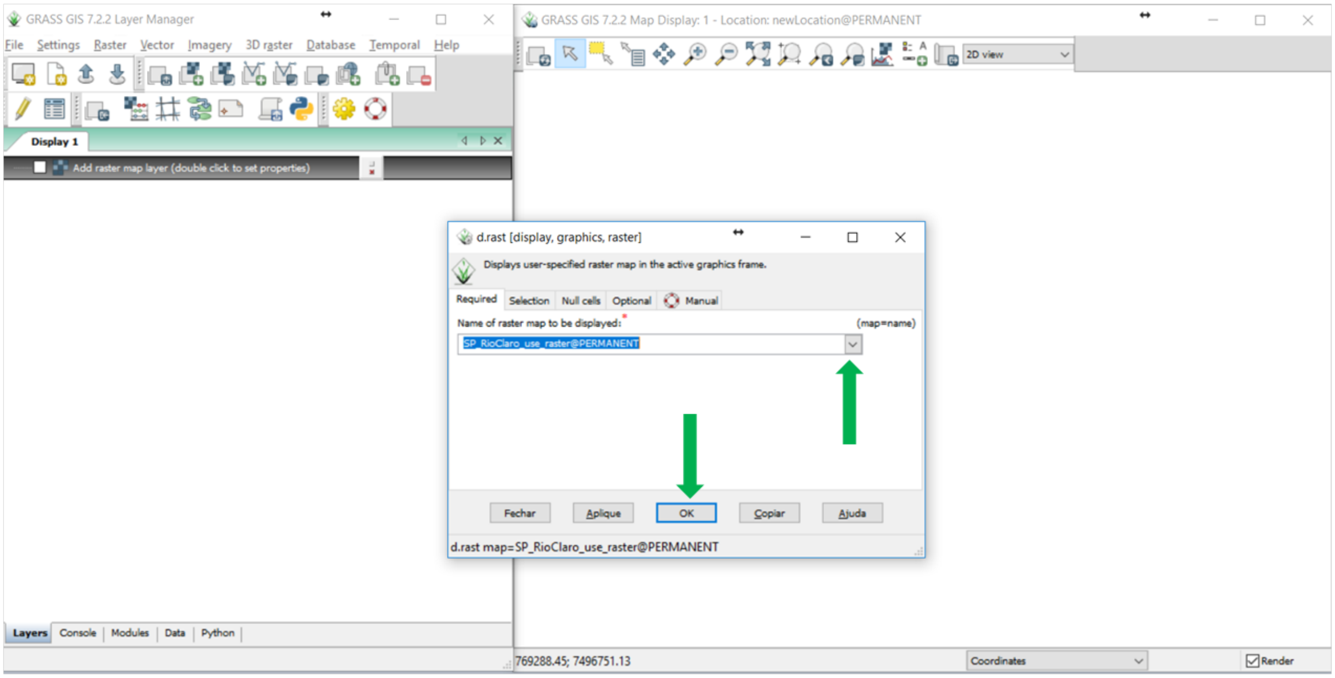

Note that you can display the maps either by coding or by clicking. If you choose clicking, then follow the steps to visualize out both example data:




Choose the São João raster map:

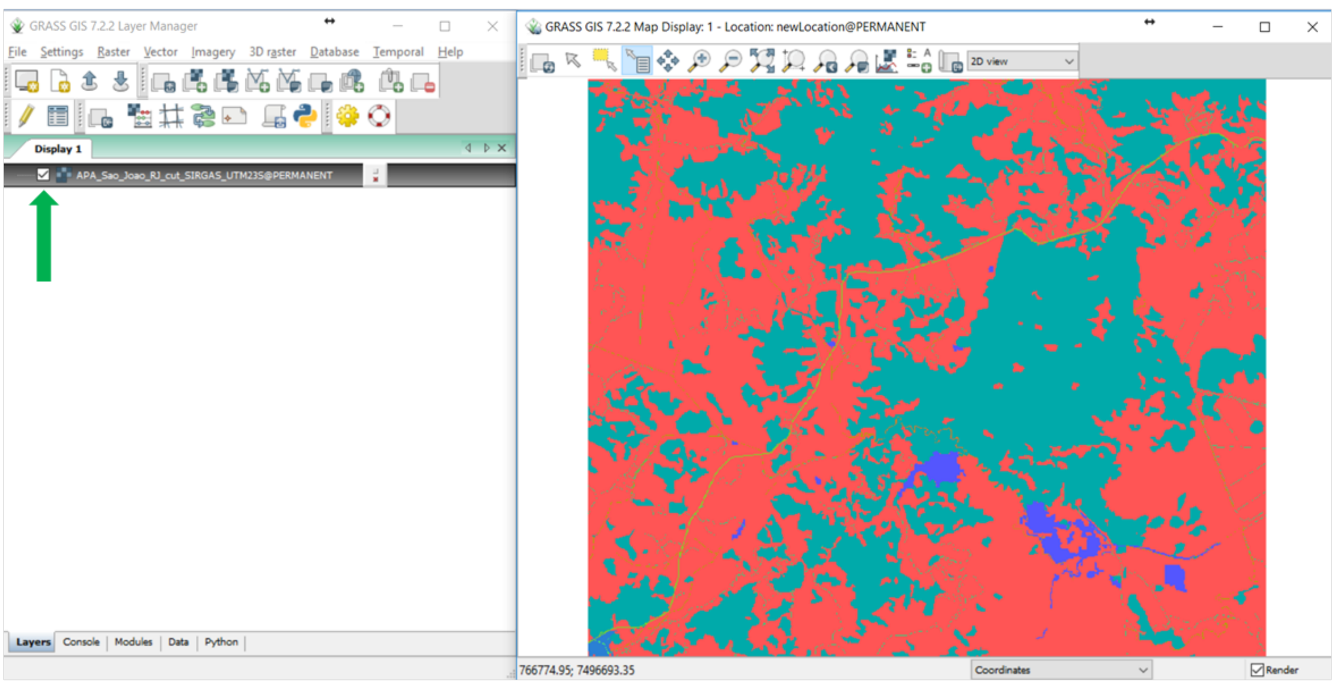



Choose the Rio Claro raster map:

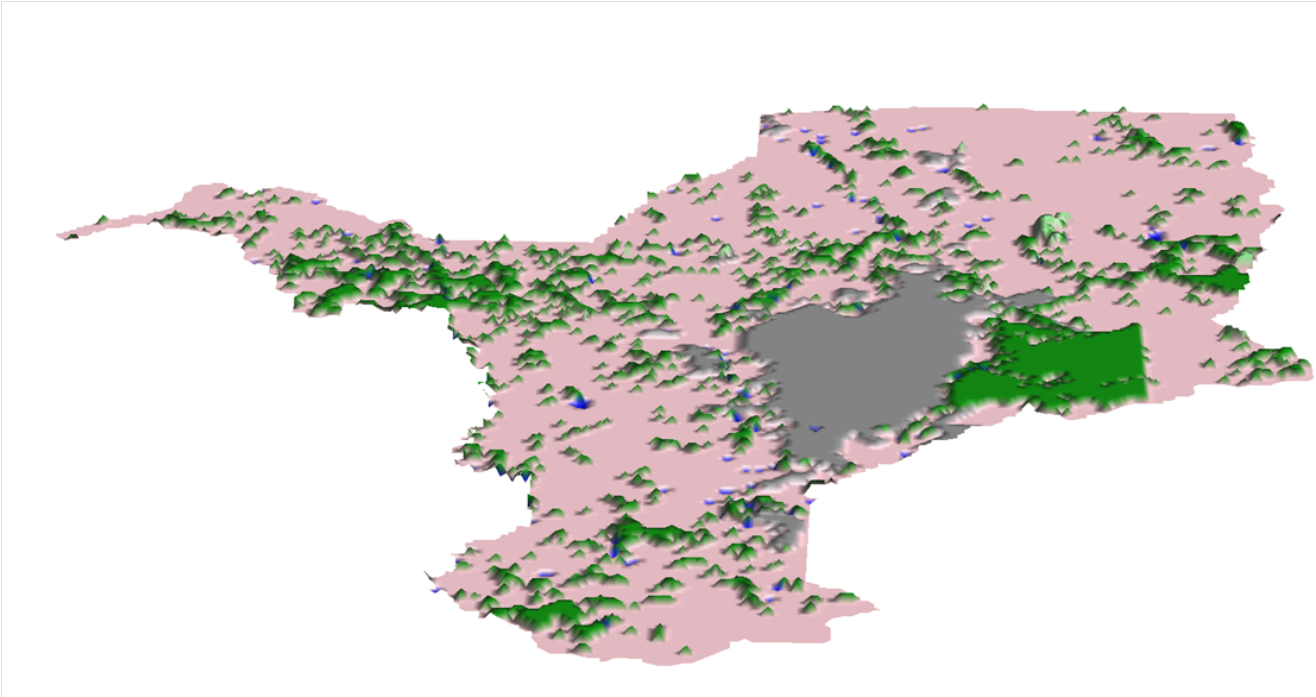



Now see below how the example maps will appear in GRASS display:

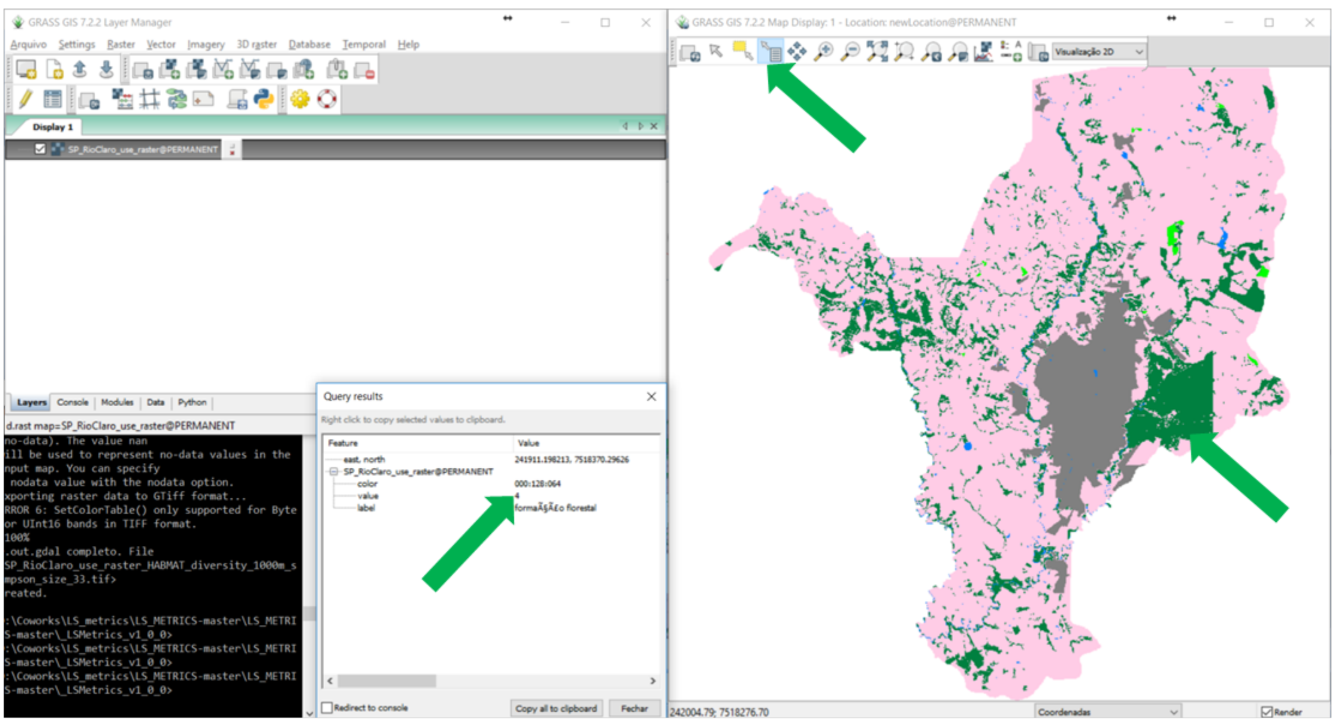
São João, State of Rio de Janeiro:




Rio Claro land use map, State of São Paulo:



On GRASS GIS display, you can evaluate the pixel values using the `Query results` tool. Se below:




Now see the code for showing the already existent example data:

```
d.rast map=APA_Sao_Joao_RJ_cut_SIRGAS_UTM23S@PERMANENT
```

```
d.rast map=SP_RioClaro_use_raster@PERMANENT
```

## Run LSMetrics

Now it is finally time to use the package itself. Use the GRASS terminal (the black screen) to start LSMetrics. First change to LSMetrics directory and run it using python:

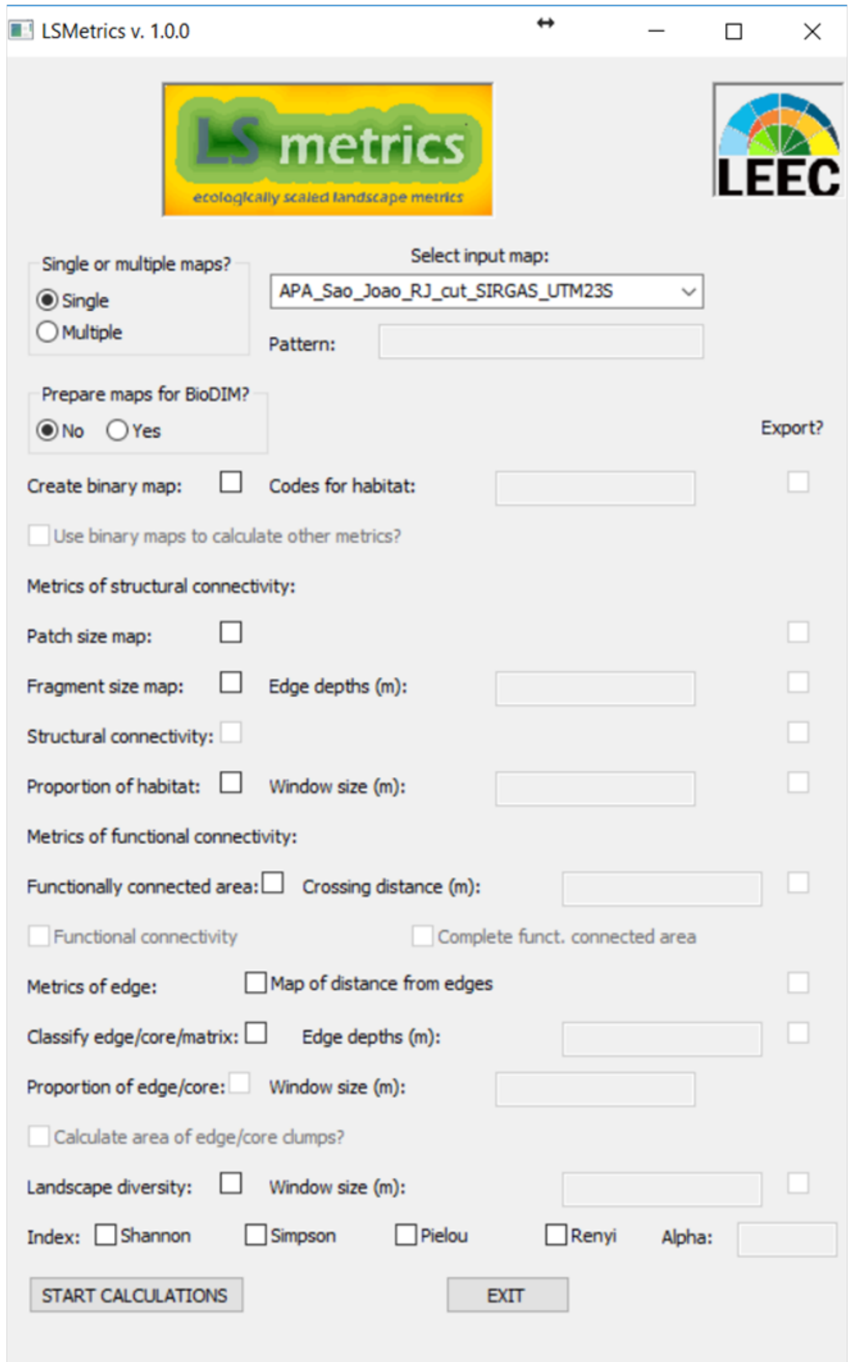
```
# Change to the code directory where you saved the program (with the version of LSMetrics you want to use)
cd "LS_METRICS/_LSMetrics_v1_0_0"

# Run LSMetrics
python LSMetrics_v1_0_0.py
```

### Note!

On MS Windows, you may have to change to the partition where the code is located before changing to its directory. E.g.:

```
# If the code is on the directory "D:/"
D:
cd "D:/LS_METRICS/_LSMetrics_v1_0_0"
```

Now take a look at the Windows GUI (the GUI may be slightly different on different operational systems, but is expected to run the same way):



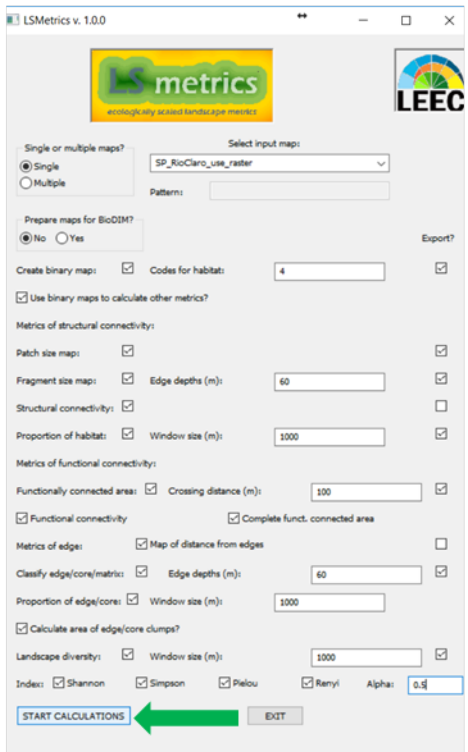

### Notes on screen resolution

If the GUI does not appear completely in your screen, check your screen resolution (at least 900 pixels in the vertical direction) and your percent of text display (up to **125%**) so that the program window entirely appears. (we'll solve that soon!).

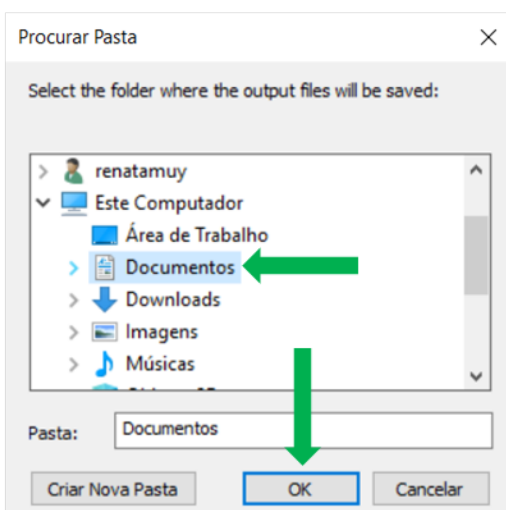
# 5. Running LSMetrics through GUI

Here we briefly explain how to set some important parameters used in LSmetrics. Again, make sure your input raster is using a metric coordinate reference system (CRS) by selecting the right EPSG. If your raster maps are not in a metric system EPSG, you will need to reproject them. For the reprojection, you need to know which is the SRC of your maps with projection different from your location. To do this, take a look at v.import.

Select the input map(s) and after you choose all the parameters that fit your analysis, you can click on "START CALCULATIONS". **Note that when choosing `Export`, the output of calculations (output raster maps and statistics text files) will be exported to a folder chosen by the user.**




Then, select the output directory where all the rasters with the chosen metrics will be saved, as follows.



## Single map or multiple maps?

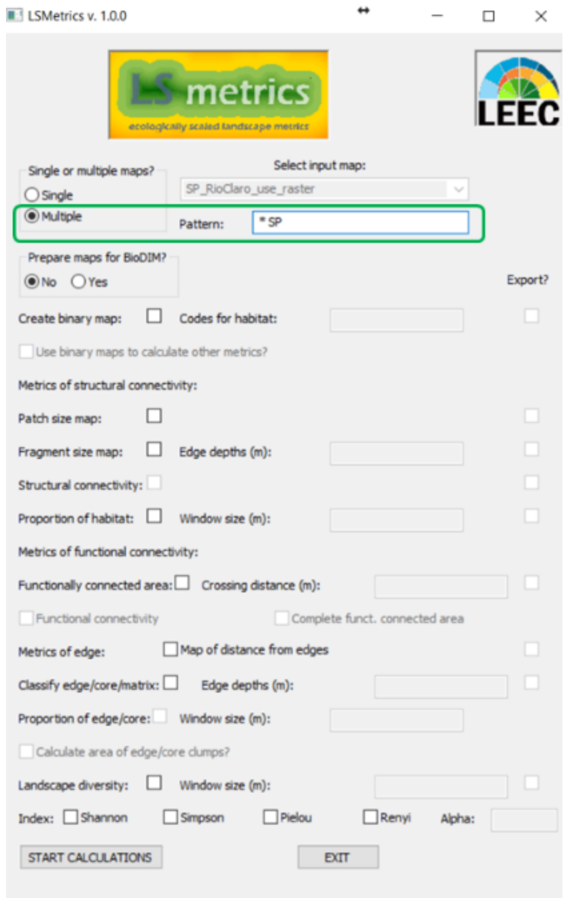
LSMetrics may calculate metrics for a single or multiple maps at once. If you select "single", choose the map name in the list of maps (only maps in you current GRASS GIS mapset will appear) and select the metrics you wish to calculate.

Alternatively, you can run the selected metrics for multiple input maps in a single run, if they share some part of their names, using regular expressions. For instance, **let's say** you have the following three maps within your current mapset:

- APA_Rio_Sao_Joao_tif
- SP_RioClaro_North_tif
- SP_RioClaro_South

In this case, as some of the maps have sequences of characters in common (e.g., "tif" between the first two, and "SP_RioClaro" between the last two maps), you can choose which ones to use as LSMetrics input using that. For running the calculations for a sequence of rasters with a string common pattern in raster file name, you must use the symbol "`*`" (asterisk). For example, in the white box of LSmetrics (`Pattern`):

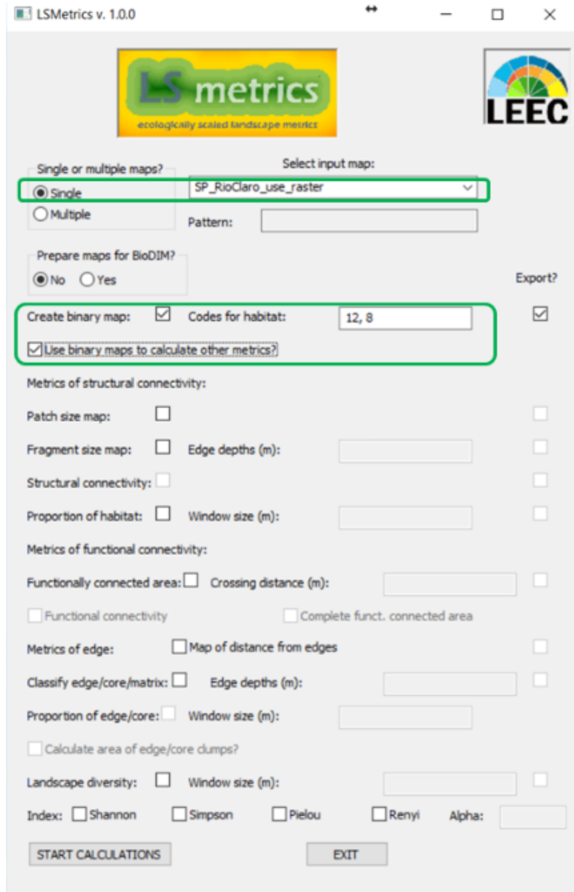
- if the file names' common pattern is all that starts with "SP", put: `SP*` (see the image below);
- if you want to select all maps that contain "forest" in any part of the file name, put: `*forest*`;
- if you want all maps that end with "SP", type: `*SP` ;
- if you want to use all maps loaded in you current mapset as input, type: `*`.




## Creating binary maps

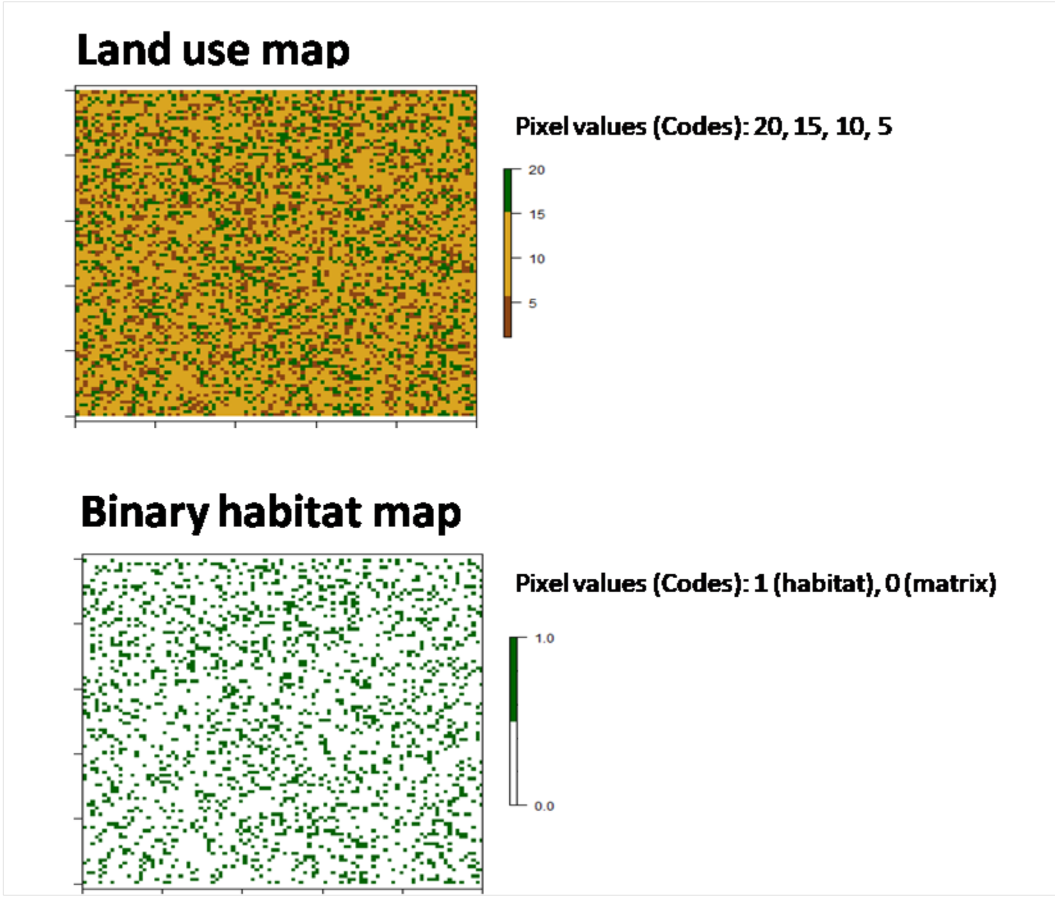
Most indices caluculated by LSMetrics are currently based on binary class (habitat/non-habitat) raster maps as input. If your map(s) are already classified this way (binary), you can use them straightforwardly as input to calculate metrics. If not, first you should select `Create binary map` and tell the app which pixel unique codes correspond to habitat. Then, you can use the resulting map as input to calculate landscape metrics by clicking in `Use binary maps to calculate other metrics`.

PS: If you select the option `Use binary maps to calculate other metrics`, you can already choose the metrics you want to calculate and generate all of them in the same run. If, instead, you create the binary map alone, you should then select the resulting map to then calculate the other metrics (in this case, unselect this option).

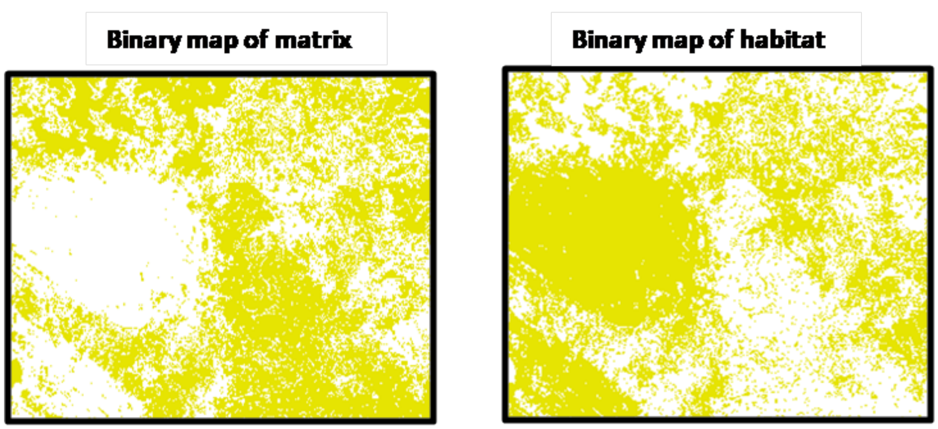



Binary maps are useful to investigate classes of interest and their metrics. See examples below:

This is an example of land use map that was transformed in a binary habitat map from specific codes for habitat classes (values from 15 to 20).



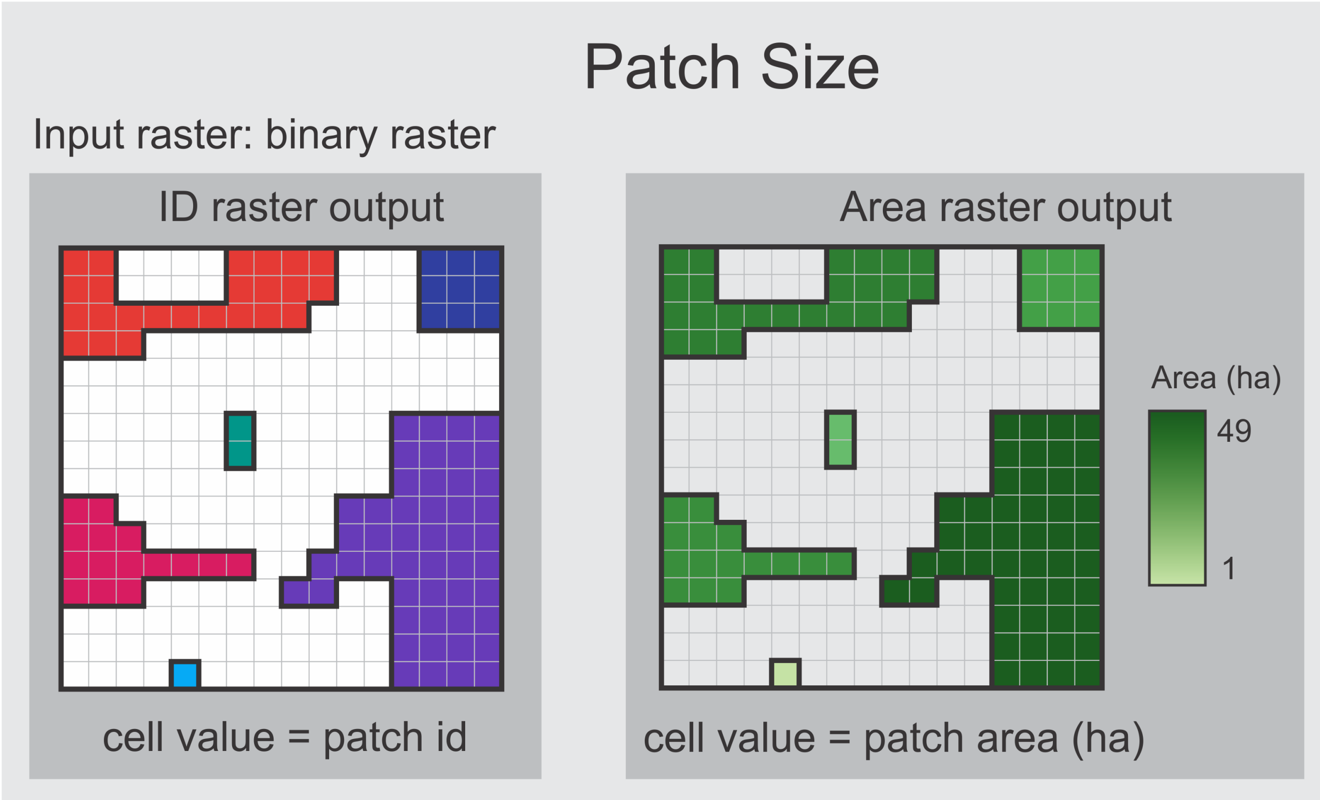

Below you may find another classic representation of binary maps. In the first, “1” (yellow) values represent the matrix, on the second the “1” (yellow) values represent the habitat areas.




## Structural connectivity metrics

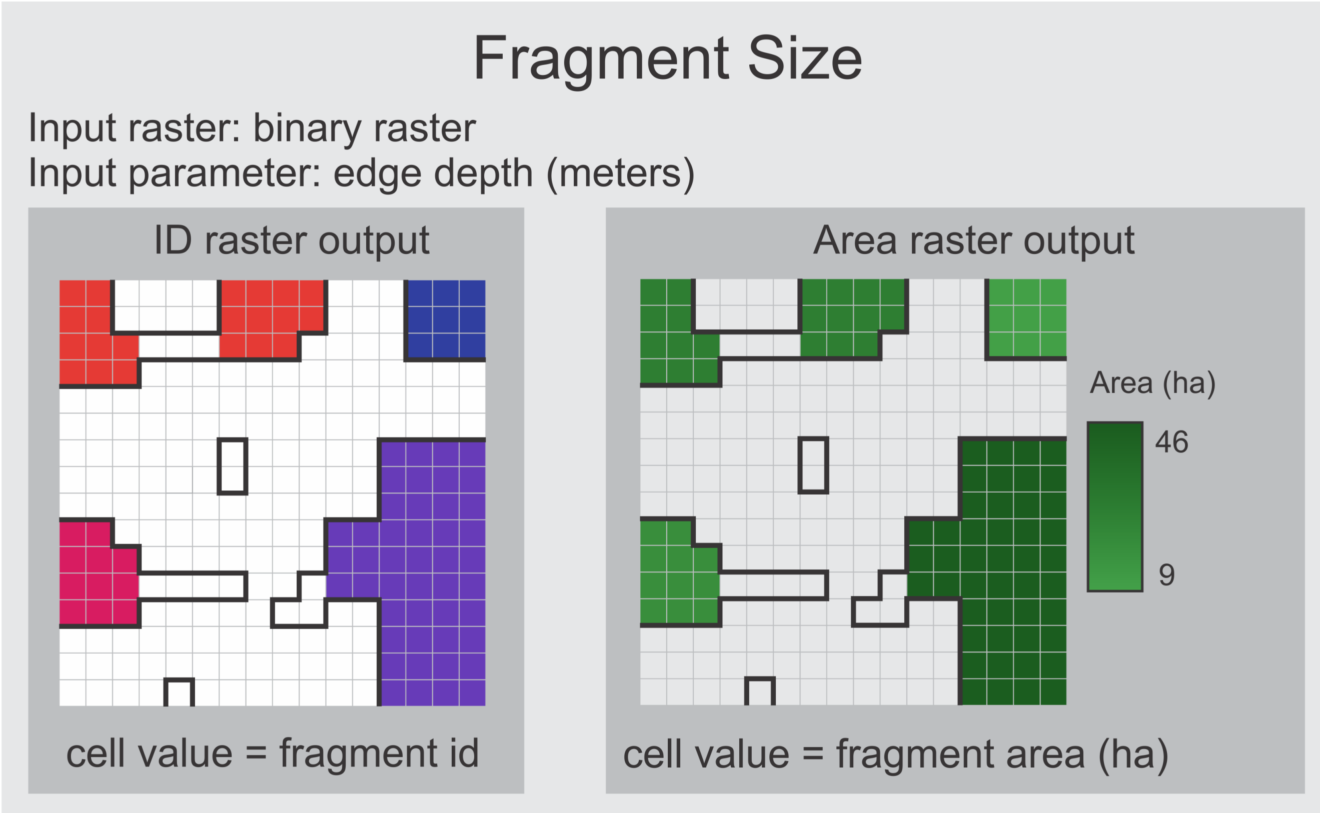
### Patch size

Summed area (in hectares) of one or more fragments connected by structural corridors (including the area of the corridors connecting them); if there are no fragments connected to the fragment in question, the value of this metric is equal to “fragment size” (see below). In the illustration map below, each cell has 100 m side.



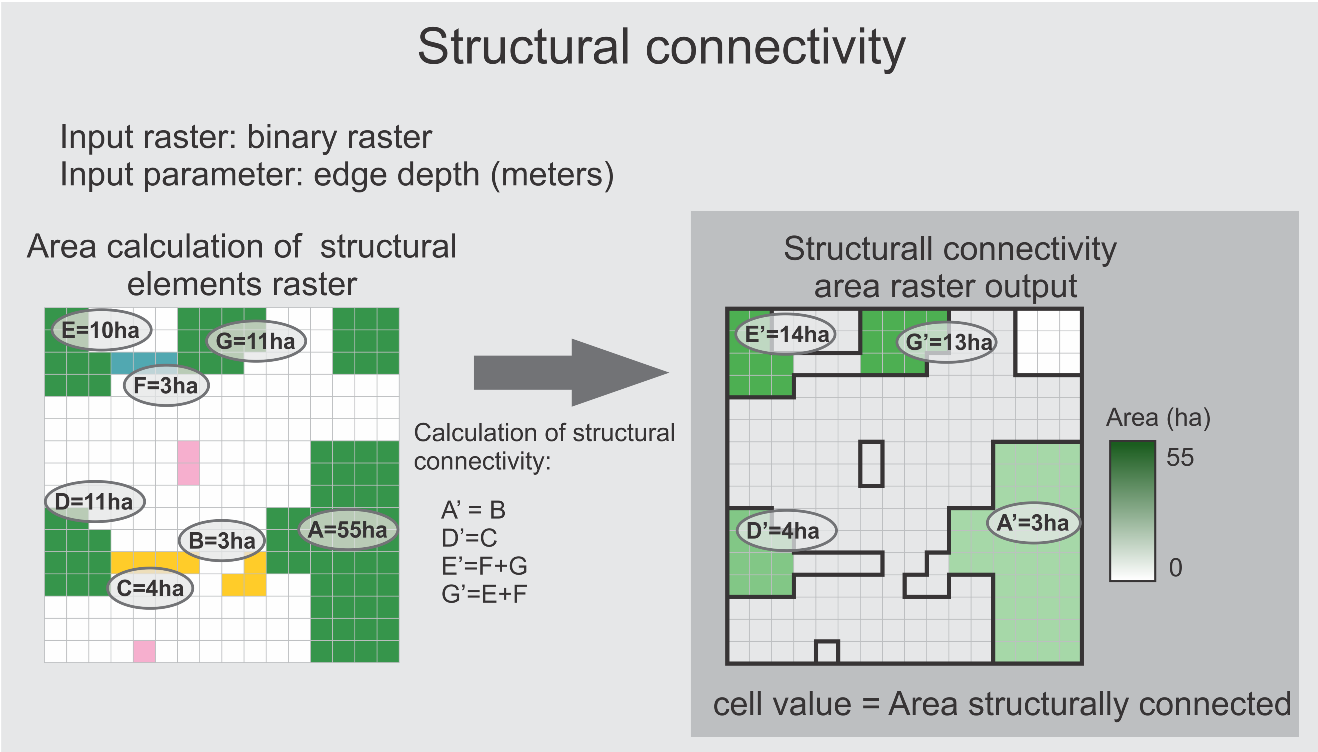

**Fragment size**

Area (in hectares) of any structurally isolated fragment, of any size, or the area of a fragment connected to another fragment by a structural corridor. Structural corridors are identified as parts of fragments with a width smaller than two times the specified edge depth.
The definition of what is a fragment then depends on the edge depth parameter chosen by the user.
In the illustration map below, each cell has 100 m side and the edge depth was chosen as 50 m.




**Structural connectivity**

Area of habitat that is structurally (contiguously) connected to a fragment. In practice, it is the difference (in hectares) between the Patch size and the Fragment size metrics; when a patch has only one fragment, its structural connectivity will be zero.
The definition of this metric depends on what is a fragment and a structural corridor, so it also depends on the edge depth parameter chosen by the user.
In the illustration map below, each cell has 100 m side and the edge depth was chosen as 50 m.




**Proportion of habitat**

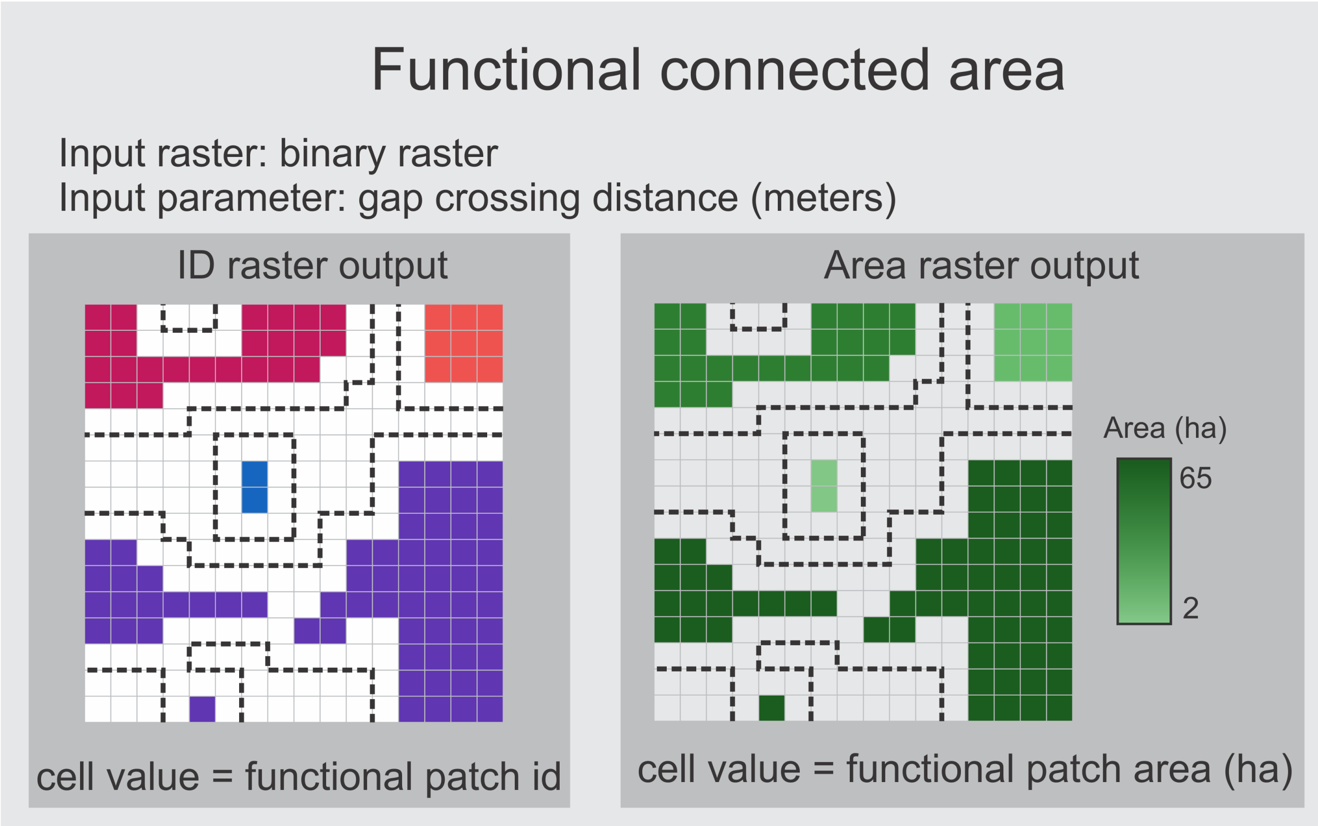
Proportion of the target landscape class (habitat or another class(es) of interest, the one represented by the value 1 in the input binary map) within a given window around each map cell.
It depends on the size of the window chosen by the user.

**How to choose the window size?** The user must decide which window size will be used to calculate metrics. It corresponds to search extent from each pixel where the calculations will be applied. It can be based on the extent of landscape perception of the target species, process or groups analyzed. It is given in meters.

## Functional connectivity metrics

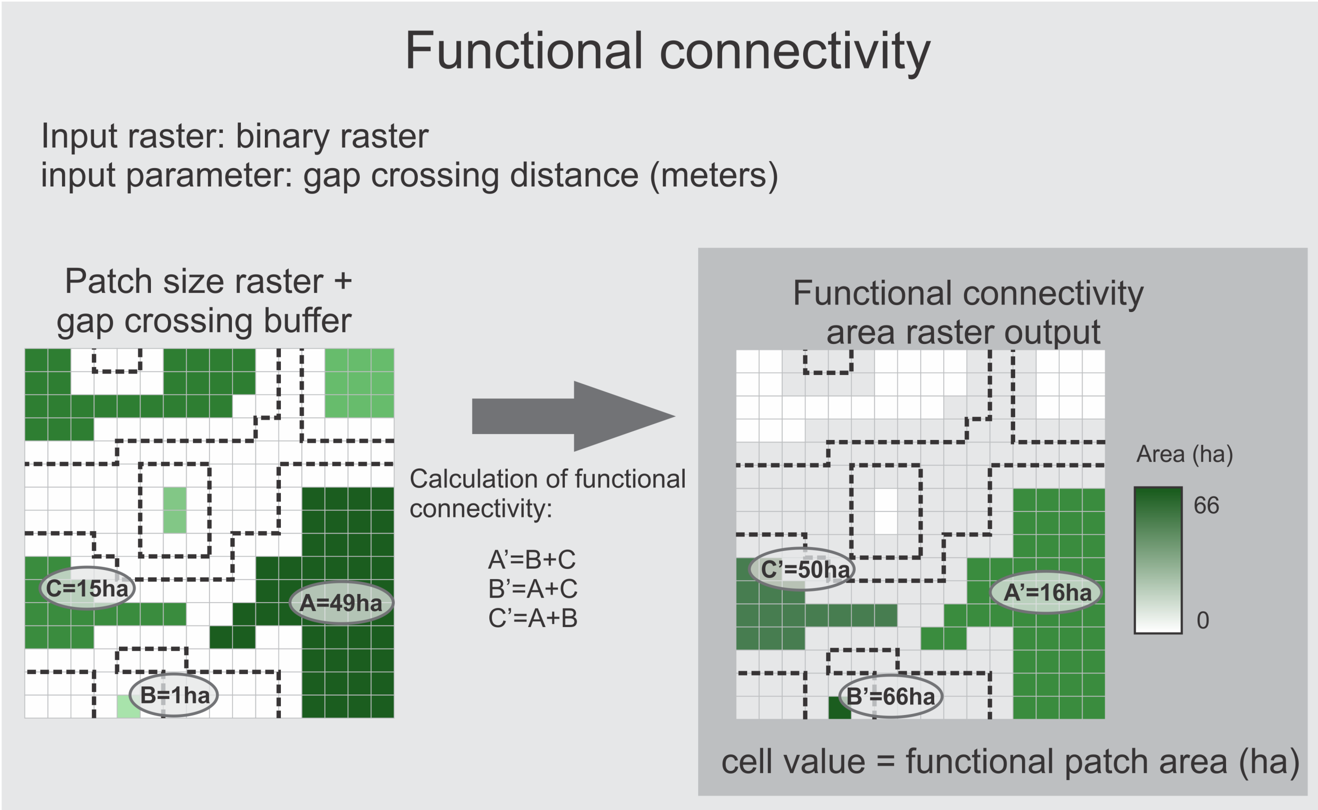
### Functionally connected area

Represents the total amount of habitat area (in hectares) functionally available for an organism located in a patch, given a gap crossing capability of the organism. This is calculated by generating a buffer of size equal to half of the gap crossing capacity of the organism, around all patches, grouping all habitat patches close enough in the same clusters of patches, and summing up the area of all the grouped patches.



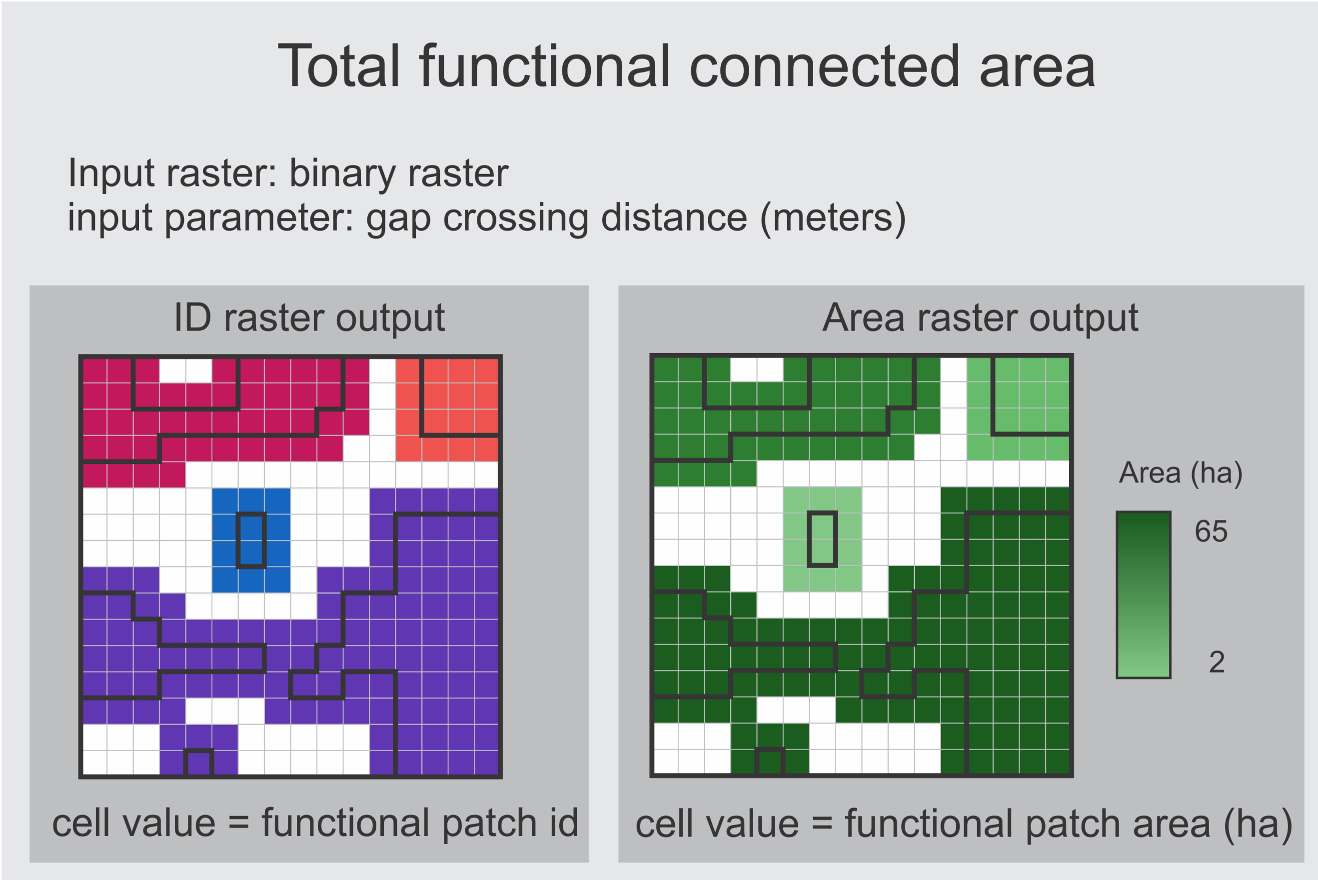

**Functional connectivity**

The amount of area functionally connected to a habitat patch, given a gap crossing capability of a species or group of interest. First, a buffer of interest equal to half of the width (in meters) that a species is able to cross is created. Then, the habitat patches that are close enough ara grouped within the same buffer. We then sum the area (ha) of all this grouped patches. We refer to these groups of nearby habitat patches as habitat **clumps**. Finally, to obtain the strict functional connectivity, we calculate the difference between each clump size (ha) and patch size (ha);




**Complete functional connected area**

The total amount of habitat functionally available for any species, given a gap crossing capability. We do all the steps for the "Functional connectivity" calculation, but do not subtract the patch size from the clump size.




### Metrics based on the delimitation of edges

See the charts and brief explanation of each metric below:

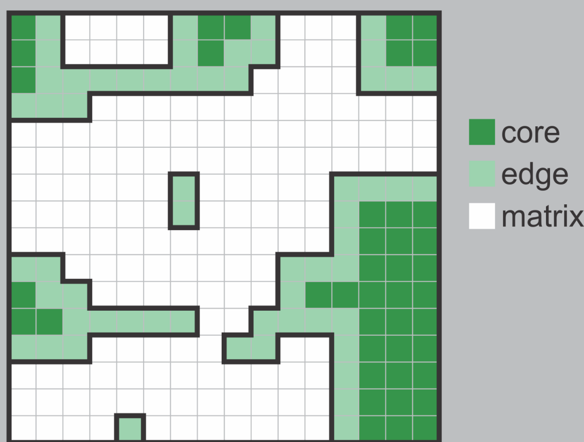
#### Classification of landscape structural elements

These metrics are classifications of landscape in three or more structural elements. The simple version of this metris divides elements in three categories: core, edge and matrix. The more complex version divides elements in: core, edge, corridor, branch, stepping stone and matrix (will be available soon). See below both versions:

# Classification of landscape structural elements (simplified version)

Input raster: binary raster
Input parameter: edge depth (meters)

structural elements raster output

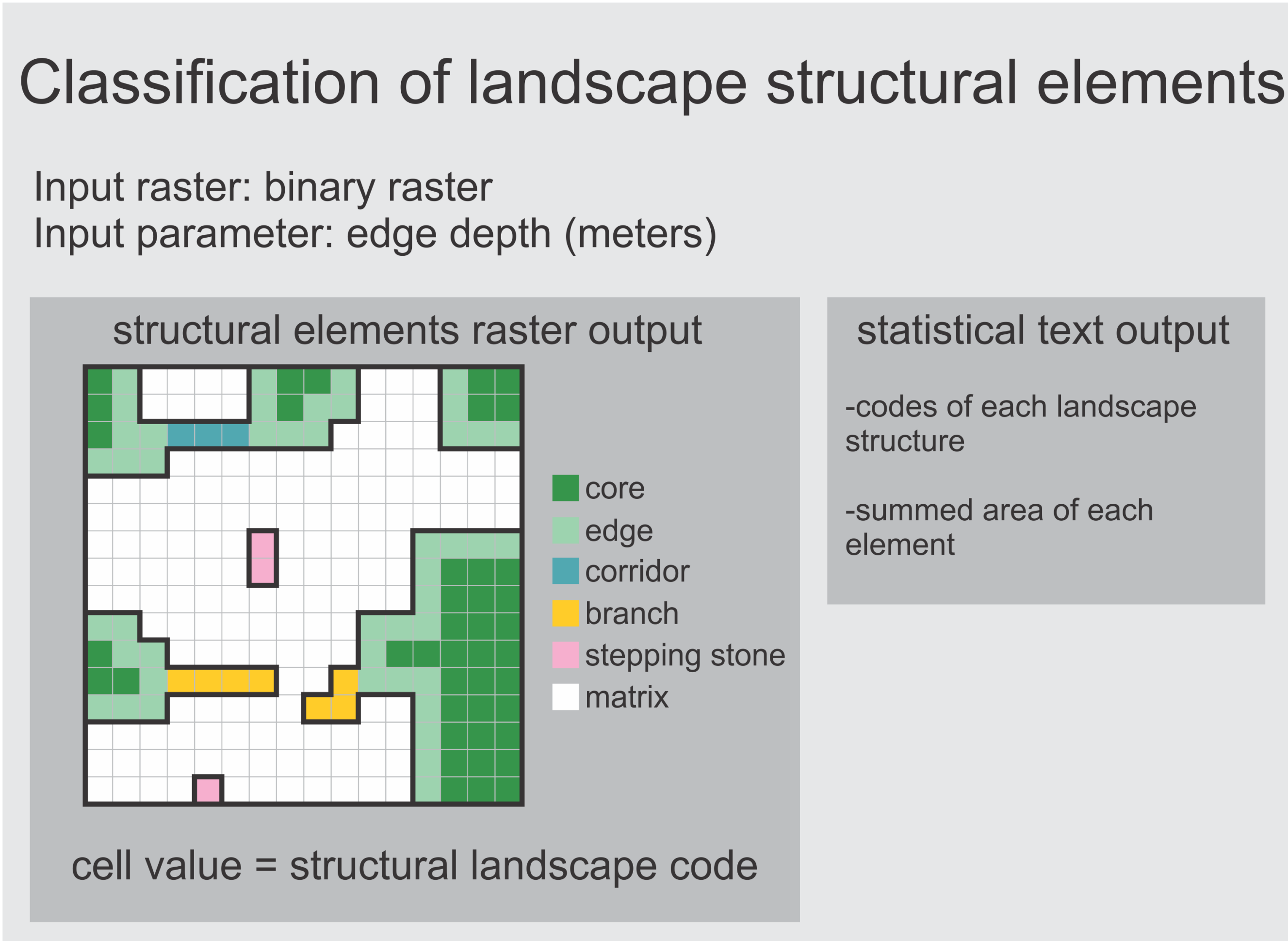



cell value = structural landscape code

statistical text output

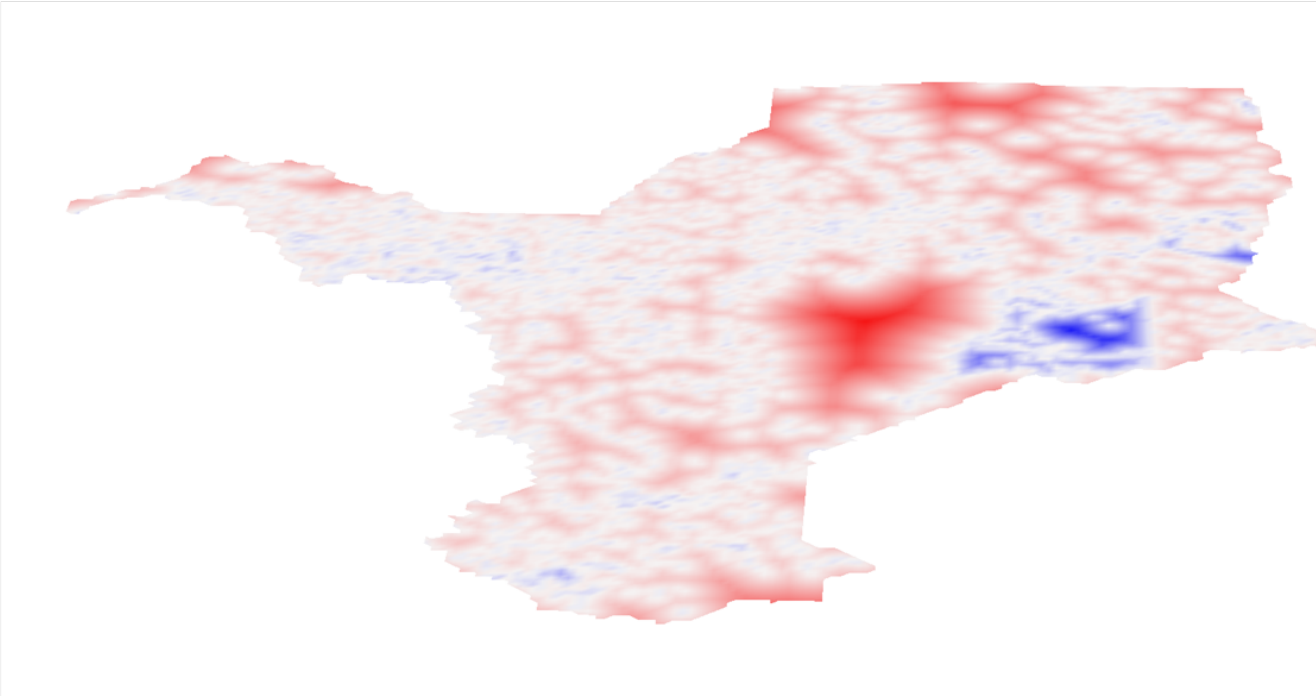
-codes of each landscape structure

-summed area of each element




**Map of distance from edges**

The output value of each pixel will represent the distance of this pixel to the nearest edge.



**Classify edge/core/matrix**

Considering a specified Edge depth, this metric classifies the pixels as belonging to three categories: edge, core and matrix. How species really use habitat area would be represented by the core area, since many species have limitations on using edges or less quality areas than their preferred habitat.

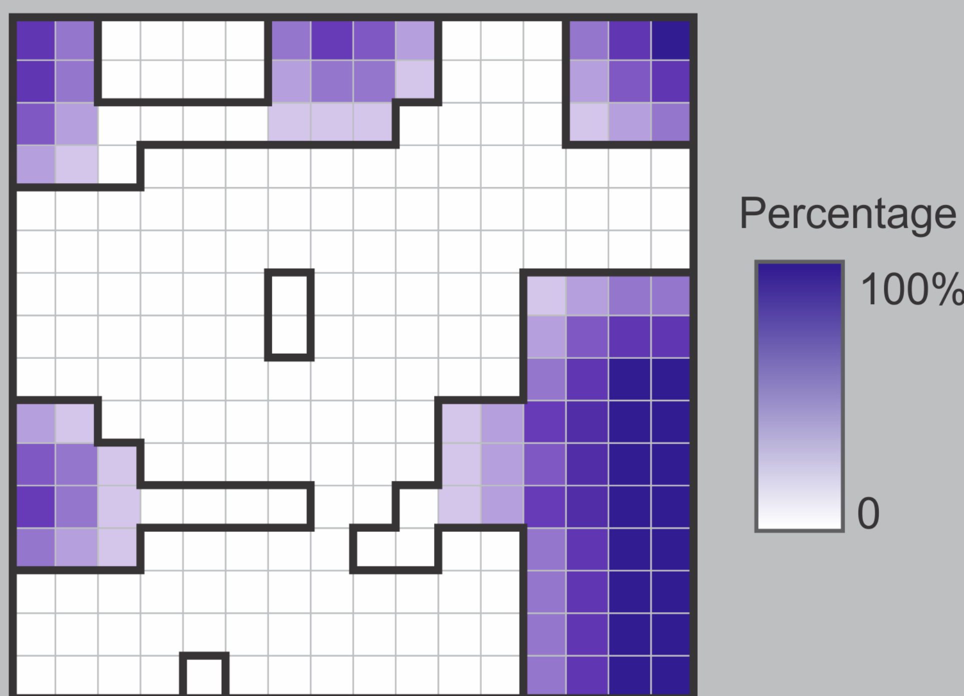
**Proportion of edge/core**

Percentage of habitat and edge within a search radius (Window size parameter), given a specified edge depth (Edge depths parameter).

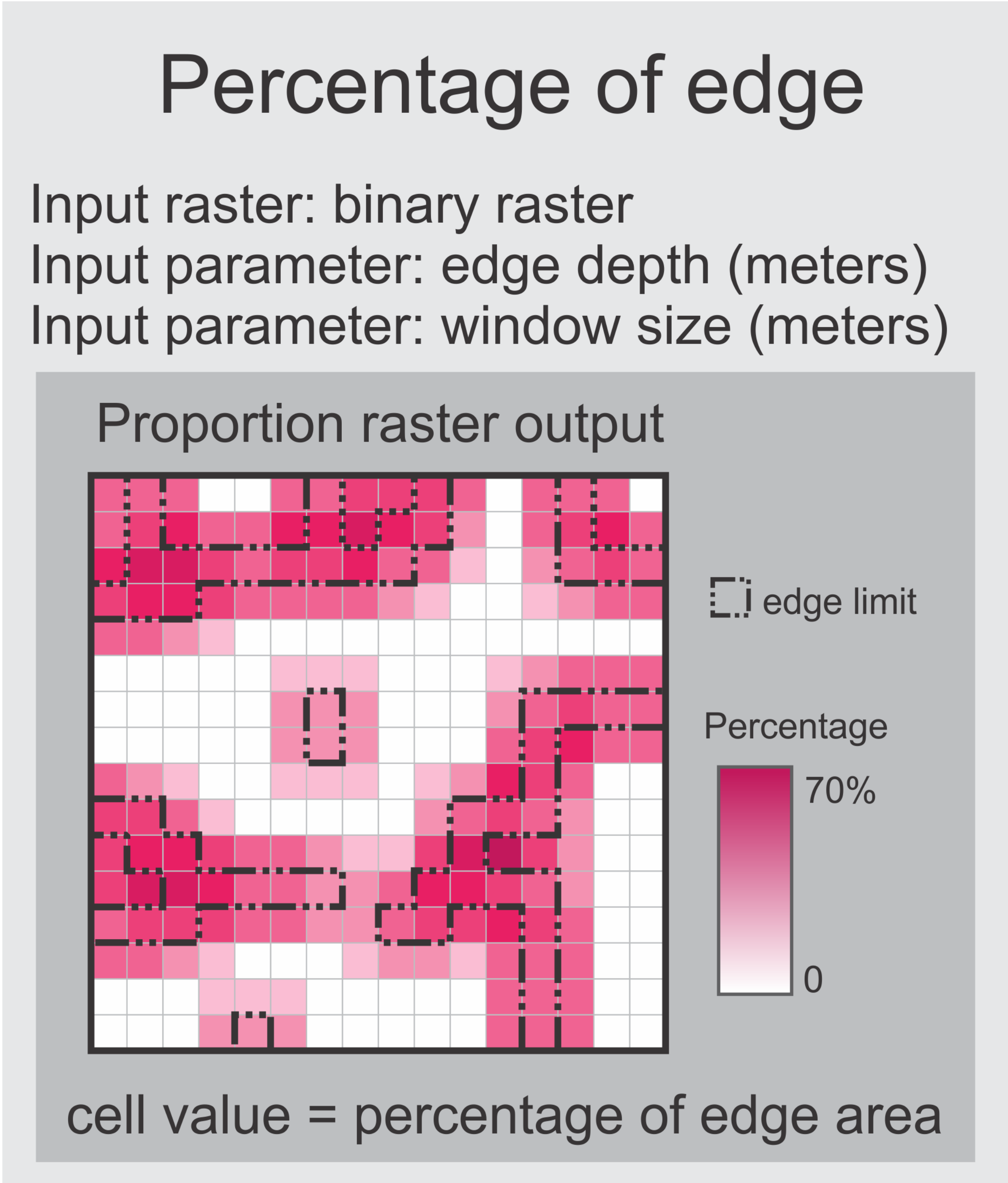
# Percentage of core

Input raster: binary raster
Input parameter: edge depth (meters)
Input parameter: window size (meters)

Proportion raster output




cell value = percentage of core area

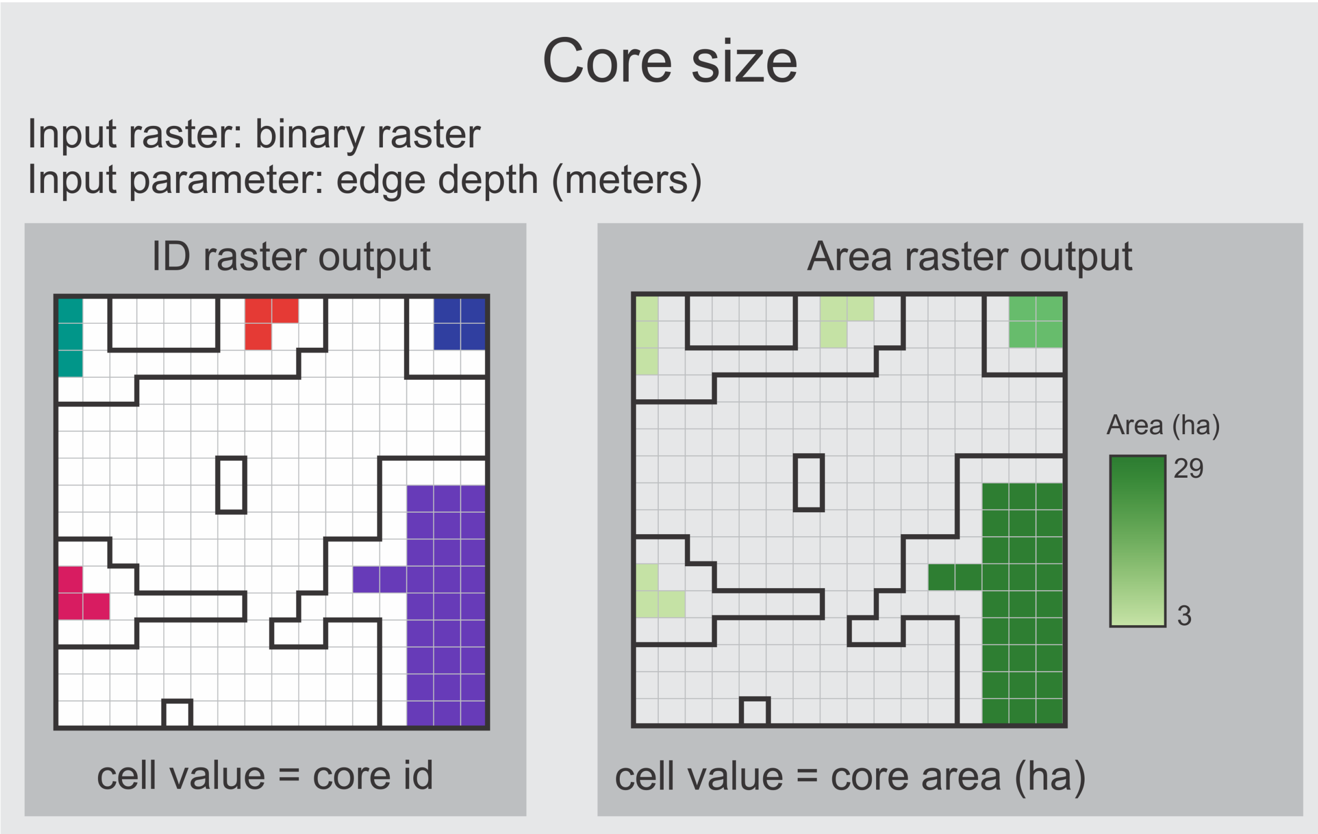


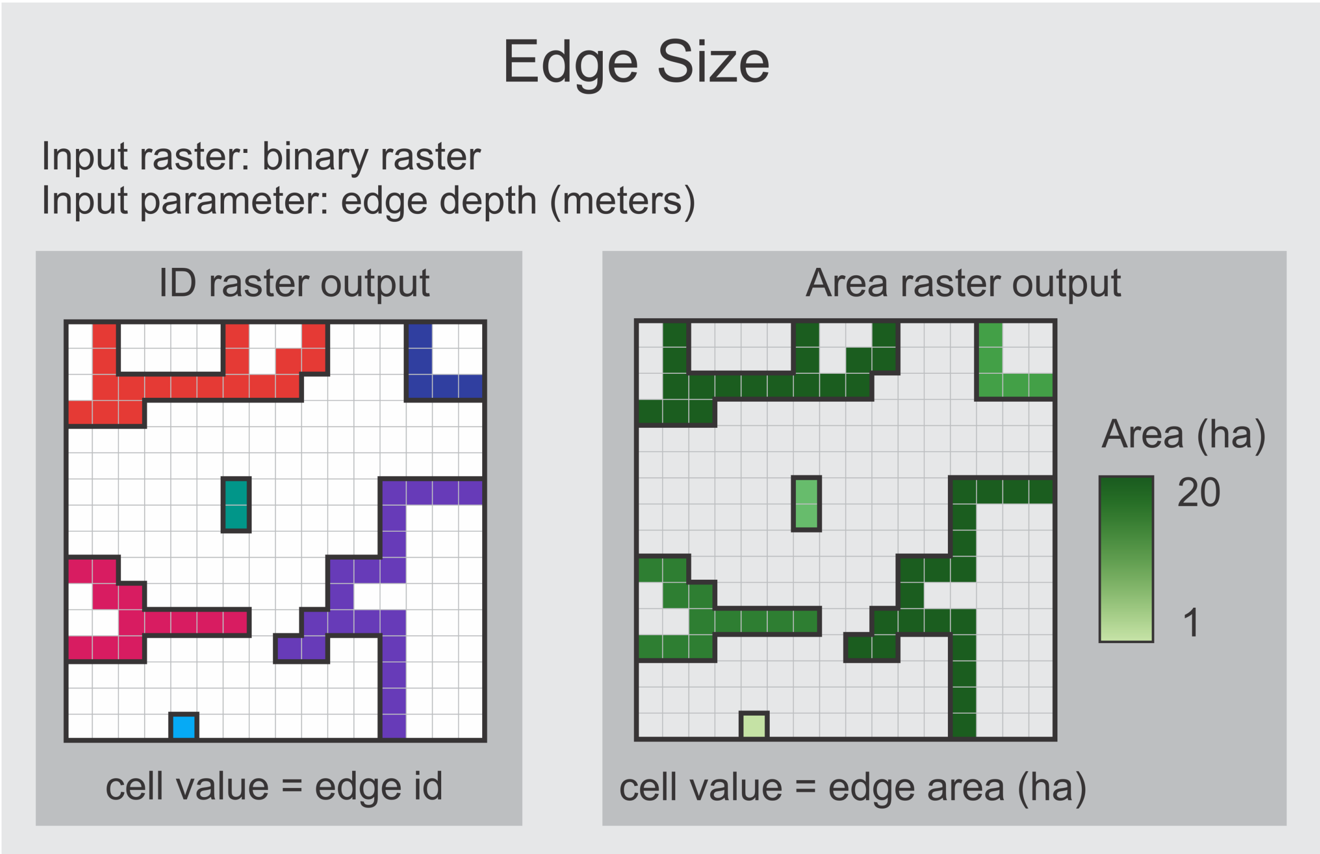

**Edge depths** This parameter is a proxy for investigating the impact of edge influence on the different pixels. It estimates the distance of edge influence and should be carefully thought a priori by the user, since it will be used to calculate the output metrics. It is set in meters, and as larger the value, larger the edge influence and extent through landscape.

**Calculate area of edge/core clumps?**

(Achei difícil reduzir essa explicação)

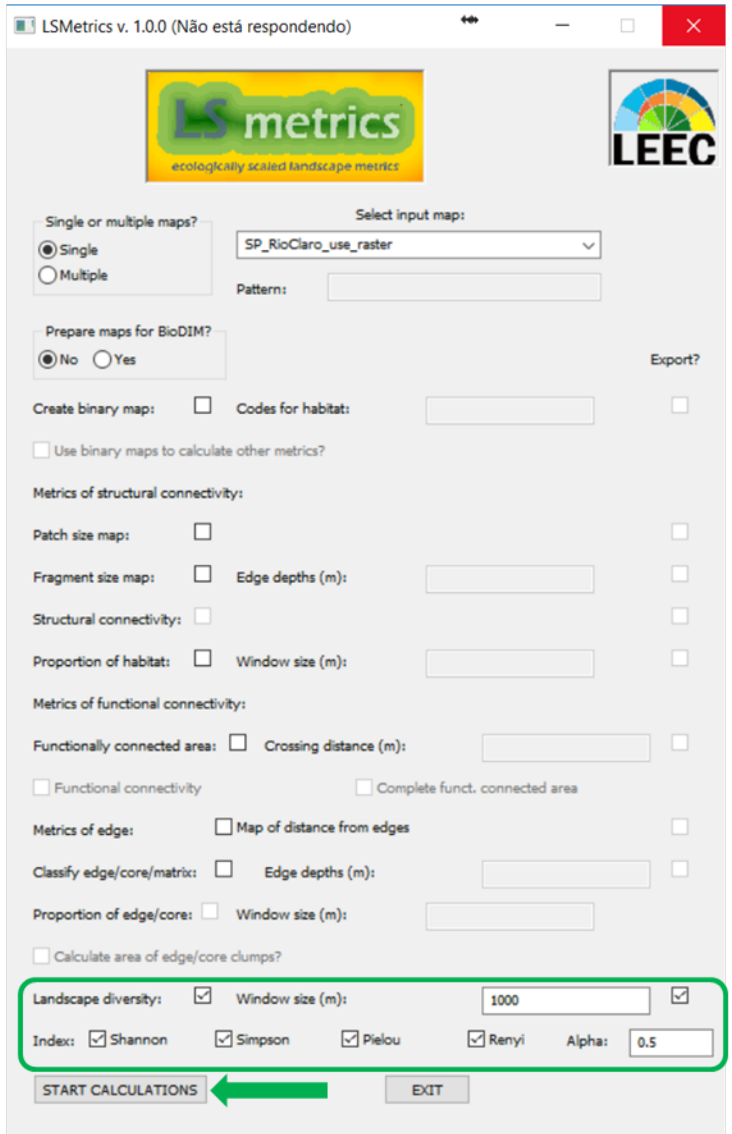
Given edge and core areas were identified, contiguous cells are clumped into edge and core clumps (i.e., cells of edge and core that pertain to the same patch), and their areas are calculated (in hectares). Two maps are created for each one (edge/core) for each landscape: an edge (core) clump ID (edge or core PID) map, which sets a distinct integer number to each clump of edge (core) cells, and a clump size map, which sets the the area of those clumps; non-edge (non-core) cells are set to NULL in both.





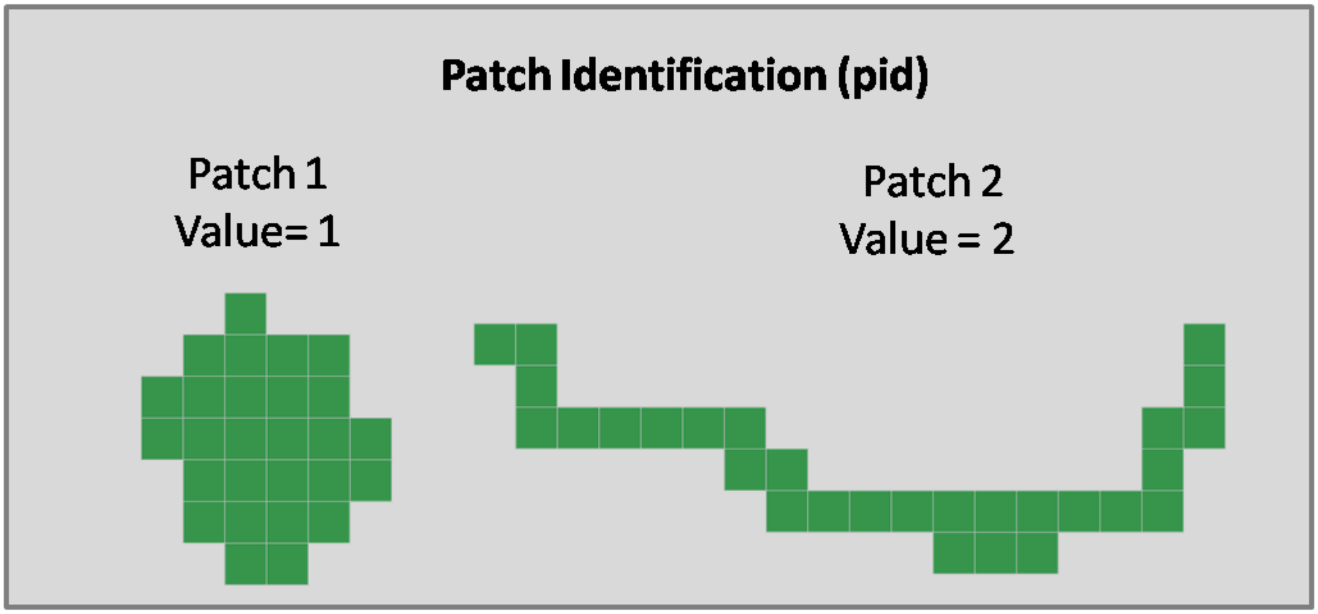

### Landscape diversity metrics

Diversity indexes require multi-class raster maps. In order to calculate landscape diversity, the user must set which land use class codes are relevant for the calculation. In our example data, we have the land use for Rio Claro municipality. LSMetrics run the landscape diversity measures with **all classes**. We suggest always having more than three classes in your land use map to increase the variability in the landscape diversity measures. The user must choose the window size that will be used to calculate the diversity in landscape. In the example, we set as 1000 m and click to calculate all diversity measures. This means that the search radius will be 1000 meters.



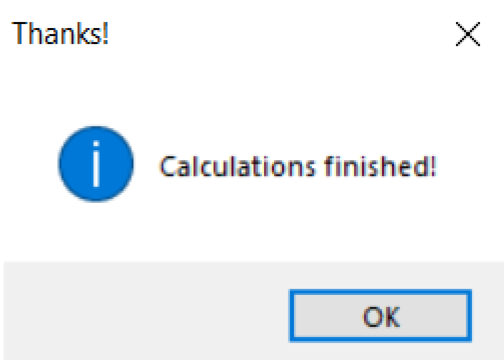

## Maps identifying patches, fragments, and functional patches

PIDs are Patch or Clump identification codes for the respective selected metrics. FIDs are identification codes for each clump generated by connectivity metrics.


Patch Identification (pid)
Patch 1
Value= 1
Patch 2
Value = 2

# 6. LSMetrics outputs

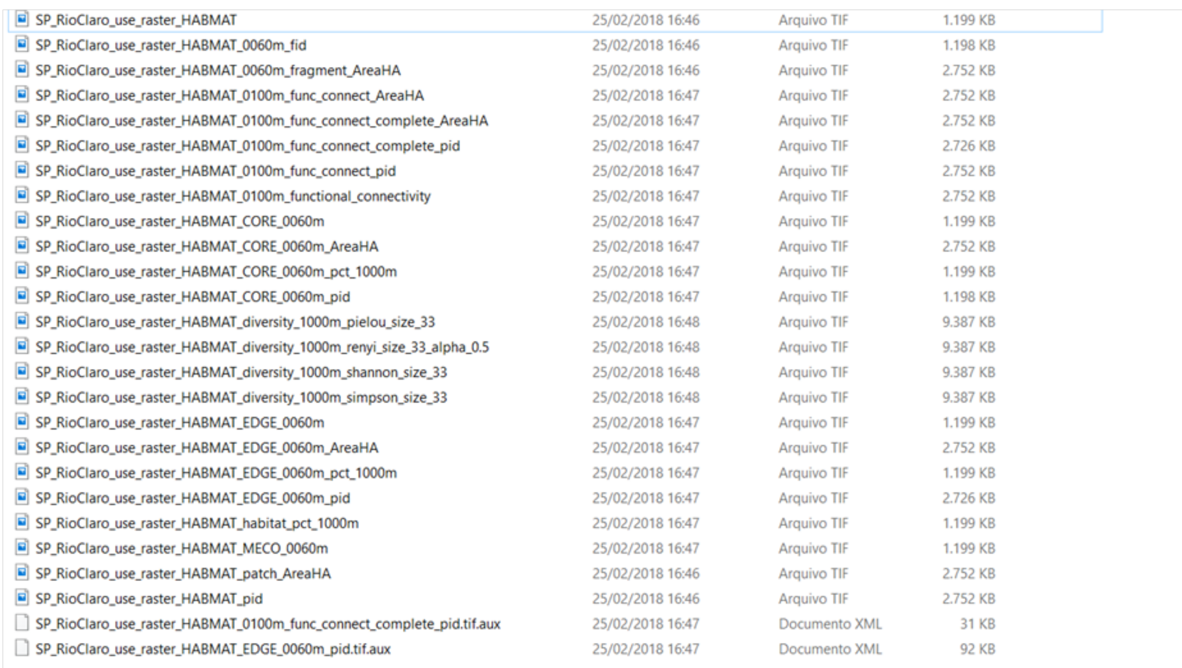
After LSMetrics runs, you will see in your screen the following message, with a sound:




By default, LSMetrics outputs are raster maps in ".tiff" format. We will use the palette "byr" from r.colors for coloring our diversity rasters. For practical reasons, we are using different points near the state forest region of FEENA to show their values for each metrics via Query results in GRASS GIS display. If you want to visualize the values of each metric for a same spatial point, you can use r.what in the command line of GRASS GIS:

```
r.what map=SP_RioClaro_use_raster_diversity_1000m_pielou_size_33@PERMANENT coordinates=241349.630457,75175
```

If you choose all metrics, you will see them in the output file you´ve chosen before.



| | | | |
|---|---|---|---|
| SP_RioClaro_use_raster_HABMAT | 25/02/2018 16:46 | Arquivo TIF | 1.199 KB |
| SP_RioClaro_use_raster_HABMAT_0060m_fid | 25/02/2018 16:46 | Arquivo TIF | 1.198 KB |
| SP_RioClaro_use_raster_HABMAT_0060m_fragment_AreaHA | 25/02/2018 16:46 | Arquivo TIF | 2.752 KB |
| SP_RioClaro_use_raster_HABMAT_0100m_func_connect_AreaHA | 25/02/2018 16:47 | Arquivo TIF | 2.752 KB |
| SP_RioClaro_use_raster_HABMAT_0100m_func_connect_complete_AreaHA | 25/02/2018 16:47 | Arquivo TIF | 2.752 KB |
| SP_RioClaro_use_raster_HABMAT_0100m_func_connect_complete_pid | 25/02/2018 16:47 | Arquivo TIF | 2.726 KB |
| SP_RioClaro_use_raster_HABMAT_0100m_func_connect_pid | 25/02/2018 16:47 | Arquivo TIF | 2.752 KB |
| SP_RioClaro_use_raster_HABMAT_0100m_functional_connectivity | 25/02/2018 16:47 | Arquivo TIF | 2.752 KB |
| SP_RioClaro_use_raster_HABMAT_CORE_0060m | 25/02/2018 16:47 | Arquivo TIF | 1.199 KB |
| SP_RioClaro_use_raster_HABMAT_CORE_0060m_AreaHA | 25/02/2018 16:47 | Arquivo TIF | 2.752 KB |
| SP_RioClaro_use_raster_HABMAT_CORE_0060m_pct_1000m | 25/02/2018 16:47 | Arquivo TIF | 1.199 KB |
| SP_RioClaro_use_raster_HABMAT_CORE_0060m_pid | 25/02/2018 16:47 | Arquivo TIF | 1.198 KB |
| SP_RioClaro_use_raster_HABMAT_diversity_1000m_pielou_size_33 | 25/02/2018 16:48 | Arquivo TIF | 9.387 KB |
| SP_RioClaro_use_raster_HABMAT_diversity_1000m_renyi_size_33_alpha_0.5 | 25/02/2018 16:48 | Arquivo TIF | 9.387 KB |
| SP_RioClaro_use_raster_HABMAT_diversity_1000m_shannon_size_33 | 25/02/2018 16:48 | Arquivo TIF | 9.387 KB |
| SP_RioClaro_use_raster_HABMAT_diversity_1000m_simpson_size_33 | 25/02/2018 16:48 | Arquivo TIF | 9.387 KB |
| SP_RioClaro_use_raster_HABMAT_EDGE_0060m | 25/02/2018 16:47 | Arquivo TIF | 1.199 KB |
| SP_RioClaro_use_raster_HABMAT_EDGE_0060m_AreaHA | 25/02/2018 16:47 | Arquivo TIF | 2.752 KB |
| SP_RioClaro_use_raster_HABMAT_EDGE_0060m_pct_1000m | 25/02/2018 16:47 | Arquivo TIF | 1.199 KB |
| SP_RioClaro_use_raster_HABMAT_EDGE_0060m_pid | 25/02/2018 16:47 | Arquivo TIF | 2.726 KB |
| SP_RioClaro_use_raster_HABMAT_habitat_pct_1000m | 25/02/2018 16:47 | Arquivo TIF | 1.199 KB |
| SP_RioClaro_use_raster_HABMAT_MECO_0060m | 25/02/2018 16:47 | Arquivo TIF | 1.199 KB |
| SP_RioClaro_use_raster_HABMAT_patch_AreaHA | 25/02/2018 16:46 | Arquivo TIF | 2.752 KB |
| SP_RioClaro_use_raster_HABMAT_pid | 25/02/2018 16:46 | Arquivo TIF | 2.752 KB |
| SP_RioClaro_use_raster_HABMAT_0100m_func_connect_complete_pid.tif.aux | 25/02/2018 16:47 | Documento XML | 31 KB |
| SP_RioClaro_use_raster_HABMAT_EDGE_0060m_pid.tif.aux | 25/02/2018 16:47 | Documento XML | 92 KB |

Next, we are going to briefly present a layout of each landscape metrics outputs, together with their sufix file names and query results in printscreens of GRASS GIS displays.

Take a look on the 27 output suffixes of file names for our example data:

| Metric | Sufix of file name |
|---|---|
| _HABMAT | Binary raster of habitat and matrix |
| _HABMAT_0060m_fid | FID of structural elements map |

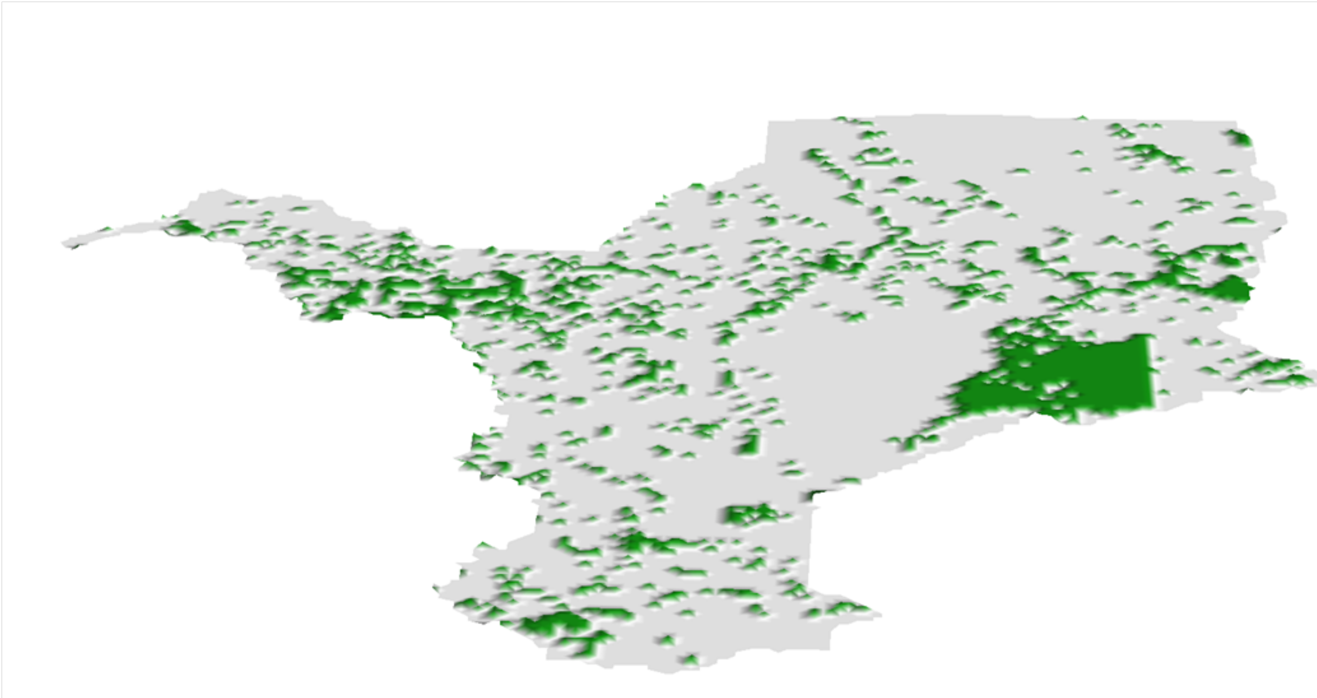
| Metric | Sufix of file name |
|---|---|
| _HABMAT_0060m_fragment_AreaHA | Fragment area in hectars |
| _HABMAT_0100m_func_connect_AreaHA | Functional connected area in hectars using a 100 crossing distance |
| _HABMAT_0060m_structural_connectivity | Classification of structural elements with a 60 m edge depth |
| _HABMAT_0100m_func_connect_AreaHA | Functional connected area in hectars using a 100 crossing distance |
| _HABMAT_0100m_func_connect_complete_AreaHA | Complete functionally connected area in hectars using a 100 window |
| _HABMAT_0100m_func_connect_complete_pid | PIDs of Complete functionally connected area in hectars using a 100 |
| _HABMAT_0100m_func_connect_pid | PIDs of clumps for functional connectivity using a 100 m crossing di |
| _HABMAT_0100m_functional_connectivity | Functional connectivity using a 100 m crossing distance |
| _HABMAT_EDGE_0060m | Edge area binary map using a 60 m edge depth |
| _HABMAT_EDGE_0060m_AreaHA | Edge area map using a 60 m edge depth |
| _HABMAT_EDGE_0060m_pct_1000m | Proportion of edge using a 60 m edge depth |
| _HABMAT_EDGE_0060m_pid | Identification of edge cumps using a 60 m edge depth |
| _HABMAT_EDGE_DIST | Map of distances from edge |
| _HABMAT_habitat_pct_1000m | Proportion of habitat using a 1000 m window size |
| _HABMAT_MECO_0060m | Structural elements (edge, core, matrix) |
| _HABMAT_patch_AreaHA | Patch area of the habitat matrix map |
| _HABMAT_pid | Patch identification of the habitat matrix map |
| _HABMAT_CORE_0060m | Core area binary map using a 60 m edge depth |
| _HABMAT_CORE_0060m_AreaHA | Core area in hectares |
| _HABMAT_0060m_pct_1000m | Proportion of habitat using a 1000 m window size and a 60 m edge |
| _HABMAT_CORE_0060m_pid | Patch identification of core areas using a 60 m edge depth value |
| _HABMAT_diversity_1000m_pielou_size_33 | Landscape diversity using Pielou´s algorithm and a 1000 m window |
| _HABMAT_diversity_1000m_shannon_size_33 | Landscape diversity using Shannon´s algorithm and a 1000 m windo |
| *HABMAT*diversity_1000m_simpson_size_33 | Landscape diversity using Simpson´s algorithm and a 1000 m windo |
| *HABMAT*diversity_1000m_renyi_size_33_alpha_0.5 | Landscape diversity using Renyis´s algorithm and a 1000 m window |

## Metrics of structural connectivity

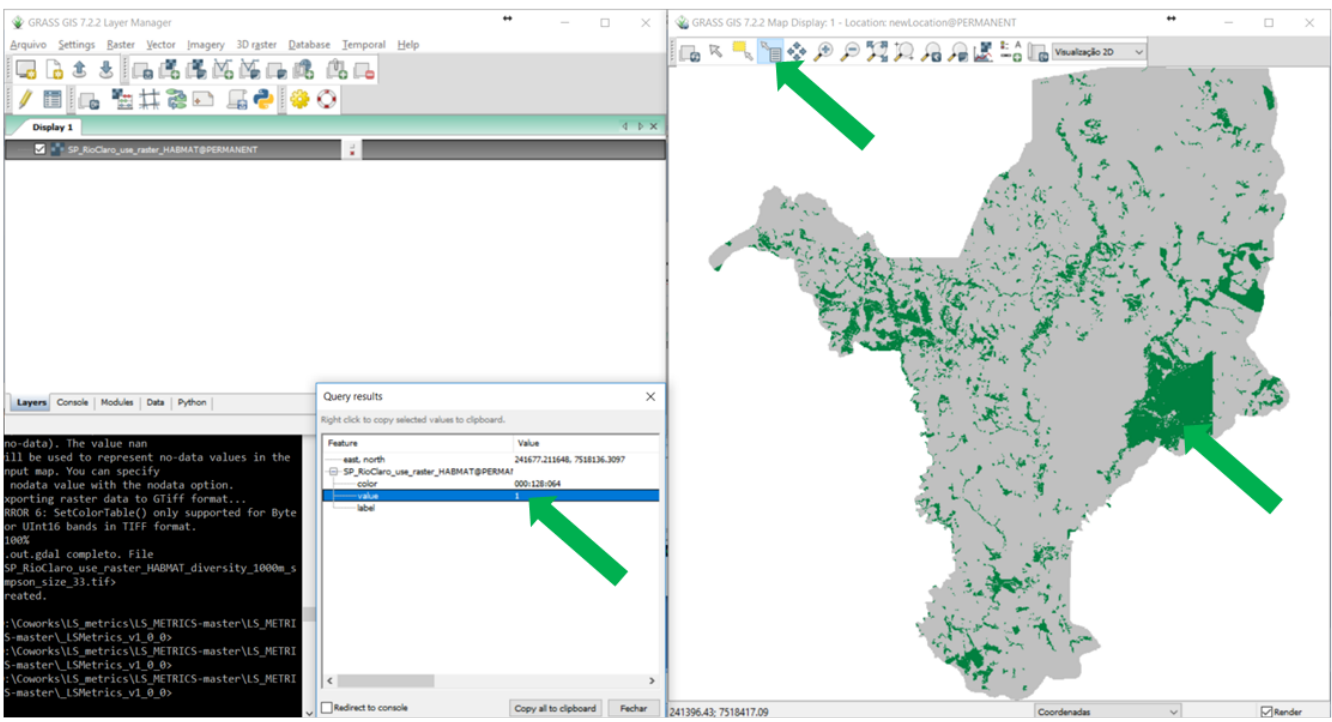
Here we will show the outputs of each metric and a print of their Query results in GRASS GIS display.

### Create Binary map

See this binary map of Rio Claro based on forest classes (not necessarily native forest).



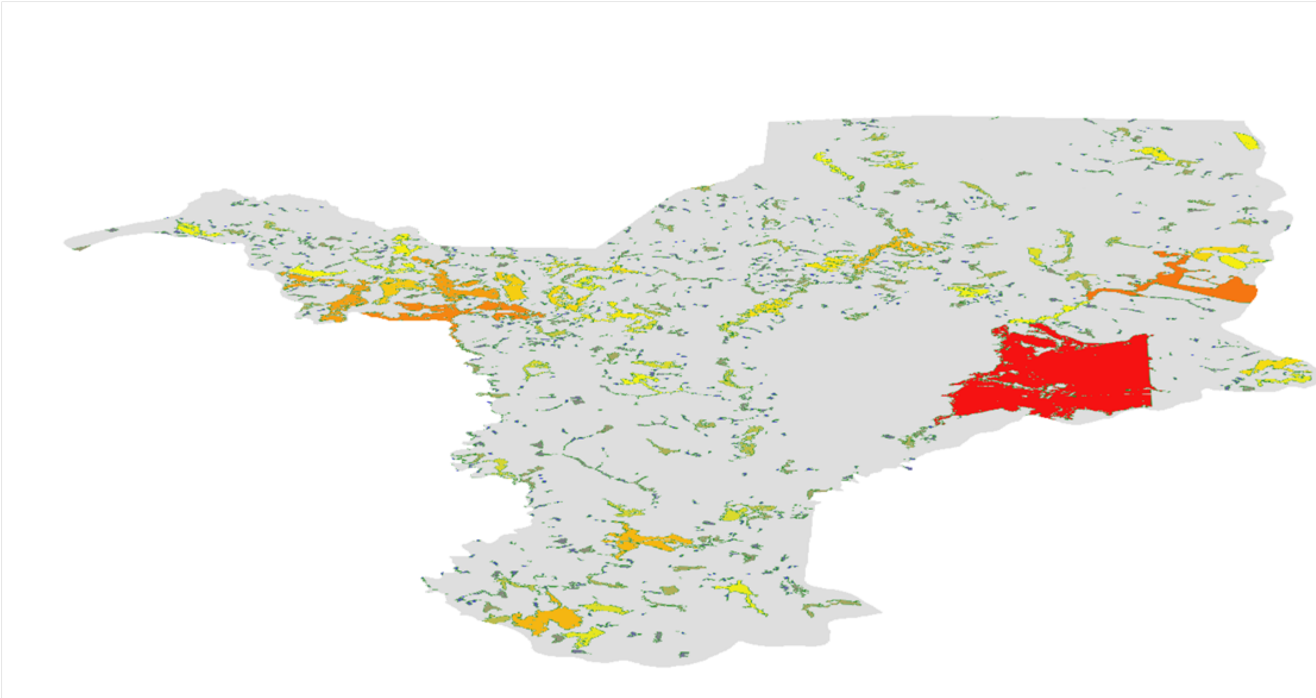
Query results are below:



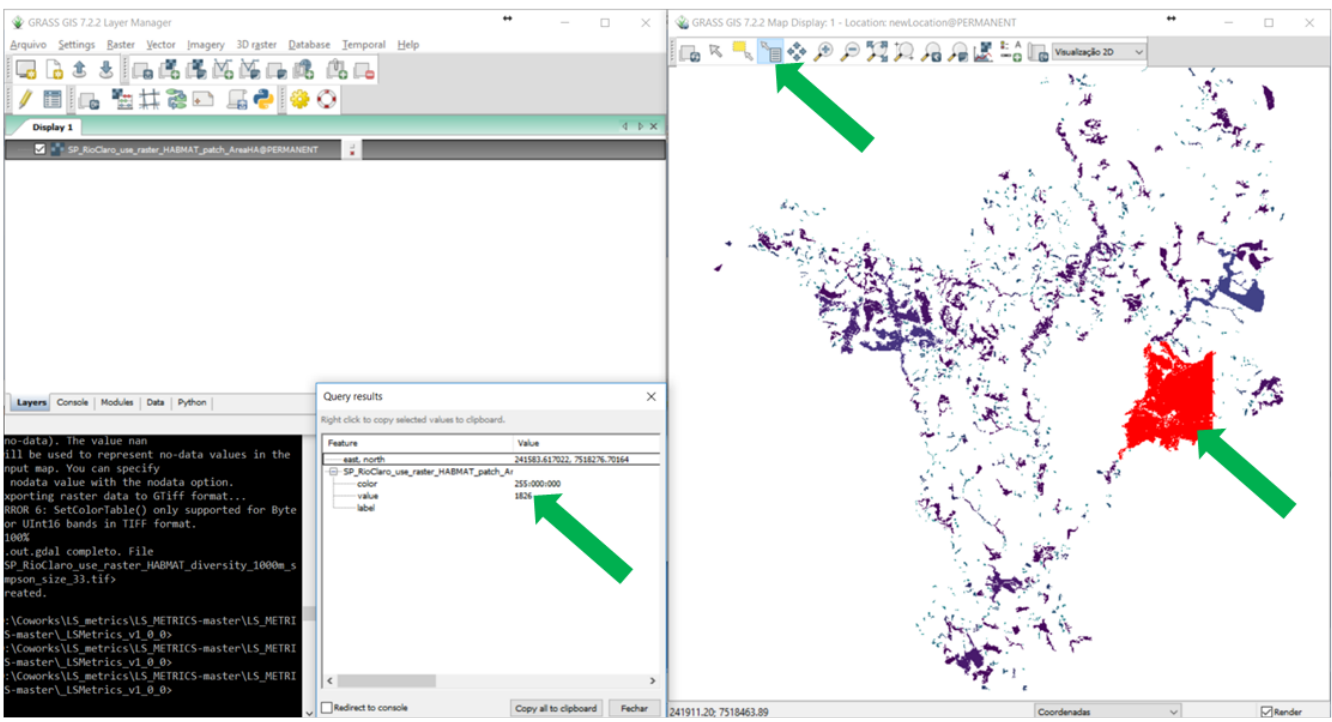

**Patch size map**

***patch*AreaHA**

Summed area of one or more fragments connected by structural corridors (including the area of the corridors connecting them); if there are no fragments connected to the fragment in question, the value of this metric is equal to "fragment size";



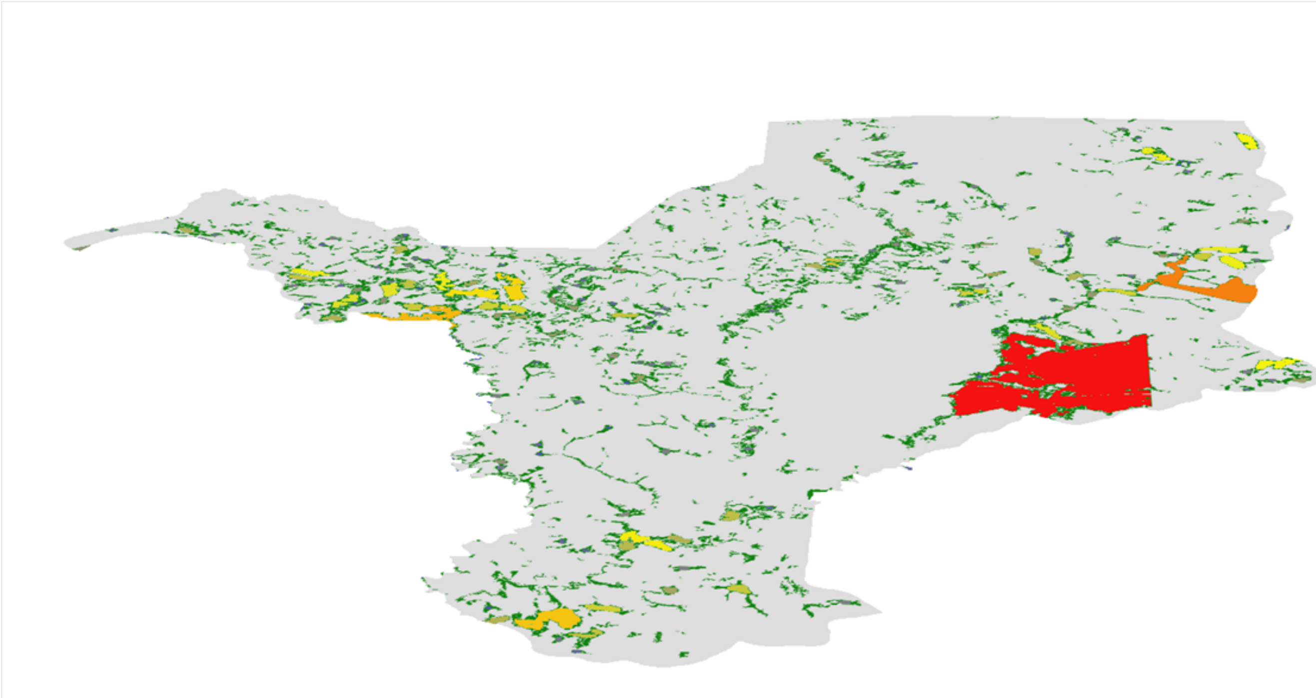
See query results below:



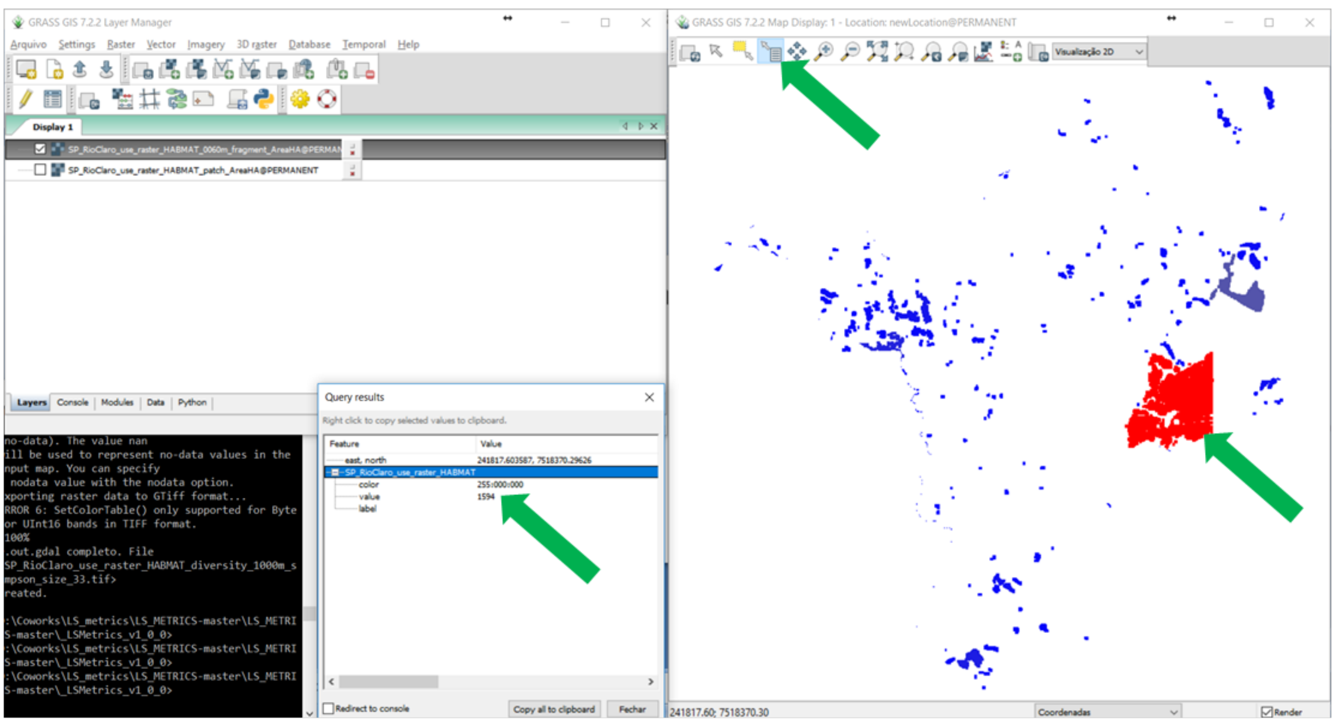

### Fragment size map

### *0060m*fragment_AreaHA

Area (given in ha) of any structurally isolated fragment, of any size, or the area of a fragment connected to another fragment by a structural corridor. Structural corridors are identified as parts of fragments with a width smaller than two times the specified edge depth;



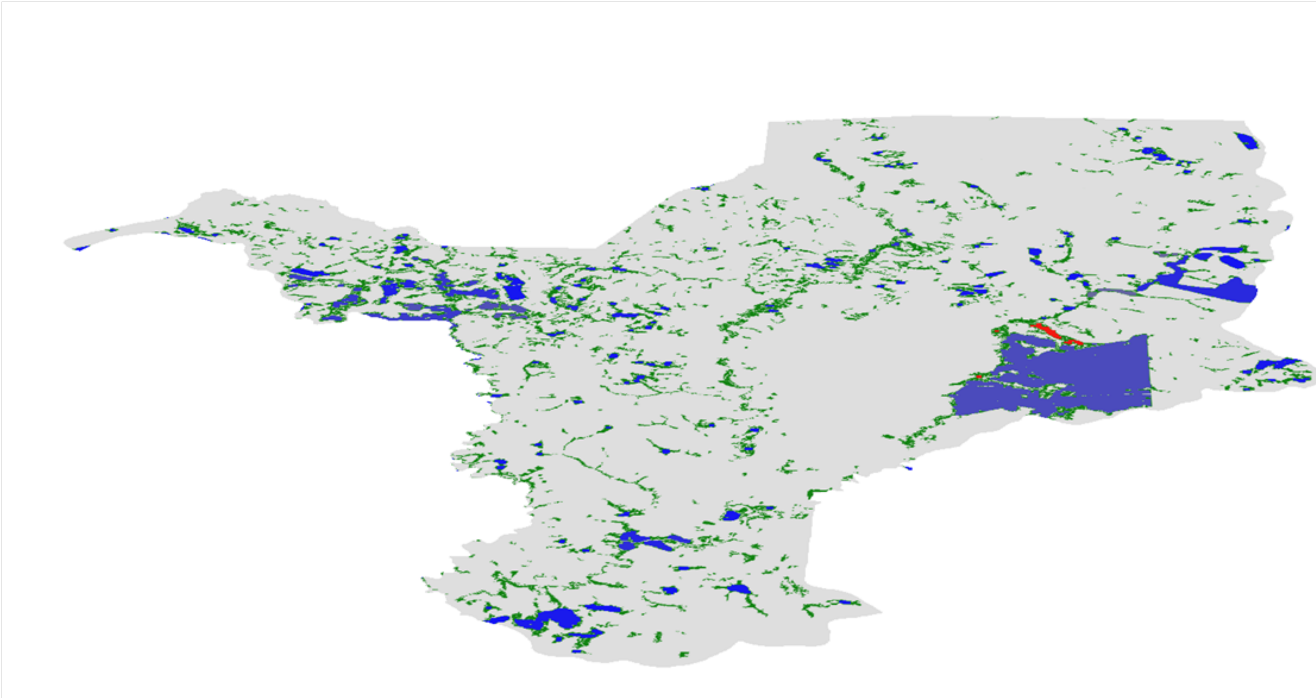
Query results are below:



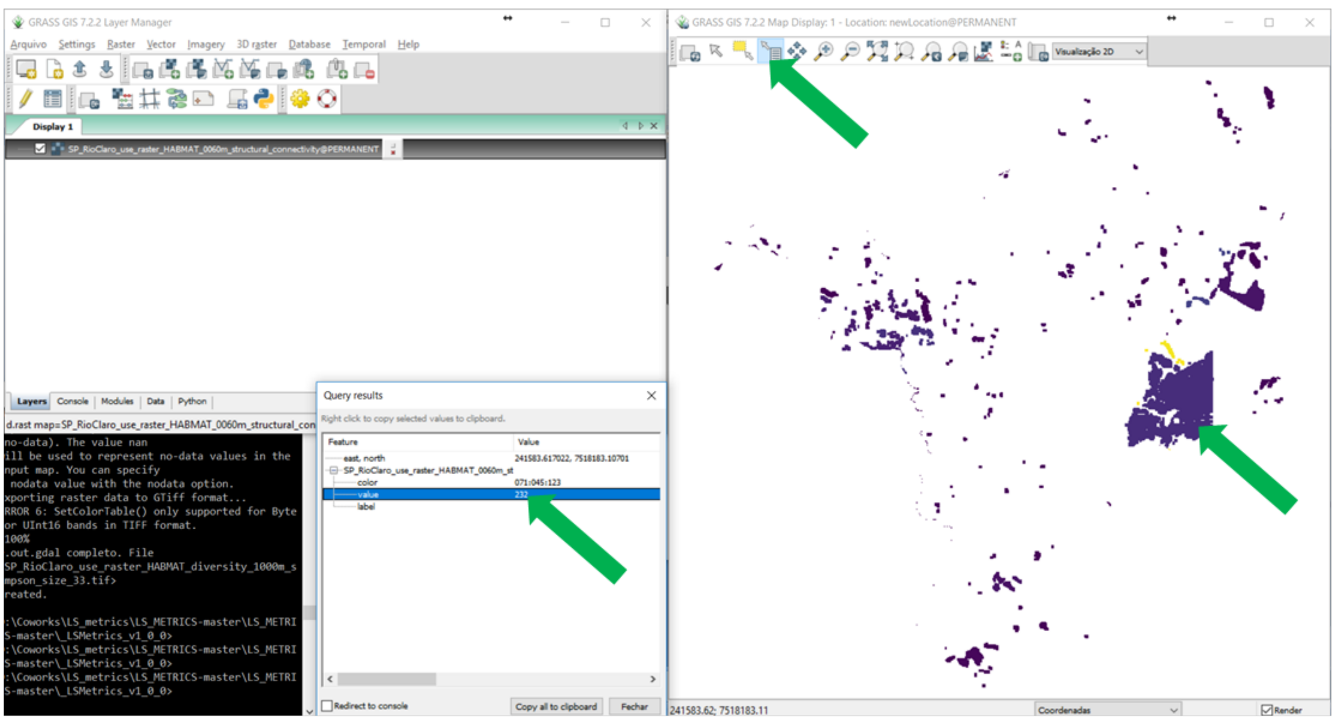

### Structural connectivity

#### *0060m*structural_connectivity

Difference (given in ha) between the Patch size and the Fragment size metrics; when a patch has only one fragment, its structural connectivity will be zero.



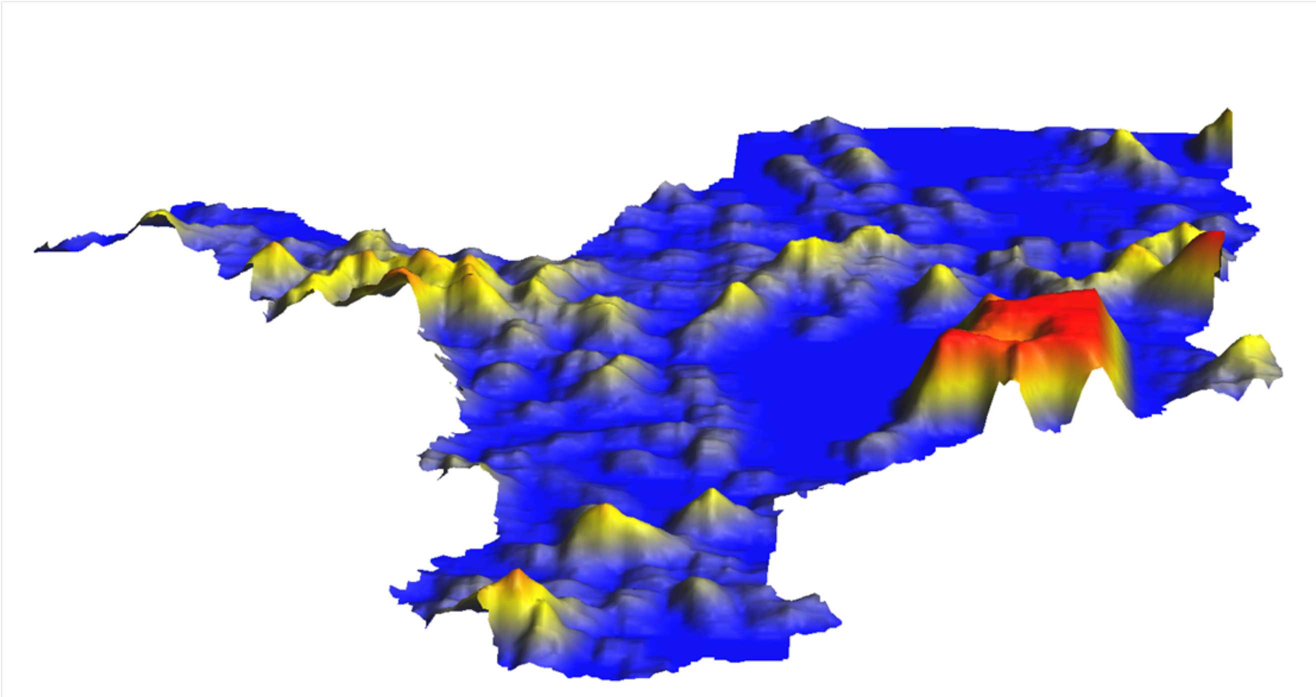
Query results are below:



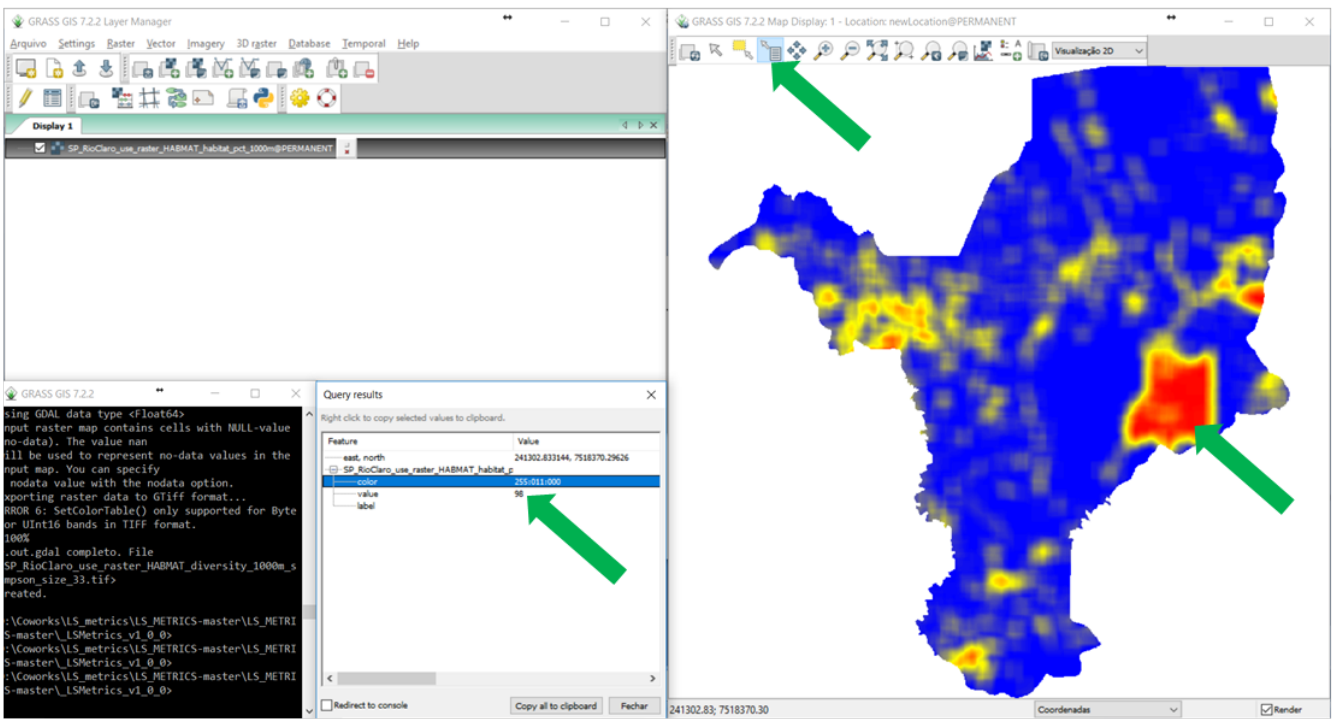

## Proportion of habitat

### *habitat*pct_1000m

Proportion of the target landscape classes considering a selected window size.



Query results are below:

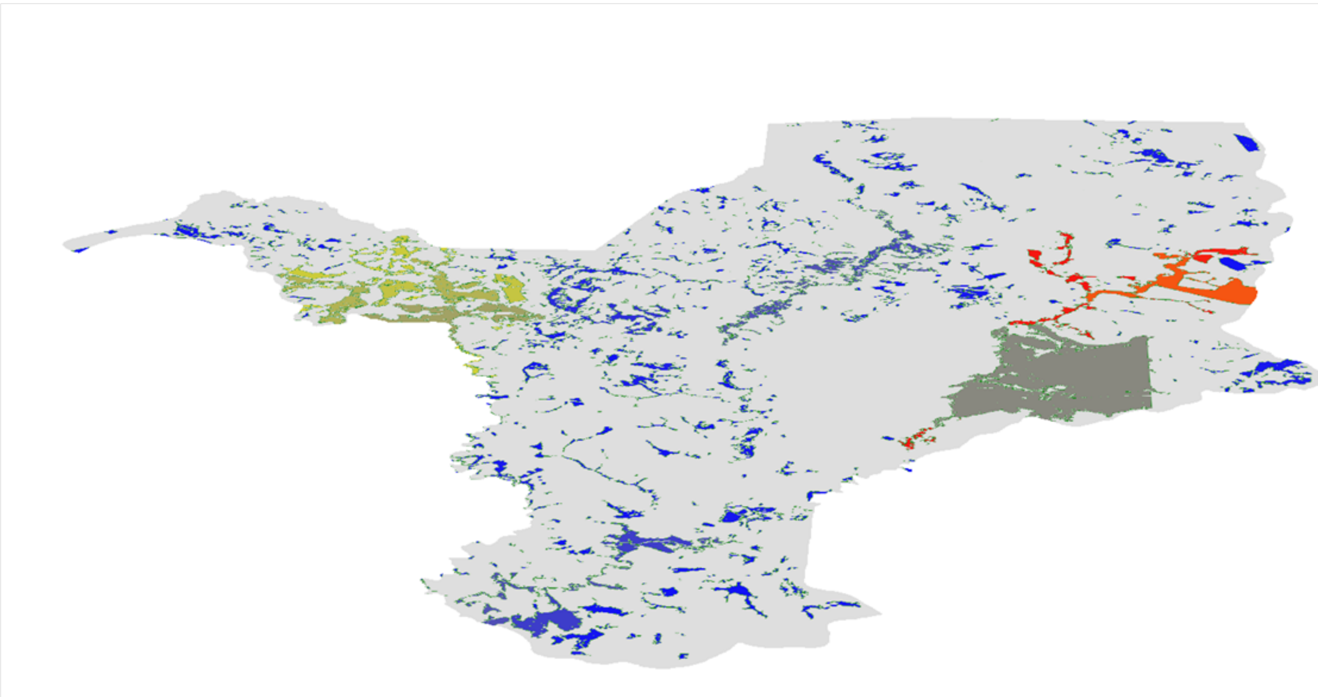



**How to choose the window size?** The user must decide which window size will be used to calculate metrics. It corresponds to search extent from each pixel where the calculations will be applied. It can be based on the extent of landscape perception of the target species, process or groups analyzed. It is given in meters.

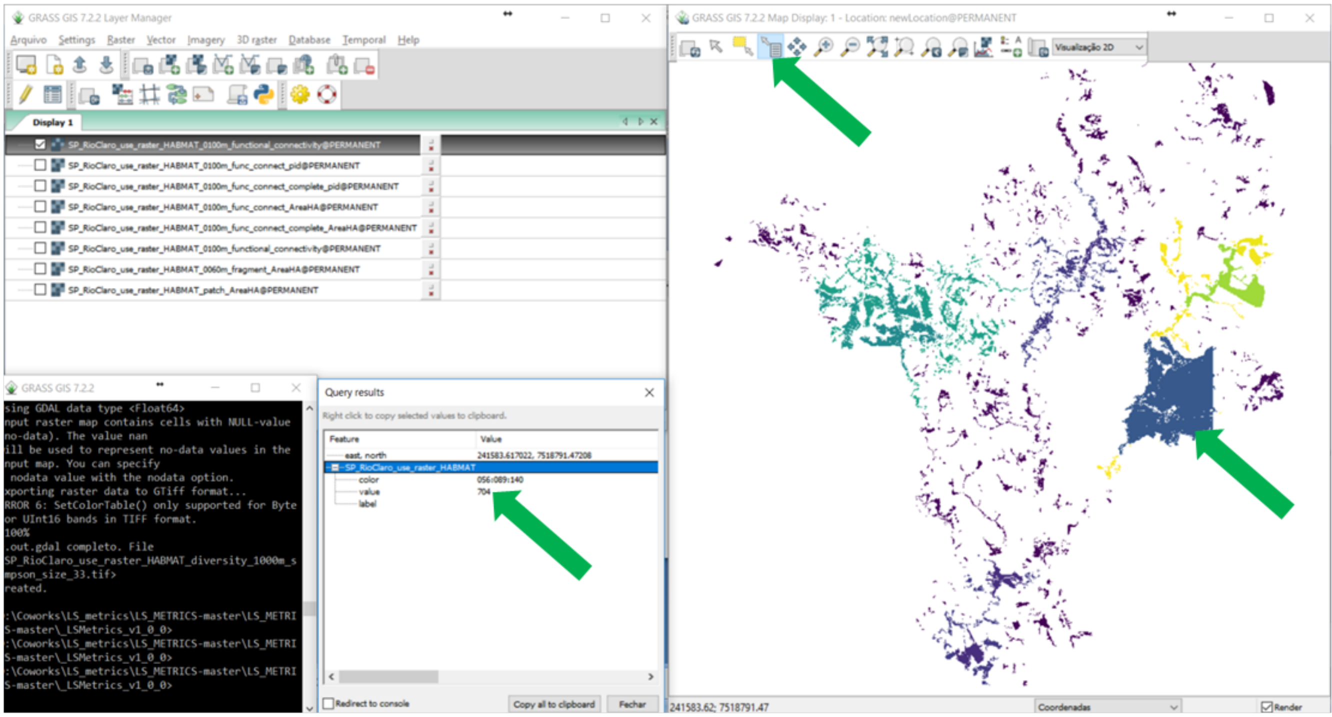
## Metrics of functional connectivity

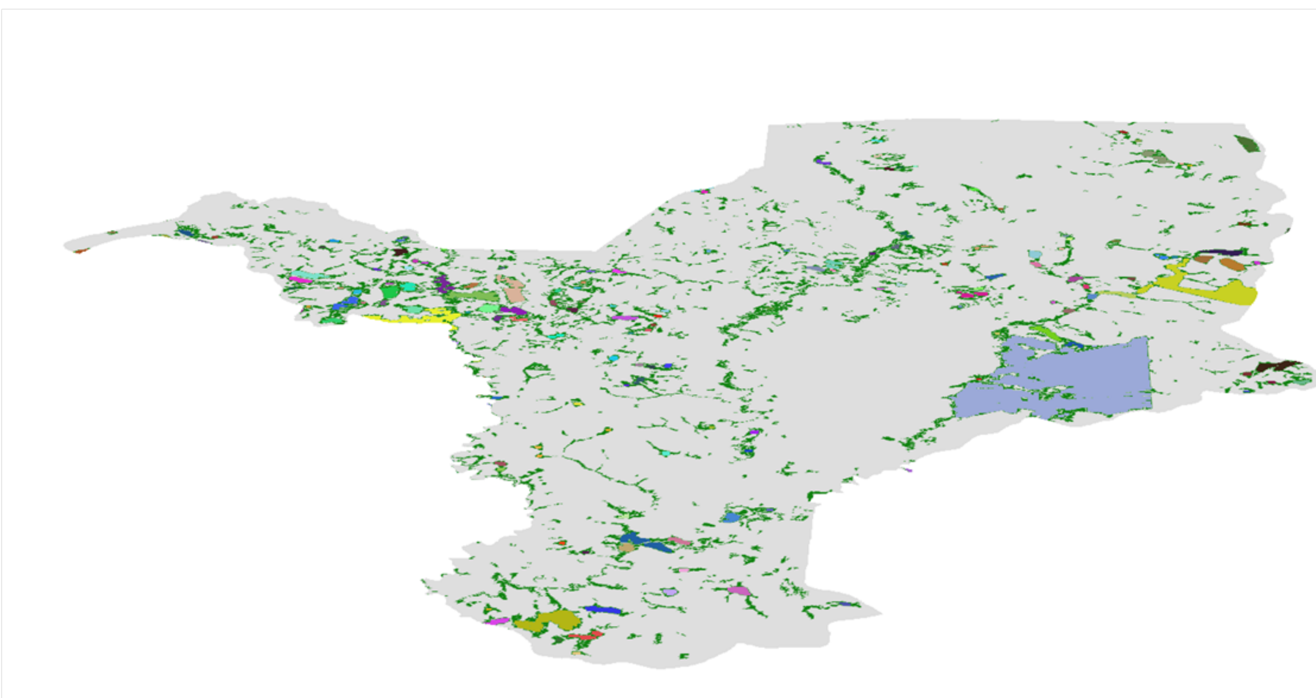
### Functional connectivity

The amount of area functionally connected to a habitat patch, given a gap crossing capability of a species or group of interest. First, a buffer of interest equal to half of the width (in meters) that a species is able to cross is created. Then, the habitat patches that are close enough ara grouped within the same buffer. We then sum the area (ha) of all this grouped patches. We refer to these groups of nearby habitat patches as habitat **clumps**. Finally, to obtain the strict functional connectivity, we calculate the difference between each clump size (ha) and patch size (ha);

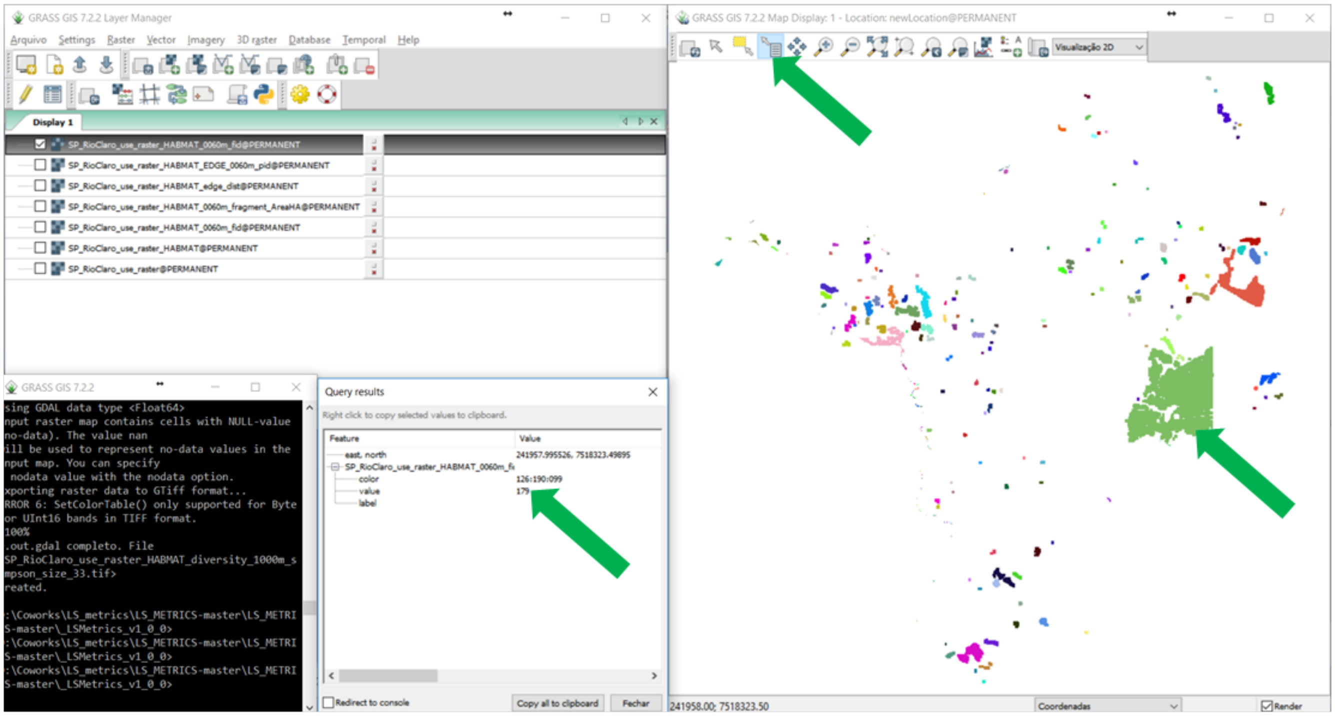


Query results are below:



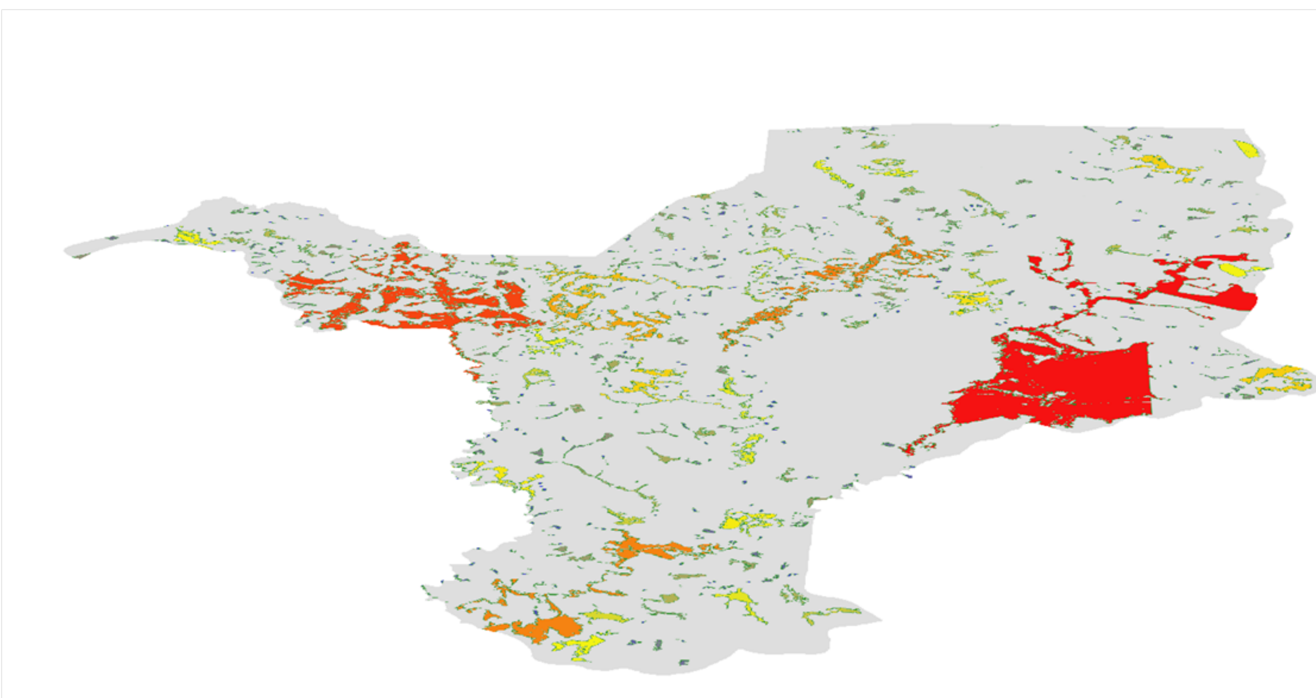



**FIDs** Query results are below:

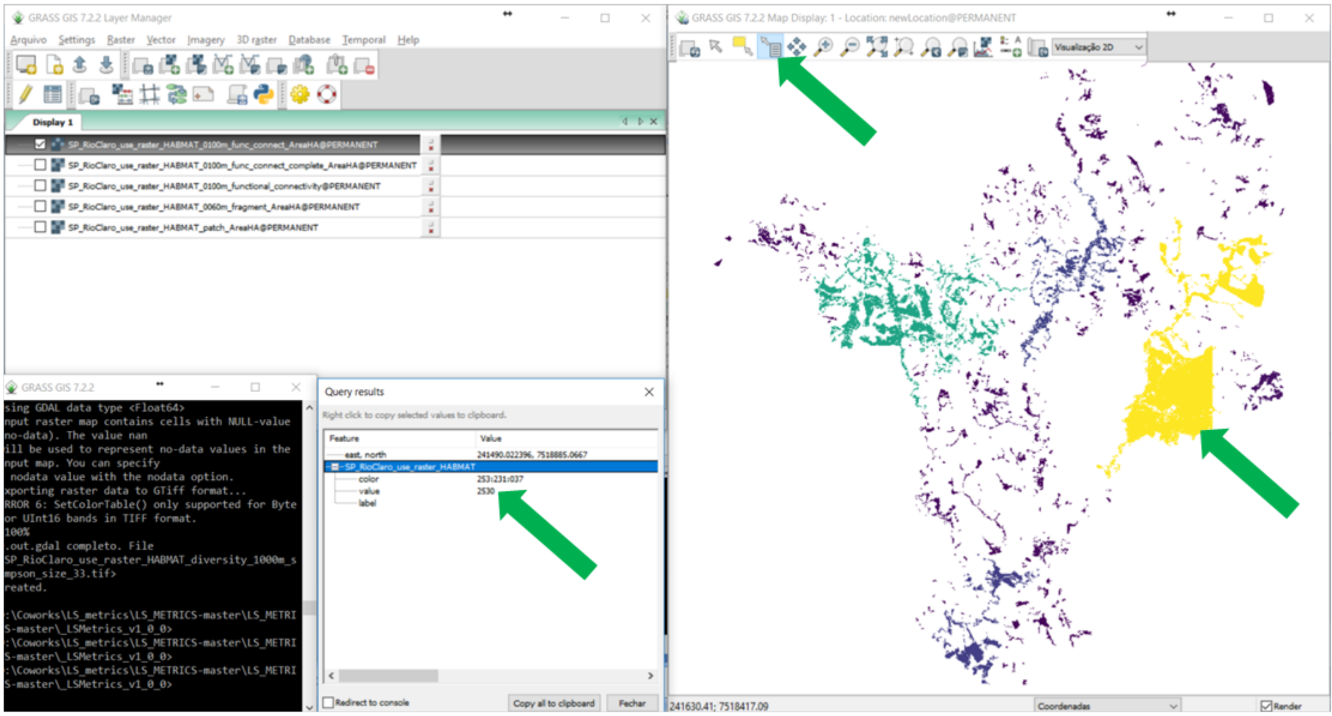


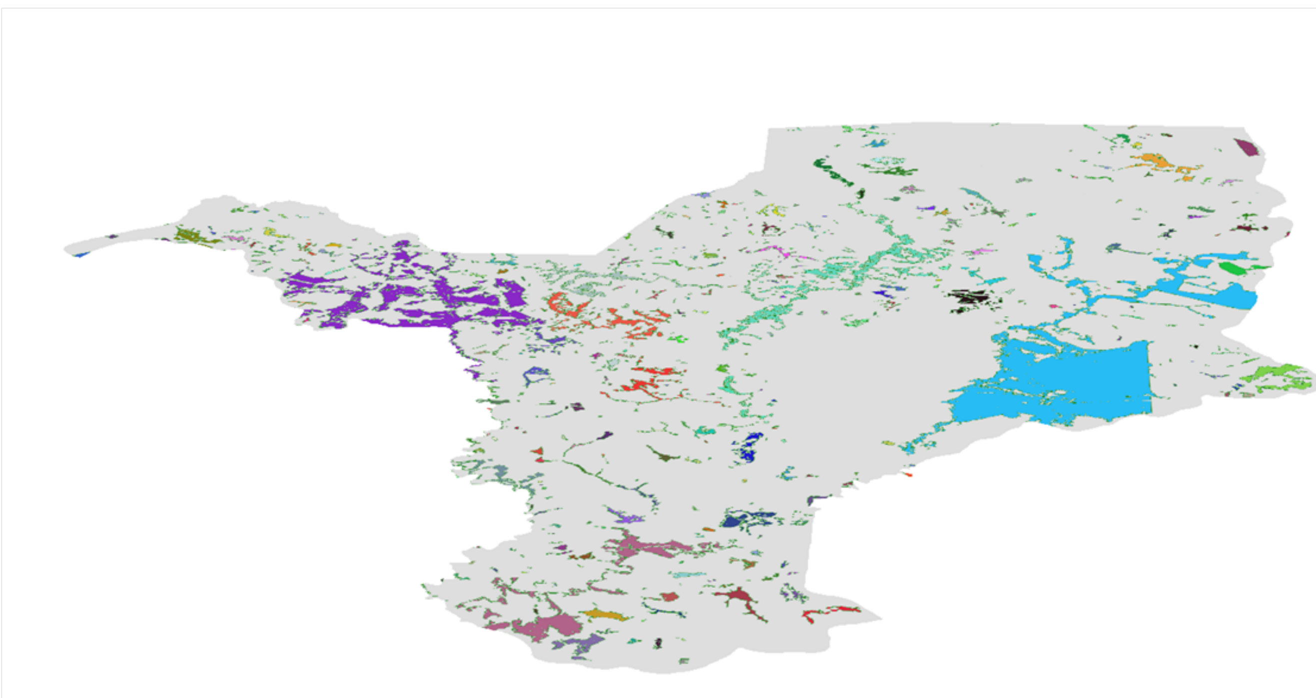

## Functionally connected area

Represents the total amount of habitat area (in hectares) functionally available for an organism located in a patch, given a gap crossing capability of the organism. This is calculated by generating a buffer of size equal to half of the gap crossing capacity of the organism, around all patches, grouping all habitat patches close enough in the same clusters of patches, and summing up the area of all the grouped patches.

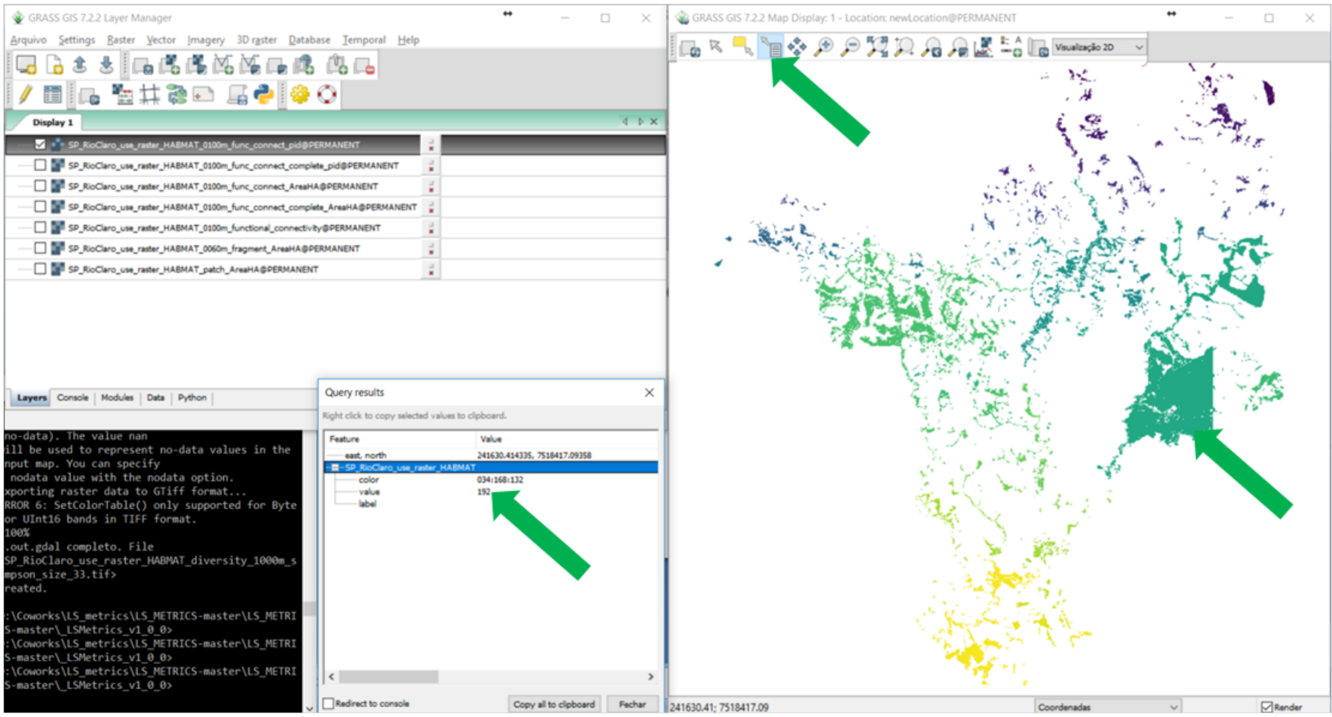


Query results are below:



**PIDs for Functionally connected area** Query results are below:

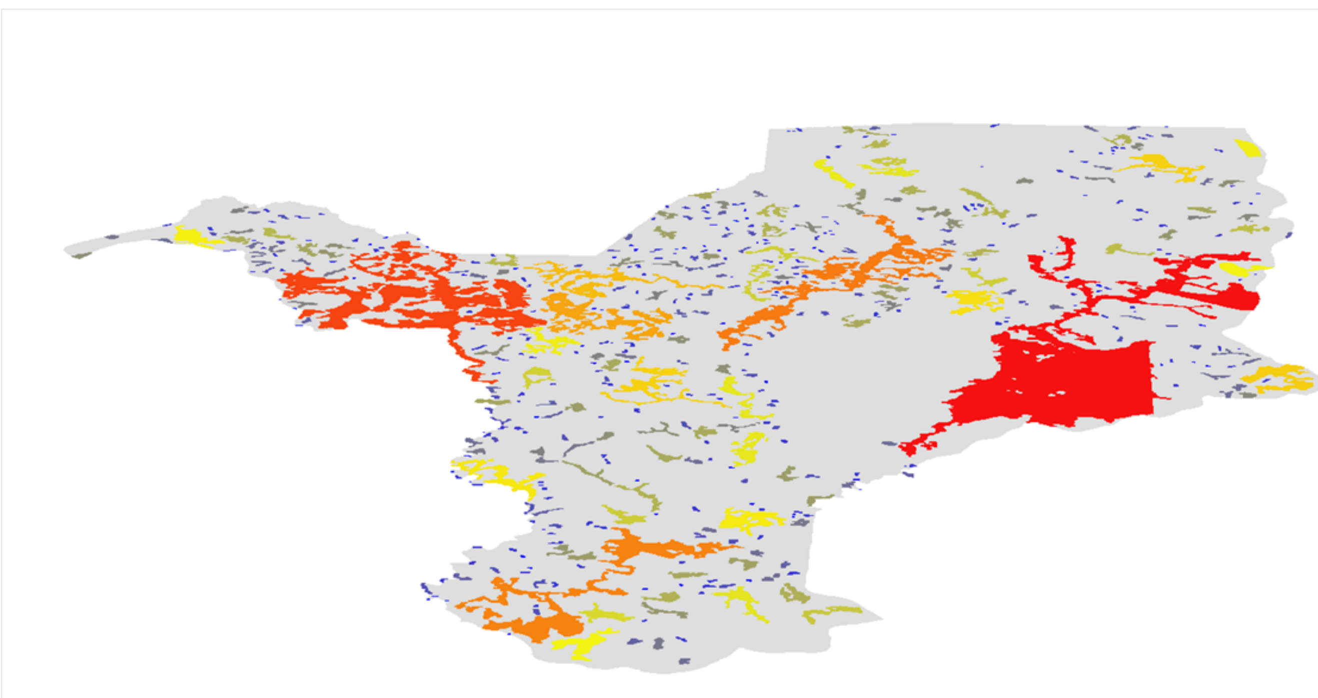


### Complete functional connected area

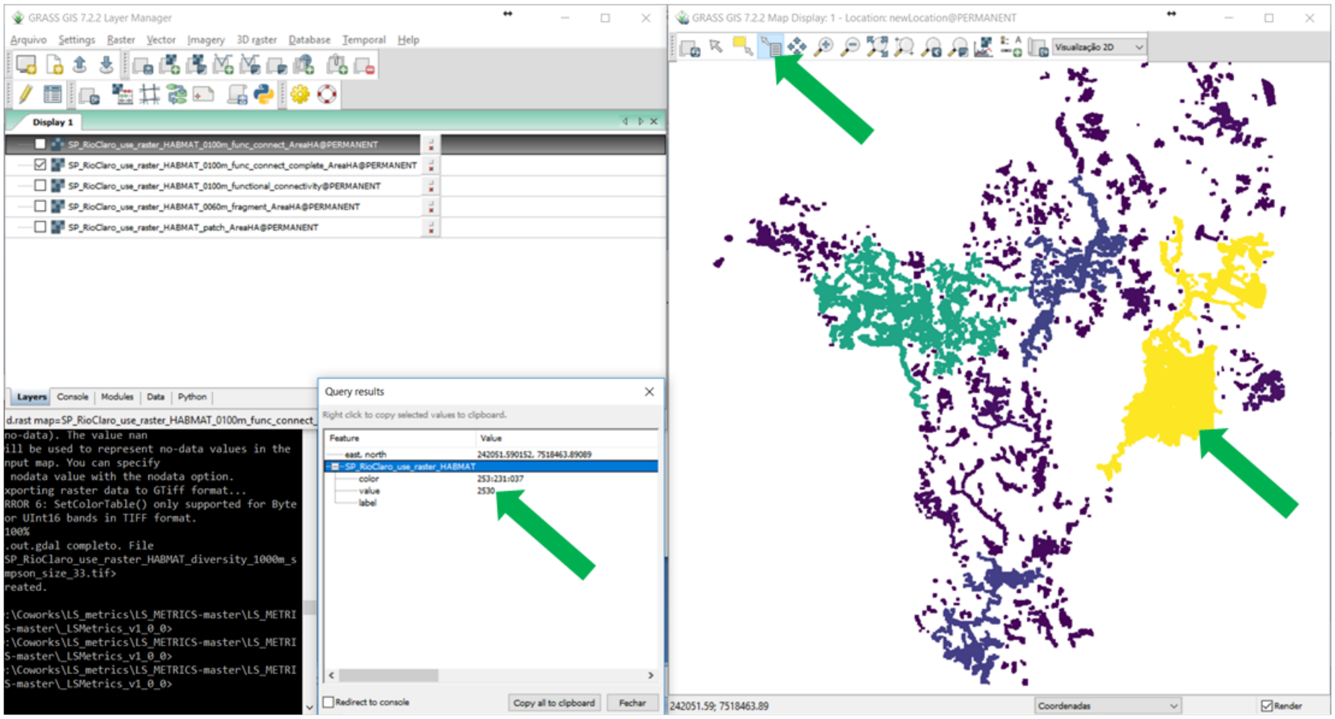
The total amount of habitat functionally available for any species, given a gap crossing capability. We do all the steps for the “Functional connectivity” calculation, but do not subtract the patch size from the clump size.

***func*connect_complete_AreaHA**

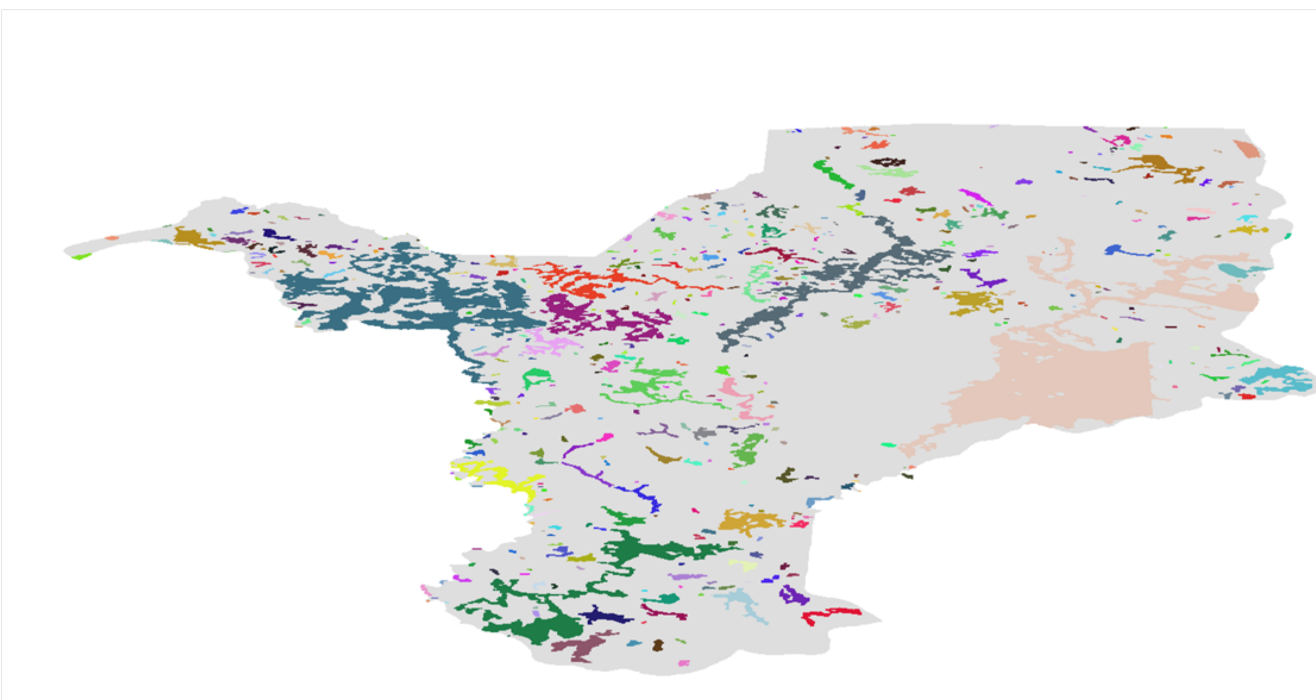
Query results are below:



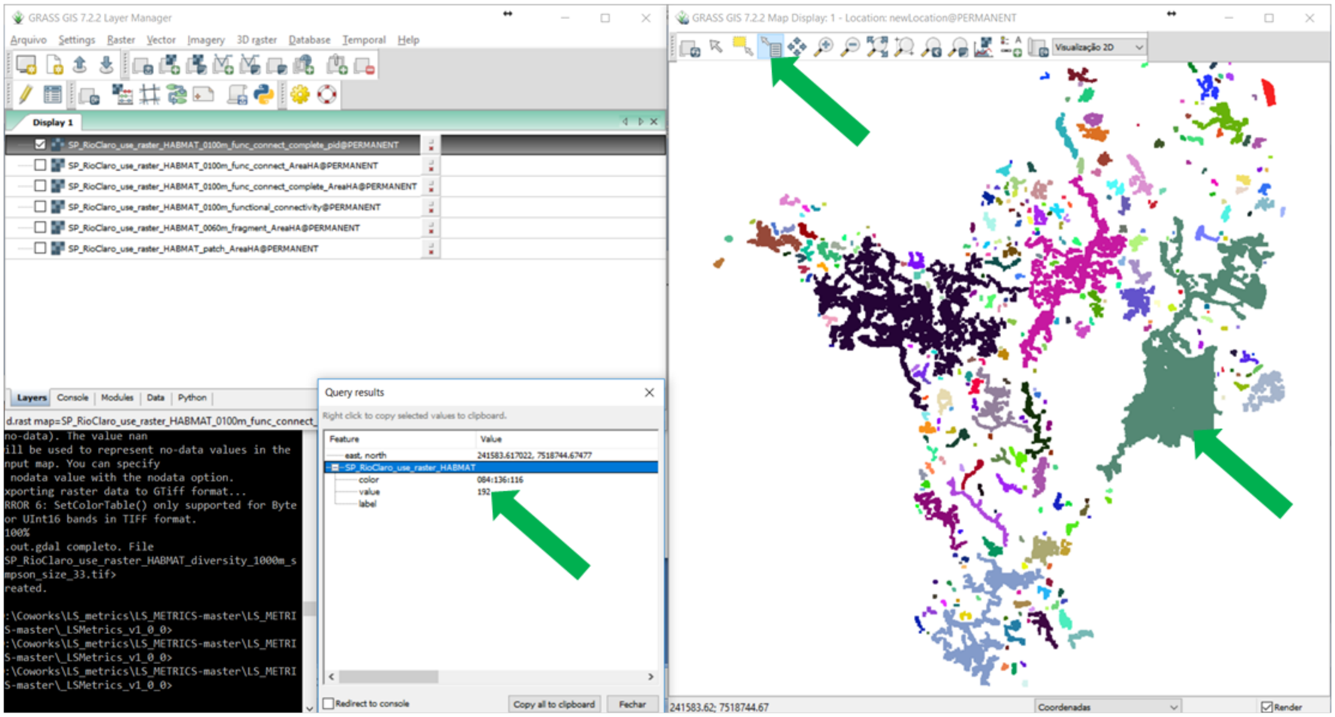
Query results are below:




## PIDs of complete functional connected area *0100*func_connect_complete_pid



Query results are below:



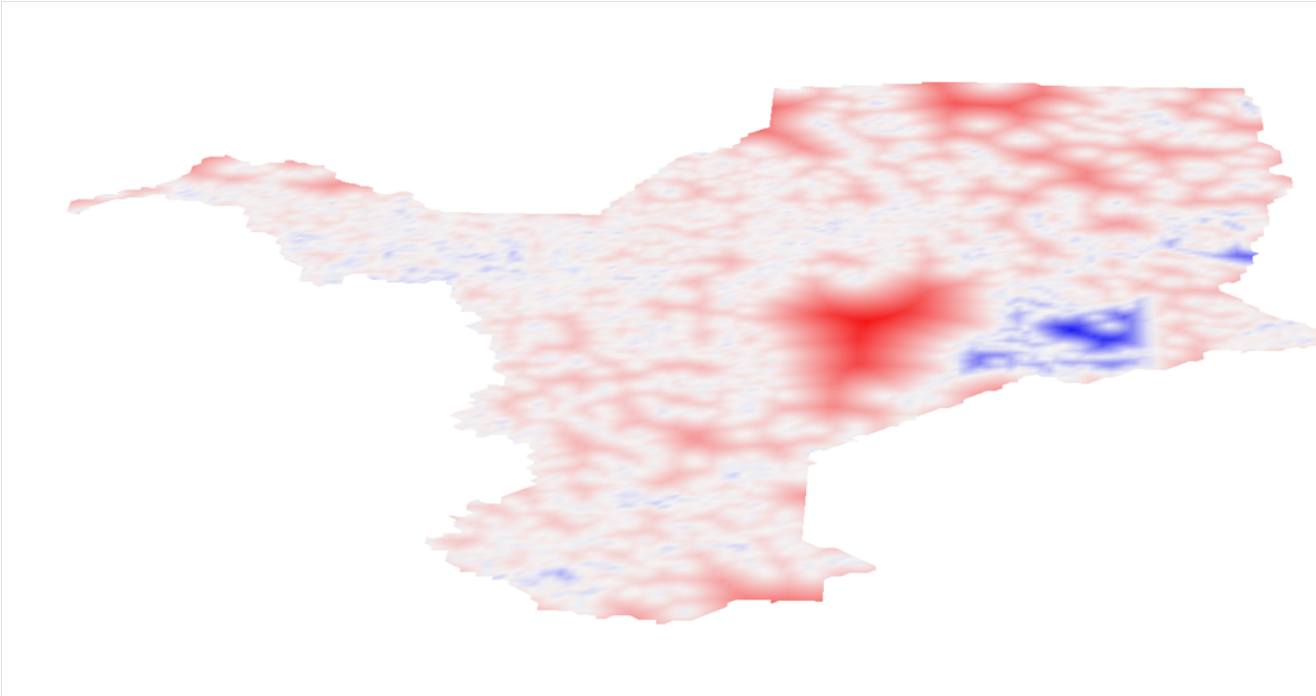

## Metrics based on edges

**Edge depths** In order to run metrics for edge, the user must select a value for edge depths. This parameter is a proxy for investigating the impact of edge influence on the different pixels. It estimates the distance of edge influence and should be carefully thought a priori by the user, since it will be used to calculate the output metrics. It is set in meters, and as larger the value, larger the edge influence and extent through landscape.

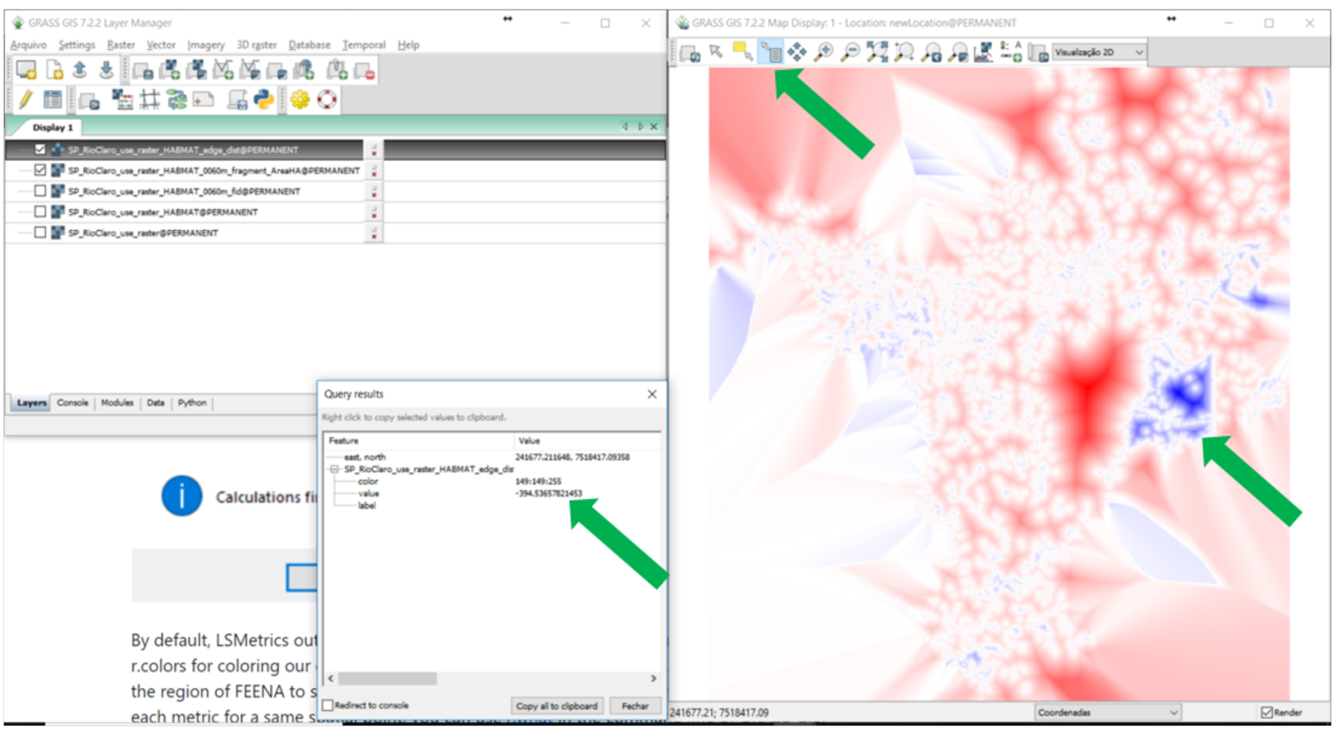
See the charts and brief explanation of each metric below:

### Map of distance from edges

The output value of each pixel will represent the distance of this pixel to the nearest edge.

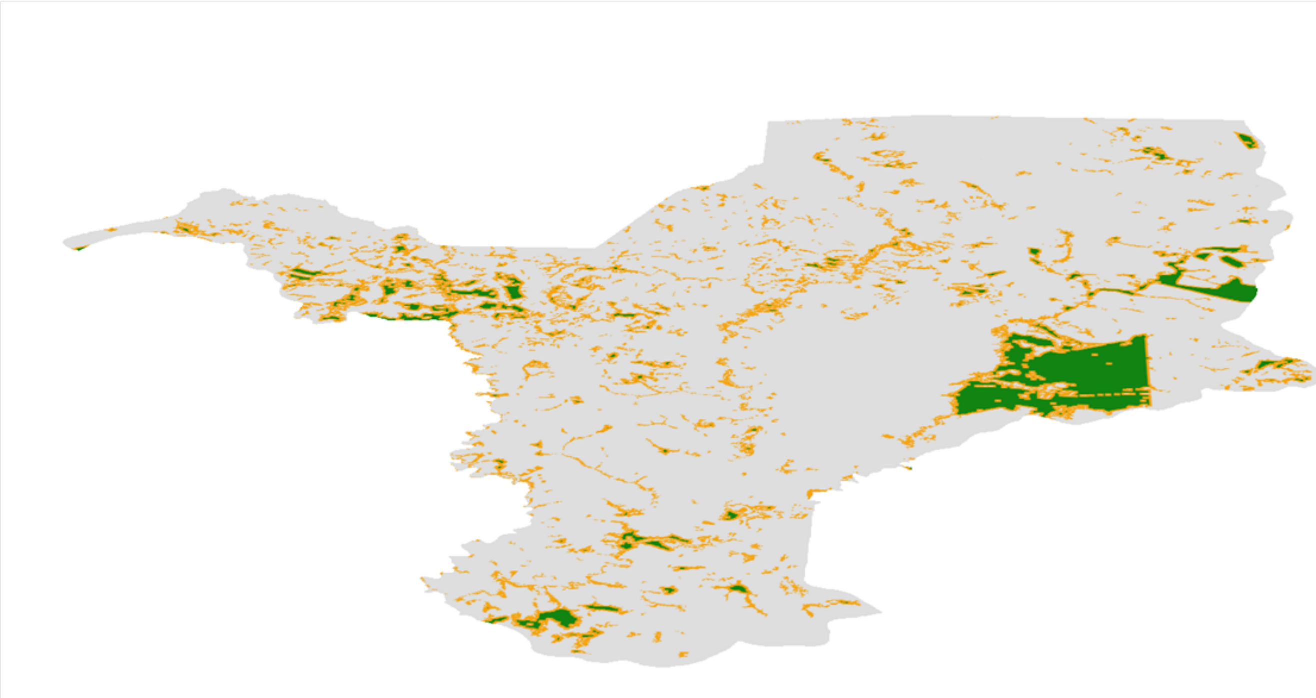


Query results are below:

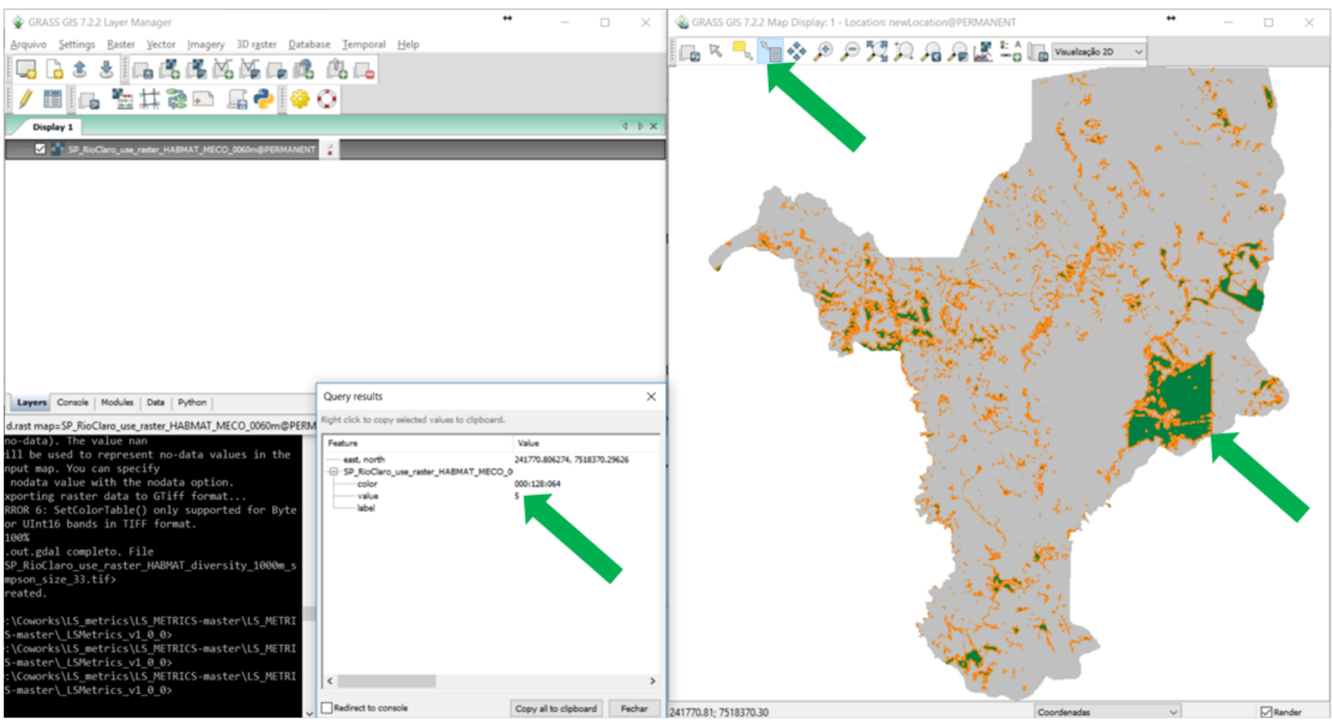



## Classify edge/core/matrix (MECO)

Considering a specified Edge depth, this metric classifies the pixels as belonging to three categories: edge, core and matrix. How species really use habitat area would be represented by the core area, since many species have limitations on using edges or less quality areas than their preferred habitat.

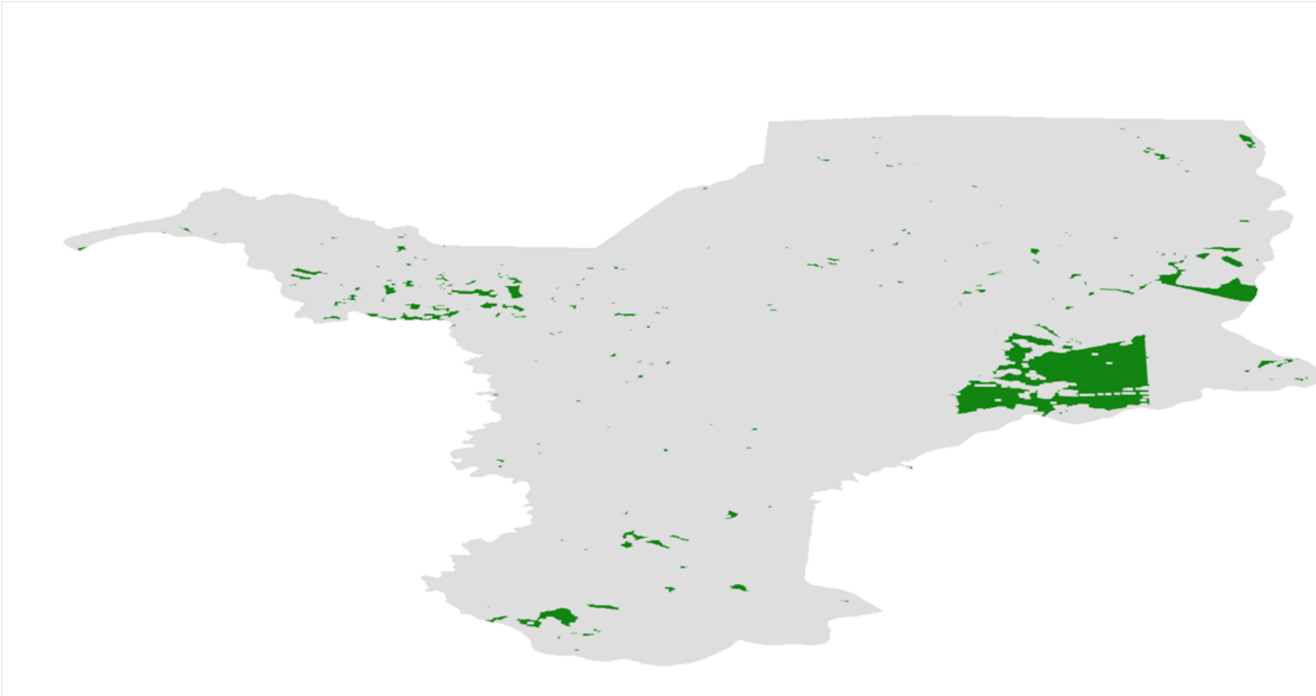


Query results are below:

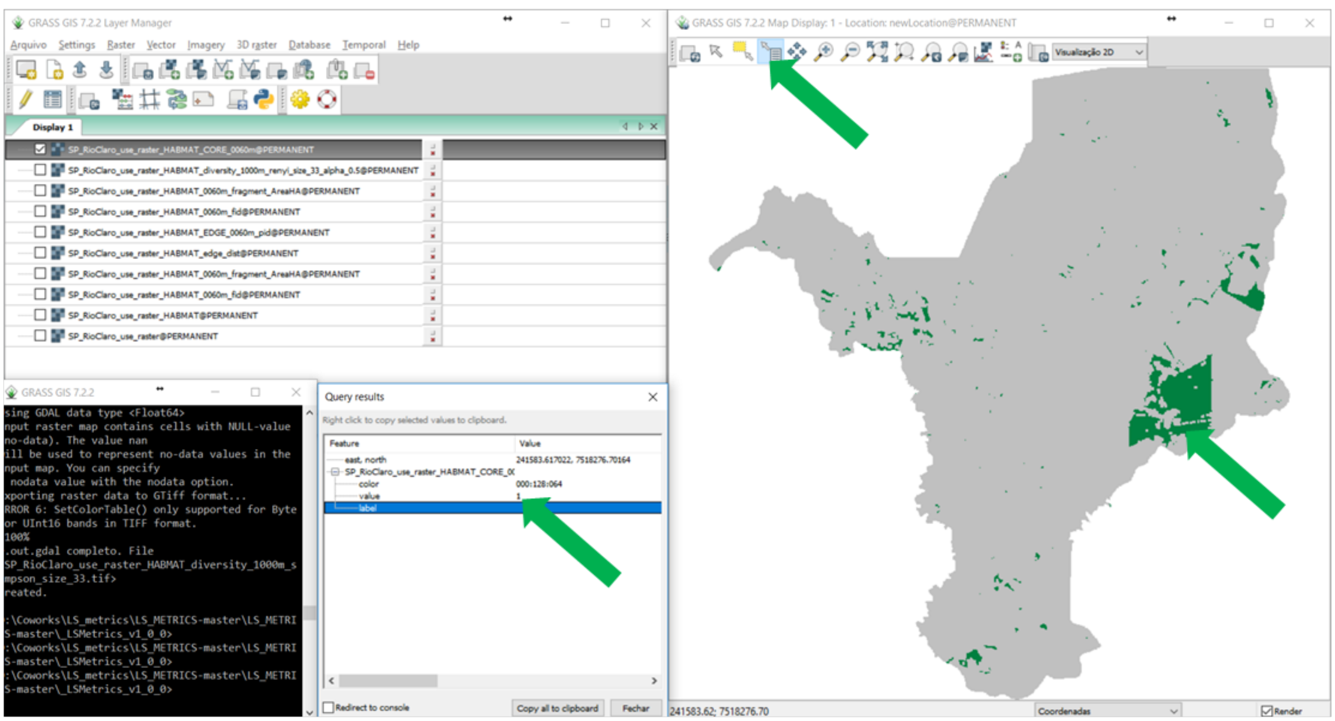



## Binary maps for edge and core

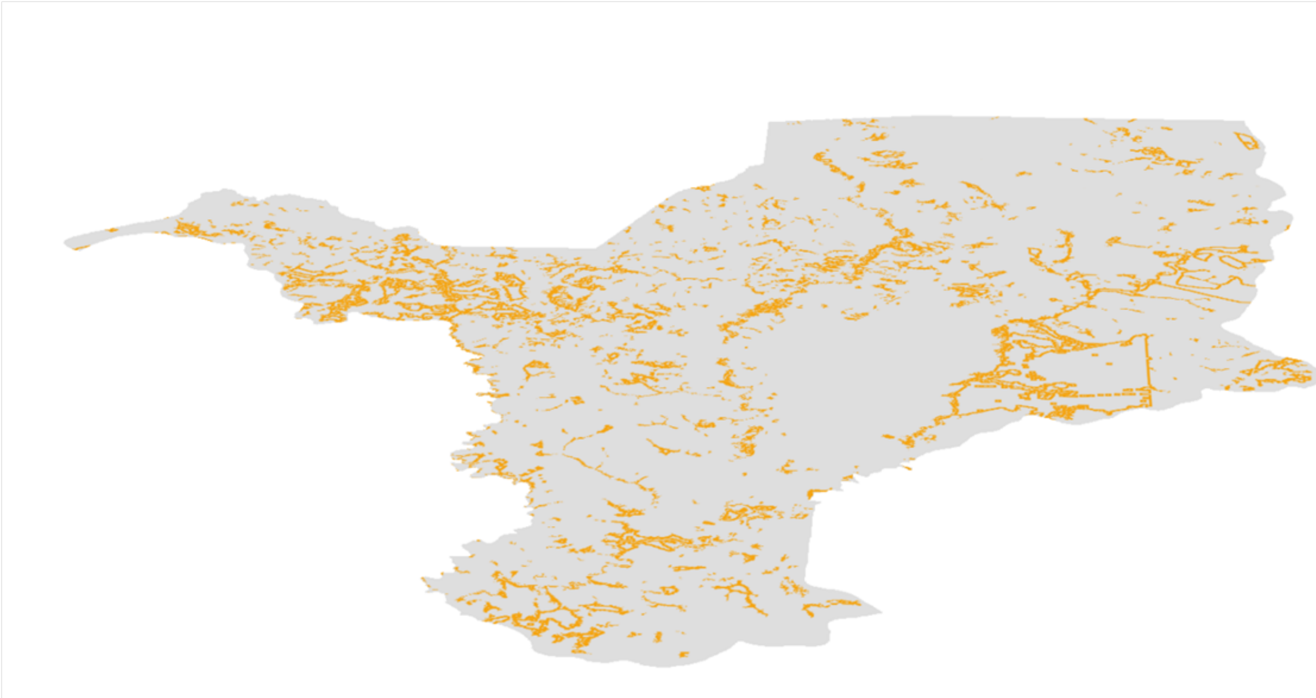
**Core binary maps** In the core binary map, core are will be set as value "1".



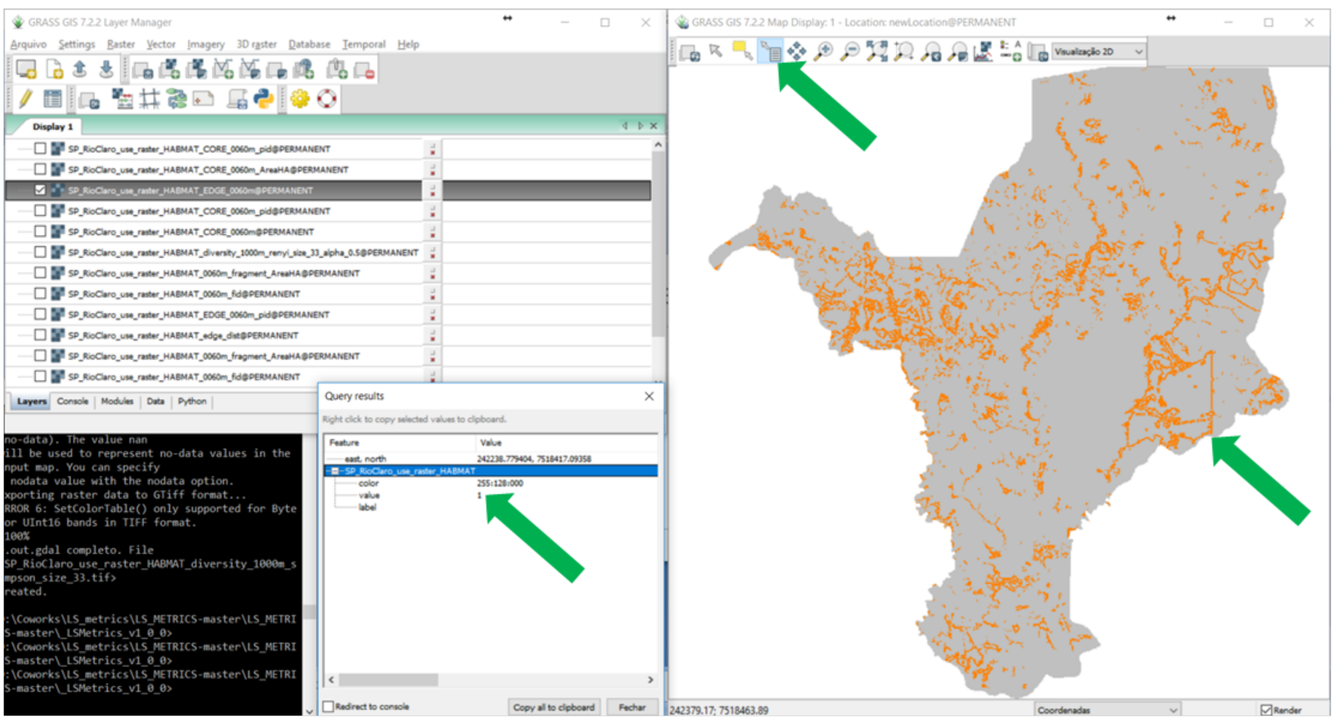
Query results are below:




**Edge binary maps** In the edge binary maps, edge are will be set as value "1".



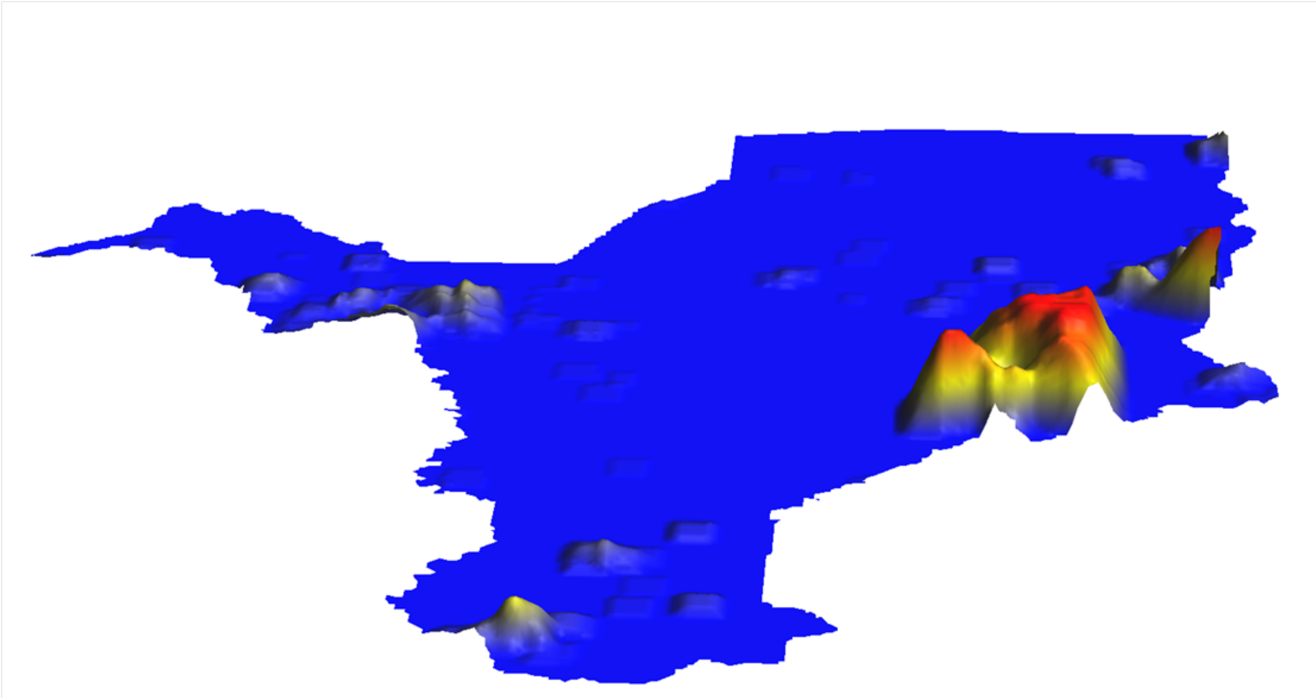
Query results are below:



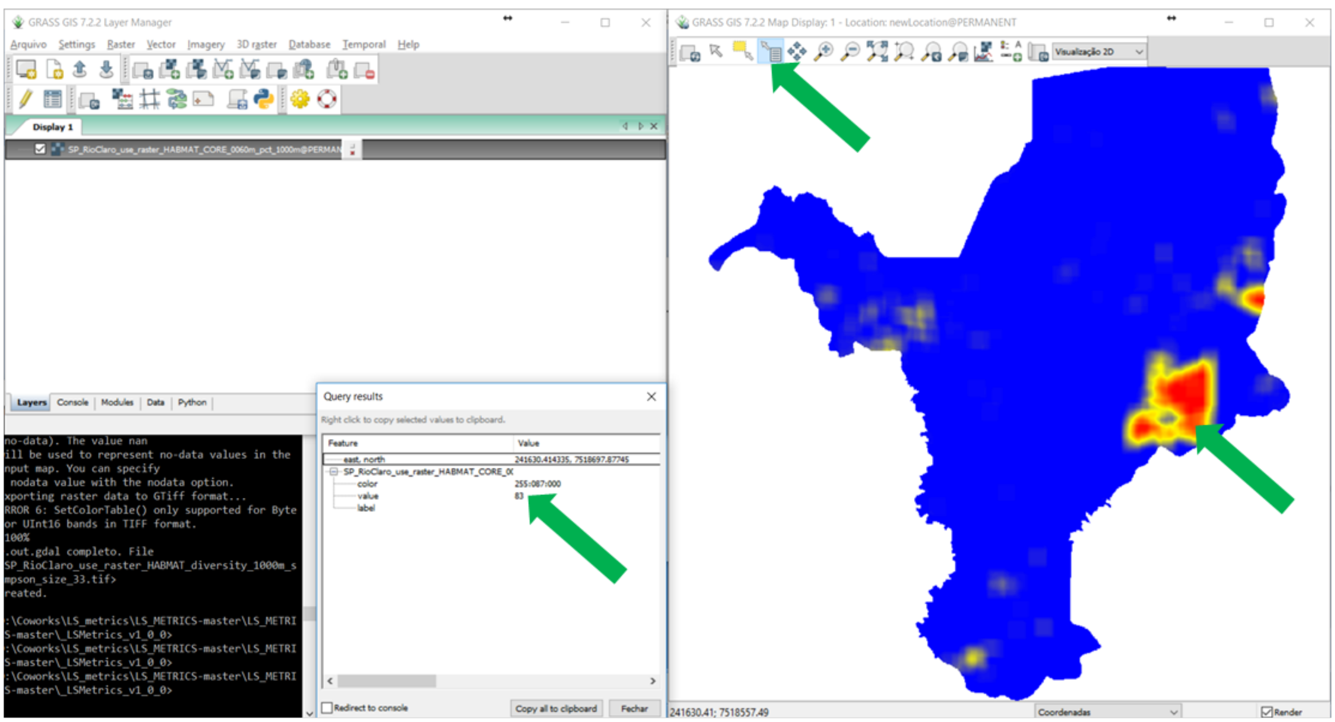

### Proportion of edge/core

Percentage of habitat and edge within a search radius (Window size parameter), given a specified edge depth (Edge depths parameter).

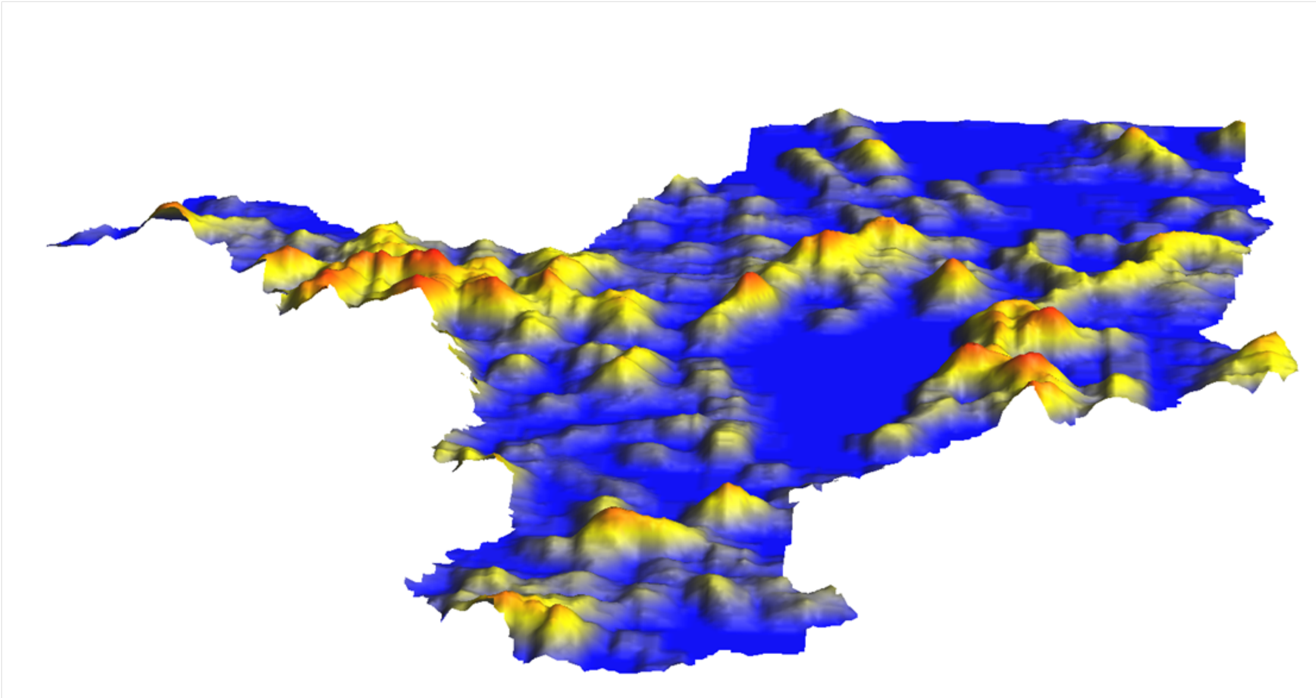
***HABMAT*CORE_0060m_pct_1000m**



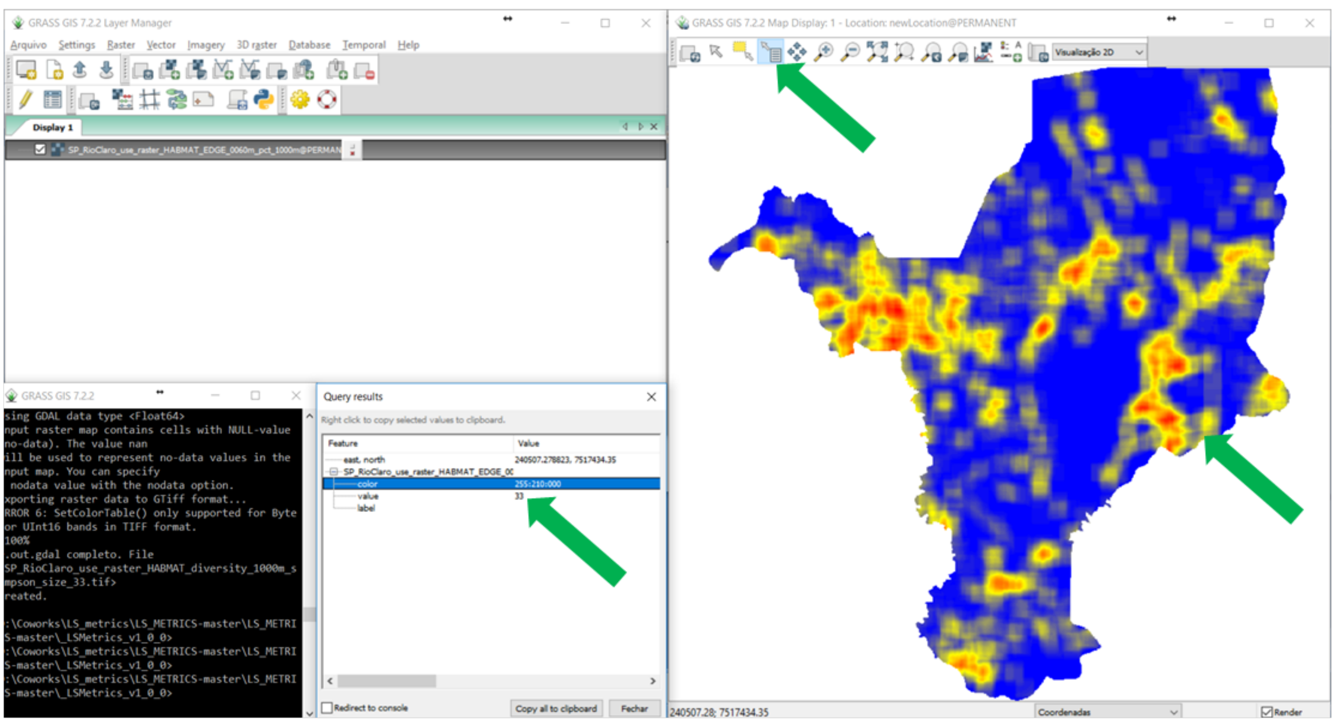
Query results are below:




**HABMATEDGE_0060m_pct_1000m**



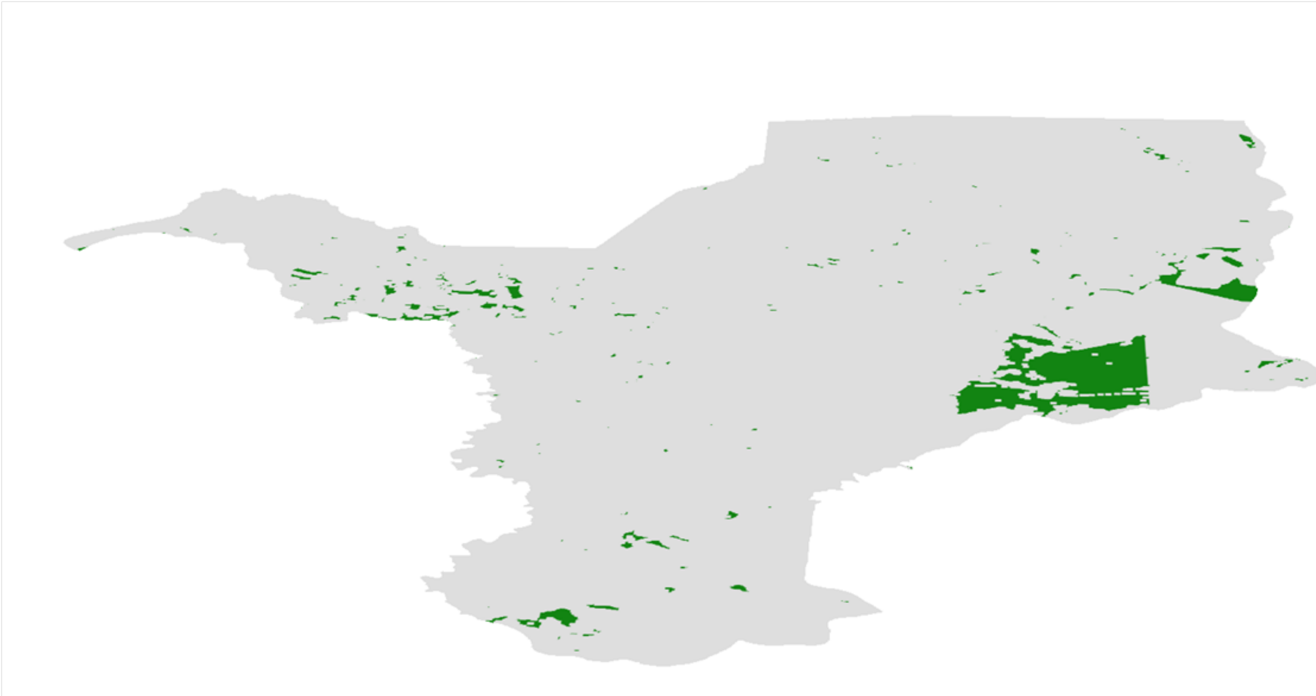
Query results are below:



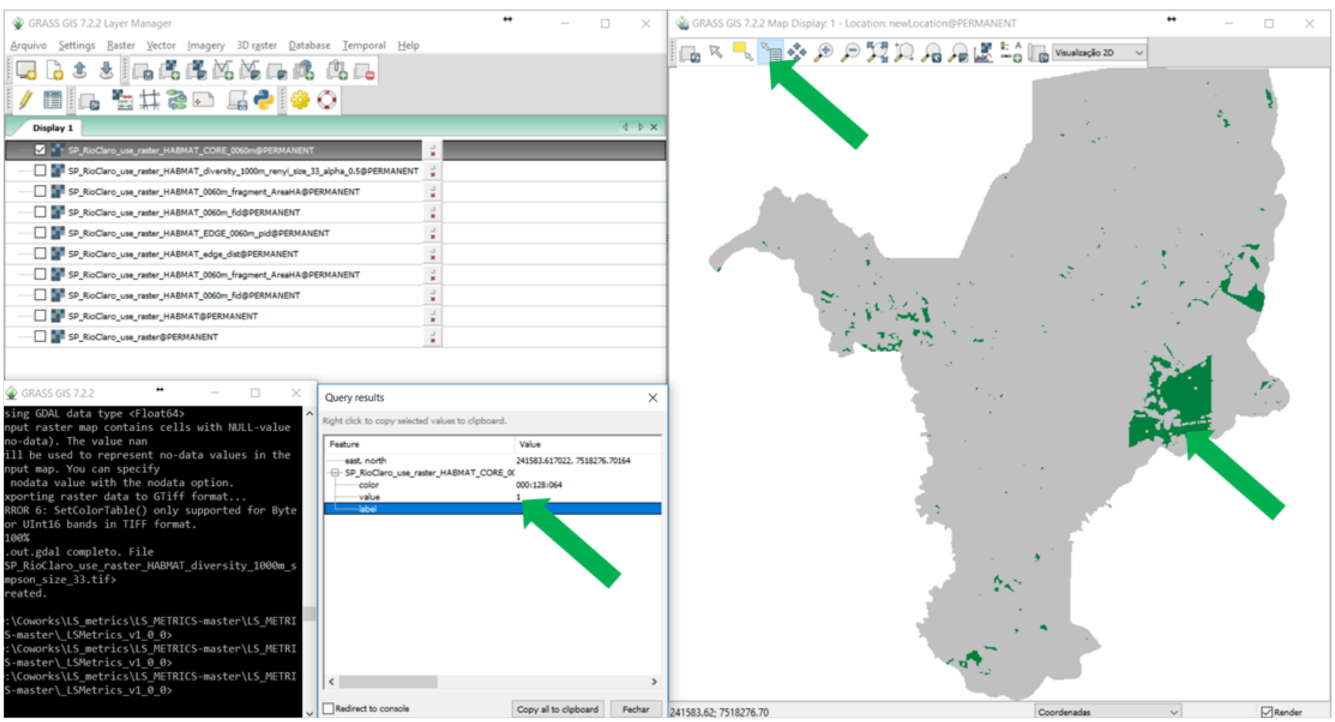

**Calculate area of edge/core clumps?**

Given edge and core areas were identified, contiguous cells are clumped into edge and core clumps (i.e., cells of edge and core that pertain to the same patch), and their areas are calculated (in hectares). Two maps are created for each one (edge/core) for each landscape: an edge (core) clump ID (edge or core PID) map, which sets a distinct integer number to each clump of edge (core) cells, and a clump size map, which sets the the area of those clumps; non-edge (non-core) cells are set to NULL in both.

### ***HABMAT*CORE_0060m_AreaHA**

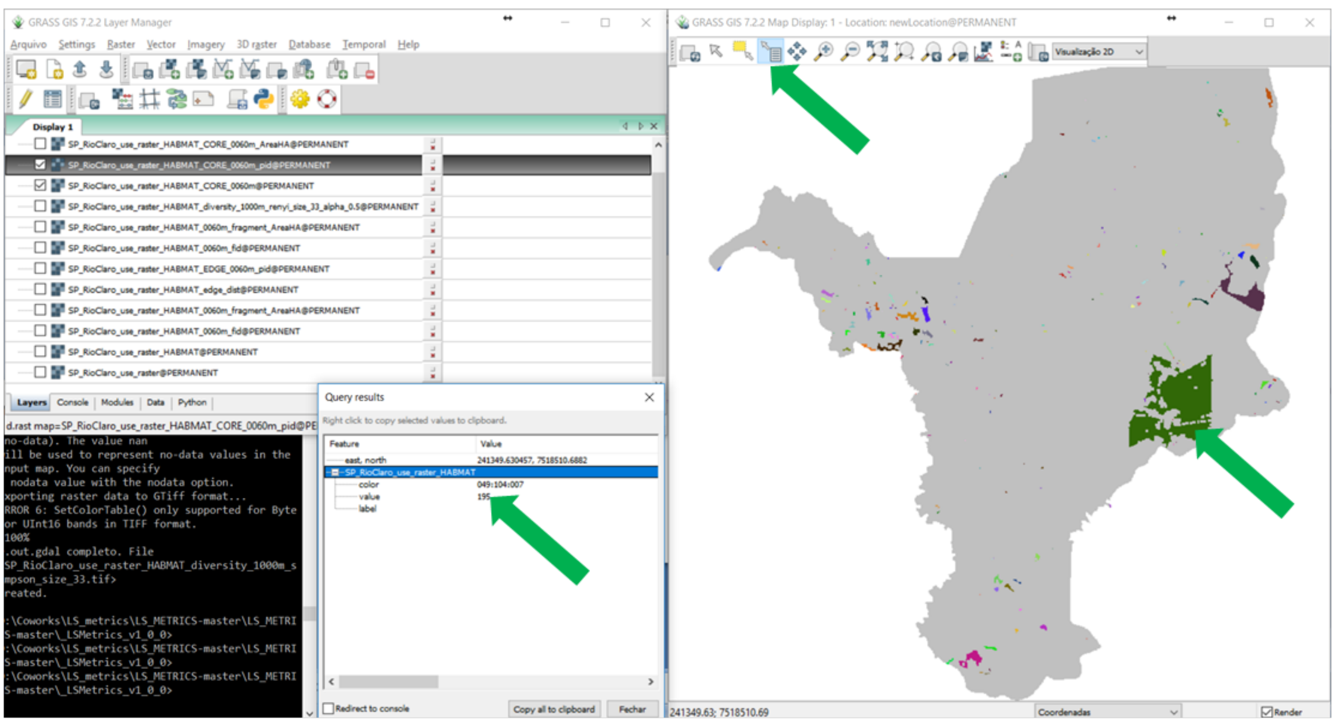


Query results are below:

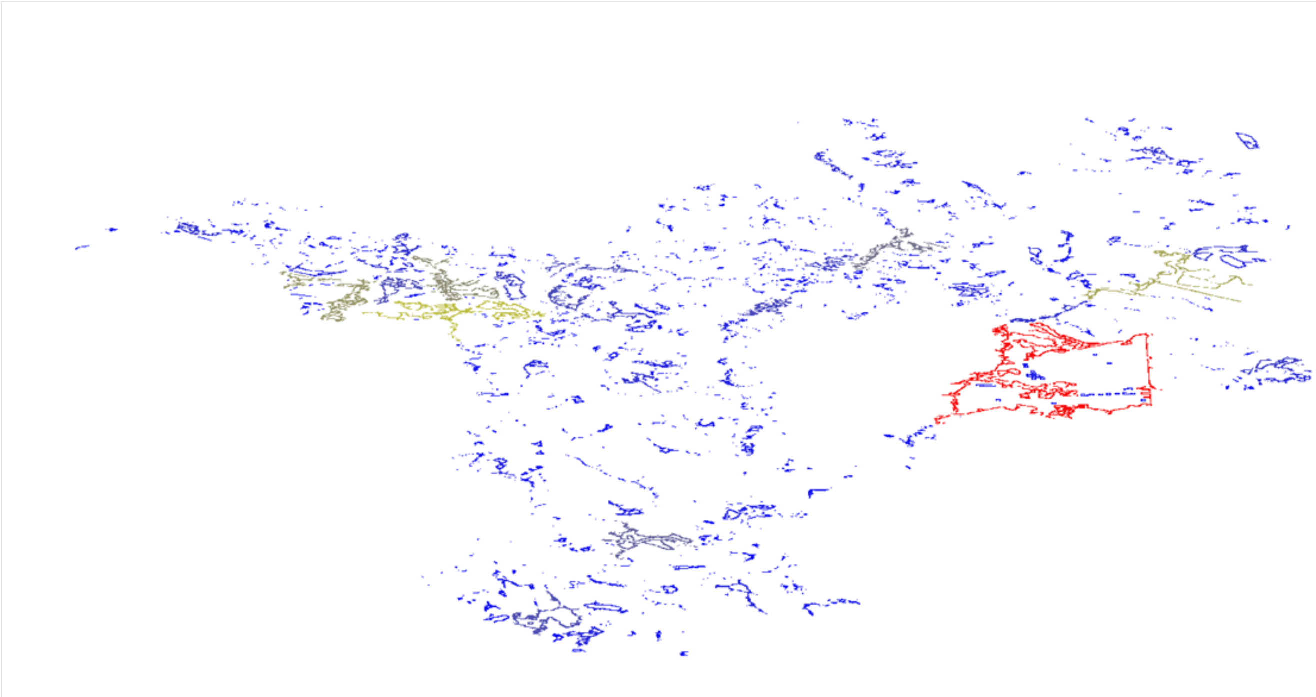



### ***HABMAT*CORE_0060m_pid**

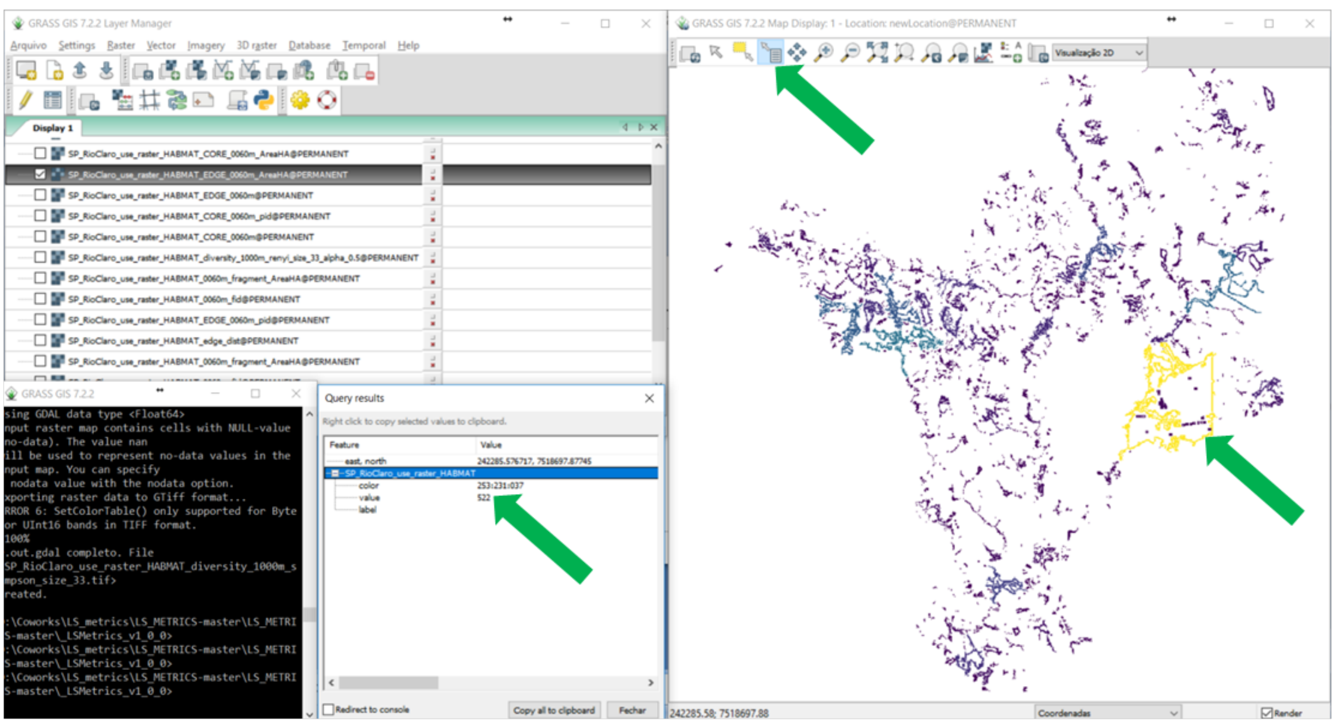
Query results are below:



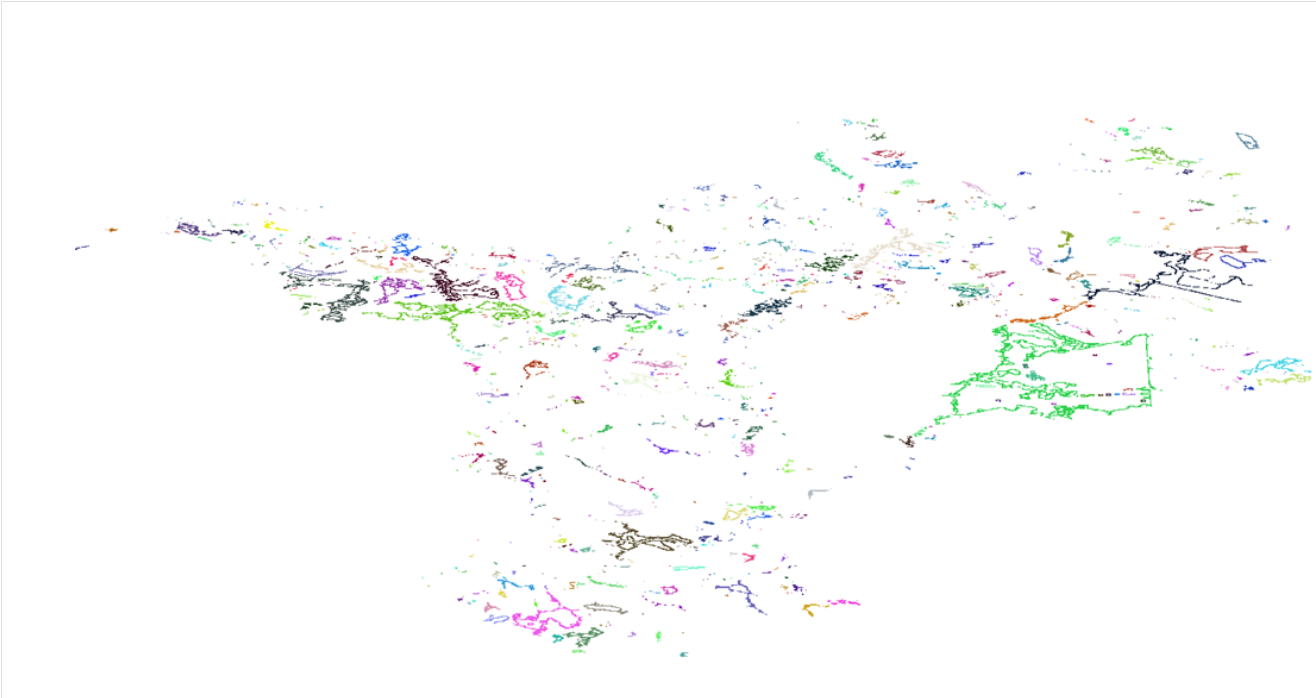

***HABMAT*EDGE_0060m_AreaHA**



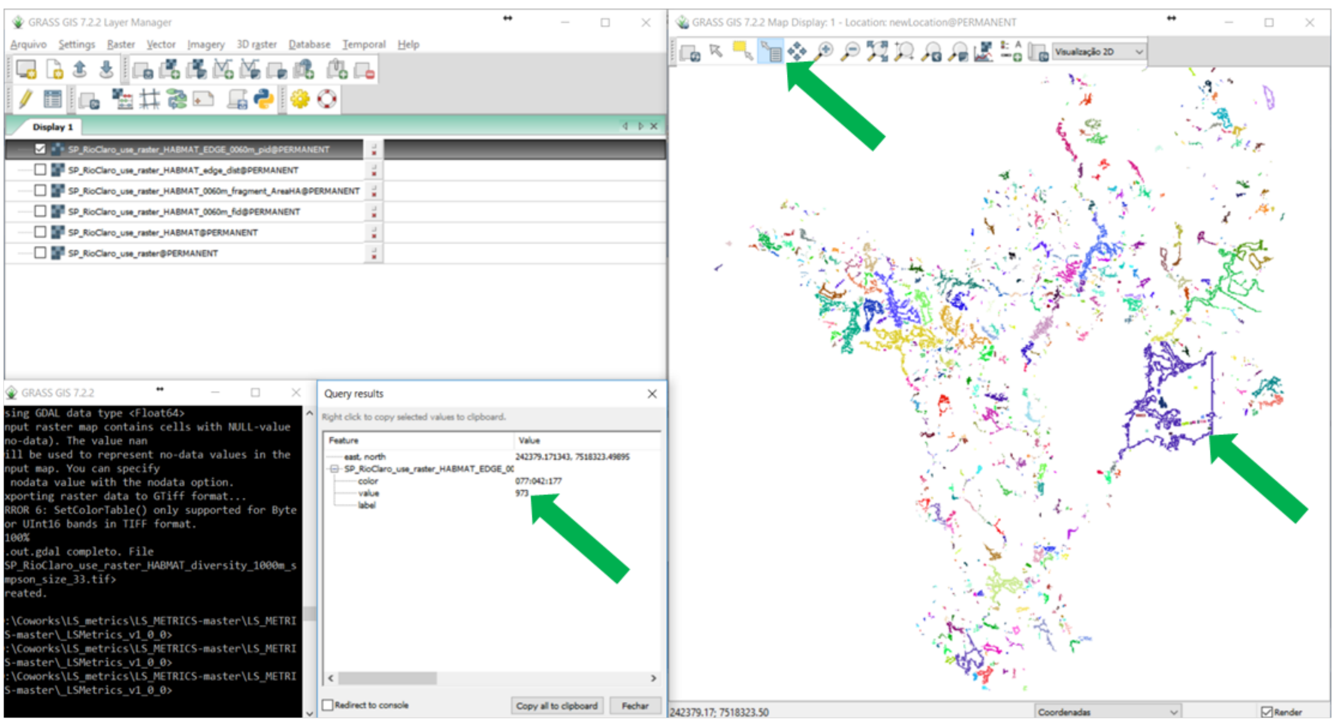
Query results are below:




## ***HABMAT*EDGE_0060m_pid**



Query results are below:

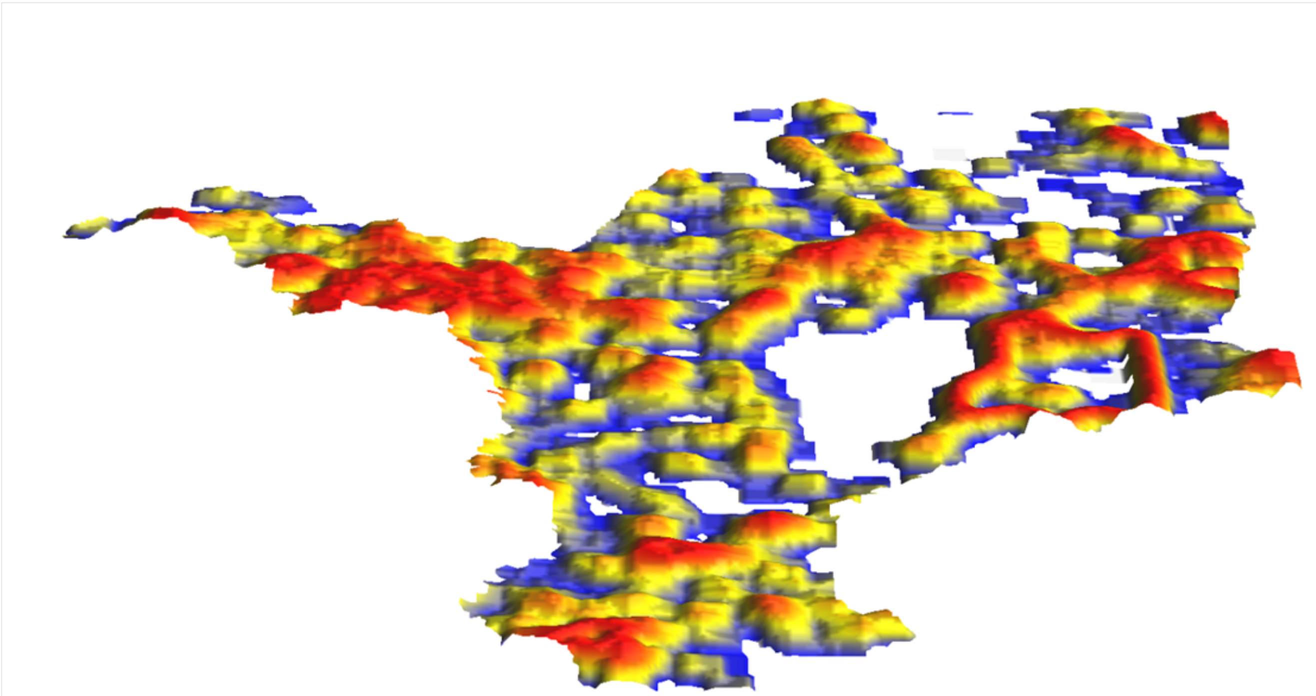



## Metrics of landscape diversity

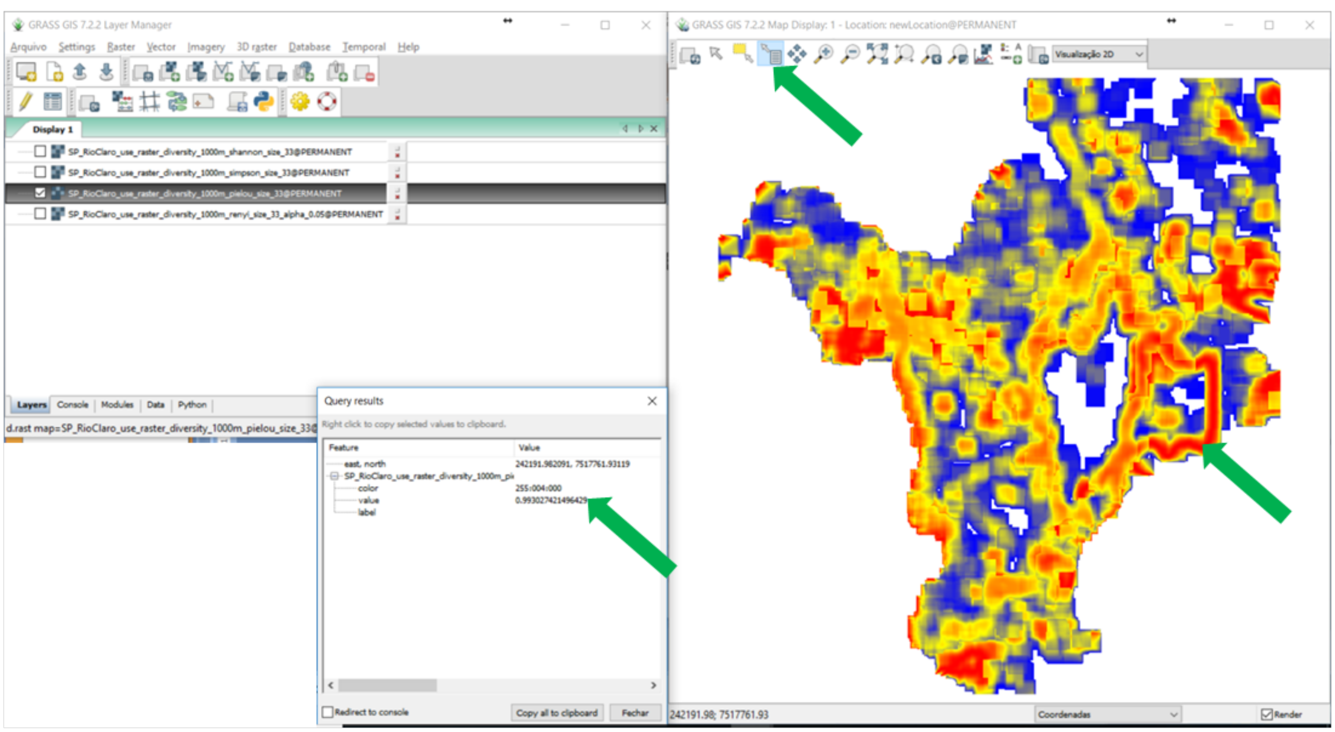
LSMetrics calculates four metrics of Landscape Diversity. See the example outputs below:

### Shannon

Given an extent value (or a list of values), the diversity around each pixel is calculated by computing the proportion pi of each land use inside a window of such an extent and calculating the Shannon index from it (Magurran, 2004). Therefore, the landscape diversity around a pixel depends on the number of different land use classes around it and on the amount of each class.

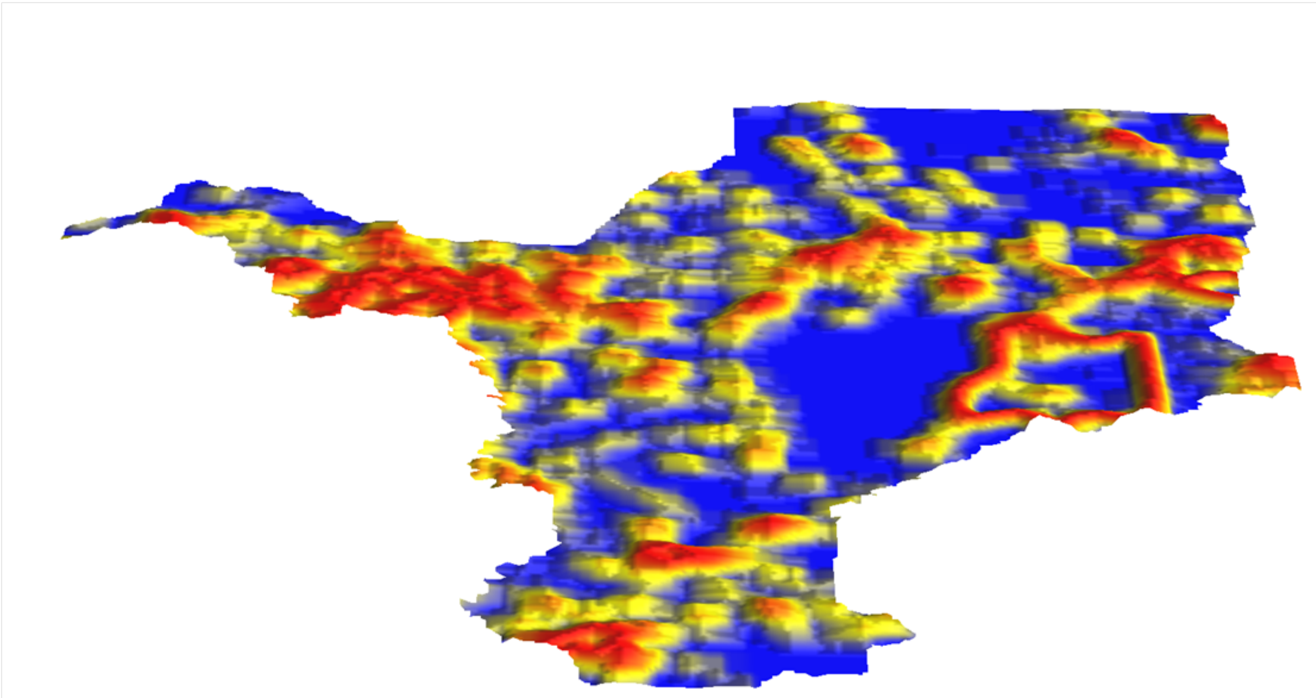


See query results below:

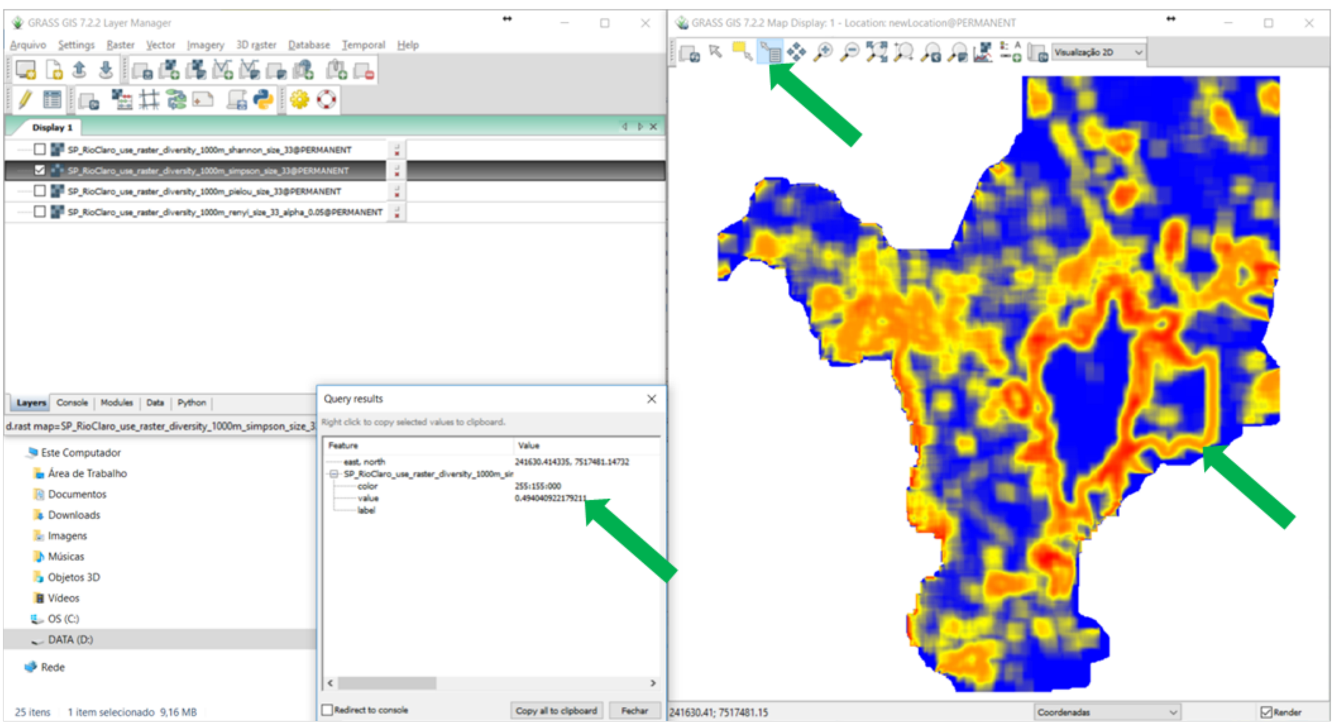


**Simpson´s diversity (Simpson, 1949)**

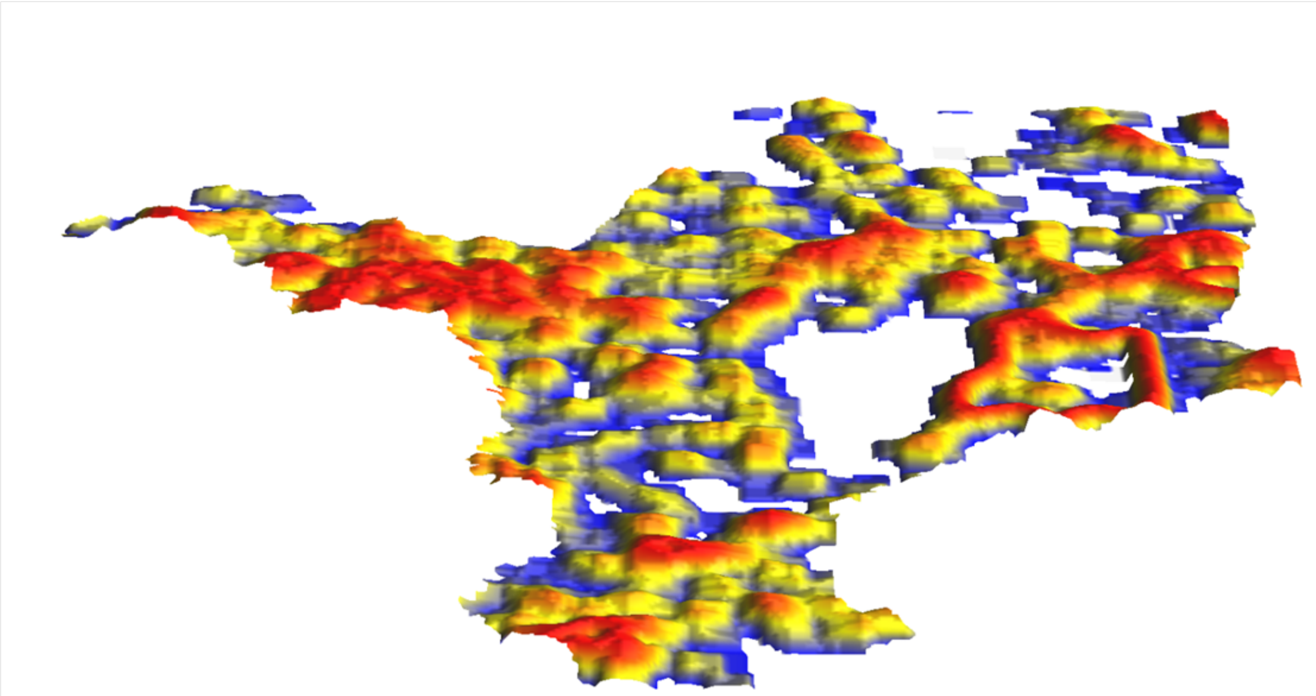
Diversity is based on the probability that two interacting individuals of a population belong to the same species. In our case, 2 patches belonging to the same land use class



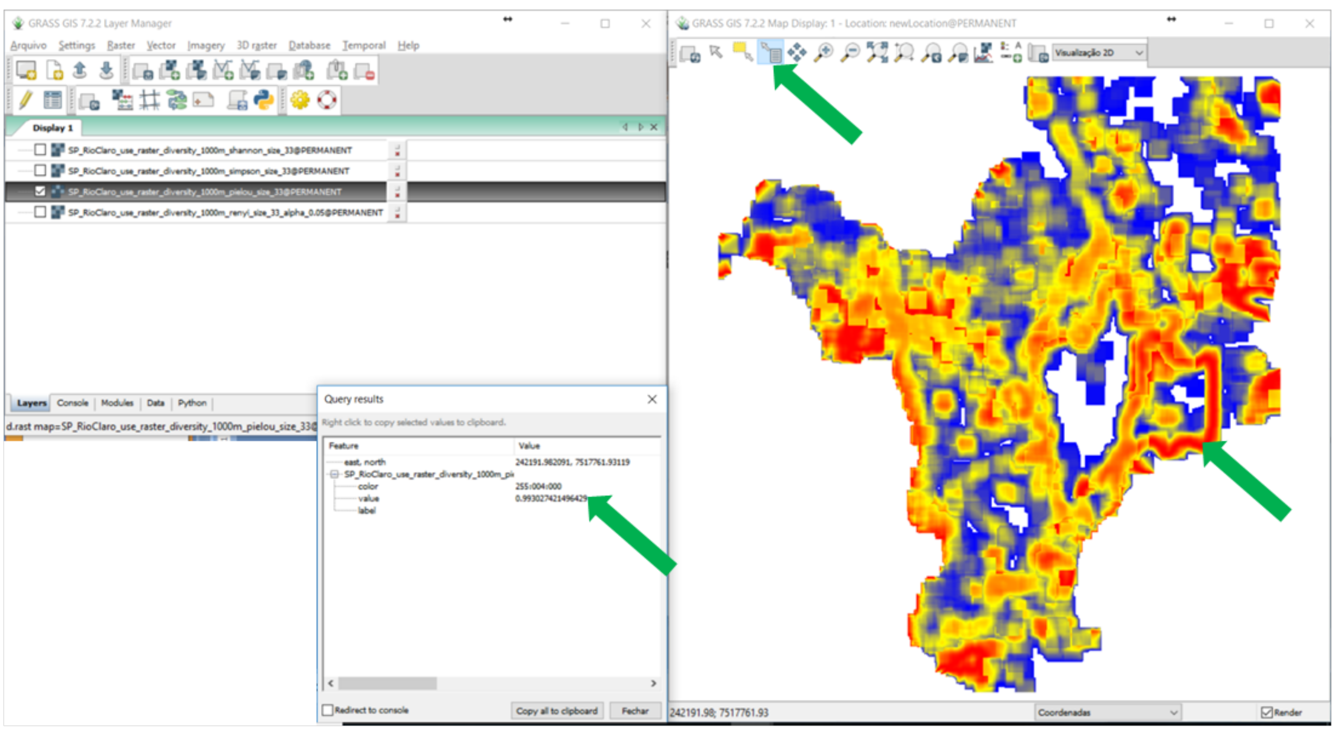
See query results below:



**Pielou´s evenness (Pielou, 1966)**



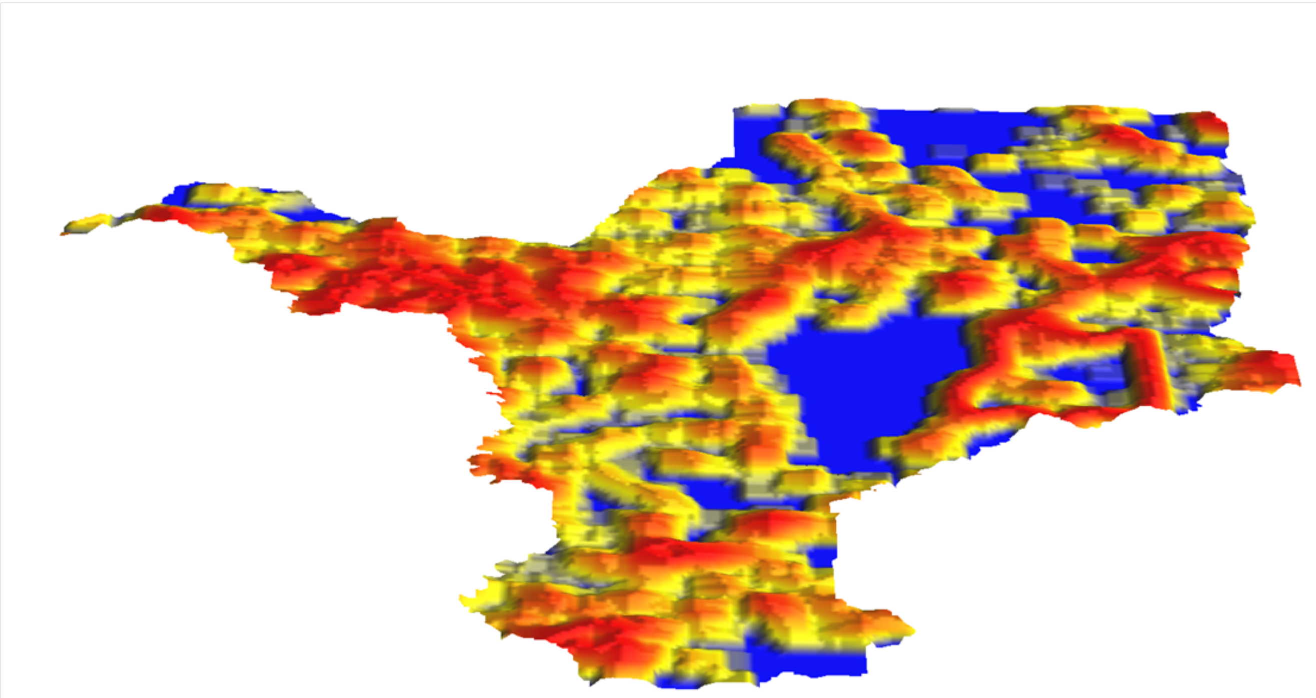
See query results below:



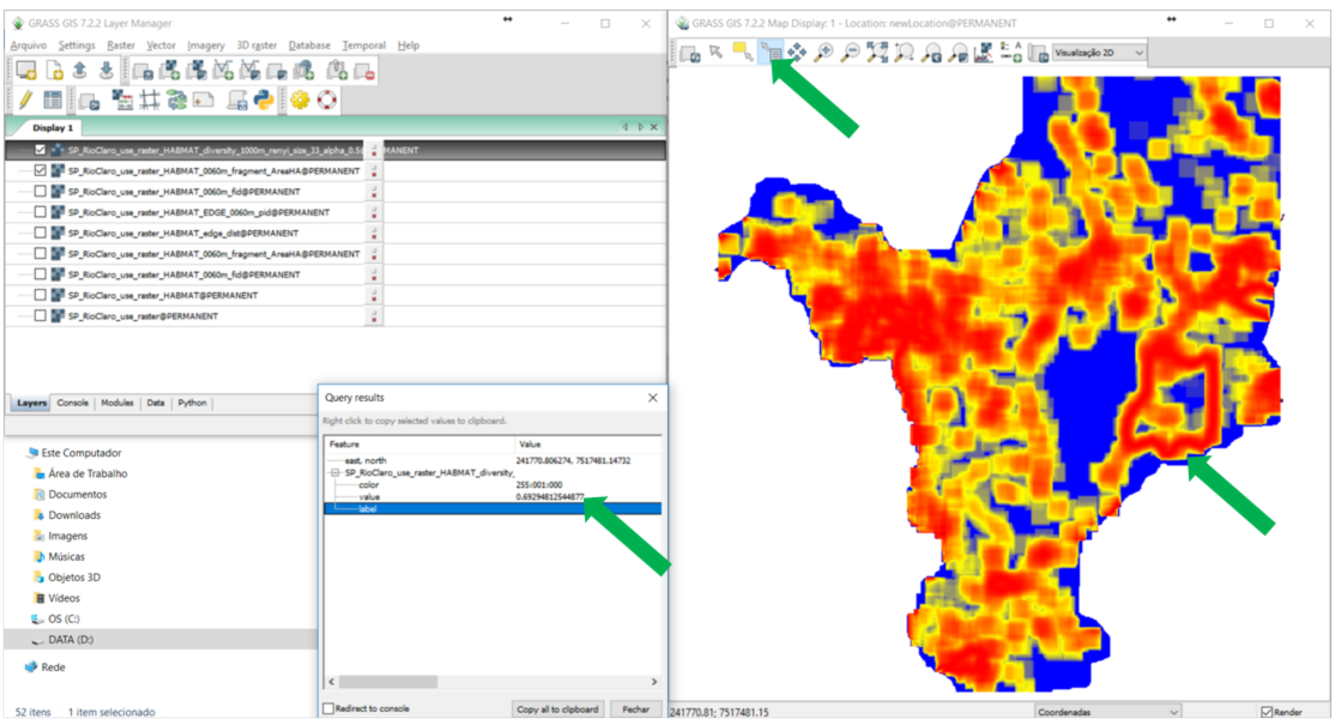

This index compares the measured diversity to the corresponding maximum value in landscape.

**Renyi** (Rényi, 1961)

The Rényi entropy is a generalization of the Shannon´s diversity index. Alpha of example is set as 0.5.



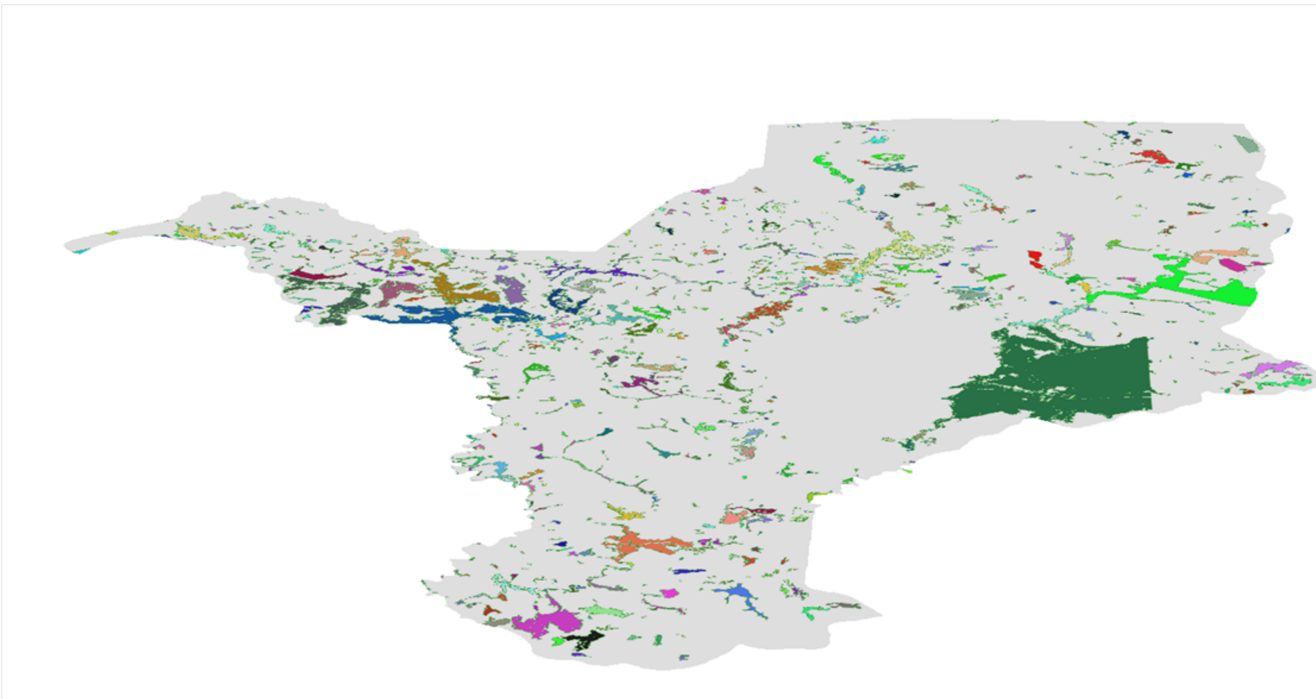
See query results below:



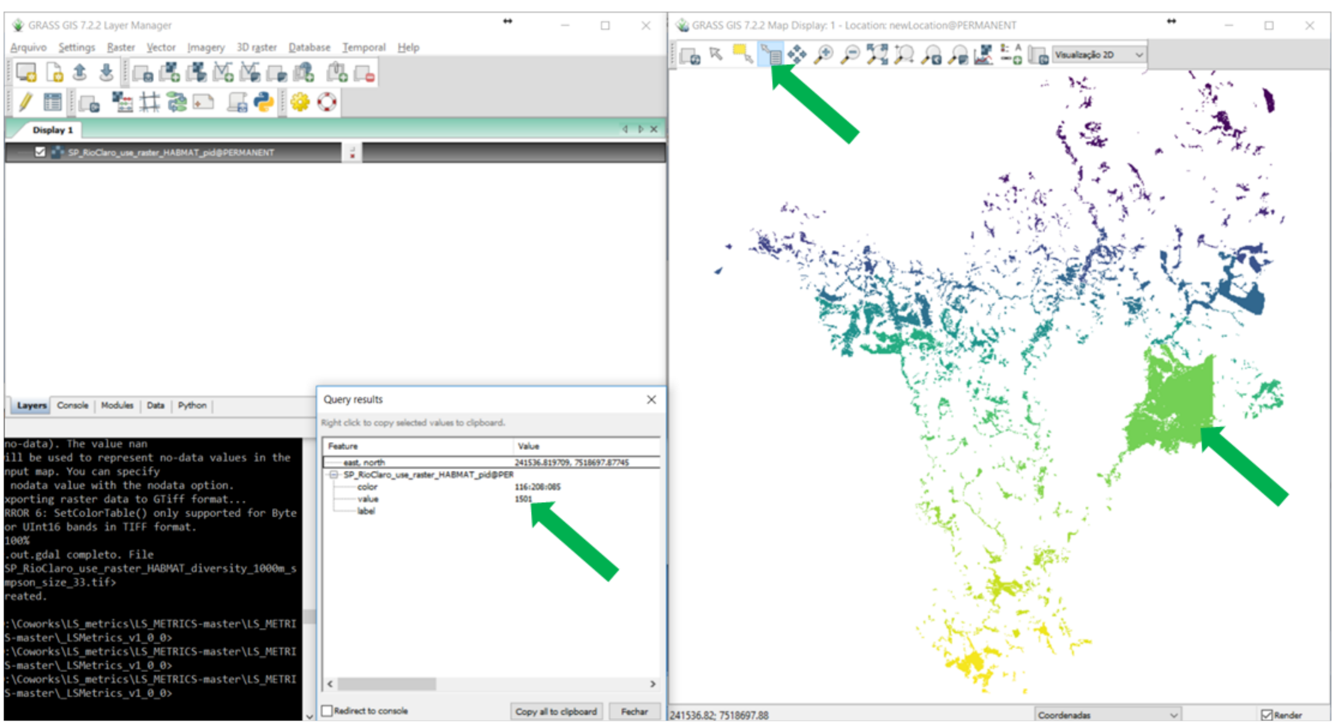
## Patch ID maps

The output files with "pid" in file name are basically the patch or clumps identification. We will use the tool "Query results" in GRASS GIS to show the values attributed for the identification of a patch (pid). Note that the large green patch (The FEENA in Rio Claro) has pid equals to 9112.

Note that only habitat patches will present patch identification in the general pid files (sufix HABMAT_pid)



Query results are below:



## Contact us

LSMetrics was developed at the Spatial Ecology and Conservation Lab (LEEC), at Universidade Estadual Paulista (UNESP), Rio Claro, SP, Brazil. If you have questions, contact us at one of the e-mails below. You can also contact us if you have any errors running LSMetrics (or you can open an issue here on GitHub).


- Bernardo Niebuhr <bernardo_brandaum@yahoo.com.br>
- Milton C. Ribeiro <mcr@rc.unesp.br>
- Felipe Martello <felipemartello@gmail.com>


## Contribute

You are also very welcome to contribute your own changes or new metrics to the LSMetrics tool. We are also open to dicuss the idea and how to implement other metrics needed for environmental research as well as for conservation and nature management purposes. Please submit a pull request or get in contact with us.

## About LEEC lab

The LEEC lab team has developed several tools for spatial and landscape analysis:

**LSMetrics**

**LSCorridors**

**Landscape analysis in R**

**Other publications**

### Take a look in our website!