GRASS GIS 7 Capabilities

An overview

Markus Neteler

http://gis.cri.fmach.it/neteler/
http://courses.neteler.org/blog
http://consulting.neteler.org/

2015

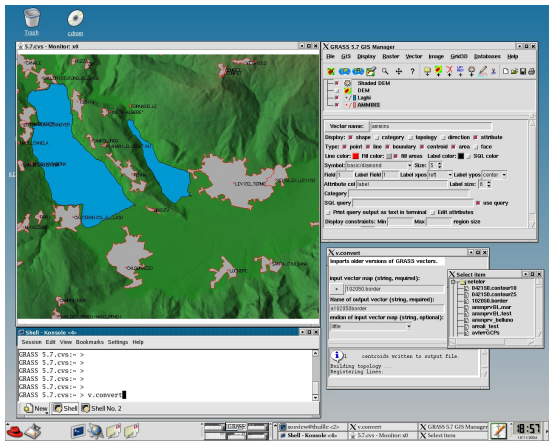
What's GRASS GIS?

http://grass.osgeo.org

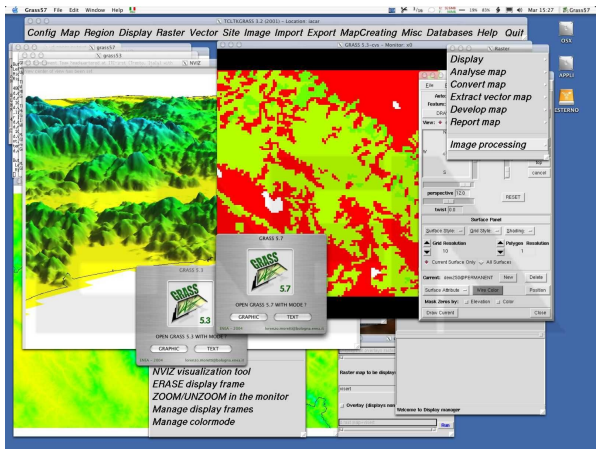
- **G**eographic **R**esources **A**nalysis **S**upport **S**ystem
- **Open Source GIS**, developed since 1984, since 1999 GNU GPL
- **Portable code** (many operating systems, 32/64bit)
- Your **GIS backbone** – linkable to:

GDAL, Qt, R, SEXTANTE, ParaView, ZOO (http://zoo-project.org), PyWPS

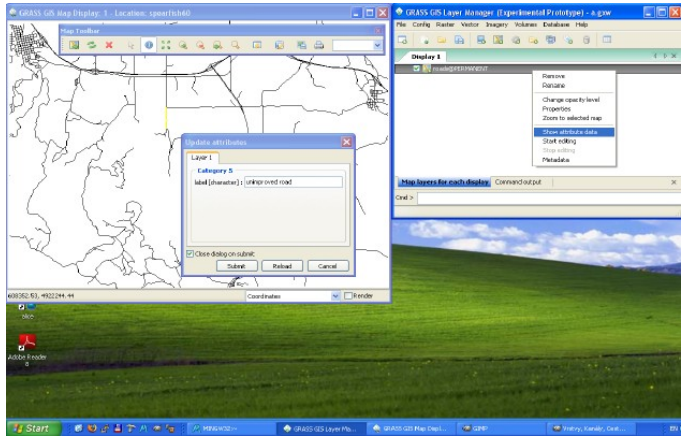
GNU/Linux



MacOSX

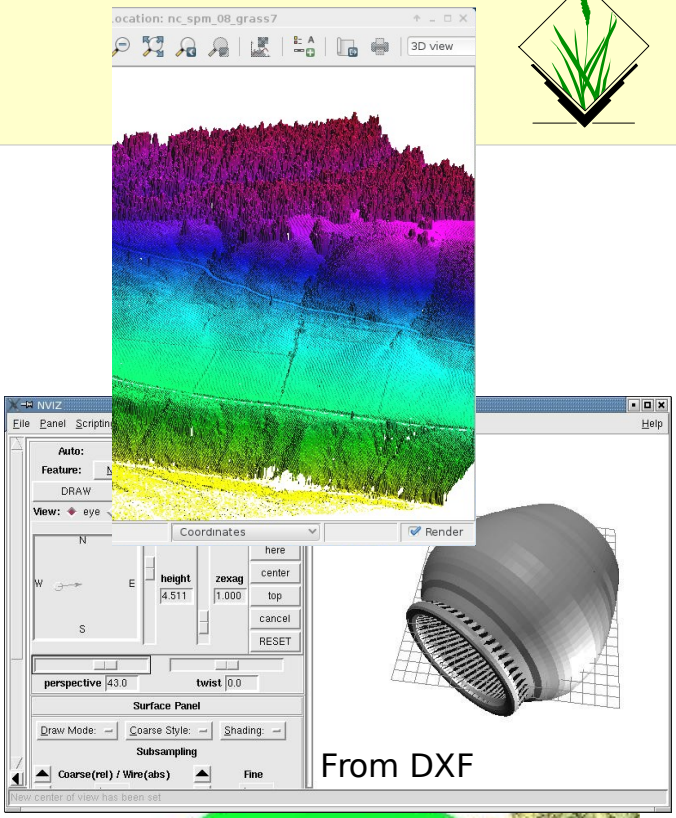


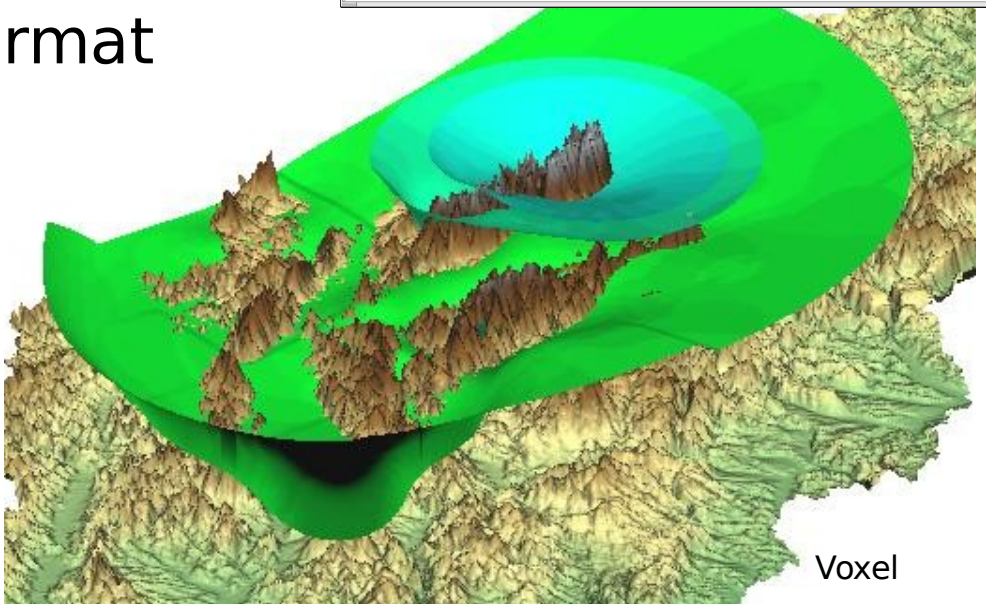
MS-Windows (IBM AIX, *BSD, ...)



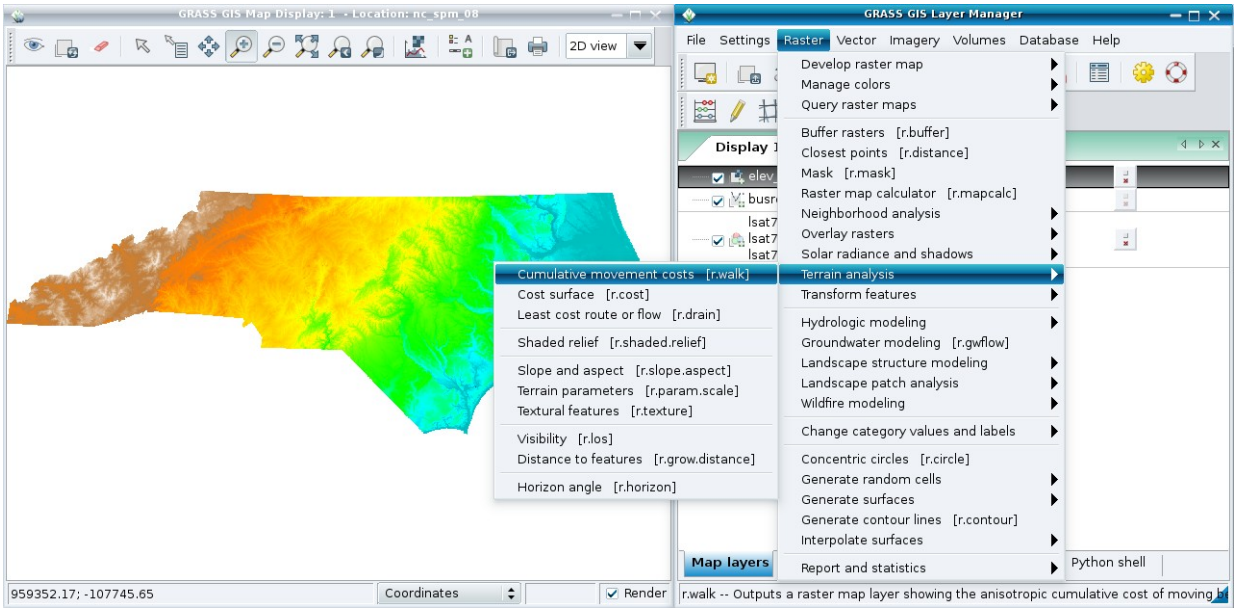
What's GRASS GIS?

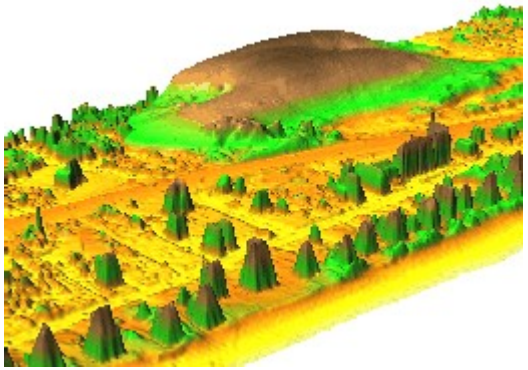
- Raster 2D/3D (voxel) processing
- Vector 2D/3D topological processing
- Vector network analysis support
- Image processing system
- Space-time cubes, temporal GIS
- Native raster and vector format
- 3D Visualization system
- DBMS integrated (SQL) with SQLite, DBF, PostgreSQL, MySQL, and ODBC drivers



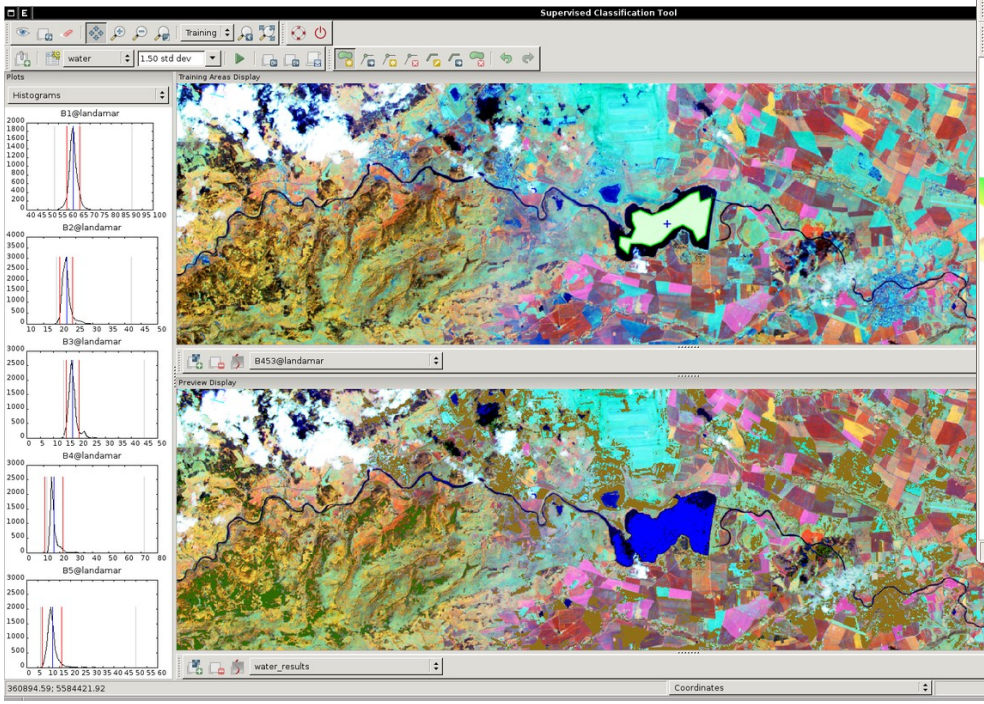


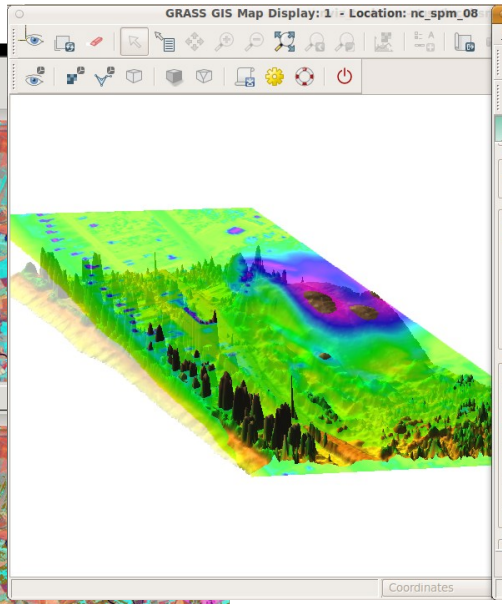
GRASS GIS 7 User interface

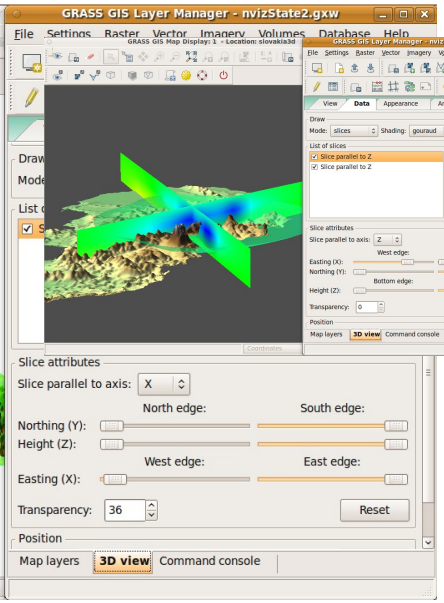




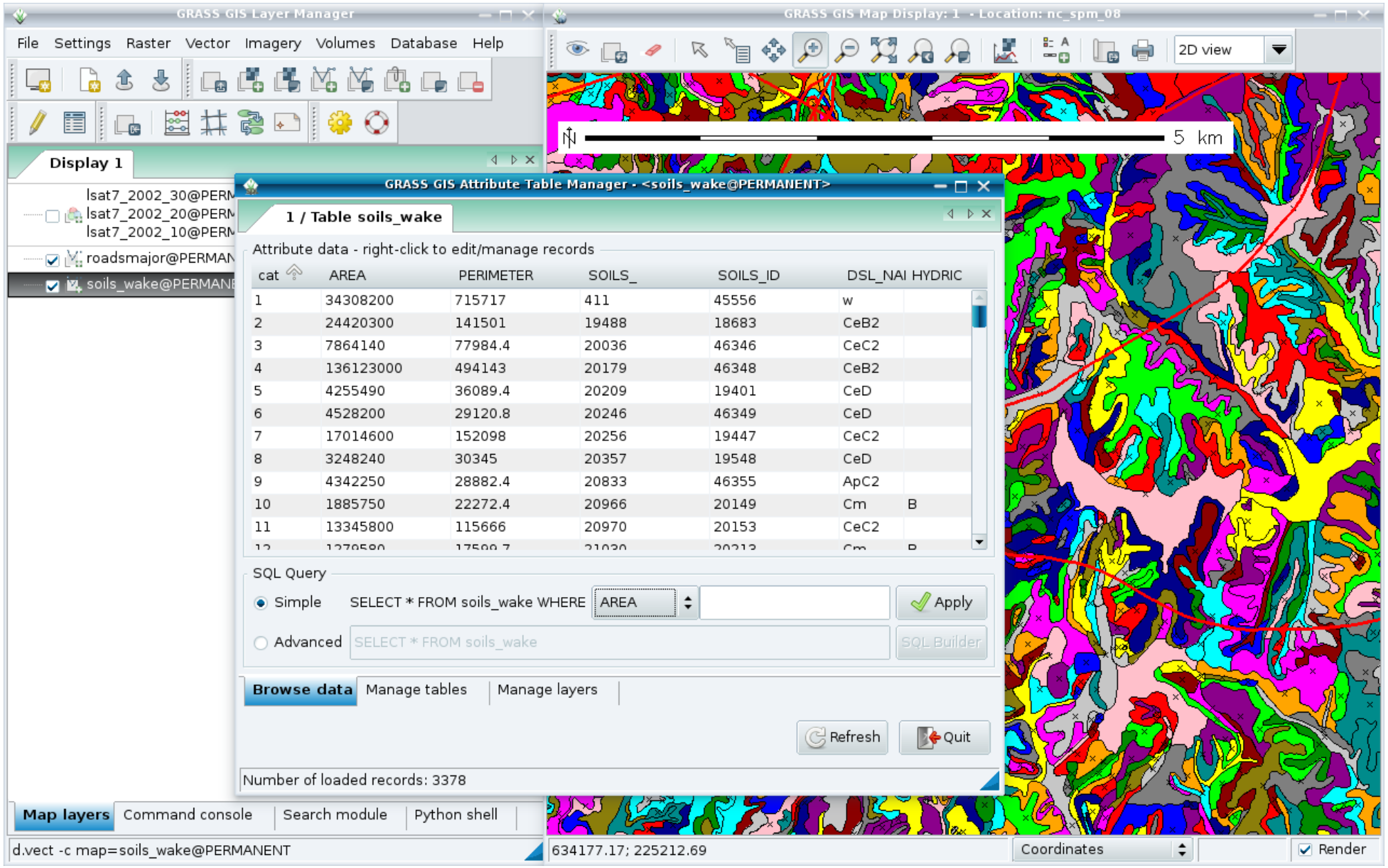
Nagshead LiDAR time series: dune moving over 9 years (NC, USA) – animation





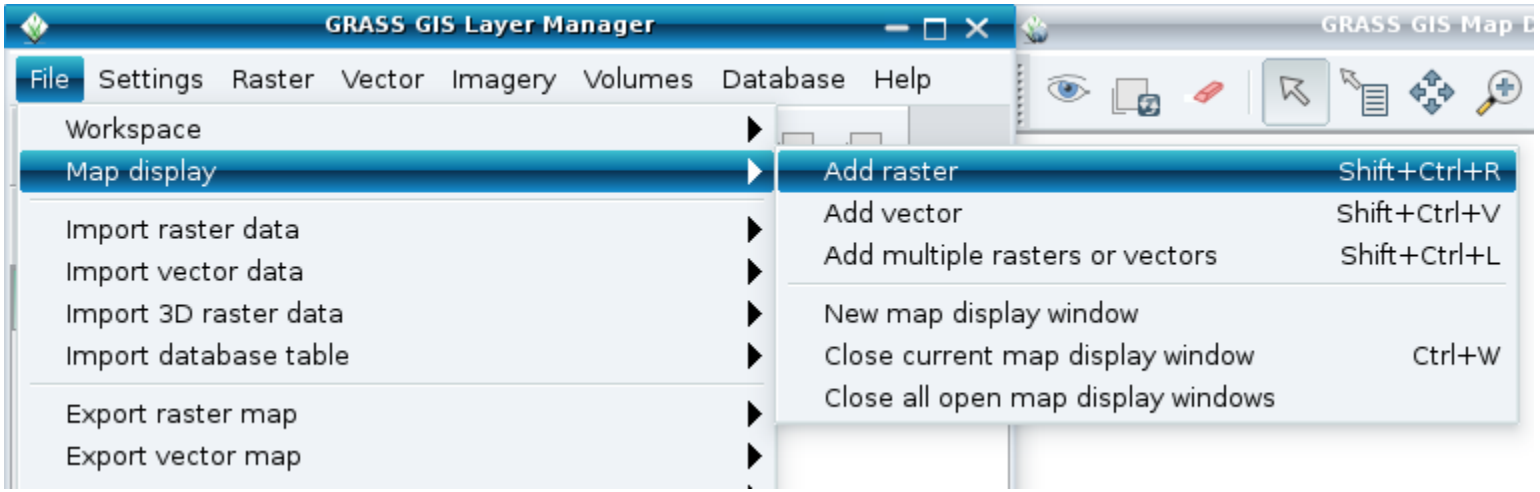


GRASS GIS 7 graphical user interface



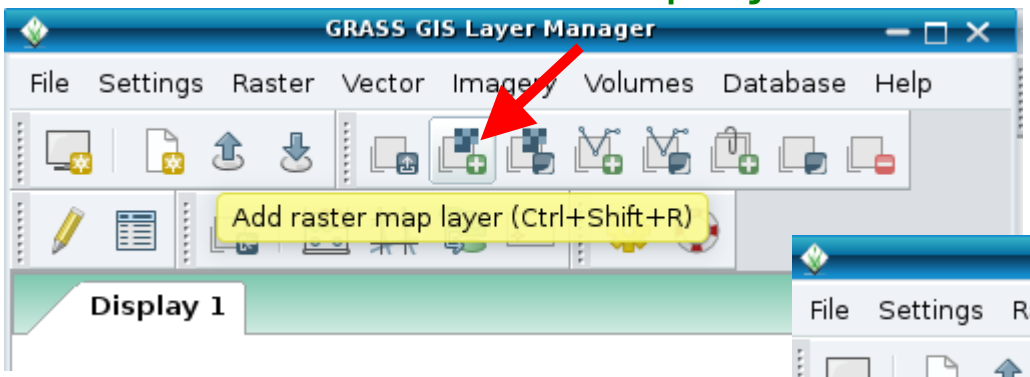
Displaying raster and vector maps

A) Using the menu

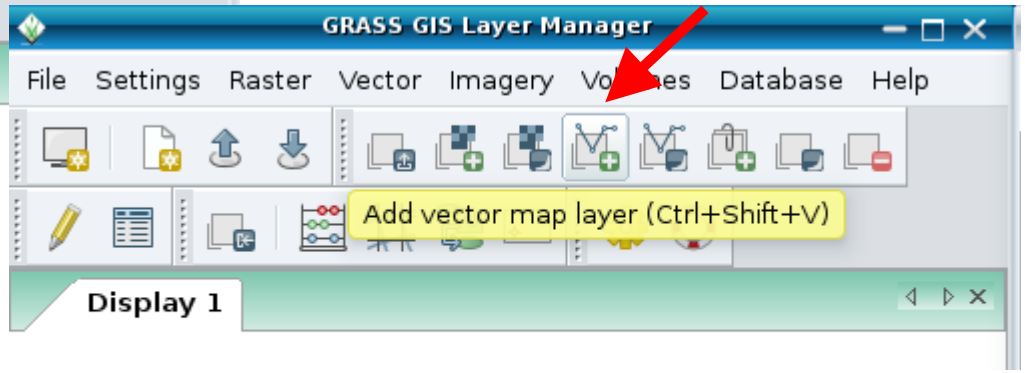


B) Using the icons

Display raster maps



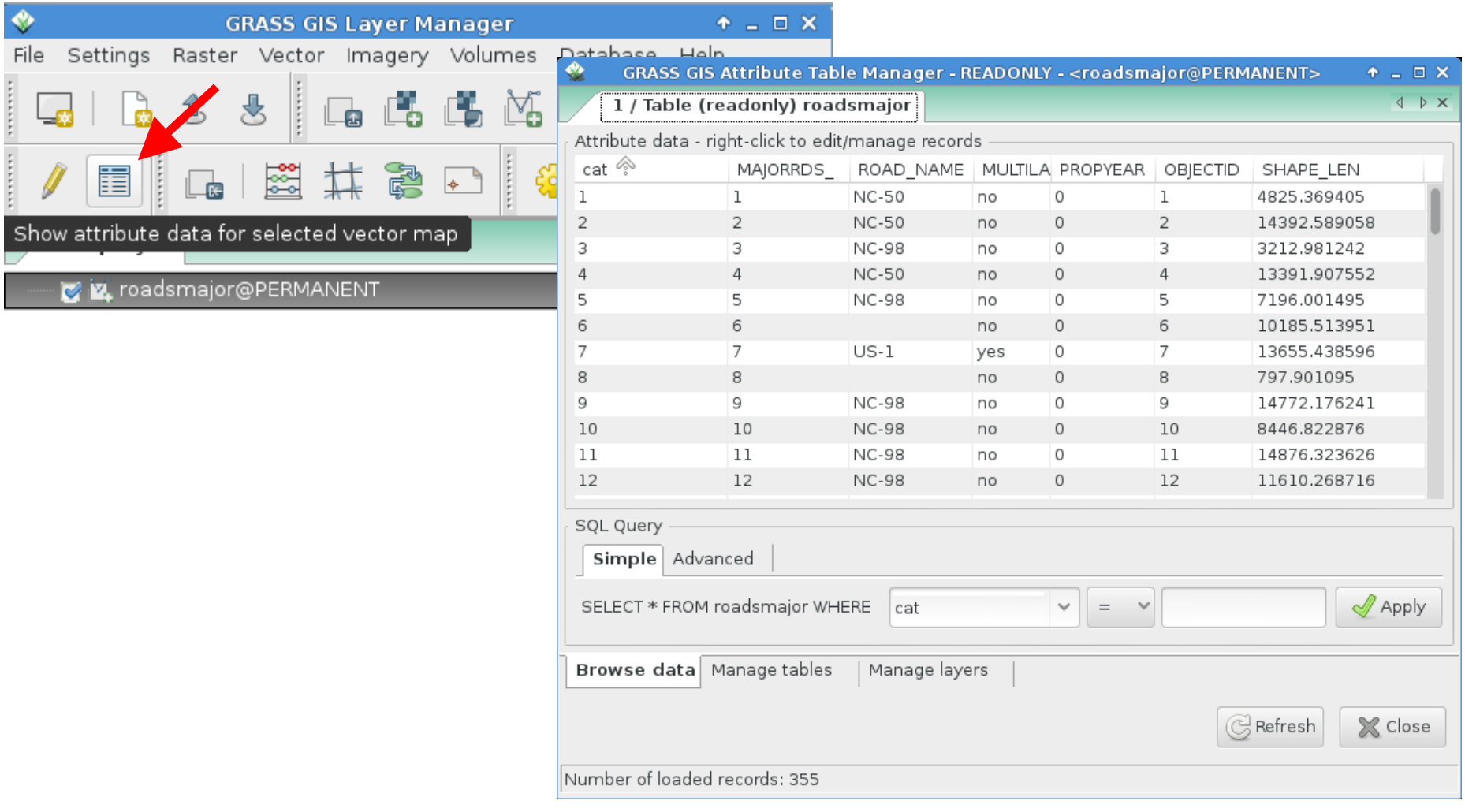
Display vector maps



Showing vector map attributes

Opening the attribute table of the “roadsmajor” vector map by

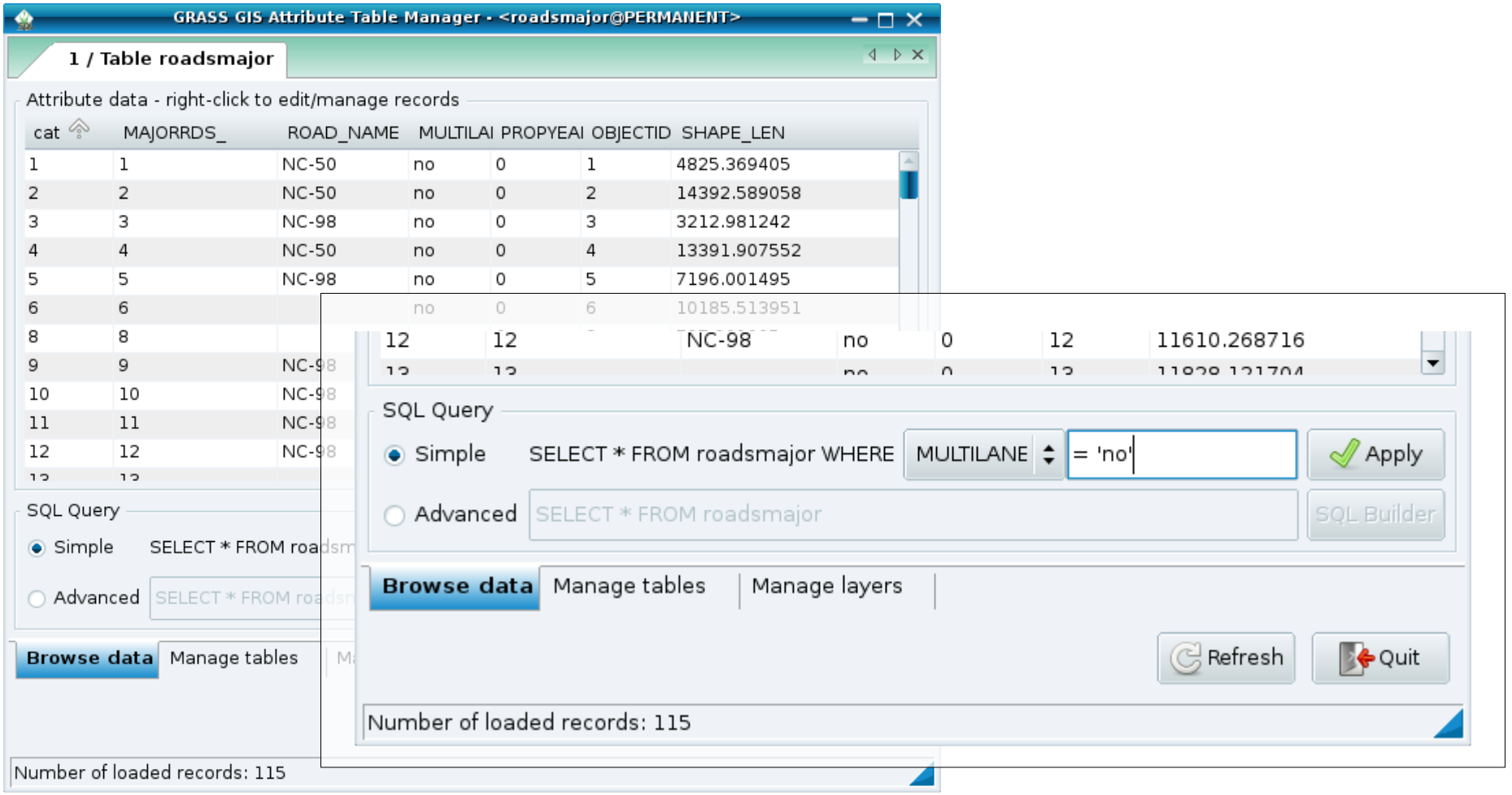
- … either *right-mouse* clicking in layer tree on map name
- … or using the related “Show Attribute table” icon



SQL queries of attributes

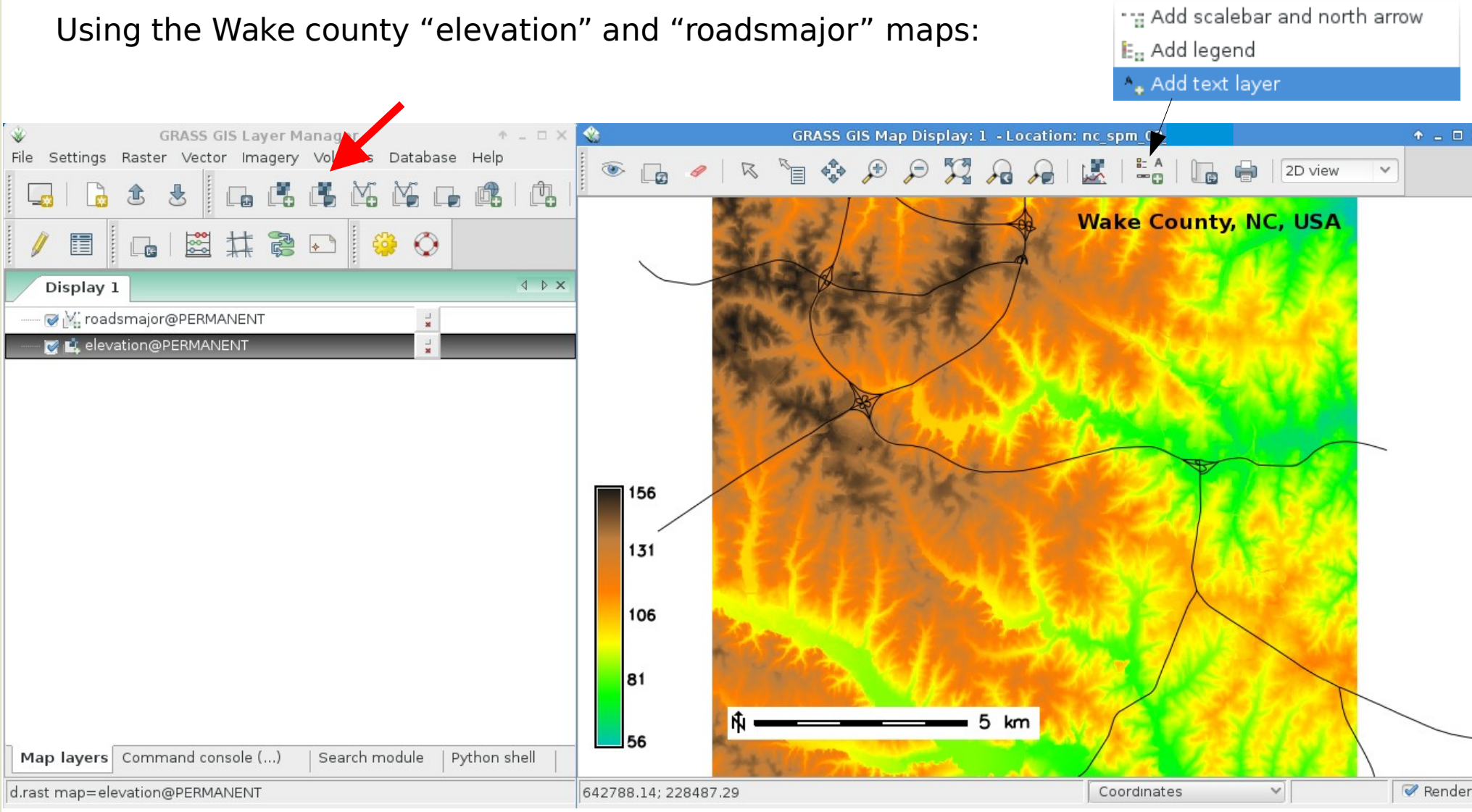
Selecting the **single lane roads** in the “roadsmajor” vector map

- Use “Simple” SQL query
- The selected vectors will be highlighted in the map display



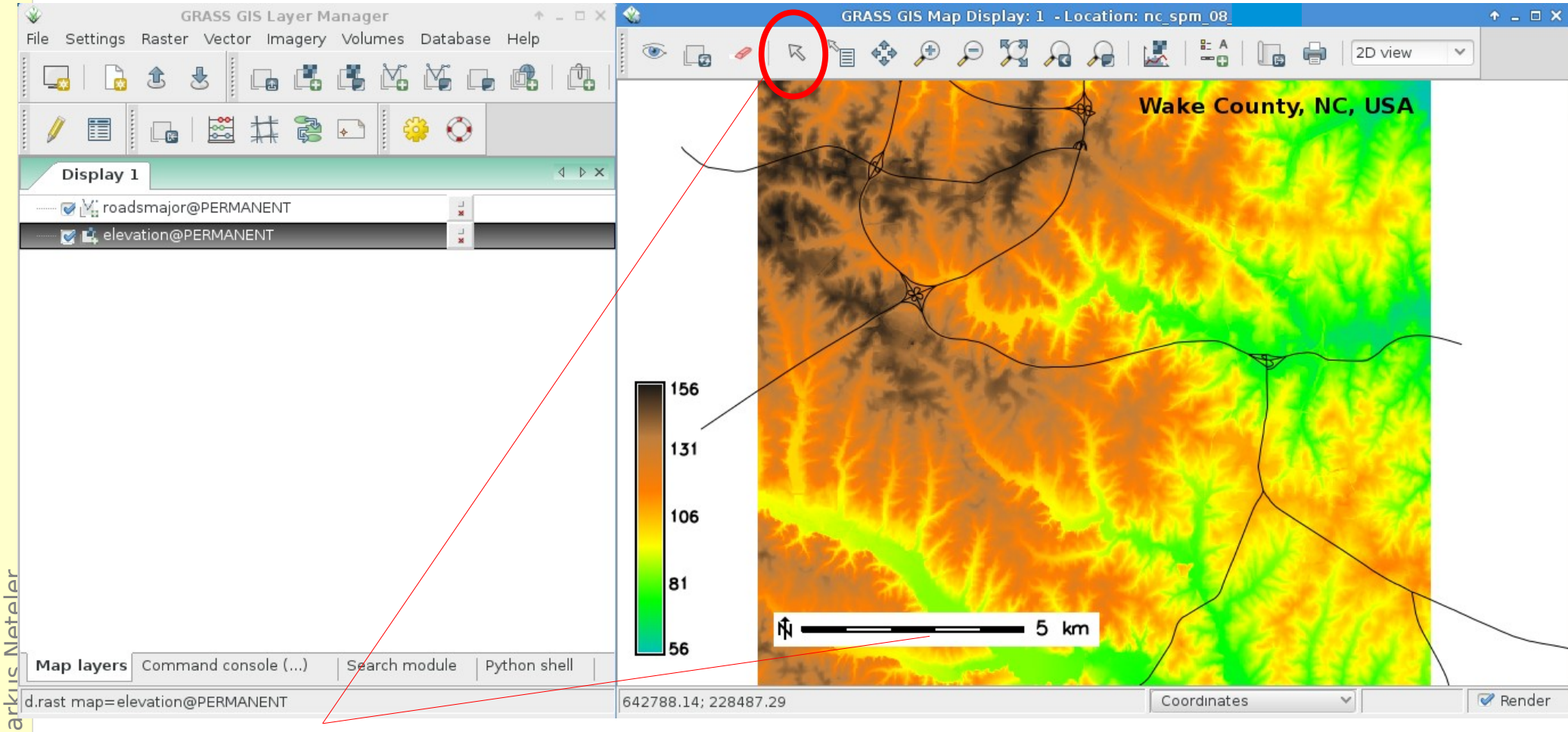
Adding map elements

Using the Wake county “elevation” and “roadsmajor” maps:



Modifying element settings and position

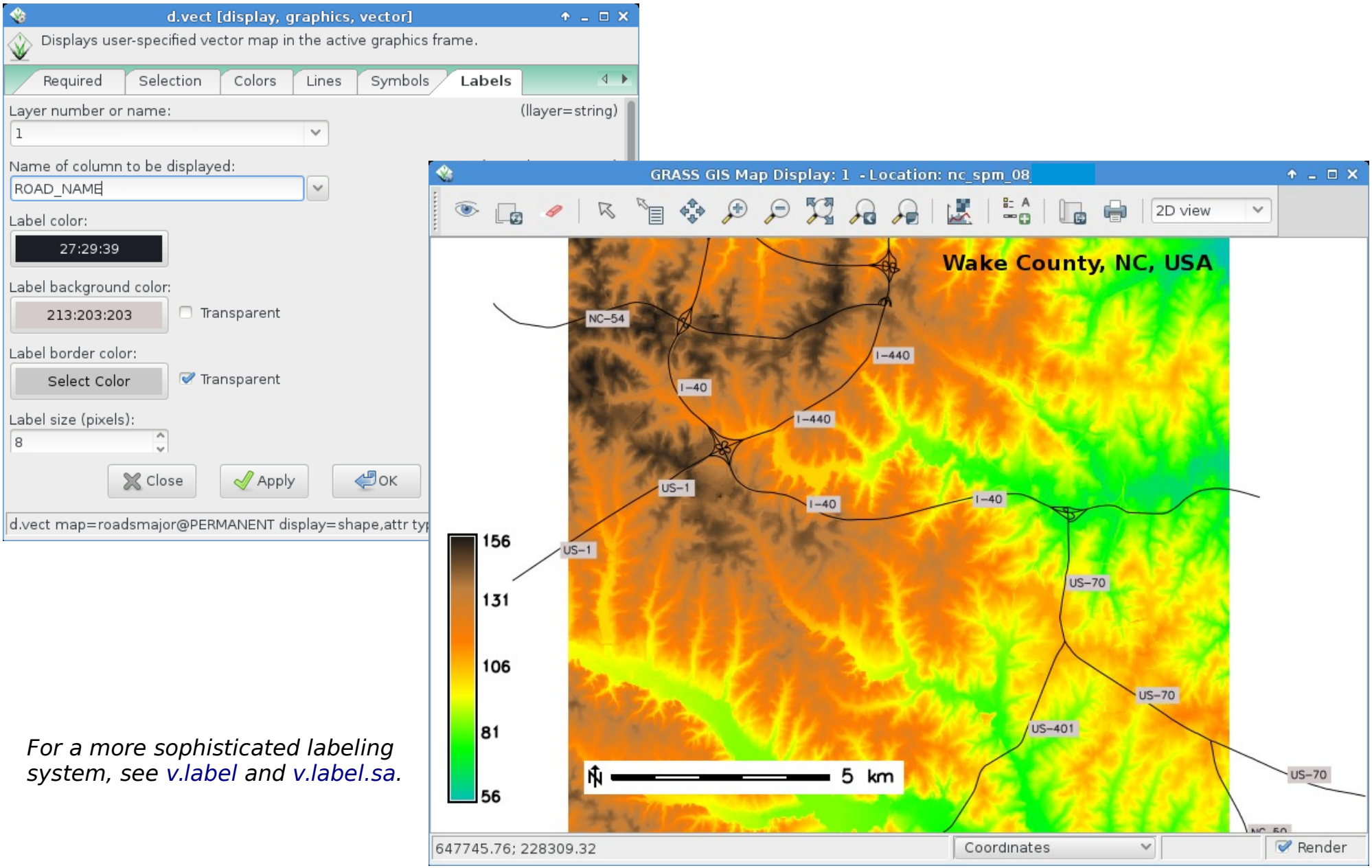
Using the Wake county “elevation” and “roadsmajor” maps:



Use pointer to
- move map elements
- edit element settings with a click

Adding vector map labels

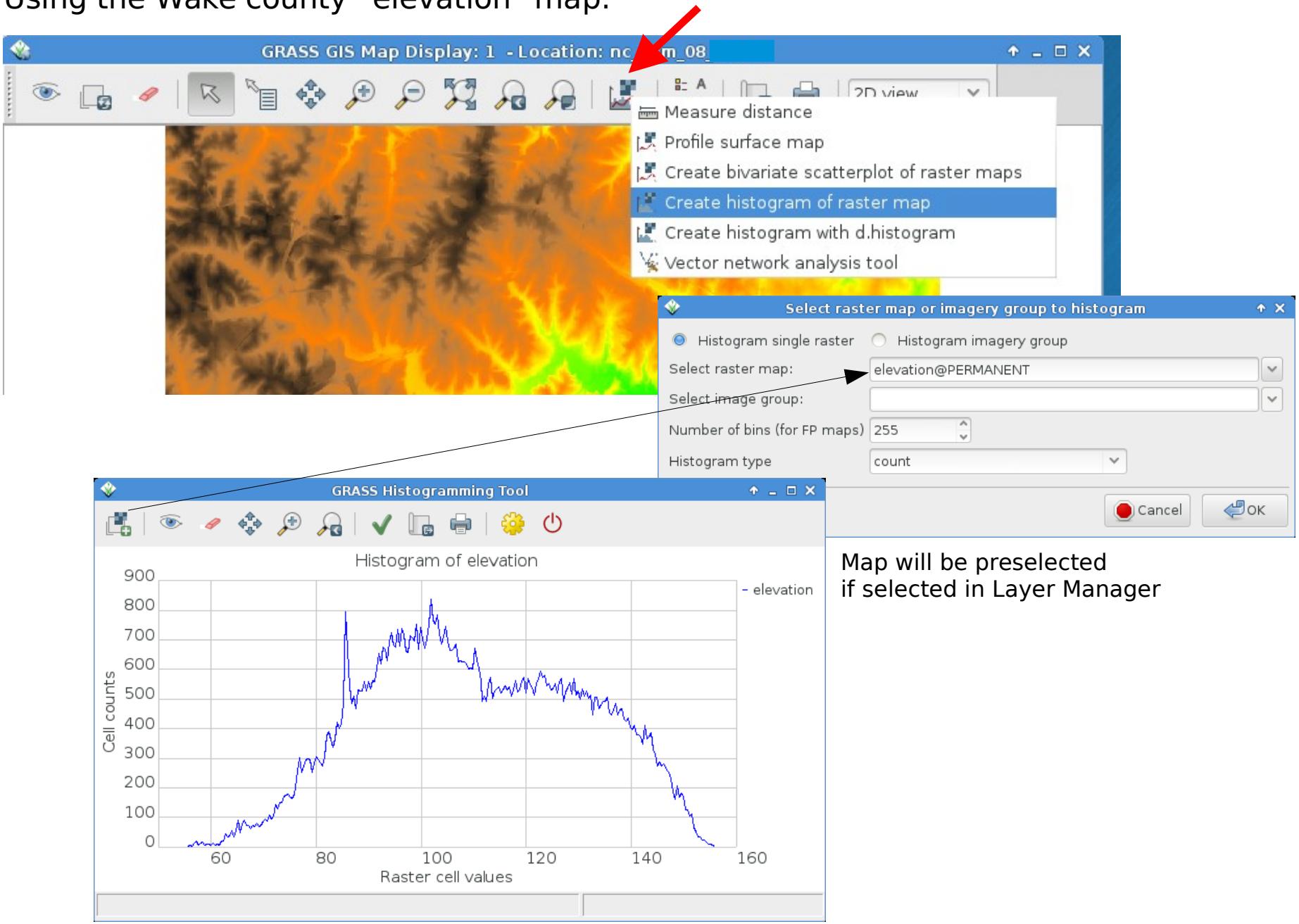
Activating vector labels for “roadsmajor” map: right-click map in Layer Manager, Properties



For a more sophisticated labeling system, see v.label and v.label.sa.

Map histogram tool

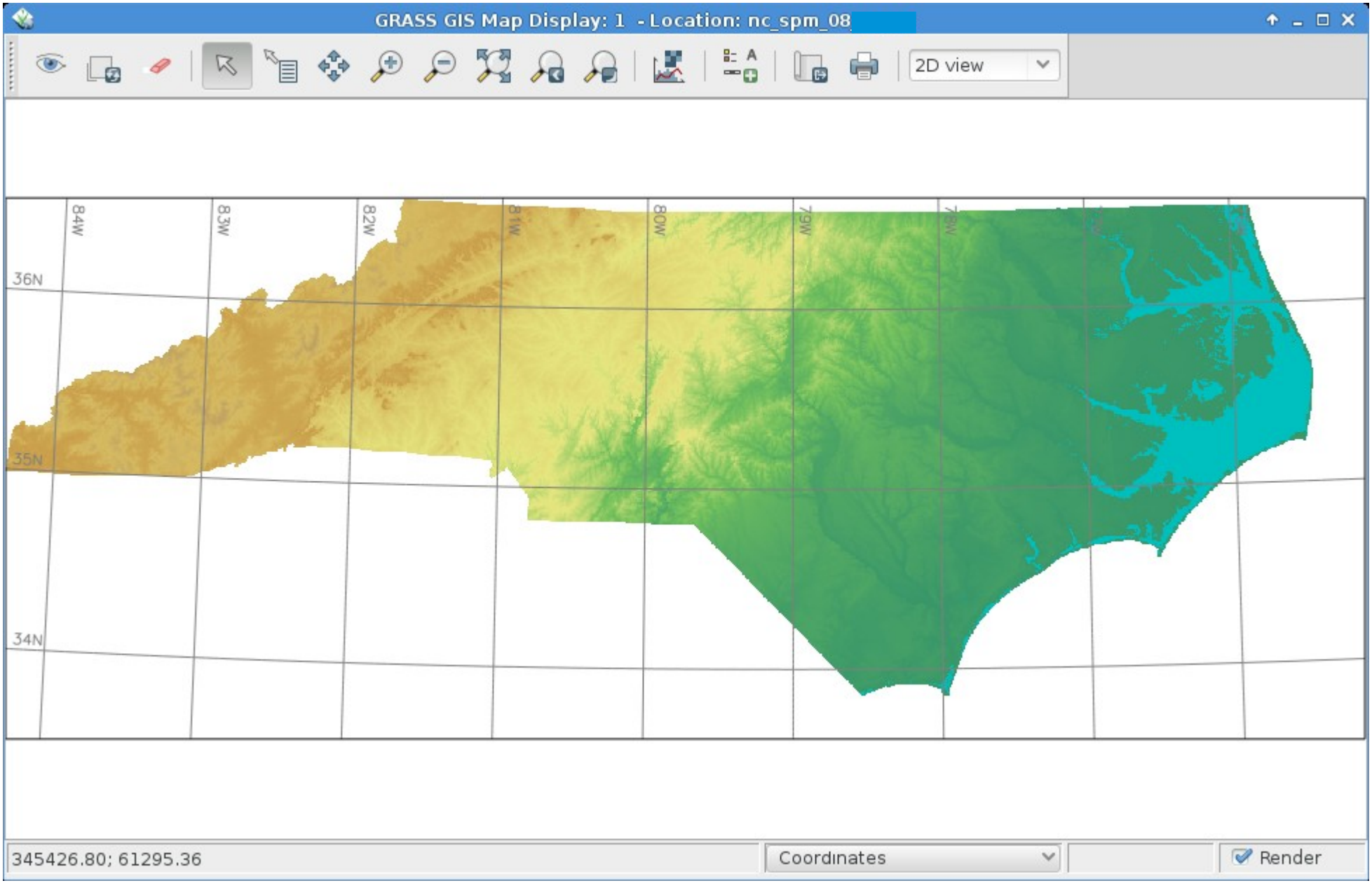
Using the Wake county “elevation” map:



Map will be preselected
if selected in Layer Manager

Adding a grid to the map

Using the NC state “elev_state_500m” map:



GRASS GIS 7 Vector features

Native vector format

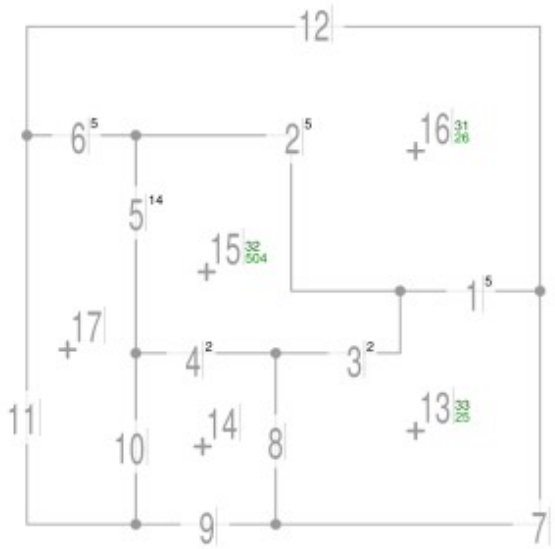
- Vector topology
- m:n mapping of geometry features to attributes
- Support of vector layers
- OGC Simple Features ←→ Topological Vector Conversion
- Database Management system (DBMS) with SQL support
- SQLite (default DB backend), PostgreSQL + PostGIS, MySQL, ODBC (, DBF)

GRASS GIS 7 Vector features

geometry

attribute

map features :



boundary

node

centroid

id* layer1
layer2
layer3

*every object on the map is automatically assigned an internal feature-id

paths

cat	name
2	private
5	public
3	unknown

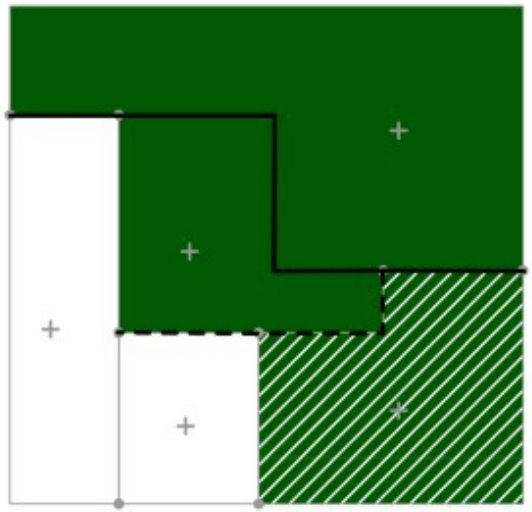
plots

code	crop
31	wheat
32	wheat
33	oat
34	barley

map is connected to :

- table **paths** by layer1 with key **cat**
- table **plots** by layer2 with key **code**

map representation :



public path

private path

wheat

oat

layer3 may not refer to any table

database may contain a table **owner** which is not related to any layer

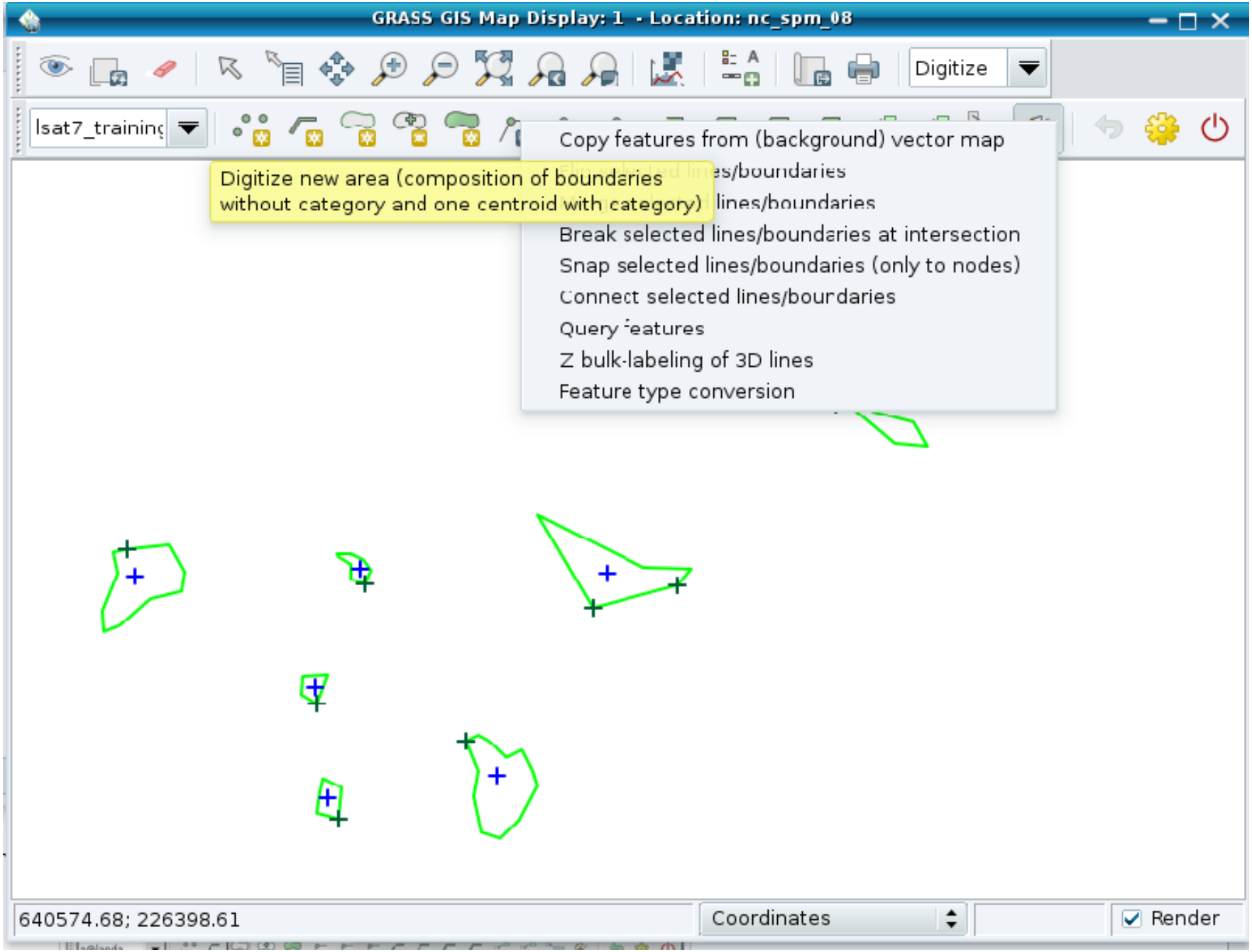
owner

cat	name
25	...
26	...
504	...

GRASS GIS Software
http://grass.osgeo.org

http://grasswiki.osgeo.org/wiki/Vector_Database_Management

GRASS GIS topological vector digitizer



Further vector processing capabilities

Example vector module groups

Topological geometry feature digitizing/editing

https://grass.osgeo.org/grass70/manuals/vectorintro.html

LiDAR analysis:

http://grasswiki.osgeo.org/wiki/LIDAR

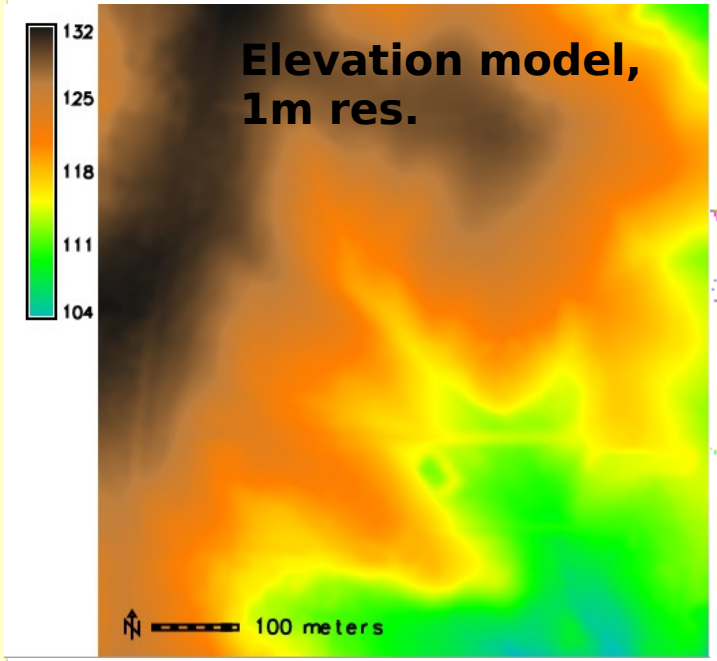
Linear referencing (LRS) – v.lrs.*:

http://grasswiki.osgeo.org/wiki/Linear_Reference_System

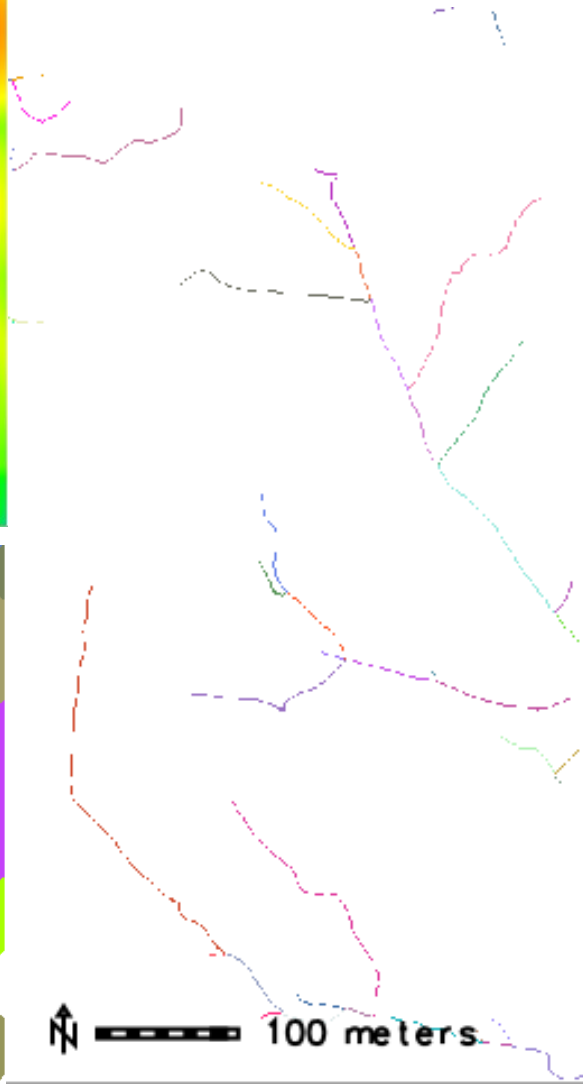
Network analysis – v.net.*:

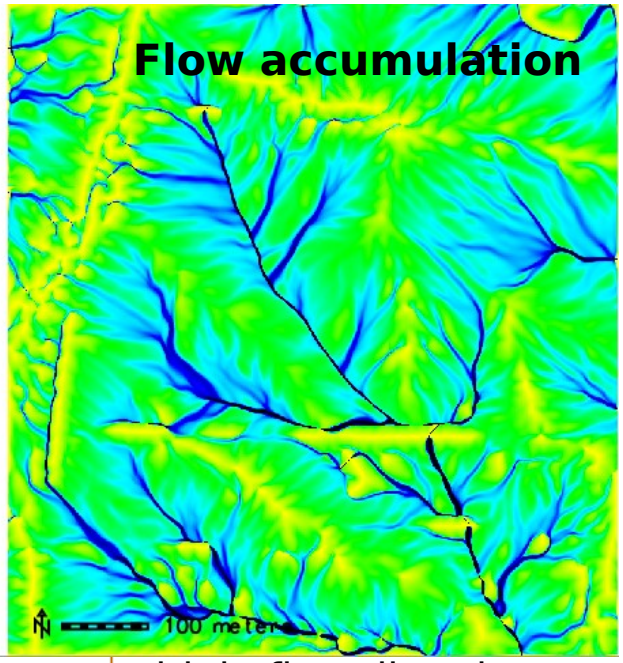
http://grasswiki.osgeo.org/wiki/Vector_network_analysis

Hydrological analysis: examples

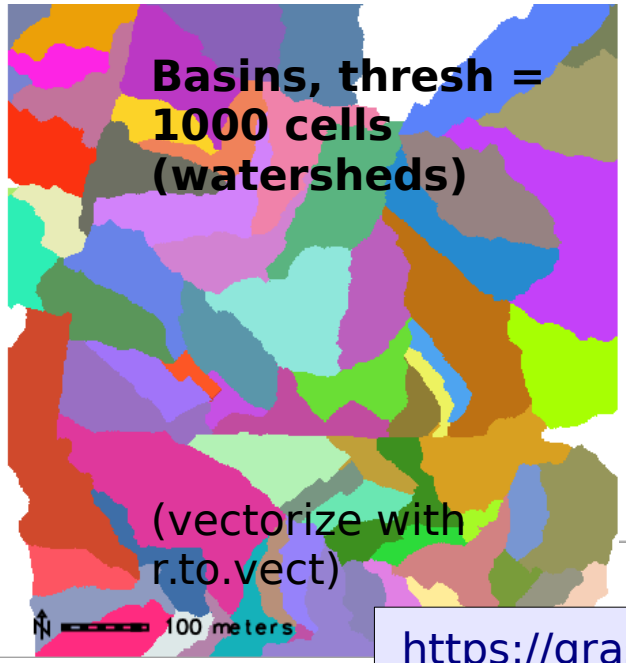


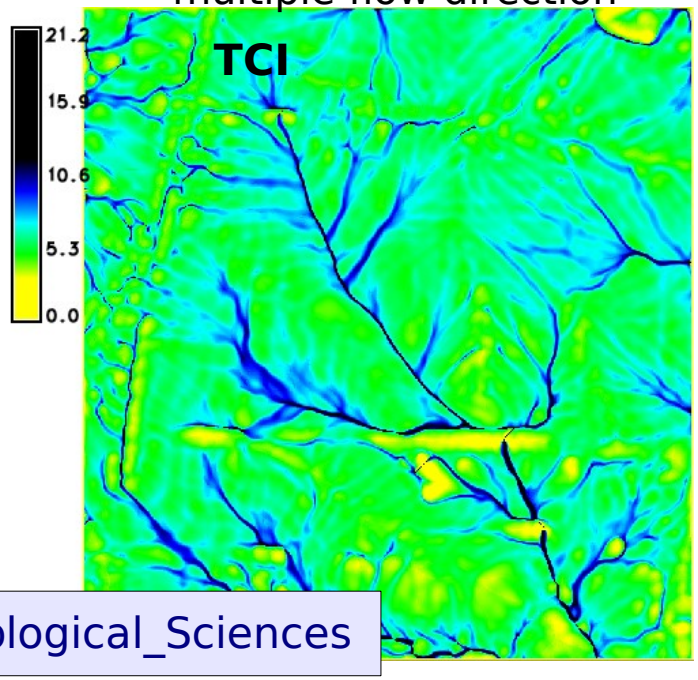
Stream network





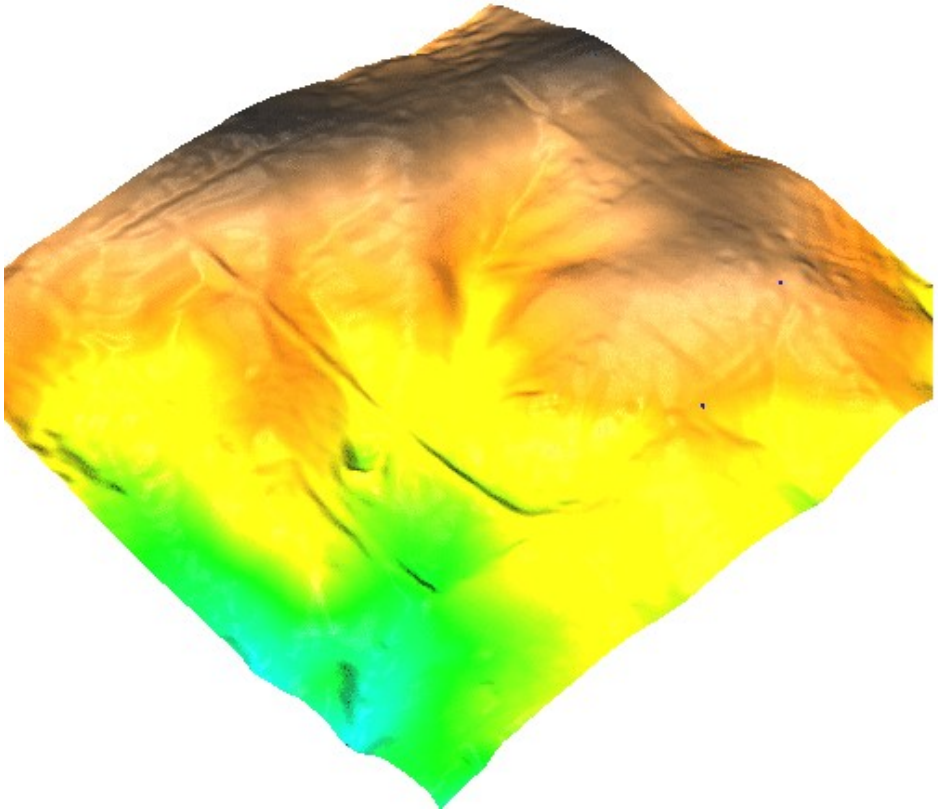
multiple flow direction



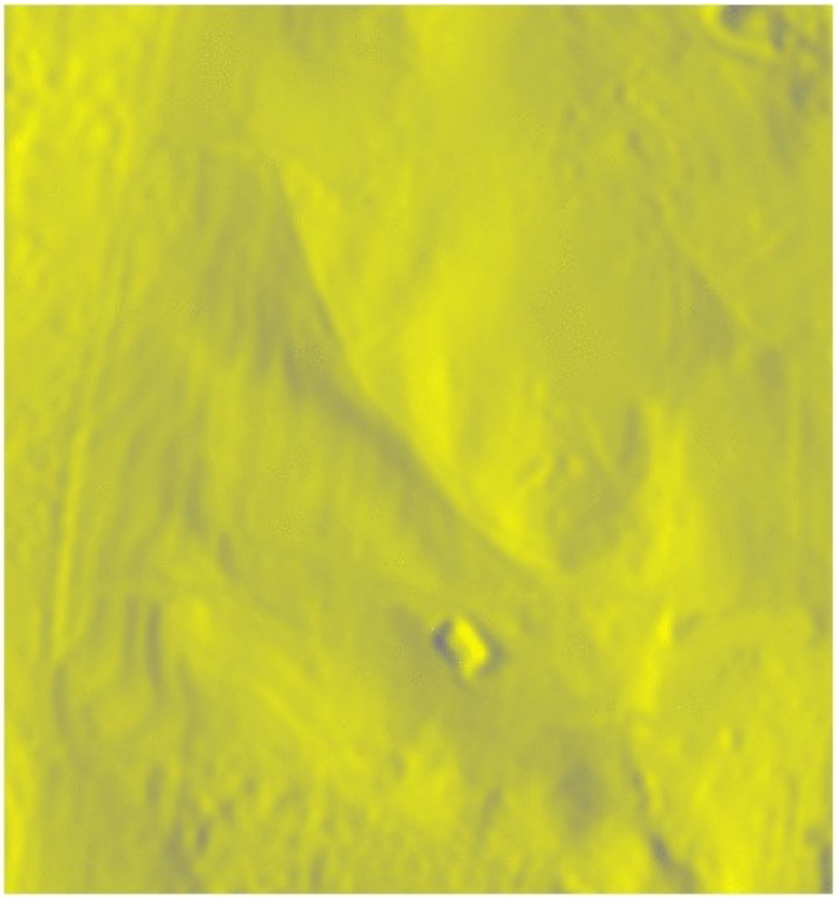


https://grasswiki.osgeo.org/wiki/Hydrological_Sciences

Hydrological analysis: examples



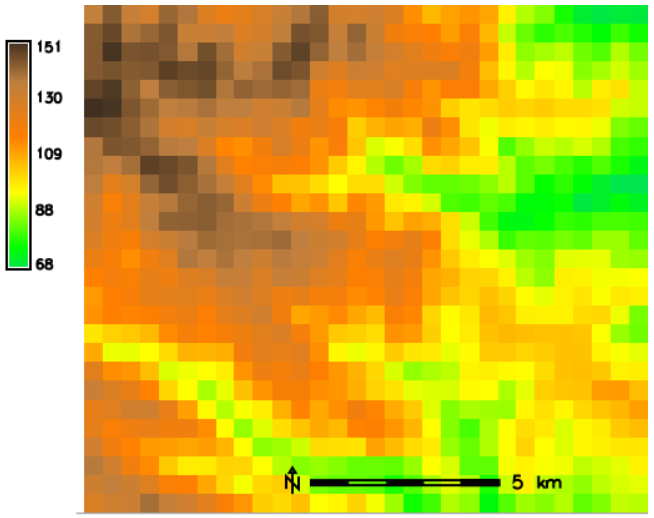
Particle flow over time (animation)



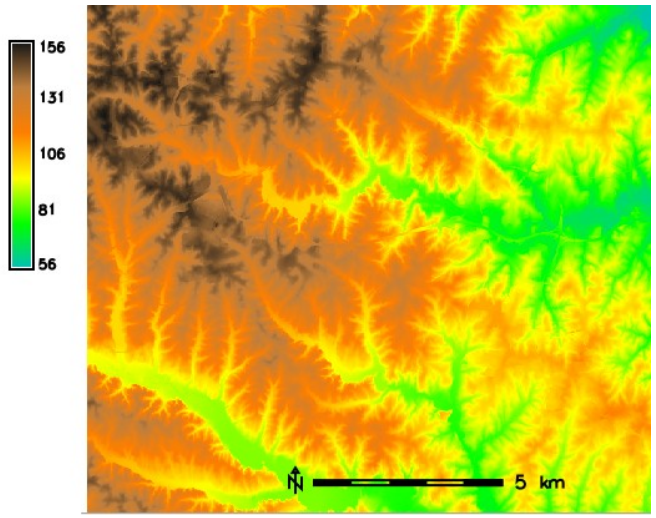
Accumulation over time (animation)

Credit:
Helena Mitasova, NCSU (web)

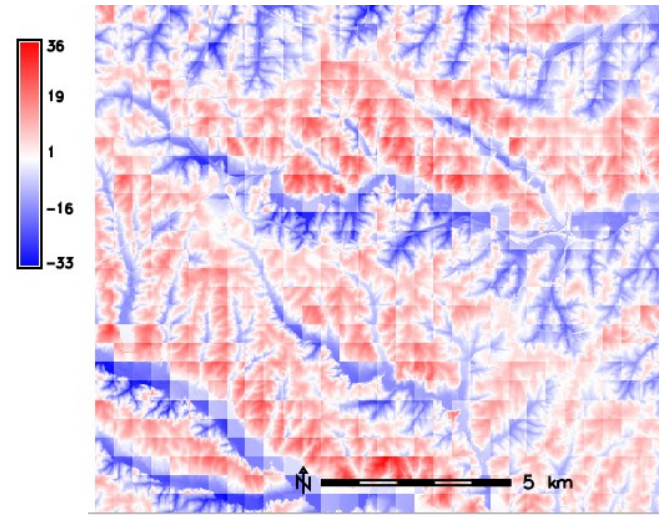
Raster resampling with interpolation



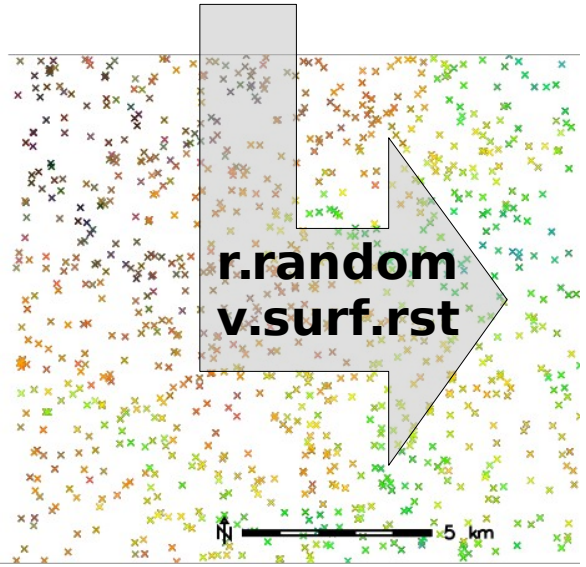
elev_state_500m (orig)



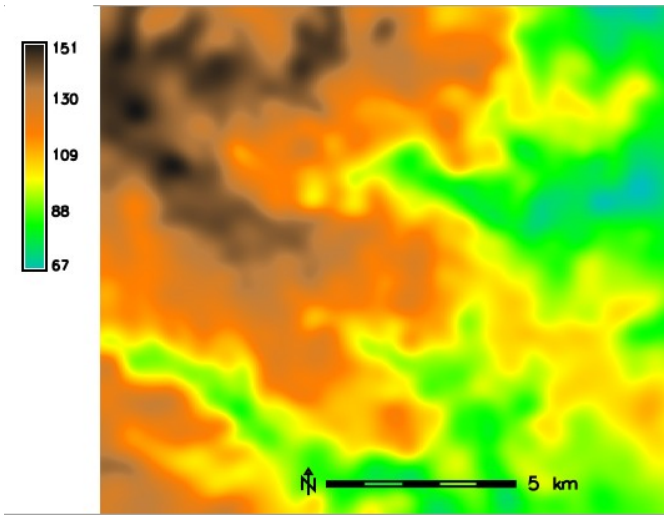
elevation 10m (orig)



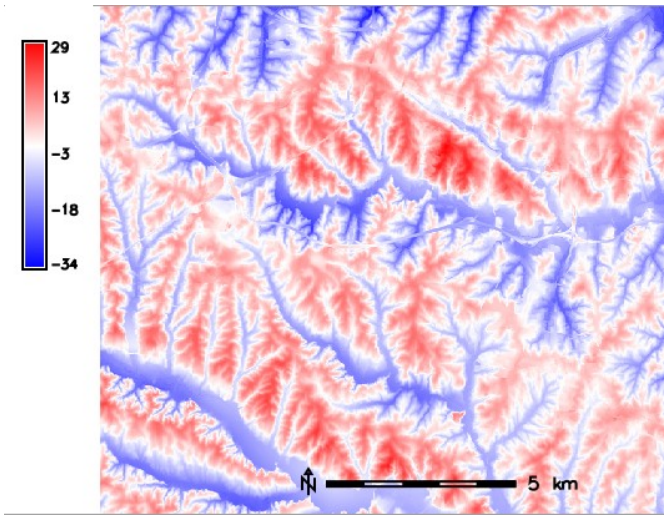
differences at 10m



920 random points
(optionally: add other sources)

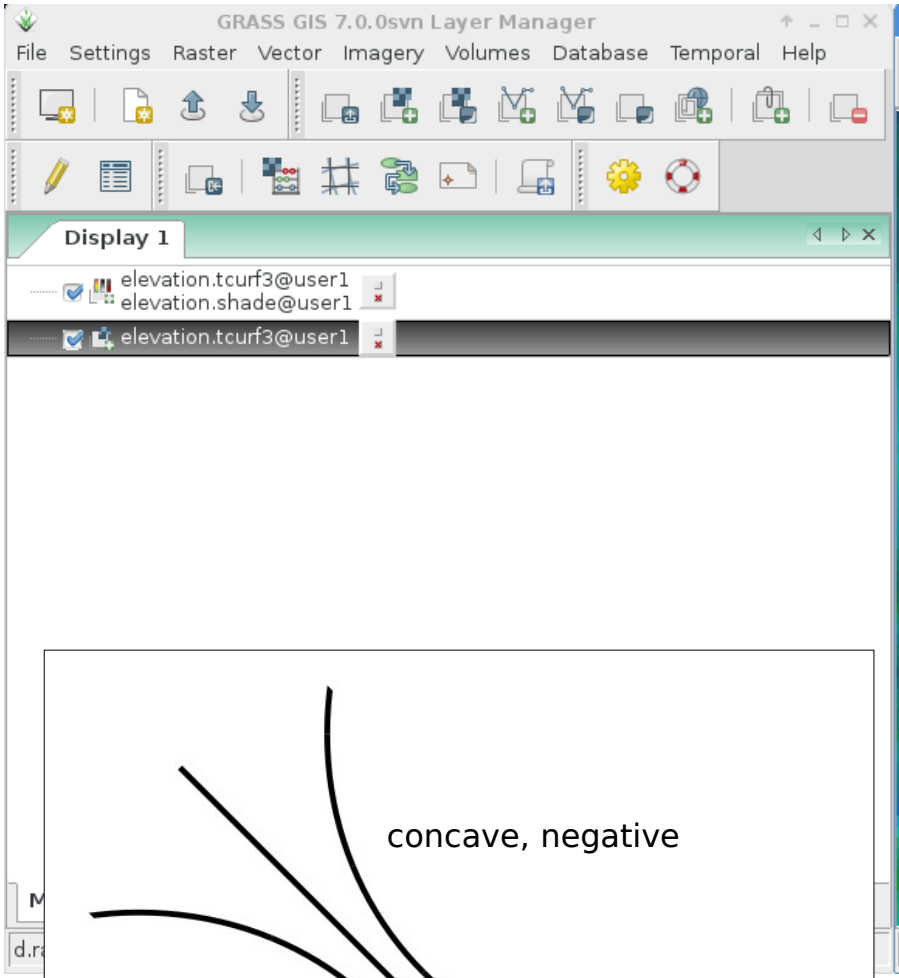


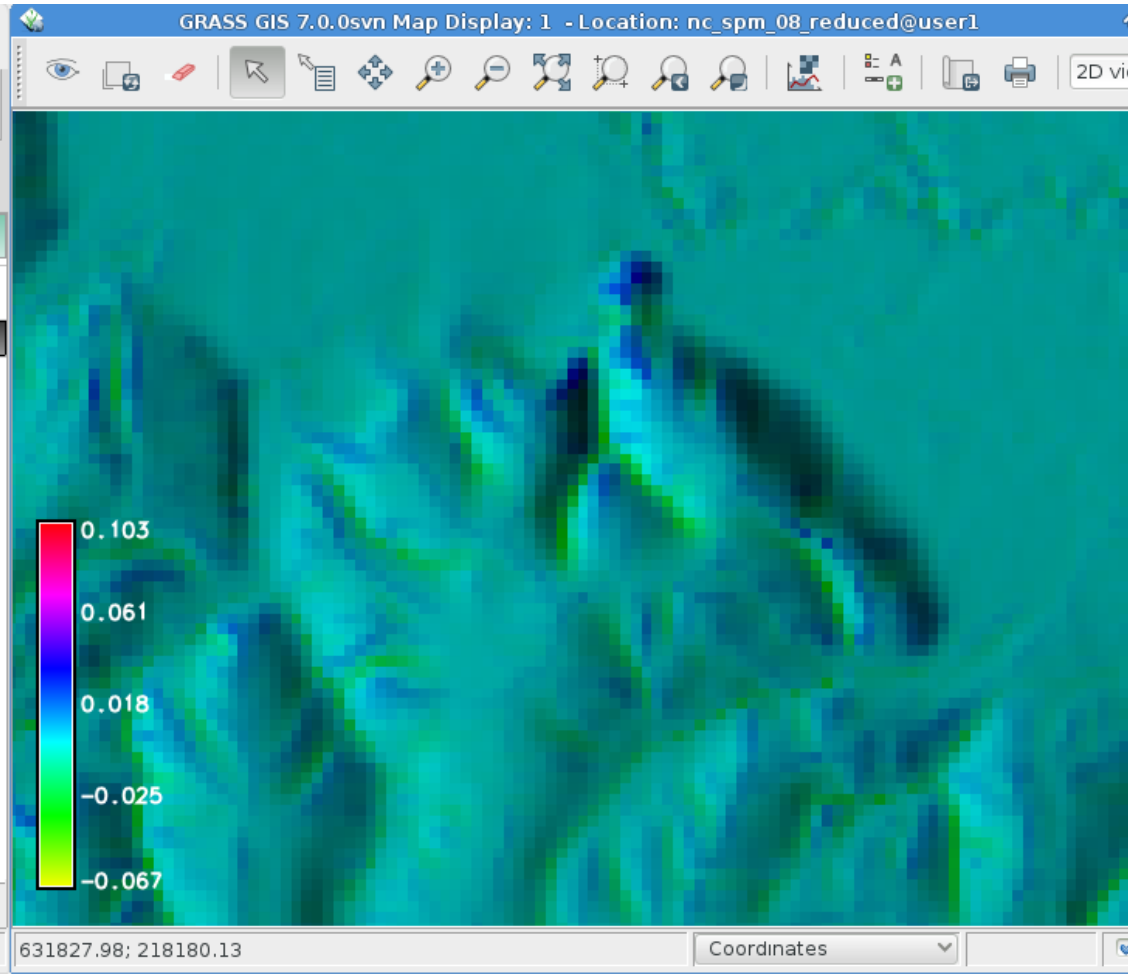
RST: elev state at 10m

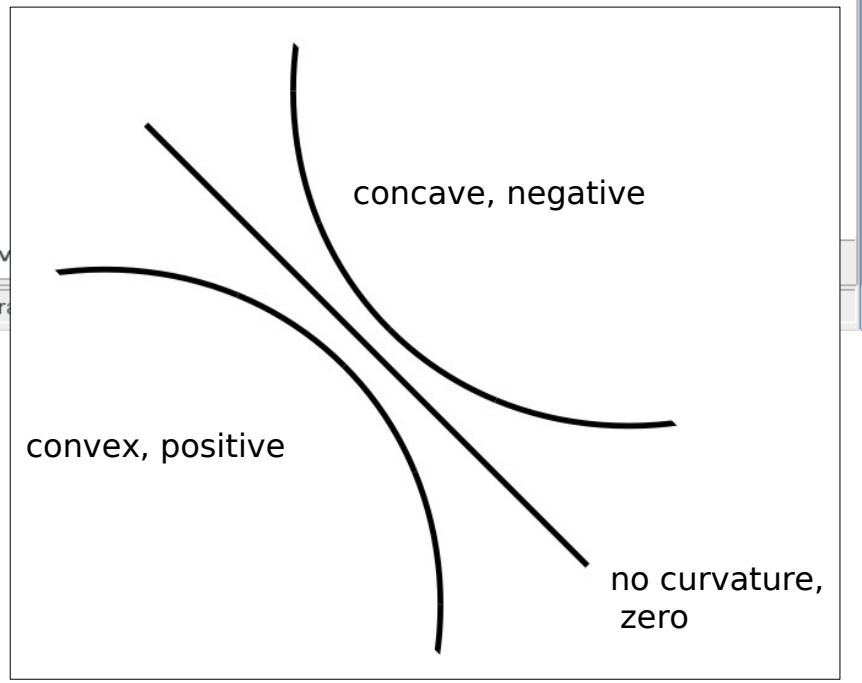


differences at 10m

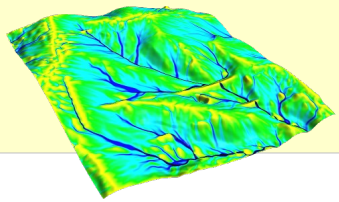
Calculation of terrain curvatures







Raster data analysis: further methods

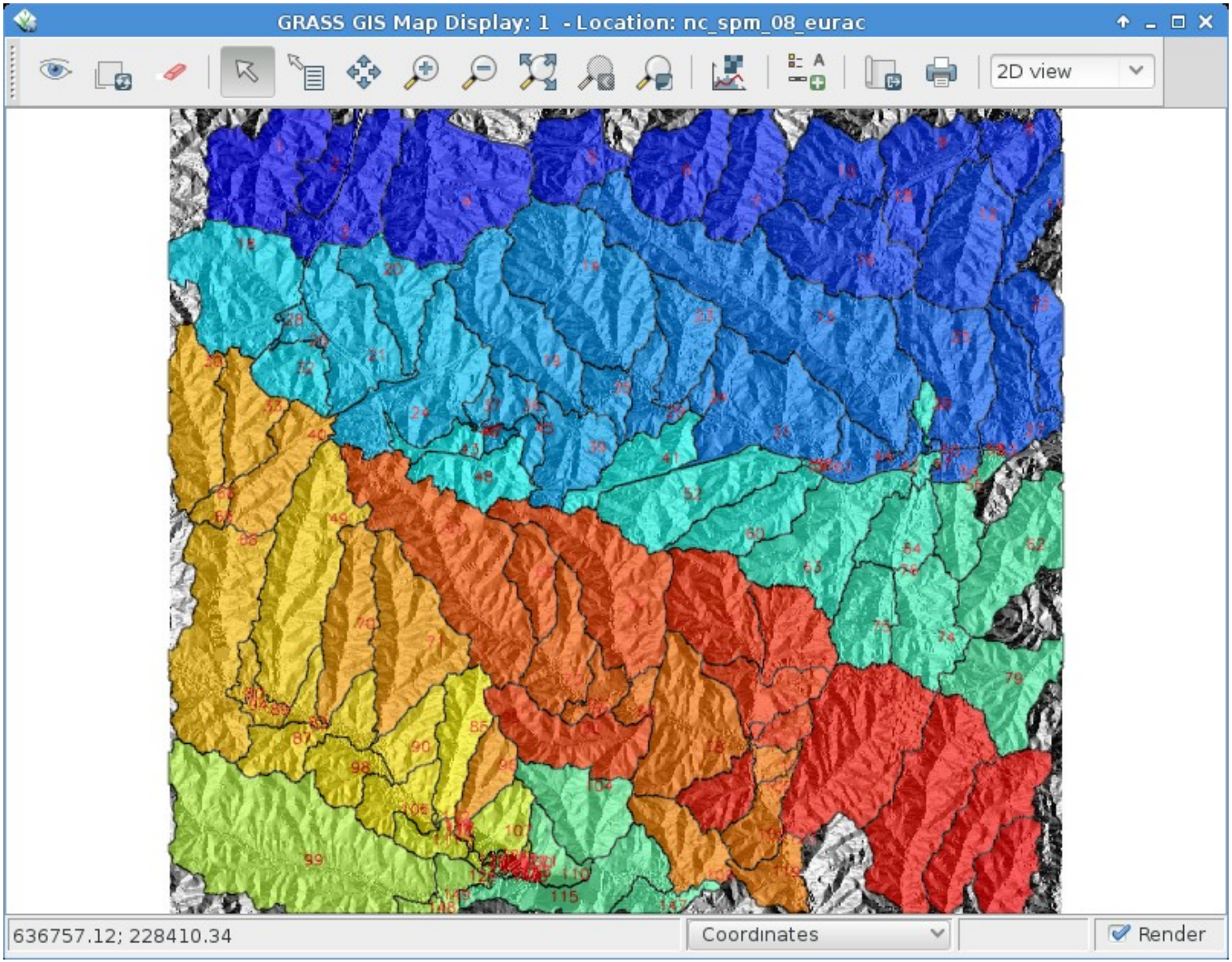


- Additional DEM analysis modules:
  - depression areas can be filled with **r.fill.dir**
  - flowlines can be calculated with **r.flow**
  - trace a flow through a DEM: **r.drain**
  - watershed analysis can be done with **r.watershed** and **r.terraflow**
  - cost surfaces: **r.cost, r.walk**
- Energy:
  - cast shadows, astronomical calculations of sun position: **r.sunmask**
  - energy budget: **r.sun**
- Line of sight:
  - viewsheds can be generated with: **r.viewshed**
- Interpolation methods
  - 2D inverse distance weighted: **v.surf.idw**
  - 2D from contour lines: **r.surf.contour**
  - 2D bilinear: **r.resamp.interp**
  - 2D regularized splines with tension (with cross validation): **v.surf.rst**
  - 3D regularized splines with tension (with cross validation): **v.vol.rst**
  - 2D/3D kernel densities: **v.kernel**

  ... and much more!

https://grass.osgeo.org/grass70/manuals/

Raster-vector statistics: DEM stats per watershed



With **v.rast.stats** we will calculate univariate statistics per polygon based on a raster map.

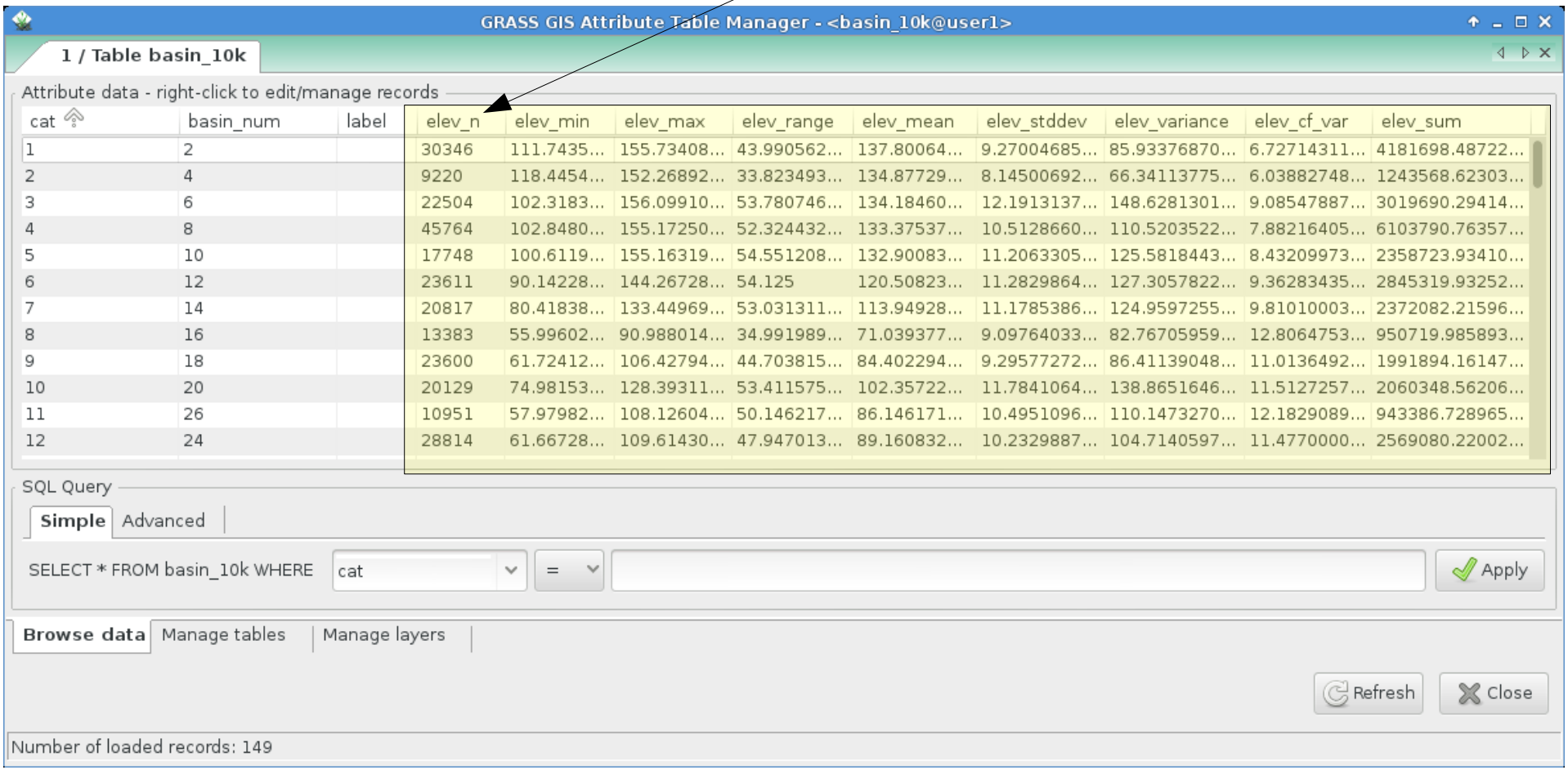
Results are added as new columns to the attribute table of the watershed map.

Raster-vector statistics: DEM stats per watershed

Zonal statistics of elevation data per watershed basin

```
v.rast.stats map=basins_10k raster=elevation \
    column_prefix=elev
v.db.select basins_10k separator=comma
```

Statistics per watershed ID ...



cat	basin_num	label	elev_n	elev_min	elev_max	elev_range	elev_mean	elev_stddev	elev_variance	elev_cf_var	elev_sum
1	2		30346	111.7435...	155.73408...	43.990562...	137.80064...	9.27004685...	85.93376870...	6.72714311...	4181698.48722...
2	4		9220	118.4454...	152.26892...	33.823493...	134.87729...	8.14500692...	66.34113775...	6.03882748...	1243568.62303...
3	6		22504	102.3183...	156.09910...	53.780746...	134.18460...	12.1913137...	148.6281301...	9.08547887...	3019690.29414...
4	8		45764	102.8480...	155.17250...	52.324432...	133.37537...	10.5128660...	110.5203522...	7.88216405...	6103790.76357...
5	10		17748	100.6119...	155.16319...	54.551208...	132.90083...	11.2063305...	125.5818443...	8.43209973...	2358723.93410...
6	12		23611	90.14228...	144.26728...	54.125	120.50823...	11.2829864...	127.3057822...	9.36283435...	2845319.93252...
7	14		20817	80.41838...	133.44969...	53.031311...	113.94928...	11.1785386...	124.9597255...	9.81010003...	2372082.21596...
8	16		13383	55.99602...	90.988014...	34.991989...	71.039377...	9.09764033...	82.76705959...	12.8064753...	950719.985893...
9	18		23600	61.72412...	106.42794...	44.703815...	84.402294...	9.29577272...	86.41139048...	11.0136492...	1991894.16147...
10	20		20129	74.98153...	128.39311...	53.411575...	102.35722...	11.7841064...	138.8651646...	11.5127257...	2060348.56206...
11	26		10951	57.97982...	108.12604...	50.146217...	86.146171...	10.4951096...	110.1473270...	12.1829089...	943386.728965...
12	24		28814	61.66728...	109.61430...	47.947013...	89.160832...	10.2329887...	104.7140597...	11.4770000...	2569080.22002...

Georectification of a scanned map

g.gui.gcp: Insertion of GCP pairs (source – target):

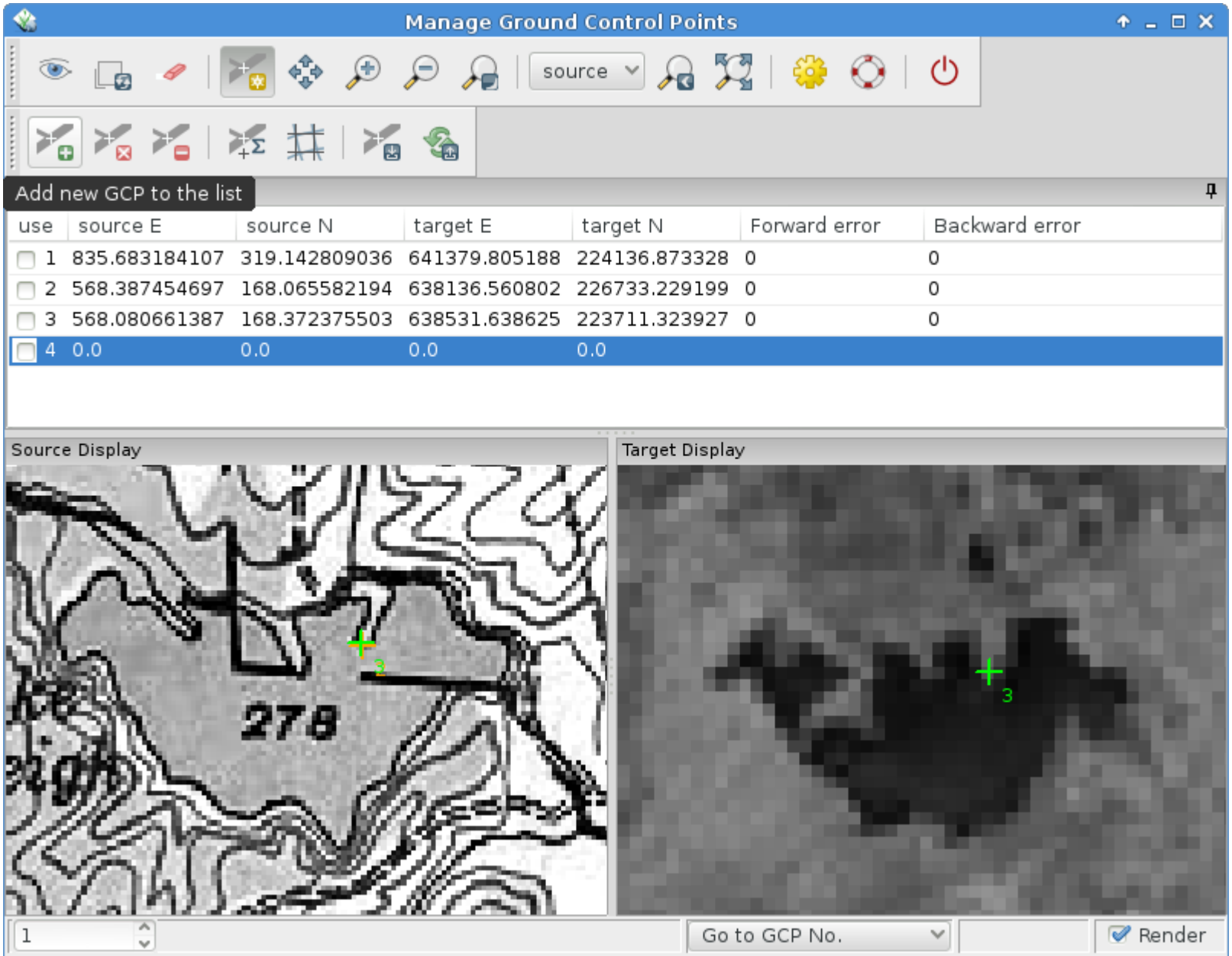
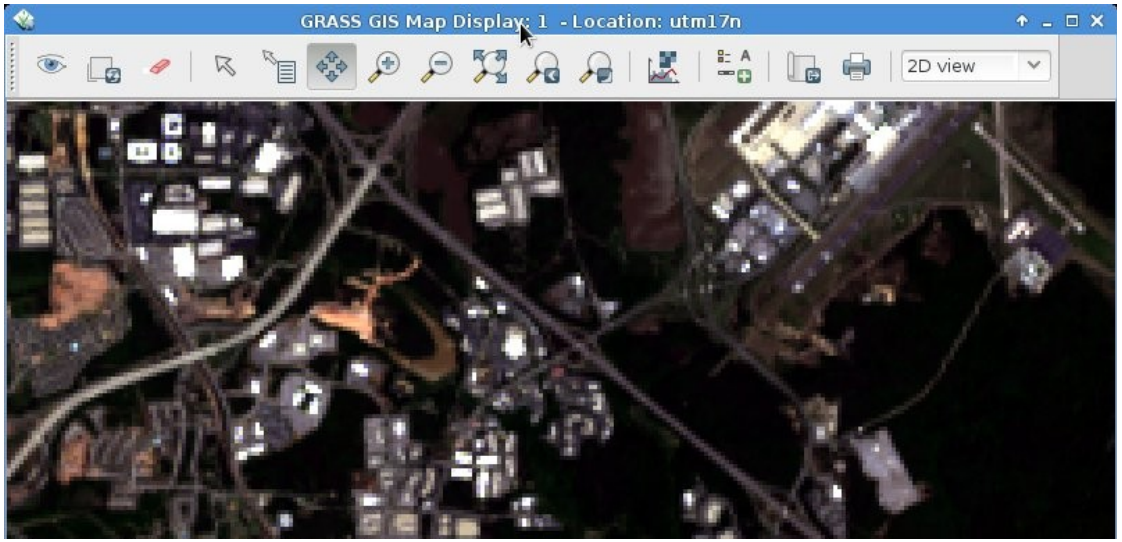
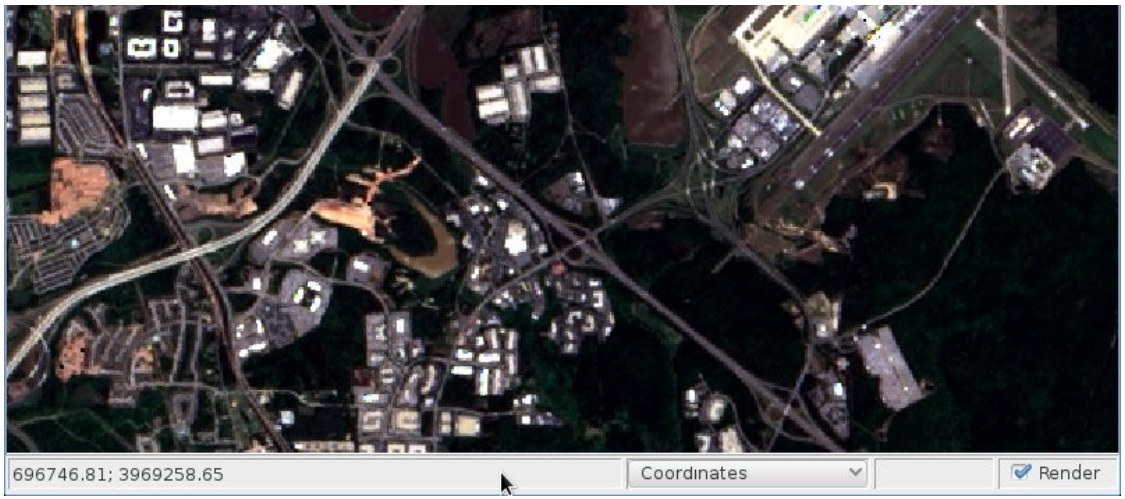


Image fusion – pansharpening

LANDSAT 8: Improving the R/G/B resolution with panchromatic band



R/G/B composite at 28m

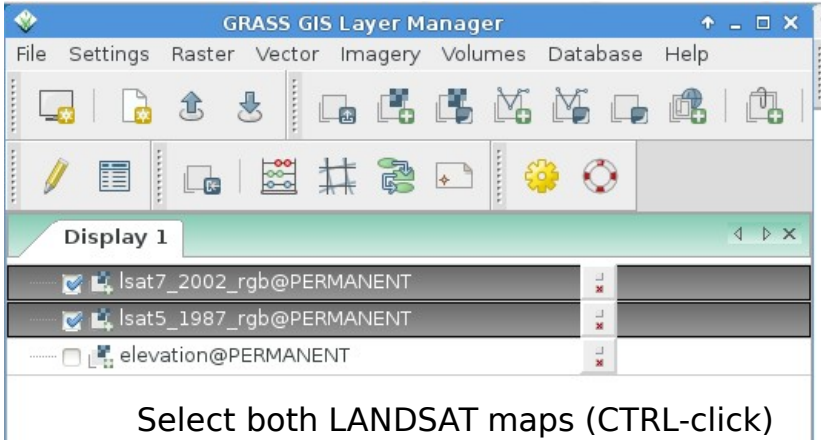


R/G/B composite at 14.25m
(IHS method, color balanced)

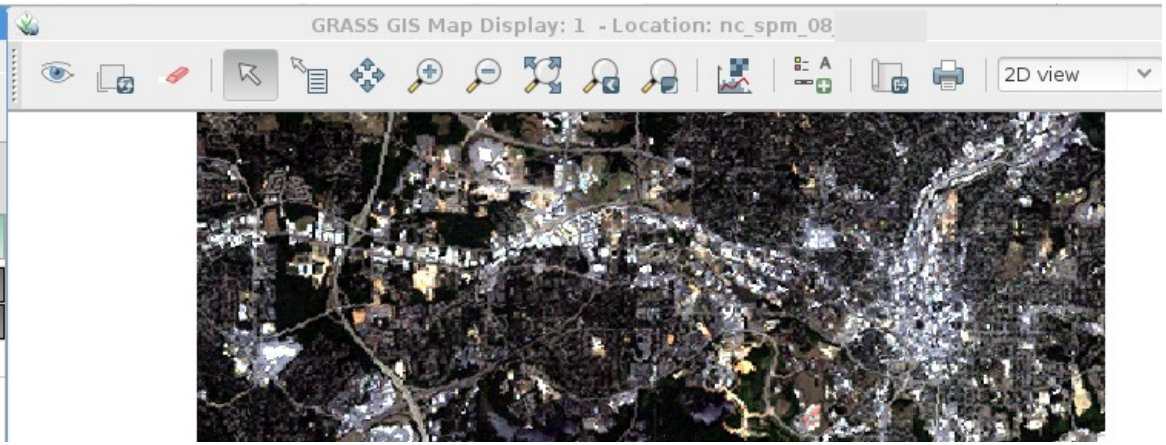
Note: the colors depend on the method used for pansharpening. Several methods are provided in **i.panscharpen**

Map swiping for multitemporal maps

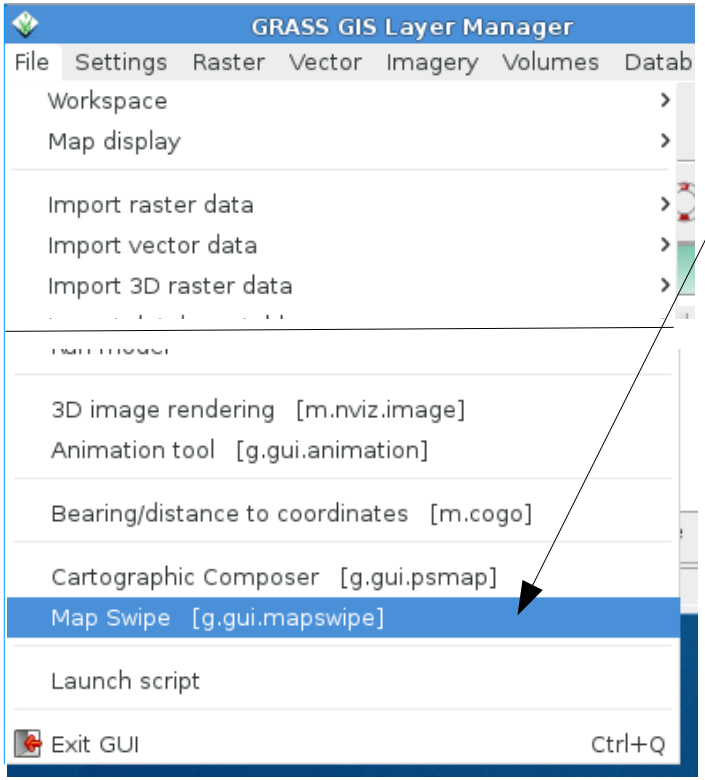
Comparing the LANDSAT 5 (1987) and 7 (2002) **RGB composites** of Wake county:

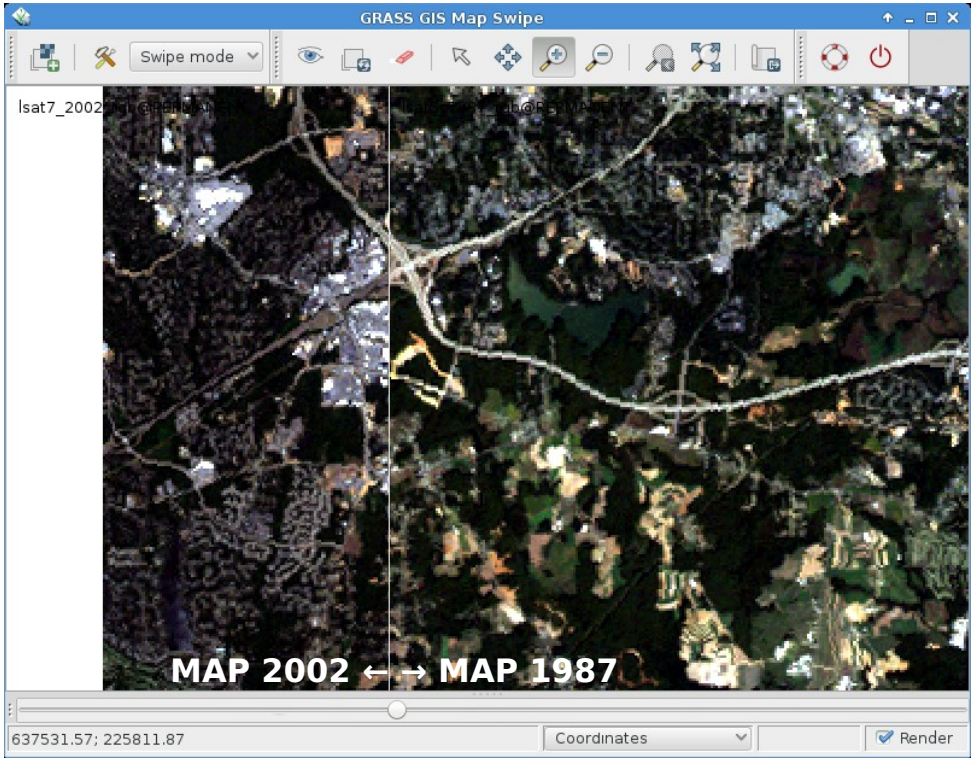


Select both LANDSAT maps (CTRL-click)



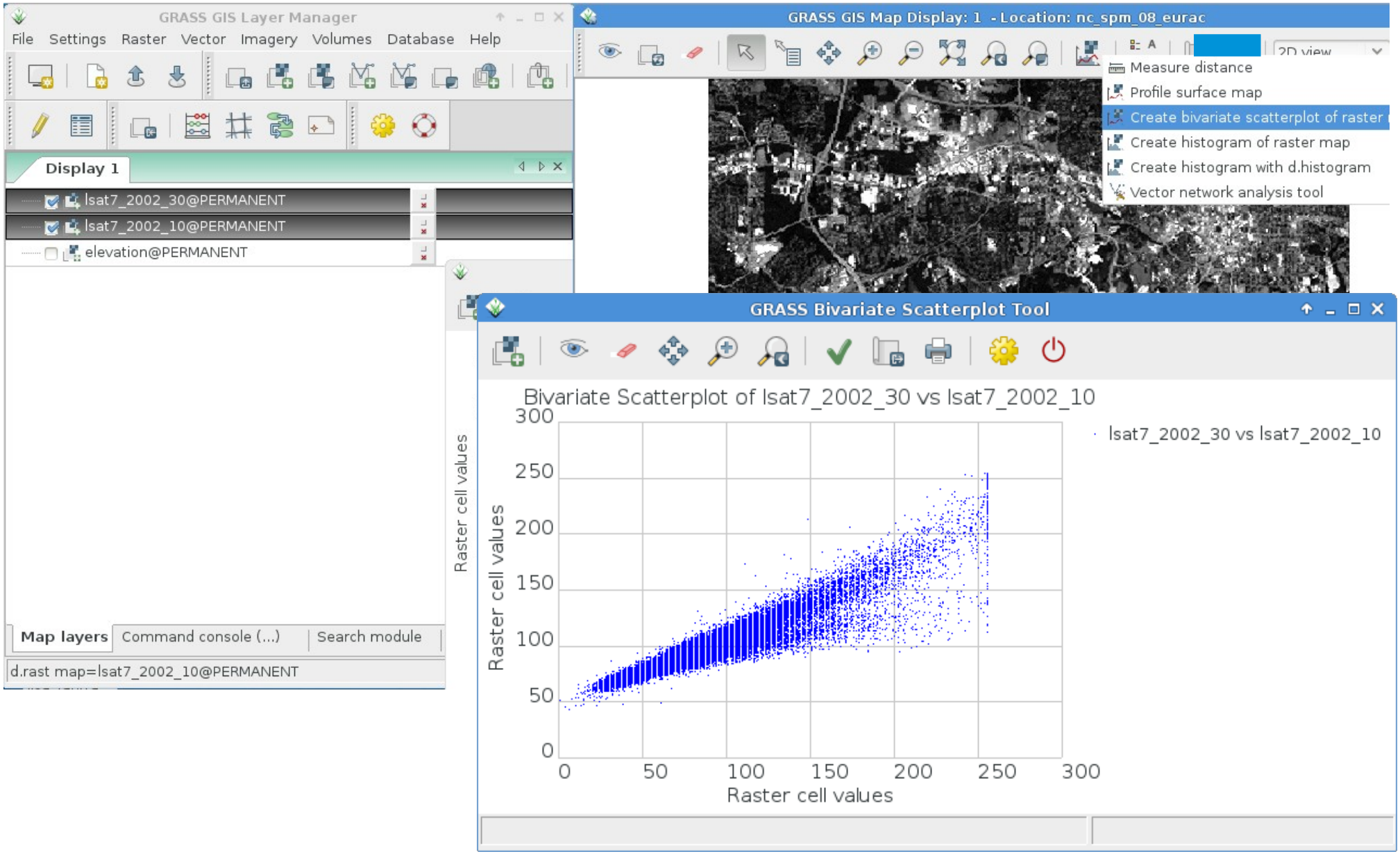
Switch to “map swipe” view





Bivariate Scatterplots

LANDSAT 7 2002 **channels 1 and 3** of Wake county.

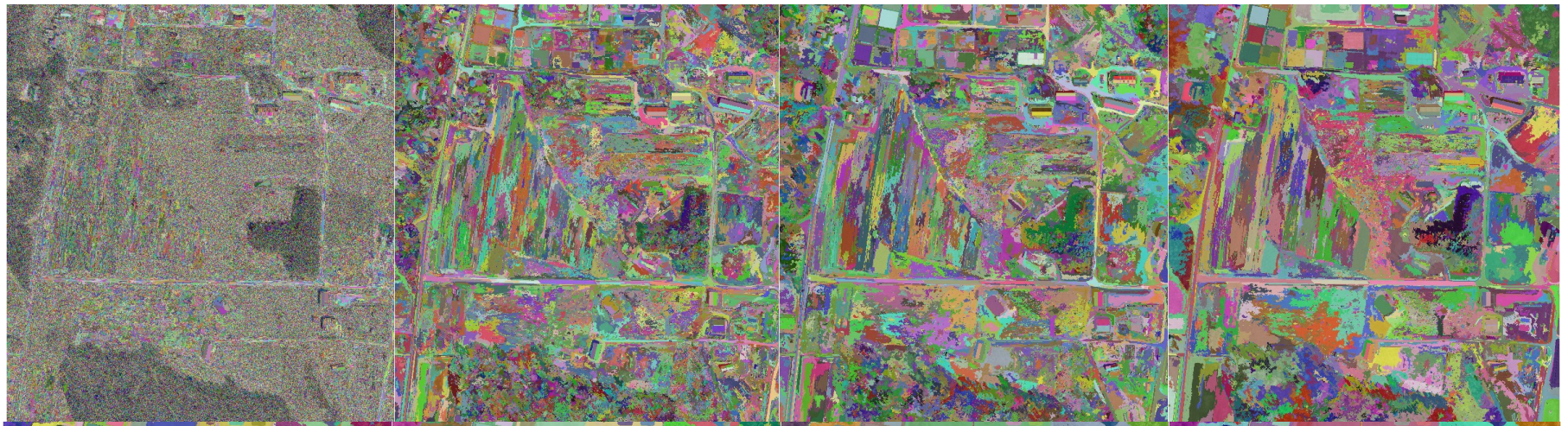


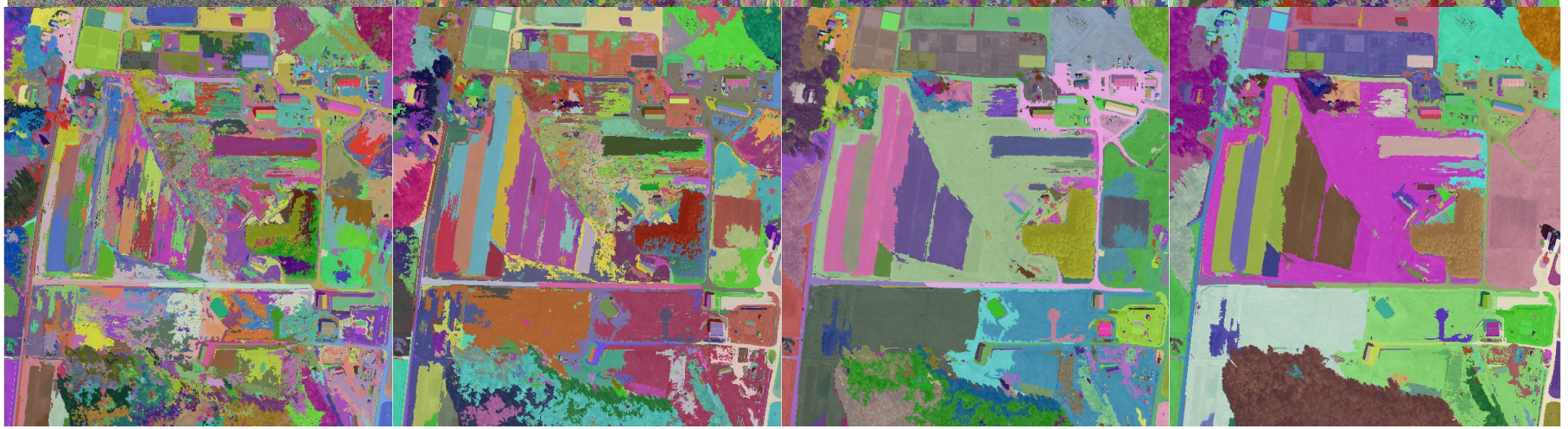
Unsupervised image classification – Segmentation

Re-use output of previous run (seed):

```
i.segment group=ortho2010_t792_subset_20cm \
          output=ortho2010_t792_subset_20cm_segment_50 \
          seeds=ortho2010_t792_subset_20cm_segment_25 \
          goodness=ortho2010_t792_subset_20cm_seg_50_fit \
          threshold=0.50
```

0.01 0.05 0.10 0.25

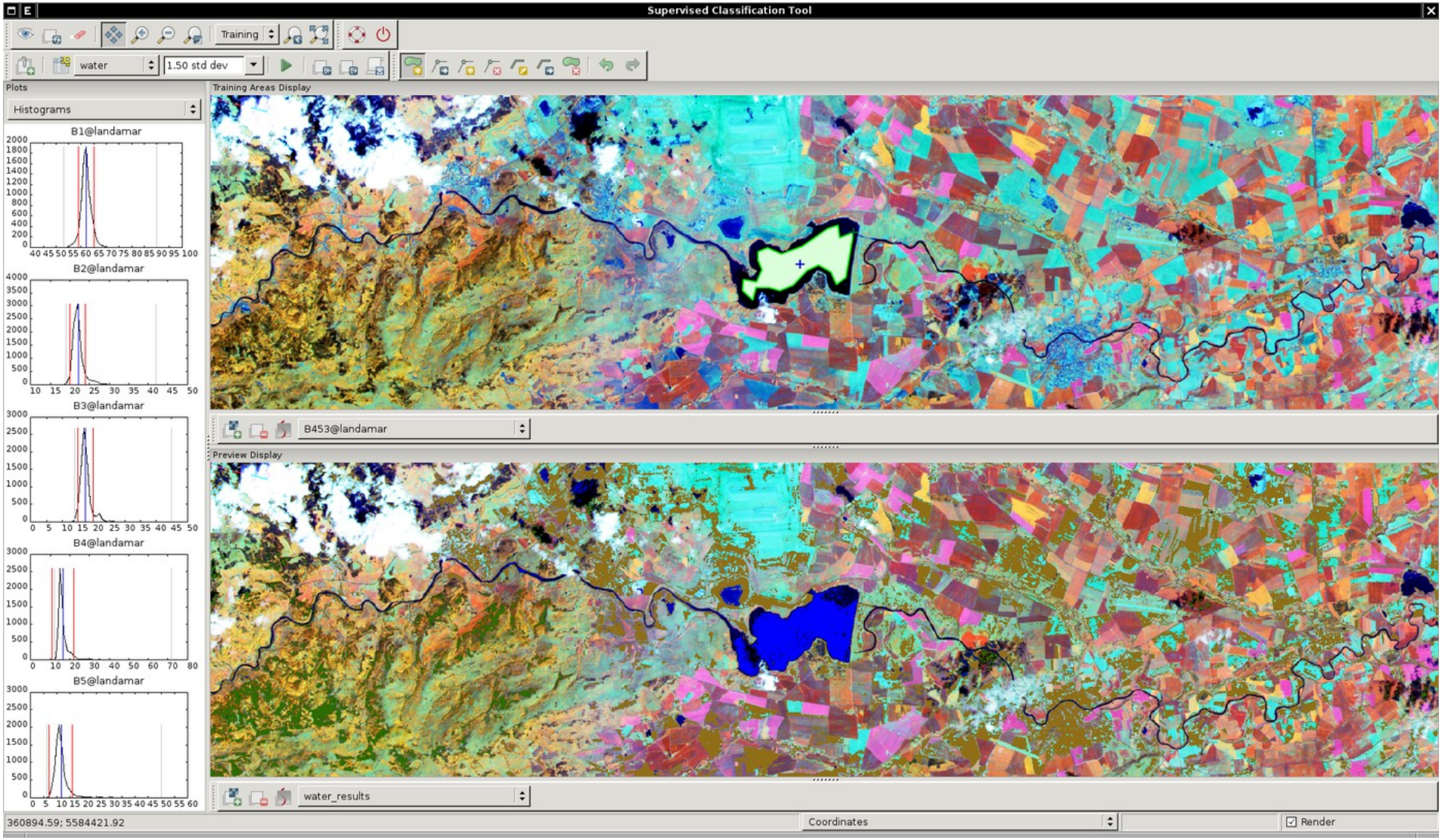




0.50 0.75 0.90 0.95

Supervised image classification

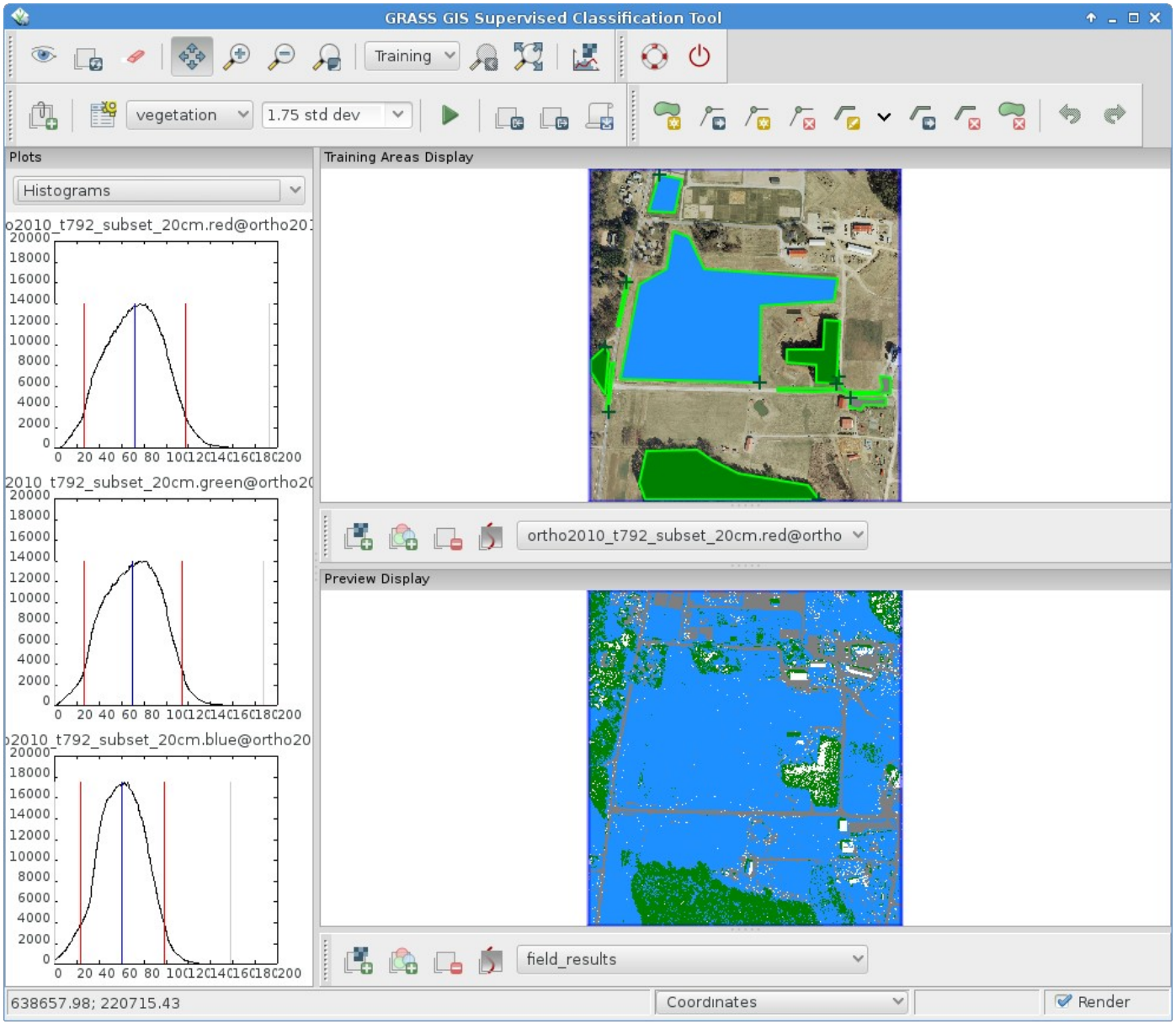
http://geo.fsv.cvut.cz/~landa/publications/2012/ogrs2012/poster/figures/



Tool for supervised classification of imagery data.

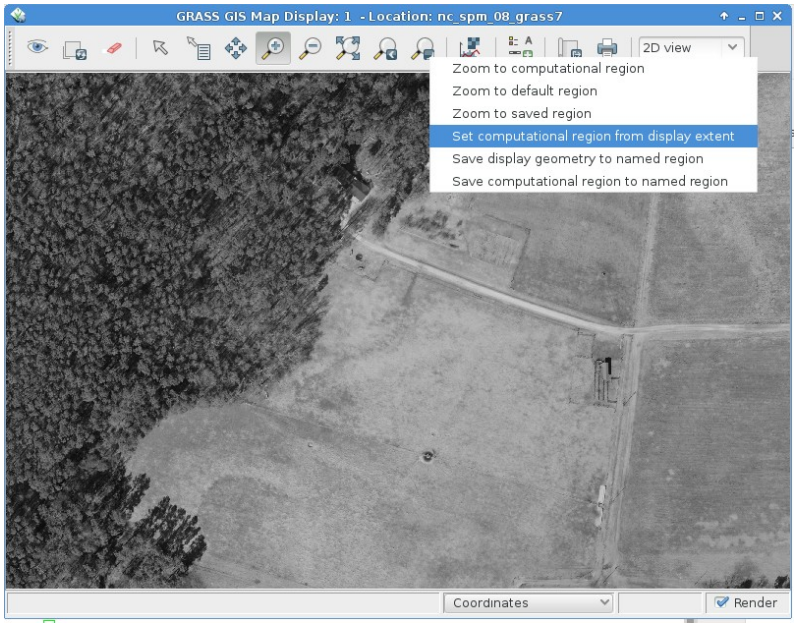
Generates spectral signatures for an image by allowing the user to outline regions of interest.

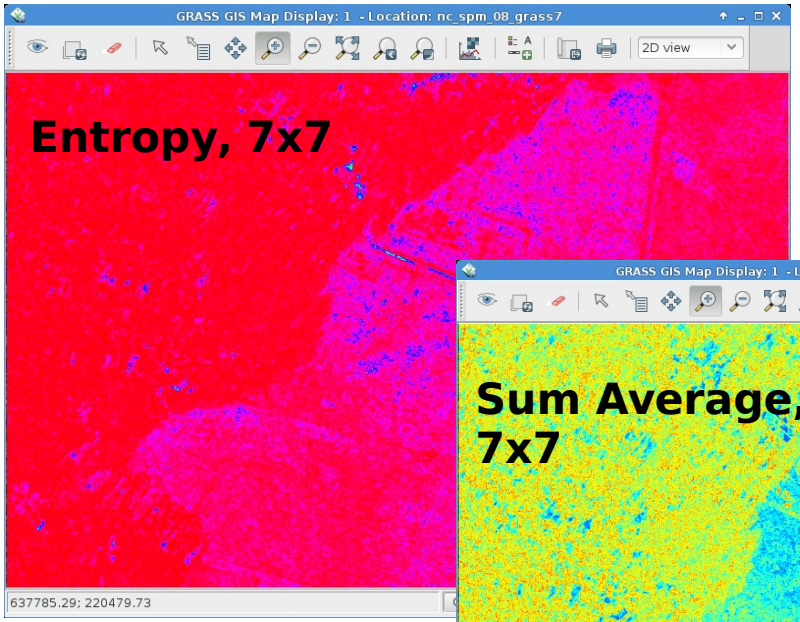
Supervised image classification

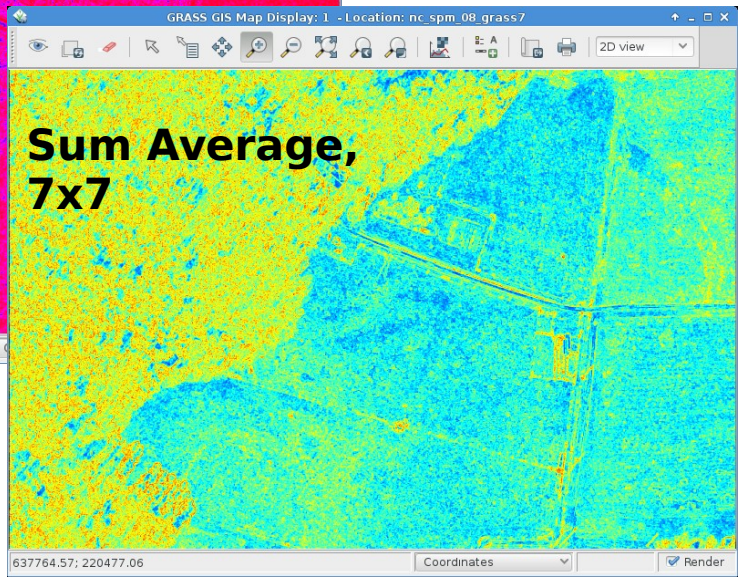


Tool for supervised classification of imagery data.
Generates spectral signatures for an image by allowing the user to outline regions of interest. Maxlik Classification and storage as separate raster map per class.

Texture analysis







r.texture

Calculation of the **sum average** (SA) and of the **entropy** (entr) measure, using 7x7 moving window size

Texture maps may be used as synthetic channels for image classification

Space-Time functionality in GRASS 7

Temporal data processing in GRASS GIS

The temporal GIS framework in GRASS introduces three new datatypes that are designed to handle time series data:

- *Space time raster datasets* (strds) are designed to manage raster map time series. Modules that process strds have the naming prefix *t.rast*.
- *Space time 3D raster datasets* (str3ds) are designed to manage 3D raster map time series. Modules that process str3ds have the naming prefix *t.rast3d*.
- *Space time vector datasets* (stvds) are designed to manage vector map time series. Modules that process stvds have the naming prefix *t.vect*.

Temporal data management in general

List of general management modules:

- t.connect
- t.create
- t.remove
- t.register
- t.unregister
- t.info
- t.list
- t.rast3d.list
- t.vect.list
- t.vect.db.select
- t.sample
- t.support
- t.topology

Export/import conversion

- t.rast.export
- t.rast.import
- t.rast.out.vtk
- t.rast.to.rast3
- r3.out.netcdf
- t.vect.export

Statistics and gap filling

- t.rast.gapfill
- t.rast.univar

Querying and map calculation

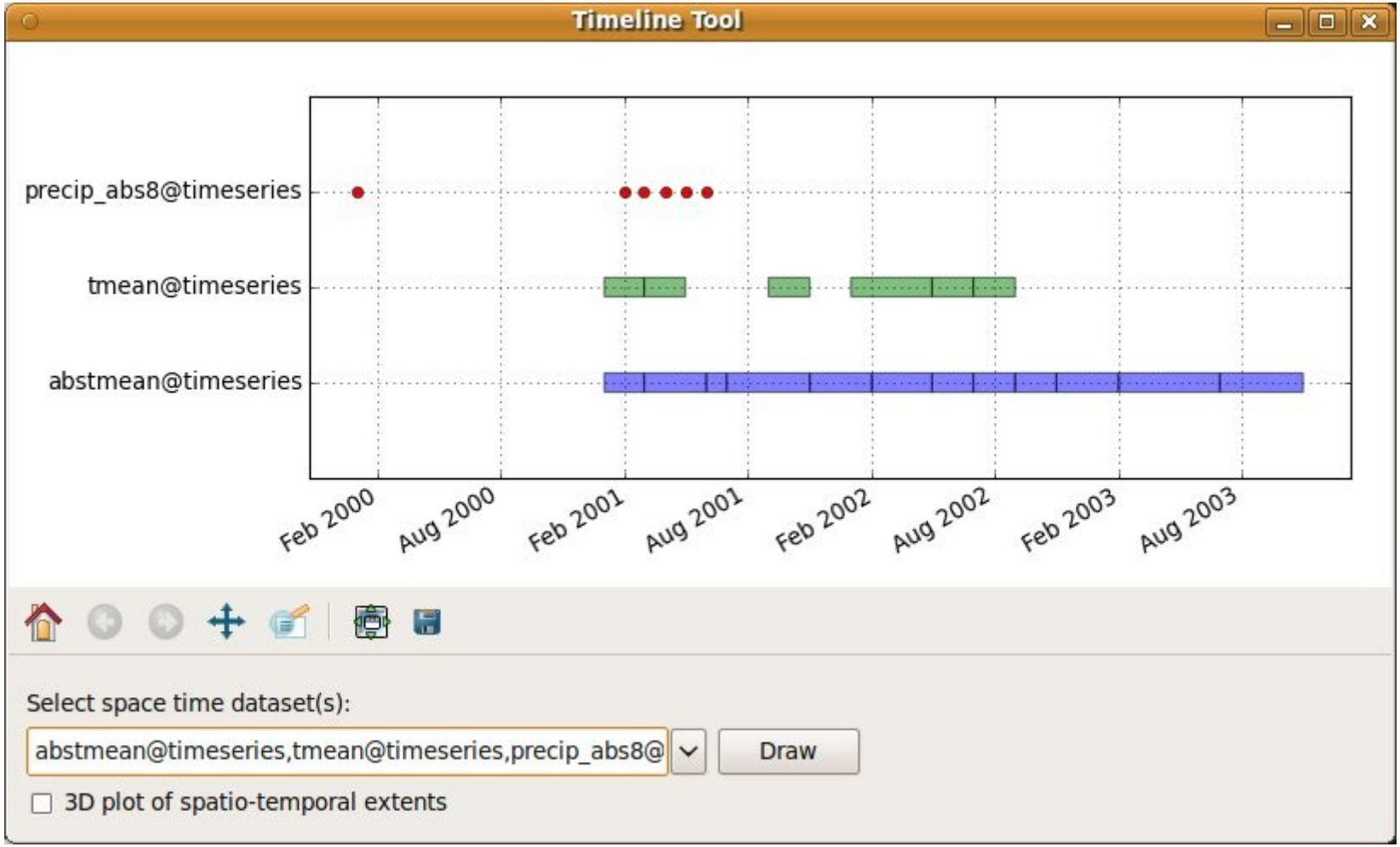
- t.rast.list
- t.rast.extract
- t.rast.gapfill
- t.rast.mapcalc
- t.rast3d.extract
- t.rast3d.mapcalc
- t.rast3d.univar
- t.vect.extract
- t.vect.import
- t.vect.observe.strds
- t.vect.univar
- t.vect.what.strds

Aggregation

- t.rast.aggregate.ds
- t.rast.aggregate
- t.rast.series

Space time datasets are stored in a temporal database. SQLite3 or PostgreSQL are supported as SQL database back end. Connection settings are performed with t.connect. As default a sqlite3 database will be created in the PERMANENT mapset that stores all space time datasets and registered time series maps from all mapsets in the location.

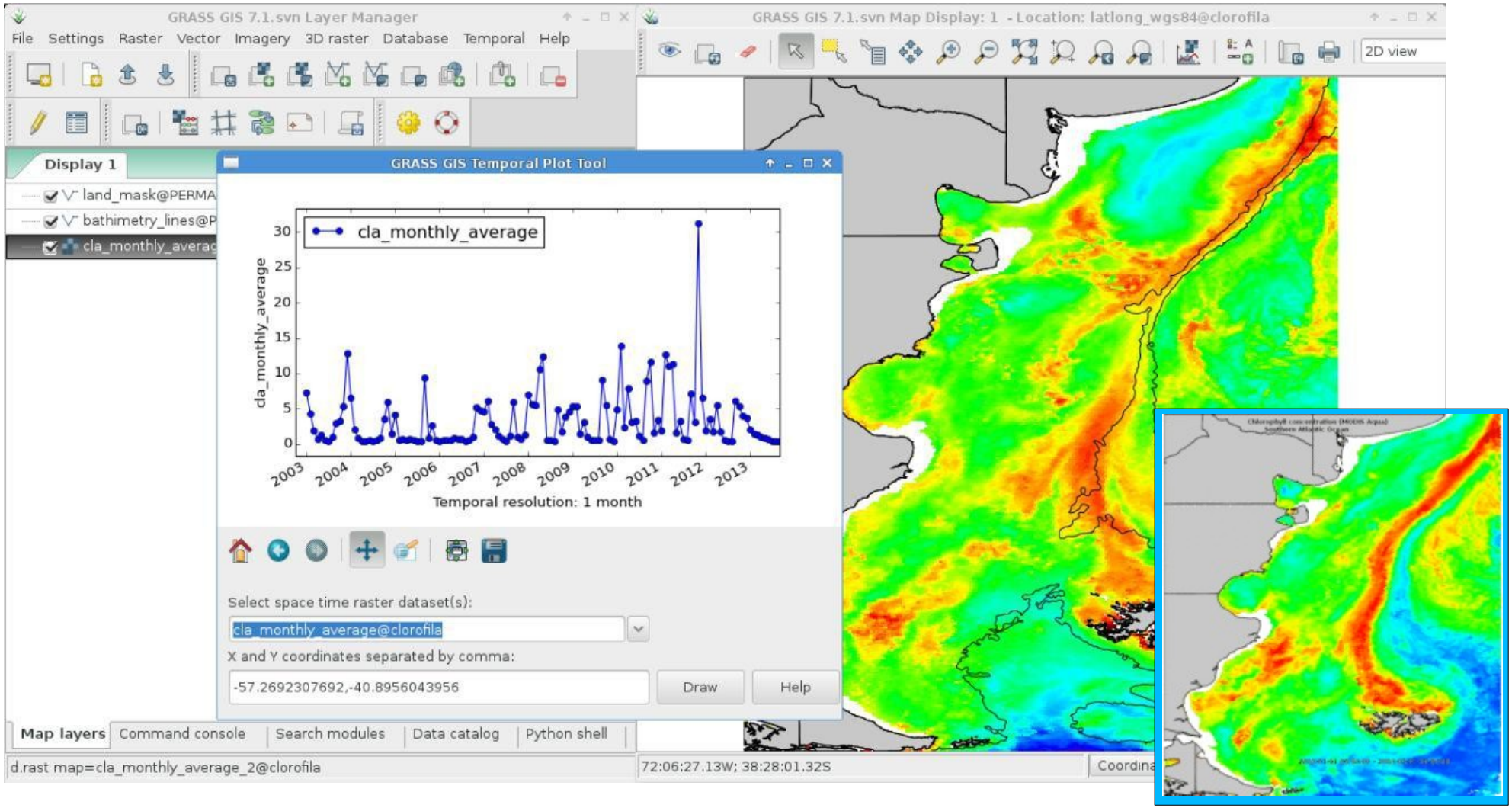
GRASS GIS 7: Space-time functionality



Screenshot: S Gebbert/A. Petrasova

t.register: Registers raster, vector and raster3d maps in a space time dataset

GRASS GIS 7: Space-time functionality

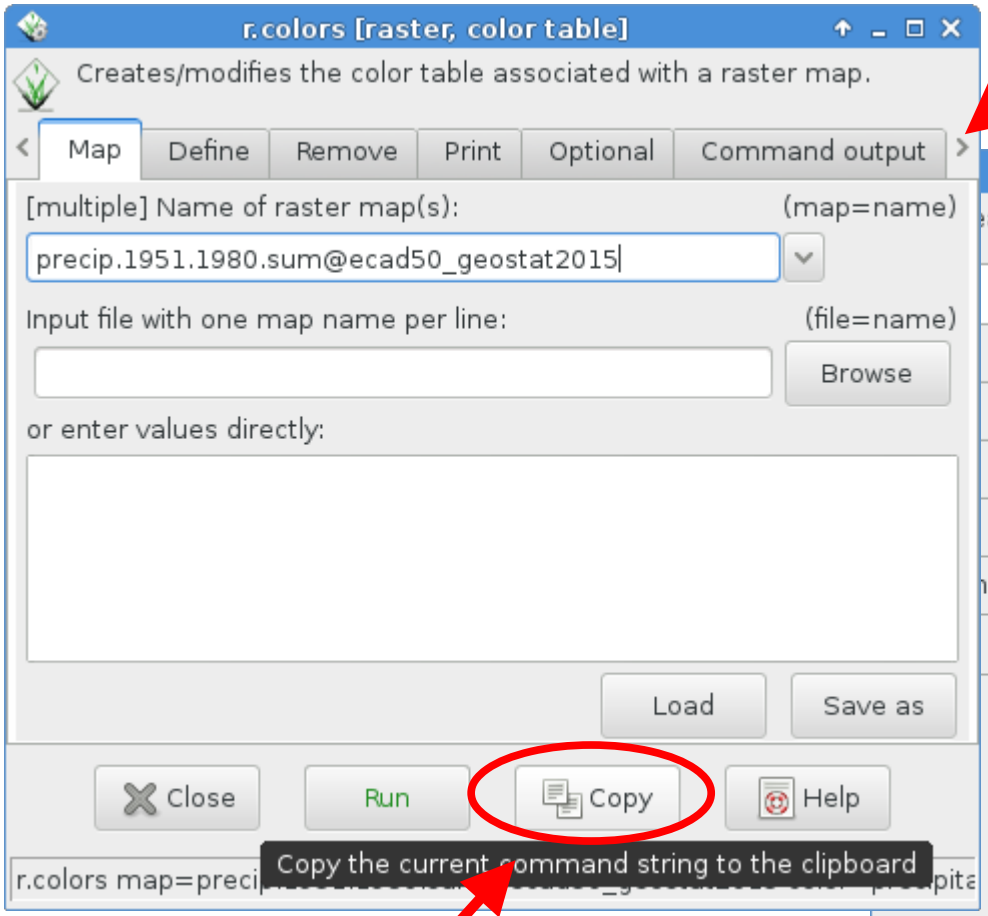


(in PDF, click for animation)

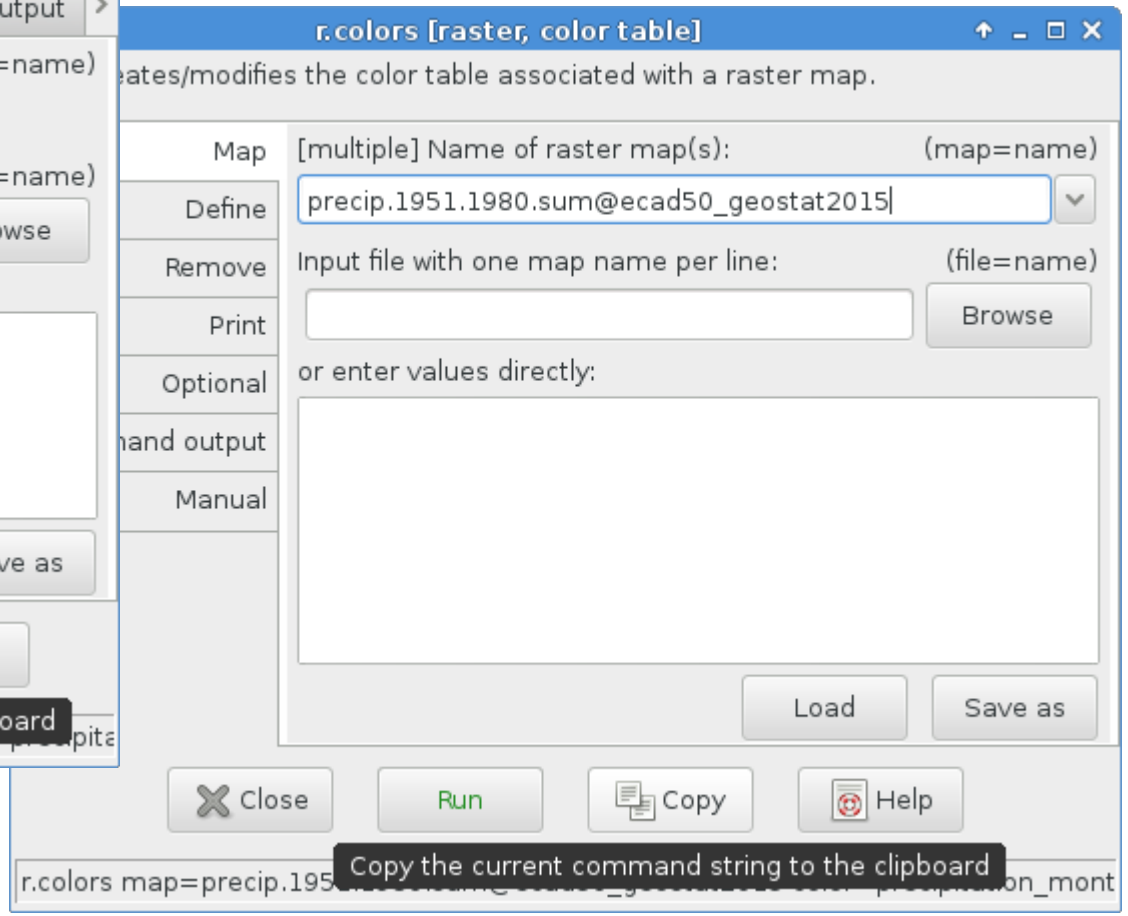
g.gui.tplot: plots the values of one or more temporal raster datasets for a queried point defined by a coordinate pair

Graphical user interface versus Command line

STYLE: Menu: Settings → GUI Settings → Appearance → Module dialog style: Basic top/left



Cycle through the various tabs

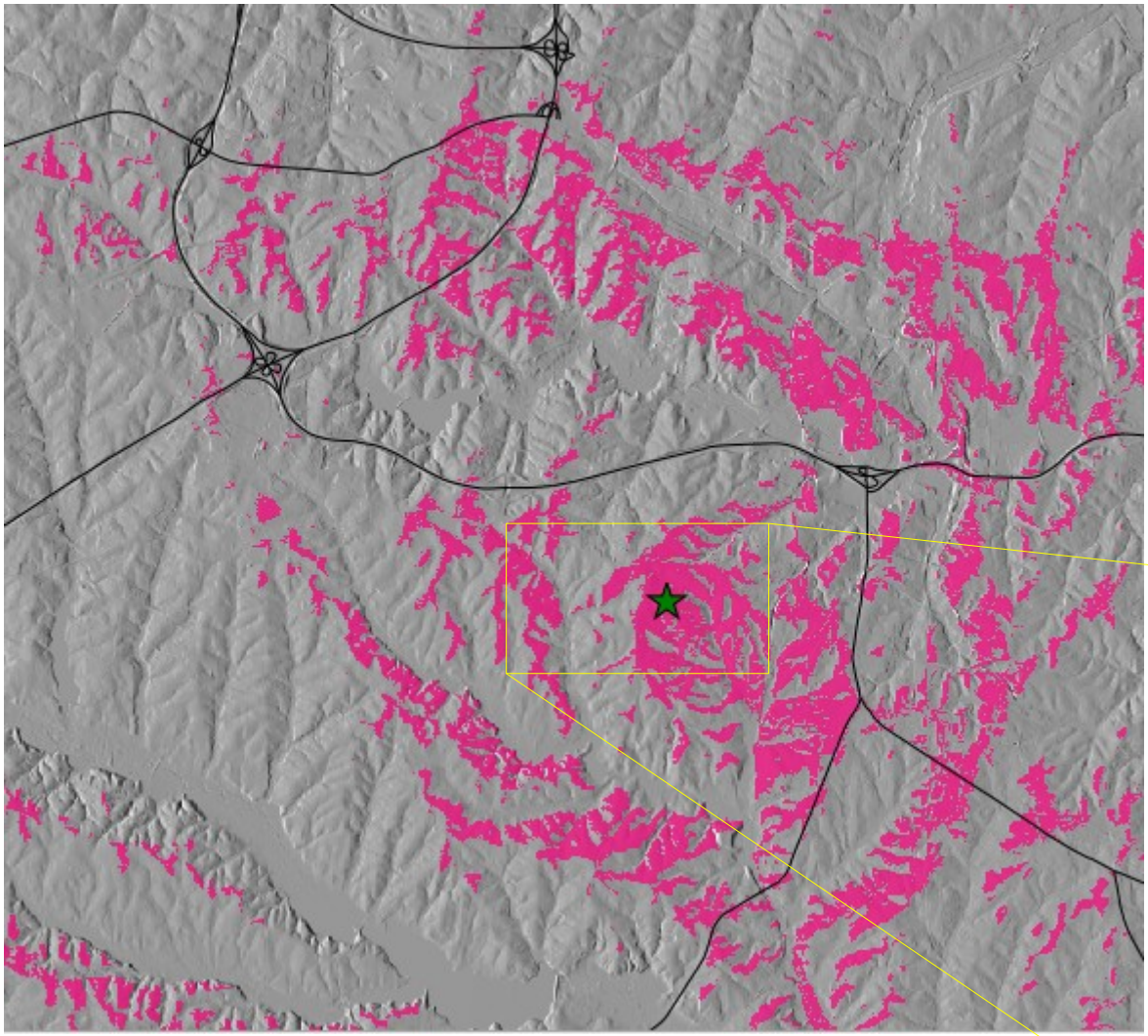


You may copy the command to your documentation

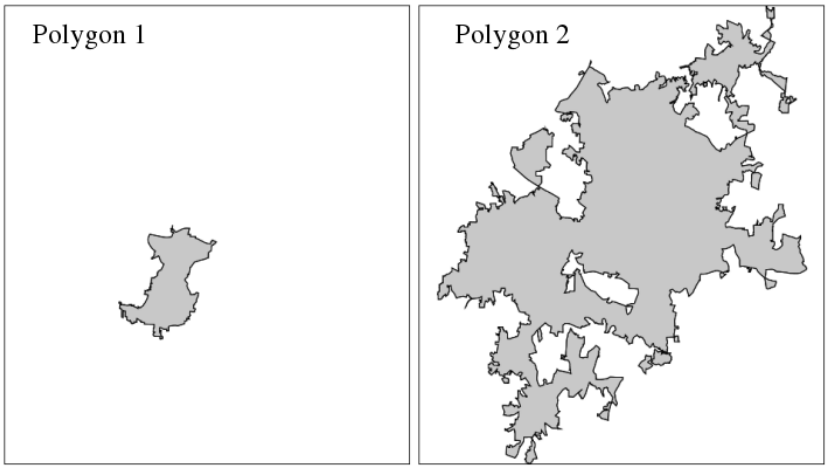
The graphical user interface effectively generates the respective command for the command line (and also writes to the shell “history”)

Viewshed (line of sight)

r.viewshed in action: what is visible from a certain viewpoint?



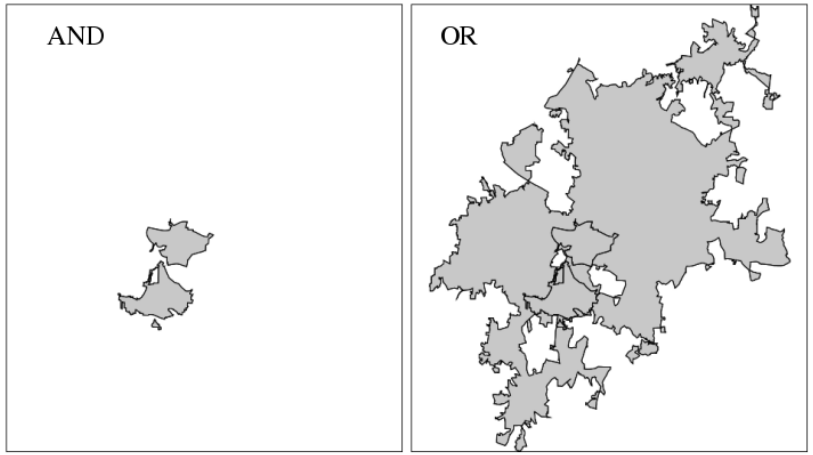
Vector feature overlay operations

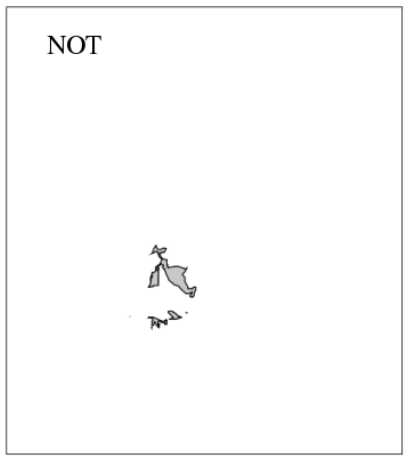


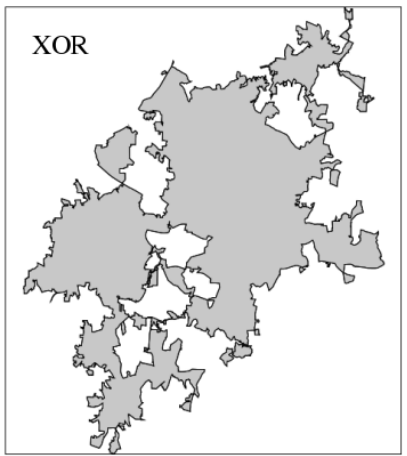
Boolean operators

GRASS GIS module:

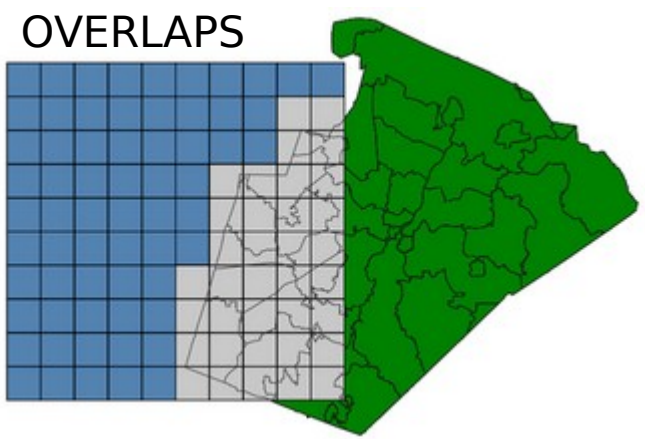
`v.overlay`

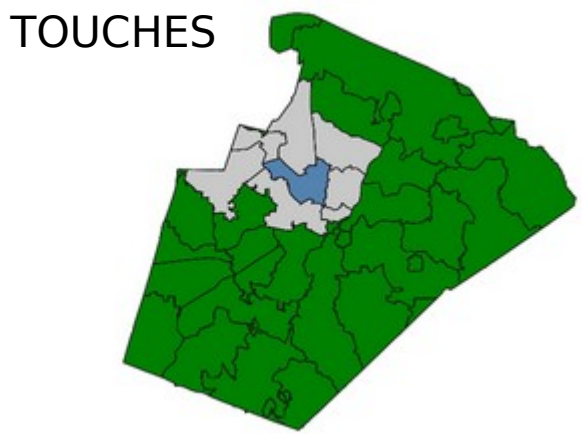


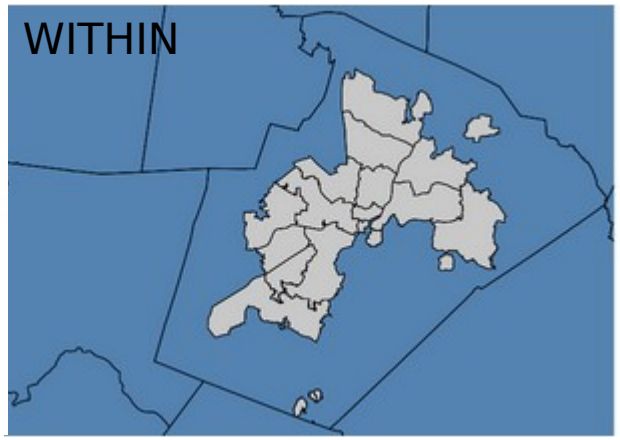


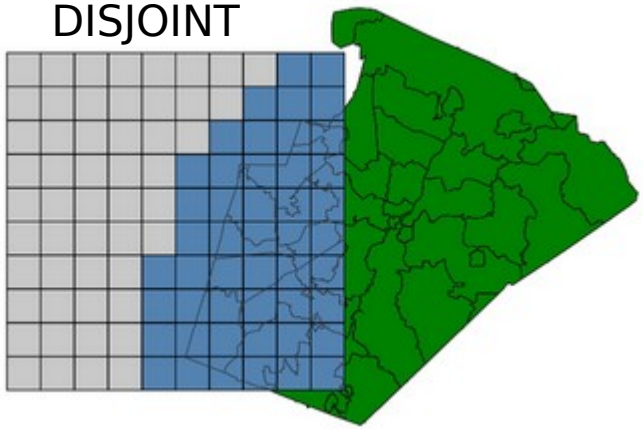


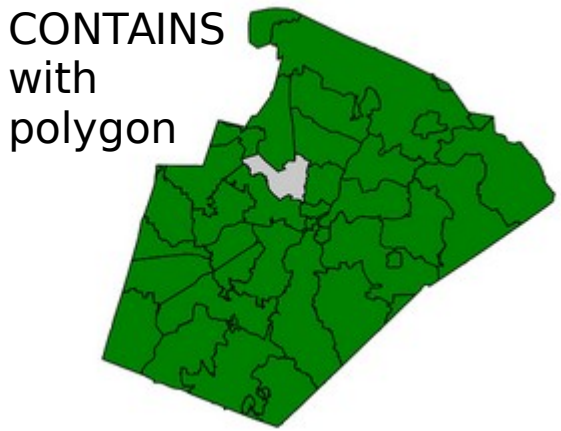
Vector feature select operations: v.select (GEOS)

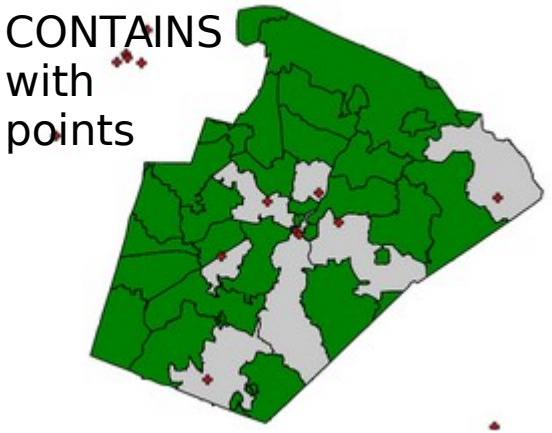


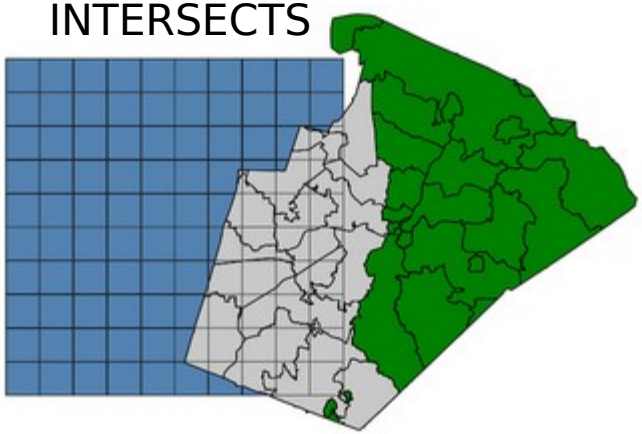


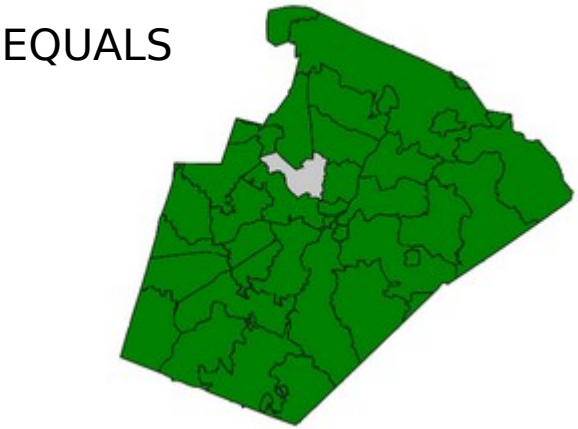


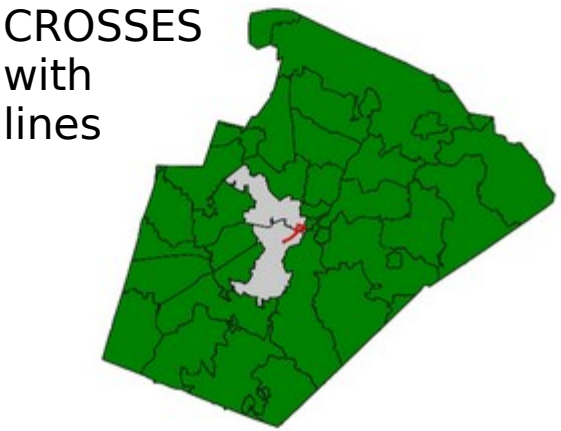




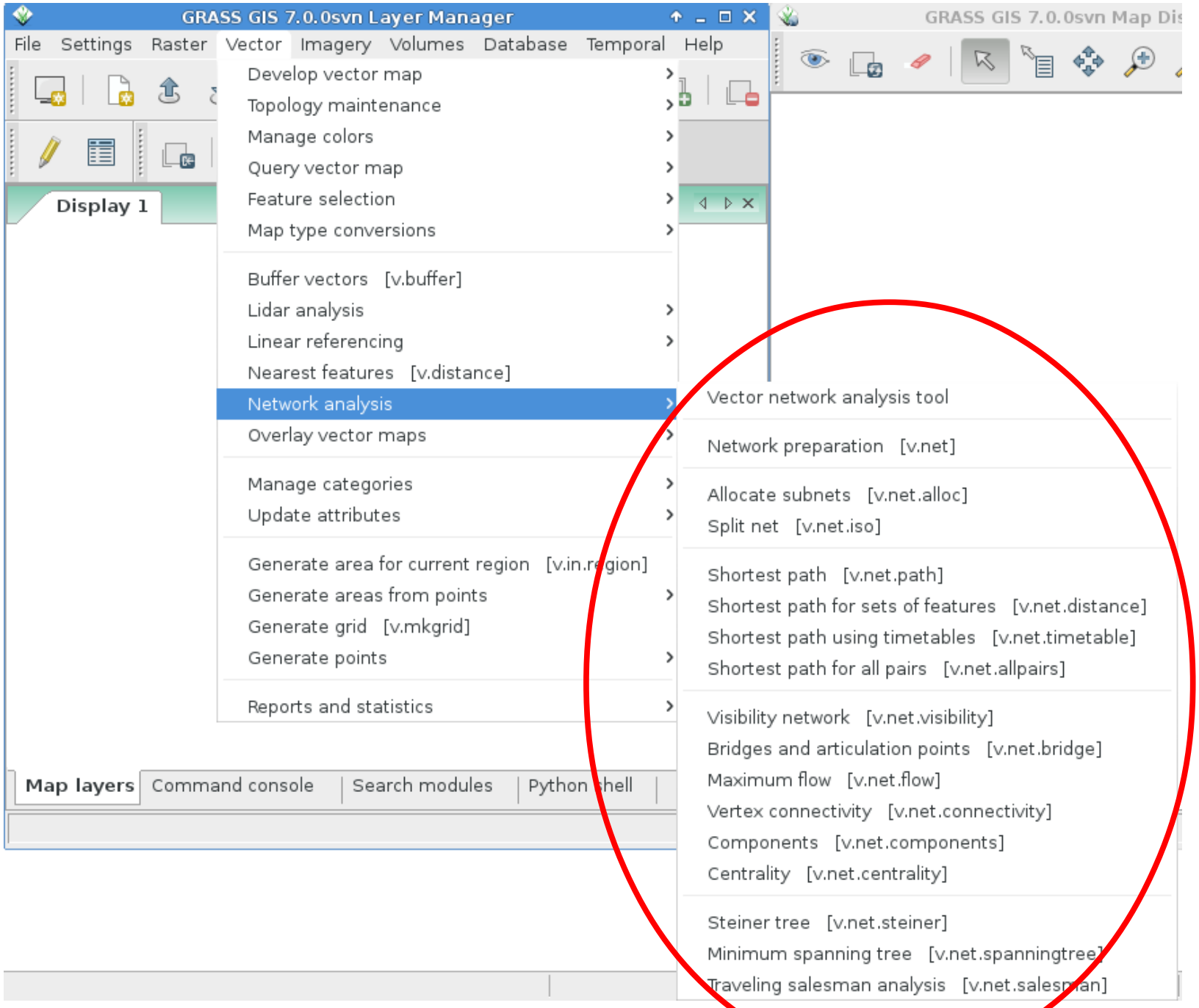






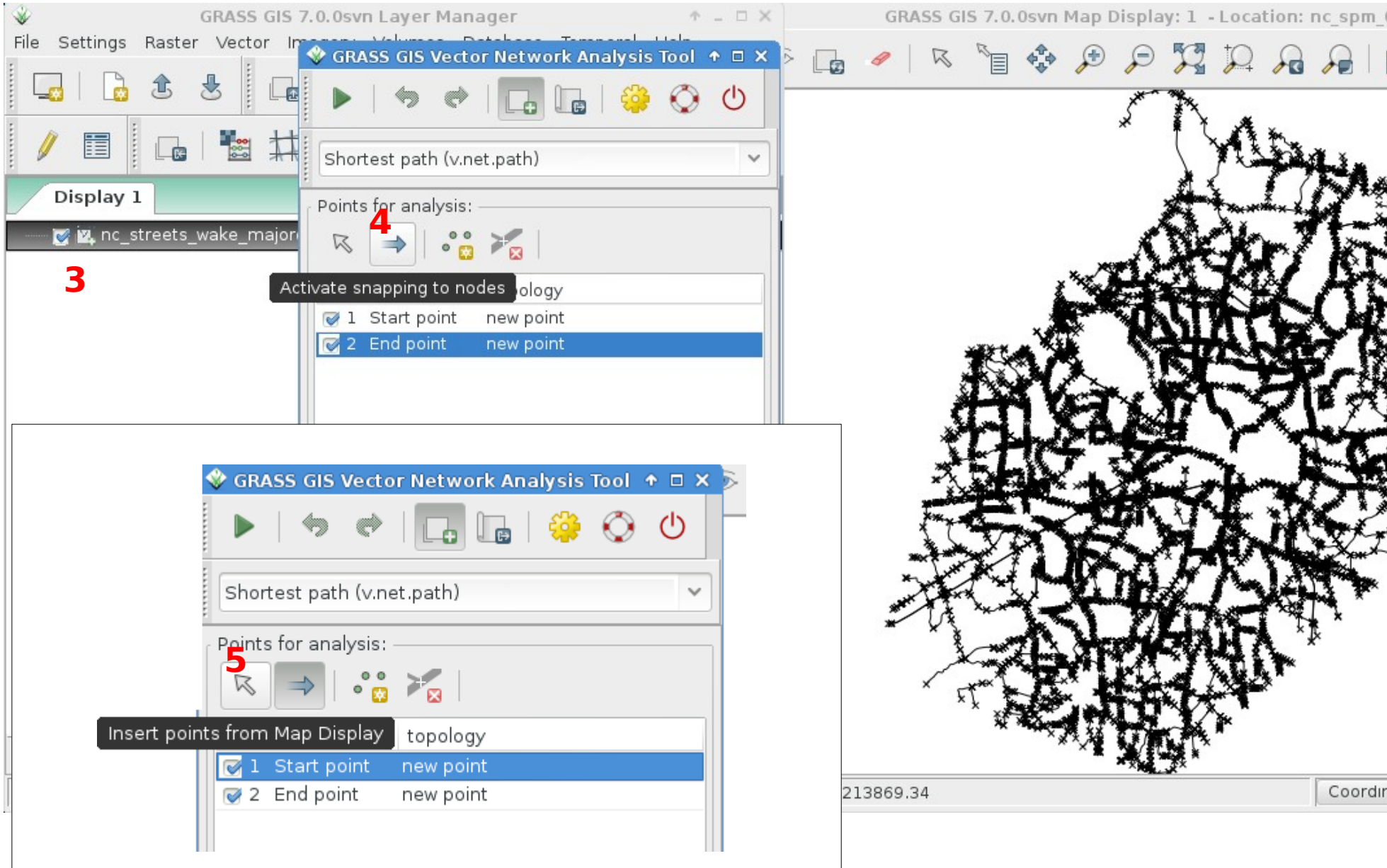


Overview: Vector network analysis in GRASS GIS



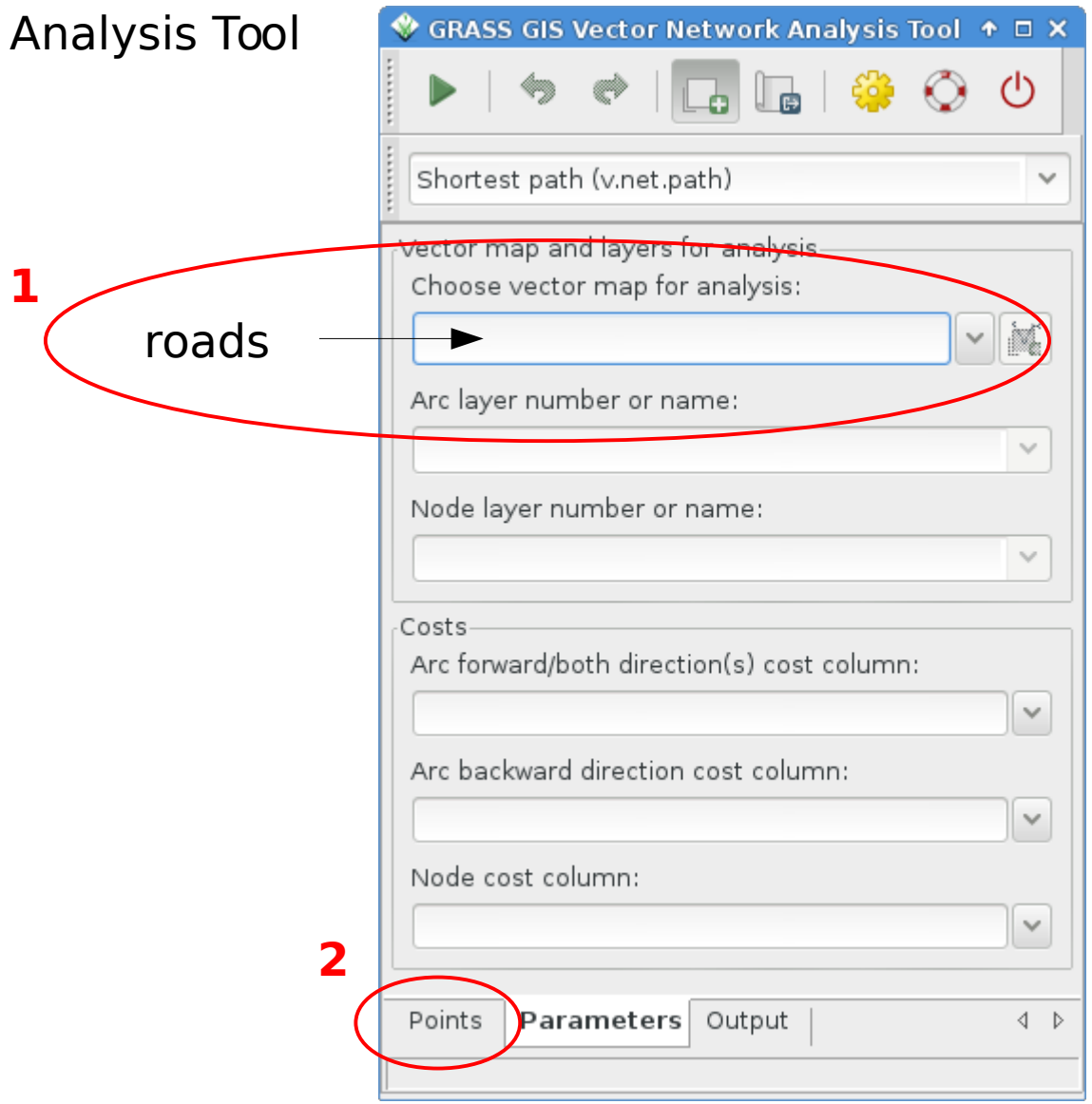
Vector network analysis in GRASS GIS

Procedure: Display the vector network, activate snapping to nodes (takes a moment) and define two points on the network



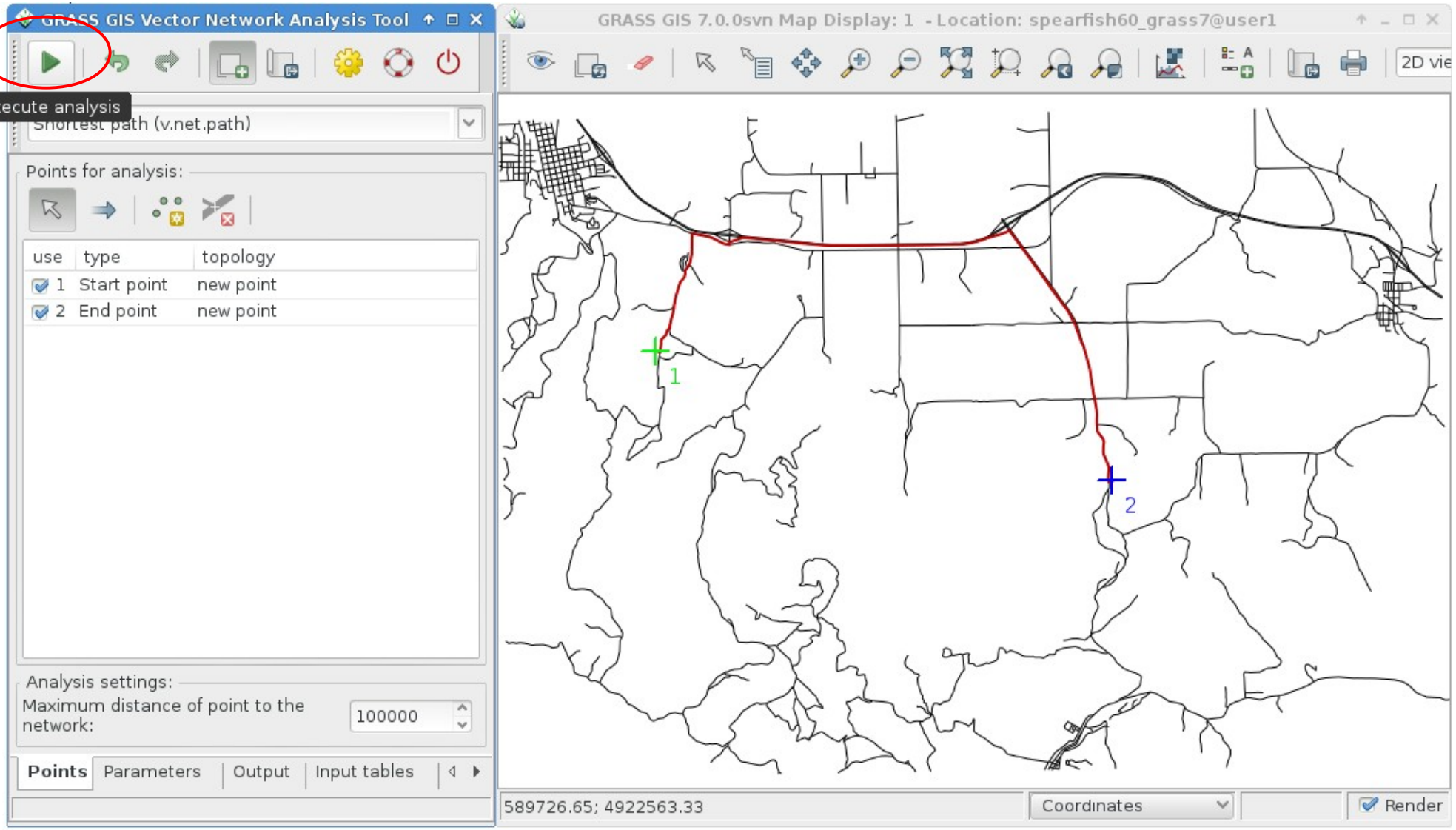
Vector network analysis in GRASS GIS

Vector Network Analysis Tool



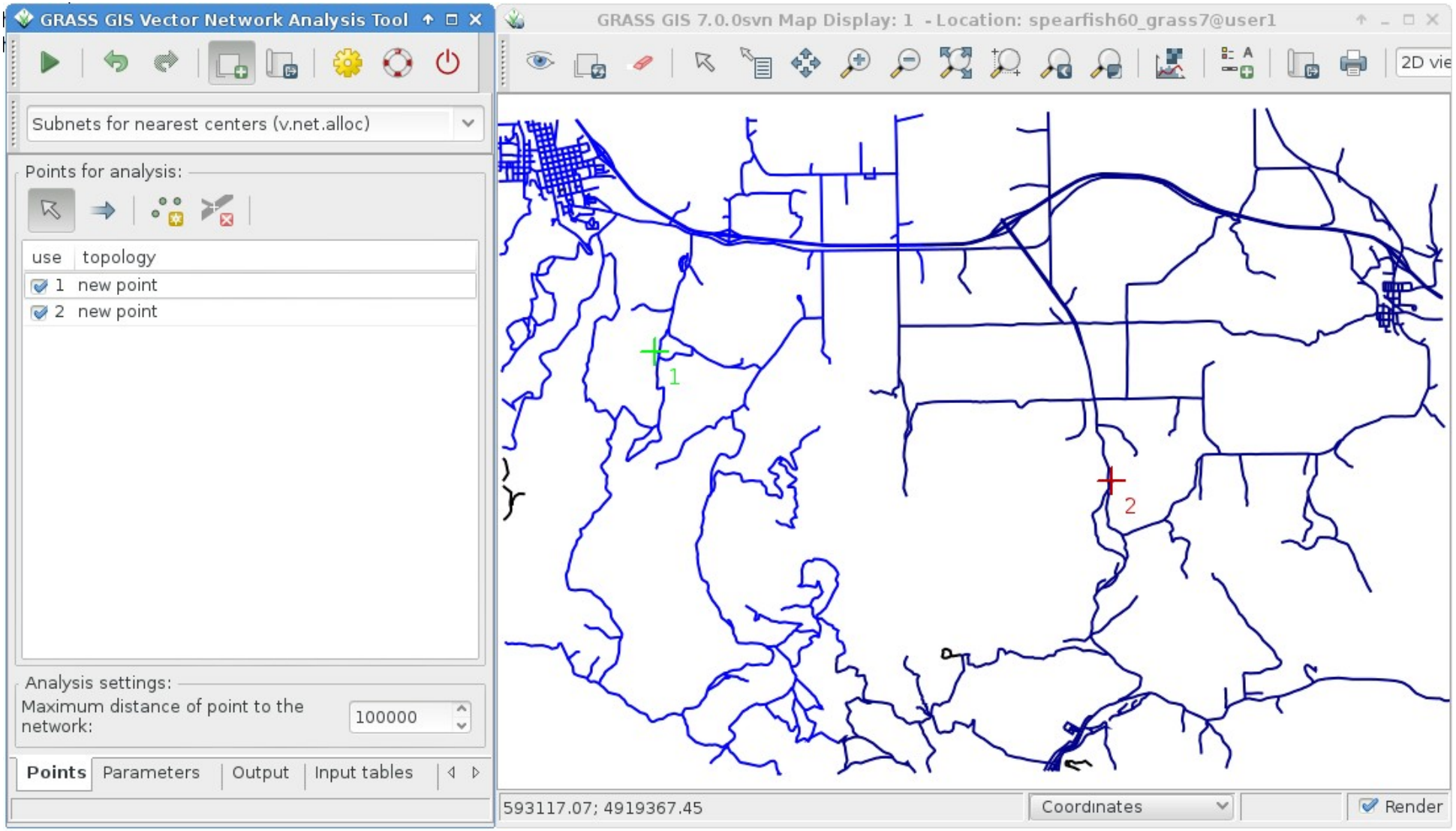
Vector network analysis in GRASS GIS

Vector Network Analysis Tool: shortest path



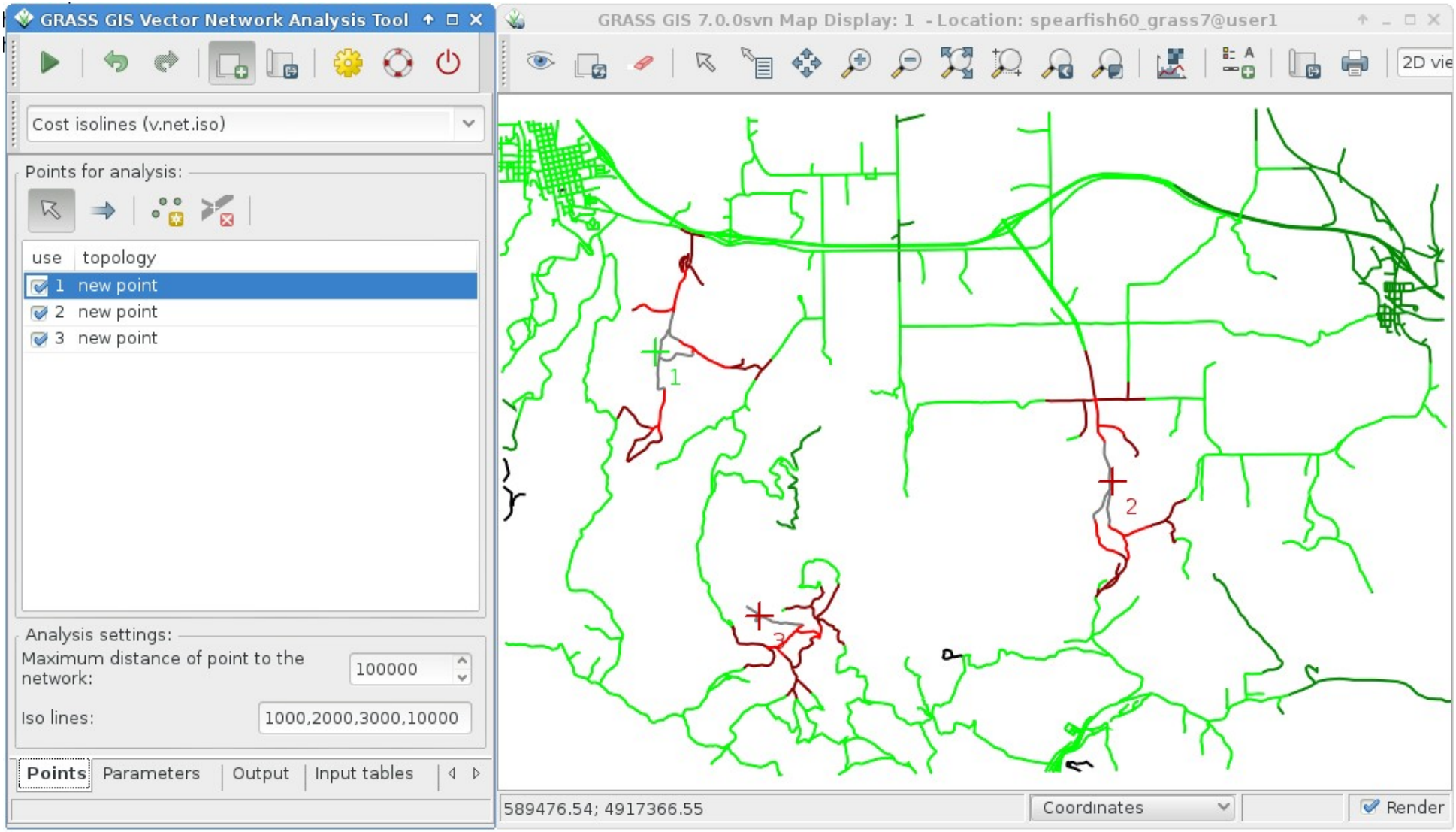
Vector network analysis in GRASS GIS

Vector Network Analysis Tool: subnet allocation



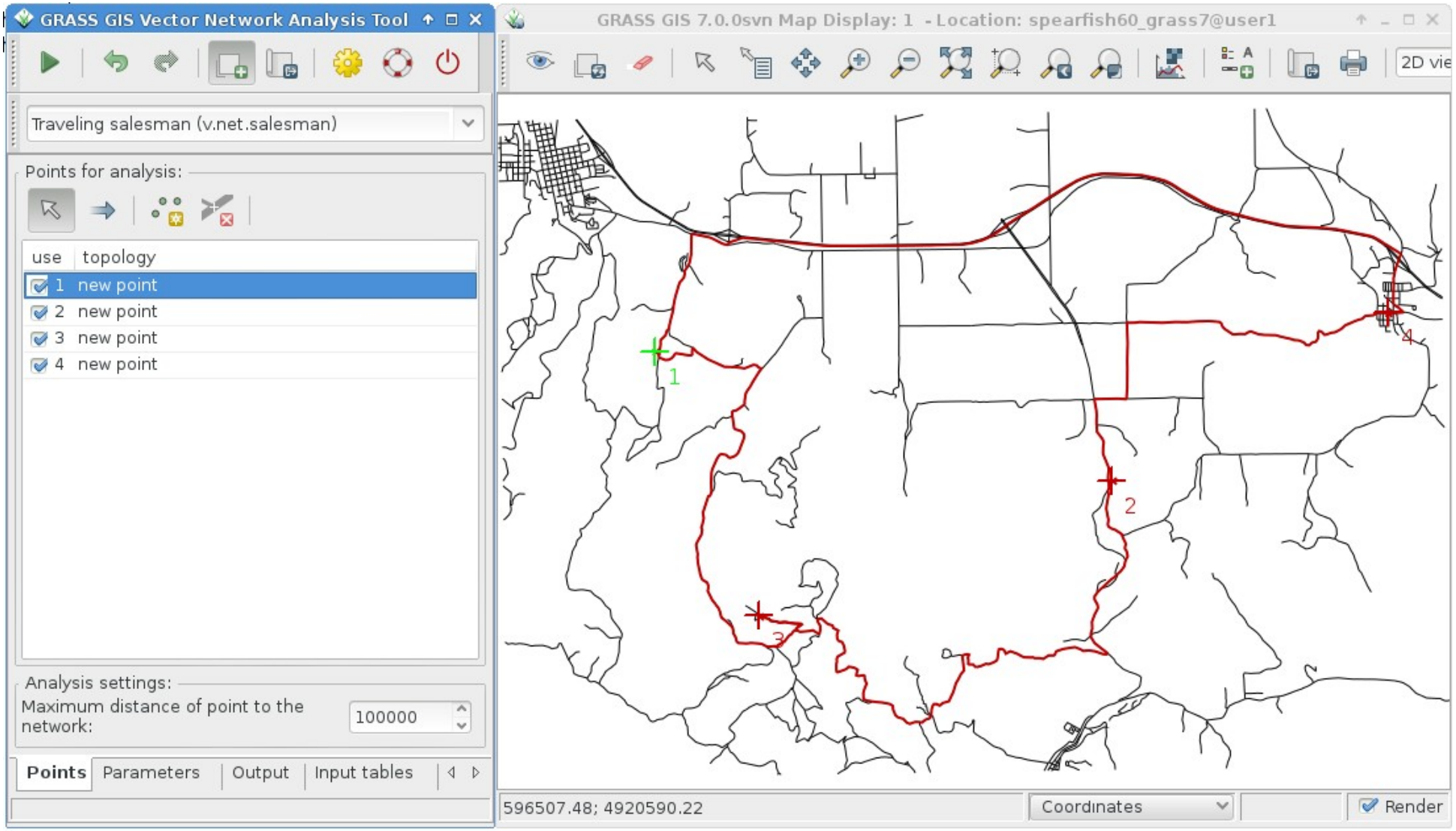
Vector network analysis in GRASS GIS

Vector Network Analysis Tool: cost isolines: 1km, 2km, 3km, 10km



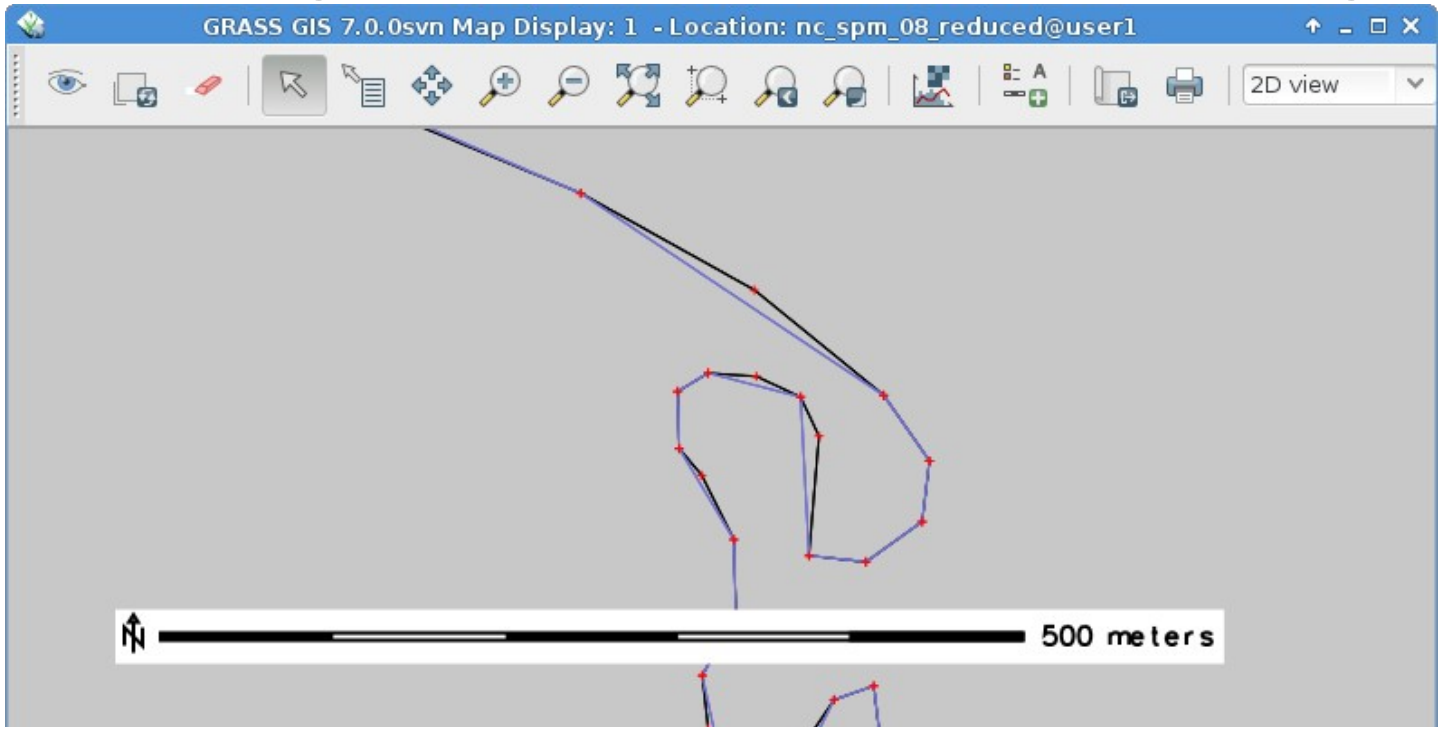
Vector network analysis in GRASS GIS

Vector Network Analysis Tool: travelling salesman, 4 points to visit



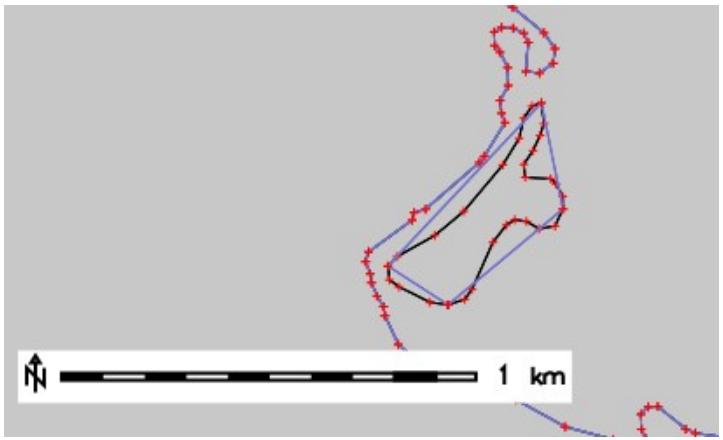
Vector simplification with v.generalize

Reducing the number of vertices with Douglas-Peucker



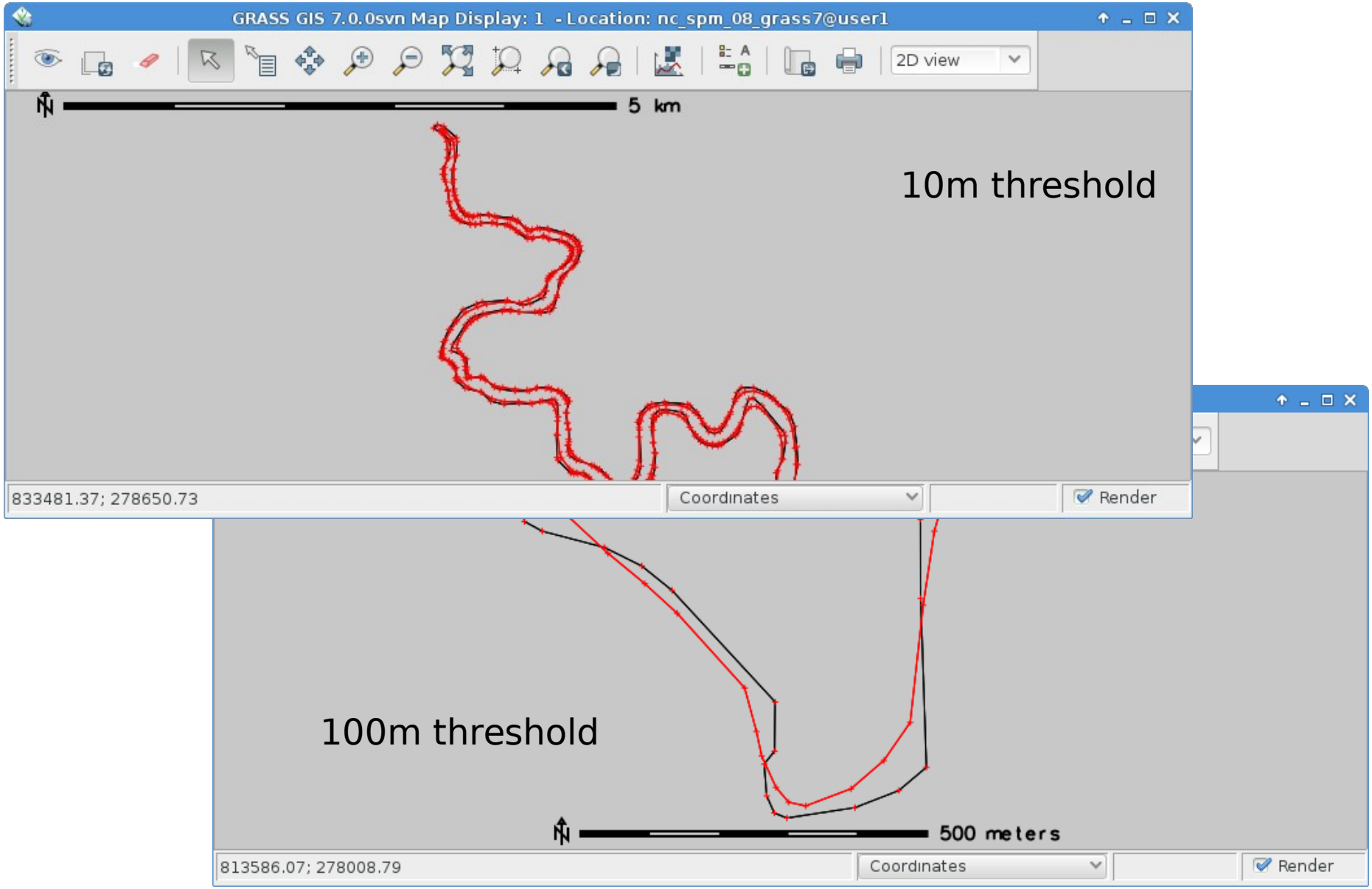
10m threshold

100m threshold



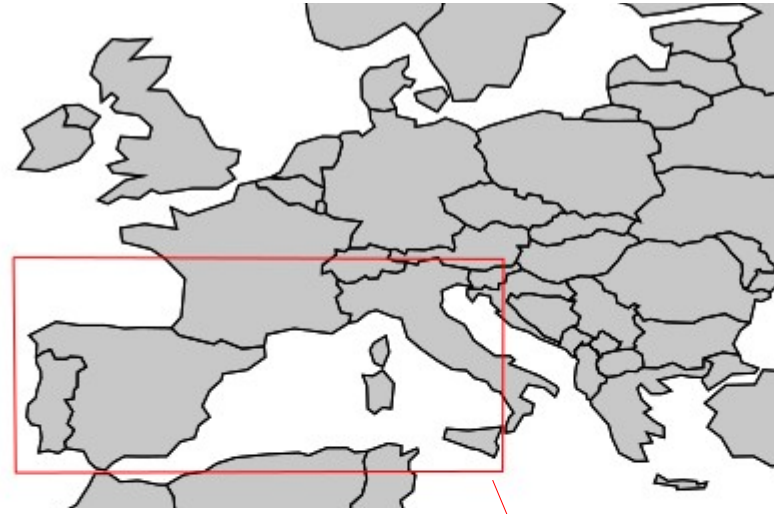
Vector smoothing with v.generalize

Increasing the number of vertices with sliding average



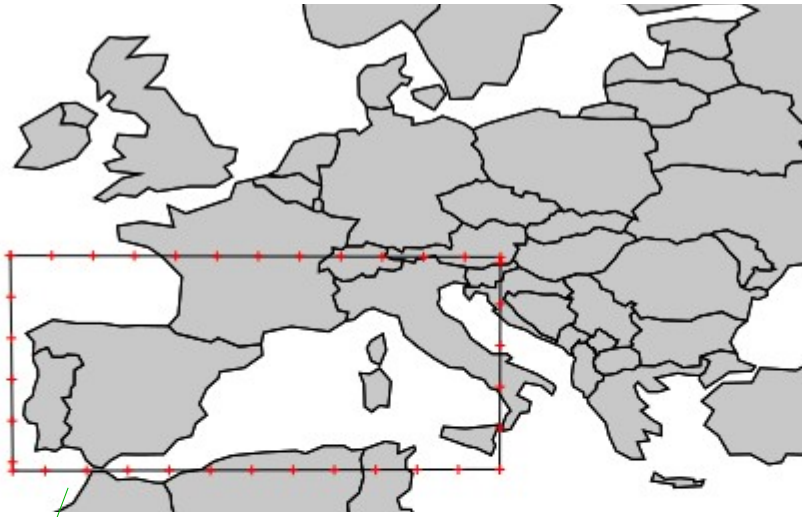
Vector reprojection: do it right!

A rectangular in LatLong being reprojected...



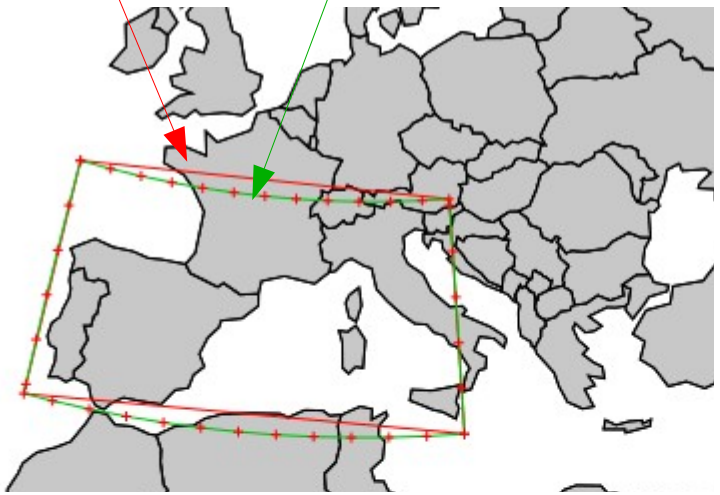
4 corner rectangle, LatLong (no further vertices there; an issue in many GIS)

Ouch...



Automated vertex densification in GRASS GIS 7 (v.proj)

OK



...to EU LAEA

New Python API integration

http://grass.osgeo.org/wiki/GRASS_and_Python

Page | Discussion | Read | Edit | View history | Go | Search

GRASS and Python

Navigation

Main Page
Community
Development
Documents
GRASS Help
Recent changes
Help

Toolbox

What links here
Related changes
Upload file
Special pages
Printable version
Permanent link

1 Python SIGs
2 Writing Python scripts
2.1 Python script ed
2.2 Using the GRA
2.3 Creating Python
2.3.1 MS-Wind
2.3.2 Linux
2.4 Running extern
2.5 Testing and inst
2.5.1 Debuggin
2.5.2 Installatio
3 Python extensions in
3.1 Python Scripting
3.2 Python Ctypes I
3.3 wxPython GUI
3.4 Python-GRASS
3.5 Using GRASS
4 FAQ
5 Links
5.1 General guides
5.2 Programming
5.3 Presentations

Introduction to Vector classes — Python library documentation documentation - Konqueror

File Edit View Go Bookmarks Tools Settings Window Help

http://grass.osgeo.org/grass71/manual/

Google

Python library documentation documentation » PyGRASS documentation »

previous | next | modules | index

Introduction to Vector classes

Details about the architecture can be found in the GRASS GIS 7 Programmer's Manual: GRASS Vector Library

Instantiation and basic interaction.

```
>>> from pygrass.vector import VectTopo
>>> municip = VectTopo('boundary_municp_sqlite')
>>> municip.is_open()
False
>>> municip.mapset
''
>>> municip.exist()  # check if exist, and if True set mapset
True
>>> municip.mapset
'user1'
```

Open the map with topology:

```
>>> municip.open()

get the number of primitive:
```

Previous topic
Introduction to Raster classes

Next topic
Interface to GRASS GIS modules

Quick search

Go

Enter search terms or a module, class or function name.

Using Python and GRASS GIS 7 with ipython

An interactive (Web based!) shortcourse on writing GRASS scripts in Python

https://github.com/wenzeslaus/python-grass-addon

https://github.com/wenzeslaus/python-grass-addon/blob/master/01_scripting_library.ipynb

Introduction to the GRASS GIS 7 Python Scripting Library

The GRASS GIS 7 Python Scripting Library provides functions to call GRASS modules within scripts as subprocesses. The most often used functions include:

- **run_command**: most often used with modules which output raster/vector data where text output is not expected
- **read_command**: used when we are interested in text output
- **parse_command**: used with modules producing text output as key=value pair
- **write_command**: for modules expecting text input from either standard input or file

Besides, this library provides several wrapper functions for often called modules.

Calling GRASS GIS modules

We start by importing GRASS GIS Python Scripting Library:

In [ ]:
```
import grass.script as gscript
```

Before running any GRASS raster modules, you need to set the computational region using g.region. In this example, we set the computational extent and resolution to the raster layer elevation.

In [ ]:
```
gscript.run_command('g.region', raster='elevation')
```

The run_command() function is the most commonly used one. Here, we apply the focal operation *average* (r.neighbors) to smooth the elevation raster layer. Note that the syntax is similar to bash syntax, just the flags are specified in a parameter.

GRASS GIS goes supercomputer: HPC

- Since **2005** (10 years) GRASS GIS is running **natively** on 64bit CPUs
- GRASS GIS 7 offers Large File Support also on 32bit Windows

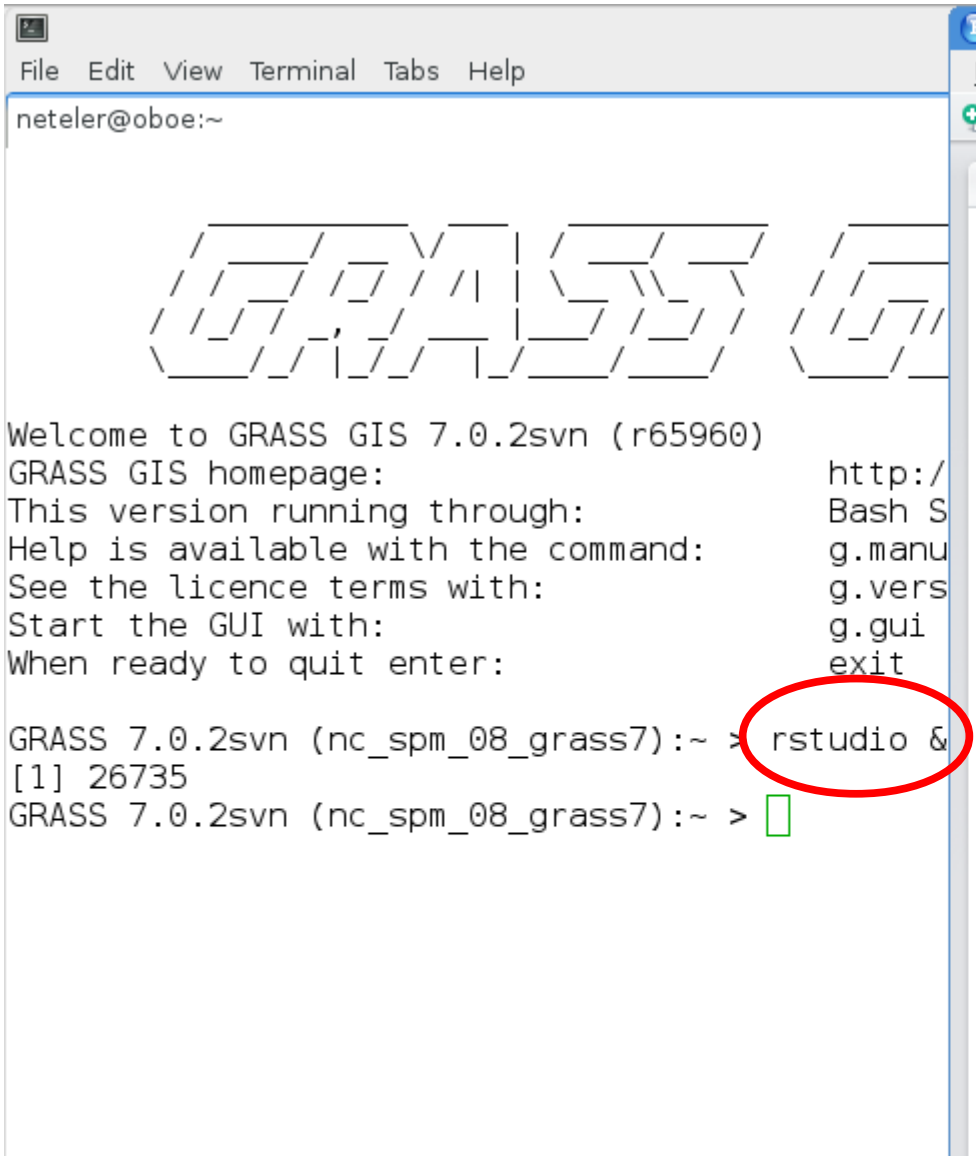
- Installed on Grids and TOP500 supercomputers (AKKA Umeå, ENEA Frascati, Aurel Bratislava, ...)
- Runs on Linux, AIX, Solaris, freeBSD, netBSD, (MS-Windows)...
- Various ways of parallelization

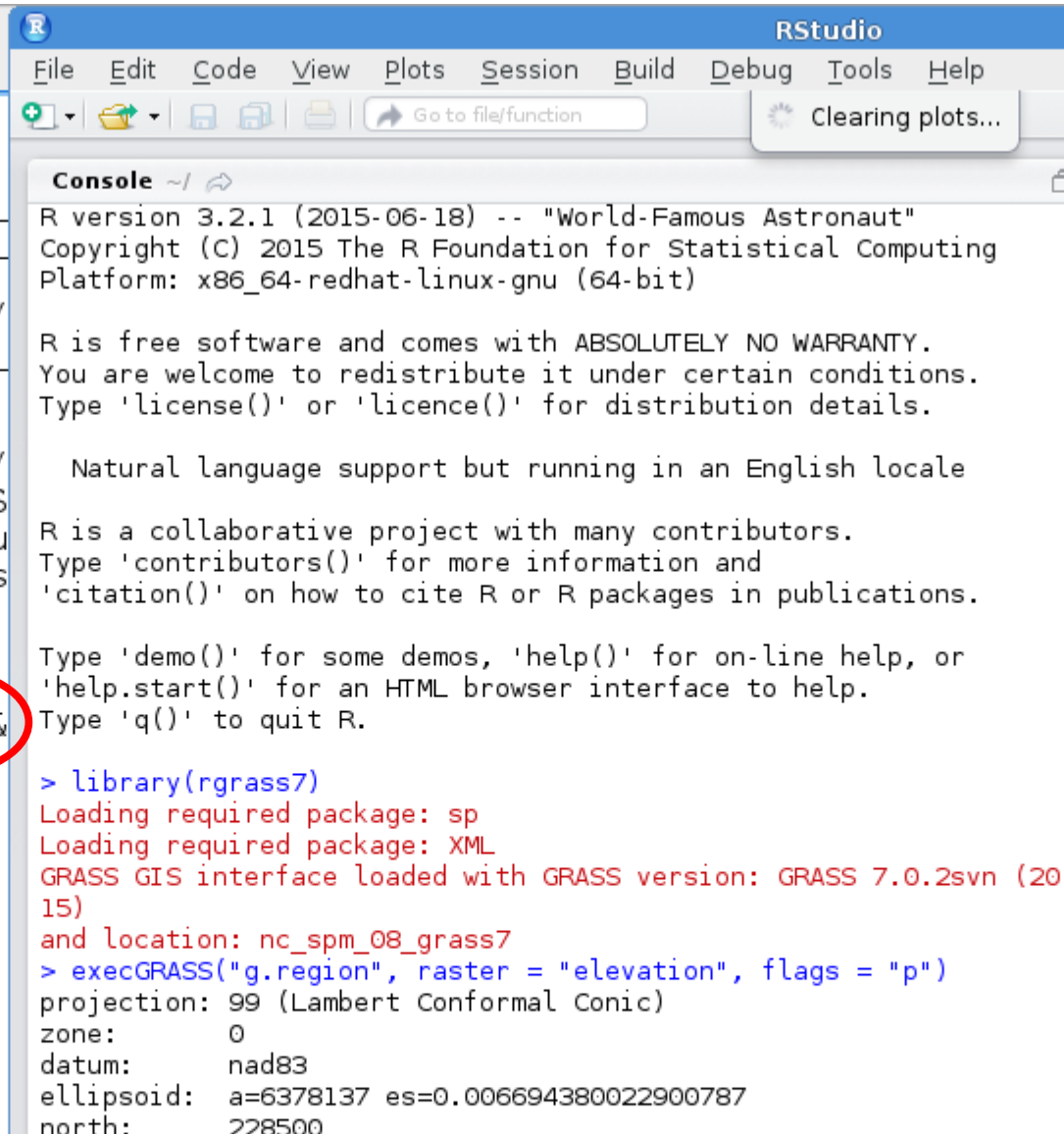
Hints: http://grasswiki.osgeo.org/wiki/Compile_and_Install

GRASS GIS 7 and R statistics

Using R (Rstudio) within a GRASS GIS session

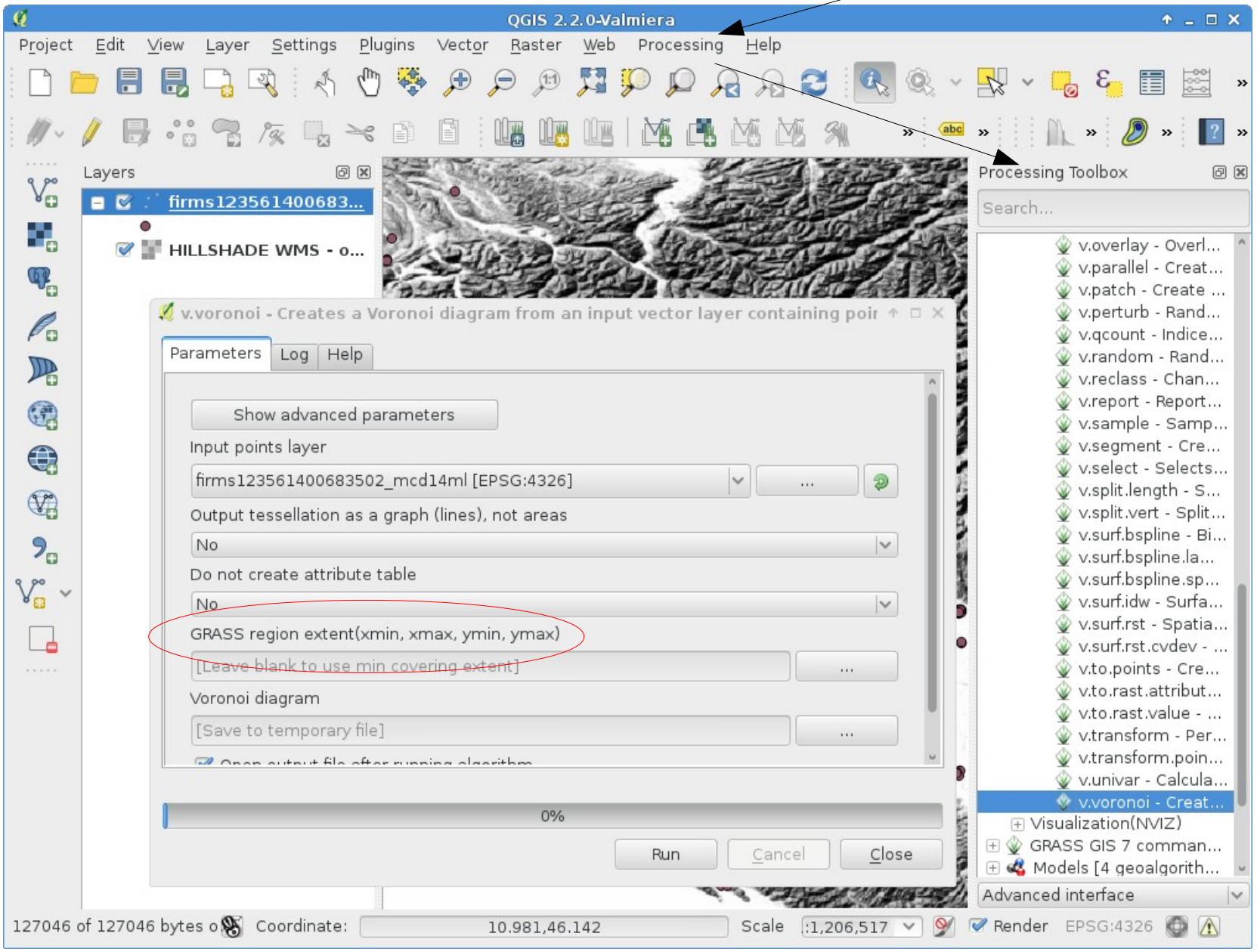
https://grasswiki.osgeo.org/wiki/R_statistics/rgrass7





Using QGIS with Processing/GRASS

Geoprocessing of the fire point layer: QGIS → Processing → GRASS GIS



GRASS Addons: User contributed extensions

The Addons repository is SVN based:

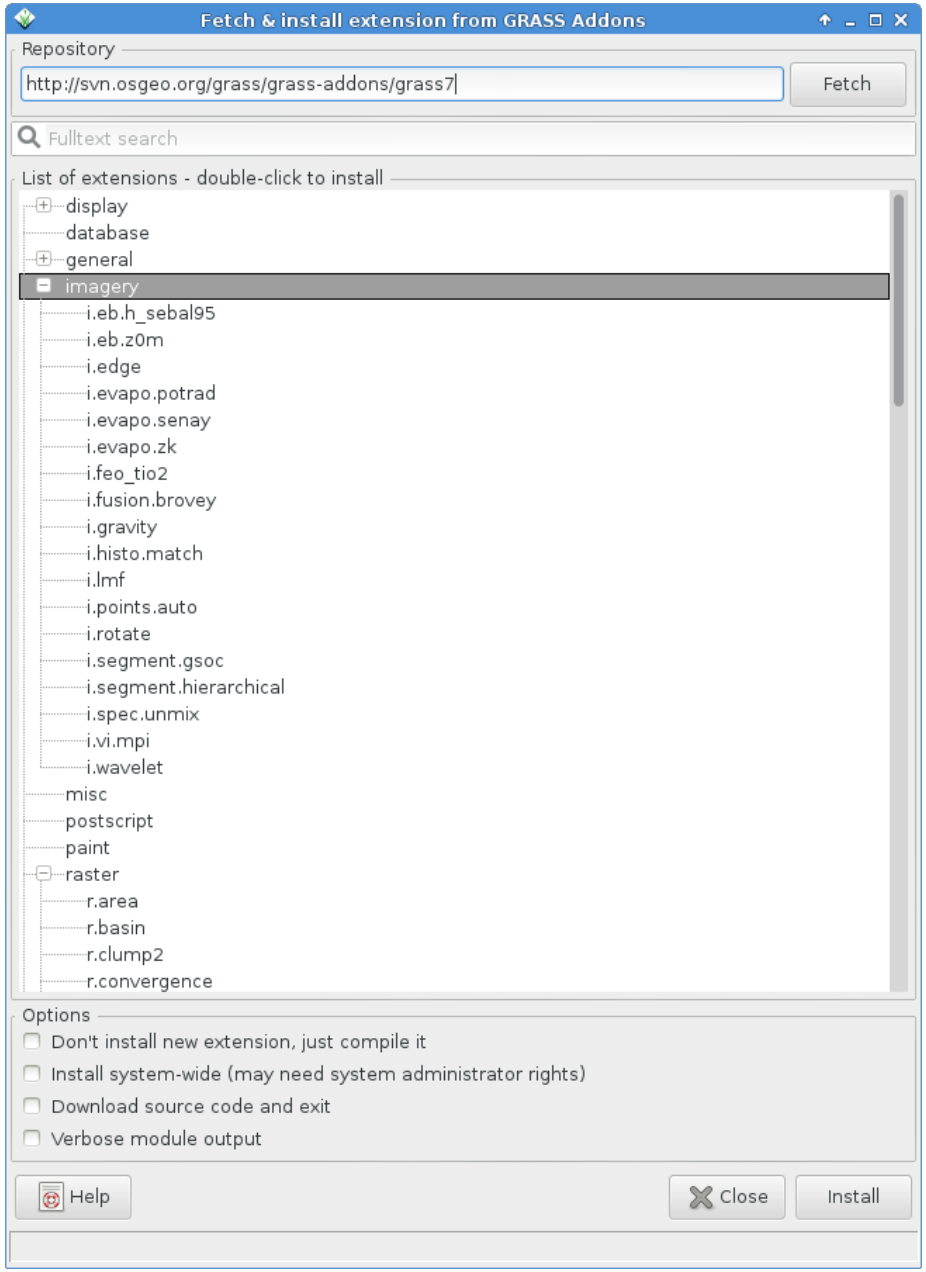
One-click installation with extension manager

Increasing inflow of Python scripts

Users can easily obtain **write** access to develop new functionality

Peer review through SVN commit email list

Also github, gitlab etc. now supported



https://grass.osgeo.org/grass70/manuals/addons/

Development meetings: Community sprints

Where's the stuff?

GRASS GIS 7 Software:

Free download for MS Windows, MacOSX, Linux and source code:

https://grass.osgeo.org/download/

Addons (user contributed extensions):

https://grass.osgeo.org/grass70/manuals/addons/

Free sample data:

Rich data set of North Carolina (NC)

... available as GRASS GIS location and in common GIS formats

https://grass.osgeo.org/download/sample-data/

User Help:

Mailing lists (also in different languages):

https://grass.osgeo.org/support/

Wiki:

https://grasswiki.osgeo.org/wiki/

https://grasswiki.osgeo.org/wiki/R_statistics/rgrass7

Manuals:

https://grass.osgeo.org/documentation/manuals/