GRASS GIS 7 workshop

GRASS GIS 7 intro

Markus Neteler

OSGeo Ireland – 1st National Symposium 2017

mundialis GmbH & Co. KG
http://www.mundialis.de

Session Objectives

- Database structure of GRASS GIS
- About the course data set
- First steps in using GRASS GIS 7
  - the graphical user interface (GUI)
    - displaying raster and vector maps
    - querying maps
    - adding map elements
    - map swiping with two different Landsat coverages
    - bivariate scatterplots
  - GRASS GIS command structure
    - command line or GUI?
  - Creating a perspective view

GRASS GIS Database concept

GRASS Database

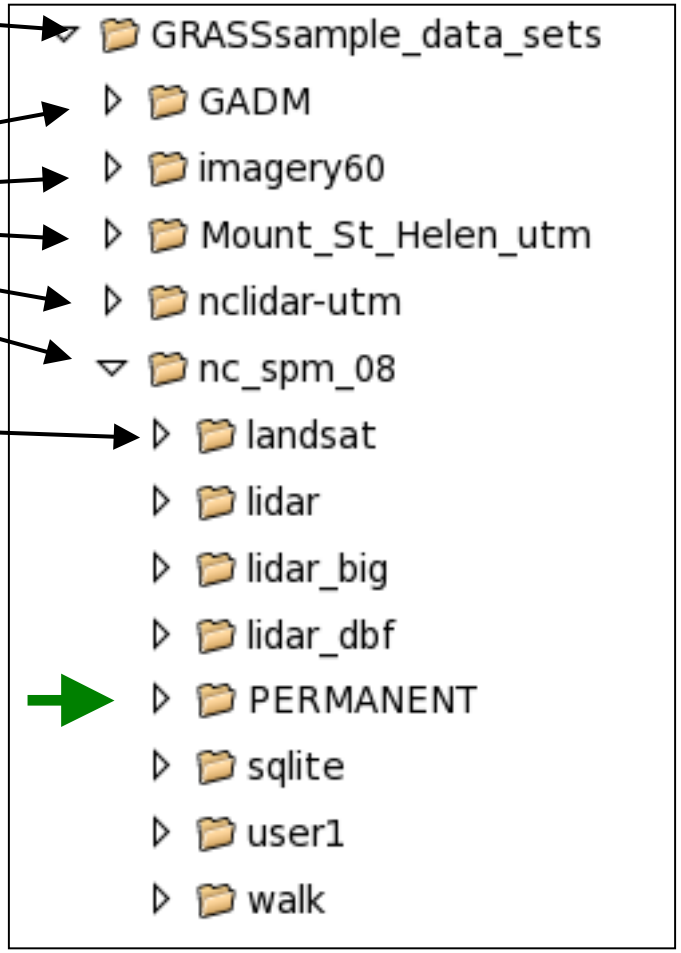
folder with Locations (“projects“)

Location

folder with Mapsets

Mapset

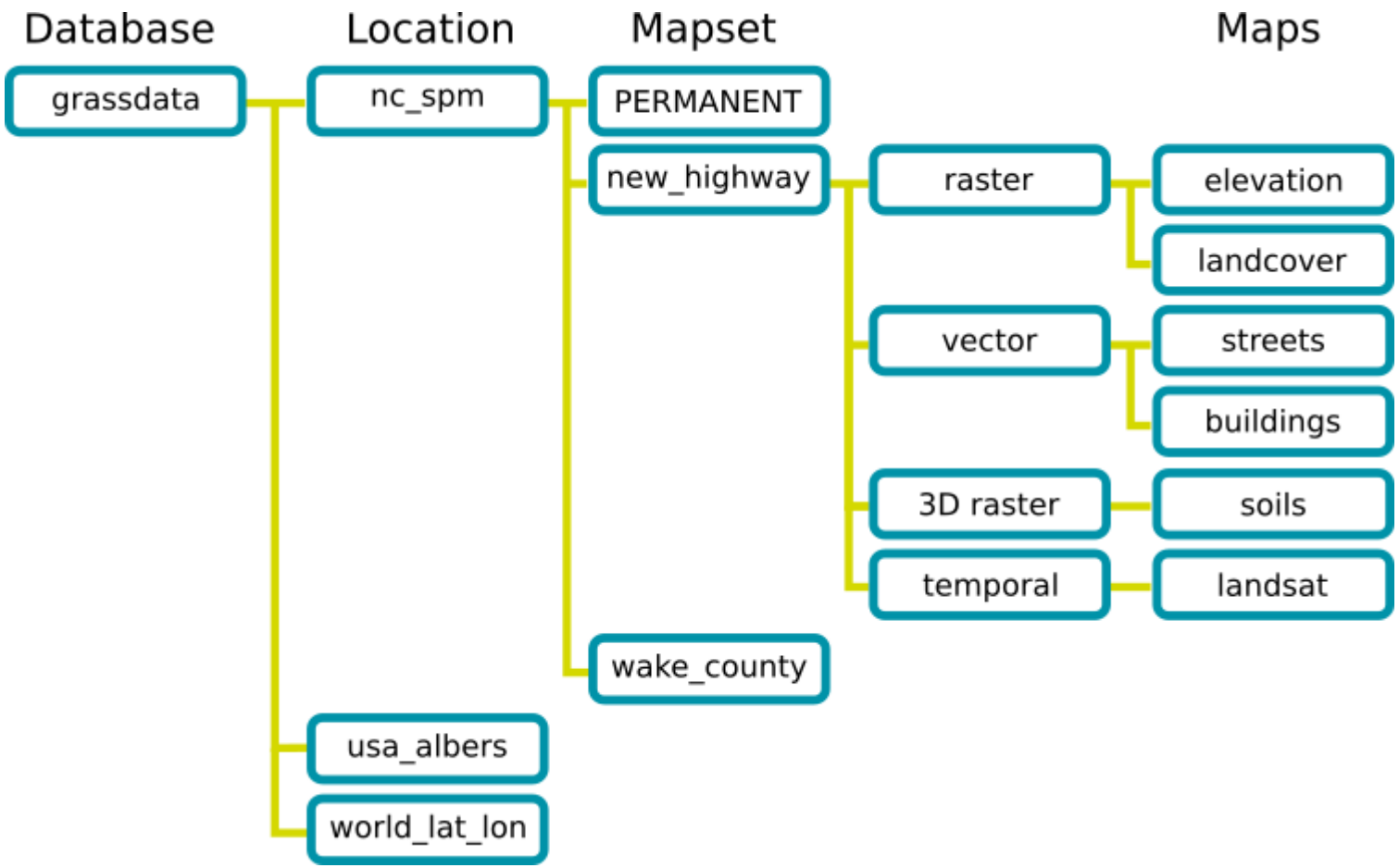
collection of maps and support data



Each Location...

- is defined by a projection
- can hold several mapsets
- has a PERMANENT mapset for base cartography

GRASS 7 Database concept: graphical view



... effectively a tree of subdirectories.

GRASS GIS is a multi-user system when storing "**grassdata**" on a network drive. Users can share "**locations**" and manage therein own "**mapsets**" for the data.

GRASS GIS will do the file management for you!

Source: https://grasswiki.osgeo.org/wiki/File:Grass_database.png

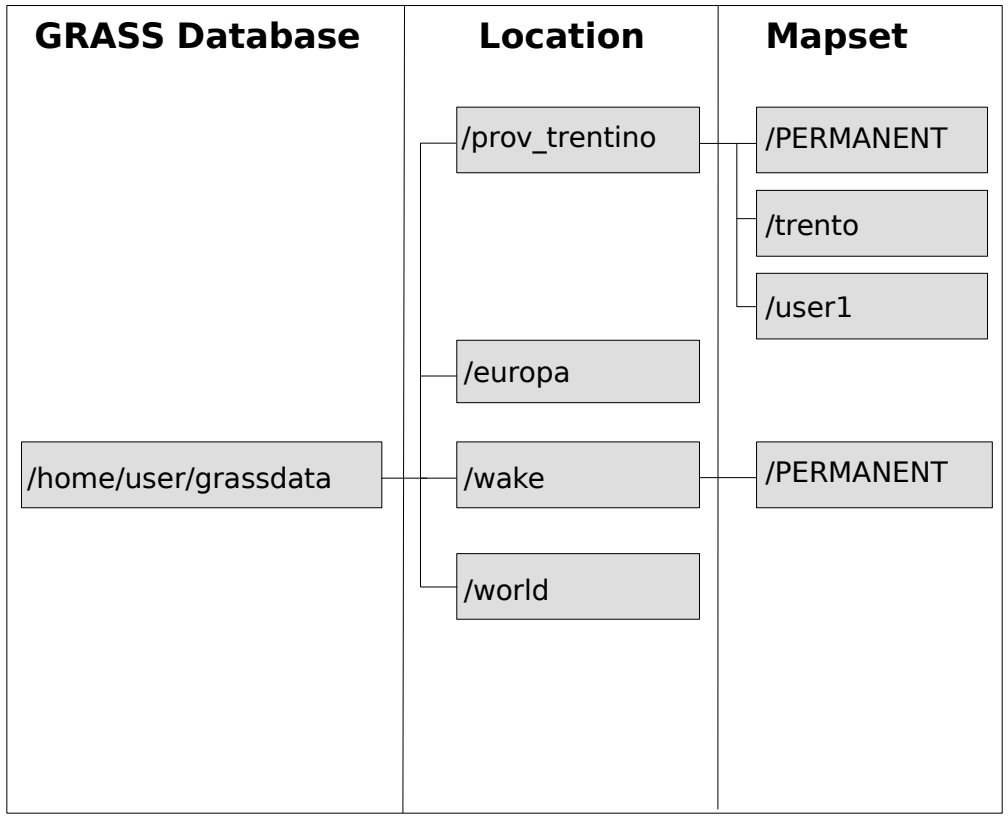
Standard GIS formats versus GRASS 7 Database

Raster
- GeoTIFF
- HDF
- NetCDF
- BIL
- ...

Vector
- SHAPE
- E00
- MapInfo
- Spatialite
- ...

Export

Import



Data in standard GIS formats.

Store in directory:
`/home/user/gisdata/`

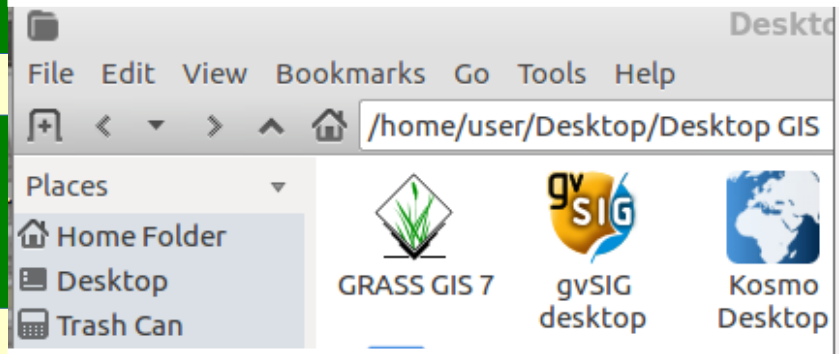
or a shared network directory

Data in GRASS GIS formats.

Store in directory:
`/home/user/grassdata/`

or a shared network directory

Exercise – GRASS startup and first steps

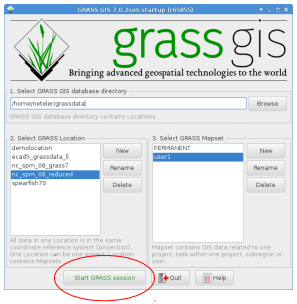


Alternative to GUI: cmd line

```
> grass70 -gui
```

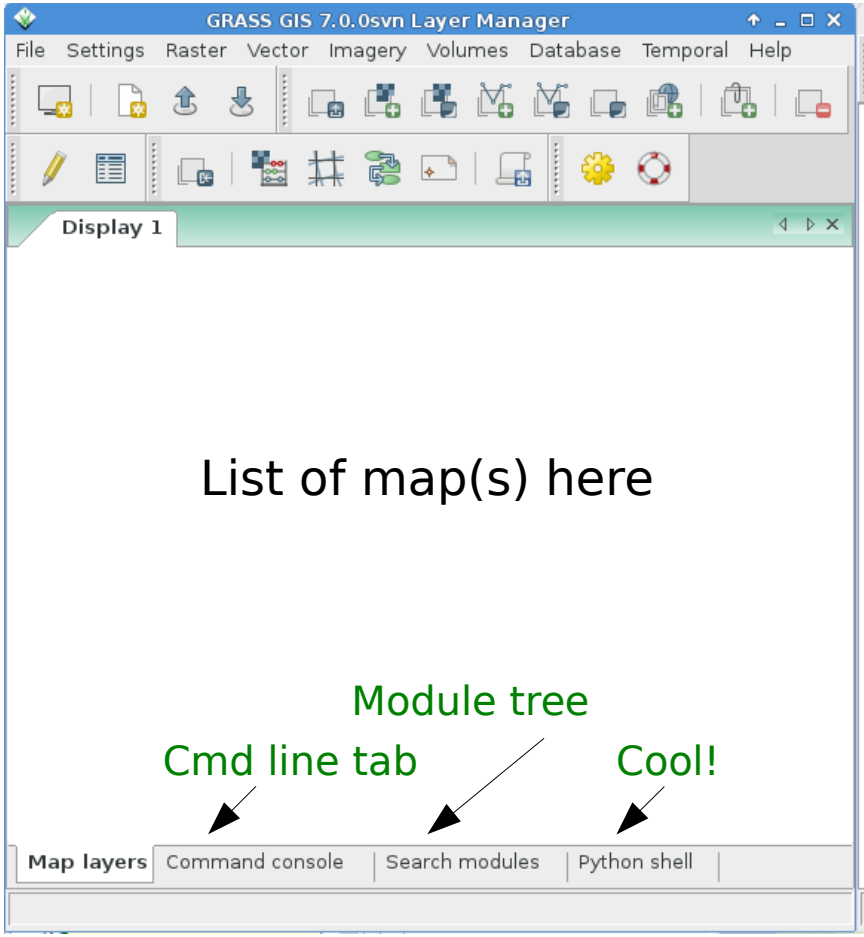
Exercise – GRASS 7 startup and first steps

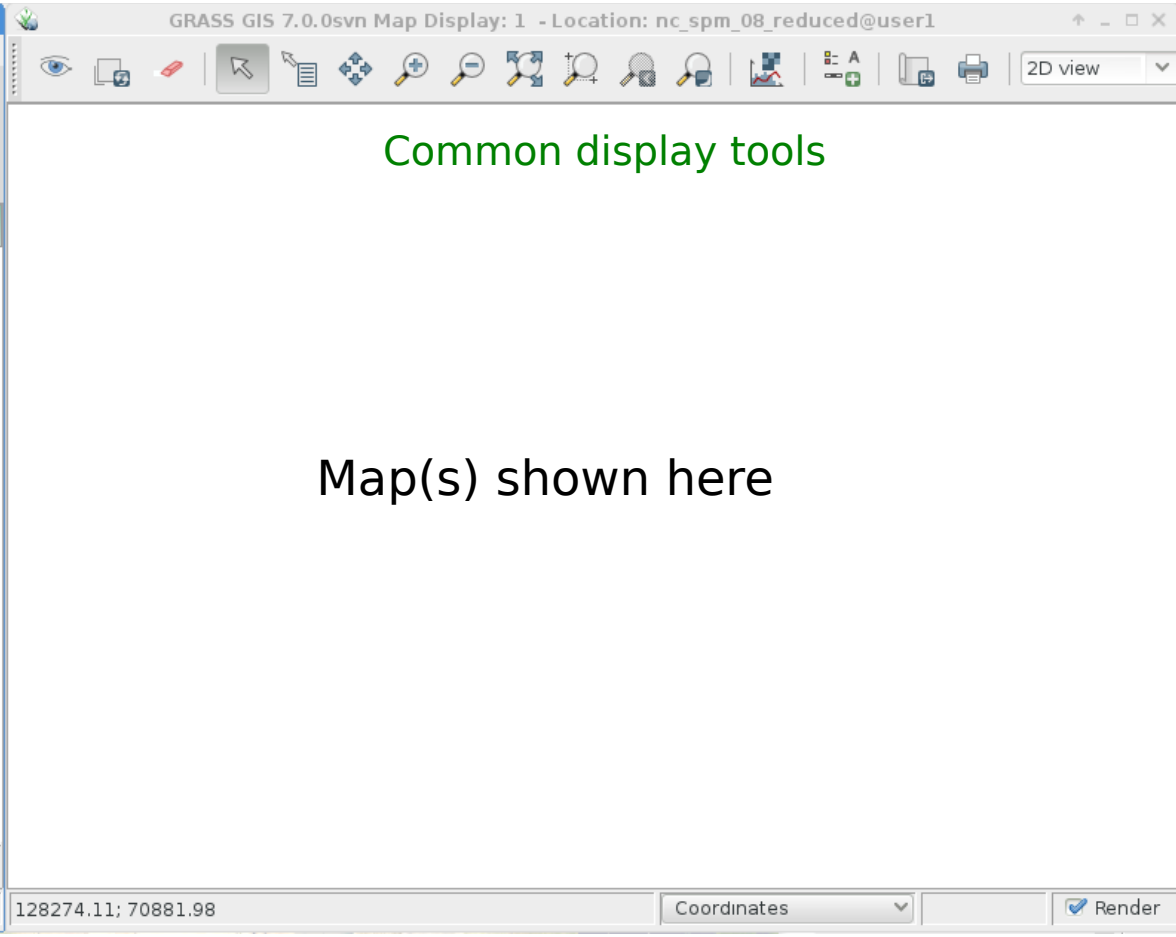
The graphical user interface at startup:



Main menu

Map display





Common display tools

List of map(s) here

Map(s) shown here

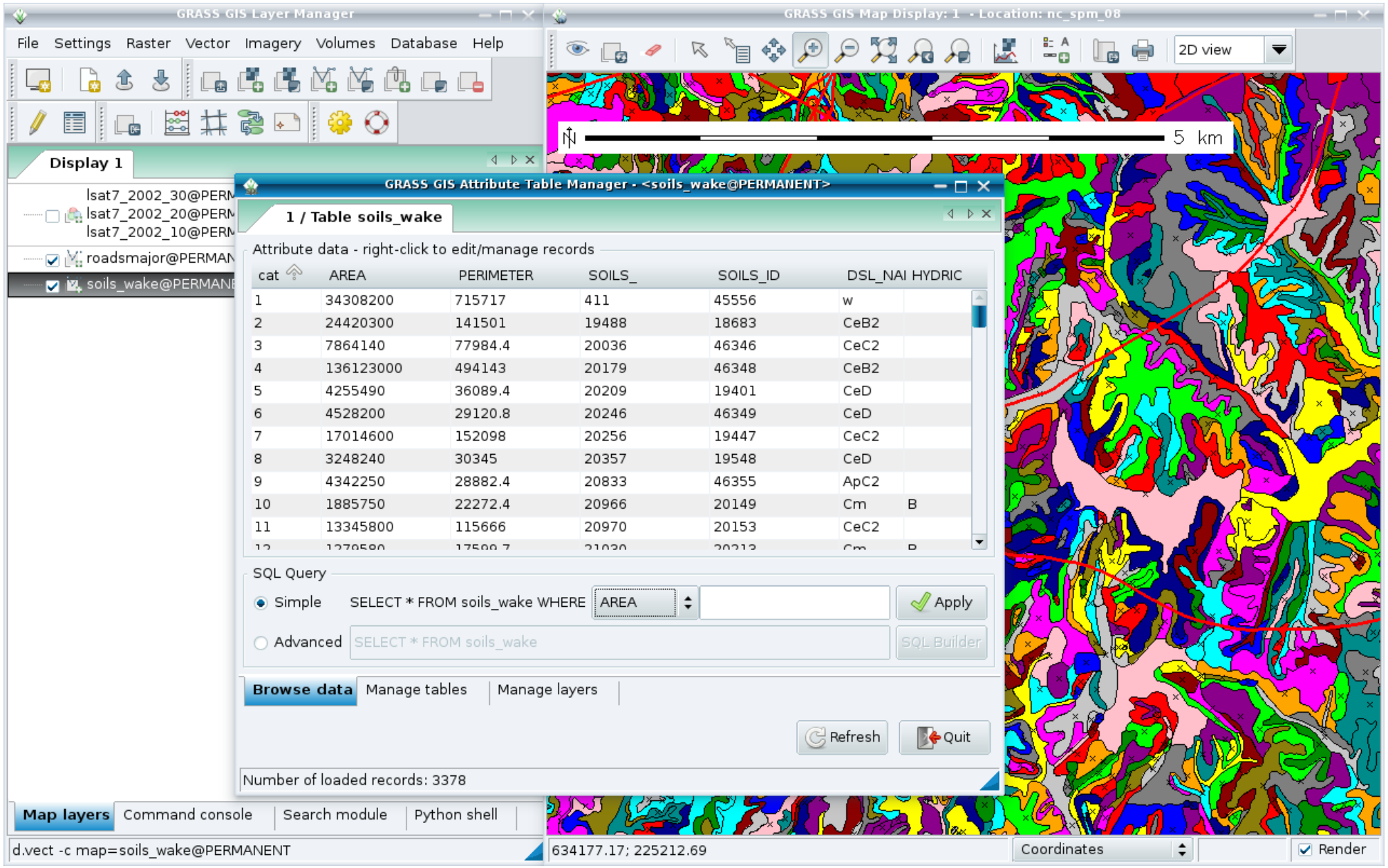
Module tree

Cmd line tab

Cool!

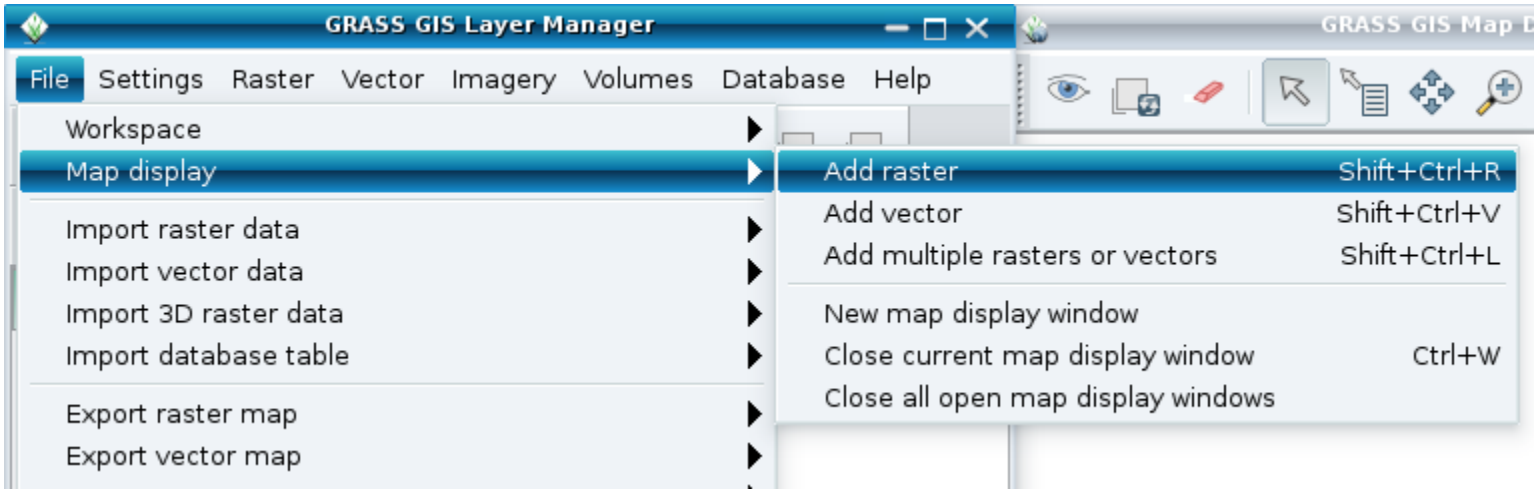
GRASS GIS GUI at full usage

(we'll reach that shortly)



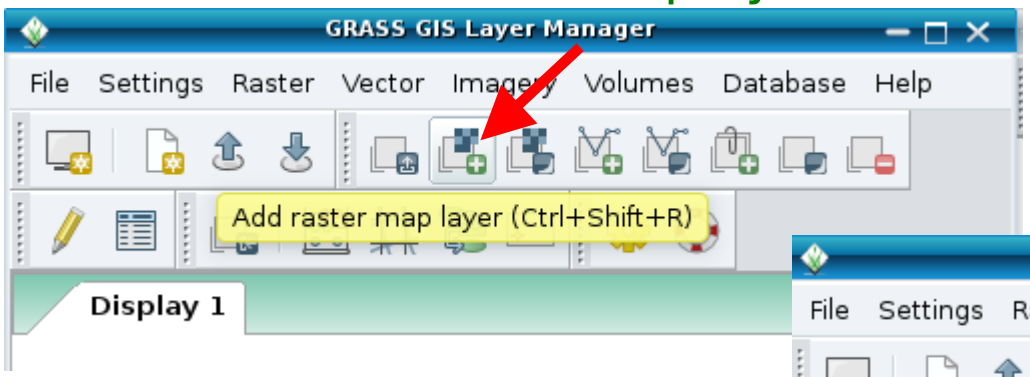
Displaying raster and vector maps

A) Using the menu

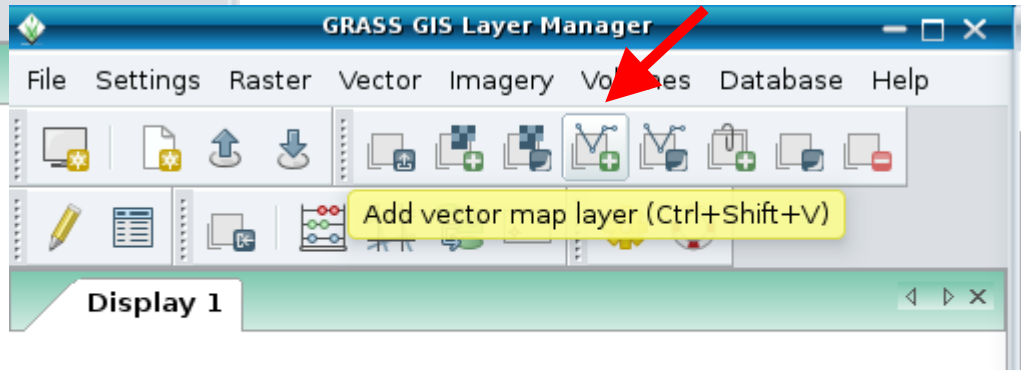


B) Using the icons

Display raster maps

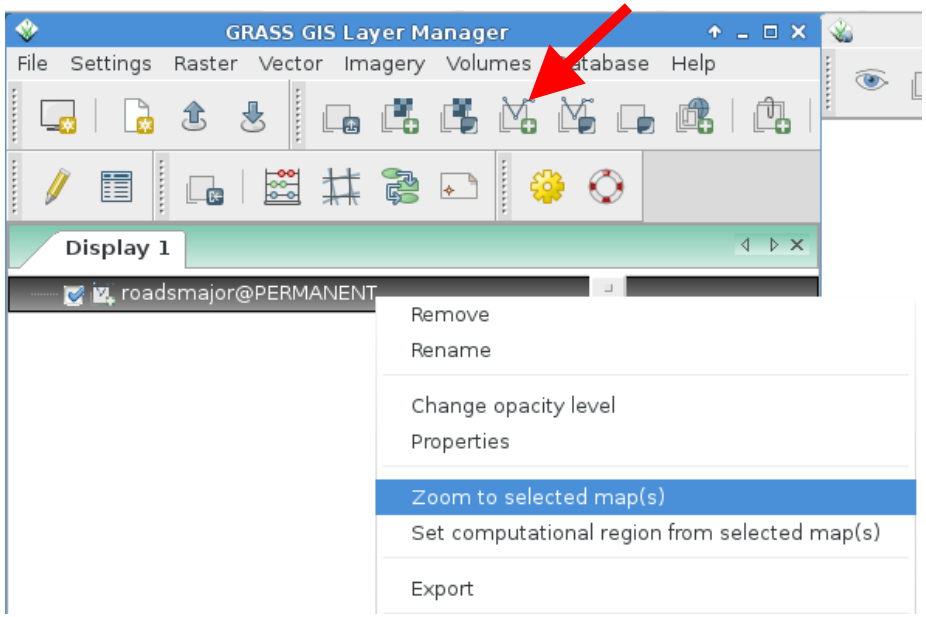


Display vector maps

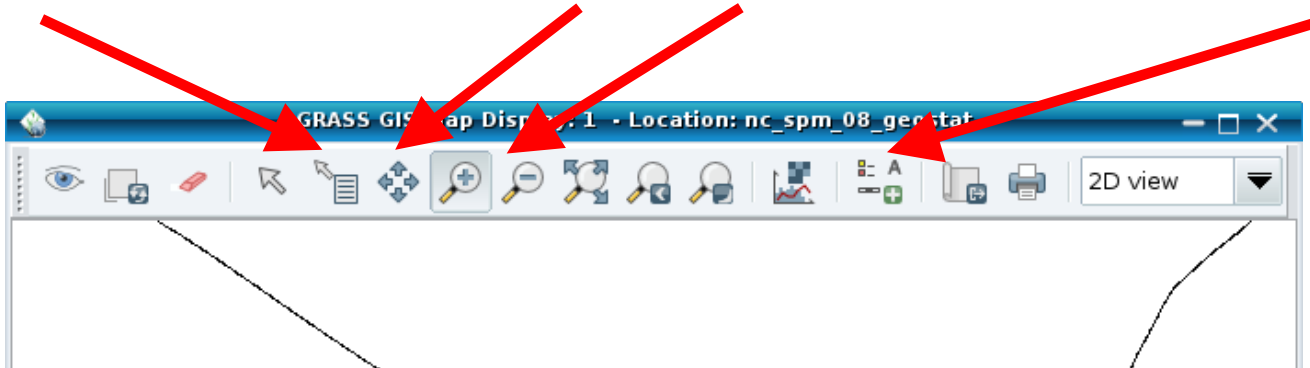


Exercise – Display the “roadsmajor” vector map

- Load the “roadsmajor” vector map into the canvas, zoom to map if needed:



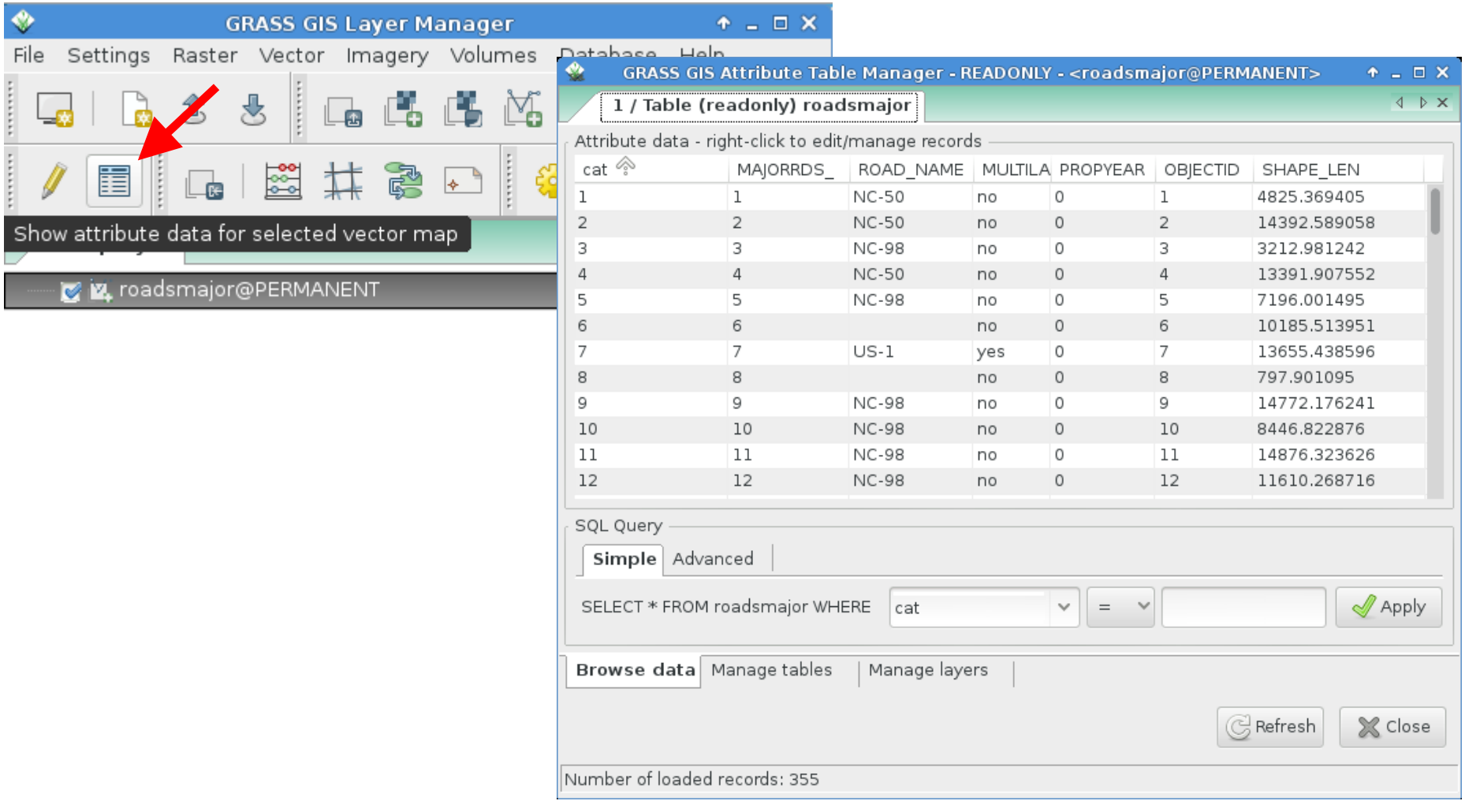
- **Query** map elements, **pan**, **zoom** in and out, add a **scale bar**



Exercise – Show vector map attributes

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

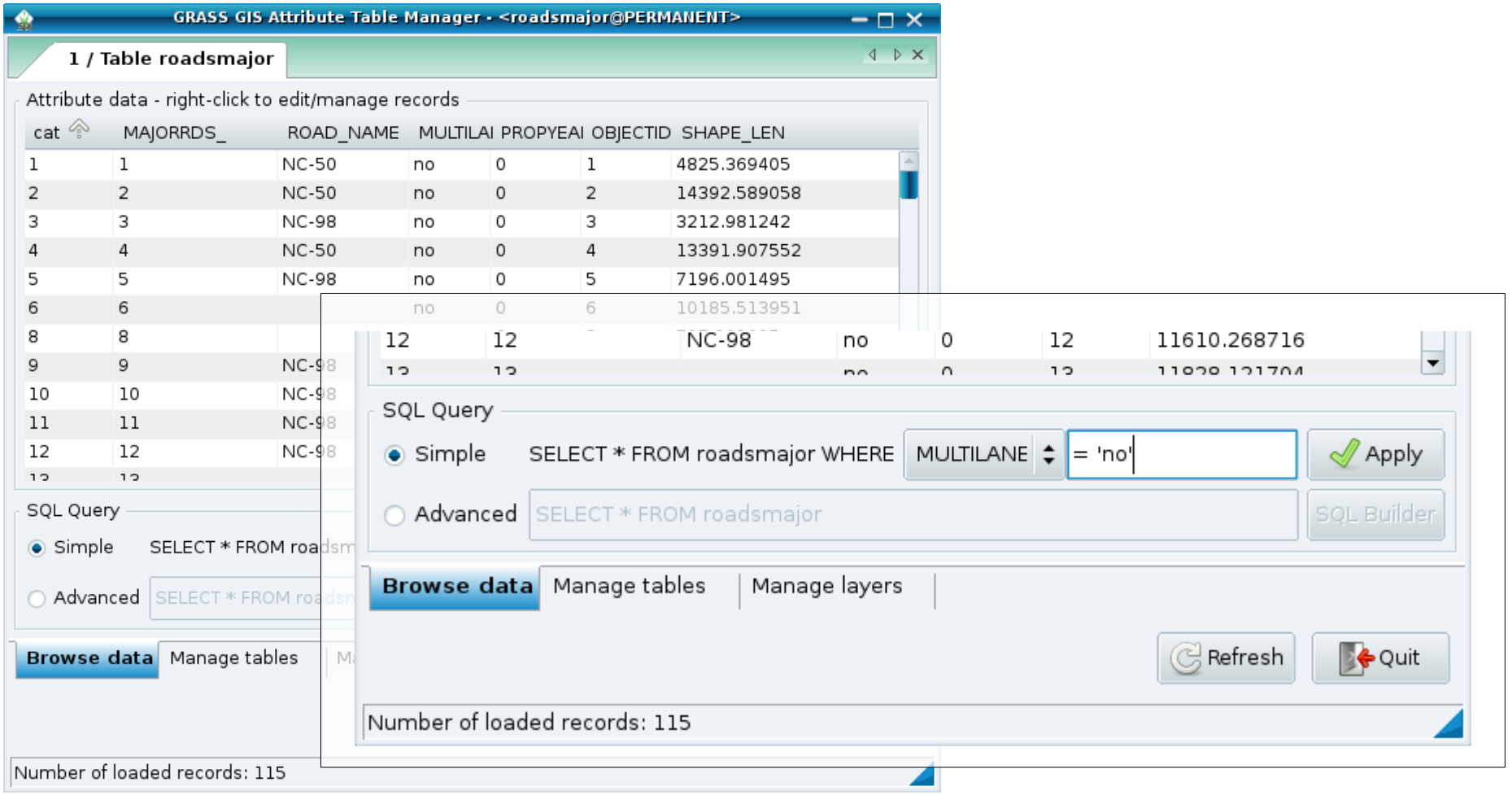
- … either *right-mouse* clicking in layer tree on map name
- … or using the related “Show Attribute table” icon



Exercise – SQL queries of attributes

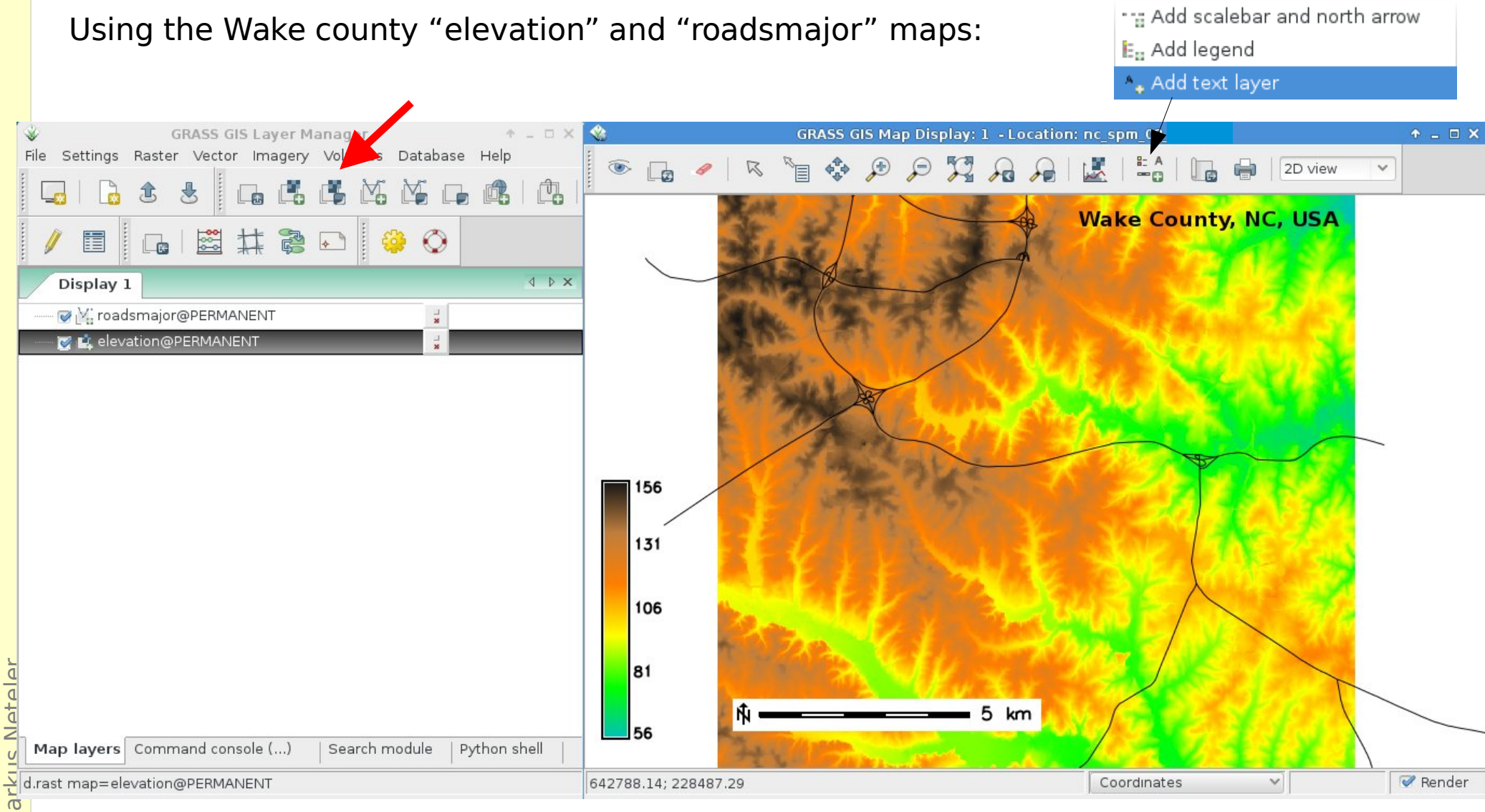
Select the **single lane roads** in the “roadsmajor” vector map

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



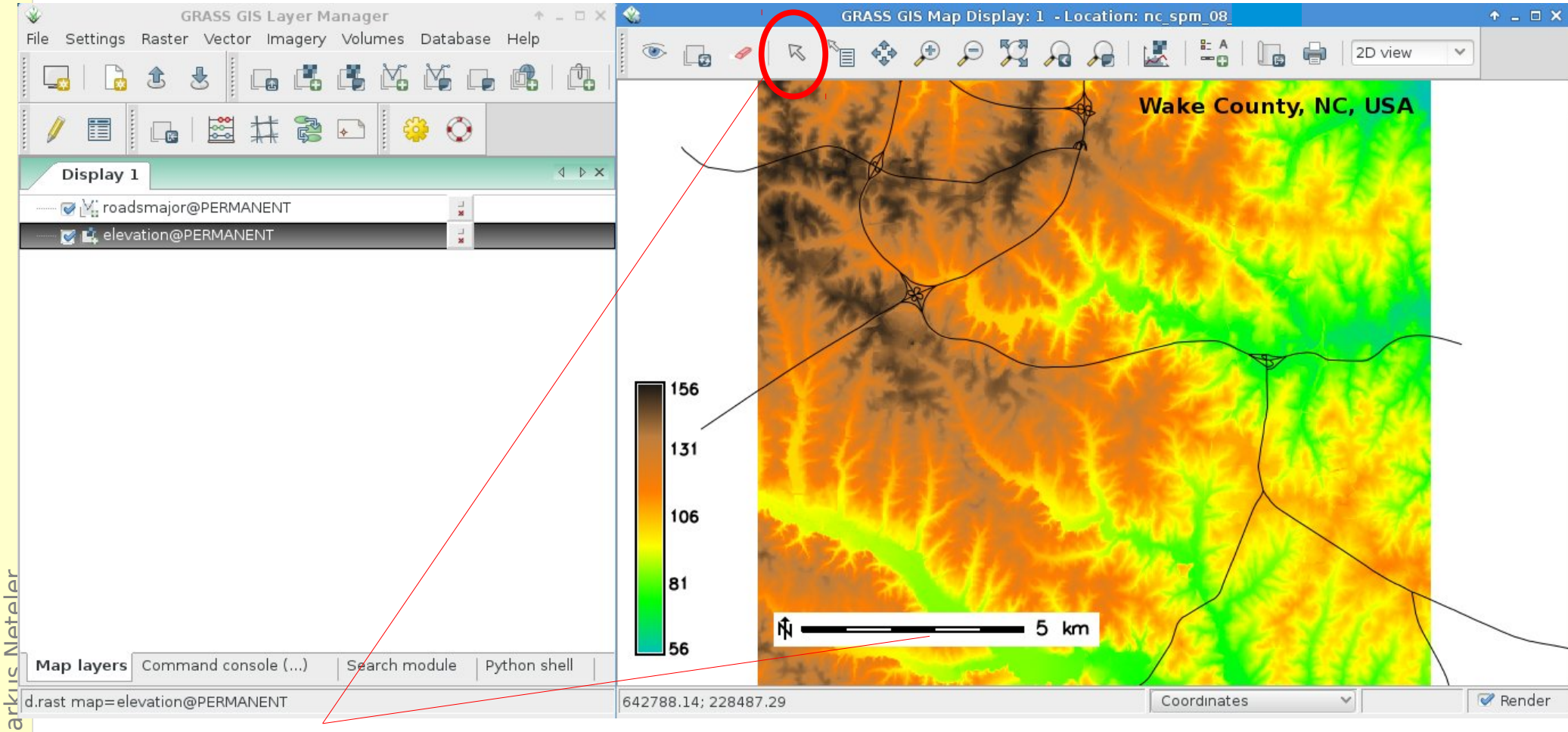
Exercise – Adding map elements

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



Exercise – Modify element settings and position

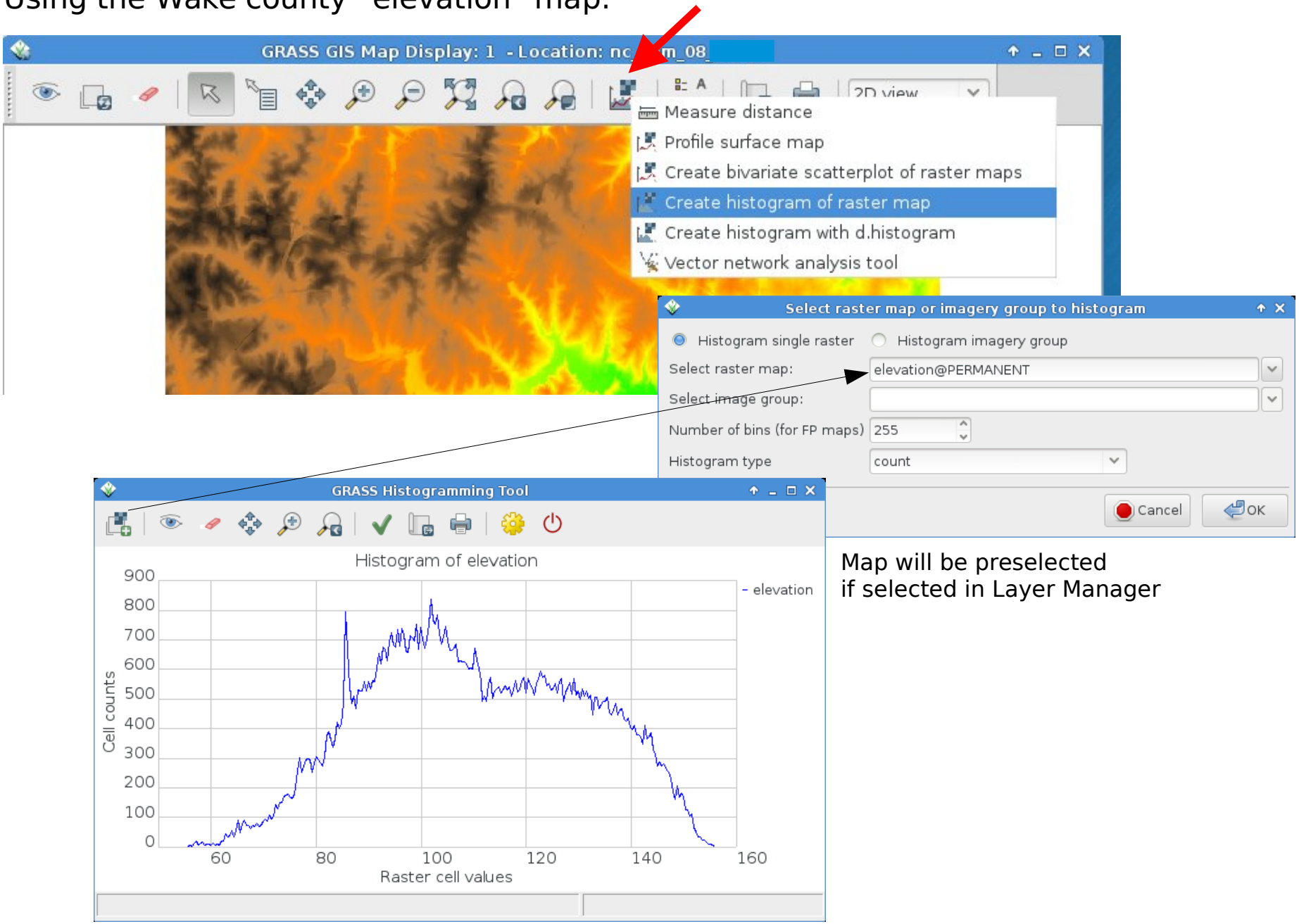
Using the Wake county “elevation” and “roadsmajor” maps:



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

Exercise – Map histogram tool

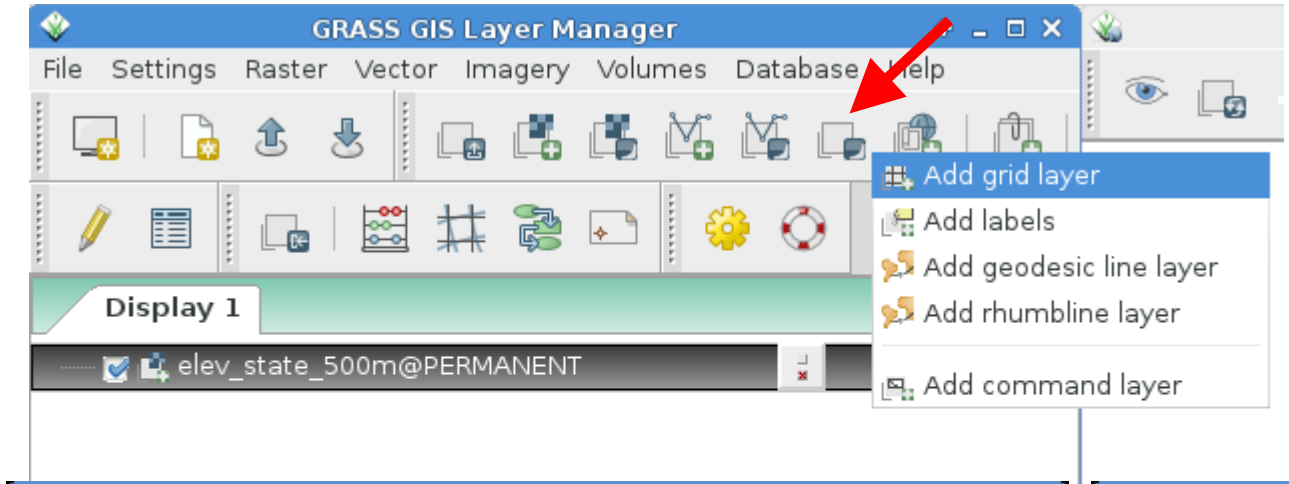
Using the Wake county “elevation” map:

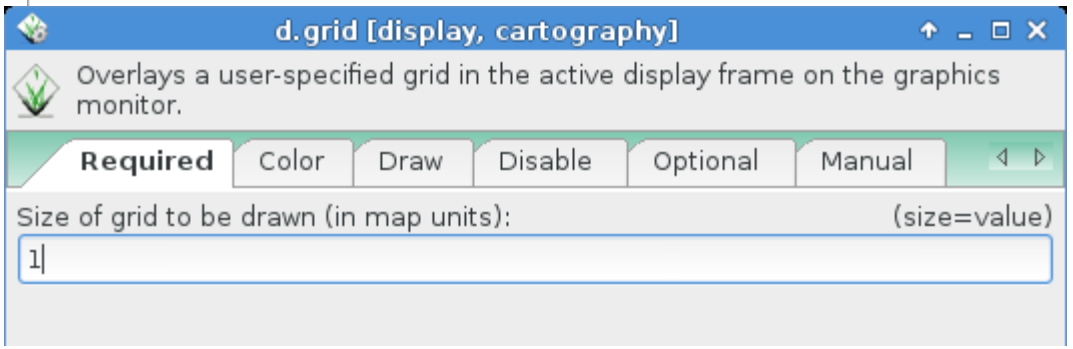


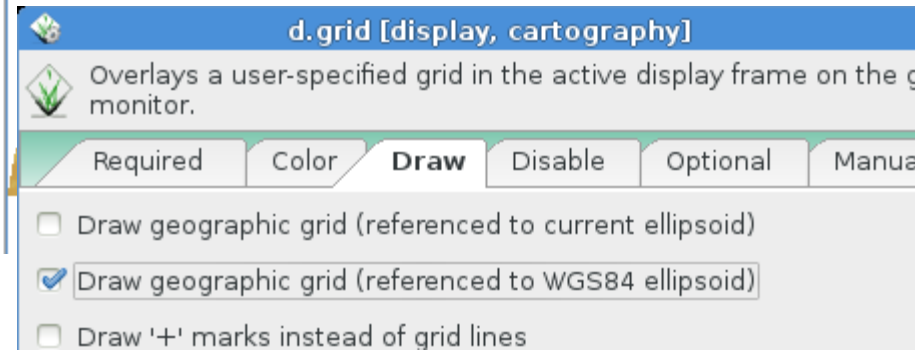
Map will be preselected
if selected in Layer Manager

Exercise – Adding a Lat-Long grid to the map

Using the NC state “elev_state_500m” map:



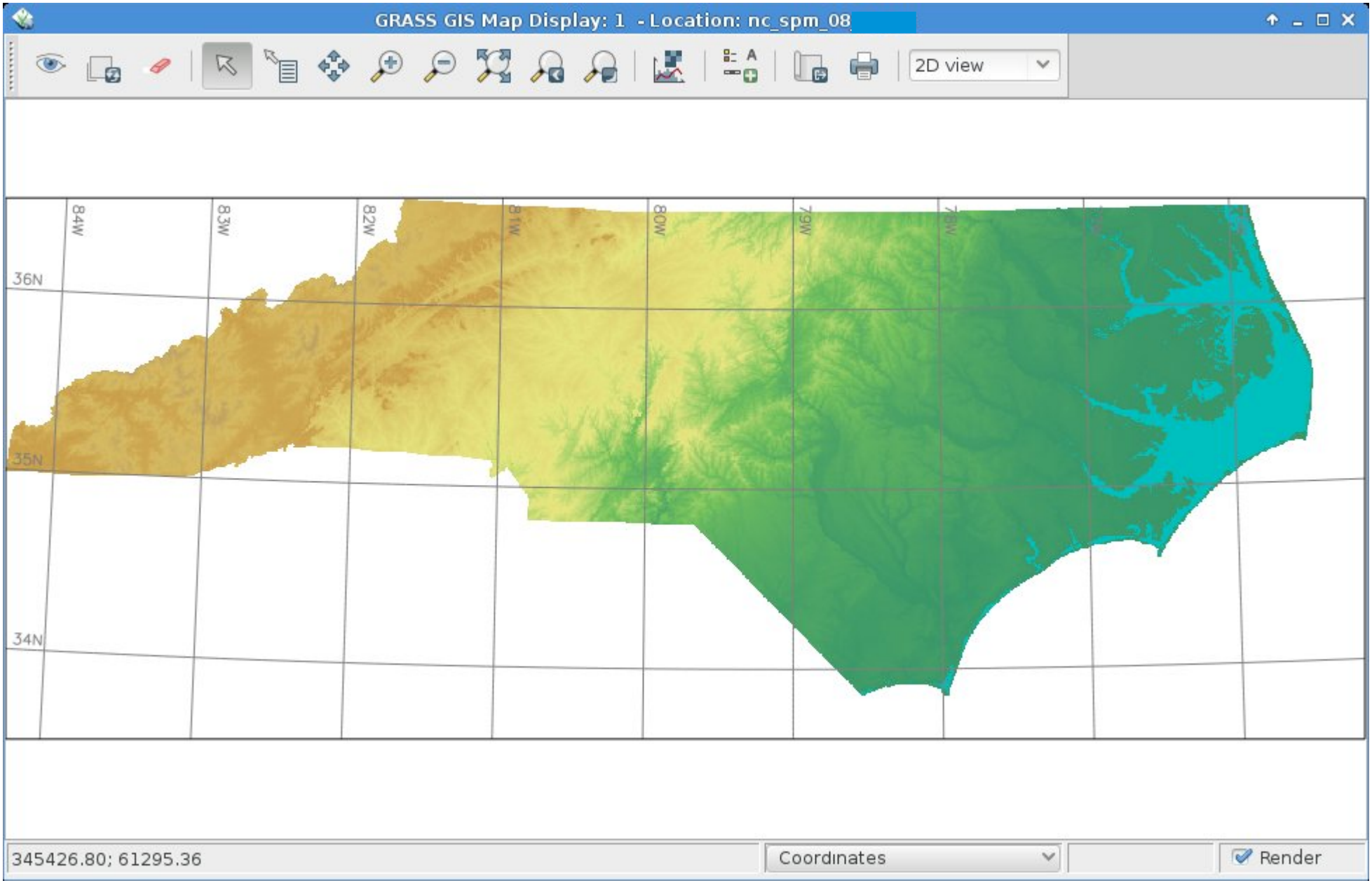




Select a grid size in grid unts (here: LL – 1 deg) → Draw grid as LatLong WGS84 grid

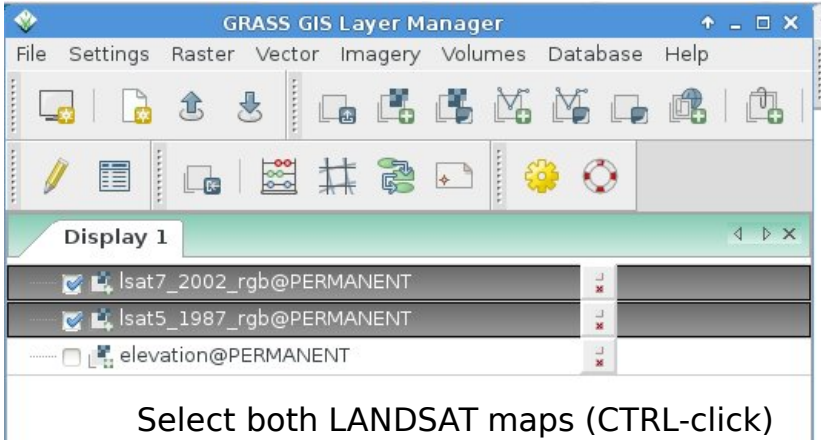
Exercise – Adding a Lat-Long grid to the map

Using the NC state “elev_state_500m” map:



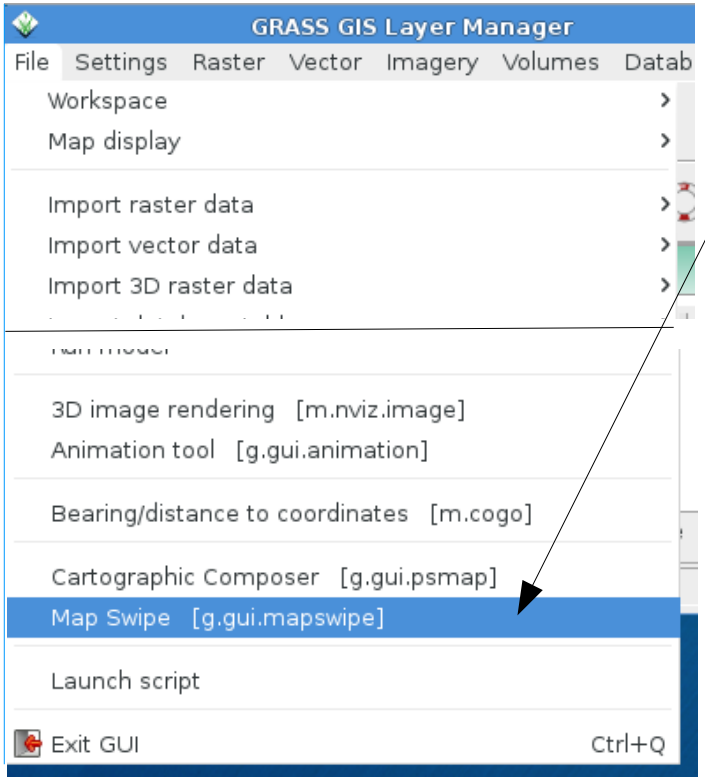
Exercise – Map swiping for multitemporal maps

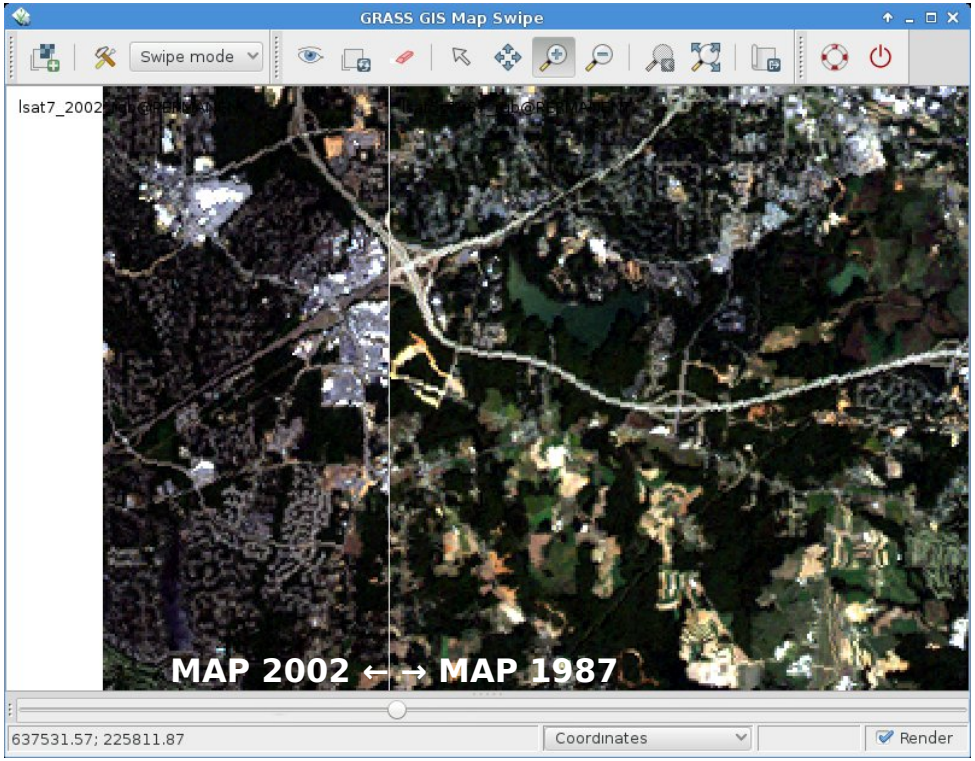
Load the LANDSAT 5 (1987) and 7 (2002) **RGB composites** of Wake county:



Select both LANDSAT maps (CTRL-click)

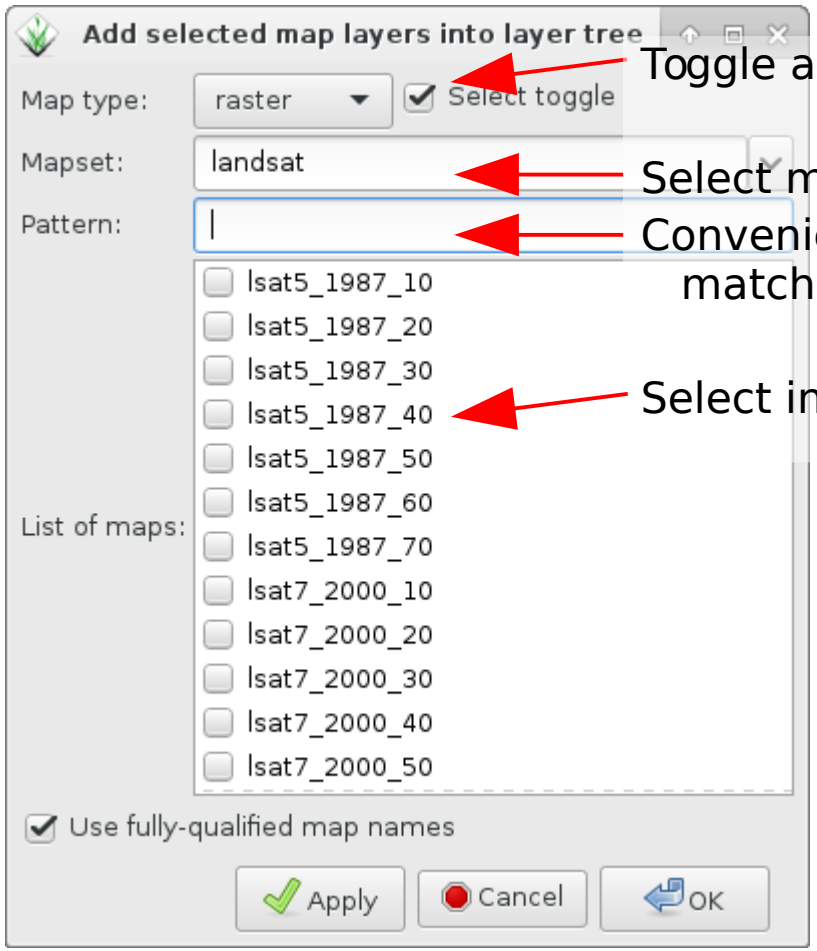
Switch to “map swipe” view





Exercise – Convenient map selector

Load the LANDSAT **channels red and green** using the convenient map selector

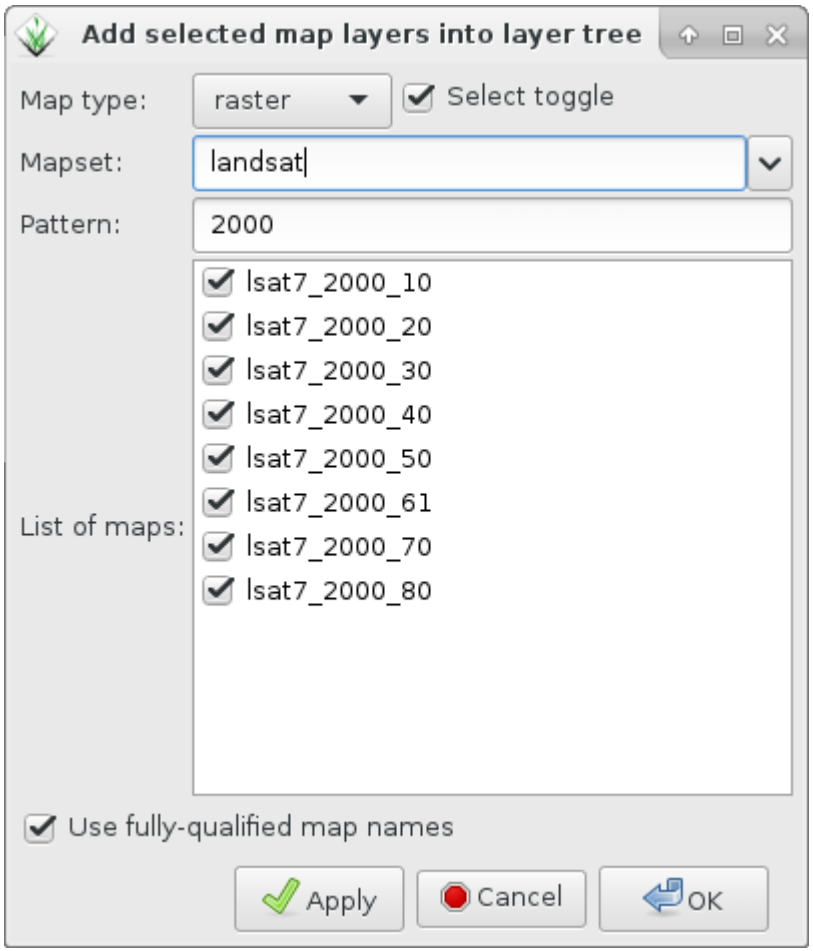


Toggle all/none

Select mapset

Convenient pattern matching

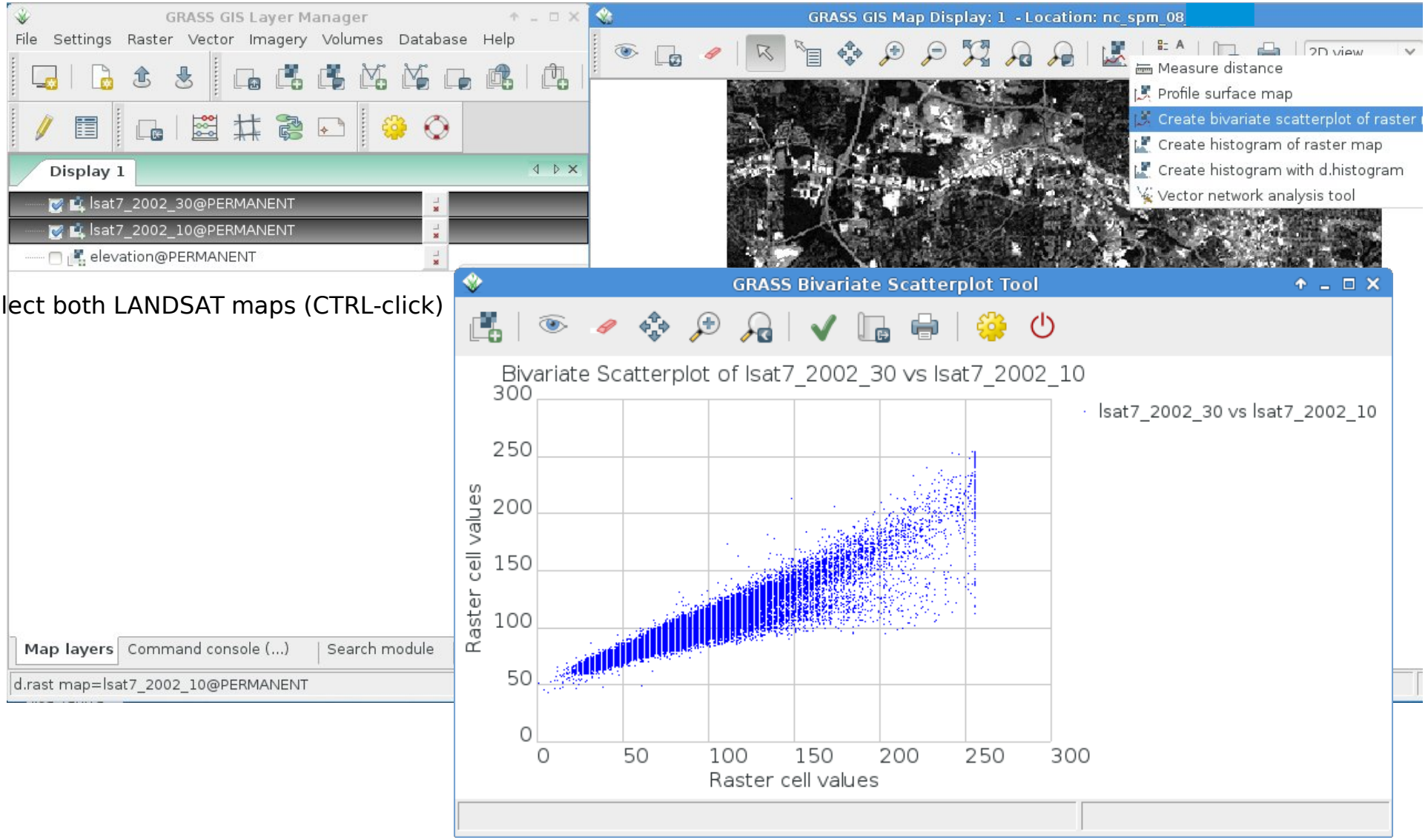
Select individual maps



Exercise – Bivariate Scatterplots

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

Select both LANDSAT maps (CTRL-click)



Overview: GRASS GIS command structure

prefix	***function class***	***type of command***	***example***
g.*	general	general data management	*g.rename: renames map*
d.*	display	graphical output	*d.rast: display raster map* *d.vect: display vector map*
r.*	raster	raster processing	*r.mapcalc: map algebra* *r.univar: univariate statistics*
v.*	vector	vector processing	*v.clean: topological cleaning*
i.*	imagery	imagery processing	*i.pca: Principal Components Analysis on imagery group*
r3.*	voxel	3D raster processing	*r3.stats: Voxel statistics*
db.*	database	database management	*db.select: select value(s) from table*
ps.*	postscript	map creation in PostScript format	*ps.map: PostScript map creation*
t.*	temporal	Space-time cubes	*t.rast.aggregate: Raster time series aggregation*

Ninja trick: on command line, type the desired prefix (e.g. v.) and then <tab><tab> to complete the command name

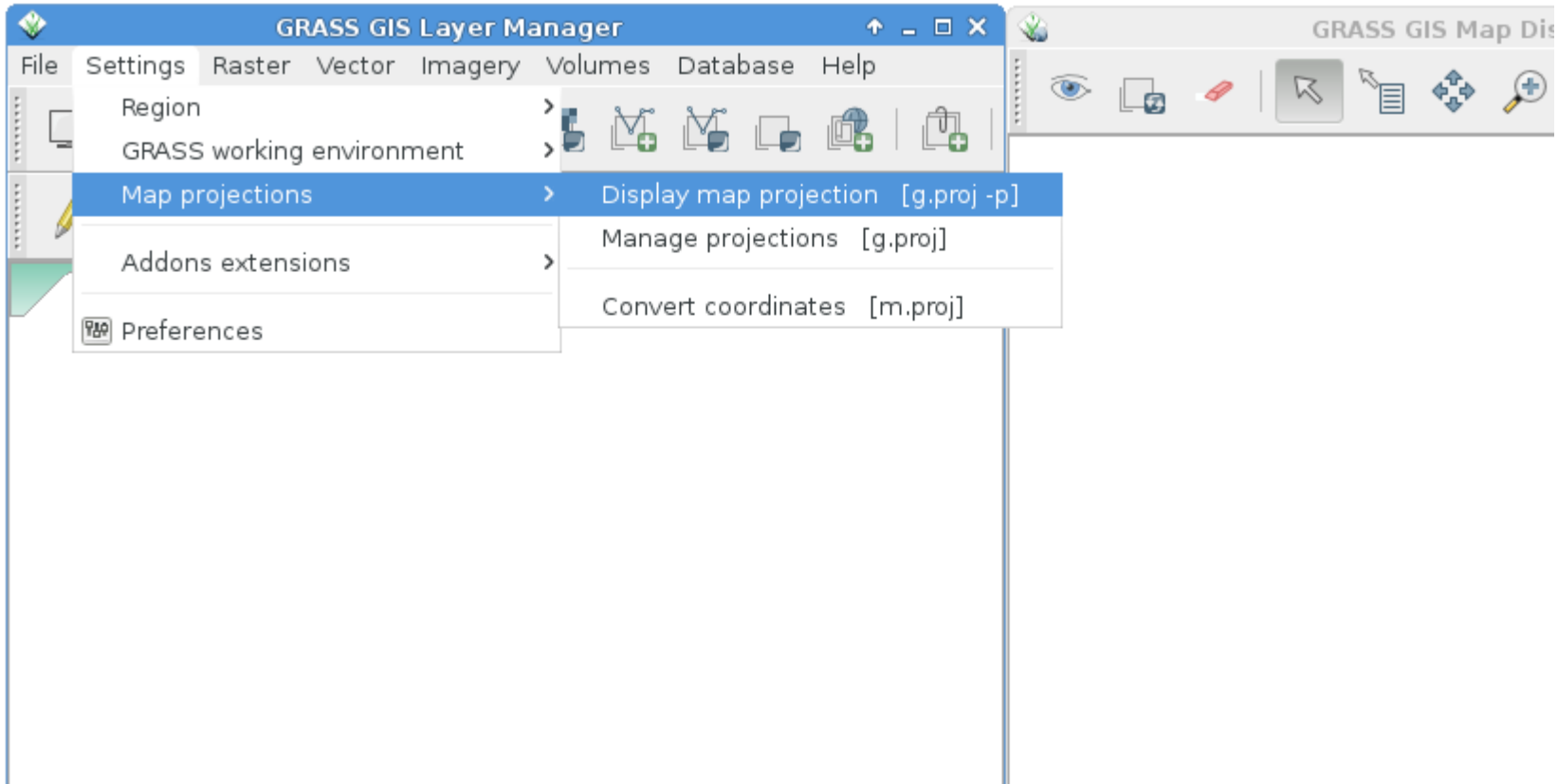
Exercise – GRASS startup and first steps

Know where you are...

get projection information for the North Carolina Location

command: `g.proj`

wxGUI: Settings → Map projections → Manage projections



Exercise – GRASS startup and first steps

Know where you are...

get projection information for the North Carolina sample data set:

```
name         : Lambert Conformal Conic
proj         : lcc
datum        : nad83
a            : 6378137.0
es           : 0.006694380022900787
lat_1        : 36.16666666666666
lat_2        : 34.33333333333334
lat_0        : 33.75
lon_0        : -79
x_0          : 609601.22
y_0          : 0
units        : meters
```

```
Different formats
> g.proj -p
> g.proj -w
> g.proj -t
> g.proj -e
```

GRASS Database concept – Projection

Know where you are...

get projection information for the
North Carolina sample data set:

```
name         : Lambert Conformal Conic
proj         : lcc
datum        : nad83
a            : 6378137.0
es           : 0.006694380022900787
lat_1        : 36.16666666666666
lat_2        : 34.33333333333334
lat_0        : 33.75
lon_0        : -79
x_0          : 609601.22
y_0          : 0
units        : meters
```

Projection name

projection

geodetic datum

} ellipsoid GRS80

} standard parallels

reference latitude

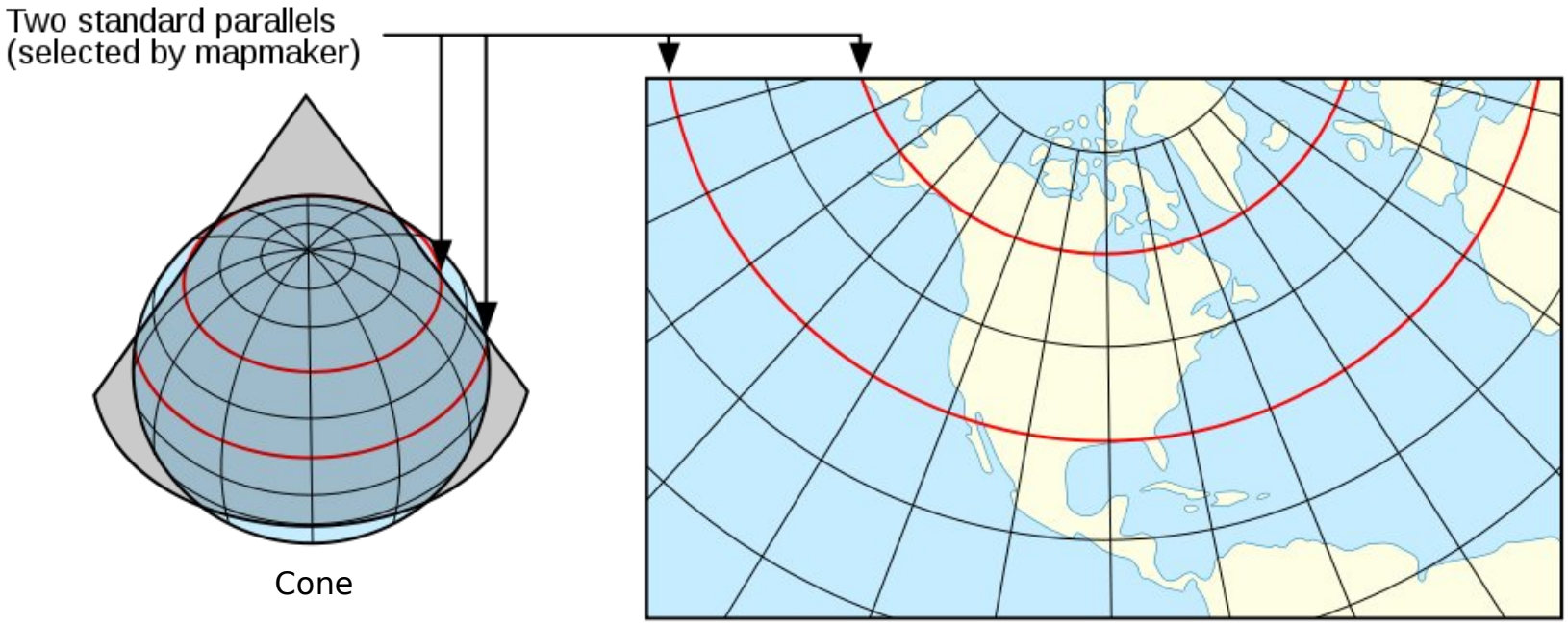
reference longitude

} lon and lat shifts
(false easting/northing)

GRASS Database concept – Projection

Projection of the North Carolina Location:

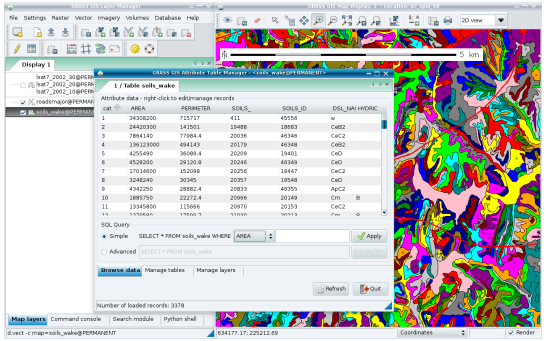
NAD83(HARN) / North Carolina – EPSG code: 3358



+proj=lcc
+lat_1=36.16666666666666 +lat_2=34.33333333333334
+lat_0=33.75 +lon_0=-79
+x_0=609601.22 +y_0=0
+ellps=GRS80 +units=m +no_defs

Graphical user interface versus Command line

- GRASS GIS offers a graphical user interface



- On command line, there is (text) help:

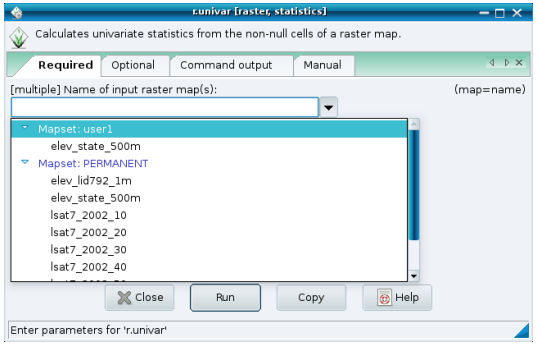
```
> r.univar --help
```

There are flags (e.g. `-g`) and parameters (e.g. `map=`)

```
GRASS 7.0.2svn (nc_spm_08_grass7):~ > r.univar --help

Description:
 Calculates univariate statistics from the non-null cells of a raster map.
 Statistics include number of cells counted, minimum and maximum cell values, range,
 arithmetic mean, population variance, standard deviation, coefficient of variation,
 and sum.

Keywords:
 raster, statistics, univariate statistics, zonal statistics

Usage:
 r.univar [-get] map=name[,name,...] [zones=name] [output=name]
   [percentile=value[,value,...]] [separator=character] [--overwrite]
   [--help] [--verbose] [--quiet] [--ui]

Flags:
  -g   Print the stats in shell script style
  -e   Calculate extended statistics
  -t   Table output format instead of standard output format
  --o  Allow output files to overwrite existing files
  --h  Print usage summary
  --v  Verbose module output
  --q  Quiet module output
  --ui  Force launching GUI dialog

Parameters:
        map   Name of raster map(s)
      zones   Raster map used for zoning, must be of type CELL
     output   Name for output file (if omitted or "-" output to stdout)
 percentile   Percentile to calculate (requires extended statistics flag)
              options: 0-100
              default: 90
  separator   Field separator
               Special characters: pipe, comma, space, tab, newline
              default: pipe
GRASS 7.0.2svn (nc_spm_08_grass7):~ >
```

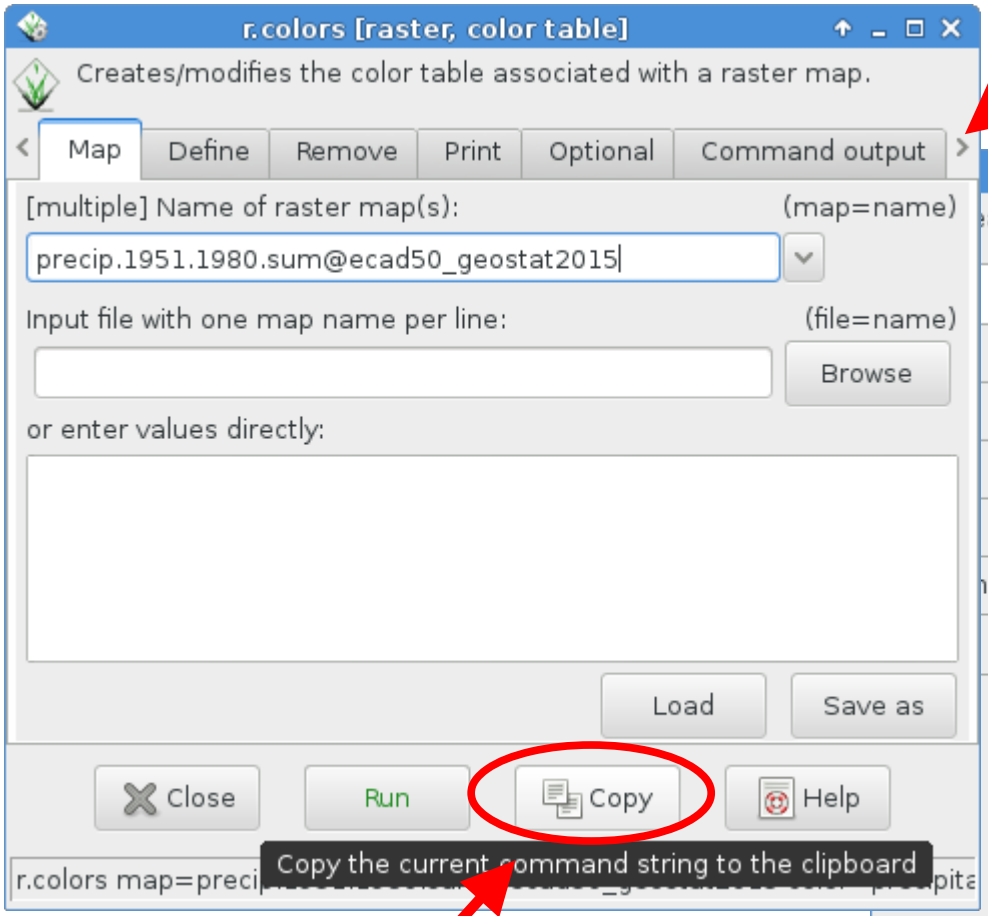
- From command line, you can open the module's GUI (just call the command without parameters)

```
> r.univar
```

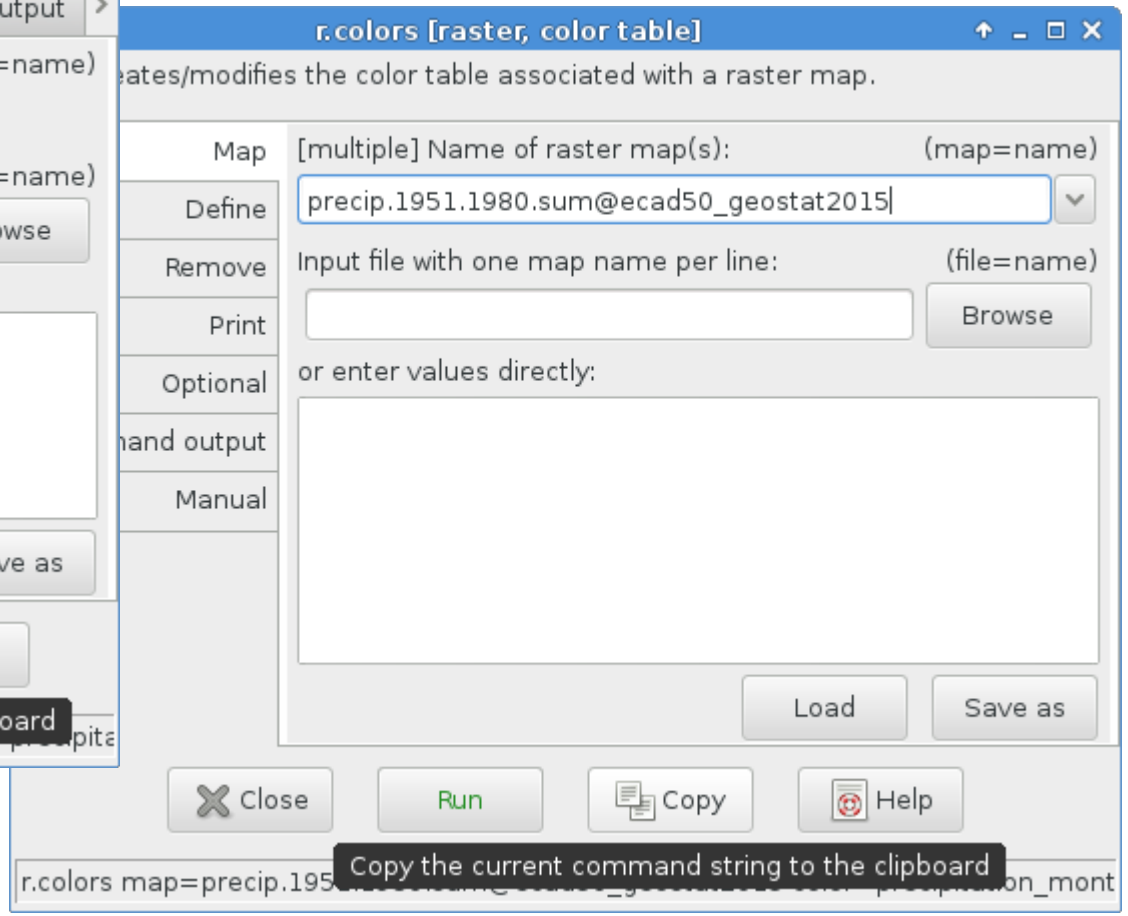


Graphical user interface versus Command line

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”)

Command line at its best: 1/2

Advantages of the command line

- Run "`history`" to see all your previous commands, "forever"
- **History** is stored individually per MAPSET (note that the history of each map is stored within the map's metadata, for this use `r|r3|v.info`)
- **Search** in history with CTRL-R
- **Save** it **to a file**:
  `history > my_protocol.sh`
- `Note for Windows users:`
  `no history command but`
  `"Command console" tab → "Log file"`

```
GRASS 7.0.4 (piemont):~ > v.univar --help

Description:
 Calculates univariate statistics of vector map features.
 Variance and standard deviation is calculated only for points if specified.

Keywords:
 vector, statistics, univariate statistics, attribute table, geometry

Usage:
 v.univar [-gewd] map=name [layer=string] [type=string[,string,...]]
  [column=name] [where=sql_query] [percentile=value] [--help]
  [--verbose] [--quiet] [--ui]

Flags:
 -g   Print the stats in shell script style
 -e   Calculate extended statistics
 -w   Weigh by line length or area size
 -d   Calculate geometric distances instead of attribute statistics
--h   Print usage summary
--v   Verbose module output
--q   Quiet module output
--ui  Force launching GUI dialog

Parameters:
        map   Name of vector map
              Or data source for direct OGR access
      layer   Layer number or name
```

Command line at its best: 2/2

Advantages of the command line

- Polish protocol file, use “# comment” to annotate it
- **Rerun** such a protocol file in a later GRASS GIS session with `sh my_protocol.sh`

Note: a simplified command line is included in the graphical user interface, tab “Command console”.

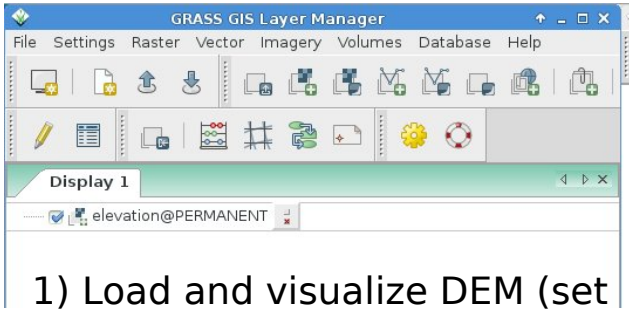
It offers a “Command prompt protocol” button.

- These script will work for decades...

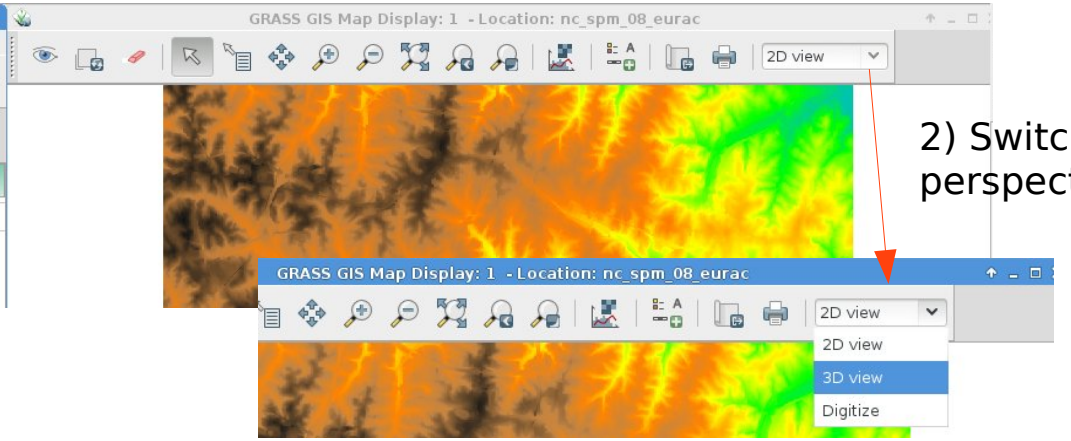
```
#!/bin/sh

# MN, 2009, 2013
# convert ECAD gridded data to final Celsius data
# run in
#  grass70 /grassdata/latlong/ecad90_climate/

VER=9.0
STARTDATE=19500101
ENDNUM=23191
# FYI:
#  date -d '1950-01-01 23191 days' +"%Y-%m-%d"
#  2013-06-30

# tx_0.25deg_reg_v9.0.nc
IN=tx
VAR=tmax

####################
if  [ -z "$GISBASE" ] ; then
    echo "You must be in GRASS GIS to run this program." >&2
    exit 1
fi
export GRASS_OVERWRITE=1
export GRASS_MESSAGE_FORMAT=plain

# import. Band 1 = first day
r.in.gdal -o input=${IN}_0.25deg_reg_v$VER.nc output=${VAR} memory=6000

# initialization, use YYYY.DOY format
MYDATE=`date -d "$STARTDATE 0 days" +"%Y.%j"`
# generates: 1950.001

# i is NC layer number, GDAL bands start with 1
for i in `seq 1 $ENDNUM` ; do

  g.region rast=${VAR}.$i
  # save with YYYY.DOY
  r.mapcalc "${VAR}.$MYDATE = if(${VAR}.$i == -9999, null(), ${VAR}.$i / 100.)"
  g.remove --q rast=${VAR}.$i
  r.colors ${VAR}.$MYDATE color=celsius

  # careful: $i starts with 0!!
  DAYOFFSET=`expr $i - 1`
  MYDATE=`date -d "$STARTDATE $DAYOFFSET days" +"%Y.%j"`
done

exit 0
```

Exercise – Perspective view

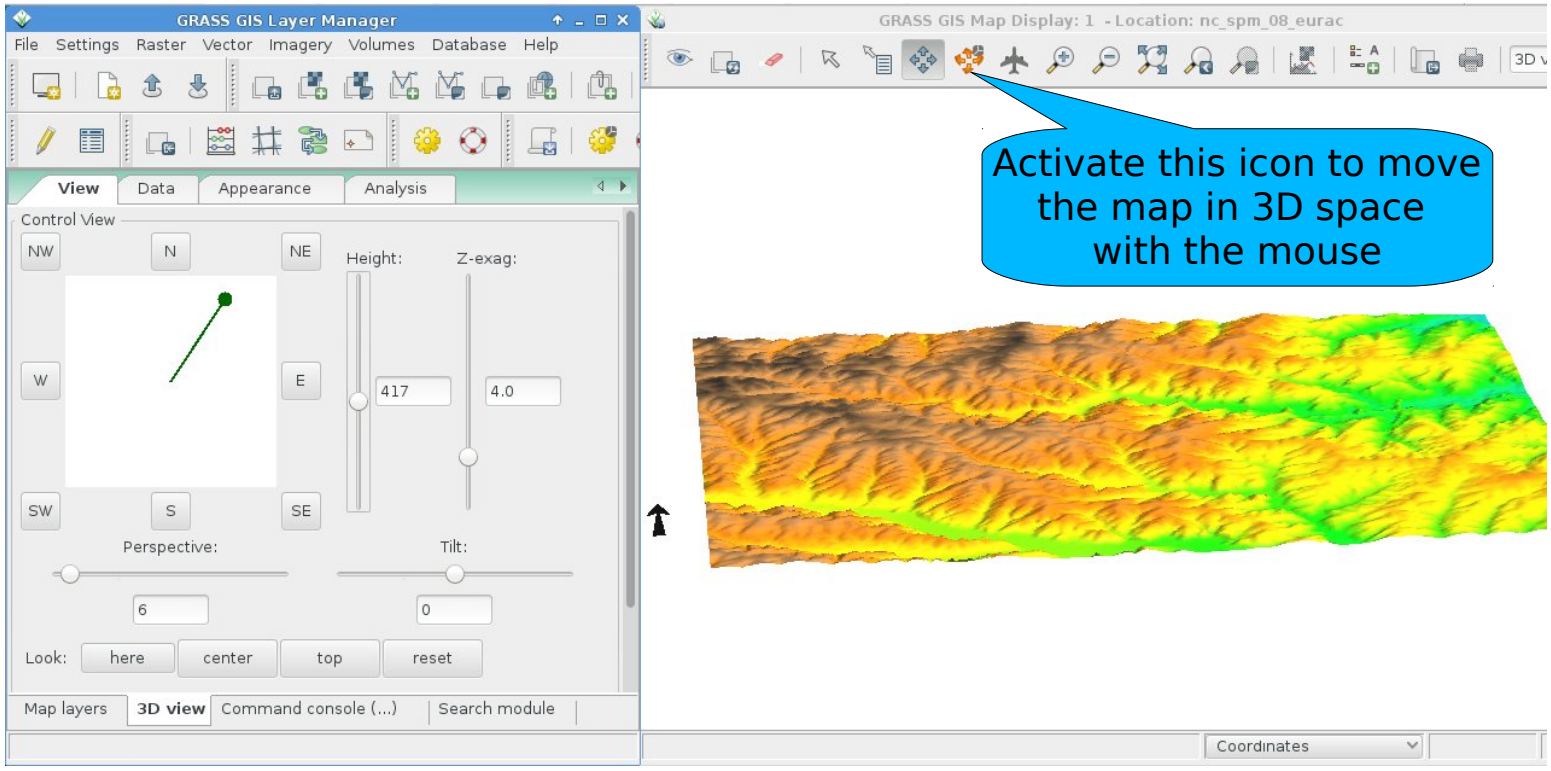
Load the Wake county “elevation” map (this tool requires OpenGL support):



1) Load and visualize DEM (set computational region to it) (load also vector map(s) if you want to use them) into the Layer Manager



2) Switch to perspective view



Eventually switch back to 2D mode...