

Introduction to GRASS GIS

Time Series Analysis of Remote Sensing Data

Micha Silver

Remote Sensing Laboratory

Ben Gurion University, Sde Boker Campus

11/06/2018

email: micha@arava.co.il



Some background

Initial setup

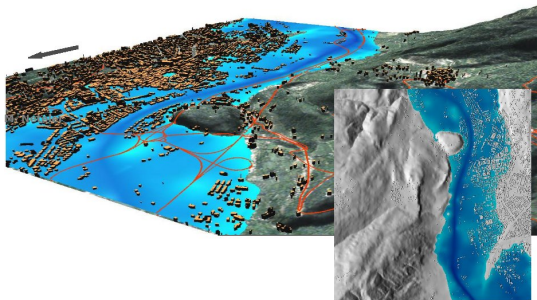
Examples of batch processing

Define and analyze GRASS Spatial-Temporal Dataset

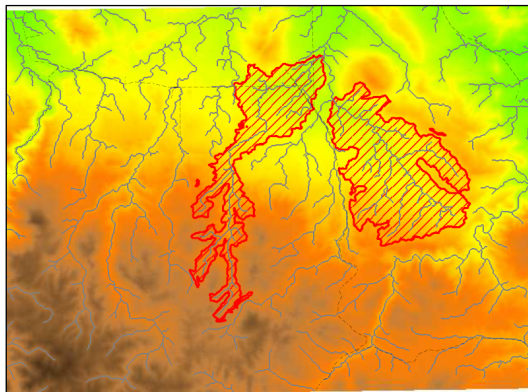


- Among the most experienced GIS in the world
- Actively developed, current stable version: 7.4.0
- License: GPL
- About 350 modules - raster, vector, database, space-time
- Over 130 addons
- Topologically clean vector format
- 3D visualizations
- Fully script-able in bash or python
- *GRASS Development Team, 2018. Geographic Resources Analysis Support System (GRASS) Software, Version 7.4. Open Source Geospatial Foundation. <https://grass.osgeo.org>*

- Flooding simulation



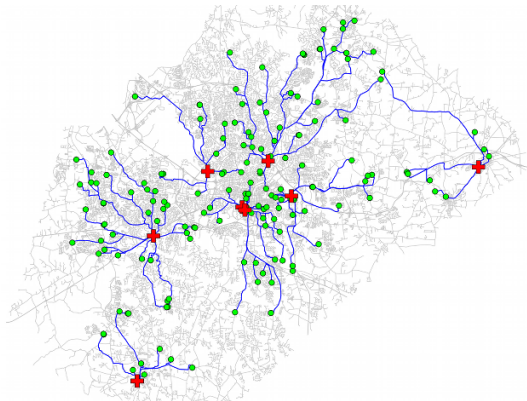
- Flooding simulation
- Flood hazards



---- Highways
— Water courses
 Catchments subject to flood risk

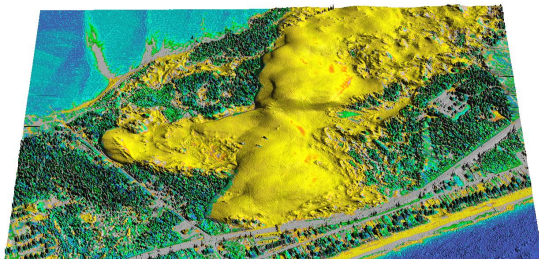
Warning issued Mon Nov 12 21:34:55 NZDT 2007

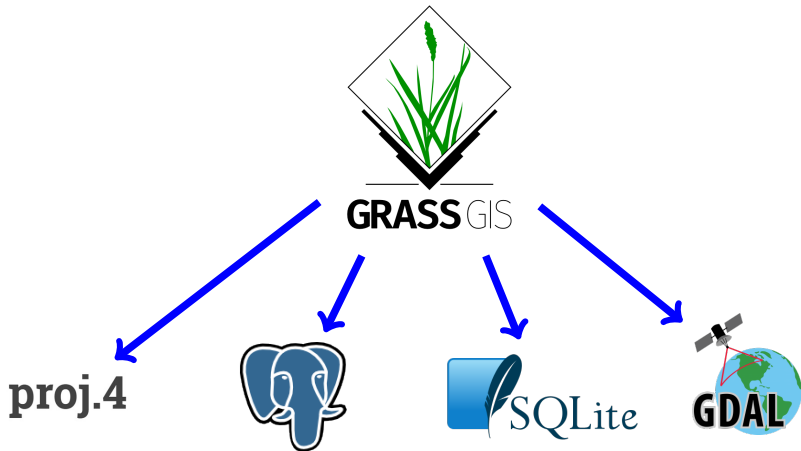
- Flooding simulation
- Flood hazards
- Shortest distance

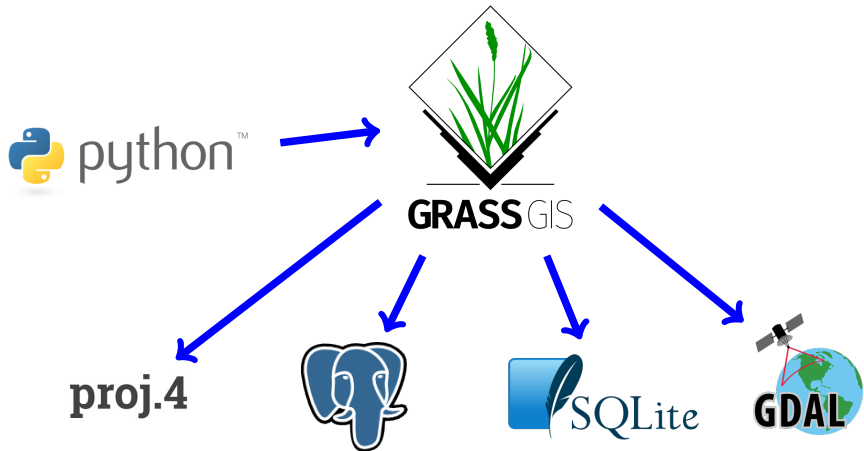


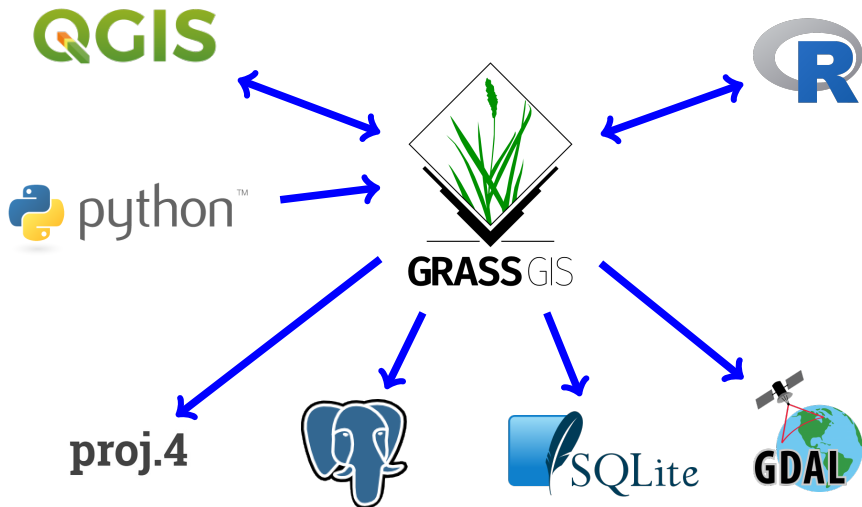
- Flooding simulation
- Flood hazards
- Shortest distance
- Landscape changes

- Flooding simulation
- Flood hazards
- Shortest distance
- Landscape changes
- LIDAR point clouds



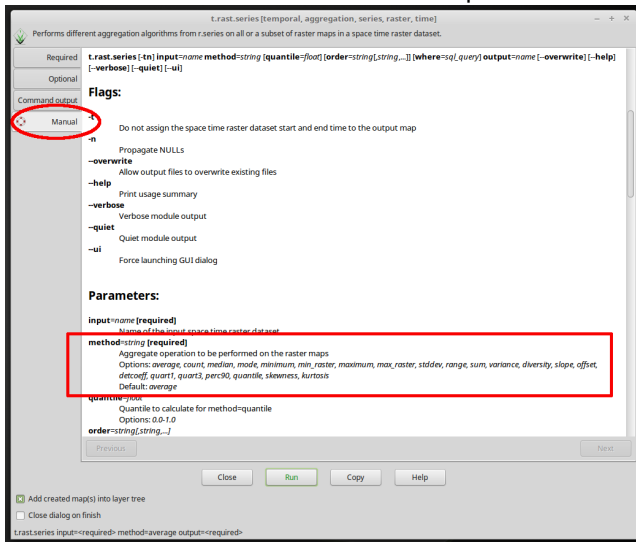






- Tutorial:
<http://www.training.gismentors.eu/grass-gis-workshop-jena-2018/units/01.html>
- Wiki:
[https://grasswiki.osgeo.org/wiki/From_GRASS_GIS_novice_to_power_user_\(workshop_at_FOSS4G_Boston_2017\)](https://grasswiki.osgeo.org/wiki/From_GRASS_GIS_novice_to_power_user_(workshop_at_FOSS4G_Boston_2017))
- The GRASS book:
<https://grassbook.org/>
- User's maillist:
<http://lists.osgeo.org/mailman/listinfo/grass-user>
- NCSU Lab:
<http://ncsu-geoforall-lab.github.io/geospatial-modeling-course/grass/>
- Spatial-Temporal workshop:
<http://ncsu-geoforall-lab.github.io/grass-temporal-workshop/>

Built-in manuals and help



t.rast.series [temporal, aggregation, series, raster, time]

Performs different aggregation algorithms from t.series on all or a subset of raster maps in a space time raster dataset.

Required
Optional
Command output
Manual

t.rast.series [-tn] input=name method=string [quantile=float] [order=string[string,...]] [where=sq[query]] output=name [-overwrite] [-help] [-verbose] [-quiet] [-ui]

Flags:

- n Do not assign the space time raster dataset start and end time to the output map
- n Propagate NULLs
- overwrite Allow output files to overwrite existing files
- help Print usage summary
- verbose Verbose module output
- quiet Quiet module output
- ui Force launching GUI dialog

Parameters:

input=name [required]
Name of the input space time raster dataset

method=string [required]
Aggregate operation to be performed on the raster maps
Options: average, count, median, mode, minimum, min_raster, maximum, max_raster, stddev, range, sum, variance, diversity, slope, offset, detcoeff, quart1, quart3, perc90, quantile, skewness, kurtosis
Default: average

quantile=float
Quantile to calculate for method=quantile
Options: 0.0-1.0

order=string[string,...]

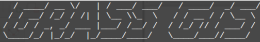
Previous Next

Close Run Copy Help

☒ Add created map(s) into layer tree
☐ Close dialog on finish

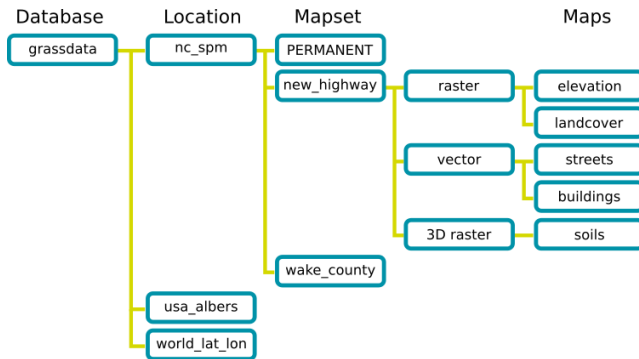
t.rast.series input=<required> method=average output=<required>

Built-in manuals and help

```
micha@RMS -  
File Edit View Search Terminal Help  
  
Welcome to GRASS GIS 7.4.0  
GRASS GIS homepage: http://grass.osgeo.org  
This version running through: Bash Shell (/bin/bash)  
Help is available with the command: g.manual -l  
See the licence terms with: g.version -c  
See citation options with: g.version -x  
If required, restart the GUI with: g.gui wxpython  
When ready to quit enter: exit  
  
Launching <wxpython> GUI in the background, please wait...  
GRASS 7.4.0 (UTM31N):- > : Unable to fetch interface description for command ''.  
  
Details: [Errno 13] Permission denied  
WARNING: Some addons failed when loading. Please consider to update your addons by running 'g.extension.all -f'.  
  
GRASS 7.4.0 (UTM31N):- > t.rast.series --h  
Performs different aggregation algorithms from r.series on all or a subset of raster maps in a space time raster dataset.  
  
Usage:  
t.rast.series [-tn] input=name method=string [quantile=value]  
[order=string[,string,...]] [where=sql_query] output=name [--overwrite]  
[--help] [--verbose] [--quiet] [--ui]  
  
Flags:  
-t Do not assign the space time raster dataset start and end time to the output map  
-n Propagate NULLs  
  
Parameters:  
input Name of the input space time raster dataset  
method Aggregate operation to be performed on the raster maps  
options: average,count,median,mode,minimum,min_raster,maximum,  
max_raster,stddev,range,sum,variance,diversity,slope,  
offset,detcoeff,quart1,quart3,perc90,quantile,skewness,  
kurtosis  
default: average  
quantile quantile to calculate for method quantile  
options: 0.0-1.0  
order Sort the maps by category  
options: id, name, creator, mapset, creation_time,  
modification_time, start_time, end_time, north, south,  
west, east, min, max  
default: start_time  
where WHERE conditions of SQL statement without 'where' keyword used in the temporal GIS framework  
output Name for output raster map  
GRASS 7.4.0 (UTM31N):- >  
GRASS 7.4.0 (UTM31N):- >
```

- **GRASS Database**
 - ▶ **Location** defined by coordinate reference system
 - ▶ **Mapset** contains data for a project
- Always check and set **computational region**

- **GRASS Database**
 - ▶ **Location** defined by coordinate reference system
 - ▶ **Mapset** contains data for a project
- Always check and set **computational region**



- **GRASS Database**
 - ▶ **Location** defined by coordinate reference system
 - ▶ **Mapset** contains data for a project
- Always check and set **computational region**

Two take-home messages:

- **Location** = Coordinate System
- Set GRASS **region**



Define new GRASS Location

Define GRASS Database and Location Name

GIS Data Directory:

Project Location:

Location Title:

☐ Set default region extent and resolution

☐ Create user mapset

Define new GRASS Location

Choose EPSG Code

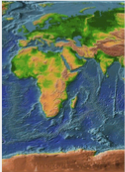
Path to the EPSG-codes file:

EPSG code:

Code	Description	Parameters
2000	Anguilla 1957 / British West Indies Grid	+proj=tmerc +lat_0=0 +lon_0=
2001	Antigua 1943 / British West Indies Grid	+proj=tmerc +lat_0=0 +lon_0=
2002	Dominica 1945 / British West Indies Grid	+proj=tmerc +lat_0=0 +lon_0=
2003	Grenada 1953 / British West Indies Grid	+proj=tmerc +lat_0=0 +lon_0=
2004	Montserrat 1958 / British West Indies Grid	+proj=tmerc +lat_0=0 +lon_0=
2005	St. Kitts 1955 / British West Indies Grid	+proj=tmerc +lat_0=0 +lon_0=
2006	St. Lucia 1955 / British West Indies Grid	+proj=tmerc +lat_0=0 +lon_0=
2007	St. Vincent 45 / British West Indies Grid	+proj=tmerc +lat_0=0 +lon_0=
2008	NAD27(CGQ77) / SCoPQ zone 2 (deprecated)	+proj=tmerc +lat_0=0 +lon_0=
2009	NAD27(CGQ77) / SCoPQ zone 3 (deprecated)	+proj=tmerc +lat_0=0 +lon_0=

Find EPSG codes here: <http://epsg.io/>

Define new GRASS Location



Summary

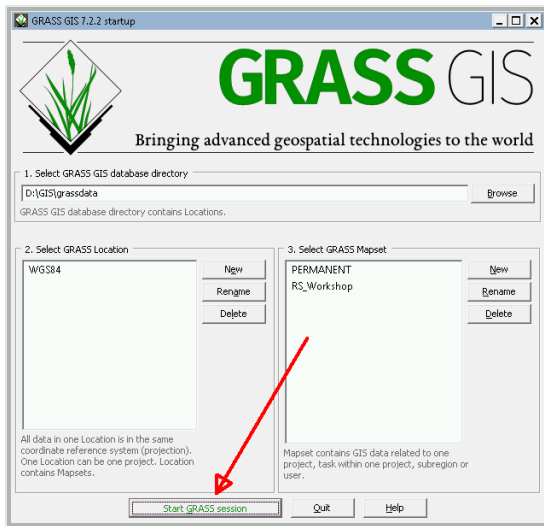
GRASS Database: D:\GIS\grassdata
Location Name: WGS84
Location Title:

Projection: EPSG code 4326 (WGS 84)

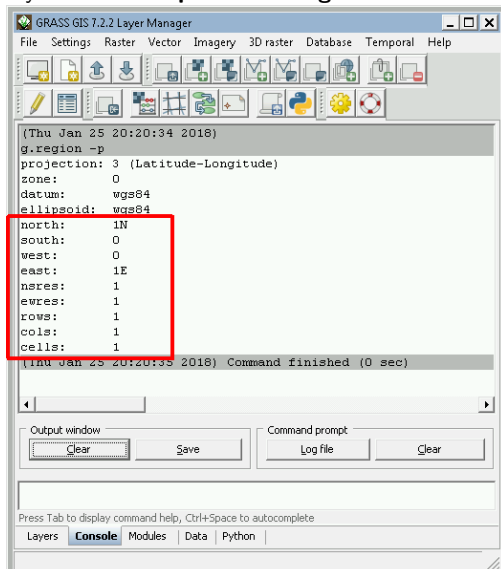
PROJ.4 definition: +proj=longlat
(non-definitive)
+no_defs
+a=6378137
+rf=298.257223563
+towgs84=0.000,0.000,0.000

Help < Back Finish Cancel



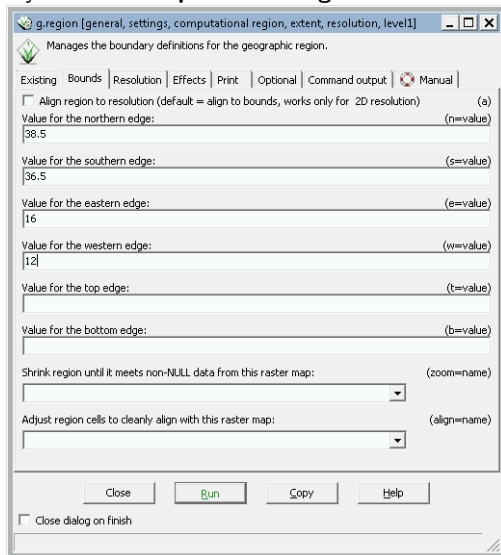


Always set the **computational region**



```
(Thu Jan 25 20:20:34 2018)
g.region -p
projection: 3 (Latitude-Longitude)
zone:      0
datum:     wgs84
ellipsoid: wgs84
north:     1N
south:     0
west:      0
east:      1E
nres:      1
ewres:     1
rows:      1
cols:      1
cells:     1
(Thu Jan 25 20:20:35 2018) Command finished (0 sec)
```

Always set the **computational region**



g.region [general, settings, computational region, extent, resolution, level1]

Manages the boundary definitions for the geographic region.

Existing | **Bounds** | Resolution | Effects | Print | Optional | Command output | Manual

☐ Align region to resolution (default = align to bounds, works only for 2D resolution) (a)

Value for the northern edge: (n=value)
38.5

Value for the southern edge: (s=value)
36.5

Value for the eastern edge: (e=value)
16

Value for the western edge: (w=value)
12

Value for the top edge: (t=value)

Value for the bottom edge: (b=value)

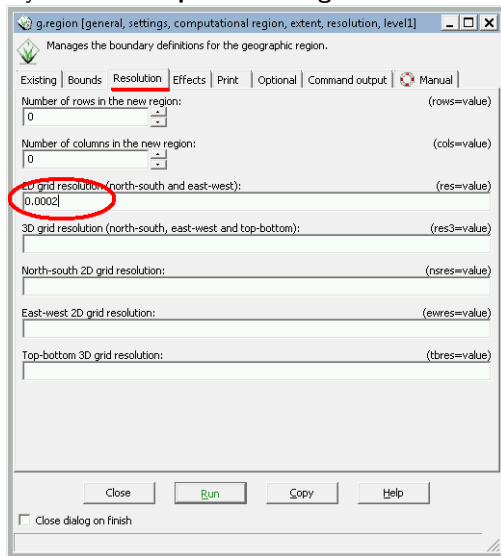
Shrink region until it meets non-NULL data from this raster map: (zoom=name)
[Dropdown menu]

Adjust region cells to cleanly align with this raster map: (align=name)
[Dropdown menu]

Close Run Copy Help

☐ Close dialog on finish

Always set the **computational region**



g.region [general, settings, computational region, extent, resolution, level1]

Manages the boundary definitions for the geographic region.

Existing | Bounds | Resolution | Effects | Print | Optional | Command output | Manual

Number of rows in the new region: (rows=value)
0

Number of columns in the new region: (cols=value)
0

2D grid resolution (north-south and east-west): (res=value)
0.0002

3D grid resolution (north-south, east-west and top-bottom): (res3=value)

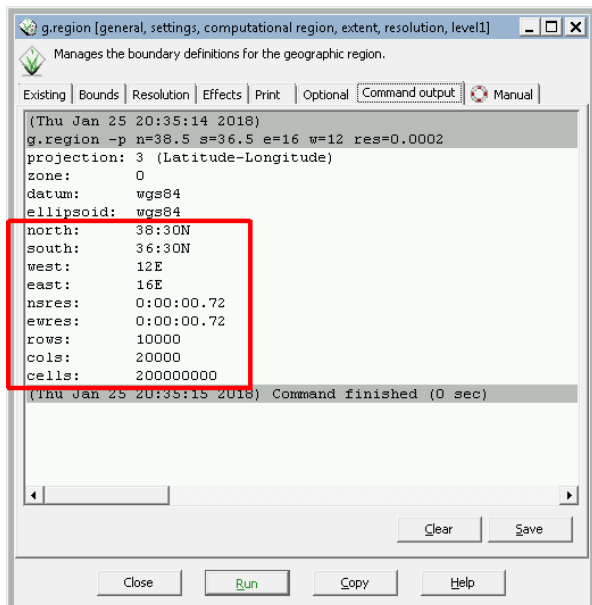
North-south 2D grid resolution: (nsres=value)

East-west 2D grid resolution: (ewres=value)

Top-bottom 3D grid resolution: (tbres=value)

Close Run Copy Help

☐ Close dialog on finish



g.region [general, settings, computational region, extent, resolution, level1]

Manages the boundary definitions for the geographic region.

Existing | Bounds | Resolution | Effects | Print | Optional | Command output | Manual

```
(Thu Jan 25 20:35:14 2018)
g.region -p n=38.5 s=36.5 e=16 w=12 res=0.0002
projection: 3 (Latitude-Longitude)
zone:      0
datum:     wgs84
ellipsoid: wgs84
north:     38:30N
south:     36:30N
west:      12E
east:      16E
nsres:     0:00:00.72
ewres:     0:00:00.72
rows:      10000
cols:      20000
cells:     200000000
(Thu Jan 25 20:35:15 2018) Command finished (0 sec)
```

Clear Save

Close Run Copy Help

Batch Processing in GRASS

Time series analysis **requires** processing large batches of input

Language choices:

- Windows command prompt `cmd.exe`
- Linux bash shell script (also on MacOS)



Language choices:

- Windows command prompt `cmd.exe`
- Linux bash shell script (also on MacOS)

- python (any OS)



Language choices:

- Windows command prompt `cmd.exe`
- Linux bash shell script (also on MacOS)



- python (any OS)
 - ▶ Python scripting in GRASS session:
`import grass.script as gscript`

GRASS command names:

- general commands: **g.***
- raster commands: **r.***
- vector commands **v.***
- temporal commands **t.***

Start GRASS pointing to the LOCATION/MAPSET that we created:

```
# The location/mapset was already created
GRASS_DIR="{BASE_DIR}/grassdata/UTM31N/camargue"
cd ${TARGET_DIR}
# Start grass
grass $GRASS_DIR

# All following commands are entered at GRASS terminal
# Import NDVI rasters
for f in 20*.tif; do
    r='basename $f .tif '
    r.in.gdal input=$f output="ndvi-{r}" band=1
done
```

GRASS - Set Computational Region



```
# Get a list of all GRASS rasters
RAST_LIST='g.list rast separator=comma pattern=ndvi_*'
# Set computational region
g.region -p rast=${RAST_LIST}
# Apply color ramp to the whole list
r.colors map=${RAST_LIST} color=ndvi
```

```
micha@RMS -
File Edit View Search Terminal Help

GRASS GIS

Welcome to GRASS GIS 7.4.0
GRASS GIS homepage: http://grass.osgeo.org
This version running through: Bash shell (/bin/bash)
Help is available with the command: g.manual -i
See the licence terms with: g.version -c
See citation options with: g.version -x
If required, restart the GUI with: g.gui wxpython
When ready to quit enter: exit

Launching <wxpython> GUI in the background, please wait...
GRASS 7.4.0 (UTM31N):- > : Unable to fetch interface description for command ''.

Details: [Errno 13] Permission denied
WARNING: Some addons failed when loading. Please consider to update your addons by running 'g.extension.all -f'.

GRASS 7.4.0 (UTM31N):- > # Get a list of all GRASS rasters
GRASS 7.4.0 (UTM31N):- > RAST_LIST=g.list rast separator=comma pattern=ndvi_*
GRASS 7.4.0 (UTM31N):- > # Set computational region
GRASS 7.4.0 (UTM31N):- > g.region -p rast=${RAST_LIST}
projection: UTM
zone: 31
datum: WGS84
ellipsoid: WGS84
north: 4842000
south: 4800000
west: 600000
east: 650000
nres: 10
sres: 10
rows: 4200
cols: 4400
cells: 18480000
GRASS 7.4.0 (UTM31N):- > # Apply color ramp to the whole list
GRASS 7.4.0 (UTM31N):- > r.colors map=${RAST_LIST} color=ndvi
Color table for raster map <ndvi_20170403> set to 'ndvi'
Color table for raster map <ndvi_20170410> set to 'ndvi'
```

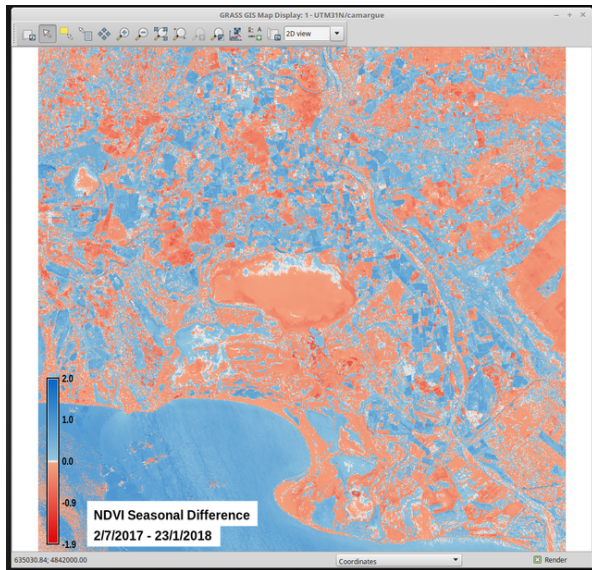
Spatial-Temporal Raster Datasets

Use map calculator to get diff between NDVI from two dates

```
# Compare Winter/Summer difference  
r.mapcalc "season_diff = 1.0*(ndvi_20170702 - ndvi_20180123)"  
r.colors season_diff color=${CODE_DIR}/diff_ramp.txt
```

Note: multiply by **1.0** to force REAL number, otherwise an INTEGER raster is created

GRASS - Seasonal Difference in NDVI



Create a new, empty Spatial-Temporal Raster Dataset (**strds**)

```
# Create and populate space-time raster dataset  
t.create output=camargue_ndvi type=strds  
temporaltype=absolute title="camargue NDVI" \  
description="camargue NDVI time series"
```

and populate with all NDVI rasters, using bash **date** manipulation to set start and end times

```
for r in `g.list rast pattern=ndvi_*`; do  
    y=${r:5:4}  
    m=${r:9:2}  
    d=${r:11:2}  
    s_dt=$y-$m-$d  
    e_dt='date -d "$s_dt + 1 days" +"%Y-%m-%d" '  
    t.register input=camargue_ndvi maps=$r \  
    start=$s_dt end=$e_dt --overwrite  
done
```

Set color ramp, and check info

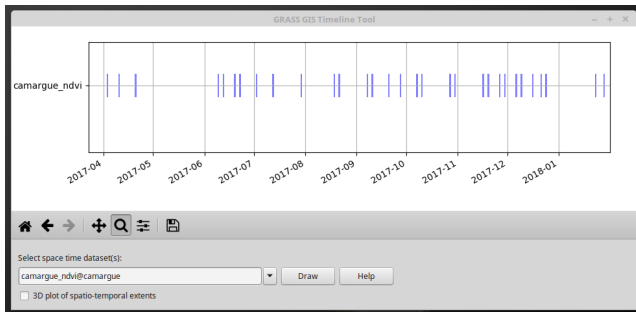
```
t.rast.colors camargue_ndvi color=ndvi  
t.info camargue_ndvi  
t.rast.list camargue_ndvi colu=id,start_time,end_time
```

Set color ramp, and check info

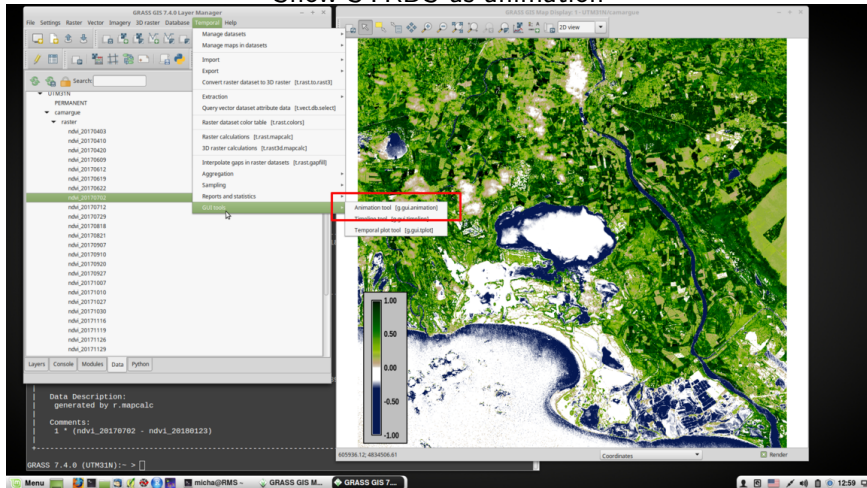
```
t.rast.colors camargue_ndvi color=ndvi  
t.info camargue_ndvi  
t.rast.list camargue_ndvi colu=id,start_time,end_time
```

View the Timeline

```
g.gui.timeline strds=camargue_ndvi  
g.gui.animation strds=camargue_ndvi
```

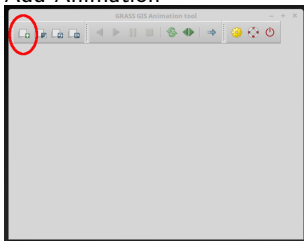


Show STRDS as animation

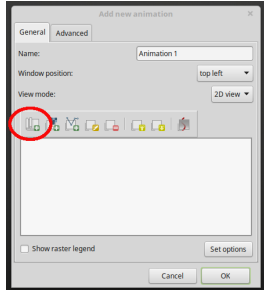
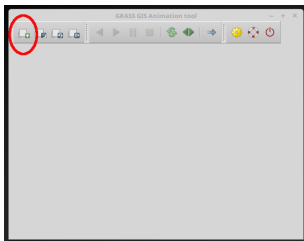


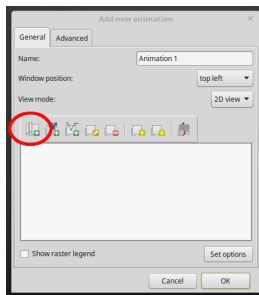
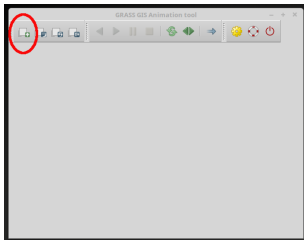
The screenshot displays the GRASS GIS 7.4.0 desktop environment. On the left, the 'Layer Manager' panel lists various datasets, including 'ndvi_20170602', which is highlighted. The 'Tools' menu is open, showing options like 'Import', 'Export', 'Extraction', 'Raster calculations', '3D raster calculations', 'Interpolate gaps in raster datasets', 'Aggregation', 'Sampling', 'Reports and statistics', and 'Animation tool'. The 'Animation tool' is highlighted with a red box. The main window shows a 2D view of a map with a color scale ranging from -1.00 to 1.00. The map displays a landscape with green vegetation, blue water bodies, and white snow/ice patches. The status bar at the bottom indicates the coordinates '605936.12, 4834506.01' and the projection 'UTM31N'.

Add Animation

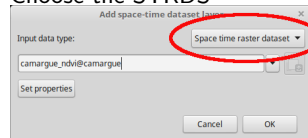


Add STRDS to animation

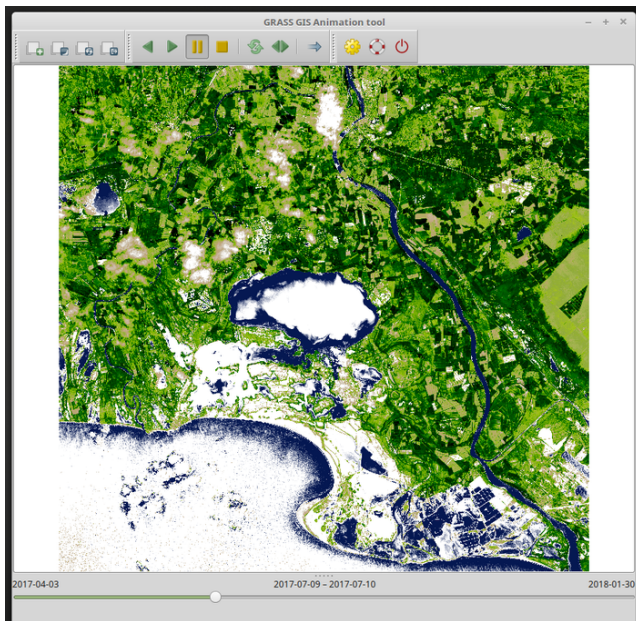




Choose the STRDS



GRASS - Animation Result



Module: t.rast.series

t.rast.series [temporal, aggregation, series, raster, time]

Performs different aggregation algorithms from r.series on all or a subset of raster maps in a space time raster dataset.

Required	Name of the input space time raster dataset: *	(input=name)
Optional	camargue_ndvi@camargue	
Command output	Aggregate operation to be performed on the raster maps: *	(method=string)
Manual	stdddev	
	Name for output raster map: *	(output=name)
	camargue_ndvi_stdev	

Close Run Copy Help

☒ Add created map(s) into layer tree
☐ Close dialog on finish

t.rast.series input=camargue_ndvi@camargue method=stdddev output=camargue_ndvi_stdev

t.rast.series example

```
t.rast.series input=camargue_ndvi@camargue method=slope \  
where="start_time > '2017-09-01 00:00' and \  
start_time < '2017-12-31'" output=camargue_ndvi_slope_fall
```



Linear interpolation between dates Module: t.rast.gapfill

t.rast.gapfill [temporal, interpolation, raster, time]

Replaces gaps in a space time raster dataset with interpolated raster maps.

Required	Name of the input space time raster dataset: *	(input=name)
Optional	camargue_ndvi@camargue	
Command output	Baseline of the new generated output maps: *	(basename=string)
Manual	ndvi_interp	

Close Run Copy Help

t.rast.gapfill --verbose input=camargue_ndvi@camargue where=start_time>'2017-09-01' and start_time<'2017-12-31' basename=

Linear interpolation between dates Module: t.rast.gapfill

t.rast.gapfill [temporal, interpolation, raster, time]

Replaces gaps in a space time raster dataset with interpolated raster maps.

Required ☐ Assign the space time raster dataset start and end time to the output map

Optional ☒ Verbose module output
☐ Quiet module output

Command output WHERE conditions of SQL statement without 'where' keyword used in the temporal GIS framework:

Manual start_time>'2017-09-01' and start_time<'2017-12-31'

Suffix to add at basename: set 'gran' for granularity, 'time' for the full time format, 'num' for numerical suffix a specific number of digits (default %05):

gran

Number of interpolation processes to run in parallel:

4

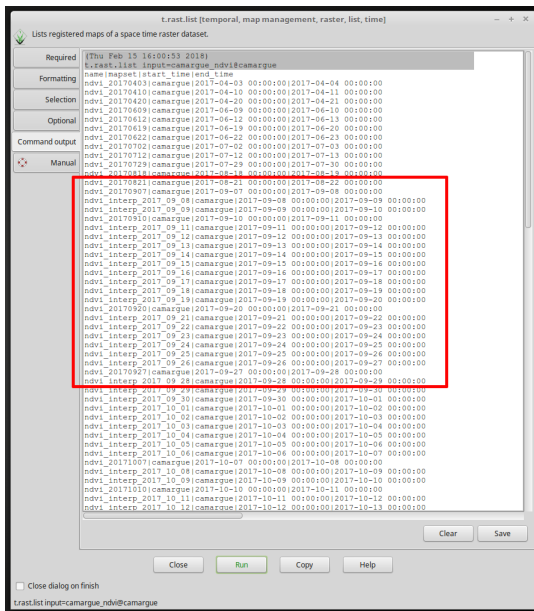
Close Run Copy Help

t.rast.gapfill --verbose input=camargue_ndvi@camargue where=start_time>'2017-09-01' and start_time<'2017-12-31' basename:

GRASS Interpolate Missing Time slots

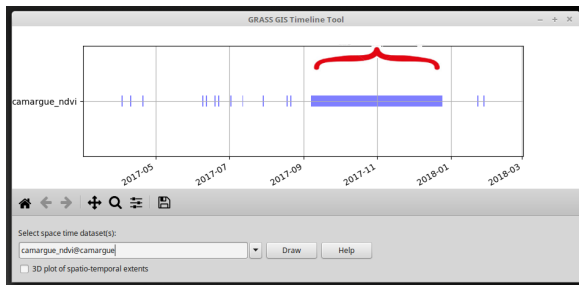


List of maps in STRDS,
including interpolated



GRASS Interpolate Missing Time slots

List of maps in STRDS,
including interpolated



Thanks

