Procesamiento de series de tiempo en GRASS GIS

Aplicaciones en Ecología y Ambiente

Dra. Verónica Andreo
CONICET - INMeT

Río Cuarto, 2018

Spatio-temporal data processing & visualization in GRASS GIS

GRASS GIS is **the first Open Source GIS** that incorporated capabilities to **manage, analyze, process and visualize spatio-temporal data**, as well as the temporal relationships among time series.

The TGRASS framework

The TGRASS framework

- TGRASS is the temporal enabled GRASS GIS designed to easily handle time series data

The TGRASS framework

- TGRASS is the temporal enabled GRASS GIS designed to easily handle time series data
- TGRASS is fully based on metadata and does not duplicate any dataset

The TGRASS framework

- TGRASS is the temporal enabled GRASS GIS designed to easily handle time series data
- TGRASS is fully based on metadata and does not duplicate any dataset
- Snapshot approach, i.e., adds time stamps to maps

The TGRASS framework

- TGRASS is the temporal enabled GRASS GIS designed to easily handle time series data
- TGRASS is fully based on metadata and does not duplicate any dataset
- Snapshot approach, i.e., adds time stamps to maps
- A collection of time stamped maps (snapshots) of the same variable are called space-time datasets or STDS

The TGRASS framework

- TGRASS is the temporal enabled GRASS GIS designed to easily handle time series data
- TGRASS is fully based on metadata and does not duplicate any dataset
- Snapshot approach, i.e., adds time stamps to maps
- A collection of time stamped maps (snapshots) of the same variable are called space-time datasets or STDS
- Maps in a STDS can have different spatial and temporal extents

Space-time datasets

- Space time raster datasets (**STRDS**)
- Space time 3D raster datasets (**STR3DS**)
- Space time vector datasets (**STVDS**)

Other TGRASS notions

Other TGRASS notions

- Time can be defined as intervals (start and end time) or instances (only start time)

Other TGRASS notions

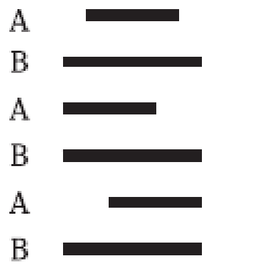
- Time can be defined as intervals (start and end time) or instances (only start time)
- Time can be absolute (e.g., 2017-04-06 22:39:49) or relative (e.g., 4 years, 90 days)

Other TGRASS notions

- Time can be defined as intervals (start and end time) or instances (only start time)
- Time can be absolute (e.g., 2017-04-06 22:39:49) or relative (e.g., 4 years, 90 days)
- Granularity is the greatest common divisor of the temporal extents (and possible gaps) of all maps in the space-time cube

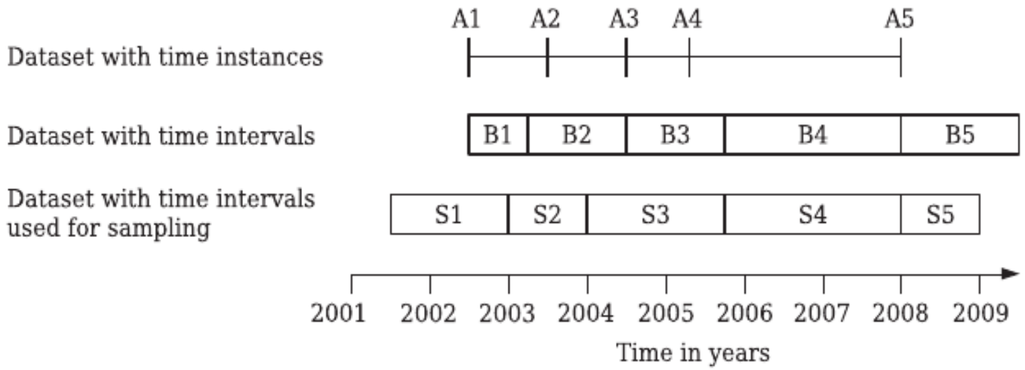
Other TGRASS notions

- Topology refers to temporal relations between time intervals in a STDS.

	A in relation to B	B in relation to A
	equivalent	equivalent
	overlap	overlap
	during	contain
	follows	precedes

Other TGRASS notions

- Temporal sampling is used to determine the state of one process during a second process.



Sampling time instances

	start
S1	A1
S2	A2
S3	A3,A4
S4	—
S5	A5

Sampling time intervals

	start	during	contain	overlap	equal
S1	B1	—	—	B1	—
S2	B2	—	—	B1,B2	—
S3	B3	B3	—	B2	—
S4	B4	—	—	—	B4
S5	B5	—	B5	—	—

Spatio-temporal modules

- **t.***: General modules to handle STDS of all types
- **t.rast.***: Modules that deal with STRDS
- **t.rast3d.***: Modules that deal with STR3DS
- **t.vect.***: Modules that deal with STVDS

TGRASS framework and workflow

SPATIO-TEMPORAL GIS MODULES

GRASS DATABASE

3D rasters | Rasters | Vectors

Create STDS
t.create

Assign time stamps
t.register

t.rast.to.vect

STR3DS **STRDS** **STVDS**

TEMPORAL DATABASE

General utility: t.list, t.info, t.remove, t.select, t.sample, t.topology, etc.

Other useful modules and Add-ons

g.gui.mapswipe

Gap-fill:
r.hants,
r.series.lwr

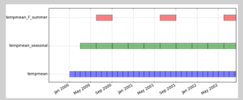
Nulls:
t.rast.nulls

Data extraction:
v.what.strds,
v.strds.stats,
t.rast.what.aggr

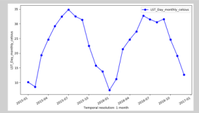
Others:
r.seasons,
r.regression.series,
r.bioclim

t.rast.out.xyz

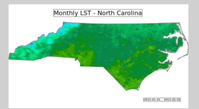
Visualization



g.gui.timeline



g.gui.tplot



g.gui.animation

t.rast.list,
t.rast.extract,
t.rast.univar

List, extract, and statistics

t.vect.list,
t.vect.extract,
t.vect.univar

t.rast.colors,
t.rast.gapfill

Pre-processing

t.rast.series

t.rast.aggregate

Processing and analysis

t.vect.what.strds
t.vect.observe.strds

t.rast.algebra

```
t.rast.algebra "C = A {+, contains} B"
t.rast.algebra "C = A + map(b)"
t.rast.algebra "C = (A[-1] + A[0] + A[1])/3"
```

t.rast.accumulate + t.rast.accdetect

ST algebra

t.vect.algebra

```
t.vect.algebra \
"C = A {|,equal|overlap,u} B"
t.vect.algebra \
"C = buff_p(A,1) {&,overlap,i} buff_p(B,1)"
```

t.rast.import
t.rast.export

Import/Export

t.vect.import
t.vect.export

Hands-on to raster time series in GRASS GIS

Let's first get the data and the code to follow this session

- modis_lst mapset (2Mb): download and unzip within the North Carolina location in `$HOME/grassdata/nc_spm_08_grass7`
- GRASS code
- R code

... and start GRASS GIS

```
grass74 $HOME/grassdata/nc_spm_08_grass7/modis_lst --gui
```

Set computational region and apply MASK

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Set computational region and apply MASK

```
  spectral="( 1 0 0 0 0 0 0 0 0 0 0 0 )"



############## For the workshop (what you have to do) ##############


## Download the ready to use mapset 'modis_lst' from:
## https://gitlab.com/veroandreo/grass-gis-geostat-2018
## and unzip it into North Carolina full LOCATION 'nc_spm_08_grass7'

# Get list of raster maps in the 'modis_lst' mapset
g.list type=raster

# Get info from one of the raster maps
r.info map=MOD11B3.A2015060.h11v05.single_LST_Day_6km



## Region settings and MASK

# Set region to NC extent with LST maps' spatial extent
```

List raster maps and get info

Set computational region and apply MASK

```
## Region settings and MASK

# Set region to NC state with LST maps' resolution
g.region -p vector=nc_state \
  align=MOD11B3.A2015060.h11v05.single_LST_Day_6km

#~ projection: 99 (Lambert Conformal Conic)
#~ zone:       0
#~ datum:      nad83
#~ ellipsoid:  a=6378137 es=0.006694380022900787
#~ north:      323380.12411493
#~ south:      9780.12411493
#~ west:       122934.46411531
#~ east:       934934.46411531
#~ nsres:      5600
#~ ewres:      5600
#~ rows:       56
#~ cols:       145
#~ cells:      8120
```

Set region

Set computational region and apply MASK

```
#~ east:         934934.46411531
#~ nsres:        5600
#~ ewres:        5600
#~ rows:         56
#~ cols:         145
#~ cells:        8120


# Set a MASK to nc_state boundary
r.mask vector=nc_state

# you should see this statement in the terminal from now on
#~ [Raster MASK present]



## Time series


# Create the STRDS
t.create type=strds temporaltype=absolute output=LST_Day_monthly \
```

Set MASK

Create a temporal dataset (STDS)

t.create

- Creates an SQLite container table in the temporal database
- Handles even huge amounts of maps by using the STDS as input
- We need to specify:
  - *type of maps* (raster, raster3d or vector)
  - *type of time* (absolute or relative)

Create a raster time series (STRDS)

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Create a raster time series (STRDS)

```
# Set a MASK to nc_state boundary
r.mask vector=nc_state

# you should see this statement in the terminal from now on:
#~ [Raster MASK present]



## Time series

# Create the STRDS
t.create type=strds temporaltype=absolute output=LST_Day_monthly \
  title="Monthly LST Day 5.6 km" \
  description="Monthly LST Day 5.6 km MOD11B3.006, 2015-2017"

# Check if the STRDS is created
t.list type=strds

# Get info about the STRDS
t.info input=LST_Day_monthly
```

Create the STRDS

Create a raster time series (STRDS)

```
## Time series

# Create the STRDS
t.create type=strds temporaltype=absolute output=LST_Day_monthly \
  title="Monthly LST Day 5.6 km" \
  description="Monthly LST Day 5.6 km MOD11B3.006, 2015-2017"

# Check if the STRDS is created
t.list type=strds

# Get info about the STRDS
t.info input=LST_Day_monthly



## Add time stamps to maps (i.e., register maps)

# in Unix systems
```

Check if the STRDS is created

Create a raster time series (STRDS)

```
# Create the STRDS
t.create type=strds temporaltype=absolute output=LST_Day_monthly \
  title="Monthly LST Day 5.6 km" \
  description="Monthly LST Day 5.6 km MOD11B3.006, 2015-2017"

# Check if the STRDS is created
t.list type=strds

# Get info about the STRDS
t.info input=LST_Day_monthly


## Add time stamps to maps (i.e., register maps)

# in Unix systems
t.register -i input=LST_Day_monthly \
 maps=`g.list type=raster pattern="MOD11B3*LST_Day*" separator=comma` \
 start="2015-01-01" increment="1 months"
```

Get info about the STRDS

Register maps into the STRDS

t.register

- Assigns time stamps to maps
- We need:
  - the *empty STDS* as input, i.e., the container table,
  - the *list of maps* to be registered,
  - the *start date*,
  - *increment* option along with the *-i* flag for interval creation

Register maps in STRDS (assign time stamps)

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################


########### Before the workshop (done for you in advance) #############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

For more options, check the t.register manual and related map registration wiki page.

Register maps in STRDS (assign time stamps)

```
# Check if the STRDS is created
t.list type=strds

# Get info about the STRDS
t.info input=LST_Day_monthly


## Add time stamps to maps (i.e., register maps)

# in Unix systems
t.register -i input=LST_Day_monthly \
 maps=`g.list type=raster pattern="MOD11B3*LST_Day*" separator=comma` \
 start="2015-01-01" increment="1 months"

# in MS Windows, first create the list of maps
g.list type=raster pattern="MOD11B3*LST_Day*" output=map_list.txt
t.register -i input=LST_Day_monthly \
 file=map_list.txt start="2015-01-01" increment="1 months"
```

Add time stamps to maps, i.e., register maps

For more options, check the t.register manual and related map registration wiki page.

Register maps in STRDS (assign time stamps)

```
## Add time stamps to maps (i.e., register maps)

# in Unix systems
t.register -i input=LST_Day_monthly \
 maps=`g.list type=raster pattern="MOD11B3*LST_Day*" separator=comma` \
 start="2015-01-01" increment="1 months"

# in MS Windows, first create the list of maps
g.list type=raster pattern="MOD11B3*LST_Day*" output=map_list.txt
t.register -i input=LST_Day_monthly \
 file=map_list.txt start="2015-01-01" increment="1 months"

# Check info again
t.info input=LST_Day_monthly

# Check the list of maps in the STRDS
t.rast.list input=LST_Day_monthly
```

Add time stamps to maps, i.e., register maps

For more options, check the t.register manual and related map registration wiki page.

Register maps in STRDS (assign time stamps)

```
t.register -i input=LST_Day_monthly \
 maps=`g.list type=raster pattern="MOD11B3*LST_Day*" separator=comma` \
 start="2015-01-01" increment="1 months"

# in MS Windows, first create the list of maps
g.list type=raster pattern="MOD11B3*LST_Day*" output=map_list.txt
t.register -i input=LST_Day_monthly \
 file=map_list.txt start="2015-01-01" increment="1 months"

# Check info again
t.info input=LST_Day_monthly

# Check the list of maps in the STRDS
t.rast.list input=LST_Day_monthly

# Check min and max per map
t.rast.list input=LST_Day_monthly columns=name,min,max
```

Check info again

For more options, check the t.register manual and related map registration wiki page.

Register maps in STRDS (assign time stamps)

```
# in MS Windows, first create the list of maps
g.list type=raster pattern="MOD11B3*LST_Day*" output=map_list.txt
t.register -i input=LST_Day_monthly \
 file=map_list.txt start="2015-01-01" increment="1 months"

# Check info again
t.info input=LST_Day_monthly

# Check the list of maps in the STRDS
t.rast.list input=LST_Day_monthly

# Check min and max per map
t.rast.list input=LST_Day_monthly columns=name,min,max


## Let's see a graphical representation of our STRDS
g.gui.timeline inputs=LST_Day_monthly
```

Check the list of maps in the STRDS

For more options, check the t.register manual and related map registration wiki page.

Register maps in STRDS (assign time stamps)

```
t.register -i input=LST_Day_monthly \
 file=map_list.txt start="2015-01-01" increment="1 months"

# Check info again
t.info input=LST_Day_monthly

# Check the list of maps in the STRDS
t.rast.list input=LST_Day_monthly

# Check min and max per map
t.rast.list input=LST_Day_monthly columns=name,min,max


## Let's see a graphical representation of our STRDS
g.gui.timeline inputs=LST_Day_monthly


## Temporal calculations: K*50 to Celsius
```

Check min and max per map

For more options, check the t.register manual and related map registration wiki page.

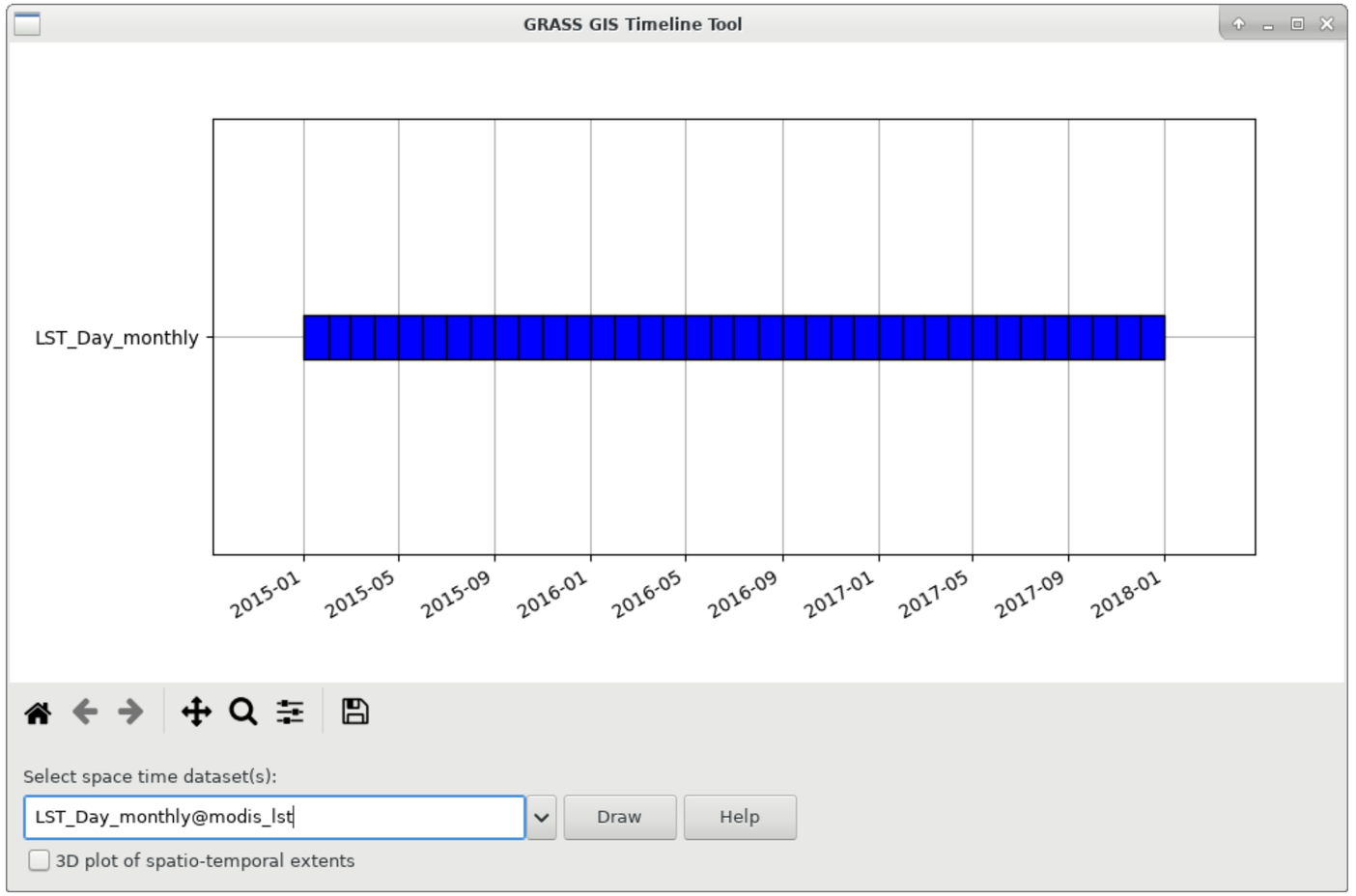
Graphical Representation of the STRDS

```
#!/bin/bash
############################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Graphical Representation of the STRDS

```
t.info input=LST_Day_monthly

# Check the list of maps in the STRDS
t.rast.list input=LST_Day_monthly

# Check min and max per map
t.rast.list input=LST_Day_monthly columns=name,min,max


## Let's see a graphical representation of our STRDS
g.gui.timeline inputs=LST_Day_monthly


## Temporal calculations: K*50 to Celsius

# Re-scale data to degrees Celsius
t.rast.algebra basename=LST_Day_monthly_celsius \
  expression="LST_Day_monthly_celsius = LST_Day_monthly * 0.02 - 273.15
```

Graphical representation of the time series



See g.gui.timeline manual page

Operations with temporal algebra

t.rast.algebra

- Wide range of temporal and spatial map algebra operations based on map's temporal topology
  - Temporal operators: union, intersection, etc.
  - Temporal functions: *start_time()*, *start_doy()*, etc.
  - Spatial operators (subset of r.mapcalc)
  - Temporal neighbourhood modifier: *[x,y,t]*
  - Other functions: *tsnap()*, *buff_t()* or *tshift()*

they can be combined in complex expressions!!

From K*50 to Celsius using the temporal calculator

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

From K*50 to Celsius using the temporal calculator

```
t.rast.list input=LST_Day_monthly columns=name,min,max



## Let's see a graphical representation of our STRDS
g.gui.timeline inputs=LST_Day_monthly



## Temporal calculations: K*50 to Celsius

# Re-scale data to degrees Celsius
t.rast.algebra basename=LST_Day_monthly_celsius \
  expression="LST_Day_monthly_celsius = LST_Day_monthly * 0.02 - 273.15

# Check info
t.info LST_Day_monthly_celsius

# some new features in upcoming grass76
t.rast.algebra basename=LST_Day_monthly_celsius suffix=gran \
```

Re-scale data to degrees Celsius

From K*50 to Celsius using the temporal calculator

```
g.gui.timeline inputs=LST_Day_monthly



## Temporal calculations: K*50 to Celsius


# Re-scale data to degrees Celsius
t.rast.algebra basename=LST_Day_monthly_celsius \
  expression="LST_Day_monthly_celsius = LST_Day_monthly * 0.02 - 273.15

# Check info
t.info LST_Day_monthly_celsius

# some new features in upcoming grass76
t.rast.algebra basename=LST_Day_monthly_celsius suffix=gran \
 expression="LST_Day_monthly_celsius = LST_Day_monthly * 0.02 - 273.15"



## Time series plots
```

Check info

Time series plot

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################


########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

For a single point, see g.gui.tplot. For a vector of points, see t.rast.what.

Time series plot

```
t.info LST_Day_monthly_celsius

# some new features in upcoming grass76
t.rast.algebra basename=LST_Day_monthly_celsius suffix=gran \
 expression="LST_Day_monthly_celsius = LST_Day_monthly * 0.02 - 273.15"



## Time series plots


# LST time series plot for the city center of Raleigh
g.gui.tplot strds=LST_Day_monthly_celsius \
  coordinates=641428.783478,229901.400746

# some new features in upcoming grass76
g.gui.tplot strds=LST_Day_monthly_celsius \
  coordinates=641428.783478,229901.400746 \
  title="Monthly LST. City center of Raleigh, NC " \
  xlabel="Time" ylabel="LST" \
  csv=raleigh_monthly_LST.csv
```

LST time series plot for the city center of Raleigh

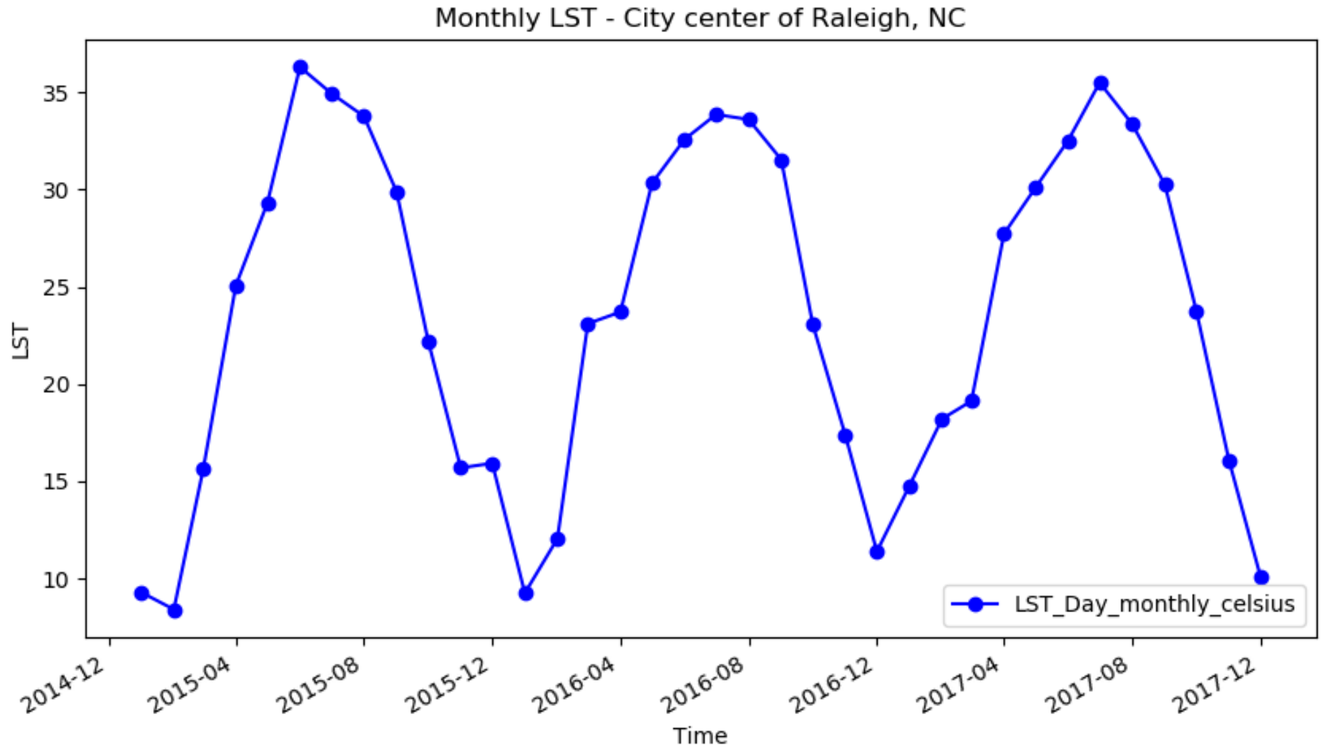
For a single point, see g.gui.tplot. For a vector of points, see t.rast.what.

Time series plot

```
## Time series plots

# LST time series plot for the city center of Raleigh
g.gui.tplot strds=LST_Day_monthly_celsius \
  coordinates=641428.783478,229901.400746

# some new features in upcoming grass76
g.gui.tplot strds=LST_Day_monthly_celsius \
  coordinates=641428.783478,229901.400746 \
  title="Monthly LST. City center of Raleigh, NC " \
  xlabel="Time" ylabel="LST" \
  csv=raleigh_monthly_LST.csv


## Get specific lists of maps

# Maps with minimum value lower than or equal to 5
```

New features in upcoming grass76

For a single point, see g.gui.tplot. For a vector of points, see t.rast.what.



Point coordinates can be typed directly, copied from the map display and pasted or directly chosen from the main map display.

Lists and selections

- **t.list** for listing STDS and maps registered in the temporal database,
- **t.rast.list** for maps in raster time series, and
- **t.vect.list** for maps in vector time series.

Listing examples

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Listing examples

```
## Get specific lists of maps

# Maps with minimum value lower than or equal to 5
t.rast.list input=LST_Day_monthly_celsius order=min \
 columns=name,start_time,min where="min <= '5.0'"

#~ name|start_time|min
#~ LST_Day_monthly_celsius_2015_02|2015-02-01 00:00:00|-1.31
#~ LST_Day_monthly_celsius_2017_01|2017-01-01 00:00:00|-0.89
#~ LST_Day_monthly_celsius_2015_01|2015-01-01 00:00:00|-0.25
#~ LST_Day_monthly_celsius_2016_01|2016-01-01 00:00:00|-0.17
#~ LST_Day_monthly_celsius_2016_02|2016-02-01 00:00:00|0.73
#~ LST_Day_monthly_celsius_2017_12|2017-12-01 00:00:00|1.69
#~ LST_Day_monthly_celsius_2016_12|2016-12-01 00:00:00|3.45

# Maps with maximum value higher than 30
```

Maps with minimum value lower than or equal to 5

Listing examples

```
#~ LST_Day_monthly_celsius_2017_12|2017-12-01 00:00:00|1.69
#~ LST_Day_monthly_celsius_2016_12|2016-12-01 00:00:00|3.45


# Maps with maximum value higher than 30
t.rast.list input=LST_Day_monthly_celsius order=max \
 columns=name,start_time,max where="max > '30.0'"

#~ name|start_time|max
#~ LST_Day_monthly_celsius_2017_04|2017-04-01 00:00:00|30.85
#~ LST_Day_monthly_celsius_2017_09|2017-09-01 00:00:00|32.45
#~ LST_Day_monthly_celsius_2016_05|2016-05-01 00:00:00|32.97
#~ LST_Day_monthly_celsius_2015_09|2015-09-01 00:00:00|33.49
#~ LST_Day_monthly_celsius_2017_05|2017-05-01 00:00:00|34.35
#~ LST_Day_monthly_celsius_2015_05|2015-05-01 00:00:00|34.53
#~ LST_Day_monthly_celsius_2017_08|2017-08-01 00:00:00|35.81
#~ LST_Day_monthly_celsius_2016_09|2016-09-01 00:00:00|36.33
#~ LST_Day_monthly_celsius_2016_08|2016-08-01 00:00:00|36.43
```

Maps with maximum value higher than 30

Listing examples

```
#~ LST_Day_monthly_celsius_2015_05|2015-05-01 00:00:00|34.53
#~ LST_Day_monthly_celsius_2017_08|2017-08-01 00:00:00|35.81
#~ LST_Day_monthly_celsius_2016_09|2016-09-01 00:00:00|36.33
#~ LST_Day_monthly_celsius_2016_08|2016-08-01 00:00:00|36.43


# Maps between two given dates
t.rast.list input=LST_Day_monthly_celsius columns=name,start_time \
 where="start_time >= '2015-05' and start_time <= '2015-08-01 00:00:00'

#~ name|start_time
#~ LST_Day_monthly_celsius_2015_05|2015-05-01 00:00:00
#~ LST_Day_monthly_celsius_2015_06|2015-06-01 00:00:00
#~ LST_Day_monthly_celsius_2015_07|2015-07-01 00:00:00
#~ LST_Day_monthly_celsius_2015_08|2015-08-01 00:00:00


# Maps from January
t.rast.list input=LST_Day_monthly_celsius columns=name,start_time \
 where="strftime('%m', start_time)='01'"
```

Maps between two given dates

Listing examples

```
#~ name|start_time
#~ LST_Day_monthly_celsius_2015_05|2015-05-01 00:00:00
#~ LST_Day_monthly_celsius_2015_06|2015-06-01 00:00:00
#~ LST_Day_monthly_celsius_2015_07|2015-07-01 00:00:00
#~ LST_Day_monthly_celsius_2015_08|2015-08-01 00:00:00


# Maps from January
t.rast.list input=LST_Day_monthly_celsius columns=name,start_time \
 where="strftime('%m', start_time)='01'"

#~ name|start_time
#~ LST_Day_monthly_celsius_2015_01|2015-01-01 00:00:00
#~ LST_Day_monthly_celsius_2016_01|2016-01-01 00:00:00
#~ LST_Day_monthly_celsius_2017_01|2017-01-01 00:00:00


## Descriptive statistics for STRDS
```

Maps from January

Descriptive statistics of LST time series

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Descriptive statistics of LST time series

```
#~ LST_Day_monthly_celsius_2015_01|2015-01-01 00:00:00
#~ LST_Day_monthly_celsius_2016_01|2016-01-01 00:00:00
#~ LST_Day_monthly_celsius_2017_01|2017-01-01 00:00:00



## Descriptive statistics for STRDS

# Print univariate stats for maps within STRDS
t.rast.univar input=LST_Day_monthly_celsius

#~ id|start|end|mean|min|max|mean_of_abs|stddev|variance|coeff_var|sum|
#~ LST_Day_monthly_celsius_2015_01@modis_lst|2015-01-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_02@modis_lst|2015-02-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_03@modis_lst|2015-03-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_04@modis_lst|2015-04-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_05@modis_lst|2015-05-01 00:00:00|2015-0

# Get extended statistics
t.rast.univar -e input=LST_Day_monthly_celsius
```

Print univariate stats for maps within STRDS

Descriptive statistics of LST time series

```
t.rast.univar input=LST_Day_monthly_celsius

#~ id|start|end|mean|min|max|mean_of_abs|stddev|variance|coeff_var|sum|
#~ LST_Day_monthly_celsius_2015_01@modis_lst|2015-01-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_02@modis_lst|2015-02-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_03@modis_lst|2015-03-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_04@modis_lst|2015-04-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_05@modis_lst|2015-05-01 00:00:00|2015-0


# Get extended statistics
t.rast.univar -e input=LST_Day_monthly_celsius

# Write the univariate stats output to a csv file
t.rast.univar input=LST_Day_monthly_celsius separator=comma \
  output=stats_LST_Day_monthly_celsius.csv


## Temporal aggregations (full series)
```

Get extended statistics

Descriptive statistics of LST time series

```
#~ LST_Day_monthly_celsius_2015_01@modis_lst|2015-01-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_02@modis_lst|2015-02-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_03@modis_lst|2015-03-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_04@modis_lst|2015-04-01 00:00:00|2015-0
#~ LST_Day_monthly_celsius_2015_05@modis_lst|2015-05-01 00:00:00|2015-0


# Get extended statistics
t.rast.univar -e input=LST_Day_monthly_celsius

# Write the univariate stats output to a csv file
t.rast.univar input=LST_Day_monthly_celsius separator=comma \
  output=stats_LST_Day_monthly_celsius.csv


## Temporal aggregations (full series)


# Get maximum LST in the STRDS
t.rast.series input=LST_Day_monthly_celsius \
  output=LST_Day_max method=maximum
```

Write the univariate stats output to a csv file

Temporal aggregation 1: Using the full time series

t.rast.series

- Aggregates full STRDS or parts of it using the *where* option
- Different methods available: average, minimum, maximum, median, mode, etc.

Maximum and minimum LST in the past 3 years

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Maximum and minimum LST in the past 3 years

```
# Get extended statistics
t.rast.univar -e input=LST_Day_monthly_celsius

# Write the univariate stats output to a csv file
t.rast.univar input=LST_Day_monthly_celsius separator=comma \
  output=stats_LST_Day_monthly_celsius.csv


## Temporal aggregations (full series)

# Get maximum LST in the STRDS
t.rast.series input=LST_Day_monthly_celsius \
  output=LST_Day_max method=maximum

# Get minimum LST in the STRDS
t.rast.series input=LST_Day_monthly_celsius \
  output=LST_Day_min method=minimum
```

Get maximum LST in the STRDS

Maximum and minimum LST in the past 3 years

```
  output=stats_LST_Day_monthly_celsius.csv


## Temporal aggregations (full series)

# Get maximum LST in the STRDS
t.rast.series input=LST_Day_monthly_celsius \
  output=LST_Day_max method=maximum

# Get minimum LST in the STRDS
t.rast.series input=LST_Day_monthly_celsius \
  output=LST_Day_min method=minimum

# Change color pallete to celsius
r.colors map=LST_Day_min,LST_Day_max color=celsius


## Display the new maps with mapswipe and compare them to elevation
```

Get minimum LST in the STRDS

Maximum and minimum LST in the past 3 years

```
# Get maximum LST in the STRDS
t.rast.series input=LST_Day_monthly_celsius \
  output=LST_Day_max method=maximum

# Get minimum LST in the STRDS
t.rast.series input=LST_Day_monthly_celsius \
  output=LST_Day_min method=minimum

# Change color pallete to celsius
r.colors map=LST_Day_min,LST_Day_max color=celsius


## Display the new maps with mapswipe and compare them to elevation

# LST_Day_max & elevation
g.gui.mapswipe first=LST_Day_max second=elev_state_500m
```

Change color pallete to celsius

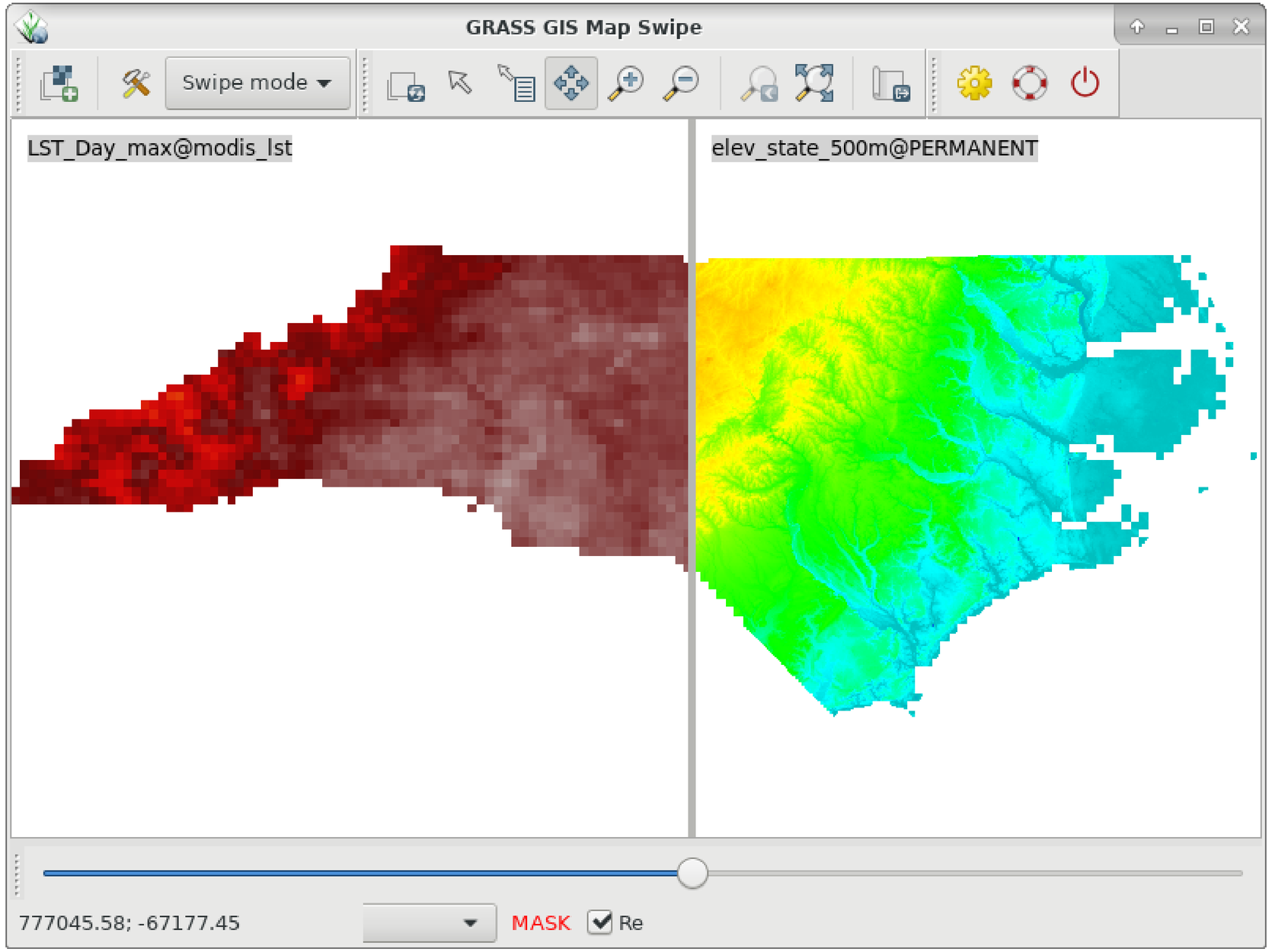
Compare maps with the Mapswipe tool

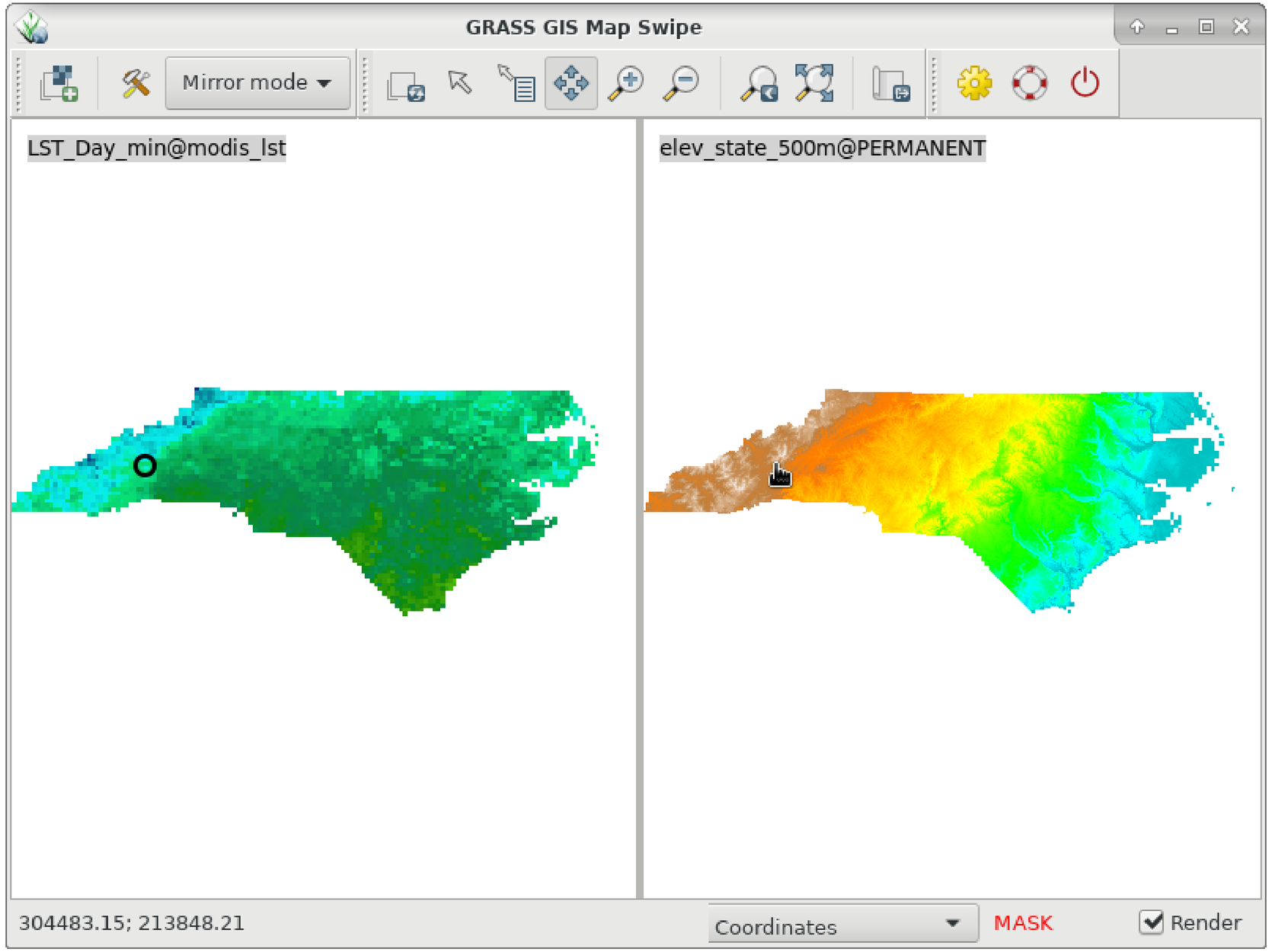
```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Compare maps with the Mapswipe tool

```
t.rast.series input=LST_Day_monthly_celsius \
  output=LST_Day_min method=minimum

# Change color pallete to celsius
r.colors map=LST_Day_min,LST_Day_max color=celsius


## Display the new maps with mapswipe and compare them to elevation

# LST_Day_max & elevation
g.gui.mapswipe first=LST_Day_max second=elev_state_500m

# LST_Day_min & elevation
g.gui.mapswipe first=LST_Day_min second=elev_state_500m


## Temporal operations with time variables
```

Display the new maps with mapswipe and compare them to elevation


GRASS GIS Map Swipe
Swipe mode
LST_Day_max@modis_lst
elev_state_500m@PERMANENT
777045.58; -67177.45
MASK
Re


GRASS GIS Map Swipe
Mirror mode
LST_Day_min@modis_lst
elev_state_500m@PERMANENT
304483.15; 213848.21
Coordinates
MASK
Render

Temporal operations using time variables

t.rast.mapcalc

- Performs spatio-temporal mapcalc expressions
- It allows for *spatial and temporal operators*, as well as *internal variables* in the expression string
- The temporal variables include: *start_time()*, *end_time()*, *start_month()*, *start_doy()*, etc.

The supported internal variables for the STRDS:

- td(), start_time(), end_time()

The supported internal variables for the current sample interval or instance:

- start_doy(), start_dow(), start_year(), start_month(), start_week(), start_day(), start_hour(), start_minute(), start_second()

... and the same for *end_**

Which is the month of the maximum LST?

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################


########### Before the workshop (done for you in advance) #############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

We could do this year-wise to know when the annual max LST occurs and assess trends

Which is the month of the maximum LST?

```
# LST_Day_max & elevation
g.gui.mapswipe first=LST_Day_max second=elev_state_500m

# LST_Day_min & elevation
g.gui.mapswipe first=LST_Day_min second=elev_state_500m


## Temporal operations with time variables

# Get month of maximum LST
t.rast.mapcalc -n inputs=LST_Day_monthly_celsius output=month_max_lst \
  expression="if(LST_Day_monthly_celsius == LST_Day_max, start_month(),
  basename=month_max_lst

# Get basic info
t.info month_max_lst
```

Get month of maximum LST

We could do this year-wise to know when the annual max LST occurs and assess trends

Which is the month of the maximum LST?

```
g.gui.mapswipe first=LST_Day_min second=elev_state_500m



## Temporal operations with time variables


# Get month of maximum LST
t.rast.mapcalc -n inputs=LST_Day_monthly_celsius output=month_max_lst \
  expression="if(LST_Day_monthly_celsius == LST_Day_max, start_month(),
  basename=month_max_lst

# Get basic info
t.info month_max_lst

# Get the earliest month in which the maximum appeared (method minimum)
t.rast.series input=month_max_lst method=minimum output=max_lst_date

# Remove month_max_lst strds
# we were only interested in the resulting aggregated map
t.remove -rf inputs=month_max_lst
```

Get basic info

We could do this year-wise to know when the annual max LST occurs and assess trends

Which is the month of the maximum LST?

```
## Temporal operations with time variables

# Get month of maximum LST
t.rast.mapcalc -n inputs=LST_Day_monthly_celsius output=month_max_lst \
  expression="if(LST_Day_monthly_celsius == LST_Day_max, start_month(),
  basename=month_max_lst

# Get basic info
t.info month_max_lst

# Get the earliest month in which the maximum appeared (method minimum)
t.rast.series input=month_max_lst method=minimum output=max_lst_date

# Remove month_max_lst strds
# we were only interested in the resulting aggregated map
t.remove -rf inputs=month_max_lst

# Note that the flags "-rf" force (immediate) removal of both
# the STRDS and the maps registered in it
```

Get the earliest month in which the maximum appeared

We could do this year-wise to know when the annual max LST occurs and assess trends

Which is the month of the maximum LST?

```
  basename=month_max_lst

# Get basic info
t.info month_max_lst

# Get the earliest month in which the maximum appeared (method minimum)
t.rast.series input=month_max_lst method=minimum output=max_lst_date

# Remove month_max_lst strds
# we were only interested in the resulting aggregated map
t.remove -rf inputs=month_max_lst

# Note that the flags "-rf" force (immediate) removal of both
# the STRDS and the maps registered in it.


## Display maps in a wx monitor

# Open a monitor
```

Remove month_max_lst strds

We could do this year-wise to know when the annual max LST occurs and assess trends

Display the resulting map from the CLI

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Display the resulting map from the CLI

```
# Remove month_max_lst strds
# we were only interested in the resulting aggregated map
t.remove -rf inputs=month_max_lst

# Note that the flags "-rf" force (immediate) removal of both
# the STRDS and the maps registered in it.



## Display maps in a wx monitor

# Open a monitor
d.mon wx0

# Display the raster map
d.rast map=max_lst_date

# Display boundary vector map
d.vect map=nc_state type=boundary color=#4D4D4D width=2
```

Open a monitor

Display the resulting map from the CLI

```
# Note that the flags "-rf" force (immediate) removal of both
# the STRDS and the maps registered in it.


## Display maps in a wx monitor

# Open a monitor
d.mon wx0

# Display the raster map
d.rast map=max_lst_date

# Display boundary vector map
d.vect map=nc_state type=boundary color=#4D4D4D width=2

# Add raster legend
d.legend -t raster=max_lst_date title="Month" \
  labelnum=6 title_fontsize=20 font=sans fontsize=18
```

Display raster map

Display the resulting map from the CLI

```
## Display maps in a wx monitor

# Open a monitor
d.mon wx0

# Display the raster map
d.rast map=max_lst_date

# Display boundary vector map
d.vect map=nc_state type=boundary color=#4D4D4D width=2

# Add raster legend
d.legend -t raster=max_lst_date title="Month" \
  labelnum=6 title_fontsize=20 font=sans fontsize=18

# Add scale bar
d.barscale length=200 units=kilometers segment=4 fontsize=14
```

Display only boundary of vector map

Display the resulting map from the CLI

```
# Open a monitor
d.mon wx0

# Display the raster map
d.rast map=max_lst_date

# Display boundary vector map
d.vect map=nc_state type=boundary color=#4D4D4D width=2

# Add raster legend
d.legend -t raster=max_lst_date title="Month" \
  labelnum=6 title_fontsize=20 font=sans fontsize=18

# Add scale bar
d.barscale length=200 units=kilometers segment=4 fontsize=14

# Add North arrow
d.northarrow style=1b text_color=black
```

Add raster legend

Display the resulting map from the CLI

```
# Display the raster map
d.rast map=max_lst_date

# Display boundary vector map
d.vect map=nc_state type=boundary color=#4D4D4D width=2

# Add raster legend
d.legend -t raster=max_lst_date title="Month" \
  labelnum=6 title_fontsize=20 font=sans fontsize=18

# Add scale bar
d.barscale length=200 units=kilometers segment=4 fontsize=14

# Add North arrow
d.northarrow style=1b text_color=black

# Add text
d.text -b text="Month of maximum LST 2015-2017" \
  color=black align=cc font=sans size=12
```

Add scale bar

Display the resulting map from the CLI

```
# Display boundary vector map
d.vect map=nc_state type=boundary color=#4D4D4D width=2

# Add raster legend
d.legend -t raster=max_lst_date title="Month" \
  labelnum=6 title_fontsize=20 font=sans fontsize=18

# Add scale bar
d.barscale length=200 units=kilometers segment=4 fontsize=14

# Add North arrow
d.northarrow style=1b text_color=black

# Add text
d.text -b text="Month of maximum LST 2015-2017" \
  color=black align=cc font=sans size=12


## Temporal aggregation (granularity of three months)
```

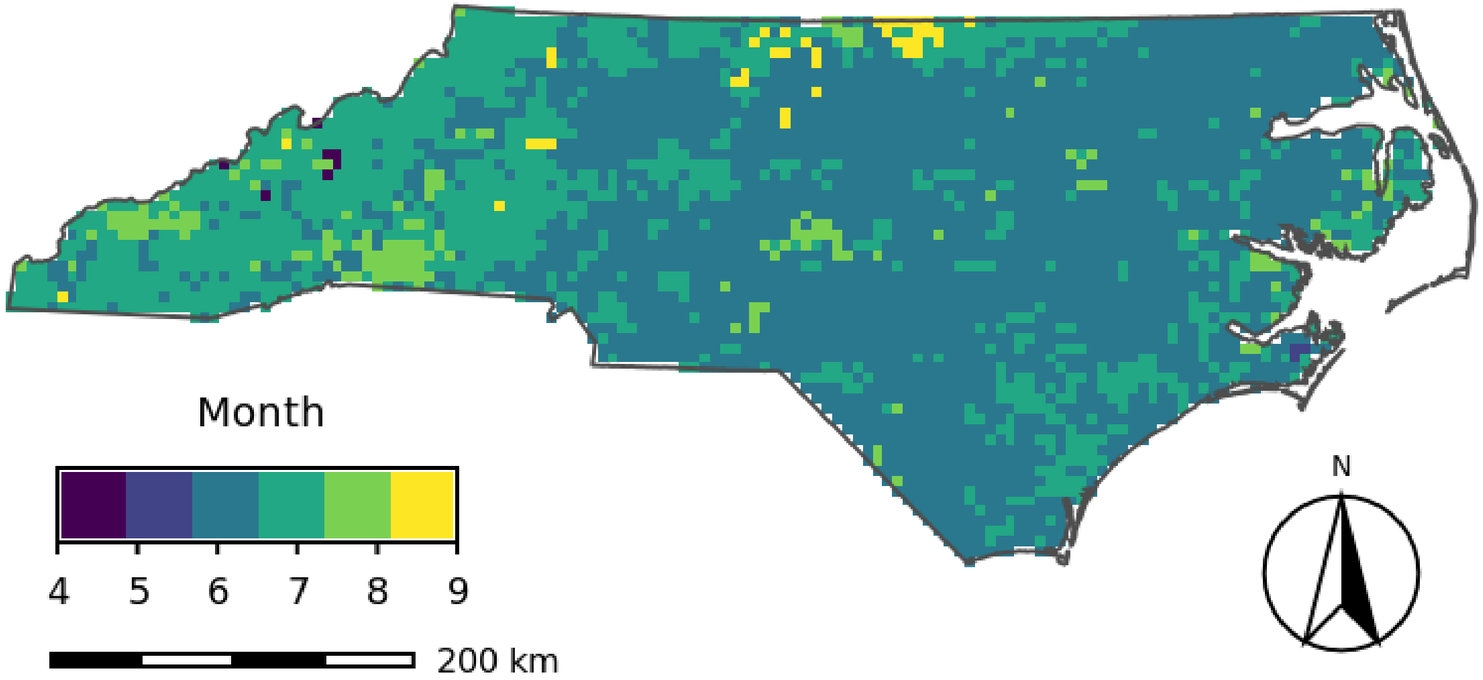
Add North arrow

Display the resulting map from the CLI

```
d.legend -t raster=max_lst_date title="Month" \
  labelnum=6 title_fontsize=20 font=sans fontsize=18

# Add scale bar
d.barscale length=200 units=kilometers segment=4 fontsize=14

# Add North arrow
d.northarrow style=1b text_color=black

# Add text
d.text -b text="Month of maximum LST 2015-2017" \
  color=black align=cc font=sans size=12


## Temporal aggregation (granularity of three months)

# 3-month mean LST
t.rast.aggregate input=LST_Day_monthly_celsius \
```

Add title text

Month of maximum LST 2015-2017



Temporal aggregation 2: using granularity

t.rast.aggregate

- Aggregates raster maps in STRDS with different **granularities**
- *where* option allows to set specific dates for the aggregation
- Different methods available: average, minimum, maximum, median, mode, etc.

From monthly to seasonal LST

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) #############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

From monthly to seasonal LST

```
d.northarrow style=1b text_color=black

# Add text
d.text -b text="Month of maximum LST 2015-2017" \
  color=black align=cc font=sans size=12


## Temporal aggregation (granularity of three months)

# 3-month mean LST
t.rast.aggregate input=LST_Day_monthly_celsius \
  output=LST_Day_mean_3month \
  basename=LST_Day_mean_3month suffix=gran \
  method=average granularity="3 months"

# Check info
t.info input=LST_Day_mean_3month
```

3-month mean LST

From monthly to seasonal LST

```
## Temporal aggregation (granularity of three months)

# 3-month mean LST
t.rast.aggregate input=LST_Day_monthly_celsius \
  output=LST_Day_mean_3month \
  basename=LST_Day_mean_3month suffix=gran \
  method=average granularity="3 months"

# Check info
t.info input=LST_Day_mean_3month

# Check map list
t.rast.list input=LST_Day_mean_3month

#~ name|mapset|start_time|end_time
#~ LST_Day_mean_3month_2015_01|modis_lst|2015-01-01 00:00:00|2015-04-01
```

Check info

From monthly to seasonal LST

```
# Check info
t.info input=LST_Day_mean_3month

# Check map list
t.rast.list input=LST_Day_mean_3month

#~ name|mapset|start_time|end_time
#~ LST_Day_mean_3month_2015_01|modis_lst|2015-01-01 00:00:00|2015-04-01
#~ LST_Day_mean_3month_2015_04|modis_lst|2015-04-01 00:00:00|2015-07-01
#~ LST_Day_mean_3month_2015_07|modis_lst|2015-07-01 00:00:00|2015-10-01
#~ LST_Day_mean_3month_2015_10|modis_lst|2015-10-01 00:00:00|2016-01-01
#~ LST_Day_mean_3month_2016_01|modis_lst|2016-01-01 00:00:00|2016-04-01
#~ LST_Day_mean_3month_2016_04|modis_lst|2016-04-01 00:00:00|2016-07-01
#~ LST_Day_mean_3month_2016_07|modis_lst|2016-07-01 00:00:00|2016-10-01
#~ LST_Day_mean_3month_2016_10|modis_lst|2016-10-01 00:00:00|2017-01-01
#~ LST_Day_mean_3month_2017_01|modis_lst|2017-01-01 00:00:00|2017-04-01
#~ LST_Day_mean_3month_2017_04|modis_lst|2017-04-01 00:00:00|2017-07-01
#~ LST_Day_mean_3month_2017_07|modis_lst|2017-07-01 00:00:00|2017-10-01
```

Check map list

***Task**: Compare the monthly and sesonal timelines with*

g.gui.timeline

Display seasonal LST using frames in wx monitor

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Display seasonal LST using frames in wx monitor

```
#~ LST_Day_mean_3month_2016_04|modis_lst|2016-04-01 00:00:00|2016-07-01
#~ LST_Day_mean_3month_2016_07|modis_lst|2016-07-01 00:00:00|2016-10-01
#~ LST_Day_mean_3month_2016_10|modis_lst|2016-10-01 00:00:00|2017-01-01
#~ LST_Day_mean_3month_2017_01|modis_lst|2017-01-01 00:00:00|2017-04-01
#~ LST_Day_mean_3month_2017_04|modis_lst|2017-04-01 00:00:00|2017-07-01
#~ LST_Day_mean_3month_2017_07|modis_lst|2017-07-01 00:00:00|2017-10-01
#~ LST_Day_mean_3month_2017_10|modis_lst|2017-10-01 00:00:00|2018-01-01



## Display seasonal LST using frames

# Set STRDS color table to celsius degrees
t.rast.colors input=LST_Day_mean_3month color=celsius

# Start a new graphics monitor, the data will be rendered to
# /tmp/map.png image output file of size 640x360px
d.mon cairo out=frames.png width=640 height=360 resolution=4
```

Set STRDS color table to celsius degrees

Display seasonal LST using frames in wx monitor

```
#~ LST_Day_mean_3month_2017_07|modis_lst|2017-07-01 00:00:00|2017-10-01
#~ LST_Day_mean_3month_2017_10|modis_lst|2017-10-01 00:00:00|2018-01-01



## Display seasonal LST using frames


# Set STRDS color table to celsius degrees
t.rast.colors input=LST_Day_mean_3month color=celsius

# Start a new graphics monitor, the data will be rendered to
# /tmp/map.png image output file of size 640x360px
d.mon cairo out=frames.png width=640 height=360 resolution=4

# create a first frame
d.frame -c frame=first at=0,50,0,50
d.rast map=LST_Day_mean_3month_2015_07
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Jul-Sep 2015' color=black font=sans size=10
```

Start a new cairo monitor

Display seasonal LST using frames in wx monitor

```
# Display seasonal LST using frames

# Set STRDS color table to celsius degrees
t.rast.colors input=LST_Day_mean_3month color=celsius

# Start a new graphics monitor, the data will be rendered to
# /tmp/map.png image output file of size 640x360px
d.mon cairo out=frames.png width=640 height=360 resolution=4

# create a first frame
d.frame -c frame=first at=0,50,0,50
d.rast map=LST_Day_mean_3month_2015_07
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Jul-Sep 2015' color=black font=sans size=10

# create a second frame
d.frame -c frame=second at=0,50,50,100
d.rast map=LST_Day_mean_3month_2015_10
d.vect map=nc_state type=boundary color=#4D4D4D width=2
```

Create first frame

Display seasonal LST using frames in wx monitor

```
d.mon cairo out=frames.png width=640 height=360 resolution=4


# create a first frame
d.frame -c frame=first at=0,50,0,50
d.rast map=LST_Day_mean_3month_2015_07
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Jul-Sep 2015' color=black font=sans size=10


# create a second frame
d.frame -c frame=second at=0,50,50,100
d.rast map=LST_Day_mean_3month_2015_10
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Oct-Dec 2015' color=black font=sans size=10


# create a third frame
d.frame -c frame=third at=50,100,0,50
d.rast map=LST_Day_mean_3month_2015_01
d.vect map=nc_state type=boundary color=#4D4D4D width=2
```

Create second frame

Display seasonal LST using frames in wx monitor

```
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Jul-Sep 2015' color=black font=sans size=10

# create a second frame
d.frame -c frame=second at=0,50,50,100
d.rast map=LST_Day_mean_3month_2015_10
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Oct-Dec 2015' color=black font=sans size=10

# create a third frame
d.frame -c frame=third at=50,100,0,50
d.rast map=LST_Day_mean_3month_2015_01
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Jan-Mar 2015' color=black font=sans size=10

# create a fourth frame
d.frame -c frame=fourth at=50,100,50,100
d.rast map=LST_Day_mean_3month_2015_04
d.vect map=nc_state type=boundary color=#4D4D4D width=2
```

Create third frame

Display seasonal LST using frames in wx monitor

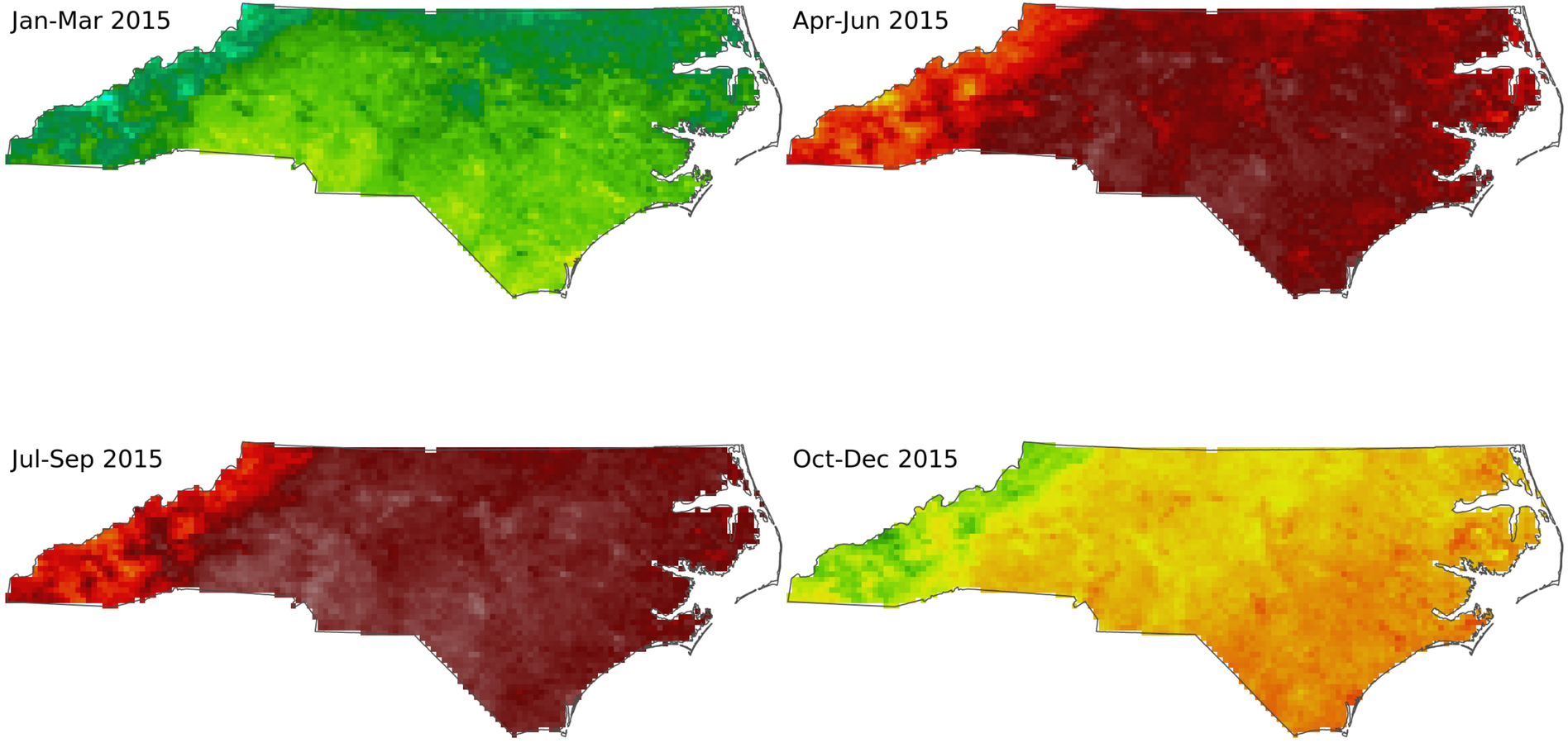
```
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Oct-Dec 2015' color=black font=sans size=10

# create a third frame
d.frame -c frame=third at=50,100,0,50
d.rast map=LST_Day_mean_3month_2015_01
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Jan-Mar 2015' color=black font=sans size=10

# create a fourth frame
d.frame -c frame=fourth at=50,100,50,100
d.rast map=LST_Day_mean_3month_2015_04
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Apr-Jun 2015' color=black font=sans size=10

# release monitor
d.mon -r
```

Create fourth frame

Display seasonal LST using frames in wx monitor

```
d.rast map=LST_Day_mean_3month_2015_01
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Jan-Mar 2015' color=black font=sans size=10

# create a fourth frame
d.frame -c frame=fourth at=50,100,50,100
d.rast map=LST_Day_mean_3month_2015_04
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Apr-Jun 2015' color=black font=sans size=10

# release monitor
d.mon -r


## Time series animation

# Animation of seasonal LST
g.gui.animation strds=LST_Day_mean_3month
```

Release monitor



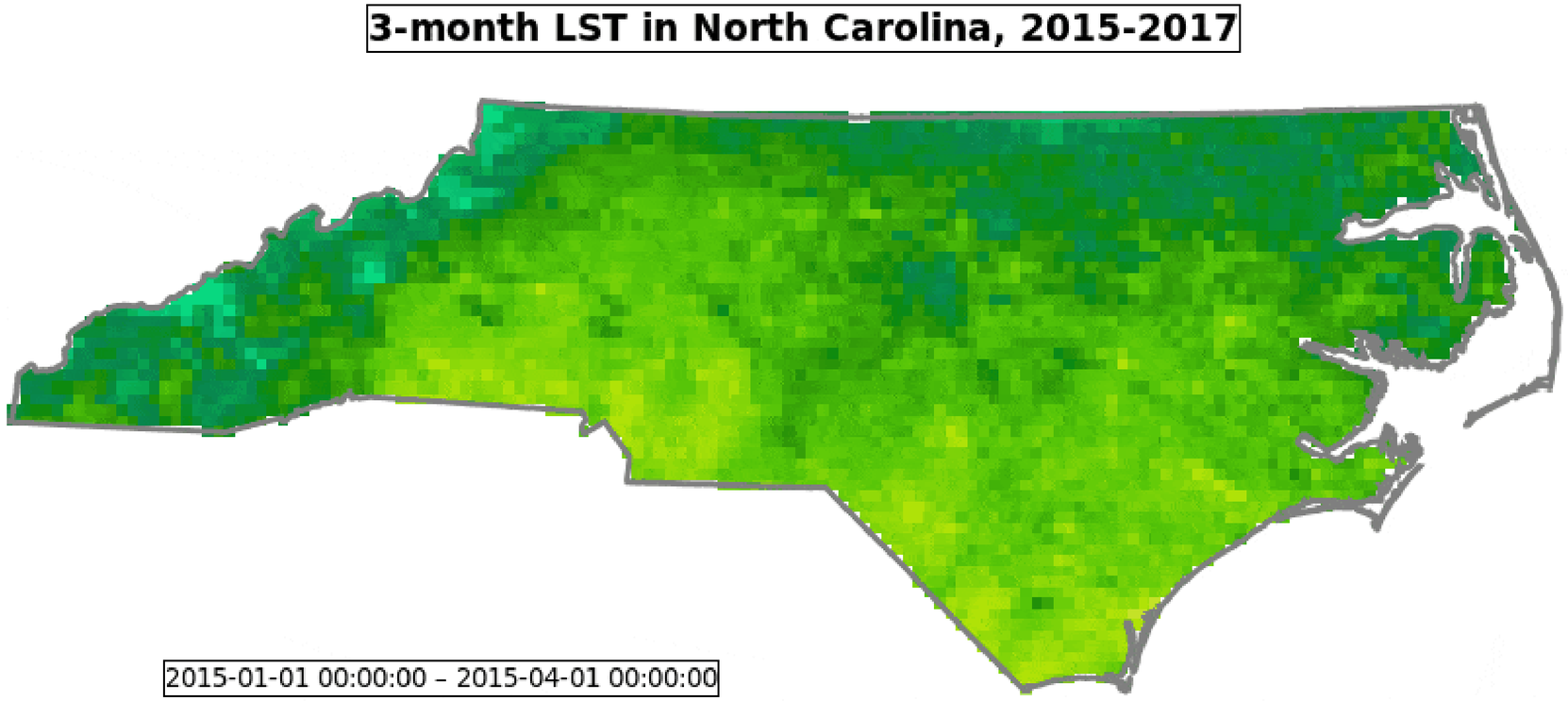
3-month average LST in 2015

***Task**: Now that you know t.rast.aggregate, extract the month of maximum LST per year and then test if there's any positive or negative trend, i.e., if maximum LST values are observed later or earlier with time (years)*

One solution could be...

```
t.rast.aggregate input=LST_Day_monthly_celsius output=month_max_LST_per
  basename=month_max_LST suffix=gran \
  method=max_raster granularity="1 year"

t.rast.series input=month_max_LST_per_year \
  output=slope_month_max_LST \
  method=slope
```

Animations


3-month LST in North Carolina, 2015-2017
2015-01-01 00:00:00 – 2015-04-01 00:00:00

Animation of seasonal LST time series

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################


########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

See g.gui.animation manual for further options and tweaks

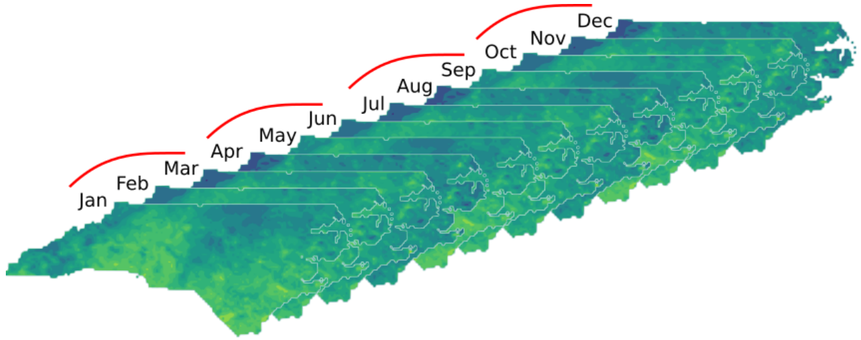
Animation of seasonal LST time series

```
# create a fourth frame
d.frame -c frame=fourth at=50,100,50,100
d.rast map=LST_Day_mean_3month_2015_04
d.vect map=nc_state type=boundary color=#4D4D4D width=2
d.text text='Apr-Jun 2015' color=black font=sans size=10

# release monitor
d.mon -r



## Time series animation

# Animation of seasonal LST
g.gui.animation strds=LST_Day_mean_3month



## Long term monthly averages

# Seasonal variation LST
```

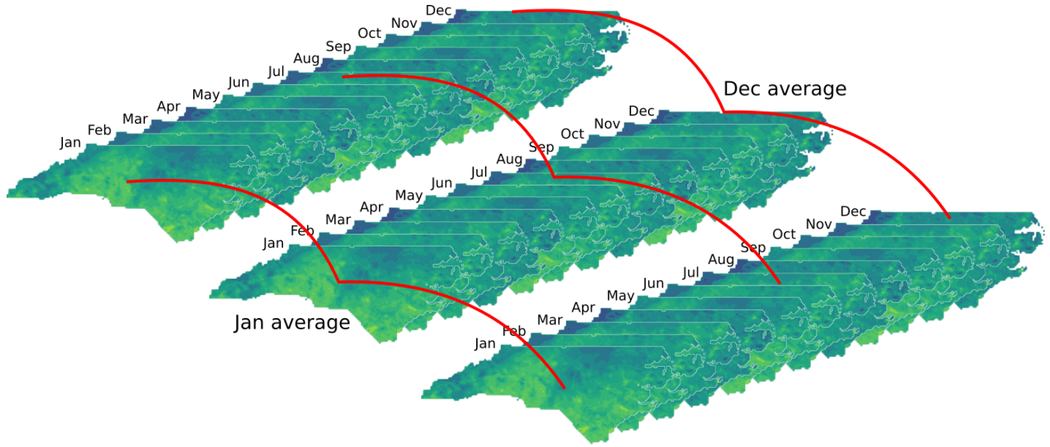
Animation of seasonal LST

See g.gui.animation manual for further options and tweaks

Aggregation vs Climatology



Granularity aggregation



Climatology aggregation

Monthly climatologies

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Monthly climatologies

```
## Time series animation

# Animation of seasonal LST
g.gui.animation strds=LST_Day_mean_3month


## Long term monthly averages

# January average LST
t.rast.series input=LST_Day_monthly_celsius method=average \
  where="strftime('%m', start_time)='01'" \
  output=LST_average_jan

# for all months
for MONTH in `seq -w 1 12` ; do
  t.rast.series input=LST_Day_monthly_celsius method=average \
  where="strftime('%m', start_time)='${MONTH}'" \
```

January average LST

Monthly climatologies

```
## Long term monthly averages

# January average LST
t.rast.series input=LST_Day_monthly_celsius method=average \
  where="strftime('%m', start_time)='01'" \
  output=LST_average_jan

# for all months
for MONTH in `seq -w 1 12` ; do
 t.rast.series input=LST_Day_monthly_celsius method=average \
  where="strftime('%m', start_time)='${MONTH}'" \
  output=LST_average_${MONTH}
done


## Anomalies
```

Climatology for all months

***Task**: Compare monthly means with "climatological" means*

Annual anomalies

$$Std_Anomaly_i = \frac{Average_i - Average}{SD}$$

- We need overall average and standard deviation
- We need yearly averages

Annual anomalies

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Annual anomalies

```
# for all months
for MONTH in `seq -w 1 12` ; do
 t.rast.series input=LST_Day_monthly_celsius method=average \
  where="strftime('%m', start_time)='${MONTH}'" \
  output=LST_average_${MONTH}
done



## Anomalies


# Get general average
t.rast.series input=LST_Day_monthly_celsius \
 method=average output=LST_average

# Get general SD
t.rast.series input=LST_Day_monthly_celsius \
 method=stddev output=LST_sd
```

Get general average

Annual anomalies

```
  output=LST_average_${MONTH}
done



## Anomalies


# Get general average
t.rast.series input=LST_Day_monthly_celsius \
 method=average output=LST_average

# Get general SD
t.rast.series input=LST_Day_monthly_celsius \
 method=stddev output=LST_sd

# Get annual averages
t.rast.aggregate input=LST_Day_monthly_celsius \
 method=average granularity="1 years" \
 output=LST_yearly_average basename=LST_yearly_average
```

Get general SD

Annual anomalies

```
# Get general average
t.rast.series input=LST_Day_monthly_celsius \
 method=average output=LST_average

# Get general SD
t.rast.series input=LST_Day_monthly_celsius \
 method=stddev output=LST_sd

# Get annual averages
t.rast.aggregate input=LST_Day_monthly_celsius \
 method=average granularity="1 years" \
 output=LST_yearly_average basename=LST_yearly_average

# Estimate annual anomalies
t.rast.algebra basename=LST_year_anomaly \
 expression="LST_year_anomaly = (LST_yearly_average - map(LST_average))
```

Get annual averages

Annual anomalies

```
# Get general SD
t.rast.series input=LST_Day_monthly_celsius \
 method=stddev output=LST_sd

# Get annual averages
t.rast.aggregate input=LST_Day_monthly_celsius \
 method=average granularity="1 years" \
 output=LST_yearly_average basename=LST_yearly_average

# Estimate annual anomalies
t.rast.algebra basename=LST_year_anomaly \
 expression="LST_year_anomaly = (LST_yearly_average - map(LST_average))

# Set difference color table
t.rast.colors input=LST_year_anomaly color=difference

# Animation of annual anomalies
```

Estimate annual anomalies

Annual anomalies

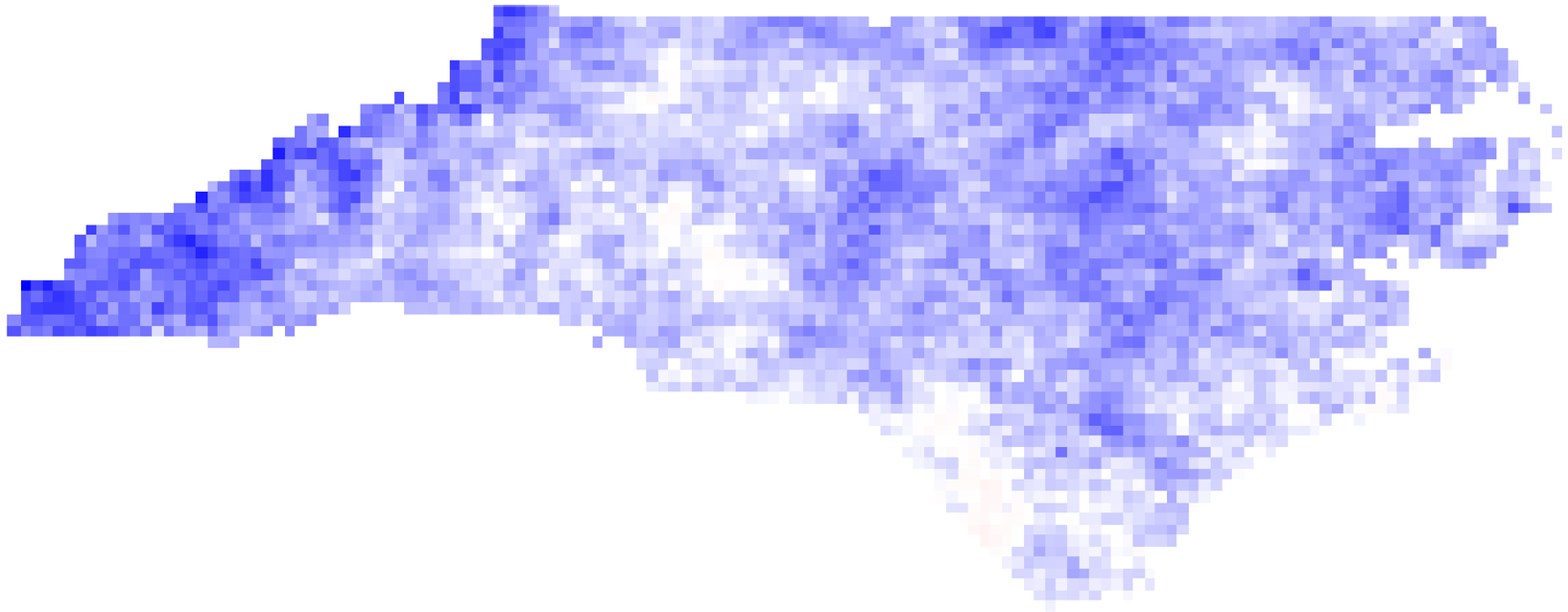
```
 method=stddev output=LST_sd

# Get annual averages
t.rast.aggregate input=LST_Day_monthly_celsius \
 method=average granularity="1 years" \
 output=LST_yearly_average basename=LST_yearly_average

# Estimate annual anomalies
t.rast.algebra basename=LST_year_anomaly \
 expression="LST_year_anomaly = (LST_yearly_average - map(LST_average))

# Set difference color table
t.rast.colors input=LST_year_anomaly color=difference

# Animation of annual anomalies
g.gui.animation strds=LST_year_anomaly


## Extract zonal statistics for areas
```

Set color table

Annual anomalies

```
t.rast.aggregate input=LST_Day_monthly_celsius \
 method=average granularity="1 years" \
 output=LST_yearly_average basename=LST_yearly_average

# Estimate annual anomalies
t.rast.algebra basename=LST_year_anomaly \
 expression="LST_year_anomaly = (LST_yearly_average - map(LST_average))"

# Set difference color table
t.rast.colors input=LST_year_anomaly color=difference

# Animation of annual anomalies
g.gui.animation strds=LST_year_anomaly


## Extract zonal statistics for areas

# Install v.strds.stats add-on
g.extension extension=v.strds.stats
```

Animation



Zonal statistics in raster time series

v.strds.stats

- Allows to obtain spatially aggregated time series data for polygons in a vector map

Extract mean LST for Raleigh (NC) urban area

```
#!/bin/bash
############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
############################################################################



########### Before the workshop (done for you in advance) ############

# Install i.modis add-on (requires pymodis library - www.pymodis.org)
g.extension extension=i.modis

# Download and import MODIS LST data
# Note: User needs to be registered at Earthdata:
# https://urs.earthdata.nasa.gov/users/new
i.modis.download settings=$HOME/gisdata/SETTING \
  product=lst_terra_monthly_5600 \
```

Extract mean LST for Raleigh (NC) urban area

```
 expression="LST_year_anomaly = (LST_yearly_average - map(LST_average))

# Set difference color table
t.rast.colors input=LST_year_anomaly color=difference

# Animation of annual anomalies
g.gui.animation strds=LST_year_anomaly



## Extract zonal statistics for areas


# Install v.strds.stats add-on
g.extension extension=v.strds.stats

# Extract seasonal average LST for Raleigh urban area
v.strds.stats input=urbanarea strds=LST_Day_mean_3month \
  where="NAME == 'Raleigh'" \
  output=raleigh_aggr_lst method=average
```

Install v.strds.stats add-on

Extract mean LST for Raleigh (NC) urban area

```
# Animation of annual anomalies
g.gui.animation strds=LST_year_anomaly



## Extract zonal statistics for areas


# Install v.strds.stats add-on
g.extension extension=v.strds.stats

# Extract seasonal average LST for Raleigh urban area
v.strds.stats input=urbanarea strds=LST_Day_mean_3month \
  where="NAME == 'Raleigh'" \
  output=raleigh_aggr_lst method=average

# Save the attribute table of the new vector into a csv file
v.db.select map=raleigh_aggr_lst file=lst_raleigh

#~ cat|OBJECTID|UA|NAME|UA_TYPE|LST_Day_monthly_celsius_2015_01_01_aver
```

Extract seasonal average LST for Raleigh urban area

Extract mean LST for Raleigh (NC) urban area

```
## Extract zonal statistics for areas

# Install v.strds.stats add-on
g.extension extension=v.strds.stats

# Extract seasonal average LST for Raleigh urban area
v.strds.stats input=urbanarea strds=LST_Day_mean_3month \
  where="NAME == 'Raleigh'" \
  output=raleigh_aggr_lst method=average

# Save the attribute table of the new vector into a csv file
v.db.select map=raleigh_aggr_lst file=lst_raleigh

#~ cat|OBJECTID|UA|NAME|UA_TYPE|LST_Day_monthly_celsius_2015_01_01_aver
#~ 55|55|73261|Raleigh|UA|8.4169230769231|7.8176923076923|15.17923076

### Some extra examples ###
```

Save the attribute table of the new vector into a csv file

Read and plot Raleigh vector in RStudio

```
##############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
##############################################################################

## Plot `raleigh_aggr_lst` vector in rstudio

# Call rstudio
rstudio &

# Load libraries
library(rgrass7)
library(sf)
library(dplyr)
library(ggplot2)
library(mapview)
```

Read and plot Raleigh vector in RStudio

```
##############################################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
##############################################################################


## Plot `raleigh_aggr_lst` vector in rstudio


# Call rstudio
rstudio &

# Load libraries
library(rgrass7)
library(sf)
library(dplyr)
library(ggplot2)
library(mapview)
```

Call RStudio

Read and plot Raleigh vector in RStudio

```
## Plot `raleigh_aggr_lst` vector in rstudio

# Call rstudio
rstudio &

# Load libraries
library(rgrass7)
library(sf)
library(dplyr)
library(ggplot2)
library(mapview)

# read vector
raleigh <- readVECT("raleigh_aggr_lst")

# sp
spplot(raleigh[,6:17])
```

Load libraries

Read and plot Raleigh vector in RStudio

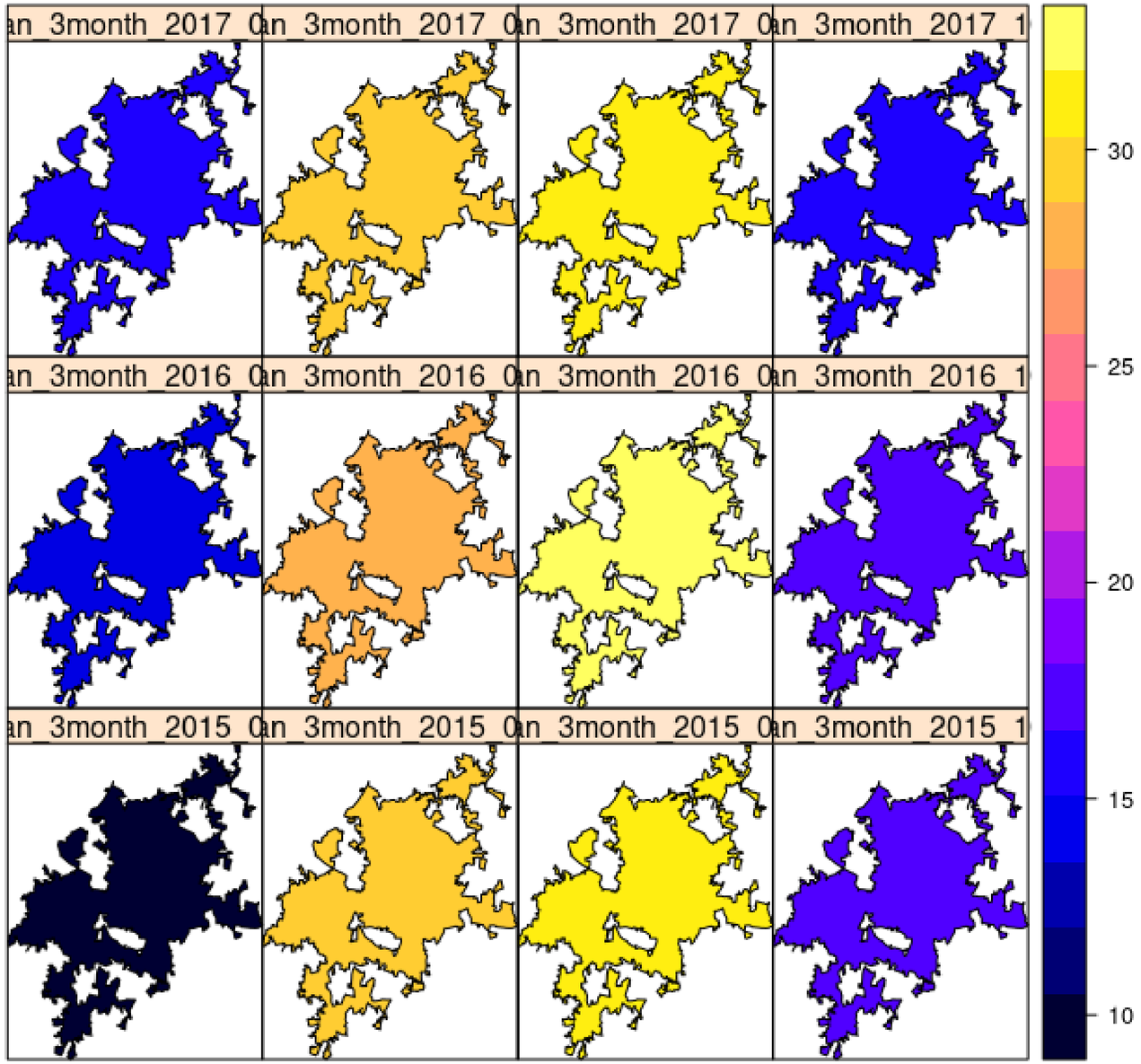
```
# Load libraries
library(rgrass7)
library(sf)
library(dplyr)
library(ggplot2)
library(mapview)

# read vector
raleigh <- readVECT("raleigh_aggr_lst")

# sp
spplot(raleigh[,6:17])

# sf + ggplot
raleigh_sf <- st_as_sf(raleigh)

# gather the table into season and mean LST (we do only 2015)
```

Read vector raleigh_aggr_lst

Read and plot Raleigh vector in RStudio

```
library(sf)
library(dplyr)
library(ggplot2)
library(mapview)

# read vector
raleigh <- readVECT("raleigh_aggr_lst")

# sp
spplot(raleigh[,6:17])

# sf + ggplot
raleigh_sf <- st_as_sf(raleigh)

# gather the table into season and mean LST (we do only 2015)
raleigh_gather <- raleigh_sf %>%
                  gather(LST_Day_mean_3month_2015_01_01_average,
                         LST_Day_mean_3month_2015_04_01_average,
```

Use sp tools to plot vector map


an_3month_2017_0
an_3month_2017_0
an_3month_2017_0
an_3month_2017_1
an_3month_2016_0
an_3month_2016_0
an_3month_2016_0
an_3month_2016_1
an_3month_2015_0
an_3month_2015_0
an_3month_2015_0
an_3month_2015_1
30
25
20
15
10

Read and plot Raleigh vector in RStudio

```
##############################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
##############################################################

## Plot `raleigh_aggr_lst` vector in rstudio

# Call rstudio
rstudio &

# Load libraries
library(rgrass7)
library(sf)
library(dplyr)
library(ggplot2)
library(mapview)
```

Read and plot Raleigh vector in RStudio

```
library(mapview)

# read vector
raleigh <- readVECT("raleigh_aggr_lst")

# sp
spplot(raleigh[,6:17])

# sf + ggplot
raleigh_sf <- st_as_sf(raleigh)

# gather the table into season and mean LST (we do only 2015)
raleigh_gather <- raleigh_sf %>%
                  gather(LST_Day_mean_3month_2015_01_01_average,
                         LST_Day_mean_3month_2015_04_01_average,
                         LST_Day_mean_3month_2015_07_01_average,
                         LST_Day_mean_3month_2015_10_01_average,
                         key="season", value="LST")
```

Convert from sp to sf

Read and plot Raleigh vector in RStudio

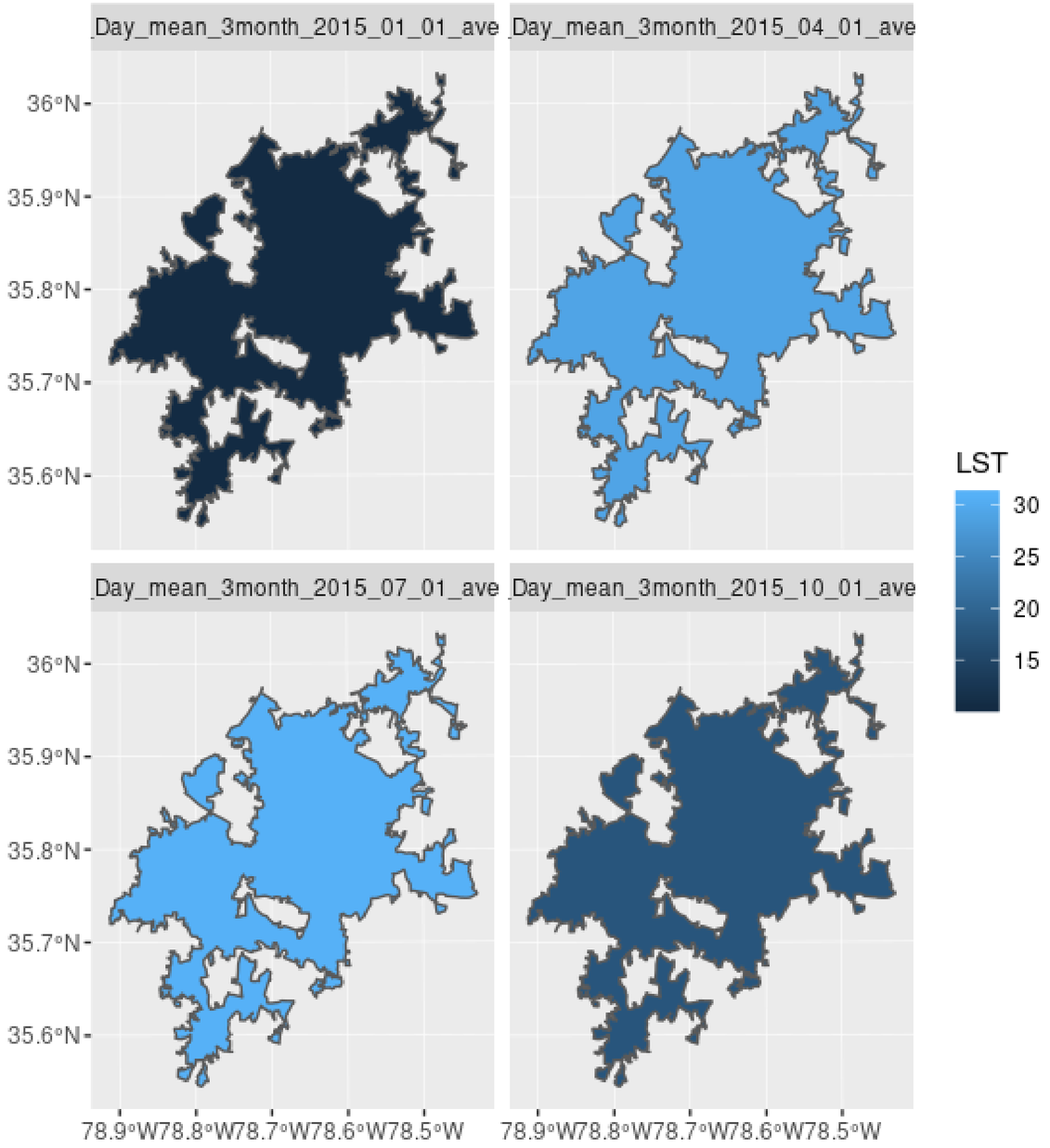
```
# sp
spplot(raleigh[,6:17])

# sf + ggplot
raleigh_sf <- st_as_sf(raleigh)

# gather the table into season and mean LST (we do only 2015)
raleigh_gather <- raleigh_sf %>%
                  gather(LST_Day_mean_3month_2015_01_01_average,
                         LST_Day_mean_3month_2015_04_01_average,
                         LST_Day_mean_3month_2015_07_01_average,
                         LST_Day_mean_3month_2015_10_01_average,
                         key="season", value="LST")

# ggplot
ggplot() + geom_sf(data = raleigh_gather, aes(fill = LST)) +
           facet_wrap(~season, ncol=2)

# mapview
```

Gather the table into season and mean LST (we do only 2015)

Read and plot Raleigh vector in RStudio

```
# sf + ggplot
raleigh_sf <- st_as_sf(raleigh)

# gather the table into season and mean LST (we do only 2015)
raleigh_gather <- raleigh_sf %>%
                  gather(LST_Day_mean_3month_2015_01_01_average,
                         LST_Day_mean_3month_2015_04_01_average,
                         LST_Day_mean_3month_2015_07_01_average,
                         LST_Day_mean_3month_2015_10_01_average,
                         key="season", value="LST")


# ggplot
ggplot() + geom_sf(data = raleigh_gather, aes(fill = LST)) +
           facet_wrap(~season, ncol=2)

# mapview
mapview(raleigh_sf[,6:17])
```

Plot sf object with ggplot


Day_mean_3month_2015_01_01_ave
Day_mean_3month_2015_04_01_ave
Day_mean_3month_2015_07_01_ave
Day_mean_3month_2015_10_01_ave
36°N
35.9°N
35.8°N
35.7°N
35.6°N
78.9°W
78.8°W
78.7°W
78.6°W
78.5°W
LST
30
25
20
15

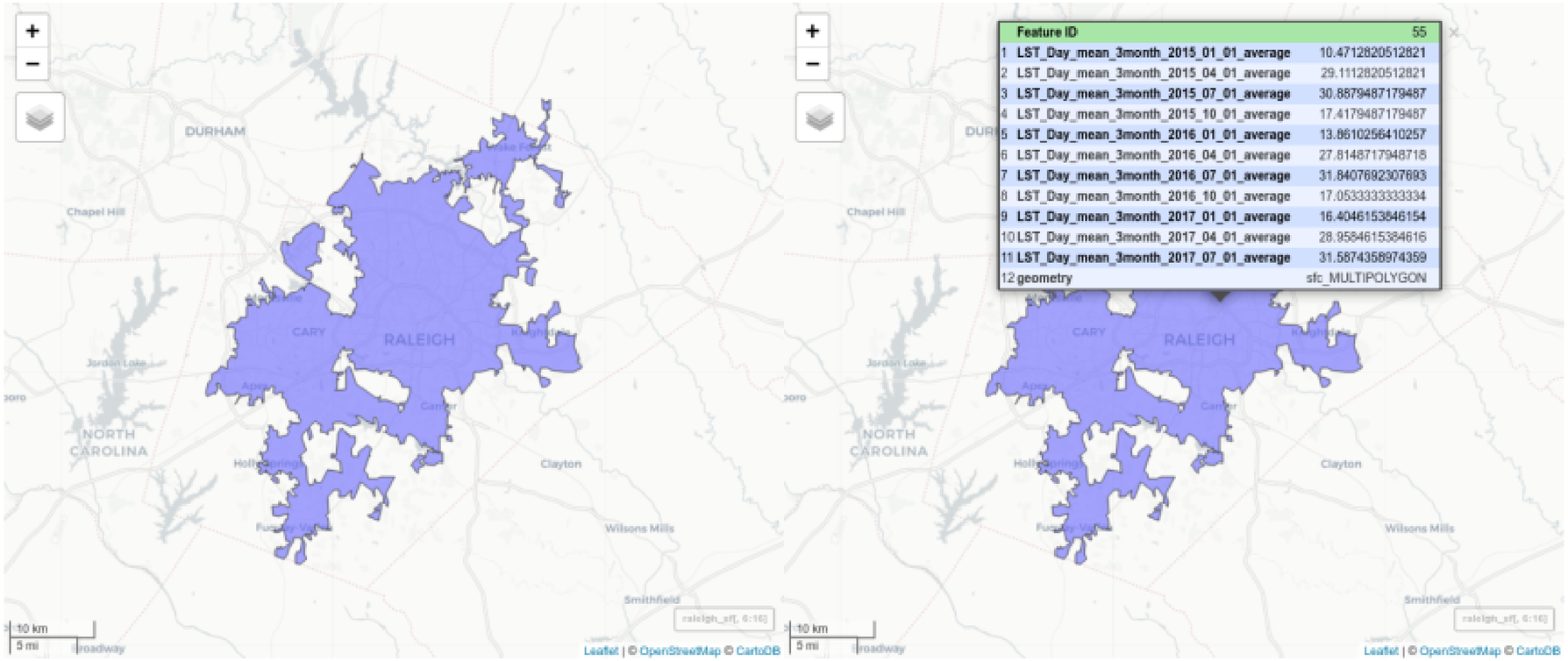
Read and plot Raleigh vector in RStudio

```
##############################################################
# Commands for the TGRASS lecture at GEOSTAT Summer School in Prague
# Author: Veronica Andreo
# Date: July - August, 2018 - Edited October, 2018
##############################################################

## Plot `raleigh_aggr_lst` vector in rstudio

# Call rstudio
rstudio &

# Load libraries
library(rgrass7)
library(sf)
library(dplyr)
library(ggplot2)
library(mapview)
```

Read and plot Raleigh vector in RStudio

```
# sf + ggplot
raleigh_sf <- st_as_sf(raleigh)

# gather the table into season and mean LST (we do only 2015)
raleigh_gather <- raleigh_sf %>%
                  gather(LST_Day_mean_3month_2015_01_01_average,
                         LST_Day_mean_3month_2015_04_01_average,
                         LST_Day_mean_3month_2015_07_01_average,
                         LST_Day_mean_3month_2015_10_01_average,
                         key="season", value="LST")

# ggplot
ggplot() + geom_sf(data = raleigh_gather, aes(fill = LST)) +
           facet_wrap(~season, ncol=2)

# mapview
mapview(raleigh_sf[,6:17])
```

Plot vector over basemap with mapview


DURHAM
Chapel Hill
CARY
RALEIGH
NORTH CAROLINA
Clayton
Wilsons Mills
Smithfield
10 km
5 mi
Leaflet | © OpenStreetMap © CartoDB
Feature ID 55
1 LST_Day_mean_3month_2015_01_01_average 10.4712820512821
2 LST_Day_mean_3month_2015_04_01_average 29.1112820512821
3 LST_Day_mean_3month_2015_07_01_average 30.8879487179487
4 LST_Day_mean_3month_2015_10_01_average 17.4179487179487
5 LST_Day_mean_3month_2016_01_01_average 13.8610256410257
6 LST_Day_mean_3month_2016_04_01_average 27.8148717948718
7 LST_Day_mean_3month_2016_07_01_average 31.8407692307693
8 LST_Day_mean_3month_2016_10_01_average 17.0533333333334
9 LST_Day_mean_3month_2017_01_01_average 16.4046153846154
10 LST_Day_mean_3month_2017_04_01_average 28.9584615384616
11 LST_Day_mean_3month_2017_07_01_average 31.5874358974359
12 geometry sfc_MULTIPOLYGON

QUESTIONS?

Other (very) useful resources

- Temporal data processing wiki
- GRASS GIS and R for time series processing wiki
- GRASS GIS temporal workshop at NCSU
- TGRASS workshop at FOSS4G Europe 2017
- GRASS GIS workshop held in Jena 2018
- GRASS GIS course IRSAE 2018

References

- Gebbert, S., Pebesma, E. (2014). *A temporal GIS for field based environmental modeling*. Environmental Modelling & Software, 53, 1-12. DOI
- Gebbert, S., Pebesma, E. (2017). *The GRASS GIS temporal framework*. International Journal of Geographical Information Science 31, 1273-1292. DOI

Thanks for your attention!!

Move on to:

Exercise: Hands-on to NDVI time series

Presentation powered by

GitPitch