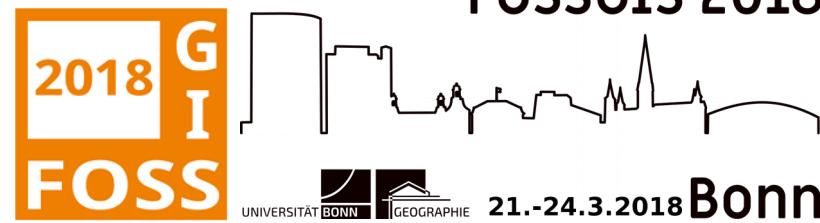


Neues aus dem GRASS GIS Projekt: die Version 7.4.0 steht bereit

Markus Neteler &
GRASS Development Team

grass.osgeo.org
www.mundialis.de

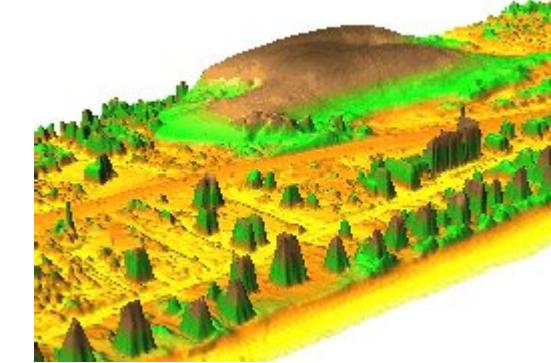
FOSSGIS 2018 – Bonn



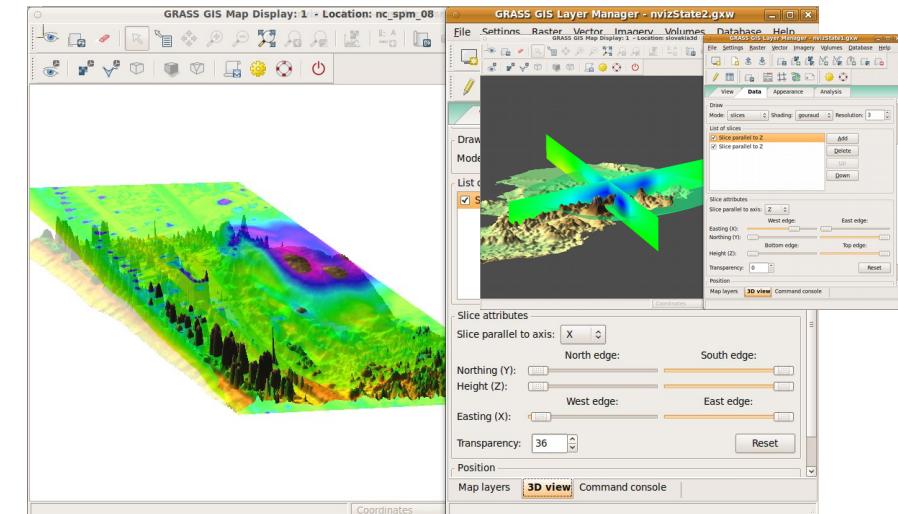


Was ist GRASS GIS?

- GRASS GIS ist eine hybride, modulare GIS-Software
- GRASS = Geographic Resources Analysis Support System
- GNU General Public License – frei verfügbar
- Raster- und topologische Vektordatenfunktionalität
- 3D-Raster-Voxelbearbeitung
- Bildverarbeitung
- Visualisierungsmöglichkeiten
- Portable Software (“alle” Betriebssysteme)
- graphischen Benutzeroberfläche
- sowie Kommandozeile



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

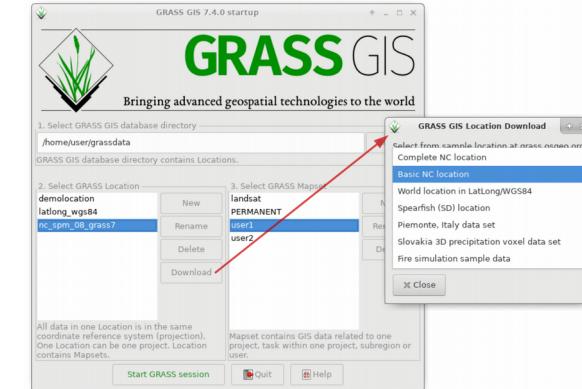




Was ist neu in GRASS GIS 7.4?

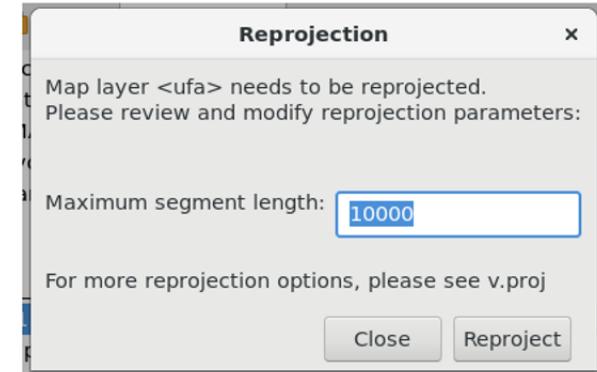
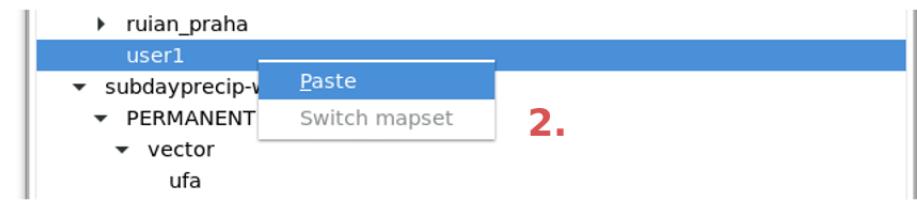
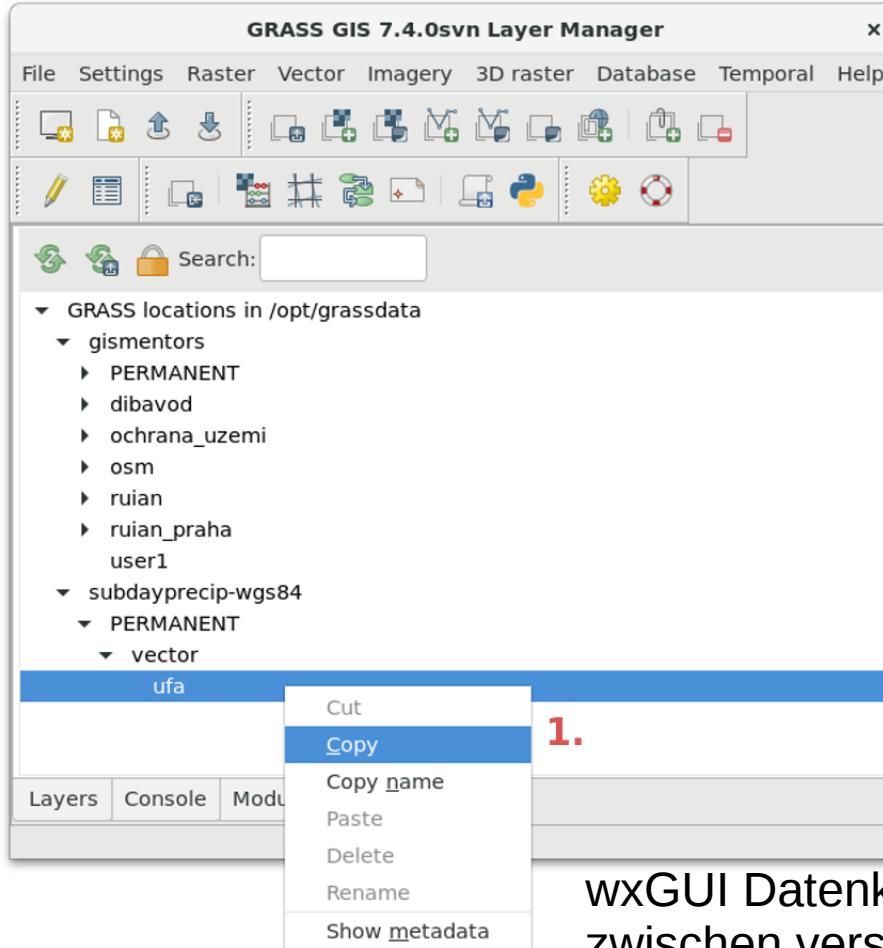
Neue stabile Version GRASS GIS 7.4.0

- Benutzerfreundlichkeit und grafische Benutzeroberfläche verbessert
- Neue “no data” compression
- Unterstützung für globale Daten, die über -180/+180, -90/+90 herausreichen
- Ortho-Rektifikation mit Benutzeroberfläche wurde in GRASS GIS 7 neu implementiert
- Neuer Download-Link für Beispieldaten
- ... über 480 Verbesserungen seit G7.2.0





Data catalog improvements



wxGUI Datenkatalog: Kopieren von Raster- und Vektorkarten zwischen verschiedenen Projekten inklusive Reprojektion



New Orthorectification GUI

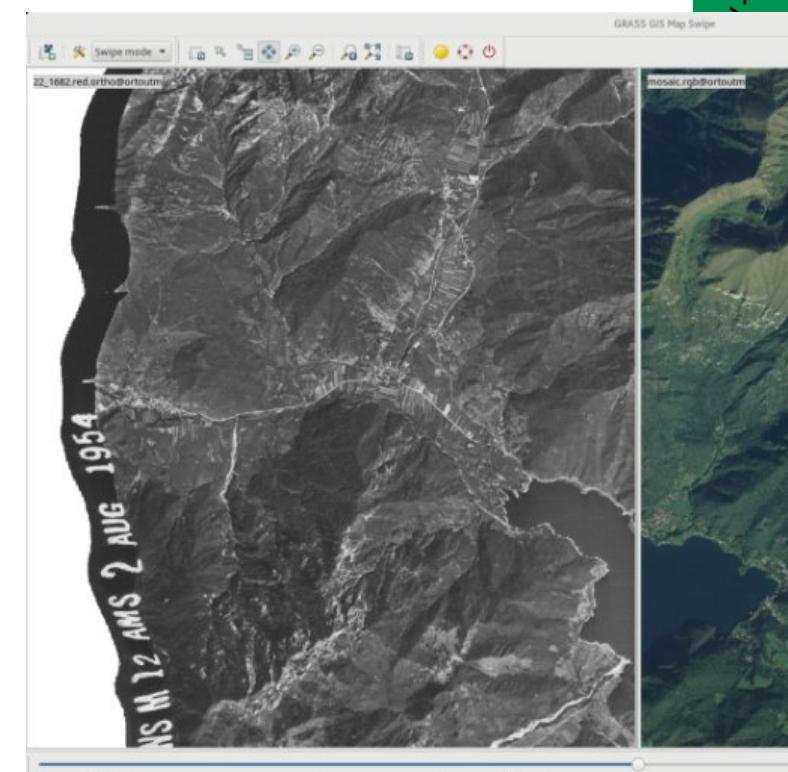
Manage Ground Control Points

GCP List

use	source E	source N	source Z	target E	target N	target Z	Forward error	Backward error	
<input checked="" type="checkbox"/>	1	3433.76399027	4013.92944039	0.0	635890.539036	5082323.73716	700	103893.989338	1206.797055
<input checked="" type="checkbox"/>	2	5663.63017032	3315.20681265	0.0	630696.420894	5083666.03223	750	756671.25929	2593.558967
<input checked="" type="checkbox"/>	3	3484.43309002	4965.997560691	0.0	635880.958794	5080131.73067	750	118984.847243	2147.070500
<input checked="" type="checkbox"/>	4	3519.34793187	4907.6642358	0.0	635757.957044	5080294.5271	720.3481	115606.753734	2096.784943
<input checked="" type="checkbox"/>	5	1980.11435523	4745.01216545	0.0	639373.871778	5080707.57861	500	803733.994864	2206.519653
<input checked="" type="checkbox"/>	6	4006.45255474	2026.76399027	0.0	634353.593597	5086780.06445	850	189157.523637	1216.300242
<input type="checkbox"/>	7	3665.12042706	4302.67206504	0.0	637220.531784	5081740.00077	700	343670.325821	1462.770170

Source Display

Target Display





Graphical Modeller

The screenshot illustrates the GRASS GIS Graphical Modeller interface integrated with the Layer Manager and Map Display windows.

GRASS GIS Graphical Modeler - buffer.gxm* window:

- Shows a workflow diagram with nodes: **(1) v.extract** and **(2) v.buffer**.
- The **Display** button in the bottom right corner of the node **(2) v.buffer** is highlighted with a red box.
- The **Layers** tab is selected in the bottom left.
- The command `d.vector map=dalnice5km` is visible at the bottom left.

GRASS GIS Layer Manager window:

- Shows the **Display 1** tab.
- Contains layers: **staty@ruian** (checked) and **dalnice5km**.

GRASS GIS Map Display: 1 - gismentors/user1 window:

- Shows a map with several vector layers, some of which are shaded in grey.
- The **Coordinates** and **Render** buttons are visible at the bottom.

List of features:

- mark data to be displayed
- print computational time elapsed
- delete intermediate data when computation finished
- export to Python

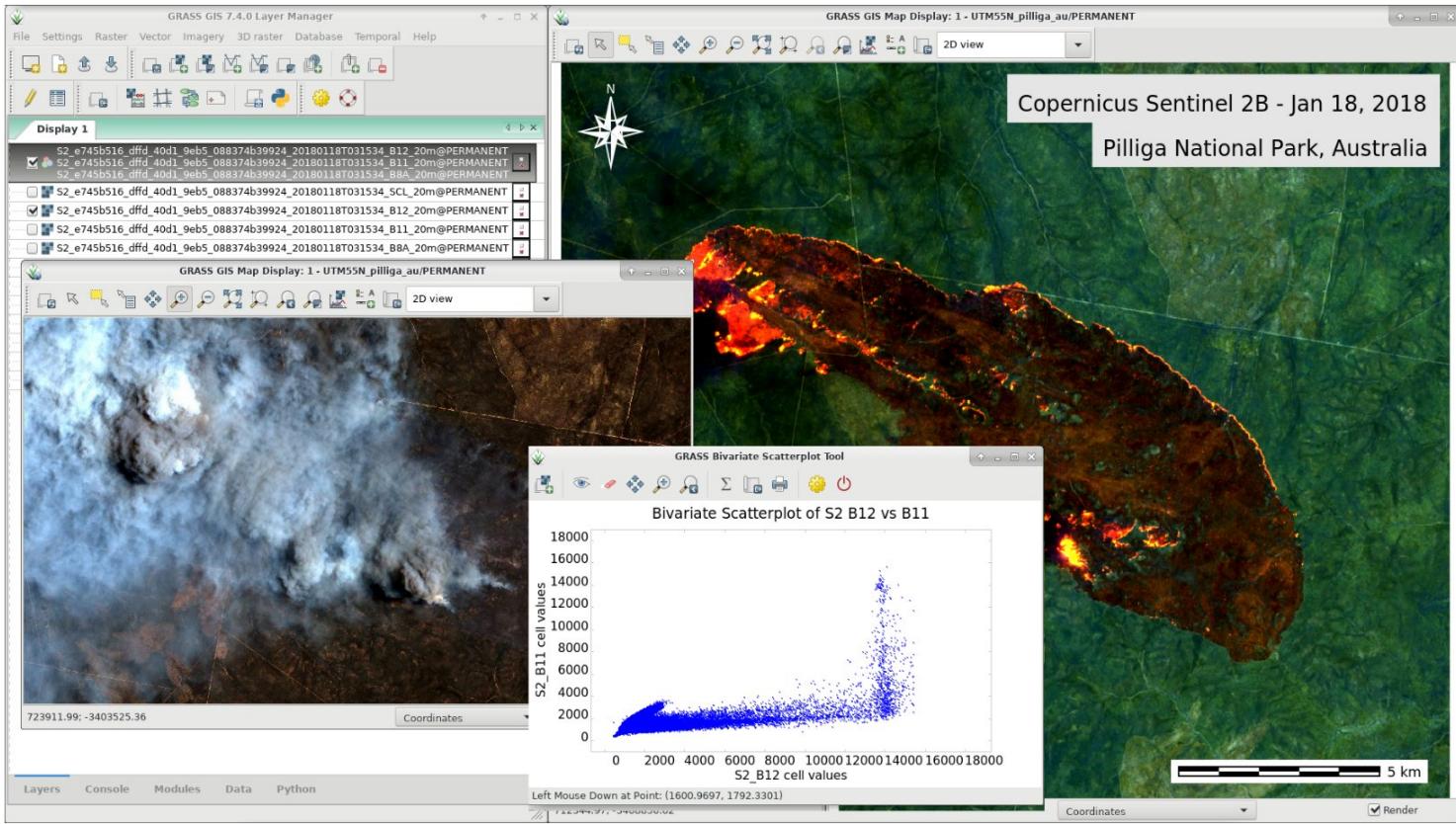


Copernicus Sentinel-2 processing

New addons:

i.sentinel.download and i.sentinel.import

Example:
Wildfire in
Australia





Python Editor

Integrated
Python editor
for rapid
prototyping

Example:
Vector buffer

The screenshot shows the GRASS GIS 7.1 Python Editor interface. On the left, the Layer Manager window displays basic information about the current session, including the Python version (2.7.10), coordinate bounds (north=258102.57210146, south=196653.34855344, east=676816.24830376, west=610869.384849504), and the top and bottom values (0.000000). The main window, titled "GRASS GIS Simple Python Editor", contains the following code:

```
#!/usr/bin/env python

import grass.script as gscript

def main():
    streets = "streets"
    buffer = "streets_buffer"
    gscript.run_command('v.buffer', input=streets, output=buffer,
                        distance=10)

if __name__ == '__main__':
    main()

>>>
```

The right side of the interface shows a map view with a blue line network representing streets. A status bar at the bottom indicates coordinates (639045.50; 229853.11) and a "Rend" checkbox.

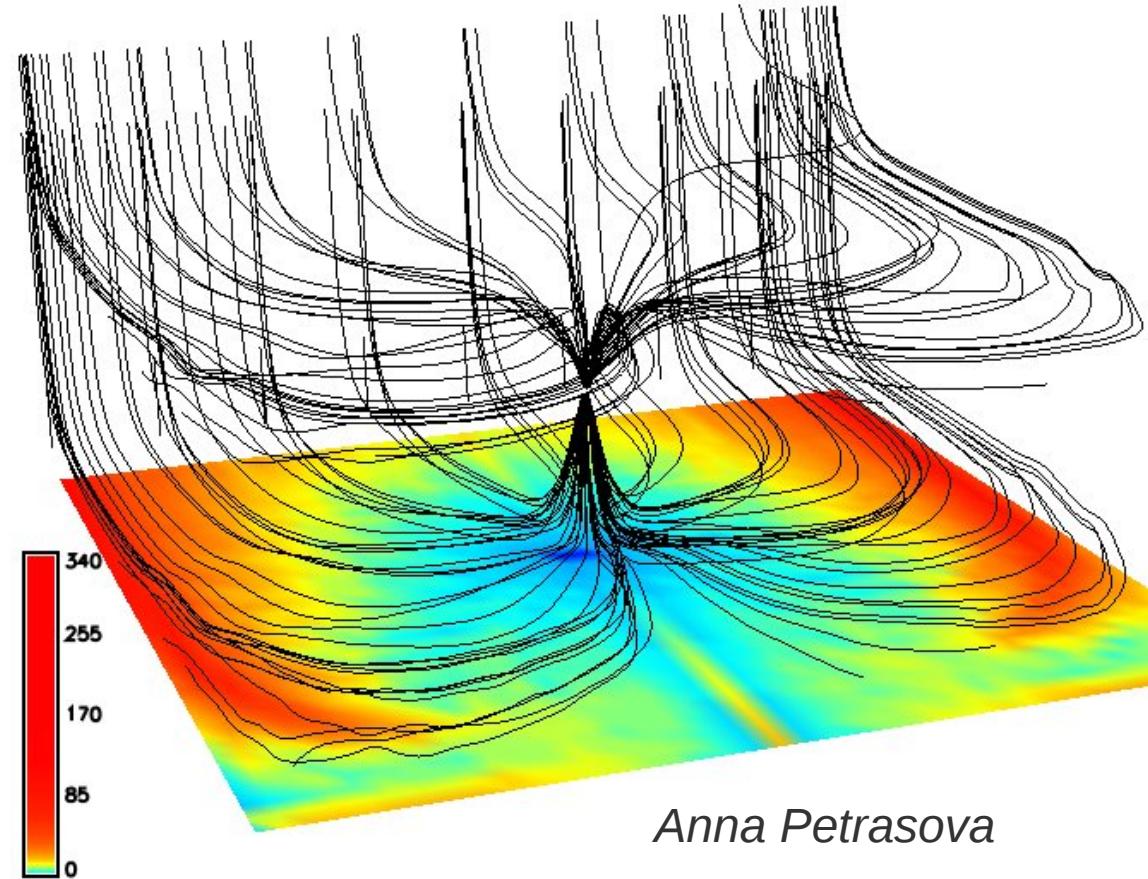
Vaclav Petras



3D raster flowlines

Voxel processing:

r3.flow and r3.gradient to
compute 3D flow lines, 3D
flow accumulation and
related gradients





TGRASS: t.rast.algebra and t.rast3d.algebra: temporal algebra

Compute annual hydro-thermal coefficients (HTC) from daily climate data

$$HTC = \frac{\sum P_{(T > 10^\circ C)}}{\sum T}$$

T := daily temperatures,
P := daily precipitation

T := STRDS of daily temperatures,
'precipitation'
mask, all cells set

~ 60 years of daily data, each pixel in time = virtual meteo station

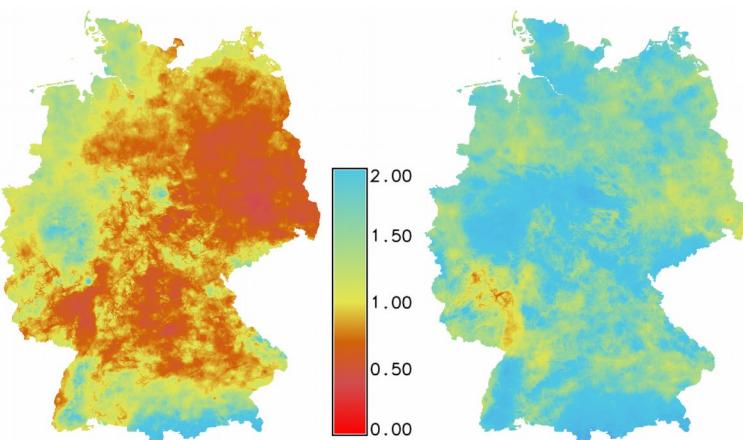


Fig. 6: HTC for 2003 and 2007

Leppelt & Gebbert, EGU 2015

```
t.rast.algebra "HTC = (D {+,contains,1} if(T >= 10, P, 0)) /  
(D {+,contains,1} if(T >= 10, T / 10, 0))"
```

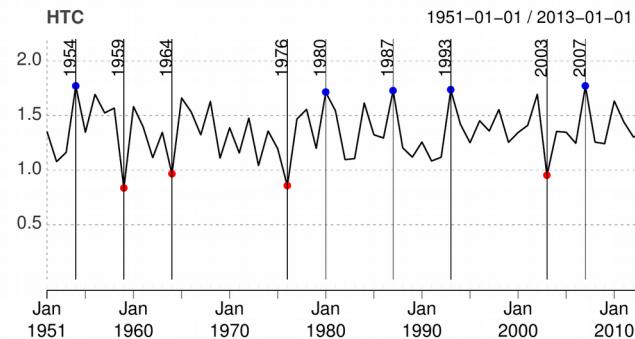


Fig. 7: HTC of extreme events for droughts (HTC < 1) in red and humid years (HTC > 1.7) in blue



GRASS GIS and Python

Using GRASS GIS from “outside” through “grass-session”

pip install grass-session

Finally an easy use of GRASS GIS
as a processing backend in Python!

Combine now with GDAL, OTB, ...

```
#!/usr/bin/env python
# filename: test_session.py

from grass_session import Session
from grass.script import core as gcore

# create a new location from EPSG code (can also be a GeoTIFF or SHP or ... file)
with Session(gisdb="/tmp", location="location",
             create_opts="EPSG:4326"):
    # do something in permanent
    print(gcore.parse_command("g.gisenv", flags="s"))
# {u'GISDBASE': u"/tmp/",  

#  u'LOCATION_NAME': u'epsg3035';,  

#  u'MAPSET': u'PERMANENT';,}

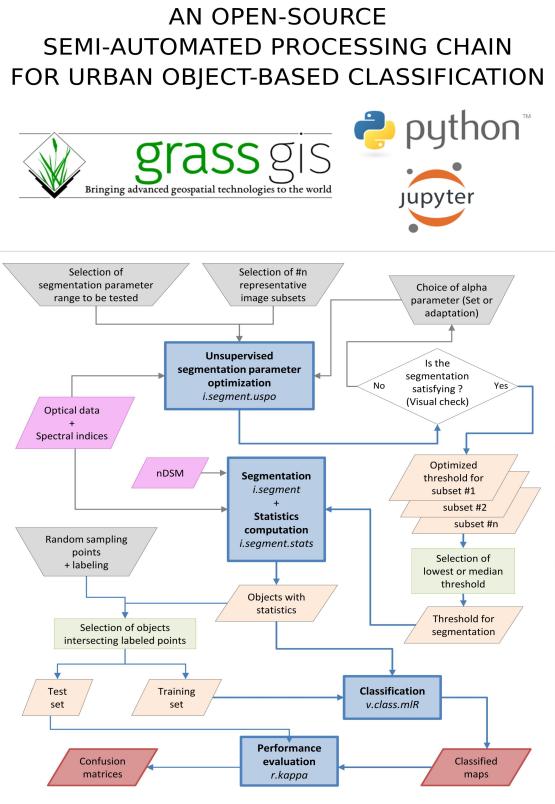
# create a new mapset in an existing location
with Session(gisdb="/tmp", location="location", mapset="test",
             create_opts=""):
    # do something in the test mapset.
    print(gcore.parse_command("g.gisenv", flags="s"))
# {u'GISDBASE': u"/tmp/",  

#  u'LOCATION_NAME': u'epsg3035';,  

#  u'MAPSET': u'test';,}
```



Remote sensing in GRASS GIS : object-based image analysis



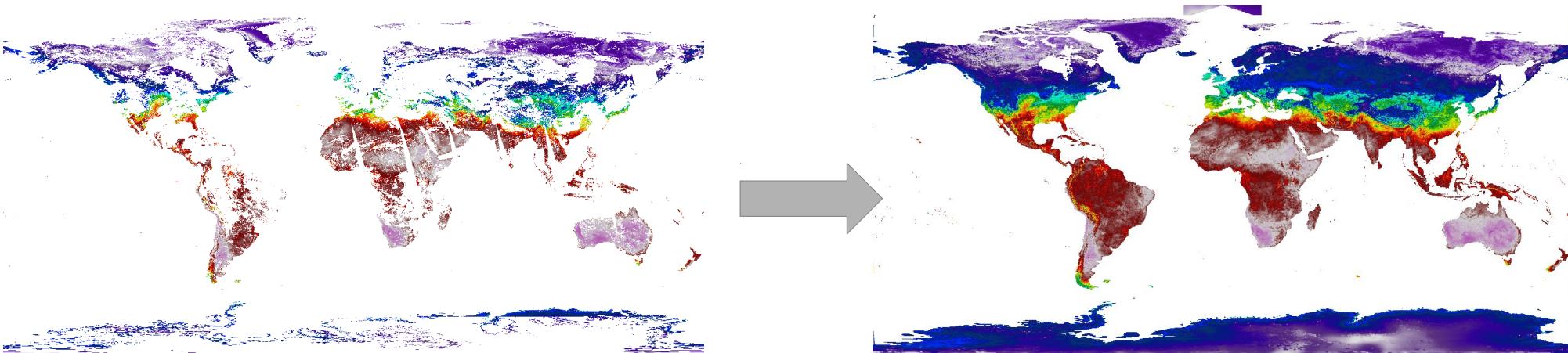
- Complete toolchain from segmentation to classification
- Including
 - unsupervised segmentation parameter optimization
 - high performance object statistics calculation
 - module-level parallelization
- Recently created module for SLIC superpixel creation

Source : <http://dx.doi.org/10.3390/rs9040358>



High-performance computing

MODIS Land Surface Temperature



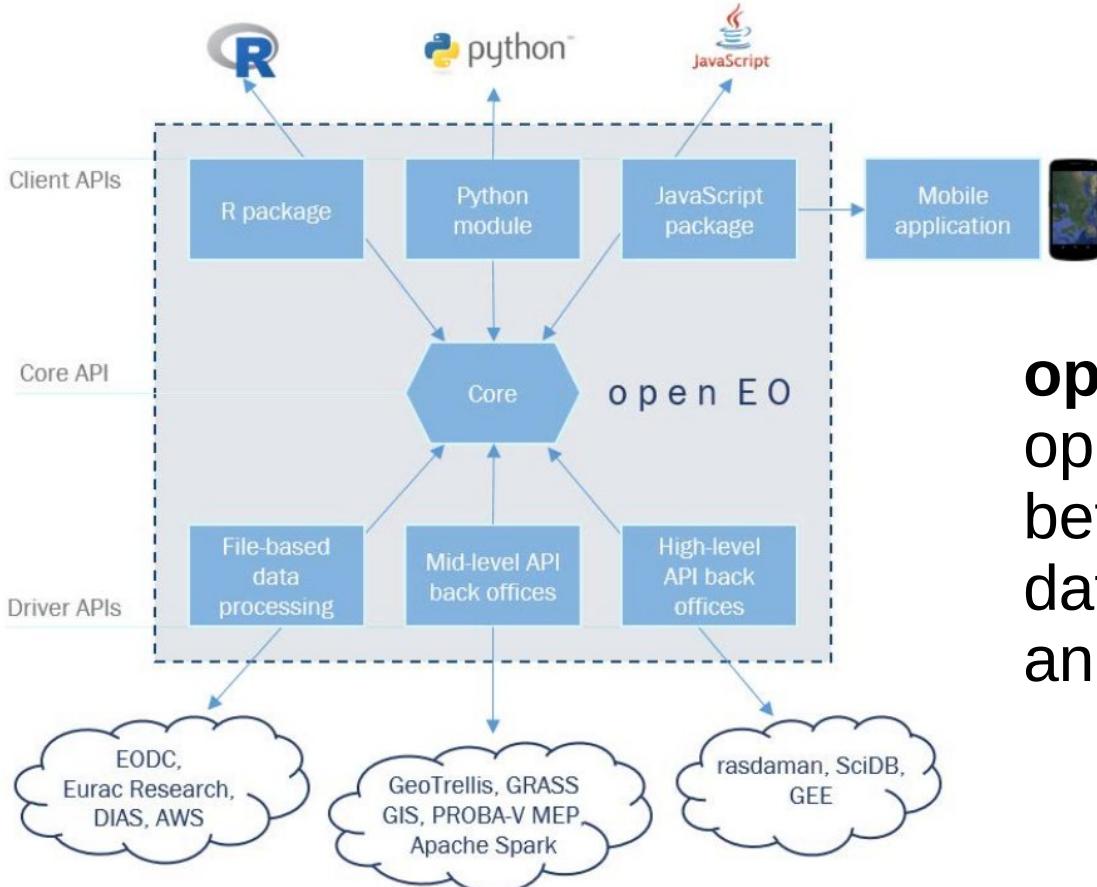
New addons for
temporal + spatial processing for
reconstruction of missing pixels

Data: <https://zenodo.org/record/1135230>



The openEO H2020 EU Project

2017-2020 – <http://www.openeo.org>



openEO - a common, open source interface between Earth Observation data infrastructures and front-end applications



Community activities: Google Code-IN for 13-17 year old pre-university students

https://grasswiki.osgeo.org/wiki/GRASS_GCI_Ideas_2017

3.1 **Install** GRASS GIS on your computer and
download North Carolina dataset

3.2 **Compile** GRASS GIS

3.3 Add examples and/or screenshots to different **manual** pages

3.4 Add **test suites** to different modules

3.5 **Designs**

3.5.1 Splash screen for GRASS GIS GUI start-up

3.5.2 T-shirt for 2018 Code Sprint

3.5.3 Banner for location wizard

3.6 **Blog** entry about GRASS GIS

3.7 **Videos**

3.7.1 How to create a location

3.7.2 Give a talk about GRASS GIS

Community activities: Code Sprint 2018 at Basecamp – Integration



20 March 2018



GRASS GIS Tutorials
@GRASSGIS

Following

GRASS GIS now also supports the new
#PROJ 5 API: lists.osgeo.org/pipermail/grass-devel/2018-March/040000.html
... #osgeo #grassgis

9:53 PM - 20 Mar 2018

4 Retweets 7 Likes



Tweet your reply

Integration with QGIS 3





Community activities: GSoC 2018

Google Summer of Code 2018 – bitte bewerben!

<https://trac.osgeo.org/grass/wiki/GSoC/2018>

OSS-Fuzz - Continuous **Fuzzing** for Open Source Software for GRASS GIS

Implement a series of **image fusion** algorithms in GRASS GIS

Enhance 3D **rendering** capabilities in GRASS GIS

Additional functionality for running GRASS GIS modules in **Jupyter** Notebook

Integration of **PDAL** into GRASS GIS

Benchmarking framework for GRASS GIS

GRASS GIS as a post-processing part of **WebODM**

Additional **GUI** tools for image analysis

Module to create quadtree **tiling**

Tools for generating **unit tests** from examples in the manual

Mapnik rendering engine for GRASS GIS

Generalized GUI code for **Qt-based GUI**

GRASS GIS **3D viewer** NVIZ module independent of the main GUI

Integration of v.profile into **GUI** profiling tool

Add **CMake** build system for GRASS GIS

Add a cloud masking module for **Sentinel** data in GRASS GIS

Full support of **Python 3** in GRASS GIS

Improve GRASS integration in **QGIS 3**

New easy-to-use CLI and **API** for GRASS GIS

Vielen Dank!



grass.osgeo.org