
Processing Large Rasters using Tiling and Parallelization: An R + SAGA GIS + GRASS GIS Tutorial



tom.hengl@envirometrix.net



@tom_hengl



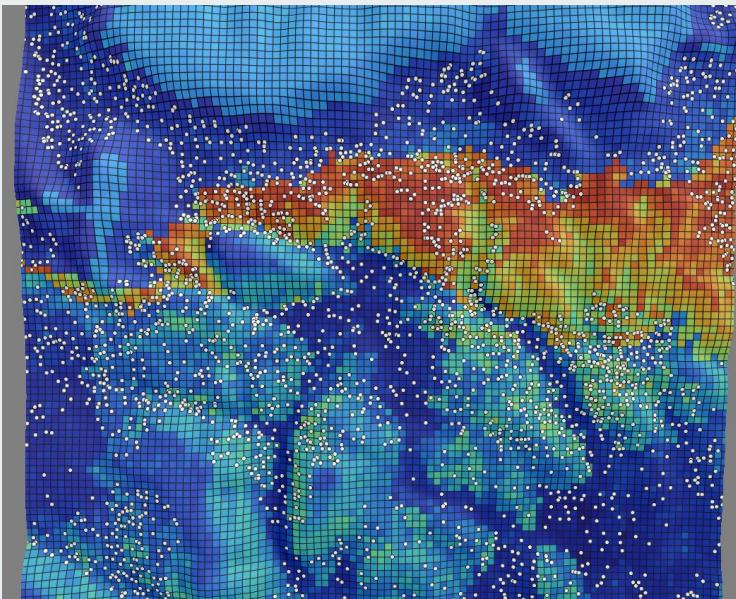
thengl



<http://envirometrix.net>



Overview



1. General concepts
 - a. A systematic approach,
 - b. Parallelization – basic principles,
2. Example 1
 - a. processing global land cover data,
3. Example 2
 - a. processing elevation data and DEM analysis,
4. Conclusions
 - a. What to expect and what not to expect,
 - b. Some work-arounds

3V's of big data

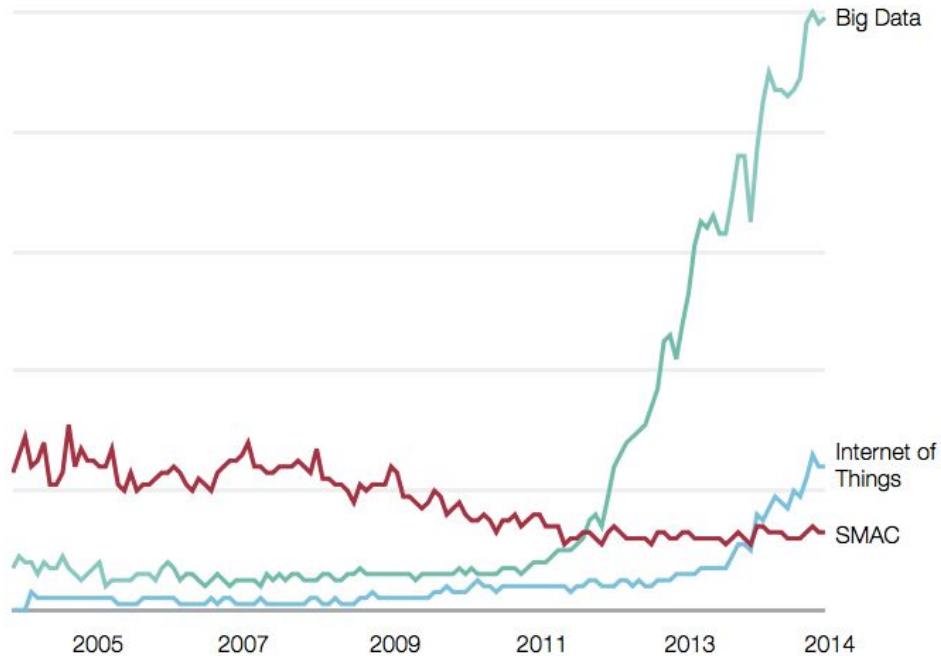


1. Volume (input output data size)
2. Velocity (read / write / compute)
3. Variety (e.g. millions of variables)



3V's of big data

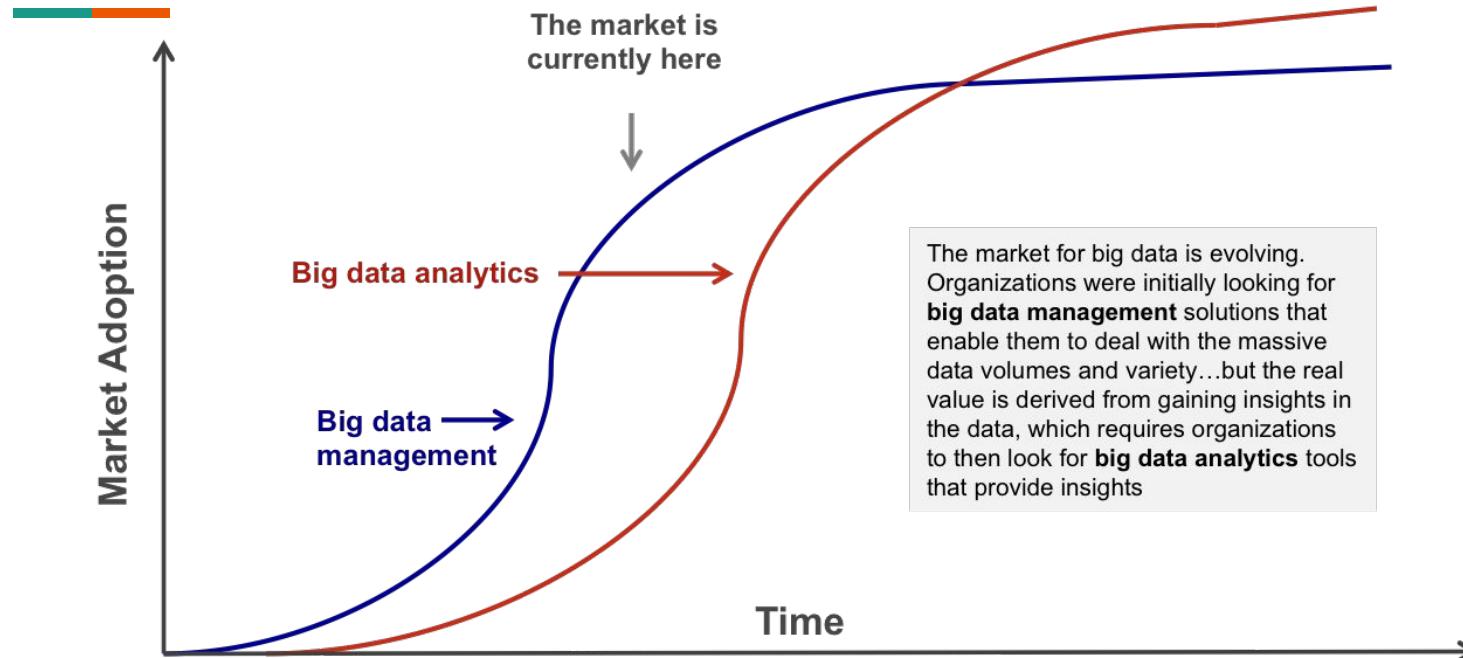
Figure 1: Interest over Time for Specific Tech Trends, 2004-2014, Google Trends



Source: Google Search Trends accessed in December 2014



BData management vs BData analytics



Big Data Management

Primary focus: manage and federate the increasingly massive volume, variety and velocity of data?



Big Data Analytics

Primary focus: Derive business value and provide insight from all the data that is being federated and managed?

First important realization



Programming with "big data" is
really a different type of game
— expect exponentially more
complexity / get ready to boost
your programming skills



advanced search


OPEN ACCESS PEER-REVIEWED

RESEARCH ARTICLE

SoilGrids250m: Global gridded soil information based on machine learning

Tomislav Hengl , Jorge Mendes de Jesus, Gerard B. M. Heuvelink, Maria Ruipez Gonzalez, Milan Kilibarda, Aleksandar Blagotić, Wei Shangguan, Marvin N. Wright, Xiaoyuan Geng, Bernhard Bauer-Marschallinger, Mario Antonio Guevara, Rodrigo Vargas, Robert A. MacMillan, [...], Bas Kempen [\[view all \]](#)

Published: February 16, 2017 • <https://doi.org/10.1371/journal.pone.0169748>

6 Save	74 Citation
17,481 View	16 Share

[Download PDF](#)
[Print](#)
[Share](#)
[Check for updates](#)

ADVERTISEMENT

Article	Authors	Metrics	Comments	Related Content
▼				

Abstract

Introduction
 Methods and materials
 Results
 Discussion
 Conclusions
 Acknowledgments
 Author Contributions
 References

Reader Comments (0)
 Media Coverage (0)
 Figures

Abstract

This paper describes the technical development and accuracy assessment of the most recent and improved version of the SoilGrids system at 250m resolution (June 2016 update). SoilGrids provides global predictions for standard numeric soil properties (organic carbon, bulk density, Cation Exchange Capacity (CEC), pH, soil texture fractions and coarse fragments) at seven standard depths (0, 5, 15, 30, 60, 100 and 200 cm), in addition to predictions of depth to bedrock and distribution of soil classes based on the World Reference Base (WRB) and USDA classification systems (ca. 280 raster layers in total). Predictions were based on ca. 150,000 soil profiles used for training and a stack of 158 remote sensing-based soil covariates (primarily derived from MODIS land products, SRTM DEM derivatives, climatic images and global landform and lithology maps), which were used to fit an ensemble of machine learning methods—random forest and gradient boosting and/or multinomial logistic regression—as implemented in the R packages *ranger*, *xgboost*, *nnet* and *caret*. The results of 10-fold cross-validation show that the ensemble models explain between 56% (coarse fragments) and 83% (pH) of variation with an overall average of 61%. Improvements in the relative accuracy considering the amount of variation explained, in comparison to the previous version of SoilGrids at 1 km spatial resolution, range from 60 to 230%. Improvements can be attributed to: (1) the use of machine learning instead of linear regression, (2) to considerable investments in preparing finer resolution covariate layers and (3) to insertion of additional soil profiles. Further development of SoilGrids could include refinement of methods to incorporate input uncertainties and derivation of posterior probability distributions (per pixel), and further automation of spatial modeling so that soil maps can be generated for potentially hundreds of soil variables. Another area of future research is the development of methods for multiscale merging of SoilGrids predictions with local and/or national gridded soil products (e.g. up to 50 m spatial resolution) so that increasingly more accurate, complete and consistent global soil information can be produced. SoilGrids are available under the Open Data Base License.

Figures

Subject Areas

- Shannon index
- Forecasting
- Machine learning
- Trees
- Agricultural soil sci...
- Remote sensing
- Glaciers
- Soil ecology

Archived Tweets



22 Feb 2017

TimC_Tech @tech_time

[Nutrient Cycling in Agroecosystems](#)September 2017, Volume 109, [Issue 1](#), pp 77–102 | [Cite as](#)

Soil nutrient maps of Sub-Saharan Africa: assessment of soil nutrient content at 250 m spatial resolution using machine learning

[Authors](#)[Authors and affiliations](#)

Tomislav Hengl , Johan G. B. Leenaars, Keith D. Shepherd, Markus G. Walsh, Gerard B. M. Heuvelink, Tekalign Mamo, Helina Tilahun, Ezra Berkhouwt, Matthew Cooper, Eric Fegraus, Ichsan Wheeler, Nketia A. Kwabena

[Open Access](#) | Original Article

First Online: 02 August 2017

2

3.2k

3

Shares

Downloads

Citations

[Download PDF](#)[Cite article](#)[Share article](#)[Article](#)[Abstract](#)[Introduction](#)[Materials and methods](#)[Results](#)[Discussion](#)[Conclusions](#)[Footnotes](#)[Notes](#)[References](#)[Copyright information](#)[About this article](#)

Abstract

Spatial predictions of soil macro and micro-nutrient content across Sub-Saharan Africa at 250 m spatial resolution and for 0–30 cm depth interval are presented. Predictions were produced for 15 target nutrients: organic carbon (C) and total (organic) nitrogen (N), total phosphorus (P), and extractable–phosphorus (P), potassium (K), calcium (Ca), magnesium (Mg), sulfur (S), sodium (Na), iron (Fe), manganese (Mn), zinc (Zn), copper (Cu), aluminum (Al) and boron (B). Model training was performed using soil samples from ca. 59,000 locations (a compilation of soil samples from the AfSIS, EthioSIS, One Acre Fund, VitalSigns and legacy soil data) and an extensive stack of remote sensing covariates in addition to landform, lithologic and land cover maps. An ensemble model was then created for each nutrient from two machine learning algorithms—random forest and gradient boosting, as implemented in R packages ranger and xgboost—and then used to generate predictions in a fully-optimized computing system. Cross-validation revealed that apart from S, P and B, significant models can be produced for most targeted nutrients (R-square between 40–85%). Further comparison with OFRA field trial database shows that soil nutrients are indeed critical for agricultural development, with Mn, Zn, Al, B and Na, appearing as the most important nutrients for predicting crop yield. A limiting factor for mapping nutrients using the existing point data in Africa appears to be (1) the high spatial clustering of sampling locations, and (2) missing more detailed information about soil depth, texture, and mineral composition, which is required for accurate predictions.



JOURNAL OF GEOPHYSICAL RESEARCH Atmospheres

AN AGU JOURNAL

Research Article | Open Access

Spatio-temporal interpolation of daily temperatures for global land areas at 1 km resolution

Milan Kilibarda , Tomislav Hengl, Gerard B. M. Heuvelink, Benedikt Gräler, Edzer Pebesma, Melita Perćec Tadić, Branislav Bajat

First published: 03 February 2014 | <https://doi.org/10.1002/2013JD020803> | Cited by: 34

SECTIONS

PDF TOOLS SHARE

Abstract

Combined Global Surface Summary of Day and European Climate Assessment and Dataset daily meteorological data sets (around 9000 stations) were used to build spatio-temporal geostatistical models and predict daily air temperature at ground resolution of 1 km for the global land mass. Predictions in space and time were made for the mean, maximum, and minimum temperatures using spatio-temporal regression-kriging with a time series of Moderate Resolution Imaging Spectroradiometer (MODIS) 8 day images, topographic layers (digital elevation model and topographic wetness index), and a geometric temperature trend as covariates. The accuracy of predicting daily temperatures was assessed using leave-one-out cross validation. To account for geographical point clustering of station data and get a more representative cross-validation accuracy, predicted values were aggregated to blocks of land of size 500×500 km. Results show that the average accuracy for predicting mean, maximum, and minimum daily temperatures is root-mean-square error (RMSE) =±2°C for areas densely covered with stations and between ±2°C and ±4°C for areas with lower station density. The lowest prediction accuracy was observed at high altitudes (>1000 m) and in Antarctica with an RMSE around 6°C. The model and predictions were built for the year 2011 only, but the same methodology could be extended for the whole range of the



Volume 119, Issue 5

16 March 2014

Pages 2294-2313

Figures References Related Information

Metrics

Citations: 34



2

Details

©2014. The Authors.

This is an open access article under the terms of the Creative Commons Attribution-NonCommercial-NoDerivs License, which permits use and distribution in any medium, provided the original work is properly cited, the use is non-commercial and no modifications or adaptations are made.

Keywords

spatio-temporal kriging

spatio-temporal interpolation

daily air temperature MODIS LST

Publication History

Issue Online:

08 April 2014

LETTER • OPEN ACCESS • FEATURED ARTICLE

A global map of mangrove forest soil carbon at 30 m spatial resolution

Jonathan Sanderman^{1,21} , Tomislav Hengl², Greg Fiske¹, Kylen Solvik¹, Maria Fernanda Adame³, Lisa Benson^{5,6}, Jacob J Bukoski⁷, Paul Carnell⁸, Miguel Cifuentes-Jara⁹, Daniel Donato¹⁹, Clare Duncan^{4,8}, Ebrahim M Eid^{10,20}, Philine zu Ermgassen^{17,18}, Carolyn J Ewers Lewis⁸, Peter I Macreadie⁸ , Leah Glass⁵, Selena Gress¹¹, Sunny L Jardine¹², Trevor G Jones^{5,13}, Eugène Ndemem Nsombo¹⁴, Md Mizanur Rahman¹⁵, Christian J Sanders¹⁶, Mark Spalding¹⁷ and Emily Landis¹⁷  Hide full author list

Published 30 April 2018 • © 2018 The Author(s). Published by IOP Publishing Ltd

Environmental Research Letters, Volume 13, Number 5

Focus on The Role of Forests and Soils in Meeting Climate Change Mitigation Goals



Figures ▾ References ▾

Article information

Abstract

With the growing recognition that effective action on climate change will require a combination of emissions reductions and carbon sequestration, protecting, enhancing and restoring natural carbon sinks have become political priorities. Mangrove forests are considered some of the most carbon-dense ecosystems in the world with most of the carbon stored in the soil. In order for mangrove forests to be included in climate mitigation efforts, knowledge of the spatial distribution of mangrove soil carbon stocks are critical. Current global estimates do not capture enough of the finer scale variability that would be required to inform local decisions on siting protection and restoration projects. To close this knowledge gap, we have compiled a large georeferenced database of mangrove soil carbon measurements and developed a novel machine-learning based statistical model of the distribution of carbon density using spatially comprehensive data at a 30 m resolution. This model, which included a prior estimate of soil carbon from the global SoilGrids 250 m model, was able to capture 63% of the vertical and horizontal variability in soil organic carbon density (RMSE of 10.9 kg m⁻³). Of the local variables, total suspended sediment load and Landsat imagery were the most important variable explaining soil carbon density. Projecting this model across the global mangrove forest distribution for the year 2000 yielded an estimate of 6.4 Pg C for the top meter of soil with an

4852 Total downloads



Turn on MathJax

Share this article



Abstract

1. Introduction

2. Methods

3. Results

4. Discussion

5. Conclusions

Data availability

Acknowledgments

References

Related content

JOURNAL ARTICLES

A global predictive model of carbon in mangrove soils

Organic carbon sequestration and storage in vegetated coastal habitats along the western coast of the Arabian Gulf

Estimating mangrove aboveground biomass from airborne LiDAR data: a case study from the Zambezi River delta

Measurement and monitoring needs, capabilities and potential for addressing reduced emissions from deforestation and forest degradation under REDD+

Soil greenhouse gas emissions reduce the contribution of mangrove plants to the atmospheric cooling effect

Effects of contemporary land-use and land-cover change on the carbon balance of terrestrial ecosystems in the United States

brightrecruits.com jobs

Short Term Scientist Assistant at the Postdoctoral Level in Computational Materials Theory

Ludwig-Maximilians-Universitaet Muenchen

PhD Call

University of Vienna

Preprint

View 12 tweets

NOT PEER-REVIEWED

"PeerJ Preprints" is a venue for early communication or feedback before peer review. Data may be preliminary.
Learn more about preprints or browse peer-reviewed articles instead.

Global mapping of potential natural vegetation: an assessment of Machine Learning algorithms for estimating land potential

[Research article](#) [Biogeography](#) [Computational Biology](#) [Plant Science](#) [Data Mining and Machine Learning](#)[Spatial and Geographic Information Science](#)

Tomislav Hengl¹, Markus G Walsh², Jonathan Sanderman³, Ichsan Wheeler⁴,
Sandy P Harrison⁵, Iain C Prentice⁶

March 30, 2018

› Author and article information

▼ Abstract

Potential Natural Vegetation (PNV) is the vegetation cover in equilibrium with climate, that would exist at a given location non-impacted by human activities. PNV is useful for raising public awareness about land degradation and for estimating land potential. This paper presents results of assessing Machine Learning Algorithms (MLA) for operational mapping of Potential Natural Vegetation (PNV). The following MLA were considered: neural networks (*nnet* package), random forest (*ranger*), gradient boosting (*gmb*), K-nearest neighborhood (*class*) and cubist. Three case studies were considered: (1) global distribution of biomes based on the BIOME 6000 data set (8057 modern pollen-based site reconstructions), (2) distribution of forest tree taxa in Europe based on detailed occurrence records (1,546,435 ground observations), and (3) global monthly Fraction of Absorbed Photosynthetically Active Radiation (FAPAR) values (30,301 randomly-sampled points). A stack of 160 global maps representing biophysical



Article to-dos

Hi Tomislav, complete these sharing tasks to maximise engagement with your work.

- ✓ Added to Wikipedia [Get link](#)
- ✓ Added to Instit Repo [Get link](#)
- ✓ Wrote blog post [Get link](#)
- ✓ Emailed colleagues [Email](#)
- ✓ Shared on Facebook [Post](#)
- ✓ Shared on Twitter [Tweet](#)

[View all impact tasks ▶](#)

Enter your institution

To find colleagues at PeerJ

 Enter to search[Download ▾](#)[Content Alert NEW](#)

↗ Tools & Info

[Citations in Google Scholar](#)[Add feedback](#)[Add to list](#)



Three options:

1. Use software that handles large data
(internal parallelization)
2. Make your own functions
3. Use infrastructure optimized for large data

Learning parallelization is rewarding!



It is of course better if you can
learn how to solve things by
making your own functions

What can go wrong



1. You optimize almost everything, but then one bottleneck leaves you hanging.
2. You do not plan carefully and examine read/write times, memory handling, computing etc — and which results in a serious loss of time.
3. You leave computer computing for a very excessive amount of time without really knowing the progress.



once you get a bit better at it, it
starts looking something like
this...

Applications Terminal Don Jan 25, 16:13 9,2°C hr Gc Ge Fr N M Op n Q Th Ql M Gl M Ae Sc W La 20 th R bo R Un Th Th Po I An Int R Tir va R RS G va Temislav

Not secure | 145.239.142.3:8787

Bookmarks Amazon RStudio RStudio VideoSta

R File Edit Code Plots Session Build Go to file/function

LDN_SOC_change.R PNV_mapping.R

/Forest_TYPE_MAP_2006_250m.sdat -of \"SAGA\"

```

479 tile.pol.eu = spTransform(tile.pol, CRS("+pr
+ellps=GRS80 +units=m +no_defs"))
480 writeOGR(tile.pol.eu, "/data/PNV/Data/EU_for
")
481 system(paste0('saga_cmd -c=64 shapes_grid 2
m.sgrd" -POLYGONS="/data/PNV/Data/EU_fore
/data/EU_forest/ov_tiles_eu.shp"'))
482 ov_tiles_eu = readOGR("/data/PNV/Data/EU_for
")
483 str(ov_tiles_eu$data)
484 summary(sels.eu <- (ov_tiles_eu$Forest_TYPE
pr.dirs.eu = paste0("T", ov_tiles_eu$ID[whic
486 ## 995 tiles only
487
488 ## Takes 30 mins without errors
489 m.eu.trees = readRDS.gz("m.eu.trees.rds")
490 col.legend.for = data.frame(Group=eu.sp$sp,
cl <- parallel::makeCluster(8, type="FORK")
492 x <- parallel::parLapply(cl, pr.dirs.eu, func
.tbl, col.legend=col.legend.for, varn="EUfor
493 parallel::stopCluster(cl)
494
495 EU forest predict :
```

Copying nodata values from source /data/GEOG/OCSTHA
0.10...20.30...40.50...60.70...80.90...10
> save.image()

Restarting R session...

Microsoft R Open 3.4.1
The enhanced R distribution from Microsoft
Microsoft packages Copyright (C) 2017 Microsoft Cor

Using the Intel MKL for parallel mathematical compu

Default CRAN mirror snapshot taken on 2017-09-01.
See: <https://mran.microsoft.com/>.

```

> setwd("/data/PNV/R_code")
> load(".RData")
```

File Edit View Search Terminal Help

ubuntu@ip-172-31-42-90: ~

1 [100.0%] 17 [100.0%] 33 [100.0%] 49 [100.0%
2 [100.0%] 18 [100.0%] 34 [100.0%] 50 [100.0%
3 [100.0%] 19 [100.0%] 35 [100.0%] 51 [100.0%
4 [100.0%] 20 [100.0%] 36 [100.0%] 52 [100.0%
5 [100.0%] 21 [100.0%] 37 [100.0%] 53 [100.0%
6 [100.0%] 22 [100.0%] 38 [100.0%] 54 [100.0%
7 [100.0%] 23 [100.0%] 39 [100.0%] 55 [100.0%
8 [100.0%] 24 [100.0%] 40 [100.0%] 56 [100.0%
9 [100.0%] 25 [100.0%] 41 [100.0%] 57 [100.0%
10 [100.0%] 26 [100.0%] 42 [100.0%] 58 [100.0%
11 [100.0%] 27 [100.0%] 43 [100.0%] 59 [100.0%
12 [100.0%] 28 [100.0%] 44 [100.0%] 60 [100.0%
13 [100.0%] 29 [100.0%] 45 [100.0%] 61 [100.0%
14 [100.0%] 30 [100.0%] 46 [100.0%] 62 [100.0%
15 [100.0%] 31 [100.0%] 47 [100.0%] 63 [100.0%
16 [100.0%] 32 [100.0%] 48 [100.0%] 64 [100.0%

Mem[31.7G/252G Tasks: 98, 53 thr; 65 running
Swp[0K/0K Load average: 63.67 39.60 18.64
Uptime: 06:39:28

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
58699	ubuntu	20	0	1138M	520M	43160	R	100.	0.2	4:44.17	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58219	ubuntu	20	0	1135M	518M	43056	R	100.	0.2	4:43.98	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58209	ubuntu	20	0	1135M	517M	42796	R	100.	0.2	4:44.36	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58359	ubuntu	20	0	1134M	517M	43380	R	100.	0.2	4:44.43	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58349	ubuntu	20	0	1134M	517M	42972	R	100.	0.2	4:44.29	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58609	ubuntu	20	0	1137M	521M	43108	R	100.	0.2	4:44.50	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58519	ubuntu	20	0	1128M	511M	43052	R	100.	0.2	4:44.06	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58729	ubuntu	20	0	1177M	559M	43116	R	100.	0.2	4:44.43	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58649	ubuntu	20	0	1138M	521M	43100	R	100.	0.2	4:44.41	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58709	ubuntu	20	0	1137M	520M	43444	R	100.	0.2	4:43.96	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58339	ubuntu	20	0	1135M	517M	42740	R	100.	0.2	4:44.40	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58129	ubuntu	20	0	1135M	518M	43312	R	100.	0.2	4:43.15	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58149	ubuntu	20	0	1134M	519M	42696	R	99.7	0.2	4:43.66	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58459	ubuntu	20	0	1131M	516M	42808	R	99.7	0.2	4:44.25	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no
58289	ubuntu	20	0	1109M	492M	43404	R	99.7	0.2	4:44.05	/opt/microsoft/ropen/3.4.2/lib64/R/bin/exec/R --slave --no

F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice -F8Nice +F9Kill F10Quit

Arguments

cl a cluster object, created by this package or by package [snow](#). If NULL, use the registered default cluster.

Some lessons I've learned:

- (a) Plan carefully,
- (b) Invest in hardware / storage,
- (c) Use appropriate tiling (linear, hierarchical),
- (d) Implement full parallelization (optimize code so it uses hardware to maximum)
- (e) Make "scalable" code (it would work with N elements running on P cores),



Google Earth Engine

The image shows a presentation slide titled "Google Earth Engine is Evil". The slide contains a bulleted list of points. At the top, there is a note: "Very personal opinion, don't quote my employer nor my co-authors!". The list includes:

- ▶ What is GEE?
 - ▶ Working on data you don't have.
 - ▶ Using processing chains you don't own and can't inspect (it's an API)
 - ▶ Running on somebody else's computer
 - ▶ So that you can't reproduce your work somewhere else.
 - ▶ It's not your washing machine. It's a laundromat. And you are not sure of getting your clothes back!
- ▶ This is not good for science
 - ▶ Private corporation agendas, there is no free lunch!
 - ▶ Science reproducibility is fragilised.
- ▶ Terms and conditions may apply
 - ▶ How much will you have to pay once all your work is inside GEE's silo?
 - ▶ Remember Google Wave, Google Reader, Google Code and Picasa?
- ▶ So what do we do?
 - ▶ Let's ask ESA, EC, our national agencies to join efforts to implement the infrastructure where

At the bottom of the slide, it says "Living Planet Symposium 2014".

HUB Remote Sensing @HumboldtRemSens · 21h
J.Inglada @ #LivingPlanet16: "GoogleEarthEngine is not a washing machine, it's rather a laundromat" Good 4 science?

12 17 ...



Software: fast reading of data

```
> ## CSV files:  
> df <- datatable::fread("test.csv")  
> ## shapefiles:  
> f = system.file("shape/nc.shp", package="sf")  
> pnt <- sf::st_read(f)  
> ## rasters:  
> r <- raster::stack(list.files)
```

read raster data in parallel

August 18, 2012

By Tim Salabim

 Like 2  Share  Tweet  Share

(This article was first published on [metvurst](#), and kindly contributed to R-bloggers)

Use library(parallel) to read raster data in parallel fashion

Recently, I have been doing some analysis for a project I am involved in. In particular, I was interested what role pacific sea surface temperatures play with regard to rainfall in East Africa. I spare you the details as I am currently writing all this up into a paper which you can have a look at once published.

For this analysis, however, I am processing quite an amount of raster files. This led me to investigate the possibilities of the `parallel` package to speed up the process.

Here's a quick example on how to read in raster data (in this case 200 global sea surface temperature files of $1^\circ \times 1^\circ$ degree resolution) using `parallel`

First, lets do it the conventional way and see how long that takes

```
system.time({  
  library(raster)  
  library(rgdal)
```

This code will work just fine and will need approximately 2min to execute for smaller rasters (4000x4000px). However, imagine you have 100 rasters you would like to process. Here using multiple cores will save you a lot of time. And this is how you do it:

Example with parallelisation

```
library(raster)  
library(doParallel) #Foreach Parallel Adaptor  
library(foreach)   #Provides foreach looping construct  
  
#Define how many cores you want to use  
UseCores <- detectCores() -1  
  
#Register CoreCluster  
cl      <- makeCluster(UseCores)  
registerDoParallel(cl)  
  
path      <- "Downloads/Stacks/"  
stack_list <- list.files(path, pattern=".tif$", full.names=T)  
  
#Use foreach loop and %dopar% command  
foreach(i=1:length(stack_list)) %dopar% {  
  library(raster)  
  
  img  <- stack(stack_list[i])  
  ndvi <- (img[[4]]-img[[3]]) / (img[[4]]+img[[3]])  
  
  outname <- sub(pattern     = ".tif",  
                  replacement = "_ndvi.tif",  
                  x            = stack_list[i])  
  
  writeRaster(ndvi,  
             filename = outname,  
             overwrite = T)  
}
```





```
> ## gdal warp in parallel:  
> system(paste0('gdalwarp AW3D30_30m.vrt AW3D30_dem_100m.tif  
-co \"BIGTIFF=YES\" -wm 2000 -co \"COMPRESS=DEFLATE\" -multi  
-wo \"NUM_THREADS=ALL_CPUS\"'))
```

Reading large geotif tile by tile



```
> fn = system.file("pictures/SP27GTIF.TIF", package =
"rgdal")
> obj <- rgdal::GDALinfo(fn)
> tiles <- GSIF::getSpatialTiles(obj, block.x=5000,
return.SpatialPolygons = FALSE)
> tiles.pol <- GSIF::getSpatialTiles(obj, block.x=5000,
return.SpatialPolygons = TRUE)
> tile.pol <- SpatialPolygonsDataFrame(tiles.pol, tiles)
> plot(raster(fn), col=bpy.colors(20))
> lines(tile.pol, lwd=2)
```



Reading and writing RDS files in parallel

```
> saveRDS.gz <- function(object,file,threads=parallel::detectCores()) {  
  con <- pipe(paste0("pigz -p", threads, " > ", file), "wb")  
  saveRDS(object, file = con)  
  close(con)  
}  
  
> readRDS.gz <- function(file,threads=parallel::detectCores()) {  
  con <- pipe(paste0("pigz -d -c -p", threads, " ", file))  
  object <- readRDS(file = con)  
  close(con)  
  return(object)  
}
```

[Tools A-Z](#)[Contents](#)

Library Grids

Tools for spatial and geostatistical analyses.

- Author: O.Conrad, V.Wichmann (c) 2002-16
- Version: 1.0
- Menu: Spatial and Geostatistics|Grids

Tools

- Categorical Coincidence
- Directional Statistics for Single Grid
- Fast Representativeness
- Focal PCA on a Grid
- Global Moran's I for Grids
- Inverse Principle Components Rotation
- Longitudinal Grid Statistics
- Meridional Grid Statistics
- Multi-Band Variation
- Principle Components Analysis
- Radius of Variance (Grid)
- Representativeness (Grid)
- Residual Analysis (Grid)
- Save Grid Statistics to Table
- Statistics for Grids
- Zonal Grid Statistics



SAGA GIS automatically uses all cores!

```
saga_cmd [-h, --help]
saga_cmd [-v, --version]
saga_cmd [-b, --batch]
saga_cmd [-d, --docs]
saga_cmd [-f, --flags] [=qrsilpxo] [-s, --story] [=#] [-c, --cores] [=#]
    <LIBRARY> <MODULE> <OPTIONS>
saga_cmd [-f, --flags] [=qrsilpxo] [-s, --story] [=#] [-c, --cores] [=#]
    <SCRIPT>

[-h], [--help]      : help on usage
[-v], [--version]: print version information
[-b], [--batch]     : create a batch file example
[-d], [--docs]       : create tool documentation in current working directory
[-s], [--story]      : maximum data history depth (default is unlimited)
[-c], [--cores]      : number of physical processors to use for computation
```



M. Neteler: "You can regard GRASS GIS not as an application, but rather as an environment that provides applications (commands/modules). The idea of running GRASS in parallel is to run several GRASS commands in parallel. This kind of parallelization becomes interesting when you want to perform the same processing (say, r.fill.stats), to many different input data like time series or a tile based approach (r.tile). You will need some job scheduler, e.g. slurm or equivalent (see <https://slurm.schedmd.com/rosetta.html>). A single server with multiple cores can be regarded as a simple HPC system. Of course also OpenStack/docker-based job schedulers might be used which are more suitable for cloud processing."

Examples



Example 1: derivation of land
cover change using global
300m resolution images



Example with the global land cover

Climate Research Data Package (CRDP)

[Download CRDP user guide](#)

Land Cover Maps - v1.6.1



3-epoch series of global land cover maps at 300m spatial resolution, where each epoch covers a 5-year period (2008-2012, 2003-2007, 1998-2002).

Each pixel value corresponds to the label of a land cover class defined using UN-LCCS classifiers.

For each epoch, the land cover map is delivered along with 4 quality flags which document the reliability of the classification:

- *qualityflag1* pixel has been processed or not,
- *qualityflag2* pixel status as defined by the pre-processing,
- *qualityflag3* number of valid observations available to derive the classification,
- *qualityflag4* level of confidence in the product (ranging from 0 to 100).

[Download quick user guide](#)

- ⬇ [2008-2012 epoch - v1.6.1 \(tif, 7z\)](#) - 2.8Go
- ⬇ [2003-2007 epoch - v1.6.1 \(tif, 7z\)](#) - 2.8Go
- ⬇ [1998-2002 epoch - v1.6.1 \(tif, 7z\)](#) - 1.7Go

- ⬇ [2008-2012 epoch - v1.6.1 \(netcdf\)](#) - 2.8Go
- ⬇ [2003-2007 epoch - v1.6.1 \(netcdf\)](#) - 2.8Go
- ⬇ [1998-2002 epoch - v1.6.1 \(netcdf\)](#) - 1.1Go

- ⬇ [Legend \(csv\)](#)
- ⬇ [Symbolology for ENVI \(.dsr\)](#)
- ⬇ [Symbolology for ArcGis \(.lyr\)](#)
- ⬇ [Symbolology for QGis \(.qml\)](#)

Only major LC changes were detected at 1km spatial resolution and limited to a certain number of classes (see CRDP user guide).

The land cover and their quality flags can be downloaded separately at [this address](#).

Some ideas about the data size



- at 300 m resolution: land mask about 1.4 billion pixels,
- about 6GB in size in memory (each image),
- About 80 CPU hours to compute,



Some ideas about the data size

```
> r = raster("ESACCI-LC-L4-LCCS-Map-300m-P5Y-2010-v1.6.1.tif")
> r
class       : RasterLayer
dimensions   : 64800, 129600, 8398080000  (nrow, ncol, ncell)
resolution   : 0.002777778, 0.002777778  (x, y)
extent       : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
coord. ref.  : +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
data source  : /data/LDN/ESACCI-LC-L4-LCCS-Map-300m-P5Y-2010-v1.6.1.tif
names        : ESACCI.LC.L4.LCCS.Map.300m.P5Y.2010.v1.6.1
values       : 0, 255  (min, max)
```

Objective: derive land cover change per pixel



Steps:

1. Prepare a tiling system,
2. Prepare functions for processing tiles,
3. Run in parallel (snowfall)
4. Generate a mosaic and build with pyramids (GDAL),



1. 2 x Intel Xeon Gold 6142 @ 2.60GHz
2. Core Count: 32 / Thread Count: 64
3. RAM MEMORY: 386048MB
4. DISK: 1920GB INTEL SSDSC2KG01 + 10001GB
Western Digital WD101KRYZ-01
5. File-System: ext4 / Mount Options: data=ordered
errors=remount-ro relatime rw
6. OPERATING SYSTEM: Ubuntu 16.04

Optional: NAS Synology (50TB)





Prepare a function

```
make_LC_tiles <- function(...) {  
  out.tif = ...  
  ...  
  m$i = plyr::join(data.frame(NAME=m$v), comb.leg,  
type="left")$Value  
  writeGDAL(m["i"], out.tif, type="Int16",  
options="COMPRESS=DEFLATE", mvFlag = -32768)  
}
```



Run in parallel

```
sfInit(parallel=TRUE, cpus=64)
```

```
snowfall 1.84-6.1 initialized (using snow 0.4-1): parallel execution  
on 64 CPUs.
```

```
sfExport("make_LC_tiles", "tile.tbl", "comb.leg", "cl.leg", "t.sel")
```

```
sfLibrary(rgdal)
```

```
Library rgdal loaded.
```

```
Library rgdal loaded in cluster.
```

```
sum.lst <- snowfall::sfClusterApplyLB(as.numeric(t.sel),  
function(x) { make_LC_tiles(x, tile.tbl=tile.tbl) })
```



You can relax and follow generation of files...

The screenshot shows the FileZilla interface. At the top, there are fields for host, port, and quickconnect. Below that, a tree view shows a remote site at `/data/LDN/tiled`. Under this path, there is a folder named `tiled` which contains several files: `MCD12Q1`, `MCD43A4`, `MDEM`, and `MOD10A2`. The main pane displays a list of files under the `tiled` directory, including `T10753`, `T10752`, `T10751`, `T10750`, `T10702`, `T10670`, `T10669`, `T10666`, and `T10665`. The list is sorted by filename. At the bottom, a status bar indicates "Queue: empty".

Filename	Filesize	Filetype	Last modified	Perm
..				
T10753		Direct...	21/09/16 17:07:32	drwxr-xr-x
T10752		Direct...	21/09/16 17:07:32	drwxr-xr-x
T10751		Direct...	21/09/16 17:07:32	drwxr-xr-x
T10750		Direct...	21/09/16 17:07:32	drwxr-xr-x
T10702		Direct...	21/09/16 17:07:32	drwxr-xr-x
T10670		Direct...	21/09/16 17:07:32	drwxr-xr-x
T10669		Direct...	21/09/16 17:07:32	drwxr-xr-x
T10666		Direct...	21/09/16 17:07:32	drwxr-xr-x
T10665		Direct...	21/09/16 17:07:32	drwxr-xr-x

7174 directories

Direction	Remote file	Size	Priorit	Status
-----------	-------------	------	---------	--------

Queue: empty



Final step: generate mosaic

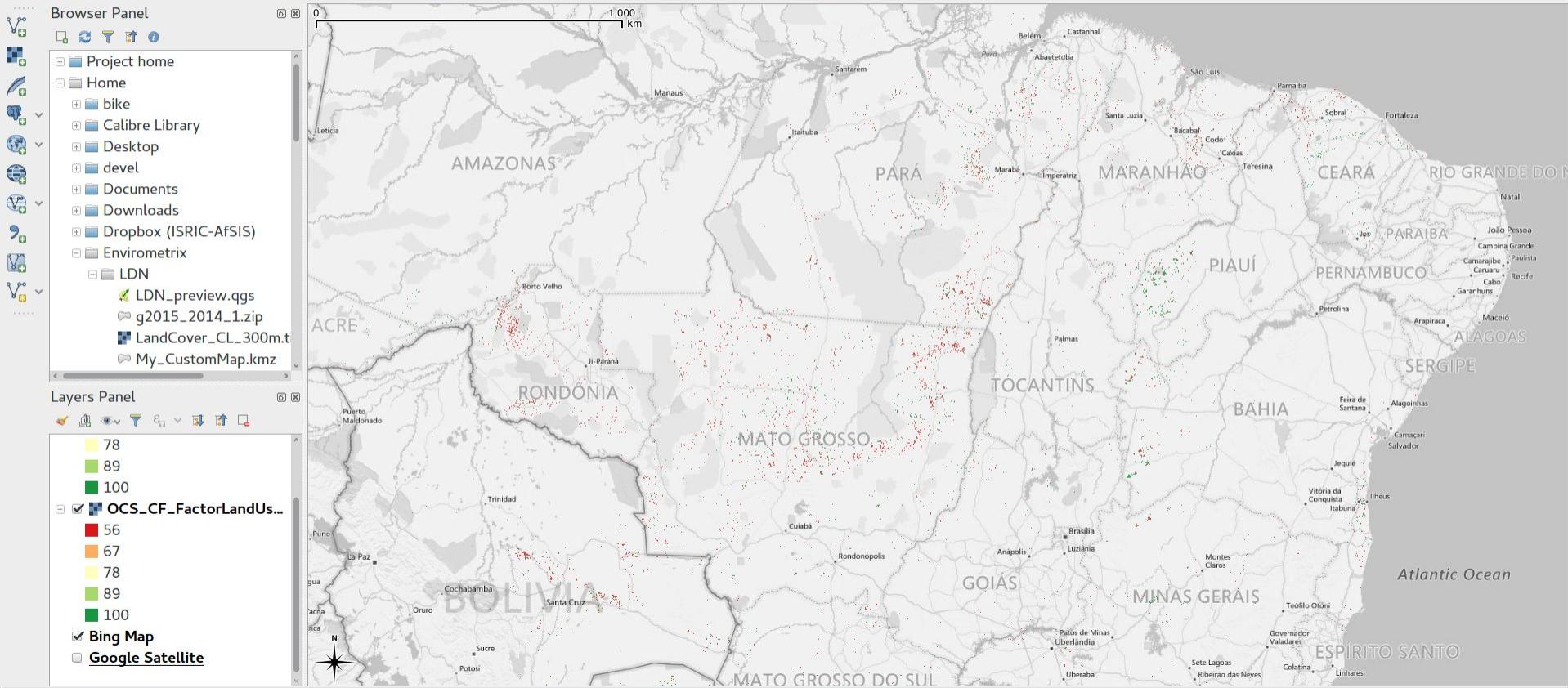
```
tmp.lst <- list.files(path="/data/LDN/tiled",
pattern=glob2rx(paste0("LandCover_CL_*.tif$")), full.names=TRUE,
recursive=TRUE)

## only 3178 tiles with values

out.tmp <- tempfile(fileext = ".txt")
vrt.tmp <- tempfile(fileext = ".vrt")
cat(tmp.lst, sep="\n", file=out.tmp)
system(paste0('gdalbuildvrt -input_file_list ', out.tmp, ' ', vrt.tmp))
system(paste0('gdalwarp ', vrt.tmp, ' LandCover_CL_300m.tif -ot
\"Int16\" -dstnodata \"-32767\" -co \"BIGTIFF=YES\" -multi -wm 2000 -co
\"COMPRESS=DEFLATE\" -r \"near\" -wo \"NUM_THREADS=ALL_CPUS\"'))
```

QGIS 2.16.1-Nødebo - LDN_preview

Project Edit View Layer Settings Plugins Vector Raster Database Web Processing Help



QGIS 2.16.1-Nødebo - LDN_preview

Project Edit View Layer Settings Plugins Vector Raster Database Web Processing Help



Browser Panel

- + Project home
- Home
 - + bike
 - + Calibre Library
 - + Desktop
 - + devel
 - + Documents
 - + Downloads
- + Dropbox (ISRIC-AFSIS)
 - Envirometric
 - + LDN
 - LDN_preview.qgs
 - g2015_2014_1.zip
 - + LandCover_CL_300m.t
 - My_CustomMap.kmz

Layers Panel

- 78
- 89
- 100
- OCS_CF_FactorLandUs...
 - 56
 - 67
 - 78
 - 89
 - 100
- Bing Map
- Google Satellite



Coordinate -5709297,-1166080



Scale 1:94,352



Magnifier 100%



Rotation 0.0



Render

EPSG:3857 (OTF)



Project Edit View Layer Settings Plugins Vector Raster Database Web Processing Help

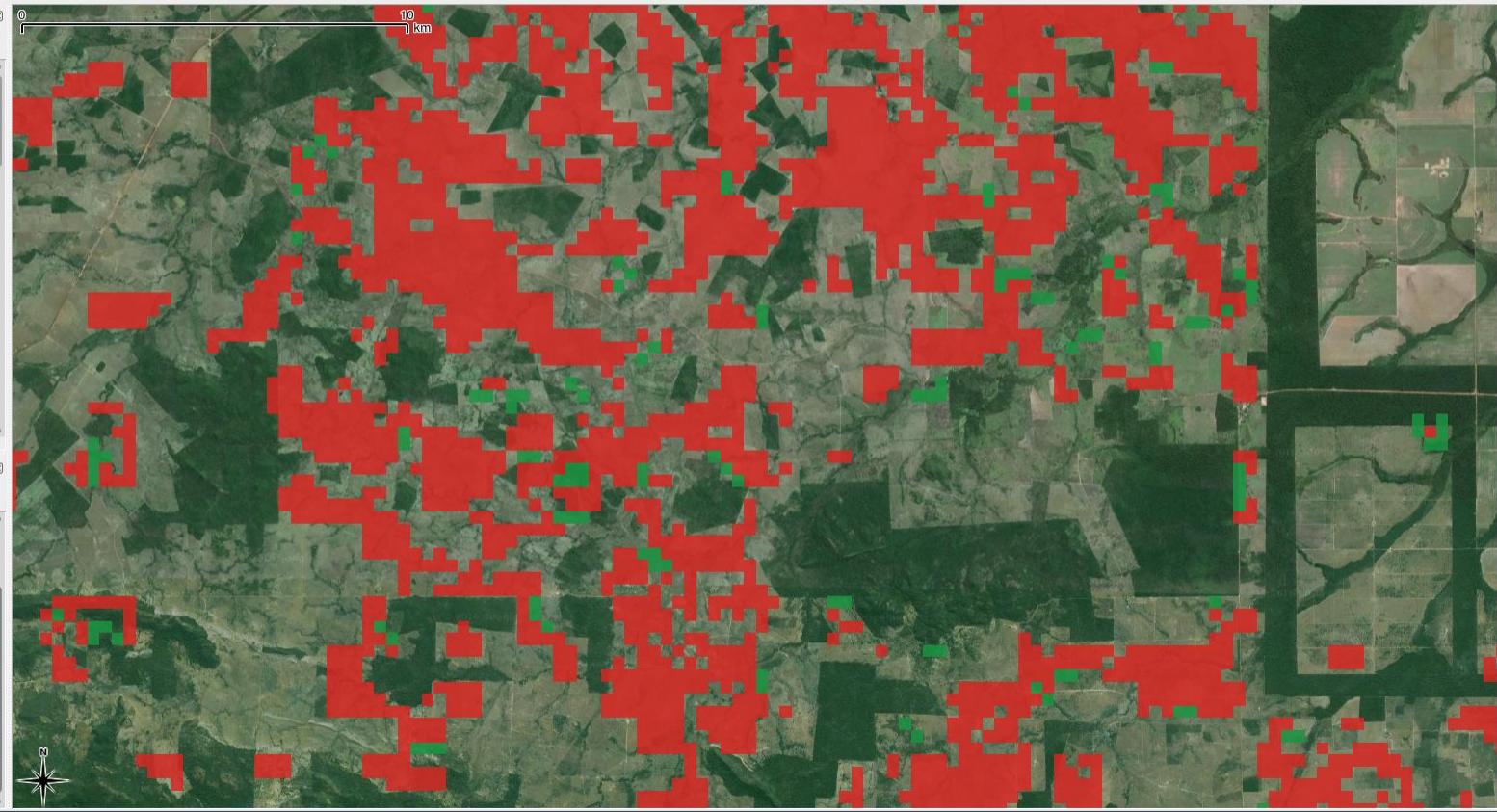


Browser Panel

- + Project home
- Home
 - + bike
 - + Calibre Library
 - + Desktop
 - + devel
 - + Documents
 - + Downloads
 - + Dropbox (ISRIC-AFSIS)
 - + Envirometrics
 - LDN
 - LDN_preview.qgs
 - g2015_2014_1.zip
 - LandCover_CL_300m.t
 - My_CustomMap.kmz

Layers Panel

- OCS_CF_FactorLandUs...
 - 56
 - 67
 - 78
 - 89
 - 100
- Bing Map
- Google Satellite



Examples



Example 2: derivation of DEM
surface parameters

Digital Land Surface Model



Two sources of DEM data:

- 1/3rd arc-second (10 m) National Elevation Dataset (NED)
- 1 arc-second (30 m) global ALOS AW3D30 Digital Surface Model

which one do we use?

Modeling DLSM



Use training points to build a model to merge various elevation data sources:

$$mDLSM = f(NED, AW3D30, NDVI, HH/HV)$$

two data sources, NDVI image and radar bands.

DLSM scheme

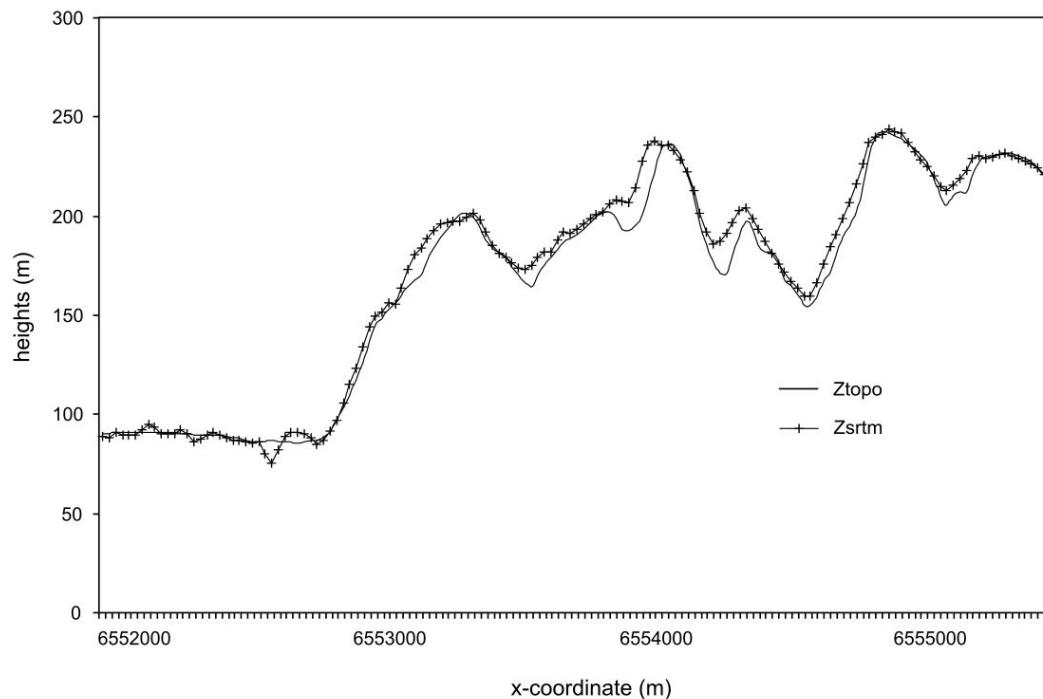


FIGURE 10 Transect of the SRTM and TOPO DEMs for Baranja Hill from West to East at $y = 5,073,012$. Note that the difference between the two DEMs can be used to map the height of canopy. In this case, we can also notice a difference between the two DEMs which is due to the foreshortening effect of the radar technology (systematic error).

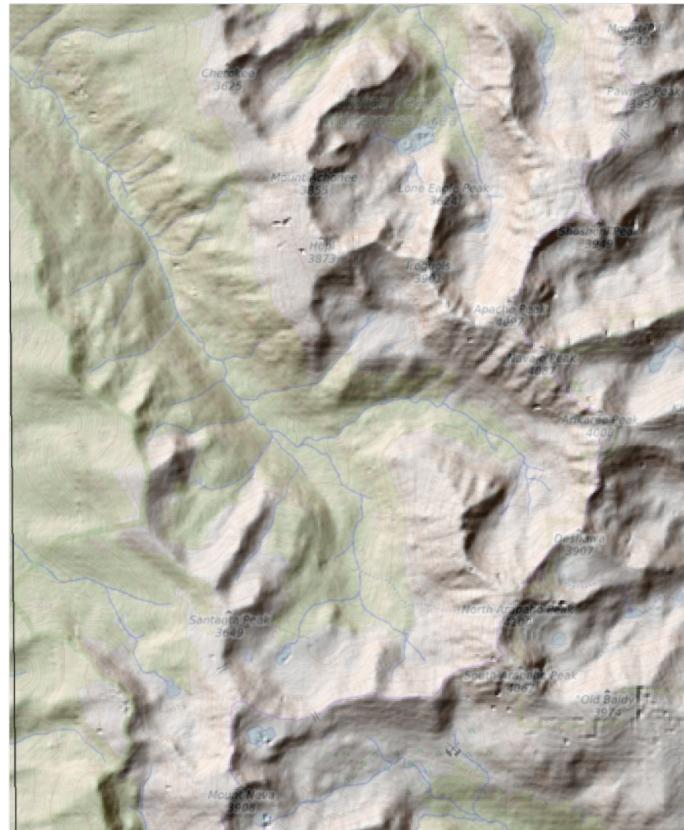


Results

AW3D30



mDLSM



Conclusions I



- Yes, you can use *R* (*in combination with GDAL, SAGA GIS and similar*) to process large rasters,
- Do not stop programming until you have reached the maximum usage of hardware — RAM + cores (at least 90%?),
- Start with smaller subsets then increase data volumes,

Conclusions II



- *SAGA GIS is already optimized for large data (automatic parallelization),*
- *GRASS GIS is quite efficient in processing large rasters, but parallelization needs to be implemented through scripting,*
- *Investing in hardware / storage will eventually be unavoidable,*