Scalable Earth-observation Analytics for Geoscientists: Spacetime Extensions to the Array Database SciDB

Marius Appel[1], Florian Lahn[1], Edzer Pebesma[1],
Wouter Buytaert[2], Simon Moulds[2]

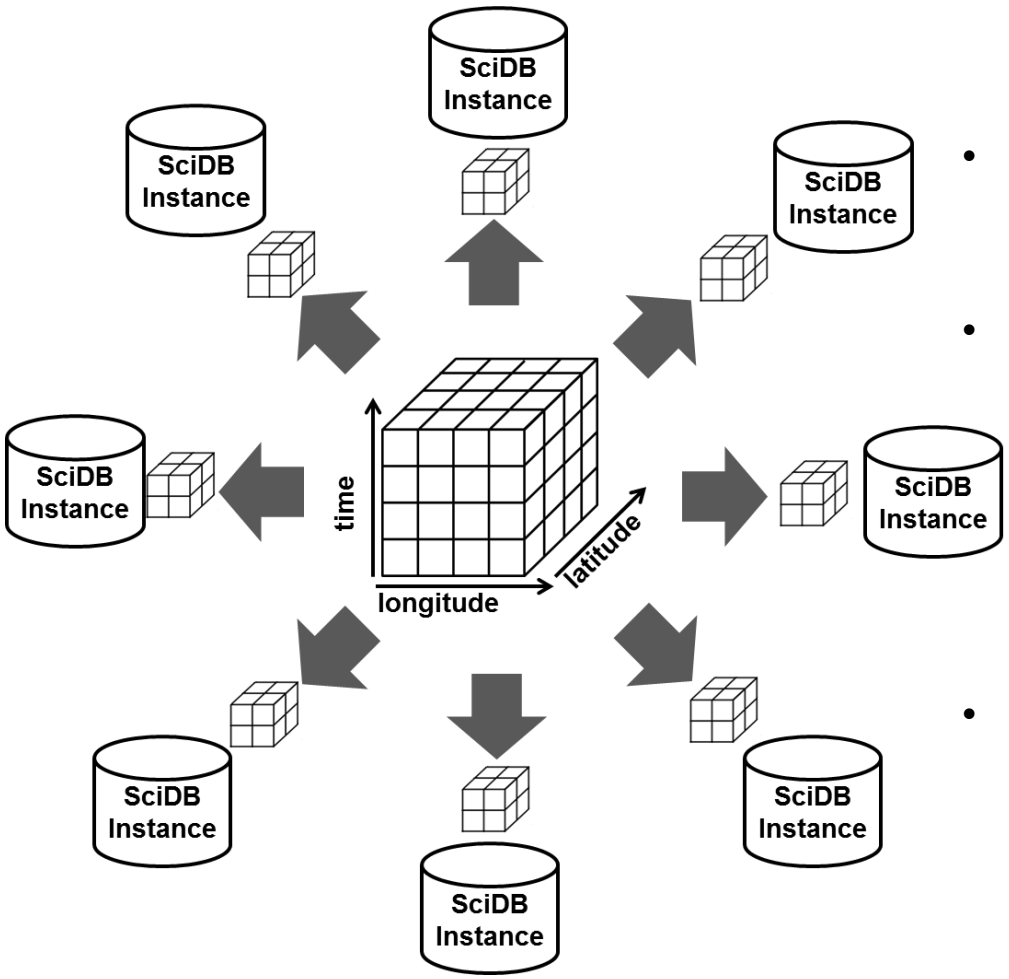
marius.appel@uni-muenster.de

ifgi
Institute for Geoinformatics
University of Münster

EGU2016-11780
April 22, 2016

[1] Institute for Geoinformatics, University of Muenster, Germany
[2] Department of Civil and Environmental Engineering, Imperial College London, United Kingdom

Scalable analytics with SciDB



- SciDB: **open-source** array database with focus on complex analytics [1]
- Distribution of storage and computational load by multidimensional chunking
- Advantages w.r.t scientific analytics:
  - Multidimensional sparse array model
  - Fits well to shared-nothing environments
  - Interfaces ScaLapack
  - Interfaces to R, python, and julia
  - extensible by user-defined functions
- Earth-observation analytics
  - Supports only CSV and custom binary file formats
  - Metadata gets lost (e.g. spatial / temporal reference)
  - Remote sensing imagery comes as temporal snapshots

[1] Stonebraker, M., Brown, P., Zhang, D., & Becla, J. (2013). SciDB: A database management system for applications with complex analytics. *Computing in Science & Engineering*, *15*(3), 54-62.

Spacetime Extensions to SciDB

Objective:

- Enrich SciDB to facilitate working with Earth observation datasets
- Publish tools as open source to support reproducibility in scalable earth observation analytics

scidb4geo

- SciDB plugin that adds new functions to the query language
- Stores spatial / temporal reference metadata in SciDB's system catalog
- General key value metadata on arrays and attributes / bands.

scidb4gdal

- A GDAL (Geospatial Data Abstraction Library) driver that supports **reading and writing** SciDB arrays to and from > 100 different file formats
- Minimal dependencies: works on Windows, Linux, and Mac
- Supports multi-tile and multi-temporal datasets

Tools and detailed documentation available at:
http://github.com/mappl/scidb4geo
http://github.com/mappl/scidb4gdal

Contact:
Marius Appel
marius.appel@uni-muenster.de

scidb4geo: a SciDB plugin for geographic reference

Idea

- Add new functions to the SciDB query language AFL to define and manipulate spatial / temporal reference of arrays
- Store metadata in SciDB's system catalog
- Let other tools automatically call these functions to use location in analyses

Features

- Annotation of arrays by spatiotemporal reference information
- Storage of general key value metadata on array and attribute / band level
- Overlay operation to join arrays based on location

Temporal Reference of arrays =

- … which array dimension is time?
- … which date / time is at array cell 0?
- … what is the time interval between successive cells?

Spatial Reference of arrays =

- … which array dimensions are lat lon?
- … which reference system or projection is used?
- … how do array integer coordinates relate to world coordinates as an affine transformation?

Details

- https://github.com/mappl/scidb4geo
- (next page)

scidb4geo: a SciDB plugin for geographic reference

New AFL (Array Functional Language) operators	
Operator	**Description**
eo_arrays()	Lists geographically referenced arrays
eo_setsrs()	Sets the spatial reference of existing arrays
eo_getsrs()	Gets the spatial reference of existing arrays
eo_regnewsrs()	Registers custom spatial reference systems
eo_extent()	Computes the geographic extent of referenced arrays
eo_cpsrs()	Copies the spatial reference from one array to another array
eo_settrs()	Sets the temporal reference of arrays
eo_gettrs()	Gets the temporal reference of arrays
eo_setmd()	Sets key value metadata of arrays and array attributes
eo_getmd()	Gets key value metadata of arrays and array attributes
eo_over()	Overlays two geographically referenced arrays

scidb4gdal: a GDAL driver to read / write SciDB arrays

Idea

- GDAL supports > 100 raster formats → A driver for SciDB would cover most earth-observation file formats
- Use SciDB's binary format and the SciDB web-service Shim to connect to the database
- use spatial and temporal reference information to automatically convert array cell indexes to coordinates

Features

- Depends only on libcurl, no SciDB binaries needed → works on Linux, Mac, and Windows systems
- Read and write access to SciDB arrays
- Support for three dimensional multi-tile and multi-temporal datasets

2d read / write access using gdal_translate

```
gdal_translate -of SciDB
        test.tif "SCIDB:array=test"

gdal_translate -of NetCDF
        -projwin 7 52 7.1 52.1  -b 1
        "SCIDB:array=test" test.nc
```

Details

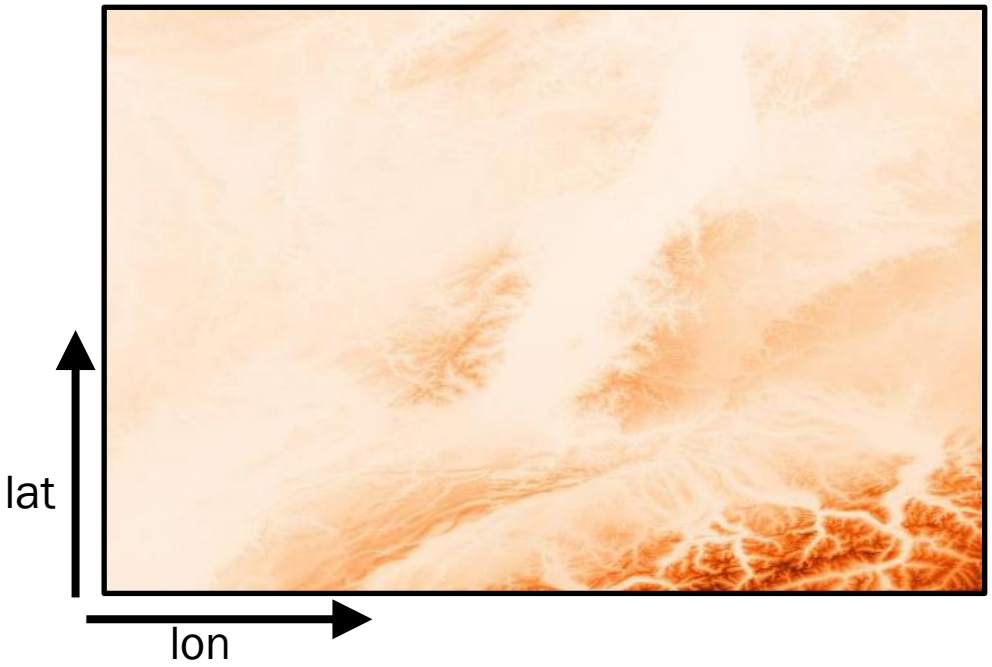
- https://github.com/mappl/scidb4gdal
- (next pages)

Creating multi-tiled arrays with GDAL

```
gdal_translate  -of SciDB
       -co bbox=5 46.6 10 50"   -co "srs=EPSG:4326"
       "srtm00.tif" "SCIDB:array=srtm"

gdal_translate -of SciDB "srtm10.tif" "SCIDB:array=srtm"
gdal_translate -of SciDB "srtm01.tif" "SCIDB:array=srtm"
gdal_translate -of SciDB "srtm11.tif" "SCIDB:array=srtm"
gdal_translate -of SciDB "srtm02.tif" "SCIDB:array=srtm"
gdal_translate -of SciDB "srtm12.tif" "SCIDB:array=srtm"
```

SciDB Array:



Creating multi-temporal 3d arrays with GDAL

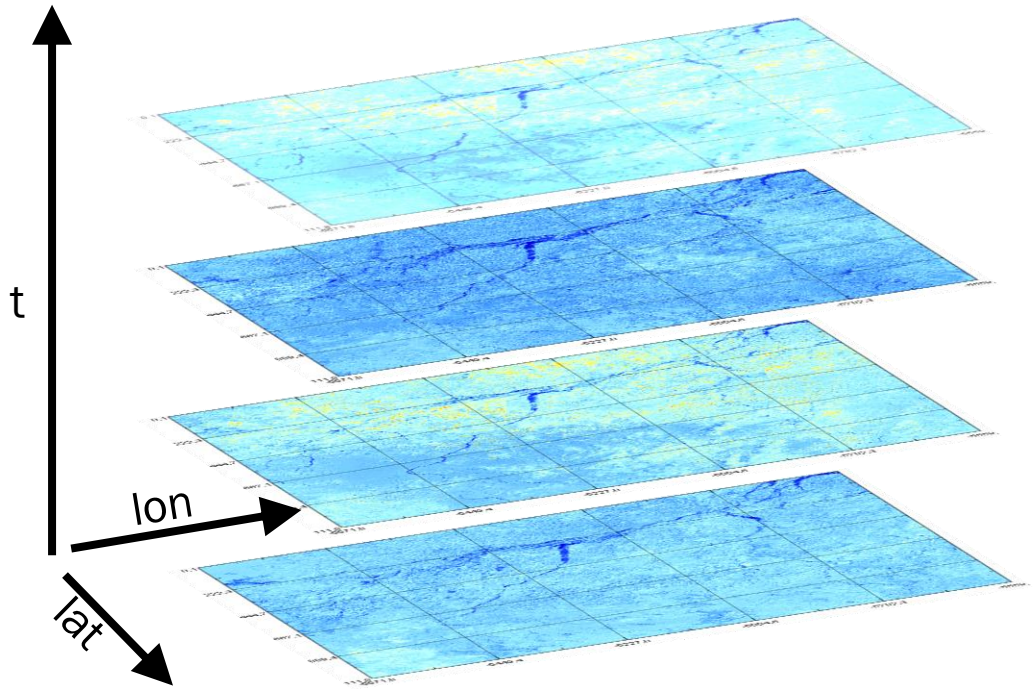
```
gdal_translate -of SciDB  -co "type=ST" -co "dt=P16D" -co "t=2016-01-01"
        "2016-01-01.tif"  "SCIDB:array=array3d"

gdal_translate -of SciDB -co "t=2016-01-16"
        "2016-01-16.tif" "SCIDB:array=array3d"

gdal_translate -of SciDB -co "t=2016-02-02"
        "2016-02-02.tif"  "SCIDB:array=array3d"

gdal_translate -of SciDB -co "t=2016-02-18"
        "2016-02-18.tif"  "SCIDB:array=array3d"
```

SciDB Array:



Example: Scalable Landsat time series analysis in R

1. Loading Landsat time series as 3d array using R and scidb4gdal

Get filenames of all Landsat images

Load first image and create a three dimensional array with daily temporal resolution

Iterate over other images

Extract dates from filenames

```
library(gdalUtils)
files = list.files(pattern="*.tif")

gdal_translate(src_dataset = files[1],
               dst_dataset = "SCIDB:array=landsat1",
               of = "SciDB",
               co = list("t=2000-01-01",
                         "type=STS","dt=P1D"))

for (i in 2:length(files)) {
  d = strptime(substr(files[i],1,7), format="%Y%j")
  gdal_translate(src_dataset = files[i],
                 dst_dataset = "SCIDB:array=landsat1", of = "SciDB",
                 co = list(paste("t=",format(d),sep="")))
}
```

Ingest current image based on its date to the correct position in the SciDB array.

Example: Scalable Landsat time series analysis in R

2. Run change detection within SciDB from R

```
library(scidb)
scidbconnect(…)

arr_temporal <-
       repart(scidb("landsat1"), chunk=c(1,1,1024))
arr_name <- str(arr_temporal)

aflquery <- paste("store(r_exec(", arr_name,
    "'output_attrs=2,'expr=
      require(bfast) # see [2]
      ndvi<- as.ts(ndvi,frequency=23,start=c(200
      chngpt <- bfast(ndvi,season=\"harmonic\",
                max.iter=1,breaks=1)
      return(list(chngpt$Time, changpt$Magnitude))'),
   resultsarray2d)", sep="")
iquery(aflquery, return=T);

iquery("eo_cpsrs(landsat1, resultarray2d)")
```

Connect to the database

Reshape the array to chunks of time series

Run R code (change detection [2]) on each time series and return the time and magnitude of the strongest change point

Copy spatial reference information to the two-dimensional result array

[2] Verbesselt, J., Hyndman, R., Newnham, G., & Culvenor, D. (2010). Detecting trend and seasonal changes in satellite image time series. Remote sensing of Environment, 114(1), 106-115.

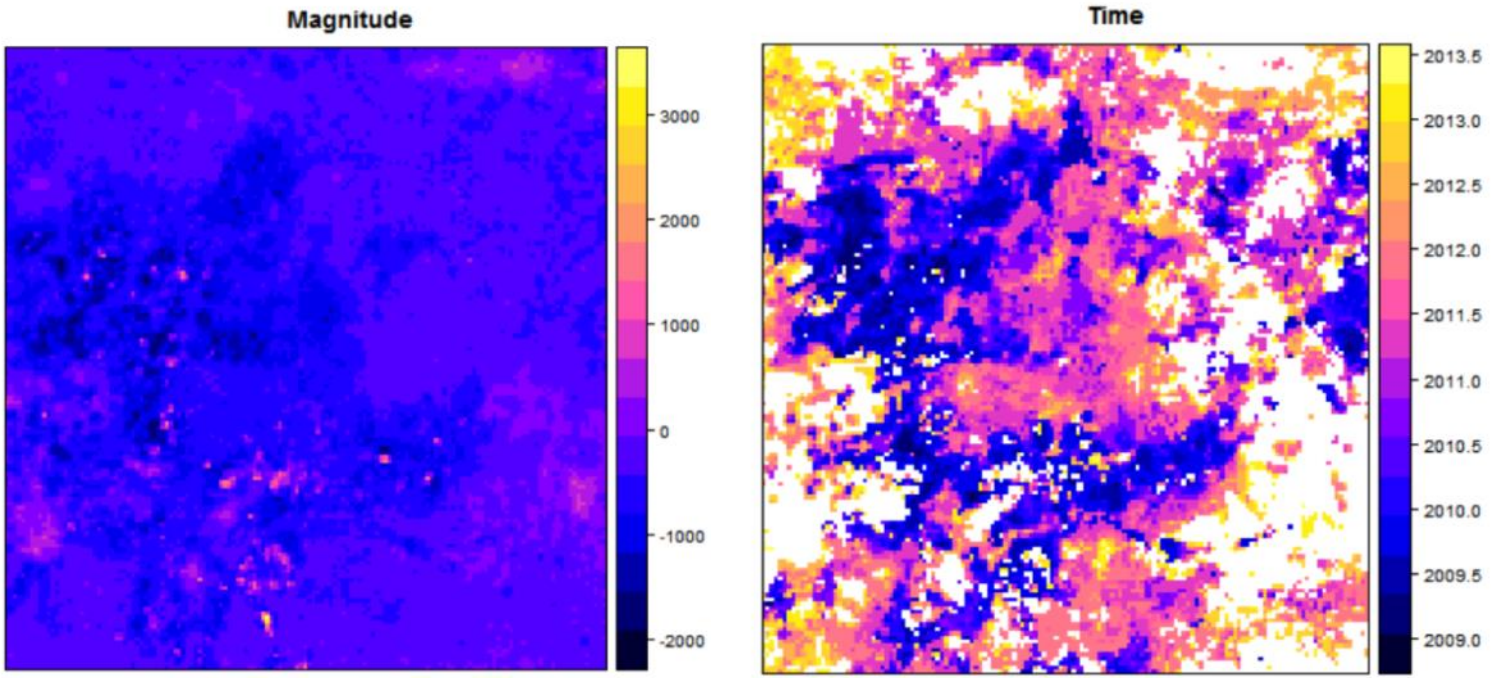
Example: Scalable Landsat time series analysis in R

3. Download results from SciDB using the rgdal package and visualize output

```
library(rgdal)
x = readGDAL("SCIDB:array=resultarray2d")
spplot(x["Time"])
spplot(x["Magnitude"])
```

Download data to R with GDAL

Visualize results using the sp package



Discussion and Outlook

Limitations

- In-database functionality still limited: e.g. no reprojection
- Fusion of multiple arrays by spacetime
- Ingestion can be time-consuming depending on selected chunk sizes
- Core SciDB operators work on array indexes only

Future work

- Overwrite core SciDB operators to work on spatiotemporal coordinates instead of array indexes using the implemented plugin
- Easier to use scripts (Python + R) around gdal_translate for ingesting specific datasets (e.g. Landsat, MODIS)
- An easier integration to run R functions within SciDB queries on spatial and temporal arrays (e.g. by providing chunks as sp objects)

Contact
Marius Appel
marius.appel@uni-muenster.de

https://github.com/mappl/scidb4geo
https://github.com/mappl/scidb4gdal