



Big data to support European policy

use case on S2 for forestry

P. Kempeneers, D. Rodriguez, V. Syrris, D. De Marchi, V. Vasilev, P. Hasenohr, A. Burger and P. Soille (1)
P. Zarco-Tejada, R. Hernandez-Clemente and P. Beck (2)

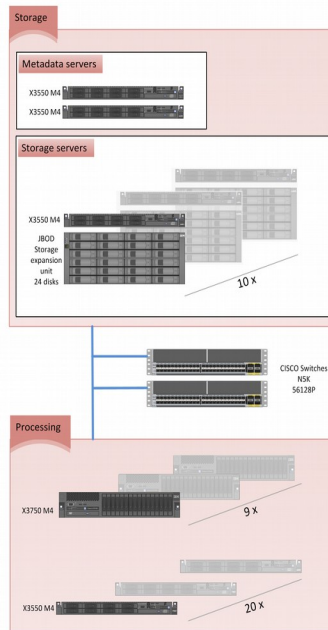
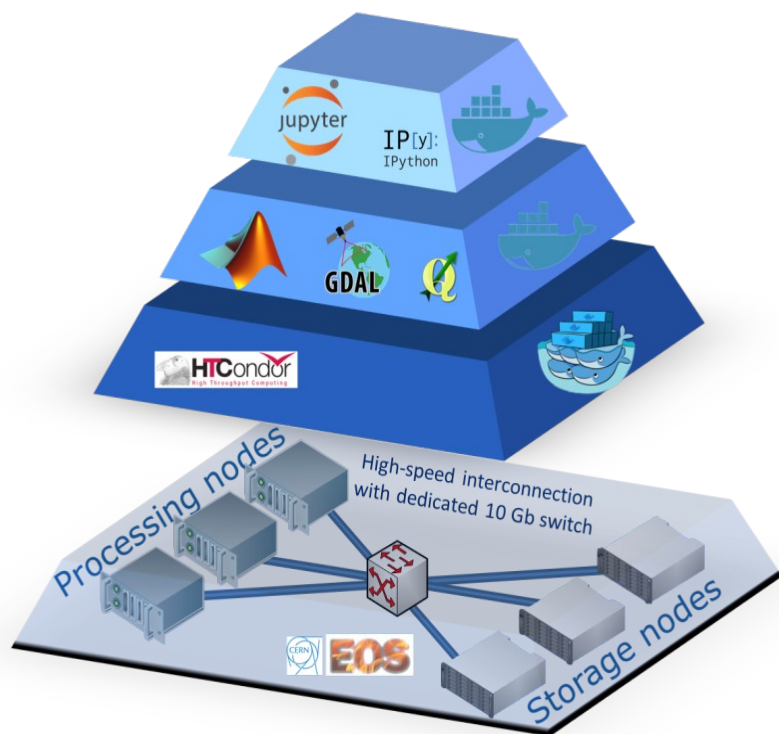
(1) Directorate I: Competences (EO&&SS@BigData Text and Data Mining)

(2) Directorate D: Sustainable Resources (Bio-economy)

OUTLINE

- PART 1: Big Data at JRC
 - Infrastructure:
 - data storage
 - processing
 - Processing library
- PART 2: Big Data for policy support
 - Sentinel data
 - Use case on canopy health monitoring

Big Data at JRC: infrastructure



22 nodes for **storage**

2 nodes for **network**
(10 Gb NIC)

29 nodes for **processing**
600 cores in total
18 (12) GB RAM/core

distributed file system



- **CERN EOS**

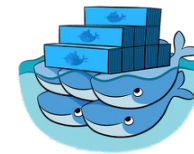
- Open source distributed file system by CERN
- Deployed at JRC in the framework of CERN-JRC collaboration;
- Currently using replica 2 leading 0.7 PB net storage (1.4 raw storage);
- RAIN 6 model to be tested for files > 100 MB

Processing infrastructure

- **HTCondor** cluster



- Jobs run in **docker** containers



- flexible management of processing environment
- custom builds for different requirements
- Facilitates upgrades of processing environment (libraries, tools)

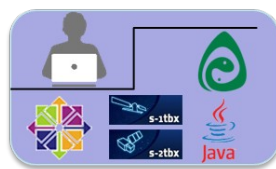
JEODPP Batch Processing System

Container-based cluster management

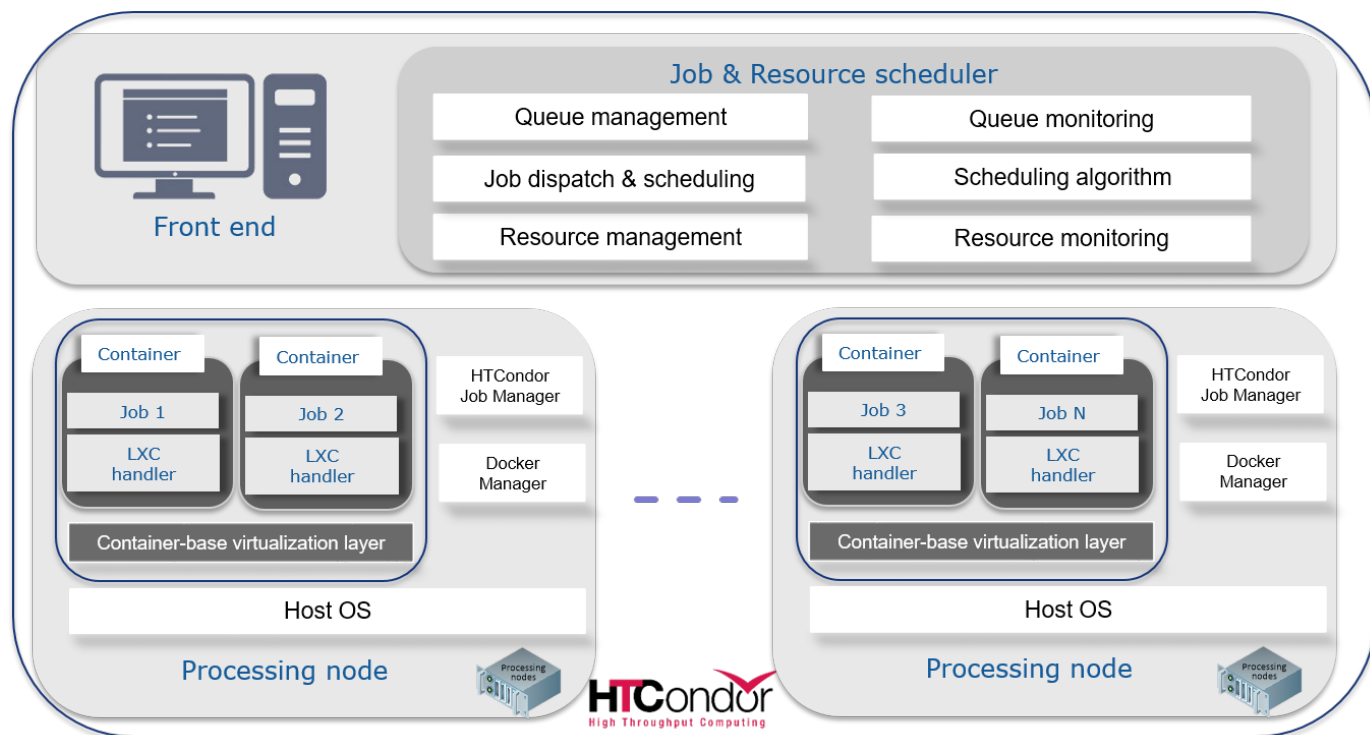
Multi-type user environment

Based on different:

- Libraries
- Tools
- Software
- Versions
- Distro: Debian/Centos



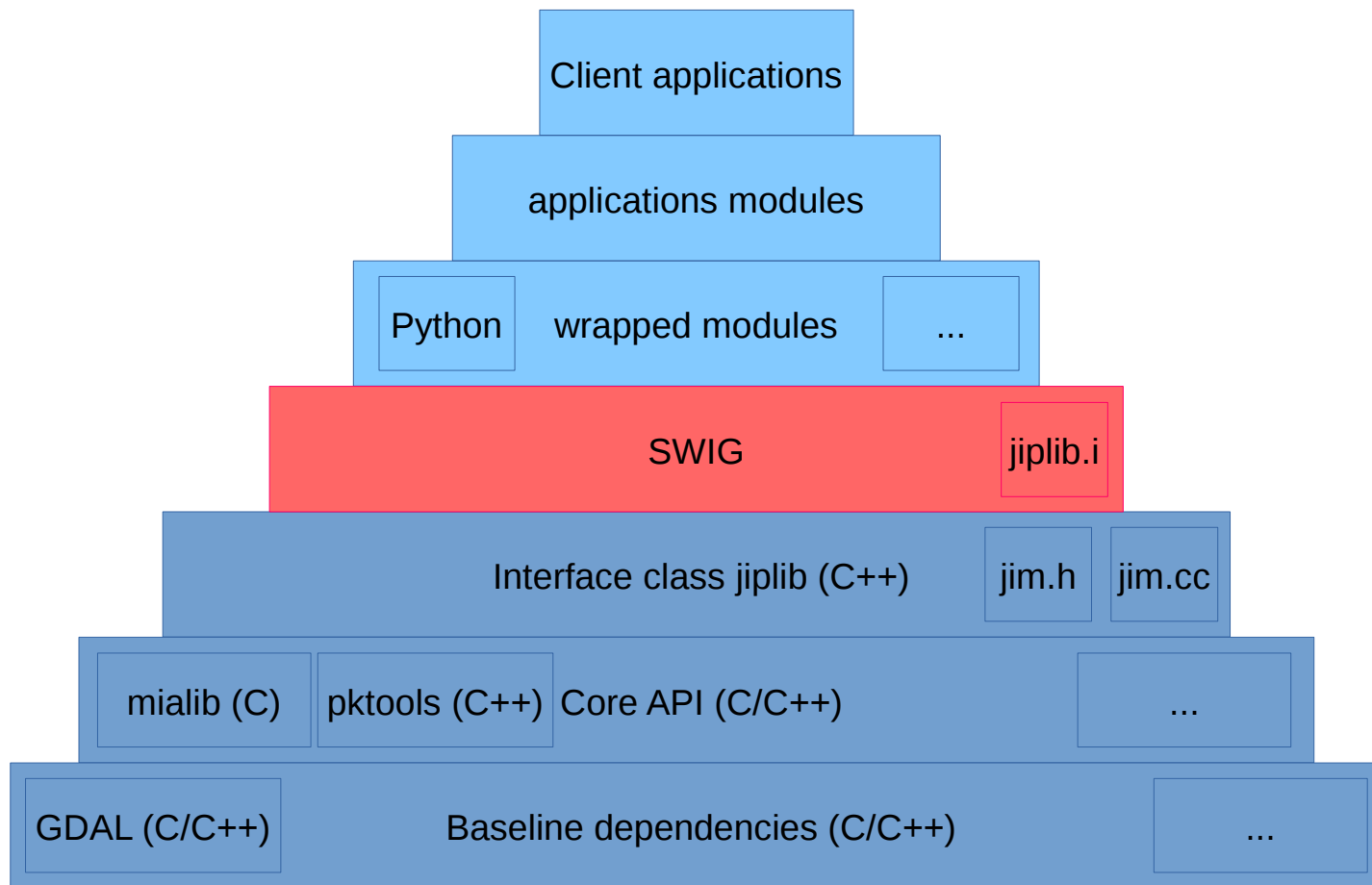
Images are built based on user requirements



Joint Image Processing Library (JIPLib)

- **Requirements**
 - **Take advantage of existing libraries (not reinvent the wheel)**
 - **Benefit from in house expertise**
 - **Big Data processing**
 - **Control and maintenance**
 - **Bind to scripting language (Python)**

JRC Image Processing library (JIPLib)



JiPLib Python package

jiplib package

I/O raster operations

filters

raster-vector

point operations

morpho operations

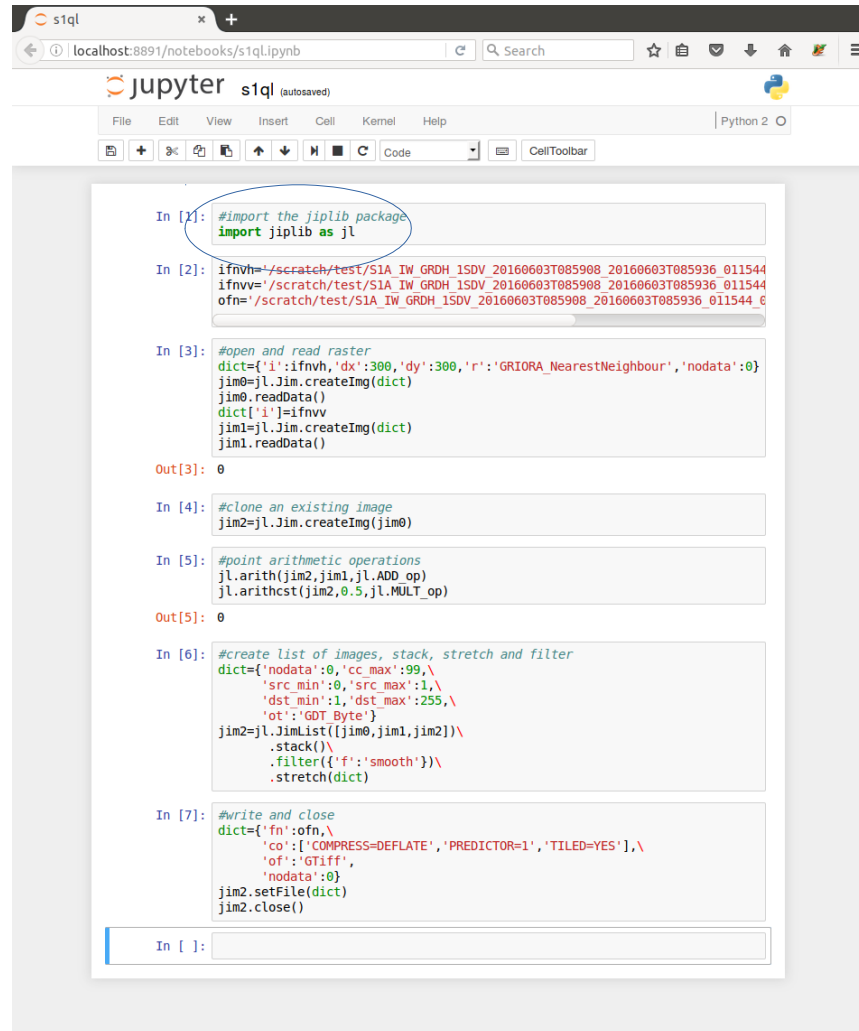
statistics

composites

classification

segmentation

...



```
In [1]: #import the jiplib package
import jiplib as jl

In [2]: ifnvh="/scratch/test/S1A_IW_GRDH_1SDV_20160603T085908_20160603T085936_011544
ifnvw="/scratch/test/S1A_IW_GRDH_1SDV_20160603T085908_20160603T085936_011544
ofn="/scratch/test/S1A_IW_GRDH_1SDV_20160603T085908_20160603T085936_011544

In [3]: #open and read raster
dict={'i':ifnvh,'dx':300,'dy':300,'r':'GRIORA_NearestNeighbour','nodata':0}
jim0=jl.Jim.createImg(dict)
jim0.readData()
dict['i']=ifnvw
jim1=jl.Jim.createImg(dict)
jim1.readData()

Out[3]: 0

In [4]: #clone an existing image
jim2=jl.Jim.createImg(jim0)

In [5]: #point arithmetic operations
jl.arith(jim2,jim1,jl.ADD_op)
jl.arithcst(jim2,0.5,jl.MULT_op)

Out[5]: 0

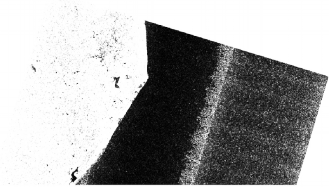
In [6]: #create list of images, stack, stretch and filter
dict={'nodata':0,'cc_max':99,\
      'src_min':0,'src_max':1,\
      'dst_min':1,'dst_max':255,\
      'ot':'GDT_Byte'}
jim2=jl.JimList([jim0,jim1,jim2])\
      .stack()\
      .filter({'f':'smooth'})\
      .stretch(dict)

In [7]: #write and close
dict={'fn':ofn,\
      'co':['COMPRESS=DEFLATE','PREDICTOR=1','TILED=YES'],\
      'of':'GTiff',\
      'nodata':0}
jim2.setFile(dict)
jim2.close()

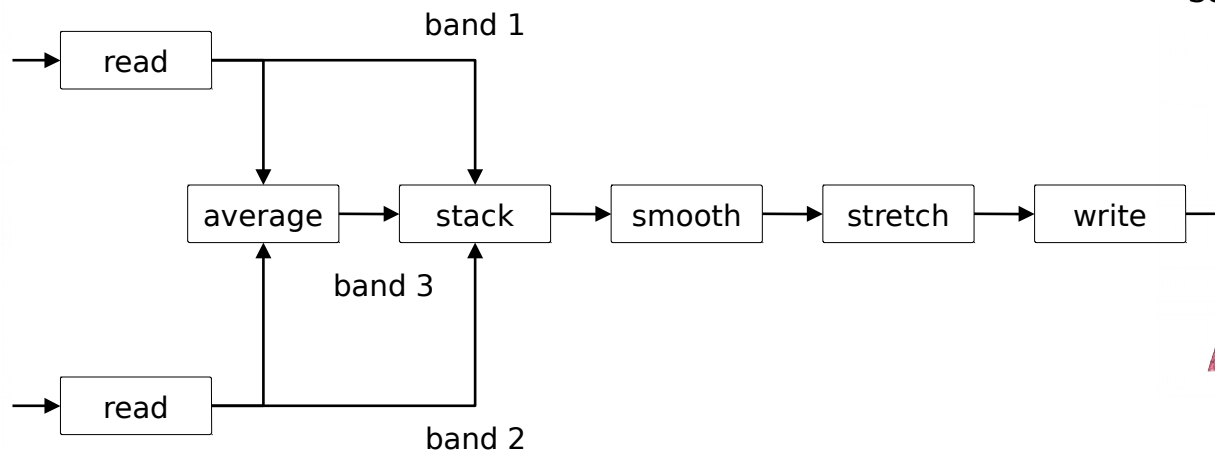
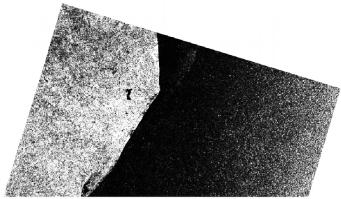
In [ ]:
```

JIPlib example: S1 quicklooks

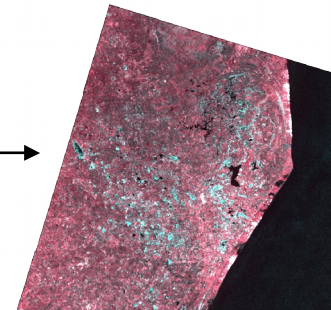
Sentinel 1 VH image



Sentinel 1 HH image



Sentinel 1 quicklook



```
In [1]: #import the jiplib package
import jiplib as jl

In [2]: ifnvh='/scratch/test/S1A_IW_GRDH_1SDV_20160603T085908_20160603T085936_011544
ifnvv='/scratch/test/S1A_IW_GRDH_1SDV_20160603T085908_20160603T085936_011544
ofn='/scratch/test/S1A_IW_GRDH_1SDV_20160603T085908_20160603T085936_011544_0

In [3]: #open and read raster
dict={'i':ifnvh,'dx':300,'dy':300,'r':'GRIORA_NearestNeighbour','nodata':0}
jim0=jl.Jim.createImg(dict)
jim0.readData()
dict['i']=ifnvv
jim1=jl.Jim.createImg(dict)
jim1.readData()

Out[3]: 0

In [4]: #clone an existing image
jim2=jl.Jim.createImg(jim0)

In [5]: #point arithmetic operations
jl.arith(jim2,jim1,jl.ADD_op)
jl.arithcst(jim2,0.5,jl.MULT_op)

Out[5]: 0

In [6]: #create list of images, stack, stretch and filter
dict={'nodata':0,'cc_max':99,\
      'src_min':0,'src_max':1,\
      'dst_min':1,'dst_max':255,\
      'ot':'GDT_Byte'}
jim2=jl.JimList([jim0,jim1,jim2])\
      .stack()\
      .filter({'f':'smooth'})\
      .stretch(dict)

In [7]: #write and close
dict={'fn':ofn,\
      'co':['COMPRESS=DEFLATE','PREDICTOR=1','TILED=YES'],\
      'of':'GTiff',\
      'nodata':0}
jim2.setFile(dict)
jim2.close()

In [ ]:
```

read

average

stack

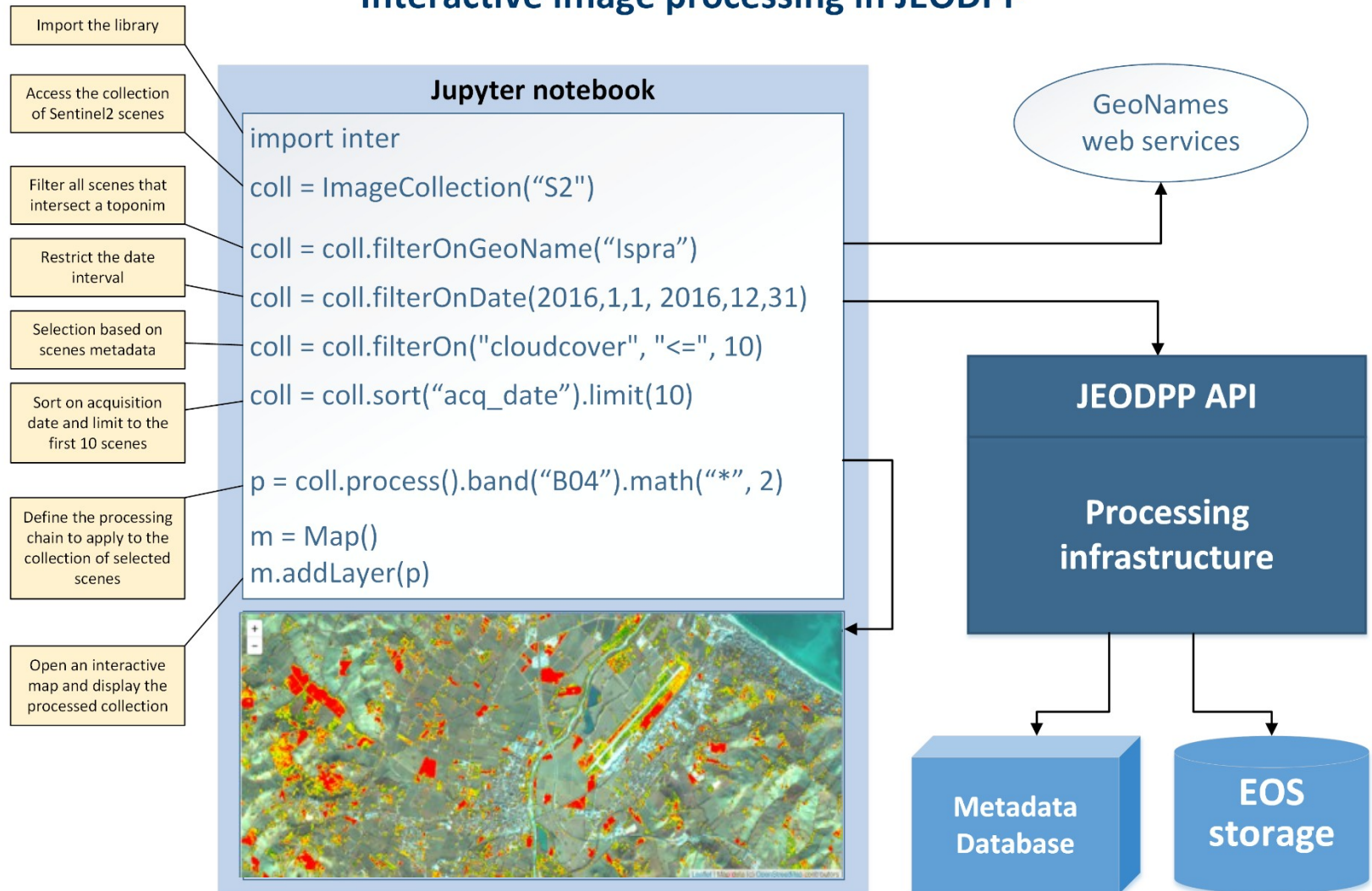
smooth

stretch

write

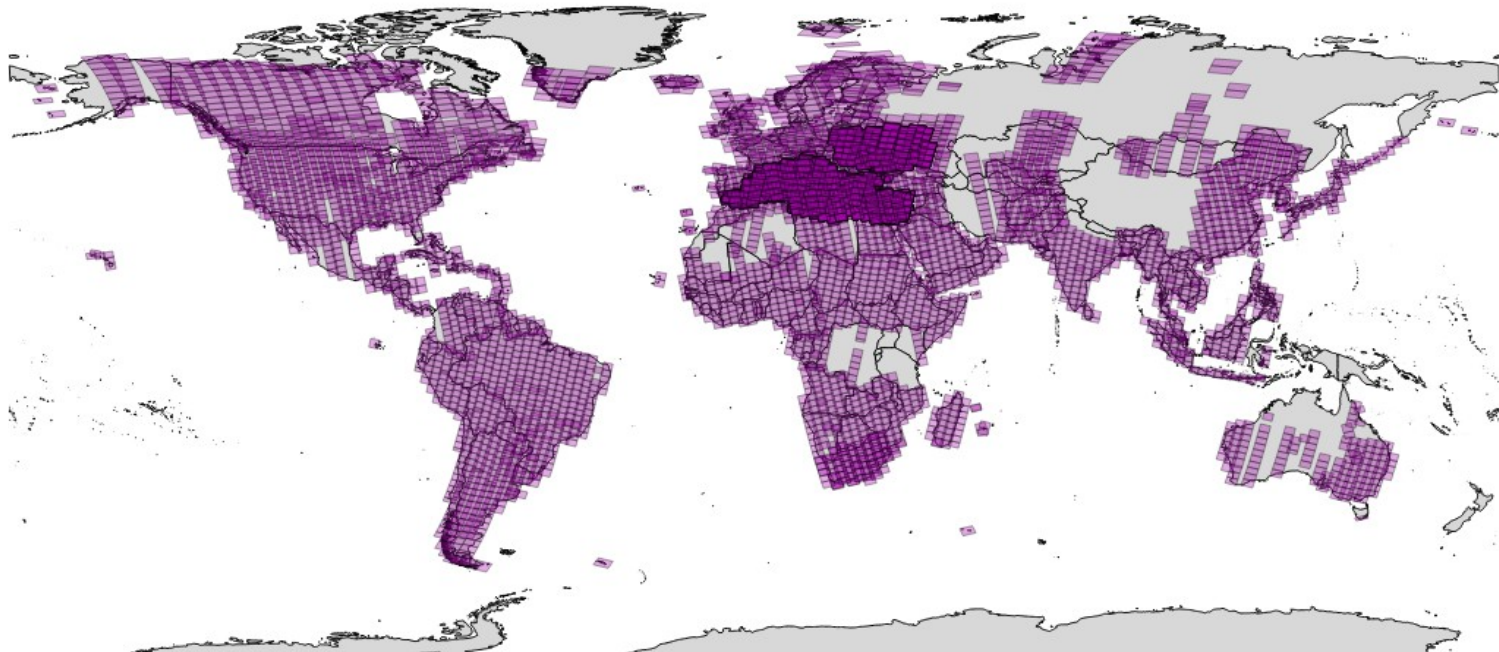
Big Data at JRC: Interactive Processing

Interactive image processing in JEODPP



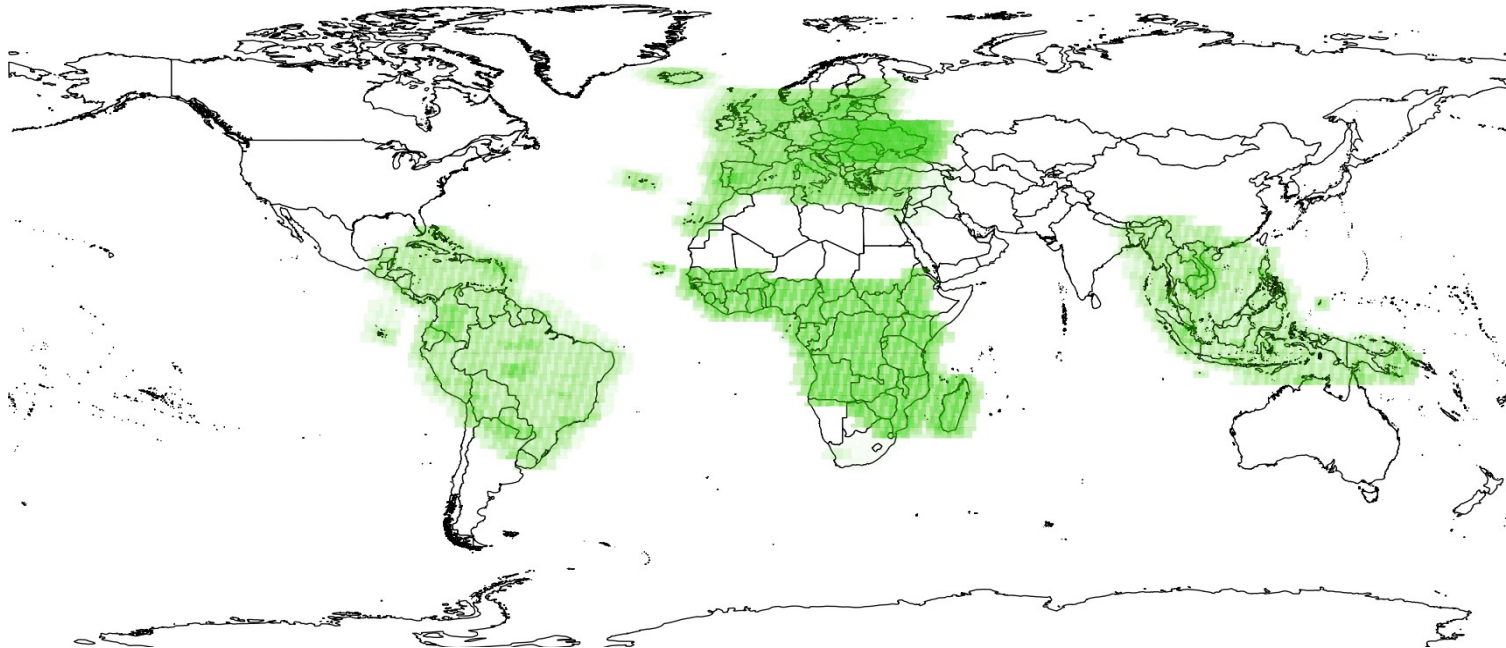
Big Data for policy support

- *Sentinel 1*
 - *Data downloaded from CopHub on request (3 projects)*
 - *Currently 30 TB of data (source)*
 - *Data pre-processing (terrain correction)*



Sentinel 1 and 2 data

- *Sentinel 2*
 - *Scheduled data download for defined AOI's (tropical zones, Europe)*
 - *Currently 35k products, 105 TB of data*
 - *Tests of sen2cor for selected datasets*



Use case: canopy health monitoring

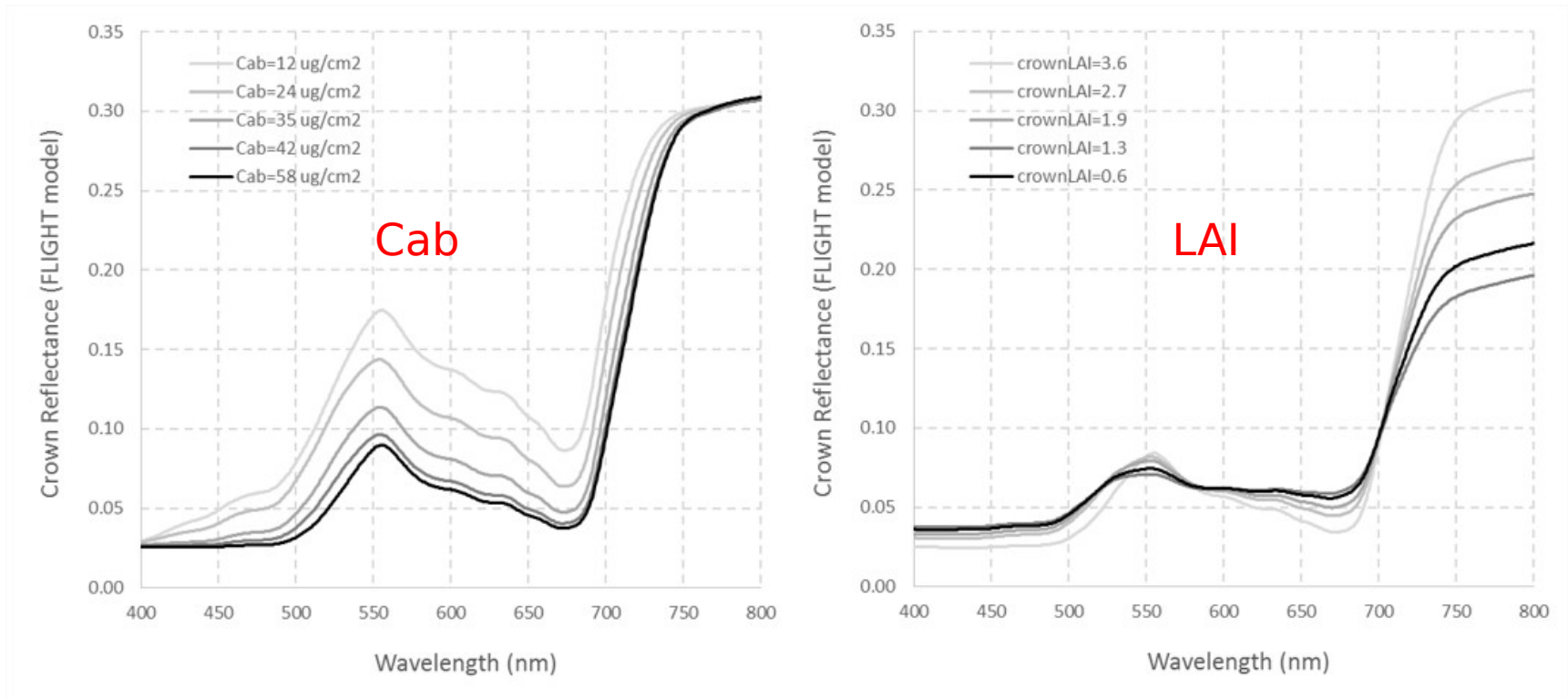
- *Early stress detection in pine forests*
 - ***pine wood Nematode***
- *requires fine spatial resolution imagery*
 - ***crown level***
- *Requires time series to monitor seasonal changes*
 - ***decline process***

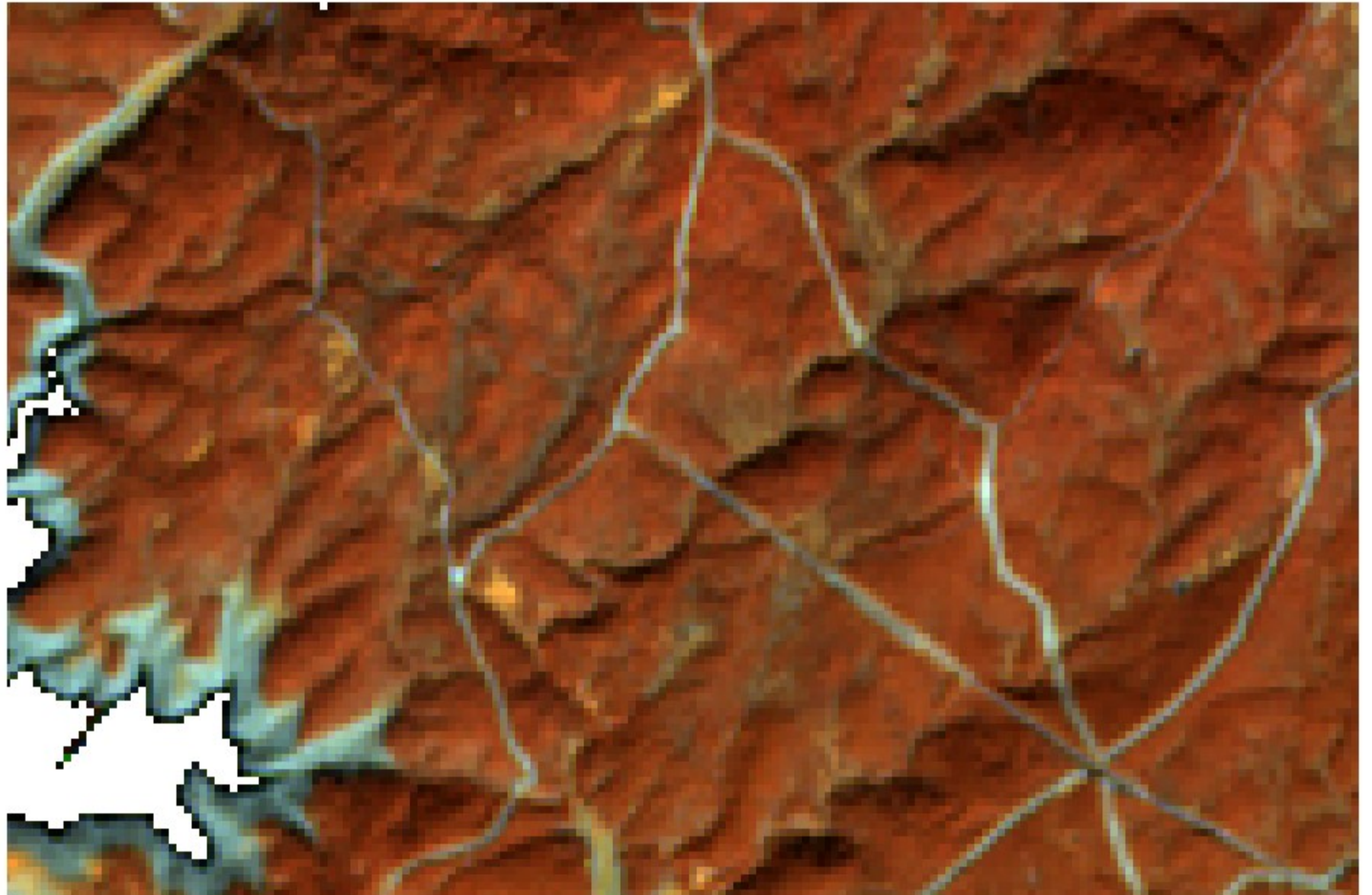
Use case: canopy health monitoring

- Objectives
 - Assess **atmospheric correction** for S2
 - *Critical for temporal monitoring*
 - Define methodology for **validation purposes**
 - Develop methodology that takes into account architecture and background effects
 - Typical in **heterogeneous pine forest**

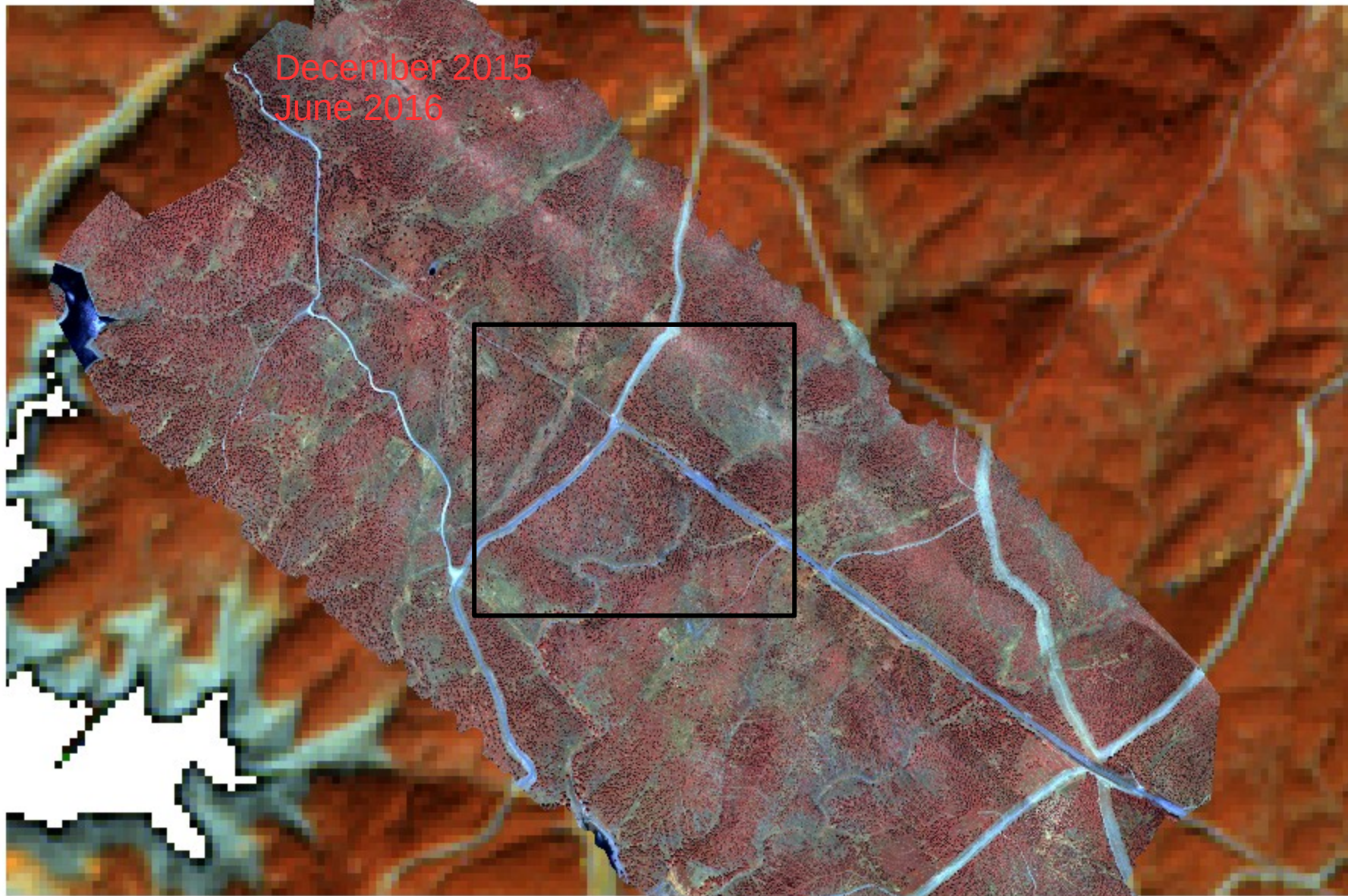
Background

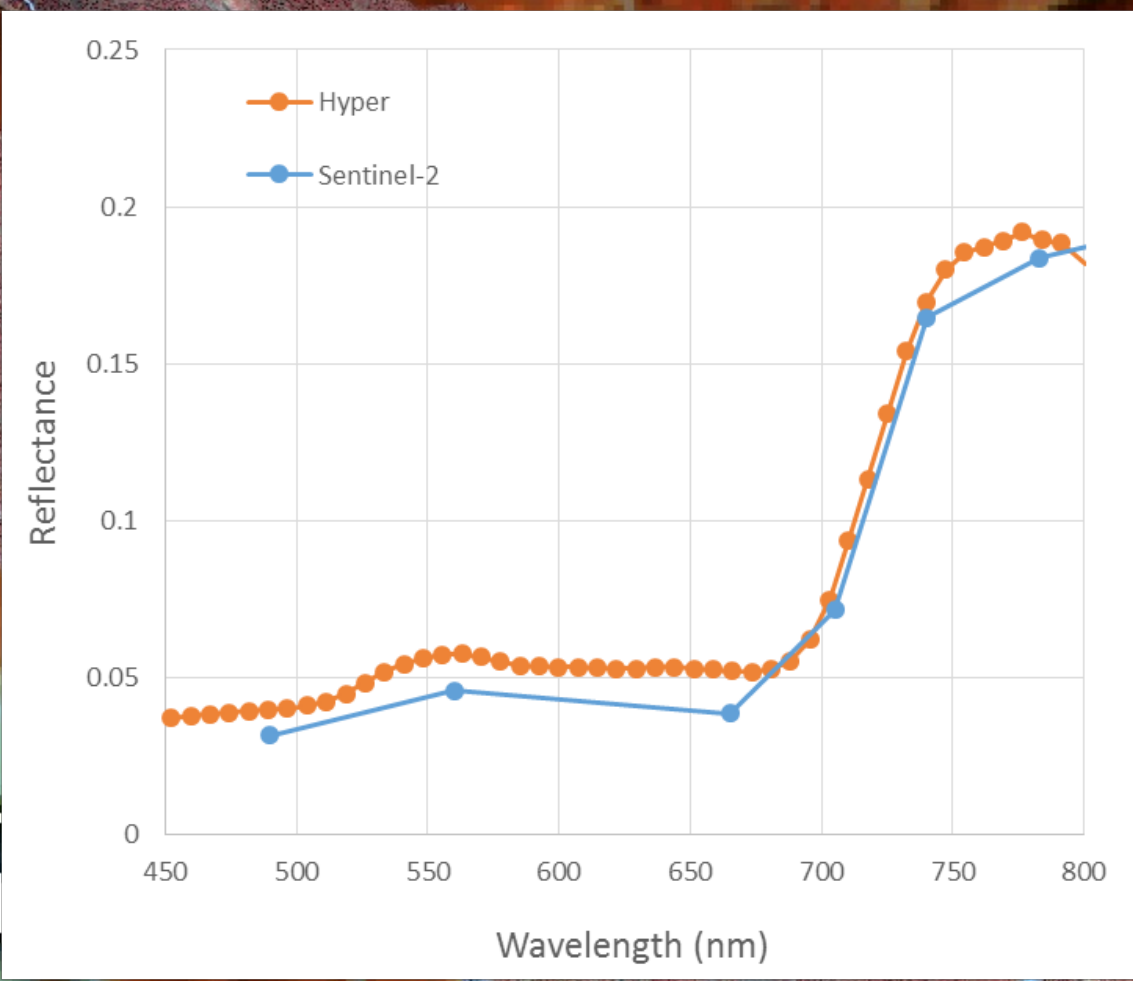
Chlorophyll & LAI retrieval is feasible using spectral bands in the visible + red edge region + NIR



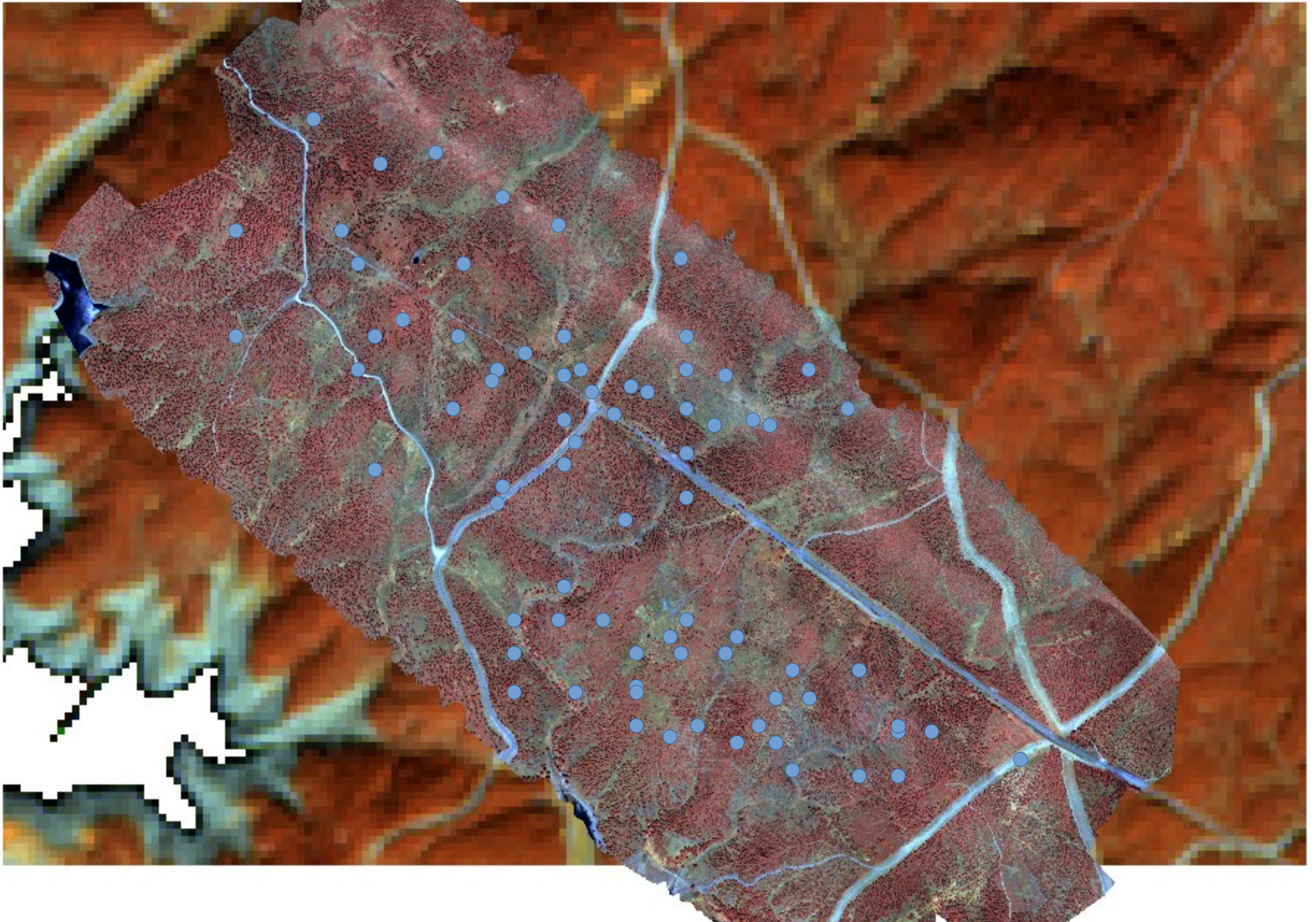


December 2015
June 2016





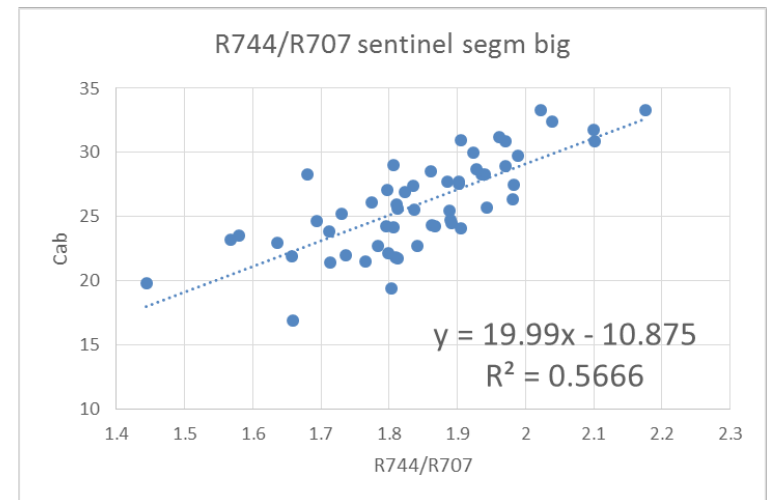
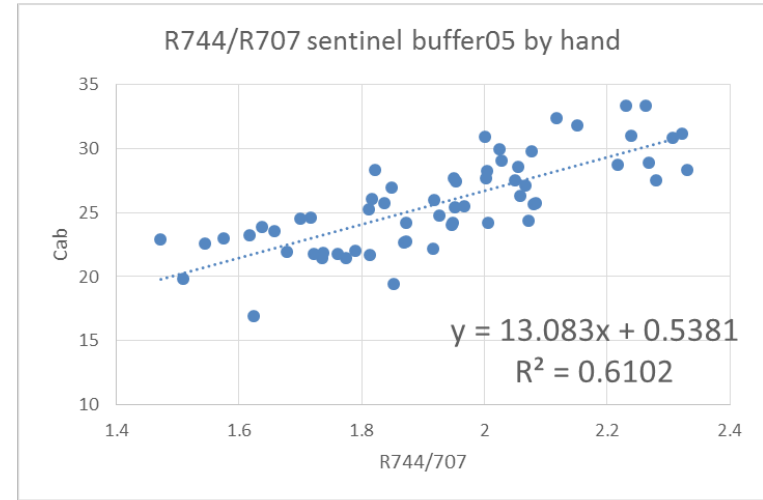
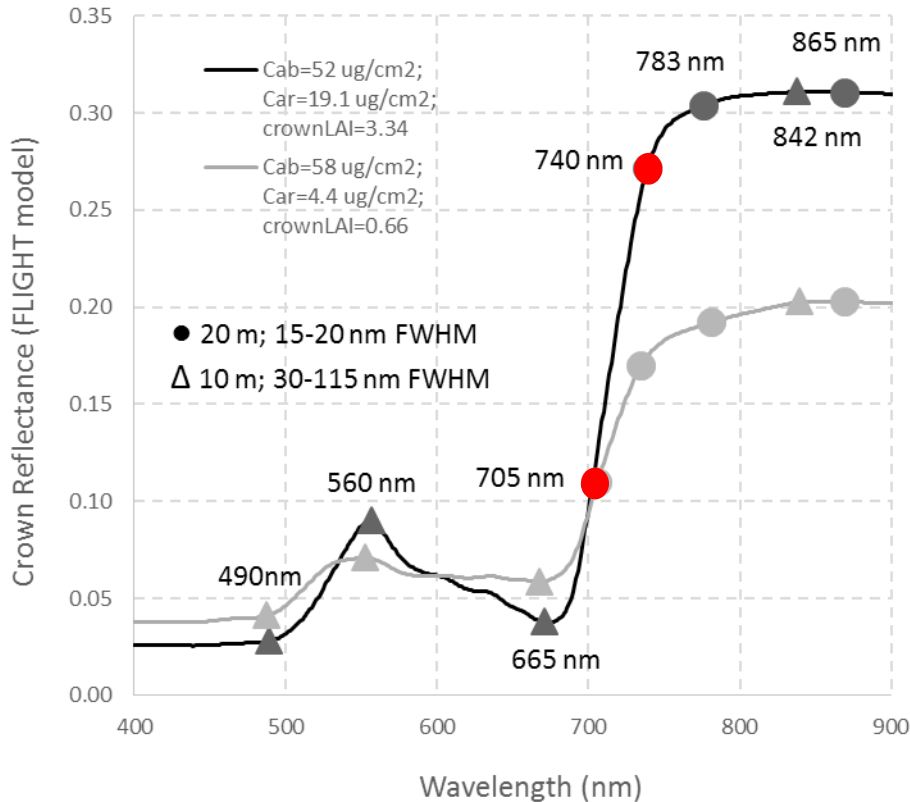
Field data: Cab and LAI for 75 trees



Cab retrieval (S2 simulated)

Sentinel-2 related indices: CI

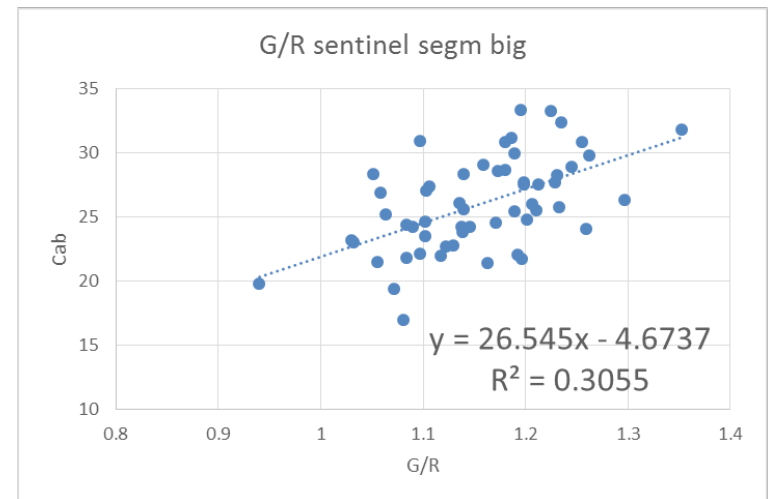
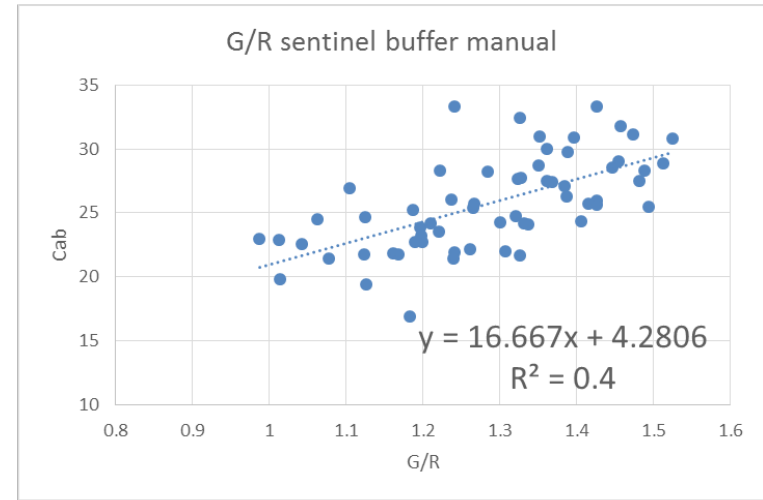
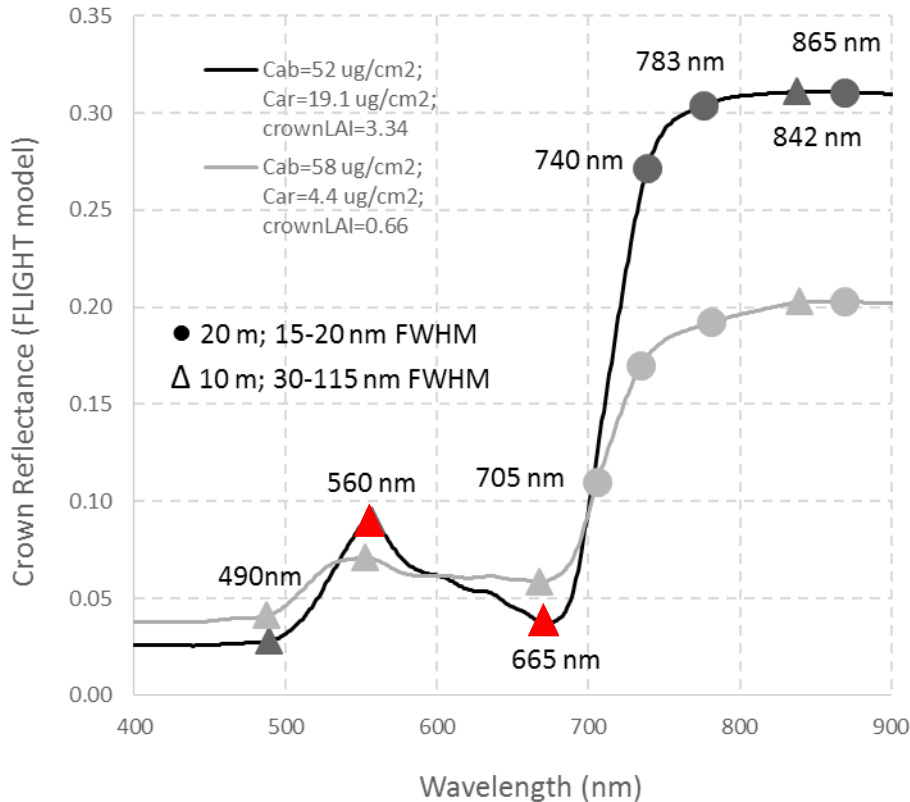
Sentinel-2 bands



Cab retrieval (S2 simulated)

Sentinel-2 related indices: G/R

Sentinel-2 bands



Approach

- Assessment of the **Sentinel-2 time series** between December & June hyperspectral flights
- To develop a methodology that takes into account the **architecture & background effects** in a typical heterogeneous pine forest
- To define a methodology for validation purpose
 - Use **hyperspectral coverage for validation**
- Link to Big Data project
 - Wall to wall airborne imagery at large scale
 - Upscale to S2 imagery and process large coverage at fine temporal dimension
 - discover new correlations S2 data – canopy health

Conclusions

- Big Data project started in 2015 (joined in Feb 2016)
- Storage infrastructure: CERN technology (EOS)
 - 1.5 PB (majority S1 and S2)
- Processing infrastructure: 600 cores
 - HTCondor
 - Docker
- Processing library (in development)
 - Jupyter iPython notebook (lab soon)
 - Based on in house expertise
- 10 use cases
 - Assessment of S2 for canopy health monitoring