

Sentinel-2 Agriculture

Design Definition File

Algorithm Theoretical Basis Document for L3 monthly composite product



Milestone	Milestone 2
Version	1.2
Authors	CESBIO - Olivier Hagolle, Mohamed Kadiri, David Morin

© UCL-Geomatics 2015

This document is the property of the Sen2-Agri partnership, no part of it shall be reproduced or transmitted without the express prior written authorisation of UCL-Geomatics (Belgium).

Contents

1	Introduction	2
2	Monthly synthesis product	2
3	Monthly synthesis processor	2
3.1	Pre-processing	3
3.1.1	Directional model function	3
3.1.2	Directional Correction	4
3.1.3	Resampling at S2 Resolution	4
3.2	Synthesis Update with new product	5
3.2.1	Update synthesis	5
3.2.2	Compute weights	7
4	Annex 1 : Table of correspondance between LANDSAT 8 and Sentinel-2 bands	10
5	Annex 2 : directional model	10

1 Introduction

This document provides the detailed processing model of a Monthly Synthesis processor, that delivers L3A products. Level 3A products provide the weighted average of the cloud free pixels observed by Sentinel-2 every month. This weighted average can be based on a synthesis duration which is longer than one month.

This document explains how the Monthly Synthesis processor should be defined. The way the Monthly Synthesis processor is launched is described in the Sen2Agri System specification (DRxx). The System Specification addresses how to do the selection of the L2A products, and how long they are awaited if turn out not to be available. An important parameter of the syntheses is the composite duration, however this parameter is handled by the Sen2Agri System and not by the L3A processor.

In order to limit the amount of L2A products to keep on line in the disk system, the L3A processor has been defined as a recurrent operation. When the first L2A Product necessary to produce the L3A product becomes available, the L3A product is initialised. It is then updated each time a new L2A product belonging to the list of L2A products to be processed is obtained.

2 Monthly synthesis product

The monthly synthesis product contains the following information :

- ◇ The weighted average surface reflectance for all Sentinel-2 bands available in the L2A products, at their original resolution (10m or 20m). The Sentinel-2 bands at 60m resolution are not provided in the L2A product, as these bands are useful for atmospheric correction but not aimed at observing the surface.
- ◇ Flags:
 - * one byte per pixel, 10m resolution
 - * values:
 - ★ no-data: pixel was never observed during compositing period
 - ★ cloud: pixel is always cloudy or within cloud shadow
 - ★ snow: pixel is always covered by snow for the available cloud free observations
 - ★ water: pixel is always covered by water for the available cloud free observations
 - ★ land: pixel was free from cloud, snow and water at least once during composite period
- ◇ Dates:
 - * a weighted average of dates used in the synthesis is provided for each pixel
 - * to be provided only at 20m resolution
- ◇ number of valid observations over the period

A table of the available dates is provided in the L3A metadata file

3 Monthly synthesis processor

For a given tile, the monthly synthesis processor works in 3 steps:

- ◇ a pre-processing step, which for each date transforms the L2A in a L2A* product
 - * to normalize directional effects for Sentinel-2 (A & B)
 - * to resample LANDSAT 8 to the same resolution as the corresponding Sentinel-2 band.
- ◇ a processing step:
 - * the synthesis product is a recurrent process that updates the product date after date
 - * the processor is run each time a new L2A* is available
 - * it uses as input a L2A* and the current L3A product
 - ★ if L3A product is not available, it is created

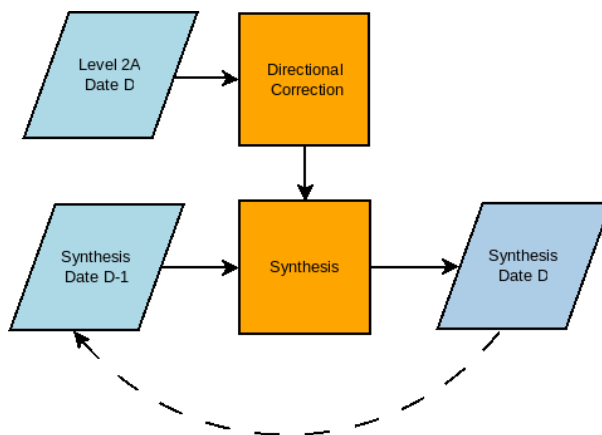


Figure 1: Diagram of synthesis processor

3.1 Pre-processing

This part is a pre-processor to be executed before the synthesis product processing. It uses a Level 2A as input, and delivers a pre-processed Level 2A referred hereafter as L2A*.

- ◊ For Sentinel-2, L2A* provides reflectances after directional correction (section ??)
- ◊ As LANDSAT-8 does not provide the viewing angles (yet?) for each pixel in its L1T product, it is not possible to apply the directional correction to LANDSAT 8. For LANDSAT 8, L2A* are L2A resampled to the resolution of the corresponding band of Sentinel-2 (see ??).

In both cases, the L2A* metadata and masks are identical to the L2A metadata and masks with only the resolution information changed in the case of LANDSAT-8.

The directional model used for the proposed correction derives from the following references :

- ◊ F. Maignan, F.-M. Breon, and R. Lacaze, “Bidirectional reflectance of Earth targets: Evaluation of analytical models using a large set of spaceborne measurements with emphasis on the Hot Spot,” Remote Sensing of Environment, vol. 90, no. 2, pp. 210–220, 2004.
- ◊ F.-M. Bréon and E. Vermote, “Correction of MODIS surface reflectance time series for BRDF effects,” Remote Sensing of Environment, vol. 125, pp. 1–9, Oct. 2012.

This work only provides the coefficients for the red and NIR bands of Sentinel-2. In order to get a bidirectional model for each Sentinel-2 band, a preliminary work at CESBIO will be necessary, based on real Sentinel-2 data, although a preliminary configuration may be provided for preliminary tests.

3.1.1 Directional model function

Variables and parameters

Input variable	role
b	spectral band
NDVI	NDVI
$(\theta_s, \theta_v, \delta\phi)$	Solar zenith Angle, Viewing zenith Angle, Relative Azimuth

Output variable	role
=mod-dir=	value of directional model for band b and for the input angles

parameter	role	default value
V0, V1	Coefficient for the volume scattering function, as a function of spectral band	
R0, R1	Coefficient for the roughness scattering function, as a function of spectral band	

Pseudocode

- ◇ For each spectral band b
 - * $\text{mod-dir}(\theta_s, \theta_v, \delta\phi, b) = 1 + (V_0(b) + V_1(b) * \text{NDVI}) \cdot F_V(\theta_s, \theta_v, \delta\phi) + (R_0(b) + R_1(b) * \text{NDVI}) \cdot F_R(\theta_s, \theta_v, \delta\phi)$
 - * a python code to compute F_V and F_R is provided in Annex, chapter ??

3.1.2 Directional Correction

The directional correction consists in applying a model to each pixel, except of this pixel is classified as cloud, cloud shadow, water or snow. The model depends on the sun and observation angles and on the NDVI of the pixel.

Variables and parameters

Input variable	role
ρ	reflectance for each pixel and each spectral band
Angles from L2A Image ($\theta_s, \theta_v, \delta\phi$)	Solar zenith Angle, Viewing zenith Angle, Relative Azimuth
Cloud-Shadow Mask	Mask of clouds and shadows
Water Mask	Mask of water pixels
Snow Mask	Mask of snow pixels

Pseudocode

- ◇ Compute NDVI at 10m resolution => $\text{NDVI}_{10\text{m}}$
- ◇ Resample it without aliasing to 20 m => $\text{NDVI}_{20\text{m}}$
- ◇ Compute ($\theta_s, \theta_v, \delta\phi$) for each pixel and each band according to Sentinel-2 L1C product Users Manual. /The viewing angles are provided as a grid per Sentinel-2 spectral band and per detector (there are 12 detectors per band). Sentinel-2 metadata also provide a map of the detectors which were used to observe each pixel. The method consist in finding the detector corresponding to the pixel to process /
- ◇ Apply directional correction to each pixel except water and snow pixels, using directional model (section ??)
- ◇ For each pixel and each band :
 - * if pixel is not cloud free, shadow free, water and snow free
 - * $\rho(\theta_s, 0, 0) = \rho(\theta_s, \theta_v, \delta\phi) * \frac{\text{mod-dir}(\theta_s, 0, 0)}{\text{mod-dir}(\theta_s, \theta_v, \delta\phi)}$

3.1.3 Resampling at S2 Resolution

Variables and parameters

Input variable	role
S2-Bands(L8-bands)	S2-Band corresponding to every L8 band (see annex ??)
$\text{Res}_{S2}(L8 - \text{bands})$	Resolution of each S2-band
ρ	reflectance for each pixel and each L8 spectral band

Output variable	role
ρ^*	reflectance value for each pixel, for each L8 spectral band with a corresponding S2 spectral band
Cloud-Shadow Mask	Mask of clouds and shadows
Water Mask	Mask of water pixels

Pseudocode

- ◇ for each l8 spectral band b_{L8}
 - * resample the L8 band to the resolution $\text{Res}(S2\text{-bands}(b_{L8}))$
- ◇ resample cloud-shadow mask and snow mask at 10 and 20m resolution

3.2 Synthesis Update with new product

The Synthesis update is made of 3 successive steps :

- ◇ Resample cloud and snow masks to 10m resolution
- ◇ Compute individual weights for each pixel in the image (??)
- ◇ Update synthesis

3.2.1 Update synthesis

This module updates the synthesis using the recurrent expression of a weighted average. A weighted average is only computed for land and water pixels. Cloud and Snow pixels are discarded from the weighted average, unless no land or water pixel was observed so far in the composite window. If it is the first iteration of the synthesis update, an initial synthesis with no-data pixels everywhere must be created. Land and Water are considered the same class to avoid issues with rice, which is flooded at the early stages.

The weighted average is computed with a weight counter per $S2_{\text{band}}$. This is not ideal as it adds a data volume equivalent to that of surface reflectance. If only Sentinel-2 (A and B) were used, it would be possible to use one counter for the 20m bands and a counter for the 10m bands (but that would not allow to account for possible failures in one of the bands of one of the satellites).

In order to reduce the data volume, the weighted average date and flag are provided for the red band only. It is assumed there will always be a red band and a blue band, which are required to choose the reflectances

Variables and parameters

input variable	role
S2-Bands(L8-bands)	S2-Band corresponding to every L8 band L2A* product
$date_N$	date from input product
ρ_{N^*}	reflectance value for each pixel, and each spectral band from input L2A* product
Cloud-Shadow Mask	Mask of clouds and shadows
Water Mask	Mask of water pixels
Snow Mask	Mask of snow pixels
L3A product	(if already created)
\overline{W}_{N-1}	Weight for each pixel obtained so far
\overline{date}_{N-1}	weighted average date for L3A product so far
$N-1$	weighted average reflectance value so far, for each pixel, and each spectral band
$flag_{N-1}$	status of each L3A pixel : cloud, water , snow

Output variable	role
L3A product	
\overline{W}_N	Weight counter for each pixel and for each band
\overline{date}_N	weighted average date for L3A product so far
N	weighted average reflectance value so far, for each pixel, and each spectral band
$flag_N$	status of each L3A pixel : cloud, water , snow

Pseudocode *initialization if necessary*

- ◇ if L3A product does not exist
 - * for each pixel
 - ★ $flag_{N-1} = \text{no-data}$
 - ★ $\overline{date}_{N-1} = \text{no-data}$
 - ★ for each S2 band
 - $\overline{\rho}_{N-1} = \text{no-data}$
 - $\overline{W}_{N-1} = 0$

update with current product

- ◇ compute weights for L2A product of date N => w_N per band and pixel
- ◇ for resolutions res in (10m,20m) :
 - * for each pixel
 - ★ if pixel not cloud/shadow, snow, water (*nominal case*)
 - $\text{flag}_N = \text{land}$
 - for each S2 band with **res** resolution
 - if band is available \ *in the case of LANDSAT 8, some bands are not available\
 - $\bar{\rho}_N = \frac{W_{N-1} \cdot \bar{\rho}_{N-1} + w_N \cdot \rho_N}{W_{N-1} + w_N}$
 - $\overline{\text{date}}_N = \frac{W_{N-1} \cdot \overline{\text{date}}_{N-1} + w_N \cdot \text{date}_N}{W_{N-1} + w_N}$
 - $W_N = W_{N-1} + w_N$
 - else :
 - $\bar{\rho}_N = \rho_{n-1}$
 - $W_N = W_{N-1}$
 - if band is red
 - $\overline{\text{date}}_N = \overline{\text{date}}_{N-1}$
 - ★ else : *degraded case*
 - if pixel is snow or water **replace the reflectance value*
 - if pixel is water
 - $\text{flag}_N = \text{water}$
 - if pixel is snow
 - $\text{flag}_N = \text{snow}$
 - for each S2 band with **res** resolution
 - if band is available
 - if $W_{N-1} == 0$ **pixel never observed without cloud, water or snow*
 - $\bar{\rho}_N = \rho$
 - $W_N = 0$
 - if band is red
 - $\overline{\text{date}}_N = \text{date}_N$
 - else: **pixel already observed cloud free, keep the previous weighted average*
 - $\bar{\rho}_N = \bar{\rho}_{N-1}$
 - $W_N = W_{N-1}$
 - if band is red
 - $\overline{\text{date}}_N = \overline{\text{date}}_{N-1}$
 - $\text{flag}_N = \text{land}$
 - else: **band not available, keep previous values*
 - $\bar{\rho}_N = \bar{\rho}_{N-1}$
 - $W_N = W_{N-1}$
 - if composite pixel is cloud or shadow **pixel never observed cloud snow or water free*
 - if flag_{N-1} is no-data **replace no_{data} with cloud*
 - for each S2 band with **res** resolution
 - if band is available
 - $\bar{\rho}_N = \rho$
 - $W_N = 0$
 - if band is red
 - $\overline{\text{date}}_N = \text{date}_N$
 - $\text{flag}_N = \text{cloud}$
 - else
 - $\bar{\rho}_N = \text{no - data}$
 - $W_N = 0$
 - if flag_{N-1} is cloud or shadow **replace value only if new reflectance in the blue is smaller*
 - if $\rho(\text{blue}) <_{N-1}(\text{blue})$

- for each S2 band with **res** resolution
 - if band is available
 - $\bar{\rho}_N = \rho$
 - $W_N = 0$
 - if band is red
 - $\overline{date}_N = date_N$
 - $flag_N = \text{cloud}$
 - else **band is not available*
 - $\bar{\rho}_N = \rho_{n-1}$
 - $W_N = 0$
 - else: **blue value too high*
 - $flag_N = flag_{N-1}$
 - for each S2 band with **res** resolution
 - $\bar{\rho}_N = \rho_{n-1}$
 - $\overline{date}_N = \overline{date}_{N-1}$
 - $W_N = 0$

3.2.2 Compute weights

Weight on Sensors

The idea here is to be able to merge different sensors in the synthesis product (for instance, LANDSAT 8 and Sentinel-2 (A or B) in the same synthesis. As the information brought by each sensor is not equivalent, different weights may be used depending on the sensor.

Input/Output variables, parameters

| input | role |
|---------------------|-------------------------|
| current date sensor | LANDSAT 8 or Sentinel-2 |

| output | role |
|--------------|--|
| W_{Sensor} | Weight on sensor (one value per product) |

| parameter | role | default value |
|------------------|-----------------------|---------------|
| $W_{Sensor-S2A}$ | Weight for Sensor S2A | 1 |
| $W_{Sensor-S2B}$ | Weight for Sensor S2A | 1 |
| $W_{Sensor-L8}$ | Weight for Sensor L8 | 0.33 |

Pseudocode

- ◇ If current date sensor is Sentinel-2A
 - * $W_{Sensor} = W_{SensorS2A}$
- ◇ If current date sensor is Sentinel-2B
 - * $W_{Sensor} = W_{SensorS2B}$
- ◇ else
 - * $W_{Sensor} = W_{SensorL8}$

Weight on clouds

Input/Output variables, parameters

| input | role |
|------------|---|
| Cloud-mask | Cloud and shadows mask from L2A product at 20m resolution |

| output | role |
|-------------|--|
| W_{Cloud} | Weight on distance to cloud (one value per pixel and per resolution, 10m, 20m) |

| parameter | role | default value |
|------------------------|--|---------------|
| coarse resolution | coarse resolution for quicker convolution | 240m |
| sigma-large-cloud-dist | standard deviation of gaussian filter for distance to large clouds | 10 |
| sigma-small-cloud-dist | standard deviation of gaussian filter for distance to small clouds | 2 |

Pseudocode

- ◇ binarize cloud mask (0 if no cloud no shadow, 1 if either cloud or shadow)
- ◇ undersample to lower resolution (240m), with a bicubic resampling, without aliasing (radius must be 2 times $\text{resampling}_{\text{factor}}$)
- ◇ binarize resampled mask (1 if above 0.5, 0 if under 0.5)
- ◇ *compute distances at low resolution*
 - * $\text{Dist}_{\text{Large Cloud, Low Res}} = \text{gaussian-filter}(\text{binarized low res cloud mask, standard deviation} = \text{sigma}_{\text{large-cloud-dist}})$
 - * $\text{Dist}_{\text{Small Cloud, Low Res}} = \text{gaussian-filter}(\text{binarized low res cloud mask, standard deviation} = \text{sigma}_{\text{small-cloud-dist}})$
 - * For each resolution (10m, 20m):
 - ★ *resample at full resolution (10 and 20m)*
 - ★ $\text{Dist}_{\text{Large Cloud, Full Res}} = \text{bilinear-oversampling}(\text{Dist}_{\text{Large Cloud, Low Res}})$
 - ★ $\text{Dist}_{\text{Small Cloud, Full Res}} = \text{bilinear-oversampling}(\text{Dist}_{\text{Large Cloud, Low Res}})$
 - ★ for each pixel :
 - *compute weight*
 - $W_{\text{Cloud}} = (1 - \text{dist}_{\text{Large Cloud}}) * (1 - \text{dist}_{\text{Small Cloud}})$

Weight on Date

The idea here is to give more weight to the dates close to the central date of the synthesis time window, and less weight to the edges of the time window. However, large difference in weights would result in not taking into account the dates at the edges of the synthesis window, unless they are the only one available in the synthesis window. As a result, we decided to weight the images by 1 in the middle of the synthesis window, and by 0.5 at the edges.

Input/Output variables, parameters

| input variable | role |
|---------------------|------------------------------|
| date_{L2A} | L2A date, expressed in days |
| date_{L3A} | L3A date, expressed in days |
| Δ_{Max} | Half Synthesis period (Days) |

| output variable | role |
|-----------------|--|
| W_{date} | Weight on date (one value per product xxx) |

| parameter | role | default value |
|-----------|---|---------------|
| W_{Min} | Minimum weight at edge of synthesis time window | 0.5 |

Pseudocode

$abs()$ is the absolute value

$$W_{date} = 1 - \frac{abs(date_{L2A} - date_{L3A})}{\Delta_{Max}} \cdot (1 - W_{min})$$

Weight on AOT

Inputs, Outputs, Parameters

| | | |
|--------------|--|---------------|
| input | role | |
| AOT | AOT values from L2A product at 20m resolution | |
| output | role | |
| W_{AOT} | Weight on AOT (one value per pixel and per resolution, 10m, 20m). The higher the AOT, the Lower the weight | |
| parameter | role | default value |
| W_{AOTMin} | min weight depending on AOT | |
| W_{AOTMax} | max weight depending on AOT | |
| AOT_{Max} | maximum value of the linear range for weights w.r.t AOT | |

Pseudocode

- ◇ resample AOT at 10m and 20m resolution
- ◇ for each resolution (10m, 20m):
 - * for each pixel :
 - ★ if (AOT(pix) <= max_{AOT}):
 - $W_{AOT} = W_{AOTMin} + (W_{AOTMax} - W_{AOTMin}) * (1 - \frac{AOT(l,p)}{AOT_{Max}})$
 - ★ else :
 - $W_{AOT} = W_{AOTMin}$

Compute total weight (product of individual weights)

Inputs, Outputs, Parameters

| | |
|--------|--|
| input | role |
| L2A | L2A product from date N |
| output | role |
| W_N | Weight for date N (one value per pixel and per resolution, 10m, 20m) |

Pseudocode

- ◇ Compute Weight on sensor (W_{sensor})
- ◇ Compute Weight on date (W_{date})
- ◇ Compute weight on AOT (W_{AOT})
- ◇ Compute Weight on clouds (W_{cloud})
- ◇ For each resolution (10m, 20m)
 - * For each pixel
 - ★ $W_N = W_{AOT} \cdot W_{date} \cdot W_{cloud} \cdot W_{sensor}$

4 Annex 1 : Table of correspondance between LANDSAT 8 and Sentinel-2 bands

| S2 Band | L8 band | wavelength (nm) | S2 Resol. (m) | provided in L3A |
|---------|---------|-----------------|---------------|-----------------|
| 1 | 1 | 450 | 60 | No |
| 2 | 2 | 490 | 10 | Yes |
| 3 | 3 | 560 | 10 | Yes |
| 4 | 4 | 670 | 10 | Yes |
| 5 | - | 705 | 20 | Yes |
| 6 | - | 740 | 20 | Yes |
| 7 | - | 780 | 20 | Yes |
| 8 | - | 820 | 10 | Yes |
| 8a | 5 | 865 | 20 | Yes |
| 9 | - | 940 | 60 | No |
| 10 | 6 | 1340 | 60 | No |
| 11 | 7 | 1650 | 20 | Yes |
| 12 | 8 | 2200 | 20 | Yes |

5 Annex 2 : directional model

#This python_code provides the Ross_Thick- Li Sparse directional model

```
import math as m
import numpy as np
import pylab as p
```

```
class angles:
```

```
    #constructor
```

```
    def __init__(self,theta_s,phi_s,theta_v,phi_v):
```

```
self.theta_s=theta_s*m.pi/180
```

```
self.theta_v=theta_v*m.pi/180
```

```
self.phi=(phi_s - phi_v)*m.pi/180
```

```
#if self.phi < 0 :
```

```
    #self.phi=self.phi + 2*m.pi
```

```
    #function delta
```

```
    def delta(self):
```

```
delta=m.sqrt(m.tan(self.theta_s)*m.tan(self.theta_s) + m.tan(self.theta_v)*m.tan(self.theta_v) - 2*m.tan
```

```
return delta
```

```
    #Air Mass
```

```
    def masse(self):
```

```
masse=1/m.cos(self.theta_s)+1/m.cos(self.theta_v)
```

```
#print masse
```

```
return masse
```

```
    #Function xsi
```

```
    def cos_xsi(self):
```

```
cos_xsi=m.cos(self.theta_s)*m.cos(self.theta_v) + m.sin(self.theta_s)*m.sin(self.theta_v)*m.cos(self.phi)
```

```
return cos_xsi
```

```
    def sin_xsi(self):
```

```
x=self.cos_xsi()
```

```
sin_xsi=m.sqrt(1 - x*x)
```

```
return sin_xsi
```

```
    def xsi(self):
```

```
xsi=m.acos(self.cos_xsi())
```

```
return xsi
```

```
    xsi_0=1.5*m.pi/180.
```

```

#Function t
def cos_t(self):
trig=m.tan(self.theta_s)*m.tan(self.theta_v)*m.sin(self.phi)
d=self.delta()
coef=1 #Coef=1 looks good, but Bréon et Vermote use Coef=2
cos_t=min(max(coef/self.masse()*m.sqrt(d*d + trig*trig),-1),1)
return cos_t

def sin_t(self):
x=self.cos_t()
sin_t=m.sqrt(1 - x*x)
return sin_t

def t(self):
#print 'theta_v %f cos_t %F'%(self.theta_v*180/m.pi,self.cos_t())
t=m.acos(self.cos_t())
return t

#function FV Ross_Thick, V stands for Volume
def FV(self):

FV=self.masse()/m.pi*(self.t() - self.sin_t()*self.cos_t() - m.pi) + (1+self.cos_xsi())/2/m.cos(self.th
return FV

#function FR Li-Sparse, R stands for Roughness
def FR(self):
A=1/(m.cos(self.theta_s)+m.cos(self.theta_v))

FR=4/3./m.pi*A*((m.pi/2-self.xsi()*self.cos_xsi()+self.sin_xsi()*(1+1/(1+self.xsi()/self.xsi_0))- 1./
return FR

def dir_mod(self,kV,kR) :
rho=1 +kV*self.FV() + kR*self.FR()
return rho
    
```