

---

# **Phymobat Documentation**

***Version 2.2***

**COMMANDRÉ Benjamin**

**sept. 19, 2018**



<b>1</b>	<b>Installation</b>	<b>3</b>
1.1	Téléchargement Phymobat . . . . .	3
1.2	Installation SIG open source . . . . .	3
1.2.1	GDAL . . . . .	3
1.2.2	Orfeo ToolBox (OTB) . . . . .	4
1.2.3	CURL . . . . .	4
1.3	Installation des modules python . . . . .	4
<b>2</b>	<b>Processus utilisées et tutoriels API</b>	<b>7</b>
2.1	Processus algorithmiques utilisées . . . . .	9
2.1.1	Traitement des images . . . . .	9
2.1.2	Traitements des échantillons . . . . .	14
2.1.3	Traitements de classification . . . . .	15
2.2	Tutoriels interface . . . . .	16
2.2.1	Interface Simplifiée . . . . .	16
2.2.2	Interface experte . . . . .	19
2.2.3	Interface du traitement des images . . . . .	19
2.2.4	Interface du traitement des échantillons . . . . .	23
2.2.5	Interface du traitement de classification . . . . .	25
<b>3</b>	<b>CarHab Phy MOBA package</b>	<b>35</b>
3.1	Image processing . . . . .	35
3.1.1	Archive . . . . .	35
3.1.2	Landsat image processing . . . . .	37
3.1.3	Texture index processing . . . . .	38
3.1.4	Slope processing . . . . .	40
3.1.5	Toolbox . . . . .	40
3.2	Vector processing . . . . .	40
3.2.1	Super vector . . . . .	40
3.2.2	Sample . . . . .	41
3.2.3	RPG . . . . .	42
3.2.4	Separability and threshold index . . . . .	43
3.2.5	Classification . . . . .	43
<b>4</b>	<b>Graphical User Interface package</b>	<b>45</b>
4.1	Interface command . . . . .	46
4.2	Control processing . . . . .	49

4.3	Interface element . . . . .	52
4.4	Popup about and warming . . . . .	52
<b>Index des modules Python</b>		<b>53</b>
<b>Index</b>		<b>55</b>

Chaîne de traitement du Fond blanc PHYsionomique des Milieux Ouverts de Basse Altitude par Télédétection (PHY-MOBAT).

---

**Note:** Outil développé sur Linux-Ubuntu 18.04.

---



### 1.1 Téléchargement Phymobat

Le programme Phymobat peut être cloné à partir du Gitlab Irstea à l'adresse suivante : [https://gitlab.irstea.fr/benjamin.commandre/PHYMOBAT\\_2018.git](https://gitlab.irstea.fr/benjamin.commandre/PHYMOBAT_2018.git)

La branche courante est develop

### 1.2 Installation SIG open source

La chaîne de traitement est construite sous divers outils open-source, comme GDAL et OGR. La démarche à suivre pour installer ces outils est indiquée ci-dessous uniquement sous Linux.

- Ajouter le dépôt ubuntuGIS-unstable

```
$ sudo add-apt-repository ppa:ubuntuGIS/ubuntuGIS-unstable
$ sudo apt-get update
```

#### 1.2.1 GDAL

```
$ sudo apt-get install gdal-bin
```

Pour vérifier que GDAL est bien installé, taper :

```
$ gdalinfo
```

Il est bien installé, si vous avez l'aide de gdalinfo qui s'affiche (Idem pour OGR) :

```
Usage: gdalinfo [--help-general] [-mm] [-stats] [-hist] [-nogcp] [-nomd]
        [-norat] [-noct] [-nofl] [-checksum] [-proj4]
        [-listmdd] [-mdd domain|`all`]*
        [-sd subdataset] datasetname
```

FAILURE: No datasource specified.

## 1.2.2 Orfeo ToolBox (OTB)

Commencer par installer les dépendances requises avec la commande suivante :

```
$ sudo aptitude install make cmake-curses-gui build-essential libtool automake git_
↳ libbz2-dev python-dev libboost-dev libboost-filesystem-dev libboost-serialization-
↳ dev libboost-system-dev zlib1g-dev libcurl4-gnutls-dev swig
```

Cloner la dernière version d'OTB présente sur le git à l'url : <https://gitlab.orfeo-toolbox.org/orfeotoolbox/otb>, par exemple :

```
$ git clone https://gitlab.orfeo-toolbox.org/orfeotoolbox/otb.git
```

Suivez les instructions du manuel d'utilisation d'OTB, <https://www.orfeo-toolbox.org/packages/OTBSoftwareGuide.pdf>, pour compiler en mode « SuperBuild ».

Cloner le module SimpleExtractionTools à l'url: <https://gitlab.irstea.fr/benjamin.commandre/SimpleExtractionToolsPhymobat> dans le répertoire OTB/Module/Remote puis compilez et installez-le.

## 1.2.3 CURL

Vérifier que le package CURL est installé, sur certaines versions de Ubuntu il ne l'est pas :

```
$ apt-cache search curl

i A curl - outil en ligne de commande pour_
↳ transférer des données avec une syntaxe URL
p curl:i386 - outil en ligne de commande pour_
↳ transférer des données avec une syntaxe URL
p curlftpfs - Système de fichiers pour accéder_
↳ à des hôtes FTP, basé sur FUSE et cURL
```

S'il ne l'est pas :

```
$ sudo apt-get install curl
```

## 1.3 Installation des modules python

La version de Python utilisée est la 3.6.5

Installer les deux modules Python gdal, scikit-learn, Shapely, numpy, lxml et PyQt5 depuis le dépôt du système de la façon suivante :

```
$ sudo aptitude install python3-gdal python3-sklearn python3-shapely python3-numpy_
↳ python3-lxml libqt5 pyqt5-dev-tools
```



Pour vérifier si les modules sont bien installé ou déjà installé, il suffit de taper dans la console Python (Par exemple pour le module gdal):

```
>>> from osgeo import gdal
>>>
```

Si ils ne sont pas encore installé vous aurez ces réponses. Dans ce cas il faudra les installer (voir section ci-dessus) :

```
>>> from osgeo import gdal
ImportError: cannot import name gdal
```

Ou

```
>>> from osgeo import gdal
ImportError: No module named gdal
```



---

### Processus utilisées et tutoriels API

---

Cette chaîne de traitement répond à un objectif du programme CarHab (Cartographie des Habitats naturels) à savoir : réaliser pour les milieux ouverts de basse altitude (MOBA) un « fond blanc physionomique », c'est-à-dire d'une carte physionomique de ces milieux à l'aide des techniques de télédétection.

Pour cela, une méthode de classification orientée-objet selon une approche experte a été développée. Le détail des étapes pour aboutir aux classifications du fond blanc physionomique est donné selon l'arbre de décision ci-dessous.

La première étape consiste à discriminer les végétations (semi-)naturelles des végétations culturales et éboulis. Pour ce faire, les données de télédétection multi-temporelles Landsat8 ou Sentinel2(+PHYMOBAT 3.0) sont utilisées. L'analyse de traitement d'image s'appuie sur l'hypothèse forte selon laquelle les cultures annuelles comprennent une étape de labour et se retrouvent donc au moins une fois en sol nu dans l'année. Cette analyse a pour objectif de calculer l'indice du minimum de NDVI sur la série temporelle équivalente à l'étape de labour (ie à une végétation non-naturelle et aux éboulis).

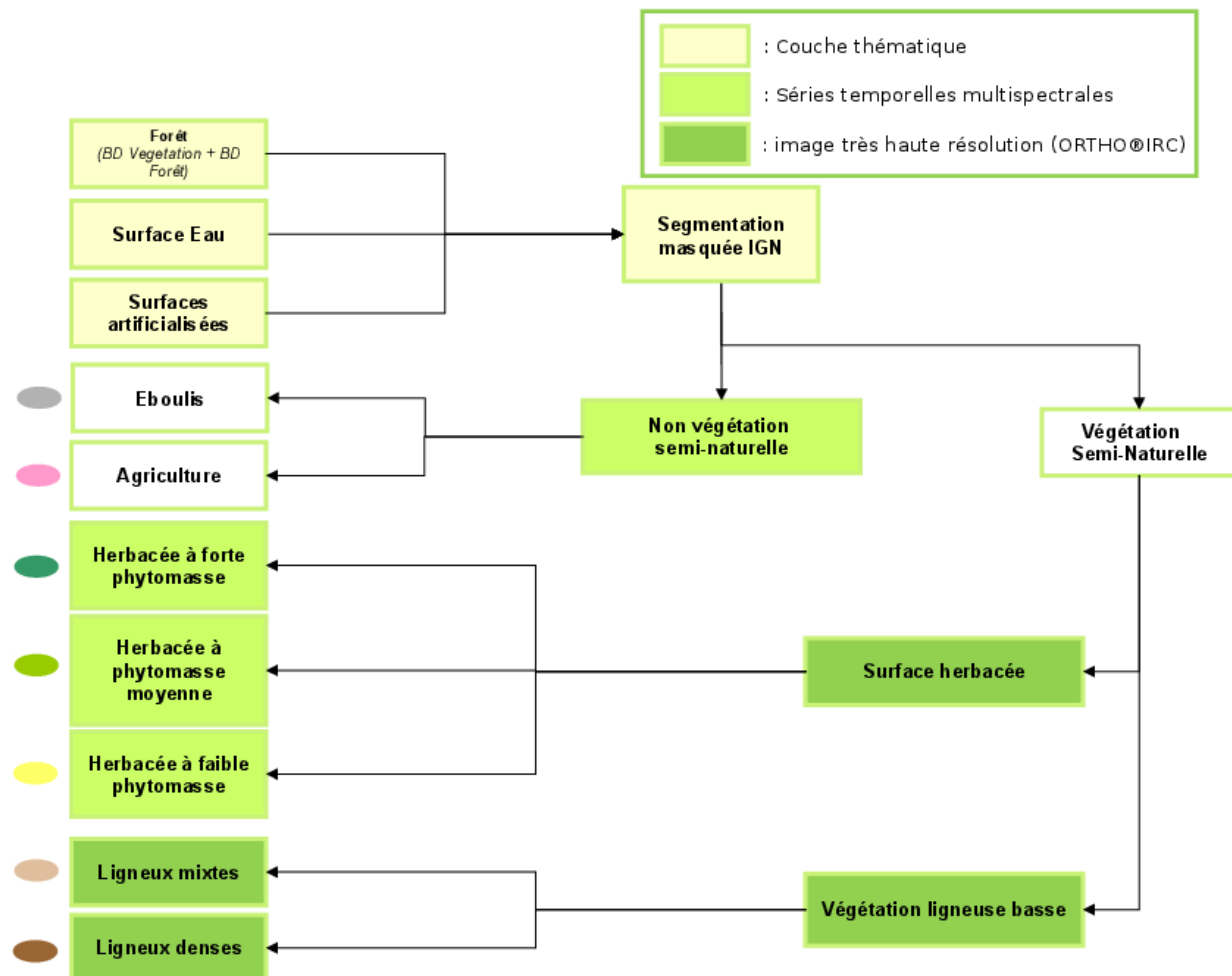
Le deuxième niveau se décompose en deux parties. Dans un premier temps, le modèle numérique de terrain est utilisé pour distinguer les éboulis (>30% en pente) des surfaces agricoles (+PHYMOBAT 1.1). Dans la seconde partie de cette étape, la télédétection permet de caractériser la végétation naturelle en termes de structure et de densité c'est-à-dire du point de vue physionomique. Cette analyse se fait par l'utilisation d'images de très hautes résolutions (ici les orthophotographies BD ORTHO © IRC). Il s'agit de distinguer les surfaces herbacées des végétations ligneuses basses à l'aide de l'indice de texture SFS'SD (Structural Feature Set Standard Deviation).

Pour le niveau suivante, les végétations ligneuses basses sont déclinées en deux niveaux de densités : mixtes (ouverts) et denses. Ils sont également caractérisés par leur structure et leur densité. La distinction entre ces deux classes se fait en calculant cette fois-ci l'indice d'Haralick, IDM (inverse Inverse Difference).

Une dernière phase consiste à extraire de l'information sur la production chlorophyllienne des zones herbacées. Cette étape utilise les séries temporelles. Elle se base sur le calcul de l'indice du maximum de NDVI dans l'année.

(Pour plus de renseignements, voir « Rapport méthodologique pour la cartographie physionomique des milieux ouverts de basse altitude par télédétection » - *Samuel Alleaume 2014*).

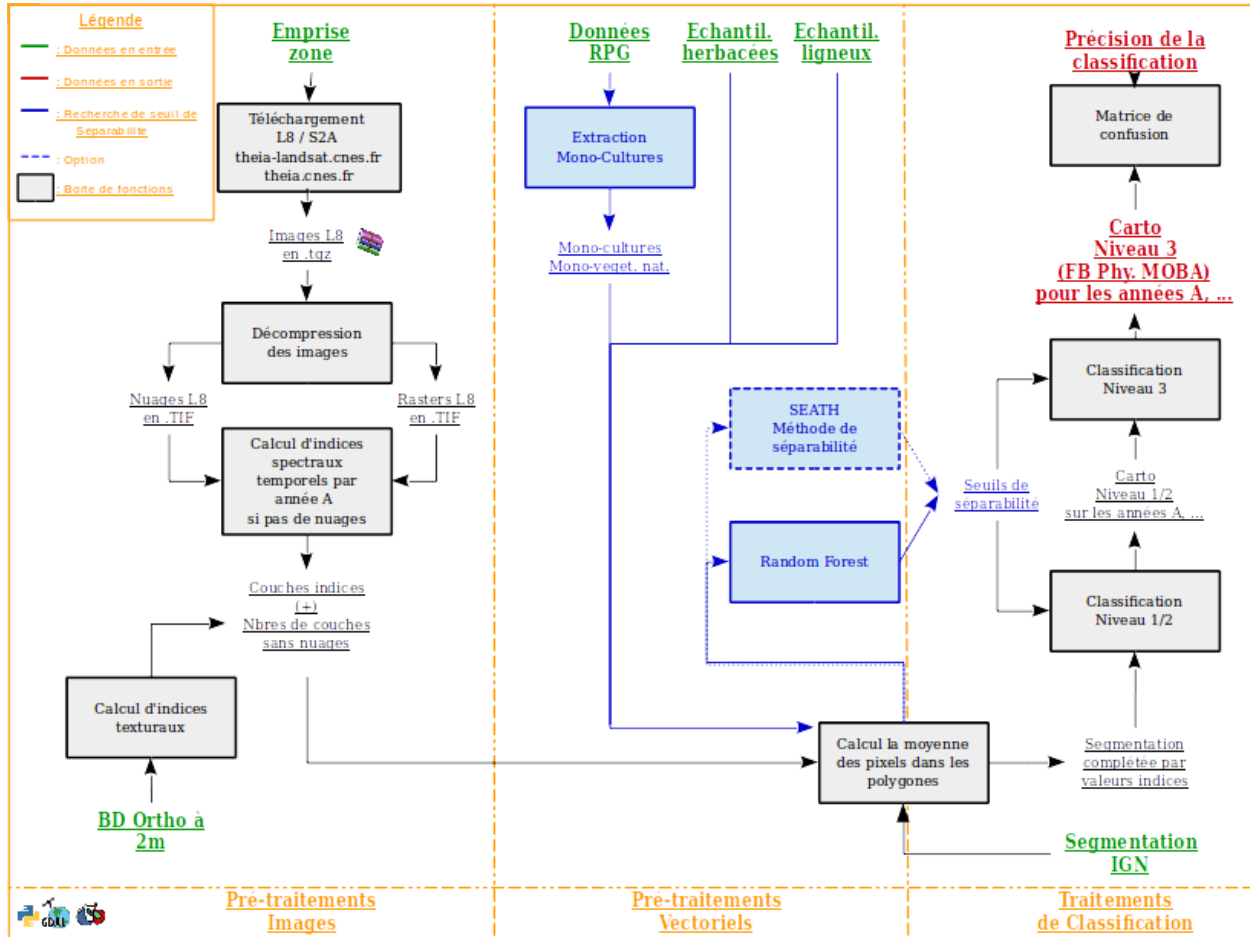
Une autre méthode avec un temps de calcul plus longue a été implémentée pour extraire la carte de végétation physionomique. Il s'agit de la méthode Random Forest (+PHYMOBAT 2.0). Elle permet de discriminer les mêmes classes sur un plus grand jeu de données.



## 2.1 Processus algorithmiques utilisées

Le Processus utilisé se décompose en trois parties :

- Traitements des images
- Traitements des échantillons
- Traitements de classification

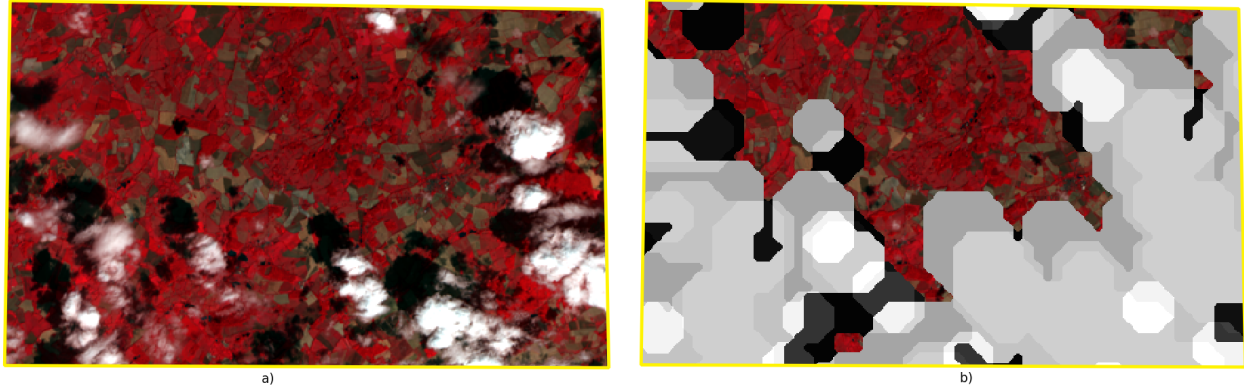


### 2.1.1 Traitement des images

Deux types d'images sont utilisés dans cette chaîne de traitement : les orthophotographies © de l'IGN en IRC (infra-rouge fausse couleur) et les images gratuites issues de la plate-forme **Theia** :

- **Landsat 8 OLI** (Operational Land Imager). En orbite depuis 2013, ces images ont une résolution spatiale de 30 mètres et 9 bandes spectrales (aérosols, bleu, vert, rouge, proche infra-rouge, moyen infra-rouge 1 et 2, panchromatique-15m, cirrus).
- **Sentinel 2A**. En orbite depuis Juin 2015, ces images ont 13 bandes spectrales et différentes résolutions spatiales 10, 20 et 30 mètres en fonction de ces mêmes bandes. PHYMOBAT 3.0 utilise que les bandes à 10 mètres soit le bleu, vert, rouge et proche infra-rouge.

A ces images, est joint un raster de détection des nuages, figure ci-dessous b (masque des nuages), utilisé pour sélection les images dans le processus.



Le traitement des images se décompose quant à lui en trois parties :

1. Listing et téléchargement des images sur la plate-forme Theia
2. Traitement des images téléchargées
3. Traitement des images très haute résolution spatiales (calcul des indices de textures)

### 1. Listing et téléchargements des images sur la plate-forme Theia

Le listing des images disponibles se fait à travers une base de données GeoJSON via une API serveur pour Landsat8 (resp. Sentinel2): [https://landsat-theia.cnes.fr/resto/api/collections/Landsat/search.json?lang=fr&\\_pretty=true&q=2013&box=4.5,43.16,5.8,43.5&maxRecord=500](https://landsat-theia.cnes.fr/resto/api/collections/Landsat/search.json?lang=fr&_pretty=true&q=2013&box=4.5,43.16,5.8,43.5&maxRecord=500) (resp. [https://theia.cnes.fr/resto/api/collections/Landsat/search.json?lang=fr&\\_pretty=true&completionDate=2015-12-31&box=4.5,43.16,5.8,43.5&maxRecord=500&startDate=2015-01-01](https://theia.cnes.fr/resto/api/collections/Landsat/search.json?lang=fr&_pretty=true&completionDate=2015-12-31&box=4.5,43.16,5.8,43.5&maxRecord=500&startDate=2015-01-01)).

Avec  $q=2013$  : Année des images disponibles

$startDate=2015-01-01$  : Date de début de la période choisie des images disponibles

$completionDate=2015-12-31$  : Date de fin de la période choisie des images disponibles

$box=4.5,43.16,5.8,43.5$  : Zone d'emprise en degrés décimaux

La fonction `Archive.Archive.listing()` utilise cette base. Elle remplace dans l'exemple « 2013 » par la période ou l'année entrée par un utilisateur. Ensuite, à l'aide du shapefile de la zone d'étude, la fonction convertie les coordonnées des extrémités du fichier vecteur en degrés décimaux pour les inclure dans l'url ci-dessus ( $box =$ ).

A l'issue du listing, le nom et l'identifiant des images sont stockés dans une liste. Le nom de l'image est utilisé pour nommer l'image après téléchargement et l'identifiant est utilisé pour le téléchargement `Archive.Archive.download_auto()` (+PHYMOBAT 1.1). Une bouton **Proxy** a été ajouté à PHYMOBAT 3.0:

```
## Source : https://github.com/olivierhagolle/theia_download
# Authentification
get_token='curl -k -s -X POST %s --data-urlencode "ident=%s" --data-urlencode "pass=%s'
↪ "%s/services/authenticate/>token.json"%(curl_proxy, user_theia, password_theia, _
↪ self.server)
os.system(get_token)
# .
# .
# .
# Téléchargement -> loop ... d in range(...)
```

(continues on next page)

(continued from previous page)

```
get_product='curl %s -o %s -k -H "Authorization: Bearer %s" %s/%s/collections/%s/%s/
↳download/?issuerId=theia'%(curl_proxy, self.list_archive[d][1], token, self.server,
↳self.resto, self._captor, self.list_archive[d][2])
os.system(get_product)
```

A la fin du téléchargement des images, `Archive.Archive.decompress()` va décompresser les archives Landsat 8 grâce au module `tarfile`. Pour les Sentinel 2A, les archives seront extraits à l'aide du module `zipfile` en sélectionnant que les bandes à 10m. En effet, les rasters Sentinel 2A sont fournis avec des bandes séparées qu'il faut « stacker ».

## 2. Traitements des images téléchargées

Toutes les images décompressées (images spectrales à 9 bandes pour L8 (à 4 bandes pour S2A) et masque de nuages) sont découpées `Toolbox.clip_raster()` en fonction de la zone d'étude. Le découpage est effectué en ligne de commande grâce au module python `subprocess` :

```
$ gdalwarp -dstnodata -10000 -q -cutline study_area.shp -crop_to_cutline -of GTiff_
↳raster.TIF Clip_raster.TIF
```

Ensuite une sélection des images est effectuée en fonction de la couverture nuageuse dans la zone d'étude. Pour cela, le processus regroupe `RasterSat_by_date.RasterSat_by_date.group_by_date()` et mosaïque d'abord les rasters par date `RasterSat_by_date.RasterSat_by_date.mosaic_by_date()`.

Le mosaïquage est réalisé en ligne de commande `RasterSat_by_date.RasterSat_by_date.vrt_translate_gdal()` en utilisant le format virtuel de `gdal`, `VRT` (*Virtual Dataset*) :

```
$ gdalbuildvrt -srcnodata -10000 dst_data.VRT raster1.TIF raster2.TIF
Input file size is 286, 467
0...10...20...30...40...50...60...70...80...90...100 - done.
$
$ gdal_translate -a_nodata -10000 dst_data.TIF dst_data.VRT
Input file size is 286, 467
0...10...20...30...40...50...60...70...80...90...100 - done.
```

La sélection est faite dans `RasterSat_by_date.RasterSat_by_date.pourc_cloud()`. Elle renvoie le pourcentage de couverture claire `cl` par mosaïque de la façon suivante :

- Extrait l'étendue de l'image en matrice :

```
mask_spec = np.in1d(data_spec[0], [-10000, math.isnan], invert=True)
```

- Extrait les pixels correspondant aux nuages :

```
mask_cloud = np.in1d(data_cloud, 0)
```

- Détermine les pixels de nuages par rapport à l'emprise de l'image :

```
cloud = np.choose(mask_cloud, (False, mask_spec))
```

- Détermine la somme de pixels en nuages et le pourcentage dans la zone d'emprise :

```
dist = np.sum(cloud)
nb0 = float(dist) / (np.sum(mask_spec))
```

Par défaut, le pourcentage de couverture nuageuse maximum accepté est de 40%.

Toutes les mosaïques ayant plus de 60% de pixels clair, passeront par les fonctions `RasterSat_by_date.RasterSat_by_date.calcul_ndvi()` (calcul de NDVI), `Toolbox.calc_serie_stats()` (calcul de minimum, maximum de ndvi et temporalité nuageuse) et `RasterSat_by_date.RasterSat_by_date.create_raster()`. Cette dernière fonction crée cinq rasters : minimum ndvi, maximum ndvi, std ndvi (écart-type), MaxMin ndvi (max ndvi - min ndvi) et un dernier raster qui correspond au nombre d'image utilisé par pixel clair (exemple sur l'image ci-dessous).

### 3. Traitements des images THRS

Le traitement des images THRS est effectué pour déterminer les ligneux et différents types de ligneux. Ligneux sont caractérisés par leur texture vis-à-vis des herbacés, et de leur type. Ces caractéristiques sont extraits à l'aide d'indices de textures issus de l'OTB via `Vhrs.Vhrs()`.

Deux indices ont été sélectionnés pour discriminer les classes par rapport aux ligneux :

- SFS'SD des indices de textures de Structural Feature Set Standard Deviation (Herbacés / Ligneux)

```
$ otbcli_SFSTextureExtraction -in raster.tif -channel 2 -parameters.spthre 50.0 -  
↪parameters.spthre 100 -out out_sfs.tif
```

- Inverse Difference Moment des indices d'Haralick (Ligneux mixtes / Ligneux denses)

```
$ otbcli_HaralickTextureExtraction -in raster.tif -channel 2 -parameters.xrad 3 -  
↪parameters.yrad 3 -texture simple -out out_haralick.tif
```

Pour extraire deux indices, il faut lancer les deux commandes ci-dessus qui calculent malgré tout 14 indices. Par conséquent, les traitements deviennent très long. Pour réduire ce temps de calcul, la chaîne de traitement utilise le multiprocessing. Il permet de lancer tous les traitements en même temps.

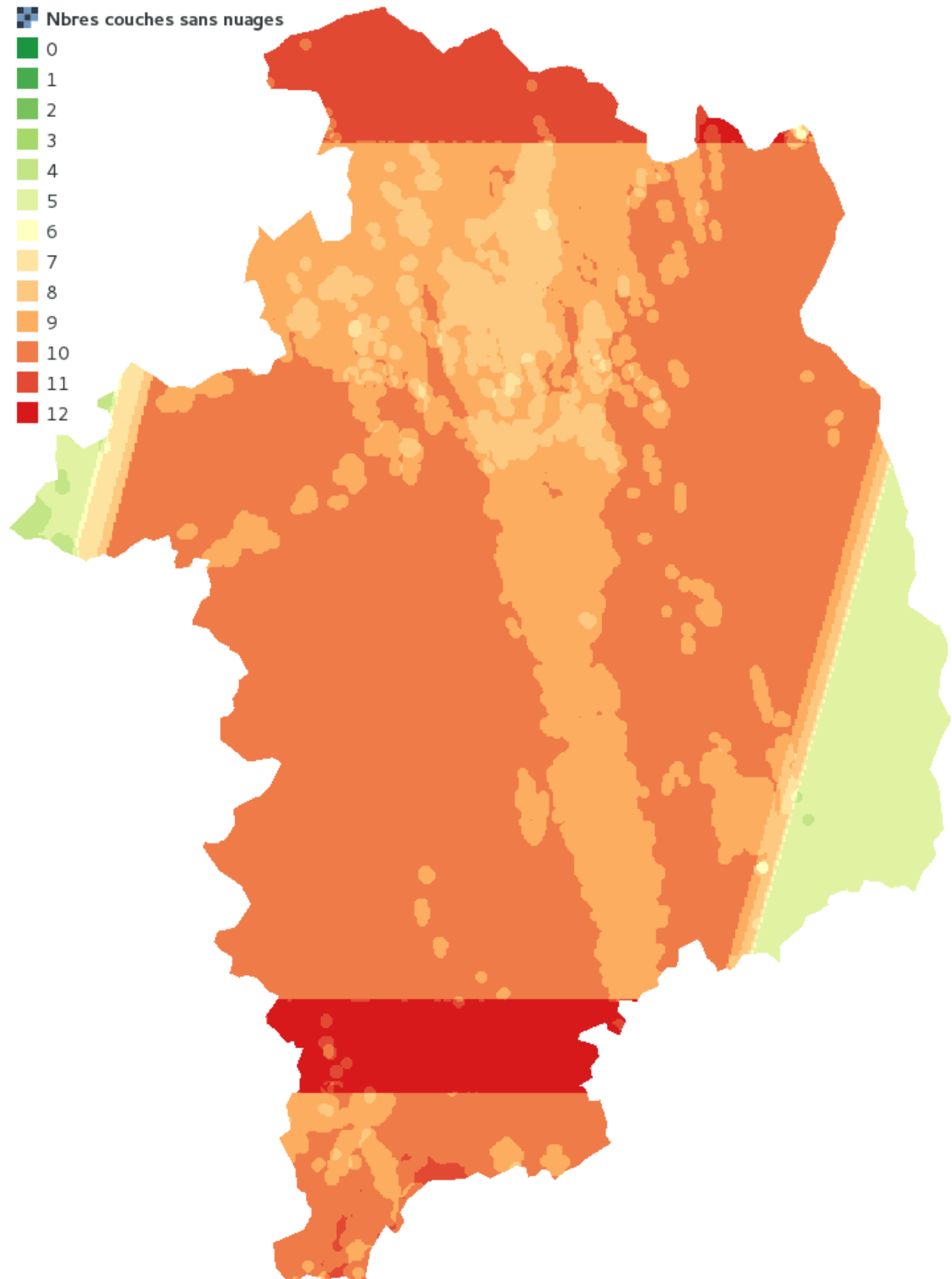
Le code Python associé au multiprocessing est le suivant :

```
>>> from multiprocessing import Process  
>>> p_sfs = Process(target=sfs_function)  
>>> p_har = Process(target=haralick_function)  
>>> # Lancement des traitements  
>>> p_sfs.start()  
>>> p_har.start()  
>>> # Attente de la fin des calculs  
>>> p_sfs.join()  
>>> p_har.join()
```

**Avertissement:** Pour utiliser le multiprocessing, il faut une machine avec minimum 12Go de mémoire vive. Sinon les traitements seront plus long que sans l'utilisation du multiprocessing!

**Note:** L'image THRS utilisée est l'orthophotographie © IGN. A la base c'est un maillage de plusieurs images carrées de 5km de côté. Ces images sont en 8bit à 50cm de résolution spatiale et au format compressé ECW (Enhanced Compression Wavelet). En dehors de la chaîne de traitement, un mosaïquage de ces images sur la zone d'étude doit être construit. Cette mosaïque doit être ré-échantillonnée à 2m car différentes études ont montré que l'image à 50cm apportait la même information. De plus, il y a un gain-temps non négligeable sur les calculs pour une image de plus basse résolution.





## 2.1.2 Traitements des échantillons

Il faut trois paires d'échantillons pour compléter l'arbre de décision défini plus haut. Un échantillon pour séparer : **Cultures / Végétation semi-naturelle, Herbacées / Ligneux et Ligneux mixtes / Ligneux denses.**

Les traitements des échantillons `Processing.Processing.i_sample()` est le même pour les trois, soit :

- En partant du principe que tous les échantillons soit dans un même shapefile, il faut indiquer le nom de la classe et le champ où il se trouve.

```
>>> kwargs['fieldname'] = self.fieldname_args[sple]
>>> kwargs['class'] = self.class_args[sple]
```

- Créer un shapefile par échantillon `Sample.Sample.create_sample()` de calibration et de validation puis réaliser une statistique zonale par polygone `Vector.Vector.zonal_stats()`.

```
>>> sample_rd[sple] = Sample(self.sample_name[sple/2], self.path_area, self.list_nb_
↳ sample[sple/2])
>>> sample_rd[sple].create_sample(**kwargs)
>>> sample_rd[sple].zonal_stats((self.raster_path[sple/2], self.list_band_
↳ outraster[sple/2]))
```

La création du shapefile est fait sur un certain nombre de polygone (chiffre indiqué par l'utilisateur) tiré aléatoirement.

- Et Extrait le modèle de distinction

### 1. Modèle Seath

A l'aide des valeurs déterminer par `Vector.Vector.zonal_stats()`, la fonction `Seath.Seath.separability_and_threshold()` détermine le seuil optimal (*SEaTH-A new tool for automated feature extraction in the context of object-based image analysis S. Nussbaum et al.*) pour discriminer les classes deux à deux issues de l'arbre de décision.

Pour l'instant, l'utilisation du RPG (Régistre Parcellaire Graphique) est indispensable comme échantillon de **Cultures**. Or le RPG possède des polygones poly-cultureux. Il se pourrait qu'un polygone renseigné soit ainsi blé, maïs et une prairie permanente. Par conséquent, ce polygones injecterait une erreur dans le calcul du seuil optimal puisque le polygone est un mélange de végétation non naturelle et semi-naturelle. Dans ce cas, `Rpg.Rpg()` a été mis en place pour créer des échantillons mono-cultureux de cultures et de prairies permanentes.

### 2. Moldèle Random Forest (RF)

Le RF quant à lui stocke toutes les bandes de textures contrairement à la l'utilisation méthode Seath dite experte. Cette méthode a été mise en place à l'aide du module Python `sklearn` avec un export des indices les plus significatifs et de l'arbre de décision généré :

```
# Build a forest of trees from the samples
self.rf = self.rf.fit(X_rf, y_rf)

# Print in a file feature important
importance = self.rf.feature_importances_
importance = [(importance[x],x+1) for x in range(len(importance))]
importance.sort()

file_feat_import = os.path.dirname(str(self.raster_path[0])) + '/Feature_important_RF.
↳ ft'
if os.path.exists(file_feat_import):
```

(continues on next page)

(continued from previous page)

```

os.remove(file_feat_import)
f_out = open(file_feat_import, "wb")
f_out.write(str(importance))
# Close the output file
f_out.close()

# Print in a file decision tree
file_decisiontree = os.path.dirname(str(self.raster_path[0])) + '/Decision_tree.dot'
if os.path.exists(file_decisiontree):
    os.remove(file_decisiontree)

tree_in_forest = self.rf.estimators_[499]
with open(file_decisiontree, 'w') as my_file:
    my_file = tree.export_graphviz(tree_in_forest, out_file = my_file)

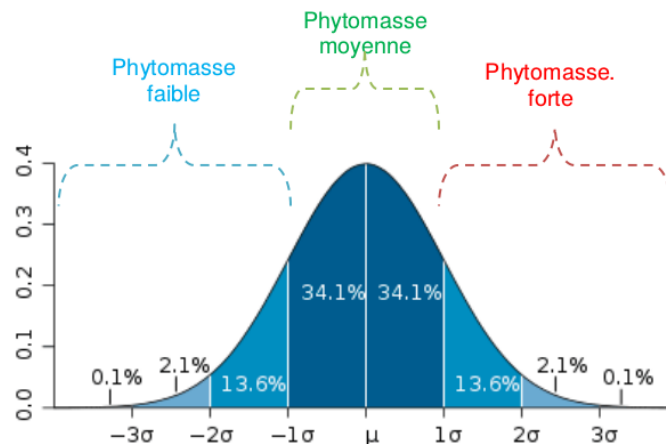
```

Le temps de calcul de ce modèle est plus long car le traitement de `Vector.Vector.zonal_stats()` se fait sur 17 images. Par contre, il a l'avantage d'être plus précis.

### 2.1.3 Traitements de classification

La classification est réalisée sur la segmentation issue de la l'orthophotographie © IGN. A chaque polygone, est affecté une valeur moyenne `Vector.Vector.zonal_stats()` par image. Les statistiques zonales sur les rasters à THRS sont très long (Plus long pour la méthode RF). Le multiprocessing est, une nouvelle fois, utilisé par ici pour accélérer le gain de temps de calcul.

`Segmentation.Segmentation.compute_biomass_density()` extrait la distribution de la densité de ligneux et de phytomasse.



`Segmentation.Segmentation.decision_tree()` (resp. `rf.predict()`) classe les polygones en fonction des seuils optimaux pré-déterminés `Seath.Seath.separability_and_threshold()` (resp. `Processing.Processing.i_sample_rf()`) et des valeurs zonales par raster.

`Segmentation.Segmentation.create_cartography()` va créer le shapefile final qui représentera le **FB** physiologique MOBA.

**Note:** Le document final contient une colonne RPG qui correspond à la donnée du RPG (îlots de culture) pour un

polygone de la segmentation inclut à 85%.

---

```
Processing.Processing.i_classifier_rf()
```

```
# Rasterize RPG shapefile to complete the final shapefile
opt = {}
opt['Remove'] = 1
rpg_tif = Vector(self.sample_name[0], self.path_area, **opt)
#         if not os.path.exists(str(rpg_tif.vector_used[:-3]+'TIF')):
kwargs['choice_nb_b'] = 1
out_carto.stats_rpg_tif = out_carto.zonal_stats_pp(rpg_tif.layer_rasterization(self.
↳path_ortho, 'CODE_GROUP', **kwargs))
```

```
Segmentation.Segmentation.create_cartography()
```

```
if pourc_inter >= 85:
    recouv_crops_RPG = self.stats_rpg_tif[in_feature.GetFID()][ 'Maj_count' ]
```

## 2.2 Tutoriels interface

Il existe 2 interfaces : une interface simplifiée et une interface experte.

**L'interface simplifiée** `ui_PHYMOBATs_tab.Ui_PHYMOBAT()` comme son nom l'indique est assez simple, elle est représentée sur une fenêtre et est très limitée au niveau des choix à faire (selection des types d'images, méthode de classification, choix des champs pour la classification finale, etc ...)

**L'interface experte** `ui_PHYMOBATE_tab.Ui_PHYMOBAT()` est plus complexe. Elle est composée de 3 sous-onglets qui permettent de dissocier les traitements (pré-traitements images, traitements vecteur et classification) mais également de choisir les méthodes de classification et le types d'images à utiliser.

Le passage d'une interface à une autre se fait à travers le sur-onglet Mode (référence 1 sur la figure ci-dessous). Il y a 3 sur-onglets : Menu, Aide et Mode

- **Menu** : cet onglet est composé quant à lui de 3 fonctions (Ouvrir, Sauver, Quitter). Les fonctions Ouvrir `PHYMOBAT.PHYMOBAT.open_backup()` et Sauver `PHYMOBAT.PHYMOBAT.save_backup()` sont utilisées pour charger ou sauvegarder dans un fichier .xml les paramètres entrés dans chaque case de l'application. La fonction Quitter `PHYMOBAT.PHYMOBAT.close_button()`, ferme l'application.
- **Aide** : Aide de PHYMOBAT et A propos de PHYMOBAT. La fonction Aide de PHYMOBAT `PHYMOBAT.PHYMOBAT.help_tools()` ouvre une page HTML décrivant les méthodes de télédétection et les scripts utilisés (Pas à jour). La fonction A propos de PHYMOBAT `PHYMOBAT.MyPopup_about()` rappelle l'objectif de l'application et la licence utilisée.
- **Mode** : Mode Simplifié `PHYMOBAT.PHYMOBAT.mode_simpli()` et Mode Expert `PHYMOBAT.PHYMOBAT.mode_expert()`. Ces deux modes sont basés sur le même algorithme. Par conséquent, bien paramétrés, le résultat de la carte finale est le même.

### 2.2.1 Interface Simplifiée

C'est l'interface par défaut. Elle s'ouvre en tapant dans la console :

```
$ python3 PHYMOBAT.py
```

Elle est présenté sous cette forme :

**PHYMOE** Menu Aide Mode

PHYMOBATS 3.0

Chemin du dossier principal

Période ou année des images Exemple pour 1 periode : AAAA-MM-DD,AAAA-MM-DD

Emprise de la zone (shp)

Identifiant Theia

Utilisateur : Mot de passe : Proxy

BD Alti (Facutatif)

Images THRS

Segmentation (shp)

Echantillons RPG (Végétation non naturelle / Semi-naturelle)

Non naturelle Semi-naturelle

Champs

Classes Nbre de polygones 100

Echantillons Herbacés / Ligneux

Herbacés Ligneux

Champs

Classes Nbre de polygones 20

Echantillons Ligneux denses / mixtes

Ligneux denses Ligneux mixtes

Champs

Classes Nbre de polygones 20

Fichier de sortie (shp)

Multi-processing

Close OK

1 - Sur-onglets : **Menu, Aide et Mode**

2 - **Chemin du dossier principal** : Chemin d'accès au dossier où sont stockées toutes les données en entrée mais également toutes les données en sortie de l'application.

3 - **Période ou année des images** : Intervalles de dates des images à télécharger et à traiter.

L'intervalle doit être sous cette forme *AAAA-MM-DD,AAAA-MM-DD* où A -> Année (2015), M -> Mois (05) et D -> Jour (25). Pour plusieurs intervalles, par exemple deux (Séparer par un point virgule) : *AAAA-MM-DD,AAAA-MM-DD;AAAA-MM-DD,AAAA-MM-DD*. Pour une année complète : *AAAA*

4 - **Emprise de la zone** : Chemin d'accès au shapefile correspondant à la zone d'emprise sur laquelle le traitement sera lancé.

5 - **Identifiant Theia** : Pour télécharger les images Landsat8 sur la plateforme Theia, il faut d'abord s'inscrire sur le site : <https://theia-landsat.cnes.fr/rocket/#/home>. Puis entrer le nom d'utilisateur et le mot de passe enregistré sur Theia-land dans l'application. Il y a également un bouton `Proxy ui_Proxy_window.Ui_Proxy_window()` qui permet de rentrer les informations concernant un éventuel **Proxy**.

---

**Note:** Dès que les images ont été téléchargées, l'outil ne les téléchargera pas à nouveau.

---

6 - **BD Alti** (Facultatif) : Chemin d'accès à la BD Alti. Calcul de pentes (+PHYMOBAT 1.1) `Slope.Slope()`.

7 - **Images THRS** : Chemin d'accès à l'orthographie IRC à 2m de résolution spatiale en GeoTiff. Les orthographies IRC sont distribuées par l'IGN par tuile, en ECW et à 50cm de résolution spatiale. Par conséquent, en utilisant les outils de GDAL (`gdalbuildvrt`, `gdal_translate`, `gdalwarp`), il faut fusionner toutes les tuiles, convertir la mosaïque finale en TIF et la dégrader à 2m de résolution spatiale (<https://github.com/SylvioL/MosaiqueIRC.git>).

8 - **Segmentation** : Chemin d'accès au shapefile correspondant à la segmentation IGN.

9 - **Echantillons RPG** : Chemin d'accès au shapefile correspondant aux surfaces des îlots culturels du RPG. Ces polygones représentent les échantillons de la végétation non naturelle (Cultures) et semi naturelle (Prairies permanentes).

10 - **Champs** : Le nom des champs où sont stockées les classes correspondantes aux grandes classes de la végétation → Non naturelle / Semi-naturelle.

11 - **Classes** : Les classes Non naturelles (1, 2, 3, 4, ... etc) et la classe semi naturelle (18).

12 - **Nbre de polygones** : Nombre d'échantillons à extraire du shapefile surfaces RPG pour lancer l'entraînement du classifieur et valider la classification.

13 - **Echantillons Herbacés / Ligneux** : Chemin d'accès au shapefile où sont stockés les échantillons d'herbacés et de ligneux. Même utilisation que pour les références 9, 10, 11, il faut entrer le nom des champs où sont les classes, écrire les classes et le nombre de polygones à utiliser.

14 - **Echantillons Ligneux denses / Ligneux mixtes** : Chemin d'accès au shapefile où sont stockés les échantillons de ligneux. Même utilisation que pour les références 9, 10, 11, il faut entrer le nom des champs où sont les classes, écrire les classes et le nombre de polygones à utiliser.

15 - **Fichier de sortie** : Chemin où sera stocké le résultat final (shapefile) en sortie de PHYMOBAT.

16 - Lancement des traitements (OK), ou fermer l'application (Close).

17 - **Multi-processing** : Sélectionné par défaut. Il permet d'accélérer le calcul des traitements en utilisant plusieurs processeurs. A décocher si la machine ne possède pas minimum 12Go de mémoire vive.

## Exemple sur un jeu de données test

---

**Note:** Jeu de données test à demander à [samuel.alleaume@irstea.fr](mailto:samuel.alleaume@irstea.fr)

---

L'image ci-dessous indique ce qu'il faut mettre dans chaque champs de l'application pour un jeu de données test bien précis. Il suffit juste de remplacer “/media/laventure/DATA/EclipseFolder/CarHab\_v2/Cher18\_small\_zone\_test” par l'arborescence où se trouve le jeu de données test sur votre machine.

Bien entendu il faut, comme indiqué dans la référence 5, avoir vos propres identifiants Theia (Utilisateur et mot de passe).

Pour les classes de cultures (Non-naturelles) du RPG qui ne sont pas bien visible sur l'image : 1, 2, 3, 4, 5, 6, 7, 8, 9, 15, 20, 21, 24

Tous ces éléments sont également enregistrés dans le fichier « Save\_test.xml » du dossier « data\_test ».

## 2.2.2 Interface experte

Pour ouvrir l'interface experte : Sur-onglet *Mode* > *Mode expert*

Cette interface est composé de trois onglets :

- Traitement des images
- Traitement des échantillons
- Traitement de classification

Chaque onglet représente les parties du processus algorithmique décrit plus haut.

---

**Note:** Les traitements sont lancés en appuyant sur le bouton « OK » et si au moins une « check box » associée à chaque onglets et fonction est cochée. Si aucune « check box » n'est cochée, aucun traitement ne sera lancé.

---

Cette présentation d'API se compose suivant les trois onglets.

## 2.2.3 Interface du traitement des images

---

**Note:** Cet onglet est indépendant des autres onglets.

---

Les références en rouge sur les images suivantes sont les mêmes que celles qui ont été présentées sur l'interface simplifiée.

Au numéro 18, le choix du **capteur des images à télécharger** (Landsat, Sentinel 2 et Spot Word Heritage)

Dans cet onglet il y a cinq traitements qui peuvent être lancés :

19. **Images disponibles** : Cherche le nombre d'images disponibles sur la plate-forme Theia et l'inscrit à la palce du zéro en face.
20. **Télécharger** : Télécharge les images à partir de la plate-forme Theia. Pour cela, il faut obligatoirement remplir dans les lignes d'édition en-dessous correspondant aux identifiants Theia (Utilisateur et mot de passe).
21. **Traitement des images** : Traitement des images satellites (mosaïque des images, calculs des indices spectraux, ...)

**PHYMOBATs 3.0**

Chemin du dossier principal  
 ...

Période ou année des images  *Exemple pour 1 periode : AAAA-MM-DD,AAAA-MM-DD*

Emprise de la zone (shp)  
 ...

Identifiant Theia

Utilisateur :  Mot de passe :

BD Alti (Facultatif)  
 ...

Images THRS  
 ...

Segmentation (shp)  
 ...

---

Echantillons RPG (Végétation non naturelle / Semi-naturelle)  
 ...

Non naturelle      Semi-naturelle

Champs

Classes   Nbre de polygones

---

Echantillons Herbacés / Ligneux  
 ...

Herbacés      Ligneux

Champs

Classes   Nbre de polygones

---

Echantillons Ligneux denses / mixtes  
 ...

Ligneux denses      Ligneux mixtes

Champs

Classes   Nbre de polygones

---

Fichier de sortie (shp)  
 ...

☒ Multi-processing



PHYMOB Menu Aide Mode

PHYMOBATE 3.0

Traitement images Echantillons Classification

Chemin du dossier principal

...

Capteur des images à télécharger Landsat

Période, année des images Exemple pour 1 période : AAAA-MM-DD,AAAA-MM-DD

Emprise de la zone (shp)

...

☐ Images disponibles : 0

☐ Télécharger

Identifiant Theia :

Utilisateur :  Mot de passe  Proxy

☐ Traitement des images téléchargées

☐ BD Alti (Facultatif)

...

☐ Images THRS

...

☒ Multi-processing

Close OK

PHYMOBATe 3.0

Traitement images   Echantillons   Classification

Chemin du dossier principal 2

Capteur des images à télécharger 18 Landsat

Période, année des images 3 Exemple pour 1 période : AAAA-MM-DD,AAAA-MM-DD

Emprise de la zone (shp) 4

☐ Images disponibles : 19 → 0

☐ Télécharger 20 → 5

Identifiant Theia :

Utilisateur :  5 Mot de passe  Proxy

☐ Traitement des images téléchargées 21

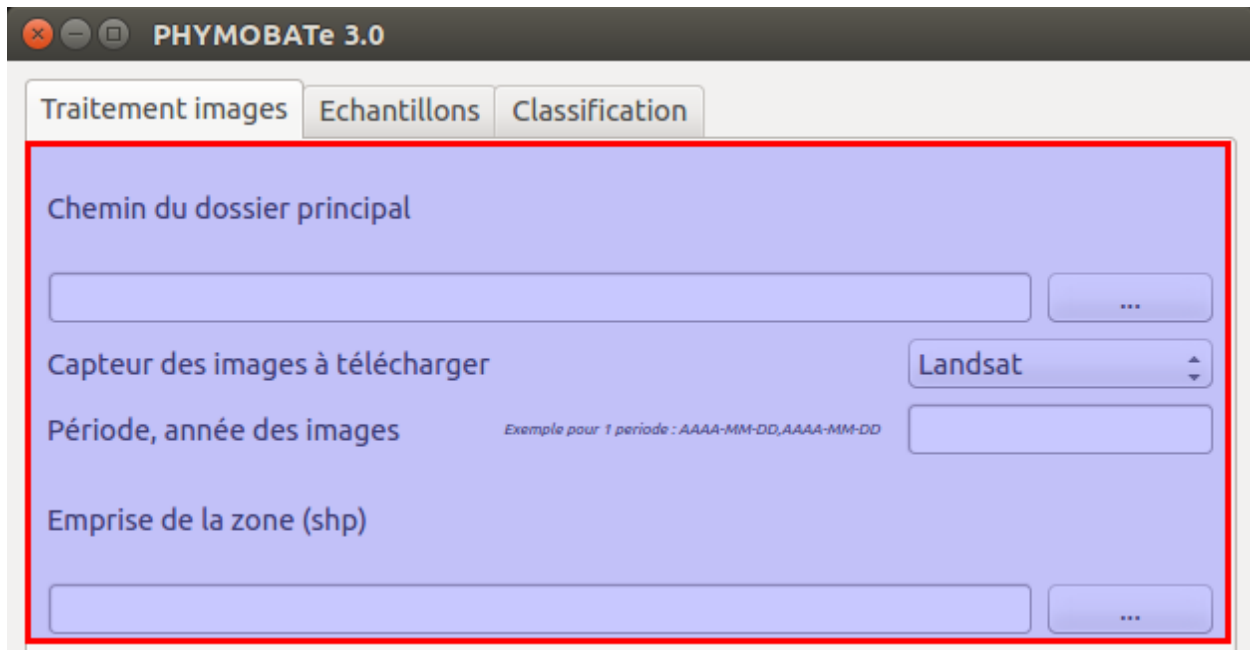
☐ BD Alti (Facultatif)

☐ Images THRS

☒ Multi-processing

**Note:** Pour **BD Alti** et **Image THRS** : Pour que le traitement soit réalisé, il faut absolument que ces cases soient cochées.

**Avertissement:** Avant de lancer un traitement, il faut absolument renseigner toutes les premières lignes d'édition et sélectionner un *capteur*.



## 2.2.4 Interface du traitement des échantillons

**Note:** Cet onglet peut être indépendant des autres onglets. Il faut cocher la « check box » **Image échantillonnée** (22) pour signaler au processus que les traitements du premier onglet n'ont pas été lancés. Ainsi renseigner le raster associé aux échantillons dans la ligne d'édition en-dessous.

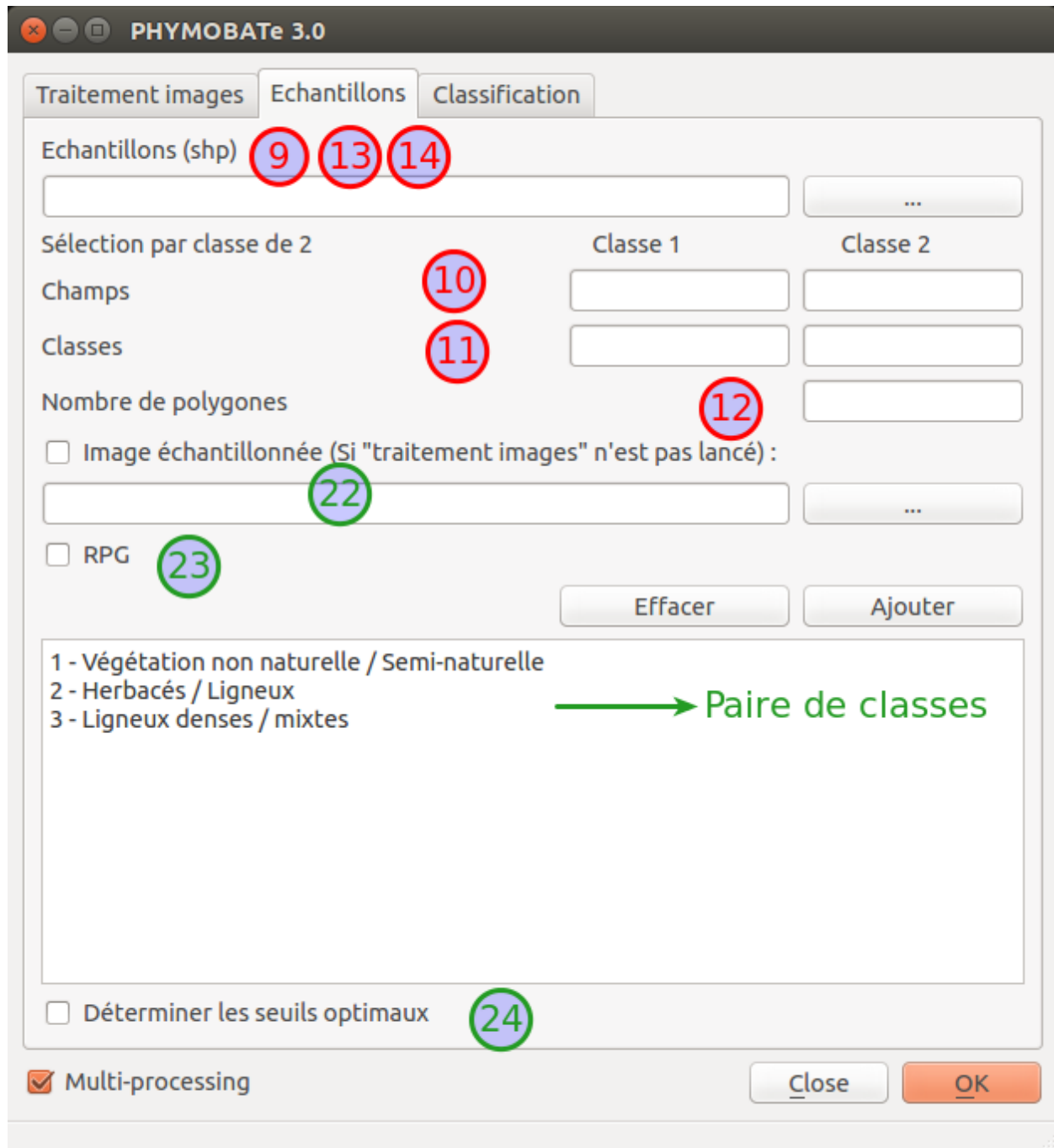
**Note:** Même si cet onglet peut être indépendant des autres onglets, il faut malgré tout renseigner l'**emprise de la zone** du premier onglet « Traitement des images ».

Dans cet onglet il y a trois « check box » dont deux correspondent à des fonctions de traitement :

23. **RPG** : Création du shapefile RPG mono-culturel `Processing.Processing.i_vhrs()`.

**Note:** Ce traitement est lancé en appuyant sur le bouton « Ajouter ».

24. **Déterminer les seuils optimaux** : Calcul des seuils optimaux par paire de classes `Seath.Seath.separability_and_threshold()`. Ce traitement est activé en appuyant sur « OK ».



Cette dernière fonction est l'objectif principal de cet onglet. Il faut ajouter à la chaîne de traitement, les échantillons associés aux paires de classes. Les deux exemples suivants montrent la démarche à suivre pour remplir les champs associés aux échantillons.

**Exemple pour la classe 1, Végétation non naturelle (cultures) / Semi-naturelle :**

L'échantillon est un fichier RPG, il faut cocher la case RPG et entrer le nom du fichier, les classes à extraire (si il y en a plusieurs, les séparer d'une virgule), les champs associés aux classes et le nombre de polygones à extraire.

En appuyant sur « Ajouter », une vérification des données entrées peut être effectuée comme indiqué ci-dessous :

**Exemple pour la classe 2, Herbacés / Ligneux :**

Ce fichier n'est pas un fichier RPG, la case RPG reste décochée.

Le bouton **Effacer**, efface toutes les informations entrées par l'utilisateur.

## 2.2.5 Interface du traitement de classification

**Note:** Cet onglet est dépendant des deux autres. Au préalable, il faut obligatoirement lancer tous les traitements précédents.

Le seul traitement qui sera lancé dans cet onglet est le processus de classification `Processing.Processing.i_classifier_s()` ou `Processing.Processing.i_classifier_rf()`.

Un fichier shapefile sera créé à l'emplacement indiqué par l'utilisateur, **fichier de sortie (15)**. Il dépend du shapefile en entrée issue de la **segmentation (8)** IGN.

Il y a un choix entre deux **méthodes de classification (25)** : Random Forest (Plus long en calcul) et Seath.

**Note:** Avec la méthode du Random Forest, la classification se fait directement sur les trois niveaux contrairement à la méthode Seath qui peut se faire sur un, deux ou trois niveaux de classe.

Pour activer les niveaux d'extraction, il faut cocher les cases associées. L'utilisateur peut personnaliser les champs des entités de sortie comme ceci :

26-1 - **Pour extraire que le premier niveau (Seath)**

26-2 - **Pour extraire les deux premiers niveaux (Seath)**

26-3 - **Pour extraire tous les niveaux (RF et Seath)**

Où :

- ID : Identifiant unique
- AREA : Superficie du polygone en ha
- NIVEAU\_1 : Non végétation semi-naturelle / Semi-naturelle
- NIVEAU\_2 : Eboulis / Agriculture | Herbacés / Ligneux
- NIVEAU\_3 : Ligneux denses / mixtes et Phytomasse faible / moyenne / forte
- POURC : Densité de ligneux et de phytomasse

Les listes déroulantes indiquent la nature des champs, il y a deux choix :

- String = Chaîne de caractères
- Real = Chiffre réel

PHYMOBATe 3.0

Traitement images Echantillons Classification

Echantillons (shp)

\_test/RPG/SURFACES-2011-ILOTS\_ANONYMES\_018\_20120616.shp ...

Sélection par classe de 2

	Classe 1	Classe 2
Champs	CODE_GROUP	CODE_GROUP
Classes	, 8, 9, 15, 20, 21, 24	18
Nombre de polygones		10

☐ Image échantillonnée (Si "traitement images" n'est pas lancé) :

☐ RPG

Effacer Ajouter

1 - Végétation non naturelle / Semi-naturelle  
2 - Herbacés / Ligneux  
3 - Ligneux denses / mixtes

☐ Déterminer les seuils optimaux

☒ Multi-processing

Close OK

PHYMOBATe 3.0

Traitement images Echantillons Classification

Echantillons (shp)

...

Sélection par classe de 2

Classe 1

Classe 2

Champs

Classes

Nombre de polygones

☐ Image échantillonnée (Si "traitement images" n'est pas lancé) :

...

☐ RPG

Effacer Ajouter

1 - Végétation non naturelle / Semi-naturelle  
2 - Herbacés / Ligneux  
3 - Ligneux denses / mixtes

/media/laventure/DATA/PHYMOBAT\_Tuto/data\_test/RPG/MONO\_ILOTS\_ANONYME  
S\_018\_20120616.shp  
CODE\_GROUP CODE\_GROUP  
1, 2, 3, 4, 5, 6, 7, 8, 9, 15, 20, 21, 24 18  
10

☒ Déterminer les seuils optimaux

☒ Multi-processing

Close OK

PHYMOBATe 3.0

Traitement images Echantillons Classification

Echantillons (shp)

/PHYMOBAT\_Tuto/data\_test/Sample\_ligneux/echant\_ligneux.shp ...

Sélection par classe de 2

	Classe 1	Classe 2
Champs	echant	echant
Classes	H	LO,LF
Nombre de polygones		10

☐ Image échantillonnée (Si "traitement images" n'est pas lancé) :

☐ RPG

Effacer Ajouter

1 - Végétation non naturelle / Semi-naturelle  
2 - Herbacés / Ligneux  
3 - Ligneux denses / mixtes

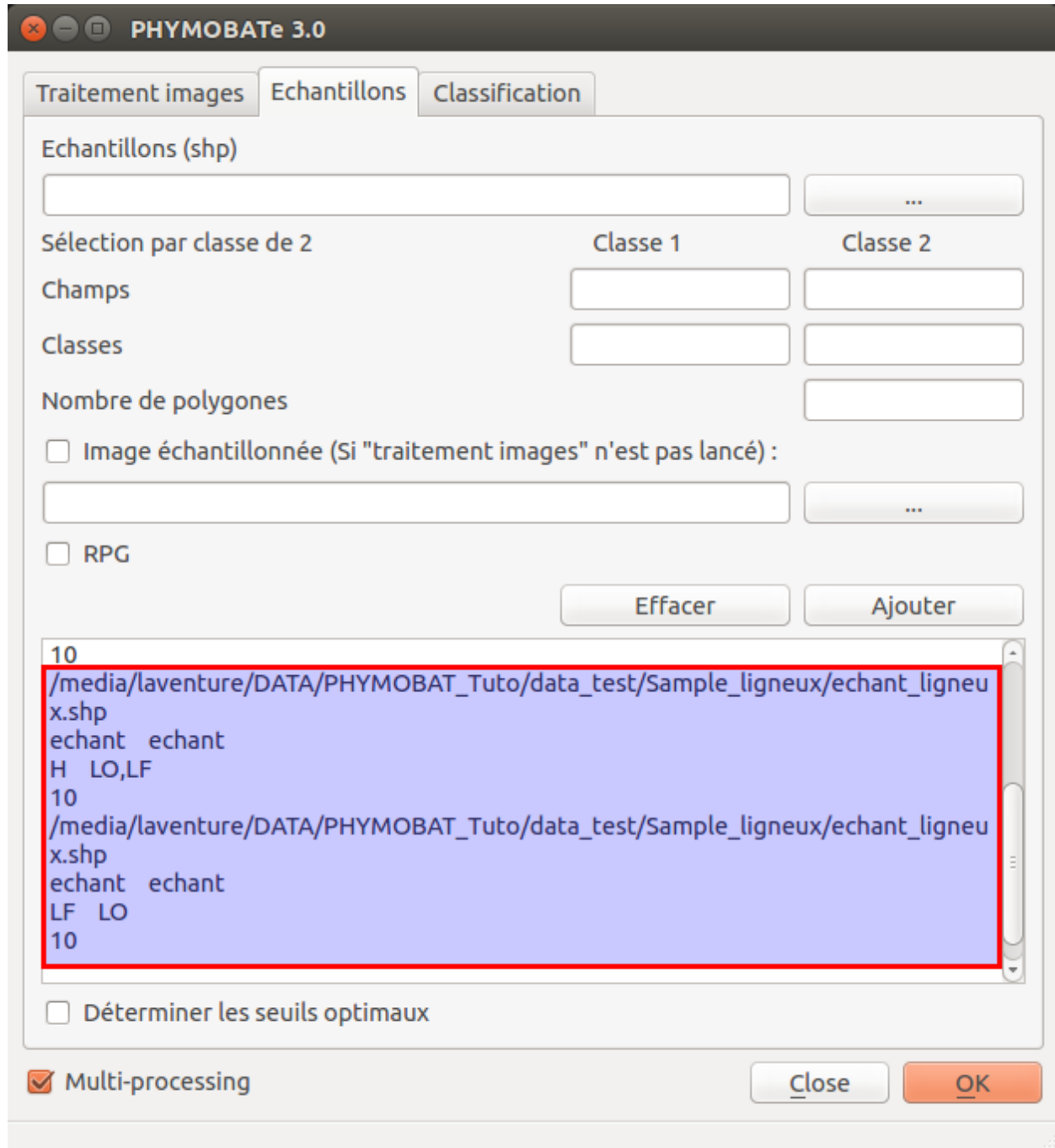
/media/laventure/DATA/PHYMOBAT\_Tuto/data\_test/RPG/MONO\_ILOTS\_ANONYMES\_018\_20120616.shp  
CODE\_GROUP CODE\_GROUP  
1, 2, 3, 4, 5, 6, 7, 8, 9, 15, 20, 21, 24 18  
10

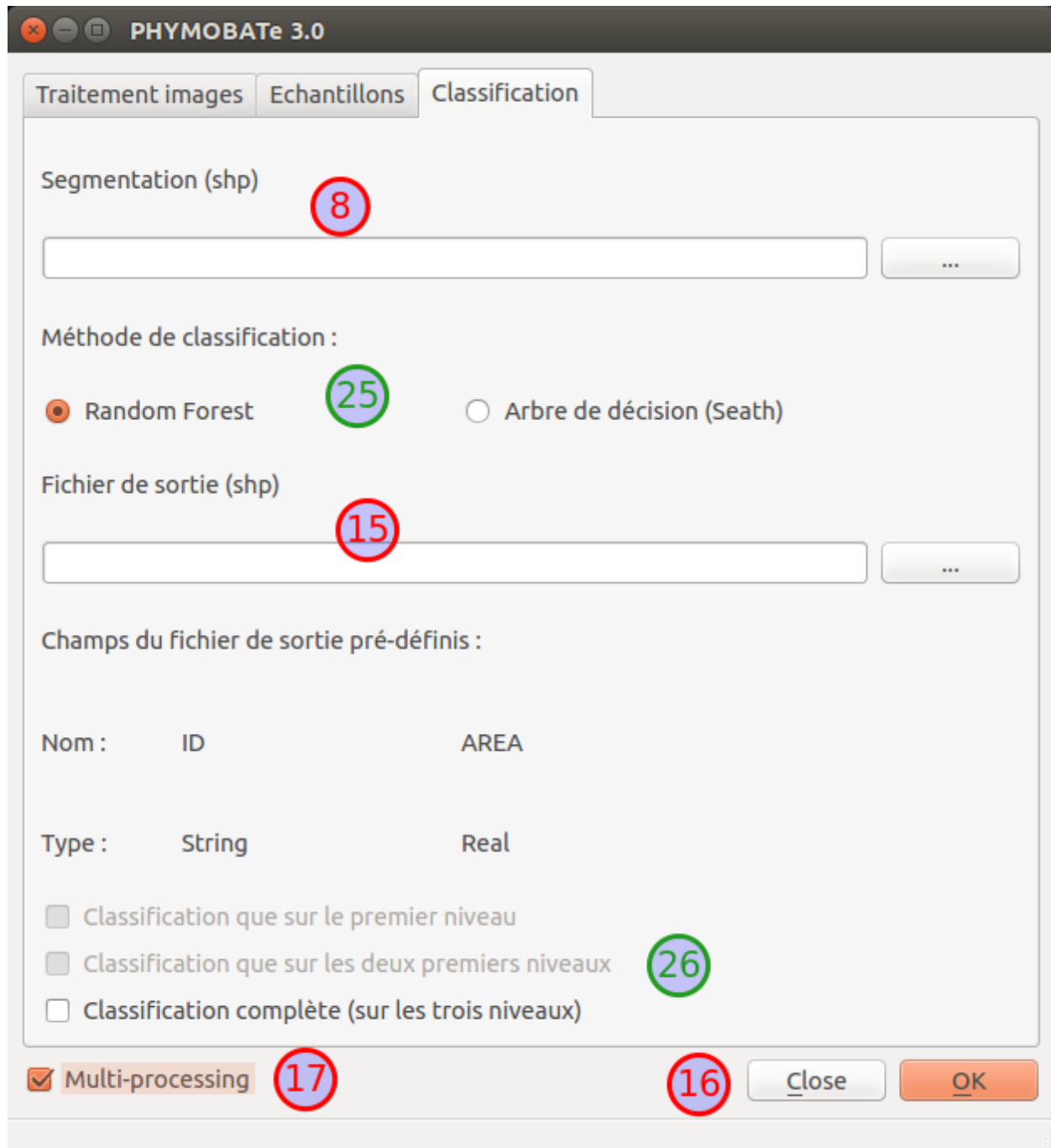
☐ Déterminer les seuils optimaux

☒ Multi-processing

Close OK







PHYMOBATE 3.0

Traitement images   Echantillons   **Classification**

Segmentation (shp)

TA/PHYMOBAT\_Tuto/data\_test/Segm18/Segm\_test\_Chér18\_150402.shp ...

Méthode de classification :

☐ Random Forest   ☒ Arbre de décision (Seath)

Fichier de sortie (shp)

a/laventure/DATA/PHYMOBAT\_Tuto/data\_test/Segm18/FB\_test\_mnt.shp ...

Champs du fichier de sortie pré-définis :

Nom :	ID	AREA
Type :	String	Real

☒ Classification que sur le premier niveau

Champs des entités

Nom : NIVEAU\_1

Type : String

☐ Classification que sur les deux premiers niveaux

☐ Classification complète (sur les trois niveaux)

☒ Multi-processing

Close OK

PHYMOBATe 3.0

Traitement images   Echantillons   **Classification**

Segmentation (shp)

...

Méthode de classification :

☐ Random Forest   ☒ Arbre de décision (Seath)

Fichier de sortie (shp)

...

Champs du fichier de sortie pré-définis :

Nom :	ID	AREA
Type :	String	Real

☐ Classification que sur le premier niveau

☒ Classification que sur les deux premiers niveaux

Champs des entités	Nom :	<input type="text" value="NIVEAU_1"/>	<input type="text" value="NIVEAU_2"/>
	Type :	<input type="text" value="String"/>	<input type="text" value="String"/>

☐ Classification complète (sur les trois niveaux)

☒ Multi-processing

PHYMOBATe 3.0

Traitement images   Echantillons   **Classification**

Segmentation (shp)

...

Méthode de classification :

☒ Random Forest   ☐ Arbre de décision (Seath)

Fichier de sortie (shp)

...

Champs du fichier de sortie pré-définis :

Nom :	ID	AREA
Type :	String	Real

☐ Classification que sur le premier niveau

☐ Classification que sur les deux premiers niveaux

☒ Classification complète (sur les trois niveaux)

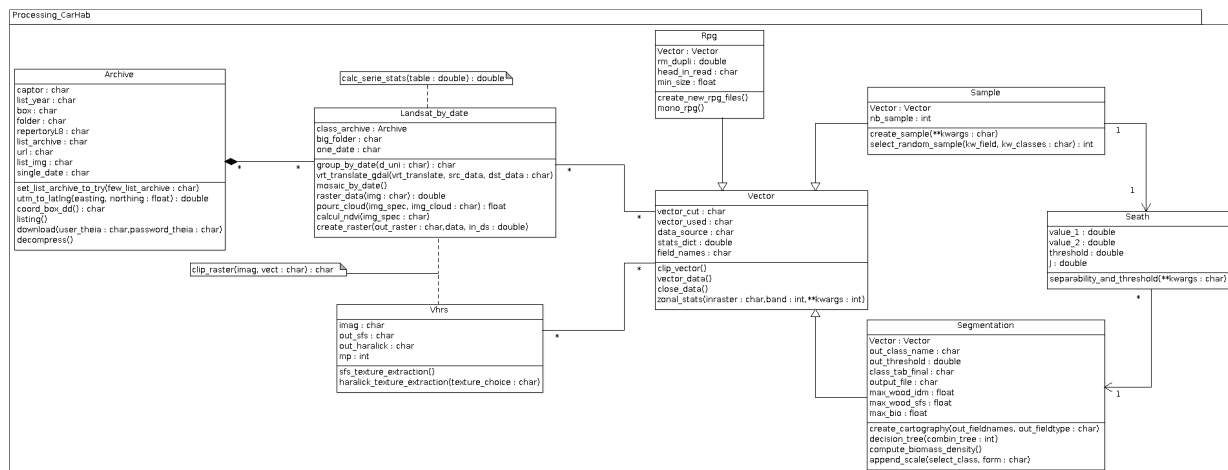
Champs des entités	Nom :	<input type="text" value="NIVEAU_1"/>	<input type="text" value="NIVEAU_2"/>	<input type="text" value="NIVEAU_3"/>	<input type="text" value="POURC"/>
	Type :	<input type="text" value="String"/>	<input type="text" value="String"/>	<input type="text" value="String"/>	<input type="text" value="Real"/>

☒ Multi-processing



## CarHab Phy MOBA package

The class diagram associated to the processing is this :



## 3.1 Image processing

### 3.1.1 Archive

**class** `Archive.Archive` (*captor, list\_year, box, repertory, proxy\_enabled*)

Class to list, download and unpack Theia image archive because of a shapefile (box). This shapefile get extent of the area.

#### Paramètres

- **captor** (*str*) – Name of the satellite (ex: Landsat or SpotWorldHeritage ...).

Name used to the url on website Theia Land

- **list\_year** (*list of str*) – Processing's year (string for one year)
- **box** (*str*) – Path of the study area
- **folder** (*str*) – Path of the source folder
- **repertory** (*str*) – Name of the archive's folder

**coord\_box\_dd()**

Function to get area's coordinates of shapefile

**Renvoie**

str – **area\_coord\_corner** : Area coordinates corner

→ Left bottom on x, Left bottom on y, Right top on x, Right top on y

**Example**

```
>>> import Archive
>>> test = Archive(captor, list_year, box, folder, repertory, proxy_enabled)
>>> coor_test = test.coord_box_dd()
>>> coor_test
'45.52, 2.25, 46.71, 3.27'
```

**decompress()**

Function to unpack archives and store informations of the images (date, path, ...)

**download\_auto** (*user\_theia, password\_theia*)

Function to download images archive automatically on Theia land data center. Source : [https://github.com/olivierhagolle/theia\\_download](https://github.com/olivierhagolle/theia_download)

**Paramètres**

- **user\_theia** (*str*) – Username Theia Land data center
- **password\_theia** (*str*) – Password Theia Land data center

**listing()**

Function to list available archive on platform Theia Land, and on the area

**set\_list\_archive\_to\_try** (*few\_list\_archive*)

Test function to download a few archives

**Paramètres few\_list\_archive** (*list dimension 2*) – [archive\_download, out\_archive] with :

- **archive\_download** : Archives downloaded
- **out\_archive** : Output archives path

**utm\_to\_latlng** (*zone, easting, northing, northernHemisphere=True*)

Function to convert UTM to geographic coordinates

**Paramètres**

- **zone** (*int*) – UTM zone
- **easting** (*float*) – Coordinates UTM in x
- **northing** (*float*) – Coordinates UTM in y
- **northernHemisphere** (*boolean*) – North hemisphere or not

**Renvoie** tuple – integer on the **longitude** and **latitude**

Source : <http://www.ibm.com/developerworks/java/library/j-coordconvert/index.html>



### 3.1.2 Landsat image processing

**class** `RasterSat_by_date.RasterSat_by_date` (*class\_archive*, *big\_folder*)

Satellite image processing's class. This class include several processes to group images by date, mosaic images by date, extract images information, compute ndvi, compute cloud pourcent and create new rasters.

@param *class\_archive*: Archive class name with every information on downloaded images @type *class\_archive*: class

@param *big\_folder*: Image processing folder @type *big\_folder*: str

@param **one\_date**: [year, month, day] ... This variable is modified in the function `mosaic_by_date()`. To append mosaic image path, mosaic cloud image path, cloud pixel value table, mosaic ndvi image path and ndvi pixel value table.

@type *one\_date*: list of str

@param **choice\_nb\_b**: A option to choice output image number of band : func: `layer_rasterization` in Vector's class. If this option is 0, it take input band. By default 0.

@type *choice\_nb\_b*: int

**calcul\_ndvi** ()

**Function to compute NDVI index for a Landsat image with OTB BandMathX**

- OTB help :

- *il* : Images list : data and cloud
- *exp* : Expression for ndvi :  $(PIR-R)/(PIR+R)$
- *out* : Feature Output Image

**group\_by\_date** (*d\_uni*)

Function to extract images on a single date

@param *d\_uni*: [year, month, day] @type *d\_uni*: list of str

@returns: list of str – variable **group** = [year, month, day, multispectral image path, cloud image path]

**mosaic\_by\_date** (*date*)

Function to merge images of the same date in a image group : func: `group_by_date`.

**pourc\_cloud** ()

**Return clear pixel percentage on the image `self.cloudiness_pourcentage[0]` because of a cloud image `cloudiness_pourcentage[1]`.**

**Renvoie** float – variable **nb0**, clear pixel percentage.

**Example**

```
>>> import RasterSat_by_date
>>> Landsat_test = RasterSat_by_date(class_archive, big_folder, one_date)
>>> nb0_test = Landsat_test.pourc_cloud(Landsat_test._one_date[3], Landsat_
↳ test._one_date[4])
>>> nb0_test
98
```

**raster\_data** (*img*)

Function to extract raster information. Return table of pixel values and raster information like line number, pixel size, ... (gdal pointer)

@param img : Raster path @type img : str

@returns : numpy.array – variable **data**, Pixel value matrix of a raster.

gdal pointer – variable **\_in\_ds**, Raster information.

**vrt\_translate\_gdal** (*vrt\_translate, src\_data, dst\_data*)

Function to launch gdal tools in command line. With `gdalbuildvrt` and `gdal_translate`. This function is used *mosaic\_by\_date()* to mosaic image by date.

@param vrt\_translate: vrt or translate @type vrt\_translate: str

@param src\_data: Data source. Several data for vrt process and one data (vrt data) for gdal\_translate

@type src\_data: list (process vrt) or str (process translate)

@param dst\_data: Output path @type dst\_data: str

### 3.1.3 Texture index processing

**class** Vhrs.Vhrs (*imag, mp*)

Class to compute Haralick and SFS textures with OTB

@param imag: The input image path to compute texture image @type imag: str

@param out\_sfs/out\_haralick: Output path @type out\_sfs/out\_haralick: str

@param mp: Boolean variable -> 0 or 1.

- 0 means, not multi-processing
- 1 means, launch process with multi-processing

@type mp: int

**haralick\_texture\_extraction** (*texture\_choice*)

Function to compute OTB Haralick texture image

- OTB help :
  - in : Input Image
  - channel : Selected Channel
  - **Texture feature parameters** [This group of parameters allows to define texture parameters.]
    - \* X Radius : X Radius
    - \* Y Radius : Y Radius
    - \* X Offset : X Offset
    - \* Y Offset : Y Offset
  - Image Minimum : Image Minimum
  - Image Maximum : Image Maximum
  - Histogram number of bin : Histogram number of bin
  - **Texture Set Selection Choice of The Texture Set Available choices are :**
    - \* Simple Haralick Texture Features: This group of parameters defines the 8 local Haralick texture feature output image. The image channels are: Energy, Entropy, Correlation, Inverse Difference Moment, Inertia, Cluster Shade, Cluster Prominence and Haralick Correlation

- \* Advanced Texture Features: This group of parameters defines the 9 advanced texture feature output image. The image channels are: Mean, Variance, Sum Average, Sum Variance, Sum Entropy, Difference of Entropies, Difference of Variances, IC1 and IC2
- \* Higher Order Texture Features: This group of parameters defines the 11 higher order texture feature output image. The image channels are: Short Run Emphasis, Long Run Emphasis, Grey-Level Nonuniformity, Run Length Nonuniformity, Run Percentage, Low Grey-Level Run Emphasis, High Grey-Level Run Emphasis, Short Run Low Grey-Level Emphasis, Short Run High Grey-Level Emphasis, Long Run Low Grey-Level Emphasis and Long Run High Grey-Level Emphasis

– out : Feature Output Image

Source : [http://otbcb.readthedocs.org/en/latest/Applications/app\\_HaralickTextureExtraction.html](http://otbcb.readthedocs.org/en/latest/Applications/app_HaralickTextureExtraction.html)

@param texture\_choice: Order texture choice -> Simple / Advanced / Higher @type texture\_choice: str

**sfs\_texture\_extraction()**

**Function to compute OTB SFS texture image**

• **OTB help :**

- in : Input Image
- channel : Selected Channel
- **parameters** [Texture feature parameters. This group of parameters allows to define SFS texture parameters. The available texture features are SFS'Length, SFS'Width, SFS'PSI, SFS'W-Mean, SFS'Ratio and SFS'SD. They are provided in this exact order in the output image.]
  - \* parameters.spethre : Spectral Threshold
  - \* parameters.spathre : Spatial Threshold
  - \* parameters.nbdir : Number of Direction
  - \* parameters.alpha : Alpha
  - \* parameters.maxcons : Ratio Maximum Consideration Number
- out : Feature Output Image

### 3.1.4 Slope processing

**class Slope.Slope(mnt)**

Class to compute a slope raster

**Paramètres**

- **mnt** (*str*) – Digital Elevation Model (DEM) path
- **out\_mnt** (*str*) – slope raster path

**extract\_slope()**

Function to compute slope in GDAL command line.

### 3.1.5 Toolbox

**class Toolbox.Toolbox**

Class used to grouped small tools to cut raster or compute statistics on a raster.

**Paramètres**

- **imag** (*str*) – Input image (path)
- **vect** (*str*) – Extent shapefile (path)

**calc\_serie\_stats** (*table, stats\_name, output\_folder*)  
Function to compute stats on temporal cloud and ndvi spectral table Ndvi stats : min max  
@param table: Spectral data, cloud raster and ndvi raster @type table: numpy.ndarray  
@returns: list of numpy.ndarray – variable **account\_stats**, list of temporal NDVI stats.  
numpy.ndarray – variable **account\_cloud**, pixel number clear on the area.

**check\_proj** ()  
Function to check if raster's projection is RFG93. For the moment, PHYMOBAT works with one projection only Lambert 93 EPSG:2154

**clip\_raster** (*\*\*kwargs*)  
Function to clip a raster with a vector. The raster created will be in the same folder than the input raster.  
With a prefix "**Clip\_**".

**Kwargs** **rm\_rast** (int) - 0 (by default) or 1. Variable to remove the output raster. 0 to keep and 1 to remove.

**Renvoie** *str* – variable **outclip**, output raster clip (path).

## 3.2 Vector processing

### 3.2.1 Super vector

**class** Vector.**Vector** (*used, cut, \*\*opt*)  
Vector class to extract a area, vector data and zonal statistic  
@param vector\_used: Input/Output shapefile to clip (path) @type vector\_used: str  
@param vector\_cut: Area shapefile (path) @type vector\_cut: str  
@param vector\_name: Name of the shapefile @type vector\_name: str  
@param vector\_folder: Name of the folder containing the shapefile @type vector\_name: str  
@param data\_source: Input shapefile information @type data\_source: ogr pointer  
@param stats\_dict: Stats results @type stats\_dict: dict  
@param remove\_shp: Remove shapefile or not. 0 : don't remove, 1 : remove @type remove\_shp: int  
@opt: **Remove** (int) - For the remove\_shp variable

**clip\_vector** (*output\_folder*)  
Function to clip a vector with a vector

**close\_data** ()  
Function to remove allocate memory

**layer\_rasterization** (*raster\_head, attribute\_r, \*\*kwargs*)  
Function to rasterize a vector using OTB.  
@param raster\_head: Raster path that will look like the final raster of the rasterization @type raster\_head: str  
@param attribute\_r: Field name of the shapefile that contains class names @type attribute\_r: str

@kwargs: **choice\_nb\_b** (int) - Output image number of band. If you choice 1, take first band. If you choice 2, take two first band etc... @returns: str – **valid\_raster** : output raster path from the rasterization

**vector\_data** ()

Function to extract vector layer information

**zonal\_stats** (*liste\_chemin*)

Function to compute the mean in every polygons on a list images with otb

:param liste\_chemin : List input image path :type liste\_chemin: list(string)

**zonal\_stats\_pp** (*inraster*)

A zonal statistics ++ to determine pxl percent in every polygon

**Paramètres inraster** (*str*) – Input image path

**Renvoie** dict – **p\_stats** : dictionnary with pxl percent in every polygon. Mainly “Maj\_count” (Majority Value) and “Maj\_count\_perc” (Majority Percent)

### 3.2.2 Sample

**class** `Sample.Sample` (*used, cut, nb\_sample, \*\*opt*)

Vector class inherits the super vector class properties. This class create training sample.

@param vector\_used: Input/Output shapefile to clip (path) @type vector\_used: str

@param vector\_cut: Area shapefile (path) @type vector\_cut: str

@param nb\_sample: Number of polygons for every sample @type nb\_sample: int

@param vector\_val: Output shapefile to validate the futur classification @type vector\_val: str

@opt: Refer to the Vector class

**create\_sample** (*\*\*kwargs*)

Function to create a sample shapefile of a specific class

**Kwargs fieldname** (list of str) - Fieldname in the input shapefile (if the user want select polygons of the class names specific)

**class** (list of str) - class names in the input shapefile (with fieldname index). Can use one or several classes like this -> example : [classname1, classname2, ...]

**fill\_sample** (*output\_sample, polygon, \*\*opt*)

Function to fill and create the output sample shapefile. This function is used in `create_sample()` to create samples polygons and validated polygons (to the take out the precision of the classification).

@param output\_sample: Path of the output shapefile @type output\_sample: str

@param **polygon: Identity of the selected random polygons.** If this variable = 0, the processing will take all polygons

@type polygon: list or int

@opt: **add\_fieldname** (int) - Variable to know if add a field. By default non (0), if it have to add (1)

**fieldname** (str) - Fieldname to add in the input shapefile

**class** (int) - class names in integer to add in the input shapefile

**select\_random\_sample** (*kw\_field, kw\_classes*)

**Function to select id with class name specific only.** This function is used in `create_sample()`

@param kw\_field: Field name in the input shapefile @type kw\_field: str

@param kw\_classes: Class names in the input shapefile like this -> "classname1, classname2" @type kw\_classes: str

@returns: list – variable **select\_id**, List of id with a class name specific.

### 3.2.3 RPG

**class** Rpg.Rpg(*used, cut*)

Vector class inherits the super vector class properties. This class create a new RPG shapefile with mono-crops. It needs a basic RPG shapefile and a basic RPG CSV file (...-GROUPES-CULTURE...) in `mono_rpg()`.

@param vector\_used: Input/Output shapefile to clip (path) @type vector\_used: str

@param vector\_cut: Area shapefile (path) @type vector\_cut: str

@param rm\_dupli: Rpg table with no duplicated crops group @type rm\_dupli: dict

@param head\_in\_read: List of rpg header @type head\_in\_read: list of str

@param min\_size: Minimum size to extract a rpg polygons @type min\_size: float

**create\_new\_rpg\_files** ()

Function to create new rpg shapefile with **rm\_dpli** variable. The output shapefile will be create in the same folder than the input shapefile with prefix *MONO\_*.

**mono\_rpg** (*path\_rpg*)

Function to extract no duplicated crops.

### 3.2.4 Separability and threshold index

**class** Seath.Seath

Get the optimal threshold and Bhattacharyya distance for a separability between two classes

Source article : SEaTH-A new tool for automated feature extraction in the context of object-based image analysis S. Nussbaum et al.

Source Info : Kenji Ose (IRSTEA) et Nathalie St-Geours (IRSTEA)

#### Paramètres

- **value\_1** (*list*) – List of index mean by polygons (sample 1)
- **value\_2** (*list*) – List of index mean by polygons (sample 2)
- **threshold** (*str*) – Optimal threshold under this form >0.56
- **J** (*list*) – Jeffries-Matusita distance (measure separability between 2 classes on a 0 to 2 scale)

#### Example

```
>>> import Seath
>>> a = Seath()
>>> a.value_1 = b[0].stats_dict
>>> a.value_2 = b[1].stats_dict
>>> a.separability_and_threshold()
>>> a.threshold[0]
'>0.56'
>>> a.J[0]
1.86523428
```

**separability\_and\_threshold** (*\*\*kwargs*)

Function to extract the optimal threshold for a separability between two classes

**Kwargs** **index** (str) - The processing will prints the string

### 3.2.5 Classification

**class** `Segmentation.Segmentation` (*used, cut*)

Vector class inherits the super vector class properties. This class create the final shapefile : Cartography on a input segmentation by decision tree.

**The output classname are (out\_class\_name variable):**

- Vegetation non naturelle
- Vegetation semi-naturelle
- Herbacees
- Ligneux
- Ligneux mixtes
- Ligneux denses
- Forte phytomasse
- Moyenne phytomasse
- Faible phytomasse

@param vector\_used: Input/Output shapefile to clip (path) @type vector\_used: str

@param vector\_cut: Area shapefile (path) @type vector\_cut: str

@param output\_file: Output shapefile cartography. This path is the same than the segmentation path. @type output\_file: str

@param out\_class\_name: List of output class name @type out\_class\_name: list of str

@param out\_threshold: List of output threshold @type out\_threshold: list of str

@param **max**...: Biomass and density (IDM, SFS index) maximum @type **max**...: float

@param class\_tab\_final: Final decision tree table @type class\_tab\_final: dict

**@param stats\_rpg\_tif: Dictionnary with pxl percent in every polygon of class RPG.** Mainly “Maj\_count” (Majority Value) and “Maj\_count\_perc” (Majority Percent)

@type stats\_rpg\_tif: dict

**append\_scale** (*select\_class, form*)

Function to complete the “class\_tab\_final” list with density and biomass information. This list will be used to build the final shapefile.

@param select\_class: Class name to add degree @type select\_class: str

@param form: Formula to add degree @type form: str

**compute\_biomass\_density** (*method='SEATH'*)

Function to compute the biomass and density distribution. It returns threshold of biomass level.

@param method: Classification method used. It can set “SEATH” (by default) or “RF” @type method: str

**create\_cartography** (*out\_fieldnames, out\_fielddtype*)

Function to create a output shapefile. In this output file, there is the final cartography. With output defined field names and field type in the main process.

### Paramètres

- **out\_fieldnames** (*list of str*) – List of output field names
- **out\_fieldtype** (*list of str*) – List of output field type

**decision\_tree** (*combin\_tree*)

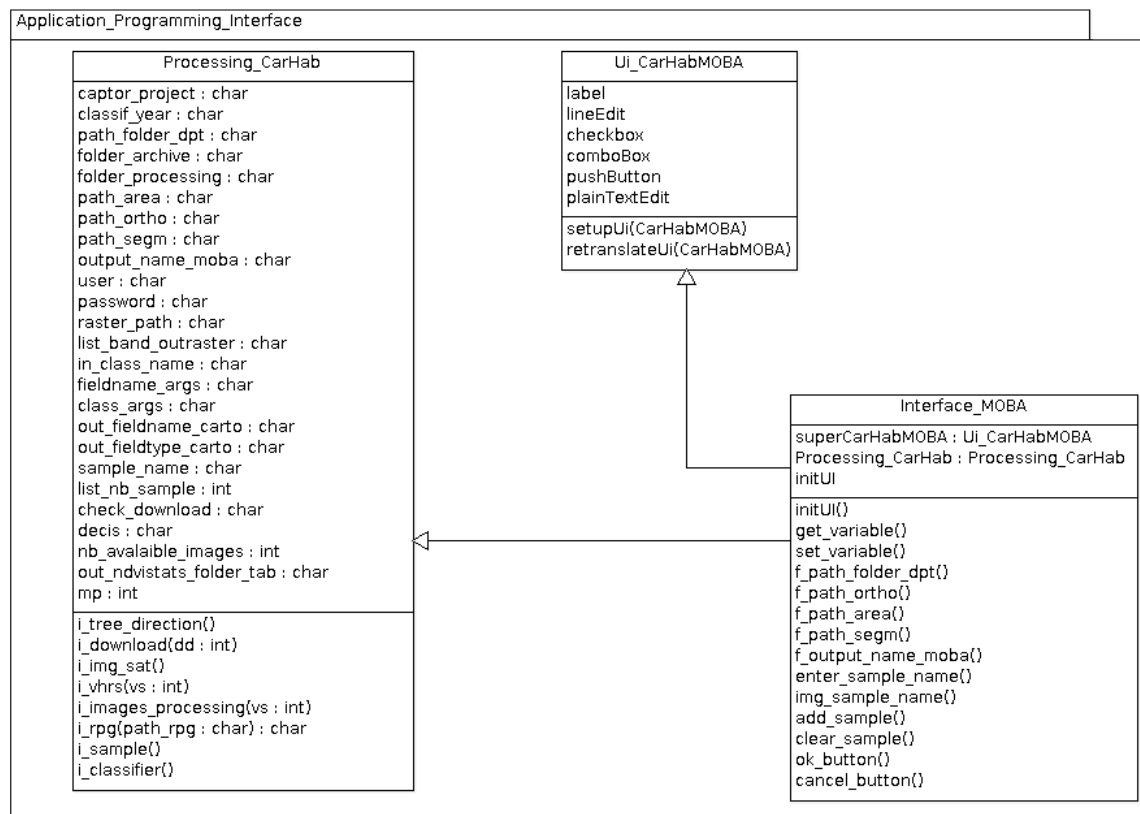
Function to build the decision tree. Taking account output threshold and input class name.

@param combin\_tree: Decision tree combination @type combin\_tree: list of number class name



## Graphical User Interface package

The class diagram associated to the interface is this :



## 4.1 Interface command

Interface main, PHYMOBAT (FB PHYsionomiques Milieux Ouverts de Basse Altitude par Télédétection)

`__name__` = « PHYMOBAT 3.0 »

`__license__` = « GPL »

`__version__` = « 3.0 »

`__author__` = « LAVENTURE Sylvio - UMR TETIS / IRSTEA »

`__date__` = « Mars 2016 »

**class** PHYMOBAT.**PHYMOBAT** (*mode=0, parent=None*)

Interface main class. It makes to link `ui_PHYMOBAT_tab` and `Processing`.

**about\_PHYMOBA** ()

Function to open a new window « About PHYMOBAT »

**activate\_level** ()

To activate the first levels with seath method. This is in pushing on the decision tree radio button. Else it activates the last level to random forest method.

**add\_sample** ()

Add sample information and location to compute optimal threshold :

For the expert mode (*mode=1*) :

- Append a sample name by line Edit. *This is a check box RPG, if the sample is RPG file. It launch the Rpg class. And append a other sample from Rpg class.*
- Append two existent sample field names by combobox. It will be the same.
- Append sample class names by line edit. One or more for every sample.
- Append number of polygons for every samples by line edit.
- Print in a plain text edit : sample name, two sample field names, sample class names and number of polygons.
- ***This check box Image echantillonee, image path for samples if the first processing image hasn't been laun***

---

**Note:** This is for a image with one spectral band

---

- Clear all widget field at the end.

For the simply mode (*mode=0*):

- Append a sample name by a different line Edit (a line Edit for each sample).
- Append sample class names, existing sample fields and number of polygons (a different line Edit for each sample)

**block\_for\_swh** ()

Function to block others function when SportWorldHeritage is selected in the combobox captor.

**change\_mode** (*new\_mode*)

Function to switch between SIMPLE and EXPERT mode of « PHYMOBAT »

**clear\_sample** ()

Function to clear sample record. Clear in the interface and in the memory list.

**close\_button()**  
Function to close the interface.

**display\_all\_levels()**  
Function to display fieldnames option to launch complete classification

**display\_one\_level()**  
Function to display fieldnames option to classifier one level

**display\_two\_levels()**  
Function to display fieldnames option to classifier two first levels

**enter\_sample\_name()**  
Open a input browser box to select the sample shapefile path by line edit. With *add\_sample()* conditions for the expert mode. For the simply mode, this function is used for the RPG shapefile.

**enter\_sample\_name\_h1()**  
Open a input browser box to select the grass and wooden sample shapefile path by line edit. With *add\_sample()* conditions for the simply mode.

**enter\_sample\_name\_ll()**  
Open a input browser box to select the wooden sample shapefile path by line edit. With *add\_sample()* conditions for the simply mode.

**f\_output\_name\_moba()**  
Set the output classification shapefile path by line edit.

**f\_path\_area()**  
Open a input browser box to select the study area shapefile path by line edit.

**f\_path\_folder\_dpt()**  
Open a input browser box to select the main folder path by line edit.

**f\_path\_mnt()**  
Open a input browser box to select the MNT image path by line edit.

**f\_path\_ortho()**  
Open a input browser box to select the VHRS image path by line edit.

**f\_path\_segm()**  
Open a input browser box to select segmentation shapefile path path by line edit.

**f\_proxy()**  
Function to open a popup in order to enter proxy ID

**field\_display\_1()**  
Function to display fieldname class 1 in the other fieldname class 2 when text changed. For the simply mode, this is RPG sample.

**field\_display\_2()**  
Function to display fieldname class 2 in the other fieldname class 2 when text changed. For the simply mode, this is RPG sample.

**field\_display\_3()**  
For the grass/wooden sample, a function to display fieldname class 1 in the other fieldname class 2 when text changed.

**field\_display\_4()**  
For the grass/wooden sample, a function to display fieldname class 2 in the other fieldname class 2 when text changed.

**field\_display\_5()**

For the wooden sample, a function to display fieldname class 1 in the other fieldname class 2 when text changed.

**field\_display\_6()**

For the wooden sample, a function to display fieldname class 2 in the other fieldname class 2 when text changed.

**forget\_raster\_sample()**

Function to open a new window “Alert” because user forgotten to declare rasters or samples.

**forget\_study\_area()**

Function to open a new window “Alert” because user forgotten to declare study area.

**get\_variable()**

Add a all system value like :

- Main folder path by line edit
- Satellite captor name by combo box
- Classification year by line edit
- Study area shapefile path by line edit
- Connexion username and password by line edit
- VHRS image path by line edit
- MNT image path by line edit
- Segmentation shapefile path path by line edit
- Output classification shapefile path by line edit
- Output shapefile field name by line edit and field type by combo box

**help\_tools()**

Function to open html help

**img\_sample\_name()**

Open a input browser box to select the image for samples path by line edit. With *add\_sample()* conditions.

**initUI()**

Get initial values from interface after a click button.

There is :

- **Connect browser button to search a path**
  - Main folder path
  - Study area shapefile path
  - VHRS image path
  - MNT image path
  - Segmentation shapefile path
  - Output classification shapefile path
  - Sample shapefile path
  - Image path for samples if the first processing image hasn’t been launched
- Connect button to add sample in the memory list

- Connect button to clear sample record. Clear in the interface and in the memory list
- Connect closelok button
- Connect menu bar tab (Open backup, save in a xml file, close, help, About PHYMOBAT, mode)
- Initialize backup variable

**ok\_button()**

Function to launch the processing. This function take account :

- **The Multi-processing check box if the processing has launched with multi process.** By default, this is checked. It need a computer with minimum 12Go memory.
- Append a few system value with `get_variable()`.
- **There are 3 principal check boxes :**
  - to get number download available images
  - for downloading and processing on theia platform
  - to compute optimal threshold.
  - to compute slope raster
  - for classification processing.

**open\_backup** (*test=None*)

Function to load input text in every fields. The input file must be a XML file.

**save\_backup** ()

Function to save input text in every fields. The output file must be a XML file.

**set\_variable** ()

Print number of available image from Theia's GeoJSON .

## 4.2 Control processing

**class Processing.Processing**

Main processing. This class launch the others system classes. It take into account CarHab classification method MOBA.

**This way is broken down into 3 parts :**

- Image Processing (Search, download and processing)
- Vector Processing (Optimal threshold, Sample processing)
- Classification
- Validation

**Main parameters**

**Paramètres**

- **captor\_project** (*str*) – Satellite captor name
- **classif\_year** (*str*) – Classification year
- **nb\_available\_images** (*int*) – Number download available images
- **path\_folder\_dpt** (*str*) – Main folder path
- **folder\_processing** (*str*) – Processing folder name. By default : “Traitement”
- **path\_area** (*str*) – Study area shapefile

- **path\_ortho** (*str*) – VHRS image path
- **path\_mnt** (*str*) – MNT image path
- **path\_segm** (*str*) – Segmentation shapefile

**Id information to download on theia platform****Paramètres**

- **user** (*str*) – Connexion Username
- **password** (*str*) – Connexion Password

**Output parameters****Paramètres**

- **output\_name\_moba** (*str*) – Output classification shapefile
- **out\_fieldname\_carto** (*list of str*) – Output shapefile field name
- **out\_fieldtype\_carto** (*list of str (eval ogr pointer)*) – Output shapefile field type

**Sample parameters****Paramètres**

- **fieldname\_args** (*list of str*) – Sample field names 2 by 2
- **class\_args** (*list of str*) – Sample class names 2 by 2
- **sample\_name** (*list of str*) – List of sample name (path)
- **list\_nb\_sample** (*list of int*) – Number of polygons for every sample

**Multi-processing parameters**

**Paramètres** **mp** (*int*) – Boolean variable -> 0 or 1.

- 0 means, not multi-processing
- 1 means, launch process with multi-processing

**i\_classifier\_rf()**

**Interface function to launch random forest classification with a input segmentation []**

func:*Segmentation.Segmentation*.

This function use the sklearn module to build the best of decision tree to extract classes. The optimal threshold are stored by class **rf** variable in *Processing.i\_sample\_rf()*. Then it computes zonal statistics by polygons for every images in multi-processing (if **mp** = 1).

**i\_classifier\_s()**

Interface function to launch decision tree classification with a input segmentation *Segmentation.Segmentation()*.

This function store optimal threshold by class **Segmentation.out\_threshold**. Then it computes zonal statistics by polygons for every images in multi-processing (if **mp** = 1).

**i\_download()**

Interface function to download archives on the website Theia Land. This function extract the number of downloadable image with *Archive.Archive.listing()*.

Then, this function download *Archive.Archive.download()* and unzip *Archive.Archive.decompress()* images in the archive folder (**folder\_archive**).

**i\_images\_processing()**

Interface function to launch processing VHRS images : func:*i\_vhrs* and satellite images *i\_img\_sat()* in multi-processing.

**i\_img\_sat()**

Interface function to processing satellite images:

1. Clip archive images and modify Archive class to integrate clip image path. With `Toolbox.clip_raster()` in Toolbox module.
2. Search cloud's percentage `RasterSat_by_date.RasterSat_by_date.pourc_cloud()`, select image and compute ndvi index `RasterSat_by_date.RasterSat_by_date.calcul_ndvi()`. If cloud's percentage is greater than 40%, then not select and compute ndvi index.
3. Compute temporal stats on ndvi index [min, max, std, min-max]. With `Toolbox.calc_serie_stats()` in Toolbox module.
4. Create stats ndvi raster and stats cloud raster.

```
>>> import RasterSat_by_date
>>> stats_test = RasterSat_by_date(class_archive, big_folder, one_
↳date)
>>> stats_test.complete_raster(stats_test.create_raster(in_raster, _
↳stats_data, in_ds), stats_data)
```

**i\_rpg(path\_rpg)**

Interface function to extract mono rpg crops.

@param path\_rpg: Input RPG shapefile. @type path\_rpg: str

@returns: str – variable **Rpg.vector\_used**, output no duplicated crops shapefile (path).

**i\_sample()**

Interface function to compute threshold with various sample. It also extract a list of validation layer (shapefile) to compute the precision of the next classification : func:*i\_validate*.

It create samples 2 by 2 with kwargs field names and class `Sample.Sample.create_sample()`. Then, it compute zonal statistics by polygons `Vector.Sample.zonal_stats()`.

**With zonal statistics computed, a optimal threshold is determined :**

func:*Seath.Seath.separability\_and\_threshold* that will print in a text file .lg in the main folder.

**Avertissement:** `Seath.Seath.separability_and_threshold()` does not always allow to discriminate optimal threshold. Then, this function will be launch at least ten time until it reaches a optimal threshold.

**i\_sample\_rf()**

This function build a random forest trees like model to create a final classification. All of this using the method described in the : func:*i\_validate* function and because of sklearn module.

**i\_slope()**

Interface function to processing slope raster. From a MNT, and with a command line gdal, this function compute slope in degrees `Slope.Slope()`.

**i\_tree\_direction()**

Interface function to can extract one level or two levels of the final classification

**i\_validate()**

Interface to validate a classification. It going to rasterize the validation shapefile and the classification shapefile with `layer_rasterization()`. Next, to compare pixel by pixel, the classification quality to built a confusion matrix in a csv file.

**i\_vhrs()**

Interface function to processing VHRS images. It create two OTB texture images :

func: *Vhrs.Vhrs* : SFS Texture and Haralick Texture

## 4.3 Interface element

**class** ui\_PHYMOBATs\_tab.**Ui\_PHYMOBAT**

Class to display “Simplify PHYMOBAT window”.

**class** ui\_PHYMOBATE\_tab.**Ui\_PHYMOBAT**

Class to display “Expert PHYMOBAT window”.

## 4.4 Popup about and warming

**class** ui\_A\_propos\_PHYMOBAT\_window.**Ui\_About**

Class to display « About PHYMOBAT ». In this windows, it prints informations on the processing, license and version.

**class** ui\_Warming\_study\_area.**Ui\_Warming\_study\_area**

Class to display a message to say there isn’t declared study area file.

**class** ui\_Warming\_forgetting.**Ui\_Warming\_forgetting**

Class to display a message to tell you if you forgotten to enter a raster or a sample.

**class** ui\_Proxy\_window.**Ui\_Proxy\_window**

Class to display “Proxy window”. In this windows, there is 3 lines edit to fill (proxy server, login and password).



### a

Archive, [35](#)

### p

PHYMOBAT, [46](#)

Processing, [49](#)

### r

RasterSat\_by\_date, [37](#)

Rpg, [42](#)

### s

Sample, [41](#)

Seath, [43](#)

Segmentation, [43](#)

Slope, [40](#)

### t

Toolbox, [40](#)

### u

ui\_A\_propos\_PHYMOBAT\_window, [52](#)

ui\_PHYMOBATE\_tab, [52](#)

ui\_PHYMOBATs\_tab, [52](#)

ui\_Proxy\_window, [52](#)

ui\_Warming\_forgetting, [52](#)

ui\_Warming\_study\_area, [52](#)

### v

Vector, [40](#)

Vhrs, [38](#)



## A

about\_PHYMOBA() (méthode PHYMOBAT.PHYMOBAT), 46  
 activate\_level() (méthode PHYMOBAT.PHYMOBAT), 46  
 add\_sample() (méthode PHYMOBAT.PHYMOBAT), 46  
 append\_scale() (méthode Segmentation.Segmentation), 44  
 Archive (classe dans Archive), 35  
 Archive (module), 35

## B

block\_for\_swh() (méthode PHYMOBAT.PHYMOBAT), 46

## C

calc\_serie\_stats() (méthode Toolbox.Toolbox), 40  
 calcul\_ndvi() (méthode RasterSat\_by\_date.RasterSat\_by\_date), 37  
 change\_mode() (méthode PHYMOBAT.PHYMOBAT), 46  
 check\_proj() (méthode Toolbox.Toolbox), 40  
 clear\_sample() (méthode PHYMOBAT.PHYMOBAT), 46  
 clip\_raster() (méthode Toolbox.Toolbox), 40  
 clip\_vector() (méthode Vector.Vector), 41  
 close\_button() (méthode PHYMOBAT.PHYMOBAT), 46  
 close\_data() (méthode Vector.Vector), 41  
 complete\_raster() (méthode RasterSat\_by\_date.RasterSat\_by\_date), 37  
 compute\_biomass\_density() (méthode Segmentation.Segmentation), 44  
 coord\_box\_dd() (méthode Archive.Archive), 36  
 create\_cartography() (méthode Segmentation.Segmentation), 44  
 create\_empty\_raster() (méthode RasterSat\_by\_date.RasterSat\_by\_date), 37  
 create\_new\_rpg\_files() (méthode Rpg.Rpg), 42  
 create\_sample() (méthode Sample.Sample), 42

## D

decision\_tree() (méthode Segmentation.Segmentation), 44  
 decompress() (méthode Archive.Archive), 36  
 display\_all\_levels() (méthode PHYMOBAT.PHYMOBAT), 47  
 display\_one\_level() (méthode PHYMOBAT.PHYMOBAT), 47  
 display\_two\_levels() (méthode PHYMOBAT.PHYMOBAT), 47  
 download\_auto() (méthode Archive.Archive), 36

## E

enter\_sample\_name() (méthode PHYMOBAT.PHYMOBAT), 47  
 enter\_sample\_name\_hl() (méthode PHYMOBAT.PHYMOBAT), 47  
 enter\_sample\_name\_ll() (méthode PHYMOBAT.PHYMOBAT), 47  
 extract\_slope() (méthode Slope.Slope), 40

## F

f\_output\_name\_moba() (méthode PHYMOBAT.PHYMOBAT), 47  
 f\_path\_area() (méthode PHYMOBAT.PHYMOBAT), 47  
 f\_path\_folder\_dpt() (méthode PHYMOBAT.PHYMOBAT), 47  
 f\_path\_mnt() (méthode PHYMOBAT.PHYMOBAT), 47  
 f\_path\_ortho() (méthode PHYMOBAT.PHYMOBAT), 47  
 f\_path\_segm() (méthode PHYMOBAT.PHYMOBAT), 47  
 f\_proxy() (méthode PHYMOBAT.PHYMOBAT), 47  
 field\_display\_1() (méthode PHYMOBAT.PHYMOBAT), 47  
 field\_display\_2() (méthode PHYMOBAT.PHYMOBAT), 47  
 field\_display\_3() (méthode PHYMOBAT.PHYMOBAT), 47  
 field\_display\_4() (méthode PHYMOBAT.PHYMOBAT), 47

field\_display\_5() (méthode PHYMOBAT.PHYMOBAT),  
47

field\_display\_6() (méthode PHYMOBAT.PHYMOBAT),  
48

fill\_sample() (méthode Sample.Sample), 42

forget\_raster\_sample() (méthode PHYMO-  
BAT.PHYMOBAT), 48

forget\_study\_area() (méthode PHYMO-  
BAT.PHYMOBAT), 48

## G

get\_variable() (méthode PHYMOBAT.PHYMOBAT), 48

group\_by\_date() (méthode Raster-  
Sat\_by\_date.RasterSat\_by\_date), 37

## H

haralick\_texture\_extraction() (méthode Vhrs.Vhrs), 38

help\_tools() (méthode PHYMOBAT.PHYMOBAT), 48

## I

i\_classifier\_rf() (méthode Processing.Processing), 50

i\_classifier\_s() (méthode Processing.Processing), 50

i\_download() (méthode Processing.Processing), 50

i\_images\_processing() (méthode Processing.Processing),  
50

i\_img\_sat() (méthode Processing.Processing), 50

i\_rpg() (méthode Processing.Processing), 51

i\_sample() (méthode Processing.Processing), 51

i\_sample\_rf() (méthode Processing.Processing), 51

i\_slope() (méthode Processing.Processing), 51

i\_tree\_direction() (méthode Processing.Processing), 51

i\_validate() (méthode Processing.Processing), 51

i\_vhrs() (méthode Processing.Processing), 51

img\_sample\_name() (méthode PHYMO-  
BAT.PHYMOBAT), 48

initUI() (méthode PHYMOBAT.PHYMOBAT), 48

## L

layer\_rasterization() (méthode Vector.Vector), 41

listing() (méthode Archive.Archive), 36

## M

mono\_rpg() (méthode Rpg.Rpg), 42

mosaic\_by\_date() (méthode Raster-  
Sat\_by\_date.RasterSat\_by\_date), 37

## O

ok\_button() (méthode PHYMOBAT.PHYMOBAT), 49

open\_backup() (méthode PHYMOBAT.PHYMOBAT),  
49

## P

PHYMOBAT (classe dans PHYMOBAT), 46

PHYMOBAT (module), 46

pourc\_cloud() (méthode Raster-  
Sat\_by\_date.RasterSat\_by\_date), 38

Processing (classe dans Processing), 49

Processing (module), 49

## R

raster\_data() (méthode Raster-  
Sat\_by\_date.RasterSat\_by\_date), 38

RasterSat\_by\_date (classe dans RasterSat\_by\_date), 37

RasterSat\_by\_date (module), 37

Rpg (classe dans Rpg), 42

Rpg (module), 42

## S

Sample (classe dans Sample), 41

Sample (module), 41

save\_backup() (méthode PHYMOBAT.PHYMOBAT), 49

Seath (classe dans Seath), 43

Seath (module), 43

Segmentation (classe dans Segmentation), 43

Segmentation (module), 43

select\_random\_sample() (méthode Sample.Sample), 42

separability\_and\_threshold() (méthode Seath.Seath), 43

set\_list\_archive\_to\_try() (méthode Archive.Archive), 36

set\_variable() (méthode PHYMOBAT.PHYMOBAT), 49

sfs\_texture\_extraction() (méthode Vhrs.Vhrs), 39

Slope (classe dans Slope), 40

Slope (module), 40

## T

Toolbox (classe dans Toolbox), 40

Toolbox (module), 40

## U

ui\_A\_propos\_PHYMOBAT\_window (module), 52

Ui\_About (classe dans  
ui\_A\_propos\_PHYMOBAT\_window), 52

Ui\_PHYMOBAT (classe dans ui\_PHYMOBATe\_tab), 52

Ui\_PHYMOBAT (classe dans ui\_PHYMOBATs\_tab), 52

ui\_PHYMOBATe\_tab (module), 52

ui\_PHYMOBATs\_tab (module), 52

Ui\_Proxy\_window (classe dans ui\_Proxy\_window), 52

ui\_Proxy\_window (module), 52

Ui\_Warming\_forgetting (classe dans  
ui\_Warming\_forgetting), 52

ui\_Warming\_forgetting (module), 52

Ui\_Warming\_study\_area (classe dans  
ui\_Warming\_study\_area), 52

ui\_Warming\_study\_area (module), 52

utm\_to\_latlng() (méthode Archive.Archive), 36

## V

Vector (classe dans Vector), 40

Vector (module), [40](#)  
vector\_data() (méthode Vector.Vector), [41](#)  
Vhrs (classe dans Vhrs), [38](#)  
Vhrs (module), [38](#)  
vrt\_translate\_gdal() (méthode Raster-  
Sat\_by\_date.RasterSat\_by\_date), [38](#)

## Z

zonal\_stats() (méthode Vector.Vector), [41](#)  
zonal\_stats\_pp() (méthode Vector.Vector), [41](#)