MODIS CLASSIFICATION CHAIN

Steps for LULC classification with MOD13Q1 data

Prelude

Before you begin

a Make sure that the "bin" directory and its subfolders, which contains the classification programs (e.g., sequence_classification_apply) is included in the system path

b Also make sure that the "scripts" directory, which contains scripts and files used throughout the processing chain (e.g., hdf2hdfraw, hdf2hdfraw.prm) is included in the system path

c You should have a georeferenced image file in INRIMAGE format that specifies the region of interest (ROI). This image should be a mask where zero indicates pixels outside the ROI, whereas non-zero values indicate pixels belonging to the ROI

d You should have reference information regarding the classes to be identified:

- a MODIS classification.
- masks of the training and testing areas.

Acquire MOD13Q1 data

a) Login to EOS Data Gateway (http://edcimswww.cr.usgs.gov/pub/imswelcome/)

b) Specify desired area and annual period (e.g. 2005-08-01 up to 2006-07-31)

c) Data can be obtained via FTP or DVD in HDF-EOS format

1Preprocessing

1.1 Process HDF-EOS data

Convert original data in HDF-EOS format to INRIMAGE format and subset that data to include only the area within the region of interest.

1.1.1 Use MODIS Reprojection Tool (MRT)

MRT should be used for the following tasks:

Reproject data from original MODIS Sinusoidal projection to Geographic (unprojected) coordinates

Convert original HDF-EOS data to HDF-RAW format

Subset the original data if the area of interest is smaller than a MODIS tile (e.g. High Taquari Basin)

Mosaic several MODIS tiles if the area of interest encompasses many tiles (e.g. Rio de la Plata Basin)

Discard layers not to be used (optional)

Example using scripts:

```
hdf2hdfraw hdf/ hdfraw/
```

will convert all HDF data in sinusoidal projection to HDF-RAW data in geographic projection; input data will be read from the “hdf” directory and output data will be written to the “hdfraw” directory).

1.1.2 Convert HDF-RAW data to INRIMAGE format

Use hdfraw2inr program to convert all HDF-RAW .dat files to INRIMAGE files. Resulting data will consist of .inr and .hdr pairs of files that are readable with INRIMAGE and ENVI software.

Example:

```
hdfraw2inr -idir hdfraw/ -odir inr/
```

will convert all HDF-RAW data in “hdfraw” directory and write output files to the “inr” directory.

1.1.3 Crop data to region of interest (ROI)

Use inr_crop program to crop all INRIMAGE images to the region of interest (ROI). This requires the existence of a georeferenced mask image representing the ROI (mask is zero for pixels outside the ROI, non-zero otherwise)

Example:

```
inr_crop -idir inr/ -odir inr/ -mask START/masktaquari-modis.inr
```

will crop all images in the “inr” directory to the ROI defined by image file “masktaquari_modis.inr”; resulting cropped images will overwrite the original ones.

1.2 Build temporal sequences

Use inr_sequence program to build temporal sequences or temporal stacks. The resulting sequence should contain one layer for each time frame available.

1.2.1 Build NDVI or EVI temporal sequence

An NDVI or EVI temporal sequence should be built for each one-year period. Thus, considering that MOD13Q1 data are 16-day composites, each sequence should be composed of 24 layers.

Example:

```
inr_sequence –idir inr/ -ofile seq/sequence.ndvi.inr -ifilter NDVI
```

will build a sequence composed of all files in directory “inr” whose name contains the filter string “NDVI.”; the resulting sequence file will be written to the directory “seq”.

1.2.2 Build temporal sequence of NDVI or EVI per-pixel quality data

This quality temporal sequence should contain NDVI/EVI metadata describing each pixel’s quality at each time frame, and should thus be the same size as the temporal sequence built in step 1.2.1.

Example:

```
inr_sequence –idir inr/ -ofile seq/sequence.ndvi_quality.inr
-ifilter NDVI_Quality
```

1.2.3 Interpret NDVI/EVI quality data to generate a temporal quality mask file

The per-pixel quality data is composed of a series of bits that encode several quality-related information (e.g., presence of clouds, aerosols, overall sensor quality, etc.). This information must be interpreted so as to generate a mask indicating which data should be used and which data should be discarded. In order to perform this interpretation, the modis_VIquality2mask program should be used.

Example:

```
modis_VIquality2mask -idir seq/ -ifilter ndvi_quality -odir seq/
```

will convert all quality files located in directory “seq” (as specified by the filter string “ndvi_quality”) and output the corresponding mask files in the same “seq” directory (note that the program automatically changes the name of the files).

1.2.4 Assert that NDVI/EVI data is within –1/+1 range

MOD13Q1 VI products usually come in integer format, so that VI values are given in the range –10000 to 10000. To convert these values to the usual –1/+1 range, use inr_multiply program.

Example:

```
inr_multiply -ifile seq/sequence.ndvi.inr -mult 0.0001
-ofile seq/sequence.ndvi.mult.inr
```

1.2.5 Correct NDVI/EVI data based on the temporal quality mask

Given the quality mask computed in step 1.2.3 and the NDVI/EVI data resulting from step 1.2.4, the sequence_cloudcorrection program should be used to suppress low quality data and optionally perform additional filtering operations, so as to compute a “corrected” version of the temporal VI data. Suppressed data are replaced using linear temporal interpolation.

Example:

```
sequence_cloudcorrection -ifile seq/sequence.ndvi.mult.inr
-mask seq/sequence.qualitymask.inr -roi START/masktaquari-modis.inr
-ofile seq/sequence.ndvi.corrected.inr
```

will suppress all data from file “sequence.ndvi.inr” whose corresponding pixel in mask file “sequence.qualitymask.inr” is labeled as low quality (value zero); corrected data using linear temporal interpolation will be written to output file “sequence.ndvi.corrected.inr”.

1.2.6 Build list of dates corresponding to each image of the sequence

Use modis_datelist to build a list of dates where each entry corresponds to the date of the corresponding image in the sequence. This is to be done by examining the original names of the MODIS files, which must follow the standard MODIS naming convention for Level 2 data namely MODxxxxx.AYYYYDDD.*(e.g.,“MOD09GQK.A2001211.h12v10.004.2003131015044”)

Example:

```
modis_datelist -idir inr/ -ofile dates.ddd.txt -ddd
```

will examine all files located in the “inr” directory and extract date information from their names, writing the output to the text file “dates.ddd.txt” in DDD format (e.g., day of year).

2 Operational chains

2.1 chain10F-ML

2.1.1 Compute profile features

Using the corrected NDVI/EVI temporal sequence data, process each pixel's temporal profile in order to compute 10F features or attributes that describe each pixel's temporal response.

Example:

```
sequence_profile_features -ifile seq/sequence.ndvi.corrected.inr
-roi START/masktaquari-modis.inr -ofile features/features10F.ndvi.inr
-dates dates.ddd.txt -nodata -9999
```

will compute temporal features for the NDVI sequence specified by file "sequence.ndvi.corrected.inr", writing the output to file "features10F.ndvi.inr"; if a specific feature cannot be computed (missing data), then -9999 will be assigned as a value.

2.1.2 Normalize feature data

The maximum likelihood classification algorithm requires input data to be normalized. For that matter, the inr_normalize program should be used to normalize the values computed for each feature for the entire region of interest.

Example:

```
inr_normalize -ifile features/features10F.ndvi.inr
-roi masktaquari-modis.inr -ofile features/features10F.ndvi.norm.inr
-nodata -9999
```

will normalize each layer of the input feature file "features10F.ndvi.inr", taking into consideration that –9999 is to be considered as missing data, writing the output to file "features10F.ndvi.norm.inr"; missing data will be given random values in the output file

2.1.3 Train classifier based on training data

Based on the input reference classification image use computed normalized temporal features to train a supervised classification algorithm, so that the typical characteristics of each class of interest can be learned.

The reference classification image, classification.10F.train.inr,is a learning area (southern part) image for which the low resolution class is known for every pixels.

Example:

```
sequence_classification_train -ifile features/features10F.ndvi.norm.inr
-class LEARNING/classification.10F.train.inr
-ofile LEARNING/chain10F-ML/classdef-10F-ML.txt
```

will analyze the data in file "features10F.ndvi.norm.inr" for the samples of each class as provided by file "classification.10F.train.inr"; the program will thus learn the feature distributions typical of each class, writing output class definitions to file "classdef-10F-ML.txt"

2.1.4 Apply classifier to the testing area (northern part)

Once the class definitions have been produced, the program sequence_classification_apply can be used to determine the classes of all pixels within the region of interest. This program implements a maximum likelihood supervised classification algorithm. Optionally, a minimum *a posteriori* probability tolerance value can be given for assigning a class to each point, so that if, for one point, the class with maximum probability does not exceed this value, that point is left unclassified.

Example:

```
sequence_classification_apply -ifile features/features10F.ndvi.norm.inr
-roi START/masktaquari-modis-north.inr
-def LEARNING/chain10F-ML/classdef-10F-ML.txt
-ofile OPERATIONAL/chain10F-ML/classification.10F-ML.over95.inr
-tol 0.95
```

will classify the data from file “features10F.ndvi.norm.inr” using the class definitions provided in file “classdef-10F-ML.txt”, writing the output to file “classification.10F-ML.over95.inr”; a minimum tolerance value of 95% is specified for the *a posteriori* probability, so that all pixels with *a posteriori* probability under 95% are considered to be of low confidence and are left unclassified

2.1.5 Translate classification result to ground truth legend

The results from the classification application have to be merged or translated using the inr_translate program, so as to make these data compatible with the ground truth data provided.

Example:

```
inr_translate
-ifile OPERATIONAL/chain10F-ML/classification.10F-ML.over95.inr
-trans START/trans.confusion.10F.txt
-ofile OPERATIONAL/chain10F-ML/classification.10F-ML.over95.trans.inr
```

1

2.2 chain12F-ML

2.2.1 Compute profile features

Using the corrected NDVI/EVI temporal sequence data, process each pixel's temporal profile in order to compute 12F features or attributes that describe each pixel's temporal response.

Example:

```
sequence_profile_features_upgraded -ifile seq/sequence.ndvi.corrected.inr
-roi START/masktaquari-modis.inr -ofile features/features12F.ndvi.inr
-dates dates.ddd.txt -nodata -9999
```

will compute temporal features for the NDVI sequence specified by file "sequence.ndvi.corrected.inr", writing the output to file "features12F.ndvi.inr"; if a specific feature cannot be computed (missing data), then -9999 will be assigned as a value.

2.2.2 Normalize feature data

The maximum likelihood classification algorithm requires input data to be normalized. For that matter, the inr_normalize program should be used to normalize the values computed for each feature for the entire region of interest.

Example:

```
inr_normalize -ifile features/features12F.ndvi.inr
-roi START/masktaquari-modis.inr
-ofile features/features12F.ndvi.norm.inr -nodata -9999
```

will normalize each layer of the input feature file "features12F.ndvi.inr", taking into consideration that –9999 is to be considered as missing data, writing the output to file "features12F.ndvi.norm.inr"; missing data will be given random values in the output file

2.2.3 Train classifier based on training data

Based on the input reference classification image use computed normalized temporal features to train a supervised classification algorithm, so that the typical characteristics of each class of interest can be learned.

The reference classification image, classification.12F.train.inr,is a learning area (southern part) image for which the low resolution class is known for every pixels.

Example:

```
sequence_classification_train -ifile features/features12F.ndvi.norm.inr
-class LEARNING/classification.12F.train.inr
-ofile LEARNING/chain12F-ML/classdef-12F-ML.txt
```

will analyze the data in file "features12F.ndvi.norm.inr" for the samples of each class as provided by file "classification.12F.train.inr"; the program will thus learn the feature distributions typical of each class, writing output class definitions to file "classdef-12F-ML.txt"

2.2.4 Apply classifier to the testing area (northern part)

Once the class definitions have been produced, the program sequence_classification_apply can be used to determine the classes of all pixels within the region of interest. This program implements a maximum likelihood supervised classification algorithm. Optionally, a minimum *a posteriori* probability tolerance value

can be given for assigning a class to each point, so that if, for one point, the class with maximum probability does not exceed this value, that point is left unclassified.

Example:

```
sequence_classification_apply -ifile features/features12F.ndvi.norm.inr
-roi START/masktaquari-modis-north.inr
-def LEARNING/chain12F-ML/classdef-12F-ML.txt
-ofile OPERATINAL/chain12F-ML/classification.12F-ML.over95.inr
-tol 0.95
```

will classify the data from file “features12F.ndvi.norm.inr” using the class definitions provided in file “classdef-12F-ML.txt”, writing the output to file “classification.12F-ML.over95.inr”; a minimum tolerance value of 95% is specified for the *a posteriori* probability, so that all pixels with *a posteriori* probability under 95% are considered to be of low confidence and are left unclassified

2.2.5 Translate classification result to ground truth legend

The results from the classification application have to be merged or translated using the inr_translate program, so as to make these data compatible with the ground truth data provided.

Example:

```
inr_translate
-ifile OPERATINAL/chain12F-ML/classification.12F-ML.over95.inr
-trans START/trans.confusion.12F.txt
-ofile OPERATINAL/chain12F-ML/classification.12F-ML.over95.trans.inr
```

2.3 chain12F-MAHA

2.3.1 Compute profile features

Using the corrected NDVI/EVI temporal sequence data, process each pixel's temporal profile in order to compute 12F features or attributes that describe each pixel's temporal response.

Example:

```
sequence_profile_features_upgraded -ifile seq/sequence.ndvi.corrected.inr
-roi START/masktaquari-modis.inr -ofile features/features12F.ndvi.inr
-dates dates.ddd.txt -nodata -9999
```

will compute temporal features for the NDVI sequence specified by file "sequence.ndvi.corrected.inr", writing the output to file "features12F.ndvi.inr"; if a specific feature cannot be computed (missing data), then -9999 will be assigned as a value.

2.3.2 Train classifier based on training data

In this chain the classification strategy use is a based on a Chi-square distribution and the Mahalanobis distance, using a specific subset of features for each class.

The sequence_classification_train_maha software need a configuration files which specifies the number of features useful for a class and the features to use for each class.

Example :

```
sequence_classification_train_maha -ifile features12F.ndvi.inr
-class LEARNING/classification.12F.train.inr
-config START/config.maha.txt
-ofile LEARNING/chain12F-MAHA/classdef-12F-MAHA.txt
```

will analyze the data in file "features12F.ndvi.inr" for the samples of each class provided by "classification.12F.train.inr"; the program will thus define each class, writing output class definitions to file "classdef-12F-MAHA.txt".

2.3.3 Compute memberships

Based on the previously selected strategy and in the classes definitions, a membership value to each class is computed for every pixel.

Example:

```
sequence_membership_maha -ifile features/features12F.ndvi.inr
-roi START/masktaquari-modis.inr
-def LEARNING/chain12F-MAHA/classdef-12F-MAHA.txt
-omemb LEARNING/chain12F-MAHA/memberships_maha.inr
```

Will compute the membership according the chain strategy (takes the necessary information from the "classdef-12F-MAHA.txt" file).

2.3.4 Apply classifier to the testing area (northern part)

Once the class definitions have been produced, the program sequence_fuzzyclassification_apply can be used to determine the classes of all pixels within the region of interest. This program implements a maximum likelihood classification algorithm. Optionally, a membership tolerance value can be given for assigning a class to a point, only if, the class with maximum membership exceeds this value.

Example:

```
sequence_fuzzyclassification_apply
-ifile LEARNING/chain12F-MAHA/membership_maha.inr
-roi START/masktaquari-modis-north.inr
-ofile OPERATINAL/chain12F-MAHA/classification.12F-MAHA.over80.inr
-tol 0.80
```

will classify the data from file “membership_maha.inr”, writing the output to file “classification.12F-MAHA.over80.inr”; a minimum tolerance value of 0.8 is specified for the membership, so that all pixels with membership values under 0.8 are considered to be of low confidence and are left unclassified.

2.3.5 Translate classification result to ground truth legend

The results from the classification application have to be merged or translated using the inr_translate program, so as to make these data compatible with the ground truth data provided.

Example:

```
inr_translate
-ifile OPERATINAL/chain12F-MAHA/classification.12F-MAHA.over80.inr
-trans START/trans.confusion.12F.txt
-ofile OPERATINAL/chain12F-MAHA/classification.12F-MAHA.over80.trans.inr
```

2.4 chain12F-FUZZY-WA

2.4.1 Compute profile features

Using the corrected NDVI/EVI temporal sequence data, process each pixel's temporal profile in order to compute 12F features or attributes that describe each pixel's temporal response.

Example:

```
sequence_profile_features_upgraded -ifile seq/sequence.ndvi.corrected.inr
-roi START/masktaquari-modis.inr -ofile features/features12F.ndvi.inr
-dates dates.ddd.txt
-nodata -9999
```

will compute temporal features for the NDVI sequence specified by file "sequence.ndvi.corrected.inr", writing the output to file "features12F.ndvi.inr"; if a specific feature cannot be computed (missing data), then -9999 will be assigned as a value.

2.4.2 Train classifier based on training data

In this chain the classification strategy use Fuzzy logic approach and Weighted Average for information fusion.

The sequence_classification_train_wa software need a configuration files which provides the weight matrix (lines correspond to classes, colums to features).

Example :

```
sequence_classification_train_wa -ifile features12F.ndvi.inr
-class LEARNING/classification.12F.train.inr
-config START/config.FusionWA.txt
-ofile LEARNING/chain12F-FUZZY-WA/classdef-12F-FUZZY-WA.txt
```

will analyze the data in file "features12F.ndvi.inr" for the samples of each class provided by "classification.12F.train.inr"; the program will thus define each class, writing output class definitions to file "classdef-12F-FUZZY-WA.txt".

2.4.3 Compute memberships

Based on the previously selected strategy and in the classes definitions, a membership value to each class is computed for every pixel.

Example:

```
sequence_membership_wa -ifile features/features12F.ndvi.inr
-roi START/masktaquari-modis.inr
-def LEARNING/chain12F-FUZZY-WA/classdef-12F-FUZZY-WA.txt
-omemb LEARNING/chain12F-FUZZY-WA/memberships_wa.inr
```

Will compute the membership according the chain strategy (takes the necessary information from the "classdef-12F-FUZZY-WA.txt" file).

2.4.4 Apply classifier to the testing area (northern part)

Once the class definitions have been produced, the program sequence_fuzzyclassification_apply can be used to determine the classes of all pixels within the region of interest. This program implements a maximum likelihood classification algorithm. Optionally, a membership tolerance value can be given for assigning a class to a point, only if, the class with maximum membership exceeds this value.

Example:

```
sequence_fuzzyclassification_apply
-ifile LEARNING/chain12F-FUZZY-WA/membership_wa.inr
-roi START/masktaquari-modis-north.inr
-ofile OPERATIONAL/chain12F-FUZZY-WA/classification.12F-FUZZY-
WA.over32.inr -tol 0.32
```

will classify the data from file “membership_wa.inr”, writing the output to file “classification.12F-FUZZY-WA.over32.inr”; a minimum tolerance value of 0.32 is specified for the membership, so that all pixels with membership values under 0.32 are considered to be of low confidence and are left unclassified.

2.4.5 Translate classification result to ground truth legend

The results from the classification application have to be merged or translated using the inr_translate program, so as to make these data compatible with the ground truth data provided.

Example:

```
inr_translate
-ifile OPERATIONAL/chain12F-FUZZY-WA/classification.12F-FUZZY-
WA.over32.inr
-trans START/trans.confusion.12F.txt
-ofile OPERATIONAL/chain12F-FUZZY-WA/classification.12F-FUZZY-
WA.over32.trans.inr
```

2.5 chain12F-FUZZY-DS

2.5.1 Compute profile features

Using the corrected NDVI/EVI temporal sequence data, process each pixel's temporal profile in order to compute 12F features or attributes that describe each pixel's temporal response.

Example:

```
sequence_profile_features_upgraded -ifile seq/sequence.ndvi.corrected.inr
-roi START/masktaquari-modis.inr -ofile features/features12F.ndvi.inr
-dates dates.ddd.txt -nodata -9999
```

will compute temporal features for the NDVI sequence specified by file "sequence.ndvi.corrected.inr", writing the output to file "features12F.ndvi.inr"; if a specific feature cannot be computed (missing data), then -9999 will be assigned as a value.

2.5.2 Train classifier based on training data

In this chain the classification strategy use Fuzzy logic approach and Dempster-Shafer rule of combination for information fusion.

The sequence_classification_train_ds software need a configuration files which specifies the number of features useful for a class and the features to use for each class.

Example :

```
sequence_classification_train_ds -ifile features12F.ndvi.inr
-class LEARNING/classification.12F.train.inr
-config START/config.FusionDS.txt
-ofile LEARNING/chain12F-FUZZY-DS/classdef-12F-FUZZY-DS.txt
```

will analyze the data in file "features12F.ndvi.inr" for the samples of each class provided by "classification.12F.train.inr"; the program will thus define each class, writing output class definitions to file "classdef-12F-FUZZY-DS.txt".

2.5.3 Compute memberships

Based on the previously selected strategy and in the classes definitions, a membership value to each class is computed for every pixel.

Example:

```
sequence_membership_ds -ifile features/features12F.ndvi.inr
-roi START/masktaquari-modis.inr
-def LEARNING/chain12F-FUZZY-DS/classdef-12F-FUZZY-DS.txt
-omemb LEARNING/chain12F-FUZZY-DS/memberships_ds.inr
```

Will compute the membership according the chain strategy (takes the necessary information from the "classdef-12F-FUZZY-DS.txt" file).

2.5.4 Apply classifier to the testing area (northern part)

Once the class definitions have been produced, the program sequence_fuzzyclassification_apply can be used to determine the classes of all pixels within the region of interest. This program implements a maximum likelihood classification algorithm. Optionally, a membership tolerance value can be given for assigning a class to a point, only if, the class with maximum membership exceeds this value.

Example:

```
sequence_fuzzyclassification_apply
-ifile LEARNING/chain12F-FUZZY-DS/membership_ds.inr
-roi START/masktaquari-modis-north.inr
-ofile OPERATIONAL/chain12F-FUZZY-DS/classification.12F-FUZZY-
DS.over80.inr -tol 0.8
```

will classify the data from file “membership_wa.inr”, writing the output to file “classification.12F-FUZZY-DS.over80.inr”; a minimum tolerance value of 0.8 is specified for the membership, so that all pixels with membership values under 0.8 are considered to be of low confidence and are left unclassified.

2.5.5 Translate classification result to ground truth legend

The results from the classification application have to be merged or translated using the inr_translate program, so as to make these data compatible with the ground truth data provided.

Example:

```
inr_translate
-ifile OPERATIONAL/chain12F-FUZZY-DS/classification.12F-FUZZY-
DS.over80.inr
-trans START/trans.confusion.12F.txt
-ofile OPERATIONAL/chain12F-FUZZY-DS/classification.12F-FUZZY-
DS.over80.trans.inr
```

3 Evaluate classification performance

3.1 Create ground truth image including deforestation

3.1.1 Create 4 classes truth image

The ground truth image is created using the classification LANDSAT resized in the modis format “classification_14classes_truth.inr” and the inr_translate function.

Example :

```
inr_translate -ifile START/classification_14classes_truth.inr
-trans START/translate_14to4.txt
-ofile OPERATIONAL/truth_4classes.inr
```

The translation file should be the following :

```
1 1
2 0
3 0
4 2
5 2
6 2
7 2
8 2
9 2
10 2
11 2
12 2
13 3
14 4
```

3.1.2 Include partially deforestation

The ground truth image and a mask of partially deforestation can be merged in order to include deforestation in the ground truth.

Example :

```
inr_translate_defor -ifile OPERATIONAL/truth_4classes.inr
-trans START/translate_5to5.txt
-maskdefor START/mask_deforesation.inr -ofile OPERATIONAL/truth.inr
```

The translation file should be the following :

```
1 1
2 2
3 3
4 4
5 5
```

3.2 Compute statistics based on test data

A quantitative evaluation of the classification performance can be done with the sequence_classification_stats program, so as to compare the results computed with the methodology, based on ground truth (e.g., samples not included in the training data but only in the testing area). The resulting statistics include the percentage of pixels of each class that were classified, as well as the confusion matrix.

Example:

```
sequence_classification_stats
-ifile OPERATIONAL/chain12F-FUZZY-DS/classification.12F-FUZZY-
DS.over80.trans.inr
-truth OPERATIONAL/truth.inr -roi START/masktaquari_modis.north.inr
```

will compare the file “classification.12F-FUZZY-DS.over80.trans.inr” with the ground truth in file “truth.inr”; the resulting statistics (confusion matrix) are written on the screen.