Préface

Dans la continuité de l'*UNEP atlas of our changing environment*, le *Global Resource Information Database* (GRID) est en train de mettre en place un processus de suivi automatique de sites à l'aide d'images satellites (LiMES – *Live Monitoring of Earth Surface*, `http://limes.grid.unep.ch/`) à l'aide de logiciels SIG et télédétection open sources (GRASS, R, etc.). C'est dans le cadre du Certificat complémentaire en géomatique de l'Université de Genève qu'il m'a été offert la possibilité de réaliser un stage de 3 mois lié à ce projet, au sein du GRID, sous la direction de Bruno Chatenoux et Karin Allenbach. Les objectifs de celui-ci ont consisté à **i)** rechercher et tester plusieurs méthodes permettant de détecter les couvertures nuageuses sur des images Landsat 8, **ii)** intégrer la solution la plus adaptée et efficace au processus de traitement LiMES, **iii)** automatiser la gestion des données manquantes résultantes des masques de nuages, et **iv)** automatiser différentes analyses liées à l'indice de végétation par différence normalisé (NDVI) sur une série d'images multitemporelles. L'intégration de plusieurs solutions open sources tels les logiciels GRASS et R ou les langages Unix Bash et Python ont permis d'aborder les processus de manière flexible. Un exemple est le script relatif aux analyses de NDVI codé en langage Python mais intégrant des fonctions R et le langage de composition de document LaTeX.

Ce rapport, rédigé en deux parties, décrit les différentes méthodologies employées associées à chaque problématique, illustre les solutions adoptées par des exemples concrets et contient toutes les instructions nécessaires à leur exécution [1].

Partie I
Images Landsat et nuages

Depuis 1972, les images à haute résolution spatiale obtenues par les satellites Landsat ont largement contribué au suivi de l'évolution de la couverture du sol à l'échelle globale (Williams *et al.* 2006). Cependant, la présence de nuages et de l'ombre projetée de ces derniers compromettent l'analyse de ces images (Ju & Roy 2008), notamment en biaisant l'estimation de l'indice de végétation par différence normalisé (NDVI) et en faussant la classification de la couverture du sol (Zhu & Woodcock 2012). En effet, les nuages épais peuvent obstruer la réflectance de la surface terrestre et les nuages fins, généralement les cirrus, peuvent corrompre la qualité des données et leur interprétation (Meng Xu *et al.* 2014; Rajitha, K. *et al.* 2015).

1 Détection des nuages et masquage

La première étape permettant de pallier ce problème consiste à créer un masquage. En effet, étant donné que les pixels associés aux nuages peuvent fausser les résultats, ils doivent

1. La marche à suivre des installations sous Linux des différents logiciels et librairies se trouve dans l'Annexe A. L'exécution des différents scripts est expliquée dans les sections correspondantes et ces derniers sont disponibles dans les Annexes correspondantes.

être exclus du processus de traitement.

Landsat 8 contient la bande *Quality Assessment* (QA) laquelle traduit un certain nombre d'états relatifs à la surface du sol, à l'atmosphère et au capteur pouvant affecter l'utilisabilité du pixel considéré. La valeur de chaque pixel doit être traduite en format binaire de 16 bits et les bits doubles (12-13 et 14-15) indique avec un certain niveau de confiance (non-défini, non, peut-être, oui) la présence de cirrus et de nuages. L'équipe de *Land Data Operational Product Evaluation* (LDOPE) développe et maintient plusieurs outils destinés à manipuler, visualiser et analyser des données de l'instrument *Moderate Resolution Imaging Spectroradiometer* (MODIS) placé dans les satellites Terra et Aqua. Un sous-ensemble de ces outils permettant notamment la création de masques et pouvant être appliqué aux images Landsat a été extrait et compilé sous le nom de Landsat LDOPE Toolbelt (USGS 2013). La bande QA peut être utilisée comme input dans la fonction *unpack_oli_qa* de l'outil L-LDOPE afin de définir un masque de nuages selon un intervalle de confiance défini (low, medium, high). Cependant, selon Shen *et al.* (2015), la seule utilisation de la bande QA ne permet pas d'identifier les nuages de manière adéquate. Il ont en effet trouvé que la majorité de pixels nuageux ont été mal classifiés.

La création d'un masque est aussi possible grâce à un algorithme appelé Fmask (Function of mask) permettant de détecter de manière précise les nuages, leurs ombres et la neige sur des images Landsat, en utilisant la réflectance au sommet de l'atmosphère (TOA) et la température de brillance (BT) (Zhu & Woodcock 2012). Cette fonction utilise des règles basées sur les propriétés physiques des nuages afin de séparer les pixels potentiels de nuage et les pixels de ciel clair. Parallèlement, les probabilités normales de température, de variation spectrale et de brillance sont combinées afin d'obtenir une probabilité de couverture de nuage. La couche de nuage est dérivée de cette dernière et des pixels potentiels de nuage calculé précédemment. Finalement, les angles de vue et d'illumination sont utilisés pour déterminer l'ombrage des nuages.

Alors que les nuages épais sont relativement faciles à détecter, les nuages fins, en particulier les cirrus, sont moins visibles (Meng Xu *et al.* 2014). Pourtant, le grand nombre de cristaux fins formant ces nuages module le transfert de rayonnement à travers des mécanismes d'absorption et de dispersion et peut biaiser les analyses de couverture du sol et autres observations environnementales. Depuis 2013, la bande spectrale 9 de Landsat 8 (*Cirrus band* : 1'360-1'390 µm) permet de détecter en haute résolution la présence de nuages de haute altitude et de corriger l'image. Des algorithmes basés sur la relation linéaire entre la réflectance à 1'380 µm et l'effet des cirrus sur la réflectance dans les longueurs d'ondes allant de 400 à 1'000 µm ont été créés afin de supprimer l'effet de ces nuages fins sur la réflectance réelle (ex. Gao *et al.* 1998; Rajitha, K. *et al.* 2015).

En ce sens, Zhu *et al.* (2015) ont amélioré l'algorithme Fmask, et celui-ci tient désormais compte de la *Cirrus band*, lorsqu'elle est disponible, dans le calcul de la probabilité de la couverture de nuages, obtenant ainsi de meilleurs résultats.

Voici une liste non-exhaustive des différents programmes/fonctions disponibles associés à la problématique des nuages dans le cadre d'analyses d'images satellites.

- La fonction Fmask (Zhu *et al.* 2015), disponible pour Matlab ou pouvant être exécutée directement sur Linux ou Windows (`https://code.google.com/archive/p/fmask/`).

Utile pour des images Landsat 4,5,7,8 et Sentinel 2.
- Pour des images Landsat 8, la bande QA permet d'identifier les nuages épais et les cirrus dans un certain intervalle de confiance. La création de masquage peut-être facilitée avec l'outil L-LDOPE Toolbelt (USGS 2013).
- La fonction GRASS *i.landsat.acca* utilise l'algorithme *Automated Cloud-Cover Assessment* (ACCA) (Irish *et al.* 2006). Utile pour des images Landsat TM/ETM+.
- La fonction *clouds* du package 'landsat' (R) basé sur les travaux de Martinuzzi *et al.* (2007). Utile pour des images Landsat ETM+. Ne prends pas en compte les ombres des nuages.
- La fonction *cloudMask* du package 'RStoolbox' (R) déstiné aux images Landsat utilise la bande spectrale bleue et la bande thermale. Ne prend pas en compte l'ombrage des nuages et ne fonctionne pas sans les bande TIRS, ce qui peut être problématique pour certaines scènes (voir section 3.2). Aussi, elle ne prend pas en compte les cirrus.

2 Tests

2.1 Méthode

La fonction Fmask et la fonction *unpack_oli_qa* de l'outil de L-LDOPE Toolbelt ont été testées en terme de vitesse et de précision et comparées sur une série de 12 scènes aux couvertures nuageuses variables. Pour le téléchargement, les services *cloud* d'Amazon et de Google ont été utilisés de manière automatique grâce au script *LiMES_batcher_v003.py* (voir Annexe B). La taille des fichiers et le temps de téléchargement et d'exécution des deux méthodes ont été pris en compte. Deux paramétrages différents ont été testés pour la fonction Fmask (seuil de probabilité de nuage à 22.5% et à 50%, voir section 3.3) et l'intervalle de confiance *medium* a été choisi pour la fonction *unpack_oli_qa*. Les output ont été traités dans GRASS (*raster calculator*) afin de pouvoir normaliser le masquage et comparer les deux méthodes (clear = 0, cloud = 1, cloud shadow = 2). Dans R, la fonction *freq* du package 'raster' a été utilisée pour calculer la fréquence de chaque valeur. Le script bash ayant permis d'automatiser en partie les différents processus liés au test se trouve dans l'Annexe C.1.

2.2 Résultats

Les résultats bruts se trouvent sous forme de tableaux, dans l'Annexe C.2.

2.2.1 Téléchargement

Trois scènes sur 12 n'ont pas pu être téléchargées sur Amazon (dates antérieures à mai 2014). De part leur compression LZW, la taille des scènes téléchargées sur Amazon a été environ 2 fois plus petite que celle des scènes téléchargées sur Google et le taux de transfert environ a été 2 fois plus rapide sur Amazon. Le temps de téléchargement a été donc environ 4 fois plus rapide sur Amazon que sur Google. Cependant, la fonction *unpack_oli_qa* n'a pas fonctionné avec les images compressées téléchargées sur Amazon.

2.2.2 unpack_oli_qa

Cette fonction de L-LDOPE Toolbelt est utilisée afin d'extraire des données codifiées dans la bande d'évaluation de qualité de Landsat 8 (*Quality Assessment*). En principe, elle permet de différencier les cirrus des autres nuages ainsi que l'ombre des nuages. Lors du test, l'ombre n'a été détectée dans aucun cas. Il est possible d'extraire les données souhaitées sur un seul raster (combinaison) ou séparément. Le temps d'exécution a été très rapide (3 secondes).

2.2.3 Fmask

Cette fonction différencie les nuages et les ombres de nuages (en plus des pixels clairs, de l'eau et de la neige). Il est possible de définir un seuil de détection et la dilatation des pixels de nuages ou de l'ombrage. Le temps de calcul a été relativement élevé et très variable (selon la scène) (médiane 159 sec. ; min. 107 sec. ; max. 1091 sec. ; sd 267 sec.) [2]. Des erreurs se sont produites relativement souvent avec des scènes téléchargées sur Amazon (5/12) et parfois avec des scènes téléchargées sur Google (2/12, puis en re-téléchargeant les scènes problématiques 1/12). Ces erreurs ont été dues à un problème lié aux bandes TIRS (voir section 3.2).

2.2.4 Comparaison

En terme de temps moyen, télécharger les images sur Google puis utiliser la fonction *unpack_oli_qa* a représenté la solution la plus rapide (158 secondes en moyenne).

En terme de précision, les résultats ont semblé dépendre du paramétrage et de la scène. Le paramétrage par défaut de Fmask (*dilated pixels* = 3, *threshold probability* = 22.5) a donné des couvertures nuageuses (sans prendre en compte l'ombrage) beaucoup plus importantes que celles obtenues par la fonction *unpack_oli_qa* par défaut (+ 35% en moyenne) avec des variations très importantes aussi (déviation standard = 43%). En diminuant la dilatation des pixels à 2 et en augmentant le seuil de probabilité à 50%, les résultats de Fmask se sont rapprochés de ceux de la fonction *unpack_oli_qa* (- 0.1% en moyenne), mais la déviation standard est restée élevée (35%).

La Figure 1 représente un exemple où les différences entre les output des deux fonctions ont été relativement importantes même lorsque l'on a diminué la dilatation des pixels à 2 et augmenté le seuil de probabilité à 50% de la fonction Fmask (*unpack_oli_qa* 10.6% vs 22.4% Fmask).

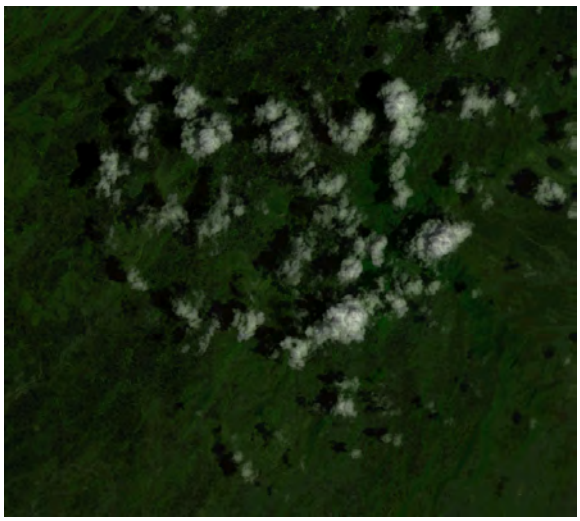
La Figure 2 représente un exemple où les différences entre les output des deux fonctions ont été moyennes lorsque l'on a diminué la dilatation des pixels à 2 et augmenté le seuil de probabilité à 50% de la fonction Fmask (*unpack_oli_qa* 26.6% vs 38.9% Fmask).

Finalement, la Figure 3 représente un exemple où les différences entre les output des deux fonctions ont été faibles lorsque l'on a diminué la dilatation des pixels à 2 et augmenté le seuil de probabilité à 50% de la fonction Fmask (*unpack_oli_qa* 61.4% vs 63.4% Fmask). Cependant cette faible différence en terme de pourcentage ne signifie pas autant que les output aient été similaires. En effet, il est à noter que la fonction Fmask paramétrée de la

2. Les temps d'exécution sont relatifs et dépendent des performances de l'ordinateur

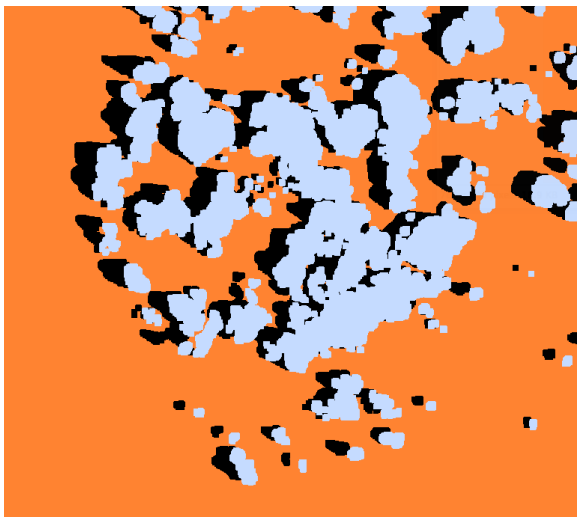
manière décrite ci-dessus a sous-estimé la présence nuageuse par rapport à ce qui est observé sur la composite RGB (Figure 3d).

Lorsque la fonction Fmask a été paramétrée avec 0 de dilatation et un seuil de 50% en utilisant la scène LC81960282015249LGN00 (geneva), le résultat s'est rapproché de celui obtenu avec la fonction de *unpack_oli_qa* (moins grossier) tout en tenant compte de l'ombre des nuages (Figure 4).

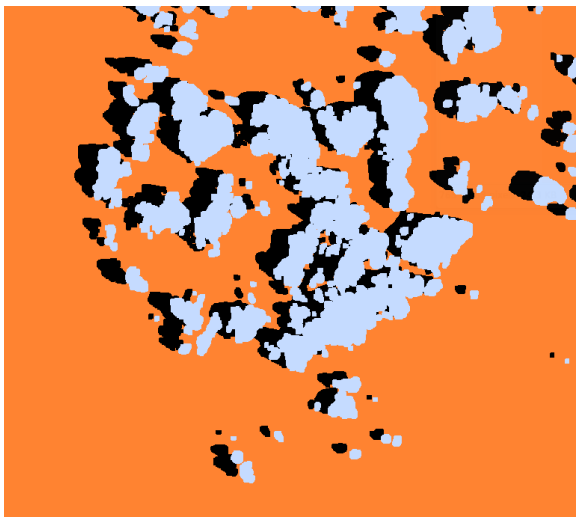


(a) RGB (4,5,2)

(b) *unpack_oli_qa* (medium)

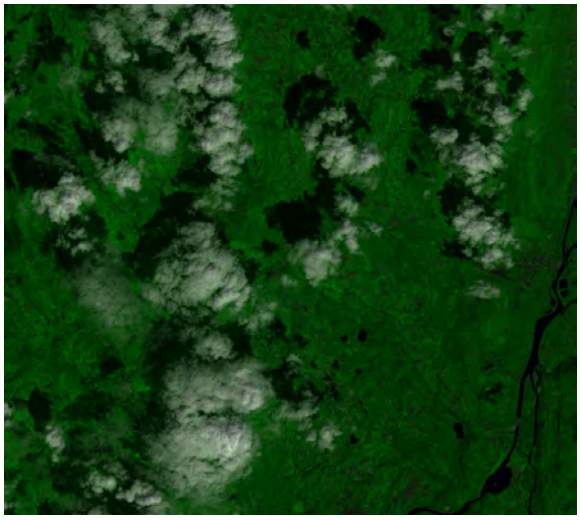


(c) Fmask (seuil = 22.5%, dilatation = 3)

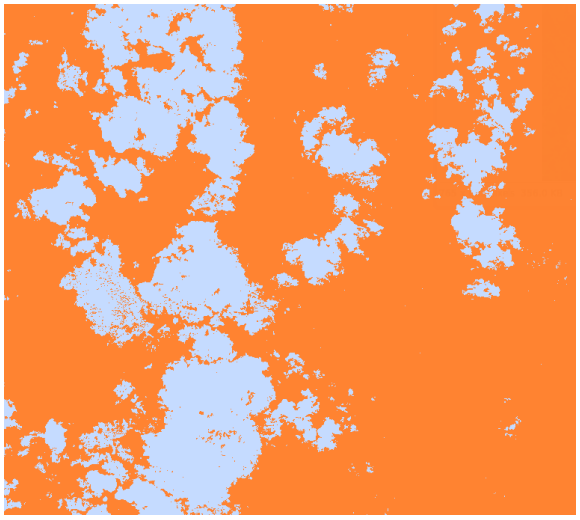


(d) Fmask (seuil = 50%, dilatation = 2)

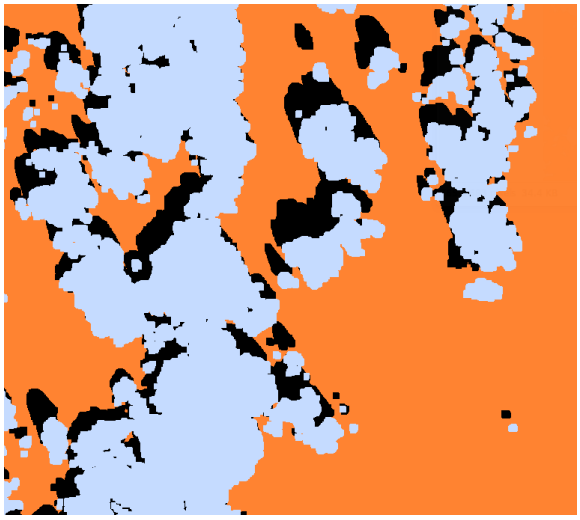
FIGURE 1 – Détection de nuages. Comparaison des différents output. Scène LC81680542016072LGN00, Addis Ababa, 12.03.2016.



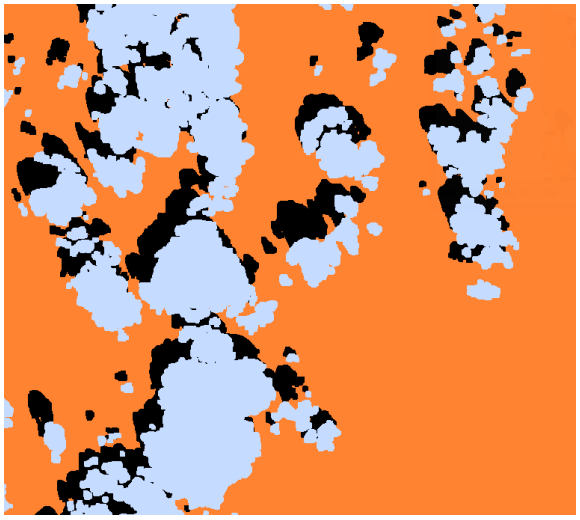
(a) RGB (4,5,2)



(b) *unpack_oli_qa* (medium)

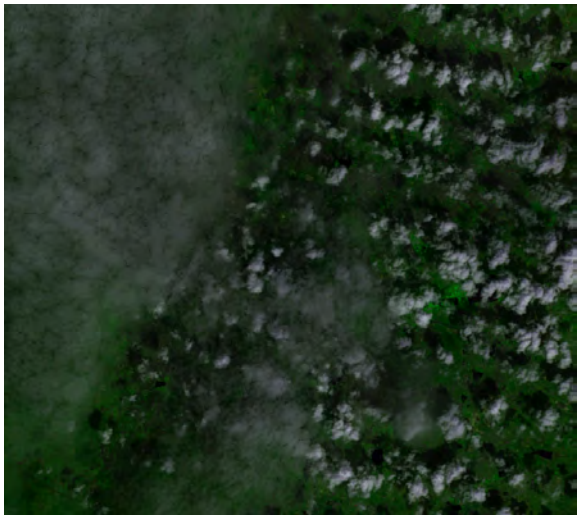


(c) Fmask (seuil = 22.5%, dilatation = 3)

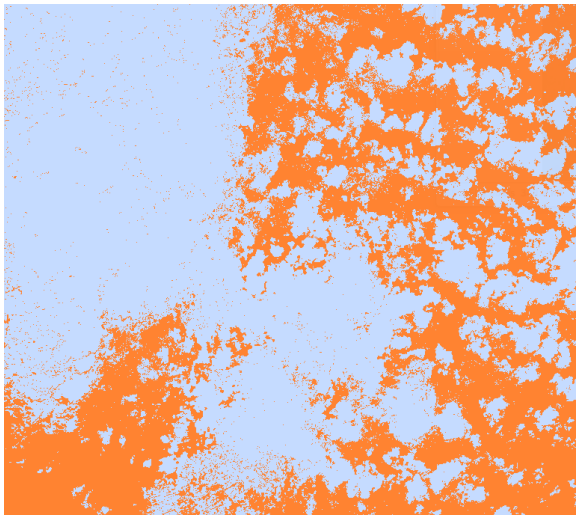


(d) Fmask (seuil = 50%, dilatation = 2)

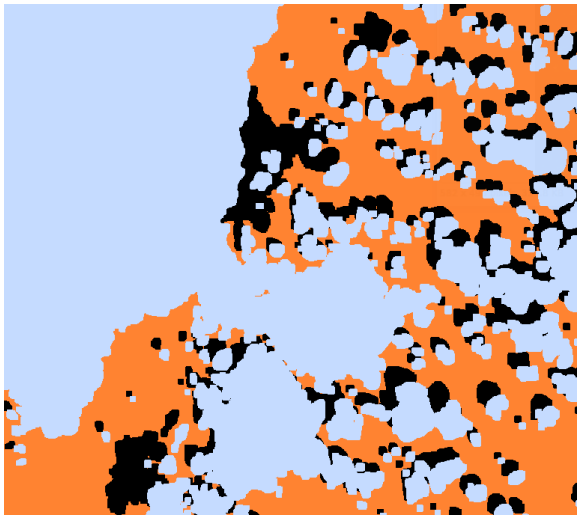
FIGURE 2 – Détection de nuages. Comparaison des différents output. Scène LC81960282015249LGN00, Geneva, 06.09.2015.



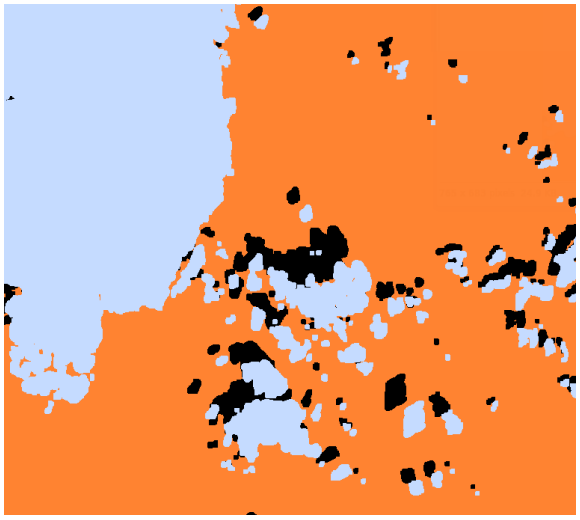
(a) RGB (4,5,2)



(b) *unpack_oli_qa* (medium)

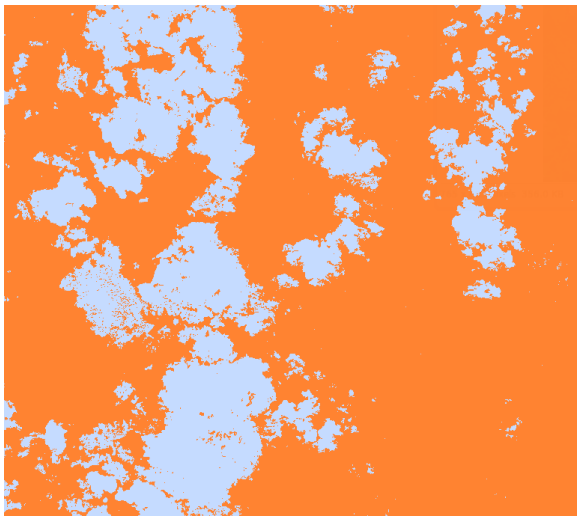


(c) Fmask (seuil = 22.5%, dilatation = 3)

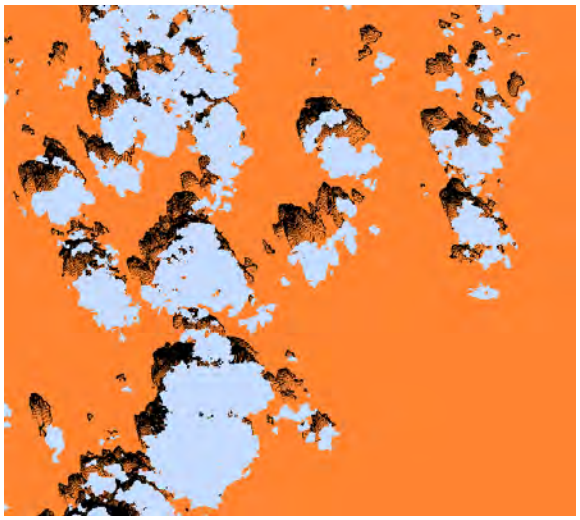


(d) Fmask (seuil = 50%, dilatation = 2)

FIGURE 3 – Détection de nuages. Comparaison des différents output. Scène LC81440512014346LGN00, Bangalore, 12.12.2014.



(a) *unpack_oli_qa* (medium)



(b) Fmask (seuil = 50%, dilatation = 0)

FIGURE 4 – Détection de nuages. Masques similaires. Scène LC81960282015249LGN00, Geneva, 06.09.2015.

2.3 Conclusion

Bien que la fonction *unpack_oli_qa* soit la solution la plus rapide, elle comporte l'inconvénient de ne pas détecter l'ombre des nuages, lesquelles représentent en moyenne entre 15 et 20% de la couverture totale détectée par la fonction Fmask, selon le paramétrage effectué. Aussi, la dilatation des pixels réalisée par la fonction Fmask assure un masque plus conservateur. Pour ces raisons, il a été décidé de poursuivre les analyses avec cette dernière plutôt qu'avec la fonction *unpack_oli_qa*. Ceci étant dit, une version C de Fmask sera prochainement utilisée comme nouvel algorithme pour l'identification des nuages de la bande QA [3]

3 Fmask

3.1 Conservation des fichiers de réflectances

Dans le cas où l'on a accès au code original Matlab, il peut être intéressant de conserver les rasters de réflectances calculées par la fonction Fmask et supprimées par défaut pour de possibles analyses ultérieures. Les bandes 1 (aérosols côtiers) et 8 (panchromatique) ne sont pas prises en compte dans la fonction Fmask et ne peuvent donc pas récupérées, à moins d'ajouter des lignes de code associées au calcul de réflectances pour ces bandes en vérifiant que cela n'ait pas d'incidence sur le reste de la fonction. Dans le cas où il est décidé de ne pas utiliser les bandes 1 et 8, il est possible d'exécuter la fonction Fmask sans télécharger ces dernières (uniquement possible via le *cloud* Amazon) et ainsi gagner du temps.

3. New Collection 1 products. `http://landsat.usgs.gov//landsatcollections.php`, consulté le 28 juin 2016

La fonction Fmask est originalement composée de 8 scripts Matlab. Le calcul des réflectances est codé dans le script *nd2toarbt.m* mais l'exportation des fichiers ENVI se codifie à la fin du script *plcloud.m* où toutes les variables nécessaires à l'exportation (dimension, résolution, etc.) sont définies (une partie de celles-ci sont extraites du fichier de métadonnées L*MTL.txt grâce au script *lndhdrread.m*). Il est à noter que la numérotation des différentes bandes exportées est décalée par rapport à la numérotation originale (1=B2 ; 2=B3 ; 3=B4 ; 4=B5 ; 5=B6 ; 6=B7 ; 7=B9).

Le fait que les images originales proviennent d'Amazon (compression LZW) ou de Google (non-compressées) n'affecte pas la taille du fichier ENVI de réflectances (composé de plusieurs bandes) exporté depuis Fmask (environ 800 MB).

Les réflectances calculées dans Fmask sont égales à celles calculées dans GRASS, chaque valeur étant simplement multipliée par 1'000 pour des raisons de codage. Des nuages de point bivarié mettant en relation les réflectances calculées dans GRASS et avec Fmask ont permis de confirmer cela (Figures 5 et 6). Les lignes de point horizontales (y=-9999) tient du fait que la fonction Fmask découpe légèrement l'image et que la valeur de -9'999 est assignée aux pixels situés en dehors de la zone de traitement. Le script R ayant permis de comparer les différents rasters de réflectances se trouve dans l'Annexe D.1.

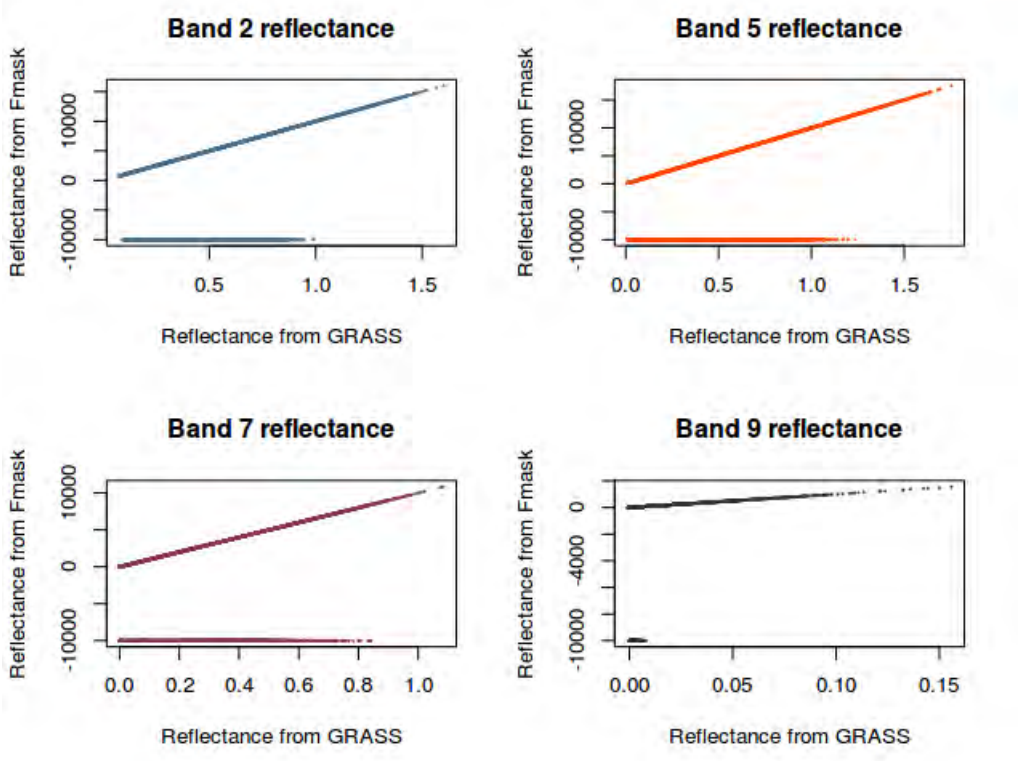


FIGURE 5 – Comparaison entre les réflectances calculées dans GRASS et avec la fonction Fmask. Scène utilisée : LC82000342015021LGN00 (sample = 500'000 pixels ; 0.8% des pixels)

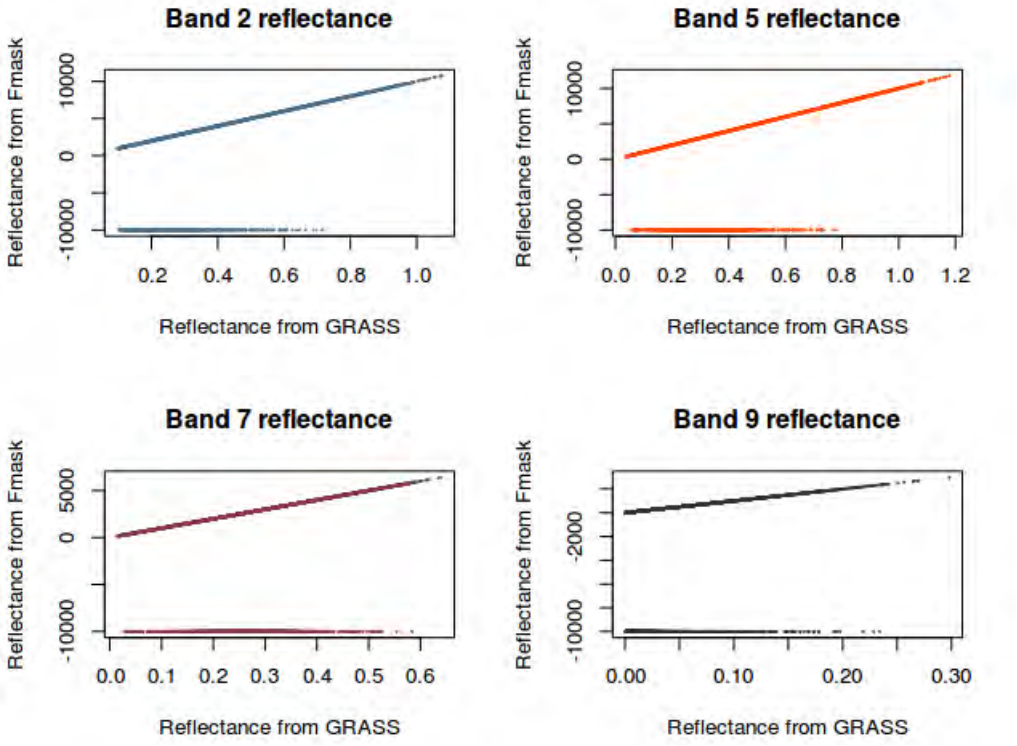


FIGURE 6 – Comparaison entre les réflectances calculées dans GRASS et avec la fonction Fmask. Scène utilisée : LC81590292015134LGN00 (sample = 500'000 pixels ; 0.8% des pixels)

3.2 Problèmes liés aux bandes TIRS

En utilisant la fonction Fmask sur certaines scènes (le plus souvent téléchargées sur Amazon), une erreur peut se produire avec le message suivant :

```
Error using imfill
Expected input number 1, I1 or BW1, to be non-NaN.
```

Ceci semble avoir lieu lorsque les bandes thermales (10 et 11) sont vides. Une anomalie au niveau des Thermal Infrared Sensor (TIRS) découverte en décembre 2014 est la cause de ce problème et depuis un traitement supplémentaire des données enregistrées est nécessaire. Lorsque les bandes TIRS téléchargées n'ont pas été recalculées et qu'elles sont vides, une fonction alternative initialement implémentée pour les images Sentinel 2 peut être utilisée. Afin d'estimer l'importance de ces bandes thermales dans le calcul de masques, l'une et l'autre fonction on été exécutées sur 6 scènes ayant toutes des données TIRS valides. Le script associé à ce test figure dans l'Annexe D.2.

Sur les 6 scènes, la fonction Fmask qui ne tient pas compte des bandes TIRS (FmaskS) a sous-estimé le masquage total (nuage + ombres) par rapport à la fonction qui en tient compte (FmaskT) dans 2/3 des cas. Dans le 1/3 restant, elle l'a très légèrement surestimé. La différence maximum a été observée sur la scène LC82000342015021LGN00 (Almeria), avec une différence de 21.24 points de pourcentage (FmaskT 56.59% vs 35.35% FmaskS) par

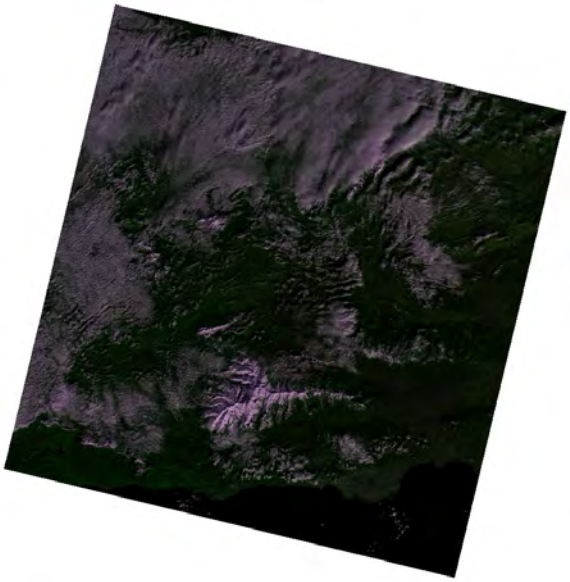
rapport au masquage total (nuage + ombre). Cependant, dans 75% des cas, la différence n'a pas dépassé 10 points.

La Figure 7 reflète la scène LC82000342015021LGN00 (Almeria), là où la différence a été maximum. Le masquage de la neige située au bas de la scène a été plus important lorsque l'on a tenu compte des bandes TIRS, bien que celle-ci ait été considérée comme nuage dans cet exemple (Fmask différencie en principe la neige). Pour le reste, il semblerait que FmaskT ait permis d'obtenir de meilleures résultats (et surtout plus conservateurs).

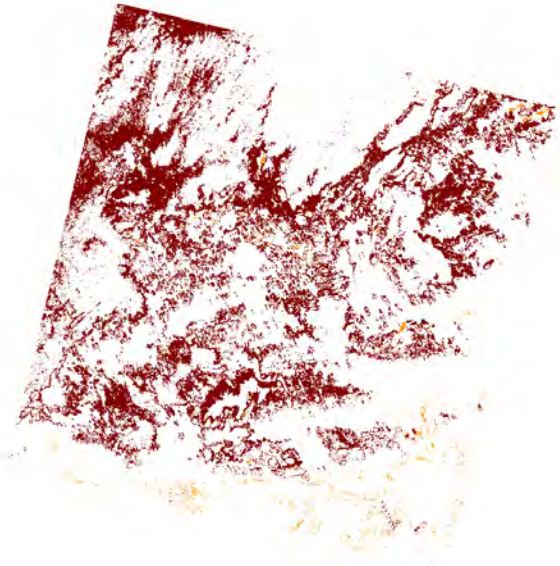
En revanche, la scène LC81960282015249LGN00 (Geneva) a présenté des résultats très similaires avec les deux fonctions en terme de masquage (FmaskT 49.4% vs 46.25% FmaskS) (Figure 8). On observe néanmoins que le masquage supplémentaire de FmaskT réparti de manière assez homogène sur l'image est compensé par une zone située en bas à gauche de l'image, masquée par FmaskS et non pas par FmaskT. Il semblerait, selon l'impression visuelle obtenue avec la composite, que FmaskT a masqué cette zone avec plus de précision. Néanmoins, on observe à nouveau de la neige classifiée comme nuage (en bas à droite de l'image). Le masque obtenu avec FmaskT y est plus important et l'ombre calculée en conséquence aussi.

La différence obtenue avec la scène LC81680612014098LGN00 (Nairobi) a été de 7.03 points de pourcentage. Les variations observées en bas à gauche de l'image (Figure 9) sont difficiles à évaluer. FmaskS a projeté des ombres de manière plus importante que FmaskT. En revanche le masquage réalisé par FmaskT semble clairement plus juste en ce qui concerne la zone en bas au milieu. Des cirrus sont effet présents et ont été correctement masqués lorsque les bandes TIRS ont été prises en compte (Figure 10).

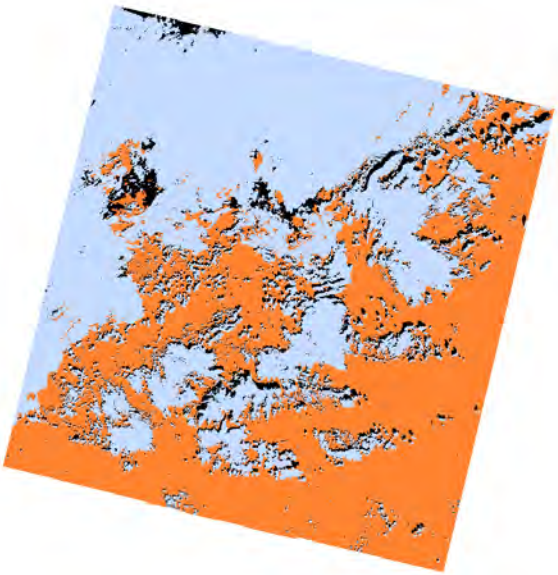
En se basant sur les quelques exemples mentionnés ci-dessus, il semblerait que la fonction Fmask qui tient compte des bandes TIRS est généralement plus précise. Cependant, les différences ne sont pas excessives et la fonction FmaskS donne des résultats plutôt satisfaisants. Elle pourrait donc être utilisée avec une certaine prudence lorsque les bandes TIRS ne sont pas disponibles.



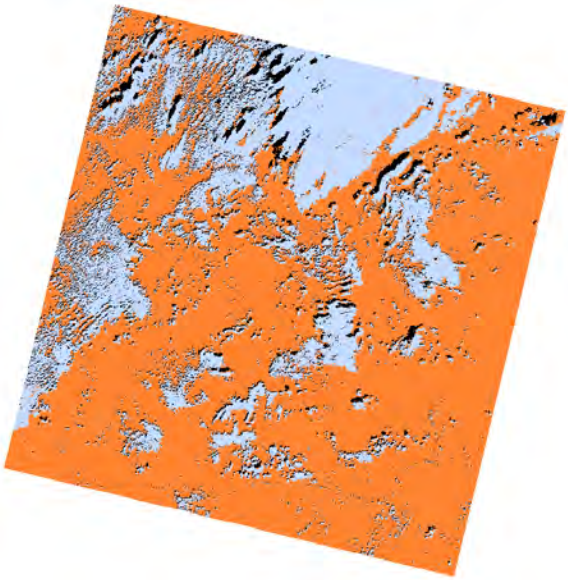
(a) RGB (4,5,2)



(b) Cross-mapping

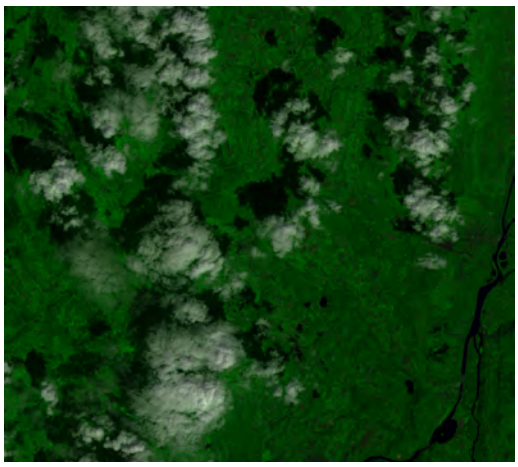


(c) FmaskT

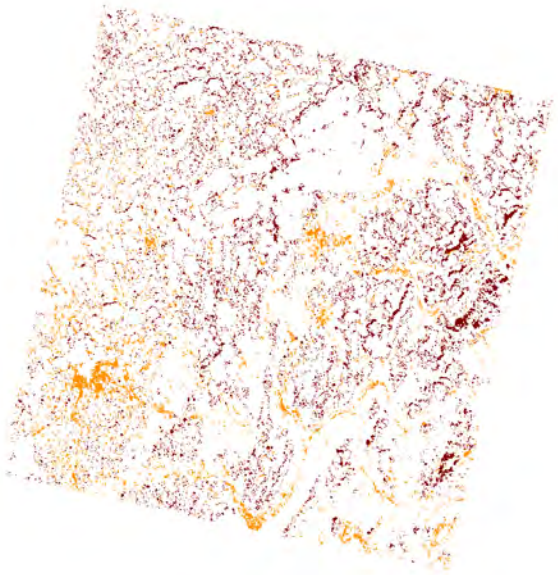


(d) FmaskS

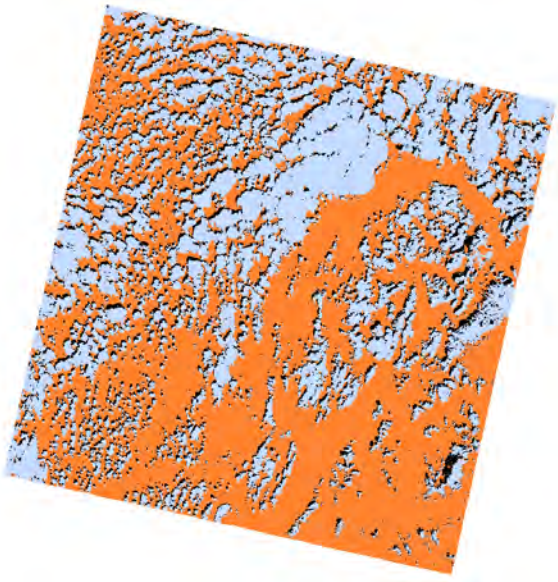
FIGURE 7 – Fmask avec et sans bandes TIRS. Scène LC82000342015021LGN00, Almeria, 21.01.2015. b) en rouge les les pixels masqués avec FmaskT et non-masqués avec FmaskS, en orange, l'inverse



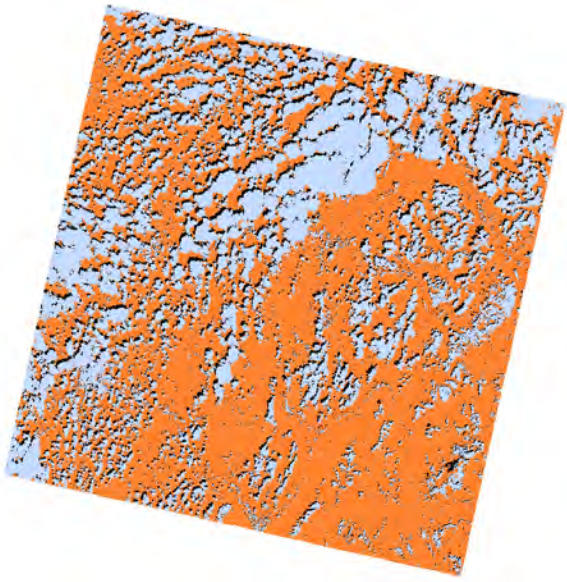
(a) RGB (4,5,2)



(b) Cross-mapping

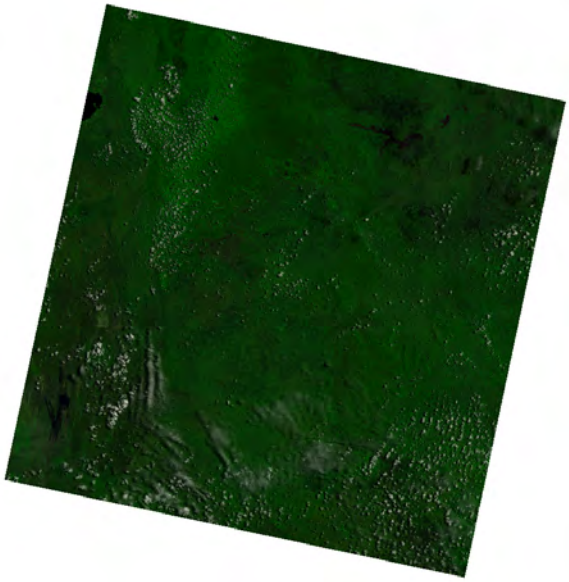


(c) FmaskT

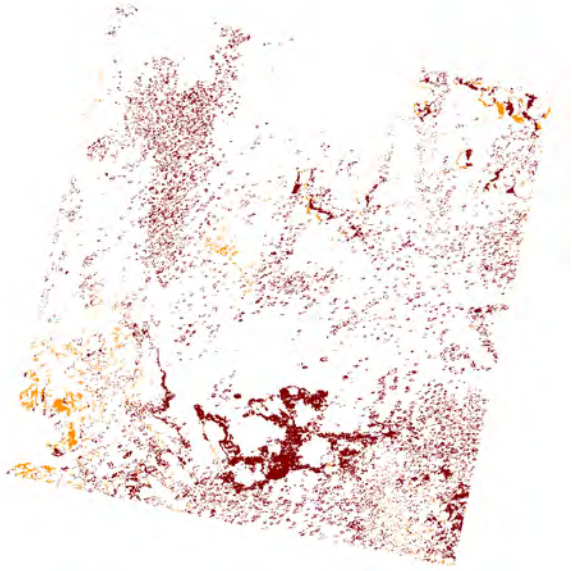


(d) FmaskS

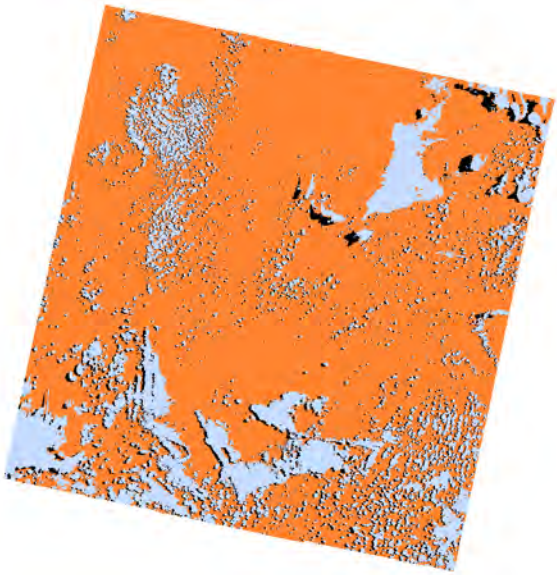
FIGURE 8 – Fmask avec et sans bandes TIRS. Scène LC81960282015249LGN00, Geneva, 06.09.2015. b) en rouge les les pixels masqués avec FmaskT et non-masqués avec FmaskS, en orange, l'inverse



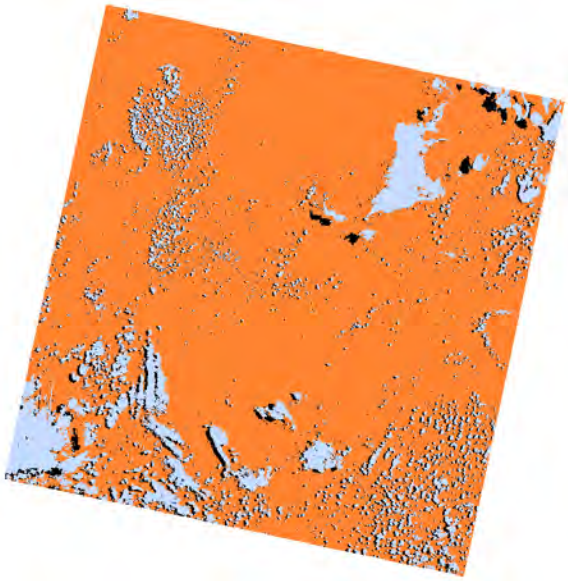
(a) RGB (4,5,2)



(b) Cross-mapping

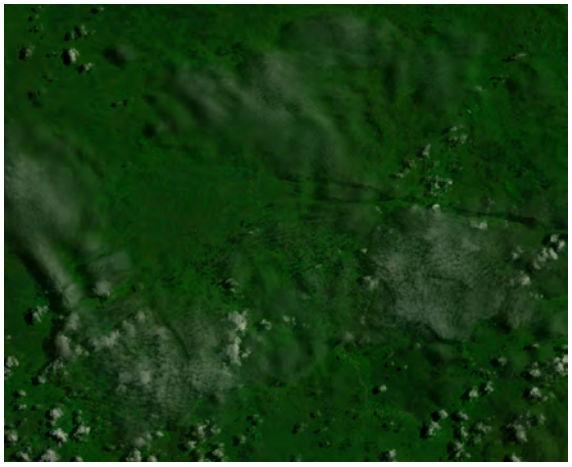


(c) FmaskT

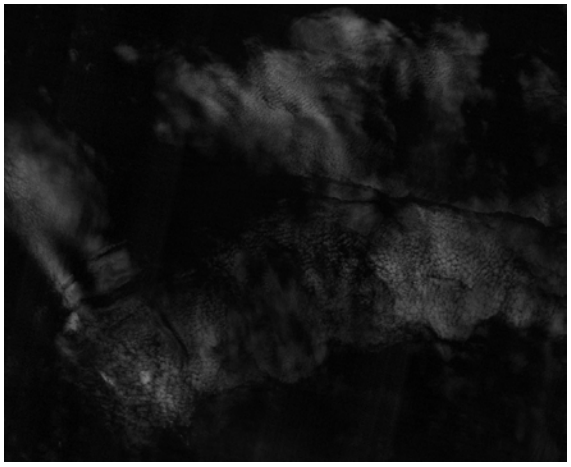


(d) FmaskS

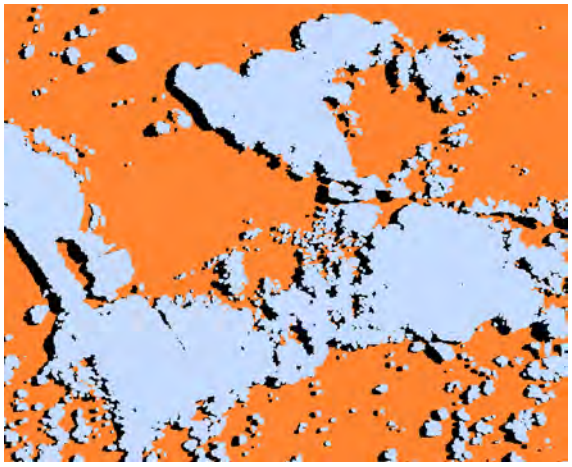
FIGURE 9 – Fmask avec et sans bandes TIRS. Scène LC81680612014098LGN00, Nairobi, 08.04.2014. b) en rouge les les pixels masqués avec FmaskT et non-masqués avec FmaskS, en orange, l'inverse



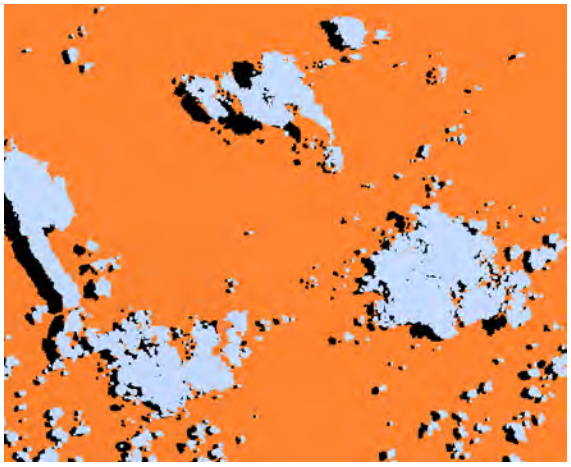
(a) RGB (4,5,2)



(b) Cross-mapping



(c) FmaskT



(d) FmaskS

FIGURE 10 – Fmask avec et sans bandes TIRS. Scène LC81680612014098LGN00 (zoom), Nairobi, 08.04.2014. b) en rouge les les pixels masqués avec FmaskT et non-masqués avec FmaskS, en orange, l'inverse

3.3 Paramétrage

Quatre variables peuvent être paramétrées lors de l'exécution de la fonction Fmask.

1. "cldpix" is dilated number of pixels for cloud with default values of 3.
2. "sdpix" is dilated number of pixels for cloud shadow with default values of 3.
3. "snpix" is dilated number of pixels for snow with default values of 0.
4. "cldprob" is the cloud probability threshold with default values of 22.5 (range from 0 100).

Les développeurs de la fonction Fmask ont testé plusieurs valeurs de seuil de probabilité de nuage sur 142 images et ils ont conclu que la valeur de 22.5% offre les meilleures résultats (Zhu *et al.* 2015) (Figure 11).

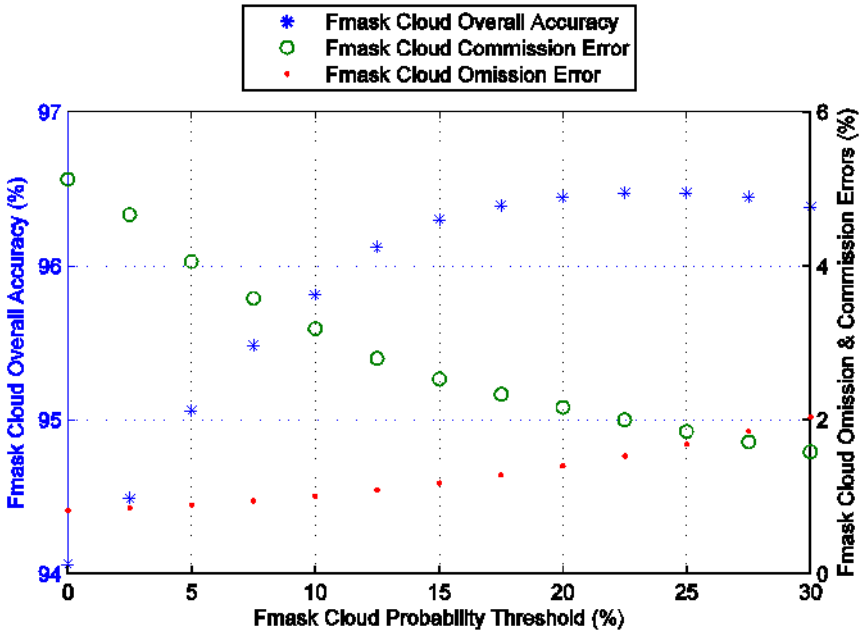


FIGURE 11 – Détection de nuages en fonction de différents seuils de probabilité, basé sur un total de 142 images. Source : `https://github.com/prs021/fmask`, consulté le 28 juin 2016.

Afin d'observer l'impact du paramétrage sur les résultats issus de la fonction Fmask, des tests ont été effectués en utilisant 3 valeurs de seuil de probabilité de nuage (15%, 22.5% et 30%, la valeur par défaut étant 22.5%) ainsi que 3 valeurs pour la dilatation des pixels pour les nuages et les ombres (1, 3, 5, la valeur par défaut étant 3). Les valeurs par défaut de seuil de probabilité de nuage ont été utilisées pour tester les différentes valeurs de dilatation de pixels et les valeurs par défaut de dilatation ont été utilisées pour tester les différentes valeurs de probabilité de nuage. Ceci représente donc 5 paramétrages différents, lesquels ont été testés sur 6 scènes.

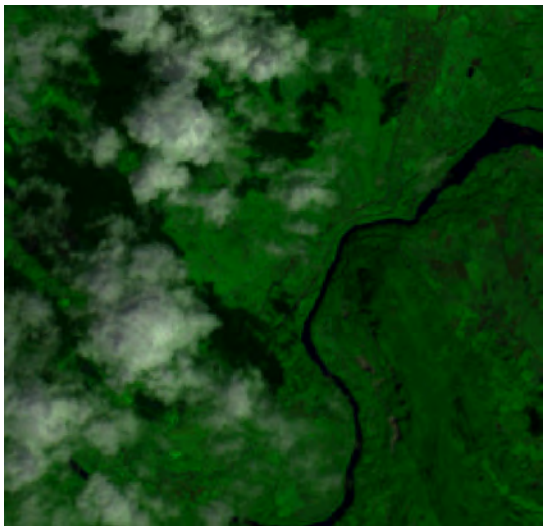
Un script à exécuter dans GRASS (voir Annexe D.3) a été créé dans le but d'automatiser ces tests en définissant trois variables :

1. Un dossier dans lequel se trouvent les images bruts Landsat 8 téléchargées
2. Une liste contenant les différentes valeurs de dilatation de pixel à tester
3. Une liste contenant les différentes valeurs de seuil de probabilité de nuage

Les grandes lignes du script consistent en les points suivants : Après avoir exécuté la fonction Fmask en utilisant les différents paramètres, les output de la fonction Fmask sont traités afin d'obtenir des masques simples (0=clear pixel, 1=nuage ou ombre). Des cross mappings sont effectués sur **i)** les résultats obtenus des tests de seuil de probabilité et **ii)** les résultats obtenus des tests de dilatation de pixels. Finalement, des fonctions R sont utilisées afin de calculer le pourcentage de masque pour chaque raster et pour créer un tableau récapitulatif de ces données. Ce dernier, tout comme les masques, les cross mappings et les rapports des cross mappings sont exportés dans un nouveau dossier.

Le script a été exécuté sur les différentes scènes et une analyse des output a pu être effectuée afin de mesurer l'influence des différents paramètres sur les résultats et de déterminer les valeurs les plus appropriées.

Le cross-mapping effectué avec les différents résultats obtenus en modifiant la valeur de dilatation des pixels des nuages et de l'ombre projetée de ceux-ci a permis de visualiser l'effet de ce paramètre. Sur tous les tests, la différence de masquage maximum a été obtenue entre les valeurs de dilatation 1 et 5 (scène LC81000652015073LGN00, Papua) et n'a pas dépassé 10 points de pourcentage (76.39% vs 86.29%). La Figure 12 illustre l'effet du paramètre de dilatation de pixels dans la fonction Fmask.



(a) RGB (4,5,2)

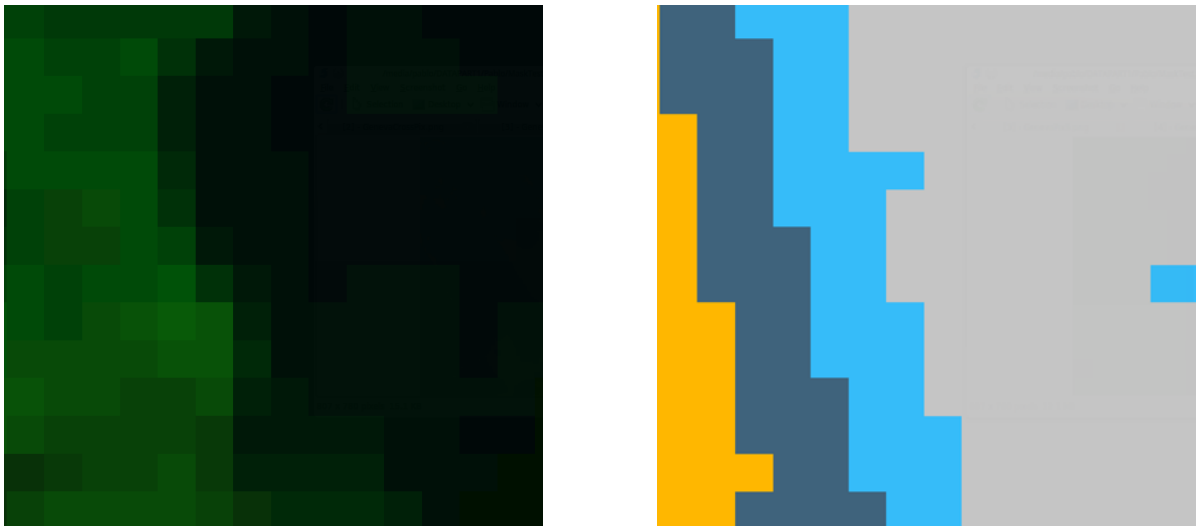
(b) Cross-mapping en fonction du paramétrage (1,3,5)

(c) Dilatation minimum (1)

(d) Dilatation maximum (5)

FIGURE 12 – Différents paramétrage de dilatation de pixels. Scène LC81960282015249LGN00, Geneva, 06.09.2015.

L'évaluation visuelle n'a pu se faire qu'en effectuant un zoom de manière importante afin de pouvoir différencier chaque pixel. En faisant cela, la tâche est restée difficile, mais il a été possible d'observer que la valeur 1 tend à sous-estimer le masque et la valeur 5 tend à le surestimer (Figure 13).



(a) RGB (4,5,2)

(b) Cross-mapping en fonction du paramétrage (1,3,5)

FIGURE 13 – Détail de l'effet du paramétrage de dilatation de pixels. Scène LC81960282015249LGN00 (zoom), Geneva, 06.09.2015.

Compte tenu de ce qui précède et de la difficulté à évaluer de manière précise le paramétrage de la dilatation des pixels, il semble judicieux d'utiliser la valeur par défaut proposée par les développeur de la fonction Fmask, à savoir 3.

En ce qui concerne le paramétrage du seuil de probabilité de nuage, les différences maximums ont été obtenues entre les valeurs de seuil de 15% et 30%. Néanmoins, elles n'ont pas dépassé les 9 points de pourcentage (moyenne = 5.7 points, déviation standard = 2.15). Bien que la surface totale masquée ait été inversement proportionnelle à la valeur du seuil, certaines régions ont été masquées avec une valeur seuil de 30% et non masquées avec une valeur seuil de 22.5% ou de 15% (zone bleue au centre de la Figure 14).

Alors que le cross-mapping associé au test des valeurs de dilatation de pixels n'a donné logiquement que 4 catégories (0,0,0 ; 0,0,1 ; 0,1,1 ; 1,1,1), celui associé au seuil de probabilité de nuages a produit chacune des 8 catégories possibles. Selon les développeurs, la précision globale augmente jusqu'à la valeur de 22.5% (définie par défaut) avant de légèrement diminuer. Pour les mêmes raisons que pour le paramétrage des valeurs de dilatations des pixels, il semble raisonnable de suivre la recommandation des développeurs et utiliser la valeur par défaut.

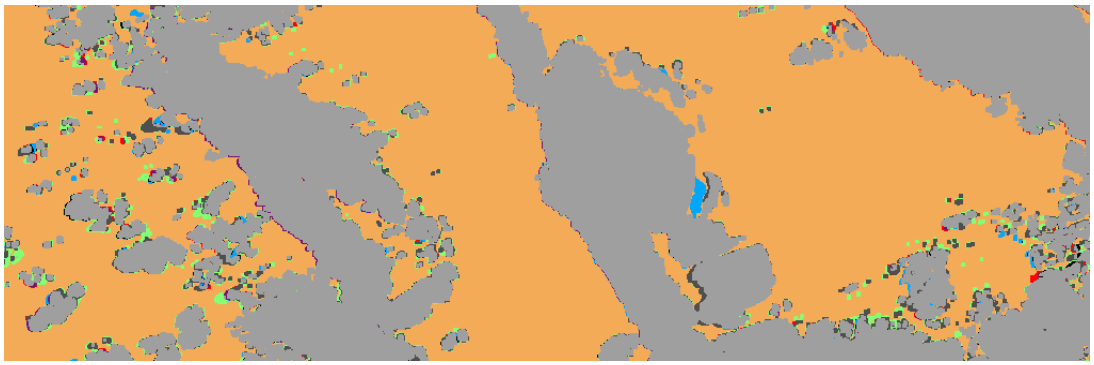


FIGURE 14 – Cross-mapping entre les différents output en fonction de la valeur seuil de probabilité de nuage. En vert clair, les pixels masqué uniquement lorsque la valeur est moindre (15%), en rouge, les pixels masqués uniquement avec la valeur seuil intermédiaire (22.5%) et en bleu clair, les pixels masqués uniquement lorsque la valeur seuil est plus importante (30%).

3.4 Code MATLAB et compilation

3.4.1 Logiciels

La fonction Fmask est codée dans MATLAB. L'accès aux codes requiert le logiciel MATLAB, lequel fonctionne sous licence, néanmoins la fonction peut être exécutée sans le logiciel, si l'on dispose d'une compilation du code sous forme de fichier exécutable. Pour cela, il est tout de même nécessaire d'installer le logiciel MATLAB Runtime. La version de MATLAB Runtime doit correspondre à la version de MATLAB utilisée pour la compilation. MATLAB Runtime ne requiert pas de licence.

3.4.2 Compilation

Disposant du logiciel MATLAB, nous avons eu accès au code original de la fonction Fmask et avons pu activer une ligne dans le script *pcloud.m*, déjà présente mais non-active, permettant de récupérer les réflectances calculées pour la fonction (Figure 15).

```
226     % refine Water mask - Zhe's water mask (no confusion water/cloud)
227 -   WT(WT==1&Cloud==0)=1;
228     % bwmorph changed Cloud to Binary
229 -   Cloud=uint8(Cloud);
230 -   Cloud(mask==0)=255;
231 -   Shadow(mask==0)=255;
232     % enviwrite('TOA3.2',data,'int16',resolu,ul,'bsq',zc);
233     % enviwrite('BT3.2',Temp,'int16',resolu,ul,'bsq',zc);
234     % enviwrite('final_prob',final_prob,'single',resolu,ul,'bsq',zc);
235     % enviwrite('thin_prob',100*Thin_prob,'single',resolu,ul,'bsq',zc);
236 -   end
```

FIGURE 15 – Ligne de code permettant d'exporter un fichier ENVI contenant les réflectances des bandes 2,3,4,5,6,7,9,10 et 11

Une fois cette opération réalisée, le programme a été compilé grâce à l'application *Compiler* (Figure 16).

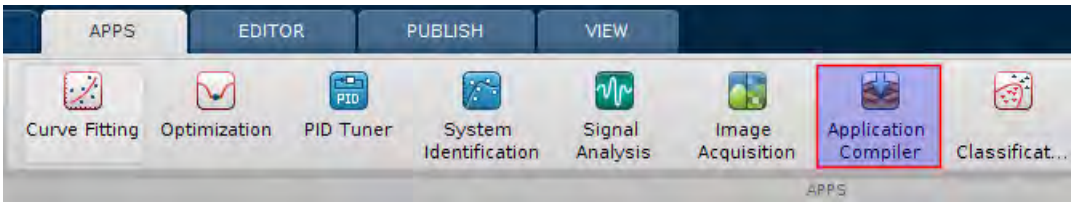


FIGURE 16 – Exécution de MATLAB Compiler

Le script principal du programme a dû être sélectionné (*autoFmask.m*) ainsi que les autres scripts requis pour l'exécution du programme (Figure 17).

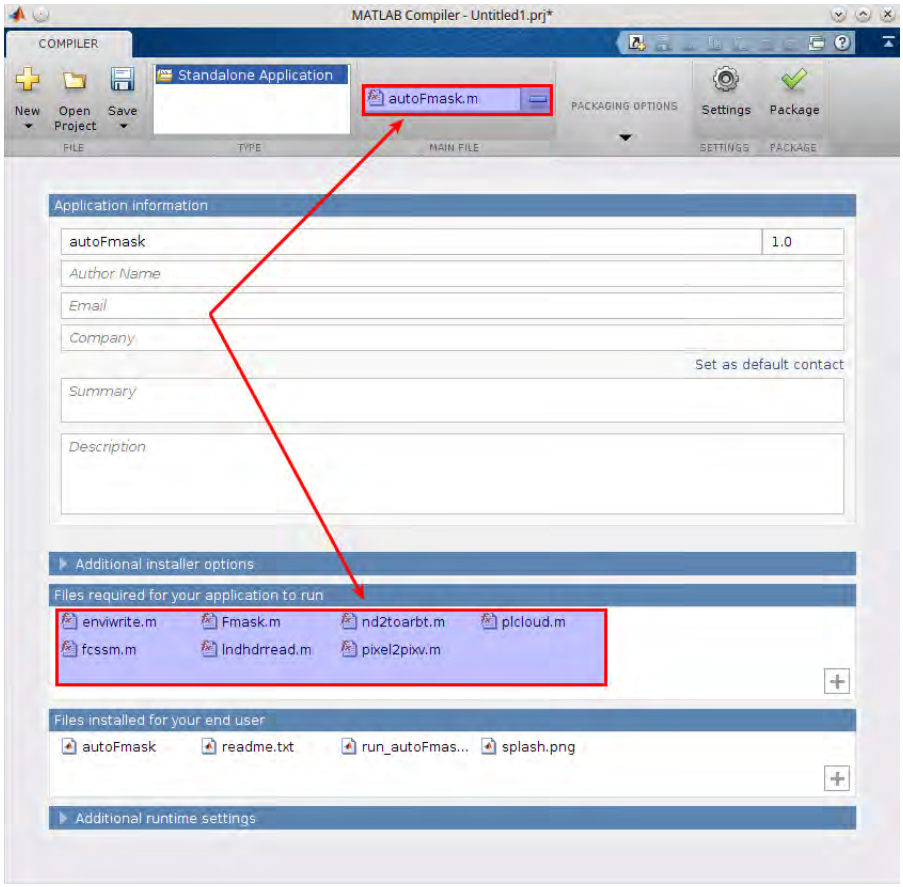


FIGURE 17 – Compilation de la fonction Fmask

4 Intégration de la fonction Fmask

Le script *LiMES_grasser* (voir Annexe D.4.1) associé au script *grass_tools* (voir Annexe D.4.2) a été initialement utilisé afin de découper les scènes téléchargées selon un *shapefile* prédéfini puis d'importer les résultats dans une base de données GRASS, avant de réaliser certains traitements éventuels (ex. calcul de réflectances). L'intégration de la fonction Fmask dans ce script permet d'importer directement des images de réflectances (calculées par la

fonction Fmask) et de masquer les zones couvertes par des nuages et leurs ombres (Figure 18).

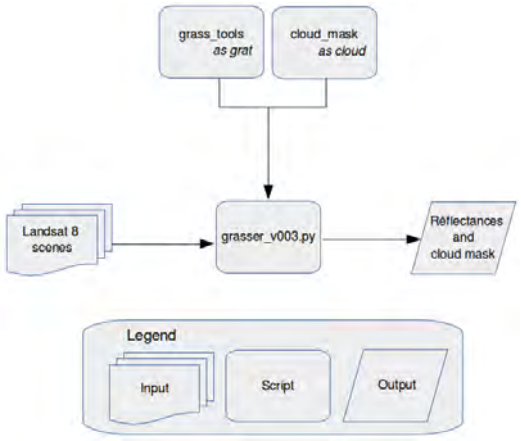


FIGURE 18 – Ordinagramme structurel du processus d'intégration de la fonction Fmask.

4.1 Scripts

4.1.1 cloud_mask.py

Lorsque les bandes TIRS ne contiennent pas d'information, la fonction Fmask originale ne peut être exécutée. Néanmoins, une fonction alternative, initialement implémentée pour des images *Sentinel 2* peut être utilisée. Afin de déterminer quelle fonction employer, une recherche des valeurs de radiance maximum pour les bandes 10 et 11 dans le fichier de métadonnées *MTL.txt* peut être réalisée. Cette méthode comporte l'avantage d'être très rapide. Un script Python parallèle décrivant la fonction *cloudmask* permet de tester cela et d'exécuter le fichier Fmask correspondant (voir Annexe D.4.3).

4.1.2 LiMES_grasser.py

Les modifications principales suivantes ont été effectuées dans le script *LiMES_grasser* afin d'intégrer la fonction Fmask :

Ligne 43, le script *cloud_mask.py* est importé afin de pouvoir appeler la fonction cloudmask définie dans celui-ci.

```
import cloud_mask as cloud
```

Ligne 49 et 50, deux arguments supplémentaires peuvent être ajoutés pour l'exécution du script. Il permettent d'une part d'activer la conversion des réflectances de type Int16 (valeurs allant de 0 à 10'000) issues de la fonction Fmask en type Float64 (valeurs allant de 0 à 1), et d'autre part, l'unification des différents masques correspondant à chaque scène de manière à ne prendre en compte seulement les pixels qui sont clairs sur toutes les scènes.

```
parser.add_argument('-c', action='store_true', default=False, help='Convert int16
    output to float64 (default: False)')
parser.add_argument('-m', action='store_true', default=False, help='Merge all
    different scene masks in order to work only with pixels that are clear in all
    the scenes (default: False)')
```

Ligne 90, avant de commencer une boucle sur les différentes scènes téléchargées, il a été ajouté le code suivant :

```
 for scene in row[2].split(','):
              mydir = './scenes/%s/' % scene
              for file in os.listdir(mydir):
                  if file.endswith("_MTL.txt"):
                      MTLfile = file
              try:
                  MTLfile
              except NameError:
                  sys.exit('MTL.txt file from %s scene is missing. Script aborted'
                       % scene)
              else:
                  pass
```

Ceci permet de vérifier que les fichiers de méta-données *MTL.txt* requis pour la fonction Fmask sont bien présents pour toutes les scènes avant que les calculs ne commencent.

Ligne 112, au tout début de la boucle effectuée sur les différentes scènes, la fonction *cloudmask* est appelée.

```
print '** Starting Fmask function on %s scene' % scene
cloud.cloudmask(scene)
```

Ligne 132, les output de la fonction Fmask (le masque et le fichier multi-bandes de réflectances) sont découpés selon le *shapefile* prédéfini.

```
# CLIP MASK
fmaskOutMask = '%s_MTLFmask' % scene
mask = './scenes/%s/%s' % (scene,fmaskOutMask)
command = 'gdal_translate,-projwin,%s,%s,%s,%s,-eco,%s,%s' % (str(float(LLLong)),
    str(float(URLat)), str(float(URLong)), str(float(LLLat)), mask, mask.replace
    ('/scenes/','/sites/%s/cropped/' % row[0]))
comlist = command.split(',')
subprocess.call(comlist, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
os.rename('./sites/%s/cropped/%s/%s' % (row[0],scene,fmaskOutMask), './sites/%s/
    cropped/%s/%s' % (row[0],scene,fmaskOutMask) + '.tif')

# CLIP REFLECTANCES
fmaskOutRef = 'TOA3.2'
reflect = './scenes/%s/%s' % (scene,fmaskOutRef)
```

```
command = 'gdal_translate,-projwin,%s,%s,%s,%s,-eco,%s,%s' % (str(float(LLLong)),
    str(float(URLat)), str(float(URLong)), str(float(LLLat)), reflect, reflect.
    replace('/scenes/','/sites/%s/cropped/' % row[0]))
comlist = command.split(',')
subprocess.call(comlist, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
```

Ligne 147, le fichier ENVI multi-bandes de réflectances est décomposé et chaque nouveau fichier correspondant à chaque bande est renommé correctement (la numération de 1 à 7 utilisée par Fmask ne correspond pas à la numération des différentes bandes spectrales). La conversion des rasters de type Int16 en rasters de type Float64 est activée par l'utilisateur par l'argument -c.

```
for band in range(1,8):
    if band == 7:
        outBand = 9
    else:
        outBand = band+1

    multiBand = './sites/%s/cropped/%s/%s' % (row[0],scene,fmaskOutRef)
    singleBand = './sites/%s/cropped/%s/%s_T%s.tif' % (row[0],scene,scene,outBand)

    if args.c == True:
        command = 'gdal_translate,-ot,Float64,-scale,0,10000,0,1,%s,%s,-b,%s' % (
            multiBand,singleBand,band)
    else:
        command = 'gdal_translate,%s,%s,-b,%s' % (multiBand,singleBand,band)

    comlist = command.split(',')
    subprocess.call(comlist, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
```

En fonction du nombre de scène et du choix de l'utilisateur (argument -m), la fonction mask définie dans le script *grass_tools* peut être appelée.

```
listScenes = row[2].split(',')
numScenes = (len(row[2].split(',')))
if (numScenes == 1) | (args.m == True):
    grat.mask(numScenes,listScenes)
else:
    pass
```

4.1.3 grass_tools.py

Dans le script *grass_tools* (voir Annexe D.4.2), la fonction *mask*, permet de construire un masque global selon le nombre de scènes et de l'appliquer sur les couches de la base de données dans GRASS. Si le nombre de scène est égale à 1, alors la fonction r.mask de GRASS est directement exécutée avec le masque original issu de la fonction Fmask, ceci en sélectionnant les valeurs de pixels correspondantes aux pixels clairs. Si ce n'est pas le cas, alors la fonction

r.mapcalc est exécutée afin de créer un raster construit sur la base des différents masques, lequel est ensuite utilisé dans la fonction r.mask.

```
def mask(numScenes,listScenes):
    print '** Creating MASK'
    g.region(raster='%s_T4' % listScenes[0])
    if numScenes == 1:
        FinalMask = '%s_MTLFmask' % listScenes[0]
        r.mask(raster=FinalMask,maskcats='0 1 3')
    else:
        if numScenes == 2:
            expr1 = '((%s_MTLFmask == 0) + (%s_MTLFmask == 1) + (%s_MTLFmask == 3))
                ' % (listScenes[0],listScenes[0],listScenes[0])
            expr2 = '((%s_MTLFmask == 0) + (%s_MTLFmask == 1) + (%s_MTLFmask == 3))
                ' % (listScenes[1],listScenes[1],listScenes[1])
            exprF = '%s * %s' % (expr1,expr2)
        else:
            expr1 = '((%s_MTLFmask == 0) + (%s_MTLFmask == 1) + (%s_MTLFmask == 3))
                ' % (listScenes[0],listScenes[0],listScenes[0])
            expr2 = '((%s_MTLFmask == 0) + (%s_MTLFmask == 1) + (%s_MTLFmask == 3))
                ' % (listScenes[1],listScenes[1],listScenes[1])
            exprF = '%s * %s' % (expr1,expr2)
            for num in range(2,numScenes):
                expr3 = '((%s_MTLFmask == 0) + (%s_MTLFmask == 1) + (%s_MTLFmask ==
                    3))' % (listScenes[num],listScenes[num],listScenes[num])
                exprF = '%s * %s' % (exprF,expr3)
        r.mapcalc(expression='FinalMask = %s' % exprF)
        r.mask(raster='FinalMask',maskcats='1')
    return 0
```

4.2 Exemples

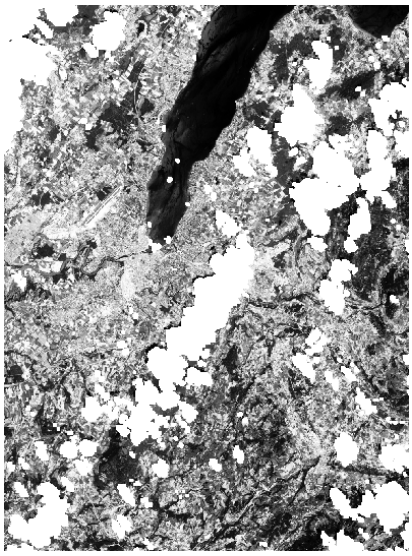
Trois tests traitant différents nombres de scènes ont été réalisés avec succès. Il est à noter qu'ils ont été exécutés avant que les options concernant la conversion des rasters de réflectances issues de la fonction Fmask (de Int16 à Float64) et la combinaison des différents masques ne soient possibles pour l'utilisateur. Le premier a été effectué sur une seule scène (LC81960282015153LGN00). Une fois le script *LiMES_batcher* exécuté et la scène téléchargée, le script *LiMES_grasser* a été lancé (Figure 19).

```
pablo@M4500-PL:~/Scripts/LiMES_20160510 > python LiMES_grasser_v003.py batch.txt
* LiMES_grasser_v003.py batch.txt STARTED
** Creating GRASS geneva_2 location
** Connecting to GRASS geneva_2 location
** Importing AOI to GRASS
** Starting Fmask function on LC81960282015153LGN00 scene
TIRS bands are valid: Original Fmask function will be executed
Fmask 3.3.1 version start ...
Cloud, cloud shadow, and snow detection for Landsat 8 images
Cloud/cloud shadow/snow dilated by 3/3/0 pixels (default)
Cloud probability threshold of 22.50% (default)
Read in header information & TIF images
From DNs to TOA ref & BT
Cloud & cloud shadow matching ...
Fmask finished for LC81960282015153LGN00_MTL.txt with 64.37% clear pixels
Elapsed time is 166.366694 seconds.
** Clipping LC81960282015153LGN00 scene
** Importing rasters to GRASS
** Creating MASK
All subsequent raster operations will be limited to the MASK area. Removing
or renaming raster map named 'MASK' will restore raster operations to
normal.
* LiMES_grasser_v003.py batch.txt COMPLETED in 183 seconds
```

FIGURE 19 – Exécution du script *LiMES_grasser* sur une seule scène

Une fois dans GRASS, il a été possible d'observer que chaque couche a été masquée selon l'output de la fonction Fmask (Figure 20).

(a) Output de la fonction Fmask. En blanc les pixels de nuage ou d'ombre de nuage, en orange les pixels clairs.



(b) Réflectances de la bande 4 masquées. Histogramme égalisé.

FIGURE 20 – Visualisation du masquage automatique de la scène LC81960282015153LGN00.

En sélectionnant deux scènes du même site, nous avons observé que la fonction Fmask alternative a été employée sur l'une d'elle (Figure 21).

```
pablo@M4500-PL ~/Scripts/LiMES_20160510 > python LiMES_grasser_v003.py batch.txt
* LiMES_grasser_v003.py batch.txt STARTED
** Creating GRASS guayaquil_51 location
** Connecting to GRASS guayaquil_51 location
** Importing AOI to GRASS
** Starting Fmask function on LC80110622016108LGN00 scene
TIRS bands are not valid: Alternative Fmask function will be executed
Fmask 3.3.0 beta version start ...
Cloud, cloud shadow, and snow detection for Landsat 8 images without TIRS (Sentinel 2 scenario)
Cloud/cloud shadow/snow dilated by 3/3/0 pixels (default)
Cloud probability threshold of 22.50% (default)
Read in header information & TIF images
From DNs to TOA ref & BT
Cloud & cloud shadow matching ...
Fmask finished for LC80110622016108LGN00_MTL.txt with 40.05% clear pixels
Elapsed time is 254.015352 seconds.
** Clipping LC80110622016108LGN00 scene
** Importing rasters to GRASS
 100%
** Starting Fmask function on LC80110622015249LGN00 scene
TIRS bands are valid: Original Fmask function will be executed
Fmask 3.3.1 version start ...
Cloud, cloud shadow, and snow detection for Landsat 8 images
Cloud/cloud shadow/snow dilated by 3/3/0 pixels (default)
Cloud probability threshold of 22.50% (default)
Read in header information & TIF images
From DNs to TOA ref & BT
Cloud & cloud shadow matching ...
Fmask finished for LC80110622015249LGN00_MTL.txt with 50.34% clear pixels
Elapsed time is 542.456242 seconds.
** Clipping LC80110622015249LGN00 scene
** Importing rasters to GRASS
 100%
** Creating MASK
 100%
All subsequent raster operations will be limited to the MASK area. Removing
or renaming raster map named 'MASK' will restore raster operations to
normal.
* LiMES_grasser_v003.py batch.txt COMPLETED in 892 seconds
```

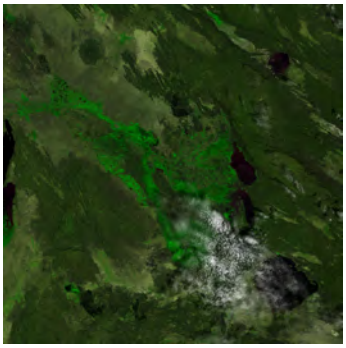
FIGURE 21 – Exécution du script *LiMES_grasser* sur deux scènes : LC80110622016108LGN00 et LC80110622015249LGN00.

Les réflectances et les masques issus de la fonction Fmask de chaque scène ont été exportés dans la base de données, et un masque global (FinalMask) permettant de ne retenir que les pixels qui sont clairs sur les deux scènes a été calculé (Figure 22).

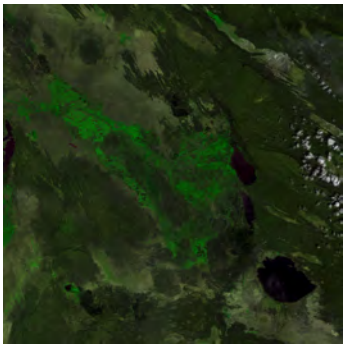
```
(Mon Jun  6 09:40:52 2016)
g.list type=raster
FinalMask
LC80110622015249LGN00_MTLFmask
LC80110622015249LGN00_T2
LC80110622015249LGN00_T3
LC80110622015249LGN00_T4
LC80110622015249LGN00_T5
LC80110622015249LGN00_T6
LC80110622015249LGN00_T7
LC80110622015249LGN00_T9
LC80110622016108LGN00_MTLFmask
LC80110622016108LGN00_T2
LC80110622016108LGN00_T3
LC80110622016108LGN00_T4
LC80110622016108LGN00_T5
LC80110622016108LGN00_T6
LC80110622016108LGN00_T7
LC80110622016108LGN00_T9
MASK
(Mon Jun  6 09:40:52 2016) Command finished (0 sec)
```

FIGURE 22 – Rasters présents dans la base de données, une fois le script exécuté.

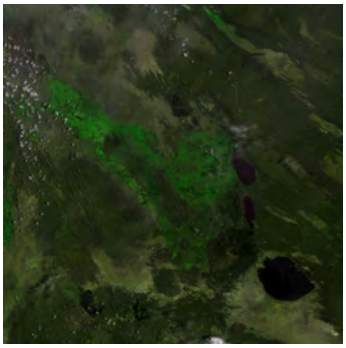
Le dernier exemple a pris compte 3 scènes du même site. Une fois le script exécuté, il a été observé que le masque global s'est correctement calculé, tenant compte des différents masques correspondant à chaque scène. Au vu de l'image composite RGB (4,5,2), le masquage a semblé être surestimé. Cependant, les cirrus (réflectances de la bande 9) ont confirmé la présence de nuages invisibles sur la composite (Figure 23).



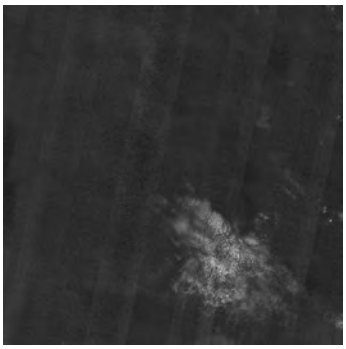
(a) Scène 1 : RGB (4,5,2)



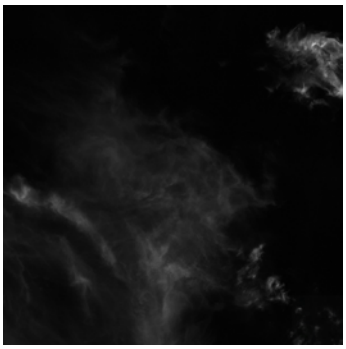
(b) Scène 2 : RGB (4,5,2)



(c) Scène 3 : RGB (4,5,2)



(d) Scène 1 : Cirrus (bande 9)



(e) Scène 2 : Cirrus (bande 9)

(f) Scène 3 : Cirrus (bande 9)

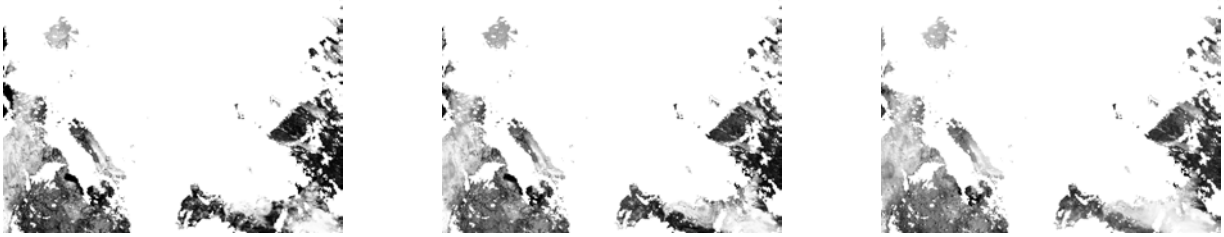
(g) Scène 1 : Masque binaire

(h) Scène 2 : Masque binaire

(i) Scène 3 : Masque binaire

FIGURE 23 – Composite RGB (4,5,2), masques et réflectances de la bande 9 (cirrus) correspondant à chaque scène. Scène 1 : LC81670522013296LGN00 ; Scène 2 : LC81670522015302LGN00 ; Scène 3 : LC81670522016097LGN00.

Le masque global calculé en fonction des 3 masques correspondant à chaque scène a été utilisé comme input dans la fonction *r.mask* (Figure 24).



(a) Scène 1 : Réflectances de la bande 4

(b) Scène 2 : Réflectances de la bande 4

(c) Scène 3 : Réflectances de la bande 4

FIGURE 24 – Réflectances de la bande 4 correspondant à chaque scène. Sont visibles seulement les pixels qui sont clairs sur les 3 scènes. Scène 1 : LC81670522013296LGN00 ; Scène 2 : LC81670522015302LGN00 ; Scène 3 : LC81670522016097LGN00.

4.3 Installation

Afin que le script *LiMES_grasser* modifié fonctionne et prenne compte de la fonction Fmask, il faut tout d'abord installer le logiciel Runtime MATLAB associé à la version de MATLAB utilisée pour compiler la fonction Fmask, dans notre cas : *MCR_R2016a*. Il peut être téléchargé depuis le site : `http://ch.mathworks.com/products/compiler/mcr/`. Lors de l'installation, il est important de noter le *LD_LIBRARY_PATH* (indiqué à fin de l'installation) et de l'introduire dans le fichier *config_cloud_mask.py* (Annexe D.4.4) associé au script *cloud_mask.py*. Cela détermine en effet le chemin de la librairie MATLAB permettant l'exécution de la fonction Fmask. Aussi, deux répertoires (autoFmask et autoFmaskS) contenant chacun le fichier exécutable des fonctions Fmask (autoFmask tenant compte des bandes TIRS et autoFmaskS n'en tenant pas compte) doivent figurer dans le dossier principal contenant les scripts.