3  Réhaussement et visualisation d’images – Traitement d'images satellites avec Python

# 3  Réhaussement et visualisation d’images

Assurez-vous de lire ce préambule avant d’exécutez le reste du notebook.

## 3.1 🚀 Préambule

### 3.1.1 🎯 Objectifs

Dans ce chapitre, nous abordons quelques techniques de réhaussement et de visualisation d’images. Ce chapitre est aussi disponible sous la forme d’un notebook Python:

### 3.1.2 Librairies

Les librairies qui vont être explorées dans ce chapitre sont les suivantes:

* SciPy
* NumPy
* OpenCV
* RasterIO
* Geemap
* [rioxarray](https://corteva.github.io/rioxarray/stable/index.html)
* xarray

Dans l’environnement Google Colab, seul rioxarray et GDAL doivent être installés:

%%capture  
!apt-get update  
!apt-get install gdal-bin libgdal-dev  
!pip install -q rioxarray  
!pip install -qU "geemap[workshop]"

Vérifier les importations:

import numpy as np  
import rioxarray as rxr  
from scipy import signal  
import xarray as xr  
import xrscipy  
import matplotlib.pyplot as plt

### 3.1.3 Données

Nous allons utilisez les images suivantes dans ce chapitre:

%%capture  
!wget https://github.com/sfoucher/TraitementImagesPythonVol1/raw/refs/heads/main/data/chapitre01/subset\_RGBNIR\_of\_S2A\_MSIL2A\_20240625T153941\_N0510\_R011\_T18TYR\_20240625T221903.tif -O RGBNIR\_of\_S2A.tif  
!wget https://github.com/sfoucher/opengeos-data/raw/refs/heads/main/raster/landsat7.tif -O landsat7.tif  
!wget https://github.com/sfoucher/opengeos-data/raw/refs/heads/main/images/berkeley.jpg -O berkeley.jpg  
!wget https://github.com/sfoucher/TraitementImagesPythonVol1/raw/refs/heads/main/data/chapitre01/subset\_0\_of\_S1A\_split\_NR\_Cal\_Deb\_ML\_Spk\_SRGR.tif -O SAR.tif

Vérifiez que vous êtes capable de les lire :

with rxr.open\_rasterio('berkeley.jpg', mask\_and\_scale= True) as img\_rgb:  
 print(img\_rgb)  
with rxr.open\_rasterio('RGBNIR\_of\_S2A.tif', mask\_and\_scale= True) as img\_rgbnir:  
 print(img\_rgbnir)  
with rxr.open\_rasterio('subset\_0\_of\_S1A\_split\_NR\_Cal\_Deb\_ML\_Spk\_SRGR.tif', mask\_and\_scale= True) as img\_SAR:  
 print(img\_SAR)

## 3.2 Réhaussements visuels

Le but du réhaussement visuel d’une image vise principalement à améliorer la qualité visuelle d’une image en améliorant le contraste, la dynamique ou la texture d’une image. De manière générale, ce réhaussement ne modifie pas la donnée d’origine mais est plutôt appliquée dynamiquement à l’affichage pour des fins d’inspection visuelle.

### 3.2.1 Statistiques d’une image

On peut considérer un ensemble de statistique globales pour chacune des bandes d’une image: - valeurs minimales et maximales - valeurs moyennes, médianes et quantiles - écart-types, skewness et kurtosis Ces statistiques doivent être calculées pour chaque bande d’une image multispectrale.

En ligne de commande, gdalinfo permet d’interroger rapidement un fichier image pour connaitre les statistiques de base:

|  |
| --- |
| !gdalinfo -stats landsat7.tif |

Bloc de code 3.1: Statistiques d’une image avec gdal

Driver: GTiff/GeoTIFF  
Files: landsat7.tif  
 landsat7.tif.aux.xml  
Size is 2181, 1917  
Coordinate System is:  
PROJCS["WGS 84 / Pseudo-Mercator",  
 GEOGCS["WGS 84",  
 DATUM["WGS\_1984",  
 SPHEROID["WGS 84",6378137,298.257223563,  
 AUTHORITY["EPSG","7030"]],  
 AUTHORITY["EPSG","6326"]],  
 PRIMEM["Greenwich",0,  
 AUTHORITY["EPSG","8901"]],  
 UNIT["degree",0.0174532925199433,  
 AUTHORITY["EPSG","9122"]],  
 AUTHORITY["EPSG","4326"]],  
 PROJECTION["Mercator\_1SP"],  
 PARAMETER["central\_meridian",0],  
 PARAMETER["scale\_factor",1],  
 PARAMETER["false\_easting",0],  
 PARAMETER["false\_northing",0],  
 UNIT["metre",1,  
 AUTHORITY["EPSG","9001"]],  
 AXIS["X",EAST],  
 AXIS["Y",NORTH],  
 EXTENSION["PROJ4","+proj=merc +a=6378137 +b=6378137 +lat\_ts=0.0 +lon\_0=0.0 +x\_0=0.0 +y\_0=0 +k=1.0 +units=m +nadgrids=@null +wktext +no\_defs"],  
 AUTHORITY["EPSG","3857"]]  
Origin = (-13651650.000000000000000,4576290.000000000000000)  
Pixel Size = (30.000000000000000,-30.000000000000000)  
Metadata:  
 AREA\_OR\_POINT=Area  
 OVR\_RESAMPLING\_ALG=NEAREST  
 TIFFTAG\_RESOLUTIONUNIT=1 (unitless)  
 TIFFTAG\_XRESOLUTION=1  
 TIFFTAG\_YRESOLUTION=1  
Image Structure Metadata:  
 COMPRESSION=DEFLATE  
 INTERLEAVE=PIXEL  
Corner Coordinates:  
Upper Left (-13651650.000, 4576290.000) (122d38' 5.49"W, 37d58'40.08"N)  
Lower Left (-13651650.000, 4518780.000) (122d38' 5.49"W, 37d34'10.00"N)  
Upper Right (-13586220.000, 4576290.000) (122d 2'49.53"W, 37d58'40.08"N)  
Lower Right (-13586220.000, 4518780.000) (122d 2'49.53"W, 37d34'10.00"N)  
Center (-13618935.000, 4547535.000) (122d20'27.51"W, 37d46'26.05"N)  
Band 1 Block=512x512 Type=Byte, ColorInterp=Red  
 Min=19.000 Max=233.000   
 Minimum=19.000, Maximum=233.000, Mean=98.433, StdDev=21.164  
 NoData Value=0  
 Overviews: 1091x959, 546x480  
 Metadata:  
 STATISTICS\_MAXIMUM=233  
 STATISTICS\_MEAN=98.433096940153  
 STATISTICS\_MINIMUM=19  
 STATISTICS\_STDDEV=21.164021026458  
Band 2 Block=512x512 Type=Byte, ColorInterp=Green  
 Min=19.000 Max=178.000   
 Minimum=19.000, Maximum=178.000, Mean=55.068, StdDev=22.204  
 NoData Value=0  
 Overviews: 1091x959, 546x480  
 Metadata:  
 STATISTICS\_MAXIMUM=178  
 STATISTICS\_MEAN=55.067787534804  
 STATISTICS\_MINIMUM=19  
 STATISTICS\_STDDEV=22.203571974581  
Band 3 Block=512x512 Type=Byte, ColorInterp=Blue  
 Min=19.000 Max=187.000   
 Minimum=19.000, Maximum=187.000, Mean=43.341, StdDev=20.330  
 NoData Value=0  
 Overviews: 1091x959, 546x480  
 Metadata:  
 STATISTICS\_MAXIMUM=187  
 STATISTICS\_MEAN=43.340507443056  
 STATISTICS\_MINIMUM=19  
 STATISTICS\_STDDEV=20.32987736339

Les librairies de base comme xarray et numpy peuvent facilement produire des statistiques comme avec la fonction [stats](https://rasterio.readthedocs.io/en/stable/api/rasterio.io.html#rasterio.io.BufferedDatasetWriter.stats):

import rasterio as rio  
import numpy as np  
with rio.open('landsat7.tif') as src:  
 stats= src.stats()  
 print(stats)

La librairie xarray donne accès à des fonctionnalités plus sophistiquées comme le calcul des quantiles:

import rioxarray as riox  
with riox.open\_rasterio('landsat7.tif', masked= True) as src:  
 print(src)  
quantiles = src.quantile(dim=['x','y'], q=[.025,.25,.5,.75,.975])  
quantiles

<xarray.DataArray (band: 3, y: 1917, x: 2181)> Size: 50MB  
[12542931 values with dtype=float32]  
Coordinates:  
 \* band (band) int64 24B 1 2 3  
 \* x (x) float64 17kB -1.365e+07 -1.365e+07 ... -1.359e+07  
 \* y (y) float64 15kB 4.576e+06 4.576e+06 ... 4.519e+06 4.519e+06  
 spatial\_ref int64 8B 0  
Attributes:  
 AREA\_OR\_POINT: Area  
 OVR\_RESAMPLING\_ALG: NEAREST  
 TIFFTAG\_RESOLUTIONUNIT: 1 (unitless)  
 TIFFTAG\_XRESOLUTION: 1  
 TIFFTAG\_YRESOLUTION: 1  
 STATISTICS\_MAXIMUM: 233  
 STATISTICS\_MEAN: 98.433096940153  
 STATISTICS\_MINIMUM: 19  
 STATISTICS\_STDDEV: 21.164021026458  
 scale\_factor: 1.0  
 add\_offset: 0.0

<xarray.DataArray (quantile: 5, band: 3)> Size: 120B  
array([[ 54., 19., 19.],  
 [ 85., 38., 27.],  
 [ 99., 54., 38.],  
 [111., 69., 57.],  
 [140., 102., 89.]])  
Coordinates:  
 \* band (band) int64 24B 1 2 3  
 \* quantile (quantile) float64 40B 0.025 0.25 0.5 0.75 0.975

xarray.DataArray

* quantile: 5
* band: 3

54.0 19.0 19.0 85.0 38.0 27.0 ... 111.0 69.0 57.0 140.0 102.0 89.0

array([[ 54., 19., 19.],  
 [ 85., 38., 27.],  
 [ 99., 54., 38.],  
 [111., 69., 57.],  
 [140., 102., 89.]])

Coordinates: (2)

band

(band)

int64

1 2 3

array([1, 2, 3])

quantile

(quantile)

float64

0.025 0.25 0.5 0.75 0.975

array([0.025, 0.25 , 0.5 , 0.75 , 0.975])

Indexes: (2)

band

PandasIndex

PandasIndex(Index([1, 2, 3], dtype='int64', name='band'))

quantile

PandasIndex

PandasIndex(Index([0.025, 0.25, 0.5, 0.75, 0.975], dtype='float64', name='quantile'))

Attributes: (0)

#### 3.2.1.1 Calcul de l’histogramme

Le calcul d’un histogramme pour une image (une bande) permet d’avoir une vue plus détaillée de la répartition des valeurs radiométriques. Le calcul d’un histogramme nécessite minimalement de faire le choix d’une valeur du nombre de *bins* (ou de la largeur). Un *bin* est un intervalle de valeurs pour lequel on peut calculer le nombre de valeurs observées dans l’image. La fonction de base pour ce type de calcul est la fonction numpy.histogram():

import numpy as np  
array = np.random.randint(0,10,100) # 100 valeurs aléatoires entre 0 et 10  
hist, bin\_limites = np.histogram(array, density=True)  
print('valeurs :',hist)  
print(';imites :',bin\_limites)

valeurs : [0.14444444 0.11111111 0.13333333 0.08888889 0.11111111 0.12222222  
 0.11111111 0.08888889 0.12222222 0.07777778]  
;imites : [0. 0.9 1.8 2.7 3.6 4.5 5.4 6.3 7.2 8.1 9. ]

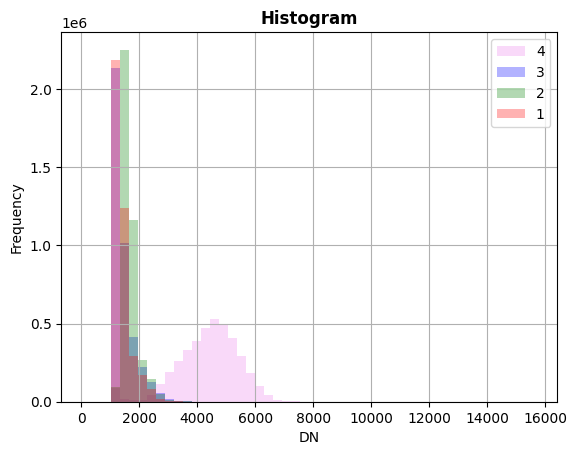
Le calcul se fait avec 10 intervalles par défaut.

Pour des besoins de visualisation, le calcul des valeurs extrêmes de l’histogramme peut aussi se faire via les quantiles comme discutés auparavant.

##### 3.2.1.1.1 Visualisation des histogrammes

La librarie rasterio est probablement l’outil le plus simples pour visualiser rapidement des histogrammes sur une image multi-spectrale:

import rasterio as rio  
from rasterio.plot import show\_hist  
with rio.open('RGBNIR\_of\_S2A.tif') as src:  
 show\_hist(src, bins=50, lw=0.0, stacked=False, alpha=0.3,histtype='stepfilled', title="Histogram")



### 3.2.2 Réhaussements linéaires

Le réhaussement linéaire d’une image est la forme la plus simple de réhaussement, elle consiste 1) à optimiser les valeurs des pixels d’une image afin de maximiser la dynamique disponibles à l’affichage, ou 2) changer le format de stockage des valeurs (e.g. de 8 bit à 16 bit):

\[ \text{nouvelle valeur d'un pixel} = \frac{\text{valeur d'un pixel} - min\_0}{max\_0 - min\_0}\times (max\_1 - min\_1)+min\_1\] Par cette opération, on passe de la dynamique de départ (\(max\_0 - min\_0\)) vers la dynamique cible (\(max\_1 - min\_1\)). Bien que cette opération semble triviale, il est important d’être conscient des trois contraintes suivantes: 1. **Faire attention à la dynamique cible**, ainsi, pour sauvegarder une image en format 8 bit, on utilisera alors \(max\_1=255\) et \(min\_1=0\). 2. **Préservation de la valeur de no data** : il faut faire attention à la valeur \(min\_1\) dans le cas d’une valeur présente pour *no\_data*. Par exemple, si *no\_data=0* alors il faut s’assurer que \(min\_1>0\). 3. **Précision du calcul** : si possible réaliser la division ci-dessus en format *float*

### 3.2.3 Réhaussements non linéaires

Calcul d’histogrammes, étirement, égalisation, styling

### 3.2.4 Composés couleurs

Le système visuel humain est sensible seulement à la partie visible du spectre électromagnétique qui compose les couleurs de l’arc-en-ciel du bleu au rouge. L’ensemble des couleurs du spectre visible peut être obtenu à partir du mélange de trois couleurs primaires (rouge, vert et bleu). Ce système de décomposition à trois couleurs est à la base de la plupart des systèmes de visualisation ou de représentation de l’information de couleur. On peut trouver des variantes comme le système HSV (*Hue-Saturation-Value*) utilisé en encodage de données vidéos.

## 3.3 Visualisation

### 3.3.1 Visualisation en Python

Il faut d’entrée mentionner que Python n’est pas vraiment fait pour visualiser de la donnée de grande taille, le niveau d’interactivité est aussi plus limité. Néanmoins, il est possible de visualiser de petites images avec la librairie Matplotlib.

### 3.3.2 Outils de visualisation

Il existe plusieurs outils gratuits de visualisation d’une image satellite, on peut mentionner les deux principaux: - QGIS - ESA Snap

### 3.3.3 Visualisation sur le Web

Une des meilleures pratiques pour visualiser une image de grande taille est d’utiliser un service de type Web Mapping Service (WMS). Cependant, type de service nécessite une architecture client-serveur qui est plus complexe à mettre en place.

Google Earth Engine offre des moyens de visualiser de la donnée locale: 🔖 *Working with Local Geospatial Data* — via [17. Geemap — Introduction to GIS Programming](https://geog-312.gishub.org/book/geospatial/geemap.html#working-with-local-geospatial-data)

🔖 \_\_ — via [data/raster at main · opengeos/data](https://github.com/opengeos/data/tree/main/raster)

### 3.3.4 Visualisation 3D

drapper une image satellite sur un DEM

## 3.4 Quiz de révision du chapitre

## 3.5 Exercices de révision