Tutoriel 4

March 31, 2021

1 Image visualization : Les différentes façons de transformer et d'agréger nighttime lights data

Nous identifierons et visualiserons les composites annuels in the DMSP-OLS nighttime lights series.

La collection d'images DMSP-OLS dans Google Earth Engine, sourced by the Earth Observation Group, Payne Institute for Public Policy, Colorado School of Mines, a été traité comme une série de composites annuels par an, par satellite (certaines années incluent deux capteurs satellites).

1.1 La collection d'images de lumières nocturnes DMSP-OLS

1.1.1 Initialisation

Nous importons `geemap` et initialisons un objet geemap centré sur le Maroc.

Nous ajouterons également le fond de carte satellite (daytime) par défaut.

[5]:
```
try:
    import geemap, ee
except ModuleNotFoundError:
    if 'google.colab' in str(get_ipython()):
        print("package not found, installing w/ pip in Google Colab...")
        !pip install geemap
    else:
        print("package not found, installing w/ conda...")
        !conda install mamba -c conda-forge -y
        !mamba install geemap -c conda-forge -y
    import geemap, ee
```

[13]:
```
try:
        ee.Initialize()
except Exception as e:
        ee.Authenticate()
        ee.Initialize()

# set our initial map parameters for Morocco 31.7917° N, 7.0926° W

center_lat = 31.7917
```

```
center_lon = 7.0926
zoomlevel=5

# initialize our map
map1 = geemap.Map(center=[center_lat,center_lon], zoom=zoomlevel)
map1.add_basemap('SATELLITE')

map1.addLayerControl()
```

1.1.2 DMSP-OLS Image Collection

La collection d'images pour DMSP-OLS se trouve à l'adresse: `NOAA/DMSP-OLS/NIGHTTIME_LIGHTS`

On peut obtenir l'adresse des différentes collections sur le site de NOAA ou sur GEE.

[8]:
```
dmsp = ee.ImageCollection("NOAA/DMSP-OLS/NIGHTTIME_LIGHTS")
```

Quel est le nombre total d'images dans cette collection? (La satelite s'étend de 1992 à 2013, mais certaines années contiennent plusieurs images en raison du chevauchement des satellites.)

Dans l'éditeur de Google Earth Engine, nous appelons la méthode `.size ()` sur notre collection.

Avec `geemap`, nous pouvons faire la même chose, cependant, `.size ()` produira un "objet" de taille, nous devons donc ajouter une étape supplémentaire consistant à utiliser la méthode `.getInfo ()` pour qu'elle imprime notre information.

[9]:
```
print(dmsp.size().getInfo())
```

```
35
```

Quelle est la plage de dates de notre collection?

GEE a un ensemble de méthodes appelées «Reducers», qui effectuent une gamme de fonctions, telles que l'obtention de la somme ou de la valeur moyenne d'une collection. Ils sont très pratiques. La fonction `Reducer.minMax ()` peut être utilisée pour obtenir une plage de dates.

[ ]:
```
imgrange = dmsp.reduceColumns(ee.Reducer.minMax(), ["system:time_start"])
start = ee.Date(imgrange.get('min')).getInfo()
end = ee.Date(imgrange.get('max')).getInfo()
print(f"Date range: {start, end}")
```

```
Date range: ({'type': 'Date', 'value': 694224000000}, {'type': 'Date', 'value':
1356998400000})
```

[ ]:
```
imgrange = dmsp.reduceColumns(ee.Reducer.minMax(), ["system:time_start"])
start = ee.Date(imgrange.get('min')).getInfo()['value']
end = ee.Date(imgrange.get('max')).getInfo()['value']
print(f"Date range: {start, end}")
```

```
Date range: (694224000000, 1356998400000)
```

1.1.3 Date conversions...

Le suivi du temps dans un monde avec de nombreux fuseaux horaires est incroyablement compliqué. Par conséquent, les applications logicielles (y compris celles qui ont prétraité nos fichiers DMSP-OLS ainsi que GEE) utilisent ce qu'on appelle «Unix time», ce que nous voyons ici. L'heure Unix ou heure Posix (aussi appelée Unix Timestamp) est une mesure du temps basée sur le nombre de secondes écoulées depuis le 1er janvier 1970 00:00:00 UTC, hors secondes intercalaires.

C'est une étape supplémentaire pour convertir cela en une date que nous, les humains, comprenons en Python, mais la compréhension des timestamps est essentielle lorsque vous travaillez avec des données temporelles en particulier des données produites par des satellites en orbite autour de la planète.

Pour convertir ces valeurs d'heure Unix en dates lisibles, nous allons: 1. Divisez par 1000 pour convertir en secondes et 2. Utilisez la bibliothèque pratique `datetime` de Python pour convertir en objet datetime en utilisant la méthode`utcfromtimestamp` (rappelez-vous que l'heure est en UTC!) 3. Convertissez cet objet datetime en une chaîne que nous pouvons lire avec le modèle: Année (% Y) - Mois (% m) - Jour (% d) Heure (% H): Minute (% M): Seconde (% S)

[ ]:
```
imgrange = dmsp.reduceColumns(ee.Reducer.minMax(), ["system:time_start"])
start = ee.Date(imgrange.get('min')).getInfo()['value']
end = ee.Date(imgrange.get('max')).getInfo()['value']

# convert date
from datetime import datetime
start = datetime.utcfromtimestamp(start/1000).strftime('%Y-%m-%d %H:%M:%S')
end = datetime.utcfromtimestamp(end/1000).strftime('%Y-%m-%d %H:%M:%S')
print(f"Date range: {start, end}")
```

```
Date range: ('1992-01-01 00:00:00', '2013-01-01 00:00:00')
```

Alors pour récapituler...

En Python:

```
from datetime import datetime

imgrange = dmsp.reduceColumns(ee.Reducer.minMax(), ["system:time_start"])
start = ee.Date(imgrange.get('min')).getInfo()['value']
end = ee.Date(imgrange.get('max')).getInfo()['value']

start = datetime.utcfromtimestamp(start/1000).strftime('%Y-%m-%d %H:%M:%S')
end = datetime.utcfromtimestamp(end/1000).strftime('%Y-%m-%d %H:%M:%S')
print(f"Date range: {start, end}")
```

S'il s'agit d'une méthode que vous voudrez exécuter plus d'une fois, vous devez créer une fonction à partir de ces lignes de code qui n'a besoin que de la collection d'images en tant que paramètre.

[ ]:
```
def get_date_range(img_collection):
    imgrange = img_collection.reduceColumns(ee.Reducer.minMax(), ["system:
 ↪time_start"])
```

```
    start = ee.Date(imgrange.get('min')).getInfo()['value']
    end = ee.Date(imgrange.get('max')).getInfo()['value']

    start = datetime.utcfromtimestamp(start/1000).strftime('%Y-%m-%d %H:%M:%S')
    end = datetime.utcfromtimestamp(end/1000).strftime('%Y-%m-%d %H:%M:%S')
    print(f"Date range: {start, end}")
```

1.2 Exemple 1: DMSP-OLS annual composite de l'année 1996

DMSP-OLS satellites by year

	F10	F12	F14	F15	F16	F18
1992	F101992					
1993	F101993					
1994	F101994	F121994				
1995		F121995				
1996		F121996				
1997		F121997	F141997			
1998		F121998	F141998			
1999		F121999	F141999			
2000			F142000	F152000		
2001			F142001	F152001		
2002			F142002	F152002		
2003			F142003	F152003		
2004				F152004	F162004	
2005				F152005	F162005	
2006				F152006	F162006	
2007				F152007	F162007	
2008					F162008	
2009					F162009	
2010						F182010
2011						F182011
2012						F182012
2013						F182013

Pour 1996, nous n'avons qu'une seule option à choisir: `F121996`

On ajouter cette image en tant que couche à notre carte.

Appliquons également le masque aux zones sans données et ajustons l'opacité `Opacity` à 75%.

[14]:
```
dmsp1996 = ee.Image("NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F101992")

# initialize our map
map2 = geemap.Map(center=[center_lat,center_lon], zoom=zoomlevel)
map2.add_basemap('SATELLITE')
```

```
map2.addLayer(dmsp1996.mask(dmsp1996), {}, "DMSP-OLS 1996", opacity=0.75)

map2.addLayerControl()
```

1.3 Exemple 2: Ajout d'un deuxième composite annuel d'une autre année (2010) et création d'un panneau coulissant pour afficher et comparer les deux

Note Attention, ceci est basé sur `ipyleaflet` une bibliothèque Python qui ne fonctionne pas bien avec Google Colab, donc ce code ne fonctionnera pas dans l'environnement Google Colab mais devrait sur votre machine locale.

[11]:
```
# get the satellite name from the reference table
dmsp2010 = ee.Image("NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182010")

map2.addLayer(dmsp2010.mask(dmsp2010), {}, "DMSP-OLS 2010", opacity=0.75)
```

[15]:
```
# initialize our map
map3 = geemap.Map(center=[center_lat,center_lon], zoom=zoomlevel)
map3.add_basemap('SATELLITE')

# generate tile layers
dmsp1996_tile = geemap.ee_tile_layer(dmsp1996.mask(dmsp1996), {}, 'DMSP-OLS␣
↪1996', opacity=0.75)
dmsp2010_tile = geemap.ee_tile_layer(dmsp2010.mask(dmsp2010), {}, 'DMSP-OLS␣
↪2010', opacity=0.75)

# create split map
map3.split_map(left_layer=dmsp1996_tile, right_layer=dmsp2010_tile)
```

Nous avons maintenant trouvé et visualisé un composite annuel et appris quelques moyens utiles d'accéder aux méta-données. Ce curseur est bon comme visualisation, mais comme indiqué dans le premier tutoriel, les satellites DMSP-OLS n'ont pas le on-board calibration. En conséquence, le passage d'un satellite à un autre (comme c'est le cas lors de la comparaison de 1996 à 2010) pourrait inclure des variations de capteur qui ne représentent pas les changements réels observés dans la lumière.

C'est pourquoi nous devons être très prudent lorsque nous comparons ou analysons directement les changements dans la série DMSP-OLS.

La façon de résoudre ce problème est d'ajuster nos composites annuels DMSP-OLS à travers un processus que nous appellerons «intercalibration» et c'est le sujet d'un tutoriel ultérieur.