Anil Prasad. V A

24-10-2023

Sea Surface Temperature extraction using sentinel 3 level-2 data

DATA Used:

The Sentinel-3A and 3B SLSTR(Sea and Land Surface Temperature Radiometer) level 2 WST satellite data.

Level-2 WST - (Water Single Temperature, i.e. Sea Surface Temperature following the GHRSST specifications)

Data availability:

SLSTR Level 2 Sea Surface Temperature (SST)

Sensing before 05/04/2018	Sensing from 06/04/2018 to 31/12/2020	Sensing after 01/01/2021
SLSTR Level 2 Sea Surface Temperature (SST) (version BC003) - Sentinel-3 - Reprocessed (EO:EUM:DAT:0582)	Data currently available only from the EUMETSAT Data Centre	**SLSTR Level 2 Sea Surface Temperature (SST) - Sentinel-3** (EO:EUM:DAT:0412)

Data store used to download SST(18-10-2023): **EO:EUM:DAT:0412**

Downloaded File format : **netCDF**

Resolution of SST **: 1KM**

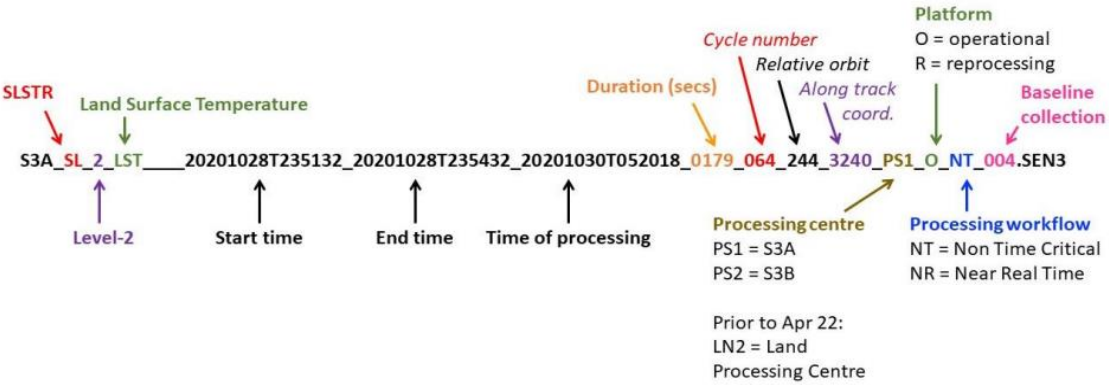
Packages used for downloading files : **eumdac**

Area of Interest:

	Latitude	Longitude
Site 1	20.24755	86.61137

Site 2	20.96497	71.5649

Data naming conversions:



(The above figure shows the naming conversions of LST data. WST data is also similar)

13 datasets downloaded for the date 18-10-2023.

```
Found Datasets: 13 datasets for the given time range
S3A_SL_2_WST____20231018T172738_20231018T173038_20231018T185023_0180_104_312_0180_MAR_O_NR_003.SEN3
S3B_SL_2_WST____20231018T165143_20231018T165443_20231018T180124_0179_085_169_0360_MAR_O_NR_003.SEN3
S3B_SL_2_WST____20231018T164843_20231018T165143_20231018T180122_0179_085_169_0180_MAR_O_NR_003.SEN3
S3A_SL_2_WST____20231018T154938_20231018T155238_20231018T165302_0179_104_311_0360_MAR_O_NR_003.SEN3
S3A_SL_2_WST____20231018T154638_20231018T154938_20231018T170323_0180_104_311_0180_MAR_O_NR_003.SEN3
S3B_SL_2_WST____20231018T054049_20231018T054349_20231018T074717_0179_085_162_2520_MAR_O_NR_003.SEN3
S3A_SL_2_WST____20231018T044144_20231018T044444_20231018T070206_0179_104_304_2700_MAR_O_NR_003.SEN3
S3A_SL_2_WST____20231018T043844_20231018T044144_20231018T070204_0179_104_304_2520_MAR_O_NR_003.SEN3
S3B_SL_2_WST____20231018T043331_20231018T061431_20231019T140951_6059_085_161______MAR_O_NT_003.SEN3
S3B_SL_2_WST____20231018T040249_20231018T040549_20231018T060637_0179_085_161_2700_MAR_O_NR_003.SEN3
S3B_SL_2_WST____20231018T035949_20231018T040249_20231018T060422_0179_085_161_2520_MAR_O_NR_003.SEN3
S3A_SL_2_WST____20231018T033126_20231018T051226_20231019T142832_6059_104_303______MAR_O_NT_003.SEN3
S3B_SL_2_WST____20231018T025232_20231018T043331_20231019T140937_6059_085_160______MAR_O_NT_003.SEN3
```

PYTHON CODE FOR DOMLOADING AND ANALYSING netCDF data.

In [2]:
```
pip install  eumdac
```

```
Requirement already satisfied: eumdac in d:\anaconda3\lib\site-packages (2.1.0)
Requirement already satisfied: requests>=2.5.0 in d:\anaconda3\lib\site-packages (from eumdac) (2.28.1)
Requirement already satisfied: pyyaml in d:\anaconda3\lib\site-packages (from eumdac) (6.0)
Requirement already satisfied: charset-normalizer<3,>=2 in d:\anaconda3\lib\site-packages (from requests>=2.5.0->eumdac) (2.0.
4)
Requirement already satisfied: idna<4,>=2.5 in d:\anaconda3\lib\site-packages (from requests>=2.5.0->eumdac) (3.3)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in d:\anaconda3\lib\site-packages (from requests>=2.5.0->eumdac) (1.26.11)
Requirement already satisfied: certifi>=2017.4.17 in d:\anaconda3\lib\site-packages (from requests>=2.5.0->eumdac) (2022.9.14)
Note: you may need to restart the kernel to use updated packages.
```

```
WARNING: Ignoring invalid distribution -yproj (d:\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -yproj (d:\anaconda3\lib\site-packages)

[notice] A new release of pip is available: 23.2.1 -> 23.3
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [103]:
```
import eumdac
import datetime
import shutil
# Insert your personal key and secret into the single quotes
consumer_key = 'qa88SaI8oAEkvXPs2uruI3L9Ttga'
consumer_secret = 'H8WpHfekE2diMSRFwGSRY93zngYa'

credentials = (consumer_key, consumer_secret)

token = eumdac.AccessToken(credentials)

print(f"This token '{token}' expires {token.expiration}")
```

```
This token 'af9ecd7c-66fd-3d8b-9dd9-61d59cf9f1cc' expires 2023-10-24 14:00:15.272801
```

Consumer key and Consumer secret key generated using **eumetsat account.**

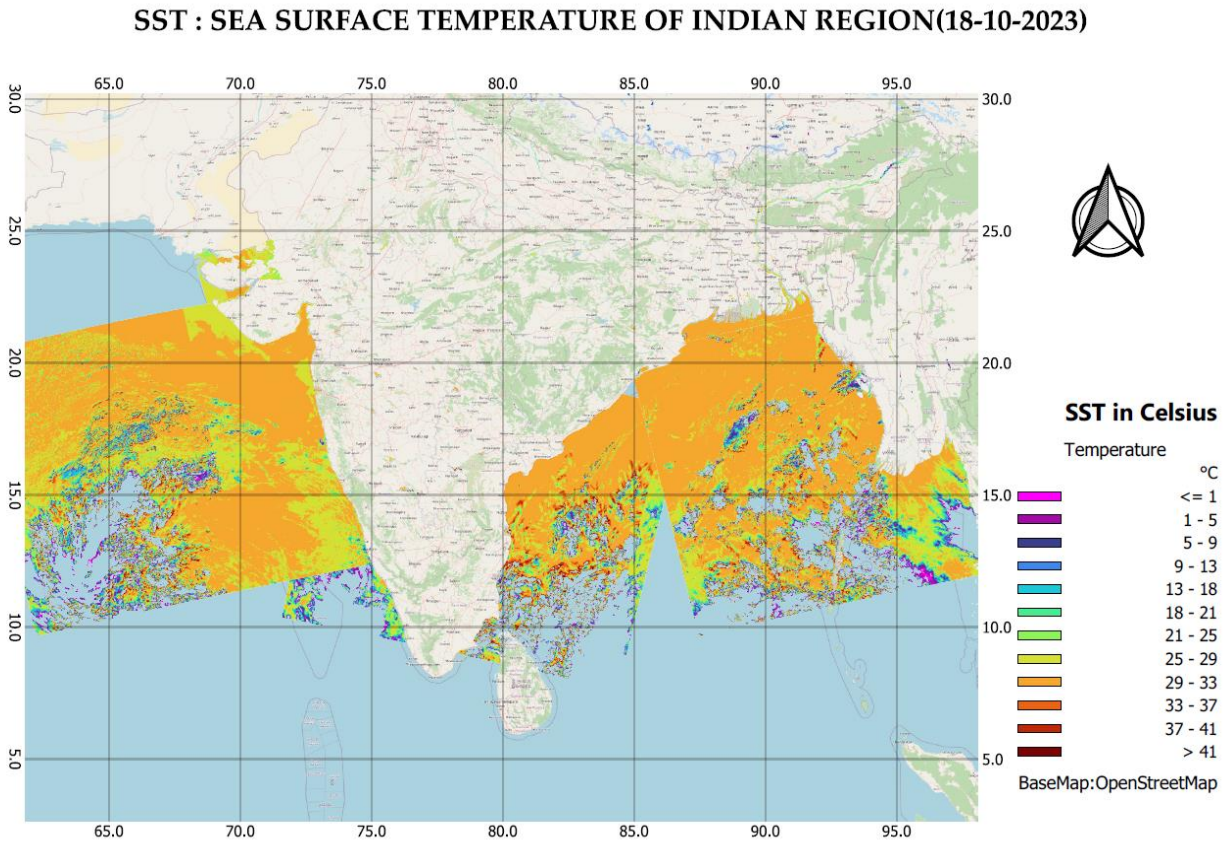
In [28]:
```
# Add vertices for polygon, wrapping back to the start point.
#ploygon box created which includes the given lat and lon

datastore = eumdac.DataStore(token)
#print(datastore.collections)
selected_collection = datastore.get_collection('EO:EUM:DAT:0412')
ex=0.5
geometry = [[86.61137+ex, 20.24755+ex],[86.61137+ex, 20.96497+ex],[71.5649+ex, 20.96497+ex],[71.5649+ex, 20.24755+ex],[86.61137+e


# Set sensing start and end time
start = datetime.datetime(2023, 10, 18, 0, 0)
end = datetime.datetime(2023, 10, 18, 23, 59)

# Retrieve datasets that match our filter
products = selected_collection.search(
    geo='POLYGON(({}))'.format(','.join(["{} {}".format(*coord) for coord in geometry])),
    dtstart=start,
    dtend=end)

print(f'Found Datasets: {len(products)} datasets for the given time range')

for product in products:
    print(str(product))
```

```
Found Datasets: 13 datasets for the given time range
S3A_SL_2_WST____20231018T172738_20231018T173038_20231018T185023_0180_104_312_0180_MAR_O_NR_003.SEN3
S3B_SL_2_WST____20231018T165143_20231018T165443_20231018T180124_0179_085_169_0360_MAR_O_NR_003.SEN3
S3B_SL_2_WST____20231018T164843_20231018T165143_20231018T180122_0179_085_169_0180_MAR_O_NR_003.SEN3
S3A_SL_2_WST____20231018T154938_20231018T155238_20231018T165302_0179_104_311_0360_MAR_O_NR_003.SEN3
S3A_SL_2_WST____20231018T154638_20231018T154938_20231018T170323_0180_104_311_0180_MAR_O_NR_003.SEN3
S3B_SL_2_WST____20231018T054049_20231018T054349_20231018T074717_0179_085_162_2520_MAR_O_NR_003.SEN3
S3A_SL_2_WST____20231018T044144_20231018T044444_20231018T070206_0179_104_304_2700_MAR_O_NR_003.SEN3
S3A_SL_2_WST____20231018T043844_20231018T044144_20231018T070204_0179_104_304_2520_MAR_O_NR_003.SEN3
S3B_SL_2_WST____20231018T043331_20231018T061431_20231019T140951_6059_085_161_____MAR_O_NT_003.SEN3
S3B_SL_2_WST____20231018T040249_20231018T040549_20231018T060637_0179_085_161_2700_MAR_O_NR_003.SEN3
S3B_SL_2_WST____20231018T035949_20231018T040249_20231018T060422_0179_085_161_2520_MAR_O_NR_003.SEN3
S3A_SL_2_WST____20231018T033126_20231018T051226_20231019T142832_6059_104_303_____MAR_O_NT_003.SEN3
S3B_SL_2_WST____20231018T025232_20231018T043331_20231019T140937_6059_085_160_____MAR_O_NT_003.SEN3
```

In [29]:
```
#downloading the avalibale sentinael 3 WST data in the region

import os

download_path = r"C:\Users\Legion\Desktop\climaview\2023_10_18"  # Replace with your desired path

for product in products:
    with product.open() as fsrc, \
            open(os.path.join(download_path, fsrc.name), mode='wb') as fdst:
        shutil.copyfileobj(fsrc, fdst)
        print(f'Download of product {product} finished.')

print('All downloads are finished.')
```

```
Download of product S3A_SL_2_WST____20231018T172738_20231018T173038_20231018T185023_0180_104_312_0180_MAR_O_NR_003.SEN3 finishe
d.
Download of product S3B_SL_2_WST____20231018T165143_20231018T165443_20231018T180124_0179_085_169_0360_MAR_O_NR_003.SEN3 finishe
d.
Download of product S3B_SL_2_WST____20231018T164843_20231018T165143_20231018T180122_0179_085_169_0180_MAR_O_NR_003.SEN3 finishe
d.
Download of product S3A_SL_2_WST____20231018T154938_20231018T155238_20231018T165302_0179_104_311_0360_MAR_O_NR_003.SEN3 finishe
d.
Download of product S3A_SL_2_WST____20231018T154638_20231018T154938_20231018T170323_0180_104_311_0180_MAR_O_NR_003.SEN3 finishe
d.
Download of product S3B_SL_2_WST____20231018T054049_20231018T054349_20231018T074717_0179_085_162_2520_MAR_O_NR_003.SEN3 finishe
d.
Download of product S3A_SL_2_WST____20231018T044144_20231018T044444_20231018T070206_0179_104_304_2700_MAR_O_NR_003.SEN3 finishe
d.
Download of product S3A_SL_2_WST____20231018T043844_20231018T044144_20231018T070204_0179_104_304_2520_MAR_O_NR_003.SEN3 finishe
d.
Download of product S3B_SL_2_WST____20231018T043331_20231018T061431_20231019T140951_6059_085_161_____MAR_O_NT_003.SEN3 finishe
d.
Download of product S3B_SL_2_WST____20231018T040249_20231018T040549_20231018T060637_0179_085_161_2700_MAR_O_NR_003.SEN3 finishe
d.
Download of product S3B_SL_2_WST____20231018T035949_20231018T040249_20231018T060422_0179_085_161_2520_MAR_O_NR_003.SEN3 finishe
d.
Download of product S3A_SL_2_WST____20231018T033126_20231018T051226_20231019T142832_6059_104_303_____MAR_O_NT_003.SEN3 finishe
d.
Download of product S3B_SL_2_WST____20231018T025232_20231018T043331_20231019T140937_6059_085_160_____MAR_O_NT_003.SEN3 finishe
d.
All downloads are finished.
```

Five datasets used to prepare the SST Map.



Tools used: SNAP, QGIS and PYTHON

Steps:

- **Georeferencing file using SNAP**
- **Converting Kelvin to degree Celsius using SNAP (Band math: sst-273.15)**
- **Exporting netCDF to GEOTIFF using SNAP**
- **Legend preparation and Map making using QGIS**

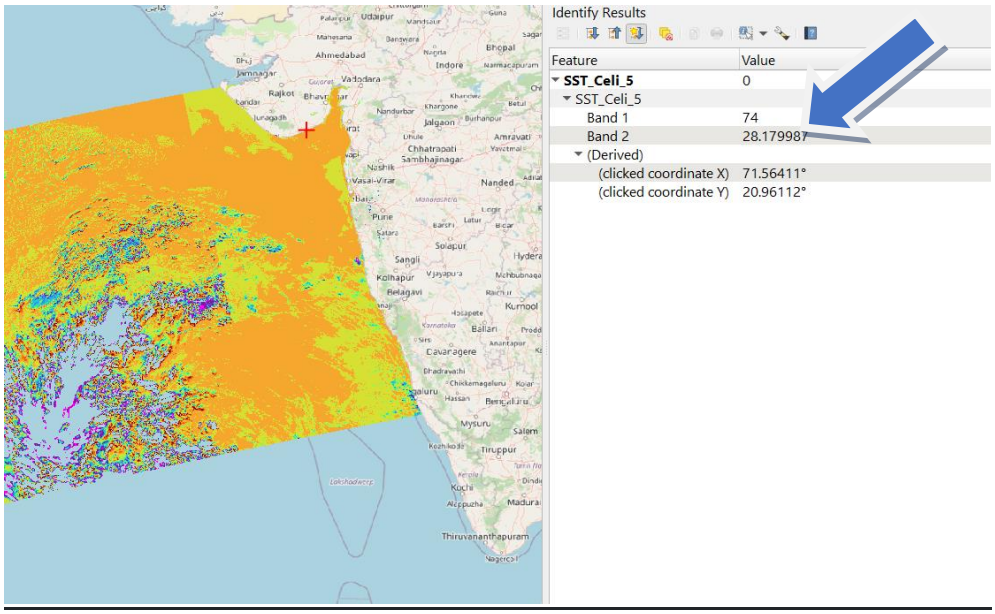
<u>PYTHON CODE FOR EXTRACTING SPECIFIC LOCATION SST value</u>

Extracting temperature value - coordinates(20.96497,71.5649)

```
In [126]: import rasterio
          data = rasterio.open(r"C:\Users\Legion\Desktop\climaview\2023_10_18\SST_DATA_1.tif")

In [127]: #The exported geotiff file have two bands, band 0 is grey scale band and band 1 contains temp value
          # so read data from the first band

          z = data.read()[1]
          # check the crs of the data
          data.crs

Out[127]: CRS.from_wkt('GEOGCS["WGS 84",DATUM["World Geodetic System 1984",SPHEROID["WGS 84",6378137,298.257223563]],PRIMEM["Greenwich",
          0],UNIT["degree",0.0174532925199433,AUTHORITY["EPSG","9122"]],AXIS["Latitude",NORTH],AXIS["Longitude",EAST]]')

In [128]:
          # check the bounding-box of the data
          data.bounds

Out[128]: BoundingBox(left=59.349449157714844, bottom=9.60668590291341, right=75.48120632425943, top=22.99787139892578)

In [129]:
          # since the raster is in regular lon/lat grid (4326) we can use
          # `dat.index()` to identify the index of a given lon/lat pair
          # (e.g. it expects coordinates in the native crs of the data)

          def getval(lon, lat):
              idx = data.index(lon, lat, precision=1E-6)
              print("Sea Surface Temperature of location",z[idx])
              return
          getval(71.5649, 20.96497) #location given in the task

          Sea Surface Temperature of location 28.179987
```

Sea Surface Temperature of the SITE 2 is **28.1798**

Cross checking with QGIS



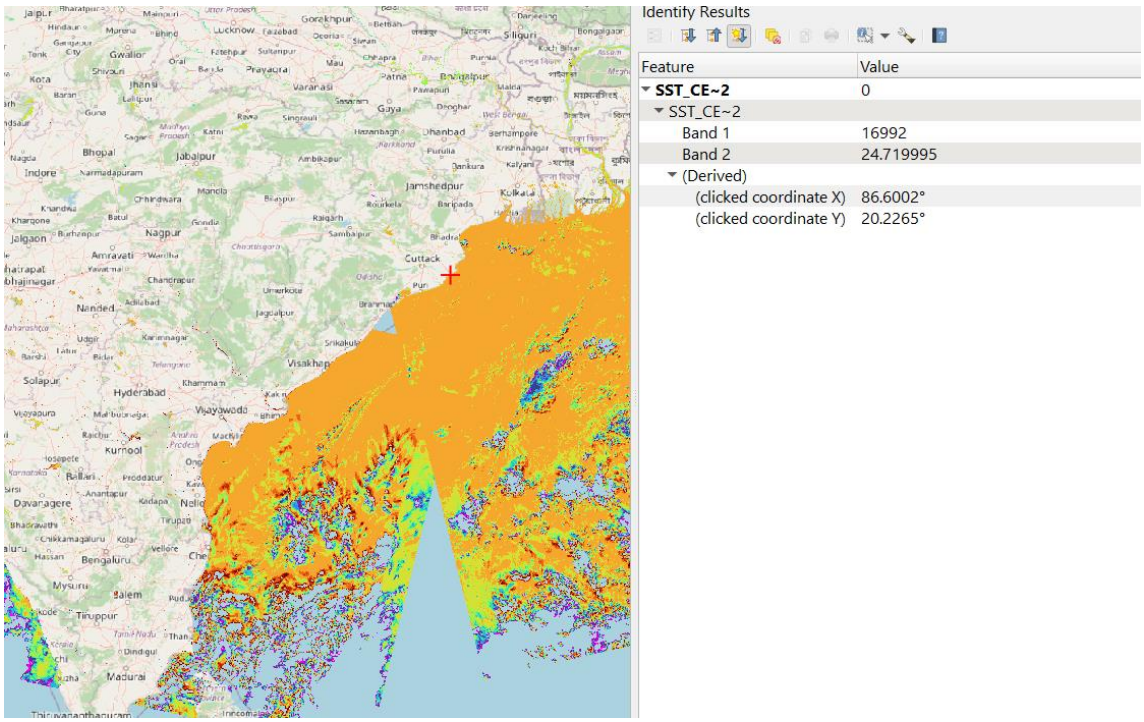
The Sea surface temperature value of the SITE 1 conatins no value. So manually took the near by location and extracted the themperature value

Location: (20.2239, 86.5899)

```
In [135]: import rasterio
data = rasterio.open(r"C:\Users\Legion\Desktop\climaview\2023_10_18\SST_DATA_2.tif")

#The exported geotiff file have two bands, band 0 is grey scale band and band 1 contains temp value
# so read data from the first band[1]
z = data.read()[1]

# check the crs of the data
data.crs

# check the bounding-box of the data
data.bounds

# since the raster is in regular lon/lat grid (4326) we can use
# `dat.index()` to identify the index of a given lon/lat pair
# (e.g. it expects coordinates in the native crs of the data)
def getval(lon, lat):
    idx = data.index(lon, lat, precision=1E-6)
    print("Sea Surface Temperature of given location:",z[idx])
    return
getval(86.5899,20.2239) #the given loaction contains no SST value so i took the near lat and lon value


Sea Surface Temperature of given location: 23.140009
```

Sea Surface Temperature of the SITE 1 is **23.14009**

Cross checking with QGIS



RESULTS

	Latitude	Longitude	SST

Site 1	20.24755	86.61137	NOT available
Site 2	20.96497	71.5649	28.1798
Site_1a	20.2239	86.5899	23.14009

Site_1a:Alternative near location for site 1

ABOUT SLSTR LEVEL2 WST PRODUCT

The SLSTR Level-2 WST product respects the Group for High Resolution Sea Surface Temperature (GHRSST) L2P specification. and includes a single SST field derived from the best performing single-coefficient SST field in any given part of the swath, plus a number of supporting data fields providing context for the SST fields.

This file, indexed by across track and along track dimensions and by reference time, provides for each thermal SLSTR channel:

- the latitude and longitude coordinates of each pixel
- the SST value
- the SST time deviation from reference time and from analysis field
- the Single Sensor Error Statistic (SSES) bias and standard deviation estimate
- several contextual parameters (wind speed at 10 m, the fractional sea-ice contamination, the aerosol contamination indicator for each pixel) with the time difference between the SST and these data measurements
- a quality flag (gathering information about sensor and surface type, geographical contamination, problems during processing, Level-1B flags) and a quality indicator (from 0, default value unknown quality, 1, excellent quality to 3, extremely suspect) for SST.
- satellite zenith angle of each pixel and each reference time
- associated Top Of Atmosphere (TOA) Brightness Temperature (BT)
- associated TOA noise equivalent BT
- SST total uncertainty.