```
#!/usr/bin/env python3

"""
This code plots MSG HRIT/HRIT RSS Image data on predefined areas
written by IMS (Israel Meteorological Service) R%D department
for questions contact: vadislavskye@ims.gov.il
"""


from satpy import Scene
from satpy.writers import to_image
from pyresample.geometry import AreaDefinition
from pyresample import parse_area_file
from glob import glob
import trollimage.colormap
import os
import sys
import time
import datetime
import numpy as np
import scipy.io
import multiprocessing
import warnings
warnings.filterwarnings("ignore")

# used functions

def plot_FOG():

        global global_scene
        global IMAGEPATH
        global time_slot

        # define filenames out
        fout1 = '%s/%s_4GISFULL_fog.png' % (IMAGEPATH,time_slot.strftime('%Y%m%d%H%
M'))
        #fout2 = '%s/%s_4GISFULL_fog.tif' % (IMAGEPATH,time_slot.strftime('%Y%m%d%H%
M'))

        # plot web full fog
        composite = 'night_fog'
        global_scene.load([composite])
        Europe_HRV = parse_area_file(area_file, 'Europe_HRV')[0]
        Europe_HRV_scene = global_scene.resample(Europe_HRV,radius_of_influence=50000)
        Europe_HRV_scene.save_dataset(composite, fout1)

        # https://satpy.readthedocs.io/en/stable/writers.html
        # Europe_HRV_scene.save_dataset(composite,fout2,writer='geotiff')

        print('Night FOG Images ready!!!')
        return 0

def plot_IR108():

        global global_scene
        global IMAGEPATH
        global time_slot

        # define filenames out
        fout1 = '%s/%s_4GISFULL_ir.png' % (IMAGEPATH,time_slot.strftime('%Y%m%d%H%M'))
        #fout2 = '%s/%s_4GISFULL_ir.tif' % (IMAGEPATH,time_slot.strftime('%Y%m%d%H%
```

```
M'))

        # plot web full IR
        global_scene.load([10.8])
        Europe_HRV = parse_area_file(area_file, 'Europe_HRV')[0]
        Europe_HRV_scene = global_scene.resample(Europe_HRV,radius_of_influence=50000)

        # https://satpy.readthedocs.io/en/stable/writers.html

        img = to_image(Europe_HRV_scene[10.8])
        img.colorize(cmIR)
        img.save(fout1)

        #img = to_image(Europe_HRV_scene[10.8])
        #img.colorize(cmIR)
        #img.save(fout2,writer='geotiff')

        print('IR 10.8  Images ready!!!')
        return 0

def plot_IR108_colored():

        global global_scene
        global IMAGEPATH
        global time_slot

        # define filenames out
        fout1 = '%s/%s_4GISFULL_ir_colored.png' % (IMAGEPATH,time_slot.strftime('%Y%m%
d%H%M'))

        # plot web full IR colored
        global_scene.load([10.8])
        Europe_HRV = parse_area_file(area_file, 'Europe_HRV')[0]
        Europe_HRV_scene = global_scene.resample(Europe_HRV,radius_of_influence=50000)

        # https://satpy.readthedocs.io/en/stable/writers.html

        img = to_image(Europe_HRV_scene[10.8])
        img.colorize(cmIRcoloredSPC)
        img.save(fout1)

        print('IR 10.8 colored Images ready!!!')
        return 0

def plot_WV():

        global global_scene
        global IMAGEPATH
        global time_slot

        # define filenames out
        fout1 = '%s/%s_4GISFULL_water_vapor.png' % (IMAGEPATH,time_slot.strftime('%Y%m
%d%H%M'))

        # plot web full WV
        global_scene.load(['WV_062'])
        Europe_HRV = parse_area_file(area_file, 'Europe_HRV')[0]
        Europe_HRV_scene = global_scene.resample(Europe_HRV,radius_of_influence=50000)
        img = to_image(Europe_HRV_scene[6.7] )
        img.invert([True])
        img.stretch("linear")
```

```
        img.save(fout1)

        print('WV Image ready!!!')
        return 0

def plot_Day_Natural():

        global global_scene
        global IMAGEPATH
        global time_slot

        # define filenames out
        fout1 = '%s/%s_4GISFULL_natural.png' % (IMAGEPATH,time_slot.strftime('%Y%m%d%H
%M'))

        # plot web full natural
        composite = 'natural_color'
        global_scene.load([composite])
        Europe_HRV = parse_area_file(area_file, 'Europe_HRV')[0]
        Europe_HRV_scene = global_scene.resample(Europe_HRV,radius_of_influence=50000)
        Europe_HRV_scene.save_dataset(composite, fout1)

        print('Day Natural Images ready!!!')
        return 0

def plot_HRV():

        global global_scene
        global IMAGEPATH
        global time_slot

        # define filenames out
        fout1 = '%s/%s_4GISFULL_hrv.png' % (IMAGEPATH,time_slot.strftime('%Y%m%d%H%
M'))
        fout2 = '%s/%s_synoptic_map_bg_hrv.png' % (IMAGEPATH,time_slot.strftime('%Y%m%
d%H%M'))

        # plot web full HRV
        global_scene.load(['HRV'])
        Europe_HRV = parse_area_file(area_file, 'Europe_HRV')[0]
        Europe_HRV_scene = global_scene.resample(Europe_HRV)

        STERE_FORECAST = parse_area_file(area_file,"STERE_FORECAST")[0]
        STERE_FORECAST_scene = global_scene.resample
(STERE_FORECAST,radius_of_influence=50000)

        img = to_image(Europe_HRV_scene['HRV'])
        img.stretch("linear")
        img.gamma(2.0)
        img.save(fout1)

        img = to_image(STERE_FORECAST_scene['HRV'])
        img.stretch("linear")
        img.gamma(2.0)
        img.save(fout2)

        print('HRV Image ready!!!')
        return 0

def plot_Airmass():
```

```
        global global_scene
        global IMAGEPATH
        global time_slot

        # define filenames out
        fout1 = '%s/%s_4GISFULL_air_mass.png' % (IMAGEPATH,time_slot.strftime('%Y%m%d%
H%M'))
        fout2 = '%s/%s_synoptic_map_bg_air_mass.png' % (IMAGEPATH,time_slot.strftime
('%Y%m%d%H%M'))

        # plot web full airmass
        composite = 'airmass'
        global_scene.load([composite])
        Europe_HRV = parse_area_file(area_file, 'Europe_HRV')[0]
        Europe_HRV_scene = global_scene.resample(Europe_HRV)
        Europe_HRV_scene.save_dataset(composite, fout1)

        STERE_FORECAST = parse_area_file(area_file,"STERE_FORECAST")[0]
        STERE_FORECAST_scene = global_scene.resample
(STERE_FORECAST,radius_of_influence=50000)
        STERE_FORECAST_scene.save_dataset(composite, fout2)

        print('Airmass Images ready!!!')
        return 0

def plot_Dust():

        global global_scene
        global IMAGEPATH
        global time_slot

        # define filenames out
        fout1 = '%s/%s_4GISFULL_dust.png' % (IMAGEPATH,time_slot.strftime('%Y%m%d%H%
M'))

        # plot web full dust
        composite = 'dust'
        global_scene.load([composite])
        Europe_HRV = parse_area_file(area_file, 'Europe_HRV')[0]
        Europe_HRV_scene = global_scene.resample(Europe_HRV,radius_of_influence=50000)
        Europe_HRV_scene.save_dataset(composite, fout1)

        print('Dust Images ready!!!')
        return 0

def plot_Day_storms():

        global global_scene
        global IMAGEPATH
        global time_slot

        # define filenames out
        fout1 = '%s/%s_4GISFULL_day_storms.png' % (IMAGEPATH,time_slot.strftime('%Y%m%
d%H%M'))

        # plot web full day storms
        composite = 'convection'
        global_scene.load([composite])
        Europe_HRV = parse_area_file(area_file, 'Europe_HRV')[0]
        Europe_HRV_scene = global_scene.resample(Europe_HRV,radius_of_influence=50000)
        Europe_HRV_scene.save_dataset(composite, fout1)
```

```
        print('Day stroms Images ready!!!')
        return 0

def plot_Day_clouds():

        global global_scene
        global IMAGEPATH
        global time_slot

        # define filenames out
        fout1 = '%s/%s_4GISFULL_day_clouds.png' % (IMAGEPATH,time_slot.strftime('%Y%m%
d%H%M'))

        # plot web full day clouds
        composite = 'day_microphysics'
        global_scene.load([composite])
        Europe_HRV = parse_area_file(area_file, 'Europe_HRV')[0]
        Europe_HRV_scene = global_scene.resample(Europe_HRV,radius_of_influence=50000)
        Europe_HRV_scene.save_dataset(composite, fout1)

        print('Day clouds Images ready!!!')
        return 0


def plotSatImage(index):

        # Case of FOG images
        if index==1:
                plot_FOG()

        # Case of IR images
        if index==2:
                plot_IR108()

        if index==3:
                plot_IR108_colored()

        # Case of Water Vapour images
        if index==4:
                plot_WV()

        # Case of Day Natural images
        if index==5:
                plot_Day_Natural()

        # Case of HRV
        if index==6:
                plot_HRV()

        # Case of AirMass
        if index==7:
                plot_Airmass()

        # Case of Dust
        if index==8:
                plot_Dust()

        # Case of Day storms
        if index==9:
                plot_Day_storms()
```

```
        # Case of Day clouds
        if index==10:
                plot_Day_clouds()

        return 0


##############################

################
# main program #
################

if __name__ == "__main__":

        start = time.time()

        ##############################
        # Define directories
        CONFIGDIR = "../Config"
        DATADIR = '../DATA_IN'
        IMAGEPATH = '../imageout'
        LAYERSPATH = '../Config'

        time_slot = datetime.datetime.strptime(sys.argv[1],'%Y%m%d%H%M')

        # read satellite data
        print("Processing channel IR108 %s UTC" % (time_slot.strftime('%d/%m/%Y %H:%
M')))
        filenames = glob('%s/*%s*' % (DATADIR,time_slot.strftime('%Y%m%d%H%M')))
        global_scene = Scene(reader="seviri_l1b_hrit", filenames=filenames)
        #global_scene.load
(['VIS006','VIS008','IR_016','WV_062',0.6,0.8,1.6,3.9,6.2,7.3,8.7,9.7,10.8,12.0,13.4,"HRV"])

        # https://www.kaggle.com/martinraspaud/quickstart-with-msg-seviri
        print(global_scene.available_composite_ids())

        # get areas definition from configuration file: areas.def
        area_file = '%s/areas.def' % CONFIGDIR

        # colormaps
        # VIS

        cmVIS = trollimage.colormap.Colormap((2.48, (0.0, 0.0, 0.0)),
(3.5199999999999996, (0.027170868347338936, 0.027170868347338936,
0.027170868347338936)), (4.5599999999999996, (0.054341736694677872,
0.054341736694677872, 0.054341736694677872)), (5.5999999999999996,
(0.081512605042016809, 0.081512605042016809, 0.081512605042016809)),
(6.6399999999999988, (0.10868347338935574, 0.10868347338935574,
0.10868347338935574)), (7.6799999999999997, (0.13585434173669469,
0.13585434173669469, 0.13585434173669469)), (8.7199999999999989,
(0.16302521008403362, 0.16302521008403362, 0.16302521008403362)), (9.759999999999998,
(0.19019607843137254, 0.19019607843137254, 0.19019607843137254)),
(10.799999999999999, (0.21736694677871149, 0.21736694677871149,
0.21736694677871149)), (11.839999999999998, (0.24453781512605044,
0.24453781512605044, 0.24453781512605044)), (12.879999999999999,
(0.27170868347338939, 0.27170868347338939, 0.27170868347338939)),
(13.919999999999998, (0.29887955182072834, 0.29887955182072834,
0.29887955182072834)), (14.959999999999997, (0.32605042016806723,
0.32605042016806723, 0.32605042016806723)), (15.999999999999998,
```

(0.35322128851540613, 0.35322128851540613, 0.35322128851540613)),
(17.039999999999999, (0.38039215686274508, 0.38039215686274508,
0.38039215686274508)), (17.039999999999999, (0.38039215686274508,
0.38039215686274508, 0.38039215686274508)), (18.079999999999998,
(0.40028011204481789, 0.40028011204481789, 0.40028011204481789)),
(19.120000000000001, (0.42016806722689076, 0.42016806722689076,
0.42016806722689076)), (20.16, (0.44005602240896358, 0.44005602240896358,
0.44005602240896358)), (21.199999999999999, (0.45994397759103639,
0.45994397759103639, 0.45994397759103639)), (22.240000000000002,
(0.47983193277310926, 0.47983193277310926, 0.47983193277310926)),
(23.280000000000001, (0.49971988795518207, 0.49971988795518207,
0.49971988795518207)), (24.32, (0.51960784313725494, 0.51960784313725494,
0.51960784313725494)), (25.359999999999999, (0.5394957983193277, 0.5394957983193277,
0.5394957983193277)), (26.400000000000002, (0.55938375350140057, 0.55938375350140057,
0.55938375350140057)), (27.440000000000001, (0.57927170868347344,
0.57927170868347344, 0.57927170868347344)), (28.480000000000004, (0.5991596638655462,
0.5991596638655462, 0.5991596638655462)), (29.520000000000003, (0.61904761904761907,
0.61904761904761907, 0.61904761904761907)), (30.560000000000002,
(0.63893557422969194, 0.63893557422969194, 0.63893557422969194)),
(31.600000000000001, (0.6588235294117647, 0.6588235294117647, 0.6588235294117647)),
(31.600000000000001, (0.6588235294117647, 0.6588235294117647, 0.6588235294117647)),
(32.640714285714289, (0.67366946778711478, 0.67366946778711478,
0.67366946778711478)), (33.681428571428576, (0.68851540616246498,
0.68851540616246498, 0.68851540616246498)), (34.722142857142856,
(0.70336134453781518, 0.70336134453781518, 0.70336134453781518)),
(35.762857142857143, (0.71820728291316527, 0.71820728291316527,
0.71820728291316527)), (36.803571428571431, (0.73305322128851536,
0.73305322128851536, 0.73305322128851536)), (37.844285714285718,
(0.74789915966386555, 0.74789915966386555, 0.74789915966386555)),
(38.885000000000005, (0.76274509803921564, 0.76274509803921564,
0.76274509803921564)), (39.925714285714285, (0.77759103641456584,
0.77759103641456584, 0.77759103641456584)), (40.966428571428573,
(0.79243697478991593, 0.79243697478991593, 0.79243697478991593)), (42.00714285714286,
(0.80728291316526612, 0.80728291316526612, 0.80728291316526612)), (43.04785714285714,
(0.82212885154061621, 0.82212885154061621, 0.82212885154061621)),
(44.088571428571427, (0.83697478991596641, 0.83697478991596641,
0.83697478991596641)), (45.129285714285714, (0.85182072829131661,
0.85182072829131661, 0.85182072829131661)), (46.170000000000002, (0.8666666666666667,
0.8666666666666667, 0.8666666666666667)), (46.170000000000002, (0.8666666666666667,
0.8666666666666667, 0.8666666666666667)), (47.210000000000001, (0.873109243697479,
0.873109243697479, 0.873109243697479)), (48.25, (0.8795518207282913,
0.8795518207282913, 0.8795518207282913)), (49.289999999999999, (0.8859943977591036,
0.8859943977591036, 0.8859943977591036)), (50.329999999999998, (0.89243697478991602,
0.89243697478991602, 0.89243697478991602)), (51.369999999999997,
(0.89887955182072832, 0.89887955182072832, 0.89887955182072832)),
(52.409999999999997, (0.90532212885154062, 0.90532212885154062,
0.90532212885154062)), (53.450000000000003, (0.91176470588235292,
0.91176470588235292, 0.91176470588235292)), (54.489999999999995,
(0.91820728291316522, 0.91820728291316522, 0.91820728291316522)),
(55.530000000000001, (0.92464985994397753, 0.92464985994397753,
0.92464985994397753)), (56.569999999999993, (0.93109243697478983,
0.93109243697478983, 0.93109243697478983)), (57.609999999999999,
(0.93753501400560213, 0.93753501400560213, 0.93753501400560213)),
(58.649999999999999, (0.94397759103641454, 0.94397759103641454,
0.94397759103641454)), (59.689999999999998, (0.95042016806722696,
0.95042016806722696, 0.95042016806722696)), (60.729999999999997,
(0.95686274509803926, 0.95686274509803926, 0.95686274509803926)),
(60.729999999999997, (0.95686274509803926, 0.95686274509803926,
0.95686274509803926)), (61.731724137931032, (0.95753887762001355,
0.95753887762001355, 0.95753887762001355)), (62.733448275862067,
(0.95821501014198784, 0.95821501014198784, 0.95821501014198784)),

```
(63.735172413793101, (0.95889114266396214, 0.95889114266396214,
0.95889114266396214)), (64.736896551724129, (0.95956727518593643,
0.95956727518593643, 0.95956727518593643)), (65.738620689655164,
(0.96024340770791072, 0.96024340770791072, 0.96024340770791072)),
(66.740344827586199, (0.96091954022988513, 0.96091954022988513,
0.96091954022988513)), (67.742068965517234, (0.96159567275185931,
0.96159567275185931, 0.96159567275185931)), (68.743793103448269,
(0.96227180527383371, 0.96227180527383371, 0.96227180527383371)),
(69.745517241379304, (0.96294793779580801, 0.96294793779580801,
0.96294793779580801)), (70.747241379310339, (0.9636240703177823, 0.9636240703177823,
0.9636240703177823)), (71.748965517241373, (0.96430020283975659, 0.96430020283975659,
0.96430020283975659)), (72.750689655172408, (0.96497633536173089,
0.96497633536173089, 0.96497633536173089)), (73.752413793103443,
(0.96565246788370518, 0.96565246788370518, 0.96565246788370518)),
(74.754137931034478, (0.96632860040567958, 0.96632860040567958,
0.96632860040567958)), (75.755862068965513, (0.96700473292765377,
0.96700473292765377, 0.96700473292765377)), (76.757586206896548,
(0.96768086544962817, 0.96768086544962817, 0.96768086544962817)),
(77.759310344827583, (0.96835699797160246, 0.96835699797160246,
0.96835699797160246)), (78.761034482758618, (0.96903313049357676,
0.96903313049357676, 0.96903313049357676)), (79.762758620689652,
(0.96970926301555105, 0.96970926301555105, 0.96970926301555105)),
(80.764482758620687, (0.97038539553752534, 0.97038539553752534,
0.97038539553752534)), (81.766206896551722, (0.97106152805949963,
0.97106152805949963, 0.97106152805949963)), (82.767931034482757,
(0.97173766058147404, 0.97173766058147404, 0.97173766058147404)),
(83.769655172413792, (0.97241379310344822, 0.97241379310344822,
0.97241379310344822)), (84.771379310344827, (0.97308992562542262,
0.97308992562542262, 0.97308992562542262)), (85.773103448275862,
(0.97376605814739692, 0.97376605814739692, 0.97376605814739692)),
(86.774827586206897, (0.97444219066937121, 0.97444219066937121,
0.97444219066937121)), (87.776551724137931, (0.9751183231913455, 0.9751183231913455,
0.9751183231913455)), (88.778275862068966, (0.9757944557133198, 0.9757944557133198,
0.9757944557133198)), (89.780000000000001, (0.97647058823529409, 0.97647058823529409,
0.97647058823529409)), (89.780000000000001, (0.97647058823529409,
0.97647058823529409, 0.97647058823529409)), (90.802000000000007,
(0.97882352941176465, 0.97882352941176465, 0.97882352941176465)),
(91.823999999999998, (0.98117647058823521, 0.98117647058823521,
0.98117647058823521)), (92.846000000000004, (0.98352941176470587,
0.98352941176470587, 0.98352941176470587)), (93.867999999999995,
(0.98588235294117654, 0.98588235294117654, 0.98588235294117654)),
(94.890000000000001, (0.9882352941176471, 0.9882352941176471, 0.9882352941176471)),
(95.912000000000006, (0.99058823529411766, 0.99058823529411766,
0.99058823529411766)), (96.933999999999997, (0.99294117647058822,
0.99294117647058822, 0.99294117647058822)), (97.956000000000003,
(0.99529411764705888, 0.99529411764705888, 0.99529411764705888)),
(98.977999999999994, (0.99764705882352944, 0.99764705882352944,
0.99764705882352944)), (100.0, (1.0, 1.0, 1.0)))

        cmIR = trollimage.colormap.Colormap((216.63999999999999, (0.0, 0.0, 0.0)),
(217.67624999999998, (0.013725490196078431, 0.013725490196078431,
0.013725490196078431)), (218.71249999999998, (0.027450980392156862,
0.027450980392156862, 0.027450980392156862)), (219.74874999999997,
(0.041176470588235294, 0.041176470588235294, 0.041176470588235294)),
(220.78499999999997, (0.054901960784313725, 0.054901960784313725,
0.054901960784313725)), (221.82124999999999, (0.068627450980392163,
0.068627450980392163, 0.068627450980392163)), (222.85749999999999,
(0.082352941176470587, 0.082352941176470587, 0.082352941176470587)),
(223.89374999999998, (0.096078431372549025, 0.096078431372549025,
0.096078431372549025)), (224.92999999999998, (0.10980392156862745,
0.10980392156862745, 0.10980392156862745)), (225.96624999999997,
```

```
(0.12352941176470589, 0.12352941176470589, 0.12352941176470589)),
(227.00249999999997, (0.13725490196078433, 0.13725490196078433,
0.13725490196078433)), (228.03874999999996, (0.15098039215686274,
0.15098039215686274, 0.15098039215686274)), (229.07499999999999,
(0.16470588235294117, 0.16470588235294117, 0.16470588235294117)),
(230.11124999999998, (0.17843137254901961, 0.17843137254901961,
0.17843137254901961)), (231.14749999999998, (0.19215686274509805,
0.19215686274509805, 0.19215686274509805)), (232.18374999999997,
(0.20588235294117646, 0.20588235294117646, 0.20588235294117646)),
(233.21999999999997, (0.2196078431372549, 0.2196078431372549, 0.2196078431372549)),
(233.21999999999997, (0.2196078431372549, 0.2196078431372549, 0.2196078431372549)),
(234.25937499999998, (0.23284313725490197, 0.23284313725490197,
0.23284313725490197)), (235.29874999999998, (0.24607843137254901,
0.24607843137254901, 0.24607843137254901)), (236.33812499999996, (0.2593137254901961,
0.2593137254901961, 0.2593137254901961)), (237.37749999999997, (0.27254901960784311,
0.27254901960784311, 0.27254901960784311)), (238.41687499999998,
(0.28578431372549018, 0.28578431372549018, 0.28578431372549018)),
(239.45624999999995, (0.29901960784313725, 0.29901960784313725,
0.29901960784313725)), (240.49562499999996, (0.31225490196078431,
0.31225490196078431, 0.31225490196078431)), (241.53499999999997,
(0.32549019607843138, 0.32549019607843138, 0.32549019607843138)),
(242.57437499999997, (0.33872549019607845, 0.33872549019607845,
0.33872549019607845)), (243.61374999999998, (0.35196078431372552,
0.35196078431372552, 0.35196078431372552)), (244.65312499999996,
(0.36519607843137253, 0.36519607843137253, 0.36519607843137253)),
(245.69249999999997, (0.3784313725490196, 0.3784313725490196, 0.3784313725490196)),
(246.73187499999997, (0.39166666666666666, 0.39166666666666666,
0.39166666666666666)), (247.77124999999995, (0.40490196078431373,
0.40490196078431373, 0.40490196078431373)), (248.81062499999996, (0.4181372549019608,
0.4181372549019608, 0.4181372549019608)), (249.84999999999997, (0.43137254901960786,
0.43137254901960786, 0.43137254901960786)), (249.84999999999997,
(0.43137254901960786, 0.43137254901960786, 0.43137254901960786)),
(250.88749999999996, (0.45000000000000001, 0.45000000000000001,
0.45000000000000001)), (251.92499999999995, (0.46862745098039216,
0.46862745098039216, 0.46862745098039216)), (252.96249999999998, (0.4872549019607843,
0.4872549019607843, 0.4872549019607843)), (253.99999999999997, (0.50588235294117645,
0.50588235294117645, 0.50588235294117645)), (255.03749999999997,
(0.52450980392156865, 0.52450980392156865, 0.52450980392156865)),
(256.07499999999999, (0.54313725490196074, 0.54313725490196074,
0.54313725490196074)), (257.11249999999995, (0.56176470588235294,
0.56176470588235294, 0.56176470588235294)), (258.14999999999998,
(0.58039215686274515, 0.58039215686274515, 0.58039215686274515)), (259.1875,
(0.59901960784313724, 0.59901960784313724, 0.59901960784313724)),
(260.22499999999997, (0.61764705882352944, 0.61764705882352944,
0.61764705882352944)), (261.26249999999999, (0.63627450980392153,
0.63627450980392153, 0.63627450980392153)), (262.29999999999995,
(0.65490196078431373, 0.65490196078431373, 0.65490196078431373)),
(263.33749999999998, (0.67352941176470593, 0.67352941176470593,
0.67352941176470593)), (264.375, (0.69215686274509802, 0.69215686274509802,
0.69215686274509802)), (265.41249999999997, (0.71078431372549022,
0.71078431372549022, 0.71078431372549022)), (266.44999999999999,
(0.72941176470588232, 0.72941176470588232, 0.72941176470588232)),
(266.44999999999999, (0.72941176470588232, 0.72941176470588232,
0.72941176470588232)), (267.48750000000001, (0.73946078431372553,
0.73946078431372553, 0.73946078431372553)), (268.52499999999998,
(0.74950980392156863, 0.74950980392156863, 0.74950980392156863)), (269.5625,
(0.75955882352941173, 0.75955882352941173, 0.75955882352941173)),
(270.59999999999997, (0.76960784313725494, 0.76960784313725494,
0.76960784313725494)), (271.63749999999999, (0.77965686274509804,
0.77965686274509804, 0.77965686274509804)), (272.67499999999995,
(0.78970588235294115, 0.78970588235294115, 0.78970588235294115)),
```

```
(273.71249999999998, (0.79975490196078436, 0.79975490196078436,
0.79975490196078436)), (274.75, (0.80980392156862746, 0.80980392156862746,
0.80980392156862746)), (275.78749999999997, (0.81985294117647056,
0.81985294117647056, 0.81985294117647056)), (276.82499999999999,
(0.82990196078431377, 0.82990196078431377, 0.82990196078431377)),
(277.86249999999995, (0.83995098039215688, 0.83995098039215688,
0.83995098039215688)), (278.89999999999998, (0.84999999999999998,
0.84999999999999998, 0.84999999999999998)), (279.93749999999994,
(0.86004901960784319, 0.86004901960784319, 0.86004901960784319)),
(280.97499999999997, (0.87009803921568629, 0.87009803921568629,
0.87009803921568629)), (282.01249999999993, (0.88014705882352939,
0.88014705882352939, 0.88014705882352939)), (283.04999999999995, (0.8901960784313725,
0.8901960784313725, 0.8901960784313725)), (283.04999999999995, (0.8901960784313725,
0.8901960784313725, 0.8901960784313725)), (284.05393939393934, (0.89352346999405818,
0.89352346999405818, 0.89352346999405818)), (285.05787878787874,
(0.89685086155674387, 0.89685086155674387, 0.89685086155674387)),
(286.06181818181813, (0.90017825311942956, 0.90017825311942956,
0.90017825311942956)), (287.06575757575752, (0.90350564468211536,
0.90350564468211536, 0.90350564468211536)), (288.06969696969691,
(0.90683303624480094, 0.90683303624480094, 0.90683303624480094)), (289.0736363636363,
(0.91016042780748663, 0.91016042780748663, 0.91016042780748663)),
(290.07757575757569, (0.91348781937017232, 0.91348781937017232,
0.91348781937017232)), (291.08151515151508, (0.916815210932858, 0.916815210932858,
0.916815210932858)), (292.08545454545452, (0.92014260249554358, 0.92014260249554358,
0.92014260249554358)), (293.08939393939391, (0.92346999405822938,
0.92346999405822938, 0.92346999405822938)), (294.09333333333331,
(0.92679738562091507, 0.92679738562091507, 0.92679738562091507)), (295.0972727272727,
(0.93012477718360076, 0.93012477718360076, 0.93012477718360076)),
(296.10121212121209, (0.93345216874628645, 0.93345216874628645,
0.93345216874628645)), (297.10515151515148, (0.93677956030897203,
0.93677956030897203, 0.93677956030897203)), (298.10909090909087,
(0.94010695187165771, 0.94010695187165771, 0.94010695187165771)),
(299.11303030303026, (0.9434343434343434, 0.9434343434343434, 0.9434343434343434)),
(300.11696969696965, (0.9467617349970292, 0.9467617349970292, 0.9467617349970292)),
(301.12090909090904, (0.95008912655971478, 0.95008912655971478,
0.95008912655971478)), (302.12484848484843, (0.95341651812240047,
0.95341651812240047, 0.95341651812240047)), (303.12878787878782,
(0.95674390968508616, 0.95674390968508616, 0.95674390968508616)),
(304.13272727272721, (0.96007130124777185, 0.96007130124777185,
0.96007130124777185)), (305.1366666666666, (0.96339869281045742, 0.96339869281045742,
0.96339869281045742)), (306.14060606060599, (0.96672608437314311,
0.96672608437314311, 0.96672608437314311)), (307.14454545454544,
(0.97005347593582891, 0.97005347593582891, 0.97005347593582891)),
(308.14848484848483, (0.9733808674985146, 0.9733808674985146, 0.9733808674985146)),
(309.15242424242422, (0.97670825906120029, 0.97670825906120029,
0.97670825906120029)), (310.15636363636361, (0.98003565062388587,
0.98003565062388587, 0.98003565062388587)), (311.160303030303, (0.98336304218657156,
0.98336304218657156, 0.98336304218657156)), (312.16424242424239,
(0.98669043374925725, 0.98669043374925725, 0.98669043374925725)),
(313.16818181818178, (0.99001782531194304, 0.99001782531194304,
0.99001782531194304)), (314.17212121212117, (0.99334521687462862,
0.99334521687462862, 0.99334521687462862)), (315.17606060606056,
(0.99667260843731431, 0.99667260843731431, 0.99667260843731431)),
(316.17999999999995, (1.0, 1.0, 1.0)))

        cmIR.reverse() # reverse lut

        ########################
        # for IR 10.8 colored

        cm3 = trollimage.colormap.Colormap((-60.0+ 273.15, (111./255.,0.,198/255.)),
```

```
                                    (-50.0+ 273.15, (0.60784314, 0.33333333,
0.77254902)),
                                    (-40.0+ 273.15, (0.90980392, 0.03921569,
0.90588235)),
                                    (-30.0+ 273.15, (0.66666667, 0.04313725,
0.0627451)),
                                    (-20.0+ 273.15, (0.79215686, 0.00392157,
0.02745098)),
                                    (-18.0+ 273.15, (1., 0.00392157, 0.00392157)),
                                    (-16.0+ 273.15, (1., 0.55686275, 0.01568627)),
                                    (-14.0+ 273.15, (0.90196078, 0.74901961,
0.00784314)),
                                    (-12.0+ 273.15, (0.96078431, 0.99215686,
0.03529412)),
                                    (-10.0+ 273.15, (0.03529412, 0.55294118,
0.01568627)),
                                    (-8.0+ 273.15, (0.02352941, 0.76862745, 0.)),
                                    (-6.0+ 273.15, (0.0627451, 0.94901961,
0.03921569)),
                                    (-4.0+ 273.15, (0.08627451, 0., 0.80392157)),
                                    (-2.0+ 273.15, (0.04705882, 0.57647059, 1.)),
                                    (-0.00001+ 273.15, (0.08235294, 0.87843137,
0.93333333)))

        cm2 = trollimage.colormap.Colormap((-70.0+ 273.15, (111./255.,0.,198/255.)),
                            (-65.0+ 273.15, (0.60784314, 0.33333333, 0.77254902)),
                            (-60.0+ 273.15, (0.90980392, 0.03921569, 0.90588235)),
                            (-55.0+ 273.15, (0.66666667, 0.04313725, 0.0627451)),
                            (-50.0+ 273.15, (0.79215686, 0.00392157, 0.02745098)),
                            (-45.0+ 273.15, (1., 0.00392157, 0.00392157)),
                            (-40.0+ 273.15, (1., 0.55686275, 0.01568627)),
                            (-35.0+ 273.15, (0.90196078, 0.74901961, 0.00784314)),
                            (-30.0+ 273.15, (0.96078431, 0.99215686, 0.03529412)),
                            (-25.0+ 273.15, (0.03529412, 0.55294118, 0.01568627)),
                            (-20.0+ 273.15, (0.02352941, 0.76862745, 0.)),
                            (-15.0+ 273.15, (0.0627451, 0.94901961, 0.03921569)),
                            (-10.0+ 273.15, (0.08627451, 0., 0.80392157)),
                            (-5.0+ 273.15, (0.04705882, 0.57647059, 1.)),
                            (-0.00001+ 273.15, (0.08235294, 0.87843137, 0.93333333)))

        cm1 = trollimage.colormap.Colormap((273.15, (0.0, 0.0, 0.0)),
(274.14999999999998, (0.027450980392156862, 0.027450980392156862,
0.027450980392156862)), (275.14999999999998, (0.054901960784313725,
0.054901960784313725, 0.054901960784313725)), (276.14999999999998,
(0.082352941176470587, 0.082352941176470587, 0.082352941176470587)),
(277.14999999999998, (0.10980392156862745, 0.10980392156862745,
0.10980392156862745)), (278.14999999999998, (0.13725490196078433,
0.13725490196078433, 0.13725490196078433)), (279.14999999999998,
(0.16470588235294117, 0.16470588235294117, 0.16470588235294117)),
(280.14999999999998, (0.19215686274509805, 0.19215686274509805,
0.19215686274509805)), (281.14999999999998, (0.2196078431372549, 0.2196078431372549,
0.2196078431372549)), (281.14999999999998, (0.2196078431372549, 0.2196078431372549,
0.2196078431372549)), (282.14999999999998, (0.24607843137254901, 0.24607843137254901,
0.24607843137254901)), (283.14999999999998, (0.27254901960784311,
0.27254901960784311, 0.27254901960784311)), (284.14999999999998,
(0.29901960784313725, 0.29901960784313725, 0.29901960784313725)),
(285.14999999999998, (0.32549019607843138, 0.32549019607843138,
0.32549019607843138)), (286.14999999999998, (0.35196078431372552,
0.35196078431372552, 0.35196078431372552)), (287.14999999999998, (0.3784313725490196,
0.3784313725490196, 0.3784313725490196)), (288.14999999999998, (0.40490196078431373,
0.40490196078431373, 0.40490196078431373)), (289.14999999999998,
```

```
(0.43137254901960786, 0.43137254901960786, 0.43137254901960786)),
(289.14999999999998, (0.43137254901960786, 0.43137254901960786,
0.43137254901960786)), (290.14999999999998, (0.46862745098039216,
0.46862745098039216, 0.46862745098039216)), (291.14999999999998,
(0.50588235294117645, 0.50588235294117645, 0.50588235294117645)),
(292.14999999999998, (0.54313725490196074, 0.54313725490196074,
0.54313725490196074)), (293.14999999999998, (0.58039215686274515,
0.58039215686274515, 0.58039215686274515)), (294.14999999999998,
(0.61764705882352944, 0.61764705882352944, 0.61764705882352944)),
(295.14999999999998, (0.65490196078431373, 0.65490196078431373,
0.65490196078431373)), (296.14999999999998, (0.69215686274509802,
0.69215686274509802, 0.69215686274509802)), (297.14999999999998,
(0.72941176470588232, 0.72941176470588232, 0.72941176470588232)),
(297.14999999999998, (0.72941176470588232, 0.72941176470588232,
0.72941176470588232)), (298.14999999999998, (0.74950980392156863,
0.74950980392156863, 0.74950980392156863)), (299.14999999999998,
(0.76960784313725494, 0.76960784313725494, 0.76960784313725494)),
(300.14999999999998, (0.78970588235294115, 0.78970588235294115,
0.78970588235294115)), (301.14999999999998, (0.80980392156862746,
0.80980392156862746, 0.80980392156862746)), (302.14999999999998,
(0.82990196078431377, 0.82990196078431377, 0.82990196078431377)),
(303.14999999999998, (0.84999999999999998, 0.84999999999999998,
0.84999999999999998)), (304.14999999999998, (0.87009803921568629,
0.87009803921568629, 0.87009803921568629)), (305.14999999999998, (0.8901960784313725,
0.8901960784313725, 0.8901960784313725)), (305.14999999999998, (0.8901960784313725,
0.8901960784313725, 0.8901960784313725)), (306.14999999999998, (0.90392156862745099,
0.90392156862745099, 0.90392156862745099)), (307.14999999999998,
(0.91764705882352937, 0.91764705882352937, 0.91764705882352937)),
(308.14999999999998, (0.93137254901960786, 0.93137254901960786,
0.93137254901960786)), (309.14999999999998, (0.94509803921568625,
0.94509803921568625, 0.94509803921568625)), (310.14999999999998,
(0.95882352941176474, 0.95882352941176474, 0.95882352941176474)),
(311.14999999999998, (0.97254901960784312, 0.97254901960784312,
0.97254901960784312)), (312.14999999999998, (0.98627450980392162,
0.98627450980392162, 0.98627450980392162)), (313.14999999999998, (1.0, 1.0, 1.0)))


        ################

        cm1.reverse() # reverse lut

        cmIRcoloredSPC = cm3+cm1

        ################################################

        #### parallel computing
        pool = multiprocessing.Pool(multiprocessing.cpu_count()-1)
        pool.map(plotSatImage, range(11))
        pool.close()
        pool.join()
        print('done')
        ####

        end = time.time()
        print('PROGRAM finished after %2.1f minutes' % ((end - start)/60.))
```