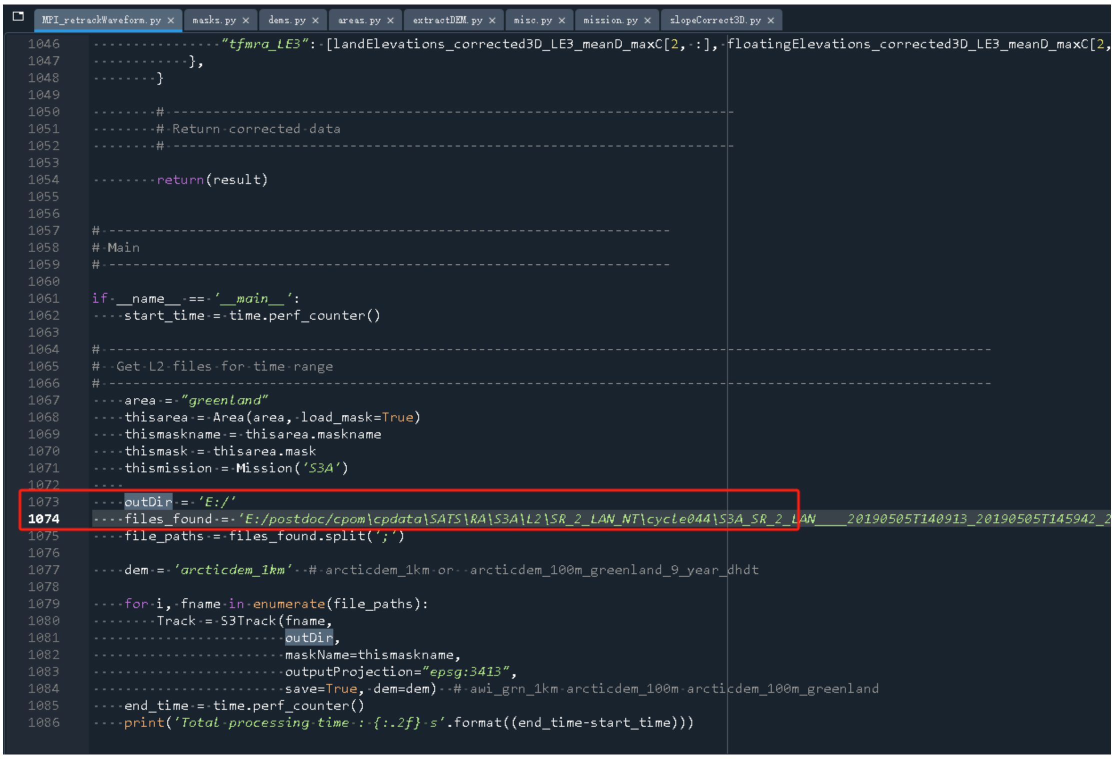
MultiPeak Ice (MPI) is a new processing approach designed specifically for complex ice surfaces, where the majority of ice mass imbalance occurs, and is able to reliably retrieve multiple elevation measurements from a single altimetry echo. MPI_retrackWaveform is the main function. Download relevant data (see below), revise the data location as per your directory, and add the code directory to your Python development environment (i.e., Spdyer).

1. Download Sentinel-3A data from Copernicus Data Space Ecosystem at https://dataspace.copernicus.eu/. One example of S3A data is below:

   /cycle044/S3A_SR_2_LAN____20190505T140913_20190505T145942_20190530T202816_3029_044_210______LN3_O_NT_003.SEN3/enhanced_measurement.nc

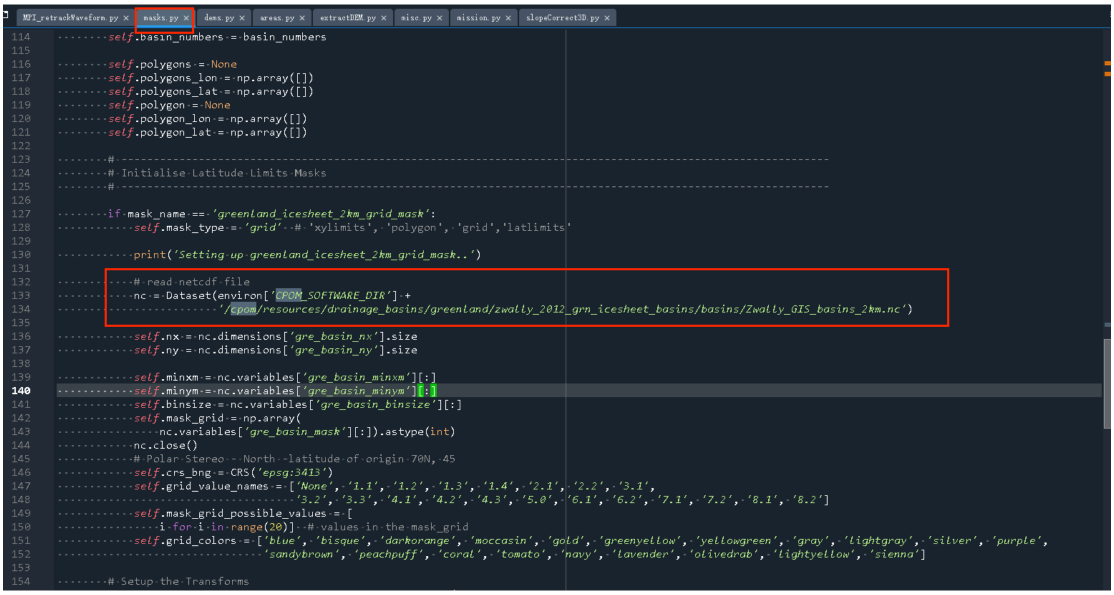
   Change the directory of S3A to your own folder in MPI_retrackWaveform.py (Lines 1073-1074) (Figure 1). For example, download and put your data in E:/ and Line 1073 would become **outDir = 'E:/'**
   Line 1074 will be: **files_found = 'E:/S3A_SR_2_LAN____20190505T140913_20190505T145942_20190530T202816_3029_044_210______LN3_O_NT_003.SEN3/enhanced_measurement.nc'**



```
1046                 "tfmra_LE3": [landElevations_corrected3D_LE3_meanD_maxC[2, :], floatingElevations_corrected3D_LE3_meanD_maxC[2,
1047             },
1048         }
1049 
1050         # ----------------------------------------------------------------------
1051         # Return corrected data
1052         # ----------------------------------------------------------------------
1053 
1054         return(result)
1055 
1056 
1057 # ----------------------------------------------------------------------
1058 # Main
1059 # ----------------------------------------------------------------------
1060 
1061 if __name__ == '__main__':
1062     start_time = time.perf_counter()
1063 
1064 # ------------------------------------------------------------------------------------------------------------
1065 #  Get L2 files for time range
1066 # ------------------------------------------------------------------------------------------------------------
1067     area = "greenland"
1068     thisarea = Area(area, load_mask=True)
1069     thismaskname = thisarea.maskname
1070     thismask = thisarea.mask
1071     thismission = Mission('S3A')
1072     
1073     outDir = 'E:/'
1074     files_found = 'E:/postdoc/cpom\cpdata\SATS\RA\S3A\L2\SR_2_LAN_NT\cycle044\S3A_SR_2_LAN____20190505T140913_20190505T145942_2
1075     file_paths = files_found.split(';')
1076 
1077     dem = 'arcticdem_1km'  # arcticdem_1km or  arcticdem_100m_greenland_9_year_dhdt
1078 
1079     for i, fname in enumerate(file_paths):
1080         Track = S3Track(fname,
1081                         outDir,
1082                         maskName=thismaskname,
1083                         outputProjection="epsg:3413",
1084                         save=True, dem=dem)  # awi_grn_1km arcticdem_100m arcticdem_100m_greenland
1085     end_time = time.perf_counter()
1086     print('Total processing time : {:.2f} s'.format((end_time-start_time)))
```

Figure 1 File directory in the main function.

2. Download Zwally_GIS_basins_2km.nc from the data folder in the repository and change it to your own direction at Line 133 in mask.py



```
        self.basin_numbers = basin_numbers

        self.polygons = None
        self.polygons_lon = np.array([])
        self.polygons_lat = np.array([])
        self.polygon = None
        self.polygon_lon = np.array([])
        self.polygon_lat = np.array([])

        # --------------------------------------------------------------------------------------------------------
        # Initialise Latitude Limits Masks
        # --------------------------------------------------------------------------------------------------------

        if mask_name == 'greenland_icesheet_2km_grid_mask':
            self.mask_type = 'grid'  # 'xylimits', 'polygon', 'grid','latlimits'

            print('Setting up greenland_icesheet_2km_grid_mask..')

            # read netcdf file
            nc = Dataset(environ['CPOM_SOFTWARE_DIR'] +
                         '/cpom/resources/drainage_basins/greenland/zwally_2012_grn_icesheet_basins/basins/Zwally_GIS_basins_2km.nc')

            self.nx = nc.dimensions['gre_basin_nx'].size
            self.ny = nc.dimensions['gre_basin_ny'].size

            self.minxm = nc.variables['gre_basin_minxm'][:]
            self.minym = nc.variables['gre_basin_minym'][:]
            self.binsize = nc.variables['gre_basin_binsize'][:]
            self.mask_grid = np.array(
                nc.variables['gre_basin_mask'][:]).astype(int)
            nc.close()
            # Polar Stereo - North -latitude of origin 70N, 45
            self.crs_bng = CRS('epsg:3413')
            self.grid_value_names = ['None', '1.1', '1.2', '1.3', '1.4', '2.1', '2.2', '3.1',
                                    '3.2', '3.3', '4.1', '4.2', '4.3', '5.0', '6.1', '6.2', '7.1', '7.2', '8.1', '8.2']
            self.mask_grid_possible_values = [
                i for i in range(20)]  # values in the mask_grid
            self.grid_colors = ['blue', 'bisque', 'darkorange', 'moccasin', 'gold', 'greenyellow', 'yellowgreen', 'gray', 'lightgray', 'silver', 'purple',
                                'sandybrown', 'peachpuff', 'coral', 'tomato', 'navy', 'lavender', 'olivedrab', 'lightyellow', 'sienna']

        # Setup the Transforms
```

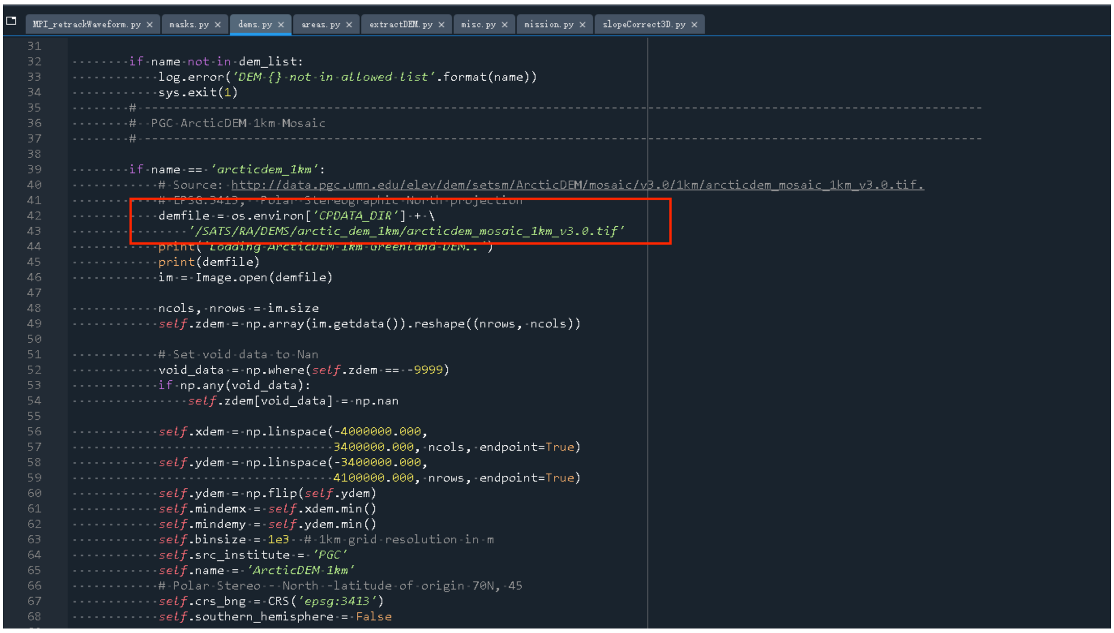
Figure 2 Mask directory in mask.py

3. Download DEMs

MPI now supports two DEMs, choose either one of them for your research and you can add your own DEMs by revising the dem.py

(a) 1km DEM downloaded from: http://data.pgc.umn.edu/elev/dem/setsm/ArcticDEM/mosaic/v3.0/1km/arcticdem_mosaic_1km_v3.0.tif.

Download the dem and rename it as 'arcticdem_mosaic_1km_v3.0.tif'. Make sure to change the data directory to your own folder at Line 42 in dem.py (Figure 3)

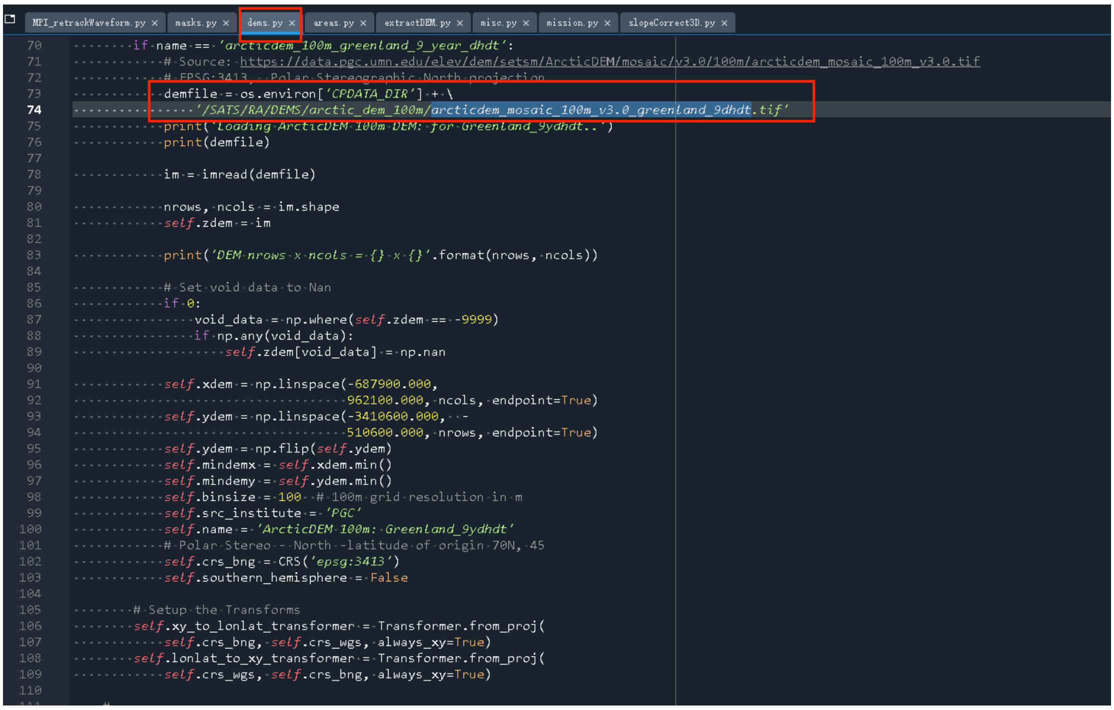


```
31
32          if name not in dem_list:
33              log.error('DEM {} not in allowed list'.format(name))
34              sys.exit(1)
35          # -----------------------------------------------------------------------------------------------------
36          #  PGC ArcticDEM 1km Mosaic
37          # -----------------------------------------------------------------------------------------------------
38
39          if name == 'arcticdem_1km':
40              # Source: http://data.pgc.umn.edu/elev/dem/setsm/ArcticDEM/mosaic/v3.0/1km/arcticdem_mosaic_1km_v3.0.tif.
41              # EPSG:3413,  Polar Stereographic North projection
42              demfile = os.environ['CPDATA_DIR'] + \
43                  '/SATS/RA/DEMS/arctic_dem_1km/arcticdem_mosaic_1km_v3.0.tif'
44              print('Loading ArcticDEM 1km Greenland DEM..')
45              print(demfile)
46              im = Image.open(demfile)
47
48              ncols, nrows = im.size
49              self.zdem = np.array(im.getdata()).reshape((nrows, ncols))
50
51              # Set void data to Nan
52              void_data = np.where(self.zdem == -9999)
53              if np.any(void_data):
54                  self.zdem[void_data] = np.nan
55
56              self.xdem = np.linspace(-4000000.000,
57                                      3400000.000, ncols, endpoint=True)
58              self.ydem = np.linspace(-3400000.000,
59                                      4100000.000, nrows, endpoint=True)
60              self.ydem = np.flip(self.ydem)
61              self.mindemx = self.xdem.min()
62              self.mindemy = self.ydem.min()
63              self.binsize = 1e3  # 1km grid resolution in m
64              self.src_institute = 'PGC'
65              self.name = 'ArcticDEM 1km'
66              # Polar Stereo - North -latitude of origin 70N, 45
67              self.crs_bng = CRS('epsg:3413')
68              self.southern_hemisphere = False
```

Figure 3 DEM directory for 1 km resolution in dem.py

(b) 100m DEM downloaded from: https://data.pgc.umn.edu/elev/dem/setsm/ArcticDEM/mosaic/v3.0/100m/arcticdem_mosaic_100m_v3.0.tif

Download the DEM, apply a 9-year linear dh/dt to the DEM(optional), rename it as ‘arcticdem_mosaic_100m_v3.0_greenland_9dhdt.tif’. Make sure to change the data directory to your own folder at Line 74 in dem.py (Figure 4)



```
 70         if name == 'arcticdem_100m_greenland_9_year_dhdt':
 71             # Source: https://data.pgc.umn.edu/elev/dem/setsm/ArcticDEM/mosaic/v3.0/100m/arcticdem_mosaic_100m_v3.0.tif
 72             # EPSG:3413, - Polar Stereographic North projection
 73             demfile = os.environ['CPDATA_DIR'] + \
 74                   '/SATS/RA/DEMS/arctic_dem_100m/arcticdem_mosaic_100m_v3.0_greenland_9dhdt.tif'
 75             print('Loading ArcticDEM 100m DEM: for Greenland_9ydhdt..')
 76             print(demfile)
 77
 78             im = imread(demfile)
 79
 80             nrows, ncols = im.shape
 81             self.zdem = im
 82
 83             print('DEM nrows x ncols = {} x {}'.format(nrows, ncols))
 84
 85             # Set void data to Nan
 86             if 0:
 87                 void_data = np.where(self.zdem == -9999)
 88                 if np.any(void_data):
 89                     self.zdem[void_data] = np.nan
 90
 91             self.xdem = np.linspace(-687900.000,
 92                                     962100.000, ncols, endpoint=True)
 93             self.ydem = np.linspace(-3410600.000, -
 94                                     510600.000, nrows, endpoint=True)
 95             self.ydem = np.flip(self.ydem)
 96             self.mindemx = self.xdem.min()
 97             self.mindemy = self.ydem.min()
 98             self.binsize = 100  # 100m grid resolution in m
 99             self.src_institute = 'PGC'
100             self.name = 'ArcticDEM 100m: Greenland_9ydhdt'
101             # Polar Stereo - North -latitude of origin 70N, 45
102             self.crs_bng = CRS('epsg:3413')
103             self.southern_hemisphere = False
104
105         # Setup the Transforms
106         self.xy_to_lonlat_transformer = Transformer.from_proj(
107             self.crs_bng, self.crs_wgs, always_xy=True)
108         self.lonlat_to_xy_transformer = Transformer.from_proj(
109             self.crs_wgs, self.crs_bng, always_xy=True)
110
```

Figure 4 DEM directory for 100 m resolution in dem.py