MSGCPP downloading guide

Index

1. Introduction

MSGCPP uses measurements from the METEOSAT second generation satellite (MSG) and a cloud physical properties calculation algorithm (CPP) in order to obtain irradiance vaules, percipitation rate, cloud top temperature, etc.

We can download irradiance values for a desired area and time using the following url in the browser:

http://msgcpp-ogc-archive.knmi.nl/msgar.cgi?%20&service=wcs&version=1.0.0&request=getcoverage&coverage=surface_downwelling_shortwave_flux_in_air&FORMAT=AAIGRID&CRS=EPSG%3A4326&BBOX=-0.06,39.99,-0.06,39.99&WIDTH=1&HEIGHT=1&time=2016-8-1T13:45:00Z

1.1. Storage

The first parameter to modify is the word archive. There are two optional words to introduce in that position that correspond to the two different data storage systems included in this data base. The first one is updated every 15 minutes with data belonging to the last 7 days (it can be accessed by replacing the word archive with realtime). The second one has data with 15 minute time steps for the last three years and it is updated once a day (using the word archive).

1.2. Physical property

The next part to modify is the "coverage", which describes the desired physical property to download. In the example, surface_downwelling_shortwave_flux_in_air corresponds to the global horizontal irradiance. It is possible to download any of the properties shown in the msgcpp web viewer. To do so, we must get the property name in the web viewer and look for its name in this web. For example, to download the precipitation rate property from the viewer, the name obtained from the web is lwe_precipitation_rate and with this name, data can be downloaded.

1.3. Coordinates

The numbers following BBOX are the coordinates of the desired location, which is limited by a rectangle defined by two points. The first two numbers are the smallest longitude and latitude, and the two following numbers are the bigger ones. It is important to write the smaller coordinates first to avoid downloading wrong data. In the example the same coordinates are used twice (-0.06 , 39.99) because the desired data is only a point (at the UJI).

The resolution of the image is limited by the size of each pixel and changes with the latitude. The minimum size of a pixel is approximately 0.03° for both latitude and longitude. This resolution equals to a 3 km^2 surface.

1.4. Matrix size

With the parameters WIDTH and HEIGHT the size of the downloaded data matrix is selected. In the example it is $1x1$ because the downloaded data is for a point. However, if a greater area is selected, it is important to have enough elements in the matrix to cover the whole area.

As the size of each pixel is 0.03° approximately, the size of the matrix can be obtained using the difference between the final and the initial coordinates and dividing it by 0.03. Obtaining the width parameter from the longitude and the height from the latitude.

As an example, when downloading a $3x3$ area from the map into a $3x3$ matrix (the dimensions match) we obtain this.

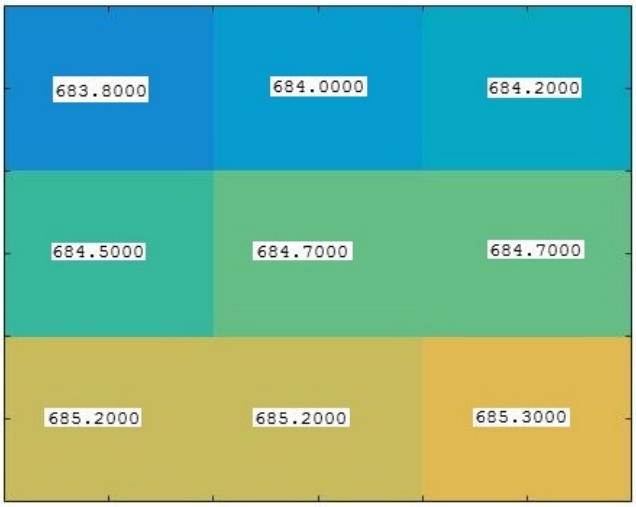
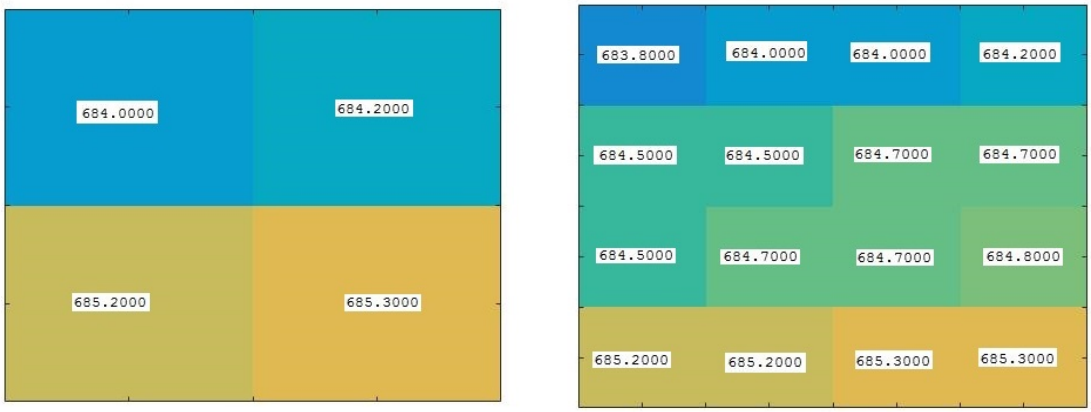


Figure 1: Matrix with the same pixels than the map.

However, if for the same area, we download a different matrix size, some pixels are omitted (if the matrix size is smaller) or duplicated and interpolated (if the matrix size is bigger).



(a) Smaller matrix (b) Bigger matrix

Figure 2: Different matrix sizes

It is possible to use RESX=0.03 and RESY=0.03 instead of width and height. This creates a pixel every 0.03° of longitude and latitude. Notice that, as pixel size changes with latitude, different matrix sizes are still downloaded, but we avoid the size calculation.

1.5. Date and time

The last parameter starts with the keyword time. It contains the date in a Year-Month-Day format, after this, there is a T and the time (hh:mm:00Z). As data is available at 15 minutes time steps, the seconds value is always 00 and the minutes can only be 00, 15, 30 or 45. In the example url the downloaded data corresponds to August 1, 2016 at 13:45.

2. Download of 1 day

Either for a point or for an area, to download the values for a whole day, the url must be completed with the desired parameters, and then time has to change from 00:00:00 to 23:45:00. This is made by creating strings with all the parameters and putting them together.

The first string ($str1$) specifies that values will be downloaded from the archive and that the physical property is the global irradiance.

```
str1='http://msgcpp-ogc-archive.knmi.nl/msgar.cgi? &service=wcs&version
    =1.0.0&request=getcoverage&coverage=
    surface_downwelling_shortwave_flux_in_air&FORMAT=AAIGRID&CRS=EPSG%3A4326
    &';
```

Then the coordinates are introduced to generate $str2$.

```
% Longitude y latitude  STR2 --> BBOX=-0.06,39.99,-0.06,39.99&
longitude1=-0.06;
longitude2=-0.06;
latitude1=39.99;
latitude2=39.99;
% Assure that the coordinates are in the correct order
longitude11=min(longitude1,longitude2);
longitude22=max(longitude1,longitude2);
latitude11=min(latitude1,latitude2);
latitude22=max(latitude1,latitude2);

longitude1=num2str(longitude11);
longitude2=num2str(longitude22);
latitude1=num2str(latitude11);
latitude2=num2str(latitude22);

str2=strcat('BBOX=',longitude1,',',latitude1,',',longitude2,',',latitude2,'
    &');
```

*Str*3 contains the matrix size.

```
% Width and height of the image  STR3 --> WIDTH=1&HEIGHT=1&
width=1;
height=1;

width=num2str(width);
height=num2str(height);

str3=strcat('WIDTH=',width,'&HEIGHT=',height,'&');
```

Then the date is introduced in *str*4.

```
% Date STR4 --> time=2016-08-25T
year=2016;
day=1;
month=8;

year=num2str(year);
day=num2str(day);
month=num2str(month);

str4=strcat('time=',year,'-',month,'-',day,'T');
```

Finally all the times of the day are introduced one by one to save the 96 values (every 15 minutes).

```
values=[];
position=1;
% Time  STR5 -->12:45:00Z
for j=0:1:23
    for i=0:15:59
        hour=num2str(j);
        minute=num2str(i);

        str5=strcat(hour,':',minute,':00Z');
        address=strcat(str1,str2,str3,str4,str5);
```

With the address obtained, content is downloaded using *urlread*.

```
web=urlread(address,'Timeout',15);
data=strsplit(web);
```

In some cases, the connection can fail, or data is not available. When this happens, the downloaded data starts with the word *Invalid*. In this cases the program tries to download the data again until the *urlwrong* number is reached. When data is successfully downloaded, the program continues, otherwise, the variable *err* is set to 1 to know that something was wrong and it shows in the screen the time that produced the error.

```
urlwrongmax=15;
urlwrong=0;
err=0;
while (strcmp(data{1},'Invalid')&& err==0) %Try to download data
again if something fails
    web=urlread(address);
    data=strsplit(web);
    urlwrong=urlwrong+1;
    if urlwrong==urlwrongmax
        urlwrongmax
        strcat(str5,' ERROR')
        err=1;
    end
end
```

The downloaded data for each loop iteration is the following.

```
ncols         1
nrows         1
xllcorner     -0.060000000000
yllcorner     39.990000000000
cellsize      0.000000000000
NODATA_value -0.10000000149011611938
 703.4000244140625
```

Figure 3: Format of the downloaded data.

There is a number ($NODATA\_value$) that appears in the matrix positions which do not have a value (usually because of the satellite positioning during night time). After this value, the matrix containing the desired data can be found. If there were no errors during the download, the $NODATA\_value$ is obtained and the number is saved, after this, the matrix is read, replacing the positions containing the $NODATA\_value$ with zeros.

```
    if err==0 %Data correctly downloaded
        nodatapos=1;
        while ~strcmp(data{nodatapos},'NODATA_value')
            nodatapos=nodatapos+1;
        end
        nodatapos=nodatapos+1;%Position of the nodata_value number

        NODATA_value=str2num(data{nodatapos});
        rows=str2num(data{4});
        columns=str2num(data{2});


        matrix=zeros(rows,columns);

        counter=1;
        for r=1:rows
            for c=1:columns
                matrix(r,c)=str2num(data{nodatapos+counter});
                if matrix(r,c)==NODATA_value
                    matrix(r,c)=0;
                end
                counter=counter+1;
            end
        end
```

If there was an error during the download, the matrix is completed with *failvalue* (-100 in this case).

```
    elseif err==1 %If data can't be downloaded, fills the matrix with
%the failvalue
        rows=str2num(height);
        columns=str2num(width);

        for r=1:rows
            for c=1:columns
                matrix(r,c)=failvalue;
            end
        end

    end
```

Finally the matrix is saved in an array, which third dimension corresponds with the time of the day (1 to 96).

```
        values(:,:,position)=matrix;
        position=position+1;
    end
end
```

3. Results

Attached to this pdf you can find the MATLAB file containing the example code explained. When the complete day is downloaded, it can be visualized using the following commands.

```
irad(:,1)=values(1,1,:);
plot(irad)
```

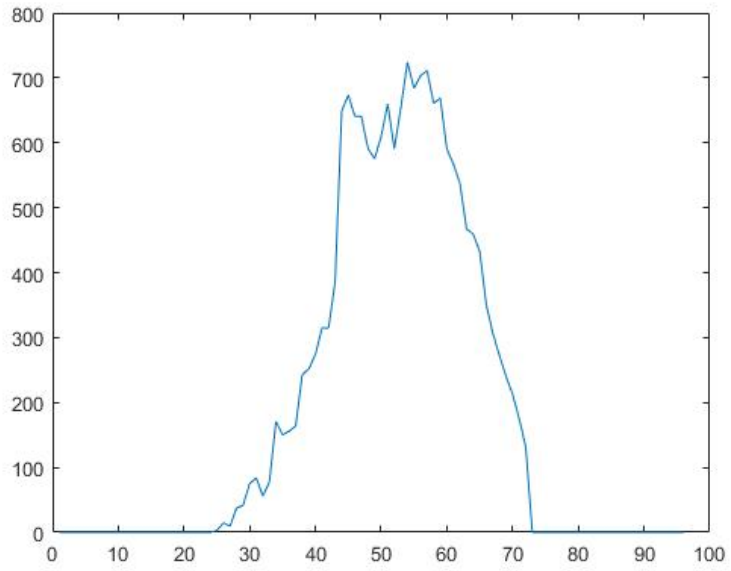


Figure 4: Irradiance at the UJI, August 1st, 2016.

4. References

WCS Primer (Easy as buying an ice cream).

MSGCPP data access.