Version 1.0
October 2016


RESEARCH
PROGRAM ON
Water, Land and
Ecosystems
CGIAR


United Nations
Educational, Scientific and
Cultural Organization
World Water
Assessment Programme


FIAT PANIS


UNESCO-IHE
Institute for Water Education


IWMI
International
Water Management
Institute

Table of Contents

How to Install Anaconda (python) 1
How to install all the necessary modules 3
Method 1: 3
Method 2: 4
Method 3: 4
Method 4: 4
Install GDAL version 1.11.1 5
How to install necessary executables 7
Introducing Water Accounting plus python tool 13
Water Accounting plus tool: Collect functions 15
General 15
CHIRPS 15
GLDAS 15
CFSR 17
DEM 18
ALEXI 19
TRMM 19
MOD13 20
MOD15 20
MOD16 21
MOD17 21
Water Accounting plus tool: Products functions 23
General 23
ETref 23
ETens 25
Water Accounting plus tool: Sheets functions 27
General 27
create_sheet1: 27
create_sheet2: 28

How to Install Anaconda (python)

There are several packages available, which include python. The WA+ team recommends to download the python package of Anaconda. The great advantage of using the standard Anaconda package is that most of the commonly used modules are included in the package. Alternatively, these modules can be installed separately by the user if a different package than Anaconda is preferred.

Anaconda can be downloaded from: http://continuum.io/downloads. It is recommended to download the 64 bits version, because this will increase the calculation capacity enormously. However, be sure that your computer/laptop is a 64 bits computer/laptop.

The WA+ python codes are made for python version 2.7. It is therefore necessary to download this version of python for running WA+ tools. Major changes are made to the python codes and functions if you compare 2.7 with 3+ versions. It is therefore not possible to run WA+ code in python 3+ versions without making some changes to the code.

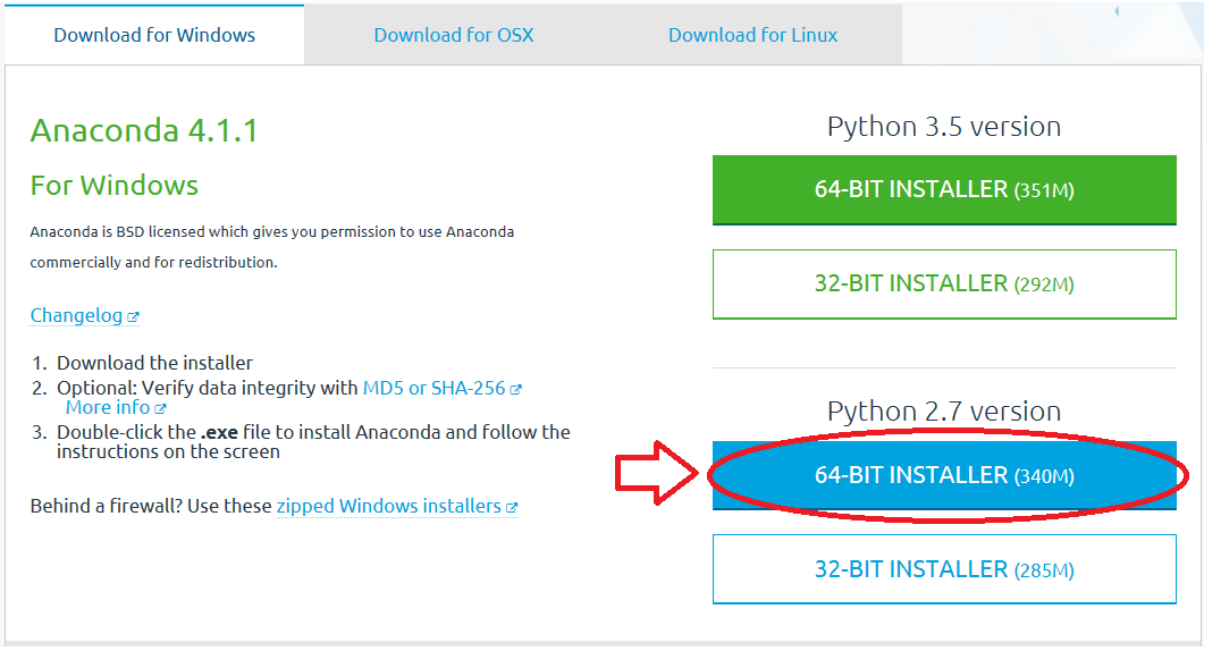


Figure 1: The version that the WA+ team uses for setting up the WA+ python module (http://continuum.io/downloads)

After downloading Anaconda you can run the installation of Anaconda. This package also includes Spyder, which is the IDE (Integrated Development Environment). This is a layout for writing and running python scripts.

How to install all the necessary modules

Modules are tools that can be imported into your python code. They usually contain stand-alone functions, which can be used within your own python code.

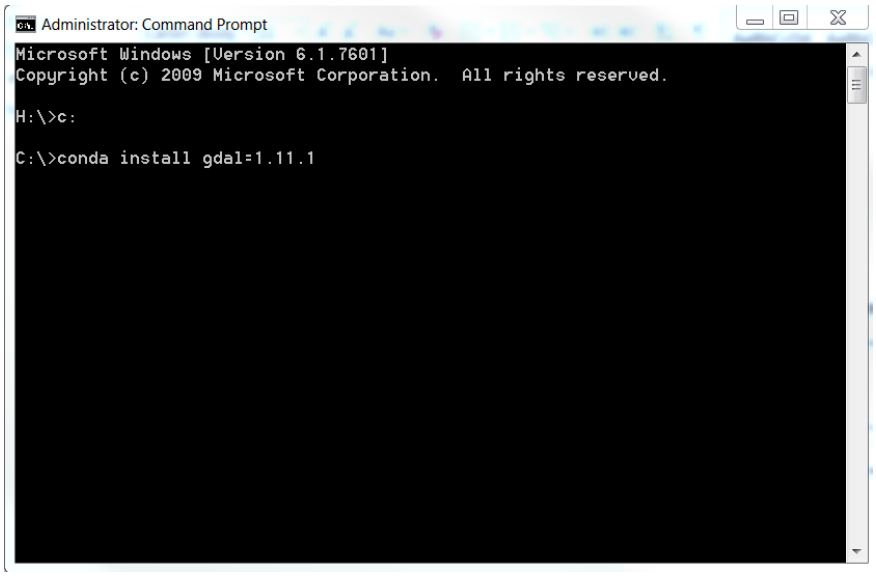
In order to run all the python codes you need the following modules:

- calendar
- datetime
- ftplib
- gdal 1.11.1
- gzip
- joblib
- math
- netCDF4
- numpy
- os
- osgeo
- osr
- pandas
- pycurl
- re
- requests
- scipy
- shutil
- struct
- subprocess
- sys
- tempfile
- urllib
- Wand
- zipfile

A list of the modules installed on your computer/laptop can be accessed by opening the command prompt and typing: “pip freeze”. The installed python modules, including the version, will be listed.

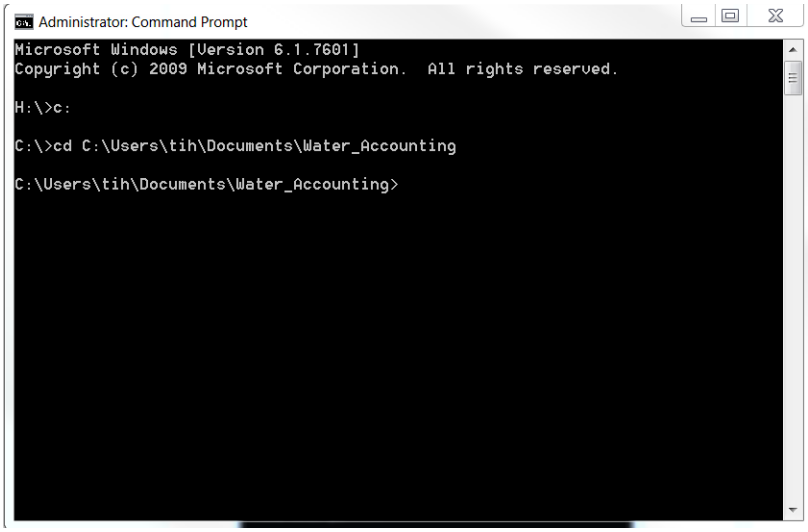
Any missing modules can be installed by using one of the following four methods.

Method 1:
If a package is missing, you can search in the anaconda library by starting the command prompt and write: `conda install ... module name ...`
This will only work when you have Anaconda installed.
Below is an example for installing gdal version 1.11.1 in command prompt.



Method 2:
You can also use the following command in the command prompt to install a module:
`pip install ... module name ...`

Method 3:
You can also install a module by using a .whl file (wheel-file).
The wheels can be downloaded here: http://pythonwheels.com/
After downloading the module whl file, open command prompt and change the directory to the directory where the .whl file is located. You can change the directory in command prompt by using the cd command, as shown in the example below.
After changing the directory, use the following command to install the module.
`pip install ... whl ...`



Method 4:
If a module has a complete folder with a setup.py script inside, you can make use of this method.
Open the command prompt and change the directory to the path where the setup.py is located. Then you can install the module by using the following command within the command prompt:
`python setup.py install`

Install GDAL version 1.11.1

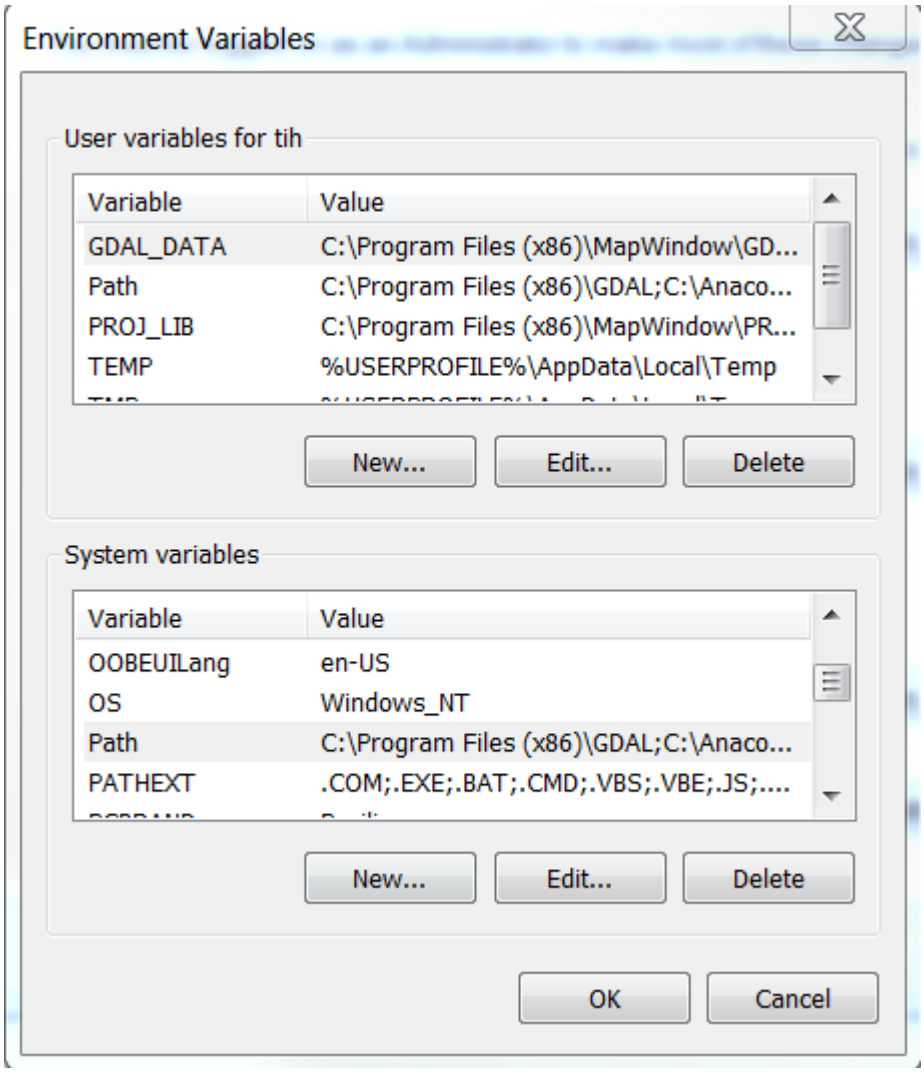
Most of the modules mentioned in the previous section are already installed when you install Anaconda. One of the exceptions is the gdal module. It is important that you download version 1.11.1 of gdal. You can do this by using the command shown in method 1 of the previous section. If a newer version of gdal is installed some functions of the WA+ will not run properly.
The disadvantage is that installing this module also downgrades the numpy module. Therefore, you need to reinstall the numpy module (`conda install numpy`). If you do not do this, opening Spyder can become problematic.

How to install necessary executables

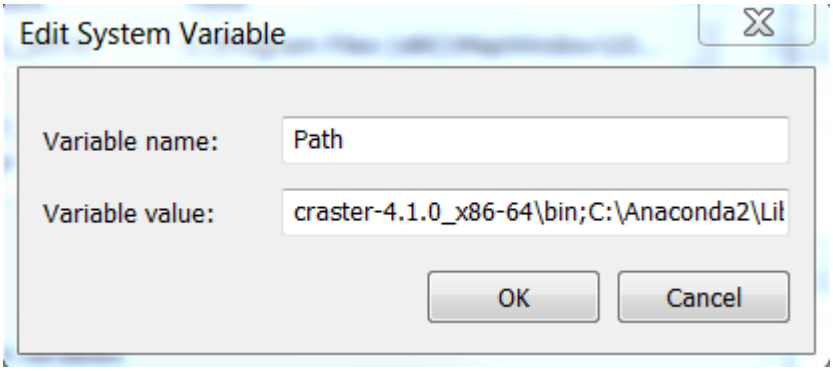
There are some necessary executables for running the WA+ scrips, which are:

- gdal_translate.exe (is installed within QGIS)
- gdalwarp.exe (is installed within QGIS)
- gdalbuildvrt.exe (is installed within QGIS)
- 7z.exe (http://www.7-zip.org/download.html)
- ImageMagick-6.9.5-9-Q8-x64-dll.exe (http://www.imagemagick.org/download/binaries)

Be sure that the executables can be found by the computer by setting the environment variables.
You can manage environment variables on Windows 7 by going to:
Control panel > System and security > System > Advanced system settings > Advanced > Environment variables.
This will give the window as shown below.

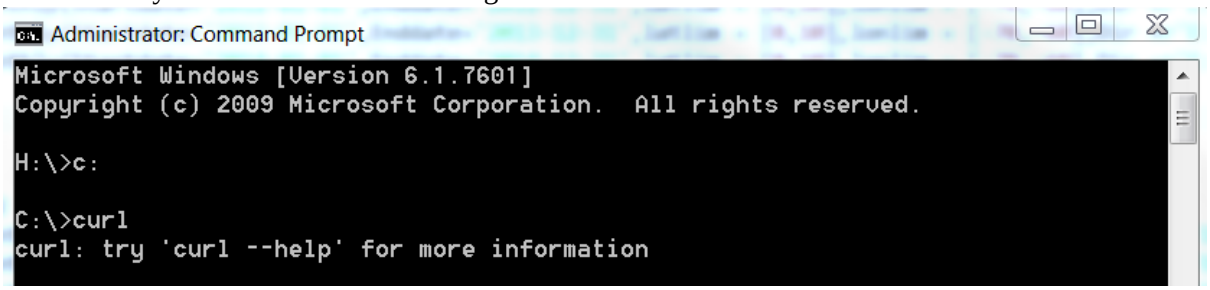


Edit the PATH variable by adding the path to the executable. Make sure you do not remove the original path in the PATH environment by separating the paths with a semi-colon (;) sign.

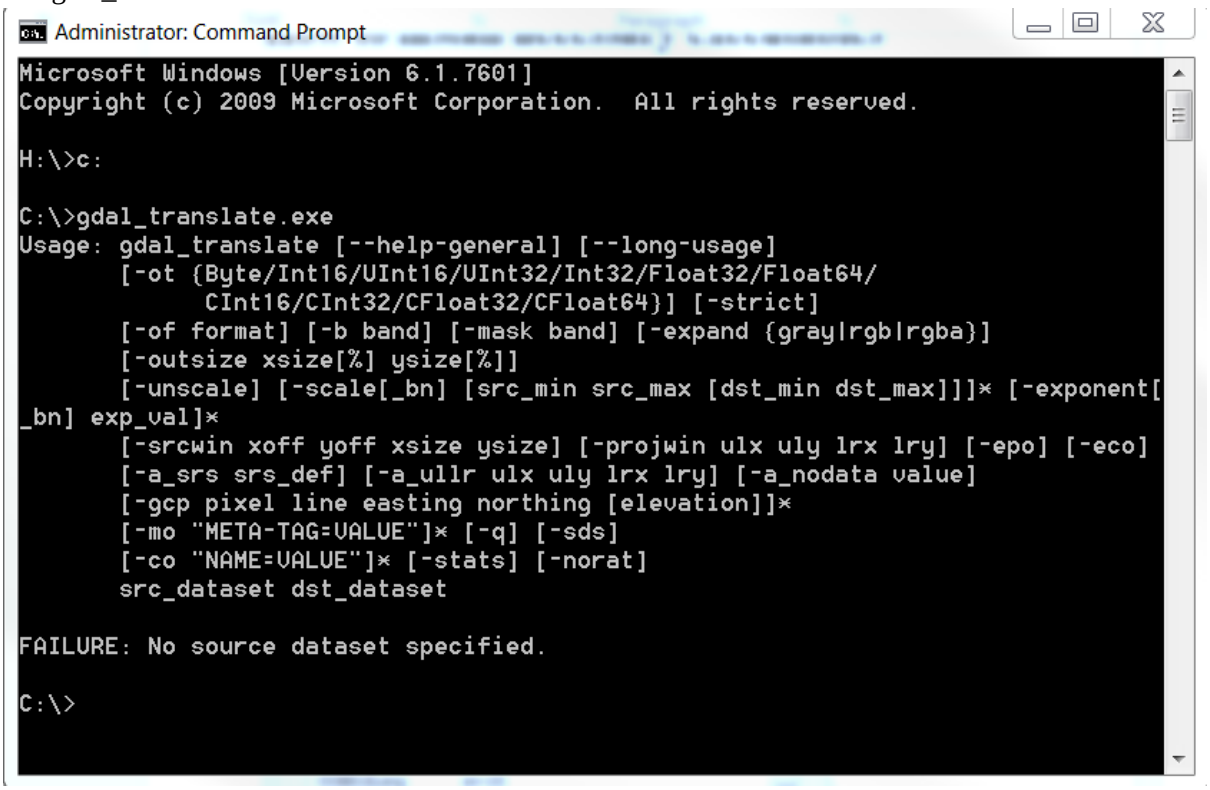


You can check if an executable can be found by typing the name of the executable in the command prompt.

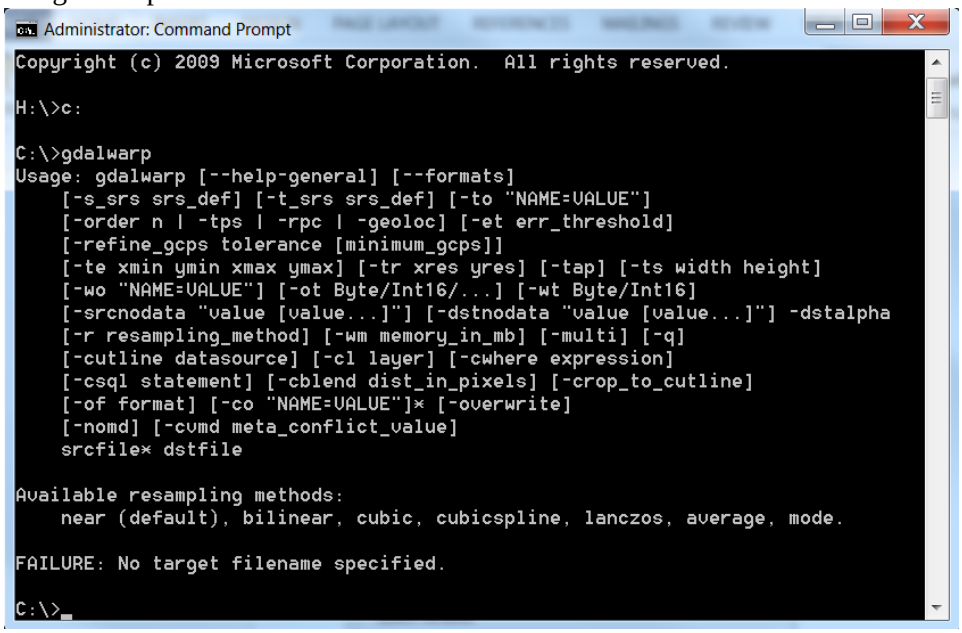
For curl.exe you need to see the following:



For gdal_translate.exe:



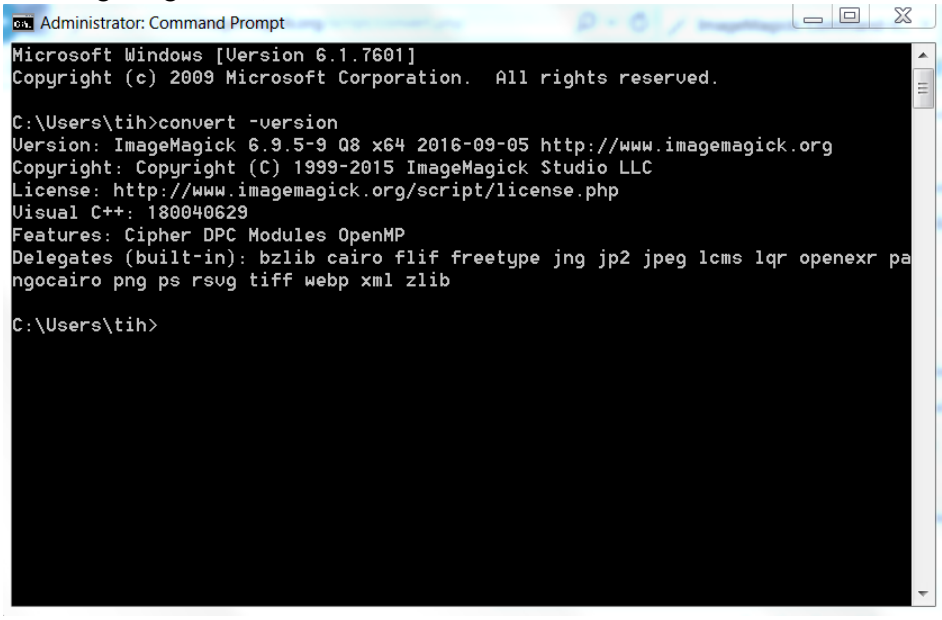
For gdalwarp.exe:



For gdalbuildvrt.exe:

For 7z.exe:

For ImageMagick.exe, use “convert –version”:



If the command prompt shows another screen than above, than you need to add the path to the environmental variables as described above or/and add the path in the WA_path.py as described in the next paragraph.

For the ImageMagick you need to add the directory to the path variable in which the covert.exe is located.

For gdal_translate.exe, gdalwarp.exe, and gdalbuildvrt.exe it is important that not the executables are used that are located inside Anaconda. Probably gdalwarp.exe is also located in another map outside Anaconda. To be sure that the right executable is used, you can define the path also in the WA_path.py. How to do this, is explained in the next section.

Introducing Water Accounting plus python tool

The most recent Water Accounting plus python tools can be downloaded from the WA+ Github account (https://github.com/wateraccounting/wa). Download the whole database and extract (unzip) the folder. After extracting the folder, rename the main folder from wa-master into wa.

Before running the tools you need to edit the WebAccounts.py. This file is located in the wa main directory and contains the log-in information needed for various websites from which data will be downloaded.

Open the file and fill in the first part of the bracket of each line your username. Then fill in the second part of the bracket your password. The username and password is separated with a comma (,). The website where you can create an account is also shown in the file (see figure below).

```
# -*- coding: utf-8 -*-
"""
Created on Thu Mar 17 09:27:36 2016

@author: tih
"""

def Accounts(Type=None):

    User_Pass = {
    'GLDAS'                 :   ['',''],  # create an account https://urs.earthdata.nasa.gov//users//new
    'GLEAM'                 :   ['',''],  # create an account http://www.gleam.eu/
    'FTP_WA'                :   ['','']   #  Only for WA+ team members
    }

    Selected_Path = User_Pass[Type]

    return(Selected_Path)
```

Also the WA_Paths.py can be adjusted to the paths of the user's computer. This is to make sure that the right executable is used. Here you can define the path to the gdal executables (within this directory the gdalwarp.exe, gdal_translate.exe, etc. are located), be sure that this is outside Anaconda. Also the path to the 7z.exe and curl.exe executables can be defined. If nothing is filled in than the path from the systems variable will be used. See example below:

```
"""
Authors: Tim Hessels
         UNESCO-IHE 2016
Contact: t.hessels@unesco-ihe.org
Repository: https://github.com/wateraccounting/wa
Module: -
"""

def Paths(Type=None):

    User_Path = {
    'GDAL'                  :   r'C:\Program Files (x86)\GDAL',          #folder
    '7z.exe'                :   r'C:\Program Files (x86)\GDAL\7z.exe',   #complete path to executable
    'curl.exe'              :   r'C:\Program Files (x86)\GDAL\curl.exe'  #complete path to executable
    }

    Selected_Path = User_Path[Type]

    return(Selected_Path)
```

There are three kind of scripts Collect, Products, and Sheets.
With the collect script you can download and collect data and with the product script a WA+ product will be made by using the collect scripts. With the sheets script a WA+ sheet will be made displaying the outputs of WA+ calculations.

For now the Collect modules are:

- CHIRPS
- GLDAS
- CFSR
- DEM
- ALEXI
- TRMM
- MOD13
- MOD15
- MOD16
- MOD17

The Products modules are:

- ETref
- ETens

The Sheets modules are:

- sheet1
- sheet2

In order to run a script you have to do some python coding inside Spyder. The needed parameters and the additional variables for all functions are shown in the next three chapters: Collect functions, Product functions, and Sheet functions.

Water Accounting plus tool: Collect functions

General

A WA+ collect module can be imported by first loading the scripts in python. This can be done by opening spyder and typing:

`from wa.Collect import …..name_module…….`

The name_module can be any of the modules defined below, which have been downloaded alongside the WA+ tools from Github.

CHIRPS

CHIRPS v2.0 rainfall data is downloaded from the ftp://chg-ftpout.geog.ucsb.edu server. There are two functions in the CHIRPS module, namely daily and monthly. The daily or monthly function will download the daily or monthly CHIRPS v.2.0 data, respectively. The outputs are tiff files in WGS84 projection.

	CHIRPS v2.0
Startdate	1981-01-01
Enddate	Now
Latitude	50 N – 50 S
Longitude	180 W – 180 E
Resolution	0.05 ˚

Required parameters are:

- Dir >> The directory where the CHIRPS data will be stored.
- Startdate >> The start day of the wanted dataset in the format yyyy-mm-dd
- Enddate >> The end day of the wanted dataset in the format yyyy-mm-dd
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

Additional parameters

- cores >> amount of cores used for collecting the data to allow parallel downloading **(at present this is not working!)**
-

The latlim and lonlim must always start with the lowest latitude or longitude.

Below is an example:

```
from wa.Collect import CHIRPS
CHIRPS.daily(Dir='C:/Temp/', Startdate='2003-02-01', Enddate='2003-04-22', latlim=[-10, 30], lonlim=[-20, 25])

CHIRPS.monthly(Dir='C:/Temp/', Startdate='2006-06-01', Enddate='2008-07-05', latlim=[40.4, 45.7], lonlim=[-20.3, -17.4])
```

GLDAS

GLDAS v1.0 is a global modelled weather dataset. In order to download the data, you need to create an account here: https://urs.earthdata.nasa.gov//users//new. The username and password must be filled in the WebAccounts.py file next to the GLDAS name, as shown below:

```
def Accounts(Type=None):

    User_Pass = {
    'NASA'                  :   ['',''],
    'GLEAM'                 :   ['',''],
    'FTP_WA'                :   ['','']
    }
```

	GLDAS v1.0
Startdate	2000-02-24
Enddate	2016-07-31
Latitude	59.875 N – 59.875 S
Longitude	180 W – 180 E
Resolution	0.25˚

After updating this WebAccounts.py python file you can collect the GLDAS data by using the WA+ collect tool. There are 3 functions in this module, namely three_hourly, daily, monthly. Those functions reveal the temporal resolution of the end product. The end products will be tiff files in WGS84 projection.

Required parameters for all functions:

- Dir >> The directory where the CHIRPS data will be stored.
- Vars >> GLDAS variables that must be downloaded see Vars table below
- Startdate >> The start day of the wanted dataset
- Enddate >> The end day of the wanted dataset
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

Vars	Parameter output
avgsurft	surface average surface temperature [C]
canopint	surface plant canopy surface water [kg/m^2]
evap	surface total evapotranspiration [mm/(month/daily/3hourly)]
lwdown	surface surface incident longwave radiation [w/m^2]
lwnet	surface net longwave radiation [w/m^2]
psurf	surface surface pressure [kPa]
qair	surface near surface specific humidity [kg/kg]
qg	surface ground heat flux [w/m^2]
qh	surface sensible heat flux [w/m^2]
qle	surface latent heat flux [w/m^2]
qs	surface surface runoff [mm/(month/daily/3hourly)]
qsb	surface subsurface runoff [mm/(month/daily/3hourly)]
qsm	surface snowmelt [mm/(month/daily/3hourly)]
rainf	surface rainfall rate [mm/(month/daily/3hourly)]
swe	surface snow water equivalent [mm/(month/daily/3hourly)]
swdown	surface surface incident shortwave radiation [w/m^2]
swnet	surface net shortwave radiation [w/m^2]
snowf	surface snowfall rate [kg/m^2/s]
soilm1	0-10 cm underground soil moisture content [mm]
soilm2	10-40 cm underground soil moisture content [mm]
soilm3	40-100 cm underground soil moisture content [mm]
soilm4	100-200 cm underground soil moisture content [mm]
tsoil1	0-10 cm underground soil temperature [C]
tsoil2	10-40 cm underground soil temperature [C]
tsoil3	40-100 cm underground soil temperature [C]
tsoil4	100-200 cm underground soil temperature [C]
tair	surface near surface air temperature [C]
wind	surface near surface wind speed [m/s]

Additional needed parameters for three hourly function:

- Periods >> Which 3 hourly period(s) must be downloaded, see period table below

Period	Time
1	00:00 - 03:00
2	03:00 - 06:00
3	06:00 - 09:00
4	09:00 - 12:00
5	12:00 - 15:00
6	15:00 - 18:00
7	18:00 - 21:00
8	21:00 - 24:00

Additional needed parameters for daily function:

- SumMean >> If 1 you get the daily average if 0 you will not get the average
- Min >> If 1 you get the daily minimum if 0 you will not get the minimum
- Max >> If 1 you get the daily maximum if 0 you will not get the maximum

Additional parameters

- cores >> amount of cores used for collecting the data to allow parallel downloading **(at present this is not working!)**

The latlim and lonlim must always start with the lowest latitude or longitude.

Examples:

```
from wa.Collect import GLDAS

GLDAS.three_hourly(Dir = 'C:/Temp/', Vars = ['qair', 'tair'], Startdate = '2004-12-20',
                           Enddate = '2005-01-10', latlim = [25, 36], lonlim = [-76, -73], Periods = [4, 5])


GLDAS.daily(Dir = 'C:/Temp/', Vars = ['qair'], Startdate = '2004-12-20', Enddate = '2005-01-01',
                           latlim=[-52, -45.3], lonlim = [-50, -20.5], SumMean = 1, Min = 0, Max = 1)

GLDAS.monthly(Dir = 'C:/Temp', Vars = ['swnet', 'wind', 'lwnet'], Startdate = '2004-12-20',
                           Enddate = '2005-03-10', latlim = [38, 41], lonlim = [-76, -73])
```

CFSR

CFSR data is modelled meteo data. This python module can only download the radiation components automatically for now. The radiation components are longwave / shortwave and downwards / upwards radiation. Other components will be added in later versions.

	CFSR	CFSR v2.0
Startdate	1979-01-01	2011-04-01
Enddate	2011-03-31	Now
Latitude	50 N – 50 S	50 N – 50 S
Longitude	180 W – 180 E	180 W – 180 E
Resolution	0.3125˚	0.204545˚

The 4 components can be downloaded by using 1 function within the CFSR module, named daily. This function can only download daily data. The output are tiff files in WGS84 projection. The function will automatically choose if it downloads the CFSR or CFSR v2.0 depending on the time range that you want to download.

Required parameters:

- Dir >> The directory where the CHIRPS data will be stored.
- Vars >> CFSR radiation variables that must be downloaded see Vars table below
- Startdate >> The start day of the wanted dataset
- Enddate >> The end day of the wanted dataset
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

Vars	Parameter output
dlwsfc	surface downward longwave radiation [w/m^2]
dswsfc	surface downward shortwave radiation [w/m^2]
ulwsfc	surface upwards longwave radiation [w/m^2]
uswsfc	surface upwards shortwave radiation [w/m^2]

Additional parameters

- cores >> amount of cores used for collecting the data to allow parallel downloading **(at present this is not working!)**

The latlim and lonlim must always start with the lowest latitude or longitude.

Examples:

```
from wa.Collect import CFSR
CFSR.daily(Dir='C:/Temp/', Vars = ['dlwsfc', 'ulwsfc'], Startdate = '2010-02-01', Enddate = '2014-02-02',
                latlim = [-10, 30], lonlim = [-20, 120])

CFSR.daily(Dir='C:/Temp/', Vars = ['dswsfc'], Startdate = '2008-04-19', Enddate = '2012-03-02',
                latlim = [-10.5, 0.6], lonlim = [10.4, 12.6])
```

DEM

DEM data is automatically downloaded from HydroSHED (http://hydrosheds.cr.usgs.gov/). It contains a Digital Elevation Model map with a spatial resolution of 90 meters.
The DEM module only contains one function to collect the DEM map of the area specified by the user.

	HydroSHED
Startdate	-
Enddate	-
Latitude	90 N – 90 S
Longitude	180 W – 180 E
Resolution	0.0008333˚

Required parameters:

- Dir >> The directory where the CHIRPS data will be stored.
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

Additional parameters

- resample >> If 1, than the image will be resampled to a resolution of 0.001 ˚

The latlim and lonlim must always start with the lowest latitude or longitude.

Examples:

```
from wa.Collect import DEM
DEM.HydroSHED(Dir='C:/Temp/', latlim=[-10, 30], lonlim=[-20, 120], Resample=1)

DEM.HydroSHED(Dir='C:/Temp/', latlim=[-10, 30], lonlim=[-20, 120])
```

ALEXI

The evaporation data of ALEXI can only be downloaded when you have an account on the FTP server of the water accounting plus group. This dataset is download from the WA+ FTP server, and is not freely accessible. Other evaporation dataset that are freely available are GLEAM or MOD16. This code is able to download weekly ALEXI data by using the weekly function. The output are tiff files in WGS84 projection. The spatial resolution of the product is 0.05˚.

ALEXI	
Startdate	2003-01-01
Enddate	2015-12-31
Latitude	70 N – 60 S
Longitude	180 W – 180 E
Resolution	0.05˚

Required parameters:

- Dir >> The directory where the ALEXI data will be stored.
- Startdate >> The start day of the wanted dataset
- Enddate >> The end day of the wanted dataset
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

The latlim and lonlim must always start with the lowest latitude or longitude.

Examples:

```
from wa.Collect import ALEXI
ALEXI.weekly(Dir='C:/Temp/', Startdate='2015-05-01', Enddate='2015-07-20',latlim=[-0.6, 9.2], lonlim=[-29.3, 6.5])
```

TRMM

The TRMM 3B42 v7 (daily) and TRMM 3B43 v7 (monthly) precipitation data can be downloaded by using this function. The data originates from the ftp server of NASA (ftp://disc2.nascom.nasa.gov). There are two functions namely the daily and monthly function to download and create daily or monthly TRMM images in Gtiff format. The TRMM data is available since 1998-01-01 till 2015-04-30. The output tiff files are in WGS84 projection and has a resolution of 0.25˚.

	TRMM 3B42/43 v7
Startdate	1998-01-01
Enddate	2015-04-30
Latitude	50 N – 50 S
Longitude	180 W – 180 E
Resolution	0.25˚

Required parameters:

- Dir >> The directory where the TRMM data will be stored.
- Startdate >> The start day of the wanted dataset
- Enddate >> The end day of the wanted dataset
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

Additional parameters

- cores >> amount of cores used for collecting the data to allow parallel downloading **(at present this is not working!)**

The latlim and lonlim must always start with the lowest latitude or longitude.

Examples:

```
from wa.Collect import TRMM
TRMM.daily(Dir='C:/Temp/', Startdate='1999-02-01', Enddate='1999-02-28',
           latlim=[-2.3, 5.6], lonlim=[16.5, 82.2])

TRMM.monthly(Dir='C:/Temp/', Startdate='2005-02-01', Enddate='2006-02-28',
             latlim=[58.0, 62.5], lonlim=[-3.2, 30.2], cores=2)
```

MOD13

	MOD 13Q1 v6 NDVI
Startdate	2000-02-18
Enddate	present
Latitude	90 N – 90 S
Longitude	180 W – 180 E
Resolution	0.0025˚

The MOD13 product obtained from the MODIS satellite includes the Normalized Difference Vegetation Index (NDVI). This 16-daily product can be downloaded with the NDVI_16daily function within the MOD13 module. The date originates from the https server of USGS (http://e4ftl01.cr.usgs.gov/). The product is available since 18 February 2000 till present. The original data is projected in MODIS Sinusoidal projection and has a spatial resolution of 250 meters. This function reproject and resamples the data into the WGS84 projection and a spatial resolution of 0.0025˚. The output files are tiff files.
Needed parameters:

- Dir >> The directory where the NDVI data will be stored.
- Startdate >> The start day of the wanted dataset
- Enddate >> The end day of the wanted dataset
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

Additional parameters

- cores >> amount of cores used for collecting the data to allow parallel downloading (**not working at the moment**!)

The latlim and lonlim must always start with the lowest latitude or longitude.
Examples:

```
from wa.Collect import MOD13
MOD13.NDVI_16daily(Dir='C:/Temp/', Startdate='2012-01-01',Enddate='2012-01-31',
                   latlim=[8.2, 10.4], lonlim=[2.2, 6.4])
```

MOD15

	MOD 15A2H v6
Startdate	2000-02-18
Enddate	present
Latitude	90 N – 90 S
Longitude	180 W – 180 E
Resolution	0.005˚

The Fractional Photosynthetically Active Radiation (FPAR) and Leaf Area Index (LAI) obtained from the MODIS satellite is an 8 daily product with a spatial resolution of 500 meter. The raw data can be downloaded from the https server of USGS (http://e4ftl01.cr.usgs.gov/). But also the FPAR_8daily function and LAI_8daily function within the MOD15 module can be used to download, clip, and reproject the data. The product is available since 18 February 2000 till present. The original data is projected in MODIS Sinusoidal projection and has a spatial resolution of 500 meters. This function reproject and resamples the data into the WGS84 projection and a spatial resolution of 0.005˚. The output files are tiff files.
Needed parameters:

- Dir >> The directory where the FPAR/LAI data will be stored.
- Startdate >> The start day of the wanted dataset

- Enddate >> The end day of the wanted dataset
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

Additional parameters

- cores >> amount of cores used for collecting the data to allow parallel downloading (**not working at the moment**!)

The latlim and lonlim must always start with the lowest latitude or longitude.
Examples:

```
from wa.Collect import MOD15
MOD15.FPAR_8daily(Dir='C:/Temp', Startdate='2012-01-01', Enddate='2012-01-31',
                  latlim=[8.2, 10.4], lonlim=[2.2, 6.4])

MOD15.LAI_8daily(Dir='C:/Temp', Startdate='2002-01-01', Enddate='2002-08-20',
                  latlim=[18.2, 20.6], lonlim=[4.5, 8.4])
```

MOD16

MOD 16A2 ET	
Startdate	2000-01-01
Enddate	2014-12-31
Latitude	90 N – 90 S
Longitude	180 W – 180 E
Resolution	0.005˚

Evapotranspiration data from MODIS is stored in the MOD16 product. The monthly data can be downloaded from the FTP server of NTSG (ftp://ftp.ntsg.umt.edu/). The MOD16 module has a function called ET_monthly, to download this data automatically for the area of interest. This monthly product is available from the start of 2000 till the end of 2014. The original data has a MODIS Sinusoidal projection and a spatial resolution of 1000 meters. The ET_monthly function will download, reproject, and clip the data to a spatial resolution of 0.005˚ in WGS84 projection by using the nearest neighbor method. The output files are tiff files.
Needed parameters:

- Dir >> The directory where the ET data will be stored.
- Startdate >> The start day of the wanted dataset
- Enddate >> The end day of the wanted dataset
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

Additional parameters

- cores >> amount of cores used for collecting the data to allow parallel downloading (**not working at the moment**!)

The latlim and lonlim must always start with the lowest latitude or longitude.
Examples:

```
from wa.Collect import MOD16
MOD16.ET_monthly(Dir='C:/Temp/', Startdate='2004-01-01',Enddate='2005-01-31',
                        latlim=[-20, -12], lonlim=[42, 49])
```

MOD17

The Net Primary Productivity (NPP) and Gross Primary Productivity (GPP) are MOD17 products, and can both be downloaded using the NPP_yearly or the GPP_8daily functions respectively. The original data can be downloaded from the USGS website (http://e4ftl01.cr.usgs.gov/) and has a spatial resolution of 500 meters. The NPP is a yearly product and the GPP is an 8-daily product. Both functions downloads

the data, reprojects, and clip the data, so the remaining is a tiff file which only includes the area of interest. The result has a spatial resolution of 0.005˚ and has a WGS84 projection.

	MOD 17A3H v6 NPP	MOD 17A2H v6 GPP
Startdate	2000-01-01	2000-02-18
Enddate	2014-12-31	Present
Latitude	90 N – 90 S	90 N – 90 S
Longitude	180 W – 180 E	180 W – 180 E
Resolution	0.005˚	0.005˚

Needed parameters:

- Dir >> The directory where the ET data will be stored.
- Startdate >> The start day of the wanted dataset
- Enddate >> The end day of the wanted dataset
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

Additional parameters

- cores >> amount of cores used for collecting the data to allow parallel downloading (**not working at the moment**!)

The latlim and lonlim must always start with the lowest latitude or longitude.
Examples:

```
from wa.Collect import MOD17
MOD17.GPP_8daily(Dir='C:/Temp/', Startdate='2012-01-01',Enddate='2012-12-31',
                         latlim=[-31.1, -25.6], lonlim=[149.5, 155.1])

MOD17.NPP_yearly(Dir='C:/Temp/', Startdate='2012-01-01',Enddate='2012-12-31',
                         latlim=[-31.1, -25.6], lonlim=[149.5, 155.1])
```

Water Accounting plus tool: Products functions

General

A WA+ products module can be imported by first loading the scripts in python. This can be done by opening spyder and typing:
from wa.Products import …..name_module…….
The products are developed by the WA+ team and make use of open access datasets.
The name_module can be any of the modules as defined below.

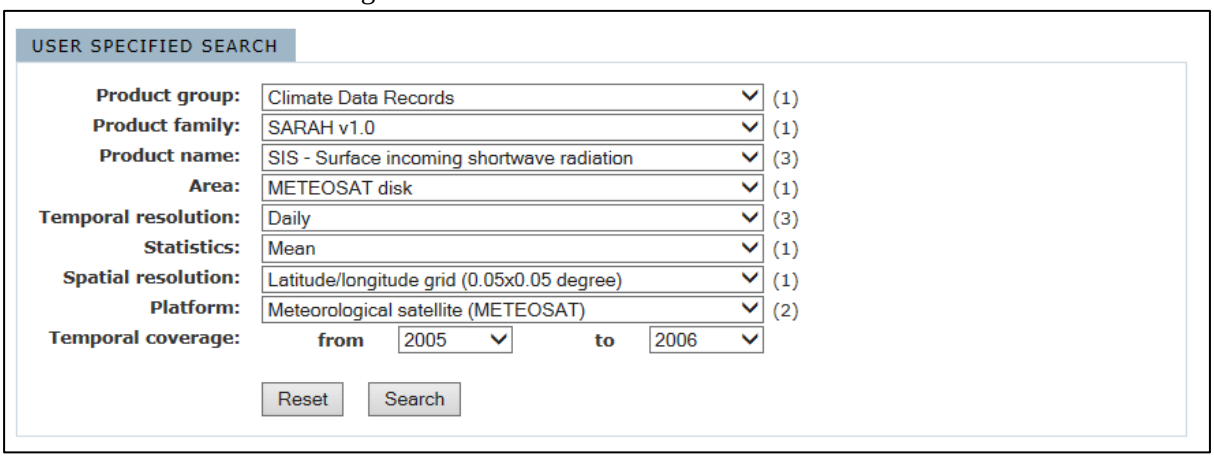
ETref

This ETref tool has two functions, namely the daily and the monthly function. Those functions will create daily or monthly reference evapotranspiration maps by using the definitions and equations as published in FAO Irrigation & Drainage Paper 56. The inputs are the DEM map from Hydroshed, Meteo data from GLDAS, and Solar Radiation from CFSR or LANDSAF. The input data will be downloaded automatically.

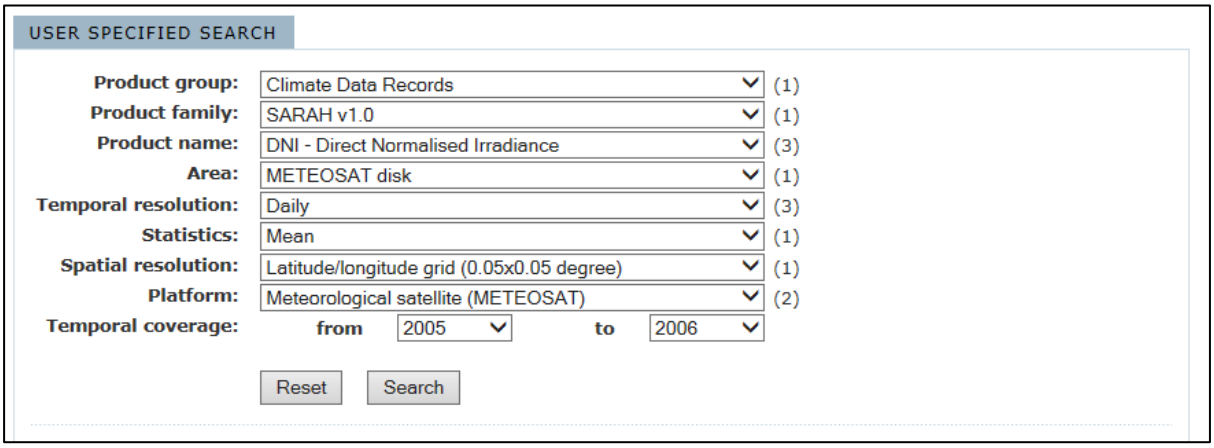
ETref WA+	
Input	Meteo Data from **GLDAS**
	Solar Radiation from **CFSR** or **LANDSAF**
	DEM from **HydroSHED**
Output	Reference Evapotranspiration (mm/day or mm/month)

The only exception is when the LANDSAF data is used for solar radiation instead of CFSR. In this case, the LANDSAF raw dataset must be downloaded before running the script. This can be done by following the steps as are described below (these can be neglected if CFSR data is used):

1. Go to http://www.cmsaf.eu/EN/Home/home_node.html
2. Go to Data Access and open the web user interface
3. Search for the surface incoming shortwave radiation as shown below:



4. Search for the Direct normalised irradiance as shown below:



5. Create a LANDSAF folder with two subfolders named SIS and SID
6. Put the data downloaded from step 3 in the SIS folder (the raw .nc.gz files)
7. Put the data downloaded from step 4 in the SID folder (the raw .nc.gz files)

The created output files will be daily or monthly tiff files with the total reference evapotranspiration in mm/day for the daily function and mm/month for the monthly function. The spatial resolution of the output data can be defined by the user. If no resolution is defined, the output will have the same resolution as the DEM map.

Required parameters:

- Dir >> The directory where the ETref data will be stored.
- Startdate >> The start day of the wanted dataset
- Enddate >> The end day of the wanted dataset
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

Additional parameters

- pixel_size >> the pixel size of the output in degrees, if not defined the output have the spatial resolution as the DEM map
- LANDSAF >> 1 if LANDSAF data must be used.
- SourceLANDSAF >> the directory to the LANDSAF data. This directory includes the SID and SIS folders including the needed datasets. If not defined the CFSR data will be used.
- cores >> amount of cores used for collecting the data to allow parallel downloading **(at present this is not working!)**

The latlim and lonlim must always start with the lowest latitude or longitude.

Examples:

```
from wa.Products import ETref
ETref.monthly(Dir='C:/Temp', Startdate='2012-09-01', Enddate='2012-12-31',
              latlim=[50.2, 55.3], lonlim=[-10.3, -7.9], pixel_size=0.01)

ETref.daily(Dir='C:/Temp', Startdate='2004-01-02', Enddate='2004-01-14',
              latlim=[-25.6, -20.3], lonlim=[17.2, 30.4], LANDSAF=1, SourceLANDSAF =r'D:\LANDSAF')
```

ETens

ETensV1.0 is an ensemble evapotranspiration product developed by the WA+ team. The end product combines 7 different ET datasets by removing the outliers for every pixel. The products that are used are ALEXI, CMRSET, ETmonitor, GLEAM, MOD16, SEBS, and SSEBop. The NDVI (MOD13) is used to downscale the product to a 0.0025˚ spatial resolution. The temporal scale is monthly and is available between Januari 2003 till December 2014. The ETensV1.0 product spans all longitudes and latitudes between 40S and 40N. The ETens module can collect the data automatically from the WA+ FTP server (ftp.wateraccounting.unesco-ihe.org). The username and password must be provided within the WebAccounts.py (FTP_WA), otherwise the tool cannot be used.

ETensV1.0	
Startdate	2003-01-01
Enddate	2014-12-31
Latitude	40 N – 40 S
Longitude	180 W – 180 E
Resolution	0.0025˚

Required parameters:

- Dir >> The directory where the ETens data will be stored.
- Startdate >> The start day of the wanted dataset
- Enddate >> The end day of the wanted dataset
- latlim >> latitude limits of the wanted dataset
- lonlim >> longitude limits of the wanted dataset

The latlim and lonlim must always start with the lowest latitude or longitude.

Examples:

```
from wa.Products import ETens
ETens.monthly(Dir = 'C:/Temp', Startdate='2003-01-01',Enddate='2003-04-05',latlim = [0,5],lonlim = [35,40])
```

Water Accounting plus tool: Sheets functions

General
A WA+ Sheets module can be imported by loading the scripts in python. This can be done by opening Spyder and typing:
from wa.Sheets import …..name_module…….

The functions will create the water accounting plus sheets automatically, based from a csv template. The csv template can be found in the ~wa\Sheets\csv folder. This example csv file can also be used to test the Sheets function. The name_module can be all the modules defined below.

create_sheet1:
The create_sheet1 function will create the first WA+ sheet (Resource Base Sheet). The data is extracted from a csv file which must be consistent with the lay-out of the ~wa\Sheets\csv\Sample_sheet1.csv. The output data will be a jpg file with the official WA+ sheet lay-out filled in with the numbers from the csv file.

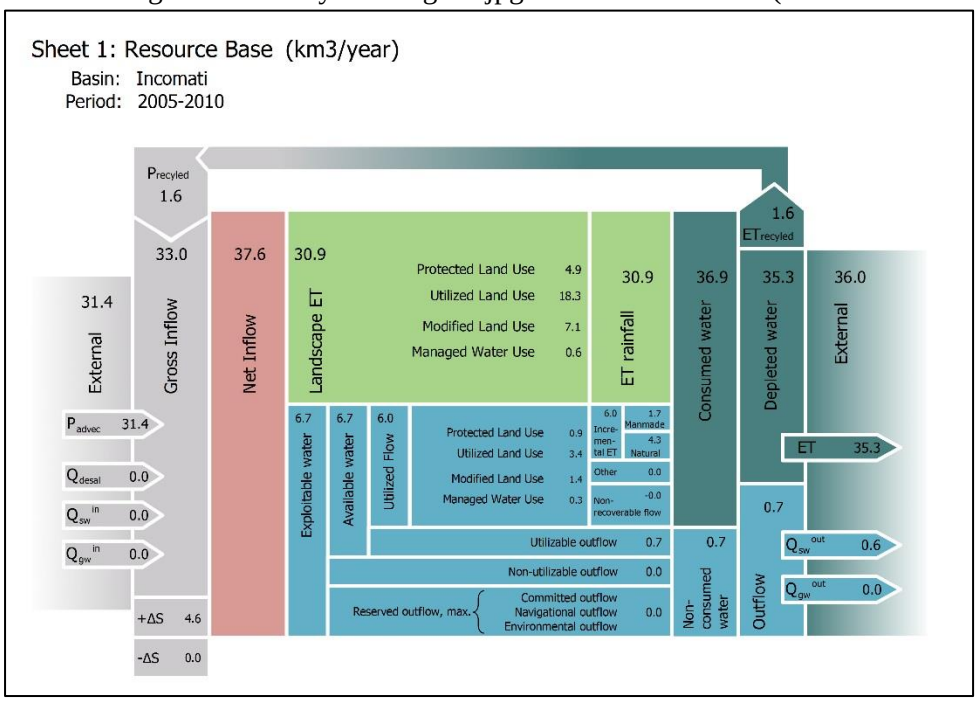
Required parameters:

- basin >> Define the basin name
- period >> Define the period of the accounts
- units >> Define the units
- data >> Path to the csv file with the data
- output >> Path to the ouput jpg file

Example:

```
from wa.Sheets import *
create_sheet1(basin='Nile Basin', period='2010', units='km3/year',
              data=r'C:\Users\tih\Documents\Water_Accounting\GitHub\wa\Sheets\csv\Sample_sheet1.csv',
              output=r'C:\sheet_1.jpg')
```

After running the function you will get a jpg file as shown below (data from the example csv file):



create_sheet2:

The create_sheet2 function will create the second WA+ sheet (Evapotranspiration sheet). The data is extracted from a csv file selected by the user, which must be consistent with the lay-out of the ~wa\Sheets\csv\Sample_sheet1.csv. The output data will be a jpg file with the official WA+ sheet lay-out filled in with the numbers from the csv file.

Required parameters:

- basin >> Define the basin name
- period >> Define the period of the accounts
- units >> Define the units
- data >> Path to the csv file with the data
- output >> Path to the ouput jpg file

Example:

```
from wa.Sheets import *
create_sheet2(basin='Nile Basin', period='2010', units='km3/year',
              data=r'C:\Users\tih\Documents\Water_Accounting\GitHub\wa\Sheets\csv\Sample_sheet2.csv',
              output=r'C:\sheet_2.jpg')
```

After running the function you will get a jpg file as shown below (data from the example csv file):