ETo Documentation

Release 1.0.5

Mike Kittridge

May 15, 2018

Modules

The ETo package contains a class and associated functions to calculate reference evapotranspiration (ETo) using the UN-FAO 56 paper [1]. Additional functions have been added to calculate historic ETo or potential evapotranspiration (PET) for comparison purposes.

The GitHub repository is found here

A parameter estimation function hs also been added to the base class to convert most any variety of metereological parameter inputs to the necessary parameters needed to calculate ETo.

CHAPTER 1

Introduction

Evapotranspiration (ET) is the combination process of evaporation from surfaces and transpiration from plant tissues and primarily through stomata. The direct measurement of actual ET is very difficult and indirect estimates are usually made as a consequence. Since ET is such a doninant part of the water cycle (~65%) and optimised agriculture require accurate estimates to determine irrigation needs, much work has been performed to determine accurate and practical methodologies for estimating ET.

Much of the development came together with the United Nations Food and Agriculture organization (UN-FAO)as they worked together with researchers for an international standard. As it was most practical to only measure metereological parameters (e.g. temperature, relative humidity, etc), the term reference ET (ETo) was coined to define a specific vegetated surface by which the ET estimation would represent. Different crop coefficients could be applied to convert the reference crop to other types of crops or vegetated surfaces.

The method utilises the Penman-Monteith ET equation and the guildline provides methods to estimate missing metereological parameters. The method and guildlines can handle as little data as minimum and maximum temperature to the full set of metereological parameters. With this standardisation, researchers and water managers can accurate estimate ETo and ultimately ET and be able to compare the results across regions.

This package is meant to assist in efficiently estimating ETo for time series metereological data where instrumentation and consequenty the parameters change over time due to changes in priorities or budgets. Additional historic ETo or potential ET (PET) methods have been added for comparison purposes.

CHAPTER 2

Methodology

2.1 Reference evapotranspiration (ETo)

The derivation of ETo had developed over many years with several different equations. The latest and hopefully last variant is derived from the Penman-Montieth equation.

Extensive documentation on the methods and concepts can be found in the UN-FAO 56 paper [1]

2.2 Hargreaves

The derivation for the Hargreaves equation can also be found in the UN-FAO 56 paper.

The History and Evaluation of Hargreaves Evapotranspiration Equation [2] is a more detailed description and background of the Hargreaves method.

2.3 References

CHAPTER 3

Installation

ETo can be installed via pip or conda:

```
pip install eto
```

or:

```
conda install -c mullenkamp eto
```

The core dependency is *Pandas <http://pandas.pydata.org/pandas-docs/stable/>*‘_.

CHAPTER 4

How to use ETo

This section will describe how to use the ETo package. The ETo class and functions depend heavily on the Pandas package. Nearly all outputs are either as Pandas Series or DataFrames.

4.1 Initialising

The package and general usage is via the main ETo class. It can be initialised without any initial input parameters.

```
from eto import ETo, datasets
import pandas as pd

et1 = ETo()
```

4.2 Parameter estimation

The input data can be read into the class at initiatisation or via the param_est function.

We first need to get an example dataset and read it in via pd.read_csv.

```
In [1]: ex1_path = datasets.get_path('example1')

In [2]: tsdata = pd.read_csv(ex1_path, parse_dates=True, infer_datetime_format=True,␣
↪index_col='date')

In [3]: tsdata.head()
Out[3]:
             R_s      T_max  T_min   e_a
date
2000-01-01  13.4  17.700001   13.7  1.60
2000-01-02  14.7  19.600000   13.0  1.68
2000-01-03  12.2  21.700001    9.2  1.09
```

(continues on next page)

(continued from previous page)

```
2000-01-04   4.2  14.400000    8.4  1.04
2000-01-05  14.1  11.700000    8.9  1.12
```

Now we can run the parameter estimation using the newly loaded in dataset using the default parameters.

```
In [4]: et1.param_est(tsdata)

In [5]: et1.ts_param.head()
Out[5]:
             R_s      T_max  T_min   e_a        R_n    G  T_mean  T_dew  \
date
2000-01-01  13.4  17.700001   13.7  1.60   9.285734  0.0   15.70    NaN
2000-01-02  14.7  19.600000   13.0  1.68  10.017707  0.0   16.30    NaN
2000-01-03  12.2  21.700001    9.2  1.09   8.475450  0.0   15.45    NaN
2000-01-04   4.2  14.400000    8.4  1.04   4.385135  0.0   11.40    NaN
2000-01-05  14.1  11.700000    8.9  1.12   9.540114  0.0   10.30    NaN

            RH_min  RH_max ...   n_sun  U_z          P     gamma     e_max  \
date                       ...
2000-01-01     NaN     NaN ...     NaN  NaN  95.527647  0.063526  2.025376
2000-01-02     NaN     NaN ...     NaN  NaN  95.527647  0.063526  2.281006
2000-01-03     NaN     NaN ...     NaN  NaN  95.527647  0.063526  2.595970
2000-01-04     NaN     NaN ...     NaN  NaN  95.527647  0.063526  1.640576
2000-01-05     NaN     NaN ...     NaN  NaN  95.527647  0.063526  1.375058

               e_min       e_s     delta        R_a  U_2
date
2000-01-01  1.567747  1.796562  0.114199  44.443793  2.0
2000-01-02  1.497771  1.889388  0.118099  44.386929  2.0
2000-01-03  1.163665  1.879817  0.112606  44.325018  2.0
2000-01-04  1.102347  1.371462  0.089314  44.258075  2.0
2000-01-05  1.140328  1.257693  0.083748  44.186116  2.0

[5 rows x 21 columns]
```

4.3 Calculate ETo

Now it's just a matter of running the specific ETo function. For example, the FAO ETo.

```
In [6]: eto1 = et1.eto_fao()

In [7]: eto1.head()
Out[7]:
date
2000-01-01    2.31
2000-01-02    2.52
2000-01-03    3.20
2000-01-04    1.49
2000-01-05    2.00
Name: ETo_FAO_mm, dtype: float64
```

CHAPTER 5

Package References

5.1 Base class

class eto.**ETo**(*df=None*, *z_msl=500*, *lat=-43.6*, *lon=172*, *TZ_lon=173*, *z_u=2*, *time_int='days'*, *K_rs=0.16*, *a_s=0.25*, *b_s=0.5*, *alb=0.23*)

Class to handle the parameter estimation of metereological values and the calcuation of reference ET and similar ET methods.

This class can be either initiated with empty parameters or will initialise to the param_est function.

5.2 Parameter estimation

ETo.**param_est**(*df*, *z_msl=500*, *lat=-43.6*, *lon=172*, *TZ_lon=173*, *z_u=2*, *time_int='days'*, *K_rs=0.16*, *a_s=0.25*, *b_s=0.5*, *alb=0.23*)

Function to estimate the parameters necessary to calculate reference ET (ETo) from the FAO 56 paper *[1]* using a minimum of T_min and T_max for daily estimates and T_mean and RH_mean for hourly, but optionally utilising the maximum number of available met parameters. The function prioritizes the estimation of specific parameters based on the available input data.

Parameters

- **df** (*DataFrame*) – Input Metereological data (see Notes section).
- **z_msl** (*float or int*) – Elevation of the met station above mean sea level (m) (only needed if P is not in df).
- **lat** (*float or int*) – The latitude of the met station (dec deg) (only needed if R_s or R_n are not in df).
- **lon** (*float or int*) – The longitude of the met station (dec deg) (only needed if calculating ETo hourly)
- **TZ_lon** (*float or int*) – The longitude of the center of the time zone (dec deg) (only needed if calculating ETo hourly).

- **z_u** (*float or int*) – The height of the wind speed measurement (m).
- **time_int** (*str*) – The time interval of the input and output (either 'days' or 'hours').
- **K_rs** (*float*) – Rs calc coefficient (0.16 for inland stations, 0.19 for coastal stations)
- **a_s** (*float*) – Rs calc coefficient
- **b_s** (*float*) – Rs calc coefficient
- **alb** (*float*) – Albedo (should be fixed for the reference crop)

Returns

Return type DataFrame

Notes

The input data must be a DataFrame with specific column names according to the met parameter. The column names should be a minimum of T_min and T_max for daily estimates and T_mean and RH_mean for hourly, but can contain any/all of the following:

R_n Net radiation (MJ/m2)

R_s Incoming shortwave radiation (MJ/m2)

G Net soil heat flux (MJ/m2)

T_min Minimum Temperature (deg C)

T_max Maximum Temperature (deg C)

T_mean Mean Temperature (deg C)

T_dew Dew point temperature (deg C)

RH_min Minimum relative humidity

RH_max Maximum relative humidity

RH_mean Mean relative humidity

n_sun Number of sunshine hours per day

U_z Wind speed at height z (m/s)

P Atmospheric pressure (kPa)

e_a Actual Vapour pressure derrived from RH

Parameter estimation values refer to the quality level of the input parameters into the ETo equations. Where a 0 (or nothing) refers to no necessary parameter estimation (all measurement data was available), while a 1 refers to parameters that have the best input estimations and up to a value of 3 is the worst. Starting from the right, the first value refers to U_z, the second value refers to G, the third value refers to R_n, the fourth value refers to R_s, the fifth value refers to e_a, the sixth value refers to T_mean, the seventh value refers to P.

References

5.3 ETo functions

ETo.**eto_fao**(*max_ETo=15*, *min_ETo=0*, *interp=False*, *maxgap=15*, *export=None*)
Function to estimate reference ET (ETo) from the FAO 56 paper *[1]* using a minimum of T_min and T_max for

daily estimates and T_mean and RH_mean for hourly, but optionally utilising the maximum number of available met parameters. The function prioritizes the estimation of specific parameters based on the available input data.

Parameters

- **max_ETo** (*float or int*) – The max realistic value of ETo (mm).
- **min_ETo** (*float or int*) – The min realistic value of ETo (mm).
- **interp** (*False or str*) – Should missing values be filled by interpolation? Either False if no interpolation should be performed, or a string of the interpolation method. See Pandas interpolate function for methods. Recommended interpolators are 'linear' or 'pchip'.
- **maxgap** (*int*) – The maximum missing value gap for the interpolation.
- **export** (*str*) – Export path for csv output or None to not export.

Returns If fill=False, then the function will return a Series of estimated ETo in mm. If fill is a str, then the function will return a DataFrame with an additional column for the filled ETo value in mm.

Return type DataFrame or Series

References

ETo.**eto_hargreaves**(*max_ETo=15*, *min_ETo=0*, *interp=False*, *maxgap=15*, *export=None*)

Function to estimate Hargreaves ETo using a minimum of T_min and T_max, but optionally utilising the maximum number of available met parameters. The function prioritizes the estimation of specific parameters based on the available input data.

Parameters

- **max_ETo** (*float or int*) – The max realistic value of ETo (mm).
- **min_ETo** (*float or int*) – The min realistic value of ETo (mm).
- **interp** (*False or str*) – Should missing values be filled by interpolation? Either False if no interpolation should be performed, or a string of the interpolation method. See Pandas interpolate function for methods. Recommended interpolators are 'linear' or 'pchip'.
- **maxgap** (*int*) – The maximum missing value gap for the interpolation.
- **export** (*str*) – Export path for csv output or None to not export.

Returns If fill=False, then the function will return a Series of estimated ETo in mm. If fill is a str, then the function will return a DataFrame with an additional column for the filled ETo value in mm.

Return type DataFrame or Series

5.4 API Pages

—

Bibliography

[1] Allen, R. G., Pereira, L. S., Raes, D., & Smith, M. (1998). Crop evapotranspiration-Guidelines for computing crop water requirements-FAO Irrigation and drainage paper 56. FAO, Rome, 300(9), D05109.

[1] Allen, R. G., Pereira, L. S., Raes, D., & Smith, M. (1998). Crop evapotranspiration-Guidelines for computing crop water requirements-FAO Irrigation and drainage paper 56. FAO, Rome, 300(9), D05109.

Index