Manual on the use of Weather generator

Introduction

The weather generator (WG) developed as part of DARe's modelling and Climate Co-Centre's climate atlas framework produces synthetic time series of weather variables for a user defined length of period. The outputs from the weather generator can be employed in impact assessment models to evaluate risk associated with the climate impacts. The foundational papers for the developed weather generator are Burton et al., 2008 and Kilsby et al., 2007 which elaborate on the governing principles involved in building the weather generator.

The python scripts hosted on GitHub are built entirely on the Windows platform however the scripts should be executable on Linux and Mac environments. This document is intended to describe how to operate the python scripts.

Framework

The WG is a combination of two stochastic models which produces outputs for both single- and multi-sites. The first model is the rainfall generator and second is the non-rainfall variable generator. It is necessary to run the rainfall generator first followed by the non-rainfall generator as the simulation of non-rainfall variables is conditioned on whether a given day is rainy or non-rainy. Running the rainfall generator is a four step process as detailed below.

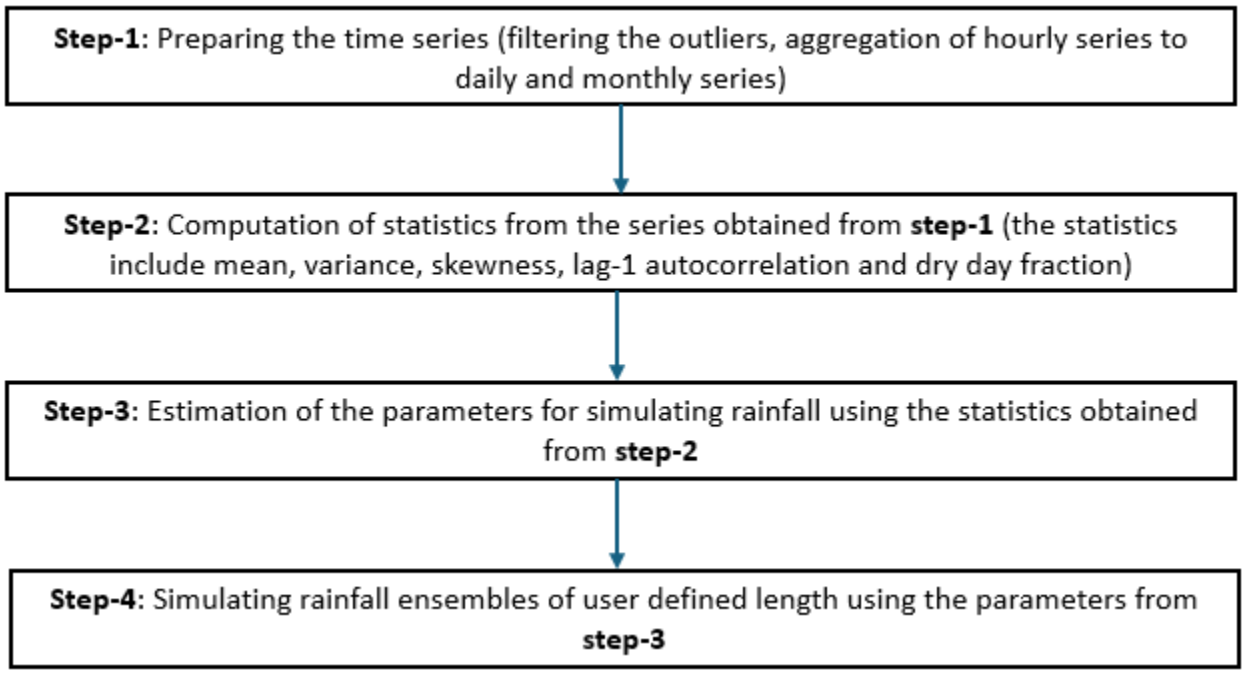

Figure-1: Steps involved in running the rainfall generator

Single-Site Rainfall Generator

The four step process is highlighted in the python scripts for both single-site and multi-site formulations. As an example the contents in **SingleSite/RainfallGenerator_SingleSite_Hourly_GaugeBased.py** is highlighted for a better understanding.

The functions corresponding to **Step-1** are **prepare_point_timeseries(), trim_array() and clip_array()**, these three functions take the data of rainfall at 1-hour resolution as the input and provide processed series with different temporal aggregations which include 24-Hour and 72-Hour and 1-month time series.

The function(s) corresponding to **Step-2** is **GetMonthStats()** which takes outputs from Step-1 (processed and aggregated series) to derive statistics on hourly, daily and monthly scales with statistics mentioned in Figure-1.

There are several functions in **Step-3** which are needed to derive the parameters of the rainfall model and the key function is **fit_by_month_point()** which returns the rainfall generator parameters (refer to Burton et al., 2008 for a detailed insight on the parameters) that are necessary to simulate rainfall realizations.

Similar to Step-3 several functions are involved in **Step-4** that are necessary to simulate multiple realizations of rainfall with **main_point_model_monthly()** and **discretise_by_point_monthly()** being the two important functions.

Input Format - the data format of the considered rain-gauge data as input is currently a '.csv' file containing two columns with first being the date in an appropriate format and second column corresponding to precipitation in 'mm'.

If the input data is not a gauge-based observation but instead a gridded based observation slight changes are needed in the script. These changes correspond to extraction of the data series and its processing which fall under **Step-1.** The gridded datasets of rainfall available for the United Kingdom are HadUK daily rainfall data from 1891 to 2024 and CEH-GEAR 1-hour rainfall data from 1990 to 2016 which come in NetCDF format. If the user desires to run the rainfall generator for these gridded datasets the the following python scripts should be used **SingleSite/RainfallGenerator_SingleGrid_Daily_HadUK.py and SingleSite/RainfallGenerator_SingleGrid_Hourly_CEHGEAR.py**

Single-Site Weather Generator

To run a single-site weather generator it is mandatory to first run the single-site rainfall generator which is embedded in these three scripts (**SingleSite/WeatherGenerator_SingleSite_HourlyRainfall_GaugeBased.py and SingleSite/WeatherGenerator_SingleGrid_DailyRainfall_HadUK.py**) depending on the selected input data and temporal resolution. The non-rainfall variables that are simulated include temperature (mean, minimum and maximum), wind speed, vapour pressure and sunshine duration at daily time scale. Note that rainfall generator provides simulations of rainfall at both 1-hour and daily time scales, however, a long record of the reference non-rainfall variables are available across UK at daily time scales hence the outputs are daily.

Running the single site weather generator is a seven step process which begins by running the single site rainfall generator, which as mentioned before is a four step process.

The **Step-5** involves preparation of the weather series which includes looking at the completeness of the weather data followed by transforming the weather series to standard normal variates. The key functions involved are **prepare_weather_series()** and **transform_series().** The output from transform_series() goes as input to Step-6.

Step-6 concerns estimation of regression parameters of the non-rainfall variables. This step is analogous to Step-3 of the rainfall generator shown in Figure-1 and the key function that derives parameters is **do_regression().**

The final step, **Step-7** takes the parameters and generates simulations of non-rainfall variables. The key function in this step is **simulate_daily_weather_point().**

Input data format - The input data is a '.csv' format file (Screenshot provided below).

Multi-Site Rainfall Generator

Similar to the single-site rainfall generator, running the multi-site rainfall generator is a four step process. The multi-site formulations provide outputs that are both point-based and grid-based (MultiSite/RainfallGenerator_MultiSite_Hourly_GaugeBased.py and MultiSite/RainfallGenerator_MultiSite_Hourly_GaugeBased_GridOutput.py) and care should be taken while feeding the input data.

The input data include a set of rain gauge stations within a considered region (which could be a catchment or city) with the description of metadata of the gauges shown below for both point and grid outputs and a Digital Elevation Model (DEM) specifically for a grid based output.

Table-1: Gauge metadata for multi-site rainfall generator

point_id	easting	northing	name	elevation	Latitude	Longitude
1	513056	213362	00471_rothamsted	90	51.807	-0.36
2	429124	206725	00605_brize_norton	78	51.758	-1.578
3	462370	191679	00613_benson	73	51.62	-1.099
4	421139	218140	00692_little_rissington	154	51.861	-1.693
5	507557	176752	00708_heathrow	31	51.479	-0.451
6	509749	184586	00709_northolt	40	51.549	-0.417
7	506209	158035	00719_wisley	30	51.311	-0.476
8	532989	157882	00726_kenley_airfield	152	51.304	-0.092
9	570682	157221	00744_east_malling	68	51.288	0.448
10	473640	149455	00862_odiham	113	51.239	-0.945
11	485569	154200	00869_south_farnborough	76	51.28	-0.773
12	482470	198863	17176_high_wycombe_hqair	162	51.682	-0.807

The key function(s) involved in Step-1 of the multi-site rainfall generator is **prepare_spatial_timeseries()** which is a wrapper to the function **prepare_point_timeseries().**

For **Step-2** which is the computation of reference statistics, a few additional functions are included apart from **GetMonthStats()** which are **GetPooledMonthStats(), GetCrossCorrel()** and

GetPoolCrossCorrel(). The key functions for single-site and multi-site remain the same for **Step-3.**

Finally for **Step-4, main_sim()** is the important function which produces realizations of rainfall.

Multi-Site Weather Generator

The multi-site weather generator (**MultiSite/WeatherGenerator_MultiSite_Hourly_GaugeBased_GridOutput.py**) requires executing the commands relevant to the multi-site rainfall generator which is followed by running the multi-site non-rainfall generator. While the multi-site weather generator asks for two metadata files, the user can provide the same file as seen in Table-1. Note that currently the multi-site weather generator produces 'point' outputs

The key functions in the multi-site non-rainfall generator include **process_stations()** and **preprocess()** which produce the processed input spatial weather series and transformed series (normal variates) respectively. Let's call this **Step-5,** considering the four steps of the multi-site rainfall generator.

In the **Step-6, do_regression()** is the key function which produces parameters to run the non-rainfall generator.

In the final stage, **Step-7** it is essential to capture the spatial association of the weather variables and to that end variograms are used. Keeping this in view, the following functions are key in this step: **estimate_statistic_variograms(), estimate_r2_variograms(), estimate_se_variograms(), interpolate_parameters_space2()** and **simulate_daily_weather_spatial()**

*** Input data are provided for the users to test all these scripts**

Future Development

The python scripts will be further developed with a view to consolidating these in a python package, to increase the flexibility and usability allowing users to run the WG more efficiently.

If you find any bugs or errors while testing the python scripts or if you want to provide any comments and suggestions please reach me at azhar.mohammed@newcastle.ac.uk