User Manual

Generation Adequacy Analysis Scripts

Julian Wuijts

July 2024

In this document, some examples will be given on how the scripts can be used to replicate the results that are presented in my thesis. It can be used alongside the thesis and readme file to understand how the scripts work and can be adapted.

Analytical Method

COPT_derated_states.py

A COPT can be obtained using this script. There are two functions, the remove_duplicates and calc_COPT functions. The remove_duplicates function is used to remove duplicate outage levels in the recursive process for obtaining a COPT.

The calc_COPT function takes an Excel file as an input, which has the generator data. The path to the Excel file can be specified in the following line of code:

generator_data = pd.read_excel(“”)

Make sure to change the backslash (\) to slash (/) when copying the path in Windows

The Excel file could look something like the one shown below, where the first capacity outage level of the first generator should be defined in cell B3 due to the way that the data is processed within the calc_COPT function. Additional outage levels can be defined in the same column and the associated state probability can be defined in the next column.

	A	B	C	D	E
1	Generator number	1		2	
2		Capacity outage [MW]	State probability	Capacity outage [MW]	State probability
3		0	0.99	0	0.99
4		5	0.01	5	0.01
5					

Selecting the COPT_example.xlsx file results in a COPT that matches the one found in [2], while the COPT_RBTS.xlsx file will give the COPT for the Roy Billinton Test System, which can be found in [3].

COPT_wind.py

A COPT for wind generation can be obtained using this script. It again uses the remove_duplicates function to remove duplicate outage levels. The power output of a wind turbine can be computed using the CreatePowerProfile function. Here the wind speeds and turbine characteristics can be defined in an Excel file by importing the following data:

Wind_speed_excel = pd.read_excel("PATH", sheet_name="Wind_speed")

Wind_speed_test = pd.read_excel ("PATH", sheet_name="Wind_speed_power_curve")

Turbine_power_profile_excel = pd.read_excel("PATH", sheet_name="Turbine_power_profile")

In this case, Wind_speed_excel has the actual wind data that is being used, and Wind_speed_test is a range from 1 until 30 m/s in order obtain the output power curve of a wind turbine. The turbine specifications are imported with Turbine_power_profile_excel. The excel files look as follows:

	A	B
1	metadata: {'mean_wind_speed': '8.94 (IEC II - Medium Wind)'	'lat': '64.0'
2		
3	datetime	wind_speed
4	01/01/2019 00:00	13.35
5	01/01/2019 01:00	12.74
6	01/01/2019 02:00	13.97
7	01/01/2019 03:00	13.9
8	01/01/2019 04:00	14.195
9	01/01/2019 05:00	16.17
10	01/01/2019 06:00	23.53

	A	B
1		
2		
3		wind_speed
4		0
5		1
6		2
7		3
8		4
9		5
10		6

	A	B	C
1		Turbine model	Vestas V90
2		Rated power [MW]	10
3		Cut-in wind speed [m/s]	4
4		Rated wind speed [m/s]	11.4
5		Cut-out wind speed [m/s]	25

The wind speeds can be defined in column B, starting from row 4.

The mechanical availability of the turbine can be define in the following section of the script:

```
### Roy Billinton example, uncomment to get the COPT for this example ###
number_of_wind_output_states = 5
FOR = 0.04
number_of_turbines = 10
P_rated = 2 # in MW
#turbine_power_profile_probability_test = [0.76564,0.23436] # 2
turbine_power_profile_probability_test = [0.50897,0.24450,0.11688,0.05944,0.07021] #5
#turbine_power_profile_probability_test = [0.40073,0.15193,0.10843,0.007985,0.05969,0.04503,0.03427,0.02629,0.02021,0.01560,0.05796] #11
```

The defined turbine_power_profile_probability_test is a hardcoded version of the power output in [1] as the used wind speeds are not known. The mechanical availability is then combined with the wind power output availability in the section below. For now, the hardcoded power profile is used, but this can be changed by swapping turbine_power_probability test for State_probability_list. To reduce the number of states in the COPT, the apportioning method can be used. The desired states can be specified in the variable called 'Apportioned_states'. The resulting COPT and power curves are printed and plotted at the bottom of the script.

Load_curves.py

This script creates the IEEE load curve in 8736 hourly increments, with the profiles for a yearly, weekly, daily and hourly peak load defined in the Load_curves.xlsx file. The right path to this file should be specified in the following code:

WPL_excel = pd.read_excel("PATH", sheet_name="WPL")

DPL_excel = pd.read_excel("PATH", sheet_name="DPL")

HPL_excel = pd.read_excel("PATH", sheet_name="HPL")

The Yearly Peak Load value should be changed based on the system and the chronological load curve as well as the load duration curve, with the load increments in descending order will be plotted.

Reliability_indices.py

The Reliability metrics LOLP, LOLE, EENS with the calc_LOLP, calc_LOLE and calc_EENS functions. The COPT and Load curve functions are imported from the previously mentioned scripts, and the generator data and YPL are again defined. The values for LOLP' and ENDS are derived as well from the LOLE and EENS values.

A load curve with a different number of hourly increments can be used as well by changing the 'time' variable, which is currently defined as an array from 1 until and including 8736.

Non-Sequential MCS Method

MCS_state_sampling_no_derated.py

A generation profile will be obtained in this script for a system without derated states. This is done to speed up the computational time. The script has various functions, for example generate_random_values, generate_wind_speed_list and generate_generator_profile_state to create a generator state and wind speed for each time increment. The number of time increments can be altered by changing the value of 'total_hours'. This data is then combined in the generation_profile function, where data is again imported from an Excel file with the following layout. Please use the same cells when using a different generation system to ensure that the code runs smoothly. Otherwise, one must change the indexing of the variables from the Excel data.

	A	B	C	D	E
1	Generator number	1		2	
2		Capacity outage [MW]	State probability	Capacity outage [MW]	State probability
3		0	0.99	0	0.99
4		5	0.01	5	0.01
5	generator type	0		0	
6	number of turbines	1		1	
7					

If a generator has multiple of the same units, they can be specified in the number of turbines row of the Excel file. The generator type is used to specify whether a generator is a wind turbine or not. Zero is used for conventional generation units and one for a wind turbine.

MCS_state_sampling_derated.py

The script for creating a generation profile with derated states is similar to the one without derated states, but with a small difference in the state selection process. The generate_generator_state_profile function now not only checks if the randomly generated number U is above or below a certain threshold, but it looks at the state probability intervals. All probabilities are summed together for a cumulative probability distribution, and the interval in which the random number falls is selected. A percentage of the installed capacity is return based on the selected state. The generator data has a slightly different layout in Excel, along with the number of total possible states

	A	B	C
1			
2	generator type	0	
3	number of generators	2	
4	number of states:	Capacity outage [MW]	State probability
5	3	0	0.99
6		5	0.01

L	M
0	
2	
Capacity outage [MW]	State probability
0	0.96
20	0.02
40	0.02

Load_curves.py

The load curves for the MCS method are the same as for [2] the Analytical method.

Reliability_indices_MCS.py

The Reliability metrics LOLE and EENS can be found for the non-sequential MCS method by combining a load curve with a generation profile. Generator data, and if applicable, turbine data can again be imported from Excel using pd.read_excel(). Within the file there is the possibility to either use the function without derated states or the one with derated states. The stopping criteria is the number of simulation years, and this number can be changed based on the desired coefficient of variation.

Bibliography

[1] R. Billinton and Y. Gao, “Multistate Wind Energy Conversion System Models for Adequacy Assessment of Generating Systems Incorporating Wind Energy,” *IEEE Transactions on Energy Conversion,* vol. 23, no. 1, pp. 163-170, 2008.

[2] R. Billinton and R. N. Allan, Reliability Evaluation of Power Systems, Boston, MA: Springer US, 1996.

[3] R. Billinton, "Test Systems for Reliability and Adequacy Assessment of Electric Power Systems," January 2010. [Online]. Available: https://www.researchgate.net/publication/260402891_Test_Systems_for_Reliability_and_Adequacy_Assessment_of_Electric_Power_Systems.