The MATLAB Toolbox for EnergyPLAN: a tool to extend energy planning studies

Pedro Cabrera[a] [1], Henrik Lund[b], Jakob Zinck Thellufsen[b], Peter Sorknæs[b]

[a] Department of Mechanical Engineering, University of Las Palmas de Gran Canaria, Campus de Tafira s/n, 35017 Las Palmas de Gran Canaria, Canary Islands, Spain.

[b] Department of Planning, Rendsburggade 14, Aalborg University, Denmark.

Abstract

EnergyPLAN is an energy system analysis tool used worldwide for scientific analyses of national and regional energy planning strategies and alternatives. This work describes the MATLAB Toolbox for EnergyPLAN (MaT4EnergyPLAN), a set of functions developed to manage the EnergyPLAN software using MATLAB. The tool allows the user to take advantage of the energy system analysis capabilities of EnergyPLAN in combination with the computational advantages of MATLAB. This allows the user to easily manage EnergyPLAN files and analyse a large number of EnergyPLAN simulations.

Keywords: Energy planning; smart energy system; renewable energy strategies; EnergyPLAN; MATLAB.

1. Introduction

In the transition towards future energy supply systems, a number of strategic issues arise for national and local governments in the world. These involve the impact of the supply systems on energy security and the environment, and their influence on economic and developmental goals. It is commonly accepted that this inevitable transition should consider the utilisation of energy sources with less environmental impacts and more local availability [1]. Renewable

[1] Corresponding author. tel.: +34 928 45 98 87; fax: +34 928 45 14 84.

E-mail address: pedro.cabrerasantana@ulgpc.es (Pedro Cabrera).

energy sources (RES) play an important role in attaining these global aims. According to [2], a set of environmental, socio-economic, and ethical sustainability aims can be satisfied in the most wide-ranging manner in 100% RES-based system scenarios. However, in the transition to 100% renewable energy systems, the intermittent nature of RES, the great diversity of subsystems, their different operation modes, and possible interconnections between energy sectors present challenges for the planners of future energy systems. In this context, energy system models are crucial tools to support decision makers and to guarantee feasible future energy system designs.

Currently, a wide range of computer tools allow users to model and analyse energy systems from different points of view and with different purposes [3]. Connolly et al. [4] scrutinized 37 energy models available in 2010 and concluded that there is no 'ideal' tool in this matter. Decision makers and researchers should choose the most suitable energy tool depending on their specific objectives. For example, only five tools are recommended to optimise the energy system to accommodate the fluctuations of renewable energy —including EnergyPLAN [4]. The EnergyPLAN model [5], with around 349 references in Scopus and 961 results in Google Scholar, is one of the most popular choices for academic studies within the field of energy systems analysis. EnergyPLAN is one of the most commonly used tools for the evaluation of high-RES energy systems [1] and some authors consider it the most suitable tool to identify a feasible RES integration within an energy system [1,6,7]. This tool uses a deterministic procedure to simulate the electricity, heating/cooling, desalination, transport and gas sectors in the energy system and to investigate the synergies between sectors by using an hourly approach [8]. The engine of EnergyPLAN is programmed and maintained in Delphi Pascal by the Sustainable Energy Planning Research Group (SEPRG) at Aalborg University. The manuals, reports, and algorithm descriptions behind the tool can be consulted on the website www.energyplan.eu. As general inputs, the software uses the energy demand, renewable energy sources, energy station capacities, costs and a number of optional regulation strategies [8]. Among other outputs, the tool returns the energy balance and resulting annual, monthly and hourly energy productions, fuel consumptions, electricity import/export and total annual carbon dioxide emissions.

2. Problems and background

EnergyPLAN has a user-friendly interface, which easily allows the user to run a limited number of subsequent executions varying a limited number of decision variables for a concrete modelled energy system. Some authors, such as van Beukekom et al. [9], Bjelić and Rajaković [10] or Mahbub et al. [11] highlighted the manual mode in which EnergyPLAN combines the optimization of both the operational phase and the planning phase. Most of them underlined the need to combine EnergyPLAN with other computational tools in order i) to be able to identify a set of future energy scenarios taking into account multiple criteria and using a multi-objective evolutionary algorithm (MOEA) [11]; ii) to minimize the total cost of an energy system scenario, as for instance in Serbia [10]; or iii) to include energy efficiency measures in the analyses and to also use a MOEA approach [12]. However, EnergyPLAN can be called and run externally by a command line specified in [13]. This possibility allows EnergyPLAN to be combined with whatever other computational framework that can deliver these desired functions.

In this paper, a set of functions wrapped in a toolbox designed to call and manage EnergyPLAN from MATLAB is presented. The aim is to take advantage of the high-level technical computing possibilities offered by the MATLAB environment to increase the potential of the EnergyPLAN model within the field of energy planning.

3. Software framework

The EnergyPLAN developers offer the user the possibility of operating the tool based on a command line call executed from the computer's operating system (OS). This call requires two parameters, one specifying an input file path and one specifying an output file path (`-i [inputfile]` and `-ascii [outputfile]`). Hence, EnergyPLAN requires i) the path of a previously created energy scenario input file, which contains the basic information structure and data of the energy system under study, and ii) a path where output results obtained after the simulation will be saved. Some valid command line call examples are presented in [13] assuming that the EnergyPLAN executable file "`energyPLAN.exe`" is located in the "`C:\kt\EnergyPLAN\run\`" path folder. The way to call these examples is presented below [13]:

- Get input file from standard data folder:

```
C:\kt\EnergyPLAN\run\energyPLAN.exe -i Inputfile.txt -ascii c:\MyResultFolder\MyResult.txt
```

- Get input file from subfolder in the standard data folder:

```
C:\kt\EnergyPLAN\run\energyPLAN.exe -i \MyInputFolder\Inputfile.txt -ascii c:\MyResultFolder\MyResult.txt
```

- Get input file from fully specified path+file:

```
C:\kt\EnergyPLAN\run\energyPLAN.exe -i c:\MyInputFolder\Inputfile.txt -ascii c:\MyResultFolder\MyResult.txt
```

3.1. Software architecture

The main goal of MaT4EnergyPLAN is not only to do valid calls of EnergyPLAN, but also to allow the users to make subsequent changes and configurations of an original definition of an energy system (input file) depending on a concrete analysis idea. This new development allows the user to modify input files before the execution of the EnergyPLAN model and to access the results from the MATLAB environment. With these capacities, MATLAB can make decisions on whether more/new executions are needed or not. The continuous and progressive development of MaT4EnergyPLAN since 2017 has simplified several MATLAB functions, achieving a final wrapped package composed of the three fundamental functions shown in Fig. 1.

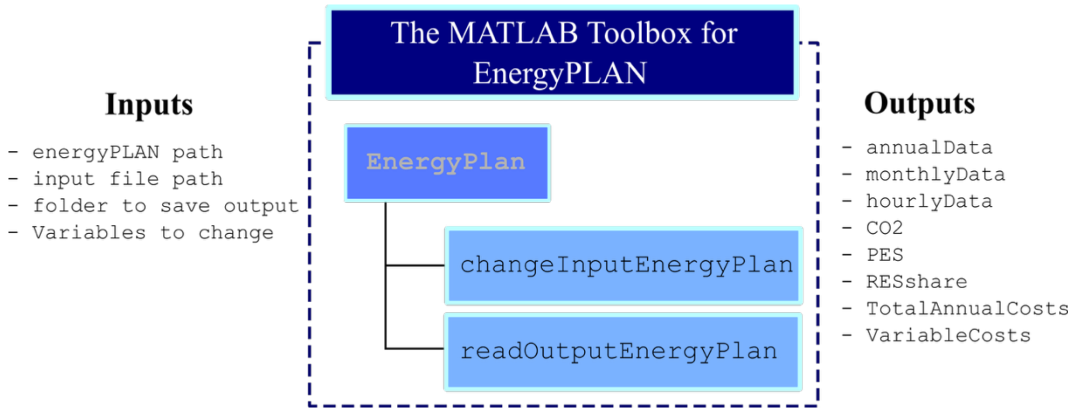

Fig. 1. Diagram of the MATLAB Toolbox for EnergyPLAN.

1. **`changeInputEnergyPlan`**`(inputFilePath, newInputFilePath, varargin)`**:**

   Description: This function changes the EnergyPLAN variables (indicated by `varargin` with two arguments) of an EnergyPLAN input file placed in a file path (`inputFilePath`*)* and generates a new EnergyPLAN input file placed in another file path (`newInputFilePath`).

   Inputs: The types and characteristics of the input parameters to this function are the following:

   – `inputFilePath`, a string which specifies the path where the EnergyPLAN input file (scenario file) is located.

– `varargin`, in MATLAB, a special input type of parameter in a function definition statement that enables the function to accept any number of input arguments. In this case, `varargin` contains - in pairs - the names of the variables to change in an EnergyPLAN scenario (input file) followed by their new values.

Example: In Listing 1, an example of input parameter definitions to call the **`changeInputEnergyPlan`** function is presented:

```
inputFilePath = 'energyPlan Data\Data\Denmark2030Alternative.txt';
newinputFilePath = 'energyPlan Data\Data\Denmark2030Alternative_temp.txt';
changeInputEnergyPlan (inputFilePath, newinputFilePath,...
       'input_RES1_capacity=', 3500, 'input_RES2_capacity=', 2000);
```

Listing 1: Example of `changeInputEnergyPlan` function use.

2. `[annualData, monthlyData, hourlyData, CO2, PES, RESshare, TotalAnnualCosts,VariableCosts]=`**`readOutputEnergyPlan`**`(outputFilePath):`

Description: This function reads the annual data results, the monthly data results, and the hourly data results of an EnergyPLAN execution. It works with a file saved in the file path defined by `outputFilePath`. `annualData`, `monthlyData`, `hourlyData`, `CO2`, `PES`, `RESshare`, `TotalAnnualCosts` and `VariableCosts` are the parameters returned by the function **`readOutputEnergyPlan`**.

Inputs: The types and characteristics of the input parameter for this function are the following:

– `outputFilePath`, a string which specifies the path where the EnergyPLAN output file is located (file with results of an EnergyPLAN model execution).

Outputs: The types and characteristics of the output parameters of this function are the following:

– `annualData`, an array of data composed of the 1x142 double-type annual values and available in the MATLAB environment and generated in the EnergyPLAN execution.

- `monthlyData`, an array of data composed of the 12x142 double-type monthly values and available in the MATLAB environment and generated in the EnergyPLAN execution.
- `hourlyData`, an array of data generated by the EnergyPLAN execution and composed of the 8784x142 double-type hourly values which are available in the MATLAB environment. Although most models use 8760 values, EnergyPLAN is designed to use 8784 hourly data (because it considers all years as leap years).
- `CO2`, a double-type numeric value with the total CO_2 emissions calculated by EnergyPLAN.
- `PES`, a double-type numeric value with the total fuel consumption of the Energy System (Primary Energy Supply (PES)) calculated by EnergyPLAN.
- `RESshare`, a double-type numeric value with the RES share of PES calculated by EnergyPLAN in the analysed energy system.
- `TotalAnnualCosts`, a double-type numeric value with the total annual costs calculated by EnergyPLAN.
- `VariableCosts`, a double-type numeric value with the variable costs calculated by EnergyPLAN.

Example: An example of an input parameter definition and a **`readOutputEnergyPlan`** function call is presented in Listing 2:

```
outputFilePath = 'Outputs\out_Denmark2030Alternative.txt';
[annualData,monthlyData,hourlyData, CO2, PES, RESshare,...
      TotalAnnualCosts, VariableCosts]=readOutputEnergyPlan(outputFilePath);
```

Listing 2: Example of `readOutputEnergyPlan` function use.

3. `[annualData, monthlyData, hourlyData, CO2, PES, RESshare, TotalAnnualCosts,VariableCosts]=`**`energyPlan`**`(energyPlanPath, inputFilePath, outputFolder,varargin)`

Description: This function calls the energyPLAN.exe file placed in the `energyPlanPath`, runs an execution of the input scenario indicated by the `inputFilePath`, and saves the results of the execution in the output folder indicated by the `outputFolder`. The output file is named the same as the input file but with the string “`out_`” at the beginning of the name.

The EnergyPlan function wraps the two functions described above (**changeInputEnergyPlan** and **readOutputEnergyPlan**). Hence, as in the function **changeInputEnergyPlan**, it is possible to directly change the EnergyPLAN variables (indicated by `varargin` with two arguments) of an EnergyPLAN input file placed in a file path (`inputFilePath`). However, in this case, MATLAB generates a temporary new EnergyPLAN input file, which is removed when the execution results are obtained. Furthermore, inside this function, the annual, monthly, and hourly data results of an EnergyPLAN execution are read and saved in a file placed in the file path defined by `outputFolder`. In this case, as with the function **readOutputEnergyPlan**, MATLAB saves the execution results with the same name as the input file but with the string "`out_`" at the beginning of the file name.

Inputs: The types and characteristics of the input parameter for this function are the following:

- `energyPlanPath`, a string which specifies the path where the EnergyPLAN execution file ("`energyPLAN.exe`") is located.
- `inputFilePath`, a string which specifies the path where the EnergyPLAN input file (scenario file) is located.
- `outputFilePath`, a string which specifies the path where the EnergyPLAN output file (file with results of an EnergyPLAN model execution) is located.
- `varargin`, in MATLAB, a special input type of parameter in a function definition statement that enables the function to accept any number of input arguments. In this case, `varargin` contains - in pairs - the names of the variables to change in an EnergyPLAN scenario (input file) followed by their new values.

Outputs: The types and characteristics of the output parameters of this function are exactly the same as those of the previous function (**readOutputEnergyPlan**).

Example: Listing 3 presents an example of input parameters definition to call the **changeInputEnergyPlan** function:

```
%% Paths and Folder definitions
%Path where executable EnergyPLAN is located.
energyPlanPath = ' C:\kt\EnergyPLAN\run\energyPLAN.exe';
%Path of input file (scenario to simulate).
inputFilePath = 'energyPlan Data\Data\Denmark2030Alternative.txt';
%Folder where results of EnergyPLAN execution will be located.
outputFolder = 'Outputs\';
```

```
%% EnergyPLAN calling and parameters/values definitions
% Definition of numeric values for reference input file parameters.
OnshoreWind=3500; %New value for onshoreWind capacity in the model.
OffshoreWind=3100; %New value for offshoreWind capacity in the model.
STR='23457000'; %Values for operation strategies.

%EnergyPLAN calling from MATLAB.
[annualData, monthlyData,hourlyData, CO2, PES, RESshare, TotalAnnualCosts,
VariableCosts] = energyPlan(...
       energyPlanPath, inputFilePath, outputFolder,...
       'input_RES1_capacity=', OnshoreWind,...
       'input_RES2_capacity=', OffshoreWind,...
       'input_keol_reg=', STR);
```

Listing 3: Example of `energyPlan` function use.

3.2. Description of the variables that can be changed in EnergyPLAN

Each EnergyPLAN scenario (energy system model) is defined by an input text file which currently contains 726 variable definitions. A sample of the complete list of variables, along with a brief description of these, is provided in Table 1.

Table 1. Sample of the list of variables which define an EnergyPLAN scenario and which can be changed using MATLAB.

#	Name	Type	Example of use	Short description
1	'EnergyPLAN version'	double	698	EnergyPLAN version
2	'EnergyUnit='	char	'TWh/year'	Energy unit used in scenario
3	'CapacityUnit='	char	'MW'	Capacity unit used in scenario
4	'MonetaryUnit='	char	'DKK'	Monetary unit used in scenario
5	'Input_el_demand_Twh='	double	29.84	Annual electricity demand in Twh
6	'Input_El_demand_elec_heating_share='	double	0	Electric heating to be subtracted from electricity demand.
7	'Input_El_demand_cooling_share='	double	0	Electric cooling to be subtracted from electricity demand.
8	'input_dh_demand_TWh='	double	21.21	Annual District Heating demand TWh
9	'input_RES1_capacity='	double	3000	RES1 capacity
10	'input_RES2_capacity='	double	3000	RES2 capacity
	⋮	⋮	⋮	⋮
51	'Filnavn_cshp='	char	'const.txt'	Text-file name with the hourly distribution of industrial CHP heat delivered.
52	'Filnavn_pv='	char	'hour_PV_eltra2001.txt'	Text-file name with the hourly distribution of PV power production.
53	'Filnavn_RES4='	char	'Hour_wave_2001.txt'	Text-file name with the hourly distribution of RES4 power production.
54	'filnavn_hydro_water='	char	'Hour_wind_1.txt'	Text-file name with the hourly distribution of hydro water power production.
55	'filnavn_nuclear='	char	'Hour_wind_1.txt'	Text-file name with the hourly distribution of nuclear power production.
56	'input_transport_TWh='	double	0.98	Annual electricity demand for transport (dump charge).
57	'input_transport_TWh_V2G='	double	0	Annual electricity demand for transport (V2G).
58	'Filnavn_transport='	char	'Hour_transport.txt'	Text-file name with the hourly distribution of transport (dump charge)
59	'Filnavn_transport_V2G='	char	'Hour_transport.txt'	Text-file name with the hourly distribution of transport (V2G)
	⋮	⋮	⋮	⋮
718	'input_Period_Boilers_dh_gr1='	double	0	Lifetime of investment for boilers district heating group 1
719	'input_FOM_Boilers_dh_gr1='	double	0	Fixed operational and maintenance costs (in percentage of investment) for boilers district heating group 1

720	'ThermalStorageDays='	double	14	Thermal storage days
721	'input_HH_EB_CapLimit='	double	1	HH electric boilers capacity limit
722	'input_CCS_El_PerUnit='	double	0.37	CCS electricity per unit
723	'input_CCS_Regulation_number='	double	1	CCS regulation number
724	'Input_CCS_Capacity='	double	0	CCS capacity
725	'input_Heatdemand_PerHouse='	double	15000	Heat demand per house
726	input_Inv_Electric Boilersgr2gr3='	double	0	Investment costs for electric boilers group 2 and group 3

4. Implementation and empirical results

The primary functionality of the MaT4EnergyPLAN is to manage EnergyPLAN executions by using MATLAB enabling the interconnection of the two tools and allowing complete access to all options of both tools when performing an energy planning study. The requirement for using this toolbox is the installation of MATLAB and EnergyPLAN.

4.1. Installation

To use the MaT4EnergyPLAN in MATLAB, the package must be downloaded at https://www.energyplan.eu/ and installed during a MATLAB session. After downloading and installing the package, the MaT4EnergyPLAN functions, including **`changeInputEnergyPlan`**, **`readOutputEnergyPlan`** and **`energyPlan`**, will be available in the MATLAB environment. The following is a step-by-step description of the installation process:

1. Download the toolbox from https://www.energyplan.eu/useful_resources/matlab-toolbox-for-energyplan/. After downloading, a ".zip" file will be available with the following four items:

    a. README.txt file, with contact information and a suggested way to quickly test the toolbox.

    b. Help.html file, with a simple description of how to use the **`energyPlan`** function.

    c. Example folder, with three simple MATLAB files to use the functions developed in the MATLAB Toolbox for EnergyPLAN.

    d. EnergyPLAN Toolbox.mltbx, is the MaT4EnergyPLAN file, which includes all the functions of the toolbox in one package.

2. Install the MaT4EnergyPLAN by double-clicking the file "EnergyPLAN Toolbox.mltbx". If this procedure does not work directly it is because the OS has not associated mltbx type files to the MATLAB interpreter by default. In this case, it is necessary to look for the EnergyPLAN Toolbox.mltbx file in the Current Folder

Browser available inside the MATLAB environment. After this process, the MaT4EnergyPLAN will be introduced in the list of Add-ons and a pop-up message will appear reporting that the installation has been successfully completed.

4.2. Using the toolbox to develop smart renewable energy penetration strategies on islands

This section describes how to use the MaT4EnergyPLAN to implement the method proposed in Cabrera et al. [14]. This method finds the optimal renewable configuration in an analysed energy system scenario with the goal of minimising the need for thermal plant electricity production and the critical excess electricity production due to the surplus of renewables produced when there is insufficient demand. For this optimization, MATLAB establishes in each iteration a new candidate energy system with renewable generation for the desired optimal renewable configuration. It then launches and executes the scenario in EnergyPLAN, which carries out an internal optimization of the operation of this candidate energy system based on a predetermined strategy (integration of the largest amount of renewables possible, ordering generation by cost, etc.) and simulates the results obtained. A schematic representation of this method is shown in Fig. 2, and a summary of the different stages is presented below:

1. The reference energy system is identified. The official data, statistics and reports that are available are analysed in detail. The demands in different energy sectors, the potential modifications and potentially exploitable energy supply sources in each of these are documented.
2. The results of the model are validated. The results obtained from EnergyPLAN and data found in official reports are cross-checked to validate the model.

3. New alternatives are proposed in each sector of the energy system using the concepts proposed by Cabrera et al. [14].

4. A RES configuration is identified for each alternative, looking for a minimum intersection point between fossil fuel requirements for thermal plant electricity production (imports) and energy surpluses (exports).

5. When step 4 is completed, step 3 is applied to a new energy sector and new alternatives are proposed and evaluated.

6. The results are analysed, and the advantages and disadvantages of the alternatives are compared.

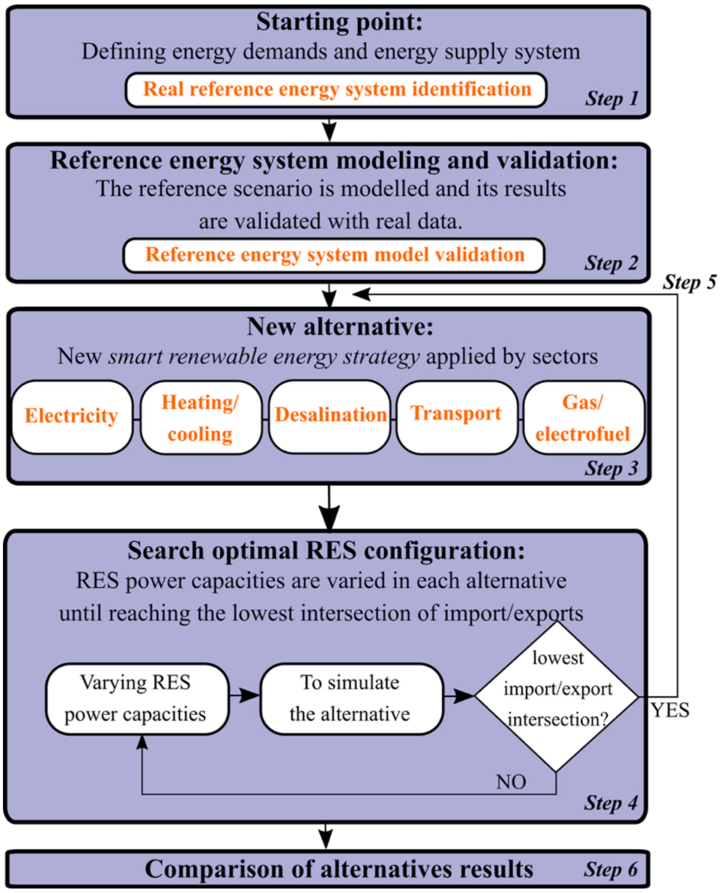

Fig. 2. Outline of the proposed approach to increase renewable energy penetration on islands using the Smart Energy Systems concept.

The use of MaT4EnergyPLAN is essential to carry out the procedure defined in step 4. With MATLAB it is possible to determine the minimum intersection point between imports (fossil fuel energy needs for thermal plants) and exports (excess electricity production-EEP) in each scenario, when wind and photovoltaic (PV) capacities are increased sequentially. The procedure is based on the optimal increment of wind and photovoltaic power capacities to satisfy the total electricity demand of the reference model. Both power capacities are varied N+1 times from their actual values in a chosen year to the corresponding values which satisfy the total electricity demand. For each PV power capacity value, each wind power capacity value is executed in EnergyPLAN from MATLAB, obtaining the N+1 import/export results for each analysed PV configuration. In reference [14] (Fig. 15), an example is shown of a sequential search for the minimum intersection between imports and exports for the energy system of the island of Gran Canaria.

The implementation of this procedure in MATLAB is applied to the 'Denmark2030Alternative.txt' input file. To analyse this energy system scenario as an isolated system (as an island), the following steps are required:

i) a definition of three paths (Listing 4):

   a. where the EnergyPLAN exe-file is placed.

   b. where the original input file is currently placed.

   c. where the output folder will be created and the optimal output file will be saved.

```
%% Paths and Folder definitions
%Path where executable EnergyPLAN is located.
energyPlanPath = ' C:\kt\EnergyPLAN\run\energyPLAN.exe';
%Path of input file (scenario to simulate).
inputFilePath = 'energyPlan Data\Data\Denmark2030Alternative.txt';
%Folder where results of EnergyPLAN execution will be located.
outputFolder = 'Outputs\';
```

Listing 4: Example of path definitions.

ii) a definition of the two variables which will be changed during the exploration. In this case, it uses the onshore wind capacity and PV capacity installed in the system (Listing

5). The maximum value of each is defined as the value needed to cover 100% of the electricity demand. Hence, as in subsection 3.1 (Listing 3), two vectors are defined in Listing 5:

a. one vector for wind capacities, with the actual wind power capacity and ten more linear-spaced values.

b. another vector, with the actual PV power capacity installed in the system and ten more linear-spaced PV capacity values.

```
%% Vectors definitions
%The Total Electricity Demand is defined.
totalElectricityDemand = 29.84;
%The installed power capacity needed to cover the electricity demand only
with wind or PV is defined.
wind100=12745;
pv100=29780;
%The number of divisions in both vectors is defined.
div=10;
%Two linear-spaced vectors are defined.
wind=linspace(wind100/div, wind100, div);
pv=linspace(pv100/div, pv100, div);
```

Listing 5: Example of vector definitions.

Additionally, in this case:

iii) a search is carried out to explore all possible configurations of installed PV (computed by the first for-loop presented in Listing 6) and wind power capacities (computed by the second for-loop in Listing 6). In both these for-loops:

a. the executions of EnergyPLAN are launched with each possible PV-wind configuration.

b. the different calculations between exports and electricity production (imports in an isolated system) are made.

c. the results (of wind and PV capacities, the values of imports, exports, maximum hourly import, maximum hourly export and the import/export difference) are saved in the cell type variable ‘r’.

```
r{length(wind)}=[];
rWind(length(wind),7)=0;

 for i=1:length(pv)
       for j=1:length(wind)
       [annualData, ~, hourlyData, ~,~,~,~,~]=energyPlan(energyPlanPath,...
```

```
        inputFilePath, outputFolder,...
        'input_RES1_capacity=', wind(j), 'input_RES2_capacity=', pv(i));

        export = hourlyData(:,94);
        import = hourlyData(:,60) + hourlyData(:,59) + hourlyData(:,58) +...
                 hourlyData(:,56) + hourlyData(:,57);

        dif=abs(sum(export)-sum(import));

        rWind(j,:)=[wind(j), pv(i), sum(import), sum(export), max(import),...
                    max(export), dif];
        end
        r{i}=rWind;
    end
```

Listing 6: Exploring all possible PV and wind configurations.

iv) a new search to find the minimal intersection between imports and exports (cross points) is carried out in Listing 7.

v) the results are plotted using the typical plot commands in MATLAB.

```
%% Find minimum values of the difference between imports and exports (cross
points)
crossPoints(length(r),7)=0;

for i=1:length(r)
    [~,f]=min(r{i}(:,7));
    crossPoints(i,:)=r{i}(f,:);
end

crossPoints(:,8)= crossPoints(:,1)/wind100*100;
crossPoints(:,9)= crossPoints(:,2)/pv100*100;
```

Listing 7: Search for the minimum values of the difference between imports and exports (cross points).

5. Illustrative examples

The following examples illustrate the options that the MaT4EnergyPLAN adds to the MATLAB code. With the aim of executing EnergyPLAN from MATLAB in a quick and understandable way, some example m-files are supplied in the compressed toolbox file:

5.1. Example 1: A first execution to launch EnergyPLAN with an m-file template supplied with the toolbox.

The first example uses the Denmark 2030 Alternative scenario – which can be found in the EnergyPLAN zip-file when the model is downloaded from www.energyplan.eu. The code in the example1.m file (shown in Fig. 3) defines:

i) three paths:

a. where the EnergyPLAN exe-file is placed.

b. where the original input file is currently placed (Denmark2030Alternative.txt file with the scenario of the Danish energy system for 2030 defined by 726 variables).

c. where an output folder will be created with the saved output file (new txt-file generated after the EnergyPLAN execution).

ii) three variables which will be changed.

a. onshore wind capacity.

b. offshore wind capacity.

c. strategy to regulate the critical excess electricity production (CEEP).

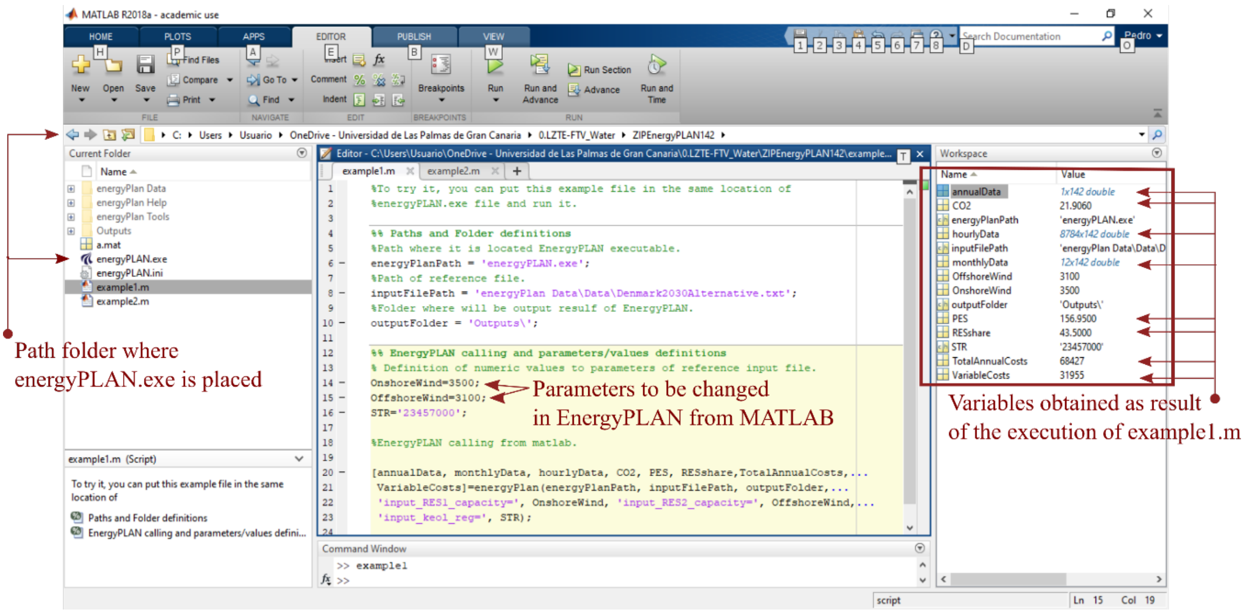

Fig. 3. MATLAB environment with the example1.m m-file opened and the results of Denmark2030Alternative.txt scenario returned after the execution of EnergyPLAN from MATLAB.

This file should be placed in the same path folder as the EnergyPLAN exe-file. After ordering execution of the m-code in MATLAB (clicking on the Run button), EnergyPLAN will be launched and the Denmark2030Alternative.txt scenario will be analysed. Once EnergyPLAN finishes the simulation, the annual, monthly, hourly, total CO_2 emissions, total fuel consumption (PES), RES share of PES, total costs and variable costs results will be available in the MATLAB environment. Additionally, an output file named "out_Denmark2030Alternative.txt" will be placed in a new folder created by MATLAB. This file contains the results of the analysis performed by EnergyPLAN. The name of the folder is "Outputs" and it is in the same path as energyPLAN.exe.

It can be seen in Fig. 3 how to interact with the MATLAB environment in order to use the functions of the MAT4EnergyPLAN Toolbox. The drop-down menu seen on the left-hand side of Fig. 3 shows the MATLAB work folder ('Current Folder'). This folder contains the executable EnergyPLAN file (energyPLAN.exe), the example1.m file and the example2.m file, among others. To access them, MATLAB has been informed of the path where the files are located in the computer (using the search bar at the top). The code of the example1.m file is shown in the centre of Fig. 3. This is only an example code which can be modified as required by the user. After executing it (clicking on the Run button), MATLAB generates a list of variables (shown on the right-hand side). These variables appear in the 'Workspace' window and are available for use in any future execution of MATLAB. A summary of the value of each of the variables can also be seen in the 'Workspace' window.

5.2. Example 2: A very simple search for the wind and PV capacities with least CO_2 emissions

Another simple routine can be found in the file named "example2.m", also available with the toolbox. In this case, similar to the example1.m, the code defines:

1. the two paths where the input file and energyPLAN.exe files are placed. It is also specified where the output folder will be created to save the results from the EnergyPLAN execution.

2. two variables ("OnshoreWind" and "PV") as 3x1 vectors, to wrap three different values of the onshore wind capacity and the PV capacity installed in the energy system. This allows a simple search to be made of a new RES configuration based on PV and onshore wind technologies.

3. a string-type variable (named "STR") with the CEEP regulation strategy.

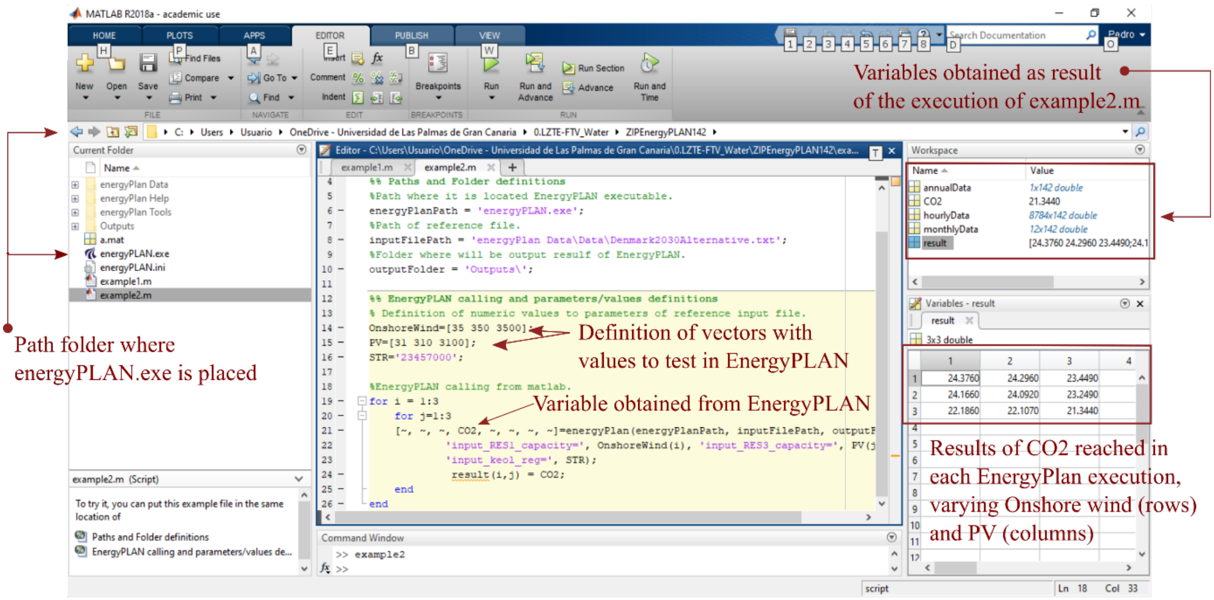

Fig. 4. MATLAB Environment with the example2.m m-file opened and the results of Denmark2030Alternative.txt scenario returned after the sequential executions of EnergyPLAN.

After these definitions, two for-loops launch nine executions (3x3 variations of onshore wind and PV installed capacities, respectively) of the Denmark2030Alternative.txt scenario. The variable chosen to test the performance of the search is 'CO2', which has the CO_2 emissions calculated for each alternative execution.

5.3. Example 3: A simple way to change multiple parameters in multiple EnergyPLAN input files

In energy planning studies, it is common to have to update many scenarios to analyse them with new hourly distributions or changes in some parameters. The file example3.m (available with MaT4EnergyPLAN) shows a code developed to change multiple parameters in multiple input files (Fig. 5). This example changes the names of wind and PV hourly distributions in all EnergyPLAN input files beginning with the word 'Denmark'. There are three original files which are placed in the 'energyPlan Data/Data' path folder ('Denmark100%RES.txt'; 'Denmark2030Alternative.txt'; and 'Denmark2030Reference.txt')

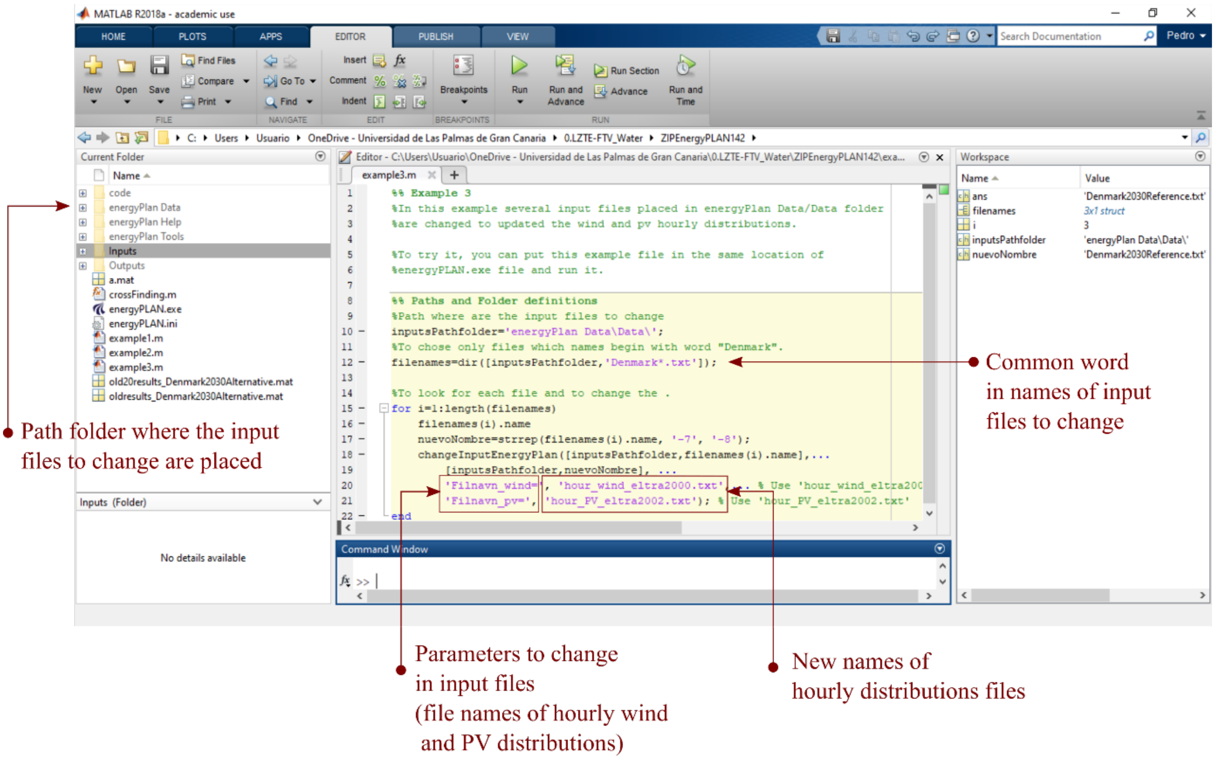

Fig. 5. Example developed in MATLAB to compute the change of multiple parameters in multiple EnergyPLAN input files using the function read.

6. Conclusions

In this paper, a new MATLAB Toolbox has been presented to manage the popular software EnergyPLAN. With the set of functions developed in MATLAB, it is possible to extend energy planning studies to take advantage of all the possibilities that MATLAB offers.

The toolbox has been designed in a simplified way to allow quick use and easy integration with the MATLAB environment. Some usage examples have been provided to facilitate an understanding of the potential applications of the toolbox. Additionally, an explanation is given of how to implement a method previously used in the literature to increase the share of renewable energy sources on islands using the MaT4EnergyPLAN.

7. Required metadata

7.1. Current executable software version

See Table 2.

Table 2. Software metadata

Nr	Code metadata description	

S1	Current software version	*V1.2*
S2	Permanent link to executables of this version	https://www.energyplan.eu/useful_resources/matlab-toolbox-for-energyplan/
S3	Legal Software License	*Apache 2.0*
S4	Computing platform / Operating System	*Microsoft Windows / Linux / OS X*
S5	Installation requirements & dependencies	*MATLAB*
S6	If available Link to user manual - if formally published include a reference to the publication in the reference list	https://www.energyplan.eu/useful_resources/matlab-toolbox-for-energyplan/
S7	Support email for questions	*pedro.cabrerasantana@ulpgc.es*

7.2. Current code version

See Table 3.

Table 3. Code metadata

Nr	**Code metadata description**	
C1	Current code version	*V1.2*
C2	Permanent link to code / repository used of this code version	https://github.com/pcabrerasantana/MATLAB_Toolbox_for_EnergyPLAN
C3	Legal code license	*Apache 2.0*
C4	Code versioning system used	*git*
C5	Software code language used	*MATLAB*
C6	Compilation requirements, operating environments & dependencies	*Not applicable*
C7	If available, link to developer documentation / manual	https://www.energyplan.eu/useful_resources/matlab-toolbox-for-energyplan/
C8	Support email for questions	*pedro.cabrerasantana@ulpgc.es*

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 837089. It has also been co-funded with ERDF funds within the INTERREG MAC 2014-2020 program [DESAL+ Project (MAC/1.1a/094)]. The authors would like to thank Ms. Mette Reiche Sørensen and Ms. Pernille Sylvest Andersen for their helpful and constructive criticism of the manuscript.

References

[1] P.A. Østergaard, Reviewing EnergyPLAN simulations and performance indicator applications in EnergyPLAN simulations, Appl. Energy. 154 (2015) 921–933.

doi:10.1016/j.apenergy.2015.05.086.

[2] M. Child, O. Koskinen, L. Linnanen, C. Breyer, Sustainability guardrails for energy scenarios of the global energy transition, Renew. Sustain. Energy Rev. 91 (2018) 321–334. doi:10.1016/J.RSER.2018.03.079.

[3] A.S. Dagoumas, N.E. Koltsaklis, Review of models for integrating renewable energy in the generation expansion planning, Appl. Energy. 242 (2019) 1573–1587. doi:10.1016/J.APENERGY.2019.03.194.

[4] D. Connolly, H. Lund, B.V. Mathiesen, M. Leahy, A review of computer tools for analysing the integration of renewable energy into various energy systems, Appl. Energy. 87 (2010) 1059–1082. doi:10.1016/j.apenergy.2009.09.026.

[5] H. Lund, J.Z. Thellufsen, P. Sorknæs, D. Connolly, B.V. Mathiesen, P.A. Østergaard, R. Lund, I.R. Skov, K. Hansen, D. Maya-Drysdale, M. Chang, A. David, EnergyPLAN. Advanced energy systems analysis computer model. Documentation V15.0, (2019). https://www.energyplan.eu/training/documentation/ (accessed July 6, 2017).

[6] D. Connolly, H. Lund, B.V. Mathiesen, M. Leahy, The first step towards a 100% renewable energy-system for Ireland, Appl. Energy. 88 (2011) 502–507. doi:10.1016/j.apenergy.2010.03.006.

[7] D. Connolly, B.V. Mathiesen, A technical and economic analysis of one potential pathway to a 100% renewable energy system, Int. J. Sustain. Energy Plan. Manag. 1 (2014) 7–28. doi:10.5278/ijsepm.2014.1.2.

[8] W. Liu, H. Lund, B.V. Mathiesen, Large-scale integration of wind power into the existing Chinese energy system, Energy. 36 (2011) 4753–4760. doi:10.1016/j.energy.2011.05.007.

[9] I. van Beuzekom, M. Gibescu, J.G. Slootweg, A review of multi-energy system planning and optimization tools for sustainable urban development, in: 2015 IEEE Eindhoven PowerTech, IEEE, 2015: pp. 1–7. doi:10.1109/PTC.2015.7232360.

[10] I. Batas Bjelić, N. Rajaković, Simulation-based optimization of sustainable national energy systems, Energy. 91 (2015) 1087–1098. doi:10.1016/J.ENERGY.2015.09.006.

[11] M.S. Mahbub, M. Cozzini, P.A. Østergaard, F. Alberti, Combining multi-objective evolutionary algorithms and descriptive analytical modelling in energy scenario design,

Appl. Energy. 164 (2016) 140–151. doi:10.1016/J.APENERGY.2015.11.042.

[12] M.G. Prina, M. Cozzini, G. Garegnani, G. Manzolini, D. Moser, U. Filippi Oberegger, R. Pernetti, R. Vaccaro, W. Sparber, Multi-objective optimization algorithm coupled to EnergyPLAN software: The EPLANopt model, Energy. 149 (2018) 213–221. doi:10.1016/J.ENERGY.2018.02.050.

[13] H. Lund, J.Z. Thellufsen, P. Sorknæs, D. Connolly, B.V. Mathiesen, P.A. Østergaard, R. Lund, I.R. Skov, K. Hansen, D. Maya-Drysdale, FAQs | EnergyPLAN, (2017). https://www.energyplan.eu/training/faqs/#eleven (accessed May 2, 2019).

[14] P. Cabrera, H. Lund, J.A. Carta, Smart renewable energy penetration strategies on islands: The case of Gran Canaria, Energy. 162 (2018) 421–443. doi:10.1016/j.energy.2018.08.020.