cvxmg

Release 2.0.0

Aug 30, 2020

Contents

CVXMG is a python based package that allows the users to compute the sizing of Isolated/Islanded MicroGrids (IMGs). Additionally to the sizing of the IMGs, CVXMG returns the optimal dispatch of the energy sources and the optimal tariffs for the energy. CVXMG allows implementing Demand Side Management (DSM) in the sizing of the IMGs. The user of CVXMG can choose seven different DSM strategies based on dynamic pricing of the energy and one DSM based on direct control of the loads:

- Time of Use of two price levels
- Time of Use with an incentive for solar generation
- Time of use of three levels of price
- Critical Peak Pricing
- Day-Ahead Dynamic Pricing
- Fixed Shape Pricing
- Incentive Based Pricing
- Directly Curtailing the Electrical Demand

CVXMG allows for creating different business models for IMG projects. CVXMG allows defining the percentage of public or private funding for the project. CVXMG defines the energy's tariffs for the customers using the business model information and the share of public and private financing. These capabilities make CVXMG a worth looking tool for different analyses for IMG planners and policymakers.

CVXMG uses CVXPY at its core. CVXPY is a Python-embedded modeling language for convex optimization problems. CVXPY allows CVXMG to build and solve deterministic and stochastic convex formulations to perform the analysis of the IMGs. Due to the speed of solution of convex formulations, CVXMG can perform a multiyear analysis in seconds! Moreover, CVXMG can execute multiyear stochastic analysis in a regular machine.

CHAPTER 1

Installing CVXMG

CVXMG can be easily installed using PyPi or downloading the GitHub repository. To install CVXMG just execute the following in your line of commands:

```
pip install cvxmg
```

Tip: The installation of CVXPY can raise errors in Windows if the user do not have Visual Studio build tools for Python 3. To solve this issue please refer to the installation official page of CVXPY, carefully read and follow the instructions explained there.

CHAPTER 2

Using CVXMG

CVXMG uses two simple dictionaries "prob_info" and "sources_info" to create different architectures of IMGs. These two dictionaries are attributes for the constructor classes. CVXMG offers three different constructor classes: One for deterministic analysis, one for multiyear analysis, and one for multiyear stochastic analysis. Each of the constructors uses the information of "prob_info" to know the architecture of the IMG and the information of "sources_info" to know the characteristics of the energy sources. CVXMG creates the energy sources of the IMG as objects using the Objected Oriented Programming capabilities offered by Phyton. The use of objects for the energy sources allows CVXMG to build the optimization formulation of the problem using a Plug and Play approach.

Once the user defines "prob_info" and "sources_info" needs to execute a constructor. For more info on how to set "prob_info" and "sources_info," please refer to the example section. Suppose the user wants to compute the sizing of an IMG using a deterministic analysis of one year. In that case, the user must execute the following command:

```
from cvxmg import cvxmg as cm
MicroGrid = cm.DeterministicDSMS(prob_info, sources_info)
```

The above line of commands will create the structure of the IMG in the object MicroGrid. Additionally, it will guarantee that all the optimization formulation follows the Disciplined Convex Programming rules already established in CVXPY. However, at this moment, CVXMG did not solve the formulation yet. To solve the formulation, the user needs to execute the solve method:

```
MicroGrid.solveMG()
```

The above commands will solve the formulation and will store the results in the MicroGrid object. To extract the results, the user must execute:

```
summary, dispatch_results = MicroGrid.resultsMG()
```

The above line of commands will create a pandas structure in the variable summary with the sizing results' essential variables. Additionally, the method will create another pandas structure to store the dispatch results of energy sources.

Finally, if the user wants to create some predetermined plots of the results can call the method plotMG:

```
MicroGrid.plotMG()
```

The following are the contents of this guide:

2.1 Setting the informationf of the IMG

A crucial part of the process using CVXMG is defining the information of the project. To do so, CVXMG needs to dictionaries, “prob_info” and “sources_info”. In the following lines present a brief description of how to create the dictionaries “prob_info” and “sources_info”.

2.1.1 Setting “prob_info”

CVXMG provides two .csv files to make easier setting the data of “prob_info”.

The file named config.csv contains the information about the architecture of the IMG. The file named resourcedata.csv cointains the information of the primary energy resources. CVXMG provides one function to read the data of config.csv and three different functions to read the reorcedata.csv file. The function to read config.csv is “variables”. The three functions to import the resource data are: resources_norm, resources_all and resources_noise. These functions are crucial for the multyyear analysis and the stochastic analysis. For more information about these functions, please refer to the docs of the functions.

The user must do the following to import the information of prob_info dictionary:

```
import  pandas          as pd
from cvxmg import cvxmg as cm              # Import cvxmg

#region to read the parameters to intialize the code
prob_info = {}
variables_csv   = pd.read_csv('config.csv', sep=';', header=None, skip_blank_
↪lines=True)
prob_info['project_life_time'], prob_info['interest_rate'], prob_info['scenarios'],␣
↪prob_info["years"], prob_info["scala"], prob_info["prxo"], prob_info["percentage_
↪yearly_growth"], prob_info["percentage_variation"], prob_info["dlcpercenthour"],␣
↪prob_info["dlcpercenttotal"], prob_info["sen_ince"], prob_info["sen_ghi"], prob_
↪info["elasticity"], prob_info["curtailment"], prob_info["capex_private"], prob_info[
↪"capex_gov"], prob_info["capex_community"], prob_info["capex_ong"], prob_info["opex_
↪private"], prob_info["opex_gov"], prob_info["opex_community"], prob_info["opex_ong
↪"], prob_info["rate_return_private"], prob_info["max_value_tariff"], prob_info[
↪'drpercentage'], prob_info['diesel_system'], prob_info['pv_system'], prob_info[
↪'battery_system'], prob_info['wind_system'], prob_info['hydro_system'], prob_info[
↪'hydrogen_system'], prob_info['gas_system'], prob_info['biomass_system'], prob_info[
↪'flat'], prob_info['tou'], prob_info['tou_sun'], prob_info['tou_three'], prob_info[
↪'cpp'], prob_info['dadp'], prob_info['shape_tar'], prob_info['ince'], prob_info[
↪'dilc'], prob_info['residential'],prob_info['commercial'],prob_info['industrial'],
↪prob_info['community']   = cm.variables(variables_csv)
#endregion

#region to read the weather data, the load and resource availability for the community
data_csv        = pd.read_csv('resourcedata.csv', sep=';', header=None, skip_blank_
↪lines=True)
prob_info["ghi"], prob_info["irrdiffuse"], prob_info["temperature"], prob_info["wind
↪"], prob_info["hydro"], prob_info["load_residential"], prob_info["load_commercial"],
↪ prob_info["load_industrial"], prob_info["load_community"] = cm.resources_norm(data_
↪csv, years=prob_info["years"], scenarios=prob_info["scenarios"], percentage_yearly_
↪growth=prob_info["percentage_yearly_growth"])
#endregion
```

2.1.2 Setting "sources_info"

The information of "sources_info" specify the characteristics of the energy sources. Each energy source expect different parameters. "sources_info" is a nested dictionary. To create "sources_info", the user first must initialize the dictionary:

```
sources_info = {}
```

If the user wants to create the information of a Battery Energy Storage System must execute:

```
sources_info = {
# Battery Energy Storage System info
"bess_1" : {
    "life_time"              : 2,
    "investment_cost"        : 420,                                              # USD
    "fuel_function"          : 0,                                                # ␣
↪Fuel function
    "fuel_cost"              : 0,                                                # USD
    "maintenance_cost"       : 6,                                                # ␣
↪Percentage of the capacity
    "min_out_power"          : 50,                                               # ␣
↪Percentage of the capacity
    "max_out_power"          : 100,                                              # ␣
↪Percentage of the capacity
    "rate_up"                : 1,                                                # ␣
↪Percentage of the capacity
    "rate_down"              : 1,                                                # ␣
↪Percentage of the capacity
    "initial_charge"         : 50,                                               # ␣
↪Percentage of the capacity
}
}
```

If the user wants to create the information of a Diesel Generator must execute:

```
sources_info = {
# Diesel generator info
"diesel_gen_1" : {
    "life_time"          : 3,
    "investment_cost"    : 550,                                                  # USD
    "fuel_function"      : np.array([0.246, 0.08415]),                           # ␣
↪Fuel function
    # "fuel_function"      : np.array([0.000203636364, 0.224872727, 4.22727273]),  # ␣
↪Fuel function
    "fuel_cost"          : 0.8,                                                  # USD
    "maintenance_cost"   : 6,                                                    # ␣
↪Percentage of the capacity
    "min_out_power"      : 0,                                                    # ␣
↪Percentage of the capacity
    "max_out_power"      : 100,                                                  # ␣
↪Percentage of the capacity
    "rate_up"            : 1,                                                    # ␣
↪Percentage of the capacity
    "rate_down"          : 1,                                                    # ␣
↪Percentage of the capacity
}
}
```

If the user wants to create the information of a Photovoltaic System must execute:

```
sources_info = {
# Photovoltaic system info
"pv_gen_1" : {
    "life_time"           : 25,
    "investment_cost"     : 1300,                                                    # USD
    "maintenance_cost"    : 6,                                                       #␣
↪Percentage of the capacity
    "rate_up"             : 1,                                                       #␣
↪Percentage of the capacity
    "rate_down"           : 1,
    "derat"               : 1,          # Derating factor
    "pstc"                : 0.3,        # Nominal capacity of the ov module              ␣
↪                                           # Percentage of the capacity
    "Ct"                  : -0.0039,    # Termic coefficient of the pv module
}
}
```

If the user wants to create the information of the wind generation System must execute:

```
sources_info = {
# Wind generator info
"wind_gen_1" : {
    "life_time"           : 15,
    "investment_cost"     : 2000,                                                    # USD
    "maintenance_cost"    : 5,                                                       #␣
↪Percentage of the capacity
    "rate_up"             : 1,                                                       #␣
↪Percentage of the capacity
    "rate_down"           : 1,
    "rated_speed"         : 13,
    "speed_cut_in"        : 3,
    "speed_cut_out"       : 12.5,
    "nominal_capacity"    : 1,

}
}
```

It is crucial to specify the information of the lack of energy and the excess of energy in “sources_info”. The user can use this information to control the desired level of reliability of the microgrid and to associate a cost to these values. To create this information the user must execute:

```
sources_info = {
# Lack of energy info
"lack_ene" : {
    "cost_function"     : 0,                                                       #␣
↪Cost function
    "reliability"       : 2,                                                       #␣
↪Percentage of reliability                                                       #␣
↪Percentage of the capacity
},

# Excess of energy info
"excess_ene" : {
    "cost_function"     : 0,                                                       #␣
↪Cost function
    "reliability"       : 2,                                                       #␣
↪Percentage of reliability
```

(continues on next page)

(continued from previous page)

```
}
}
```

2.2 Types of constructors for the IMG

CVXMG have three different types of constructors that perform different analysis for IMGs. All the constructors hace the following constraints: * One constraint to perform the energy balance of the IMG. * One constraint to guarantee that the total delivered energy remains cosntant after the aplication of the DSM. * One constraint to guarantee that the lack of energy do not exceed the desired reliability. * One constraint to guarantee that the excess of energy do not exceed the desired reliability. * One constraint to guarantee that the private investors recover their investments and the desired return of investment.

The only difference between them is the horizon of optimization (one year, multiyear) and the type of analysis (deterministic, stochastic).

A brief description of each of the constructors proceeds.

2.2.1 DeterministicDSMS

The deterministic constructor performs the sizing of the microgrid considering one year of operation. Additionlly, as the name suggests, the deterministic constructor use a deterministic analysis.

To use this constructor the user must execute:

```
import cvxmg as cm
MicroGrid = cm.DeterministicDSMS(prob_info, sources_info)
```

2.2.2 MultiyearDSMS

The multiyear constructor performs the sizing of the microgrid considering one or several years of operation. The number of years are specified by the user in "prob_info" dictionary. The multiyear constructor use a deterministic analysis. The multy year constructor implements the adaptative method described in [pece2019]. However, the multy year analysis here does not consider three day each month as the authors propose in the article. The multy-year analysis here considers the full year analysis (8760 hours).

To use this constructor the user must execute:

```
import cvxmg as cm
MicroGrid = cm.MultiyearDSMS(prob_info, sources_info)
```

2.2.3 StochasticDSMS

The stochastic constructor performs the sizing of the microgrid considering one or several years of operation. However, the stochastic constructor performs a stochastic analysis. By using the functions resources_norm, resources_all and resources_noise CVXMG creates the data to perform the multiyear stochastic analysis. This constructor requires that the user specify the number of years for the multiyear analysis and the number of escenarios for the stochastic analysis.

To use this constructor the user must execute:

```
import cvxmg as cm
MicroGrid = cm.StochasticDSMS(prob_info, sources_info)
```

2.3 Example of how to use CVXMG

Imagine that a user wants to compute the sizing of an IMG with photovoltaic panels, a diesel generator , a wind generator and a battery energy storage system, using a stochastic analysis. To do so, the user must first create the information for “prob_info” and “sources_info”. Afterwards, the user must set the constructor and solve the problem. The following lines present a brief example using the multiyear constructor.

2.3.1 First step: Create the information of “prob_info” and “sources_info”

To import the information of prob_info dictionary, must do the following:

```
import numpy                as      np              # Library to work with arrays␣
↪and math
import pandas               as      pd              # Library for date frames␣
↪handling
import cvxpy                as      cp              # Library for convex␣
↪optimization
import cvxmg                as      cm              # Library for the planning of␣
↪Islanded Microgrids
import matplotlib.pyplot    as      plt             # Ploting command
plt.style.use('default')                            # Restore default values for␣
↪graphs

#region to read the parameters to intialize the code
prob_info = {}
variables_csv   = pd.read_csv('config.csv', sep=';', header=None, skip_blank_
↪lines=True)
prob_info['project_life_time'], prob_info['interest_rate'], prob_info['scenarios'],␣
↪prob_info["years"], prob_info["scala"], prob_info["prxo"], prob_info["percentage_
↪yearly_growth"], prob_info["percentage_variation"], prob_info["dlcpercenthour"],␣
↪prob_info["dlcpercenttotal"], prob_info["sen_ince"], prob_info["sen_ghi"], prob_
↪info["elasticity"], prob_info["curtailment"], prob_info["capex_private"], prob_info[
↪"capex_gov"], prob_info["capex_community"], prob_info["capex_ong"], prob_info["opex_
↪private"], prob_info["opex_gov"], prob_info["opex_community"], prob_info["opex_ong
↪"], prob_info["rate_return_private"], prob_info["max_value_tariff"], prob_info[
↪'drpercentage'], prob_info['diesel_system'], prob_info['pv_system'], prob_info[
↪'battery_system'], prob_info['wind_system'], prob_info['hydro_system'], prob_info[
↪'hydrogen_system'], prob_info['gas_system'], prob_info['biomass_system'], prob_info[
↪'flat'], prob_info['tou'], prob_info['tou_sun'], prob_info['tou_three'], prob_info[
↪'cpp'], prob_info['dadp'], prob_info['shape_tar'], prob_info['ince'], prob_info[
↪'dilc'], prob_info['residential'],prob_info['commercial'],prob_info['industrial'],
↪prob_info['community']   = cm.variables(variables_csv)
#endregion

#region to read the weather data, the load and resource availability for the community
data_csv        = pd.read_csv('resourcedata.csv', sep=';', header=None, skip_blank_
↪lines=True)
prob_info["ghi"], prob_info["irrdiffuse"], prob_info["temperature"], prob_info["wind
↪"], prob_info["hydro"], prob_info["load_residential"], prob_info["load_commercial"],
↪ prob_info["load_industrial"], prob_info["load_community"] = cm.resources_norm(data_
↪csv, years=prob_info["years"], scenarios=prob_info["scenarios"], percentage_yearly
↪growth=prob_info["percentage_yearly_growth"])
```

(continues on next page)

(continued from previous page)

```
#endregion
```

To define the characteristics of the energy sources that the IMG will use the user must do the following:

```
#region to set the characteristics of the energy sources

sources_info = {
   # Battery Energy Storage System info
   "bess_1" : {
      "life_time"            : 2,
      "investment_cost"      : 420,                                              #␣
↪USD
      "fuel_function"        : 0,                                                #␣
↪Fuel function
      "fuel_cost"            : 0,                                                #␣
↪USD
      "maintenance_cost"     : 6,                                                #␣
↪Percentage of the capacity
      "min_out_power"        : 50,                                               #␣
↪Percentage of the capacity
      "max_out_power"        : 100,                                              #␣
↪Percentage of the capacity
      "rate_up"              : 1,                                                #␣
↪Percentage of the capacity
      "rate_down"            : 1,                                                #␣
↪Percentage of the capacity
      "initial_charge"       : 50,                                               #␣
↪Percentage of the capacity
   },

   # Diesel generator info
   "diesel_gen_1" : {
      "life_time"        : 3,
      "investment_cost"  : 550,                                                  #␣
↪USD
      "fuel_function"    : np.array([0.246, 0.08415]),                           #␣
↪Fuel function
      # "fuel_function"    : np.array([0.000203636364, 0.224872727, 4.22727273]),  #␣
↪Fuel function
      "fuel_cost"        : 0.8,                                                  #␣
↪USD
      "maintenance_cost" : 6,                                                    #␣
↪Percentage of the capacity
      "min_out_power"    : 0,                                                    #␣
↪Percentage of the capacity
      "max_out_power"    : 100,                                                  #␣
↪Percentage of the capacity
      "rate_up"          : 1,                                                    #␣
↪Percentage of the capacity
      "rate_down"        : 1,                                                    #␣
↪Percentage of the capacity
   },

   # Photovoltaic system info
   "pv_gen_1" : {
      "life_time"        : 25,
      "investment_cost"  : 1300,                                                  #␣
↪USD
```

(continues on next page)

(continued from previous page)

```
        "maintenance_cost"  : 6,                                                    #␣
↪Percentage of the capacity
        "rate_up"           : 1,                                                    #␣
↪Percentage of the capacity
        "rate_down"         : 1,
        "derat"             : 1,        # Derating factor
        "pstc"              : 0.3,      # Nominal capacity of the ov module          ␣
↪                                           # Percentage of the capacity
        "Ct"                : -0.0039,  # Termic coefficient of the pv module
    },

    # Wind generator info
    "wind_gen_1" : {
        "life_time"         : 15,
        "investment_cost"   : 2000,                                                 #␣
↪USD
        "maintenance_cost"  : 5,                                                    #␣
↪Percentage of the capacity
        "rate_up"           : 1,                                                    #␣
↪Percentage of the capacity
        "rate_down"         : 1,
        "rated_speed"       : 13,
        "speed_cut_in"      : 3,
        "speed_cut_out"     : 12.5,
        "nominal_capacity"  : 1,

    },

    # Lack of energy info
    "lack_ene" : {
        "cost_function"    : 0,                                                    #␣
↪Cost function
        "reliability"      : 2,                                                    #␣
↪Percentage of reliability                                                    #␣
↪Percentage of the capacity
    },

    # Excess of energy info
    "excess_ene" : {
        "cost_function"    : 0,                                                    #␣
↪Cost function
        "reliability"      : 2,                                                    #␣
↪Percentage of reliability
    }
    }

#endregion
```

2.3.2 Second step: Set the constructor

To use the constructor, the user must execute the following:

```
MicroGrid = cm.StochasticDSMS(prob_info_input=prob_info, sources_info=sources_info)
```

To extract the results of the optimization the user must execute:

```
summary=MicroGrid.resultsMG()
```

All the results are stored inside of the summary variable.

2.4 Methodology formulation

The problem that aims to solve CVXMG is to study the effects of Demand Side Management (DSM) strategies over the planning of Islanded/Isolated Microgrids (IMGs). The study should consider the impact of the technical and environmental aspects. The study will address regulatory aspects as well. This requires a methodology capable of:

- Integrate different energy sources for the IMG.
- Compute the sizing of the energy sources.
- Compute the energy dispatch of the energy sources.
- Consider the effects of the DSM over the lifetime of the project.
- Consider business models to recreate the real-life conditions of the development of IMG projects.
- Set the tariffs of the energy for the customers.
- Evaluate the impact of the DSM strategies over the planning of IMGs.

A methodology with the above characteristics does not exist in the reviewed literature. In this regard, CVXMG builds a methodology to solve that.

2.4.1 Proposed solution

The proposed solution implements a multiyear-stochastic analysis using Disciplined Convex Stochastic Programming (DCSP). DCSP builds on principles from stochastic optimization and convex analysis, representing a considerable advantage to build the desired methodology [Ali2015]. Equation (2.1) presents the general formulation of a convex stochastic problem:

$$\begin{aligned} &minimize\ E(a_1(x,\xi)) \\ subject\ to\ &E(b_i(x,\xi)) = 0 \ \ i=1,\dots,B \\ &c_i(x,\xi) \geq 0 \ \ i=1,\dots,C \end{aligned} \tag{2.1}$$

where $b_i : \mathbf{R}^n \times \mathbf{R}^q \rightarrow \mathbf{R}$, $i = 1,\dots,B$ are convex functions in x for each value of the random variable $\xi \in \mathbf{R}^q$, and $c_i : \mathbf{R}^n \rightarrow \mathbf{R}$, $i = 1,\dots,C$ are (deterministic) affine functions; since expectations preserve convexity, the objective and inequality constraint functions in (2.1) are (also) convex in x, making (2.1) a convex optimization problem [Ali2015], [Liberti2008].

2.4.2 Main assumptions

The formulation of the methodology assumes that the planner can have at least one year of historical data of weather variables and electrical demand. The formulation use this historical data to build the multiyear, and multiyear-stochastic analysis of the methodology by using a scenario construction technique.

The methodology assumes that there is no presence of smart or controllable loads in the IMGs. Considering this, it is not possible to apply advanced DSM strategies for IMGs. Due to this limitation, the present study proposes to use price-based DSM strategies and one DSM strategy based on Direct Load Curtailment. Both kinds of DSM strategies offer less technical difficulty as their more sophisticated counterparts.

The formulation also assumes that the planner can know the price elasticity of the demand of the customers. By using the price elasticity of the customers' demand, it is possible to compute how they will react to different stimuli. Additionally, the price elasticity of the demand intrinsically implies that without any external stimulus, the customers do not have any incentive to modify their consumption patterns. This assumption means that customers will not alter their consumption patterns if the IMG uses a flat tariff.

2.4.3 Mathematical formulation

The formulation of the problem aims to minimize the total costs of the IMG project. The total costs of the project are Capital Expenditures (ζ), Operational Expenditures (ϑ), Maintenance Expenditures (μ) and Carbon Taxes Expenditures (ϕ):

$$\zeta = \sum_{u=1}^{U} C_u I_u (2.2)(2.2)$$

$$\vartheta = \sum_{t=1}^{T} \sum_{u=1}^{U} \lambda_{u,t} E_{u,t} (2.3)(2.3)$$

$$\mu = \sum_{t=1}^{T} \sum_{u=1}^{U} \Lambda_{u,t} E_{u,t} (2.4)(2.4)$$

$$\Phi = \sum_{t=1}^{T} \sum_{u=1}^{U} B_u F_{u,t} (2.5)(2.5)$$

and C_u, I_u, $\lambda_{u,t}$, $\Lambda_{u,t}$, $E_{u,t}$, B_u and $F_{u,t}$ represent the installed capacity, unitary investment cost, unitary dispatch costs, unitary maintenance costs, dispatched energy, carbon dioxide production by liter, and fuel consumption of the u energy source at time t, respectively. T represents the horizon of the optimization.

The mathematical formulation allows the planner to build all kinds of business models by considering that a $i \in I$ number of different investors (φ) can fund the IMG project. These $i \in I$ investors can contribute to pay capital ($\varphi_{i,\zeta}$), operational ($\varphi_{i,\vartheta}$) or maintenance ($\varphi_{i,\mu}$) expenditures. The objective function captures the different sources of money to fund the project:

$$X_1 = argmin_{C_u, E_{u,t}} \sum_{i=1}^{I} \varphi_{i,\zeta}\zeta + \varphi_{i,\vartheta}\vartheta + \varphi_{i,\mu}\mu + \varphi_{i,\phi}\phi (2.6)(2.6)$$

The formulation considers the energy prices as the only revenue stream for the investors that aim to recover their investment and have profits. If the business model has private investors ($varphi^{priv}$) the formulation allows to guarantee an expected Rate of Return (R) using the following constraint:

$$(1+R)\sum_{y=1}^{Y}(\varphi^{priv,\zeta}\zeta_y+\varphi^{priv,\vartheta}\vartheta_y+\varphi^{priv,\mu}\mu_y+\varphi^{priv,\phi}\phi_y)\geq\sum_{t=1}^{YT}\pi_{x,t}D_t^{dr}(2.7)(2.7)$$

where $pi_{n,t}$ is the price of the energy at time t using the n DSM strategy, and D_{t}^{dr} is the electrical demand after the x DSM strategy is applied. However, it is crucial to highlight that the horizon of this constraint is the life time of the project. The life time of the project is measured in years (Y) for the sum in the left, and in hours for the sum in the right (Y multiplied by T).

Equation (2.8) uses the demand with flat tariff (D_{t}^{flat}) as the base demand, the flat tariff (pi^{flat}) as the base price, the x price ($pi_{x,t}$) as the DSM tariff, and the price-elasticity (e_{t}) of the customers to compute the response of the demand D_{t}^{dr}.

$$e_t=\frac{\pi^{flat}(D_t^{dr}-D_t^{flat})}{D_t^{flat}(\pi_{x,t}-\pi^{flat})}(2.8)(2.8)$$

The formulation allows defining the changes in the total electrical demand after the introduction of the DSM using factor Psi^{c} in Equation (2.9). Factor Psi^{c} is an input parameter that the planner choose according to the conditions of the IMG project. Values $Psi^{c} leq 1$ decreases the total energy consumption, while values $Psi^{c} geq 1$ increases the total energy consumption over the optimization horizon. A value $Psi^{c}=1$ indicates that the total energy consumption over the optimization horizon remains constant after the introduction of DSM.

$$\sum_{t=1}^{T}D_t^{dr}-\Psi^c\sum_{t=1}^{T}D_t^{flat}=0(2.9)(2.9)$$

The formulation naturally includes the balance Equation:

$$\sum_{t=1}^{T}\sum_{u=1}^{U}E_{u,t}-EE_t+LE_t-D_t^{dr}=0(2.10)(2.10)$$

where \$EE_t\$ and \$LE_t\$ are the excess and lack of energy. According to [Chauhan2014], [Diaf2008], the loss of power supply probability (LPSP) is:

$$LPSP = \frac{\sum_{t=1}^{T} LE_t}{\sum_{t=1}^{T} D_t^{dr}} (2.11)(2.11)$$

Similarly, Equation (2.12) defines the excess of power supply probability (EPSP) as:

$$EPSP = \frac{\sum_{t=1}^{T} EE_t}{\sum_{t=1}^{T} D_t^{dr}} (2.12)(2.12)$$

By using Equations (2.11) and (2.12) it is possible to create two constraints to control LPSP (2.13) and EPSP (2.14) over the optimization horizon:

$$\sum_{t=1}^{T} LE_t \leq LPSP \sum_{t=1}^{T} D_t^{dr} (2.13)(2.13)$$

$$\sum_{t=1}^{T} EE_t \leq EPSP \sum_{t=1}^{T} D_t^{dr} (2.14)(2.14)$$

2.5 DSM integration into the sizing

The methodology integrates ToU, CPP, DADP, IBP, Fixed Shape Pricing (FSP) and DLCt as DSM strategies into the sizing of the IMG. The baseline case for comparisons does not use a DSM strategy, it only uses a flat tariff. The description of the baseline case and each of the DSM strategies proceeds in the following subsections [Celik2017].

2.5.1 Flat tariff (Baseline case):

In general terms, the value of a flat tariff is the sum of all the costs of producing the energy divided by the total amount of energy produced [Inversin2000]. Equation (2.15) describes the yearly payments using a regular flat tariff.

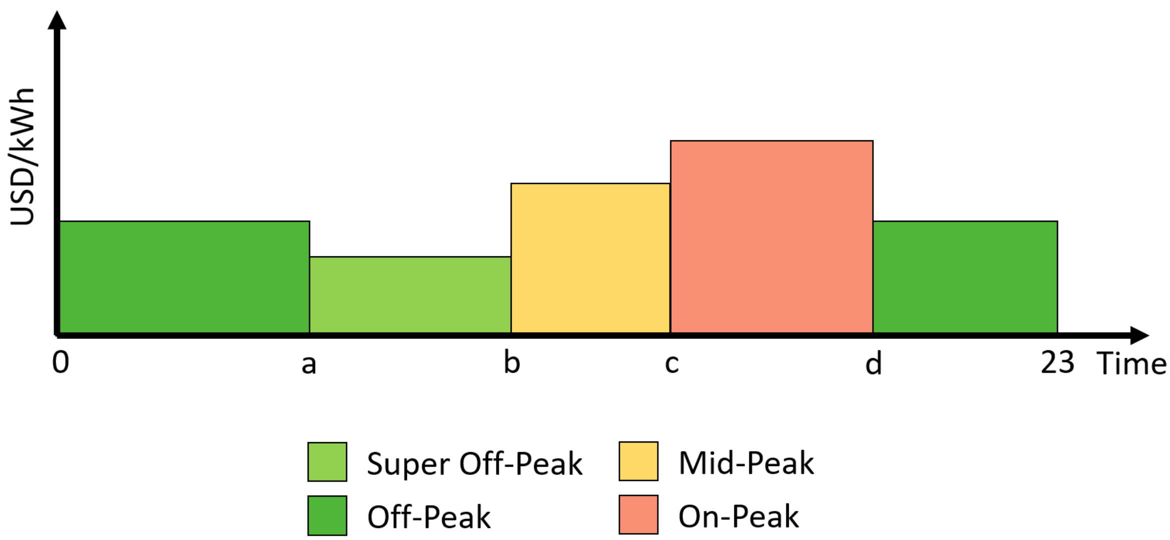
$$\Gamma_n^{flat} = \frac{\zeta_y + \vartheta_y + \mu_y + \phi_y}{\sum_{t=1}^{T} D_t^{dr}} (1+R) \sum_{t=1}^{T} D_{n,t}^{dr} (2.15)(2.15)$$

However, this traditional approach does not set an optimal tariff to recover investments while minimizing energy costs. Here we propose to introduce a decision variable π^{flat} into the formulation to find the optimum price for the tariff.

$$\Gamma_n^{flat} = \pi^{flat} \sum_{t=1}^{T} D_{n,t}^{dr} (2.16)(2.16)$$

2.5.2 Time of use tariff:

ToU tariffs vary daily or seasonally on a fixed schedule, using two or more constant prices [Baatz2017]. One of the main benefits of this type of fare is its stability over long periods, which gives the customer a better ability to adapt to it [Glick2014, Kostkova2013]_. To create a ToU tariff, the planner must define the number of Z blocks, and the starting and ending hours of each z block [Glick2014]. The optimization problem considers the prices π_z of the Z number of blocks as decision variables. The following figure shows the main components of a ToU tariff, and equation (2.17) presents the yearly payments using Z different block hours of prices.



$$\Gamma_n^{ToU} = \sum_{t=1}^{T} \sum_{z=1}^{Z} \pi_z D_{z,n,t}^{dr} (2.17)(2.17)$$

2.5.3 Critical peak pricing:

CPP tariff can be 3 to 5 times higher than the usual tariff but is allowed only a few days per year [Kostkova2013]. In Equation (2.18), π_{base} is a scalar variable, that is chosen to be equal to the flat tariff π^{flat}. π_{peak} is a decision variable of dimension T. Equation (2.18) defines the day-ahead forecasted payments using a CPP tariff, and Equation (2.19) defines the day-ahead hourly critical peak price.

$$\Gamma_n^{CPP} = \sum_{t=1}^{T}(\pi^{base} + \pi_t^{peak})D_{n,t}^{dr}(2.18)(2.18)$$

$$\pi_t^{CPP} = \pi^{base} + \pi_t^{peak}(2.19)(2.19)$$

A critical forecasted event as high demand or low generation capacity triggers the critical peak price in a CPP tariff. In this regard, the CPP tariff must include a predictor of the critical event and a decision mechanism to set the value of the critical price. The formulation uses historical data, which implies that the formulation has full knowledge over the optimization horizon (T:=:8760:hours). The perfect knowledge allows the formulation to state constraint (2.20), limiting the apparition of the critical price only to a few hours in a year. Equation (2.20) uses variable φ_{peak}, to control the number of hours with critical price allowed and δ_{peak} to define how many times the base price π_{base} is scaled up. The planner defines φ_{peak} and δ_{peak}. π_{base}, π_{peak}, τ_{base} and τ_{peak} are decision variables that the optimization formulation needs to compute.

$$\sum_{t=1}^{T}\pi_{peak,t} \leq \varphi_{peak}T\delta_{peak}\pi_{base}(2.20)(2.20)$$

2.5.4 Day ahead dynamic pricing:

DADP refers to a tariff that is announced one day in advance to customers and has hourly variations. This scheme offers less uncertainty to customers than *hour-ahead pricing* or *real-time pricing,* thus allowing them to plan their activities [Wong2012], [Borenstein2002]. Equation (2.21) introduces the payments under DADP tariff, using π_t as a decision variable vector of dimension T.

$$\Gamma_n^{DADP} = \sum_{t=1}^{T}\pi_t D_{n,t}^{dr}(2.21)(2.21)$$

2.5.5 Incentive-based pricing:

The IBP tariff provides discounts on the tariff to the customers to increase the electric energy consumption or an extra fare to penalize it. The planner can decide the IBP base price to be equal to the flat tariff π^{flat} to guarantee a constant value each day. Variable $\pi_{inc,t}$ computes the hourly incentives and can take positive or negative values. Equation (2.22) defines the payments using the IBP tariff.

$$\Gamma_n^{IBP} = \sum_{t=1}^{T} D_{n,t}^{dr}(\pi^{base} + \pi_t^{inc})(2.22)(2.22)$$

$$\pi_t^{IBP} = \pi^{base} + \pi_t^{inc}(2.23)(2.23)$$

2.5.6 Fixed Shape Pricing:

Dole et al. affirm that tariffs must be simple, transparent, and predictable for the customers [Dole2004]. By following these recommendations, it is possible to design a pricing scheme that combines the benefits of DADP with the predictability of the ToU tariff. This pricing scheme receives the name of Fixed Shape Pricing (FSP). FSP tariffs can provide more stimulus than the ToU tariff. However, the FSP tariff has the same predictability of the ToU tariff. Although the FSP tariff will not be as simple as the ToU, it will be simpler for the customers than DADP tariffs.

The FSP tariff fixes one price for each hour over all the days of the year. FSP tariff does not reflect the real costs of producing electricity in the IMG, which is a drawback. However, in the long run, the FSP tariff might offer better results than the ToU pricing. Additionally, it might be easier to accept by the IMG customers than the DADP tariff.

To build the FSP tariff the methodology assigns one variable for each hour of the day. All these variables are one-dimensional. By using these variables the methodology builds a vector of 24 positions, and repeat it till reaching the optimization horizon. The resulting vector is the price of the tariff. Equation (2.24) shows the payments of the n customer when the planner choose to use the FSP tariff as DSM strategy.

$$\Gamma_n^{FSP} = \sum_{t=1}^{T} \sum_{h=1}^{24} \pi_h^{FSP} D_{n,t}^{dr}(2.24)(2.24)$$

All the tariffs must have restrictions to avoid null or excessive pricing. Governments, policymakers, or IMG owners can guarantee fair fares to the customers with the following constraint:

$$\pi^{min} \leq \pi_x \leq \pi^{max}(2.25)(2.25)$$

2.5.7 Direct Load Curtailment Strategy:

The DLCt strategy curtails a portion ϵ_t out of the demand if required. The planner of the IMG decides the percentage of max curtailed hourly demand θ, and the percentage of the total energy curtailed in the optimization period κ. The final demand and payments are defined as follows:

$$D_t^{dr} = D_t^{flat} - \epsilon_t (2.26)(2.26)$$

$$\Gamma_n^{DLCt} = \sum_{t=1}^{T} D_{n,t}^{dr} \pi^{flat} (2.27)(2.27)$$

The general restrictions for the DLCt strategy are defined as follows:

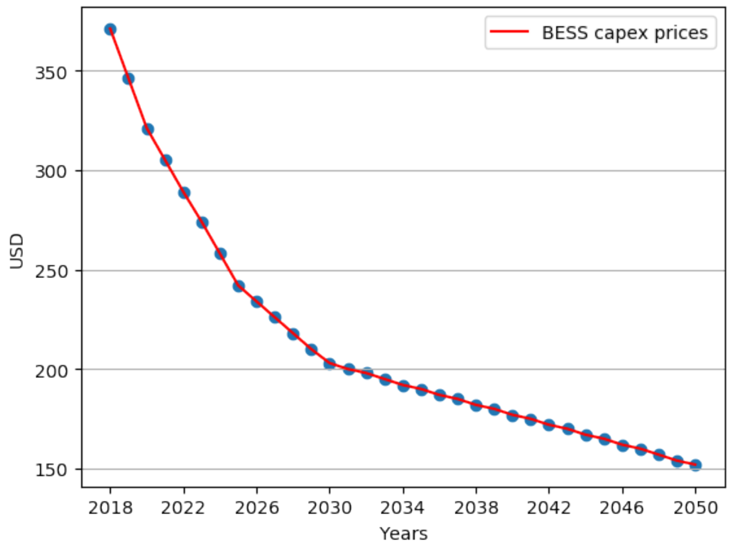
$$\epsilon_t \leq \theta D_t^{dr} (2.28)(2.28)$$

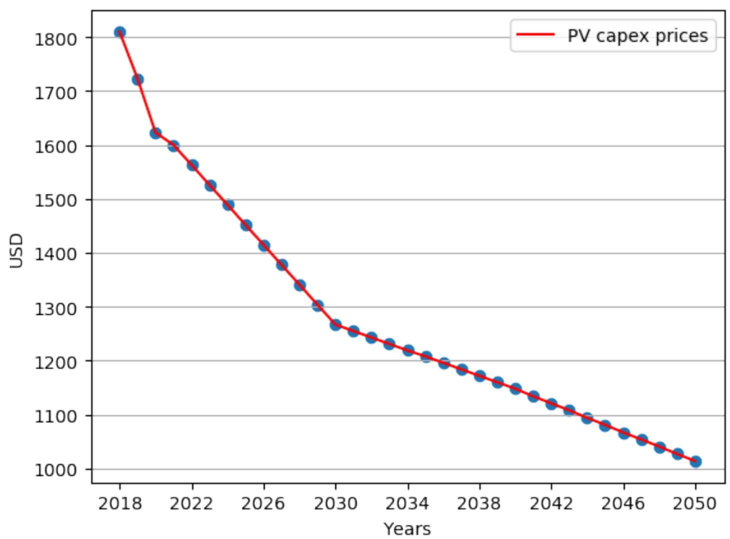
$$\sum_{t=1}^{T} \epsilon_t \leq \kappa \sum_{t=1}^{T} D_t^{dr} (2.29)(2.29)$$

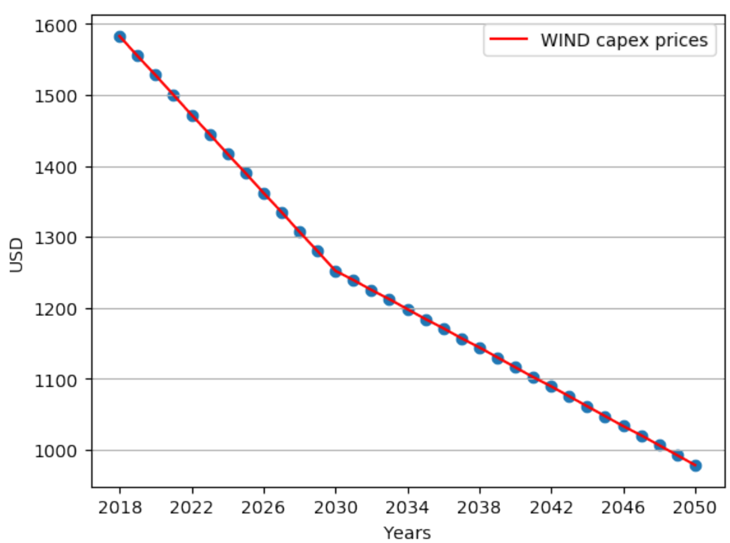
It is important to notice that Equation (2.9) establish a constraint to guarantee that the sum of the demand with flat tariff (base case) is equal to the sum of the demand after the application of any of the DSM strategies. However, the DLCt strategy need to violate this constraint, otherwise the only way to guarantee that the base case demand is equal to the demand with DSM is by making the variable ϵ_t equal to zero. In order to avoid making ϵ_t equal to zero the methodology removes constraint (2.9) for the DLCt DSM strategy.

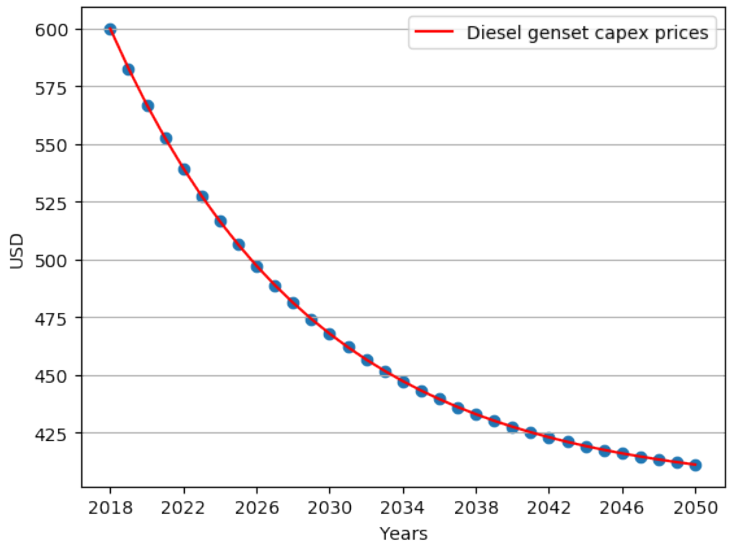
2.6 Multiyear analysis

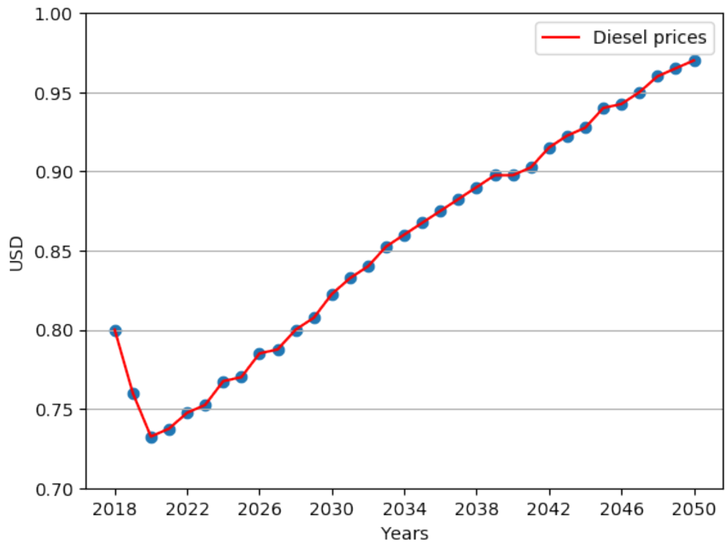
Most of the methodologies found in literature to compute the sizing of IMGs consider one single year for the analysis. However, by considering this, these methodologies are implicitly assuming that the capital, operational, and maintenance expenditures will remain constant during the lifetime of the projects (20 to 25 years). These kinds of methodologies only consider the interest rate to compute future values of capital, operational, and maintenance. Nevertheless, this is not a straightforward justifiable assumption, especially considering that renewable energy sources' costs are decreasing fast in the last years. Moreover, new policies taxing carbon emissions can significantly benefit renewable energy projects in the future. To have a better understanding of the variations in prices in the future, the following figure is introduced.

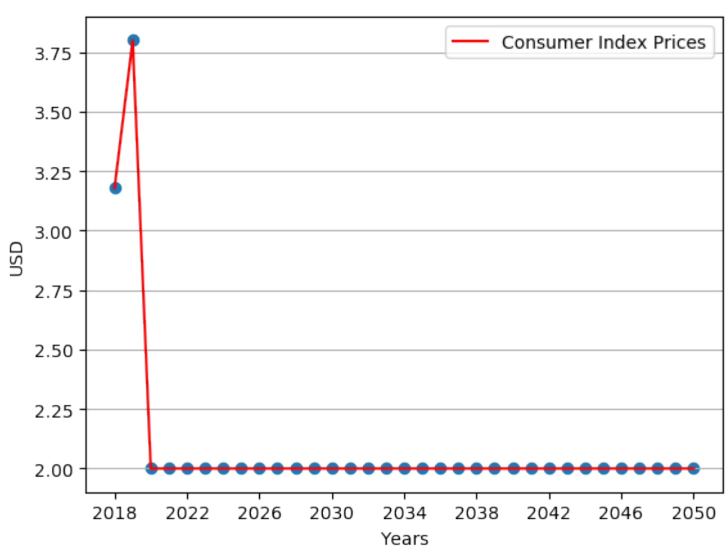

BESS capex prices
USD
350
300
250
200
150
2018
2022
2026
2030
2034
2038
2042
2046
2050
Years


PV capex prices
USD
1800
1700
1600
1500
1400
1300
1200
1100
1000
2018
2022
2026
2030
2034
2038
2042
2046
2050
Years


WIND capex prices
USD
1600
1500
1400
1300
1200
1100
1000
2018
2022
2026
2030
2034
2038
2042
2046
2050
Years


Diesel genset capex prices
USD
Years
600
575
550
525
500
475
450
425
2018
2022
2026
2030
2034
2038
2042
2046
2050


Diesel prices
USD
Years
1.00
0.95
0.90
0.85
0.80
0.75
0.70
2018
2022
2026
2030
2034
2038
2042
2046
2050


Consumer Index Prices
USD
Years
3.75
3.50
3.25
3.00
2.75
2.50
2.25
2.00
2018
2022
2026
2030
2034
2038
2042
2046
2050

Hint:

- BESS capital expenditures [atb_data],
- PV capital expenditures [atb_data],
- Wind capital expenditures [atb_data],
- Diesel generator capital expenditures.
- Diesel price [energy_outlook],
- Carbon Tax price [tax_data].

These figures shows the trends in capital expenditures for different energy sources for the following years. By using a multiyear analysis, it is possible to capture those trends in the prices. However, a methodology that uses one single year approach can not incorporate these trends. This assumption does not seem appropriate for future replacements of the energy sources.

Reference [Pecenak2019] classifies multiyear methodologies in two main categories: the forward-looking model and the adaptive model. On one side, the forward-looking model deals with an optimization formulation that has as a horizon the lifetime of the IMG project (20 to 25 years). This approach has the advantage of being able to integrate future information. However, the enormous size of the optimization formulation can make the problem difficult to solve. Additionally, the formulation will require binary variables to integrate the technologies' replacement, which adds even more complexity to the problem. On the other side, the adaptive model uses a rolling horizon of smaller windows of time (usually one year). This approach does not require binary variables, which represents an advantage. The model easily integrates growth of demand, price forecasts, and energy resources. Additionally, this approach does not require to modify the optimization formulation. Instead, it solves a single year optimization until it reaches the project's lifetime.

CVXMG chooses to work with the adaptive method. However, despite its advantages, the implementation of the model requires careful attention to previous years' input parameters. The investment decisions of previous years should be known for the model in each window of time. The following algorithm shows a simplified step by step guide for the multiyear analysis. The following lines provide a brief description of each line of the algorithm.

```
MultiyearDSMS

Inputs: Weather, forecasted acquisition prices of energy sources, forecasted fuel␣
↪prices over the lifetime of the IMG project.
Outputs: Tariffs of energy, yearly acquisition, yearly dispatch of energy sources␣
↪over the life time of the IMG project.

prob_info = Set problem information
historic_data = Save historic weather, save demand data
synthetic_data = create_synthetic_data(historic_data)
for year = 0 in range(lifetime):
    prev_data = Read results of previous years
    act_param = Actualize solver parameters
    resul = yearly_solver(prob_info, synthetic_data[year], prev_data, act_param)
    summary[year] = resul
```

2.6.1 Set problem information

This line saves the configuration of the analysis in the variable prob_info. This variable contains a list of the energy sources that the optimization includes, the technical and economic characteristics of those energy sources, the lifetime of the project, and interest rate. This variable contains all the information about the multiyear analysis.

2.6.2 Save historic weather and demand data:

This line reads the historic weather and electrical demand data. Afterwards the data is stored in the variable historic_data.

2.6.3 create_synthetic_data(historic_data):

This line creates the synthetic data for the multiyear optimization formulation. A single year approach can use the historical data (of one year). However, to build the multiyear optimization formulation, synthetic data is required for the project's lifetime. The function create_synthetic_data takes as inputs the historical data of weather and electrical demand profiles (one year) and returns as output the synthetic data over the lifetime of the project (20 or 25 years). The function follows a four-step process to create synthetic data for the optimization formulation:

1. Divide the historical data by months.
2. Take the data of each month and group it by hours.
3. Fit each hour group to the probability distributions recommended by the literature to each kind of data (Weibull for wind, Beta for Global Horizontal Radiation, log-normal for the demand, amongst others).
4. Build the synthetic new profiles by random sampling the fitted probability distributions at each hour and month.

The above-described process is similar to a Gaussian process without a covariance matrix. Two main reasons force to adopt the above-described process and not the well know Gaussian process. The first reason is that the Gaussian process can model only processes that follow a Gaussian distribution. This limitation forces the study to assume that the wind and Global Horizontal Radiation (GHI) follow a Gaussian distribution, which is not accurate. The second reason is that fitting and sampling a Gaussian process consume more computational power and requires more time to build synthetic data than the above-described process. Equation (2.30) describes the sampling process to create the synthetic data.

$SD_t|m, h \sim f(\psi_{m,h})$(2.30)(2.30)

where SD_t represents the Synthetic Data at time t. This variable represents the electrical demand, wind speed, global horizontal radiation, temperature, and others. $\psi_{m,h}$ represents the monthly/hourly fitted distributions using the historical data.

2.6.4 Read results of previous years:

This line read the results of the previous years and store the values in variable prev_data. This variable contains the capacities of the energy sources acquired in the past. Additionally, this variable contains a detailed register of the costs paid for buying those energy sources in previous years.

2.6.5 Actualize solver parameters:

This part of the algorithm actualizes the cost parameters of the solver. These parameters include the acquisition costs of the energy sources and the fuel costs of that year in particular.

2.6.6 yearly_solver(prob_info, synthetic_data[year], prev_data, act_param):

The function $yearly\_solver$ contains the formulation described at the beginning of this section (Equations (2.2) to (2.29)). This function solves the DCSP optimization formulation for over one year. This function returns the capacities of the energy sources to install in that year, the dispatch of the energy sources and the energy tariffs for the customers. Additionally, this function returns the payments of each one of the stakeholders of the project.

2.6.7 summary[year] = resul:

This line save the results of $yearly\_solver$. Summary is a list that contains the results of each year.

2.7 Stochastic multiyear analysis

The study proposes a stochastic analysis to deal with the uncertainties of electric demand, weather variables, and future prices. The stochastic approach uses a Montecarlo Sampling (MCS) approach. The MCS approach creates random samples of the probability distribution functions using Equation (2.30) to build the scenarios. The following algorithm describes the multiyear stochastic analysis.

Inputs: Weather, forecasted acquisition prices of energy sources, forecasted fuel prices over the lifetime of the IMG project. Outputs: Tariffs of energy for the customers, average yearly acquisition, yearly dispatch of energy sources over the life time of the IMG project.

```
prob_info = Set problem information
historic_data = Save historic weather and demand data
synthetic_data = create_synthetic_data(historic_data)

for scenario in range(scenarios):
    for year in range(lifetime):
        prev_data = Read results of previous years
        act_param = Actualize solver parameters
        resul = yearly_solver(prob_info, synthetic_data[year], prev_data, act_param)
        summary[year] = resul
    total_summary[scenario] = summary
```

The above algorithm uses the multiyear analysis in its core. The only difference with the multiyear analysis is an additional loop. The stochastic multiyear solves one multiyear problem for each scenario that the MCS approach builds. Variable summary stores the results of installing and operating the IMG each year of the simulations. Variable total_summary stores the results of installing and operating each of the scenarios of the stochastic analysis. In the end, the results are the average of all the simulations.

2.8 Energy sources models

2.8.1 Photovoltaic system

References [Li2017], [Zhang2016], [lasnier1990] describe the output power $E_{PV,t}$ of a N_{PV} number of photovoltaic panels as:

$$E_{PV,t} = N_{PV}\rho_{PV}P_{STC}\frac{G_{A,t}}{G_{STC}}(1 + C_T(T_{C,t} - T_{STC}))(2.31)(2.31)$$

where ρ_{PV}, P_{STC}, $G_{A,t}$, G_{STC}, and C_T are the derating factor (unitless), output power of the PV module (kW), GHI (kW/m^2), GHI at standard conditions (kW/m^2), and temperature coefficient of the PV module ($\%/°C$), respectively. $T_{C,t}$ is the working temperature of the PV cell at hour t ($°C$), and T_{STC} is the temperature at standard conditions ($°C$). Reference [Skoplaki2009] describes $T_{C,t}$ as a function of the ambient temperature and incident solar radiation over the PV module.

$$T_{C,t} = T_{A,t} + \frac{G_{A,t}}{G_{NOCT}}(T_{NOCT} - T_{a,t,NOCT})(2.32)(2.32)$$

where G_{NOCT}, T_{NOCT} and $T_{a,t,NOCT}$ are the solar radiation (kW/m^2), working temperature ($°C$) and ambient temperature ($°C$) at Nominal Operational Cell Temperature (NOCT) conditions [Duffie2013], [librosolar].

2.8.2 Battery energy storage system

A battery is an element strongly coupled in time [Xiaoping2010]. The lack or excess of energy in one hour can be demanded or stored in the battery. To guarantee that the battery is not charged and discharged simultaneously, the BESS model can integrate binary variables. However, as discussed before, the proposed methodology tries to avoid using binary variables. The methodology proposes to model the BESS as an accumulator to avoid using binary variables. The battery is a deposit to store something temporarily. The deposit can *charge* if there is still space available, and *discharge* when required. Operations Research modeled this problem long before, and it is well known as the inventory problem [Silver2008].

The model of the BESS does not use separate optimization variables for charging and discharging of the BESS. Instead uses one single variable for the dispatch that controls the residual energy of the battery [Zhang2018ab]. Equation (2.33) presents a simple way of defining the residual energy in a BESS.

$$RE_{B,t} = SOC_t C_B(2.33)(2.33)$$

If the following state of the residual energy is superior to the previous, the battery was charged $E_{B,t}$ units during time t. If the following state of the residual energy is inferior to the previous, the battery was discharged $E_{B,t}$ units during time t. Equations (2.34) and (2.35) show this.

$$RE_{B,t+1} = RE_{B,t} + E_{B,t}(2.34)(2.34)$$

$$RE_{B,t+1} = RE_{B,t} - E_{B,t}(2.35)(2.35)$$

Equation (2.36) describes the initial residual energy of the BESS. The simulations assume that the battery starts half charged (50% of its nominal capacity). Additionally, the simulation assumes that the minimum level of discharge of the battery is 50% and that the maximum level of charge is 100% of its nominal capacity. Equation (2.37) describes those limits. Moreover, the simulations consider the maximum rate of charge and discharge of the battery. The simulation assumes that the maximum charge and discharge rate in each time slot is 30% of its nominal capacity. For all the simulations, the slot of time is one hour. Equation (2.38) and (2.39) describes the limits of charge and discharge of the battery for each time slot, respectively.

$E_{B,0} = 0.5C_B$(2.36)(2.36)

$0.5C_B \leq RE_{B,t} \leq C_B$(2.37)(2.37)

$E_{B,t+1} \geq E_{B,t} - 0.3C_B$(2.38)(2.38)

$E_{B,t+1} \leq E_{B,t} + 0.3C_B$(2.39)(2.39)

2.8.3 Diesel generator

The fuel consumption of a diesel generator is a function of its capacity and output power. This function uses linear or quadratic formulations [Arun2008], [Ashok2006]. Reference [Scioletti2017] makes a quadratic fit to estimate α, β, and γ parameters as a function of the capacity of the generator using manufacturer-provided fuel consumption data. Bukar et al. use a linear approximation to describe the diesel consumption of a Diesel Generator [Bukar2019]. Equation (2.40) describes the function that [Bukar2019] use.

$F_{DG,t} = 0.246E_{DG,t} + 0.08415C_{DG}s$(2.40)(2.40)

where, $E_{DG,t}$, $F_{DG,t}$, and C_{DG} denote the generated power (kW), the fuel consumption (L/hour), and the installed capacity (kW) of the diesel generator.

subsection{Wind generator}

The output power of a wind turbine is a function of the wind speed and its rated capacity. Equation (2.41) presents a well-accepted model to compute the output power of a wind turbine [Ramli2018], [Kaabeche2017]. The proposed methodology uses this model.

$$E_{WT} = \begin{cases} 0, & V_{A,t} < V_{cut-in}, V_{A,t} > V_{cut-out} \\ V_{A,t}^3 \left(\frac{E_{WT,R}}{V_{Rated}^3 - V_{cut-in}^3} \right) - E_{WT,R} \left(\frac{V_{cut-in}^3}{V_{Rated}^3 - V_{cut-in}^3} \right), & V_{cut-in} \le V_{A,t} < V_{Rated} \\ E_{WT,R}, & V_{Rated} \le V_{A,t} < V_{cut-out} \end{cases} \quad (2.41)(2.41)$$

$V_{cut-in} \le V_{A,t} < V_{Rated} E_{WT,R},$

(2.41)

where $V_{A,t}$ is the wind speed (m/s), $E_{WT,R}$ is the rated power (kW), V_{cut-in}, V_{Rated}, $V_{cut-out}$ represent the cut-in, nominal and cut-out speed of the wind turbine (m/s), respectively.

2.9 Classes, Objects, and Methods of CVXMG

2.10 References

In here the list of references used in this documentation:

Bibliography

[pece2019] 26. K. Pecenak, M. Stadler, and K. Fahy, “Efficient multi-year economic energy planning in microgrids,” Appl. Energy, vol. 255, no. August, p. 113771, 2019.

[ov2020] Oviedo Cepeda, J.C.; Osma-Pinto, G.; Roche, R.; Duarte, C.; Solano, J.; Hissel, D. Design of a Methodology to Evaluate the Impact of Demand-Side Management in the Planning of Isolated/Islanded Microgrids. Energies 2020, 13, 3459.

[bu2019] A. L. Bukar, C. W. Tan, and K. Y. Lau, “Optimal sizing of an autonomous photovoltaic/wind/battery/diesel generator microgrid using grasshopper optimization algorithm,” Sol. Energy, vol. 188, no. March, pp. 685–696, 2019.

[Celik2017] B. Celik, R. Roche, S. Suryanarayanan, D. Bouquain, and A. Miraoui, “Electric energy management in residential areas through coordination of multiple smart homes,” Renew. Sustain. Energy Rev., vol. 80, no. April 2016, pp. 260–275, 2017.

[Inversin2000] A. R. Inversin, “Mini-grid design manual,” 2000.

[Baatz2017] B. Baatz, “Rate Design Matters: The Intersection of Residential Rate Design and Energy Efficiency,” 2017.

[Glick2014] D. Glick, M. Lehrman, and O. Smith, “Rate Design for the Distribution Edge,” Rocky Mt. Inst., no. August, 2014.

[Kostkova2013] K. Kostkova, L. Omelina, P. Kycina, and P. Jamrich, “An introduction to load management,” Electr. Power Syst. Res., vol. 95, pp. 184–191, 2013.

[Wong2012] C. Joe-Wong, S. Sen, S. Ha, and M. Chiang, “Optimized day-ahead pricing for smart grids with device-specific scheduling flexibility,” IEEE J. Sel. Areas Commun., vol. 30, no. 6, pp. 1075–1085, 2012.

[Borenstein2002] S. Borenstein, M. Jaske, and A. Rosenfeld, “Dynamic Pricing , Advanced Metering and Demand Response in Electricity Markets,” 2002.

[Ali2015] A. Ali, J. Z. Kolter, S. Diamond, and S. Boyd, “Disciplined convex stochastic programming: A new framework for stochastic optimization,” in Uncertainty in Artificial Intelligence, 2015, no. 3, pp. 62–71.

[Liberti2008] L. Liberti and N. Maculan, Global optimization, from theory to implementation, vol. 105, no. 28. Springer, 2008.

[Chauhan2014] A. Chauhan and R. P. Saini, "A review on Integrated Renewable Energy System based power generation for stand-alone applications: Configurations, storage options, sizing methodologies and control," Renew. Sustain. Energy Rev., vol. 38, pp. 99–120, 2014.

[Diaf2008] S. Diaf, G. Notton, M. Belhamel, M. Haddadi, and A. Louche, "Design and techno-economical optimization for hybrid PV/wind system under various meteorological conditions," Appl. Energy, vol. 85, no. 10, pp. 968–987, 2008.

[atb_data] National Renewable Energy Laboratory.Annual Technology Baseline data. 2019.

[energy_outlook] Energy Information Administration.Anual Energy Outlook 2020. 2020.

[tax_data] Canadian government.Greenhouse Gas Pollution Pricing Act. 2018.

[Pecenak2019] 26. K. Pecenak, M. Stadler, and K. Fahy, "Efficient multi-year economic energy planning in microgrids," Appl. Energy, vol. 255, no. August, p. 113771, 2019.

[Li2017] B. Li, R. Roche, D. Paire, and A. Miraoui, "Sizing of a stand-alone microgrid considering electric power, cooling/heating, hydrogen loads and hydrogen storage degradation," Appl. Energy, vol. 205, no. July, pp. 1244–1259, 2017.

[Zhang2016] J. Zhang et al., "A Bi-Level Program for the Planning of an Islanded Microgrid Including CAES," IEEE Trans. Ind. Appl., vol. 52, no. 4, pp. 2768–2777, 2016.

[lasnier1990] F. Lasnier, Photovoltaic Engineering Handbook. Taylor & Francis, 1990.

[Skoplaki2009] E. Skoplaki and J. A. Palyvos, "Operating temperature of photovoltaic modules: A survey of pertinent correlations," Renew. Energy, vol. 34, no. 1, pp. 23–29, 2009.

[Duffie2013] J. A. Duffie and W. A. Beckman, Solar Engineering of Thermal Processes, 4th Editio. Wiley, 2013.

[librosolar] Tomas Markvart, ed.Solar electricity. 2nd Editio. Wiley, 2000, p. 298.ISBN:9780471988526.

[Xiaoping2010] L. Xiaoping, D. Ming, H. Jianghong, H. Pingping, and P. Yali, "Dynamic Economic Dispatch for Microgrids Including Battery Energy Storage," in International Symposium on Power Electronics for Distributed Generation Systems, 2010, no. 2, pp. 914–917.

[Silver2008] E. A. Silver, "Operations Research in Inventory Management: A Review and Critique," Oper. Res., vol. 29, no. 4, pp. 628–645, 2008.

[Zhang2018ab] Xu Zhang et al. "A novel method for lithium-ion battery state of energy andstate of power estimation based on multi-time-scale filter". In:Applied Energy216.February (2018), pp. 442–451.ISSN: 03062619.

[Arun2008] P. Arun, Rangan Banerjee, and Santanu Bandyopadhyay. "Optimum sizing ofbattery-integrated diesel generator for remote electrification through design-space approach". In:Energy33.7 (2008), pp. 1155–1168.ISSN: 03605442.

[Ashok2006] S Ashok. "Optimised model for community-based hybrid energy system".In:Renewable Energy32 (2006), pp. 1155–1164.

[Scioletti2017] M. S. Scioletti, A. M. Newman, J. K. Goodman, A. J. Zolan, and S. Leyffer, "Optimal design and dispatch of a system of diesel generators, photovoltaics and batteries for remote locations," Optim. Eng., vol. 18, no. 3, pp. 755–792, 2017.

[Bukar2019] A. L. Bukar, C. W. Tan, and K. Y. Lau, "Optimal sizing of an autonomous photovoltaic/wind/battery/diesel generator microgrid using grasshopper optimization algorithm," Sol. Energy, vol. 188, no. June, pp. 685–696, 2019.

[Ramli2018] M. A. M. Ramli, H. R. E. H. Bouchekara, and A. S. Alghamdi, "Optimal sizing of PV/wind/diesel hybrid microgrid system using multi-objective self-adaptive differential evolution algorithm," Renew. Energy, vol. 121, pp. 400–411, 2018.

[Kaabeche2017] A. Kaabeche, S. Diaf, and R. Ibtiouen, "Firefly-inspired algorithm for optimal sizing of renewable hybrid system considering reliability criteria," Sol. Energy, vol. 155, pp. 727–738, 2017.

[Dole2004] D. Dole and I. Bartlett, "Beyond cost recovery: setting user charges for financial, economic, and social goals," 2004.

Python Module Index

c

Index

C