ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

SCHOOL OF ENGINEERING AND ARCHITECTURE

Department of Electrical, Electronics, and Information Engineering “Guglielmo Marconi”- DEI

DEGREE PROGRAM
Automation Engineering

THESIS TITLE

Optimization model of an electric vehicle parking lot for the provision of flexibility services to the distribution system operator

Supervisor
Prof. Alberto Borghetti

Co-supervisors
Prof. Fabio Napolitano
Prof. Fabio Tossani

Presented by
Jereney Johnson
Matricola: 0000925431

Academic Year- 2022/23

Abstract

This thesis focuses on the optimization of a parking lot equipped with charging stations for electric vehicles (EVs) to provide flexibility services to the distribution system operator (DSO). The proposed models consider the charging and discharging of EVs, and the use of their batteries to provide flexibility services. The objective is to minimize the consumption of the energy from the grid while ensuring that the DSO's flexibility requirements are met. The models take into account the uncertainty in EV's charging and discharging patterns. Four models are presented, and the performances are illustrated for a case study with different scenarios. The results show that the proposed model can effectively provide flexibility services to the DSO while providing maximum reduction of energy from the grid. The presence of an auxiliary battery is also considered to reduce the influence of the uncertainties associated with the number and characteristics of the parked EVs. The study provides insights into the potential benefits of EV parking lots in providing flexibility services to the electricity grid.

Keywords: electric vehicles, EV parking lot, flexibility services, mathematical programming, optimization

Contents

Chapters

List of figures

Chapter 1
Introduction

General overview of the thesis topic

There are two clear tendencies that affect or are expecting to affect the operation of electric power distribution systems:

- the installation of distributed generation, i.e., generation units connect directly to the medium voltage (MV) or even low voltage (LV) distribution networks, which use renewable energy resources, mainly solar;
- the increase adoption of electric vehicles for private mobility.

These two tendencies make the forecasting of system operating conditions more difficult and require the installation of storage systems. Since these systems are expensive, a cheaper alternative for the distribution system operator is the procurement of flexibility services from the network users.

The flexibility services consist in the stated and verified willingness of the user to increase or decrease active and reactive power injections considering a predefined time horizon.

Among the users that can provide active power flexibility services are the parking lots equipped with several charging stations. When there are cars connected to the charging stations, which can be even bidirectional, the parking lot as a whole (managed by a, so called, aggregator) can postpone or anticipate the charging/discharging processes to reduce or increase the load consumption from the network, following the distribution system operator requests, using the energy storage provided by the vehicles' batteries.

Specific scope of the thesis

The scope of this thesis is the development and the test of optimization procedures for the calculation of the flexibility margin by the parking lot aggregator.

An optimization model for a parking lot of electric vehicles would aim to maximize the utilization of the available charging stations, by minimizing the energy procurement costs needed to satisfy the requests of the customers, i.e. the electric vehicle users. The following are the key components that should be included in the optimization model for the management of the parking lot by the aggregator:

1. Demand forecast: the model would need to estimate the number of electric vehicles that are likely to use the parking lot is all times during the day. This could be based on historical data, surveys, or other relevant means (e.g., booking services).
2. Charging schedule: the model needs to optimize the charging schedule of the electric vehicles to ensure that all vehicles are fully charged when their owners return to the parking lot. This involves predicting the time that each vehicle will be parked and determining the optimal charging schedule based on the available charging stations.
3. Flexibility: the model needs to calculate the maximum power reduction in each time of the optimization horizon that can be offered to the distribution system operator, with a reward.

The thesis does not address the topic of pricing strategy, i.e., the choice of the optimal pricing strategy for charging services to ensure that the parking lot is profitable while also being competitive with other charging stations in the area. This could be based on factors such as the cost of electricity, the cost of maintaining the charging infrastructure, and the demand for charging services. In the thesis, both the price of the energy procurement and the reward rate for the provision of the active power reduction flexibility service are considered predefined by the utility or the energy authority, although they can be different in different hours of the day.

Overall, the optimization model for a parking lot for electric vehicles would need to balance the needs of the customers (e.g., availability of charging stations, convenience, affordability) with the goals of the parking lot operator (e.g., maximizing utilization, profitability).

Structure of the thesis

Chapter 2 is devoted to a review of the flexibility services, with specific reference to the case of parking lots equipped with several charging stations. For the scope of this thesis the behaviour of parking lot is described with an aggregate model, neglecting the specific charging and discharging operation of the single changing stations. We assume that an aggregator is responsible for the interface with the utility network, both for the procurement of the energy needed to charge the vehicles, exploiting the differences of price in the various time periods, and for the provision of the power reduction service at the requests of the distribution system operator.

Chapter 3 is devoted to the presentation of the optimization models. Four models are considered. The first minimizes the energy procurement costs for each considered scenario that describes the presence and characteristics of the parked vehicles during the next day. The

second model extends the analysis by including the calculation of a common profile of electric power consumption for the next day that can be declared to the distribution system operator as reference profile. The third model calculates the maximum power reduction that can be offered with respect to the reference profile, under the assumption of a single power request by the distribution system operator during the day. The fourth model considers the presence of an auxiliary battery storage system in the parking lot, which is used to compensate the uncertainty of the number and characteristics of the parked vehicles during the day.

Chapter 4 presents the implementation of the optimization models by using Pyomo, which is a Python library for the description of mathematical optimization models and their solution by using appropriate solvers. The models considered in this thesis are quadratic in the objective function with linear constraints without binary or integer variables.

Chapter 5 presents the results of some numerical tests. The numerical test includes checking the variation of the objective value and the variables by changing the parameter values. Also, for each of the models, numerical results, graphs, and comparisons among the models are described.

Chapter 6 is devoted to the conclusions and to the description of the possible improvements of the developed models.

Chapter 2

Parking lot with several charging stations as new load for the power network

This chapter describes the operation of a parking lot with several charging stations and the possibility to provide flexibility services to the distribution system operator.

2.1 Integration of charging stations of electric vehicles in microgrids with renewable power generation

Fig. 2.1 illustrates a microgrid that includes an integrated system of renewable generation (e.g., PV panels), stationary BES units, local load, and the EV parking lot.

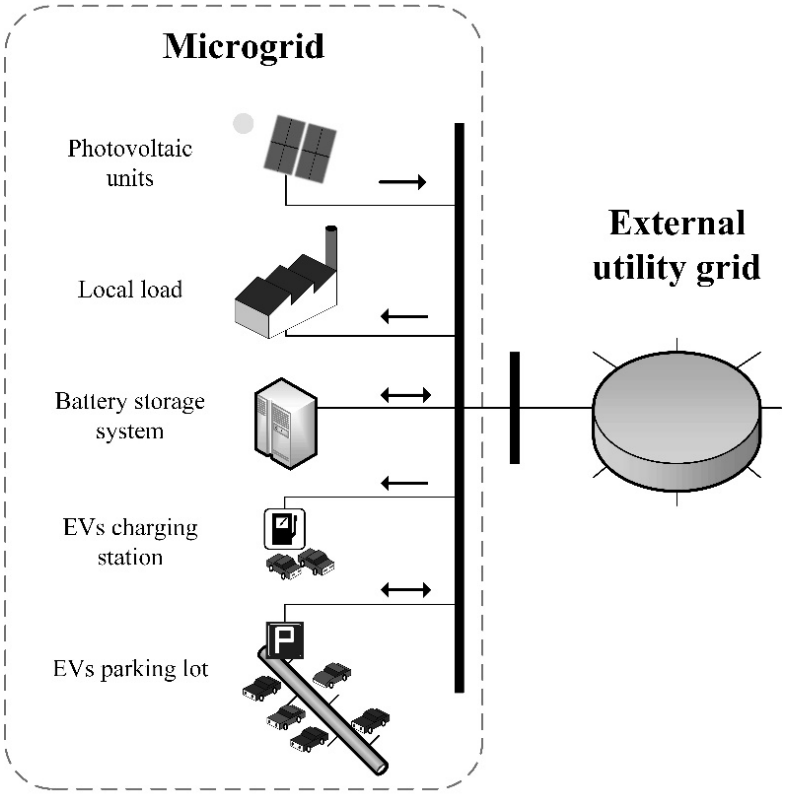


Figure 2.1 - Scheme of a microgrid with both local generation and EV charging stations.

The integration of charging stations of electric vehicles (EVs) in microgrids with renewable power generation is an important step towards achieving a sustainable and reliable energy system. Microgrids are small-scale power grids that can operate independently or in

connection with the larger grid. The energy procurement cost associated with the operation of the considered site is minimized by a central dispatching system.

The literature on the approaches to manage the charging of electric vehicles is quite large, as shown in the recent survey presented in [1]. An analysis of the advantages and drawbacks of different approaches to the integration of EVs is presented in [2]. It is generally acknowledged that a specific operating strategy is needed in order to avoid the overload of the network in case of many vehicles charge at the same time. The presence of fast charging stations with rating of several tens of kW make the problem more important, as shown, in e.g., [3] that presents a study of the state-of-art of fast charging stations including experimental tests. In [3] provides valuable insights into the practical implementation of fast charging stations and energy storage technologies in a smart micro grid, highlighting the importance of such technologies in managing the load on the grid during peak charging periods and maximizing the efficiency of the system.

The integrated operation of parking facilities with renewable energy resources has been studied in, e.g., [4]. In general, the idea behind the integrated operation is the attempt to use the storage capability of parked vehicles to compensate the fluctuations of the production from renewable resources, with specific reference to photovoltaic (PV) generation. An evaluation of the integration of charging stations with PV systems, in order to cope with the fluctuation of solar irradiance, has been performed in [5], with the presentation of specific strategies and power electronic components.

A specific characteristics of parking lots is that the presence of vehicles connected to the charging stations depends on the number of arrivals and departures in each period. These numbers are stochastic. An approach that takes into account the uncertainties of electric vehicle arrivals and departure, as well as the grid power price, has been presented in [6]. The scheduling of the electric vehicles fully exploits the production from renewable resources and the periods when the power price is lower.

The case of the integration of electricity and heat generation, electrical loads, PV units and charging stations in an industrial microgrid is presented in [7]. Also, the benefits of the coordination approach, such as the ability to reduce the peak demand on the grid, improve the utilization of renewable energy sources.

2.2 Impacts of electric vehicle charging on the state of the distribution network and countermeasures.

As mentioned, introducing a parking lot with several charging stations can have a significant impact on the power network, especially if the charging stations are high-capacity and if a large number of electric vehicles are being charged simultaneously.

The ways in which a parking lot with charging stations affect the power network are:

- Increased demand: each charging station draws a significant amount of power, especially if it is a fast charger. As a result, a parking lot with multiple charging stations can significantly increase the overall power demand in the area.
- Peak demand: the charging stations are likely to be used during peak periods, such as in the morning or evening when people are commuting to and from work. This can create a peak demand on the power network, which can be challenging for utilities to manage.

The typical countermeasures against the negative effects on the operation of the distribution network are:

- Load balancing: to manage the increased demand, utilities introduce load balancing measures, such as time-of-use pricing or demand response programs, which incentivize customers to charge their vehicles during off-peak periods.
- Infrastructure upgrades: in some cases, the power network may require upgrades to support the increased demand from the charging stations. This could involve installing new transformers, upgrading distribution lines, or building new substations.
- Renewable energy integration: parking lots with charging stations could be an opportunity to integrate more renewable energy into the power network. For example, solar panels could be installed on the roof of the parking lot to offset the energy consumption of the charging stations.

Overall, introducing a parking lot with charging stations can have a significant impact on the power network. Utilities and policymakers will need to work together to manage the increased demand, balance the load, and ensure that the necessary infrastructure upgrades are made to support the growth of electric vehicles.

Another countermeasure is the exploitation of vehicle to grid capabilities. The charging stations may be made bidirectional, so there is the possibility not only to charge the vehicle battery but also to discharge the battery and inject back the power to the network or use it to charge other vehicles with more stringent requirements.

2.3 Vehicle to grid

Another countermeasure is the exploitation of vehicle to grid (V2G) capabilities. The charging stations may be made bidirectional, so there is the possibility not only to charge the vehicle battery but also to discharge the battery and inject back the power to the network or use it to charge other vehicles with more stringent requirements.

An optimization model for the assessment of the contribution of V2G systems has been proposed in [8]. The presented procedure allows the minimum cost management of the microgrid that includes both the bidirectional charging stations and the renewable energy production. The set of vehicles connected to the bidirectional charging stations are operated as a storage system, compensating the fluctuations of the renewable production.

When the number of charging stations is large, the methods presented in the literature includes often a specific operator, called aggregator. The method described in [9] contemplates the presence of an aggregator acting as an intermediate agent between end-users and the distribution system operators. End users include the clusters of charging stations.

Specific contributions refer to the use of tariffs rates designed to facilitate the installation of bidirectional charging stations and their use in V2G mechanisms. For example, a study of the feasibility of premium tariff rates for V2G services is similar to feed-in-tariff (FIT) programs for renewable energy generation, has been presented in [10].

2.4 Uncertainties associated with the operation of a cluster of charging stations

To cost-effectively operate the microgrids equipped with clusters of charging stations, other than the renewable generation and the typical load, it is crucial to consider the uncertainties associated with the presence and state of the vehicles in the parking lot.

The uncertainties are associated with:

- number of vehicles entering the parking lot at each time;
- number of vehicles leaving the parking lot at each time;
- characteristics of the vehicles, such as the size of the battery;
- initial level of the charge when the vehicle enters in the parking lot;
- level of the charge that should be guarantee to the vehicle when it leaves the parking lot.

In order to deal with all these uncertainties, the aggregated model used in this thesis makes

reference to the linear programming model presented in [11]. Different stochastic models for the optimal operation of electric vehicle parking lots in microgrids are those presented in [8] and [12]. Specific approaches have been proposed in [13] for the integration of parking lots in the main power system, with the coordination with the production of large wind farms and the unit commitment of thermal and hydro power plants.

To achieve the objectives of the optimal operation of a microgrid, energy management systems implement smart charging approaches to align the charging and dispatching processes with the optimization objectives of the site. In [16] it is described the construction of a scenario tree obtained by means of the scenario reduction technique. The tree is employed by the energy management system to solve the day-ahead scheduling of the energy resources in the site. The scenario tree is built by means of a reduction technique based on k-medoids so that all the representative scenarios included in the tree are feasible. The operator of the parking lot commonly seeks an economic benefit, whilst offering the users the option of charging their vehicles at the lowest possible cost. The objective is the minimization of the expected daily procurement costs. The use of the initial energy in the electric vehicles entering the parking is limited. The model considers other typical constraints (such as maximum number of available charging stations, size of the vehicle batteries, and the power ratings of charging stations). Moreover, a procedure aimed at guaranteeing a feasible solution for the operation of the site is introduced. The day-ahead optimal scheduling of a parking lot with several bidirectional charging stations for plug-in electric vehicles, part of a grid-connected system, is obtained by a central dispatching unit that implements a multistage stochastic optimization considering the uncertainties of connected electric vehicles. Some of the scenarios of the tree obtained by applying the approach presented [16] are used in this thesis as described in Chapter 3.

2.5 Capability of electric vehicle parking lots to provide flexibility services to the distribution system operator

It is a burden on utility companies to ensure that the electricity demand is always met. The increasing integration of large amounts of intermittent renewable energy resources into the distribution grid and the increased use of electric vehicle brings some emerging economic and technical challenges. The intermittent and stochastic nature of most renewable generation can

lead to changing temporal dynamics of net demand, increasing its variability and unpredictability. The characteristics of the renewable generation can undermine and make uncertain the satisfaction of the energy demand and the maintenance of the stability of the power supply system, both in terms of voltage and frequency regulation.

To counteract the negative effects of a high penetration of renewable resources and the load due to charging stations, it is necessary to increase the flexibility of both planning and operation of the power system. In fact, it is widely recognized that increased flexibility is the key to reliable operation of a modern electricity system. Flexibility services can be offered by individual providers such as: passive end users via demand response programs, active end users with distributed generators, energy communities and electric vehicle aggregators, which are available to vary the injected/absorbed powers following requests from both the distribution system operator (DSO) and the transmission system operator (TSO).

Some flexibility services with specific reference to the balancing of active power and to voltage regulation are provided by the electric vehicle aggregators, to reduce the impact of the simultaneous recharging on the functioning of the distribution network. The aggregation functions of the operation of the charging stations of a parking lot can be incorporated into the framework of the energy community.

In [14] the storage capability of electric vehicle parking lot is represented so that the parking lot can participate in a demand response scheme. In this scheme, price signals are calculated by the distribution system operator and broadcasted to all the customers. The customers reduce the load demand in periods of high price and recover the consumption in period of low price. This scheme may be effective for the case of parking lots, particularly if bidirectional charging stations are used, so that it is possible to reduce temporarily the load consumption and provide the energy needed to the vehicles leaving the parking lot by using the energy already stored in the other cars.

New regulatory frameworks have been set up to increase final user participation in the electricity market in several parts of the world. Emerging regulations are fostering the participation of final users, single or aggregated collectives, in both the energy market and the ancillary services markets. As an example, the Italian Regulatory Authority for Energy, Networks and Environment (ARERA) has issued a call for projects for the provisions of local ancillary services (resolution August 3, 2021 352/2021/R/eel), i.e. those useful for the operation of distribution networks, that completes a previous resolution (May 5 2017 300/2017/R/eel) relevant to global ancillary services, i.e. those acquired by the transmission system operator, in the framework of the electricity market regulation (July 23, 2019 322/2019/R/eel).

As described in Chapter 3, this thesis focuses on the flexibility service that is the reduction of the load consumption by the parking lot as soon as a specific request is made by the distribution system operator. In order to provide this service, the parking lot aggregator needs to calculate the reference consumption profile in a future horizon (one day in this thesis) and to calculate the maximum power reduction that can be offered as flexibility service for all the periods of the day.

Chapter 3
Description of the Model

In this chapter, the detailed description of the proposed model is discussed. The main objective function and the constraints for all the models chosen is provided in detail. The input data for each model are also depicted.

The models described in this section is aimed at defining the day-ahead scheduling of the global charging of EVs batteries connected to the charging stations in order to minimize the cost of energy procurement. The considered site is connected to the external utility grid and includes a PV unit and local loads.

The linear programming model presented in [11] is characterized by the introduction of specific operating rules relevant to the initial energy available in the EVs entering the parking lot. In the following, the two-stage stochastic model proposed in [11] will be extended into a multistage stochastic programming model, following the approach of [15],[16].

List of the models:

- Model A: Model with single next-day scenario
- Model B: Model to calculate a common profile for all the scenarios
- Model C: Model to calculate the maximum load reduction for the flexibility service
- Model D: Model for the provision of the flexibility service with an additional battery

3.1 Formulation of model A: model with single scenario

The stochastic optimization problem considered by the dispatching centre is represented by (3.1) with an optimization horizon corresponding to the next day (divided into 1-hour periods):

$$\sum_t \rho_{\mathrm{TOU}}^t E_{\mathrm{GRID}}^t \tag{3.1}$$

parameter ρ_{TOU}^t corresponds to the Time-of-Use (TOU) tariff for purchasing energy from the grid in period t in $[0,T]$ (we consider T=24, i.e., one day divided in 1h periods). Nonnegative variable E_{GRID}^t corresponds to the energy bought from the utility grid.

The constraints that represent the behaviour of the V2G parking lot are:

$$E_{\mathrm{S}}^t - E_{\mathrm{S}}^{t-1} - E_{\mathrm{No\_Loss}}^t - E_{\mathrm{S+}}^t + E_{\mathrm{S-}}^t = 0 \tag{3.2}$$

$$E_{\mathrm{Snet}}^t - E_{\mathrm{Snet}}^{t-1} - E_{\mathrm{No\_Loss}}^t - \mu^t E_{\mathrm{S+}}^t + E_{\mathrm{S-}}^t - \sum_{j=0}^t (1-\mu^t) E_{\mathrm{ini}}^{j,t} = 0 \tag{3.3}$$

$$\mu^t - \mu^t_{\max} \leq 0 \tag{3.4}$$

$$E^t_{\mathrm{GRID}} - P_{\mathrm{maxEV}}\, N^{t-1}_{\mathrm{park}}\, \Delta t \leq 0 \tag{3.5}$$

$$E^t_{\mathrm{S}} - E^{Rated}_{EV} N^t_{\mathrm{park}} \leq 0 \tag{3.6}$$

$$P^t_{\mathrm{park}}\, \Delta t - \; E^t_{\mathrm{GRID}} = 0 \tag{3.7}$$

$$E^t_{\mathrm{GRID}}\, \eta - E^t_{\mathrm{No\_Loss}} \; = 0 \tag{3.8}$$

$$P^t_{\mathrm{park}} \; - \; P_{\mathrm{p_{max}}} \leq 0 \tag{3.9}$$

Constraint (3.2) defines the energy E^t_{S} stored by the EV's batteries at the end of period *t*. Nonnegative parameter $E^t_{\mathrm{S+}}$ is the initial energy inside the cars that enter in the parking lot at time *t*. Nonnegative parameter $E^t_{\mathrm{S-}}$ is the energy inside the cars that leave the parking lot at time *t*. $E^t_{\mathrm{No\_Loss}}$ is the net energy injected in the battery from the grid, considering the reduction due to the losses in the charging station and the battery). The model assumes that all the cars enter or leave only at the end of the period.

Constraint (3.3) limits the use of the energy initially stored in the cars that enter in the parking lot by using variable $\mu^t \in [0,1]$. $E^{j,t}_{\mathrm{ini}}$ is the initial energy of the cars that enter in the parking lot at time *j* and leave at time *t*.

Constraint (3.4) limits the maximum value of μ^t to be below the input data the maximum value of $\mu^t_{\mathrm{maxInput}}$. To avoid the discharge of the EV batteries below a minimum value, the initial charge of the cars that enter at time *t* (N^t_{in}) may be used only for the amount exceeding a predefined minimum fraction ($e_{\min}$) of the rated energy size $E^{\mathrm{Rated}}_{\mathrm{EV}}$:

$$\mu^t_{\max} = \min\,(\mu^t_{\mathrm{maxInput}}, \Delta E^t_{\mathrm{S+}}) \text{ if } \Delta E^t_{\mathrm{S+}} \geq 0 \text{ othewise } \mu^t_{\max} = 0 \tag{3.10}$$

where $\Delta E^t_{\mathrm{S+}} = E^t_{\mathrm{S+}} - N^t_{\mathrm{in}}\, E^{Rated}_{EV}\, e_{\min}$

The model assumes that all the cars have the same $E^{\mathrm{Rated}}_{\mathrm{EV}}$ value.

Constraints (3.5) limits the maximum energy from the grid considering the number of charging stations occupied N^{t-1}_{park} at time *t*-1, the maximum power of each charging station P_{maxEV} , and the duration of each period Δt.

Constraints (3.6) limits the maximum level of stored energy in the parked cars at the end of period *t* considering the number of parked cars and the rated energy size.

Constraints (3.7) links the energy injected in the cars from the grid and the average power P^t_{park} consumed by the parking lot during period *t*.

Constraint (3.8) introduces the charging process efficiency η in the link between E^t_{GRID} and $E^t_{\mathrm{No\_Loss}}$.

Constraint (3.9) limits the maximum power P_{pmax} of the connection between the parking lot and the distribution network.

This scheme permits the implementation of the battery-to-battery charging strategy in a V2G system.

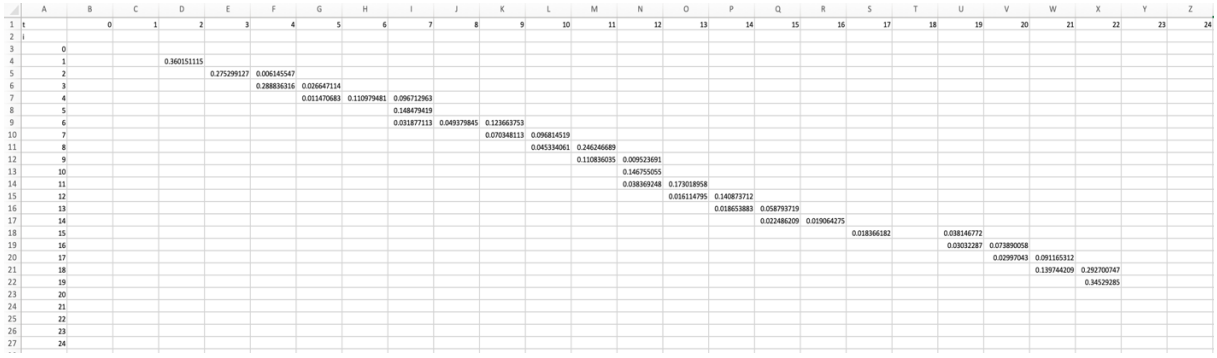
Input data of the model have been taken from different scenarios as described in the next section.

3.1.1 Input data of model A

The input data for model A are the different parameters used to describe the model. They are based on the forecast of the number of cars that enter the parking lot at time *t* N_{in}^{t} and the number of cars that leave at time *t*. From these forecasts, the number of cars parked at the end of each period *t* N_{park}^{t} is obtained. The initial energy inside the cars that enter in the parking lot $E_{\mathrm{S+}}^{t}$ is estimated assuming for each car a positive random value obtained with a truncated normal distribution with mean value of 0.3 the battery capacity. The energy that leaves the parking lot with the cars $E_{\mathrm{S-}}^{t}$ is evaluated assuming that all the cars leave the parking lot fully charged. The parameters values for 24 hours represent, fed into excel file, include also the tariff for purchasing energy from the grid at time t ρ_{TOU}^{t}, and 24×24 matrix with elements $E_{\mathrm{ini}}^{j,t}$, each representing the initial energy of the cars that enter in the parking lot at time *j* and leave at time *t*. The figures below show the values chosen for model A for one scenario.

	A	B	C	D	E	F
1	t	priceEn	ESpos	ESneg	Nparch	Nin
2	0					
3	1	51.62	0.36015112		19	19
4	2	51.62	0.28144467	0.76	21	21
5	3	51.62	0.31548343	0.76	20	18
6	4	51.62	0.21916313	0.72	20	18
7	5	51.62	0.14847942	0.12	25	8
8	6	51.62	0.20492071	0.28	32	14
9	7	72.39	0.16716263	0.8	27	15
10	8	72.39	0.29158075	0.16	44	21
11	9	72.39	0.12035973	0.64	37	9
12	10	72.39	0.14675506	0.48	38	13
13	11	72.39	0.21138821	0.96	30	16
14	12	72.39	0.15698851	0.68	22	9
15	13	72.39	0.0774476	0.56	12	4
16	14	72.39	0.04155048	0.36	5	2
17	15	72.39	0.05651295	0.16	4	3
18	16	72.39	0.10421293	0.04	9	6
19	17	72.39	0.12113574	0.04	16	8
20	18	72.39	0.43244496		42	26
21	19	72.39	0.34529285	0.12	59	20
22	20	72.39		0.24	53	
23	21	72.39		0.76	34	
24	22	72.39		1.36		
25	23	51.62				
26	24	51.62				
27						

Figure3.1- Input values of model A



t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0																									
1	0.360151115																								
2	0.275299127	0.006145547																							
3		0.288836316	0.026647114																						
4			0.011470683	0.110979481	0.096712963																				
5					0.148479419																				
6					0.031877113	0.049379845	0.123663753																		
7							0.070348113	0.096814519																	
8								0.045334061	0.246246689																
9									0.110836035	0.009523691															
10										0.146755055															
11										0.038369248	0.173018958														
12											0.016114795	0.140873712													
13												0.018653883	0.058793719												
14													0.022486209	0.019064275											
15															0.018366182				0.038146772						
16																			0.03032287	0.073890058					
17																				0.02997043	0.091165312				
18																					0.139744209	0.292700747			
19																						0.34529285			
20																									
21																									
22																									
23																									
24																									

Figure 3.2- $E_{ini}^{j,t}$ of model A

Here, the time has considered from *t*=1 to *t*=24. The price for purchasing the energy from grid is almost the same till 6th hour and 23rd, 24th equals to 51.62 and shows an increase from 7th hour to 22nd hour. The predefined values of time period duration Δt , rated power of the charging stations P_{maxEV}, EV battery capacity (assumed for simplicity the same for each car) E_{EV}^{Rated}, the minimum level of energy in the battery cars $e_{\min}$, and the charging/discharging efficiency η are:

$\Delta t = 1\mathrm{h}$

$P_{\mathrm{maxEV}} = 40\ \mathrm{kW}$

$E_{EV}^{Rated} = 40\ \mathrm{kWh}$

$e_{\min} = 0.2\ \mathrm{pu}$

$\eta = 0.96$

$P_{\mathrm{p\_max}} = 100 P_{\mathrm{maxEV}}$

The input data are repeated for 8 different scenarios, ω, fed into separate excel files. The parameters are different for each scenario are $E_{\mathrm{S+}}^{\omega,t}$ the initial energy inside the cars that enter in the parking lot, $E_{\mathrm{S-}}^{\omega,t}$ is the energy inside the cars that leave the parking lot, $N_{\mathrm{in}}^{\omega,t}$ the initial charge of the cars that enter at time *t* and scenario ω, the number of charging stations occupied $N_{\mathrm{park}}^{\omega,t}$ at time *t* and scenario ω, $E_{\mathrm{ini}}^{\omega,j,t}$ is the initial energy of the cars that enter in the parking lot at time *j* and leave at time *t* in scenario ω.

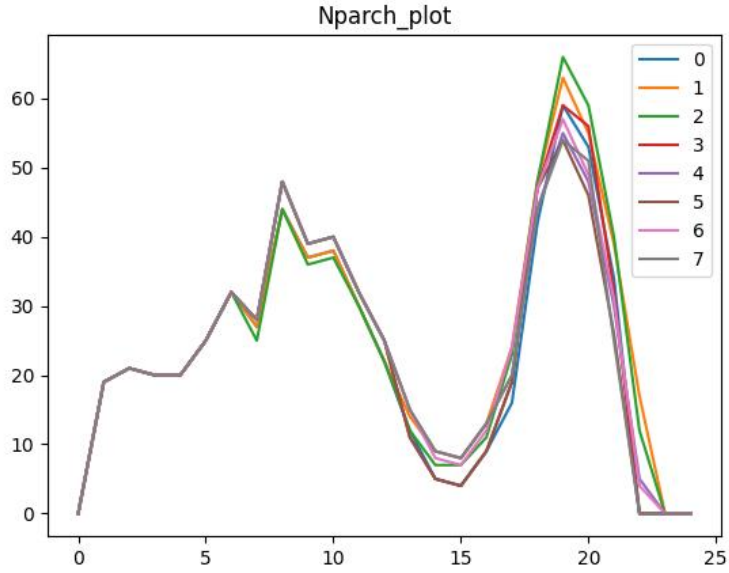


Figure3.3- Profiles of Nparch for all the 8 scenarios

The above figure shows the number of parked cars at each scenario. It shows slight variation only to each scenarios.

3.2 Formulation of Model B: common profile for all the scenarios

The aim of the model is to create a common profile P^t_{prof} that represent the reference profile that can meet the requirements of the different scenarios with the minimum mismatch for the next day consumption.

The implemented model considers the eight scenarios that we defined in model A. For this purpose, a variation has been made in the objective function and the constraints. The objective function and the constraints for the common profile creation are depicted below:

$$\sum_{t,\omega}[\rho^t_{\text{TOU}} E^{\omega,t}_{\text{GRID}} + \text{pen\_dev}(P^t_{\text{prof}} - P^{\omega,t}_{\text{park}})^2] \quad (3.11)$$

The objective function considers the summation of the daily energy procurement costs for all the scenarios (assuming the same probability for all the scenarios) and the summation of the square of differences between P^t_{prof} and $P^{\omega,t}_{\text{park}}$ in period t for each scenario ω, penalized by the predefined parameter pen_dev.

The constraints for the robust profile are (3.2)-(3.9), replacing E^t_{GRID} with $E^{\omega,t}_{\text{GRID}}$, P^t_{park} with $P^{\omega,t}_{\text{park}}$, $E^t_{\text{No\_Loss}}$ with $E^{\omega,t}_{\text{No\_Loss}}$, E^t_{Snet} with $E^{\omega,t}_{\text{Snet}}$, E^t_{S} with $E^{\omega,t}_{\text{S}}$, and

$$P^t_{\text{prof}} - P^{\omega,t}_{\text{park}} \leq \text{tol\_prof} \quad (3.12)$$

$$P_{\text{park}}^{\omega,t} - P_{\text{prof}}^{t} \leq tol\_prof \tag{3.13}$$

Constraints (3.12) and (3.13) constraints the maximum difference between P_{prof}^{t} and the profiles of each scenario by using the tolerance *tol_prof*.

The value of the tolerance can be reduced by adding a battery energy storage in the system. With the battery, the objective function becomes:

$$\begin{aligned}&\sum_{t,\omega}[\rho_{\text{TOU}}^{t} E_{\text{GRID}}^{\omega,t} + \text{pen\_dev}\left(P_{\text{prof}}^{t} - P_{\text{park}}^{\omega,t}\right)^2] \\ &+ \sum_{\omega}(\text{pen\_dev}(ES_{\text{battery}}^{\omega,24\text{h}} - ES\_ini_{\text{battery}}))^2\end{aligned} \tag{3.14}$$

The combined objective function also penalizes the difference between the auxiliary battery energy at the end of the next day $ES_{\text{battery}}^{\omega,24\text{h}}$ and the energy of the battery at the beginning of the next day ES_ini_{battery}. For simplicity, the model does not consider the charging and discharging power loss in the auxiliary battery.

The additional constraints of the model with the auxiliary battery are:

$$ES_{\text{battery}}^{\omega,t} - ES_{\text{battery}}^{\omega,t-1} - E_{\text{battery}}^{\omega,t} = 0 \quad \text{for} \quad t > 0 \tag{3.15}$$

$$ES_{\text{battery}}^{\omega,t} - ES\_ini_{\text{battery}} = 0 \qquad \text{if } t=0 \tag{3.16}$$

where $E_{\text{battery}}^{\omega,t}$ is the energy that is stored (positive) or discharged (negative) by the battery.

The energy storage in the battery is limited between a minimum and maximum level. The maximum corresponds to the battery size.

Moreover, (3.5) is replaced by

$$E_{\text{GRID}}^{\omega,t} - \left(\left(P_{\text{maxEV}}\, N_{\text{park}}^{\omega,t-1}\right) + P\text{max}_{\text{battery}}\right) \Delta t \leq 0 \tag{3.17}$$

where $P\text{max}_{\text{battery}}$ is the maximum power output of the battery.

Constraint (3.8) is replaced by

$$E_{\text{GRID}}^{\omega,t} - 1/\eta E_{\text{No\_Loss}}^{\omega,t} - E_{\text{battery}}^{\omega,t} = 0 \tag{3.18}$$

Constraint (3.9) is replaced by

$$P_{\text{park}}^{\omega,t} - P_{\text{p}_{\text{max}}} - \frac{1}{\Delta t} E_{\text{battery}}^{\omega,t} \leq 0 \tag{3.19}$$

3.2.1 Input data of model B

For the penalization pen_dev, the rather high value of 200 has been arbitrarily chosen. If the tolerance is too small, the model may be infeasible. Therefore, the calculation has been repeated for several values of *tol_prof*, with a final selection of *tol_prof* equal to the rated value of a single charging station (40 kW).

For the case with the auxiliary battery, the data of the battery size, maximum power of charging and discharging of the battery are

battery_size=1 MWh

$\text{Pmax}_{\text{battery}}$ =1MW

The model assumes that the auxiliary battery is fully charged at time equal to zero.

$ES_{ini_{\text{battery}}}$ =battery_size

With the presence of the battery the tolerance can be avoided, i.e.,

tol_prof=0 MW

3.3 Formulation of Model C: maximum reduction for the flexibility service

The objective of this model is to calculate the maximum reduction of the consumption of energy from the grid at each period with respect to the profile calculated by Model B, i.e., $P^{t}_{\text{prof\_ref}}$, considered as the reference profile for the next day. This reduction is offered to the distribution system operator as flexibility service. The optimization is repeated for each period t_{flex} when the reduction can be requested to the parking lot.

The objective function for maximum reduction is described below:

$$\sum_{t,\omega}[\rho^{t}_{\text{TOU}}\, E^{\omega,t}_{\text{GRID}} - \text{reward}_{\text{flex}}\, \Delta_{\text{flex}}] \tag{3.20}$$

The objective function considers the summation of the daily energy procurement costs for all the scenarios and difference reduction term. The reduction terms is given by the product between the predefined reward and the maximum reduction describe by positive variable Δ_{flex}. These two terms play a significance role in reducing the overall objective function at every period.

The constraints for maximum reduction are (3.2),(3.3),(3.4),(3.5),(3.6),(3.7),(3.8),(3.9), replacing E^{t}_{GRID} with $E^{\omega,t}_{\text{GRID}}$, P^{t}_{park} with $P^{\omega,t}_{\text{park}}$, $E^{t}_{\text{No\_Loss}}$ with $E^{\omega,t}_{\text{No\_Loss}}$, E^{t}_{Snet} with $E^{\omega,t}_{\text{Snet}}$, E^{t}_{S} with $E^{\omega,t}_{\text{S}}$, μ^{t} with $\mu^{\omega,t}$, μ^{t}_{max} with $\mu^{\omega,t}_{\text{max}}$.

The model assumes that the reduction can be requested in a single period t_{flex} during the day. Moreover, the parking lot can recover the energy in a period after t_{flex} with a duration *flex_recover*. The profile that considers both the reduction and the recovery is $P^{t}_{\text{prof\_flex}}$. Constraint (3.22) is used when either one of the conditions $t < t_{\text{flex}}$ or $t < t_{\text{flex}} + flex\_recover$ is satisfied (i.e., before t_{flex} and after the recovery period).

$$P^{t}_{\text{prof\_flex}} - P^{t}_{\text{prof\_ref}} = 0 \tag{3.21}$$

If $t = t_{\text{flex}}$

$$\Delta_{\text{flex}} + P^{t}_{\text{prof\_flex}} - P^{t}_{\text{prof\_ref}} = 0 \quad (3.22)$$

During the recovery interval (i.e., $t > t_{\text{flex}}$ and $t \leq t_{\text{flex}} + flex\_recover$) constraint (3.23) limits the recovery according to the maximum reduction Δ_{flex} and the duration of the recovery interval.

$$(\Delta_{\text{flex}}/flex\_recover) - P^{t}_{\text{prof\_flex}} + P^{t}_{\text{prof\_ref}} \geq 0 \quad (3.23)$$

This model allows to get a maximum reduction on the energy procurement cost and the input to this model C is listed below.

3.3.1 Input data of model C

To get maximum reduction, two terms have been introduced: *flex_recover* and *reward_flex*. The input values given to these two terms are 3 and 200 respectively. Whenever there occurs a reduction, then within the next 3 period itself it's got compensated.

3.4 Formulation of Model D: flexibility service with additional battery

Model D calculates the maximum reduction Δ_{flex} considering the presence of the auxiliary battery already described in model B to reduce the tolerance limit and to compensate the uncertainties relevant to the parking lot demand. The objective function of this model is:

$$\sum_{t,\omega}[\rho^{t}_{\text{TOU}}\, E^{\omega,t}_{\text{GRID}} - \text{reward}_{\text{flex}}\, \Delta_{\text{flex}}] + \sum_{\omega}(\text{pen\_dev}(ES^{\omega,24\text{h}}_{\text{battery}} - ES_{ini_{\text{battery}}}))^2 \quad (3.24)$$

The combined objective function includes the penalization of the difference between the auxiliary battery energy at the beginning and at the end of the day other than the reward for the provision of the flexibility service.

The constraints for this model are the same as discussed in the model C as it considers the maximum reduction without the auxiliary battery. This model considers the auxiliary battery in the objective function.

3.4.1 Input data of model D

The data of this model are those already described for model B including the additional battery and model C.

Chapter 4

Implementation of the models by using Python – Pyomo

This chapter describes how the optimization models has implemented in Pyomo. A description about modelling, its features and detailed report of the coding used has clearly depicted.

4.1 Mathematical Modeling

Modeling is the elementary process in many aspects of scientific research, engineering, and business fields. It involves the formulation of a simplified representation of a real-world object or real system. Models are a means of understanding the problems involved in building something; an aid to communicate between those involved in the project, specially between the requirement analyst and the user; a component of the methods used in development activities such as the analysis and the design of artefact. It is an abstraction, which allows people to concentrate on the essentials of a problem by keeping out non-essential details.

Mathematics has an important role in representing and formulating our knowledge. Mathematical modelling has become formally as a new framework to express complex systems. Mathematical modelling is the process of describing a real-world problem in mathematical terms, usually in the form of equations, and then using these equations both to help understand the problem, and to discover new features about the problem. Modelling lies at the heart of much of our understanding of the world, and it allows engineers to design the technology of the future. The following mathematical concepts are central to modern modeling activities:

- Variables: these represent the unknown or changing parts of a model (e.g., whether to or not to take a decision, or the characteristic of a system outcome).
- Parameters: these are symbolic representations for real-world data and might vary for different problem instances or scenarios.
- Relations: these are equations, inequalities or other mathematical relationships defining how different parts of a model are related to each other.

4.2 Optimization Models

Optimization models are mathematical models with functions representing goals or objectives for the system being modelled. Optimization is the central to any problem involving decision making, whether in engineering or in economics or in management. The task of decision-making entails choosing between various alternatives. This choice is governed by our desire to make the 'best' decision. The measure of goodness of alternatives is described by an objective function or performance index. Optimization theory and methods deal with selecting the best alternative from a given objective function.

The end goal of all such decisions is either to minimize the effort required or maximize the desired benefit. One possible definition of optimization model is "Mathematical models designed to help institutions and individuals decide how to allocate scarce resources, to activities and to make best use of their circumstances".

4.2.1 Features of Optimization Models

An optimization model has three main components:

- An objective function: this is the function that needs to be optimized.
- A collection of decision variables: the solution to the optimization problem is the set of values of the decision variables for which the objective function reaches its optimal value.
- A collection of constraints: these are the constraint that restrict the values of decision variables.

4.3 PYOMO – Optimization Modelling Tool

To implement the above said optimization problem in chapter 3, I choose Pyomo (Python Optimization Modelling object), uses python scripting. Pyomo is an open-source Python-based modelling language and optimization tool for developing and solving large-scale mathematical models. It provides a flexible and intuitive way to formulate and solve a wide range of optimization problems, including linear programming, integer programming, nonlinear programming, and mixed-integer programming.

Pyomo allows users to define optimization problems using an algebraic syntax that is similar to the mathematical equations used to describe the problem. This makes it easy to

formulate complex optimization problems and to modify the model as needed. Pyomo also supports a wide range of solvers, including open-source solvers like GLPK and CBC, and commercial solvers like Gurobi and CPLEX. This allows users to choose the solver that best fits their needs and to easily switch between solvers if needed.

Key features of Pyomo include:

- Support for a wide range of optimization problem types, including linear programming, integer programming, nonlinear programming, and mixed-integer programming.
- A flexible and intuitive algebraic syntax for defining optimization models.
- Support for a wide range of solvers, including open-source and commercial solvers.
- Integration with Python, allowing users to easily incorporate other Python libraries and tools into their optimization models.
- Support for parallel computing, allowing users to solve large-scale optimization problems on multiple processors or nodes.

Overall, Pyomo is a powerful and flexible optimization tool that can be used to develop and solve a wide range of optimization problems.

4.3.1 Overview of Modelling in PYOMO

The basic steps of a modelling process are:

- create model and declare components;
- instantiate the model;
- apply solver;
- interrogate solver results.

A Pyomo model consists of a collection of modelling components that define various aspects of model. It contains modelling components that are commonly used by modern AMLs: index sets, symbolic parameters, decision variables, objectives, and constraints. These modeling components are defined in Pyomo through the following python classes: Set, Param, Var, Objective, and Constraint. Set and Param are the set data and parameter data that is used to define a model instance. Var is the decision variables in a model and the expressions that are minimized or maximized in a model is its objective. The constraints are the expressions that impose restrictions on the variable values in a model.

4.3.2 Abstract and Concrete Models

If the mathematical model can be defined using the following equations,

$$\min \sum_{j=1}^{n} c_j x_j$$

$$\text{s.t } \sum_{j=1}^{n} a_{ij} x_j \geq b_i \quad \forall i = 1 \ldots m$$

$$x_j \geq 0 \qquad \forall j = 1 \ldots m$$

This type of models is called abstract or symbolic mathematical modelling since it relies unspecified parameter values. Data values can be used to specify model instance. The *AbstractModel* class gives a context for defining and initializing abstract optimization models in Pyomo when the data values will be supplied at the time a solution is obtained. On the other hand, a mathematical model can be directly defined with the data values supplied at the time of the model definition. These are called concrete models. For example,

$$\min 2x_1 + 3x_2$$

$$\text{s.t } 3x_1 + 4x_2 \geq 1$$

$$x_1, x_2 \geq 0$$

The *ConcreteModel* class is used to define concrete optimization models in Pyomo.

4.3.3 Implemetation of Pyomo in Defined Models

To work in Pyomo, Thonny has been used as a free Python IDE (Integrated Development Environment). The primary step of modeling in Pyomo is to import the Pyomo environment by using the command *'from pyomo.environ import*'*. This is the main step as the purpose is to make the symbols used by the Pyomo known to python. Then we need to import the required libraries needed for specific functions to operate. The libraries used here are pandas, numpy and matplotlib. The pandas are used for data analysis while numpy is used for working with arrays. Matplotlib is a data visualization and graphical plotting library where we can obtain the results graphically, which is easy to compare. After importing the libraries, import the solver factory to create the solver and then read the excel file which contains the case-study data by the function *pd.read_excel()*. Here pd stands for pandas. In order to replace the missing values with a specified value, use fillna() function. It allows us to specify a particular value to replace Nan's, by default it takes None. The *dict()* function creates a dictionary which are used to store values that can be unordered, changeable and indexed. All the parameters used in this model must be implemented by using dictionary or numpy. NumPy

is a popular Python library used for scientific computing, including numerical operations on arrays and matrices. NumPy can be used in Pyomo to perform mathematical operations on arrays and matrices that are used in optimization models. Pyomo has a built-in NumPy interface that allows users to create NumPy arrays and matrices directly within Pyomo and use them in optimization models.

To use NumPy in Pyomo, first there is the need to import the NumPy library in the Python script. By adding the following line of code at the beginning of the script: *import numpy as np*. Openpyxl is a Python library that can be used to read and write Microsoft Excel files. In Pyomo, Openpyxl can be used to read data from Excel files and use it to initialize Pyomo parameters, variables, and constraints.

```
import pandas as pd
from pyomo.environ import *
from pyomo.opt import SolverFactory
import numpy as np
import matplotlib.pyplot as plt
import openpyxl
```

4.3.4 Implementation of Pyomo in Model-A

a) Extraction of the data from excel file

Gurobi is the solver. The link to the solver is created by using the command *opt = solverfactory('gurobi')*. Then defined the model-Abstract model and defined the sets needed. The set here is the timeslot starting 0 and ends 24. The parameters and variables that are required to define the optimization model are introduced by using *param()* and *var()* functions respectively. To read a single scenario saved in excel file, the model uses *pd.read_excel*(*the file path*). In Pyomo, a dictionary is a Python data structure used to store key-value pairs. Dictionaries are often used in Pyomo to represent parameter data or to index variables.

```
df = pd.read_excel(r'data4.xlsx',sheet_name='parking',header=0)
df=df.fillna(0)
dictionary=df.to_dict()
priceEn=dictionary['priceEn']
ESpos=dictionary['ESpos']
ESneg=dictionary['ESneg']
Nparch=dictionary['Nparch']
Nin=dictionary['Nin']
gr = pd.read_excel('data4.xlsx', sheet_name='Eini', header=None, index_col=0)
gr=gr.fillna(0)
gr_array=gr.to_numpy()
gr_array=np.delete(gr_array,[0,1],0)
Eini=dict(((i,j), gr_array[i][j]) for i in range(len(gr_array)) for j in range(len(gr_array[0])))
```

To read other scenarios, the model collected all the excel data files under a name and read that name for initiating all files and obtain the objective value for all as it did for a single scenario.

```
name_case=['data5.xlsx','data6.xlsx','data7.xlsx','data8.xlsx','data9.xlsx','data10.xlsx','data11.xlsx']
for name in name_case:
    df = pd.read_excel(name,sheet_name='parking',header=0)
    df=df.fillna(0)
    dictionary=df.to_dict()
    priceEn=dictionary['priceEn']
    ESpos=dictionary['ESpos']
    ESneg=dictionary['ESneg']
    Nparch=dictionary['Nparch']
    Nin=dictionary['Nin']
    gr = pd.read_excel(name, sheet_name='Eini', header=None, index_col=0)
    gr=gr.fillna(0)
    gr_array=gr.to_numpy()
    gr_array=np.delete(gr_array,[0,1],0)
    Eini=dict(((i,j), gr_array[i][j]) for i in range(len(gr_array)) for j in range(len(gr_array[0])))
```

b) Definition of the parameters

In Pyomo, a parameter is a fixed quantity used in the mathematical formulation of an optimization problem. It is a constant value that is set before solving the optimization problem, and its value does not change during the optimization process. Once a parameter is defined, it can be used in the mathematical formulation of the optimization problem by referencing its name in the objective function, constraints, or other model components.

```
m.priceEn = Param(m.set_timeslots, initialize=priceEn)
m.ESpos = Param(m.set_timeslots, initialize=ESpos)
m.ESneg = Param(m.set_timeslots, initialize=ESneg)
m.Nparch = Param(m.set_timeslots, initialize=Nparch)
m.Eini = Param(m.set_timeslots,m.set_timeslots, initialize=Eini)
m.muMax = Param(m.set_timeslots, initialize=mu_max_dict)
```

c) Definition of the variables

In Pyomo, a variable is an unknown quantity that needs to be determined as part of an optimization problem. Variables are used to represent decision variables in mathematical models and their values are determined by the solver during the optimization process. To create a variable in Pyomo, its name, domain, and bounds are needed. The domain specifies the set of allowable values that the variable can take on, while the bounds define the range of values that the variable can have.

```
m.Egrid = Var(m.set_timeslots, domain=NonNegativeReals)
m.Pparking = Var(m.set_timeslots, domain=NonNegativeReals)
m.ES = Var(m.set_timeslots, domain=NonNegativeReals)
m.mu = Var(m.set_timeslots, domain=NonNegativeReals)
m.ESnet = Var(m.set_timeslots, domain=NonNegativeReals)
```

d) Objective function

In Pyomo, the objective function is the function that needs to be optimized as part of the optimization problem. The objective function is a mathematical expression that depends on the decision variables and reflects the optimization goals. The goal is typically to maximize or minimize the objective function subject to constraints.

The objective function is defined using the Objective component in Pyomo. The Objective component requires two arguments: a name for the objective function, and the expression that defines the objective function.

```
def obj_expression(m):
    return summation(m.priceEn, m.Egrid)
m.OBJ = Objective(rule=obj_expression)
```

e) Constraints

In Pyomo, constraints are used to specify the relationships between the decision variables and model parameters. Constraints restrict the feasible region of the optimization problem by specifying allowable values for the decision variables. The constraints can be linear or nonlinear equations or inequalities and can involve any combination of decision variables and parameters.

Constraints are defined using the Constraint component in Pyomo. The Constraint component requires three arguments: a name for the constraint, an index set, and the expression that defines the constraint.

```
#constraint (enegy balance)
def en_balance(m, t):
    expr2 = 0 # initialize summing variable
    expr2 +=m.ES[t]-m.Egrid[t]-m.ESpos[t]+m.ESneg[t]
    if t>0:
        expr2 +=-m.ES[t-1]
    # return the expression for the constraint for i
    return expr2==0
m.en_balance = Constraint(m.set_timeslots, rule=en_balance)

#constraint (net enegy balance)
def net_en_balance(m, t):
    expr2 = m.ESnet[t]-m.Egrid[t]-m.mu[t]*m.ESpos[t]+m.ESneg[t]-sum((1-m.mu[t])*m.Eini[j,t] for j in range(t))
    if t>0:
        expr2 +=-m.ESnet[t-1]
    # return the expression for the constraint for i
    return expr2==0
m.net_en_balance = Constraint(m.set_timeslots, rule=net_en_balance)
```

```
#constraint (max Egrid)
def Ppk_max(m, t):
    if t>0:
        return m.Pparking[t]-Pmax_EV*m.Nparch[t-1]<=0
    return m.Egrid[t]<=0
m.Ppk_max = Constraint(m.set_timeslots, rule=Ppk_max)


#constraint (max ES assuming that ESnet<ES)
def ES_max(m, t):
    if t>0:
        return m.ES[t]-Cnom_EV*m.Nparch[t]<=0
    return m.ES[t]<=0
m.ES_max = Constraint(m.set_timeslots, rule=ES_max)

#constraint (max mu)
def mu_max(m, t):
    return m.mu[t]-m.muMax[t]<=0
m.mu_max = Constraint(m.set_timeslots, rule=mu_max)
```

```
#constraint (battery power loss)
def P_pk(m, t):
    return m.Pparking[t]*deltaT*eta-m.Egrid[t]==0
m.P_pk = Constraint(m.set_timeslots, rule=P_pk)
```

4.3.5 Implementation of Pyomo in Model-B

a) Extraction of the data from excel file

The extraction of data from excel file is as the same explained in the model A. All the data are extracted under one name. the functions Numpy provides is np.zeros(), which creates an array of a specified shape and initializes all the elements to zero. To repeat the reading from excel to all the scenarios, for loop is used. The for loop is used to iterative over a sequence of elements in the list.

```
name_case=['data4.xlsx','data5.xlsx','data6.xlsx','data7.xlsx','data8.xlsx','data9.xlsx','data10.xlsx','data11.xlsx']
df = pd.read_excel(name_case[0],sheet_name='parking',header=0)
df=df.fillna(0)
dictionary=df.to_dict()
priceEn=dictionary['priceEn']
n_periods=len(priceEn)
n_scenarios=len(name_case)
ESpos_array=np.zeros((n_periods,n_scenarios))
ESneg_array=np.zeros((n_periods,n_scenarios))
Nparch_array=np.zeros((n_periods,n_scenarios))
Nin_array=np.zeros((n_periods,n_scenarios))
ESpos_array[:,0]=df['ESpos']
ESneg_array[:,0]=df['ESneg']
Nparch_array[:,0]=df['Nparch']
Nin_array[:,0]=df['Nin']
```

```
count=0
for name in name_case:
    if name!=name_case[0]:
        count+=1
        df = pd.read_excel(name,sheet_name='parking',header=0)
        df=df.fillna(0)
        ESpos_array[:,count]=df['ESpos']
        ESneg_array[:,count]=df['ESneg']
        Nparch_array[:,count]=df['Nparch']
        Nin_array[:,count]=df['Nin']
ESpos=dict(((i,j), ESpos_array[i][j]) for i in range(n_periods) for j in range(n_scenarios))
ESneg=dict(((i,j), ESneg_array[i][j]) for i in range(n_periods) for j in range(n_scenarios))
Nparch=dict(((i,j), Nparch_array[i][j]) for i in range(n_periods) for j in range(n_scenarios))
Nin=dict(((i,j), Nin_array[i][j]) for i in range(n_periods) for j in range(n_scenarios))
```

```
gr = pd.read_excel(name_case[0], sheet_name='Eini', header=None, index_col=0)
gr=gr.fillna(0)
gr_array=gr.to_numpy()
gr_array=np.delete(gr_array,[0,1],0)
Eini_array=np.zeros((n_periods,n_periods,n_scenarios))
Eini_array[:,:,0]=gr_array

count=0
for name in name_case:
    if name!=name_case[0]:
        count+=1
        gr = pd.read_excel(name, sheet_name='Eini', header=None, index_col=0)
        gr=gr.fillna(0)
        gr_array=gr.to_numpy()
        gr_array=np.delete(gr_array,[0,1],0)
        Eini_array[:,:,count]=gr_array

Eini=dict(((i,j,s), Eini_array[i][j][s]) for i in range(n_periods) for j in range(n_periods) for s in range(n_scenarios))
```

b) Definition of the parameters

The description of parameters is the same as explained for model A. In addition to that, mutable=True is included as it is an argument that can be passed to certain functions in Python to specify that the object being passed as an argument is mutable, which means that it can be changed or modified.

```
# parameters
m.priceEn = Param(m.set_timeslots, initialize=priceEn,mutable=True)
m.ESpos = Param(m.set_timeslots, m.set_scenarios, initialize=ESpos,mutable=True)
m.ESneg = Param(m.set_timeslots, m.set_scenarios, initialize=ESneg,mutable=True)
m.Nparch = Param(m.set_timeslots, m.set_scenarios, initialize=Nparch,mutable=True)
m.Eini = Param(m.set_timeslots,m.set_timeslots, m.set_scenarios, initialize=Eini,mutable=True)
m.muMax = Param(m.set_timeslots,m.set_scenarios, initialize=mu_max_dict,mutable=True)
```

c) Definition of the variables

The variables for this model are defined as depicted below. Apart from model A, two more variables added.

```
# variables
m.Egrid = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals) # wtih losses
m.ENoLoss = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals) # without losses
m.Pparking = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.Pprofile = Var(m.set_timeslots, domain=NonNegativeReals)
m.ES = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.mu = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.ESnet = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
```

d) Objective function

The objective function for this model is shown below. It has a penalization term as discussed before in the description of models.

```
# objective function
def obj_expression(m):
    return sum(m.priceEn[t]*m.Egrid[t,s]+pen_dev*(m.Pprofile[t]-m.Pparking[t,s])**2 for t in m.set_timeslots for s in m.set_scenarios)
m.OBJ = Objective(rule=obj_expression)
```

e) Constraints

The constraints of the model C are listed below.

```
#constraint (enegy balance)
def en_balance(m, t, s):
    expr2 = 0 # initialize summing variable
    expr2 +=m.ES[t,s]-m.ENoLoss[t,s]-m.ESpos[t,s]+m.ESneg[t,s]
    if t>0:
        expr2 +=-m.ES[t-1,s]
    # return the expression for the constraint for i
    return expr2==0
m.en_balance = Constraint(m.set_timeslots, m.set_scenarios, rule=en_balance)

#constraint (net enegy balance)
def net_en_balance(m, t, s):
    expr2 = m.ESnet[t,s]-m.ENoLoss[t,s]-m.mu[t,s]*m.ESpos[t,s]+m.ESneg[t,s]-sum((1-m.mu[t,s])*m.Eini[j,t,s] for j in range(t))
    if t>0:
        expr2 +=-m.ESnet[t-1,s]
    # return the expression for the constraint for i
    return expr2==0
m.net_en_balance = Constraint(m.set_timeslots, m.set_scenarios, rule=net_en_balance)
```

```
#constraint (max Egrid)
def Egrid_max(m, t, s):
    if t>0:
        return m.Egrid[t,s]-Pmax_EV*m.Nparch[t-1,s]*deltaT<=0
    return m.Egrid[t,s]<=0
m.Egrid_max = Constraint(m.set_timeslots, m.set_scenarios, rule=Egrid_max)


#constraint (max ES assuming that ESnet<ES)
def ES_max(m, t,s):
    if t>0:
        return m.ES[t,s]-Cnom_EV*m.Nparch[t,s]<=0
    return m.ES[t,s]<=0
m.ES_max = Constraint(m.set_timeslots, m.set_scenarios, rule=ES_max)

#constraint (max mu)
def mu_max(m, t, s):
    return m.mu[t,s]-m.muMax[t,s]<=0
m.mu_max = Constraint(m.set_timeslots, m.set_scenarios, rule=mu_max)
```

```
#constraint (battery power loss)
def P_pk(m, t, s):
    return m.Egrid[t,s]*eta-m.ENoLoss[t,s]==0
m.P_pk = Constraint(m.set_timeslots, m.set_scenarios, rule=P_pk)

#constraint (Pparking)
def Pp(m, t, s):
    return m.Pparking[t,s]*deltaT-m.Egrid[t,s]==0
m.Pp = Constraint(m.set_timeslots, m.set_scenarios, rule=Pp)

#constraint (max Pparking)
def Pparking_max(m, t, s):
    if t>0:
        return m.Pparking[t,s]-Pp_max<=0
    return m.Pparking[t,s]<=0
m.Pparking_max = Constraint(m.set_timeslots, m.set_scenarios, rule=Pparking_max)
```

```
#constraint (difference_profiles)
def difference_profiles_up(m, t, s):
    return m.Pprofile[t]-m.Pparking[t,s]<=tol_profile
m.difference_profiles_up = Constraint(m.set_timeslots, m.set_scenarios, rule=difference_profiles_up)

def difference_profiles_down(m, t, s):
    return m.Pparking[t,s]-m.Pprofile[t]<=tol_profile
m.difference_profiles_down = Constraint(m.set_timeslots, m.set_scenarios, rule=difference_profiles_down)
```

4.3.6 Implementation of Pyomo in Model-C

a) Extraction of the data from the excel file

```
name_case=['data4.xlsx','data5.xlsx','data6.xlsx','data7.xlsx','data8.xlsx','data9.xlsx','data10.xlsx','data11.xlsx']
df = pd.read_excel(name_case[0],sheet_name='parking',header=0)
df=df.fillna(0)
dictionary=df.to_dict()
priceEn=dictionary['priceEn']
n_periods=len(priceEn)
n_scenarios=len(name_case)
ESpos_array=np.zeros((n_periods,n_scenarios))
ESneg_array=np.zeros((n_periods,n_scenarios))
Nparch_array=np.zeros((n_periods,n_scenarios))
Nin_array=np.zeros((n_periods,n_scenarios))
ESpos_array[:,0]=df['ESpos']
ESneg_array[:,0]=df['ESneg']
Nparch_array[:,0]=df['Nparch']
Nin_array[:,0]=df['Nin']
```

```
count=0
for name in name_case:
    if name!=name_case[0]:
        count+=1
        df = pd.read_excel(name,sheet_name='parking',header=0)
        df=df.fillna(0)
        ESpos_array[:,count]=df['ESpos']
        ESneg_array[:,count]=df['ESneg']
        Nparch_array[:,count]=df['Nparch']
        Nin_array[:,count]=df['Nin']
ESpos=dict(((i,j), ESpos_array[i][j]) for i in range(n_periods) for j in range(n_scenarios))
ESneg=dict(((i,j), ESneg_array[i][j]) for i in range(n_periods) for j in range(n_scenarios))
Nparch=dict(((i,j), Nparch_array[i][j]) for i in range(n_periods) for j in range(n_scenarios))
Nin=dict(((i,j), Nin_array[i][j]) for i in range(n_periods) for j in range(n_scenarios))
```

```
gr = pd.read_excel(name_case[0], sheet_name='Eini', header=None, index_col=0)
gr=gr.fillna(0)
gr_array=gr.to_numpy()
gr_array=np.delete(gr_array,[0,1],0)
Eini_array=np.zeros((n_periods,n_periods,n_scenarios))
Eini_array[:,:,0]=gr_array

count=0
for name in name_case:
    if name!=name_case[0]:
        count+=1
        gr = pd.read_excel(name, sheet_name='Eini', header=None, index_col=0)
        gr=gr.fillna(0)
        gr_array=gr.to_numpy()
        gr_array=np.delete(gr_array,[0,1],0)
        Eini_array[:,:,count]=gr_array

Eini=dict(((i,j,s), Eini_array[i][j][s]) for i in range(n_periods) for j in range(n_periods) for s in range(n_scenarios))
```

b) Definition of the parameters

```
# parameters
m.priceEn = Param(m.set_timeslots, initialize=priceEn,mutable=True)
m.ESpos = Param(m.set_timeslots, m.set_scenarios, initialize=ESpos,mutable=True)
m.ESneg = Param(m.set_timeslots, m.set_scenarios, initialize=ESneg,mutable=True)
m.Nparch = Param(m.set_timeslots, m.set_scenarios, initialize=Nparch,mutable=True)
m.Eini = Param(m.set_timeslots,m.set_timeslots, m.set_scenarios, initialize=Eini,mutable=True)
m.muMax = Param(m.set_timeslots,m.set_scenarios, initialize=mu_max_dict,mutable=True)
```

```
# parameters
m.priceEn = Param(m.set_timeslots, initialize=priceEn,mutable=True)
m.ESpos = Param(m.set_timeslots, m.set_scenarios, initialize=ESpos,mutable=True)
m.ESneg = Param(m.set_timeslots, m.set_scenarios, initialize=ESneg,mutable=True)
m.Nparch = Param(m.set_timeslots, m.set_scenarios, initialize=Nparch,mutable=True)
m.Eini = Param(m.set_timeslots,m.set_timeslots, m.set_scenarios, initialize=Eini,mutable=True)
m.muMax = Param(m.set_timeslots,m.set_scenarios, initialize=mu_max_dict,mutable=True)
m.Pprofile_ref = Param(m.set_timeslots, initialize=Pprofile_ref_dict,mutable=True)
```

c) Definition of the variables

```
# variables
m.Egrid = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals) # wtih losses
m.ENoLoss = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals) # without losses
m.Pparking = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.Pprofile = Var(m.set_timeslots, domain=NonNegativeReals)
m.ES = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.mu = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.ESnet = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
```

```
# variables
m.Egrid = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals) # wtih losses
m.ENoLoss = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals) # without losses
m.Pparking = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.Pprofile_flex = Var(m.set_timeslots, domain=NonNegativeReals)
m.ES = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.mu = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.ESnet = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.max_reduction=Var(domain=NonNegativeReals)
```

d) Objective function

```
# objective function
def obj_expression(m):
    return sum(m.priceEn[t]*m.Egrid[t,s]+pen_dev*(m.Pprofile[t]-m.Pparking[t,s])**2 for t in m.set_timeslots for s in m.set_scenarios)
m.OBJ = Objective(rule=obj_expression)
```

```
# objective function
def obj_expression(m):
    return sum(m.priceEn[t]*m.Egrid[t,s]-reward_flex*(m.max_reduction) for t in m.set_timeslots for s in m.set_scenarios)
m.OBJ = Objective(rule=obj_expression)
```

e) Constraints

```
#constraint (enegy balance)
def en_balance(m, t, s):
    expr2 = 0 # initialize summing variable
    expr2 +=m.ES[t,s]-m.ENoLoss[t,s]-m.ESpos[t,s]+m.ESneg[t,s]
    if t>0:
        expr2 +=-m.ES[t-1,s]
    # return the expression for the constraint for i
    return expr2==0
m.en_balance = Constraint(m.set_timeslots, m.set_scenarios, rule=en_balance)

#constraint (net enegy balance)
def net_en_balance(m, t, s):
    expr2 = m.ESnet[t,s]-m.ENoLoss[t,s]-m.mu[t,s]*m.ESpos[t,s]+m.ESneg[t,s]-sum((1-m.mu[t,s])*m.Eini[j,t,s] for j in range(t))
    if t>0:
        expr2 +=-m.ESnet[t-1,s]
    # return the expression for the constraint for i
    return expr2==0
m.net_en_balance = Constraint(m.set_timeslots, m.set_scenarios, rule=net_en_balance)
```

```
def Egrid_max(m, t, s):
    if t>0:
        return m.Egrid[t,s]-Pmax_EV*m.Nparch[t-1,s]*deltaT<=0
    return m.Egrid[t,s]<=0
m.Egrid_max = Constraint(m.set_timeslots, m.set_scenarios, rule=Egrid_max)


#constraint (max ES assuming that ESnet<ES)
def ES_max(m, t,s):
    if t>0:
        return m.ES[t,s]-Cnom_EV*m.Nparch[t,s]<=0
    return m.ES[t,s]<=0
m.ES_max = Constraint(m.set_timeslots, m.set_scenarios, rule=ES_max)

#constraint (max mu)
def mu_max(m, t, s):
    return m.mu[t,s]-m.muMax[t,s]<=0
m.mu_max = Constraint(m.set_timeslots, m.set_scenarios, rule=mu_max)

#constraint (battery power loss)
def P_pk(m, t, s):
    return m.Egrid[t,s]*eta-m.ENoLoss[t,s]==0
m.P_pk = Constraint(m.set_timeslots, m.set_scenarios, rule=P_pk)
```

```
#constraint (Pparking)
def Pp(m, t, s):
    return m.Pparking[t,s]*deltaT-m.Egrid[t,s]==0
m.Pp = Constraint(m.set_timeslots, m.set_scenarios, rule=Pp)

#constraint (max Pparking)
def Pparking_max(m, t, s):
    if t>0:
        return m.Pparking[t,s]-Pp_max<=0
    return m.Pparking[t,s]<=0
m.Pparking_max = Constraint(m.set_timeslots, m.set_scenarios, rule=Pparking_max)

#constraint (difference_profiles)
def difference_profiles_up(m, t, s):
    return m.Pprofile[t]-m.Pparking[t,s]<=tol_profile
m.difference_profiles_up = Constraint(m.set_timeslots, m.set_scenarios, rule=difference_profiles_up)

def difference_profiles_down(m, t, s):
    return m.Pparking[t,s]-m.Pprofile[t]<=tol_profile
m.difference_profiles_down = Constraint(m.set_timeslots, m.set_scenarios, rule=difference_profiles_down)
```

```
#constraint (difference_profiles)
def difference_profiles_up(m, t, s):
    return m.Pprofile_flex[t]-m.Pparking[t,s]<=tol_profile
m.difference_profiles_up = Constraint(m.set_timeslots, m.set_scenarios, rule=difference_profiles_up)

def difference_profiles_down(m, t, s):
    return m.Pparking[t,s]-m.Pprofile_flex[t]<=tol_profile
m.difference_profiles_down = Constraint(m.set_timeslots, m.set_scenarios, rule=difference_profiles_down)

#constraint (max reduction)
def Max_reduction(m, t):
    if t<t_flex or t>t_flex+flex_recover:
        return m.Pprofile_flex[t]-m.Pprofile_ref[t]==0
    elif t==t_flex:
        return m.max_reduction+m.Pprofile_flex[t]-m.Pprofile_ref[t]==0
    elif t>t_flex and t<=t_flex+flex_recover:
        return m.max_reduction/flex_recover-m.Pprofile_flex[t]+m.Pprofile_ref[t]>=0
    return m.Pprofile_flex[t]>=0
m.Max_reduction = Constraint(m.set_timeslots, rule=Max_reduction)
```

4.3.7 Implementation of Pyomo in Model-D

a) Extraction of the data from the excel files

The extraction of data from excel files is done as described in the model C.

```
name_case=['data4.xlsx','data5.xlsx','data6.xlsx','data7.xlsx','data8.xlsx','data9.xlsx','data10.xlsx','data11.xlsx']
df = pd.read_excel(name_case[0],sheet_name='parking',header=0)
df=df.fillna(0)
dictionary=df.to_dict()
priceEn=dictionary['priceEn']
n_periods=len(priceEn)
n_scenarios=len(name_case)
ESpos_array=np.zeros((n_periods,n_scenarios))
ESneg_array=np.zeros((n_periods,n_scenarios))
Nparch_array=np.zeros((n_periods,n_scenarios))
Nin_array=np.zeros((n_periods,n_scenarios))
ESpos_array[:,0]=df['ESpos']
ESneg_array[:,0]=df['ESneg']
Nparch_array[:,0]=df['Nparch']
Nin_array[:,0]=df['Nin']
```

b) Definition of the parameters

```
# parameters
m.priceEn = Param(m.set_timeslots, initialize=priceEn,mutable=True)
m.ESpos = Param(m.set_timeslots, m.set_scenarios, initialize=ESpos,mutable=True)
m.ESneg = Param(m.set_timeslots, m.set_scenarios, initialize=ESneg,mutable=True)
m.Nparch = Param(m.set_timeslots, m.set_scenarios, initialize=Nparch,mutable=True)
m.Eini = Param(m.set_timeslots,m.set_timeslots, m.set_scenarios, initialize=Eini,mutable=True)
m.muMax = Param(m.set_timeslots,m.set_scenarios, initialize=mu_max_dict,mutable=True)
```

```
m.priceEn = Param(m.set_timeslots, initialize=priceEn,mutable=True)
m.ESpos = Param(m.set_timeslots, m.set_scenarios, initialize=ESpos,mutable=True)
m.ESneg = Param(m.set_timeslots, m.set_scenarios, initialize=ESneg,mutable=True)
m.Nparch = Param(m.set_timeslots, m.set_scenarios, initialize=Nparch,mutable=True)
m.Eini = Param(m.set_timeslots,m.set_timeslots, m.set_scenarios, initialize=Eini,mutable=True)
m.muMax = Param(m.set_timeslots,m.set_scenarios, initialize=mu_max_dict,mutable=True)
m.Pprofile_ref = Param(m.set_timeslots, initialize=Pprofile_ref_dict,mutable=True)
```

c) Definition of the variables

```
# variables
m.Egrid = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals) # wtih losses
m.ENoLoss = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals) # without losses
m.ENoLoss_battery = Var(m.set_timeslots,m.set_scenarios, domain=Reals) # without losses
m.Pparking = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.Pprofile = Var(m.set_timeslots, domain=NonNegativeReals)
m.ES = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.ES_battery = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.mu = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.ESnet = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
```

```
m.Egrid = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals) # wtih losses
m.ENoLoss = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals) # without losses
m.ENoLoss_battery = Var(m.set_timeslots,m.set_scenarios, domain=Reals) # without losses
m.Pparking = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.Pprofile_flex = Var(m.set_timeslots, domain=NonNegativeReals)
m.ES = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.ES_battery = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.mu = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.ESnet = Var(m.set_timeslots,m.set_scenarios, domain=NonNegativeReals)
m.max_reduction=Var(domain=NonNegativeReals)
```

d) Objective function

```
# objective function
def obj_expression(m):
    OF1 = sum(m.priceEn[t]*m.Egrid[t,s]+pen_dev*(m.Pprofile[t]-m.Pparking[t,s])**2 for t in m.set_timeslots for s in m.set_scenarios)
    OF2 = sum(pen_dev*(m.ES_battery[n_periods-1,s]-ES_ini_battery)**2 for s in m.set_scenarios)
    return OF1+OF2
m.OBJ = Objective(rule=obj_expression)
```

```
# objective function
def obj_expression(m):
    OF1 = sum(m.priceEn[t]*m.Egrid[t,s]-reward_flex*(m.max_reduction) for t in m.set_timeslots for s in m.set_scenarios)
    OF2 = sum(pen_dev*(m.ES_battery[n_periods-1,s]-ES_ini_battery)**2 for s in m.set_scenarios)
    return OF1+OF2
m.OBJ = Objective(rule=obj_expression)
```

e) Constraints

```
#constraint (enegy balance)
def en_balance(m, t, s):
    expr2 = 0 # initialize summing variable
    expr2 +=m.ES[t,s]-m.ENoLoss[t,s]-m.ESpos[t,s]+m.ESneg[t,s]
    if t>0:
        expr2 +=-m.ES[t-1,s]
    # return the expression for the constraint for i
    return expr2==0
m.en_balance = Constraint(m.set_timeslots, m.set_scenarios, rule=en_balance)

#constraint (net enegy balance)
def net_en_balance(m, t, s):
    expr2 = m.ESnet[t,s]-m.ENoLoss[t,s]-m.mu[t,s]*m.ESpos[t,s]+m.ESneg[t,s]-sum((1-m.mu[t,s])*m.Eini[j,t,s] for j in range(t))
    if t>0:
        expr2 +=-m.ESnet[t-1,s]
    # return the expression for the constraint for i
    return expr2==0
m.net_en_balance = Constraint(m.set_timeslots, m.set_scenarios, rule=net_en_balance)

#constraint (enegy balance for the additional battery)
def en_balance_battery(m, t, s):
    expr2 = 0 # initialize summing variable
    expr2 +=m.ES_battery[t,s]-m.ENoLoss_battery[t,s]
    if t>0:
        expr2 +=-m.ES_battery[t-1,s]
    else: expr2 = m.ES_battery[t,s]-ES_ini_battery
    # return the expression for the constraint for i
    return expr2==0
m.en_balance_battery = Constraint(m.set_timeslots, m.set_scenarios, rule=en_balance_battery)
```

```
#constraint (max Egrid)
def Egrid_max(m, t, s):
    if t>0:
        return m.Egrid[t,s]-(Pmax_EV*m.Nparch[t-1,s]+Pmax_battery)*deltaT<=0
    return m.Egrid[t,s]<=0
m.Egrid_max = Constraint(m.set_timeslots, m.set_scenarios, rule=Egrid_max)


#constraint (max ES assuming that ESnet<ES)
def ES_max(m, t,s):
    if t>0:
        return m.ES[t,s]-Cnom_EV*m.Nparch[t,s]<=0
    return m.ES[t,s]<=0
m.ES_max = Constraint(m.set_timeslots, m.set_scenarios, rule=ES_max)

#constraint
def ES_max_battery(m, t,s):
#    if t<n_periods-1:
        return m.ES_battery[t,s]-battery_size<=0
#    return m.ES_battery[t,s]==ES_ini_battery
m.ES_max_battery = Constraint(m.set_timeslots, m.set_scenarios, rule=ES_max_battery)
```

```
#constraint (max mu)
def mu_max(m, t, s):
    return m.mu[t,s]-m.muMax[t,s]<=0
m.mu_max = Constraint(m.set_timeslots, m.set_scenarios, rule=mu_max)

#constraint (energy from the grid)
def P_pk(m, t, s):
    return m.Egrid[t,s]-1/eta*(m.ENoLoss[t,s])-m.ENoLoss_battery[t,s]==0#-m.loss_battery[t,s]==0
m.P_pk = Constraint(m.set_timeslots, m.set_scenarios, rule=P_pk)
```

```
#constraint (Pparking power from the grid)
def Pp(m, t, s):
    return m.Pparking[t,s]*deltaT-m.Egrid[t,s]==0
m.Pp = Constraint(m.set_timeslots, m.set_scenarios, rule=Pp)

#constraint (max Pparking)
def Pparking_max(m, t, s):
    if t>0:
        return m.Pparking[t,s]-Pp_max-Pmax_battery<=0
    return m.Pparking[t,s]<=0
m.Pparking_max = Constraint(m.set_timeslots, m.set_scenarios, rule=Pparking_max)

#constraint (difference_profiles)
def difference_profiles_up(m, t, s):
    return m.Pprofile[t]-m.Pparking[t,s]<=tol_profile
m.difference_profiles_up = Constraint(m.set_timeslots, m.set_scenarios, rule=difference_profiles_up)

def difference_profiles_down(m, t, s):
    return m.Pparking[t,s]-m.Pprofile[t]<=tol_profile
m.difference_profiles_down = Constraint(m.set_timeslots, m.set_scenarios, rule=difference_profiles_down)
```

```
#constraint (max reduction)
def Max_reduction(m, t):
    if t<t_flex or t>t_flex+flex_recover:
        return m.Pprofile_flex[t]-m.Pprofile_ref[t]==0
    elif t==t_flex:
        return m.max_reduction+m.Pprofile_flex[t]-m.Pprofile_ref[t]==0
    elif t>t_flex and t<=t_flex+flex_recover:
        return m.max_reduction/flex_recover-m.Pprofile_flex[t]+m.Pprofile_ref[t]>=0
    return m.Pprofile_flex[t]>=0
m.Max_reduction = Constraint(m.set_timeslots, rule=Max_reduction)
```

Chapter 5

Test Results

The results of all the model are illustrated in this section. The model comparison with different values of parameters also discussed.

5.1 Results for model A

The simulation result done in Pyomo are discussed here. Firstly, the results obtained from the optimization model with single data are reviewed. The objective value of the particular single case is OF=398.991. The graph of energy in the grid in MWhr versus time is shown below.

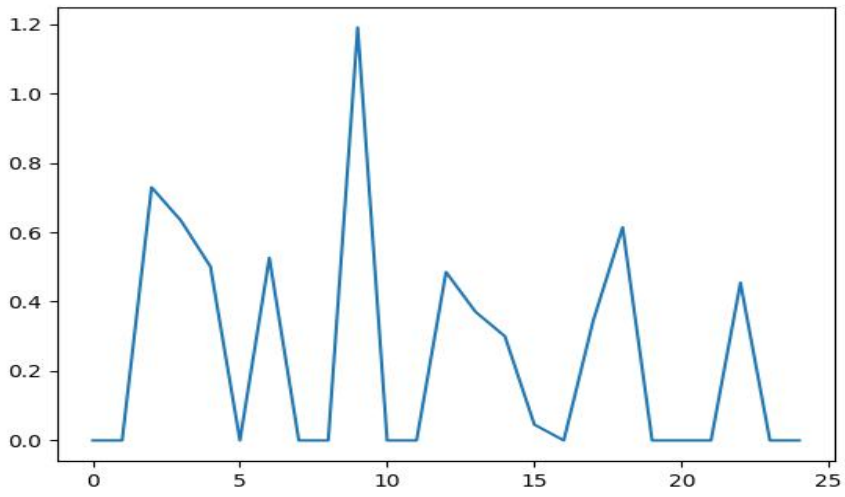


Figure 5.1- Energy in the grid vs time ($\mu^{\omega,t}$ =0)

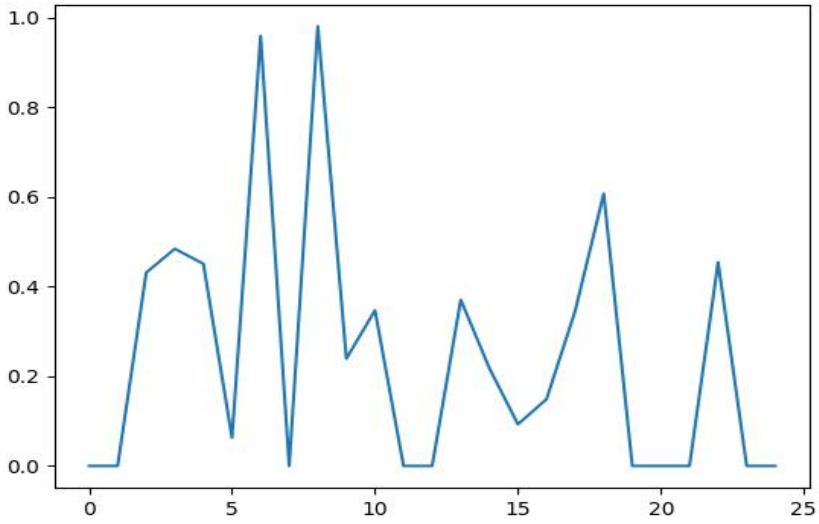


Figure 5.2- Energy in the grid vs time ($\mu^{\omega,t} \leq 1$)

From these two graphs, we can infer that the energy in the grid has been adjusted without any change in the objective value. i.e., in both condition the same objective value. Variable $\mu^{\omega,t}$ describes the utilization of the energy initially stored in the vehicles that enters in the parking lot. For $\mu^{\omega,t} = 0$ (no utilization of the initial energy), there are two points the graph shows peak values, one in the beginning t=1 and in t=18 while for $\mu^{\omega,t} \leq 1$ (possible full utilization of the initial energy),, it shows high values in t=6 and t=8.

The graph below shows the comparison of variables by considering the $\mu^{\omega,t} = 0$ and $\mu^{\omega,t} \leq 1$.

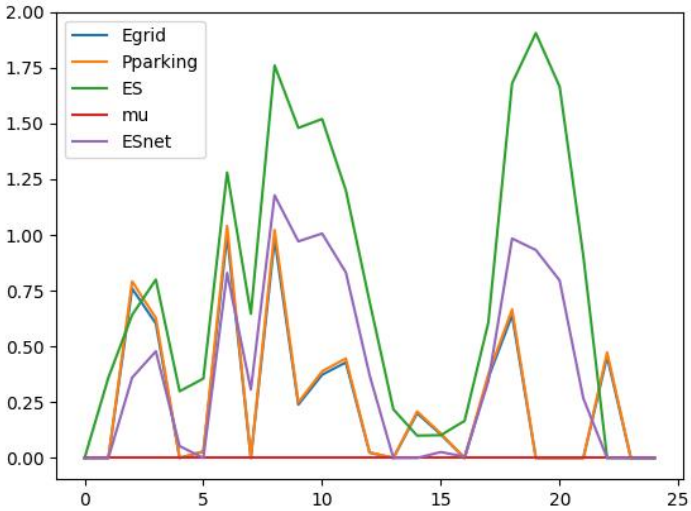


Figure 5.3- Variables vs time when $\mu^{\omega,t} = 0$

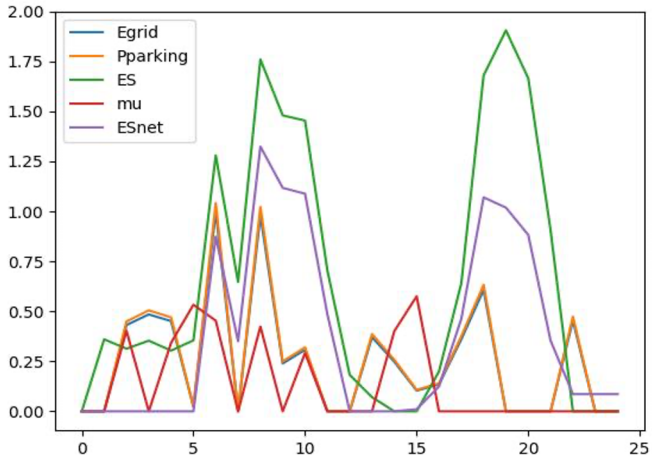


Figure 5.4- Variables vs time when $\mu^{\omega,t} \leq 1$

By comparing the above two graphs we could conclude that as the $\mu^{\omega,t}$ value changes, all the other variables shows variations. The $\mu^{\omega,t}$ value have a significant role in providng changes in the variables.

The figure below depicts the output obtained for single scenario when $\mu^{\omega,t} = 0$ and the objective values of eight scenarios.

```
OF= 415.61574844919795
OF= 418.63182180841676
OF= 431.61027371955214
OF= 451.02825439966665
OF= 406.875804512698
OF= 425.04523199526045
OF= 423.4543654365208
OF= 425.2850995781354
```

	A	B	C	D	E	F
1	Egrid	ENoLoss	Pparking	ES	mu	ESnet
2	0	0	0	0	0	0
3	0	0	0	0.360151	0	0
4	0.76	0.7296	0.76	0.611196	0	0.329751
5	0.659709	0.633321	0.659709	0.8	0	0.478371
6	0.070247	0.067437	0.070247	0.3666	0	0.12079
7	0	0	0	0.395079	0	0.038907
8	1	0.96	1	1.28	0	0.829887
9	0	0	0	0.647163	0	0.306956
10	1.022142	0.981257	1.022142	1.76	0	1.177593
11	0	0	0	1.24036	0	0.731605
12	0	0	0	0.907115	0	0.393753
13	0.217879	0.209164	0.217879	0.367667	0	0
14	0.505575	0.485352	0.505575	0.330007	0	0
15	0.386319	0.370866	0.386319	0.218321	0	0
16	0.312634	0.300128	0.312634	0.2	0	0.099656
17	0.047597	0.045693	0.047597	0.142206	0	0.066629
18	0	0	0	0.206419	0	0.045693
19	0.36	0.3456	0.36	0.633155	0	0.36966
20	0.64	0.6144	0.64	1.68	0	0.98406
21	0	0	0	1.905293	0	0.932529
22	0	0	0	1.665293	0	0.79639
23	0	0	0	0.905293	0	0.267299
24	0.473653	0.454707	0.473653	0	0	0
25	0	0	0	0	0	0
26	0	0	0	0	0	0

Figure 5.5 Objective values of scenarios and the output of the variables for a single scenario.

The graph of the output Egrid of eight different scenarios is listed below.

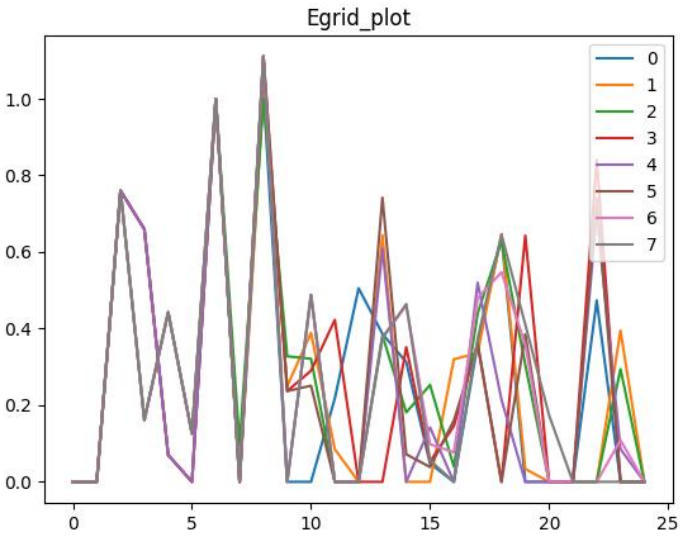


Figure5.6 - Egrid vs time plot for 8 scenarios

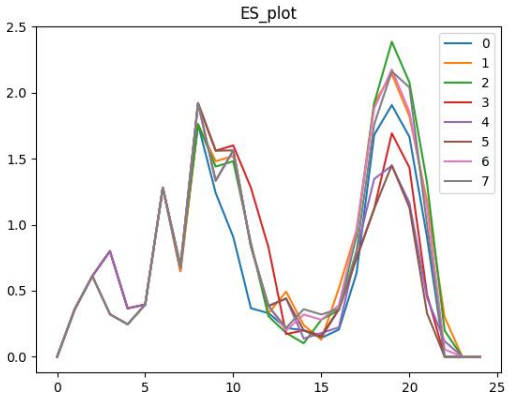


Figure 5.7 - ES vs time plot

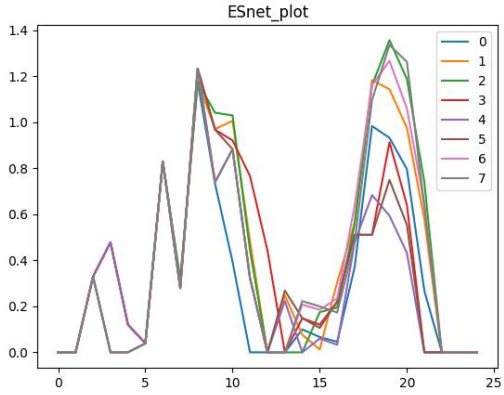


Figure 5.8- ESnet vs time plot

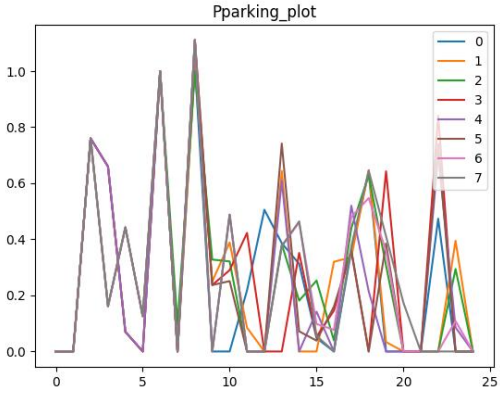


Figure 5.9- Pparking vs time plot

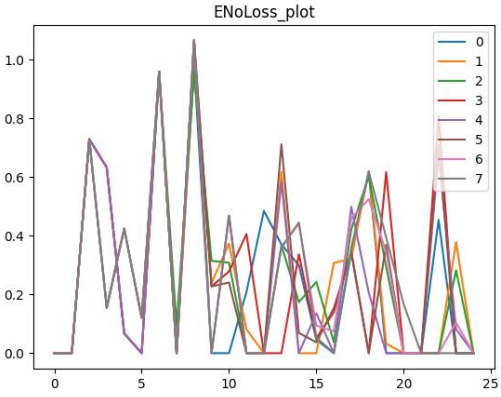


Figure 5.10- ENoloss vs time plot

The fig. (5.7-5.10) above shows the variables versus time of eight scenarios used in model.

5.2 Results for model B

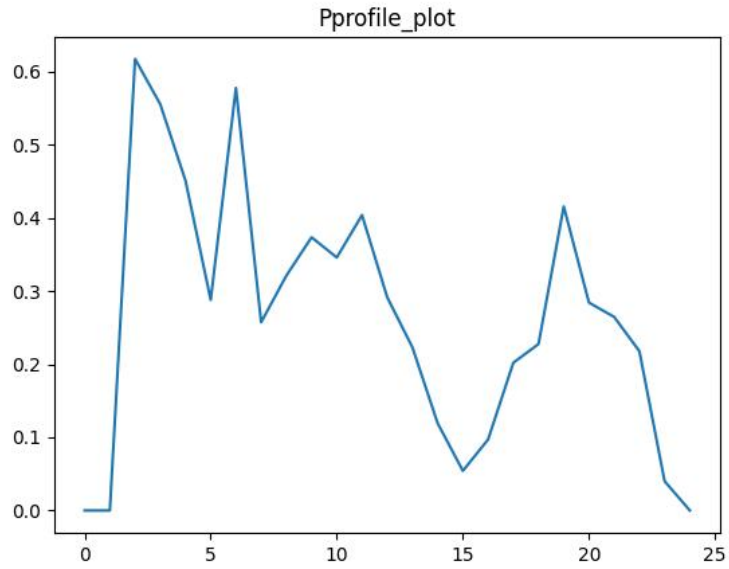


Figure 5.11 - Common Pprofile vs time plot

The figure above shows the common forecasted profile for the next day. The forecasted profile helps in reducing the consumption of energy from the grid.

The model created the robust profile for all the other eight scenarios also and it is shown in figure 5.12.

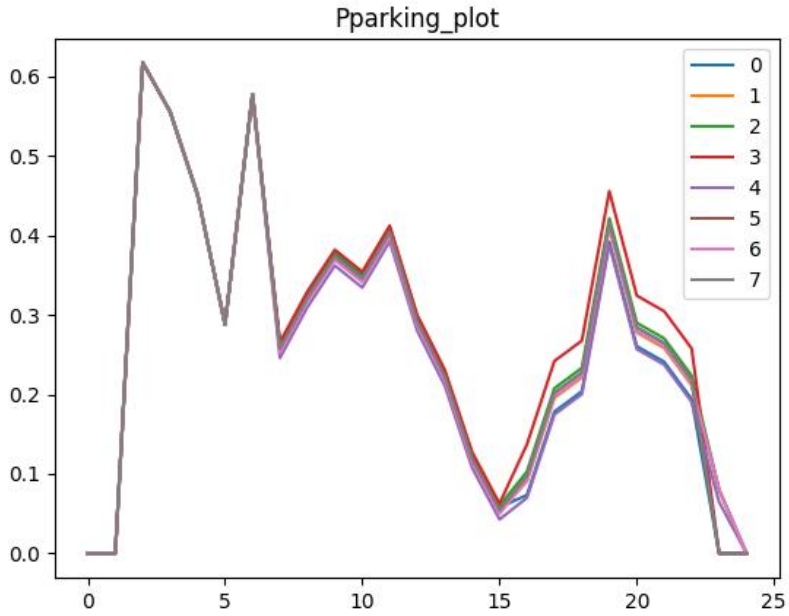


Figure 5.12- Pparking plot vs time for 8 scenarios

5.3 Results for model C

The figure below shows the results of the maximum reduction that occurs at which time has been obtained in a single excel file and how much reduction can be made is also expressed in this excel file.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC
1		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24		reference profile	
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	
3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	
4	2	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509	0.376509		0.376509	
5	3	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897	0.464897		0.464897	
6	4	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727	0.402727		0.402727	
7	5	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823	0.085823		0.085823	
8	6	0.96	0.96	0.96	0.96	0.96	0.96	0.84251	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96		0.96	
9	7	0	0	0	0	0	0	0.039163	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	
10	8	0.96	0.96	0.96	0.96	0.96	0.96	0.999163	0.96	0	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96		0.96	
11	9	0.209625	0.209625	0.209625	0.209625	0.209625	0.209625	0.248789	0.209625	0.529625	0	0.209625	0.209625	0.209625	0.209625	0.209625	0.209625	0.209625	0.209625	0.209625	0.209625	0.209625	0.209625	0.209625	0.209625	0.209625		0.209625	
12	10	0	0	0	0	0	0	0	0	0.32	0.069875	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	
13	11	0	0	0	0	0	0	0	0	0.32	0.069875	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	
14	12	0.375688	0.375688	0.375688	0.375688	0.375688	0.375688	0.375688	0.375688	0.375688	0.445563	0.375688	0.375688	0.353829	0.375688	0.375688	0.375688	0.375688	0.375688	0.375688	0.375688	0.375688	0.375688	0.375688	0.375688	0.375688		0.375688	
15	13	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.353605	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319	0.346319		0.346319	
16	14	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.165095	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809	0.157809		0.157809	
17	15	0.042	0.042	0.042	0.042	0.042	0.042	0.042	0.042	0.042	0.042	0.042	0.042	0.049286	0.042	0.042	0.042	0.042	0.042	0.042	0.042	0.042	0.042	0.042	0.042	0.042		0.042	
18	16	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12	0.12		0.12	
19	17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	
20	18	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0	0.68	0.68	0.68	0.68	0.68	0.68		0.68	
21	19	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	0.790679	1.017345	0	0.790679	0.790679	0.790679	0.790679	0.790679		0.790679	
22	20	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.266667	0.30356	0.04	0.04	0.04	0.04	0.04		0.04	
23	21	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.03895	0.265616	0.302509	0.03895	0.03895	0.03895	0.03895	0.03895		0.03895	
24	22	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.053921	0.317481	0.053921	0.053921	0.053921	0.053921	0.053921		0.053921	
25	23	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0	0.04		0.04	
26	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	
27																													
28																													
29																													
30																													
31																													
32	max_reduction in MW							0.11749		0.96	0.209625									0.68	0.790679								
33	objectives in €							-1270.831		-34990.75	-4975.757									-23790.21	-28217.35							3409.254	
34																													

Figure5.13- Data shows the maximum reduction.

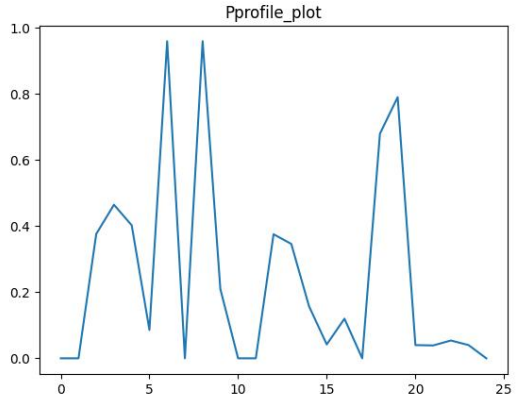


Figure 5.14- Pprofile vs time plot

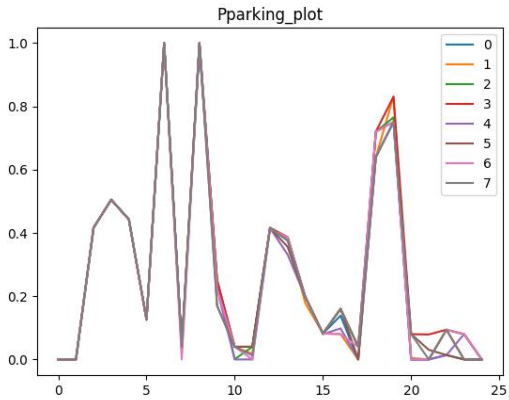


Figure 5.15- Pparking vs time plot

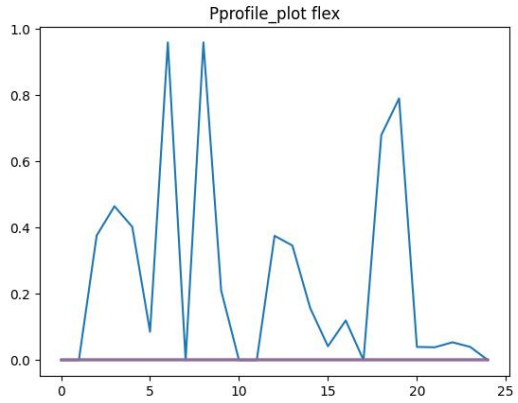


Figure 5.16- Pprofile_flex vs time at t=0

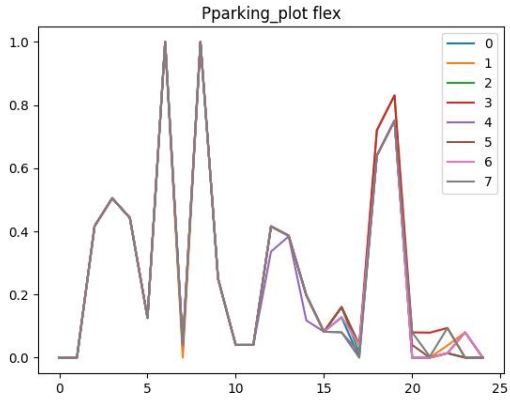


Figure 5.17- Pparking_flex vs time at t=0

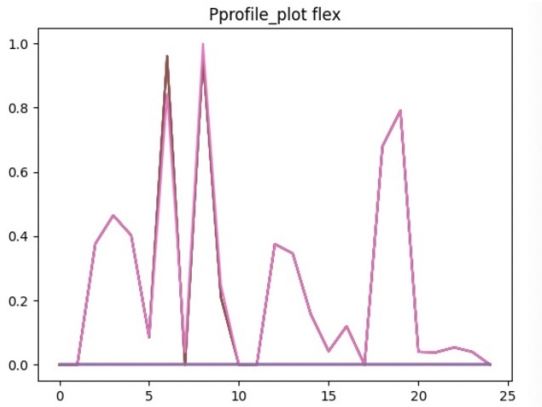


Figure 5.18- Pprofile-flex vs time at t=6

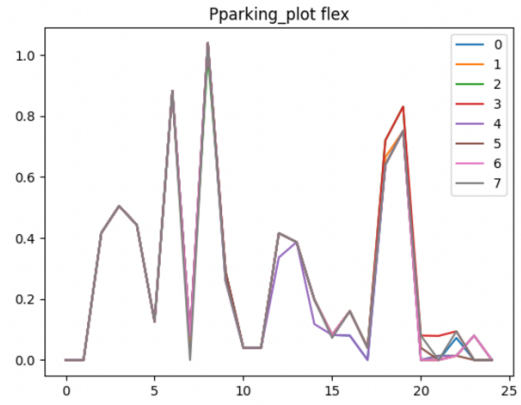


Figure 5.19- Pparking_flex vs time at t=6

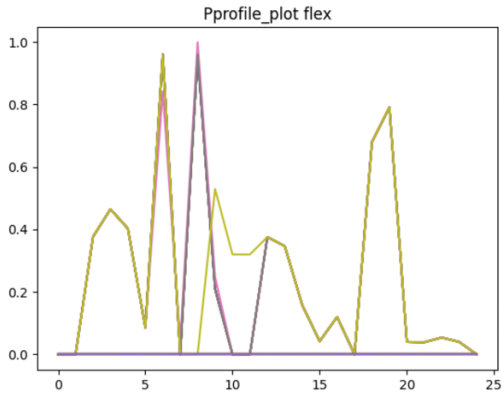


Figure 5.20- Pprofile-flex vs time at t=8

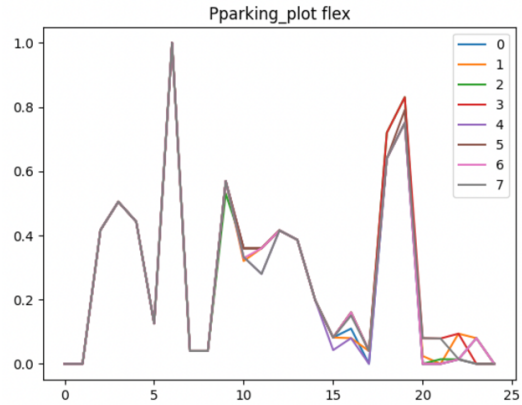


Figure 5.21- Pparking_flex vs time at t=8

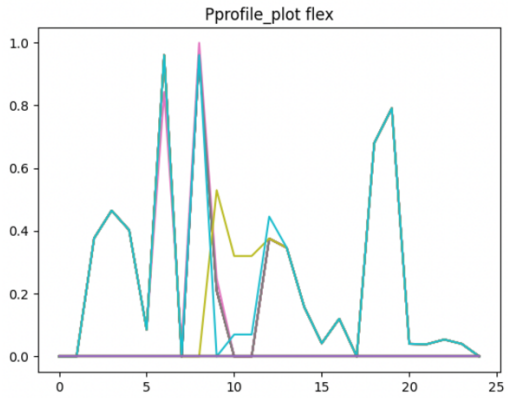


Figure 5.22- Pprofile-flex vs time at t=9

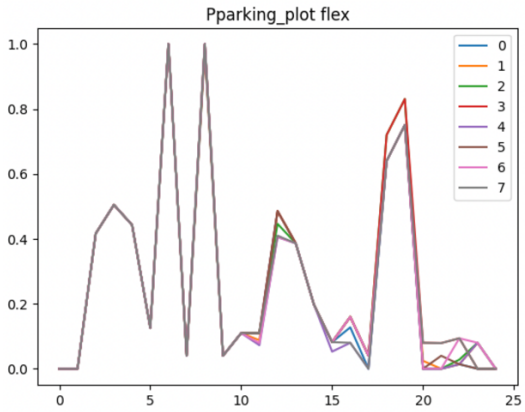


Figure 5.23- Pparking_flex vs time at t=9

The figure 5.14 and 5.15 shows the *Pprofile* and *Pparking* versus time plot respectively. The next two images 5.16 and 5.17 *Pprofile_flex* and *Pparking_flex* versus time at t=0. At this time there is no reduction occurs as it can be clearly seen from the figure 5.13. The figures from 5.18 to 5.23 show that the reduction happens at specific time.

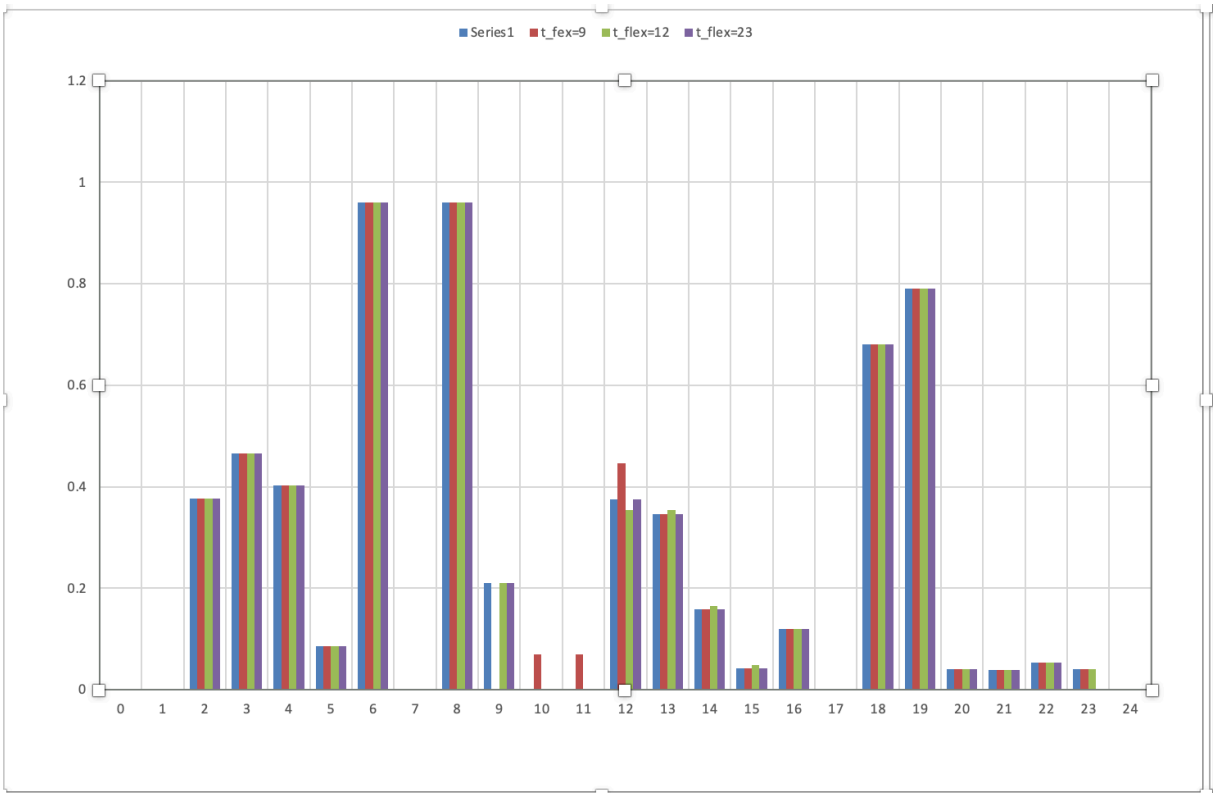


Figure 5.24- Graph shows the max reduction.

The above graph, figure 5.24, shows the maximum reduction happens at time *t_flex*=9, 12 and 23. From the inference we can obtain that whenever there occurs a reduction it is recovered in the next 3 periods.

5.4 Results for model D

The data file below figure 5.25 shows the maximum reduction that occurs at each hour. Figure 5.26 shows the common profile data which details the reduction and recovery happens at every period. The forecasted profile plot and the parking plot are shown in figure 5.27 and figure 5.28 respectively. There is not much difference in both the plots as the parking plot follows exactly the common profile for all the eight scenarios.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	0.056098	8.04E-15	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098
4	2	0.207397	0.226096	1.94E-14	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397
5	3	0.46365	0.482349	0.532782	7.89E-14	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365
6	4	0.41278	0.431479	0.481912	0.56733	3.68E-16	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278
7	5	0.375508	0.375508	0.44464	0.530058	0.513101	2.4E-14	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508
8	6	0.974523	0.974523	0.974523	1.129073	1.112117	1.099693	1.15E-15	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523
9	7	0.136049	0.136049	0.136049	0.136049	0.273643	0.261219	0.460891	3.59E-15	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049
10	8	0.196965	0.196965	0.196965	0.196965	0.196965	0.322134	0.521806	0.242314	3.28E-17	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965
11	9	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.632093	0.352602	0.372907	3.01E-15	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252
12	10	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.342533	0.362838	0.3996	9.01E-15	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183
13	11	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.397432	0.434194	0.430838	3.12E-15	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777
14	12	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.374738	0.371382	0.382913	4.55E-14	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321
15	13	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.314989	0.32652	0.306701	3.68E-14	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928
16	14	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.241173	0.221355	0.202557	1.65E-16	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581
17	15	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.191856	0.173058	0.144609	9.33E-14	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082
18	16	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.17376	0.145311	0.135478	9.33E-16	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784
19	17	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.174702	0.164869	0.165103	4.91E-15	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175
20	18	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.225527	0.225761	0.235558	1.82E-15	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833
21	19	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.423008	0.432804	0.453024	1.28E-16	0.38908	0.38908	0.38908	0.38908	0.38908
22	20	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.345684	0.365903	0.431652	0.301959	0.301959	0.301959	0.301959	0.301959
23	21	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.292302	0.358051	0.228358	0.228358	0.228358	0.228358	0.228358
24	22	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.251358	0.121664	0.121664	0.121664	0.121664	0.121664
25	23	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255068	0.255091	0.255087	4.26E-15	0.255115
26	24	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154654	0.154651	0.239724	5.06E-14
27																										
28																										
29																										
30																										
31																										
32	max_reduction in MW	0.056098	0.207397	0.46365	0.41278	0.375508	0.974523	0.136049	0.196965	0.307252	0.297183	0.331777	0.272321	0.215928	0.130581	0.101082	0.101784	0.131175	0.191833	0.38908	2.15E-14	3.36E-14	5.88E-14	0.255115	0.154686	
33	objectives in €	1029.966	-5021.96	-15272.1	-13214.4	-11704.8	-35545.1	-2168.08	-4604.68	-9016.17	-8613.43	-9997.17	-7618.93	-5363.21	-1949.34	-769.392	-797.466	-1973.08	-4399.42	-12289.3	3273.901	3273.901	3273.901	-6907.54	-2894.41	
34																										

Figure 5.25- Data of maximum reduction with battery

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	0.056098	8.04E-15	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098	0.056098
4	2	0.207397	0.226096	1.94E-14	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397	0.207397
5	3	0.46365	0.482349	0.532782	7.89E-14	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365	0.46365
6	4	0.41278	0.431479	0.481912	0.56733	3.68E-16	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278	0.41278
7	5	0.375508	0.375508	0.44464	0.530058	0.513101	2.4E-14	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508	0.375508
8	6	0.974523	0.974523	0.974523	1.129073	1.112117	1.099693	1.15E-15	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523	0.974523
9	7	0.136049	0.136049	0.136049	0.136049	0.273643	0.261219	0.460891	3.59E-15	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049	0.136049
10	8	0.196965	0.196965	0.196965	0.196965	0.196965	0.322134	0.521806	0.242314	3.28E-17	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965	0.196965
11	9	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.632093	0.352602	0.372907	3.01E-15	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252	0.307252
12	10	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.342533	0.362838	0.3996	9.01E-15	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183	0.297183
13	11	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.397432	0.434194	0.430838	3.12E-15	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777	0.331777
14	12	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.374738	0.371382	0.382913	4.55E-14	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321	0.272321
15	13	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.314989	0.32652	0.306701	3.68E-14	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928	0.215928
16	14	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.241173	0.221355	0.202557	1.65E-16	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581	0.130581
17	15	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.191856	0.173058	0.144609	9.33E-14	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082	0.101082
18	16	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.17376	0.145311	0.135478	9.33E-16	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784	0.101784
19	17	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.174702	0.164869	0.165103	4.91E-15	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175	0.131175
20	18	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833	0.225527	0.225761	0.235558	1.82E-15	0.191833	0.191833	0.191833	0.191833	0.191833	0.191833
21	19	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.38908	0.423008	0.432804	0.453024	1.28E-16	0.38908	0.38908	0.38908	0.38908	0.38908
22	20	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.301959	0.345684	0.365903	0.431652	0.301959	0.301959	0.301959	0.301959	0.301959
23	21	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.228358	0.292302	0.358051	0.228358	0.228358	0.228358	0.228358	0.228358
24	22	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.121664	0.251358	0.121664	0.121664	0.121664	0.121664	0.121664
25	23	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255115	0.255068	0.255091	0.255087	4.26E-15	0.255115
26	24	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154686	0.154654	0.154651	0.239724	5.06E-14
27																										

Figure 5.26- Data of profiles that contain both the reduction and recovery.

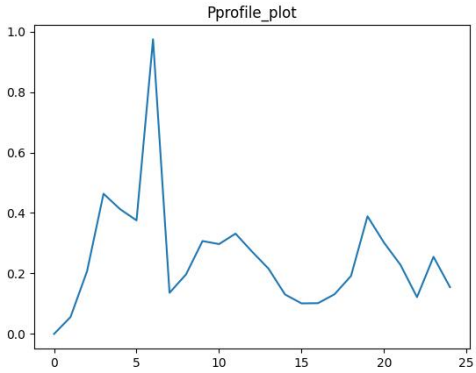


Figure 5.27- Pprofile vs time plot

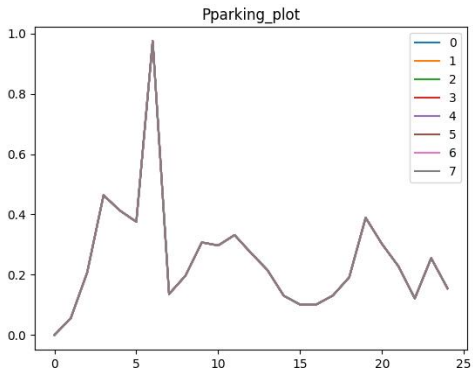


Figure 5.28- Pparking vs time plot

` The energy balancing of additional battery versus time is shown in figure 5.29. The auxiliary battery is considered fully charged at the beginning of the day. It can be seen from the graph that, at the end of the day, in all scenarios, the battery tries to reach fully charged condition. Figure 5.30 shows the maximum reduction and its recovery with auxiliary battery at *t_flex* equals to 9, 12 and 23. The reduced amount of energy is recovered in the next three periods.

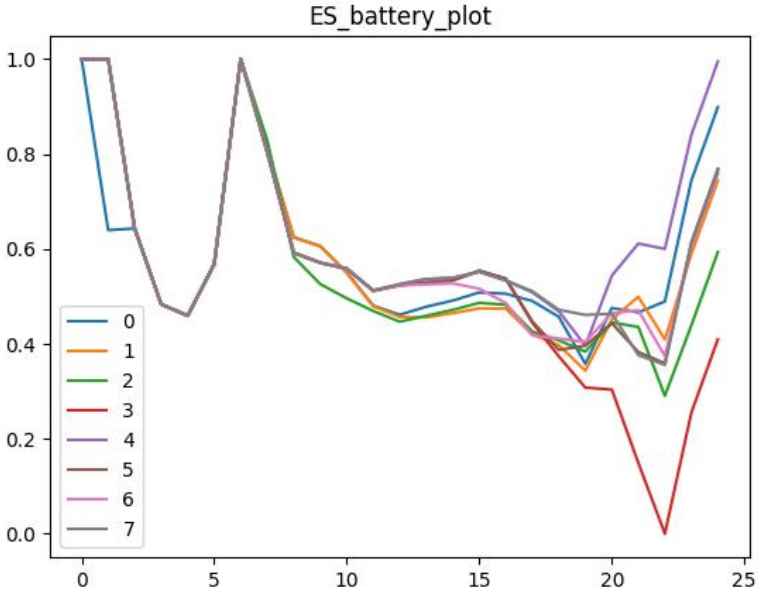


Figure5.29- ES_battery vs time plot for 8 scenarios

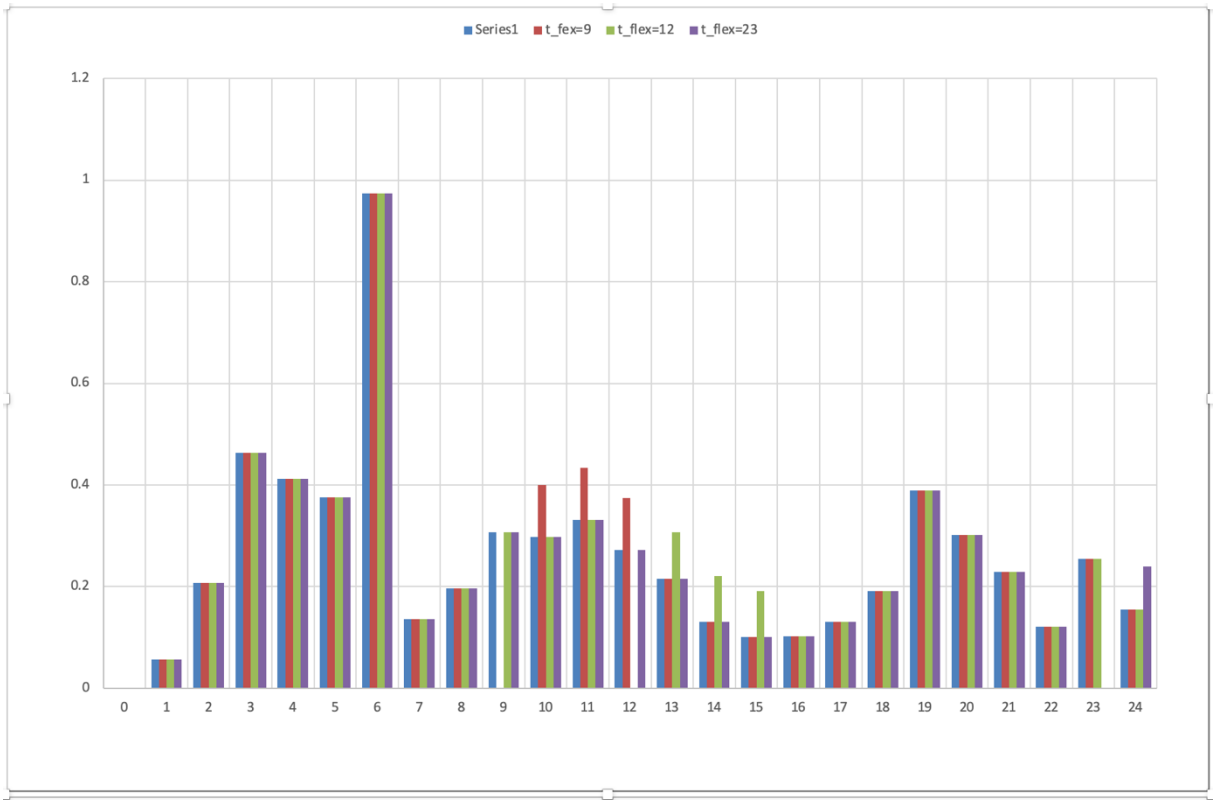


Figure 5.30- Graph showing the maximum reduction with battery.

Chapter 6
Conclusion

The thesis presents a detailed study of the optimization of electric vehicle parking lot for the provision of flexible services. The study starts by analyzing the scheduling of charging the electric vehicles. To achieve this, a deterministic approach has been implemented by considering several scenarios. From model A, we could obtain an optimal value of procurement cost for single scenario and for eight other scenarios which we considered. This needs an approach that need robust optimization. The robust optimization framework involves formulating the optimization problem with a set of possible scenarios or uncertainty sets.

The approach presented in model B is to obtain a forecasted common profile for the next day so that the model can work according to that specific profile and thus it tries to reduce the energy consumption from the grid. We also extended the generation of common profile to all the scenarios.

In model C, the aim is to obtain a maximum reduction at each duration. The results show that the model reduces the energy consumption from grid at certain period and then started to compensate within the next periods. If the reduction can be requested more than one period per day and with a delay between consecutive requests smaller than the recovery period, a more complex robust approach should be implemented.

Final model D considers reducing the tolerance and to compensate the uncertainties by adding an auxiliary battery. The presence of the auxiliary battery acts as a flexibility service for both the provider and consumer as it aids in the maximum reduction, compensating the uncertainties associated with the number and characteristics of the parked vehicles.

The optimization model for an electric vehicle parking lot to provide flexibility services to the distribution system operator is a useful tool for managing the integration of electric vehicles into the power system. The implemented models consider various factors such as parking lot capacity, EV charging demand, and distribution system operator’s flexibility needs.

The models aim to minimize the operational costs of the parking lot while ensuring that the distribution system operator’s flexibility requirements are met. This is achieved by scheduling EV charging based on the parking lot's capacity, the electricity price, and the forecasted distribution system operator’s flexibility needs.

Implementing this optimization approach can lead to several benefits such as reducing peak demand on the electricity grid, increasing renewable energy integration, and improving the stability and reliability of the power system. Moreover, electric vehicle owners can benefit from lower charging costs and the possibility of receiving incentives for providing flexibility services.

The optimization of the electric vehicle parking lot is needed for managing the integration of electric vehicles into the power system while also ensuring the stability and reliability of the grid.

References

[1] Xiang, Yue, Shuai Hu, Youbo Liu, Xin Zhang, and Junyong Liu. 2019. "Electric Vehicles in Smart Grid: A Survey on Charging Load Modelling." IET Smart Grid 2 (1): 25–33. https://doi.org/10.1049/iet-stg.2018.0053.

[2] García-Villalobos, J., I. Zamora, J. I. San Martín, F. J. Asensio, and V. Aperribay. 2014. "Plug-in Electric Vehicles in Electric Distribution Networks: A Review of Smart Charging Approaches." Renewable and Sustainable Energy Reviews 38: 717–31. https://doi.org/10.1016/j.rser.2014.07.040.

[3] Sbordone, D., I. Bertini, B. Di Pietra, M. C. Falvo, A. Genovese, and L. Martirano. 2015. "EV Fast Charging Stations and Energy Storage Technologies: A Real Implementation in the Smart Micro Grid Paradigm." Electric Power Systems Research 120: 96–108. https://doi.org/10.1016/j.epsr.2014.07.033.

[4] Mwasilu, Francis, Jackson John Justo, Eun-Kyung Kim, Ton Duc Do, and Jin-Woo Jung. 2014. "Electric Vehicles and Smart Grid Interaction: A Review on Vehicle to Grid and Renewable Energy Sources Integration." Renewable and Sustainable Energy Reviews 34 (June): 501–16. https://doi.org/10.1016/j.rser.2014.03.031.

[5] Traube, Joshua, Fenglong Lu, Dragan Maksimovic, Joseph Mossoba, Matthew Kromer, Peter Faill, Stan Katz, Bogdan Borowy, Steve Nichols, and Leo Casey. 2013. "Mitigation of Solar Irradiance Intermittency in Photovoltaic Power Systems with Integrated Electric-Vehicle Charging Functionality." IEEE Transactions on Power Electronics 28 (6): 3058–67. https://doi.org/10.1109/TPEL.2012.2217354.

[6] Zhang, Tian, Wei Chen, Zhu Han, and Zhigang Cao. 2014. "Charging Scheduling of Electric Vehicles with Local Renewable Energy under Uncertain Electric Vehicle Arrival and Grid Power Price." IEEE Transactions on Vehicular Technology 63 (6): 2600–2612. https://doi.org/10.1109/TVT.2013.2295591.

[7] Derakhshandeh, S. Y., Amir S. Masoum, Sara Deilami, Mohammad A. S. Masoum, and M. E. Hamedani Golshan. 2013. "Coordination of Generation Scheduling with PEVs Charging in Industrial Microgrids." IEEE Transactions on Power Systems 28 (3): 3451–61. https://doi.org/10.1109/TPWRS.2013.2257184.

[8] Battistelli, C., L. Baringo, and a. J. Conejo. 2012. "Optimal Energy Management of Small Electric Energy Systems Including V2G Facilities and Renewable Energy Sources." Electric Power Systems Research 92: 50–59. https://doi.org/10.1016/j.epsr.2012.06.002.

[9] Zakariazadeh, Alireza, Shahram Jadid, and Pierluigi Siano. 2015. "Integrated Operation of Electric Vehicles and Renewable Generation in a Smart Distribution System." Energy Conversion and Management 89: 99–110. https://doi.org/10.1016/j.enconman.2014.09.062.

[10] Richardson, David B. 2013. "Encouraging Vehicle-to-Grid (V2G) Participation through Premium Tariff Rates." Journal of Power Sources 243: 219–24. https://doi.org/10.1016/j.jpowsour.2013.06.024.

[11] Dabbagh, Saeed Rahmani, Mohammad Kazem Sheikh-El-Eslami, and Alberto Borghetti. 2016. "Optimal Operation of Vehicle-to-Grid and Grid-to-Vehicle Systems Integrated with Renewables." 19th Power Systems Computation Conference, PSCC 2016, 1–7. https://doi.org/10.1109/PSCC.2016.7540933.

[12] Honarmand, Masoud, Alireza Zakariazadeh, and Shahram Jadid. 2014. "Integrated Scheduling of Renewable Generation and Electric Vehicles Parking Lot in a Smart Microgrid." Energy Conversion and Management 86: 745–55. https://doi.org/10.1016/j.enconman.2014.06.044.

[13] Khodayar, Mohammad E., Lei Wu, and Mohammad Shahidehpour. 2012. "Hourly Coordination of Electric Vehicle Operation and Volatile Wind Power Generation in SCUC." IEEE Transactions on Smart Grid 3 (3): 1271–79. https://doi.org/10.1109/TSG.2012.2186642.

[14] Shafie-Khah, Miadreza, Ehsan Heydarian-Forushani, Gerardo J. Osorio, Fabio A.S. Gil, Jamshid Aghaei, Mostafa Barani, and Joao P.S. Catalao. 2016. "Optimal Behavior of Electric

Vehicle Parking Lots as Demand Response Aggregation Agents." IEEE Transactions on Smart Grid 7 (6): 2654–65. https://doi.org/10.1109/TSG.2015.2496796.

[15] Borghetti, A., F. Napolitano, S. Rahmani-Dabbagh, and F. Tossani. 2017. "Scenario Tree Generation for the Optimization Model of a Parking Lot for Electric Vehicles." In 2017 AEIT International Annual Conference. Cagliari, Italy. https://doi.org/10.23919/AEIT.2017.8240519.

[16] Orozco C.; Borghetti A.; Napolitano F.; Tossani F., Day-ahead Multistage Stochastic Optimization of a Group of Electric Vehicle Charging Stations, in: 2021 IEEE 15th International Conference on Compatibility, Power Electronics and Power Engineering, CPE-POWERENG 2021. https://dx.doi.org/10.1109/CPE-POWERENG50821.2021.9501228