Faculty of Engineering of the University of Porto

An Integrated Approach to Electric Vehicle Charging Management in Multi-Dwelling Buildings: From Dynamic Load Allocation to Infrastructure Planning

Salvador Moreira Paes Carvalhosa

FINAL VERSION

Thesis carried out under the
Doctoral Program in Sustainable Energy Systems

Advisor: Professor José Rui Ferreira, Ph.D.
Co-Advisor: Professor Rui Esteves Araújo, Ph.D.

January 2026

Resumo

Embora os veículos elétricos (VE) sejam contemporâneos dos veículos movidos a combustíveis fósseis, a sua adoção permaneceu marginal durante grande parte do século XX. Nesse intervalo, o parque edificado e os modelos de vida urbana evoluíram sem atender às especificidades de carregamento dos VE. A generalisação de edifícios multifamiliares com garagens e estacionamentos partilhados conduziu a instalações elétricas dimensionadas para iluminação e serviços comuns, com contratação e medição orientadas ao consumo convencional dos condóminos, e não a cargas de mobilidade elétrica.

Com a expansão recente dos VE, emergem limitações técnicas (capacidade do alimentador principal, seletividade e coordenação de proteções), económicas (tarifas e repartição de custos) e regulatórias (regras de medição e exploração em condomínios), que dificultam uma integração célere e eficiente. Esta tese aborda estas limitações de forma integrada e propõe uma solução tecnicamente robusta e economicamente proporcional para a realidade dos edifícios existentes, abrindo o caminho para que a adoção da mobilidade elétrica seja uma realida a curto prazo.

Em concreto, apresenta-se: (i) uma topologia de carregamento centralizado a partir do alimentador principal, com gestão dinâmica da potência disponível; (ii) um optimisador em tempo quase real que garante energia requerida até ao momento de saída declarado pelo utilizador, respeitando limites de tensão/corrente e uma reserva de segurança para consumos não-VE; (iii) um quadro de coeficientes de simultaneidade específico para frotas em condomínios, que fundamenta o dimensionamento de condutores e proteções sem recorrer ao caso extremo C_s=1; e (iv) uma camada de agendamento sensível à degradação das baterias, que internaliza paralelismos do envelhecimento eletroquímico e constrangimentos térmicos, preservando a mobilidade e a longevidade das baterias.

Resultados em dados reais de um edifício multifamiliar mostram que a abordagem proposta contém os picos de corrente no quadro geral, cumpre os objetivos de autonomia dos utilizadores e aumenta a energia servida face a alternativas não otimizadas, reduzindo simultaneamente a necessidade de reforços da infraestrutura. Complementarmente, discute-se a compatibilidade regulatória e delineiam-se instrumentos práticos de engenharia (tabelas de projeto, mapas de decisão e roteiros de aceitação) que facilitam a adoção por projetistas, decisores e operadores de rede.

Em síntese, a tese demonstra que a integração em escala de VE em condomínios pode ser alcançada sem investimentos estruturais elevados, combinando otimização centralizada, dimensionamento realista e alinhamento regulatório, ou seja, uma via pragmática para acelerar a mobilidade elétrica no parque habitacional existente.

Abstract

Although electric vehicles (EVs) emerged alongside vehicles powered by fossil fuels, their adoption remained marginal for much of the twentieth century. During that period, the built environment and urban lifestyles evolved without accounting for EV charging requirements. The widespread shift to multi-dwelling buildings with shared garages and parking areas has led to electrical installations sized for lighting and common services, with contracting and metering geared toward conventional household consumption rather than e-mobility loads.

With the recent expansion of EVs, technical constraints (main-feeder capacity, protection selectivity and coordination), economic issues (tariffs and cost allocation), and regulatory gaps (rules for metering and operation in condominiums) now hinder rapid and efficient integration. This thesis addresses these limitations in an integrated manner and proposes a technically robust and economically proportionate solution for existing buildings.

Specifically, it introduces: (i) a centralised charging topology from the main feeder with dynamic management of available power; (ii) a near-real-time optimiser that guarantees the user's required energy by the declared departure time while respecting voltage/current limits and a safety reserve for non-EV loads; (iii) a simultaneity-coefficient framework tailored to condominium fleets, providing a basis for conductor and protection sizing without defaulting to the extreme case C_f=1; and (iv) a degradation-aware scheduling layer that internalises proxies for electrochemical ageing and thermal constraints, preserving both mobility and battery longevity.

Results on real data from a multi-dwelling building demonstrate that the proposed approach contains main-feeder current peaks, meets users' autonomy targets, and increases delivered energy relative to non-optimised alternatives, while simultaneously reducing the need for infrastructure reinforcement. In addition, the thesis examines regulatory compatibility and provides practical instruments, design tables, decision maps, and acceptance roadmaps to support adoption by designers, condominium administrators, and network operators.

In summary, the thesis demonstrates that large-scale EV integration in condominiums can be achieved without major structural investment by combining centralised optimisation, realistic sizing, and regulatory alignment, a pragmatic pathway to accelerate e-mobility across the existing housing stock.

Acknowledgments

I would like to dedicate this thesis to everyone who has stood by my side throughout this journey over the past five years.

Without their support, I would not have been able to see this through. Their guidance, motivation, and encouragement kept me going during challenging times.

To my advisors, **Professor José Rui da Rocha Ferreira** and **Professor Rui Manuel Esteves Araújo**, I am deeply grateful for their invaluable knowledge, expertise, and patience in guiding me to the finish line.

To my mother, **Helena**, and my late father, **João**, thank you for instilling in me a curious mind, a strong work ethic, and a drive for academic success.

To my wife, **Francisca**, I am deeply grateful for her unwavering support, for always being by my side, and for keeping me motivated with her words and actions.

To my sister, **Bárbara**, and my brother, **João**, thank you for being outstanding role models in your professional lives, examples I have always aspired to follow.

To my uncle, **Paulo**, I extend my heartfelt gratitude for his generosity in funding this endeavour when I was unemployed, allowing me to focus entirely on my studies.

To my friend, **Tomás,** thank you for being by my side and always pushing me forward, your resilience to adversity has always been inspiring.

To my colleagues at **INESC TEC**, thank you for providing an environment that allowed me to pursue my research and grow as a researcher.

Finally, to the rest of my family, friends, and colleagues, thank you for keeping me sane, for reminding me that there is more to life than work, and for constantly inspiring me to improve myself, even when you had to scold me when necessary.

To all of you, thank you.

"No one, who achieves success, does so without acknowledging the help of others. The wise and confident acknowledge this help with gratitude."
\- commonly attributed to ***Alfred North Whitehead*** -

Table of Contents

Resumo v

Abstract vii

Acknowledgments ix

Table of Contents xi

List of Figures xv

List of Tables xix

Abbreviations and Symbols xxi

Chapter 1 1

Introduction 1
- 1.1 - Background and Motivation 1
- 1.2 - Problem Statement 2
- 1.3 - Research Questions and Objectives 3
  - Primary Research Question 3
  - Secondary Research Questions 3
  - Research Objectives 3
  - Research Hypothesis 3
- 1.4 - Methodology Overview, Scopes and Limitations 4
- 1.5 - Contributions of the Work 4
- 1.6 - Thesis Structure 6

Chapter 2 7

Literature Review 7
- 2.1 - Review Scope and Methodology 7
  - Scope and Objectives 7
  - Search Strategy and Source Selection 8
  - Conceptual Framework 8
- 2.2 - Electric Vehicle Market and Residential Charging Ecosystem 8
  - Global EV Adoption Trajectories 8
  - Residential Charging Dominance and the MUD Challenge 9
  - Technology Ecosystem: Hardware, Software, and Standards 10
  - Market Actors and Value Chain 11
  - Policy Framework: European Union and Portugal 12
  - Section Synthesis 12
- 2.3 - Portuguese Low-Voltage Network Architecture and Regulatory Framework 13
  - Distribution Network topology 13
  - Regulatory Framework: RTIEBT and Electrical Installation Codes 15
  - Condominium Law and Shared infrastructure 16
  - Grid Connection Procedures 17
  - Implications for EV Charging Infrastructure Design 17
- 2.4 - Electric Vehicle Charging Behaviour and User Needs 18
  - Temporal Charging Patterns 18
  - Location Preferences 19
  - State of Charge Behaviour and Daily Energy Needs 20
  - Charging Flexibility 20
  - Behavioural Parameters for Simulation 21
- 2.5 - Electric Vehicle Charging Optimisation: A Critical Review 21

Taxonomy of Optimisation Approaches 22
Cost and Emission Minimisation Approaches 22
Other Optimisation Approaches 22
Dynamic Load Management for Multi-unit Dwellings 23
Coincidence Factors in Infrastructure Sizing 23
Fairness and Equity Considerations 24
Section Synthesis 24
2.6 - Battery Degradation: State of the Art and Scheduling Implications 24
Degradation Mechanisms 24
Empirical Degradation Models 25
Field Evidence on Charging and Degradation 25
Degradation-Aware Scheduling Approaches 26
Section Synthesis 26
2.7 - Research Gap Synthesis 26
Consolidated Research Gap 26
Mapping Gaps to Research Questions 27
Contribution Positioning 28
2.8 - Chapter Summary 28

Chapter 3 31

Coincidence Factor Analysis 31
3.1 - Contextualisation 31
3.2 - Methodology 32
3.3 - Data Generation 34
3.4 - Simulation Results 35
Optimisation Minimisation Problem 35
Optimisation – Maximisation Problem 40
3.5 - Electrical Feeder's Analysis 44
3.6 - Circuit Breakers and Fuses Analysis 48
3.7 - Results Discussion 49
3.8 - Validation and Comparative analysis of Results 51
3.9 - Chapter Summary 56

Chapter 4 59

Charging Optimisation 59
4.1 - Condominium 59
4.2 - Research Questions and Hypotheses 60
4.3 - Mathematical Formulation 61
4.4 - Simulation Principles 66
4.5 - Simulation Results 69
Optimised Charging Switchboard 71
Non-Optimised Garage Switchboard 73
Non-Optimised Charging Switchboard 74
4.6 - Results Discussion 76
4.7 - Chapter Summary 78

Chapter 5 81

Battery Degradation Considerations 81
5.1 - Contextualisation 81
5.2 - Battery Degradation Framework 81
Battery Pack Cycling Degradation Model 81
Battery Pack Resistance Model 82
Battery Pack Temperature Estimation Model 84
Sensitivity analysis of variables 88
5.3 - Scheduling Methodology 88
5.4 - Results of Scheduling Methodology on Battery Degradation 94
Scenario I – Optimisation of Charging Current without Scheduling 95
Scenario II – Optimisation of Impact on SoH with Scheduling 97

5.5 - Chapter Summary 100

Chapter 6 103

Conclusions 103
6.1 - Overview and Principal Findings 103
Central Research Question 103
Hypothesis Evaluation 104
Derived Research Question 104
Operational Methodology and Algorithm Development 106
Infrastructure Sizing Framework 106
Practical Implementation Artifacts 107
Battery Health Integration 107
6.2 - Contribution Positioning and Significance 107
6.3 - Limitations and Practical Considerations 108
Empirical Validation Scope 108
Methodological Assumptions and Their Implications 109
6.4 - Future Research Directions 110
Deployment, Validation, and Regulatory Integration Track 110
Scientific Validation and Model Refinement Track 111
Technical Maturity and Ecosystem Integration Track 112
Cross-Cutting Research Elements 112
6.5 - Broader Impact and Closing Remarks 112

References 115

Annex A 123
Overview 123
A1. Fuzzy Inference Model 123

Annex B 125
Overview 125
B.1 Optimisation Module 125
B.2 Battery Model 137
B.3 Grid Model 149
B.4 Optimised Charging Simulation 161
Reference and Technical Notes 175
Software Dependencies 175
Repository 175

List of Figures

Figure 1. Portuguese Electric Vehicle Market Performance in European Context. 9

FIGURE 2. Low Voltage Grid typical topology. 14

FIGURE 3. Low Voltage Grid typical topology inside a Private Condominium 14

FIGURE 4. Low Voltage Grid typical topology in the case of a single building 15

Figure 5. "Last journey of the day?" survey responses [56]. 18

Figure 6. User charging location preference according to *Plug in America* 2022 survey report, among EV owners with a single dominant charging location (83% of respondents). 19

FIGURE 7. Complete methodology overview from data generation to $C_{f\text{-}fuzzy}$ estimation, including optimal charging simulations and result generation. 35

FIGURE 8. Cumulative frequency of the allocated current with 99% threshold. 38

FIGURE 9. Scatter plot of C_{f1} according to simulation results. 39

FIGURE 10. Scatter plot of C_{f2} according to simulation results. 40

FIGURE 11. Scatter plot of C_{f3} according to simulation results. 41

FIGURE 12. Scatter plot of C_{f4} according to simulation results. 42

FIGURE 13. Scatter plot of all the calculated C_f according to simulation results. 43

FIGURE 14. Scatter plot of Sc3 with power function trendline. 43

FIGURE 15. Coincidence factor graph with coincidence factor plotted and corrected via the fuzzy logic inference model. 44

FIGURE 16. Feeder boundaries per fleet size and maximum allocated current with C_{f1}. 47

FIGURE 17. Feeder boundaries per EV fleet size, maximum allocated current and C_{f1}. 47

FIGURE 18. Simultaneity factor correction via fuzzy logic inference model, for a fleet of 20 EVs, with high optimisation and low demand. 51

FIGURE 19. Coincidence factor versus fleet size and uniform versus exponential distribution for start times between 20h00 and 08h00. 53

FIGURE 20. Comparative of current study Coincidence Factors with literature Coincidence Factors. 54

FIGURE 21. Condominium 1 electrical infrastructure topography. 59

FIGURE 22. Plot of the second derivative of f(x) evaluated between x=6 and x=16, demonstrating its continuity in the required interval of solutions. 64

FIGURE 23. Charge curve of a Lithium-ion cell, SoC vs. Charge Time. 65

FIGURE 24. Flowchart of charging process for each EV in the building's fleet. 67

FIGURE 25. Average EV battery maximum energy capacity in kWh throughout the years [71]. 68

FIGURE 26. Percentage of vehicles commuting during the day [65]. 68

FIGURE 27. Plot of available current (unused by the tenants) and used current (for charging EVs) in phase 1 through a 7-day period. 70

FIGURE 28. Plot of available current (unused by the tenants) and used current (for charging EVs) in phase 2 through a 7-day period. 70

FIGURE 29. Plot of available current (unused by the tenants) and used current (for charging EVs) in phase 3 through a 7-day period. 71

FIGURE 30. Example 1 of EV SoC evolution throughout the 7-day period. 71

FIGURE 31. Example 2 of EV SoC evolution throughout the 7-day period. 72

FIGURE 32. Charging current allocated to EV in example 1. 72

FIGURE 33. Charging current allocated to EV in example 2. 73

FIGURE 34. Phase 1 plot of available current in the Garage Switchboard and the used current. 73

FIGURE 35. Example 3 of EV that was connected to the charging system via the Garage Switchboard. 74

FIGURE 36. Example 4 of EV that was connected to the charging system via the Garage Switchboard. 74

FIGURE 37. Example 5 of EV charging using a non-optimised algorithm. 75

FIGURE 38. Example 6 of EV charging using a non-optimised algorithm. 75

FIGURE 39. Example 7 of EV charging using a non-optimised algorithm. 76

FIGURE 40. Example 8 of EV charging using a non-optimised algorithm. 76

FIGURE 41. Visualisation of the evolution of supplied energy, and km travelled, according to the safety coefficient. 78

FIGURE 42. Electrical model analogue representation of the EV battery thermal model. 85

FIGURE 43. Evolution of voltage at capacitor terminals, analogue to evolution of temperature within EV battery. 86

FIGURE 44. (a) Plot of Teq in function of Tambient and IDC considering a 0.5 ohms Rpack, $4m^2$ battery surface area and a heat transfer coefficient of 5 $W/m^2.K$. (b) Plot of Teq in function of Tambient and IDC considering a 0.5 ohms Rpack, $4m^2$ battery surface area and a heat transfer coefficient of 20 $W/m^2.K$. 87

FIGURE 45. Effect of key parameters on battery equilibrium temperature (T_{eq}). The plot shows how Teq varies with (a) internal resistance (R_{pack}), convection coefficient (h), and (b) cooling surface area (A), and charging current (I_{DC}), considering a 21°C ambient temperature. 88

FIGURE 46. Scheduling of 1 EV using optimiser with per-charge-day penalty to avoid front or back lumping. 92

FIGURE 47. Scheduling of 1 EV using optimiser with per-charge-day penalty to avoid front or back lumping and smoothing term to avoid erratic charging. 93

FIGURE 48. Scheduling for 1 EV using optimiser with per-charge-day penalty term, smoothing term, and minimal gap day and target SoC constraints. 94

FIGURE 49. Degradation value (ξ_{cyc_i}) per charging event during an 8.3h charging period in function of charging current, SoC, and battery temperature, in scenario I. 96

FIGURE 50. Degradation value (ξcyc_i) per charging event, in scenario IV, considering an optimised schedule. Notice the +2.1e1 scaling on Temperature for 21ºC. 99

FIGURE 51. Degradation value (ξcyc_i) per charging event, in scenario IV, not considering an optimised schedule. Notice the +2.1e1 scaling on Temperature for 21ºC. 100

FIGURE 52. Coincidence Factor Correction membership function, output. 123

FIGURE 53. Demand Level membership function, input. 124

FIGURE 54. Optimisation Level membership function, input 124

List of Tables

TABLE I List of peer-reviewed publications resulting from the research conducted in this thesis........ 5

TABLE II Summary of key Standards and their relevance to MUD Charging........ 11

TABLE III Standardised Power Levels for PTs in the Portuguese MV/LV Grid (E-Redes, 2021). .. 13

TABLE IV Behavioural Parameter Summary for Simulation Design........ 21

TABLE V Coincidence factor according to Portuguese regulations [84]........ 23

TABLE VI Demand Factor according to the North American regulation [85]........ 23

TABLE VII Gap-to-contribution mapping........ 28

TABLE VIII Analysis of previous works on coincidence factor estimation for EV charging 32

TABLE IX Charging current requirements by EV fleet size, to charge at maximum charging rate. 33

TABLE X Charging fleet size versus absolute fleet size comparison chart, under minimisation problem. 36

TABLE XI Coincidence factors for calculating maximum charging fleet size........ 36

TABLE XII Maximum allocated charging current for each EV fleet size........ 37

TABLE XIII Frequency distribution of allocated charging current across intervals........ 37

TABLE XIV S_c for current calculation per fleet size considering minimisation problem........ 38

TABLE XV Charging fleet size versus absolute fleet size comparison chart, under max. problem. 40

TABLE XVI Coincidence factor for current calculations per fleet size, considering max. problem. 41

TABLE XVII R-squared values of $C_{f\text{-}avg}$ against remaining C_f values and curves. 42

TABLE XVIII Maximum allowed current by feeder conductor section and material, isolation material and installation type [84]........ 45

TABLE XIX Feeder phase section per fleet size considering no C_f........ 45

TABLE XX Feeder phase section per fleet size considering C_{f1} and C_{f2}. 46

TABLE XXI Feeder section per stipulated current and conventional triggering current for each circuit breaker [84]. 48

TABLE XXII (PART I) Circuit breaker phase section per fleet size considering C_{f1} and C_{f2}........ 48

TABLE XXIII Feeder section and circuit breaker stipulated current per EV fleet size considering S_{c1} and S_{c2}. 50

TABLE XXIV Dataset used to estimate a baseline for the Coincidence Factors based on real data. 51

TABLE XXV Comparison between simulated coincidence factors and baseline coincidence factors. 53

TABLE XXVI Inputs for charging/discharging simulations. 68

TABLE XXVII Methodology for calculating the next Desired SoE. 69

TABLE XXVIII Summary of charging methodologies simulations outcomes over a 7-day period. 77

TABLE XXIX Relation between battery SoC and its internal resistance. 83

TABLE XXX Battery pack architecture based on cell type. 84

TABLE XXXI Summary of the impact of the different variables in Teq. 88

TABLE XXXII Indices and Parameters for Scheduling Optimisation Problem 90

TABLE XXXIII Decision Variables for Scheduling Optimisation Problem 91

TABLE XXXIV Scenario I - Impact on SoH Summary 95

TABLE XXXV Manufacturer recommendations for battery charging according to electric vehicle user's manual. 97

TABLE XXXVI Scheduling Algorithm Output for one EV. 98

TABLE XXXVII Impact on SoH – Scenario IV summary with schedule. 99

TABLE XXXVIII Impact on SoH – Scenario IV summary without schedule. 99

TABLE XXXIX Fuzzy Logic Inference Model Rules 123

Abbreviations and Symbols

List of abbreviations

AC	Alternating Current
ACAP	Portugal's Automobile Association
ACEA	European Automobile Manufacturer's Association
AFIR	Alternative Fuels Infrastructure Regulation
Abs	Absolute
Avg	Average
BMS	Battery Management System
BES	Battery Energy System
BEV	Battery Electric Vehicle
CAPEX	Capital Expenditure
CCS	Combined Charging System
CPMS	Charge Point Management System
CPO	Charge Point Operator
DC	Direct Current
DLM	Dynamic Load Management
DoD	Depth of Discharge
DSO	Distribution System Operator
EPBD	Energy Performance of Buildings Directive
ESCO	Energy Service Company
e-MSP	e-Mobility Service Provider
EU	European Union
EV	Electric Vehicle
EVSE	Electric Vehicle Supply Equipment
Ibid	Ibidem, used when citing the same reference as before
IEA	International Energy Agency
INE	National Estatistics Institute (Portugal)
ISO	International Organisation for Standardization
LoL	Loss-of-Life
LV	Low Voltage
Max	Maximum
MID	Measuring Instruments Directive
Min	Minimum
MUD	Multi-Unit Dwelling
MV	Medium Voltage
NLP	Non-linear Programming
OCPI	Open Charge Point Interface
OCPP	Open Charge Point Protocol
OCV	Open Circuti Voltage
OEM	Original Equipment Manufacturer
PHEV	Plug-in Hybrid Electric Vehicle
PT	Power Transformer
PV	Photovoltaic

SLSQP	Sequential Least Square Programming
SoC	State of Charge
SoE	State of Energy
SoH	State of Health
UK	United Kingdom
V1G	Vehicle-to-Grid (unidirectional)
V2G	Vehicle-to-Grid
V2H	Vehicle-to-Home
V2X	Vehicle to X

List of symbols

A	Cooling surface area of battery pack [m^2]
C_f	Coincidence Factor
$C_{f\text{-}fuzzy}$	Coincidence Factor after fuzzy processing
C_p	Specific Heat Capacity [J/kg.K]
D_f	Demand Factor
f(x)	Objective Function
h	Convection heat transfer coefficient [$W/m^2 \cdot K$]
I_2	Conventional non-tripping current of circuit breaker [A]
I_{AC}	Measured/Charging Current AC [A]
I_B	Maximum measured current per phase for the EV fleet [A]
$I_{Charging}$	Charging current [A]
I_{DC}	Measured/Charging current DC [A]
I_{max}	Feeder maximum allowed current [A]
I_n	Rated current of circuit breaker [A]
I_z	Maximum allowable current for each feeder phase [A]
m	Battery mass [kg]
n	EV fleet size
N	Overall EV fleet size
N_{EVs}	Number of Electric Vehicles
$N_{Series/Parallel}$	Battery Cell Configuration
$P_{available}$	Available Power [kW]
Q_{gen}	Generated Heat [J]
Q_{diss}	Dissipated Heat [J]
Q_{net}	Accumulated Heat [J]
Q_{proc}	Processed Charge (throughput) [Ah]
R_{pack}	Internal resistance of the battery pack [Ω]
R_{Th}	Thermal Resistance [K/W]
$T_{ambient}$	Ambient temperature [°C]
$T_{available}$	Time available [min]
$T_{Battery}$	Battery temperature [ºC]
T_{eq}	Battery equilibrium temperature [°C]
$U_{Charging}$	Charging voltage [V]
ξ_{cyc_i}	Cycle-degradation increment per charging event
ρ_{safety}	Safety coefficient [-]
τ (tau)	Time constant [s]

All abbreviations, acronyms and symbols are defined at first occurrence in the text.

Chapter 1

Introduction

1.1 - Background and Motivation

This work intends to present an optimised electric-vehicle (EV) charging management system for existing condominiums, capable of dealing with EV massification, using the available power, in real-time, in the main feeder of the building, based on optimal charging strategies.

The main goal will be the development of an intelligent charging methodology, incorporating advanced power allocation mechanisms that maximise driver/owner satisfaction and the efficiency of the electrical installation, seeking to increase the use of renewable sources and energy storage to minimise changes in the existing infrastructure and/or reinforcements of the power grid.

Current Distribution Grids were not planned to accommodate the present rise in sales of EVs that require charging. The growing number of these types of vehicles will have negative impacts on current distribution grids if pre-emptive measures are not implemented [1].

In a report from 2021 [2] McKinsey states that the sales of EVs could surpass those of more than 130 million cars worldwide in the next 5 years, and this could mean an estimated investment of between $110 billion and $180 billion in charging stations, both public and residential, to meet the demand. Recent evidence indicates the market has continued to scale, global electric-car sales are expected to exceed 20 million in 2025, public charging points have more than doubled since 2022 to surpass 5 million, and in the Stated Policies Scenario, around 150 million additional charging points are installed between 2025 and 2030, with approximately two-thirds of these at homes. Projections in the IEA's (International Energy Agency) 2024–2025 outlook also indicate that public charging points will surpass 15 million by 2030, underscoring the ongoing need for substantial investment alongside the growth in vehicle sales.

According to the same report, the power demand, in terawatt-hour (TWh), could rise from 14-16 in 2022 to 130-195 in 2030, if we consider just the EU-27 (European Union) plus the United Kingdom.

The authors of [3] set out to answer the following question: How much electric infrastructure do Electric Vehicles Need? – as stated in the paper title, they concluded (i) that home charging is currently the most important charging option in the observed countries; (ii) public slow charging is a "must have" and will remain important mainly in metropolitan areas and regular work charging options are also important; (iii) Direct Current (DC) high power charging importance will differ from country to country depending on a countries highway coverage and if long commutes are a common need; and finally (iv) charging infrastructure deployment in different countries should be discussed in a larger context rather than being based on the vehicle-to-refuelling index as sole measure.

This shift, caused by the electrification of the transportation sector, such as private means of transportation (e.g., cars), must be planned carefully and accounted for. Having this in mind, some questions must be answered promptly, such as (i) is there enough power generation capability to absorb this shift? (ii) Will the shift be powered by renewable energy, or does it have to rely on fossil fuels? (iii) Are the transportation and distribution energy grids prepared, or will we face a bottleneck? (iv) Will we witness a change in several levels, including energy efficiency?

These problems were already present back in 2010 and were addressed by the authors of [4]. In this paper, the authors present a conceptual framework to successfully integrate EVs into electric power systems. To combat this, advanced centralised EV charging strategies are proposed as a countermeasure to avoid the need for hefty grid reinforcements.

As they conclude, and as has been visible in the years that followed, the large deployment of EVs has imposed and continues to impose tremendous challenges to the operation and management of electric power systems.

In another report [5] it is stated that domestic charging of EVs, if unmanaged, could lead to substations' peak-load increases that would, eventually, push local transformers beyond their capacity and if, as mentioned before, no countermeasures are taken, the cumulative grid investments could exceed several hundred euros per EV.

Taking all this into account, it becomes evident that there is a need for charging solutions that prevent unnecessary, costly investments in the distribution grid and unnecessary ramp-ups in energy production, as this would defeat the sustainability improvements of shifting to EVs in the first place. The need arises to create a system that can optimise the available energy and the existing distribution grid infrastructure, as well as the available electrical infrastructure in condominiums, to minimise the environmental impact of this type of EV and maximise the eco-friendly profits that can be drawn from this technology.

1.2 - Problem Statement

The central problem addressed by this thesis is the integration of large numbers of EVs into existing buildings, mainly multi-dwelling residential buildings, but also office buildings and other facilities with shared parking, whose electrical infrastructure was not designed to accommodate such loads.

Current distribution grids were not designed to support the current rise in EV sales, and the growing number of these vehicles will have a negative impact on distribution infrastructure if pre-emptive measures are not implemented.

Domestic EV charging, if unmanaged, could lead to substation peak-load increases that would eventually push local transformers beyond their capacity. Without countermeasures, cumulative grid investments could exceed several hundred euros per EV. The challenge is compounded in condominiums where multiple stakeholders share common infrastructure, creating governance, billing, and technical coordination complexities that are not present in single-family dwellings. Considering these problems a set of research questions, presented in the next subchapter, was thought out with the goal of giving concrete answers.

1.3 - Research Questions and Objectives

Primary Research Question

What is the most cost-effective approach to enable high numbers of EV owners to charge their vehicles in existing multi-dwelling buildings without requiring substantial infrastructure reinforcement?

This primary research questions was broken down into several, smaller, questions.

Secondary Research Questions

RQ1: How can the idle power capacity of residential buildings be leveraged to charge EVs during low-demand periods?

RQ2: What coincidence factors are appropriate for EV fleet charging in condominiums under optimised charging regimes?

RQ3: How can charging schedules be designed to minimise battery degradation while meeting user mobility requirements?

RQ4: What practical design guidelines can facilitate adoption by designers, condominium administrators, and network operators?

Research Objectives

To give answer to these previous questions, this thesis pursues the following specific objectives:

O1: Develop a centralised charging topology with dynamic management of available power from the main feeder.

O2: Design a near-real-time optimiser that guarantees user energy requirements while respecting infrastructure constraints.

O3: Establish a coincidence factor framework specific to condominium EV fleets for conductor and protection sizing.

O4: Create a degradation-aware scheduling layer that preserves both mobility and battery longevity.

O5: Validate the proposed approach using real data from a multi-dwelling building in Porto, Portugal.

Research Hypothesis

HYPOTHESIS: It is possible to use the idle power capacity of a residential building to charge high numbers of EVs during low-demand hours by implementing minor electrical infrastructure changes and applying a centralised, optimised charging methodology.

1.4 - Methodology Overview, Scopes and Limitations

Overview:

This research employs a simulation-based methodology combined with validation using observed real-world data. The approach consists of four interconnected phases:

Phase 1 – Data Collection: Six months of power consumption measurements from a multi-dwelling building in Porto, Portugal, collected at 3-minute intervals.

Phase 2 – Algorithm Development: Design and implementation of a non-linear optimisation algorithm using Sequential Least-Squares Programming (SLSQP) for real-time current allocation.

Phase 3 – Simulation: Extensive simulations to establish coincidence factors and validate charging methodology under various fleet sizes and demand conditions, as well as different levels of optimisation.

Phase 4 – Validation: Comparison of results against existing literature, regulatory frameworks, and baseline scenarios.

The research focuses on the Portuguese regulatory context, which serves as a representative case study within the broader European Union framework. The methodology and findings are designed to be transferable to other jurisdictions with similar low-voltage network architectures.

Scope:

Geographic focus: Portuguese low-voltage network regulations and infrastructure.

Building type: Existing multi-dwelling residential buildings with shared parking facilities.

Charging mode: AC Mode 3 slow charging (6-16A single-phase, 16-32A three-phase).

Fleet sizes: Up to 70 EVs per condominium.

Limitations:

DC fast charging is not addressed in the primary analysis. This type of charging would require significant infrastructure reforms at the building level, which contradicts the primary research question.

Vehicle-to-Grid (V2G) bidirectional flows are discussed but not implemented in the optimisation, as EV manufacturers still limit this capability.

The study does not directly target new construction scenarios where infrastructure can be designed ab initio, but the coincidence factors provided are suitable for both existing and new construction.

Economic optimisation (tariff-based scheduling) is not the primary focus of the study, and it is not considered.

1.5 - Contributions of the Work

This thesis makes the following original contributions to the field of EV integration in, not only, but focused on residential buildings:

Theoretical Contributions:

A coincidence factor framework specifically tailored for optimised EV charging in condominium settings, providing a basis for conductor and protection sizing without defaulting to the conservative case of C_f=1.

A fuzzy logic inference model for adjusting coincidence factors, during the planning phase, based on optimisation level and demand characteristics.

Methodological Contributions:

A near-real-time charging optimisation algorithm that minimises the deviation between required charging time and available time while respecting infrastructure constraints and user needs.

A degradation-aware scheduling methodology that internalises battery ageing and thermal constraints into the charging decision process, that occurs within the domestic charging sessions.

A battery thermal model integrated with the charging optimisation framework.

Practical Contributions:

Design tables for feeder sizing and circuit breaker selection under various fleet sizes and coincidence factors.

Decision maps linking optimisation levels, demand characteristics, and appropriate coincidence factors, and implementation guidelines for condominium administrators and electrical designers.

Furthermore, this work has produced three distinct journal publications and one conference paper, as seen in TABLE I.

TABLE I
List of peer-reviewed publications resulting from the research conducted in this thesis.

Publication Type	Publication Venue	Name	Year	Chapter	DOI
Journal	IEEE Access	Data-Driven Charging Strategies to Mitigate EV Battery Degradation	2025	Chapter 5	10.1109/ACCESS.2025.3638156
Journal	MDPI Energies	Fuzzy Logic Estimation of Coincidence Factors for EV Fleet Charging Infrastructure Planning in Residential Buildings	2025	Chapter 3	10.3390/en18174679
Journal	IEEE Access	Electric Vehicle Charging Method for Existing Residential Condominiums	2024	**Error! Not a valid result for table.**	10.1109/ACCESS.2024.3491379
Conference	IEEE VPCC	Optimisation Algorithm for the Charging Management of Electric Vehicles in Multi-Dwelling Residential Buildings	2023	**Error! Not a valid result for table.**	10.1109/vppc60535.2023.10403268

1.6 - Thesis Structure

The thesis is organised into six chapter that flow a logical progression from the problem contextualisation through solution development to validation and conclusions:

Chapter 2 (Literature Review) establishes the theoretical foundation across five strands: residential charging market dynamics, Portuguese low-voltage network architecture and regulations, observed charging habits and preferences, existing EV charging optimisation approaches, and battery degradation fundamentals. This chapter identifies the research gaps that motivate the subsequent technical work.

Chapter 3 (Charging Optimisation) presents the core technical contribution: a centralised charging topology and near-real-time optimisation algorithm. The chapter details the mathematical formulation, simulation methodology, and comparative results against baseline scenarios, demonstrating the feasibility of the proposed approach.

Chapter 4 (Coincidence Factor Analysis) extends the analysis to infrastructure dimensioning by developing coincidence factors appropriate for optimised EV charging. The chapter presents simulation results, derives analytical expressions, validates against literature values, and provides practical design tables for feeder and protection sizing.

Chapter 5 (Battery Degradation Considerations) integrates battery health into the charging framework through cycling degradation models, thermal estimation, and a scheduling methodology that balances user convenience with battery longevity.

Chapter 6 (Conclusions) synthesises the main outcomes against the research objectives, discusses external validity and practical implications, and delineates directions for future research.

Chapter 2

Literature Review

This chapter presents a systematic review of the literature that substantiates the research undertaken in this thesis. The review establishes the state of the art across five interconnected domains: the electric vehicle market and residential charging ecosystem; the Portuguese low-voltage network architecture and regulatory framework; electric vehicle charging behaviour and user needs; charging optimisation approaches; and battery degradation considerations. The chapter concludes with a synthesis of research gaps that motivate the contributions presented in subsequent chapters.

The structure of this chapter reflects the multidisciplinary nature of the problem. Understanding the feasibility and design of EV charging solutions for Multi-Unit Dwellings (MUDs) requires simultaneous consideration of market dynamics driving adoption, technical constraints imposed by existing electrical infrastructure, behavioural patterns that determine charging flexibility, algorithmic approaches to optimisation, and asset management considerations that affect long-term viability. Each section builds upon the previous, culminating in a clear identification of the research gaps that this thesis addresses.

2.1 - Review Scope and Methodology

Scope and Objectives

The primary objective of this literature review is to establish the current state of knowledge regarding residential EV charging, with particular emphasis on the challenges facing multi-unit dwellings. The review aims to: (i) characterise the market context and adoption trajectories that create demand for residential charging solutions; (ii) document the technical and regulatory constraints specific to Portuguese low-voltage networks; (iii) synthesise empirical evidence on EV user charging behaviour and the flexibility it affords; (iv) critically evaluate existing optimisation approaches and identify their limitations for MUD applications; and (v) summarise battery degradation science relevant to charging strategy design.

The scope is explicitly bounded to residential charging in multi-unit buildings, with a focus on the Portuguese context, where regulatory and infrastructure characteristics present specific challenges. Public charging infrastructure, heavy-duty vehicle charging, and vehicle-to-grid services are discussed only insofar as they provide context for the residential charging problem. The review prioritises literature from 2018 onwards to capture the rapid evolution of the EV market, while including seminal earlier works that established foundational concepts.

Search Strategy and Source Selection

The literature search employed multiple academic databases, including Scopus, Web of Science, and IEEE Xplore, supplemented by Google Scholar for broader coverage. Search terms were constructed around key concepts: "electric vehicle charging", "residential charging", "multi-unit dwelling" OR "apartment" OR "condominium", "load management", "optimisation", "battery degradation", and "coincidence factor" OR "simultaneity factor". Boolean combinations were tailored to each thematic strand.

Inclusion criteria prioritised peer-reviewed journal articles and conference proceedings, supplemented by authoritative industry reports from organisations such as the International Energy Agency (IEA), Bloomberg NEF, and the European Automobile Manufacturers' Association (ACEA). Portuguese regulatory documents, standards specifications (OCPP, ISO 15118, OCPI), and EU directives were included as primary sources for regulatory and technical requirements. Sources were excluded if they predated 2015 (except for foundational works), were not available in English or Portuguese, or lacked methodological transparency.

Conceptual Framework

The literature is organised according to a conceptual framework that recognises the interdependence between market drivers, technical constraints, user behaviour, and solution design. Market dynamics and policy frameworks create the demand for residential charging and shape the regulatory environment within which solutions must operate. Technical constraints imposed by electrical infrastructure and national codes bound the feasible solution space. User behaviour determines both the challenge (temporal clustering of demand) and the opportunity (flexibility from long dwell times). Optimisation approaches represent the algorithmic response to these constraints and opportunities, while battery degradation considerations introduce long-term asset management into the design calculus.

This framework guides the structure of the subsequent sections and informs the synthesis of the research gap that concludes the chapter. Importantly, it positions the MUD charging problem as one that requires integrated consideration of multiple domains, rather than isolated technical optimisation.

2.2 - Electric Vehicle Market and Residential Charging Ecosystem

Global EV Adoption Trajectories

Electric-vehicle uptake has accelerated dramatically over the past five years, transforming private premises, especially homes, into the dominant charging venues and making residential infrastructure central to EV system integration [6], [7]. Between 2019 and 2023, global plug-in vehicle sales expanded from roughly 2–3 million to approximately 14 million units annually, with public charging stock growing rapidly and DC fast charging outpacing AC in relative terms [6], [8]. By 2024–2025, EVs exceeded one-fifth of global new-car sales, underpinned by resilient growth in China and more heterogeneous dynamics across Europe and North America as incentives, supply chains, and product mixes evolved [7], [9].

Looking to 2030–2035, stated-policies scenarios converge on EVs capturing around 40–45% of global light-duty sales by 2030, with subsequent trajectories contingent on technology costs, policy ambition, and the pace of charging deployment [7], [9]. Across these scenarios, home charging remains the backbone of energy delivery through the 2030s; however, the share of energy delivered away from home rises with urbanisation and fleet size, amplifying the importance of MUD-ready infrastructure, Dynamic Load Management (DLM), MID-grade metering, and interoperable billing frameworks [6], [10], [11].

Portugal has emerged as a frontrunner in European EV adoption, with battery electric vehicles (BEVs) capturing a record-breaking 22.5% market share in January 2025, representing a 40.97% increase compared to January 2024 [12]. This performance significantly exceeds the EU average, positioning Portugal ahead of larger markets including Germany (20.2%), France (25.7%), Spain (11.5%), and Italy (7.6%) [13]. According to data from the *Associação Automóvel de Portugal* (ACAP) and the European Alternative Fuels Observatory, Portugal recorded 41,741 BEV sales in 2024, with the EV sector growing 34.9% through September 2025 to reach an 18% market share [14]. This growth trajectory has been supported by government incentives, including up to €4,000 for private BEV purchases through the Environmental Fund, full exemption from registration tax for electric vehicles, and VAT deduction allowances [15], summarised in Figure 1, where in panel (a) we see a comparison of BEV market share across major European markets in January 2025, showing Portugal's 22.5% market share exceeding the EU average and larger economies, and (b) the evolution of Portugal's BEV market share from 2020 to January 2025, demonstrating accelerating adoption driven by government incentives and policy support.

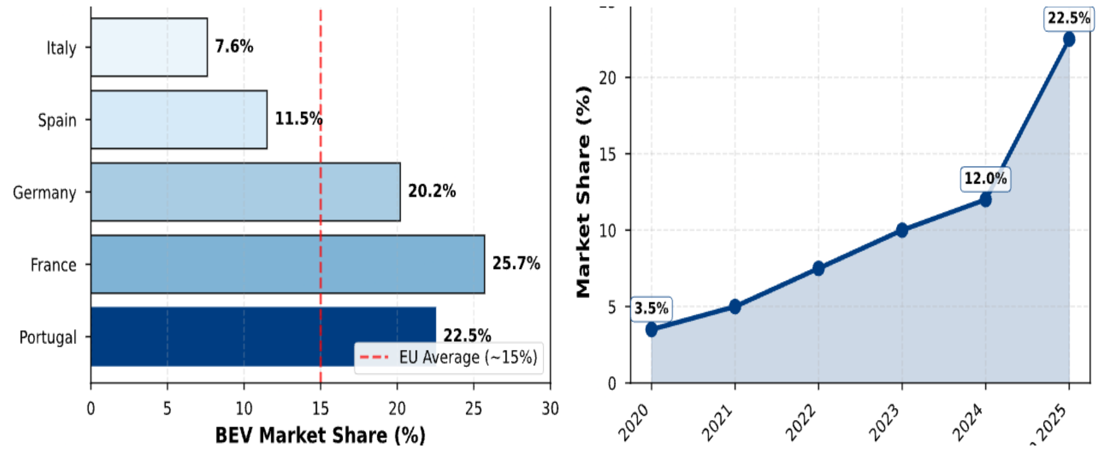


Figure 1. Portuguese Electric Vehicle Market Performance in European Context.

Residential Charging Dominance and the MUD Challenge

Empirical data indicate that most EV charging sessions occur at private locations, primarily the home, with public and workplace charging playing a complementary role that grows with urban density and limited private parking [6], [9]. China's lower home-parking access has driven a larger public network build-out, while Europe's higher home-charging access coexists with the rapid expansion of public infrastructure under EU regulation [6], [10].

While single-family houses typically enable dedicated overnight AC charging with modest coordination requirements, multi-unit dwellings face capacity constraints, shared-infrastructure governance, and billing complexity that necessitate more sophisticated designs [16], [8]. MUDs present a distinct optimisation problem: their deployments must reconcile limited main-service and feeder capacity, split incentives between landlords and tenants, and meet stringent requirements for access control, metering, and fair cost allocation [16], [10].

The Portuguese housing stock presents challenges for EV charging integration. According to the 2021 Census conducted by Statistics Portugal (*Instituto Nacional de Estatística*, INE), Portugal has approximately 3.59 million residential buildings containing 5.98 million dwellings [17]. The average number of dwellings per building is 1.7 nationally, but this figure rises dramatically in urban areas: Lisbon's Metropolitan area averages 3.3 dwellings per building, while Porto's Metropolitan area shows similar density patterns [18]. In the major metropolitan areas of Lisbon and Porto, with populations of approximately 2.9 million and 1.7 million, respectively, families predominantly live in apartment blocks [19]. This concentration of population in multi-unit buildings, where most urban Portuguese residents reside, underscores why MUD-specific charging solutions are crucial for achieving national EV adoption goals.

According to survey-based studies, a large proportion of vehicle owners report lacking access to home EV charging. For instance, 78% of all vehicle owners in the USA, according to one study [20], and nearly half of UK homes are reportedly unprepared to install EV chargers, according to another [21]. Note that this latter study examined single-family homes; the situation is likely more challenging for multi-dwelling residential buildings. These statistics underscore the urgency of developing scalable MUD charging solutions.

Technology Ecosystem: Hardware, Software, and Standards

In detached or semi-detached homes, the residential market is dominated by AC Mode 3 "wall box" equipment in the 3.7–22 kW range, utilising Type 2/CCS interfaces. Devices are increasingly integrating residual-current protection, MID-grade metering, and network connectivity for monitoring and control [6], [8]. Smart-charging functionality has matured to include tariff-aligned scheduling, photovoltaic (PV) surplus utilisation, and utility-signalled set-point control via Charge-Point Management systems (CPMS) [11], [22].

On the communications layer, Open Charge Point Protocol (OCPP) 1.6 remains widespread, though adoption of OCPP 2.0.1 is accelerating due to its enhanced telemetry, profiles for smart charging, device-management features, and improved security postures [11], [22]. At the vehicle interface, ISO 15118-20 offers standardised Plug-and-Charge authentication and defines options for bidirectional power flows (V2H/V2G), albeit with heterogeneous implementation across Original Equipment Manufacturers (OEMs) and charger vendors [23], [7]. Where roaming is required, OCPI provides the interconnection fabric and settlement mechanisms beyond a single property [24].

TABLE II summarises the key standards relevant to MUD charging infrastructure and their respective functions.

TABLE II
Summary of key Standards and their relevance to MUD Charging

Standard	Function	MUD Relevance
OCPP 2.0.1	Charger-to-CPMS communication, smart charging profiles, device management	Enables centralised DLM, per-socket power modulation, and remote monitoring
ISO 15118-20	Plug-and-Charge authentication, bidirectional power flow (V2X)	Simplifies user experience, enables future V2H/V2B services for building flexibility
OCPI 2.2.1	Roaming and inter-operator settlement, tariff exchange	Supports mixed-use buildings with visitor access and CPO concession models
MID (2014/32/EU)	Metering accuracy for billing purposes	Required for transparent cost allocation among MUD residents

Market Actors and Value Chain

The residential ecosystem can be understood as a layered market. Hardware OEMs supply 3.7–22 kW AC Electric Vehicle Supply Equipment (EVSE) equipped with MID-compliant metering, integrated residual-current devices, and communication roadmaps aligned with OCPP and ISO 15118 [6], [8]. Software providers deliver CPMS platforms that bundle fleet/site management, tariff engines, access control, DLM logic, billing, and integrations with building energy systems [11]. CPOs and e-mobility service providers (e-MSPs) extend turnkey build-own-operate propositions for MUDs [24]. Finally, electrical integrators and energy service companies (ESCOs) implement backbones, panel upgrades, and DLM commissioning—capabilities that are decisive for retrofit viability in existing MUDs [16].

In Portugal, the electric mobility ecosystem is anchored by several key institutions. e-Redes (formerly EDP Distribuição) serves as the principal Distribution System Operator (DSO), operating the high-voltage, medium-voltage, and low-voltage distribution systems in 278 of Portugal's 308 municipalities, accounting for 99.5% of consumers connected at low voltage [25]. By 2024, E-Redes completed the installation of 6.6 million smart meters throughout mainland Portugal, representing an investment of €330 million [26]. MOBI.E operates the national electric mobility network, providing interoperability across all charging operators and enabling any user to access any public charging point with a single access method. As of mid-2025, the MOBI.E network integrates over 7,000 charging stations and more than 13,000 charging points, with almost 40% being fast or ultra-fast chargers [27]. Private charging network operators including Galp, EDP, Atlante, and Powerdot operate approximately 74% of the public charging network [28]. Notably, MOBI.E allows condominiums and private individuals to connect their charging infrastructure to the national network by becoming Charging Point Owners (DPC), creating a pathway for MUD integration [29].

Policy Framework: European Union and Portugal

EU-level regulation exerts an indirect but substantial influence on residential charging. The Alternative Fuels Infrastructure Regulation (AFIR) [10] mandates dense minimums for publicly accessible charging and requires ad-hoc payment options; although it does not directly regulate private home sites, its interoperability requirements increasingly shape mixed-use and visitor areas within MUDs. The 2024 recast Energy Performance of Buildings Directive (EPBD) [30] reinforces "EV-readiness" by extending ducting and pre-cabling requirements in renovations and new construction—a provision particularly salient for MUD cost and scalability outcomes.

In Portugal, the regulatory framework for electric mobility underwent significant reform with the publication of Decreto-Lei n.º 93/2025 on 14 August, which establishes the new legal framework for electric mobility (*regime jurídico da mobilidade elétrica*) and aligns national legislation with the European AFIR regulation [31], [32]. This decree-law eliminates the figure of the Comercializador de Electricidade para a Mobilidade Elétrica (CEME), with charging services now provided directly by Charging Point Operators (OPC) who can acquire energy through market contracts or self-consumption arrangements [33]. The new regime introduces mandatory ad-hoc charging options at all public charging points, allowing users to charge without requiring contracts, and mandates diverse electronic payment methods including QR codes and bank cards [34]. For existing charging points with power equal to or greater than 50 kW, operators must complete necessary renovations to ensure universal access and payment diversity by 1 January 2027 [35].

For condominiums specifically, Portuguese legislation permits owners, tenants, or legal occupants to install charging points or electrical outlets in common areas, subject to compliance with technical requirements stipulated by the Directorate-General for Energy and Geology (DGEG), and the low voltage installations technical guide for the energy supply to electrical vehicles [36]. The condominium administration must be notified in writing at least 30 days before installation. The administration may only oppose installation if: (i) a shared charging point with equivalent technology already exists; (ii) there is an effective risk to the safety of persons or property; or (iii) the condominium itself intends to install shared infrastructure within 90 days [37]. Additionally, through Despacho nº. 5126/2023, of 3 May, the Government established an incentive programme covering 80% of charger purchase costs (up to €800 per station) and electrical installation costs (up to €1,000 per parking space) for condominiums connecting to the MOBI.E network [38].

Section Synthesis

Lessons from recent deployments endorse several design regularities. First, pre-cabled backbones combined with socket-level metering in MUDs reduce marginal connection costs for subsequent EV adopters and support equitable allocation of energy costs [30]. Second, prioritising DLM to multiply connectable sockets within existing capacity, while using targeted upgrades only, when necessary, maximises utilisation of legacy infrastructure and, when coupled with tariff automation, maintains peak compliance [39]. Third, standards-led procurement that specifies OCPP 2.0.1, ISO 15118-20 readiness, and OCPI connectivity mitigates vendor lock-in, preserves roaming options, and future-proofs assets for emerging V2X services [11], [23], [24].

Against the backdrop of EV sales roughly tripling since 2019 and tracking toward a 40–45% global market share by 2030, the urgency of EV-ready buildings and smart-managed residential charging is clear [6], [8]. The convergence of market growth, Portuguese urban housing patterns dominated by MUDs, and policy frameworks that increasingly mandate EV-readiness creates a compelling case for solutions that enable high EV penetration in existing multi-unit buildings without costly infrastructure upgrades.

2.3 - Portuguese Low-Voltage Network Architecture and Regulatory Framework

When it comes to Low-Voltage (LV) installations, each country has a set of regulations and legislation with which those installations must comply. This means that any solutions aiming to address the problem of EV charging in existing and future condominiums must comply with these regulatory requirements. This section presents the technical and regulatory context necessary to understand how the proposed solution fits within the legal framework of Portuguese law regarding electrical LV installations.

Distribution Network topology

The starting point of the LV network is the Distribution Transformer (PTD) that converts from Medium Voltage (MV) to Low Voltage. In the Portuguese electrical system, PTs that convert from MV to LV fall into a fixed range of standardised power levels. Although more than 16 power levels are available (ranging from 50 kVA to above 2500 kVA for custom installations) as seen in TABLE III, the Distribution System Operator (DSO), E-Redes, generally installs only the following versions: 250 kVA, 400 kVA, or 630 kVA [40]. Knowing these standardised power levels is important as they play a key role in determining the available power at each condominium downstream in the power grid for charging EVs.

TABLE III
Standardised Power Levels for PTs in the Portuguese MV/LV Grid (E-Redes, 2021).

Power Levels in kVA	
50	630
100	800
160	1000
200	1250
250	1600
315	2000
400	2500
500	(*)

(*) Above 2500 kVA PTs are made to order parameters.

The PTs typically have six outputs on the LV side: four for general use and two for public lighting. Each of the general-use outputs can supply power to one or more buildings, depending on their power requirements. This power distribution is accomplished via switching cabinets that are placed along the main feeder and work as ramification points. At the end of each ramification, there will be a hatch cabinet that serves as the frontier point between the public and private domains. This topology is depicted in FIGURE 2.

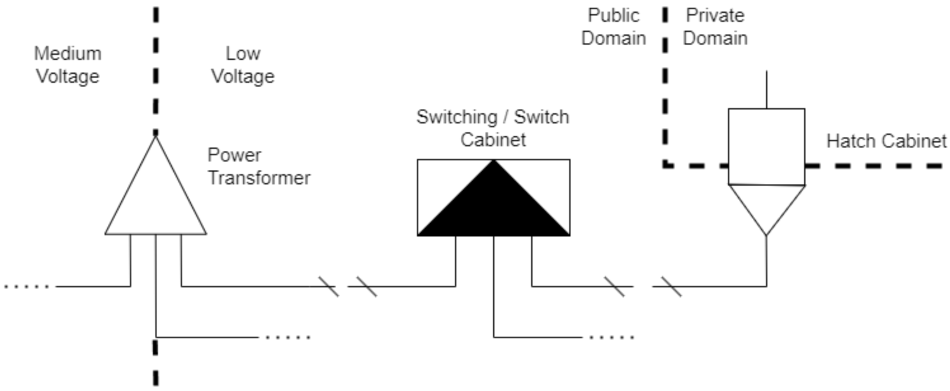


FIGURE 2. Low Voltage Grid typical topology.

Downstream from the hatch cabinet is a privately owned domain that can be either a single building or a private condominium composed of multiple buildings and infrastructure. In the case of a private condominium, the LV grid will have the same topology as a public domain grid: starting at the hatch cabinet, which in some cases may not exist[1] but is mandatory by the DSO, it moves downstream and branches off at switching cabinets, with those branches of the main feeder ending at the respective buildings. This is depicted in FIGURE 3.

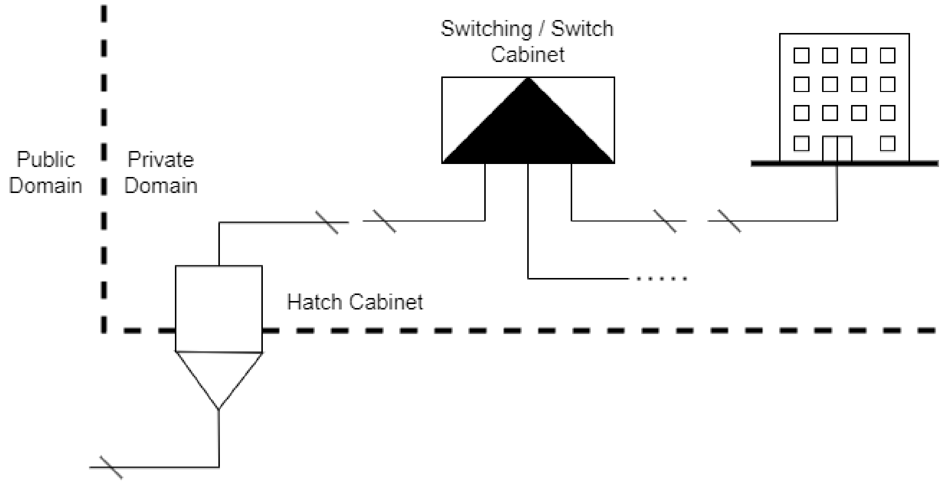


FIGURE 3. Low Voltage Grid typical topology inside a Private Condominium

In the case of a single building, the hatch cabinet represents the direct interface between the public network and the building's internal distribution, as shown in FIGURE 4.

[1] There was a period where these hatch cabinets were not mandatory.

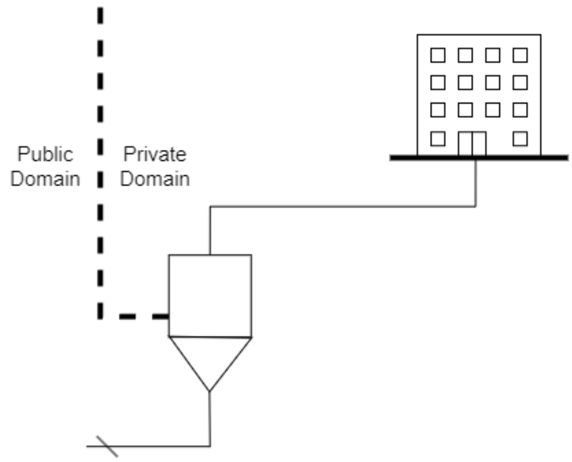


FIGURE 4. Low Voltage Grid typical topology in the case of a single building

Understanding this topology is essential for two reasons. First, it determines the maximum power available for EV charging at any point downstream of the transformer, which depends on the PT rating, the number of buildings sharing that transformer, and the existing loads on each feeder. Second, it identifies the measurement and control points where a DLM system can obtain real-time data and enforce capacity limits.

Regulatory Framework: RTIEBT and Electrical Installation Codes

Low-voltage electrical installations in Portugal must comply with the Regras Técnicas das Instalações Elétricas de Baixa Tensão (RTIEBT), established by Portaria n.º 949-A/2006 of 11 September, pursuant to Decreto-Lei n.º 226/2005 of 28 December, and subsequently amended by Portaria n.º 252/2015 of 19 August [41]. The RTIEBT comprises eight parts and defines the rules for installation and safety of electrical utilisation installations, covering design, dimensioning, execution, and verification of LV electrical systems [41]. These regulations are based on the harmonised European standard HD 60364, which implements IEC 60364 (Low-Voltage electrical installations), ensuring alignment with international best practices while accommodating Portuguese-specific requirements [42].

Key provisions of the RTIEBT relevant to EV charging infrastructure include:

- Protection coordination requirements: Circuit breakers, fuses, and Residual Current Devices (RCDs) must be coordinated to ensure selectivity (discrimination) between protective devices at different levels of the installation. For EV charging, this means that a fault on a charging circuit should trip only the local protective device without affecting other circuits or the main supply [43].
- Conductor sizing standards: Conductor cross-sections must be calculated according to the methods specified in HD 60364-5-52, considering correction factors for ambient temperature, grouping of cables, and installation method. The RTIEBT provides reference tables for maximum allowable currents (Iz) for various conductor types and installation condition [44].
- Metering requirements: For billing purposes in shared installations, meters must comply with the Measuring Instruments Directive (MID 2014/32/EU). Sub-metering

arrangements for individual charging points within a condominium must ensure accuracy class appropriate for commercial transactions [45].

- Earthing and bonding: EV charging equipment requires appropriate earthing arrangements and equipotential bonding in accordance with Section 4 of the RTIEBT, ensuring that exposed conductive parts are connected to the protective earth and that touch voltages remain within safe limits under fault conditions [46].

Portaria n.º 252/2015 introduced Section 722 to Part 7 of the RTIEBT, specifically addressing requirements for EV charging installations in residential and commercial buildings [47]. All calculations and designs in subsequent chapters of this thesis were carried out according to the Portuguese low-voltage regulations for electrical infrastructure design [48].

Condominium Law and Shared infrastructure

The installation of EV charging infrastructure in multi-unit dwellings intersects with Portuguese condominium law (*propriedade horizontal*), governed by Articles 1414 to 1438-A of the Portuguese Civil Code (Código Civil). Under this regime, each co-owner holds exclusive ownership of their individual fraction (autonomous unit) while sharing co-ownership of common parts of the building [49]. The common parts, as defined in Article 1421 of the Civil Code, include: the soil, foundations, columns, pillars, and main structural walls; the roof or roof terraces; entrances, halls, stairways, and corridors for common use; and general installations for water, electricity, heating, air conditioning, gas, and communications [50].

For EV charging infrastructure, several aspects of condominium law are particularly relevant:

- Decision-making for common areas: Under Law 8/2022 of 10 January, which updated the horizontal property regime, decisions affecting common parts generally require approval by the assembly of co-owners. However, modifications to individual fractions or installations that do not affect the building structure may proceed with written notification to the administration [51].

- Cost allocation: The standard rule for common expenses is allocation according to each fraction's permillage (the proportional value of each fraction relative to the total building value). However, the condominium regulations may establish alternative allocation methods, such as equal division or allocation based on actual usage, for specific services [51].

- Individual installation rights: Portuguese legislation now explicitly permits individual owners, tenants, or legal occupants to install charging points in common areas, provided they notify the administration 30 days in advance and comply with Direção Geral de Energia e Geologia (DGEG) technical requirements. The administration's grounds for opposition are limited to the three specific circumstances noted in sub section Policy Framework: European Union and Portugal [37].

- New construction requirements: From 2023 onwards, all residential buildings must include a charging station for electric vehicles per parking space, significantly improving the situation for new developments while the retrofit challenge remains for existing buildings [52].

These legal and governance considerations significantly influence the feasibility and design of MUD charging solutions, as technical optimality must be balanced against the practical constraints of collective decision-making.

Grid Connection Procedures

When existing electrical capacity is insufficient for EV charging loads, building owners must navigate the grid connection upgrade process administered by E-Redes. The process involves several steps and costs that provide strong motivation for solutions that maximise utilisation of existing capacity [53].

For new connections or power increases, applicants must submit a request through the e-Redes digital portal (balcão digital), providing documentation including proof of legal occupation and, for new installations, plans prepared by a certified electrician along with the occupation approval (Licença de Habitação). The connection budget is provided within 15 working days and includes: (i) process charges fixed by ERSE and updated annually; (ii) network connection costs that depend on the requested power level; and (iii) costs for any necessary infrastructure reinforcement.

For existing buildings seeking to add significant EV charging capacity, the typical options include increasing the contracted power within existing infrastructure limits; upgrading the building's main feeder and associated protective devices; or in extreme cases, requesting transformer capacity expansion. The latter can involve lead times of 1-3 years and significant costs, particularly in urban areas where underground infrastructure modifications are required [54]. These costs and complexities provide strong motivation for solutions that maximise utilisation of existing capacity through intelligent load management, rather than defaulting to infrastructure reinforcement.

Implications for EV Charging Infrastructure Design

The Portuguese LV network architecture and regulatory framework impose specific constraints that any practical MUD charging solution must address. Capacity constraints derive directly from the standardised PT ratings and feeder topology: a building served by a 400 kVA transformer shared with other buildings has a fundamentally limited power envelope for EV charging, regardless of the sophistication of the charging equipment installed.

Protection coordination requirements mean that any EV charging branch must integrate properly with existing protection schemes, maintaining selectivity (discrimination) between protective devices at different levels. This constrains both the maximum current that can be drawn on a single circuit and the aggregate current across multiple circuits.

Measurement points available in the existing infrastructure determine where a DLM system can obtain the real-time data needed for capacity management. Typically, this means measurement at the main feeder level (post-hatch cabinet) and potentially at individual apartment or common-area switchboards.

These constraints directly inform the coincidence factor analysis in Chapter 4 and the optimisation formulation developed in Chapter 4. Understanding the topology is essential for locating control and measurement points and for estimating the power available for charging downstream of any given point in the network.

2.4 - Electric Vehicle Charging Behaviour and User Needs

Understanding EV charging behaviour is essential for two reasons: it defines the demand profile that charging systems must accommodate, and it reveals the flexibility that enables optimisation. This section synthesises empirical evidence on when, where, and how EV users charge their vehicles, with particular attention to the implications for MUD charging system design.

Temporal Charging Patterns

Residential charging management is important due to the typical load curve: private EV users tend to charge their vehicles at roughly the same time, causing congestion. According to one study [55], there is a concentration of charging that begins around midnight as users tend to prefer this period for lower tariffs. However, according to another study [56], charging may start earlier in the day, closer to 18:00, as this is the time when most users complete their last journey of the day.

Figure 5 showcases how interviewees responded when asked the time of return from their last journey each day of the week and weekend. It is possible to observe that answers concentrate in the later hours of the afternoon. This information is valuable when considering that a person will only charge their EV when they know or think they will not be leaving home again until the next day.

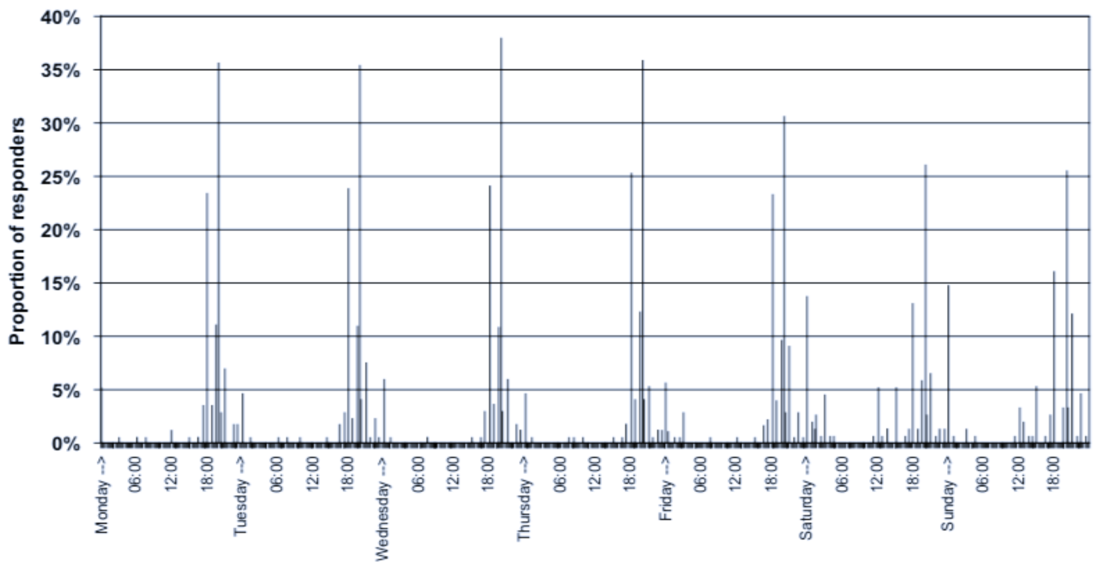


Figure 5. “Last journey of the day?” survey responses [56].

It should be noted that this work [56] was carried out before the 2019–2022 COVID-19 pandemic, which heavily impacted daily commute habits through the implementation of remote work and travelling restrictions. Post-pandemic patterns may show greater variability in arrival times, though the fundamental clustering around evening hours is likely to persist for many commuters.

More recent large-scale empirical studies have confirmed these temporal patterns while providing additional granularity. A 2024 study published in PNAS analysing over 6.4 million charging session records from California (2011-2023) found that home charging events cluster strongly in evening hours, with uncontrolled residential charging creating peak demand that coincides with existing grid peaks [57]. A comprehensive 2025 study examining 1.6 million EVs across seven major Chinese cities similarly found day-time high-power charging presenting high loads across all vehicle types, with maximum loads concentrated in city centres [58]. These findings underscore the importance of controlled charging strategies that can shift demand away from peak periods.

Location Preferences

Studies consistently demonstrate a strong preference for home charging among EV users. According to a survey report, user charging location preferences show home as the dominant choice, followed by workplace and public charging locations (Figure 6).

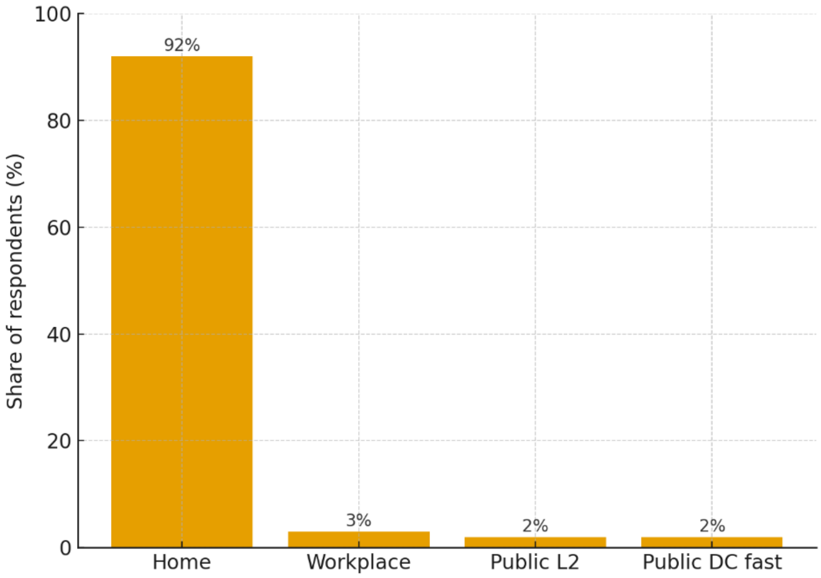


Figure 6. User charging location preference according to *Plug in America* 2022 survey report, among EV owners with a single dominant charging location (83% of respondents).

According to European data [59], in the EU approximately 60% of charging hours take place at home, compared to 15% at work, 15% at the destination, and 10% during the commute to public charging stations. This distribution reinforces the centrality of residential charging to overall EV energy delivery and underscores the importance of viable MUD solutions for urban EV adoption.

Research on charging location choices indicates that non-home charging is particularly important for PEV owners in multi-unit dwellings who may lack dedicated home charging access [60]. A study of 7,979 plug-in electric vehicle owners in California found that owners of long-range and short-range BEVs have different mixed usage patterns of charging infrastructure, with commute distance being an important factor explaining charging behaviour [Ibid][2]. While Level 1 or Level 2 charging is preferable in locations with long dwell times such as home or work, DC fast charging is typically reserved for long-distance trips [Ibid]. These findings suggest that providing adequate home charging in MUDs is crucial for reducing reliance on public infrastructure and enhancing the overall EV ownership experience.

State of Charge Behaviour and Daily Energy Needs

When designing an effective charging method, it should be made clear that EV users will not always need to charge their batteries to 100% State of Charge (SoC) [61], nor is it recommended by manufacturers or current literature to charge to maximum capacity every day [62]. The work carried out in [63] showed that most charging events took place when SoC was between 20% and 80%, and that charging sessions at home tended to start at the lower end of this interval.

Daily commuting and arrival/departure times from the household are well studied. Authors in [64] observe that a substantial proportion of commuters have morning and afternoon peaks of arriving at and departing from work. Work in [65] presents clear traffic measurements at different times of day, backing this claim. Authors in [66] present valuable insights on the distances travelled between charging sessions, indicating that EV users can be empowered with information about how much charge they require and how much time they need to meet their needs.

Studies such as [67] suggest that not only is charging at home the preferred alternative, but charging at night is also the preferred time. Authors in [68] conclude that 59.1% of consumers prefer to charge their EVs at night and that consumers prefer to charge at home.

Charging Flexibility

A critical concept for MUD charging system design is charging flexibility, which refers to the difference between the time a vehicle is connected (dwell time) and the minimum time required to deliver the necessary energy (charging time). For residential overnight charging, this flexibility is typically substantial: a vehicle arriving at 19:00 and departing at 07:00 has a 12-hour dwell time, while delivering 20 kWh at 7 kW requires less than 3 hours of actual charging.

This flexibility is the resource that enables capacity-constrained optimisation. By distributing charging across the available dwell time window, a DLM system can serve multiple vehicles with aggregate energy demands that exceed instantaneous capacity limits. The greater the ratio of dwell time to required charging time, the more vehicles can share a given capacity allocation.

Work in [69] splits EV users into nine types, where three of them (representing 49%) charge overnight. The authors also demonstrate that these EV owners charge, on average, between 1.86 to 4.60 times per week. This pattern arises because EV users do not require a full charge

[2] Refers to the previous reference.

every day, nor do they complete a full discharge every day, further increasing the flexibility available to scheduling algorithms.

Recent research has provided quantitative insights into the relationship between charging behaviour and infrastructure utilisation. A 2024 study on managed residential EV charging found that implementing controlled charging strategies avoids on-peak periods and reduces households' daily electricity costs by 38.87% during summer and 44.3% during winter [70]. The study demonstrated that charging at a shared station results in the lowest share of charging time over dwell time, while charging for valley filling and peak shaving offers less flexibility but has the advantage of flattening the load curve and mitigating high peak loads [Ibid]. These findings directly support the DLM approach developed in this thesis, which seeks to exploit dwell time flexibility to accommodate multiple vehicles within constrained capacity.

Behavioural Parameters for Simulation

The behavioural patterns documented above must be translated into parameters for simulation and optimisation. Drawing on the reviewed literature, TABLE IV summarises the key parameters that inform the simulation design in Chapter 4.

TABLE IV
Behavioural Parameter Summary for Simulation Design

Parameter	Distribution | Value	Source	Notes
Plug-in time	Normal dist. mean 19:00-21:00	[56] [67]	Evening Clustering
Plug-out time	Normal dist. mean 07:00 – 09:00	[64] [65]	Morning Clustering
Daily Energy Need	Based on commutes 5-50 kWh	[66]	Variable by User
Desired SOC	Typically, 80%, user-declared	[61] [62] [63]	Not always 100%
Battery Capacity	Normal dist. mean 40-60 kWh	[71][72]	20-100 kWh range

These parameters enable the generation of synthetic charging demand profiles that respect observed behavioural patterns while allowing systematic variation for sensitivity analysis. The explicit documentation of parameter sources and distributions supports reproducibility and enables future refinement as more empirical data becomes available.

2.5 - Electric Vehicle Charging Optimisation: A Critical Review

With the development and progress of electric vehicle technology, several issues have been the focus of industry and academia, including range, charging times, and battery recycling. However, one key issue remains under addressed: how EV owners will charge their vehicles when they live in multi-dwelling residential buildings with electrical infrastructure that was not designed or built to support the increased power demand.

Current literature presents charging management solutions, but these solutions tend to centre around principles that do not directly address the MUD capacity constraint problem. This section critically reviews existing optimisation approaches, organises them into a taxonomy, and identifies the gaps that motivate this research.

Taxonomy of Optimisation Approaches

EV charging optimisation approaches can be classified along several dimensions that reveal their applicability, or lack thereof, to the MUD context. By objective function: (i) cost minimisation through tariff-aware scheduling; (ii) emissions minimisation through carbon-aware scheduling; (iii) capacity maximisation through peak shaving and load levelling; (iv) user satisfaction maximisation through SoC targets and departure readiness; (v) degradation minimisation through battery health preservation; and (vi) multi-objective combinations that balance competing goals.

By decision variables: (i) binary decisions (charge yes/no in each time slot); (ii) continuous rate modulation (charging current as a continuous variable); and (iii) timing decisions (start time, duration, interruption). By solution method: (i) Linear Programming (LP) and Mixed-Integer LP (MILP) for convex or convexified problems; (ii) Non-Linear Programming (NLP) including Sequential Quadratic Programming methods such as SLSQP; (iii) heuristics and meta-heuristics including genetic algorithms and particle swarm optimisation; and (iv) reinforcement learning approaches for adaptive control. By time scale: (i) real-time control operating at sub-minute to minute granularity; (ii) day-ahead scheduling for known or predicted sessions; and (iii) strategic planning for infrastructure investment decisions.

Cost and Emission Minimisation Approaches

The first category of solutions defines the management criterion as reducing the overall charging cost of the consumer, considering that electricity prices vary during the day. A solution centered on this principle is proposed for residential use in [73], which presents an algorithm that aims to ensure minimum charging costs based on day-ahead electricity prices. Authors in [74] have presented a pricing methodology that aims to avoid concentration in hours of peak demand. Authors in [75] proposed a forecasting model to predict next-day electricity prices, enabling economic EV charging scheduling. Authors in [76] note that traditional optimisation studies of EV charging have mostly focused on the profit for operators, costs to consumers, or stability of the electricity network, whereas this study aims to optimise EV charging schedules with the goal of minimising greenhouse gas emissions from electricity generation.

However, these studies overlook a critical problem: while reducing charging costs is an attractive feature for the end consumer, it can lead to high concentrations of EVs charging simultaneously on a large geographical scale, thereby exacerbating the main problems of energy availability and the distribution system's capacity. Cost and emissions optimisation, by design, shifts load to periods of low price or low carbon intensity, periods that may coincide across many users, creating new demand peaks that stress local infrastructure.

Other Optimisation Approaches

The literature also addresses public charging station optimisation [77], photovoltaic and battery energy storage integration [78], [79], [80], and grid reinforcement strategies [81], [82], [83]. However, public station solutions address a fundamentally different problem, one of reservation and routing. PV/BES solutions are capital-intensive, require weather-dependent operation, and face collective decision-making challenges in MUDs. The author in [82] concludes that the costs and emissions of grid reinforcements outweigh the benefits, suggesting that software-based optimisation within existing capacity is economically superior to hardware-based capacity expansion.

Dynamic Load Management for Multi-unit Dwellings

Within the MUD context, Dynamic Load Management (DLM) emerges as the keystone technology for scale. By continuously modulating per-socket power against a global capacity cap and observed building load, DLM allows the number of active connectors to grow without immediate investment in expensive service upgrades [7], [16]. A widely replicated reference architecture comprises a site energy gateway that measures the main income and EVSE branches, enforces capacity limits and priorities, and interfaces with a CPMS over the OCPP protocol. Operating models vary with building governance and financing conditions. Comparative analyses show that strategies privileging pre-cabling and DLM are more cost-effective and scalable than ad-hoc retrofits as EV penetration increases [16], [10].

Coincidence Factors in Infrastructure Sizing

One of the steps in electrical feeder sizing involves applying a coincidence factor (Cf), also known as the simultaneity coefficient or demand factor. This coefficient enables a reduction in the cross-section of power supply conductors by leveraging the observation that the probability of simultaneity in maximum consumption peaks decreases as the number of loads increases. Different countries have different regulations and coincidence factors for residential loads, as seen in TABLE V and TABLE VI

TABLE V
Coincidence factor according to Portuguese regulations [84]

Number of electrical installations downstream	Coincidence factor
2 to 4	1,00
5 to 9	0,75
10 to 14	0,56
…	…
35 to 39	0,37
40 to 49	0,36
More than 49	0,34

TABLE VI
Demand Factor according to the North American regulation [85]

Number of Dwelling Units	Demand Factor (%)
3 to 5	45%
6 to 7	44%
8 to 10	43%
…	…
51 to 55	25%
56 to 61	24%
62 and over	23%

As can be observed, studies have tended to focus on fixed levels of charging power with non-optimised power supply, as well as primarily examining public charging stations and single dwellings located in residential areas. Critical limitations include: (i) assumption of fixed charging power rather than modulated power under DLM; (ii) focus on single-family dwellings or public stations rather than MUDs; (iii) no consideration of optimised charging that would reduce coincident demand; and (iv) limited fleet sizes that may not capture the diversity effects at the MUD scale.

Fairness and Equity Considerations

In MUD contexts where multiple users share constrained capacity, the allocation of charging resources raises fairness considerations that most optimisation literature does not explicitly address. When capacity is insufficient to simultaneously serve all connected vehicles at the maximum rate, how should the available capacity be distributed? Several fairness principles could apply, including proportional allocation based on energy needs, max-min fairness that maximises the minimum allocation, priority-based allocation favouring users with earlier departure times or greater urgency, or market-based allocation through pricing mechanisms.

The optimisation formulation developed in this thesis addresses fairness implicitly through its objective function, which minimises the "time error" between achieving the desired SoC and the declared departure time across all users. This approach prioritises users with tighter constraints while distributing capacity equitably among those with similar needs.

Section Synthesis

Although the problem of EV charging in MUDs is clearly stated, neither the industry nor the research community has given it the attention it deserves, focusing instead on presenting solutions to other problems that do not directly transfer to the MUD capacity constraint problem. This analysis positions the present research: what is needed is a capacity-constrained, user-needs-aware optimisation methodology specifically designed for MUDs, accompanied by coincidence factors that account for the diversity benefits of optimised charging.

2.6 - Battery Degradation: State of the Art and Scheduling Implications

Charging optimisation that ignores battery health may solve the capacity management problem while creating a new problem: accelerated degradation of vehicle batteries. This section reviews the current understanding of Li-ion battery degradation mechanisms, empirical degradation models suitable for scheduling applications, and the implications for MUD charging system design. The section is intentionally condensed, as Chapter 5 develops the degradation-aware scheduling methodology in detail.

Degradation Mechanisms

Lithium-ion traction batteries degrade through a combination of calendar aging (time, temperature, and state-of-charge exposure while idle) and cycle aging (usage-dependent stress from charging/discharging). Macroscopically, degradation manifests as capacity loss and impedance rise; microscopically, it is driven by growth of the solid-electrolyte interphase (SEI), loss of active lithium and active material, structural changes in electrodes, and side reactions accelerated by elevated temperature and high potentials, especially at high state of charge (SoC). Authoritative reviews synthesise these pathways and their dependence on chemistry and operating conditions [86].

Calendar aging is particularly sensitive to temperature and SoC: empirical studies across commercial cell chemistries show Arrhenius-like temperature dependence and faster degradation at sustained high SoC, with non-linear effects indicating that the worst degradation does not always occur strictly at 100% SoC but typically in upper-SoC bands relevant to EV usage [87]. Elevated charging rates (C-rates) increase overpotentials and heat generation, and can induce lithium plating at low temperatures and high SoC, accelerating capacity fade [88], [89].

Empirical Degradation Models

For scheduling applications, physics-based electrochemical models are typically too computationally expensive for real-time use. Instead, empirical or semi-empirical models that capture the dominant degradation factors through parameterised equations are preferred. These models trade mechanistic completeness for computational tractability.

Battery aging models can be broadly categorised into three types: physics-based electrochemical models that quantify the impact of various factors on aging and are used to optimise physical battery aspects; equivalent circuit models that characterise parameters such as voltage, power, and current; and empirical or data-driven models that establish connections between aging and specific variables using experimental data [90]. Semi-empirical models such as those developed by Naumann et al. [91], [92] Schmalstieg et al. and Ecker et al. have been widely used for EV applications due to their balance of accuracy and computational efficiency [90].

This thesis employs the cycle-aging model from Naumann et al. [91], [92], which parameterises degradation as a function of processed charge, average SoC, SoC deviation, and temperature. This model was selected for several reasons: it has been validated against commercial LFP and Nickel Manganese Cobalt (NMC) chemistries representative of current EV batteries through comprehensive aging studies spanning 19 cycle aging and 17 calendar aging test points over 885 days [93]; it requires only inputs that are available or estimable in a charging management context (SoC, current, temperature); and it is computationally lightweight, enabling real-time application. The model details and parameterisation are presented in Chapter 5.

Field Evidence on Charging and Degradation

While laboratory evidence cautions about plating and thermal stress during extreme fast charging, fleet telematics and large-sample field datasets show more nuanced outcomes. Across thousands of vehicles and millions of vehicle-days, average annual range losses in modern EVs is approximately 1% to 2% per year, with model-to-model differences largely explained by chemistry and thermal management rather than DC-fast-charging frequency per se [94]. Recent analyses report no statistically significant additional degradation in certain vehicle cohorts that frequently use DC fast charging versus those that rarely do, within observed usage ranges.

For MUD overnight charging, the AC Level 2 context (typically 7–22 kW, implying C-rates well below 0.5C for most battery sizes) is inherently gentler than the DCFC scenarios studied in field degradation analyses. This suggests that degradation-aware scheduling in MUDs is primarily about avoiding unnecessary stress (prolonged high SoC, unnecessary full charges) rather than managing extreme conditions.

Degradation-Aware Scheduling Approaches

In buildings and fleets, optimisation frameworks that co-optimise energy cost, grid constraints, and battery wear can materially lower life-cycle cost. Models penalising deep cycling, high C-rates at cold temperatures, or prolonged high-SoC parking produce schedules that meet mobility requirements while mitigating degradation, often with negligible impact on user convenience when combined with dynamic load management [95], [96].

Key strategies for degradation-aware scheduling include: targeting mid-SoC bands (e.g., 20–80% for daily use) to minimise time at high voltage and temperature; moderating C-rate where thermal rise would be excessive; temperature-aware timing that delays charging to warmer periods in unheated garages; and distributing energy delivery across available dwell time rather than charging as fast as possible.

Section Synthesis

The scientific evidence converges on a pragmatic message: temperature control, SoC management, and protocol shape dominate battery longevity outcomes, and these levers are accessible to both vehicle BMS and site-level schedulers. For planners of multi-residential charging, embedding these insights into DLM policies, favoring moderate C-rates, mid-SoC targets, and temperature-aware timing, can mitigate degradation while meeting user needs and grid constraints [97], [98].

For MUD contexts specifically, the overnight AC charging paradigm offers inherent advantages: long dwell times enable gentle charging rates, and the flexibility to schedule delivery across the night allows for the avoidance of unnecessary high-SoC dwell. Chapter 5 develops a lightweight degradation-aware scheduling layer that internalises these considerations while remaining computationally tractable for real-time implementation.

2.7 - Research Gap Synthesis

The preceding sections have systematically reviewed the state of the art across five domains relevant to MUD EV charging. This section synthesises the identified gaps and demonstrates how they motivate the research questions stated in Chapter 1.

Consolidated Research Gap

Based on the literature review, the following specific gaps remain unaddressed:

GAP 1 (MARKET-INFRASTRUCTURE MISMATCH): Despite accelerating EV adoption and clear evidence that home charging dominates energy delivery, cost-effective and scalable charging solutions for existing MUD buildings remain underdeveloped. The Portuguese context, with high urban density and prevalence of multi-unit housing, makes this gap particularly acute. Existing solutions either assume single-family dwelling contexts require capital-intensive infrastructure upgrades, or address different problems (public charging, cost optimisation).

GAP 2 (COINCIDENCE FACTORS): Existing coincidence factor studies for EV charging focus on fixed charging power levels, single-family dwellings, or public charging contexts. No studies provide coincidence factors specifically for MUD configurations under optimised charging regimes. This forces designers to use conservative Cf = 1 assumptions that systematically over-size infrastructure, or to adopt values from inapplicable contexts.

GAP 3 (BEHAVIOURAL ASSUMPTIONS): Many optimisation studies employ simplified or unrealistic behavioural assumptions, fixed arrival times, 100% SoC targets, and identical vehicles, which poorly represent the actual diversity and flexibility of MUD users. The flexibility inherent in overnight residential charging (long dwell times relative to charging needs) is underexploited in existing formulations.

GAP 4 (INTEGRATED METHODOLOGY): No integrated methodology exists that combines real-time capacity-constrained optimisation, coincidence factor derivation, and Portuguese regulatory compliance for existing MUDs. Existing work addresses these elements in isolation, without the integration needed for practical deployment.

GAP 5 (DEGRADATION METHODOLOGY): While degradation science is mature and degradation-aware scheduling has been demonstrated in other contexts, lightweight models suitable for real-time MUD scheduling applications remain underdeveloped. The integration of degradation considerations with capacity-constrained MUD optimisation has not been addressed.

Mapping Gaps to Research Questions

The research questions stated in Chapter 1 directly address these identified gaps. The central question, “What is the most cost-effective way to enable high EV penetration in existing multi-dwelling buildings?” emerges from Gap 1. The hypothesis that idle building capacity can be leveraged through centralised, optimised charging addresses Gaps 1, 3, and 4 simultaneously. The need for appropriate coincidence factors (Gap 2) motivates the analysis in Chapter 3. The integration of degradation considerations (Gap 5) motivates Chapter 5.

This alignment between identified gaps and research questions confirms that the research addresses genuine lacunae in the current state of the art rather than problems already solved by existing work.

Contribution Positioning

TABLE VII maps each identified gap to the thesis contributions that address it.

TABLE VII
Gap-to-contribution mapping

Gap	Description	Thesis Contribution	Chapter
1	Lack of cost-effective MUD solutions in the European context.	SLSQP optimiser validated with real data.	Chapter 4
2	No MUD Cf with optimised charging considerations.	C_f framework for 1-70 EVs with fuzzy correction.	Chapter 3
3	Simplified behavioural assumptions.	User-declared SoC, plus behaviour-constrained data.	Chapter 4
4	No integrated methodologies.	Planning artefacts with Portuguese compliance.	Chapter 3
5	No lightweight degradation model for MUDs.	Naumann-based cycle aging, plus thermal-aware scheduling	Chapter 5

Recent analysis of large-scale driving data confirms the viability of this approach: Krein [99] demonstrates that 90% of daily passenger EV range needs can be supported with basic AC infrastructure, and that coordinated slow charging enables a given electrical supply to serve approximately 30 times more vehicles than equivalent fast-charging provision. This reinforces the central premise of the present thesis: that idle building capacity, harvested through intelligent coordination, represents a cost-effective pathway to high EV penetration in existing multi-dwelling buildings

2.8 - Chapter Summary

This chapter has systematically reviewed the state of the art across five interconnected domains to establish the foundation for this research.

The market context analysis established that EV adoption is accelerating globally and in Portugal, with residential charging, particularly in MUDs, emerging as the critical bottleneck for sustainable electrification. Portugal achieved record BEV market share of 22.5% in January 2025, significantly exceeding the EU average, while most urban Portuguese residents live in multi-unit buildings. The technology ecosystem of hardware, software, and standards is maturing, but solutions remain oriented toward single-family dwellings. EU and Portuguese policy, including the new Decreto-Lei n.º 93/2025, increasingly mandates EV-readiness, creating urgency for practical MUD solutions.

The Portuguese LV network analysis documented the specific architecture (standardised transformers, feeder topology, public/private domain boundaries) and regulatory framework (RTIEBT, condominium law, grid connection procedures) that constrain feasible solutions. Understanding this context is essential for designing solutions that are not merely technically optimal but practically deployable within Portuguese regulatory and infrastructure constraints.

The charging behaviour review established that temporal clustering of demand creates the challenge, while substantial flexibility (dwell time far exceeding charging time) creates the opportunity for optimisation. Recent studies confirm that managed charging can reduce electricity costs by 38-44% while exploiting the ratio of dwell time to required charging time. Behavioural parameters from the literature inform the design of the simulation in subsequent chapters.

The critical review of optimisation approaches revealed that existing methods address different problems (cost minimisation, public charging, PV integration) or contexts (single-family dwellings, workplaces) that do not transfer directly to MUDs. DLM emerges as the keystone technology, but MUD-specific optimisation formulations and coincidence factors are lacking.

The battery degradation review established that while degradation science is mature, its integration with MUD charging optimisation remains underdeveloped. The overnight AC charging context is inherently battery-friendly, but explicit degradation-aware scheduling using validated semi-empirical models such as Naumann et al. can provide additional benefits.

Chapter 3

Coincidence Factor Analysis

3.1 - Contextualisation

When considering the planning and dimensioning of electrical feeders for residential power delivery, regulations dictate that these electrical feeders are designed and sized according to several factors, such as feeder length to be installed, power to be delivered, electrical current, feeder material, feeder isolation material, amongst others.

One of the steps in electrical feeder sizing involves applying a coincidence factor, according to IEC 60050, or C_f that is also known as coincidence factor or demand factor (DF), this coefficient allows a reduction in the extent of the increase in the cross-section of the power supply conductors, capitalising on the observation that the probability of simultaneity in the maximum consumption peaks of the various dwellings decreases as the number of dwellings increases. What this means is that the maximum expected current consumed by the building at any given time will not increase in direct proportion to the rise in number of dwellings.

When it comes to EV charging in multi dwelling residential buildings the risk aversion is even higher because, regardless of the number of EVs, the coincidence factor considered is still 1, which would mean that all EVs would be charging at the same, at full capacity, everyday which is a flagrantly penalising assumption that leads to an oversized electrical infrastructure, including electrical feeders, circuit breakers, and power transformers.

Furthermore, the authors of [100] concluded that domestic charging mainly takes place during the night period when EV owners do not need their vehicle for the longest period, this means that there is also no need to charge the batteries at maximum rate, and because electricity tariffs tend to be lower during nighttime. These two factors have a significant impact on the charging requirements presented by larger EV fleets, meaning that, it is not expected for the electrical current requirements to increase in direct proportion with the size of the EV fleet.

Some studies, as seen in TABLE VIII, have aimed at calculating similar C_f but their results differ greatly, however on commonality holds; the C_f is described as a function dependent on the number of EVs charging. This study aims at solving some oversight by leaning on simulations instead of probabilistic studies and considering optimised charging methodologies.

TABLE VIII
Analysis of previous works on coincidence factor estimation for EV charging

Work	Charging Power [kW]	Max Fleet Considered	Typology	Optimised Charging
[101]	6.6, 19.2, 12.9	6 EVs	Single dwelling	No
[102]	3.6	200 EVs	Residential Areas	No
[103]	3.7, 11.0	20 EVs	Single Dwelling	No
[104]	12.0	Not specified	Charging Stations	No
[105]	2, ,6, 3, 15	1000 EVs	Residential Areas	No
[106]	3.7, 14.0	54 EVs	Charging Stations	No
[107]	Not specified	10 EVs	Residential Areas	No
[108]	3.7, 11.0, 22.0	100 EVs	Single Dwelling	No
Current	From 3.68 to 12.8	70 EVs	Multi Dwelling	Yes

As can be observed in TABLE VIII, studies tended to focus on fixed levels of charging power, with non-optimised power supply, as well as mainly looking at public charging stations and single dwellings inserted on residential areas, or focusing on the impact at the level of the distribution grid.

This study, on the other hand, focuses on multi dwelling residential buildings, where restrictions are tighter, and leverages on optimised charging methodologies and simulations, demonstrated in [109].

Beyond this, the present work leverages on other studies to complement the carried-out simulations. Several factors come into play when considering charging patterns and charging habits of EV users, these patterns and habits play a key role in the dimensioning of electrical infrastructures.

The work carried out in [69] splits EV user into 9 types, where three of them, representing 49%, charge overnight the authors also demonstrate that these EV owners charge, on average, between 1.86 to 4.60 times per week. This can happen because, in simple terms, EV users do not require a full charge every day nor do they complete a full discharge every day.

3.2 - Methodology

To estimate the coincidence factors, it is paramount to have access to large amounts of data from EV charging in the context of multi-dwelling buildings. However, due to the current levels of adoption of EV, there are insufficient real-life examples from which the necessary data can be extracted.

To overcome this barrier this work relied on synthetic data, generated from software capable of simulating the intrinsic phenomena related to the daily charging and discharging needs of EV and their owners.

This synthetic data is then used to develop a methodology for the estimation of the coincidence factors that, in the future, can be fine-tuned as EV penetration increases.

To generate synthetic data the algorithm and methodology presented in [12] was used, this methodology allows for the integration of high levels of EVs in multi dwelling residential buildings and considers behavioural aspects such as daily discharge needs, industry parameters such as battery size and battery mileage.

As previously explained, coincidence factors allow for an optimised sizing of the electrical feeder, in this case, the feeder that will connect the building main electrical switch board to the charging electrical switch board located in the EVs vicinity.

To calculate the coincidence factor of EV fleets with different sizes were considered, namely 1, 2, 5, 10, 15, 20, 25, 30, 40, 50, 60 and 70 EVs, this will give us 12 points from which it was possible to extrapolate an equation based on the number of EVs as input. In this case, a slow-charging scheme was considered meaning that the maximum charging current per EV is 16A, with this information it was possible to establish the current requirements for each EV fleet as can be seen in the following TABLE IX:

TABLE IX
Charging current requirements by EV fleet size, to charge at maximum charging rate.

EV Fleet Size	Charging Current	Per phase
n = 1	n x 16 = 16 A	16 / 3 = 5.3 A
2	32 A	10.7 A
5	80 A	26.7 A
10	160 A	53.3 A
...	...	...
40	640 A	213.3 A
50	800 A	266.7 A
60	960 A	320 A
70	1120 A	373.3 A

So, as an example, an EV fleet with 40 vehicles would require 640 A (approx. 213.3 A per phase) if all of them were to charge at their maximum rate at the same time.

Once a base case has been established, it is necessary to calculate the real needs of the fleet based on the assumption that the likelihood of every EV of the fleet charging at the same time and at maximum capacity is low.

For each EV fleet size, it is possible to calculate the C_f using the formulation seen in equation (1).

$$C_f = \frac{\text{Maximum Current Observed}}{\text{EV Fleet Size } \times \text{ EV Maximum Current}} \qquad (1)$$

Where:

- Maximum Current Observed is the highest current observed for the respective EV fleet size during the simulations.
- EV Fleet Size is the overall number of EVs in the garage.
- EV Maximum Current is the maximum value at which each EV charges, in this case 16A.

However, this approach, which considers the maximum current observed for each fleet size, may be subject to bias. This is because that value may not be representative of the common needs of the fleet. To avoid this bias, alternative values can be used. For example, the highest measurements can be disregarded, and unlikely outliers can be excluded.

3.3 - Data Generation

The initial stage of this methodology entails the generation of synthetic time series of EV charging power patterns. To this end, a total of 20 simulations were conducted for each of the fleets under consideration, resulting in a total of 240 simulations. Each simulation was executed for the equivalent of 80 days at a 3-minute time-step.

For each fleet size the first 10 simulations were carried out using the same fleet characteristics, meaning that the starting conditions were the same. The remaining 10 simulations were split into two groups of 5, with one group more demanding conditions such as higher daily discharges and larger battery capacities and the other group with less demanding conditions, as illustrated in FIGURE 7.

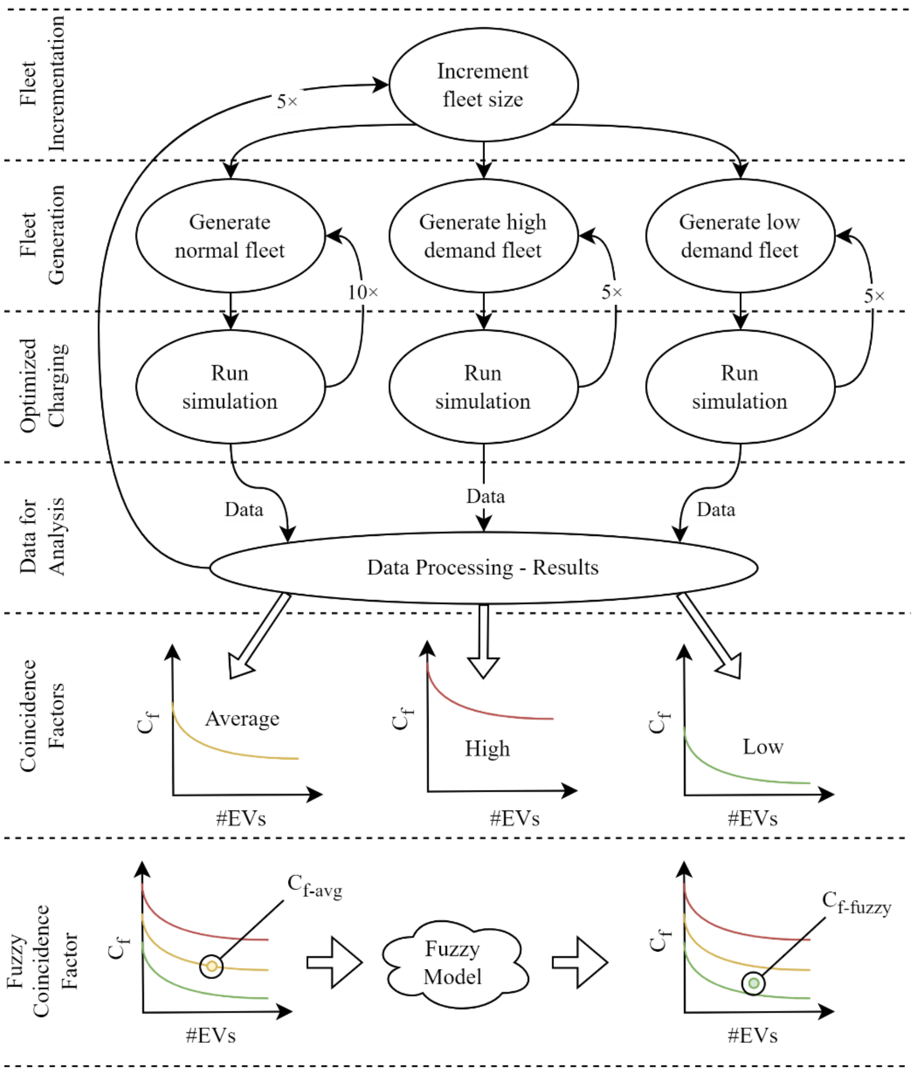


FIGURE 7. Complete methodology overview from data generation to $C_{f\text{-fuzzy}}$ estimation, including optimal charging simulations and result generation.

Finally, this process was run twice, each time with a different optimisation technique. The first optimisation technique aims to minimise the current supplied to each EV without compromising the desired state of charge (SoC) at the end of the charging session. The second optimisation technique aims at maximising the amount of power delivered to the EVs at any given moment.

3.4 - Simulation Results

Optimisation Minimisation Problem

After running the simulations, it was possible to extract the following insights. Regarding the fleet sizes, as expected, as the overall fleet size increases the maximum charging fleet size also increases, however and as expected, it does not increase in a direct proportion as can be seen in the following TABLE X.

So, just by considering the relation between overall fleet size and maximum charging fleet size it is possible to establish a simultaneity factor, as seen in equation (2).

TABLE X
Charging fleet size versus absolute fleet size comparison chart, under minimisation problem.

EV Fleet Size	Charging EV Fleet Size		
	Maximum	Minimum	Rounded Average *
n = 1	1	1	1
2	2	2	2
5	5	5	5
10	10	9	10
...	...	...	...
40	34	23	29
50	48	37	41
60	58	40	47
70	59	37	49

(*) The rounded average is calculated for every group of 20 simulations, and the minimum and maximum values are overall.

$$C_f = \frac{\sum_1^{20} \text{Charging Fleet Size}}{\text{Overall Charging Fleet Size}} \qquad (2)$$

Where:

- Charging Fleet Size is the highest register number of EVs charging during each simulation, at any given moment.
- Overall Charging Fleet Size is the overall number of EVs in the garage.

By solving for C_f in (2), it is possible to calculate the values in the following TABLE XI:

TABLE XI
Coincidence factors for calculating maximum charging fleet size.

Overall Fleet Size	C_f
n = 1	1.00
2	1.00
5	1.00
10	0.99*
...	...
40	0.73
50	0.81
60	0.78
70	0.70

(*) e.g. this value is not $(10 + 9)/10$, but a sum of twenty different measurements over the overall fleet size, this holds for all sizes.

With these C_f values it is possible to extrapolate equation (3) that allows us to calculate the respective C_f value for each size of the overall EV fleet size and consequently the expected maximum size of the charging EV fleet, at any given moment in a charging session.

$$C_{f.fleet\_size} = -0.0044 \times \mathrm{n} + 0.9972, where\ n\ is\ the\ fleet\ size \quad (3)$$

Notice that the sample size for this C_f is relatively limited. Nonetheless, it was feasible to fit a linear curve to the sample and achieve a R^2 value of 0.811. This coefficient, for fleet size, is useful when considering dummy methodologies where the allocated charging current to each EV is not optimised according to its needs.

Regarding the current allocated, as expected once more, the value increased as the overall fleet size increased. These values can be observed in the following TABLE XII:

TABLE XII
Maximum allocated charging current for each EV fleet size.

EV Fleet Size	C_f = 1	Allocated Charging Current	
		Abs. Maximum	99% Maximum
n = 1	16 A	16 A	13.36 A
2	32 A	28.78 A	21.93 A
5	80 A	59.53 A	39.61 A
10	160 A	99.92 A	67.58 A
15	240 A	150.50 A	87.60 A
20	320 A	171.43 A	126.63 A
25	400 A	219.03 A	139.56 A
30	480 A	291. 59 A	150.23 A
40	640 A	294.98 A	202.03 A
50	800 A	435.08 A	257.84 A
60	960 A	507. 30 A	281.41 A
70	1120 A	498.09 A	338.44 A

By observing the values in the previous TABLE XII the reader may notice that the trend, for the "Abs. Maximum" column is more consistent than the trend presented for the fleet size Sc, and this becomes more evident when observing the trend in the "99% Maximum" column. The "99% Maximum" column considers 99% of the observed values, excluding the highest 1% values, that are outliers. In TABLE XIII the reader can see the distribution of values for the fleet with size 70.

TABLE XIII
Frequency distribution of allocated charging current across intervals.

From [A]	To [A]	Max. Measured	Count	Percentage [%]
6	16	16,00	22052	10.40%
16	32	32,00	15997	7.54%
32	48	48,00	11426	5.39%
48	64	64,00	10360	4.88%
64	80	80,00	9706	4.58%
		…		
336	352	351,99	502	0.24%
		…		
432	448	447,85	131	0.06%
448	464	462,27	79	0.04%
464	480	479,90	80	0.04%
480	496	495,81	160	0.08%
496	512	498,09	2	0.00%

As we can see, just by discarding the highest 1%, the maximum drops from 498.09 A to 351.99 A. Furthermore, as demonstrated in FIGURE 8 the actual knee of the cumulative sum curve is closer to the 90% mark at 256.0 A, nonetheless, to disregard 10% of the values and design the system for a maximum of 256.0 A would put the feeders at higher risk of overheating and causing recurring system shutdowns.

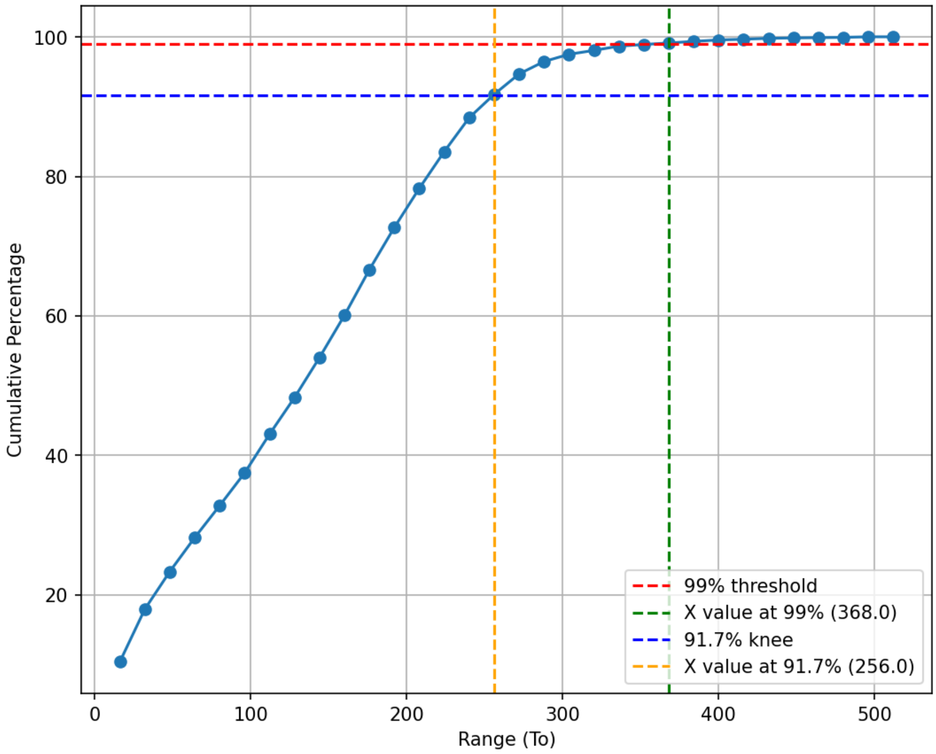


FIGURE 8. Cumulative frequency of the allocated current with 99% threshold.

On the basis of this data it was possible to calculate the C_f values for the allocated current for each of the fleet's size, taking into account both the maximum allocated current and the maximum allocated current when disregarding the highest 1% values, these C_f values can be seen in TABLE XIV:

TABLE XIV
S_c for current calculation per fleet size considering minimisation problem.

Overall Fleet Size	C_{f1} - Abs. Maximum	C_{f2} - 99% Maximum
n = 1	0.90	0.65
2	0.78	0.56
5	0.65	0.46
10	0.57	0.39
...	...	...
40	0.43	0.29
50	0.41	0.27
60	0.39	0.26
70	0.38	0.25

In the following FIGURE 9 and FIGURE 10 the reader can see the plotting of both C_f. For both C_f values, the curve that best fits the distribution is a power trendline which are described in the following formulation seen in equations (4) and (5).

$$C_{f1} = \frac{0.9023}{n^{0.203}} \text{ , where n is the fleet size} \quad (4)$$

$$C_{f2} = \frac{0.6542}{n^{0.224}} \text{ , where n is the fleet size} \quad (5)$$

Both trendlines have relatively high R^2 values, of 0.85 and 0.89, well within the range of what is considered a good fit.

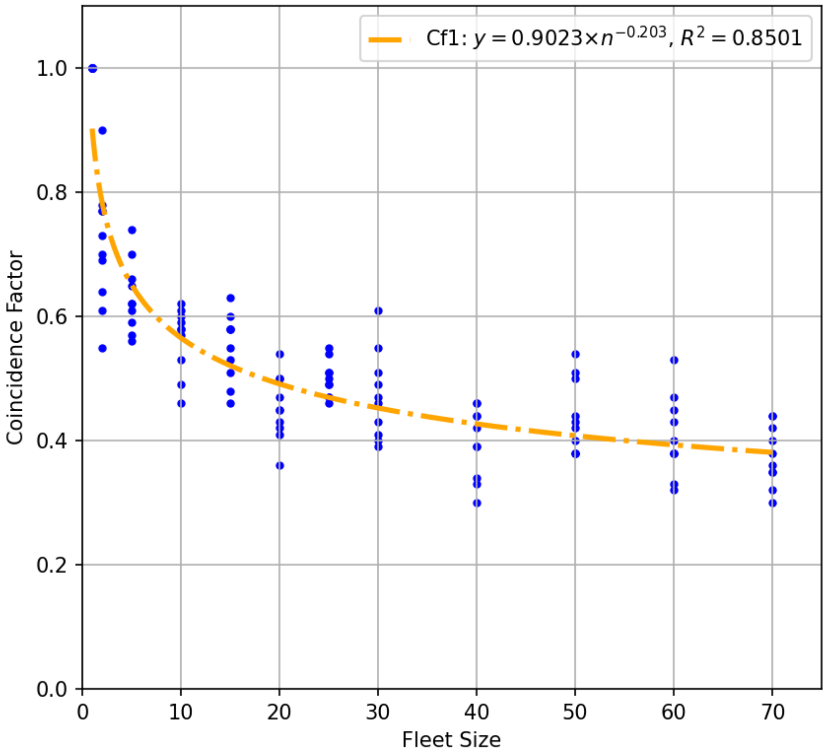


FIGURE 9. Scatter plot of C_{f1} according to simulation results.

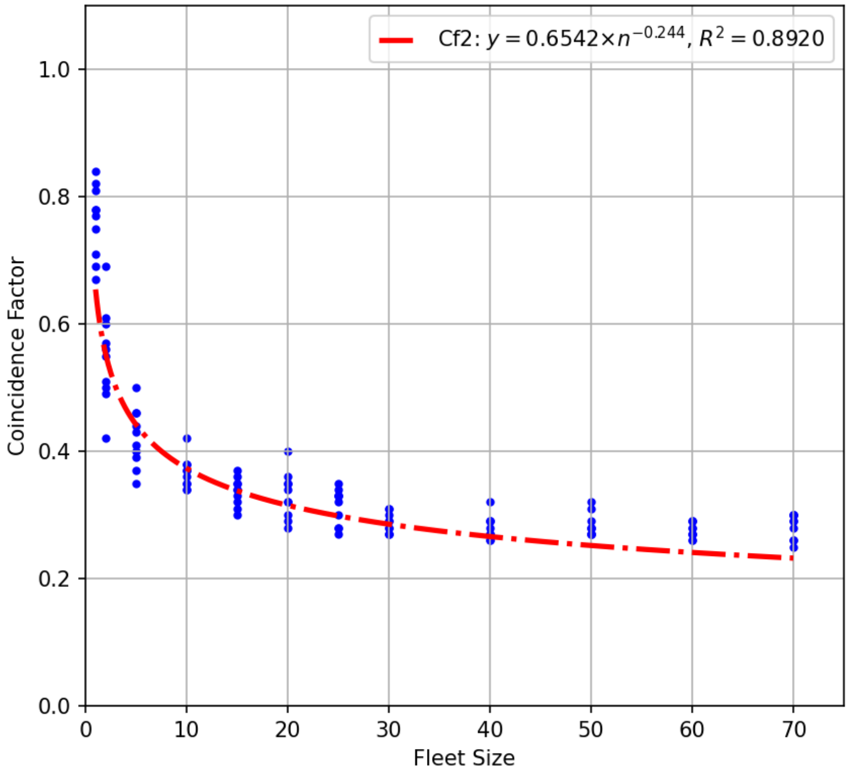


FIGURE 10. Scatter plot of C_{f2} according to simulation results.

Optimisation – Maximisation Problem

Contrasting with the minimisation problem, the maximisation problem aims to allocate the maximum possible electrical current to each charging EV, even if this means that the EV will reach its desired SoC before the expected unplug time, TABLE XV presents the number of EVs charging in relation to the fleet size.

TABLE XV
Charging fleet size versus absolute fleet size comparison chart, under max. problem.

EV Fleet Size	Charging EV Fleet Size		
	Maximum	Minimum	Rounded Average
n = 1	1	1	1
2	2	2	2
5	5	5	5
10	10	10	10
…	…	…	…
40	36	21	30
50	44	33	40
60	53	33	43
70	63	43	53

As expected, this leads to higher overall charging currents which translate into higher coincidence factors as seen in TABLE XVI, where C_{f3} and C_{f4} are presented.

TABLE XVI
Coincidence factor for current calculations per fleet size, considering max. problem.

Overall Fleet Size	C_{f3}	C_{f4}
n = 1	1.00	1.00
2	1.00	0.87
5	1.00	0.71
10	0.98	0.61
…	…	…
40	0.79	0.45
50	0.72	0.43
60	0.66	0.41
70	0.59	0.40

Both the above coincidence factors are described by the (6) and (7) respectively.

$$C_{f3} = -0.0065 \times \mathrm{n} + 1.0478 \tag{6}$$

$$C_{f4} = \frac{1.008}{\mathrm{n}^{0.219}} \tag{7}$$

The following FIGURE 11 and FIGURE 12 present the plot for S_{c3} and S_{c4} respectively.

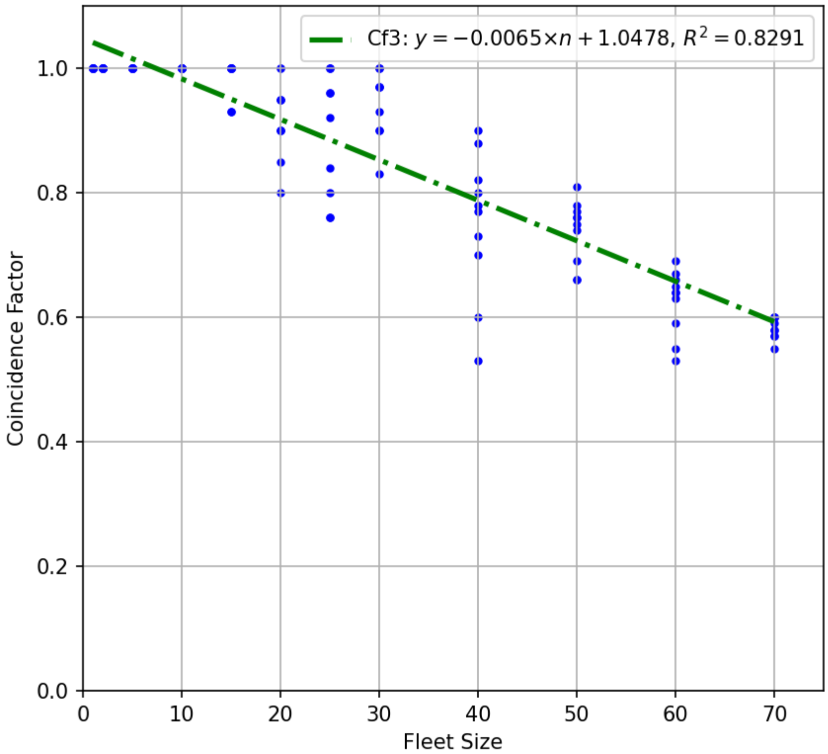


FIGURE 11. Scatter plot of C_{f3} according to simulation results.

When considering the maximisation of power delivery problem, without excluding the top 1% measures, we can see that the coincidence factor behaves almost linearly in relation to the size of the fleet.

Contrarily, if we exclude the top 1% measures the coincidence factor regains the non-linear behaviour, in relation to the fleet size, that was demonstrated previously.

In FIGURE 13 a combined plot of all the trendlines is demonstrated together with a plot of the average trendline. This average trendline only considers C_{f1}, C_{f2}, and C_{f4}, due to the behaviour of the curve and the assumptions for its calculation C_{f3} was not considered. The trendline $C_{f\text{-}avg}$ can be described by the formulation seen in equation (8).

$$C_{f-avg} = \frac{0.8548}{n^{0.2223}} \tag{8}$$

This $S_{c.avg}$ has the following R^2 when compared against the datasets of S_{c1}, S_{c2}, and S_{c4}, as seen in TABLE XVII.

TABLE XVII
R-squared values of $C_{f\text{-}avg}$ against remaining C_f values and curves.

Data	R-squared Value
C_{f1}	0.753
C_{f2}	0.257
C_{f4}	0.638

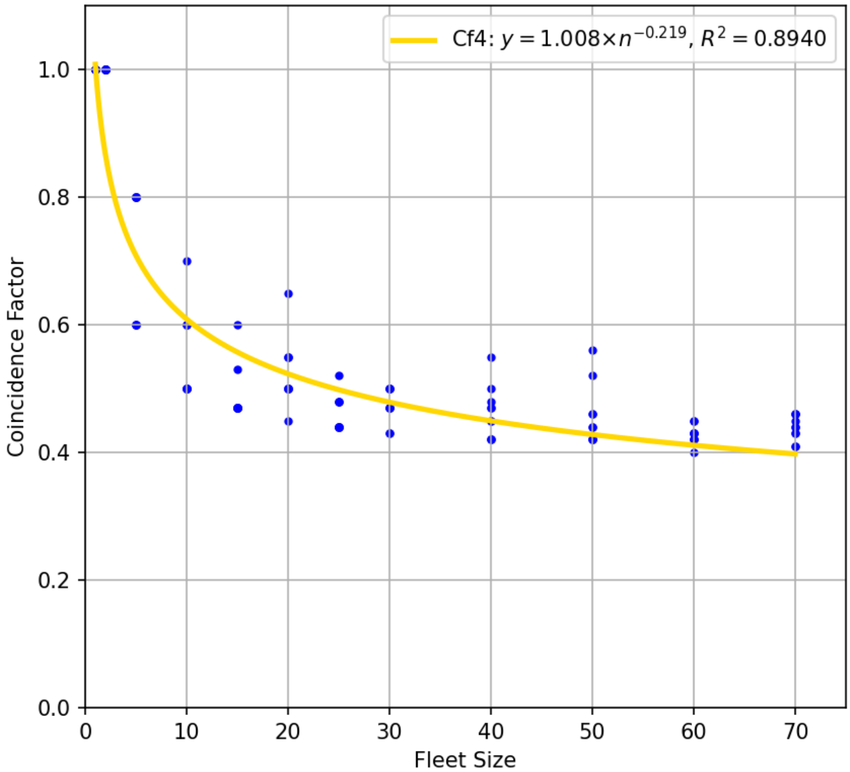


FIGURE 12. Scatter plot of C_{f4} according to simulation results.

To consider C_{f3} into this estimation first we need to calculate a new trendline as a power function. This new approximation can be seen in TABLE XVII, and as expected the R^2 is significantly lower, it drops from 0.8291 to 0.476.

The trendlines for the coincidence factors of C_{f2} and C_{f3} will be the boundaries for the $C_{f\text{-}fuzzy}$, meaning that this area will be the sample space for the results, using $C_{f.avg}$ as the starting point.

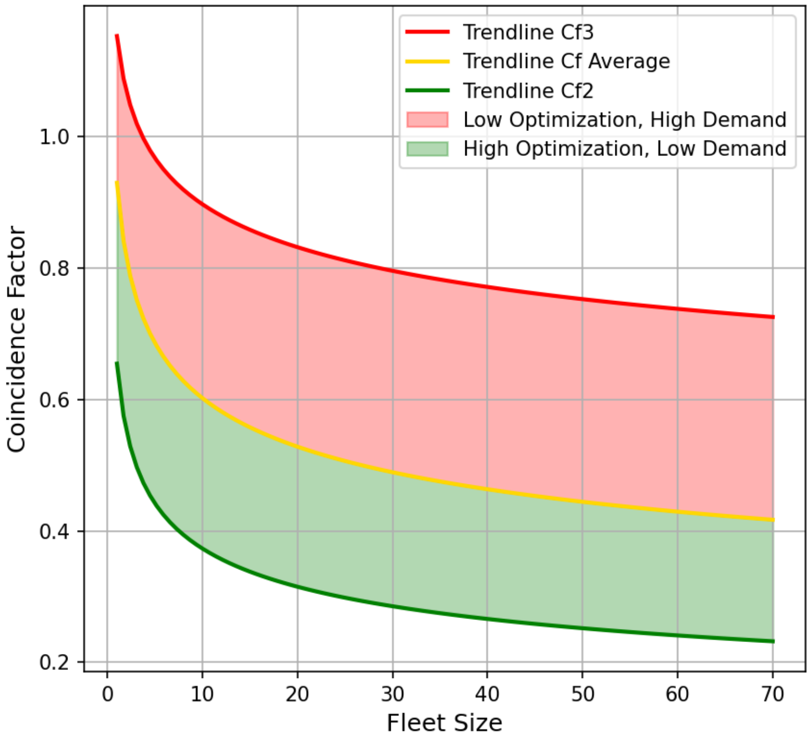


FIGURE 13. Scatter plot of all the calculated C_f according to simulation results.

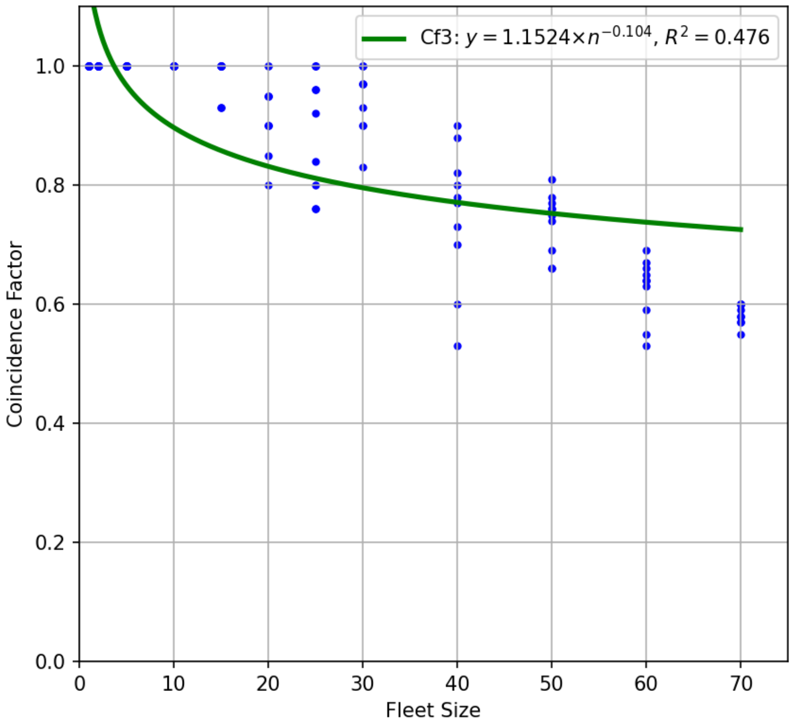


FIGURE 14. Scatter plot of Sc3 with power function trendline.

This new power function trendline for C_{f3} allows us to create a new $C_{f\text{-avg}}$ as seen in equation (9).

$$C_{\mathrm{f-avg}} = \frac{0.9292}{\mathrm{n}^{0.1887}} \qquad (9)$$

Applying this transformation to C_{f3} and calculating a new $C_{f\text{-}avg}$ allows us to extrapolate a new plot where the range of coincidence factors can be seen based and classified in accordance with the type of charging system to be installed in the multi dwelling residential as well as the type of demand.

It is possible to see that, in a theoretical planning stage, if the $C_{f\text{-}avg}$ is used, it can safely account for the $C_{f\,1}$ and C_{f4} by underestimating the electrical current needs, on the other hand, it does not predict correctly the behaviour of C_{f2} and can result in a substantial over dimensioning of the electrical feeder.

The type of demand refers to the optimisation technique use, low demand in case of an optimisation that aims on allocating the minimum electrical current as possible, and high demand in case of an optimisation that aims on allocating the maximum electrical current available for charging. This can be seen in the following FIGURE 15.

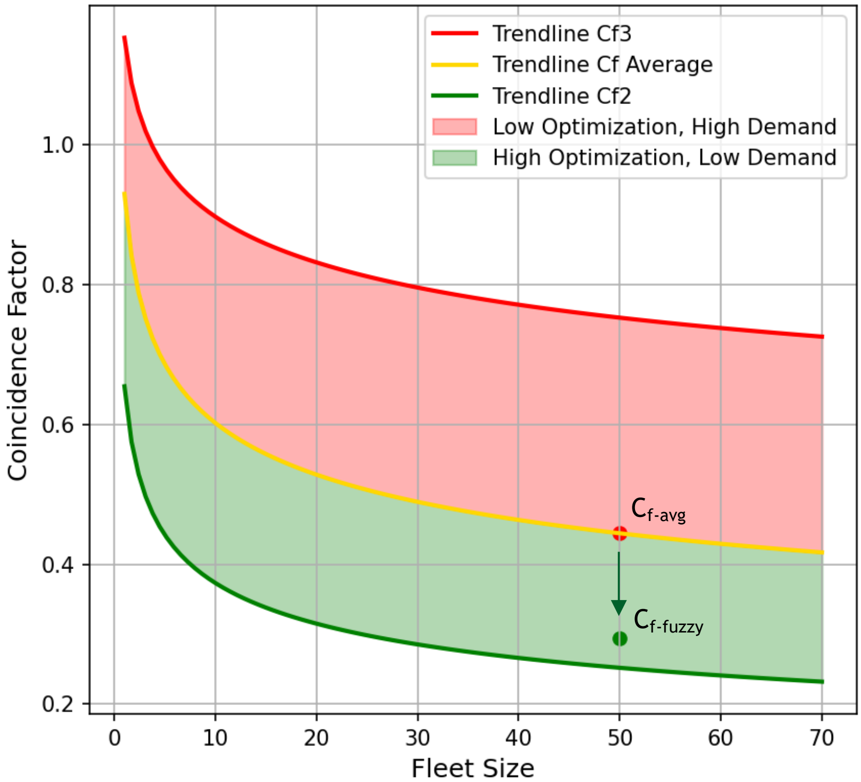


FIGURE 15. Coincidence factor graph with coincidence factor plotted and corrected via the fuzzy logic inference model.

Now, it is possible to create a fuzzy logic inference method (Annex A) that, using as input the $S_{c.avg}$, plus the demand and optimisation levels, outputs a $C_{f\text{-}fuzzy}$ for the fleet size being considered and the characteristics of the charging equipment. For example, if we consider a fleet size of 35, we have a $C_{f\text{-}avg}$ of 0.475, but with a demand level of 2 (in a scale of 1 to 10) and an optimisation level of 9, it is corrected to 0.29, as seen in FIGURE 15.

3.5 - Electrical Feeder’s Analysis

To validate the calculated C_{f1} and C_{f2} it there is a need to cross-check with the commonly used feeders in these types of electrical installations. For the purposes of this study it was considered the commonly used sheeted copper and aluminium cables with the following characteristics as seen in TABLE XVIII.

TABLE XVIII
Maximum allowed current by feeder conductor section and material, isolation material and installation type [84].

Feeder [active conductor section in mm^2	Maximum allowed current - I_z		
	Copper - PVC	Aluminium - PVC	Type
2.5	25 A	19.5 A	Multi conductor
4	34 A	26 A	
6	43 A	33 A	
10	60 A	46 A	
16	80 A	61 A	
25	131 A	84 A	Single conductor
35	137 A	105 A	
50	167 A	128 A	
70	216 A	166 A	
95	264 A	203 A	
120	308 A	237 A	
150	356 A	274 A	
185	409 A	315 A	

TABLE XIX for S_c equal to 1, where the maximum observed current is considered.

TABLE XIX
Feeder phase section per fleet size considering no C_f.

EV Fleet Size	C_f = 1	Per phase	Feeder [Cu]
n = 1	16 A	16 A *	2.5 mm^2
2	32 A	16 A *	2.5 mm^2
5	80 A	26.67 A	4 mm^2
10	160 A	53.33 A	10 mm^2
15	240 A	80.00 A	16 mm^2
20	320 A	106.67 A	25 mm^2
25	400 A	133.33 A	50 mm^2
30	480 A	160.00 A	50 mm^2
40	640 A	213.33 A	70 mm^2
50	800 A	266.67 A	120 mm^2
60	960 A	320.00 A	150 mm^2
70	1120 A	373.33 A	185 mm^2

(*) 16 A it is the max-min charging rate per phase per EV.

By using the previously extrapolated equations for the C_f calculations it is possible to crosscheck the electrical feeder that each sized fleet would require to be installed to safely charge the EVs.

TABLE XX presents the crosscheck for C_{f1} and C_{f2} and the respective suitable feeder. The section of feeder used depends on several factors, because the maximum allowed current can change for the same feeder depending on its length, type of installation, neighbouring feeders.

For this case we disregard the impact of voltage drop as it varies from installation to installation and considered an isolated deployment in perforated cable trays.

TABLE XX
Feeder phase section per fleet size considering C_{f1} and C_{f2}.

EV Fleet Size	C_f = 1	Current Calculation			Feeder [Cu]
		$C_{f\,1}$	Maximum Current	Per Phase	
n = 1	16 A	1 *	14.4 A	16 A **	2.5 mm^2
2	32 A	0.78	25.1	16 **	2.5
5	80 A	0.65	52.1	17.37	2.5
10	160 A	0.57	90.5	30.17	4
15	240 A	0.52	125.0	41.7	6
20	320 A	0.49	157.2	52.4	10
25	400 A	0.47	187.8	62.6	16
30	480 A	0.45	217.1	72.37	16
40	640 A	0.43	273.1	91.03	25
50	800 A	0.41	326.2	108.73	25
60	960 A	0.39	377.3	125.76	25
70	1120 A	0.38	426.6	142.2	50
EV Fleet Size	**C_f = 1**	**Current Calculation**			**Feeder [Cu]**
		C_{f2}	**Maximum Current**	**Per Phase**	
n = 1	16 A	1*	10.5 A	16 A **	2.5 mm^2
2	32 A	0.56	17.9	16 **	2.5
5	80 A	0.46	36.5	16 **	2.5
10	160 A	0.39	62.5	20.83	2.5
15	240 A	0.36	85.6	28.53	4
20	320 A	0.33	107.0	35.67	6
25	400 A	0.32	127.2	42.40	6
30	480 A	0.31	146.6	48.87	10
40	640 A	0.29	183.2	61.07	16
50	800 A	0.27	217.9	72.63	16
60	960 A	0.26	251.0	83.67	25
70	1120 A	0.25	282.9	94.30	25

(*) If n equals 1 then C_f must be 1 as well as there is always the likelihood of 1 EV charging at its maximum rate.
(**) 16 A it is the max-min charging rate per phase per EV.

Continuing, FIGURE 16 presents the 2D plot of fleet size versus maximum current, under layered by the recommended feeder for C_{f1}, and FIGURE 17 presents the 3D plot with the variation of the coincidence factor S_{c1}, in relation with fleet size and maximum current consumed.

So, if no C_f was applied, in other words an C_f equal to 1, a fleet of 25 EVs would already require a feeder with a section of 50mm^2 per phase, contrarily by applying either C_{f1} this feeder would already manage a fleet of 70 EVs, and in the case of C_{f2} the maximum required feeder would have a section of 25mm^2, this is a reduction of 86%, lastly, this crosscheck does not take into consideration voltage drops in the feeder associated with its length. The main take way is that groups of C_f values will fall under the same feeder.

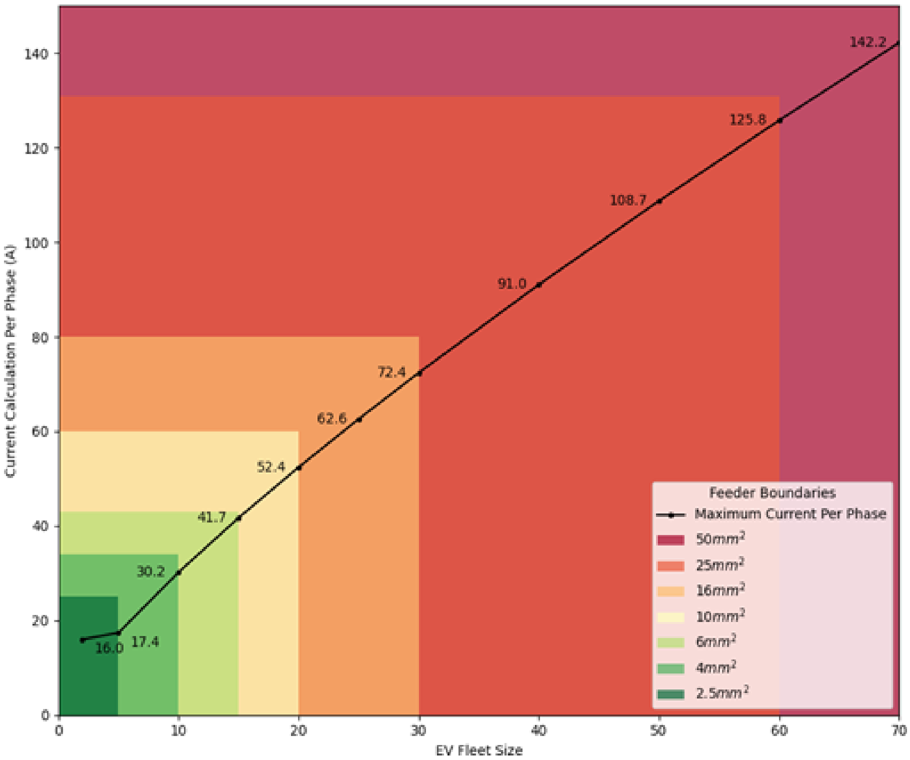


FIGURE 16. Feeder boundaries per fleet size and maximum allocated current with C_{f1}.

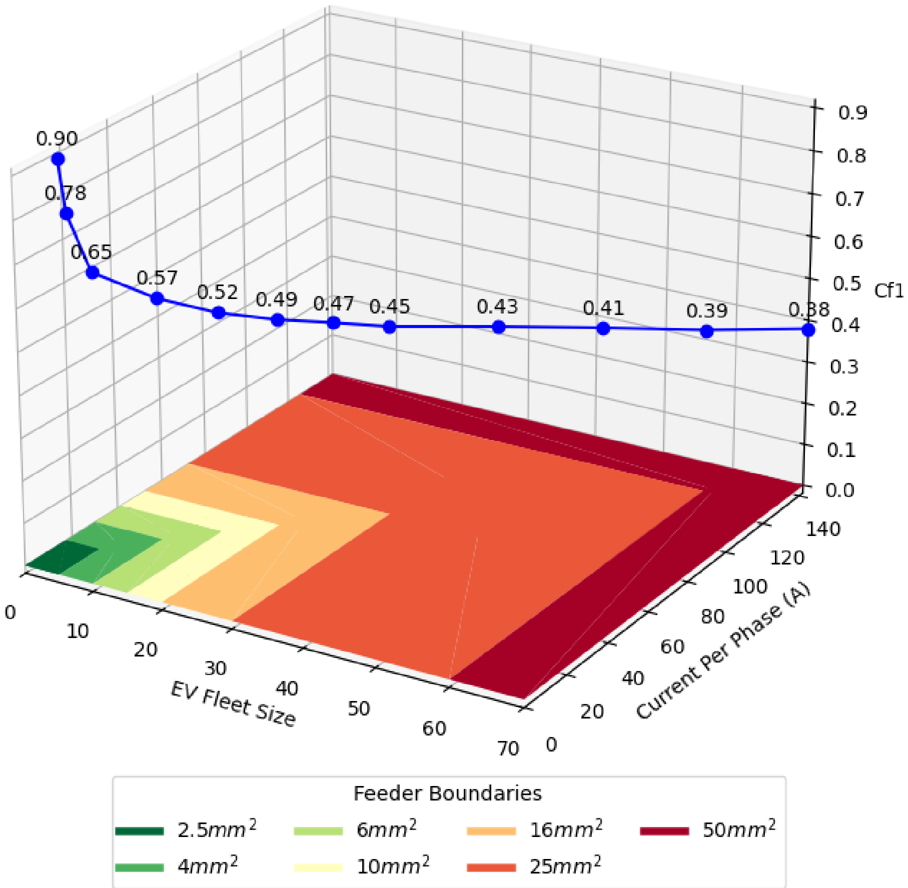


FIGURE 17. Feeder boundaries per EV fleet size, maximum allocated current and C_{f1}.

3.6 - Circuit Breakers and Fuses Analysis

In TABLE XXI the most common circuit breaker values for domestic use are presented, and TABLE XXII presents the match up of those circuit breakers with the respective fleet size.

TABLE XXI
Feeder section per stipulated current and conventional triggering current for each circuit breaker [84].

I_n - Stipulated current	I_2 - Conventional triggering current	Protects up to section* [Cu]
10 A	14 A	All
16 A	23 A	All
20 A	29 A	> 2.5 mm^2
25 A	36 A	> 4 mm^2
32 A	46 A	> 6 mm^2
40 A	58 A	> 10 mm^2
50 A	72 A	> 16 mm^2
63 A	91 A	> 25 mm^2
80 A	116 A	> 25 mm^2
100 A	145 A	> 50 mm^2
125 A	181 A	> 70 mm^2

(*) Not including the specified section.

So, for example, the way to interpret this table is: a circuit breaker with a stipulated current, I_n, of 32A and an associated conventional triggering current, I_2, of 46A will ensure that feeders larger than 6 mm^2 are protected against overcurrent, but a feeder of 6 mm^2 is not protected.

TABLE XXII (PART I)
Circuit breaker phase section per fleet size considering C_{f1} and C_{f2}.

	EV Fleet Size	Current for C_f = 1	C_{f1}	Maximum Current	Per Phase - I_B	Circuit Breaker I_n
C_{f1}	n = 1	16 A	1 *	14.4 A	16 A **	16 A
	2	32 A	0.78	25.1 A	16 A **	16 A
	5	80 A	0.65	52.1 A	17.4 A	16 A
	10	160 A	0.57	90.5 A	30.4 A	32 A
	15	240 A	0.52	125.0 A	41.7 A	50 A
	20	320 A	0.49	157.2 A	52.4 A	63 A
	25	400 A	0.47	187.8 A	62.6 A	63 A ***
	30	480 A	0.45	217.1 A	72.4 A	80 A
	40	640 A	0.43	273.1 A	91.0 A	100 A
	50	800 A	0.41	326.2 A	108.7 A	125 A
	60	960 A	0.39	377.3 A	125.8 A	****
	70	1120 A	0.38	426.6 A	142.2 A	

(*) If n equals 1 then C_f must be 1 as well as there is always the likelihood of 1 EV charging at its maximum rate.
(**) 16 A it is the max-min charging rate per phase per EV.
(***) Close to I_B but within regulations.
(****) For I_n above 125A several manufactures have the industrial line.

TABLE XII (PART II)
Circuit breaker phase section per fleet size considering C_{f2}.

	EV Fleet Size	Current for C_f = 1	C_{f2}	Maximum Current	Per Phase - I_B	Circuit Breaker I_n
C_{f2}	n = 1	16 A	1*	10.5 A	16 A **	20 A
	2	32 A	0.56	17.9 A	16 A **	20 A
	5	80 A	0.46	36.5 A	16 A **	20 A
	10	160 A	0.39	62.5 A	20.83 A	25 A
	15	240 A	0.36	85.6 A	28.53 A	32 A
	20	320 A	0.33	107.0 A	35.67 A	40 A
	25	400 A	0.32	127.2 A	42.40 A	50 A
	30	480 A	0.31	146.6 A	48.87 A	50 A ***
	40	640 A	0.29	183.2 A	61.07 A	63 A ***
	50	800 A	0.27	217.9 A	72.63 A	80 A
	60	960 A	0.26	251.0 A	83.67 A	100 A
	70	1120 A	0.25	282.9 A	94.30 A	100 A

(*) If n equals 1 then C_f must be 1 as well as there is always the likelihood of 1 EV charging at its maximum rate.
(**) 16 A it is the max-min charging rate per phase per EV.
(***) Close to I_B but within regulations.

3.7 - Results Discussion

Again, this crosscheck is simplified as it does not consider other factors such as downstream feeder length, upstream circuits breakers I_n, other feeder and isolator materials, amongst other variables, nonetheless it was carried out considering a common scenario where feeders lengths are short, and they are installed in perforated cable trays.

The goal is to determine which are the limiting factors in C_f stipulation, feeder size or circuit breaker calibre.

To do that TABLE XXIII contains the necessary information that allows us to cross check which component, if any limit the applicable C_{f1} or C_{f2}, to do it was necessary to use the following equations (10) and (11).

$$I_B \leq I_n \leq I_Z \tag{10}$$

$$I_2 \leq 1.45 \times I_Z \tag{11}$$

Where, I_n and I_2 are characteristics of the chosen circuit breaker and can be consulted in TABLE XXI, I_B is the maximum measured current per phase for each EV fleet size, available in TABLE XXII, and I_Z is the maximum allowed current for each phase of the electrical feeder, available in TABLE XXIII.

So, considering TABLE XXIII we can observe that in case of C_{f1} to comply with equations (10) and (11) the scenario of 15 and 20 EVs required an incrementation of feeder to the next feeder size, for example in the case of fleet size n = 15, the feeder chosen first was 6 mm^2 however to comply with the regulations it was necessary to increase to 10 mm^2, this means that for an I_B of 41.7A we would install a feeder that can withstand currents up to 60 A.

TABLE XXIII
Feeder section and circuit breaker stipulated current per EV fleet size considering S_{c1} and S_{c2}.

C_f	N	I_n [A]	Feeder [mm^2]	I_z [A]	Increment
1	1	20	2.5 * or 4	25	0
	2			25	0
	5			25	0
	10	32	4	34	0
	15	50	10	60	+1
	20	63	16	80	+1
	25			80	0
	30	80		80	0
	40	100	25	131	0
	50	125		131	0
	60	160**	70	131	+2
	70			167	+1
C_f	N	I_n [A]	Feeder [mm^2]	I_z [A]	Increment
2	1	20	2.5 * or 4	25	0
	2			25	0
	5			25	0
	10	25		25	0
	15	32	4	34	0
	20	40	6	43	0
	25	50	10	60	+1
	30			60	0
	40	63	16	80	0
	50	80		80	0
	60	100	25	131	0
	70			131	0

(*) The 2.5 mm^2 needs additional calculations to be validated.
(**) Fuse instead of circuit breaker.

This holds true for all cases of EV fleet size N that have an increment of "+ n". Coupling this analysis with the fuzzy inference model we can condense all information in FIGURE 18.

Considering these findings, we can state that in some cases the feeder size limits the application of the C_f as an increase in section is necessary to meet the regulations and circuit breaker demands, also it is possible to observe that for 12 different fleet sizes we have 8 circuit breaker calibres and 6 feeder sections, so due to the smaller granularity in feeder size they also act as a limiting factor.

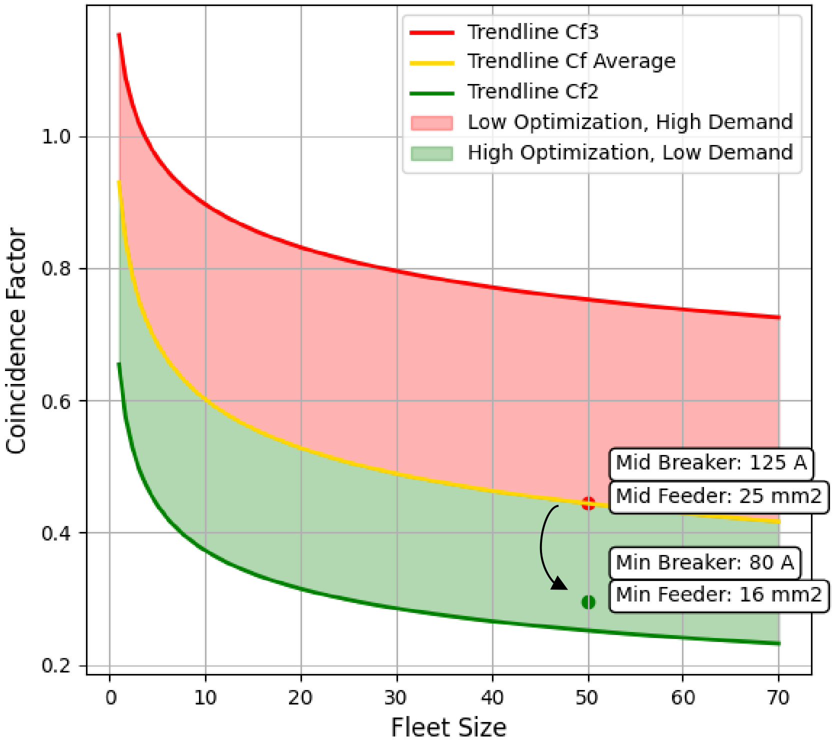


FIGURE 18. Simultaneity factor correction via fuzzy logic inference model, for a fleet of 20 EVs, with high optimisation and low demand.

3.8 - Validation and Comparative analysis of Results

To validate these findings, we compared them to coincidence factors extracted from a publicly available dataset of EV charging. The dataset used in this comparison is the "Data set of a Norwegian energy community", available on Mendeley Data [110]. It characterises electric vehicle charging behaviour by specifying the statistical distributions of charging start probabilities and charging durations, which are provided for all seasons, weekdays and weekends, and 4h groupings for an energy community.

Each entry is defined using an exponential distribution for both start probability and charging duration. The parameters include λ_{start} and $\lambda_{duration}$, as well as minimum and maximum values used to bound each variable in the simulations as demonstrated in the following TABLE XXIV.

TABLE XXIV
Dataset used to estimate a baseline for the Coincidence Factors based on real data.

			charging start probability [%]			charging duration [h]		
		Hour	λ_{start}	maximum	minimum	$\lambda_{duration}$	maximum	minimum
Spring	weekday	00-04	2,08	7,69	0	2,34	14,3	0
		04-08	2,47	7,69	0	2,98	105,74	0
		08-12	2,43	14,29	0	2,71	29,87	0
		12-16	2,04	30,77	0	2,33	44,7	0
		16-20	1,62	38,46	0	2,01	69,94	0
		20-24	1,37	23,08	0	2,02	61,46	0

For the baseline estimation we focused on weekdays during the spring months, considering a 12-hour time window between 20h00 and 08h00, and an average charging duration of 6h for each EV.

In this case we defined the coincidence factor as follows in equation (12).

$$C_{(f-\text{baseline})} = \frac{E[n_{\max}]}{N} \tag{12}$$

Where n_{max} is the maximum number of EVs charging simultaneously during the observed timed window and E is the expectation across multiple simulations. This coincidence factor reflects the peak simultaneous number of EVs charging relative to the total fleet.

To estimate the $C_{f\text{-baseline}}$ we used a Monte Carlo simulation approach, and for each simulation we selected a fleet size N.

Regarding the simulation we repeated for each trial and for each EV the following steps: generate a start time t_s; define a duration of charging period D, the charging duration for an EV. It is modelled as a continuous random variable that follows an exponential distribution with a mean of 6 hours; compute the charging interval $[t_{s,}\ t_s + D]$. Then we collect all the intervals and compute the maximum number of overlapping intervals at any moment which provides us the n_{max}. Finally, we compute the $C_{f\text{-baseline}}$ over all trials.

The likelihood of an EV beginning its charging period is based on the information provided by the previously mentioned dataset.

Two different distributions were considered for the charging start time $t_s \in [20, 32]$ (i.e., 20h00 to 08h00): an uniform distribution that assumes users are equally likely to plug in at any given time, which reflects and assumption of perfect temporal randomness; and an exponential distribution, that assumes a more realistic model, assuming users are more likely to plug in soon after returning home, typically in the evening, capturing the behavioural skew toward earlier evening hours and reflecting user behaviour without managed charging systems.

The $C_{f\text{-baseline}}$ results obtained for different sized fleets are presented in the following FIGURE 19.

We note that the uniform distribution underestimates coincidence factors, especially in small to medium fleets, as it spread charging evenly. The exponential start times produces higher coincidence factors, indicating more overlapping charging activities during early evening hours.

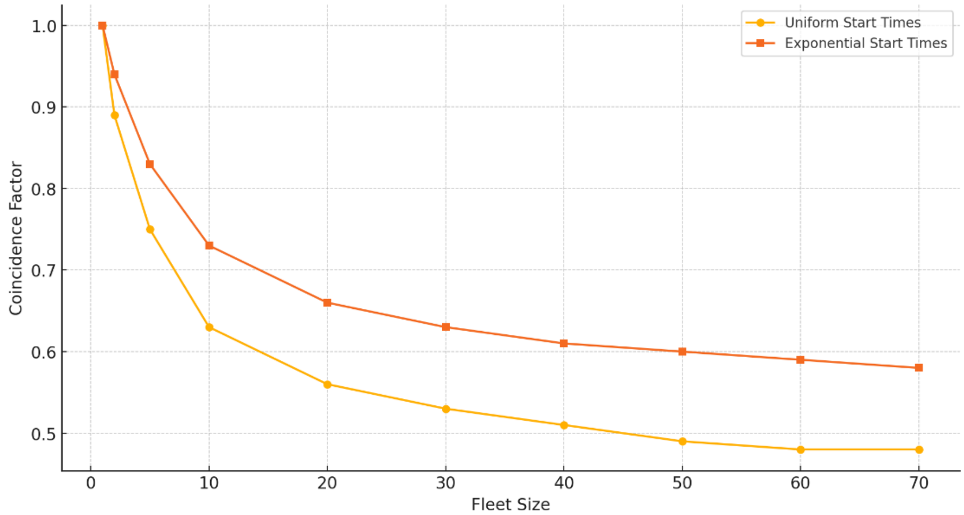


FIGURE 19. Coincidence factor versus fleet size and uniform versus exponential distribution for start times between 20h00 and 08h00.

These differences, observed, are non-trivial for grid planning, at 50 EVs, the difference is close to 22% higher coincidence under exponential start times.

Thus, using more realistic distributions aligned with empirical behaviour can significantly improve accuracy of demand forecasts.

However, this baseline does not consider that when multiple EVs are charging they can do so at different charging rates, drawing more or less power depending on the EV needs, resulting in a dummy charging assumption that can over penalise the coincidence factors when we look at consumed power instead of the number of EVs charging simultaneously. The nuances and empirical behaviours, as well as optimised charging mechanisms were considered during these simulations.

In the following TABLE XXV we can observe how the simulated coincidence factors are compared to the initially calculated baseline coincidence factors.

TABLE XXV
Comparison between simulated coincidence factors and baseline coincidence factors.

Fleet Size	Baseline Uniform	Baseline Exponential	C_{f2} (lower bound)	C_{f3} (higher bound)
1	1.00	1.00	1.00	1.00
2	0.89	0.94	0,560	1.00
5	0.75	0.83	0,456	0,975
10	0.63	0.73	0,391	0,907
20	0.56	0.66	0,334	0,844
30	0.53	0.63	0,305	0,809
40	0.51	0.61	0,286	0,785
50	0.49	0.60	0,272	0,767
60	0.48	0.59	0,261	0,753
70	0.48	0.58	0,253	0,741

It is possible to observe that C_{f3}, which aims to mimic a high demand low optimisation scenario closely mimics the exponential baseline in the small to medium fleet size and from the medium to large fleet size it overestimates the coincidence factors which was expected based on previously explained assumptions.

On the other hand, C_{f2} which resides at the other extreme of the spectrum, meaning high levels of optimisation and low demand, always underestimates the coincidence factor when compared to the baselines.

Meaning that the baselines are encompassed by both simulated coincidence factors as intended, leaving room for the decision makers and planners.

As previously mentioned, other studies also aimed at estimating coincidence factors for residential EV charging, utilising other methodologies and assumptions. To validate these results, we cross-checked the findings with [111] in the following FIGURE 20.

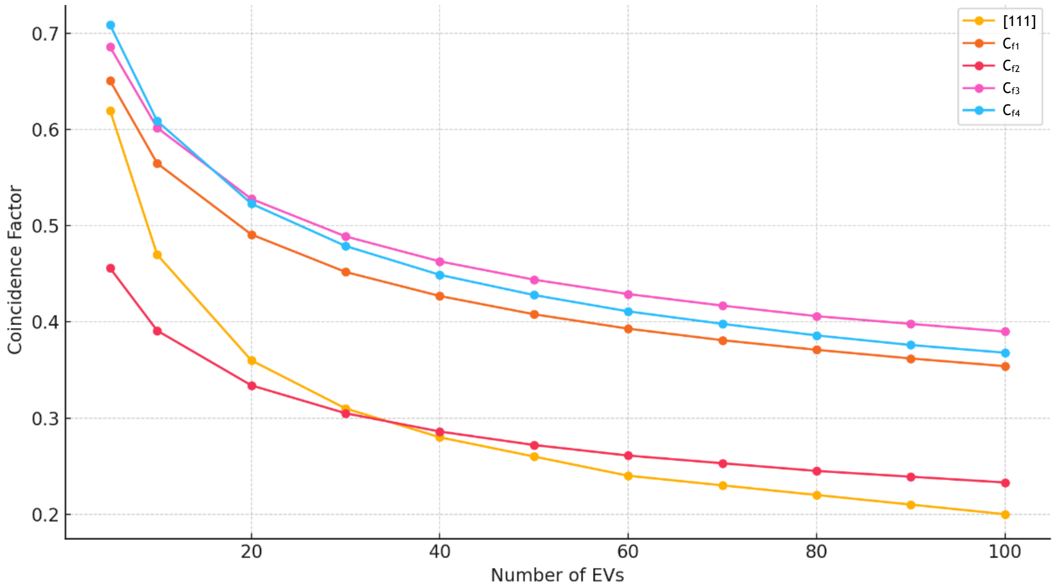


FIGURE 20. Comparative of current study Coincidence Factors with literature Coincidence Factors.

As it is possible to observe in FIGURE 20, C_{f1}, C_{f3} and C_{f4} trends above the coincidence factors of [111], while C_{f2} is more closely related to it, so it is safe to assume that , C_{f1}, C_{f3} and C_{f4} are indeed more conservative but can be modulated to closer match the referenced coincidence factor by using the fuzzy correction factor.

A recent study by [112] analysed residential EV charging behaviours in Norway and revealed C_f values typically below 15% for uncontrolled (normal) charging and as high as 32% during early morning hours underprice-driven smart charging. These empirical observations confirm how behavioural incentives, and synchronisation, can shift but not eliminate high simultaneity in charging, often introducing new peaks at nontraditional hours (e.g., early morning instead of evening). By contrast, the methodology, although it accounts for, it is not derived from behavioural datasets but rather enables planners to correct estimated C_f's ex ante, using fuzzy rules linked to optimisation levels and fleet demand characteristics. While the Norwegian study captures variability through observational data, it does not generalise well to planning stages in contexts lacking local empirical profiles. The fuzzy-inference model fills this gap by offering a behavioural proxy that adapts to usage assumptions, thereby reducing reliance on empirical data collection. Moreover, the work demonstrates that mass smart charging has the potential to generate new critical peaks. This insight serves to emphasise the importance of incorporating coincidence uncertainty directly into infrastructure planning, a contribution that is precisely what this work aims to deliver.

In [113] present a thorough investigation of large-scale battery-electric vehicle (BEV) charging behaviour under varying ambient temperatures and available charging power levels. The study emphasised the influence of these factors on charging load profiles and coincidence factors across different locations. Their findings reveal that coincidence increases under colder conditions and lower charging power, with the highest value being observed in workplaces, hotels, and homes. However, even in extreme cases (e.g., −20 °C), coincidence factors remained below 0.44, reflecting the general diversity of charging behaviour across large fleets. Additionally, they observe that greater available power leads to sharper peaks and deeper valleys, while lower power flattens the load, confirming the sensitivity of demand profiles to infrastructure parameters. It is possible to observe that, whilst the overall coincidence remains moderate in large populations, local deviations can be substantial in smaller groups of vehicles, where stochastic effects dominate.

The present study builds upon this insight by proposing an innovative methodology for infrastructure planners, particularly in contexts where small to medium-scale residential EV adoption may lead to sizing uncertainty. In contrast to the simulation of charging events based on environmental parameters, the approach focuses on the analysis of behavioural patterns. Utilising a fuzzy inference system, we correct coincidence factors based on system-level assumptions, notably charging system optimisation level and fleet energy demand. In contrast to the author's data-driven modelling, which is ideally suited for evaluating operational impacts under defined scenarios, the method aims to support early-stage design by providing planners with an adaptive tool that generalises across varying usage intensities and control sophistication. Together, both approaches contribute to a richer understanding of how charging simultaneity influences infrastructure design, one through empirical modelling, the other through flexible inference for planning under uncertainty.

The authors of [114] analyse EV charging simultaneity by segmenting users into energy demand buckets and simulating their interaction with different charger power levels across rural, urban, and semi-urban settings. Their findings show that vehicles with higher daily energy demands tend to exhibit greater simultaneity factors (SFs), particularly in cases where mid-range chargers (3.5–7 kW) are utilised. The coincidence factors (C_f) values in such cases frequently range from 0.38 to 0.58. While their analysis does not vary fleet size directly, the patterns they report reflect the fundamental relationship between behavioural clustering and charging overlap, a phenomenon the present study also addresses, but from a different modelling perspective.

Rather than disaggregating behaviour by individual energy consumption, the fuzzy-based methodology proposed here estimates CF as a function of fleet size and system optimisation level. This approach generalises charging diversity through aggregate rules, allowing planners to capture behavioural uncertainty and infrastructure stress without detailed energy-use profiles. Although indexed differently, both models reflect the inverse relationship between user diversity and peak simultaneity with the C_{f1} values decreasing from 1.0 (for 1 EV) to 0.38 (for 70 EVs), comparable in range and trend to the author's CFs across energy demand segments. Thus, while the methodological frames differ, the results are thematically aligned and reinforce the robustness of modelling approaches that incorporate diversity into EV charging infrastructure planning.

And thus, compared to prior work, the proposed methodology adopts fundamentally different assumptions about the availability of user data, the regularity of charging behaviour, and the adaptability of coincidence estimation. Whereas empirical and simulation-based studies assume well-defined user schedules, location-specific load profiles, or access to charging event data, this study assumes a more typical planning scenario in which such information is unavailable. By embedding key behavioural and control characteristics into a fuzzy inference system, the model enables planners to adjust coincidence factors based on abstract but meaningful parameters like system optimisation level and energy demand scale. This shift in assumptions supports a more resilient and generalisable infrastructure design process, particularly suited to early planning phases or emerging residential developments. While it may lead to more conservative sizing compared to models fine-tuned to historical data, it reduces the risk of undersized infrastructure in novel or evolving usage contexts.

3.9 - Chapter Summary

In this chapter it is demonstrated that applying a coincidence factor (C_f) of 1, when considering electrical installations for electrical vehicles (EVs), results in a costly over dimensioning of the electrical infrastructure. Relying on known driving and charging patterns, as well as EV users' preferences plus industry recommendations, it was possible to achieve two main and important outcomes:

1) The coincidence factor tables for installations up to 70 EVs considering two methodologies of optimised charging.
2) The equations to calculate the C_f considering two methodologies of optimised charging.
3) A fuzzy logic inference model, it is possible to establish a methodology to calculate C_f based on fleet size and optimisation level of the charging system.
4) A methodology to reduce the costs in materials by reducing feeder sections to values up to 7.4 times smaller.

By leveraging on both real life data and synthetic data it was possible to generate a considerable number of simulations that grant a high degree of validity to the presented numbers, however, parameters such as driving and charging patterns, EV capacity and charging currents can and will evolve with time meaning further fine tuning will be required, nonetheless, the methodology is dynamic and can easily account for these uncertainties. Depending on the type of system to be installed, optimised versus non-optimised, the planers can decide which S_c better suits the requirements of the building, by relying on a fuzzy logic inference model.

Chapter 4

Charging Optimisation

In this chapter the study case is presented. These are real buildings situated in the city of Porto, Portugal. For the study, data was gathered regarding building energy consumption and electrical infrastructure.

4.1 - Condominium

This condominium, henceforward C1, is a six-floor building composed of four separate entrances, each with twelve apartments, and one, collective, underground garage.

The electrical power of the building is supplied by a public power transformer situated in the street.

The power transformer connects to a public switching cabinet, situated in the street, via a main feeder, inside the public switching cabinet this main feeder branches into two, and one of these branches connects to a second public switching cabinet inside the garage of condominium.

From the second switching cabinet, inside the garage, the main feeder branches into five different feeders, one for each of the entrance's main low voltage switchboard plus one for the garage switchboard. Knowing this configuration, shown in FIGURE 21, is important and will prove vital when calculating the power available for charging.

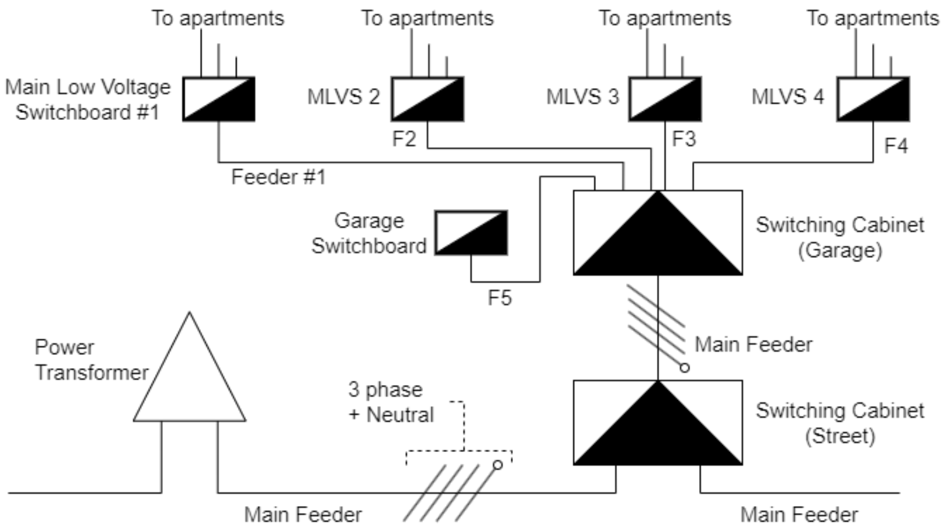


FIGURE 21. Condominium 1 electrical infrastructure topography.

For the sake of simplicity, and figure readability, the energy meters were omitted, and it is implied that feeders one through five have three phases plus neutral wire and the feeders to the apartments have one phase plus neutral and protection.

By analysing FIGURE 21 it is possible to see that, in the current configuration, the maximum electrical power that can be supplied to the EVs in the garage will be imposed by feeder number 5 (F5).

4.2 - Research Questions and Hypotheses

This work has posed a question that has served as the seed for this research. The main purpose was to find a solution that can tackle the gap that the other technical solutions discussed in the introduction, do not address, namely the problem of integrating high number of EVs into existing multi-dwelling buildings.

Question – what is the most cost-effective way to allow high number of EV owners to charge their vehicles in existing multi dwelling buildings? And by asking the question the following hypothesis was formulated.

Hypothesis – it is possible to use the idle power capacity of a residential building to charge high number of EVs during low demand hours, such as the night period, by implementing minor electrical infrastructural changes and applying a centralised, optimised, charging methodology.

This hypothesis can then be broken down into its constituent parts for better understanding. By idle power capacity this work is referring to the difference between the maximum electric power that can be delivered by the building's main feeder and the electric power that is being delivered. This idle power will be at its maximum during low demand hours, like nighttime in the case of residential buildings.

Continuing, by centralised charging methodology this work proposes an optimised algorithm that manages the charging of EVs as a whole fleet instead of each EV managing itself. This is a necessary condition because the electric power being required must be managed in real time to prevent the fleet from demanding more electric power that the main feeder can supply and to ensure smooth transitions when electric power is required by the building, reducing the available or idle power capacity.

Finally, the optimised charging methodology component is proposed as the means to assure that each EV receives exactly the electric power required which in turn will guarantee that the minimum idle power capacity is used and assuring a safety margin of operation for the whole system.

This strategy may raise concerns regarding the potential for unforeseen demand in power consumption by other elements within the building, which could precipitate electrical supply interruptions or, more critically, could inflict damaging effects upon the building's electrical infrastructure.

This is certainly a concern that needs to be addressed at several levels of the proposed solution, and in this case, it was first addressed in the mathematical formulation of the problem.

In a non-optimised methodology, a similar system would simply divide the available power amongst all the connected EVs, as can be seen in equation (13).

$$I_{charging} = \frac{I_{available}}{N_{EVs}} \quad (13)$$

Where:

- $I_{charging}$ is the current attributed to each EV for charging and the decision variable for the optimisation problem.
- $I_{available}$ is the total current available for the charging of the fleet.
- N_{EVs} is the number of EVs connected to the charging system.

Equation (13) would hold true for a non-optimised system with a few caveats such as assuring that $I_{Charging}$ remains between 6A and 16A [115], and it is the base concept for the baseline that will serve as benchmark for the optimised approach.

This type of system would operate close to the limit in situations of high EV demand so; to solve this, an optimised strategy better suits the interests of both EV owners and building inhabitants.

The system will operate in safer conditions in terms of power availability by inferring that EV users will be able to supply two bits of crucial information to the charging management system, 1) how much charge do they require, and 2) at what time will they unplug the EV or how long will the EV be plugged to the charger.

This means we can add two variables to the system: State of Charge desired ($SoC_{Desired}$), which is the desired SoC at the end of the charging period, and $T_{available}$, which is the time available to achieve the $SoC_{Desired}$.

So, there is no need to charge any given EV at the maximum available current rate, there is only the need to charge at the current rate that will guarantee that the $SoC_{Desired}$ is achieved at the unplugging time.

This is a minimisation problem, where the aim is to minimise the time gap between the time at which the EV reaches the $SoC_{Desired}$, or $T_{required}$, and the time at which the EV is unplugged, $T_{available}$.

4.3 - Mathematical Formulation

The objective is to allocate the power available in real-time in a way that minimises the expected time to achieve a specific SoC. The mathematical depiction of this concept is represented in equation (14).

$$\min \Delta T = \left| T_{required} - T_{available} \right| \quad (14)$$

From equation (14) it is then possible to extract the $I_{Charging}$ for each EV as it is demonstrated in equation (15) and (16).

$$T_{required} = \frac{SoC_{Desired} - SoC_{Present}}{I_{Charging}} = \frac{SoC_{Required}}{I_{Charging}} \qquad (15)$$

and

$$\min \Delta T = \left| \frac{SoC_{Required}}{I_{Charging}} - T_{available} \right| \qquad (16)$$

Where $SoC_{Present}$ is the EV's present SoC, or when the calculation is made. This holds true if we consider $SoC_{Desired}$ to be in A hour or Ah like it is shown in equation (17)

$$T_{required} = \frac{SoC_{Required}}{I_{Charging}} \rightarrow [h] = \frac{[A.h]}{[A]} \qquad (17)$$

However, because the voltage fluctuations are also considered it is possible to convert $SoC_{Desired}$ to Watts hour just by considering the charging voltage, as shown in equation (18)

$$T_{required} = \frac{SoC_{Required}}{I_{Charging} \times V_{Charging}} \rightarrow [h] = \frac{[W.h]}{[A].[V]} \qquad (18)$$

We can present $I_{charging}$ as the limit presented in equation (19)

$$I_{charging} = \lim_{\Delta T \to 0} \frac{SoC_{Required}}{(T_{available} - \Delta T) \times U_{Charging}} \rightarrow [A] = \frac{[Wh]}{[h] \times [V]} \qquad (19)$$

Mathematically, the optimisation problem to apply to an EV fleet is defined by objective function (OF) in equation (20).

$$\min \sum_{i=1}^{n} \left| \frac{SoC_{Required.i}}{I_{Charging.i} \times U_{Charging}} - T_{available.i} \right| \qquad (20)$$

Where n is the number of EVs that are plugged into the system and charging, $U_{Charging}$ is the grid side voltage at each iteration and $I_{charging}$ is the decision variable for the optimisation problem. Upon closer inspection of (8) the decision variable, $I_{charging}$, is on the denominator of the objective function, so the problem at hand is in fact a non-linear problem.

Although there are several ways to linearise non-linear problems [116][117] this work has kept the formulation in this non-linear form.

Continuing with the mathematical formulation of the problem, we can present the constraints that will apply to the proposed optimisation. The first constraint addresses the charging topologies chosen, which are slow charging, meaning that any EV connected to the system, if it is being charged, must have an allocated $I_{charging}$, no smaller than 6 A and no larger than 16 amps, as portrayed in equation (21) and (22).

$$subject\ to \rightarrow I_{charging}(EV_i) > 6A \qquad (21)$$

and

$$s.t. \rightarrow I_{charging}(EV_i) \leq 16A \qquad (22)$$

As well as fast charging, meaning that any EV connected to the system, if it is being charged, must have an allocated $I_{charging}$, no smaller than 16 A and no larger than 32 amps, as portrayed in equations (23) and (24).

$$s.t. \rightarrow \ I_{charging}(EV_i) > 16A \qquad (23)$$

and

$$s.t. \rightarrow I_{charging}(EV_i) \leq 32A \qquad (24)$$

Where $I_{Charging}(EV_i)$ is the electrical current allocated by the charging manager to EV_i.

To solve the non-linear optimisation minimisation problem this work has implemented a Sequential Least-Square Quadratic Programming optimisation (SLSQP) methodology, a methodology that was originally implemented in a programming package by Dieter Kraft [118][119] and it is a variant of Sequential Quadratic Programming (SQP) that can be consulted in the Appendix A.

Moving forward, if we look back at equation (20), we can re-write as shown in equation (25):

$$f(x) = \frac{\lambda_1}{x \times \lambda_2} - \lambda_3 \qquad (25)$$

Where, f(x) is the Objective Function, x is the $I_{charging}$ variable, and λ_1 to λ_3 are constant.

We can now check, in equation (26), that the proposed objective function is twice differentiable as requested by the methodology.

$$\frac{\partial}{\partial x} f(x) = -\frac{\lambda_1}{\lambda_2 \times x^2} \rightarrow \frac{\partial^2}{\partial^2 x} f(x) = \frac{2\lambda_1}{\lambda_2 \times x^3} \qquad (26)$$

Observe that when x is equal to 0, the first and second derivatives are undefined in the neighbourhood of 0, this could pose a problem to the SLSQP methodology as it requires that the objective function and constraint functions be at least twice continuously differentiable (i.e., have continuous first and second derivatives) however this will not be the case as x, or charging, is only evaluated between 6A and 16A, as presented in Figure 22.

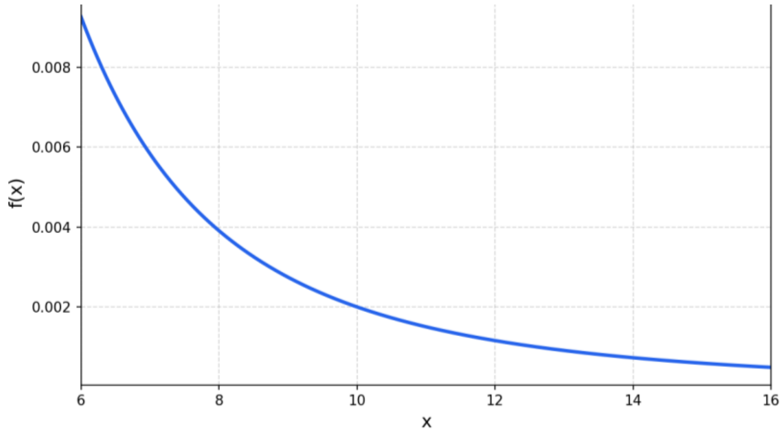


FIGURE 22. Plot of the second derivative of f(x) evaluated between x=6 and x=16, demonstrating its continuity in the required interval of solutions.

Equation (27) holds the constraint that addresses the power or current limitations imposed by the building's main feeder.

$$s.t. \quad \sum_{i=1}^{n} I_{Charging.EV_i} \leq I_{Max} \times \rho_{Safety}|_{]0.1;1]} \qquad (27)$$

Where I_{Max} is the maximum allowed current in the building's main feeder, considering the three phases and ρ_{Safety} is a safety coefficient.

Equations (28) and (29) present the constraints that were imposed for the EV parameters and ensure the correct functioning of the methodology.

$$s.t. \quad EV_i^{SoC} \leq EV_i^{Capacity} \qquad (28)$$

$$s.t. \quad EV_i^{SoC} < EV_i^{Desired\ SoC} \qquad (29)$$

Where EV_i^{SOC} is the current SoC of EV_i, $EV_i^{Capacity}$ is the maximum EV_i SoC or battery capacity, and $EV_i^{Desired\ SOC}$ is the desired SoC at the end of the charging period.

Finally, the last problem addressed by the mathematical formulation is the EV charging profile. This is important because batteries do not charge linearly all the way from low SoC to high SoC.

Battery models are considerably non-linear, and consequently not convex, however in this case a convex function is sufficient, so we considered a linear model for the battery charging, this model can be consulted in equation (30) as is based on the Coulomb-Counting methodology [120].

$$EV_i^{SoE}|_{(t+1)} = EV_i^{SoE}|_t - \frac{\int_t^{t+1} I(t) \times U(t)dt}{E_n} \rightarrow [-] = [-] - \frac{[Wh]}{[Wh]} \qquad (30)$$

Where SoE is the state of energy of the EV_i battery, when compared to SoC that varies between minimum capacity and maximum capacity, in Ah, the SoE varies between the minimum battery energy and maximum battery energy, in Wh [121], I(t) correspond to $I_{Charging.i}$ in (20), U(t) corresponds to $U_{Charging.i}$ in (20), and E_n is the total available energy of the battery.

The application of a linear model is validated because for now this work only considered charging up to 80% SoC, which is within the linear interval of the battery model [122] as seen in Figure 23, demonstrates how between 0 to ~0.8 (constant current phase) the charge profile is close to linear, tapering at the cv phase (constant voltage), and although this is a simplification it is backed by industry best practices which recommend not charging EV batteries constantly to 100% SoC [123] [124].

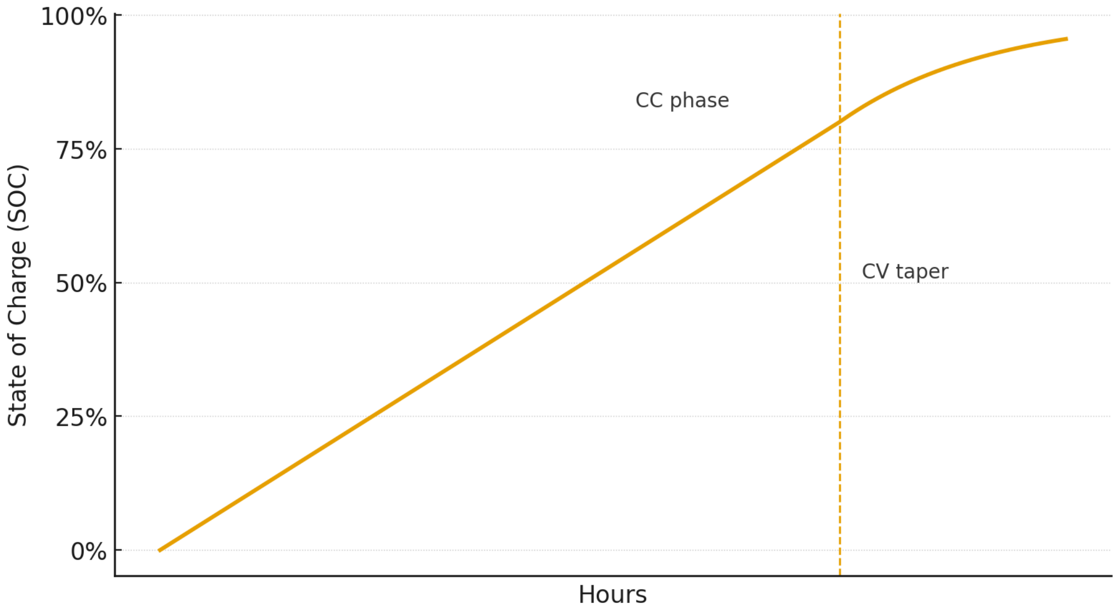


FIGURE 23. Charge curve of a Lithium-ion cell, SoC vs. Charge Time.

This concludes the mathematical formulation of the problem and the foundations of the proposed solution.

This methodology, suitable for the proposed non-linear problem, was originally proposed by Klaus Schittkowski [125] and is extensively described in literature [126][127], applies to non-linear programming (NLP) problems such as the one presented in the following (31) to (33).

$$\min f(x) \tag{31}$$
$$s.t.\ h(x) = 0 \ and \ g(x) \geq 0 \tag{32}$$
$$x \ \in R^n \tag{33}$$

Where $f: R^n \to R, h: R^n \to R^{m_E} \ and \ g: R^n \to R^{m_I}$ are all twice differentiable functions. Equations (34) to (36) present an LSQ problem solved at iteration k:

$$\min \frac{1}{2} \|R^k d^k - q^k\|^2 \tag{34}$$
$$s.t.\ \nabla h^{k^T} d^k + h^k = 0 \tag{35}$$
$$\nabla g^{k^T} d^k + g^k \geq 0 \tag{36}$$

Where, d^k is the search direction to be solved, g^k and h^k are values of constraint functions at iterate x^k, ∇g^k and ∇h^k are current iterate gradients of inequalities and equalities, and R^k and q^k are the least square matrix and observation vector respectively which are updated during optimisation.

Line search is commonly used to assure the global convergence of the SLSQP algorithm, by calculating an increment of length α that provides enough decrease for the merit function $\mathrm{M}(x)$ along the search direction d_k. After α is calculated, the iteration can be updated using equation (37).

$$x^{k+1} \leftarrow x^k + \alpha d^k \quad (37)$$

An often-used condition to ensure that an appropriate decrease for constrained NLP is achieved is the Armijo condition [128] as shown in equation (38).

$$M(x^k + \alpha d^k) \leq M(x^k) + \alpha \rho DM^k(0) \quad (38)$$

Where $DM^k(0)$ corresponds to the directional derivative of $M(x^k + \alpha d^k)$ along d^k at $\alpha = 0$, and $\rho \in [0; 0.5]$ is a constant.

SLSQP iteratively solves a subproblem regarding the current iterate x^k until convergence is achieved. The algorithm maintains feasibility by ensuring that the constraints are satisfied at each iteration. In FIGURE 24, the algorithm is displayed in a flowchart.

4.4 - Simulation Principles

This algorithm, as seen in

Figure 24, was implemented in PyCharm Community Edition 2021.3 and implemented in Python. To tackle the non-linear optimisation objective function the SciPy [129] library was used which allowed for the implementation of the before mentioned SLSQP methodology.

On the other hand, inputs provided by the EV, such as maximum battery capacity, are randomly generated in accordance with industry and market standards [71][72].

The code relies also on other imported libraries such as NumPy [130] and Pandas [131], these libraries are used to manage, and work on the imported data, the random library [132] was used to generate inputs, and finally to present the results the Matplotlib [133] library was used.

Each seven-day simulation takes on average one and a half minutes to complete, however, each optimisation or current allocation, for each three-minute period, takes less than 0.01 seconds to find an optimal solution.

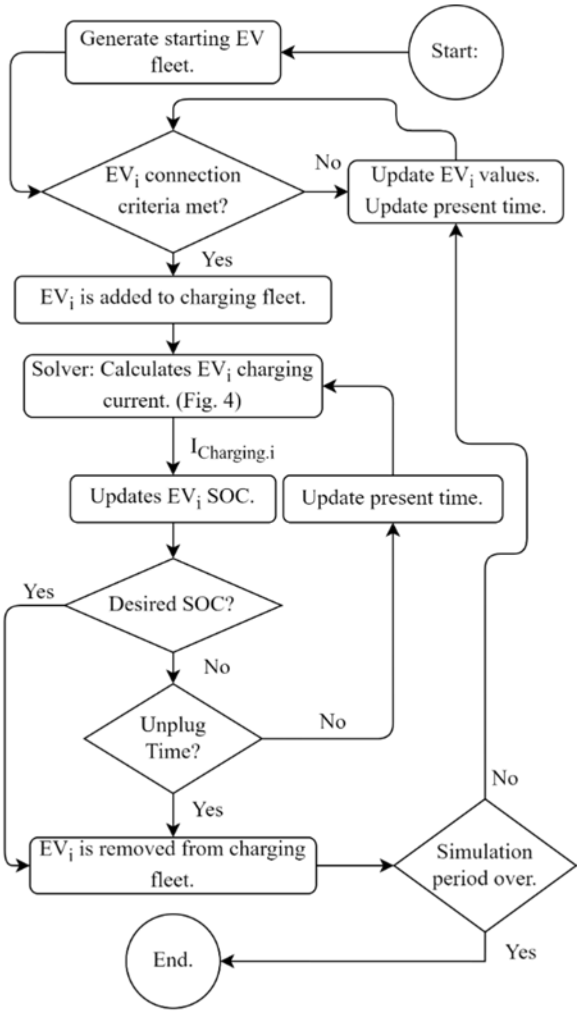


FIGURE 24. Flowchart of charging process for each EV in the building's fleet.

Due to the way the problem is addressed the "Desired SoC?" check will almost always be achieved closely before or after the "Unplug time?" check.

To prove the previously exposed methodology this work carried out a set of simulations using real-life data from a multi residential building situated in Porto, Portugal.

For 6 months a grid analyser was connected to the distribution cabinet that was situated upstream from the building, meaning that the readings were carried out at the level of the building's main feeder. These measurements looked at the voltage and current per phase.

Once the measurements were completed it was necessary to calculate some parameters related to the building's electrical infrastructure that was not available on consultation.

All the following calculations were carried out according to the Portuguese low voltage regulations for electrical infrastructures design [48].

Looking back to FIGURE 24 it is noticeable that a set of inputs are required for each EV, these inputs can either be provided by the user or the EV itself. The inputs that are provided by the user, plug-in time, plug-out time, and desired SoC, daily commute distance, are randomly generated at the beginning or end of each charging period based on commuting studies and common practices described in literature. The inputs mentioned can be consulted in the following table, TABLE XXVI.

TABLE XXVI
Inputs for charging/discharging simulations.

Input	Unit	Origin
Maximum SoE	kWh	Based on market offers.
Present SoE	kWh	Random at start.
Desired SoE	kWh	Based on user needs.
Plug-in time	HH:mm	Based on user behaviour.
Plug-out time	HH:mm	
Available charging time	HH:mm	Calculated.
Daily discharge	km to kWh	Based on commuting needs.
Charging phase	N/A	Sequentially attributed.

For the maximum SOE, for each EV, a battery is randomly generated according to a normal distribution with mean 40 kWh, minimum 20 kWh and maximum 100 kWh. The plug-in and plug-out times are generated in accordance with a normal distribution between 19h00 and 23h30 and between 07h00 and 11h00 respectively. These values were defined based on the information presented in FIGURE 25 and FIGURE 26.

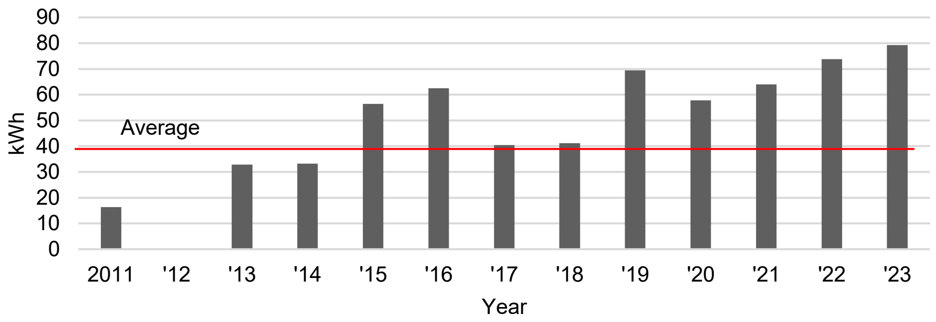


FIGURE 25. Average EV battery maximum energy capacity in kWh throughout the years [71].

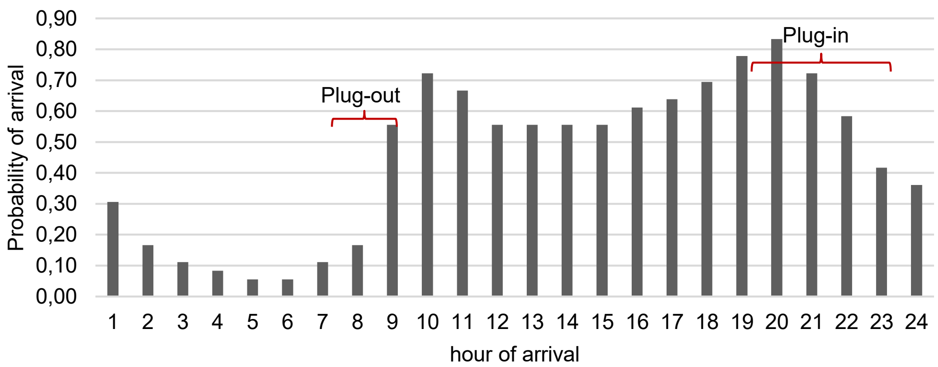


FIGURE 26. Percentage of vehicles commuting during the day [65].

The average commuting distance in the European union is approximately 28.56 km per day [134], and in accordance with [135] the average EV can travel 1km with 0.195 kWh of energy, so we are able to infer that the average daily requirements of an EV user will be around 5.7 kWh.

Based on this, at the end of each charging period or at the end of each day, we can extract both the daily discharge input and the desired SoE input. The first one is calculated based on a normal distribution curve with a mean of 5.7 kWh, the second one is calculated based on the present SoE and the requirements for the next days. The process of generating a desired SoE can be seen in (39) to (41) presented in TABLE XXVII.

TABLE XXVII
Methodology for calculating the next Desired SoE.

Desired SoE calculation.	
Step I: Calculated commuting distance for the next 3 days, each day based on a normal distribution, with a mean of 28.56 km. $\Delta D_T = \Delta D_1 + \Delta D_2 + \Delta D_3, where\ D\ is\ the\ distance.$	(39)
Step II: Convert commuting distance km to kWh. $SOE_{Required}[kWh] = \Delta D_T\ [km] \times 0.195\ [kWh/km]$	(40)
Step III: Calculate Desired SoE. $EV_i^{Desired.SoE} = EV_i^{Present.SoE} + SoE_{Required}$ $s.t.\ EV_i^{Desired.SoE} \leq 0.8 \times EV_i^{Maximum.SoE}$	(41)
Step IV: Return Desired SoE.	

The daily discharge is not calculated based on previously computed commuting distances. This is because the goal of the "Desired SOE" is to simulate a user estimating their needs for the next day, which most times will differ from the actual values verified.

We considered a 3-day-ahead guessing period to account for the user's likelihood of avoiding daily charges, [136] They have identified a specific type of overnight user who, on average, charges their EV between 1.8 and 4.6 times per week. They infer that this type of behaviour closely aligns with residential users.

4.5 - Simulation Results

As previously mentioned, this work used real data measured from a multi-dwelling residential building in Porto, Portugal. The measurements were performed for 6 months with a time-step of 3 minutes. This gave the current work a considerable amount of data to run the simulations. In the first iteration, the simulations were performed for a period of a single day to fine tune the code, once this was done larger periods were considered. First, in Figure 27, Figure 28, and Figure 29 we present the overall use of current within the 7-day simulation period.

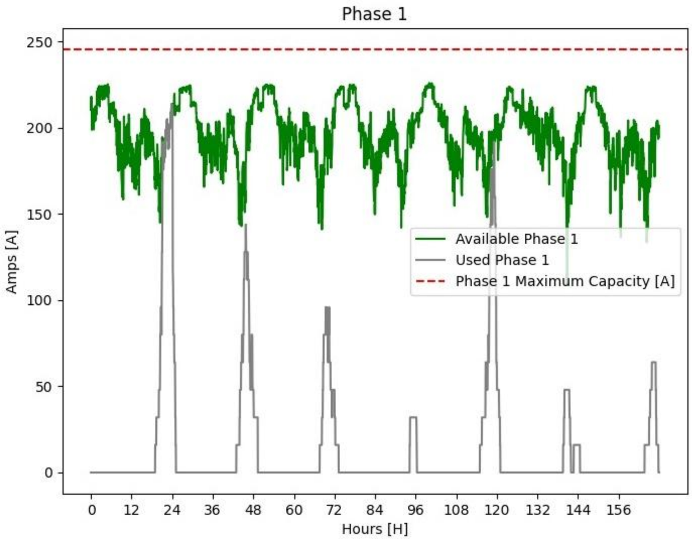


FIGURE 27. Plot of available current (unused by the tenants) and used current (for charging EVs) in phase 1 through a 7-day period.

It is possible to see that the required current for charging only matched the current for charging in the first charging period, or during the first day, this is because of the method used to generate the EV fleet and its parameters, which causes the first day to correspond to the highest demand, this is rapidly mitigated and can be seen in the second day where the demand is much lower because on the first day all the needs were met.

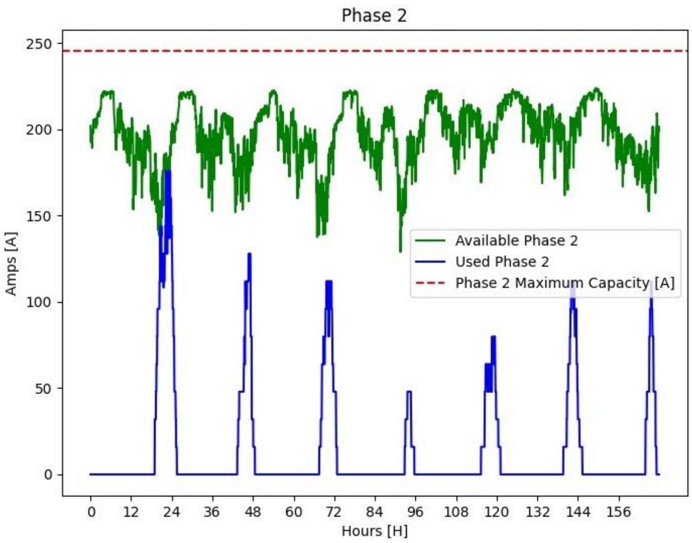


FIGURE 28. Plot of available current (unused by the tenants) and used current (for charging EVs) in phase 2 through a 7-day period.

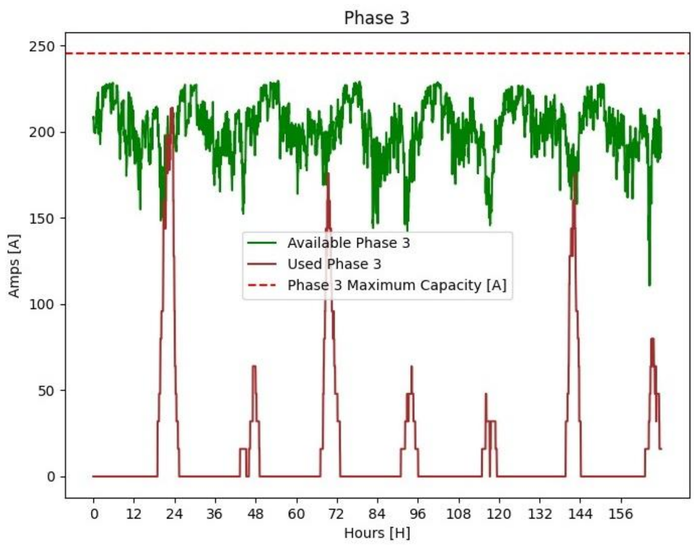


FIGURE 29. Plot of available current (unused by the tenants) and used current (for charging EVs) in phase 3 through a 7-day period.

For this work, the simulations considered a 7-day interval and a total integration of EVs and there are 70 EVs in the building's fleet.

Two validate the proposed solution and other simulations were conducted. In these baseline simulations this work considered 1) a non-optimised centralised system that is connected to the garage switchboard; and 2) a non-optimised centralised system that is connected to the charging switchboard. Regarding the proposed solution, different values of ρ_{Safety}, as seen in equation (21), were tested with the goal of estimating the maximum safety margin applicable in this case that still ensures EV user satisfaction.

Optimised Charging Switchboard

In the following FIGURE 30 and FIGURE 31 the SoC evolution for a couple of EVs during the 7-day period is presented.

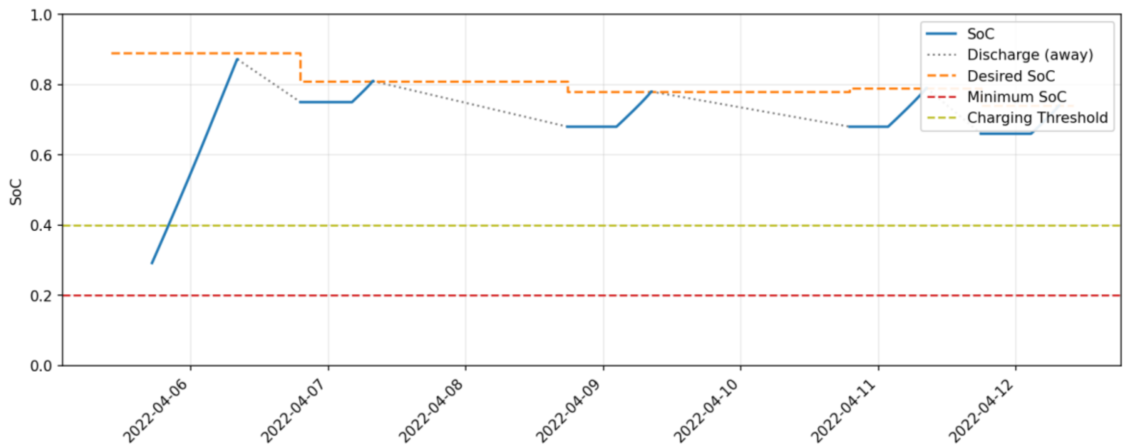


FIGURE 30. Example 1 of EV SoC evolution throughout the 7-day period.

It is possible to see that the Desired SoC is always reached and is updated for the next charging period.

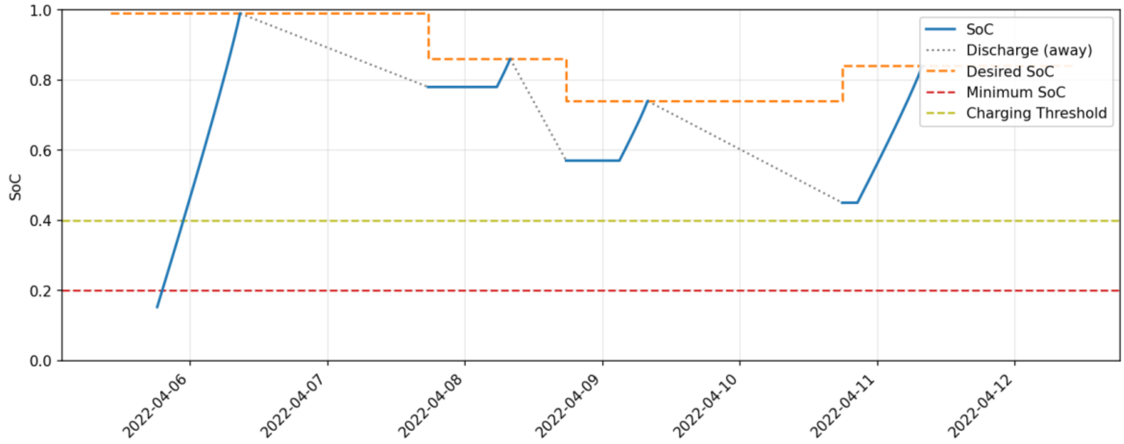


FIGURE 31. Example 2 of EV SoC evolution throughout the 7-day period.

It is possible to see that the Desired SoC is always reached.

In the following Figure 32 and Figure 33 it is possible to analyse the charging current allocated to the previously shown EVs. Although only two examples are shown, the rest of the EVs in the fleet have the same charging success rate, since always achieving the desired SoC when connected to the system.

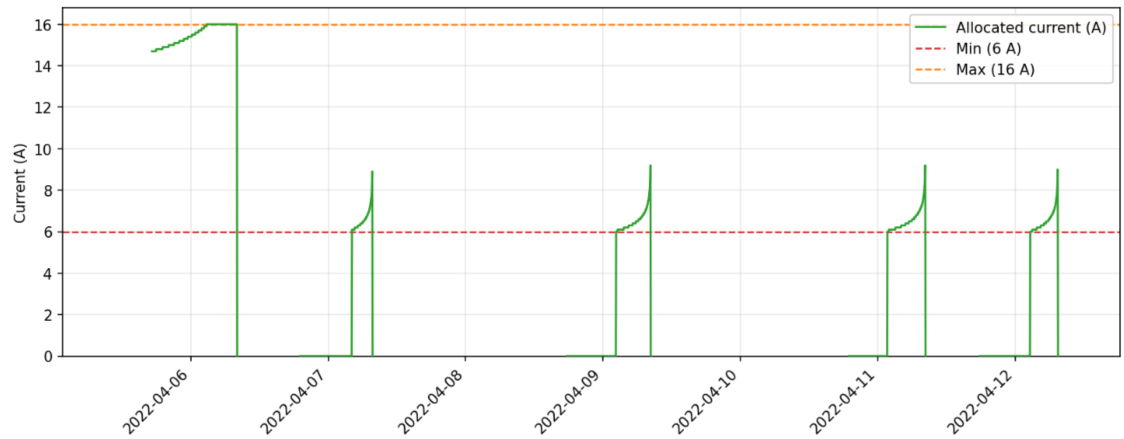


FIGURE 32. Charging current allocated to EV in example 1.

It is possible to notice that during the first charging period the current varied between 16A and 15A, because there was a high demand for power.

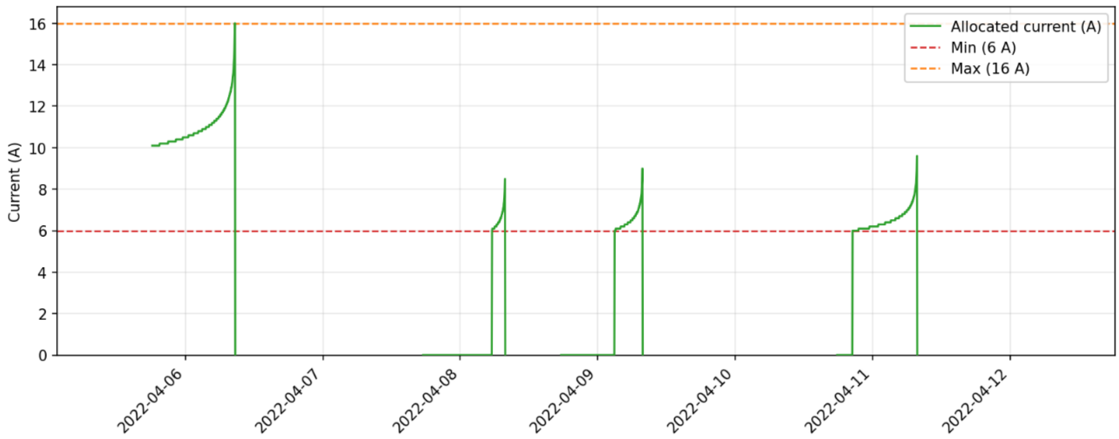


FIGURE 33. Charging current allocated to EV in example 2.

Here the EV was always able to charge at 16A, even on the first day, because it was completely discharged.

Non-Optimised Garage Switchboard

This solution heavily contrasts with the results the system can achieve if it is only given access to the Garage Switchboard as shown in Figure 34. Here we can see that most of the fleet EVs are struggling to even get into the charging queue, resulting in EVs being at 0% SoC for several consecutive days. This is because the Garage Switchboard only allows 20A per phase, so in the best-case scenario it could charge three EVs per phase at 6A, for 9 EVs, if we disregard the other previously mentioned utilities connected to this switchboard.

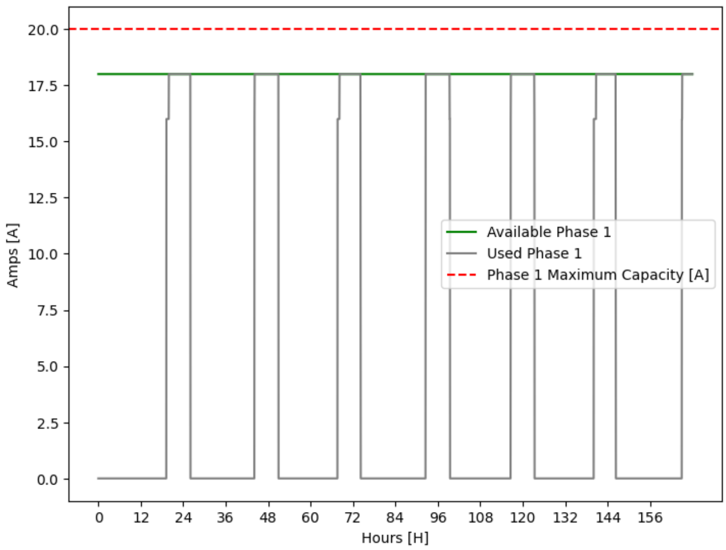


FIGURE 34. Phase 1 plot of available current in the Garage Switchboard and the used current.

After analysing Figure 34, it becomes evident that in this case the available current is always used up completely, and even by doing this it is not possible to maintain an acceptable number of EVs charged as their user's desire. This becomes even more evident after analysing Figure 35 and Figure 36.

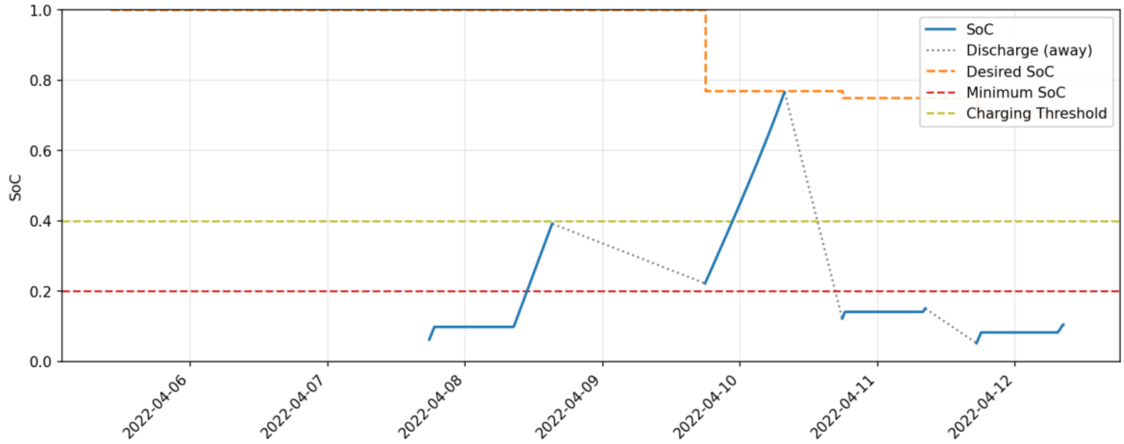


FIGURE 35. Example 3 of EV that was connected to the charging system via the Garage Switchboard.

It is possible to see that the EV could not charge more than once during the whole week.

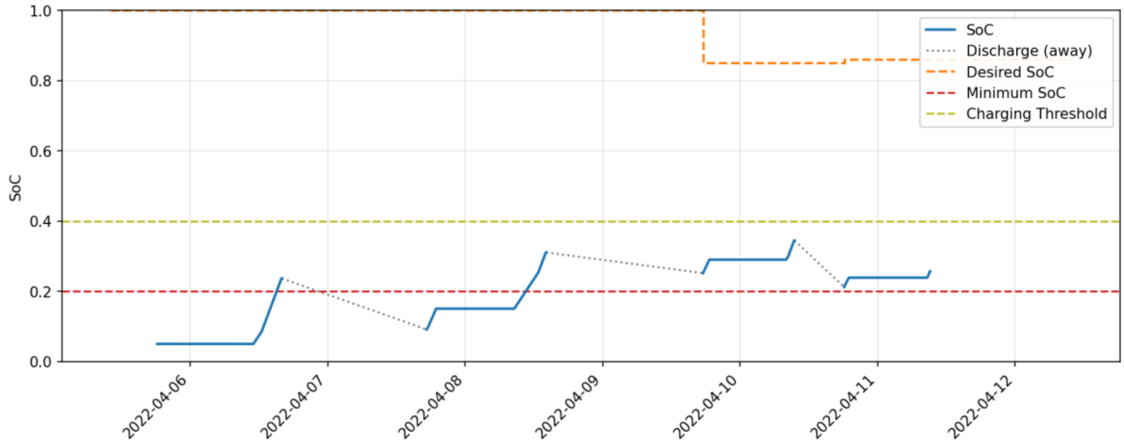


FIGURE 36. Example 4 of EV that was connected to the charging system via the Garage Switchboard.

It is possible to see that the EV could not charge more than once during the whole week.

In Figure 35 an EV is demonstrated that could not get a charging slot for the first five days and its SoC reached 0 at the end of the second day, contrarily the other EV, in Figure 36, could get a charging slot at the end of the second day and this was enough to keep going until the sixth day.

This assumes that the garage switchboard is fully available during the night, which is an unlikely situation even in the best-case scenario.

Non-Optimised Charging Switchboard

When the non-optimised version was tested, with access to the same power as the optimised version, the results were, as expected, better than when the system was connected to the garage switchboard.

However, this version runs against the opposite problem of supplying more energy per charging session than is initially required by the EV users, and experiencing higher supply rates, which means that the system operates closer to the limits of the available power.

In the following FIGURE 37 and FIGURE 38 notice how the EVs were charged above their user's desired SoC at the end of the charging sessions.

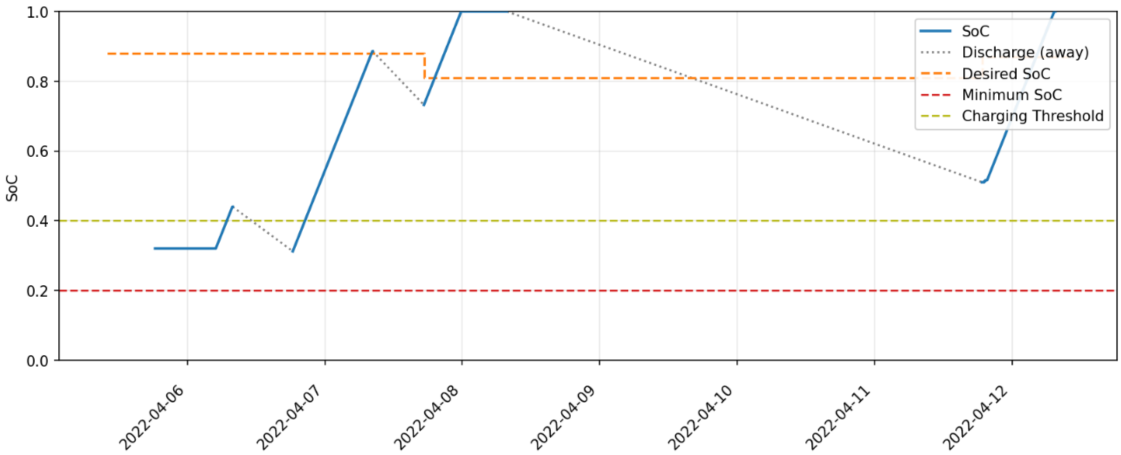


FIGURE 37. Example 5 of EV charging using a non-optimised algorithm.

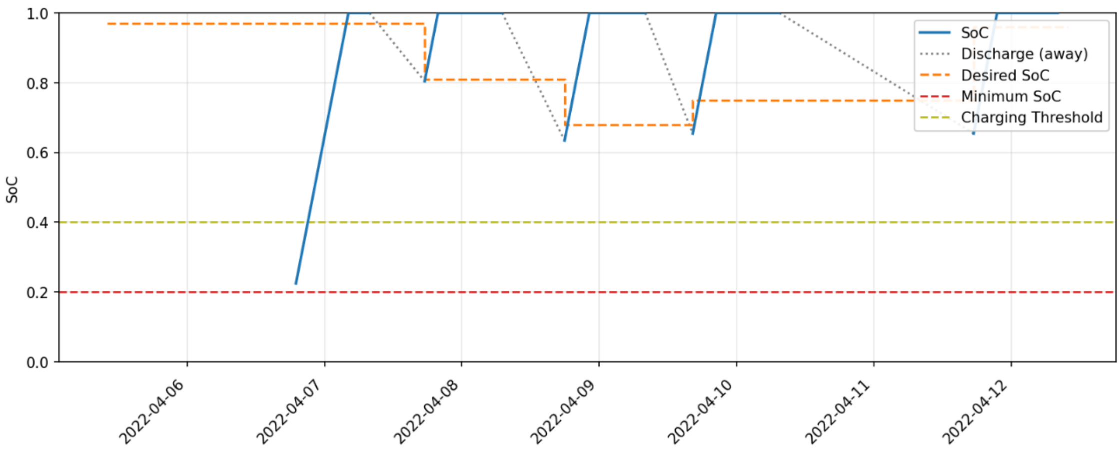


FIGURE 38. Example 6 of EV charging using a non-optimised algorithm.

This overcharging leads to longer periods of time between charging sessions. At first glance, even when considering the EV owner perspective, this may seem positive, however it means that the EV battery is being subject to longer depths of discharge and being charge at higher current rates which can negatively impact the batteries state of health (SoH) and reduce its lifetime.

In extreme cases, the non-optimised algorithm can lead to scenarios as shown in Figure 39 and Figure 40.

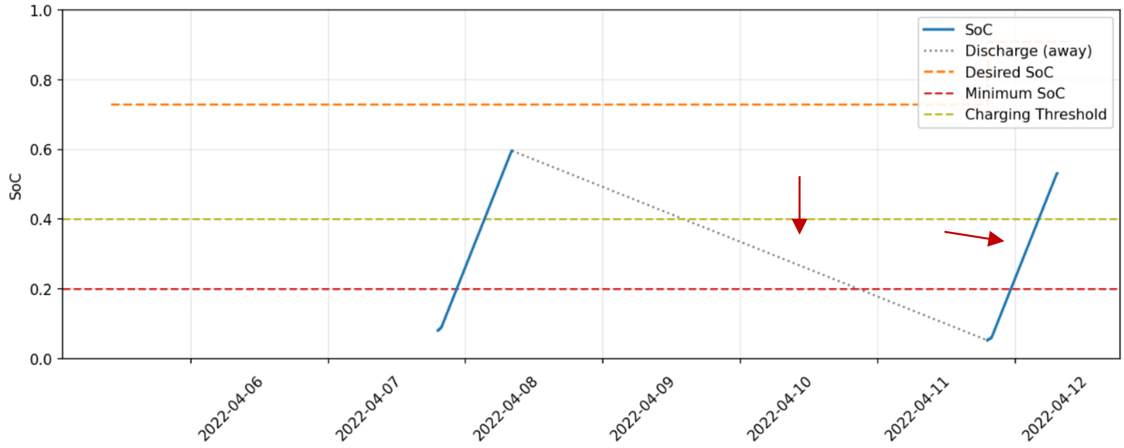


FIGURE 39. Example 7 of EV charging using a non-optimised algorithm.

The red arrows highlight the periods where the EV could not charge followed by the periods where they experienced a full charge.

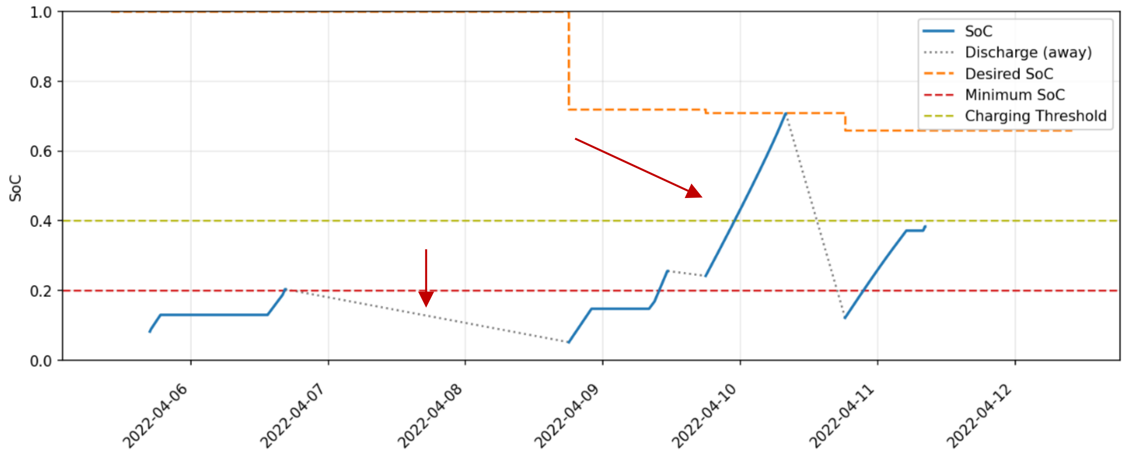


FIGURE 40. Example 8 of EV charging using a non-optimised algorithm.

The red arrows highlight the periods where the EV could not charge followed by the periods where they experienced a full charge.

The EVs in Figure 39 and Figure 40 experience a days where they could not join the charging session followed by a day where they experienced a deep charge, and in both cases, they did not require a full charge for the upcoming days.

4.6 - Results Discussion

Table XXVIII summarises the outcomes of each methodology and to maintain a just comparison each was simulated 30 times with a new fleet generated every 5 simulations.

TABLE XXVIII
Summary of charging methodologies simulations outcomes over a 7-day period.

Output (averages)	Optimised + main feeder access	Non optimised + main feeder access	Non optimised + garage switchboard access
Total energy delivered	64.31 MWh	81.74 MWh	20.20 MWh
Total distance travelled	11 832 km	11 933 km	4 166 km
EVs per charging session	48	46	21
Peak current supplied	411.78 A	627.81 A	67.20 A

When analising the results of both the optimised methodology and the non-optimised methodology, when it has access to the main feeder, outperforms the non-optimised methodology that has access to the garage switchboard. To simply put, the garage switchboard is under-dimensioned for the needs of EV owners when considering large numbers of EV integration.

The true comparison is between the two methods when they both have access to the power in the main feeder.

The optimised methodology outperforms the non-optimised methodology by achieving the same autonomy to the EVs, 11 832 km vs. 11 933 km, with a reduction of approximately 20% in power supplied.

This happens because the optimised methodology considers the EV owners' needs and halts charging when the desired SoC is met, avoiding overcharging beyond the user needs.

This optimised methodology also outperforms the non-optimised methodology by operating at lower currents. The maximum current measured was around 34% lower than the maximum current measured with the non-optimised methodology. This ensures a higher safety margin during charging as the system is further away from the maximum current allowed.

After running the simulations to compare all the methodologies, a second test was carried out to determine the safety coefficient ρ_{Safety} minimum value that still allowed the system to work properly.

To do so the value of ρ_{Safety} was decreased from 1 to 0.5 sequentially and at each iteration, 5 simulations were conducted to assure the validity of the results.

With this this work was able to conclude that from ρ_{Safety} equal to 1 to 0.8 the system could supply the desired energy demand with an average supply remaining at 32 MWh.

From 0.7 to 0.3 the system surprisingly experienced an increase in energy supplied, and could supply on average 35 MWh, however this increase does not translate to an increased user satisfactions, it just so happens that by having less power available, fewer EVs can be charged but for each EV that is removed from the charging system an additional 6A are available for the remaining EVs, this causes an increase in energy supplied to the connected EVs but a decrease in overall satisfaction.

Then beyond 0.3 the value of supplied energy decreased rapidly first to an average 26 MWh supplied and then to 14 MWh at ρ_{Safety} equal to 0.1; however, it still outperformed the second scenario conditions as expected.

So, ideally, in this case the ρ_{Safety} should be set at 0.4 as to maximise user satisfaction and minimise system operation risk. The previously explained information is demonstrated in the following Figure 41.

Poster that if the energy supplied by the systems increases but the overall distance travelled decreases it means that there are EVs that are accumulating energy without requiring it. Consequently, a more reasonable ρ_{Safety} would be 0.8.

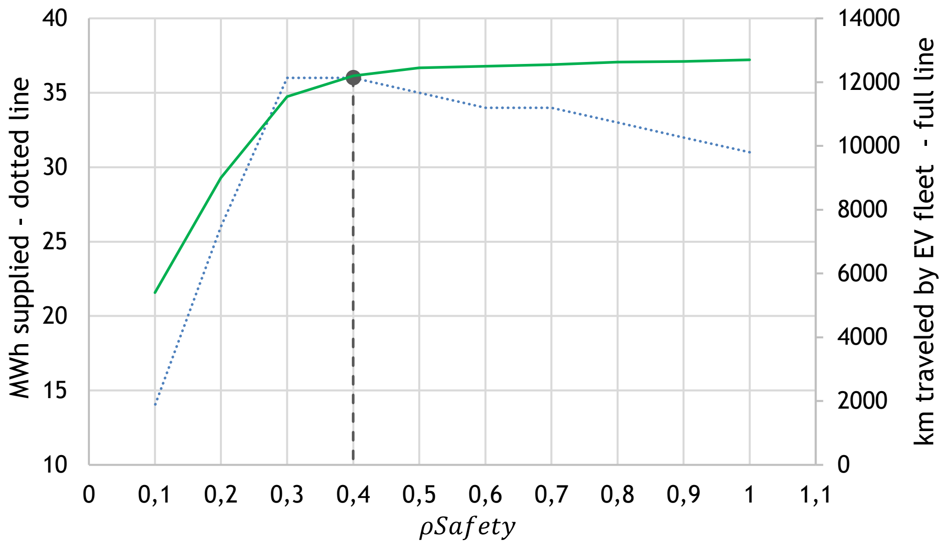


FIGURE 41. Visualisation of the evolution of supplied energy, and km travelled, according to the safety coefficient.

4.7 - Chapter Summary

Although the simulations only consider one case study, its characteristics present a solid proxy for other scenarios, furthermore the safety margins that were considered demonstrate that the proposed solution significantly outperforms a solution where the EV chargers are connected to the garage switchboard or do not rely on optimised real-time power allocation.

As demonstrated the proposed solution allows for a complete fulfilment of desired SoC levels, even when applying a safety coefficient ρ_{Safety} of 0.4.

In the case presented, the main feeder was over dimensioned by a factor close to x2.2 when looking at the peak measurements.

The main findings of this study are:

- The proposed solution applies to multi dwelling buildings where the power availability is the main issue for high integration of EVs.

- The proposed solution almost doubles the number of EVs per installation compared to the non-optimised solution which only has access to the garage switchboard.
- As the systems are based on centralised management, it allows users who do not have an EV to join later without requiring new upgrades to the electrical infrastructure.
- The optimised system allows for a 200% increase in energy delivered, from 20.20 MWh to 64.31 MWh.

Chapter 5

Battery Degradation Considerations

5.1 - Contextualisation

Electric vehicles (EVs) are central to the decarbonisation of transportation, yet battery longevity remains a key limitation to mass adoption. Battery degradation not only reduces driving range over time but also significantly affects resale value and user confidence.

While technical stressors like high temperature and deep cycling are well-known contributors to degradation, user behaviour plays an equally important role. For instance, frequent charging to 100% or plugging in overnight, driven by range anxiety, can accelerate capacity loss [137], [138].

Many EVs rely on abstract indicators such as the state of charge (SoC), which users often misinterpret. Studies show that reframing SoC in terms of real-world driving needs (e.g., "desired range for tomorrow") improves user decisions and reduces unnecessary charging [139].

Despite the impact of charging habits on battery health, manufacturer guidance is often limited to vague suggestions like “charge as needed,” providing little actionable support to users.

This work addresses that gap by proposing a user-aware, data-driven charging strategy that minimises battery degradation, it is an approach that combines electrochemical aging models, thermal modelling, and user-defined driving needs to create a personalised charging plan.

The primary objective is to extend battery life without compromising convenience. We demonstrate how this framework can be embedded into smart Battery Management Systems (BMS) to guide both individual users and fleet operators.

5.2 - Battery Degradation Framework

In this section, we present the battery degradation framework used to guide charging decisions. The framework combines aging, resistance, and thermal modelling to quantify battery health impacts during charging.

Battery Pack Cycling Degradation Model

We begin with the cycle aging model, which estimates degradation based on charge throughput, average state of charge (SoC), and usage patterns. This model lays the foundation for understanding how specific charging events affect battery State of Health (SoH).

A model for practical capacity fading for Li-Ion battery cells in EVs was developed by the authors in [91] where, once more the temperature influence on capacity fading has been modelled with the Arrhenius equation stating that the proposed capacity fading model is valid above 25°C, insufficient below 0°C, and possible too optimistic between 0°C and 25°C. They conclude that the battery lifetime can be optimised by keeping the SoC low and only charging enough energy for the upcoming trip, preventing regenerative braking at low temperatures and proper thermal management.

So, the focus is on cycle aging and not calendar aging, based on the following set of equations, (42) to (45), from [92].

$$\varepsilon_{cyc}\left(T, SoC_{avg}, SoC_{dev}, Q_{proc}\right) = \tag{42}$$

$$= \sum_{i}^{E} \left(k_{s1} \times SoC_{dev,i} \times e^{\left(k_{s2} \times SoC_{avg,i}\right)} + k_{s3} \times e^{\left(k_{s4} \times SoC_{dev,i}\right)}\right) e^{\left(-\frac{E_a}{R\left(\frac{1}{T_i} - \frac{1}{T_{ref}}\right)}\right)} Q_{proc,i} \tag{43}$$

The above terms in equation (43) can be broken down into the following equations

$$SoC_{avg,m} = \frac{1}{\Delta Q_{proc,m}} \times \int_{Q_{proc,m-1}}^{Q_{proc,m}} SoC\, dQ_{proc} \tag{44}$$

$$SoC_{dev,m} = \sqrt{\frac{1}{\Delta Q_{proc,m}} \int_{Q_{proc,m-1}}^{Q_{proc,m}} \left(SoC - SoC_{avg,m}\right)^2 dQ_{proc}} \tag{45}$$

Where $SoC_{avg,m}$ is the average SoC in an event; $Q_{proc,m-1}$ is the initial amount of charge processed at any certain event m; $Q_{proc,m}$ is the charge processed at the end of any certain event m; $\Delta Q_{proc,m}$ - is the difference between the two above; K_{s1} is a constant that scales the influence of SoC deviation linearly; K_{s2} is a constant that exponentially weights the average SoC; K_{s3} is a constant that accounts for an additional degradation term with its own scaling; K_{s4} is a constant that exponentially weights the average SoC. This then feeds into the calculation of the battery SoH, in equation (46).

$$SoH = \left(1 - \frac{\varepsilon_{cyc}}{Q_{nom}}\right) \tag{46}$$

Where Q_{nom} – is the nominal capacity of the cell.

To model the temperature variations within the battery pack, as previously stated, it is necessary to model the battery pack both physically, such as size and weight, as well as electrically, such as internal resistance, capacity and operating voltage.

While the aging model accounts for SoC and charge throughput, it is also important to model the internal resistance of the battery, which influences both heat generation and charging efficiency.

Battery Pack Resistance Model

Next, we examine battery internal resistance, a key parameter affecting both thermal behaviour and aging. Since EV manufacturers typically do not disclose resistance values, we rely on published data to approximate this variable across SoC levels.

With the outputs provided by the battery degradation model presented in the previous sub-chapter, together with the results from the simulations of degradation for the different charging schemes, it is possible to integrate these metrics into the original charging optimisation methodology in such a way that accounts for, and aims to minimise, the battery degradation according to the EV users preferences, or at least, as much as possible.

To estimate the heat generated during a charging event the internal resistance of the battery pack will play a key role, however, as mentioned before, EV manufactures do not provide this information to the public, so it is necessary to estimate it by extrapolating from other research works. In the following TABLE XXIX the relation between internal resistance and SoC of a battery pack according to [140] is shown.

TABLE XXIX
Relation between battery SoC and its internal resistance.

Internal Resistance in Ohms		SoC in %
Lithium Ion	Nickel Metal Hydride	
3.13	2.70	0
3.00	2.20	10
2.70	1.80	20
2.30	1.50	30
2.20	1.40	40
2.10	1.30	50
2.00	1.30	60
2.00	1.30	70
2.10	1.40	80
2.20	1.50	90
2.30	1.40	100

It is possible to observe, by analysing the previous TABLE XXIX, that in both cases, but more prominent in the Li-Ion battery, there is a valley of low internal resistance between the values of 30% and 80% SoC.

According to [141] LFP batteries have an internal resistance of 35 to 70 mΩ per Ah. So, it is possible to extrapolate the overall battery pack internal resistance through the battery nominal capacity in Ah, or by estimating the number of battery cells in the battery pack and applying the equation (47).

$$R_{total} = R_{cell} \times \frac{n_{series}}{n_{parallel}} \tag{47}$$

Where R_{cell} is the single cell internal resistance value; n_{series} is the number of cells in series within the battery pack; $n_{parallel}$ is the number of cells in parallel within the battery pack.

To determine the number of cells in series and in parallel withing a battery pack we can use the following equations (48) and (49).

$$Q_{nom.pack} = P \times Q_{nom.cell} \Leftrightarrow n_{parallel} = \frac{Q_{nom.pack}}{Q_{nom.cell}} \tag{48}$$

$$U_{nom.pack} = S \times U_{nom.cell} \Leftrightarrow n_{series} = \frac{U_{nom.pack}}{U_{nom.cell}} \quad (49)$$

Where $Q_{nom.pack}$ is the nominal capacity of the battery pack, in Ah; $Q_{nom.cell}$ is the nominal capacity of the battery cell, in Ah; $U_{nom.pack}$ is the nominal voltage of the battery pack, in volts; $U_{nom.cell}$ is the nominal voltage of the battery cell, in volts.

So, for the following types of battery cells, by knowing the nominal voltage for the pack, around 400V, and a standard capacity of 250 Ah we can infer the number of cells in series and in parallel, as shown in TABLE XXX. The following values are true for new cells and can degrade over time and with use, however as a simplification for the simulations the values are considered static for the useful lifetime of the battery where SoH is larger than 0.8.

TABLE XXX
Battery pack architecture based on cell type.

Form Factor	Nominal Cell Voltage	Nominal Cell Capacity	Series (n_S)	Parallel (n_P)
Cylindrical	3.6 – 3.7 V	3 – 5 Ah	Approx. 110	Approx. 105
Prismatic	3.2 – 3.7 V	50 – 280 Ah	Approx. 115	Approx. 2
Pouch Cells	2.3 – 2.4 V	30 – 100 Ah	Approx. 170	Approx. 4

With the ability to estimate both degradation and thermal effects for each charging session, we are now equipped to develop a charging strategy that balances battery health with user convenience.

Battery Pack Temperature Estimation Model

To capture the thermal impact of charging, we introduce a simplified heat generation and dissipation model based on internal resistance and charging current. This thermal model is essential for quantifying temperature-induced degradation and supports dynamic charging control.

To calculate the heat generated during the charging process it is necessary to calculate the actual charging current in the battery pack. This is because the current supplied by the charging socket, I_{AC}, will not be the same as the current entering the battery pack, I_{DC}. So, to estimate this I_{DC} value we can do it with the following formula, in equation (50).

$$I_{DC} = \frac{V_{AC} \times I_{AC} \times \eta}{V_{DC}}, where\ \eta \text{ is the efficiency of the charger} \quad (50)$$

Knowing the I_{DC} value is possible to calculate the heat being generated by the charging process with the following formula, in equation (51).

$$P_{heat} = I_{DC}^2 \times R_{pack} \quad (51)$$

Where P_{heat} is the heat generated, in Watts; R_{pack} is the internal resistance of the battery pack, in Ohms.

It is possible to make an analogue electrical system that models the EV battery thermal model. Considering the following differential equation (52).

$$m.C_p \frac{dT}{dt} = \dot{Q}_{gen} - \frac{T_{batt} - T_{amb}}{R_{th}} \quad (52)$$

It is possible to directly map it to equation (53).

$$C\frac{dV}{dt} = I_{in} - \frac{V - V_{source}}{R} \quad (53)$$

Where $\dot{Q}_{gen}$ maps to $I_{DC}^2.R_{pack}$ - These feed heat into the system like a current source feed current to a node; $m.C_p$ maps to C_{th} - Integrates heat over time, raising temperature like a capacitor raises voltage; R_{th} maps to ${}^{1}/_{h.A}$ - Represents the thermal resistance between the battery and the environment around; V_{source} maps to T_{amb} – Just like ambient temperature sets a reference temperature for the EV battery, the voltage source sets a reference voltage for the capacitor. The system described above can be represented by the following RC circuit, in FIGURE 42.

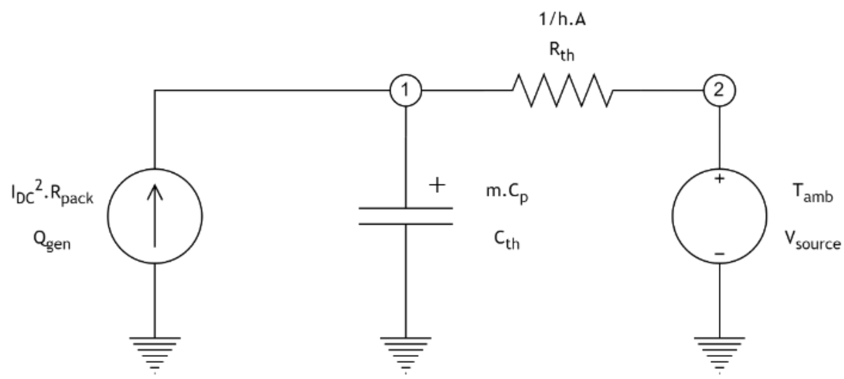


FIGURE 42. Electrical model analogue representation of the EV battery thermal model.

By analysing the RC circuit, it is possible to observe that if the voltage in node 2 is higher than the voltage in node 1 then the current flows from node 2 to node 1, meaning that the battery is being heated by the charging process and the environment around. If the voltage in node 1 is higher than the voltage in node 2 that means that the current is flowing from node 1 to node 2, meaning that the battery can dissipate heat to the environment, and if there is no I_{DC} then the battery is effectively cooling. Finally, when the voltage in node 1 is equal to the voltage in node 2 plus the voltage drop in R_{th}, then we have the situation that is analogous to the equilibrium temperature being reached.

Furthermore, as the goal is to use a set of electrical behaviours to simulate thermal dynamics, the values of each component can be directly mapped from the thermal model to the electrical model. For an EV battery with an R_{pack} value of 0.128 Ω, C_p equal to 1000 J/kg·K, a mass of 410 kg, a surface area of 5m^2 and a convection coefficient of 20 W/m^2.K, being charged at a steady current of 8,3 amp I_{DC}, and considering a T_{amb} of 21ºC the analogue electrical model would produce the voltage curve seen in FIGURE 43 at the capacitor terminals.

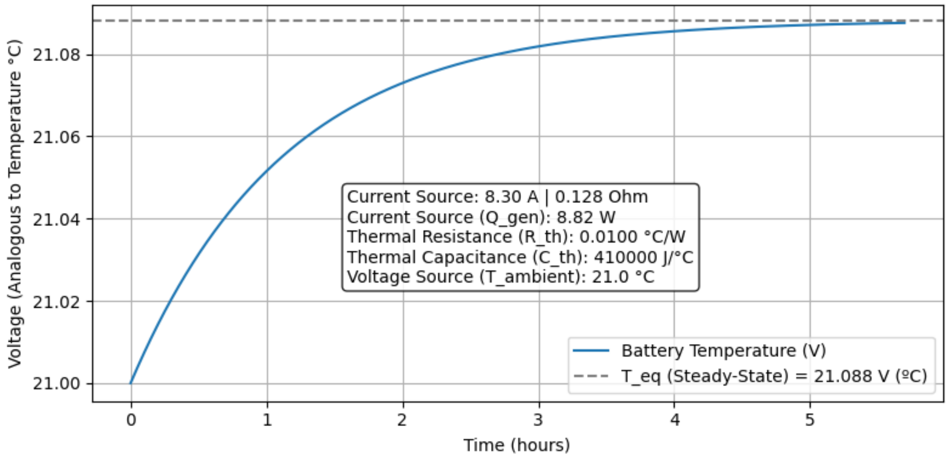


FIGURE 43. Evolution of voltage at capacitor terminals, analogue to evolution of temperature within EV battery.

The ability for the battery pack to dissipate and accumulate part of the heat it is generating while charging will be dependent on several factors that are described in the following equations, (54) to (58), and based on [142], and the previously demonstrated electrical analogue system.

$$Q_{gen} = P_{heat} . t \; [J] \tag{54}$$

$$Q_{diss} = h \times A \times \left(T_{battery} - T_{ambient}\right) \times t \; [J] \tag{55}$$

$$Q_{net} = Q_{gen} - Q_{diss} \; [J] \tag{56}$$

$$\Delta T = \frac{Q_{net}}{m \times C_p} \; [\mathrm{K}] \tag{57}$$

$$T_{batt(i)} = T_{batt(i-1)} + \Delta T \; [\mathrm{ºC}] \tag{58}$$

Where h - is the heat transfer coefficient, in $W/m^2{\cdot}K$, that will vary depending on the cooling system; A - is the surface area, in m^2, of the battery pack, typically between 1 to 2 m^2; $T_{battery}$ - is the battery temperature in Kelvin, K; $T_{ambient}$ - is the ambient temperature in Kelvin; t - is the time in seconds, s; m - is the mass of the battery pack, in kg; C_p - is the heat capacity of the battery, in J/kg·K, typically 1000 J/kg·K for Li-Ion battery packs; ΔT - is the temperature rise of the battery pack in K.

And heat dissipated and heat accumulated or net, Q_{diss} and Q_{net} respectively, we know the heat generated which is equation (51) times the time (t) of the charging event.

As general information regarding active heat dissipation mechanisms, like ventilation, is not disclosed by EV manufacturers it is only possible to consider passive cooling, as described in equation (55) and to so it is necessary to estimate the battery surface area, while assuming it has a smooth cuboid shape with no heat sinks. From the list of EVs in TABLE XXXV only the Chevrolet Bolt EV has a publicly disclosed battery size, with a surface area of approximately 4.9 m^2.

Now, with degradation model demonstrated in equations (42) to (46) for cycling aging, and the temperature model demonstrated in equations (54) to (58) it is possible to estimate the impact on battery SoH, at the end of each charging session.

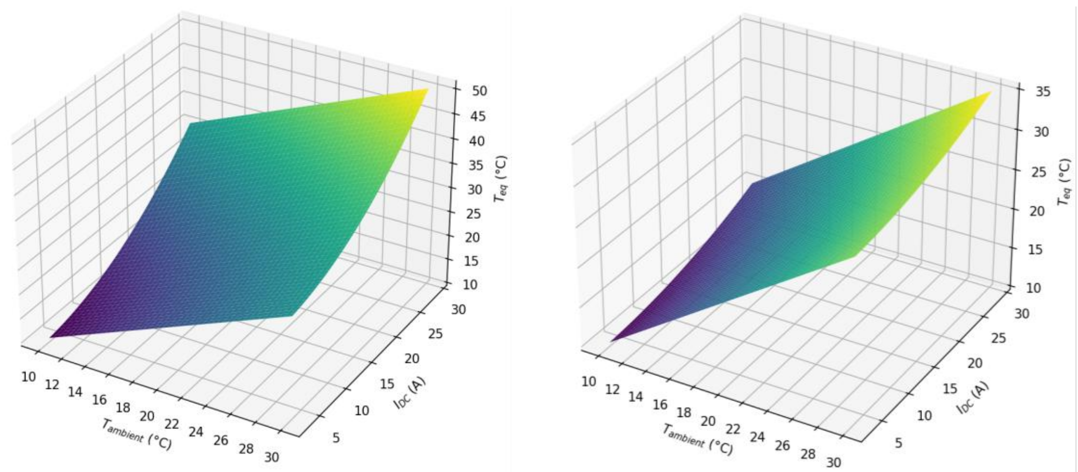
The values for temperature equilibrium, T_{eq}, or the maximum temperature the EV battery will reach given the charging conditions, where the heat generated matches the heat dissipated, can be calculated as demonstrated in equation (59).

$$T_{eq} = T_{ambient} + \frac{I_{DC}^2 \times R_{pack}}{h \times A} \tag{59}$$

So, as it is possible to conclude from equation (59) the maximum temperature reached by the EV battery during a charging period is dependent on, not only the battery characteristics, and the charging current, but also the ambient temperature, still, from these variables the charging current will be the easiest to control, however the final temperature will always be at least as high as the ambient temperature.

Take notice that equation (59) only provides an approximate estimate of T_{eq} when compared with the simulation results, this happens because it considers a steady state calculation that does not account for the over or under shoots caused by the time step in the simulations. Nonetheless, these over or under shoot situations can be partially mitigated by considering smaller time steps.

As expected, FIGURE 44, increasing the heat transfer coefficient of the battery, from 5 to 20 W/m^2.K, significantly decreased the maximum temperature reached by the battery, and as stated previously this is related to the cooling mechanisms at the battery's disposal, such as forced air cooling or liquid cooling, which in turn will allow the battery to operate at higher charging currents. However, these active cooling mechanisms will not have a significant impact if ambient temperatures are relatively high.



(a) (b)

FIGURE 44. (a) Plot of Teq in function of Tambient and IDC considering a 0.5 ohms Rpack, 4m^2 battery surface area and a heat transfer coefficient of 5 W/m^2.K. (b) Plot of Teq in function of Tambient and IDC considering a 0.5 ohms Rpack, 4m^2 battery surface area and a heat transfer coefficient of 20 W/m^2.K

Sensitivity analysis of variables

If we take the derivative of T_{eq}, in relation to all its variables one at a time, we can see how each one of them impacts the maximum temperature reached by the EV battery, considering a constant $T_{ambient}$, as seen in FIGURE 45.

It is possible to summarise the information in FIGURE 45 in the following TABLE XXXI.

TABLE XXXI
Summary of the impact of the different variables in Teq.

Variable	Impact on T_{eq}	Type	Notes
R_{pack}	Linear	Linear	Rises with battery age.
h	Curved	Diminishing returns	Active cooling mechanism boosts cooling.
A	Curved	Diminishing returns	A bigger surface area helps with passive cooling.
I_{DC}	Quadratic	Non-linear	Dominant - Faster charging causes faster heating.

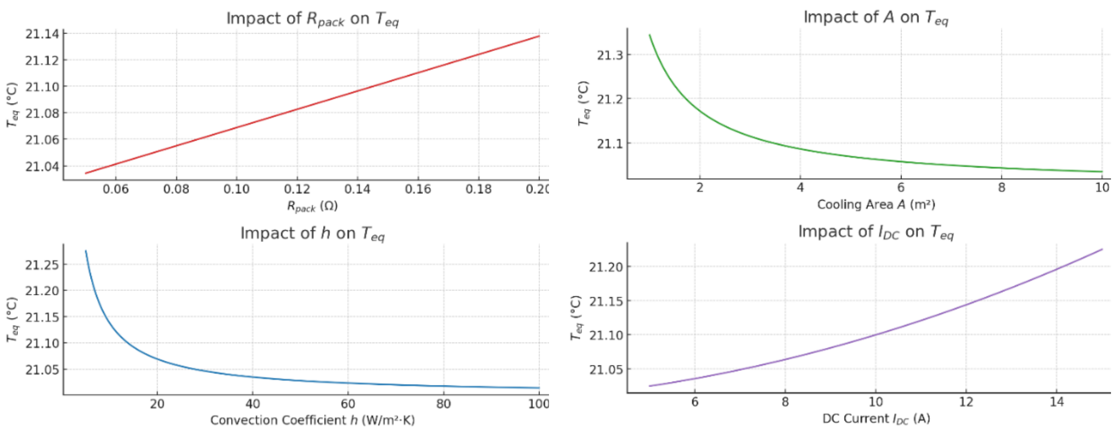


FIGURE 45. Effect of key parameters on battery equilibrium temperature (T_{eq}). The plot shows how Teq varies with (a) internal resistance (R_{pack}), convection coefficient (h), and (b) cooling surface area (A), and charging current (I_{DC}), considering a 21°C ambient temperature.

5.3 - Scheduling Methodology

With the degradation and thermal dynamics defined, we now present the charging scheduling methodology. This two-level optimisation approach minimises battery wear by adjusting charging frequency and current while respecting user energy requirements.

Looking at the problem from the perspective of temperature variation, there is a need to estimate how long it will take for the system, in this case the EV battery, to reach the equilibrium temperature, or the maximum temperature, this will vary from EV to EV depending on the battery characteristics. In a mathematical sense the equilibrium temperature is an asymptote meaning that the battery never actually reaches that temperature but in turn gets infinitely close to it. So, to estimate how long it takes for the EV battery to reach the equilibrium temperature first it is necessary to define a thermal time constant specific for each EV, as demonstrated in the following equation (59) and according to [142].

$$\tau = \frac{m \times C_p}{h \times A} \tag{60}$$

So, the EV battery temperature evolves according to equation (61)

$$T(t) = T_{eq} - \left(T_{eq} - T_0\right) \times e^{-t/\tau} \tag{61}$$

Solving for t, in equation (62).

$$t = -\tau \times \ln\left(\frac{T_{eq} - T(t)}{T_{eq} - T_0}\right) \tag{62}$$

It is possible to observe that charging current intensity does not influence the time it takes to reach the equilibrium temperature but instead influences the value of the temperature reached.

Knowing the time it takes to reach the equilibrium temperature, the SoC at the beginning of the charging period, the desired SoC and the time available for the charging period, it is possible to estimate the degree of degradation the EV battery will incur and inform the user on how changing certain inputs or charging habits can improve the overall impact on the battery's SoH.

On the other hand, authors in [143] present a set of equations as a mean to balance the duration of the charging period and its impact on the batteries SoH by varying the value of a single variable ß that is a variable between 0 and 1 that varies to act as a balance between charging as fast as possible or incurring in the least damage as possible.
The authors in [144] provide a similar version as seen in equation (63).

$$J = \beta \cdot t_f + (1 - \beta) \cdot \left(SoH_f - SoH_0\right) \tag{63}$$

From both works, the proposed variable ß can likewise be extracted and used by the optimisation algorithm as the user input to decide is desire to incur in more, or less, battery degradation at each charging period. So, the control of battery degradation can either be controlled by imposing constraints on the objective function or directly acting on it.

However, the work developed in [145], with objective function presented in (64), presents an optimisation algorithm that minimises the charging current and passively minimises the impact on SoH, while still achieving the desired SoC by the end of the charging period which means that to minimise the impact on SoH beyond would mean not reaching the desired SoC.

$$\min \sum_{i=1}^{n} \left| \frac{SOC_{Required.i}}{I_{Charging.i} \times U_{Charging}} - T_{available.i} \right| \tag{64}$$

So, to minimise the impact on the EV battery SoH it is necessary to look beyond the timeframe of individual charging sessions and focus on the users' charging habits.

It becomes a problem of minimising the users' desired SoC that will ensure enough range to meet their needs for the next day, meaning to differ as much charging as possible.

The first thing is to shift the concept of desired SoC to desired daily range, so instead of prompting the user for his desired SoC at the end of the charging period we should prompt him for his desired minimal range.

So, the goal is to complement the previously developed optimisation algorithm with a macro perspective and improve the overall impact on SoH during consecutive charging periods.

This will be a two-level, hierarchical, optimisation framework where the new aim is to optimise the charging frequency for EV charging in such a way that user driving needs are met, e.g. daily km demand, as well as respecting SoC bounds, e.g. between 40%-80%, and minimising long term battery wear.

The optimisation problem at hand is a Mixed-Integer Linear Programming (MILP) problem, where the decision variables include binary charging decisions ($charge_d \in \{0,1\}$) and continuous variables representing energy charged (E_d^{ch}) and the state of charge (SoC_d). The problem features linear constraints that describe the dynamics of the state of charge and energy balance over time, as well as linear objective terms that aim to minimise the total energy charged over the scheduling horizon. In the case where quadratic terms are added to the objective or constraints, such as for load smoothing, the problem transitions to a Mixed-Integer Quadratic Programming (MIQP) problem. To solve this class of problems, the solver *ECOS_BB* was employed in a python environment. This solver utilises branch-and-bound methods, combined with linear relaxation and cutting planes, to explore the feasible solution space while efficiently handling the binary decision variables.

Looking at this problem as a single EV problem, where the goal is to schedule each driving event in such a way that the energy provided is minimised, we can formalise the problem as presented in the following tables TABLE XXXII and THE PROBLEM is inherently combinatorial due to the inclusion of binary variables, and the solution method balances exploration of the binary decision space with the linear constraints to determine the optimal charging schedule.

TABLE XXXIII.

TABLE XXXII
Indices and Parameters for Scheduling Optimisation Problem

Indices and Parameters
→ d = 1, …, D; day index
→ D: planning horizon, (e.g. 30 days)
→ Q_{nom} [kWh]: battery's nominal capacity
→ SoC_{min}, SoC_{max} [pu]: allowable SoC bounds (e.g. 0.4-0.8)
→ $E_{max} = (SoC_{max} - SoC_{min}) \cdot Q_{nom}$ [kWh]: maximum energy system can charge in one session
→ SoC_0 [pu]: initial state of charge (e.g. 0.6)
→ E_d^{drive} [kWh]: energy consumed driving on day d (e.g. 8kWh/day)

The problem is inherently combinatorial due to the inclusion of binary variables, and the solution method balances exploration of the binary decision space with the linear constraints to determine the optimal charging schedule.

TABLE XXXIII
Decision Variables for Scheduling Optimisation Problem

Decision Variables
→ $charge_d \in \{0,1\}$: 1 if EV charges on day d, 0 otherwise
→ $E_d^{ch} \geq 0$ [kWh]: energy system charges EV on day d
→ $SoC_d \in [SoC_{min}, SoC_{max}]$ [pu]: SoC at the start of day d

In terms of constraints, they can be formulated as following in equation (65) for SoC dynamics, equation (66) for SoC bounds, equation (67) for charge gating and equation (68) for initial conditions.

$$SoC_{d+1} = SoC_d - \frac{E_d^{drive}}{Q_{nom}} + \frac{E_d^{ch}}{Q_{nom}} \tag{65}$$

$$SoC_{min} \leq SoC_d \leq SoC_{max}\ \forall d \tag{66}$$

$$0 \leq E_d^{ch} \leq charge_d \times E_{max}\ \forall d \tag{67}$$

$$SoC_0 = given\ initial\ SoC \tag{68}$$

Finally, the goal, or objective function is to minimise the total energy processed, in kWh, over the horizon, or planning timeframe, as seen in equation (69).

$$\min_{charge_d, E_d^{ch}} \sum_{d=1}^{D} E_d^{ch} \tag{69}$$

This drives the optimiser to satisfy both the SoC or range needs of the user with the least total charging, which in turn will passively reduce the cycle-related SoH impact. However, as is the algorithm could opt for front or back loading, meaning, performing longer charging events either at the beginning or the end of the charging timeframe. To avoid this lumping and spread out the charging events it is necessary to add a penalty that forces the optimiser to opt out of lumping, because essentially these two solutions are equally valid, but the lumping all the necessary charging into one day to fulfill the needs of the EV for the foreseen scheduling period is not desired because from an user perspective it can create distrust and lead the user to opt for adding more charging events outside the scope of the schedule due to range anxiety and other concerns.

To avoid lumping, the objective function in equation (69) is updated to the objective function in (70), by adding a per-charge-day penalty, where ε acts as a small penalty per charging day.

$$\min_{charge_d, E_d^{ch}} \sum_{d=1}^{D} E_d^{ch} + \varepsilon \sum_{d=1}^{D} charge_d \tag{70}$$

This correction produces the effect seen in FIGURE 46 where the optimiser avoids lumping all the necessary charge in one single day.

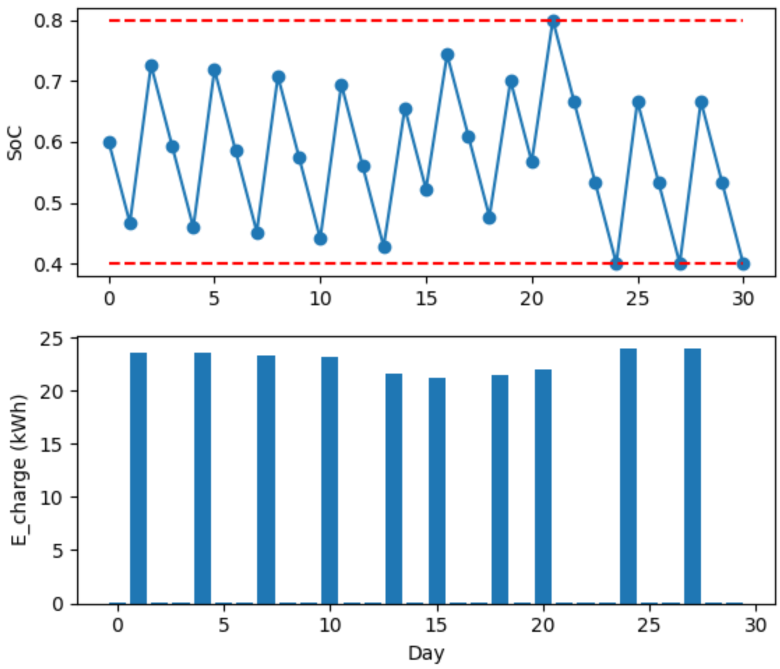


FIGURE 46. Scheduling of 1 EV using optimiser with per-charge-day penalty to avoid front or back lumping.

Furthermore, to avoid the erratic behaviour that arises from the per-charge-day penalty it is possible to add a new term to smooth out charging events, to do so the objective function in equation (70) can be updated to the one in (71), where λ is the weight on daily-charge smoothing.

To do so, it is necessary to first assess how different charging practices impact the SoH of an EV battery pack, such as charging currents, charging times, and charging events frequency.

Having seen how different approaches to the allocation compare to each other and how the temperature model works, it is possible to make changes to the algorithm used in scenario I so that better to consider the degradation of the EV battery in each charging period.

This can be done by either adding new constraints to the optimisation problem or by changing the objective function all together. At this point it becomes more evident why choosing $I_{charging}$ as the objective function target variable was a tactically sound choice, as it is, not only one of the key variables but also and easily targeted variables, in the heat generation process within the EV battery.

However, the trade-off between SoC and SoH impact at the end of each charging period should ultimately be the user's choice, otherwise any algorithm would tend to charge as little as possible at the lowest charging current as possible.

The key issue is to construct an algorithm that once more considers user input, such as the original algorithm that considers the user desired SoC and unplugging time. Still, directly inquiring users about what degree of degradation they want to incur in each charging period may also lead them to undercharge their EVs and ultimately result in an increase of overall short charging periods, where individually they may seem to have a small impact on the EV battery but cumulatively be worse than longer more spaced-out charging periods. So, the goal is to steer the user behaviour in such a way that he reduces battery degradation without compromising comfort (available SoC) at each charging period.

$$\min_{charge_d, E_d^{ch}} \sum_{d=1}^{D} E_d^{ch} + \cdots + \lambda \sum_{d=1}^{D} \left(E_d^{ch} - \bar{E}\right)^2 \tag{71}$$

This correction produces the effect seen in Figure 47 where the optimise not only avoids lumping but also avoids an erratic charging patter and opts to charge daily the exact amount of energy necessary to reach the lower bound at the end of the scheduling timeframe.

Complementary to this smoothing term is the following auxiliary definition for the average daily charge, as seen in equation (72).

$$\bar{E} = \frac{1}{D} \sum_{d=1}^{D} E_d^{ch} \tag{72}$$

Finally, two new constraints can be added, one to ensure a minimal gap between charging days, to better suit the users' availability, and a second one to ensure that in the last day, D, the optimiser ensures a pre-determined level of SoC, instead of the lower bound. These constraints can be seen in the following equations (73) and (74).

$$\sum_{k=1}^{mgd-1} charge_{d+k} \leq 1 - charge_d \tag{73}$$

$$for\ \forall d \in \{0,1,\ldots, D - \min gap\ days\ \ or\ mgd\}$$

$$SoC_D \geq target\ SoC \tag{74}$$

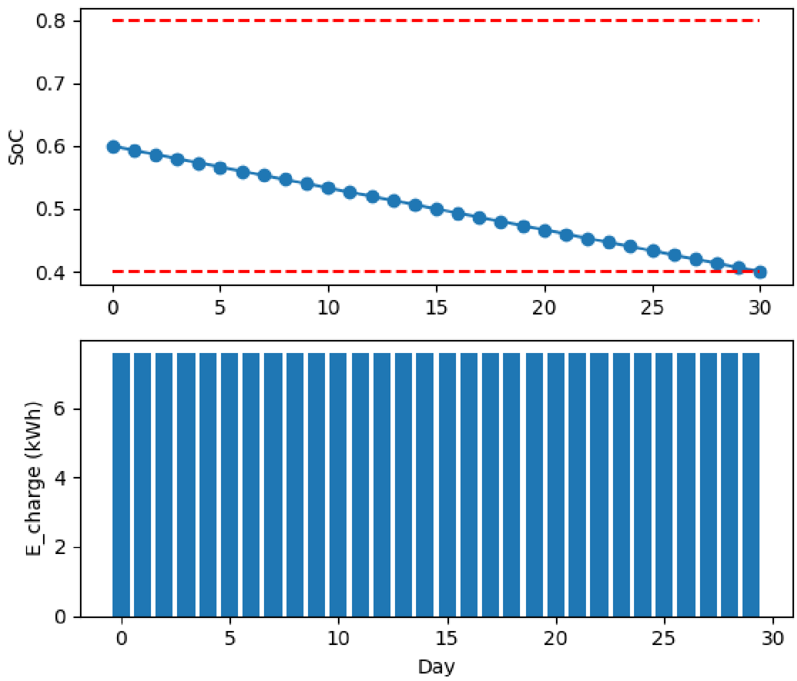


FIGURE 47. Scheduling of 1 EV using optimiser with per-charge-day penalty to avoid front or back lumping and smoothing term to avoid erratic charging.

However, enforcement of these constraints may lead the solver to a non-optimal solution, as they can oppose each other. For example, by enforcing a minimum of 2 days gap may cause this constraint to be overridden if the target SoC is not met in the final day, as see in FIGURE 48 where a 0.5 SoC was required in the final day.

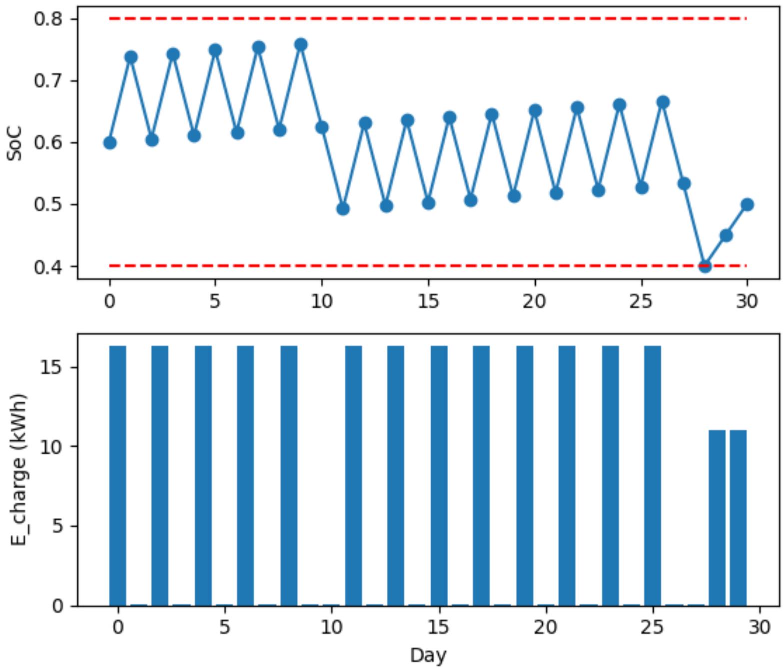


FIGURE 48. Scheduling for 1 EV using optimiser with per-charge-day penalty term, smoothing term, and minimal gap day and target SoC constraints.

To evaluate the effectiveness of the proposed charging strategies, we conducted a series of simulations under real-world usage scenarios, described in the following section.

5.4 - Results of Scheduling Methodology on Battery Degradation

This section presents the simulation results for different charging strategies, comparing their impact on battery degradation, temperature, and performance metrics. Results are benchmarked against current literature and typical EV manufacturer recommendations.

The results obtained in the previous chapters were compared with other values presented in current literature.

In [146] the authors study the energy efficiency evaluation of a stationary lithium-ion battery container storage system via electro-thermal modelling and detailed component analysis. In this experimental data is used to determine the internal thermal resistance of different components of battery packs.

The study conducted in [147], through numerical interpolation was able to pinpoint a value of 19,2 $W/m^2 \cdot K$ for the heat transfer coefficient, h. Although this study looked at thermal batteries, the value is for convective heat transfer, the same type of heat transfer considered for the models in this study.

These values also align with the findings of [148] where not only h is calculated to be between 9.91 and 8.57 $W/m^2 \cdot K$ for natural convection and 24.40 and 22.80 $W/m^2 \cdot K$ for forced convection, but also an extensive list of values for Cp is presented, ranging from 920 to 824 J/kg·K, validating the values used in this study.

The authors in [149] explore an electro-thermal model for EV applications, and in this study the values measured align with the values proposed in this work methodology in terms of temperature range.

For the purposes of the impact in the charging methodology the simulations were carried out considering a single EV allowing for a more fine-tuned approach, that will allow for a more user centric approach in the development of recommendations for minimising battery degradation.

First, consider the charging methodology presented in [145], where the objective is to vary the charging current so that the EV desired SoC is reached at the predicted unplugging time.

Second, considering a charging methodology that opts to charge the EV at the maximum current possible, so it reaches the desired SoC as fast as possible.

Third, considering a charging methodology that opts to charge the EV as the minimum current possible, delay the moment where desired SoC is reached as much as possible.

Finally, consider a charging methodology where the objective is to vary the charging current so that the degradation at the end of each charging period is as low as possible, and see how it compares to the others, mainly the first one, in terms of reaching desired SoC within available charging times.

For these different methodologies the same starting conditions will be considered in terms of SoC, SoH, Rpack and other battery specifications as well as available charging time.

Scenario I – Optimisation of Charging Current without Scheduling

In this scenario the algorithm chooses the best suited value of charging current to assure that the EV reaches the desired SoC as close as possible to the predicted unplugging time. In FIGURE 49 it is possible to see a close-in version of a single charging period. The information presented in TABLE XXXIV summarises all the charging events for the period considered.

TABLE XXXIV
Scenario I - Impact on SoH Summary

Variable	Value
ξ_{cyc} Total:	2.928
SoH:	0.980
Highest Measured Temperature:	21.005 ºC
Average Current:	7.980 A

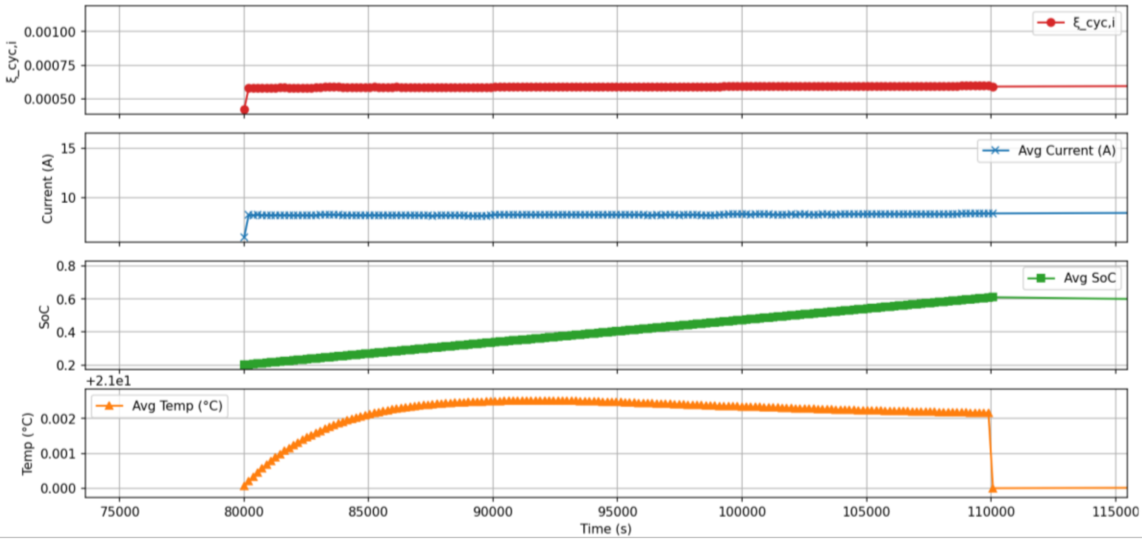


FIGURE 49. Degradation value (ξ_{cyc_i}) per charging event during an 8.3h charging period in function of charging current, SoC, and battery temperature, in scenario I.

In this scenario, a total of 40 kAh of charged was processed by the battery in charging periods, considering a 1.7% decrease in battery SoH it translates to a 1% decrease in battery at every 24 kAh of charge processed approximately. It is acknowledge that the temperature rise is considerably small, this aligns with what is expected considering the mass and surface area of the battery, as well as the low charging currents.

In the work carried out in [150] while the authors do not give "% per Ah" directly, they cite data from empirical testing [39] where LFPs maintain 80% SoH after approximately 5000 effective full cycles at 1C, for a 40 kWh battery, that equates to 5000 times 100 Ah or 500 kAh processed a before 20% loss, or a 1% per 25 kAh.

The authors of [151] carried after extensive testing on a 2.05 Ah lithium cell were able to measure the decrease in capacity after successive cycling, at 1C charging and 3C discharging. For these aging tests, when cycling between 25% and 75% SoC the battery decreased from 100% to 80% capacity after approximately 1400 equivalent full cycles, this equates to 2870 Ah processed in total, or a 1% decrease every 143.5 Ah. This value is significantly more penalising than the demonstrated, however in their work the authors maintained the battery at 35 ºC versus the 21 ºC considered and charged at 1C, which also contrasts with the value of approximately 0.046C experienced by the EV charging at an average of 8A.

In [152] the authors measure battery degradation through the distance an EV can travel between two consecutive charges, and how that distance decreases as the number of cycles increases. In their work it is possible to observe three distinct phases of battery health, a first stage where the distance between charging decreases, a second stage where this distance holds steady for several cycles, and a final third stage where the distance decreases in a more accentuated rhythm. They observe that the battery starts with a 210 km autonomy and after 972 cycles it decreases to 130 km. The battery observed is a 60-kWh battery, which is the rough equivalent of a 180 Ah battery, after 972 cycles this amounts to 175 kAh processed current. Considering consistent and controlled conditions it is possible to transpose this decrease in autonomy to a decrease in battery capacity, so if 210 km is a capacity of 100% then 130 km will amount to a decrease in capacity of 38.1%, which amounts to an approximate decrease of 1% every 5 kAh, however this also accounts for discharging and calendar aging, resulting in a more aggressive degradation pattern.

Scenario II – Optimisation of Impact on SoH with Scheduling

In TABLE XXXV, a comparative overview highlights the degree of information provided by leading EV models in 2024, underscoring a widespread shortfall in consumer-facing guidance.

TABLE XXXV
Manufacturer recommendations for battery charging according to electric vehicle user's manual.

Brand and Model	Charging					
	Lvl.	Time	Current	Interval	Max. Power	Max. Current
Tesla Model Y [153]	2	8-12 hours	Up to 48A	Charge as needed	250 kW	Up to 520A
Tesla Model 3 [154]	2	8-12 hours	Up to 48A	Charge as needed	250 kW	Up to 520A
Volkswagen ID.4 [155]*	2	7.5 hours	Up to 32A	Charge as needed	125 kW	Up to 300A
Hyundai Ioniq 5 [156]	2	6.5 hours	Up to 32A	Charge as needed	350 kW	Up to 500A
Ford Mustang Mach-E [157]	2	10-11 hours	Up to 32A	Charge as needed	150 kW	Up to 375A
Chevrolet Bolt EV [158]	2	9.5 hours	Up to 32A	Charge as needed	55 kW	Up to 150A
Nissan Leaf [159]	2	7.5 hours	Up to 32A	Charge as needed	100 kW	Up to 250A
BMW i4 [160]*	2	8.5 hours	Up to 32A	Charge as needed	200 kW	Up to 500A
Audi e-tron [161]*	2	9 hours	Up to 32A	Charge as needed	150 kW	Up to 375A
Porsche Taycan [162]*	2	10.5 hours	Up to 32A	Charge as needed	270 kW	Up to 675A

* Not retrieved from official source

Beyond this information some manufacturers also include other recommendations such as:

- Volkswagen: Upper battery charge limit: maximum 80%. Lower battery charge limit: at least 20% at mild to warm outside temperatures and 40% at cool to cold outside temperatures.
- Mustang: Allow for 2-3 hours between plugging in the EV and the moment the charging starts, to allow time for the battery pack to cool. For daily use do not charge beyond 90% SoC, to reduce the stress, the battery is subject to.

- Chevrolet: Allows the user to choose between 8A to 12A but does not specify which is better for the battery pack.
- BMW: Recommends that users opt for AC charging for daily use and keep the battery pack between 10% to 80% SoC when possible. They also recommend that, when parked, the battery pack should have a charge level of 30% to 50% SoC, and to not leave idle for longer than 14 days.
- Porsche: Recommends that users use the timer or profile function to set the high-voltage battery to charge to a maximum 80% for the vehicle's daily use, excluding long-distance trips.

So, it is possible to observe that the information provided is limited to the bare minimum for safe use of the EV.

When looking beyond the manufacturer's instructions and into the information provided by the state-of-the-art literature and the body of research one can find more material, however, the focus seems to be, as previously stated, in scheduling techniques, charging station integration, pricing schemes and other utility-centric perspectives, missing the user-centric perspective, meaning for example, how to maximise battery life with correct charging habits and practices.

To tackle this gap, this work provides a framework for battery degradation estimation together with a scheduling algorithm that aims to minimise the impact on battery SoH caused by charging events without compromising the charge availability.

It is necessary to determine if, by following the previous schedules presented in FIGURE 48, the user can incur a smaller impact on SoH than what he would just by charging at random days or in a less orderly manner.

For that, the scheduling algorithm outputs the following information, in TABLE XXXVI, that allows the charging algorithm to manage the fleet charging.

TABLE XXXVI
Scheduling Algorithm Output for one EV.

Day	Charge (1 = Yes | 0 = No)	Energy Charged [kWh]
0	1	16.308
1	0	0
...	...	...
28	1	11.000
29	1	11.000

After generating the month-long charging schedule, it was necessary to submit it to the charging simulator to calculate the impact it would have on the EVs SoH. This impact can be seen in the following TABLE XXXVII.

TABLE XXXVII
Impact on SoH - Scenario IV summary with schedule.

Variable	Value
ξ_{cyc} Total:	3.9865
SoH:	0.9847
Highest Measured Temperature:	21,0059 °C
Average Current:	7,5262 A
Distance Travelled	3225 km

While the following Figure 50 demonstrate the simulator adherence to the schedule provided.

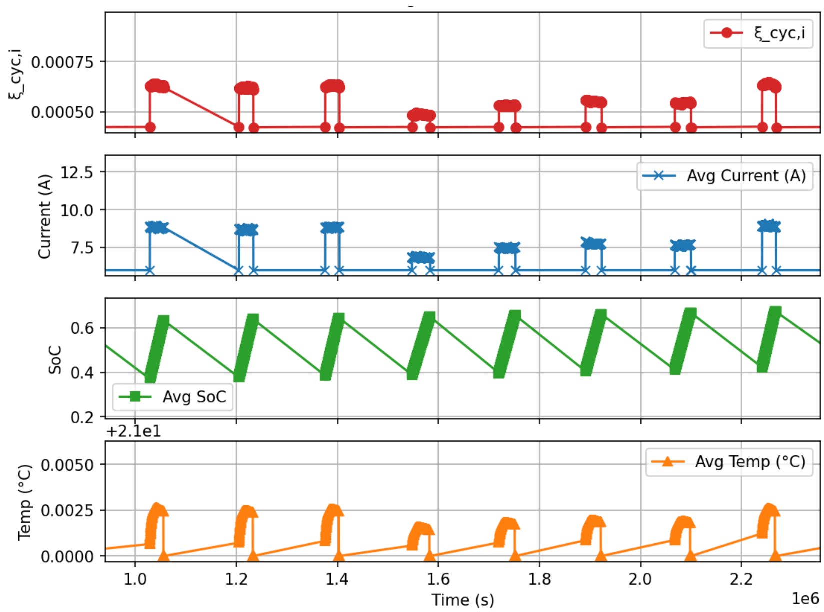


FIGURE 50. Degradation value (ξcyc_i) per charging event, in scenario IV, considering an optimised schedule. Notice the +2.1e1 scaling on Temperature for 21°C.

And in TABLE XXXVIII we can see the same impact on SoH summary when no schedule is considered, this differs from the previous scenario I because different EV characteristics were considered.

TABLE XXXVIII
Impact on SoH - Scenario IV summary without schedule.

Variable	Value
ξ_{cyc} Total:	4.0966
SoH:	0.9842
Highest Measured Temperature:	21.0080 °C
Average Current:	10.9549 A
Distance Travelled:	3225 km

In Figure 51 we can see the charging behaviour of the EV user without the optimised scheduling recommendations.

So, it is possible to observe that by scheduling the charging events, instead of relying on user behaviour, after a 30-day period we can achieve a lower impact on SoH, in this case approximately 2,7% better. This is mainly because of the average electrical current allocated to the EV, which kept the battery, during charging, at a lower temperature. Although the differing impact may seem low, these simulations only span for a 30-day period.

While these results highlight the measurable benefits of intelligent scheduling, the need to reflect on the broader implications for system design, user interfaces, and BMS integration becomes apparent.

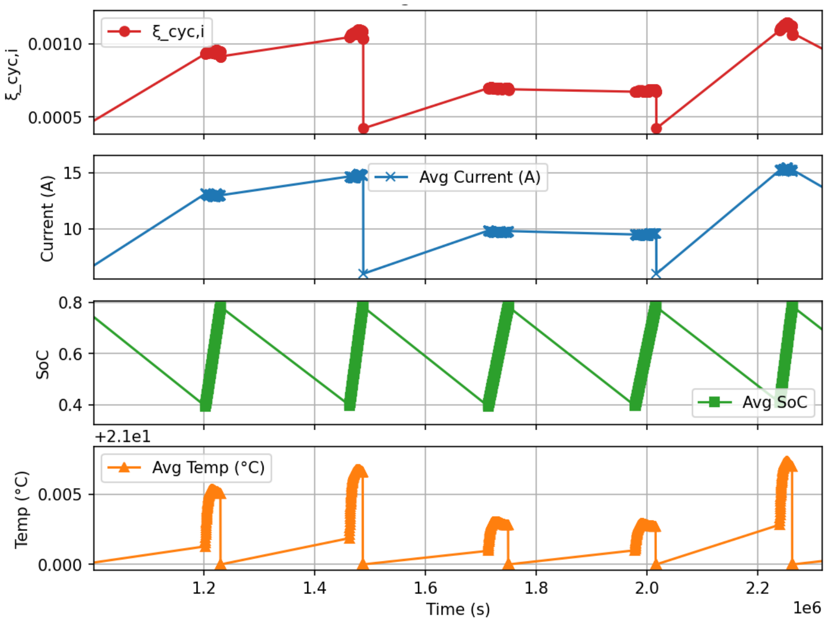


FIGURE 51. Degradation value (ξcyc_i) per charging event, in scenario IV, not considering an optimised schedule. Notice the +2.1e1 scaling on Temperature for 21°C.

5.5 - Chapter Summary

Here, we discuss the broader implications of the results, focusing on integration into real-world systems, the role of user behaviour, and potential software improvements in BMS. We also explore how range-based interfaces can further reduce battery stress.

The results demonstrate that embedding thermal and aging models in Battery Management Systems (BMS) can substantially reduce degradation without compromising usability. For example, Scenario IV showed that a smart algorithm can modulate charging current to reduce internal heating and cycle stress, preserving SoH with no change in user experience. This supports the integration of real-time SoH-aware control into commercial EV software.

As seen in the literature and confirmed by these simulations, users often unintentionally accelerate battery degradation through routine practices like full overnight charging. A simple interface that provides battery health feedback, or suggests optimal SoC targets, could steer users toward healthier habits. These small nudges, especially when tied to visible degradation metrics, could have significant cumulative effects over time.

One major insight from this user modelling is that asking for SoC percentages is both unintuitive and misleading. A range-based input (e.g., “How many km do you need tomorrow?”) aligns more directly with user expectations and can reduce overcharging. This framework is compatible with such range-based goals, translating them into minimal necessary energy input.

Optimised scheduling over multiple days proved to reduce SoH degradation by ~2.7% compared to ad hoc charging. This finding highlight that even modest algorithmic refinements at the scheduling level can yield long-term gains. It also suggests a two-level strategy: local control of charging current per session, and global planning of charge events based on mobility patterns.

Future implementations of this approach can tailor charging based on individual usage, battery condition, ambient temperatures, and even local electricity prices. The flexibility of this algorithm makes it possible to deliver per-user strategies that dynamically adapts as the battery ages, an important feature for extending EV longevity and building driver trust.

The concluding section summarises the contributions and outlines future directions for EV battery management through personalised, degradation-aware charging strategies.

We demonstrated that intelligent, user-aware charging strategies can significantly reduce electric vehicle (EV) battery degradation without compromising range availability or driver convenience. By integrating cycle aging models, thermal simulations, and user-defined range needs, this methodology dynamically adjusts charging behaviour to minimise stress on the battery. Importantly, we showed that both session-level optimisation and multi-day charging scheduling offer measurable improvements in battery longevity, even with unchanged driving habits.

A key takeaway is that user behaviour, not just hardware limits, plays a decisive role in battery health. With simple software enhancements to Battery Management Systems (BMS), including range-based interfaces and real-time SoH feedback, it is possible to shift charging patterns toward more sustainable practices. The proposed framework is scalable, adaptable to individual usage patterns, and ready to support the next generation of EV systems. By embedding battery intelligence where it matters most, at the intersection of software, hardware, and behaviour, we can extend EV lifespan and accelerate their adoption.

Chapter 6

Conclusions

6.1 - Overview and Principal Findings

The accelerating adoption of electric vehicles presents a critical infrastructure challenge for existing multi-dwelling buildings. Throughout Europe, and particularly in Portugal where over 70% of the urban population resides in apartment buildings, the electrical installations of residential condominiums were designed decades ago for lighting and conventional appliances, not for the substantial loads demanded by EV charging. This thesis addressed the fundamental question of how to enable large-scale EV integration in these buildings without requiring costly electrical infrastructure upgrades.

The research was motivated by a practical observation: residential buildings possess significant idle electrical capacity during low-demand periods, particularly overnight, yet this capacity remains inaccessible to EV owners due to the limitations of conventional charging approaches. Traditional solutions, either connecting chargers to undersized garage switchboards or duplicating feeders from the distribution network, either fail to meet user needs or impose disproportionate capital costs on building communities.

This thesis proposed and validated an integrated solution combining three interconnected elements: a centralised charging topology that taps the main building feeder with dynamic load management; a data-driven coincidence factor framework that enables realistic infrastructure sizing; and a degradation-aware scheduling layer that protects battery assets while meeting mobility requirements. The following sections present the research questions, evaluate the hypothesis, summarise the principal contributions, acknowledge limitations, and outline directions for future work.

Central Research Question

The thesis was guided by a central research question posed:

> *What is the most cost-effective way to allow a high number of EV owners to charge their vehicles in existing multi-dwelling buildings?*

Answer: The most cost-effective approach is to harvest the idle capacity of the building's main feeder through a centralised, optimised charging system that dynamically allocates available power among connected EVs based on user-declared departure times and energy requirements. This topology, combined with realistic coincidence factors for infrastructure sizing, enables a greater-than-200% increase in delivered energy compared to conventional garage-switchboard connections, while avoiding the capital expenditure associated with feeder duplication or transformer upgrades. The approach requires only minor infrastructure modifications, primarily pre-cabling and a site controller, making it economically proportionate for condominium associations. (Chapters 3–4)

Hypothesis Evaluation

The central hypothesis formulated in Chapter 4 stated:

> *It is possible to use the idle power capacity of a residential building to charge a high number of EVs during low-demand hours by implementing minor electrical infrastructural changes and applying a centralised, optimised charging methodology.*

Evaluation: The hypothesis is supported by empirical evidence across multiple dimensions, subject to the boundary conditions identified in Section 6.5. The validation demonstrates that the proposed approach achieves its stated objectives across four critical performance domains.

Regarding idle capacity utilisation, six months of high-resolution measurements (3-minute intervals) from a Porto condominium confirmed that the main feeder operates at approximately 45% of rated capacity during peak residential demand periods, leaving substantial headroom during overnight hours. The proposed optimiser successfully harvested this idle capacity to serve EV loads while maintaining safety margins for non-EV consumption, demonstrating that building electrical infrastructure possesses latent capacity that can be accessed through intelligent coordination. From an infrastructure perspective, the centralised topology proved economically and technically feasible, requiring only pre-cabling from the main distribution board to parking spaces and installation of a site controller, modifications achievable within existing condominium frameworks without requiring DSO intervention for feeder upgrades or transformer replacement.

The system's ability to enable high EV penetration was validated through extensive simulations, which demonstrated that the optimised approach could serve nearly twice the number of EVs compared to a non-optimised garage-switchboard baseline. Delivered energy increased from 20.20 MWh to 64.31 MWh over a representative 7-day period, representing a 218% improvement in charging capacity from the same electrical infrastructure. Critically, this performance improvement did not compromise user satisfaction: the optimiser consistently achieved user-declared desired SoC targets by declared departure times, even with conservative safety coefficients (ρ_{safety}) as low as 0.4, demonstrating that the approach preserves mobility requirements without compromising electrical safety. The hypothesis boundary conditions, access to main-feeder telemetry, user provision of departure times, and compliance with Portuguese LV regulations, were validated as realistic operational requirements rather than prohibitive constraints, confirming the practical viability of the proposed methodology.

Derived Research Question

The central research question was systematically decomposed into four specific research questions, each addressing a distinct technical or methodological dimension of the integration challenge. These questions, together with their answers, form the core contributions of this thesis.

The first research question examined how to effectively leverage the idle power capacity of residential buildings for EV charging during low-demand periods. Through the development and validation of a non-linear optimisation formulation based on Sequential Least Squares Programming (SLSQP), this thesis demonstrated that minimising the time error between SoC achievement and user-declared departure times, while respecting infrastructure constraints, enables effective capacity harvesting. The optimisation algorithm operates subject to voltage variation limits, per-connector current bounds (Mode 2: 6-16 A; Mode 3: 16-32 A), and a building-level safety coefficient that reserves capacity for conventional residential loads. Executing in near-real-time at 3-minute intervals, the optimiser demonstrated approximately 20% improvement in delivered energy and 34% reduction in peak feeder current compared to a non-optimised central allocator, while maintaining equivalent fleet autonomy and user satisfaction.

The second research question investigated appropriate coincidence factors for EV fleet charging in condominium settings under optimised charging regimes, addressing the critical gap in existing electrical standards. Analysis revealed that applying $C_f = 1$ (worst-case simultaneity) systematically oversizes infrastructure, leading to unnecessary capital expenditure. Through generation of synthetic charging series for fleets ranging from 1 to 70 EVs, this thesis derived power-law coincidence factor relationships (C_{f1} through C_{f4}) under both minimisation and maximisation optimisation objectives. The introduction of a fuzzy-logic inference layer that corrects nominal C_f values based on expected demand intensity and optimisation level produces $C_{f\text{-fuzzy}}$ values that enable conductor section reductions of up to 7× compared to conservative $C_f = 1$ assumptions, while maintaining full compliance with Portuguese RTIEBT requirements. This framework provides electrical designers with a transparent, evidence-based methodology for infrastructure sizing that reflects actual operational diversity rather than theoretical worst-case conditions.

The third research question addressed the design of charging schedules that minimise battery degradation while meeting user mobility requirements, recognising that EV batteries represent significant capital assets whose longevity directly affects the economic viability of electric mobility. The solution integrates a degradation-aware scheduling layer that couples the charging allocator with a cycle-aging proxy—parameterised by processed charge and average state of charge—and a lumped thermal model that estimates pack equilibrium temperature from internal resistance, cooling pathway characteristics, and ambient conditions. Comparative scenario studies demonstrated approximately 2.7% reduction in state-of-health degradation metrics through optimised multi-day scheduling compared to ad-hoc charging approaches, achieved without measurable impact on range availability or user convenience. Importantly, the scheduling layer maintains computational tractability, requiring only commodity inputs and remaining deployable on standard site controllers without specialised hardware.

The fourth research question focused on translating theoretical contributions into practical design guidelines that facilitate adoption by the diverse stakeholders involved in condominium charging infrastructure. This thesis consolidates a suite of transferable planning artifacts that bridge the gap between academic methodology and procurement practice. These include comprehensive look-up tables that link fleet size, coincidence factor variant, and per-phase design current to standard copper and aluminum conductor sections and commercially available circuit breaker calibers; two-dimensional and three-dimensional boundary maps that visualise feasible operating regions in the {fleet size; I_{max}} parameter space; and systematic decision procedures for selecting the safety coefficient ρ_{safety}, with analysis indicating that $\rho_{safety} \approx 0.8$ provides a robust compromise between energy delivery and electrical safety, while operability extends down to $\rho_{safety} \approx 0.4$ under favorable conditions. Collectively, these artifacts enable condominium associations, electrical designers, and distribution system operators to reason systematically from user-centric charging needs to conductor and protection bills-of-materials without defaulting to worst-case simultaneity assumptions, while remaining parameterisable for adaptation to other building contexts as local measurement data accumulate.

This thesis advances the field of electric vehicle integration in residential buildings through four interconnected contributions that span theoretical development, methodological innovation, and practical implementation frameworks.

Operational Methodology and Algorithm Development

The first contribution establishes an operational methodology for centralised, optimised charging that harvests available capacity from the building's main feeder through dynamic load management. The SLSQP-based real-time allocator, validated against six months of empirical measurements from a Porto condominium, demonstrates that intelligent coordination can achieve step-change improvements in infrastructure utilisation. Key performance metrics include a greater than 200% increase in delivered energy compared to conventional garage-switchboard connections, approximately 34% reduction in peak feeder current relative to non-optimised central charging, and 100% achievement of user-declared state-of-charge targets by departure time. The methodology is documented with sufficient technical detail to enable reproduction and adaptation to other building contexts, providing a validated reference implementation for dynamic load management in multi-dwelling environments.

Infrastructure Sizing Framework

The second contribution addresses a critical gap in electrical engineering standards through the development of a data-driven coincidence factor framework specifically tailored for multi-unit dwelling EV infrastructure. This thesis provides the first comprehensive coincidence factor tables derived explicitly for EV charging in shared residential contexts, covering fleet sizes from 1 to 70 vehicles under both optimised and non-optimised charging regimes. The derived power-law relationships, validated through high R^2 fits to synthetic charging populations, coupled with fuzzy-logic correction methodology, enable infrastructure planners to select conductor sections and protection devices with transparent, evidence-based rationale. This framework replaces the conservative $C_f = 1$ assumption that has historically driven infrastructure over-investment, enabling more economically proportionate designs while maintaining full compliance with Portuguese technical regulations.

Practical Implementation Artifacts

The third contribution translates theoretical contributions into transferable planning artifacts that bridge the persistent gap between academic methodology and procurement practice. The consolidated suite includes comprehensive look-up tables linking fleet size, coincidence factor variant, and per-phase design current to standard copper and aluminum conductor sections and commercially available circuit breaker calibers; two-dimensional and three-dimensional boundary maps that visualise feasible operating regions and enable rapid feasibility assessment; and systematic decision procedures for selecting safety coefficients under different operational scenarios. These artifacts enable condominium associations, electrical designers, and distribution system operators to move from user-centric charging requirements to complete bills-of-materials without requiring specialised expertise in optimisation theory, while maintaining the flexibility to adapt parameters as building-specific measurement data become available.

Battery Health Integration

The fourth contribution extends the charging optimisation framework to incorporate long-term asset management considerations through a degradation-aware scheduling layer. By embedding semi-empirical proxies for cycle-aging and thermal stress directly into the charging scheduler, the solution achieves simultaneous optimisation across user satisfaction, electrical safety, and battery longevity, objectives traditionally treated as independent or conflicting. The scheduling layer maintains computational tractability by requiring only commodity inputs readily available in standard installations: state of charge, daily distance estimates, user-declared departure times, and ambient temperature. This lightweight approach enables deployment on commodity site controllers or integration into commercial charge point management systems without specialised hardware or extensive parameterisation. Comparative scenario studies demonstrate approximately 2.7% reduction in state-of-health degradation metrics over multi-day scheduling horizons, achieved without measurable impact on range availability, validating that battery health and user convenience can be co-optimised rather than traded off.

6.2 - Contribution Positioning and Significance

The contributions of this thesis address specific gaps identified in the literature review:

Gap 1 (Market-Infrastructure Mismatch): While the literature extensively addresses single-family home charging and public infrastructure, optimisation frameworks specifically designed for the shared-infrastructure constraints of multi-dwelling buildings remain scarce. This thesis provides the first comprehensive treatment integrating topology design, dynamic load management, and infrastructure sizing within the MUD context, validated against Portuguese regulatory requirements.

Gap 2 (Coincidence Factors): Existing C_f tables in electrical codes (DMA-C13-912/N, IEC 60050) address conventional residential loads but provide no guidance for EV charging installations. Prior work by Almutairi et al. [16] and others assumed $C_f = 1$ or arbitrary values without empirical justification. This thesis derives the first behaviour-constrained C_f families specifically for MUD EV fleets, validated against synthetic populations parameterised from European driving statistics.

Gap 3 (Behavioural Assumptions): Many optimisation studies assume deterministic arrival/departure times or homogeneous user behaviour. This thesis incorporates stochastic behavioural parameters (plug-in/out distributions, SoC arrival patterns, energy requirements) derived from recent empirical studies, producing results that better reflect real-world operational conditions.

Gap 4 (Integrated Methodology): Prior work typically addresses charging optimisation, infrastructure sizing, or battery health as isolated problems. This thesis demonstrates an integrated methodology where optimisation outputs feed infrastructure dimensioning, which in turn constrains operational parameters—creating a coherent framework from planning to operation.

Gap 5 (Degradation Methodology): While sophisticated battery degradation models exist in the electrochemistry literature, their integration into real-time charging schedulers has been limited by computational complexity. This thesis demonstrates that lightweight semi-empirical proxies can capture the dominant degradation mechanisms (cycle aging, thermal stress) with sufficient fidelity for scheduling applications, without requiring detailed cell-level models.

6.3 - Limitations and Practical Considerations

The validity of the results and the applicability of the proposed methodology are subject to several limitations and boundary conditions that should be considered when interpreting findings or adapting the approach to other contexts. These constraints span data availability, methodological assumptions, and deliberate scope boundaries.

Empirical Validation Scope

The empirical validation draws on six months of high-resolution feeder measurements from a single Porto condominium, supplemented by synthetic EV populations parameterised from European mobility statistics. While the validation building possesses characteristics representative of Portuguese urban multi-dwelling environments, three-phase supply, shared parking facilities, mixed residential usage patterns, the specific numerical thresholds derived from this dataset, including optimal safety coefficient ranges (ρ_{safety}) and coincidence factor curve coefficients, may require recalibration when applied to buildings with substantially different load profiles, feeder ratings, or occupancy characteristics. The reliance on synthetic EV populations, necessitated by limited availability of measured charging sessions from Portuguese EV owners at the time of research, introduces parametric uncertainty that should diminish as national EV adoption matures and Portuguese-specific behavioral datasets become available for model refinement. Additionally, the six-month temporal coverage, while spanning multiple seasons, may not fully capture extreme seasonal variations such as winter heating loads or holiday-period occupancy patterns that could affect both baseline residential demand and EV charging behavior.

Methodological Assumptions and Their Implications

The optimisation framework operates under several methodological assumptions whose relaxation would require algorithm modifications. The system assumes users provide reasonably accurate departure time declarations; the impact of systematic departure-time errors or deliberate user non-compliance was not extensively tested, though the incorporation of safety coefficients provides operational margin for moderate discrepancies. The battery thermal model employs quasi-equilibrium assumptions that accurately capture steady-state operation but do not fully resolve rapid thermal transients, such as cold-start charging immediately following highway driving. This limitation is partially mitigated through conservative C-rate limiting when estimated pack temperatures approach or exceed predefined thresholds, providing thermal safety margins at the cost of some charging rate optimisation. The simulation framework assumes homogeneous charging equipment capabilities, either Mode 2 or Mode 3 uniformly across all parking spaces, whereas mixed installations combining different power levels would require adaptation of the phase allocation and current distribution algorithms to account for heterogeneous connector constraints.

Deliberate Scope Boundaries

Several scope boundaries were deliberately established to maintain research focus and align with current technological and regulatory realities. This thesis addresses unidirectional vehicle-to-grid charging (V1G) exclusively; while the centralised topology and optimisation framework are architecturally compatible with bidirectional vehicle-to-grid (V2G) operation, V2G capabilities were not modeled or evaluated due to their limited adoption by electric vehicle manufacturers and the absence of established regulatory frameworks for residential V2G in Portugal. Infrastructure sizing methodologies and compliance analyses are grounded specifically in Portuguese electrical regulations (RTIEBT), recent legislative updates (Decreto-Lei 93/2025), and condominium property law. Adaptation to other European or international jurisdictions would require systematic mapping to local electrical codes, protection coordination requirements, and property ownership frameworks, which may differ substantially in their treatment of shared infrastructure and cost allocation. Importantly, while the methodology is validated through simulation and historical data analysis, physical prototype deployment with real-time controller operation, communication infrastructure, and user interface testing has not yet been executed. This represents planned future work essential for validating real-world controller behavior, quantifying communication latency impacts, and assessing user acceptance and interaction patterns. Finally, the degradation-aware scheduling layer employs semi-empirical proxy models calibrated to literature-reported aging parameters rather than longitudinal measurements from the specific battery chemistries and thermal management systems of contemporary Portuguese EV fleet. Longitudinal validation against measured capacity fade in operational vehicles, while methodologically desirable, exceeded the temporal scope of the doctoral research period.

The findings of this thesis have direct implications for multiple stakeholder groups involved in EV integration in residential buildings:

For building owners and condominium associations, pre-cabling, combined with a centralised controller on the main feeder, sized using the $C_{f\text{-}fuzzy}$ framework, represents a pragmatic pathway to EV-readiness that defers heavy capital expenditure while serving substantially more residents than conventional approaches. The approach enables incremental deployment, residents without EVs today can join the system later without requiring additional infrastructure upgrades. The planning artefacts provided in this thesis enable informed dialogue with electrical contractors and DSOs.

For distribution system operators and regulators: Adopting non-unitary, evidence-based coincidence factors for residential EV branches can unlock material savings in grid reinforcement costs without eroding safety margins. The methodology provides a transparent, reproducible basis for updating technical norms (e.g., DMA-C13-912/N) to reflect the diversity of modern EV charging loads under managed charging regimes.

For charge point management system providers: Exposing "desired SoC by departure time" interfaces and implementing lightweight health-aware charging policies improves user satisfaction and equalises access in shared garage environments. The optimisation formulation and degradation proxies are documented with sufficient detail for integration into commercial CPMS platforms.

For policymakers: The demonstrated feasibility of high EV penetration in existing buildings without major infrastructure investment supports policy frameworks that promote building-level charging solutions alongside public network expansion. The work provides technical evidence for potential updates to building codes requiring EV-ready pre-cabling in new construction and major renovations

6.4 - Future Research Directions

Building on the validated methodology presented in this thesis, future research should pursue complementary directions that consolidate technical maturity, strengthen external validity, and accelerate real-world adoption. These directions are organised into three interconnected research tracks, each addressing distinct but mutually reinforcing aspects of system development and deployment.

Deployment, Validation, and Regulatory Integration Track

The immediate priority involves transitioning from simulation-validated methodology to operational deployment through comprehensive prototype development and field trials. This track encompasses the design, construction, and deployment of a full-stack system integrating site controller hardware, metering pathways, communication infrastructure, and electric vehicle supply equipment actuation in a pilot multi-dwelling building. Key objectives include validating real-time controller behavior under actual residential load variability, establishing commissioning procedures accessible to non-research technical staff, and quantifying system availability and resilience under seasonal demand variations and fault conditions. Expected deliverables include a technology demonstrator with reproducible bill of materials and comprehensive site deployment playbook. In parallel, regulatory alignment requires development of a detailed compliance matrix mapping the methodology to Portuguese electrical codes (RTIEBT), protection coordination and discrimination rules, and distribution system operator grid-connection requirements. Attention is needed for regulatory grey areas, including the treatment of dynamic demand limitation on shared feeders, data protection requirements for real-time telemetry systems, and condominium decision-making procedures for shared infrastructure investment. This effort should culminate in an acceptance-testing checklist suitable for use by electrical inspectors and distribution system operators during commissioning and periodic compliance verification. A critical regulatory challenge that requires policy-level attention concerns the incompatibility of the proposed approach with current taxation and contracting frameworks. Under existing Portuguese legislation, the sum of contracted power levels for individual apartments must be physically assured by the building's main feeder capacity. While this capacity is already paid for by residents through their individual electricity contracts, no regulatory mechanism currently exists to make this capacity available to a shared charging switchboard without creating new supply contracts and additional points of delivery, effectively requiring residents to pay twice for the same electrical capacity. Addressing this regulatory gap will require engagement with energy regulators and policymakers to develop frameworks that recognise shared utilisation of existing contracted capacity for building-level electric vehicle charging infrastructure.

Scientific Validation and Model Refinement Track

The second research track focuses on strengthening the empirical foundation of key model components through longitudinal field studies. Priority should be given to degradation validation through instrumentation of a volunteer electric vehicle cohort to collect time-aligned telemetry on actual charging patterns, ambient operating conditions, and periodic battery capacity assessments using standardised diagnostic protocols. Statistical analysis relating observed degradation trajectories to scheduler penalty terms and charging strategy variations would enable reparameterisation of aging model coefficients and establishment of credible intervals for battery life-extension claims, transforming proxy-based scheduling from a theoretically grounded approach to an empirically validated intervention. Complementing this validation effort, the scheduler should be extended to incorporate dynamic electricity tariffs and distribution system operator flexibility service requests while maintaining strict primacy of user mobility requirements and battery health constraints. This extension would enable evaluation of achievable flexibility envelopes, the range of load-shifting and peak-shaving services the building fleet can provide without compromising primary objectives, and quantification of economic benefits under representative Portuguese tariff structures, including time-of-use rates and potential future real-time pricing mechanisms.

Technical Maturity and Ecosystem Integration Track

Achieving production-ready system maturity requires systematic attention to interoperability and cybersecurity. This track encompasses engineering of protocol compliance with OCPP, ISO 15118 vehicle-to-infrastructure communication standards, and relevant smart charging specifications. Adversarial security testing and development of cybersecurity hardening guidelines should be conducted to identify and mitigate vulnerabilities in communication pathways, authentication mechanisms, and data handling procedures. These efforts should be consolidated into a certification framework suitable for third-party assessment, enabling independent verification of security posture and standards compliance. Additionally, the system architecture should be extended to support commercial and fleet-operator contexts, where predictable duty cycles, centralised management structures, and professional operation may enable even greater optimisation benefits than residential settings while presenting distinct technical requirements around availability guarantees and service-level agreements.

Cross-Cutting Research Elements

Three cross-cutting themes would enhance the value and societal acceptability of all research tracks. First, data governance and ethics protocols must establish clear consent flows for collection of mobility and energy consumption data, robust anonymisation procedures to protect individual privacy, and transparent retention and deletion policies aligned with European data protection regulations. Second, user experience research should employ A/B testing of alternative interface designs and feedback mechanisms to optimise user understanding, maintain transparency in charging allocation decisions, and sustain perceived equity over extended operational periods, factors critical for long-term user acceptance in shared residential environments. Third, systematic investigation of the methodology's transferability to commercial fleet contexts, including delivery vehicles, corporate fleets, and shared mobility services, would leverage the architectural similarity of centralised charging while addressing sector-specific requirements around vehicle availability, duty cycle predictability, and operational constraints.

6.5 - Broader Impact and Closing Remarks

Portugal's electric vehicle market has reached an inflection point, with battery electric vehicles capturing 22.5% of new car registrations in the first quarter of 2025, a proportion exceeding that of most European peers. Yet the nation's housing stock, where over 70% of urban residents live in multi-dwelling buildings constructed before EV adoption was contemplated, presents a structural barrier to continued growth. Without practical charging solutions for existing condominiums, the pace of EV adoption may be constrained by infrastructure rather than vehicle availability or consumer willingness.

This thesis demonstrates that the barrier is surmountable. By combining centralised optimisation, evidence-based infrastructure sizing, and user-centric operation, high EV penetration can be achieved in existing buildings without the heavy capital investment typically assumed to be necessary. The methodology respects the technical constraints of Portuguese LV networks, the legal frameworks governing condominiums, and the practical realities of shared residential infrastructure.

The contributions extend beyond the Portuguese context. The fundamental insight, that idle building capacity can be harvested through intelligent coordination rather than brute-force infrastructure expansion, applies wherever multi-dwelling housing predominates, and EV adoption is accelerating. The coincidence factor framework and planning artefacts provide a template for adaptation to other regulatory environments.

Looking forward, the transition to electric mobility will require coordination across multiple scales, from individual vehicles and buildings to distribution networks and national energy systems. The building-level solutions developed in this thesis represent one essential component of this multi-scale transition, enabling the "last hundred meters" of charging infrastructure that connects vehicle batteries to the broader grid.

In conclusion, this thesis has shown that large-scale EV integration in existing condominiums is technically feasible, economically proportionate, and operationally practical. By harvesting idle capacity with centralised control, sizing infrastructure with realistic coincidence factors, and protecting battery assets through degradation-aware scheduling, the proposed approach transforms potential trade-offs into co-optimised solutions. The result is a pragmatic pathway to accelerate electric mobility across the existing housing stock, advancing Portugal's transport decarbonisation while demonstrating approaches applicable to the broader European challenge of sustainable urban mobility.

References

[1] N. Haque and A. M. Saleque, “Impacts of Electric Vehicle Charging On Distribution Grid,” *AIUB Journal of Science and Engineering (AJSE)*, vol. 18, no. 3, pp. 88–96, Dec. 2019, doi: 10.53799/ajse.v18i3.57.

[2] Z. Hoover, N. Florian, E. Polymeneas, and S. Sahdev, “How charging in buildings can power up the electric-vehicle industry,” Jan. 2021.

[3] S. Á. Funke, F. Sprei, T. Gnann, and P. Plötz, “How much charging infrastructure do electric vehicles need? A review of the evidence and international comparison,” *Transp Res D Transp Environ*, vol. 77, pp. 224–242, Dec. 2019, doi: 10.1016/j.trd.2019.10.024.

[4] J. A. P. Lopes, F. J. Soares, and P. M. R. Almeida, “Integration of Electric Vehicles in the Electric Power System,” *Proceedings of the IEEE*, vol. 99, no. 1, pp. 168–183, Jan. 2011, doi: 10.1109/JPROC.2010.2066250.

[5] H. Engel, R. Hensley, S. Knupfer, and S. Sahdev, “The potential impact of electric vehicles on global energy systems.” Accessed: Mar. 11, 2025. [Online]. Available: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/the-potential-impact-of-electric-vehicles-on-global-energy-systems#/

[6] IEA, “Global EV Outlook 2024.,” Paris, 2024.

[7] IEA, “Global EV Outlook 2025.,” Paris, 2025.

[8] BloombergNEF, “Electric Vehicle Outlook 2025,” New York, 2024.

[9] ACEA, “New passenger car registrations in the EU — BEV market share update.,” ACEA.

[10] European Comission, “REGULATION (EU) 2023/1804 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL ,” Sep. 23, 2023, *European Comission, Brussels*.

[11] Open Charge Alliance, “What is new in OCPP 2.0,” Arnhem, 2023.

[12] European Commission - EAFO, “Portugal January 2025: BEVs Reach Record 22.5% Market Share,” News Article. Accessed: Dec. 16, 2025. [Online]. Available: https://alternative-fuels-observatory.ec.europa.eu/general-information/news/portugal-january-2025-bevs-reach-record-225-market-share

[13] AUTOVISTA24, “Portugal packs a punch in the European EV market.,” AUTOVISTA24. Accessed: Dec. 16, 2025. [Online]. Available: https://autovista24.autovistagroup.com/news/portugal-packs-a-punch-in-the-european-ev-market/

[14] FOCUS2MOVE, “Portugal 2025. Q3 Opens On An Upward Note, EV Sector On The Rise,” FOCUS2MOVE. Accessed: Dec. 16, 2025. [Online]. Available: https://www.focus2move.com/portuguese-auto-market/

[15] I. Platini, “eMobility Portugal report: Electric car sales in 2024 already surpass those of 2022,” Mobility Portal Europe. Accessed: Dec. 16, 2025. [Online]. Available: https://mobilityportal.eu/emobility-portugal-report-2024/

[16] A. Almutairi, N. Albagami, S. Almesned, O. Alrumayh, and H. Malik, “Optimal management of electric vehicle charging loads for enhanced sustainability in shared residential buildings,” *Front Energy Res*, vol. 12, May 2024, doi: 10.3389/fenrg.2024.1396899.

[17] Instituto Nacional de Estatística, “2021 CENSUS - Final Results Release,” Lisbon, Nov. 2022.

[18] Instituto Nacional de Estatística, “2021 CENSUS - Preliminary Results Release,” Lisbon, Jul. 2021. Accessed: Dec. 16, 2025. [Online]. Available: https://www.ine.pt/ngt_server/attachfileu.jsp?look_parentBoui=521457903&att_display=n&att_download=y

[19] Wikipedia, “Housing in Portugal,” Wikipedia. Accessed: Dec. 16, 2025. [Online]. Available: https://en.wikipedia.org/wiki/Housing_in_Portugal

[20] J. Muller, “Charging an electric car at home isn’t easy for many,” Axios. [Online]. Available: https://www.axios.com/2022/03/23/electric-car-charging-at-home-difficult

[21] Lloyds Bank, “Nearly half of UK homes unsuitable for electric cars,” Press Release. [Online]. Available: https://www.lloydsbankinggroup.com/assets/pdfs/media/press-

releases/2022-press-releases/lloyds-bank/221230-lloyds-bank-many-uk-homes-not-suitable-for-electric-vehicles.pdf
[22] Open Charge Alliance, “Improved security for OCPP 1.6-J,” Arnhem, 2022.
[23] ISO, “ ISO 15118-20:2022 Road vehicles — Vehicle to grid communication interfacePart 20: 2nd generation network layer and application layer requirements,” Geneva, 2022.
[24] EVRoaming Foundation, “OCPI 2.2.1-d3-pre1,” Houten, 2023.
[25] International Energy Agency, “Portugal,” International Energy Agency. Accessed: Dec. 16, 2025. [Online]. Available: https://www.iea.org/countries/portugal
[26] E-REDES, “About us,” E-REDES. Accessed: Dec. 16, 2025. [Online]. Available: https://www.e-redes.pt/en/about-us
[27] M. Green, “MOBI.E maps Portugal’s EV success,” Infrastructure News. Accessed: Dec. 16, 2025. [Online]. Available: https://www.evinfrastructurenews.com/ev-networks/mobi-e-maps-portugal-s-ev-success
[28] P. Magic, “Portugal’s EV Charging Landscape in 2025: Accelerating Towards a Sustainable Future,” ZONIQ. Accessed: Dec. 16, 2025. [Online]. Available: https://www.zoniq.io/post/portugal-ev-charging-landscape-in-2025-accelerating-towards-a-sustainable-future
[29] MOBI.E, “Charging a Vehicle,” MOBI.E. Accessed: Dec. 16, 2025. [Online]. Available: https://www.mobie.pt/en/
[30] European Comission, “DIRECTIVE (EU) 2024/1275 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL ,” 2024, *European Comission, Brussels*.
[31] Entidade Reguladora dos Serviços Energéticos, “Tarifas da Entidade Gestora da Rede de Mobilidade Elétrica para 2025,” 2025, *República Portuguesa, Lisboa*.
[32] Presidência do Conselho de Ministros, “Decreto-Lei n.o 93/2025, de 14 de agosto,” 2025, *República Portuguesa, Lisbon*.
[33] M. Carneiro Pacheco, M. Cassiano Neves, and B. Cunha Ferreira, “Novo Regime da Mobilidade Elétrica,” CMS Law. Accessed: Dec. 16, 2025. [Online]. Available: https://cms.law/pt/prt/publication/novo-regime-da-mobilidade-eletrica
[34] S. Magno, “Nova lei para mobilidade eléctrica: o que muda para os utilisadores,” *Público*, Porto, Aug. 14, 2025. Accessed: Dec. 16, 2025. [Online]. Available: https://www.publico.pt/2025/08/14/enter/noticia/nova-lei-mobilidade-electrica-muda-utilisadores-2143964#
[35] PWC Portugal, “Novo Regime Jurídico da Mobilidade Elétrica,” PWC Portugal. Accessed: Dec. 16, 2025. [Online]. Available: https://www.pwc.pt/pt/servicos/legal/legal-flash/novo-regime-juridico-mobilidade-eletrica.html
[36] The Resident, “Installation of electric car charging points,” The Resident. Accessed: Dec. 16, 2025. [Online]. Available: https://www.portugalresident.com/installation-of-electric-car-charging-points/
[37] MOBI.E, “Support for chargers in condominiums,” MOBI.E. Accessed: Dec. 16, 2025. [Online]. Available: https://mobie.pt/en/support-for-chargers-in-condominiums
[38] M. Ecker *et al.*, “Development of a lifetime prediction model for lithium-ion batteries based on extended accelerated aging test data,” *J Power Sources*, vol. 215, pp. 248–257, Oct. 2012, doi: 10.1016/j.jpowsour.2012.05.012.
[39] S. A. EDP Distribuição – Energia, “DMA-C13-912/N,” Lisboa, Nov. 2011.
[40] DGEG, “Regras Técnicas das Instalações Eléctricas de Baixa Tensão,” 2025, *DGEG, Lisbon*.
[41] ISQ, “Projecto Elétricto de Baixa Tensão,” 2025, *ISQ*. Accessed: Dec. 16, 2025. [Online]. Available: https://academy.isq.pt/informacaocurso.aspx?id=91
[42] H. D. Nogueira, *ABC das Regras Técnicas*, 2.a Edição. Engebook, 2019.
[43] International Electrotechnical Commission, “IEC 60364-5-52,” 2009, *International Electrotechnical Commission*. Accessed: Dec. 16, 2025. [Online]. Available: https://webstore.iec.ch/en/publication/1878
[44] European Union, “Directive 2014/32/EU,” Feb. 26, 2014, *EUR-Lex*. Accessed: Dec. 16, 2025. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32014L0032
[45] C. V. S. Soares, *nstalações eléctricas de baixa tensão : projecto, execução e exploração : origem e interpretação das RTIEBT - 1a edição e principais diferenças face ao “740/74.”* Lisbon: DGEG, 2006.

[46] O. do T. e E. Ministério do Ambiente, “Portaria n.o 252/2015,” Aug. 19, 2015, *Diário da República, Lisbon*. Accessed: Dec. 16, 2025. [Online]. Available: https://diariodarepublica.pt/dr/detalhe/portaria/252-2015-70055493

[47] Ministério da Economia e da Inovação, “Portaria n.o 949-A/2006,” *Diário da República*, no. 2, p. 190, 2006.

[48] ELRA20, “Applicable Law - Condominium,” 2017, *ELRA20*. Accessed: Dec. 16, 2025. [Online]. Available: https://www.elra.eu/contact-point-contribution/portugal/applicable-law-3/

[49] República Portuguesa, “Código Civil Português - 1414-1438, A,” Nov. 25, 1966, *Diário da República, Lisbon*. Accessed: Dec. 16, 2025. [Online]. Available: https://diariodarepublica.pt/dr/legislacao-consolidada/decreto-lei/1966-34509075-49923475

[50] South Port Realty, “Condominium: Common parts, rules and expenses,” South Port Realty. Accessed: Dec. 16, 2025. [Online]. Available: https://www.spralgarve.com/en/noticias/show/condominium-common-parts-rules-and-expenses_42/

[51] The Portugal News, “New homes must have electric car charging points,” The Portugal News. Accessed: Dec. 16, 2025. [Online]. Available: https://www.theportugalnews.com/news/2024-05-21/new-homes-must-have-electric-car-charging-points/89040

[52] E-REDES, “Frequently Asked Questions,” E-REDES. Accessed: Dec. 16, 2025. [Online]. Available: https://www.e-redes.pt/en/can-we-help/frequently-asked-questions

[53] ANARI, “Portugal Powers Ahead: Europe’s New EV Charging Hotspot — A Deep Dive into Policy, Market, and Strategic Levers,” ANARI. Accessed: Dec. 16, 2025. [Online]. Available: https://www.anariev.com/portugal-powers-ahead-europe-ev-charging-hotspot/

[54] J. D. Kim, “Insights into residential EV charging behaviour using energy meter data,” *Energy Policy*, vol. 129, pp. 610–618, Jun. 2019, doi: 10.1016/j.enpol.2019.02.049.

[55] J. A. Peças Lopes, P. M. Rocha Almeida, and J. Soares, “Electric Vehicles Grid Integration with RES – Definition of Regulatory and Policy Solutions,” Porto, 2021.

[56] Y. Li and A. Jenn, “Impact of electric vehicle charging demand on power distribution grid congestion,” *Proceedings of the National Academy of Sciences*, vol. 121, no. 18, Apr. 2024, doi: 10.1073/pnas.2317599121.

[57] W. Zhan, Y. Liao, J. Deng, Z. Wang, and S. Yeh, “Large-scale empirical study of electric vehicle usage patterns and charging infrastructure needs,” *npj Sustainable Mobility and Transport*, vol. 2, no. 1, p. 9, Mar. 2025, doi: 10.1038/s44333-024-00023-3.

[58] M. Hagenmaier, C. Wagener, J. Bert, J. Carrasco, N. Niese, and A. Wang, “What Electric Vehicle Owners Really Want from Charging Networks,” 2023.

[59] J. H. Lee, D. Chakraborty, S. J. Hardman, and G. Tal, “Exploring electric vehicle charging patterns: Mixed usage of charging infrastructure,” *Transp Res D Transp Environ*, vol. 79, p. 102249, Feb. 2020, doi: 10.1016/j.trd.2020.102249.

[60] M. Etxandi-Santolaya, L. Canals Casals, T. Montes, and C. Corchero, “Are electric vehicle batteries being underused? A review of current practices and sources of circularity,” *J Environ Manage*, vol. 338, no. December 2022, 2023, doi: 10.1016/j.jenvman.2023.117814.

[61] E. D. Kostopoulos, G. C. Spyropoulos, and J. K. Kaldellis, “Real-world study for the optimal charging of electric vehicles,” *Energy Reports*, vol. 6, pp. 418–426, 2020, doi: 10.1016/j.egyr.2019.12.008.

[62] J. Smart and S. Schey, “Battery Electric Vehicle Driving and Charging Behaviour Observed Early in The EV Project,” *SAE International Journal of Alternative Powertrains*, vol. 1, no. 1, pp. 2012-01–0199, Apr. 2012, doi: 10.4271/2012-01-0199.

[63] R. Su, E. C. McBride, and K. G. Goulias, “Unveiling daily activity pattern differences between telecommuters and commuters using human mobility motifs and sequence analysis,” *Transp Res Part A Policy Pract*, vol. 147, no. February, pp. 106–132, 2021, doi: 10.1016/j.tra.2021.03.002.

[64] J. M. Branco de Brito, “Caracterização da flutuação do tráfego na cidade de Lisboa,” Universidade Nova de Lisboa, 2012.

[65] D. Cui *et al.*, “Battery electric vehicle usage pattern analysis driven by massive real-world data,” *Energy*, vol. 250, p. 123837, 2022, doi: 10.1016/j.energy.2022.123837.

[66] A. A. Visaria, A. F. Jensen, M. Thorhauge, and S. E. Mabit, “User preferences for EV charging, pricing schemes, and charging infrastructure,” *Transp Res Part A Policy Pract*, vol. 165, no. August, pp. 120–143, 2022, doi: 10.1016/j.tra.2022.08.013.

[67] H. Bin Moon, S. Y. Park, C. Jeong, and J. Lee, “Forecasting electricity demand of electric vehicles by analyzing consumers’ charging patterns,” *Transp Res D Transp Environ*, vol. 62, no. February, pp. 64–79, 2018, doi: 10.1016/j.trd.2018.02.009.

[68] J. R. Helmus, M. H. Lees, and R. van den Hoed, “A data driven typology of electric vehicle user types and charging sessions,” *Transp Res Part C Emerg Technol*, vol. 115, p. 102637, Jun. 2020, doi: 10.1016/j.trc.2020.102637.

[69] T. Ye, S. Liu, and E. Kontou, “Managed residential electric vehicle charging minimises electricity bills while meeting driver and community preferences,” *Transp Policy (Oxf)*, vol. 149, pp. 122–138, Apr. 2024, doi: 10.1016/j.tranpol.2024.01.022.

[70] European Alternative Fuels Observatory, “Electric vehicle model statistics,” Directorate-General for Mobility and Transport. [Online]. Available: https://alternative-fuels-observatory.ec.europa.eu/policymakers-and-public-authorities/electric-vehicle-model-statistics

[71] J. A. Sanguesa, V. Torres-Sanz, P. Garrido, F. J. Martinez, and J. M. Marquez-Barja, “A review on electric vehicles: Technologies and challenges,” *Smart Cities*, vol. 4, no. 1, pp. 372–404, 2021, doi: 10.3390/smartcities4010022.

[72] S. Hussain *et al.*, “A Heuristic Charging Cost Optimisation Algorithm for Residential Charging of Electric Vehicles,” *Energies (Basel)*, vol. 15, no. 4, pp. 1–18, 2022, doi: 10.3390/en15041304.

[73] Z. J. Lee, J. Z. F. Pang, and S. H. Low, “Pricing EV charging service with demand charge,” *Electric Power Systems Research*, vol. 189, no. April, p. 106694, 2020, doi: 10.1016/j.epsr.2020.106694.

[74] Q. Dang, D. Wu, and B. Boulet, “EV Charging Management with ANN-Based Electricity Price Forecasting,” *2020 IEEE Transportation Electrification Conference and Expo, ITEC 2020*, pp. 626–630, 2020, doi: 10.1109/ITEC48692.2020.9161659.

[75] R. Tu, Y. (Jessie) Gai, B. Farooq, D. Posen, and M. Hatzopoulou, “Electric vehicle charging optimisation to minimise marginal greenhouse gas emissions from power generation,” *Appl Energy*, vol. 277, no. July 2020, p. 115517, 2020, doi: 10.1016/j.apenergy.2020.115517.

[76] D. Xiao, S. An, H. Cai, J. Wang, and H. Cai, “An optimisation model for electric vehicle charging infrastructure planning considering queuing behaviour with finite queue length,” *J Energy Storage*, vol. 29, p. 101317, Jun. 2020, doi: 10.1016/j.est.2020.101317.

[77] R. Ghotge, Y. Snow, S. Farahani, Z. Lukszo, and A. van Wijk, “Optimised scheduling of EV charging in solar parking lots for local peak reduction under EV demand uncertainty,” *Energies (Basel)*, vol. 13, no. 5, 2020, doi: 10.3390/en13051275.

[78] B. Sun, “A multi-objective optimisation model for fast electric vehicle charging stations with wind, PV power and energy storage,” *J Clean Prod*, vol. 288, p. 125564, 2021, doi: 10.1016/j.jclepro.2020.125564.

[79] N. A. El-Taweel, H. Farag, M. F. Shaaban, and M. E. Alsharidah, “Optimisation Model for EV Charging Stations With PV Farm Transactive Energy,” *IEEE Trans Industr Inform*, vol. 18, no. 7, pp. 4608–4621, 2022, doi: 10.1109/TII.2021.3114276.

[80] B. Debnath, S. Biswas, and M. F. Uddin, “Optimisation of Electric Vehicle Charging to Shave Peak Load for Integration in Smart Grid,” *2020 IEEE Region 10 Symposium, TENSYMP 2020*, no. June, pp. 483–488, 2020, doi: 10.1109/TENSYMP50017.2020.9231029.

[81] N. B. G. Brinkel, W. L. Schram, T. A. AlSkaif, I. Lampropoulos, and W. G. J. H. M. van Sark, “Should we reinforce the grid? Cost and emission optimisation of electric vehicle charging under different transformer limits,” *Appl Energy*, vol. 276, no. May, p. 115285, 2020, doi: 10.1016/j.apenergy.2020.115285.

[82] D. Lopez-Behar, M. Tran, T. Froese, J. R. Mayaud, O. E. Herrera, and W. Merida, “Charging infrastructure for electric vehicles in Multi-Unit Residential Buildings: Mapping feedbacks and policy recommendations,” *Energy Policy*, vol. 126, no. May 2018, pp. 444–451, 2019, doi: 10.1016/j.enpol.2018.10.030.

[83] Direção Geral da Energia, “Parte 5 / Secção 51,” in *Regras Técnicas de Instalações Eléctricas de Baixa Tensão*, 2000, p. 1 to 736.

[84] National Fire Protection Association, “Section 220,” in *National Electrical Code*, 2023.

[85] J. S. Edge *et al.*, “Lithium ion battery degradation: what you need to know,” *Physical Chemistry Chemical Physics*, vol. 23, no. 14, pp. 8200–8221, 2021, doi: 10.1039/D1CP00359C.
[86] D. Werner, S. Paarmann, and T. Wetzel, “Calendar Aging of Li-Ion Cells—Experimental Investigation and Empirical Correlation,” *Batteries*, vol. 7, no. 2, p. 28, Apr. 2021, doi: 10.3390/batteries7020028.
[87] P. M. Attia *et al.*, “Closed-loop optimisation of fast-charging protocols for batteries with machine learning,” *Nature*, vol. 578, no. 7795, pp. 397–402, Feb. 2020, doi: 10.1038/s41586-020-1994-5.
[88] A. Bhowmik and T. Vegge, “AI Fast Track to Battery Fast Charge,” *Joule*, vol. 4, no. 4, pp. 717–719, Apr. 2020, doi: 10.1016/j.joule.2020.03.016.
[89] G. Yarimca and E. Cetkin, “Review of Cell Level Battery (Calendar and Cycling) Aging Models: Electric Vehicles,” *Batteries*, vol. 10, no. 11, p. 374, Oct. 2024, doi: 10.3390/batteries10110374.
[90] L. Lam and P. Bauer, “Practical Capacity Fading Model for Li-Ion Battery Cells in Electric Vehicles,” *IEEE Trans Power Electron*, vol. 28, no. 12, pp. 5910–5918, Dec. 2013, doi: 10.1109/TPEL.2012.2235083.
[91] O. Juhlin, “Modelling of Battery Degradation in Electrified Vehicles,” 2016.
[92] M. Naumann, F. B. Spingler, and A. Jossen, “Analysis and modelling of cycle aging of a commercial LiFePO4/graphite cell,” *J Power Sources*, vol. 451, p. 227666, Mar. 2020, doi: 10.1016/j.jpowsour.2019.227666.
[93] H. Miranda and J. Witt, “Update: Scientists Reveal how EV Fast Charging Impacts Battery Health,” Recurrent.
[94] T. Montes, F. Pinsach Batet, L. Igualada, and J. Eichman, “Degradation-conscious charge management: Comparison of different techniques to include battery degradation in Electric Vehicle Charging Optimisation,” *J Energy Storage*, vol. 88, p. 111560, May 2024, doi: 10.1016/j.est.2024.111560.
[95] V. B. Truong and L. B. Le, “Building energy management and Electric Vehicle charging considering battery degradation and random vehicles’ arrivals and departures,” *J Energy Storage*, vol. 64, p. 107141, Aug. 2023, doi: 10.1016/j.est.2023.107141.
[96] Geotab, “New Geotab data highlights how EV batteries can last 20 years or more,” *Geotab*, London, Sep. 17, 2024.
[97] M. Yakub, “EV batteries are likely to outlive vehicles they’re in, finds latest Geotab data,” *Electric Autonomy*, Ontario, Sep. 18, 2024.
[98] P. Krein, “Thinking About Ubiquitous Slow Charging for Daily Electric Vehicle Energy: Fast charging has few benefits at work and home.,” *IEEE Electrification Magazine*, vol. 13, no. 3, pp. 36–43, Sep. 2025, doi: 10.1109/MELE.2025.3589210.
[99] A. A. Rastad, H. S. Busengdal, and E. Hiep, “Report on Consumer Behaviour,” 2023.
[100] A. Palomino and M. Parvania, “Probabilistic Impact Analysis of Residential Electric Vehicle Charging on Distribution Transformers,” in *2018 North American Power Symposium (NAPS)*, IEEE, Sep. 2018, pp. 1–6. doi: 10.1109/NAPS.2018.8600630.
[101] J. Quiros-Tortos, L. F. Ochoa, and B. Lees, “A statistical analysis of EV charging behaviour in the UK,” in *2015 IEEE PES Innovative Smart Grid Technologies Latin America (ISGT LATAM)*, IEEE, Oct. 2015, pp. 445–449. doi: 10.1109/ISGT-LA.2015.7381196.
[102] L. Calearo, A. Thingvad, K. Suzuki, and M. Marinelli, “Grid Loading Due to EV Charging Profiles Based on Pseudo-Real Driving Pattern and User Behaviour,” *IEEE Transactions on Transportation Electrification*, vol. 5, no. 3, pp. 683–694, Sep. 2019, doi: 10.1109/TTE.2019.2921854.
[103] M. G. Flammini, G. Prettico, A. Julea, G. Fulli, A. Mazza, and G. Chicco, “Statistical characterisation of the real transaction data gathered from electric vehicle charging stations,” *Electric Power Systems Research*, vol. 166, pp. 136–150, Jan. 2019, doi: 10.1016/j.epsr.2018.09.022.
[104] Xin Tong, Chunlin Guo, Xiaoyan Yang, and Chenchen Chen, “Research on characteristics of electric vehicle charging load and distribution network supportability,” in *2016 IEEE PES Asia-Pacific Power and Energy Engineering Conference (APPEEC)*, IEEE, Oct. 2016, pp. 1539–1542. doi: 10.1109/APPEEC.2016.7779748.
[105] M. Aunedi, M. Woolf, G. Strbac, O. Babalola, and M. Clark, “Characteristic Demand Profiles of Residential and Commercial EV Users and Opportunities for Smart

Charging," in *Internation Conference on Electricity Distribution*, Lyon: CIRED, 2015, p. 1088.

[106] N. Thie, E. Junge, S. Hillenbrand, and M. Konermann, "Evaluation of Grid Compatible Load Management Concepts for E-Mobility in Distribution Grids," in *2019 54th International Universities Power Engineering Conference (UPEC)*, IEEE, Sep. 2019, pp. 1–5. doi: 10.1109/UPEC.2019.8893580.

[107] J. Bollerslev *et al.*, "Coincidence Factors for Domestic EV Charging From Driving and Plug-In Behaviour," *IEEE Transactions on Transportation Electrification*, vol. 8, no. 1, pp. 808–819, Mar. 2022, doi: 10.1109/TTE.2021.3088275.

[108] S. Carvalhosa, J. R. Ferreira, and R. E. Araújo, "Electric Vehicle Charging Method for Existing Residential Condominiums," *IEEE Access*, pp. 1–1, 2024, doi: 10.1109/ACCESS.2024.3491379.

[109] K. Berg and M. Löschenbrand, "Data set of a Norwegian energy community," Sep. 13, 2023.

[110] J. Bollerslev *et al.*, "Coincidence Factors for Domestic EV Charging From Driving and Plug-In Behaviour," *IEEE Transactions on Transportation Electrification*, vol. 8, no. 1, pp. 808–819, Mar. 2022, doi: 10.1109/TTE.2021.3088275.

[111] A. Opstad, B. H. Bakken, G. Doorman, H. S. Nygard, and K. Sevdari, "CSE N°35 December 2024," Dec. 2024.

[112] I. Jokinen and M. Lehtonen, "Modelling of Electric Vehicle Charging Demand and Coincidence of Large-Scale Charging Loads in Different Charging Locations," *IEEE Access*, vol. 11, pp. 114291–114315, 2023, doi: 10.1109/ACCESS.2023.3322278.

[113] H. Fani, M. U. Hashmi, and G. Deconinck, "Impact of electric vehicle charging simultaneity factor on the hosting capacity of LV feeder," *Sustainable Energy, Grids and Networks*, vol. 40, Dec. 2024, doi: 10.1016/j.segan.2024.101581.

[114] M. Safayatullah *et al.*, "A Comprehensive Review on Charging Topologies and Power Electronic Converter Solutions for Electric Vehicles," *Journal of Modern Power Systems and Clean Energy*, vol. XX, no. Xx, pp. 1–19, 2022, doi: 10.35833/MPCE.2023.000107.

[115] P. Heid and T. P. Wihler, "Adaptive iterative linearization Galerkin methods for nonlinear problems," *Math Comput*, vol. 89, 2020.

[116] M. Asghari, A. M. Fathollahi-Fard, S. M. J. Mirzapour Al-E-Hashem, and M. A. Dulebenets, "Transformation and Linearization Techniques in Optimisation: A State-of-the-Art Survey," *Mathematics*, vol. 10, no. 2, 2022, doi: 10.3390/math10020283.

[117] D. Kraft, *A software package for Sequential Quadratic Programming*. Oberpfaffenhofen: Institut für Dynamik der Flugsysteme, 1988.

[118] D. Kraft, "Algorithm 733: TOMP–Fortran Modules for Optimal Control Calculations," *ACM Transactions on Mathematical Software (TOMS)*, vol. 20, no. 3, pp. 262–281, 1994, doi: 10.1145/192115.192124.

[119] R. Ramon, "Battery State of Charge calculation with EPC Converters," 2020.

[120] L. Ma, C. Hu, and F. Cheng, "State of Charge and State of Energy Estimation for Lithium-Ion Batteries Based on a Long Short-Term Memory Neural Network," *J Energy Storage*, vol. 37, no. December 2020, 2021, doi: 10.1016/j.est.2021.102440.

[121] R. Bollini, "Understanding Charge-Discharge Curves of Li-ion Cells," *EVreporter*, 2022.

[122] M. Eider and A. Berl, "Dynamic EV battery health recommendations," *e-Energy 2018 - Proceedings of the 9th ACM International Conference on Future Energy Systems*, pp. 586–592, 2018, doi: 10.1145/3208903.3213896.

[123] F. Leng, Z. Wei, C. M. Tan, and R. Yazami, "Hierarchical degradation processes in lithium-ion batteries during ageing," *Electrochim Acta*, vol. 256, pp. 52–62, 2017, doi: 10.1016/j.electacta.2017.10.007.

[124] K. Schittkowski, "The nonlinear programming method of Wilson, Han, and Powell with an augmented Lagrangian type line search function - Part 1: Convergence analysis," *Numer Math (Heidelb)*, vol. 38, no. 1, pp. 83–114, 1982, doi: 10.1007/BF01395810.

[125] J. Frédéric Bonnans, J. Charles Gilbert, C. Lemaréchal, and C. A. Sagastizábal, "Numerical optimisation: Theoretical and practical aspects," *Numerical Optimisation: Theoretical and Practical Aspects*, pp. 1–494, 2006, doi: 10.1007/978-3-540-35447-5.

[126] Y. Ma, N. Zhang, and J. Li, *Improved Sequential Least Squares Programming-Driven Feasible Path Algorithm for Process Optimisation*, vol. 51. Elsevier Masson SAS, 2022. doi: 10.1016/B978-0-323-95879-0.50214-9.

[127] L. Armijo, “Minimisation of functions having lipschitz continuous first partial derivatives,” *Pac J Math*, vol. 16, no. 1, pp. 1–3, 1966, doi: 10.2140/pjm.1966.16.1.
[128] A. Nelson, C. Harris, C. Baumgarten, and et. al, “SciPy,” minimise(method=’SLSQP’). Accessed: Feb. 14, 2024. [Online]. Available: https://scipy.org/
[129] NumPy Team, “NumPy.” [Online]. Available: https://numpy.org/
[130] Pandas Team, “Python Data Analysis Library.” [Online]. Available: https://pandas.pydata.org/
[131] Python, “Python Random.” [Online]. Available: https://docs.python.org/3/library/random.html
[132] Matplotlib Dev Team, “Matplotlib.” [Online]. Available: https://matplotlib.org/
[133] SD Workx, “More than 20 % of Europeans Commute at Least 90 Minutes Daily,” 2018.
[134] EV Database, “Energy consumption of full electric vehicles.” [Online]. Available: https://ev-database.org/cheatsheet/energy-consumption-electric-car
[135] J. R. Helmus, M. H. Lees, and R. van den Hoed, “A data driven typology of electric vehicle user types and charging sessions,” *Transp Res Part C Emerg Technol*, vol. 115, no. November 2019, p. 102637, 2020, doi: 10.1016/j.trc.2020.102637.
[136] A. Ndiaye, R. German, A. Bouscayrol, M. Gaetani-Liseo, P. Venet, and E. Castex, “Impact of the User Charging Practice on the Battery Aging in an Electric Vehicle,” *IEEE Trans Veh Technol*, vol. 73, no. 4, pp. 4578–4588, Apr. 2024, doi: 10.1109/TVT.2024.3356116.
[137] D. Pevec, J. Babic, A. Carvalho, Y. Ghiassi-Farrokhfal, W. Ketter, and V. Podobnik, “Electric Vehicle Range Anxiety: An Obstacle for the Personal Transportation (R)evolution?,” in *2019 4th International Conference on Smart and Sustainable Technologies (SpliTech)*, IEEE, Jun. 2019, pp. 1–8. doi: 10.23919/SpliTech.2019.8783178.
[138] H. WU and W. YANG, “A Case Study on the Intersection of EV Charge UX and User Onboarding Design,” *International Symposium on Affective Science and Engineering*, vol. ISASE2024, no. 0, pp. 1–4, 2024, doi: 10.5057/isase.2024-C000034.
[139] V. Karkuzhali, P. Rangarajan, V. Tamilselvi, and P. Kavitha, “Analysis of battery management system issues in electric vehicles,” *IOP Conf Ser Mater Sci Eng*, vol. 994, no. 1, p. 012013, Dec. 2020, doi: 10.1088/1757-899X/994/1/012013.
[140] F. Sufang Lu, “What is the internal resistance of lithium-ion batteries (LiFePO4-LFP)?” Accessed: Mar. 13, 2025. [Online]. Available: https://www.linkedin.com/pulse/what-internal-resistance-lithium-ion-batteries-fancy-sufang-lu-ucb2c
[141] Y. A. Çengel and A. J. Ghajar, *Heat and Mass Transfer - Fundamentals and Applications*, 5th ed. New York: Mc-Graw-Hill Education, 2015.
[142] H. E. Perez, X. Hu, S. Dey, and S. J. Moura, “Optimal Charging of Li-Ion Batteries With Coupled Electro-Thermal-Aging Dynamics,” *IEEE Trans Veh Technol*, vol. 66, no. 9, pp. 7761–7770, Sep. 2017, doi: 10.1109/TVT.2017.2676044.
[143] S. Appleton and A. Fotouhi, “A Model-Based Battery Charging Optimisation Framework for Proper Trade-offs Between Time and Degradation,” *Automotive Innovation*, vol. 6, pp. 204–219, May 2023.
[144] S. Carvalhosa, J. R. Ferreira, and R. E. Araújo, “Electric Vehicle Charging Method for Existing Residential Condominiums,” *IEEE Access*, pp. 1–1, 2024, doi: 10.1109/ACCESS.2024.3491379.
[145] M. Schimpe *et al.*, “Energy efficiency evaluation of a stationary lithium-ion battery container storage system via electro-thermal modelling and detailed component analysis,” *Appl Energy*, vol. 210, pp. 211–229, Jan. 2018, doi: 10.1016/j.apenergy.2017.10.129.
[146] J.-H. Choi, S.-Y. Park, H.-C. Lee, S.-J. Lee, G. Kim, and J.-E. Byun, “Heat Transfer Simulation and Analysis of Thermal Battery,” *Technology and Economics of Smart Grids and Sustainable Energy*, vol. 7, no. 1, p. 12, Mar. 2022, doi: 10.1007/s40866-022-00126-1.
[147] X. Zhang *et al.*, “Evaluation of convective heat transfer coefficient and specific heat capacity of a lithium-ion battery using infrared camera and lumped capacitance method,” *J Power Sources*, vol. 412, pp. 552–558, Feb. 2019, doi: 10.1016/j.jpowsour.2018.11.064.
[148] T. Mesbahi, R. B. Sugrañes, R. Bakri, and P. Bartholomeüs, “Coupled electro-thermal modelling of lithium-ion batteries for electric vehicle application,” *J Energy Storage*, vol. 35, p. 102260, Mar. 2021, doi: 10.1016/j.est.2021.102260.

[149] P. Keil and A. Jossen, "Charging protocols for lithium-ion batteries and their impact on cycle life—An experimental study with different 18650 high-power cells," *J Energy Storage*, vol. 6, pp. 125–141, May 2016, doi: 10.1016/j.est.2016.02.005.

[150] M. Ecker *et al.*, "Calendar and cycle life study of Li(NiMnCo)O2-based 18650 lithium-ion batteries," *J Power Sources*, vol. 248, pp. 839–851, Feb. 2014, doi: 10.1016/j.jpowsour.2013.09.143.

[151] Z. Tian, L. Tu, C. Tian, Y. Wang, and F. Zhang, "Understanding Battery Degradation Phenomenon in Real-Life Electric Vehicle Use Based on Big Data," in *2017 3rd International Conference on Big Data Computing and Communications (BIGCOM)*, IEEE, Aug. 2017, pp. 334–339. doi: 10.1109/BIGCOM.2017.15.

[152] Tesla, "Tesla Model Y Owner Manual," https://www.tesla.com/sites/default/files/model_y_owners_manual_north_america_en.pdf.

[153] Tesla, "Tesla Model 3 Owner Manual," https://www.tesla.com/sites/default/files/model_3_owners_manual_north_america_en.pdf.

[154] Volkswagen, "Volkswagen ID.4 Owner Manual," https://vwid4.se/wp-content/uploads/2023/10/Manual-ID4-Owners-manual-10_2020.pdf.

[155] Hyundai, "Hyundai Ioniq 5 Owner Manual," https://www.hyundai.com/content/dam/hyundai/in/en/data/connect-to-service/owners-manual/new/ioniq5oct2022-present.pdf.

[156] Ford, "Ford Mustang Mach-E Owner Manual," https://www.ford.com/support/vehicle/mustang-mach-e/2023/.

[157] Chevrolet, "Chevrolet Bolt EV Owner Manual," https://experience.gm.com/ownercenter/content/dam/gmownercenter/gmna/dynamic/manuals/2017/Chevrolet/BOLT%20EV/Owner's%20Manual.pdf.

[158] Nissan, "Nissan Leaf Owner Manual," https://www.nissan.pt/clientes/manuais-do-propriet%C3%A1rio-da/manuais/iom/leaf/0ze1/e0/2023/carga-normal-1.shtml.

[159] BMW, "BMW i4 Owner Manual," https://bmwi.bimmerpost.com/forums/showthread.php?t=1868965.

[160] Audi, "Audi e-Tron Owner Manual," https://www.dropbox.com/scl/fi/8zj1qls4t2ial3sd9wgzt/2022-audi-e-tron-gt-25.pdf?rlkey=t9s9qixwka9rvexp84nj6tie6&e=1&dl=0.

[161] Porsche, "Porsche Taycan Owner Manual," https://www.taycanforum.com/forum/attachments/taycan_pdf_2021_en_compressed-pdf.12850/?force-system-browser=1.

Annex A

Overview

This annex contains the fuzzy inference model and model rules that were implemented to extract the $C_{f.fuzzy}$ values.

A1. Fuzzy Inference Model

The fuzzy logic inference model was implemented in python using the *skfuzzy* library.
Three inputs were utilised, the $C_{f.avg.}$ the demand level and the optimisation level, only the latter two are direct inputs to the fuzzy logic inference model, these are show in TABLE XXXIX.

TABLE XXXIX
Fuzzy Logic Inference Model Rules

Inputs: If Demand …	Inputs: & Optimisation …	Then Correction Factor …
Low	High	Low
Medium	Medium	Medium
High	Low	High
Low	Medium	Medium
Medium	Low	High
High	High	Medium
Low	Low	Medium
Medium	High	Medium
High	Medium	Medium

The membership functions of both inputs and outputs can be consulted in the following figures FIGURE 52, FIGURE 53, and FIGURE 54

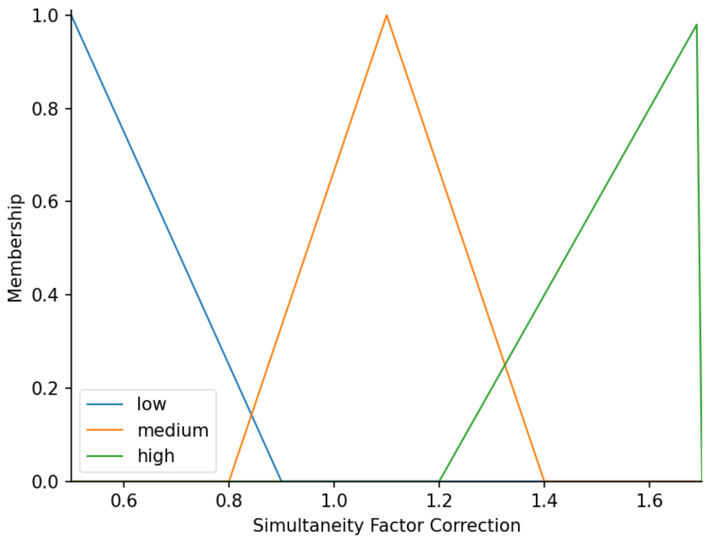


FIGURE 52. Coincidence Factor Correction membership function, output.

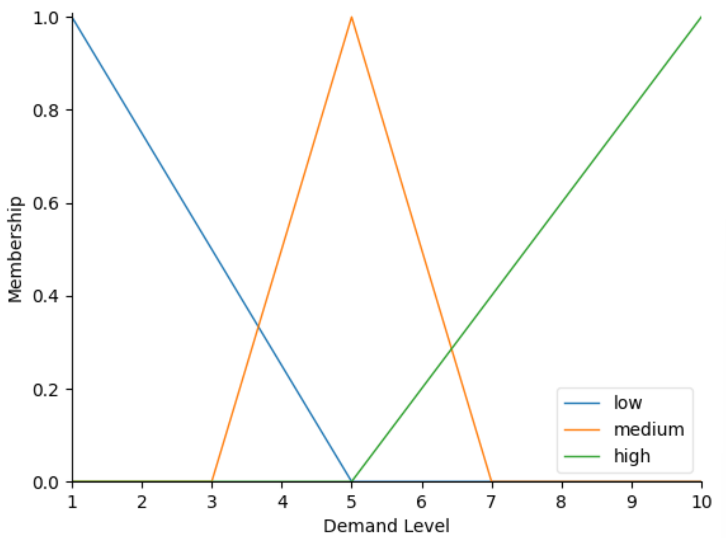


FIGURE 53. Demand Level membership function, input.

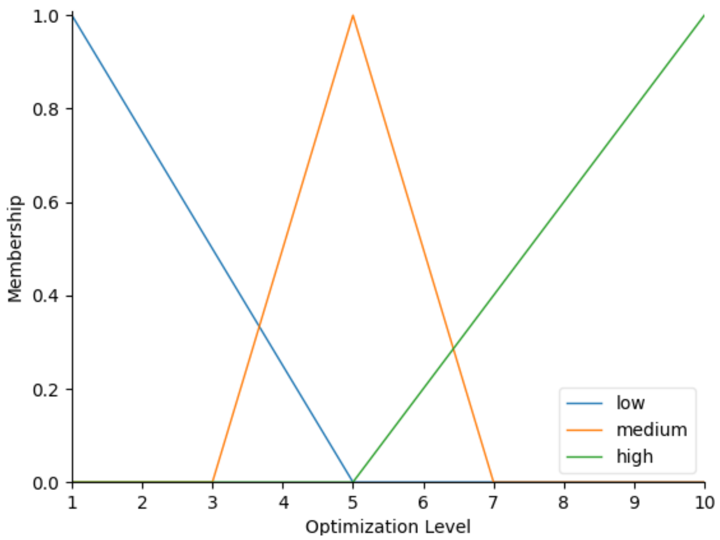


FIGURE 54. Optimisation Level membership function, input

Finally, $C_{f.fuzzy}$ is calculated according to equation (75).

$$C_{f.fuzzy} = C_{f.avg} \times F_{cf} \tag{75}$$

Where, $C_{f.avg}$ is the output of the average coincidence factor, and F_{cf} is the Fuzzy correction factor, that results from the fuzzy inference model output.

Annex B

Overview

This annex contains the core source code for the EV Fleet Charging Optimisation System developed as part of this doctoral research. The code implements three charging strategies: ad-hoc optimised charging, scheduled charging with demand forecasting, and baseline equal distribution for comparison purposes.

The implementation follows IEC 61851 standards for current bounds (6-16A single-phase, 16-32A three-phase) and incorporates battery degradation modelling based on Naumann et al. (2018). The thermal model uses an RC circuit analogue with convection cooling.

The complete codebase includes 48 Python files with approximately 23,800 lines of code, 354 unit tests, and 76% test coverage. This annex presents the four core modules essential for understanding the optimisation methodology.

B.1 Optimisation Module

This module provides the optimisation functions used across all charging strategies, including objective functions for different optimisation goals, constraint factories for grid and battery limits, and the SLSQP optimisation wrapper.

```
 """
Optimisation Utilities for EV Charging

This module provides shared optimisation functions used across all charging strategies:
- Objective functions for different optimisation goals
- Constraint factory functions for grid and battery limits
- Bounds generation for optimisation variables
- Fleet availability checking and rebalancing
- High-level optimisation wrapper functions

The optimisation uses scipy.optimise.minimise with SLSQP method, which is well-suited
for constrained nonlinear optimisation problems like EV charging allocation.

Key Components:
    - Objective Functions: Define what to minimise (time deviation, total time, etc.)
    - Constraints: Ensure grid capacity limits are respected
    - Bounds: Enforce min/max current per EV (IEC 61851 compliance)
    - Fleet Availability: Manage EVs when insufficient current is available

Original Source: functions.py (optimisation portions), Optimised_Charging_3_Phases_fast_soh.py

Author: Salvador Carvalhosa
"""

from typing import List, Dict, Tuple, Callable, Optional, Any, Union
from dataclasses import dataclass
import numpy as np
from scipy.optimise import minimise, OptimiseResult


# ============================================================================
# Constants
# ============================================================================

# IEC 61851 compliant current bounds
MIN_CURRENT_SINGLE_PHASE = 6.0   # Minimum current for single-phase charging (A)
MAX_CURRENT_SINGLE_PHASE = 16.0  # Maximum current for slow single-phase charging (A)
MIN_CURRENT_THREE_PHASE = 16.0   # Minimum current for three-phase charging (A)
MAX_CURRENT_THREE_PHASE = 32.0   # Maximum current for fast three-phase charging (A)
```

```
# Default safety factor for grid constraints
DEFAULT_SAFETY_FACTOR = 1.0

# Optimisation algorithm parameters
DEFAULT_MAX_ITERATIONS = 5000
DEFAULT_TOLERANCE = 1e-4


# ============================================================================
# Objective Functions
# ============================================================================

def objective_time_deviation(
    currents: np.ndarray,
    fleet: List[Dict],
    voltage: float
) -> float:
    """
    Objective function: Minimise deviation from desired charging time.

    This is the primary objective used in the ad-hoc optimisation strategy.
    It minimises the difference between the time required to charge each EV
    at the given current and the time remaining until their departure.

    Minimises: Σ |E_missing / (I_i × V) - T_remaining|

    Args:
        currents: Array of charging currents for each EV (A)
        fleet: List of EV dictionaries with keys:
            - 'Missing SOC': Energy needed to reach desired SoC (Wh)
            - 'Remaining Time': Hours until departure
        voltage: Grid voltage (V)

    Returns:
        Total objective value (sum of time deviations)

    Example:
        >>> fleet = [{'Missing SOC': 5000, 'Remaining Time': 3.0}]
        >>> currents = np.array([10.0])
        >>> obj = objective_time_deviation(currents, fleet, 230)
        >>> print(f"Objective value: {obj:.2f}")

    Notes:
        - Lower values indicate better matching between charging rate and available time
        - An objective of 0 would mean each EV finishes exactly at departure time
        - The function handles zero/negative currents gracefully
    """
    obj_value = 0.0

    for i, ev in enumerate(fleet):
        missing_soc = ev.get('Missing SOC', 0)
        remaining_time = ev.get('Remaining Time', 0)

        if missing_soc > 0 and currents[i] > 0:
            # Time required to charge at given current (hours)
            charging_time_required = missing_soc / (currents[i] * voltage)
            # Deviation from available time
            time_deviation = abs(charging_time_required - remaining_time)
            obj_value += time_deviation

    return obj_value


def objective_minimise_total_time(
    currents: np.ndarray,
    fleet: List[Dict],
    voltage: float
) -> float:
    """
    Objective function: Minimise total charging time across fleet.

    This objective prioritises finishing charging as quickly as possible,
    which may result in higher peak demand but shorter overall charging duration.
```

```
    Minimises: Σ E_missing / (I_i × V)

    Args:
        currents: Array of charging currents for each EV (A)
        fleet: List of EV dictionaries
        voltage: Grid voltage (V)

    Returns:
        Total charging time (hours)

    Example:
        >>> fleet = [{'Missing SOC': 5000}, {'Missing SOC': 3000}]
        >>> currents = np.array([16.0, 16.0])
        >>> time = objective_minimise_total_time(currents, fleet, 230)
    """
    total_time = 0.0

    for i, ev in enumerate(fleet):
        missing_soc = ev.get('Missing SOC', 0)

        if missing_soc > 0 and currents[i] > 0:
            charging_time = missing_soc / (currents[i] * voltage)
            total_time += charging_time

    return total_time


def objective_minimise_peak(
    currents: np.ndarray,
    fleet: List[Dict],
    voltage: float
) -> float:
    """
    Objective function: Minimise peak current (load leveling).

    This objective spreads charging load evenly to reduce peak demand,
    which is beneficial for grid stability and may reduce demand charges.

    Minimises: max(currents) - mean(currents) + small penalty for total current

    Args:
        currents: Array of charging currents for each EV (A)
        fleet: List of EV dictionaries (not directly used but kept for API consistency)
        voltage: Grid voltage (V)

    Returns:
        Peak minimisation objective value
    """
    if len(currents) == 0:
        return 0.0

    # Penalise deviation from mean (encourages uniform distribution)
    mean_current = np.mean(currents)
    variance_penalty = np.sum((currents - mean_current) ** 2)

    # Small penalty for total current to break ties
    total_penalty = 0.01 * np.sum(currents)

    return variance_penalty + total_penalty


def objective_battery_health(
    currents: np.ndarray,
    fleet: List[Dict],
    voltage: float
) -> float:
    """
    Objective function: Minimise battery degradation.

    This objective prioritises battery longevity by preferring lower
    charging currents, weighted by each EV's battery state.

    Higher currents cause more thermal stress and faster degradation.
    The objective is weighted by SoC (higher degradation at high SoC)
```

```
    and temperature.

    Args:
        currents: Array of charging currents for each EV (A)
        fleet: List of EV dictionaries with optional keys:
            - 'Current SOC': Current state of charge (Wh)
            - 'Maximum SOC': Maximum capacity (Wh)
            - 'Temperature': Battery temperature (°C)
        voltage: Grid voltage (V)

    Returns:
        Battery health degradation objective value
    """
    degradation_cost = 0.0

    for i, ev in enumerate(fleet):
        current = currents[i]

        # Get SoC ratio (0-1)
        current_soc = ev.get('Current SOC', 0)
        max_soc = ev.get('Maximum SOC', 1)
        soc_ratio = current_soc / max_soc if max_soc > 0 else 0

        # Get temperature factor (degradation increases with temperature)
        temp = ev.get('Temperature', 21)
        temp_factor = 1.0 + max(0, (temp - 25) / 20)  # Increased penalty above 25°C

        # Higher SoC = higher degradation from charging
        soc_factor = 1.0 + soc_ratio  # 1.0 at empty, 2.0 at full

        # Degradation proportional to I² (Joule heating)
        degradation_cost += (current ** 2) * soc_factor * temp_factor

    return degradation_cost


# ==============================================================================
# Constraint Functions
# ==============================================================================

def constraint_current_limit(
    currents: np.ndarray,
    available_current: float,
    safety_factor: float = DEFAULT_SAFETY_FACTOR
) -> float:
    """
    Constraint: Total current must not exceed available capacity.

    This is an inequality constraint for scipy.optimise. The constraint
    is satisfied when the return value is non-negative.

    Constraint: available_current × safety_factor - Σ currents ≥ 0

    Args:
        currents: Array of charging currents for each EV (A)
        available_current: Available grid current capacity (A)
        safety_factor: Safety coefficient (typically 1.0)

    Returns:
        Remaining capacity (positive if constraint satisfied, negative if violated)

    Example:
        >>> currents = np.array([10.0, 12.0, 8.0])
        >>> remaining = constraint_current_limit(currents, 50.0)
        >>> print(f"Remaining capacity: {remaining:.1f} A")  # 20.0 A
    """
    return available_current * safety_factor - np.sum(currents)


def create_current_constraint(
    available_current: float,
    safety_factor: float = DEFAULT_SAFETY_FACTOR
) -> Dict[str, Any]:
    """
```

```
    Factory function to create a scipy-compatible constraint dictionary.

    Creates an inequality constraint ensuring total allocated current
    does not exceed available grid capacity.

    Args:
        available_current: Available grid current (A)
        safety_factor: Safety coefficient

    Returns:
        Dictionary with keys 'type', 'fun', and 'args' for scipy.optimise

    Example:
        >>> constraint = create_current_constraint(100.0)
        >>> result = minimise(objective, x0, constraints=[constraint])
    """
    return {
        'type': 'ineq',
        'fun': constraint_current_limit,
        'args': (available_current, safety_factor)
    }


def create_lambda_constraint(
    available_current: float,
    safety_factor: float = DEFAULT_SAFETY_FACTOR
) -> Dict[str, Any]:
    """
    Create a constraint using lambda function (alternative implementation).

    This version uses a lambda closure, which can be more convenient
    when the constraint function needs to be modified dynamically.

    Args:
        available_current: Available grid current (A)
        safety_factor: Safety coefficient

    Returns:
        Dictionary with 'type' and 'fun' keys for scipy.optimise
    """
    return {
        'type': 'ineq',
        'fun': lambda x: available_current * safety_factor - np.sum(x)
    }


# ============================================================================
# Bounds Generation
# ============================================================================

def generate_bounds_single_phase(
    fleet: Union[List[Dict], int],
    lower: float = MIN_CURRENT_SINGLE_PHASE,
    upper: float = MAX_CURRENT_SINGLE_PHASE
) -> List[Tuple[float, float]]:
    """
    Generate current bounds for single-phase EVs.

    Creates bounds tuples for scipy.optimise. Each EV gets the same
    min/max current bounds based on IEC 61851 requirements.

    Args:
        fleet: Either a list of EV dictionaries or an integer fleet size
        lower: Minimum current (A) per IEC 61851 (default: 6A)
        upper: Maximum current (A) for slow charging (default: 16A)

    Returns:
        List of (min, max) tuples, one for each EV

    Example:
        >>> bounds = generate_bounds_single_phase(5)  # 5 EVs
        >>> print(bounds)
        [(6.0, 16.0), (6.0, 16.0), (6.0, 16.0), (6.0, 16.0), (6.0, 16.0)]
```

```
        >>> fleet = [{'ID': 1}, {'ID': 2}]
        >>> bounds = generate_bounds_single_phase(fleet)
        >>> print(len(bounds))
        2

    Notes:
        - 6A minimum is required by IEC 61851 for EVSE
        - 16A maximum is typical for household single-phase connections
    """
    n = len(fleet) if isinstance(fleet, list) else fleet
    return [(lower, upper)] * n


def generate_bounds_three_phase(
    fleet: Union[List[Dict], int],
    lower: float = MIN_CURRENT_THREE_PHASE,
    upper: float = MAX_CURRENT_THREE_PHASE
) -> List[Tuple[float, float]]:
    """
    Generate current bounds for three-phase EVs.

    Three-phase chargers typically operate at higher power levels,
    with correspondingly higher minimum and maximum currents.

    Args:
        fleet: Either a list of EV dictionaries or an integer fleet size
        lower: Minimum three-phase current (A) (default: 16A)
        upper: Maximum three-phase current (A) (default: 32A)

    Returns:
        List of (min, max) tuples, one for each EV

    Example:
        >>> bounds = generate_bounds_three_phase(3)
        >>> print(bounds)
        [(16.0, 32.0), (16.0, 32.0), (16.0, 32.0)]
    """
    n = len(fleet) if isinstance(fleet, list) else fleet
    return [(lower, upper)] * n


def generate_bounds_mixed(
    fleet: List[Dict],
    single_phase_range: Tuple[float, float] = (MIN_CURRENT_SINGLE_PHASE,
MAX_CURRENT_SINGLE_PHASE),
    three_phase_range: Tuple[float, float] = (MIN_CURRENT_THREE_PHASE, MAX_CURRENT_THREE_PHASE)
) -> List[Tuple[float, float]]:
    """
    Generate bounds for a mixed fleet (single-phase and three-phase EVs).

    Automatically determines bounds based on each EV's phase assignment.

    Args:
        fleet: List of EV dictionaries with 'Phase' key (1-3 for single, 4 for three-phase)
        single_phase_range: (min, max) for single-phase EVs
        three_phase_range: (min, max) for three-phase EVs

    Returns:
        List of (min, max) tuples, appropriate for each EV's phase type

    Example:
        >>> fleet = [{'Phase': 1}, {'Phase': 4}, {'Phase': 2}]
        >>> bounds = generate_bounds_mixed(fleet)
        >>> print(bounds)
        [(6.0, 16.0), (16.0, 32.0), (6.0, 16.0)]
    """
    bounds = []
    for ev in fleet:
        phase = ev.get('Phase', 1)
        if phase == 4:  # Three-phase
            bounds.append(three_phase_range)
        else:  # Single-phase (1, 2, or 3)
            bounds.append(single_phase_range)
    return bounds
```

```
# ==============================================================================
# Optimisation Functions
# ==============================================================================

@dataclass
class OptimisationResult:
    """
    Result container for charging current optimisation.

    Attributes:
        currents: Optimised current array for each EV (A)
        success: Whether optimisation converged successfully
        objective_value: Final objective function value
        message: Status message from optimiser
        iterations: Number of optimisation iterations
        total_current: Sum of all allocated currents (A)
    """
    currents: np.ndarray
    success: bool
    objective_value: float
    message: str
    iterations: int
    total_current: float


def optimise_charging_currents(
    fleet: List[Dict],
    voltage: float,
    available_current: float,
    objective: Callable = objective_time_deviation,
    safety_factor: float = DEFAULT_SAFETY_FACTOR,
    bounds: Optional[List[Tuple[float, float]]] = None,
    max_iterations: int = DEFAULT_MAX_ITERATIONS,
    tolerance: float = DEFAULT_TOLERANCE,
    initial_guess: Optional[np.ndarray] = None
) -> OptimisationResult:
    """
    Optimise charging currents using SLSQP algorithm.

    This is the main optimisation function that allocates charging current
    among EVs to minimise the specified objective while respecting grid
    capacity constraints and per-EV current limits.

    Args:
        fleet: List of EV dictionaries with charging state
        voltage: Grid voltage (V)
        available_current: Available grid current capacity (A)
        objective: Objective function to minimise (default: time deviation)
        safety_factor: Safety coefficient for grid constraint
        bounds: Optional current bounds per EV (generated if None)
        max_iterations: Maximum optimisation iterations
        tolerance: Convergence tolerance
        initial_guess: Optional starting point for optimisation

    Returns:
        OptimisationResult with optimised currents and metadata

    Example:
        >>> fleet = [
        ...     {'Missing SOC': 5000, 'Remaining Time': 3.0},
        ...     {'Missing SOC': 3000, 'Remaining Time': 2.0}
        ... ]
        >>> result = optimise_charging_currents(fleet, 230.0, 30.0)
        >>> print(f"Currents: {result.currents}")
        >>> print(f"Success: {result.success}")

    Notes:
        - Uses SLSQP (Sequential Least Squares Programming) method
        - Falls back to equal distribution if optimisation fails
        - Empty fleet returns zero-current result
    """
    # Handle empty fleet
```

```
    if len(fleet) == 0:
        return OptimisationResult(
            currents=np.array([]),
            success=True,
            objective_value=0.0,
            message="Empty fleet",
            iterations=0,
            total_current=0.0
        )

    # Generate bounds if not provided
    if bounds is None:
        bounds = generate_bounds_single_phase(fleet)

    # Create initial guess if not provided
    if initial_guess is None:
        # Start with equal distribution, clipped to bounds
        equal_share = available_current / len(fleet)
        initial_guess = np.array([
            np.clip(equal_share, b[0], b[1]) for b in bounds
        ])

    # Create constraint
    constraint = create_lambda_constraint(available_current, safety_factor)

    # Run optimisation
    try:
        result = minimise(
            objective,
            initial_guess,
            args=(fleet, voltage),
            method='SLSQP',
            bounds=bounds,
            constraints=[constraint],
            options={'maxiter': max_iterations, 'ftol': tolerance}
        )

        if result.success:
            currents = result.x
        else:
            # Fall back to initial guess (equal distribution)
            currents = initial_guess

        return OptimisationResult(
            currents=currents,
            success=result.success,
            objective_value=float(result.fun) if result.success else float('inf'),
            message=result.message if hasattr(result, 'message') else str(result.status),
            iterations=result.nit if hasattr(result, 'nit') else 0,
            total_current=float(np.sum(currents))
        )

    except Exception as e:
        # On exception, return equal distribution
        return OptimisationResult(
            currents=initial_guess,
            success=False,
            objective_value=float('inf'),
            message=f"Optimisation error: {str(e)}",
            iterations=0,
            total_current=float(np.sum(initial_guess))
        )


def quick_optimise(
    fleet: List[Dict],
    available_current: float,
    voltage: float = 230.0,
    phase_type: str = 'single'
) -> np.ndarray:
    """
    Quick optimisation with sensible defaults.

    Convenience function for simple optimisation scenarios.
```

```
    Args:
        fleet: List of EV dictionaries
        available_current: Available grid current (A)
        voltage: Grid voltage (V) (default: 230V)
        phase_type: 'single' or 'three' for bound selection

    Returns:
        Optimised current array

    Example:
        >>> fleet = [{'Missing SOC': 5000, 'Remaining Time': 3.0}]
        >>> currents = quick_optimise(fleet, 50.0)
    """
    if phase_type == 'three':
        bounds = generate_bounds_three_phase(fleet)
    else:
        bounds = generate_bounds_single_phase(fleet)

    result = optimise_charging_currents(
        fleet=fleet,
        voltage=voltage,
        available_current=available_current,
        bounds=bounds
    )

    return result.currents


# ==============================================================================
# Fleet Availability Checking
# ==============================================================================

def check_phase_availability(
    fleet: List[Dict],
    available_current: float,
    min_current_per_ev: float,
    safety_factor: float = DEFAULT_SAFETY_FACTOR
) -> Tuple[List[Dict], List[Dict]]:
    """
    Check if a phase has enough current for all EVs and rebalance if needed.

    If insufficient current is available to charge all EVs at minimum rate,
    removes EVs with lowest missing SoC until constraint is satisfied.

    Args:
        fleet: List of EV dictionaries on this phase
        available_current: Available current for this phase (A)
        min_current_per_ev: Minimum required current per EV (A)
        safety_factor: Safety coefficient

    Returns:
        Tuple of (remaining_fleet, removed_evs) where:
            - remaining_fleet: EVs that can be charged
            - removed_evs: EVs that were removed due to insufficient capacity

    Example:
        >>> fleet = [
        ...     {'ID': 1, 'Missing SOC': 5000},
        ...     {'ID': 2, 'Missing SOC': 2000},
        ...     {'ID': 3, 'Missing SOC': 8000}
        ... ]
        >>> remaining, removed = check_phase_availability(fleet, 15.0, 6.0)
        >>> print(f"Can charge {len(remaining)} EVs")  # 2 EVs (15/6 = 2.5)
    """
    fleet = list(fleet)  # Create a copy
    removed_evs = []
    effective_current = available_current * safety_factor

    while effective_current - len(fleet) * min_current_per_ev < 0 and len(fleet) > 0:
        # Find EV with lowest missing SOC
        missing_socs = [ev.get('Missing SOC', 0) for ev in fleet]
        min_idx = np.argmin(missing_socs)
```

```
        # Remove EV with lowest need
        removed_ev = fleet.pop(min_idx)
        removed_evs.append(removed_ev)

    return fleet, removed_evs


def check_fleet_availability(
    fleet_phase_1: List[Dict],
    fleet_phase_2: List[Dict],
    fleet_phase_3: List[Dict],
    fleet_tri_phase: List[Dict],
    available_amp_1: float,
    available_amp_2: float,
    available_amp_3: float,
    available_amp_4: float,
    current_hour: float,
    fleet_discharging: List[Dict],
    safety_factor: float = DEFAULT_SAFETY_FACTOR,
    update_plugin_time: bool = True,
    plugin_delay: float = 0.25
) -> Tuple[List[Dict], List[Dict], List[Dict], List[Dict], List[Dict]]:
    """
    Check if there's enough current for minimum charging rates across all phases.

    If insufficient current is available for any phase, moves EVs with lowest
    missing SoC back to discharging fleet with delayed plugin time.

    This implements the availability_check logic from the original code,
    ensuring grid constraints are always satisfied.

    Args:
        fleet_phase_1/2/3: Single-phase charging fleets
        fleet_tri_phase: Three-phase charging fleet
        available_amp_1/2/3/4: Available currents per phase (A)
        current_hour: Current simulation time (decimal hours)
        fleet_discharging: Fleet of EVs not currently charging
        safety_factor: Safety coefficient
        update_plugin_time: Whether to delay plugin time for removed EVs
        plugin_delay: Hours to delay plugin time (default: 0.25 = 15 minutes)

    Returns:
        Tuple of (fleet_1, fleet_2, fleet_3, fleet_tri, fleet_discharging)
        All returned as lists (converted from any input type)

    Example:
        >>> # Check availability and rebalance fleets
        >>> f1, f2, f3, ft, fd = check_fleet_availability(
        ...     fleet_1, fleet_2, fleet_3, fleet_tri,
        ...     50.0, 50.0, 50.0, 100.0,
        ...     21.0, fleet_discharging
        ... )

    Notes:
        - Single-phase EVs require minimum 6A
        - Three-phase EVs require minimum 16A
        - Removed EVs get updated plugin time if before their plugout time
    """
    from utils.time_utils import convert_decimal_to_time

    # Convert to lists
    fleet_phase_1 = list(fleet_phase_1)
    fleet_phase_2 = list(fleet_phase_2)
    fleet_phase_3 = list(fleet_phase_3)
    fleet_tri_phase = list(fleet_tri_phase)
    fleet_discharging = list(fleet_discharging)

    # Helper to process a single phase
    def process_phase(fleet, available, min_current):
        remaining, removed = check_phase_availability(
            fleet, available, min_current, safety_factor
        )

        for ev in removed:
```

```
            # Update plugin time if before plugout
            if update_plugin_time:
                new_plugin = convert_decimal_to_time(current_hour + plugin_delay)
                if new_plugin < ev.get('Plugout Time', '23:59'):
                    ev['Plugin Time'] = new_plugin

            fleet_discharging.append(ev)

        return remaining

    # Process each phase
    fleet_phase_1 = process_phase(fleet_phase_1, available_amp_1, MIN_CURRENT_SINGLE_PHASE)
    fleet_phase_2 = process_phase(fleet_phase_2, available_amp_2, MIN_CURRENT_SINGLE_PHASE)
    fleet_phase_3 = process_phase(fleet_phase_3, available_amp_3, MIN_CURRENT_SINGLE_PHASE)
    fleet_tri_phase = process_phase(fleet_tri_phase, available_amp_4, MIN_CURRENT_THREE_PHASE)

    return fleet_phase_1, fleet_phase_2, fleet_phase_3, fleet_tri_phase, fleet_discharging


# ============================================================================
# Utility Functions
# ============================================================================

def compute_charging_time(
    missing_soc: float,
    current: float,
    voltage: float
) -> float:
    """
    Calculate time required to charge a given energy amount.

    Args:
        missing_soc: Energy needed (Wh)
        current: Charging current (A)
        voltage: Charging voltage (V)

    Returns:
        Charging time (hours)
    """
    if current <= 0 or voltage <= 0:
        return float('inf')
    return missing_soc / (current * voltage)


def compute_required_current(
    missing_soc: float,
    remaining_time: float,
    voltage: float
) -> float:
    """
    Calculate current required to complete charging in given time.

    Args:
        missing_soc: Energy needed (Wh)
        remaining_time: Available time (hours)
        voltage: Charging voltage (V)

    Returns:
        Required current (A)
    """
    if remaining_time <= 0 or voltage <= 0:
        return float('inf')
    return missing_soc / (remaining_time * voltage)


def validate_currents(
    currents: np.ndarray,
    bounds: List[Tuple[float, float]],
    available_current: float,
    safety_factor: float = DEFAULT_SAFETY_FACTOR
) -> Tuple[bool, List[str]]:
    """
    Validate that currents satisfy all constraints.
```

```
    Args:
        currents: Array of allocated currents
        bounds: Per-EV bounds
        available_current: Grid capacity
        safety_factor: Safety coefficient

    Returns:
        Tuple of (is_valid, list_of_violations)
    """
    violations = []

    # Check total current
    total = np.sum(currents)
    limit = available_current * safety_factor
    if total > limit + 1e-6:  # Small tolerance for floating point
        violations.append(f"Total current ({total:.2f}A) exceeds limit ({limit:.2f}A)")

    # Check individual bounds
    for i, (current, (lower, upper)) in enumerate(zip(currents, bounds)):
        if current < lower - 1e-6:
            violations.append(f"EV {i}: Current ({current:.2f}A) below minimum ({lower:.2f}A)")
        if current > upper + 1e-6:
            violations.append(f"EV {i}: Current ({current:.2f}A) above maximum ({upper:.2f}A)")

    return len(violations) == 0, violations


def equal_distribution(
    n_evs: int,
    available_current: float,
    bounds: Optional[List[Tuple[float, float]]] = None
) -> np.ndarray:
    """
    Distribute current equally among EVs (baseline algorithm).

    This is the non-optimised distribution used in BaselineChargingSimulation.

    Args:
        n_evs: Number of EVs
        available_current: Total available current (A)
        bounds: Optional per-EV bounds

    Returns:
        Array of currents for each EV
    """
    if n_evs == 0:
        return np.array([])

    if bounds is None:
        bounds = generate_bounds_single_phase(n_evs)

    # Equal share
    share = available_current / n_evs

    # Clip to bounds
    currents = np.array([
        np.clip(share, lower, upper) for lower, upper in bounds
    ])

    # Scale down if total exceeds available
    total = np.sum(currents)
    if total > available_current and total > 0:
        currents = currents * (available_current / total)

    return currents
```

B.2 Battery Model

This module implements battery modelling, including State of Health (SoH) degradation based on the Naumann et al. (2018) model, thermal behaviour using an RC circuit analogue, and SoC-dependent internal resistance.

```
"""
Battery Model - State of Health and Thermal Behaviour

This module implements battery modelling including:
1. State of Health (SoH) degradation based on cycle statistics
2. Battery temperature evolution using RC thermal circuit analogue
3. SoC-dependent internal resistance
4. Cell configuration (series/parallel) optimisation

Key Models:
- Cycle-based degradation (Naumann et al., 2018)
- RC thermal circuit with convection cooling
- Internal resistance as function of SoC

Original Source: functions.py (battery portions)

Author: Salvador Carvalhosa
"""

from typing import Tuple, List, Dict, Optional, Union
from dataclasses import dataclass
from datetime import datetime
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt


@dataclass
class BatteryParameters:
    """Physical parameters for battery modelling."""

    # Capacity
    nominal_capacity_kwh: float = 60.0  # kWh
    nominal_capacity_wh: float = 60000.0  # Wh

    # SoC limits
    soc_min: float = 0.2
    soc_max: float = 0.8

    # Thermal parameters
    mass: float = 410.0  # kg (approximately 6-7 kg/kWh)
    specific_heat: float = 1000.0  # J/(kg·K)
    surface_area: float = 5.0  # m²
    convection_coefficient: float = 20.0  # W/(m²·K)
    ambient_temperature: float = 21.0  # °C

    # Degradation model constants (Naumann et al., 2018)
    k_s1: float = -4.092e-4
    k_s2: float = -2.167
    k_s3: float = 1.408e-3
    k_s4: float = 6.130
    activation_energy: float = 25087.0  # J/mol
    gas_constant: float = 8.314  # J/(mol·K)
    reference_temperature_celsius: float = 21.0  # °C
    reference_temperature_kelvin: float = 294.15  # K (21°C)


class BatteryModel:
    """
    Comprehensive battery model with degradation and thermal behaviour.

    Attributes:
```

```
        params: Battery physical parameters
        current_soc: Current state of charge (0-1)
        current_temperature: Current temperature (°C)
        soh: State of health (0-1, 1=new, 0=end-of-life)
        accumulated_degradation: Cumulative degradation (ξ_cyc_total)
        cells_in_series: Number of cells connected in series
        cells_in_parallel: Number of cells connected in parallel

    Example:
        >>> battery = BatteryModel(capacity_kwh=60.0)
        >>> battery.charge(current=10.0, voltage=230.0, duration_hours=0.05)
        >>> print(f"SoC: {battery.current_soc:.2%}")
        >>> print(f"Temperature: {battery.current_temperature:.1f}°C")
    """

    def __init__(
        self,
        capacity_kwh: float = 60.0,
        initial_soc: float = 0.6,
        initial_temperature: float = 21.0,
        params: Optional[BatteryParameters] = None
    ):
        """
        Initialize battery model.

        Args:
            capacity_kwh: Battery capacity in kWh
            initial_soc: Initial state of charge (0-1)
            initial_temperature: Initial temperature (°C)
            params: Optional battery parameters
        """
        self.params = params or BatteryParameters()
        self.params.nominal_capacity_kwh = capacity_kwh
        self.params.nominal_capacity_wh = capacity_kwh * 1000

        self.current_soc = initial_soc
        self.current_temperature = initial_temperature
        self.soh = 1.0
        self.accumulated_degradation = 0.0

        # Cell configuration
        self.cells_in_series, self.cells_in_parallel = self._calculate_cell_configuration()

    def _calculate_cell_configuration(self) -> Tuple[int, int]:
        """
        Calculate optimal series/parallel cell configuration.

        Uses golden ratio (1.618) as target for series/parallel ratio.
        Assumes 10 Wh per cell (nominal voltage ~3.7V, ~2.7Ah capacity).

        Returns:
            Tuple of (cells_in_series, cells_in_parallel)
        """
        # Calculate total number of cells (10 Wh per cell)
        total_cells = int(self.params.nominal_capacity_wh / 10)

        # Use golden ratio as desired ratio for series/parallel
        return find_best_series_parallel_combination(total_cells, 1.618)

    def get_internal_resistance(self) -> float:
        """
        Get battery pack internal resistance at current SoC.

        Based on empirical data from research literature:

https://www.researchgate.net/publication/347699330_Analysis_of_battery_management_system_issues_in_electric_
vehicles
        https://www.batterydesign.net/electrical/pack-internal-resistance/

        Returns:
            Internal resistance in Ohms
        """
        return get_internal_resistance(
            self.current_soc,
            self.cells_in_series,
```

```
        self.cells_in_parallel
    )

def charge(
    self,
    current: float,
    voltage: float,
    duration_hours: float
) -> Dict[str, float]:
    """
    Charge battery and update all states (SoC, temperature, SoH).

    Args:
        current: Charging current (A)
        voltage: Charging voltage (V)
        duration_hours: Charging duration (hours)

    Returns:
        Dictionary with updated states:
            - delta_soc: Change in SoC
            - delta_temperature: Change in temperature (°C)
            - delta_soh: Change in SoH
            - energy_charged: Energy added (Wh)
    """
    # Store initial values
    initial_soc = self.current_soc
    initial_temperature = self.current_temperature
    initial_soh = self.soh

    # Energy charged
    energy_wh = current * voltage * duration_hours

    # Update SoC
    delta_soc = energy_wh / self.params.nominal_capacity_wh
    self.current_soc += delta_soc

    # Clip SoC to limits
    self.current_soc = np.clip(
        self.current_soc,
        self.params.soc_min,
        self.params.soc_max
    )

    # Update temperature (convert hours to seconds)
    duration_seconds = duration_hours * 3600
    self.current_temperature = self.update_temperature(
        current,
        duration_seconds
    )

    # Compute degradation (simplified single-event)
    if current > 0:
        ev_dict = {
            'Current SOC': self.current_soc * self.params.nominal_capacity_wh,
            'Maximum SOC': self.params.nominal_capacity_wh,
            'CiS': self.cells_in_series,
            'CiP': self.cells_in_parallel
        }
        # Note: Full degradation calculation requires event-based processing
        # This is a simplified approximation


    return {
        'delta_soc': self.current_soc - initial_soc,
        'delta_temperature': self.current_temperature - initial_temperature,
        'delta_soh': self.soh - initial_soh,
        'energy_charged': energy_wh
    }

def discharge(self, energy_wh: float) -> Dict[str, float]:
    """
    Discharge battery by given energy.

    Args:
        energy_wh: Energy to discharge (Wh)
```

```
        Returns:
            Dictionary with:
                - delta_soc: Change in SoC (negative)
                - final_soc: Final SoC after discharge
        """
        initial_soc = self.current_soc

        # Calculate SoC change
        delta_soc = -energy_wh / self.params.nominal_capacity_wh
        self.current_soc += delta_soc

        # Ensure SoC doesn't go negative
        self.current_soc = max(0.0, self.current_soc)

        return {
            'delta_soc': self.current_soc - initial_soc,
            'final_soc': self.current_soc
        }

    def update_temperature(
        self,
        current: float,
        duration_seconds: float
    ) -> float:
        """
        Update battery temperature using RC thermal model.

        Uses RC circuit analogue:
        - Heat generation: Q_gen = I²R (Joule heating)
        - Heat dissipation: Q_diss = h × A × (T_battery - T_ambient)
        - Temperature change: ΔT = (Q_gen - Q_diss) / (m × c)

        Args:
            current: Charging/discharging current (A)
            duration_seconds: Time step duration (s)

        Returns:
            New temperature (°C)
        """
        # Get internal resistance at current SoC
        internal_resistance = self.get_internal_resistance()

        # Heat generated (Joule heating)
        heat_generated = (current ** 2) * internal_resistance * duration_seconds  # Joules

        # Heat dissipated (convection to ambient)
        heat_dissipated = (
            self.params.convection_coefficient *
            self.params.surface_area *
            (self.current_temperature - self.params.ambient_temperature) *
            duration_seconds
        )  # Joules

        # Net heat
        net_heat = heat_generated - heat_dissipated

        # Temperature change
        delta_temperature = net_heat / (self.params.mass * self.params.specific_heat)

        # Update temperature
        new_temperature = self.current_temperature + delta_temperature

        return new_temperature

    def compute_degradation(
        self,
        soc_low: float,
        soc_high: float,
        current: float,
        duration_hours: float
    ) -> float:
        """
        Compute capacity fade from a single charging event.
```

```
        Based on Naumann et al. (2018) empirical model:
        ξ_cyc = (k_s1 × σ_SoC × exp(k_s2 × SoC_avg) + k_s3 × exp(k_s4 × σ_SoC))
                × exp(-E_a/R × (1/T - 1/T_ref)) × ΔQ_proc

        Args:
            soc_low: Starting SoC (0-1)
            soc_high: Ending SoC (0-1)
            current: Charging current (A)
            duration_hours: Duration (hours)

        Returns:
            Degradation increment (ξ_cyc)
        """
        # Calculate processed charge (Ah)
        delta_q_proc = current * duration_hours

        # Average SoC
        soc_avg = (soc_low + soc_high) / 2

        # SoC deviation (simplified for single event)
        soc_dev = abs(soc_high - soc_low) / 2

        # Temperature factor
        temp_kelvin = self.current_temperature + 273.15
        temp_ref_kelvin = self.params.reference_temperature_celsius + 273.15
        temperature_factor = np.exp(
            -(self.params.activation_energy / self.params.gas_constant) *
            ((1 / temp_kelvin) - (1 / temp_ref_kelvin))
        )

        # Degradation calculation
        term1 = self.params.k_s1 * soc_dev * np.exp(self.params.k_s2 * soc_avg)
        term2 = self.params.k_s3 * np.exp(self.params.k_s4 * soc_dev)

        xi_cyc = (term1 + term2) * temperature_factor * delta_q_proc

        # Update accumulated degradation and SoH
        self.accumulated_degradation += xi_cyc
        self.soh = 1 - self.accumulated_degradation / self.params.nominal_capacity_kwh

        return xi_cyc


def compute_capacity_fade(
    time: np.ndarray,
    soc: np.ndarray,
    temperature: np.ndarray,
    current: np.ndarray,
    q_nom: float,
    plot_results: bool = False,
    save_csv: bool = False
) -> Tuple[float, float]:
    """
    Compute total capacity fade over entire charging history.

    Implements Naumann et al. (2018) cycle-based degradation model:

    ξ_cyc = (k_s1 × σ_SoC × exp(k_s2 × SoC_avg) + k_s3 × exp(k_s4 × σ_SoC))
            × exp(-E_a/R × (1/T - 1/T_ref)) × ΔQ_proc

    The function divides the charging history into 3-minute events and computes
    degradation for each event based on:
    - Average SoC during the event
    - SoC deviation (standard deviation weighted by charge processed)
    - Average temperature
    - Total charge processed (Ah)

    Args:
        time: Time array (datetime objects or seconds)
        soc: State of charge array (0-1)
        temperature: Temperature array (°C)
        current: Current array (A)
        q_nom: Nominal capacity (kWh)
        plot_results: Whether to plot per-event degradation (default: False)
        save_csv: Whether to save results to CSV (default: False)
```

```
    Returns:
        Tuple of (xi_cyc_total, final_soh):
            - xi_cyc_total: Total accumulated degradation
            - final_soh: Final state of health (0-1)

    Example:
        >>> time = np.arange(0, 10000, 180)  # 3-minute steps in seconds
        >>> soc = np.linspace(0.3, 0.8, len(time))
        >>> temp = np.full(len(time), 25.0)
        >>> current = np.full(len(time), 10.0)
        >>> xi_total, soh = compute_capacity_fade(time, soc, temp, current, 60.0)
        >>> print(f"Total degradation: {xi_total:.6f}, Final SoH: {soh:.4f}")

    Notes:
        - Event duration is fixed at 3 minutes (180 seconds)
        - Temperature must be in Celsius (converted to Kelvin internally)
        - Based on empirical aging model from:
          Naumann et al. (2018) "Analysis and modelling of calendar aging of a
          commercial LiFePO4/graphite cell"
    """
    # Convert time to seconds if in datetime format
    if isinstance(time[0], datetime):
        start_time = time[0]
        time = np.array([(t - start_time).total_seconds() for t in time])
    else:
        time = np.array(time)

    # Constants and parameters (Naumann et al., 2018)
    r = 8.314       # Gas constant in J/(mol·K)
    e_a = 25087     # Activation energy in J/mol
    t_ref = 21      # Reference temperature in °C (294.15 K)

    # Degradation model constants
    ks1 = -4.092e-4
    ks2 = -2.167
    ks3 = 1.408e-3
    ks4 = 6.130

    # Ensure all inputs are numpy arrays
    soc = np.array(soc)
    temperature = np.array(temperature)
    current = np.array(current)

    # Compute time differences
    dt = np.diff(time, prepend=time[0])

    # Ensure all arrays have the same length
    min_length = min(len(time), len(soc), len(temperature), len(current))
    time = time[:min_length]
    soc = soc[:min_length]
    temperature = temperature[:min_length]
    current = current[:min_length]
    dt = dt[:min_length]

    # Charge processed at each time step (Ah)
    delta_q_proc_k = np.abs(current) * dt / 3600  # Convert seconds to hours

    # Event-based processing (3-minute events)
    event_duration = 3 * 60  # 3 minutes in seconds
    total_time = time[-1] - time[0]
    num_events = int(np.ceil(total_time / event_duration))

    # Initialize accumulators
    xi_cyc_total = 0.0
    xi_cyc_list = []
    event_times = []
    avg_current_list = []
    avg_soc_list = []
    avg_temp_list = []

    # Process each event
    for i in range(num_events):
        t_start = time[0] + i * event_duration
        t_end = t_start + event_duration
```

```
    indices = np.where((time >= t_start) & (time < t_end))[0]

    if len(indices) == 0:
        continue

    # Extract event data
    soc_event = soc[indices]
    temp_event = temperature[indices]
    curr_event = current[indices]
    delta_q_proc_event = delta_q_proc_k[indices]
    delta_q_proc_i = np.sum(delta_q_proc_event)

    # Skip if no charge processed
    if delta_q_proc_i == 0:
        continue

    # Calculate weighted average SoC and SoC deviation
    soc_avg_i = np.sum(soc_event * delta_q_proc_event) / delta_q_proc_i
    soc_dev_i = np.sqrt(
        np.sum(((soc_event - soc_avg_i)**2) * delta_q_proc_event) / delta_q_proc_i
    )

    # Average temperature for the event
    t_i = np.mean(temp_event) + 273.15  # Convert to Kelvin

    # Calculate degradation for this event
    term1 = ks1 * soc_dev_i * np.exp(ks2 * soc_avg_i)
    term2 = ks3 * np.exp(ks4 * soc_dev_i)
    temperature_factor = np.exp(-(e_a / r) * ((1 / t_i) - (1 / (t_ref + 273.15))))

    xi_cyc_i = (term1 + term2) * temperature_factor * delta_q_proc_i
    xi_cyc_total += xi_cyc_i

    # Store event data for plotting/analysis
    xi_cyc_list.append(xi_cyc_i)
    event_times.append(t_start + event_duration / 2)
    avg_current_list.append(np.mean(curr_event))
    avg_soc_list.append(soc_avg_i)
    avg_temp_list.append(np.mean(temp_event))  # Store in Celsius

# Compute final State of Health
soh = 1 - xi_cyc_total / q_nom

# Optional: Plot results
if plot_results and len(xi_cyc_list) > 0:
    fig, axs = plt.subplots(4, 1, figsize=(10, 10), sharex=True)

    axs[0].plot(event_times, xi_cyc_list, marker='o', color='tab:red', label="ξ_cyc,i")
    axs[0].set_ylabel("ξ_cyc,i")
    axs[0].set_title("Per-Event Degradation and Conditions")
    axs[0].grid(True)
    axs[0].legend()

    axs[1].plot(event_times, avg_current_list, marker='x', color='tab:blue', label="Avg Current (A)")
    axs[1].set_ylabel("Current (A)")
    axs[1].grid(True)
    axs[1].legend()

    axs[2].plot(event_times, avg_soc_list, marker='s', color='tab:green', label="Avg SoC")
    axs[2].set_ylabel("SoC")
    axs[2].grid(True)
    axs[2].legend()

    axs[3].plot(event_times, avg_temp_list, marker='^', color='tab:orange', label="Avg Temp (°C)")
    axs[3].set_ylabel("Temp (°C)")
    axs[3].set_xlabel("Time (s)")
    axs[3].grid(True)
    axs[3].legend()

    plt.tight_layout()
    plt.show()

# Optional: Save to CSV
if save_csv and len(xi_cyc_list) > 0:
    df = pd.DataFrame({
```

```
            "Event_Times": event_times,
            "event_id": range(1, len(xi_cyc_list) + 1),
            "xi_cyc": xi_cyc_list,
            "avg_current": avg_current_list,
            "avg_soc": avg_soc_list,
            "avg_temp": avg_temp_list
        })
        df.to_csv("capacity_fade_events.csv", index=False, float_format="%.10g")

    return xi_cyc_total, soh


def calculate_battery_temperature(
    charging_current: float,
    ev: Dict,
    battery_temperature: float,
    air_temperature: float,
    time_step_minutes: float
) -> float:
    """
    Calculate battery temperature after time step using RC thermal circuit model.

    Uses RC thermal circuit analogue:
    1. Heat generation from Joule heating: Q_gen = I²R
    2. Heat dissipation via convection: Q_diss = h × A × (T_battery - T_ambient)
    3. Net temperature change: ΔT = (Q_gen - Q_diss) / (m × c)

    Args:
        charging_current: Charging current in amperes (A)
        ev: EV dictionary with battery parameters:
            - 'Current SOC': Current state of charge (Wh)
            - 'Maximum SOC': Maximum capacity (Wh)
            - 'CiS': Cells in series
            - 'CiP': Cells in parallel
        battery_temperature: Current battery temperature (°C)
        air_temperature: Ambient air temperature (°C) - currently unused, fixed at 21°C
        time_step_minutes: Duration of time step (minutes)

    Returns:
        Updated battery temperature (°C)

    Example:
        >>> ev = {
        ...     'Maximum SOC': 60000,  # 60 kWh in Wh
        ...     'Current SOC': 36000,  # 60% SoC
        ...     'CiS': 96,  # Series cells
        ...     'CiP': 60   # Parallel cells
        ... }
        >>> new_temp = calculate_battery_temperature(
        ...     charging_current=10.0,
        ...     ev=ev,
        ...     battery_temperature=25.0,
        ...     air_temperature=21.0,
        ...     time_step_minutes=3
        ... )

    Notes:
        - Internal resistance varies with SoC based on empirical data
        - Ambient temperature is fixed at 21°C (parameter air_temperature not used)
        - Battery mass is fixed at 410 kg
        - Convection coefficient h = 20 W/(m²·K)
        - Surface area A = 5 m²
    """
    # Physical constants
    c = 1000  # Specific heat capacity of lithium-ion batteries (J/(kg·K))
    time_step_seconds = time_step_minutes * 60  # Convert to seconds
    battery_mass = 410  # kg (fixed, approximately 6-7 kg/kWh for a 60 kWh battery)

    # Internal resistance data from literature
    # Source: https://www.researchgate.net/publication/347699330
    # Source: https://www.batterydesign.net/electrical/pack-internal-resistance/
    soc_data = np.array([0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0])
    ir_data = np.array([0.0058, 0.0038, 0.0029, 0.0025, 0.0022, 0.0020,
                        0.0019, 0.0018, 0.0018, 0.0019, 0.0020])
```

```
    # Calculate current SoC (0-1)
    soc = ev["Current SOC"] / ev["Maximum SOC"]

    # Interpolate cell internal resistance at current SoC
    internal_resistance_cell = np.interp(soc, soc_data, ir_data)

    # Calculate pack internal resistance
    # R_pack = R_cell × (cells_in_series / cells_in_parallel)
    internal_resistance_pack = internal_resistance_cell * ev["CiS"] / ev["CiP"]

    # Heat generated due to Joule heating: Q_gen = I² × R × t
    heat_generated = (charging_current ** 2) * internal_resistance_pack * time_step_seconds  # Joules

    # Heat dissipation parameters
    h = 20.0  # Heat transfer coefficient (W/(m²·K)) - natural convection
    area = 5.0  # Surface area of the battery pack (m²)

    # Heat dissipated due to convection: Q_diss = h × A × (T_battery - T_ambient) × t
    # Note: Using fixed ambient temperature of 21°C
    heat_dissipated = h * area * (battery_temperature - 21.0) * time_step_seconds  # Joules

    # Net heat: Q_net = Q_generated - Q_dissipated
    net_heat = heat_generated - heat_dissipated

    # Calculate temperature increase: ΔT = Q_net / (m × c)
    delta_temperature = net_heat / (battery_mass * c)

    # Update battery temperature
    updated_battery_temperature = battery_temperature + delta_temperature

    return updated_battery_temperature


def get_internal_resistance(
    soc: float,
    cells_in_series: int,
    cells_in_parallel: int
) -> float:
    """
    Get battery pack internal resistance based on SoC and cell configuration.

    Uses interpolation from empirical data on cell internal resistance as a
    function of state of charge. Pack resistance is calculated as:
    R_pack = R_cell × (N_series / N_parallel)

    Args:
        soc: State of charge (0-1)
        cells_in_series: Number of cells connected in series
        cells_in_parallel: Number of cells connected in parallel

    Returns:
        Pack internal resistance in Ohms

    Example:
        >>> resistance = get_internal_resistance(
        ...     soc=0.6,
        ...     cells_in_series=96,
        ...     cells_in_parallel=60
        ... )
        >>> print(f"Pack resistance: {resistance:.6f} Ohms")

    Notes:
        - Based on empirical data from research literature
        - IR data source: https://www.researchgate.net/publication/347699330
        - Cell IR ranges from ~0.0018 Ω (at 70% SoC) to 0.0058 Ω (at 0% SoC)
    """
    # Empirical data from literature
    # State of charge points (0 to 100%)
    soc_data = np.array([0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0])

    # Internal resistance data (Ohms) - IR_1 dataset used for more conservative estimates
    ir_data = np.array([0.0058, 0.0038, 0.0029, 0.0025, 0.0022, 0.0020,
                        0.0019, 0.0018, 0.0018, 0.0019, 0.0020])

    # Interpolate cell resistance at given SoC
```

```
    cell_resistance = np.interp(soc, soc_data, ir_data)

    # Calculate pack resistance
    # Series cells increase resistance, parallel cells decrease resistance
    pack_resistance = cell_resistance * cells_in_series / cells_in_parallel

    return pack_resistance


def find_best_series_parallel_combination(
    total_cells: int,
    desired_ratio: float
) -> Tuple[int, int]:
    """
    Find optimal series/parallel cell configuration for a battery pack.

    Searches for the combination of series and parallel cells that:
    1. Uses exactly the total number of cells
    2. Minimises the difference from the desired series/parallel ratio

    The golden ratio (1.618) is commonly used as it provides a good balance
    between voltage and capacity.

    Args:
        total_cells: Total number of cells in the battery pack
        desired_ratio: Target ratio of series to parallel cells (typically 1.618)

    Returns:
        Tuple of (cells_in_series, cells_in_parallel)

    Example:
        >>> series, parallel = find_best_series_parallel_combination(
        ...     total_cells=6000,
        ...     desired_ratio=1.618  # Golden ratio
        ... )
        >>> print(f"Configuration: {series}S{parallel}P")
        >>> print(f"Total cells: {series * parallel}")
        >>> print(f"Ratio: {series/parallel:.3f}")

    Notes:
        - Only considers configurations where total_cells is evenly divisible
        - For a 60 kWh battery with ~10 Wh/cell: total_cells = 6000
        - Typical result: ~96S × 62P (ratio ≈ 1.548)
    """
    best_series = 1
    best_parallel = total_cells
    best_difference = float('inf')

    # Try all possible series configurations
    for series in range(1, total_cells + 1):
        # Check if this creates an even division
        if total_cells % series == 0:
            parallel = total_cells // series
            current_ratio = series / parallel
            difference = abs(current_ratio - desired_ratio)

            # Update if this is closer to the desired ratio
            if difference < best_difference:
                best_difference = difference
                best_series = series
                best_parallel = parallel

    return best_series, best_parallel


# Helper functions for degradation calculations

def calculate_soc_avg(
    q_proc: float,
    soc_low: float,
    soc_high: float
) -> float:
    """
    Calculate weighted average SoC during a charging event.
```

```
    Uses trapezoidal integration to compute the average SoC over the
    charging period, weighted by the charge processed.

    Args:
        q_proc: Charge processed (Ah)
        soc_low: Starting SoC (0-1)
        soc_high: Ending SoC (0-1)

    Returns:
        Weighted average SoC (0-1)
    """
    x = np.array([0.0, float(q_proc)])
    y = np.array([float(soc_low), float(soc_high)])
    integral = float(np.trapz(y, x=x))
    soc_avg = integral / float(q_proc)
    return soc_avg


def calculate_soc_dev(
    q_proc: float,
    soc_low: float,
    soc_high: float,
    soc_avg: float
) -> float:
    """
    Calculate SoC deviation (standard deviation) during a charging event.

    Computes the weighted standard deviation of SoC during charging,
    where the weighting is proportional to charge processed.

    Args:
        q_proc: Charge processed (Ah)
        soc_low: Starting SoC (0-1)
        soc_high: Ending SoC (0-1)
        soc_avg: Average SoC (0-1)

    Returns:
        SoC standard deviation (0-1)
    """
    # Validate scalar inputs
    if not np.isscalar(q_proc) or not np.isscalar(soc_low) or not np.isscalar(soc_high) or not np.isscalar(soc_avg):
        raise ValueError("All inputs must be scalar values")

    # Calculate integrand at start and end points
    integrand = [(float(soc_low) - float(soc_avg)) ** 2, (float(soc_high) - float(soc_avg)) ** 2]  # type: ignore[arg-type]

    # Trapezoidal integration
    integral = float(np.trapz(integrand, x=[0.0, float(q_proc)]))  # type: ignore[arg-type]

    # Standard deviation
    soc_dev = float(np.sqrt(integral / float(q_proc)))  # type: ignore[arg-type]

    return soc_dev
```

B.3 Grid Model

This module handles the three-phase electrical grid representation and current allocation logic. It implements dynamic allocation between single-phase and three-phase charging based on fleet composition and energy needs.

```
"""
Three-Phase Grid Model and Current Allocation Module.

This module handles the electrical grid representation, including three single-phase
connections and one three-phase connection, along with logic for allocating available
current across different charging phases based on fleet composition and energy needs.

Original Source:
    - amperage_splitter.py
    - functions.py (grid-related constraints and availability checks)
"""

from typing import Dict, List, Tuple, Optional, Union, Any
import numpy as np
from dataclasses import dataclass


@dataclass
class GridLimits:
    """
    Physical limits of the grid infrastructure.

    Attributes:
        max_current_per_phase: Maximum current per single phase in A (default: 245.6 A)
        max_current_three_phase: Maximum three-phase current in A (default: 736.8 A)
        single_phase_voltage: Single-phase voltage in V (default: 230 V)
        three_phase_voltage: Three-phase voltage in V (default: 400 V)
        safety_coefficient: Safety factor for current limits (default: 1.0)
    """
    max_current_per_phase: float = 245.6
    max_current_three_phase: float = 736.8
    single_phase_voltage: float = 230.0
    three_phase_voltage: float = 400.0
    safety_coefficient: float = 1.0


class ThreePhaseGrid:
    """
    Represents a three-phase electrical grid for EV charging.

    This class manages the available current on each phase and handles the allocation
    of current between single-phase (phases 1-3) and three-phase (phase 4) charging.

    The grid consists of:
    - Three single-phase connections (Phase 1, 2, 3)
    - One three-phase connection (Phase 4)

    The allocation algorithm dynamically distributes available current based on:
    - Energy needs of each fleet (single-phase vs three-phase)
    - Average remaining time for charging
    - Minimum current requirements per EV type

    Attributes:
        limits: GridLimits instance defining infrastructure limits
        available_current_phase1: Available current on phase 1 in A
        available_current_phase2: Available current on phase 2 in A
        available_current_phase3: Available current on phase 3 in A
        available_current_three_phase: Available current for three-phase charging in A
        voltage_phase1: Measured voltage on phase 1 in V
        voltage_phase2: Measured voltage on phase 2 in V
        voltage_phase3: Measured voltage on phase 3 in V
        voltage_three_phase: Calculated three-phase voltage in V

    Example:
        >>> grid = ThreePhaseGrid()
```

```
        >>> grid.update_from_measurements(
        ...     phase_1_current=100, phase_2_current=90, phase_3_current=95,
        ...     phase_1_voltage=230, phase_2_voltage=229, phase_3_voltage=231
        ... )
        >>> allocation = grid.allocate_current(fleet_phase_1, fleet_phase_2,
        ...                                    fleet_phase_3, fleet_tri_phase)
        >>> print(f"Allocated to phase 1: {allocation[0]:.1f} A")
    """

    def __init__(self, limits: Optional[GridLimits] = None):
        """
        Initialize the three-phase grid.

        Args:
            limits: GridLimits instance or None for defaults

        Example:
            >>> # Use default limits
            >>> grid = ThreePhaseGrid()
            >>>
            >>> # Use custom limits
            >>> custom_limits = GridLimits(
            ...     max_current_per_phase=200.0,
            ...     safety_coefficient=0.95
            ... )
            >>> grid = ThreePhaseGrid(limits=custom_limits)
        """
        self.limits = limits if limits is not None else GridLimits()

        # Initialize available currents
        self.available_current_phase1: float = 0.0
        self.available_current_phase2: float = 0.0
        self.available_current_phase3: float = 0.0
        self.available_current_three_phase: float = 0.0

        # Initialize voltages
        self.voltage_phase1: float = self.limits.single_phase_voltage
        self.voltage_phase2: float = self.limits.single_phase_voltage
        self.voltage_phase3: float = self.limits.single_phase_voltage
        self.voltage_three_phase: float = self.limits.three_phase_voltage

    def update_from_measurements(
        self,
        phase_1_current: float,
        phase_2_current: float,
        phase_3_current: float,
        phase_1_voltage: float,
        phase_2_voltage: float,
        phase_3_voltage: float
    ) -> None:
        """
        Update grid state from sensor measurements.

        Args:
            phase_1_current: Measured available current on phase 1 in A
            phase_2_current: Measured available current on phase 2 in A
            phase_3_current: Measured available current on phase 3 in A
            phase_1_voltage: Measured voltage on phase 1 in V
            phase_2_voltage: Measured voltage on phase 2 in V
            phase_3_voltage: Measured voltage on phase 3 in V

        Example:
            >>> grid = ThreePhaseGrid()
            >>> grid.update_from_measurements(
            ...     phase_1_current=120.0, phase_2_current=115.0, phase_3_current=118.0,
            ...     phase_1_voltage=230.0, phase_2_voltage=229.5, phase_3_voltage=230.5
            ... )
            >>> print(f"Phase 1: {grid.available_current_phase1:.1f} A")
            Phase 1: 120.0 A
        """
        # Update available currents
        self.available_current_phase1 = phase_1_current
        self.available_current_phase2 = phase_2_current
        self.available_current_phase3 = phase_3_current
```

```
        # Update voltages
        self.voltage_phase1 = phase_1_voltage
        self.voltage_phase2 = phase_2_voltage
        self.voltage_phase3 = phase_3_voltage

        # Calculate three-phase voltage
        self.voltage_three_phase = calculate_three_phase_voltage(
            phase_1_voltage, phase_2_voltage, phase_3_voltage
        )

    def allocate_current(
        self,
        fleet_phase_1: Any,
        fleet_phase_2: Any,
        fleet_phase_3: Any,
        fleet_tri_phase: Any
    ) -> Tuple[float, float, float, float]:
        """
        Allocate available current across phases based on fleet composition.

        This function implements the dynamic allocation strategy that:
        1. Calculates total energy needs and average remaining time for each fleet
        2. Determines the power rate (W) for single-phase vs three-phase fleets
        3. Allocates current proportionally based on power rates
        4. Ensures minimum current for three-phase charging (16A per EV)

        Algorithm:
        - If only three-phase EVs: allocate all current to three-phase
        - If only single-phase EVs: allocate to individual phases
        - If mixed: proportionally allocate based on energy rate ratios
        - Special case: if no energy needs, use 75/25 split (single/three-phase)

        Args:
            fleet_phase_1: List of EVs (or EV dicts) on phase 1
            fleet_phase_2: List of EVs (or EV dicts) on phase 2
            fleet_phase_3: List of EVs (or EV dicts) on phase 3
            fleet_tri_phase: List of EVs (or EV dicts) on three-phase charging

        Returns:
            Tuple of (allocated_phase_1, allocated_phase_2, allocated_phase_3,
                     allocated_tri_phase) in Amperes

        Example:
            >>> grid = ThreePhaseGrid()
            >>> grid.update_from_measurements(100, 100, 100, 230, 230, 230)
            >>> # fleet_p1, fleet_p2, fleet_p3 contain EVs needing charge
            >>> alloc_1, alloc_2, alloc_3, alloc_4 = grid.allocate_current(
            ...     fleet_p1, fleet_p2, fleet_p3, fleet_tp
            ... )
            >>> print(f"Phase 1: {alloc_1:.1f} A, Three-phase: {alloc_4:.1f} A")
        """
        # Calculate fleet sizes
        fleet_slow_size = len(fleet_phase_1) + len(fleet_phase_2) + len(fleet_phase_3)
        fleet_fast_size = len(fleet_tri_phase)

        # Case 1: Only three-phase EVs (no single-phase)
        if fleet_slow_size == 0:
            return (
                0.0,
                0.0,
                0.0,
                self.available_current_phase1 + self.available_current_phase2 +
                self.available_current_phase3
            )

        # Case 2: Only single-phase EVs (no three-phase)
        if fleet_fast_size == 0:
            return (
                self.available_current_phase1,
                self.available_current_phase2,
                self.available_current_phase3,
                0.0
            )
```

```
        # Case 3: Mixed fleet - calculate energy needs
        overall_energy_needs_slow = 0.0
        overall_energy_needs_fast = 0.0
        sum_remaining_time_slow = 0.0
        sum_remaining_time_fast = 0.0

        # Sum up energy needs for single-phase EVs
        for partial_fleet in (fleet_phase_1, fleet_phase_2, fleet_phase_3):
            for ev in partial_fleet:
                # Handle both EV objects and dictionaries
                missing_soc = ev['Missing SOC'] if isinstance(ev, dict) else ev.missing_soc
                remaining_time = ev['Remaining Time'] if isinstance(ev, dict) else
ev.remaining_time

                if missing_soc > 0:
                    overall_energy_needs_slow += missing_soc
                    sum_remaining_time_slow += remaining_time

        # Sum up energy needs for three-phase EVs
        for ev in fleet_tri_phase:
            missing_soc = ev['Missing SOC'] if isinstance(ev, dict) else ev.missing_soc
            remaining_time = ev['Remaining Time'] if isinstance(ev, dict) else ev.remaining_time

            if missing_soc > 0:
                overall_energy_needs_fast += missing_soc
                sum_remaining_time_fast += remaining_time

        # Calculate average remaining times
        average_remaining_time_slow = (sum_remaining_time_slow / fleet_slow_size
                                       if fleet_slow_size > 0 else 0.0)
        average_remaining_time_fast = (sum_remaining_time_fast / fleet_fast_size
                                       if fleet_fast_size > 0 else 0.0)

        # Calculate energy rates (Wh/h = W)
        energy_rate_slow = (overall_energy_needs_slow / average_remaining_time_slow
                            if average_remaining_time_slow > 0 else 0.0)
        energy_rate_fast = (overall_energy_needs_fast / average_remaining_time_fast
                            if average_remaining_time_fast > 0 else 0.0)

        total_energy_rate = energy_rate_slow + energy_rate_fast

        # Special case: no energy needs or no time - use default 75/25 split
        if total_energy_rate <= 0:
            total_current = (self.available_current_phase1 +
                             self.available_current_phase2 +
                             self.available_current_phase3)

            allocated_4 = total_current * 0.25
            allocated_1 = self.available_current_phase1 * 0.75
            allocated_2 = self.available_current_phase2 * 0.75
            allocated_3 = self.available_current_phase3 * 0.75

            return allocated_1, allocated_2, allocated_3, allocated_4

        # Normal case: allocate based on energy rate ratio
        fast_ratio = energy_rate_fast / total_energy_rate

        total_current = (self.available_current_phase1 +
                         self.available_current_phase2 +
                         self.available_current_phase3)

        # Allocate to three-phase, ensuring minimum 16A per EV
        allocated_4 = max(total_current * fast_ratio, 16.0 * fleet_fast_size)

        # Remaining current distributed equally across single phases
        allocated_1 = self.available_current_phase1 - allocated_4 / 3
        allocated_2 = self.available_current_phase2 - allocated_4 / 3
        allocated_3 = self.available_current_phase3 - allocated_4 / 3

        return allocated_1, allocated_2, allocated_3, allocated_4

    def check_availability(
        self,
```

```
        fleet_phase_1: Union[List[Dict], List],
        fleet_phase_2: Union[List[Dict], List],
        fleet_phase_3: Union[List[Dict], List],
        fleet_tri_phase: Union[List[Dict], List],
        fleet_discharging: Union[List[Dict], List],
        current_hour: float,
        min_current_single: float = 6.0,
        min_current_three: float = 16.0
    ) -> Tuple[List, List, List, List, List]:
        """
        Verify that sufficient current is available for all EVs.

        If there is insufficient current to charge all EVs at their minimum rates,
        EVs with the lowest missing SoC are moved to a waiting (discharging) fleet
        until the available current can support the remaining fleet.

        The algorithm:
        1. For each phase, check if available_current * safety_coefficient >= fleet_size *
min_current
        2. If insufficient, remove EVs with lowest missing SoC until constraint satisfied
        3. Update removed EVs' plugin time to retry later (current_hour + 0.25h)

        Args:
            fleet_phase_1: EVs on phase 1
            fleet_phase_2: EVs on phase 2
            fleet_phase_3: EVs on phase 3
            fleet_tri_phase: EVs on three-phase
            fleet_discharging: EVs waiting to charge
            current_hour: Current simulation time in decimal hours
            min_current_single: Minimum current per single-phase EV in A
            min_current_three: Minimum current per three-phase EV in A

        Returns:
            Tuple of (updated_fleet_phase_1, updated_fleet_phase_2,
                     updated_fleet_phase_3, updated_fleet_tri_phase,
                     updated_fleet_discharging) with EVs redistributed

        Example:
            >>> # Available current insufficient for all EVs
            >>> fleet_1, fleet_2, fleet_3, fleet_4, fleet_d = grid.check_availability(
            ...     fleet_phase_1, fleet_phase_2, fleet_phase_3,
            ...     fleet_tri_phase, fleet_discharging, current_hour=19.5
            ... )
            >>> print(f"Postponed {len(fleet_d) - original_discharging_size} EVs")
        """
        from .ev_fleet import decimal_to_time_string

        # Convert to numpy arrays for easier manipulation (if not already)
        fleet_1 = np.array(fleet_phase_1) if not isinstance(fleet_phase_1, np.ndarray) else
fleet_phase_1
        fleet_2 = np.array(fleet_phase_2) if not isinstance(fleet_phase_2, np.ndarray) else
fleet_phase_2
        fleet_3 = np.array(fleet_phase_3) if not isinstance(fleet_phase_3, np.ndarray) else
fleet_phase_3
        fleet_4 = np.array(fleet_tri_phase) if not isinstance(fleet_tri_phase, np.ndarray) else
fleet_tri_phase
        fleet_d = np.array(fleet_discharging) if not isinstance(fleet_discharging, np.ndarray)
else fleet_discharging

        safety = self.limits.safety_coefficient

        # Check Phase 1
        while (len(fleet_1) > 0 and
               self.available_current_phase1 * safety - len(fleet_1) * min_current_single < 0):
            # Get missing SoC for all EVs on this phase
            missing_soc = np.array([
                (ev['Missing SOC'].item() if isinstance(ev['Missing SOC'], np.ndarray)
                 else ev['Missing SOC'])
                if isinstance(ev, dict) else ev.missing_soc
                for ev in fleet_1
            ])

            # Find EV with lowest missing SoC (least urgent)
            least_urgent_idx = np.argmin(missing_soc)
```

```
            ev_to_move = fleet_1[least_urgent_idx]

            # Update plugin time to retry in 15 minutes (0.25 hours)
            new_plugin_time = decimal_to_time_string(current_hour + 0.25)
            plugout_time = (ev_to_move['Plugout Time'] if isinstance(ev_to_move, dict)
                           else ev_to_move.plugout_time)

            # Only postpone if there's still time before plugout
            if new_plugin_time < plugout_time:
                if isinstance(ev_to_move, dict):
                    ev_to_move['Plugin Time'] = new_plugin_time
                else:
                    ev_to_move.plugin_time = new_plugin_time

            # Move to discharging fleet
            fleet_d = np.append(fleet_d, ev_to_move)
            fleet_1 = np.delete(fleet_1, least_urgent_idx)

        # Check Phase 2
        while (len(fleet_2) > 0 and
               self.available_current_phase2 * safety - len(fleet_2) * min_current_single < 0):
            missing_soc = np.array([
                (ev['Missing SOC'].item() if isinstance(ev['Missing SOC'], np.ndarray)
                 else ev['Missing SOC'])
                if isinstance(ev, dict) else ev.missing_soc
                for ev in fleet_2
            ])

            least_urgent_idx = np.argmin(missing_soc)
            ev_to_move = fleet_2[least_urgent_idx]

            new_plugin_time = decimal_to_time_string(current_hour + 0.25)
            plugout_time = (ev_to_move['Plugout Time'] if isinstance(ev_to_move, dict)
                           else ev_to_move.plugout_time)

            if new_plugin_time < plugout_time:
                if isinstance(ev_to_move, dict):
                    ev_to_move['Plugin Time'] = new_plugin_time
                else:
                    ev_to_move.plugin_time = new_plugin_time

            fleet_d = np.append(fleet_d, ev_to_move)
            fleet_2 = np.delete(fleet_2, least_urgent_idx)

        # Check Phase 3
        while (len(fleet_3) > 0 and
               self.available_current_phase3 * safety - len(fleet_3) * min_current_single < 0):
            missing_soc = np.array([
                (ev['Missing SOC'].item() if isinstance(ev['Missing SOC'], np.ndarray)
                 else ev['Missing SOC'])
                if isinstance(ev, dict) else ev.missing_soc
                for ev in fleet_3
            ])

            least_urgent_idx = np.argmin(missing_soc)
            ev_to_move = fleet_3[least_urgent_idx]

            new_plugin_time = decimal_to_time_string(current_hour + 0.25)
            plugout_time = (ev_to_move['Plugout Time'] if isinstance(ev_to_move, dict)
                           else ev_to_move.plugout_time)

            if new_plugin_time < plugout_time:
                if isinstance(ev_to_move, dict):
                    ev_to_move['Plugin Time'] = new_plugin_time
                else:
                    ev_to_move.plugin_time = new_plugin_time

            fleet_d = np.append(fleet_d, ev_to_move)
            fleet_3 = np.delete(fleet_3, least_urgent_idx)

        # Check Three-Phase
        # Note: available_current_three_phase needs to be set by allocate_current first
        available_tri = self.available_current_three_phase
        if available_tri == 0:  # Not yet allocated
```

```
            available_tri = (self.available_current_phase1 +
                             self.available_current_phase2 +
                             self.available_current_phase3)

        while (len(fleet_4) > 0 and
               available_tri * safety - len(fleet_4) * min_current_three < 0):
            missing_soc = np.array([
                (ev['Missing SOC'].item() if isinstance(ev['Missing SOC'], np.ndarray)
                 else ev['Missing SOC'])
                if isinstance(ev, dict) else ev.missing_soc
                for ev in fleet_4
            ])

            least_urgent_idx = np.argmin(missing_soc)
            ev_to_move = fleet_4[least_urgent_idx]

            new_plugin_time = decimal_to_time_string(current_hour + 0.25)
            plugout_time = (ev_to_move['Plugout Time'] if isinstance(ev_to_move, dict)
                            else ev_to_move.plugout_time)

            if new_plugin_time < plugout_time:
                if isinstance(ev_to_move, dict):
                    ev_to_move['Plugin Time'] = new_plugin_time
                else:
                    ev_to_move.plugin_time = new_plugin_time

            fleet_d = np.append(fleet_d, ev_to_move)
            fleet_4 = np.delete(fleet_4, least_urgent_idx)

        # Convert back to lists
        return (fleet_1.tolist() if len(fleet_1) > 0 else [],
                fleet_2.tolist() if len(fleet_2) > 0 else [],
                fleet_3.tolist() if len(fleet_3) > 0 else [],
                fleet_4.tolist() if len(fleet_4) > 0 else [],
                fleet_d.tolist() if len(fleet_d) > 0 else [])

    def get_voltage(self, phase: int) -> float:
        """
        Get voltage for a specific phase.

        Args:
            phase: Phase number (1-4)

        Returns:
            Voltage in V (single-phase voltage for 1-3, three-phase voltage for 4)

        Raises:
            ValueError: If phase number is not 1-4

        Example:
            >>> grid = ThreePhaseGrid()
            >>> grid.update_from_measurements(100, 100, 100, 230, 229, 231)
            >>> print(f"Phase 1 voltage: {grid.get_voltage(1):.1f} V")
            Phase 1 voltage: 230.0 V
            >>> print(f"Three-phase voltage: {grid.get_voltage(4):.1f} V")
            Three-phase voltage: 398.4 V
        """
        if phase == 1:
            return self.voltage_phase1
        elif phase == 2:
            return self.voltage_phase2
        elif phase == 3:
            return self.voltage_phase3
        elif phase == 4:
            return self.voltage_three_phase
        else:
            raise ValueError(f"Invalid phase number: {phase}. Must be 1-4.")

    def get_available_current(self, phase: int) -> float:
        """
        Get available current for a specific phase.

        Args:
            phase: Phase number (1-4)
```

```
        Returns:
            Available current in A

        Raises:
            ValueError: If phase number is not 1-4

        Example:
            >>> grid = ThreePhaseGrid()
            >>> grid.update_from_measurements(100, 95, 105, 230, 230, 230)
            >>> print(f"Phase 1 current: {grid.get_available_current(1):.1f} A")
            Phase 1 current: 100.0 A
        """
        if phase == 1:
            return self.available_current_phase1
        elif phase == 2:
            return self.available_current_phase2
        elif phase == 3:
            return self.available_current_phase3
        elif phase == 4:
            return self.available_current_three_phase
        else:
            raise ValueError(f"Invalid phase number: {phase}. Must be 1-4.")

    def is_within_limits(self, phase: int, requested_current: float) -> bool:
        """
        Check if requested current is within grid limits for a phase.

        Args:
            phase: Phase number (1-4)
            requested_current: Requested current in A

        Returns:
            True if within limits, False otherwise

        Raises:
            ValueError: If phase number is not 1-4

        Example:
            >>> grid = ThreePhaseGrid()
            >>> # Check if 200A is within limits for single phase
            >>> grid.is_within_limits(1, 200.0)
            True
            >>> # Check if 300A is within limits (exceeds 245.6A default)
            >>> grid.is_within_limits(1, 300.0)
            False
        """
        if phase in (1, 2, 3):
            return requested_current <= self.limits.max_current_per_phase
        elif phase == 4:
            return requested_current <= self.limits.max_current_three_phase
        else:
            raise ValueError(f"Invalid phase number: {phase}. Must be 1-4.")


class CurrentBounds:
    """
    Defines current bounds for optimisation constraints.

    This class provides the min/max current limits for different types of
    charging connections as required by the optimisation algorithms.

    The bounds are based on:
    - IEC 61851 standard (minimum 6A for single-phase)
    - Slow charging infrastructure (maximum 16A single-phase, 32A three-phase)

    Attributes:
        single_phase_min: Minimum current for single-phase charging in A (default: 6.0)
        single_phase_max: Maximum current for single-phase charging in A (default: 16.0)
        three_phase_min: Minimum current for three-phase charging in A (default: 16.0)
        three_phase_max: Maximum current for three-phase charging in A (default: 32.0)

    Example:
        >>> bounds = CurrentBounds()
```

```
        >>> fleet_bounds = bounds.get_bounds_for_fleet(my_fleet, is_three_phase=False)
        >>> print(f"Single EV bounds: {fleet_bounds[0]}")  # (6.0, 16.0)
    """

    def __init__(
        self,
        single_phase_min: float = 6.0,
        single_phase_max: float = 16.0,
        three_phase_min: float = 16.0,
        three_phase_max: float = 32.0
    ):
        """
        Initialize current bounds.

        Args:
            single_phase_min: Minimum current for single-phase in A
            single_phase_max: Maximum current for single-phase in A
            three_phase_min: Minimum current for three-phase in A
            three_phase_max: Maximum current for three-phase in A

        Example:
            >>> # Default bounds (IEC 61851 compliant)
            >>> bounds = CurrentBounds()
            >>>
            >>> # Custom bounds for different infrastructure
            >>> custom_bounds = CurrentBounds(
            ...     single_phase_min=8.0,
            ...     single_phase_max=20.0
            ... )
        """
        self.single_phase_min = single_phase_min
        self.single_phase_max = single_phase_max
        self.three_phase_min = three_phase_min
        self.three_phase_max = three_phase_max

    def get_bounds_for_fleet(
        self,
        fleet: Union[List[Dict], List],
        is_three_phase: bool = False
    ) -> List[Tuple[float, float]]:
        """
        Generate optimisation bounds for a fleet.

        Creates a list of (min, max) tuples for scipy.optimise functions,
        one tuple per EV in the fleet.

        Args:
            fleet: List of EVs (or EV dictionaries)
            is_three_phase: True if fleet uses three-phase charging

        Returns:
            List of (min, max) tuples, one per EV

        Example:
            >>> bounds = CurrentBounds()
            >>> single_fleet = [ev1, ev2, ev3]  # 3 single-phase EVs
            >>> fleet_bounds = bounds.get_bounds_for_fleet(single_fleet, is_three_phase=False)
            >>> print(fleet_bounds)
            [(6.0, 16.0), (6.0, 16.0), (6.0, 16.0)]
            >>>
            >>> three_fleet = [ev4, ev5]  # 2 three-phase EVs
            >>> fleet_bounds = bounds.get_bounds_for_fleet(three_fleet, is_three_phase=True)
            >>> print(fleet_bounds)
            [(16.0, 32.0), (16.0, 32.0)]
        """
        fleet_size = len(fleet)

        if is_three_phase:
            return [(self.three_phase_min, self.three_phase_max)] * fleet_size
        else:
            return [(self.single_phase_min, self.single_phase_max)] * fleet_size

    def get_single_phase_bounds(self) -> Tuple[float, float]:
        """
```

```
        Get (min, max) bounds for single-phase charging.

        Returns:
            Tuple of (min_current, max_current) in A

        Example:
            >>> bounds = CurrentBounds()
            >>> min_i, max_i = bounds.get_single_phase_bounds()
            >>> print(f"Single-phase: {min_i}-{max_i} A")
            Single-phase: 6.0-16.0 A
        """
        return (self.single_phase_min, self.single_phase_max)

    def get_three_phase_bounds(self) -> Tuple[float, float]:
        """
        Get (min, max) bounds for three-phase charging.

        Returns:
            Tuple of (min_current, max_current) in A

        Example:
            >>> bounds = CurrentBounds()
            >>> min_i, max_i = bounds.get_three_phase_bounds()
            >>> print(f"Three-phase: {min_i}-{max_i} A")
            Three-phase: 16.0-32.0 A
        """
        return (self.three_phase_min, self.three_phase_max)


def calculate_three_phase_voltage(
    voltage_1: float,
    voltage_2: float,
    voltage_3: float
) -> float:
    """
    Calculate effective three-phase voltage from single-phase measurements.

    Uses the formula: V_3phase = avg(V1, V2, V3) × √3

    This approximation is valid for balanced three-phase systems where
    the line-to-line voltage is √3 times the phase voltage.

    Args:
        voltage_1: Phase 1 voltage in V
        voltage_2: Phase 2 voltage in V
        voltage_3: Phase 3 voltage in V

    Returns:
        Effective three-phase voltage in V

    Example:
        >>> # Typical European grid voltages
        >>> v_3phase = calculate_three_phase_voltage(230, 229, 231)
        >>> print(f"Three-phase voltage: {v_3phase:.1f} V")
        Three-phase voltage: 398.4 V
        >>>
        >>> # Perfect balance
        >>> v_3phase = calculate_three_phase_voltage(230, 230, 230)
        >>> print(f"Three-phase voltage: {v_3phase:.1f} V")
        Three-phase voltage: 398.4 V
    """
    average_voltage = (voltage_1 + voltage_2 + voltage_3) / 3.0
    return average_voltage * np.sqrt(3)


def calculate_total_available_power(
    phase_1_current: float,
    phase_2_current: float,
    phase_3_current: float,
    phase_1_voltage: float,
    phase_2_voltage: float,
    phase_3_voltage: float
) -> float:
    """
```

```
    Calculate total available power from all three phases.

    Computes: P_total = V1×I1 + V2×I2 + V3×I3

    Args:
        phase_1_current: Available current on phase 1 in A
        phase_2_current: Available current on phase 2 in A
        phase_3_current: Available current on phase 3 in A
        phase_1_voltage: Voltage on phase 1 in V
        phase_2_voltage: Voltage on phase 2 in V
        phase_3_voltage: Voltage on phase 3 in V

    Returns:
        Total available power in W

    Example:
        >>> # Calculate total power with 100A available on each phase
        >>> power = calculate_total_available_power(
        ...     100, 100, 100,  # currents in A
        ...     230, 230, 230   # voltages in V
        ... )
        >>> print(f"Total power: {power/1000:.1f} kW")
        Total power: 69.0 kW
        >>>
        >>> # Unbalanced phases
        >>> power = calculate_total_available_power(
        ...     120, 100, 90,    # currents in A
        ...     230, 229, 231    # voltages in V
        ... )
        >>> print(f"Total power: {power/1000:.1f} kW")
        Total power: 71.1 kW
    """
    power_phase_1 = phase_1_voltage * phase_1_current
    power_phase_2 = phase_2_voltage * phase_2_current
    power_phase_3 = phase_3_voltage * phase_3_current

    return power_phase_1 + power_phase_2 + power_phase_3


def create_optimisation_constraints(
    available_current: float,
    safety_coefficient: float = 1.0
) -> Dict:
    """
    Create constraint dictionary for scipy.optimise.minimise.

    Generates an inequality constraint of the form:
        available_current × safety_coefficient - sum(x) >= 0

    Where x is the array of currents allocated to individual EVs.

    Args:
        available_current: Maximum available current for this phase in A
        safety_coefficient: Safety factor (default: 1.0)

    Returns:
        Constraint dictionary with 'type', 'fun', and 'args'

    Example:
        >>> # Create constraint for phase with 100A available
        >>> constraint = create_optimisation_constraints(100.0, safety_coefficient=1.0)
        >>>
        >>> # Use in scipy.optimise.minimise
        >>> from scipy.optimise import minimise
        >>> result = minimise(
        ...     objective_function,
        ...     x0,
        ...     constraints=[constraint],
        ...     bounds=bounds
        ... )
    """
    def constraint_function(x: np.ndarray, avail_amp: float, safety: float) -> float:
        """Constraint: available_current * safety - sum(x) >= 0"""
        return avail_amp * safety - np.sum(x)
```

```
    return {
        'type': 'ineq',
        'fun': constraint_function,
        'args': (available_current, safety_coefficient)
    }
```

B.4 Optimised Charging Simulation

This module implements the real-time optimised charging algorithm that optimises charging currents as EVs plug in, considering battery State of Health degradation and thermal behaviour. The objective minimises deviation from desired charging times while respecting grid constraints.

```
"""
Ad-hoc Optimised Charging Algorithm

This module implements the real-time charging optimisation strategy that:
1. Optimises charging currents as EVs plug in
2. Considers battery State of Health (SoH) degradation
3. Models battery thermal behaviour
4. Minimises deviation from desired charging times

Key Components:
- OptimisedChargingSimulation: Main simulation class
- Optimisation objective: Minimise |E_missing/(I×V) - T_remaining|
- Constraints: Grid capacity, current limits, SoC bounds

Original File: Optimised_Charging_3_Phases_fast_soh.py

Author: Salvador Carvalhosa
"""

from typing import Dict, List, Optional, Tuple, Any
import numpy as np
import pandas as pd
from dataclasses import dataclass
from scipy.optimise import minimise
import datetime as dt

from models.grid import ThreePhaseGrid
from models.battery import BatteryModel
from utils.time_utils import convert_decimal_to_time


from enum import Enum


class ObjectiveType(Enum):
    """Available optimisation objective functions."""
    TIME_COMPLETION = "time_completion"  # Minimise deviation from target time
    BATTERY_HEALTH = "battery_health"    # Minimise battery degradation
    MINIMUM_CURRENT = "minimum_current"  # Use minimum charging current
    LOAD_BALANCING = "load_balancing"    # Balance load across EVs


@dataclass
class SimulationConfig:
    """Configuration for optimised charging simulation."""

    data_file: str
    time_step: float = 0.05  # hours (3 minutes)
    safety_coefficient: float = 1.0
    lower_bound_single_phase: float = 6.0  # Amps
    upper_bound_single_phase: float = 16.0  # Amps
    lower_bound_three_phase: float = 16.0  # Amps
    upper_bound_three_phase: float = 32.0  # Amps
    min_soc: float = 0.2
    max_soc: float = 0.8
    charging_threshold: float = 0.4
    discharge_time: str = "18:00"  # Time when EVs discharge daily

    # Objective function selection
    objective: ObjectiveType = ObjectiveType.TIME_COMPLETION


class OptimisedChargingSimulation:
    """
    Real-time optimised charging simulation with battery health modelling.
```

```
    This class implements the ad-hoc optimisation strategy where charging
    currents are optimised at each time step to minimise the deviation
    between required charging time and available time, while considering
    battery degradation and thermal constraints.

    Attributes:
        config: Simulation configuration
        fleet_discharging: EVs not currently charging
        fleet_charging_phase_1: EVs charging on phase 1
        fleet_charging_phase_2: EVs charging on phase 2
        fleet_charging_phase_3: EVs charging on phase 3
        fleet_charging_tri_phase: EVs charging on three phases
        grid: Three-phase grid model
        grid_data: DataFrame with grid availability data
        results: Dictionary containing simulation results

    Example:
        >>> from models.ev_fleet import EVFleet
        >>> fleet = EVFleet(fleet_size=10)
        >>> evs = fleet.get_ev_dicts()  # Get dictionary format
        >>> sim = OptimisedChargingSimulation(evs, "Time_Step_3_Long.csv")
        >>> results = sim.run()
        >>> print(f"Peak demand: {results['peak_demand']:.1f} A")
    """

    def __init__(
        self,
        fleet: List[Dict],
        data_file: str,
        config: Optional[SimulationConfig] = None
    ):
        """
        Initialize the optimised charging simulation.

        Args:
            fleet: List of EV dictionaries with initial state
            data_file: Path to grid availability CSV file
            config: Optional simulation configuration
        """
        self.config = config or SimulationConfig(data_file=data_file)
        self.grid_data = self._load_grid_data()

        # Initialize fleet arrays (all EVs start in discharging state)
        self.fleet_discharging = np.array(fleet)
        self.fleet_charging_phase_1 = np.empty(shape=(0,))
        self.fleet_charging_phase_2 = np.empty(shape=(0,))
        self.fleet_charging_phase_3 = np.empty(shape=(0,))
        self.fleet_charging_tri_phase = np.empty(shape=(0,))

        # Initialize grid model
        self.grid = ThreePhaseGrid()

        # Initialize state tracking
        self.current_hour = 0.0
        self.order = 0  # Plugin order counter

        # Initialize tracking arrays
        self._init_tracking_arrays()

        # Initialize results dictionary
        self.results: Dict[str, Any] = {}

    def _load_grid_data(self) -> pd.DataFrame:
        """
        Load and validate grid availability data.

        Returns:
            DataFrame with columns: Phase_1, Phase_2, Phase_3, Total, U1, U2, U3
        """
        df = pd.read_csv(self.config.data_file)

        # Validate required columns
        required_cols = ['Phase_1', 'Phase_2', 'Phase_3', 'Total', 'U1', 'U2', 'U3']
```

```
        missing_cols = [col for col in required_cols if col not in df.columns]
        if missing_cols:
            raise ValueError(f"Missing required columns in grid data: {missing_cols}")

        return df

    def _init_tracking_arrays(self):
        """Initialize arrays for tracking simulation metrics."""
        n_steps = len(self.grid_data)

        self.allocated_amps_phase_1 = np.empty(shape=(0,))
        self.allocated_amps_phase_2 = np.empty(shape=(0,))
        self.allocated_amps_phase_3 = np.empty(shape=(0,))
        self.allocated_amps_tri_phase = np.empty(shape=(0,))
        self.allocated_amps_overall = np.empty(shape=(0,))

        # Track maximum values
        self.max_charging_fleet_size = 0
        self.max_fleet_phase_1_size = 0
        self.max_fleet_phase_2_size = 0
        self.max_fleet_phase_3_size = 0
        self.max_fleet_tri_phase_size = 0
        self.max_overall_current = 0

    def run(self) -> Dict:
        """
        Run the complete simulation.

        Iterates through all time steps, managing EV plugin/plugout events,
        optimising charging currents, updating EV states, and collecting metrics.

        Returns:
            Dictionary with results:
                - total_consumption: Total energy consumed (Wh)
                - peak_demand: Maximum simultaneous demand (A)
                - max_charging_fleet_size: Maximum number of EVs charging simultaneously
                - allocated_currents: Dictionary with per-phase allocated currents
                - fleet_state: Final state of all EVs
        """
        # Main simulation loop over all time steps
        for i in range(len(self.grid_data)):
            self._simulate_time_step(i)

            # Update current hour
            self.current_hour = (self.current_hour + self.config.time_step) % 24

        # Compute final results
        return self._compute_results()

    def _simulate_time_step(self, step_index: int):
        """
        Simulate a single time step.

        Args:
            step_index: Index of current time step in grid_data
        """
        # 1. Update logs for all EVs
        self._update_logs()

        # 2. Check for plugin events
        self._process_plugin_events()

        # 3. Get grid availability
        available_currents, voltages = self._get_grid_state(step_index)

        # 4. Update grid state with current measurements
        self.grid.update_from_measurements(
            phase_1_current=available_currents['phase_1'],
            phase_2_current=available_currents['phase_2'],
            phase_3_current=available_currents['phase_3'],
            phase_1_voltage=voltages['phase_1'],
            phase_2_voltage=voltages['phase_2'],
            phase_3_voltage=voltages['phase_3']
        )
```

```
        # 5. Allocate currents across phases using grid model
        phase_1, phase_2, phase_3, tri_phase = self.grid.allocate_current(
            self.fleet_charging_phase_1,
            self.fleet_charging_phase_2,
            self.fleet_charging_phase_3,
            self.fleet_charging_tri_phase
        )

        # 6. Check availability and adjust fleets if needed
        self._check_and_adjust_availability(phase_1, phase_2, phase_3, tri_phase)

        # 7. Optimise and charge EVs on each phase
        self._charge_all_phases(phase_1, phase_2, phase_3, tri_phase, voltages, step_index)

        # 8. Check for plugout events
        self._process_plugout_events()

        # 9. Process daily discharge if at discharge time
        self._process_daily_discharge()

        # 10. Finalize current logs for charging EVs
        self._finalize_current_logs()

    def _finalize_current_logs(self):
        """
        Finalize current logs for all charging EVs after optimisation.

        This ensures Current Log stays synchronised with SOC Log.
        Charging EVs that weren't charged this step get 0 recorded.
        """
        for fleet in (self.fleet_charging_phase_1, self.fleet_charging_phase_2,
                      self.fleet_charging_phase_3, self.fleet_charging_tri_phase):
            for ev in fleet:
                # Get the pending current (set in _update_ev_states or initialized to 0)
                current = ev.get("_pending_current", 0.0)
                ev["Current Log"] = np.append(ev["Current Log"], current)
                # Clean up temporary field
                if "_pending_current" in ev:
                    del ev["_pending_current"]

    def _update_logs(self):
        """
        Update state logs for all EVs at current time step.

        Note: Current Log is handled separately to ensure synchronisation:
        - For discharging EVs: Current Log = 0 (added here)
        - For charging EVs: Current Log = actual current (added in _update_ev_states)
        """
        # Update discharging fleet logs
        for ev in self.fleet_discharging:
            ev["SOC Log"] = np.append(ev["SOC Log"], ev["Current SOC"] / ev["Maximum SOC"])
            ev["Current Log"] = np.append(ev["Current Log"], 0)
            ev["Cost Log"] = np.append(ev["Cost Log"], 0)
            ev["Desired Log"] = np.append(ev["Desired Log"], ev["Desired SOC"] / ev["Maximum
SOC"])
            ev["Temperature Log"] = np.append(ev["Temperature Log"], ev["Temperature"])

        # Update charging fleet logs (SOC, Desired, Temperature)
        # Current Log is updated in _charge_phase() after optimisation
        for fleet in (self.fleet_charging_phase_1, self.fleet_charging_phase_2,
                      self.fleet_charging_phase_3, self.fleet_charging_tri_phase):
            for ev in fleet:
                ev["SOC Log"] = np.append(ev["SOC Log"], ev["Current SOC"] / ev["Maximum SOC"])
                ev["Desired Log"] = np.append(ev["Desired Log"], ev["Desired SOC"] / ev["Maximum
SOC"])
                ev["Temperature Log"] = np.append(ev["Temperature Log"], ev["Temperature"])
                # Initialize current for this step as 0 (will be overwritten if charging occurs)
                ev["_pending_current"] = 0.0

    def _process_plugin_events(self):
        """Check for EVs that should plugin and move them to charging fleets."""
        for ev in list(self.fleet_discharging):  # Create copy to allow modification
            # Check if it's plugin time and SOC is below charging threshold
```

```
        if (convert_decimal_to_time(self.current_hour - 0.05) < ev["Plugin Time"] <
            convert_decimal_to_time(self.current_hour + 0.05) and
            ev["Current SOC"] < self.config.charging_threshold * ev["Maximum SOC"]):

            # Check if EV should be postponed (would require <6A to meet time)
            required_time = ev["Missing SOC"] / (6 * 230)  # Minimum charging time
            if required_time < ev["Remaining Time"]:
                ev["Plugin Time"] = convert_decimal_to_time(self.current_hour + 0.25)
                ev["Remaining Time"] -= 0.05
                continue

            # Assign plugin order
            ev["Order"] = self.order
            self.order += 1

            # Move to appropriate charging fleet based on phase
            phase = ev.get('Phase')
            if phase == 1:
                self.fleet_charging_phase_1 = np.append(self.fleet_charging_phase_1, ev)
            elif phase == 2:
                self.fleet_charging_phase_2 = np.append(self.fleet_charging_phase_2, ev)
            elif phase == 3:
                self.fleet_charging_phase_3 = np.append(self.fleet_charging_phase_3, ev)
            elif phase == 4:
                self.fleet_charging_tri_phase = np.append(self.fleet_charging_tri_phase, ev)

            # Remove from discharging fleet
            self.fleet_discharging = np.delete(self.fleet_discharging,
                                               np.where(self.fleet_discharging == ev))

        elif ev["Current SOC"] < self.config.charging_threshold * ev["Maximum SOC"]:
            # Decrement remaining time for EVs waiting to charge
            ev["Remaining Time"] -= 0.05

def _get_grid_state(self, step_index: int) -> Tuple[Dict[str, float], Dict[str, float]]:
    """
    Get grid availability and voltages at current time step.

    Args:
        step_index: Index in grid_data

    Returns:
        Tuple of (available_currents, voltages) dictionaries
    """
    row = self.grid_data.iloc[step_index]

    available_currents = {
        'phase_1': row['Phase_1'],
        'phase_2': row['Phase_2'],
        'phase_3': row['Phase_3'],
        'total': row['Total']
    }

    voltages = {
        'phase_1': row['U1'],
        'phase_2': row['U2'],
        'phase_3': row['U3'],
        'tri_phase': ((row['U1'] + row['U2'] + row['U3']) / 3) * 1.732  # Line voltage
    }

    return available_currents, voltages

def _check_and_adjust_availability(
    self,
    phase_1: float,
    phase_2: float,
    phase_3: float,
    tri_phase: float
):
    """
    Check if available current is sufficient and adjust fleets if needed.

    This implements the availability_check logic from the original code,
    moving EVs back to discharging fleet if insufficient current is available.
```

```
        Args:
            phase_1: Available current for phase 1 (A)
            phase_2: Available current for phase 2 (A)
            phase_3: Available current for phase 3 (A)
            tri_phase: Available current for three-phase charging (A)
        """
        safety = self.config.safety_coefficient

        # Check each phase and remove EVs with lowest missing SOC if needed
        self._check_phase_availability(self.fleet_charging_phase_1, phase_1 * safety, 6, 1)
        self._check_phase_availability(self.fleet_charging_phase_2, phase_2 * safety, 6, 2)
        self._check_phase_availability(self.fleet_charging_phase_3, phase_3 * safety, 6, 3)
        self._check_phase_availability(self.fleet_charging_tri_phase, tri_phase * safety, 16, 4)

    def _check_phase_availability(
        self,
        fleet: np.ndarray,
        available: float,
        min_current: float,
        phase: int
    ):
        """
        Check if a specific phase has enough current and adjust if needed.

        Args:
            fleet: Array of EVs on this phase
            available: Available current (A)
            min_current: Minimum current per EV (A)
            phase: Phase number (1-4)
        """
        while available - len(fleet) * min_current < 0 and len(fleet) > 0:
            # Find EV with lowest missing SOC
            missing_socs = np.array([ev["Missing SOC"] for ev in fleet])
            min_idx = np.argmin(missing_socs)
            ev_to_remove = fleet[min_idx]

            # Update plugin time if before plugout time
            if convert_decimal_to_time(self.current_hour + 0.25) < ev_to_remove["Plugout Time"]:
                ev_to_remove["Plugin Time"] = convert_decimal_to_time(self.current_hour + 0.25)

            # Move back to discharging fleet
            self.fleet_discharging = np.append(self.fleet_discharging, ev_to_remove)

            # Remove from charging fleet
            if phase == 1:
                self.fleet_charging_phase_1 = np.delete(fleet, min_idx)
                fleet = self.fleet_charging_phase_1
            elif phase == 2:
                self.fleet_charging_phase_2 = np.delete(fleet, min_idx)
                fleet = self.fleet_charging_phase_2
            elif phase == 3:
                self.fleet_charging_phase_3 = np.delete(fleet, min_idx)
                fleet = self.fleet_charging_phase_3
            elif phase == 4:
                self.fleet_charging_tri_phase = np.delete(fleet, min_idx)
                fleet = self.fleet_charging_tri_phase

    def _charge_all_phases(
        self,
        phase_1: float,
        phase_2: float,
        phase_3: float,
        tri_phase: float,
        voltages: Dict[str, float],
        step_index: int
    ):
        """
        Optimise and charge EVs on all phases.

        Args:
            phase_1: Available current phase 1 (A)
            phase_2: Available current phase 2 (A)
            phase_3: Available current phase 3 (A)
```

```
            tri_phase: Available current three-phase (A)
            voltages: Dictionary of voltages for each phase
            step_index: Current time step index
        """
        # Charge each phase
        current_1 = self._charge_phase(
            self.fleet_charging_phase_1, phase_1, voltages['phase_1'],
            (self.config.lower_bound_single_phase, self.config.upper_bound_single_phase),
            step_index
        )
        self.allocated_amps_phase_1 = np.append(self.allocated_amps_phase_1, current_1)

        current_2 = self._charge_phase(
            self.fleet_charging_phase_2, phase_2, voltages['phase_2'],
            (self.config.lower_bound_single_phase, self.config.upper_bound_single_phase),
            step_index
        )
        self.allocated_amps_phase_2 = np.append(self.allocated_amps_phase_2, current_2)

        current_3 = self._charge_phase(
            self.fleet_charging_phase_3, phase_3, voltages['phase_3'],
            (self.config.lower_bound_single_phase, self.config.upper_bound_single_phase),
            step_index
        )
        self.allocated_amps_phase_3 = np.append(self.allocated_amps_phase_3, current_3)

        current_tri = self._charge_phase(
            self.fleet_charging_tri_phase, tri_phase, voltages['tri_phase'],
            (self.config.lower_bound_three_phase, self.config.upper_bound_three_phase),
            step_index
        )
        self.allocated_amps_tri_phase = np.append(self.allocated_amps_tri_phase, current_tri)

        # Track overall current
        total_current = current_1 + current_2 + current_3 + current_tri
        self.allocated_amps_overall = np.append(self.allocated_amps_overall, total_current)

        # Update maximum tracking
        self.max_overall_current = max(self.max_overall_current, total_current)
        total_fleet_size = (len(self.fleet_charging_phase_1) + len(self.fleet_charging_phase_2)
+
                          len(self.fleet_charging_phase_3) + len(self.fleet_charging_tri_phase))
        self.max_charging_fleet_size = max(self.max_charging_fleet_size, total_fleet_size)

    def _charge_phase(
        self,
        fleet: np.ndarray,
        available_current: float,
        voltage: float,
        bounds: Tuple[float, float],
        step_index: int
    ) -> float:
        """
        Optimise and charge EVs on a single phase.

        Args:
            fleet: Array of EVs on this phase
            available_current: Available current (A)
            voltage: Phase voltage (V)
            bounds: (min_current, max_current) tuple
            step_index: Current time step

        Returns:
            Total allocated current (A)
        """
        if len(fleet) == 0:
            return 0.0

        # Generate bounds for optimisation
        optimisation_bounds = [bounds] * len(fleet)

        # Create constraint
        constraint = {
            'type': 'ineq',
```

```
            'fun': lambda x: available_current * self.config.safety_coefficient - np.sum(x)
        }

        # Optimise currents
        currents = self._optimise_charging_currents(fleet, available_current, voltage,
                                                    optimisation_bounds, constraint)

        # Update EV states
        if currents is not None:
            self._update_ev_states(fleet, currents, voltage)
            return float(np.sum(currents))

        return 0.0

    def _optimise_charging_currents(
        self,
        charging_fleet: np.ndarray,
        available_current: float,
        voltage: float,
        bounds: List[Tuple[float, float]],
        constraint: Dict
    ) -> Optional[np.ndarray]:
        """
        Optimise charging currents for active EVs using scipy.minimise.

        Objective: Minimise sum of |E_missing/(I×V) - T_remaining| for all EVs

        Args:
            charging_fleet: Array of EVs currently charging
            available_current: Available grid current (A)
            voltage: Grid voltage (V)
            bounds: List of (min, max) current bounds for each EV
            constraint: Dictionary defining inequality constraint

        Returns:
            Array of optimised currents for each EV, or None if optimisation fails
        """
        if len(charging_fleet) == 0:
            return None

        # Initial guess (equal distribution)
        x0 = np.array([available_current / len(charging_fleet)] * len(charging_fleet))
        x0 = np.clip(x0, bounds[0][0], bounds[0][1])  # Respect bounds

        # Define objective function
        def objective(currents):
            total = 0.0
            for i, ev in enumerate(charging_fleet):
                if currents[i] > 0:
                    charging_time = ev["Missing SOC"] / (currents[i] * voltage)
                    time_diff = abs(charging_time - ev["Remaining Time"])
                    total += time_diff
            return total

        # Define battery health objective function
        def objective_battery_health(currents):
            """Minimise battery degradation (prefer low currents, low SoC charging)."""
            degradation_cost = 0.0
            for i, ev in enumerate(charging_fleet):
                current = currents[i]
                # Get SoC ratio (0-1)
                current_soc = ev.get('Current SOC', 0)
                max_soc = ev.get('Maximum SOC', 1)
                soc_ratio = current_soc / max_soc if max_soc > 0 else 0

                # Get temperature factor
                temp = ev.get('Temperature', 21)
                temp_factor = 1.0 + max(0, (temp - 25) / 20)

                # Higher SoC = higher degradation
                soc_factor = 1.0 + soc_ratio

                # Degradation proportional to I²
                degradation_cost += (current ** 2) * soc_factor * temp_factor
```

```
            return degradation_cost

        # Define minimum current objective function
        def objective_minimum_current(currents):
            """Use minimum current while still meeting requirements."""
            return np.sum(currents)

        # Define load balancing objective function
        def objective_load_balancing(currents):
            """Balance current evenly across EVs."""
            avg_current = np.mean(currents)
            variance = np.sum((currents - avg_current) ** 2)
            return variance + 0.01 * np.sum(currents)

        # Select objective based on config
        if self.config.objective == ObjectiveType.BATTERY_HEALTH:
            selected_objective = objective_battery_health
        elif self.config.objective == ObjectiveType.MINIMUM_CURRENT:
            selected_objective = objective_minimum_current
        elif self.config.objective == ObjectiveType.LOAD_BALANCING:
            selected_objective = objective_load_balancing
        else:  # TIME_COMPLETION (default)
            selected_objective = objective

        # Run optimisation
        try:
            result = minimise(
                selected_objective,
                x0,
                method='SLSQP',
                bounds=bounds,
                constraints=[constraint],
                options={'maxiter': 5000, 'ftol': 1e-4}
            )

            if result.success:
                return result.x
            else:
                # If optimisation fails, return equal distribution
                return x0

        except Exception as e:
            print(f"Optimisation warning: {e}")
            return x0

    def _update_ev_states(
        self,
        charging_fleet: np.ndarray,
        currents: np.ndarray,
        voltage: float
    ):
        """
        Update EV states (SoC, temperature, degradation) after charging.

        Args:
            charging_fleet: Array of EVs being charged
            currents: Charging currents for each EV
            voltage: Grid voltage
        """
        from models.battery import calculate_battery_temperature

        for i, ev in enumerate(charging_fleet):
            current = currents[i]

            # Update remaining time
            ev["Remaining Time"] -= self.config.time_step

            # Update SoC if below desired level
            if ev["Current SOC"] < ev["Desired SOC"]:
                # Energy added in this time step (Wh)
                energy_added = voltage * current * self.config.time_step
                ev["Current SOC"] += energy_added
                ev["Missing SOC"] = ev["Desired SOC"] - ev["Current SOC"]
```

```
            # Update temperature using standalone function
            ev["Temperature"] = calculate_battery_temperature(
                charging_current=current,
                ev=ev,
                battery_temperature=ev["Temperature"],
                air_temperature=21.0,
                time_step_minutes=self.config.time_step * 60
            )

            # Store current for this time step (will be logged in _finalize_current_logs)
            ev["_pending_current"] = current

    def _process_plugout_events(self):
        """Check for EVs that should plugout and move them back to discharging fleet."""
        # Process each phase fleet separately to avoid reference issues

        # Phase 1
        for ev in list(self.fleet_charging_phase_1):
            if (convert_decimal_to_time(self.current_hour - 0.05) < ev["Plugout Time"] <
                convert_decimal_to_time(self.current_hour + 0.05) or
                ev["Current SOC"] >= ev["Desired SOC"]):
                ev['Temperature'] = 21
                self.fleet_discharging = np.append(self.fleet_discharging, ev)
                self.fleet_charging_phase_1 = np.delete(
                    self.fleet_charging_phase_1,
                    np.where(self.fleet_charging_phase_1 == ev))

        # Phase 2
        for ev in list(self.fleet_charging_phase_2):
            if (convert_decimal_to_time(self.current_hour - 0.05) < ev["Plugout Time"] <
                convert_decimal_to_time(self.current_hour + 0.05) or
                ev["Current SOC"] >= ev["Desired SOC"]):
                ev['Temperature'] = 21
                self.fleet_discharging = np.append(self.fleet_discharging, ev)
                self.fleet_charging_phase_2 = np.delete(
                    self.fleet_charging_phase_2,
                    np.where(self.fleet_charging_phase_2 == ev))

        # Phase 3
        for ev in list(self.fleet_charging_phase_3):
            if (convert_decimal_to_time(self.current_hour - 0.05) < ev["Plugout Time"] <
                convert_decimal_to_time(self.current_hour + 0.05) or
                ev["Current SOC"] >= ev["Desired SOC"]):
                ev['Temperature'] = 21
                self.fleet_discharging = np.append(self.fleet_discharging, ev)
                self.fleet_charging_phase_3 = np.delete(
                    self.fleet_charging_phase_3,
                    np.where(self.fleet_charging_phase_3 == ev))

        # Three-phase
        for ev in list(self.fleet_charging_tri_phase):
            if (convert_decimal_to_time(self.current_hour - 0.05) < ev["Plugout Time"] <
                convert_decimal_to_time(self.current_hour + 0.05) or
                ev["Current SOC"] >= ev["Desired SOC"]):
                ev['Temperature'] = 21
                self.fleet_discharging = np.append(self.fleet_discharging, ev)
                self.fleet_charging_tri_phase = np.delete(
                    self.fleet_charging_tri_phase,
                    np.where(self.fleet_charging_tri_phase == ev))

    def _process_daily_discharge(self):
        """
        Process daily discharge at configured time.

        This simulates EVs returning home after daily driving. Each EV:
        1. Loses energy based on their daily driving consumption
        2. Gets new randomized plugin/plugout times for tonight
        3. Gets new randomized daily discharge for tomorrow
        4. Gets new desired SoC based on forecasted driving needs

        This creates realistic day-to-day variation in charging patterns.
        """
        current_time_str = convert_decimal_to_time(self.current_hour)
```

```
        # Check if within 3-minute window of discharge time (18:00 to 18:03)
        # Using string comparison - discharge_time is "18:00", window ends at "18:03"
        discharge_start = self.config.discharge_time  # "18:00"
        discharge_end = self.config.discharge_time[:3] + "03"  # "18:03"

        if discharge_start < current_time_str <= discharge_end:
            from models.ev_fleet import (
                generate_daily_discharge,
                generate_plugin_time,
                generate_plugout_time,
                calculate_desired_soc
            )
            from utils.time_utils import decimal_to_time_string

            for ev in self.fleet_discharging:
                # Process discharge - EV uses energy during the day
                if ev["Current SOC"] - ev["Daily Discharge"] > 0:
                    ev["Current SOC"] = ev["Current SOC"] - ev["Daily Discharge"]
                    ev["Distance Traveled"] += (ev["Daily Discharge"] * 4.8) / 1000  # km
                else:
                    ev["Current SOC"] = 0

                # Update parameters for next day
                ev["Daily Discharge"] = generate_daily_discharge()

                # Generate new plugin/plugout times (randomized each day)
                plugin_decimal = generate_plugin_time()
                plugout_decimal = generate_plugout_time()

                # Calculate remaining time
                remaining_time = plugout_decimal - plugin_decimal
                if remaining_time < 0:
                    remaining_time += 24

                # Update schedule attributes
                ev["Plugin Time"] = decimal_to_time_string(plugin_decimal)
                ev["Plugout Time"] = decimal_to_time_string(plugout_decimal)
                ev["Remaining Time"] = remaining_time

                # Calculate new desired SoC based on forecasted driving needs
                # This creates day-to-day variation in how much each EV wants to charge
                ev["Desired SOC"] = calculate_desired_soc(
                    ev["Current SOC"],
                    ev["Maximum SOC"]
                )

                # Update missing SOC based on new desired level
                ev["Missing SOC"] = ev["Desired SOC"] - ev["Current SOC"]

                # Reset temperature to ambient
                ev["Temperature"] = 21.0

    def _compute_results(self) -> Dict:
        """
        Compute final simulation results.

        Returns:
            Dictionary with comprehensive simulation results
        """
        # Combine all fleet arrays
        all_evs = np.concatenate([
            self.fleet_charging_phase_1,
            self.fleet_charging_phase_2,
            self.fleet_charging_phase_3,
            self.fleet_charging_tri_phase,
            self.fleet_discharging
        ])

        # Calculate total consumption and mileage
        total_consumption = 0.0
        total_mileage = 0.0

        for ev in all_evs:
            # Calculate power based on phase
```

```
            if ev['Phase'] != 4:
                ev_power = np.sum(ev["Current Log"] * 230)  # Single phase
            else:
                ev_power = np.sum(ev["Current Log"] * 400)  # Three phase

            total_consumption += ev_power
            total_mileage += ev["Distance Traveled"]

        # Compile results
        self.results = {
            'total_consumption': total_consumption,  # Wh
            'total_mileage': total_mileage,  # km
            'peak_demand': self.max_overall_current,  # A
            'max_charging_fleet_size': self.max_charging_fleet_size,
            'max_phase_1_size': self.max_fleet_phase_1_size,
            'max_phase_2_size': self.max_fleet_phase_2_size,
            'max_phase_3_size': self.max_fleet_phase_3_size,
            'max_tri_phase_size': self.max_fleet_tri_phase_size,
            'allocated_currents': {
                'phase_1': self.allocated_amps_phase_1,
                'phase_2': self.allocated_amps_phase_2,
                'phase_3': self.allocated_amps_phase_3,
                'tri_phase': self.allocated_amps_tri_phase,
                'overall': self.allocated_amps_overall
            },
            'fleet_state': all_evs
        }

        return self.results

    def export_results(self, output_dir: str):
        """
        Export simulation results to files.

        Args:
            output_dir: Directory for output files
        """
        import os
        os.makedirs(output_dir, exist_ok=True)

        # Export summary
        summary_path = os.path.join(output_dir, 'simulation_summary.txt')
        with open(summary_path, 'w') as f:
            f.write("=== Optimised Charging Simulation Results ===\n\n")
            f.write(f"Total Consumption: {self.results['total_consumption']:.2f} Wh\n")
            f.write(f"Total Mileage: {self.results['total_mileage']:.2f} km\n")
            f.write(f"Peak Demand: {self.results['peak_demand']:.2f} A\n")
            f.write(f"Max Charging Fleet Size: {self.results['max_charging_fleet_size']}\n")
            f.write(f"  Phase 1: {self.results['max_phase_1_size']}\n")
            f.write(f"  Phase 2: {self.results['max_phase_2_size']}\n")
            f.write(f"  Phase 3: {self.results['max_phase_3_size']}\n")
            f.write(f"  Three-Phase: {self.results['max_tri_phase_size']}\n")

        # Export current profiles
        currents_df = pd.DataFrame({
            'Phase_1': self.results['allocated_currents']['phase_1'],
            'Phase_2': self.results['allocated_currents']['phase_2'],
            'Phase_3': self.results['allocated_currents']['phase_3'],
            'Three_Phase': self.results['allocated_currents']['tri_phase'],
            'Overall': self.results['allocated_currents']['overall']
        })
        currents_path = os.path.join(output_dir, 'current_profiles.csv')
        currents_df.to_csv(currents_path, index=False)

        print(f"Results exported to {output_dir}")


def run_optimised_simulation(
    fleet: List[Dict],
    data_file: str,
    output_dir: Optional[str] = None
) -> Dict:
    """
    Convenience function to run optimised charging simulation.
```

```
    Args:
        fleet: List of EV dictionaries
        data_file: Path to grid data CSV
        output_dir: Optional directory for results export

    Returns:
        Dictionary of simulation results

    Example:
        >>> from models.ev_fleet import EVFleet
        >>> fleet = EVFleet(10)
        >>> evs = fleet.get_ev_dicts()
        >>> results = run_optimised_simulation(evs, "Time_Step_3_Long.csv")
        >>> print(f"Peak: {results['peak_demand']:.1f} A")
    """
    sim = OptimisedChargingSimulation(fleet, data_file)
    results = sim.run()

    if output_dir:
        sim.export_results(output_dir)

    return results
```

Reference and Technical Notes

Battery Degradation Model: The cycle-based degradation model follows Naumann, M., Schimpe, M., Keil, P., Hesse, H.C., Jossen, A. (2018). Analysis and modelling of calendar aging of a commercial LiFePO4/graphite cell. Journal of Energy Storage.
Charging Standards: Current bounds comply with IEC 61851-1:2017 (Electric vehicle conductive charging system) which specifies minimum charging current of 6A for Mode 3 charging.
Grid Standards: Feeder sizing follows IEC 60364-5-52 (Electrical installations of buildings - Selection and erection of electrical equipment - Wiring systems).
Optimisation Algorithm: The SLSQP (Sequential Least Squares Programming) algorithm from scipy.optimise is used for constrained nonlinear optimisation, suitable for problems with inequality constraints.

Software Dependencies

- Python 3.9+
- NumPy (numerical computing)
- SciPy (optimisation algorithms)
- Pandas (data manipulation)
- Matplotlib (visualisation)
- CVXPY (convex optimisation, optional)

Repository

The complete source code, including tests and documentation, is available at:
https://github.com/SaCarvalhosa/ev_charging_optimisation/tree/main