Scalable Aggregation of Demand Side Flexibilities in Power Systems

A thesis approved for the academic degree of

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

at the

Faculty of Electrical Engineering and Information Technology

TU Dortmund University

by

Emrah Öztürk, M.Sc.

Supervisor:	Prof. Dr.-Ing. Timm Faulwasser, Hamburg University of Technology
Co-Advisor:	Prof. Stefan Palzer, PhD, TU Dortmund University

Day of Oral Examination: 16.01.2026

Acknowledgments

This thesis is the culmination of my doctoral studies, which I have completed as an external student at the Department of Electrical Engineering and Information Technology at TU Dortmund University, Germany. During my dissertation, I was employed as a doctoral researcher at Vorarlberg University of Applied Sciences, Austria. I would like to take this opportunity to thank everyone whose support made this work possible.

I would like to express my gratitude to Dr. Klaus Rheinberger at Vorarlberg University of Applied Sciences for his valuable guidance and advice. His expertise and mentorship have greatly contributed to my growth and development in various fields including computer science, mathematics, physics, modeling, and teaching. Dr. Rheinberger's willingness to assist and guide me in the right direction has been instrumental in the success of this research.

Furthermore, I would like to express my sincere gratitude to Prof. Dr.-Ing. Timm Faulwasser, my PhD supervisor at Hamburg University of Technology (formerly working at TU Dortmund University), for accepting me into his team. His expertise in the fields of energy engineering, and systems and control has greatly enriched my learning process during my doctoral studies. I enjoyed our discussions and am grateful for the opportunity to write my doctoral thesis under his guidance.

I would also like to thank Prof. (FH) Dr.-Ing. Markus Preißinger from Vorarlberg University of Applied Sciences for granting me a high degree of scientific freedom and for sharing his writing expertise; Prof. Dr. Karl Worthmann from Technische Universität Ilmenau for his support in mathematics; Kevin Chris Kaspar, M.Sc., from Vorarlberg University of Applied Sciences for our collaborative work; Prof. (FH) Dr. Peter Kepplinger from Vorarlberg University of Applied Sciences for sharing his knowledge in mathematics and engineering; Prof. Stefan Palzer, PhD, for reviewing this thesis; and Prof. Dr.-Ing. Stephan Frei as well as Prof. Dr.-Ing. Prof. h. c. Dr. h. c. Torsten Betram for serving on the examination committee.

In addition, during my doctoral studies, I participated in and received support from the following projects:

- Industry of the Future – Data-driven, Energy-efficient, Sustainable (funded by the European Regional Development Fund (ERDF) and the State of Vorarlberg as part of the EU program "Investment for Growth and Jobs in Austria 2014-2020").

- EBusCharge – Flexible E-Bus Fleet Charge Management [FFG, No. 899915] (funded by the Climate and Energy Fund).
- Josef Ressel Centre for Intelligent Thermal Energy Systems (funded by the Federal Ministry for Digitization and Business Location and the National Foundation for Research, Technology and Development).
- Hub4FlECs – Scientific Hub for Flexible Energy Communities [FFG, No. 898053] (funded by the Austrian Research Promotion Agency FFG).

Finally, I am grateful for the love and encouragement from my family and friends, especially my mother and my late father, who inspired me to pursue my PhD. Despite his struggle with cancer, he never stopped encouraging me to continue. His pride and joy in sharing my educational journey with others allowed me to continue. I will forever hold his memory close to my heart and feel blessed to have had such a caring and supportive father.

Feldkirch, February 2026
Emrah Öztürk

Abstract

The increasing integration of renewable energy sources such as solar and wind energy coupled with the rising demand for electricity poses a series of new challenges for modern power grids. Maintaining a balance between fluctuating energy production and rising demand is becoming increasingly difficult. At the same time, flexible devices such as battery energy storage systems, thermostatically controlled loads, and electric vehicles are being integrated into the system on an increasing scale. Used individually, a single device may not offer significant flexibility. However, when combined with multiple devices, this combination can act as a virtual power plant and help to stabilize grid operations.

In response to the need for methods that enable large-scale control, aggregation is introduced in the literature as a viable solution. Aggregators can leverage the flexibility offered by various devices to participate in energy markets. In this setting, the aggregator is the only entity that knows the system specifics and has authority over contracted consumers' devices. This fosters data protection and helps to prevent unauthorized access by third parties.

A key challenge in aggregation-based control is, however, the precise quantification of the aggregated flexibility. The flexibility of a device can be represented by a set of its feasible power profiles, and the aggregated flexibility of multiple devices by the addition of these individual flexibility sets. However, the set addition (Minkowski sum) is known to be computationally prohibitive. As a result, approximation methods are an active field of research.

Approximation strategies can be roughly categorized into inner and outer approximations. Inner approximations are conservative, i.e., the feasible region is usually reduced, leaving out potential power profiles. Outer approximations, on the other hand, overestimate the potential feasible region and may contain infeasible power profiles. Auxiliary service providers must ensure the fulfillment of the power profiles to avoid contractual penalties, which favors inner approximations. However, as this thesis shows, state-of-the-art inner approximations suffer from objective-dependent performance, increased computational complexity, and may exclude nominal power profiles, such as the idle state.

On this canvas, this thesis examines the concept of aggregation-based control and its potential for providing ancillary services. First, common flexible devices in power grids are identified and mapped to a convex storage model. The relationship between flexibility sets

and their aggregation is analyzed, along with the associated challenges. The significance of utilizing approximation techniques is emphasized, accompanied by a review of the existing literature. Additionally, an open-source benchmark is developed, and state-of-the-art techniques are compared to identify the limitations of current methods.

Based on these observations, a vertex-based inner approximation for suitable polytopes is proposed to overcome the deficiencies of existing methods. This method is then extended to battery energy storage systems and tested against ten state-of-the-art inner approximation methods to demonstrate its superior performance in terms of computational complexity and accuracy across various objectives.

Furthermore, the proposed approach is extended to convex storage models that can be used to represent a variety of practical devices. Numerical examples with different types of flexibility and hierarchical aggregation settings are conducted to validate the extended approach. It is shown that the aggregated flexibility of multiple devices can be effectively leveraged to lower the peak demand in residential areas. Finally, an efficient disaggregation strategy is proposed that does not require any optimization. These strategies are then implemented in an open-source Python package, with example codes provided to showcase the software tool's functionality.

Ultimately, the results of this thesis support researchers and industry professionals by offering algorithms that can effectively (dis-)aggregate the flexibility of multiple devices in practical scenarios with accuracy and efficiency.

Deutsche Kurzfassung

Die zunehmende Integration von erneuerbaren Energien wie Photovoltaik und Windenergie in Verbindung mit der wachsenden Nachfrage nach Strom stellt moderne Stromnetze vor wachsende Herausforderungen. Das Gleichgewicht zwischen schwankender Energieerzeugung und steigender Nachfrage zu halten, wird immer schwieriger. Gleichzeitig werden flexible Geräte wie Batteriespeicher, thermostat gesteuerte Lasten und Elektrofahrzeuge in großem Umfang ins System integriert. Ein einzelnes Gerät mag nicht viel Flexibilität bieten, aber wenn mehrere Geräte kombiniert werden, kann ein virtuelles Kraftwerk entstehen, das dabei hilft, den Netzbetrieb zu stabilisieren.

Um einen sicheren und effizienten Netzbetrieb zu gewährleisten, wurde in der Literatur Aggregation als praktikable Lösung vorgestellt. Aggregation erlaubt die Flexibilität verschiedener Geräte zu nutzen, um an Energiemärkten teilzunehmen. Der Aggregator ist in diesem Szenario die einzige Entität, die die Systemdetails kennt und die Betriebsautorität über die Geräte der Vertragsverbraucher besitzt. Dadurch wird der Datenschutz gegenüber Dritten gewährleistet und unbefugter Zugriff verhindert.

Eine Herausforderung bei der aggregationsbasierten Steuerung besteht darin, die aggregierte Flexibilität genau zu quantifizieren. Die Flexibilität eines Geräts kann durch die Menge seiner potenziellen Lastprofile dargestellt werden, und die aggregierte Flexibilität mehrerer Geräte durch die Addition dieser individuellen Flexibilitätsmengen. Die Mengenaddition, auch Minkowski-Summe genannt, ist jedoch im Allgemeinen nicht effizient berechenbar, weshalb in der Literatur verschiedene Approximationsmethoden vorgeschlagen wurden.

Grob lassen sich zwei Arten von Approximationsstrategien unterscheiden: innere und äußere. Innere Approximationen sind konservativ und reduzieren den zulässigen Bereich, während äußere Approximationen den Bereich überschätzen und nicht zulässige Profile enthalten können. Systemdienstleister bevorzugen in der Regel innere Approximationen, um vertragliche Strafen zu vermeiden. Wie diese Arbeit jedoch zeigt, leiden moderne innere Approximationen unter einer zielabhängigen Leistung, erhöhter Rechenkomplexität und können nominale Leistungsprofile, wie z.B. den Zustand des Nichtstuns, ausschließen.

In diesem Kontext untersucht diese Dissertation das Konzept der aggregationsbasierten Betriebsführung und ihr Potenzial für die Bereitstellung von Regel- und Zusatzdiensten. Zu

Beginn werden flexible Geräte in Stromnetzen identifiziert und einem konvexen Speichermodell zugeordnet. In weiterer Folge wird die Beziehung zwischen Flexibilitätsmengen und ihrer aggregierten Flexibilität analysiert sowie die damit verbundenen Herausforderungen untersucht. Die Bedeutung der Nutzung von Näherungsverfahren wird hervorgehoben, begleitet von einem Überblick über die bestehende Literatur. Darüber hinaus wird ein Open-Source-Benchmark für Näherungsverfahren entwickelt, um die Einschränkungen der aktuellen Methoden zu identifizieren.

Auf der Grundlage dieser Beobachtungen wird eine eckpunktbasierte innere Approximation für Polytope vorgeschlagen um so die Schwächen bestehender Methoden zu überwinden. Diese Methode wird dann auf Batteriespeichersysteme erweitert und mit zehn modernen inneren Approximationstechniken verglichen, um ihre Leistungsfähigkeit in Bezug auf Rechenkomplexität und Genauigkeit für verschiedene Szenarien zu demonstrieren.

Der vorgeschlagene Ansatz wird weiterhin auf konvexe Speicher-Modelle erweitert, die zur Darstellung einer Vielzahl realer Systemkomponenten verwendet werden können. Numerische Beispiele mit unterschiedlichen Arten von Flexibilität validieren den erweiterten Ansatz. Es wird gezeigt, dass die aggregierte Flexibilität mehrerer Geräte genutzt werden kann, um die Spitzenlast in Wohngebieten zu senken. Schließlich wird eine effiziente Disaggregationsstrategie vorgeschlagen, die keine online Optimierung erfordert. Die entwickelten numerischen Verfahren sind in ein Open-Source-Python toolbox implementiert, wobei Beispielcodes bereitgestellt werden, um die Funktionalität des Softwaretools zu demonstrieren.

Das maßgebliche Anliegen dieser Dissertation ist es, Forscher und Fachkräfte aus der Industrie zu unterstützen, indem Algorithmen bereitgestellt werden, die es ermöglichen, die Flexibilität mehrerer Geräte in realen Situationen genau und effizient zu aggregieren und zu disaggregieren.

Contents

Acknowledgments **III**

Abstract **V**

Deutsche Kurzfassung **VII**

Contents **IX**

Index of Notation **XIII**

List of Figures **XIX**

List of Tables **XX**

1 Introduction **1**

1.1 Flexibility in Power Grids 1
1.2 Aggregation of Demand-Side-Flexibilities 3
1.3 Challenges 5
1.4 Outline and Contributions 6

2 Problem Statement and Modelling **9**

2.1 Flexible Devices 9
2.1.1 Battery Energy Storage Systems 9
2.1.2 Thermostatically Controlled Loads 11
2.1.3 Electric Vehicles 12
2.1.4 Pumped Hydro Energy Storages 13
2.2 Convex Storage Models 15
2.3 Flexibility Aggregation 18
2.3.1 Polytope Basics 18
2.3.2 Minkowski Summation 19
2.4 Summary 24

3 Approximation Strategies **25**
3.1 Overview of Approximation Strategies . . . 25
3.1.1 Outer Approximation by Right-Hand Summation . . . 26
3.1.2 Inner Approximation by Zonotopes . . . 27
3.1.3 Inner Approximation by Cuboid Homothets . . . 29
3.1.4 Inner and Outer Approximation by Battery Homothets . . . 32
3.1.5 Inner Approximation by Battery Homothet Projection . . . 34
3.1.6 Inner Approximation by Ellipsoid Projection . . . 36
3.1.7 Inner Approximation by Ellipsoid Projection with Linear Decision Rule . . . 38
3.1.8 Further Approximation Strategies . . . 39
3.2 Comparative Benchmarking . . . 40
3.2.1 Overview of Communication and Computation Effort . . . 41
3.2.2 Benchmark Formulation . . . 43
3.2.3 Benchmark Results . . . 46
3.3 Summary . . . 49

4 Scalable Aggregation and Dissaggregation **51**
4.1 Aggregation Strategy for Special Polytopes . . . 51
4.1.1 Overview of Aggregation Strategy . . . 52
4.1.2 Main Mathematical Results . . . 55
4.2 Extension to Battery Energy Storage Systems . . . 60
4.2.1 Aggregation Strategy for Battery Energy Storage Systems . . . 61
4.2.2 Benchmarking . . . 66
4.3 Extension to Convex Storage Models . . . 70
4.3.1 Aggregation Strategy for Convex Storage Models . . . 70
4.3.2 Use Case – One Aggregator and One Flexibility Type . . . 74
4.3.3 Use Case – Multiple Aggregators and Multiple Flexibility Types . . . 76
4.3.4 Scalability . . . 78
4.4 Disaggregation Strategy for Convex Storage Models . . . 80
4.5 Summary . . . 83

5 Software Implementation **85**
5.1 Aggregation and Disaggregation Tool PyFlexAD . . . 85
5.1.1 Software Structure . . . 85
5.1.2 Software Architecture . . . 88
5.2 Illustrative Code Examples . . . 90

5.2.1 Simple Aggregation 90
5.2.2 Hierarchical Aggregation 92
5.3 Summary 94

6 Extensions towards Non-convexity and Uncertainty 95
6.1 Aggregation Strategies for Non-convex Sets 95
6.1.1 Introduction 95
6.1.2 Related Literature 98
6.1.3 Extension 99
6.2 Aggregation Strategies under Uncertain Data 101
6.2.1 Introduction 101
6.2.2 Related Literature 102
6.2.3 Extension 103
6.3 Summary 104

7 Conclusions and Perspectives 105
7.1 Grid-Aware Aggregation Strategies 105
7.1.1 Introduction 105
7.1.2 Related Literature 107
7.2 Concluding Remarks 108

Bibliography 111

Appendix 119

A Illustrative Code for Simple Aggregation 119

B Illustrative Code for Hierarchical Aggregation 121

Index of Notation

Abbreviations and Acronyms

This list serves as a reference for abbreviations and acronyms.

BESS	Battery energy storage system
CP	Convex program
DR	Demand response
DSO	Distribution system operator
EV	Electric vehicle
HVAC	Heating, ventilation, and air conditioning
IA	Inner approximation
IER	Imbalance energy ratio
LDR	Linear decision rule
LP	Linear program
LTI	Linear Time-Invariant
MIE	Minimum imbalance energy
MILP	Mixed Integer Linear Programming
MPC	Model Predictive Control
OA	Outer approximation
OP	Operating point
PHES	Pumped hydro energy storage
RHS	Right-hand side
SDP	Semidefinite program
TCL	Thermostatically controlled load
TSO	Transmission system operator
UPR	Unused potential ratio
V2G	Vehicle-to-Grid

Mathematical Notation

The following symbols are used throughout the thesis.

d	number of time periods $d \in \mathbb{N}$
n	number of devices $d \in \mathbb{N}$
Δt	length of time interval $\Delta t \in \mathbb{R}$
$\lvert\mathcal{X}\rvert$	cardinality of a set $\mathcal{X}$
$\oplus_{i=1}^{n}\mathcal{X}_i$	Minkowski sum, $\oplus_{i=1}^{n}\mathcal{X}_i = \left\{x \in \mathbb{R}^d : x = \sum_{i=1}^{n} x_i, x_i \in \mathcal{X}_i\right\}$
$\mathrm{Conv}(\mathcal{X})$	convex hull of $\mathcal{X}$
$\lVert x\rVert_2$	ℓ_2 norm, $\lVert x\rVert_2 = \left(\sum_{i=1}^{d} x_i^2\right)^{\frac{1}{2}}$
$\lVert x\rVert_1$	ℓ_1 norm, $\lVert x\rVert_1 = \sum_{i=1}^{d} \lvert x_i\rvert$
$\lVert x\rVert_\infty$	ℓ_∞ norm, $\lVert x\rVert_\infty = \max_{i=1,\dots,d} \lvert x_i\rvert$
$\mathbf{0}_d$	vector of zeros $\mathbf{0}_d \in \mathbb{R}^d$
$\mathbf{1}_d$	vector of ones $\mathbf{1}_d \in \mathbb{R}^d$
$\mathrm{Proj}^t(x)$	projection in t-dimensional space, $\mathrm{Proj}^t(x) = (x_1, \dots, x_t, \mathbf{0}_{d-t})^\top$
$x_{[t]}$	vector consisting of the first t components of $x \in \mathbb{R}^d$, $x_{[t]} \in \mathbb{R}^t$
S	vector of stored energy $S \in \mathbb{R}^d$
$\overline{S}$	vector of upper energy limits $\overline{S} \in \mathbb{R}^d$
$\underline{S}$	vector of lower energy limits $\underline{S} \in \mathbb{R}^d$
S_{init}	initial energy $S_{\text{init}} \in \mathbb{R}$
S_f	minimum final energy $S_f \in \mathbb{R}$
α	self discharge factor $\alpha \in \mathbb{R}$
x	vector of power (in-)output $x \in \mathbb{R}^d$
$\overline{x}$	vector of upper power limits $\overline{x} \in \mathbb{R}^d$
$\underline{x}$	vector of lower power limits $\underline{x} \in \mathbb{R}^d$
η_C	charging efficiency $\eta_C \in \mathbb{R}$
η_D	discharging efficiency $\eta_D \in \mathbb{R}$
$\mathcal{B}(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}})$	convex storage model
$\mathcal{B}(S_{\text{init}}, S_{\text{f}}, p)$	BESS model
$\mathcal{B}\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}, \eta_C, \eta_D\right)$	storage with (dis-)charge efficiencies
$\mathcal{B}^C\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}, \eta_C, \eta_D\right)$	storage with (dis-)charge efficiencies (only charging)
$\mathcal{B}^D\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}, \eta_C, \eta_D\right)$	storage with (dis-)charge efficiencies (only discharging)
$\mathcal{X}\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}, \eta_C, \eta_D\right)$	non-convex storage model
$\mathcal{P}$	Polytope $\mathcal{P} \subseteq \mathbb{R}^d$

A	Polytope matrix $A \in \mathbb{R}^{m \times d}$
b	Polytope right-hand vector $b \in \mathbb{R}^m$
j	extreme direction $j \in \{-1, 1\}^d$
$\mathcal{J}$	set of extreme directions
y^j	extreme action $y^j \in \mathbb{R}^d$ to extreme direction j
Y	matrix of extreme actions $Y \in \mathbb{R}^{d \times \lvert\mathcal{J}\rvert}$
v^j	summed extreme action $v^j \in \mathbb{R}^d$
V	matrix of summed extreme actions $V \in \mathbb{R}^{d \times \lvert\mathcal{J}\rvert}$
$\odot$	element-wise multiplication operator for vectors
$\binom{n}{k}$	binomial coefficient, $\binom{n}{k} = \frac{n!}{(n-k)!k!}$
$G \succ 0$	positive definite Matrix
$\mathcal{Z}(G, v, \bar{\lambda})$	Zonotope, $\mathcal{Z}(G, v, \bar{\lambda}) = \{x \in \mathbb{R}^d : x = v + G\lambda, -\bar{\lambda} \leq \lambda \leq \bar{\lambda}\}$
$\beta\mathcal{P} + t$	Homothet of $\mathcal{P}$, $\beta\mathcal{P} + t = \{x \in \mathbb{R}^d : x = \beta\zeta + t,\ \zeta \in \mathcal{P}\}$
$\emptyset$	empty set

List of Figures

1.1 Sketch of a system featuring flexible devices and an aggregator. The flexibilities provided by the devices are aggregated and forwarded to purchasers. Specific information is encapsulated and remains unknown to third parties. 4

2.1 Polytopes $\mathcal{P}_1$ and $\mathcal{P}_2$ in dashed green and dashed-dotted black, and their Minkowski sum $\mathcal{M}$ in solid orange. The vertices are shown as red crosses. . . . 21
2.2 Number of vertices with increasing time periods: The orange line represents the average number of vertices in 50 polytopes, while the blue line shows the number of vertices in the Minkowski sum of these 50 polytopes. 22

3.1 Illustration of the preconditioning process. The polytope is represented in violet. The half-planes of the polytope are shown on the left before preconditioning and on the right after preconditioning. 26
3.2 Illustration of zonotope-based approximations. The top left shows the objective with the ℓ_∞ norm, the top right with the ℓ_1 norm, the bottom left with the ℓ_2 norm, and the bottom right illustrates the weighted approach. 29
3.3 Inner approximation with cuboid homothets. Solid line: feasible region of a storage model, turquoise area: stage zero solution, bright red areas: stage one solutions. 31
3.4 Illustration of the approximation strategies based on battery homothets. The inner approximation is shown in dashed-dotted blue, while the outer approximation is depicted in dashed green. 33
3.5 Illustration of the homothet projection strategy. The approximation is shown in violet, while the Minkowski sum of storage models is represented in solid black. 36
3.6 Illustration of the maximal volume ellipsoid approximation methods. The projected ellipsoid is shown in dashed blue, the improved version with the LDR formulation in dashed-dotted green, and the Minkowski sum of polytopes in solid black. 39

3.7 Overview of the benchmark: The aggregator calculates the aggregated flexibility that is sent to the grid operator. The grid operator calculates the optimal value based on predefined objectives. A centralized controller solves the same problem and a separate evaluation is performed without taking flexibility into account. The inputs for the benchmark consist of battery parameters, the number of time periods and households, demand curves, and day-ahead prices, while the output are the UPR and IER values. 45
3.8 Results for experiments with tuples $(n, d) \in \{20\} \times \{4, 8, 12, 16, 20, 24\}$ in the first column and experiments with tuples $(n, d) \in \{2, 6, 10, 20, 30, 40, 50\} \times \{12\}$ in the second column. The cost UPR values are shown in the first row, the peak UPR values in the second row, and the calculation times in the third row. 47
3.9 Results for experiments with tuples $(n, d) \in \{20\} \times \{4, 8, 12, 16, 20, 24\}$ in the first column and experiments with tuples $(n, d) \in \{2, 6, 10, 20, 30, 40, 50\} \times \{12\}$ in the second column. The cost IER values are shown in the first row, the peak IER values in the second row, and the calculation times in the third row. 48

4.1 Illustration of Assumption 2. While the polytope on the left satisfies the assumption, the polytope on the right does not. 52
4.2 Tree structure for extreme action calculation: left branch with -1, right branch with 1, and increasing time periods indicated by the arrow pointing downwards. 53
4.3 Left: vectors $y_1^{(-1,1)}$, $y_2^{(-1,1)}$ within the polytopes shown in dashed blue and solid green, and the sum $v^{(-1,1)}$ shown in the Minkowski sum $\mathcal{M}$ in dash-dotted black. Right: all possible vectors v^j, $j \in \{-1, 1\}^2$ in the Minkowski sum with the resulting set $\mathcal{A}$ in light-green. 54
4.4 The set $\mathcal{B}(S_{\text{init}}, S_f, p)$ shown in solid green and the set $\mathcal{B}(S_{\text{init}}, S_{\min}, p)$ in dashed blue. The crosses in $\mathcal{B}(S_{\text{init}}, S_{\min}, p)$ show the extreme actions, and the arrow with the dot visualizes the correction process. 63
4.5 UPR values and computation times for tuples $(n, d) \in \{100\} \times \{24, 48, 72, 96\}$ with increasing number of vectors in $\mathcal{J}$. 67
4.6 Results for experiments with tuples $(n, d) \in \{30\} \times \{4, 8, 12, 16, 24\}$ in the first column and experiments with tuples $(n, d) \in \{2, 6, 10, 20, 30\} \times \{24\}$ in the second column. The cost UPR values are shown in the first row, the peak UPR values in the second row, and the calculation times in the third row. Figure reproduced (Öztürk, Faulwasser, et al., 2024). 69

4.7 Example cases, in which the energy constraints are violated. Left: the case in which the energy in the storage is lower than the lower limit, while right: the energy in the storage is higher than the upper limit. Figure reproduced (Öztürk, Kaspar, et al., 2024). 71
4.8 Visualization of EV data. Top: Available EVs. Bottom: Total EV trip consumption (orange) and total uncontrolled EV charging (gray). 74
4.9 Residential loads: the base load is shown in blue, the aggregated controlled load in orange, and the central controlled load in green. 75
4.10 Scenario with three residential areas and three local aggregators. The residents may own EVs, BESSs, and TCLs (air conditioning). The collective local flexibility is passed to a global aggregator, which in turn aggregates and forwards them to a grid operator for ancillary services. 76
4.11 Residential loads: the base load is shown in blue, the aggregate controlled load in orange, and the central controlled load in green. 78
4.12 Comparison of modified UPR ($\mathrm{UPR_M}$) and CPU time across settings $(n, d) \in \{50, 100, 250, 500, 750, 1000\} \times \{24, 48, 72, 96\}$. The results for the central control approach are shown in red, while those for the proposed method are illustrated in blue. 79
4.13 Typical configuration for a hierarchical aggregation setting, where each aggregator is associated with a matrix that represents the summed extreme actions. At the bottom, storage devices are displayed along with their corresponding device matrices. Figure reproduced (Öztürk et al., 2025). 82

5.1 Diagram of main classes in the PyFlexAD package using UML notation. Figure reproduced (Öztürk et al., 2025). 86
5.2 Diagram of the parameter classes in the PyFlexAD package using UML notation. Figure reproduced (Öztürk et al., 2025). 87
5.3 Software Architecture: Left: Central control structure. Right: Proposed aggregation-based control structure, featuring a virtual aggregation level at the center and a hierarchical aggregation structure on the far right. Figure reproduced (Öztürk et al., 2025). 89
5.4 An illustration featuring two polytopes, one in dashed blue and the other in dashed orange, along with their Minkowski sum in solid red, and the approximation in green. The operational points are indicated as dots within the corresponding flexibilities, each matching the respective colors. 91

5.5 Hierarchical aggregation setting used in the example computation, featuring seven aggregators, each connected to two devices. 93

6.1 Storage model with (dis-)charging efficiencies. The model without complementary constraint is shown in orange and the model with complementary constraint is shown in dotted blue. 96

6.2 The black line represents a convex storage model. The blue lines depict storages exhibiting variations in power (left), energy (center), and both power and energy (right). The orange shape illustrates the resulting polytope obtained via robust pre-processing. 102

7.1 Example of a power network with five buses connected via branches. 105

List of Tables

2.1 Conversion table for convex storage parameters: $\underline{x}, \overline{x}, \underline{S}, \overline{S}$, α, and S_{init}. Table reproduced (Öztürk et al., 2025). 16

3.1 Communication effort (floating-point numbers sent to the auxiliary service purchaser) and computation effort as functions of the number of devices n and the number of time periods d. Table reproduced (Öztürk et al., 2022). 42

4.1 Max UPR values and max calculation time for 4, 8, 12, 16, 20, 24 time periods and 2, 6, 10, 20, 30 batteries for different inner approximation methods. Table reproduced (Öztürk, Faulwasser, et al., 2024). 68
4.2 Maximum UPR values and maximum computation time for 12, 24, . . ., 96 time periods and 50, 100, . . ., 500 batteries. Table reproduced (Öztürk, Faulwasser, et al., 2024). 70

5.1 BESS hardware parameters. Table reproduced (Öztürk et al., 2025). 86
5.2 EV hardware parameters. Table reproduced (Öztürk et al., 2025). 87
5.3 TCL hardware parameters. Table reproduced (Öztürk et al., 2025). 88

1. Introduction

The European Green Deal, launched by the European Commission in December 2019 (European Commission, 2021), outlines an ambitious plan to transform the European Union into a climate-neutral and sustainable economy by 2050. Central to the Green Deal's vision are significant reductions in greenhouse gas emissions, a push for renewable energy, enhanced energy efficiency, and the preservation of biodiversity, all with the ultimate aim of achieving net-zero emissions by mid-century. This transformation is expected to drive innovation, create green jobs, and position Europe as a global leader in environmental sustainability. In line with this vision, Austria has committed to achieving climate neutrality by 2040 and has outlined an ambitious plan to transition to renewable energy sources (IEA, 2020).

This thesis offers potential contributions toward these goals by proposing auxiliary service strategies that can help reduce peaks in power grids and, consequently, minimize the need for extensive power generation. The following chapter introduces key topics such as flexibility in power grids, demand response strategies, and the aggregation of demand-side flexibilities. Additionally, gaps in the current literature are identified, the research question is formulated, and the contributions are outlined.

1.1. Flexibility in Power Grids

The increasing integration of renewable energy resources such as solar and wind energy coupled with the rising demand for electricity poses a series of new challenges for modern power grids (Deguenon et al., 2023). As the energy consumption increases, it is becoming increasingly difficult to balance fluctuating energy production with rising demand. Conventional supply-side management strategies, such as the control of flexible generators by system operators, are no longer suitable in view of the transforming energy landscape from monopolies to distributed energy resources and competitive markets. At the same time, flexible devices such as battery energy storage systems (BESSs), thermostatically controlled loads (TCLs), and electric vehicles (EVs) are being integrated into the system on a large scale, e.g., the share of air conditioning systems in Europe is estimated at 19% (IEA, 2023), while the market share of EVs in Austria is expected to reach around 38.5% in 2050 (Gryparis et al., 2020). Moreover, the number of registered flexible devices in Germany

reached 1.8 million in 2022 (Bundesnetzagentur, 2023). Accommodating these flexibilities on the demand side in addition to the flexibility on the supply side will be essential to ensure reliable future grid operations.

Flexibility, in this context, refers to the device's ability to respond to changes in demand, supply, or other external factors: EVs are often inactive for long periods of time, such as overnight or during daytime parking, which opens the possibility of modifying charging schedules. BESSs can store excess energy generated during peak times and release it during periods of high demand. The power consumption of TCLs such as heating, ventilation, and air conditioning (HVAC) systems can be modified without affecting the specified room temperatures. By utilizing the potential flexibility of these devices, consumers can adapt their electricity consumption and help to manage electricity demand more effectively, which prevents unnecessary investments in generation, transmission, and distribution systems.

Demand response (DR) programs, which are broadly divided into incentive-based DR programs and price-based DR programs, are designed to take advantage of this potential flexibility of consumer devices. Incentive-based DR programs offer consumers incentives to change their power profile, e.g., direct load control, where consumers are registered to enable a system operator to control their appliances. Price-based DR programs, on the other hand, charge consumers different prices at different times, e.g., time-of-use prices, critical peak prices, and real-time prices. The purpose here is to encourage consumers to adjust their energy consumption in line with price trends. For a comprehensive overview of DR programs, we refer to Jordehi, 2019.

Used individually, a single device such as a BESS may not offer significant flexibility. However, when paired with multiple other appliances, this combination can act as a virtual power plant and assist in maintaining grid operations. Various methods can be utilized to effectively manage a set of flexible appliances.

A straightforward method is to use a central control scheme, e.g., a system operator, which directly controls and communicates with each device individually. Although effective, this control scheme suffers from drawbacks, such as privacy issues, as the central control unit requires system-specific information, e.g., user profile of EVs, and assigns power profiles directly to individual devices. In addition, this method suffers from limited scalability, as communication and the size of the control problem increase with the number of devices (Öztürk et al., 2022).

An alternative approach is to use distributed optimization methods, which can alleviate scalability issues by clustering multiple devices into nodes. However, these methods often require a predefined problem structure, i.e., restrictions on the objective function, cf. Boyd, 2010. Additionally, information must be exchanged between nodes, which can raise data protection issues and result in communication overhead. Furthermore, not all distributed optimization algorithms guarantee convergence. For recent reviews of distributed optimization strategies and their applications, see Braun et al., 2018; Engelmann, 2022; Engelmann et al., 2020, 2017, 2019; Yang et al., 2019; Zheng and Liu, 2022.

Yet, another approach involves the introduction of a third entity, e.g., an aggregator, which coordinates the individual appliances via a central platform. This aggregator can leverage the flexibility of various appliances to participate in energy markets, e.g., as part of demand response programs.

1.2. Aggregation of Demand-Side-Flexibilities

Aggregators generally act as intermediaries between grid operators and consumers by monitoring consumers' devices, assessing the overall flexibility, and participating in energy markets (Gkatzikis et al., 2013). In return for their participation, consumers receive financial incentives, lower energy costs, or other benefits. The aggregator, in this setting, is the only entity that knows the system specifics and has authority over contracted consumers' devices, which ensures data protection and prevents unauthorized access by third parties. In addition, system operators are relieved from the burden of overseeing and controlling numerous devices, which reduces the communication effort significantly.

Figure 1.1 depicts a typical scenario involving an aggregator. The flexibilities provided by the devices on the right are collected and made accessible to third parties on the left by the aggregator. This guarantees access to the potential flexibility while safeguarding critical information.

A key challenge in aggregation-based control is, however, the precise quantification of the aggregated flexibility. A device's flexibility can be represented by a set of its potential power profiles. These power profiles are typically modeled as piecewise constant functions, i.e., they are assumed to remain constant over designated time intervals, typically 15 minutes in power systems. This characteristic enables these functions to be expressed as vectors in d-dimensional space. Consequently, the number of time periods dictates the dimension of the flexibility sets.

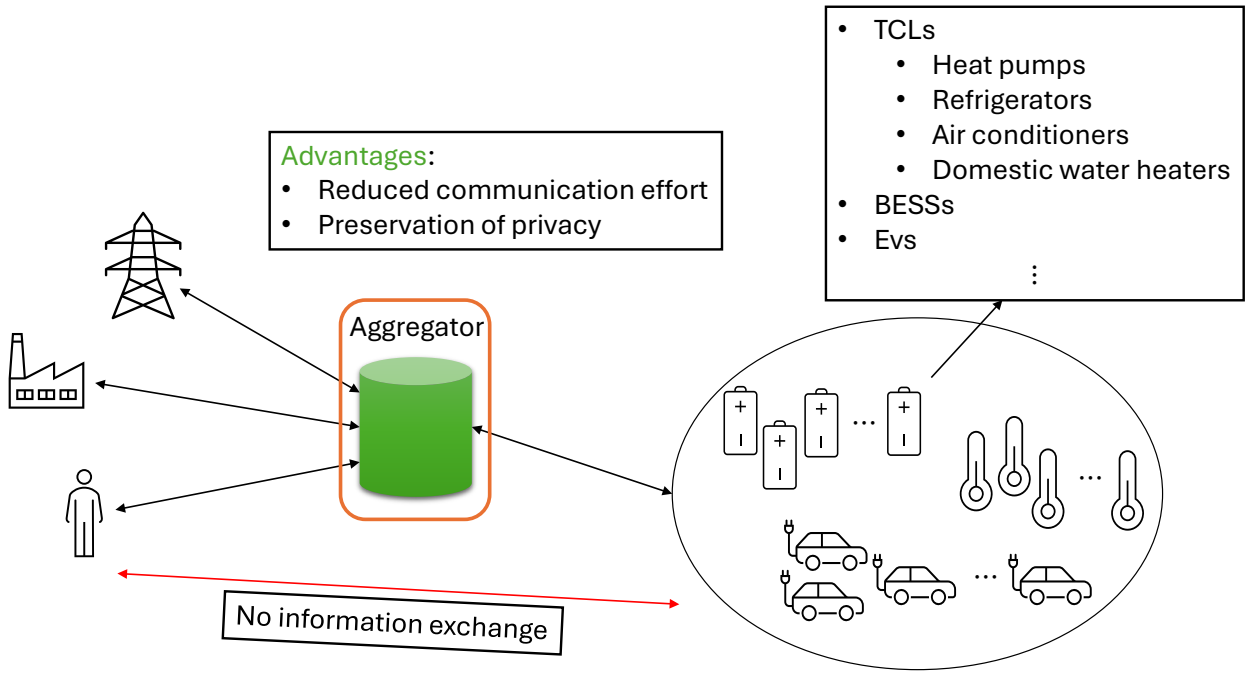

Figure 1.1.: Sketch of a system featuring flexible devices and an aggregator. The flexibilities provided by the devices are aggregated and forwarded to purchasers. Specific information is encapsulated and remains unknown to third parties.

Given flexibility sets $\mathcal{X}_i \subseteq \mathbb{R}^d$, $i = 1, \ldots, n$, the aggregated flexibility of multiple devices is expressed as the sum of individual flexibility sets. This operation is commonly referred to as the Minkowski sum, which is formally defined as

$$\oplus_{i=1}^{n} \mathcal{X}_i := \left\{ x \in \mathbb{R}^d : x = \sum_{i=1}^{n} x_i,\ x_i \in \mathcal{X}_i \right\}. \tag{1.1}$$

However, since this addition is known to be computationally prohibitive (Tiwary, 2008), numerous approximate quantification methods have been proposed in the literature.

Approximation strategies can be categorized into top-down and bottom-up approaches. Top-down approaches estimate the aggregated flexibility directly using statistical methods, machine learning or geometric projection techniques. Bottom-up approaches first approximate the individual sets by, e.g., zonotopes, homothets, etc. Then, the approximated sets are added instead of the original sets to obtain the approximate aggregate flexibility.

Approximation strategies can be further categorized into inner and outer approximations. Inner approximations are conservative, i.e., the feasible region is usually reduced, leaving out potential power profiles. Outer approximations, on the other hand, overestimate the potential feasible region and may contain infeasible power profiles.

Auxiliary service providers must ensure the fulfillment of the power profiles to avoid contractual penalties, which favors inner approximations. However, state-of-the-art inner approximations suffer from objective-dependent performance, increased computational complexity, and may exclude nominal power profiles, such as the idle state (Öztürk et al., 2022). Indeed, the increased computational complexity limits the application potential of various methods considerably. In addition, aggregation strategies are often designed for specific devices, requiring both device-specific algorithm selection and, considering the objective-dependent performance, objective-specific algorithm selection.

Given the aggregated flexibility in the system, the aggregator can market the set of feasible power profiles in energy markets. Third parties, e.g., network operators, can select power profiles that the aggregator then distributes (disaggregates) to the individual devices, i.e., the operation schedule of the devices is configured. This disaggregation is not unique, as different individual power profiles can be combined to obtain a specific power profile in the aggregated set. Consequently, this has led to the development of various methods in the literature to decompose a selected power profile into its components. However, many state-of-the-art disaggregation strategies, e.g., by Barot, 2017; Müller et al., 2015, involve solving optimization problems that require considerable computational effort.

1.3. Challenges

While the increasing integration of flexible devices into power grids presents significant opportunities, it also introduces several challenges that must be addressed to fully harness this potential. A key concern is the need for effective aggregation techniques that enable multiple devices to function as a virtual power plant. However, accurately quantifying the aggregated flexibility of multiple devices requires calculating the Minkowski sum, which can be complex and challenging.

As a result, reliance on approximate quantification techniques has become necessary. For the provision of auxiliary services, inner approximations are preferred to avoid penalties associated with unmet power profiles. Nonetheless, current state-of-the-art inner approximations encounter several hurdles, including:

- *Variable performance*, i.e., the effectiveness of current approximations can differ based on the specific objectives, leading to inconsistent performance.

- *Computational complexity*, i.e., the methods often require significant computational resources, particularly when navigating high-dimensional spaces, making them less viable for day-ahead computations.
- *Exclusion of nominal profiles*, i.e., many existing techniques may overlook critical power profiles, such as the idle state, which can undermine the overall flexibility.
- *Disaggregation challenges*, i.e., techniques aimed at disaggregating frequently rely on optimizations that are resource-intensive, further complicating their implementation in dynamic environments.

Given these challenges, a need exists for the development of accurate and scalable approximation techniques to quantify the aggregated flexibility of flexible devices. Additionally, effective disaggregation strategies must be established to ensure that these systems operate efficiently and reliably within modern power grids.

1.4. Outline and Contributions

This thesis develops novel and performant (dis-)aggregation strategies that overcome the challenges described in the introductory paragraph and enable aggregators to play a more effective role in emerging energy markets.

Chapter 2 – Problem Statement and Modelling

Preliminaries, including the detailed modeling of flexible devices such as BESSs, TCLs, and EVs are given in Chapter 2. It is shown that these models can be mapped to a convex storage model. A guideline for the conversion of selected devices is developed and provided in the form of a compact table. In addition, the relationship between the set addition and the aggregated flexibility is highlighted. Chapter 2 ends with an analysis of the computational complexity of the set addition, and the necessity for approximate methods.

Chapter 3 – Approximation Strategies

Chapter 3 provides a comprehensive examination of approximate aggregation strategies found in the literature. Additionally, an open-source benchmark is introduced, specifically designed to evaluate these strategies. Thirteen state-of-the-art algorithms are implemented

and tested within this framework. A thorough analysis is conducted, focusing on the limitations of the existing algorithms. The results of this chapter have been published (Öztürk et al., 2022). The primary contribution of this chapter is a thorough analysis of existing algorithms, along with the development of an open-source benchmark that highlights their weaknesses.

Chapter 4 – Scalable Aggregation and Disaggregation

Based on these findings, a novel method for the approximate aggregation of convex storage models is presented in Chapter 4. Section 4.1 proposes a novel aggregation strategy for polytopes that satisfy two specific assumptions. Analytical results are derived, and their implications are discussed. In particular, an important finding is that the proposed computations yield vertices within the aggregated set, introducing a new technique for computing a subset of Minkowski sum vertices.

The proposed aggregation strategy is then extended to BESS with minimum final energy restriction in Section 4.2, and tested within the previously developed benchmark against ten state-of-the-art inner approximations. It is shown that the proposed method outperforms the others in terms of computational complexity and accuracy for different objectives.

Section 4.3 extends the proposed aggregation strategy further to include convex storage models that can be applied in various real-world contexts. To this end, two correction algorithms are developed and thoroughly discussed. The extended algorithm is then tested in practical settings, which include scenarios with a single aggregator utilizing one type of flexibility, as well as scenarios with multiple aggregators and multiple flexibility types. In these experiments, the flexibility provided by the devices is utilized to reduce the peak demand in residential areas.

Section 4.4 concludes the aggregation-based control problem by introducing a novel disaggregation strategy. This strategy is compatible with all aggregation strategies proposed in this thesis and can be computed efficiently without requiring any optimization. The results of this chapter have been published (Öztürk, Faulwasser, et al., 2024; Öztürk, Kaspar, et al., 2024; Öztürk et al., 2025). The main contribution of this chapter is a broadly applicable (dis-)aggregation strategy for convex storage models.

Chapter 5 – Software Implementation

In Chapter 5, all previous findings are brought together to develop a comprehensive open-source Python package for the (dis-)aggregation of flexible devices. The package is described in detail together with class diagrams and the software architecture. Furthermore, sample codes are provided and analyzed to ease the introduction to the software. The results of this chapter have been published (Öztürk et al., 2025). The main contribution of this chapter is a comprehensive open-source Python package designed for application in real-world data environments.

Chapter 6 – Extensions towards Non-convexity and Uncertainty

Chapter 6 introduces further extensions to the aggregation strategy, starting with a discussion on aggregating non-convex sets and a review of existing literature on non-convex aggregation methods. An improved algorithm is then presented to address the challenges posed by non-convex sets. The chapter also explores aggregation in the context of uncertainties, reviewing relevant literature and proposing an extension to address uncertainties in practical applications. These extensions are currently being prepared for publication and have not yet been submitted or published.

Chapter 7 – Conclusions and Perspectives

Finally, Chapter 7 addresses related topics such as grid-aware aggregation strategies, i.e., aggregation of flexible devices subject to grid constraints. The latest research findings in this area are briefly presented and discussed, followed by concluding remarks and an outline of potential directions for future research.

2. Problem Statement and Modelling

This chapter provides an overview of the mathematical basics used in this thesis. Initially, a detailed analysis of common flexible devices within power grids is conducted, recalling their mathematical characteristics. Subsequently, a comprehensive convex storage model is introduced along with a guideline on how to convert typical flexible devices into convex storage models. This standardization simplifies future research, focusing attention solely on the convex storage model. Finally, the concept of set addition is outlined, highlighting its relationship with the aggregation of flexible devices and analyzing its computational complexity.

2.1. Flexible Devices

This section provides detailed modeling of specific flexible devices in real-world scenarios. Readers may skip this section if mathematical details are not required and proceed to the next section, which discusses the convex storage model that applies to all devices covered in this section.

In a discrete-time setting comprising d time intervals of length Δt, the power in and out of a device within each interval can be modeled by a vector $x \in \mathbb{R}^d$, with the t^{th} element denoting the power flow during time period t. Real-world devices are subject to constraints that restrict power flow, thereby defining the range of feasible power profiles. The constraints and consequently the feasible power profiles are detailed below for common devices in practical applications, whereby the feasible regions are rewritten to emphasize the mathematical similarities between the devices.

2.1.1. Battery Energy Storage Systems

A BESS, e.g., a stationary battery, is a device that can store and release energy on demand, enabling flexible operation. These devices can be characterized by the parameter vector

$$(x_{\min}, x_{\max}, S_{\min}, S_{\max}, \alpha, S_{\text{init}}, S_f)^\top \in (-\infty, 0] \times [0, \infty)^2 \times (S_{\min}, \infty) \times (0, 1] \times [S_{\min}, S_{\max}]^2,$$

where $x_{\min}, x_{\max}$ (kW) are the lower and upper power bounds for (dis-)charging, $S_{\min}, S_{\max}$ (kWh) the lower and upper bounds on the energy in the BESS, α the self-discharge factor, S_{init} (kWh) the initial energy, and S_f (kWh) the minimum final energy.

The energy stored at a given time period, S_t, is determined by the energy in the previous time period, S_{t-1}, multiplied by the discharge factor, α, plus the energy (dis-)charged during the current time period, $x_t \Delta t$. This relationship can be expressed mathematically as

$$S_t = \alpha S_{t-1} + x_t \Delta t.$$

The discharge factor α reduces the energy in each time period, while $x_t \Delta t$ decreases the energy in the storage if $x_t < 0$, and increases the energy in the storage if $x_t > 0$. Together with the (dis-)charging limits, energy limits, and the specification of the initial energy, the system dynamics and constraints of a BESS can be described as

$$x_{\min} \leq x_t \leq x_{\max}, \forall t = 1, \ldots, d, \tag{2.1a}$$
$$S_t = \alpha S_{t-1} + x_t \Delta t, \forall t = 1, \ldots, d, \tag{2.1b}$$
$$S_{\min} \leq S_t \leq S_{\max}, \forall t = 1, \ldots, d-1, \tag{2.1c}$$
$$S_f \leq S_d \leq S_{\max}, \tag{2.1d}$$
$$S_0 = S_{\text{init}}. \tag{2.1e}$$

Equations (2.1a), (2.1c), and (2.1d) impose limitations on the power flow and energy within the BESS, (2.1b) defines the procedure for updating the energy level, and (2.1e) establishes the initial energy. The above model can be subsumed to

$$x_{\min} \leq x_t \leq x_{\max}, \forall t = 1, \ldots, d, \tag{2.2a}$$
$$S_{\min} \leq \alpha^t S_{\text{init}} + \sum_{\tau=1}^{t} \alpha^{t-\tau} x_\tau \Delta t \leq S_{\max}, \forall t = 1, \ldots, d-1, \tag{2.2b}$$
$$S_f \leq \alpha^d S_{\text{init}} + \sum_{\tau=1}^{d} \alpha^{d-\tau} x_\tau \Delta t \leq S_{\max}. \tag{2.2c}$$

Note that this model does not account for (dis-)charging efficiencies, which depend on the type of battery used (e.g., lead-acid, NiCd, Li-Ion, etc.). A comprehensive analysis of various battery types and their efficiencies is provided by Andújar Márquez et al., 2023. Including these efficiencies leads to a more accurate model, but also introduces non-convexities, as discussed in Chapter 6. Additionally, some researchers, including Zargari et al., 2023,

have incorporated ramp constraints, which are omitted in this model.

2.1.2. Thermostatically Controlled Loads

TCLs such as air conditioning, refrigerators, heat pumps, and water heaters are appliances that are specifically designed to regulate and maintain a desired temperature range. These devices can be parameterized by a vector

$$(p_{\max}, R, C, \eta, \Delta, \theta_a, \theta_r, \theta_{\text{init}})^\top \in [0, \infty)^7 \times [\theta_r - \Delta, \theta_r + \Delta],$$

where $p_{\max}$ (kW) is the maximum power, R (K/kW) the thermal resistance, C (kWh/K) the thermal capacitance, η the coefficient of performance, Δ (K) the dead band, θ_a (K) the ambient temperature, θ_r (K) the set point temperature, and θ_{init} (K) the initial temperature.

The temperature change in a TCL is usually described by a linear time dependence on the temperature difference, together with an additional term for the power consumption $p \in \mathbb{R}^d$ and the demand $q \in \mathbb{R}^d_{\geq 0}$, i.e.,

$$\frac{\theta_t - \theta_{t-1}}{\Delta t} = a(\theta_a - \theta_{t-1}) \pm \left(\frac{q_t}{C} - b p_t\right),$$

where $a = \frac{1}{RC}$, and $b = \frac{\eta}{C}$. Note that the $\pm$ on the right-hand side is required to distinguish between cooling and heating. In the following, we use the plus sign for cooling and the minus sign for heating, unless otherwise specified and $\pm$ appears. With this convention, the system dynamics and constraints of a TCL can be described by

$$0 \leq p_t \leq p_{\max}, \forall t = 1, \ldots, d, \tag{2.3a}$$

$$\theta_t = (1 - a\Delta t)\theta_{t-1} + a\theta_a \Delta t \pm \Delta t \left(\frac{q_t}{C} - b p_t\right), \forall t = 1, \ldots, d, \tag{2.3b}$$

$$\theta_r - \frac{\Delta}{2} \leq \theta_t \leq \theta_r + \frac{\Delta}{2}, \forall t = 1, \ldots, d, \tag{2.3c}$$

$$\theta_0 = \theta_{\text{init}}. \tag{2.3d}$$

Equation (2.3a) limits the power flow, (2.3b) provides a rule for updating the temperature, (2.3c) requires the temperature to be within the dead band around the setpoint temperature, and (2.3d) specifies the initial temperature. The power required to maintain the setpoint temperature in the absence of a demand q_t from the appliance is given by $x_0 = \pm\frac{\theta_a - \theta_r}{\eta R}$. With

the variable transformations

$$S_t = \frac{C(\theta_t - \theta_r)}{\eta}, \text{ and } x_t = \pm(-p_t + x_0),$$

(2.3) can be rewritten for a cooling device as

$$x_0 - p_{\max} \leq x_t \leq x_0, \forall t = 1, \ldots, d, \tag{2.4a}$$

$$S_t = \alpha S_{t-1} + x_t \Delta t + \frac{q_t}{\eta} \Delta t, \forall t = 1, \ldots, d, \tag{2.4b}$$

$$-\frac{C\Delta}{2\eta} \leq S_t \leq \frac{C\Delta}{2\eta}, \forall t = 1, \ldots, d, \tag{2.4c}$$

$$S_0 = S_{\text{init}}, \tag{2.4d}$$

where $\alpha = (1 - a\Delta t)$, and $S_{\text{init}} = \frac{C(\theta_{\text{init}} - \theta_r)}{\eta}$. The above equations can be subsumed to

$$x_0 - p_{\max} \leq x_t \leq x_0, \forall t = 1, \ldots, d, \tag{2.5a}$$

$$-\frac{C\Delta}{2\eta} - \frac{\Delta t}{\eta} \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau \leq \alpha^t S_{\text{init}} + \sum_{\tau=1}^{t} \alpha^{t-\tau} x_\tau \Delta t \leq \frac{C\Delta}{2\eta} - \frac{\Delta t}{\eta} \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau, \forall t = 1, \ldots, d. \tag{2.5b}$$

The equations for a heating device can be derived equivalently and are expressed as

$$-x_0 \leq x_t \leq p_{\max} - x_0, \forall t = 1, \ldots, d, \tag{2.6a}$$

$$-\frac{C\Delta}{2\eta} + \frac{\Delta t}{\eta} \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau \leq \alpha^t S_{\text{init}} + \sum_{\tau=1}^{t} \alpha^{t-\tau} x_\tau \Delta t \leq \frac{C\Delta}{2\eta} + \frac{\Delta t}{\eta} \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau, \forall t = 1, \ldots, d. \tag{2.6b}$$

Note that these models assume continuous power flow, cf. (2.3a), which introduces a simplification. A more accurate model functions at a fixed power level that can be switched on or off, typically represented by a binary variable, cf. Zhao et al., 2017. For a recent review of modeling thermostatically controlled loads, we refer to Tian et al., 2024.

2.1.3. Electric Vehicles

EVs can be described as BESSs with an additional binary availability vector $\lambda \in \{0, 1\}^d$ and a trip consumption vector $q \in \mathbb{R}^d_{\geq 0}$. When the EV is connected to a charging station, the availability is set to one during this period; otherwise, it is set to zero. The behavior of the

system is then governed by the following dynamics and constraints

$$\lambda_t x_{\min} \leq x_t \leq \lambda_t x_{\max}, \forall t = 1, \ldots, d, \tag{2.7a}$$
$$S_t = \alpha S_{t-1} + x_t \Delta t - q_t \Delta t, \forall t = 1, \ldots, d, \tag{2.7b}$$
$$S_{\min} \leq S_t \leq S_{\max}, \forall t = 1, \ldots, d-1, \tag{2.7c}$$
$$S_f \leq S_d \leq S_{\max}, \tag{2.7d}$$
$$S_0 = S_{\text{init}}. \tag{2.7e}$$

Equations (2.7a), (2.7c) and (2.7d) pose limitations on the power flow and the stored energy, (2.7b) provides an update rule for the energy in the storage, and (2.7e) specifies the initial energy. The above model can be subsumed to

$$\lambda_t x_{\min} \leq x_t \leq \lambda_t x_{\max}, \forall t = 1, \ldots, d, \tag{2.8a}$$
$$S_{\min} + \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau \Delta t \leq \alpha^t S_{\text{init}} + \sum_{\tau=1}^{t} \alpha^{t-\tau} x_\tau \Delta t \leq S_{\max} + \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau \Delta t, \forall t = 1, \ldots, d-1, \tag{2.8b}$$
$$S_f + \sum_{\tau=1}^{d} \alpha^{d-\tau} q_\tau \Delta t \leq \alpha^d S_{\text{init}} + \sum_{\tau=1}^{d} \alpha^{d-\tau} x_\tau \Delta t \leq S_{\max} + \sum_{\tau=1}^{d} \alpha^{d-\tau} q_\tau \Delta t. \tag{2.8c}$$

Note that the lower limit for discharging is modeled by $\lambda_t x_{\min}$, which accommodates vehicle-to-grid (V2G) technology. If this is not desired, one can simply set $x_{\min}$ to zero. Note also that the above model disregards the (dis-)charging efficiencies and is, therefore, a simplification. The more accurate model with (dis-)charging efficiencies results in a non-convex model and is briefly discussed in Chapter 6.

2.1.4. Pumped Hydro Energy Storages

In a pumped hydro energy storage (PHES), water is pumped from a lower reservoir to an upper reservoir to increase its potential energy. This stored energy can then be converted back into electrical energy when needed. Hydropower (including run-of-river, reservoir, and pumped storage plants) plays a vital role in Austria's electricity system — supplying around 60 % of the country's electricity generation (Ember, 2024). A PHES can be characterized by the parameter vector

$$(x_{\min}, x_{\max}, R_{\min}, R_{\max}, h, R_{\text{init}})^\top \in (-\infty, 0] \times [0, \infty)^4 \times [R_{\min}, R_{\max}],$$

where $x_{\min}, x_{\max}$ (kW) are the lower and upper power bounds, $R_{\min}, R_{\max}$ (m^3) are the lower and upper volume limits for the upper reservoir, R_{init} (m^3) the initial water volume in the upper reservoir, and h (m) the height between upper and lower reservoirs.

The power needed to move water from the lower reservoir to the upper reservoir, denoted as $x_{p,t} \in \mathbb{R}_{\geq 0}$, along with the power $x_{g,t} \in \mathbb{R}_{\geq 0}$ generated during turbine operation in time period t, can be expressed as

$$x_{p,t} = \frac{m_{p,t}gh}{\Delta t} = \rho \dot{V}_{p,t} gh,$$
$$x_{g,t} = \frac{m_{g,t}gh}{\Delta t} = \rho \dot{V}_{g,t} gh,$$

where $m_{p,t}, m_{g,t}$ (kg) and $\dot{V}_{p,t}, \dot{V}_{g,t}$ (m^3/s) are the masses and volume flow rates of the water pumped into or released from the upper reservoir in time period t, respectively, $g = 9.81$ (m/s^2) is the gravitational acceleration, and $\rho = 1000$ (kg/m^3) is the water density. We define further the unit conversion parameter

$$\eta^\star = \frac{\dot{V}_{p,t}}{x_{p,t}} = \frac{\dot{V}_{g,t}}{x_{g,t}} = \frac{1}{\rho g h}.$$

Using $x_t = x_{p,t} - x_{g,t}$, the system dynamics and constraints can be written as

$$x_{\min} \leq x_t \leq x_{\max}, \forall t = 1, \ldots, d, \tag{2.10a}$$
$$R_t = R_{t-1} + \eta^\star x_t \Delta t, \forall t = 1, \ldots, d, \tag{2.10b}$$
$$R_{\min} \leq R_t \leq R_{\max}, \forall t = 1, \ldots, d, \tag{2.10c}$$
$$R_0 = R_{\text{init}}. \tag{2.10d}$$

Equations (2.10a) and (2.10c) pose limitations on the power and water volume in the upper reservoir, (2.10b) provides an update rule for the volume in the upper reservoir, and (2.10d) specifies the initial water volume in the upper reservoir. The above equations can be subsumed to

$$x_{\min} \leq x_t \leq x_{\max}, \forall t = 1, \ldots, d \tag{2.11a}$$
$$\frac{R_{\min}}{\eta^\star} \leq \frac{R_{\text{init}}}{\eta^\star} + \sum_{\tau=1}^{t} x_\tau \Delta t \leq \frac{R_{\max}}{\eta^\star}, \forall t = 1, \ldots, d. \tag{2.11b}$$

Note that the above model is expressed in SI units and that a conversion factor of $3.6^{-1} \cdot 10^{-6}$ must be taken into account when using the unit kWh. Also note that this model disregards

the efficiencies in pumping and generation, i.e, it is assumed that energy is not lost during these processes. Actual pump and turbine efficiencies are between $75 - 85\%$ and $93 - 95\%$, respectively (Giesecke & Heimerl, 2014). For a more detailed model that takes these efficiencies into account, we refer to Sass et al., 2020; Zhou et al., 2017.

2.2. Convex Storage Models

So far, the reader may have noticed the similarities between the previous models, which raises the possibility of integrating these devices into one cohesive model. Hence, this section focuses on that very objective: introducing a storage model that unifies the previous models.

A storage model that encompasses the previous models can be described by the system dynamics and constraints

$$\underline{x}_t \leq x_t \leq \overline{x}_t, \forall t = 1, \dots, d, \tag{2.12}$$

$$S_t = \alpha S_{t-1} + x_t \Delta t, \forall t = 1, \dots, d, \tag{2.13}$$

$$\underline{S}_t \leq S_t \leq \overline{S}_t, \forall t = 1, \dots, d, \tag{2.14}$$

$$S_0 = S_{\text{init}}, \tag{2.15}$$

where $\underline{x}, \overline{x}, \underline{S}, \overline{S} \in \mathbb{R}^d$ denote the lower and upper limits for the variables x and S, respectively, $S_{\text{init}} \in \mathbb{R}$ denotes the initial energy, and $\alpha \in (0, 1]$ the self-discharge factor. This model can be written as

$$\underline{x}_t \leq x_t \leq \overline{x}_t, \forall t = 1, \dots, d, \tag{2.16a}$$

$$\underline{S}_t \leq \alpha^t S_{\text{init}} + \sum_{\tau=1}^{t} \alpha^{t-\tau} x_\tau \Delta t \leq \overline{S}_t, \forall t = 1, \dots, d. \tag{2.16b}$$

Note that the power and energy limits in this model may change over time. Hence, by selecting the limits and parameters appropriately, the previous models can be mapped to this model. Table 2.1 lists the parameters that need to be selected for modeling the previously discussed devices by the above storage model. The conversions outlined in Table 2.1 are derived directly from the system dynamics and constraints discussed in the previous section for the respective devices.

Table 2.1.: Conversion table for convex storage parameters: $\underline{x}, \overline{x}, \underline{S}, \overline{S}$, α, and S_{init}. Table reproduced (Öztürk et al., 2025).

	$\underline{x}_t$	$\overline{x}_t$	$\underline{S}_t$	$\overline{S}_t$	α	S_{init}
BESS	$x_{\min}$	$x_{\max}$	$S_{\min}\ \forall t = 1, \dots d-1,$ S_f for $t = d$	$S_{\max}$	α	S_{init}
TCL (cooling)	$\frac{\theta_a - \theta_r}{\eta R} - p_{\max}$	$\frac{\theta_a - \theta_r}{\eta R}$	$-\frac{C\Delta}{2\eta} - \frac{\Delta t}{\eta} \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau$	$\frac{C\Delta}{2\eta} - \frac{\Delta t}{\eta} \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau$	$1 - \frac{\Delta t}{RC}$	$\frac{C(\theta_{\text{init}} - \theta_r)}{\eta}$
TCL (heating)	$\frac{\theta_a - \theta_r}{\eta R}$	$p_{\max} + \frac{\theta_a - \theta_r}{\eta R}$	$-\frac{C\Delta}{2\eta} + \frac{\Delta t}{\eta} \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau$	$\frac{C\Delta}{2\eta} + \frac{\Delta t}{\eta} \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau$	$1 - \frac{\Delta t}{RC}$	$\frac{C(\theta_{\text{init}} - \theta_r)}{\eta}$
EV	$\lambda_t x_{\min}$	$\lambda_t x_{\max}$	$S_{\min} + \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau \Delta t\ \forall t = 1, \dots d-1,$ $S_f + \sum_{\tau=1}^{d} \alpha^{d-\tau} q_\tau \Delta t$ for $t = d$	$S_{\max} + \sum_{\tau=1}^{t} \alpha^{t-\tau} q_\tau \Delta t$	α	S_{init}
PHES	$x_{\min}$	$x_{\max}$	$\rho g h R_{\min}$	$\rho g h R_{\max}$	1	$\rho g h R_{\text{init}}$

The storage model defined by (2.16) results in a polyhedron, which we define as follows. Let $\underline{x}, \overline{x}, \underline{S}, \overline{S} \in \mathbb{R}^d$, $\alpha \in (0, 1]$, and $S_{\text{init}} \in \mathbb{R}$ be given. Then, we define the following set as the convex storage model

$$\mathcal{B}(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}) \coloneqq \left\{ x \in \mathbb{R}^d : A(\alpha)x \leq b(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}) \right\}. \tag{2.17}$$

In the above definition, $A \in \mathbb{R}^{4d \times d}$ and $b \in \mathbb{R}^{4d}$ are defined by

$$A(\alpha) \coloneqq \left(-I, I, \Gamma^\top, -\Gamma^\top \right)^\top, \tag{2.18a}$$

$$b(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}) \coloneqq \left(-\underline{x}^\top, \overline{x}^\top, \frac{1}{\Delta t}(\overline{S} - S_{\text{init}} a_d)^\top, \frac{1}{\Delta t}(-\underline{S} + S_{\text{init}} a_d)^\top \right)^\top, \tag{2.18b}$$

where $a_d \coloneqq (\alpha, \alpha^2, \ldots, \alpha^d)^\top$, $I \in \mathbb{R}^{d \times d}$ is the identity matrix, and $\Gamma \in \mathbb{R}^{d \times d}$ is a Toeplitz matrix[1] with first column and row defined by $(1, \alpha, \ldots, \alpha^{d-1})^\top$ and $(1, 0, \ldots, 0)$, respectively. Additionally, the polyhedron above is bounded by the power constraints and is, therefore, a polytope.

By mapping the previous devices to the convex storage model, we can focus our analysis exclusively on this model. All (dis-)aggregation strategies that apply to the convex storage model also apply to the previous devices.

It should also be noted that the application of the convex storage model is not limited to the devices discussed above. Alternative energy storage systems, including flywheel energy storage that harnesses angular momentum, chemical storage systems like hydrogen storage, and mechanical systems such as compressed air energy storage, are also viable options to consider. For a more comprehensive study on storage applications, we refer to Andújar Márquez et al., 2023; Calero et al., 2023.

Finally, since the polytope representation of a BESS model (cf. (2.1)) will be referenced later in the thesis, we provide a brief definition in this section. Given a parameter vector

$$p = (\alpha, x_{\min}, x_{\max}, S_{\min}, S_{\max}, \Delta t)^\top \in (0, 1] \times \mathbb{R}^4 \times (0, \infty),$$

[1] A Toeplitz matrix is defined as a matrix with constant diagonals that descend from left to right.

we use the notation

$$\mathcal{B}(S_{\text{init}}, S_{\text{f}}, p) \coloneqq \mathcal{B}(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}), \tag{2.19a}$$

$$\underline{x} = x_{\min}\mathbf{1}_d, \tag{2.19b}$$

$$\overline{x} = x_{\max}\mathbf{1}_d, \tag{2.19c}$$

$$\underline{S} = (S_{\min}\mathbf{1}_{d-1}^{\top}, S_f)^{\top}, \tag{2.19d}$$

$$\overline{S} = S_{\max}\mathbf{1}_d, \tag{2.19e}$$

for the storage model defined in (2.17) with the above-specified bounds, which represents a BESS model.

2.3. Flexibility Aggregation

Next, the concept of set addition, also known as Minkowski sum, is introduced and its connection to the aggregation of flexible devices is examined. In addition, basic polytope representations and their transformations are discussed. For a more detailed introduction to polytopes, we recommend the textbook by Boyd and Vandenberghe, 2004, which provides a comprehensive overview to the theory of convex sets and functions and forms the basis for various applications in optimization and related fields.

2.3.1. Polytope Basics

In the previous section, it was shown that the set of feasible power profiles of selected devices form polytopes. Polytopes are geometric objects characterized as the convex hull of a finite collection of points or alternatively as the intersection of a finite number of half-spaces. They can be described by their vertices, edges (the line segments that connect the vertices), and facets (the surfaces that form their boundaries). Polytopes, being bounded convex sets,[2] constitute a subcategory within the more general categories of polyhedra and convex sets. In mathematical terms, a polytope can be expressed in either half-space representation, i.e., with a matrix $A \in \mathbb{R}^{m\times d}$ and vector $b \in \mathbb{R}^m$ as

$$\mathcal{P} = \left\{x \in \mathbb{R}^d : Ax \leq b\right\},$$

[2] A set of points is convex if, for every pair of points within the set, the line segment connecting them is entirely contained within the set.

or the vertex representation, i.e., with the set of its vertices $\mathcal{V}$ as

$$\mathcal{P} = \text{Conv}(\mathcal{V}) := \left\{ x \in \mathbb{R}^d : x = \sum_{v \in \mathcal{V}} \alpha_v v,\ \sum_{v \in \mathcal{V}} \alpha_v = 1,\ \alpha_v \geq 0\ \forall v \right\}.$$

The equivalence of these representations is established by the Weyl-Minkowski theorem, which is often referred to as the *Main theorem*, cf. Ziegler, 1995.

The vertex representation can be derived from the half-space representation by determining all potential intersections of d half-planes. This involves selecting d rows in A and b and solving the resulting system of equations. Subsequently, any vector v that does not satisfy the condition $Av \leq b$ is discarded. Note that, since there are m half-planes, this method involves solving $\binom{m}{d}$ systems of equations.

Conversely, to convert the vertex representation into half-space representation, one must identify half-planes defined by a normal vector $a \in \mathbb{R}^d$ and an offset $p \in \mathbb{R}$. These half-planes are formed by selecting subsets of d vertices from $\mathcal{V}$. In total, there are $\binom{|\mathcal{V}|}{d}$ such subsets, where $|\mathcal{V}|$ is the cardinality of $\mathcal{V}$. Subsequently, all half planes that do not fulfill the condition $a^\top v \leq p, \forall v \in \mathcal{V}$ must be removed.

The conversion from the half-space representation to the vertex representation is known as vertex enumeration, while the reverse operation is known as facet enumeration. Unfortunately, both of these problems are in general classified as NP-hard problems,[3] cf. Avis et al., 2009.

2.3.2. Minkowski Summation

Given the feasible regions of n devices, the aggregated feasible region can be calculated by adding the individual feasible regions, e.g., for polytopes $\mathcal{P}_i$, $i = 1, \ldots, n$, the aggregated feasible region is described by the Minkowski sum

$$\oplus_{i=1}^{n} \mathcal{P}_i = \left\{ x \in \mathbb{R}^d : x = \sum_{i=1}^{n} x_i,\ x_i \in \mathcal{P}_i \right\}.$$

In the following, we briefly recall key properties of the Minkowski sum before discussing its computation. Some of these properties will be referenced in later sections.

[3] NP-hard problems are a class of computational problems that are at least as difficult to solve as the hardest problems in the complexity class NP (nondeterministic polynomial time).

Property 1 (Identity and Annihilation)**.** *Let $\mathcal{X} \subset \mathbb{R}^d$ be given. Then the following hold:*

- *The zero set, which contains only the zero vector, acts as the identity element, i.e., for any set $\mathcal{X}$, we have $\mathcal{X} \oplus \{\mathbf{0}_d\} = \mathcal{X}$.*
- *The empty set annihilates other sets, i.e., for every set $\mathcal{X}$, the sum of $\mathcal{X}$ and the empty set is the empty set: $\mathcal{X} \oplus \emptyset = \emptyset$.*

Property 2 (Convexity)**.** *Let non-empty sets $\mathcal{X}_i \subset \mathbb{R}^d$, $i = 1, \ldots, n$, be given. Then the following hold:*

- *The convex hull of the Minkowski sum is the Minkowski sum of the convex hulls, i.e.,*

$$Conv\left(\oplus_{i=1}^{n} \mathcal{X}_i\right) = \oplus_{i=1}^{n} Conv\left(\mathcal{X}_i\right).$$

- *The Minkowski sum preserves convexity. If $\mathcal{X}_i$, $i = 1, \ldots, n$, are convex sets, then $\oplus_{i=1}^{n} \mathcal{X}_i$ is convex.*

Property 3 (Linearity with Scalars)**.** *Let $\mathcal{X} \subset \mathbb{R}^d$, and $\mu, \lambda \in \mathbb{R}$ be given. Then, the following hold:*

- *If $\mathcal{X}$ is convex, and the scalars $\mu, \lambda > 0$, the operation satisfies:*

$$(\mu + \lambda)\mathcal{X} = \mu\mathcal{X} \oplus \lambda\mathcal{X}.$$

- *Conversely, if the above property holds for all non-negative real numbers λ, μ, then the set $\mathcal{X}$ is convex.*

Property 4 (Distributive)**.** *Let sets $\mathcal{X}_i, \subset \mathbb{R}^d$, $i = 1, \ldots, 4$, be given. Then, it holds that*

$$(\mathcal{X}_1 \cup \mathcal{X}_2) \oplus (\mathcal{X}_3 \cup \mathcal{X}_4) = (\mathcal{X}_1 \oplus \mathcal{X}_3) \cup (\mathcal{X}_1 \oplus \mathcal{X}_4) \cup (\mathcal{X}_2 \oplus \mathcal{X}_3) \cup (\mathcal{X}_2 \oplus \mathcal{X}_4).$$

Along with these properties, the Minkowski sum is both associative and commutative. For simplicity, the symbol $\mathcal{M}$ will be used throughout the remainder of this text to represent the Minkowski sum of sets. Having reviewed the fundamental properties, we can now proceed with the computation of the Minkowski sum.

Given sets of vertices $\mathcal{V}_i, i = 1, \ldots, n$, the Minkowski sum can be calculated by first adding all pairs of vertices, i.e., forming the set

$$\mathcal{V} = \left\{ v \in \mathbb{R}^d : v = \sum_{i=1}^{n} v_i,\ v_i \in \mathcal{V}_i \right\},$$

and then removing all redundant points, which is usually done by formulating linear programs (LPs), cf. Fukuda, 2004. Please note that this step necessitates

$$|\mathcal{V}_1||\mathcal{V}_2| \cdots |\mathcal{V}_n|$$

additions, along with a corresponding number of LP solutions, as each point in $\mathcal{V}$ must be verified. Figure 2.1 depicts the necessity of eliminating redundant computations, featuring two polytopes, $\mathcal{P}_1$ and $\mathcal{P}_2$, along with their Minkowski sum, denoted as $\mathcal{M}$. Each polytope consists of 5 vertices (indicated by red crosses), which results in a total of 25 potential vertex combinations. However, since the Minkowski sum only has 5 vertices, not all pos-

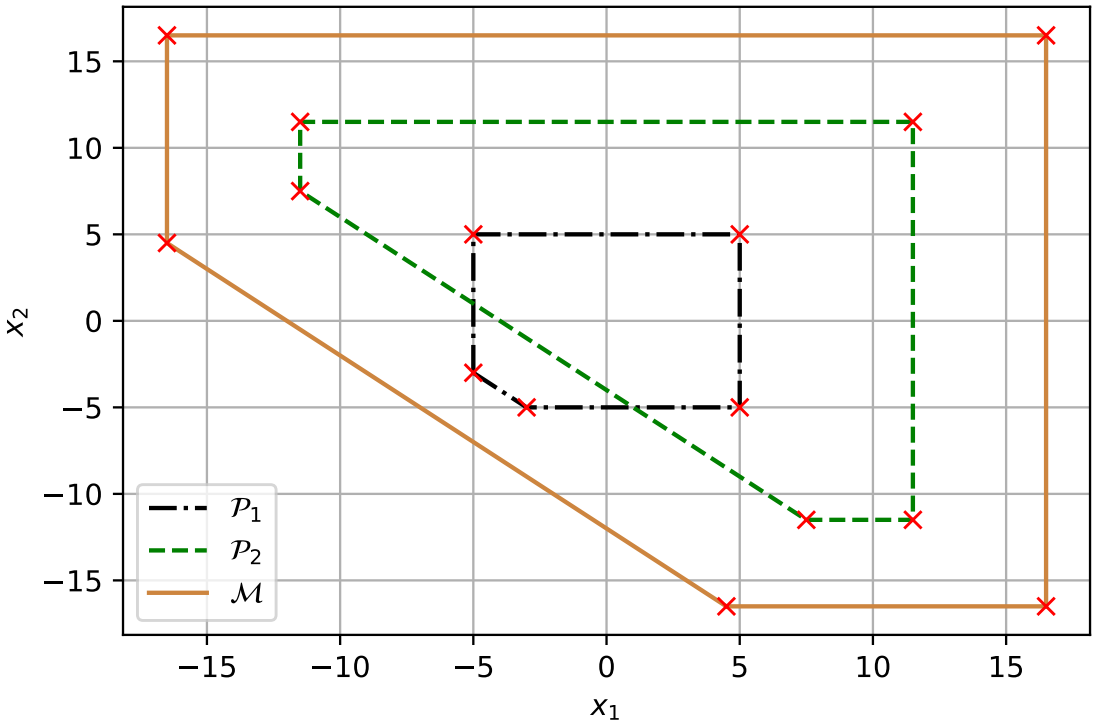

Figure 2.1.: Polytopes $\mathcal{P}_1$ and $\mathcal{P}_2$ in dashed green and dashed-dotted black, and their Minkowski sum $\mathcal{M}$ in solid orange. The vertices are shown as red crosses.

sible combinations of polytope vertices lead to a vertex of the Minkowski sum and must therefore be excluded. Note that the method described above can only be applied if the polytopes are available in vertex representation. If this is not the case, all polytopes must first be converted from the half-space to the vertex representation. Recall, however, that this conversion is already classified as an NP-hard problem.

A general difficulty associated with calculating the Minkowski sum arises from the increase in dimensionality (time periods), which leads to a significant rise in the number of vertices in the Minkowski sum compared to the number of vertices in each individual set. This phenomenon is illustrated in Fig. 2.2, where the average number of vertices of 50 polytopes is plotted alongside the number of vertices in the Minkowski sum across five time periods. It can be seen that the average number of vertices in the individual polytopes and the number of vertices in the aggregated polytope differ significantly. While the average number of vertices in the individual polytopes remains below 100 across five time periods, the aggregated polytope contains more than 500 vertices. This substantial increase in the number of vertices further complicates the computation of the Minkowski sum, especially considering that even simple polytopes, such as hypercubes, have 2^d vertices in d-dimensional

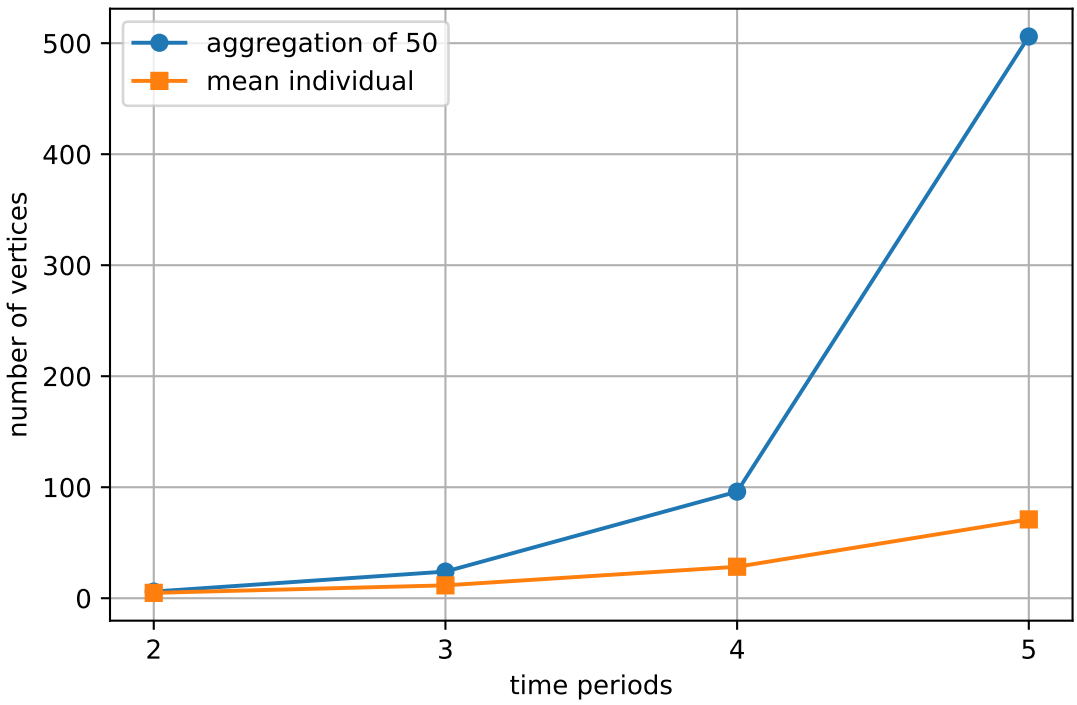

Figure 2.2.: Number of vertices with increasing time periods: The orange line represents the average number of vertices in 50 polytopes, while the blue line shows the number of vertices in the Minkowski sum of these 50 polytopes.

space. The computational complexity is illustrated in an example below for the case of a hypercube, highlighting the immense increase in computation.

Example 1 (Hypercube)**.** Assume n convex storage models (cf. (2.17)) with power constraints only, i.e., no energy constraints. This results in hypercubes with matrix dimensions $2d \times d$. In power systems, the half-space representation is typically given. Therefore, the hypercubes must first be converted to the vertex representation in order to compute the Minkowski sum. The following steps outline the computational requirements based on the method described above:

- Solve $n\binom{2d}{d}$ systems of linear equations (to compute all intersections of half-planes).
- Solve $n\left(\binom{2d}{d} - 2^d\right)$ LPs to remove redundant points (as hypercubes have 2^d vertices).
- Compute 2^{nd} additions.
- Solve 2^{nd} LPs to remove redundant points.

Note that the second step is included for completeness. For hypercubes, the first step already results in 2^d intersections instead of $\binom{2d}{d}$, because the remaining systems are infeasible. Therefore, the second step is not necessary.

The example above demonstrates the numerical complexity for a simple case. It is evident that these computations are not feasible, particularly for a day-ahead decision with $d = 96$ and large values of n. Note that convex storage models with energy constraints typically have more than 2^d vertices, further complicating the computations.

Yet, alternative approaches exist for accurately quantifying the aggregated flexibility. Mukhi and Abate, 2023 introduce a method for aggregating simple EV models, revealing that the feasible regions of these models take the form of permutahedra. This enables the identification of vertices within each device, which can then be used to calculate the aggregated feasible region.

Similarly, Trangbaek et al., 2011 propose a recursive strategy to compute the vertices of a simple storage unit by determining extreme bounds. The corresponding vertices are subsequently combined to quantify the aggregated flexibility. However, this approach suffers from combinatorial complexity, as it attempts to compute all vertices using a scheme that may lead to redundant calculations. Trangbaek and Bendtsen, 2012 enhance and extend this approach, deriving a half-space representation of the aggregated flexibility.

Finally, Wen et al., 2022 develop a technique for calculating the aggregated flexibility of simple storage models based on the Fourier-Motzkin elimination.[4] They first describe the Minkowski sum in higher-dimensional space, followed by the elimination of variables using the Fourier-Motzkin method. The repeated elimination of variables projects the Minkowski sum into lower dimensions, resulting in the aggregated flexibility. In addition, this approach can compute outer approximations if not all calculations are completed.

Generally speaking, the computation of the Minkowski sum is considered, however, an NP-hard problem (Tiwary, 2008), and available methods either face similar computational challenges or are limited to a narrow range of applications.

2.4. Summary

This chapter provided an in-depth examination of various flexible devices in power systems and their modeling techniques. We demonstrated that devices such as BESS, TCL, EV, and PHES can be effectively represented using a convex storage model. Table 2.1 guides the reader in calculating the convex storage parameters from specific parameters. The chapter also explored fundamental polytope representations, including half-space and vertex representations, as well as their transformations. We discussed the concept of the aggregated flexibility, which involves the Minkowski sum of individual flexibility sets, and analyzed its computational complexity in detail.

[4] Fourier–Motzkin elimination, also known as the FME method, is a mathematical algorithm for eliminating variables from a system of linear inequalities. It is used to reduce an n-variable problem to an equivalent $(n-1)$-variable problem, similar to Gaussian elimination but for a system of inequalities.

3. Approximation Strategies

As the exact quantification of the aggregated flexibility is numerically intractable, various approximate quantification strategies have been proposed in the literature. This diversity makes it challenging for applicants to select the most suitable method for a particular scenario. In addition, many existing approximations are only accessible via published papers, which requires programming for practical application. Therefore, a comprehensive open-source benchmark for approximation strategies is presented in this chapter. State-of-the-art approximation techniques are implemented and evaluated with regard to their accuracy and computational complexity using real data. The aim of this chapter is threefold:

- First, to provide the code for state-of-the-art approximation techniques.
- Second, to assist the reader in selecting the most suitable algorithm for their specific application.
- Third, to highlight the shortcomings of existing approximation strategies.

The results of this chapter have been published (Öztürk et al., 2022).

3.1. Overview of Approximation Strategies

Approximation strategies identified in the literature can be classified into two categories: inner and outer approximations. Outer approximations are supersets of the Minkowski sum, generally overestimating the feasible region, while inner approximations are subsets that typically underestimate it. Additionally, a distinction is made between top-down and bottom-up approaches. Top-down approaches estimate the Minkowski sum directly, whereas bottom-up approaches estimate each set separately and then sum the approximated sets to derive an approximation of the Minkowski sum.

The approximations revisited henceforth are organized in a mathematically coherent order: first, the summation-based approximations; followed by the zonotope and homothet-based approximations; and finally, the ellipsoid-based approximations. Further details about the algorithms can be found in the referenced literature.

3.1.1. Outer Approximation by Right-Hand Summation

Barot, 2017; Barot and Taylor, 2017 propose a top-down outer approximation strategy. This approach considers n flexibilities over d time periods, each characterized by an identical matrix $A \in \mathbb{R}^{m \times d}$ and individual vectors $b_i \in \mathbb{R}^m$, $i = 1, \ldots, n$. The aggregated flexibility can then be approximated by a polytope defined by the matrix A and the sum of the individual right-hand vectors, i.e.,

$$\left\{ x \in \mathbb{R}^d : Ax \leq \sum_{i=1}^{n} b_i, \right\}.$$

This procedure entails $m(n-1)$ summations, where m denotes the number of rows in the matrix A. To enhance accuracy, a preconditioning (PC) step can be implemented. This involves modifying the right-hand vectors prior to summation. For the k$^{\text{th}}$ row of A (denoted

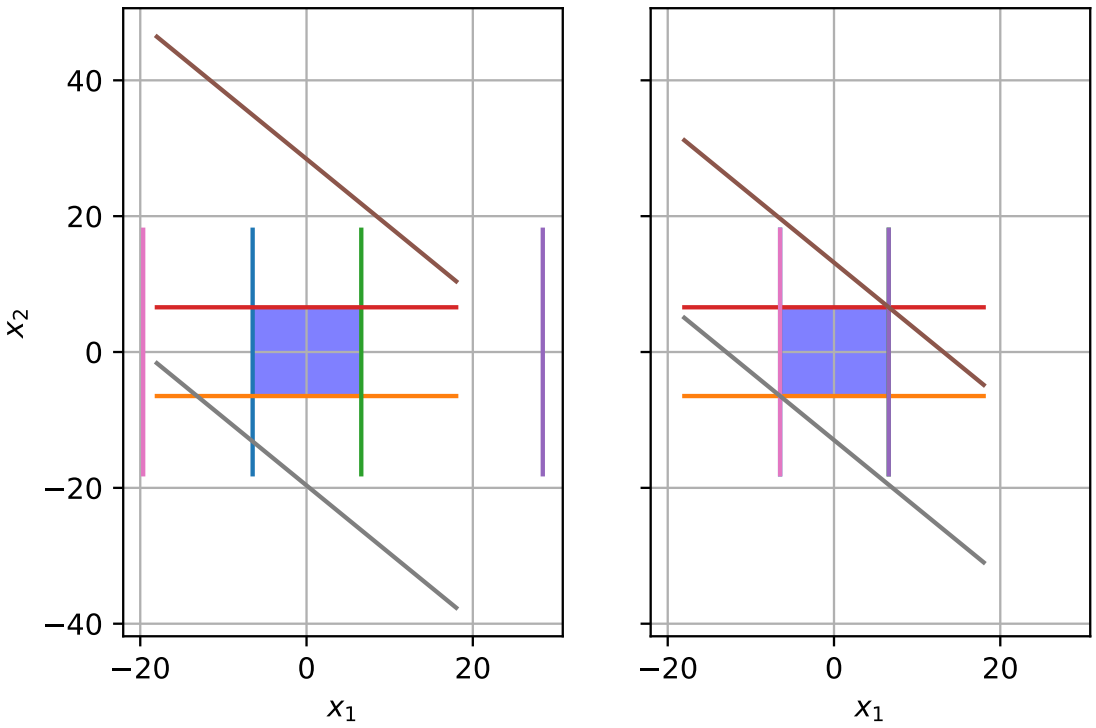

Figure 3.1.: Illustration of the preconditioning process. The polytope is represented in violet. The half-planes of the polytope are shown on the left before preconditioning and on the right after preconditioning.

as a_k), the k$^{\text{th}}$ entry of the right-hand vector b is recomputed by solving

$$b_k = \max_x \; a_k^\top x \tag{3.1a}$$

$$\text{subject to } Ax \leq b. \tag{3.1b}$$

This recomputation shrinks the right-hand vector while preserving the feasible region, cf. Fig. 3.1.

For n devices, the process requires solving mn LPs, each with m constraints and d variables. The communication effort consists of transmitting two components to the auxiliary services purchaser: the matrix A and the vector $\sum_{i=1}^{n} b_i$ representing the right-hand summation. Finally, note that when the matrices differ, redundant constraints can be added to ensure uniformity; thus, this approach is also applicable to devices with different A matrices.

3.1.2. Inner Approximation by Zonotopes

Müller et al., 2019, 2015 employ a bottom-up approach based on zonotopes. Zonotopes are centrally symmetric polytopes described by

$$\mathcal{Z}(G, \nu, \bar{\lambda}) = \left\{ x \in \mathbb{R}^d : x = \nu + G\lambda, -\bar{\lambda} \leq \lambda \leq \bar{\lambda} \right\},$$

where G is a matrix with normalized column vectors serving as generating directions, λ is a vector of scaling factors, and $\nu \in \mathbb{R}^d$ is the zonotope center. Following the approach of Müller et al., 2015, we choose the d unit vectors along with additional $d-1$ vectors of the form

$$\left(0, \ldots, -\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, \ldots, 0\right)^\top$$

as generating directions, i.e., $G \in \mathbb{R}^{d \times (2d-1)}$ and $\lambda \in \mathbb{R}^{2d-1}$. To determine the matrix F of zonotope normals, at most $d^2 + d$ normals must be computed; thereafter, the offset vector u is computed by shifting the zonotope hyperplanes to ensure that each one supports a given polytope $\mathcal{P} = \{x \in \mathbb{R}^d : Ax \leq b\}$. This step is analogous to (3.1) and requires the solution of at most $d^2 + d$ LPs per device. The final step is to find inner approximating zonotopes with respect to a given objective. The constraint that enforces a zonotope to be a subset of a polytope $\mathcal{P}$ is provided in Müller et al., 2019 as

$$\mathcal{Z}(G, \nu, \bar{\lambda}) \subseteq \mathcal{P} \iff A\nu + |AG|\bar{\lambda} \leq b$$

where $|AG|$ is the elementwise absolute value of the matrix AG. Müller et al., 2015 finds optimal inner zonotopes by minimizing the distance between the offsets in a given norm

$$\min_{v,\bar{\lambda}} \|u - \delta_Z(v, \bar{\lambda})\|_\ell \tag{3.2a}$$

$$\text{subject to } Av + |AG|\bar{\lambda} \leq b \tag{3.2b}$$

$$\bar{\lambda} \geq 0, \tag{3.2c}$$

with $\ell \in \{1, 2, \infty\}$. Here, u is the previously calculated polytope offset vector, and

$$\delta_Z(v, \bar{\lambda}) = \begin{pmatrix} F \\ -F \end{pmatrix} v + \begin{pmatrix} |FG| \\ |FG| \end{pmatrix} \bar{\lambda}$$

is the zonotpe offsets in terms of the center v and the vector of scaling limits $\bar{\lambda}$. Müller et al., 2019 propose a slightly different approximation where problem (3.3) is solved instead of (3.2)

$$\max_{v,\bar{\lambda}} \; w^\top \bar{\lambda} \tag{3.3a}$$

$$\text{subject to } Av + |AG|\bar{\lambda} \leq b \tag{3.3b}$$

$$\bar{\lambda} \geq 0, \tag{3.3c}$$

with the weight $w = \frac{2}{d^2+d}\tilde{u}^\top |FG|$, where $\tilde{u}$ is the element-wise reciprocal of u. The different strategies, each with a distinct objective function, are illustrated in Fig. 3.2. In the top left, the approach using the ℓ_∞ norm is shown; in the top right, the approach using the ℓ_1 norm; in the bottom left, the approach with the ℓ_2 norm; and in the bottom right, the weighted approach is presented.

Finally, the Minkowski sum of zonotopes can be efficiently calculated by summing the v_i and the $\bar{\lambda}_i$ for $i = 1, \ldots, n$ to obtain an inner approximation of the aggregated flexibility, i.e,

$$\mathcal{Z}\left(G, \sum_{i=1}^{n} v_i, \sum_{i=1}^{n} \bar{\lambda}_i\right).$$

A maximum of $(d^2+d)n$ LPs with m constraints and d variables must be solved. In addition, n LPs with $2d^2 + 4d + m - 1$ constraints and $d^2 + 4d - 1$ variables for ℓ_1 norm, n LPs with $2d^2 + 4d + m - 1$ constraints and $3d$ variables for ℓ_∞ norm, n convex programs (CPs) with $2d - 1 + m$ constraints and $3d - 1$ variables for ℓ_2 norm and n LPs with $2d - 1 + m$ constraints and $3d - 1$ variables for the weighted approach, cf. (3.3) are solved. Note that we have used

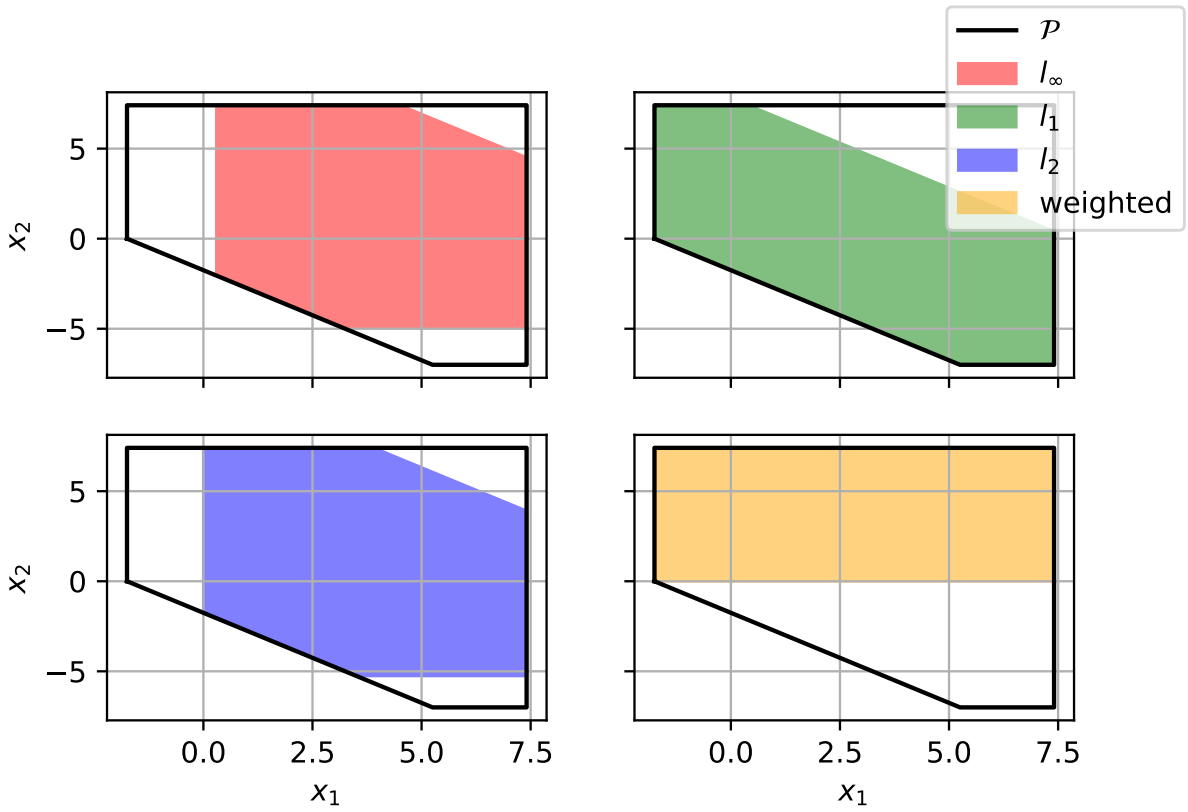

Figure 3.2.: Illustration of zonotope-based approximations. The top left shows the objective with the ℓ_∞ norm, the top right with the ℓ_1 norm, the bottom left with the ℓ_2 norm, and the bottom right illustrates the weighted approach.

the fact that the problems involving the ℓ_1 and ℓ_∞ norms can be reformulated as LPs.

The communication effort consists of transmitting the matrix of generators G, the sum of zonotope centers v_i, and the sum of scaling limits $\bar{\lambda}_i$ to the auxiliary services purchaser.

3.1.3. Inner Approximation by Cuboid Homothets

Nazir et al., 2018 employ a bottom-up strategy based on homothets to compute an inner approximation of the aggregate flexibility. Given a compact[5] convex set $\mathcal{P}_0$, a homothet of $\mathcal{P}_0$ is defined as the set

$$\beta\mathcal{P}_0 + t := \left\{x \in \mathbb{R}^d : x = \beta\zeta + t,\ \zeta \in \mathcal{P}_0\right\},$$

[5] A set in $\mathbb{R}^d$ is said to be compact if it is both bounded and closed.

with a scaling factor $\beta > 0$ and a translation vector $t \in \mathbb{R}^d$; that is, a homothet of $\mathcal{P}_0$ is a scaled and translated version of $\mathcal{P}_0$. Following Nazir et al., 2018, the maximum volume cuboid that fits in a selected polytope $\mathcal{P} = \{x \in \mathbb{R}^d : Ax \leq b\}$ is defined as the prototype set $\mathcal{P}_0$ by solving

$$\max_{x^+,x^-} \prod_{k=1}^{d} x_k^+ - x_k^- \tag{3.4a}$$

$$\text{subject to } A^+x^+ - A^-x^- \leq b \tag{3.4b}$$

$$x_k^- \leq x_k^+, \quad \text{for } \; k = 1, \ldots, d, \tag{3.4c}$$

where $A_{ij}^+ = \max(0, A_{ij})$ and $A_{ij}^- = \max(0, -A_{ij})$. The objective (3.4a) can be equivalently written as

$$\max_{x^+,x^-} \sum_{k=1}^{d} \log(x_k^+ - x_k^-),$$

which leads to a CP. Subsequently, the edge distances $\delta_k^0 = (x_k^+ - x_k^-)$ for $k \in \{1, 2, \ldots, d\}$ and the distance ratios $\rho_{1,k}^0 = \delta_1^0/\delta_k^0$ for $k \in \{2, \ldots, d\}$ are calculated. The following optimization problem finds then the maximum volume homothet of $\mathcal{P}_0$ in a given polytope $\mathcal{P}$

$$\max_{x^+,x^-} \prod_{k=1}^{d} x_k^+ - x_k^- \tag{3.5a}$$

$$\text{subject to } A^+x^+ - A^-x^- \leq b \tag{3.5b}$$

$$x_k^- \leq x_k^+, \quad \text{for } \; k = 1, \ldots, d \tag{3.5c}$$

$$(x_1^+ - x_1^-) = \rho_{1,k}^0 (x_k^+ - x_k^-), \quad \text{for } \; k = 2, \ldots, d. \tag{3.5d}$$

Note that this procedure can be applied in multiple stages to cover more flexibility: In stage zero, a maximum volume homothet of $\mathcal{P}_0$ is fitted into $\mathcal{P}$. In stage one, maximum volume homothets of $\mathcal{P}_0$ are fitted into regions not covered by the stage zero solution. This is done by extending matrix A and vector b by a row of the half-space inequalities of the stage zero solution multiplied by -1. Each stage s requires the solution of at most $(2d)^s$ optimization problems, leading to a total of at most

$$\sum_{i=0}^{s} (2d)^i = \frac{(2d)^{s+1} - 1}{(2d) - 1}$$

CPs per device. The procedure is shown for stages zero and one in Fig. 3.3.

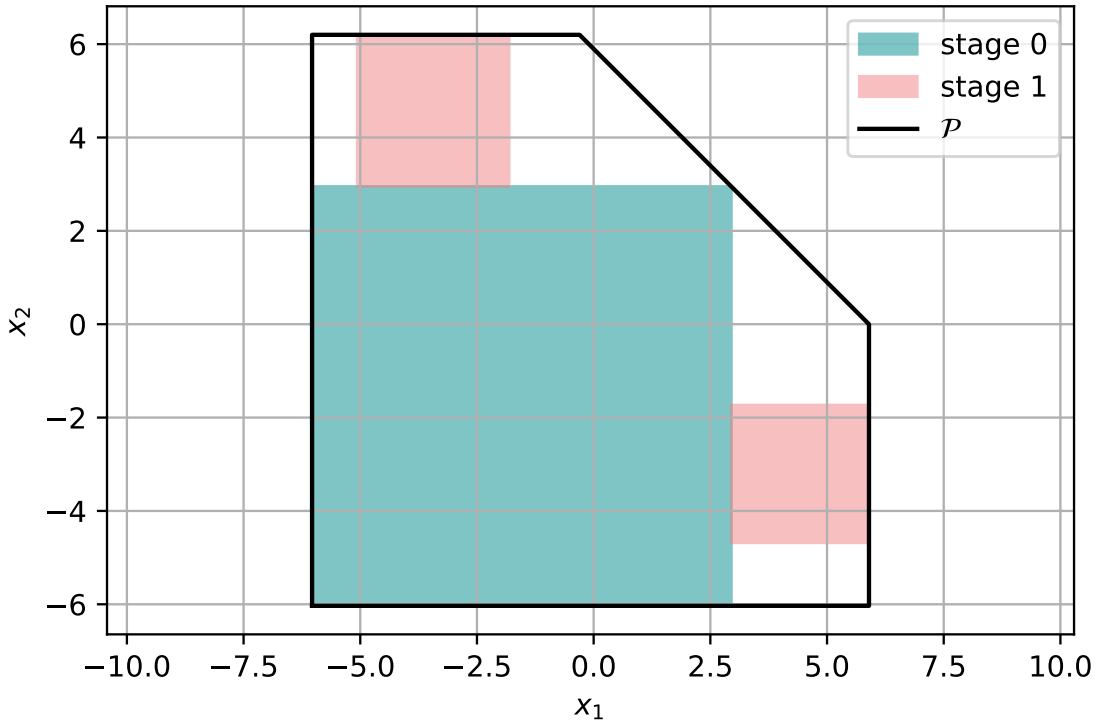

Figure 3.3.: Inner approximation with cuboid homothets. Solid line: feasible region of a storage model, turquoise area: stage zero solution, bright red areas: stage one solutions.

Finally, the distributive property of the Minkowski sum (cf. Property 4) is applied to derive an inner approximation of the aggregated flexibility. Rather than first forming the union and then calculating the Minkowski sum, the Minkowski sum is computed first, followed by the union. The Minkowski sum of cuboid homothets is calculated by adding the right-hand vectors in the half-space representation. Nazir et al., 2018 propose to use corresponding combinations for axis-aligned cuboids, i.e., the k^{th} cuboid of the s^{th} stage in $\mathcal{P}_i$ is added to the k^{th} cuboid of the s^{th} stage in $\mathcal{P}_l$.

The total computational effort consists of solving $n(2d)^0$ CPs with $(m + 2d - 1)$ constraints and $2d$ variables for stage 0, $n(2d)^1$ CPs with $(m + 2d - 1 + 1)$ constraints and $2d$ variables for stage 1, and up to $n(2d)^s$ CPs with $(m + 2d - 1 + s)$ constraints and $2d$ variables for stage s. In addition, one CP with $m + d$ constraints and $2d$ variables has to be solved.

The communication effort consists of sending the matrix $(I, -I)^\top \in \mathbb{R}^{2d \times d}$, the vector $((x^+)^\top, -(x^-)^\top)^\top \in \mathbb{R}^{2d}$ of problem (3.4), and a maximum of $(2d)^{s+1} - 1)(2d - 1)^{-1}$ scaling factors β_k and translations t_k to the purchaser of auxiliary services.

3.1.4. Inner and Outer Approximation by Battery Homothets

Zhao et al., 2017 employ a battery model (cf. (2.19)) for the prototype set $\mathcal{P}_0$ to obtain a bottom-up, homothet-based inner approximation. The prototype matrix A_p and the right-hand vector b_p are selected based on the average of individual battery parameters. To find the maximum volume homothet of $\mathcal{P}_0$ in a polytope $\mathcal{P} = \{x \in \mathbb{R}^d : Ax \leq b\}$ one solves

$$\max_{\beta,t} \beta \tag{3.6a}$$

$$\text{subject to } \beta\mathcal{P}_0 + t \subset \mathcal{P} \tag{3.6b}$$

$$\beta > 0. \tag{3.6c}$$

Zhao et al., 2017 show that the solution to this problem is given by $\beta^\star = \frac{1}{s^\star} \in \mathbb{R}$ and $t^\star = -\frac{r^\star}{s^\star} \in \mathbb{R}^d$ if $(s^\star, G^\star, r^\star) \in \mathbb{R} \times \mathbb{R}^{m\times m} \times \mathbb{R}^d$ is the minimizer to

$$\min_{s,G,r} s \tag{3.7a}$$

$$\text{subject to } GA_p = A \tag{3.7b}$$

$$Gb_p \leq sb + Ar \tag{3.7c}$$

$$s > 0,\ G \geq 0, \tag{3.7d}$$

where $G \geq 0$ is the element-wise inequality. To obtain a battery homothet outer approximation, one solves the problem

$$\min_{\beta,t} \beta \tag{3.8a}$$

$$\text{subject to } \beta\mathcal{P}_0 + t \supset \mathcal{P} \tag{3.8b}$$

$$\beta > 0. \tag{3.8c}$$

This can be reformulated as

$$\min_{\beta,G,t} \beta \tag{3.9a}$$

$$\text{subject to } GA_p = A \tag{3.9b}$$

$$Gb \leq \beta b_p + At \tag{3.9c}$$

$$\beta > 0,\ G \geq 0. \tag{3.9d}$$

Both the inner and outer approximation strategies are shown in Fig. 3.4 along with a storage

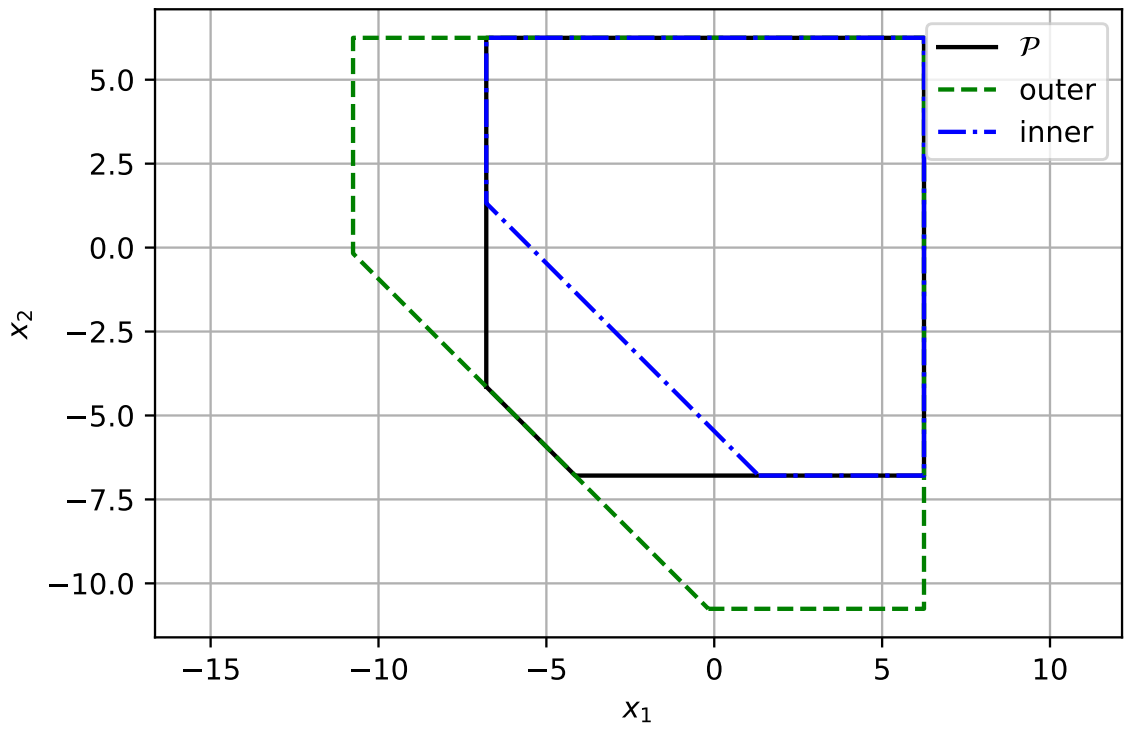

Figure 3.4.: Illustration of the approximation strategies based on battery homothets. The inner approximation is shown in dashed-dotted blue, while the outer approximation is depicted in dashed green.

model $\mathcal{P}$. Once an approximation for all $\mathcal{P}_i$, $i = 1, \ldots, n$, is obtained, the Minkowski sum of homothets is computed by simply adding the scaling factors and offsets (cf. Property 3) to obtain an approximation to the aggregated flexibility, i.e.,

$$\left(\sum_{i=1}^{n} \beta_i\right) \mathcal{P}_0 + \sum_{i=1}^{n} t_i.$$

The overall computation effort is given by solving n LPs with $2m^2 + m + 1$ constraints and $m^2 + d + 1$ variables.

The communication effort consists of transmitting the matrix A_p, the vector b_p, the sum of scaling factors β_i, and translations t_i to the purchaser of the auxiliary services.

3.1.5. Inner Approximation by Battery Homothet Projection

Zhao et al., 2016 employ a top-down approach to obtain an inner approximation based on a homothet projection technique. The Minkowski sum of polytopes $\mathcal{P}_i$, $i = 1, \ldots, n$ can be implicitly written as

$$\left\{ z \in \mathbb{R}^d : \begin{pmatrix} A_n & -A_n & -A_n & \ldots & -A_n & -A_n \\ 0 & A_1 & 0 & \ldots & 0 & 0 \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & \ldots & A_{n-2} & 0 \\ 0 & 0 & 0 & \ldots & 0 & A_{n-1} \end{pmatrix} \begin{pmatrix} z \\ x_1 \\ \vdots \\ x_{n-2} \\ x_{n-1} \end{pmatrix} \leq \begin{pmatrix} b_n \\ b_1 \\ \vdots \\ b_{n-2} \\ b_{n-1} \end{pmatrix} \right\}, \tag{3.10}$$

where $z = \sum_{i=1}^{n} x_i$. We denote the left matrix as $Q \in \mathbb{R}^{mn \times dn}$ and the right vector as $p \in \mathbb{R}^{mn}$. The battery model derived in Section 2.1.1 is used as the prototype set $\mathcal{P}_0$, where the prototype matrix A_p and right-hand vector b_p are calculated using the average of the individual battery parameters. For a fixed vector $u_0 \in \mathbb{R}^{d(n-1)}$ let

$$\tilde{\mathcal{P}}_0 = \left\{ (x, u_0)^\top \in \mathbb{R}^{dn} : A_p x \leq b_p \right\}$$

denote the corresponding lift of $\mathcal{P}_0$ in dn-dimensional space. The problem of finding a maximal homothet of $\tilde{\mathcal{P}}_0$ in $\mathcal{P} = \{x \in \mathbb{R}^{dn} : Qx \leq p\}$ is formulated as

$$\max_{\beta, t, u_0} \beta \tag{3.11a}$$

$$\text{subject to } \beta \tilde{\mathcal{P}}_0 + \tilde{t} \subset \mathcal{P} \tag{3.11b}$$

$$\beta > 0, \tag{3.11c}$$

where $\tilde{t} = (t^\top, \mathbf{0}_{d(n-1)}^\top)^\top$ is the lift of t in dn dimensional space by setting the additional values to zero. This problem is transformed by the authors to the following LP

$$\min_{s, G, r, u_0} s \tag{3.12a}$$

$$\text{subject to } GA_p = \begin{pmatrix} Q_{11} \\ Q_{21} \end{pmatrix} \tag{3.12b}$$

$$Gb_p \leq Q \begin{pmatrix} r \\ -u_0 \end{pmatrix} + sp \tag{3.12c}$$

$$s > 0, \; G \geq 0, \tag{3.12d}$$

where $s = \frac{1}{\beta}$, $r = -\frac{t}{\beta}$, $Q_{11} \in \mathbb{R}^{m \times d}$, and $Q_{21} \in \mathbb{R}^{(m-1)d \times d}$ are the sub matrices in Q corresponding to the first d columns, and $G \geq 0$ indicates elementwise inequality.

The above formulation is conservative since only solutions contained in the homothet of $\tilde{\mathcal{P}}_0$ are allowed. However, for the approximation only the following condition is required:

$$\forall\, u \in \beta\mathcal{P}_0 + t\ \exists\, \tilde{u}(u) \ \text{ such that } \ (u, \tilde{u}(u)) \in \mathcal{P}$$

This can be cast into a robust optimization problem. The function $\tilde{u}(u)$ is used as a decision rule which is assumed to be linear, i.e.,

$$\tilde{u}(u) = Wu + V.$$

Using these ideas, the problem is restated as

$$\min_{s,G,r,W,V} \ s \tag{3.13a}$$

$$\text{subject to } GA_p = Q\begin{pmatrix} I \\ W \end{pmatrix} \tag{3.13b}$$

$$Gb_p \leq Q\begin{pmatrix} r \\ -V \end{pmatrix} + sp \tag{3.13c}$$

$$s > 0,\ G \geq 0, \tag{3.13d}$$

where $I \in \mathbb{R}^{d \times d}$ is the identity matrix. Upon solving this problem, the approximation is simply given by

$$\beta\mathcal{P}_0 + t.$$

The approximation, denoted as $\mathcal{A}$, is shown in Fig. 3.5 along with the Minkowski sum of storage models, denoted as $\mathcal{M}$.

The computation effort consists of solving a LP with $nmd + nm + nm^2 + 1$ constraints and $nm^2 + d + (n-1)(d^2 + d) + 1$ variables.

The matrix $A_p \in \mathbb{R}^{m \times n}$, the vector $b_p \in \mathbb{R}^m$, the scaling factor $\beta \in \mathbb{R}$, and the offset $t \in \mathbb{R}^d$ must be transmitted to the purchaser of the auxiliary services.

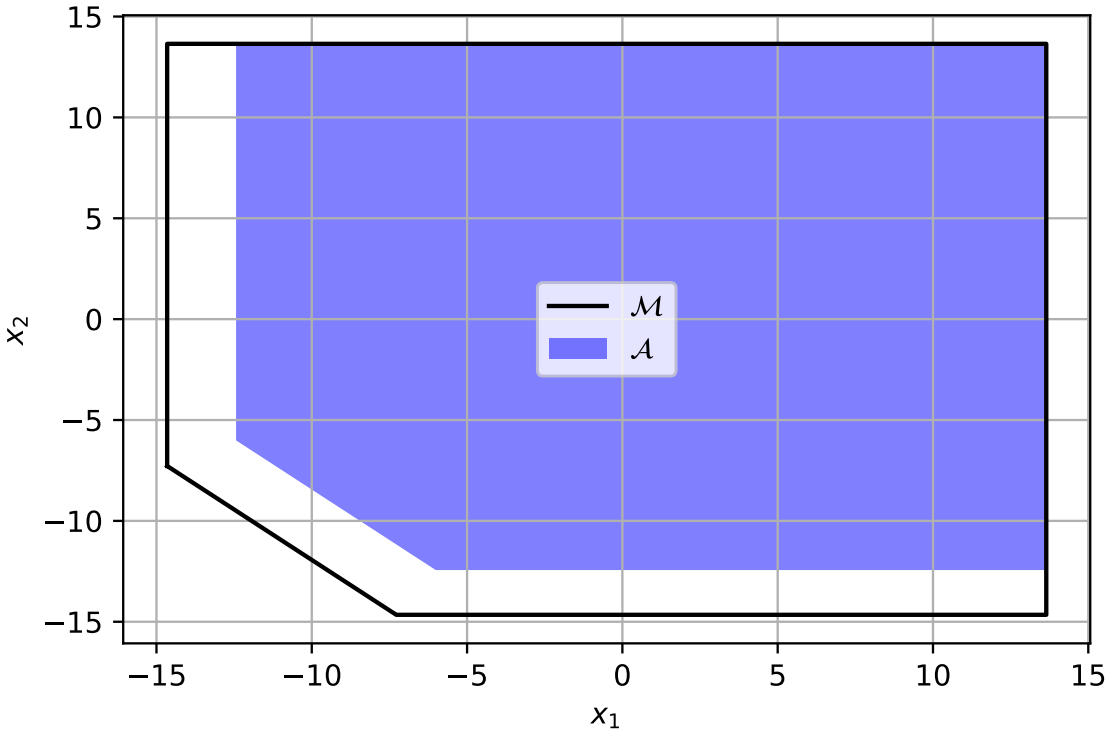

Figure 3.5.: Illustration of the homothet projection strategy. The approximation is shown in violet, while the Minkowski sum of storage models is represented in solid black.

3.1.6. Inner Approximation by Ellipsoid Projection

The main idea of this approach, as presented by Barot, 2017, is to fit an ellipsoid with maximum volume in the implicitly described higher-dimensional Minkowski sum (3.10) with matrix Q and right-hand vector p.

An ellipsoid can be expressed as the image of the unit ball under an affine transformation, i.e.,

$$\{Gu + h : \|u\|_2 \leq 1\},$$

where the positive definite matrix $G \in \mathbb{R}^{nd \times nd}$ and the center $h \in \mathbb{R}^{nd}$ are partitioned as

$$G = \begin{pmatrix} G_z & G_{xz} \\ G_{xz}^\top & G_x \end{pmatrix}, \; h = \begin{pmatrix} h_z \\ h_x \end{pmatrix},$$

with $x = (x_1^\top, \ldots, x_{n-1}^\top)^\top$.

The volume of an ellipsoid is proportional to the determinant of G, hence the problem of finding an ellipsoid with maximum volume in the implicitly described Minkowski sum can be formulated as

$$\max_{G,h} \ \log(\det(G)) \tag{3.15a}$$

$$\text{subject to } \|Gq_i\|_2 + q_i^\top h \leq p_i \ \text{ for } \ i = 1, \ldots, mn \tag{3.15b}$$

$$G \succ 0, \tag{3.15c}$$

where q_i is the i$^{\text{th}}$ row in Q and the notation $G \succ 0$ denotes a positive definite matrix.

Barot, 2017 proposes to maximize the determinant of the submatrix $G_z \in \mathbb{R}^{d \times d}$ associated with the Minkowski sum rather than that of G, leading to the following semidefinite program (SDP)

$$\max_{G,h} \ \log(\det(G_z)) \tag{3.16a}$$

$$\text{subject to } \|Gq_i\|_2 + q_i^\top h \leq p_i \ \text{ for } \ i = 1, \ldots, mn \tag{3.16b}$$

$$G \succ 0. \tag{3.16c}$$

Solving this problem yields an ellipsoid in the z-x-space which must be projected back to the z-space. Once problem (3.16) is solved, the approximation is given by

$$\{G_z u + h_z : \|u\|_2 \leq 1\}. \tag{3.17}$$

The computation effort consists of solving one SDP with mn constraints and $(mn)^2 + dn$ variables.

The communication effort consists of transmitting the matrix G_z and the ellipsoid center h_z to the purchaser of the auxiliary services.

3.1.7. Inner Approximation by Ellipsoid Projection with Linear Decision Rule

Another top-down strategy based on ellipsoid projection technique is described by Zhen and den Hertog, 2018. They define the inclusion constraint solely in the z-space

$$\max_{h_z, y, G_z} \ \log(\det(G_z)) \tag{3.18a}$$

$$\text{subject to } \forall\ u \in \mathcal{B}^d\ \exists\ y \in \mathbb{R}^{d(n-1)} : Q \begin{pmatrix} h_z + G_z u \\ y \end{pmatrix} \leq p, \tag{3.18b}$$

where $\mathcal{B}^d = \{u \in \mathbb{R}^d : \|u\|_2 \leq 1\}$ is the d-dimensional unit ball, and Q and p are the matrix and right-hand vector in the implicitly described Minkowski sum (3.10). This problem can be interpreted as a robust optimization problem, where $h_z \in \mathbb{R}^d$ and $G_z \in \mathbb{R}^{d \times d}$ are the here and now decision variables, and $y \in \mathbb{R}^{d(n-1)}$ the wait and see variable. Assuming a linear decision rule (LDR)

$$y = Vu + w,$$

the problem above can be reformulated as

$$\max_{h_z, w, G_z, V} \ \log(\det(G_z)) \tag{3.19a}$$

$$\text{subject to } \left\| \begin{pmatrix} G_z \\ V \end{pmatrix}^\top q_i \right\|_2 + q_i^\top \begin{pmatrix} h_z \\ w \end{pmatrix} \leq p_i \ \text{ for } \ i = 1, \ldots, mn \tag{3.19b}$$

$$G_z \succ 0. \tag{3.19c}$$

Solving this problem yields an ellipsoid in z-space, which serves as an inner approximation of the aggregate flexibility. The approximation itself is, similar to the previous approach, given by (3.17). Figure 3.6 depicts both the projected method discussed in the previous section and the enhanced version introduced in this section, within the Minkowski sum of storage models, denoted as $\mathcal{M}$.

The overall computation effort consists of solving a SDP with mn constraints and $d^2n + dn$ variables.

The center h_z and the matrix G_z need to be transmitted to the purchaser of the auxiliary services.

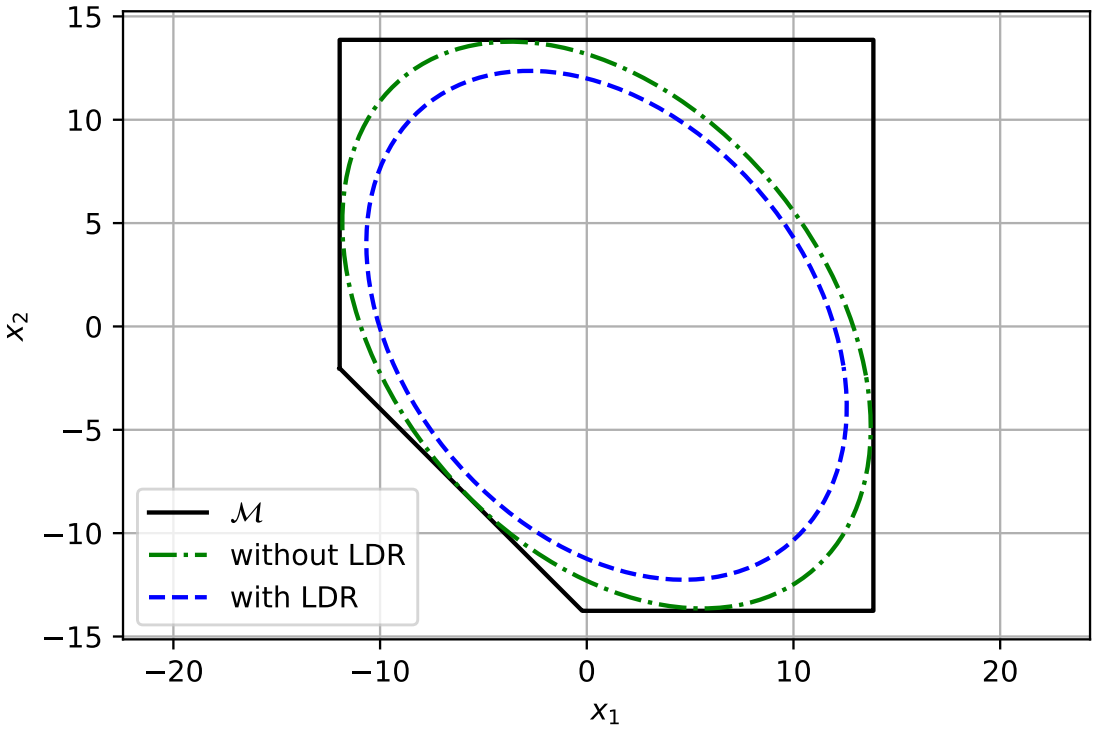

Figure 3.6.: Illustration of the maximal volume ellipsoid approximation methods. The projected ellipsoid is shown in dashed blue, the improved version with the LDR formulation in dashed-dotted green, and the Minkowski sum of polytopes in solid black.

3.1.8. Further Approximation Strategies

We briefly review other aggregation strategies that do not fall within the categories outlined above.

An alternative inner approximation is given by Appino et al., 2021, which initially add the individual power and energy constraints, resulting in an outer approximation. In a second step, the approximation of the aggregated flexibility is constructed, whereby feasible disaggregation is guaranteed.

A top-down technique that employs Markov chains to evaluate the aggregated flexibility of a collection of TCLs is proposed by Koch et al., 2011. The authors use a Linear Time-Invariant (LTI) state space model to represent a collection of TCLs. The transition matrix is derived analytically and compared to an identified matrix obtained through a Markov chain construction process. The overall control mechanism is then implemented through a Model

Predictive Control (MPC) framework and tested in use cases with up to 1000 TCLs.

Another statistical approach is presented by Sajjad et al., 2016. This data-driven method does not rely on specific models of flexible devices. The authors employ a Monte Carlo simulation to generate aggregated load profiles. Statistical analysis is then conducted using the maximum likelihood estimator, binomial distribution, and the central limit theorem to characterize the aggregated flexibility. Use cases are presented for up to 150 individuals, with one day divided into intervals of 10, 15, and 30 minutes.

Nieße et al., 2014 propose an agent-based approach for the dynamic aggregation of flexibilities. In this framework, agents manage decentralized energy units and can form coalitions to combine the capabilities of the units they oversee. Initially, agents create individual product portfolios. In the second step, neighborhoods are formed based on a distance function. Then, the neighborhoods begin to form coalitions to achieve a global goal. Finally, after the coalition product is accepted on the market, agents divide the payoff received from the product. The authors use the Combinatorial Optimization Heuristic for Distributed Agents (COHDA) procedure to solve their scheduling problems. Additionally, a support vector classifier is used to estimate the feasible regions, and a decoder to compute feasible operation schedules. The proposed strategy is then tested within a smart grid simulation framework using Mosaik.

Techniques that characterize the flexibility of heterogeneous storage units using the so-called E-p transformation date bake to Evans, Tindemans, and Angeli, 2019; Evans, Tindemans, Angeli, and Strbac, 2019; Evans et al., 2020. The authors demonstrate this technique in various use cases, such as deriving the optimal policy to minimize unserved energy during supply shortfall events.

3.2. Comparative Benchmarking

This section compares state-of-the-art approximation algorithms, starting with a summary of their computational and communication costs, which are provided in a table for detailed evaluation and analysis. Next, a comprehensive open-source benchmark for Minkowski sum approximations is introduced, which is used to evaluate the described approximation strategies. The benchmark also highlights identified shortcomings, discusses associated challenges, and points to potential research opportunities. The code for the benchmark and all referenced algorithms are available (Rheinberger, 2021b).

3.2.1. Overview of Communication and Computation Effort

In this section, we evaluate the overall communication and computational efforts of the presented algorithms, with a specific focus on the storage model (2.16), i.e., the number of rows, m, in A is $4d$. Table 3.1 shows the overall computation and communication efforts of the algorithms, which are directly derived from the descriptions in the previous section.

The communication effort of all algorithms is quadratic in d, with a factor varying between 1 and 4 and independent of n. The ellipsoid-based algorithms exhibit the lowest communication effort. On the other hand, the highest communication effort occurs when the Battery Homothet algorithms or the *Cuboid Homothets Stage 1* algorithm is used. For comparison, the communication effort without aggregation involves sending n matrices $A_i \in \mathbb{R}^{4d \times d}$ and n vectors $b_i \in \mathbb{R}^{4d}$, resulting in $4d^2n + 4dn$, which is a function of d and n.

The computational effort is shown in Table 3.1 in terms of LPs, CPs, and SDPs, as a function of the number of devices n and the number of time periods d. We have specified the problem dimensions in the columns *Constraints* and *Variables*. The computation effort for the algorithm right-hand side (RHS) summation is left out, as this only requires the summation of the right-hand vectors in the half-space representation. It can be seen that the problem dimension increases with d and n for the projection-based algorithms, while for the non-projection-based algorithms, the problem dimension is a function of d only. Conversely, for the non-projection-based algorithms, the number of problems to be solved increases with n, while for the projection-based algorithms, only one problem needs to be solved.

Table 3.1.: Communication effort (floating-point numbers sent to the auxiliary service purchaser) and computation effort as functions of the number of devices n and the number of time periods d. Table reproduced (Öztürk et al., 2022).

Algorithms	**Ref.**	**Comm. effort**	**Problems**	**Constraints**	**Variables**
RHS Summation	Barot and Taylor, 2017	$4d^2 + 4d$	-	-	-
RHS Summation with PC	Barot, 2017	$4d^2 + 4d$	$4dn$ **LPs**	$4d$	d
Zonotopes ℓ_1	Müller et al., 2015	$2d^2 + 2d - 1$	n **LPs** $n(d^2 + d)$ **LPs**	$2d^2 + 8d - 1$ $4d$	$d^2 + 4d - 1$ d
Zonotopes ℓ_∞	Müller et al., 2015	$2d^2 + 2d - 1$	n **LPs** $n(d^2 + d)$ **LPs**	$2d^2 + 8d - 1$ $4d$	$3d$ d
Zonotopes ℓ_2	Müller et al., 2015	$2d^2 + 2d - 1$	n **CPs** $n(d^2 + d)$ **LPs**	$6d - 1$ $4d$	$3d - 1$ d
Zonotopes weighted	Müller et al., 2019	$2d^2 + 2d - 1$	n **LPs** $n(d^2 + d)$ **LPs**	$6d - 1$ $4d$	$3d - 1$ d
Cuboid Homothets Stage 0	Nazir et al., 2018	$2d^2 + 3d + 1$	n **CPs** 1 **CP**	$6d - 1$ $5d$	$2d$ $2d$
Cuboid Homothets Stage 1	Nazir et al., 2018	$4d^2 + 5d + 1$	n **CPs** $2nd$ **CPs** 1 **CP**	$6d - 1$ $6d$ $5d$	$2d$ $2d$ $2d$
Battery Homothets (OA)	Zhao et al., 2017	$4d^2 + 5d + 1$	n **LPs**	$32d^2 + 4d + 1$	$16d^2 + d + 1$
Battery Homothets (IA)	Zhao et al., 2017	$4d^2 + 5d + 1$	n **LPs**	$32d^2 + 4d + 1$	$16d^2 + d + 1$
Battery Homothet Projection with LDR	Zhao et al., 2016	$4d^2 + 5d + 1$	1 **LP**	$20d^2n + 4dn + 1$	$d^2(17n - 1) + dn + 1$
Ellipsoid Projection	Barot, 2017	$d^2 + d$	1 **SDP**	$4dn$	$d^2n^2 + dn$
Ellipsoid Projection with LDR	Zhen and den Hertog, 2018	$d^2 + d$	1 **SDP**	$4dn$	$d^2n + dn$

3.2.2. Benchmark Formulation

We develop a comprehensive open-source benchmark for evaluating approximate aggregation strategies. To this end, we consider residential areas with n households, where each household is equipped with a stationary battery, modeled by

$$\mathcal{B}\left(S_{\text{init},i}, \frac{1}{2}S_{\text{init},i}, p_i\right) \text{ with } p_i = \left(1, x_{\min,i}, x_{\max,i}, 0, S_{\max,i}, \frac{1}{4}\right)^\top,$$

cf. (2.19). The battery parameters are sampled from intervals: $S_{\max,i} \in [10.5, 13.5]$ (KWh), $S_{\text{init},i} \in [0, 10.5]$ (kWh), $x_{\max,i} \in [4, 6]$ (kW), and $x_{\min,i} \in [-6, -4]$ (kW) $\forall i = 1, \dots, n$. We assume that the residents have signed contracts with an aggregator that monitors and manages the batteries. The flexibility, provided by the batteries, is utilized by the aggregator and offered on energy markets. A third party (within the framework of the benchmark the grid operator) acquires the flexibility offered, which is then used to fulfill various objectives.

Given an objective function $f : \mathbb{R}^d \to \mathbb{R}$ and an approximate aggregate flexibility $\mathcal{A}$, the buyer, i.e., the third party, solves the following problem

$$z_{\text{approx}} \coloneqq \min_x f(x) \tag{3.20a}$$

$$\text{subject to } x \in \mathcal{A}, \tag{3.20b}$$

to determine the optimal power profile for the respective application. Given matrices A_i and right-hand vectors b_i, cf. (2.18), the above problem can also be solved without aggregation by a central controller within the exact feasibility region as follows

$$z_{\text{exact}} \coloneqq \min_{x, x_i} f(x) \tag{3.21a}$$

$$\text{subject to } x = \sum_{i=1}^{n} x_i \tag{3.21b}$$

$$A_i x_i \leq b_i. \tag{3.21c}$$

Furthermore, we define the solution for a scenario without any flexibility as $z_{\text{no flex}} \coloneqq f(\mathbf{0}_d)$. The objectives

$$f(x) = c^\top \left(x + \sum_{i=1}^{n} q_i\right) \Delta t$$

for economic cost minimization and

$$f(x) = \left\| x + \sum_{i=1}^{n} q_i \right\|_{\infty}$$

for peak power reduction are used within this study, where q_i is the household demand and c the energy price.

The publicly available half-hourly energy consumption data for households in Ireland from July 2009 to December 2010 (Commission for Energy Regulation, 2012) are used for the q_i and the hourly day-ahead prices for the Germany-Austria-Luxembourg region from 2016 (Entsoe, 2015) for the c. Both data sets were resampled to quarter-hourly data and time-synchronized with the publicly available code by Rheinberger, 2021a.

With these definitions and the data, we define the unused potential ratio (UPR) as a quality criterion for inner approximations as

$$\text{UPR} := \frac{z_{\text{approx}} - z_{\text{exact}}}{z_{\text{no flex}} - z_{\text{exact}}}. \tag{3.22}$$

The UPR is a measure for the degree of unused flexibility in the exact feasible region. If the UPR is close to zero, then the approximate problem yields almost the same result as the exact problem. Otherwise, if the UPR is close to 100%, then there is a large unused potential in the exact feasible region available. Thus, an approximation with close to zero UPR values is preferable. It is also possible that the UPR value exceeds 100%. In such cases, using the solution that corresponds to no flexibility (,i.e., $\mathbf{0}_d$) leads to better results than solving the approximate problem (3.20). It should be noted that in this case, the approximation is ineffective, as better results can be obtained without including flexibility. To be of practical use, any approximation must at least outperform a scenario in which there is no flexibility. Moreover, the absence of $\mathbf{0}_d$ implies continuous use of the aggregate storage, which may negatively impact the longevity of individual devices.

For outer approximations, we first solve the optimization problem defined in (3.20), which results in an optimal solution $x^{\star}_{\text{approx}}$ that minimizes a given objective function. However, this solution may be infeasible with respect to the exact flexibility, in which case an additional power exchange is needed to make the solution feasible. The corresponding Minimum Imbalance Energy (MIE) can be computed by solving the central problem in (3.21) using the objective

$$f(x) = \|x^{\star}_{\text{approx}} - x\|_1 \Delta t.$$

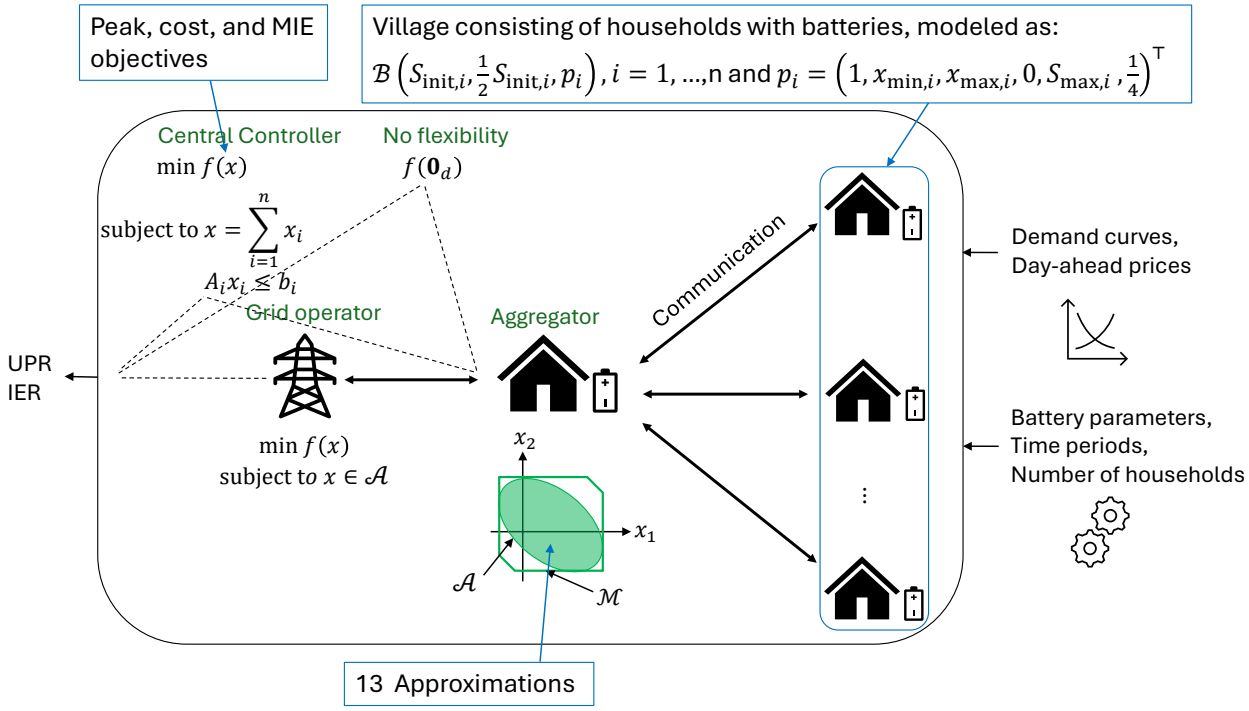

Figure 3.7.: Overview of the benchmark: The aggregator calculates the aggregated flexibility that is sent to the grid operator. The grid operator calculates the optimal value based on predefined objectives. A centralized controller solves the same problem and a separate evaluation is performed without taking flexibility into account. The inputs for the benchmark consist of battery parameters, the number of time periods and households, demand curves, and day-ahead prices, while the output are the UPR and IER values.

Let $x^\star_{\text{exact}}$ be the optimal solution to this problem. If $\|x^\star_{\text{exact}}\|_1 > 0$, then we define the Imbalance Energy Ratio (IER)

$$\text{IER} := \frac{\text{MIE}}{\|x^\star_{\text{exact}}\|_1 \Delta t} \tag{3.23}$$

as a quality criterion for outer approximations. The IER represents the MIE as a fraction of the feasible aggregate energy used for that purpose. Note that the IER is a positive value, and an outer approximation that achieves low IER values is preferable.

The UPR and IER can be computed within the benchmark for tuples (n, d) representing the number of devices in the residential area and the number of time periods, respectively. Due to random battery parameters, the calculations are performed (if not otherwise stated) five times for each approximation and fixed tuple (n, d) by constructing five random villages. An overview of the benchmark with inputs and outputs is shown in Fig. 3.7. Once the objective function and the approximation method have been set, the UPR calculations are carried out five times for a tuple (n, d). The median of five runs, i.e., the median of five

UPR or IER values, is used for further analysis. Finally, the algorithms that require more than 10 minutes for a tuple (n, d) are skipped to avoid excessive calculation times. A total of thirteen approximations are implemented and benchmarked within this framework.

3.2.3. Benchmark Results

This section presents and analyzes the benchmark results for the previously discussed algorithms. First, the results for the inner approximations are presented and discussed. Next, the performance of the outer approximations is evaluated. Additionally, a detailed analysis of the shortcomings of the approximation strategies is provided. These shortcomings will then serve as the foundation for developing improved strategies in the following chapter.

Figure 3.8 displays the results obtained from the inner approximations for tuples

$$(n, d) \in \{20\} \times \{4, 8, 12, 16, 20, 24\}$$

in the first column and for tuples

$$(n, d) \in \{2, 6, 10, 20, 30, 40, 50\} \times \{12\}$$

in the second column, i.e., with fixed number of devices and varying time periods and fixed number of time periods and varying numbers of devices, respectively. The cost UPR values are shown in the first row, the peak UPR values in the second row, and the computation times in the third row.

The results show that only the *Ellipsoid Projection*, *Ellipsoid Projection with LDR*, and *Battery Homothets* algorithms achieve peak UPR values below 100%. This indicates that $\mathbf{0}_d$, which corresponds to no flexibility, performs better than the flexibility set provided by the other algorithms. Furthermore, Fig. 3.8 shows that certain algorithms exhibit objective-dependent performance, e.g., the *Ellipsoid Projection with LDR* algorithm ranks highest in peak UPR but performs the poorest in cost UPR.

Moreover, as shown in Fig. 3.8, most inner approximation strategies display non-linear behavior in computation time as the number of time periods increases. This limits their computational efficiency, rendering them impractical for longer time frames, such as day-ahead computations involving 15-minute intervals. In contrast, regarding the number of devices, most inner approximations demonstrate linear behavior, except for the projection-based algorithms, which show non-linear behavior.

The results for the outer approximations are presented in Fig. 3.9. In the first column, results for the tuples

$$(n, d) \in \{20\} \times \{4, 8, 12, 16, 20, 24\}$$

are shown, while the second column displays results for tuples

$$(n, d) \in \{2, 6, 10, 20, 30, 40, 50\} \times \{12\}.$$

The first row indicates the cost IER values, the second row shows the peak IER values, and the third row presents the computation times. The approximation strategy *RHS Summation*

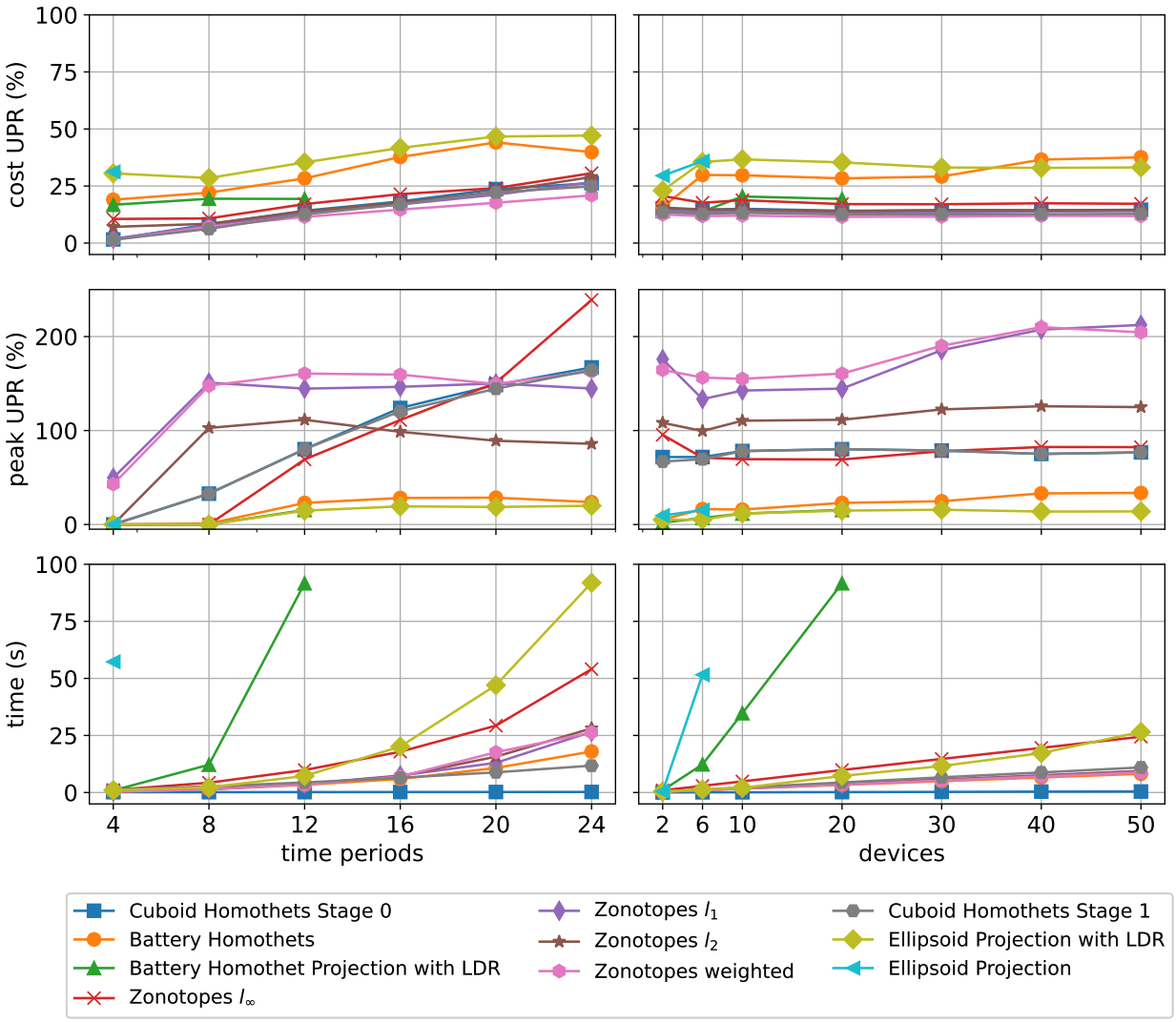

Figure 3.8.: Results for experiments with tuples $(n, d) \in \{20\} \times \{4, 8, 12, 16, 20, 24\}$ in the first column and experiments with tuples $(n, d) \in \{2, 6, 10, 20, 30, 40, 50\} \times \{12\}$ in the second column. The cost UPR values are shown in the first row, the peak UPR values in the second row, and the calculation times in the third row.

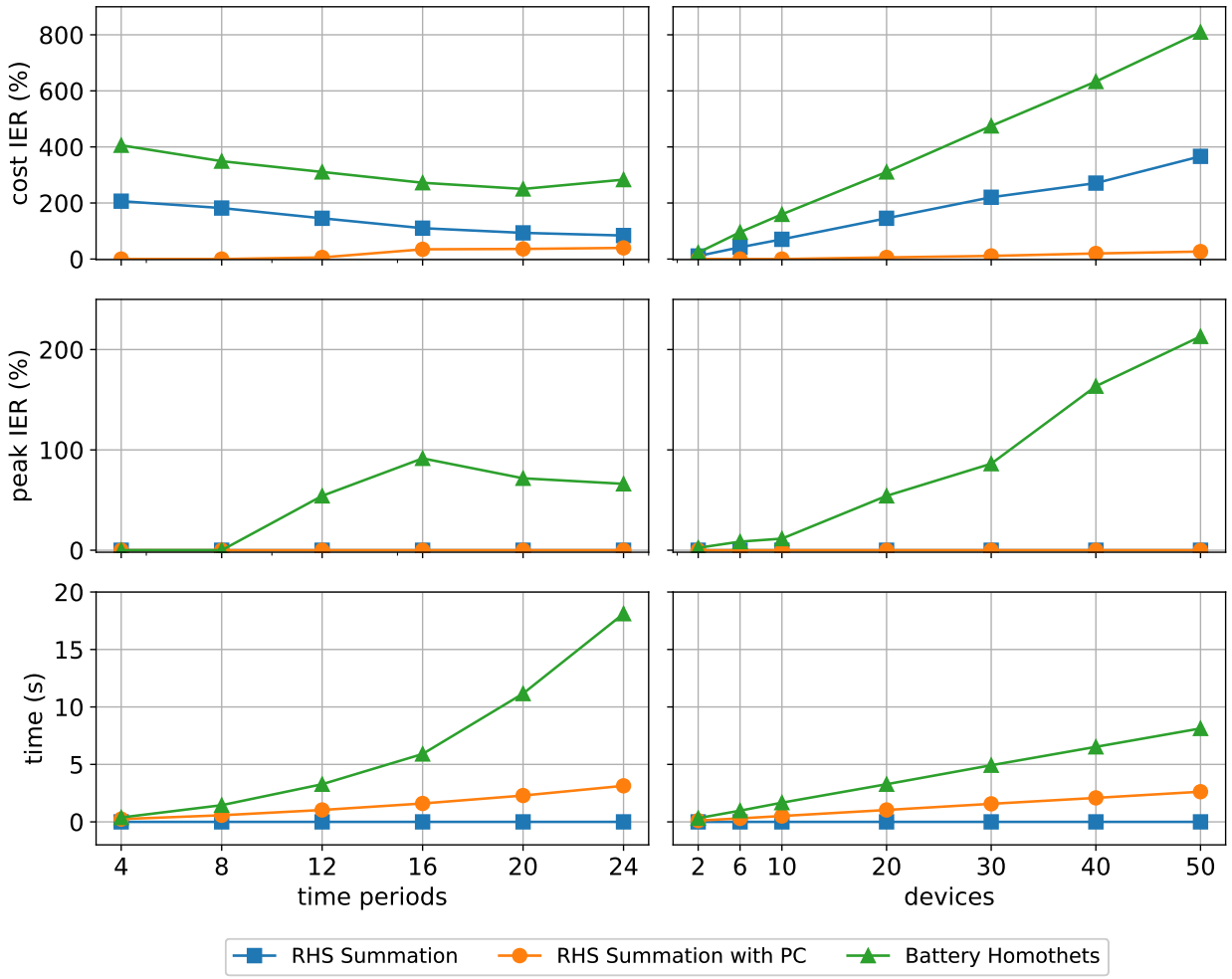

Figure 3.9.: Results for experiments with tuples $(n, d) \in \{20\} \times \{4, 8, 12, 16, 20, 24\}$ in the first column and experiments with tuples $(n, d) \in \{2, 6, 10, 20, 30, 40, 50\} \times \{12\}$ in the second column. The cost IER values are shown in the first row, the peak IER values in the second row, and the calculation times in the third row.

with PC achieves the best results in both peak and cost IER, and ranks second best in computation time. In contrast, the *Battery Homothets* algorithm yields the worst results compared to the other two algorithms.

3.3. Summary

This chapter provided a detailed examination of state-of-the-art approximation strategies. Additionally, an open-source benchmark was proposed for testing these strategies. A total of thirteen approximation methods—ten inner and three outer approximations—were evaluated within this framework. Our findings can be summarized as follows.

For outer approximations, the algorithm *RHS summation with PC* outperforms the others in terms of peak and cost IER, while ranking second in computation time.

However, for providing ancillary services in energy markets, inner approximations are favored, as penalties are incurred if the promised power profiles are not met. On the other hand, state-of-the-art inner approximations may exclude nominal power profiles, such as the vector of zeros, resulting in a flexibility set that does not allow for the idle state, may exhibit objective-dependent performance, and usually have prohibitive computational complexity (with respect to time periods).

In addition, while bottom-up strategies generally exhibit linear time complexity with respect to the number of devices, as each set is approximated individually before being added, this linear behavior is not consistently observed in top-down strategies, e.g., the projection-based strategies studied in this chapter exhibit non-linear time complexity with respect to the number of devices, which suggests a preference for bottom-up strategies.

These insights indicate that there is a need for innovative inner approximation strategies that prioritize a bottom-up approach, achieving high precision across various objectives while also addressing the computational constraints of existing approximation methods. In the subsequent chapters, we aim to elaborate on the development of such aggregation strategies. All findings in this chapter have been published (Öztürk et al., 2022).

4. Scalable Aggregation and Dissaggregation

The preceding chapter highlighted the shortcomings of existing approximation techniques and emphasized the need for innovative approaches. In this chapter, we propose a novel aggregation strategy for convex storage models.

Section 4.1 develops a novel aggregation strategy for polytopes that fulfill two specific assumptions. The assumptions are outlined, example polytopes that satisfy these assumptions are given, and definitions of extreme actions within polytopes satisfying the assumptions are defined.

In Section 4.2, the aggregation strategy is extended to BESS models. The enhanced strategy is then compared with ten state-of-the-art aggregation algorithms using the established benchmark, demonstrating superior performance in accuracy and computational complexity across different objectives.

Section 4.3 further extends the aggregation strategy to convex storage models. Use cases with one aggregator, multiple aggregators, and various flexibility types demonstrate the applicability and scalability of the extended strategy.

Finally, Section 4.4 concludes the chapter by introducing a novel disaggregation strategy for convex storage models. All findings presented in this chapter have been published (Öztürk, Faulwasser, et al., 2024; Öztürk, Kaspar, et al., 2024; Öztürk et al., 2025).

4.1. Aggregation Strategy for Special Polytopes

This section aims to develop a novel aggregation strategy. In Section 4.1.1, we present an overview of the proposed approach, including its definition and example calculations. First, the scope is restricted by introducing two assumptions. Then, extreme actions are defined within polytopes that fulfill the assumptions. Corresponding extreme actions are then added to compute extreme elements in the aggregated flexibility. Finally, the convex hull of the summed extreme actions is defined as an approximation to the aggregated flexibility. Following this, Section 4.1.2 details the properties of the computations and the aggregation strategy, accompanied by the relevant proofs.

4.1.1. Overview of Aggregation Strategy

Let $x \in \mathbb{R}^d$ and $t \leq d$, then we define the projection of x into t-dimensional space as $\text{Proj}^t(x) := (x_1, \ldots, x_t, \mathbf{0}_{d-t})^\top$, where $\mathbf{0}_{d-t}$ is the $d-t$ dimensional vector of zeros. We consider the following assumptions for polytopes $\mathcal{P} \subset \mathbb{R}^d$:

Assumption 1 (Required Flexibility)**.** *If* $\text{Proj}^t(x) \in \mathcal{P}$ *for* $t \in \{1, \ldots, d-1\}$*, then there exists an* $\varepsilon \in \mathbb{R} \setminus \{0\}$ *such that* $(x_1, \ldots, x_t, \varepsilon, \mathbf{0}_{d-(t+1)})^\top \in \mathcal{P}$*. Furthermore, there exists an* $\varepsilon \in \mathbb{R} \setminus \{0\}$ *such that* $(\varepsilon, \mathbf{0}_{d-1})^\top \in \mathcal{P}$*.*

Assumption 2 (Projection Feasibility)**.** *If* $x \in \mathcal{P}$*, then* $\text{Proj}^t(x) \in \mathcal{P}$ *for all* $t \in \{1, \ldots, d-1\}$*. Furthermore,* $\mathbf{0}_d \in \mathcal{P}$*.*

The first assumption requires a minimum flexibility in each time period, and the second requires that any projection of x be feasible, if x is feasible. The requirement that $\mathbf{0}_d \in \mathcal{P}$ represents the extension of the projections beyond $t = 1$. Figure 4.1 illustrates Assumption 2, displaying two polytopes. The polytope on the left satisfies Assumption 2, while the one on the right violates it. Both Assumptions are fulfilled, e.g., for the BESS model (2.1), if

$$x_{\max} > 0,\ x_{\min} < 0,\ S_{\min} < S_{\max},\ \alpha^d S_{\text{init}} \geq S_{\min},\ \text{and } S_f = S_{\min}.$$

Subsequently, our aim is to find extreme vectors (extreme actions) within polytopes that fulfill the Assumptions 1 and 2, which we define below.

Definition 1 (Extreme actions)**.** Let polytopes $\mathcal{P}_i \subset \mathbb{R}^d$, $i \in \{1, \ldots, n\}$ satisfy Assumptions 1 and 2. Then, for $j \in \{-1, 1\}^d$ the vectors $y_i^j \in \mathbb{R}^d$ defined by

$$y_{i,1}^j := j_1 \cdot \max\{j_1 \cdot x \in \mathbb{R} : (x, \mathbf{0}_{d-1})^\top \in \mathcal{P}_i\}, \tag{4.1}$$

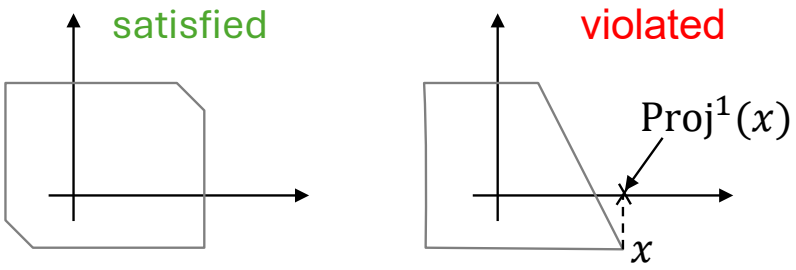

Figure 4.1.: Illustration of Assumption 2. While the polytope on the left satisfies the assumption, the polytope on the right does not.

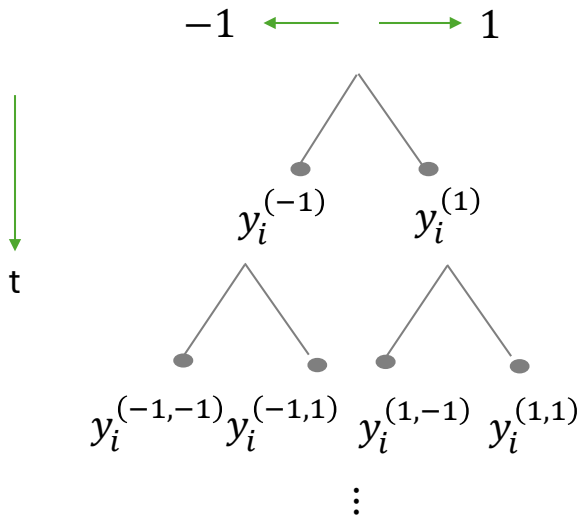

Figure 4.2.: Tree structure for extreme action calculation: left branch with -1, right branch with 1, and increasing time periods indicated by the arrow pointing downwards.

and

$$y_{i,t}^{j} \coloneqq j_t \cdot \max\{j_t \cdot x \in \mathbb{R} : (y_{i,[t-1]}^{j}, x, \mathbf{0}_{d-t})^\top \in \mathcal{P}_i\}, \tag{4.2}$$

for $t \in \{2, \dots, d\}$ are called extreme actions.

Note that $j_t = -1$ in (4.1) and (4.2) is equivalent to replacing the maximization with a minimization. Intuitively, the vectors are obtained by moving maximally in the positive direction when $j_t = 1$ and in the negative direction when $j_t = -1$, cf. Fig. 4.3. Moreover, the extreme actions can be organized in a tree structure as shown in Fig. 4.2. The following example details the calculations outlined in Definition 1 for the BESS model.

Example 2 (Extreme Actions)**.** Let a BESS model with minimum final energy $S_f = S_{\min}$ be given. Then, for a vector $j \in \{-1, 1\}^d$, the calculations in Definition 1 are equivalent to

$$y_{i,1}^{j} = j_1 \cdot \max\left\{j_1 \cdot x \in \mathbb{R} : x_{\min,i} \le x \le x_{\max,i},\ S_{\min,i} \le \alpha_i S_{\text{init},i} + x\Delta t \le S_{\max,i}\right\},$$

$$y_{i,t}^{j} = j_t \cdot \max\left\{j_t \cdot x \in \mathbb{R} : x_{\min,i} \le x \le x_{\max,i},\ S_{\min,i} \le \alpha_i^t S_{\text{init},i} + \sum_{\tau=1}^{t-1} \alpha_i^{t-\tau} y_{i,\tau}^{j} \Delta t + x\Delta t \le S_{\max,i}\right\}.$$

Note that the constraints specified in these sets are merely the power and energy constraints in (2.2). With the above expressions, the calculations can be easily carried out, e.g., for $j_1 = 1$ we have $y_{i,1}^{j} = \max\{x_{\max,i}, \frac{S_{\max,i} - \alpha_i S_{\text{init},i}}{\Delta t}\}$. Consequently, either the maximum power limit is selected, if possible, or the power level that fully charges the BESS.

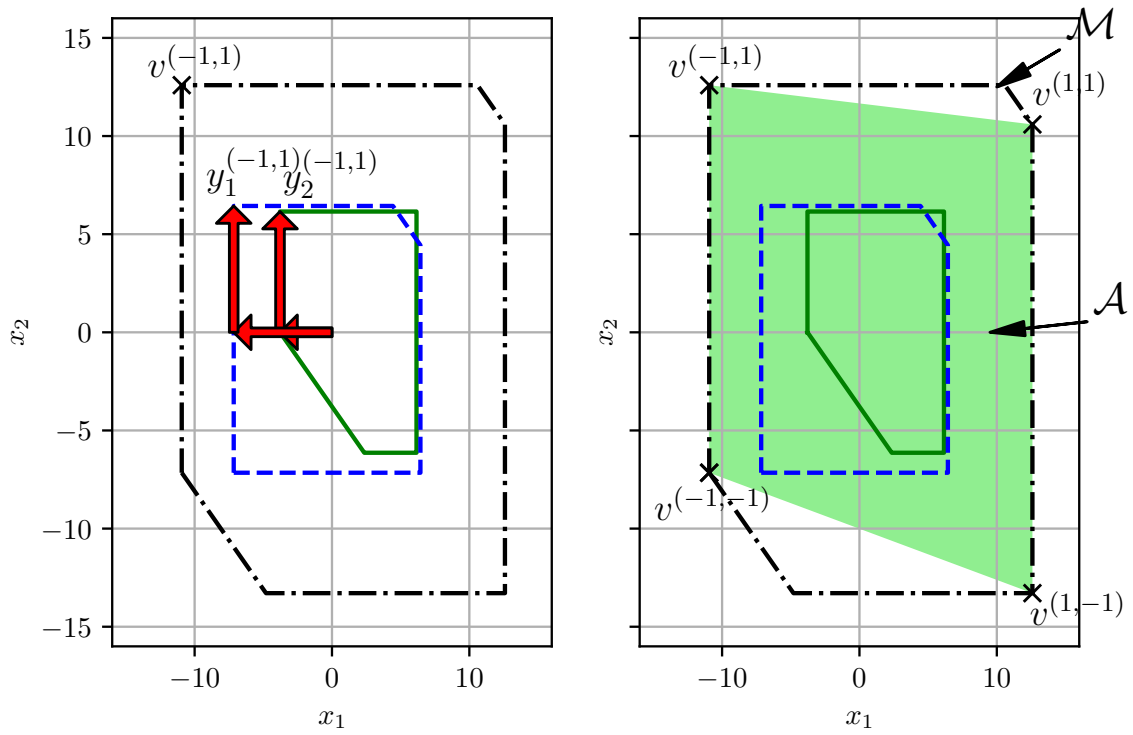

Figure 4.3.: Left: vectors $y_1^{(-1,1)}$, $y_2^{(-1,1)}$ within the polytopes shown in dashed blue and solid green, and the sum $v^{(-1,1)}$ shown in the Minkowski sum $\mathcal{M}$ in dash-dotted black. Right: all possible vectors v^j, $j \in \{-1,1\}^2$ in the Minkowski sum with the resulting set $\mathcal{A}$ in light-green.

Given extreme actions computed for polytopes $\mathcal{P}_i$ for $i = 1, \ldots, n$, we define the sum over all extreme actions with a fixed $j \in \{-1,1\}^d$ by

$$v^j := \sum_{i=1}^{n} y_i^j. \tag{4.3}$$

Moreover, the convex hull of the set of summed extreme actions provides an inner approximation of the Minkowski sum, i.e.,

$$\mathcal{A} = \mathrm{Conv}\left(\left\{v^j : j \in \{-1,1\}^d\right\}\right). \tag{4.4}$$

The set $\mathcal{A}$ can be described as a deformed cuboid, cf. Fig. 4.3, and it holds by definition that $\mathcal{A} \subseteq \mathcal{M}$. Moreover, an intrinsic property of the summed extreme actions is that they form distinct vertices of the Minkowski sum, which we will prove in the following. Furthermore, the origin is shown to lie in $\mathcal{A}$.

4.1.2. Main Mathematical Results

Readers may skip this section if mathematical details are not required and proceed with Section 4.2, where the proposed aggregation strategy is extended to BESS models with arbitrary final energy restriction. We begin by providing some common definitions that will be used in the subsequent proofs.

Definition 2 (Proper Convex Combination)**.** We say $v \in \mathcal{P}$ is a proper convex combination of $p, q \in \mathcal{P}$ if $v = tp + (1-t)q$, with $p \neq q$ and $t \in (0,1)$.

Definition 3 (Vertex)**.** A vector $v \in \mathcal{P}$ is called vertex if and only if it cannot be expressed as a proper convex combination of elements in $\mathcal{P}$.

Definition 4 (Convex Independent)**.** A set of vectors is said to be convex-independent if no vector in the set can be expressed as a convex combination of the others.

For a concise introduction to these definitions, as well as other equivalent definitions, we refer to, e.g., Conforti et al., 2014; Matoušek, 2002.

Lemma 1. *Let polytopes $\mathcal{P}_i \subset \mathbb{R}^d$, $i \in \{1, \dots, n\}$, fulfill the Assumptions 1 and 2. Further, let $v^j, v^k \in \mathbb{R}^d$, $j, k \in \{-1, 1\}^d$ satisfy* (4.3)*. Then, the following hold*

1. $v_t^j \begin{cases} \geq 0 & \text{for } j_t = 1 \\ \leq 0 & \text{for } j_t = -1 \end{cases} \qquad \forall\, t \in \{1, \dots, d\},$
2. *if $j \neq k$, then $v^j \neq v^k$.*

Proof. (Property 1) For $t = 2, \dots, d$ we have $v_t^j = \sum_{i=1}^n j_t \cdot \max\{j_t \cdot x \in \mathbb{R} : (y^j_{i,[t-1]}, x, \mathbf{0}_{d-t})^\top \in \mathcal{P}_i\}$. By construction holds $y_i^j \in \mathcal{P}_i$, and by Assumption 2 that $\text{Proj}^{t-1}(y_i^j) \in \mathcal{P}_i$, hence $(y^j_{i,[t-1]}, 0, \mathbf{0}_{d-t})^\top \in \mathcal{P}_i$. Therefore, if $j_t = 1$, then $\max\{x \in \mathbb{R} : (y^j_{i,[t-1]}, x, \mathbf{0}_{d-t})^\top \in \mathcal{P}_i\} \geq 0 \;\forall\, i$, hence $v_t^j \geq 0$. Otherwise, if $j_t = -1$, then $-\max\{-x \in \mathbb{R} : (y^j_{i,[t-1]}, x, \mathbf{0}_{d-t})^\top \in \mathcal{P}_i\} \leq 0 \;\forall\, i$, thus $v_t^j \leq 0$.

For $t = 1$ we have $v_1^j = \sum_{i=1}^n j_1 \cdot \max\{j_1 \cdot x \in \mathbb{R} : (x, \mathbf{0}_{d-1})^\top \in \mathcal{P}_i\}$. Assumption 2 gives $\mathbf{0}_d \in \mathcal{P}_i$, and by the same reasoning it follows that $v_1^j \geq 0$ if $j_1 = 1$ and $v_1^j \leq 0$ if $j_1 = -1$.

(Property 2) Suppose that $v_t^k = v_t^j$ for $k_t \neq j_t$. Without loss of generality let $j_t = 1$ and $k_t = -1$. Then, $v_t^j = \sum_{i=1}^n y_t^j = \sum_{i=1}^n y_t^k = v_t^k$. Since $y_t^j \geq 0$ and $y_t^k \leq 0$ (Property 1), we have $y_t^j = y_t^k = 0 \;\forall\, i$. This is, however, impossible as by Assumption 2 $\text{Proj}^{t-1}(y_i^j) \in \mathcal{P}_i$, which

implies that $(y^j_{i,[t-1]}, 0, \mathbf{0}_{d-t})^\top \in \mathcal{P}_i$. Furthermore, Assumption 1 ensures the existence of an $\varepsilon \in \mathbb{R} \setminus \{0\}$ with $(y^j_{i,[t-1]}, \varepsilon, \mathbf{0}_{d-t})^\top \in \mathcal{P}_i$, which gives at least two elements, and therefore

$$\max\{x \in \mathbb{R} : (y^j_{i,[t-1]}, x, \mathbf{0}_{d-t})^\top \in \mathcal{P}_i\} \neq -\max\{-x \in \mathbb{R} : (y^k_{i,[t-1]}, x, \mathbf{0}_{d-t})^\top \in \mathcal{P}_i\}.$$

Thus, the assumption that $v^k_t = v^j_t$ must be false, which yields $v^k_t \neq v^j_t$.

For $j \neq k$, there exists an index t with $j_t \neq k_t$ and by the above reasoning holds $v^k_t \neq v^j_t$, hence $v^j \neq v^k$. □

Lemma 1 states that the t^{th} coordinate of v^j is positive or zero if $j_t = 1$, and negative or zero if $j_t = -1$. Furthermore, if $j \neq k$, then $v^j \neq v^k$. Hence, the summed extreme actions are distinct.

Lemma 2. *Let polytopes $\mathcal{P}_i \subset \mathbb{R}^d$, $i \in \{1, \dots, n\}$ with Minkowski sum $\mathcal{M}$ fulfill the Assumptions 1 and 2, and $p \in \mathbb{R}^d$. Further, let $v^j \in \mathbb{R}^d$, $j \in \{-1, 1\}^d$ satisfy* (4.3)*. For $t \in \{2, \dots, d\}$, if $p_{[t-1]} = v^j_{[t-1]}$ and $p_t > v^j_t$ with $j_t = 1$, or $p_t < v^j_t$ with $j_t = -1$, then $p \notin \mathcal{M}$. Furthermore, if $p_1 > v^j_1$ with $j_1 = 1$ or $p_1 < v^j_1$ with $j_1 = -1$, then $p \notin \mathcal{M}$.*

Proof. Assume that $p \in \mathcal{M}$ and $t > 1$, then there are $p_i \in \mathcal{P}_i$ with $p = \sum_{i=1}^n p_i$, and $p_{[t-1]} = \sum_{i=1}^n p_{i,[t-1]} = \sum_{i=1}^n y^j_{i,[t-1]} = v^j_{[t-1]}$. We distinguish between two cases: one where all individual vectors in the summation up to coordinate $t-1$ are equal, and another where at least one is different. Together, these cases exhaust all possibilities.

Case 1: Let $p_{i,[t-1]} = y^j_{i,[t-1]}\ \forall\ i$. If $p_t > v^j_t$ and $j_t = 1$, then there is a $k \in \{1, \dots, n\}$ with $p_{k,t} > y^j_{k,t}$. Since $p_{k,[t-1]} = y^j_{k,[t-1]}$ and $y^j_{k,t} = \max\{x \in \mathbb{R} : (y^j_{k,[t-1]}, x, \mathbf{0}_{d-t})^\top \in \mathcal{P}_k\}$ it follows that $(p_{k,[t-1]}, p_{k,t}, \mathbf{0}_{d-t})^\top \notin \mathcal{P}_k$, thus $\text{Proj}^t(p_k) \notin \mathcal{P}_i$. Assumption 2 yields $p_k \notin \mathcal{P}_k$, which contradicts the assumption $p \in \mathcal{M}$. If $p_t < v^j_t$ and $j_t = -1$, then by similar reasoning it follows that $p \notin \mathcal{M}$.

Case 2: $\exists\, l, k \in \{1, \dots, n\}$ with $p_{l,[t-1]} \neq y^j_{l,[t-1]}$ and $p_{k,[t-1]} \neq y^j_{k,[t-1]}$. Note that the negation of Case 1 yields at least two indices $l, k \in \{1, \dots, n\}$, and a minimum index $m \in \{1, \dots, d\}$ with $p_{l,m} < y^j_{l,m}$ and $p_{k,m} > y^j_{k,m}$. For $m \neq 1$ we have $p_{l,[m-1]} = y^j_{l,[m-1]}$ and $y^j_{l,[m-1]} = p_{k,[m-1]}$. If $j_m = -1$ then $\text{Proj}^m(p_l) \notin \mathcal{P}_l$ and by Assumption 2 $p_l \notin \mathcal{P}_l$. Otherwise, if $j_m = 1$, then $\text{Proj}^m(p_k) \notin \mathcal{P}_k$ and by Assumption 2 $p_k \notin \mathcal{P}_k$. For $m = 1$ we have that $p_{l,1} < y^j_{l,1}$ and $p_{k,1} > y^j_{k,1}$. If $j_1 = 1$, then $p_k \notin \mathcal{P}_k$. Otherwise, if $j_1 = -1$, then $p_l \notin \mathcal{P}_l$.

For $t = 1$ holds that if $p_1 > v^j_1$ and $j_1 = 1$, then there is an index $k \in \{1, \dots, n\}$ with $p_{k,1} > y^j_{k,1}$ therefore $\text{Proj}^1(p_k) \notin \mathcal{P}_k$ and by Assumption 2 $p_k \notin \mathcal{P}_k$. Otherwise, if $p_1 < v^j_1$

and $j_1 = -1$, then by similar reasoning $p_k \notin \mathcal{P}_k$. Hence, all cases lead to contradictions and therefore $p \notin \mathcal{M}$. □

Lemma 2 states that there can be no vector in $\mathcal{M}$ that has $t-1$ coordinates equal to v^j and a value greater than v_t^j in the t^{th} coordinate if $j_t = 1$. Similarly, there cannot be a vector with equal $t-1$ coordinates in $\mathcal{M}$ with a value less than v_t^j when $j_t = -1$. This characteristic behavior is also illustrated in Fig. 4.3.

Proposition 1. *Let polytopes $\mathcal{P}_i \subset \mathbb{R}^d$, $i \in \{1, \dots, n\}$, fulfill the Assumptions 1 and 2. Further, let $v^j \in \mathbb{R}^d$, $j \in \{-1, 1\}^d$ satisfy* (4.3)*, and $\mathcal{A}$ satisfy* (4.4)*. Then, v^j is a vertex of $\mathcal{A}$.*

Proof. We prove the convex independence of the set of vectors $\{v^j : j \in \{-1, 1\}^d\}$ by induction over d. It then follows that v^j is a vertex in $\mathcal{A}$.

Base case: $(d = 1)$ In one-dimensional space, two distinct numbers $v^{(-1)}$ and $v^{(1)}$ are computed (cf. Lemma 1), which are convex independent by definition.

Induction hypothesis: Let the set of vectors $\{v^j : j \in \{-1, 1\}^d\}$ be convex independent for a $d \in \mathbb{N}$.

Induction step: $(d \to d+1)$ The $d+1$ dimensional vectors are constructed by $\{v^{(j,-1)}, v^{(j,1)} : j \in \{-1, 1\}^d\}$. Assume the set of vectors $\{v^{(j,-1)}, v^{(j,1)} : j \in \{-1, 1\}^d\}$ is convex dependent. Then, for some $k \in \{-1, 1\}^d$, we can assume without loss of generality that

$$v^{(k,-1)} = \sum_{j \in \{-1,1\}^d, j \neq k} \alpha_j v^{(j,-1)} + \sum_{j \in \{-1,1\}^d} \beta_j v^{(j,1)}, \tag{4.5a}$$

$$\sum_{j \in \{-1,1\}^d, j \neq k} \alpha_j + \sum_{j \{-1,1\}^d} \beta_j = 1, \tag{4.5b}$$

$$\alpha_j, \beta_j \geq 0. \tag{4.5c}$$

Projecting (4.5) to the first d coordinates gives

$$v^k = \beta_k v^k + \sum_{j \in \{-1,1\}^d, j \neq k} (\alpha_j + \beta_j) v^j,$$

$$(1 - \beta_k) v^k = \sum_{j \in \{-1,1\}^d, j \neq k} (\alpha_j + \beta_j) v^j,$$

where $0 \leq \beta_k \leq 1$. We distinguish two cases: $\beta_k < 1$ and $\beta_k = 1$.

Case 1: If $\beta_k < 1$, then $1 - \beta_k > 0$ and we have

$$v^k = \sum_{j \in \{-1,1\}^d, j \neq k} \frac{(\alpha_j + \beta_j)}{(1 - \beta_k)} v^j,$$
$$\frac{(\alpha_j + \beta_j)}{(1 - \beta_k)} \geq 0,$$
$$\sum_{j \in \{-1,1\}^d, j \neq k} \frac{(\alpha_j + \beta_j)}{(1 - \beta_k)} = 1,$$

hence v^k is convex combination of vectors in $\{v^j : j \in \{-1, 1\}^d\} \setminus \{v^k\}$, which contradicts the induction hypothesis.

Case 2: If $\beta_k = 1$, then $\alpha_j = \beta_j = 0 \ \forall j \in \{-1, 1\}^d \setminus \{k\}$, and it follows from (4.5) that $v^{(k,-1)} = v^{(k,1)}$, which is impossible as the vectors are distinct by Lemma 1. These contradictions show the convex independence of the vectors.

Since $\mathcal{A} = \text{Conv}(\{v^j : j \in \{-1, 1\}^d\})$, and v^k for any $k \in \{-1, 1\}^d$ is not a convex combination of vectors in $\{v^j : j \in \{-1, 1\}^d\} \setminus \{v^k\}$, it follows that v^k is not a convex combination of vectors in $\mathcal{A} \setminus \{v^k\}$, which proofs that v^k is a vertex of $\mathcal{A}$. □

The proposition states that v^j is a vertex of $\mathcal{A}$, and by Lemma 1, the elements of $\{v^j : j \in \{-1, 1\}^d\}$ are distinct. Thus, they are distinct vertices of $\mathcal{A}$.

Proposition 2. *Let polytopes $\mathcal{P}_i \subset \mathbb{R}^d$, $i \in \{1, \ldots, n\}$, with Minkowski sum $\mathcal{M}$ fulfill the Assumptions 1 and 2. Further, let $v^j \in \mathbb{R}^d$, $j \in \{-1, 1\}^d$ satisfy* (4.3)*, $\mathcal{A}$ satisfy* (4.4)*, and $p, q \in \mathcal{M}$ with $v^j = tp + (1 - t)q, t \in (0, 1)$, then, $p, q \in \mathcal{A}$.*

Proof. This statement is obvious if $\mathcal{M} \setminus \mathcal{A} = \emptyset$, i.e., $\mathcal{M} = \mathcal{A}$. Therefore, we temporarily assume that $\mathcal{M} \setminus \mathcal{A} \neq \emptyset$. We distinguish two cases: one where $p, q \in \mathcal{M} \setminus \mathcal{A}$, and one where $p \in \mathcal{M} \setminus \mathcal{A}$ and $q \in \mathcal{A}$ (note that this case also includes $q \in \mathcal{M} \setminus \mathcal{A}$ and $p \in \mathcal{A}$), both of which lead to a contradiction, leaving the only possible case $p, q \in \mathcal{A}$.

Case 1: Let $p, q \in \mathcal{M} \setminus \mathcal{A}$. Assume that $v^j = tp + (1 - t)q$ with $t \in (0, 1)$. Since $v^j \in \mathcal{A}$ and $p, q \notin \mathcal{A}$, it follows that $p \neq v^j$ and $q \neq v^j$. Since $t \in (0, 1)$, it follows that $p \neq q$, hence there are indices in $\{1, \ldots, d\}$ where the entries in p and q are different. Let m be the minimum of these indices. For this index holds $v^j_m = tq_m + (1 - t)p_m$, $t \in (0, 1)$ and $p_t \neq q_t$. Without loss of generality, suppose that $p_m < q_m$, then $p_m < v^j_m < q_m$. Since m is the minimum index, we have equality in v, p and q for the indices $\{1, \ldots, m - 1\}$. If $j_m = 1$,

then by Lemma 2 we have $q \notin \mathcal{M}$. Otherwise, if $j_m = -1$, then by Lemma 2 we see that $p \notin \mathcal{M}$. Therefore we have a contradiction in both cases and the assumption must be false. Hence there are no $p, q \in \mathcal{M} \setminus \mathcal{A}$ with $v^j = tp + (1-t)q$ and $t \in (0,1)$.

Case 2: Let $p \in \mathcal{M} \setminus \mathcal{A}$ and $q \in \mathcal{A}$. The proof for this case is almost a copy of the previous one. Assume that $v^j = tp + (1-t)q$ with $t \in (0,1)$. Since $p \notin \mathcal{A}$, $q \in \mathcal{A}$ and $t \in (0,1)$ it follows that $v^j \neq p$ and $p \neq q$. Since $p \neq q$, there is a minimum index m where the components of p and q are different. For this index holds $v^j_m = tp_m + (1-t)q_m$ and $q_m \neq p_m$. Without loss of generality assume that $p_m < q_m$. Since m is the minimum index, we have equality in the indices $\{1, \dots, m-1\}$. If $j_m = 1$, then it follows by Lemma 2 that $q \notin \mathcal{M}$ otherwise, if $j_m = -1$, then by the same reasoning it follows that $p \notin \mathcal{M}$. This shows that there are no $p \in \mathcal{M} \setminus \mathcal{A}$ and $q \in \mathcal{A}$ with $v^j = tp + (1-t)q$ and $t \in (0,1)$.

In conclusion, we see that the only possible case is $p, q \in \mathcal{A}$. This concludes the proof. □

Proposition 2 states that if v^j is a proper convex combination of elements $p, q \in \mathcal{M}$, then p, q must be in $\mathcal{A}$. We can now state our first main result, which shows that the readily computable 2^d vectors v^j are indeed vertices of the Minkowski sum. Therefore, their convex hull constitutes an inner approximation.

Theorem 1 (Extreme actions define distinct vertices)**.** *Let polytopes $\mathcal{P}_i \subset \mathbb{R}^d$, $i \in \{1, \dots, n\}$, with Minkowski sum $\mathcal{M}$ fulfill the Assumptions 1 and 2. Then, any $v^j \in \mathbb{R}^d$, $j \in \{-1, 1\}^d$, satisfying* (4.3) *is a distinct vertex of $\mathcal{M}$.*

Proof. Suppose that v^j is not a vertex of $\mathcal{M}$, then $v^j = tp + (1-t)q$ with $p, q \in \mathcal{M}$, $p \neq q$ and $t \in (0,1)$. From Proposition 2 it follows that $p, q \in \mathcal{A}$, which gives v^j as a proper convex combination of elements in $\mathcal{A}$. Thus v^j cannot be a vertex of $\mathcal{A}$, which contradicts Proposition 1. Hence, the assumption that v^j is not a vertex of $\mathcal{M}$ must be false. Combining this result with Lemma 1 shows that the sums of extreme actions are distinct vertices of $\mathcal{M}$. □

Theorem 1 implies that $\mathcal{A}$ (cf. (4.4)) is exact for cuboids since cuboids have 2^d vertices and $|\{v^j : j \in \{-1, 1\}^d\}| = 2^d$, i.e., all vertices are computed, and the convex hull covers the entire cuboid. In general, however, the polytopes defined in (2.17) and their Minkowski sum have more than 2^d vertices (cf. Figs. 2.1, 2.2, and 4.3), making any set $\mathrm{Conv}(\{v^j : j \in \mathcal{J}\})$ with $\mathcal{J} \subseteq \{-1, 1\}^d$ an inner approximation of the Minkowski sum. Moreover, the proposed method is besides an inner approximation, also a bottom-up strategy, since the extreme

actions (vertices) in each polytope are computed first and then combined to form distinct vertices of the Minkowski sum. Finally, the origin lies in $\mathcal{A}$, as the following proposition shows.

Proposition 3. *Let the set $\mathcal{A}$ be defined by* (4.4)*, then it holds that $\mathbf{0}_d \in \mathcal{A}$.*

Proof. Let $\{v^j : j \in \{-1, 1\}^d\}$ be the set of vectors defined by (4.3). By construction the set of vectors can be rewritten as $\{v^{(j,-1)}, v^{(j,1)} : j \in \{-1, 1\}^{d-1}\}$, i.e., there are pairwise vectors with equal first $d-1$ coordinates. We define $\alpha \in \mathbb{R}$ for these vectors as

$$\alpha = -\frac{v_d^{(j,-1)}}{v_d^{(j,1)} - v_d^{(j,-1)}}.$$

Since $v_d^{(j,-1)} \leq 0$, $v_d^{(j,1)} \geq 0$, $v_d^{(j,-1)} \neq v_d^{(j,1)}$ cf. Lemma 1, and since $v_d^{(j,1)} - v_d^{(j,-1)} \geq -v_d^{(j,-1)}$, we have $0 \leq \alpha \leq 1$. Using this α for the convex combination yields

$$(1-\alpha)v^{(j,-1)} + \alpha v^{(j,1)} = \left(\left(v_{[d-1]}^{(j,1)}\right)^\top, 0\right)^\top = \left(\left(v_{[d-1]}^{(j,-1)}\right)^\top, 0\right)^\top,$$

which gives the projection of vectors $v^{(j,1)}, v^{(j,-1)}$ into $d-1$-dimensional space. This procedure can be applied to all pairwise vectors with equal first $d-1$ coordinates to obtain the projections of these as a convex combination in the $d-1$-dimensional space. The same condition, namely that there are pairwise vectors with equal $d-2$ components, also holds by construction for the projected vectors. Therefore, the procedure can be applied to these vectors as well to project these into $d-2$-dimensional space. Repeated application of the same procedure results in projection onto 1-dimensional space. Finally, the zero vector can be constructed as a convex combination of the summed extreme actions. Since each convex combination of elements in $\mathcal{A}$ is an element of $\mathcal{A}$, it follows that $\mathbf{0}_d \in \mathcal{A}$. □

4.2. Extension to Battery Energy Storage Systems

We extend the proposed aggregation strategy to BESS models with arbitrary final energy restrictions. The extended strategy is then compared against ten state-of-the-art inner approximations using the previously developed benchmark, cf. Section 3.2.2.

4.2.1. Aggregation Strategy for Battery Energy Storage Systems

The model $\mathcal{B}(S_{\text{init}}, S_{\text{f}}, p)$ defined in (2.19) may violate the Assumption 2 for arbitrary $S_f \in [S_{\min}, S_{\max}]$, cf. Fig. 4.4. Therefore, the method proposed in Section 4.1 is not readily applicable to this model and requires additional modifications. The aim of this section is to present a corrective algorithm that allows the calculation of extreme actions within the model $\mathcal{B}(S_{\text{init}}, S_{\text{f}}, p)$.

We start by noting that the model $\mathcal{B}(S_{\text{init}}, S_{\min}, p)$, which represents a BESS with minimum final energy $S_f = S_{\min}$, fulfills the Assumptions 1 and 2. Hence, the method proposed in Section 4.1 is applicable to this model.

Given an initial energy S_{init}, a parameter vector p, and a $j \in \{-1, 1\}^d$, the corresponding extreme action can be computed using Algorithm 1. The procedure iterates through the components j_t of j. If $j_t = 1$, then Line 4 sets y_t^j to $x_{\max}$ if there is more than $x_{\max}\Delta t$ capacity remaining in the BESS. Otherwise, y_t^j is set to the maximum power that can fit in the BESS to ensure full charging. If $j_t = -1$, then Line 6 sets y_t^j to $x_{\min}$ if $x_{\min}\Delta t$ can be withdrawn from the BESS. Otherwise, y_t^j is set to the power level that fully discharges the BESS. Please be aware that this extreme behavior of charging and discharging in Algorithm 1 is equivalent to the mathematics developed in Section 4.1.1.

Now, suppose that y^j satisfying Definition 1 is obtained for the model $\mathcal{B}(S_{\text{init}}, S_{\min}, p)$ using, e.g., Algorithm 1. The set $\{x \in \mathbb{R} : (y^j_{[d-1]}, x)^\top \in \mathcal{B}(S_{\text{init}}, S_f, p)\}$ is equivalent to

$$\left\{ x \in \mathbb{R} : x_{\min} \leq x \leq x_{\max},\ S_f \leq \alpha^d S_{\text{init}} + \sum_{\tau=1}^{d-1} \alpha^{d-\tau} y_\tau^j \Delta t + x\Delta t \leq S_{\max} \right\}.$$

Algorithm 1 (extremeActionBESS)

Require: $S_{\text{init}}, p, j \in \{-1, 1\}^d$
Ensure: y^j
1: $y^j \leftarrow \mathbf{0}_d$
2: **for** $t = 1$ **to** d **do**
3: **if** $j_t = 1$ **then**
4: $y_t^j \leftarrow \min\left(x_{\max}, \frac{S_{\max} - (\alpha^t S_{\text{init}} + \sum_{\tau=1}^{t-1} \alpha^{t-\tau} y_\tau^j \Delta t)}{\Delta t}\right)$
5: **else if** $j_t = -1$ **then**
6: $y_t^j \leftarrow \max\left(x_{\min}, \frac{S_{\min} - (\alpha^t S_{\text{init}} + \sum_{\tau=1}^{t-1} \alpha^{t-\tau} y_\tau^j \Delta t)}{\Delta t}\right)$
7: **end if**
8: **end for**

Algorithm 2 (correctiveIncreaseBESS)

Require: $S_{\text{init}}, S_f, p, y^j \in \mathcal{B}(S_{\text{init}}, S_{\text{min}}, p)$
Ensure: y^j
1: **if** $S_f > \alpha^d S_{\text{init}} + \sum_{\tau=1}^{d} \alpha^{d-\tau} y_\tau^j \Delta t$ **then**
2: $\quad y_d^j \leftarrow \min\left(x_{\max}, \frac{S_f - (\alpha^d S_{\text{init}} + \sum_{\tau=1}^{d-1} \alpha^{d-\tau} y_\tau^j \Delta t)}{\Delta t}\right)$
3: $\quad t \leftarrow d - 1$
4: $\quad$ **while** $S_f \neq \alpha^d S_{\text{init}} + \sum_{\tau=1}^{d} \alpha^{d-\tau} y_\tau^j \Delta t$ **do**
5: $\quad\quad y_t^j \leftarrow x_{\max}$
6: $\quad\quad y_d^j \leftarrow \min\left(x_{\max}, \frac{S_f - (\alpha^d S_{\text{init}} + \sum_{\tau=1}^{d-1} \alpha^{d-\tau} y_\tau^j \Delta t)}{\Delta t}\right)$
7: $\quad\quad t \leftarrow t - 1$
8: $\quad$ **end while**
9: **end if**

Hence, if $y^j \notin \mathcal{B}(S_{\text{init}}, S_f, p)$, then $S_f > \alpha^d S_{\text{init}} + \sum_{\tau=1}^{d} \alpha^{d-\tau} y_\tau^j \Delta t$ since $y^j \in \mathcal{B}(S_{\text{init}}, S_{\text{min}}, p)$ and the remaining inequalities are identical for both sets. Therefore, increasing the components of y^j within the power constraints such that the inequality corresponding to S_f is satisfied yields $y^j \in \mathcal{B}(S_{\text{init}}, S_f, p)$.

This corrective increase is achieved with Algorithm 2. First, the d^{th} component of y^j is set to the minimum of the maximum power $x_{\max}$ and the power at which the minimum final energy S_f is reached in Line 2. Algorithm 2 terminates if S_f is reached, otherwise the $d-1^{\text{th}}$ component is set to the maximum power level in Line 5 and the d^{th} component is set to the minimum of the maximum power level and the power level at which S_f is reached in Line 6. Algorithm 2 terminates if S_f is reached, otherwise the process is repeated with the $d-2^{\text{th}}$ component and so forth until S_f is reached. Essentially, Algorithm 2 corrects the power profile backwards by increasing the energy in the BESS to reach the specified minimum final energy S_f.

Figure 4.4 illustrates the correction process. The polytopes $\mathcal{B}(S_{\text{init}}, S_{\text{min}}, p)$ and $\mathcal{B}(S_{\text{init}}, S_f, p)$ are shown in dashed blue and solid green, respectively. The extreme actions computed via Algorithm 1 within $\mathcal{B}(S_{\text{init}}, S_{\text{min}}, p)$ are indicated as black crosses. However, one of the crosses at the bottom left is outside the polytope $\mathcal{B}(S_{\text{init}}, S_f, p)$. To achieve the desired energy level S_f, the values must be increased, as indicated by the arrow and the dot. The following lemma ensures that the corrected vectors are well-defined and that Algorithm 2 terminates without errors.

Lemma 3. *Let* $\mathcal{B}(S_{init}, S_{min}, p)$ *with parameter vector* $p = (\alpha, x_{min}, x_{max}, S_{min}, S_{max}, \Delta t)^\top$

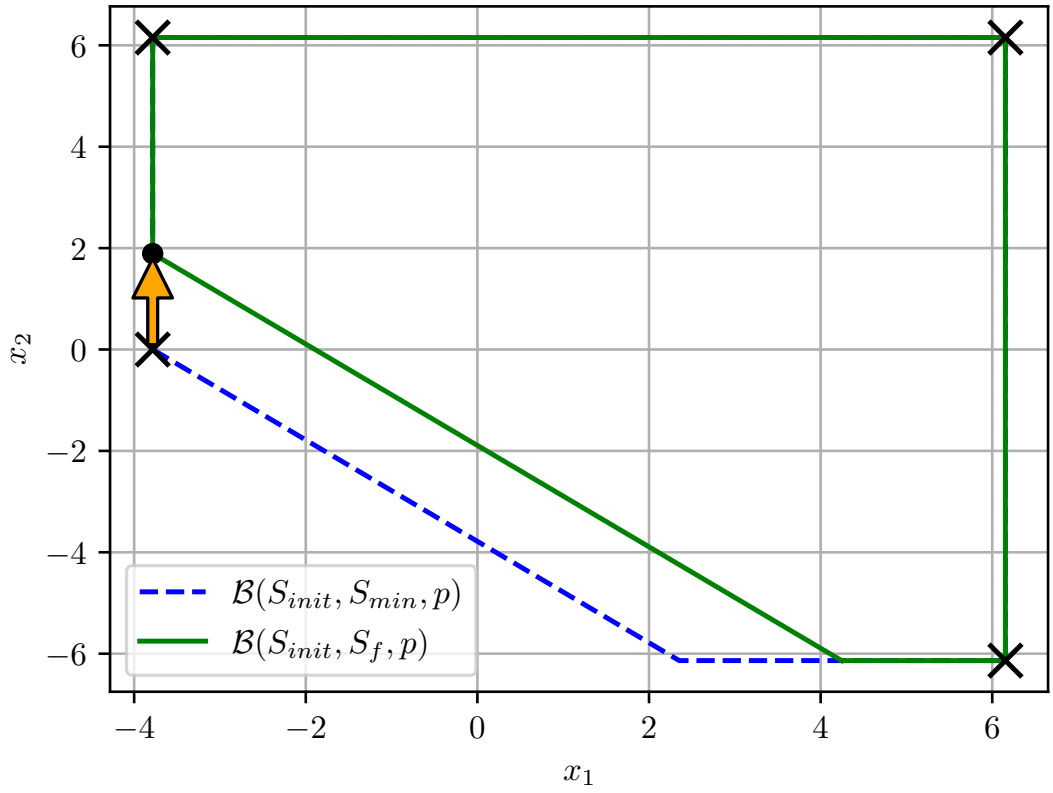

Figure 4.4.: The set $\mathcal{B}(S_{\text{init}}, S_f, p)$ shown in solid green and the set $\mathcal{B}(S_{\text{init}}, S_{\text{min}}, p)$ in dashed blue. The crosses in $\mathcal{B}(S_{\text{init}}, S_{\text{min}}, p)$ show the extreme actions, and the arrow with the dot visualizes the correction process.

fulfill the Assumptions 1 and 2. Further, let $y^j \in \mathbb{R}^d$, $j \in \{-1, 1\}^d$ *satisfy Definition 1 for* $\mathcal{B}(S_{init}, S_{min}, p)$. *Then, Algorithm 2 terminates without errors, if* $\mathcal{B}(S_{init}, S_f, p) \neq \emptyset$.

Proof. If $y^j \in \mathcal{B}(S_{\text{init}}, S_f, p)$, then there is nothing to show. Hence, we assume that $y^j \notin \mathcal{B}(S_{\text{init}}, S_f, p)$. If the assignments in Lines 2 or 6 are applied with

$$\frac{S_f - (\alpha^d S_{\text{init}} + \sum_{\tau=1}^{d-1} \alpha^{d-\tau} y_\tau^j \Delta t)}{\Delta t} \leq x_{\max},$$

then

$$\alpha^d S_{\text{init}} + \sum_{\tau=1}^{d} \alpha^{d-\tau} \tilde{y}_\tau^j \Delta t = \alpha^d S_{\text{init}} + \sum_{\tau=1}^{d-1} \alpha^{d-\tau} \tilde{y}_\tau^j \Delta t + \frac{S_f - (\alpha^d S_{\text{init}} + \sum_{\tau=1}^{d-1} \alpha^{d-\tau} \tilde{y}_\tau^j \Delta t)}{\Delta t} \Delta t = S_f.$$

Therefore, Algorithm 2 terminates with $y^j \in \mathcal{B}(S_{\text{init}}, S_f, p)$ if there is a correction index k

such that one of the assignments in Lines 2 or 6 are applied with

$$\frac{S_f - (\alpha^d S_{\text{init}} + \sum_{\tau=1}^{d-1} \alpha^{d-\tau} y_\tau^j \Delta t)}{\Delta t} \leq x_{\max}.$$

Assume

$$\frac{S_f - (\alpha^d S_{\text{init}} + \sum_{\tau=1}^{d-1} \alpha^{d-\tau} \tilde{y}_\tau^j \Delta t)}{\Delta t} > x_{\max}$$

for all correction indices $k = 1, \ldots, d$, then also for $k = 1$. Therefore, $S_f > \alpha^d S_{\text{init}} + \sum_{\tau=1}^{d-1} \alpha^{d-\tau} x_{\max} \Delta t + x_{\max} \Delta t$. Since $\mathcal{B}(S_{\text{init}}, S_f, p) \neq \emptyset$, there exists an $x \in \mathcal{B}(S_{\text{init}}, S_f, p)$ with $S_f \leq \alpha^d S_{\text{init}} + \sum_{\tau=1}^{d} \alpha^{d-\tau} x_\tau \Delta t$. This gives $\alpha^d S_{\text{init}} + \sum_{\tau=1}^{d} \alpha^{d-\tau} x_{\max} \Delta t < S_f \leq \alpha^d S_{\text{init}} + \sum_{\tau=1}^{d} \alpha^{d-\tau} x_\tau \Delta t$ hence $\sum_{\tau=1}^{d} \alpha^{d-\tau} x_{\max} < \sum_{\tau=1}^{d} \alpha^{d-\tau} x_\tau$, which implies that there is an index m with $x_{\max} < x_m$ and, therefore $x \notin \mathcal{B}(S_{\text{init}}, S_f, p)$, contradicting $x \in \mathcal{B}(S_{\text{init}}, S_f, p)$.

Hence, we conclude that there exists an index for which the assignments in Lines 2 or 6 are applied; therefore, Algorithm 2 terminates without errors. □

Note that the corrected vector is a vertex of $\mathcal{B}(S_{\text{init}}, S_f, p)$, as illustrated in Fig. 4.4. This occurrence is not a mere coincidence, as demonstrated by the following theorem.

Theorem 2. *Let $\mathcal{B}(S_{init}, S_{min}, p) \subset \mathbb{R}^d$ with $p = (\alpha, x_{min}, x_{max}, S_{min}, S_{max}, \Delta t)^\top$ fulfill the Assumptions 1 and 2, and $\mathcal{B}(S_{init}, S_f, p)$ be nonempty. Further, let $y^j \in \mathbb{R}^d$, $j \in \{-1, 1\}^d$ satisfy Definition 1 for $\mathcal{B}(S_{init}, S_{min}, p)$. Then, the corrected extreme action defined by Algorithm 2 is a vertex of $\mathcal{B}(S_{init}, S_f, p)$.*

Proof. For differentiation, we denote the corrected extreme action to y^j as $\tilde{y}^j$. Assume that $\tilde{y}^j$ is not a vertex of $\mathcal{B}(S_{\text{init}}, S_f, p)$, then there are $w, q \in \mathcal{B}(S_{\text{init}}, S_f, p)$ with $\tilde{y}^j = wt + q(1-t)$, $w \neq q$ and $t \in (0, 1)$. We consider two cases: $\tilde{y}^j = y^j$ and $\tilde{y}^j \neq y^j$.

Case 1: $\tilde{y}^j = y^j$ and therefore y^j is not changed by Algorithm 2. Since $w, q \in \mathcal{B}(S_{\text{init}}, S_f, p)$, and $\mathcal{B}(S_{\text{init}}, S_f, p) \subseteq \mathcal{B}(S_{\text{init}}, S_{\min}, p)$ we have that $w, q \in \mathcal{B}(S_{\text{init}}, S_{\min}, p)$. Therefore, we have $y^j = \tilde{y}^j = wt + q(1-t)$, $w \neq q$ and $t \in (0, 1)$. Hence, y^j is not a vertex of $\mathcal{B}(S_{\text{init}}, S_{\min}, p)$, contradicting Theorem 1.

Case 2: $\tilde{y}^j \neq y^j$ and therefore y^j is changed by Algorithm 2. Since $\tilde{y}^j \neq y^j$, there exists a maximum correction index f with $\tilde{y}^j_{[f-1]} = y^j_{[f-1]}$ and $\tilde{y}^j_t = \overline{x}\, \forall t \in \{f, \ldots, d-1\}$. Since $w \neq q$, there is a minimum index m with $w_{[m-1]} = q_{[m-1]}$ and $w_m \neq q_m$. Without loss of generality, let $w_m > q_m$. If $f > m$, then $\tilde{y}^j_{[m]} = y^j_{[m]}$, hence $w_{[m-1]} = y^j_{[m-1]} = q_{[m-1]}$ and $w_m > y^j_m > q_m$. From this, it follows that $\text{Proj}^m(w) \notin \mathcal{B}(S_{\text{init}}, S_{\min}, p)$ or $\text{Proj}^m(q) \notin \mathcal{B}(S_{\text{init}}, S_{\min}, p)$. With

Assumption 2 we have that $w \notin \mathcal{B}(S_{\text{init}}, S_{\min}, p)$ or $q \notin \mathcal{B}(S_{\text{init}}, S_{\min}, p)$, which contradicts $w, q \in \mathcal{B}(S_{\text{init}}, S_f, p)$. Hence, it holds that $f \leq m$. Moreover, $\tilde{y}_t^j = \overline{x}$, $\forall t \in \{f, \dots, d-1\}$ holds. For $m \neq d$, the inequality $\tilde{y}_m^j < q_m$ implies that $q \notin \mathcal{B}(S_{\text{init}}, S_f, p)$. Hence $m = d$, which gives $q_{[d-1]} = w_{[d-1]} = \tilde{y}_{[d-1]}^j$ and $w_d < \tilde{y}_d^j < q_d$. This is, however, impossible as by Algorithm 2 holds $\alpha^d S_{\text{init}} + \sum_{\tau=1}^{d} \alpha^{d-\tau} \tilde{y}_\tau^j \Delta t = S_f$. Using w instead leads to $\alpha^d S_{\text{init}} + \sum_{\tau=1}^{d} \alpha^{d-\tau} w_\tau \Delta t < S_f$. Hence, we conclude $w \notin \mathcal{B}(S_{\text{init}}, S_f, p)$. □

Note that although the corrected extreme actions are vertices, there is no guarantee that they are distinct vertices, i.e., different extreme actions can be corrected to the same vector.

Finally, given a set of extreme directions $\mathcal{J} \subseteq \{-1, 1\}^d$, initial energy levels $S_{\text{init},i}$, minimum final energy levels $S_{f,i}$, and parameter vectors p_i for $i = 1, \dots, n$, Algorithm 3 can be used to calculate the summed corrected extreme actions. In Line 2, the procedure iterates over the devices, whereas in Line 5, it iterates through the extreme directions. The calculations for the extreme actions are carried out in Line 6 and, if necessary, corrected in Line 7. For a specific device, all extreme actions are then organized in a matrix Y in Line 8. These matrices are then summed in Line 11 to obtain the matrix of summed corrected extreme actions. The convex hull of the elements of this matrix finally constitutes an inner approximation to the Minkowski sum of the polytopes $\mathcal{B}(S_{\text{init},i}, S_{f,i}, p_i)$, $i = 1, \dots, n$.

Note that the corrected extreme actions are calculated for each $j \in \mathcal{J}$ and for each $i = 1, \dots, n$, therefore the computational complexity is given by $|\mathcal{J}|n$. The set $\mathcal{J}$ thus

Algorithm 3 (CompleteAlgorithmBESS)

Require: $S_{\text{init},i}, S_{f,i}, p_i, i = 1, \dots, n, \mathcal{J}$
Ensure: V

```
1:  V ← 0_{d×|J|}
2:  for i = 1 to n do
3:      Y ← 0_{d×|J|}
4:      k ← 1
5:      for j ∈ J do
6:          y_i^j ← extremeActionBESS(S_{init,i}, p_i, j)
7:          y_i^j ← correctiveIncreaseBESS(S_{init,i}, S_{f,i}, p_i, y_i^j)
8:          Y[:, k] ← y_i^j
9:          k ← k + 1
10:     end for
11:     V ← V + Y
12: end for
```

controls the accuracy and computational complexity of the proposed approximation strategy. For high-dimensional spaces, e.g., $d > 8$, we propose to select the set of vectors $\mathcal{J}$ by selecting g distinct elements in $\{-1, 1\}^d$, where g must be a function of time periods, e.g., a hypercube has 2^d vertices in d-dimensional space. If all 2^d corrected extreme actions are computed within polytopes that fulfill the Assumptions 1, and 2, then it holds that $\mathbf{0}_d \in \mathcal{A}$, cf. Proposition 3, otherwise $\mathbf{0}_d$ needs to be appended to the matrix V if $\alpha_i^d S_{\text{init},i} \geq S_{f,i}\ \forall i \in \{1, \ldots, n\}$.

Finally, note that the calculations related to the matrix of extreme actions Y_i are independent, enabling parallel computation. Additionally, these computations do not require any commercial packages or optimizations, relying solely on basic instructions, making them suitable for implementation on a standard microcontroller. The complete Python code for the algorithm, including example calculations, is available (Öztürk & Rheinberger, 2024).

4.2.2. Benchmarking

The experiments in this section aim to analyze the computational complexity and accuracy of the proposed algorithm across varying numbers of time periods and devices. To this end, we use the benchmark developed in the previous chapter to compare the proposed strategy against ten state-of-the-art inner approximation methods. First, we establish the number of extreme action computations. Then, the algorithm is tested within the benchmark.

The first experiment aims to determine the size of $\mathcal{J}$, cf. Algorithm 3. For this purpose, UPR values are calculated ten times for tuples $(n, d) \in \{100\} \times \{24, 48, 72, 96\}$ with varying numbers of vectors in $\mathcal{J}$. The median of ten runs, i.e., the median of ten UPR values and computation times, are shown in Fig. 4.5. It can be seen that the UPR values increase with fixed $|\mathcal{J}|$ and increasing number of time periods, but decrease with increasing number of vectors in $|\mathcal{J}|$ and fixed number of time periods. This reflects the fact that the vertices increase as the number of time periods increases, e.g., a hypercube has 2^d vertices in d-dimensional space, implying that the size of $\mathcal{J}$ must be chosen as a strictly increasing function of time periods. The peak power UPR in Fig. 4.5 is reduced quickly, while the cost UPR is reduced more slowly. The computation time increases linearly with increasing size of $\mathcal{J}$, which reflects the fact that extreme actions are computed for each vector in $\mathcal{J}$. At most, we have computation times of less than 10 seconds for experiments with up to 96 time periods and 100 devices. Thus, the maximum in Fig. 4.5, i.e., 8000 or 10000 extreme

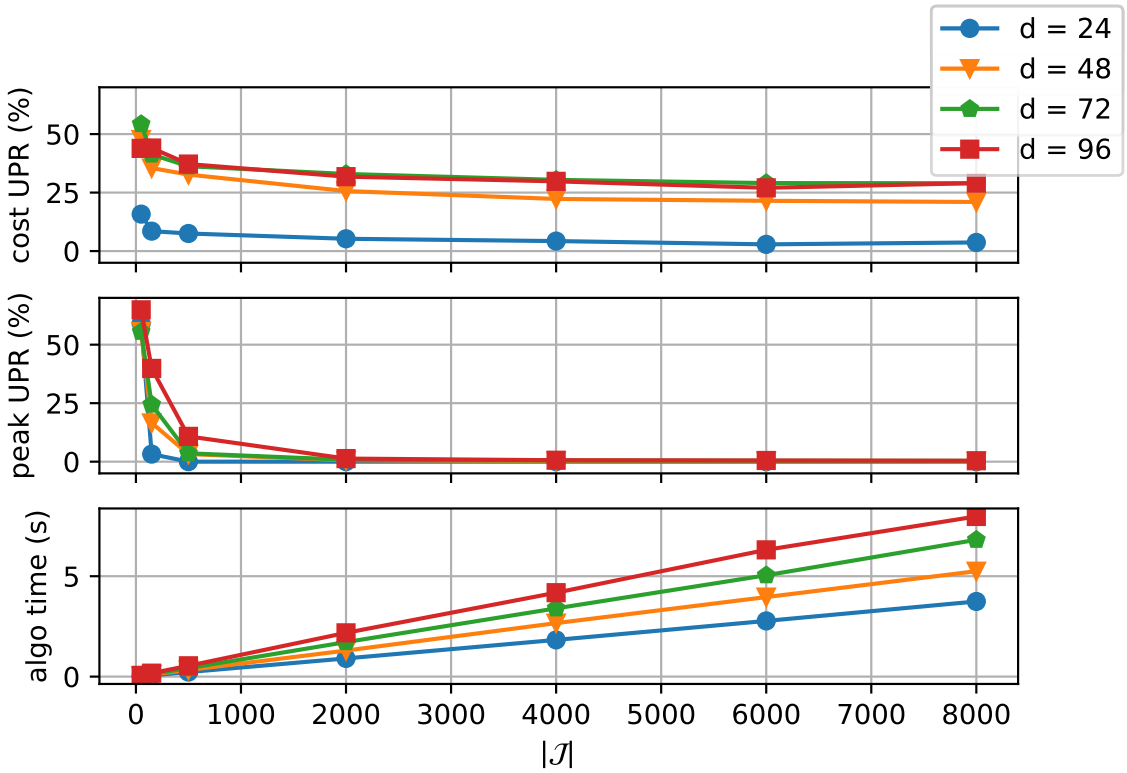

Figure 4.5.: UPR values and computation times for tuples $(n, d) \in \{100\} \times \{24, 48, 72, 96\}$ with increasing number of vectors in $\mathcal{J}$.

directions, could be chosen. Putting these findings together, we choose a quadratic time dependence, i.e., $|\mathcal{J}| = d^2$. Note that the vectors in $\mathcal{J}$ are selected at random, and only those extreme actions that correspond to the vectors in $\mathcal{J}$ are computed. Moreover, with this choice, the complexity of Algorithm 3 becomes $|\mathcal{J}|n = d^2 n$, i.e., quadratic in time periods and linear in the number of devices.

For the next experiment conducted in this section, the UPR values are computed for tuples $(n, d) \in \{2, 6, 10, 20, 30\} \times \{4, 8, 12, 16, 20, 24\}$ according to the benchmark description in Section 3.2.2. Table 4.1 shows the maximum, i.e., worst case, peak, and cost UPR as well as the maximum computation time across tuples $(n, d) \in \{2, 6, 10, 20, 30\} \times \{4, 8, 12, 16, 20, 24\}$.

In addition, Fig. 4.6 displays the results for tuples $(n, d) \in \{30\} \times \{4, 8, 12, 16, 24\}$ in the first column and for tuples $(n, d) \in \{2, 6, 10, 20, 30\} \times \{24\}$ in the second column, i.e., with fixed number of devices and varying time periods and fixed number of time periods and varying number of devices, respectively. The cost UPR values are shown in the first row, the peak UPR values in the second row and the computation times in the third row.

Table 4.1.: Max UPR values and max calculation time for 4, 8, 12, 16, 20, 24 time periods and 2, 6, 10, 20, 30 batteries for different inner approximation methods. Table reproduced (Öztürk, Faulwasser, et al., 2024).

Algorithm	Reference	Time (s)	UPR (%)	
			Peak	Cost
Cuboid Homothets Stage 0	Nazir et al., 2018	3.05	178.22	27.81
Battery Homothets	Zhao et al., 2017	-	-	-
Battery Homothet Projection with LDR	Zhao et al., 2016	-	-	-
Zonotopes l_∞	Müller et al., 2015	233.74	284.77	33.27
Zonotopes l_1	Müller et al., 2015	250.19	190.98	27.78
Zonotopes l_2	Müller et al., 2015	92.83	121.53	30.83
Zonotopes weighted	Müller et al., 2019	88.57	187.18	21.14
Cuboid Homothets Stage 1	Nazir et al., 2018	161.44	176.67	26.56
Ellipsoid Projection with LDR	Zhen and den Hertog, 2018	173.16	20.21	61.66
Ellipsoid Projection	Barot, 2017	-	-	-
Proposed Vertex Generation	Öztürk, Faulwasser, et al., 2024	**0.24**	**5.46**	**7.91**

Our proposed algorithm (*Vertex Generation*) demonstrates superior performance in terms of accuracy (peak and cost UPR) and computation time compared to the other algorithms, as shown in Table 4.1 and Fig. 4.6. The data in Table 4.1 also show that our *Vertex Generation* and the *Ellipsoidal projection with LDR* algorithms are the only ones that achieve peak UPR values below 100%. This indicates that $\mathbf{0}_d$, which corresponds to no flexibility, performs better than the flexibility set provided by the other algorithms. Additionally, Table 4.1 shows that certain algorithms exhibit objective-dependent performance. For instance, the *Ellipsoid Projection with LDR* algorithm ranks second in peak UPR but performs the poorest in cost UPR. This is in contrast to our *Vertex Generation* algorithm, which achieves the best results for both peak and cost UPR. Furthermore, most current algorithms are limited in their computational complexity, making them impractical for longer time periods, e.g., day-ahead computations when 15-minute intervals are considered (Öztürk et al., 2022).

However, our algorithm allows for day-ahead computation, as shown in Table 4.2. For this experiment, UPR values are calculated for tuples $(n, d) \in \{50, 100, \ldots, 500\} \times \{12, 24, \ldots, 96\}$. The maximum cost UPR value across all tuples is 35.78, the maximum peak UPR is 6.71 and the maximum calculation time is about 3 minutes. The highest UPR value achieved in this experiment, using up to 500 devices and 96 time periods, surpasses

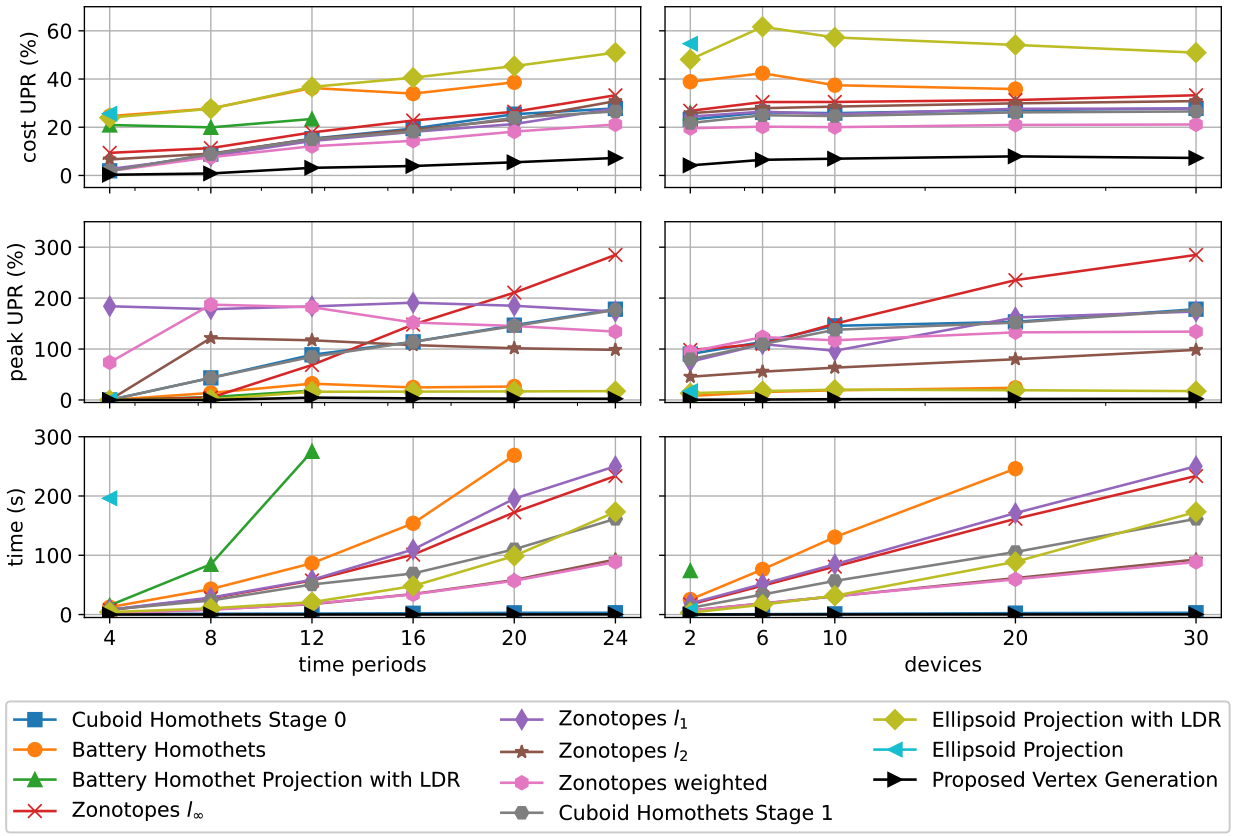

Figure 4.6.: Results for experiments with tuples $(n, d) \in \{30\} \times \{4, 8, 12, 16, 24\}$ in the first column and experiments with tuples $(n, d) \in \{2, 6, 10, 20, 30\} \times \{24\}$ in the second column. The cost UPR values are shown in the first row, the peak UPR values in the second row, and the calculation times in the third row. Figure reproduced (Öztürk, Faulwasser, et al., 2024).

that of the other algorithms tested in the previous experiment, which only involved up to 30 devices and 24 time periods (cf. Table 4.1). Additionally, the cost UPR value obtained in this experiment outperforms that of one of the other algorithms tested in the previous experiment.

Lastly, we analyze the proposed method in contrast to utilizing a central controller as outlined in (3.21). When using a central controller, the data is directly transmitted to the optimizing instance without passing through an aggregator. Consequently, all data is accessible to the optimizing instance, which manages the devices and communicates with them directly. In addition, the central control scheme works within the exact feasible region, i.e., with perfect accuracy.

In an experiment with 500 devices and 96 time periods, the centralized controller achieved computation times of 54.98 seconds for the cost objective and 96.58 seconds for the peak

Table 4.2.: Maximum UPR values and maximum computation time for 12, 24, . . ., 96 time periods and 50, 100, . . ., 500 batteries. Table reproduced (Öztürk, Faulwasser, et al., 2024).

Algorithm	Reference	Time (s)	UPR (%)	
			Peak	Cost
Proposed Vertex Generation	Öztürk, Faulwasser, et al., 2024	**204.43**	**6.71**	**35.78**

objective. Gurobi (Gurobi Optimization, LLC, 2024) was used to solve the centralized optimization problem. Although this approach achieves better performance than the proposed method (cf. Table 4.2), centralized control is impractical due to privacy concerns, the purchasing entity's need to maintain control, and the associated communication overhead.

4.3. Extension to Convex Storage Models

In this section, the aggregation strategy is further extended to incorporate convex storage models. The extended algorithm is first introduced, followed by its application in numerical experiments to demonstrate its accuracy and computational complexity.

4.3.1. Aggregation Strategy for Convex Storage Models

As outlined in Chapter 2, a wide range of practical devices can be mapped to the convex storage model, cf. (2.17). Therefore, an algorithm capable of aggregating such a variety of devices is desirable.

The algorithms developed so far have been successful in aggregating BESSs that maintain a certain minimum final energy level. One of the main advantages of models such as BESS is that they generally have constant energy and power limits, with exceptions only occurring in the final time period, a scenario that was addressed in the previous chapter through a corrective algorithm. In contrast, convex storage models usually have varying energy and power limits. For instance, the power limits of EVs are zero when not connected to a charging station and between the maximum and minimum power limits when connected. Moreover, the trip consumption of EVs and the demand associated with TCLs result in variable energy limits, cf. Table 2.1. Therefore, it is essential to extend the previously

Algorithm 4 (CorrectedExtremeAction)

Require: $\underline{x}, \overline{x}, \underline{S}, \overline{S}, S_{\text{init}}, \alpha, j \in \{-1, 1\}^d$
Ensure: y^j
1: $y^j \leftarrow \mathbf{0}_d$
2: **for** $t = 1$ **to** d **do**
3: **if** $j_t = 1$ **then**
4: $y_t^j \leftarrow \max\left(\min\left(\overline{x}_t, \frac{\overline{S}_t - (\alpha^t S_{\text{init}} + \sum_{\tau=1}^{t-1} \alpha^{t-\tau} y_\tau^j \Delta t)}{\Delta t}\right), \underline{x}_t\right)$
5: $y^j \leftarrow \text{correctiveIncrease}(y^j, \underline{x}, \overline{x}, \underline{S}, \overline{S}, S_{\text{init}}, \alpha, t)$
6: $y^j \leftarrow \text{correctiveDecrease}(y^j, \underline{x}, \overline{x}, \underline{S}, \overline{S}, S_{\text{init}}, \alpha, t)$
7: **else if** $j_t = -1$ **then**
8: $y_t^j \leftarrow \min\left(\max\left(\underline{x}_t, \frac{\underline{S}_t - (\alpha^t S_{\text{init}} + \sum_{\tau=1}^{t-1} \alpha^{t-\tau} y_\tau^j \Delta t)}{\Delta t}\right), \overline{x}_t\right)$
9: $y^j \leftarrow \text{correctiveIncrease}(y^j, \underline{x}, \overline{x}, \underline{S}, \overline{S}, S_{\text{init}}, \alpha, t)$
10: $y^j \leftarrow \text{correctiveDecrease}(y^j, \underline{x}, \overline{x}, \underline{S}, \overline{S}, S_{\text{init}}, \alpha, t)$
11: **end if**
12: **end for**

developed algorithm to account for non-constant energy and power limits, which will be achieved below through additional corrective algorithms.

The corrected extreme actions within the convex storage model can be computed via Algorithm 4. The extensions compared to the previous procedure, cf. Algorithm 1, are twofold: First, additional max and min operators are inserted in Lines 4 and 8 to ensure the power constraints, and second, two correction algorithms are introduced in Lines 5, 6, 9, and 10

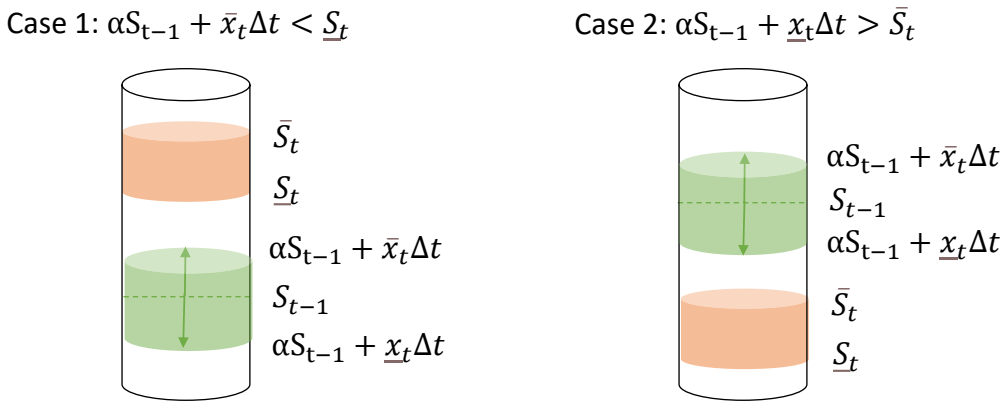

Figure 4.7.: Example cases, in which the energy constraints are violated. Left: the case in which the energy in the storage is lower than the lower limit, while right: the energy in the storage is higher than the upper limit. Figure reproduced (Öztürk, Kaspar, et al., 2024).

Algorithm 5 (correctiveIncrease)

Require: $y^j, \underline{x}, \overline{x}, \underline{S}, \overline{S}, S_{\text{init}}, \alpha, t$
Ensure: y^j
1: **if** $\underline{S}_t > \alpha^t S_{\text{init}} + \sum_{\tau=1}^{t} \alpha^{t-\tau} y_\tau^j \Delta t$ **then**
2: init i ▷ first index with $\overline{x}_i > 0$ in reversed $\overline{x}_{[t]}$
3: $y_{t-i+1}^j \leftarrow \min\left(\max\left(\underline{x}_{t-i+1}, \frac{\underline{S}_t - (\alpha^t S_{\text{init}} + \sum_{\tau=1}^{t-1} \alpha^{t-\tau} y_\tau^j \Delta t)}{\Delta t}\right), \overline{x}_{t-i+1}\right)$
4: $k \leftarrow t - i$
5: **while** $\underline{S}_t \neq \alpha^t S_{\text{init}} + \sum_{\tau=1}^{t} \alpha^{t-\tau} y_\tau^j \Delta t$ **do**
6: **for** $l = k$ to $t - i$ **do**
7: $y_l^j \leftarrow \max\left(\min\left(\overline{x}_l, \frac{\overline{S}_l - (\alpha^l S_{\text{init}} + \sum_{\tau=1}^{l-1} \alpha^{l-\tau} y_\tau^j \Delta t)}{\Delta t}\right), \underline{x}_l\right)$
8: **end for**
9: $y_{t-i+1}^j \leftarrow \min\left(\max\left(\underline{x}_{t-i+1}, \frac{\underline{S}_t - (\alpha^t S_{\text{init}} + \sum_{\tau=1}^{t-1} \alpha^{t-\tau} y_\tau^j \Delta t)}{\Delta t}\right), \overline{x}_{t-i+1}\right)$
10: $k \leftarrow k - 1$
11: **end while**
12: **end if**

to account for the varying energy constraints.

The two cases in which correction is required are illustrated in Fig. 4.7. The case where the energy in the storage is less than the lower bound, left in Fig. 4.7, is handled via the corrective algorithm to increase the energy, cf. Algorithm 5. In Line 2, i is initialized with the first index of reversed $\overline{x}_{[t]}$, where $\overline{x}_i > 0$ applies. This step is mandatory to identify the last index prior that allows for correction, i.e., positive power. Correction to increase the energy must be carried out with strictly positive values and is not possible with indices $l > t - i + 1$, as they are either zero or negative due to $\overline{x}_l \leq 0$. The power profile is then changed backwards without constraint violations, starting with the index $t - i + 1$ to reach $\underline{S}_t$. Lines 3, 7, and 9 enforce the power constraints, i.e., it holds that $\underline{x}_{t-i+1} \leq y_{t-i+1}^j \leq \overline{x}_{t-i+1}$. In Line 7, y^j is modified to increase the energy in the storage. Moreover, the equality in Line 5 is enforced by the choice

$$\frac{\underline{S}_t - (\alpha^t S_{\text{init}} + \sum_{\tau=1}^{t-1} \alpha^{t-\tau} y_\tau^j \Delta t)}{\Delta t}$$

in Lines 3 and 9 as

$$\alpha^t S_{\text{init}} + \sum_{\tau=1}^{t-1} \alpha^{t-\tau} y_\tau^j \Delta t + \frac{\underline{S}_t - (\alpha^t S_{\text{init}} + \sum_{\tau=1}^{t-1} \alpha^{t-\tau} y_\tau^j \Delta t)}{\Delta t} \Delta t = \underline{S}_t.$$

Algorithm 6 (correctiveDecrease)

Require: $y^j, \underline{x}, \overline{x}, \underline{S}, \overline{S}, S_{\text{init}}, \alpha, t$
Ensure: y^j
1: **if** $\overline{S}_t < \alpha^t S_{\text{init}} + \sum_{\tau=1}^{t} \alpha^{t-\tau} y_\tau^j \Delta t$ **then**
2: init i ▷ first index with $\underline{x}_i < 0$ in reversed $\underline{x}_{[t]}$
3: $y_{t-i+1}^j \leftarrow \max\left(\min\left(\overline{x}_{t-i+1}, \frac{\overline{S}_t - (\alpha^t S_{\text{init}} + \sum_{\tau=1}^{t-1} \alpha^{t-\tau} y_\tau^j \Delta t)}{\Delta t}\right), \underline{x}_{t-i+1}\right)$
4: $k \leftarrow t - i$
5: **while** $\underline{S}_t \neq \alpha^t S_{\text{init}} + \sum_{\tau=1}^{t} \alpha^{t-\tau} y_\tau^j \Delta t$ **do**
6: **for** $l = k$ to $t - i$ **do**
7: $y_k^j \leftarrow \min\left(\max\left(\underline{x}_l, \frac{\underline{S}_l - (\alpha^l S_{\text{init}} + \sum_{\tau=1}^{l-1} \alpha^{l-\tau} y_\tau^j \Delta t)}{\Delta t}\right), \overline{x}_l\right)$
8: **end for**
9: $y_{t-i+1}^j \leftarrow \max\left(\min\left(\overline{x}_{t-i+1}, \frac{\overline{S}_t - (\alpha^t S_{\text{init}} + \sum_{\tau=1}^{t-1} \alpha^{t-\tau} y_\tau^j \Delta t)}{\Delta t}\right), \underline{x}_{t-i+1}\right)$
10: $k \leftarrow k - 1$
11: **end while**
12: **end if**

The second case, right in Fig. 4.7, where the energy in the storage is greater than $\overline{S}_t$, is covered similarly in the corrective algorithm to decrease the energy, cf. Algorithm 6. These corrections are called, if necessary, in Lines 5, 6, 9, and 10 of Algorithm 4.

The summed corrected extreme actions can then be computed via Algorithm 7. The corrected extreme actions are computed in Line 6 within a for loop for fixed devices and written to the matrix Y in Line 7. These matrices are then subsumed in Line 10 to obtain

Algorithm 7 (CompleteAlgorithm)

Require: $\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, S_{\text{init, i}}, \alpha_i \; i = 1, \ldots, n, \mathcal{J}$
Ensure: V
1: $V \leftarrow \mathbf{0}_{d \times |\mathcal{J}|}$
2: **for** $i = 1$ **to** n **do**
3: $Y \leftarrow \mathbf{0}_{d \times |\mathcal{J}|}$
4: $k \leftarrow 1$
5: **for** $j \in \mathcal{J}$ **do**
6: $y_i^j \leftarrow$ CorrectedExtremeAction$(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, S_{\text{init,i}}, \alpha_i, j)$
7: $Y[:, k] \leftarrow y_i^j$
8: $k \leftarrow k + 1$
9: **end for**
10: $V \leftarrow V + Y$
11: **end for**

the matrix of summed corrected extreme actions.

Since the corrected extreme actions are calculated for each device $|\mathcal{J}|$ times, the complexity is $n|\mathcal{J}|$, i.e., linear in the number of devices and dependent upon $|\mathcal{J}|$ in the number of time periods. Note that the extreme actions computed within convex storage models have not been proven to be vertices. Furthermore, these extreme actions are in general non-unique. This distinguishes them from the extreme actions computed within polytopes that fulfill the Assumptions 1 and 2, which are distinct vertices of the Minkowski sum, cf. Theorem 1, as well as from the extreme actions computed within the BESS, which are vertices in the individual polytopes, cf. Theorem 2, but not necessarily distinct. In the following, this extended algorithm will be used in numerical experiments to demonstrate its accuracy and computational complexity.

4.3.2. Use Case – One Aggregator and One Flexibility Type

In our first experiment, we consider a residential area with 300 residents, where 90 individuals, i.e., 30%, own a Nissan Leaf, cf. Table 5.2. We assume that the vehicles are

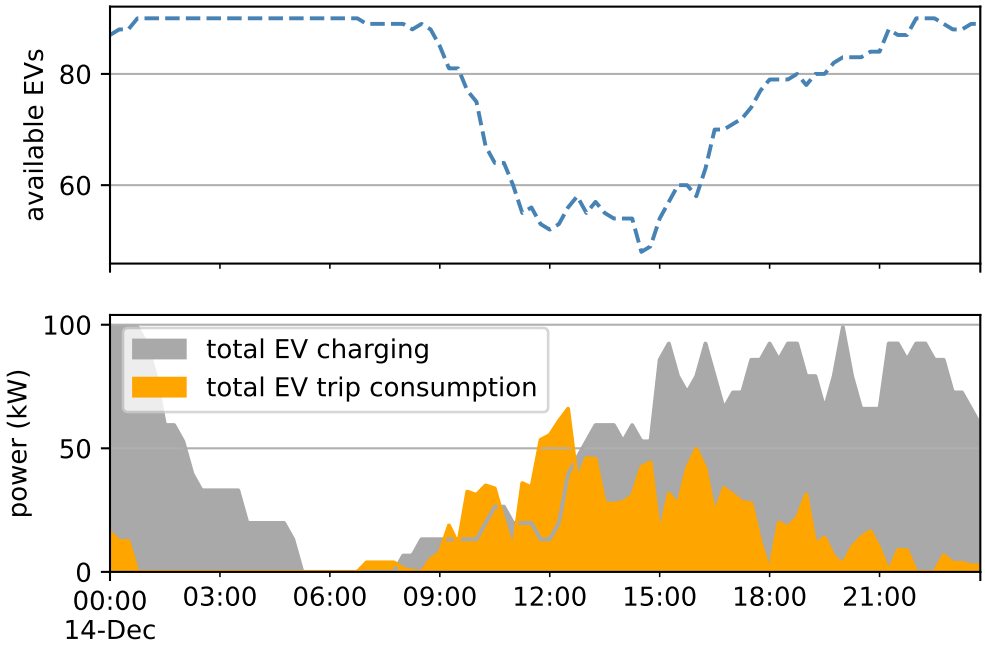

Figure 4.8.: Visualization of EV data. Top: Available EVs. Bottom: Total EV trip consumption (orange) and total uncontrolled EV charging (gray).

equipped with V2G technology and that the owners of the Nissan Leafs have agreed to allow the aggregator to control the charging and discharging of their EVs. The aggregator calculates the aggregated flexibility offered by the EVs and communicates the information to a grid operator, who uses it to reduce the residential area's peak demand. We assume a discrete-time setting of one day divided into 15-minute intervals.

The data for the household demand, the EV trip consumption, the EV availability, and EV charging are taken from (My Electric Avenue, 2015; Commission for Energy Regulation, 2012). Both data sets were resampled to quarter-hourly data and time-synchronized by a publicly available code (Rheinberger, 2021a). Figure 4.8 shows the aggregated EV data, with the total number of available EVs at the top and the total charging power together with the total trip consumption at the bottom.

At start, the EV batteries are charged to 50%. At the end of the operating window, the EVs must retain a minimum energy level, which is determined by subtracting the trip consumption from the initial energy and adding the charged energy given by the data.

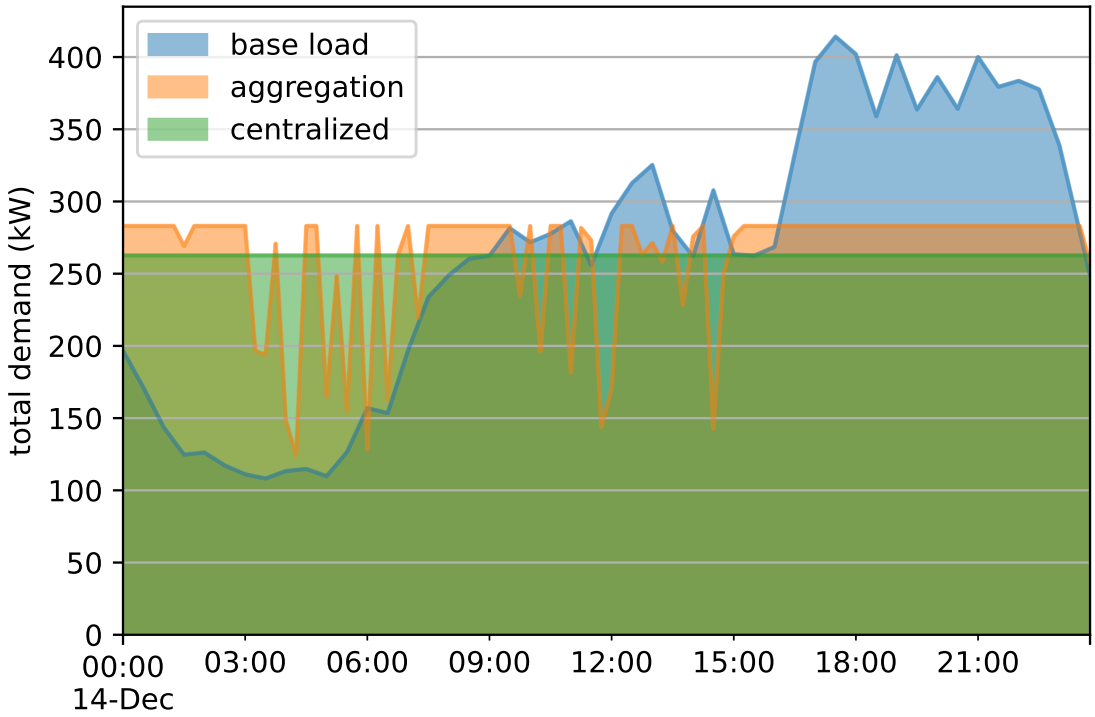

Figure 4.9.: Residential loads: the base load is shown in blue, the aggregated controlled load in orange, and the central controlled load in green.

To this end, the proposed aggregation-based approach is compared with a central controller. Similar to the experiments in Chapter 4, we use a quadratic time dependence for the extreme directions, i.e., $|\mathcal{J}| = d^2$ in Algorithm 7.

Figure 4.9 shows the total residential demand compared to the load curve achieved by the centralized, as well as the aggregation-based approach. The peak demand is effectively reduced to 283.08 kW by the proposed method and to 262.68 kW by the centralized approach. The calculation times are 35 seconds and 5 seconds for the proposed approach and the centralized approach, respectively.

Please note that although the centralized approach performs slightly better, it compromises privacy and requires additional communication effort. Our approach, on the other hand, masks the data and reduces the communication effort significantly, i.e., only the aggregator communicates with the purchaser of the ancillary services. In addition, the proposed method can be implemented on a simple microcontroller, while the central controller requires special, possibly commercial, software.

4.3.3. Use Case – Multiple Aggregators and Multiple Flexibility Types

In our next experiment, we consider three residential areas, each with 100 residents and local aggregators. Sensitive information within the local community is restricted and only

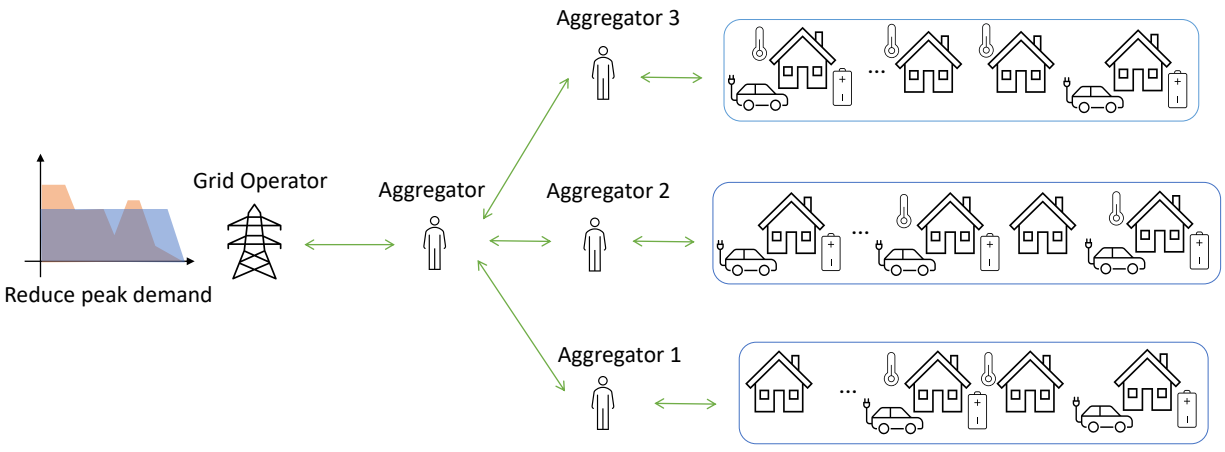

Figure 4.10.: Scenario with three residential areas and three local aggregators. The residents may own EVs, BESSs, and TCLs (air conditioning). The collective local flexibility is passed to a global aggregator, which in turn aggregates and forwards them to a grid operator for ancillary services.

known within the local community, i.e., it is not passed to third parties outside the residential area. The residents have signed contracts with local aggregators and can own BESSs, EVs, and TCLs (air conditioning). The collective flexibility of these devices is aggregated by the local aggregators and passed to a global aggregator. The global aggregator aggregates the local aggregated flexibilities and offers the total flexibility to a system operator for ancillary services, cf. Fig. 4.10.

The used system parameters are listed in Table 5.1 for the BESS, in Table 5.2 for the EVs, and in Table 5.3 for the air-conditioning systems. The devices are selected at random from these pre-programmed options. The data for the household demand, the EV trip consumption, the EV availability, and EV charging are taken from (My Electric Avenue, 2015; Commission for Energy Regulation, 2012). Both data sets were resampled to quarter-hourly data and time-synchronized by a publicly available code (Rheinberger, 2021a).

At the beginning, the EV batteries are charged to 50%, while the initial energy of the BESS is randomly selected from the interval $[\frac{1}{2}S_{\text{max,i}}, S_{\text{max,i}}]$. At the end, EVs must maintain a minimum energy level, which is determined by subtracting the trip consumption from the initial energy and adding the charged energy given by the data. The BESS, on the other hand, must retain the initial energy level at the end of the operating window.

We assume an ambient temperature of $30°C$. The initial temperature and dead band of air conditioners are selected uniformly from the intervals $[19, 20]$ (°C) and $[3, 5]$ (°C), respectively. The setpoint temperatures are set to 20°C for the air conditioners.

We define the penetration of a device in a residential area as the percentage of residents that own this device. An EV penetration of 38%, cf. Parliament, 2019, a TCL penetration of 19%, cf. IEA, 2023, and a BESS penetration of 1%[6], is assumed.

To this end, we compare the proposed hierarchical aggregation technique with a central controller that utilizes the exact feasible region. A quadratic time dependence is used in Algorithm 7 for the extreme directions, i.e., $|\mathcal{J}| = d^2$.

Figure 4.11 displays the total uncontrolled demand of the residential areas compared to the load curve obtained with both the central controller and the aggregation-based approach. The peak demand is effectively reduced to 277.15 kW by the proposed method and to 262.68 kW by the central controller. The calculation times are 44.5 seconds and 14.6 seconds for the proposed approach and the central controller, respectively. Note that although

[6] 650,000 BESS installations from 2023 to 2022 in Germany, cf. Figgener et al., 2022 divided by the population in Germany $\approx 1\%$.

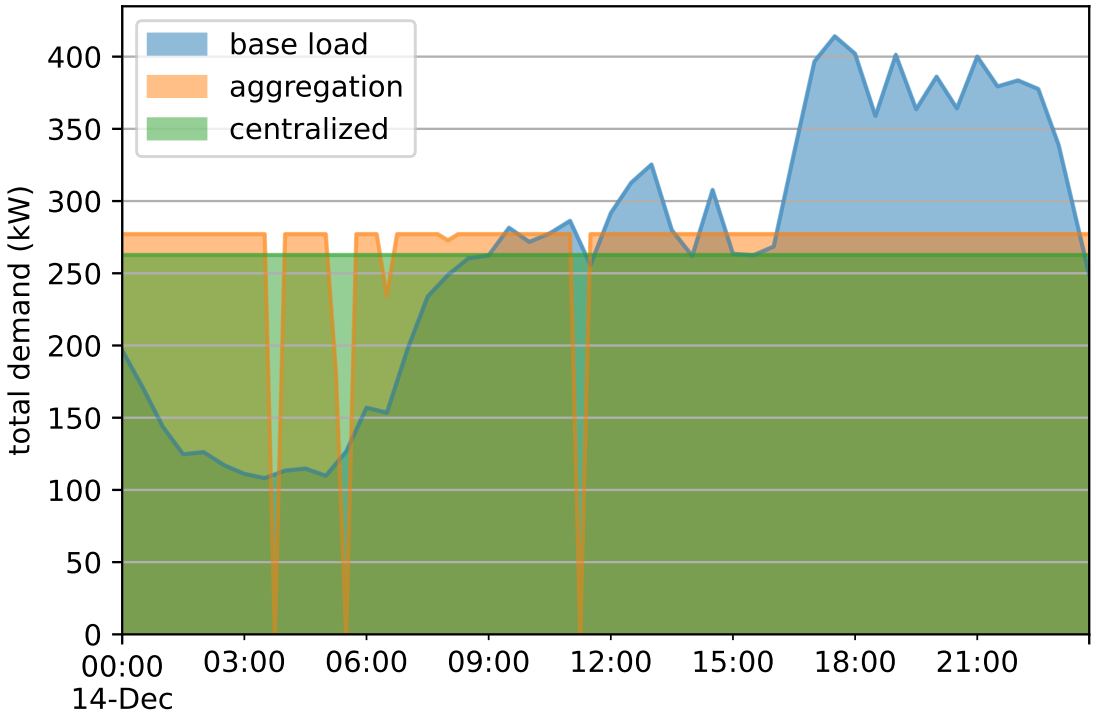

Figure 4.11.: Residential loads: the base load is shown in blue, the aggregate controlled load in orange, and the central controlled load in green.

the central controller performs better, it is only used for comparison purposes and is not practical due to the lack of data security, and the increased communication effort for large-scale problems.

4.3.4. Scalability

In this section, the algorithm's scalability is demonstrated across different time periods and devices. We consider a discrete-time scenario of one day divided into 15-minute intervals. Our analysis focuses on residential areas where aggregators manage contracts with residents. These residents may own EVs, BESSs, and TCLs (air conditioning systems) that can be controlled by the aggregator. The aggregator aggregates these flexibilities before forwarding them to the grid operator for ancillary service purposes. The selection of data for the devices is carried out as outlined in the preceding section.

We assess the effectiveness of the proposed algorithm via the modified unused potential

ratio (UPR_M)

$$\mathrm{UPR}_\mathrm{M} := \frac{z_{\text{approx}} - z_{\text{exact, best}}}{z_{\text{exact, worst}} - z_{\text{exact, best}}}, \tag{4.6}$$

where z_{approx} is the solution of an optimization within the feasible region defined by the proposed algorithm, $z_{\text{exact, best}}$ denotes the solution within the exact feasibility region computed by a central controller, and $z_{\text{exact, worst}}$ denotes the solution within the exact feasibility region when the objective is maximization rather than minimization.

Note that we use the UPR_M rather than the UPR (cf. (3.22)), since $z_{\text{no flex}}$ within the UPR necessitates $\mathbf{0}_d$ to be within the feasibility region, which is typically not guaranteed for convex storage models. With this modification, we note that the UPR_M is limited to a range of $0 \leq \mathrm{UPR}_\mathrm{M} \leq 100\%$. A UPR_M value near zero suggests that the approximation closely matches the output of the central controller. In contrast, when the UPR_M value nears 100%, it indicates that a considerable amount of potential within the exact feasibility region is left unused and not represented by the approximation.

In our experiments, we examine tuples

$$(n, d) \in \{50, 100, 250, 500, 750, 1000\} \times \{24, 48, 72, 96\},$$

representing a maximum of 1000 devices and up to 96 time periods. To account for variability, we compute the UPR_M five times for each tuple (n, d) and use the median of these

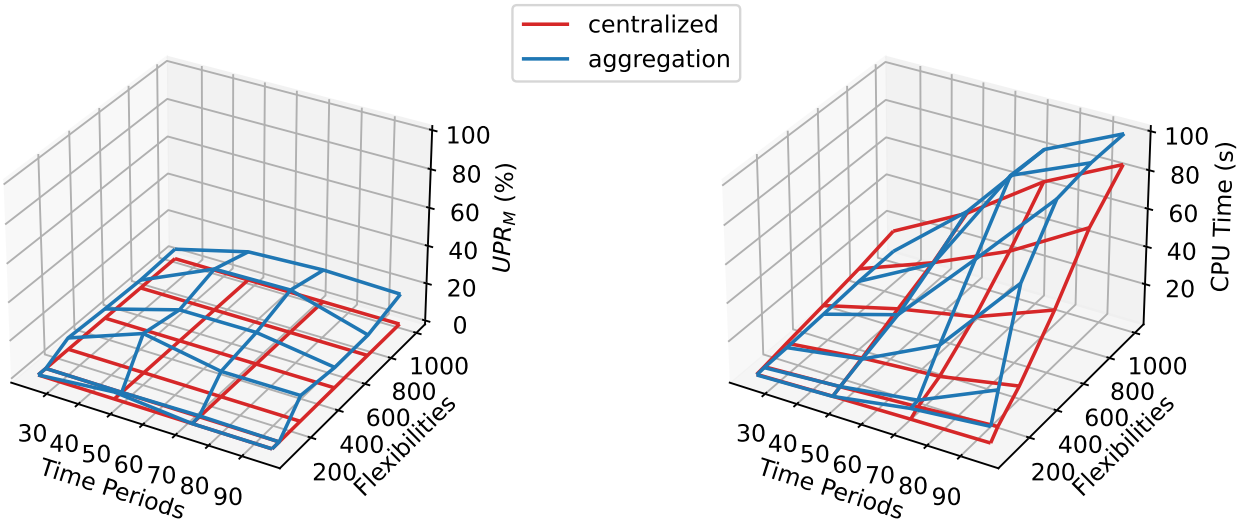

Figure 4.12.: Comparison of modified UPR (UPR_M) and CPU time across settings $(n, d) \in \{50, 100, 250, 500, 750, 1000\} \times \{24, 48, 72, 96\}$. The results for the central control approach are shown in red, while those for the proposed method are illustrated in blue.

values. Furthermore, a quadratic time dependence is employed for the set of extreme directions, i.e., $|\mathcal{J}| = 2^d$. Our objective within the optimizations is to reduce the total peak load in the residential area.

The results are shown in Fig. 4.12. The proposed approach achieves a maximum $\mathrm{UPR_M}$ of 22.76% and a computation time of 100.56 seconds. In contrast, the central control records a maximum $\mathrm{UPR_M}$ of zero, as the central controller operates within the exact feasibility region, with a maximum computation time of 84.7 seconds.

Note that, unlike the proposed aggregation strategy, the central controller requires complete information and manages all devices directly. While this enables the use of the exact feasible region, it also introduces challenges such as privacy concerns and excessive communication overhead. These experiments demonstrate the scalability of our approach across different time periods and devices, addressing a well-documented issue with current approximation techniques.

4.4. Disaggregation Strategy for Convex Storage Models

This section concludes the aggregation-based control problem by introducing a novel method for decomposing (disaggregating) given power profiles in the aggregated set into their components in the individual sets. Disaggregation is the necessary second component of aggregation-based control and is carried out by the aggregator in order to set power profiles in the individual devices.

The aggregation for polytopes $\mathcal{P}_i \subset \mathbb{R}^d, i = 1, \ldots, n$ can be described as a mapping

$$f : \mathcal{P}_1 \times \mathcal{P}_2 \times \cdots \times \mathcal{P}_n \mapsto \mathbb{R}^d, \; f(x_1, \ldots, x_n) \coloneqq \sum_{i=1}^{n} x_i.$$

Disaggregation represents the inverse operation to aggregation and is in general a non-unique operation, as different vectors can be combined to form the same vector in the aggregated set. In mathematical terms, this means that the mapping above is not injective. Therefore, optimization problems are usually formulated and solved to obtain the components of a certain power profile in the aggregated set, cf. Barot, 2017; Müller et al., 2015. This is, however, computationally expensive in higher-dimensional spaces and not necessary for the proposed method.

Given the approximate quantification of the aggregated flexibility, i.e., the matrix of summed corrected extreme actions V, the aggregator offers the aggregated flexibility on energy markets. The purchaser selects a desired power profile from the set of possible power profiles, i.e., the following optimization problem is solved

$$\min_{\alpha,x} f(x) \tag{4.7a}$$

$$\text{subject to } x = V\alpha \tag{4.7b}$$

$$\mathbf{1}_{|\mathcal{J}|}^{\top}\alpha = 1 \tag{4.7c}$$

$$\alpha_j \geq 0, \forall j \in \mathcal{J}. \tag{4.7d}$$

The weight vector $\alpha \in \mathbb{R}^{|\mathcal{J}|}$ is then sent back to the aggregator. The aggregator then needs to distribute (disaggregate) the chosen power profile to the individual devices. Using the identity $V = \sum_{i=1}^{n} Y_i$, (4.7b) can be rewritten as

$$x = V\alpha = \left(\sum_{i=1}^{n} Y_i\right)\alpha = \sum_{i=1}^{n} Y_i\alpha.$$

Hence, the contribution of device i is $Y_i\alpha$, which must be adjusted by the individual devices. Note that the optimizing instance has no access to the device's parameters and only receives the matrix of summed corrected extreme actions V, which ensures privacy. Moreover, the proposed strategy can be considered fair, as each device receives the same vector α.

Algorithm 8 outlines a procedure for computing the disaggregated power profiles of the individual devices. In Line 2, the algorithm iterates through each device, placing the contribution of device i, i.e., $Y_i\alpha$ into the i^{th} column of the matrix P (Line 3). On completion, the matrix P is returned, which contains the contribution of all devices in its columns.

Finally, note that the derived expression remains unchanged for scenarios with multiple aggregators (hierarchical aggregation settings), as each aggregation level can be represented by the summation of the previous levels. By applying this rule successively, the matrix

Algorithm 8 (Disaggregation)

Require: $\alpha, Y_i, i = 1, \ldots, n$
Ensure: P
1: $P \leftarrow \mathbf{0}_{d\times n}$
2: **for** $i = 1$ **to** n **do**
3: $\quad P[:, i] \leftarrow Y_i\alpha$
4: **end for**

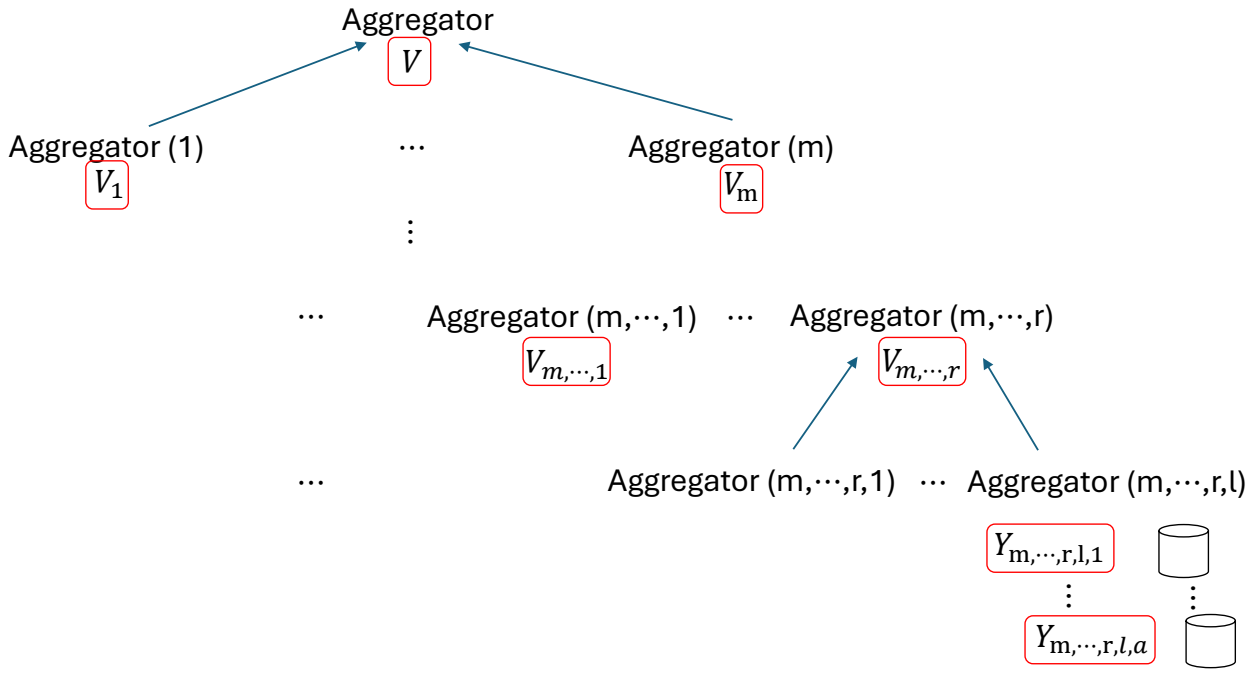

Figure 4.13.: Typical configuration for a hierarchical aggregation setting, where each aggregator is associated with a matrix that represents the summed extreme actions. At the bottom, storage devices are displayed along with their corresponding device matrices. Figure reproduced (Öztürk et al., 2025).

corresponding to the top-level aggregator V can be represented by the summation of the device-level matrices.

Figure 4.13 illustrates a typical hierarchical aggregation structure, where the number of indices corresponds to the hierarchical level: no indices for the top level, one index for the subsequent level, etc. The matrix V corresponding to the top-level aggregator can be expressed as $V = \sum_{i=1}^{m} V_i$. Following the path to the device level aggregators, this can be re-expressed as

$$V = \sum_{i=1}^{m} \cdots \sum_{k=1}^{r} \sum_{h=1}^{l} \sum_{f=1}^{a} Y_{i,\cdots,k,h,f}.$$

Using the same reasoning as above, we obtain

$$x = V\alpha = \left(\sum_{i=1}^{m} \cdots \sum_{k=1}^{r} \sum_{h=1}^{l} \sum_{f=1}^{a} Y_{i,\cdots,k,h,f} \right) \alpha = \sum_{i=1}^{m} \cdots \sum_{k=1}^{r} \sum_{h=1}^{l} \sum_{f=1}^{a} Y_{i,\cdots,k,h,f} \alpha.$$

Hence, the contribution of a specific device is given by the device's matrix multiplied by the weight vector α.

4.5. Summary

This chapter introduced a novel (dis-)aggregation strategy for convex storage models. In Section 4.1, we presented a method specifically designed for polytopes that satisfy two particular assumptions. This procedure not only demonstrates that the calculations yield distinct vertices of the Minkowski sum but also offers a novel approach for determining a subset of Minkowski sum vertices.

Section 4.2 expanded the strategy to incorporate BESS with arbitrary final energy restrictions. The extended algorithm was benchmarked against ten state-of-the-art algorithms, showcasing its superiority in computational complexity and accuracy across various objectives. Moreover, the proposed algorithm effectively addressed significant issues found in existing methods, including objective-dependent performance, computational limitations, and the exclusion of nominal power profiles such as the idle state.

Section 4.3 further broadened the strategy to encompass convex storage models relevant to a variety of real-world applications, with numerical examples validating its computational efficiency and accuracy.

Lastly, Section 4.4 concluded the discussion on aggregation-based control by introducing an efficient disaggregation strategy.

Overall, the proposed strategy is efficient, accurate, overcomes the limitations of current state-of-the-art algorithms, and outperforms selected algorithms in the developed benchmark. All findings presented in this chapter have been published (Öztürk, Faulwasser, et al., 2024; Öztürk, Kaspar, et al., 2024; Öztürk et al., 2025).

5. Software Implementation

This chapter brings together the findings and algorithms from the previous chapters to introduce a comprehensive software for the aggregation and disaggregation of convex storage models. Initially, the software structure is presented through class diagrams, followed by a discussion of the software architecture. Finally, to further clarify the concepts presented, simple code examples are provided and discussed in detail.

The software is implemented in Python 3 and licensed under the MIT license. The Python package is available for download from the Python Project Index (PyPI), cf. Kaspar, 2024b. The source code and additional materials can be accessed on GitHub, cf. Kaspar, 2024a. The results of this chapter have been published (Öztürk et al., 2025).

5.1. Aggregation and Disaggregation Tool PyFlexAD

This section provides a brief introduction to the (dis-)aggregation tool PyFlexAD. First, the software structure is outlined, followed by a discussion of the software architecture.

5.1.1. Software Structure

The proposed software is structured in upper and lower-level classes and supports multiple types of energy storage systems, e.g., BESSs, EVs, TCLs, and PHESs. These models extend the more general *EnergyStorage* class, cf. Fig. 5.1. In order to calculate the general parameters of an energy storage system, hardware and usage parameters must be provided, cf. Table 2.1. Each subclass of *EnergyStorage* requires a specific set of parameters, e.g., to instantiate the class *ElectricVehicle* instances of *EVHardware* and *EVUsage* are required. For ease of use, the package already includes a sample of hardware parameters for EV models from manufacturers like Tesla, Nissan, and Renault, cf. Table 5.2, Tesla and GENERAC for BESS, cf. Table 5.1, and generic air conditioning and water heater for TCL, cf. Table 5.3.

The class *Aggregator* is a subclass of *VirtualEnergyStorage*. In addition to inheriting all of its properties, it introduces additional methods for (dis-)aggregation, as illustrated in Fig. 5.1. To perform aggregation, the aggregator requires instances of *VirtualEnergyStorage*.

Table 5.1.: BESS hardware parameters. Table reproduced (Öztürk et al., 2025).

	$x_{\min}$ (kW)	$x_{\max}$ (kW)	$S_{\min}$ (kWh)	$S_{\max}$ (kWh)	α
Tesla Powerwall 2	-5	5	0	13.5	1
Tesla Powerwall 3	-11.5	11.5	0	13.5	1
Tesla Powerwall +	-5.8	5.8	0	13.5	1
GENERAC PWRcell M3	−3.4	3.4	0	9	1
GENERAC PWRcell M4	−4.5	4.5	0	12	1
GENERAC PWRcell M5	−5.6	5.6	0	15	1
GENERAC PWRcell M6	−6.7	6.7	0	18	1

VirtualEnergyStorage objects can be generated from *EnergyStorage* instances by invoking the method *to_virtual*. When provided with compatible direction vectors, *to_virtual* calculates the corresponding corrected extreme actions, cf. Algorihtm 4, and returns an instance of *VirtualEnergyStorage*. The link to the device's physical counterpart remains only via a shared identification key.

The *VirtualEnergyStorage* objects can then be passed via a list to the *Aggregator* class. The class method *aggregate* then directly sums up the corresponding corrected extreme actions of a given list of *VirtualEnergyStorage* objects and returns an *Aggregator* instance. Note that the number of extreme directions involved in the virtualization process directly affects the number of calculated corrected extreme actions, and therefore determines the

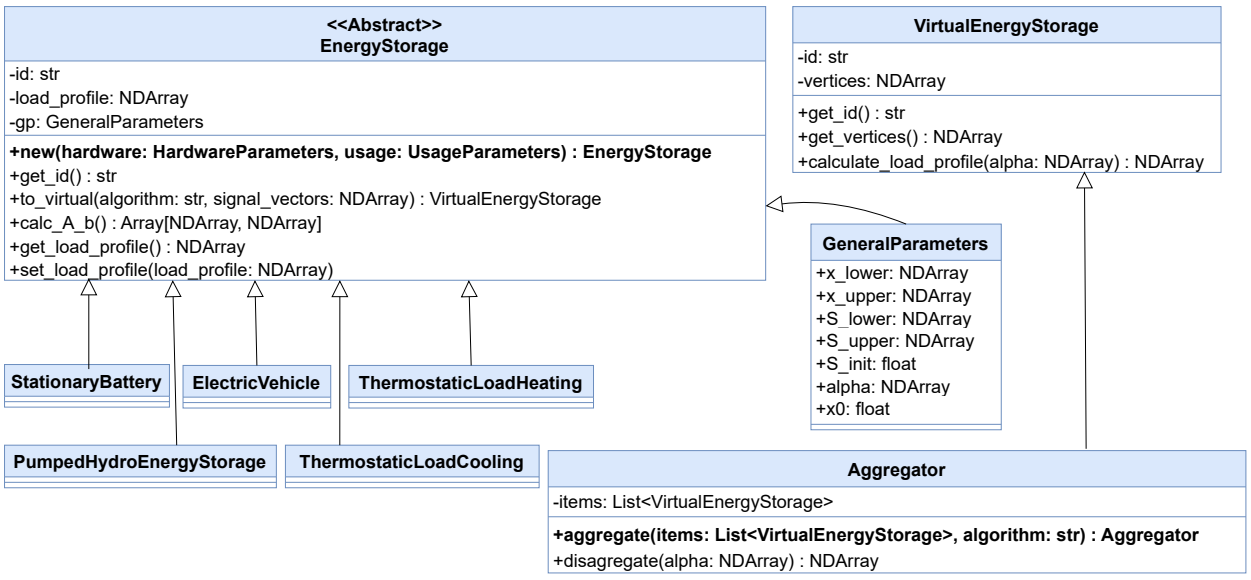

Figure 5.1.: Diagram of main classes in the PyFlexAD package using UML notation. Figure reproduced (Öztürk et al., 2025).

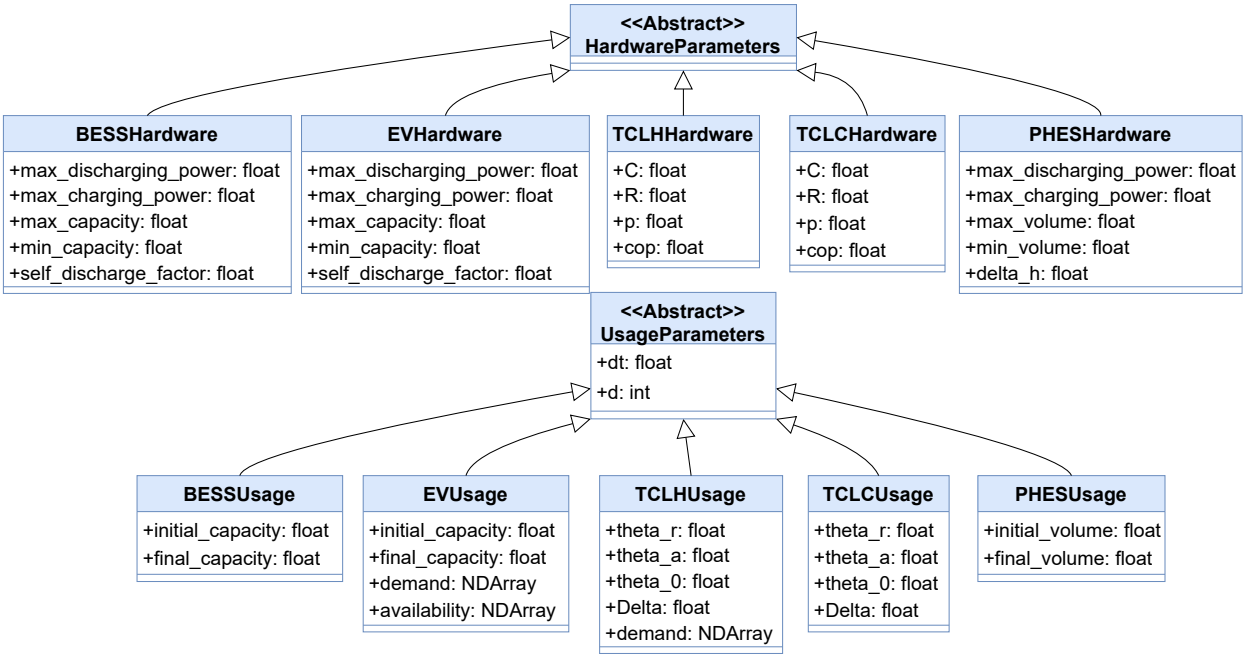

Figure 5.2.: Diagram of the parameter classes in the PyFlexAD package using UML notation. Figure reproduced (Öztürk et al., 2025).

complexity and accuracy of the approximation.

To apply the optimized power profile to the physical energy storage devices via the aggregated virtual storages, the aggregator performs the *disaggregate* method. This method signals each of its virtual energy storages to execute the *calculate_load_profile* method using the α-values provided by the optimizing entity. The disaggregated power profiles are then forwarded to the corresponding physical devices. For the mathematical details of disaggregation, see Section 4.4.

Alongside the classes depicted in Fig. 5.1 and 5.2, the package features two pre-built opti-

Table 5.2.: EV hardware parameters. Table reproduced (Öztürk et al., 2025).

	$x_{\min}$ (kW)	$x_{\max}$ (kW)	$S_{\min}$ (kWh)	$S_{\max}$ (kWh)	α
Nissan Leaf 6.6kW AC	−6.6	6.6	0	39	1
Tesla Model Y 11kW AC	−11	11	0	57.5	1
Tesla Model S P100D 16.5kW AC	−16.5	16.5	0	95	1
Renault Zoe ZE40 R110 22kW AC	−22	22	0	41	1
Renault Zoe ZE40 R110 40kW DC	−40	40	0	41	1
Renault Zoe ZE50 R135 22kW AC	−22	22	0	52	1
Renault Zoe ZE50 R135 41kW DC	−41	41	0	52	1

Table 5.3.: TCL hardware parameters. Table reproduced (Öztürk et al., 2025).

	C (kWh/K)	R (K/kW)	$p_{\max}$ (kW)	η
Generic air conditioner	2	2	5	2.5
Generic water heater	6	800	3	3

mization units designed for specific objectives: the *VertexBasedPowerController* and *VertexBasedCostController* classes. Users can create instances of these classes and call the *optimize* method, which solves problem (4.7). This method requires an *Aggregator* instance as input and produces the optimal aggregated power profile based on the set objective.

For comparison purposes, central control schemes with predefined objectives are implemented to directly optimize using the A-matrices and b-vectors, cf. (3.21). The A matrices and the b vectors required for the half-space representation can be computed by calling the *calc_A_b* method for *EnergyStorage* objects, which applies (2.18). The optimal power profiles can then be determined by employing the *optimize* method from instances of the *CentralizedPowerController* and *CentralizedCostController* classes.

Furthermore, the package includes an exact aggregation strategy as detailed in (Fukuda, 2021). This method can be accessed by invoking the *from_physical* method on an *Aggregator* object, which necessitates providing a list of physical devices and selecting the algorithm option *Algorithms.EXACT*. However, bear in mind that the exact aggregation only applies in low-dimensional spaces and is not applicable in higher dimensions.

Finally, the *VirtualEnergyStorage* also includes integrated methods for visualizing the entire (dis-)aggregation process. By calling the *plot_polytope_2d* and *plot_load_profile_2d* methods in combination, users can effectively visualize each step, from aggregation to disaggregation. The method *plot_polytope_2d* displays the feasible region of the respective *VirtualEnergyStorage*, and the method *plot_load_profile_2d* displays the operating point within the feasible region. Note that these methods are implemented only for two-dimensional spaces and cannot be used in higher dimensions.

5.1.2. Software Architecture

This section outlines the software architecture, which is depicted in Fig. 5.3. The figure illustrates the centralized and aggregation-based control mechanisms along with their respective communication channels.

The central control framework, depicted on the left, includes an optimizer that actively monitors the physical devices, performs calculations, and sets load profiles. Since the devices are directly controlled, information is shared openly, making it impossible to safeguard privacy. In addition, since the optimizer monitors all devices, communication requirements increase as the number of devices increases.

In contrast, the aggregation-based control introduces a virtual layer that facilitates the virtualization of devices. In this layer, the corrected extreme actions are determined and the physical devices are represented exclusively with their corrected extreme actions. This is achieved within the software by applying the *to_virtual* method on *EnergyStorage* objects, resulting in instances of *VirtualEnergyStorage*. A key advantage of this approach is that, since extreme actions lack any device-specific information, privacy is maintained. The aggregator, which is also a *VirtualEnergyStorage*, then sums the calculated corrected extreme actions. Additionally, given that each *Aggregator* instance is a *VirtualEnergyStorage* object, hierarchical aggregation of aggregators is feasible. This structure allows for initial technical aggregations to deliver flexibility services, which can then be followed by market-driven aggregation to combine multiple technical aggregators into a portfolio with enhanced flexibility resources, as depicted on the far right of Fig. 5.3. The aggregated corrected extreme actions are then sent to the optimizing instance. This instance calculates the weights, referred to as α, which are then forwarded by the aggregator to the individual

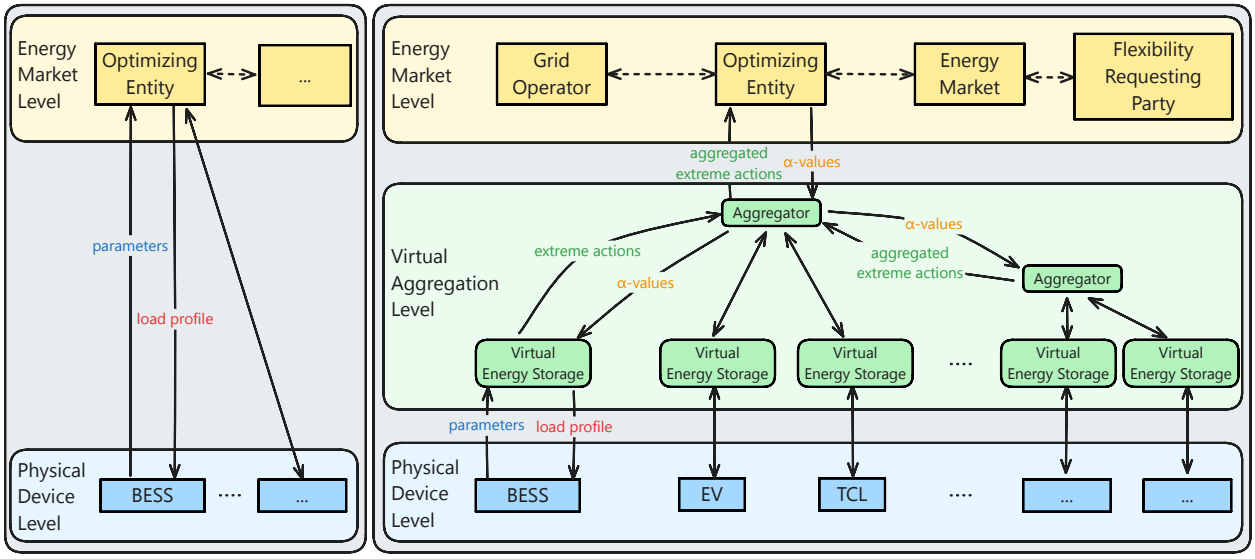

Figure 5.3.: Software Architecture: Left: Central control structure. Right: Proposed aggregation-based control structure, featuring a virtual aggregation level at the center and a hierarchical aggregation structure on the far right. Figure reproduced (Öztürk et al., 2025).

devices.

Note that, because the optimizing entity engages only with the aggregator, the communication overhead is significantly reduced, i.e., from n communications to one. In addition, it is crucial to emphasize that the aggregation process can only be executed when all devices calculate the corrected extreme actions using the same set of extreme directions $\mathcal{J}$, necessitating a uniform alignment of extreme directions across all devices.

5.2. Illustrative Code Examples

This section provides sample code for a range of potential applications and is intended to aid users in effectively leveraging the implemented software tool. We begin with a simple example that features a single type of flexibility along with one aggregator. Thereafter, a more comprehensive example that incorporates hierarchical aggregation is presented. Note that the examples provided here are merely a selection of the additional scripts available (Kaspar, 2024a), which are intended to support users and ease the introduction to the software. To create a more complex application, it is often sufficient to copy and combine the codes from the basic examples provided.

5.2.1. Simple Aggregation

In the following example, the controller's objective is to reduce the system's peak power consumption by utilizing the flexibility of energy resources. The Python code for this example can be found in Appendix A. Readers are encouraged to review the code for a more comprehensive understanding of the methods and classes discussed.

Our analysis involves two discrete time intervals, each lasting 15 minutes, and two BESS devices. First, the processes involved in aggregation-based control are outlined. Then the procedures for centralized control are discussed. Finally, the calculations are visualized along with an exact aggregation strategy.

The necessary steps for aggregation-based control are as follows. Flexible devices are generated in Lines 23 and 24 by instantiating *StationaryBattery* objects, using the predefined hardware parameters *tesla.power_wall_2* and *tesla.power_wall_3* together with selected usage parameters. The extreme directions are generated in Line 27 by instantiating a *SignalVectors* object and passing the dimension d. The flexible devices are then virtualized in

Lines 28 and 29 by calling the method *to_virtual*, which returns an instance of *VirtualEnergyStorage*. During this step, the corrected extreme actions in the respective devices are calculated, whereby the devices are represented exclusively by their corrected extreme actions. In Line 32, the *aggregate* class method is called to aggregate the virtual devices, returning an instance of an *Aggregator* object. The aggregator is then passed to an instance of *VertexBasedPowerController* in Line 41, which calculates the optimal power profiles by calling the *optimize* method.

The required procedures for centralized control are outlined in Lines 36 and 37. In Line 36, an instance of *CentralizedPowerController* is generated. Then, in Line 37, the optimal power profiles are calculated by calling the *optimize* method, which requires instances of *EnergyStorage*.

For illustration purposes, the exact aggregation is computed in Line 33. The class method *from_physical* is called, which requires instances of *EnergyStorage* and the algorithm *Al-*

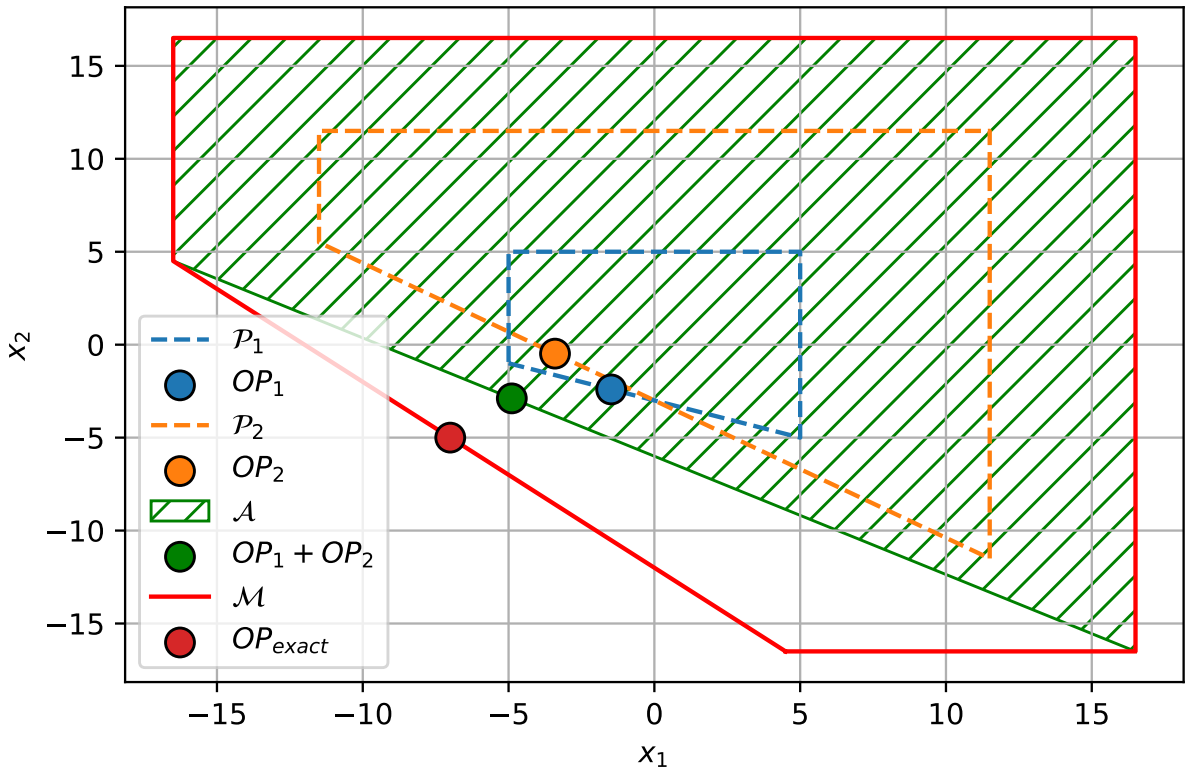

Figure 5.4.: An illustration featuring two polytopes, one in dashed blue and the other in dashed orange, along with their Minkowski sum in solid red, and the approximation in green. The operational points are indicated as dots within the corresponding flexibilities, each matching the respective colors.

gorithms.EXACT.

Figure 5.4 shows the output of *plot_polytope_2d* and *plot_load_profile_2d* in Lines 46 - 53. The two flexible devices are shown in dashed orange and dashed blue together with their Minkowski sum in solid red and the approximation in green. The operating points within the flexibilities are marked by dots in the respective colors.

In this example, the maximum peak power without any flexibility is 23.0 kW. When employing the approximate aggregation-based strategy, the peak power decreases to 18.11 kW. With the central control approach that utilizes the exact feasible region, the peak power further reduces to 16.0 kW.

5.2.2. Hierarchical Aggregation

The following example illustrates the application of the PyFlexAd tool in hierarchical aggregation settings. We analyze five discrete time intervals, each lasting 15 minutes. The objective of the controller is to reduce the overall peak of the system. First, the devices and aggregators are defined. Next, the procedures required for aggregation-based control are outlined. Following that, we discuss the processes involved in centralized control. The Python code for this example can be found in Appendix B. Readers are encouraged to review the code for a more comprehensive understanding.

First, we outline the procedures required for aggregation-based control. The first for loop (Line 24) iterates through all stationary batteries from predefined options, cf. Table 5.1. In a second for loop (Line 25), two devices are generated and stored in a list for a fixed type. This list is then forwarded to an aggregator along with the algorithm option and the signal vectors needed to compute the extreme actions (Line 27). Executing both loops results in a list of seven aggregators, each containing two devices. This list of aggregators is then forwarded to another aggregator (Line 29), representing the overall aggregation. The controller for aggregation-based control is created based on the system's power demand (Line 42), and the optimized aggregated power profile is computed (Line 43). Note that after this computation, each device holds the disaggregated power profiles.

Central control begins with the creation of the devices list (Line 26), which contains the devices generated for the aggregation process. Upon completion of the two for loops, the devices list holds two instances of seven predefined stationary battery options. Central control is then initiated using the system's power demand (Line 38), and the optimal aggregated power profile is computed (Line 39).

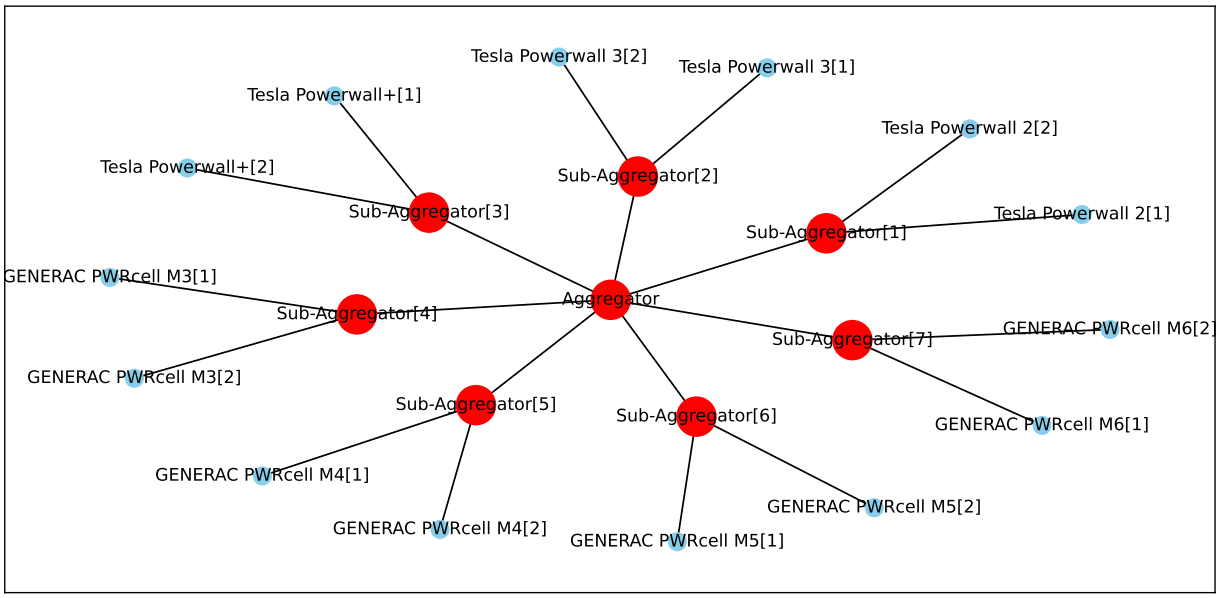

Figure 5.5.: Hierarchical aggregation setting used in the example computation, featuring seven aggregators, each connected to two devices.

In this example, the peak power when using aggregation-based control is 18.71 kW, whereas the peak power with the central controller is 15 kW. In comparison, the peak power without flexibility is 100 kW. Thus, a significant decrease in peak power can be achieved with both controllers. However, due to the operation within the exact feasible region, the central controller achieves slightly better results.

Finally, note that the tool provides a comprehensive visualization for hierarchical aggregation settings. The required code is illustrated in Lines 32 – 35. Figure 5.5 shows the output of this function, visualizing a network graph with seven aggregators, each comprising two stationary batteries.

5.3. Summary

This chapter provided an in-depth introduction to a comprehensive, open-source Python package designed for the (dis-)aggregation of convex storage models. The package was fully described, with an emphasis on its core functionalities and features. A detailed explanation of the underlying code structure was offered, including class diagrams that illustrate the package's architecture and its relationships between different components.

Furthermore, the communication channels between the package and the broader system architecture were outlined, ensuring that users understand how the package interfaces with external systems or processes.

To facilitate practical usage, the chapter presented several example calculations, accompanied by visualizations that demonstrate the capabilities of the package. These examples were supplemented by a comprehensive discussion of the code, offering clear and detailed explanations of each part of the process. The results of this chapter have been published (Öztürk et al., 2025).

6. Extensions towards Non-convexity and Uncertainty

Now, we extend the proposed approach to cover non-convexity and uncertainty. We begin by investigating the aggregation of non-convex sets, which more accurately reflect realistic storage models. We introduce a non-convex storage model, analyze its implications, and review the existing literature on non-convex aggregation strategies. Additionally, we propose an extended strategy for aggregating non-convex storage models.

Next, we explore aggregation under uncertain conditions, illustrating how the feasible region evolves with varying data. We review the literature on aggregation strategies in uncertain scenarios. Finally, we introduce an extension for aggregation in uncertain data environments. The extensions proposed in this chapter are currently being prepared for publication and have not yet been submitted or published.

6.1. Aggregation Strategies for Non-convex Sets

This section provides an overview of non-convex aggregation strategies. We begin with an example of a non-convex storage model and discuss its implications. Then, the existing research on non-convex aggregation strategies is outlined. Finally, an extended aggregation strategy is proposed to capture non-convex storage models.

6.1.1. Introduction

In Section 2.2, a convex storage model was presented that disregards (dis-)charging efficiencies. A more realistic storage model that includes (dis-)charging efficiencies can be

described as follows, cf. Rheinberger et al., 2021

$$0 \leq x_t^C \leq \overline{x}_t, \forall t = 1, \dots d, \tag{6.1a}$$
$$0 \leq x_t^D \leq \underline{x}_t, \forall t = 1, \dots d, \tag{6.1b}$$
$$x_t^C x_t^D = 0, \forall t = 1, \dots d, \tag{6.1c}$$
$$S_t = \alpha S_{t-1} + \eta_C x_t^C \Delta t - \frac{1}{\eta_D} x_t^D \Delta t, \forall t = 1, \dots d, \tag{6.1d}$$
$$\underline{S}_t \leq S_t \leq \overline{S}_t, \forall t = 1, \dots d, \tag{6.1e}$$
$$S_0 = S_{\text{init}}, \tag{6.1f}$$

where $x^C, x^D \in \mathbb{R}^d$ denote the (dis-)charging power and $\eta_C, \eta_D \in (0, 1]$ denote the (dis-)charging efficiencies. All other values are specified in Section 2.2 and remain unchanged. Note, that if (dis-)charging efficiencies are neglected in the above model, i.e., $\eta_C = \eta_D = 1$, and $x = x^C - x^D$, then (6.1) simplifies to the convex storage model (2.17).

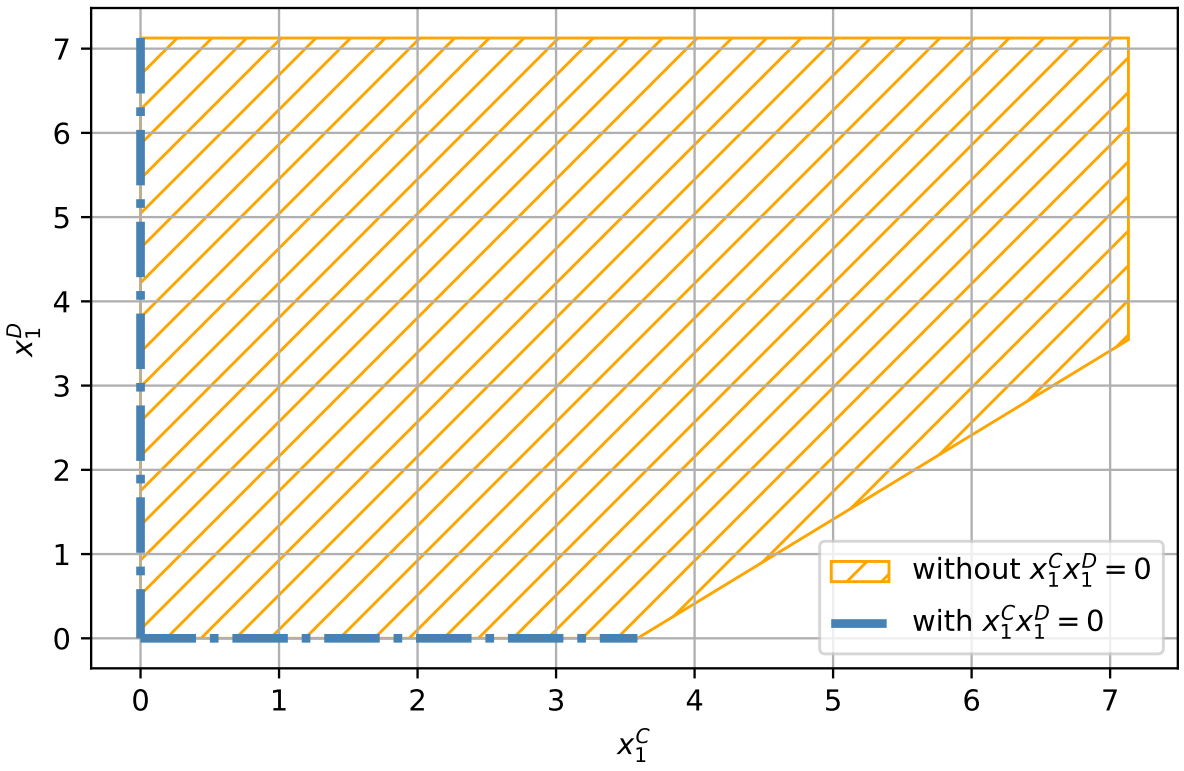

Figure 6.1.: Storage model with (dis-)charging efficiencies. The model without complementary constraint is shown in orange and the model with complementary constraint is shown in dotted blue.

The complementary constraints (6.1c) prevent simultaneous (dis-)charging, which results in a non-convex feasible region. Figure 6.1 illustrates this concept, where the storage model without complementary constraints is shown in orange and the model with complementary constraints in blue. The model with complementary constraints is non-convex, as lines between points on the x_1^D and x_1^C axes are not included in the blue set. However, a polytope, and thus a convex set, is obtained if the complementary constraints are omitted while taking the (dis-)charging efficiencies into account.

Let $a_d \coloneqq (\alpha, \alpha^2, \dots, \alpha^d)^\top$, $I \in \mathbb{R}^{d\times d}$ be the identity matrix, $\mathbf{0}_{d\times d} \in \mathbb{R}^{d\times d}$ the matrix of zeros, and $\Gamma \in \mathbb{R}^{d\times d}$ a Toeplitz matrix with first column and row defined by $(1, \alpha, \dots, \alpha^{d-1})^\top$ and $(1, 0, \dots, 0)$, respectively. Then, the feasible region for the model without complementary constraints and with (dis-)charging efficiencies is given by

$$\mathcal{B}\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}, \eta_C, \eta_D\right) \coloneqq \left\{ \begin{pmatrix} x^C \\ x^D \end{pmatrix} \in \mathbb{R}^{2d} : A\left(\alpha, \eta_C, \eta_D\right) \begin{pmatrix} x^C \\ x^D \end{pmatrix} \leq b\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}\right) \right\}, \tag{6.2}$$

where $A \in \mathbb{R}^{6d\times 2d}$ and $b \in \mathbb{R}^{6d}$ are defined by

$$A\left(\alpha, \eta_C, \eta_D\right) = \begin{pmatrix} I & \mathbf{0}_{d\times d} \\ -I & \mathbf{0}_{d\times d} \\ \mathbf{0}_{d\times d} & -I \\ \mathbf{0}_{d\times d} & I \\ \eta_C\Gamma & -\frac{1}{\eta_D}\Gamma \\ -\eta_C\Gamma & \frac{1}{\eta_D}\Gamma \end{pmatrix},$$

$$b\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}\right) = \left(\overline{x}^\top, \mathbf{0}_d^\top, \mathbf{0}_d^\top, \underline{x}^\top, \frac{1}{\Delta t}(\overline{S} - S_{\text{init}} a_d)^\top, \frac{1}{\Delta t}(-\underline{S} + S_{\text{init}} a_d)^\top\right)^\top.$$

Note that this model is a superset of the non-convex model. In addition, two variables, namely x_t^D and x_t^C, must be considered for each time period t, resulting in a polytope in $2d$-dimensional space. With this preliminary work, the non-convex model is given by

$$\mathcal{X}\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}, \eta_C, \eta_D\right) \coloneqq$$
$$\left\{ \begin{pmatrix} x^C \\ x^D \end{pmatrix} \in \mathbb{R}^{2d} : A\left(\alpha, \eta_C, \eta_D\right) \begin{pmatrix} x^C \\ x^D \end{pmatrix} \leq b\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}\right), x_t^C x_t^D = 0, \forall t = 1, \dots, d \right\}, \tag{6.3}$$

i.e, the polytop constraints together with the complementary constraints. For models that only allow for charging or discharging, the following d-dimensional polytopes are obtained

$$\mathcal{B}^C\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}, \eta_C, \eta_D\right) \coloneqq \left\{ x^C \in \mathbb{R}^d : A\left(\alpha, \eta_C, \eta_D\right) \begin{pmatrix} x^C \\ \mathbf{0}_d \end{pmatrix} \leq b\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}\right) \right\}, \tag{6.4a}$$

$$\mathcal{B}^D\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}, \eta_C, \eta_D\right) \coloneqq \left\{ x^D \in \mathbb{R}^d : A\left(\alpha, \eta_C, \eta_D\right) \begin{pmatrix} \mathbf{0}_d \\ x^D \end{pmatrix} \leq b\left(\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha, S_{\text{init}}\right) \right\}. \tag{6.4b}$$

Note that these models permit either charging or discharging, which means that the complementary condition is inherently satisfied and does not require separate consideration. In addition, these models are subsets of both the non-convex model (6.3) and the model with (dis-)charging efficiencies without complementary constraints (6.2). The proposed method can be applied to the models in (6.4) by merely adjusting the algorithms to accommodate the new system dynamics and constraints. In the following, we provide a brief literature review before proposing an extended algorithm for the aggregation of non-convex storage models.

6.1.2. Related Literature

As outlined above, the aggregation of non-convex storage models poses significant challenges due to their non-convex nature and the time coupling inherent in storage devices. As a result, the number of studies available on this topic is limited.

Taheri et al., 2022 propose a data-driven approach to bypass the exact calculations and estimate the aggregated flexibility of possibly non-convex sets. A convex quadratic classifier is trained to approximate the feasible region by an ellipsoid, which is then further approximated by a polytope. The training data are generated by solving disaggregation problems, i.e., finding a feasible power profile that is closest to a given power profile. Uncertainties are taken into account by varying parameters and solving the disaggregation problem multiple times. The authors propose to generate a balanced ratio of feasible and infeasible power profiles for the training set. Consequently, this method cannot be classified as an inner or outer approximation strategy. Non-convex TCL, BESS, and EV models are then aggregated and further tested in numerical examples.

6.1.3. Extension

The current form of the proposed aggregation strategy does not apply to the non-convex model (6.3). Nevertheless, it is possible to develop the proposed method even further to capture these models as well. In the following, we first propose an aggregation strategy for the models with (dis-)charging efficiency and without complementary constraints. Then, we extend this strategy to capture non-convex storage models.

We start by formulating an approximation strategy for the Minkowski sum of storage models $\mathcal{B}\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{\text{init, i}}, \eta_{C,i}, \eta_{D,i}\right), i = 1, \ldots, n$. The following proposition provides a critical relationship.

Proposition 4 (Convex aggregation)**.** *Let extreme actions $y_i^{j,C}, y_i^{j,D} \in \mathbb{R}^d$ calculated for the storage models $\mathcal{B}^C\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{\text{init, i}}, \eta_{C,i}, \eta_{D,i}\right), \mathcal{B}^D\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{\text{init, i}}, \eta_{C,i}, \eta_{D,i}\right)$, and the set of extreme directions $\mathcal{J} \subseteq \{1, -1\}^d$ be given. Further, let the sum of extreme actions be $v^{j,C} = \sum_{i=1}^n y_i^{j,C}, v^{j,D} = \sum_{i=1}^n y_i^{j,D}$. Then, it holds that*

$$\begin{aligned} &\mathit{Conv}\left(\begin{pmatrix} v^{j,C} \\ \mathbf{0}_d \end{pmatrix}, \begin{pmatrix} \mathbf{0}_d \\ v^{j,D} \end{pmatrix} : j \in \mathcal{J}\right) \subseteq \\ &\left\{ x \in \mathbb{R}^{2d} : x = \sum_{i=1}^n x_i, x_i \in \mathcal{B}\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{\text{init,i}}, \eta_{C,i}, \eta_{D,i}\right) \right\}. \end{aligned} \tag{6.5}$$

Proof. By design, it holds that $y_i^{j,C} \in \mathcal{B}^C\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{\text{init, i}}, \eta_{C,i}, \eta_{D,i}\right)$. Therefore, it follows that $(y_i^{j,C}, \mathbf{0}_d)^\top \in \mathcal{B}\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{\text{init, i}}, \eta_{C,i}, \eta_{D,i}\right)$, and by a similar line of argument that $(\mathbf{0}_d, y_i^{j,D})^\top \in \mathcal{B}\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{\text{init, i}}, \eta_{C,i}, \eta_{D,i}\right)$. Since $(y_i^{j,C}, \mathbf{0}_d)^\top, (\mathbf{0}_d, y_i^{j,D})^\top$ are elements in the storage model $\mathcal{B}\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{\text{init, i}}, \eta_{C,i}, \eta_{D,i}\right)$, it follows that $(v^{j,C}, \mathbf{0}_d)^\top, (\mathbf{0}_d, v^{j,D})^\top$ are elements in the Minkowski sum of these models. Since $\mathcal{B}\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{\text{init,}i}, \eta_{C,i}, \eta_{D,i}\right)$ is a polytope, and the Minkowski sum of polytopes results in a polytope, cf. Ziegler, 1995, it follows that any convex combination of elements $(v^{j,C}, \mathbf{0}_d)^\top, (\mathbf{0}_d, v^{j,D})^\top$ is also an element of the Minkowski sum. As a result, for any collection of extreme directions $\mathcal{J} \subseteq \{-1, 1\}^d$, the convex hull generated by the vectors $(v^{j,C}, \mathbf{0}_d)^\top, (\mathbf{0}_d, v^{j,D})^\top$ is contained within the Minkowski sum. □

The above result provides a convex inner approximation to the Minkowski sum of storage models with (dis-)charging efficiencies and without complementary constraints. As indicated in (6.3), the non-convex model consists of two parts: the polytope constraints and

the complementary constraints. Since an approximation is obtained for the polytope constraints, we proceed below with the complementary constraints, which leads directly to an inner approximation to the Minkowski sum of non-convex storage models.

Theorem 3 (Non-convex aggregation)**.** *Let models* $\mathcal{X}\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{init,i}, \eta_{C,i}, \eta_{D,i}\right), i = 1, \ldots, n$, *a set of extreme directions* $\mathcal{J} \subseteq \{1, -1\}^d$, *and extreme actions* $y_i^{j,C}, y_i^{j,D} \in \mathbb{R}^d$ *calculated for the models* $\mathcal{B}^C\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{init,\,i}, \eta_{C,i}, \eta_{D,i}\right)$, $\mathcal{B}^D\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{init,\,i}, \eta_{C,i}, \eta_{D,i}\right)$ *be given. Further, let the sum of extreme actions be* $v^{j,C} = \sum_{i=1}^n y_i^{j,C}$, $v^{j,D} = \sum_{i=1}^n y_i^{j,D}$. *Then, it holds that*

$$\left\{\begin{pmatrix} x^C \\ x^D \end{pmatrix} \in Conv\left(\left\{\begin{pmatrix} v^{j,C} \\ \mathbf{0}_d \end{pmatrix}, \begin{pmatrix} \mathbf{0}_d \\ v^{j,D} \end{pmatrix} : j \in \mathcal{J}\right\}\right) : x_t^C x_t^D = 0, \forall t = 1, \ldots, d\right\} \subseteq$$
$$\left\{x \in \mathbb{R}^{2d} : x = \sum_{i=1}^n x_i,\ x_i \in \mathcal{X}\left(\underline{x}_i, \overline{x}_i, \underline{S}_i, \overline{S}_i, \alpha_i, S_{init,i}, \eta_{C,i}, \eta_{D,i}\right)\right\}.$$

Proof. Suppose that we apply the complementary condition directly to the aggregated variables, i.e.,

$$x_t^C x_t^D = \left(\sum_{i=1}^n x_{i,t}^C\right)\left(\sum_{k=1}^n x_{k,t}^D\right) = \sum_{i=1}^n \sum_{k=1}^n x_{i,t}^C x_{k,t}^D = 0.$$

Since both $x_{i,t}^C$ and $x_{k,t}^D$ are positive values, this summation can only be zero, if and only if $x_{i,t}^C x_{k,t}^D = 0, \forall i, k = 1, \ldots, n$. Therefore, applying the complementary constraints to the aggregated variables inherently involves applying those same constraints to the individual variables. The non-convex model is formulated by incorporating the polytope constraints along with the complementary constraints, as referenced in (6.3). Proposition 4 offers already an approximation to the polytope constraints. Accordingly, the set established in Proposition 4, when augmented by the complementary constraints applied to the aggregated variables, forms a subset of the Minkowski sum of non-convex storage models. □

The above relation provides an inner approximation to the Minkowski sum of non-convex storage models. Note that optimal solutions within non-convex storage models are typically obtained by reformulating these models into a mixed-integer linear programming (MILP) format (Pozo, 2023). This process involves the introduction of additional $2d$ binary variables per device, which significantly increases the computational complexity. Our formulation, on the other hand, reduces the complexity by reducing the number of variables from $2dn$ to $2d$, which is independent of the number of devices.

However, additional research is necessary to develop the algorithms for calculating the extreme actions and to assess the accuracy and computational complexity of the proposed approximation strategy.

6.2. Aggregation Strategies under Uncertain Data

This section examines the aggregation of flexible devices in the context of uncertain data. Initially, we assess the impact of uncertainty using examples of convex storage models. Following this, we present a review of the current literature on aggregation strategies in uncertain data scenarios. Lastly, we explore potential solutions for applying the proposed method in uncertain data environments.

6.2.1. Introduction

The convex storage model detailed in (2.17) includes various hardware data relevant to the power and energy characteristics. In particular, the model contains power-related parameters, denoted as $\overline{x}$ and $\underline{x}$, and energy-related parameters, denoted as $\overline{S}$, $\underline{S}$, and α. When focusing solely on the power constraints, the result is a set of cuboids that define the feasible regions. Consequently, fluctuations in the power data cause variations in these cuboids. Conversely, when only the energy-related data are considered, hyperplanes are formed that intersect the cuboids. As a result, changes in the energy-related data lead to changes in these hyperplanes.

This concept is depicted in Fig. 6.2. The left side displays the feasible region of a typical storage device in black, with storage devices that exhibit slightly different power data shown in light blue. In the center, the same storage device is illustrated alongside others with slightly altered energy data. The right side of the figure showcases storage devices with variations in both energy and power data, alongside a robust storage option, which will be relevant later in this section, highlighted in orange. In the following, we briefly review the literature on aggregation strategies in uncertain data environments, before proposing an extended algorithm.

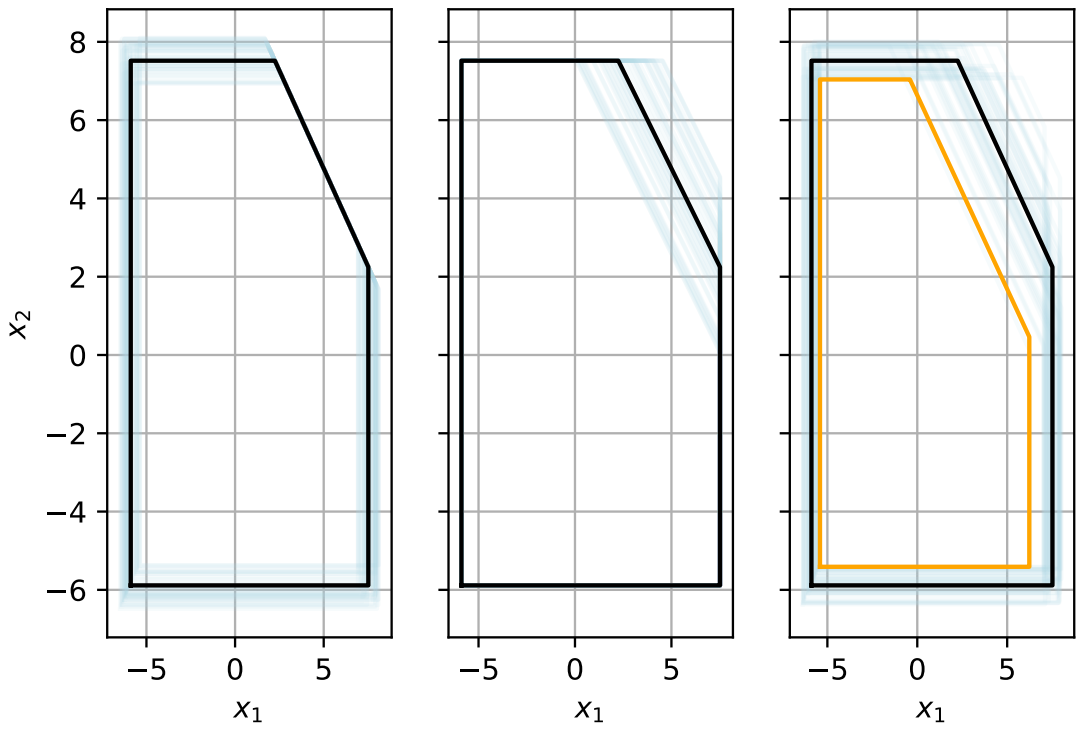

Figure 6.2.: The black line represents a convex storage model. The blue lines depict storages exhibiting variations in power (left), energy (center), and both power and energy (right). The orange shape illustrates the resulting polytope obtained via robust pre-processing.

6.2.2. Related Literature

In this section, we briefly review aggregation strategies in the presence of uncertain data.

Jian et al., 2023; Zhang et al., 2024 introduce an inner approximation method for EVs under uncertain conditions such as arrival and departure times, initial energy levels, and charging requirements. The authors first group EVs with similar arrival and departure times into clusters, which can be collectively accessed during their designated time intervals. A prototype polytope is created by averaging the data, and homothets of this prototype are fit into each cluster. The authors then efficiently reformulate the fitting problem and incorporate chance constraints to address energy-related uncertainties. Zhang et al., 2024 propose further projection techniques to standardize the cluster polytopes to the same dimension. In addition, both studies propose disaggregation methods that do not require optimization.

Barot and Taylor, 2016 present an outer approximation method for problems involving

linear, second-order cone, and semidefinite constraints. They account for uncertain data, such as the charging efficiency of EVs and their charging requirements, by initially framing these uncertainties as chance constraints and subsequently reformulating them into second-order cone constraints. The EVs are modeled with consideration of charging efficiency, but without incorporating V2G technology. The proposed strategy is then tested in a numerical example involving 100 EVs over 5 time periods.

6.2.3. Extension

Our proposal for a potential extension is centered on the robust convex storage model illustrated in Fig. 6.2. This storage is designed based on the least favorable data, such as the minimum values of $\overline{x}_t$, the maximum values of $\underline{x}_t$, the minimum values of α, etc.

Algorithm 9 details a procedure for robust preprocessing. In Line 1, the least favorable (i.e., minimum) self-discharge factor is selected. In Lines 3 – 6, within a for loop, the least favorable values for $\underline{x}_t, \underline{S}_t$ (i.e., the maximum of the lower power and energy limits) and the least favorable values for $\overline{x}_t, \overline{S}_t$ (i.e., the minimum of the upper power and energy limits) are selected. The resulting polytope represents the most constricted feasible region, taking into account the uncertainty in the data.

By applying this preprocessing to each polytope, we can determine the smallest feasible regions possible. The robust convex storages can then be aggregated using the proposed aggregation strategy, resulting in aggregated power profiles that remain feasible across all potential realizations of the uncertain data. However, note that uncertainties regarding the availability of EVs can reduce the dimensionality of the polytopes, which can lead to a significant reduction in the feasible regions.

Algorithm 9 (RobustPreprocessing)

Require: $\{\underline{x}_i\}_{i=1,\dots,k_1}, \{\overline{x}_i\}_{i=1,\dots,k_2}, \{\underline{S}_i\}_{i=1,\dots,k_3}, \{\overline{S}_i\}_{i=1,\dots,k_4}, \{\alpha_i\}_{i=1,\dots,k_5}$
Ensure: $\underline{x}, \overline{x}, \underline{S}, \overline{S}, \alpha$
1: $\alpha = \min\{\alpha_i\}_{i=1,\dots,k_5}$
2: **for** $t = 1$ **to** d **do**
3: $\quad \underline{x}_t = \max\{\underline{x}_{i,t}\}_{i=1,\dots,k_1}$
4: $\quad \overline{x}_t = \min\{\overline{x}_{i,t}\}_{i=1,\dots,k_2}$
5: $\quad \underline{S}_t = \max\{\underline{S}_{i,t}\}_{i=1,\dots,k_3}$
6: $\quad \overline{S}_t = \min\{\overline{S}_{i,t}\}_{i=1,\dots,k_4}$
7: **end for**

Algorithm 9 can be applied to robustly select the hardware parameters. For the remaining usage parameter (S_{init}), the maximum and minimum values in $\{S_{\text{init},i}\}_{i=1,\dots,k_6}$ must be computed. The maximum initial energy is then used whenever the minus sign appears with S_{init} in (2.18), and the minimum is used when the plus sign appears with S_{init} in (2.18), ensuring a robust selection.

6.3. Summary

This chapter provided further extensions to the proposed aggregation strategy. It discussed the concept of non-convex sets using an example of a storage model that includes (dis-)charging efficiencies and reviewed the relevant literature on the aggregation of non-convex storage models. An extended strategy for aggregating non-convex storage models was then proposed.

The chapter also addressed aggregation in uncertain data environments, presenting an introductory example and outlining the pertinent literature. Additionally, an extended strategy for the aggregation in uncertain data environments was proposed. The proposed extensions are currently in development and have not yet been submitted or published in any journal.

7. Conclusions and Perspectives

This chapter concludes the thesis. First, the concept of grid-aware aggregation is introduced as a potential future research direction. Next, a comprehensive derivation of grid constraints is provided, followed by a review of relevant literature on grid-aware aggregation strategies. Finally, concluding remarks are drawn.

7.1. Grid-Aware Aggregation Strategies

In this section, aggregation is examined in greater detail in the context of grid constraints. First, an introduction to grid constraints, based on the work of Frank and Rebennack, 2016, is provided. Then, an overview of aggregation strategies that consider grid constraints is presented.

7.1.1. Introduction

Electric power systems can be represented as a network of buses that are connected by branches, such as transmission lines, cables, breakers, transformers, and other equipment

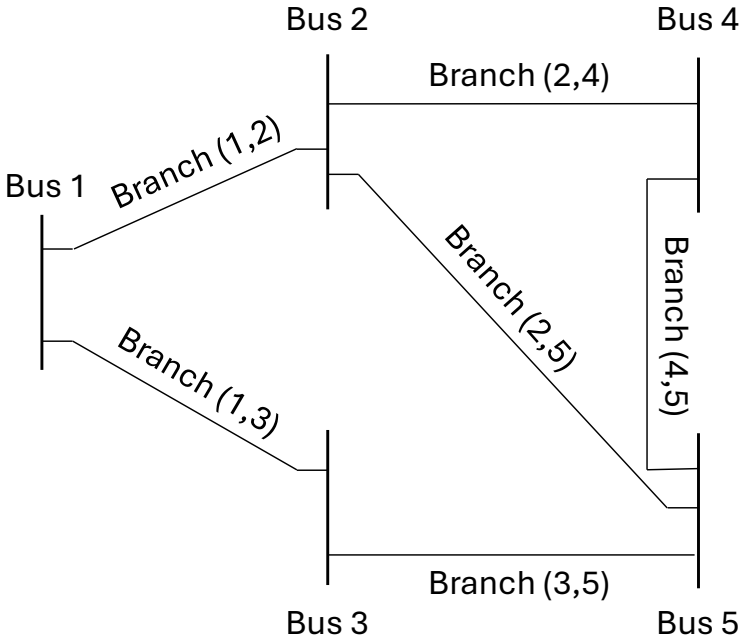

Figure 7.1.: Example of a power network with five buses connected via branches.

used in power systems, cf. Fig. 7.1. The primary function of this system is to transmit electrical energy from generation units to loads. Buses are identified by nodes with index $k \in \mathcal{N}$, while branches are identified as lines between nodes $(k, l) \in \mathcal{L}$, where $k, l \in \mathcal{N}$. The connectivity of a network can then be fully characterized by an undirected graph $(\mathcal{N}, \mathcal{L})$.

Each system bus k has an associated voltage $\tilde{V}_k$, which, when connected, generates a current in each branch proportional to the branch's admittance. Given the system admittance matrix $\tilde{Y} \in \mathbb{C}^{|\mathcal{N}|\times|\mathcal{N}|}$, the bus voltages $\tilde{V} \in \mathbb{C}^{|\mathcal{N}|}$ in each bus, the injected current $\tilde{I} \in \mathbb{C}^{|\mathcal{N}|}$, and the injected power $\tilde{S} \in \mathbb{C}^{|\mathcal{N}|}$ can be described as

$$\begin{aligned}\tilde{I} &= \tilde{Y}\tilde{V},\\ \tilde{S} &= \tilde{V} \odot \tilde{I}^{\star} = \tilde{V} \odot (\tilde{Y}\tilde{V})^{\star},\end{aligned}$$

where $\tilde{S}_k = P_k + iQ_k$, $\tilde{I}^{\star}$ is the element-wise complex conjugate of $\tilde{I}$, and $\odot$ is the element-wise multiplication operator, also called the Hadamard product. Note that we use the notation i to represent the imaginary unit, rather than the j notation that is commonly found in electrical engineering.

Using rectangular coordinates for the entries in the admittance matrix, i.e., $\tilde{Y}_{k,l} = G_{k,l} + iB_{k,l}$, and polar coordinates for the voltages, i.e., $\tilde{V}_k = \|\tilde{V}_k\|(\cos(\delta_k) + i\sin(\delta_k))$, together with the trigonometric relations one gets

$$\begin{aligned}\tilde{S}_k &= P_k + iQ_k = \tilde{V}_k \sum_{l\in\mathcal{N}} (\tilde{Y}_{k,l}\tilde{V}_l)^{\star}, \forall k \in \mathcal{N},\\ P_k &= \|\tilde{V}_k\| \sum_{l\in\mathcal{N}} \|\tilde{V}_l\|(G_{k,l}\cos(\delta_k - \delta_l) + B_{k,l}\sin(\delta_k - \delta_l)), \forall k \in \mathcal{N},\\ Q_k &= \|\tilde{V}_k\| \sum_{l\in\mathcal{N}} \|\tilde{V}_l\|(G_{k,l}\sin(\delta_k - \delta_l) - B_{k,l}\cos(\delta_k - \delta_l)), \forall k \in \mathcal{N}.\end{aligned}$$

The equations above are called the power flow equations that are essential for establishing the active and reactive power balances at each node. Flexibility options, such as BESSs, can be incorporated into these equations by adding constraints and defining the active power P_k and reactive power Q_k for the relevant case. Restrictions, such as branch voltage limits, can be included by introducing additional constraints.

The power flow equations are non-linear and lead to non-convex feasible regions. As a result, non-convex aggregation strategies are required to assess the aggregated flexibility resulting from the flexibilities at each node. However, the equations also do not include temporal coupling, which simplifies the calculations, especially in scenarios where no stor-

age devices are present. Note that alternative formulations and liner approximations to the power flow equations exist; detailed introductions can be found, e.g., Faulwasser et al., 2018; Frank and Rebennack, 2016. In the following, we present aggregation and control strategies in the literature that take grid constraints into account.

7.1.2. Related Literature

The non-linear characteristics of the power flow equations in conjunction with temporal coupling in storage applications pose significant challenges. As a result, strategies that effectively address both aspects are rare in current literature. Typically, the literature uses a linear approximation of the power flow equations to simplify the integration of storage systems, avoiding the complexities of non-linearities, as demonstrated by Jian et al., 2023; Wen et al., 2024.

However, some strategies successfully address the non-linear power flow component. Engelmann et al., 2025 introduce a method that integrates the power flow equations by employing an approximate dynamic programming approach. Given a problem in tree structure, the problem is divided into subproblems and solved via a dynamic programming approach. The feasible regions of the subproblems are approximated, e.g., by an ellipsoidal inner approximation. Numerical experiments show that this method can solve both AC and DC optimal power flow problems with feasibility guarantees.

Another method, that could potentially complement our proposed aggregation strategy, is detailed by Silva et al., 2018. In this study, the authors present a technique for estimating the aggregated flexibility at the TSO-DSO (transmission system operator - distribution system operator) level based on the flexibilities at each individual node. The flexibility area consists of reactive and active power, i.e., there are two components in each time period. Agents like aggregators could provide flexibility at each node. However, this flexibility is associated with costs that the user is willing to pay.

Silva et al., 2018 minimize the objective $\alpha P_{\text{DSO} \rightarrow \text{TSO}} + \beta Q_{\text{DSO} \rightarrow \text{TSO}}$, i.e., a linear objective function, together with non-linear power balance equations at each node. The α and β values are varied, e.g., $\alpha = \pm 1$ and $\beta = 0$, $\alpha = 0$ and $\beta = \pm 1$, as well as $\alpha = \pm 1$ and $\beta = \pm 1$, to compute eight distinct points that are located on the perimeter of the P-Q feasibility region. This procedure is repeated, if necessary, when the points are close to each other by changing the α and β values until a stooping criterion is reached. In addition, the maximum cost that the user is willing to pay is included as a constraint. Adjusting

this maximum cost results in varying levels of flexibility regions. Higher maximum costs result in increased flexibility regions at the TSO-DSO level. In order to solve the resulting optimization problems, a primal-dual interior point method is employed.

Note that non-convex feasibility regions usually lead to high computation times, and in the proposed method this is accompanied by solving the optimization problem at least eight times with different α and β values. Nevertheless, this strategy could complement our proposed method to account for the gird constraints. The feasible regions aggregated at each node (computed by our method) can serve as input to this technique. The sequential execution of both strategies could effectively consolidate storage-like loads while considering grid constraints.

7.2. Concluding Remarks

The increasing integration of renewable energy sources, coupled with rising demand, presents significant challenges for modern power grids. Additionally, the market structure is shifting from centralized monopolies and generation to a model dominated by distributed energy resources and competitive markets. Simultaneously, flexible devices are being integrated into the grid on a large scale, offering promising potential for the provision of ancillary services. While a single device may offer limited flexibility, when combined with others, these devices can function as a virtual power plant, contributing to grid stability. However, harnessing the aggregated flexibility of multiple devices requires the computation of the set addition, which is computationally intractable. Several authors have proposed approximation strategies to address this challenge. Nevertheless, state-of-the-art approximation methods are often constrained by computational limitations, may exclude nominal power profiles, and exhibit objective-dependent performance.

This thesis explored the aggregation-based control of multiple flexible devices in power grids. Initially, the main challenges in aggregation-based control, such as accurately quantifying the aggregated flexibility, were identified, and the need for approximate aggregation strategies was highlighted. Subsequently, typical flexible devices in power grids were examined and represented within a convex storage model.

Next, a comprehensive open-source benchmark for evaluating aggregation strategies was developed. State-of-the-art algorithms were implemented and thoroughly analyzed. The limitations of current algorithms were identified, including the objective-dependent performance, the exclusion of nominal power profiles, and the high computational complexity.

A novel aggregation strategy was then proposed to harness the aggregated flexibility of polytopes that fulfill two specific assumptions. The theoretical findings were outlined and discussed in detail. The proposed aggregation strategy was further extended to BESS models. This proposed strategy was tested within the framework of the developed benchmark and shown to outperform ten state-of-the-art inner approximations in terms of computational complexity and accuracy. Moreover, the proposed strategy was shown to overcome the identified shortcomings of current strategies.

Subsequently, the proposed strategy was extended to convex storage models. The extended strategy was then tested in numerical examples with one aggregator and one flexibility type, as well as with multiple aggregators and multiple flexibility types. It was shown that the proposed method can be used to reduce the peak demand in residential areas. Furthermore, an efficient disaggregation strategy for convex storage models was proposed that does not require any optimization.

The proposed strategy was then developed in a comprehensive open-source Python package. The package was explained in detail, including the software structure and architecture. Practical code examples were provided and discussed in detail to facilitate the introduction to the software tool.

Finally, related topics such as the aggregation of non-convex storage devices, and aggregation strategies under uncertain data were addressed. Potential extensions to the proposed method for covering these scenarios were outlined as future research topics.

In summary, this thesis offered an in-depth analysis of aggregation-based control strategies and presented novel algorithms for (dis-)aggregation, designed to assist researchers and industry professionals in making informed decisions in settings with multiple flexible devices.

Bibliography

Andújar Márquez, J. M., Segura Manzano, F., & Rey Luengo, J. (2023). *Energy Storage Systems: Fundamentals, Classification and a Technical Comparative*. Springer Nature Switzerland. doi: 10.1007/978-3-031-38420-2.

Appino, R. R., Hagenmeyer, V., & Faulwasser, T. (2021). Towards Optimality Preserving Aggregation for Scheduling Distributed Energy Resources. *IEEE Transactions on Control of Network Systems*, *8*(3), 1477–1488. doi: 10.1109/TCNS.2021.3070664.

Avis, D., Bremner, D., & Deza, A. (Eds.). (2009, March 19). *Polyhedral Computation* (Vol. 48). American Mathematical Society. doi: 10.1090/crmp/048.

Barot, S. (2017, June). *Aggregate load modeling for Demand Response via the Minkowski sum (Doctoral thesis, University of Toronto)*.

Barot, S., & Taylor, J. A. (2017). A concise, approximate representation of a collection of loads described by polytopes. *International Journal of Electrical Power & Energy Systems*, *84*, 55–63. doi: 10.1016/j.ijepes.2016.05.001.

Barot, S., & Taylor, J. A. (2016). An outer approximation of the Minkowski sum of convex conic sets with application to demand response. *2016 IEEE 55th Conference on Decision and Control (CDC)*, 4233–4238. doi: 10.1109/CDC.2016.7798912.

Boyd, S. (2010). Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends® in Machine Learning*, *3*(1), 1–122. doi: 10.1561/2200000016.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Braun, P., Faulwasser, T., Grüne, L., Kellett, C. M., Weller, S. R., & Worthmann, K. (2018). Hierarchical distributed ADMM for predictive control with applications in power networks. *IFAC Journal of Systems and Control*, *3*, 10–22. doi: 10.1016/j.ifacsc.2018.01.001.

Bundesnetzagentur. (2023, November 29). *Monitoringbericht 2023 (accessed on 2024-11-08)*. https://data.bundesnetzagentur.de/Bundesnetzagentur/SharedDocs/Mediathek/Monitoringberichte/MonitoringberichtEnergie2023.pdf

Calero, F., Cañizares, C. A., Bhattacharya, K., Anierobi, C., Calero, I., de Souza, M. F. Z., Farrokhabadi, M., Guzman, N. S., Mendieta, W., Peralta, D., Solanki, B. V., Padmanabhan, N., & Violante, W. (2023). A Review of Modeling and Applications of Energy Storage Systems in Power Grids. *Proceedings of the IEEE*, *111*(7), 806–831. doi: 10.1109/JPROC.2022.3158607.

Conforti, M., Cornuéjols, G., & Zambelli, G. (2014). *Integer Programming* (Vol. 271). Springer International Publishing. doi: 10.1007/978-3-319-11008-0.

Deguenon, L., Yamegueu, D., Moussa Kadri, S., & Gomna, A. (2023). Overcoming the challenges of integrating variable renewable energy to the grid: A comprehensive review of electrochemical battery storage systems. *Journal of Power Sources*, *580*, 233343. doi: 10.1016/j.jpowsour.2023.233343.

Ember. (2024, February 7). *Distribution of electricity generation in Austria in 2023, by source (accessed on 2024-10-24)*. https://www.statista.com/statistics/1234896/austria-distribution-of-electricity-production-by-source/

Engelmann, A. (2022). *Distributed Optimization with Application to Power Systems and Control (Doctoral thesis, Karlsruhe Institute of Technology)*. KIT Scientific Publishing.

Engelmann, A., Bandeira, M. B., & Faulwasser, T. (2025). Approximate dynamic programming with feasibility guarantees. *IEEE Transactions on Control of Network Systems*, 1–12. doi: 10.1109/TCNS.2025.3526715.

Engelmann, A., Jiang, Y., Houska, B., & Faulwasser, T. (2020). Decomposition of Nonconvex Optimization via Bi-Level Distributed ALADIN. *IEEE Transactions on Control of Network Systems*, *7*(4), 1848–1858. doi: 10.1109/TCNS.2020.3005079.

Engelmann, A., Jiang, Y., Mühlpfordt, T., Houska, B., & Faulwasser, T. (2019). Toward Distributed OPF Using ALADIN. *IEEE Transactions on Power Systems*, *34*(1), 584–594. doi: 10.1109/TPWRS.2018.2867682.

Engelmann, A., Mühlpfordt, T., Jiang, Y., Houska, B., & Faulwasser, T. (2017). Distributed AC Optimal Power Flow using ALADIN. *IFAC-PapersOnLine*, *50*(1), 5536–5541. doi: 10.1016/j.ifacol.2017.08.1095.

Entsoe. (2015, January 5). *ENTSO-E Transparency Platform (accessed on 2022-02-10)*. https://transparency.entsoe.eu/

European Commission. (2021, July 14). *The European Green Deal (accessed on 2024-12-12)*. https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/european-green-deal_en

Evans, M. P., Tindemans, S., & Angeli, D. (2019). Minimising Unserved Energy Using Heterogeneous Storage Units. *IEEE Transactions on Power Systems*, *34*(5), 3647–3656. doi: 10.1109/TPWRS.2019.2910388.

Evans, M. P., Tindemans, S. H., & Angeli, D. (2020). A graphical Measure of Aggregate Flexibility for Energy-Constrained Distributed Resources. *IEEE Transactions on Smart Grid*, *11*(1), 106–117. doi: 10.1109/TSG.2019.2918058.

Evans, M. P., Tindemans, S. H., Angeli, D., & Strbac, G. (2019). Chance-Constrained Ancillary Service Specification for Heterogeneous Storage Devices. *2019 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*, 1–5. doi: 10.1109/ISGTEurope.2019.8905723.

Faulwasser, T., Engelmann, A., Mühlpfordt, T., & Hagenmeyer, V. (2018). Optimal Power Flow: An Introduction to Predictive, Distributed and Stochastic Control Challenges. *at - Automatisierungstechnik*, *66*(7), 573–589. doi: 10.1515/auto-2018-0040.

Figgener, J., Hecht, C., Haberschusz, D., Bors, J., Spreuer, K. G., Kairies, K.-P., Stenzel, P., & Sauer, D. U. (2022). The development of battery storage systems in Germany: A market review (status 2023). doi: 10.48550/ARXIV.2203.06762.

Frank, S., & Rebennack, S. (2016). An introduction to optimal power flow: Theory, formulation, and examples. *IIE Transactions*, *48*(12), 1172–1197. doi: 10.1080/0740817X.2016.1189626.

Fukuda, K. (2021). *Cddlib Reference Manual (accessed on 2024-11-06)*. https://people.inf.ethz.ch/~fukudak/cdd_home/cddlibman2021.pdf

Fukuda, K. (2004). *Frequently Asked Questions in Polyhedral Computation (accessed on 2024-11-06)*. https://people.inf.ethz.ch/fukudak/Doc_pub/polyfaq040618.pdf

Giesecke, J., & Heimerl, S. (2014). *Wasserkraftanlagen: Planung, Bau und Betrieb*. Springer Berlin Heidelberg. doi: 10.1007/978-3-642-53871-1.

Gkatzikis, L., Koutsopoulos, I., & Salonidis, T. (2013). The Role of Aggregators in Smart Grid Demand Response Markets. *IEEE Journal on Selected Areas in Communications*, *31*(7), 1247–1257. doi: 10.1109/JSAC.2013.130708.

Gryparis, E., Papadopoulos, P., Leligou, H. C., & Psomopoulos, C. S. (2020). Electricity demand and carbon emission in power generation under high penetration of electric vehicles. A European Union perspective. *Energy Reports*, *6*, 475–486. doi: 10.1016/j.egyr.2020.09.025.

Gurobi Optimization, LLC. (2024). *Gurobi Optimizer Reference Manual (accessed on 2024-04-01)*. https://www.gurobi.com

IEA. (2020). *Austria 2020 (accessed on 2024-12-12)*. https://www.iea.org/reports/austria-2020

IEA. (2023, June 16). *IEA, Share of population living in a hot climate, 2022, and penetration of air conditioners, 2000-2022 (accessed on 2024-03-15)*. https://www.iea.org/data-and-statistics/charts/share-of-population-living-in-a-hot-climate-2022-and-penetration-of-air-conditioners-2000-2022

Jian, J., Zhang, M., Xu, Y., Tang, W., & He, S. (2023). An Analytical Polytope Approximation Aggregation of Electric Vehicles Considering Uncertainty for the Day-Ahead

Distribution Network Dispatching. *IEEE Transactions on Sustainable Energy*, 1–12. doi: 10.1109/TSTE.2023.3275566.

Jordehi, A. R. (2019). Optimisation of demand response in electric power systems, a review. *Renewable and Sustainable Energy Reviews*, *103*, 308–319. doi: 10.1016/j.rser.2018.12.054.

Kaspar, K. (2024a). *GitHub of the Energy Research Centre - FH Vorarlberg University of Applied Sciences. Repository PyFlexAD (accessed on 2024-11-21)*. https://github.com/rce-fhv/PyFlexAD

Kaspar, K. (2024b). *The Python Package Index. PyFlexAD: Python Flexibility Aggregation and Disaggregation (accessed on 2024-11-21)*. https://pypi.org/project/pyflexad/

Koch, S., Mathieu, J. L., & Callaway, D. S. (2011). Modeling and control of aggregated heterogeneous thermostatically controlled loads for ancillary services. *17th Power Systems Computation Conference*.

Matoušek, J. (Ed.). (2002). *Lectures on Discrete Geometry* (Vol. 212). Springer New York. doi: 10.1007/978-1-4613-0039-7.

Mukhi, K., & Abate, A. (2023). An exact characterisation of flexibility in populations of electric vehicles. *2023 62nd IEEE Conference on Decision and Control (CDC)*, 6582–6587. doi: 10.1109/CDC49753.2023.10383521.

Müller, F. L., Sundström, O., Szabó, J., & Lygeros, J. (2015). Aggregation of energetic flexibility using zonotopes. *2015 54th IEEE Conference on Decision and Control (CDC)*, 6564–6569. doi: 10.1109/CDC.2015.7403253.

Müller, F. L., Szabó, J., Sundström, O., & Lygeros, J. (2019). Aggregation and Disaggregation of Energetic Flexibility from Distributed Energy Resources. *IEEE Transactions on Smart Grid*, *10*(2), 1205–1214. doi: 10.1109/TSG.2017.2761439.

My Electric Avenue. (2015). *My Electric Avenue (accessed on 2022-01-01)*. https://myelectricavenue.info/

Nazir, M. S., Hiskens, I. A., Bernstein, A., & Dall'Anese, E. (2018). Inner Approximation of Minkowski Sums: A Union-Based Approach and Applications to Aggregated Energy Resources. *2018 IEEE Conference on Decision and Control (CDC)*, 5708–5715. doi: 10.1109/CDC.2018.8618731.

Nieße, A., Beer, S., Bremer, J., Hinrichs, C., Lünsdorf, O., & Sonnenschein, M. (2014). Conjoint Dynamic Aggregation and Scheduling Methods for Dynamic Virtual Power Plants. *2014 Federated Conference on Computer Science and Information Systems*, 1505–1514. doi: 10.15439/2014F76.

Öztürk, E., Faulwasser, T., Worthmann, K., PreißInger, M., & Rheinberger, K. (2024). Alleviating the Curse of Dimensionality in Minkowski Sum Approximations

of Storage Flexibility. *IEEE Transactions on Smart Grid*, *15*(6), 5733–5743. doi: 10.1109/TSG.2024.3420156.

Öztürk, E., Kaspar, K., Faulwasser, T., Worthmann, K., Kepplinger, P., & Rheinberger, K. (2025). A python toolbox for flexibility aggregation and disaggregation: Pyflexad. *Sustainable Energy, Grids and Networks*, *44*, 102033. doi: 10.1016/j.segan.2025.102033.

Öztürk, E., Kaspar, K., Faulwasser, T., Worthmann, K., Kepplinger, P., & Rheinberger, K. (2024). Towards efficient aggregation of storage flexibilities in power grids. *NEIS 2024; Conference on Sustainable Energy Supply and Energy Storage Systems*, 151–155. doi: 10.30420/566464020.

Öztürk, E., & Rheinberger, K. (2024, January 9). *Supplement Website (accessed on 2024-02-01)*. https://github.com/klaus-rheinberger/AggrFlex_Vertex_Supplement

Öztürk, E., Rheinberger, K., Faulwasser, T., Worthmann, K., & Preißinger, M. (2022). Aggregation of Demand-Side Flexibilities: A Comparative Study of Approximation Algorithms. *Energies*, *15*(7), 2501. doi: 10.3390/en15072501.

Parliament, E. (2019, June). *DIRECTIVE (EU) (accessed on 2024-09-03)*. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32019L1161

Pozo, D. (2023). Convex Hull Formulations for Linear Modeling of Energy Storage Systems. *IEEE Transactions on Power Systems*, *38*(6), 5934–5936. doi: 10.1109/TPWRS.2023.3304131.

Commission for Energy Regulation. (2012). *Irish Social Science Data Archive (accessed on 2022-02-10)*. https://www.ucd.ie/issda/data/commissionforenergyregulationcer/

Rheinberger, K. (2021a, August 11). *DSM-Data (accessed on 2022-02-10)*. https://github.com/klaus-rheinberger/DSM-data

Rheinberger, K. (2021b, November 9). *Supplement Website (accessed on 2024-01-11)*. https://github.com/klaus-rheinberger/AggrFlex_Supplement

Rheinberger, K., Kepplinger, P., & Preißinger, M. (2021). Flexibility Control in Autonomous Demand Response by Optimal Power Tracking. *Energies*, *14*(12), 3568. doi: 10.3390/en14123568.

Sajjad, I. A., Chicco, G., & Napoli, R. (2016). Definitions of Demand Flexibility for Aggregate Residential Loads. *IEEE Transactions on Smart Grid*, *7*(6), 2633–2643. doi: 10.1109/TSG.2016.2522961.

Sass, S., Faulwasser, T., Hollermann, D. E., Kappatou, C. D., Sauer, D., Schütz, T., Shu, D. Y., Bardow, A., Gröll, L., Hagenmeyer, V., et al. (2020). Model compendium, data, and optimization benchmarks for sector-coupled energy systems. *Computers & chemical engineering*, *135*, 106760.

Silva, J., Sumaili, J., Bessa, R. J., Seca, L., Matos, M. A., Miranda, V., Caujolle, M., Goncer, B., & Sebastian-Viana, M. (2018). Estimating the Active and Reactive Power Flexibility Area at the TSO-DSO Interface. *IEEE Transactions on Power Systems*, *33*(5), 4741–4750. doi: 10.1109/TPWRS.2018.2805765.

Taheri, S., Kekatos, V., Veeramachaneni, S., & Zhang, B. (2022). Data-Driven Modeling of Aggregate Flexibility Under Uncertain and Non-Convex Device Models. *IEEE Transactions on Smart Grid*, *13*(6), 4572–4582. doi: 10.1109/TSG.2022.3185532.

Tian, X., Liu, L., & Shen, G. (2024). A review on the mathematical models of thermostatically controlled load. *Architectural Intelligence*, *3*(1), 33. doi: 10.1007/s44223-024-00075-y.

Tiwary, H. R. (2008). On the Hardness of Computing Intersection, Union and Minkowski Sum of Polytopes. *Discrete & Computational Geometry*, *40*(3), 469–479. doi: 10.1007/s00454-008-9097-3.

Trangbaek, K., & Bendtsen, J. (2012). Exact constraint aggregation with applications to smart grids and resource distribution. *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, 4181–4186. doi: 10.1109/CDC.2012.6426475.

Trangbaek, K., Petersen, M., Bendtsen, J., & Stoustrup, J. (2011). Exact power constraints in smart grid control. *IEEE Conference on Decision and Control and European Control Conference*, 6907–6912. doi: 10.1109/CDC.2011.6160539.

Wen, Y., Hu, Z., He, J., & Guo, Y. (2024). Improved inner approximation for aggregating power flexibility in active distribution networks and its applications. *IEEE Transactions on Smart Grid*, *15*(4), 3653–3665. doi: 10.1109/TSG.2023.3340157.

Wen, Y., Hu, Z., You, S., & Duan, X. (2022). Aggregate Feasible Region of DERs: Exact Formulation and Approximate Models. *IEEE Transactions on Smart Grid*, *13*(6), 4405–4423. doi: 10.1109/TSG.2022.3179998.

Yang, T., Yi, X., Wu, J., Yuan, Y., Wu, D., Meng, Z., Hong, Y., Wang, H., Lin, Z., & Johansson, K. H. (2019). A survey of distributed optimization. *Annual Reviews in Control*, *47*, 278–305. doi: 10.1016/j.arcontrol.2019.05.006.

Zargari, N., Ofir, R., Chowdhury, N. R., Belikov, J., & Levron, Y. (2023). An Optimal Control Method for Storage Systems with Ramp Constraints, Based on an On-Going Trimming Process. *IEEE Transactions on Control Systems Technology*, *31*(1), 493–496. doi: 10.1109/TCST.2022.3169906.

Zhang, M., Xu, Y., Shi, X., & Guo, Q. (2024). A Fast Polytope-Based Approach for Aggregating Large-Scale Electric Vehicles in the Joint Market under Uncertainty. *IEEE Transactions on Smart Grid*, *15*(1), 701–713. doi: 10.1109/TSG.2023.3274198.

Zhao, L., Hao, H., & Zhang, W. (2016). Extracting flexibility of heterogeneous deferrable loads via polytopic projection approximation. *2016 IEEE 55th Conference on Decision and Control (CDC)*, 6651–6656. doi: 10.1109/CDC.2016.7799293.

Zhao, L., Zhang, W., Hao, H., & Kalsi, K. (2017). A Geometric Approach to Aggregate Flexibility Modeling of Thermostatically Controlled Loads. *IEEE Transactions on Power Systems*, *32*(6), 4721–4731. doi: 10.1109/TPWRS.2017.2674699.

Zhen, J., & den Hertog, D. (2018). Computing the Maximum Volume Inscribed Ellipsoid of a Polytopic Projection. *INFORMS Journal on Computing*, *30*(1), 31–42. doi: 10.1287/ijoc.2017.0763.

Zheng, Y., & Liu, Q. (2022). A review of distributed optimization: Problems, models and algorithms. *Neurocomputing*, *483*, 446–459. doi: 10.1016/j.neucom.2021.06.097.

Zhou, B., Liu, S., Lu, S., Cao, X., & Zhao, W. (2017). Cost–benefit analysis of pumped hydro storage using improved probabilistic production simulation method. *The Journal of Engineering*, *2017*(13), 2146–2151. doi: 10.1049/joe.2017.0709.

Ziegler, G. M. (1995). *Lectures on Polytopes* (Vol. 152). Springer New York. doi: 10.1007/978-1-4613-8431-1.

Appendix

A. Illustrative Code for Simple Aggregation

The following code demonstrates the application of the aggregation tool PyFlexAD in a simple example featuring two stationary batteries. The aggregated flexibility is computed and compared to a central control scheme, while the included plotting functions illustrate their application. This example is reproduced (Öztürk et al., 2025).

```
import matplotlib.pyplot as plt
import numpy as np

import PyFlexAD.models.bess.tesla as tesla
from PyFlexAD.physical.stationary_battery import StationaryBattery
from PyFlexAD.physical.stationary_battery import BESSUsage
from PyFlexAD.virtual.aggregator import Aggregator
from PyFlexAD.math.signal_vectors import SignalVectors
from PyFlexAD.utils.algorithms import Algorithms
from PyFlexAD.optimization.vertex_based_power_controller import
  VertexBasedPowerController
from PyFlexAD.optimization.centralized_power_controller import
  CentralizedPowerController

"""settings"""
d = 2  # number of time intervals
dt = 0.25  # interval duration in hours
system_power_demand = np.array([[23.0, 21.0]])  # total power demand in kW
algorithm = Algorithms.IABVG  # virtualization and aggregation algorithm
S_0 = 6.5  # initial battery capacity in kWh
S_f = 5.0  # final battery capacity in kWh

"""instantiate energy storage resources"""
usage = BESSUsage(initial_capacity=S_0, final_capacity=S_f, d=d, dt=dt)
bess_1 = StationaryBattery.new(hardware=tesla.power_wall_2, usage=usage)
bess_2 = StationaryBattery.new(hardware=tesla.power_wall_3, usage=usage)

"""virtualize"""
direction_vectors = SignalVectors.new(d)
virtual_ess_1 = bess_1.to_virtual(algorithm, direction_vectors)
virtual_ess_2 = bess_2.to_virtual(algorithm, direction_vectors)

"""aggregate"""
aggregator = Aggregator.aggregate([virtual_ess_1, virtual_ess_2], algorithm)
agg_exact = Aggregator.from_physical([bess_1, bess_2], algorithm=Algorithms.EXACT)

"""optimize power with centralized controller"""
centralized_controller = CentralizedPowerController(system_power_demand)
```

```
cc_power = centralized_controller.optimize([bess_1, bess_2])

"""optimize power with vertex-based controller"""
vertex_controller = VertexBasedPowerController(system_power_demand)
vc_power = vertex_controller.optimize(aggregator)
exact_power = vertex_controller.optimize(agg_exact)

  """plot polytopes"""
fig, axes = plt.subplots()
aggregator.get_items()[0].plot_polytope_2d(axes,line_style='--',label=r"$\mathcal{P}_1$")
aggregator.get_items()[0].plot_load_profile_2d(axes,label=r"$OP_1$")
aggregator.get_items()[1].plot_polytope_2d(axes,line_style='--',label=r"$\mathcal{P}_2$")
aggregator.get_items()[1].plot_load_profile_2d(axes,label=r"$OP_2$")
aggregator.plot_polytope_2d(axes,label=r'$\mathcal{A}$',color='b',hatch='//',fill=True)
aggregator.plot_load_profile_2d(axes,label=r'$OP_1 + OP_2$',color='g')
agg_exact.plot_polytope_2d(axes,label=r'$\mathcal{M}$',color='r')
agg_exact.plot_load_profile_2d(axes, label=r'$OP_{exact}$')
axes.legend(loc="lower left")
```

B. Illustrative Code for Hierarchical Aggregation

The following code demonstrates the application of the aggregation tool PyFlexAD in a hierarchical aggregation setting. The hierarchically aggregated flexibility is computed and compared with a centralized control scheme that does not use a hierarchical structure. The accompanying visualization function for plotting the hierarchical system is also shown.

```
import matplotlib.pyplot as plt
import numpy as np

from PyFlexAD.math.signal_vectors import SignalVectors
from PyFlexAD.models.bess import bess_models
from PyFlexAD.physical.stationary_battery import StationaryBattery
from PyFlexAD.utils.algorithms import Algorithms
from PyFlexAD.utils.network_graph import NetworkGraph
from PyFlexAD.virtual.aggregator import Aggregator
from PyFlexAD.optimization.vertex_based_power_controller import
VertexBasedPowerController
from PyFlexAD.optimization.centralized_power_controller import
CentralizedPowerController

"""settings"""
d = 5 # number of time intervals
dt = 0.25  # interval duration in hours
n_entities = 2 # number of devices per aggregator
system_power_demand = np.array([[50,100,70,60,50]]) # total power demand in kW

"""main"""
np.random.seed(2)

agg_list, dev_list = ([] for k in range(2))
signal_vectors = SignalVectors.new(d)
for index, bess_model in enumerate(bess_models):
    esr_list_1 = [StationaryBattery.random_usage(hardware=bess_model, d=d, dt=dt, id=f
"{bess_model.name}[{i + 1}]") for i in range(n_entities)]
    dev_list += esr_list_1
    agg_list += [Aggregator.from_physical(items=esr_list_1, algorithm=Algorithms.IABVG
, signal_vectors=signal_vectors,id=f"Sub-Aggregator[{index+1}]")]

agg = Aggregator.aggregate(agg_list, id="Aggregator", algorithm=Algorithms.IABVG)

"""plot network graph"""
graph = NetworkGraph.from_virtual([agg, ])
graph.create_tree()
_, ax = plt.subplots(figsize=(11,6)) #figsize=(10, 10)
graph.plot_tree(ax=ax, node_size=100, parent_node_size=500)
```

```
36
37      """optimize power with centralized controller"""
38      centralized_controller = CentralizedPowerController(system_power_demand)
39      cc_power = centralized_controller.optimize(dev_list)
40
41      """optimize power with vertex-based controller"""
42      vertex_controller = VertexBasedPowerController(system_power_demand)
43      vc_power = vertex_controller.optimize(agg)
```