Duality-driven optimization in energy management

*Offline and online algorithms
for resource allocation problems*

Martijn H. H. Schoot Uiterkamp

Members of the graduation committee:

prof. dr.	J. L. Hurink	University of Twente (promotor)
dr. ir.	M. E. T. Gerards	University of Twente (assistant-promotor)
prof. dr. ir.	G. J. M. Smit	University of Twente
prof. dr.	M. J. Uetz	University of Twente
prof. dr. ir.	D. den Hertog	University of Amsterdam
prof. dr.	A. P. Zwart	Eindhoven University of Technology
prof. dr.	P. Pinson	Technical University of Denmark
prof. dr.	J. N. Kok	University of Twente (chairman and secretary)

UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering, Mathematics and Computer Science, chairs of Computer Architecture for Embedded Systems (CAES) and Discrete Mathematics and Mathematical Programming (DMMP).

**DIGITAL SOCIETY
INSTITUTE**

DSI Ph.D. Thesis Series No. 21-005
Digital Society Institute
PO Box 217, 7500 AE Enschede, The Netherlands

This research has been conducted within the SIMPS project (project number 647.002.003). This research is supported by the Dutch Research Council (NWO).

Copyright © 2021 Martijn H. H. Schoot Uiterkamp, Enschede, the Netherlands. This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc/4.0/deed.en_US.

This thesis was typeset using $\text{\LaTeX}$, $\text{\textit{TikZ}}$, Vim, and PGFPlots. This thesis was printed by Gildeprint Drukkerijen, The Netherlands. The cover image is credited to Johannes Plenio.

ISBN 978-90-365-5161-8
ISSN 2589-7721; DSI Ph.D. Thesis Series No. 21-005
DOI 10.3990/1.9789036551618

DUALITY-DRIVEN OPTIMIZATION IN ENERGY
MANAGEMENT

OFFLINE AND ONLINE ALGORITHMS FOR RESOURCE ALLOCATION
PROBLEMS

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. ir. A. Veldkamp,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op vrijdag 23 april 2021 om 16.45 uur

door

Martijn Hermannus Hendrik Schoot Uiterkamp

geboren op 13 oktober 1993
te Enschede

Dit proefschrift is goedgekeurd door:

prof. dr. J. L. Hurink (promotor)
dr. ir. M. E. T. Gerards (assistant-promotor)

ABSTRACT

Energy has been the driving force behind virtually all major technological developments of the last 200 years. Over time, a stable and reliable energy infrastructure has been created that ensures an adequate supply of energy to our society, whose proper operation depends completely on this infrastructure. However, the future stability and reliability of our current energy systems are threatened as shown by new scientific insights on the effect of these systems on our planet. For instance, fossil fuels form a crucial energy source for our current energy infrastructure but, at the current consumption rate, they will have been depleted way before the end of the 21st century. Moreover, the combustion of these fuels leads to a major increase in CO₂ emissions, which in turn accelerates human-induced global warming. A widely advocated solution to this problem is to switch to more sustainable and cleaner energy sources such as solar and wind power. Moreover, many innovations have been done that led to an increase in energy-efficiency of devices.

Although these new energy sources and innovations in energy-efficiency are seen as promising developments towards a reliable future energy system, they come with several disadvantages that have become apparent particularly in the Netherlands. One major disadvantage is the uncertainty of both energy supply and demand, i.e., the dependence of supply on weather conditions and the dependence of demand on human behavior. Furthermore, due to the rapidly increasing electrification of heating and transport, there is also a rapid increase in the amount of electricity that needs to be transported throughout the electricity distribution network. The capacity of the current (Dutch) network, however, is not sufficiently large to accommodate this increase in transported electricity.

To resolve these two issues and to avoid huge financial investments in the energy infrastructure, it is crucial to actively manage the energy flows within an energy system or device. Hereby, one can adequately react to unexpected demand or supply behavior and ensure the proper operation of the system or device without violating its operational constraints. This leads to the formulation of *energy management* problems, which are often viewed from either of two perspectives. The first one is the practical perspective, where the primary goal is to obtain a solution that can actually be implemented and used in practice. Although the proposed solutions often perform well in practice, one disadvantage of this perspective is that it is generally very hard to provide insight in or justification of why this is the case. The second perspective is the theoretical one, where the primary goal is to understand the structural properties of a given model or

problem. Hereby, the focus is usually on the theoretical (asymptotic) efficiency of solution approaches, i.e., how well the execution time and required space of an approach scale with the problem input size. Although the resulting approaches usually come with good (theoretical) performance guarantees, it is often questionable whether they perform well in practice. One reason for this is that they often require complex and advanced subroutines or data structures to achieve the promised performance guarantee. Moreover, in order to derive theoretical performance guarantees, usually severe problem simplifications are required. As a consequence, these approaches are not suitable for practical use because either they are too hard and time-consuming to implement or the simplified problem is not an appropriate representation of the given real-life problem.

The aim of this thesis is to develop solution approaches and algorithms for energy management problems that combine the merits of both the practical and theoretical perspectives as described above. This means that, on the one hand, the developed approaches can be implemented in practice within reasonable time and with limited effort and, on the other hand, they come with theoretical performance guarantees on, e.g., the execution time or solution accuracy. To achieve this, energy management problems are studied in this thesis from a mathematical perspective as a special class of so-called *resource allocation problems* (RAPs). This type of problems has been studied extensively since the 1950s and has many applications within and outside of energy management. The resulting extensive body of research on these problems forms a suitable starting point for modeling energy management problems and incorporating practical aspects into these models while maintaining nice theoretical performance guarantees.

To solve the considered energy management problems and RAPs, a so-called duality perspective is employed. The idea behind this perspective is, roughly, that the problem might become significantly easier to solve if first some of its restrictions are relaxed and only enforced in a later stadium. Over the last decades, this idea has developed into a well-studied and mature mathematical theory of duality that is used to solve a wide range of optimization problems occurring in operations research, finance, and engineering. In particular, for many classes of the aforementioned RAPs, the state-of-the-art solution approaches and algorithms are based on this mathematical duality theory. Thus, this “duality-driven optimization” serves as a promising technique for solving energy management problems.

The concrete contributions of this thesis are twofold. The first contribution is the extension of several existing energy management problems and the corresponding RAPs so that practical requirements and physical properties of the given system or device are incorporated while maintaining theoretical performance guarantees. In particular, several practical aspects of electric vehicle charging are considered such as the usage of three-phase charging systems, the choice between a fixed number of distinct charging rates, and the influence of the charging rate on the efficiency of the charging process. Using simpler and well-studied

RAPs as building blocks, the resulting new problems are solved by newly developed approaches that take these practical aspects into account. Furthermore, a new solution approach and algorithm is presented for RAPs occurring in, e.g., storage system scheduling, vessel speed optimization, and processor scheduling. This algorithm is an improvement over existing algorithms for these problems because it achieves the best-known theoretical performance guarantee with regard to execution time without relying on complex and advanced data structures. As a consequence, the proposed algorithm is much more suitable for actual practical implementation. For all the newly presented algorithms, performance guarantees on the asymptotic worst-case execution time are established, which are subsequently verified and confirmed by extensive computational experiments.

The second contribution is a new framework for solving energy management problems where uncertainty is present in several of the problem parameters. This framework, called “Online Duality-Driven Optimization” (ODDO), solves the given problem *online*, i.e., instead of determining the energy usage of the given system or device completely a priori for a given time horizon, the energy usage at a given moment in time is decided at that particular moment. As opposed to existing frameworks for optimization under uncertainty, the proposed ODDO framework does not require predictions of the uncertain data themselves. Instead, a duality perspective is employed wherein a solution to the given problem is characterized not in terms of the (uncertain) input data but in terms of so-called *multipliers* that represent the structure of the solution. The key idea in this duality-driven approach is to predict these multipliers instead of the actual uncertain data itself and use these multiplier predictions as guidance for the online solution process. Although the duality perspective is usually used to tackle several types of mathematical optimization problems offline, i.e., when all problem parameters are known, this thesis shows that it can also be used to solve problems with uncertain problem parameters. As a proof of concept, the resulting ODDO framework is applied to several energy management problems and computational experiments are conducted to assess the performance of the framework. The results of these experiments indicate that the framework is able to compute near-optimal solutions to the considered energy management problems using simple multiplier prediction strategies. This demonstrates the potential of the ODDO framework as a successful method for solving energy management problems without a need for predicting detailed energy supply and demand profiles.

SAMENVATTING

ix

Energie is al 200 jaar lang de drijfveer achter vrijwel alle grote technologische ontwikkelingen. Langzamerhand is er een stabiele en betrouwbare energie-infrastructuur gecreëerd die een adequaat aanbod van energie garandeert voor onze samenleving, die volledig afhankelijk is van deze infrastructuur. Echter, de toekomstige stabiliteit en betrouwbaarheid van onze huidige energiesystemen wordt bedreigd, zoals aangetoond door nieuwe wetenschappelijke inzichten over het effect van deze systemen op onze planeet. Fossiele brandstoffen, bijvoorbeeld, vormen een cruciale energiebron voor onze huidige energie-infrastructuur maar zullen, onder het huidige verbruik, ver voor het einde van de 21^{ste} eeuw opgeraakt zijn. Verder leidt het verbranden van deze brandstoffen tot een grote toename van de CO₂ uitstoot, die op zijn beurt de door mensen veroorzaakte opwarming van de aarde versnelt. Een breed gedragen oplossing voor dit probleem is om over te stappen op duurzamere en schonere energiebronnen zoals zon- en windkracht. Verder hebben er veel innovaties plaatsgevonden die geleid hebben tot een toename van de energie-efficiëntie van apparaten.

Hoewel deze nieuwe energiebronnen en innovaties in energie-efficiëntie als veelbelovende ontwikkelingen richting een betrouwbaar toekomstig energiesysteem worden gezien, brengen ze een aantal nadelen met zich mee die zich met name in Nederland openbaard hebben. Een van deze nadelen is de onzekerheid van zowel het aanbod van als de vraag naar energie, i.e., het aanbod is afhankelijk van het weer en de vraag is afhankelijk van menselijk gedrag. Bovendien, vanwege de snel toenemende elektrificatie van warmte en vervoer is er ook een snelle toename van de hoeveelheid elektriciteit die door het elektriciteitsnetwerk vervoerd moet worden. De capaciteit van het huidige (Nederlandse) netwerk is echter niet groot genoeg om deze toename van de vervoerde elektriciteit aan te kunnen.

Om deze twee kwesties op te lossen en gigantische financiële investeringen in de energie-infrastructuur te vermijden, is het cruciaal om de energiestromen binnen een energiesysteem of apparaat actief te beheren. Hierdoor kan men adequaat reageren op onverwacht vraag- of aanbodgedrag en de goede werking van het systeem of apparaat garanderen zonder dat zijn operationele beperkingen overschreden worden. Dit leidt tot het formuleren van *energiemanagementproblemen*, die vaak vanuit een van de volgende twee perspectieven beschouwd worden. Het eerste perspectief is het praktische perspectief, waar het voornaamste doel is om een oplossing te verkrijgen die daadwerkelijk in de praktijk geïmplementeerd en gebruikt kan worden. Hoewel de voorgestelde oplossingen in de praktijk vaak goed werken, is een nadeel van dit perspectief dat het in het algemeen erg

moeilijk is om inzicht of een verklaring te geven voor waarom dit zo is. Het tweede perspectief is het theoretische perspectief, waar het voornaamste doel is om de structurele eigenschappen van een gegeven model of probleem te begrijpen. Hierbij is de focus meestal op de theoretische (asymptotische) efficiëntie van de oplossings technieken, i.e., hoe goed de uitvoeringstijd en benodigde opslagruimte van een aanpak schaalbaar is met de grootte van de invoer van het probleem. Hoewel de daaruitvolgende aanpakken meestal goede (theoretische) prestatiegaranties met zich meebrengen, is het vaak de vraag hoe goed ze in de praktijk werken. Een reden hiervoor is dat er vaak complexe en geavanceerde subroutines nodig zijn om de beloofde prestatiegaranties te behalen. Verder is het vaak nodig om het probleem stevig te versimpelen om deze theoretische prestatiegaranties af te kunnen leiden. Als gevolg hiervan zijn deze aanpakken niet geschikt voor de praktijk, danwel omdat ze te moeilijk en tijdrovend zijn om te implementeren, danwel omdat het versimpelde probleem geen geschikte veergave is van het gegeven echte probleem.

Het doel van dit proefschrift is om oplossings technieken en algoritmes voor energiemanagerproblemen te ontwikkelen die de voordelen combineert van de praktische en theoretische perspectieven zoals hierboven beschreven. Dit betekent dat, aan de ene kant, de ontwikkelde aanpakken kunnen binnen een redelijk tijd en zonder al te veel moeite in de praktijk geïmplementeerd worden en, aan de andere kant, ze theoretische prestatiegaranties met zich meebrengen met betrekking tot, e.g., de uitvoeringstijd of de nauwkeurigheid van de oplossing. Om dit te bereiken worden energiemanagerproblemen in dit proefschrift bestudeerd vanuit een wiskundig perspectief als een speciaal geval van zogenoemde *allocatieproblemen*. Dit soort problemen wordt al sinds de jaren vijftig intensief bestudeerd en heeft veel toepassingen binnen en los van energiemanager. Het daaruitvolgende vele onderzoek naar deze problemen vormt een geschikt startpunt voor het modelleren van energiemanagerproblemen en het integreren van praktische aspecten in deze modellen met behoud van de mooie theoretische prestatiegaranties.

Om de beschouwde energiemanagerproblemen en allocatieproblemen op te lossen, wordt een zogenoemd dualiteitsperspectief aangewend. Het idee achter dit perspectief is grofweg dat het probleem mogelijk aanzienlijk eenvoudiger is om op te lossen als eerst een aantal restricties van het probleem weggelaten wordt en pas in een later stadium opgelegd worden. Gedurende de laatste decennia is dit idee doorontwikkeld tot een goed bestudeerd en volwassen wiskunde dualiteits theorie die gebruikt wordt om een breed scala aan operationele, financiële, en technische optimaliseringsproblemen op te lossen. In het bijzonder zijn de state-of-the-art oplossings technieken en algoritmes voor veel van dit soort allocatieproblemen gebaseerd op deze wiskunde dualiteitstheorie. Deze “dualiteits-gedreven optimalisatie” dient dus als een veelbelovende techniek voor het oplossen van energiemanagerproblemen.

De concrete bijdragen van dit proefschrift zijn tweevoudig. De eerste bijdrage is de uitbreiding van een aantal bestaande energiemanagerproblemen en de bijbehorende allocatieproblemen zodat praktische restricties en fysieke eigenschappen van het gegeven systeem of apparaat meegenomen worden met behoud van theoretische prestatiegaranties. In het bijzonder wordt een aantal praktische aspecten van het laden van elektrische voertuigen beschouwd, zoals het gebruik van driefase laders, de keuze tussen een beperkt aantal vastgestelde laadcapaciteiten, en de invloed van de laadcapaciteit op de efficiëntie van het laadproces. Door simpelere en goed bestudeerde allocatieproblemen als bouwblokken te gebruiken, worden de daaruitvolgende nieuwe problemen opgelost met behulp van nieuw ontwikkelde technieken die deze praktische aspecten meenemen. Verder wordt een nieuwe oplossingstechniek en algoritme gepresenteerd voor allocatieproblemen met betrekking tot, e.g., het aansturen van energieopslag, de kruissnelheid van voertuigen, en het uitvoeren van taken op een processor. Dit algoritme is een verbetering ten opzichte van bestaande algoritmes voor deze problemen omdat het de beste bekende theoretische prestatiegarantie behaalt met betrekking tot uitvoeringstijd, zonder het gebruik van complexe en geavanceerde datastructuren. Als gevolg hiervan is het voorgestelde algoritme veel geschikter voor daadwerkelijke implementatie in de praktijk. Voor alle nieuwe gepresenteerde algoritmes worden prestatiegaranties met betrekking tot de asymptotische worst-case uitvoeringstijd vastgesteld, die vervolgens geverifieerd en bevestigd worden door uitvoerige computationele experimenten.

De tweede bijdrage is een nieuw framework voor het oplossen van energiemanagerproblemen waarin onzekerheid aanwezig is in een aantal van de probleemparameters. Dit framework, genaamd “Online Duality-Driven Optimization” (online dualiteits-gedreven optimalisatie; ODDO), lost een gegeven probleem *online* op, i.e., in plaats van dat het energieverbruik van het gegeven systeem of apparaat geheel van tevoren wordt bepaald voor een gegeven tijdsbestek, wordt het energieverbruik op een gegeven moment bepaald op dat specifieke moment. In tegenstelling tot bestaande frameworks voor optimalisatie onder onzekerheid, het voorgestelde ODDO framework heeft geen voorspelling nodig van de onzekere data zelf. In plaats daarvan wordt een dualiteitsperspectief aangewend waarin een oplossing voor het gegeven probleem niet gekarakteriseerd wordt in termen van de (onzekere) invoerdata maar in termen van zogenoemde *multipliers* die de structuur van de oplossing weergeven. Het belangrijkste idee in deze dualiteits-gedreven aanpak is om deze multipliers te voorspellen in plaats van de daadwerkelijke onzekere data zelf en om deze voorspelling van de multipliers te gebruiken als invoer voor het online oplossingsproces. Hoewel het dualiteitsperspectief meestal gebruikt wordt om bepaalde soorten optimaliseringsproblemen offline aan te pakken, i.e., wanneer alle probleemparameters zeker zijn, laat dit proefschrift zien dat het ook gebruikt kan worden om problemen met onzekere probleemparameters op te lossen. Het daaruitvolgende ODDO framework wordt als een proof-of-concept toegepast op een aantal energiemanagerproblemen en computationele experimenten worden uitgevoerd

om de werking van het framework te beoordelen. De resultaten van deze experimenten geven aan dat het framework in staat is om oplossingen voor de beschouwde energiemanagementproblemen te bepalen die bijna optimaal zijn en waarvoor slechts simpele voorspellingsstrategieën benodigd zijn. Dit laat de potentie van het ODDO framework zien als een succesvolle methode voor het oplossen van energiemanagementproblemen zonder dat voorspellingen van gedetailleerde vraag- en aanbodprofielen van energie nodig zijn.

DANKWOORD

xiii

Dit proefschrift is het resultaat van vier jaar (en een beetje) aan promotieonderzoek, inclusief alle ups en downs die daar vaak mee gepaard gaan. Hoewel alleen mijn naam op de kaft staat, zou dit proefschrift er niet zijn geweest zonder de hulp en support van vele anderen. Omdat ik zeker weet dat ik mensen ga vergeten als ik al deze anderen probeer te noemen, wil ik hierbij allereerst iedereen bedanken die op welke manier dan ook heeft bijgedragen aan het tot stand komen van dit proefschrift. Daarnaast zijn er een aantal personen die ik persoonlijk wil noemen en bedanken.

Ten eerste wil ik Johann en Marco bedanken. Jullie hebben me vrij gelaten om mijn eigen richting aan het onderzoek te geven, waarvoor ik jullie erg dankbaar ben. Johann, nadat we elkaar voor het eerst ontmoet hadden (dat was al aan het begin van mijn master in 2014!), merkte ik al vrij snel dat er een “klik” was en dat we niet alleen over de inhoud, maar ook over heel veel andere zaken op één lijn zaten. Je hebt het vermogen om structuur en duidelijkheid te scheppen in de meest diffuse situaties, of het nu gaat om het oplossen van een ingewikkeld wiskundig probleem of om het verhelderen van vage communicatie en sociale structuren. Ik bewonder je openheid naar en je betrokkenheid bij de mensen om je heen en heb veel van je geleerd over de sociale kant van onderzoek doen en van de wetenschappelijke wereld. Dankjewel voor al je inzichten, voor alle mooie en soms ook bijzondere gesprekken, en voor de geweldige begeleiding de afgelopen jaren!

Marco, ondanks dat je bij vlagen erg druk was met onderwijs en minder plezierige management-achtige zaken, maakte je altijd tijd voor me vrij om mijn onderzoek te bespreken. Ik heb veel gehad aan onze discussies en in het bijzonder aan je brede kennis die veel verder gaat dan “ons” onderwerp energiemangement. Daarnaast was je altijd geïnteresseerd in discussies over andere wetenschappelijke en minder wetenschappelijke onderwerpen, waar ik met veel plezier aan terug denk. Je hebt me altijd veel tips gegeven over hoe ik bepaalde zaken kon aanpakken, variërend van het schrijven van papers en de thesis tot het begeleiden van studenten en effectief tijdmanagement. Dankjewel voor al je hulp en begeleiding de afgelopen jaren!

I would like to thank all committee members for taking their place in the committee and for assessing my thesis. In het bijzonder, Gerard, dankjewel voor het opzetten van het SIMPS project en voor je hulp en betrokkenheid bij het begin van mijn promotie. Also, Pierre, thank you for hosting me at DTU: I learned

a lot about forecasting in the energy domain and enjoyed our discussions about all kinds of aspects of doing research.

Binnen de UT maakte ik deel uit van de vakgroep CAES, misschien wel een van de meest sociale en ondernemende vakgroepen die de Zilverling rijk is. Ik wil graag iedereen in de vakgroep bedanken voor alle koffiepauzes, lunchwandelingen, vrijdagmiddagborrels, en alle andere leuke activiteiten die we ondernomen hebben. Binnen de vakgroep maakte ik deel uit van de “energie-groep”¹ en vertoefde ik in het befaamde “energie-hok”, toch wel het energieke middelpunt van de vijfde verdieping (letterlijk in ieder geval). Gerwin, Thijs, Gijs, Victor, Jens, en alle bachelor- en masterstudenten met wie ik hier samen heb gezeten, dank jullie wel voor de gezellige sfeer, leuke discussies, en alle spontane gekkigheid. Zonder dit alles zouden de afgelopen jaren een stuk saaier zijn geweest! Verder wil ik Marlous en Nicole graag bedanken voor al hun ondersteuning en de positiviteit die ze in de groep brengen.

In het bijzonder binnen de energie-groep en het energie-hok wil ik Victor noemen. Sinds we elkaar ontmoetten tijdens de Kick-In van 2011, hebben we het grootste deel van ons UT-avontuur, zowel studie/promotie als daarbuiten, samen afgelegd. Ik ben dan ook heel blij dat jij een van mijn paranimfen wilt zijn bij deze afsluiting van mijn UT-avontuur.

Ik wil ook mijn familie en vrienden bedanken voor hun aanmoedigingen en voor de broodnodige afleiding van het werk. Soms heb ik de neiging om me te veel op te laten slokken door alle wiskundige formules, maar dankzij jullie word ik er steeds beter in om de juiste balans te vinden (alhoewel dit soms nog steeds “Work in Progress” is). In het bijzonder wil ik mijn ouders en zus (en tevens paranimf) bedanken. Jullie hebben me geleerd om mijn eigen pad te volgen en nooit op te geven. Zonder jullie had ik nooit kunnen bereiken wat ik nu bereikt heb. Dank jullie wel voor jullie onvoorwaardelijke steun en liefde.

Lastly, I would like to thank Katharina. Thank you for always believing in me, even when I do not. I am happy that I could share the majority of my Ph.D. adventure with you and I am looking forward to our future together. Danke, dass du für mich da bist.

Martijn
Enschede, maart 2021

¹Het verbaast me overigens nog steeds dat er voor zover ik weet nog nooit een “Clash” is ontstaan om het claimen van de naam “energie-groep”, aangezien min of meer iedereen in de CAES-groep zich bezighoudt met energie-efficiëntie...

CONTENTS

xv

1	INTRODUCTION	1
1.1	Energy and the need for energy management	1
1.2	Examples of energy management problems	3
1.2.1	<i>Micro-grids</i>	3
1.2.2	<i>Vehicle routing</i>	4
1.2.3	<i>Information and communication technology</i>	5
1.3	Problem statement and approach	5
1.4	Contributions and outline	9
2	OVERVIEW OF RESOURCE ALLOCATION PROBLEMS	15
2.1	Introduction	15
2.2	Problem formulation and preliminaries	16
2.2.1	<i>Constraints structures and cost functions</i>	16
2.2.2	<i>Notation and definitions</i>	17
2.2.3	<i>Problem classification</i>	18
2.3	Algorithms and complexity results	20
2.3.1	$\alpha/\text{Box}/\gamma$: Optimization over box constraints	20
2.3.2	$\alpha/\text{GBC}/\gamma$: Optimization over generalized bound constraints	23
2.3.3	$\alpha/\text{NC}/\gamma$: Optimization over nested constraints	23
2.3.4	$\alpha/\text{LC}/\gamma$: Optimization over laminar constraints	25
2.3.5	$\alpha/\text{SC}/\gamma$: Optimization over submodular constraints	26
2.4	Reduction of (a, b, f) -separable RAPs to quadratic RAPs	28
2.4.1	<i>Reduction results in the literature</i>	28
2.4.2	<i>The reduction result</i>	29
2.5	Applications	31
2.5.1	<i>Power allocation in multi-channel communication systems</i>	32
2.5.2	<i>Storage operation in energy systems</i>	33
2.5.3	<i>Stratified sampling</i>	35
2.5.4	<i>Vessel speed optimization</i>	35
2.5.5	<i>Speed scaling</i>	37
2.6	Conclusions and outlook	38

2.7	Appendix: Proof that laminar constraints are a special case of submodular constraints	39
2.8	Appendix: Proof that Condition 2.1 holds for the RAP with laminar constraints	40
3	QUADRATIC RESOURCE ALLOCATION	47
3.1	Introduction	47
3.2	Problem formulation, analysis, and general solution approach	48
3.3	Two breakpoint search algorithms	51
3.3.1	<i>An $O(n \log n)$ time algorithm based on sequential breakpoint search</i>	51
3.3.2	<i>An $O(n)$ time algorithm based on binary breakpoint search</i>	53
3.4	Application to electric vehicle charging	57
3.5	Outlook	58
4	QUADRATIC RESOURCE ALLOCATION WITH NESTED CONSTRAINTS	61
4.1	Introduction	61
4.2	Problem formulation and an initial sequential algorithm	64
4.3	A fast $O(n \log n)$ time algorithm for Problem QRAP-NC	67
4.3.1	<i>Notation</i>	68
4.3.2	<i>Computing the optimal Lagrange multipliers of the subproblems</i>	69
4.3.3	<i>Recovering the optimal solution to Problem QRAP-NC</i>	73
4.3.4	<i>An $O(n \log n)$ time algorithm</i>	75
4.4	Evaluation	81
4.4.1	<i>Problem instances</i>	81
4.4.2	<i>Implementation details</i>	82
4.4.3	<i>Results and discussion</i>	83
4.5	Conclusions	88
5	NONSEPARABLE QUADRATIC RESOURCE ALLOCATION WITH GENERALIZED BOUND CONSTRAINTS	91
5.1	Introduction	91
5.2	Problem formulation and motivation	93
5.2.1	<i>Problem formulation and background</i>	93
5.2.2	<i>Load balancing in three-phase electricity networks</i>	94
5.2.3	<i>Modeling the three-phase EV charging problem</i>	95
5.3	Analysis	97
5.3.1	<i>Convex instances of Problem QRA-NonSep-GBC</i>	97

5.3.2	Constraint elimination	98
5.3.3	Monotonicity of optimal solutions	100
5.4	Solution approach	102
5.4.1	Outline	102
5.4.2	Computational aspects	104
5.5	Two algorithms for Problem QRA-NonSep-GBC	108
5.5.1	An $O(n \log n)$ time algorithm based on sequential breakpoint search	109
5.5.2	An $O(n \log n)$ time algorithm based on binary breakpoint search	111
5.5.3	Complexity results for special cases and related problems	115
5.6	Evaluation	117
5.6.1	Problem instances	117
5.6.2	Implementation details	118
5.6.3	Results	119
5.7	Concluding remarks	123
5.8	Appendix: Formulation of Problem EV-3Phase	126
5.9	Appendix: Average execution times of Algorithms 5.2 and 5.3 and MOSEK	127
6	ONLINE DUALITY-DRIVEN OPTIMIZATION	133
6.1	Introduction	133
6.2	Background	135
6.2.1	Existing paradigms for optimization under uncertainty	135
6.2.2	Multiplier prediction	137
6.2.3	Comparison to ODDO	138
6.3	The ODDO framework	139
6.3.1	Problem formulation	139
6.3.2	Lagrangian duality revisited	140
6.3.3	Solution approach	142
6.3.4	An illustrative example	144
6.4	Motivation and robustness of ODDO	146
6.4.1	Predictive value of the optimal Lagrange multipliers	146
6.4.2	Robustness of ODDO against prediction errors	148
6.5	Evaluation	154
6.5.1	Battery scheduling	154
6.5.2	Inventory management	156
6.5.3	Setup of the evaluation	157
6.5.4	Results	159
6.6	Conclusions and outlook	163
6.7	Appendix: Computing the projection sets $\mathcal{C}_{\text{proj}}^t$ for Problem BATTERY	164

7	MULTIPLIER PREDICTION FOR ONLINE DUALITY-DRIVEN OPTI-	167
	MIZATION	
7.1	Introduction	167
7.2	Two prediction approaches	169
7.2.1	<i>The online EV charging problem</i>	169
7.2.2	<i>Incorporating a preference for under- or over-prediction</i>	170
7.2.3	<i>Predictions based on the problem structure</i>	172
7.3	Simulation study	173
7.3.1	<i>Setup and scenarios</i>	174
7.3.2	<i>Results and discussion</i>	174
7.4	Conclusions and discussion	178
8	EXTENSIONS OF ONLINE DUALITY-DRIVEN OPTIMIZATION	183
8.1	Introduction	183
8.2	Piece-wise linear cost functions	185
8.2.1	<i>Motivation and problem formulation</i>	185
8.2.2	<i>Offline optimization approach</i>	188
8.2.3	<i>Online optimization approach</i>	190
8.2.4	<i>Evaluation</i>	191
8.3	Minimum charging threshold	193
8.3.1	<i>Problem formulation</i>	196
8.3.2	<i>Offline optimization approach</i>	197
8.3.3	<i>Online optimization approach</i>	200
8.3.4	<i>Evaluation</i>	204
8.4	Conclusions and outlook	206
9	CONCLUSION	209
9.1	Summary	209
9.2	Conclusions	211
9.3	Recommendations for future research	213
A	TECHNICAL PROOFS	217
A.1	Proofs of Chapter 4	217
A.1.1	<i>Proof of Lemma 4.1</i>	217
A.1.2	<i>Proof of Lemma 4.4</i>	220
A.1.3	<i>Proof of Lemma 4.5</i>	222
A.2	Proofs of Chapter 5	223
A.2.1	<i>Proof of Lemma 5.1</i>	223

A.2.2	<i>Proof of Lemma 5.2</i>	224
A.2.3	<i>Proof of Lemma 5.4</i>	226
A.2.4	<i>Proof of Lemma 5.5</i>	228
A.2.5	<i>Proof of Lemma 5.6</i>	229
A.3	Proofs of Chapter 6	229
A.3.1	<i>Proof of Lemma 6.3</i>	229
A.3.2	<i>Proof of Lemma 6.4</i>	230
 ACRONYMS		 235
 BIBLIOGRAPHY		 237
 LIST OF PUBLICATIONS		 253

1

INTRODUCTION

ABSTRACT – This chapter serves as a general introduction to this thesis. We introduce the general topic of energy management, give several examples of its applications, and discuss the field of research on energy management in general. Based on this, we formulate the main research question that is answered in this thesis and the two corresponding subquestions. Finally, we list the contributions and provide an outline of the remainder of this thesis.

1.1 ENERGY AND THE NEED FOR ENERGY MANAGEMENT

During the last few centuries, the development of society has been fueled by energy, both literally and figuratively. Since the invention and widespread use of the steam engine in the 18th and 19th centuries, the availability and ongoing development of relatively cheap fuels and new efficient and powerful engines has been the base for many new technological advancements in manufacturing, transportation, and computation. All these developments have led to a society that is completely dependent on a stable and reliable energy system. However, in the last few decades, several weaknesses and negative consequences of this system have come to light. The leading cause of these negative effects is the use of fossil fuels such as coal and gas as the main energy sources within the system. These fuels have been formed within the Earth's crust over the course of millions of years. However, current estimates of these fossil fuel reserves suggest that, at the current energy production and consumption rates, these reserves will be fully depleted within 50 years (oil and natural gas) or 132 years (coal) [1]. Moreover, the large-scale burning of fossil fuels leads to an increased greenhouse effect. The consensus among climate scientists is that this is the primary thriving force behind recent increases in global warming and extreme climate changes (see also [37, 132, 133]).

These insights were the driving force for researchers and governments to search for other energy sources that are plentiful, contribute less to the greenhouse

effect, and cause minimal pollution. This has led to the development of technologies to harvest energy from so-called *renewable* energy sources (RES) such as solar irradiation, wind, and water. Moreover, in the last years, research on reducing energy usage by increasing the energy efficiency of, e.g., computing devices, transportation vehicles, and household appliances, increased significantly. Hereby, the focus was on improving both the devices themselves (e.g., smaller computer chips or lighter materials) and their utilization (e.g., varying processor speeds, traveling through shorter routes, or incorporating power-saving modes).

However, the introduction and integration of RES within energy systems also comes with several weaknesses and challenges. One major weakness of an energy system based on RES concerns the inherent uncertainty of the supply from these sources. More precisely, whereas the supply of fossil fuels for an energy system can be fully controlled by increasing the production at power plants, the supply from RES is less controllable and determined primarily by environmental factors such as the weather. As a consequence, the energy supply within a system based on RES is less reliable than that within a system based on fossil fuels.

The transition to RES has as a side effect that electricity as one of the energy forms gets an increased share in the overall energy usage. This is called the *electrification* of energy production and supply. For instance, solar and wind power are mainly transported in the form of electricity, whereas power from traditional power plants is also transported as gas or via heating networks. Moreover, in the traditional fossil fuel-based system, space heating and transportation required energy in the form of coal, gas, or petrol, whereas in a future RES-based system this energy is mainly required to be electricity due to the extensive use of, e.g., heat pumps and electric vehicles. Therefore, as a consequence of electrification, a huge increase in energy flows within the electricity distribution grid is expected, which may lead to an increased wearing of grid assets such as cables and transformers and can even lead to blackouts [80]. One way to overcome this problem would be to simply increase the capacity of the grid by, e.g., installing more or thicker cables. However, this is an extremely costly operation (see, e.g., [68]), since this increase of capacity is mainly needed for accommodating sporadic large peaks in demand and supply and thereby leads to a lower utilization rate.

With regard to energy efficiency of devices, one way to increase a device's efficiency is to exploit any flexibility that it has in accomplishing its task to reduce its energy consumption. A concrete example of this is the presence of “eco-friendly” settings for many modern white goods: washing laundry at a lower temperature takes more time but requires overall less energy than washing at a higher temperature. Similar trade-offs between, e.g., time and energy usage, can be observed in transportation and computing. Here operating a vehicle or processor at higher speeds reduces the time needed to reach a destination or complete a task but requires disproportionately more energy. Unfortunately, it is not always obvious how exactly this trade-off should be made, partly because this trade-off is

often user-dependent. In particular, it is unlikely that users themselves are able to properly make this trade-off (see, e.g., [140]).

These challenges of RES-based energy systems and the trade-offs regarding energy efficiency suggest that some form of *energy management* is needed to maintain a proper operation of such energy systems and devices. More precisely, in order to ensure that the energy demands are satisfied or certain tasks of devices are achieved, energy flows must be actively managed within the physical boundaries and limitations of the system and the involved device. Furthermore, to be successful, this should be done automatically via algorithms that determine proper energy schedules, given knowledge of the properties and limitations of the energy system and the involved devices and their energy demand and supply requirements. Such schedules specify the energy consumption of (each part of) the system and/or of devices at each moment in time and, thereby, should fulfill as much as possible the given requirements and take into account the preferences of the users.

The focus of this thesis is on these types of energy management problems and algorithms to solve these problems. In this first chapter, a brief introduction to this topic is provided, the studied research questions are stated, and the main contributions of the thesis are summarized. In detail, the remainder of this chapter is organized as follows. First, in Section 1.2, we discuss several concrete examples of energy management problems and applications. Subsequently, in Section 1.3, we state the research questions of this thesis and our approach for answering these questions and in Section 1.4, we summarize the contributions of this thesis. Finally, in Section 1.4, we provide an outline of the remainder of this thesis.

1.2 EXAMPLES OF ENERGY MANAGEMENT PROBLEMS

In the previous section, we stated the need for active energy management in future energy systems and for the involved devices. In this section, we briefly discuss several examples of such systems and devices.

1.2.1 MICRO-GRIDS

Micro-grids are branches of the main distribution grid that aim to be as self-sustainable as possible and to this end aim to minimize the import and export of energy from the main grid. To achieve this, energy flows within a micro-grid are actively managed and the flexibility of devices such as electric vehicles (EVs) is exploited. Hereby, the main grid still functions as a common “buffer” to compensate for moments when the micro-grid is not able to be entirely self-sustainable. Next to reducing the energy exchange between the micro-grid and the main distribution grid, this form of energy management has also other advantages. For instance, by reducing the amount of energy that is transported through the grid

and the distance that this energy is traveling, also transport losses within the system are reduced.

One of the most used paradigms to manage energy flows within micro-grids is called *decentralized energy management* (DEM) [79, 166]. Within this paradigm, the main objective (self-sustainability of the micro-grid) is achieved by influencing the energy usage of each device individually and coordinating this individual steering on the micro-grid level. By distributing the actual optimization over the individual devices instead of optimizing the usage of all devices within the micro-grid as a whole, computational effort can be saved. Moreover, by analyzing properties of the given micro-grid and its devices, one may often be able to derive clever coordination schemes that ensure convergence of the optimization process towards a good (enough) or even the “optimal” energy schedule for the entire micro-grid. Based on this, DEM is seen as a suitable successor of centralized energy management approaches that are based on traditional models and techniques for fossil fuel-based systems such as the unit commitment problems [135].

For a detailed overview of the rise and development of micro-grids, DEM, and energy systems in general, we refer the interested reader to the quite complete overview given in [164, Appendix A].

1.2.2 VEHICLE ROUTING

The transport sector is one of the larger energy users. Truck transport and shipping are estimated to be responsible for around 6% and 10% of the global CO₂ emissions in 2020 respectively [2, 88]. Because of this, vehicle routing and shipping companies have been interested in ways to reduce this negative impact on the environment while maintaining the service quality, i.e., ensuring that delivery deadlines are met. This interest has led to a relatively young research area within the broader field of vehicle routing usually called “green routing” [105].

Traditionally, the goal within vehicle routing problems is to minimize either the (total) travel time or distance of the used vehicles. Hereby, the underlying implicit assumption is that the cost for routing a vehicle is proportional to either the traveled time or distance or both. However, as mentioned before, this is in general not true for the costs of fuel and the environmental impact of driving a vehicle, since the negative force that a vehicle undergoes while driving (e.g., air or rolling resistance) increases quadratically in the vehicle’s speed. As a consequence, the total amount of fuel used to travel a given fixed distance is proportional to the used driving speed. Hence, driving at lower speeds leads to an increase in travel time but also to a reduction of fuel usage and thereby may be an attractive alternative if this increase in travel time is allowable with regard to delivery deadlines.

It is estimated that around 3 to 4 percent of the global CO₂ emissions in 2020 is due to information and communication technology (ICT) and that this share will increase in the coming years [20]. Therefore, significant energy savings can be realized by increasing the efficiency of ICT devices and developing better task management strategies for processors.

Addressing the first aspect is essential for ICT systems whose computational power and energy supply are limited due to, e.g., space or capacity constraints. Prominent examples of these are wireless devices such as smartphones. These devices are required to be small for a proper user experience and are usually powered through a battery contained within the device. Thus, both the computation hardware and the battery must be relatively small. As a consequence, these elements must be highly efficient if the device is to be used for more extended periods of time without intermediate recharging.

Addressing the second aspect (improving task management strategies) is important for systems where the way the system is operated has a (large) influence on the energy consumption. Examples of this are large data centers where the processing of the data produces (a lot of) heat and cooling equipment is installed to prevent the servers from overheating. Within a data center, cooling accounts for around 40% of the total energy usage of the data center (see, e.g., [157]). Therefore, modern energy-efficient data centers apply task scheduling techniques whereby, e.g., tasks are distributed over multiple servers and any flexibility in the deadlines of tasks is exploited to reduce the operating speed of the servers. The goal of these actions is to achieve a reduction in the energy used for computing and the temperature of the servers, which in turn reduces the necessity for extensive cooling (see, e.g., [184]).

1.3 PROBLEM STATEMENT AND APPROACH

Solving a given problem in the natural and exact sciences is almost always started by formulating a (mathematical) model of the studied phenomenon and analyzing the properties of this model. Subsequently, an algorithm is designed to compute a solution to the model (or “solve the model”). Generally, there is a large variance of possible models for a given phenomenon and many ways to design an algorithm for a given model. The eventual choice for a concrete model and/or algorithm is determined to a large extend by the objective of the modeler and/or designer.

Scientific models for the optimization of real-life systems, such as the optimization of energy schedules in energy management, are generally constructed from either of two perspectives. The first perspective is a practical and pragmatic one, where one encounters a problem in real-life and the main goal is to find a simple solution to (re)solve this problem. For the resulting approaches, it is generally

important that they work well in practice, but usually little or no justification of the chosen approach nor an analysis of the behavior is/can be given. In particular, often no information on the (expected) performance of the approach in terms of, e.g., efficiency or goodness of the solution is available. As a consequence, the reliability of the approach cannot always be fully guaranteed. The second perspective is a theoretical and formal one, which is mainly employed within research fields such as theoretical computer science, scheduling, or discrete mathematics. In this perspective, the goal when studying a given problem or problem class is often not primarily to solve an actual problem in real life, but rather to investigate the computational boundaries of the considered problem (class). The resulting approaches usually have well-described performance guarantees in terms of efficiency (e.g., time complexity) and/or solution quality (e.g., approximation ratios) that scale well with the size of the considered problem. However, this can often be done only after simplifying the problem to the extent that, in an extreme case, the solutions produced by the resulting algorithms cannot be used and/or implemented anymore in practice to solve the actual real-life problem. Additionally, often little attention is paid to the practical performance of such algorithms. In the best case, this is because practical performance was simply not the (main) focus of the research. However, in the worst case, the developed algorithms are made so complex (in order to achieve certain worst-case complexities) that they are no longer suitable for implementation in practical settings, for instance as part of larger software tools.

The difference between the mentioned practical and theoretical research perspective is in particular visible in energy management research. Here, a large part of research is initiated from the fields of theoretical computer science and mathematical optimization. The used models in this field are often rather simplistic and far away from reality, not taking into account practical limitations of devices and/or energy systems. Also, the focus is often on solution approaches and algorithms with a provable low computational complexity, i.e., algorithms whose number of required operations scales well in the input size of the problem. However, this type of complexity does not say anything about the actual number of required operations, which is a major determinant of the overall execution time of an implemented algorithm.

Ideally, one would like to combine the benefits of both perspectives, namely to obtain algorithms that are fast in practice *and* have some kind of (theoretical) performance guarantee. Recently, a new research area has emerged within (theoretical) computer science called *algorithm engineering* that aims to address exactly this issue (see also [125]). More precisely, in this field, the goal is to develop algorithms for real problems that both have nice theoretical performance guarantees and are suitable for practical usage and implementation within a given application.

This thesis aims to contribute towards the area of energy management by approaching it from the field of algorithm engineering. More precisely, with this thesis, we aim to answer the following research question:

How can we develop algorithms for energy management problems that are efficient and give good solutions both in theory and in practice?

The approach for answering this question is to focus on two different perspectives on energy management problems. The first perspective is that of resource allocation, meaning that we view energy management problems as problems where a given amount of resource (energy) must be divided over a set of activities (devices, time periods) such that the cost of this allocation is minimized and potential additional restrictions on the allocated amounts are satisfied. This more general problem type is known in the literature as the *resource allocation problem* (RAP) and has been studied extensively since the 1950s within many different fields such as operations research, finance, and telecommunications [137]. Since then, many efficient algorithms have been developed for this problem and many of its variants and extensions (see, e.g., [85, 94, 138]). Practical limitations and requirements of devices within energy systems lead to several new variants of the RAP. For these types of problems, we develop efficient and fast solution algorithms.

The second perspective is that of so-called *duality theory*. Duality is a common concept in several areas of (applied) mathematics [10] and several types of duality theory have been developed. In this thesis, we utilize so-called *Lagrangian* duality (see, e.g., [29]), where the dual perspective leads to an alternative formulation of an optimization problem in terms of penalties (called *Lagrange multipliers*) on constraint violations. More precisely, the dual of an optimization problem is the problem of finding values for these penalties such that the best possible solution for the initial problem can be obtained by finding the solution that incurs the least combined cost. Under several mild and practically justified assumptions on the problem at hand, these Lagrange multipliers completely characterize an optimal solution to the problem. This means that an optimal solution can be deduced relatively easily from these (optimal) Lagrange multipliers.

We argue that the Lagrangian dual perspective is particularly suitable to solve energy management problems. The first reason is that Lagrangian duality theory has been used to solve the RAP since the earliest studies on the problem in the 1950s [137]. Henceforth, it has served as an important tool to solve several variants of this problem, in particular the RAP with so-called *submodular* constraints [12, 49, 75] that can be used to model a wide range of energy management problems. The second reason is that Lagrange multipliers within energy management problems often have a concrete and useful interpretation. Examples of this are energy management problems involving EVs and storage systems in microgrids. A commonly desired goal with regard to the operation of these devices is that they use energy such that the combined energy consumption of the device and, e.g., a household with solar panels, is flattened as much as possible. In this

setting, the Lagrange multipliers can be interpreted as the level of “flatness” of the combined profile. This interpretation of the Lagrange multiplier has been used successfully in, e.g., [34, 59, 122] to solve a range of scheduling problems for EVs.

With these two perspectives in mind, we formulate the following two subquestions of the main research question. The first subquestion is concerned with the integration of physical properties and requirements into the rather theoretical models of RAPs:

- » How can we incorporate physical properties of devices in models and algorithms for energy management problems while maintaining good theoretical performance guarantees for corresponding solution approaches?

To answer this subquestion, we study several energy management problems arising in RES-based energy systems and other applications such as processor scheduling and vessel speed optimization. For these problems, we develop algorithms that compute an optimal solution to the problem. All these algorithms explicitly exploit the (Lagrangian) dual perspective by first computing the optimal Lagrange multipliers and, subsequently, derive from these values the optimal (primal) solution to the problem. When developing these algorithms, we adequately address both the theoretical efficiency and practical speed of the algorithm. In detail, we derive theoretical performance guarantees with regard to the scalability of the algorithm (the worst-case time complexity) and assess the practical speed of the algorithm via numerical evaluations. To obtain a deeper insight into the practical behavior of the algorithms, we evaluate them on both synthetic and realistic instances of the corresponding energy management problem. Thereby, we are able to validate the quality of the theoretical efficiency bounds and the usability of the algorithms within existing software for simulating and operating energy systems and devices.

The second subquestion is concerned with the dependency of proper energy management solutions on predictions of future energy demand and production:

- » How can we obtain good solutions to energy management problems that do not require detailed predictions of uncertain data such as future energy demand and supply?

To answer this subquestion, we propose a novel concept to solve optimization problems where uncertain data that is only revealed over time is present. The core idea of this concept is that we move away from predicting the actual data itself and instead focus on predicting the optimal Lagrange multipliers of the problem at hand. This implies that we transform the problem of uncertainty in the data into the problem of uncertainty in the optimal Lagrange multipliers. This shift in focus leads to several advantages since the Lagrange multipliers encode information that is directly concerned with the optimal solution of the optimization problem, whereas the original uncertain problem data often does not

provide such information. For instance, Lagrange multipliers directly represent useful concrete information on an optimal solution in terms of, e.g., redundant constraints, which could in principle have been omitted, or tight constraints, which indicate the limitations of the given problem with regard to achieving better solutions.

In order to solve optimization problems using predictions of the optimal Lagrange multipliers as input, we develop a new framework that we call “Online Duality-Driven Optimization” (ODDO). The term “online” represents the nature of the arrival of the uncertain data, i.e., the data is revealed gradually in stages as time progresses and after each stage a decision on the optimization variables corresponding to this stage must be made. We stress that, as opposed to other existing approaches for optimization under data uncertainty (see also Section 6.2), this framework does not require any additional predictions of the future original data as input. Moreover, when applied to a certain broad class of RAPs, the framework provides a worst-case performance guarantee on the difference in objective value between the online and optimal offline solution. To also assess the practical performance of the ODDO framework, we apply the framework to several energy management problems and demonstrate the effectiveness of several simple strategies for predicting the optimal Lagrange multipliers. Finally, we also study the possible extension of the framework towards problems wherein the concept of Lagrangian duality either cannot be applied or leads to different characterizations of optimal solutions.

1.4 CONTRIBUTIONS AND OUTLINE

In the following, we summarize the contributions and give an outline of the remainder of this thesis.

In Chapter 2, we provide an overview of RAPs. We discuss several well-studied variants and extensions of the problem in the literature and give an overview of known algorithms for these problems. Furthermore, we show that several objectives used for RAPs are in a sense equivalent. This means that for a given instance of a given RAP, there often exists a solution to this instance that is optimal for not just one objective but for a whole range of different objectives. Finally, we present several concrete applications where some of the core problems can be formulated as RAPs.

In Chapter 3, we discuss one of the most basic RAPs, namely the RAP with quadratic cost functions and additional lower and upper bounds on the allowed allocation of resource to activities. We choose to dedicate a separate chapter to this problem since it is an important building block in many of the problems discussed in later chapters. We analyze the structural properties of this problem and discuss a well-established solution approach to solve it. Moreover, we discuss two efficient algorithms based on this approach that also form an important

building block of solution approaches and algorithms for the problems discussed later in the thesis.

In Chapter 4, we focus on the RAP with quadratic cost functions and additional nested constraints. Applications of this problem in energy management include energy storage scheduling, vessel speed optimization, and speed scaling (see also Section 2.5). Several theoretically efficient algorithms exist to solve this problem. However, we argue that the currently most efficient algorithm might not be very fast in practice due to the required complex data structures and repeated initialization of parameters for its subroutines. Therefore, we derive an alternative solution approach and algorithm for this problem that has the same theoretical worst-case time complexity, but requires only simple data structures and saves computational effort by minimizing the number of parameter values that need to be initialized. Computational experiments confirm that our algorithm outperforms the aforementioned currently most efficient algorithm in practice by one order of magnitude. Moreover, numerical comparisons with an algorithm implemented within an existing simulation tool for energy management research demonstrate the potential of our approach for its application in real energy management systems.

In Chapter 5, we consider a RAP where the allocation costs depend not only on the allocations to individual activities, but also on the allocations to subsets of the activities. This leads to a so-called *non-separable* RAP, which has hardly been studied in the literature. A direct application of this problem is the scheduling of EV charging via three-phase chargers. The latter is an increasingly emerging technology for residential EV charging that is considered to be crucial for the proper integration of EVs in residential areas. We develop a theoretically efficient solution approach and algorithm for this problem and show via computational experiments that this algorithm is also fast enough in practice for usage in real energy management systems.

In Chapter 6, we introduce the online duality-driven optimization (ODDO) framework for solving online optimization problems mentioned in the previous section. As already stated, this framework requires as input predictions of only the optimal Lagrange multipliers of the given optimization problems. Advantages of this framework, when compared to existing frameworks for optimization problems involving uncertain data, include

1. that it does not require any quantitative knowledge on the original uncertain data,
2. that the problems that must eventually be solved within this framework are not computationally harder than the original problem with deterministic data and can thus be solved using the same optimization methodology, and
3. that it is relatively easy to use and apply in practice since it is based only on basic results from the field of convex optimization.

Furthermore, for the special case of RAPs with increasing cost functions and sub-modular constraints, we derive a theoretical performance guarantee in the form of a bound on the difference between the online and optimal offline solution, provided that the optimal Lagrange multipliers are ‘under-predicted’. Moreover, we apply the framework to both an energy management problem (scheduling the operation of a storage system) and a problem from outside energy management (inventory management of a warehouse). In computational experiments, we compare for these applications the performance of ODDO to that of an alternative optimization approach where we explicitly use predictions of the original uncertain data as input. The results of these experiments indicate that ODDO is able to outperform this approach, which indicates that the optimal Lagrange multipliers either are easier to predict than the uncertain data, provide more knowledge on the actual optimal solution, or both. This serves as a promising starting point for usage of ODDO in applications.

In Chapter 7, we continue our research on the ODDO framework by focusing in more detail on the actual prediction of the optimal Lagrange multipliers. More precisely, we present two different approaches for making these predictions. The first approach is based on incorporating a specific preference for the difference between the predicted and the optimal Lagrange multiplier. The second approach is based on the specific problem structure of especially RAPs and exploits a given relation between the uncertain data and the optimal Lagrange multipliers. As a proof of concept, we apply both approaches within the ODDO framework to the specific energy management problem of scheduling the charging of an EV within a household. The resulting numerical results indicate that both approaches are able to compute predictions that lead to online solutions with near-optimal objective value, which demonstrates the potential of these approaches for their general application within the ODDO framework.

In Chapter 8, we study possible extensions of the ODDO framework to optimization problems that do not fully satisfy the necessary assumptions for deriving this framework and several of its performance guarantees. We do this by studying two energy management problems in the context of EV charging that actively incorporate several practical limitations of EV batteries and the charging infrastructure. For these problems, we present and discuss algorithms to solve their deterministic versions. Moreover, we show how the original ODDO framework can be adjusted to solve these problems when some of the data is uncertain. Additional simulation studies confirm that these adjusted ODDO frameworks are able to compute near-optimal online solutions, similarly to the original ODDO framework. This shows the potential of the ODDO framework for successfully handling practical limitations of devices and energy systems.

Finally, in Chapter 9, we summarize the results of this thesis and answer the research question and the two subquestions. Furthermore, we state our recommendations for relevant and interesting future research on the considered problems but also for energy management in general.

We conclude this chapter with three general remarks regarding this thesis. First, all simulations and computations in this thesis have been implemented in Python (version 3.5) and executed on a 2.60 GHz Dell Inspiron 15 with an Intel Core i7-6700HQ CPU and 16 GB of RAM, unless stated otherwise. Second, to guide the reader through this thesis, especially when the reader wishes to read only about specific topics discussed in only some of the chapters, Figure 1.1 depicts the reading dependencies between the chapters. Finally, I hope that the reader finds this thesis informative and wish them much enjoyment while reading!

Figure 1.1: Reading dependencies between the chapters of this thesis. Solid arrows represent dependencies for the entire content of a chapter, whereas dashed arrows represent dependencies for only those parts regarding applications and evaluations.

2

OVERVIEW OF RESOURCE ALLOCATION PROBLEMS

ABSTRACT – Energy management problems can often be interpreted as resource allocation problems (RAPs). In these problems, the goal is to divide a given amount of resource over a set of activities while minimizing the cost of this allocation and possibly satisfying constraints on allocations to subsets of the activities. This chapter provides an introduction to RAPs and discusses several relevant special cases of these problems. In particular, we focus on the cost function and show that a particular class of cost functions is an equivalence class in the sense that, given an instance of the RAP, there exists a solution to the RAP that minimizes all cost functions in this class simultaneously. Furthermore, we discuss several applications of the RAP to energy management problems and the implications of the aforementioned equivalence result for these problems.

2.1 INTRODUCTION

The resource allocation problem (RAP) is a classical problem within operations research and has been studied extensively and continuously since the 1950s [137]. In its most basic and most studied form, this problem asks for the allocation of a given amount of resource over a set of activities while minimizing a given cost function. Over the years, several variations and extensions of this basic setting have been studied, with specific types of individual cost functions, additional constraints, and allocation restrictions such as integer-valued allocations [94].

The goal of this chapter is threefold. First, we aim to provide an overview of the most relevant RAPs in the literature and corresponding algorithms and their worst-case time complexities. Second, we show that a particular class of objective

This chapter is based on and contains parts of [SU:1].

functions is equivalent, i.e., given a constraint structure of a RAP, there exists a solution that minimizes all objective functions within this class simultaneously. Finally, we model several energy management problems as RAPs and show the impact of the aforementioned equivalence result on these problems.

On a higher level and perhaps of independent interest, this chapter aims to demonstrate that methodological research on RAPs is conducted independently in many different research fields, often under different names. As a consequence, many conceptual insights, structural properties, and solution approaches for RAPs have been re-invented and re-discovered many times over the years, both within the same field and independently in several fields. Therefore, we aim to promote a cross-disciplinary approach to studying RAPs. Such an approach will both reduce the number of future re-discoveries and re-inventions and allow researchers to benefit from the many available different perspectives on RAPs.

The organization of the remainder of this chapter is as follows. In Section 2.2, we formally describe different variants of the RAP that we study in this chapter. Subsequently, in Section 2.3, we provide an overview of existing solution approaches and algorithms for each of these variants. In Section 2.4, we show the equivalence result mentioned above. Section 2.5 presents several applications of the RAP to energy management problems and discusses the implications of the equivalence result for these problems. Finally, Section 2.6 provides some concluding remarks.

2.2 PROBLEM FORMULATION AND PRELIMINARIES

In this section, we formulate the studied RAPs and introduce the used notation and definitions.

2.2.1 CONSTRAINTS STRUCTURES AND COST FUNCTIONS

The basic version of the RAP calls for an allocation $x \in \mathbb{R}^n$ of a given amount $R \in \mathbb{R}$ of resource to a set of activities $\mathcal{N}$ such that a given cost function $\Phi(x)$ of the allocation is minimized. This problem can be formulated as follows:

$$\begin{aligned} \text{RAP : } & \min_x \Phi(x) \\ & \text{s.t. } \sum_{i \in \mathcal{N}} x_i = R. \end{aligned}$$

Based on this basic version, we can formulate several variants of the RAP with specific types of cost functions, additional constraints, and different types of decision variables.

Concerning the objective, most of the existing research on RAPs consider a *separable* cost function, i.e., each activity has its own cost function ϕ_i and the objective is to minimize the sum of these individual functions. State-of-the-art

solution approaches for RAPs generally allow each activity to have its own arbitrary (convex) cost function. Although this is an interesting aspect from a mathematical point of view, it is questionable whether this is a given situation for practical problems. In applications, often the structure of the cost functions is the same for all activities (e.g., all functions are quadratic) and the difference lies primarily in different parameters for these functions (e.g., each function has different multiplicative factors). In fact, this is the case for the large majority of applications studied in (the works surveyed in) [5, 137, 138, 172].

This motivates us to study in particular RAPs where the cost functions share a common structure and differ only in the parameter choice. More precisely, we introduce the notion of (a, b, f) -separable RAPs, wherein the cost function of each activity i is of the form $a_i f(\frac{x_i}{a_i} + b_i)$, where f is the shared convex function and $a_i > 0$ and b_i are given parameters that can be different for different activities i . Such cost functions are an extension of so-called d -separable functions introduced in [170]. Moreover, they are closely related to the concept of perspective functions [35, 36], which arise naturally in many problems in applied mathematics. In particular, most of the applications in (works surveyed in) [5, 137, 138, 172] can be modeled as (a, b, f) -separable RAPs.

With regard to the constraint structure, we focus on a general version of the RAP that occurs widely in applications, namely the RAP with additional submodular constraints (see, e.g., [49, 64]). In this problem, for each subset of the activities, there is an upper bound on the total amount of resource allocated to these activities and this bound is given by a submodular set function (see also Section 2.2.2). Submodular functions have the so-called diminishing-returns property, meaning that the marginal function value, i.e., the function value divided by the number of elements in the input set, does not increase when the input set is extended. Because of this property, the RAP with submodular constraints has many applications in, e.g., machine learning [12, 13], scheduling [107, 154], and game theory [66, 71, 90]. Moreover, important special cases of this problem are the RAP with box constraints (see [137]), the RAP with generalized bound constraints (see [74] and Chapter 5), and the RAP with nested constraints (see [172]). Important application areas for in particular these special cases include regularized learning [40, 112], telecommunications and energy management [136, 167, 179], and statistics [46, 130] (see also the overviews in [137] and [172]).

2.2.2 NOTATION AND DEFINITIONS

In the following, we introduce some notation and properties of the used functions and sets. For this, let $\mathcal{N} := \{1, \dots, n\}$ be the index set of the given activities. We call a convex function $\Phi : \mathbb{R}^n \rightarrow \mathbb{R}$ *separable* if it can be written as the sum of single-variable convex functions, i.e., if $\Phi(x) = \sum_{i \in \mathcal{N}} \phi_i(x_i)$ for some single-variable convex functions $\phi_i : \mathbb{R} \rightarrow \mathbb{R}, i \in \mathcal{N}$. Moreover, given two vectors $a \in \mathbb{R}_{>0}^n$ and $b \in \mathbb{R}^n$ and a single-variable convex function $f : \mathbb{R} \rightarrow \mathbb{R}$, we say that a separable function Φ is (a, b, f) -*separable* if each function ϕ_i can be

written as

$$\phi_i(x_i) = a_i f\left(\frac{x_i}{a_i} + b_i\right).$$

Note that we do not pose any restrictions on f other than convexity. Hence, both f and ϕ_i are not necessarily *strictly* convex or differentiable. We denote the left and right derivatives of f by f^- and f^+ respectively. It follows that the left derivative ϕ_i^- of ϕ_i is given by

$$\begin{aligned}\phi_i^-(x_i) &:= \lim_{y \uparrow x_i} \frac{\phi_i(y) - \phi_i(x_i)}{y - x_i} = \lim_{y \uparrow x_i} \frac{a_i f\left(\frac{y}{a_i} + b_i\right) - a_i f\left(\frac{x_i}{a_i} + b_i\right)}{y - x_i} \\ &= f^-\left(\frac{x_i}{a_i} + b_i\right).\end{aligned}$$

Analogously, the right derivative ϕ_i^+ of ϕ_i is given by $\phi_i^+(x_i) = f^+\left(\frac{x_i}{a_i} + b_i\right)$. Throughout this thesis, we call a resource allocation problem (a, b, f) -separable if its objective function is (a, b, f) -separable.

We denote the vector of ones of dimension n by $\bar{e}$. Furthermore, for each index $i \in \mathcal{N}$, we denote by $e^i \in \mathbb{R}^n$ the standard basis vector associated with i , i.e., the vector whose i^{th} entry is 1 and whose other entries are all zero. For a given set $\mathcal{C} \subseteq \mathbb{R}^n$ and $x \in \mathcal{C}$, let $\mathcal{E}_{\mathcal{C}}(x)$ denote the set of index pairs $(i, k) \in \mathcal{N}^2$ for which we can shift some amount from x_i to x_k without violating feasibility. More precisely, we define

$$\mathcal{E}_{\mathcal{C}}(x) := \{(i, k) \in \mathcal{N}^2 \mid x + \epsilon(e^k - e^i) \in \mathcal{C} \text{ for some } \epsilon > 0\}.$$

Each pair in $(i, k) \in \mathcal{E}_{\mathcal{C}}(x)$ is called an *exchangeable pair*.

Finally, we define the concept of *submodular set functions*. Let $r : 2^{\mathcal{N}} \rightarrow \mathbb{R}$ be a set function on the index set $\mathcal{N}$. We say that r is *submodular* if $r(\mathcal{X} \cup \mathcal{Y}) + r(\mathcal{X} \cap \mathcal{Y}) \leq r(\mathcal{X}) + r(\mathcal{Y})$ for any $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{N}$, where we assume that $r(\emptyset) = 0$. An alternative definition in terms of diminishing returns is that r is said to be submodular if we have for any $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{N}$ with $\mathcal{X} \subseteq \mathcal{Y}$ and every $i \in \mathcal{N} \setminus \mathcal{Y}$ that $r(\mathcal{Y} \cup \{i\}) - r(\mathcal{Y}) \leq r(\mathcal{X} \cup \{i\}) - r(\mathcal{X})$.

2.2.3 PROBLEM CLASSIFICATION

In this section, we introduce all the variants of the basic RAP, which was introduced in Section 2.2.1, that we discuss in this chapter. To clearly distinguish between these problems, we adopt a classification scheme similar to that in [85] and [94], i.e., we specify each problem variation by three fields $\alpha/\beta/\gamma$, where α specifies the objective function, β describes the constraint set, and γ specifies the nature of the decision variables.

For α , we consider the following options:

1. *Separable* (S): Φ is separable.
2. (a, b, f) -*separable* $((a, b, f)$ -S): Φ is (a, b, f) -separable.
3. *Quadratic* $((a, b)$ -Q): Φ is (a, b, f) -separable using $f(y) = \frac{1}{2}y^2$. This means that Φ is both separable and quadratic.

For β , we consider the follows special constraint structures, where we use $\mathcal{M} := \{1, \dots, m\}$ as an index set for additional constraints:

1. *Box constraints* (Box): $l_i \leq x_i \leq u_i$ for all $i \in \mathcal{N}$.
2. *Generalized bound constraints* (GBC): Next to the box constraints also constraints of the form $L_j \leq \sum_{i \in \mathcal{N}_j} x_i \leq U_j, j \in \mathcal{M}$ are given, where the sets $\mathcal{N}_1, \dots, \mathcal{N}_m$ form a partition of $\mathcal{N}$.
3. *Nested constraints* (NC): Next to the box constraints also constraints of the form $L_j \leq \sum_{i \in \mathcal{N}_j} x_i \leq U_j, j \in \mathcal{M}$ are given, where the sets $\mathcal{N}_1, \dots, \mathcal{N}_m$ are such that $\mathcal{N}_1 \subset \dots \subset \mathcal{N}_m \subset \mathcal{N}$.
4. *Laminar (or tree) constraints* (LC): Constraints of the form $L_j \leq \sum_{i \in \mathcal{N}_j} x_i \leq U_j, j \in \mathcal{M}$ are given, where the subsets $\mathcal{N}_1, \dots, \mathcal{N}_m$ of $\mathcal{N}$ have the following property: if $\mathcal{N}_j \cap \mathcal{N}_\ell \neq \emptyset$, then either $\mathcal{N}_j \subset \mathcal{N}_\ell$ or $\mathcal{N}_j \supset \mathcal{N}_\ell$ for all $j, \ell \in \mathcal{M}$.
5. *Submodular constraints* (SC): Constraints of the form $\sum_{i \in \mathcal{S}} x_i \leq r(\mathcal{S}), \mathcal{S} \subset \mathcal{N}$ and $\sum_{i \in \mathcal{N}} x_i = r(\mathcal{N})$ are given, where r is a given submodular function with $r(\mathcal{N}) = R$.

Note that the constraint structures Box, GBC, and NC are special cases of the structure LC. Moreover, it can be shown that the structure LC is a special case of the structure SC (see Appendix 2.7). Thus, all constraint structures are special cases of SC. We discuss each of these special cases in more detail in Section 2.3.

For γ we consider the following two cases:

1. *Continuous decision variables* (C): $x \in \mathbb{R}^n$.
2. *Integer decision variables* (I): $x \in \mathbb{Z}^n$. In this case, we assume without loss of generality that the chosen convex objective function is replaced by its piece-wise linear approximation with breakpoints at the integers.

Table 2.1 summarizes the possible entries of α , β , and γ as a compact reference. To simplify the presentation, we assume that whenever we specify β , the parameters that define the corresponding constraints are fixed. For example, when we consider the problems (a, b, f) -S/LC/C and (a, b) -Q/LC/C for some vectors a, b and convex function f , we assume that the subsets $\mathcal{N}_1, \dots, \mathcal{N}_m$ and vectors $L := (L_j)_{j \in \mathcal{M}}$ and $U := (U_j)_{j \in \mathcal{M}}$ are fixed.

Finally, apart from these variants of the RAP, we study a general constraint class where a constraint $x \in \mathcal{C}$ for some set $\mathcal{C} \subseteq \mathbb{R}^n$ is given. We pose no restrictions on this set other than that it is nonempty. We denote this constraint class by $\mathcal{C}$

Field	Entry	Meaning
α	S	Separable objective function
	(a, b, f) -S	(a, b, f) -separable objective function
	(a, b) -Q	(a, b, f) -separable objective function with $f(x_i) = \frac{1}{2}x_i^2$
β	Box	Bounds on individual variables
	GBC	Bounds on individual variables and disjoint sums of variables
	NC	Bounds on individual variables and nested sums of variables
	LC	Bounds on laminar sums of variables
	SC	Bounds on sums of variables given by a submodular function
γ	C	Continuous variables ($x \in \mathbb{R}^n$)
	I	Integer variables ($x \in \mathbb{Z}^n$)

Table 2.1: Overview of entries for the problem classification $\alpha/\beta/\gamma$.

and denote the corresponding separable, (a, b, f) -separable, and (a, b) -quadratic versions of this problem by S/C , (a, b, f) - S/C , and (a, b) - Q/C respectively.

In the following section, we focus in more detail on the special cases of $\alpha/SC/\gamma$.

2.3 ALGORITHMS AND COMPLEXITY RESULTS

During the last few decades, many different solution approaches and algorithms have been developed for all kinds of RAPs, in particular for the RAPs that fall within the problem classification presented in the previous section. The purpose of the current section is to give a brief overview of these approaches and algorithms. In particular, we focus on algorithms and complexity results for the quadratic versions of these problems.

As a compact reference, Tables 2.2 and 2.3 summarize the complexity results discussed in this section.

2.3.1 $\alpha/\text{Box}/\gamma$: OPTIMIZATION OVER BOX CONSTRAINTS

In this subsection, we focus on the RAP with box constraints. The general version of this problem, $S/\text{Box}/\gamma$, can be formulated as follows:

$$\begin{aligned}
 S/\text{Box}/\gamma : \min_x \quad & \sum_{i \in \mathcal{N}} \phi_i(x_i) \\
 \text{s.t.} \quad & \sum_{i \in \mathcal{N}} x_i = R,
 \end{aligned} \tag{2.1}$$

$$l_i \leq x_i \leq u_i, \quad i \in \mathcal{N}, \tag{2.2}$$

$$x \in \begin{cases} \mathbb{R}^n & \text{if } \gamma = C, \\ \mathbb{Z}^n & \text{if } \gamma = I. \end{cases}$$

β	γ	
	C	I
Box	$O(n)$	$O(n)$
GBC	$O(n)$	$O(n)$
NC	$O(n \log m)$	$O(n \log m)$
LC	$O(n^2), O(n \log n)$	$O(n^2)$
SC	(only upper constraints)	
	$O(n^2 + n\tilde{E})$	$O(n^2 F \log r(\mathcal{N}) + n\tilde{F})$
	(decomposition), $O(n(\log n + \tilde{F}) \log \frac{r(\mathcal{N})}{\epsilon n})$ (greedy)	(decomposition), $O(n(\log n + \tilde{F}) \log \frac{r(\mathcal{N})}{n})$ (greedy)

Table 2.2: Overview of the worst-case time complexity results for the problem $(a, b)\text{-Q}/\beta/\gamma$

β	γ	
	C	I
Box	$O(n \log \frac{nR}{\epsilon})$	$O(n \log \frac{R}{n})$
GBC	$O(n \log \frac{nR}{\epsilon})$	$O(n \log \frac{R}{n})$
NC	$O(n \log m \log \frac{nR}{\epsilon})$	$O(n \log m \log R)$
LC	$O(n^2 \log n \log \frac{nR}{\epsilon}),$	$O(n^2 \log n \log \frac{mR}{n}),$
	$O(n \log n \log \frac{nR}{\epsilon})$	$O(n \log n \log \frac{R}{n})$
SC	(only upper constraints)	(only upper constraints)
	$O(n^2 \log \frac{nr(\mathcal{N})}{\epsilon} + n\tilde{E})$	$O(n^2 (\log \frac{r(\mathcal{N})}{n} + F \log r(\mathcal{N})) + n\tilde{F})$
	(decomposition), $O(n(\log n + \tilde{F}) \log \frac{r(\mathcal{N})}{\epsilon n})$ (greedy)	(decomposition), $O(n(\log n + \tilde{F}) \log \frac{r(\mathcal{N})}{n})$ (greedy)

Table 2.3: Overview of the worst-case time complexity results for the problem $S/\beta/\gamma$.

This problem has been extensively studied since the 1950s [137] and many solution approaches and algorithms have been proposed for this problem, especially for the problems $(a, b)\text{-Q}/\text{Box}/\gamma$. We refer to [137, 138] for surveys on the continuous version $S/\text{Box}/C$ and to [94] for a brief but thorough review on the integer version $S/\text{Box}/I$.

The best-known complexities for $S/\text{Box}/C$ and $S/\text{Box}/I$ are $O(n \log \frac{nR}{\epsilon})$ and $O(n \log \frac{R}{n})$ respectively, where ϵ is an accuracy parameter [45, 74]. Furthermore, their quadratic versions $(a, b)\text{-Q}/\text{Box}/C$ and $(a, b)\text{-Q}/\text{Box}/I$ can be solved in $O(n)$ time ([32] and [85] respectively). The linear-time algorithms for $(a, b)\text{-Q}/\text{Box}/C$ belong to the class of so-called *breakpoint search* algorithms that solve the problem by efficiently searching for the optimal Lagrange multiplier corresponding to the resource constraint (2.1) (see also [100]). The linear-time

algorithm for (a, b) -Q/Box/I in [85] first solves the continuous version (a, b) -Q/Box/C of this problem using a linear-time algorithm such as in [32]. Subsequently, it uses this solution and a specific rounding scheme to construct an instance of (a, b) -Q/Box/I with $R = O(n)$ that has the same optimal solution as the original instance of (a, b) -Q/Box/I. Using the algorithm in, e.g., [45, 74] for S/Box/I, this instance can be solved in $O(n \log \frac{O(n)}{n}) = O(n)$ time.

With regard to the actual execution time of the algorithms, there are several classes of algorithms that outperform the aforementioned linear-time algorithms. For example, for the problem (a, b) -Q/Box/C, it is shown in [101] that so-called variable-fixing algorithms that run in $O(n^2)$ time are in general faster than linear-time algorithms such as in [32]. These algorithms first compute a solution to the problem without the box constraints (2.2) and subsequently determine the optimal value of either all variables that exceed their lower bound or their upper bound in this solution. This process is repeated with the yet undetermined variables until none of these variables exceeds its bounds in the solution to the relaxed problem. The worst-case time complexity of $O(n^2)$ is attained when only one variable can be fixed to its optimal value during each step in the procedure. However, this is quite an extreme case since it has the property that in the optimal solution all variables are equal to one of their bounds.

Moreover, [177] shows that for several instances of (a, b, f) -S/Box/C, a specialized interior-point method significantly outperforms, among others, the linear-time breakpoint search approaches. Interior-point methods are iterative approaches where each intermediate solution is obtained from the previous one by taking a step in a search direction that is the solution of a perturbed version of the Karush-Kuhn-Tucker optimality conditions (see also [61]). Usually, the computation of this search direction is the computationally most expensive step of the interior-point method since it requires solving a linear system involving the constraint matrix. However, by exploiting the sparse structure of the constraint matrix for S/Box/C, the number of operations required to solve this system can be reduced from $O(n^3)$ to $O(n)$.

One reason for the bad practical performance of linear-time algorithms for (a, b) -Q/Box/C compared to other algorithms is that they require a linear-time computation of the median of sets of numbers such as given in [28]. These methods are in general significantly slower than alternative sorting-based approaches that run in linearithmic time [7, 99].

For the problem (a, b) -Q/Box/I, the linear-time complexity is based on the linear-time complexity of (a, b) -Q/Box/C and the existence of linear-time algorithms for selecting a k^{th} smallest element from a collection of sorted lists [85]. For the latter problem, many studies refer to [45] for such a linear-time algorithm. Analogously to the breakpoint search algorithms for (a, b) -Q/Box/C, the linear-time complexity of this algorithm depends on a linear-time algorithm for median-finding and may thus be slower in practice than alternative sorting-based approaches. It should be noted, however, that recently new linear-time

algorithms have been developed that are based on specialized heap data structures and have a better performance in practice (see, e.g., [93]).

The problem (a, b) -Q/Box/C with a quadratic objective function forms an important building block for RAPs studied in later chapters. Therefore, we consider this problem in more detail in Chapter 3 and discuss several algorithms for this problem, including a linear-time algorithm similar to those in [32, 100].

2.3.2 α /GBC/ γ : OPTIMIZATION OVER GENERALIZED BOUND CONSTRAINTS

In this subsection, we focus on the RAP with generalized bound constraints. The general version of this problem, S/GBC/ γ , can be formulated as follows:

$$\begin{aligned}
 \text{S/GBC}/\gamma : \min_x \quad & \sum_{i \in \mathcal{N}} \phi_i(x_i) \\
 \text{s.t.} \quad & \sum_{i \in \mathcal{N}} x_i = R, \\
 & L_j \leq \sum_{i \in \mathcal{N}_j} x_i \leq U_j, \quad j \in \mathcal{M}, \\
 & l_i \leq x_i \leq u_i, \quad i \in \mathcal{N}, \\
 & x \in \begin{cases} \mathbb{R}^n & \text{if } \gamma = \text{C}, \\ \mathbb{Z}^n & \text{if } \gamma = \text{I}, \end{cases}
 \end{aligned} \tag{2.3}$$

where $\mathcal{N}_1, \dots, \mathcal{N}_m$ is a partition of the index set $\mathcal{N}$, $\mathcal{M} = \{1, \dots, m\}$, and $L, U \in \mathbb{R}^m$. Applications of this problem include portfolio optimization [108], transportation problems [39], stratified sampling [146], and three-phase electric vehicle (EV) charging (see Chapter 5).

In the literature, this problem is studied primarily with only the upper bound constraints in (2.3). [74] shows that S/GBC/ γ with only generalized *upper* bound constraints can be solved with the same time complexity as S/Box/ γ by reducing the problem to a sequence of subproblems S/Box/ γ over in total n variables. We show in Chapter 5 (more precisely, Section 5.5.3) a similar result for (a, b) -Q/GBC/ γ with both generalized lower and upper bound constraints, which yields an $O(n)$ algorithm for solving (a, b) -Q/GBC/ γ . Finally, the continuous version of the problem, (a, b) -Q/GBC/C, can be solved alternatively in $O(n)$ time as a special case of quadratic programming with a fixed number of constraints [114].

2.3.3 α /NC/ γ : OPTIMIZATION OVER NESTED CONSTRAINTS

In this subsection, we focus on the RAP with nested constraints. The general version of this problem, S/NC/ γ , can be formulated as follows:

$$\begin{aligned}
\text{S/NC}/\gamma : \min_x \quad & \sum_{i \in \mathcal{N}} \phi_i(x_i) \\
\text{s.t.} \quad & \sum_{i \in \mathcal{N}} x_i = R, \\
& L_j \leq \sum_{i \in \mathcal{N}_j} x_i \leq U_j, \quad j \in \mathcal{M}, \tag{2.4}
\end{aligned}$$

$$l_i \leq x_i \leq u_i, \quad i \in \mathcal{N}, \tag{2.5}$$

$$x \in \begin{cases} \mathbb{R}^n & \text{if } \gamma = \text{C}, \\ \mathbb{Z}^n & \text{if } \gamma = \text{I}, \end{cases}$$

where $\mathcal{N}_1, \dots, \mathcal{N}_m$ are subsets of $\mathcal{N}$ such that $\mathcal{N}_1 \subset \dots \subset \mathcal{N}_m \subset \mathcal{N}$ and $\mathcal{M} = \{1, \dots, m\}$. Research on this problem has almost exclusively focused on the case with either the lower or upper nested constraints in (2.4) but not both. We refer to [5] for a survey on this version of the problem.

The theoretically most efficient algorithm for both S/NC/ γ and (a, b) -Q/NC/ γ is the decomposition algorithm in [172]. This algorithm solves the problem as a sequence of S/Box/ γ subproblems where the single-variable bounds (2.2) of each subproblem are optimal solutions to subproblems deeper in the decomposition hierarchy. The worst-case time complexity of this algorithm is $O(n \log m \log \frac{nR}{\epsilon})$ for S/NC/C, $O(n \log m \log R)$ for S/NC/I, and $O(n \log m)$ for both (a, b) -Q/NC/C and (a, b) -Q/NC/I. The latter two time complexities are attained by utilizing the linear-time algorithms for (a, b) -Q/Box/ γ to solve occurring subproblems. As mentioned in Section 2.3.1, these are not the fastest algorithms for these subproblems in practice. As a consequence, it can be expected that using, e.g., variable-fixing algorithms [101] for the subproblems in practice significantly improves the overall execution time of the algorithm.

It has been shown that infeasibility-guided algorithms such as in [167, 178] are significantly faster in practice than the decomposition algorithm in [172] (see also Chapter 4). These algorithms first compute a solution to S/NC/ γ without the nested constraints (2.4) and, based on which nested constraint is violated most in this solution, subsequently divide the problem into two smaller instances of this problem. Analogously to the variable-fixing algorithms for (a, b) -Q/Box/C, the maximum number of divisions is $O(n)$, which results in a worst-case time complexity of $O(n^2 \log \frac{nR}{\epsilon})$ for S/NC/C [167] and $\Theta(n^2 \log \frac{R}{n})$ for S/NC/I [178]. However, this worst-case complexity occurs only in cases where the number of divisions is $n - 1$, which are instances where each nested constraint is tight in an optimal solution. For the case with only upper nested constraints (2.4), lower single-variable bounds (2.5), and randomly generated problem parameters, [171] shows that the expected number of tight constraints in an optimal solution to (a, b, f) -S/NC/C is $O(\log n)$.

Recently, for the problem (a, b) -Q/NC/C with only upper nested constraints, [176] shows that a specialized interior-point method is able to outperform the decomposition-based approach in [171], which is similar to the approach in [172] that yielded the currently most efficient algorithm, when the number m of nested constraints is more than 10% of the number n of variables. As mentioned in Section 2.3.1, this method exploits the constraint structure of S/NC/C to compute search directions in $O(n)$ time instead of $O(n^3)$ time analogously to [177]. Although the authors in [176] consider only upper nested constraints, it is straightforward to generalize their results to problems involving also lower nested constraints [156].

Interestingly, [5, 171] show that we can solve the problem (a, b, f) -S/NC/C with only nested upper constraints and without the box constraints (2.5) in $O(n)$ time. More precisely, they show that this problem can be reduced to the problem of finding a concave cover of n points in $\mathbb{R}^2$ and give an $O(n)$ time algorithm to find this cover.

2.3.4 $\alpha/\text{LC}/\gamma$: OPTIMIZATION OVER LAMINAR CONSTRAINTS

In this subsection, we focus on the RAP with laminar constraints. The general version of this problem, S/LC/ γ , can be formulated as follows:

$$\begin{aligned}
 \text{S/LC}/\gamma : \quad & \min_x \sum_{i \in \mathcal{N}} \phi_i(x_i) \\
 \text{s.t.} \quad & \sum_{i \in \mathcal{N}} x_i = R, \\
 & L_j \leq \sum_{i \in \mathcal{N}_j} x_i \leq U_j, \quad j \in \mathcal{M}, \\
 & l_i \leq x_i \leq u_i, \quad i \in \mathcal{N}, \\
 & x \in \begin{cases} \mathbb{R}^n & \text{if } \gamma = \text{C}, \\ \mathbb{Z}^n & \text{if } \gamma = \text{I}, \end{cases}
 \end{aligned} \tag{2.6}$$

where $L, U \in \mathbb{R}^m$ and $\mathcal{N}_1, \dots, \mathcal{N}_m$ are subsets of $\mathcal{N}$ that satisfy the following property: if $\mathcal{N}_j \cap \mathcal{N}_\ell \neq \emptyset$, then either $\mathcal{N}_j \subset \mathcal{N}_\ell$ or $\mathcal{N}_j \supset \mathcal{N}_\ell$ for all $j, \ell \in \mathcal{M} = \{1, \dots, m\}$.

Similarly to S/NC/ γ , the problem S/LC/ γ was studied mainly with only the upper laminar constraints in (2.6). The algorithms with the lowest computational complexities for these problems are given by [74] and have time complexities of $O(n \log n \log \frac{nR}{\epsilon})$ for $\gamma = \text{C}$ and $O(n \log n \log \frac{R}{n})$ for $\gamma = \text{I}$. For the general problem S/LC/ γ , i.e., with both lower and upper laminar constraints, we obtain an efficient algorithm by combining results on the complexity of general separable convex optimization problems with linear constraints [76]

and of S/LC/C with a linear objective function [134]. More precisely, the time complexities of S/LC/C and S/LC/I are $O(P_{\text{linear}}(8n^2, m) \log \frac{Rn}{\epsilon})$ and $O(P_{\text{linear}}(4n^2, m) \log \frac{Rm}{n})$ respectively, where $P_{\text{linear}}(n, m)$ is the time complexity of solving an instance of S/LC/C with a linear objective function [76]. The latter can be solved in $O(n \log n)$ time using the algorithm in [134]. Thus, we obtain time complexities of $O(n^2 \log n \log \frac{Rn}{\epsilon})$ and $O(n^2 \log n \log \frac{Rm}{n})$ for S/LC/C and S/LC/I respectively.

With regard to the quadratic version of the problem, the special case of (a, b) -Q/LC/C with only upper laminar constraints can be solved in $O(n \log n)$ time [75]. This is done by reducing the problem to an instance of (a, b) -Q/NC/C with only upper nested constraints, which can be solved in $O(n \log n)$ time [75]. The general version of (a, b) -Q/NC/C with both lower and upper laminar constraints can be solved in $O(n^2)$ time as an instance of the quadratic convex cost flow problem on a tree network [163]. Finally, the integer-valued problem (a, b) -Q/NC/I can be solved in $O(n^2)$ time by first computing a solution to the continuous version of this problem and subsequently applying a specific rounding procedure to round this solution to an optimal integer solution [120].

As far as we are aware, the problem S/LC/ γ has been studied primarily from an academic point of view, i.e., little attention is paid to possible applications. One relevant application that has received quite some importance in the past years is the scheduling of the (dis)charging of an electrical storage system within a smart grid (see also Section 2.5.2) where the energy can be drawn from all three phases within the low-voltage distribution network (see also Chapter 5). The resulting problem is an instance of S/LC/C where the feasible set is the intersection of nested constraints (to model the storage capacity limits) and generalized upper bound constraints (to model the charging limits). The further investigation of this problem is an interesting topic for future research.

2.3.5 α /SC/ γ : OPTIMIZATION OVER SUBMODULAR CONSTRAINTS

In this subsection, we focus on the RAP with submodular constraints. The general version of this problem, S/SC/ γ , can be formulated as follows:

$$\begin{aligned}
 \text{S/SC}/\gamma : \quad & \min_x \sum_{i \in \mathcal{N}} \phi_i(x_i) \\
 \text{s.t.} \quad & \sum_{i \in \mathcal{N}} x_i = r(\mathcal{N}), \\
 & \sum_{i \in \mathcal{S}} x_i \leq r(\mathcal{S}), \quad \mathcal{S} \subset \mathcal{N}, \\
 & x \in \begin{cases} \mathbb{R}^n & \text{if } \gamma = \text{C}, \\ \mathbb{Z}^n & \text{if } \gamma = \text{I}, \end{cases}
 \end{aligned} \tag{2.7}$$

where r is a submodular function over the ground set $\mathcal{N}$.

For this problem, there are classes of algorithms in the literature. The first class consists of decomposition algorithms that first compute a solution to the problem without the submodular constraints (2.7) and, based on which constraints are violated by this solution, split up the problem into two smaller instances of $\text{S/SC}/\gamma$ [47, 64]. Note that the infeasibility-guided algorithms for $\text{S/NC}/\gamma$ as discussed in Section 2.3.3 are based on the same principle. The best worst-case time complexities of such algorithms are $O(n^2 \log \frac{nr(\mathcal{N})}{\epsilon} + n\tilde{E})$ for $\text{S/SC}/\text{C}$ [127] and $O(n^2(\log \frac{r(\mathcal{N})}{n} + F \log r(\mathcal{N})) + n\tilde{F})$ for $\text{S/SC}/\text{I}$ [94], where $\tilde{E}$ is the time required to minimize a given submodular function and F is the time required to check the feasibility of a given vector for the submodular constraints. Moreover, $\tilde{F}$ is the time required to determine for a given solution x that is feasible for the submodular constraints (2.7) by how much we can increase a given variable x_i without violating any of these submodular constraints. For the quadratic problems $(a, b)\text{-Q/SC}/\gamma$, these complexities reduce to $O(n^2 + n\tilde{E})$ for $(a, b)\text{-Q/SC}/\text{C}$ and to $O(n^2 F \log r(\mathcal{N}) + n\tilde{F})$ for $(a, b)\text{-Q/SC}/\text{I}$.

The second class consists of greedy algorithms that solve the integer version $\text{S/SC}/\text{I}$ by incrementally building an optimal solution (see, e.g., [74, 119]). However, instead of incrementing the total amount of allocated resource by unit steps, these algorithms apply a scaling procedure to determine larger step sizes that speed up the building process while still maintaining feasibility of the current solution. To solve the continuous version $\text{S/SC}/\text{C}$, these algorithms exploit a proximity result between optimal solutions of $\text{S/SC}/\text{C}$ and $\text{S/SC}/\text{I}$ (see, e.g., [120]) that states that for any optimal solution x^* to $\text{S/SC}/\text{I}$ there exists an optimal solution $\tilde{x}$ to $\text{S/SC}/\text{C}$ such that $|\tilde{x}_i - x_i^*| \leq n - 1$. As a consequence, to solve $\text{S/SC}/\text{C}$ with a given accuracy ϵ , one can scale all problem parameters by a factor $\lceil \frac{n}{\epsilon} \rceil$, solve the scaled problem with integer variables using the greedy algorithm, and scale back the resulting solution. The most efficient algorithms of this class run in $O(n(\log n + \tilde{F}) \log \frac{r(\mathcal{N})}{\epsilon n})$ time for $\text{S/SC}/\text{C}$ and $O(n(\log n + \tilde{F}) \log \frac{r(\mathcal{N})}{n})$ for $\text{S/SC}/\text{I}$ [74, 119], which are also the time complexities for the quadratic cases $(a, b)\text{-Q/SC}/\text{C}$ and $(a, b)\text{-Q/SC}/\text{I}$.

One notable special case of $(a, b)\text{-Q/SC}/\text{C}$ is the problem of computing the point with minimum Euclidean norm of a base polytope (see, e.g., [49]). This problem is equivalent to $(\bar{e}, 0)\text{-Q/SC}/\text{C}$ and plays an important role as a subroutine in several algorithms for machine learning problems and submodular function minimization [12, 50]. One of the most popular algorithms in practice for finding the point with minimum norm is Wolfe's algorithm [175], which solves the problem by iteratively updating a hyperplane and the minimum-norm point on this hyperplane based on the feasibility of this point. The authors in [33] show that this algorithm computes an ϵ -approximate solution to the problem $(\bar{e}, 0)\text{-Q/SC}/\text{C}$ in $O(\frac{nM^2}{\epsilon})$ time, where M is the norm of the point in the base polytope with the *largest* Euclidean norm. Although there are algorithms for finding the point with minimum norm that have a better computational complexity, e.g.,

the aforementioned decomposition and greedy algorithms, Wolfe’s algorithm has been shown to be among the fastest algorithms in practice [12, 50].

2.4 REDUCTION OF (a, b, f) -SEPARABLE RAPs TO QUADRATIC RAPs

This section shows a reduction result concerning (a, b, f) -separable RAPs with submodular constraints. More precisely, we show that for given parameters a and b and an instance of this class of RAPs, there exists a feasible solution to this instance that is optimal for any choice of the convex function f . In particular, we show that any solution that is optimal to the basic quadratic version of this RAP, i.e., where $f(x_i) = \frac{1}{2}x_i^2$, is also optimal for the (a, b, f) -separable version for any choice of f . This means that solving any (a, b, f) -separable RAP reduces to solving the quadratic version of this RAP and allows us to solve this problem using any tailored algorithm that solves the quadratic RAP. Thus, to solve this problem, we do not require algorithms designed to solve the more general version with arbitrary convex cost functions, which are in general much slower and less efficient than the tailored algorithms for the quadratic RAP. As a consequence, our reduction result allows us to solve a wide range of RAPs using the extensive collection of solution approaches and algorithms for quadratic RAPs. In particular, all algorithms for quadratic RAPs discussed in the previous section can be used to solve (a, b, f) -separable RAPs.

We first provide in Section 2.4.1 a brief overview of existing reduction and equivalence results for RAPs. Subsequently, in Section 2.4.2, we prove the stated reduction result.

2.4.1 REDUCTION RESULTS IN THE LITERATURE

In the literature, reduction results similar to the one that we derive in this section already exist for specific RAPs. For instance, for RAPs over submodular constraints, [47] showed that the problem with quadratic cost functions is equivalent to the problem of computing a lexicographically optimal base with regard to a given weight vector. [127] extends this result to a range of different strictly convex cost functions for the case of continuous variables. Their result is used in [128] to solve optimization problems on graphs and in [153] to derive efficient algorithms for processor scheduling problems. For a special case of RAPs with nested constraints, the equivalence of (a, b, f) -separable RAPs is proven in [5] for the case where the functions f are strictly convex and differentiable, $b = 0$, and where the variables are continuous.

Some reduction results can be derived using existing algorithms for specific applications in the literature. An example of this concerns the vessel speed optimization problem (see, e.g., [131]). In this problem, a ship traverses a given route between ports and must dock at each given port within a specific time window. The goal is to determine the ship’s speed between each leg of the route

while minimizing fuel costs. The authors in [131] propose a recursive-smoothing algorithm (RSA) for this problem, which is shown to be optimal by [84]. This algorithm, however, does not require knowledge on the fuel cost function other than that it is convex. Thus, the optimal solution outputted by this algorithm is indifferent of the choice of cost function.

Another example is the scheduling of tasks on a single processor with agreeable deadlines (see, e.g., [55]). Here, we are given a number of tasks that must be processed on a single processor, each of which has its own workload, arrival time, and deadline. The goal is to assign processor speeds to tasks such that all tasks are finished before their deadline while minimizing the total energy usage of the processor. This energy usage depends on the workload of each task and on the required power to maintain a given processor speed. Analogously to the vessel speed optimization problem, the processor scheduling problem can be solved using the RSA without any knowledge of the exact relation between the speed and the energy usage other than that it is convex [83]. Thus, also the optimal solution given by this algorithm does not depend on the power function.

Our reduction result generalizes all the above results to general convex functions f , i.e., not necessarily *strictly* convex or differentiable, and to both continuous and integer variables. In particular, in the case of continuous variables and a *strictly* convex function f , our reduction result becomes an equivalence result since the optimal solution to any strictly convex optimization problem is unique. In fact, given the parameters a and b , an instance to the RAP, and two strictly convex functions f and $\bar{f}$, we show that the (a, b, f) -separable and $(a, b, \bar{f})$ -separable versions of this RAP have the same unique optimal solution and are thus equivalent in that sense.

2.4.2 THE REDUCTION RESULT

We show for all the RAPs introduced in Section 2.2 that their (a, b, f) -separable versions reduce to their quadratic versions. More precisely, given a constraint structure β , variable type γ , convex function f , and vectors $a \in \mathbb{R}_{>0}^n$ and $b \in \mathbb{R}^n$, we show that any optimal solution to (a, b) -Q/ β/γ is also optimal for (a, b, f) -S/ β/γ . This means that we can solve (a, b, f) -S/ β/γ by solving (a, b) -Q/ β/γ . Note that for many of these quadratic RAPs, tailored algorithms exist that are faster and more efficient than other algorithms for the case with arbitrary convex cost functions. Thus, this reduction result allows us to solve (a, b, f) -S/ β/γ problems using fast algorithms for their quadratic special case.

We start by considering the general constrained optimization problem S/C with a convex separable objective function:

$$\begin{aligned} \text{S/C : } \min_x \quad & \sum_{i \in \mathcal{N}} \phi_i(x_i) \\ \text{s.t. } \quad & x \in \mathcal{C}, \end{aligned}$$

where $C \subseteq \mathbb{R}^n$ and $\phi_1, \dots, \phi_n$ are single-variable convex functions. We do not assume any properties on the set C other than that it is nonempty. Note that all RAPs introduced in the previous section are special instances of this problem.

We show that if S/C satisfies a certain optimality condition, any optimal solution to $(a, b)\text{-}Q/C$ is also optimal for $(a, b, f)\text{-}S/C$. This optimality condition states that a feasible solution x to S/C is optimal if and only if moving an arbitrary amount from one variable x_i to another variable x_k while maintaining feasibility never leads to a decrease in objective value. We state this condition as Condition 2.1 and give the mentioned reduction result in Theorem 2.1.

Condition 2.1. *Given separable convex functions $\phi_\ell : \mathbb{R} \rightarrow \mathbb{R}$ for $\ell \in \mathcal{N}$ and a set $C \subset \mathbb{R}^n$, a feasible solution x to S/C is optimal if and only if we have for each exchangeable pair $(i, k) \in \mathcal{E}_C(x)$ that $\phi_k^+(x_k) \geq \phi_i^-(x_i)$.*

Theorem 2.1. *Let the set C , a convex function f , and $a \in \mathbb{R}_{>0}^n$ and $b \in \mathbb{R}^n$ be given. If Condition 2.1 holds for S/C both for the separable convex functions $\phi_i(x_i) = \frac{1}{2} \frac{x_i^2}{a_i} + b_i x_i$ for all $i \in \mathcal{N}$ and $\phi_i(x_i) = a_i f(\frac{x_i}{a_i} + b_i)$ for all $i \in \mathcal{N}$ and $x \in C$ is optimal for $(a, b)\text{-}Q/C$, then x is also optimal for $(a, b, f)\text{-}S/C$.*

Proof. Let x be an optimal solution to $(a, b)\text{-}Q/C$. Since this problem is an instance of S/C , we may apply Condition 2.1, i.e., we have that $\phi_k^+(x_k) \geq \phi_i^-(x_i)$ for all $i, k \in \mathcal{E}_C$ where $\phi_\ell^-(x_\ell) = \phi_\ell^+(x_\ell) = \frac{x_\ell}{a_\ell} + b_\ell$ for all $\ell \in \mathcal{N}$. Thus, we have that $\frac{x_k}{a_k} + b_k \geq \frac{x_i}{a_i} + b_i$ for all $(i, k) \in \mathcal{E}_C(x)$. Since by convexity of f the right derivative f^+ is non-decreasing and we have $f^+(y) \geq f^-(y)$ for all $y \in \mathbb{R}$, it follows that $f^+\left(\frac{x_k}{a_k} + b_k\right) \geq f^+\left(\frac{x_i}{a_i} + b_i\right) \geq f^-\left(\frac{x_i}{a_i} + b_i\right)$ for all $i, k \in \mathcal{E}_C(x)$. Since Condition 2.1 holds in particular for the choice $\phi_\ell(x_\ell) := a_\ell f(\frac{x_\ell}{a_\ell} + b_\ell)$, which is the cost function for the problem $(a, b, f)\text{-}S/C$, we may apply this condition again to conclude that x is optimal for $(a, b, f)\text{-}S/C$. $\square$

Note that Theorem 2.1 does not explicitly require the problem S/C to be a RAP. This means that this theorem and thus our reduction result might be more widely applicable to other problems, provided that they satisfy Condition 2.1.

It is well-known that $S/SC/\gamma$ satisfies Condition 2.1 for $\gamma \in \{C, I\}$ (see, e.g., [49, 64]). To gain some insight into why this is the case, we provide for the interested reader in Appendix 2.8 an alternative proof for this claim for the relevant special case $S/LC/\gamma$ that relies only on basic concepts from convex analysis such as subgradients. It follows that Theorem 2.1 can be applied to $S/SC/\gamma$ and in particular also to all special cases of this problem:

Corollary 2.1. *Let a convex function f , vectors $a \in \mathbb{R}_{>0}^n$, $b \in \mathbb{R}^n$, and entries β and γ as specified in Table 2.1 be given. If x is optimal for $(a, b)\text{-}Q/\beta/\gamma$, then x is also optimal for $(a, b, f)\text{-}S/\beta/\gamma$.*

This corollary is an extension of the equivalence results in [127], where the reduction result is shown for the two special cases with continuous variables where f is *strictly* convex and differentiable or where $\phi_i = f$ for all $i \in \mathcal{N}$.

The validity of the reduction result of Theorem 2.1 for RAPs with submodular constraints and its special cases implies that any algorithm for solving the quadratic version of this problem can be used to solve the (a, b, f) -separable version. In particular, any time complexity or efficiency results for the quadratic version as discussed in Section 2.3 also apply to the (a, b, f) -separable version:

Corollary 2.2. *Let a convex function f , vectors $a \in \mathbb{R}_{>0}^n$, $b \in \mathbb{R}^n$, and entries β and γ as specified in Table 2.1 be given. The worst-case time complexity of (a, b, f) - $S/\beta/\gamma$ is at most that of (a, b) - $Q/\beta/\gamma$.*

In particular, the complexity results for the quadratic RAPs in Table 2.2 apply also to their (a, b, f) -separable versions.

Finally, for the case of continuous variables, Theorem 2.1 holds also when we replace the problem (a, b) - $Q/\beta/C$ by (a, b, f) - $S/\beta/C$, where f is a *strictly* convex function. This effectively turns our reduction result into an equivalence result between these two problems:

Corollary 2.3. *Let $a \in \mathbb{R}_{>0}^n$, $b \in \mathbb{R}^n$, and entries β as specified in Table 2.1 be given, and let $\bar{f}$ be a strictly convex function and f be an arbitrary convex function. If x is optimal for $(a, b, \bar{f})$ - $S/\beta/C$, then x is also optimal for (a, b, f) - $S/\beta/C$.*

Proof. Since $\bar{f}$ is *strictly* convex, x is the *unique* optimal solution to $(a, b, \bar{f})$ - $S/\beta/C$. It follows from Theorem 2.1 that the unique optimal solution to (a, b) - $Q/\beta/C$ is x and thus that x is also optimal for (a, b, f) - $S/\beta/C$. $\square$

Corollary 2.3 allows us to solve a given continuous (a, b, f) -separable RAP using any algorithm that solves the $(a, b, \bar{f})$ -separable version of the problem for some strictly convex function $\bar{f}$, i.e., not only just for quadratic objectives. This can be beneficial in cases where efficient algorithms have already been developed for a specific choice of a non-quadratic objective function, motivated by the given application.

2.5 APPLICATIONS

In this section, we model several energy management problems and a problem in statistics as RAPs. In particular, we show the relevance of (a, b, f) -separable RAPs and the reduction result proved in the previous section. Moreover, the reduction result leads to new insights into common practices to solve these problems. In particular, we show that two problems in the area of vessel routing and processor scheduling can be solved in $O(n \log n)$ time rather than $O(n^2)$

time, which was the previously best-known complexity for these problems. Finally, with this collection of applications and the included references, we intent to stimulate cross-disciplinary research that leads to new structural results and algorithms for RAPs that are applicable to many different research fields.

2.5.1 POWER ALLOCATION IN MULTI-CHANNEL COMMUNICATION SYSTEMS

In many telecommunication systems, data can be transmitted over several parallel channels to reduce the amount of noise experienced when transmitting the data (see, e.g., [150]). The amount of data that can be transmitted through a given channel i , i.e., the channel capacity, depends on the power x_i spent on this channel, its bandwidth B_i , and a “gain” parameter c_i that represents the amount of noise on the channel. One goal in these systems is to allocate a given budget of total power P^{tot} over a set $\mathcal{N}$ of n channels such that the overall channel capacity is maximized while respecting power limits on each channel. This problem can be formulated mathematically as

$$\begin{aligned} \text{POWER : } \max_{x \in \mathbb{R}^n} \quad & \sum_{i \in \mathcal{N}} B_i \log(1 + c_i x_i) \\ \text{s.t.} \quad & \sum_{i \in \mathcal{N}} x_i = P^{\text{tot}}, \\ & 0 \leq x_i \leq \bar{P}_i, \quad i \in \mathcal{N}, \end{aligned}$$

where $\bar{P}_i$ is the maximum allowed power on channel i .

Note that for a given channel $i \in \mathcal{N}$ we have

$$B_i \log(1 + c_i x_i) = B_i \log \left(\frac{\frac{1}{c_i} + x_i}{B_i} B_i c_i \right) = B_i \log \left(\frac{\frac{1}{c_i} + x_i}{B_i} \right) + B_i \log(B_i c_i).$$

Since the second term $B_i \log(B_i c_i)$ in the above expression is constant, we can replace the objective function of Problem POWER by $\sum_{i \in \mathcal{N}} B_i \log \left(\frac{x_i}{B_i} + \frac{1}{B_i c_i} \right)$ without changing the optimal solution to the problem. The resulting problem is an instance of $(B, \bar{B}, f)$ -S/Box/C with $B := (B_i)_{i \in \mathcal{N}}$, $\bar{B} := (\frac{1}{B_i c_i})_{i \in \mathcal{N}}$, and $f(x_i) := -\log(x_i)$. Thus, by Corollary 2.1, we can solve this problem as an instance of $(B, \bar{B})$ -Q/Box/C in $O(n)$ time, i.e., we can replace each term $B_i \log(1 + c_i x_i)$ by $\frac{x_i^2}{B_i} + \frac{x_i}{B_i c_i}$. Note that this leads to a more efficient approach than several existing approaches for solving Problem POWER that claim a linear time complexity (see, e.g., [98, 106]). The reason for this is that these algorithms achieve this complexity only if the gain parameter c has already been sorted, which is, however, only the case for some specific communication systems (see, e.g., [136]).

Another common objective for the channel power allocation problem POWER (see, e.g., [179]) is to minimize the mean square error between different channels

from a set $\mathcal{N}$. This objective is given by

$$\min_{x \in \mathbb{R}^n} \sum_{i \in \mathcal{N}} \frac{w_i}{A_i x_i + D_i},$$

where w_i , A_i , and D_i are positive parameters for each $i \in \mathcal{N}$. This objective function is (a, b, f) -separable by choosing $a_i := \sqrt{\frac{w_i}{A_i}}$ and $b_i := \frac{D_i}{\sqrt{w_i A_i}}$ for each $i \in \mathcal{N}$ and $f(x_i) := \frac{1}{x_i}$:

$$a_i f\left(\frac{x_i}{a_i} + b_i\right) = \sqrt{\frac{w_i}{A_i}} \frac{1}{x_i \sqrt{\frac{A_i}{w_i} + \frac{D_i}{\sqrt{w_i A_i}}}} = \frac{\sqrt{w_i}}{\frac{A_i x_i}{\sqrt{w_i}} + \frac{D_i}{\sqrt{w_i}}} = \frac{w_i}{A_i x_i + D_i}.$$

Moreover, several variations of the channel power allocation problem have been studied with, e.g., bounds on disjoint or nested subsets of allocations (see, e.g., [70] and [41] respectively). Analogously to Problem POWER, one can show that these problems are instances of (a, b, f) -S/GBC/C and (a, b, f) -S/NC/C and thus can be solved as instances of (a, b) -Q/GBC/C and (a, b) -Q/NC/C respectively.

2.5.2 STORAGE OPERATION IN ENERGY SYSTEMS

Storage systems are becoming a crucial part of current and future sustainable energy systems (see, e.g., [109, 144, 183]). Such systems support satisfying the energy demand of, e.g., a neighborhood, when renewable energy sources such as solar and wind are insufficient due to, e.g., unfavorable weather conditions. Commonly, the operation of the storage systems is steered in a way that the stress on the overall grid is reduced as much as possible. Determining for a given time horizon the best operational schedule for the storage, i.e., how much energy should be (dis)charged at each moment to reach the overall goal in the best way, leads to an optimization problem. In this problem, we discretize the overall time horizon into n equidistant time intervals of length Δt indexed by the set $\mathcal{N} := \{1, \dots, n\}$ and determine for each interval $i \in \mathcal{N}$ the (dis)charging power x_i during this interval. This amount is limited by the given minimum and maximum charging rates $X_{\min}$ and $X_{\max}$. Moreover, the charging must be done such that the storage capacity D is not exceeded. Given an initial amount of energy S_{start} in the storage and a desired eventual target amount S_{end} , the storage operation problem can be formulated as follows (see also [167]):

$$\begin{aligned}
\text{BATTERY : } & \min_{x \in \mathbb{R}^n} \sum_{i \in \mathcal{N}} \phi_i(x_i) \\
\text{s.t. } & 0 \leq S_{\text{start}} + \Delta t \sum_{i=1}^j x_i \leq D, \quad j \in \mathcal{N} \setminus \{n\}, \\
& S_{\text{start}} + \Delta t \sum_{i \in \mathcal{N}} x_i = S_{\text{end}}, \\
& X_{\min} \leq x_i \leq X_{\max}, \quad i \in \mathcal{N},
\end{aligned}$$

where the functions ϕ_i represent the desired objective of the grid to which the storage asset is connected. Note that if each function ϕ_i is convex, which is in general the case in this problem setting, this problem is an instance of S/NC/C.

Three commonly used objectives to reduce grid stress and congestion are: minimal import and export of energy from the main grid (also known as energy autarky, see, e.g., [124]), load profile flattening (see, e.g., [58]), and minimizing peak consumption (see, e.g., [165]). One often-used way to model the latter case is to set a maximum level M for the overall power consumption of the neighborhood and to penalize any consumption above this level.

Given the power consumption $p := (p_i)_{i \in \mathcal{N}}$ of the neighborhood, we can model these objectives as follows:

Minimizing main grid exchange: $\phi_i(x_i) = |x_i + p_i|$;

Load profile flattening: $\phi_i(x_i) = (x_i + p_i)^2$;

Threshold peak shaving: $\phi_i(x_i) = \begin{cases} 0 & \text{if } x_i + p_i \leq M, \\ \underline{f}(x_i + p_i) & \text{if } x_i + p_i > M, \end{cases}$

where $\underline{f}$ is a convex non-decreasing function with $\underline{f}(M) = 0$. Note that for the objective of load profile flattening, Problem BATTERY is an instance of $(\bar{e}, p)$ -Q/NC/C, where $\bar{e}$ is the vector of ones. Moreover, for the other two objectives, Problem BATTERY is an instance of $(\bar{e}, p, f)$ -S/NC/C where f is the absolute value function or the piecewise function

$$f(y) = \begin{cases} 0 & \text{if } y \leq M, \\ \underline{f}(y) & \text{if } y > M. \end{cases}$$

It follows by Corollary 2.1 that the optimal solution to $(\bar{e}, p)$ -Q/NC/C is also optimal for $(\bar{e}, p, f)$ -S/NC/C for these two functions. This implies that we can schedule the storage (dis)charging such that all three objectives are satisfied simultaneously by aiming for load profile flattening. This is an effect that can also be observed for other renewable energy systems such as a photovoltaic (solar panel) system combined with an EV charging infrastructure (see, e.g., [126])

and heat pumps (see, e.g., [168]). Moreover, energy tariff systems that employ piecewise linear cost functions have been shown to be able to flatten the load profile, i.e., the objective modeled by a quadratic cost function (see, e.g., [143]). Since such tariff systems are simpler to explain to end-users, they are more likely to be accepted than systems using quadratic cost functions while still achieving the desired objective of load profile flattening.

2.5.3 STRATIFIED SAMPLING

Stratified sampling is a sampling method suitable for situations where it is likely that a random sample is not a proper representation of the population [130]. Such a situation occurs, e.g., when several subclasses of the population score extremely on the to-be-estimated characteristic. To deal with this specific case, the given population is partitioned into n so-called *strata* with sizes $N_1, \dots, N_n$ that, ideally, represent the aforementioned subclasses. Given the desired overall sample size R , the goal is to determine for each stratum $i \in \mathcal{N} := \{1, \dots, n\}$ the number of samples x_i drawn from this stratum while minimizing the variance of the given characteristic. Following the formulation in [46], the optimal sample allocation is the solution of the following optimization problem:

$$\begin{aligned} \min_{x \in \mathbb{Z}^n} \quad & \sum_{i \in \mathcal{N}} \left(\frac{N_i^2 S_i^2}{x_i} - N_i S_i^2 \right) \\ \text{s.t.} \quad & \sum_{i \in \mathcal{N}} x_i = R, \\ & 0 \leq x_i \leq N_i, \quad i \in \mathcal{N}, \end{aligned}$$

where S_i^2 is the variance of the characteristic within stratum i . Similarly to [46], the sample bounds of 0 and N_i can be chosen differently to ensure a minimum or maximum number of samples drawn from a given stratum.

Let $D \in \mathbb{R}^n$ be a vector with $D_i := N_i^2 S_i^2$ for all $i \in \mathcal{N}$. Then the above problem is an instance of the problem $(D, 0, f)$ -S/Box/I with $f(x_i) = \frac{1}{x_i}$. Thus, by Corollary 2.2, we can solve this problem as an instance of $(D, 0)$ -Q/Box/I in $O(n)$ time. Note that, in contrast to the approaches in, e.g., [46], this complexity depends only on the number n of strata and not on the actual strata sizes $N_1, \dots, N_n$ or desired sample size R . As a consequence, our reduction result yields a promising approach to determine optimal sample sizes in large datasets, which can contain billions of samples (see, e.g., [115]).

2.5.4 VESSEL SPEED OPTIMIZATION

A recent trend in ship routing is to actively manage the ship's sailing speed to reduce fuel costs and carbon emissions [141]. As a consequence, when determining the routes of a fleet of ships to deliver cargo within given timing constraints, one needs a subroutine to determine the minimum cost of having a ship sail a

given route. This problem is known as the vessel speed optimization problem (see, e.g., [84, 131]). In this problem, we are given a route between $n + 1$ ports starting at port 0 at time t^{start} and required to finish at time t^{end} at port n . The distance between consecutive ports $i - 1$ and i is given by d_i and each port i must be served by the ship within a given time window $[A_i, D_i]$. The goal is to determine for each leg $i \in \mathcal{N} := \{1, \dots, n\}$ of the tour, with distance d_i , a speed v_i such that the fuel cost of sailing at these speeds is minimized. Following [84, 131], this leads to the following problem:

$$\begin{aligned} \text{VESSEL : } \quad & \min_{t \in \mathbb{R}^{n+1}, v \in \mathbb{R}^n} \sum_{i \in \mathcal{N}} d_i c(v_i) \\ \text{s.t. } & t_i + \frac{d_i}{v_i} = t_{i+1}, \quad i \in \{0\} \cup \mathcal{N} \setminus \{n\}, \\ & A_i \leq t_i \leq D_i, \quad i \in \mathcal{N} \setminus \{n\}, \\ & t_0 = t^{\text{start}}, \quad t_n = t^{\text{end}}, \\ & v^{\min} \leq v_i \leq v^{\max}, \quad i \in \mathcal{N}. \end{aligned}$$

Here, $v^{\min}$ and $v^{\max}$ are the minimum and maximum cruising speeds and c is a non-decreasing convex function that models the relation between sailing speed and fuel costs per unit distance.

From this formulation, it follows by induction on i that $t_i = t^{\text{start}} + \sum_{k=1}^i \frac{d_k}{v_k}$ for all $i \in \mathcal{N}$ and that $t^{\text{start}} + \sum_{k=1}^n \frac{d_k}{v_k} = t^{\text{end}}$. Let $x_i := \frac{d_i}{v_i}$ and $q(x_i) := c(1/x_i)$. It follows that

$$d_i c(v_i) = d_i c\left(\frac{d_i}{x_i}\right) = d_i q\left(\frac{x_i}{d_i}\right).$$

Note that q is convex since c is convex and non-decreasing. This means that Problem VESSEL is equivalent to the following convex optimization problem:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \sum_{i \in \mathcal{N}} d_i q\left(\frac{x_i}{d_i}\right) \\ \text{s.t. } & A_i - t^{\text{start}} \leq \sum_{k=1}^i x_k \leq D_i - t^{\text{start}}, \quad i \in \mathcal{N} \setminus \{n\}, \\ & \sum_{i \in \mathcal{N}} x_i = t^{\text{end}} - t^{\text{start}}, \\ & \frac{d_i}{v^{\max}} \leq x_i \leq \frac{d_i}{v^{\min}}, \quad i \in \mathcal{N}. \end{aligned}$$

This problem is an instance of $(d, 0, q)$ -S/NC/C. Hence, by Corollary 2.2, this problem and thus Problem VESSEL can be solved in $O(n \log n)$ time.

Efficient energy usage is an important topic within the development of computing systems [185]. To reduce energy consumption, modern computer processors can adjust their speed to save energy while still meeting their performance constraints. This leads to scheduling problems where a set of tasks needs to be scheduled and processor speeds need to be chosen such that all tasks are executed before their deadline (see [59] for a survey). One special case of these types of scheduling problems is the case where the deadlines are agreeable, i.e., deadlines are ordered according to the arrival times of the tasks (see also [15]). In this problem, we are given n tasks indexed by the set $\mathcal{N}$ that must be processed on a single processor. Each task $i \in \mathcal{N}$ has an arrival time A_i , deadline D_i , and amount of work w_i that can be interpreted as the amount of operations and calculations the processor must execute to perform this task. The goal is to select for each task i an execution speed s_i and starting time B_i such that each task is processed before its deadline and the total energy usage of the processor is minimized.

Since the deadlines are agreeable, we have that $D_i \leq D_k$ if $A_i \leq A_k$ for any two tasks $i, k \in \mathcal{N}$. Moreover, it is known that in an optimal schedule the tasks can be scheduled in non-decreasing order of their deadlines [15]. This means that we can formulate this speed scaling problem as follows (see also [55, Chapter 4]):

$$\begin{aligned} \text{SPEEDSCALING : } & \min_{s \in \mathbb{R}^n, B \in \mathbb{R}^n} \sum_{i \in \mathcal{N}} p(s_i) \frac{w_i}{s_i} \\ & \text{s.t. } B_i + \frac{w_i}{s_i} \leq D_i, \quad i \in \mathcal{N}, \\ & B_i \geq A_i, \quad i \in \mathcal{N}, \\ & B_i + \frac{w_i}{s_i} \leq B_{i+1}, \quad i \in \mathcal{N} \setminus \{n\}, \\ & 0 < s_i \leq s^{\max}, \quad i \in \mathcal{N}, \end{aligned}$$

where $s^{\max}$ is the maximum processor speed and p is a convex function that models the relation between processor speed and its energy usage. Note that we can impose a nonzero lower bound on each s_i so that the feasible set of this problem is guaranteed to be closed. Since each task cannot be processed longer than the maximum time that is available for it, we have that $\frac{w_i}{s_i} \leq D_i - A_i$. This yields a lower bound on s_i of $\frac{w_i}{D_i - A_i}$ that is nonzero since $w_i > 0$ and $D_i > A_i$.

If the processor shuts down only after the latest task deadline, even when all tasks have already been completed, there exists an optimal schedule with no idle time [89]. This means that we can add without loss of generality the constraint $B_i = A_i + \sum_{k=1}^{i-1} \frac{w_k}{s_k}$ for all $i \in \mathcal{N}$ to the formulation of Problem SPEEDSCALING. Defining $x_i := \frac{w_i}{s_i}$ for all $i \in \mathcal{N}$ and $q(x_i) := x_i p(1/x_i)$ (note that q is convex), it follows that

$$p(s_i) \frac{w_i}{s_i} = p\left(\frac{w_i}{x_i}\right) x_i = w_i q\left(\frac{x_i}{w_i}\right).$$

Using the lower bound on s_i , the added constraint on B_i , the transformation $x_i = \frac{w_i}{s_i}$, and the function q , we can reformulate Problem SPEEDSCALING to

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \sum_{i \in \mathcal{N}} w_i q\left(\frac{x_i}{w_i}\right) \\ \text{s.t.} \quad & A_{i+1} \leq \sum_{k=1}^i x_k \leq D_i, \quad i \in \mathcal{N} \setminus \{n\}, \\ & \sum_{i \in \mathcal{N}} x_i = D_n - A_1, \\ & \frac{w_i}{s_{\max}} \leq x_i \leq D_i - A_i, \quad i \in \mathcal{N}. \end{aligned}$$

This is an instance of $(w, 0, q)$ -S/NC/S. Hence, by Corollary 2.2, this problem and thus Problem SPEEDSCALING can be solved in $O(n \log n)$ time. This result also leads to complexity improvements for speed scaling problems that can be reduced to Problem SPEEDSCALING, e.g., for the multi-core processor scheduling problem considered in [57].

Recently, [153] applied the equivalence result in [127] to improve the time complexity of several other speed scaling problems. Together with the result in this section, this suggests that there is a great potential for using the reduction result in this chapter to contribute to more efficient algorithms within this research field.

2.6 CONCLUSIONS AND OUTLOOK

In this chapter, we discussed the resource allocation problem (RAP). We provided an overview of existing variants of and solution approaches and algorithms for RAPs. Moreover, we proved that RAPs with submodular constraints and a specific type of objective function, namely those that are (a, b, f) -separable, can be solved efficiently as RAPs with quadratic objective functions. Finally, we discussed several applications of the RAP to energy management problems and we have shown the impact of our reduction result on these problems.

One major direction for future research on this topic is the extension of the reduction result to other problems. Promising candidates for this are problems that are variations on the RAPs studied in this chapter. Two of such problems are the topic of further parts of this thesis. In Chapter 5, we study a RAP whose objective function includes additional nonseparable terms. This particular RAP can be used to model the problem of scheduling electric vehicle (EV) charging where we can use three-phase charging technology, i.e., energy can be drawn from all three phases of the low voltage network. In Section 8.3, we study a RAP with on-off constraints, i.e., the allowed allocated amount of resource to a given activity is either zero or lies within a given range not including zero. This particular RAP can be used to model the in practice relevant limitation of

EVs that charging at rates close to zero is often not possible due to technical limitation of the EV and/or the charging infrastructure.

Besides this more technical direction, in the light of a more cross-disciplinary approach towards the study of RAPs, it is worthwhile to identify more research fields and applications (next to the ones that we discussed in this chapter) where RAPs occur and where our results can have an impact and lead to new insights.

2.7 APPENDIX: PROOF THAT LAMINAR CONSTRAINTS ARE A SPECIAL CASE OF SUBMODULAR CONSTRAINTS

In this appendix, we show that laminar (or tree) constraints are a special case of submodular constraints. Recall that

- » laminar constraints are of the form $L_j \leq \sum_{i \in \mathcal{N}_j} x_i \leq U_j, j \in \mathcal{M}$, where the subsets $\mathcal{N}_1, \dots, \mathcal{N}_m$ of $\mathcal{N}$ have the property that either $\mathcal{N}_j \cap \mathcal{N}_\ell = \emptyset$, $\mathcal{N}_j \subset \mathcal{N}_\ell$, or $\mathcal{N}_j \supset \mathcal{N}_\ell$ for all $j, \ell \in \mathcal{M}$;
- » submodular constraints are of the form $\sum_{i \in \mathcal{S}} x_i \leq r(\mathcal{S}), \mathcal{S} \subset \mathcal{N}$ and $\sum_{i \in \mathcal{N}} x_i = r(\mathcal{N})$, where r is a submodular function.

To derive the stated result, we use a result from [48, 49] on so-called *cross-free* families of subsets. A family $\mathcal{F} \subseteq 2^{\mathcal{N}}$ is called cross-free if none of its elements cross, i.e., for any two subsets $\mathcal{X}, \mathcal{Y} \in \mathcal{F}$ we have that at least one of the sets $\mathcal{X} \cap \mathcal{Y}$, $\mathcal{X} \cap (\mathcal{N} \setminus \mathcal{Y})$, $(\mathcal{N} \setminus \mathcal{X}) \cap \mathcal{Y}$, or $(\mathcal{N} \setminus \mathcal{X}) \cap (\mathcal{N} \setminus \mathcal{Y})$ is empty. For a given cross-free family $\mathcal{F}$ containing $\emptyset$ and $\mathcal{N}$ and for any set function $r : \mathcal{F} \rightarrow \mathbb{R}$ with $r(\emptyset) = 0$, the set

$$\mathcal{B}(\mathcal{F}, r) := \left\{ x \in \mathbb{R}^n \mid \sum_{i \in \mathcal{X}} x_i \leq r(\mathcal{X}) \forall \mathcal{X} \in \mathcal{F}, \sum_{i \in \mathcal{N}} x_i = r(\mathcal{N}) \right\}$$

is a base polyhedron [48, 49]. This means that there exists a submodular function $r' : 2^{\mathcal{N}} \rightarrow \mathbb{R}$ such that

$$\mathcal{B}(\mathcal{F}, r) = \left\{ x \in \mathbb{R}^n \mid \sum_{i \in \mathcal{X}} x_i \leq r'(\mathcal{X}) \forall \mathcal{X} \subset \mathcal{N}, \sum_{i \in \mathcal{N}} x_i = r(\mathcal{N}) \right\}.$$

Thus, we can show that laminar constraints are a special case of submodular constraints if for a given feasible set $\mathcal{C}'$ determined by laminar constraints we can find a cross-free family $\mathcal{F}$ and a set function $r : \mathcal{F} \rightarrow \mathbb{R}$ such that $\mathcal{B}(\mathcal{F}, r) = \mathcal{C}'$.

For given laminar constraints $L_j \leq \sum_{i \in \mathcal{N}_j} x_i \leq U_j, j \in \mathcal{M}$ and a corresponding feasible set $\mathcal{C}' := \{x \in \mathbb{R}^n \mid L_j \leq \sum_{i \in \mathcal{N}_j} x_i \leq U_j, j \in \mathcal{M}\}$, we define the following family of subsets of $\mathcal{N}$:

$$\mathcal{N}' := \{\mathcal{N}_j \mid j \in \mathcal{M}\} \cup \{\mathcal{N} \setminus \mathcal{N}_j \mid j \in \mathcal{M}\}.$$

Note that the feasible set $\mathcal{C}'$ is equal to $\mathcal{B}(\mathcal{N}', r')$, where $r' : \mathcal{N}' \rightarrow \mathbb{R}$ is a set function on $\mathcal{N}'$ given by

$$r'(\mathcal{X}) := \begin{cases} U_j & \text{if } \mathcal{X} = \mathcal{N}_j \text{ for some } j \in \mathcal{M}, \\ R - L_j & \text{if } \mathcal{X} = \mathcal{N} \setminus \mathcal{N}_j \text{ for some } j \in \mathcal{M}. \end{cases}$$

We claim that $\mathcal{N}'$ is a cross-free family, which immediately implies that the set $\mathcal{B}(\mathcal{N}', r')$ is a base polyhedron and thus that laminar constraints are a special case of submodular constraints. For this, we consider for two given distinct sets $\mathcal{X}, \mathcal{Y} \in \mathcal{N}'$ four cases:

1. If $\mathcal{X} = \mathcal{N}_j$ and $\mathcal{Y} = \mathcal{N}_\ell$ for some $j, \ell \in \mathcal{M}$, then either $\mathcal{X} \cap \mathcal{Y} = \emptyset$, $\mathcal{X} \subset \mathcal{Y}$, or $\mathcal{X} \supset \mathcal{Y}$. The latter two cases imply that $\mathcal{X} \cap (\mathcal{N} \setminus \mathcal{Y}) = \emptyset$ and $(\mathcal{N} \setminus \mathcal{X}) \cap \mathcal{Y} = \emptyset$ respectively. Thus, in all three cases, $\mathcal{X}$ and $\mathcal{Y}$ do not cross.
2. If $\mathcal{X} = \mathcal{N}_j$ and $\mathcal{Y} = \mathcal{N} \setminus \mathcal{N}_\ell$ for some $j, \ell \in \mathcal{M}$, then either $\mathcal{X} \cap (\mathcal{N} \setminus \mathcal{Y}) = \emptyset$, $\mathcal{X} \subset (\mathcal{N} \setminus \mathcal{Y})$, or $\mathcal{X} \supset (\mathcal{N} \setminus \mathcal{Y})$. The latter two cases imply that $\mathcal{X} \cap \mathcal{Y} = \emptyset$ and $(\mathcal{N} \setminus \mathcal{X}) \cap (\mathcal{N} \setminus \mathcal{Y}) = \emptyset$ respectively. Thus, in all three cases, $\mathcal{X}$ and $\mathcal{Y}$ do not cross.
3. If $\mathcal{X} = \mathcal{N} \setminus \mathcal{N}_j$ and $\mathcal{Y} = \mathcal{N}_\ell$ for some $j, \ell \in \mathcal{M}$, we can use the argument in case 2 with the roles of $\mathcal{X}$ and $\mathcal{Y}$ interchanged to conclude that $\mathcal{X}$ and $\mathcal{Y}$ do not cross.
4. If $\mathcal{X} = \mathcal{N} \setminus \mathcal{N}_j$ and $\mathcal{Y} = \mathcal{N} \setminus \mathcal{N}_\ell$ for some $j, \ell \in \mathcal{M}$, then either $(\mathcal{N} \setminus \mathcal{X}) \cap (\mathcal{N} \setminus \mathcal{Y}) = \emptyset$, $(\mathcal{N} \setminus \mathcal{X}) \subset (\mathcal{N} \setminus \mathcal{Y})$, or $(\mathcal{N} \setminus \mathcal{X}) \supset (\mathcal{N} \setminus \mathcal{Y})$. The latter two cases imply that $(\mathcal{N} \setminus \mathcal{X}) \cap \mathcal{Y} = \emptyset$ and $\mathcal{X} \cap (\mathcal{N} \setminus \mathcal{Y}) = \emptyset$ respectively. Thus, in all three cases, $\mathcal{X}$ and $\mathcal{Y}$ do not cross.

2.8 APPENDIX: PROOF THAT CONDITION 2.1 HOLDS FOR THE RAP WITH LAMINAR CONSTRAINTS

Here we present an alternative proof of the claim that Condition 2.1 holds for the resource allocation problem with laminar constraints (S/LC/ γ). Before we prove this result in Lemma 2.3, we first show that the difference between any two feasible solutions x and z of S/LC/ γ can be written as a nonnegative combination of vectors in the set $\mathcal{V} := \{e^k - e^i \mid (i, k) \in \mathcal{E}_{\mathcal{C}'}(x)\}$, where $\mathcal{C}'$ is the feasible set of S/LC/ γ . In other words, $z - x$ belongs to the cone generated by the vectors in $\mathcal{V}$. To this end, we present the following procedure to obtain this combination. Starting from the solution $\bar{x}^0 := x$, we construct a series of intermediate vectors $(\bar{x}^t)_{t \geq 0}$ that finally leads to z by iteratively transferring amounts between two variables. We do this in such a way that the distance $\sum_{i \in \mathcal{N}} |z_i - \bar{x}_i^t|$ reduces as t increases and becomes zero for some $\bar{t} \geq 0$, meaning that $\bar{x}^{\bar{t}} = z$. To ensure finiteness of this process, we always choose two variables with indices i, k such that $\bar{x}_i^t > z_i$ and $\bar{x}_k^t < z_k$. By transferring an amount of $\lambda_{ik} := \min(\bar{x}_i^t - z_i, z_k - \bar{x}_k^t)$ between those variables, we have for the subsequent

vector $\bar{x}^{t+1}$ that either $\bar{x}_i^{t+1} = z_i$ or $\bar{x}_k^{t+1} = z_k$. By repeating this process, we finally reach an intermediate vector $\bar{x}^t$ that equals z . For each selected pair (i, k) , the value λ_{ik} represents a positive coefficient in the desired conic combination.

To ensure that each index pair with a positive coefficient is an exchangeable pair (see also Lemma 2.2), i.e., is in $\mathcal{E}_{C'}(x)$, we restrict the choice of index pair in the procedure in the following way. First, we order the subsets such that $\mathcal{N}_j \subset \mathcal{N}_{j'}$ implies $j > j'$ for all $j, j' \in \mathcal{M}$. Moreover, we define $\mathcal{N}_0 := \mathcal{N}$. Now we iterate through the subsets from $\mathcal{N}_m$ to $\mathcal{N}_0$ and during iteration j we allow only exchanges between variables whose indices belong to the current subset $\mathcal{N}_j$.

The procedure is summarized in Algorithm 2.1. In this algorithm, for any $j \in \{0\} \cup \mathcal{M}$, t_j is the last iteration index such that no exchanges are allowed between a variable whose index is in $\mathcal{N}_j$ and a variable whose index is not in $\mathcal{N}_j$.

Algorithm 2.1: Computing $z - x$ as a conic combination of vectors in $\mathcal{V}$.

```

1 Input: Two feasible solutions  $x, z$  to S/LC/ $\gamma$ 
2 Output: Weight matrix  $\lambda \in \mathbb{R}_{\geq 0}^{n \times n}$ 
3 Initialize  $\lambda_{ik} = 0$  for all  $i, k \in \mathcal{N}$ 
4 Order subsets such that  $\mathcal{N}_j \subset \mathcal{N}_{j'}$  implies  $j > j'$  for all  $j, j' \in \mathcal{M}$ 
5  $\mathcal{N}_0 := \mathcal{N}; t = 0; \bar{x}^0 := x$ 
6 for  $j = m$  down to 0 do
7   while there exist  $i, k \in \mathcal{N}_j$  such that  $\bar{x}_i^t > z_i$  and  $\bar{x}_k^t < z_k$  do
8      $\lambda_{ik} := \min(\bar{x}_i^t - z_i, z_k - \bar{x}_k^t)$ 
9      $\bar{x}^{t+1} := \bar{x}^t + \lambda_{ik}(e^k - e^i)$ 
10     $t = t + 1$ 
11   $t_j = t$ 
12 return  $\lambda$ 

```

In Lemma 2.1, we prove several properties of the output λ of the algorithm and of the intermediate vectors $(\bar{x}^t)_{t \geq 0}$.

Lemma 2.1. *The following statements hold for the output λ and the sequence of intermediate vectors $(\bar{x}^t)_{t \geq 0}$ of Algorithm 2.1 when applied to two feasible solutions x and z to S/LC/ γ :*

1. $\sum_{i \in \mathcal{N}} \bar{x}_i^t = R$ for all $t \geq 0$.
2. If $\bar{x}_i^t > z_i$ for a given $t \geq 0$, then $z_i \leq \bar{x}_i^{t'} \leq \bar{x}_i^t \leq x_i$ for all $t' > t$;
3. If $\bar{x}_i^t < z_i$ for a given $t \geq 0$, then $z_i \geq \bar{x}_i^{t'} \geq \bar{x}_i^t \geq x_i$ for all $t' > t$;
4. If $\bar{x}_i^t = z_i$ for a given $t \geq 0$, then $\bar{x}_i^{t'} = z_i$ for all $t' > t$.
5. For a given j and $t \geq t_j$, we have that either $\bar{x}_i^t \leq z_i$ for all $i \in \mathcal{N}_j$ or $\bar{x}_i^t \geq z_i$ for all $i \in \mathcal{N}_j$.

6. Each index pair $(i, k) \in \mathcal{N}^2$ is selected at most once over the entire course of the algorithm.
7. For a given j and any $t \leq t_j$, it holds that $\sum_{\ell \in \mathcal{N}_j} \bar{x}_\ell^t = \sum_{\ell \in \mathcal{N}_j} x_\ell$.
8. $z - x = \sum_{(i,k) \in \mathcal{N}^2} \lambda_{ik} (e^k - e^i)$.

Proof. Part (1): Follows by induction on t since $\sum_{\ell \in \mathcal{N}} \bar{x}_\ell^{t+1} = \sum_{\ell \in \mathcal{N}} \bar{x}_\ell^t + \lambda_{ik} - \lambda_{ik} = \sum_{\ell \in \mathcal{N}} \bar{x}_\ell^t$ for all $t \geq 0$ and $\bar{x}^0 = x$.

Part (2): For a given $t \geq 0$, we have that $\bar{x}_i^t > z_i$ implies that either $\bar{x}_i^{t+1} = \bar{x}_i^t$ (if i is not selected during iteration t) or $z_i \leq \bar{x}_i^{t+1} < \bar{x}_i^t$ (if i is selected during iteration t). Thus, we have that $\bar{x}_i^t > z_i$ implies that $z_i \leq \bar{x}_i^{t+1} \leq \bar{x}_i^t$. By induction, one can deduce that if $x_i^t > z_i$, then $z_i \leq \bar{x}_i^{t'} \leq \bar{x}_i^t \leq x_i$ for all $t \geq 0$ and $t' > t$.

Part (3): Is analogous to the proof of Part (2).

Part (4): If $x_i^t = z_i$, then i will not be selected anymore as part of an exchangeable pair. Hence, $x_i^t = x_i^{t+1} = \dots = z_i$.

Part (5): By definition of t_j , we have that either $\bar{x}_\ell^{t_j} \geq z_\ell$ for all $\ell \in \mathcal{N}_j$ or $\bar{x}_\ell^{t_j} \leq z_\ell$ for all $\ell \in \mathcal{N}_j$. It follows directly from Parts (2)-(4) that in the first case $\bar{x}_\ell^t \geq z_\ell$ for all $\ell \in \mathcal{N}_j$ and that in the second case $\bar{x}_\ell^t \leq z_\ell$ for all $\ell \in \mathcal{N}_j$.

Part (6): If the pair (i, k) is chosen during some iteration t , then either $\bar{x}_i^{t+1} = z_i$ or $\bar{x}_k^{t+1} = z_k$. Thus, at least one of the indices i, k cannot be chosen again as part of a pair, hence the pair (i, k) is selected at most once.

Part (7): For a given $t \leq t_j$, let (i, k) denote the selected pair during iteration $t-1$. Thus, there is a subset $\mathcal{N}_{j'}$ with $j' > j$ such that $i, k \in \mathcal{N}_{j'}$. By the ordering of the subsets, we have either $\mathcal{N}_j \cap \mathcal{N}_{j'} = \emptyset$ or $\mathcal{N}_{j'} \subset \mathcal{N}_j$. Thus, either both or neither of the indices i and k are in $\mathcal{N}_j$. This implies that $\sum_{\ell \in \mathcal{N}_j} \bar{x}_\ell^t = \sum_{\ell \in \mathcal{N}_j} \bar{x}_\ell^{t-1}$. By induction on t , it follows that $\sum_{\ell \in \mathcal{N}_j} \bar{x}_\ell^t = \sum_{\ell \in \mathcal{N}_j} \bar{x}_\ell^0 = \sum_{\ell \in \mathcal{N}_j} x_\ell$.

Part (8): Follows from Part (6) and the fact that $\lambda_{ik} = 0$ if the pair (i, k) has not been chosen during any iteration. $\square$

Lemma 2.1 implies that for any two feasible solutions x and z , the difference $z - x$ can be written as a nonnegative combination of the vectors in $\{e^k - e^i\}_{(i,k) \in \mathcal{N}^2}$. We strengthen this result in Lemma 2.2 by proving that $z - x$ can be written as a nonnegative combination of the vectors in $\mathcal{V}$.

Lemma 2.2. *Let λ and $(\bar{x}^t)_{t \geq 0}$ be the output of Algorithm 2.1 applied to two feasible solutions x and z of the problem $S/LC/\gamma$. If $\lambda_{ik} > 0$ for a given pair $(i, k) \in \mathcal{N}^2$, then $(i, k) \in \mathcal{E}_{C'}(x)$ and $\lambda_{\ell,i} = \lambda_{k,\ell} = 0$ for all $\ell \in \mathcal{N}$.*

Proof. Note that for any two indices $i, k \in \mathcal{N}$, the solution $x + \epsilon(e^k - e^i)$ is feasible for some $\epsilon > 0$ if and only if we have for each subset $\mathcal{N}_j$ that contains i but not k that $\sum_{\ell \in \mathcal{N}_j} x_\ell > L_j$, and for each subset $\mathcal{N}_{j'}$ that contains k but not i that $\sum_{\ell \in \mathcal{N}_{j'}} x_\ell < U_{j'}$. Let $\mathcal{N}_{j'}$ be the minimal subset that contains both i and k , i.e., there is no other subset $\mathcal{N}_j$ such that $\mathcal{N}_j \subset \mathcal{N}_{j'}$ and $i, k \in \mathcal{N}_j$. If $\lambda_{ik} > 0$, then there exists $t_{j'+1} < t \leq t_{j'}$ such that the pair (i, k) has been selected during iteration t . Thus, $\bar{x}_i^t > \bar{x}_i^{t+1} \geq z_i$ and $\bar{x}_k^t < \bar{x}_k^{t+1} \leq z_k$. By Parts (2) and (3) of Lemma 2.1, this means that $x_i > z_i$ and $x_k < z_k$ and that $\bar{x}_i^t \geq z_i$ and $\bar{x}_k^t \leq z_k$ for all $t \geq 0$. By Part (5) of Lemma 2.1, this means that for any subset $\mathcal{N}_j$ that contains i but not k we have that $\bar{x}_\ell^{t_j} \geq z_\ell$ for all $\ell \in \mathcal{N}_j$ since $j > j'$. In particular, we have by Part (2) that $\bar{x}_i^{t_j} > z_i$ since $\bar{x}_i^t > z_i$. It follows from feasibility of z and Part (7) that $L_j \leq \sum_{\ell \in \mathcal{N}_j} z_\ell < \sum_{\ell \in \mathcal{N}_j} \bar{x}_\ell^{t_j} = \sum_{\ell \in \mathcal{N}_j} x_\ell$. Analogously, we can show that $U_j > \sum_{\ell \in \mathcal{N}_j} x_\ell$. Thus, the solution $x + \epsilon(e^k - e^i)$ is feasible for some $\epsilon > 0$, hence $(i, k) \in \mathcal{E}_{C'}(x)$.

Note that for any $\ell \in \mathcal{N}$, we can only have that $\lambda_{\ell i} > 0$ if there is some iteration t with $\bar{x}_i^t < z_i$. Since $\bar{x}_i^t \geq z_i$ for all $t \geq 0$, we must have that $\lambda_{\ell i} = 0$. Analogously, we must have that $\lambda_{k\ell} = 0$ since $\bar{x}_k^t \leq z_k$ for all $t \geq 0$. $\square$

Lemma 2.2 implies that we can partition $\mathcal{N}$ into three subsets such that one subset contains all indices i for which $\lambda_{ik} > 0$ for at least one $k \in \mathcal{N}$, one subset contains all indices i for which $\lambda_{ki} > 0$ for at least one $k \in \mathcal{N}$, and one subset contains all indices i such that $\lambda_{ik} = \lambda_{ki} = 0$ for all $k \in \mathcal{N}$. More precisely, we define the following partition of $\mathcal{N}$:

$$\begin{aligned}\mathcal{L}(x) &:= \{i \in \mathcal{N} \mid \lambda_{ik} > 0 \text{ for some } k \in \mathcal{N}\}, \\ \mathcal{U}(x) &:= \{i \in \mathcal{N} \mid \lambda_{ki} > 0 \text{ for some } k \in \mathcal{N}\}, \\ \mathcal{F}(x) &:= \mathcal{N} \setminus (\mathcal{L}(x) \cup \mathcal{U}(x)) = \{i \in \mathcal{N} \mid \lambda_{ik} = \lambda_{ki} = 0 \text{ for all } k \in \mathcal{N}\}.\end{aligned}$$

Using this partition and Lemma 2.2, we can show that S/LC/ γ satisfies Condition 2.1.

Lemma 2.3. *For $\gamma \in \{C, I\}$, the problem S/LC/ γ satisfies Condition 2.1.*

Proof. First, we prove the “only if”-part. Suppose x is optimal for S/LC/ γ and there exists an index pair $(i, k) \in \mathcal{E}_{C'}(x)$ such that $\phi_k^+(x_k) < \phi_i^-(x_i)$. We consider the two cases $\gamma = C$ and $\gamma = I$ separately and show for both cases that there exists a feasible solution whose objective value is smaller than that of x . Thus, x cannot be optimal, which is a contradiction. It follows that $\phi_k^+(x_k) \geq \phi_i^-(x_i)$ for all $(i, k) \in \mathcal{E}_{C'}(x)$:

$\gamma = C$: By definition of $\mathcal{E}_{C'}(x)$ and the left and right derivatives ϕ_k^- and ϕ_i^+ , there exists $\epsilon > 0$ such that $x + \epsilon(e^k - e^i)$ is feasible and $\phi_k^+(x_k + \epsilon) < \phi_i^-(x_i - \epsilon)$. Thus, it follows from convexity of ϕ_i and ϕ_k that

$$\phi_i(x_i) - \phi_i(x_i - \epsilon) \geq \phi_i^-(x_i - \epsilon)\epsilon > \phi_k^+(x_k + \epsilon)\epsilon \geq \phi_k(x_k + \epsilon) - \phi_k(x_k).$$

This implies that

$$\phi_k(x_k + \epsilon) + \phi_i(x_i - \epsilon) < \phi_k(x_k) + \phi_i(x_i),$$

and thus that the objective value of $x + \epsilon(e^k - e^i)$ is smaller than that of x .

$\gamma = \mathbf{I}$: Recall from Section 2.2.3 that we may assume without loss of generality that the functions ϕ_i and ϕ_k are piece-wise linear with breakpoints at the integers. This means in particular that $\phi_i^-(x_i) = \phi_i(x_i) - \phi_i(x_i - 1)$ and $\phi_k^+(x_k) = \phi_k(x_k + 1) - \phi_k(x_k)$. It follows that

$$\phi_i(x_i) - \phi_i(x_i - 1) = \phi_i^-(x_i) > \phi_k^+(x_k) = \phi_k(x_k + 1) - \phi_k(x_k),$$

which implies that

$$\phi_k(x_k + 1) + \phi_i(x_i - 1) < \phi_k(x_k) + \phi_i(x_i).$$

Thus, the objective value of $x + (e^k - e^i)$ is smaller than that of x . Finally, by definition of $\mathcal{E}_{C'}(x)$, we have that $x + (e^k - e^i)$ is feasible, i.e., we may choose $\epsilon = 1$ in this definition given that $(i, k) \in \mathcal{E}_{C'}(x)$.

Second, we prove the “if”-part. Let x be a feasible solution such that $\phi_k^+(x_k) \geq \phi_i^-(x_i)$ for all $(i, k) \in \mathcal{E}_{C'}(x)$ and let z be an arbitrary feasible solution. Moreover, let $\lambda \in \mathbb{R}^{n \times n}$ denote the output of Algorithm 2.1 when applied to x and z . By Lemma 2.2 and definition of the sets $\mathcal{L}(x)$, $\mathcal{U}(x)$, and $\mathcal{F}(x)$, we have that

$$\begin{aligned} z - x &= \sum_{(i,k) \in \mathcal{N}^2} \lambda_{ik}(e^k - e^i) = \sum_{(i,k) \in \mathcal{E}_{C'}(x)} \lambda_{ik}(e^k - e^i) \\ &= \sum_{\substack{(i,k) \in \mathcal{E}_{C'}(x), \\ i \in \mathcal{L}(x)}} \lambda_{ik}(e^k - e^i) = \sum_{\substack{(i,k) \in \mathcal{E}_{C'}(x), \\ i \in \mathcal{L}(x), \\ k \in \mathcal{U}(x)}} \lambda_{ik}(e^k - e^i). \end{aligned}$$

We define the following subgradient $g \in \mathbb{R}^n$ at the solution x :

$$g_i \begin{cases} := \phi_i^-(x_i) & \text{if } i \in \mathcal{L}(x), \\ := \phi_i^+(x_i) & \text{if } i \in \mathcal{U}(x), \\ \in [\phi_i^-(x_i), \phi_i^+(x_i)] & \text{if } i \in \mathcal{F}(x). \end{cases}$$

By convexity of the functions ϕ_i , it follows that

$$\begin{aligned} \sum_{i \in \mathcal{N}} (\phi_i(z_i) - \phi_i(x_i)) &\geq g^\top(z - x) = \sum_{\substack{(i,k) \in \mathcal{E}_{C'}(x), \\ i \in \mathcal{L}(x), \\ k \in \mathcal{U}(x)}} \lambda_{ik} g^\top(e^k - e^i) \\ &= \sum_{\substack{(i,k) \in \mathcal{E}_{C'}(x), \\ i \in \mathcal{L}(x), \\ k \in \mathcal{U}(x)}} \lambda_{ik} (\phi_k^+(x_k) - \phi_i^-(x_i)) \geq 0. \end{aligned}$$

It follows that x is optimal since z is an arbitrary feasible solution. $\square$

3

QUADRATIC RESOURCE ALLOCATION

ABSTRACT – In this chapter, we focus on the continuous quadratic resource allocation problem with box constraints. This problem returns in several chapters in this thesis either as a subproblem within studied extensions of the problem (Chapters 4, 5, and 8) or as an example problem to illustrate online optimization and prediction methodologies (Chapters 7 and 8). We provide an analysis of and solution approach for the problem based on Lagrangian relaxation and discuss two algorithms. These algorithms fall into the class of so-called breakpoint search algorithms, which form an important building block for the presented algorithms for extensions of the problem studied in Chapters 4, 5, and 8.

3.1 INTRODUCTION

The focus of this chapter is on the resource allocation problem (RAP) with a quadratic objective function, subject to additional box constraints, and involving continuous variables. In terms of the classification scheme introduced in Section 2.2.3, this is the problem $(a, 0)$ -Q/Box/C, where $a \in \mathbb{R}_{>0}$. For convenience in this and later chapters, we call this problem the quadratic resource allocation problem (Problem QRA).

We present a solution approach for Problem QRA based on Lagrangian relaxation and so-called breakpoint search. Hereby, one exploits the fact that an optimal solution to Problem QRA can be characterized by a single value called the *optimal Lagrange multiplier*. In the breakpoint search approach, one searches for an optimal Lagrange multiplier and uses this value to reconstruct the optimal (primal) solution. The search for this value can be done structurally due to the monotonic relation between the Lagrange multiplier and the resulting primal solution, i.e., the solution to the Lagrangian relaxation. As a consequence, the

This chapter contains parts of [SU:2].

multiplier search can be reduced to searching through a finite set of candidate values called breakpoints. We discuss two algorithms for this searching problem that each use a different strategy to obtain the optimal Lagrange multiplier. The first algorithm sorts the candidate values and searches them sequentially, whereas the second algorithm applies binary search and uses a sophisticated bookkeeping scheme to reduce the computational overhead. An overview of the pros and cons of these two algorithms is given in Table 3.1: we discuss them in more detail later in this chapter.

	Sequential search	Binary search
Time complexity	$O(n \log n)$	$O(n)$
Implementation	Requires simple bookkeeping scheme	Requires sophisticated bookkeeping scheme and repeated median-finding
Performance	Good: sorting is the heaviest operation	Depends on efficiency of bookkeeping and median-finding implementation

Table 3.1: High-level differences between the sequential search- and binary search-based algorithms for Problem QRA.

The outline of the remainder of this chapter is as follows. In Section 3.2, we formulate the considered Problem QRA and derive a general solution approach for it based on breakpoint search. Subsequently, in Section 3.3, we discuss the two mentioned breakpoint search algorithms in Table 3.1. In Section 3.4, we give an example of an application of this model in the context of electric vehicle (EV) charging. This example forms the basis of several other occurrences of RAPs within energy management, in particular those in Chapters 5, 7, and 8. Finally, in Section 3.5, we discuss the relevance of Problem QRA and the EV charging problem introduced in Section 3.4 for later chapters of this thesis.

3.2 PROBLEM FORMULATION, ANALYSIS, AND GENERAL SOLUTION APPROACH

Problem QRA is given as follows:

$$\begin{aligned}
 \text{QRA : } \min_{x \in \mathbb{R}^n} \quad & \sum_{i \in \mathcal{N}} \frac{1}{2} \frac{x_i^2}{a_i} \\
 \text{s.t.} \quad & \sum_{i \in \mathcal{N}} x_i = R, \\
 & l_i \leq x_i \leq u_i, \quad i \in \mathcal{N},
 \end{aligned} \tag{3.1}$$

where $a \in \mathbb{R}_{>0}^n$, $R \in \mathbb{R}$, $l, u \in \mathbb{R}^n$, and $\mathcal{N} := \{1, \dots, n\}$. The so-called Lagrangian relaxation of this problem, where the resource constraint (3.1) is

replaced by a penalty term in the objective function, is given by:

$$\begin{aligned} \text{QRA}(\lambda) : \min_{x \in \mathbb{R}^n} \quad & \sum_{i \in \mathcal{N}} \left(\frac{1}{2} \frac{x_i^2}{a_i} - \lambda x_i \right) \\ \text{s.t.} \quad & l_i \leq x_i \leq u_i, \quad i \in \mathcal{N}, \end{aligned}$$

where $\lambda \in \mathbb{R}$ is the Lagrange multiplier corresponding to the resource constraint (3.1). We denote the optimal solution to this problem by $x(\lambda) := (x_i(\lambda))_{i \in \mathcal{N}}$. Since the objective function of this problem is separable, the optimal solution to $\text{QRA}(\lambda)$ is given by

$$x_i(\lambda) = \begin{cases} l_i & \text{if } \lambda < \frac{l_i}{a_i}, \\ a_i \lambda & \text{if } \frac{l_i}{a_i} \leq \lambda < \frac{u_i}{a_i}, \\ u_i & \text{if } \frac{u_i}{a_i} \leq \lambda. \end{cases} \quad (3.2)$$

Observe that $x_i(\lambda)$ is a continuous piece-wise linear non-decreasing function of λ . More precisely, $x_i(\lambda)$ is constant for $\lambda \leq \frac{l_i}{a_i}$, linear with slope a_i for $\lambda \in [\frac{l_i}{a_i}, \frac{u_i}{a_i}]$, and again constant for $\lambda \geq \frac{u_i}{a_i}$ (see also Figure 3.1). For each $i \in \mathcal{N}$, we call the points where $x_i(\lambda)$ has “kinks”, i.e., where $x_i(\lambda)$ is non-differentiable, the *breakpoints* of $x_i(\lambda)$. We denote these breakpoints for $i \in \mathcal{N}$ by α_i and β_i respectively, i.e., $\alpha_i := \frac{l_i}{a_i}$ and $\beta_i := \frac{u_i}{a_i}$, where we refer to α_i as the *lower* breakpoint of $x_i(\lambda)$ and to β_i as the *upper* breakpoint of $x_i(\lambda)$. We denote the multiset of lower breakpoints by $\mathcal{A} := \{\alpha_i \mid i \in \mathcal{N}\}$ and the multiset of upper breakpoints by $\mathcal{B} := \{\beta_i \mid i \in \mathcal{N}\}$. The reason for defining $\mathcal{A}$ and $\mathcal{B}$ as *multisets* is because this allows us to readily associate each breakpoint value in the set with one index in $\mathcal{N}$.

Note that also the sum $z(\lambda) := \sum_{i \in \mathcal{N}} x_i(\lambda)$ is continuous, piece-wise linear, and non-decreasing. Moreover, it has $2n$ breakpoints, namely those of all terms $x_i(\lambda)$. Thus, the multiset of breakpoints of $z(\lambda)$ is given by $\mathcal{A} \cup \mathcal{B}$.

Figure 3.1: The function $x_i(\lambda)$ for a given $i \in \mathcal{N}$. The slope of the line segment for $\lambda \in [\frac{l_i}{a_i}, \frac{u_i}{a_i}]$ is a_i .

By definition of $z(\lambda)$, feasibility of the original Problem QRA implies that there exists a value λ^* of the Lagrange multiplier such that $z(\lambda^*) = R$. As shown in the following lemma, this property ensures that $x(\lambda^*)$ is not only an optimal solution to Problem QRA(λ^*) but also to the original Problem QRA:

Lemma 3.1. *If for a given $\lambda^* \in \mathbb{R}$ we have $z(\lambda^*) = R$, the solution $x(\lambda^*)$ is optimal for both QRA(λ^*) and Problem QRA.*

Proof. Suppose that $x(\lambda^*)$ is not optimal for Problem QRA, i.e., there exists an optimal solution $\bar{x}$ to Problem QRA whose objective value is lower than that of $x(\lambda^*)$. Since $x(\lambda^*)$ is optimal for QRA(λ^*), we have

$$\begin{aligned} \sum_{i \in \mathcal{N}} \left(\frac{1}{2} \frac{x_i(\lambda^*)^2}{a_i} - \lambda^* x_i(\lambda^*) \right) &\leq \sum_{i \in \mathcal{N}} \left(\frac{1}{2} \frac{\bar{x}_i^2}{a_i} - \lambda^* \bar{x}_i \right) \\ &< \sum_{i \in \mathcal{N}} \left(\frac{1}{2} \frac{x_i(\lambda^*)^2}{a_i} - \lambda^* \bar{x}_i \right). \end{aligned}$$

It follows that $-\lambda^* \sum_{i \in \mathcal{N}} x_i(\lambda^*) < -\lambda^* \sum_{i \in \mathcal{N}} \bar{x}_i$. However, since both $x(\lambda^*)$ and $\bar{x}$ are feasible for Problem QRA and thus $\sum_{i \in \mathcal{N}} x_i(\lambda^*) = \sum_{i \in \mathcal{N}} \bar{x}_i = R$, this implies that $-\lambda^* R < -\lambda^* R$. This is a contradiction, hence $x(\lambda^*)$ must also be optimal for Problem QRA. $\square$

We call λ^* the *optimal Lagrange multiplier* for Problem QRA. Note that this multiplier is not necessarily unique: in general, there may exist an interval $I \subset \mathbb{R}$ such that $\lambda \in I$ implies $z(\lambda) = R$. However, Lemma 3.1 implies that we can easily compute an optimal solution to Problem QRA using Equation (3.2) when we have found *any* optimal Lagrange multiplier in this interval I . Based on this, in the remainder of this chapter, we assume without loss of generality that the optimal Lagrange multiplier λ^* is unique.

In the following, we describe our two-step approach to find the optimal Lagrange multiplier λ^* . First, we aim to find two consecutive breakpoints λ_1 and λ_2 such that $\lambda_1 \leq \lambda^* \leq \lambda_2$. Since z is non-decreasing, this is equivalent to finding two consecutive breakpoints λ_1 and λ_2 such that $z(\lambda_1) \leq R \leq z(\lambda_2)$. In the second step, given these consecutive breakpoints, we determine λ^* and $x(\lambda^*)$. Note that, since $x(\lambda)$ is non-decreasing, we have for each $i \in \mathcal{N}$:

- » $x_i(\lambda^*) = l_i$ if and only if $x_i(\lambda_2) = l_i$, and
- » $x_i(\lambda^*) = u_i$ if and only if $x_i(\lambda_1) = u_i$.

Thus, given λ_1 and λ_2 , we know whether a given variable $x_i(\lambda^*)$ equals its lower bound l_i , its upper bound u_i , or is strictly in between these bounds. To find $x_i(\lambda^*)$ for the remaining variables that are strictly in between their bounds, note that, by definition of $x(\lambda)$,

$$R = z(\lambda^*) = \sum_{i: x_i(\lambda^*)=l_i} l_i + \sum_{i: l_i < x_i(\lambda^*) < u_i} a_i \lambda^* + \sum_{i: x_i(\lambda^*)=u_i} u_i.$$

It follows that

$$\lambda^* = \frac{R - \sum_{i: x_i(\lambda^*)=l_i} l_i - \sum_{i: x_i(\lambda^*)=u_i} u_i}{\sum_{i: l_i < x_i(\lambda^*) < u_i} a_i},$$

from which we can directly compute $x_i(\lambda^*)$ by $x_i(\lambda^*) = a_i \lambda^*$.

It remains to find the consecutive breakpoints λ_1 and λ_2 with $\lambda^* \in [\lambda_1, \lambda_2]$. In the next section, we discuss two strategies for this, which lead to two different algorithms for solving Problem QRA. The first strategy consists of searching the set $\mathcal{A} \cup \mathcal{B}$ of breakpoints sequentially, whereas the second strategy consists of a binary search.

3.3 TWO BREAKPOINT SEARCH ALGORITHMS

In this section, we discuss two algorithms for Problem QRA, which are based on the existing sequential breakpoint search strategy in [72] (Section 3.3.1, Algorithm 3.1) and the existing binary breakpoint search strategy in [100] (Section 3.3.2, Algorithm 3.2).

3.3.1 AN $O(n \log n)$ TIME ALGORITHM BASED ON SEQUENTIAL BREAKPOINT SEARCH

In the first strategy, we consider all breakpoints in $\mathcal{A} \cup \mathcal{B}$ in non-decreasing order until we have found the first, i.e., smallest, breakpoint λ such that $\lambda^* < \lambda$. In detail, for each candidate breakpoint λ , we compute $z(\lambda)$ and if $z(\lambda) > R$ for the first time, we set $\lambda_2 := \lambda$ and λ_1 as the previously considered breakpoint. To compute $z(\lambda)$ efficiently, we keep track of the sums

$$P(\lambda) := \sum_{i: \lambda < \frac{l_i}{a_i}} l_i + \sum_{i: \lambda \geq \frac{u_i}{a_i}} u_i, \quad Q(\lambda) := \sum_{i: \frac{l_i}{a_i} \leq \lambda < \frac{u_i}{a_i}} a_i$$

and update these values after a new breakpoint has been considered (see Table 3.2).

Type of λ	Update $P(\lambda)$	Update $Q(\lambda)$
$\lambda \equiv \alpha_i$	$P(\lambda) - l_i$	$Q(\lambda) + a_i$
$\lambda \equiv \beta_i$	$P(\lambda) + u_i$	$Q(\lambda) - a_i$

Table 3.2: Updating the bookkeeping sums $P(\lambda)$ and $Q(\lambda)$ when searching the breakpoints in non-decreasing order.

Algorithm 3.1 summarizes the sketched strategy. To efficiently compute the minimum breakpoint λ_k , we can implement the multisets $\mathcal{A}$ and $\mathcal{B}$ as sorted lists. As a consequence, each iteration of the repeat-statement takes $O(1)$ time.

Since the maximum number of iterations is $2n$ (one for each breakpoint), the overall complexity of this approach is $O(n \log n)$ due to the initial sorting of the breakpoints. If this sorting has already been given (for example if the breakpoints have already been sorted in a previous run of the algorithm), the time complexity of the algorithm reduces to $O(n)$.

Algorithm 3.1: An $O(n \log n)$ time algorithm for Problem QRA.

```

1 Input: Parameters  $a \in \mathbb{R}_{>0}^n$ ,  $R \in \mathbb{R}$ , and  $l, u \in \mathbb{R}^n$ 
2 Output: Optimal solution  $\bar{x}$  to Problem QRA
3 Compute the breakpoint multisets  $\mathcal{A} := \{\alpha_i \mid i \in \mathcal{N}\}$  and
    $\mathcal{B} := \{\beta_i \mid i \in \mathcal{N}\}$ 
4 Initialize  $P := \sum_{i \in \mathcal{N}} l_i$ ;  $Q := 0$ 
5 repeat
6   Determine smallest breakpoint  $\lambda_i := \min(\mathcal{A} \cup \mathcal{B})$ 
7   if  $P + Q\lambda_i = R$  then                                     ▷  $\lambda^* = \lambda_i$ 
8     Compute  $x(\lambda^*)$  using Equation (3.2)
9     return  $\bar{x} := x(\lambda^*)$ 
10  else if  $P + Q\lambda_i > R$  then                                   ▷  $\lambda^* < \lambda_i$ 
11     $\lambda^* = \frac{R-P}{Q}$ ; compute  $x(\lambda^*)$  using Equation (3.2)
12    return  $\bar{x} := x(\lambda^*)$ 
13  else                                                         ▷  $\lambda^* > \lambda_i$ 
14    if  $\lambda_i$  is lower breakpoint ( $\lambda_i = \alpha_i$ ) then
15       $P := P - l_i$ ;  $Q := Q + a_i$ 
16       $\mathcal{A} := \mathcal{A} \setminus \{\alpha_i\}$ 
17    else
18       $P := P + u_i$ ;  $Q := Q - a_i$ 
19       $\mathcal{B} := \mathcal{B} \setminus \{\beta_i\}$ 

```

We conclude this subsection with two observations that are crucial for the problems studied in Chapters 4 and 8 and the resulting solution approaches and algorithms:

1. Instead of searching the breakpoints in non-decreasing order, we may also search them in *non-increasing order* and continue the search until we find the first, i.e., largest breakpoint λ_1 such that $\lambda_1 < \lambda^*$.
2. Solving two instances of Problem QRA that differ only in the value of R in the resource constraint (3.1) can be done sequentially in one run of Algorithm 3.1. This is because the multisets of the breakpoints for these two instances of Problem QRA are the same. Thus, we can modify Algorithm 3.1 such that it continues the breakpoint search after the optimal multiplier for the smallest of the given values of R has been found. Note that, essentially, the optimal multiplier for a given value R serves as the

starting candidate for the optimal multiplier for instances with a higher value of R .

3.3.2 AN $O(n)$ TIME ALGORITHM BASED ON BINARY BREAKPOINT SEARCH

53

The second strategy is based on binary search combined with a sophisticated bookkeeping scheme. In this strategy, we maintain a valid lower bound $\lambda_\downarrow$ and a valid upper bound $\lambda_\uparrow$ on λ^* and tighten the interval $[\lambda_\downarrow, \lambda_\uparrow]$ until this interval lies in between two consecutive breakpoints. To update the bounds, a sequence of candidates for λ^* is chosen. For each candidate $\hat{\lambda}$, the value $z(\hat{\lambda})$ is evaluated and based on this either the lower or upper bound is updated accordingly. An important aspect hereby is that we guarantee that any breakpoint outside the interval $[\lambda_\downarrow, \lambda_\uparrow]$ cannot be a candidate for λ^* and thus does not have to be considered any longer.

The key to efficiency in this approach is twofold. First, similarly to Algorithm 3.1, we can compute $z(\hat{\lambda})$ efficiently by keeping track of several sets and sums that are updated throughout the algorithm. More precisely, we keep track of the following sets:

$$\mathcal{L}(\lambda_\downarrow, \lambda_\uparrow) := \{i \mid \alpha_i \geq \lambda_\uparrow\}, \quad (3.3a)$$

$$\mathcal{F}(\lambda_\downarrow, \lambda_\uparrow) := \{i \mid \alpha_i < \lambda_\downarrow \leq \lambda_\uparrow < \beta_i\}, \quad (3.3b)$$

$$\mathcal{U}(\lambda_\downarrow, \lambda_\uparrow) := \{i \mid \beta_i \leq \lambda_\downarrow\}, \quad (3.3c)$$

$$\mathcal{I}(\lambda_\downarrow, \lambda_\uparrow) := \{i \mid \lambda_\downarrow < \alpha_i < \lambda_\uparrow \text{ or } \lambda_\downarrow < \beta_i < \lambda_\uparrow\}. \quad (3.3d)$$

Note that for any pair $(\lambda_\downarrow, \lambda_\uparrow)$, these four sets form a partition of $\mathcal{N}$ (see also Figure 3.2).

Figure 3.2: Example of the variable partitioning based on their breakpoints and the interval $[\lambda_\downarrow, \lambda_\uparrow]$.

Initially, we start with lower and upper bounds $\lambda_{\downarrow} \equiv -\infty$ and $\lambda_{\uparrow} \equiv \infty$, meaning that initially we have $\mathcal{L}(\lambda_{\downarrow}, \lambda_{\uparrow}) = \mathcal{F}(\lambda_{\downarrow}, \lambda_{\uparrow}) = \mathcal{U}(\lambda_{\downarrow}, \lambda_{\uparrow}) = \emptyset$ and $\mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow}) = \mathcal{N}$. Observe that by increasing $\lambda_{\downarrow}$ and decreasing $\lambda_{\uparrow}$, an index i can move from $\mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow})$ to any of the three sets $\mathcal{L}(\lambda_{\downarrow}, \lambda_{\uparrow})$, $\mathcal{F}(\lambda_{\downarrow}, \lambda_{\uparrow})$, or $\mathcal{U}(\lambda_{\downarrow}, \lambda_{\uparrow})$. Furthermore, if an index i once moved to another set, it stays there when $\lambda_{\downarrow}$ is further increased or $\lambda_{\uparrow}$ is further decreased. Moreover, it follows from the definitions of $x(\lambda^*)$ in Equation (3.2) and the sets $\mathcal{L}(\lambda_{\downarrow}, \lambda_{\uparrow})$, $\mathcal{F}(\lambda_{\downarrow}, \lambda_{\uparrow})$, and $\mathcal{U}(\lambda_{\downarrow}, \lambda_{\uparrow})$ in Equation (3.3) that for any bounds $\lambda_{\downarrow} < \lambda_{\uparrow}$ we have

$$\begin{aligned} i \in \mathcal{L}(\lambda_{\downarrow}, \lambda_{\uparrow}) &\Rightarrow \alpha_i \geq \lambda_{\uparrow} \geq \lambda^* &\Rightarrow x_i(\lambda^*) = l_i, \\ i \in \mathcal{F}(\lambda_{\downarrow}, \lambda_{\uparrow}) &\Rightarrow \alpha_i < \lambda^* < \beta_i &\Rightarrow x_i(\lambda^*) = a_i \lambda^*, \\ i \in \mathcal{U}(\lambda_{\downarrow}, \lambda_{\uparrow}) &\Rightarrow \beta_i \leq \lambda_{\downarrow} \leq \lambda^* &\Rightarrow x_i(\lambda^*) = u_i. \end{aligned}$$

This means that by tightening the interval $[\lambda_{\downarrow}, \lambda_{\uparrow}]$, we implicitly determine for additional variables whether in the optimal solution they are equal to their lower or upper bound or are strictly in between these bounds. This process continues until this has been determined for all variables, i.e., until $[\lambda_{\downarrow}, \lambda_{\uparrow}]$ lies in between two consecutive breakpoints.

Given the interval $[\lambda_{\downarrow}, \lambda_{\uparrow}]$, we can compute $z(\lambda)$ for any $\lambda \in \mathbb{R}$ as

$$\begin{aligned} z(\lambda) &= \sum_{i \in \mathcal{L}(\lambda_{\downarrow}, \lambda_{\uparrow})} x_i(\lambda) + \sum_{i \in \mathcal{F}(\lambda_{\downarrow}, \lambda_{\uparrow})} x_i(\lambda) \\ &\quad + \sum_{i \in \mathcal{U}(\lambda_{\downarrow}, \lambda_{\uparrow})} x_i(\lambda) + \sum_{i \in \mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow})} x_i(\lambda) \\ &= \sum_{i \in \mathcal{L}(\lambda_{\downarrow}, \lambda_{\uparrow})} l_i + \sum_{i \in \mathcal{F}(\lambda_{\downarrow}, \lambda_{\uparrow})} a_i \lambda + \sum_{i \in \mathcal{U}(\lambda_{\downarrow}, \lambda_{\uparrow})} u_i \\ &\quad + \sum_{i \in \mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow}): \alpha_i \geq \lambda} l_i + \sum_{i \in \mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow}): \alpha_i < \lambda < \beta_i} a_i \lambda + \sum_{i \in \mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow}): \beta_i \leq \lambda} u_i. \end{aligned}$$

For an efficient calculation of this value throughout the breakpoint search, we keep track of the sums

$$P(\lambda_{\downarrow}, \lambda_{\uparrow}) := \sum_{i \in \mathcal{L}(\lambda_{\downarrow}, \lambda_{\uparrow})} l_i + \sum_{i \in \mathcal{U}(\lambda_{\downarrow}, \lambda_{\uparrow})} u_i, \quad Q(\lambda_{\downarrow}, \lambda_{\uparrow}) := \sum_{i \in \mathcal{F}(\lambda_{\downarrow}, \lambda_{\uparrow})} a_i$$

and update these sums whenever a variable index is moved from $\mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow})$ to any of the other sets $\mathcal{L}(\lambda_{\downarrow}, \lambda_{\uparrow})$, $\mathcal{F}(\lambda_{\downarrow}, \lambda_{\uparrow})$, or $\mathcal{U}(\lambda_{\downarrow}, \lambda_{\uparrow})$.

Second, in each iteration of the breakpoint search, we choose the candidate $\hat{\lambda}$ to be the median of the set of still to-be-considered breakpoints. This has two advantages with regard to efficiency. First, finding the median of a set of m elements can be done in $O(m)$ time [28]. Second, the set of still to-be-considered breakpoints is at least halved during each iteration, meaning that the number of iterations of the overall strategy is $O(\log n)$.

Algorithm 3.2 outlines the sketched strategy. To determine the time complexity of this algorithm, note that

- » The initialization and final computations in Lines 3-4 and Lines 25-28 take $O(n)$ time.
- » In each iteration of the breakpoint search, Lines 6, 13, and 16 each take $O(|\mathcal{D}|)$ time and Line 7 and Lines 17-23 take $O(|\mathcal{I}|)$ time.
- » We have $|\mathcal{I}| \leq |\mathcal{D}|$ by definition of $\mathcal{I}$ and because $\mathcal{D}$ consists of all breakpoints within $[\lambda_{\downarrow}, \lambda_{\uparrow}]$. As a consequence, all operations in a given iteration of the breakpoint search take $O(|\mathcal{D}|)$ operations.

Since the size of $\mathcal{D}$ is at least halved in each iteration, the total time complexity of Algorithm 3.2 is

$$O(n) + O\left(\sum_{k=0}^{\log(n)} \frac{n}{2^k}\right) \leq O(n) + O\left(n + \frac{n}{2} + \frac{n}{4} + \frac{n}{8} + \cdots\right) = O(n).$$

We note that several improvements listed in [100] can be used to improve the practical performance of Algorithm 3.2. For example, in order to speed up the computation of $z(\hat{\lambda})$ in Line 7, one could also keep track of the sums $\sum_{i \in \mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow}) : \alpha_i \geq \lambda} l_i$, $\sum_{i \in \mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow}) : \alpha_i < \lambda < \beta_i} a_i$, and $\sum_{i \in \mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow}) : \beta_i \leq \lambda} u_i$ and update these sums accordingly after the update of $\mathcal{D}$ in Lines 13 and 16.

We conclude this section with two remarks on the two breakpoint search strategies presented in this section.

1. Both strategies can be generalized to solve Problem QRA where we replace each term $\frac{1}{2} \frac{x_i^2}{a_i}$ by a convex function f_i . However, the major computational gain when applying these strategies to Problem QRA is that we can compute $z(\lambda)$ and eventually λ^* efficiently by means of book-keeping via the sums $P(\lambda)$ and $Q(\lambda)$ (in the sequential search strategy) and $P(\lambda_{\downarrow}, \lambda_{\uparrow})$ and $Q(\lambda_{\downarrow}, \lambda_{\uparrow})$ (in the binary search strategy). This is in contrast to the case of general convex functions, where computing $z(\lambda)$ for an arbitrary λ takes $O(n)$ time and a nonlinear equation has to be solved to find λ^* [138].
2. Although Algorithm 3.2 has a better worst-case time complexity than Algorithm 3.1, it is not necessarily faster in practice. This is due to the relatively large constant factor that is "hidden" in the worst-case time complexity of the median-find algorithm in [28]. As a consequence, this type of algorithm is rarely used in practice. However, recent studies have demonstrated the potential of similar median-find algorithms that have smaller constant factors and are thereby faster (see, e.g., [7]).

Algorithm 3.2: An $O(n)$ algorithm for Problem QRA.

```

1 Input: Parameters  $a \in \mathbb{R}_{>0}^n$ ,  $R \in \mathbb{R}$ , and  $l, u \in \mathbb{R}^n$ 
2 Output: Optimal solution  $\bar{x}$  to Problem QRA
3 Compute the breakpoint sets  $\mathcal{A} := \{\alpha_i \mid i \in \mathcal{N}\}$  and  $\mathcal{B} := \{\beta_i \mid i \in \mathcal{N}\}$ 
4 Initialize  $\mathcal{D} := \mathcal{A} \cup \mathcal{B}$ ;  $\mathcal{I} := \mathcal{N}$ ;  $P := 0$ ;  $Q := 0$ ;  $\lambda_{\downarrow} := -\infty$ ;  $\lambda_{\uparrow} := \infty$ 
5 repeat
6    $\hat{\lambda} := \text{median}(\mathcal{D})$ 
7   Compute  $z(\hat{\lambda}) := P + Q\hat{\lambda} + \sum_{i \in \mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow}) : \alpha_i \geq \hat{\lambda}} l_i +$ 
      $\sum_{i \in \mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow}) : \alpha_i < \hat{\lambda} < \beta_i} a_i \hat{\lambda} + \sum_{i \in \mathcal{I}(\lambda_{\downarrow}, \lambda_{\uparrow}) : \beta_i \leq \hat{\lambda}} u_i$ 
8   if  $z(\hat{\lambda}) = R$  then  $\triangleright \lambda^* = \hat{\lambda}$ 
9     | Compute  $x(\lambda^*)$  using Equation (3.2)
10    | return  $\bar{x} := x(\lambda^*)$ 
11  else if  $z(\hat{\lambda}) > R$  then  $\triangleright \lambda^* < \hat{\lambda}$ 
12    | Update upper bound  $\lambda_{\uparrow} = \hat{\lambda}$ 
13    | Update  $\mathcal{D}$ :  $\mathcal{D} = \{\lambda \in \mathcal{D} \mid \lambda < \hat{\lambda}\}$ 
14  else  $\triangleright \lambda^* > \hat{\lambda}$ 
15    | Update lower bound  $\lambda_{\downarrow} = \hat{\lambda}$ 
16    | Update  $\mathcal{D}$ :  $\mathcal{D} = \{\lambda \in \mathcal{D} \mid \lambda > \hat{\lambda}\}$ 
17  for  $i \in \mathcal{I}$  do
18    | if  $\alpha_i \geq \lambda_{\uparrow}$  then
19      |  $\mathcal{I} := \mathcal{I} \setminus \{i\}$ ;  $P := P + l_i$ 
20    | else if  $\alpha_i < \lambda_{\downarrow} \leq \lambda_{\uparrow} < \beta_i$  then
21      |  $\mathcal{I} := \mathcal{I} \setminus \{i\}$ ;  $Q := Q + a_i$ 
22    | else if  $\beta_i \leq \lambda_{\downarrow}$  then
23      |  $\mathcal{I} := \mathcal{I} \setminus \{i\}$ ;  $P := P + u_i$ 
24  until  $\mathcal{D} = \emptyset$ 
25  if  $Q = 0$  then
26    | Compute  $x(\lambda^*)$  as  $x(\lambda_{\downarrow})$  using Equation (3.2)
27  else
28    |  $\lambda^* := \frac{R-P}{Q}$ ; compute  $x(\lambda^*)$  using Equation (3.2)
29  return  $\bar{x} := x(\lambda^*)$ 

```

3.4 APPLICATION TO ELECTRIC VEHICLE CHARGING

57

In this section, we discuss how Problem QRA appears in the context of scheduling the charging of EVs. More precisely, we consider the problem of load profile flattening for the whole grid, whereby we treat EVs as a special type of storage system that can only charge energy and does not discharge energy back into the grid. Thus, we can model the problem of scheduling EV charging analogously to that of scheduling the operation of a storage system that we introduced in Section 2.5.2, i.e., as an instance of Problem BATTERY.

Recall that, in Problem BATTERY, we divide the overall time horizon into n equidistant time intervals of length Δt indexed by the set $\mathcal{N} := \{1, \dots, n\}$ and determine for each interval $i \in \mathcal{N}$ the (dis)charging power x_i during this interval. In the case of EV charging, this amount is limited by 0 and the maximum charging rate $X_{\max}$. Given the amount of energy $\bar{R}$ that must be charged over the overall time horizon, the problem of scheduling the EV charging can be formulated as follows:

$$\begin{aligned} \text{EV : } \min_{x \in \mathbb{R}^n} \quad & \sum_{i \in \mathcal{N}} \frac{1}{2} (x_i + p_i)^2 \\ \text{s.t.} \quad & \sum_{i \in \mathcal{N}} x_i = \frac{\bar{R}}{\Delta t}, \\ & 0 \leq x_i \leq X_{\max}, \quad i \in \mathcal{N}. \end{aligned}$$

Note that this problem is an instance of Problem QRA by applying the variable transform $y_i = x_i + p_i$ for all $i \in \mathcal{N}$. Moreover, we can also model an EV charging scheduling problem involving a nonzero lower charging rate bound $X_{\min}$, i.e., with the constraints $X_{\min} \leq x_i$ instead of $0 \leq x_i$ for all $i \in \mathcal{N}$, as an instance of Problem EV by applying the variable transform $y_i = x_i + X_{\min}$ for all $i \in \mathcal{N}$. Finally, note that a more general model where these rates can differ per interval i is also an instance of Problem QRA.

In Problem EV, the optimal Lagrange multiplier has a nice physical interpretation. To see this, observe that for $\lambda \in \mathbb{R}$ the optimal solution to the Lagrangian relaxation of this problem is given by

$$x_i(\lambda) = \begin{cases} 0 & \text{if } \lambda - p_i < 0, \\ \lambda - p_i & \text{if } 0 \leq \lambda - p_i < X_{\max}, \\ X_{\max} & \text{if } X_{\max} \leq \lambda - p_i. \end{cases}$$

This implies that the optimal Lagrange multiplier λ^* represents a "best" level of "flatness" of the combined EV and base load profile. This is also illustrated by Figure 3.3.

Figure 3.3: EV charging using the optimal Lagrange multiplier λ^* . The EV starts charging at 18:00h and its charging deadline is 7:00h on the next day.

3.5 OUTLOOK

In this chapter, we studied a specific resource allocation problem (RAP), namely the RAP with quadratic cost functions, additional box constraints, and continuous variables (Problem QRA). In particular, we discussed a solution approach for the problem of finding an optimal Lagrange multiplier that characterizes the unique optimal solution to the problem. Moreover, we studied two search strategies for finding this Lagrange multiplier that are based on sequential and binary search respectively.

Problem QRA and Problem EV form important building blocks for the RAPs discussed in Chapters 4, 5, and 8 and for the exposition of our online duality-driven optimization (ODDO) framework given in Chapters 6-8. We conclude this chapter by pointing out the relevance of Problem QRA for later chapters:

1. In Chapter 4, we study the RAP with quadratic cost functions, additional box and nested constraints, and continuous variables, i.e., $(a, 0)$ -Q/NC/C in terms of the classification scheme introduced in Section 2.2.3. We show that solving this problem can be reduced to solving a sequence of specific instances of Problem QRA whose box constraints are optimal solutions to instances earlier in the sequence.
2. In Chapter 5, we study the RAP with a nonseparable quadratic objective function, additional generalized bound constraints, and continuous variables. We show that each instance of this problem can be reduced to an instance without the generalized bound constraints by solving a sequence of specific instances of Problem QRA. Moreover, we show that the resulting instance can be solved using a sequential breakpoint search approach that is in spirit similar to the one discussed in Section 3.3.1.

3. In Chapter 7, we present two prediction strategies for the ODDO framework introduced in Chapter 6. Hereby, we use Problem EV as an illustrative example.
4. In Chapter 8, we study the performance of ODDO when some of the original assumptions made in the framework are relaxed. For this, we use two variants of Problem EV as illustrative problems. The first variant is obtained from Problem EV by replacing the objective function with a piece-wise linear approximation of this function. To solve this variant, we use a sequential search approach that is similar to that described in Section 3.3.1. The second variant is obtained from Problem EV by adding the restriction that charging rates between zero and a given minimum threshold are not allowed. We show that this variant can be solved as a sequence of specific instances of Problem EV.

4

QUADRATIC RESOURCE ALLOCATION WITH NESTED CONSTRAINTS

ABSTRACT – In this chapter, we study the quadratic resource allocation problem with lower and upper constraints on nested sums of variables. As discussed in Chapter 2, this problem occurs in many applications. In particular, we consider in this chapter battery scheduling within decentralized energy management (DEM) for smart grid as a leading example. We present an algorithm for this problem that matches the state-of-the-art time complexity $O(n \log n)$ but, in contrast to existing algorithms for this problem, that achieves this time complexity using relatively simple and easy-to-implement subroutines and data structures. This makes our algorithm very attractive for real-life application and implementation. Numerical comparisons of our algorithm with a subroutine for battery scheduling within an existing tool for DEM indicates that our algorithm significantly reduces the overall execution time of the DEM system, especially when the battery is expected to be completely full or empty multiple times in the optimal schedule. Moreover, computational experiments with synthetic data show that our algorithm outperforms the currently most efficient algorithm by more than one order of magnitude. In particular, our algorithm can solve the considered instances with up to one million variables in less than 17 seconds on a personal computer.

4.1 INTRODUCTION

In this chapter, we study the resource allocation problem (RAP) with a quadratic objective function, additional box and nested constraints, and continuous variables. In terms of the classification scheme introduced in Section 2.2.3, this is the problem $(a, 0)\text{-Q/NC/C}$, where $a \in \mathbb{R}_{>0}$. For convenience in this and later

This chapter is based on and contains parts of [SU:2].

chapters, we call this problem the quadratic resource allocation problem with nested constraints (Problem QRA-NC).

In Section 2.3.3, we already surveyed several approaches for solving this problem. In particular, we discussed the only two known tailored algorithms for this problem, namely the infeasibility-guided algorithm in [167], hereafter referred to as ALG_{inf} , and the decomposition algorithm in [172], hereafter referred to as ALG_{dec} . Both algorithms are based on a divide-and-conquer strategy, wherein the problem is split up into subproblems that are instances of the quadratic resource allocation problem (Problem QRA, see Chapter 3). The worst-case time complexities of these algorithms are $O(n^2)$ for ALG_{inf} and $O(n \log n)$ for ALG_{dec} , making the latter algorithm the currently known most efficient algorithm for Problem QRA-NC.

Although the decomposition-based algorithm ALG_{dec} has a good worst-case time complexity, we observe several disadvantages of this algorithm that may make it less favorable in practice than its good worst-case time complexity suggests:

1. Each level of recursion within ALG_{dec} solves a series of instances of Problem QRA whose parameters are determined by optimal solutions to multiple instances of Problem QRA on earlier levels. Since each instance is solved from scratch, much time is spent on initializing the subproblems.
2. ALG_{dec} achieves for each level of the recursion an $O(n)$ time complexity by solving the QRA subproblems using an $O(n)$ time algorithm such as the ones in [100] and Section 3.3.2. These $O(n)$ time algorithms repeatedly invoke linear-time algorithms such as [28] to find the median of a set. However, as already mentioned in Chapter 2, these median-find algorithms are relatively slow in practice due to a big constant factor in their complexity [28]. Moreover, they are significantly more difficult to implement than straight-forward sorting or sampling-based strategies [7, 99].

To alleviate these issues, we propose in this chapter a new algorithm for Problem QRA-NC, called ALG_{seq} , which has the same time complexity as ALG_{dec} , namely $O(n \log n)$, but in contrast requires only relatively simple and fast subroutines to attain this complexity. As a consequence, this algorithm is both faster in practice and easier to implement than ALG_{dec} . These are in practice generally more important criteria for the actual adaptation of a given algorithm than the polynomial worst-case time complexity [125]. Our algorithm builds upon the monotonicity results for Problem QRA-NC derived in [172] and solves a sequence of QRAP subproblems that have a sequential nested structure rather than the divide-and-conquer structure of both ALG_{dec} and ALG_{inf} . More precisely, for each $j \in \mathcal{N}$, the j^{th} subproblem involves only the first j variables $x_1, \dots, x_j$. As a consequence, our approach can solve its first j subproblems without any knowledge on the parameters involving indices higher than j , whereas both ALG_{inf} and ALG_{dec} require all problem parameters to be known a priori. This

makes our algorithm particularly useful in situations where problem parameters become known only over time. Moreover, due to the nested structure, each input and bookkeeping parameter is accessed within a relatively small time period instead of frequently throughout the entire course of the algorithm. This is beneficial in combination with caching as used in micro-processors since this increases the number of times a value can be accessed quickly from a cache instead of relatively slowly from the main memory.

We attain the $O(n \log n)$ complexity using an efficient implementation of double-ended priority queues [30, 102] for several bookkeeping parameters. This data type supports insertion of arbitrary elements and finding and deletion of minimum and maximum elements in at most $O(\log n)$ time. Our approach requires $O(n)$ of such operations, which leads to an overall time complexity of $O(n \log n)$. Double-ended priority queues can be implemented using specialized data structures such as min-max heaps [11] or by a simple coupling of a standard min-heap and max-heap (see also [30]). The latter heaps are one of the most basic data structures and many efficient implementations exist for different programming languages [31]. Thus, we can achieve the time complexity of $O(n \log n)$ using relatively simple data structures, as opposed to ALG_{dec} that requires a more involved implementation of a linear-time median algorithm.

Note that by the reduction result of Corollary 2.1, our algorithm for Problem QRA-NC also leads to efficient and fast algorithms for instances of this problem where we replace the term $\frac{1}{2} \frac{x_i^2}{a_i}$ by $a_i f(\frac{x_i}{a_i})$ for each $i \in \mathcal{N}$ with a given convex function f , i.e., for the problem $(a, 0, f)$ -S/NC/C. Such a structure is present in many applications considered in the literature, in particular in most of the applications surveyed or evaluated in [5, 172]. This leads to faster algorithms for a wide range of practical problems, including the vessel speed optimization problem [84, 131] and processor scheduling with agreeable deadlines [55, 83] (see also Sections 2.5.4 and 2.5.5).

In Section 4.4, we evaluate the performance of our algorithm ALG_{seq} and compare it to the state-of-the-art algorithms ALG_{inf} and ALG_{dec} . For this evaluation, we use both synthetic instances and realistic instances of the battery scheduling problem BATTERY introduced in Section 2.5.2 using real power consumption data as input, where in Problem BATTERY we choose load profile flattening as objective. We compare our approach to a tailored implementation of ALG_{inf} within DEMKit, an existing simulation tool for decentralized energy management (DEM) research [82]. Within DEMKit, the battery scheduling problem is used as a subroutine within a distributed optimization framework that coordinates the energy consumption of multiple devices [58]. Our results indicate that the number of tight nested constraints in an optimal solution greatly influences which algorithm is faster for a given problem instance. In particular, ALG_{seq} is on average faster than ALG_{inf} , except when the percentage of tight nested constraints is relatively low (less than 2%). Moreover, the execution time of ALG_{seq} is more stable than that of ALG_{inf} , which makes our algorithm more suitable for

use in DEM systems that employ a high level of parallelism (see, e.g., [81]). With regard to the synthetic instances, we study the scalability of ALG_{seq} , ALG_{inf} , and ALG_{dec} . Our results indicate that both our algorithm ALG_{seq} and ALG_{inf} are at least one order of magnitude faster than ALG_{dec} . Moreover, ALG_{seq} is on average almost twice as fast as ALG_{inf} . In particular, ALG_{seq} solves instances with up to one million variables in less than 17 seconds.

The outline of the remainder of this chapter is as follows. In Section 4.2, we formulate Problem QRA-NC and present an initial sequential algorithm for solving this problem with an $O(n^2)$ worst-case time complexity. Based on this algorithm, we derive in Section 4.3 our $O(n \log n)$ time algorithm ALG_{seq} . In Section 4.4, we evaluate the performance of this algorithm and compare it to the state-of-the-art. Finally, we provide our conclusions in Section 4.5.

4.2 PROBLEM FORMULATION AND AN INITIAL SEQUENTIAL ALGORITHM

Problem QRA-NC can be formulated as follows:

$$\begin{aligned} \text{QRA-NC : } \min_{x \in \mathbb{R}^n} \quad & \sum_{i \in \mathcal{N}} \frac{1}{2} \frac{x_i^2}{a_i} \\ \text{s.t. } \quad & L_j \leq \sum_{i \in \mathcal{N}_j} x_i \leq U_j, \quad j \in \mathcal{N}_{n-1}, \\ & \sum_{i \in \mathcal{N}} x_i = R, \\ & l_i \leq x_i \leq u_i, \quad i \in \mathcal{N}, \end{aligned} \tag{4.1}$$

where $\mathcal{N}_j := \{1, \dots, j\}$ for $j \in \mathcal{N} \setminus \{n\}$, $L, U \in \mathbb{R}^{n-1}$, and where we define $L_n = U_n = R$ for convenience. Note that if $L_j = U_j$ for some $j \in \mathcal{N}_{n-1}$, we may split the problem into two smaller instances of Problem QRA-NC that involve the variables $x_1, \dots, x_j$ and $x_{j+1}, \dots, x_n$ respectively. Thus, we assume without loss of generality that $L_j < U_j$ for all $j \in \mathcal{N}_{n-1}$. Moreover, we may assume that $L_1 = l_1$, $U_1 = u_1$, and $L_j \geq L_{j-1} + l_j$ and $U_j \leq U_{j-1} + u_j$ for $j \in \mathcal{N}_{n-1} \setminus \{1\}$.

In this section, we present an initial sequential algorithm for Problem QRA-NC. This algorithm solves the problem as a sequence of $2n - 1$ instances of Problem QRA whose single-variable bounds (4.1) are given by optimal solutions to previous QRAP subproblems. For this, we consider a sequence of restricted subproblems where we take into account only a subset of the variables. More precisely, we define for each $j \in \mathcal{N}$ and $C \in \mathbb{R}$ the following subproblem:

$$\begin{aligned} \text{QRA-NC}^j(C) : \min_{x \in \mathbb{R}^j} \sum_{i \in \mathcal{N}_j} \frac{1}{2} \frac{x_i^2}{a_i} \\ \text{s.t.} \quad \sum_{i \in \mathcal{N}_j} x_i = C, \end{aligned} \quad (4.2)$$

$$L_k \leq \sum_{i \in \mathcal{N}_k} x_i \leq U_k, \quad k \in \mathcal{N}_{j-1}, \quad (4.3)$$

$$l_i \leq x_i \leq u_i, \quad i \in \mathcal{N}_j.$$

Throughout this chapter, we denote the optimal solution to this subproblem by $x^j(C) := (x_i^j(C))_{i \in \mathcal{N}_j}$. Note that this optimal solution is unique since the objective function of the corresponding problem is *strictly* convex and all constraints are linear. Moreover, observe that the n^{th} subproblem $\text{QRA-NC}^n(R)$ is equal to the original Problem QRA-NC.

The key ingredient to our algorithm is that we can replace the nested constraints (4.3) by specific single-variable constraints without changing the optimal solution. By doing this, we transform an instance of Problem QRA-NC into an equivalent instance of Problem QRA. More precisely, we show that each subproblem $\text{QRA-NC}^j(C)$ yields the same optimal solution as the following instance of Problem QRA:

$$\begin{aligned} \text{QRA}^j(C) : \min_{x \in \mathbb{R}^j} \sum_{i \in \mathcal{N}_j} \frac{1}{2} \frac{x_i^2}{a_i} \\ \text{s.t.} \quad \sum_{i \in \mathcal{N}_j} x_i = C, \end{aligned} \quad (4.4)$$

$$x_i^{j-1}(L_{j-1}) \leq x_i \leq x_i^{j-1}(U_{j-1}), \quad i \in \mathcal{N}_{j-1}, \quad (4.5)$$

$$l_j \leq x_j \leq u_j, \quad (4.6)$$

where the bounds $x_i^{j-1}(L_{j-1})$ and $x_i^{j-1}(U_{j-1})$ in (4.5) are the optimal solutions of the problems $\text{QRA-NC}^{j-1}(L_{j-1})$ and $\text{QRA-NC}^{j-1}(U_{j-1})$ respectively. Note that the single-variable bounds for x_j in (4.6) are the same as those of the original subproblem $\text{QRA-NC}^j(C)$.

The validity of this transformation is proven by Lemmas 4.1-4.3. First, Lemma 4.1 shows that the optimal solution $x^j(C)$ to the subproblem $\text{QRA-NC}^j(C)$ is non-decreasing in C . Subsequently, Lemma 4.2 uses this property to show that when adding the alternative single-variable bounds (4.5) to the problem formulation of $\text{QRA-NC}^j(C)$, the optimal solution $x^j(C)$ to $\text{QRA-NC}^j(C)$ is not cut off. Finally, Lemma 4.3 shows that the alternative single-variable bounds (4.5) are stronger than the nested constraints (4.3).

Lemma 4.1. *If $L_j \leq A \leq B \leq U_j$, we have $x^j(A) \leq x^j(B)$ for a given $j \in \mathcal{N}$.*

Proof. This proof is based on the proof of Theorem 2 in [172] and given in Appendix A.1.1. $\square$

Lemma 4.2. *For a given $j \in \mathcal{N}_{n-1}$ and $C \in [L_j, U_j]$, we have that $x_i^j(L_j) \leq x_i^{j+1}(C) \leq x_i^j(U_j)$.*

Proof. Let $x' := (x_1^{j+1}(C), \dots, x_j^{j+1}(C))$ be the vector of the first j components of the optimal solution to the problem $\text{QRA-NC}^{j+1}(C)$. Since x' is feasible for all nested constraints (4.3) for $k \in \mathcal{N}_j$, this vector is also the optimal solution to $\text{QRA-NC}^j(A)$ where $A := \sum_{i \in \mathcal{N}_j} x'_i$, i.e., we have $x' = x^j(A)$. Since $A \in [L_j, U_j]$, Lemma 4.1 implies that $x_i^j(L_j) \leq x_i^j(A) \leq x_i^j(U_j)$ for all $i \in \mathcal{N}_j$. It follows that $x_i^j(L_j) \leq x_i^{j+1}(C) \leq x_i^j(U_j)$ for all $i \in \mathcal{N}_j$. $\square$

Lemma 4.3. *If for a given $j \in \mathcal{N}_{n-1}$ and vector $y \in \mathbb{R}^j$ we have $x^j(L_j) \leq y \leq x^j(U_j)$, then $L_k \leq \sum_{i \in \mathcal{N}_k} y_i \leq U_k$ for all $k \in \mathcal{N}_j$.*

Proof. The sum of the inequalities $x_i^j(L_j) \leq y_i \leq x_i^j(U_j)$ over all $i \in \mathcal{N}_k$ yields

$$\sum_{i \in \mathcal{N}_k} x_i^j(L_j) \leq \sum_{i \in \mathcal{N}_k} y_i \leq \sum_{i \in \mathcal{N}_k} x_i^j(U_j).$$

Since $x^j(L_j)$ and $x^j(U_j)$ are feasible for $\text{QRA-NC}^j(L_j)$ and $\text{QRA-NC}^j(U_j)$ respectively and $k \leq j$, we have $L_k \leq \sum_{i \in \mathcal{N}_k} x_i^j(L_j)$ and $\sum_{i \in \mathcal{N}_k} x_i^j(U_j) \leq U_k$ and the result of the lemma follows. $\square$

Lemma 4.2 implies that, given optimal solutions $x^{j-1}(L_{j-1})$ and $x^{j-1}(U_{j-1})$ to $\text{QRA-NC}^{j-1}(L_{j-1})$ and $\text{QRA-NC}^{j-1}(U_{j-1})$ respectively, we can replace the nested constraints (4.3) in $\text{QRA-NC}^j(C)$ by the single-variable bounds (4.5) without cutting off the optimal solution to $\text{QRA-NC}^j(C)$. Moreover, since these single-variable bounds are stronger than the nested constraints by Lemma 4.3, adding these constraints does not change the optimal objective value. It follows directly that any optimal solution to $\text{QRA}^j(C)$ is also optimal for $\text{QRA-NC}^j(C)$.

Based on Lemmas 4.1-4.3, we use the following approach to solve Problem QRA-NC . We successively solve the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ from $j = 1$ to $n - 1$ and finally the subproblem $\text{QRA}^n(R)$, whereby in each step we use the optimal solutions to the preceding subproblems $\text{QRA}^{j-1}(L_{j-1})$ and $\text{QRA}^{j-1}(U_{j-1})$ as input. Note that each of the subproblems is an instance of Problem QRA . This approach is summarized in Algorithm 4.1.

Since each subproblem $\text{QRA}^j(\cdot)$ can be solved in $O(n)$ time, the worst-case time complexity of Algorithm 4.1 is $O(n^2)$. However, recall that linear-time algorithms for Problem QRA attain their linear time complexity by employing linear-time algorithms for median-finding, which are, as already mentioned, in general slower than simple sorting- or sampling-based approaches [7, 99]. Note,

Algorithm 4.1: An initial sequential algorithm for Problem QRA-NC.

```

1 Input: Parameters  $a \in \mathbb{R}_{\geq 0}^n$ ,  $L, U \in \mathbb{R}^{n-1}$ ,  $R \in \mathbb{R}$ , and  $l, u \in \mathbb{R}^n$ 
2 Output: Optimal solution  $x$  to Problem QRA-NC
3 Initialize  $x_1^1(L_1) = L_1$ ;  $x_1^1(U_1) = U_1$ 
4 for  $j = 2, \dots, n-1$  do
5   | Compute optimal solutions  $x^j(L_j)$  and  $x^j(U_j)$  to  $\text{QRA}^j(L_j)$  and
   |    $\text{QRA}^j(U_j)$  respectively
6 Compute optimal solution  $x^n(R)$  to  $\text{QRA}^n(R)$ 
7 return  $\bar{x} := x^n(R)$ 

```

moreover, that also the $O(n \log n)$ time algorithm ALG_{dec} attains its worst-case time complexity by using such slow linear-time algorithms as a subroutine.

In the next section, we propose an algorithm to solve Problem QRA-NC in $O(n \log n)$ time that, as opposed to ALG_{dec} , does not require linear-time median-finding subroutines. Instead, it only requires a simple data structure for double-ended priority queues to store several bookkeeping parameters.

We conclude this section with two remarks that may be of independent interest:

1. It can be shown that Lemmas 4.1-4.3 also hold for the case where the variables are integer-valued, i.e., $x \in \mathbb{Z}^n$ (see also Theorem 5 in [172]), given that all parameters a , L , U , l , and u are also integer-valued and nonnegative. As a consequence, when solving each subproblem $\text{QRA}^j(\cdot)$ with integer variables, Algorithm 4.1 computes an optimal solution to Problem QRA-NC with integer variables. The worst-case time complexity of this algorithm is $O(n^2)$ since each $\text{QRA}^j(\cdot)$ subproblem with integer variables can be solved in $O(j)$ time ([85], see also Section 2.3.1).
2. Lemmas 4.1-4.3 can be generalized to the case where the objective function is the sum of separable convex cost functions f_i , i.e., where we replace each term $\frac{1}{2} \frac{x_i^2}{a_i}$ by a convex function $f_i(x_i)$. For this more general problem, this leads to a sequential algorithm that is very similar to Algorithm 4.1. However, initial computational tests indicated that both this algorithm and Algorithm 4.1 are in practice much slower than both ALG_{inf} and ALG_{dec} .

4.3 A FAST $O(n \log n)$ TIME ALGORITHM FOR PROBLEM QRAP-NC

The sequential algorithm derived in the previous section does not match the best-known time complexity of ALG_{dec} . However, we show in this section that we can implement Algorithm 4.1 such that its time complexity reduces to $O(n \log n)$ without requiring a linear-time median finding algorithm. Instead,

we only require a data type that supports insertion of elements and the finding and removing of minimum and maximum elements in $O(\log n)$ time. A double-ended priority queue is an example of such a data type.

The key to efficiency in our approach is that we do not explicitly compute the solution to each QRA subproblem. Instead, we only compute an optimal Lagrange multiplier corresponding to the resource constraint (4.4) that characterizes the entire optimal solution to this subproblem. Subsequently, we use these multipliers to reconstruct the optimal solution to the original Problem QRA-NC using two sets of simple recursive relations that can be executed in $O(n)$ time. In order to compute the Lagrange multipliers without explicitly storing intermediate solutions, we exploit the special structure of these multipliers and of a specific algorithm for solving Problem QRA.

First, in Section 4.3.1, we introduce some of the used notation. Second, in Section 4.3.2, we derive an efficient approach for computing the optimal Lagrange multipliers of the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$. Based on these optimal Lagrange multipliers, we derive in Section 4.3.3, two simple recursions to compute the optimal solution x to Problem QRA-NC. Finally, in Section 4.3.4, we present an $O(n \log n)$ algorithm for Problem QRA-NC and discuss an implementation that attains this worst-case time complexity.

4.3.1 NOTATION

We introduce the following notation concerning the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ that we use throughout the remainder of this chapter. We denote for $j \in \mathcal{N}$ and $C \in [L_j, U_j]$ the lower and upper single variable bounds (4.5) and (4.6) of $\text{QRA}^j(C)$ by $\bar{l}^j := (\bar{l}_i^j)_{i \in \mathcal{N}_j}$ and $\bar{u}^j := (\bar{u}_i^j)_{i \in \mathcal{N}_j}$, where $\bar{l}_i^j := x_i^{j-1}(L_{j-1})$ and $\bar{u}_i^j := x_i^{j-1}(U_{j-1})$ for $i < j$, and $\bar{l}_j^j := l_j$ and $\bar{u}_j^j := u_j$. Furthermore, we denote by $\alpha^j := (\alpha_i^j)_{i \in \mathcal{N}_j}$ and $\beta^j := (\beta_i^j)_{i \in \mathcal{N}_j}$ the lower and upper breakpoints for the $\text{QRA}^j(C)$ subproblem (see Section 3.2 for more background on breakpoints). We call the breakpoints corresponding to $i = j$, i.e., α_j^j and β_j^j , *initial* breakpoints since the problem $\text{QRA}^j(C)$ is the first subproblem, i.e., the problem with lowest index j , in which we have to compute breakpoint values for the variable x_j . Note that we can compute these breakpoints directly as $\alpha_j^j := \frac{l_j}{a_j}$ and $\beta_j^j := \frac{u_j}{a_j}$ by definition of the subproblem $\text{QRA}^j(C)$.

Furthermore, let κ^j and λ^j denote the optimal Lagrange multipliers for the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ respectively and define $\kappa := (\kappa^j)_{j \in \mathcal{N}}$ and $\lambda := (\lambda^j)_{j \in \mathcal{N}}$, where we set $\kappa^1 := \alpha_1^1$ and $\lambda^1 := \beta_1^1$. If the optimal Lagrange multiplier for a given subproblem $\text{QRA}^j(L_j)$ is not unique, we define without loss of generality κ^j as the *largest* optimal Lagrange multiplier. Analogously, we define λ^j as the *smallest* optimal Lagrange multiplier of subproblem $\text{QRA}^j(U_j)$. In this chapter, we write the optimal solution to the Lagrangian relaxation of a QRA subproblem as $x^j[\cdot]$, i.e., using the brackets $[\cdot]$ instead of $(\cdot)$, to emphasize

the distinction of this solution from the optimal solution to a subproblem of the form $\text{QRA-NC}^j(\cdot)$. Note that $x^j(L_j) = x^j[\kappa^j]$ and $x^j(U_j) = x^j[\lambda^j]$ by definition of the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ and of κ^j and λ^j .

Finally, for a given subproblem $\text{QRA}^j(C)$, we define the set of its lower breakpoints as $\mathcal{A}^j := \{\alpha_i^j \mid i \in \mathcal{N}_j\}$ and the set of its upper breakpoints as $\mathcal{B}^j := \{\beta_i^j \mid i \in \mathcal{N}_j\}$. Recall that in Section 3.2 breakpoint sets are defined as *multisets* for convenience when solving Problem QRA. However, for our approach for a fast algorithm for Problem QRA-NC, it is crucial that the breakpoint sets do not contain duplicate elements. Therefore, in this section and the remainder of this chapter, we regard $\mathcal{A}^j$ and $\mathcal{B}^j$ as ordinary sets.

4.3.2 COMPUTING THE OPTIMAL LAGRANGE MULTIPLIERS OF THE SUBPROBLEMS

The goal of this subsection is to derive an efficient approach for computing the optimal Lagrange multiplier of each QRA subproblem in Algorithm 4.1 without explicitly calculating any of the intermediate optimal solutions $x^j(L_j)$ and $x^j(U_j)$ for $j \in \mathcal{N}$. If we would follow the latter strategy, i.e., if we solve each pair of subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ from scratch, e.g., using Algorithm 3.1, we would have to explicitly compute the breakpoint sets for each pair of subproblems. This leads to $O(n^2)$ computations and thus forms an efficiency bottleneck within this algorithm.

In the following, we show that we can apply the sequential breakpoint search procedure for Problem QRA in Algorithm 3.1 of Section 3.3.1 for solving the subproblems such that each breakpoint set $\mathcal{A}^{j+1}$ can be obtained from the previous set $\mathcal{A}^j$ in $O(1)$ amortized steps, i.e., the total number of steps required to carry out this construction for all $j \in \mathcal{N}_{n-1}$ is $O(n)$. This is based on two intermediate results that we show in this subsection:

1. First, the number of distinct values that the breakpoints can take is not $O(n^2)$ but $O(n)$. We obtain this result by unveiling a useful relation between breakpoints of consecutive subproblems, i.e., between α^j , β^j and α^{j+1} , β^{j+1} .
2. Second, when constructing the breakpoint sets, each distinct breakpoint value is included in or removed from a breakpoint set at most twice during the entire procedure. For this, it is important that we solve each lower subproblem $\text{QRA}^j(L_j)$ by considering the breakpoints in non-decreasing order and each upper subproblem $\text{QRA}^j(U_j)$ by considering the breakpoints in non-increasing order.

Together, these two results imply that the construction of the breakpoint sets requires in total $O(n)$ additions and removals of breakpoint values. By using an appropriate data structure such as double-ended priority queues for maintaining the breakpoint sets, each of these steps can be executed in $O(\log n)$ time, which leads to an overall $O(n \log n)$ complexity for computing the optimal Lagrange multipliers κ and λ .

The outline of the remainder of this subsection is as follows. First, we analyze the relation between breakpoints of consecutive subproblems and show that the number of distinct breakpoint values is $O(n)$. Subsequently, we use this information and the structure of Algorithm 3.1 to construct the breakpoint sets for each subproblem from those of the preceding subproblems. Finally, we discuss how the updating of the bookkeeping parameters within the breakpoint search procedure must be adjusted when applying this procedure to the subproblems $\text{QRA}^j(L_j)$ and $\text{QRAP}^j(U_j)$.

Relation between consecutive breakpoints

We first show how we can efficiently obtain the breakpoint set of a given subproblem $\text{QRA}^{j+1}(C)$ based on the breakpoint set and optimal Lagrange multipliers of the preceding subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$. We establish for a given $j \in \mathcal{N}_{n-1}$ and $i < j$ the following relation between the subsequent lower breakpoints α_i^j and α_i^{j+1} :

- » If $\kappa^j < \alpha_i^j$, it follows from Equation (3.2) that $x_i^j[\kappa^j] = \bar{l}_i^j$ since $\alpha_i^j = \frac{\bar{l}_i^j}{a_i}$. This implies that $\bar{l}_i^{j+1} = x_i^j(L_j) = x_i^j[\kappa^j] = \bar{l}_i^j$ and thus $\alpha_i^{j+1} = \alpha_i^j$.
- » If $\alpha_i^j \leq \kappa^j < \beta_i^j$, it follows from Equation (3.2) that $x_i^j[\kappa^j] = a_i \kappa^j$. Thus, $\alpha_i^{j+1} = \frac{\bar{l}_i^{j+1}}{a_i} = \frac{x_i^j(L_j)}{a_i} = \frac{x_i^j[\kappa^j]}{a_i} = \kappa^j$.
- » If $\beta_i^j \leq \kappa^j$, then it follows from Equation (3.2) that $x_i^j[\kappa^j] = \bar{u}_i^j$. This implies that $\bar{l}_i^{j+1} = x_i^j(L_j) = x_i^j[\kappa^j] = \bar{u}_i^j$ and thus $\alpha_i^{j+1} = \beta_i^j$.

Summarizing, we can determine α_i^{j+1} from the previous breakpoints α_i^j and β_i^j and the optimal Lagrange multiplier κ^j as follows:

$$\alpha_i^{j+1} = \begin{cases} \alpha_i^j & \text{if } \kappa^j < \alpha_i^j, \\ \kappa^j & \text{if } \alpha_i^j \leq \kappa^j < \beta_i^j, \\ \beta_i^j & \text{if } \beta_i^j \leq \kappa^j. \end{cases} \quad (4.7)$$

Analogously, we obtain the following expression for the upper breakpoint β_i^{j+1} in terms of the previous breakpoints α_i^j and β_i^j and the optimal Lagrange multiplier λ^j :

$$\beta_i^{j+1} = \begin{cases} \beta_i^j & \text{if } \lambda^j > \beta_i^j, \\ \lambda^j & \text{if } \beta_i^j \geq \lambda^j > \alpha_i^j, \\ \alpha_i^j & \text{if } \alpha_i^j \geq \lambda^j. \end{cases} \quad (4.8)$$

Note that it follows from these relations that $\alpha_i^j \leq \alpha_i^{j+1}$ and $\beta_i^j \geq \beta_i^{j+1}$ for each $j \in \mathcal{N}_{n-1}$. Moreover, note that the only values that the breakpoints can take are those of the initial breakpoints α_j^j and β_j^j or of the optimal Lagrange multipliers κ and λ . Thus, the number of distinct values among all breakpoints is limited by $4n$.

Constructing consecutive breakpoint sets

As observed at the end of Section 3.3.1, we can solve a given QRA subproblem by searching its breakpoints either in non-decreasing or non-increasing order. In particular, we can solve all lower subproblems $\text{QRA}^j(L_j)$ by searching the breakpoints in non-decreasing order and all upper subproblems $\text{QRA}^j(U_j)$ by searching the breakpoints in non-increasing order. When doing this, note that for solving the upper subproblem $\text{QRA}^j(U_j)$ we can use as breakpoint sets the sets that “remain” from the breakpoint search for the lower subproblem. More precisely, instead of using the sets $\mathcal{A}^j$ and $\mathcal{B}^j$ as breakpoint sets for the lower subproblem $\text{QRA}^j(L_j)$, we can use the sets $\{\alpha_i^j \in \mathcal{A}^j \mid \alpha_i^j \geq \kappa^j\}$ and $\{\beta_i^j \in \mathcal{B}^j \mid \beta_i^j \geq \kappa^j\}$ respectively. This is because $\kappa^j \leq \lambda^j$ and thus in the breakpoint search for the upper problem $\text{QRA}^j(U_j)$ no breakpoints smaller than κ^j need to be considered.

Based on this, we define the sets $\tilde{\mathcal{A}}^j$ and $\tilde{\mathcal{B}}^j$ as the sets of lower and upper breakpoints that remain to be considered after solving the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$, i.e., we have

$$\begin{aligned}\tilde{\mathcal{A}}^j &:= \{\alpha_i^j \in \mathcal{A}^j \mid \kappa^j \leq \alpha_i^j \leq \lambda^j\}, \\ \tilde{\mathcal{B}}^j &:= \{\beta_i^j \in \mathcal{B}^j \mid \kappa^j \leq \beta_i^j \leq \lambda^j\}.\end{aligned}$$

We call these sets the *remaining breakpoint sets* of the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$. In the following, we relate these two remaining breakpoint sets to the breakpoint sets of the next two subproblems, i.e., to the sets $\mathcal{A}^{j+1}$ and $\mathcal{B}^{j+1}$. For this, we focus on the relation between the lower remaining breakpoint sets $\tilde{\mathcal{A}}^j$ and the lower breakpoint set $\mathcal{A}^{j+1}$; the relation between the upper remaining breakpoint set $\tilde{\mathcal{B}}^j$ and the upper breakpoint set $\mathcal{B}^{j+1}$ is analogous.

To calculate α_i^{j+1} for each $i \in \mathcal{N}_j$, we consider four cases:

1. If $\kappa^j \leq \alpha_i^j \leq \lambda^j$, it follows from Equation (4.7) that $\alpha_i^{j+1} = \alpha_i^j$. Thus, all values in $\tilde{\mathcal{A}}^j$ act as remaining breakpoint values for the next subproblems, i.e., $\tilde{\mathcal{A}}^j \subseteq \mathcal{A}^{j+1}$.
2. If $\alpha_i^j < \kappa^j$ and $\kappa^j < \beta_i^j$, it follows from Equation (4.7) that $\alpha_i^{j+1} = \kappa^j$.
3. If $\alpha_i^j < \kappa^j$ and $\beta_i^j \leq \kappa^j$, it follows from Equations (4.7) and (4.8) that $x_i^j(L_j) = \bar{u}_i^j$ and $\beta_i^{j+1} = \beta_i^j = \alpha_i^{j+1}$ respectively. Thus,

$$\beta_i^{j+1} = \alpha_i^{j+1} \leq \dots \leq \alpha_i^n \leq \beta_i^n \leq \dots \leq \beta_i^{j+1}.$$

This means that $\alpha_i^{j'} = \beta_i^{j'} = \beta_i^j$ and $\bar{l}_i^{j'} = \bar{u}_i^{j'} = \bar{u}_i^j$ for all $j' > j$. In particular, we have $\alpha_i^{j+1} = \beta_i^j$. Note, moreover, that in all remaining subproblems, the lower and upper breakpoints of i coincide and $x_i^{j'}(C) = \bar{u}_i^j$ for any $j' > j$ and $L_{j'} \leq C \leq U_{j'}$, regardless of the values of the

future optimal Lagrange multipliers $\kappa^{j'}$ and $\lambda^{j'}$. This means that this index (variable) does not play an active role anymore in the breakpoint search and can thus be removed from the search.

4. Finally, if $\alpha_i^j > \lambda^j$, it follows from Equations (4.7) and (4.8) that $\alpha_i^{j+1} = \alpha_i^j$ and $\beta_i^{j+1} = \alpha_i^j$ respectively. Thus, $\alpha_i^{j+1} = \beta_i^{j+1} = \alpha_i^j$. Analogously to the case $\alpha_i^j \leq \beta_i^j < \kappa^j$, it follows that $\bar{l}_i^{j'} = \bar{u}_i^{j'} = \bar{l}_i^j$ and $x_i^{j'}(C) = \bar{l}_i^j$ for all $j' > j$ and $L_{j'} \leq C \leq U_{j'}$. Thus, also in this case the index i does not play an active role anymore in the breakpoint search. This means that this index (variable) does not play an active role anymore in the breakpoint search and can thus be removed from the search.

These four cases imply that we can construct $\mathcal{A}^{j+1}$ from $\tilde{\mathcal{A}}^j$ as follows:

$$\mathcal{A}^{j+1} = \tilde{\mathcal{A}}^j \cup \{\alpha_{j+1}^{j+1}\} \cup \begin{cases} \{\kappa^j\} & \text{if there exists } i \text{ such that } \alpha_i^j < \kappa^j < \beta_i^j, \\ \emptyset & \text{otherwise.} \end{cases}$$

Analogously, we can construct $\mathcal{B}^{j+1}$ from $\tilde{\mathcal{B}}^j$ as follows:

$$\mathcal{B}^{j+1} = \tilde{\mathcal{B}}^j \cup \{\beta_{j+1}^{j+1}\} \cup \begin{cases} \{\lambda^j\} & \text{if there exists } i \text{ such that } \alpha_i^j < \lambda^j < \beta_i^j, \\ \emptyset & \text{otherwise.} \end{cases}$$

The above constructions show how the breakpoint sets evolve over the course of the algorithm. In these constructions, at most $4n$ additions of breakpoint values to a breakpoint set occur. Moreover, during the breakpoint search procedure of Algorithm 3.1, breakpoints are only removed and not added. This means that updating the breakpoint sets can be done in $O(n)$ steps, i.e., by $O(n)$ additions and removals of breakpoint values.

Updating bookkeeping parameters

In order to efficiently compute the sums $z^j[\delta] := \sum_{i \leq j} x_i^j[\delta]$ for a given breakpoint δ , we define the following bookkeeping parameters analogously to those in the breakpoint search procedure for Problem QRA in Algorithm 3.1:

$$P^j(\delta) := \sum_{i \leq j: \delta < \frac{\bar{l}_i^j}{a_i}} \bar{l}_i^j + \sum_{i \leq j: \delta \geq \frac{\bar{u}_i^j}{a_i}} \bar{u}_i^j, \quad Q^j(\delta) := \sum_{i \leq j: \frac{\bar{l}_i^j}{a_i} \leq \delta < \frac{\bar{u}_i^j}{a_i}} a_i;$$

Each breakpoint value κ^ℓ and λ^ℓ in a given breakpoint set acts as a collective breakpoint for one or multiple activities. As a consequence, within the breakpoint search procedure, they have the same function as the “regular” initial lower and upper breakpoint values α_i^i and β_i^i . Thus, when a breakpoint value of the form κ^ℓ or λ^ℓ has been considered, we require an efficient update of the bookkeeping sums $P^j(\kappa^\ell)$, $Q^j(\kappa^\ell)$ or $P^j(\lambda^\ell)$, $Q^j(\lambda^\ell)$ respectively. In the case of κ^ℓ , we update $P^j(\kappa^\ell)$ by subtracting from this value the sum of the lower bounds $\bar{l}_i^j$

of those activities i whose lower breakpoint equals κ^ℓ , i.e., for which $\alpha_i^j = \kappa^\ell$. The sum of these values is

$$\begin{aligned} \sum_{i < j: \alpha_i^j = \kappa^\ell} \bar{l}_i^j &= \sum_{i < j: \alpha_i^j = \kappa^\ell} a_i \alpha_i^j = \sum_{i < j: \alpha_i^j = \kappa^\ell} a_i \kappa^\ell \\ &= \kappa^\ell \sum_{i \leq \ell: \alpha_i^\ell \leq \kappa^\ell < \beta_i^\ell} a_i = Q^\ell(\kappa^\ell) \kappa^\ell, \end{aligned}$$

since $\alpha_i^j = \frac{\bar{l}_i^j}{a_i}$ for each $i \in \mathcal{N}_j$ and we have that $\alpha_i^j = \kappa^\ell$ if and only if $\alpha_i^{\ell'} = \kappa^\ell$ for all $\ell' \in \{\ell, \dots, j\}$. Thus, the update of $P^j(\kappa^\ell)$ takes the form $P^j(\kappa^\ell) := P^j(\kappa^\ell) - Q^\ell(\kappa^\ell) \kappa^\ell$. Analogously, we update the bookkeeping sum $Q^j(\kappa^\ell)$ by adding to this value the sum of the parameters a_i for which $\alpha_i^j = \kappa^\ell$. This sum is

$$\sum_{i < j: \alpha_i^j = \kappa^\ell} a_i = \sum_{i \leq \ell: \alpha_i^\ell \leq \kappa^\ell < \beta_i^\ell} a_i = Q^\ell(\kappa^\ell).$$

Thus, the update takes the form $Q^j(\kappa^\ell) := Q^j(\kappa^\ell) + Q^\ell(\kappa^\ell)$.

The updates for the case of λ^ℓ , i.e., for $P^j(\lambda^\ell)$ and $Q^j(\lambda^\ell)$, are analogous to those for the case of κ^ℓ . Table 4.1 provides an overview of the updates of the bookkeeping sums for both these cases for each of the four types of breakpoint values α_i^j , β_i^j , κ^ℓ , and λ^ℓ .

Type of δ	In $\text{QRA}^j(L_j)$ (non-decreasing search)	
	$P^j(\delta)$	$Q^j(\delta)$
$\delta \equiv \alpha_i^j$	$P^j(\delta) - l_i$	$Q^j(\delta)_i + a_i$
$\delta \equiv \beta_i^j$	$P^j(\delta) + u_i$	$Q^j(\delta) - a_i$
$\delta \equiv \kappa^\ell, \ell < j$	$P^j(\delta) - Q^\ell(\kappa^\ell) \kappa^\ell$	$Q^j(\delta) + Q^\ell(\kappa^\ell)$
$\delta \equiv \lambda^\ell, \ell < j$	$P^j(\delta) + Q^\ell(\lambda^\ell) \lambda^\ell$	$Q^j(\delta) - Q^\ell(\lambda^\ell)$
Type of δ	In $\text{QRA}^j(U_j)$ (non-increasing search)	
	$P^j(\delta)$	$Q^j(\delta)$
$Q^j(\delta)_i + a_i$	$P^j(\delta) + l_i$	$Q^j(\delta) - a_i$
$Q^j(\delta) - a_i$	$P^j(\delta) - u_i$	$Q^j(\delta) + a_i$
$Q^j(\delta) + Q^\ell(\kappa^\ell)$	$P^j(\delta) + Q^\ell(\kappa^\ell) \kappa^\ell$	$Q^j(\delta) - Q^\ell(\kappa^\ell)$
$Q^j(\delta) - Q^\ell(\lambda^\ell)$	$P^j(\delta) - Q^\ell(\lambda^\ell) \lambda^\ell$	$Q^j(\delta) + Q^\ell(\lambda^\ell)$

Table 4.1: Updating the bookkeeping sums $P^j(\delta)$ and $Q^j(\delta)$ when searching the breakpoints in non-decreasing order ($\text{QRA}^j(L_j)$) and non-increasing order ($\text{QRA}^j(U_j)$).

4.3.3 RECOVERING THE OPTIMAL SOLUTION TO PROBLEM QRAP-NC

In the previous section, we discussed an efficient way to compute the optimal Lagrange multipliers κ^j and λ^j for the QRA subproblems $\text{QRA}^j(L_j)$ and

$\text{QRA}^j(U_j)$. In this section, we show how we can use these values to compute the optimal solution $x^n(R)$. For this, we first determine which nested constraints are tight in $x^n(R)$ and use this information to reconstruct the individual terms $x_i^n(R)$ for all $i \in \mathcal{N}$. To this end, for each $j \in \mathcal{N}_{n-1}$, let ℓ_j denote the smallest index larger than or equal to j such that one of its corresponding nested constraints is tight in $x^n(R)$. More precisely,

$$\ell_j := \min \left(k \geq j \mid \sum_{i \in \mathcal{N}_k} x_i^n(R) = L_k \text{ or } \sum_{i \in \mathcal{N}_k} x_i^n(R) = U_k \right).$$

Furthermore, let V_j denote the value of the tight nested constraint corresponding to the index ℓ_j , i.e., $V_j \in \{L_{\ell_j}, U_{\ell_j}\}$, and χ^j the corresponding multiplier, i.e., $\chi^j \in \{\kappa^{\ell_j}, \lambda^{\ell_j}\}$. More precisely,

- » $\sum_{i \in \mathcal{N}_{\ell_j}} x_i^n(R) = L_{\ell_j}$ implies $V_j = L_{\ell_j}$ and $\chi^j = \kappa^{\ell_j}$;
- » $\sum_{i \in \mathcal{N}_{\ell_j}} x_i^n(R) = U_{\ell_j}$ implies $V_j = U_{\ell_j}$ and $\chi^j = \lambda^{\ell_j}$.

The main result in this subsection is that the values χ^j act as optimal Lagrange multipliers for the resource constraint (4.4) in the subproblem $\text{QRA}^n(R)$. As a consequence, given these values, we can calculate $x^n(R)$ directly using a relation similar to the Lagrangian relaxation solution in Equation (3.2). To show this result, we prove Lemmas 4.4 and 4.5. First, Lemma 4.4 shows how we can iteratively compute χ from the optimal multipliers κ and λ using a simple recursive relation. Second, Lemma 4.5 shows how we can calculate $x^n(R)$ from χ using a relation similar to that in Equation (3.2) for Problem QRA.

Lemma 4.4. *It holds that $\chi^n = \kappa^n = \lambda^n$. Moreover, for each $j \in \mathcal{N}_{n-1}$, the following implications are valid:*

1. $\chi^{j+1} \leq \kappa^j$ implies $\sum_{i \in \mathcal{N}_j} x_i^n(R) = L_j$ and $\chi^j = \kappa^j$;
2. $\lambda^j \leq \chi^{j+1}$ implies $\sum_{i \in \mathcal{N}_j} x_i^n(R) = U_j$ and $\chi^j = \lambda^j$;
3. $\kappa^j < \chi^{j+1} < \lambda^j$ implies $L_j < \sum_{i \in \mathcal{N}_j} x_i^n(R) < U_j$ and $\chi^j = \chi^{j+1}$.

Proof. See Appendix A.1.2. □

Lemma 4.5. *For each $i \in \mathcal{N}$, we can calculate $x_i^n(R)$ by*

$$x_i^n(R) = \begin{cases} l_i & \text{if } \chi^i < \alpha_i^i, \\ a_i \chi^i & \text{if } \alpha_i^i \leq \chi^i < \beta_i^i, \\ u_i & \text{if } \beta_i^i \leq \chi^i. \end{cases} \quad (4.9)$$

Proof. See Appendix A.1.3. □

Note that, starting from $\chi^n = \kappa^n$ and using Lemma 4.4, we can compute the values χ^j recursively as

$$\chi^j = \begin{cases} \kappa^j & \text{if } \chi^{j+1} \leq \kappa^j, \\ \lambda^j & \text{if } \chi^{j+1} \geq \lambda^j, \\ \chi^{j+1} & \text{otherwise.} \end{cases} \quad (4.10)$$

Thus, given the optimal Lagrange multipliers κ and λ , we can compute x in $O(n)$ time as $x^n(R)$ using the two simple recursions in Equations (4.9) and (4.10).

4.3.4 AN $O(n \log n)$ TIME ALGORITHM

In the previous two subsections, we derived an efficient approach to compute the optimal Lagrange multipliers κ and λ for the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ and to compute from these multipliers the optimal solution x . In this subsection, we combine these two ingredients to formulate a fast and efficient algorithm for Problem QRA-NC (Algorithm 4.2). More precisely, in the first part of this subsection, we present our algorithm and discuss several of its details regarding the subroutines for computing the optimal Lagrange multipliers of the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$. This includes several procedures that deal with corner cases and with the updating of the breakpoint sets and the bookkeeping parameters. In the second part, we focus on the efficiency of the algorithm. In particular, we prove in Lemma 4.6 that the algorithm has an $O(n \log n)$ worst-case time complexity when using an appropriate data structure.

Description of the algorithm

Algorithm 4.2 captures our approach for solving Problem QRA-NC. First, in Lines 3-11, the algorithm initializes all problem parameters, the initial breakpoint values and breakpoint sets, and the initial bookkeeping parameters. Throughout the entire algorithm, four separate sets $\mathcal{A}$, $\mathcal{B}$, $\mathcal{K}$, and $\mathcal{L}$ of breakpoint values indicating the “source” of the values are maintained, i.e., they specify whether the corresponding value is one of the initial breakpoint values α_i^i or β_i^i or one of the optimal Lagrange multipliers κ^j or λ^j respectively. Second, in Lines 12-13, the algorithm applies for each $j \in \mathcal{N} \setminus \{1\}$ the procedure $\text{SOLVESUBPROBLEMS}(j)$ (see Algorithm 4.3) that computes the optimal Lagrange multipliers κ^j and λ^j for the two subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$. Finally, using the obtained vectors of optimal Lagrange multipliers κ and λ , the algorithm computes in Lines 14-18 the (alternative) multiplier values χ using the recursion in Equation (4.10) and from these values the solution $x^n(R)$ using Equation (4.9).

The procedure $\text{SOLVESUBPROBLEMS}(j)$ carries out the breakpoint search procedure for the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ as described in Section 3.3.1 (Lines 33-48). This is done by first initializing the bookkeeping parameters for these breakpoint search procedures in Lines 33-39 and Lines 41-47 and subsequently applying the procedures $\text{LOWERSUBPROBLEM}(j)$ (Line 40, Algorithm 4.4)

Algorithm 4.2: An $O(n \log n)$ time algorithm for Problem QRA-NC.

```

1 Input: Parameters  $a \in \mathbb{R}_{>0}^n$ ,  $L, U \in \mathbb{R}^{n-1}$ ,  $R \in \mathbb{R}$ , and  $l, u \in \mathbb{R}^n$ 
2 Output: Optimal solution  $x$  to Problem QRA-NC
3  $L_1 = \max(\bar{L}_1, l_1)$ ;  $U_1 = \min(U_1, u_1)$ 
4 for  $j = 2$  to  $n$  do
5    $L_j = \max(L_j, L_{j-1} + l_j)$ 
6    $U_j = \min(U_j, U_{j-1} + u_j)$ 
7 for  $i = 1$  to  $n$  do
8    $\alpha_i^i = \frac{l_i}{a_i}$ ;  $\beta_i^i = \frac{u_i}{a_i}$ 
9  $\kappa^1 = \alpha_1^1$ ;  $\lambda^1 = \beta_1^1$ ;  $\kappa^j = \infty$ ,  $\lambda^j = -\infty$  for  $j > 1$ 
10 Initialize breakpoint sets:  $\mathcal{A} := \{\alpha_1^1\}$ ;  $\mathcal{B} := \{\beta_1^1\}$ ;  $\mathcal{K} := \emptyset$ ;  $\mathcal{L} := \emptyset$ 
11 Initialize bookkeeping sums:  $\bar{P}_L^1 = \bar{P}_U^1 = 0$ ;  $\bar{Q}_L^1 = \bar{Q}_U^1 = a_1$ 
12 for  $j = 2$  to  $n$  do
13   Apply procedure SOLVESUBPROBLEMS( $j$ )
14  $\chi^n := \kappa^n$ 
15 for  $i = n - 1$  down to  $1$  do
16   Compute  $\chi^i$  using Equation (4.10)
17 Compute  $x^n(R)$  using Equation (4.9)
18 return  $\bar{x} := x^n(R)$ 

```

Algorithm 4.3: Procedure SOLVESUBPROBLEMS(j).

```

1 if  $L_{j-1} + \max(l_j, \min(a_j \kappa^{j-1}, u_j)) = L_j$  then
2    $\kappa^j := \kappa^{j-1}$ ; replace  $\kappa^{j-1}$  in  $\mathcal{K}$  by  $\kappa^j$ 
3   if  $\kappa^j < \alpha_j^j$  then
4      $\bar{P}_L^j := \bar{P}_L^{j-1} + l_j$ ;  $\bar{Q}_L^j := \bar{Q}_L^{j-1}$ 
5   else if  $\beta_j^j < \kappa^j$  then
6      $\bar{P}_L^j = \bar{P}_L^{j-1} + u_j$ ;  $\bar{Q}_L^j := \bar{Q}_L^{j-1}$ 
7   else
8      $\bar{P}_L^j := \bar{P}_L^{j-1}$ ;  $\bar{Q}_L^j := \bar{Q}_L^{j-1} + a_j$ 
9 else if  $L_{j-1} + \max(l_j, \min(a_j \kappa^{j-1}, u_j)) > L_j$  then
10   ( $\kappa^j < \kappa^{j-1}$ ;)  $\kappa^j = (L_j - L_{j-1})/a_j$ 
11    $\bar{P}_L^j := L_{j-1}$ ;  $\bar{Q}_L^j := a_j$ 
12   Add  $\kappa^j$  to  $\mathcal{K}$ 
13 else
14   ( $\kappa^j > \kappa^{j-1}$ ;) remove  $\kappa^{j-1}$  from  $\mathcal{K}$ 

```

```

15 if  $U_{j-1} + \max(l_j, \min(a_i \lambda^{j-1}, u_j)) = U_j$  then
16    $\lambda^j := \lambda^{j-1}$ ; replace  $\lambda^{j-1}$  in  $\mathcal{L}$  by  $\lambda^j$ 
17   if  $\lambda^j < \alpha_j^j$  then
18      $\bar{P}_U^j := \bar{P}_U^{j-1} + l_j$ ;  $\bar{Q}_U^j := \bar{Q}_U^{j-1}$ 
19   else if  $\beta_j^j < \lambda^j$  then
20      $\bar{P}_U^j := \bar{P}_U^{j-1} + u_j$ ;  $\bar{Q}_U^j := \bar{Q}_U^{j-1}$ 
21   else
22      $\bar{P}_U^j := \bar{P}_U^{j-1}$ ;  $\bar{Q}_U^j := \bar{Q}_U^{j-1} + a_j$ 
23 else if  $U_{j-1} + \max(l_j, \min(a_j \lambda^{j-1}, u_j)) < U_j$  then
24    $(\lambda^j > \lambda^{j-1}) \lambda^j = (U_j - U_{j-1})/a_j$ 
25    $\bar{P}_U^j := U_{j-1}$ ;  $\bar{Q}_U^j := a_j$ 
26   Add  $\lambda^j$  to  $\mathcal{L}$ 
27 else
28    $(\lambda^j < \lambda^{j-1})$  remove  $\lambda^{j-1}$  from  $\mathcal{L}$ 
29 if  $\min(\kappa^{j-1}, \kappa^j) < \alpha_j^j \leq \max(\lambda^{j-1}, \lambda^j)$  then
30   Add  $\alpha_j^j$  to  $\mathcal{A}$ 
31 if  $\min(\kappa^{j-1}, \kappa^j) \leq \beta_j^j < \max(\lambda^{j-1}, \lambda^j)$  then
32   Add  $\beta_j^j$  to  $\mathcal{B}$ 
33 if  $\kappa^j > \kappa^{j-1}$  then
34   if  $\kappa^{j-1} < \alpha_j^j$  then
35      $P := \bar{P}_L^{j-1} + l_j$ ;  $Q := \bar{Q}_L^{j-1}$ 
36   else if  $\alpha_j^j \leq \kappa^{j-1} < \beta_j^j$  then
37      $P := \bar{P}_L^{j-1}$ ;  $Q := \bar{Q}_L^{j-1} + a_j$ 
38   else
39      $P := \bar{P}_L^{j-1} + u_j$ ;  $Q := \bar{Q}_L^{j-1}$ 
40   Apply procedure LOWERSUBPROBLEM( $j$ )
41 if  $\lambda^j < \lambda^{j-1}$  then
42   if  $\beta_j^j < \lambda^{j-1}$  then
43      $P := \bar{P}_U^{j-1} + u_j$ ;  $Q := \bar{Q}_U^{j-1}$ 
44   else if  $\alpha_j^j < \lambda^{j-1} \leq \beta_j^j$  then
45      $P := \bar{P}_U^{j-1}$ ;  $Q := \bar{Q}_U^{j-1} + a_j$ 
46   else
47      $P := \bar{P}_U^{j-1} + l_j$ ;  $Q := \bar{Q}_U^{j-1}$ 
48   Apply procedure UPBERSUBPROBLEM( $j$ )

```

Algorithm 4.4: Procedure LOWERSUBPROBLEM(j).

```

1  repeat
2      Choose minimum to-be-considered breakpoint:  $\delta := \max(\mathcal{A}, \mathcal{B}, \mathcal{K}, \mathcal{L})$ 
        and corresponding member set  $\mathcal{D} \in \{\mathcal{A}, \mathcal{B}, \mathcal{K}, \mathcal{L}\}$ 
3      if  $P + Q\delta = L_j$  then
4           $\kappa^j := \delta$ ; add  $\kappa^j$  to  $\mathcal{K}$ 
5           $\bar{P}_L^j := P, \bar{Q}_L^j := Q$ 
6          return
7      else if  $P + Q\delta > L_j$  then
8          ( $\kappa^j < \delta$ ):  $\kappa^j := (L_j - P)/Q$ ; add  $\kappa^j$  to  $\mathcal{K}$ 
9           $\bar{P}_L^j := P, \bar{Q}_L^j := Q$ 
10         return
11     else
12         ( $\kappa^j > \delta$ ): breakpoint  $\delta$  will be considered
13         if  $\mathcal{D} \equiv \mathcal{A}$  then
14             Let breakpoint be  $\delta \equiv \alpha_k^k$ 
15              $P := P - l_k; Q := Q + a_k$ 
16             Remove  $\alpha_k^k$  from  $\mathcal{A}$ 
17         else if  $\mathcal{D} \equiv \mathcal{B}$  then
18             Let breakpoint be  $\delta \equiv \beta_k^k$ 
19              $P := P + u_k; Q := Q - a_k$ 
20             Remove  $\beta_k^k$  from  $\mathcal{B}$ 
21         else if  $\mathcal{D} \equiv \mathcal{K}$  then
22             Let breakpoint be  $\delta \equiv \kappa^k$ 
23              $P := P - \bar{Q}_L^k \kappa^k, Q := Q + \bar{Q}_L^k$ 
24             Remove  $\kappa^k$  from  $\mathcal{K}$ 
25         else
26             Let breakpoint be  $\delta \equiv \lambda^k$ 
27              $P := P + \bar{Q}_U^k \lambda^k; Q := Q - \bar{Q}_U^k$ 
28             Remove  $\lambda^k$  from  $\mathcal{L}$ 

```

and UPPERSUBPROBLEM(j) (Line 48, Algorithm 4.5), which are identical in nature to Lines 5-20 of Algorithm 3.1. Before carrying out the breakpoint search procedure, two possible corner cases are considered in Lines 1-32 with regard to the relation between the to-be-computed multipliers κ^j and λ^j and their predecessors κ^{j-1} and λ^{j-1} . We briefly discuss these corner cases for κ^j ; the corner cases for λ^j are analogous.

The first corner case occurs when $\kappa^j = \kappa^{j-1}$ (Lines 1-8 in SOLVESUBPROBLEMS(j)). This case corresponds to Lines 7-9 in Algorithm 3.1, where the currently considered candidate multiplier δ_i leads to a solution $x[\delta_i]$ that sums

```

1  repeat
2      Choose maximum to-be-considered breakpoint:  $\delta := \min(\mathcal{A}, \mathcal{B}, \mathcal{K}, \mathcal{L})$ 
        and corresponding member set  $\mathcal{D} \in \{\mathcal{A}, \mathcal{B}, \mathcal{K}, \mathcal{L}\}$ 
3      if  $P + Q\delta = U_j$  then
4           $\lambda^j := \delta$ ; add  $\lambda^j$  to  $\mathcal{L}$ 
5           $\bar{P}_U^j := P, \bar{Q}_U^j := Q$ 
6          return
7      else if  $P + Q\delta < U_j$  then
8          ( $\lambda^j > \delta$ ):  $\lambda^j := (U_j - P)/Q$ ; add  $\lambda^j$  to  $\mathcal{L}$ 
9           $\bar{P}_U^j := P, \bar{Q}_U^j := Q$ 
10         return
11     else
12         ( $\lambda^j < \delta$ ): breakpoint will be considered
13         if  $\mathcal{D} \equiv \mathcal{A}$  then
14             Let breakpoint be  $\delta \equiv \alpha_k^k$ 
15              $P := P + l_k, Q := Q - a_k$ 
16             Remove  $\alpha_k^k$  from  $\mathcal{A}$ 
17         else if  $\mathcal{D} \equiv \mathcal{B}$  then
18             Let breakpoint be  $\delta \equiv \beta_k^k$ 
19              $P := P - u_k; Q := Q + a_k$ 
20             Remove  $\beta_k^k$  from  $\mathcal{B}$ 
21         else if  $\mathcal{D} \equiv \mathcal{K}$  then
22             Let breakpoint be  $\delta \equiv \kappa^k$ 
23              $P := P + \bar{Q}_L^k \kappa^k, Q := Q - \bar{Q}_L^k$ 
24             Remove  $\kappa^k$  from  $\mathcal{K}$ 
25         else
26             Let breakpoint be  $\delta \equiv \lambda^k$ 
27              $P := P - \bar{Q}_U^k \lambda^k; Q := Q + \bar{Q}_U^k$ 
28             Remove  $\lambda^k$  from  $\mathcal{L}$ 
    
```

up to C , i.e., $z[\delta_i] = R$. For $\text{QRA-NC}^j(L_j)$, this case thus occurs if and only if $L_{j-1} + x_j^j[\kappa^j] = L_j$, i.e., if and only if $x_i^j(L_j) = x_i^{j-1}(L_{j-1})$ for all $i \in \mathcal{N}_{j-1}$ and $L_{j-1} + \max(l_j, \min(a_j \chi^j, u_j)) = L_j$. The second case (Lines 9-12) occurs when $\kappa^j < \kappa^{j-1}$ and corresponds to Lines 10-12 of Algorithm 3.4, where the candidate multiplier δ_i leads to a solution $x[\delta_i]$ whose sum is larger than R , i.e., $z[\delta_i] > R$. In $\text{QRA-NC}^j(L_j)$, this case occurs if and only if $L_{j-1} + x_j^j[\kappa^j] > L_j$, i.e., if and only if $x_i^j(L_j) = x_i^{j-1}(L_{j-1})$ for all $i \in \mathcal{N}_{j-1}$ and $L_{j-1} + \max(l_j, \min(a_j \chi^j, u_j)) > L_j$. In both cases, it is not necessary to

carry out the actual breakpoint search to find κ^j since either $\kappa^j = \kappa^{j-1}$ (the first case) or $\kappa^j = (L_j - L_{j-1})/a_j$ (the second case).

Whether or not one of the above-mentioned corner cases occurs partly determines whether or not we have to include the new initial breakpoint values α_j^j and β_j^j in the breakpoint search procedure. The algorithm makes this decision in Lines 29-32: α_j^j and β_j^j are included only if they are in between the lowest and highest breakpoint values that can be considered in the breakpoint search. This lowest value is κ^j if $\kappa^j \leq \kappa^{j-1}$ (when one of the two corner cases for κ^j occurs and thus this value has already been determined) and κ^{j-1} otherwise (when breakpoint search is required to find κ^j). Analogously, the highest value is λ^j if $\lambda^j \geq \lambda^{j-1}$ and λ^{j-1} otherwise.

Time complexity

We now establish the worst-case time complexity of Algorithm 4.2 by means of the following lemma:

Lemma 4.6. *Algorithm 4.2 can be implemented such that its worst-case time complexity is $O(n \log n)$.*

Proof. Observe that, if we exclude the four operations on the sets $\mathcal{A}$, $\mathcal{B}$, $\mathcal{K}$, and $\mathcal{L}$ of to-be-considered breakpoints, the remaining part of the algorithm has a total time complexity of $O(n)$.

For a given breakpoint set $\mathcal{D} \in \{\mathcal{A}, \mathcal{B}, \mathcal{K}, \mathcal{L}\}$, the corresponding four operations are finding the minimum and maximum breakpoint in $\mathcal{D}$ (Lines 2 and 16 in Algorithm 4.3 and Line 2 in Algorithms 4.4 and 4.5), inserting a breakpoint value in $\mathcal{D}$ (Lines 12, 26, 30, and 32 in Algorithm 4.3), and removing the minimum or maximum breakpoint from $\mathcal{D}$ (Lines 14 and 28 in Algorithm 4.3 and Lines 16, 20, 24, and 28 in Algorithms 4.4 and 4.5). As we showed in Section 4.3.2, each breakpoint value is inserted and removed at most once during the course of the algorithm. Moreover, in the worst case, we have to find the minimum and maximum breakpoint value in $\mathcal{D}$ at most n times. Thus, the total number of breakpoint set operations is $O(n)$. If we maintain the breakpoint sets as min-max heaps [11], each of these operations can be executed in $O(1)$ (finding the minimum and maximum) and $O(\log n)$ (inserting and removing a breakpoint) time. This means that the total time complexity of all four breakpoint set operations is $O(n \log n)$ if we use min-max heaps to store the breakpoint sets. It follows that Algorithm 4.2 can be implemented such that its worst-case time complexity is $O(n \log n)$. $\square$

In practice, carrying out the breakpoint set operations might be faster if we use a different data structure than min-max heaps to maintain the breakpoint sets $\mathcal{A}$, $\mathcal{B}$, $\mathcal{K}$, and $\mathcal{L}$. For instance, when n is small, simple arrays might be sufficient for fast insertion and removal of breakpoints, even though this increases the

worst-case time complexity to $O(n^2)$. On the other hand, [75] suggests to keep the breakpoint sets by means of a so-called disjoint set data structure (see, e.g., [38]). Using such a structure, a sequence of $O(n)$ breakpoint insertions and deletions in sets of size at most n can be done in $O(n)$ time using the algorithm in [52]. However, it is unclear whether the algorithm in [52] is fast in practice for two reasons. First, it is complicated and cumbersome to implement compared to other algorithms for insertion and removal operations on disjoint set data structures [53]. Second, although the authors mention in the preliminary study [51] that their algorithm outperforms the then state-of-the-art, the literature contains hardly if any studies on its practical performance. If possible, one can use any other algorithm (e.g., those evaluated in [139]) even if it has a worse worst-case time complexity, as long as it has been shown to be fast in practice.

4.4 EVALUATION

In this section, we evaluate the performance of our Algorithm 4.2 as presented in Section 4.3.4, to which we shall refer as ALG_{seq} for clarity, and compare it with the state-of-the-art algorithms ALG_{inf} from [167] and ALG_{dec} from [172]. We carry out two types of experiments. First, we evaluate the performance of our algorithm on realistic instances of the battery scheduling problem BATTERY. Here, we take as objective load profile flattening, meaning that the cost functions take the form $f^t(x^t) = (x^t + p^t)^2$ for each $t \in \mathcal{T}$. For this, we tailor ALG_{seq} to this specific problem and compare this implementation to a tailored implementation of ALG_{inf} within the simulation tool DEMKit [82]. Second, we compare the execution time and scalability of our algorithm and of ALG_{inf} and ALG_{dec} on synthetic instances with sizes ranging from 10 to one million variables. We have implemented all three algorithms in Python (version 3.5) to be able to compare them to the implementation in DEMKit, which is also written in Python.

In Section 4.4.1, we describe in more detail the problem instances that we use in the evaluation. Subsequently, in Section 4.4.2, we discuss several implementation choices and in Section 4.4.3 we present and discuss the results of our evaluation.

4.4.1 PROBLEM INSTANCES

For the comparison of the tailored implementation of our algorithm ALG_{seq} with the tailored implementation of ALG_{inf} within DEMKit, we generate realistic instances of Problem BATTERY. For this, we consider the setting where a battery charging schedule for two consecutive days needs to be computed. This scheduling horizon is divided into 15-minute time intervals, resulting in $n = 192$. To study the influence of the battery size on the solving time, we consider three scenarios that correspond to three different battery sizes and denote them by SMALL, MEDIUM, and LARGE. In these scenarios, the battery capacity is chosen to be 20 kWh, 100 kWh, or 180 kWh and the (dis)charging rate is 4 kW, 20 kW,

or 36 kW respectively. This leads to $\Delta t = \frac{1}{4}$ and to the values for $X_{\min}$, $X_{\max}$, and D as given in Table 4.2. These parameter choices are made such that they represent a scenario where either 10, 50, or 90 percent of the households have installed a smaller “home” battery with a capacity of 5 kWh and a (dis)charging rate of 1 kW (the latter values correspond to a situation as given in real-life field tests such as described in [142]). We set both the initial and target amount of energy (called the initial and target state-of-charge (SoC)) to a given fraction of the capacity, i.e., $S_{\text{start}} = S_{\text{end}} = sD$, where $s \in \{0, 0.1, 0.2, \dots, 1\}$. For each scenario, we simulate 50 battery schedules over two days. As input for the base load p , we use measurement data of the actual power consumption of 40 households for 100 consecutive days that were obtained in the field test described in [80].

	$X_{\min}$	$X_{\max}$	D
SMALL	$-4.0 \cdot 10^3$	$4.0 \cdot 10^3$	$8.0 \cdot 10^4$
MEDIUM	$-2.0 \cdot 10^4$	$2.0 \cdot 10^4$	$4.0 \cdot 10^5$
LARGE	$-3.6 \cdot 10^4$	$3.6 \cdot 10^4$	$7.2 \cdot 10^5$

Table 4.2: Parameter choices for the battery scheduling problem for each scenario.

For the scalability analysis, we generate synthetic instances in the same way as in [172]. We consider instance sizes n in the set $\{10, 20, 50, 100, \dots, 10^6\}$ and for each of these sizes, we generate 10 instances. In each instance, we sample the parameters a , l , and u from the uniform distributions $U(0, 1)$, $U(0.1, 0.5)$, and $U(0.5, 0.9)$ respectively. To generate the nested bounds L and U , we first draw for each $i \in \mathcal{N}$ two values X_i and Y_i from the uniform distribution $U(l_i, u_i)$. Subsequently, we define for each $j \in \mathcal{N}$ the values $v_j := \sum_{i \in \mathcal{N}_j} X_i$ and $w_j := \sum_{i \in \mathcal{N}_j} Y_i$ and we set $L_j := \min(v_j, w_j)$ and $U_j := \max(v_j, w_j)$ for $j < n$ and $R = L_n = U_n = \frac{1}{2}(v_n + w_n)$.

4.4.2 IMPLEMENTATION DETAILS

In both the divide-and-conquer algorithm ALG_{dec} and the infeasibility-guided algorithm ALG_{inf} , we use Algorithm 3.1 to solve the QRA subproblems. Note that using this algorithm instead of linear-time algorithms such as in [100] increases the worst-case time complexity of ALG_{dec} and ALG_{inf} by a factor $O(\log n)$. However, for practically relevant problems sizes, this procedure is generally faster in practice and easier to implement than the linear-time algorithms in, e.g., [100].

For the double-ended queues needed in ALG_{seq} to store the optimal Lagrange multipliers κ and λ , we use the Python container datatype `deque`. Moreover, we initially implemented the double-ended priority queues for the lower and upper

initial breakpoint values $(\alpha_i^i)_{i \in \mathcal{N}}$ and $(\beta_i^i)_{i \in \mathcal{N}}$ as symmetric min-max heaps [9]. However, initial tests indicated that using instead a coupled min-heap and max-heap implementation leads to similar or even lower execution times of the overall algorithm. Moreover, the latter data structure is much simpler to implement using the standard Python library `heapq`. Therefore, we use this method instead of min-max heaps. In this alternative method, we insert new breakpoints in both the min-heap and the max-heap and use the min-heap to find and delete a minimum breakpoint (in the lower subproblems) and the max-heap to find and delete a maximum breakpoint (in the upper subproblems). Moreover, we assign to each breakpoint a flag that is 1 if the breakpoint has been removed from either of the heaps and 0 otherwise. This prevents that we find a minimum (maximum) breakpoint in the min-heap (max-heap) that was already considered in the other heap and thus has been removed from the breakpoint search.

4.4.3 RESULTS AND DISCUSSION

In this section, we present and discuss the results of our evaluation. First, we discuss the results of the comparison of the tailored implementation of ALG_{seq} with the tailored implementation of ALG_{inf} within DEMKit. Figure 4.1 shows the ratios between the execution times of the tailored implementation of ALG_{inf} and that of ALG_{seq} . Moreover, Tables 4.3-4.5 contain for each scenario and each initial and target SoC value the mean, maximum, and coefficient of variation (CoV) of the execution times. The CoV is the sample deviation divided by the sample mean and is a suitable measure of the variation between samples when comparing different collections of samples with significantly different sample means.

s	ALG_{seq}			ALG_{inf} within DEMKit		
	Mean (s)	Max (s)	CoV	Mean (s)	Max (s)	CoV
0	$1.80 \cdot 10^{-3}$	$2.09 \cdot 10^{-3}$	$5.80 \cdot 10^{-2}$	$6.15 \cdot 10^{-3}$	$7.23 \cdot 10^{-3}$	$7.61 \cdot 10^{-2}$
0.1	$1.78 \cdot 10^{-3}$	$2.14 \cdot 10^{-3}$	$6.18 \cdot 10^{-2}$	$6.15 \cdot 10^{-3}$	$7.16 \cdot 10^{-3}$	$6.52 \cdot 10^{-2}$
0.2	$1.81 \cdot 10^{-3}$	$2.77 \cdot 10^{-3}$	$9.19 \cdot 10^{-2}$	$6.10 \cdot 10^{-3}$	$7.15 \cdot 10^{-3}$	$7.17 \cdot 10^{-2}$
0.3	$1.79 \cdot 10^{-3}$	$2.25 \cdot 10^{-3}$	$7.70 \cdot 10^{-2}$	$6.10 \cdot 10^{-3}$	$7.10 \cdot 10^{-3}$	$7.34 \cdot 10^{-2}$
0.4	$1.79 \cdot 10^{-3}$	$2.44 \cdot 10^{-3}$	$7.25 \cdot 10^{-2}$	$6.07 \cdot 10^{-3}$	$7.11 \cdot 10^{-3}$	$7.10 \cdot 10^{-2}$
0.5	$1.77 \cdot 10^{-3}$	$2.25 \cdot 10^{-3}$	$6.28 \cdot 10^{-2}$	$6.02 \cdot 10^{-3}$	$6.94 \cdot 10^{-3}$	$6.41 \cdot 10^{-2}$
0.6	$1.83 \cdot 10^{-3}$	$4.00 \cdot 10^{-3}$	$1.89 \cdot 10^{-1}$	$6.01 \cdot 10^{-3}$	$7.09 \cdot 10^{-3}$	$6.78 \cdot 10^{-2}$
0.7	$1.77 \cdot 10^{-3}$	$2.21 \cdot 10^{-3}$	$6.49 \cdot 10^{-2}$	$6.05 \cdot 10^{-3}$	$6.97 \cdot 10^{-3}$	$7.69 \cdot 10^{-2}$
0.8	$1.80 \cdot 10^{-3}$	$2.68 \cdot 10^{-3}$	$1.00 \cdot 10^{-1}$	$6.05 \cdot 10^{-3}$	$7.06 \cdot 10^{-3}$	$7.52 \cdot 10^{-2}$
0.9	$1.79 \cdot 10^{-3}$	$2.87 \cdot 10^{-3}$	$1.24 \cdot 10^{-1}$	$5.97 \cdot 10^{-3}$	$7.88 \cdot 10^{-3}$	$8.48 \cdot 10^{-2}$
1	$1.79 \cdot 10^{-3}$	$2.18 \cdot 10^{-3}$	$6.02 \cdot 10^{-2}$	$6.01 \cdot 10^{-3}$	$8.22 \cdot 10^{-3}$	$8.81 \cdot 10^{-2}$

Table 4.3: The mean, maximum, and coefficient of variation of the execution times of the tailored implementation of ALG_{seq} and the tailored implementation of ALG_{inf} within DEMKit for the scenario SMALL.

Tables 4.3-4.5 show that the mean execution time of ALG_{seq} is similar in each scenario, whereas that of ALG_{inf} appears to decrease as the battery size increases.

This implies that also the ratios between the execution times decrease as the battery size increases, which is confirmed by the boxplots in Figure 4.1. In particular, a smaller battery size seems to imply that ALG_{seq} is likely to be faster than ALG_{inf} whereas ALG_{inf} is likely to be faster for larger battery size. The reason for this is that the execution time of ALG_{inf} heavily depends on the number of tight nested constraints in an optimal solution. To support this fact, we plot in Figure 4.2 boxplots of these numbers. Note that when the initial SoC is 20% or 30% of the battery capacity in the scenario LARGE, in only 4 of the 50 instances the number of tight constraints was more than 1, meaning that in the remaining 46 instances the optimal solution to the relaxation of the problem did not violate any of the nested constraints. The relation between the number of tight nested constraints and the ratios is also strongly visible when comparing Figures 4.1 and 4.2: the ratios increase as the number of tight constraints increases.

From these results, we can derive a “rule of thumb” for the choice of a proper algorithm that depends on the expected number of tight nested constraints. To

Figure 4.1: Boxplots of the execution time of the tailored implementation of ALG_{inf} within DEMKit divided by that of the tailored implementation of ALG_{seq} for the three scenarios. Ratios larger than 1 imply that ALG_{seq} was faster than ALG_{inf} .

s	ALG _{seq}			ALG _{inf} within DEMKit		
	Mean (s)	Max (s)	CoV	Mean (s)	Max (s)	CoV
0	$1.71 \cdot 10^{-3}$	$2.40 \cdot 10^{-3}$	$1.19 \cdot 10^{-1}$	$1.88 \cdot 10^{-3}$	$3.45 \cdot 10^{-3}$	$3.10 \cdot 10^{-1}$
0.1	$1.64 \cdot 10^{-3}$	$1.96 \cdot 10^{-3}$	$7.39 \cdot 10^{-2}$	$1.45 \cdot 10^{-3}$	$3.31 \cdot 10^{-3}$	$4.47 \cdot 10^{-1}$
0.2	$1.66 \cdot 10^{-3}$	$2.40 \cdot 10^{-3}$	$1.03 \cdot 10^{-1}$	$1.47 \cdot 10^{-3}$	$3.34 \cdot 10^{-3}$	$4.78 \cdot 10^{-1}$
0.3	$1.76 \cdot 10^{-3}$	$3.50 \cdot 10^{-3}$	$1.85 \cdot 10^{-1}$	$1.68 \cdot 10^{-3}$	$3.43 \cdot 10^{-3}$	$4.41 \cdot 10^{-1}$
0.4	$1.65 \cdot 10^{-3}$	$2.36 \cdot 10^{-3}$	$1.07 \cdot 10^{-1}$	$1.76 \cdot 10^{-3}$	$3.51 \cdot 10^{-3}$	$3.45 \cdot 10^{-1}$
0.5	$1.66 \cdot 10^{-3}$	$2.42 \cdot 10^{-3}$	$1.08 \cdot 10^{-1}$	$1.95 \cdot 10^{-3}$	$3.27 \cdot 10^{-3}$	$2.83 \cdot 10^{-1}$
0.6	$1.65 \cdot 10^{-3}$	$2.48 \cdot 10^{-3}$	$9.75 \cdot 10^{-2}$	$2.23 \cdot 10^{-3}$	$3.52 \cdot 10^{-3}$	$2.26 \cdot 10^{-1}$
0.7	$1.68 \cdot 10^{-3}$	$2.09 \cdot 10^{-3}$	$8.46 \cdot 10^{-2}$	$2.32 \cdot 10^{-3}$	$3.42 \cdot 10^{-3}$	$1.85 \cdot 10^{-1}$
0.8	$1.64 \cdot 10^{-3}$	$1.94 \cdot 10^{-3}$	$6.79 \cdot 10^{-2}$	$2.36 \cdot 10^{-3}$	$3.19 \cdot 10^{-3}$	$1.61 \cdot 10^{-1}$
0.9	$1.70 \cdot 10^{-3}$	$2.17 \cdot 10^{-3}$	$1.02 \cdot 10^{-1}$	$2.53 \cdot 10^{-3}$	$3.50 \cdot 10^{-3}$	$1.63 \cdot 10^{-1}$
1	$1.67 \cdot 10^{-3}$	$2.11 \cdot 10^{-3}$	$8.41 \cdot 10^{-2}$	$2.61 \cdot 10^{-3}$	$4.05 \cdot 10^{-3}$	$1.54 \cdot 10^{-1}$

Table 4.4: The mean, maximum, and coefficient of variation of the execution times of the tailored implementation of ALG_{seq} and the tailored implementation of ALG_{inf} within DEMKit for the scenario MEDIUM.

s	ALG _{seq}			ALG _{inf} within DEMKit		
	Mean (s)	Max (s)	CoV	Mean (s)	Max (s)	CoV
0	$1.59 \cdot 10^{-3}$	$2.15 \cdot 10^{-3}$	$7.33 \cdot 10^{-2}$	$1.60 \cdot 10^{-3}$	$2.76 \cdot 10^{-3}$	$2.98 \cdot 10^{-1}$
0.1	$1.58 \cdot 10^{-3}$	$1.90 \cdot 10^{-3}$	$7.33 \cdot 10^{-2}$	$8.93 \cdot 10^{-4}$	$2.11 \cdot 10^{-3}$	$5.22 \cdot 10^{-1}$
0.2	$1.58 \cdot 10^{-3}$	$1.94 \cdot 10^{-3}$	$8.61 \cdot 10^{-2}$	$7.82 \cdot 10^{-4}$	$2.04 \cdot 10^{-3}$	$4.77 \cdot 10^{-1}$
0.3	$1.55 \cdot 10^{-3}$	$2.16 \cdot 10^{-3}$	$8.79 \cdot 10^{-2}$	$7.45 \cdot 10^{-4}$	$2.00 \cdot 10^{-3}$	$4.66 \cdot 10^{-1}$
0.4	$1.54 \cdot 10^{-3}$	$2.05 \cdot 10^{-3}$	$7.06 \cdot 10^{-2}$	$8.22 \cdot 10^{-4}$	$2.30 \cdot 10^{-3}$	$5.18 \cdot 10^{-1}$
0.5	$1.54 \cdot 10^{-3}$	$2.05 \cdot 10^{-3}$	$7.67 \cdot 10^{-2}$	$1.02 \cdot 10^{-3}$	$2.42 \cdot 10^{-3}$	$5.51 \cdot 10^{-1}$
0.6	$1.53 \cdot 10^{-3}$	$1.83 \cdot 10^{-3}$	$6.24 \cdot 10^{-2}$	$1.34 \cdot 10^{-3}$	$2.63 \cdot 10^{-3}$	$4.38 \cdot 10^{-1}$
0.7	$1.55 \cdot 10^{-3}$	$1.77 \cdot 10^{-3}$	$6.22 \cdot 10^{-2}$	$1.72 \cdot 10^{-3}$	$2.61 \cdot 10^{-3}$	$2.75 \cdot 10^{-1}$
0.8	$1.53 \cdot 10^{-3}$	$1.86 \cdot 10^{-3}$	$5.80 \cdot 10^{-2}$	$2.09 \cdot 10^{-3}$	$2.95 \cdot 10^{-3}$	$1.87 \cdot 10^{-1}$
0.9	$1.53 \cdot 10^{-3}$	$1.94 \cdot 10^{-3}$	$6.34 \cdot 10^{-2}$	$2.31 \cdot 10^{-3}$	$2.98 \cdot 10^{-3}$	$1.69 \cdot 10^{-1}$
1	$1.52 \cdot 10^{-3}$	$1.77 \cdot 10^{-3}$	$4.58 \cdot 10^{-2}$	$2.51 \cdot 10^{-3}$	$3.16 \cdot 10^{-3}$	$1.36 \cdot 10^{-1}$

Table 4.5: The mean, maximum, and coefficient of variation of the execution times of the tailored implementation of ALG_{seq} and the tailored implementation of ALG_{inf} within DEMKit for the scenario LARGE.

this end, we compute for each number of tight constraints the percentage of instances where the tailored implementation of ALG_{seq} runs faster than the tailored implementation of ALG_{inf} within DEMKit given the optimal solution has this particular number of tight nested constraints (see Table 4.6). These values suggest that when the number of tight constraints is more than $\frac{4}{192} \approx 2.1$ percent, our algorithm is faster in more than 50% of the instances. In particular, when the number of tight constraints is $\frac{7}{192} \approx 3.6$ percent or more, the tailored implementation of our algorithm ALG_{seq} is always faster.

Note that this rule-of-thumb is in line with the physical interpretation of tight nested constraints in Problem BATTERY. For this, note that a battery being

Figure 4.2: Boxplots of the number of tight constraints in the optimal solutions for the three scenarios.

Number of tight nested constraints	1	2	3	4	5	6	≥ 7
“Win” percentage	0.0	2.2	30.2	62.8	83.8	93.4	100

Table 4.6: Percentage of instances where the tailored implementation of ALG_{seq} is faster than the tailored implementation of ALG_{inf} within DEMKit given the number of tight nested constraints in their optimal solutions.

completely empty or full is equivalent to a nested constraint of Problem BATTERY being tight. When the charging rates of the battery are large, the battery is better able to, at a given moment, flatten large peaks or drops in power consumption. However, the latter is also dependent on whether there is enough space (energy) left in the battery to store (dispatch) this energy, which is more likely when the battery capacity is large. Thus, when adopting a large battery for load profile flattening, it is less likely that it will be completely empty or full.

Although the ratio between the execution times of ALG_{seq} and ALG_{inf} appears to depend significantly on the battery size, the maximum and CoV of the execution times of ALG_{seq} is on average around 1.9 and 3.0 times smaller than that of ALG_{inf} respectively. This means that the execution times of ALG_{seq} are on average more stable than those of ALG_{inf} , regardless of the battery size. For DEM in

general and DEMKit in particular, this is beneficial since the coordination and optimization of schedules for different devices are often done in parallel due to the decentralized nature of the coordination (see, e.g., [81]). As a consequence, the execution time of the entire coordination and optimization framework is constrained by the maximum execution time required for solving one (subset of) device-level optimization problem(s). Thus, using ALG_{seq} instead of ALG_{inf} within such a framework may significantly reduce the overall execution time of the framework.

In the following, we present and discuss some results of the scalability evaluation. Figure 4.3 shows the execution times of the three algorithms ALG_{seq} , ALG_{inf} , and ALG_{dec} , and Table 4.7 shows for each studied instance size n the mean and CoV of the execution times of the corresponding instances. The added regression lines in Figure 4.3 are the fitted power laws of the execution times, i.e., for each algorithm we fit the function $\phi(n) = c_1 \cdot n^{c_2}$ to the execution times. These lines indicate that the practical execution time of both ALG_{seq} and ALG_{dec} is close to $O(n)$ and that of ALG_{inf} is actually slightly less than $O(n)$. Thus, it can be expected that for very large values of n , more precisely for $n > 2.90 \cdot 10^9$, ALG_{inf} is on average faster than ALG_{seq} . However, the CoVs for ALG_{inf} are around one order of magnitude larger than those of both ALG_{seq} and ALG_{dec} . This suggests that the execution time of the latter two algorithms is significantly less affected by the choice of problem parameters than ALG_{inf} . This is in line with the results of the comparison of the tailored implementation of ALG_{seq} for Problem BATTERY with that of ALG_{inf} within DEMKit.

Figure 4.3: Execution times of ALG_{seq} (circles, black), ALG_{inf} (triangles, gray), and ALG_{dec} (squares, open).

n	Mean (s)			CoV		
	ALG _{seq}	ALG _{inf}	ALG _{dec}	ALG _{seq}	ALG _{inf}	ALG _{dec}
10	$9.47 \cdot 10^{-5}$	$3.17 \cdot 10^{-4}$	$1.90 \cdot 10^{-3}$	$7.40 \cdot 10^{-2}$	$3.94 \cdot 10^{-1}$	$2.74 \cdot 10^{-1}$
20	$1.92 \cdot 10^{-4}$	$4.92 \cdot 10^{-4}$	$4.17 \cdot 10^{-3}$	$1.03 \cdot 10^{-1}$	$2.40 \cdot 10^{-1}$	$1.10 \cdot 10^{-1}$
50	$4.82 \cdot 10^{-4}$	$1.30 \cdot 10^{-3}$	$1.14 \cdot 10^{-2}$	$9.69 \cdot 10^{-2}$	$3.21 \cdot 10^{-1}$	$8.19 \cdot 10^{-2}$
100	$1.08 \cdot 10^{-3}$	$2.14 \cdot 10^{-3}$	$2.37 \cdot 10^{-2}$	$2.46 \cdot 10^{-1}$	$3.37 \cdot 10^{-1}$	$3.88 \cdot 10^{-2}$
200	$2.62 \cdot 10^{-3}$	$4.01 \cdot 10^{-3}$	$5.57 \cdot 10^{-2}$	$2.45 \cdot 10^{-1}$	$2.75 \cdot 10^{-1}$	$1.58 \cdot 10^{-1}$
500	$5.29 \cdot 10^{-3}$	$1.02 \cdot 10^{-2}$	$1.39 \cdot 10^{-1}$	$7.11 \cdot 10^{-2}$	$3.70 \cdot 10^{-1}$	$4.51 \cdot 10^{-2}$
1,000	$1.06 \cdot 10^{-2}$	$1.82 \cdot 10^{-2}$	$2.89 \cdot 10^{-1}$	$4.61 \cdot 10^{-2}$	$4.09 \cdot 10^{-1}$	$4.56 \cdot 10^{-2}$
2,000	$2.29 \cdot 10^{-2}$	$4.05 \cdot 10^{-2}$	$6.11 \cdot 10^{-1}$	$8.10 \cdot 10^{-2}$	$2.33 \cdot 10^{-1}$	$1.79 \cdot 10^{-2}$
5,000	$5.82 \cdot 10^{-2}$	$9.49 \cdot 10^{-2}$	1.63	$3.83 \cdot 10^{-2}$	$3.50 \cdot 10^{-1}$	$3.10 \cdot 10^{-2}$
10,000	$1.170 \cdot 10^{-1}$	$1.91 \cdot 10^{-1}$	3.41	$4.05 \cdot 10^{-2}$	$3.38 \cdot 10^{-1}$	$2.50 \cdot 10^{-2}$
20,000	$2.37 \cdot 10^{-1}$	$3.85 \cdot 10^{-1}$	7.18	$3.75 \cdot 10^{-2}$	$2.63 \cdot 10^{-1}$	$1.60 \cdot 10^{-2}$
50,000	$6.19 \cdot 10^{-1}$	1.12	$1.94 \cdot 10^{01}$	$4.93 \cdot 10^{-2}$	$3.53 \cdot 10^{-1}$	$2.24 \cdot 10^{-2}$
100,000	1.28	2.35	$4.03 \cdot 10^{01}$	$3.20 \cdot 10^{-2}$	$2.57 \cdot 10^{-1}$	$1.56 \cdot 10^{-2}$
200,000	2.61	4.21	$8.40 \cdot 10^{01}$	$1.67 \cdot 10^{-2}$	$2.59 \cdot 10^{-1}$	$1.35 \cdot 10^{-2}$
500,000	7.07	$1.09 \cdot 10^{01}$	$2.22 \cdot 10^{02}$	$4.11 \cdot 10^{-2}$	$1.76 \cdot 10^{-1}$	$1.11 \cdot 10^{-2}$
1,000,000	$1.56 \cdot 10^{01}$	$2.66 \cdot 10^{01}$	$4.66 \cdot 10^{02}$	$3.32 \cdot 10^{-2}$	$3.07 \cdot 10^{-1}$	$1.29 \cdot 10^{-2}$

Table 4.7: Mean and coefficient of variation of the execution times.

The results in Figure 4.3 and Table 4.7 indicate that on average ALG_{seq} is 27.2 times faster than ALG_{dec} and 1.95 times faster than ALG_{inf}. With regard to the performance of ALG_{dec}, we acknowledge that ALG_{dec} and in particular the updating scheme for the single-variable bounds can probably be implemented more efficiently than in the current implementation. To reduce the influence of the overall implementation on the results of this study, we measured the total time that is spent in ALG_{dec} on solving QRA subproblems and compared this to the execution times of ALG_{seq} and ALG_{inf}. This 'reduced' time represents the time that is minimally required to solve all QRA subproblems regardless of the implementation of the scheme used to update the single-variable bounds. These measurements indicate that on average around 59% of the total execution time of ALG_{dec} is spent on solving QRA subproblems. However, this time is still on average 15.9 times more than the execution time of ALG_{seq} and 9.5 times more than that of ALG_{inf}.

4.5 CONCLUSIONS

In this chapter, we proposed an $O(n \log n)$ time algorithm for quadratic resource allocation problems (RAPs) with lower and upper bound constraints on nested sums of variables (Problem QRA-NC). As opposed to existing algorithms with the same time complexity, our algorithm can achieve the $O(n \log n)$ time complexity using only basic data structures and is therefore easier to implement. In computational experiments, we demonstrate the good practical performance of our approach, both on synthetic instances and on realistic instances from the application area of decentralized energy management (DEM) for smart grids.

Our approach builds upon monotonicity arguments that find their origin in the validity of greedy algorithms for convex optimization problems over poly-

matroids [74, 75]. Such monotonicity arguments have been primarily studied for resource allocation problems where the objective function is separable, i.e., can be written as the sum of single-variable functions. In the next chapter, we prove the validity of similar monotonicity arguments to solve a *nonseparable* RAP with generalized bound constraints. Moreover, recent results on the use of interior-point methods for nested resource allocation problems [156, 176] suggest that incorporating specific nonseparable terms in the objective function does not increase the complexity of the used solution method. Thus, one interesting direction for future research is to investigate whether one can use monotonicity arguments to derive efficient algorithms for RAPs over nested constraints with nonseparable objective functions.

With regard to the application of Problem QRA-NC within DEM systems, we compared our algorithm with an existing implementation of the state-of-the-art algorithm in [167] within a simulation tool for DEM research. One of our objectives was to decide which of these two algorithms is more suitable to be used for a given (type of) problem instance. It would be worthwhile to conduct a more thorough comparison and to develop an automated procedure to decide which algorithm is most likely to be faster. Moreover, the nonseparable version of the studied problem mentioned already in the previous paragraph is related to energy management of batteries in three-phase distribution networks, where load profile flattening on all three phases together is required to avoid blackouts in these networks [80, 174]. Thus, this further emphasizes that research in this direction may be of value also in the context of DEM.

5

NONSEPARABLE QUADRATIC RESOURCE ALLOCATION WITH GENERALIZED BOUND CONSTRAINTS

ABSTRACT – *In this chapter, we study a quadratic nonseparable resource allocation problem that arises in the area of decentralized energy management (DEM), where unbalance in electricity networks has to be minimized. The problem is to allocate a given resource over a set of activities that are partitioned into subsets, whereby a cost is assigned to the overall allocated amount of resources both to individual activities and to activities within the same subset. We derive two efficient $O(n \log n)$ time algorithms to solve this problem. For the special case where all subsets have the same size, one of these algorithms even runs in linear time given the subset size. Numerical evaluations on both real and synthetic data confirm the theoretical efficiency of both algorithms and demonstrate their suitability for integration in DEM systems.*

5.1 INTRODUCTION

The resource allocation problems (RAPs) studied in the previous chapters of this thesis had separable objective functions, i.e., a single cost function was associated with each activity. However, in this chapter, we focus on a quadratic *nonseparable* RAP. In this problem, next to costs for each individual activity, we also consider costs on sums of allocations to subsets of the activities. We call this problem QRA-NonSep-GBC, where the part "GBC" expresses that the subsets form a partition of the whole set of activities and thus have the same structure as in the RAP with generalized bound constraints (see Section 2.3.2).

This chapter is based on and contains parts of [SU:4] and [SU:9].

Our interest in studying this problem stems from its application in decentralized energy management (DEM). More precisely, we are interested in the local optimization of a specific device class within DEM, namely the scheduling of electric vehicles (EVs) that are equipped with a three-phase charger. This means that the EV can distribute its charging power arbitrarily over all the three phases of the distribution network. Recent studies show that three-phase EV charging, as opposed to single-phase EV charging, can help to reduce losses in the electricity grid, reduce the stress on grid assets, and thereby prevent outages caused by a high penetration of EVs charging simultaneously on a single phase [174]. We discuss this topic in more detail in Section 5.2 and we show that the three-phase EV charging problem can indeed be modeled as an instance of Problem QRA-NonSep-GBC.

In this chapter, we present two $O(n \log n)$ time algorithms for Problem QRA-NonSep-GBC. Our algorithms are, in their essence, breakpoint search algorithms similar to Algorithms 3.1 and 3.2 for the quadratic RAP with box constraints (Problem QRA) and Algorithm 4.2 for the quadratic RAP with nested constraints (Problem QRA-NC). This means that our algorithms are based on the Lagrangian relaxation of the original problem and search for the optimal Lagrange multiplier associated with the resource constraint. For this, we exploit the structure of the well-known Karush-Kuhn-Tucker (KKT) optimality applied to this problem (see, e.g., [29]). We show that for (strictly) convex instances of Problem QRA-NonSep-GBC, this leads to a monotonicity relation between the Lagrange multiplier and the solution to the Lagrangian relaxation, i.e., to the KKT-conditions. For the case where all subsets have the same size K , we show that one of the derived algorithms runs in $O(n \log K)$ time, meaning that for a given K , this algorithm has a linear time complexity. We show, furthermore, that this property of same-sized subsets holds in particular for the three-phase EV charging problem and that thereby this problem can be solved in $O(n)$ time.

We evaluate the performance of our algorithms on both realistic instances of the three-phase EV charging problem and synthetically generated instances of different sizes. These evaluations indicate that our algorithms are suitable for integration in DEM systems since they are fast in practice and do not require much memory. Furthermore, they show that our algorithms scale well when the number of subsets or their sizes increase. More precisely, the evolution of their execution time matches the theoretical worst-case complexity of $O(n \log n)$. In fact, we show that our algorithms are capable of outperforming the commercial solver MOSEK by two orders of magnitude for instances of up to one million variables.

The remainder of this chapter is organized as follows. In Section 5.2, we formulate Problem QRA-NonSep-GBC and explain in more detail the application of this problem in DEM and, specifically, in three-phase EV scheduling. In Section 5.3, we analyze the structure of Problem QRA-NonSep-GBC and derive a crucial property of feasible solutions to the problem. We use this property

to derive our solution approach to solve Problem QRA-NonSep-GBC in Section 5.4 and in Section 5.5, we present two $O(n \log n)$ algorithms based on this approach. In Section 5.6, we evaluate the performance of our algorithms and, finally, Section 5.7 contains some concluding remarks.

5.2 PROBLEM FORMULATION AND MOTIVATION

In this section, we describe in more detail our motivation for studying Problem QRA-NonSep-GBC. For this, we first provide in Section 5.2.1 the mathematical formulation of this problem and some background on existing solution approaches. Subsequently, Section 5.2.2 provides a short introduction to load balancing in three-phase electricity networks and discusses the relevance of minimizing load unbalance. In Section 5.2.3, we formulate the three-phase EV charging problem and show that this problem is an instance of Problem QRA-NonSep-GBC.

5.2.1 PROBLEM FORMULATION AND BACKGROUND

We consider the following nonseparable RAP, which is a generalization of Problem QRA:

$$\begin{aligned} \text{QRAP-NonSep-GBC : } \min_{x \in \mathbb{R}^n} \quad & \sum_{j=1}^m \frac{1}{2} w_j \left(\sum_{i \in \mathcal{N}_j} x_i \right)^2 + \sum_{i=1}^n \left(\frac{1}{2} \frac{x_i^2}{a_i} + b_i x_i \right) \\ \text{s.t.} \quad & \sum_{i=1}^n x_i = R \end{aligned} \quad (5.1a)$$

$$L_j \leq \sum_{i \in \mathcal{N}_j} x_i \leq U_j, \quad j \in \{1, \dots, m\} \quad (5.1b)$$

$$l_i \leq x_i \leq u_i, \quad i \in \{1, \dots, n\}. \quad (5.1c)$$

where $b, l, u \in \mathbb{R}^n$, $a \in \mathbb{R}_{>0}^n$, $R \in \mathbb{R}$, and $w, L, U \in \mathbb{R}^m$ are given inputs. Furthermore, in this problem, a partition of the index set $\mathcal{N} := \{1, \dots, n\}$ into m disjoint subsets $\mathcal{N}_1, \dots, \mathcal{N}_m$ of size $n_1, \dots, n_j$ indexed by $\mathcal{M} := \{1, \dots, m\}$ is given. The objective function of Problem QRA-NonSep-GBC assigns for each subset $\mathcal{N}_j$ a cost to the sum of all allocated amounts associated with this subset and also costs to the individual amounts. Furthermore, Constraints (5.1b) and (5.1c) put bounds on the sum of all variables associated with each given subset and on the individual variables.

Problem QRA-NonSep-GBC can be classified as a quadratic *nonseparable* RAP with generalized bound constraints (Constraint (5.1b)). The nonseparability is due to the terms $(\sum_{i \in \mathcal{N}_j} x_i)^2$, which cannot be written as the sum of single-variable functions and are thus nonseparable. Choosing the factors w_j as zero makes that these nonseparable terms disappear and Problem QRA-NonSep-GBC

becomes the quadratic *separable* RAP with generalized bound constraints, i.e., (a, b) -Q/GBC/C in the classification of Section 2.2.3 (see also Section 2.3.2).

To the best of our knowledge, no tailored algorithm exists for Problem QRA-NonSep-GBC. However, Problem QRA-NonSep-GBC can be modeled as a minimum convex quadratic cost flow problem if $w \geq 0$. Therefore, this case can be solved in strongly polynomial time [169]. In fact, since the resulting network structure is series-parallel, it can be solved by the algorithms in [120, 163] in $O(n^2)$ time. However, when some of the factors w_j are negative, existing approaches for solving this type of flow problem do not apply anymore. This situation in particular occurs in the aforementioned EV scheduling problem in DEM, where the objective of minimizing load unbalance is modeled as an instance of Problem QRA-NonSep-GBC by setting some or all of the factors w_j to a negative number (see also Section 5.2.3). In fact, by allowing some factors w_j to be negative, convexity of the objective function is not guaranteed anymore. As a consequence, also general solvers for convex optimization problems may not be able to compute an optimal solution to the problem. Thus, one important question regarding Problem QRA-NonSep-GBC is for which parameter choices the problem is convex and thus for which parameter choices the problem is most likely to be efficiently solvable. We address this issue in Section 5.3.1.

5.2.2 LOAD BALANCING IN THREE-PHASE ELECTRICITY NETWORKS

Load balancing in electricity distribution grids has as goal to distribute the power consumption of a neighborhood over a given time horizon and over the three phases of the electricity distribution network such that peak consumption and unbalance between phases is minimized. Increased peak consumption occurs when the consumption is not spread out equally over the time horizon but instead is concentrated within certain time periods. This is generally seen as non-desirable since it induces an increase in energy losses and stress on grid assets, such as transformers, and can even lead to outages (see, e.g., [80]). As a consequence, many DEM systems take into account the minimization of peak consumption when scheduling, e.g., EV charging (see, e.g., [54, 58, 122]).

However, minimization of load unbalance between phases is hardly considered in optimization approaches for EV scheduling within DEM. To explain the relevance of load unbalance minimization, we provide in the following some background on three-phase electricity networks in general (for a more detailed and comprehensive introduction to this topic, we refer to [160] and [103]).

In residential electricity distribution networks (or, in general, low-voltage networks), electrical energy is transported by electric current that flows through a conductor (e.g., a wire). This current can be seen as a signal with a given frequency and amplitude that transports (alternating current) power, i.e., the average energy transported in each cycle. In principle, only one supply conductor is required to transport electrical energy between two points. However, it is more efficient to divide this energy over three bundled conductors whose

currents have the same frequency but an equidistant phase shift. This means that there is a phase difference of 120 degrees between each pair of conductors. Networks wherein the conductors are bundled in this way are referred to as *three-phase* networks, where the term “phase” generally refers to one of the three bundled conductors. Figure 5.1 illustrates the concept of three-phase systems.

In order to maximize the efficiency of a three-phase network, ideally the power flow over all three phases is equal. When this is not the case, negative effects similar to those of peak consumption can occur, i.e., energy losses, wearing of grid assets, and outages. With the increasing penetration of single-phase connected EVs in the low-voltage network, actively maintaining load balance becomes important. This is mainly because the power consumption of an EV is in general much larger than the average power consumption of a household (see, e.g., [147]) and most EVs, especially in the Netherlands, are connected to only one of the three phases. As a consequence, when charging multiple EVs simultaneously, large load unbalance can occur when the (charging of the) EVs are (is) not divided equally over the phases [80].

Recently, [174] explored the potential of three-phase EV charging, i.e., allowing an EV to distribute its charging over the three phases for minimizing load unbalance. This study suggests that three-phase EV charging can significantly reduce the distribution losses and stress on the grid compared to single-phase EV charging, even when using the same DEM methodology.

Figure 5.1: Schematic view of the three-phase system. I_1 , I_2 and I_3 represent the current on each of the three phases and ϕ_1 , ϕ_2 , and ϕ_3 represent the phase angles (with regard to the horizontal axis). The light gray arrows represent a balanced load distribution, whereas the black arrows represent load unbalance.

5.2.3 MODELING THE THREE-PHASE EV CHARGING PROBLEM

The problem of three-phase EV charging with the objective to minimize peak consumption and load unbalance can be modeled as an instance of Problem QRA-

NonSep-GBC. For this, we consider a division of the scheduling horizon into m equidistant time intervals of length Δt labeled by to $\mathcal{M} := \{1, \dots, m\}$. Furthermore, we define the set $\mathcal{P} := \{1, 2, 3\}$ as the set of phases. We introduce for each $j \in \mathcal{M}$ and $p \in \mathcal{P}$ the variable $z_{j,p}$ that denotes the power consumption of the EV drawn from phase p during time interval j . Moreover, we denote by $q_{j,p}$ the remaining household power consumption drawn from phase p during interval j . This consumption is assumed to be known. Furthermore, we assume that we know on forehand the total required energy that must be charged by the EV and denote this requirement by $\bar{R}$. Finally, we denote the minimum and maximum allowed power consumption from phase p during interval j by $\tilde{l}_{j,p}$ and $\tilde{u}_{j,p}$ respectively and the minimum and maximum allowed consumption from all three phases summed together by $\tilde{L}_j$ and $\tilde{U}_j$ respectively.

The objective of minimizing peak consumption can be achieved by “flattening out” the overall consumption as much as possible over the time intervals. As the term $\sum_{p=1}^3 (q_{j,p} + z_{j,p})$ represents the total power consumption during interval j , we model this objective by minimizing the function

$$\sum_{j \in \mathcal{M}} \left(\sum_{p=1}^3 (q_{j,p} + z_{j,p}) \right)^2.$$

For minimizing load unbalance, we aim to equally distribute the consumption during each time interval j over the three phases. We can model the objective of minimizing load unbalance by minimizing the function

$$\sum_{j \in \mathcal{M}} \left(\frac{3}{2} \sum_{p=1}^3 (q_{j,p} + z_{j,p})^2 - \frac{1}{2} \left(\sum_{p=1}^3 (q_{j,p} + z_{j,p}) \right)^2 \right) \quad (5.2)$$

(see Appendix 5.8 for the derivation of this expression). This leads to the following optimization problem that we denote by EV-3Phase

EV-3Phase :

$$\begin{aligned} \min_{z \in \mathbb{R}^{m \times 3}} \quad & W_1 \sum_{j \in \mathcal{M}} \left(\sum_{p=1}^3 (q_{j,p} + z_{j,p}) \right)^2 \\ & + W_2 \sum_{j \in \mathcal{M}} \left(\frac{3}{2} \sum_{p=1}^3 (q_{j,p} + z_{j,p})^2 - \frac{1}{2} \left(\sum_{p=1}^3 (q_{j,p} + z_{j,p}) \right)^2 \right) \\ \text{s.t.} \quad & \sum_{j \in \mathcal{M}} \sum_{p=1}^3 z_{j,p} \Delta t = \bar{R}, \\ & \tilde{L}_j \leq \sum_{p=1}^3 z_{j,p} \leq \tilde{U}_j, \quad j \in \mathcal{M} \\ & \tilde{l}_{j,p} \leq z_{j,p} \leq \tilde{u}_{j,p}, \quad j \in \mathcal{M}, p \in \mathcal{P}. \end{aligned}$$

Here, W_1 and W_2 are positive weights that express the trade-off between the two objectives. By choosing the parameters as given in Table 5.1, this problem becomes an instance of Problem QRA-NonSep-GBC (see also Appendix 5.8). Observe that if $W_2 > 2W_1$, the weights w_j are negative and thus Problem EV-3Phase cannot be solved as a minimum convex quadratic cost flow problem using, e.g., the algorithms in [120, 163].

In Problem QRA-NonSep-GBC		In Problem EV-3Phase	
$\mathcal{N}_j,$	$j \in \mathcal{M}$	$\mathcal{P} := \{1, 2, 3\}$	
$(x_i)_{i \in \mathcal{N}_j},$	$j \in \mathcal{M}$	$(z_{j,p})_{p \in \mathcal{P}},$	$j \in \mathcal{M}$
$w_j,$	$j \in \mathcal{M}$	$2W_1 - W_2$	
$a_i,$	$i \in \mathcal{N}$	$\frac{1}{3W_2}$	
$b_i,$	$j \in \mathcal{M}, i \in \mathcal{N}_j$	$(W_1 - \frac{1}{2}W_2) \sum_{p=1}^3 q_{j,p} + \frac{3}{2}W_2 q_{j,p}$	
R		$\frac{\bar{R}}{\Delta t}$	
$(l_i)_{i \in \mathcal{N}_j},$	$j \in \mathcal{M}$	$(l_{j,p})_{p \in \mathcal{P}}$	
$(u_i)_{i \in \mathcal{N}_j},$	$j \in \mathcal{M}$	$(\tilde{u}_{j,p})_{p \in \mathcal{P}}$	
$L_j,$	$j \in \mathcal{M}$	$\tilde{L}_j,$	$j \in \mathcal{M}$
$U_j,$	$j \in \mathcal{M}$	$\tilde{U}_j,$	$j \in \mathcal{M}$

Table 5.1: Modeling Problem EV-3Phase as an instance of Problem QRA-NonSep-GBC.

5.3 ANALYSIS

In this section, we consider the general version of Problem QRA-NonSep-GBC and derive some of its properties. First, in Section 5.3.1, we derive a necessary and sufficient condition on the vectors w and a for strict convexity of Problem QRA-NonSep-GBC. Moreover, we show that the three-phase EV charging problem as presented in Section 5.2.3 satisfies this condition. Second, in Section 5.3.2, we show that we may replace Constraint (5.1b) by equivalent single-variable constraints without changing the optimal solution to the problem. This greatly simplifies the derivation of our solution approach in Section 5.4. Third, in Section 5.3.3, we derive a structural property of optimal solutions to Problem QRA-NonSep-GBC that forms the crucial ingredient for our solution approach to solve the problem.

5.3.1 CONVEX INSTANCES OF PROBLEM QRA-NonSep-GBC

Since all constraints of Problem QRA-NonSep-GBC are linear, the problem is strictly convex if and only if the second-derivative matrix (the Hessian) of its objective function is positive definite. Since this objective function is separable

over the indices j , it suffices to investigate for each $j \in \mathcal{M}$ if the function

$$f_j((x_i)_{i \in \mathcal{N}_j}) := \frac{1}{2} w_j \left(\sum_{i \in \mathcal{N}_j} x_i \right)^2 + \sum_{i \in \mathcal{N}_j} \left(\frac{1}{2} \frac{x_i^2}{a_i} + b_i x_i \right)$$

is strictly convex. We do this by checking whether the Hessian H^j of f_j is positive definite. This Hessian is given by

$$H^j := w_j \bar{e} \bar{e}^\top + \text{diag}(a^j),$$

where $\bar{e}$ is the vector of ones of appropriate size and $a^j := (\frac{1}{a_i})_{i \in \mathcal{N}_j}$. Lemma 5.1 provides a characterization for which choices of w_j and a^j the Hessian H^j is positive definite. This characterization can also be obtained as a special case of Theorem 1 in [158].

Lemma 5.1. *H^j is positive definite if and only if $1 + w_j \sum_{i \in \mathcal{N}_j} a_i > 0$.*

Proof. See Appendix A.2.1. □

Lemma 5.1 implies that an instance of Problem QRA-NonSep-GBC is strictly convex if and only if $1 + w_j \sum_{i' \in \mathcal{N}_j} a_{i'} > 0$ for each $j \in \mathcal{M}$. To stress the importance of this relation and for future reference, we state this relation as a property:

Property 5.1. *For each $j \in \mathcal{M}$, it holds that $1 + w_j \sum_{i' \in \mathcal{N}_j} a_{i'} > 0$.*

For the remainder of this chapter, we consider only instances of Problem QRA-NonSep-GBC that satisfy Property 5.1. We conclude this subsection by observing that the parameters for Problem EV-3Phase satisfy this property:

$$1 + w_j \sum_{i \in \mathcal{N}_j} a_i = 1 + (2W_1 - W_2) \sum_{p=1}^3 \frac{1}{3W_2} = 1 + \frac{2W_1 - W_2}{W_2} = \frac{2W_1}{W_2} > 0.$$

5.3.2 CONSTRAINT ELIMINATION

In the previous subsection, we studied properties of the objective function of Problem QRA-NonSep-GBC. In contrast, in this section we focus on properties of the *constraints* of Problem QRA-NonSep-GBC. We show that we can reduce the complexity of the constraint set by replacing the generalized bound constraints (5.1b) by a set of single-variable constraints without changing the optimal solution to Problem QRA-NonSep-GBC. As these single-variable constraints can be integrated into the existing single-variable constraints (5.1c), we can focus without loss of generality on solving Problem QRA-NonSep-GBC without this constraint.

To derive this result, we first define for each $j \in \mathcal{M}$ and $C \in \mathbb{R}$ the following subproblem of Problem QRA-NonSep-GBC:

$$\text{QRA}^j(C) : \min_{x \in \mathbb{R}^n} \sum_{i \in \mathcal{N}_j} \left(\frac{1}{2} \frac{x_i^2}{a_i} + b_i x_i \right) \quad (5.3)$$

$$\text{s.t.} \quad \sum_{i \in \mathcal{N}_j} x_i = C, \quad (5.4)$$

$$l_i \leq x_i \leq u_i, \quad i \in \mathcal{N}_j. \quad (5.5)$$

Note that this problem is an instance of (a, b) -Q/Box/C, i.e., of Problem QRA when applying the variable transform $y_i = x_i + a_i b_i$. Moreover, note that this definition of $\text{QRA}^j(C)$ is slightly different from that in Section 4.2 with regard to the subproblems of Problem QRA-NC. In the latter case, the subproblems have a nested structure and the single-variable bounds within each subproblem are optimal solutions to previous subproblems. In contrast, now the subproblems have a disjoint structure and represent the problem obtained after "decoupling" the dependencies between (variables corresponding to) different subsets in the original problem by removing the resource constraint (5.1a) and setting $w_j = 0$.

Lemma 5.2 states the main result of this subsection, namely that optimal solutions to $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ for $j \in \mathcal{M}$ are component-wise valid lower and upper bounds on optimal solutions to Problem QRA-NonSep-GBC. The proof of this lemma is inspired by the proof of Lemma 6.2.1 in [74] and can be found in Appendix A.2.2.

Lemma 5.2. *For a given $j \in \mathcal{M}$, let $\underline{x}^j := (\underline{x}_i)_{i \in \mathcal{N}_j}$ and $\bar{x}^j := (\bar{x}_i)_{i \in \mathcal{N}_j}$ be optimal solutions to $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ respectively. Then there exists an optimal solution $x^* := (x_i^*)_{i \in \mathcal{N}}$ to Problem QRA-NonSep-GBC that satisfies $\underline{x}_i \leq x_i^* \leq \bar{x}_i$ for each $i \in \mathcal{N}_j$.*

Lemma 5.2 implies that adding the inequalities $\underline{x}_i \leq x_i \leq \bar{x}_i$, $i \in \mathcal{N}$ to the formulation of Problem QRA-NonSep-GBC does not cut off the optimal solution to the problem. Moreover, these inequalities imply the generalized bound constraints (5.1b) since we have for each $j \in \mathcal{M}$ that $\sum_{i \in \mathcal{N}_j} \underline{x}_i = L_j$ and $\sum_{i \in \mathcal{N}_j} \bar{x}_i = U_j$ by definition of $\underline{x}$ and $\bar{x}$. This means that Problem QRA-NonSep-GBC has the same optimal solution as the following problem:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \sum_{j \in \mathcal{M}} \frac{1}{2} w_j \left(\sum_{i \in \mathcal{N}_j} x_i \right)^2 + \sum_{i \in \mathcal{N}} \left(\frac{1}{2} \frac{x_i^2}{a_i} + b_i x_i \right) \\ \text{s.t.} \quad & \sum_{i \in \mathcal{N}} x_i = R \\ & \underline{x}_i \leq x_i \leq \bar{x}_i, \quad i \in \mathcal{N}. \end{aligned}$$

To compute the new variable bounds $\underline{x}_i$ and $\bar{x}_i$, we solve the $2m$ subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$. Since each subproblem is an instance of QRAP, this can be done in $O(n \log n)$ using Algorithm 3.1 (sequential breakpoint search, see Section 3.3.1) or in $O(n)$ time using Algorithm 3.2 (binary breakpoint search, see Section 3.3.2). Thus, in the remainder of this chapter and without loss of generality, we focus on solving Problem QRA-NonSep-GBC without the generalized bound constraints (5.1b).

5.3.3 MONOTONICITY OF OPTIMAL SOLUTIONS

In this section, we analyze Problem QRA-NonSep-GBC (without Constraint (5.1b)) and in particular the structure of its optimal solutions. More precisely, we study the KKT-conditions for this problem and derive a property of solutions satisfying all but one of these conditions. This property is the crucial ingredient for our solution approach for Problem QRA-NonSep-GBC since it allows us to apply breakpoint search strategies similar to those for Problem QRA in Chapter 3.

For convenience, we define $y_j := \sum_{i \in \mathcal{N}_j} x_i$ for $j \in \mathcal{M}$. The KKT-conditions for Problem QRA-NonSep-GBC are given as follows:

$$w_j y_j + \frac{x_i}{a_i} + b_i - \lambda + \mu_i = 0, \quad j \in \mathcal{M}, i \in \mathcal{N}_j, \quad (5.6a)$$

$$\sum_{i \in \mathcal{N}} x_i = R, \quad (5.6b)$$

$$l_i \leq x_i \leq u_i, \quad i \in \mathcal{N}, \quad (5.6c)$$

$$\mu_i^+(x_i - u_i) = 0, \quad i \in \mathcal{N}, \quad (5.6d)$$

$$\mu_i^-(x_i - l_i) = 0, \quad i \in \mathcal{N}, \quad (5.6e)$$

$$\lambda, \mu_i \in \mathbb{R}, \quad i \in \mathcal{N}. \quad (5.6f)$$

Here, μ_i^+ and μ_i^- are the positive and negative part of μ_i respectively; i.e., $\mu_i^+ = \max(0, \mu_i)$ and $\mu_i^- = \min(0, \mu_i)$. Assuming that Slater's condition holds [29], the KKT-conditions are necessary and sufficient for optimality. Moreover, since Problem QRA-NonSep-GBC is *strictly* convex, it has a unique optimal solution x^* .

For a given λ , let $(x(\lambda), \mu(\lambda)) \in \mathbb{R}^{2n}$ be the unique solution that satisfies all KKT-conditions (5.6) except (5.6b). Moreover, define $y_j(\lambda) := \sum_{i \in \mathcal{N}_j} x_i(\lambda)$ for $j \in \mathcal{M}$. It follows that $x(\lambda)$ is the optimal solution to Problem QRA-NonSep-GBC if and only if it satisfies KKT-condition (5.6b), i.e., if $\sum_{i \in \mathcal{N}} x_i(\lambda) = R$. The core of our solution approach is to find a value λ^* such that $\sum_{i \in \mathcal{N}} x_i(\lambda^*) = R$ and reconstruct the corresponding solution $x(\lambda^*)$ that, by definition, is optimal to Problem QRA-NonSep-GBC. We call λ^* an *optimal (Lagrange) multiplier*.

The main result of this section is Lemma 5.4, which states that each $x_i(\lambda)$ is a non-decreasing function in λ . This result allows us to use approaches for separable

RAPs such as those discussed in Section 3.3 to find λ^* . To prove Lemma 5.4, we first identify in Lemma 5.3 a useful relation between $x_i(\lambda)$ and $\mu_i(\lambda)$.

Lemma 5.3. *For any $\lambda_1, \lambda_2 \in \mathbb{R}$ and $i \in \mathcal{N}$, we have that $x_i(\lambda_1) < x_i(\lambda_2)$ implies $\mu_i(\lambda_1) \leq \mu_i(\lambda_2)$.*

Proof. Suppose $x_i(\lambda_1) < x_i(\lambda_2)$ for some $i \in \mathcal{N}$. Then $l_i \leq x_i(\lambda_1) < x_i(\lambda_2) \leq u_i$, which implies $x_i(\lambda_1) < u_i$ and $x_i(\lambda_2) > l_i$. Together with KKT-conditions (5.6d) and (5.6e), it follows that $\mu_i(\lambda_1) \leq 0$ and $\mu_i(\lambda_2) \geq 0$ respectively, which implies that $\mu_i(\lambda_1) \leq \mu_i(\lambda_2)$. $\square$

Lemma 5.4. *For any $\lambda_1, \lambda_2 \in \mathbb{R}$ with $\lambda_1 < \lambda_2$, it holds that $x_i(\lambda_1) \leq x_i(\lambda_2)$ for all $i \in \mathcal{N}$.*

Proof. See Appendix A.2.3. $\square$

Lemma 5.4 implies that the values $x_i(\lambda)$ are monotonically non-decreasing in λ . As a consequence, that all possible values for the optimal multiplier λ^* necessarily form a closed interval $I \subset \mathbb{R}$, i.e., $\lambda \in I$ if and only if $\sum_{i \in \mathcal{N}} x_i(\lambda) = R$. Thus, to simplify the discussion, we may assume in the derivation of our approach without loss of generality that the optimal multiplier λ^* is unique since λ^* is non-unique only in an extreme case. More precisely, λ^* is non-unique if and only if the interval I contains more than one point, which can happen only if for each index $i \in \mathcal{N}$ one of the two bound constraints 5.1c are tight for i in the optimal solution, i.e., for all $i \in \mathcal{N}$ either $x_i^* = l_i$ or $x_i^* = u_i$.

The monotonicity of the values $x_i(\lambda)$ forms the main ingredient for our solution approach to Problem QRA-NonSep-GBC, which we derive in Section 5.4. We conclude this section with two corollaries of Lemma 5.4 that we require for the derivation of this approach. The first corollary states that not only the values $x_i(\lambda)$ are non-decreasing in λ , but also the values $y_j(\lambda)$ for $j \in \mathcal{M}$. The second corollary is a stronger version of Lemma 5.3 for the case where $i \in \mathcal{N}_j$ with $w_j < 0$.

Corollary 5.1. *For any $\lambda_1, \lambda_2 \in \mathbb{R}$ with $\lambda_1 < \lambda_2$, it holds that $y_j(\lambda_1) \leq y_j(\lambda_2)$, $j \in \mathcal{M}$.*

Proof. Follows directly from Lemma 5.4. $\square$

Corollary 5.2. *If $w_j < 0$, then for any $\lambda_1, \lambda_2 \in \mathbb{R}$ with $\lambda_1 < \lambda_2$, it holds that $\mu_i(\lambda_1) \leq \mu_i(\lambda_2)$ for $i \in \mathcal{N}_j$.*

Proof. By Lemma 5.4, we have $x_i(\lambda_1) \leq x_i(\lambda_2)$. If this is strict, i.e., if $x_i(\lambda_1) < x_i(\lambda_2)$, then it follows from Lemma 5.3 that $\mu_i(\lambda_1) \leq \mu_i(\lambda_2)$. Otherwise, if

$x_i(\lambda_1) = x_i(\lambda_2)$, KKT-condition (5.6a) together with $w_j < 0$ and Corollary 5.1 implies

$$\begin{aligned} w_j y_j(\lambda_1) + \frac{x_i(\lambda_1)}{a_i} + b_i + \mu_i(\lambda_1) &= \lambda_1 < \lambda_2 \\ &= w_j y_j(\lambda_2) + \frac{x_i(\lambda_2)}{a_i} + b_i + \mu_i(\lambda_2) \\ &\leq w_j y_j(\lambda_1) + \frac{x_i(\lambda_1)}{a_i} + b_i + \mu_i(\lambda_2). \end{aligned}$$

It follows that $\mu_i(\lambda_1) < \mu_i(\lambda_2)$, proving the corollary. $\square$

5.4 SOLUTION APPROACH

In this section, we present our approach for solving Problem QRA-NonSep-GBC. First, in Section 5.4.1, we provide an outline of the approach based on the analysis conducted in Section 5.3. Second, Section 5.4.2 focuses in detail on several computational aspects of the approach.

5.4.1 OUTLINE

The monotonicity of $x_i(\lambda)$, proven in Lemma 5.4, has two important implications. First, for each $i \in \mathcal{N}$, there exist unique *breakpoints* $\alpha_i < \beta_i$ such that

$$\lambda \leq \alpha_i \Leftrightarrow x_i(\lambda) = l_i, \quad (5.7a)$$

$$\alpha_i < \lambda < \beta_i \Leftrightarrow l_i < x_i(\lambda) < u_i, \quad (5.7b)$$

$$\beta_i \leq \lambda \Leftrightarrow x_i(\lambda) = u_i. \quad (5.7c)$$

For now, we assume that these breakpoints are known. In Section 5.4.2, we discuss how they can be computed efficiently. The second implication of the monotonicity is that, given the optimal multiplier λ^* , we have

$$\lambda \leq \lambda^* \Rightarrow \sum_{i \in \mathcal{N}} x_i(\lambda) \leq \sum_{i \in \mathcal{N}} x_i(\lambda^*) = R, \quad (5.8a)$$

$$\lambda \geq \lambda^* \Rightarrow \sum_{i \in \mathcal{N}} x_i(\lambda) \geq \sum_{i \in \mathcal{N}} x_i(\lambda^*) = R. \quad (5.8b)$$

These two implications are the base to determine the optimal multiplier λ^* . Let us define the set of all breakpoints by $\mathcal{D} := \{\alpha_i \mid i \in \mathcal{N}\} \cup \{\beta_i \mid i \in \mathcal{N}\}$. Equations (5.7a)-(5.7c) imply that $\min(\mathcal{D}) \leq \lambda^* \leq \max(\mathcal{D})$. This means that there exist two consecutive breakpoints $\delta_1, \delta_2 \in \mathcal{D}$ such that $\delta_1 \leq \lambda^* < \delta_2$. Figure 5.2 illustrates the relation between $x(\lambda)$, $y(\lambda)$, the total resource R , the breakpoints in $\mathcal{D}$, and the special breakpoints δ_1 , λ^* , and δ_2 .

Figure 5.2: Illustrative example of the relation between $x(\lambda)$, $y(\lambda) = \sum_{i=1}^3 x_i(\lambda)$, R , and the special breakpoints α_i , β_i , δ_1 , λ^* , and δ_2 . In this example, $\delta_1 = \beta_3$, $\delta_2 = \beta_2$, and λ^* is represented by the black square.

The key to our approach is that once we have found δ_1 and δ_2 , we can easily compute λ^* and the resulting optimal solution $x(\lambda^*)$. To see this, note that by Equations (5.7a)-(5.7c) and by definition of δ_1 and δ_2 , we have for all $i \in \mathcal{N}$ that

$$\begin{aligned} x_i(\delta_2) = l_i &\Leftrightarrow x_i(\lambda^*) = l_i, \\ l_i < x_i(\delta_1) < u_i &\Leftrightarrow l_i < x_i(\lambda^*) < u_i, \\ x_i(\delta_1) = u_i &\Leftrightarrow x_i(\lambda^*) = u_i. \end{aligned}$$

As a consequence, we know that $x_i(\lambda^*) = l_i$ if $\alpha_i \geq \delta_2$ and $x_i(\lambda^*) = u_i$ if $\beta_i \leq \delta_1$. Thus, we may eliminate these variables from the problem. As a consequence for the remaining problem, the box constraints (5.1c) become redundant and Problem QRA-NonSep-GBC reduces to a quadratic optimization problem with a single equality constraint. We show in Section 5.4.2 that this specific structure allows us to derive an explicit expression for λ^* that can be determined in $O(n)$ time.

To find the breakpoint δ_1 , we may either consider all breakpoints in the set $\mathcal{D}$ monotonically or apply a binary search to $\mathcal{D}$. This is because the variable sum $y_i(\lambda) = \sum_{i \in \mathcal{N}} x_i(\lambda)$ induces an order on the breakpoints by Corollary 5.1. Moreover, we know by Equations (5.8a) and (5.8b) that δ_1 is the largest breakpoint λ in the set $\mathcal{D}$ such that $\sum_{i \in \mathcal{N}} x_i(\lambda) \leq R$. Note, that the two approaches to find δ_1 lead to different algorithms.

The kernel of both approaches is an efficient method to evaluate $x(\lambda)$ for any given $\lambda \in \mathbb{R}$ as this is the base for computing the breakpoint set $\mathcal{D}$, and to compute λ^* from δ_1 . We focus on each of these three aspects in the next subsection.

5.4.2 COMPUTATIONAL ASPECTS

In the approach outlined in Section 5.4.1, there are three quantities whose computation is not straightforward. These quantities are the solution $x(\lambda)$ (and thereby $y(\lambda)$) for a given $\lambda \in \mathbb{R}$, the set of breakpoints $\mathcal{D}$, and the optimal multiplier λ^* . In the following subsections, we discuss how these quantities can be computed efficiently.

Computing $x(\lambda)$ and $y(\lambda)$ for a given λ

To compute $x(\lambda)$ for a given λ , we need to find a feasible solution to the KKT-conditions (5.6) without (5.6b). We call these KKT-conditions the *primary* KKT-conditions. Instead of trying to derive $x(\lambda)$ directly from the primary KKT-conditions, we first determine which variables in $x(\lambda)$ are equal to one of their bounds and which ones are strictly in between their bounds. To this end, for each $j \in \mathcal{M}$, the set of variables $\mathcal{N}_j$ can be partitioned into the following sets:

$$\begin{aligned}\mathcal{N}_j^{\text{lower}}(\lambda) &:= \{i \in \mathcal{N}_j \mid x_i(\lambda) = l_i\}, \\ \mathcal{N}_j^{\text{upper}}(\lambda) &:= \{i \in \mathcal{N}_j \mid x_i(\lambda) = u_i\}, \\ \mathcal{N}_j^{\text{free}}(\lambda) &:= \{i \in \mathcal{N}_j \mid l_i < x_i(\lambda) < u_i\}.\end{aligned}$$

Based on Equations (5.7), the following equivalent definition of these sets can be achieved:

$$\mathcal{N}_j^{\text{lower}}(\lambda) = \{i \in \mathcal{N}_j \mid \alpha_i \geq \lambda\}, \quad (5.9a)$$

$$\mathcal{N}_j^{\text{upper}}(\lambda) = \{i \in \mathcal{N}_j \mid \beta_i \leq \lambda\}, \quad (5.9b)$$

$$\mathcal{N}_j^{\text{free}}(\lambda) = \{i \in \mathcal{N}_j \mid \alpha_i < \lambda < \beta_i\}. \quad (5.9c)$$

Thus, given the set $\mathcal{D}$ of breakpoints, we can easily determine these sets in $O(n)$ time by checking whether $\alpha_i \geq \lambda$, $\beta_i \leq \lambda$, or $\alpha_i < \lambda < \beta_i$.

Given the partition $(\mathcal{N}_j^{\text{lower}}(\lambda), \mathcal{N}_j^{\text{upper}}(\lambda), \mathcal{N}_j^{\text{free}}(\lambda))$, we can compute $x(\lambda)$ as follows. As $x_i(\lambda) = l_i$ for all $i \in \mathcal{N}_j^{\text{lower}}(\lambda)$ and $x_i(\lambda) = u_i$ for all $\mathcal{N}_j^{\text{upper}}(\lambda)$, it remains to compute $x_i(\lambda)$ for all $i \in \mathcal{N}_j^{\text{free}}(\lambda)$. Let

$$\begin{aligned}y_j^{\text{free}}(\lambda) &:= \sum_{i \in \mathcal{N}_j^{\text{free}}(\lambda)} x_i(\lambda), \\ y_j^{\text{fixed}}(\lambda) &:= \sum_{i \in \mathcal{N}_j \setminus \mathcal{N}_j^{\text{free}}(\lambda)} x_i(\lambda) = \sum_{i \in \mathcal{N}_j^{\text{lower}}(\lambda)} l_i + \sum_{i \in \mathcal{N}_j^{\text{upper}}(\lambda)} u_i.\end{aligned}$$

By KKT-conditions (5.6d)-(5.6e), we have $\mu_i(\lambda) = 0$ for each $i \in \mathcal{N}_j^{\text{free}}(\lambda)$. As a consequence, after substituting $y_j^{\text{fixed}}(\lambda)$ and $x_i(\lambda)$ for $i \in \mathcal{N}_j^{\text{lower}}(\lambda) \cup \mathcal{N}_j^{\text{upper}}(\lambda)$ into the primary KKT-conditions (5.6a) and (5.6c)-(5.6e), the only non-redundant primary KKT-conditions are (5.6a) and (5.6f) for $i \in \mathcal{N}_j^{\text{free}}(\lambda)$:

$$w_j y_j^{\text{free}}(\lambda) + w_j y_j^{\text{fixed}}(\lambda) + \frac{x_i(\lambda)}{a_i} + b_i - \lambda = 0, \quad j \in \mathcal{M}, i \in \mathcal{N}_j^{\text{free}}(\lambda), \lambda \in \mathbb{R}.$$

We show that the solution to these equations in terms of $x_i(\lambda)$ can be given in closed form. For convenience, we define the following quantities:

$$A_j(\lambda) := \sum_{\ell \in \mathcal{N}_j^{\text{free}}(\lambda)} a_\ell, \quad B_j(\lambda) := \sum_{\ell \in \mathcal{N}_j^{\text{free}}(\lambda)} a_\ell b_\ell.$$

Using, e.g., the Sherwood-Morrison formula (see, e.g., [17]), one can deduce and verify that the solution to the non-redundant primary KKT-conditions is

$$x_i(\lambda) = a_i \frac{\lambda - w_j y_j^{\text{fixed}}(\lambda) + w_j B_j(\lambda)}{1 + w_j A_j(\lambda)} - a_i b_i, \quad i \in \mathcal{N}_j^{\text{free}}(\lambda). \quad (5.10)$$

It follows that

$$\begin{aligned} y_j(\lambda) &:= y_j^{\text{free}}(\lambda) + y_j^{\text{fixed}}(\lambda) = -\frac{x_i(\lambda)}{a_i w_j} + \frac{\lambda - b_i}{w_j} \\ &= \frac{-\frac{\lambda}{w_j} + y_j^{\text{fixed}}(\lambda) - B_j(\lambda)}{1 + w_j A_j(\lambda)} + \frac{\lambda}{w_j} \\ &= \frac{y_j^{\text{fixed}}(\lambda) - B_j(\lambda)}{1 + w_j A_j(\lambda)} + \frac{A_j(\lambda)}{1 + w_j A_j(\lambda)} \lambda. \end{aligned} \quad (5.11)$$

Note that for each $j \in \mathcal{M}$, the value $y_j(\lambda)$ can be computed in $O(n_j)$ time given the breakpoint set $\mathcal{D}$. As a consequence, computing the sum $\sum_{j \in \mathcal{M}} y_j(\lambda) (= \sum_{i \in \mathcal{N}} x_i(\lambda))$ takes $O(n)$ time.

Computing the breakpoints

To compute the breakpoints, we exploit two important properties of these breakpoints that we state and prove in Lemmas 5.5 and 5.6. The first property is concerned with the value μ introduced in the KKT-conditions (5.6). Recall from KKT-conditions (5.6d) and (5.6e) that, for a given $\lambda \in \mathbb{R}$ and $i \in \mathcal{N}$, we have that $\mu_i(\lambda) \geq 0$ if $x_i(\lambda) = u_i$, $\mu_i(\lambda) = 0$ if $l_i < x_i(\lambda) < u_i$, and $\mu_i(\lambda) \leq 0$ if $x_i(\lambda) = l_i$. It follows from Equation (5.7) that $\mu_i(\lambda) \leq 0$ if $\lambda \leq \alpha_i$, $\mu_i(\lambda) = 0$ if $\alpha_i < \lambda < \beta_i$, and $\mu_i(\lambda) \geq 0$ if $\beta_i \leq \lambda$. Lemma 5.5 shows that $\mu_i(\lambda)$ is also 0 for $\lambda = \alpha_i$ and $\lambda = \beta_i$.

Lemma 5.5. *For all $i \in \mathcal{N}$, we have $\mu_i(\alpha_i) = \mu_i(\beta_i) = 0$.*

Proof. See Appendix A.2.4. $\square$

Next, Lemma 5.6 states that we can use the values $\mathcal{P}_j := \{\frac{l_i}{a_i} + b_i \mid i \in \mathcal{N}_j\}$ and $\mathcal{Q}_j := \{\frac{u_i}{a_i} + b_i \mid i \in \mathcal{N}_j\}$ to determine the order of the corresponding breakpoints.

Lemma 5.6. *For $j \in \mathcal{M}$ and $i, k \in \mathcal{N}_j$, we have:*

- » $\frac{l_i}{a_i} + b_i < \frac{l_k}{a_k} + b_k$ implies $\alpha_i \leq \alpha_k$, and;
- » $\frac{u_i}{a_i} + b_i < \frac{u_k}{a_k} + b_k$ implies $\beta_i \leq \beta_k$.

Proof. See Appendix A.2.5. $\square$

Based on Lemmas 5.5 and 5.6, the following strategy can be used to compute the breakpoints. From the KKT-condition (5.6a) for $i \in \mathcal{N}_j$, $j \in \mathcal{M}$, we have for the breakpoints α_i and β_i that

$$\alpha_i = w_j y_j(\alpha_i) + \frac{x_i(\alpha_i)}{a_i} + b_i + \mu_i(\alpha_i) = w_j y_j(\alpha_i) + \frac{l_i}{a_i} + b_i + \mu_i(\alpha_i), \quad (5.12a)$$

$$\beta_i = w_j y_j(\beta_i) + \frac{x_i(\beta_i)}{a_i} + b_i + \mu_i(\beta_i) = w_j y_j(\beta_i) + \frac{u_i}{a_i} + b_i + \mu_i(\beta_i). \quad (5.12b)$$

By applying Lemma 5.5 and plugging Equation (5.11) into Equation (5.12a), we can obtain the following expression for a given breakpoint α_i :

$$\begin{aligned} \alpha_i &= w_j y_j(\alpha_i) + \frac{l_i}{a_i} + b_i + \mu_i(\alpha_i) \\ &= \frac{w_j y_j^{\text{fixed}}(\alpha_i) - w_j B_j(\alpha_i)}{1 + w_j A_j(\alpha_i)} + \frac{w_j A_j(\alpha_i)}{1 + w_j A_j(\alpha_i)} \alpha_i + \frac{l_i}{a_i} + b_i. \end{aligned}$$

This is equivalent to

$$\alpha_i = w_j (y_j^{\text{fixed}}(\alpha_i) - B_j(\alpha_i)) + \left(\frac{l_i}{a_i} + b_i \right) (1 + w_j A_j(\alpha_i)). \quad (5.13)$$

Analogously, we can deduce the following expression for β_i by applying Lemma 5.5 and plugging Equation (5.11) into Equation (5.12b):

$$\beta_i = w_j (y_j^{\text{fixed}}(\beta_i) - B_j(\beta_i)) + \left(\frac{u_i}{a_i} + b_i \right) (1 + w_j A_j(\beta_i)). \quad (5.14)$$

Using these two expressions, we can compute the breakpoints sequentially in ascending order. Note that this order can be determined using Lemma 5.6 without knowledge of the actual breakpoint values. Furthermore, for each breakpoint

η_k , we can compute the terms $y_j^{\text{fixed}}(\eta_k)$, $A_j(\eta_k)$, and $B_j(\eta_k)$ efficiently from the preceding breakpoint η_i by exploiting the dependencies between the partitions $(\mathcal{N}_j^{\text{lower}}(\eta_i), \mathcal{N}_j^{\text{upper}}(\eta_i), \mathcal{N}_j^{\text{free}}(\eta_i))$ and $(\mathcal{N}_j^{\text{lower}}(\eta_k), \mathcal{N}_j^{\text{upper}}(\eta_k), \mathcal{N}_j^{\text{free}}(\eta_k))$ summarized in Table 5.2 (see also Figure 5.2 and Equation (5.9)). Given the smallest breakpoint $\bar{\eta}$, the sequential computation of the terms $y_j^{\text{fixed}}(\cdot)$, $A_j(\cdot)$, and $B_j(\cdot)$ is initialized by $y_j^{\text{fixed}}(\bar{\eta}) := \sum_{i \in \mathcal{N}_j} l_i$, $A_j(\bar{\eta}) := 0$, and $B_j(\bar{\eta}) := 0$. To determine if $\eta_k \equiv \alpha_k$ or $\eta_k \equiv \beta_k$, let k_1 be the index of the next lower breakpoint and k_2 the index of the next upper breakpoint. This implies either $\eta_k = \alpha_{k_1}$ or $\eta_k = \beta_{k_2}$. Observe that the partition corresponding to the breakpoint η_k does not depend on whether η_k is a lower or upper breakpoint. Thus, it follows from the breakpoint expressions in Equations (5.13) and (5.14) that $\eta_k = \alpha_{k_1}$ if $\frac{l_{k_1}}{\alpha_{k_1}} + b_{k_1} < \frac{u_{k_2}}{\alpha_{k_2}} + b_{k_2}$ and $\eta_k = \beta_{k_2}$ otherwise.

Type of η_i	$\mathcal{N}_j^{\text{lower}}(\eta_k)$	$\mathcal{N}_j^{\text{upper}}(\eta_k)$	$\mathcal{N}_j^{\text{free}}(\eta_k)$	$y_j^{\text{fixed}}(\eta_k)$	$A_j(\eta_k)$	$B_j(\eta_k)$
$\eta_i \equiv \alpha_i$	$\mathcal{N}_j^{\text{lower}}(\eta_i) \setminus \{i\}$	$\mathcal{N}_j^{\text{upper}}(\eta_i)$	$\mathcal{N}_j^{\text{free}}(\eta_i) \cup \{i\}$	$y_j^{\text{fixed}}(\eta_i) - l_i$	$A_j(\eta_i) + a_i$	$B_j(\eta_i) + a_i b_i$
$\eta_i \equiv \beta_i$	$\mathcal{N}_j^{\text{lower}}(\eta_i)$	$\mathcal{N}_j^{\text{upper}}(\eta_i) \cup \{i\}$	$\mathcal{N}_j^{\text{free}}(\eta_i) \setminus \{i\}$	$y_j^{\text{fixed}}(\eta_i) + u_i$	$A_j(\eta_i) - a_i$	$B_j(\eta_i) - a_i b_i$

Table 5.2: Relation between consecutive breakpoints η_i and η_k and their index set partitions.

Algorithm 5.1 summarizes this approach to compute the breakpoints. Each new smallest breakpoint η_k in Line 6 can be retrieved in $O(1)$ time if we implement the values in $\mathcal{P}_j$ and $\mathcal{Q}_j$ as sorted lists. As a consequence, the time complexity of Algorithm 5.1 for a given $j \in \mathcal{M}$ is $O(n_j \log n_j)$ and the overall computation of the breakpoints for all variables can be done in $O(n \log n)$ time. Note that if each n_j is equal to a given constant K , i.e., all subsets $\mathcal{N}_j$ have the same constant cardinality, this complexity can be refined to $O\left(\sum_{j \in \mathcal{M}} K \log K\right) = O(Km \log K) = O(n \log K)$. Thus, for a given K the breakpoints can be computed in linear time.

Computing λ^*

To finalize our approach, we need to specify how to compute λ^* for a given δ_1 , which is the largest breakpoint such that $\delta_1 \leq \lambda^*$. In Section 5.4.1, we showed that the partitioning of the variables under λ^* can be derived from the partitioning under δ_1 and δ_2 , i.e., for each $j \in \mathcal{M}$, we have $\mathcal{N}_j^{\text{lower}}(\lambda^*) = \mathcal{N}_j^{\text{lower}}(\delta_1)$, $\mathcal{N}_j^{\text{upper}}(\lambda^*) = \mathcal{N}_j^{\text{upper}}(\delta_2)$, and $\mathcal{N}_j^{\text{free}}(\lambda^*) = \mathcal{N}_j^{\text{free}}(\delta_1)$. Moreover, as we have $\sum_{j \in \mathcal{M}} y_j(\lambda^*) = R$ by definition of y_j , we can apply the derived expression for

Algorithm 5.1: Computing the breakpoints for $j \in \mathcal{M}$.

```

1 Input: Parameters  $a_i \in \mathbb{R}_{>0}$ ,  $l_i, u_i, b_i \in \mathbb{R}$  for  $i \in \mathcal{N}_j$ , and  $w_j \in \mathbb{R}$ 
2 Output: Lower and upper breakpoints  $\alpha_i$  and  $\beta_i$  for  $i \in \mathcal{N}_j$ 
3 Compute the sets  $\mathcal{P}_j := \{\frac{l_i}{a_i} + b_i \mid i \in \mathcal{N}_j\}$  and  $\mathcal{Q}_j := \{\frac{u_i}{a_i} + b_i \mid i \in \mathcal{N}_j\}$ 
4 Initialize  $\bar{Y} := \sum_{i \in \mathcal{N}_j} l_i$ ;  $\bar{A} := 0$ ;  $\bar{B} := 0$ 
5 repeat
6   Determine smallest value  $\eta_k := \min(\mathcal{P}_j \cup \mathcal{Q}_j)$ 
7   if  $\eta_k \in \mathcal{Q}_j$  then  $\triangleright \eta_k \equiv \alpha_k$ 
8      $y_j^{\text{fixed}}(\alpha_k) = \bar{Y}$ ;  $A_j(\alpha_k) = \bar{A}$ ;  $B_j(\alpha_k) = \bar{B}$ 
9      $\alpha_k := w_j(y_j^{\text{fixed}}(\alpha_k) - B_j(\alpha_k)) + \left(\frac{l_k}{a_k} + b_k\right)(1 + w_j A_j(\alpha_k))$ 
10     $\bar{Y} = \bar{Y} - l_k$ ;  $\bar{A} = \bar{A} + a_k$ ;  $\bar{B} = \bar{B} + a_k b_k$ 
11     $\mathcal{Q}_j = \mathcal{Q}_j \setminus \{\eta_k\}$ 
12  else  $\triangleright \eta_k \equiv \beta_k$ 
13     $y_j^{\text{fixed}}(\beta_k) = \bar{Y}$ ;  $A_j(\beta_k) = \bar{A}$ ;  $B_j(\beta_k) = \bar{B}$ 
14     $\beta_k := w_j(y_j^{\text{fixed}}(\beta_k) - B_j(\beta_k)) + \left(\frac{u_k}{a_k} + b_k\right)(1 + w_j A_j(\beta_k))$ 
15     $\bar{Y} = \bar{Y} + u_k$ ;  $\bar{A} = \bar{A} - a_k$ ;  $\bar{B} = \bar{B} - a_k b_k$ 
16     $\mathcal{P}_j = \mathcal{P}_j \setminus \{\eta_k\}$ 
17 until  $\mathcal{P}_j \cup \mathcal{Q}_j = \emptyset$ 
18 return  $\alpha_i, \beta_i$  for  $i \in \mathcal{N}_j$ 

```

general $y_j(\lambda)$ in Equation (5.11) to obtain the following linear equation in λ^* :

$$\begin{aligned}
 R &= \sum_{j \in \mathcal{M}} y_j(\lambda^*) = \sum_{j \in \mathcal{M}} \left(\frac{y_j^{\text{fixed}}(\lambda^*) - B_j(\lambda^*)}{1 + w_j A_j(\lambda^*)} + \frac{A_j(\lambda^*)}{1 + w_j A_j(\lambda^*)} \lambda^* \right) \\
 &= \sum_{j \in \mathcal{M}} \left(\frac{y_j^{\text{fixed}}(\delta_1) - B_j(\delta_1)}{1 + w_j A_j(\delta_1)} + \frac{A_j(\delta_1)}{1 + w_j A_j(\delta_1)} \lambda^* \right).
 \end{aligned}$$

It follows that

$$\lambda^* = \frac{R - \left(\sum_{j \in \mathcal{M}} \frac{y_j^{\text{fixed}}(\delta_1) - B_j(\delta_1)}{1 + w_j A_j(\delta_1)} \right)}{\sum_{j \in \mathcal{M}} \frac{A_j(\delta_1)}{1 + w_j A_j(\delta_1)}}. \quad (5.15)$$

Note that, given the partitioning sets $\mathcal{N}_j^{\text{lower}}(\lambda^*)$, $\mathcal{N}_j^{\text{upper}}(\lambda^*)$, and $\mathcal{N}_j^{\text{free}}(\lambda^*)$, this expression allows us to compute λ^* in $O(\sum_{j \in \mathcal{M}} n_j) = O(n)$ time.

5.5 TWO ALGORITHMS FOR PROBLEM QRA-NONSEP-GBC

In this section, we present two algorithms that solve Problem QRA-NonSep-GBC based on the approach presented in Section 5.4. This approach can be summarized by means of the following four steps:

1. Replace the generalized bound constraints (5.1b) by the box constraints $\underline{x}_i \leq x_i \leq \bar{x}_i$, $i \in \mathcal{N}$ (Section 5.3.2),
2. Compute for each $i \in \mathcal{N}$ the lower and upper breakpoints α_i and β_i (Section 5.4.2),
3. Find δ_1 , i.e., the largest breakpoint smaller than or equal to λ^* (Section 5.4.1),
4. Compute the optimal Lagrange multiplier λ^* and the optimal solution $x(\lambda^*)$ (Section 5.4.2).

Both algorithms follow these four steps. Their difference is in the execution of Step 3 or, more precisely, in how we search for δ_1 through the breakpoint set $\mathcal{D}$. In the first algorithm, we consider the breakpoints sequentially starting from the smallest breakpoint, whereas in the second algorithm, we apply binary search on $\mathcal{D}$. We present and discuss these algorithms and their breakpoint search strategies in more detail in Sections 5.5.1 and 5.5.2.

5.5.1 AN $O(n \log n)$ TIME ALGORITHM BASED ON SEQUENTIAL BREAKPOINT SEARCH

The sequential breakpoint search strategy is similar to Algorithm 3.1 for solving Problem QRA and Algorithm 5.1 to compute the breakpoints, i.e., we search through the breakpoint set $\mathcal{D}$ in ascending order. For each considered breakpoint η_k , $k \in \mathcal{N}_j$, we compute the sum $\sum_{j \in \mathcal{M}} y_j(\eta_k)$ using Equation (5.11). If $\sum_{j \in \mathcal{M}} y_j(\eta_k) < R$, it follows from Equation (5.8b) that $\eta_k < \lambda^*$ and we continue the search. Otherwise, if $\sum_{j \in \mathcal{M}} y_j(\eta_k) > R$, then it follows from Equation (5.8a) that $\eta_k > \lambda^*$, meaning that δ_1 is the breakpoint preceding η_k and that $\delta_2 = \eta_k$. Subsequently, we can use Equation (5.15) to compute λ^* . Finally, if $\sum_{j \in \mathcal{M}} y_j(\eta_k) = R$, then $\lambda^* = \eta_k$ by definition of the values $y_j(\cdot)$.

To efficiently compute the sum $\sum_{j \in \mathcal{M}} y_j(\eta_k)$, we exploit the dependencies between η_k and its preceding breakpoint η_i , $i \in \mathcal{N}_j$, given in Table 5.2. This means that we can compute the terms $y_j^{\text{fixed}}(\eta_k)$, $A_j(\eta_k)$, and $B_j(\eta_k)$ in $O(1)$ time from the terms $y_j^{\text{fixed}}(\eta_i)$, $A_j(\eta_i)$, and $B_j(\eta_i)$. Moreover, by defining for a given $\lambda \in \mathbb{R}$

$$F(\lambda) := \sum_{j \in \mathcal{M}} \frac{y_j^{\text{fixed}}(\lambda) - B_j(\lambda)}{1 + w_j A_j(\lambda)}, \quad V(\lambda) := \frac{A_j(\lambda)}{1 + w_j A_j(\lambda)}.$$

and by using these values and the dependencies in Table 5.2, we can easily compute $\sum_{j \in \mathcal{M}} y_j(\eta_k)$ from $\sum_{j \in \mathcal{M}} y_j(\eta_i)$ in $O(1)$ time. For this, note that $\sum_{j \in \mathcal{M}} y_j(\lambda) = F(\lambda) + \lambda V(\lambda)$ by Equation (5.11).

Algorithm 5.2 summarizes the four steps of our overall solution approach where Step 3 is carried out using the sequential breakpoint search strategy. In this algorithm, Line 4 corresponds to Step 1, Line 5 to Step 2, Lines 6-26 to Step 3, and Lines 14 and 17 to Step 4. During each iteration τ of the sequential breakpoint

Algorithm 5.2: Solving Problem QRA-NonSep-GBC using sequential breakpoint search.

```

1 Input:  $w, L, U \in \mathbb{R}^m$ ,  $a \in \mathbb{R}_{>0}^n$ ,  $b, l, u \in \mathbb{R}^n$ , and  $R \in \mathbb{R}$ 
2 Output: Optimal solution  $\bar{x}$  to Problem QRA-NonSep-GBC
3 for  $j \in \mathcal{M}$  do
4   Solve QRAj( $L_j$ ) and QRAj( $U_j$ ); set  $l_i := \max(l_i, \underline{x}_i)$  and
    $u_i := \min(u_i, \bar{x}_i)$ 
5   Compute  $\alpha_i$  and  $\beta_i$  for each  $i \in \mathcal{N}_j$  using Algorithm 5.1
6  $\mathcal{D} := \{\alpha_i \mid i \in \mathcal{N}\} \cup \{\beta_i \mid i \in \mathcal{N}\}$ ;  $\tau := 0$ ;  $\mathcal{D}^\tau := \mathcal{D}$ ;  $F := \sum_{i \in \mathcal{N}} l_i$ ;
    $V := 0$ 
7 For  $j \in \mathcal{M}$ : Initialize  $\bar{Y}_j := \sum_{i \in \mathcal{N}_j} l_i$ ;  $\bar{A}_j := 0$ ;  $\bar{B}_j := 0$ 
8 repeat
9   Determine  $\eta_k := \min(\mathcal{D}^\tau)$  and define  $j$  as the index for which  $k \in \mathcal{N}_j$ 
10  for  $j' \in \mathcal{M}$  do
11     $y_{j'}^{\text{fixed}}(\eta_k) = \bar{Y}_{j'}$ ;  $A_{j'}(\eta_k) = \bar{A}_{j'}$ ;  $B_{j'}(\eta_k) = \bar{B}_{j'}$ 
12  Compute  $\sum_{j'=1}^m y_{j'}(\eta_k) = F + V\alpha_k$ 
13  if  $\sum_{j'=1}^m y_{j'}(\eta_k) = R$  then
14     $\lambda^* := \eta_k$ ; compute  $x(\lambda^*)$  as  $x(\eta_k)$  using Equation (5.10)
15    return  $\bar{x} := x(\lambda^*)$ 
16  else if  $\sum_{j'=1}^m y_{j'}(\eta_k) > R$  then
17     $\lambda^* := \frac{R-F}{V}$ ; compute  $x(\lambda^*)$  using Equation (5.10)
18    return  $\bar{x} := x(\lambda^*)$ 
19  else
20     $F := F - \frac{\bar{Y}_j - \bar{B}_j}{1+w_j \bar{A}_j}$ ;  $V := V - \frac{\bar{A}_j}{1+w_j \bar{A}_j}$ 
21    if  $\eta_k \equiv \alpha_k$  then
22       $\bar{Y}_j := \bar{Y}_j - l_i$ ;  $\bar{A}_j := \bar{A}_j + a_i$ ;  $\bar{B}_j := \bar{B}_j + a_i b_i$ 
23    else
24       $\bar{Y}_j := \bar{Y}_j + u_i$ ;  $\bar{A}_j := \bar{A}_j - a_i$ ;  $\bar{B}_j := \bar{B}_j - a_i b_i$ 
25     $F := F + \frac{\bar{Y}_j - \bar{B}_j}{1+w_j \bar{A}_j}$ ;  $V := V + \frac{\bar{A}_j}{1+w_j \bar{A}_j}$ 
26     $\mathcal{D}^{\tau+1} := \mathcal{D}^\tau \setminus \{\eta_k\}$ ;  $\tau := \tau + 1$ 

```

search in Lines 6-26, the set $\mathcal{D}^\tau$ is the subset of the original breakpoint set $\mathcal{D}$ that has not yet been searched in this iteration.

We state the time complexity of this algorithm in the following theorem:

Theorem 5.1. *Algorithm 5.2 has a worst-case time complexity of $O(n \log n)$.*

Proof. First, the replacement of Constraint (5.1b) by the single-variable constraints $\underline{x}_i \leq x_i \leq \bar{x}_i$ in Line 4 takes in total $O(n)$ time. Second, the compu-

tation of the breakpoints by means of Algorithm 5.1 in Line 5 takes in total $O(n \log n)$ time. Third, each iteration of the sequential breakpoint procedure can be executed in $O(1)$ time if we maintain the breakpoint sets as sorted lists so that computing the smallest value η_k in Line 9 can be done in $O(1)$ time. Finally, once λ^* has been found in either Line 14 or 17, we can compute the optimal solution $x(\lambda^*)$ in $O(n)$ time using Equation (5.10). Summarizing, the worst-case time complexity of Algorithm 5.2 is $O(n \log n)$. $\square$

Note that besides the computation and sorting of the breakpoints, Algorithm 5.2 runs in linear time.

5.5.2 AN $O(n \log n)$ TIME ALGORITHM BASED ON BINARY BREAKPOINT SEARCH

In this subsection we present an alternative strategy, where we apply binary search on the set of breakpoints. Note that this strategy is similar to Algorithm 3.2 for solving Problem QRA. During each iteration τ of the binary search, we compute the median $\hat{\delta}^\tau$ of the current breakpoint set $\mathcal{D}^\tau$, i.e., of the subset of the original breakpoint set that is guaranteed to contain the breakpoint δ_1 . For this median breakpoint, we compute the sum $\sum_{j \in \mathcal{M}} y_j(\hat{\delta}^\tau)$ and compare this value to the given amount R of the resource. If $\sum_{j \in \mathcal{M}} y_j(\hat{\delta}^\tau) = R$, then $\lambda^* = \hat{\delta}^\tau$. Otherwise, if $\sum_{j \in \mathcal{M}} y_j(\hat{\delta}^\tau) > R$, then $\hat{\delta}^\tau < \lambda^*$ and during the next iteration $\tau + 1$ we take as breakpoint set $\mathcal{D}^{\tau+1} := \{\lambda \in \mathcal{D}^\tau \mid \lambda > \hat{\delta}^\tau\}$. Finally, if $\sum_{j \in \mathcal{M}} y_j(\hat{\delta}^\tau) < R$, we have that $\hat{\delta}^\tau > \lambda^*$ and during the next iteration $\tau + 1$ we take as breakpoint set $\mathcal{D}^{\tau+1} := \{\lambda \in \mathcal{D}^\tau \mid \lambda \leq \hat{\delta}^\tau\}$.

To efficiently compute each sum $\sum_{j \in \mathcal{M}} y_j(\hat{\delta}^\tau)$, we use the following observation that is analogous to the approach in Section 3.3.2 for solving Problem QRA in $O(n)$ time. For a given iteration τ of the binary search, let $\lambda_\downarrow^\tau$ and $\lambda_\uparrow^\tau$ denote the minimum and maximum breakpoint in the current breakpoint set $\mathcal{D}^\tau$. Then for any multiplier λ that lies within the interval $[\lambda_\downarrow^\tau, \lambda_\uparrow^\tau]$ and each $j \in \mathcal{M}$ and $i \in \mathcal{N}_j$, the following is true due to Equation (5.9):

$$\lambda_\uparrow^\tau \leq \alpha_i \Rightarrow i \in \mathcal{N}_j^{\text{lower}}(\lambda), \quad (5.16a)$$

$$\alpha_i \leq \lambda_\downarrow^\tau \leq \lambda_\uparrow^\tau \leq \beta_i \Rightarrow i \in \mathcal{N}_j^{\text{free}}(\lambda), \quad (5.16b)$$

$$\beta_i \leq \lambda_\downarrow^\tau \Rightarrow i \in \mathcal{N}_j^{\text{upper}}(\lambda). \quad (5.16c)$$

We introduce the following sets, which partition the set $\mathcal{N}_j$ according to Equation (5.16) during iteration τ :

$$\mathcal{L}_j^\tau := \{i \in \mathcal{N}_j \mid \lambda_\uparrow^\tau \leq \alpha_i\}, \quad (5.17a)$$

$$\mathcal{F}_j^\tau := \{i \in \mathcal{N}_j \mid \alpha_i \leq \lambda_\downarrow^\tau \leq \lambda_\uparrow^\tau \leq \beta_i\}, \quad (5.17b)$$

$$\mathcal{U}_j^\tau := \{i \in \mathcal{N}_j \mid \beta_i \leq \lambda_\downarrow^\tau\}, \quad (5.17c)$$

$$\mathcal{I}_j^\tau := \mathcal{N}_j \setminus (\mathcal{L}_j^\tau \cup \mathcal{F}_j^\tau \cup \mathcal{U}_j^\tau) = \{i \in \mathcal{N}_j \mid \lambda_\downarrow^\tau < \alpha_i < \lambda_\uparrow^\tau \text{ or } \lambda_\downarrow^\tau < \beta_i < \lambda_\uparrow^\tau\} \quad (5.17d)$$

(see also Figure 5.3). Note, that for any λ with $\lambda_{\downarrow}^{\tau} \leq \lambda \leq \lambda_{\uparrow}^{\tau}$, we have

$$\begin{aligned} i \in \mathcal{L}_j^{\tau} &\Rightarrow i \in \mathcal{N}_j^{\text{lower}}(\lambda), \\ i \in \mathcal{F}_j^{\tau} &\Rightarrow i \in \mathcal{N}_j^{\text{free}}(\lambda), \\ i \in \mathcal{U}_j^{\tau} &\Rightarrow i \in \mathcal{N}_j^{\text{upper}}(\lambda). \end{aligned}$$

Figure 5.3: Example of the partitioning of the variables based on their breakpoints and the interval $[\lambda_{\downarrow}^{\tau}, \lambda_{\uparrow}^{\tau}]$.

Due to the construction of the sets $\mathcal{D}^{\tau}$, the sequence $(\lambda_{\downarrow}^{\tau})_{\tau \in \mathbb{N}}$ is nondecreasing and the sequence $(\lambda_{\uparrow}^{\tau})_{\tau \in \mathbb{N}}$ is non-increasing. This implies that as soon as one of the three cases (5.16a)-(5.16b) occurs during an iteration τ for an index $i \in \mathcal{N}_j$, we know for any future candidate breakpoint $\hat{\delta}^{\tau}$ that $i \in \mathcal{N}_j^{\text{lower}}(\hat{\delta}^{\tau})$, $i \in \mathcal{N}_j^{\text{free}}(\hat{\delta}^{\tau})$, or $i \in \mathcal{N}_j^{\text{upper}}(\hat{\delta}^{\tau})$ respectively. In particular, we know that $i \in \mathcal{N}_j^{\text{lower}}(\lambda^*)$, $i \in \mathcal{N}_j^{\text{free}}(\lambda^*)$, or $i \in \mathcal{N}_j^{\text{upper}}(\lambda^*)$ respectively. Thus, when determining the partition $(\mathcal{N}_j^{\text{lower}}(\hat{\delta}^{\tau}), \mathcal{N}_j^{\text{free}}(\hat{\delta}^{\tau}), \mathcal{N}_j^{\text{upper}}(\hat{\delta}^{\tau}))$, we only need to determine the membership of $x_k(\hat{\delta}^{\tau})$ for all $k \in \mathcal{I}_j^{\tau}$ instead of for all $k \in \mathcal{N}_j$ when the sets $\mathcal{L}_j^{\tau}$, $\mathcal{F}_j^{\tau}$, and $\mathcal{U}_j^{\tau}$ are known.

The main computational gain is obtained by introducing for each iteration τ the following bookkeeping parameters:

$$Y_j^{\tau} := \sum_{i \in \mathcal{L}_j^{\tau}} l_i + \sum_{i \in \mathcal{U}_j^{\tau}} u_i, \quad \bar{A}_j^{\tau} := \sum_{i \in \mathcal{F}_j^{\tau}} a_i, \quad \bar{B}_j^{\tau} := \sum_{i \in \mathcal{F}_j^{\tau}} a_i b_i.$$

Observe that if the set $\mathcal{I}_j^{\tau}$ and the bookkeeping parameters Y_j^{τ} , $\bar{A}_j^{\tau}$, and $\bar{B}_j^{\tau}$ are known, then computing $y_j(\lambda)$ for any $\lambda_{\downarrow}^{\tau} \leq \lambda \leq \lambda_{\uparrow}^{\tau}$ via Equation (5.11) can be done in $O(|\mathcal{I}_j^{\tau}|)$ time instead of $O(n_j)$ time.

We summarize the resulting realization of the four steps of our overall solution approach, using in Step 3 the discussed binary breakpoint search strategy, in

Algorithm 5.3. In this algorithm, Line 4 corresponds to Step 1, Line 5 to Step 2, Lines 6-36 to Step 3, and Lines 21 and 37-38 to Step 4. In each iteration τ , the new set $\mathcal{I}_j^{\tau+1}$ and bookkeeping values $Y_j^{\tau+1}$, $\bar{A}_j^{\tau+1}$, and $\bar{B}_j^{\tau+1}$ are constructed after the new breakpoint set $\mathcal{D}^{\tau+1}$ and lower and upper bounds $\lambda_{\downarrow}^{\tau+1}$ and $\lambda_{\uparrow}^{\tau+1}$ have been determined. This update can be done using the definition of the sets $\mathcal{L}_j^{\tau+1}$, $\mathcal{F}_j^{\tau+1}$, $\mathcal{U}_j^{\tau+1}$, and $\mathcal{I}_j^{\tau+1}$ in Equation (5.17).

We establish the worst-case time complexity of Algorithm 5.3 by means of Lemma 5.7 and Theorem 5.2. First, Lemma 5.7 states that the binary search procedure can be carried out in $O(n)$ time.

Lemma 5.7. *The binary breakpoint search procedure in Lines 6-36 of Algorithm 5.3 has a time complexity of $O(n)$.*

Proof. We show that iteration τ of the binary breakpoint search has a time complexity of $O(|\mathcal{D}^\tau|)$. Since $|\mathcal{D}^{\tau+1}| \leq \frac{1}{2}|\mathcal{D}^\tau|$ for each iteration τ , it follows that the time complexity of the binary search procedure is

$$O\left(\sum_{\tau=0}^{\log(n)} |\mathcal{D}^\tau|\right) = O\left(\sum_{\tau=0}^{\log(n)} \frac{n}{2^\tau}\right) = O(n).$$

We establish the time complexity of an iteration τ using the following two observations:

1. We consider the computation of the candidate multiplier $\hat{\delta}^\tau$ in Line 9. Recall that the median of an unsorted set of breakpoints $\mathcal{D}^\tau$ can be computed in $O(|\mathcal{D}^\tau|)$ time using, e.g., the median-of-medians algorithm in [28]. This means that instead of sorting the initial breakpoint set $\mathcal{D}$ in $O(n \log n)$ time and retrieving median elements in $O(1)$ time, we can compute each candidate multiplier $\hat{\delta}^\tau$ in $O(|\mathcal{D}^\tau|)$ time.
2. By introducing the partition sets $\mathcal{L}_j^{\tau+1}$, $\mathcal{F}_j^{\tau+1}$, $\mathcal{U}_j^{\tau+1}$, and $\mathcal{I}_j^{\tau+1}$ and the bookkeeping values Y_j^τ , $\bar{A}_j^\tau$, and $\bar{B}_j^\tau$, we reduce the worst-case time complexity of computing $\sum_{j \in \mathcal{M}} y_j(\hat{\delta}^\tau)$ from $O(n)$ to $O(\sum_{j \in \mathcal{M}} |\mathcal{I}_j^\tau|)$. On

Algorithm 5.3: Solving Problem QRA-NonSep-GBC using binary breakpoint search.

- 1 **Input:** $w, L, U \in \mathbb{R}^m$, $a \in \mathbb{R}_{>0}^n$, $b, l, u \in \mathbb{R}^n$, and $R \in \mathbb{R}$
 - 2 **Output:** Optimal solution $\bar{x}$ to Problem QRA-NonSep-GBC
 - 3 **for** $j \in \mathcal{M}$ **do**
 - 4 Solve $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$; set $l_i := \max(l_i, x_i)$ and $u_i := \min(u_i, \bar{x}_i)$
 - 5 Compute α_i and β_i for each $i \in \mathcal{N}_j$ using Algorithm 5.1
-

```

6  $\mathcal{D} := \{\alpha_i \mid i \in \mathcal{N}\} \cup \{\beta_i \mid i \in \mathcal{N}\}; \mathcal{D}^0 := \mathcal{D}; \tau := 0; \lambda_{\downarrow}^0 := -\infty; \lambda_{\uparrow}^0 := \infty$ 
7 For  $j \in \mathcal{M}$ :  $\mathcal{I}_j^0 := \mathcal{N}_j; Y_j^0 := \bar{A}_j^0 := \bar{B}_j^0 := 0$ 
8 while  $|\mathcal{D}^\tau| > 1$  do
9    $\hat{\delta}^\tau := \text{median}(\mathcal{D}^\tau)$ 
10  for  $j \in \mathcal{M}$  do
11     $y_j^{\text{fixed}}(\hat{\delta}^\tau) := Y_j^\tau; A_j(\hat{\delta}) := \bar{A}_j^\tau; B_j(\hat{\delta}) := \bar{B}_j^\tau$ 
12    for  $k \in \mathcal{I}_j^\tau$  do
13      if  $k \in \mathcal{N}_j^{\text{lower}}(\hat{\delta}^\tau)$  then
14         $y_j^{\text{fixed}}(\hat{\delta}^\tau) := y_j^{\text{fixed}}(\hat{\delta}^\tau) + l_k$ 
15      else if  $k \in \mathcal{N}_j^{\text{upper}}(\hat{\delta}^\tau)$  then
16         $y_j^{\text{fixed}}(\hat{\delta}^\tau) := y_j^{\text{fixed}}(\hat{\delta}^\tau) + u_k$ 
17      else
18         $A_j(\hat{\delta}^\tau) := A_j(\hat{\delta}^\tau) + a_k; B_j(\hat{\delta}^\tau) := B_j(\hat{\delta}^\tau) + a_k b_k$ 
19  Compute  $\sum_{j \in \mathcal{M}} y_j(\hat{\delta}^\tau)$  using Equation (5.11)
20  if  $\sum_{j \in \mathcal{M}} y_j(\hat{\delta}^\tau) = R$  then  $\triangleright \lambda^* = \hat{\delta}^\tau$ 
21    Compute  $x(\lambda)$  as  $x(\hat{\delta}^\tau)$  using Equation (5.10)
22    return  $\bar{x} := x(\lambda^*)$ 
23  else if  $\sum_{i \in \mathcal{N}} x_i(\hat{\delta}^\tau) < R$  then  $\triangleright \hat{\delta}^\tau \leq \lambda^* \leq \delta_1$ 
24     $\mathcal{D}^{\tau+1} := \{\lambda \in \mathcal{D}^\tau \mid \lambda > \hat{\delta}^\tau\}; \lambda_{\downarrow}^{\tau+1} := \lambda_{\downarrow}^\tau; \lambda_{\uparrow}^{\tau+1} := \hat{\delta}^\tau$ 
25  else  $\triangleright \hat{\delta}^\tau > \lambda^* > \delta_2$ 
26     $\mathcal{D}^{\tau+1} := \{\lambda \in \mathcal{D}^\tau \mid \lambda \leq \hat{\delta}^\tau\}; \lambda_{\downarrow}^{\tau+1} := \hat{\delta}^\tau; \lambda_{\uparrow}^{\tau+1} := \lambda_{\uparrow}^\tau$ 
27  for  $j \in \mathcal{M}$  do
28     $\mathcal{I}_j^{\tau+1} := \mathcal{I}_j^\tau; Y_j^{\tau+1} := Y_j^\tau; \bar{A}_j^{\tau+1} := \bar{A}_j^\tau; \bar{B}_j^{\tau+1} := \bar{B}_j^\tau$ 
29    for  $k \in \mathcal{I}_j^\tau$  do
30      if  $\lambda_{\uparrow}^{\tau+1} \leq \alpha_k$  then
31        Remove  $k$  from  $\mathcal{I}_j^{\tau+1}; Y_j^{\tau+1} := Y_j^{\tau+1} + l_k$ 
32      else if  $\alpha_i \leq \lambda_{\downarrow}^{\tau+1} \leq \lambda_{\uparrow}^{\tau+1} \leq \beta_i$  then
33        Remove  $k$  from  $\mathcal{I}_j^{\tau+1}; \bar{A}_j^{\tau+1} := \bar{A}_j^{\tau+1} + a_k;$ 
34         $\bar{B}_j^{\tau+1} := \bar{B}_j^{\tau+1} + a_k b_k$ 
35      else if  $\beta_k \leq \lambda_{\downarrow}^{\tau+1}$  then
36        Remove  $k$  from  $\mathcal{I}_j^{\tau+1}; Y_j^{\tau+1} := Y_j^{\tau+1} + u_k$ 
37   $\tau := \tau + 1$ 
38  Determine  $\delta_1$  as the single element of  $\mathcal{D}^\tau$ 
39  Compute  $\lambda^*$  using Equation (5.15) and  $x(\lambda^*)$  using Equation (5.10)
40  return  $\bar{x} := x(\lambda^*)$ 

```

the other hand, constructing the new set $\mathcal{I}_j^{\tau+1}$ and the bookkeeping values $Y_j^{\tau+1}$, $\bar{A}_j^{\tau+1}$, and $\bar{B}_j^{\tau+1}$ in Lines 23-35 takes $O(\sum_{j \in \mathcal{M}} |\mathcal{I}_j^\tau|)$ time.

Thus, the time complexity of the τ^{th} iteration of the binary search loop is $O(|\mathcal{D}^\tau| + \sum_{j \in \mathcal{M}} |\mathcal{I}_j^\tau|)$. Observe that by definition of $\mathcal{I}_j^\tau$, for each $j \in \mathcal{M}$ and each index $k \in \mathcal{I}_j^\tau$ there is at least one breakpoint (α_k or β_k or both) in the set of current breakpoints $\mathcal{D}^\tau$. This implies that $\sum_{j \in \mathcal{M}} |\mathcal{I}_j^\tau| \leq |\mathcal{D}^\tau|$. It follows that the time complexity of the τ^{th} iteration of the binary search loop reduces to $O(|\mathcal{D}^\tau|)$. $\square$

Using this lemma, we establish the time complexity of Algorithm 5.3:

Theorem 5.2. *Algorithm 5.3 has a time complexity of $O(n \log n)$.*

Proof. Analogously to Theorem 5.1, all operations other than the binary search procedure in Lines 6-36 take $O(n \log n)$ time. Since the binary search procedure takes $O(n)$ time by Lemma 5.7, the overall time complexity of Algorithm 5.3 is $O(n \log n)$. $\square$

5.5.3 COMPLEXITY RESULTS FOR SPECIAL CASES AND RELATED PROBLEMS

In this section, we use Algorithms 5.2 and 5.3 and the complexity results in Theorems 5.1 and 5.2 to state complexity results for several special cases of Problem QRA-NonSep-GBC and related problems. Some of these cases are of interest for the problem of three-phase electric vehicle charging, whereas other cases may be of independent interest.

The first special case is when all subsets $\mathcal{N}_j$ have the same cardinality, i.e., $n_j = K$ for some $K \in \mathbb{N}$. For this case, we can show that, given K , the time complexity of Algorithm 5.3 is linear. Note that this special case includes the problem of three-phase electric vehicle charging introduced in Section 5.2.3 (see also Table 5.1) as we have $K = 3$ in this case.

Theorem 5.3. *If $n_j = K$ for all $j \in \mathcal{M}$ for a given $K \in \mathbb{N}$, the time complexity of Algorithm 5.3 is $O(n \log K)$.*

Proof. The only part of the algorithm that does not have a linear time complexity is the computation of the breakpoints, which needs $O(n \log n)$ operations for the general Problem QRA-NonSep-GBC. However, when $n_j = K$, the complexity analysis can be refined to $O(\sum_{j \in \mathcal{M}} n_j \log n_j) = O(\sum_{j \in \mathcal{M}} K \log K) = O(mK \log K) = O(n \log K)$. It follows that the time complexity of Algorithm 5.3 for this special case is $O(n \log K)$. $\square$

Next, we focus on the special case where $w_j = 0$ for all $j \in \mathcal{M}$, i.e., we consider the quadratic *separable* RAP with generalized bound constraints ((a,b)-Q/GBC/C). With regard to three-phase EV charging, this case represents the

situation where the only objective is to minimize peak consumption and we do not consider minimization of load unbalance. As mentioned in Section 2.3.2, this case can be solved in $O(n)$ time [114]. In particular, it can be solved in $O(n)$ time by solving a sequence of instances of Problem QRA:

Theorem 5.4. *If $w_j = 0$ for all $j \in \mathcal{M}$, Problem QRA-NonSep-GBC can be solved in $O(n)$ time by solving $2m + 1$ specific instances of Problem QRA.*

Proof. First, recall that the first step of our solution approach, namely the replacement of the generalized bound constraints (5.1b) by the box constraints $\underline{x}_i \leq x_i \leq \bar{x}_i$, $i \in \mathcal{N}$ (see Section 5.3.2), can be done in $O(n)$ time by solving the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ for each $j \in \mathcal{M}$. The remaining problem is an instance of Problem QRA since $w_j = 0$ for each $j \in \mathcal{M}$. Thus, we can solve this problem in $O(n)$ time, which implies that we can also solve the complete Problem QRA-NonSep-GBC in $O(n)$ time by solving $2m + 1$ instances of Problem QRA. $\square$

Finally, we consider the integer version of Problem QRA-NonSep-GBC, i.e., the problem with the additional constraint that $x_i \in \mathbb{Z}$ for all $i \in \mathcal{N}$. If $w_j = 0$ for all $j \in \mathcal{M}$, we can solve the integer version in $O(n)$ time:

Theorem 5.5. *If $w_j = 0$ for all $j \in \mathcal{M}$, the integer version of Problem QRA-NonSep-GBC can be solved in $O(n)$ time.*

Proof. Without loss of generality, we assume that $l, u \in \mathbb{Z}^n$, $L, U \in \mathbb{Z}^m$, and $R \in \mathbb{Z}$. Note that all steps and statements in the proof of Lemma 5.2 are valid for the integer version of Problem QRA-NonSep-GBC, provided that the chosen optimal solution to $\text{QRA}^j(U_j)$ is component-wise larger than that to $\text{QRA}^j(L_j)$, i.e., we have $\underline{x} \leq \bar{x}$. In more detail, the proof of Lemma 5.2 is valid since we have $\bar{\epsilon} > 1$ and we can choose $\epsilon = 1$ to obtain feasible solutions x' and $(\underline{x}')^j$. Thus, analogously to the case of continuous variables in Theorem 5.4, we can solve this version of the problem by solving the $2m$ subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ and one instance of Problem QRA with n variables (see also the proof of Theorem 5.4) as integer RAPs. Since an instance of Problem QRA with integer variables can be solved in linear time ([85], see also Section 2.3.1), we can solve these $2m + 1$ problems in $O(n)$ time. $\square$

If $w_j \geq 0$ for all $j \in \mathcal{M}$, the integer version can be solved in $O(n^2)$ time [120]. Finally, we conjecture that the integer version of the general Problem QRA-NonSep-GBC, i.e., instances that satisfy Property 5.1, is solvable in strongly polynomial time, but leave this as an open question for future research.

5.6 EVALUATION

In this section, we evaluate the two algorithms that were presented in Sections 5.5.1 and 5.5.2. We carry out two types of evaluations. First, we evaluate the performance of our algorithms on realistic instances of the EV charging problem EV-3Phase introduced in Section 5.2.3. Second, to assess the practical scalability of our algorithms, we evaluate them on problem instances with varying numbers m of generalized bound constraints and numbers K of variables associated with a given constraint. Since for Problem QRA-NonSep-GBC no other tailored algorithms are available, we compare the performance of our algorithms to that of the commercial solver MOSEK [121].

In Section 5.6.1, we describe in more detail the problem instances that we use in the evaluations. Subsequently, in Section 5.6.2, we discuss several implementation details. Finally, in Section 5.6.3, we present and discuss the evaluation results.

5.6.1 PROBLEM INSTANCES

To evaluate the performance of our algorithms on instances of Problem EV-3Phase, we consider a setting wherein an EV is empty and available for residential charging from 18:00h and must be fully charged by 8:00h on the next day. This charging horizon of 14 hours is divided into 15-minute time intervals, meaning that $m = 56$ and that $\Delta t = \frac{1}{4}$. For the power consumption constraints of the EV, we follow the balancing framework in [174] and use the Tesla Model 3 as a reference EV [4]. This means that we choose $\bar{R} = 40,000$ (Wh), $\tilde{L}_j = 0$ (W) and $\tilde{U}_j = 11,500$ (W) for each $j \in \mathcal{M}$, and $\tilde{l}_i = -\frac{11,500}{3}$ (W) and $\tilde{u}_i = \frac{11,500}{3}$ (W) for each $i \in \mathcal{N}$. We simulate 100 charging sessions, where each session corresponds to charging on a different day. As input for the different days, we use power consumption measurement data of 40 households for 100 consecutive days that were obtained in the field test described in [80]. Thereby, we distribute the single-phase power consumption profiles of these 40 households randomly over the three phases, i.e., we assign each profile to one of the three phases. For the usage for a given day, the total load $q_{j,p}$ during interval j on phase p is given as the sum of the power consumption of all households assigned to phase p during that interval j . To study the influence of different trade-offs between the two objectives (minimizing peak consumption and minimizing load unbalance) on the time required to solve the problem, we simulate each of the 100 charging sessions using three different combinations of the weights W_1 and W_2 , namely $(W_1, W_2) \in \{(1, 1), (1, 100), (100, 1)\}$.

We now describe the instances of the general Problem QRA-NonSep-GBC used to assess the scalability of our algorithms. We focus on the case where the subset sizes n_j are equal to some positive integer K . We generate random instances for a number of fixed values of K and numbers of subsets m (Table 5.3 shows for the used values of K and m the uniform distribution used for generating the

parameter values). For each combination of K and m , we generate 10 instances according to the given distributions. Hereby, the distribution of each weight w_j is chosen such that the resulting problem instances satisfy Property 5.1, which ensures by Lemma 5.1 that their objective functions are strictly convex. The distributions of the values L_j and U_j are chosen such that none of the generalized bound constraints (5.1b) is redundant. As a consequence, all of these constraints need to be replaced according to the constraint simplification procedure in Section 5.3.2. In this way, we maximize the time that Algorithms 5.2 and 5.3 require for this step and thus improve the fairness of the comparison with MOSEK.

Parameter	Values
K	$\{1; 2; 5; 10; 20; 50; 100; 200; 500; 1,000\}$
m	$\{1; 2; 5; 10; 20; 50; 100; 200; 500; 1,000\}$
a_j	$\sim 1/U(0, 10)$
b_i	$\sim U(-10, 10)$
w_j	$\sim U\left(-\frac{1}{\sum_{i \in \mathcal{N}_j} a_j}, -\frac{1}{\sum_{i \in \mathcal{N}_j} a_j} + 10\right)$
l_i	$\sim U(-10, 0)$
u_i	$\sim U(0, 10)$
L_j	$\sim U\left(\sum_{i \in \mathcal{N}_j} l_i, 0.8 \sum_{i \in \mathcal{N}_j} l_i\right)$
U_j	$\sim U\left(0.8 \sum_{i \in \mathcal{N}_j} u_i, \sum_{i \in \mathcal{N}_j} u_i\right)$
R	$\sim U\left(\sum_{j=1}^m L_j, \sum_{j \in \mathcal{M}} U_j\right)$

Table 5.3: Parameter choices for the scalability evaluation.

5.6.2 IMPLEMENTATION DETAILS

For solving the subproblems $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ in Line 4 of both Algorithms 5.2 and 5.3, we implemented a sophisticated version of the sequential breakpoint search algorithm in Section 3.3.1, Algorithm 3.1, that allows us to solve both subproblems simultaneously and approximately twice as fast as the original algorithm. Furthermore, in Algorithm 5.3, we compute the candidate breakpoint, i.e., the median of a breakpoint set, in the same way as in Algorithm 5.2, namely by sorting the original breakpoint set and retrieving the desired breakpoints in $O(1)$ time (see also Section 5.5.1).

In both algorithms, it would be possible to reduce the time complexity of sorting all breakpoints from $O(n \log n)$ to $O(n \log m)$ by using a multi-way merging algorithm (see, e.g., [102]) to merge the $2m$ sorted lists of breakpoints. However, preliminary testing has shown that in both algorithms the time needed for sorting the breakpoints using a standard sorting algorithm is at least one order of magnitude smaller than the time needed for computing the breakpoints and carrying out the breakpoint search. Thus, we have chosen not to use a multi-way

merging algorithm to simplify the implementation of the algorithms without significantly increasing the overall execution time.

5.6.3 RESULTS

In this section we present and discuss the results of the evaluation as described in Section 5.6.1. First, we focus on the performance of our algorithms on the instances of Problem EV-3Phase. Table 5.4 shows the average execution times of our algorithms and of MOSEK for each combination of weights. Moreover, Figure 5.4 shows for each combination of weights the boxplots of the ratios between the execution times of each combination of algorithms. The results in Figure 5.4 indicate that our algorithms solve realistic instances of Problem EV-3Phase four to five times as fast as MOSEK for each weight combination. Moreover, Algorithm 5.3 appears to be slightly faster than Algorithm 5.2, although the difference in their execution times is less than 4% of the execution time of Algorithm 5.3 for the majority of the problem instances. The results in Table 5.4 and Figure 5.4 suggest that the choice of weights has little to no effect on the execution times of both our algorithms and MOSEK. The results in Table 5.4 indicate that our algorithms can solve realistic instances of Problem EV-3Phase in the order of milliseconds. This is significantly lower than common speed and delay requirements for communication networks in DEM systems [43]. Thus, our algorithms will most likely not be the (computational) bottleneck in DEM systems and are therefore suitable for integration in such systems.

Weight combination	Execution time (s)		
	Algorithm 5.2	Algorithm 5.3	MOSEK
(1, 1)	$4.64 \cdot 10^{-3}$	$4.89 \cdot 10^{-3}$	$2.33 \cdot 10^{-2}$
(1, 100)	$4.69 \cdot 10^{-3}$	$4.94 \cdot 10^{-3}$	$2.11 \cdot 10^{-2}$
(100, 1)	$4.71 \cdot 10^{-3}$	$4.86 \cdot 10^{-3}$	$2.07 \cdot 10^{-2}$

Table 5.4: Average execution times of Algorithms 5.2 and 5.3 and MOSEK for each combination of weights.

In the following, we discuss the results of the scalability evaluation. For this, we first compare the performance of the two different breakpoint search approaches, since these are the only aspect in which Algorithms 5.2 and 5.3 are different. To this end, we show in Figure 5.5 for each combination of K and m the boxplot of the ratios between the execution times of the sequential breakpoint search in Algorithm 5.2 and of the binary breakpoint search in Algorithm 5.3, i.e., the execution time of the breakpoint search procedure of Algorithm 5.2 divided by that of Algorithm 5.3. Moreover, Figure 5.6 shows for each combination of K and m the boxplot of ratios between the overall execution times of Algorithms 5.2 and 5.3. The results in Figure 5.5 indicate that for $K \leq 10$ the ratios regarding the breakpoint search procedures decrease significantly as m increases. For these values of K , most of the ratios are greater than 1 when $m \leq 5$ and smaller than 1

Figure 5.4: Boxplots of the ratios of the execution times between MOSEK and Algorithm 5.2 ($\frac{\text{MOS.}}{5.2}$), MOSEK and Algorithm 5.3 ($\frac{\text{MOS.}}{5.3}$), and Algorithms 5.3 and 5.2 ($\frac{5.3}{5.2}$) for each combination of weights.

when $m \geq 10$. This suggests that the binary breakpoint search procedure is faster than the sequential breakpoint search procedure when $m \geq 10$. For $K > 10$, the relation between the ratios and m is less clear. However, for each of these values of K , most of the ratios are greater than 1 for almost every value of m , which suggests that the binary breakpoint procedure in general outperforms the sequential breakpoint procedure.

It should be noted that the differences in execution time of the breakpoint searches of Algorithms 5.2 and 5.3 are less than an order of magnitude. Since the breakpoint search is the only aspect in which the algorithms differ, we expect that the differences in execution time of the entire algorithms are even less. This is confirmed by the results in Figure 5.6, i.e., in almost all cases, the difference in execution time between the two algorithms is significantly less than a factor 2. However, the behavior of these ratios is similar to that of those in Figure 5.5. For example, for $K \leq 10$, most ratios are larger than 1 when $m \leq 5$ and smaller than 1 when $m \geq 10$, whereas for $K > 10$ most ratios are greater than 1. This suggests that Algorithm 5.3 is in general faster than Algorithm 5.2 unless $K \leq 10$ and $m \leq 5$.

The marginal difference in overall execution time between Algorithms 5.2 and 5.3 is also confirmed by the results presented in Table 5.5, which shows the fitted power laws for Algorithms 5.2 and 5.3, i.e., for each K , we fit the function $f(m) = c_1 \cdot m^{c_2}$ to the execution times corresponding to K . These power laws suggest that the execution time of both algorithms grows linearly as m increases, i.e., all the exponents c_2 in Table 5.5 are close to one. This observation is consistent with the theoretical worst-case complexity of Algorithm 5.3, which is $O(n \log K) = O(mK \log K)$ and demonstrates its practical scalability.

Finally, we compare the performance of our algorithms to MOSEK. To this end, Figure 5.7 shows for each considered combination of K and m the execution time of Algorithm 5.3 and MOSEK. Moreover, Figure 5.8 shows all these execution

Figure 5.5: Boxplots of ratios between the execution times of the breakpoint search procedures of Algorithms 5.2 and 5.3, i.e., the execution time of the breakpoint search procedure of Algorithm 5.2 divided by that of Algorithm 5.3.

Figure 5.6: Boxplots of ratios between the execution times of Algorithms 5.2 and 5.3, i.e., the execution time of Algorithm 5.2 divided by that of Algorithm 5.3

K	Algorithm 5.2	Algorithm 5.3
1	$7.27 \cdot 10^{-5} \cdot m^{0.827}$	$5.52 \cdot 10^{-5} \cdot m^{0.912}$
2	$1.08 \cdot 10^{-4} \cdot m^{0.899}$	$9.31 \cdot 10^{-5} \cdot m^{0.944}$
5	$1.40 \cdot 10^{-4} \cdot m^{0.932}$	$1.27 \cdot 10^{-4} \cdot m^{0.956}$
10	$1.91 \cdot 10^{-4} \cdot m^{0.961}$	$1.78 \cdot 10^{-4} \cdot m^{0.970}$
20	$3.06 \cdot 10^{-4} \cdot m^{0.977}$	$2.87 \cdot 10^{-4} \cdot m^{0.977}$
50	$6.69 \cdot 10^{-4} \cdot m^{0.987}$	$6.34 \cdot 10^{-4} \cdot m^{0.985}$
100	$1.22 \cdot 10^{-3} \cdot m^{0.993}$	$1.11 \cdot 10^{-3} \cdot m^{0.993}$
200	$2.50 \cdot 10^{-3} \cdot m^{0.983}$	$2.15 \cdot 10^{-3} \cdot m^{0.995}$
500	$5.77 \cdot 10^{-3} \cdot m^{0.991}$	$5.24 \cdot 10^{-3} \cdot m^{0.998}$
1,000	$1.14 \cdot 10^{-2} \cdot m^{1.009}$	$1.03 \cdot 10^{-2} \cdot m^{1.003}$

Table 5.5: Power law regression functions for Algorithms 5.2 and 5.3 for each K , i.e., the fitted functions $c_1 \cdot m^{c_2}$.

times in dependence of the total number of variables $n(= Km)$. We do not plot the execution times of Algorithm 5.2 in these figures, since the differences in execution time between Algorithms 5.2 and 5.3 are so small that plotting them together in the same figure would unnecessarily obscure the results. Note that for $K = 1,000$, MOSEK was unable to solve any of the instances for $m = 500$ and $m = 1,000$ due to out-of-memory errors. Finally, to provide additional insight into the reported execution times, Tables 5.6-5.8 in Appendix 5.9 show for each combination of K and m the average execution time of Algorithms 5.2 and 5.3 and MOSEK respectively.

The execution times of MOSEK for $K \leq 50$ in Figures 5.7a-5.7f are quite large compared to the size of the problems. This is most likely due to the initialization time of MOSEK, which for smaller problem instances is relatively large compared to the actual time required by the internal solver to solve the corresponding instance. As a consequence, Algorithms 5.2 and 5.3 are at least one order of magnitude faster than MOSEK for instances with $K \leq 50$ and $m \leq 10$.

For $K \geq 100$, the results in Figure 5.7 and Tables 5.6-5.8 indicate that Algorithms 5.2 and 5.3 are at least one order of magnitude faster than MOSEK regardless of m . In fact, for $K = 1,000$, both our algorithms are even two orders of magnitude faster. For these instances, our algorithms solve all instances with $m = 500$ and $m = 1,000$ in less than 16 seconds (Algorithm 5.2) and 12 seconds (Algorithm 5.3), whereas MOSEK was unable to compute a solution due to out-of-memory errors.

5.7 CONCLUDING REMARKS

In this chapter, we studied a quadratic nonseparable resource allocation problem (RAP) with generalized bound constraints. This problem was motivated by its application in decentralized energy management (DEM) and in particular

Figure 5.7: Execution times of Algorithm 5.3 (circles, black) and MOSEK (triangles, gray) in dependence of K and m .

Figure 5.8: Execution times of Algorithm 5.3 (circles, black) and MOSEK (triangles, gray) in dependence of n .

for scheduling electric vehicles (EVs) to minimize load unbalance in electricity networks. We derived two algorithms with $O(n \log n)$ time complexity for this problem, of which one runs in linear time for a subclass containing the EV scheduling problem. Numerical evaluations demonstrate the practical efficiency of our algorithms both for realistic instances of the EV scheduling problem and for instances with synthetic data. In fact, our algorithms solve problem instances with up to 1 million variables in less than 16 seconds on a personal computer and are up to 100 times faster than a standard commercial solver. This practical efficiency of our algorithms makes them suitable for the aforementioned EV scheduling problems since these problems have to be solved in a short time on embedded systems with low computational power and low memory.

From a theoretical perspective, the results presented in this chapter add a new problem to the class of quadratic nonseparable RAPs that can be solved efficiently, i.e., in (strongly) polynomial time. The question remains how this class can be extended further. Existing work on optimization under submodular constraints [75, 120] suggests that the class of nonseparable RAPs where both the constraints and nonseparability are induced by so-called cross-free families (see also Section 2.7) constitutes a promising direction for this extension. We expect that new efficient and practical algorithms can be obtained for these problems by combining insights from existing methodologies to solve similar problems, including minimum quadratic cost flow problems [75, 163], scaling algorithms [120], and monotonicity-based optimization ([172], Chapter 4, and this chapter).

With regard to the application for DEM, such algorithms can be used to solve local optimization problems of other devices that draw their power from all three phases simultaneously other than EVs. As already mentioned in Section 4.5, in particular the derivation of an algorithm for the (quadratic) nonseparable RAP with nested constraints is an interesting direction for future research since this models the problem of scheduling large-scale batteries with three-phase chargers. Such batteries are widely recognized as vital components of current and future

residential distribution grids with a high infeed from renewable energy sources and integrated devices such as EVs (see also Section 2.5.2). Therefore, this is a relevant and important direction of future research that can contribute greatly to a sustainable future energy supply.

5.8 APPENDIX: FORMULATION OF PROBLEM EV-3PHASE

In this appendix, we derive the expression in Equation (5.2) for the objective of minimizing load unbalance and show that Problem EV-3Phase is an instance of Problem QRA-NonSep-GBC.

First, as a measure for load unbalance during a given interval $j \in \mathcal{M}$, we utilize the squared 2-norm of the resulting vector of the three phase loads $q_{j,1} + z_{j,1}$, $q_{j,2} + z_{j,2}$, and $q_{j,3} + z_{j,3}$ according to the phase arrangement depicted in Figure 5.1. This resulting vector equals

$$P_j^{\text{res}} := \left[\sum_{p=1}^3 (q_{j,p} + z_{j,p}) \cos \phi_p, \sum_{p=1}^3 (q_{j,p} + z_{j,p}) \sin \phi_p \right],$$

where ϕ_1, ϕ_2 , and ϕ_3 are the angles of the three phases. Thus, we can model the objective of minimizing unbalance by minimizing the function $\sum_{j \in \mathcal{M}} \|P_j^{\text{res}}\|^2$, where $\|\cdot\|$ denotes the 2-norm on $\mathbb{R}^2$. Note that we can assume without loss of generality that the phases are arranged as depicted in Fig. 5.1. This means that we may assume that $\phi_1 = 1\frac{5}{6}\pi$, $\phi_2 = 1\frac{1}{6}\pi$, and $\phi_3 = \frac{1}{2}\pi$. Thus, for each $j \in \mathcal{M}$, it follows that

$$\begin{aligned} \|P_j^{\text{res}}\|^2 &= \left(\sum_{p=1}^3 (q_{j,p} + z_{j,p}) \cos \phi_p \right)^2 + \left(\sum_{p=1}^3 (q_{j,p} + z_{j,p}) \sin \phi_p \right)^2 \\ &= \sum_{p=1}^3 (q_{j,p} + z_{j,p})^2 (\cos^2 \phi_p + \sin^2 \phi_p) \\ &\quad + 2 \sum_{\substack{p, p' \in \{1, 2, 3\}: \\ p \neq p'}} ((q_{j,p} + z_{j,p}) (q_{j,p'} + z_{j,p'}) (\cos \phi_p \cos \phi_{p'} + \sin \phi_p \sin \phi_{p'})) \\ &= \sum_{p=1}^3 (q_{j,p} + z_{j,p})^2 + 2(q_{j,1} + z_{j,1})(q_{j,2} + z_{j,2}) \left(-\frac{3}{4} + \frac{1}{4} \right) \\ &\quad + 2(q_{j,1} + z_{j,1})(q_{j,3} + z_{j,3}) \left(0 - \frac{1}{2} \right) \\ &\quad + 2(q_{j,2} + z_{j,2})(q_{j,3} + z_{j,3}) \left(0 - \frac{1}{2} \right) \end{aligned}$$

$$\begin{aligned}
&= \sum_{p=1}^3 (q_{j,p} + z_{j,p})^2 - (q_{j,1} + z_{j,1})(q_{j,2} + z_{j,2}) \\
&\quad - (q_{j,1} + z_{j,1})(q_{j,3} + z_{j,3}) - (q_{j,2} + z_{j,2})(q_{j,3} + z_{j,3}) \\
&= \frac{3}{2} \sum_{p=1}^3 (q_{j,p} + z_{j,p})^2 - \frac{1}{2} \left(\sum_{p=1}^3 (q_{j,p} + z_{j,p}) \right)^2.
\end{aligned}$$

Second, to show that Problem EV-3Phase is an instance of Problem QRA-NonSep-GBC, observe that the objective function of Problem EV-3Phase can be rewritten to

$$\begin{aligned}
&W_1 \sum_{j \in \mathcal{M}} \left(\sum_{p=1}^3 (q_{j,p} + z_{j,p}) \right)^2 \\
&\quad + W_2 \sum_{j \in \mathcal{M}} \left(\frac{3}{2} \sum_{p=1}^3 (q_{j,p} + z_{j,p})^2 - \frac{1}{2} \left(\sum_{p=1}^3 (q_{j,p} + z_{j,p}) \right)^2 \right) \\
&= \left(W_1 - \frac{1}{2} W_2 \right) \sum_{j \in \mathcal{M}} \left(\sum_{p=1}^3 (q_{j,p} + z_{j,p}) \right)^2 + \frac{3}{2} W_2 \sum_{j \in \mathcal{M}} \sum_{p=1}^3 (q_{j,p} + z_{j,p})^2 \\
&= \left(W_1 - \frac{1}{2} W_2 \right) \sum_{j \in \mathcal{M}} \left(\sum_{p=1}^3 z_{j,p} \right)^2 + \frac{3}{2} W_2 \sum_{j \in \mathcal{M}} \sum_{p=1}^3 z_{j,p}^2 \\
&\quad + \left(W_1 - \frac{1}{2} W_2 \right) \sum_{j \in \mathcal{M}} \left(\sum_{p=1}^3 q_{j,p} \right) \sum_{p=1}^3 z_{j,p} + \frac{3}{2} W_2 \sum_{j \in \mathcal{M}} \sum_{p=1}^3 q_{j,p} z_{j,p} \\
&\quad + \left(W_1 - \frac{1}{2} W_2 \right) \sum_{j \in \mathcal{M}} \left(\sum_{p=1}^3 q_{j,p} \right)^2 + \frac{3}{2} W_2 \sum_{j \in \mathcal{M}} \sum_{p=1}^3 q_{j,p}^2.
\end{aligned}$$

The latter expression leads directly to the values in Table 5.1.

5.9 APPENDIX: AVERAGE EXECUTION TIMES OF ALGORITHMS 5.2 AND 5.3 AND MOSEK

$K \setminus m$	1	2	5	10	20	50	100	200	500	1,000
1	$1.07 \cdot 10^{-4}$	$1.49 \cdot 10^{-4}$	$2.38 \cdot 10^{-4}$	$3.68 \cdot 10^{-4}$	$8.87 \cdot 10^{-4}$	$1.51 \cdot 10^{-3}$	$2.94 \cdot 10^{-3}$	$5.75 \cdot 10^{-3}$	$1.47 \cdot 10^{-2}$	$2.75 \cdot 10^{-2}$
2	$1.58 \cdot 10^{-4}$	$2.06 \cdot 10^{-4}$	$3.93 \cdot 10^{-4}$	$7.23 \cdot 10^{-4}$	$1.34 \cdot 10^{-3}$	$3.29 \cdot 10^{-3}$	$6.49 \cdot 10^{-3}$	$1.34 \cdot 10^{-2}$	$3.13 \cdot 10^{-2}$	$6.29 \cdot 10^{-2}$
5	$1.75 \cdot 10^{-4}$	$2.77 \cdot 10^{-4}$	$5.97 \cdot 10^{-4}$	$1.08 \cdot 10^{-3}$	$1.92 \cdot 10^{-3}$	$4.96 \cdot 10^{-3}$	$9.91 \cdot 10^{-3}$	$2.09 \cdot 10^{-2}$	$4.78 \cdot 10^{-2}$	$9.89 \cdot 10^{-2}$
10	$2.40 \cdot 10^{-4}$	$3.72 \cdot 10^{-4}$	$8.04 \cdot 10^{-4}$	$1.61 \cdot 10^{-3}$	$2.89 \cdot 10^{-3}$	$8.73 \cdot 10^{-3}$	$1.60 \cdot 10^{-2}$	$3.22 \cdot 10^{-2}$	$7.95 \cdot 10^{-2}$	$1.53 \cdot 10^{-1}$
20	$3.63 \cdot 10^{-4}$	$5.61 \cdot 10^{-4}$	$1.41 \cdot 10^{-3}$	$2.88 \cdot 10^{-3}$	$5.29 \cdot 10^{-3}$	$1.37 \cdot 10^{-2}$	$2.94 \cdot 10^{-2}$	$5.40 \cdot 10^{-2}$	$1.33 \cdot 10^{-1}$	$2.72 \cdot 10^{-1}$
50	$6.63 \cdot 10^{-4}$	$1.54 \cdot 10^{-3}$	$3.09 \cdot 10^{-3}$	$6.56 \cdot 10^{-3}$	$1.36 \cdot 10^{-2}$	$3.09 \cdot 10^{-2}$	$6.35 \cdot 10^{-2}$	$1.29 \cdot 10^{-1}$	$3.10 \cdot 10^{-1}$	$6.10 \cdot 10^{-1}$
100	$1.16 \cdot 10^{-3}$	$2.84 \cdot 10^{-3}$	$6.28 \cdot 10^{-3}$	$1.22 \cdot 10^{-2}$	$2.33 \cdot 10^{-2}$	$6.12 \cdot 10^{-2}$	$1.14 \cdot 10^{-1}$	$2.20 \cdot 10^{-1}$	$6.29 \cdot 10^{-1}$	$1.18 \cdot 10^0$
200	$3.45 \cdot 10^{-3}$	$4.64 \cdot 10^{-3}$	$1.38 \cdot 10^{-2}$	$2.37 \cdot 10^{-2}$	$4.66 \cdot 10^{-2}$	$1.10 \cdot 10^{-1}$	$2.15 \cdot 10^{-1}$	$4.56 \cdot 10^{-1}$	$1.11 \cdot 10^0$	$2.45 \cdot 10^0$
500	$6.08 \cdot 10^{-3}$	$1.13 \cdot 10^{-2}$	$2.87 \cdot 10^{-2}$	$5.78 \cdot 10^{-2}$	$1.11 \cdot 10^{-1}$	$2.83 \cdot 10^{-1}$	$5.29 \cdot 10^{-1}$	$1.04 \cdot 10^0$	$3.01 \cdot 10^0$	$5.43 \cdot 10^0$
1,000	$1.36 \cdot 10^{-2}$	$2.17 \cdot 10^{-2}$	$5.84 \cdot 10^{-2}$	$1.09 \cdot 10^{-1}$	$2.34 \cdot 10^{-1}$	$5.43 \cdot 10^{-1}$	$1.12 \cdot 10^0$	$2.28 \cdot 10^0$	$6.89 \cdot 10^0$	$1.27 \cdot 10^1$

Table 5.6: Average execution times (s) of Algorithm 5.2 for each combination of K and m .

$K \setminus m$	1	2	5	10	20	50	100	200	500	1,000
1	$7.31 \cdot 10^{-5}$	$1.11 \cdot 10^{-4}$	$2.11 \cdot 10^{-4}$	$3.77 \cdot 10^{-4}$	$8.84 \cdot 10^{-4}$	$1.69 \cdot 10^{-3}$	$3.76 \cdot 10^{-3}$	$6.90 \cdot 10^{-3}$	$1.73 \cdot 10^{-2}$	$3.50 \cdot 10^{-2}$
2	$1.20 \cdot 10^{-4}$	$1.80 \cdot 10^{-4}$	$3.79 \cdot 10^{-4}$	$7.23 \cdot 10^{-4}$	$1.49 \cdot 10^{-3}$	$3.69 \cdot 10^{-3}$	$6.74 \cdot 10^{-3}$	$1.47 \cdot 10^{-2}$	$3.48 \cdot 10^{-2}$	$6.93 \cdot 10^{-2}$
5	$1.47 \cdot 10^{-4}$	$2.63 \cdot 10^{-4}$	$5.59 \cdot 10^{-4}$	$1.08 \cdot 10^{-3}$	$1.98 \cdot 10^{-3}$	$4.95 \cdot 10^{-3}$	$1.01 \cdot 10^{-2}$	$2.01 \cdot 10^{-2}$	$5.07 \cdot 10^{-2}$	$1.03 \cdot 10^{-1}$
10	$1.98 \cdot 10^{-4}$	$3.55 \cdot 10^{-4}$	$8.26 \cdot 10^{-4}$	$1.61 \cdot 10^{-3}$	$2.99 \cdot 10^{-3}$	$7.82 \cdot 10^{-3}$	$1.56 \cdot 10^{-2}$	$3.01 \cdot 10^{-2}$	$7.63 \cdot 10^{-2}$	$1.55 \cdot 10^{-1}$
20	$3.10 \cdot 10^{-4}$	$5.57 \cdot 10^{-4}$	$1.35 \cdot 10^{-3}$	$2.56 \cdot 10^{-3}$	$5.94 \cdot 10^{-3}$	$1.27 \cdot 10^{-2}$	$2.52 \cdot 10^{-2}$	$5.00 \cdot 10^{-2}$	$1.26 \cdot 10^{-1}$	$2.55 \cdot 10^{-1}$
50	$6.44 \cdot 10^{-4}$	$1.34 \cdot 10^{-3}$	$3.20 \cdot 10^{-3}$	$6.23 \cdot 10^{-3}$	$1.20 \cdot 10^{-2}$	$2.85 \cdot 10^{-2}$	$5.74 \cdot 10^{-2}$	$1.23 \cdot 10^{-1}$	$2.89 \cdot 10^{-1}$	$5.84 \cdot 10^{-1}$
100	$1.17 \cdot 10^{-3}$	$2.17 \cdot 10^{-3}$	$6.03 \cdot 10^{-3}$	$1.05 \cdot 10^{-2}$	$2.14 \cdot 10^{-2}$	$5.19 \cdot 10^{-2}$	$1.02 \cdot 10^{-1}$	$2.06 \cdot 10^{-1}$	$5.62 \cdot 10^{-1}$	$1.12 \cdot 10^{+0}$
200	$2.24 \cdot 10^{-3}$	$4.23 \cdot 10^{-3}$	$1.29 \cdot 10^{-2}$	$2.04 \cdot 10^{-2}$	$4.03 \cdot 10^{-2}$	$1.00 \cdot 10^{-1}$	$2.07 \cdot 10^{-1}$	$4.05 \cdot 10^{-1}$	$1.05 \cdot 10^{+0}$	$2.22 \cdot 10^{+0}$
500	$5.69 \cdot 10^{-3}$	$1.05 \cdot 10^{-2}$	$2.53 \cdot 10^{-2}$	$5.26 \cdot 10^{-2}$	$9.80 \cdot 10^{-2}$	$2.55 \cdot 10^{-1}$	$5.06 \cdot 10^{-1}$	$1.02 \cdot 10^{+0}$	$2.63 \cdot 10^{+0}$	$5.42 \cdot 10^{+0}$
1,000	$1.17 \cdot 10^{-2}$	$2.00 \cdot 10^{-2}$	$4.98 \cdot 10^{-2}$	$1.00 \cdot 10^{-1}$	$2.00 \cdot 10^{-1}$	$5.14 \cdot 10^{-1}$	$1.04 \cdot 10^{+0}$	$2.07 \cdot 10^{+0}$	$5.39 \cdot 10^{+0}$	$1.10 \cdot 10^{+1}$

Table 5.7: Average execution times (s) of Algorithm 5.3 for each combination of K and m .

$K \setminus m$	1	2	5	10	20	50	100	200	500	1,000
1	$1.09 \cdot 10^{-2}$	$1.26 \cdot 10^{-2}$	$1.26 \cdot 10^{-2}$	$1.26 \cdot 10^{-2}$	$1.45 \cdot 10^{-2}$	$1.26 \cdot 10^{-2}$	$4.19 \cdot 10^{-2}$	$1.89 \cdot 10^{-2}$	$2.33 \cdot 10^{-2}$	$2.95 \cdot 10^{-2}$
2	$1.21 \cdot 10^{-2}$	$1.20 \cdot 10^{-2}$	$1.26 \cdot 10^{-2}$	$1.70 \cdot 10^{-2}$	$1.80 \cdot 10^{-2}$	$2.29 \cdot 10^{-2}$	$2.13 \cdot 10^{-2}$	$2.87 \cdot 10^{-2}$	$4.29 \cdot 10^{-2}$	$7.25 \cdot 10^{-2}$
5	$1.21 \cdot 10^{-2}$	$1.20 \cdot 10^{-2}$	$1.25 \cdot 10^{-2}$	$1.24 \cdot 10^{-2}$	$1.53 \cdot 10^{-2}$	$2.00 \cdot 10^{-2}$	$2.83 \cdot 10^{-2}$	$4.23 \cdot 10^{-2}$	$7.94 \cdot 10^{-2}$	$1.36 \cdot 10^{-1}$
10	$1.23 \cdot 10^{-2}$	$1.27 \cdot 10^{-2}$	$1.42 \cdot 10^{-2}$	$1.57 \cdot 10^{-2}$	$2.61 \cdot 10^{-2}$	$3.55 \cdot 10^{-2}$	$4.60 \cdot 10^{-2}$	$7.39 \cdot 10^{-2}$	$1.57 \cdot 10^{-1}$	$3.15 \cdot 10^{-1}$
20	$1.27 \cdot 10^{-2}$	$1.38 \cdot 10^{-2}$	$1.68 \cdot 10^{-2}$	$2.09 \cdot 10^{-2}$	$3.31 \cdot 10^{-2}$	$5.26 \cdot 10^{-2}$	$8.72 \cdot 10^{-2}$	$1.54 \cdot 10^{-1}$	$3.65 \cdot 10^{-1}$	$8.12 \cdot 10^{-1}$
50	$1.52 \cdot 10^{-2}$	$1.98 \cdot 10^{-2}$	$4.20 \cdot 10^{-2}$	$5.41 \cdot 10^{-2}$	$7.62 \cdot 10^{-2}$	$1.61 \cdot 10^{-1}$	$3.27 \cdot 10^{-1}$	$6.35 \cdot 10^{-1}$	$1.55 \cdot 10^0$	$3.27 \cdot 10^0$
100	$2.34 \cdot 10^{-2}$	$4.32 \cdot 10^{-2}$	$7.07 \cdot 10^{-2}$	$1.20 \cdot 10^{-1}$	$2.54 \cdot 10^{-1}$	$5.83 \cdot 10^{-1}$	$1.21 \cdot 10^0$	$2.15 \cdot 10^0$	$5.85 \cdot 10^0$	$1.20 \cdot 10^1$
200	$5.70 \cdot 10^{-2}$	$1.00 \cdot 10^{-1}$	$2.21 \cdot 10^{-1}$	$4.42 \cdot 10^{-1}$	$8.32 \cdot 10^{-1}$	$2.03 \cdot 10^0$	$4.21 \cdot 10^0$	$8.73 \cdot 10^0$	$2.40 \cdot 10^1$	$4.41 \cdot 10^1$
500	$3.08 \cdot 10^{-1}$	$5.92 \cdot 10^{-1}$	$1.55 \cdot 10^0$	$3.30 \cdot 10^0$	$5.98 \cdot 10^0$	$1.63 \cdot 10^1$	$2.98 \cdot 10^1$	$6.20 \cdot 10^1$	$1.60 \cdot 10^2$	$3.51 \cdot 10^2$
1,000	$1.06 \cdot 10^0$	$3.05 \cdot 10^0$	$7.37 \cdot 10^0$	$1.54 \cdot 10^1$	$3.41 \cdot 10^1$	$7.84 \cdot 10^1$	$1.71 \cdot 10^2$	$3.75 \cdot 10^2$	-	-

Table 5.8: Average execution times (s) of MOSEK for each combination of K and m .

ONLINE DUALITY-DRIVEN OPTIMIZATION

ABSTRACT – In this chapter, we focus on sequential decision-making under data uncertainty in energy management problems. More generally, we study convex optimization problems with uncertainty in the objective function. To deal with these problems, we propose a new framework called “Online Duality-Driven Optimization” (ODDO). This framework distinguishes itself from existing paradigms for optimization under uncertainty in its efficiency, simplicity, and ability to solve problems without any quantitative assumptions on the uncertain data. The key idea behind this framework is that instead of the actual uncertain data, the optimal Lagrange multipliers are predicted. Subsequently, these predictions are used to construct an online primal solution by exploiting strong duality of the problem. We show that the framework is robust against prediction errors in the optimal Lagrange multipliers both theoretically and in practice. In fact, evaluations of the framework on problems with both measured and randomly generated input data show that ODDO can achieve near-optimal online solutions, even when only elementary statistics are used to predict the optimal Lagrange multipliers.

6.1 INTRODUCTION

Dealing with uncertainty plays an important role in modern decision-making. As a consequence, many paradigms have been developed for optimization under data uncertainty (we discuss several of them in Section 6.2.1). Each of these paradigms has its own pros and cons, and it depends strongly on the given application which of them is most suitable. Therefore, to focus the discussion, we consider a specific application domain as a leading example, namely optimization under uncertainty in decentralized energy management (DEM).

This chapter is based on and contains parts of [SU:3].

Within DEM, scheduling the usage of energy requires information on energy production and consumption in the upcoming time period, i.e., information that is in general unknown at the current time. Often, this information is directly predicted and a deterministic version of the scheduling problem at hand is considered based on this predicted data [16]. However, in practice, predicting especially the base load consumption of all non-steerable devices is difficult since it is to some extent equivalent to the prediction of human behavior on a small scale (see, e.g., [91]). For this reason, current research directions in this area focus on the development of advanced prediction approaches [78] and the application of existing frameworks for optimization under data uncertainty (see, e.g., [110]).

In this chapter, we consider a general class of constrained convex optimization problems with uncertainty in the objective function. In these problems, the optimization horizon is divided into stages and we associate with each stage a vector of decision variables and an uncertain cost function. At the start of a given stage, this uncertain cost function is revealed and the decision variables corresponding to this stage have to be decided. Depending on the application, this revelation can be based on, e.g., a measurement or a (very) short-term forecast of the cost function. The goal is to obtain in this way a feasible solution to the problem whose objective value is close to the optimum of the deterministic version of the optimization problem. We call this decision-making problem the *online* version or setting of the optimization problem. Note that deciding about the variables for a given stage is also related to the uncertain cost functions for the future stages since the given constraints of the problem also involve variables of future stages. Many scheduling problems in DEM fall into this problem class of constrained convex optimization problems (see, e.g., [167]). More details on these problems are given in Section 6.5.

There is quite some research that especially focuses on problems with only uncertainty in the objective function. Examples of such problems can be found in the fields of linear programming [42], network flow problems [25], distributed optimization [104], and multi-objective optimization [86]. Examples of applications are task scheduling on processors [59], production and inventory management [123], and speed optimization for truck and ship routing [84].

In this chapter, we propose a new framework for solving optimization problems with uncertain objective functions. We call this framework “Online Duality-Driven Optimization” (ODDO). The key idea in ODDO is that we can characterize the optimal decision vector for a given stage using the optimal Lagrange multipliers of the deterministic optimization problem and the parameters corresponding to the given stage. In particular, for this characterization, we require no information on the cost functions other than the one associated with the given stage. As a consequence, there is no need to predict these future cost functions and/or the uncertain data underlying these functions. Instead, we only require predictions of the optimal Lagrange multipliers of the deterministic version of

the problem, i.e., the version where all input data is known on forehand. This approach is especially beneficial in problems where the dimension of the dual space, i.e., the number of constraints, is low compared to the dimension of the uncertain data since it reduces the number of values that need to be predicted. Furthermore, we show that this approach is robust against prediction errors in the Lagrange multipliers. In particular, for a specific class of separable optimization problems with submodular constraints (see, e.g., [49, 75] and Section 2.3.5)), we derive several bounds on the difference in objective value between our online solution and the optimal offline solution, provided that we "under-predict" the optimal Lagrange multipliers (we define this type of prediction in Section 6.4.2).

We evaluate the performance of the ODDO framework by applying it to two problems. The first problem is the scheduling of a large neighborhood battery with the goal to minimize the net peak consumption of the neighborhood. This is an instance of Problem BATTERY introduced in Section 2.5.2. The second problem is an adaption of an inventory management problem studied in [21] that the authors used to introduce and demonstrate the concept of adjustable robust optimization, which is a well-known approach to solve specific optimization problems under uncertainty (see also Section 6.2.1). Simulation results show that in both cases, ODDO can achieve near-optimal online solutions and that these solutions can be achieved using easy-to-compute predictions for the optimal Lagrange multipliers. Moreover, the results indicate that ODDO is able to significantly outperform optimization over nominal or expected values, i.e., sample average approximation.

The outline of the remainder of this chapter is as follows. First, in Section 6.2, we provide some background on existing approaches for optimization under uncertainty and for incorporating multiplier predictions into online optimization. In Section 6.3, we formulate the studied class of optimization problems and present the ODDO framework for solving these problems in an online setting. In Section 6.4, we analyze the theoretical performance of the ODDO framework and prove several results regarding its robustness to prediction errors in its optimal Lagrange multipliers. In Section 6.5, we evaluate the performance of the ODDO framework and, finally, Section 6.6 contains the conclusions and discusses several limitations and possible extensions of the framework.

6.2 BACKGROUND

This section provides some background on existing approaches for optimization under uncertainty and predicting optimal Lagrange multipliers. Moreover, we provide a brief comparison of these approaches to ODDO.

6.2.1 EXISTING PARADIGMS FOR OPTIMIZATION UNDER UNCERTAINTY

The optimization problems considered in this chapter are concerned with sequential (or, alternatively, online or multistage) decision-making under uncer-

tainty. There exists a vast literature on such optimization problems (see [14] for a survey). Here, we briefly discuss three of the most prominent paradigms for solving these problems and their advantages and disadvantages, namely (multi-stage) stochastic programming, (adjustable) robust optimization, and online convex optimization.

Stochastic programming (SP) is applied to solve optimization problems where the uncertain data is assumed to be stochastic under a given joint probability distribution. The goal is to design a policy that maximizes the expected outcome of some objective function on the decision variables and under the constraint that all or almost all constraints are fulfilled for all possible data outcomes [152]. In particular, in a *multi-stage* SP, the aim is to iteratively derive a policy for a single (the current) stage and a set of policies for the future stages for each outcome of the stochastic data of the current stage.

The advantage of SP is that it takes all information on the probability distribution into account in the decision-making, which leads to more accurate results. However, a major drawback of SP is that knowledge of the distribution underlying the data is needed but not always available and cannot always be estimated well from historical data, e.g., due to the data not being stationary. Moreover, multi-stage SP problems are generally intractable [44] and, although several approaches exist to compute approximate solutions (see, e.g., [151]), the exponential growth of the dimension of the to-be-solved SP problem is a limiting factor for applications where fast computations are crucial.

A different approach is taken in robust optimization (RO), which, in contrast to SP, is a solution paradigm that does not assume any probability distribution on the uncertain data. Instead, the essential assumption in RO is that one can construct a so-called *uncertainty set* that contains all possible or likely realizations of the uncertain data [22]. Given such an uncertainty set, the goal is to compute the best worst-case solution to the problem, i.e., the best solution that is feasible for any realization of the data within the uncertainty set. A variant of RO for sequential or online decision-making is *adjustable* RO [180]. Here, one distinguishes between here-and-now variables, which have to be decided at the current stage, and wait-and-see variables, which can or must be decided in later stages.

The major advantage of RO and adjustable RO is that any quantitative information on the uncertain data can be taken into account when constructing the uncertainty set. Moreover, the decision-maker can control the level of conservatism involved in the process of determining a solution by adjusting the uncertainty set. However, a disadvantage of RO is that its solutions can be too conservative since not all realizations within the uncertainty set are equally likely to occur. Indeed, it is known that a careful design of the uncertainty set is crucial for the success of RO for solving real-life problems [62]. Another disadvantage is that solving a RO-model is in general much harder than solving the original deterministic problem and greatly increases the size of the problem at hand [23].

A third considered paradigm is online convex optimization (OCO). In the OCO framework [149], the goal is to solve convex optimization problems with objective uncertainty by learning the structure of the objective function “on the go”. At each stage, the then-available information on the objective function is used to determine the decision for this stage. Subsequently, the part of the objective function corresponding to this stage is revealed and the resulting cost is incurred.

The major advantage of this framework is that it requires no assumptions on the uncertainty in the objective function other than that this function is convex. Moreover, it provides performance guarantees in terms of both objective value and constraint violations [111]. However, some of these guarantees require that the optimization horizon is extremely large [92], which makes them less informative in practice for problems with smaller optimization horizons. Moreover, the practical performance of OCO heavily depends on estimates of several properties of the uncertain objective function such as its Lipschitz constant. Such an estimate depends on the actual uncertain data underlying the objective function and thus requires knowledge on this data itself. In any case, the classical OCO framework does not consider the availability of further information (e.g., obtained from measurements) on the part of the objective function corresponding to the current stage. It is only very recently that OCO-models are studied with so-called *1-step look-ahead* features, where such measurements are (partly) available (see, e.g., [73]).

6.2.2 MULTIPLIER PREDICTION

The idea of predicting optimal Lagrange multipliers shares similarities with active set learning for continuous and mixed-integer optimization [26, 118]. These works solve optimization problems by predicting the set of constraints that is active at the optimal solution. Subsequently, they use this set to reduce the original problem to a convex equality-constrained optimization problem that, depending on the linearity of the constraints, might be easier to solve. Active set learning and ODDO are similar in the sense that both approaches predict sets of values (active constraints or Lagrange multipliers) from which an optimal solution can be easily reconstructed. In fact, since the values of optimal Lagrange multipliers directly imply the set of active constraints by complementary slackness (see, e.g., [29]), active set learning is able to reconstruct an optimal solution with less information than ODDO. However, solving the aforementioned reduced equality-constrained problem requires all uncertain cost functions to be known a priori. As a consequence, the approaches in [26, 118] are only able to solve offline optimization problems and cannot be used to solve sequential optimization problems where uncertain data is revealed over time. In our approach, however, we reduce the original problem to a collection of smaller problems, one for each stage, that each can be solved without using input data that has not yet been revealed. This means that given a prediction of the optimal Lagrange multipliers,

we can compute a solution for a given stage using this prediction and with only data that has already been revealed.

It is worth mentioning that, in energy management, several specific optimization problems under uncertainty are already being solved by similar approaches, be it under a different name. For example, in the case of electric vehicle (EV) charging, this approach is known as *online valley filling* [34, 56, 122] and the single Lagrange multiplier that is predicted is called the *fill-level*. The ODDO framework generalizes these approaches to a setting with convex objective function and constraints.

6.2.3 COMPARISON TO ODDO

Our approach has several advantages compared to the existing paradigms for optimization under uncertainty that were mentioned in Section 6.2.1:

1. We do not pose any quantitative assumptions on the uncertain data in terms of support, underlying probability distribution, or uncertainty set. Instead, we only assume that the data possesses some structure, meaning that the data is similar for, e.g., consecutive problem instances.
2. A disadvantage of, e.g., SP and RO is the large increase in complexity of the problems that eventually have to be solved, e.g., the robust counterpart in RO. In our approach, however, at each stage, one has to solve a problem that is as easy as the original problem but with a slightly different, though still convex, objective function. The construction of this new objective function consists of adding a linear term to the objective. Thus, the online problem can be solved within the same (time) complexity as the deterministic problem.
3. Our approach does not only work for the case where underlying data is uncertain, but also when the structure of the cost functions (e.g., quadratic, exponential) is uncertain. The only information that we require of these functions is that they are convex and continuously differentiable on a given compact support set.
4. The validity of the performance guarantee that we derive is independent of the number of stages and the guarantee itself only depends on the prediction errors in the optimal Lagrange multipliers. As a consequence, and as opposed to some works on OCO, this guarantee holds for any number of stages and makes our approach successfully applicable also to problems with only a small number of stages.
5. The derivation and validation of our approach requires only well-known optimality and stability results from convex programming. This means that the approach is relatively easy to apply by practitioners.

For the envisioned application of our framework to energy management, these advantages are relevant in the following way:

1. We circumvent the problem of having to predict energy consumption data of individual households, which is in practice very difficult [91] and already an entire discipline on its own [78].
2. The limited increase in complexity of the online problem compared to the deterministic problem is relevant because the computation of device schedules within DEM often has to be done on embedded systems with relatively low computational power [19].
3. Depending on new information received by the measured energy consumption data within, e.g., the houses in a neighborhood, one may decide to change the operational mode of the devices within the neighborhood, e.g., to reduce the peak consumption of a particular group of houses.
4. The optimization horizon in DEM typically consists of only a few days that are divided into time slots of 10 or 15 minutes (see, e.g., [113]). This means that the total number of stages is only a few hundreds, whereas some of the performance guarantees and bounds in OCO, e.g., in [111] require that the number of stages is in the order of tens of millions.
5. The relative simplicity of our approach is relevant since energy management is a multi-disciplinary research area with many experts from engineering that might not have the time and/or mathematical knowledge to delve into the rather technical literature of, e.g., SP, RO, and OCO.

6.3 THE ODDO FRAMEWORK

In this section, we present the ODDO framework for solving online constrained convex optimization problems. We start this section by formulating the studied optimization problem in Section 6.3.1 and formalizing the online decision-making process. Subsequently, we revisit in Section 6.3.2 several basic results from Lagrangian duality theory. In Section 6.3.3, we present our approach. Note, that we postpone the motivation for and robustness analysis of the ODDO framework to Section 6.4. Finally, in Section 6.3.4, we provide an illustrative example of the ODDO framework.

6.3.1 PROBLEM FORMULATION

We consider a finite horizon consisting of T stages indexed by the set $\mathcal{T} := \{1, \dots, T\}$. With each stage $t \in \mathcal{T}$, we associate an index set $\mathcal{N}^t$ of size N^t and a decision vector $x^t := (x_i^t)_{i \in \mathcal{N}^t}$ of dimension N^t . Furthermore, for each stage $t \in \mathcal{T}$, a continuously differentiable and strictly convex cost function $f^t(x^t)$ is given. The objective is to minimize the sum of the cost functions over all stages. We consider two types of constraints. First, we impose for each stage t that x^t has to be chosen from a compact convex set $\mathcal{C}^t \subset \mathbb{R}^{N^t}$. Second, we consider both equality and inequality constraints that are separable over the stages, i.e.,

that are of the form

$$\begin{aligned} \sum_{t \in \mathcal{T}} g_j^t(x^t) &\leq 0, \quad j \in \mathcal{M}, \\ \sum_{t \in \mathcal{T}} h_k^t(x^t) &= 0, \quad k \in \mathcal{L}. \end{aligned}$$

Here $\mathcal{M}$ and $\mathcal{L}$ denote the index sets of the inequality and equality constraints respectively. Moreover, we assume that each function g_j^t is convex and continuously differentiable and each function h_k^t is affine. Summarizing, we obtain the following convex optimization problem:

$$\text{P : } \min_{x^1, \dots, x^T} \sum_{t \in \mathcal{T}} f^t(x^t) \quad (6.1a)$$

$$\text{s.t. } \sum_{t \in \mathcal{T}} g_j^t(x^t) \leq 0, \quad j \in \mathcal{M}, \quad (6.1b)$$

$$\sum_{t \in \mathcal{T}} h_k^t(x^t) = 0, \quad k \in \mathcal{L}, \quad (6.1c)$$

$$x^t \in \mathcal{C}^t, \quad t \in \mathcal{T}. \quad (6.1d)$$

In particular, we study Problem P in the online setting, i.e., the setting wherein the cost functions f^t are uncertain. In this online version, the nature of the uncertainty can be related to either uncertain input parameters, e.g., uncertain coefficients of a given polynomial, or an uncertain function type, e.g., polynomial or exponential. In any case, we assume that the uncertainty is such that f^t remains convex and continuously differentiable for any possible outcome of the uncertain data underlying the function or any possible structure of the function. Most importantly, however, *we do not assume any quantitative knowledge of the uncertainty*. This includes knowledge in terms of, e.g., types of cost functions, probability distributions, and uncertainty sets.

The decision-making process in the online setting is as follows. At each stage $t \in \mathcal{T}$, the corresponding cost function f^t is revealed and the decision-maker must decide on the corresponding decision vector x^t . The goal is to obtain a good online solution $\hat{x} := (\hat{x}^1, \dots, \hat{x}^T)$. In this context, “good” means that the objective value of the solution is close to the optimum of the deterministic counterpart of Problem P. Note that the presence of the constraints (6.1b) and (6.1c) implies that for deciding on x^t , the uncertainty for the remaining stages $t + 1, \dots, T$, i.e., the uncertain cost functions $f^{t+1}, \dots, f^T$, should be taken into consideration.

6.3.2 LAGRANGIAN DUALITY REVISITED

For formulating the dual problem related to Problem P, we introduce the Lagrange multipliers $\mu := (\mu_j)_{j \in \mathcal{M}}$ and $\lambda := (\lambda_k)_{k \in \mathcal{L}}$ that correspond to Constraints (6.1b) and (6.1c) respectively. For convenience in later sections, we

assume that both μ and λ are row vectors. Furthermore, we denote the concatenation of the two vectors μ and λ by (μ, λ) , meaning that (μ, λ) is a vector of dimension $|\mathcal{M}| + |\mathcal{L}|$. Using μ and λ , the Lagrangian $L(x^1, \dots, x^T, \mu, \lambda)$ of Problem P is given by

$$L(x^1, \dots, x^T, \mu, \lambda) := \sum_{t \in \mathcal{T}} f^t(x^t) + \sum_{j \in \mathcal{M}} \mu_j \sum_{t \in \mathcal{T}} g_j^t(x^t) + \sum_{k \in \mathcal{L}} \lambda_k \sum_{t \in \mathcal{T}} h_k^t(x^t). \quad (6.2)$$

The corresponding Lagrangian dual function $q(\mu, \lambda)$ is

$$q(\mu, \lambda) := \min_{x^1, \dots, x^T} (L(x^1, \dots, x^T, \mu, \lambda) \mid x^t \in \mathcal{C}^t, t \in \mathcal{T}).$$

Note that for any Lagrange multiplier vector (μ, λ) , the solution to the inner optimization problem of the Lagrangian dual function $q(\mu, \lambda)$ is unique since each cost function f^t is strictly convex. We denote this unique solution by the vector $x(\mu, \lambda) := (x^t(\mu, \lambda))_{t \in \mathcal{T}}$ and call this vector the *Lagrangian solution* to $q(\mu, \lambda)$.

Throughout this chapter, we assume that Lagrange multipliers (μ^*, λ^*) exist such that the optimal solution $x(\mu^*, \lambda^*)$ to the Lagrangian dual function $q(\mu^*, \lambda^*)$ is optimal for the original primal Problem P. We call these multipliers *optimal* for Problem P. In the case of Problem P, the existence of such optimal Lagrange multipliers is equivalent to Problem P satisfying strong duality, which can be achieved through a relatively basic constraint qualification such as Slater's condition [29]. Moreover, note that the optimal multipliers (μ^*, λ^*) are unique if and only if the optimal solution to Problem P satisfies the so-called Linear Independence Constraint Qualification [173]. However, to simplify the discussion and without loss of generality, we assume in the derivation of our approach that the optimal multipliers (μ^*, λ^*) are unique.

Many solution approaches for convex optimization problems exploit strong duality, i.e., they iteratively evaluate the Lagrangian dual function for a guess of the Lagrange multipliers and update this guess based on the resulting function value. The reason for this is that often the special structure of the Lagrangian dual function can be exploited to evaluate it efficiently for given Lagrange multipliers (μ, λ) . A well-known example of this is the quadratic resource allocation problem QRA that we studied in Chapter 3.

For many optimization problems occurring in practice, the Lagrangian is separable in the primal variables. This separability allows us to decompose the optimization problem within the Lagrangian dual function into smaller subproblems whose optimal solutions together form the optimal solution to the whole problem. Since the Lagrangian $L(x^1, \dots, x^T, \mu, \lambda)$ of Problem P is indeed separable in $x^1, \dots, x^T$, we can apply this decomposition also for Problem P. More precisely, we can define for each $t \in \mathcal{T}$ the *local* Lagrangian $L^t(x^t, \mu, \lambda)$

and *local* Lagrangian dual function $q^t(x^t, \mu, \lambda)$ by:

$$L^t(x^t, \mu, \lambda) := f^t(x^t) + \sum_{j \in \mathcal{M}} \mu_j g_j^t(x^t) + \sum_{k \in \mathcal{L}} \lambda_k h_k^t(x^t)$$

and

$$q^t(\mu, \lambda) := \min_{x^t} (L(x^t, \mu, \lambda) \mid x^t \in \mathcal{C}^t).$$

Note that solving the inner problem of $q(\mu, \lambda)$ is equivalent to solving the inner problems of $q^t(\mu, \lambda)$ for each $t \in \mathcal{T}$. In other words, for each $t \in \mathcal{T}$, the solution to the inner problem of $q^t(\mu, \lambda)$ is equal to the t^{th} component of the solution to the inner problem of $q(\mu, \lambda)$, namely $x^t(\mu, \lambda)$. We call this solution the *local Lagrangian solution* corresponding to stage t .

In many applications of Lagrangian duality theory, this separability of the Lagrangian is exploited to allow for parallel computation of the local Lagrangian dual functions $q^t(\mu, \lambda)$ (see, e.g., [24]). However, we exploit this separability for a different purpose, which we explain in the next subsection.

6.3.3 SOLUTION APPROACH

The main idea behind our solution approach is as follows. Suppose we are at the start of stage $\bar{t}$ and the corresponding cost function $f^{\bar{t}}$ has been revealed. If we know the optimal Lagrange multipliers (μ^*, λ^*) , we can compute the optimal stage decision vector $(x^*)^{\bar{t}}$ by solving the inner problem of $q^{\bar{t}}(\mu^*, \lambda^*)$. This means that we can do this — and this is the key observation in this article — without any knowledge of the future cost functions $f^{\bar{t}+1}, \dots, f^T$. As a consequence, the only required data that is uncertain in the computation of the optimal solution vector $(x^*)^{\bar{t}}$ is the vector of optimal Lagrange multipliers (μ^*, λ^*) (recall, that the current cost function $f^{\bar{t}}$ has already been revealed). Thus, if we have a prediction $(\hat{\mu}, \hat{\lambda})$ for (μ^*, λ^*) , we can obtain an approximation or *online solution* $\hat{x}^{\bar{t}}$ to the optimal solution vector $(x^*)^{\bar{t}}$ by solving the inner problem of $q^{\bar{t}}(\hat{\mu}, \hat{\lambda})$.

One important question is whether the resulting online solution $\hat{x} := (\hat{x}^t)_{t \in \mathcal{T}}$ is feasible. Note that when the optimal Lagrange multipliers μ^* and λ^* are known, we have the guarantee that the solution to the inner problem of the Lagrangian dual function $q(\mu^*, \lambda^*)$ (or, equivalently, the solution obtained by solving the local Lagrangian dual functions $q^t(\mu^*, \lambda^*)$) is feasible for Constraints (6.1b) and (6.1c). This is true although these constraints are not enforced explicitly in the Lagrangian dual function. However, when we replace the optimal Lagrange multipliers by predictions $\hat{\mu}$ and $\hat{\lambda}$, i.e., when we use the solutions of the inner problems of $q(\hat{\mu}, \hat{\lambda})$, we do not have this guarantee anymore. Thus, we must find a different way to ensure that the online solution $\hat{x}^t$ for a given stage $t \in \mathcal{T}$ is chosen such that there always exists a feasible extension of this solution for the future stages.

We discuss two approaches for solving this problem. The first approach is to simply add all original constraints to the inner problem of $q^{\bar{t}}(\hat{\mu}, \hat{\lambda})$, substitute the online solution corresponding to the previous stages in $\{1, \dots, \bar{t} - 1\}$, and solve this adjusted optimization problem instead. This adjusted problem, which we denote by $(P^{\bar{t}}(\hat{\mu}, \hat{\lambda}))$, can be formulated as follows:

$$\begin{aligned} (P^{\bar{t}}(\hat{\mu}, \hat{\lambda})) : \quad & \min_{x^{\bar{t}}, \dots, x^T} f^{\bar{t}}(x^{\bar{t}}) + \sum_{j \in \mathcal{M}} \hat{\mu}_j g_j^{\bar{t}}(x^{\bar{t}}) + \sum_{k \in \mathcal{L}} \hat{\lambda}_k h_k^{\bar{t}}(x^{\bar{t}}) \\ \text{s.t.} \quad & \sum_{t=1}^{\bar{t}-1} g_j^t(\hat{x}^t) + \sum_{t=\bar{t}}^T g_j^t(x^t) \leq 0, \quad j \in \mathcal{M}, \\ & \sum_{t=1}^{\bar{t}-1} h_k^t(\hat{x}^t) + \sum_{t=\bar{t}}^T h_k^t(x^t) = 0, \quad k \in \mathcal{L}, \\ & x^t \in \mathcal{C}^t, \quad t \in \{\bar{t}, \dots, T\}. \end{aligned}$$

Observe that this problem is an instance of Problem P with horizon $\{\bar{t}, \dots, T\}$ and without the unknown cost functions f^t for $t > \bar{t}$. This suggests that solving this adjusted problem is as easy as solving the deterministic version of the original Problem P. In fact, one could even use the same, possibly tailored, solution methodology, provided this methodology does not require all cost functions to be *strictly* convex.

A second approach, which leads to the same solution but is sometimes easier to execute, is to construct the projection of the feasible set onto the current decision vector and adjust the set $\mathcal{C}^{\bar{t}}$ accordingly. This means that we construct an alternative feasible set $\mathcal{C}_{\text{proj}}^{\bar{t}} \subset \mathbb{R}^{N^{\bar{t}}}$ for $x^{\bar{t}}$ such that choosing $\hat{x}^{\bar{t}}$ from this set ensures that there still exists a feasible solution for the future stages. More precisely, we solve the following problem:

$$\begin{aligned} (P_{\text{proj}}^{\bar{t}}(\hat{\mu}, \hat{\lambda})) : \quad & \min_{x^{\bar{t}}} f^{\bar{t}}(x^{\bar{t}}) + \sum_{j \in \mathcal{M}} \hat{\mu}_j g_j^{\bar{t}}(x^{\bar{t}}) + \sum_{k \in \mathcal{L}} \hat{\lambda}_k h_k^{\bar{t}}(x^{\bar{t}}) \\ \text{s.t.} \quad & x^{\bar{t}} \in \mathcal{C}_{\text{proj}}^{\bar{t}}, \end{aligned}$$

where

$$\mathcal{C}_{\text{proj}}^{\bar{t}} := \{x^{\bar{t}} \mid \exists (x^{\bar{t}+1}, \dots, x^T) \text{ s.t. } (\hat{x}^1, \dots, \hat{x}^{\bar{t}-1}, x^{\bar{t}}, \dots, x^T) \text{ is feasible for P}\}. \quad (6.3)$$

Note that this projection is not the same as the projection operation that is commonly used in OCO. The latter operation is used to project a candidate solution onto a feasible set (see also, e.g., [69]), whereas here we project a feasible set onto a decision vector.

The second approach is worth considering over the first approach if constructing the projection $\mathcal{C}_{\text{proj}}^{\bar{t}}$ is relatively easy. An example of when this is the case is when

all constraints (6.1b) and (6.1c) are affine and the sets $\mathcal{C}^t$ are polytopes. In this case, the feasible set of Problem P is given by a set of linear inequalities and, as a consequence, one can use a method such as Fourier-Motzkin elimination (FME) (see, e.g., [27]) to compute the projection $\mathcal{C}_{\text{proj}}^t$. Although FME has an exponential worst-time complexity, several improvements to the original FME algorithm can be made to make it fast in practice (see, e.g., [18]). In addition, specific problem structures of the considered instance of Problem P can be exploited to increase the efficiency of FME. In Section 6.5.1, we show that this is indeed possible for a particular battery charging scheduling problem.

Summarizing, using the ODDO framework we can solve the online version of Problem P as follows. First, we compute a prediction $(\hat{\mu}, \hat{\lambda})$ of the optimal Lagrange multipliers (μ^*, λ^*) (we discuss several approaches to make such a prediction in Section 6.5.3). Subsequently, at the start of each stage $\bar{t} \in \mathcal{T}$, we compute the corresponding online solution vector $\hat{x}^{\bar{t}}$ in one of the following two ways:

- » If the alternative feasible set $\mathcal{C}_{\text{proj}}^{\bar{t}}$ can be computed easily, e.g., by means of FME, then we solve Problem $(P_{\text{proj}}^{\bar{t}}(\hat{\mu}, \hat{\lambda}))$ using this set as input and choose $\hat{x}^{\bar{t}}$ as the solution to this problem.
- » Otherwise, we solve Problem $(P^{\bar{t}}(\hat{\mu}, \hat{\lambda}))$, i.e., we enforce all original constraints into the formulation of the inner problem of the Lagrangian dual function $q^{\bar{t}}(\hat{\mu}, \hat{\lambda})$, and take the resulting solution vector corresponding to stage $\bar{t}$ as the online solution $\hat{x}^{\bar{t}}$.

Algorithm 6.1 summarizes the ODDO framework as presented in this section. The check in Line 4 has been included mainly for the sake of completeness: when implementing the algorithm for a specific instance of Problem P one can most likely determine on forehand whether or not the projection $\mathcal{C}_{\text{proj}}^t$ is easy to compute and directly go to Line 5 or Line 7.

6.3.4 AN ILLUSTRATIVE EXAMPLE

As an illustrative example of the ODDO framework, we consider the following example problem, which is an instance of Problem QRA:

$$\text{E : } \min_{x^1, x^2, x^3} \sum_{t=1}^3 (x^t)^2 + Y^t x^t \quad (6.4a)$$

$$\text{s.t. } x^1 + x^2 + x^3 = 10, \quad (6.4b)$$

$$0 \leq x^t \leq 6, \quad t \in \{1, 2, 3\}. \quad (6.4c)$$

Since all stage vectors x^t are one-dimensional, we omit the subscript index, meaning that in this example $x^t \in \mathbb{R}$ for all $t \in \mathcal{T}$. In this problem, the parameters Y^1 , Y^2 , and Y^3 are uncertain and are revealed at the start of stages 1, 2, and 3

Algorithm 6.1: Solving the online version of Problem P using the ODDO framework.

```

1 Input: prediction  $(\hat{\mu}, \hat{\lambda})$  of optimal Lagrange multipliers
2 Output: online solution  $\hat{x} := (\hat{x}^1, \dots, \hat{x}^T)$ 
3 for  $\bar{t} = 1, \dots, T$  do
4   if Projection  $C_{\text{proj}}^{\bar{t}}$  can be computed easily then
5     | Compute  $C_{\text{proj}}^{\bar{t}}$  and obtain solution  $\tilde{x}^{\bar{t}}$  to Problem  $(P_{\text{proj}}^{\bar{t}}(\hat{\mu}, \hat{\lambda}))$ 
6   else
7     | Obtain solution  $(\bar{x}^{\bar{t}}, \dots, \bar{x}^T)$  to Problem  $(P^{\bar{t}}(\hat{\mu}, \hat{\lambda}))$ 
8    $\hat{x}^{\bar{t}} := \tilde{x}^{\bar{t}}$ 
9 return  $\hat{x}$ 

```

respectively. Furthermore, there is only a single Lagrange multiplier λ that corresponds to the equality constraint (6.4b). For this multiplier, we take as prediction $\hat{\lambda} = 2$.

We start the online optimization process at the beginning of stage 1. At this moment it is revealed that $Y^1 = -4$. Thus, the local Lagrangian dual function at stage 1 is

$$q_E^1(\lambda) := \min_{x^1} ((x^1)^2 + (-4 + \lambda)x^1 \mid 0 \leq x^1 \leq 6).$$

Hereby, we still need to take future feasibility into account. Due to the simple structure of Problem E, we can easily compute the projection of the feasible region onto x^1 . For this, note that we must have $\hat{x}^1 = 10 - x^2 - x^3$ for some x^2 and x^3 that satisfy their box constraints (6.4c). It follows that $-2 \leq \hat{x}^1 \leq 10$. Since these bounds are less tight than the original bounds imposed by Constraint (6.4c), we can ignore them and thus we can choose $\hat{x}^1$ as the solution to the inner problem of $q_{(E)}^1(\hat{\lambda})$. This yields $\hat{x}^1 = 1$.

Subsequently, at the beginning of stage 2, it is revealed that $Y^2 = 1$. In order to ensure future feasibility, we must choose $\hat{x}^2$ such that $\hat{x}^2 = 10 - \hat{x}^1 - x^3$ for some x^3 that satisfies the bound constraints (6.4c). It follows that we must have $3 \leq \hat{x}^2 \leq 9$. Thus, we must choose $\hat{x}^2$ as the solution to the projected problem $(E_{\text{proj}}^2(2))$, which is given by

$$\min_{x^2} (x^2)^2 + 3x^2 \quad \text{s.t. } 3 \leq x^2 \leq 6.$$

It follows that $\hat{x}^2 = 3$.

Finally, we arrive at stage 3, where we assume that the last uncertain parameter, Y^3 , is revealed to be -5 . However, its value has no influence on the online decision $\hat{x}^3$ since it is required to be $\hat{x}^3 = 10 - \hat{x}^1 - \hat{x}^2 = 6$. Note that this does

not violate the box constraints (6.4c) for x^3 . Thus, the online solution obtained by choosing $\hat{\lambda} = 2$ is $\hat{x} = (1, 3, 6)$ and its objective value is 25. The optimal offline solution, i.e., where the values for Y^t are known on forehand, is given by $x^* = (4, \frac{3}{2}, \frac{9}{2})$ with an objective value of $\frac{3}{2}$ and the corresponding optimal Lagrange multiplier is $\lambda^* = -4$.

6.4 MOTIVATION AND ROBUSTNESS OF ODDO

With regard to the ODDO framework presented in Section 6.3, two important questions arise. The first question is about the predictive value of the optimal Lagrange multipliers: what is the value of knowing the optimal Lagrange multipliers when the cost functions are different from what was expected? In other words, can we say something about the difference between the optimal Lagrange multipliers of two problem instances whose cost functions are (slightly) different? The second question is about the robustness of this approach: how accurate must the predictions of the optimal Lagrange multipliers be such that this approach yields a good approximate online solution? In other words, what can we say about the difference in objective value between the online and optimal offline solution when we use a prediction of the optimal Lagrange multipliers as input for the online approach?

For the first question, we provide in Section 6.4.1 several arguments that suggest that a small change in the cost functions will lead to only proportionally small changes or even no changes at all in the optimal Lagrange multipliers. In Section 6.4.2, we answer the second question by deriving for a special class of optimization problems under submodular constraints a bound on the difference in objective value between the online and offline optimal solution.

6.4.1 PREDICTIVE VALUE OF THE OPTIMAL LAGRANGE MULTIPLIERS

As mentioned above, to answer the first question, we argue that the effect of small changes in the cost functions on changes in the optimal Lagrange multipliers is small. Additionally, we argue that, from a practical point of view, knowledge of the optimal Lagrange multipliers of a given problem provides useful structural information on the problem and its optimal solutions.

The first argument is based on results from sensitivity analysis on parametric convex programs. For this, we assume for the moment that the uncertainty in the cost functions is due to an uncertain parameter w that resides in a given set $\mathcal{W} \subset \mathbb{R}^W$. In other words, each cost function f^t is a function of both x^t and w , i.e., $f^t(x^t) \equiv f^t(x^t, w)$. Let $\mathcal{F} \subset \mathbb{R}^N$ denote an arbitrary feasible region for Problem P. For each $w \in \mathcal{W}$, we denote by $I(w)$ the problem instance of P with feasible set $\mathcal{F}$ and cost functions $f^t(x^t, w)$, $t \in \mathcal{T}$. Moreover, let $(\mu^*(w), \lambda^*(w))$ be the optimal Lagrange multipliers for $I(w)$. It can be shown under a relatively weak constraint qualification, such as the so-called Mangasarian Fromovitz Constraint Qualification, that $(\mu^*(w), \lambda^*(w))$ is Lipschitz continuous in w (see, e.g.,

[161]). In particular, this means that there exists a positive constant K_0 such that for any $w^1, w^2 \in \mathcal{W}$ it holds that

$$\|(\mu^*(w^1), \lambda^*(w^1)) - (\mu^*(w^2), \lambda^*(w^2))\|_2 \leq K_0 \|w^1 - w^2\|_2.$$

This implies that the difference in optimal Lagrange multipliers grows only linearly in the change of the uncertainty parameter w .

The second argument is that, for a given feasible set $\mathcal{F}$, there are in general multiple instances of Problem P over $\mathcal{F}$, i.e., multiple different cost functions, whose optimal Lagrange multipliers are the same. In other words, the mapping from the set of possible cost functions to the corresponding optimal Lagrange multipliers is not necessarily one-to-one but many-to-one. This is because Constraint (6.1d) is not dualized in the formulation of the Lagrangian in Equation (6.2). As a consequence, the Lagrange multipliers that correspond to any constraints imposed by the sets $\mathcal{C}^t$ are not specified in a vector of optimal Lagrange multipliers. Thus, for two problem instances with different cost functions, their Lagrange multipliers may differ only in these non-specified Lagrange multipliers. As an example of this, we again consider Problem QRA from Chapter 3, where we adjust the formulation slightly to match it with that of Problem P:

$$\begin{aligned} \text{QRA : } \min_{x^1, \dots, x^T} \quad & \sum_{t \in \mathcal{T}} \frac{1}{2} \frac{(x^t)^2}{a^t} \\ \text{s.t. } \quad & \sum_{t \in \mathcal{T}} x^t = R, \\ & l^t \leq x^t \leq u^t, \quad t \in \mathcal{T}. \end{aligned}$$

Given a Lagrange multiplier $\lambda \in \mathbb{R}$, the Lagrangian dual function $q_{\text{QRA}}(\lambda)$ of Problem QRA is given by

$$q_{\text{QRA}}(\lambda) := \min_{x^1, \dots, x^T} \left(\sum_{t \in \mathcal{T}} \frac{1}{2} \frac{(x^t)^2}{a^t} + \lambda \left(\sum_{t \in \mathcal{T}} x^t - R \right) \right),$$

and the optimal solution $x(\lambda)$ to the inner optimization problem of $q_{\text{QRA}}(\lambda)$ is given by¹

$$x^t(\lambda) = \begin{cases} l^t & \text{if } -\lambda < \frac{l^t}{a^t}, \\ -a^t \lambda & \text{if } \frac{l^t}{a^t} \leq -\lambda < \frac{u^t}{a^t}, \\ u^t & \text{if } \frac{u^t}{a^t} \leq -\lambda. \end{cases}$$

Considering the optimal Lagrange multiplier λ^* and optimal solution $x(\lambda^*)$ of Problem QRA, observe that λ^* remains the optimal Lagrange multiplier for,

¹Note that in this formulation, the sign of the Lagrange multiplier has been "flipped" compared to the formulation of $x(\lambda)$ in Equation (3.2) in Section 3.2. This is due to the different way in which this multiplier is incorporated into the Lagrangian relaxation $q_{\text{QRA}}(\lambda)$ and into the Lagrangian dual function $q_{\text{QRA}}(\lambda)$, whether we implicitly multiply the resource constraint by -1 .

e.g., any permutation of $\mathcal{T}$. Note, however, that the Lagrange multipliers that correspond to the box constraints of Problem QRA would be different.

The two given arguments together suggest that the optimal Lagrange multipliers often may be robust against changes in the cost functions as:

- » changes in the cost functions do not always lead to a change in the optimal Lagrange multipliers;
- » even if the optimal Lagrange multipliers change, this change is bounded by the change of the uncertain data that underlies the cost functions.

The above discussion indicates that the optimal Lagrange multipliers may provide more structural information about the problem and its optimal solution than the cost functions. Besides the properties discussed in this subsection, there are other properties of the optimal Lagrange multipliers that support this claim. One of these properties is complementary slackness: if for a given inequality constraint (6.1b) with index $j \in \mathcal{M}$ the corresponding optimal Lagrange multiplier μ_j^* is nonzero, then this constraint is tight in the optimal primal solution x^* . This is not only a nice theoretical structural relation between the optimal Lagrange multipliers and the optimal primal solution, but can also have a useful physical interpretation depending on the application. Two examples of this are the two problems that we consider in the evaluation of ODDO in Section 6.5, namely battery scheduling and inventory management. In the first problem, a tight inequality constraint implies that, in the optimal solution, the battery is either completely empty or full at a given stage (see also Sections 4.4.3 and 6.5.1). In the latter problem, tight inequality constraints imply that, in the optimal solution, a given factory produces at a capacity utilization rate of 100% or that the level of stock in the warehouse has reached either its minimum or maximum allowed level at a given stage (see also Section 6.5.2). These insights can not only be used to determine the optimal operation of the battery or the inventory management systems, but also to design these systems and decide on the necessity and profitability of, e.g., investments in battery or warehouse capacity.

6.4.2 ROBUSTNESS OF ODDO AGAINST PREDICTION ERRORS

In this section, we consider the second question concerning the relation between the prediction $(\hat{\mu}, \hat{\lambda})$ of the optimal Lagrange multipliers and the difference in objective value between the online and optimal offline solution. To answer this question, we derive a bound on the difference in objective value between the online solution $\hat{x}(\hat{\mu}, \hat{\lambda})$ and the optimal offline solution for the special case of optimization problems with submodular constraints and increasing cost functions. A weakness of this bound is that it depends not only on the Lagrange multipliers but also on the uncertain cost functions. Therefore, in a next step, we refine this bound for the special case where each cost function is $(a, b, \bar{f})$ -separable, i.e., of the form $a^t \bar{f}^t(\frac{x^t}{a^t} + b^t)$ (see also Chapter 2), where $\bar{f}^t$ is a *known* increasing strictly convex differentiable function, $a := (a^t)_{t \in \mathcal{T}}$ is a *known* positive-valued

vector, and $b := (b^t)_{t \in \mathcal{T}}$ is an *uncertain* vector. As already shown in Chapter 2, in many applications, the cost functions are $(a, b, \bar{f})$ -separable. In particular, this is the case for the battery charging scheduling problem that we consider in Section 6.5.1. The resulting refined bound depends only on the Lagrange multipliers and the known functions $\bar{f}^t$ and parameter a .

We first give a precise definition of the considered problem class. For this, we require the concepts of submodular functions and base polyhedra. Recall from Section 2.2.2 that a set function $r : 2^{\mathcal{T}} \rightarrow \mathbb{R}$ is said to be *submodular* if we have for any subsets $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{T}$ that $r(\mathcal{X} \cup \mathcal{Y}) + r(\mathcal{X} \cap \mathcal{Y}) \leq r(\mathcal{X}) + r(\mathcal{Y})$. Hereby, we assume without loss of generality that $r(\emptyset) = 0$. The *base polyhedron* associated with a submodular set function r is defined as

$$\mathcal{B}(r) := \left\{ x \in \mathbb{R}^T \mid \sum_{t \in \mathcal{X}} x^t \leq r(\mathcal{X}) \forall \mathcal{X} \subset \mathcal{T}, \sum_{t \in \mathcal{T}} x^t = r(\mathcal{T}) \right\}.$$

A well-known property of base polyhedra is that for any vector $x \in \mathcal{B}(r)$ the set of tight inequalities that define $\mathcal{B}(r)$ is closed under union and intersection:

Lemma 6.1. *If for a submodular function $r : 2^{\mathcal{T}} \rightarrow \mathbb{R}$, vector $x \in \mathcal{B}(r)$, and two sets $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{T}$ we have that $\sum_{t \in \mathcal{X}} x^t = r(\mathcal{X})$ and $\sum_{t \in \mathcal{Y}} x^t = r(\mathcal{Y})$, then we also have that $\sum_{t \in \mathcal{X} \cup \mathcal{Y}} x^t = r(\mathcal{X} \cup \mathcal{Y})$ and $\sum_{t \in \mathcal{X} \cap \mathcal{Y}} x^t = r(\mathcal{X} \cap \mathcal{Y})$.*

Proof. Since $\sum_{t \in \mathcal{X}} x^t = r(\mathcal{X})$ and $\sum_{t \in \mathcal{Y}} x^t = r(\mathcal{Y})$, it follows from the submodularity of r that

$$\begin{aligned} \sum_{t \in \mathcal{X}} x^t + \sum_{t \in \mathcal{Y}} x^t &= r(\mathcal{X}) + r(\mathcal{Y}) \geq r(\mathcal{X} \cup \mathcal{Y}) + r(\mathcal{X} \cap \mathcal{Y}) \\ &\geq \sum_{t \in \mathcal{X} \cup \mathcal{Y}} x^t + \sum_{t \in \mathcal{X} \cap \mathcal{Y}} x^t = \sum_{t \in \mathcal{X}} x^t + \sum_{t \in \mathcal{Y}} x^t. \end{aligned}$$

Thus, $\sum_{t \in \mathcal{X} \cup \mathcal{Y}} x^t + \sum_{t \in \mathcal{X} \cap \mathcal{Y}} x^t = r(\mathcal{X} \cup \mathcal{Y}) + r(\mathcal{X} \cap \mathcal{Y})$, which directly implies the result of the lemma. $\square$

For a thorough treatment of submodular functions and base polyhedra, we refer to [49].

We now define the class I_{sub} of problem instances that we consider in this section:

Definition 6.1. *The class I_{sub} consists of all problem instances of the form*

$$\begin{aligned} \min_{x^1, \dots, x^T} \quad & \sum_{t \in \mathcal{T}} f^t(x^t) \\ \text{s.t.} \quad & (x^1, \dots, x^T) \in \mathcal{B}(r), \end{aligned}$$

where for all $t \in \mathcal{T}$ the vector x^t is one-dimensional (i.e., $N^t = 1$), the function f^t is increasing, strictly convex, and continuously differentiable, and r is a submodular set function on $\mathcal{T}$.

Note that each instance in I_{sub} is also an instance of Problem P by choosing

$$\begin{aligned}\mathcal{M} &:= \{\mathcal{X} \mid \mathcal{X} \in 2^{\mathcal{T}}, 1 < |\mathcal{X}| < T\}, \\ \mathcal{L} &:= \{\mathcal{T}\}, \\ \mathcal{C}^t &:= [r(\mathcal{T}) - r(\mathcal{T} \setminus \{t\}), r(\{t\})], \\ g_{\mathcal{X}}^t(x^t) &:= \begin{cases} x^t - \frac{r(\mathcal{X})}{|\mathcal{X}|} & \text{if } t \in \mathcal{X}, \\ 0 & \text{otherwise,} \end{cases} \\ h_{\mathcal{T}}^t(x^t) &:= x^t - \frac{r(\mathcal{T})}{T}, \quad t \in \mathcal{T}.\end{aligned}$$

Thus, for a given $t \in \mathcal{T}$, we can write the local Lagrangian dual function $q_{\text{sub}}^t(\mu, \lambda)$ as

$$q_{\text{sub}}^t(\mu, \lambda) := \min_{x^t} \left(\begin{array}{c} f^t(x^t) + (\sum_{\mathcal{X} \in \mathcal{M}: \mathcal{X} \ni t} \mu_{\mathcal{X}} + \lambda_{\mathcal{T}}) x^t \\ r(\mathcal{T}) - r(\mathcal{T} \setminus \{t\}) \leq x^t \leq r(\{t\}) \end{array} \right),$$

and, by letting $\nu^t(\mu, \lambda) := -\sum_{\mathcal{X} \in \mathcal{M}: \mathcal{X} \ni t} \mu_{\mathcal{X}} - \lambda_{\mathcal{T}}$, the local Lagrangian solution is given by

$$x^t(\mu, \lambda) := \begin{cases} r(\mathcal{T}) - r(\mathcal{T} \setminus \{t\}) & \text{if } (\nabla f^t)^{-1}(\nu^t(\mu, \lambda)) < r(\mathcal{T}) - r(\mathcal{T} \setminus \{t\}), \\ (\nabla f^t)^{-1}(\nu^t(\mu, \lambda)) & \text{if } r(\mathcal{T}) - r(\mathcal{T} \setminus \{t\}) \\ & \leq (\nabla f^t)^{-1}(\nu^t(\mu, \lambda)) \leq r(\{t\}), \\ r(\{t\}) & \text{if } (\nabla f^t)^{-1}(\nu^t(\mu, \lambda)) > r(\{t\}). \end{cases} \quad (6.5)$$

For this specific instance class I_{sub} , we can bound the difference in objective value between the online and optimal offline solution when the multiplier prediction $(\hat{\mu}, \hat{\lambda})$ is an “under-prediction” of the optimal Lagrange multipliers (μ^*, λ^*) , i.e., the inequality $(\hat{\mu}, \hat{\lambda}) \leq (\mu^*, \lambda^*)$ holds component-wise. Apart from the predicted and optimal Lagrange multipliers $(\hat{\mu}, \hat{\lambda})$ and (μ^*, λ^*) , this bound depends on the cost functions f^t and the inverses of their gradients $(\nabla f^t)^{-1}$. More precisely, the bound is the difference in objective value between the (not necessarily feasible) solutions $((\nabla f^t)^{-1}(\nu^t(\hat{\mu}, \hat{\lambda})))_{t \in \mathcal{T}}$ and $((\nabla f^t)^{-1}(\nu^t(\mu^*, \lambda^*)))_{t \in \mathcal{T}}$. These are closely related to the Lagrangian solutions $x(\hat{\mu}, \hat{\lambda})$ and $x(\mu^*, \lambda^*)$ as given by Equation (6.5). We state this bound by means of Theorem 6.1.

Theorem 6.1. *For any problem instance in I_{sub} and a given prediction $(\hat{\mu}, \hat{\lambda})$ of the optimal Lagrange multipliers (μ^*, λ^*) such that $(\hat{\mu}, \hat{\lambda}) \leq (\mu^*, \lambda^*)$, it holds that*

$$\begin{aligned}& \sum_{t \in \mathcal{T}} f^t(\hat{x}^t(\hat{\mu}, \hat{\lambda})) - \sum_{t \in \mathcal{T}} f^t(\hat{x}^t(\mu^*, \lambda^*)) \\ & \leq \sum_{t \in \mathcal{T}} f^t((\nabla f^t)^{-1}(\nu^t(\hat{\mu}, \hat{\lambda}))) - \sum_{t \in \mathcal{T}} f^t((\nabla f^t)^{-1}(\nu^t(\mu^*, \lambda^*))).\end{aligned}$$

We prove Theorem 6.1 using Lemmas 6.2-6.4. Lemma 6.2 states that the local Lagrangian solution $x(\mu, \lambda)$ is non-increasing in (μ, λ) . This implies that the behavior of the initial online solution $x^t(\mu, \lambda)$ can, to some extent, be controlled via the predicted Lagrange multipliers. Lemma 6.3 implies that whenever the local Lagrangian solution $x^t(\hat{\mu}, \hat{\lambda})$ needs to be reduced in order to preserve future feasibility, at least one constraint involving the current stage t but no future stages is tight in the online solution. This means that this constraint is fulfilled “earlier” than it would have been when using the optimal Lagrange multipliers as input. This suggests that the constraint is (nearly) tight in the optimal solution. Finally, Lemma 6.4 uses Lemmas 6.2 and 6.3 to prove that, in the case of an under-prediction of the Lagrange-multipliers, the cost of an online solution $\hat{x}^t(\mu, \lambda)$ will always be less than or equal to the cost of the corresponding local Lagrangian solution $x^t(\mu, \lambda)$, even when adjustments have been made in order to maintain feasibility.

Lemma 6.2. *For any instance in I_{sub} , the local Lagrangian solutions $x^t(\mu, \lambda)$ are nonincreasing in (μ, λ) , i.e., for any two vectors of Lagrange multipliers $(\underline{\mu}, \underline{\lambda})$ and $(\bar{\mu}, \bar{\lambda})$ such that $(\underline{\mu}, \underline{\lambda}) \leq (\bar{\mu}, \bar{\lambda})$, it holds that $x^t(\underline{\mu}, \underline{\lambda}) \geq x^t(\bar{\mu}, \bar{\lambda})$ for all $t \in \mathcal{T}$.*

Proof. Since f^t is strictly convex and continuously differentiable, its gradient ∇f^t is continuous and monotonically increasing. It follows from the inverse function theorem that also the inverse $(\nabla f^t)^{-1}$ is continuous and monotonically increasing (see, e.g., [159]). Thus, for any two vectors of multipliers $(\underline{\mu}, \underline{\lambda})$ and $(\bar{\mu}, \bar{\lambda})$ such that $(\underline{\mu}, \underline{\lambda}) \leq (\bar{\mu}, \bar{\lambda})$, it holds that

$$\begin{aligned} (\nabla f^t)^{-1}(v^t(\underline{\mu}, \underline{\lambda})) &= (\nabla f^t)^{-1} \left(- \sum_{\mathcal{X} \in \mathcal{M}: \mathcal{X} \ni t} \underline{\mu}_{\mathcal{X}} - \underline{\lambda}_{\mathcal{T}} \right) \\ &\geq (\nabla f^t)^{-1} \left(- \sum_{\mathcal{X} \in \mathcal{M}: \mathcal{X} \ni t} \bar{\mu}_{\mathcal{X}} - \bar{\lambda}_{\mathcal{T}} \right) \\ &= (\nabla f^t)^{-1}(v^t(\bar{\mu}, \bar{\lambda})). \end{aligned}$$

Equation (6.5) for the local Lagrangian solution $x^t(\mu, \lambda)$ implies that $x^t(\mu, \lambda)$ can be seen as a piecewise nondecreasing function of $(\nabla f^t)^{-1}(v^t(\mu, \lambda))$. Thus, it follows that $x^t(\underline{\mu}, \underline{\lambda}) \geq x^t(\bar{\mu}, \bar{\lambda})$, which proves the lemma. $\square$

Lemma 6.3. *For any instance in I_{sub} , a given $\bar{t} \in \mathcal{T}$, and given multipliers (μ, λ) and corresponding online solution $\hat{x}^t(\mu, \lambda)$ and local Lagrangian solution $x^{\bar{t}}(\mu, \lambda)$, we have:*

$$\begin{aligned} &\text{if } \hat{x}^{\bar{t}}(\mu, \lambda) < x^{\bar{t}}(\mu, \lambda), \text{ we have:} \\ &\exists \bar{\mathcal{S}}^{\bar{t}} \subseteq \{1, \dots, \bar{t}\} \text{ with } \bar{\mathcal{S}}^{\bar{t}} \ni \bar{t} \text{ such that } \sum_{t \in \bar{\mathcal{S}}^{\bar{t}}} \hat{x}^t(\mu, \lambda) = r(\bar{\mathcal{S}}^{\bar{t}}). \end{aligned}$$

Proof. See Appendix A.3.1. □

Lemma 6.4. *For any instance in I_{sub} and given any Lagrange multipliers (μ, λ) such that $(\mu, \lambda) \leq (\mu^*, \lambda^*)$, it holds that $\hat{x}^t(\mu, \lambda) \leq x^t(\mu^*, \lambda^*)$ for all $t \in \mathcal{T}$.*

Proof. See Appendix A.3.2. □

Using this last lemma, we now prove Theorem 6.1:

Proof of Theorem 6.1. We prove the theorem by showing that the inequality stated in the theorem holds for each individual $t \in \mathcal{T}$, i.e., $f^t(\hat{x}^t(\hat{\mu}, \hat{\lambda})) - f^t(x^t(\mu^*, \lambda^*)) \leq f^t((\nabla f^t)^{-1}(\nu^t(\hat{\mu}, \hat{\lambda}))) - f^t((\nabla f^t)^{-1}(\nu^t(\mu^*, \lambda^*)))$. The result then follows by summing this inequality over all t .

Without loss of generality, we may assume that for the local Lagrangian solutions we have that $x^t(\hat{\mu}, \hat{\lambda}) \neq x^t(\mu^*, \lambda^*)$ and that $\nu^t(\hat{\mu}, \hat{\lambda}) \neq \nu^t(\mu^*, \lambda^*)$. Thus, by Lemma 6.4 and the fact that $v^t(\mu, \lambda)$ is non-increasing in (μ, λ) , it follows that $x^t(\hat{\mu}, \hat{\lambda}) < x^t(\mu^*, \lambda^*)$ and $\nu^t(\hat{\mu}, \hat{\lambda}) > \nu^t(\mu^*, \lambda^*)$ respectively. By the definition of the local Lagrangian solution $x^t(\mu, \lambda)$ in Equation (6.5), this implies that $x^t(\hat{\mu}, \hat{\lambda}) \leq (\nabla f^t)^{-1}(v(\hat{\mu}, \hat{\lambda}))$ and $x^t(\mu^*, \lambda^*) \geq (\nabla f^t)^{-1}(v(\mu^*, \lambda^*))$. Thus, for each $t \in \mathcal{T}$, it follows by Lemma 6.4 and the fact that f^t is increasing that

$$\begin{aligned} f^t(\hat{x}^t(\hat{\mu}, \hat{\lambda})) - f^t(x^t(\mu^*, \lambda^*)) &\leq f^t(x^t(\hat{\mu}, \hat{\lambda})) - f^t(x^t(\mu^*, \lambda^*)) \\ &\leq f^t((\nabla f^t)^{-1}(\nu^t(\hat{\mu}, \hat{\lambda}))) \\ &\quad - f^t((\nabla f^t)^{-1}(\nu^t(\mu^*, \lambda^*))). \end{aligned}$$

□

Theorem 6.1 provides us with a bound on the difference in objective value between the online and optimal offline solution to Problem P. Note, however, that the bound in Theorem 6.1 depends on the uncertain cost functions f^t and the inverses of their gradients $(\nabla f^t)^{-1}$. Therefore, in the following, for a particular type of cost functions, we refine the bound in Theorem 6.1 so that it depends only on the Lagrange multipliers and some other known structures. For this, suppose that each cost function f^t is of the form $a^t \bar{f}^t(\frac{x^t}{a^t} + b^t)$, where the increasing strictly convex differentiable function $\bar{f}^t$ and the parameter $a^t \in \mathbb{R}_{>0}$ are known but the parameter $b^t \in \mathbb{R}$ is uncertain. We show in Corollary 6.1 that for instances in I_{sub} whose cost functions have this structure, the bound in Theorem 6.1 does not depend on any uncertain functions or parameters. Subsequently, as an example of how the bound can be simplified further for specific choices of $\bar{f}^t$, we apply this result to the cases where each function $\bar{f}^t$ is a power function or an exponential function in Corollaries 6.2 and 6.3 respectively. The former case applies to the battery charging problem considered in Section 6.5.1.

Corollary 6.1. For $f^t(x^t) := a^t \bar{f}^t(\frac{x^t}{a^t} + b^t)$ with a given increasing strictly convex differentiable function $\bar{f}^t$, a known parameter $a^t \in \mathbb{R}_{>0}$, and an uncertain parameter $b^t \in \mathbb{R}$ for all $t \in \mathcal{T}$, the bound in Theorem 6.1 becomes

$$\sum_{t \in \mathcal{T}} a^t \bar{f}^t((\nabla \bar{f}^t)^{-1}(\nu^t(\hat{\mu}, \hat{\lambda}))) - \sum_{t \in \mathcal{T}} a^t \bar{f}^t((\nabla \bar{f}^t)^{-1}(\nu^t(\mu^*, \lambda^*))).$$

Proof. Since $\nabla f^t(x^t) = \nabla \bar{f}^t(\frac{x^t}{a^t} + b^t)$ and for any δ in the range of ∇f^t it holds that $(\nabla f^t)^{-1}(\delta) = a^t((\nabla \bar{f}^t)^{-1}(\delta) - b^t)$, we have

$$f^t((\nabla f^t)^{-1}(\delta)) = f^t(a^t((\nabla \bar{f}^t)^{-1}(\delta) - b^t)) = a^t \bar{f}^t((\nabla \bar{f}^t)^{-1}(\delta)).$$

The result follows by substituting this expression in the bound of Theorem 6.1 for $\delta \in \{v^t(\hat{\mu}, \hat{\lambda}), v^t(\mu^*, \lambda^*)\}$. $\square$

Corollary 6.2. If for all $t \in \mathcal{T}$ we have $\bar{f}^t(y) = Ky^c$ with $K \in \mathbb{R}_{>0}$ and $c \geq 1$, the bound in Theorem 6.1 becomes

$$K \left(\frac{1}{cK} \right)^{\frac{c}{c-1}} \left(\sum_{t \in \mathcal{T}} a^t (\nu^t(\hat{\mu}, \hat{\lambda}))^{\frac{c}{c-1}} - \sum_{t \in \mathcal{T}} a^t (\nu^t(\mu^*, \lambda^*))^{\frac{c}{c-1}} \right).$$

Proof. The result follows by substituting $(\nabla \bar{f}^t)^{-1}(\delta) = (\frac{\delta}{cK})^{\frac{1}{c-1}}$ for $\delta \in \{v^t(\hat{\mu}, \hat{\lambda}), v^t(\mu^*, \lambda^*)\}$ in the refined bound of Corollary 6.1. $\square$

Corollary 6.3. If for all $t \in \mathcal{T}$ we have $\bar{f}^t(y) = Ke^y$ with $K \in \mathbb{R}_{>0}$, the bound in Theorem 6.1 becomes

$$\sum_{t \in \mathcal{T}} a^t \nu^t(\hat{\mu}, \hat{\lambda}) - \sum_{t \in \mathcal{T}} a^t \nu^t(\mu^*, \lambda^*)$$

Proof. Analogously to the proof of Corollary 6.2, the result follows by substituting $(\nabla \bar{f}^t)^{-1}(\delta) = \ln \frac{\delta}{K}$ for $\delta \in \{v^t(\hat{\mu}, \hat{\lambda}), v^t(\mu^*, \lambda^*)\}$ in the refined bound of Corollary 6.1. $\square$

An alternative bound can be deduced using the fact that both f^t as a function of x^t and $x^t(\mu, \lambda)$ as a function of (μ, λ) are Lipschitz continuous (see, e.g., [145, 155]). This yields a bound of the form $K\|(\hat{\mu}, \hat{\lambda}) - (\mu^*, \lambda^*)\|_2$ where K is a constant that depends on the Lipschitz constants of the functions f^t and $x^t(\mu, \lambda)$. This bound suggests that the difference in objective values grows linearly in the 2-norm of the difference between the predicted and optimal Lagrange multipliers. However, this bound depends on the uncertain cost functions via the constant K . Moreover, the linear behavior of the bound is inconsistent with the nature of

the gradients for cost functions that are not quadratic. In contrast, our bounds in Theorem 6.1 and Corollaries 6.1-6.3 actively incorporate the (gradients of) the specific cost function structure itself.

We conclude this section with a note on whether the result of Theorem 6.1 could also be extended to instances that do not fall within the class I_{sub} . We observe that Lemma 6.2 can be extended to broader classes of problems with, e.g., multi-dimensional stage vectors, nonlinear and non-submodular inequality constraints, and nonseparable objectives. For instance, in a sense it is analogous to the monotonicity result for Problem QRA-NonSep-GBC in Lemma 5.4, which has a nonseparable objective function. This leads us to the question of whether one can also extend Lemmas 6.3 and 6.4 to broader classes of problems. Since the proofs of Lemmas 6.3 and 6.4 rely heavily on the submodular constraint structure (as opposed to the proof of Lemma 6.2), it seems unlikely that these lemmas can be extended to problem instances with non-submodular constraints. However, the incorporation of other extensions such as multi-dimensional stage vectors and nonseparable objectives is not limited by the submodular constraint structure and it may thus be very well possible to extend the result to such problem classes.

6.5 EVALUATION

In this section, we evaluate the performance of the ODDO framework on two types of optimization problems. We introduce and formulate these problems in Sections 6.5.1 and 6.5.2 respectively. Subsequently, in Section 6.5.3, we explain the goal and setup of our evaluation, including our approach for predicting the optimal Lagrange multipliers. Finally, in Section 6.5.4, we present and discuss the results of our evaluations.

6.5.1 BATTERY SCHEDULING

The first considered problem is the battery scheduling problem BATTERY introduced in Section 2.5.2. We choose as objective for this problem load profile flattening, meaning that the cost functions take the form $f^t(x^t) = (x^t + p^t)^2$ for each $t \in \mathcal{T}$. The resulting problem is the following instance of Problem P:

$$\text{BATTERY : } \min_{x^1, \dots, x^T} \sum_{t \in \mathcal{T}} (x^t + p^t)^2 \quad (6.6a)$$

$$\text{s.t. } 0 \leq S_{\text{start}} + \Delta t \sum_{s=1}^t x^s \leq D, \quad t \in \mathcal{T} \setminus \{T\}, \quad (6.6b)$$

$$S_{\text{start}} + \Delta t \sum_{t \in \mathcal{T}} x^t = S_{\text{end}}, \quad (6.6c)$$

$$X_{\min} \leq x^t \leq X_{\max}, \quad t \in \mathcal{T}. \quad (6.6d)$$

We note that Problem BATTERY belongs to the problem class I_{sub} defined in Section 6.4.2 when $p^t + X_{\min} \geq 0$ for each $t \in \mathcal{T}$ (see Appendix 2.7). Also, note that the bound given in Corollary 6.2 applies to this problem by choosing $K = 1$, $c = 2$, $a^t = 1$, and $b^t = p^t$ for all $t \in \mathcal{T}$.

For each $t \in \mathcal{T} \setminus \{T\}$, let μ_t^- and μ_t^+ denote the Lagrange multipliers for the lower bound and upper bound constraint in (6.6b) respectively. The Lagrangian dual function $q_{\text{BATTERY}}(\mu, \lambda)$ of Problem BATTERY can be written as

$$q_{\text{BATTERY}}(\mu, \lambda) := \min_{x^1, \dots, x^T} \left(\begin{array}{l} \sum_{t \in \mathcal{T}} (x^t + p^t)^2 + \sum_{t=1}^{T-1} \mu_t^+ \sum_{s=1}^t x^s \\ - \sum_{t=1}^{T-1} \mu_t^- \sum_{s=1}^t x^s + \lambda \sum_{t \in \mathcal{T}} x^t \\ \text{s.t. } X_{\min} \leq x^t \leq X_{\max}, \quad t \in \mathcal{T} \end{array} \right).$$

The corresponding local Lagrangian dual function $q_{\text{BATTERY}}^t(\mu, \lambda)$ for $t \in \mathcal{T} \setminus \{T\}$ is given by

$$q_{\text{BATTERY}}^t(\mu, \lambda) := \min_{x^t} \left(\begin{array}{l} (x^t + p^t)^2 + \sum_{s=t}^{T-1} (\mu_s^+ - \mu_s^-) x^t + \lambda x^t \\ \text{s.t. } X_{\min} \leq x^t \leq X_{\max} \end{array} \right)$$

and for $t = T$ by

$$q_{\text{BATTERY}}^T(\mu, \lambda) := \min_{x^T} ((x^T + p^T)^2 + \lambda x^T \mid X_{\min} \leq x^T \leq X_{\max}).$$

We mentioned in Section 6.3.3 that, for some cases, the structure of the constraints can be exploited to compute the projection set C_{proj}^t efficiently. Problem BATTERY is such a case where this can be done. More precisely, applying FME to these constraints allows to compute the projection for all variables in $O(T)$ arithmetical operations that can be done at the start of the first time interval (see Appendix 6.7). As a consequence, we do not need to solve at the start of each time interval an entire instance of Problem BATTERY but can instead use the pre-calculated sets C_{proj}^t . This saves a lot of computation time, which is crucial for using the algorithm on embedded systems in energy management applications with low computational power.

In our evaluation, we consider the problem of scheduling the battery over one day divided into 15-minute time intervals (i.e., $T = 96$). This interval length is chosen with regard to contract durations within several European energy markets (see, e.g., [113]). Furthermore, we assume that the initial and desired SoC is set to 50% of the capacity. For the uncertain consumption p , we use real consumption data of 72 households that were obtained in a field test in the Dutch town of Heeten [142]. We base the battery parameters on the setting in the same field test. More precisely, we choose $S_{\text{start}} = S_{\text{end}} = 5.89 \cdot 10^3$, $D = 1.18 \cdot 10^4$, $\Delta t = \frac{1}{4}$, $X_{\min} = -8.67 \cdot 10^3$, and $X_{\max} = 8.67 \cdot 10^3$. We note that, given the used consumption profile p , the resulting problem instance satisfies the assumptions of Theorem 6.1.

As the second problem, we consider a basic inventory management problem that was considered in [21] to evaluate the performance of adjustable RO for linear programs. This model is a single product inventory system that consists of a warehouse and multiple factories. The goal is to minimize the production cost for all factories over a time horizon divided into stages while satisfying the required demand of the product for each stage.

We denote the set of factories by $\mathcal{N}$ and for each stage $t \in \mathcal{T}$ and factory $i \in \mathcal{N}$, we have to decide on the amount x_i^t of (divisible) product that factory i produces during stage t . Moreover, we denote by c_i^t the cost of producing one unit of the product at factory i during stage t . For each stage $t \in \mathcal{T}$, there is a demand d^t for the product that must be satisfied and at the start of the first stage, an amount of S is already present in the warehouse. Furthermore, a minimum required level of stock in the warehouse L and the maximum capacity U are given. Finally, each factory $i \in \mathcal{N}$ has a maximum production capacity of u_i^t for stage $t \in \mathcal{T}$ and a maximum total production capacity of C_i over the entire horizon. Summarizing, we get the following optimization problem:

$$\begin{aligned} \text{IM : } \min_{x^1, \dots, x^T} \quad & \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{N}} c_i^t x_i^t \\ \text{s.t. } \quad & \sum_{t \in \mathcal{T}} x_i^t \leq C_i, \quad i \in \mathcal{N}, \end{aligned} \quad (6.7a)$$

$$L \leq S + \sum_{s=1}^t \sum_{i \in \mathcal{N}} x_i^s - \sum_{s=1}^t d^s \leq U, \quad t \in \mathcal{T}, \quad (6.7b)$$

$$0 \leq x_i^t \leq u_i^t, \quad t \in \mathcal{T}, i \in \mathcal{N}.$$

In [21], the demand was considered as the uncertain parameter. Here, we instead assume the demand to be certain but the production costs to be uncertain.

Let $\mu' := (\mu'_i)_{i \in \mathcal{N}}$, $\mu^- := (\mu_t^-)_{t \in \mathcal{T}}$, and $\mu^+ := (\mu_t^+)_{t \in \mathcal{T}}$ denote the Lagrange multipliers corresponding to Constraints (6.7a) and the lower and upper bound Constraints (6.7b) respectively. The Lagrangian dual function $q_{\text{IM}}(\mu)$ of Problem IM can be written as

$$q_{\text{IM}}(\mu) := \min_{x^1, \dots, x^T} \left(\begin{array}{l} \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{N}} c_i^t x_i^t + \sum_{i \in \mathcal{N}} \mu'_i \sum_{t \in \mathcal{T}} x_i^t \\ + \sum_{t \in \mathcal{T}} \mu_t^+ \sum_{s=1}^t \sum_{i \in \mathcal{N}} x_i^s \\ - \sum_{t \in \mathcal{T}} \mu_t^- \sum_{s=1}^t \sum_{i \in \mathcal{N}} x_i^s \\ \text{s.t. } 0 \leq x_i^t \leq u_i^t, \quad t \in \mathcal{T}, i \in \mathcal{N}. \end{array} \right),$$

and the local Lagrangian dual function $q_{\text{IM}}^t(\mu)$ for stage $t \in \mathcal{T}$ is given by

$$q_{\text{IM}}^t(\mu) := \min_{x^t} \left(\sum_{i \in \mathcal{N}} \left(c_i^t + \mu'_i + \sum_{s=t}^T (\mu_s^+ - \mu_s^-) \right) x_i^t \mid 0 \leq x_i^t \leq u_i^t, i \in \mathcal{N} \right).$$

For our evaluations, we choose the same parameter values as in [21], which are given here for the sake of self-containedness. We consider a horizon of 48 weeks, where a production decision has to be made every two weeks (i.e., $T = 24$). Moreover, there are $N = 3$ factories and the seasonal demand is given by

$$d^t = 1000 \left(1 + \frac{1}{2} \sin \left(\frac{\pi(t-1)}{12} \right) \right), \quad t \in \mathcal{T}.$$

The maximum production capacity per two-week period u_i^t of each factory $i \in \mathcal{N}$ is 567 and the maximum production capacity over the entire horizon C_i is 13,600. The minimum and maximum level of inventory at the warehouse is $L = 500$ and $U = 2000$ and the initial stock level S is 500. Finally, the random production cost data c_i^t for each factory $i \in \mathcal{N}$ are drawn uniformly from the interval $[0.8\bar{c}_i^t, 1.2\bar{c}_i^t]$, where $\bar{c}_i^t$ is the expected value of the cost and given by

$$\bar{c}_i^t = \bar{E}_i \left(1 - \frac{1}{2} \sin \left(\frac{\pi(t-1)}{12} \right) \right), \quad t \in \mathcal{T},$$

where $\bar{E} := (1, 1.5, 2)$.

6.5.3 SETUP OF THE EVALUATION

For Problem BATTERY, the “training” instances for a given test instance are created by taking the household consumption data of a given number of previous days as input for the cost functions. For Problem IM, the “training” instances are created by taking as input for the uncertain parameters in the objective function randomly generated values according to the distributions specified in Section 6.5.2. In detail, for each training instance, we compute the optimal Lagrange multipliers and if for a given instance the optimal Lagrange multipliers are not unique (see Section 6.3.2), we take an arbitrary vector of optimal Lagrange multipliers. These values serve as the training data for predicting the optimal Lagrange multipliers of future instances.

We carry out two sets of evaluations. First, we study the performance of four different types of candidate multiplier vectors. For this, for a given test instance, let $\mathcal{H}$ denote the set of optimal Lagrange multipliers to the corresponding training instances. As four candidate vectors we choose the minimum, maximum, mean and median of the multipliers in $\mathcal{H}$. More precisely, for a given set $\mathcal{H}$, we compute the four candidate multiplier vectors $C^{\min} \equiv (\hat{\mu}^{\min}, \hat{\lambda}^{\min})$, $C^{\max} \equiv (\hat{\mu}^{\max}, \hat{\lambda}^{\max})$,

$C^{\text{mean}} \equiv (\hat{\mu}^{\text{mean}}, \hat{\lambda}^{\text{mean}})$, and $C^{\text{med}} \equiv (\hat{\mu}^{\text{med}}, \hat{\lambda}^{\text{med}})$ by

$$\begin{aligned} C^{\min} : \quad \hat{\mu}_j^{\min} &:= \min_{(\mu, \lambda) \in \mathcal{H}} \mu_j, & \hat{\lambda}_k^{\min} &:= \min_{(\mu, \lambda) \in \mathcal{H}} \lambda_k, \\ C^{\max} : \quad \hat{\mu}_j^{\max} &:= \max_{(\mu, \lambda) \in \mathcal{H}} \mu_j, & \hat{\lambda}_k^{\max} &:= \max_{(\mu, \lambda) \in \mathcal{H}} \lambda_k \\ C^{\text{mean}} : \quad \hat{\mu}_j^{\text{mean}} &:= \frac{1}{|\mathcal{H}|} \sum_{(\mu, \lambda) \in \mathcal{H}} \mu_j, & \hat{\lambda}_k^{\text{mean}} &:= \frac{1}{|\mathcal{H}|} \sum_{(\mu, \lambda) \in \mathcal{H}} \lambda_k, \\ C^{\text{med}} : \quad \hat{\mu}_j^{\text{med}} &:= \text{med}\{\mu_j \mid (\mu, \lambda) \in \mathcal{H}\}, & \hat{\lambda}_k^{\text{med}} &:= \text{med}\{\lambda_k \mid (\mu, \lambda) \in \mathcal{H}\}. \end{aligned}$$

Furthermore, we consider two different sizes for the set of training instances, namely 10 and 50. Thus, for each test instance, we get two values for each of the four candidate multiplier vectors: one that corresponds to $|\mathcal{H}| = 10$ and one that corresponds to $|\mathcal{H}| = 50$. Overall, we evaluate the performance of these eight candidate values for ten test instances.

In the second set of evaluations, we assess the additional value of past knowledge of the optimal Lagrange multipliers over past knowledge of the uncertain data itself. To this end, we compare ODDO to a simple strategy that resembles optimization over nominal values. In this simple strategy, we obtain the online solution to a given test instance by replacing the uncertain parameters in the cost functions by the mean of the uncertain parameters of the training instances and solving the resulting deterministic optimization problem. More precisely, for a given test instance, let $\mathcal{P}$ denote the set of cost function parameters for the training instances. Subsequently, we solve the test instance where for Problem BATTERY we take as value for the uncertain cost function parameters the values

$$\hat{p}^t := \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} p^t$$

and for Problem IM the values

$$\hat{c}_i^t := \frac{1}{|\mathcal{P}|} \sum_{c \in \mathcal{P}} c_i^t, \quad i \in \{1, 2, 3\},$$

where $c := (c_1^t, c_2^t, c_3^t)_{t \in \mathcal{T}}^\top$. We call this strategy the *nominal* strategy since it represents optimization over the nominal values of the uncertain data. Alternatively, one could see this strategy as a simplified version of model predictive control (see, e.g., [65]). We compare this nominal approach to ODDO where the prediction of the multiplier is given by the optimal Lagrange multipliers of the problem solved in the nominal strategy, i.e., of the problem instance where the uncertain parameters are substituted by the values $\hat{p} := (\hat{p}^t)_{t \in \mathcal{T}}$ or $\hat{c} := (\hat{c}_1^t, \hat{c}_2^t, \hat{c}_3^t)_{t \in \mathcal{T}}^\top$. In this comparison, we consider four different sizes for the set of training instances, namely 1, 3, 5, and 10. We carry out the comparison for 50 problem instances.

In this section, we present and discuss the results of the evaluation as described in Section 6.5.3. All simulations and computations are coded in MATLAB using CVX [63]. For each computed online solution, we calculated the ratio between the online and optimal offline objective value. This ratio serves as a measure for how good the online solution is, where a ratio of 1 implies that the online solution is optimal. We first discuss the achieved results for the comparison of the four candidate multipliers $C^{\min}$, $C^{\max}$, C^{mean} , and C^{med} and subsequently for the comparison of ODDO and the nominal strategy.

Figure 6.1 shows the observed ratios for each of the four candidate multipliers. First, we compare the performance of the candidates $C^{\min}$ and $C^{\max}$. It follows from Figures 6.1a and 6.1b that in all cases, candidate $C^{\max}$ yields the worst performance out of all candidates and its “opposite” candidate $C^{\min}$ performs significantly better than $C^{\max}$. This is in line with the robustness analysis in Section 6.4.2: when we under-predict the optimal Lagrange multipliers, the difference between the online and optimal offline objective value is bounded. In fact, the better performance of $C^{\min}$ for Problem IM as depicted in Figure 6.1b suggests that the robustness result of Theorem 6.1 might apply to a broader class of instances than the ones specified in the class I_{sub} since Problem IM does not belong to this class.

For both problem types BATTERY and IM, at least one candidate multiplier has been able to compute an online solution whose objective value is at most 1% worse than the optimal objective value ($C^{\min}$ for Problem IM and C^{mean} and C^{med} for both problems). Moreover, candidate multipliers C^{mean} and C^{med} have been able to compute for most of their instances an online solution whose objective value was at most 1.77% (Problem BATTERY) and 1.24% (Problem IM) worse than the optimal objective value. This demonstrates that ODDO is capable of obtaining near-optimal solutions not only incidentally, but also has a stable good performance over a longer set of instances.

In order to obtain more insight into the overall performance of the candidate multipliers, we focus on their median performance. For most of the considered cases in Problem BATTERY, Figure 6.1a suggests that the median performance of the candidates C^{mean} and C^{med} is better than the median performance of candidates $C^{\min}$ and $C^{\max}$. An explanation for this is that the candidates C^{mean} and C^{med} depend on all training data, whereas the candidate $C^{\min}$ depends solely on the value of (extreme) outliers. In contrast, C^{mean} and C^{med} are more “balanced” and thus seem good candidates for the “average” instance.

Finally, we focus on the difference in performance between different sizes for the sets of training instances. Figure 6.1b suggests that, in Problem IM, the median performance of candidates $C^{\min}$, C^{mean} , and C^{med} does not differ significantly between the case of 10 training instances and the case of 50 training instances. Interestingly however, Figure 6.1a indicates that for Problem BATTERY these

(a) Problem BATTERY.

(b) Problem IM.

Figure 6.1: Boxplots of the observed ratios for the four candidate predictions, where for each candidate the black (left) boxplot corresponds to the case of 10 training instances and the gray (right) boxplot to the case of 50 training instances.

candidates seem to perform better in the case of 10 training instances than in the case of 50 training instances. One explanation for this could be that the latter case takes into account the data of the households for approximately three months, whereas the former case is based on data from only the past one-and-a-half week. However, household consumption is heavily influenced by weather (see, e.g., [95]). Thus, by taking into account the weather from three months ago when computing the candidate multiplier vectors, we consider situations that are not representative for the current household consumption and thus also

do not characterize well the optimal Lagrange multipliers corresponding to the current day.

Figure 6.2 shows the observed ratios of the online solutions for Problem BATTERY and Problem IM for both ODDO and the nominal strategy. Moreover, Table 6.1 shows the share of problem instances wherein ODDO outperforms the nominal strategy. For Problem BATTERY, Table 6.1 suggests that the share of problem instances wherein ODDO outperforms the nominal strategy decreases as the number of training instances increases. However, for Problem IM, ODDO outperforms the nominal strategy in all cases by a large margin. The main cause of this seems to be the large difference in overall performance of the nominal strategy between Problem BATTERY and Problem IM: the ratios of this strategy for Problem IM are significantly larger than for Problem BATTERY, whereas the ratios of ODDO are similar for both problems. Additionally, for Problem IM, in the nominal strategy, the overall behavior of the ratios does not significantly change as the number of training instances increases, whereas for ODDO increasing the number of training instances leads to a rapid decrease and a reduced spread of the ratios. One explanation for this difference is that the uncertain data in Problem IM are mutually independent both within one instance and between multiple instances. As a consequence, the training instances do not provide additional information on the data structure other than an improved estimate of the variance of the data. Another explanation is that in Problem IM the relative variance of the uncertain data is larger than in Problem BATTERY. More precisely, for Problem BATTERY the (estimated) coefficient of variation for multivariate data (as calculated according to [6]) of the uncertain data of the 50 test instances is only 0.1018, whereas these coefficients for the uncertain data $c_1 := (c_1^t)_{t \in \mathcal{T}}$, $c_2 := (c_2^t)_{t \in \mathcal{T}}$, and $c_3 := (c_3^t)_{t \in \mathcal{T}}$ of Problem IM are 0.1450, 0.2175, and 0.2900 respectively. This supports the claim made in Section 6.4.1 that large variances and unanticipated realizations of the uncertain data only have a limited influence on the performance of ODDO.

Size of training set	Success rate	
	Problem BATTERY	Problem IM
1	0.7	1
3	0.54	1
5	0.5	1
10	0.44	1

Table 6.1: Fraction of test instances wherein ODDO outperforms the nominal strategy.

Lastly, we focus on the influence of the number of training instances on the performance of both ODDO and the nominal strategy. On the one hand, for ODDO, the results in Figures 6.2a and 6.2b indicate that the ratios are only marginally affected by the chosen number of training instances. On the other

(a) Problem BATTERY; ODDO (left, black) and the nominal strategy (right, gray).

(b) Problem IM; ODDO.

(c) Problem IM; nominal strategy.

Figure 6.2: Boxplots of the observed ratio in the comparison between ODDO and the nominal strategy for each considered training set size.

hand, for the nominal strategy, Figure 6.2a suggests that the ratios for Problem BATTERY decrease with an increasing number of training instances and become competitive with ODDO only when the number of training instances is five or more. This suggests that ODDO can obtain good online solutions with a small training set and that these solutions are better than those produced by the nominal strategy.

Summarizing, the evaluation in this section shows that ODDO is able to achieve near-optimal results using relatively easy-to-compute candidates as input for problems with measured data (Problem BATTERY) and also for problems with randomly generated data (Problem IM). Moreover, the results for Problem IM

indicate that ODDO performs well in practice also for problems that do not fall into the class I_{sub} . This suggests the existence of a wider class of problems for which ODDO yields good online solutions. We plan to further investigate such a broader class in future research.

6.6 CONCLUSIONS AND OUTLOOK

In this chapter, we presented a new framework for optimization under uncertainty called “Online Duality-Driven Optimization” (ODDO). This framework is motivated by applications in decentralized energy management, where peak energy consumption needs to be minimized in order to maintain a proper operation of the distribution grid. The presented approach does not require any quantitative assumptions on the uncertain data involved in the problem such as uncertainty sets or probability distributions. The key idea of the framework is to predict the optimal Lagrange multipliers of the optimization problem instead of the actual uncertain data or objective function. We analyzed the robustness of this approach both in theory and in practice. For a specific but important class of problems, we derived bounds on the difference in objective value between the online and optimal offline solution. Moreover, evaluations suggest that in practice this robustness carries over to problems that fall outside this class. For the studied problems, simple statistics such as the mean and median of previously observed optimal Lagrange multipliers seem to perform well as input predictions to the ODDO framework. Summarizing, we are convinced that the ODDO framework is a promising addition to the set of paradigms for optimization under uncertainty and provides the community with a new approach to tackle these types of problems.

An important direction for future research is to find a structural and unifying way to predict the optimal Lagrange multipliers used in our approach. The statistics that we used as predictions in Section 6.5.4 were chosen mainly for their simplicity and as a means to demonstrate the good practical performance of ODDO. In Chapter 7, we focus on two alternative prediction approaches that explicitly exploit the structure of and robustness results for the problems studied in the current chapter. Using the electric vehicle EV charging problem introduced in Section 3.4 as an illustrative problem, we show that both approaches lead to structurally accurate online solutions.

The ODDO framework requires the feasible set of the given problem to be convex and the cost functions to be continuously differentiable and *strictly* convex. This restriction ensures that properties such as strong duality hold and enables us to derive the robustness results given in Section 6.4. In Chapter 8, we investigate how the validity of these properties and results are affected when we relax some of the used restrictions on the feasible set and the cost functions:

1. *One or more of the cost functions are not strictly convex and/or not continuously differentiable.* We already considered such a problem, namely the

inventory management problem IM. It can now happen that not only the optimal Lagrange multipliers might be non-unique, but also the optimal primal solution. Thus, when solving a subproblem during the online optimization process, we might have to choose from a set of multiple online solutions. In our evaluation, we did not include any preference for or requirement on choosing an online solution from this set; we simply used the solution computed by the solver. It would be interesting to investigate under which conditions it is useful to incorporate such a preference or requirement and what the effect of this is on the overall online optimization process. In Section 8.2, we investigate this issue for the specific problem of EV charging where the cost functions are piece-wise linear. This choice of cost function is motivated by a practical limitation of several EV charging infrastructures, namely that charging can be done only at a finite number of possible charging rates instead of at any rate between zero and a given maximum.

2. *One or more of the decision variables are binary or integer.* This implies that the feasible region of Problem P is not convex. In the area of energy management, binary and integer decision variables often arise when a device has several operation modes. It would be interesting to gain insight into if and how such a structure could be exploited to derive an approach similar to ODDO to solve the online version of such problems. As a first direction towards this goal, we derive in Section 8.3 an optimal solution characterization and an online optimization approach similar to ODDO specifically for an EV charging problem with binary state-switching variables. This variant of the EV charging problem is motivated by another practical limitation of EV battery chargers, namely that charging at a relatively low rate is either not possible or very inefficient.

6.7 APPENDIX: COMPUTING THE PROJECTION SETS $\mathcal{C}_{\text{PROJ}}^t$ FOR PROBLEM BATTERY

Suppose that we need to compute the online decision $\hat{x}^{\bar{t}}(\mu, \lambda)$ at the start of the interval $\bar{t}$. To compute the projection set $\mathcal{C}_{\text{proj}}^{\bar{t}}$, we iteratively eliminate the variables $x^T, x^{T-1}, \dots, x^{\bar{t}+1}$ from the constraints (6.6b)-(6.6d). First, we eliminate x^T . This yields the following inequalities on the remaining non-fixed variables $x^{\bar{t}}, \dots, x^{T-1}$:

$$\begin{aligned}
 0 &\leq S_{\text{start}} + \Delta t \sum_{s=1}^{\bar{t}-1} \hat{x}^s(\mu, \lambda) + \Delta t \sum_{s=\bar{t}}^t x^s \leq D, \quad t \in \{\bar{t}, \dots, T-1\}, \\
 S_{\text{end}} - \Delta t X_{\text{max}} &\leq S_{\text{start}} + \Delta t \sum_{s=1}^{\bar{t}-1} x^s(\mu, \lambda) + \Delta t \sum_{s=\bar{t}}^{T-1} x^s \leq S_{\text{end}} - \Delta t X_{\text{min}}, \\
 X_{\text{min}} &\leq x^t \leq X_{\text{max}}, \quad t \in \{\bar{t}, \dots, T-1\}.
 \end{aligned}$$

Observe that there are now two lower and upper bound constraints on the term $S_{\text{start}} + \Delta t \sum_{s=1}^{\bar{t}-1} \hat{x}^s(\mu, \lambda) + \sum_{s=\bar{t}}^t x^s$. We merge them by defining $\underline{\hat{D}}^{T-1} := \max(0, S_{\text{end}} - \Delta t X_{\text{max}})$ and $\hat{\hat{D}}^{T-1} := \min(D, S_{\text{end}} - \Delta t X_{\text{min}})$. Thus, we obtain

$$\begin{aligned} 0 &\leq S_{\text{start}} + \Delta t \sum_{s=1}^{\bar{t}-1} \hat{x}^s(\mu, \lambda) + \Delta t \sum_{s=\bar{t}}^t x^s \leq D, \quad t \in \{\bar{t}, \dots, T-2\}, \\ \underline{\hat{D}}^{T-1} &\leq S_{\text{start}} + \Delta t \sum_{s=1}^{\bar{t}-1} x^s(\mu, \lambda) + \Delta t \sum_{s=\bar{t}}^{T-1} x^s \leq \hat{\hat{D}}^{T-1}, \\ X_{\text{min}} &\leq x^t \leq X_{\text{max}}, \quad t \in \{\bar{t}, \dots, T-1\}. \end{aligned}$$

Now we eliminate the next variable x^{T-1} . For this, we can apply the same reasoning as in the elimination of x^T . This yields the following inequalities:

$$\begin{aligned} 0 &\leq S_{\text{start}} + \Delta t \sum_{s=1}^{\bar{t}-1} \hat{x}^s(\mu, \lambda) + \Delta t \sum_{s=\bar{t}}^t x^s \leq D, \quad t \in \{\bar{t}, \dots, T-3\}, \\ \underline{\hat{D}}^{T-2} &\leq S_{\text{start}} + \Delta t \sum_{s=1}^{\bar{t}-1} x^s(\mu, \lambda) + \Delta t \sum_{s=\bar{t}}^{T-2} x^s \leq \hat{\hat{D}}^{T-2}, \\ X_{\text{min}} &\leq x^t \leq X_{\text{max}}, \quad t \in \{\bar{t}, \dots, T-2\}, \end{aligned}$$

where $\underline{\hat{D}}^{T-2} := \max(0, \underline{\hat{D}}^{T-1} - \Delta t X_{\text{max}})$ and $\hat{\hat{D}}^{T-2} := \min(D, \hat{\hat{D}}^{T-1} - \Delta t X_{\text{min}})$. We can continue this process until all variables x^t with $t > \bar{t}$ have been eliminated. The only remaining constraint is then

$$\begin{aligned} &\max \left(\underline{\hat{D}}^{\bar{t}} - S_{\text{start}} - \Delta t \sum_{s=1}^{\bar{t}-1} \hat{x}^s(\mu, \lambda), X_{\text{min}} \right) \\ &\leq x^{\bar{t}} \\ &\leq \min \left(\hat{\hat{D}}^{\bar{t}} - S_{\text{start}} - \Delta t \sum_{s=1}^{\bar{t}-1} \hat{x}^s(\mu, \lambda), X_{\text{max}} \right), \end{aligned}$$

where $\underline{\hat{D}}^t := \max(0, \underline{\hat{D}}^{t+1} - \Delta t X_{\text{min}})$ and $\hat{\hat{D}}^t := \min(D, \hat{\hat{D}}^{t+1} - \Delta t X_{\text{max}})$ for $t \in \{1, \dots, T-1\}$. Thus, the projected problem is the minimization of a quadratic function over a closed interval whose boundaries can be computed efficiently. Moreover, observe that we can compute the entire vectors $\underline{\hat{D}}$ and $\hat{\hat{D}}$ in $O(T)$ time at the start of the first interval since they do not depend on any of the online decisions $\hat{x}^t(\mu, \lambda)$.

7

MULTIPLIER PREDICTION FOR ONLINE DUALITY-DRIVEN OPTIMIZATION

ABSTRACT – *In the previous chapter, we introduced a mathematical framework for online optimization problems. This methodology requires as input predictions of the optimal Lagrange multipliers of the deterministic version of the problem. In this chapter, we present two approaches for predicting these Lagrange multipliers. The first is based on quantile prediction and incorporates a given preference for under-or over-predicting the Lagrange multipliers. The second approach exploits the structure of optimal solutions to optimization problems with submodular constraints. Evaluations of these two approaches applied to the problem of scheduling electric vehicle charging suggest that both approaches are able to produce predictions that lead to online solutions with an optimality gap of less than 1%. Moreover, while the quantile-based approach is able to yield better results than the structure-based approach, the latter is more robust in the sense that its good performance does not depend on the fine-tuning of hyper-parameters.*

7.1 INTRODUCTION

Many optimization problems in practice require input data that is not known on forehand. Several methods and techniques exist wherein such data uncertainty can be incorporated to compute a solution that is in some sense robust against unexpected realizations of this data (see Section 6.2.1). We introduced in the previous chapter a new methodology for incorporating data uncertainty into optimization problems called "Online Duality-Driven Optimization" (ODDO). The key idea in this approach is that instead of aiming to predict the uncertain

This chapter is based on and contains parts of [SU:3] and [SU:8].

data itself, we predict the optimal Lagrange multipliers of the deterministic version of the problem. Subsequently, via the concept of Lagrangian duality, we use this prediction to compute an online solution to the problem, whereby in each step we do not require any other information on future data.

A question that remained open is how the optimal Lagrange multipliers should be predicted. The results in the previous chapter showed that taking as predictions simple statistics, such as the mean and median of the optimal Lagrange multipliers of historical instances of the problem, already yield good online solutions. The reason behind choosing these statistics as candidate multipliers was their simplicity. Moreover, the main goal was to demonstrate the robustness of ODDO against prediction errors. In general, to successfully use ODDO within a given application, it is important to have a prediction approach for the Lagrange multipliers that provides guarantees on the quality of the eventual online solution. This requires a deeper understanding of the relation between these multipliers, the uncertain data, and the quality of the online solution.

As a first step in this direction, we present in this chapter two simple but accurate approaches to predict the optimal Lagrange multipliers. The first approach is motivated by the robustness results in Section 6.4.2 and the evaluation results in Section 6.5.4. These findings suggested that the quality of the online solution is influenced significantly by whether we under- or over-predict the multipliers. In our first approach, we predict optimal Lagrange multipliers while actively incorporating a given preference for under- or over-prediction. We do this by considering the optimal Lagrange multipliers as stochastic variables with a certain probability distribution and expressing the preference as the probability that the computed prediction is smaller than the optimal Lagrange multiplier. This approach corresponds with the concept of quantile functions in statistics [148].

The second approach is motivated by the structure of resource allocation problems (RAPs) and in particular (a, b, f) -separable RAPs with box constraints (see Section 2.3.1). Under the assumption that only a few inequality constraints of the problem are tight in an optimal solution, one can obtain from the Karush-Kuhn-Tucker (KKT) optimality conditions (see, e.g., [29]) an expression for the optimal Lagrange multipliers. This expression does not depend on individual uncertain data, but only on a single value that is in a sense the aggregation of the uncertain data.

We illustrate the effectiveness of these simple approaches by taking as an example the problem of scheduling the charging of electric vehicles (EVs), i.e., Problem EV introduced in Section 3.4. For this problem, we show that our method is able to predict the optimal Lagrange multiplier such that an online solution can be obtained with an optimality gap of less than 1% in the considered case study. Similarly to the evaluation in Section 6.5.4, we use information of previous hypothetical charging windows as input for the multiplier prediction and in this way we learn the behavior of the optimal Lagrange multiplier for a

given charging session. We show that this behavior is very stable over consecutive days, which enables us to compute accurate predictions that lead to accurate online solutions.

The remainder of this chapter is organized as follows. In Section 7.2, we analyze the EV charging problem that is used as a leading example in the context of ODDO and present our two prediction approaches. In Section 7.3, we perform a simulation study to validate our approaches. Finally, Section 7.4 provides conclusions and a short discussion on the generalization of our prediction approaches to other problems.

7.2 TWO PREDICTION APPROACHES

In this section, we first revisit in Section 7.2.1 the EV charging problem introduced in Section 3.4. This problem serves as a leading example for our prediction approaches. Moreover, we analyze this problem and its online version in terms of ODDO, i.e., in terms of Problem P introduced in Section 6.3.1. Subsequently, we present our prediction approaches in Sections 7.2.2 (the quantile-based approach) and 7.2.3 (the structure-based approach).

7.2.1 THE ONLINE EV CHARGING PROBLEM

As a leading example, we take the EV charging problem introduced in Section 3.4. Using the notation for the general Problem P as introduced in Chapter 6, this problem can be formulated as follows:

$$\begin{aligned} \text{EV} : \quad & \min_{x^1, \dots, x^T} \sum_{t \in \mathcal{T}} \frac{1}{2} (x^t + p^t)^2 \\ & \text{s.t.} \quad \sum_{t \in \mathcal{T}} x^t = \frac{\bar{R}}{\Delta t}, \\ & \quad 0 \leq x^t \leq X_{\max}, \quad t \in \mathcal{T}. \end{aligned}$$

Its local Lagrangian solution for a given multiplier $\lambda \in \mathbb{R}$ is given by

$$x^t(\lambda) = \begin{cases} 0 & \text{if } \lambda - p^t < 0, \\ \lambda - p^t & \text{if } 0 \leq \lambda - p^t < X_{\max}, \\ X_{\max} & \text{if } X_{\max} \leq \lambda - p^t, \end{cases} \quad (7.1)$$

where we have "flipped" the sign of λ compared to the original formulation of Problem P analogously to Problem QRA in Section 6.4.1. The main reason for this is to maintain the physical interpretation of the optimal Lagrange multiplier λ^* for Problem EV, namely the level of flatness of the combined EV and base load profile.

In the following, we adapt Algorithm 6.1 for the online version of Problem P to the specific situation of Problem EV based on the "flipped" local Lagrangian

solution in Equation (7.1). For this, note that, for a prediction $\hat{\lambda}$ of λ^* , the projection sets $\mathcal{C}_{\text{proj}}^{\bar{t}}$ (see Equation (6.3)) are given by

$$\mathcal{C}_{\text{proj}}^{\bar{t}} := \left[\max\left(0, \frac{\bar{R}}{\Delta t} - \sum_{t=1}^{\bar{t}-1} \hat{x}^t(\hat{\lambda}) - (T - \bar{t})X_{\max}\right), \min(X_{\max}, \hat{R}^{\bar{t}}(\hat{\lambda})) \right],$$

where $\hat{R}^{\bar{t}}(\hat{\lambda}) := \frac{\bar{R}}{\Delta t} - \sum_{t=1}^{\bar{t}-1} \hat{x}^t(\hat{\lambda})$. Combining this with the definition of the local Lagrangian solution in Equation (7.1), we obtain the following expression for the online solution $\hat{x}^t(\hat{\lambda})$ to Problem EV corresponding to a given prediction $\hat{\lambda}$ of λ^* :

$$\hat{x}^{\bar{t}}(\hat{\lambda}) := \max\left(0, \hat{R}^{\bar{t}}(\hat{\lambda}) - (T - \bar{t})X_{\max}, \min\left(X_{\max}, \hat{R}^{\bar{t}}(\hat{\lambda}), \hat{\lambda} - p^{\bar{t}}\right)\right), \quad (7.2)$$

Figure 7.1 illustrates this decision rule and shows the impact of under- and over-predicting of λ^* on the structure of the overall online solution, i.e., on the general charging behavior of the EV.

7.2.2 INCORPORATING A PREFERENCE FOR UNDER- OR OVER-PREDICTION

In this section, we present our first approach to predict optimal Lagrange multipliers. As mentioned before, this approach actively incorporates a preference for under- or over-predicting the multipliers. We now discuss why one could have a reason to prefer under- or over-prediction of the optimal Lagrange multiplier. The first reason is based on the robustness results for ODDO as derived in Chapter 6, whereas the second reason is based on the application of the online optimization approach within a neighborhood with multiple EVs. Subsequently, we present our prediction approach.

Observe that, assuming $p^t \geq 0$ for all $t \in \mathcal{T}$, the robustness result of Theorem 6.1 applies to Problem EV. In particular, the special case of Corollary 6.2 for polynomial cost functions applies, where $K = \frac{1}{2}$, $c = 2$, and $v^t(\lambda) = \lambda$ (recall that we "flipped" λ). Thus, the following robustness result holds for the online version of Problem EV and the online solution given by Equation (7.2):

Corollary 7.1. *Given an instance of Problem EV and a prediction $\hat{\lambda}$ of the optimal Lagrange multiplier λ^* such that $\hat{\lambda} \geq \lambda^*$, it holds that*

$$\sum_{t \in \mathcal{T}} \frac{1}{2} (\hat{x}^t(\hat{\lambda}) + p^t)^2 - \sum_{t \in \mathcal{T}} \frac{1}{2} (x^t(\lambda^*) + p^t)^2 \leq \frac{1}{2} T (\hat{\lambda}^2 - (\lambda^*)^2),$$

i.e., the difference in objective value between the online and optimal offline solution is bounded by $\frac{1}{2} T (\hat{\lambda}^2 - (\lambda^*)^2)$.

Since this robustness result holds only for the case that we over-predict the optimal Lagrange multiplier, i.e., only when $\hat{\lambda} \geq \lambda^*$, over-predicting the optimal

Figure 7.1: Online EV charging using an optimal Lagrange multiplier λ^* with (a) $\hat{\lambda} > \lambda^*$ and (b) $\hat{\lambda} < \lambda^*$.

Lagrange multiplier will be the preferred option when we consider the charging of a single EV and we wish to flatten the combined profile as much as possible.

However, from a neighborhood perspective, under-predicting λ^* can be beneficial for the aggregated power profile of the entire neighborhood. To see this, suppose multiple EVs within the same neighborhood are charging simultaneously and that their individual charging is managed by means of ODDO. Also suppose that they have a similar charging window and deadline (e.g., during the night and deadlines between 7:00h and 9:00h). If the optimal Lagrange multipliers of all EVs are over-predicted, then all EVs will be done charging before their actual deadline (see also Figure 7.1a). Near the end of the charging window, almost no charging is done anymore by any of the EVs. Therefore, it would

not be so bad if the optimal Lagrange multipliers of a few of the EVs are under-predicted: this simply means that a few EVs will have to charge at full power near the end of the charging window. As a consequence, they fill up the valley near the end of the charging window and in addition decrease the aggregated power consumption during the main part of their charging window (see also Figure 7.1b).

We conclude that a preference for either under- or over-prediction of the optimal fill-level should be taken actively into account when developing a prediction approach. We choose to model this preference by means of a risk parameter $\alpha \in [0, 1]$, where $\alpha = 0$ implies that we always want $\hat{\lambda} \leq \lambda^*$ and $\alpha = 1$ implies that we always want $\hat{\lambda} \geq \lambda^*$. Our goal is now to compute to a given α a prediction $\hat{\lambda}$ such that the probability that $\hat{\lambda} \leq \lambda^*$ is exactly α . For this, suppose that λ^* is a stochastic variable that has a cumulative distribution function (CDF) F_{λ^*} . Then the goal stated above corresponds to finding $\hat{\lambda}$ such that

$$\alpha = \mathbb{P}(\hat{\lambda} \geq \lambda^*) = F_{\lambda^*}(\hat{\lambda}).$$

This is equivalent to

$$\hat{\lambda} = F_{\lambda^*}^{-1}(\alpha). \quad (7.3)$$

This is the base for a simple approach for predicting λ^* . Given $F_{\lambda^*}^{-1}$ and α , we simply compute $\hat{\lambda}$ by applying Equation (7.3). Since F_{λ^*} and thus $F_{\lambda^*}^{-1}$ are unknown, we approximate F_{λ^*} by an appropriate distribution. More precisely, we construct its empirical CDF $F_{\lambda^*}^{\text{emp}}$ from a set of samples of λ^* obtained from solving a series of instances of Problem EV whose parameter choices are the same except for the base load profile p . Subsequently, we use this function as an approximation of the CDF of λ^* . One disadvantage of using an empirical CDF to compute $\hat{\lambda}$ is that it is a discontinuous function, meaning that there exist values of α for which $(F_{\lambda^*}^{\text{emp}})^{-1}$ is not defined. To avoid this when computing $\hat{\lambda}$ using Equation (7.3), we construct from $F_{\lambda^*}^{\text{emp}}$ a piece-wise linear approximation that has the discontinuities of $F_{\lambda^*}^{\text{emp}}$ as its breakpoints. The resulting CDF, which we denote by F'_{λ^*} , is continuous and thus its inverse $(F'_{\lambda^*})^{-1}$ exists and can be used to evaluate (7.3).

7.2.3 PREDICTIONS BASED ON THE PROBLEM STRUCTURE

In this section, we present our second prediction approach that is based on the structure of the optimal solution $x(\lambda^*)$ to Problem EV. For this, suppose that in this optimal solution we have that each variable is strictly in between its bounds, i.e., we have $0 < x^t(\lambda^*) < X_{\max}$ for all $t \in \mathcal{T}$. This means that the optimal solution is equal to the local Lagrangian solution evaluated for the optimal Lagrange multiplier λ^* , i.e., $x^t(\lambda^*) = \lambda^* - p^t$. Summing the terms $x^t(\lambda^*)$ over t yields

$$\frac{\bar{R}}{\Delta t} = \sum_{t \in \mathcal{T}} x^t(\lambda^*) = \sum_{t \in \mathcal{T}} \lambda^* - p^t = T\lambda^* - \sum_{t \in \mathcal{T}} p^t.$$

It follows that

$$\lambda^* = \frac{\frac{\bar{R}}{\Delta t} + \sum_{t \in \mathcal{T}} p^t}{T}. \quad (7.4)$$

Note that in this expression, λ^* does not depend explicitly on each individual element of the uncertain vector p but only on the aggregated term $\sum_{t \in \mathcal{T}} p^t$. This suggests that we can obtain a promising prediction for λ^* via Equation (7.4) using a prediction of this aggregated term instead of a prediction of each individual element of p . By the law of large numbers, accurately predicting such an aggregated term is in general easier than predicting each individual element. In particular, this is true for power consumption since the average power consumption over a larger time period is in general significantly easier to predict than the power consumption of individual smaller intervals (see, e.g., [91]).

We can predict the aggregated term similarly to the creation of a sample set for λ^* in our first prediction approach in Section 7.2.2. More precisely, we compute for a given set of previous days the aggregated power consumption over the time period specified by $\mathcal{T}$ and thereby create a sample set for the term $\sum_{t \in \mathcal{T}} p^t$.

The relation between the optimal Lagrange multiplier and the uncertain vector p in Equation (7.4) can be generalized to the case where the cost functions have the form $a^t f(\frac{x^t}{a^t} + b^t)$ for a known convex differentiable function f , a known vector $a \in \mathbb{R}_{>0}^n$, and an uncertain vector $b \in \mathbb{R}^n$ (see also Section 6.4). In that case, the expression for λ^* becomes

$$\lambda^* = \nabla f \left(\frac{\frac{\bar{R}}{\Delta t} + \sum_{t \in \mathcal{T}} a^t b^t}{\sum_{t \in \mathcal{T}} a^t} \right). \quad (7.5)$$

This relation is particularly interesting for the problem class I_{sub} . This is because for each problem of this class, there exists a partition of the decision variables such that the problem can be decomposed into a collection of RAPs, one over each element of the partition (see, e.g., [49]). This partition corresponds directly with the set of constraints that are tight in the optimal primal solution and thereby with the set of nonzero multipliers in the optimal Lagrange multiplier vector. If we are able to predict this partition accurately, we can also obtain a promising prediction of the optimal Lagrange multiplier for each RAP subproblem using Equation (7.5).

7.3 SIMULATION STUDY

In this section, we validate the prediction approaches presented in Section 7.2 and evaluate their efficiency. For this, we simulate the charging of a single EV using ODDO, where we predict $\hat{\lambda}$ using the two presented approaches. Section 7.3.1 explains the setup of the simulations and the different used scenarios. Section 7.3.2 presents and discusses the simulation results.

For all simulations, we divide the charging windows into 15-minute time intervals, i.e., $\Delta t = \frac{1}{4}$. We consider two different charging windows that represent charging during the day and charging during the night, namely 7:00h-19:00h and 19:00h-7:00h respectively. We use the Nissan Leaf 2018 model as a reference EV, which means that the considered battery capacity is 40 kWh [3]. In order to assess the impact of different charging requirements, we consider both the situation where the battery is empty and needs to be fully charged ($\bar{R} = 40$ kWh) and the situation where only 25% of the capacity needs to be charged ($\bar{R} = 10$ kWh). We assume that a home charger with a maximum rate of 6.6 kW is used. For the base load profile, we use real power consumption data of a household that was obtained in the field test described in [60]. To construct $(F')_{\hat{\lambda}^*}^{-1}$ in the first approach and the sample set of aggregated power consumption $\sum_{t \in \mathcal{T}} p^t$, we consider four different sets of previous days as training data. These sets consist of the previous 3, 10, 50 and 100 days respectively.

For the first quantile-based approach, we compute the sample set of optimal Lagrange multipliers, i.e., we solve Problem EV for these previous days, by applying the sequential breakpoint search algorithm 3.1 discussed in Section 3.3.1. For each of the above scenarios, we consider a broad range of risk parameters $\tilde{\alpha} = \{0.05, 0.15, \dots, 0.95\}$ and for each choice of $\alpha \in \tilde{\alpha}$, we compute $\hat{\lambda}$ using (7.3). To compute $\hat{\lambda}$ for the second structure-based approach, we use the average aggregated power consumption of the previous days and using this as input in Equation (7.4). In both approaches, for each prediction $\hat{\lambda}$, we apply ODDO via Equation (7.2) using this $\hat{\lambda}$ as input. Subsequently, we compute the objective value of the online solution and compare this to the optimal offline solution for this problem instance. Moreover, for the first approach, we also determine whether $\hat{\lambda}$ is either an under- or over-prediction of λ^* . We repeat this procedure for 100 subsequent days.

To also assess the influence of local photovoltaic (PV) production, we also evaluate our approaches on two similar households of which one has PV panels and thus a large production peak during the day. The base load data for these houses has been obtained in the same field test in [60] and Figure 7.2 shows a typical daily base load profile for these two houses. For the evaluation, this time we consider the charging window 7:00h-19:00h, use 10 previous days as sample days, and repeat the approach for 10 subsequent days. As $\bar{R} = 40$ kWh is too small to be able to flatten the base load profile of the house with PV, we also simulated these cases for an EV with a larger battery, where we used a Tesla Model S as reference EV ($\bar{R} = 100$ kWh, $X_{\max} = 11.5$ kW).

7.3.2 RESULTS AND DISCUSSION

First, we discuss the accuracy of the quantile-based approach with regard to incorporating a preference for under- or over-prediction. Tables 7.1 and 7.2 show

Figure 7.2: Typical base load profile of the two houses used in the PV study.

the ratio of days where $\hat{\lambda} \geq \lambda^*$ for each scenario and each risk parameter in the set $\tilde{\alpha}$ for the charging windows 7:00h-19:00h and 19:00h-7:00h respectively. Note that when this ratio is close to its corresponding α , our approach is able to appropriately take into account the preference α for under- or over-prediction. The charging requirements do not seem to influence the ratios significantly. However, in contrast, for the cases where 10 or 50 days are used as training days, the ratios tend to be closer to α than for the cases where 3 or 100 days are used. One possible explanation for this is that the previous 10 and 50 days lead to a good combination of both short-term and long-term behavior of λ^* . Information on both behaviors is required to make a good prediction (see also [95] and the results in Section 6.5.4). The previous 3 days clearly cannot provide us with sufficient information on the long-term behavior of λ^* . Similarly, if we use the previous 100 days and treat all these days equally, i.e., do not assign larger weights to recent days, we neglect any short-term behavior. Summarizing, the results in Tables 7.1 and 7.2 suggest that for the cases of 10 or 50 training days, the quantile-based approach is able to appropriately take into account the preference α for under- or over-predicting the optimal Lagrange multiplier λ^* .

We now compare the performance of our two prediction approaches, i.e., the quality of their resulting online solutions. To this end, Tables 7.3-7.5 show the median ratio between the objective value of the online solution and of the optimal solution for the quantile-based approach (Table 7.3 for the charging window 7:00h-19:00h and Table 7.4 for the charging window 19:00h-7:00h) and for the structure-based approach (Table 7.5). In both approaches, the number of training days does not seem to significantly affect the ratios. In contrast, the online solutions for $\bar{R} = 40$ kWh appear to be much better than those for $\bar{R} = 10$ kWh and in some cases even have an optimality gap of less than 1%. Moreover, the ratios for the charging window 19:00h-7:00h are significantly lower than those for the window 7:00h-19:00h. One explanation for this is that

α	$\bar{R} = 10 \text{ kWh}$				$\bar{R} = 40 \text{ kWh}$			
	3d	1od	5od	1ood	3d	1od	5od	1ood
0.05	0.25	0.13	0.09	0.07	0.24	0.12	0.09	0.07
0.15	0.29	0.22	0.20	0.21	0.30	0.22	0.19	0.20
0.25	0.38	0.31	0.25	0.25	0.37	0.30	0.26	0.26
0.35	0.45	0.36	0.30	0.35	0.45	0.36	0.30	0.35
0.45	0.50	0.49	0.45	0.49	0.50	0.48	0.45	0.49
0.55	0.55	0.51	0.58	0.60	0.55	0.51	0.58	0.60
0.65	0.58	0.62	0.64	0.70	0.57	0.62	0.64	0.70
0.75	0.61	0.71	0.74	0.77	0.60	0.71	0.73	0.77
0.85	0.64	0.80	0.83	0.91	0.64	0.80	0.84	0.91
0.95	0.73	0.89	0.95	0.98	0.72	0.88	0.92	0.97

Table 7.1: Fraction of cases where $\hat{\lambda} > \lambda^*$ for the charging window 7:00h-19:00h for the quantile-based approach. Values close to α imply an accurate modeling of the risk specified by α .

α	$\bar{R} = 10 \text{ kWh}$				$\bar{R} = 40 \text{ kWh}$			
	3d	1od	5od	1ood	3d	1od	5od	1ood
0.05	0.27	0.16	0.10	0.12	0.26	0.16	0.10	0.12
0.15	0.31	0.20	0.19	0.23	0.31	0.19	0.19	0.23
0.25	0.34	0.26	0.27	0.29	0.35	0.25	0.27	0.27
0.35	0.40	0.37	0.39	0.44	0.45	0.37	0.39	0.44
0.45	0.47	0.49	0.56	0.66	0.51	0.50	0.55	0.65
0.55	0.56	0.55	0.69	0.78	0.56	0.57	0.67	0.76
0.65	0.59	0.64	0.80	0.90	0.60	0.64	0.75	0.82
0.75	0.65	0.73	0.84	0.94	0.66	0.75	0.80	0.90
0.85	0.69	0.82	0.93	0.95	0.71	0.80	0.89	0.93
0.95	0.70	0.86	0.97	1.00	0.74	0.86	0.95	0.98

Table 7.2: Fraction of cases where $\hat{\lambda} > \lambda^*$ for the charging window 19:00h-7:00h for the quantile-based approach. Values close to α imply an accurate modeling of the risk specified by α .

the base load profile of a household generally contains fewer human-created peaks during the night than during the day. As a consequence, the base load is relatively constant and is not very different for consecutive days, which reduces the variation among the samples of λ^* and $\sum_{t \in \mathcal{T}} p^t$. Finally, we observe for the quantile-based approach that the online solutions are generally better for $\alpha > 0.5$ than for $\alpha < 0.5$. This is in line with the robustness results and numerical evaluations for ODDO in Sections 6.4.2 and 6.5.4 respectively (see also Corollary 7.1) and with the observations in [56, 147] that predicting λ^* too high leads to a lower objective value than predicting λ^* too low.

α	$\bar{R} = 10 \text{ kWh}$				$\bar{R} = 40 \text{ kWh}$			
	3d	10d	50d	100d	3d	10d	50d	100d
0.05	1.212	1.549	1.669	1.697	1.029	1.039	1.049	1.050
0.15	1.153	1.460	1.503	1.490	1.023	1.034	1.035	1.035
0.25	1.128	1.311	1.410	1.414	1.021	1.028	1.026	1.029
0.35	1.104	1.143	1.143	1.136	1.022	1.023	1.026	1.022
0.45	1.108	1.112	1.113	1.099	1.020	1.021	1.018	1.018
0.55	1.090	1.085	1.085	1.079	1.021	1.018	1.017	1.016
0.65	1.089	1.088	1.070	1.064	1.021	1.017	1.015	1.015
0.75	1.086	1.071	1.059	1.046	1.019	1.019	1.016	1.018
0.85	1.072	1.060	1.050	1.059	1.020	1.020	1.019	1.023
0.95	1.073	1.063	1.060	1.079	1.019	1.022	1.028	1.035

Table 7.3: Median ratio between online and optimal objective value for the charging window 7:00h-19:00h for the quantile-based prediction approach.

α	$\bar{R} = 10 \text{ kWh}$				$\bar{R} = 40 \text{ kWh}$			
	3d	10d	50d	100d	3d	10d	50d	100d
0.05	1.058	1.121	1.132	1.122	1.008	1.011	1.012	1.011
0.15	1.046	1.074	1.082	1.061	1.006	1.007	1.007	1.005
0.25	1.033	1.046	1.033	1.031	1.004	1.005	1.004	1.004
0.35	1.025	1.028	1.025	1.019	1.004	1.003	1.003	1.002
0.45	1.022	1.019	1.015	1.012	1.003	1.002	1.002	1.002
0.55	1.022	1.015	1.011	1.017	1.004	1.001	1.002	1.003
0.65	1.021	1.015	1.016	1.019	1.003	1.002	1.002	1.004
0.75	1.023	1.017	1.018	1.030	1.003	1.003	1.004	1.006
0.85	1.020	1.018	1.024	1.036	1.003	1.004	1.006	1.010
0.95	1.017	1.022	1.034	1.049	1.004	1.008	1.016	1.020

Table 7.4: Median ratio between online and optimal objective value for the charging window 19:00h-7:00h for the quantile-based prediction approach.

Charging window	$\bar{R} = 10 \text{ kWh}$				$\bar{R} = 40 \text{ kWh}$			
	3d	10d	50d	100d	3d	10d	50d	100d
7:00h-19:00h	1.095	1.102	1.115	1.103	1.021	1.021	1.019	1.018
19:00h-7:00h	1.022	1.017	1.013	1.012	1.003	1.002	1.002	1.002

Table 7.5: Median ratio between online and optimal objective value for the structure-based prediction approach.

When comparing the median ratios between the two approaches, we observe that for almost all combinations of charging window and charging requirement $\bar{R}$, the median ratio of the structure-based approach is always only slightly higher than the best median ratio attained in the quantile-based approach among all values

of α . This suggests on the one hand that the quantile-based approach is able to obtain better online solutions than the structure-based approach. However, on the other hand, depending on the choice of α , the online solutions produced by the quantile-based approach can also be significantly worse than those of the structure-based approach, whereas the solution quality of the latter approach is not influenced by the choice of α . Thus, the structure-based approach seems to be more robust than the quantile-based approach in the sense that it often computes online solutions with a good objective value with only a small chance of outliers.

For the charging window 7:00h-19:00h and the case where $\bar{R} = 10$ kWh, the median ratios of the structure-based approach are significantly higher than the best median ratios of the quantile-based approach. This suggests that the combination of a relatively peaked base load profile (as compared to the charging window 19:00h-7:00h) and a lower charging requirement reduce the validity of the assumptions behind the structure-based approach, namely that the aggregated power consumption $\sum_{t \in \mathcal{T}} p^t$ is relatively stable over the course of consecutive days and that the combined EV and base load is the same for many of the time intervals. Future research should investigate in more detail for which practical EV charging scenarios these assumptions are valid.

Finally, Tables 7.6 and 7.7 show the results of the PV study. These results indicate that the presence of PV production diminishes the effectiveness of both prediction approaches. However, the median ratios improve, i.e., decrease as $\bar{R}$ increases and for the structure-based approach even result in an optimality gap of less than 0.3% for the Tesla case. Similarly to the results in Tables 7.3-7.5, the median ratios of the structure-based approach are better than the best median ratios of the quantile-based approach. This indicates that the presence of PV does not significantly influence the difference in performance between the two approaches.

7.4 CONCLUSIONS AND DISCUSSION

In this chapter, we presented two approaches for predicting the optimal Lagrange multipliers within the online duality-driven optimization framework introduced in Chapter 6. The first approach models the unknown optimal Lagrange multipliers as stochastic variables and computes a prediction that actively incorporates a desired preference for under- or over-prediction of the multipliers. The second approach exploits the structure of (a, b, f) -separable resource allocation problems (RAPs) over submodular constraints and in particular of the (a, b, f) -separable RAP over box constraints to compute a prediction based on the aggregation of the unknown vector b . Evaluations of both approaches with regard to the electric vehicle charging problem introduced in Section 3.4 suggest that both approaches are able to compute predictions that lead to online solutions with near-optimal objective value.

α	$\bar{R} = 10 \text{ kWh}$		$\bar{R} = 40 \text{ kWh}$		Tesla case	
	PV	No PV	PV	No PV	PV	No PV
0.05	2.486	1.133	2.766	1.017	1.140	1.003
0.15	1.979	1.093	2.114	1.011	1.132	1.002
0.25	1.799	1.050	1.925	1.014	1.113	1.002
0.35	1.621	1.048	1.847	1.014	1.104	1.002
0.45	1.260	1.025	1.719	1.014	1.091	1.002
0.55	1.199	1.023	1.794	1.013	1.094	1.002
0.65	1.375	1.026	1.940	1.015	1.103	1.003
0.75	1.986	1.039	2.101	1.019	1.192	1.005
0.85	1.653	1.047	2.226	1.022	1.213	1.007
0.95	1.791	1.051	2.385	1.031	1.240	1.012

Table 7.6: Median ratio between online and optimal objective value for the PV study for the quantile-based approach.

$\bar{R} = 10 \text{ kWh}$		$\bar{R} = 40 \text{ kWh}$		Tesla case	
PV	No PV	PV	No PV	PV	No PV
1.598	1.021	1.652	1.011	1.081	1.002

Table 7.7: Median ratio between online and optimal objective value for the PV study for the structure-based approach.

An important direction for future research is to assess the performance of our prediction approaches for problems with more than one long-term constraint, i.e., with more than one Lagrange multiplier. Although the good results of these prediction approaches for the case of one Lagrange multiplier seem promising for this generalization, moving from one to multiplier Lagrange multipliers introduces several methodological challenges, which we briefly discuss below.

The quantile-based approach is based on the well-defined concept of quantile functions for one-dimensional stochastic variables. However, this concept is not well-defined anymore for multi-dimensional stochastic variables, i.e., for multiple Lagrange multipliers. As a consequence, one must choose one of the many alternative definitions of quantile functions in this situation that all have their own pros and cons (see also [148]). In this, one should carefully analyze and incorporate dependencies among the different multipliers, which is often either ignored or drastically simplified in practice (see, e.g., [96]).

The structure-based approach builds on the assumption that in an optimal solution all or very few of the box constraints are tight. Already for the case of only a single Lagrange multiplier, the evaluation results in Section 7.3.2 indicate that this assumption is not always valid for realistic problem instances. As mentioned in that section, further research is required to assess the validity of this assumption for different problems, both with only a single and with multiple

Lagrange multipliers. Furthermore, in the case of multiple Lagrange multipliers, the success of the structure-based approach depends strongly on how well one can predict which constraints are tight, i.e., which Lagrange multipliers are nonzero, in the optimal solution. Only when we are able to do this accurately, we can properly apply the prediction approach for problems with a single Lagrange multiplier to each of the subproblems induced by the division between tight and loose constraints in the optimal solution.

EXTENSIONS OF ONLINE DUALITY-DRIVEN OPTIMIZATION

ABSTRACT – In Chapter 6, we introduced a mathematical framework for online optimization problems. Within this framework, we made several assumptions on the nature of the uncertain cost functions and of the feasible set of the problem. These assumptions ensured the validity of the approach and allowed to prove several performance guarantees. In this chapter, we study the necessity of three of these assumptions by means of two variants of the electric vehicle scheduling problem introduced in Section 3.4. For these two variants, we discuss the consequences of lifting the assumptions and present an adapted version of the online optimization framework. Simulation results indicate that the quality of the online solutions obtained by this adapted version is comparable to that obtained by the original framework for their respective problems. This suggests that although the three assumptions are necessary for formal analyses of the framework, they are not per se required for its execution and good practical performance.

8.1 INTRODUCTION

We introduced in Chapter 6 a new framework for solving online optimization problems called "Online Duality-Driven Optimization" (ODDO). In this framework, optimal offline solutions can be characterized in terms of optimal Lagrange multipliers and subsequently predictions of these Lagrange multipliers can be used as input to compute an online solution. For a specific problem class consisting of resource allocation problems with additional submodular constraints, i.e., problems of the type S/SC/C (see Section 2.3.5), this structure and the convexity of the problem allowed to derive guarantees on the quality of the online solution.

This chapter is based on and contains parts of [SU:5] and [SU:7].

In order to be able to apply Lagrangian duality theory in Sections 6.3.2 and 6.3.3 and prove the robustness results of Section 6.4.2, we made several assumptions on the problem instances. In particular, we made the following three assumptions:

1. The cost functions are continuously differentiable, i.e., their gradients are continuous and contain no "kinks";
2. The cost functions are *strictly* convex, i.e., their gradients are strictly increasing;
3. The feasible set is convex.

The proofs of the robustness results of Section 6.4.2 require all these three assumptions. However, not all of them are necessary for the derivation and practical performance of ODDO. For instance, in Problem IM introduced in Section 6.5.2 the cost functions are not strictly convex; yet the evaluation results in Section 6.5.4 indicate that applying ODDO to Problem IM yields good on-line solutions for this problem. Thus, the question arises to what extent these assumptions are necessary for the validity and practical performance of ODDO. This is an important question since relaxing one or more of these assumptions would allow to apply ODDO to a much broader range of relevant optimization problems.

The goal of this chapter is to assess whether the above mentioned three assumptions are necessary for the validity and good practical performance of ODDO. In particular, we show how the original framework can be adapted to account for the omission of one or more of these assumptions. We do this by considering as leading examples two variants of the electric vehicle (EV) scheduling problem introduced in Section 3.4. Both variants are motivated by practical limitations with regard to the charging power. The first variant is motivated by the fact that EVs can often choose from a finite set of charging rates, i.e., the allowed charging rates are discrete instead of continuous. Moreover, it is not desirable to switch frequently between several charging rates within a short amount of time since it significantly increases the degradation of the EV battery [97]. The second variant is motivated by the facts that charging at a relatively low rate is not always possible due to restrictions on the minimum charging current set by EV battery manufacturers (see, e.g., [87, 182]) and generally reduces the efficiency of the charging process (see, e.g., [8]).

In the first variant, we replace the quadratic cost functions of the original EV scheduling problem with a piece-wise linear approximation of this function. This means that in this variant the assumptions of continuous differentiability and strict convexity are not satisfied. In the second variant, we add the restriction that the charging rate is either zero or lies within a given interval, meaning that charging at a rate between zero and a given threshold is not allowed. As a consequence, the feasible set of the problem is no longer convex, meaning that the third assumption is not satisfied by this model. We show for both variants how the original ODDO framework can be adapted so that it can be used to

solve the online versions of these variants. Moreover, we conduct simulation studies to assess the practical performance of this adapted version. These studies indicate that the online solution quality of this adapted version of ODDO is comparable to that of the original ODDO framework. This suggests that the three assumptions are not crucial for the practical application of ODDO, which significantly increases the number of problems to which it can be successfully applied.

The remainder of this chapter is organized as follows. In Section 8.2, we formulate the first variant of the EV scheduling problem, i.e., the variant with piece-wise linear cost functions, and describe the necessary adaptation of ODDO to solve the online version of this problem. In Section 8.3, we formulate the second variant, i.e., the variant wherein a minimum threshold is given, and present an alternative version of ODDO for this problem. Finally, Section 8.4 contains our conclusions and a brief outlook on the further applicability of ODDO.

8.2 PIECE-WISE LINEAR COST FUNCTIONS

In this section, we study a variant of the EV charging scheduling problem wherein the cost functions are piece-wise linear. We show how ODDO can be adapted to solve the online version of this problem. To this end, we first describe in Section 8.2.1 the practical motivation for this EV scheduling problem and provide a formal problem formulation. Subsequently, in Section 8.2.2, we describe a solution approach for the offline (i.e., deterministic) version of the problem that is similar to the sequential breakpoint search strategy for Problem QRA and Problem EV in Section 3.3.1. Based on this approach, we present in Section 8.2.3 an adaptation of ODDO for the online version of the problem. Finally, in Section 8.2.4, we conduct a simulation study to assess the performance of the proposed adaption of ODDO. We note that the analysis in this section can be generalized to EV charging problems with general convex cost functions (see [SU:5]). However, in this section, we focus on quadratic cost functions, which connects the analysis better to that of the other specific energy management problems studied in this thesis.

8.2.1 MOTIVATION AND PROBLEM FORMULATION

The EV charging scheduling problem was introduced in Section 3.4 and reformulated in terms of ODDO in Section 7.2.1:

$$\begin{aligned}
\text{EV : } \min_{x^1, \dots, x^T} \quad & \sum_{t \in \mathcal{T}} \frac{1}{2} (x^t + p^t)^2 \\
\text{s.t. } \quad & \sum_{t \in \mathcal{T}} x^t = \frac{\bar{R}}{\Delta t}, \\
& 0 \leq x^t \leq X_{\max}, \quad t \in \mathcal{T}.
\end{aligned}$$

As already mentioned, EVs can often charge only at a finite number of different rates that usually correspond to different user preferences such as "slow" or "fast" charging (see also [97]). To include this limitation in Problem EV, we replace the charging rate constraints $0 \leq x^t \leq X_{\max}$ by the constraint $x^t \in \mathcal{Z}$, where $\mathcal{Z} := \{z_0, \dots, z_m\}$ denotes the set of available charging rates. This leads to the following optimization problem for EV charging with discrete charging rates:

$$\begin{aligned}
\text{DiscreteEV : } \min_{x^1, \dots, x^T} \quad & \sum_{t \in \mathcal{T}} \frac{1}{2} (x^t + p^t)^2 \\
\text{s.t. } \quad & \sum_{t \in \mathcal{T}} x^t = \frac{\bar{R}}{\Delta t}, \\
& x_t \in \mathcal{Z}, \quad t \in \mathcal{T}.
\end{aligned}$$

If the EV can only be charged at one charging rate, i.e., if $\mathcal{Z} = \{0, Z\}$ for some $Z \in \mathbb{R}_{>0}$, we can solve Problem DiscreteEV easily by a greedy approach. More precisely, the intervals that are used for charging are those intervals for which the difference $\frac{1}{2}(Z + p^t)^2 - \frac{1}{2}(p^t)^2$ is smallest. However, when the EV can choose between multiple charging rates, Problem DiscreteEV is in general NP-hard [167]. Thus, it seems unlikely that we can derive an efficient, i.e., polynomial-time approach to solve this problem, let alone an efficient online approach in the style of ODDO.

Therefore, we consider a relaxation of Problem DiscreteEV that allows the EV to switch between charging rates within each interval. This means that now x^t represents a convex combination of the charging rates during a time interval t instead of a single rate that is used during the entire interval. We define the cost for this combined charging in interval t as the corresponding convex combination of the costs of using the individual charging rates within this interval. Since we aim to minimize the cost and the function $\frac{1}{2}(x^t + p^t)^2$ is convex, only two consecutive charging rates have to be considered in an optimal schedule as part of the combined charging rate x^t (see also [167]). More precisely, if $z_{j-1} < x^t < z_j$ for some $j \in \{1, \dots, m\}$, then only the charging rates z_{j-1} and z_j are used for charging during this interval and x^t is a convex combination of z_{j-1} and z_j . Thus, there exist nonnegative values y_{j-1}^t and y_j^t with $y_{j-1}^t + y_j^t = 1$ such that $x^t = y_{j-1}^t z_{j-1}^t + y_j^t z_j^t$ and the cost for interval t is given

by $\frac{1}{2}y_{j-1}^t(z_{j-1}+p^t)^2 + \frac{1}{2}y_j^t(z_j^t+p^t)$. Essentially, this means that we obtain a new cost function F^t that is piece-wise linear and that is obtained from $\frac{1}{2}(x^t + p^t)^2$ by linearizing it on each of the intervals $(z_0, z_1), (z_1, z_2), \dots, (z_{m-1}, z_m)$. We call the points $z_0, \dots, z_m$ the *breakpoints* of F^t , (see also the definition of breakpoints for Problem QRA in Section 3.2). Note, that the slope of the piece between two consecutive breakpoints z_{j-1} and z_j is given by

$$s_j^t := \frac{\frac{1}{2}(z_j + p^t)^2 - \frac{1}{2}(z_{j-1} + p^t)^2}{z_j - z_{j-1}} = \frac{1}{2}(z_j + z_{j-1}) + p^t, \quad j \in \{1, \dots, m\}. \quad (8.1)$$

For technical reasons, we define $s_0 := -\infty$ and $s_{m+1} := \infty$. We can now define the cost functions F^t of our relaxed problem as follows (this is also illustrated in Figure 8.1):

$$F^t(x^t) := \frac{1}{2}(z_{j-1} + p^t)^2 + s_j^t(x^t - z_{j-1}), \quad \text{if } z_{j-1} \leq x^t < z_j.$$

This implies that we can formulate a relaxation of Problem DiscreteEV as follows:

$$\begin{aligned} \text{RelaxEV : } & \min_{x^1, \dots, x^T} \sum_{t \in \mathcal{T}} F^t(x^t) \\ & \text{s.t. } \sum_{t \in \mathcal{T}} x^t = \frac{\bar{R}}{\Delta t}, \\ & \quad z_0 \leq x^t \leq z_m, \quad t \in \mathcal{T}. \end{aligned}$$

In the remainder of this section, we focus on solving Problem RelaxEV. We first sketch an approach from [167] to optimally solve Problem RelaxEV. Subsequently, in Section 8.2.3, we derive an online approach for Problem RelaxEV.

Figure 8.1: An example of a piece-wise linear function F^t obtained from linearizing $\frac{1}{2}(x^t + p^t)^2$.

Figure 8.2: The structure of an optimal solution $x(\lambda)$ to $\text{RelaxEV}(\lambda)$.

8.2.2 OFFLINE OPTIMIZATION APPROACH

For a given $\lambda \in \mathbb{R}$, the Lagrangian relaxation of Problem RelaxEV is given as

$$\begin{aligned} \text{RelaxEV}(\lambda) : \min_{x^1, \dots, x^T} \sum_{t \in \mathcal{T}} (F^t(x^t) - \lambda x^t) \\ \text{s.t. } z_0 \leq x^t \leq z_m, \quad t \in \mathcal{T}. \end{aligned}$$

For each $t \in \mathcal{T}$, let $\bar{j}^t(\lambda)$ be the unique charging rate index j for which $s_j^t - \lambda < 0 \leq s_{j+1}^t - \lambda$, i.e., for which λ is in between s_j^t and s_{j+1}^t . Then a solution $x(\lambda)$ is optimal for $\text{RelaxEV}(\lambda)$ if and only if it is of the following form:

$$x^t(\lambda) \begin{cases} := z_{\bar{j}^t(\lambda)} & \text{if } \lambda < s_{\bar{j}^t(\lambda)+1}^t, \\ \in [z_{\bar{j}^t(\lambda)}, z_{\bar{j}^t(\lambda)+1}^t] & \text{if } \lambda = s_{\bar{j}^t(\lambda)+1}^t. \end{cases} \quad (8.2)$$

(see also Figure 8.2). Note that, as opposed to the case of Problem QRA, the solution $x(\lambda)$ is unique if and only if $\lambda \neq s_j^t$ for all $t \in \mathcal{T}$ and $j \in \{1, \dots, m\}$.

Note that $x(\lambda)$ can be seen a piece-wise constant non-decreasing function of λ with discontinuities at the slope values in $\mathcal{S}^t := \{s_1^t, \dots, s_m^t\}$. This implies that also the sum $z(\lambda) := \sum_{t \in \mathcal{T}} x^t(\lambda)$ can be seen as a piece-wise constant non-decreasing function of λ with discontinuities at all slope values in $\mathcal{S} := \cup_{t \in \mathcal{T}} \mathcal{S}^t$.

For Problem QRA, we proved in Lemma 3.1 that if for a given $\lambda^* \in \mathbb{R}$ the optimal solution to the Lagrangian relaxation $\text{QRA}(\lambda^*)$ satisfies the resource constraint (3.1), this solution is also optimal to the original Problem QRA. This result also holds for Problem RelaxEV , i.e., if for a given $\lambda^* \in \mathbb{R}$ there exists an optimal solution to $\text{RelaxEV}(\lambda^*)$ of the form $x(\lambda)$ such that $z(\lambda) = \frac{\bar{R}}{\Delta t}$, then $x(\lambda)$ is also optimal for Problem RelaxEV . To see this, note that in the proof of Lemma 3.1 all arguments remain valid when we replace the original objective function of Problem QRA by *any* function $f(x^1, \dots, x^T)$.

This means that we can find an optimal solution to Problem RelaxEV by searching for an optimal Lagrange multiplier λ^* such that a solution of the form $x(\lambda^*)$

exists with $z(\lambda^*) = \sum_{t \in \mathcal{T}} x^t(\lambda^*) = \frac{\bar{R}}{\Delta t}$. We can find this Lagrange multiplier using an approach similar to the sequential breakpoint search procedure of Section 3.3.1 for Problem QRA. For this, observe that $z(\lambda)$ is a non-decreasing piece-wise constant function with discontinuities at the points in $\mathcal{S}$. Therefore, we have for any two consecutive slopes $\bar{s}_1, \bar{s}_2 \in \mathcal{S}$ that $x(\lambda)$ and thereby also $z(\lambda)$ is the same and uniquely defined for all $\lambda \in (\bar{s}_1, \bar{s}_2)$ (see Equation (8.2)). This implies that either

- » there exist two unique consecutive slope values $\bar{s}_1, \bar{s}_2 \in \mathcal{S}$ such that $z(\lambda) = \frac{\bar{R}}{\Delta t}$ for all $\lambda \in (\bar{s}_1, \bar{s}_2)$, or;
- » there exists a unique slope value s^* such that $z(\lambda_1) < \frac{\bar{R}}{\Delta t} < z(\lambda_2)$ for all λ_1, λ_2 with $\lambda_1 < s^* < \lambda_2$.

In both cases, we can find the desired slope values by sequentially searching through the set $\mathcal{S}$. More precisely, for a given $s \in \mathcal{S}$, we compute the maximum possible value Z of $z[s]$ and consider the following three options. If $Z < \frac{\bar{R}}{\Delta t}$, we know that s cannot be $\bar{s}_2$ or s^* and we continue the search. If $Z > \frac{\bar{R}}{\Delta t}$, we know that the first case applies and $\bar{s}_2 = s$, from which we can directly determine $\bar{s}_1$. Finally, if $Z = \frac{\bar{R}}{\Delta t}$, we know that the second case applies and $s = s^*$. In this second case, an optimal solution x^* to Problem RelaxEV is given by finding a solution of the form $x = (s^*)$ such that $z(s^*) = \frac{\bar{R}}{\Delta t}$. Exploiting the structure of this solution by means of Equation (8.2) yields the following general approach for computing x^* :

1. Set $(x^*)^t := z_{\bar{j}^t(s^*)}$ for all $t \in \mathcal{T}$.
2. Divide the remaining amount of resource $V := \frac{\bar{R}}{\Delta t} - \sum_{t \in \mathcal{T}} (x^*)^t$ somehow over the stages t with $s^* = s_j^t$ for some $j \in \{1, \dots, m\}$.

Observe that the division of the remaining amount of resource V can be done in various ways, e.g., proportionally according to the difference $z_{\bar{j}^t(s^*)+1} - z_{\bar{j}^t(s^*)}$. We propose to do this in a greedy way, i.e., such that we have $(x^*)^t = z_{\bar{j}^t(s^*)+1}$ for as many of these stages as possible. The reason for this is that the greedy division ensures that the charging rates within the optimal solution correspond to initial (discrete) charging rates in $\mathcal{Z}$ as much as possible. By doing this, we adhere as well as possible to the practical requirement of not switching too much between different charging rates. Moreover, it allows us to present our overall approach as a relatively straightforward greedy algorithm.

Algorithm 8.1 captures this sequential greedy slope search approach. During each iteration of the while-loop, we select the smallest slope that has not yet been used (Line 7), where we break ties by selecting the slope with the highest stage index. Subsequently, we increase the charging rate of the corresponding stage to the next rate (Lines 8 and 9). Notice that in this selection process, we only need to consider for each interval the slope with the lowest slope-index j that has not been used. This is because, for each $t \in \mathcal{T}$, we have $s_t^1 < s_t^2 < \dots <$

$s_t^{M_t}$ due to the convexity of the original function $\frac{1}{2}(x^t + p^t)^2$ (as illustrated in Figure 8.1). The slope selection process iterates until we cannot fully increase the charging rate of the selected interval to the next charging rate (this happens when $V < z_j - z_{j-1}$ in Line 8). In that case, we increase the power during this stage such that the original charging requirement $\frac{\bar{R}}{\Delta t}$ is precisely met.

Algorithm 8.1: Optimal greedy algorithm for Problem RelaxEV.

```

1 Input:  $p \in \mathbb{R}^n$ ,  $\bar{R} \in \mathbb{R}$ ,  $\Delta t \in \mathbb{R}_{>0}$ , set of charging rates  $\mathcal{Z}$ 
2 Output: Optimal solution  $x^*$  to Problem RelaxEV
3 Compute slopes  $s_j^t$  for all  $t \in \mathcal{T}$ ,  $j \in \{1, \dots, m\}$  using Equation (8.1)
4 Initialize ordered set  $\mathcal{S} := \{s_1^1, \dots, s_1^T\}$ 
5 Initialize  $x := 0^T$ ;  $V := \frac{\bar{R}}{\Delta t}$ 
6 while  $V > 0$  do
7   Take first slope value  $s_j^t$  from  $\mathcal{S}$ 
8    $Q = \min(V, z_j - z_{j-1})$ 
9    $x^t = x^t + Q$ 
10   $V = V - Q$ 
11   $\mathcal{S} = \mathcal{S} \setminus \{s_j^t\}$ 
12  if  $j < m$  then
13    Insert  $s_{j+1}^t$  into the ordered set  $\mathcal{S}$ 
14 return  $x^* = x$   $\triangleright \lambda^*$  is lastly removed slope value  $s_j^t$ 

```

An important property of the optimal solution computed by Algorithm 8.1 is that $x^t \notin \mathcal{Z}$ for at most one stage t due to the greedy slope choice. This means that in the optimal solution, there is at most one interval wherein the EV switches between two charging rates. This follows directly from Algorithm 8.1 since in Line 8, we have $Q = z_j - z_{j-1}$ and thus $x^t = z_j$ unless $V < z_j - z_{j-1}$. In that case, $Q = V$ and thus at the end of the current iteration we get $V = 0$, hence the while-loop terminates.

8.2.3 ONLINE OPTIMIZATION APPROACH

In this section, we describe how the concept of ODDO can be applied to obtain an algorithm for the online version of Problem RelaxEV. For this, we use the analysis of the previous section regarding the structure of optimal solutions to this problem and to its Lagrangian relaxation RelaxEV(λ).

The local Lagrangian dual function for Problem RelaxEV is given by

$$q_{\text{RelaxEV}}^t(\lambda) : \min_{x^t} (F^t(x^t) - \lambda x^t \mid z_0 \leq x^t \leq z_m).$$

Since the cost functions F^t are not *strictly* convex, the solution to the inner problem of q_{RelaxEV}^t might not be unique. In line with the robustness results for

ODDO in Section 6.4.2 in general and those for Problem EV in Section 7.2.1 in particular, we choose to select the maximum out of all optimal solutions to the inner problem. This means that we define the local Lagrangian solution as

$$x^t(\lambda) := \begin{cases} z_{\bar{j}^t(\lambda)} & \text{if } \lambda < s_{\bar{j}^t(\lambda)+1}^t, \\ z_{\bar{j}^t(\lambda)+1} & \text{if } \lambda = s_{\bar{j}^t(\lambda)+1}^t. \end{cases} \quad (8.3)$$

Note that, for a prediction $\hat{\lambda}$ of λ^* , the projection sets $\mathcal{C}_{\text{proj}}^{\bar{t}}$ (see Equation (6.3)) are analogous to those of Problem EV in Section 7.2.1 and given by

$$\mathcal{C}_{\text{proj}}^{\bar{t}} := \left[\max(z_0, \hat{R}^{\bar{t}}(\hat{\lambda}) - (T - \bar{t})z_m), \min(z_m, \hat{R}^{\bar{t}}(\hat{\lambda}) - (T - \bar{t})z_0) \right],$$

where $\hat{R}^{\bar{t}}(\hat{\lambda}) := \frac{\bar{R}}{\Delta \bar{t}} - \sum_{t=1}^{\bar{t}-1} \hat{x}^t(\hat{\lambda})$. Combining this with the definition of the local Lagrangian solution in Equation (8.3), we obtain the following expression for the online solution $\hat{x}^t(\hat{\lambda})$ to Problem RelaxEV given a prediction $\hat{\lambda}$ of λ^* :

$$\hat{x}^{\bar{t}}(\hat{\lambda}) := \max \left(\hat{R}^{\bar{t}}(\hat{\lambda}) - (T - \bar{t})z_m, \min \left(\hat{R}^{\bar{t}}(\hat{\lambda}) - (T - \bar{t})z_0, \hat{x}^{\bar{t}}(\hat{\lambda}) \right) \right). \quad (8.4)$$

Observe that this expression is similar to that for the online solution for Problem EV in Equation (7.2). One major difference is that when using an optimal Lagrange multiplier as input for these expressions, i.e., if $\hat{\lambda} = \lambda^*$, the online solution for Problem EV is also optimal, whereas we do not have this guarantee for Problem RelaxEV. The reason for this is that the local Lagrangian solution in Equation (8.3) is not necessarily the unique optimal solution to the inner problem of the local Lagrangian dual function. More precisely, we only have the guarantee that $\hat{x}(\lambda^*)$ is optimal if the charging rates in the corresponding optimal solution are all in the (discrete) charging rate set $\mathcal{Z}$. In practice, such a solution will hardly ever exist. Nonetheless, we expect that the impact of this non-uniqueness on the eventual online solution is minimal since the freedom of choice for the local Lagrangian solution is only present when $\hat{\lambda}$ is exactly equal to one of the slope values in $\mathcal{S}$.

8.2.4 EVALUATION

In this section, we evaluate the performance of the online approach presented in the previous section by means of a simulation study. The evaluation consists of two parts. First, we study the robustness of the approach against prediction errors in the optimal Lagrange multiplier λ^* . Second, we investigate how well we can predict λ^* using historical data.

We first describe the general setup of these simulations. We apply our online approach to schedule the charging of an EV between 18:00h and 7:00h. We divide this charging window into 15-minute time intervals, meaning that $\Delta t = \frac{1}{4}$. Usually, residential charging is done at either “Level 1” (1.9 kW) or “Level 2”

(4 kW) [181]. We assume that our EV can switch between these charging rates and no charging at all, meaning that we choose $\mathcal{Z} = \{0, 1900, 4000\}$. To study the effect of the charging requirement on our approach, we consider values of $\bar{R}$ that represent charging requirements of 5, 10, $\dots$, 45, 50 kWh. We denote these values of $\bar{R}$ by the set $\mathcal{R} := \{5000; 10,000; \dots; 45,000; 50,000\}$. Finally, for the base load profile, we use the same consumption measurement data as used in the simulation study for the online version of Problem EV in Section 7.3.1.

Robustness study

First, we study the robustness of the online solution given by Equation (8.4). For this, we apply our approach to a single charging session for different input predictions $\hat{\lambda}$. Figure 8.3 shows the used base load profile for the corresponding charging window. We compare the quality of each online solution $\hat{x}(\hat{\lambda})$ to that of the optimal solution x^* by computing the ratio between their objective values. An online solution $\hat{x}(\hat{\lambda})$ is considered a good approximation of x^* if this ratio is close to 1 (if the ratio is 1, then $\hat{x}(\hat{\lambda})$ is optimal) and, generally, the online solution in Equation (8.4) is considered robust if a large change in the input prediction $\hat{\lambda}$ leads to only a small change in this ratio.

Figure 8.4 shows the results of the simulations as sensitivity to the prediction $\hat{\lambda}$. More precisely, Figure 8.4 shows for each of the charging requirements in the set $\mathcal{R}$ the ratio between the online and optimal offline solution for different input predictions $\hat{\lambda}$ (lines) and, as a reference point, the maximum value among all multiplier values λ^* that are optimal (dots). For $\bar{R} \in \{5,000; 10,000; 15,000\}$, the ratio increases significantly when $\hat{\lambda}$ is slightly smaller than λ^* , whereas it barely increases when $\hat{\lambda}$ is slightly larger than λ^* . On the other hand, for $\bar{R} \in \{20,000; \dots; 50,000\}$, this effect is reversed: when $\hat{\lambda}$ is slightly smaller than λ^* , the ratio barely increases, whereas it increases greatly when $\hat{\lambda}$ is slightly larger than λ^* . A possible explanation for this is that λ^* is relatively high for charging requirements of 20 kWh and higher. More precisely, λ^* is close to the slope

Figure 8.3: The base load profile used in the robustness study in Section 8.2.4.

values of the stages before 21:00h, i.e., the stages with a relatively high base load. As a consequence, when $\hat{\lambda}$ is slightly larger than λ^* , more of these “expensive” stages will be used for charging in the online solution. On the other hand, for charging requirements of 15 kWh and lower, λ^* is relatively low and close to the slopes of the stages after 21:00h, i.e., the stages with a relatively low base load. Thus, when $\hat{\lambda}$ is slightly larger than λ^* , only these “cheap” stages will be used for charging in the online solution.

Nevertheless, Figure 8.4 shows that in almost all cases there is a wide range of values of $\hat{\lambda}$ for which the ratio is very small. This suggests that, although λ^* lies at the boundary of this range, a larger deviation of $\hat{\lambda}$ from λ^* can still yield a good online solution. The main explanation for this behavior lies in the piece-wise nature of the decision rule in Equation (8.4), which in turn is a consequence of the fact that the piece-wise linear function F_t is not continuously differentiable. It is remarkable that the latter fact seems to improve the robustness of the online algorithm, since in many other cases dropping the assumption of continuous differentiability often increases the difficulty of solving optimization problems (see, e.g., [129]).

*Predictability of λ^**

Next to the robustness of our online approach, we also assess whether it is possible to predict the maximum optimal Lagrange multiplier λ^* based on historical values of λ^* for (fictional) past charging sessions. For this, we simulate 100 charging sessions on consecutive days for each charging requirement in $\mathcal{R}$ and compute for each session the corresponding λ^* . As input for these simulations, we use actual base load data of the household for 100 consecutive days.

Figure 8.5 shows that for most of the charging requirements, the corresponding values of λ^* are very close together, which implies that these historical values of λ^* accurately represent the behavior of λ^* for future days. However, for $\bar{R} \in \{20,000; 45,000; 50,000\}$, the difference between these values is significantly larger. For $\bar{R} = 20,000$, this is because the share of charging on “expensive” stages, i.e., the stages corresponding to the evening demand peak, varies more over the course of the days since for these charging requirements almost all stages after this peak are used for charging. For $\bar{R} \in \{45,000; 50,000\}$, the reason for the large variance is that the share of expensive stages (the stages before the “night dip”) is relatively large for these charging requirements since the load of these stages varies more than the base load during the cheap stages, i.e., the base load during the night.

8.3 MINIMUM CHARGING THRESHOLD

In this section, we study a variation of the original Problem EV with an additional minimum threshold on the allowed charging rate. First, in Section 8.3.1,

(a) Ratios for $\bar{R} \in \{5,000; 10,000; 15,000; 20,000\}$. (b) Ratios for $\bar{R} \in \{25,000; 30,000; 35,000; 40,000; 45,000; 50,000\}$.

Figure 8.4: Ratios between the online and optimal offline objective value for each of the charging requirements in $\mathcal{R}$ for different choices of $\hat{\lambda}$. The dots represent the maximum value among all corresponding multiplier values λ^* that are optimal.

(a) Values of λ^* for $\bar{R} \in \{5,000; 10,000; 15,000; 20,000\}$.

(b) Values of λ^* for $\bar{R} \in \{25,000; 30,000; 35,000; 40,000; 45,000; 50,000\}$.

Figure 8.5: Values of λ^* for 100 days for each of the charging requirements in $\mathcal{R}$.

we formulate this problem and analyze its computational complexity. Subsequently, in Section 8.3.2, we present an efficient approach and algorithm to solve the offline version of this problem. Based on this approach, we present in Section 8.3.3 an alternative version of ODDO to solve the online version of the problem. Finally, we evaluate the performance of this approach in Section 8.3.4.

8.3.1 PROBLEM FORMULATION

In practice, mostly a lower limit $X_{\min}$ on the possible charging rates is given. To include this restriction of charging either nothing or at a certain rate within the interval $[X_{\min}, X_{\max}]$, we introduce a binary variable y^t that is 1 if charging is done in stage t and 0 otherwise. This leads to the following optimization problem:

$$\begin{aligned} \text{ThresholdEV : } \min_{\substack{x^1, \dots, x^T, \\ y^1, \dots, y^T}} \quad & \sum_{t \in \mathcal{T}} \frac{1}{2} (x^t + p^t)^2 \\ \text{s.t. } \quad & \sum_{t \in \mathcal{T}} x^t = \frac{\bar{R}}{\Delta t}, \\ & X_{\min} y^t \leq x^t \leq X_{\max} y^t, \quad t \in \mathcal{T}, \\ & y^t \in \{0, 1\}, \quad t \in \mathcal{T}. \end{aligned} \quad (8.5)$$

Throughout this section, we assume without loss of generality that the stages are sorted according to non-increasing values of p^t , i.e., we assume that $p^1 \geq p^2 \geq \dots \geq p^T$. Moreover, we assume that $X_{\max} \geq 2X_{\min}$, i.e., the range of admissible charging rates is sufficiently large. This last assumption implies that Problem ThresholdEV is feasible for any $\frac{\bar{R}}{\Delta t} \in [X_{\min}, TX_{\max}]$. Note that this assumption does not exclude the mentioned technical limitations of EV batteries since the minimum allowed charging rate is usually around 1.38 (kW) and the maximum charging rate is at least 3.3 (kW) for virtually every EV (see, e.g., [77, 182]).

We note that all analyses, algorithms, and online optimization approaches in this section can be generalized to the case where the maximum charging rates $X_{\max}$ are stage-dependent, i.e., we have $X_{\min} y^t \leq x^t \leq X_{\max}^t y^t$ for all $t \in \mathcal{T}$, provided that $X_{\max}^t \geq 2X_{\min}$ (see [SU:8]). Moreover, they can also be generalized to the case where the objective function is $(0, p, f)$ -separable for some convex function f , i.e., where we replace each term $\frac{1}{2}(x^t + p^t)^2$ of the objective function by $f(x^t + p^t)$ for a given convex function f (see also Section 2.2.2). However, similarly to the analysis of EV charging problems with piecewise linear cost functions in Section 8.2, we restrict ourselves to the case where $X_{\max}^t$ is the same for all $t \in \mathcal{T}$ in order to connect the problem more easily to the original Problem EV.

Problems of type ThresholdEV occur in the literature as knapsack problems with setups [117] or semi-continuous knapsack problems [162]. However, in

both cases, the cost functions are generally assumed to be linear. In contrast, little to no literature exists on these types of problems with quadratic objective functions, except for a preliminary analysis of Problem ThresholdEV in [116].

8.3.2 OFFLINE OPTIMIZATION APPROACH

We first analyze the structure of Problem ThresholdEV and its optimal solutions. We show that solving Problem ThresholdEV can be reduced to solving a specific sequence of instances of Problem EV. Based on this reduction, we first derive a solution approach that runs in $O(T^2)$ time. Subsequently, we derive an $O(T \log T)$ time algorithm for the problem by following an approach similar to that in Chapter 4 to solve Problem QRA-NC. More precisely, we apply the sequential breakpoint search procedure from Section 3.3.1 to solve the QRA subproblems, whereby we do not solve each subproblem from scratch but use the optimal Lagrange multiplier of the “preceding” subproblem as a starting point for the multiplier search in the current QRA subproblem.

We start by showing an important property of optimal solutions to Problem ThresholdEV, namely that there exists an optimal solution (x^*, y^*) to Problem ThresholdEV in which x^* is a non-decreasing vector:

Lemma 8.1. *There exists an optimal solution (x^*, y^*) to Problem ThresholdEV with $(x^*)^s \leq (x^*)^t$ for all $s, t \in \mathcal{T}$ with $s < t$.*

Proof. Suppose that for an optimal solution (x^*, y^*) to Problem ThresholdEV there exist stages $\bar{s}, \bar{t}$ such that $\bar{s} < \bar{t}$ and $(x^*)^{\bar{s}} > (x^*)^{\bar{t}}$. Let $(\bar{x}, \bar{y})$ denote the solution obtained from (x^*, y^*) by switching the values of stages $\bar{s}$ and $\bar{t}$, i.e., $(\bar{x}^{\bar{s}}, \bar{y}^{\bar{s}}) = ((x^*)^{\bar{t}}, (y^*)^{\bar{t}})$, $(\bar{x}^{\bar{t}}, \bar{y}^{\bar{t}}) = ((x^*)^{\bar{s}}, (y^*)^{\bar{s}})$, and $(\bar{x}^t, \bar{y}^t) = ((x^*)^t, (y^*)^t)$ for all $t \in \mathcal{T} \setminus \{\bar{s}, \bar{t}\}$. Then we obtain the following relation between the objective values of these solutions:

$$\begin{aligned}
 & \sum_{t \in \mathcal{T}} \frac{1}{2} ((x^*)^t + p^t)^2 - \sum_{t \in \mathcal{T}} \frac{1}{2} (\bar{x}^t + p^t)^2 \\
 &= \frac{1}{2} \left(((x^*)^{\bar{s}} + p^{\bar{s}})^2 + ((x^*)^{\bar{t}} + p^{\bar{t}})^2 - (\bar{x}^{\bar{s}} + p^{\bar{s}})^2 - (\bar{x}^{\bar{t}} + p^{\bar{t}})^2 \right) \\
 &= \frac{1}{2} \left(((x^*)^{\bar{s}} + p^{\bar{s}})^2 + ((x^*)^{\bar{t}} + p^{\bar{t}})^2 - ((x^*)^{\bar{t}} + p^{\bar{s}})^2 - ((x^*)^{\bar{s}} + p^{\bar{t}})^2 \right) \\
 &= (x^*)^{\bar{s}}(p^{\bar{s}} - p^{\bar{t}}) + (x^*)^{\bar{t}}(p^{\bar{t}} - p^{\bar{s}}) \\
 &\geq (x^*)^{\bar{t}}(p^{\bar{s}} - p^{\bar{t}}) + (x^*)^{\bar{t}}(p^{\bar{t}} - p^{\bar{s}}) \\
 &= 0.
 \end{aligned}$$

This implies that $(\bar{x}, \bar{y})$ must be an optimal solution since its objective value is not larger than that of (x^*, y^*) . This argument can be repeated until we have arrived at a feasible solution (x, y) with $x^1 \leq \dots \leq x^T$ that is by construction also optimal $\square$

For a given $K \leq T$, let $\text{ThresholdEV}(K)$ denote the problem of solving Problem ThresholdEV with the additional constraint that no charging is done in exactly K stages, i.e., $\sum_{t \in \mathcal{T}} y^t = T - K$. Lemma 8.1 implies that we can find an optimal solution x^* to this problem, assuming that it has a feasible solution, by setting $(y^*)^t = 0$ and $(x^*)^t = 0$ for $t \leq K$, setting $(y^*)^t = 1$ for $t > K$, and solving the remaining problem as an instance of Problem EV :

$$\begin{aligned} \text{ThresholdEV}(K) : \quad & \min_{x^{K+1}, \dots, x^T} \sum_{t=K+1}^T \frac{1}{2} (x^t + p^t)^2 \\ \text{s.t.} \quad & \sum_{t=K+1}^T x^t = \frac{\bar{R}}{\Delta t}, \\ & X^{\min} \leq x^t \leq X_{\max}, \quad t \in \{K+1, \dots, T\}. \end{aligned}$$

Problem $\text{ThresholdEV}(K)$ can be solved in $O(T - K)$ time using the binary search procedure for Problem QRA in Algorithm 3.2. Thus, we can solve Problem ThresholdEV in $O(T^2)$ time by computing a solution to Problem $\text{ThresholdEV}(K)$ for each $K \in \{0, \dots, T-1\}$ and, from these solutions, choose the solution with the smallest objective value.

We can improve upon this approach using an approach similar to the fast sequential algorithm for Problem QRA-NC in Chapter 4. More precisely, instead of solving each problem $\text{ThresholdEV}(K)$ from scratch, we apply the sequential breakpoint search procedure of Section 3.3.1 and use the optimal Lagrange multiplier of the “preceding” subproblem $\text{ThresholdEV}(K-1)$ as a starting point for the sequential breakpoint search.

For this, note that for a given $K \in \{0, \dots, T-1\}$, $\text{ThresholdEV}(K)$ can be written as an instance of Problem QRA by the variable transform $\bar{z} = x + p$ and choosing $a^t = 1$, $l^t = X_{\min} + p^t$, $u^t = X_{\max} + p^t$ for $t \in \{K+1, \dots, T\}$ and $R = \frac{\bar{R}}{\Delta t} + \sum_{t=K+1}^T p^t$. Thus, we can write $\text{ThresholdEV}(K)$ as

$$\begin{aligned} \min_{\bar{z}^{K+1}, \dots, \bar{z}^T} \quad & \sum_{t=K+1}^T \frac{1}{2} (\bar{z}^t)^2 \\ \text{s.t.} \quad & \sum_{t=K+1}^T \bar{z}^t = \frac{\bar{R}}{\Delta t} + \sum_{t=K+1}^T p^t, \\ & X^{\min} + p^t \leq \bar{z}^t \leq X_{\max} + p^t, \quad t \in \{K+1, \dots, T\}. \end{aligned}$$

It follows that the solution to the Lagrangian relaxation of $\text{ThresholdEV}(K)$ for a given Lagrange multiplier $\lambda \in \mathbb{R}$ is given by

$$\bar{z}_K^t(\lambda) = \begin{cases} X_{\min} & \text{if } \lambda - p^t < X_{\min}, \\ \lambda - p^t & \text{if } 0 \leq \lambda - p^t < X_{\max}, \\ X_{\max} & \text{if } X_{\max} \leq \lambda - p^t, \end{cases}$$

where $t \in \{K+1, \dots, T\}$. Note that $\bar{z}_K^t(\lambda)$ is independent of K ; In particular, the lower and upper breakpoints are independent of K , meaning that for each $t \in \mathcal{T}$ the breakpoints are the same for all problems $\text{ThresholdEV}(K)$ with $K < t$. Hence, we can drop the subscript K from $\bar{z}_K$. Furthermore, given an *optimal* Lagrange multiplier λ^K of $\text{ThresholdEV}(K)$, the optimal solution to $\text{ThresholdEV}(K)$ is given by $(\bar{z}^{K+1}(\lambda^K), \dots, \bar{z}^T(\lambda^K))$. Finally, note that

$$\sum_{t=K+2}^T \bar{z}^t(\lambda^{K+1}) = \frac{\bar{R}}{\Delta t} + \sum_{t=K+2}^T p^t = \sum_{t=K+1}^T \bar{z}^t(\lambda^K) - p^{K+1} > \sum_{t=K+2}^T \bar{z}^t(\lambda^K).$$

Since each $\bar{z}^t(\lambda)$ is non-decreasing in λ , this implies that $\lambda^{K+1} > \lambda^K$ for all $K \in \{0, \dots, T-2\}$.

The above analysis implies that we can find all the optimal Lagrange multipliers $\lambda^0, \dots, \lambda^{T+1}$ together using the sequential breakpoint search strategy from Section 3.3.1. Hereby, we do not start each breakpoint search from scratch, but instead use the previous multiplier as a starting point for the current multiplier. This requires two main alterations to the original sequential breakpoint search procedure in Algorithm 3.1:

- » After a given optimal Lagrange multiplier λ^K has been found, the required resource R must be adjusted for the next problem $\text{ThresholdEV}(K)$ by subtracting p^{K+1} from the current amount of resource.
- » If the breakpoints α^{K+1} or β^{K+1} have not been considered yet, they must be removed from the breakpoint sets $\mathcal{A}$ or $\mathcal{B}$ respectively.

The resulting procedure is given by Algorithm 8.2. In this algorithm, the updating of the resource amount R is done in Line 19 and the removal of the breakpoints α^{K+1} and β^{K+1} in Lines 20-25.

Moreover, two additional adjustments can be included to reduce the time needed to determine the value of K for which the objective value of the optimal solution to $\text{ThresholdEV}(K)$ is smallest among all K :

- » We limit K to the interval $[K_1, K_2]$ with $\bar{K}_1 := \max\left(0, T - \lfloor \frac{\bar{R}}{\Delta t} \frac{1}{X_{\min}} \rfloor\right)$ and $\bar{K}_2 := \min\left(T-1, T - \lceil \frac{\bar{R}}{\Delta t} \frac{1}{X_{\max}} \rceil\right)$. These values represent the maximum and minimum number of stages where charging must be done for any feasible solution to Problem ThresholdEV respectively. In other words, $\text{ThresholdEV}(K)$ is feasible if and only if $K \in [K_1, K_2]$.
- » To efficiently compute for a given $K \in [K_1, K_2]$ the objective value of the optimal solution to $\text{ThresholdEV}(K)$, we keep track of the sum

$$N(\lambda) := \sum_{t > K: \lambda < \alpha^t} (X_{\min} + p^t)^2 + \sum_{t > K: \lambda \geq \beta^t} (X_{\max} + p^t)^2,$$

and update this value each time a new breakpoint has been considered or when λ^K has been found and K is increased. Note that by construction of the bookkeeping sum $Q(\lambda)$ (see also Section 3.3.1), the objective value of the solution $\bar{z}(\lambda)$ to the Lagrangian relaxation of ThresholdEV(K) is given by $N(\lambda) + Q(\lambda)\lambda^2$.

Analogously to Algorithm 3.1, the worst-case time complexity of the algorithm is $O(T \log T)$ since each iteration of the repeat-statement takes $O(1)$ time and is executed at most $2T$ times. Moreover, all operations outside of the repeat-statement can be executed in $O(T)$ time except for the initial sorting of p , which takes $O(T \log T)$ time.

8.3.3 ONLINE OPTIMIZATION APPROACH

In this section, we use the analysis of Section 8.3.2 to derive an algorithm for solving the online version of Problem ThresholdEV, when the vector p is unknown beforehand. Since the feasible set of Problem ThresholdEV is not convex, we are not guaranteed that an optimal Lagrange multiplier exists. More precisely, we are not guaranteed that the optimal solution to the inner problem of the Lagrangian dual function is optimal for the original problem. This means that we require an alternative characterization of an optimal solution to Problem ThresholdEV. We give such a suitable characterization using the analysis of the previous subsection and use this characterization to derive an alternative algorithm to solve the online version of Problem ThresholdEV.

The analysis in the previous subsection implies that an optimal solution to Problem ThresholdEV is characterized by two values, namely

- » the number K of stages where no charging is done, i.e., where $x^t = y^t = 0$ in the optimal solution, and
- » the optimal Lagrange multiplier λ^K of the corresponding subproblem Threshold(K).

Note that, given K , we have for the corresponding solution (x, y) that

$$y^t := \begin{cases} 0 & \text{if } t < K, \\ 1 & \text{if } t \geq K. \end{cases} \quad (8.6)$$

Observe that this characterization depends explicitly on the stage index t and thereby on the order of the elements of the vector p . Recall that we assumed for the derivation of the offline optimization approach in Section 8.3.2 that p is a non-increasing vector. However, in the online setting, this assumption is not valid: we do not know on forehand the order of p . This is a major disadvantage of this characterization with regard to an online optimization approach for Problem ThresholdEV. Therefore, we propose a slightly different characterization of

Algorithm 8.2: An $O(n \log n)$ time algorithm for Problem ThresholdEV.

```

1 Input: Parameters  $p \in \mathbb{R}^n$ ,  $\bar{R} \in \mathbb{R}$ , and  $\Delta t, X_{\min}, X_{\max} \in \mathbb{R}_{>0}^n$ 
2 Output: Optimal solution  $(\bar{x}, \bar{y})$  to Problem ThresholdEV
3 Sort  $p$  non-descendingly and permute  $\mathcal{T}$  accordingly
4 Initialize  $\mathcal{A} := \{X_{\min} + p^t \mid t \in \mathcal{T}\}; \mathcal{B} := \{X_{\max} + p^t \mid t \in \mathcal{T}\};$ 
    $\bar{K}_1 := \max\left(0, T - \lfloor \frac{\bar{R}}{\Delta t} \frac{1}{X_{\min}} \rfloor\right); \bar{K}_2 := \min\left(T - 1, T - \lceil \frac{\bar{R}}{\Delta t} \frac{1}{X_{\max}} \rceil\right)$ 
5  $P := (T - \bar{K}_1)X_{\min}; Q := 0; N = \sum_{t=1}^{\bar{K}_1} (p^t)^2 + \sum_{t=\bar{K}_1+1}^T (X_{\min} + p^t)^2;$ 
    $R := \frac{\bar{R}}{\Delta t} + \sum_{t \in \mathcal{T}} p^t$ 
6 for  $K = \bar{K}_1, \dots, \bar{K}_2$  do
7   repeat
8     Determine smallest breakpoint  $\delta^i := \min(\mathcal{A} \cup \mathcal{B})$ 
9     if  $P + Q\delta^t = R$  then
10        $\lambda^K = \delta^t$ 
11     else if  $P + Q\delta^t > R$  then  $\triangleright \lambda^K < \delta^t$ 
12        $\lambda^K = \frac{R-P}{Q}$ 
13     else  $\triangleright \lambda^K > \delta^t$ 
14       if  $\delta^t$  is lower breakpoint ( $\delta^t = \alpha^t$ ) then
15          $P := P - X_{\min} - p^t; Q := Q + 1;$ 
16          $N = N - (X_{\min} + p^t)^2; \mathcal{A} := \mathcal{A} \setminus \{\alpha^t\}$ 
17       else
18          $P := P + X_{\max} + p^t; Q := Q - 1;$ 
19          $N = N + (X_{\max} + p^t)^2; \mathcal{B} := \mathcal{B} \setminus \{\beta^t\}$ 
20     until multiplier  $\lambda^K$  has been found
21      $V^K := N + Q(\lambda^K)^2; R = R - p^{K+1}$ 
22     if  $\lambda^K \leq \alpha^{K+1}$  then
23        $N = N - (X_{\min} + p^{K+1})^2 + (p^{K+1})^2; P = P - X_{\min} - p^{K+1};$ 
24        $\mathcal{A} = \mathcal{A} \setminus \{\alpha^{K+1}\}; \mathcal{B} = \mathcal{B} \setminus \{\beta^{K+1}\}$ 
25     else if  $\alpha^{K+1} < \lambda^K \leq \beta^{K+1}$  then
26        $N = N + (p^{K+1})^2; Q = Q - 1; \mathcal{B} = \mathcal{B} \setminus \{\beta^{K+1}\}$ 
27     else
28        $N = N - (X_{\max} + p^{K+1})^2 + (p^{K+1})^2; P = p - X_{\max} - p^{K+1}$ 
29   Determine  $K^* := \arg \min_{\bar{K}_1 \leq K \leq \bar{K}_2} V^K$ 
30   for  $K = 1, \dots, K^*$  do
31      $\bar{x}^t = 0; \bar{y}^t = 0$ 
32   for  $K = K^* + 1, \dots, T$  do
33      $\bar{x}^t = \bar{z}^t(\lambda_K) - p^t; \bar{y}^t = 1$ 
34   Reverse index ordering of  $\mathcal{T}$  and  $(\bar{x}, \bar{y})$ 
35 return  $(\bar{x}, \bar{y})$ 

```

the idle stages that is almost equivalent to that in Equation (8.6):

$$y^t := \begin{cases} 0 & \text{if } X_{\min} + p^t \geq Z^{\text{act}}, \\ 1 & \text{if } X_{\min} + p^t < Z^{\text{act}}, \end{cases}$$

where $Z^{\text{act}} := X_{\min} + p^K$. In terms of EV charging, the value Z^{act} can be interpreted as an activation threshold: in a given stage $t \in \mathcal{T}$ we only charge if the minimum resulting combined load $X_{\min} + p^t$ "stays below" Z^{act} . The main difference between this characterization and that in Equation (8.6) is that the current characterization does not depend on the stage index t . Thus, if the value Z^{act} is known, we can use this characterization to compute y^t without using knowledge of any stage other than t . Moreover, both characterization rules are equivalent if all elements of p are distinct, meaning that computing y using either of the characterizations yields the same result. Note that in practice it is highly unlikely that two elements of p are exactly the same since they are measurements of power consumption.

From the above analysis, we may conclude that we can adequately characterize an optimal solution $(\bar{x}, \bar{y})$ to Problem ThresholdEV wherein $\bar{x}^t = 0$ for K stages by the two values $Z^{\text{act}} = X_{\min} + p^K$ and $Z^{\text{mult}} := \lambda^K$. More generally, we define for any two values Z^{act} and Z^{mult} the following solution:

$$y^t(Z^{\text{act}}, Z^{\text{mult}}) := \begin{cases} 0 & \text{if } X_{\min} + p^t \geq Z^{\text{act}}, \\ 1 & \text{if } X_{\min} + p^t < Z^{\text{act}}, \end{cases} \quad (8.7a)$$

$$x^t(Z^{\text{act}}, Z^{\text{mult}}) := \begin{cases} 0 & \text{if } y^t(Z^{\text{act}}, Z^{\text{mult}}) = 0, \\ X_{\min} & \text{if } y^t(Z^{\text{act}}, Z^{\text{mult}}) = 1 \text{ and } Z^{\text{mult}} - p^t < X_{\min}, \\ Z^{\text{mult}} - p^t & \text{if } y^t(Z^{\text{act}}, Z^{\text{mult}}) = 1 \\ & \text{and } 0 \leq Z^{\text{mult}} - p^t < X_{\max}, \\ X_{\max} & \text{if } y^t(Z^{\text{act}}, Z^{\text{mult}}) = 1 \text{ and } X_{\max} \leq Z^{\text{mult}} - p^t. \end{cases} \quad (8.7b)$$

Under the (practically justified) assumption that all elements of p are distinct, we have that the solution $(x(Z^{\text{act}}, Z^{\text{mult}}), y(Z^{\text{act}}, Z^{\text{mult}}))$ is equal to the optimal solution $(\bar{x}, \bar{y})$. Note that this is similar to the optimality of the Lagrangian solution $x(\mu^*, \lambda^*)$ for the optimal Lagrange multipliers (μ^*, λ^*) within ODDO. Moreover, note that for each $t \in \mathcal{T}$, the determination of $\bar{x}^t$ and $\bar{y}^t$ using this characterization does not require any elements of p other than p^t . This property is similar to the key observation of ODDO, namely that in certain cases the components $\bar{x}^t$ of an optimal solution $\bar{x}$ to a convex optimization problem can be reconstructed from the optimal Lagrange multipliers and using parameters that depend only on the stage t .

It follows that we can solve the online version of Problem ThresholdEV in the same spirit as ODDO by predicting Z^{act} and Z^{mult} and using the resulting

predictions $\hat{Z}^{\text{act}}$ and $\hat{Z}^{\text{mult}}$ as input for the characterization in Equation (8.7). One issue that must be resolved is how we can ensure that the resulting online solution $(x(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}), y(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}))$ is feasible. Recall that within ODDO we introduced the concept of a projection set, which for a given stage $t \in \mathcal{T}$ contains all allowed values for the current stage decision x^t such that there still exists a feasible solution for the future stages. Since we assumed that $X_{\max} \geq 2X_{\min}$ and thus Problem ThresholdEV is feasible for any $\frac{\bar{R}}{\Delta t} \in \{0\} \cup [X_{\min}, TX_{\max}]$, we can construct these sets for Problem ThresholdEV in a similar way as for Problem EV and Problem RelaxEV. More precisely, for a given stage $\bar{t} \in \mathcal{T}$, let $(\hat{x}^1(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}), \dots, \hat{x}^{\bar{t}-1}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}))$ denote the online solution for stages 1 to $\bar{t} - 1$ for given predictions $\hat{Z}^{\text{act}}$ and $\hat{Z}^{\text{mult}}$. Then the projection set for stage $\bar{t}$ is

$$\begin{aligned} \mathcal{C}_{\text{proj}}^{\bar{t}} := & (\{0\} \cup [X_{\min}, X_{\max}]) \cap \\ & ([\hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}) - (T - \bar{t})X_{\max}, \hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}) - X_{\min}] \\ & \cup \{\hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})\}), \end{aligned} \quad (8.8)$$

where $\hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}) := \frac{\bar{R}}{\Delta t} - \sum_{t=1}^{\bar{t}-1} \hat{x}^t(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$. If $x^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$ lies within this set, then $\hat{x}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}) = x^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$. Otherwise, we aim to respect the initial activation decision $y^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$ as much as possible. For this, we choose the online solution $\hat{x}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$ to be the feasible solution that is closest to the initial solution $x^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$.

In the following, we derive an expression for this online solution $\hat{x}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$ given the initial feasible solution $x^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$ and the projection set $\mathcal{C}_{\text{proj}}^{\bar{t}}$. For this, note that it follows from Equation (8.8) that $\mathcal{C}_{\text{proj}}^{\bar{t}}$ can be formulated according to Table 8.1 given the value of $\hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$.

Range of $\hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$	$\mathcal{C}_{\text{proj}}^{\bar{t}}$
$[0]$	$\{0\}$
$[X_{\min}, 2X_{\min}]$	$\{0, \hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})\}$
$[2X_{\min}, X_{\max}]$	$\{0\} \cup [X_{\min}, \hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}) - X_{\min}]$ $\cup \{\hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})\}$
$(X_{\max}, (T - \bar{t})X_{\max}]$	$\{0\} \cup [X_{\min},$ $\min(\hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}) - X_{\min}, X_{\max})]$
$((T - \bar{t})X_{\max},$ $(T - \bar{t} + 1)X_{\max}]$	$[\max(\hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}) - (T - \bar{t})X_{\max}, X_{\min}),$ $\min(X_{\max}, \hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}) - X_{\min})]$

Table 8.1: Structure of the projection set $\mathcal{C}_{\text{proj}}^{\bar{t}}$ based on the value of $\hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$.

Given that $\hat{x}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$ is required to be as close to $x^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$ as possible and that $\hat{x}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}) \notin \mathcal{C}_{\text{proj}}^{\bar{t}}$, we use Table 8.1 to obtain the following expressions for $\hat{x}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$, where we omit the arguments of $\hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$, $x^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$, and $\hat{x}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}})$ to improve the readability:

1. If $\hat{R}^{\bar{t}} = 0$, then $\hat{x}^{\bar{t}} = 0$.
2. If $\hat{x}^{\bar{t}} \in [X_{\min}, 2X_{\min})$, then

$$\hat{x}^{\bar{t}} = \begin{cases} 0 & \text{if } x^{\bar{t}} \leq \frac{1}{2}\hat{R}^{\bar{t}}, \\ \hat{R}^{\bar{t}} & \text{otherwise.} \end{cases}$$

3. If $\hat{R}^{\bar{t}}(\hat{Z}^{\text{act}}, \hat{Z}^{\text{mult}}) \in [2X_{\min}, X_{\max}]$, then

$$\hat{x}^{\bar{t}} = \begin{cases} 0 & \text{if } x^{\bar{t}} \leq \frac{1}{2}X_{\min}, \\ X_{\min} & \text{if } \frac{1}{2}X_{\min} < x^{\bar{t}} < X_{\min}, \\ \bar{R}^{\bar{t}} - X_{\min} & \text{if } \bar{R}^{\bar{t}} - X_{\min} < x^{\bar{t}} \leq \bar{R}^{\bar{t}} - \frac{1}{2}X_{\min}, \\ \hat{R}^{\bar{t}} & \text{otherwise.} \end{cases}$$

4. If $\hat{R}^{\bar{t}} \in (X_{\max}, (T - \bar{t})X_{\max}]$, then

$$\hat{x}^{\bar{t}} = \begin{cases} 0 & \text{if } x^{\bar{t}} \leq \frac{1}{2}X_{\min}, \\ X_{\min} & \text{if } \frac{1}{2}X_{\min} < x^{\bar{t}} < X_{\min}, \\ \bar{R}^{\bar{t}} - X_{\min} & \text{if } \bar{R}^{\bar{t}} - X_{\min} < x^{\bar{t}} \leq \bar{R}^{\bar{t}} - \frac{1}{2}X_{\min}, \\ \min(\hat{R}^{\bar{t}} - X_{\min}, X_{\max}) & \text{otherwise.} \end{cases}$$

5. If $\hat{R}^{\bar{t}} \in [(T - \bar{t})X_{\max}, (T - \bar{t} + 1)X_{\max}]$, then

$$\hat{x}^{\bar{t}} = \max(\hat{R}^{\bar{t}} - (T - \bar{t})X_{\max}, X_{\min}, \min(\hat{R}^{\bar{t}} - X_{\min}, X_{\max}, x^{\bar{t}}))$$

Using this scheme, we can efficiently compute an online solution $\hat{x}$ to Problem ThresholdEV.

8.3.4 EVALUATION

In this section, we study the performance of the online optimization approach derived in the previous subsection. In particular, we validate its ability to compute online solutions whose objective values are close to the optimal offline objective value. Moreover, we study the predictability of the two characterizing values Z^{act} and Z^{mult} .

We consider a situation where an EV is charged between 19:00 PM and 7:00 AM and where the charging window is divided into 15-minute stages, meaning that $\Delta t = \frac{1}{4}$. For the base load profile p , we use the same data as used in the evaluation

	Minimum	Maximum	Mean	Median
Z^{act}	1595	3947	2707	2954
Z^{mult}	3474	3761	3611	3615
Relative objective value	1.0001	1.3770	1.0679	1.0339

Table 8.2: Evaluation results. A relative objective value of 1 implies that the online solution is optimal.

of the online versions of Problem EV in Section 7.3 and Problem RelaxEV in Section 8.2.4. Moreover, we again choose the Nissan Leaf as reference EV [3], meaning that we set $\bar{R} = 40$ (kWh). Furthermore, we assume that a home charger with a maximum charging rate of $X_{\max} = 6.6$ kW is used and we set $X_{\min} = 1.1$ kW, which is in line with the recommendations in [8] regarding an acceptable level of efficiency when charging at low rates.

We generate predictions of Z^{act} and Z^{mult} by computing the optimal EV schedule for 70 previous days under the same circumstances as for the current charging session, i.e., the same values of $\bar{R}$, $X_{\min}$, and $X_{\max}$, using the base load profile of each of the days respectively. Furthermore, for each previous day, we compute an online solution using the approach of Section 8.3.3 and the predictions corresponding to that day as input. We compare each resulting solution to the optimal solution of the current charging session by computing its *relative objective value* as the ratio between the objective value of the resulting solution and the optimal objective value. Note that the smallest possible relative objective value is 1, which occurs when the online solution is optimal.

Figure 8.6 shows the relative objective value of the online solution for different choices of predictions $\hat{Z}^{\text{act}}$ and $\hat{Z}^{\text{mult}}$. Here, the minimum is attained at $Z^{\text{act}} = 3206$ and $Z^{\text{mult}} = 3705$. The figure implies that slight deviations of $\hat{Z}^{\text{act}}$ and $\hat{Z}^{\text{mult}}$ from Z^{act} and Z^{mult} respectively hardly influence the relative objective value. This suggests that the online approach is robust against prediction errors in these two characterizing values.

Table 8.2 provides a summary of the evaluation results. Since Z^{act} varies significantly over the course of consecutive days and the predictions $\hat{Z}^{\text{mult}}$ remains within a small interval around Z^{mult} , these results imply that Z^{mult} is easier to predict than Z^{act} . The objective value of the online solution is at most 3.39% higher than the optimal objective value for 50% of the days, which implies that the online optimization approach often computes online solutions that are near-optimal.

Figure 8.6: Relative objective value for different combinations of $\hat{Z}^{\text{act}}$ and $\hat{Z}^{\text{mult}}$ for one day. The minimum of 1 occurs at $(Z^{\text{act}}, Z^{\text{mult}}) = (3206, 3705)$.

8.4 CONCLUSIONS AND OUTLOOK

In this chapter, we studied the necessity of some of the core assumptions of the online duality driven optimization (ODDO) framework introduced in Chapter 6. These assumptions were:

1. The cost functions are continuously differentiable;
2. The cost functions are *strictly* convex;
3. The feasible set is convex.

To this end, we formulated two variants of the EV charging scheduling problem introduced in Section 3.4 that take into account several practical limitations of EV batteries and the charging infrastructure. In the first variant, the first two assumptions are not satisfied, while in the second variant the third assumption is not satisfied. For both these variants, we discussed solution approaches to solve their offline version and presented adaptations of the original ODDO framework to solve their online version. Simulation results indicate that the practical

performance of these approaches is similar to that of the original framework on the problems discussed in Section 6.5. They are able to compute near-optimal online solutions using only simple statistics to predict the optimal Lagrange multipliers and additional characterizing values. Thus, the aforementioned three assumptions within the original framework are not always strictly necessary to ensure a good performance.

In the first variant, where the cost functions are not continuously differentiable and not *strictly* convex, the predictability of the optimal Lagrange multiplier varied significantly depending on the amount of resource, i.e., on the charging requirement. More precisely, the variance of the optimal Lagrange multiplier with regard to charging sessions on consecutive days depends strongly on the amount of resource, which is due to the step-wise nature of the Lagrange multiplier for this problem. As a consequence, we expect that the practical performance of our online approach can also differ significantly depending on this amount. This is a disadvantage for the practical usage of the approach since it reduces the stability of its outcomes and thereby the reliability of the approach. Therefore, more research is needed to identify more specifically the influence of the problem parameters on the predictability of the optimal Lagrange multiplier and to derive performance guarantees that possibly depend on these parameters.

In the second variant, where the feasible set is non-convex, we were able to derive a characterization of the optimal offline solution in terms of the number of "idle" stages, i.e., where no charging is done and thus the primal solution value is 0, and the optimal Lagrange multiplier corresponding to the resulting instance of the quadratic resource allocation problem (Problem QRA). The main reason for this is that we could identify and exploit the monotonic structure of the optimal solution, meaning that given the number of idle stages, we can efficiently determine which stages are idle. This structure is closely related to that of Problem QRA, where a similar structure can be identified. Thus a promising possible extension of this approach is to other resource allocation problems (RAPs) with, for example, different cost functions and / or constraint structures. As mentioned in Section 8.3.1, two successful examples of such extensions are to the case where the objective function is $(0, b, f)$ -separable for a given vector $b \in \mathbb{R}^n$ and some convex function f (see also Section 2.2.2), or where the upper single-variable bound, i.e., the maximum charging rate, must be sufficiently larger than the minimum threshold but may be different for each stage. One interesting direction for future research is to identify possible extensions to RAPs with additional submodular constraints or special cases of these constraints (see also Section 2.2.3).

9

CONCLUSION

ABSTRACT – In this concluding chapter, we first summarize the main results obtained in this thesis. Subsequently, we answer the research question and its two subquestions posed at the beginning of this thesis. Based on these answers and on the conducted research, we also state relevant and interesting directions for future research. Finally, we conclude this thesis with some general remarks regarding energy management research.

9.1 SUMMARY

Current developments with regard to energy, energy consumption, and the energy transition have led to a substantial increase in research on alternative energy systems based on renewable energy sources (RES) and on energy-efficient devices. One crucial aspect of those systems and devices is that they require some form of *energy management* to actively manage the energy flows within the system. This management is needed to ensure that the capacity restrictions of the system are respected. In this thesis, we considered several of these energy management problems from the mathematical perspectives of resource allocation and (Lagrangian) duality theory and proposed tailored optimization algorithms to solve these problems.

In Chapter 2, we provided an overview of the resource allocation perspective and discussed the relation between different possible objectives of energy management problems. By exploiting known theoretical properties of resource allocation problems (RAPs), we showed that often there exists a solution to a given energy management problem that optimizes several different goals simultaneously. This has important consequences for developing solution approaches to these problems, which we highlighted and discussed for several important applications such as communication systems, micro-grids, and processor scheduling.

In Chapter 4, we considered a RAP with additional restrictions on the allocation of resource to nested sequences of activities. This problem occurs as a core problem in many applications such as the ones mentioned in the previous paragraph. We presented an algorithm for this problem that attains the same worst-case time complexity as the current state-of-the-art approaches, but is around one order of magnitude faster and can be implemented using much simpler subroutines. This makes the algorithm highly suitable for application in real energy management systems.

In Chapter 5, we focused on an energy management problem and a corresponding RAP motivated by recent developments in three-phase electric vehicle charging. We provide a thorough formal analysis of this problem and its structural properties and, based on this, present an efficient solution approach and two algorithms to solve this problem. Both algorithms are fast enough for proper usage in real energy management systems.

In Chapters 6-8, we presented a new framework called "Online Duality-Driven Optimization" (ODDO) for solving online optimization problems wherein not all problem data is known beforehand. The key idea in this framework is to transform the uncertainty in input data into an uncertainty on values that characterize an optimal solution such as Lagrange multipliers and focus on predicting these characterizing values to obtain good online solutions. In Chapter 6, we formally introduced the ODDO framework and explained its suitability for solving energy management problems that require information on future energy usage profiles. For a specific but relevant class of RAPs and energy management problems, we derived a theoretical guarantee on the quality of the resulting online solutions. Moreover, as a proof-of-concept, we applied the framework to several practical problems and showed that it is able to deliver good online solutions for these problems, thereby demonstrating the promising potential of the ODDO framework.

In Chapter 7, we presented two approaches for predicting the characterizing values (Lagrange multipliers) of an optimization problem and demonstrated their good practical performance by applying them within the ODDO framework to a specific EV scheduling problem. Finally, in Chapter 8, we extended the ODDO framework to two different types of energy management problems in the context of EV charging. These problems do not fully satisfy the original assumptions of the framework but, compared to the EV charging scheduling problem considered in Chapter 7, incorporate several important practical characteristics of the physical EV charging process. We showed that by making several adjustments to the original ODDO framework, we were successful in solving also these problems in practice.

9.2 CONCLUSIONS

To conclude, we answer the research question and its two subquestions posed in Section 1.3. The main research question was formulated as follows:

211

How can we develop algorithms for energy management problems that are efficient and give good solutions both in theory and in practice?

This question led to two subquestions, of which the first one was:

- » How can we incorporate physical properties of devices in models and algorithms for energy management problems while maintaining good theoretical performance guarantees for corresponding solution approaches?

The algorithms developed in this thesis, first of all, indicate that it is indeed possible to develop algorithms that satisfy all these properties. Especially for scheduling the charging of EVs, we have integrated several physical properties of the EV charging process in existing basic optimization models while maintaining good theoretical performance guarantees. For instance, in Chapter 5, we extended the basic EV charging optimization model introduced in Chapter 3 to also support three-phase EV charging. This type of charging is seen as a promising solution for integrating an increasing number of EVs in residential distribution grids while avoiding grid congestion and blackouts [80, 174]. Another example is given in Chapter 8, where we integrate in the basic model an additional restriction present in most EV charging infrastructures that charging at rates close to zero is not possible [87, 182]. This extension increased the applicability of the models in real-life energy management systems, as demonstrated by the successful application of our algorithm for this problem in the field tests conducted in [77].

Several of the algorithms presented in this thesis and other alternative solution approaches use algorithms for existing, simpler models as building blocks. More precisely, the problems presented in Chapters 4, 5, and 8 could be reduced to specific sequences of the quadratic RAP introduced in Chapter 3. The used algorithms for this simpler problem already had both good theoretical performance guarantees and a good practical performance. As a consequence, also the algorithms for extensions of this particular simpler RAP took over these properties. This indicates that one way to combine all three desired properties into an algorithm is to start from simple problems that already have a good theoretical and practical performance and try to reduce the problem at hand (i.e., a problem that incorporates a practical requirement of physical property) to such a problem or a specific sequence of such problems.

With regard to the actual implementation of an algorithm, it is important to clearly define a desired and/or acceptable trade-off between these three aspects. This trade-off depends on the envisioned application of the algorithm (e.g., the environment wherein the algorithm has to be embedded or the background

of those who implement or use it) and the availability and necessity of alternatives for necessary subroutines. Sometimes, this means that a more general "multi-purpose" approach should be favored over a tailored approach. In the following, we illustrate the necessity of this trade-off by highlighting two examples encountered while carrying out the research presented in this thesis.

When different procedures with similar theoretical performance guarantees are available, practical aspects such as ease of implementation should determine the choice for the procedure. One example of this was the implementation of double-ended priority queues in Chapter 4. Usually, it can be expected that a tailored procedure to solve a given problem yields better results than a more generally applicable procedure. Therefore, initially, we implemented a tailored data structure, namely symmetrical min-max heaps [9], for the double-ended priority queues. However, the actual implementation and its validation were time-consuming, which prompted us to consider simpler alternatives with a similar performance. It turned out that a much simpler but more generic approach involving "ordinary" heaps was at least as fast in practice as the specialized min-max heaps structure (see also [30]) and thus eventually this simpler approach was used in the final implementation. This choice significantly reduced the complexity of the final implementation and the specialized knowledge needed for the implementation, which increased the usability of the implementation in real energy management systems.

Another aspect is the potential reduction in execution time of a candidate procedure. If this is not significant enough for the given application, it may not be worth the additional time and effort required to implement such an alternative procedure. More precisely, it can occur that even if an alternative approach is in practice faster than a theoretically efficient method, the potential reduction in execution is very small compared to the execution time of the overall solution procedure. This was observed in Chapter 5 with regard to sorting so-called breakpoints. This was an important step of our presented solution approach and, moreover, together with another step defined the worst-case time complexity of the presented algorithms. To sort the breakpoints, we could have exploited the structure of the input data, which consisted of a collection of already sorted lists, by using a specialized sorting algorithm for data with this particular structure [102]. However, within the overall solution approach, the practical execution time of the initial "naive" sorting algorithm was already more than one order of magnitude smaller than the execution time of the entire approach. Thus, since implementing or calling from an existing library a standard sorting algorithm is easier, more robust, and faster than implementing a specialized sorting algorithm, it was decided not to exploit the present structure and instead use a standard sorting algorithm to implement this step.

The second subquestion was formulated as follows:

- » How can we obtain good solutions to energy management problems that do not require detailed predictions of uncertain data such as future energy demand and supply?

We have addressed this question by proposing a new framework for optimization with uncertain data called "Online Duality-Driven Optimization" (ODDO). This framework, presented in Chapters 6-8 of this thesis, answers the posed subquestion by transforming the need for detailed predictions of the uncertain data themselves into a need for predictions of values (optimal Lagrange multipliers) that characterize optimal solutions. This transformation is advantageous compared to other approaches for solving optimization problems with uncertain data because it nullifies the need for detailed predictions of, essentially, human behavior that underlies energy consumption patterns. Moreover, the transformation is beneficial especially for (energy management) problems wherein the (physical) properties of the system or device can be described using only a few rules or equations. The reason for this is that for such problems the number of to-be-predicted characterizing values is relatively small since this number is proportional to the number of imposed constraints.

Computational evaluations of the ODDO framework indicate that predicting optimal Lagrange multipliers is not necessarily more difficult but often even easier than predicting the actual uncertain data themselves. One of the reasons for this is that these values often represent general information on an optimal solution to the problem or on a given aggregation of the data. For instance, in Chapter 7, we proposed a prediction approach that exploits the present problem structure explicitly. The resulting prediction depends directly on an aggregation of the uncertain data, i.e., on the sum of the uncertain data points. This indicates that knowledge on such an aggregated value directly leads to knowledge on the optimal Lagrange multipliers, which suggests that such aggregated values are sufficient to solve the given problem. This confirms the underlying motivation for the posed subquestion, namely that detailed predictions of individual data points are not necessary to obtain good solutions to energy management problems with data uncertainty.

9.3 RECOMMENDATIONS FOR FUTURE RESEARCH

In this final section, we present our recommendations for future research regarding the results obtained in this thesis and regarding the field of energy management in general.

We stressed in Chapter 5 the importance of considering the division of the energy load of devices such as EVs over the three phases of the residential distribution network instead of just a single phase. Therefore, also managing the energy flows on all three phases within other energy management problems is an important direction for future research. A promising starting point for this, based on the results in Chapters 4 and 5, is to develop a three-phase energy manage-

ment approach for storage systems such as neighborhood batteries. In terms of RAPs, this corresponds to developing a fast and efficient algorithm to solve nonseparable quadratic RAPs with *laminar* or tree-structured constraints (see also Section 2.3.4). Such an algorithm could contribute both to our understanding of nonseparable RAPs and to actively manage energy flows over all three phases.

Within the ODDO framework presented in Chapters 6-8, we focused on optimization problems where the uncertainty is present only in the objective of the problem, i.e., not in the constraints. Assuming constraint uncertainty in the current version of the framework does not change the applicability of the framework in the sense that still an initial online solution for each stage can be computed. This is due to the separable nature of the constraints, which ensures that no (uncertain) constraint data for future stages is required to compute this initial solution. However, feasibility of the entire solution, i.e., the solution over all stages, cannot be guaranteed because the constraint functions for the future stages are unknown. To overcome this problem, one might be able to approximate the uncertain constraints by using other techniques for optimization under uncertainty such as stochastic programming, (adjustable) robust optimization, or online convex optimization, depending on which information on the constraint uncertainty is available. Combining such an approximation with ODDO is an interesting direction for future research on solving optimization problems with uncertain data.

The main purpose of the prediction approaches for optimal Lagrange multipliers presented in Chapter 7 and the computational studies in Chapters 6-8 is to serve as a proof-of-concept to assess the potential of the ODDO framework for solving practical energy management problems. To further strengthen the framework, a more elaborate validation is required that includes more different energy management problems and a more sophisticated analysis of the sensitivity of the prediction approaches. Furthermore, a significant improvement of the framework would be to develop a prediction approach that gives (probabilistic) guarantees on the prediction error and to extend the derived theoretical performance guarantee to other problem classes. Together, this leads to a more complete framework that provides a computable performance guarantee for any used multiplier prediction, which significantly increases its potential for usage in, e.g., real energy management systems.

In the energy management problems considered in this thesis, only the short-term effects of energy usage are taken into account in the objectives, i.e., it is assumed that the "cost" of energy usage at a given moment in time is independent of the cost at any other moment. However, hereby several long-term effects of a given energy usage profile that might influence the proper functioning of the system or device are not taken into account. One important example of this is the effect of a given energy usage profile on the temperature of the device and/or specific assets of the considered energy system. For example, electricity cables degrade faster and can even overheat and break down if their capacity is exceeded

for a longer consecutive time period. Moreover, battery-powered devices can overheat if they consume much energy for a longer consecutive period of time and are not ventilated properly. Thus, one important goal of energy management strategies is to minimize the peak temperature of the device and/or grid assets. Apart from the properties of the device and/or system themselves, this requires an accurate model of the relation between energy usage and temperature. The optimization models considered in this thesis do not actively incorporate temperature as a system variable. Thus, one important and interesting direction for future research is the development and integration of temperature models into energy management optimization problems.

Finally, the field of optimization approaches for energy management problems has seen a huge increase in publications over the last few years. For instance, the search term "scheduling of electric vehicle charging" yields over 45,000 publications in Google Scholar at the time of writing, of which more than 15,000 are from 2019 or more recent. Therefore, in order to create some clarity and structure in this increasing mass of literature, it is increasingly important to clearly state the criteria that one uses as guidelines for developing new models and approaches. Moreover, the eventual developed solution approach should match the stated criteria. For instance, when the stated goal is to develop approaches that are usable in practice, the resulting algorithms should not contain advanced subroutines that work in theory but are virtually impossible to be implemented correctly in practice. By a clear formulation of the assessment criteria on the one hand and a clear explanation of how the developed approaches satisfy these criteria on the other hand, the research actually becomes accessible to fellow scientists and other interested parties. Hereby, these people are able to quickly put the work into context and decide if it is relevant for their own projects and interests. More importantly, making research more accessible in this way allows not only for more efficient searches of the literature, but also for a critical evaluation and a deserved valuation of the research itself. The latter two aspects are important drivers for science in general and energy management research in particular. Thus, more effort should be put into making research accessible, both on the technical level towards fellow scientists and on a more intuitive level towards the general public.

TECHNICAL PROOFS

ABSTRACT – *This appendix contains the proofs of several technical lemmas in this thesis.*

A.1 PROOFS OF CHAPTER 4

A.1.1 PROOF OF LEMMA 4.1

Lemma. *If $L_j \leq A \leq B \leq U_j$, we have $x^j(A) \leq x^j(B)$ for a given $j \in \mathcal{N}$.*

Proof. For convenience, we include the equality constraint (4.2) into the nested constraints (4.3) by replacing these nested constraints by

$$\tilde{L}_k \leq \sum_{i \in \mathcal{N}_k} x_i \leq \tilde{U}_k, \quad k \in \mathcal{N}_j,$$

where $\tilde{L}_k = L_k$ and $\tilde{U}_k = U_k$ for $k < j$, and $\tilde{L}_j = \tilde{U}_j = C$. The Karush-Kuhn-Tucker (KKT) optimality conditions (see, e.g., [29]) for the subproblem QRA-NC^j(C) are as follows:

$$\frac{x_i}{a_i} + \sum_{k=i}^j (\eta_k^j - \zeta_k^j) + \mu_i^j - \nu_i^j = 0, \quad i \in \mathcal{N}_j, \quad (\text{A.1a})$$

$$\tilde{L}_i \leq \sum_{k \in \mathcal{N}_i} x_k \leq \tilde{U}_i, \quad i \in \mathcal{N}_j, \quad (\text{A.1b})$$

$$\eta_i^j \left(\tilde{U}_i - \sum_{k \in \mathcal{N}_i} x_k \right) = 0, \quad i \in \mathcal{N}_j, \quad (\text{A.1c})$$

$$\zeta_i^j \left(\sum_{k \in \mathcal{N}_i} x_k - \tilde{L}_i \right) = 0, \quad i \in \mathcal{N}_j, \quad (\text{A.1d})$$

$$\mu_i^j(u_i - x_i) = 0, \quad i \in \mathcal{N}_j, \quad (\text{A.1e})$$

$$\nu_i^j(x_i - l_i) = 0, \quad i \in \mathcal{N}_j, \quad (\text{A.1f})$$

$$\eta_i^j, \zeta_i^j, \mu_i^j, \nu_i^j \geq 0, \quad i \in \mathcal{N}_j. \quad (\text{A.1g})$$

Let $(\zeta^j(C), \eta^j(C), \mu^j(C), \nu^j(C))$ denote the Lagrange multipliers corresponding to the optimal solution $x^j(C)$. Thus, $(x^j(C), \zeta^j(C), \eta^j(C), \mu^j(C), \nu^j(C))$ satisfy the KKT-conditions (A.1).

Suppose that there exists an index $s \in \mathcal{N}$ such that $x_s^j(A) > x_s^j(B)$. Let r be the largest index with $r \leq s$ such that $\sum_{k \in \mathcal{N}_{r-1}} x_k^j(A) \geq \sum_{k \in \mathcal{N}_{r-1}} x_k^j(B)$, and let t be the smallest index with $t \geq s$ such that $\sum_{k \in \mathcal{N}_t} x_k^j(A) \leq \sum_{k \in \mathcal{N}_t} x_k^j(B)$. By definition of r , s , and t , we have that

$$\begin{aligned} \sum_{i=r}^t x_i^j(B) &= \sum_{i \in \mathcal{N}_t} x_i^j(B) - \sum_{i \in \mathcal{N}_{r-1}} x_i^j(B) \\ &\geq \sum_{i \in \mathcal{N}_t} x_i^j(A) - \sum_{i \in \mathcal{N}_{r-1}} x_i^j(A) = \sum_{i=r}^t x_i^j(A). \end{aligned}$$

Moreover, observe that we cannot have $r = s = t$ simultaneously. Indeed, if $r = s = t$, then we have by definition of r , s , and t that

$$\sum_{k \in \mathcal{N}_s} x_k^j(A) \leq \sum_{k \in \mathcal{N}_s} x_k^j(B) \leq \sum_{k \in \mathcal{N}_{s-1}} x_k^j(A) + x_s^j(B).$$

This implies $x_s^j(A) \leq x_s^j(B)$, which is a contradiction. Thus, either $r < s$ or $s < t$ or both.

We show that we obtain a contradiction if $r < s$. The proof for the case where $s < t$ is symmetrical. If $r < s$, the following holds:

» By definition of r and s , we have

$$\begin{aligned} \sum_{k \in \mathcal{N}_r} x_k^j(A) &< \sum_{k \in \mathcal{N}_r} x_k^j(B) = \sum_{k \in \mathcal{N}_{r-1}} x_k^j(B) + x_r^j(B) \\ &\leq \sum_{k \in \mathcal{N}_{r-1}} x_k^j(A) + x_r^j(B). \end{aligned}$$

Thus, $x_r^j(A) < x_r^j(B)$.

» For each k such that $r \leq k \leq s-1$, we have by definition of r and s and KKT-condition (A.1b) that

$$\tilde{L}_k \leq \sum_{i \in \mathcal{N}_k} x_i^j(A) < \sum_{i \in \mathcal{N}_k} x_i^j(B) \leq \tilde{U}_k.$$

It follows from KKT-conditions (A.1c), (A.1d), and (A.1g) that $\zeta_k^j(B) = \eta_k^j(A) = 0$. Thus, for each $r \leq k \leq s-1$, we have

$$\sum_{i=k}^j (\eta_i^j(A) - \zeta_i^j(A)) - \sum_{i=k+1}^j (\eta_i^j(A) - \zeta_i^j(A)) = \eta_k^j(A) - \zeta_k^j(A) \leq 0$$

and

$$\sum_{i=k}^j (\eta_i^j(B) - \zeta_i^j(B)) - \sum_{i=k+1}^j (\eta_i^j(B) - \zeta_i^j(B)) = \eta_k^j(B) - \zeta_k^j(B) \geq 0.$$

In particular, this implies that

$$\sum_{i=r}^j (\eta_i^j(A) - \zeta_i^j(A)) \leq \sum_{i=s}^j (\eta_i^j(A) - \zeta_i^j(A)) \quad (\text{A.2})$$

and

$$\sum_{i=r}^j (\eta_i^j(B) - \zeta_i^j(B)) \geq \sum_{i=s}^j (\eta_i^j(B) - \zeta_i^j(B)). \quad (\text{A.3})$$

» We have $l_r \leq x_r^j(A) < x_r^j(B) \leq u_r$. It follows from KKT-conditions (A.1e)-(A.1g) that

$$\nu_r^j(B) = \mu_r^j(A) = 0. \quad (\text{A.4})$$

Similarly, since $l_s \leq x_s^j(B) < x_s^j(A) \leq u_s$, we have by KKT-conditions (A.1e)-(A.1g) that

$$\nu_s^j(A) = \mu_s^j(B) = 0. \quad (\text{A.5})$$

We can now derive a contradiction as follows:

$$\sum_{i=s}^j (\eta_i^j(A) - \zeta_i^j(A)) = -\frac{x_s^j(A)}{a_s} - \mu_s^j(A) + \nu_s^j(A) \quad (\text{A.6a})$$

$$< -\frac{x_s^j(B)}{a_s} - \mu_s^j(B) + \nu_s^j(B) \quad (\text{A.6b})$$

$$= \sum_{i=s}^j (\eta_i^j(B) - \zeta_i^j(B)) \quad (\text{A.6c})$$

$$\leq \sum_{i=r}^j (\eta_i^j(B) - \zeta_i^j(B)) \quad (\text{A.6d})$$

$$= -\frac{x_r^j(B)}{a_r} - \mu_r^j(B) + \nu_r^j(B) \quad (\text{A.6e})$$

$$< -\frac{x_r^j(A)}{a_r} - \mu_r^j(A) + \nu_r^j(A) \quad (\text{A.6f})$$

$$= \sum_{i=r}^j (\eta_i^j(A) - \zeta_i^j(A)) \quad (\text{A.6g})$$

$$\leq \sum_{i=s}^j (\eta_i^j(A) - \zeta_i^j(A)). \quad (\text{A.6h})$$

Here,

- » (A.6a), (A.6c), (A.6e), and (A.6g) follow from KKT-condition (A.1a);
- » (A.6b) follows from Equation (A.5) and the fact that $x_s^j(A) > x_s^j(B)$ and $a_s > 0$;
- » (A.6d) follows from Equation (A.3);
- » (A.6f) follows from Equation (A.4) and the fact that $x_r^j(A) < x_r^j(B)$ and $a_s > 0$;
- » (A.6h) follows from Equation (A.2).

It follows that $x_s^j(A) \leq x_s^j(B)$. □

A.1.2 PROOF OF LEMMA 4.4

Lemma. *It holds that $\chi^n = \kappa^n = \lambda^n$. Moreover, for each $j \in \mathcal{N}_{n-1}$, the following implications are valid:*

1. $\chi^{j+1} \leq \kappa^j$ implies $\sum_{i \in \mathcal{N}_j} x_i^n(R) = L_j$ and $\chi^j = \kappa^j$;
2. $\lambda^j \leq \chi^{j+1}$ implies $\sum_{i \in \mathcal{N}_j} x_i^n(R) = U_j$ and $\chi^j = \lambda^j$;
3. $\kappa^j < \chi^{j+1} < \lambda^j$ implies $L_j < \sum_{i \in \mathcal{N}_j} x_i^n(R) < U_j$ and $\chi^j = \chi^{j+1}$.

Proof. We have $\chi^n = \kappa^n = \lambda^n$ since we defined $L_n = U_n = R$ and by definition of the solution $x^n(R)$ the nested constraints $L_n \leq \sum_{i \in \mathcal{N}} x_i^n(L_n)$ and $\sum_{i \in \mathcal{N}} x_i^n(U_n) \leq U_n$ are tight. We prove the lemma by considering each of its three cases separately for each $j < n$:

1. We prove this part of the lemma for the case that j is the largest index smaller than ℓ_{j+1} such that $\chi^{j+1} \leq \kappa^j$, i.e., $\chi^{k+1} > \kappa^k$ for all $k \in \{j+1, \dots, \ell_{j+1}-1\}$. Using this result, we show as follows that the other case, i.e., both the situations where either $j = \ell_{j+1}$ or where there exists an index $k > j$ that it is the largest index in the set $\{j+1, \dots, \ell_{j+1}-1\}$ such that $\chi^{k+1} \leq \kappa^k$, leads to a contradiction. In the former situation, it follows that $j+1 > j = \ell_{j+1} \geq j+1$, which is a contradiction. In the latter situation, the lemma applies for k , meaning that $\sum_{i \in \mathcal{N}_k} x_i^n(R) = L_k$ and thus $\ell_k = k$. However, we also have by definition of ℓ_{j+1} that $\ell_k = \ell_{j+1}$ since $j+1 \leq k < \ell_{j+1}$. This implies $k = \ell_{j+1}$, which is a contradiction.

If $\chi^{j+1} \leq \kappa^j$, it follows from the lower breakpoint relations in Equation (4.7) that we have either $\alpha_i^{j+1} \geq \kappa^j \geq \chi^{j+1}$ (if $\kappa^j < \beta_i^j$) or $\alpha_i^{j+1} = \beta_i^j \leq \kappa^j \leq \lambda^j$ (if $\beta_i^j \leq \kappa^j$) for all $i \leq j+1$. We show that in both cases it holds that $x_i^{\ell_{j+1}}(V_{\ell_{j+1}}) = x_i^j(L_j)$:

- » In the former case, note that $\alpha_i^k \leq \alpha_i^{k+1}$ for all $k < n$ by Equation (4.7) and that $\chi^{j+1} = \chi^k$ for all $k \in \{j+1, \dots, \ell_{j+1}\}$ by definition of χ^{j+1} . Since $\chi^{k+1} > \kappa^k$ for all $k \in \{j+1, \dots, \ell_{j+1}-1\}$, we have that $\alpha_i^k \geq \alpha_i^{j+1} \geq \chi^{j+1} = \chi^{k+1} > \kappa^k$ for all $k \in \{j+1, \dots, \ell_{j+1}-1\}$. Thus, $x_i^k(L_k) = \bar{l}_i^k = x_i^{k-1}(L_{k-1})$ for all $k \in \{j+1, \dots, \ell_{j+1}-1\}$, which implies that $x_i^j(L_j) = x_i^{\ell_{j+1}-1}(L_{\ell_{j+1}-1})$. Moreover, note that since $\alpha^{\ell_{j+1}} \geq \alpha^{j+1} \geq \chi^{j+1} = \chi^{\ell_{j+1}}$, we have that $x_i^{\ell_{j+1}}(V_{\ell_{j+1}}) = x_i^{\ell_{j+1}-1}(L_{\ell_{j+1}-1})$. Therefore, it follows that $x_i^{\ell_{j+1}}(V_{\ell_{j+1}}) = x_i^j(L_j)$.
- » The latter case implies that $x_i^j(L_j) = x_i^j(U_j) = \bar{u}_i^j$. It follows by Lemmas 4.1 and 4.2 that $x_i^{\ell_{j+1}}(V_{\ell_{j+1}}) \leq x_i^{\ell_{j+1}}(U_{\ell_{j+1}}) \leq x_i^j(U_j) = x_i^j(L_j) \leq x_i^{\ell_{j+1}}(L_{\ell_{j+1}}) \leq x_i^{\ell_{j+1}}(V_{\ell_{j+1}})$.

On the one hand, if $V_{\ell_{j+1}} = L_{\ell_{j+1}}$, we have

$$\begin{aligned} L_j &= \sum_{i \in \mathcal{N}_j} x_i^j(L_j) = \sum_{i \in \mathcal{N}_j} x_i^{\ell_{j+1}}(L_{\ell_{j+1}}) = L_{\ell_{j+1}} - \sum_{i=j+1}^{\ell_{j+1}} x_i^{\ell_{j+1}}(L_{\ell_{j+1}}) \\ &\geq \sum_{i \in \mathcal{N}_{\ell_{j+1}}} x_i^n(R) - \sum_{i=j+1}^{\ell_{j+1}} x_i^n(R) = \sum_{i \in \mathcal{N}_j} x_i^n(R) \geq L_j, \end{aligned}$$

where the inequality follows since $\sum_{i \in \mathcal{N}_{\ell_{j+1}}} x_i^n(R) = L_{\ell_{j+1}}$ and by Lemma 4.2. On the other hand, if $V_{\ell_{j+1}} = U_{\ell_{j+1}}$, we have by Lemma 4.2 that

$$L_j = \sum_{i \in \mathcal{N}_j} x_i^j(L_j) = \sum_{i \in \mathcal{N}_j} x_i^{\ell_{j+1}}(U_{\ell_{j+1}}) \geq \sum_{i \in \mathcal{N}_j} x_i^n(U_n) \geq L_j.$$

In both cases, it follows that $\sum_{i \in \mathcal{N}_j} x_i^n(R) = L_j$, from which it follows directly that $\chi^j = \kappa^j$.

2. The proof for the case $\lambda^j \geq \chi^{j+1}$ is analogous to the proof for the case $\chi^{j+1} \leq \kappa^j$.
3. Suppose that $x_i^j(L_j) = x_i^n(L_n)$ holds for all $i < j+1$. By Lemma 4.2, this implies that $x_i^k(L_k) = x_i^j(L_j) = x_i^n(L_n)$ for all $k \in \{j, \dots, n\}$ and $i < j+1$. In particular, we have that $x_i^k(L_k) = \bar{l}_i^k$ for all $k \in \{j+1, \dots, n\}$, which implies that $\kappa^k \leq \alpha_i^k$. Furthermore, note that for any $k' \in \mathcal{N}$ there is at least one index $i_{k'} \leq k$ such that $\alpha_{i_{k'}}^{k'} \leq \kappa^{k'} < \beta_{i_{k'}}^{k'}$. Otherwise, there exists $\epsilon > 0$ such that $\kappa^{k'} + \epsilon$ is an optimal Lagrange multiplier. It

follows from the relation between $\alpha_{i_{k'}}^{k'}$ and $\alpha_{i_{k'}}^{k'+1}$ in Equation (4.7) that $\alpha_{i_{k'}}^{k'+1} = \kappa^{k'}$ for any $k' < n$. This implies in particular that $\alpha_{i_k}^{k+1} = \kappa^k \leq \alpha_{i_{k-1}}^k$ for all $k \in \{j+1, \dots, n\}$. It follows that $\kappa^{\ell_{j+1}} \leq \alpha_{i_j}^{j+1} = \kappa^j$ and thus that $\kappa^{\ell_{j+1}} < \chi^{j+1} = \chi^{\ell_{j+1}}$. Since $\chi^{\ell_{j+1}} \in \{\kappa^{\ell_{j+1}}, \lambda^{\ell_{j+1}}\}$, we have $\chi^{\ell_{j+1}} = \lambda^{\ell_{j+1}}$, from which it follows that $\sum_{i \in \mathcal{N}_{\ell_{j+1}}} x_i^n(R) = U_{\ell_{j+1}}$. However, this implies that

$$\begin{aligned} \sum_{i \in \mathcal{N}_{\ell_{j+1}}} x_i^{\ell_{j+1}}(L_{\ell_{j+1}}) &= \sum_{i \in \mathcal{N}_{\ell_{j+1}}} x_i^{\ell_{j+1}}(R) = U_{\ell_{j+1}} \\ &\geq \sum_{i \in \mathcal{N}_{\ell_{j+1}}} x_i^{\ell_{j+1}}(U_{\ell_{j+1}}) \geq \sum_{i \in \mathcal{N}_{\ell_{j+1}}} x_i^{\ell_{j+1}}(L_{\ell_{j+1}}), \end{aligned}$$

and thus that $\sum_{i \in \mathcal{N}_{\ell_{j+1}}} x_i^{\ell_{j+1}}(L_{\ell_{j+1}}) = \sum_{i \in \mathcal{N}_{\ell_{j+1}}} x_i^{\ell_{j+1}}(U_{\ell_{j+1}})$. It follows that $L_{\ell_{j+1}} = U_{\ell_{j+1}}$ by the monotonicity of $x^{\ell_{j+1}}(\cdot)$ as proven in Lemma 4.1. However, this is a contradiction with the assumption that $L_k < U_k$ for all $k < n$. Hence, there must be at least one index i' such that $x_{i'}^j(L_j) < x_{i'}^n(R)$. It follows that $L_j = \sum_{i \in \mathcal{N}_j} x_i^j(L_j) < \sum_{i \in \mathcal{N}_j} x_i^n(R)$. To prove that $\sum_{i \in \mathcal{N}_j} x_i^n(R) < U_j$, we can use a similar argument wherein we show that the proposition $x_i^j(U_j) = x_i^n(R)$ cannot be true for all $i < n$. Together, this implies that $L_j < \sum_{i \in \mathcal{N}_j} x_i^n(R) < U_j$, from which it follows directly that $\chi^j = \chi^{\ell_{j+1}} = \chi^{j+1}$.

□

A.1.3 PROOF OF LEMMA 4.5

Lemma. *For each $i \in \mathcal{N}$, we have*

$$x_i^n(R) = \begin{cases} l_i & \text{if } \chi^i < \alpha_i^i, \\ a_i \chi^i & \text{if } \alpha_i^i \leq \chi^i < \beta_i^i, \\ u_i & \text{if } \beta_i^i \leq \chi^i. \end{cases}$$

Proof. Let $\mathcal{J}$ denote the set of indices whose corresponding nested lower or upper constraint is tight in $x^n(R)$. More precisely,

$$\mathcal{J} := \{\ell_j \mid j \in \mathcal{N}\} \equiv \{j_1, \dots, j_q\},$$

where $q := |\mathcal{J}|$ and $j_1 < \dots < j_q$. For a given $p \in \{1, \dots, q\}$, note that since either the lower or upper nested constraint corresponding to j_p is tight in the solution $x^n(R)$, we have that $\sum_{i \in \mathcal{N}_{j_p}} x_i^n(R) = V_{j_p}$. This implies that the vector $(x_i^n(R))_{1 \leq i \leq j_p}$ is the optimal solution to the subproblem QRA-NC $^{j_p}(V_{j_p})$, i.e.,

to the problem

$$\begin{aligned}
 \text{QRA-NC}^{j_p}(V_{j_p}) : \min_{x \in \mathbb{R}^{j_p}} \sum_{i \in \mathcal{N}_{j_p}} \frac{1}{2} \frac{x_i^2}{a_i} \\
 \text{s.t.} \quad \sum_{i \in \mathcal{N}_{j_p}} x_i = V_{j_p}, \\
 L_k \leq \sum_{i \in \mathcal{N}_k} x_i \leq U_k, \quad k \in \{1, \dots, j_p - 1\}, \quad (\text{A.7}) \\
 l_i \leq x_i \leq u_i, \quad i \in \{1, \dots, j_p\}.
 \end{aligned}$$

Note that in the optimal solution $(x_i^n(R))_{i \in \mathcal{N}_{j_p}}$ to this problem, none of the nested constraints (A.7) for k with $j_{p-1} < k < j_p$ are tight. As a consequence, when deriving the reformulated equivalent problem $\text{QRA}^{j_p}(V_{j_p})$, it follows from Lemmas 4.2 and 4.3 that we may replace the single-variable bounds (4.5) for i with $j_{p-1} < i < j_p$ by the original variable bounds $l_i \leq x_i \leq u_i$. Thus, we can reformulate $\text{QPRAP-NC}^{j_p}(V_{j_p})$ to

$$\begin{aligned}
 \text{QRA}^{j_p}(V_{j_p}) : \min_{x \in \mathbb{R}^j} \sum_{i \in \mathcal{N}_{j_p}} \frac{1}{2} \frac{x_i^2}{a_i} \\
 \text{s.t.} \quad \sum_{i \in \mathcal{N}_{j_p}} x_i = V_{j_p}, \\
 x_i^{j_p-1}(L_{j_p-1}) \leq x_i \leq x_i^{j_p-1}(U_{j_p-1}), \quad i \in \{1, \dots, j_{p-1}\}, \\
 l_j \leq x_j \leq u_j, \quad i \in \{j_{p-1} + 1, \dots, j_p\}.
 \end{aligned}$$

Recall that χ^{j_p} is the optimal Lagrange multiplier for this problem. As a consequence, we can directly compute $x_i^{j_p}(R)$ for $i \in \{j_{p-1} + 1, \dots, j_p\}$ using Equation (3.2):

$$x_i^n(R) = x_i^{j_p}(V_{j_p}) = \begin{cases} l_i & \text{if } \chi^{j_p} < \alpha_i^i, \\ a_i \chi^{j_p} & \text{if } \alpha_i^i \leq \chi^{j_p} < \beta_i^i, \\ u_i & \text{if } \beta_i^i \leq \chi^{j_p}. \end{cases}$$

The result of the lemma follows since $\chi^{j_p} = \chi^i$ for each $i \in \{j_{p-1}, \dots, j_p\}$ by definition of j_p and χ^i . $\square$

A.2 PROOFS OF CHAPTER 5

A.2.1 PROOF OF LEMMA 5.1

Lemma. H^j is positive definite if and only if $1 + w_j \sum_{i \in \mathcal{N}_j} a_i > 0$.

Proof. Suppose that H^j is positive definite. Then its determinant is strictly positive. Due to the special structure of H^j , we can rewrite its determinant to

the following form by applying the matrix determinant lemma (see, e.g., [67]):

$$\det(H^j) = \det(w_j ee^\top + \text{diag}(a^j)) = \left(1 + w_j \sum_{i \in \mathcal{N}_j} a_i\right) \det(\text{diag}(a^j)).$$

Since $a_i > 0$ for all $i \in \mathcal{N}_j$, we have that $\det(\text{diag}(a^j)) > 0$ and thus we also have that $1 + w_j \sum_{i \in \mathcal{N}_j} a_i > 0$.

Now suppose that $1 + w_j \sum_{i \in \mathcal{N}_j} a_i > 0$. We show that all leading principal minors of H^j are positive, i.e., that the determinant of each upper-left submatrix of H^j is positive. For this, we label the indices of $\mathcal{N}_j$ as $1, \dots, n_j$ such that, for any $1 \leq \ell \leq n_j$, the $\ell \times \ell$ upper-left submatrix of H^j is formed by the first ℓ rows and columns of H^j . Let us denote this submatrix by $H_{1:\ell;1:\ell}^j$.

To show that $\det(H_{1:\ell;1:\ell}^j) > 0$, we compute this determinant by applying the matrix determinant lemma to $H_{1:\ell;1:\ell}^j$. This yields

$$\det(H_{1:\ell;1:\ell}^j) = \left(1 + w_j \sum_{i=1}^{\ell} a_i\right) \prod_{i=1}^{\ell} \frac{1}{a_i}.$$

Note that $1 + w_j \sum_{i=1}^{\ell} a_i > 0$ since $1 + w_j \sum_{i \in \mathcal{N}_j} a_i > 0$ and all values a_i are positive. It follows that $\det(H_{1:\ell;1:\ell}^j) > 0$. Since ℓ was chosen arbitrarily, this implies that all leading principal minors of H^j are positive and thus that H^j is positive definite. $\square$

A.2.2 PROOF OF LEMMA 5.2

Lemma. *For a given $j \in \mathcal{M}$, let $\underline{x}^j := (\underline{x}_i)_{i \in \mathcal{N}}$ and $\bar{x}^j := (\bar{x}_i)_{i \in \mathcal{N}}$ be optimal solutions to $\text{QRA}^j(L_j)$ and $\text{QRA}^j(U_j)$ respectively. Then there exists an optimal solution $x^* := (x_i^*)_{i \in \mathcal{N}}$ to Problem QRA-NonSep-GBC that satisfies $\underline{x}_i \leq x_i^* \leq \bar{x}_i$ for each $i \in \mathcal{N}_j$.*

Proof. We prove the validity of the lower bounds $\underline{x}_i \leq x_i^*$; the proof for the upper bounds $x_i^* \leq \bar{x}_i$ is analogous. If for a given $j \in \mathcal{M}$ there is no optimal solution x^* to Problem QRA-NonSep-GBC that satisfies the bounds $\underline{x}_i \leq x_i^* \leq \bar{x}_i$ for each $i \in \mathcal{N}_j$, then choose out of all these solutions the one solution x^* for which the value $d := \sum_{\ell \in \mathcal{N}_j} \max(\underline{x}_\ell - x_\ell^*, 0)$ is minimum. Let $i \in \mathcal{N}$ be an index with $x_i^* < \underline{x}_i$ and let j be such that $i \in \mathcal{N}_j$. Then there must exist $k \in \mathcal{N}_j \setminus \{i\}$ such that $x_k^* > \underline{x}_k$ since otherwise $\sum_{\ell \in \mathcal{N}_j} x_\ell^* < \sum_{\ell \in \mathcal{N}_j} \underline{x}_\ell = L_j$.

Let $\bar{\epsilon} := \min(\underline{x}_i - x_i^*, x_k^* - \underline{x}_k)$ and let $\epsilon \in (0, \bar{\epsilon}]$. Then the solution x' given by

$$x'_\ell = \begin{cases} x_\ell^* + \epsilon & \text{if } \ell = i, \\ x_\ell^* - \epsilon & \text{if } \ell = k, \\ x_\ell^* & \text{otherwise,} \end{cases}$$

is feasible for Problem QRA-NonSep-GBC since $x'_i = x_i^* + \epsilon \leq x_i^* + \bar{\epsilon} \leq x_i^* + \underline{x}_i - x_i^* = \underline{x}_i$, $x'_k = x_k^* - \epsilon \geq x_k^* - \bar{\epsilon} \geq x_k^* - x_k^* + \underline{x}_k = \underline{x}_k$, and $\underline{x}^j$ and x^* are feasible for $\text{QRA}^j(L_j)$ and QRA-NonSep-GBC respectively. Moreover, since x^* is an optimal solution to Problem QRA-NonSep-GBC, we have that

$$\begin{aligned} & \sum_{j'=1}^m \frac{1}{2} w_{j'} \left(\sum_{\ell \in \mathcal{N}_{j'}} x_i^* \right)^2 + \sum_{\ell=1}^n \left(\frac{1}{2} \frac{(x_\ell^*)^2}{a_\ell} + b_\ell x_\ell^* \right) \\ & \leq \sum_{j'=1}^m \frac{1}{2} w_{j'} \left(\sum_{\ell \in \mathcal{N}_{j'}} x'_i \right)^2 + \sum_{\ell=1}^n \left(\frac{1}{2} \frac{(x'_\ell)^2}{a_\ell} + b_\ell x'_\ell \right). \end{aligned}$$

It follows by definition of x' that

$$\begin{aligned} 0 & \leq \frac{1}{2} \frac{(x'_i)^2}{a_i} + b_i x'_i + \frac{1}{2} \frac{(x'_k)^2}{a_k} + b_k x'_k - \frac{1}{2} \frac{(x_i^*)^2}{a_i} - b_i x_i^* - \frac{1}{2} \frac{(x_k^*)^2}{a_k} - b_k x_k^* \\ & = \frac{1}{2} \frac{(x_i^* + \epsilon)^2}{a_i} + b_i(x_i^* + \epsilon) + \frac{1}{2} \frac{(x_k^* - \epsilon)^2}{a_k} + b_k(x_k^* - \epsilon) \\ & \quad - \frac{1}{2} \frac{(x_i^*)^2}{a_i} - b_i x_i^* - \frac{1}{2} \frac{(x_k^*)^2}{a_k} - b_k x_k^* \\ & = \frac{x_i^* \epsilon}{a_i} + \frac{1}{2} \frac{\epsilon^2}{a_i} + b_i \epsilon - \frac{x_k^* \epsilon}{a_k} + \frac{1}{2} \frac{\epsilon^2}{a_k} + b_k \epsilon. \end{aligned} \tag{A.8}$$

Analogously, the solution $(\underline{x}')^j$ given by

$$\underline{x}'_\ell = \begin{cases} \underline{x}_\ell - \epsilon & \text{if } \ell = i, \\ \underline{x}_\ell + \epsilon & \text{if } \ell = k, \\ \underline{x}_\ell & \text{otherwise,} \end{cases}$$

is feasible for $\text{QRA}^j(L_j)$ since $\underline{x}'_i = \underline{x}_i - \epsilon \geq \underline{x}_i - \bar{\epsilon} \geq \underline{x}_i - \underline{x}_i + x_i^* = x_i^*$, $\underline{x}'_k = \underline{x}_k + \epsilon \leq \underline{x}_k + \bar{\epsilon} \leq \underline{x}_k + x_k^* - \underline{x}_k = x_k^*$, and x^* and $\underline{x}^j$ are feasible for Problem QRA-NonSep-GBC and $\text{QRA}^j(L_j)$ respectively. Moreover, since $\underline{x}^j$ is optimal for $\text{QRA}^j(L_j)$, we have that

$$\sum_{\ell \in \mathcal{N}_j} \left(\frac{1}{2} \frac{(\underline{x}_\ell)^2}{a_\ell} + b_\ell \underline{x}_\ell \right) \leq \sum_{\ell \in \mathcal{N}_j} \left(\frac{1}{2} \frac{(\underline{x}'_\ell)^2}{a_\ell} + b_\ell \underline{x}'_\ell \right).$$

It follows by definition of $(\underline{x}')^j$ that

$$\begin{aligned} 0 & \leq \frac{1}{2} \frac{(\underline{x}'_i)^2}{a_i} + b_i \underline{x}'_i + \frac{1}{2} \frac{(\underline{x}'_k)^2}{a_k} + b_k \underline{x}'_k - \frac{1}{2} \frac{(\underline{x}_i)^2}{a_i} - b_i \underline{x}_i - \frac{1}{2} \frac{(\underline{x}_k)^2}{a_k} - b_k \underline{x}_k \\ & = \frac{1}{2} \frac{(\underline{x}_i - \epsilon)^2}{a_i} + b_i(\underline{x}_i - \epsilon) + \frac{1}{2} \frac{(\underline{x}_k + \epsilon)^2}{a_k} + b_k(\underline{x}_k + \epsilon) \\ & \quad - \frac{1}{2} \frac{(\underline{x}_i)^2}{a_i} - b_i \underline{x}_i - \frac{1}{2} \frac{(\underline{x}_k)^2}{a_k} - b_k \underline{x}_k \end{aligned}$$

$$= -\frac{\underline{x}_i \epsilon}{a_i} + \frac{1}{2} \frac{\epsilon^2}{a_i} - b_i \epsilon + \frac{\underline{x}_k \epsilon}{a_k} + \frac{1}{2} \frac{\epsilon^2}{a_k} + b_k \epsilon. \quad (\text{A.9})$$

226

Adding Equations (A.8) and (A.9) yields

$$0 \leq \frac{\epsilon(\underline{x}_i^* - \underline{x}_i + \epsilon)}{a_i} + \frac{\epsilon(\underline{x}_k - \underline{x}_k^* + \epsilon)}{a_k} \leq \frac{\epsilon(-\bar{\epsilon} + \epsilon)}{a_i} + \frac{\epsilon(-\bar{\epsilon} + \epsilon)}{a_k} \leq 0.$$

This implies that both Equations (A.8) and (A.9) are equalities and thus that x' and $(\underline{x}')^j$ are optimal for Problem QRA-NonSep-GBC and $\text{QRA}^j(L_j)$ respectively. However, since $x'_i \leq \underline{x}_i$ and $x'_k \geq \underline{x}_k$, it holds that

$$\begin{aligned} \sum_{\ell \in \mathcal{N}_j} \max(\underline{x}_\ell - x'_\ell, 0) &= d - \max(\underline{x}_i - x_i^*, 0) - \max(\underline{x}_k - x_k^*, 0) \\ &\quad + \max(\underline{x}_i - x'_i, 0) + \max(\underline{x}_k - x'_k, 0) \\ &= d - \underline{x}_i + x_i^* - 0 + \underline{x}_i - x'_i + 0 \\ &= d + x_i^* - x'_i \\ &= d - \epsilon. \end{aligned}$$

This is a contradiction with the definition of x^* as the optimal solution that minimizes the expression $\sum_{\ell \in \mathcal{N}_j} \max(\underline{x}_\ell - x_\ell^*, 0)$. Hence, there exists an optimal solution satisfying the lower bounds $\underline{x}$. $\square$

A.2.3 PROOF OF LEMMA 5.4

Lemma. For any $\lambda_1, \lambda_2 \in \mathbb{R}$ with $\lambda_1 < \lambda_2$, it holds that $x_i(\lambda_1) \leq x_i(\lambda_2)$, $i \in \mathcal{N}$.

Proof. Suppose that there exist λ_1, λ_2 with $\lambda_1 < \lambda_2$ such that for some $j \in \mathcal{M}$ and $i \in \mathcal{N}_j$ we have $x_i(\lambda_1) > x_i(\lambda_2)$. First, we show that there must exist an index $k \in \mathcal{N}_j \setminus \{i\}$ such that $x_k(\lambda_1) \leq x_k(\lambda_2)$. Subsequently, we show that the existence of such an index k leads to a contradiction.

For each $\ell \in \mathcal{N}_j$, we multiply KKT-condition (5.6a) by a_ℓ :

$$a_\ell w_j y_j + x_\ell + a_\ell(b_\ell - \lambda + \mu_\ell) = 0, \quad \ell \in \mathcal{N}_j. \quad (\text{A.10})$$

By summing Equation (A.10) over $\mathcal{N}_j$, we obtain

$$\begin{aligned} 0 &= w_j y_j \sum_{\ell \in \mathcal{N}_j} a_\ell + \sum_{\ell \in \mathcal{N}_j} (x_\ell + a_\ell(b_\ell - \lambda + \mu_\ell)) \\ &= \left(1 + w_j \sum_{\ell \in \mathcal{N}_j} a_\ell\right) y_j + \sum_{\ell \in \mathcal{N}_j} a_\ell(b_\ell - \lambda + \mu_\ell). \end{aligned} \quad (\text{A.11})$$

Suppose that there is no index $k \in \mathcal{N}_j \setminus \{i\}$ such that $x_k(\lambda_1) \leq x_k(\lambda_2)$. Then for all $\ell \in \mathcal{N}_j$, we have $x_\ell(\lambda_1) > x_\ell(\lambda_2)$, which in turn implies $y_j(\lambda_1) > y_j(\lambda_2)$. It

follows from Equation (A.11), Property 5.1, and Lemma 5.3 that

$$\begin{aligned}
0 &= \left(1 + w_j \sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell} \right) (y_j(\lambda_1) - y_j(\lambda_2)) \\
&\quad + \sum_{\ell \in \mathcal{N}_j} a_\ell (b_\ell - b_\ell - \lambda_1 + \lambda_2 + \mu_\ell(\lambda_1) - \mu_\ell(\lambda_2)) \\
&> \sum_{\ell \in \mathcal{N}_j} a_\ell (b_\ell - b_\ell - \lambda_1 + \lambda_2 + \mu_\ell(\lambda_1) - \mu_\ell(\lambda_2)) \\
&> \sum_{\ell \in \mathcal{N}_j} a_\ell (\mu_\ell(\lambda_1) - \mu_\ell(\lambda_2)) \\
&\geq 0.
\end{aligned}$$

227

A.2.3 – PROOF OF LEMMA 5.4

This is a contradiction and thus there must exist an index $k \in \mathcal{N}_j \setminus \{i\}$ with $x_k(\lambda_1) \leq x_k(\lambda_2)$.

We now show that the existence of the index k leads to a contradiction. By Lemma 5.3, we have $\mu_k(\lambda_1) \leq \mu_k(\lambda_2)$. It follows that

$$\frac{x_k(\lambda_1)}{a_k} + b_k + \mu_k(\lambda_1) \leq \frac{x_k(\lambda_2)}{a_k} + b_k + \mu_k(\lambda_2). \quad (\text{A.12})$$

However, KKT-condition (5.6a) implies that

$$w_j y_j(\lambda_1) + \frac{i(\lambda_1)}{a_i} + b_i + \mu_i(\lambda_1) = \lambda_1 = w_j y_j(\lambda_1) + \frac{x_k(\lambda_1)}{a_k} + b_k + \mu_k(\lambda_1),$$

which yields

$$\frac{x_i(\lambda_1)}{a_i} + b_i + \mu_i(\lambda_1) = \frac{x_k(\lambda_1)}{a_k} + b_k + \mu_k(\lambda_1). \quad (\text{A.13})$$

Analogously, we have

$$\frac{x_i(\lambda_2)}{a_i} + b_i + \mu_i(\lambda_2) = \frac{x_k(\lambda_2)}{a_k} + b_k + \mu_k(\lambda_2). \quad (\text{A.14})$$

It follows from Equations (A.12)-(A.14) and Lemma 5.3 that

$$\begin{aligned}
\frac{x_k(\lambda_1)}{a_k} + b_k + \mu_k(\lambda_1) &\leq \frac{x_k(\lambda_2)}{a_k} + b_k + \mu_k(\lambda_2) \\
&= \frac{x_i(\lambda_2)}{a_i} + b_i + \mu_i(\lambda_2) \\
&< \frac{x_i(\lambda_1)}{a_i} + b_i + \mu_i(\lambda_1) \\
&= \frac{x_k(\lambda_1)}{a_k} + b_k + \mu_k(\lambda_1).
\end{aligned}$$

This is a contradiction, hence it must be that $x_i(\lambda_1) \leq x_i(\lambda_2)$. As this implies that $x_i(\lambda_1) \leq x_i(\lambda_2)$ for all $\lambda_1 < \lambda_2$ and $i \in \mathcal{N}$, the lemma is proven. $\square$

A.2.4 PROOF OF LEMMA 5.5

Lemma. For all $i \in \mathcal{N}$, we have $\mu_i(\alpha_i) = \mu_i(\beta_i) = 0$.

228

Proof. Let $i \in \mathcal{N}_j$ for some $j \in \mathcal{M}$. We prove the lemma for $\mu_i(\alpha_i)$; the proof for $\mu_i(\beta_i)$ is analogous. Consider the solutions $x(\alpha_i)$ and $x(\alpha_i + \epsilon)$ for an arbitrary $\epsilon > 0$ with $\alpha_i + \epsilon < \beta_i$. Note that $\mu_i(\alpha_i + \epsilon) = 0$ by Equation (5.7b) and KKT-conditions (5.6d) and (5.6e). It follows from KKT-condition (5.6a) that

$$w_j y_j(\alpha_i) + \frac{x_\ell(\alpha_i)}{a_\ell} + b_\ell - \alpha_i + \mu_\ell(\alpha_i) = 0, \quad \ell \in \mathcal{N}_j, \quad (\text{A.15a})$$

$$w_j y_j(\alpha_i + \epsilon) + \frac{x_\ell(\alpha_i + \epsilon)}{a_\ell} + b_\ell - \alpha_i - \epsilon + \mu_\ell(\alpha_i + \epsilon) = 0, \quad \ell \in \mathcal{N}_j. \quad (\text{A.15b})$$

To show that $\mu_i(\alpha_i) = 0$, we show that $\mu_i(\alpha_i) \geq -\epsilon$ if $w_j \geq 0$ and $\mu_i(\alpha_i) \geq -\epsilon \sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell}$ if $w_j < 0$. Since ϵ was chosen arbitrarily, $\sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell} > 0$, and $\mu_i(\alpha_i) \leq 0$ by definition of α_i , this implies in both cases that $\mu_i(\alpha_i) = 0$.

First, if $w_j \geq 0$, then $w_j y_j(\cdot)$ is non-decreasing by Corollary 5.1. Together with Lemma 5.4 and the fact that $\mu_i(\alpha_i + \epsilon) = 0$, subtracting Equation (A.15b) from Equation (A.15a) for $\ell = i$ yields

$$\begin{aligned} 0 &= w_j y_j(\alpha_i) - w_j y_j(\alpha_i + \epsilon) + \frac{x_i(\alpha_i) - x_i(\alpha_i + \epsilon)}{a_i} + \epsilon \\ &\quad + \mu_i(\alpha_i) - \mu_i(\alpha_i + \epsilon) \\ &\leq \epsilon + \mu_i(\alpha_i). \end{aligned}$$

It follows that $\mu_i(\alpha_i) \geq -\epsilon$.

Second, if $w_j < 0$, then we can apply the same proof mechanism as was used in the proof of Lemma 5.4 (see also Equations (A.10) and (A.11)). By multiplying Equations (A.15a) and (A.15b) by a_ℓ and summing them over the index set $\mathcal{N}_j$, we get the following together with Property 5.1 and Corollary 5.2:

$$\begin{aligned} 0 &= \sum_{\ell \in \mathcal{N}_j} (a_\ell w_j y_j(\alpha_i) - a_\ell w_j y_j(\alpha_i + \epsilon) + x_\ell(\alpha_i) - x_\ell(\alpha_i + \epsilon) + \epsilon \\ &\quad + a_\ell \mu_\ell(\alpha_i) - a_\ell \mu_\ell(\alpha_i + \epsilon)) \\ &= \left(1 + w_j \sum_{\ell \in \mathcal{N}} a_\ell \right) (y_j(\alpha_i) - y_j(\alpha_i + \epsilon)) \\ &\quad + \epsilon \sum_{\ell \in \mathcal{N}_j} a_\ell + \sum_{\ell \in \mathcal{N}_j} a_\ell (\mu_\ell(\alpha_i) - \mu_\ell(\alpha_i + \epsilon)) \end{aligned}$$

$$\begin{aligned}
&\leq \epsilon \sum_{\ell \in \mathcal{N}_j} a_\ell + \sum_{\ell \in \mathcal{N}_j} a_\ell (\mu_\ell(\alpha_i) - \mu_\ell(\alpha_i + \epsilon)) \\
&\leq \epsilon \sum_{\ell \in \mathcal{N}_j} a_\ell + \mu_i(\alpha_i).
\end{aligned}$$

Here, the first inequality follows from Property 5.1 and Lemma 5.4 and the second equality follows from Corollary 5.2 and the fact that $\mu_i(\alpha_i + \epsilon) = 0$. It follows that $\mu_i(\alpha_i) \geq -\epsilon \sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell}$. $\square$

A.2.5 PROOF OF LEMMA 5.6

Lemma. For $j \in \mathcal{M}$ and $i, k \in \mathcal{N}_j$, we have:

$$\begin{aligned}
&\gg \frac{l_i}{a_i} + b_i < \frac{l_k}{a_k} + b_k \text{ implies } \alpha_i \leq \alpha_k, \text{ and;} \\
&\gg \frac{u_i}{a_i} + b_i < \frac{u_k}{a_k} + b_k \text{ implies } \beta_i \leq \beta_k.
\end{aligned}$$

Proof. We prove the lemma for the case $\frac{l_i}{a_i} + b_i < \frac{l_k}{a_k} + b_k$; the proof for the case $\frac{u_i}{a_i} + b_i < \frac{u_k}{a_k} + b_k$ is analogous. We show that $x_i(\alpha_k) > l_i$, which implies by definition of α_i that $x_i(\alpha_k) > l_i = x_i(\alpha_i)$. Using Lemma 5.4, this yields $\alpha_i \leq \alpha_k$.

It follows from KKT-condition (5.6a) that

$$\begin{aligned}
w_j y_j(\alpha_k) + \frac{x_k(\alpha_k)}{a_k} + b_k + \mu_k(\alpha_k) &= \alpha_k \\
&= w_j y_j(\alpha_k) + \frac{x_i(\alpha_k)}{a_i} + b_i + \mu_i(\alpha_k).
\end{aligned}$$

Since $\mu_k(\alpha_k) = 0$ by Lemma 5.5 and $x_k(\alpha_k) = l_k$ by definition of α_k , the above is equivalent to

$$\frac{l_k}{a_k} + b_k = \frac{x_i(\alpha_k)}{a_i} + b_i + \mu_i(\alpha_k). \quad (\text{A.16})$$

Suppose that $x_i(\alpha_k) = l_i$. Then $\mu_i(\alpha_k) \leq 0$ by KKT-condition (5.6d). It follows from Equation (A.16) that

$$\frac{l_k}{a_k} + b_k = \frac{l_i}{a_i} + b_i + \mu_i(\alpha_k) < \frac{l_k}{a_k} + b_k,$$

which is a contradiction. Thus, it must hold that $x_i(\alpha_k) > l_i$. $\square$

A.3 PROOFS OF CHAPTER 6

A.3.1 PROOF OF LEMMA 6.3

Lemma. For any instance in I_{sub} , a given $\bar{t} \in \mathcal{T}$, and given multipliers (μ, λ) and corresponding online solution $\hat{x}^{\bar{t}}(\mu, \lambda)$ and local Lagrangian solution $x^{\bar{t}}(\mu, \lambda)$, we

have:

if $\hat{x}^{\bar{t}}(\mu, \lambda) < x^{\bar{t}}(\mu, \lambda)$, we have:

$$\exists \bar{S}^{\bar{t}} \subseteq \{1, \dots, \bar{t}\} \text{ with } \bar{S}^{\bar{t}} \ni \bar{t} \text{ such that } \sum_{t \in \bar{S}^{\bar{t}}} \hat{x}^t(\mu, \lambda) = r(\bar{S}^{\bar{t}}).$$

Proof. Suppose $\hat{x}^{\bar{t}}(\mu, \lambda) < x^{\bar{t}}(\mu, \lambda)$. Since the function $f^{\bar{t}}$ is *strictly* convex, the local Lagrangian solution $x^{\bar{t}}(\mu, \lambda)$ is the *unique* optimal solution to the inner problem of the local Lagrangian dual function $q^{\bar{t}}(\mu, \lambda)$ and thus $L^{\bar{t}}(x^{\bar{t}}(\mu, \lambda), \mu, \lambda) < L^{\bar{t}}(\hat{x}^{\bar{t}}(\mu, \lambda), \mu, \lambda)$. This implies that $L^{\bar{t}}(\cdot, \mu, \lambda)$ is decreasing on the interval $[\hat{x}^{\bar{t}}(\mu, \lambda), x^{\bar{t}}(\mu, \lambda)]$ since any solution in this interval is feasible for the inner problem of $q^{\bar{t}}(\mu, \lambda)$ and $x^{\bar{t}}(\mu, \lambda)$ is optimal for this problem.

Consider a vector $y \in \mathbb{R}^T$ for which we have that $y^t = \hat{x}^t(\mu, \lambda)$ for $t < \bar{t}$ and $y^{\bar{t}} \in (\hat{x}^{\bar{t}}(\mu, \lambda), x^{\bar{t}}(\mu, \lambda))$. If this vector would have been a feasible solution, the value $y^{\bar{t}}$ had been chosen as online solution for stage $\bar{t}$ since $L^{\bar{t}}(y^{\bar{t}}, \mu, \lambda) < L^{\bar{t}}(\hat{x}^{\bar{t}}, \mu, \lambda)$. In particular, this means that there is no $\epsilon \in (0, x^{\bar{t}}(\mu, \lambda) - \hat{x}^{\bar{t}}(\mu, \lambda))$ such that for some $s > \bar{t}$ the solution wherein we move a positive amount of ϵ from $\hat{x}^s(\mu, \lambda)$ to $\hat{x}^{\bar{t}}(\mu, \lambda)$, i.e., the solution $\bar{v} := (\bar{v}^t)_{t \in \mathcal{T}}$ given by

$$\bar{v}^t = \begin{cases} \hat{x}^{\bar{t}}(\mu, \lambda) + \epsilon & \text{if } t = \bar{t}, \\ \hat{x}^s(\mu, \lambda) - \epsilon & \text{if } t = s, \\ \hat{x}^t(\mu, \lambda) & \text{otherwise,} \end{cases}$$

is feasible. Since $\sum_{t \in \mathcal{T}} \bar{v}^t = r(\mathcal{T})$, this implies that there is at least one inequality constraint that is violated by $\bar{v}$. Since ϵ may be arbitrary close to 0, it follows for the online solution $\hat{x}(\mu, \lambda)$ that there is at least one inequality constraint involving $\hat{x}^{\bar{t}}(\mu, \lambda)$ but not involving $\hat{x}^s(\mu, \lambda)$ that is tight in this online solution. In other words, for each $s > \bar{t}$, there is a subset $\hat{\mathcal{V}}^s \subset \mathcal{T}$ such that $\bar{t} \in \hat{\mathcal{V}}^s$, $s \notin \hat{\mathcal{V}}^s$, and $\sum_{t \in \hat{\mathcal{V}}^s} \hat{x}^t(\mu, \lambda) = r(\hat{\mathcal{V}}^s)$. We denote the intersection of these sets by $\hat{\mathcal{V}}$, i.e., $\hat{\mathcal{V}} := \cap_{s > \bar{t}} \hat{\mathcal{V}}^s$. Note that $\hat{\mathcal{V}}$ is not empty since each set $\hat{\mathcal{V}}^s$ contains $\bar{t}$.

We claim that the set $\hat{\mathcal{V}}$ satisfies all properties of $\bar{S}^{\bar{t}}$ that are required by the lemma, which implies that such a set exists. First, we have that $\bar{t} \in \hat{\mathcal{V}}$ since each subset $\hat{\mathcal{V}}^s$ contains $\bar{t}$. Second, since for each $s > \bar{t}$ the subset $\hat{\mathcal{V}}^s$ does not contain s , the intersection $\hat{\mathcal{V}}$ cannot contain any indices exceeding $\bar{t}$. Third, since for each $s > \bar{t}$ the constraint corresponding to $\hat{\mathcal{V}}^s$ is tight in the online solution $\hat{x}(\mu, \lambda)$, Lemma 6.1 implies that also the constraint corresponding to $\hat{\mathcal{V}}$ is tight in $\hat{x}(\mu, \lambda)$. $\square$

A.3.2 PROOF OF LEMMA 6.4

Lemma. For any instance in I_{sub} and given any Lagrange multipliers (μ, λ) such that $(\mu, \lambda) \leq (\mu^*, \lambda^*)$, it holds that $\hat{x}^t(\mu, \lambda) \leq x^t(\mu^*, \lambda^*)$ for all $t \in \mathcal{T}$.

Proof. Consider any Lagrange multipliers (μ, λ) such that $(\mu, \lambda) \leq (\mu^*, \lambda^*)$. We prove the lemma by induction on the stage index $\bar{t}$. First, we consider the case $\bar{t} = 1$. Observe that, when defining the rank function r , we can assume without loss of generality that $r(\{1\})$ is the maximum value of x^1 in any feasible solution x in $\mathcal{F}$. As a consequence, the local Lagrangian solution $x^1(\mu, \lambda)$ cannot lead to any future constraint violation. Hence, $\hat{x}^1(\mu, \lambda) = x^1(\mu, \lambda)$, which proves the case $\bar{t} = 1$.

Second, suppose that $\hat{x}^s(\mu, \lambda) \leq x^s(\mu, \lambda)$ for all s smaller than some $\bar{t} \in \mathcal{T}$ with $\bar{t} > 1$. We prove that this implies $\hat{x}^{\bar{t}}(\mu, \lambda) \leq x^{\bar{t}}(\mu, \lambda)$ by contradiction. For this, suppose that $\hat{x}^{\bar{t}}(\mu, \lambda) > x^{\bar{t}}(\mu, \lambda)$. It follows from an argument analogous to the first part of the proof of Lemma 6.3 in Appendix A.3.1 that there is no ϵ with $0 < \epsilon < \hat{x}^{\bar{t}}(\mu, \lambda) - x^{\bar{t}}(\mu, \lambda)$ such that for some $s > \bar{t}$ the solution wherein we move an amount of ϵ from $\hat{x}^{\bar{t}}(\mu, \lambda)$ to $\hat{x}^s(\mu, \lambda)$, i.e., the solution $\bar{w} := (\bar{w}^t)_{t \in \mathcal{T}}$ given by

$$\bar{w}^t = \begin{cases} \hat{x}^{\bar{t}}(\mu, \lambda) - \epsilon & \text{if } t = \bar{t}, \\ \hat{x}^s(\mu, \lambda) + \epsilon & \text{if } t = s, \\ \hat{x}^t(\mu, \lambda) & \text{otherwise,} \end{cases}$$

is infeasible. Since $\sum_{t \in \mathcal{T}} \bar{w}^t = r(\mathcal{T})$, this implies for the online solution $\hat{x}(\mu, \lambda)$ that there must be at least one inequality constraint involving $\hat{x}^s(\mu, \lambda)$ but not involving $\hat{x}^{\bar{t}}(\mu, \lambda)$ that is tight in this online solution. In other words, for each $s > \bar{t}$, there exists a subset $\hat{\mathcal{S}}^s \subset \mathcal{T}$ such that $s \in \hat{\mathcal{S}}^s$, $\bar{t} \notin \hat{\mathcal{S}}^s$, and $\sum_{t \in \hat{\mathcal{S}}^s} \hat{x}^t(\mu, \lambda) = r(\hat{\mathcal{S}}^s)$.

Let $\bar{\mathcal{T}}$ denote the set of stages whose local Lagrangian and online solution are not the same, i.e., $\bar{\mathcal{T}} := \{t \in \mathcal{T} \mid \hat{x}^t(\mu, \lambda) \neq x^t(\mu, \lambda)\}$. By the induction hypothesis and Lemma 6.3, it follows that for each $s \in \bar{\mathcal{T}}$ with $s < \bar{t}$ there exists a set $\bar{\mathcal{S}}^s \subseteq \{1, \dots, s\}$ with $s \in \bar{\mathcal{S}}^s$ such that $\sum_{s \in \bar{\mathcal{S}}} \hat{x}^s(\mu, \lambda) = r(\bar{\mathcal{S}}^s)$. Thus, by Lemma 6.1, also the inequality constraint corresponding to the set

$$\mathcal{S}' := \left(\bigcup_{s \in \bar{\mathcal{T}}, s < \bar{t}} \bar{\mathcal{S}}^s \right) \cup \left(\bigcup_{s > \bar{t}} \hat{\mathcal{S}}^s \right)$$

is tight in the online solution $\hat{x}(\mu, \lambda)$, i.e., $\sum_{t \in \mathcal{S}'} \hat{x}^t(\mu, \lambda) = r(\mathcal{S}')$. Note, that

- » $\{\bar{t} + 1, \dots, T\} \subseteq \mathcal{S}'$ since for each $s > \bar{t}$ the set $\hat{\mathcal{S}}^s$ contains s ;
- » $\bar{t} \notin \mathcal{S}'$ since for each $s \in \bar{\mathcal{T}}$ with $s < \bar{t}$ we have $\bar{\mathcal{S}}^s \subseteq \{1, \dots, s\}$ and for each $s > \bar{t}$ we have by definition that $\bar{t} \notin \hat{\mathcal{S}}^s$;
- » $s \in \mathcal{S}'$ for all $s \in \bar{\mathcal{T}}$ with $s < \bar{t}$ since each set $\bar{\mathcal{S}}^s$ contains s .

It follows that $(\mathcal{T} \setminus \mathcal{S}') \subseteq \{1, \dots, \bar{t}\}$, $\bar{t} \in (\mathcal{T} \setminus \mathcal{S}')$, and $(\mathcal{T} \setminus \mathcal{S}') \cap \bar{\mathcal{T}} = \emptyset$. We use this information to derive the following inequality:

$$r(\mathcal{T}) - r(\mathcal{S}') = \sum_{t \in \mathcal{T} \setminus \mathcal{S}'} \hat{x}^t(\mu, \lambda) \tag{A.17a}$$

$$= \sum_{t \in \mathcal{T} \setminus \mathcal{S}', t \neq \bar{t}} x^t(\mu, \lambda) + \hat{x}^{\bar{t}}(\mu, \lambda) \quad (\text{A.17b})$$

$$\geq \sum_{t \in \mathcal{T} \setminus \mathcal{S}', t \neq \bar{t}} x^t(\mu^*, \lambda^*) + \hat{x}^{\bar{t}}(\mu, \lambda) \quad (\text{A.17c})$$

$$= r(\mathcal{T}) - \sum_{t \in \mathcal{S}'} x^t(\mu^*, \lambda^*) - x^{\bar{t}}(\mu^*, \lambda^*) + \hat{x}^{\bar{t}}(\mu, \lambda) \quad (\text{A.17d})$$

$$\geq r(\mathcal{T}) - r(\mathcal{S}') - x^{\bar{t}}(\mu^*, \lambda^*) + \hat{x}^{\bar{t}}(\mu, \lambda). \quad (\text{A.17e})$$

Here, (A.17a) follows since the constraints corresponding to $\mathcal{T}$ and $\mathcal{S}'$ are tight in $\hat{x}(\mu, \lambda)$, i.e., $\sum_{t \in \mathcal{T}} \hat{x}^t(\mu, \lambda) = r(\mathcal{T})$ and $\sum_{t \in \mathcal{S}'} \hat{x}^t(\mu, \lambda) = r(\mathcal{S}')$, (A.17b) follows since $t \notin \bar{\mathcal{T}}$ for each $t \in \mathcal{T} \setminus \mathcal{S}'$ with $t < \bar{t}$, (A.17c) follows by Lemma 6.2, and (A.17d) and (A.17e) follow due to feasibility of the optimal solution $x(\mu^*, \lambda^*)$ for the submodular constraints. The inequality derived in (A.17) implies that $x^{\bar{t}}(\mu^*, \lambda^*) \geq \hat{x}^{\bar{t}}(\mu, \lambda)$. However, since we assumed that $\hat{x}^{\bar{t}}(\mu, \lambda) > x^{\bar{t}}(\mu, \lambda)$, this implies that $x^{\bar{t}}(\mu^*, \lambda^*) > x^{\bar{t}}(\mu, \lambda)$, which is a contradiction with Lemma 6.2. Thus, we must have that $\hat{x}^{\bar{t}}(\mu, \lambda) \leq x^{\bar{t}}(\mu, \lambda)$. $\square$

ACRONYMS

235

B	Box	box constraints
C	CDF CoV	cumulative distribution function coefficient of variation
D	DEM	decentralized energy management
E	EV EV-3Phase	electric vehicle three-phase electric vehicle charging
F	FME	Fourier-Motzkin elimination
G	GBC	generalized bound constraints
I	ICT	information and communication technology
K	KKT	Karush-Kuhn-Tucker
L	LC	laminar constraints
N	NC	nested constraints
O	OCO ODDO	online convex optimization online duality-driven optimization
P	PV	photovoltaic
Q	QRA QRA-NC QRA-NonSep-GBC	quadratic resource allocation quadratic resource allocation with nested constraints nonseparable quadratic resource allocation with GBC
R	RAP RES RO RSA	resource allocation problem renewable energy sources robust optimization recursive smoothing algorithm
S	SC SoC SP	submodular constraints state-of-charge stochastic programming

BIBLIOGRAPHY

237

- [1] BP statistical review of world energy 2020. URL <https://www.bp.com/en/global/corporate/energy-economics/statistical-review-of-world-energy.html>. Accessed on November 15, 2020. (Cited on page 1).
- [2] Climate watch data explorer. URL <https://www.climatewatchdata.org/data-explorer>. Accessed on November 15, 2020. (Cited on page 4).
- [3] New Nissan Leaf prices and specifications. URL <https://www.nissan.co.uk/vehicles/new-vehicles/leaf/prices-specifications.html>. Accessed on September 5, 2018. (Cited on pages 174 and 205).
- [4] Tesla Model 3 Standard Range price and specifications - EV database, 2020. URL <https://ev-database.org/car/1060/Tesla-Model-3-Standard-Range>. Accessed on March 4, 2020. (Cited on page 117).
- [5] P. T. Akhil and R. Sundaresan. Algorithms for separable convex optimization with linear ascending constraints. *Sādhanā*, 43(9):146, 2018. ISSN 0973-7677. doi: 10.1007/s12046-018-0890-2. (Cited on pages 17, 24, 25, 28, and 63).
- [6] A. Albert and L. Zhang. A novel definition of the multivariate coefficient of variation. *Biometrical Journal*, 52(5):667–675, 2010. ISSN 0323-3847. doi: 10.1002/bimj.201000030. (Cited on page 161).
- [7] A. Alexandrescu. Fast deterministic selection. In C. S. Iliopoulos, S. P. Pissis, S. J. Puglisi, and R. Raman, editors, *Leibniz International Proceedings in Informatics*, volume 75, pages 24:1–24:9, 2017. ISBN 9783959770361. doi: 10.4230/LIPIcs.SEA.2017.24. (Cited on pages 22, 55, 62, and 66).
- [8] E. Apostolaki-Iosifidou, P. Codani, and W. Kempton. Measurement of power loss during electric vehicle charging and discharging. *Energy*, 127:730–742, 2017. ISSN 0360-5442. doi: 10.1016/j.energy.2017.03.015. (Cited on pages 184 and 205).
- [9] A. Arvind and C. Rangan. Symmetric min-max heap: A simpler data structure for double-ended priority queue. *Information Processing Letters*, 69(4):197–199, 1999. ISSN 0020-0190. doi: 10.1016/S0020-0190(99)00014-9. (Cited on pages 83 and 212).
- [10] M. Atiyah. Duality in mathematics and physics, 2007. URL https://fme.upc.edu/ca/arxiu/butlleti-digital/riemann/071218_conferencia_atiyah-d_article.pdf. Transcription of a talk delivered at the Institut de Matemàtica de la Universitat de Barcelona. (Cited on page 7).

- [11] M. D. Atkinson, J.-R. Sack, N. Santoro, and T. Strothotte. Min-max heaps and generalized priority queues. *Communications of the ACM*, 29(10):996–1000, 1986. ISSN 0001-0782. doi: 10.1145/6617.6621. (Cited on pages 63 and 80).
- [12] F. Bach. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends® in Machine Learning*, 6(2-3):145–373, 2013. ISSN 1935-8237. doi: 10.1561/22000000039. (Cited on pages 7, 17, 27, and 28).
- [13] F. R. Bach. Structured sparsity-inducing norms through submodular functions. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 118–126. 2010. doi: 10.5555/2997189.2997203. (Cited on page 17).
- [14] H. Bakker, F. Dunke, and S. Nickel. A structuring review on multi-stage optimization under uncertainty: Aligning concepts from theory and practice. *Omega*, 96:102080, 2020. ISSN 0305-0483. doi: 10.1016/j.omega.2019.06.006. (Cited on page 136).
- [15] E. Bampis, C. Dürr, F. Kacem, and I. Milis. Speed scaling with power down scheduling for agreeable deadlines. *Sustainable Computing: Informatics and Systems*, 2(4):184–189, 2012. ISSN 2210-5379. doi: 10.1016/j.suscom.2012.10.003. (Cited on page 37).
- [16] A. Barbato and A. Capone. Optimization models and methods for demand-side management of residential users: A survey. *Energies*, 7(9):5787–5824, 2014. ISSN 1996-1073. doi: 10.3390/en7095787. (Cited on page 134).
- [17] M. S. Bartlett. An inverse matrix adjustment arising in discriminant analysis. *The Annals of Mathematical Statistics*, 22(1):107–111, 1951. ISSN 0003-4851. doi: 10.1214/aoms/1177729698. (Cited on page 105).
- [18] S. I. Bastrakov and N. Y. Zolotykh. Fast method for verifying Chernikov rules in Fourier-Motzkin elimination. *Computational Mathematics and Mathematical Physics*, 55(1):160–167, 2015. ISSN 1555-6662. doi: 10.1134/S0965542515010042. (Cited on page 144).
- [19] M. Beaudin and H. Zareipour. Home energy management systems: A review of modelling and complexity. *Renewable and Sustainable Energy Reviews*, 45:318–335, 2015. ISSN 1364-0321. doi: 10.1016/j.rser.2015.01.046. (Cited on page 139).
- [20] L. Belkhir and A. Elmeligi. Assessing ICT global emissions footprint: Trends to 2040 & recommendations. *Journal of Cleaner Production*, 177:448 – 463, 2018. ISSN 0959-6526. doi: 10.1016/j.jclepro.2017.12.239. (Cited on page 5).
- [21] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski. Adjustable robust solutions of uncertain linear programs. *Mathematical Programming*, 99(2):351–376, 2004. ISSN 1436-4646. doi: 10.1007/s10107-003-0454-y. (Cited on pages 135, 156, and 157).
- [22] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust optimization*. Princeton University Press, Princeton, NJ, 1st edition, 2009. ISBN 978-0-691-14368-2. (Cited on page 136).

- [23] A. Ben-Tal, D. den Hertog, and J.-P. Vial. Deriving robust counterparts of non-linear uncertain inequalities. *Mathematical Programming*, 149(1):265–299, 2015. ISSN 1436-4646. doi: 10.1007/s10107-014-0750-8. (Cited on page 136).
- [24] D. P. Bertsekas and J. Tsitsiklis. *Parallel and distributed computation: Numerical methods*. Athena Scientific, Belmont, MA, 1st edition, 1997. ISBN 1-886529-01-9. (Cited on page 142).
- [25] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical Programming*, 98(1):49–71, 2003. ISSN 1436-4646. doi: 10.1007/s10107-003-0396-4. (Cited on page 134).
- [26] D. Bertsimas and B. Stellato. The voice of optimization. *Machine Learning*, 110: 249–277, 2021. doi: 10.1007/s10994-020-05893-5. (Cited on page 137).
- [27] D. J. Bertsimas and J. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, Belmont, MA, 1st edition, 1997. ISBN 1886529191. (Cited on page 144).
- [28] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4):448–461, 1973. ISSN 0022-0000. doi: 10.1016/S0022-0000(73)80033-9. (Cited on pages 22, 54, 55, 62, and 113).
- [29] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, MA, 7th edition, 2004. ISBN 978-0-521-83378-3. (Cited on pages 7, 92, 100, 137, 141, 168, and 217).
- [30] P. Brass. *Advanced data structures*. Cambridge University Press, Cambridge, MA, 1st edition, 2008. ISBN 9780511800191. doi: 10.1017/CBO9780511800191. (Cited on pages 63 and 212).
- [31] G. S. Brodal. A survey on priority queues. In A. Brodnik, A. López-Ortiz, V. Raman, and A. Viola, editors, *Space-Efficient Data Structures, Streams, and Algorithms*, pages 150–163. Berlin, Heidelberg, 2013. ISBN 978-3-642-40273-9. doi: 10.1007/978-3-642-40273-9_11. (Cited on page 63).
- [32] P. Brucker. An $O(n)$ algorithm for quadratic knapsack problems. *Operations Research Letters*, 3(3):163–166, 1984. ISSN 0167-6377. doi: 10.1016/0167-6377(84)90010-5. (Cited on pages 21, 22, and 23).
- [33] D. Chakrabarty, P. Jain, and P. Kothari. Provable submodular minimization using Wolfe’s algorithm. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems* 27, pages 802–809. 2014. doi: 10.5555/2968826.2968916. (Cited on page 27).
- [34] N. Chen, C. W. Tan, and T. Q. S. Quek. Electric vehicle charging in smart grid: optimality and valley-filling algorithms. *IEEE Journal of Selected Topics in Signal Processing*, 8(6):1073–1083, 2014. ISSN 1932-4553. doi: 10.1109/JSTSP.2014.2334275. (Cited on pages 8 and 138).
- [35] P. L. Combettes. Perspective functions: Properties, constructions, and examples. *Set-Valued and Variational Analysis*, 26(2):247–264, 2018. ISSN 1877-0541. doi: 10.1007/s11228-017-0407-x. (Cited on page 17).

- [36] P. L. Combettes and C. L. Müller. Perspective functions: Proximal calculus and applications in high-dimensional statistics. *Journal of Mathematical Analysis and Applications*, 457(2):1283–1306, 2018. ISSN 0022-247X. doi: 10.1016/j.jmaa.2016.12.021. (Cited on page 17).
- [37] J. Cook, N. Oreskes, P. T. Doran, W. R. L. Anderegg, B. Verheggen, E. W. Maibach, J. S. Carlton, S. Lewandowsky, A. G. Skuce, S. A. Green, D. Nuccitelli, P. Jacobs, M. Richardson, B. Winkler, R. Painting, and K. Rice. Consensus on consensus: a synthesis of consensus estimates on human-caused global warming. *Environmental Research Letters*, 11(4):048002, 2016. doi: 10.1088/1748-9326/11/4/048002. (Cited on page 1).
- [38] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. The MIT Press, Cambridge, MA, 3rd edition, 2009. ISBN 9780262033848. (Cited on page 81).
- [39] S. Cosares and D. S. Hochbaum. Strongly polynomial algorithms for the quadratic transportation problem with a fixed number of sources. *Mathematics of Operations Research*, 19(1):94–111, 1994. ISSN 0364-765X. doi: 10.1287/moor.19.1.94. (Cited on page 23).
- [40] Y.-H. Dai and R. Fletcher. New algorithms for singly linearly constrained quadratic programs subject to lower and upper bounds. *Mathematical Programming*, 106(3):403–421, 2006. doi: 10.1007/s10107-005-0595-2. (Cited on page 17).
- [41] A. A. D’Amico, L. Sanguinetti, and D. P. Palomar. Convex separable problems with linear constraints in signal processing and communications. *IEEE Transactions on Signal Processing*, 62(22):6045–6058, 2014. ISSN 1941-0476 VO - 62. doi: 10.1109/TSP.2014.2360143. (Cited on page 33).
- [42] V. De Angelis. Linear programming with uncertain objective function: Minimax solution for relative loss. *Calcolo*, 16(2):125–141, 1979. ISSN 1126-5434. doi: 10.1007/BF02575922. (Cited on page 134).
- [43] J. G. Deshpande, E. Kim, and M. Thottan. Differentiated services QoS in smart grid communication networks. *Bell Labs Technical Journal*, 16(3):61–81, 2011. ISSN 1089-7089 VO - 16. doi: 10.1002/bltj.20522. (Cited on page 119).
- [44] M. Dyer and L. Stougie. Computational complexity of stochastic programming problems. *Mathematical Programming*, 106(3):423–432, 2006. ISSN 1436-4646. doi: 10.1007/s10107-005-0597-0. (Cited on page 136).
- [45] G. N. Frederickson and D. B. Johnson. The complexity of selection and ranking in $X + Y$ and matrices with sorted columns. *Journal of Computer and System Sciences*, 24(2):197–208, 1982. ISSN 0022-0000. doi: 10.1016/0022-0000(82)90048-4. (Cited on pages 21 and 22).
- [46] U. Friedrich, R. Münnich, S. de Vries, and M. Wagner. Fast integer-valued algorithms for optimal allocations under constraints in stratified sampling. *Computational Statistics & Data Analysis*, 92:1–12, 2015. ISSN 0167-9473. doi: 10.1016/j.csda.2015.06.003. (Cited on pages 17 and 35).

- [47] S. Fujishige. Lexicographically optimal base of a polymatroid with respect to a weight vector. *Mathematics of Operations Research*, 5(2):186–196, 1980. ISSN 0364-765X. doi: 10.1287/moor.5.2.186. (Cited on pages 27 and 28).
- [48] S. Fujishige. Structures of polyhedra determined by submodular functions on crossing families. *Mathematical Programming*, 29(2):125–141, 1984. ISSN 0025-5610. doi: 10.1007/BF02592217. (Cited on page 39).
- [49] S. Fujishige. *Submodular functions and optimization*, volume 58 of *Annals of Discrete Mathematics*. Elsevier, 2nd edition, 2005. ISBN 9780080461625. (Cited on pages 7, 17, 27, 30, 39, 135, 149, and 173).
- [50] S. Fujishige and S. Isotani. A submodular function minimization algorithm based on the minimum-norm base. *Pacific Journal of Optimization*, 7(1):3–17, 2011. (Cited on pages 27 and 28).
- [51] H. N. Gabow and R. E. Tarjan. A linear-time algorithm for a special case of disjoint set union. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing (STOC)*, pages 246–251, Boston, MA, 1983. ISBN 0-89791-099-0. doi: 10.1145/800061.808753. (Cited on page 81).
- [52] H. N. Gabow and R. E. Tarjan. A linear-time algorithm for a special case of disjoint set union. *Journal of Computer and System Sciences*, 30(2):209–221, 1985. ISSN 0022-0000. doi: 10.1016/0022-0000(85)90014-5. (Cited on page 81).
- [53] Z. Galil and G. F. Italiano. Data structures and algorithms for disjoint set union problems. *ACM Computing Surveys (CSUR)*, 23(3):319–344, 1991. ISSN 0360-0300. doi: 10.1145/116873.116878. (Cited on page 81).
- [54] L. Gan, U. Topcu, and S. H. Low. Optimal decentralized protocol for electric vehicle charging. *IEEE Transactions on Power Systems*, 28(2):940–951, 2013. ISSN 0885-8950. doi: 10.1109/TPWRS.2012.2210288. (Cited on page 94).
- [55] M. E. T. Gerards. *Algorithmic power management: Energy minimisation under real-time constraints*. PhD thesis, University of Twente, 2014. (Cited on pages 29, 37, and 63).
- [56] M. E. T. Gerards and J. L. Hurink. Robust peak-shaving for a neighborhood with electric vehicles. *Energies*, 9(8):594, 2016. ISSN 1996-1073. doi: 10.3390/en9080594. (Cited on pages 138 and 176).
- [57] M. E. T. Gerards, J. L. Hurink, P. K. F. Hölzenspies, J. Kuper, and G. J. M. Smit. Analytic clock frequency selection for global DVFS. In *2014 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pages 512–519, Turin, 2014. ISBN 2377-5750 VO -. doi: 10.1109/PDP.2014.103. (Cited on page 38).
- [58] M. E. T. Gerards, H. A. Toersche, G. Hoogsteen, T. van der Klauw, J. L. Hurink, and G. J. M. Smit. Demand side management using profile steering. In *2015 IEEE Eindhoven PowerTech*, Eindhoven, 2015. IEEE. ISBN 978-1-4799-7693-5. doi: 10.1109/PTC.2015.7232328. (Cited on pages 34, 63, and 94).

- [59] M. E. T. Gerards, J. L. Hurink, and P. K. F. Hölzenspies. A survey of offline algorithms for energy minimization under deadline constraints. *Journal of Scheduling*, 19(1):3–19, 2016. ISSN 1099-1425. doi: 10.1007/s10951-015-0463-8. (Cited on pages 8, 37, and 134).
- [60] M. E. T. Gerards, J. L. Hurink, and R. Hübner. Demand side management in a field test: Lessons learned. *CIREN - Open Access Proceedings Journal*, 2017(1): 1678–1681, 2017. ISSN 2515-0855. doi: 10.1049/oap-cired.2017.1238. (Cited on page 174).
- [61] J. Gondzio. Interior point methods 25 years later. *European Journal of Operational Research*, 218(3):587–601, 2012. ISSN 0377-2217. doi: 10.1016/j.ejor.2011.09.017. (Cited on page 22).
- [62] B. L. Gorissen, I. Yanıkoğlu, and D. den Hertog. A practical guide to robust optimization. *Omega*, 53:124–137, 2015. ISSN 0305-0483. doi: 10.1016/j.omega.2014.12.006. (Cited on page 136).
- [63] M. C. Grant and S. P. Boyd. Graph implementations for nonsmooth convex programs. In V. D. Blondel, S. P. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, pages 95–110, London, 2008. ISBN 978-1-84800-155-8. doi: 10.1007/978-1-84800-155-8_7. (Cited on page 159).
- [64] H. Groenevelt. Two algorithms for maximizing a separable concave function over a polymatroid feasible region. *European Journal of Operational Research*, 54(2): 227–236, 1991. ISSN 0377-2217. doi: 10.1016/0377-2217(91)90300-K. (Cited on pages 17, 27, and 30).
- [65] L. Grüne and J. Pannek. Nonlinear model predictive control. In L. Grüne and J. Pannek, editors, *Nonlinear model predictive control: Theory and algorithms*, chapter 3, pages 45–69. Springer International Publishing, Cham, 2017. ISBN 978-3-319-46024-6. doi: 10.1007/978-3-319-46024-6_3. (Cited on page 158).
- [66] T. Harks, M. Klimm, and B. Peis. Resource competition on integral polymatroids. In T.-Y. Liu, Q. Qi, and Y. Ye, editors, *10th International Conference on Web and Internet Economics*, pages 189–202, Beijing, 2014. ISBN 978-3-319-13129-0. doi: 10.1007/978-3-319-13129-0_14. (Cited on page 17).
- [67] D. A. Harville. *Matrix algebra from a statistician's perspective*. Springer Verlag, New York, NY, 1st edition, 1997. ISBN 978-0-387-22677-4. doi: 10.1007/b98818. (Cited on page 224).
- [68] J. Haucap and B. Pagel. Ausbau der Stromnetze im Rahmen der Energiewende: Effizienter Netzausbau und Struktur der Netznutzungsentgelte. *List Forum für Wirtschafts- und Finanzpolitik*, 39(3):235–253, 2013. ISSN 2364-3943. doi: 10.1007/BF03373052. In German. (Cited on page 2).
- [69] E. Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016. ISSN 2167-3888. doi: 10.1561/2400000013. (Cited on page 143).

- [70] P. He, L. Zhao, S. Zhou, and Z. Niu. Water-filling: A geometric approach and its application to solve generalized radio resource allocation problems. *IEEE Transactions on Wireless Communications*, 12(7):3637–3647, 2013. ISSN 1558-2248 VO - 12. doi: 10.1109/TWC.2013.061713.130278. (Cited on page 33).
- [71] S. He, J. Zhang, and S. Zhang. Polymatroid optimization, submodularity, and joint replenishment games. *Operations Research*, 60(1):128–137, 2012. ISSN 0030-364X. doi: 10.1287/opre.1110.1000. (Cited on page 17).
- [72] R. Helgason, J. Kennington, and H. Lall. A polynomially bounded algorithm for a singly constrained quadratic program. *Mathematical Programming*, 18(1):338–343, 1980. doi: 10.1007/BF01588328. (Cited on page 51).
- [73] N. Ho-Nguyen and F. Kılınç-Karzan. Exploiting problem structure in optimization under uncertainty via online convex optimization. *Mathematical Programming*, 177(1):113–147, 2019. ISSN 1436-4646. doi: 10.1007/s10107-018-1262-8. (Cited on page 137).
- [74] D. S. Hochbaum. Lower and upper bounds for the allocation problem and other nonlinear optimization problems. *Mathematics of Operations Research*, 19(2):390–409, 1994. ISSN 0364765X. doi: 10.1287/moor.19.2.390. (Cited on pages 17, 21, 22, 23, 25, 27, 89, and 99).
- [75] D. S. Hochbaum and S.-P. Hong. About strongly polynomial time algorithms for quadratic optimization over submodular constraints. *Mathematical Programming*, 69:269–309, 1995. ISSN 1436-4646. doi: 10.1007/BF01585561. (Cited on pages 7, 26, 81, 89, 125, and 135).
- [76] D. S. Hochbaum and J. G. Shanthikumar. Convex separable optimization is not much harder than linear optimization. *Journal of the ACM*, 37(4):843–862, 1990. ISSN 0004-5411. doi: 10.1145/96559.96597. (Cited on pages 25 and 26).
- [77] K. Hoekstra. *On the realization of a smart grid demo-site at Coteq in Almelo*. M.Sc. thesis, University of Twente, 2018. (Cited on pages 196 and 211).
- [78] T. Hong and S. Fan. Probabilistic electric load forecasting: A tutorial review. *International Journal of Forecasting*, 32(3):914–938, 2016. ISSN 0169-2070. doi: 10.1016/j.ijforecast.2015.11.011. (Cited on pages 134 and 139).
- [79] G. Hoogsteen. *A cyber-physical systems perspective on decentralized energy management*. PhD thesis, University of Twente, 2017. (Cited on page 4).
- [80] G. Hoogsteen, A. Molderink, J. L. Hurink, G. J. Smit, B. Kootstra, and F. Schuring. Charging electric vehicles, baking pizzas, and melting a fuse in Lochem. *CIREN - Open Access Proceedings Journal*, 2017(1):1629–1633, 2017. ISSN 2515-0855. doi: 10.1049/oap-cired.2017.0340. (Cited on pages 2, 82, 89, 94, 95, 117, and 211).
- [81] G. Hoogsteen, M. E. T. Gerards, and J. L. Hurink. On the scalability of decentralized energy management using profile steering. In *2018 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, Sarajevo, 2018. ISBN 978-1-5386-4505-5. doi: 10.1109/ISGTEurope.2018.8571823. (Cited on pages 64 and 87).

- [82] G. Hoogsteen, J. L. Hurink, and G. J. M. Smit. DEMKit: A decentralized energy management simulation and demonstration toolkit. In *2019 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*, Bucharest, 2019. ISBN 978-1-5386-8218-0. doi: 10.1109/ISGT-Europe.2019.8905439. (Cited on pages 63 and 81).
- [83] W. Huang and Y. Wang. An optimal speed control scheme supported by media servers for low-power multimedia applications. *Multimedia Systems*, 15(2):113–124, 2009. ISSN 1432-1882. doi: 10.1007/s00530-009-0153-5. (Cited on pages 29 and 63).
- [84] L. M. Hvattum, I. Norstad, K. Fagerholt, and G. Laporte. Analysis of an exact algorithm for the vessel speed optimization problem. *Networks*, 62(2):132–135, 2013. ISSN 0028-3045. doi: 10.1002/net.21503. (Cited on pages 29, 36, 63, and 134).
- [85] T. Ibaraki and N. Katoh. *Resource allocation problems: Algorithmic approaches*. The MIT Press, Cambridge, MA, 1st edition, 1988. ISBN 978-0-262-09027-8. (Cited on pages 7, 18, 21, 22, 67, and 116).
- [86] J. Ide and A. Schöbel. Robustness for uncertain multi-objective optimization: a survey and analysis of different concepts. *OR Spectrum*, 38(1):235–271, 2016. ISSN 1436-6304. doi: 10.1007/s00291-015-0418-7. (Cited on page 134).
- [87] IEC-61851-1. Electric vehicle conductive charging system - Part 1: General requirements, 2017. (Cited on pages 184 and 211).
- [88] International Energy Agency. Transport sector CO₂ emissions by mode in the sustainable development scenario, 2000-2030. URL <https://www.iea.org/data-and-statistics/charts/transport-sector-co2-emissions-by-mode-in-the-sustainable-development-scenario-2000-2030>. Accessed on November 15, 2020. (Cited on page 4).
- [89] S. Irani, S. Shukla, and R. Gupta. Algorithms for power savings. *ACM Transactions on Algorithms*, 3(4):41:1–41:23, 2007. ISSN 1549-6325. doi: 10.1145/1290672.1290678. (Cited on page 37).
- [90] K. Jain and V. V. Vazirani. Eisenberg–Gale markets: Algorithms and game-theoretic properties. *Games and Economic Behavior*, 70(1):84–106, 2010. ISSN 0899-8256. doi: 10.1016/j.geb.2008.11.011. (Cited on page 17).
- [91] F. Javed, N. Arshad, F. Wallin, I. Vassileva, and E. Dahlquist. Forecasting for demand response in smart grids: An analysis on use of anthropologic and structural data and short term multiple loads forecasting. *Applied Energy*, 96:150–160, 2012. ISSN 03062619. doi: 10.1016/j.apenergy.2012.02.027. (Cited on pages 134, 139, and 173).
- [92] R. Jenatton, J. Huang, and C. Archambeau. Adaptive algorithms for online convex optimization with long-term constraints. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pages 402–411, 2016. doi: 10.5555/3045390.3045434. (Cited on page 137).
- [93] H. Kaplan, L. Kozma, O. Zamir, and U. Zwick. Selection from heaps, row-sorted matrices, and X+Y using soft heaps. In J. T. Fineman and M. Mitzenmacher, editors, *2nd Symposium on Simplicity in Algorithms (SOSA 2019)*, pages 5:1–5:21, San Diego, 2019. doi: 10.4230/OASlcs.SOSA.2019.5. (Cited on page 23).

- [94] N. Katoh, A. Shioura, and T. Ibaraki. Resource allocation problems. In P. M. Pardalos, D.-Z. Du, and R. L. Graham, editors, *Handbook of Combinatorial Optimization*, pages 2897–2988. Springer, New York, NY, 2nd edition, 2013. ISBN 978-1-4419-7997-1. doi: 10.1007/978-1-4419-7997-1_44. (Cited on pages 7, 15, 18, 21, and 27).
- [95] A. Kavousian, R. Rajagopal, and M. Fischer. Determinants of residential electricity consumption: Using smart meter data to examine the effect of climate, building characteristics, appliance stock, and occupants' behavior. *Energy*, 55:184–194, 2013. ISSN 0360-5442. doi: 10.1016/j.energy.2013.03.086. (Cited on pages 160 and 175).
- [96] D. A. Kenny and C. M. Judd. Consequences of violating the independence assumption in analysis of variance. *Psychological Bulletin*, 99(3):422–431, 1986. doi: 10.1037/0033-2909.99.3.422. (Cited on page 179).
- [97] M. Kesler, M. C. Kisacikoglu, and L. M. Tolbert. Vehicle-to-grid reactive power operation using plug-in electric vehicle bidirectional offboard charger. *IEEE Transactions on Industrial Electronics*, 61(12):6778–6784, 2014. ISSN 0278-0046 VO - 61. doi: 10.1109/TIE.2014.2314065. (Cited on pages 184 and 186).
- [98] S. Khakurel, C. Leung, and T. Le-Ngoc. A generalized water-filling algorithm with linear complexity and finite convergence time. *IEEE Wireless Communications Letters*, 3(2):225–228, 2014. ISSN 2162-2345 VO - 3. doi: 10.1109/WCL.2014.020314.130839. (Cited on page 32).
- [99] K. C. Kiwiel. On Floyd and Rivest's SELECT algorithm. *Theoretical Computer Science*, 347(1):214–238, 2005. ISSN 0304-3975. doi: 10.1016/j.tcs.2005.06.032. (Cited on pages 22, 62, and 66).
- [100] K. C. Kiwiel. Breakpoint searching algorithms for the continuous quadratic knapsack problem. *Mathematical Programming*, 112(2):473–491, 2007. ISSN 0025-5610. doi: 10.1007/s10107-006-0050-z. (Cited on pages 21, 23, 51, 55, 62, and 82).
- [101] K. C. Kiwiel. Variable fixing algorithms for the continuous quadratic knapsack problem. *Journal of Optimization Theory and Applications*, 136(3):445–458, mar 2008. ISSN 0022-3239. doi: 10.1007/s10957-007-9317-7. (Cited on pages 22 and 24).
- [102] D. E. Knuth. *The art of computer programming - Volume 3: Sorting and searching*. Addison-Wesley, Reading, MA, 2nd edition, 1998. ISBN 978-0-201-89685-5. (Cited on pages 63, 118, and 212).
- [103] T. R. Kuphaldt. Polyphase AC circuits. In *Lessons In Electric Circuits - Volume II - AC*, chapter 10. 6th edition, 2007. URL <https://www.ibiblio.org/kuphaldt/electricCircuits/AC/index.html>. (Cited on page 94).
- [104] M. Li and T. Liu. Distributed robust resource allocation with convex-concave uncertain objective functions. In *2018 57th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pages 368–373, 2018. ISBN 978-4-907764-60-9. doi: 10.23919/SICE.2018.8492684. (Cited on page 134).

- [105] C. Lin, K. L. Choy, G. T. S. Ho, S. H. Chung, and H. Y. Lam. Survey of green vehicle routing problem: Past and future trends. *Expert Systems with Applications*, 41(4, Part 1):1118 – 1138, 2014. ISSN 0957-4174. doi: 10.1016/j.eswa.2013.07.107. (Cited on page 4).
- [106] X. Ling, B. Wu, P. Ho, F. Luo, and L. Pan. Fast water-filling for agile power allocation in multi-channel wireless communications. *IEEE Communications Letters*, 16(8):1212–1215, 2012. ISSN 1558-2558 VO - 16. doi: 10.1109/L-COMM.2012.061912.120518. (Cited on page 32).
- [107] S. Liu. A review for submodular optimization on machine scheduling problems. In D.-Z. Du and J. Wang, editors, *Complexity and Approximation: In Memory of Ker-I Ko*, pages 252–267. Springer International Publishing, Cham, 2020. ISBN 978-3-030-41672-0. doi: 10.1007/978-3-030-41672-0_16. (Cited on page 17).
- [108] M. S. Lobo, M. Fazel, and S. Boyd. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, 152(1):341–365, 2007. ISSN 1572-9338. doi: 10.1007/s10479-006-0145-1. (Cited on page 23).
- [109] H. Lund, P. A. Østergaard, D. Connolly, I. Ridjan, B. V. Mathiesen, F. Hvelplund, J. Z. Thellufsen, and P. Sorknæs. Energy storage and smart energy systems. *International Journal of Sustainable Energy Planning and Management*, 11:3–14, 2016. doi: 10.5278/ijsepm.2016.11.2. (Cited on page 33).
- [110] W. Ma, J. Wang, V. Gupta, and C. Chen. Distributed energy management for networked microgrids using online ADMM with regret. *IEEE Transactions on Smart Grid*, 9(2):847–856, 2018. ISSN 1949-3053 VO - 9. doi: 10.1109/TSG.2016.2569604. (Cited on page 134).
- [111] M. Mahdavi, R. Jin, and T. Yang. Trading regret for efficiency: Online convex optimization with long term constraints. *Journal of Machine Learning Research*, 13:2503–2528, 2012. doi: 10.5555/2503308.2503322. (Cited on pages 137 and 139).
- [112] J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Convex and network flow optimization for structured sparsity. *Journal of Machine Learning Research*, 12(81):2681–2720, 2011. doi: 10.5555/1953048.2078191. (Cited on page 17).
- [113] J. Märkle-Huß, S. Feuerriegel, and D. Neumann. Contract durations in the electricity market: Causal impact of 15min trading on the EPEX SPOT market. *Energy Economics*, 69:367–378, 2018. ISSN 0140-9883. doi: 10.1016/j.eneco.2017.11.019. (Cited on pages 139 and 155).
- [114] N. Megiddo and A. Tamir. Linear time algorithms for some separable quadratic programming problems. *Operations Research Letters*, 13(4):203–211, 1993. ISSN 0167-6377. doi: 10.1016/0167-6377(93)90041-E. (Cited on pages 23 and 116).
- [115] X. Meng. Scalable simple random sampling and stratified sampling. In S. Dasgupta and D. McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML)*, volume 28, pages 531–539, Atlanta, GA, 2013. doi: 10.5555/3042817.3042996. (Cited on page 35).

- [116] L. Meussen. *Electrical vehicle charging under two types of steering signals with a lower bound*. M.Sc. thesis, Radboud University Nijmegen, 2015. (Cited on page 197).
- [117] S. Michel, N. Perrot, and F. Vanderbeck. Knapsack problems with setups. *European Journal of Operational Research*, 196(3):909–918, 2009. ISSN 0377-2217. doi: <https://doi.org/10.1016/j.ejor.2008.05.001>. (Cited on page 196).
- [118] S. Misra, L. Roald, and Y. Ng. Learning for constrained optimization: Identifying optimal active constraint sets. 2019. URL <https://arxiv.org/abs/1802.09639v4>. (Cited on page 137).
- [119] S. Moriguchi and A. Shioura. On Hochbaum’s proximity-scaling algorithm for the general resource allocation problem. *Mathematics of Operations Research*, 29(2):394–397, 2004. ISSN 0364-765X. doi: 10.1287/moor.1030.0076. (Cited on page 27).
- [120] S. Moriguchi, A. Shioura, and N. Tsuchimura. M-convex function minimization by continuous relaxation approach: proximity theorem and algorithm. *SIAM Journal on Optimization*, 21(3):633–668, 2011. ISSN 1052-6234. doi: 10.1137/080736156. (Cited on pages 26, 27, 94, 97, 116, and 125).
- [121] MOSEK-ApS. MOSEK Optimizer API for Python. Release 9.1.13, 2019. (Cited on page 117).
- [122] Y. Mou, H. Xing, Z. Lin, and M. Fu. Decentralized optimal demand-side management for PHEV charging in a smart grid. *IEEE Transactions on Smart Grid*, 6(2):726–736, 2015. ISSN 1949-3053. doi: 10.1109/TSG.2014.2363096. (Cited on pages 8, 94, and 138).
- [123] J. Mula, R. Poler, J. P. García-Sabater, and F. C. Lario. Models for production planning under uncertainty: A review. *International Journal of Production Economics*, 103(1):271–285, 2006. ISSN 0925-5273. doi: 10.1016/j.ijpe.2005.09.001. (Cited on page 134).
- [124] M. O. Müller, A. Stämpfli, U. Dold, and T. Hammer. Energy autarky: A conceptual framework for sustainable regional development. *Energy Policy*, 39(10):5800–5810, 2011. ISSN 0301-4215. doi: 10.1016/j.enpol.2011.04.019. (Cited on page 34).
- [125] M. Müller-Hannemann and S. Schirra, editors. *Algorithm engineering: Bridging the gap between algorithm theory and practice*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1 edition, 2010. ISBN 978-3-642-14866-8. doi: 10.1007/978-3-642-14866-8_1. (Cited on pages 6 and 62).
- [126] J. Munkhammar, P. Grahm, and J. Widén. Quantifying self-consumption of on-site photovoltaic power generation in households with electric vehicle home charging. *Solar Energy*, 97:208–216, 2013. ISSN 0038-092X. doi: 10.1016/j.solener.2013.08.015. (Cited on page 34).
- [127] K. Nagano and K. Aihara. Equivalence of convex minimization problems over base polytopes. *Japan Journal of Industrial and Applied Mathematics*, 29(3):519–534, 2012. ISSN 1868-937X. doi: 10.1007/s13160-012-0083-z. (Cited on pages 27, 28, 31, and 38).

- [128] K. Nagano and Y. Kawahara. Structured convex optimization under submodular constraints. In *Proceedings of the Twenty-Ninth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-13)*, pages 459–468, Corvallis, OR, 2013. doi: 10.5555/3023638.3023685. (Cited on page 28).
- [129] Y. Nesterov. Nonsmooth Convex Optimization. In Y. Nesterov, editor, *Lectures on Convex Optimization*, pages 139–240. Springer International Publishing, Cham, 1st edition, 2018. ISBN 978-3-319-91578-4. doi: 10.1007/978-3-319-91578-4_3. (Cited on page 193).
- [130] J. Neyman. On the two different aspects of the representative method: The method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society*, 97(4):558–606, 1934. ISSN 0952-8385. doi: 10.1111/j.2397-2335.1934.tb04184.x. (Cited on pages 17 and 35).
- [131] I. Norstad, K. Fagerholt, and G. Laporte. Tramp ship routing and scheduling with speed optimization. *Transportation Research Part C: Emerging Technologies*, 19(5):853–865, 2011. ISSN 0968-090X. doi: 10.1016/j.trc.2010.05.001. (Cited on pages 28, 29, 36, and 63).
- [132] J. G. J. Olivier and J. A. H. W. Peters. Trends in global CO₂ and total greenhouse gas emissions: 2019 report, 2020. URL <https://www.pbl.nl/en/trends-in-global-co2-emissions>. Accessed on November 15, 2020. (Cited on page 1).
- [133] N. Oreskes. The scientific consensus on climate change: How do we know we’re not wrong? In E. A. Lloyd and E. Winsberg, editors, *Climate Modelling: Philosophical and Conceptual Issues*, pages 31–64. Springer International Publishing, Cham, 2018. ISBN 978-3-319-65058-6. doi: 10.1007/978-3-319-65058-6_2. (Cited on page 1).
- [134] J. B. Orlin and B. Vaidyanathan. Fast algorithms for convex cost flow problems on circles, lines, and trees. *Networks*, 62(4):288–296, 2013. ISSN 0028-3045. doi: 10.1002/net.21517. (Cited on page 26).
- [135] N. P. Padhy. Unit commitment-a bibliographical survey. *IEEE Transactions on Power Systems*, 19(2):1196–1205, 2004. ISSN 1558-0679. doi: 10.1109/TPWRS.2003.821611. (Cited on page 4).
- [136] D. P. Palomar and J. R. Fonollosa. Practical algorithms for a family of waterfilling solutions. *IEEE Transactions on Signal Processing*, 53(2):686–695, 2005. ISSN 1941-0476 VO - 53. doi: 10.1109/TSP.2004.840816. (Cited on pages 17 and 32).
- [137] M. Patriksson. A survey on the continuous nonlinear resource allocation problem. *European Journal of Operational Research*, 185(1):1–46, 2008. ISSN 0377-2217. doi: 10.1016/j.ejor.2006.12.006. (Cited on pages 7, 15, 17, and 21).
- [138] M. Patriksson and C. Strömberg. Algorithms for the continuous nonlinear resource allocation problem - new implementations and numerical studies. *European Journal of Operational Research*, 243(3):703–722, 2015. ISSN 0377-2217. doi: 10.1016/j.ejor.2015.01.029. (Cited on pages 7, 17, 21, and 55).

- [139] M. M. A. Patwary, J. Blair, and F. Manne. Experiments on union-find algorithms for the disjoint-set data structure. In P. Festa, editor, *International Symposium on Experimental Algorithms (SEA)*, pages 411–423, Naples, 2010. ISBN 978-3-642-13193-6. doi: 10.1007/978-3-642-13193-6_35. (Cited on page 81).
- [140] G. Pearson and A. T. Young, editors. *Technically speaking: Why all Americans need to know more about technology*. National Academy Press, Washington D.C., 1st edition. ISBN 0-309-08262-5. (Cited on page 3).
- [141] H. N. Psaraftis and C. A. Kontovas. Ship speed optimization: Concepts, models and combined speed-routing scenarios. *Transportation Research Part C: Emerging Technologies*, 44:52–69, 2014. ISSN 0968-090X. doi: 10.1016/j.trc.2014.03.001. (Cited on page 35).
- [142] V. M. J. J. Reijnders, M. E. T. Gerards, J. L. Hurink, and G. J. M. Smit. Testing grid-based electricity prices and batteries in a field test. In *CIREN 2018 Workshop Proceedings*, Ljubljana, 2018. doi: 10.34890/219. (Cited on pages 82 and 155).
- [143] V. M. J. J. Reijnders, M. E. T. Gerards, and J. L. Hurink. A hybrid pricing mechanism for joint system optimization and social acceptance. In *2020 6th IEEE International Energy Conference (ENERGYCon)*, pages 900–905, Tunis, 2020. ISBN 978-1-7281-2956-3. doi: 10.1109/ENERGYCon48941.2020.9236603. (Cited on page 35).
- [144] B. P. Roberts and C. Sandberg. The role of energy storage in development of smart grids. *Proceedings of the IEEE*, 99(6):1139–1144, 2011. ISSN 1558-2256 VO - 99. doi: 10.1109/JPROC.2011.2116752. (Cited on page 33).
- [145] S. M. Robinson. Strongly regular generalized equations. *Mathematics of Operations Research*, 5(1):43–62, 1980. ISSN 0364-765X. doi: 10.1287/moor.5.1.43. (Cited on page 153).
- [146] L. Sanathanan. On an allocation problem with multistage constraints. *Operations Research*, 19(7):1647–1663, 1971. ISSN 0030-364X. doi: 10.1287/opre.19.7.1647. (Cited on page 23).
- [147] M. H. H. Schoot Uiterkamp. *Robust planning of electric vehicle charging*. M.Sc. thesis, University of Twente, 2016. (Cited on pages 95 and 176).
- [148] R. Serfling. Quantile functions for multivariate analysis: approaches and applications. *Statistica Neerlandica*, 56(2):214–232, 2002. ISSN 0039-0402. doi: 10.1111/1467-9574.00195. (Cited on pages 168 and 179).
- [149] S. Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012. ISSN 1935-8237. doi: 10.1561/22000000018. (Cited on page 137).
- [150] F. Shams, G. Bacci, and M. Luise. A survey on resource allocation techniques in OFDM(A) networks. *Computer Networks*, 65:129–150, 2014. ISSN 1389-1286. doi: 10.1016/j.comnet.2014.03.017. (Cited on page 32).

- [151] A. Shapiro and A. Nemirovski. On complexity of stochastic programming problems. In V. Jeyakumar and A. Rubinov, editors, *Continuous optimization: Current trends and modern applications*, pages 111–146. Springer US, Boston, 2005. ISBN 978-0-387-26771-5. doi: 10.1007/0-387-26771-9_4. (Cited on page 136).
- [152] A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lecture Notes on Stochastic Programming: Modeling and Theory*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1st edition, 2009. (Cited on page 136).
- [153] A. Shioura, N. V. Shakhlevich, and V. A. Strusevich. Machine speed scaling by adapting methods for convex optimization with submodular constraints. *INFORMS Journal on Computing*, 29(4):724–736, 2017. ISSN 1091-9856. doi: 10.1287/ijoc.2017.0758. (Cited on pages 28 and 38).
- [154] A. Shioura, N. V. Shakhlevich, and V. A. Strusevich. Preemptive models of scheduling with controllable processing times and of scheduling with imprecise computation: A review of solution approaches. *European Journal of Operational Research*, 266(3):795–818, 2018. ISSN 0377-2217. doi: 10.1016/j.ejor.2017.08.034. (Cited on page 17).
- [155] I. Shvartsman. On stability of minimizers in convex programming. *Nonlinear Analysis: Theory, Methods & Applications*, 75(3):1563–1571, 2012. ISSN 0362-546X. doi: 10.1016/j.na.2011.03.033. (Cited on page 153).
- [156] J. Slager. *Nonlinear convex optimisation problems in the smart grid*. B.Sc. thesis, University of Twente, 2019. (Cited on pages 25 and 89).
- [157] Z. Song, X. Zhang, and C. Eriksson. Data center energy and cost saving evaluation. *Energy Procedia*, 75:1255 – 1260, 2015. doi: 10.1016/j.egypro.2015.07.178. (Cited on page 5).
- [158] E. Spedicato. A bound to the condition number of canonical rank-two corrections and applications to the variable metric method. *Calcolo*, 12(2):185–199, 1975. ISSN 1126-5434. doi: 10.1007/BF02576015. (Cited on page 98).
- [159] M. Spivak. *Calculus on manifolds: A modern approach to classical theorems of advanced calculus*. Addison-Wesley Publishing Company, Reading, MA, 1st edition, 1965. ISBN 0-8053-9021-9. (Cited on page 151).
- [160] W. D. Stevenson. *Elements of Power System Analysis*. McGraw Hill, New York, NY, 3rd edition, 1975. ISBN 9780070857568. (Cited on page 94).
- [161] G. Still. Lecture notes on parametric optimization: An introduction. 2018. URL http://www.optimization-online.org/DB_FILE/2018/04/6587.pdf. (Cited on page 147).
- [162] X. Sun, X. Zheng, and D. Li. Recent advances in mathematical programming with semi-continuous variables and cardinality constraint. *Journal of the Operations Research Society of China*, 1(1):55–77, 2013. ISSN 2194-6698. doi: 10.1007/s40305-013-0004-0. (Cited on page 196).

- [163] A. Tamir. A strongly polynomial algorithm for minimum convex separable quadratic cost flow problems on two-terminal series—parallel networks. *Mathematical Programming*, 59(1):117–132, 1993. ISSN 1436-4646. doi: 10.1007/BF01581240. (Cited on pages 26, 94, 97, and 125).
- [164] H. A. Toersche. *Effective and efficient coordination of flexibility in smart grids*. PhD thesis, University of Twente, oct 2016. (Cited on page 4).
- [165] M. Uddin, M. F. Romlie, M. F. Abdullah, S. Abd Halim, A. H. Abu Bakar, and T. Chia Kwang. A review on peak load shaving strategies. *Renewable and Sustainable Energy Reviews*, 82:3323–3332, 2018. ISSN 1364-0321. doi: 10.1016/j.rser.2017.10.056. (Cited on page 34).
- [166] T. van der Klauw. *Decentralized energy management with profile steering: Resource allocation problems in energy management*. PhD thesis, University of Twente, 2017. (Cited on page 4).
- [167] T. van der Klauw, M. E. T. Gerards, and J. L. Hurink. Resource allocation problems in decentralized energy management. *OR Spectrum*, 39(3):749–773, 2017. ISSN 0171-6468. doi: 10.1016/0167-6377(84)90010-5. (Cited on pages 17, 24, 33, 62, 81, 89, 134, 186, and 187).
- [168] D. Vanhoudt, D. Geysen, B. Claessens, F. Leemans, L. Jespers, and J. Van Bael. An actively controlled residential heat pump: Potential on peak shaving and maximization of self-consumption of renewable energy. *Renewable Energy*, 63:531–543, 2014. ISSN 0960-1481. doi: 10.1016/j.renene.2013.10.021. (Cited on page 35).
- [169] L. Végh. A strongly polynomial algorithm for a class of minimum-cost flow problems with separable convex objectives. *SIAM Journal on Computing*, 45(5):1729–1761, 2016. ISSN 0097-5397. doi: 10.1137/140978296. (Cited on page 94).
- [170] A. F. Veinott. Least d-majorized network flows with inventory and statistical applications. *Management Science*, 17(9):547–567, 1971. ISSN 0025-1909. doi: 10.1287/mnsc.17.9.547. (Cited on page 17).
- [171] T. Vidal, P. Jaillet, and N. Maculan. A decomposition algorithm for nested resource allocation problems. *SIAM Journal on Optimization*, 26(2):1322–1340, 2016. ISSN 1052-6234. doi: 10.1137/140965119. (Cited on pages 24 and 25).
- [172] T. Vidal, D. Gribel, and P. Jaillet. Separable convex optimization with nested lower and upper constraints. *INFORMS Journal on Optimization*, 1(1):71–90, 2019. ISSN 2575-1484. doi: 10.1287/ijoo.2018.0004. (Cited on pages 17, 24, 25, 62, 63, 66, 67, 81, 82, and 125).
- [173] G. Wachsmuth. On LICQ and the uniqueness of Lagrange multipliers. *Operations Research Letters*, 41(1):78–80, 2013. ISSN 0167-6377. doi: 10.1016/j.orl.2012.11.009. (Cited on page 141).
- [174] S. Weekx and J. Driesen. Load balancing with EV chargers and PV inverters in unbalanced distribution grids. *IEEE Transactions on Sustainable Energy*, 6(2):635–643, 2015. ISSN 1949-3029. doi: 10.1109/TSTE.2015.2402834. (Cited on pages 89, 92, 95, 117, and 211).

- [175] P. Wolfe. Finding the nearest point in a polytope. *Mathematical Programming*, 11(1):128–149, 1976. ISSN 1436-4646. doi: 10.1007/BF01580381. (Cited on page 27).
- [176] S. E. Wright and S. Lim. Solving nested-constraint resource allocation problems with an interior point method. *Operations Research Letters*, 48(3):297–303, 2020. ISSN 0167-6377. doi: 10.1016/j.orl.2020.04.001. (Cited on pages 25 and 89).
- [177] S. E. Wright and J. J. Rohal. Solving the continuous nonlinear resource allocation problem with an interior point method. *Operations Research Letters*, 42(6):404–408, 2014. doi: 10.1016/j.orl.2014.07.001. (Cited on pages 22 and 25).
- [178] Z. Wu. *Fast exact algorithms for optimization problems in resource allocation and switched linear systems*. PhD thesis, University of Minesota, 2019. (Cited on page 24).
- [179] C. Xing, Y. Jing, S. Wang, S. Ma, and H. V. Poor. New viewpoint and algorithms for water-filling solutions in wireless communications. *IEEE Transactions on Signal Processing*, 68:1618–1634, 2020. ISSN 1941-0476 VO - 68. doi: 10.1109/TSP.2020.2973488. (Cited on pages 17 and 32).
- [180] I. Yanıkoğlu, B. L. Gorissen, and D. den Hertog. A survey of adjustable robust optimization. *European Journal of Operational Research*, 277(3):799–813, 2019. ISSN 0377-2217. doi: 10.1016/j.ejor.2018.08.031. (Cited on page 136).
- [181] M. Yilmaz and P. T. Krein. Review of battery charger topologies, charging power levels, and infrastructure for plug-in electric and hybrid vehicles. *IEEE Transactions on Power Electronics*, 28(5):2151–2169, 2013. ISSN 0885-8993. doi: 10.1109/TPEL.2012.2212917. (Cited on page 192).
- [182] K. Young, C. Wang, L. Y. Wang, and K. Strunz. Electric vehicle battery technologies. In R. Garcia-Valle and J. A. Peças Lopes, editors, *Electric vehicle integration into modern power networks*, chapter 2, pages 15–56. Springer, New York, 2013. ISBN 978-1-4614-0134-6. doi: 10.1007/978-1-4614-0134-6_2. (Cited on pages 184, 196, and 211).
- [183] K. K. Zame, C. A. Brehm, A. T. Nitica, C. L. Richard, and G. D. Schweitzer III. Smart grid and energy storage: Policy recommendations. *Renewable and Sustainable Energy Reviews*, 82:1646–1654, 2018. ISSN 1364-0321. doi: 10.1016/j.rser.2017.07.011. (Cited on page 33).
- [184] J. Zhang, F. R. Yu, S. Wang, T. Huang, Z. Liu, and Y. Liu. Load balancing in data center networks: A survey. *IEEE Communications Surveys Tutorials*, 20(3):2324–2352, 2018. doi: 10.1109/COMST.2018.2816042. (Cited on page 5).
- [185] S. Zhuravlev, J. C. Saez, S. Blagodurov, A. Fedorova, and M. Prieto. Survey of energy-cognizant scheduling techniques. *IEEE Transactions on Parallel and Distributed Systems*, 24(7):1447–1464, 2013. ISSN 1558-2183 VO - 24. doi: 10.1109/T-PDS.2012.20. (Cited on page 37).

LIST OF PUBLICATIONS

253

- [SU:1] M. H. H. Schoot Uiterkamp, M. E. T. Gerards, and J. L. Hurink. On a reduction for a class of resource allocation problems. 2020. URL <https://arxiv.org/abs/2008.11829>.
- [SU:2] M. H. H. Schoot Uiterkamp, J. L. Hurink, and M. E. T. Gerards. A fast algorithm for quadratic resource allocation problems with nested constraints. 2020. URL <https://arxiv.org/abs/2009.03880>.
- [SU:3] M. H. H. Schoot Uiterkamp, M. E. T. Gerards, and J. L. Hurink. ODDO: Online duality-driven optimization. 2020. URL <https://arxiv.org/abs/2008.09838>.
- [SU:4] M. H. H. Schoot Uiterkamp, M. E. T. Gerards, and J. L. Hurink. Quadratic nonseparable resource allocation problems with generalized bound constraints. 2020. URL <https://arxiv.org/abs/2007.06280>.
- [SU:5] M. H. H. Schoot Uiterkamp, M. E. T. Gerards, and J. L. Hurink. Online electric vehicle charging with discrete charging rates. *Sustainable Energy, Grids and Networks*, 25:100423, 2021. ISSN 2352-4677. doi: 10.1016/j.segan.2020.100423.
- [SU:6] J. L. Hurink, S. Klootwijk, B. Manthey, V. M. J. J. Reijnders, and M. H. H. Schoot Uiterkamp, editors. *Proceedings of the 17th Cologne-Twente Workshop on Graphs and Combinatorial Optimization*, number WP 19-01 in DSI Workshop Proceedings Series, 2019. University of Twente.
- [SU:7] M. H. H. Schoot Uiterkamp, T. van der Klauw, M. E. T. Gerards, and J. L. Hurink. Offline and online scheduling of electric vehicle charging with a minimum charging threshold. In *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids*, Aalborg, Denmark, 2018. doi: 10.1109/SmartGridComm.2018.8587556.
- [SU:8] M. H. H. Schoot Uiterkamp, M. E. T. Gerards, and J. L. Hurink. Fill-level prediction in online valley-filling algorithms for electric vehicle charging. In *2018 IEEE PES Innovative Smart Grid Technologies Europe*, Sarajevo, Bosnia-Herzegovina, 2018. doi: 10.1109/ISGTEurope.2018.8571605.
- [SU:9] M. H. H. Schoot Uiterkamp, G. Hoogsteen, M. E. T. Gerards, J. L. Hurink, and G. J. M. Smit. Multi-commodity support in profile steering. In *2017 IEEE PES Innovative Smart Grid Technologies Europe*, Turin, Italy, 2017. doi: 10.1109/ISGTEurope.2017.8260190.

Martijn H. H. Schoot Uiterkamp is the main author of all publications in this list except [SU:6], for which the main editorship is shared with all editors, and [SU:9], for which the main authorship is shared with G. Hoogsteen.

THIS THESIS

```
@phdthesis{schoot uiterkamp2021:thesis,
  author={Schoot Uiterkamp, Martijn H. H.},
  title={Duality-driven optimization in energy management -- Offline and
         online algorithms for resource allocation problems},
  school={University of Twente},
  address={PO Box 217, 7500 AE Enschede, The Netherlands},
  year={2021},
  month=apr,
  day={23},
  number={DSI Ph.D. Thesis Series No. 21-005},
  issn={2589-7721},
  isbn={978-90-365-5161-8},
  doi={10.3990/1.9789036551618}
}
```

About the author

Martijn H. H. Schoot Uiterkamp received his B.Sc. and M.Sc. degrees in Applied Mathematics from the University of Twente, the Netherlands, in 2014 and 2016 respectively. From 2016 until 2020 he was a PhD student at the same university in the groups Computer Architecture for Embedded Systems (CAES) and Discrete Mathematics and Mathematical Programming (DMMP). His Ph.D. research focused on the development of efficient and fast algorithms for optimization problems arising in energy management.

His research interests are the theory and application of mathematical optimization and the interaction between these two. In particular, he likes to work on the design of practically usable algorithms for specific optimization problems and on applications that contribute to the Sustainable Development Goals.

ISBN 978-90-365-5161-8

9 789036 551618

PROPOSITIONS

accompanying the Ph.D. thesis

Duality-driven optimization in energy management *Offline and online algorithms for resource allocation problems*

by Martijn H. H. Schoot Uiterkamp,
to be defended on Friday April 23, 2021

- 1 — The broad concept of duality, in the sense that problems are viewed from different perspectives and disciplines, is crucial for the success of energy management research.
(This thesis)
- 2 — The resource allocation problem is a suitable and successful model for many optimization problems in engineering due to its simplicity and intuitive application-specific interpretation.
(This thesis)
- 3 — When developing algorithms, it must not only be specified which problem the algorithm solves but also which criteria a good solution must satisfy.
(This thesis)
- 4 — In the development of smart ICT systems, data-driven optimization techniques that can successfully handle data uncertainty will become more and more important and necessary.
(This thesis)
- 5 — The shared aim to work together and be open to different perspectives is more crucial for the success of multi-disciplinary research than the expertise level of the individual researchers.
- 6 — Publishing on why an experiment did not yield the expected result may be more valuable to the scientific community than publishing only on successful experiments.
- 7 — A friendly and supportive research environment at the home university may stimulate the success and well-being of a researcher better than a large (international) network.
- 8 — Good scientific research requires common sense, but good common sense does not require scientific research.
- 9 — Science and art have in common that both explore, deepen, and extend the boundaries of human knowledge and perception.
- 10 — An increase in proportional and diverse representation in all layers of government is the key to solving many of today's societal problems.

STELLINGEN

behorende bij het proefschrift

Duality-driven optimization in energy management *Offline and online algorithms for resource allocation problems*

door Martijn H. H. Schoot Uiterkamp,
te verdedigen op vrijdag 23 april 2021

- 1 — Het brede concept van dualiteit, in de zin dat problemen vanuit verschillende perspectieven en disciplines bekeken worden, is cruciaal voor het succes van energiemanageronderzoek. *(Dit proefschrift)*
- 2 — Het allocatieprobleem is een geschikt en succesvol model voor veel technische optimalisatieproblemen vanwege zijn eenvoud en intuïtieve, toepassingsspecifieke interpretatie. *(Dit proefschrift)*
- 3 — Bij het ontwikkelen van algoritmes moet niet alleen gespecificeerd worden welk probleem het algoritme oplost, maar ook aan welke voorwaarden een goede oplossing moet voldoen. *(Dit proefschrift)*
- 4 — Bij de ontwikkeling van slimme ICT-systemen zullen data-gedreven optimalisatietechnieken die goed met data-onzekerheid om kunnen gaan alleen maar belangrijker en noodzakelijker worden. *(Dit proefschrift)*
- 5 — De gedeelde wil om samen te werken en open te staan voor andere perspectieven is cruciaal voor het succes van multidisciplinair onderzoek dan het deskundigheidsniveau van de individuele onderzoekers.
- 6 — Publiceren over waarom een experiment niet het verwachte resultaat opleverde kan waardevoller zijn voor de wetenschappelijke gemeenschap dan alleen over de succesvolle experimenten publiceren.
- 7 — Een vriendelijke en ondersteunende onderzoeksomgeving bij de eigen universiteit kan het succes en welzijn van een onderzoeker beter stimuleren dan een groot (internationaal) netwerk.
- 8 — Goed wetenschappelijk onderzoek vereist gezond verstand, maar goed gezond verstand vereist geen wetenschappelijk onderzoek.
- 9 — Wetenschap en kunst hebben gemeen dat ze beiden de grenzen van de menselijke kennis en waarneming verkennen, verdiepen, en verleggen.
- 10 — Een toename van evenredige en diverse vertegenwoordiging in alle lagen van de overheid is de sleutel tot het oplossen van veel huidige maatschappelijke problemen.