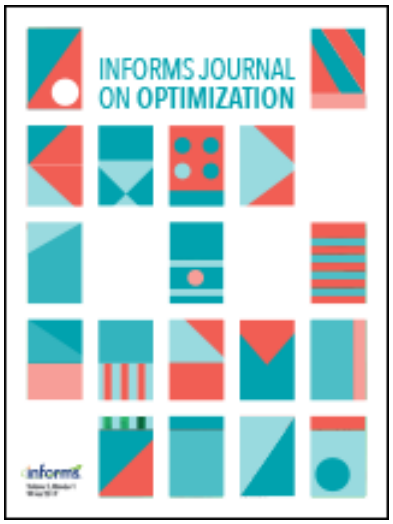


INFORMS Journal on Optimization

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

A Class of Convex Quadratic Nonseparable Resource Allocation Problems with Generalized Bound Constraints

Martijn H. H. Schoot Uiterkamp, Marco E. T. Gerards, Johann L. Hurink

To cite this article:
Martijn H. H. Schoot Uiterkamp, Marco E. T. Gerards, Johann L. Hurink (2022) A Class of Convex Quadratic Nonseparable Resource Allocation Problems with Generalized Bound Constraints. INFORMS Journal on Optimization 4(2):215-247. https://doi.org/10.1287/ijoo.2021.0065

http://pubsonline.informs.org/journal/ijoo

INFORMS JOURNAL ON OPTIMIZATION
Vol. 4, No. 2, Spring 2022, pp. 215–247
ISSN 2575-1484 (print), ISSN 2575-1492 (online)

A Class of Convex Quadratic Nonseparable Resource Allocation Problems with Generalized Bound Constraints

Martijn H. H. Schoot Uiterkamp,[a,*] **Marco E. T. Gerards,**[b] **Johann L. Hurink**[b]

[a] Department of Mathematics and Computer Science, Eindhoven University of Technology, 5612 AZ Eindhoven, Netherlands; [b] Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, 7522 NB Enschede, Netherlands
*Corresponding author

Contact: m.h.h.schoot.uiterkamp@tue.nl, https://orcid.org/0000-0002-8125-0479 (MHHSU); e.t.gerards@utwente.nl, https://orcid.org/0000-0002-3870-4826 (METG); j.l.hurink@utwente.nl, https://orcid.org/0000-0001-6986-5633 (JLH)

Received: July 13, 2020
Revised: February 26, 2021; June 16, 2021
Accepted: July 17, 2021
Published Online in Articles in Advance: February 28, 2022

https://doi.org/10.1287/ijoo.2021.0065

Abstract. We study a convex quadratic nonseparable resource allocation problem that arises in the area of decentralized energy management (DEM), where unbalance in electricity networks has to be minimized. In this problem, the given resource is allocated over a set of activities that is divided into subsets, and a cost is assigned to the overall allocated amount of resources to activities within the same subset. We derive two efficient algorithms with $O(n \log n)$ worst-case time complexity 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. Both algorithms are inspired by well-studied breakpoint search methods for separable convex resource allocation problems. Numerical evaluations on both real and synthetic data confirm the theoretical efficiency of both algorithms and demonstrate their suitability for integration in DEM systems.

Funding: This work was supported by the Nederlandse Organisatie voor Wetenschappelijk Onderzoek [Grant 647.002.003] within the SIMPS project and also supported by Eneco.

Keywords: resource allocation • energy management • nonseparable optimization

1. Introduction

1.1. Problem Formulation and Applications

Resource allocation problems belong to the fundamental problems in the operations research literature (Patriksson 2008). These problems involve the allocation of a given resource (e.g., money or energy) over a set of activities (e.g., projects or time slots) while minimizing a given cost function or maximizing a given utility function. In its simplest form, the problem can be formulated mathematically as follows:

$$\begin{aligned} \text{RAP}: \min_{x \in \mathbb{R}^n} \quad & \sum_{i=1}^{n} f_i(x_i), \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i = R, \\ & l_i \le x_i \le u_i, \quad i \in \{1, \dots, n\}. \end{aligned}$$

Here, each variable x_i represents the amount of the total resource $R \in \mathbb{R}$ that is allocated to activity i and the values $l_i, u_i \in \mathbb{R}$ are lower and upper bounds on the amount allocated to activity i. Moreover, each function $f_i : \mathbb{R} \to \mathbb{R}$ assigns a cost to allocating resource to activity i.

In this article, we study the following more specific convex resource allocation problem, which is an extension of the convex quadratic resource allocation problem:

$$\text{QRAP-NonSep-GBC}: \min_{x \in \mathbb{R}^n} \quad \sum_{j=1}^{m} \frac{1}{2} w_j \left(\sum_{i \in \mathcal{N}_j} x_i \right)^2 + \sum_{i=1}^{n} \left(\frac{1}{2} a_i x_i^2 + b_i x_i \right), \tag{1a}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_i = R, \tag{1b}$$

$$L_j \le \sum_{i \in \mathcal{N}_j} x_i \le U_j, \quad j \in \{1, \dots, m\}, \tag{1c}$$

$$l_i \le x_i \le u_i, \quad i \in \{1, \dots, n\}, \tag{1d}$$

where $\boldsymbol{w}, \boldsymbol{b}, \boldsymbol{l}, \boldsymbol{u} \in \mathbb{R}^n$, $\boldsymbol{a} \in \mathbb{R}^n_{>0}$, $R \in \mathbb{R}$, and $\boldsymbol{L}, \boldsymbol{U} \in \mathbb{R}^m$ 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 $n_1, \dots, n_m$ indexed by $\mathcal{M} := \{1, \dots, m\}$ is given. The objective function of Problem QRAP-NonSep-GBC assigns for each subset $\mathcal{N}_j$ a cost to the sum of all allocated amounts associated with this subset and to the individual amounts. Similarly, Constraints (1c) and (1d) put bounds on the sum of all variables associated with each given subset and on the individual variables.

Our primary interest in studying this problem stems from its application in decentralized energy management (DEM) (Siano 2014, Esther and Kumar 2016). The aim of DEM is to optimize the simultaneous energy consumption of multiple devices within a neighborhood. Compared with other energy management paradigms such as *centralized* energy management, within a DEM system, devices optimize their own consumption locally and the control system coordinates the local optimization of these devices to optimize certain neighborhood objectives.

In particular, 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 arbitrarily over all the three phases of the low-voltage network. Recent studies show that three-phase EV charging, as opposed to single-phase EV charging, can 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 (Weckx and Driesen 2015, Schoot Uiterkamp et al. 2017). We discuss this issue in more detail in Section 2, and we show that the three-phase EV charging problem can be modeled as an instance of Problem QRAP-NonSep-GBC.

An important aspect of the DEM paradigm is that device-level problems, such as the aforementioned three-phase EV charging problem, are solved locally. This means that the corresponding device-level optimization algorithms are executed on embedded systems located within, for example, households or the charging equipment. It is important that these algorithms are very efficient with regard to both execution time and storage space, because often they are called multiple times within the DEM system and the embedded systems on which the algorithms run have limited computational power and memory (Beaudin and Zareipour 2015). Therefore, efficient and tailored device-level optimization algorithms are crucial ingredients for the real-life implementation of DEM systems. In particular, to solve the three-phase EV charging problem, an efficient algorithm to solve Problem QRAP-NonSep-GBC is required.

Another application of Problem QRAP-NonSep-GBC is in portfolio optimization, where the question is how to invest in a set of stocks to maximize the (expected) return of the investments. Within the popular paradigm of mean-variance portfolio optimization, introduced by Markowitz (1952), the goal is to find a maximum mean-variance portfolio, that is, a portfolio that maximizes the expected return minus the variance of the portfolio. To compute this portfolio, one requires the expected returns and the covariance matrix of the assets. Usually, the covariance matrix itself is not available and must be estimated based on, for example, a sample covariance matrix. However, such estimates are unstable when the sample size is small and the number of variables is large (Ledoit and Wolf 2003). One way to simplify the estimation procedure and obtain more stable estimators is to assume a certain low-rank structure underlying the covariance matrix and subsequently estimate the parameters of this structure. One popular example of such a structure is equicorrelation (Engle and Kelly 2012), which states that particular groups of assets have the same pairwise correlation, that is, each pair of assets within a given group has the same covariance. Given the equicorrelation ρ_j for a given set $\mathcal{N}_j$ of assets, the covariance matrix Σ^j of these assets is given by $(1-\rho_j)I + \rho_j \boldsymbol{e}\boldsymbol{e}^\top$, where I is the identity matrix and $\boldsymbol{e}$ is the vector of ones of appropriate size.

Under the assumption of equicorrelation, the mean-variance portfolio can be obtained as follows. Suppose an amount of R is to be invested into n stocks whose expected return is given by $\boldsymbol{r} \in \mathbb{R}^n$ and that are separated in $\mathcal{N}_j$ groups so that

- The correlation between two assets within a given group $\mathcal{N}_j$ is ρ_j; and
- The correlation between two assets not belonging to the same group is zero.

Then the mean-variance portfolio is the solution to the following optimization problem:

$$\begin{aligned} \max_{\boldsymbol{x} \in \mathbb{R}^n} \quad & \boldsymbol{r}^\top \boldsymbol{x} - \sum_{j=1}^{m} (\boldsymbol{x}^j)^\top \Sigma^j \boldsymbol{x}^j, \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i = R, \\ & x_i \geq 0, \quad i \in \{1, \dots, n\}, \end{aligned}$$

where $\boldsymbol{x}^j := (x_i)_{i\in\mathcal{N}_j}$ for $j \in \{1,\dots,m\}$. To formulate this problem as an instance of Problem QRAP-NonSep-GBC, observe that for a given $j \in \{1,\dots,m\}$, we have

$$(\boldsymbol{x}^j)^\top \Sigma^j \boldsymbol{x}^j = (\boldsymbol{x}^j)^\top (1-\rho_j) I \boldsymbol{x}^j + (x^j)^\top \rho_j \boldsymbol{e}\boldsymbol{e}^\top \boldsymbol{x}^j = \sum_{i\in\mathcal{N}_j} (1-\rho_j) x_i^2 + \rho_j \left(\sum_{i\in\mathcal{N}_j} x_i\right)^2.$$

Thus, by choosing the objective function parameters of Problem QRAP-NonSep-GBC as $w_j = 2\rho_j$, $a_i = 2(1-\rho_j)$, and $b_i = -r_i$ for $j \in \{1,\dots,m\}$ and $i \in \mathcal{N}_j$, we can formulate the mean-variance portfolio optimization problem with equicorrelation as an instance of Problem QRAP-NonSep-GBC.

Finally, the special case of Problem QRAP-NonSep-GBC where $w = 0$ has applications in, for example, power allocation in communication networks (He et al. 2013). In many telecommunication systems, the amount of noise that accompanies the transmission of data can be reduced by transmitting the data over several parallel channels (Shams et al. 2014). In some advanced systems, these channels are divided into subchannels that represent so-called second users (He et al. 2013). One goal in these systems is to allocate power to channels, each potentially divided into subchannels, so that the capacity of the subchannels is maximized and the power allocation limits of both the channels and (sub)channels are satisfied. We refer to Schoot Uiterkamp et al. (2022) for more details on this application and the formulation of this special case of Problem QRAP-NonSep-GBC.

1.2. Problem Classification and Related Work

Problem QRAP-NonSep-GBC can be classified as a quadratic *nonseparable* resource allocation problem with generalized bound constraints (Constraint (1c)). The nonseparability is because of 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. When the factors w_j are zero, these nonseparable terms disappear and Problem QRAP-NonSep-GBC becomes the convex quadratic *separable* resource allocation problem with generalized bound constraints. In the literature, this problem has hardly been studied: a special case that includes only generalized upper bound constraints is studied by Hochbaum and Hong (1995) and Bretthauer and Shetty (1997). However, generalized bound constraints are a special case of *submodular* constraints (as shown by Fujishige 1984). As a consequence, several algorithms for convex separable resource allocation problems with general submodular constraints can be specialized to generalized bound constraints (Katoh et al. 2013).

When in addition the generalized bound constraints are omitted, Problem QRAP-NonSep-GBC reduces to the convex quadratic *simple* separable resource allocation problem. This problem and its extension to convex cost functions have been well studied (Patriksson 2008, Patriksson and Strömberg 2015 and the references therein). Because our approach for solving the nonseparable resource allocation problem QRAP-NonSep-GBC relies on several techniques and properties also observed in approaches for solving convex separable resource allocation problems, we highlight three significantly different types of solution approaches for the latter problems.

1. Breakpoint Search Methods (Helgason et al. 1980, De Waegenaere and Wielhouwer 2012). These methods consider the Lagrangian dual of the original problem and exploit the structure of the Karush-Kuhn-Tucker (KKT) optimality conditions (Boyd and Vandenberghe 2004) to efficiently search for the optimal (dual) multiplier associated with Resource Constraint (1b). More precisely, they exploit the fact that the solution to the KKT conditions is a piece-wise monotone function of this multiplier. Subsequently, they determine the two breakpoints in between which the optimal multiplier must lie based on this monotonicity property. Finally, a simplified version of the KKT-conditions based on these two breakpoints is solved to find the optimal multiplier. Breakpoint search methods have been succesfully applied to both the convex simple resource allocation problem (see Patriksson 2008, Patriksson and Strömberg 2015, and the references therein) and several of its extensions such as the resource allocation problem with additional nested constraints (Vidal et al. 2019).

2. Variable-Fixing Methods (Bretthauer and Shetty 2002, Kiwiel 2008). In these methods, first a solution to the problem without Box Constraints (1d) is computed. Subsequently, the optimal value of several variables that exceed their bounds in this solution is determined based on whether the variable sum is smaller or larger than the resource value R. The variables whose value in the eventual optimal solution have been determined are removed from the problem and the amount of resource is adjusted accordingly. Thus, a new instance of the problem is obtained with at least one variable less than the original problem. This process continues until none of the variables in the solution to the relaxed problem exceeds its bounds. The worst-case time complexity of variable-fixing methods is in general worse than that of breakpoint search methods. However, the former methods sometimes show better practical performance, both for the convex simple resource allocation problem (Patriksson and Strömberg 2015)

and for several of its extensions such as the resource allocation problem with additional nested constraints (see, e.g., Schoot Uiterkamp et al. (2021) that contains a numerical comparison between the variable-fixing based algorithm of van der Klauw et al. (2017) and the monotonicity-based algorithm of Vidal et al. (2019)).

3. Interior-Point Methods (Cominetti et al. 2014, Wright and Rohal 2014). Interior-point methods are an important class of iterative methods for convex optimization (Gondzio 2012). In each iteration, the step from the current to the next iterate is determined as the solution to a perturbed version of the KKT optimality conditions. This boils down to solving a linear system involving the constraint matrix, which generally requires $O(n^3)$ time. However, for several resource allocation problems, the number of operations required to solve this system can be significantly reduced due to the special sparse structure of the constraint matrix. In particular, for the convex simple resource allocation problem and its extension with nested constraints, this number can be reduced to $O(n)$ (Wright and Rohal 2014, Wright and Lim 2020). It is shown that these methods can significantly outperform alternative approaches such as breakpoint search and pegging depending on the complexity of the objective function.

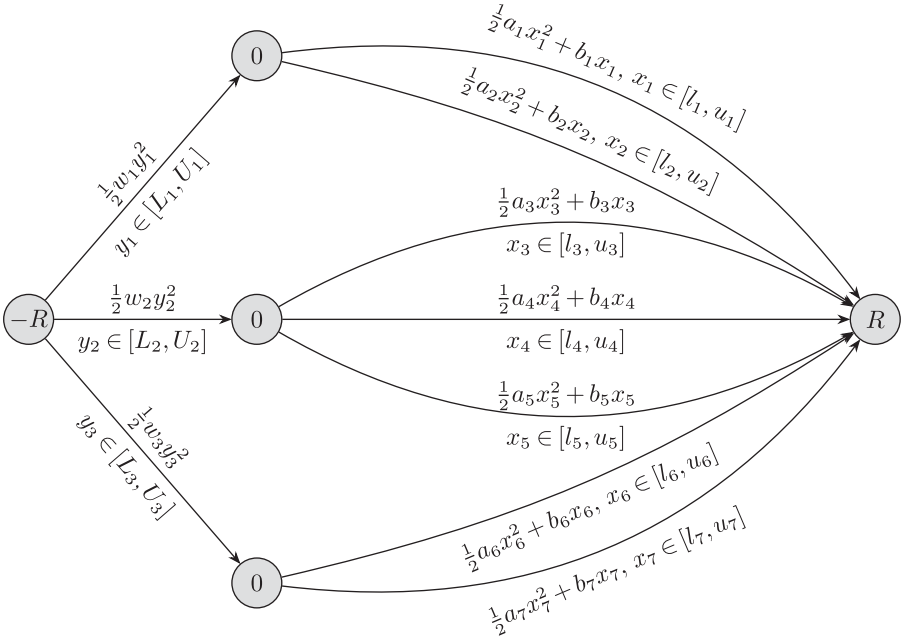
Without any additional restrictions on w, Problem QRAP-NonSep-GBC is NP-hard because the special case where $n_j = 1$, $w_j = -2$, and $a_i = 1$ for all $i \in \mathcal{N}$ and $j \in \mathcal{M}$ is a separable *concave* quadratic resource allocation problem that is known to be NP-hard (Sahni 1974). Therefore, we focus on instances where the parameters are chosen such that the objective function is convex (we come back to this later in this section and in Section 3.1).

Observe that Problem QRAP-NonSep-GBC can be modeled as a minimum convex quadratic cost flow problem if $w \geq 0$ (Figure 1). Therefore, this case can be solved in strongly polynomial time (Végh 2016). In fact, because its network structure is series-parallel, it can be solved by the algorithms from Tamir (1993) and Moriguchi et al. (2011) 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. In particular, this holds for the aforementioned EV scheduling problem in DEM, where the objective of minimizing load unbalance is modeled as an instance of Problem QRAP-NonSep-GBC by setting some or all of the factors w_j to a negative number (see also Section 2.2).

1.3. Contribution

In this article, we present two $O(n \log n)$ time algorithms for strictly convex instances of Problem QRAP-NonSep-GBC, thereby adding a new problem to the small class of quadratic programming problems that can be solved efficiently in strongly polynomial time. For this, we derive a property of problem instances that uniquely

Figure 1. Problem QRAP-NonSep-GBC Cast as a Minimum Quadratic Cost Flow Problem



Notes. The node values are the required net input flow and the edge values are the flow costs and flow bounds respectively. In this illustrative example, $n = 7$, $m = 3$, $\mathcal{N}_1 = \{1,2\}$, $\mathcal{N}_2 = \{3,4,5\}$, $\mathcal{N}_3 = \{6,7\}$, and each flow variable y_j has the interpretation that $y_j \equiv \sum_{i \in \mathcal{N}_j} x_i$ for $j \in \mathcal{M} := \{1,2,3\}$.

characterizes the class of strictly convex instances to the problem. This class includes problems in which some or all the factors w_j are negative and, in particular, includes the three-phase EV charging problem. Our algorithms are, in their essence, breakpoint search algorithms that, as mentioned before, exploit the structure of the KKT conditions to efficiently search for the optimal (dual) multiplier associated with Resource Constraint (1b). We show that for (strictly) convex instances of Problem QRAP-NonSep-GBC, these conditions can be exploited in a similar way.

For the case where all subsets $\mathcal{N}_j$ have the same size, that is, where all n_j's are equal to some constant C, we show that one of the derived algorithms runs in $O(m\text{C}\log C)$ time, that is, given C this algorithm has a linear time complexity. In particular, we show that the three-phase EV charging problem can be solved in $O(n)$ time. Furthermore, we show for the special case where all weights w_j are zero, that is, the quadratic separable resource allocation problem with generalized bound constraints, that both Problem QRAP-NonSep-GBC and its version with integer variables can be solved in $O(n)$ time. Although the version with integer variables is not the main focus of this article, it may be of independent interest for research on general resource allocation problems where often both the continuous and integer version of a given resource allocation problem are studied in parallel (Hochbaum 1994, Moriguchi et al. 2011).

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 suggest that our algorithms are suitable for integration in DEM systems because they are fast and do not require much memory. Furthermore, they show that our algorithms scale well when the number m of subsets or the subset sizes n_j increases, that is, 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 article is organized as follows. In Section 2, we explain in more detail the application of Problem QRAP-NonSep-GBC in DEM and, specifically, in three-phase EV scheduling. In Section 3, we analyze the structure of Problem QRAP-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 QRAP-NonSep-GBC in Section 4, and in Section 5, we present two $O(n\log n)$ algorithms based on this approach. In Section 6, we evaluate the performance of our algorithms, and finally, Section 7 contains some concluding remarks.

Summarizing, the contributions of this article are as follows:

1. We derive two $O(n\log n)$ time algorithms for Problem QRAP-NonSep-GBC. In contrast to existing work of Tamir (1993) and Moriguchi et al. (2011), this algorithm can be applied to all strictly convex instances of Problem QRAP-NonSep-GBC, even those where some or all the factors w_j are negative.
2. For the special case where all subsets $\mathcal{N}_j$ have the same size C, we show that one of our algorithm runs in $O(m\text{C}\log C)$ time, which implies that for a fixed C the execution time of this algorithm scales linearly in the number m of subsets and thereby also in the total number n of variables.
3. Our algorithm solves an important problem in DEM and can make a significant impact on the integration of EVs in residential distribution grids.

2. Motivation

In this section, we describe in more detail our motivation for studying Problem QRAP-NonSep-GBC. For this, Section 2.1 provides a short introduction to load balancing in three-phase electricity networks and discusses the relevance of minimizing load unbalance. In Section 2.2, we formulate the three-phase EV charging problem and show that this problem is an instance of Problem QRAP-NonSep-GBC.

2.1. Load Balancing in Three-Phase Electricity Networks

Load balancing has as goal to distribute the power consumption of a neighborhood over a given time horizon and over the three phases of the low-voltage network such that peak consumption and unbalance between phases is minimized. 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 undesirable because it induces an increase in energy losses, stress on grid assets such as transformers, and can even lead to outages (Hoogsteen et al. 2017). As a consequence, many DEM systems in the literature take into account the minimization of peak consumption when scheduling, for example, EV charging (Gan et al. 2013, Gerards et al. 2015, Mou et al. 2015).

However, minimization of load unbalance between phases is hardly considered in optimization approaches for EV scheduling. To explain the relevance of load unbalance minimization, in the following we first consider

three-phase electricity networks in general (for a more detailed and comprehensive introduction to this topic, we refer to Stevenson (1975) and Kuphaldt (2007)).

In residential electricity distribution networks (or, more generally, low-voltage networks), electrical energy is transported by electrical current that flows through a conductor (e.g., a wire). This current can be seen as a signal with a given frequency and amplitude, which leads to (alternating current) power, that is, 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 equidistance 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 2 illustrates the concept of three-phase systems.

To maximize the efficiency of a three-phase network, ideally the power consumption from all three phases is equal. When this is not the case, negative effects similar to those of peak consumption can occur, that is, energy losses, wearing of grid assets, and outages. With the increasing penetration of 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 (Schoot Uiterkamp 2016), 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 (Hoogsteen et al. 2017).

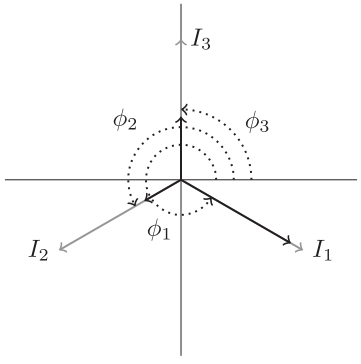
Recently, Weckx and Driesen (2015) and Schoot Uiterkamp et al. (2017) explored the potential of three-phase EV charging, that is, allowing an EV to distribute its charging over the three phases for minimizing load unbalance. Both works suggest that three-phase EV charging can significantly reduce the distribution losses and stress on the grid compared with single-phase EV charging, even when using the same DEM methodology.

2.2. 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 QRAP-NonSep-GBC. For this, we consider a division of the scheduling horizon into m equidistant time intervals of length Δt labeled according 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 $q_{j,p}$ be the remaining household power consumption drawn from phase p during interval j. This consumption is assumed to be known or, alternatively, could be substituted by, for example, the expected value of the consumption. Furthermore, we assume that we know on forehand the total required energy that must be charged by the EV and denote this requirement by $\tilde{R}$. 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. Thus, noting that the term $\sum_{p=1}^{3}(q_{j,p} + z_{j,p})$ represents the total power

Figure 2. Three-Phase System



Notes. 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.

consumption during interval j, we model this objective by minimizing the function

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

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

$$\sum_{j=1}^{m}\left(\frac{3}{2}\sum_{p=1}^{3}(q_{j,p}+z_{j,p})^2-\frac{1}{2}\left(\sum_{p=1}^{3}(q_{j,p}+z_{j,p})\right)^2\right) \tag{2}$$

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

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

Here, W_1 and W_2 are positive weights that express the tradeoff between the two objectives. This tradeoff could, for example, be chosen based on the resistance of the three-phase cable, say r_1, and of the cables of the individual phases within the household, say r_2, so that energy losses are minimized. More precisely, by choosing $W_1=\frac{1}{2}r_1+\frac{1}{6}r_2$ and $W_2=\frac{1}{3}r_2$, the objective function becomes

$$\begin{aligned}&\left(\frac{1}{2}r_1+\frac{1}{6}r_2\right)\sum_{j=1}^{m}\left(\sum_{p=1}^{3}(q_{j,p}+z_{j,p})\right)^2+\frac{1}{3}r_2\sum_{j=1}^{m}\left(\frac{3}{2}\sum_{p=1}^{3}(q_{j,p}+z_{j,p})^2-\frac{1}{2}\left(\sum_{p=1}^{3}(q_{j,p}+z_{j,p})\right)^2\right),\\ &=\frac{1}{2}r_1\sum_{j=1}^{m}\left(\sum_{p=1}^{3}(q_{j,p}+z_{j,p})\right)^2+\frac{1}{2}r_2\sum_{j=1}^{m}\sum_{p=1}^{3}(q_{j,p}+z_{j,p})^2.\end{aligned}$$

Because energy losses within a cable are proportional to the square of the power, minimizing this objective function ensures that the energy losses within the cables are minimized.

When varying the weights W_1 and W_2, the tradeoff between the two parts of the objective changes and thus also the resulting optimal solution to the problem changes. More precisely, given an instance of the problem and values for the weights W_1 and W_2, we observe the following changes to the optimal solution when one of the weights is slightly increased. If W_1 increases, it becomes more important to divide the total load equally over the time intervals. As a consequence, in the new optimal solution some EV load shifts from intervals with extreme total load (e.g., relatively high consumption peaks) to intervals with a close to average load, compared with the optimal solution before adapting the weights. Whether a certain time interval is selected for this load shifting depends on the extremeness of the total interval load and on the load balance between the phases within the interval.

In contrast, increasing W_2 balances the new optimal solution over phases. Again, load shifts between time intervals within the original optimal solution and these shifts depend on the load balance within the interval and the extremeness of the load of the interval. More precisely, intervals with much unbalance are favored for load shifts over those that are already quite balanced. Moreover, intervals whose total load is extreme compared with that of other intervals are more likely to receive (lose) more load when their total load is very low (high) compared with that of other intervals. We note that a mathematically based derivation of this behavior can be achieved based on the KKT optimality conditions (3) given in Section 3.3 for the general Problem QRAP-NonSep-GBC.

By choosing the parameters as given in Table 1, this problem becomes an instance of Problem QRAP-NonSep-GBC (see also Appendix A). 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, for example, the algorithms by Tamir (1993) and Moriguchi et al. (2011).

Table 1. Modeling Problem EV-3Phase as an Instance of Problem QRAP-NonSep-GBC

Parameter/variable in Problem QRAP-NonSep-GBC		Parameter/variable 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,	$i \in \mathcal{N}$	$3W_2$	
b_i,	$j \in \mathcal{M}, i \in \mathcal{N}_j$	$\left(W_1 - \frac{1}{2}W_2\right)\sum_{p=1}^{3} q_{j,p} + \frac{3}{2}W_2 q_{j,p}$	
R		$\frac{\tilde{R}}{\Delta t}$	
$(l_i)_{i\in\mathcal{N}_j}$,	$j \in \mathcal{M}$	$(\tilde{l}_{j,p})_{p\in\mathcal{P}}$	
$(u_i)_{i\in\mathcal{N}_j}$,	$j \in \mathcal{M}$	$(\tilde{u}_{j,p})_{p\in\mathcal{P}}$	
L_j,	$j \in \mathcal{M}$	$\tilde{L}_j$,	$j \in \mathcal{M}$
U_j,	$j \in \mathcal{M}$	$\tilde{U}_j$,	$j \in \mathcal{M}$

In this article, we develop algorithms to solve Problem EV-3Phase when all power consumption values $q_{j,p}$ are known. In a stochastic loads setting, where these values are not known on forehand, existing frameworks for optimization under uncertainty can be used to solve a stochastic loads setting, for example, robust optimization, stochastic programming, and online optimization (see Bakker et al. 2020 for a survey). Alternatively, recently, we have developed a new framework for solving particular classes of convex optimization problems under uncertainty (Schoot Uiterkamp et al. 2020). Applied to Problem EV-3Phase, this framework solves the problem where the quantities $q_{j,p}$ become known only at the start of time interval j and the variables $z_{j,p}$ have to be determined directly afterward. As input, the framework requires a prediction of the optimal Lagrange multiplier of this problem. Such a prediction could be made by computing the optimal multiplier for instances with representative historical values of $q_{j,p}$, which can be done using the algorithms developed in this article.

3. Analysis

In this section, we consider the general version of Problem QRAP-NonSep-GBC and derive some of its properties. First, in Section 3.1, we derive a necessary and sufficient condition on the vectors w and a for strict convexity of Problem QRAP-NonSep-GBC. Moreover, we show that the three-phase EV charging problem as presented in Section 2.2 satisfies this condition. Second, in Section 3.2, we show that we may replace Constraint (1c) by equivalent single-variable constraints without changing the optimal solution to the problem. This greatly simplifies the derivation of our solution approach in Section 4. Third, in Section 3.3, we derive a property of the structure of optimal solutions to Problem QRAP-NonSep-GBC that forms the crucial ingredient for our solution approach to solve the problem.

3.1. Convex Instances of Problem QRAP-NonSep-GBC

Because all constraints of Problem QRAP-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. Because this objective function is separable over the indices j, it suffices to investigate for each $j \in \mathcal{M}$ separately if the function

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

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

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

where $\boldsymbol{e}$ is the vector of ones of appropriate size and $\boldsymbol{a}^j := (a_i)_{i\in\mathcal{N}_j}$. Lemma 1 provides a characterization for which choices of w_j and $\boldsymbol{a}^j$ the Hessian H^j is positive definite. This characterization can also be obtained as a special case of theorem 1 of Spedicato (1975).

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

Proof. See Appendix B.1. □

Lemma 1 implies that an instance of Problem QRAP-NonSep-GBC is strictly convex if and only if $1 + w_j\sum_{i'\in\mathcal{N}_j} 1/a_{i'} > 0$ 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 1. For each $j \in \mathcal{M}$, it holds that $1 + w_j \sum_{i' \in \mathcal{N}_j} 1/a_{i'} > 0$.

For the remainder of this article, we consider only instances of Problem QRAP-NonSep-GBC that satisfy Property 1. We conclude this section by observing that the parameters for Problem EV-3Phase satisfy this property:

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

3.2. Constraint Elimination

In Section 3.1, we studied properties of the objective function of Problem QRAP-NonSep-GBC. In contrast, we focus in this section on properties of the *constraints* of Problem QRAP-NonSep-GBC. For this, note that it is the addition of the lower and upper bound constraints (1c) that make the constraint set of Problem QRAP-NonSep-GBC complex compared with the constraint set of the original resource allocation problem RAP. Therefore, the goal of this section is to reduce this complexity. More precisely, in this section, we show that we can replace the lower and upper bound constraints (1c) by a set of single-variable constraints without changing the optimal solution to Problem QRAP-NonSep-GBC. As these single-variable constraints can be integrated into the existing single-variable constraints (1d), we can focus without loss of generality on solving Problem QRAP-NonSep-GBC without this constraint.

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

$$\begin{aligned} \text{QRAP}^j(S): \min_{\boldsymbol{x} \in \mathbb{R}^{n_j}} \ & \sum_{i \in \mathcal{N}_j} \left(\frac{1}{2} a_i x_i^2 + b_i x_i \right), \\ \text{s.t.} \ & \sum_{i \in \mathcal{N}_j} x_i = S, \\ & l_i \le x_i \le u_i, \quad i \in \mathcal{N}_j. \end{aligned}$$

Lemma 2 states the main result of this section, namely that optimal solutions to QRAP$^j(L_j)$ and QRAP$^j(U_j)$ for $j \in \mathcal{M}$ are component-wise valid lower and upper bounds on optimal solutions to Problem QRAP-NonSep-GBC. The proof of this lemma is inspired by the proof of lemma 6.2.1 in Hochbaum (1994) and can be found in Appendix B.2.

Lemma 2. *For a given $j \in \mathcal{M}$, let $\underline{\boldsymbol{x}}^j := (\underline{x}_i)_{i \in \mathcal{N}}$ and $\bar{\boldsymbol{x}}^j := (\bar{x}_i)_{i \in \mathcal{N}}$ be optimal solutions to QRAP$^j(L_j)$ and QRAP$^j(U_j)$ respectively. Then there exists an optimal solution $\boldsymbol{x}^* := (x_i^*)_{i \in \mathcal{N}}$ to Problem QRAP-NonSep-GBC that satisfies $\underline{x}_i \le x_i^* \le \bar{x}_i$ for each $i \in \mathcal{N}_j$.*

Lemma 2 implies that adding the inequalities $\underline{x}_i \le x_i \le \bar{x}_i$, $i \in \mathcal{N}$ to the formulation of Problem QRAP-NonSep-GBC does not cut off the optimal solution to the problem. Moreover, these inequalities imply the generalized bound constraints (1c) because 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 QRAP-NonSep-GBC has the same optimal solution as the following problem:

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

To compute the new variable bounds $\underline{x}_i$ and $\bar{x}_i$, we solve the $2m$ subproblems QRAP$^j(L_j)$ and QRAP$^j(U_j)$. Because each subproblem is a simple resource allocation problem, this can be done in $O(n)$ time using, for example, the algorithms in Kiwiel (2007). Thus, in the remainder of this article and without loss of generality, we focus on solving Problem QRAP-NonSep-GBC without Constraint (1c).

3.3. Monotonicity of Optimal Solutions

In this section, we analyze Problem QRAP-NonSep-GBC (without Constraint (1c)) and the structure of its optimal solutions. More precisely, we study the KKT conditions (Boyd and Vandenberghe 2004) 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 QRAP-NonSep-GBC because it allows us to apply breakpoint search methods for separable convex resource allocation problems.

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

$$w_j y_j + a_i x_i + b_i + \lambda + \mu_i = 0, \quad j \in \mathcal{M}, i \in \mathcal{N}_j \qquad \text{(stationarity)}, \tag{3a}$$

$$\sum_{i=1}^{n} x_i = R \qquad \text{(primal feasibility)}, \tag{3b}$$

$$l_i \le x_i \le u_i, \quad i \in \mathcal{N} \qquad \text{(primal feasibility)}, \tag{3c}$$

$$\mu_i^+ (x_i - u_i) = 0, \quad i \in \mathcal{N} \qquad \text{(complementary slackness)}, \tag{3d}$$

$$\mu_i^- (x_i - l_i) = 0, \quad i \in \mathcal{N} \qquad \text{(complementary slackness)}, \tag{3e}$$

$$\lambda, \mu_i, \in \mathbb{R}, \quad i \in \mathcal{N} \qquad \text{(dual feasibility)}. \tag{3f}$$

Here, μ_i^+ and μ_i^- are the positive and negative part of μ_i respectively; that is, $\mu_i^+ = \max(0, \mu_i)$ and $\mu_i^- = \min(0, \mu_i)$. Slater's condition holds because the objective function of the problem is convex, all constraints are linear, and a feasible solution to the problem exists. Therefore, the KKT conditions are necessary and sufficient for optimality (Boyd and Vandenberghe 2004). Moreover, because Problem QRAP-NonSep-GBC is *strictly* convex, it has a unique optimal solution $\boldsymbol{x}^*$.

For a given λ, let $(\boldsymbol{x}(\lambda), \boldsymbol{\mu}(\lambda)) \in \mathbb{R}^{2n}$ be the solution that satisfies all KKT conditions (3) except (3b). Moreover, define $y_j(\lambda) := \sum_{i \in \mathcal{N}_j} x_i(\lambda)$ for $j \in \mathcal{M}$. It follows that $\boldsymbol{x}(\lambda)$ is the optimal solution to Problem QRAP-NonSep-GBC if and only if it satisfies KKT Condition (3b), that is, if $\sum_{i=1}^{n} x_i(\lambda) = R$. The core of our solution approach is to find a value λ^* such that $\sum_{i=1}^{n} x_i(\lambda^*) = R$ and reconstruct the corresponding solution $\boldsymbol{x}(\lambda^*)$ that, by definition, is optimal to Problem QRAP-NonSep-GBC. We call λ^* an *optimal (Lagrange) multiplier*.

The main result of this section is Lemma 4, which states that each $x_i(\lambda)$ can be seen as a nonincreasing function of λ. This result allows us to use approaches for separable convex resource allocation problems to find λ^*. To prove Lemma 4, we first identify in Lemma 3 a relation between $x_i(\lambda)$ and $\mu_i(\lambda)$.

Lemma 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 $\mu_i(\lambda_1) \le \mu_i(\lambda_2)$.*

Proof. Suppose $x_i(\lambda_1) < x_i(\lambda_2)$ for some i. Then $l_i \le x_i(\lambda_1) < x_i(\lambda_2) \le u_i$, which implies $x_i(\lambda_1) < u_i$ and $x_i(\lambda_2) > l_i$. Together with KKT conditions (3d) and (3e), it follows that $\mu_i(\lambda_1) \le 0$ and $\mu_i(\lambda_2) \ge 0$, respectively, which implies that $\mu_i(\lambda_1) \le \mu_i(\lambda_2)$. □

Lemma 4. *For any $\lambda_1, \lambda_2 \in \mathbb{R}$ such that $\lambda_1 < \lambda_2$, it holds that $x_i(\lambda_1) \ge x_i(\lambda_2)$, $i \in \mathcal{N}$.*

Proof. See Appendix B.3. □

Lemma 4 implies that the values $x_i(\lambda)$ are monotonically decreasing in λ. As a consequence, all possible values for the optimal multiplier λ^* form a closed interval $I \subset \mathbb{R}$, i.e., $\lambda \in I$ if and only if $\sum_{i=1}^{n} x_i(\lambda) = R$. It follows that λ^* is nonunique if and only if for each index $i \in \mathcal{N}$ one of the two bound constraints (1d) are tight for i, that is, either $x_i^* = l_i$ or $x_i^* = u_i$ for all $i \in \mathcal{N}$. Because this constitutes an extreme case and to simplify the discussion, we assume in the derivation of our approach without loss of generality that the optimal multiplier λ^* is unique.

The monotonicity of the values $x_i(\lambda)$ forms the main ingredient for our solution approach to Problem QRAP-NonSep-GBC, which we derive in Section 4. We conclude this section with two corollaries of Lemma 4 that we require for the derivation of this approach. The first corollary states that not only the values $x_i(\lambda)$ are decreasing in λ, but also each value $y_j(\lambda)$. The second corollary is a stronger version of Lemma 3 for the case where $i \in \mathcal{N}_j$ with $w_j < 0$.

Corollary 1. *For any $\lambda_1, \lambda_2 \in \mathbb{R}$ such that $\lambda_1 < \lambda_2$, it holds that $y_j(\lambda_1) \ge y_j(\lambda_2)$, $j \in \mathcal{M}$.*

Proof. Follows directly from Lemma 4. □

Corollary 2. *If $w_j < 0$, then for any $\lambda_1, \lambda_2 \in \mathbb{R}$ such that $\lambda_1 < \lambda_2$, it holds that $\mu_i(\lambda_1) \ge \mu_i(\lambda_2)$ for $i \in \mathcal{N}_j$.*

Proof. By Lemma 4, we have $x_i(\lambda_1) \ge x_i(\lambda_2)$. If this is a strict inequality, that is, if $x_i(\lambda_1) > x_i(\lambda_2)$, then it follows from Lemma 3 that $\mu_i(\lambda_1) \ge \mu_i(\lambda_2)$. Otherwise, if $x_i(\lambda_1) = x_i(\lambda_2)$, KKT Condition (3a), together with $w_j < 0$ and Corollary 1, implies

$$\begin{aligned} w_j y_j(\lambda_1) + a_i x_i(\lambda_1) + b_i + \mu_i(\lambda_1) &= -\lambda_1 > -\lambda_2, \\ &= w_j y_j(\lambda_2) + a_i x_i(\lambda_2) + b_i + \mu_i(\lambda_2), \\ &\geq w_j y_j(\lambda_1) + a_i x_i(\lambda_1) + b_i + \mu_i(\lambda_2). \end{aligned}$$

It follows that $\mu_i(\lambda_1) > \mu_i(\lambda_2)$, proving the corollary. □

4. Solution Approach

In this section, we present our approach to solve Problem QRAP-NonSep-GBC. First, in Section 4.1, we provide an outline of the approach using the analysis conducted in Section 3. Second, Section 4.2 focuses in detail on several computational aspects of the approach.

4.1. Outline

The monotonicity of $x_i(\lambda)$, proven in Lemma 4, has two important implications. First, for each $i \in \mathcal{N}$, there exist unique *breakpoints* $\alpha_i < \beta_i$ such that

$$\lambda \leq \alpha_i \Longleftrightarrow x_i(\lambda) = u_i, \tag{4a}$$

$$\alpha_i < \lambda < \beta_i \Longleftrightarrow l_i < x_i(\lambda) < u_i, \tag{4b}$$

$$\beta_i \leq \lambda \Longleftrightarrow x_i(\lambda) = l_i. \tag{4c}$$

For now, we assume that these breakpoints are known. In Section 4.2.2, we discuss how they can be computed efficiently. The second implication of the monotonicity is that, given the optimal multiplier λ^*, we have

$$\lambda \leq \lambda^* \Rightarrow \sum_{i=1}^{n} x_i(\lambda) \geq \sum_{i=1}^{n} x_i(\lambda^*) = R, \tag{5a}$$

$$\lambda \geq \lambda^* \Rightarrow \sum_{i=1}^{n} x_i(\lambda) \leq \sum_{i=1}^{n} x_i(\lambda^*) = R. \tag{5b}$$

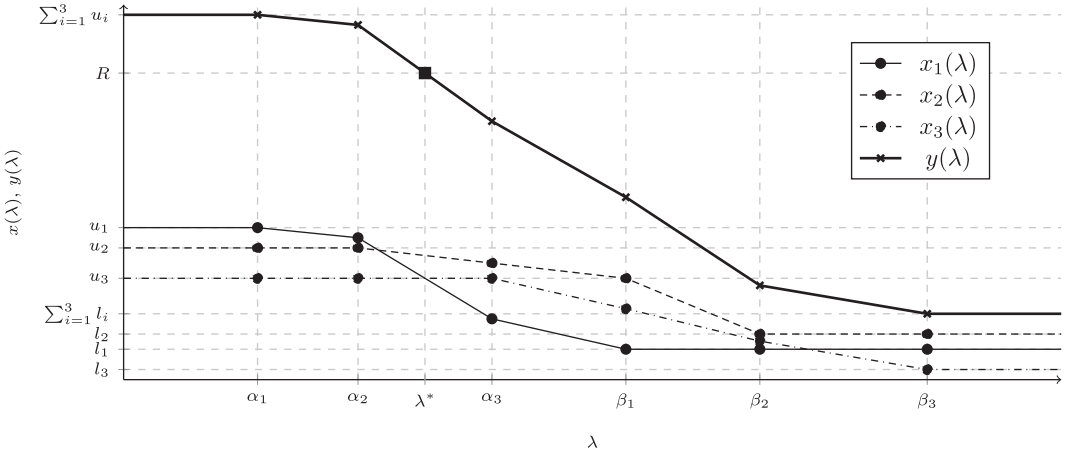
These two implications are the base to determine the optimal multiplier λ^*. For this, we define the set of all breakpoints by $\mathcal{B} := \{\alpha_i | i \in \mathcal{N}\} \cup \{\beta_i | i \in \mathcal{N}\}$. Equations (4a)–(4c) imply that $\min(\mathcal{B}) \leq \lambda^* \leq \max(\mathcal{B})$. This means that there exist two consecutive breakpoints $\gamma, \delta \in \mathcal{B}$ such that $\gamma \leq \lambda^* < \delta$. Figure 3 illustrates the relation between $\boldsymbol{x}(\lambda)$, $y(\lambda)$, the total resource R, the breakpoints in $\mathcal{B}$, and the breakpoints γ, λ^*, and δ.

The key of our approach is that once we have found γ and δ, we can easily compute λ^* and the resulting optimal solution $\boldsymbol{x}(\lambda^*)$. To see this, note that by Equations (4a)–(4c) and by definition of γ, we have for all $i \in \mathcal{N}$ that

$$\begin{aligned} x_i(\delta) = u_i &\Longleftrightarrow x_i(\lambda^*) = u_i, \\ l_i < x_i(\gamma) < u_i &\Longleftrightarrow l_i < x_i(\lambda^*) < u_i, \\ x_i(\gamma) = l_i &\Longleftrightarrow x_i(\lambda^*) = l_i. \end{aligned}$$

As a consequence, we know that $x_i(\lambda^*) = u_i$ if $\alpha_i \geq \delta$ and $x_i(\lambda^*) = l_i$ if $\beta_i \leq \gamma$. Thus, we may eliminate these variables from the problem. As a consequence, for the remaining problem, Box Constraints (1d) become redundant

Figure 3. Relation Between $\boldsymbol{x}(\lambda)$, $y(\lambda) = \sum_{i=1}^{3} x_i(\lambda)$, R, and the Breakpoints α_i, β_i, γ, λ^*, and δ



Note. In this example, $\gamma = \alpha_2$, $\delta = \alpha_3$, and λ^* is represented by the black square.

and Problem QRAP-NonSep-GBC reduces to a quadratic optimization problem with a single equality constraint. We show in Section 4.2.3 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 γ, we may either consider all breakpoints monotonically in the set $\mathcal{B}$ of breakpoints or apply a binary search to $\mathcal{B}$. This is because the variable sum $y_i(\lambda) = \sum_{i=1}^{n} \boldsymbol{x}_i(\lambda)$ induces an order on the breakpoints by Corollary 1. Moreover, we know by Equations (5a) and (5b) that γ is the largest breakpoint λ in the set $\mathcal{B}$ such that $\sum_{i=1}^{n} x_i(\lambda) \geq R$. Each of the two approaches to find γ leads to a different algorithm.

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

4.2. Computational Aspects

In the approach outlined in Section 4.1, there are three quantities whose computation is not straight-forward. These quantities are the solution $\boldsymbol{x}(\lambda)$ for a given $\lambda \in \mathbb{R}$, the set of breakpoints $\mathcal{B}$, and the optimal multiplier λ^*. In the following sections, we discuss how these quantities can be computed efficiently.

4.2.1. Computing $\boldsymbol{x}(\lambda)$ and $\boldsymbol{y}(\lambda)$ for a Given λ. To compute $\boldsymbol{x}(\lambda)$ for a given λ, we need to find a feasible solution to KKT Conditions (3) without (3b). We call these KKT conditions the *primary* KKT conditions. Instead of trying to derive $\boldsymbol{x}(\lambda)$ directly from the primary KKT-conditions, we first determine which variables in $\boldsymbol{x}(\lambda)$ are equal to one of their bounds and which ones are strictly in between their bounds. To this end, for each $j \in \mathcal{M}$, we first partition the set of variables $\mathcal{N}_j$ into the following sets:

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

Observe that Equation (4) imply the following equivalent definition of these sets:

$$\mathcal{N}_j^{\text{lower}}(\lambda) = \{i \in \mathcal{N}_j | \beta_i \leq \lambda\}, \tag{6a}$$
$$\mathcal{N}_j^{\text{upper}}(\lambda) = \{i \in \mathcal{N}_j | \alpha_i \geq \lambda\}, \tag{6b}$$
$$\mathcal{N}_j^{\text{free}}(\lambda) = \{i \in \mathcal{N}_j | \alpha_i < \lambda < \beta_i\}. \tag{6c}$$

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

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

By KKT Conditions (3d) and (3e), 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 primary KKT Conditions (3a) and (3c)–(3e), the only non-redundant primary KKT conditions are (3a) and (3f) for $i \in \mathcal{N}_j^{\text{free}}(\lambda)$:

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

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:

$$A_j(\lambda) := \sum_{\ell \in \mathcal{N}_j^{\text{free}}(\lambda)} \frac{1}{a_\ell}, \quad B_j(\lambda) := \sum_{\ell \in \mathcal{N}_j^{\text{free}}(\lambda)} \frac{b_\ell}{a_\ell}.$$

Using, for example, the Sherwood-Morrison formula (Bartlett 1951), one can deduce and verify that the solution to the nonredundant primary KKT conditions is

$$x_i(\lambda) = \frac{1}{a_i}\frac{-w_j y_j^{\text{fixed}}(\lambda) - \lambda}{1 + w_j A_j(\lambda)} - \frac{b_i}{a_i} + \frac{w_j}{a_i}\frac{B_j(\lambda)}{1 + w_j A_j(\lambda)}, \quad i \in \mathcal{N}_j^{\text{free}}(\lambda). \tag{7}$$

It follows that

$$\begin{aligned} y_j(\lambda) &:= y_j^{\text{free}}(\lambda) + y_j^{\text{fixed}}(\lambda) = -\frac{a_i x_i(\lambda) + b_i + \lambda}{w_j} \\ &= \frac{y_j^{\text{fixed}}(\lambda) + \frac{\lambda}{w_j}}{1 + w_j A_j(\lambda)} - \frac{B_j(\lambda)}{1 + w_j A_j(\lambda)} - \frac{\lambda}{w_j} = \frac{y_j^{\text{fixed}}(\lambda) - B_j(\lambda)}{1 + w_j A_j(\lambda)} - \frac{A_j(\lambda)}{1 + w_j A_j(\lambda)}\lambda. \end{aligned} \tag{8}$$

For each $j \in \mathcal{M}$, $y_j(\lambda)$ can be computed in $O(n_j)$ time given the breakpoint set $\mathcal{B}$. As a consequence, computing the sum $\sum_{j=1}^{m} y_j(\lambda) (= \sum_{i=1}^{n} x_i(\lambda))$ takes $O(n)$ time.

4.2.2. Computing the Breakpoints. To derive our approach for computing the breakpoints, we exploit two important properties of these breakpoints that we state and prove in Lemmas 5 and 6. The first property is concerned with the value μ introduced in KKT Conditions (3). Recall from KKT Conditions (3d) and (3e) 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 (4) that $\mu_i(\lambda) \geq 0$ if $\lambda \leq \alpha_i$, $\mu_i(\lambda) = 0$ if $\alpha_i < \lambda < \beta_i$, and $\mu_i(\lambda) \leq 0$ if $\beta_i \leq \lambda$. Lemma 5 shows that $\mu_i(\alpha)$ and $\mu_i(\beta_i)$ are equal to the value of $\mu_i(\lambda)$ for $\alpha_i < \lambda < \beta_i$, that is, are equal to zero.

Lemma 5. *For all $i \in \mathcal{N}$, we have $\mu_i(\alpha_i) = \mu_i(\beta_i) = 0$.*

Proof. See Appendix B.4. □

Next, Lemma 6 states that, for each $j \in \mathcal{M}$, we can use the values given by $\mathcal{P}_j := \{a_i l_i + b_i | i \in \mathcal{N}_j\}$ and $\mathcal{Q}_j := \{a_i u_i + b_i | i \in \mathcal{N}_j\}$ to determine the order of the corresponding breakpoints.

Lemma 6. *For $j \in \mathcal{M}$ and $i,k \in \mathcal{N}_j$, we have:*
- *that $a_i u_i + b_i > a_k u_k + b_k$ implies $\alpha_i \leq \alpha_k$, and;*
- *that $a_i l_i + b_i > a_k l_k + b_k$ implies $\beta_i \leq \beta_k$.*

Proof. See Appendix B.5. □

Lemmas 5 and 6 give rise to the following strategy to compute the breakpoints. From KKT Condition (3a) for $i \in \mathcal{N}_j$, $j \in \mathcal{M}$, we have for the breakpoints α_i and β_i that

$$\alpha_i = -w_j y_j(\alpha_i) - a_i x_i(\alpha_i) - b_i - \mu_i(\alpha_i) = -w_j y_j(\alpha_i) - a_i u_i - b_i - \mu_i(\alpha_i), \tag{9a}$$

$$\beta_i = -w_j y_j(\beta_i) - a_i x_i(\beta_i) - b_i - \mu_i(\beta_i) = -w_j y_j(\beta_i) - a_i l_i - b_i - \mu_i(\beta_i). \tag{9b}$$

We can obtain the following expression for a given breakpoint α_i by applying Lemma 5 and plugging Equation (8) into Equation (9a):

$$\alpha_i = -w_j y_j(\alpha_i) - a_i u_i - b_i = \frac{-w_j y_j^{\text{fixed}}(\alpha_i) + w_j B_j(\alpha_i)}{1 + w_j A_j(\alpha_i)} + \frac{w_j A_j(\alpha_i)}{1 + w_j A_j(\alpha_i)}\alpha_i - a_i u_i - b_i. \tag{10}$$

This is equivalent to

$$\alpha_i = w_j(B_j(\alpha_i) - y_j^{\text{fixed}}(\alpha_i)) - (a_i u_i + b_i)(1 + w_j A_j(\alpha_i)). \tag{11}$$

Analogously, we can deduce the following expression for β_i by applying Lemma 5 and plugging Equation (8) into Equation (9b):

$$\beta_i = w_j(B_j(\beta_i) - y_j^{\text{fixed}}(\beta_i)) - (a_i l_i + b_i)(1 + w_j A_j(\beta_i)).$$

Using these two expressions, we can compute the breakpoints sequentially, that is, in ascending order. This order can be determined using Lemma 6 without knowledge of the actual values of the breakpoints. 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 $(\mathcal{N}_j^{\text{lower}}(\eta_k),$

Table 2. Relation Between Consecutive Breakpoints η_i and η_k and Their Index Set Partitions

Type of η_i	$\mathcal{N}_j^{\text{lower}}(\eta_k)$	$\mathcal{N}_j^{\text{upper}}(\eta_k)$	$\mathcal{N}_j^{\text{free}}(\eta_k)$	$y_j^{\text{fixed}}(\eta_k)$	$A_j(\eta_k)$	$B_j(\eta_k)$
$\eta_i \equiv \alpha_i$	$\mathcal{N}_j^{\text{lower}}(\eta_i)$	$\mathcal{N}_j^{\text{upper}}(\eta_i)\backslash\{i\}$	$\mathcal{N}_j^{\text{free}}(\eta_i) \cup \{i\}$	$y_j^{\text{fixed}}(\eta_i) - u_i$	$A_j(\eta_i) + \frac{1}{a_i}$	$B_j(\eta_i) + \frac{b_i}{a_i}$
$\eta_i \equiv \beta_i$	$\mathcal{N}_j^{\text{lower}}(\eta_i) \cup \{i\}$	$\mathcal{N}_j^{\text{upper}}(\eta_i)$	$\mathcal{N}_j^{\text{free}}(\eta_i)\backslash\{i\}$	$y_j^{\text{fixed}}(\eta_i) + l_i$	$A_j(\eta_i) - \frac{1}{a_i}$	$B_j(\eta_i) - \frac{b_i}{a_i}$

$\mathcal{N}_j^{\text{upper}}(\eta_k), \mathcal{N}_j^{\text{free}}(\eta_k)$) summarized in Table 2 (see also Figure 3 and Equation (6)). 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 $y_j^{\text{fixed}}(\bar{\eta}) := \sum_{i \in \mathcal{N}_j} u_i$, $A_j(\bar{\eta}) := 0$, and $B_j(\bar{\eta}) := 0$. To determine whether $\eta_k \equiv \alpha_k$ 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. Thus, 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 (10) and (11) that $\eta_k = \alpha_{k_1}$ if $a_{k_1} u_{k_1} + b_{k_1} > a_{k_2} l_{k_2} + b_{k_2}$ and $\eta_k = \beta_{k_2}$ otherwise.

Algorithm 1 summarizes this approach to compute the breakpoints. Each new smallest breakpoint η_k in Line 4 can be retrieved in $O(1)$ time if we maintain the values in $\mathcal{P}_j$ and $\mathcal{Q}_j$ as sorted lists. As a consequence, the time complexity of Algorithm 1 for a given $j \in \mathcal{M}$ is $O(n_j \log n_j)$. Thus, the computation of the breakpoints for all variables can be done in $O(n \log n)$ time. If each n_j is equal to a given constant C, that is, all subsets $\mathcal{N}_j$ have the same cardinality, this complexity can be refined to $O\left(\sum_{j=1}^{m} C \log C\right) = O(Cm \log C) = O(n \log C)$. Thus, for a given C in this case the breakpoints can be computed in linear time.

Algorithm 1 (Computing the Breakpoints for $j \in \mathcal{M}$)

Compute the sets $\mathcal{P}_j := \{a_i l_i + b_i \,|\, i \in \mathcal{N}_j\}$ and $\mathcal{Q}_j := \{a_i u_i + b_i | i \in \mathcal{N}_j\}$
Initialize $\bar{Y} := \sum_{i \in \mathcal{N}_j} u_i$; $\bar{A} := 0$; $\bar{B} := 0$
repeat
  Take smallest value $\eta_k := \min(\mathcal{P}_j \cup \mathcal{Q}_j)$
5: **if** $\eta_k \in \mathcal{Q}_j$ **then** {$\eta_k \equiv \alpha_k$; η_k is a lower breakpoint}
    $y_j^{\text{fixed}}(\alpha_k) = \bar{Y}$; $A_j(\alpha_k) = \bar{A}$; $B_j(\alpha_k) = \bar{B}$
    $\alpha_k := w_j(B_j(\alpha_k) - y_j^{\text{fixed}}(\alpha_k)) - (a_k u_k + b_k)(1 + w_j A_j(\alpha_k))$
    $\bar{Y} = \bar{Y} - u_k$; $\bar{A} = \bar{A} + \frac{1}{a_k}$; $\bar{B} = \bar{B} + \frac{b_k}{a_k}$
    $\mathcal{Q}_j = \mathcal{Q}_j \backslash \{\eta_k\}$
10: **else** {$\eta_k \equiv \beta_k$; η_k is an upper breakpoint}
    $y_j^{\text{fixed}}(\beta_k) = \bar{Y}$; $A_j(\beta_k) = \bar{A}$; $B_j(\beta_k) = \bar{B}$
    $\beta_k := w_j(B_j(\beta_k) - y_j^{\text{fixed}}(\beta_k)) - (a_k l_k + b_k)(1 + w_j A_j(\beta_k))$
    $\bar{Y} = \bar{Y} + l_k$; $\bar{A} = \bar{A} - \frac{1}{a_k}$; $\bar{B} = \bar{B} - \frac{b_k}{a_k}$ $\mathcal{P}_j = \mathcal{P}_j \backslash \{\eta_k\}$
15: **end if**
until $\mathcal{P}_j \cup \mathcal{Q}_j = \emptyset$

4.2.3. Computing λ^*. To finalize our approach, we need to compute λ^* for a given γ, which is the largest breakpoint such that $\gamma \le \lambda^*$. In Section 4.1, we showed that the partitioning of the variables under λ^* can be derived from the partitioning under γ and δ, that is, for each $j \in \mathcal{M}$, we have $\mathcal{N}_j^{\text{lower}}(\lambda^*) = \mathcal{N}_j^{\text{lower}}(\gamma)$, $\mathcal{N}_j^{\text{upper}}(\lambda^*) = \mathcal{N}_j^{\text{upper}}(\delta)$, and $\mathcal{N}_j^{\text{free}}(\lambda^*) = \mathcal{N}_j^{\text{free}}(\gamma)$. Moreover, as we have $\sum_{j=1}^{m} y_j(\lambda^*) = R$ by definition of y_j, we can apply the derived expression for general $y_j(\lambda)$ in Equation (8) to obtain the following linear equation in λ^*:

$$R = \sum_{j=1}^{m} y_j(\lambda^*) = \sum_{j=1}^{m} \left(\frac{y_j^{\text{fixed}}(\lambda^*) - B_j(\lambda^*)}{1 + w_j A_j(\lambda^*)} - \frac{A_j(\lambda^*)}{1 + w_j A_j(\lambda^*)} \lambda^* \right),$$
$$= \sum_{j=1}^{m} \left(\frac{y_j^{\text{fixed}}(\gamma) - B_j(\gamma)}{1 + w_j A_j(\gamma)} - \frac{A_j(\gamma)}{1 + w_j A_j(\gamma)} \lambda^* \right).$$

It follows that

$$\lambda^* = \frac{\left(\sum_{j=1}^{m} \frac{y_j^{\text{fixed}}(\gamma) - B_j(\gamma)}{1 + w_j A_j(\gamma)}\right) - R}{\sum_{j=1}^{m} \frac{A_j(\gamma)}{1 + w_j A_j(\gamma)}}. \tag{12}$$

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 $O\left(\sum_{j=1}^{m} n_j\right) = O(n)$ time.

5. Two Algorithms for Problem QRAP-NonSep-GBC

In this section, we present two algorithms that solve Problem QRAP-NonSep-GBC according to the approach presented in Section 4. This approach can be summarized by means of the following four steps:

1. Replace the generalized bound Constraints (1c) by the box constraints $\underline{x}_i \le x_i \le \bar{x}_i$, $i \in \mathcal{N}$ (Section 3.2),
2. Compute for each $i \in \mathcal{N}$ the lower and upper breakpoints α_i and β_i (Section 4.2.2),
3. Find γ (Section 4.1), and
4. Compute the optimal Lagrange multiplier λ^* (Section 4.2.3) and the optimal solution $x(\lambda^*)$ (Section 4.2.1).

Both algorithms follow these four steps. Their difference is in the execution of Step 3 or, more precisely, in how we search for γ through the breakpoint set $\mathcal{B}$. In the first algorithm, we consider the breakpoints sequentially starting from the smallest breakpoint, whereas in the second algorithm, we apply binary search on $\mathcal{B}$. We present and discuss these algorithms and their breakpoint search strategies in more detail in Sections 5.1 and 5.2.

5.1. An $O(n \log n)$ Time Algorithm Based on Sequential Breakpoint Search

The sequential breakpoint search strategy is similar to Algorithm 1 to compute the breakpoints, that is, we search through the breakpoint set $\mathcal{B}$ in ascending order. For each considered breakpoint η_k, $k \in \mathcal{N}_j$, we compute the sum $\sum_{j=1}^{m} y_j(\eta_k)$ using Equation (8). If $\sum_{j=1}^{m} y_j(\eta_k) > R$, it follows from Equation (5b) that $\eta_k < \lambda^*$ and we continue the search. Otherwise, if $\sum_{j=1}^{m} y_j(\eta_k) < R$, then it follows from Equation (5a) that $\eta_k > \lambda^*$, meaning that γ is the breakpoint preceding η_k and that $\delta = \eta_k$. Subsequently, we can use Equation (12) to compute λ^*. Finally, if $\sum_{j=1}^{m} y_j(\eta_k) = R$, then $\lambda^* = \eta_k$ by definition of the values $y_j(\cdot)$.

To efficiently compute the sum $\sum_{j=1}^{m} y_j(\eta_k)$, we exploit the dependencies between η_k and its preceding breakpoint η_i, $i \in \mathcal{N}_j$, given in Table 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 $\lambda \in \mathbb{R}$

$$F(\lambda) := \sum_{j=1}^{m} \frac{y_j^{\text{fixed}}(\lambda) - B_j(\lambda)}{1 + w_j A_j(\lambda)},$$

$$V(\lambda) := \frac{A_j(\lambda)}{1 + w_j A_j(\lambda)}.$$

and using these values and the dependencies in Table 2, we can easily compute $\sum_{j=1}^{m} y_j(\eta_k)$ from $\sum_{j=1}^{m} y_j(\eta_i)$ in $O(1)$ time. For this, $\sum_{j=1}^{m} y_j(\lambda) = F(\lambda) - \lambda V(\lambda)$ by Equation (8).

Algorithm 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 2 corresponds to Step 1, Line 3 to Step 2, Lines 5–36 to Step 3, and Lines 14 and 17 to Step 4. During each iteration τ of the sequential breakpoint search in Lines 5–38, the set $\mathcal{B}^\tau$ is the part of the original breakpoint set $\mathcal{B}$ that has not yet been searched in this iteration.

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

Theorem 1. *Algorithm 2 has a worst-case time complexity of* $O(n \log n)$.

Proof. First, the elimination of Constraint (1c) in Line 2 takes $O(n)$ time. Second, the computation of the breakpoints by means of Algorithm 1 in Line 3 takes $O(n \log n)$ 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 10 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 (7). Summarizing, the worst-case time complexity of Algorithm 2 is $O(n \log n)$. □

Besides the computation and sorting of the breakpoints, Algorithm 2 runs in linear time.

Algorithm 2 (Solving Problem QRAP-NonSep-GBC Using Sequential Breakpoint Search)

for $j \in \mathcal{M}$ **do**
  Solve QRAP$^j(L_j)$ and QRAP$^j(U_j)$ and set $l_i := \max(l_i, \underline{x}_i(\underline{\lambda}^j(L_j)))$ and $u_i := \min(u_i, \underline{x}_i(\underline{\lambda}^j(U_j)))$ {Solve QRAP subproblems to eliminate generalized bound constraints}
  Compute α_i and β_i for each $i \in \mathcal{N}_j$ using Algorithm 1 {Compute breakpoint values}
end for
$\mathcal{B} := \{\alpha_i | i \in \mathcal{N}\} \cup \{\beta_i | i \in \mathcal{N}\}$; $\tau := 0$; $\mathcal{B}^0 := \mathcal{B}$; $F := \sum_{i=1}^n u_i$; $V := 0$ {Initialize breakpoint sets and bookkeeping parameters}
For $j \in \mathcal{M}$: Initialize $\bar{Y}_j := \sum_{i \in \mathcal{N}_j} u_i$; $\bar{A}_j := 0$; $\bar{B}_j := 0$
while λ^* has not been found yet **do** {Breakpoint search procedure}
  Take smallest value $\eta_k := \min(\mathcal{B}^\tau)$ and j with $k \in \mathcal{N}_j$
  for $j' \in \mathcal{M}$ **do** {Update bookkeeping parameters}
    $y_{j'}^{\text{fixed}}(\eta_k) = \bar{Y}_{j'}$; $A_{j'}(\eta_k) = \bar{A}_{j'}$; $B_{j'}(\eta_k) = \bar{B}_{j'}$
  end for
  Compute $\sum_{j'=1}^m y_{j'}(\eta_k) = F - V\alpha_k$ {Variable sum under current candidate multiplier η_k}
  If $\sum_{j'=1}^m y_{j'}(\eta_k) = R$ **then**
    $\lambda^* = \eta_k$; compute $\boldsymbol{x}(\lambda^*)$ as $\boldsymbol{x}(\eta_k)$ using Equation (7)
    return
  else if $\sum_{j'=1}^m y_{j'}(\eta_k) < R$ **then**
    $\lambda^* = \frac{F-R}{V}$; compute $\boldsymbol{x}(\lambda^*)$ using Equation (7)
    return
  else {Breakpoint has not been found yet; update bookkeeping parameters and prepare for next candidate breakpoint}
    $F = F - \frac{\bar{Y}_j - \bar{B}_j}{1 + w_j \bar{A}_j}$
    $V = V - \frac{\bar{A}_j}{1 + w_j \bar{A}_j}$
    If $\eta_k \equiv \alpha_k$ **then**
      $\bar{Y}_j = \bar{Y}_j - u_i$
      $\bar{A}_j = \bar{A}_j + \frac{1}{a_i}$
      $\bar{B}_j = \bar{B}_j + \frac{b_i}{a_i}$
    else
      $\bar{Y}_j = \bar{Y}_j + l_i$
      $\bar{A}_j = \bar{A}_j - \frac{1}{a_i}$
      $\bar{B}_j = \bar{B}_j - \frac{b_i}{a_i}$
    end if
    $F = F + \frac{\bar{Y}_j - \bar{B}_j}{1 + w_j \bar{A}_j}$
    $V = V + \frac{\bar{A}_j}{1 + w_j \bar{A}_j}$
    $\mathcal{B}^{\tau+1} := \mathcal{B}^\tau \setminus \{\eta_k\}$
    $\tau = \tau + 1$
  end if
end while

5.2. An $O(n \log n)$ Time Algorithm Based on Binary Breakpoint Search

In this section, we present an alternative approach, where we apply binary search on the set of breakpoints. During each iteration τ of the binary search, we compute the median $\hat{\gamma}^\tau$ of the current breakpoint set $\mathcal{B}^\tau$, that is, of the subset of the original breakpoint set that is guaranteed to contain the breakpoint γ. For this median

breakpoint, we compute the sum $\sum_{j=1}^{m} y_j(\hat{\gamma}^\tau)$ and compare this value to the given amount R of the resource. If $\sum_{j=1}^{m} y_j(\hat{\gamma}^\tau) = R$, then $\lambda^* = \hat{\gamma}^\tau$. Otherwise, if $\sum_{j=1}^{m} y_j(\hat{\gamma}^\tau) < R$, then $\hat{\gamma}^\tau > \lambda^* \geq \gamma$ and during the next iteration $\tau+1$ we take as breakpoint set $\mathcal{B}^{\tau+1} := \{\lambda \in \mathcal{B}^\tau | \lambda < \hat{\gamma}^\tau\}$. Finally, if $\sum_{j=1}^{m} y_j(\hat{\gamma}^\tau) > R$, we have that $\hat{\gamma}^\tau < \lambda^* < \delta$ and during the next iteration $\tau+1$ we take as breakpoint set $\mathcal{B}^{\tau+1} := \{\lambda \in \mathcal{B}^\tau | \lambda \geq \hat{\gamma}^\tau\}$.

To efficiently compute each sum $\sum_{j=1}^{m} y_j(\hat{\gamma}^\tau)$, we use the following observation that is inspired by the breakpoint search approach in Kiwiel (2007) for separable quadratic resource allocation problems. 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 $\mathcal{B}^\tau$. 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 because of Equation (6):

$$\beta_i \leq \lambda_\downarrow^\tau \Rightarrow i \in \mathcal{N}_j^{\text{lower}}(\lambda), \tag{13a}$$

$$\alpha_i \leq \lambda_\downarrow^\tau \leq \lambda_\uparrow^\tau \leq \beta_i \Rightarrow i \in \mathcal{N}_j^{\text{free}}(\lambda), \tag{13b}$$

$$\lambda_\uparrow^\tau \leq \alpha_i \Rightarrow i \in \mathcal{N}_j^{\text{upper}}(\lambda). \tag{13c}$$

We introduce the following sets, which partition the set $\mathcal{N}_j$ according to which of the previous cases applies during iteration τ:

$$\mathcal{L}_j^\tau := \{i \in \mathcal{N}_j | \beta_i \leq \lambda_\downarrow^\tau\}, \tag{14a}$$

$$\mathcal{F}_j^\tau := \{i \in \mathcal{N}_j | \alpha_i \leq \lambda_\downarrow^\tau \leq \lambda_\uparrow^\tau \leq \beta_i\}, \tag{14b}$$

$$\mathcal{U}_j^\tau := \{i \in \mathcal{N}_j | \lambda_\uparrow^\tau \leq \alpha_i\}, \tag{14c}$$

$$\mathcal{I}_j^\tau := \mathcal{N}_j \backslash (\mathcal{L}_j^\tau \cup \mathcal{F}_j^\tau \cup \mathcal{U}_j^\tau) = \{i \in \mathcal{N}_j | \lambda_\downarrow^\tau < \alpha_i < \lambda_\uparrow^\tau \text{ or } \lambda_\downarrow^\tau < \beta_i < \lambda_\uparrow^\tau\} \tag{14d}$$

(see also Figure 4). For any λ such that $\lambda_\downarrow^\tau \leq \lambda \leq \lambda_\uparrow^\tau$, we have

$$i \in \mathcal{L}_j^\tau \Rightarrow i \in \mathcal{N}_j^{\text{lower}}(\lambda),$$

$$i \in \mathcal{F}_j^\tau \Rightarrow i \in \mathcal{N}_j^{\text{free}}(\lambda),$$

$$i \in \mathcal{U}_j^\tau \Rightarrow i \in \mathcal{N}_j^{\text{upper}}(\lambda).$$

Because of the construction of the sets $\mathcal{B}^\tau$, the sequence $(\lambda_\downarrow^\tau)_{\tau \in \mathbb{N}}$ is nondecreasing and the sequence $(\lambda_\uparrow^\tau)_{\tau \in \mathbb{N}}$ is nonincreasing. This implies that as soon as one of the three cases (13a), (13b), or (13c) occurs during an iteration τ for an index $i \in \mathcal{N}_j$, we already know for any future candidate breakpoint $\hat{\gamma}\bar{\tau}$ that $i \in \mathcal{N}_j^{\text{lower}}(\hat{\gamma}\bar{\tau})$, $i \in \mathcal{N}_j^{\text{free}}(\hat{\gamma}\bar{\tau})$, or $i \in \mathcal{N}_j^{\text{upper}}(\hat{\gamma}\bar{\tau})$ 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 $(\mathcal{N}_j^{\text{lower}}(\hat{\gamma}\bar{\tau}), \mathcal{N}_j^{\text{free}}(\hat{\gamma}^\tau), \mathcal{N}_j^{\text{upper}}(\hat{\gamma}\bar{\tau}))$, we only need to determine the membership of $x_k(\hat{\gamma}\bar{\tau})$ 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.

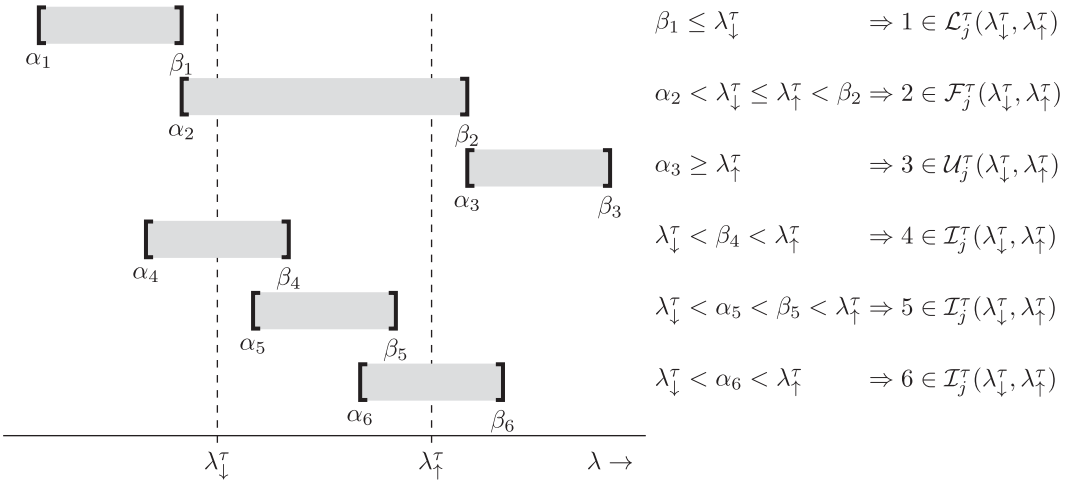
The main computational gain is obtained by introducing for each iteration τ the following bookkeeping parameters:

$$Y_j^\tau := \sum_{i \in \mathcal{L}_j^\tau} l_i + \sum_{i \in \mathcal{U}_j^\tau} u_i, \quad \bar{A}_j^\tau := \sum_{i \in \mathcal{F}_j^\tau} \frac{1}{a_i}, \quad \bar{B}_j^\tau := \sum_{i \in \mathcal{F}_j^\tau} \frac{b_i}{a_i}.$$

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 (8) can be done in $O(|\mathcal{I}_j^\tau|)$ time instead of $O(n_j)$ time.

We summarize the resulting four steps of our overall solution, using in Step 3 the discussed binary breakpoint search strategy, in Algorithm 3. In this algorithm, Line 2 corresponds to Step 1, Line 3 to Step 2, Lines 8–46 to Step 3, and Lines 24 and 47–48 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 $\mathcal{B}^{\tau+1}$ and lower and upper bounds $\lambda_\downarrow^{\tau+1}$ and $\lambda_\uparrow^{\tau+1}$ have been determined. This update can be done in line with 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 (14).

Figure 4. Partitioning of the Variables Based on Their Breakpoints and the Interval $[\lambda_{\downarrow}^{\tau}, \lambda_{\uparrow}^{\tau}]$



Algorithm 3 (Solving Problem QRAP-NonSep-GBC Using Binary Breakpoint Search)

for $j \in \mathcal{M}$ **do**
  Solve QRAP$^j(L_j)$ and QRAP$^j(U_j)$ and set $l_i := \max(l_i, \underline{x}_i(\underline{\lambda}^j(L_j)))$ and $u_i := \min(u_i, \underline{x}_i(\underline{\lambda}^j(U_j)))$ {Solve QRAP subproblems to eliminate generalized bound constraints}
  Compute α_i and β_i for each $i \in \mathcal{N}_j$ using Algorithm 1 {Compute breakpoint values}
end for
$\mathcal{B} := \{\alpha_i | i \in \mathcal{N}\} \cup \{\beta_i | i \in \mathcal{N}\}$; $\mathcal{B}^0 := \mathcal{B}$; $\tau := 0$ {Initialize breakpoint sets} For $j \in \mathcal{M}$: $\mathcal{I}_j^0 := \mathcal{N}_j$, $Y_j^0 = \bar{A}_j^0 = \bar{B}_j^0 = 0$
$\lambda_{\downarrow}^0 = -\infty$; $\lambda_{\uparrow}^0 = \infty$ {Initialize bookkeeping parameters and lower and upper bounds on optimal multiplier}
while $|\mathcal{B}^{\tau}| > 1$ {Breakpoint search procedure}
  $\hat{\gamma}^{\tau} := \text{median}(\mathcal{B}^{\tau})$
for $j \in \mathcal{M}$ **do** {Update bookkeeping parameters and determine which variables are fixed under current candidate multiplier η_k}
  $y_j^{\text{fixed}}(\hat{\gamma}^{\tau}) := Y_j^{\tau}$; $A_j(\hat{\gamma}) := \bar{A}_j^{\tau}$; $B_j(\hat{\gamma}) := \bar{B}_j^{\tau}$
  for $k \in \mathcal{I}_j^{\tau}$ **do**
    if $k \in \mathcal{N}_j^{\text{lower}}(\hat{\gamma}^{\tau})$ **then**
      $y_j^{\text{fixed}}(\hat{\gamma}^{\tau}) = y_j^{\text{fixed}}(\hat{\gamma}^{\tau}) + l_k$
    else if $k \in \mathcal{N}_j^{\text{upper}}(\hat{\gamma}^{\tau})$ **then**
      $y_j^{\text{fixed}}(\hat{\gamma}^{\tau}) = y_j^{\text{fixed}}(\hat{\gamma}^{\tau}) + u_k$
    else
      $A_j(\hat{\gamma}^{\tau}) = A_j(\hat{\gamma}^{\tau}) + 1/a_k$; $B_j(\hat{\gamma}^{\tau}) = B_j(\hat{\gamma}^{\tau}) + b_k/a_k$
    end if
  end for
end for
Compute $\sum_{j=1}^{m} y_j(\hat{\gamma}^{\tau})$ using Equation (5)
  if $\sum_{j=1}^{m} y_j(\hat{\gamma}^{\tau}) = R$ **then**
    $\lambda^* = \hat{\gamma}^{\tau}$; compute $\boldsymbol{x}(\lambda)$ as $\boldsymbol{x}(\hat{\gamma}^{\tau})$ using Equation (7)
    return
else if $\sum_{i=1}^{n} x_i(\hat{\gamma}^{\tau}) < R$ **then**
  $\mathcal{B}^{\tau+1} := \{\lambda \in \mathcal{B}^{\tau} | \lambda < \hat{\gamma}^{\tau}\}$
  Determine new bounds: $\lambda_{\downarrow}^{\tau+1} := \lambda_{\downarrow}^{\tau}$; $\lambda_{\uparrow}^{\tau+1} := \hat{\gamma}^{\tau}$
else
  $\mathcal{B}^{\tau+1} := \{\lambda \in \mathcal{B}^{\tau} | \lambda \geq \hat{\gamma}^{\tau}\}$
  Determine new bounds: $\lambda_{\downarrow}^{\tau+1} := \hat{\gamma}^{\tau}$; $\lambda_{\uparrow}^{\tau+1} := \lambda_{\uparrow}^{\tau}$
end if
for $j \in \mathcal{M}$ do {Update sets of variables whose state has not been determined yet}

$\mathcal{I}_j^{\tau+1} := \mathcal{I}_j^{\tau}$; $Y_j^{\tau+1} := Y_j^{\tau}$; $\bar{A}_j^{\tau+1} := \bar{A}_j^{\tau}$; $\bar{B}_j^{\tau+1} := \bar{B}_j^{\tau}$
for $k \in \mathcal{I}_j^{\tau}$ **do**
if $\beta_k \leq \lambda_{\downarrow}^{\tau+1}$ **then**
Remove k from $\mathcal{I}_j^{\tau+1}$; $Y_j^{\tau+1} = Y_j^{\tau+1} + l_k$
else if $\alpha_i \leq \lambda_{\downarrow}^{\tau+1} \leq \lambda_{\uparrow}^{\tau+1} \leq \beta_i$ then
Remove k from $\mathcal{I}_j^{\tau+1}$; $\bar{A}_j^{\tau+1} = \bar{A}_j^{\tau+1} + 1/a_k$; $\bar{B}_j^{\tau+1} = \bar{B}_j^{\tau+1} + b_k/a_k$
else if $\lambda_{\uparrow}^{\tau+1} \leq \alpha_k$ **then**
Remove k from $\mathcal{I}_j^{\tau+1}$; $Y_j^{\tau+1} = Y_j^{\tau+1} + u_k$
end if
end for
end for
$\tau = \tau + 1$
end while
Determine γ as the single element of $\tilde{\mathcal{B}}$
Compute λ^* using Equation (12) and $\boldsymbol{x}(\lambda^*)$
using Equation (7)
return

We establish the worst-case time complexity of Algorithm 3 by means of Lemma 7 and Theorem 2. First, Lemma 7 states that the binary search procedure can be carried out in $O(n)$ time.

Lemma 7. *The binary breakpoint search procedure in Lines 8–46 of Algorithm 3 has a time complexity of $O(n)$.*

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

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

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

1. First, we consider the computation of the candidate multiplier $\hat{\gamma}^{\tau}$ in Line 9. The median of an unsorted set of breakpoints $\mathcal{B}^{\tau}$ can be computed in $O(|\mathcal{B}^{\tau}|)$ time using, for example, the median-of-medians algorithm of Blum et al. (1973). This means that instead of sorting the initial breakpoint set $\mathcal{B}$ in $O(n \log n)$ time and retrieving median elements in $O(1)$ time, we can compute each candidate multiplier $\hat{\gamma}^{\tau}$ in $O(|\mathcal{B}^{\tau}|)$ time.

2. Second, 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 $\sum_{j=1}^{m} y_j(\hat{\gamma}^{\tau})$ from $O(n)$ to $O\left(\sum_{j=1}^{m} |\mathcal{I}_j^{\tau}|\right)$. In contrast, 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 33–44 takes $O\left(\sum_{j=1}^{m} |\mathcal{I}_j^{\tau}|\right)$ time.

Thus, the time complexity of the τ^{th} iteration of the binary search loop is $O(|\mathcal{B}^{\tau}| + \sum_{j=1}^{m} |\mathcal{I}_j^{\tau}|)$. 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 $\mathcal{B}^{\tau}$. This implies that $\sum_{j=1}^{m} |\mathcal{I}_j^{\tau}| \leq |\mathcal{B}^{\tau}|$. It follows that the time complexity of the τ^{th} iteration of the binary search loop reduces to $O(|\mathcal{B}^{\tau}|)$. □

Using this lemma, we establish the time complexity of Algorithm 3:

Theorem 2. *Algorithm 3 has a time complexity of $O(n \log n)$.*

Proof. Analogously to Theorem 1, all operations other than the binary search procedure in Lines 8–46 take $O(n \log n)$ time. Because the binary search procedure takes $O(n)$ time by Lemma 7, the overall time complexity of Algorithm 3 is $O(n \log n)$. □

Although the binary search strategy of Algorithm 3 has a better time complexity than the sequential search strategy of Algorithm 2, the linear-time algorithm in Blum et al. (1973) is known to be relatively slow in practice (Alexandrescu 2017). Thus, an alternative method for computing medians with a time complexity worse than $O(n)$ might be preferable in practice. We come back to this point in Section 6.2.

Finally, we comment on the role of the breakpoint search strategies within Algorithms 2 and 3. As stated in the proof of Theorem 2, Algorithms 2 and 3 differ only in the way the optimal multiplier λ^* is found after the breakpoints have been computed by means of Algorithm 1. Thus, the former procedure could technically be replaced by any other breakpoint search algorithm, as long as it can use, for example, the breakpoints to eventually output the optimal Lagrange multiplier. We are not aware of any breakpoint search strategy that is not a variant of either the sequential or binary search strategy. However, if such a potential alternative breakpoint search algorithm depends only on evaluation of the primal variables for given candidate multipliers, then such an algorithm could be embedded by using Equations (7) and (8) to do these evaluations. The question is then whether the algorithm can be adapted so that the overall time complexity is at most $O(n \log n)$. Recall that the time complexity of a single evaluation without using bookkeeping parameters is $O(n)$ and that at most $2n$ evaluations are required (one for each breakpoint). Thus, the overall time complexity can be reduced to $O(n \log n)$ only if either at most $O(\log n)$ evaluations are required (as in the binary strategy) or the amortized complexity of performing an evaluation can be reduced to $O(\log n)$ using, for example, bookkeeping parameters (as in the sequential strategy).

5.3. Complexity Results for Special Cases and Related Problems

In this section, we use Algorithms 2 and 3 and the complexity results in Theorems 1 and 2 to state complexity results for several special cases of Problem QRAP-NonSep-GBC and related problems. Some of these cases are of interest for the problem of scheduling 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, that is, $|\mathcal{N}_j| = C$ for some natural number C. For this case, we can show that, given C, the time complexity of Algorithm 3 is linear. This special case includes the problem of scheduling three-phase electric vehicle charging that we introduced in Section 2.2 (see also Table 1) as we have $C = 3$ in this case.

Theorem 3. *If $n_j = C$ for all $j \in \mathcal{M}$ and $C \in \mathbb{N}$, the time complexity of Algorithm 3 is $O(n \log C)$.*

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 QRAP-NonSep-GBC. However, when $n_j = C$, the complexity analysis can be refined to $O(\sum_{j=1}^{m} n_j \log n_j) = O(\sum_{j=1}^{m} C \log C) = O(mC \log C) = O(n \log C)$. It follows that the time complexity of Algorithm 3 for this special case is $O(n \log C)$. □

Next, we focus on the special case where $w_j = 0$ for all $j \in \mathcal{M}$, that is, the quadratic *separable* resource allocation problem with generalized bound constraints. 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. This case can be solved in $O(n)$ time.

Theorem 4. *If $w_j = 0$ for all $j \in \mathcal{M}$, Problem QRAP-NonSep-GBC can be solved in O(n) time.*

Proof. After elimination of the generalized bound constraints (1c) according to the constraint simplification procedure described in Section 3.2, the remaining problem is a quadratic separable resource allocation problem because $w_j = 0$ for each $j \in \mathcal{M}$. Thus, we can solve this problem in $O(n)$ time, which implies that we can solve the whole Problem QRAP-NonSep-GBC in $O(n)$ time. □

Alternatively, this result can be obtained as a special case of the separable quadratic programming problem studied by Megiddo and Tamir (1993).

Subsequently, we consider the integer version of Problem QRAP-NonSep-GBC, that is, 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. *If $w_j = 0$ for all $j \in \mathcal{M}$, the integer version of Problem QRAP-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 $R \in \mathbb{Z}$. All steps and statements in the proof of Lemma 2 are valid for the integer version of Problem QRAP-NonSep-GBC because $\bar{\epsilon} > 1$, and we can choose $\epsilon = 1$ to obtain feasible solutions $\boldsymbol{x}'$ and $(\underline{\boldsymbol{x}}')^j$. Thus, to solve this version, we are required to solve the $2m$ subproblems $\text{QRAP}^j(L_j)$ and $\text{QRAP}^j(U_j)$ and one instance of the quadratic simple resource allocation problem with n variables (see also the proof of Theorem 4) as integer resource allocation problems. Because quadratic

simple resource allocation problems with integer variables can be solved in linear time. (sections 4.6 and 4.7 in Ibaraki and Katoh 1988), we can solve these $2m-1$ problems in $O(n)$ time. □

If $w_j \geq 0$ for all $j \in \mathcal{M}$, the integer version can be solved in $O(n^2)$ time (Moriguchi et al. 2011). Finally, we conjecture that the integer version of the general Problem QRAP-NonSep-GBC, that is, instances that satisfy Property 1, is solvable in strongly polynomial time but leave this as an open question for future research.

Finally, we consider instances of Problem QRAP-NonSep-GBC where we replace the objective function by a strictly convex function $F(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ that is *permutation-invariant*. This means that given a vector $\boldsymbol{x} \in \mathbb{R}^n$ and a permutation π of the index set $\mathcal{N}$, we have $F(\boldsymbol{x}) = F(\boldsymbol{y})$ where $\boldsymbol{y} := (x_{\pi[1]}, \ldots, x_{\pi[n]})$. Thus, informally, the value of $F(\boldsymbol{x})$ does not depend on the order of the elements of $\boldsymbol{x}$. Minimizing these functions over generalized bound constraints is equivalent to solving a particular instance of the original Problem QRAP-NonSep-GBC with $w_j = 0$ for all $j \in \mathcal{M}$ and $a_i = 2$ and $b_i = 0$ for all $i \in \mathcal{N}$, that is, whose objective function is $\sum_{i \in \mathcal{N}} x_i^2$. This follows from the observations that generalized bound constraints are a special case of submodular constraints (Fujishige 1984) and that the unique minimizer of a strictly convex permutation-invariant function over submodular constraints is the minimum-norm point of the feasible region given by these constraints, that is, the vector with minimal two-norm (Nagano and Aihara 2012). Together, this implies that the unique minimizer of any strictly convex permutation-invariant function over given generalized bound constraints also minimizes the two-norm over this region and thereby is the unique minimizer of the mentioned instance of Problem QRAP-NonSep-GBC.

6. Evaluation

In this section, we evaluate the two algorithms presented in Sections 5.1 and 5.2. We carry out two types of evaluation. First, we evaluate the performance of our algorithms on realistic instances of the EV charging problem EV-3Phase that we introduced in Section 2.2. 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 C of variables associated with a given constraint. Because for Problem QRAP-NonSep-GBC no other tailored algorithms are available, we compare the performance of our algorithms to that of the commercial solver MOSEK (MOSEK-ApS 2019).

In Section 6.1, we describe in more detail the problem instances that we use in the evaluations. Subsequently, in Section 6.2, we discuss several implementation details. Finally, in Section 6.3, we present and discuss the evaluation results.

6.1. Problem Instances

We carry out two types of evaluations. First, we evaluate the performance of our algorithms on instances of Problem EV-3Phase. For this, we consider a setting wherein an EV is empty and available for residential charging from 1800 hours and must be fully charged by 0800 hours the next day. This charging horizon of 14 hours is divided into 15-minute time intervals, meaning that $m = 56$. For the power consumption constraints of the EV, we follow the balancing framework of Weckx and Driesen (2015) and use the Tesla model 3 as a reference EV (Electric Vehicle Database 2020). This means that we choose $R = 4 \times 40{,}000 = 160{,}000$ Wh, $L_j = 0$ W, and $U_j = 11{,}500$ W for each $j \in \mathcal{M}$, and $l_i = -\frac{11{,}500}{3}$ W and $u_i = \frac{11{,}500}{3}$ 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 this, we use real power consumption measurement data of 40 households for 100 consecutive days that were obtained in the field test described in Hoogsteen et al. (2017). More precisely, we assign each power consumption profiles to one of the three phases with equal probability and, for a given day, choose each parameter $q_{j,p}$ as the sum of the power consumption during interval j of all households that have been assigned to phase p. Thus, for each of the 100 days one overall power profile $(q_{j,p})_{j \in \mathcal{M}, p \in \{1,2,3\}}$ is created. 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 $(W_1, W_2) \in \{(1,1), (1,100), (100,1)\}$. The choice (1,1) represents no preference for one of the objectives, and the choices (1,100) and (100,1) represent a preference for minimizing load unbalance and peak shaving, respectively.

Second, we assess the scalability of our algorithms. For this, we focus on the case where the subset sizes n_j are equal to some positive integer C. We generate random instances for a number of fixed values of C and m. Table 3 shows these fixed values of C and m and for each problem parameter the uniform distribution from which the parameter values are drawn. For each combination of C and m, we generate 10 instances according to the

Table 3. Parameter Choices for the Scalability Evaluation

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

given distributions. The distribution of each weight w_j is chosen such that the resulting problem instances satisfy Property 1, which ensures by Lemma 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 (1c) is redundant. As a consequence, all of these constraints need to be replaced according to the constraint simplification procedure in Section 3.2. Thereby, we maximize the time that Algorithms 2 and 3 require for this step and thus improve the fairness of the comparison with MOSEK.

6.2. Implementation Details

We implemented our algorithms in Python (version 3.5) to integrate them into DEMKIT, an existing simulation tool for DEM research (Hoogsteen et al. 2019). For solving the subproblems $\text{QRAP}^j(L_j)$ and $\text{QRAP}^j(U_j)$ in Line 2 of both Algorithms 2 and 3, we implement a sophisticated version of the sequential breakpoint search algorithm of Helgason et al. (1980) that allows us to solve both subproblems simultaneously and thereby is approximately twice as fast as the original sequential breakpoint search algorithm. Preliminary testing has shown that this algorithm is in general faster than the linear-time algorithms in Kiwiel (2007), despite its worse time complexity of $O(n_j \log n_j)$. Furthermore, in Algorithm 3, we compute the median of a breakpoint set in the same way as in Algorithm 2, namely by sorting the original breakpoint set and retrieving the desired breakpoints in $O(1)$ time (see also Section 5.1). The reason for this is that linear-time algorithms for median finding such as that of Blum et al. (1973) are in general slower than alternative sampling-or sorting-based approaches (Alexandrescu 2017).

In both algorithms, we could reduce the time complexity of sorting all breakpoints from $O(n \log n)$ to $O(n \log m)$ using a multiway merging algorithm (Knuth 1998) 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 multiway merging algorithm to simplify the implementation of the algorithms without significantly increasing the overall execution time.

6.3. Results

In this section, we present and discuss the results of the evaluation as described in Section 6.1. All simulations and computations are executed on a 2.60-GHz Dell Inspiron 15 with an Intel Core i7-6700HQ CPU and 16 GB of RAM.

First, we focus on the performance of our algorithms on the instances of Problem EV-3Phase. Table 4 shows the average execution times of our algorithms and MOSEK over all considered 100 days for each combination of

Table 4. Average Execution Times (s) of Algorithms 2 and 3 and MOSEK for Each Combination of Weights

Weight combination	Algorithm 2	Algorithm 3	MOSEK
(1, 1)	4.64×10^{-3}	4.89×10^{-3}	2.33×10^{-2}
(1, 100)	4.69×10^{-3}	4.94×10^{-3}	2.11×10^{-2}
(100, 1)	4.71×10^{-3}	4.86×10^{-3}	2.07×10^{-2}

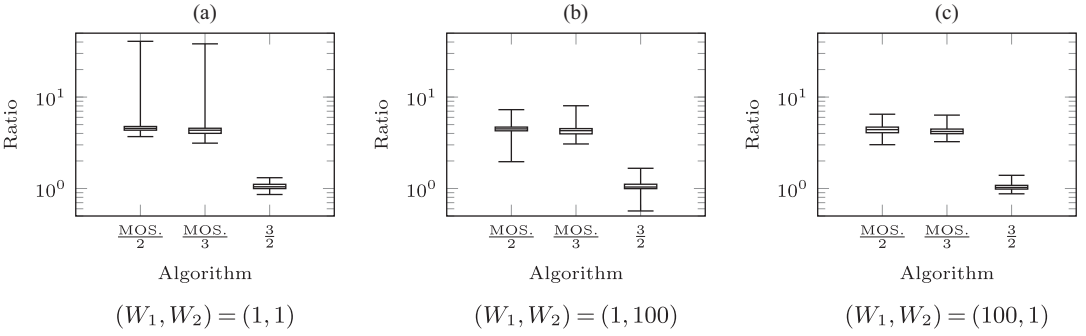
weights. Moreover, Figure 5 shows for each combination of weights the boxplots of the ratios between the execution times of each combination of algorithms. The results in Figure 5 indicate that our algorithms solve realistic instances of Problem EV-3Phase four to five times as fast as MOSEK for each weight combination. Moreover, Algorithm 3 appears to be slightly faster than Algorithm 2, although the difference in their execution times is less than 4% of the execution time of Algorithm 3 for most problem instances. The results in both Table 4 and Figure 5 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 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 (Deshpande et al. 2011). Thus, our algorithms will most likely not be the (computational) bottleneck in DEM systems and are therefore suitable for integration in such systems.

Second, we discuss the results of the scalability evaluation. For this, we first compare the performance of the two different breakpoint search approaches, since this is the only aspect in which Algorithms 2 and 3 are different. To this end, we show in Figure 6 for each combination of C and m the boxplot of ratios between the execution times of the sequential breakpoint search in Algorithm 2 and of the binary breakpoint search in Algorithm 3, that is, the execution time of the breakpoint search procedure of Algorithm 2 divided by that of Algorithm 3. Moreover, Figure 7 shows for each combination of C and m the boxplot of ratios between the overall execution times of Algorithms 2 and 3. The results in Figure 6 indicate that for $C \leq 10$ the ratios regarding the breakpoint search procedures decrease significantly as m increases. For these values of C, most of these ratios are greater than one when $m \leq 5$ and smaller than one 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 $C > 10$, the relation between the ratios and m is less clear. However, for each of these values of C, most of the ratios are greater than one 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 2 and 3 are less than an order of magnitude. Because 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 7, that is, 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 6. For example, for $C \leq 10$, most ratios are larger than 1 when $m \leq 5$ and smaller than 1 when $m \geq 10$, whereas for $C > 10$ most ratios are greater than 1. This suggests that Algorithm 3 is in general faster than Algorithm 2 unless $C \leq 10$ and $m \leq 5$.

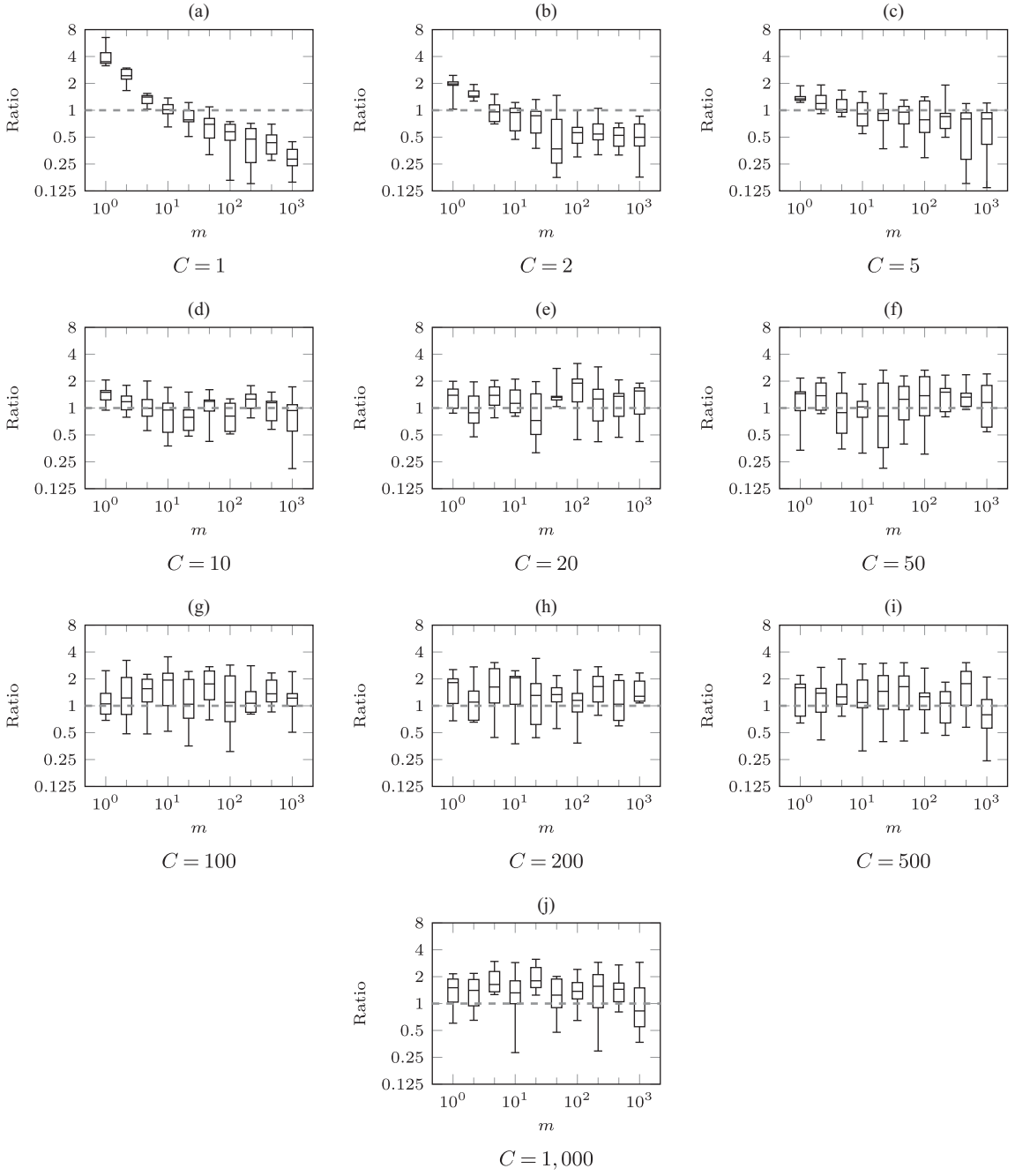
Finally, we compare the performance of our algorithms to MOSEK. To this end, Figure 8 shows for each combination of C and m the execution time of Algorithm 3 and MOSEK on each problem instance. We do not plot the execution times of Algorithm 2 in this figure, because the differences in execution time between Algorithms 2 and 3 are so small that plotting them together in the same figure would unnecessarily obscure the results. Furthermore, Table 5 shows the fitted power laws for Algorithms 2 and 3, that is, for each C, we fit the function $f(m) = c_1 \cdot m^{c_2}$ to the execution times corresponding to C. For $C = 1{,}000$, MOSEK was not able to solve any of the instances for $m = 500$ and $m = 1{,}000$ because of out-of-memory errors. Finally, to provide additional insight into

Figure 5. Ratios of Execution Times Between MOSEK and Algorithm 2 $\left(\frac{\text{MOS.}}{2}\right)$, MOSEK and Algorithm 3 $\left(\frac{\text{MOS.}}{3}\right)$, and Algorithms 3 and 2 $\left(\frac{3}{2}\right)$ for Each Combination of Weights



Notes. (a) $(W_1, W_2) = (1, 1)$. (b) $(W_1, W_2) = (1, 100)$. (c) $(W_1, W_2) = (100, 1)$.

Figure 6. Ratios Between Execution Time of the Breakpoint Search Procedures of Algorithms 2 and 3 (Execution Time of the Breakpoint Search Procedure of Algorithm 2 Divided by That of Algorithm 3)

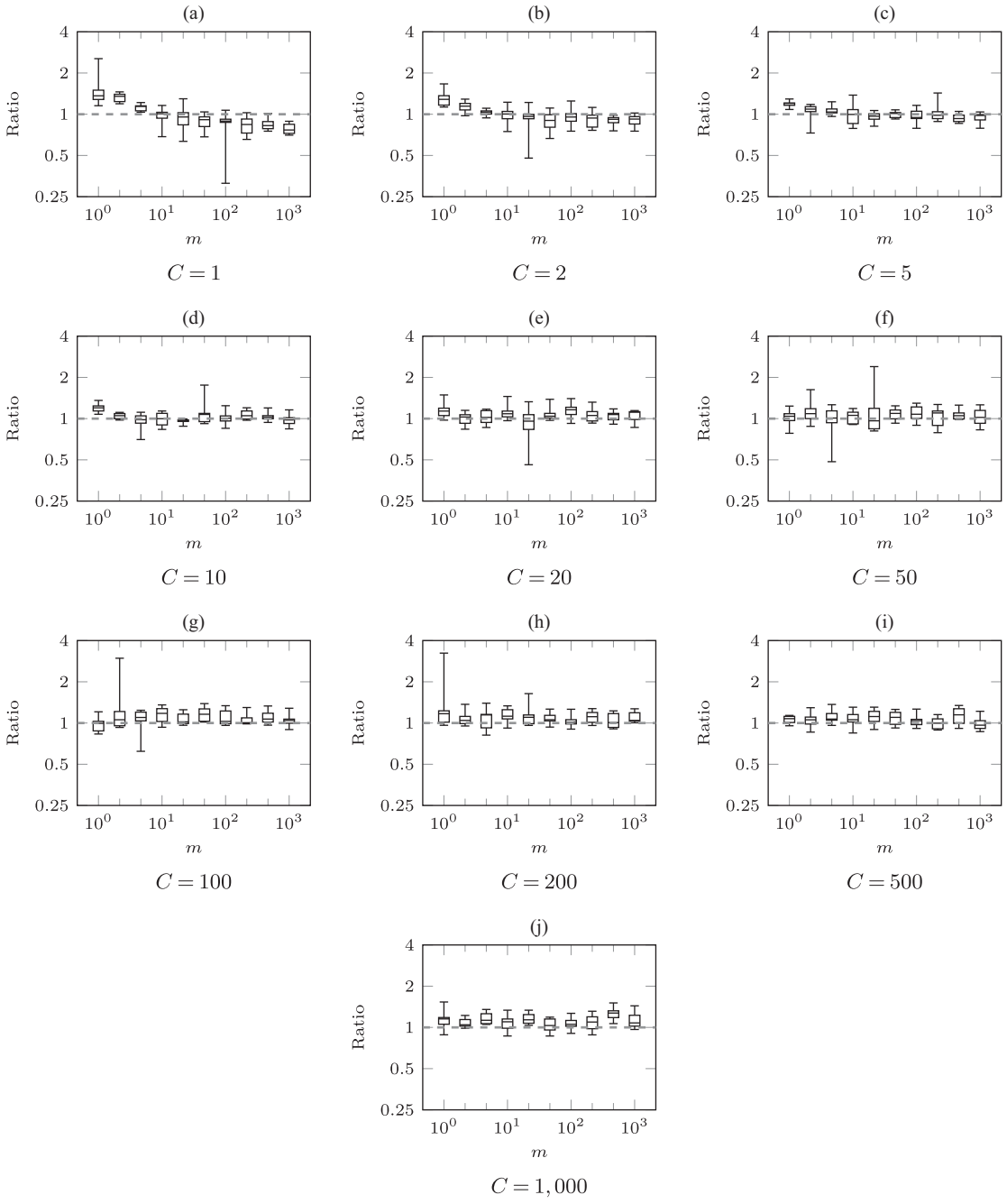


Notes. (a) $C = 1$. (b) $C = 2$. (c) $C = 5$. (d) $C = 10$. (e) $C = 20$. (f) $C = 50$. (g) $C = 100$. (h) $C = 200$. (i) $C = 500$. (j) $C = 1{,}000$.

the reported execution times, Tables C.1–C.3 show for each combination of C and m the average execution time of Algorithms 2 and 3 and MOSEK, respectively.

The power laws in Figure 8 and Table 5 suggest that the execution time of Algorithms 2 and 3 grows linearly as m increases, that is, the exponents c_2 in Table 5 are close to one. This observation is consistent with the theoretical worst-case complexity of Algorithm 3, which is $O(n \log C) = O(mC \log C)$ and demonstrates its practical scalability. On the other hand, the execution time of MOSEK does not seem to behave polynomially. Given the execution times of MOSEK for $C \leq 50$ in Figure 8, (a)–(f), this is most likely because of the initialization time of MOSEK, which for smaller problem instances is relatively large compared with the actual time required by the

Figure 7. Ratios Between Execution Times of Algorithms 2 and 3 (Execution Time of Algorithm 2 Divided by That of Algorithm 3)

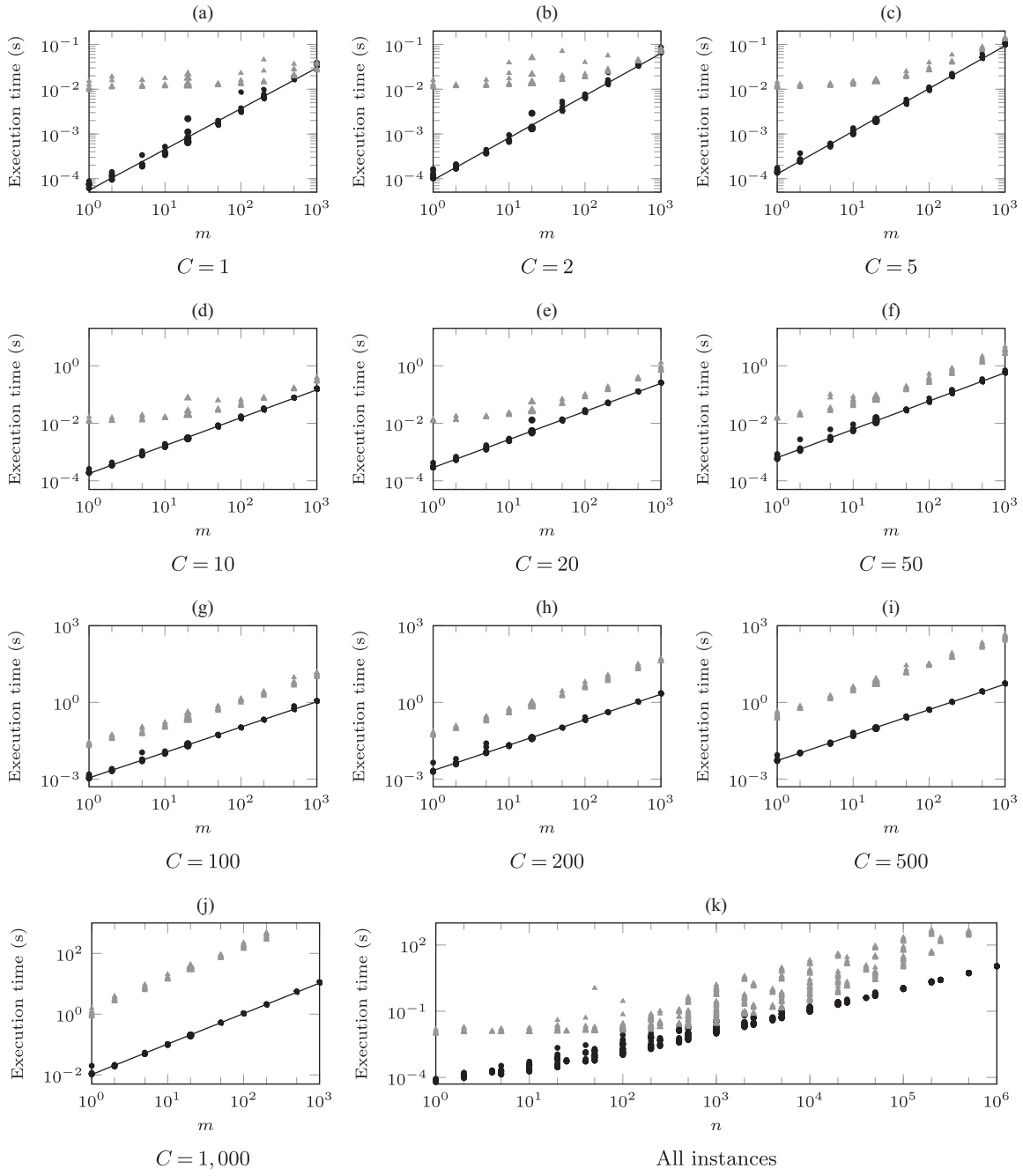


Notes. (a) $C = 1$. (b) $C = 2$. (c) $C = 5$. (d) $C = 10$. (e) $C = 20$. (f) $C = 50$. (g) $C = 100$. (h) $C = 200$. (i) $C = 500$. (j) $C = 1{,}000$.

internal solver to solve the corresponding instance. As a consequence, Algorithms 2 and 3 are at least one order of magnitude faster for instances with $C \leq 50$ and $m \leq 10$.

For $C \geq 100$, the results in Figure 8 and Tables C.1–C.3 indicate that Algorithms 2 and 3 are at least one order of magnitude faster than MOSEK regardless of m. In fact, for $C = 1{,}000$, both our algorithms are even two orders of magnitude faster. In this case, our algorithms solve all instances with $m = 500$ and $m = 1{,}000$ in less than 16 seconds (Algorithm 2) and 12 seconds (Algorithm 3), whereas MOSEK was not able to compute a solution because of out-of-memory errors.

Figure 8. Execution Times of Algorithm 3 (Circles, Black) and MOSEK (Triangles, Gray)



Notes. (a) $C = 1$. (b) $C = 2$. (c) $C = 5$. (d) $C = 10$. (e) $C = 20$. (f) $C = 50$. (g) $C = 100$. (h) $C = 200$. (i) $C = 500$. (j) $C = 1{,}000$. (k) All instances.

7. Concluding Remarks

In this article, we studied a quadratic nonseparable resource allocation problem with generalized bound constraints. This problem was motivated by its application in decentralized energy management and in particular for scheduling 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 one million variables in less than 16 seconds on a personal computer and are up to

Table 5. Power Law Regression Functions for Algorithms 2 and 3 for Each C (Fitted Functions $c_1 \times m^{c_2}$)

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

100 times faster than a standard commercial solver. This practical efficiency of our algorithms makes them suitable for the aforementioned electric vehicle scheduling problems because these problems have to be solved on embedded systems with low computational power and low memory.

This work adds a new problem to the class of quadratic nonseparable resource allocation problems that can be solved in strongly polynomial time by efficient algorithms. The question remains how this class can be extended further. Existing work on optimization under submodular constraints (Hochbaum and Hong 1995, Moriguchi et al. 2011) suggests that the class of nonseparable resource allocation problems where both the constraints and nonseparability are induced by a so-called *laminar family* 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 (Tamir 1993, Hochbaum and Hong 1995), scaling algorithms (Moriguchi et al. 2011), and monotonicity-based optimization (Vidal et al. 2019 and this article).

Regarding the application of decentralized energy management, these algorithms can be used to solve local optimization problems of devices that are equipped with three-phase chargers other than EVs. In particular, the derivation of an algorithm for the convex (quadratic) nonseparable resource allocation with nested constraints is an interesting direction for future research because 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. Therefore, this is a relevant and important direction of future research that can contribute greatly to a sustainable future energy supply.

Acknowledgments

The authors thank the anonymous referees for helpful comments and suggestions for improving this article. In particular, the authors thank one reviewer for pointing out the relation of our work to the nonseparable objective studied by Nagano and Aihara (2012).

Appendix A. Formulation of Problem EV-3Phase

In this appendix, we derive the expression in Equation (2) for the objective of minimizing load unbalance and show that Problem EV-3Phase is an instance of Problem QRAP-NonSep-GBC.

First, as a measure for load unbalance during a given interval $j \in \mathcal{M}$, we use the squared two-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 2. 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 $\sum_{j=1}^{m} \|P_j^{\text{res}}\|^2$, where $\|\cdot\|$ denotes the two-norm on $\mathbb{R}^2$. We can assume without loss of generality that the phases are arranged as depicted in Figure 2. 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

$$
\begin{aligned}
\|P_j^{\text{res}}\|^2 &= \left(\sum_{p=1}^{3}(q_{j,p}+z_{j,p})\cos\phi_p\right)^2 + \left(\sum_{p=1}^{3}(q_{j,p}+z_{j,p})\sin\phi_p\right)^2,\\
&= \sum_{p=1}^{3}(q_{j,p}+z_{j,p})^2(\cos^2\phi_n+\sin^2\phi_n),\\
&\quad +2\sum_{p=1}^{3}\sum_{p'=p+1}^{3}(q_{j,p}+z_{j,p})(q_{j,p'}+z_{j,p'})(\cos\phi_p\cos\phi_{p'}+\sin\phi_p\sin\phi_{p'}),\\
&= \sum_{p=1}^{3}(q_{j,p}+z_{j,p})^2 + 2(q_{j,1}+z_{j,1})(q_{j,2}+z_{j,2})\left(-\frac{3}{4}+\frac{1}{4}\right),\\
&\quad +2(q_{j,1}+z_{j,1})(q_{j,3}+z_{j,3})\left(0-\frac{1}{2}\right)+2(q_{j,2}+z_{j,2})(q_{j,3}+z_{j,3})\left(0-\frac{1}{2}\right),\\
&= \sum_{p=1}^{3}(q_{j,p}+z_{j,p})^2 - (q_{j,1}+z_{j,1})(q_{j,2}+z_{j,2}) - (q_{j,1}+z_{j,1})(q_{j,3}+z_{j,3}) - (q_{j,2}+z_{j,2})(q_{j,3}+z_{j,3}),\\
&= \frac{3}{2}\sum_{p=1}^{3}(q_{j,p}+z_{j,p})^2 - \frac{1}{2}\left(\sum_{p=1}^{3}(q_{j,p}+z_{j,p})\right)^2.
\end{aligned}
$$

Second, to show that Problem EV-3Phase is an instance of Problem QRAP-NonSep-GBC, observe that the objective function of Problem EV-3Phase can be rewritten to

$$
\begin{aligned}
&W_1\sum_{j=1}^{m}\left(\sum_{p=1}^{3}(q_{j,p}+z_{j,p})\right)^2 + W_2\sum_{j=1}^{m}\left(\frac{3}{2}\sum_{p=1}^{3}(q_{j,p}+z_{j,p})^2 - \frac{1}{2}\left(\sum_{p=1}^{3}(q_{j,p}+z_{j,p})\right)^2\right),\\
&= \left(W_1-\frac{1}{2}W_2\right)\sum_{j=1}^{m}\left(\sum_{p=1}^{3}(q_{j,p}+z_{j,p})\right)^2 + \frac{3}{2}W_2\sum_{j=1}^{m}\sum_{p=1}^{3}(q_{j,p}+z_{j,p})^2,\\
&= \left(W_1-\frac{1}{2}W_2\right)\sum_{j=1}^{m}\left(\sum_{p=1}^{3}z_{j,p}\right)^2 + \frac{3}{2}W_2\sum_{j=1}^{m}\sum_{p=1}^{3}z_{j,p}^2 + \left(W_1-\frac{1}{2}W_2\right)\sum_{j=1}^{m}\left(\sum_{p=1}^{3}q_{j,p}\right)\sum_{p=1}^{3}z_{j,p},\\
&\quad +\frac{3}{2}W_2\sum_{j=1}^{m}\sum_{p=1}^{3}q_{j,p}z_{j,p} + \left(W_1-\frac{1}{2}W_2\right)\sum_{j=1}^{m}\left(\sum_{p=1}^{3}q_{j,p}\right)^2 + \frac{3}{2}W_2\sum_{j=1}^{m}\sum_{p=1}^{3}q_{j,p}^2.
\end{aligned}
$$

The latter expression corresponds directly with the values in Table 1.

Appendix B. Proofs of Lemmas 1, 2, and 4–6

B.1. Proof of Lemma 1

Suppose that H^j is positive definite. Then its determinant is strictly positive. Because of the special structure of H^j, we can rewrite its determinant to the following form by applying the matrix determinant lemma (Harville 1997):

$$
\det(H^j) = \det(w_j\boldsymbol{e}\boldsymbol{e}^\top + \operatorname{diag}(\boldsymbol{a}^j)) = \left(1 + w_j\sum_{i\in\mathcal{N}_j}\frac{1}{a_i}\right)\det(\operatorname{diag}(\boldsymbol{a}^j)).
$$

Because $a_i > 0$ for all $i \in \mathcal{N}_j$, we have that $\det(\operatorname{diag}(\boldsymbol{a}^j)) > 0$ and thus we also have that $1 + w_j\sum_{i\in\mathcal{N}_j}1/a_i > 0$.

Now suppose that $1 + w_j\sum_{i\in\mathcal{N}_j}1/a_i > 0$. We show that all leading principal minors of H^j are positive, that is, that the determinant of each upper-left submatrix of H^j is positive. For this, we label the indices of $\mathcal{N}_j$ as $1,\ldots,n_j$ such that, for any $1 \le \ell \le n_j$, 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 $H^j_{1:\ell;1:\ell}$.

To show that $\det(H^j_{1:\ell;1:\ell}) > 0$, we compute this determinant by applying the matrix determinant lemma to $H^j_{1:\ell;1:\ell}$. This yields

$$
\det(H^j_{1:\ell;1:\ell}) = \left(1 + w_j\sum_{i=1}^{\ell}\frac{1}{a_i}\right)\prod_{i=1}^{\ell}a_i.
$$

Note that $1+w_j\sum_{i=1}^{\ell}1/a_i>0$ because $1+w_j\sum_{i\in\mathcal{N}_j}1/a_i>0$ and all values a_i are positive. It follows that $\det(H^j_{1:\ell;1:\ell})>0$. Because ℓ was chosen arbitrarily, this implies that all leading principal minors of H^j are positive and thus that H^j is positive definite. □

B.2. Proof of Lemma 2

We prove the validity of the lower bounds $\underline{x}_i \le x^*_i$; the proof for the upper bounds $x^*_i \le \bar{x}_i$ is analogous. If for a given $j \in \mathcal{M}$ there is no optimal solution x^* to Problem QRAP-NonSep-GBC that satisfies the bounds $\underline{x}_i \le x^*_i \le \bar{x}_i$ for each $i \in \mathcal{N}_j$, then choose out of all these solutions the one solution $\boldsymbol{x}^*$ 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 $k \in \mathcal{N}_j\backslash\{i\}$ such that $x^*_k > \underline{x}_k$, because 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 $\boldsymbol{x}'$ 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 QRAP-NonSep-GBC because $x'_i = x^*_i + \epsilon \le x^*_i + \bar{\epsilon} \le x^*_i + \underline{x}_i - x^*_i = \underline{x}_i$, $x'_k = x^*_k - \epsilon \ge x^*_k - \bar{\epsilon} \ge x^*_k - x^*_k + \underline{x}_k = \underline{x}_k$, and $\underline{\boldsymbol{x}}^j$ and $\boldsymbol{x}^*$ are feasible for Problem QRAP$^j(L_j)$ and Problem QRAP-NonSep-GBC, respectively. Moreover, because $\boldsymbol{x}^*$ is an optimal solution to Problem QRAP-NonSep-GBC, we have that

$$\sum_{j'=1}^{m}\frac{1}{2}w_{j'}\left(\sum_{\ell\in\mathcal{N}_{j'}}x^*_i\right)^2 + \sum_{\ell=1}^{n}\left(\frac{1}{2}a_\ell(x^*_\ell)^2 + b_\ell x^*_\ell\right) \le \sum_{j'=1}^{m}\frac{1}{2}w_{j'}\left(\sum_{\ell\in\mathcal{N}_{j'}}x'_i\right)^2 + \sum_{\ell=1}^{n}\left(\frac{1}{2}a_\ell(x'_\ell)^2 + b_\ell x'_\ell\right).$$

It follows by definition of $\boldsymbol{x}'$ that

$$\begin{aligned} 0 &\le \frac{1}{2}a_i(x'_i)^2 + b_i x'_i + \frac{1}{2}a_k(x'_k)^2 + b_k x'_k - \frac{1}{2}a_i(x^*_i)^2 - b_i x^*_i - \frac{1}{2}a_k(x^*_k)^2 - b_k x^*_k \\ &= \frac{1}{2}a_i(x^*_i+\epsilon)^2 + b_i(x^*_i+\epsilon) + \frac{1}{2}a_k(x^*_k-\epsilon)^2 + b_k(x^*_k-\epsilon) - \frac{1}{2}a_i(x^*_i)^2 - b_i x^*_i - \frac{1}{2}a_k(x^*_k)^2 - b_k x^*_k \\ &= a_i x^*_i\epsilon + \frac{1}{2}a_i\epsilon^2 + b_i\epsilon - a_k x^*_k\epsilon + \frac{1}{2}a_k\epsilon^2 + b_k\epsilon. \end{aligned} \tag{B.1}$$

Analogously, the solution $(\underline{\boldsymbol{x}}')^j$ 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 QRAP$^j(L_j)$ because $\underline{x}'_i = \underline{x}_i - \epsilon \ge \underline{x}_i - \bar{\epsilon} \ge \underline{x}_i - \underline{x}_i + x^*_i = x^*_i$, $\underline{x}'_k = \underline{x}_k + \epsilon \le \underline{x}_k + \bar{\epsilon} \le \underline{x}_k + x^*_k - \underline{x}_k = x^*_k$, and x^* and $\underline{\boldsymbol{x}}^j$ are feasible for Problem QRAP-NonSep-GBC and Problem QRAP$^j(L_j$, respectively. Moreover, because $\underline{\boldsymbol{x}}^j$ is optimal for Problem QRAP$^j(L_j)$, we have that

$$\sum_{\ell\in\mathcal{N}_j}\left(\frac{1}{2}a_\ell(\underline{x}_\ell)^2 + b_\ell\underline{x}_\ell\right) \le \sum_{\ell\in\mathcal{N}_j}\left(\frac{1}{2}a_\ell(\underline{x}'_\ell)^2 + b_\ell\underline{x}'_\ell\right).$$

It follows by definition of $(\underline{\boldsymbol{x}}')^j$ that

$$\begin{aligned} 0 &\le \frac{1}{2}a_i(\underline{x}'_i)^2 + b_i\underline{x}'_i + \frac{1}{2}a_k(\underline{x}'_k)^2 + b_k\underline{x}'_k - \frac{1}{2}a_i(\underline{x}_i)^2 - b_i\underline{x}_i - \frac{1}{2}a_k(\underline{x}_k)^2 - b_k\underline{x}_k \\ &= \frac{1}{2}a_i(\underline{x}_i-\epsilon)^2 + b_i(\underline{x}_i-\epsilon) + \frac{1}{2}a_k(\underline{x}_k+\epsilon)^2 + b_k(\underline{x}_k+\epsilon) - \frac{1}{2}a_i(\underline{x}_i)^2 - b_i\underline{x}_i - \frac{1}{2}a_k(\underline{x}_k)^2 - b_k\underline{x}_k \\ &= -a_i\underline{x}_i\epsilon + \frac{1}{2}a_i\epsilon^2 - b_i\epsilon + a_k\underline{x}_k\epsilon + \frac{1}{2}a_k\epsilon^2 + b_k\epsilon. \end{aligned} \tag{B.2}$$

Adding Equations (B.1) and (B.2) yields

$$0 \le a_i\epsilon(x^*_i - \underline{x}_i + \epsilon) + a_k\epsilon(\underline{x}_k - x^*_k + \epsilon) \le a_i\epsilon(-\bar{\epsilon} + \epsilon) + a_k\epsilon(-\bar{\epsilon} + \epsilon) \le 0.$$

This implies that both Equations (B.1) and (B.2) are equalities and thus that $\boldsymbol{x}'$ and $(\underline{\boldsymbol{x}}')^j$ are optimal for Problem QRAP-NonSep-GBC and QRAP$^j(L_j)$ respectively. However, because $x'_i \le \underline{x}_i$ and $x'_k \ge \underline{x}_k$, it holds that

$$\begin{aligned} \sum_{\ell\in\mathcal{N}_j}\max(\underline{x}_\ell - x'_\ell, 0) &= d - \max(\underline{x}_i - x^*_i, 0) - \max(\underline{x}_k - x^*_k, 0) + \max(\underline{x}_i - x'_i, 0) + \max(\underline{x}_k - x'_k, 0), \\ &= d - \underline{x}_i + x^*_i - 0 + \underline{x}_i - x'_i + 0, \end{aligned}$$

$$= d + x_i^* - x_i',$$
$$= d - \epsilon.$$

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 $\underline{\boldsymbol{x}}$. □

B.3. Proof of Lemma 4

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 $k \in \mathcal{N}_j \backslash \{i\}$ such that $x_k(\lambda_1) \geq x_k(\lambda_2)$. Subsequently, we show that the existence of such an index k leads to a contradiction.

For each $\ell \in \mathcal{N}_j$, we divide KKT Condition (3a) by a_ℓ:

$$\frac{w_j y_j}{a_\ell} + x_\ell + \frac{b_\ell + \lambda + \mu_\ell}{a_\ell} = 0, \quad \ell \in \mathcal{N}_j. \tag{B.3}$$

By summing Equation (B.3) over $\mathcal{N}_j$, we obtain

$$0 = w_j y_j \sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell} + \sum_{\ell \in \mathcal{N}_j} \left(x_\ell + \frac{b_\ell + \lambda + \mu_\ell}{a_\ell} \right) = \left(1 + w_j \sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell} \right) y_j + \sum_{\ell \in \mathcal{N}_j} \frac{b_\ell + \lambda + \mu_\ell}{a_\ell}. \tag{B.4}$$

Suppose that there is no index $k \in \mathcal{N}_j \backslash \{i\}$ such that $x_k(\lambda_1) \geq x_k(\lambda_2)$. 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 (B.4), Property 1, and Lemma 3 that

$$\begin{aligned} 0 &= \left(1 + w_j \sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell} \right) (y_j(\lambda_2) - y_j(\lambda_1)) + \sum_{\ell \in \mathcal{N}_j} \frac{b_\ell - b_\ell + \lambda_2 - \lambda_1 + \mu_\ell(\lambda_2) - \mu_\ell(\lambda_1)}{a_\ell}, \\ &> \sum_{\ell \in \mathcal{N}_j} \frac{b_\ell - b_\ell + \lambda_2 - \lambda_1 + \mu_\ell(\lambda_2) - \mu_\ell(\lambda_1)}{a_\ell}, \\ &> \sum_{\ell \in \mathcal{N}_j} \frac{\mu_\ell(\lambda_2) - \mu_\ell(\lambda_1)}{a_\ell} \geq 0. \end{aligned}$$

This is a contradiction; hence, there must exist an index $k \in \mathcal{N}_j \backslash \{i\}$ with $x_k(\lambda_1) > x_k(\lambda_2)$.

We now show that the existence of the index k leads to a contradiction. By Lemma 3, we have $\mu_k(\lambda_1) \geq \mu_k(\lambda_2)$. It follows that

$$a_k x_k(\lambda_2) + b_k + \mu_k(\lambda_2) \leq a_k x_k(\lambda_1) + b_k + \mu_k(\lambda_1). \tag{B.5}$$

However, KKT Condition (3a) implies that

$$w_j y_j(\lambda_1) + a_i x_i(\lambda_1) + b_i + \mu_i(\lambda_1) = -\lambda_1 = w_j y_j(\lambda_1) + a_k x_k(\lambda_1) + b_k + \mu_k(\lambda_1),$$

which yields

$$a_i x_i(\lambda_1) + b_i + \mu_i(\lambda_1) = a_k x_k(\lambda_1) + b_k + \mu_k(\lambda_1). \tag{B.6}$$

Analogously, we have

$$a_i x_i(\lambda_2) + b_i + \mu_i(\lambda_2) = a_k x_k(\lambda_2) + b_k + \mu_k(\lambda_2). \tag{B.7}$$

It follows from Equations (B.5)–(B.7) and Lemma 3 that

$$\begin{aligned} a_k x_k(\lambda_2) + b_k + \mu_k(\lambda_2) &\leq a_k x_k(\lambda_1) + b_k + \mu_k(\lambda_1), \\ &= a_i x_i(\lambda_1) + b_i + \mu_i(\lambda_1), \\ &< a_i x_i(\lambda_2) + b_i + \mu_i(\lambda_2), \\ &= a_k x_k(\lambda_2) + b_k + \mu_k(\lambda_2). \end{aligned}$$

This is a contradiction; hence, it must be that $x_i(\lambda_1) \geq x_i(\lambda_2)$. As this implies that $x_i(\lambda_1) \geq x_i(\lambda_2)$ for all $\lambda_1 < \lambda_2$ and $i \in \mathcal{N}$, the lemma is proven. □

B.4. Proof of Lemma 5

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 (4b) and KKT Conditions (3d) and (3e). It follows from KKT Condition (3a) that

$$w_j y_j(\alpha_i) + a_\ell x_\ell(\alpha_i) + b_\ell + \alpha_i + \mu_\ell(\alpha_i) = 0, \quad \ell \in \mathcal{N}_j, \tag{B.8a}$$
$$w_j y_j(\alpha_i + \epsilon) + a_\ell x_\ell(\alpha_i + \epsilon) + b_\ell + \alpha_i + \epsilon + \mu_\ell(\alpha_i + \epsilon) = 0, \quad \ell \in \mathcal{N}_j. \tag{B.8b}$$

To show that $\mu_i(\alpha_i) = 0$, we show that $\mu_i(\alpha_i) \le \epsilon$ if $w_j \ge 0$ and $\mu_i(\alpha_i) \le \epsilon \sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell}$ if $w_j < 0$. Because ϵ was chosen arbitrarily, $\sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell} > 0$, and $\mu_i(\alpha_i) \ge 0$ by definition of α_i, this implies in both cases that $\mu_i(\alpha_i) = 0$.

First, if $w_j \ge 0$, then $w_j y_j(\cdot)$ is nonincreasing by Corollary 1. Together with Lemma 4 and the fact that $\mu_i(\alpha_i + \epsilon) = 0$, subtracting Equation (B.8b) from Equation (B.8a) for $\ell = i$ yields

$$\begin{aligned} 0 &= w_j y_j(\alpha_i) - w_j y_j(\alpha_i + \epsilon) + a_i x_i(\alpha_i) - a_i x_i(\alpha_i + \epsilon) - \epsilon + \mu_i(\alpha_i) - \mu_i(\alpha_i + \epsilon), \\ &\ge -\epsilon + \mu_i(\alpha_i). \end{aligned}$$

It follows that $\mu_i(\alpha_i) \le \epsilon$.

Second, if $w_j < 0$, then we can apply the same proof mechanism as was used in the proof of Lemma 4 (see also Equations (B.3) and (B.4)). By dividing Equations (B.8a) and (B.8b) by a_ℓ and summing them over the index set $\mathcal{N}_j$, we get the following together with Property 1 and Corollary 2:

$$\begin{aligned} 0 &= \sum_{\ell \in \mathcal{N}_j} \left(\frac{w_j y_j(\alpha_i)}{a_\ell} - \frac{w_j y_j(\alpha_i + \epsilon)}{a_\ell} + x_\ell(\alpha_i) - x_\ell(\alpha_i + \epsilon) - \frac{\epsilon}{a_\ell} + \frac{\mu_\ell(\alpha_i)}{a_\ell} - \frac{\mu_\ell(\alpha_i + \epsilon)}{a_\ell} \right), \\ &= \left(1 + w_j \sum_{\ell \in \mathcal{N}} \frac{1}{a_\ell} \right) (y_j(\alpha_i) - y_j(\alpha_i + \epsilon)) - \epsilon \sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell} + \sum_{\ell \in \mathcal{N}_j} \frac{\mu_\ell(\alpha_i) - \mu_\ell(\alpha_i + \epsilon)}{a_\ell}, \\ &\ge -\epsilon \sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell} + \sum_{\ell \in \mathcal{N}_j} \frac{\mu_\ell(\alpha_i) - \mu_\ell(\alpha_i + \epsilon)}{a_\ell}, \\ &\ge -\epsilon \sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell} + \mu_i(\alpha_i). \end{aligned}$$

Here, the first inequality follows from Property 1 and Lemma 4 and the second equality follows from Corollary 2 and the fact that $\mu_i(\alpha_i + \epsilon) = 0$. It follows that $\mu_i(\alpha_i) \le \epsilon \sum_{\ell \in \mathcal{N}_j} \frac{1}{a_\ell}$. □

B.5. Proof of Lemma 6

We prove the lemma for the case $a_i u_i + b_i > a_k u_k + b_k$; the proof for the case $a_i l_i + b_i > a_k l_k + b_k$ is analogous. We show that $x_i(\alpha_k) < u_i$, which implies by definition of α_i that $x_i(\alpha_k) < u_i = x_i(\alpha_i)$. Using Lemma 4, this yields $\alpha_i \le \alpha_k$.

It follows from KKT Condition (3a) that

$$w_j y_j(\alpha_k) + a_k x_k(\alpha_k) + b_k + \mu_k(\alpha_k) = -\alpha_k = w_j y_j(\alpha_k) + a_i x_i(\alpha_k) + b_i + \mu_i(\alpha_k).$$

Because $\mu_k(\alpha_k) = 0$ by Lemma 5 and $x_k(\alpha_k) = u_k$ by definition of α_k, this is equivalent to

$$a_k u_k + b_k = a_i x_i(\alpha_k) + b_i + \mu_i(\alpha_k). \tag{B.9}$$

Suppose that $x_i(\alpha_k) = u_i$. Then $\mu_i(\alpha_k) \ge 0$ by KKT Condition (3d). It follows from Equation (B.9) that

$$a_i x_i(\alpha_k) + b_i = a_k u_k + b_k - \mu_i(\alpha_k) < a_i u_i b_i = a_i x_k(\alpha_k),$$

which is a contradiction. Thus, it must hold that $x_i(\alpha_k) < u_i$. □

Appendix C. Average Execution Times of Algorithms 2 and 3 and MOSEK

Table C1. Average Execution Times (s) of Algorithm 2 for Each Combination of C and m

$C \setminus m$	1	2	5	10	20	50	100	200	500	1,000
1	1.07×10^{-4}	1.49×10^{-4}	2.38×10^{-4}	3.68×10^{-4}	8.87×10^{-4}	1.51×10^{-3}	2.94×10^{-3}	5.75×10^{-3}	1.47×10^{-2}	2.75×10^{-2}
2	1.58×10^{-4}	2.06×10^{-4}	3.93×10^{-4}	7.23×10^{-4}	1.34×10^{-3}	3.29×10^{-3}	6.49×10^{-3}	1.34×10^{-2}	3.13×10^{-2}	6.29×10^{-2}
5	1.75×10^{-4}	2.77×10^{-4}	5.97×10^{-4}	1.08×10^{-3}	1.92×10^{-3}	4.96×10^{-3}	9.91×10^{-3}	2.09×10^{-2}	4.78×10^{-2}	9.89×10^{-2}
10	2.40×10^{-4}	3.72×10^{-4}	8.04×10^{-4}	1.61×10^{-3}	2.89×10^{-3}	8.73×10^{-3}	1.60×10^{-2}	3.22×10^{-2}	7.95×10^{-2}	1.53×10^{-1}
20	3.63×10^{-4}	5.61×10^{-4}	1.41×10^{-3}	2.88×10^{-3}	5.29×10^{-3}	1.37×10^{-2}	2.94×10^{-2}	5.40×10^{-2}	1.33×10^{-1}	2.72×10^{-1}
50	6.63×10^{-4}	1.54×10^{-3}	3.09×10^{-3}	6.56×10^{-3}	1.36×10^{-2}	3.09×10^{-2}	6.35×10^{-2}	1.29×10^{-1}	3.10×10^{-1}	6.10×10^{-1}
100	1.16×10^{-3}	2.84×10^{-3}	6.28×10^{-3}	1.22×10^{-2}	2.33×10^{-2}	6.12×10^{-2}	1.14×10^{-1}	2.20×10^{-1}	6.29×10^{-1}	$1.18 \times 10^{+0}$
200	3.45×10^{-3}	4.64×10^{-3}	1.38×10^{-2}	2.37×10^{-2}	4.66×10^{-2}	1.10×10^{-1}	2.15×10^{-1}	4.56×10^{-1}	$1.11 \times 10^{+0}$	$2.45 \times 10^{+0}$
500	6.08×10^{-3}	1.13×10^{-2}	2.87×10^{-2}	5.78×10^{-2}	1.11×10^{-1}	2.83×10^{-1}	5.29×10^{-1}	$1.04 \times 10^{+0}$	$3.01 \times 10^{+0}$	$5.43 \times 10^{+0}$
1,000	1.36×10^{-2}	2.17×10^{-2}	5.84×10^{-2}	1.09×10^{-1}	2.34×10^{-1}	5.43×10^{-1}	$1.12 \times 10^{+0}$	$2.28 \times 10^{+0}$	$6.89 \times 10^{+0}$	$1.27 \times 10^{+1}$

Table C2. Average Execution Times (s) of Algorithm 3 for Each Combination of C and m

$C \setminus m$	1	2	5	10	20	50	100	200	500	1,000
1	7.31×10^{-5}	1.11×10^{-4}	2.11×10^{-4}	3.77×10^{-4}	8.84×10^{-4}	1.69×10^{-3}	3.76×10^{-3}	6.90×10^{-3}	1.73×10^{-2}	3.50×10^{-2}
2	1.20×10^{-4}	1.80×10^{-4}	3.79×10^{-4}	7.23×10^{-4}	1.49×10^{-3}	3.69×10^{-3}	6.74×10^{-3}	1.47×10^{-2}	3.48×10^{-2}	6.93×10^{-2}
5	1.47×10^{-4}	2.63×10^{-4}	5.59×10^{-4}	1.08×10^{-3}	1.98×10^{-3}	4.95×10^{-3}	1.01×10^{-2}	2.01×10^{-2}	5.07×10^{-2}	1.03×10^{-1}
10	1.98×10^{-4}	3.55×10^{-4}	8.26×10^{-4}	1.61×10^{-3}	2.99×10^{-3}	7.82×10^{-3}	1.56×10^{-2}	3.01×10^{-2}	7.63×10^{-2}	1.55×10^{-1}
20	3.10×10^{-4}	5.57×10^{-4}	1.35×10^{-3}	2.56×10^{-3}	5.94×10^{-3}	1.27×10^{-2}	2.52×10^{-2}	5.00×10^{-2}	1.26×10^{-1}	2.55×10^{-1}
50	6.44×10^{-4}	1.34×10^{-3}	3.20×10^{-3}	6.23×10^{-3}	1.20×10^{-2}	2.85×10^{-2}	5.74×10^{-2}	1.23×10^{-1}	2.89×10^{-1}	5.84×10^{-1}
100	1.17×10^{-3}	2.17×10^{-3}	6.03×10^{-3}	1.05×10^{-2}	2.14×10^{-2}	5.19×10^{-2}	1.02×10^{-1}	2.06×10^{-1}	5.62×10^{-1}	$1.12 \times 10^{+0}$
200	2.24×10^{-3}	4.23×10^{-3}	1.29×10^{-2}	2.04×10^{-2}	4.03×10^{-2}	1.00×10^{-1}	2.07×10^{-1}	4.05×10^{-1}	$1.05 \times 10^{+0}$	$2.22 \times 10^{+0}$
500	5.69×10^{-3}	1.05×10^{-2}	2.53×10^{-2}	5.26×10^{-2}	9.80×10^{-2}	2.55×10^{-1}	5.06×10^{-1}	$1.02 \times 10^{+0}$	$2.63 \times 10^{+0}$	$5.42 \times 10^{+0}$
1,000	1.17×10^{-2}	2.00×10^{-2}	4.98×10^{-2}	1.00×10^{-1}	2.00×10^{-1}	5.14×10^{-1}	$1.04 \times 10^{+0}$	$2.07 \times 10^{+0}$	$5.39 \times 10^{+0}$	$1.10 \times 10^{+1}$

Table C3. Average Execution Times (s) of MOSEK for Each Combination of C and m

$C \setminus m$	1	2	5	10	20	50	100	200	500	1,000
1	1.09×10^{-2}	1.26×10^{-2}	1.26×10^{-2}	1.26×10^{-2}	1.45×10^{-2}	1.26×10^{-2}	4.19×10^{-2}	1.89×10^{-2}	2.33×10^{-2}	2.95×10^{-2}
2	1.21×10^{-2}	1.20×10^{-2}	1.26×10^{-2}	1.70×10^{-2}	1.80×10^{-2}	2.29×10^{-2}	2.13×10^{-2}	2.87×10^{-2}	4.29×10^{-2}	7.25×10^{-2}
5	1.21×10^{-2}	1.20×10^{-2}	1.25×10^{-2}	1.24×10^{-2}	1.53×10^{-2}	2.00×10^{-2}	2.83×10^{-2}	4.23×10^{-2}	7.94×10^{-2}	1.36×10^{-1}
10	1.23×10^{-2}	1.27×10^{-2}	1.42×10^{-2}	1.57×10^{-2}	2.61×10^{-2}	3.55×10^{-2}	4.60×10^{-2}	7.39×10^{-2}	1.57×10^{-1}	3.15×10^{-1}
20	1.27×10^{-2}	1.38×10^{-2}	1.68×10^{-2}	2.09×10^{-2}	3.31×10^{-2}	5.26×10^{-2}	8.72×10^{-2}	1.54×10^{-1}	3.65×10^{-1}	8.12×10^{-1}
50	1.52×10^{-2}	1.98×10^{-2}	4.20×10^{-2}	5.41×10^{-2}	7.62×10^{-2}	1.61×10^{-1}	3.27×10^{-1}	6.35×10^{-1}	$1.55 \times 10^{+0}$	$3.27 \times 10^{+0}$
100	2.34×10^{-2}	4.32×10^{-2}	7.07×10^{-2}	1.20×10^{-1}	2.54×10^{-1}	5.83×10^{-1}	$1.21 \times 10^{+0}$	$2.15 \times 10^{+0}$	$5.85 \times 10^{+0}$	$1.20 \times 10^{+1}$
200	5.70×10^{-2}	1.00×10^{-1}	2.21×10^{-1}	4.42×10^{-1}	8.32×10^{-1}	$2.03 \times 10^{+0}$	$4.21 \times 10^{+0}$	$8.73 \times 10^{+0}$	$2.40 \times 10^{+1}$	$4.41 \times 10^{+1}$
500	3.08×10^{-1}	5.92×10^{-1}	$1.55 \times 10^{+0}$	$3.30 \times 10^{+0}$	$5.98 \times 10^{+0}$	$1.63 \times 10^{+1}$	$2.98 \times 10^{+1}$	$6.20 \times 10^{+1}$	$1.60 \times 10^{+2}$	$3.51 \times 10^{+2}$
1,000	$1.06 \times 10^{+0}$	$3.05 \times 10^{+0}$	$7.37 \times 10^{+0}$	$1.54 \times 10^{+1}$	$3.41 \times 10^{+1}$	$7.84 \times 10^{+1}$	$1.71 \times 10^{+2}$	$3.75 \times 10^{+2}$	—	—

References

Alexandrescu A (2017) Fast deterministic selection. Raman CSI, Pissis SP, Puglisi SJ, Rajeev, eds. *Leibniz Internat. Proc. in Informatics*, vol. 75, (Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany), 24:1–24:9.

Bakker H, Dunke F, Nickel S (2020) A structuring review on multi-stage optimization under uncertainty: Aligning concepts from theory and practice. *Omega* 96:102080.

Bartlett MS (1951) An inverse matrix adjustment arising in discriminant analysis. *Ann. Math. Statist.* 22(1):107–111.

Beaudin M, Zareipour H (2015) Home energy management systems: A review of modelling and complexity. *Renewable Sustainable Energy Rev.* 45:318–335.

Blum M, Floyd RW, Pratt V, Rivest RL, Tarjan RE (1973) Time bounds for selection. *J. Comput. Systems Sci.* 7(4):448–461.

Boyd S, Vandenberghe L (2004) *Convex Optimization*, 7th ed. (Cambridge University Press, Cambridge, UK).

Bretthauer KM, Shetty B (1997) Quadratic resource allocation with generalized upper bounds. *Oper. Res. Lett.* 20(2):51–57.

Bretthauer KM, Shetty B (2002) A pegging algorithm for the nonlinear resource allocation problem. *Comput. Oper. Res.* 29(5):505–527.

Cominetti R, Mascarenhas WF, Silva PJS (2014) A Newton's method for the continuous quadratic knapsack problem. *Math. Programming Comput.* 6(2):151–169.

De Waegenaere A, Wielhouwer JL (2012) A breakpoint search approach for convex resource allocation problems with bounded variables. *Optim. Lett.* 6(4):629–640.

Deshpande JG, Kim E, Thottan M (2011) Differentiated services QoS in smart grid communication networks. *Bell Labs Tech. J.* 16(3):61–81.

Electric Vehicle Database (2020) Tesla model 3 standard range price and specifications: EV database. Accessed March 4, 2020, https://ev-database.org/car/1060/Tesla-Model-3-Standard-Range.

Engle R, Kelly B (2012) Dynamic equicorrelation. *J. Bus. Econom. Statist.* 30(2):212–228.

Esther BP, Kumar KS (2016) A survey on residential demand side management architecture, approaches, optimization models and methods. *Renewable Sustainable Energy Rev.* 59:342–351.

Fujishige S (1984) Structures of polyhedra determined by submodular functions on crossing families. *Math. Programming* 29(2):125–141.

Gan L, Topcu U, Low SH (2013) Optimal decentralized protocol for electric vehicle charging. *IEEE Trans. Power Systems* 28(2):940–951.

Gerards MET, Toersche HA, Hoogsteen G, van der Klauw T, Hurink JL, Smit GJM (2015) Demand side management using profile steering. *Proc. IEEE Eindhoven PowerTech* (IEEE, Piscataway, NJ), 457759:1–457759:6.

Gondzio J (2012) Interior point methods 25 years later. *Eur. J. Oper. Res.* 218(3):587–601.

Harville DA (1997) *Matrix Algebra from a Statistician's Perspective*, 1st ed. (Springer Verlag, New York).

He P, Zhao L, Zhou S, Niu Z (2013) Water-filling: A geometric approach and its application to aolve generalized radio resource allocation problems. *IEEE Trans. Wireless Comm.* 12(7):3637–3647.

Helgason R, Kennington J, Lall H (1980) A polynomially bounded algorithm for a singly constrained quadratic program. *Math. Programming* 18(1):338–343.

Hochbaum DS (1994) Lower and upper bounds for the allocation problem and other nonlinear optimization problems. *Math. Oper. Res.* 19(2):390–409.

Hochbaum DS, Hong SP (1995) About strongly polynomial time algorithms for quadratic optimization over submodular constraints. *Math. Programming* 69:269–309.
Hoogsteen G, Hurink JL, Smit GJM (2019) DEMKit: A decentralized energy management simulation and demonstration toolkit. *Proc. IEEE PES Innovative Smart Grid Technologies Europe* (IEEE, Piscataway, NJ), 1–5.
Hoogsteen G, Molderink A, Hurink JL, Smit GJ, Kootstra B, Schuring F (2017) Charging electric vehicles, baking pizzas, and melting a fuse in Lochem. *CIRED Open Access Proc. J.* 2017(1):1629–1633.
Ibaraki T, Katoh N (1988) *Resource Allocation Problems: Algorithmic Approaches*, 1st ed. (MIT Press, Cambridge, MA).
Katoh N, Shioura A, Ibaraki T (2013) Resource allocation problems. Pardalos PM, Du DZ, Graham RL, eds. *Handbook of Combinatorial Optimization* (Springer, New York), 2897–2988.
Kiwiel KC (2007) Breakpoint searching algorithms for the continuous quadratic knapsack problem. *Math. Programming* 112(2):473–491.
Kiwiel KC (2008) Variable fixing algorithms for the continuous quadratic knapsack problem. *J. Optim. Theory Appl.* 136(3):445–458.
Knuth DE (1998) *Sorting and searching. The Art of Computer Programming*, vol. 3, 2nd ed. (Addison-Wesley, Reading, MA), 722.
Kuphaldt TR (2007) Polyphase AC circuits. *Lessons In Electric Circuits*, 6th ed., 283–346.
Ledoit O, Wolf M (2003) Improved estimation of the covariance matrix of stock returns with an application to portfolio selection. *J. Empirical Finance* 10(5):603–621.
Markowitz H (1952) Portfolio selection. *J. Finance* 7(1):77–91.
Megiddo N, Tamir A (1993) Linear time algorithms for some separable quadratic programming problems. *Oper. Res. Lett.* 13(4):203–211.
Moriguchi S, Shioura A, Tsuchimura N (2011) M-convex function minimization by continuous relaxation approach: proximity theorem and algorithm. *SIAM J. Optim.* 21(3):633–668.
MOSEK-ApS (2019) MOSEK optimizer API for Python, release 9.1.13. (MOSEK-ApS, Copenhagen, Denmark).
Mou Y, Xing H, Lin Z, Fu M (2015) Decentralized optimal demand-side management for PHEV charging in a smart grid. *IEEE Trans. Smart Grid* 6(2):726–736.
Nagano K, Aihara K (2012) Equivalence of convex minimization problems over base polytopes. *Japan J. Industrial Appl. Math.* 29(3):519–534.
Patriksson M (2008) A survey on the continuous nonlinear resource allocation problem. *Eur. J. Oper. Res.* 185(1):1–46.
Patriksson M, Strömberg C (2015) Algorithms for the continuous nonlinear resource allocation problem: New implementations and numerical studies. *Eur. J. Oper. Res.* 243(3):703–722.
Sahni S (1974) Computationally related problems. *SIAM J. Comput.* 3(4):262–279.
Schoot Uiterkamp MHH (2016) Robust planning of electric vehicle charging. MSc thesis, University of Twente, Enschede, Netherlands.
Schoot Uiterkamp MHH, Gerards MET, Hurink JL (2020) ODDO: Online duality-driven optimization. Preprint, submitted August 22, https://arxiv.org/abs/2008.09838.
Schoot Uiterkamp MHH, Gerards MET, Hurink JL (2022) On a reduction for a class of resource allocation problems. *INFORMS J. Comput.* Forthcoming.
Schoot Uiterkamp MHH, Hoogsteen G, Gerards MET, Hurink JL, Smit GJM (2017) Multi-commodity support in profile steering. *Proc. IEEE PES Innovative Smart Grid Technologies Conf. Europe* (IEEE, Piscataway, NJ), 1–6.
Schoot Uiterkamp MHH, Hurink JL, Gerards MET (2021) A fast algorithm for quadratic resource allocation problems with nested constraints. *Comput. Oper. Res.* 135:105451.
Shams F, Bacci G, Luise M (2014) A survey on resource allocation techniques in OFDM(A) networks. *Comput. Networks* 65:129–150.
Siano P (2014) Demand response and smart grids: A survey. *Renewable Sustainable Energy Rev.* 30:461–478.
Spedicato E (1975) A bound to the condition number of canonical rank-two corrections and applications to the variable metric method. *Calcolo* 12(2):185–199.
Stevenson WD (1975) *Elements of Power System Analysis*, 3rd ed. (McGraw Hill, New York).
Tamir A (1993) A strongly polynomial algorithm for minimum convex separable quadratic cost flow problems on two-terminal series-parallel networks. *Math. Programming* 59(1):117–132.
van der Klauw T, Gerards MET, Hurink JL (2017) Resource allocation problems in decentralized energy management. *OR Spectrum* 39(3):749–773.
Végh L (2016) A strongly polynomial algorithm for a class of minimum-cost flow problems with separable convex objectives. *SIAM J. Comput.* 45(5):1729–1761.
Vidal T, Gribel D, Jaillet P (2019) Separable convex optimization with nested lower and upper constraints. *INFORMS J. Optim.* 1(1):71–90.
Weckx S, Driesen J (2015) Load balancing with EV chargers and PV inverters in unbalanced distribution grids. *IEEE Trans. Sustainable Energy* 6(2):635–643.
Wright SE, Lim S (2020) Solving nested-constraint resource allocation problems with an interior point method. *Oper. Res. Lett.* 48(3):297–303.
Wright SE, Rohal JJ (2014) Solving the continuous nonlinear resource allocation problem with an interior point method. *Oper. Res. Lett.* 42(6):404–408.