

WILEY

Intl. Trans. in Op. Res. 33 (2026) 3128–3157
DOI: 10.1111/itor.70001INTERNATIONAL
TRANSACTIONS
IN OPERATIONAL
RESEARCH

A novel dual-decomposition method for non-convex two-stage stochastic mixed-integer quadratically constrained quadratic problems

Nikita Belyak and Fabricio Oliveira* *Department of Mathematics and Systems Analysis, Aalto University, Espoo, Finland*
E-mail: nikita.belyak@aalto.fi [Belyak]; fabricio.oliveira@aalto.fi [Oliveira]

Received 11 April 2024; received in revised form 24 October 2024; accepted 26 January 2025

Abstract

We propose the novel p -branch-and-bound method for solving two-stage stochastic programming problems whose deterministic equivalents are represented by non-convex mixed-integer quadratically constrained quadratic programming (MIQCQP) models. The precision of the solution generated by the p -branch-and-bound method can be arbitrarily adjusted by altering the value of the precision factor p . The proposed method combines two key techniques. The first one, named p -Lagrangian decomposition, generates a mixed-integer relaxation of a dual problem with a separable structure for a primal non-convex MIQCQP problem. The second one is a version of the classical dual decomposition approach that is applied to solve the Lagrangian dual problem and ensures that integrality and non-anticipativity conditions are met once the optimal solution is obtained. This paper also presents a comparative analysis of the p -branch-and-bound method efficiency considering two alternative solution methods for the dual problems as a subroutine. These are the proximal bundle method and Frank–Wolfe progressive hedging. The latter algorithm relies on the interpolation of linearisation steps similar to those taken in the Frank–Wolfe method as an inner loop in the classic progressive hedging. The p -branch-and-bound method's efficiency was tested on randomly generated instances and demonstrated superior performance over commercial solver Gurobi.

Keywords: two-stage stochastic programming; normalised multiparametric disaggregation; Lagrangian relaxation; branch-and-bound

1. Introduction

Presently, the majority of engineering sectors utilise mathematical optimisation as a modelling framework to represent the behaviour of various processes. Areas such as electrical and process engineering are arguably the most prominent employers of mathematical optimisation techniques

*Corresponding author.

© 2025 The Author(s).

International Transactions in Operational Research published by John Wiley & Sons Ltd on behalf of International Federation of Operational Research Societies.

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

to improve their operational performance. For instance, Andiappan (2017) emphasises the efficiency of mathematical optimisation as an approach to designing energy systems. The author highlights the superior performances of mathematical optimisation in terms of comprehensiveness and ability to explicitly determine the topology of energy systems compared to its alternatives, for example, heuristic and insight-based approaches. Grossmann and Harjunkoski (2019) discuss the importance of mathematical optimisation in chemical and petrochemical operations, allowing, in some cases, up to 30% in energy savings.

Mathematical optimisation approaches closely rely on solving to optimality a mathematical optimisation problem – the set of mathematical relationships representing the real-world problem (Kallrath, 2021). Kallrath (2021) classifies mathematical optimisation problems into four groups such as linear programming problems, mixed-integer linear programming (MILP) problems, non-linear programming problems, and mixed-integer nonlinear programming (MINLP) problems. Mixed-integer problems involve decision variables that can have both continuous and discrete domains. The linearity or nonlinearity of the problem refers to the type of constraints and objective function. Lee and Leyffer (2011) highlight that MINLP problems are particularly challenging due to the difficulties arising from solving over integer variables and non-linear functions. At the same time, both MILP and nonlinear programming problems are known to be NP-hard (Forrest et al., 1974; Murtagh and Saunders, 1995). Nevertheless, the range of applications of MINLP is noticeably diverse (Lee and Leyffer, 2011). It includes modelling block layout design problems with unequal areas (Castillo et al., 2005), structural flow sheet problems (Kocis and Grossmann, 1987) and finding the optimal design of water distribution networks (Bragalli et al., 2012) to mention only a few relevant applications.

In this paper, we focus on a subclass of MINLP problems that represent deterministic equivalents for two-stage stochastic mixed-integer programming (2SSMIP) problems. Such problems involve two decision-variable sets that are separated by an intermediate probabilistic event. These two distinct decision-variable sets represent decisions made at different stages, that is, before and after the intermediate probabilistic event occurred and its outcome is observed. The modelling of probabilistic events for the most part involves consideration of mutually exclusive and exhaustive alternatives (scenarios) and the definition of probabilities associated with them (Bush and Mosteller, 1953). Despite their vast applicability, 2SSMIP problems raise serious conceptual and computational challenges (Küçükyavuz and Sen, 2017). For instance, Liao et al. (2019) exploited a multi-step MINLP problem to optimise the recovery process for network and load during power system restoration. In Wang et al. (2021), a 2SSMIP model has been used to formulate a container slot allocation problem for a liner shipping service. The authors used the sample-average approximation to approximate the expected value function rendering a nonlinear integer programming model.

Our interest resides in 2SSMIP problems whose deterministic equivalent representations are non-convex mixed-integer quadratically constrained quadratic programming (MIQCQP) models. These models arise in several practically relevant applications, such as the pooling problem, which is a non-convex MIQCQP under the assumption of linearly blending qualities (Misener and Floudas, 2012) and as the equivalent, single-level reformulation of some bi-level optimisation problems (Ding et al., 2014; Virasjoki et al., 2020). Examples of pooling problem applications include the design of water networks (Jezowski, 2010), modelling refinery processes (Amos et al., 1997) and transportation systems (Calvo et al., 2004) as some relevant examples.

The formulation of a general 2SSMIP is

$$z^{\text{SMIP}} = \min_x \{c^\top x + Q(x) : x \in X\}, \quad (1)$$

where the vector $c \in \mathbb{R}^{n_x}$ is known, X is a mixed-integer linear set consisting of linear constraints and integrality restrictions on some components of x . The recourse function $Q : \mathbb{R}^{n_x} \rightarrow \mathbb{R}$ is the expected recourse value function

$$Q(x) = \mathbb{E} \left[\min_y \{f(y, \xi) : g(x, y, \xi) = 0, y \in Y(\xi)\} \right], \quad (2)$$

where, for any realisation of the random variable ξ , $f : \mathbb{R}^{n_y} \rightarrow \mathbb{R}$ is defined as

$$f(y, \xi) = q(\xi)^\top y + \sum_{(i,j) \in B_Q} Q(\xi)_{i,j} y_i y_j,$$

$g = [g_1, \dots, g_{|M|}]^\top$ where $g_m : \mathbb{R}^{n_x \times n_y} \rightarrow \mathbb{R}$, $\forall m \in \{1, \dots, |M|\} = M$, is defined as

$$g_m(x, y, \xi) = T(\xi)_m x + W(\xi)_m y + \sum_{(i,j) \in B_U} U(\xi)_{m,i,j} y_i y_j - h(\xi)_m,$$

and B_Q (B_U) comprise the index pairs (i, j) for which the entry $|Q_{i,j}| > 0$ ($|U_{i,j}| > 0$), implying the presence of the bi-linear terms $y_i y_j$; $Y(\xi)$ is a mixed-integer set containing both linear constraints and integrality requirements on some of the variables $y(\xi)$; and $\mathbb{E}[\cdot]$ denotes the expectation of $\cdot$ in terms of the random variable ξ . As it is standard practice in the stochastic programming literature, we assume that the random variable ξ is represented by a finite set S of realisations $\xi_1, \dots, \xi_{|S|}$, each with associated probability value $\pi_1, \dots, \pi_{|S|}$. In particular, each realisation ξ_s of ξ encodes the realisation observed for each of the random elements $(q(\xi_s), Q(\xi_s))$ and $(T(\xi_s)_m, W(\xi_s)_m, U(\xi_s)_{m,i,j}, h(\xi_s)_m)$, $\forall m \in M$. For the sake of notation compactness, we refer to these collections as (q^s, Q^s) and $(T_m^s, W_m^s, U_{m,i,j}^s, h_m^s)$, $\forall m \in M$, respectively.

Problem (1) can be then posed as the deterministic equivalent

$$\begin{aligned} z^{\text{SMIP}} = \min_{x,y} \quad & c^\top x + \sum_{s \in S} \pi^s \left(q^{s\top} y^s + \sum_{(i,j) \in B_Q} Q_{i,j}^s y_i^s y_j^s \right) \\ \text{subject to: } \quad & x \in X \\ & T_m^s x + W_m^s y^s + \sum_{(i,j) \in B_U} U_{m,i,j}^s y_i^s y_j^s = h_m^s, \quad \forall m \in M, \forall s \in S \\ & y^s \in Y^s, \quad \forall s \in S. \end{aligned} \quad (3)$$

Due to the challenging nature of the non-convex MIQCQP problems open source and commercial global solvers, such as Gurobi (Gurobi Optimization, 2020), Couenne (Belotti, 2023) or Baron (The Optimization Firm, 2019) still present performance issues when addressing large-scale instances. There have been several solution approaches developed for non-convex MIQCQP

problems, which can be categorised into three groups. The first one involves approximation of problem (3) with a continuous or mixed-integer relaxation (Cui et al., 2013; Andrade et al., 2019). Another group is formed by those employing variants of the branch-and-bound (BnB) method. In particular, typically for non-convex problems, spatial BnB is used, which involves convexification of non-convex terms as a sub-routine (Berthold et al., 2012; Ding et al., 2014; Castro, 2016b). The last group involves methods relying on the introduction of non-anticipativity conditions (NAC) and the decomposition of the problem into more tractable sub-problems.

The block-angular structure of problem (3) allows for formulating an almost decomposable equivalent problem by making explicit NAC of the first-stage variables x . The reformulated deterministic equivalent model (RDEM) with an almost separable structure can be represented as

$$\begin{aligned}
 z^{\text{SMIP}} = \min_{x, y} \quad & \sum_{s \in S} \pi^s \left(c^\top x^s + q^{s\top} y^s + \sum_{(i,j) \in B_Q} Q_{i,j}^s y_i^s y_j^s \right) \\
 \text{s.t.:} \quad & y^s \in Y^s, \forall s \in S \\
 & x^s \in X, \forall s \in S \\
 & T_m^s x^s + W_m^s y^s + \sum_{(i,j) \in B_U} U_{m,i,j}^s y_i^s y_j^s = h_m^s, \forall m \in M, \forall s \in S \\
 & x^s - \bar{x} = 0, \forall s \in S,
 \end{aligned} \tag{4}$$

where the constraint $x^s - \bar{x} = 0, \forall s \in S$, enforces non-anticipativity for the first-stage decisions and $\bar{x}$ is an auxiliary variable used to enforce non-anticipativity for all $x^s, \forall s \in S$. The RDEM problem (4) could be fully decomposed into $|S|$ non-convex MIQCQP problems if one could remove the set of linear constraints $x^s - \bar{x} = 0, \forall s \in S$, that relates variables from distinct sub-problems, a structure commonly known as complicating constraints.

To tackle the non-convex problem (4), Andrade et al. (2022) developed an algorithm named p -Lagrangian decomposition. The p -Lagrangian decomposition method involves exploiting Lagrangian relaxation for decomposing the primal problem (4) into $|S|$ independent sub-problems and employing the reformulated normalised multiparametric disaggregation technique (RNMDT) (Andrade et al., 2019) to construct mixed-integer-based relaxations. The necessity to formulate a mixed-integer-based relaxation arises from the non-convex nature of the primal problem (4), leading to a lack of monotonicity in the behaviour of its dual bound value. Therefore, relying solely on Lagrangian decomposition becomes insufficient in ensuring a valid lower (dual) bound for the primal problem (4). However, applying Lagrangian decomposition to the mixed-integer-based relaxation of the primal problem yields a valid lower (dual) bound. As a subroutine, the p -Lagrangian decomposition algorithm employs a dynamic precision-based method, ensuring the tightening of the relaxation bounds as the precision parameter value approaches $-\infty$, and a bundle method (BM) approach for updating the dual multipliers. Additionally, the decomposable structure of the Lagrangian dual problem is amenable to parallelisation, which can significantly enhance the computational performance.

As suggested by the numerical results in Andrade et al. (2022), the p -Lagrangian decomposition demonstrated superior performance compared to commercial solver Gurobi (Gurobi Optimization, 2020) when solving non-convex MIQCQP problems with the decomposable structure. Nevertheless, the p -Lagrangian decomposition algorithm has an important shortcoming related to the duality gap arising from the mixed-integer nature of the primal problem combined with the imprecision of

the RNMDT relaxation. It is worth highlighting that the convergence of a p -Lagrangian relaxation problem requires the precision parameter value to approach $-\infty$. Nevertheless, for most of the practical applications, convergence with a predetermined tolerance is sufficient.

Our primary contribution involves the introduction of a solution method named p -branch-and-bound (p -BnB) for non-convex MIQCQP problem (4). The p -BnB algorithm evolves from the advancement of p -Lagrangian relaxation, effectively mitigating the duality-gap issue and ensuring convergence to a global optimum. Similar to the performance observed with the p -Lagrangian method, the proposed p -BnB surpasses the performance of the commercial solver Gurobi (Gurobi Optimization, 2020). Importantly, our methodology marks the first instance of incorporating p -Lagrangian relaxation within the framework of a duality-based BnB approach. The p -BnB method is inspired by the decomposition method for two-stage stochastic integer programs proposed in Carøe and Schultz (1999). The technically challenging synchronisation of p -BnB and decomposition method proposed in Carøe and Schultz (1999) relies on the repeatedly solving p -Lagrangian relaxation of problem (4) by means of p -BnB and iteratively restricting the feasible region via the BnB framework whenever the solution of p -Lagrangian relaxation violates integrality or non-anticipativity conditions. Consequently, p -BnB provides the upper bound for problem (4) that can be made arbitrarily precise against the value of the Lagrangian relaxation bound by decreasing the value of precision factor p .

Our subsequent contribution involves the evaluation of the numerical efficiency of p -BnB on randomly generated instances considering two different solution methods for the dual problem stemming from p -Lagrangian relaxation. The first one is the Frank–Wolfe Progressive Hedging (FWPH) method, originally presented in Boland et al. (2018). The classic progressive hedging (Rockafellar and Wets, 1991) method is proved to converge to the solution of the deterministic equivalent of a two-stage stochastic programming if such a point exists and the problem is convex. However, there is no guarantee of convergence in case the primary problem is mixed-integer, hence non-convex. In contrast, the FWPH is an enhancement of the classic progressive hedging method with convergence guarantees even in the presence of mixed-integer variables. That is, FWPH guarantees convergence to the optimal dual value of p -Lagrangian relaxation. The other solution method for dual problems tested in p -BnB is the proximal BM (Oliveira and Sagastizábal, 2014; Kim and Dandurand, 2022). The proximal BM relies on the classic BM (Lemaréchal, 1974) but involves regularisation of the dual space search allowing for fewer iterations until convergence (Kim and Dandurand, 2022).

The first step of the proposed p -BnB method involves the construction of the mixed-integer relaxation of the primal non-convex RDEM problem (4) by means of employing the RNMDT technique described in Section 2.1 to relax all quadratic terms. Next, we apply a Lagrangian duality-based BnB method reviewed in Section 3. The method involves the composition of BnB strategies and dual decomposition. Subsequently, each branching tree node is represented by an assembly of decomposable dual sub-problems. To solve the dual sub-problems within the BnB search, we consider the FWPH method discussed in Section 2.3.2 and the proximal BM presented in Section 2.3.1. To the best of our knowledge, this is the first time the efficiency of the FWPH method has been assessed within a Lagrangian duality-based BnB framework. The proposed method was tested on randomly generated instances, and the results of the numerical experiments are presented in Section 4. Finally, in Section 5, we provide conclusions and directions for further research.

2. Technical background

In what follows, we present the technical elements that form our proposed method. In essence, p -BnB is formed by the combination of three main techniques, namely p -Lagrangian decomposition, of which RNMDT is a key concept, solution methods for dual Lagrangian problems (FWPH and proximal BM) and a BnB coordination algorithm.

2.1. Reformulated normalised multiparametric disaggregation technique

To ensure the validity of the dual (lower) bound value for the non-convex problem (4), we utilise arbitrarily precise mixed-integer linear relaxation of (4), which will then serve as the basis for deriving the corresponding dual problem.

The normalised multiparametric disaggregation technique (NMDT) is an efficient technique to relax quadratic terms in non-convex MIQCQP problems (Castro, 2016a). NMDT involves the discretisation of the domain of one variable in each bi-linear term. The discretisation procedure in NMDT closely resembles the piecewise McCormick envelopes approach (McCormick, 1976; Bergamini et al., 2005) as it involves splitting the variable domain into a number of uniform partitions. The accuracy of the NMDT approximation of the primary non-convex MIQCIP problem is directly related to the size of these partitions and can be made arbitrarily precise. However, improving the accuracy of NMDT approximation follows the increase in the number of continuous and binary variables. Therefore, there is a trade-off between the accuracy of the NMDT approximation and its computational tractability.

The enhancement of NMDT has been proposed by Andrade et al. (2019), and this new version was named reformulated NMDT (RNMDT). The authors suggested reducing the numerical base used for the discretised domain from 10 to 2 (or binary). Hence, the smallest interval used for discretising the domain would be 0.5 when $p = -1$ instead of 0.1 as in the case of base 10. This modification allowed for a significant reduction in the number of auxiliary binary variables required in NMDT relaxation. Additionally, the authors performed a series of reformulations leading to the elimination of a number of redundant constraints and variables. Therefore, RNMDT relaxation became more easily tractable compared to NMDT. For further details on the reformulation of NMDT relaxation please refer to Andrade et al. (2019). The RNMDT relaxation of the primal RDEM problem can be constructed by employing RNMDT to discretise the second-stage variables y_j^s in the primal RDEM. Following the notation in Andrade et al. (2019), formally, for a fixed value of the precision factor p , the mixed-integer relaxation RNMDT_p can be stated as

$$\begin{aligned} z^{\text{RNMDT}} &= \min_{x,y,w} \sum_{s \in S} \pi^s \left(c^\top x^s + q^{s\top} y^s + \sum_{(i,j) \in B_Q} Q_{i,j}^s w_{i,j}^s \right) \\ \text{s.t.:} \quad & y^s \in Y^s, \quad \forall s \in S, \\ & x^s \in X, \quad \forall s \in S \end{aligned}$$

$$\begin{aligned}
x^s - \bar{x} &= 0, \quad \forall s \in S \\
T_m^s x^s + W_m^s y^s + \sum_{(i,j) \in B_U} U_{m,i,j}^s w_{i,j}^s &= h_m^s, \quad \forall m \in M, \forall s \in S \\
y_j^s &= (N_j^{U,s} - N_j^{L,s}) \left(\sum_{l \in P} 2^l z_{j,l}^s + \Delta y_j^s \right) + N_j^{L,s}, \quad \forall s \in S, \forall j \in \{j \mid (i,j) \in B_Q \cup B_U\} \\
0 &\leq \Delta y_j^s \leq 2^p, \quad \forall s \in S, \forall j \in \{j \mid (i,j) \in B_Q \cup B_U\} \\
w_{i,j}^s &= (N_j^{U,s} - N_j^{L,s}) \left(\sum_{l \in P} 2^l \hat{y}_{i,j,l}^s + \Delta w_{i,j}^s \right) + y_i^s N_j^{L,s}, \quad \forall s \in S, \forall (i,j) \in B_Q \cup B_U \\
2^p (y_i^s - N_i^{U,s}) + N_i^{U,s} \Delta y_j^s &\leq \Delta w_{i,j}^s \leq 2^p (y_i^s - N_i^{L,s}) + N_i^{L,s} \Delta y_j^s, \quad \forall s \in S, \forall (i,j) \in B_Q \cup B_U \\
N_i^{L,s} \Delta y_j^s &\leq \Delta w_{i,j}^s \leq N_i^{U,s} \Delta y_j^s, \quad \forall s \in S, \forall (i,j) \in B_Q \cup B_U \\
N_i^{L,s} z_{j,l}^s &\leq \hat{y}_{i,j,l}^s \leq N_i^{U,s} z_{j,l}^s, \quad \forall s \in S, \forall (i,j) \in B_Q \cup B_U, l \in P \\
N_i^{L,s} (1 - z_{j,l}^s) &\leq y_i^s - \hat{y}_{i,j,l}^s \leq N_i^{U,s} (1 - z_{j,l}^s), \quad \forall s \in S, (i,j) \in B_Q \cup B_U, l \in P \\
z_{j,l}^s &\in \{0, 1\}, \quad \forall s \in S, \forall j \in \{j \mid (i,j) \in B_Q \cup B_U\}, \forall l \in P,
\end{aligned} \tag{5}$$

where $P = \{p, \dots, -1\}$ and $w_{i,j}^s$ represent the product $y_i^s y_j^s$. Andrade et al. (2019) have demonstrated that as the precision parameter p approaches $-\infty$ the corresponding RNMDT $_p$ relaxation becomes increasingly tighter.

2.2. p -Lagrangian relaxation

Let us consider the RNMDT $_p$ problem (5) defined in Section 2.1, where the precision factor p is fixed to some negative integer value. The p -Lagrangian decomposition of RNMDT $_p$ can be obtained by applying Lagrangian relaxation to relax the NAC

$$x^s - \bar{x} = 0, \quad \forall s \in S.$$

Let $\lambda = (\lambda^1, \dots, \lambda^{|S|}) \in \mathbb{R}^{n_x \times |S|}$ be the vector of dual multipliers associated with the relaxed NAC. By setting $\mu^s = \frac{1}{\pi^s} \lambda^s$, $\forall s \in S$, the p -Lagrangian dual function can be defined as

$$L(\mu) = \left\{ \min_{x, \bar{x}, y, w} \sum_{s \in S} \pi^s \left(c^\top x^s + q^{s\top} y^s + \sum_{(i,j) \in B_Q} Q_{i,j}^s w_{i,j}^s + \mu^s (x^s - \bar{x}) \right) : (x^s, y^s, \Gamma^s) \in G^s, \forall s \in S \right\}, \tag{6}$$

where $\Gamma^s = \{w^s, \Delta y^s, \Delta w^s, \hat{y}^s, z^s\}$ and G^s is defined by the following set of constraints:

$$G^s = \begin{cases} x^s \in X \\ y^s \in Y^s \\ T_m^s x + W_m^s y^s + \sum_{(i,j) \in B_U} U_{m,i,j}^s w_{i,j}^s = h_m^s, \forall m \in M \\ y_j^s = (N_j^{U,s} - N_j^{L,s}) \left(\sum_{l \in P} 2^l z_{j,l}^s + \Delta y_j^s \right) + N_j^{L,s}, \forall j \in \{j \mid (i,j) \in B_Q \cup B_U\} \\ 0 \leq \Delta y_j^s \leq 2^p, \forall j \in \{j \mid (i,j) \in B_Q \cup B_U\} \\ w_{i,j}^s = (N_j^{U,s} - N_j^{L,s}) \left(\sum_{l \in P} 2^l \hat{y}_{i,j,l}^s + \Delta w_{i,j}^s \right) + y_i^s N_j^{L,s}, \forall (i,j) \in B_Q \cup B_U \\ 2^p (y_i^s - N_i^{U,s}) + N_i^{U,s} \Delta y_j^s \leq \Delta w_{i,j}^s \leq 2^p (y_i^s - N_i^{L,s}) + N_i^{L,s} \Delta y_j^s, \forall (i,j) \in B_Q \cup B_U \\ N_i^{L,s} \Delta y_j^s \leq \Delta w_{i,j}^s \leq N_i^{U,s} \Delta y_j^s, \forall (i,j) \in B_Q \cup B_U \\ N_i^{L,s} z_{j,l}^s \leq \hat{y}_{i,j,l}^s \leq N_i^{U,s} z_{j,l}^s, \forall (i,j) \in B_Q \cup B_U, \forall l \in P \\ N_i^{L,s} (1 - z_{j,l}^s) \leq y_i^s - \hat{y}_{i,j,l}^s \leq N_i^{U,s} (1 - y_{j,l}^s), \forall (i,j) \in B_Q \cup B_U, \forall l \in P \\ z_{j,l}^s \in \{0, 1\}, \forall j \in \{j \mid (i,j) \in B_Q \cup B_U\}, \forall l \in P. \end{cases}$$

The variable $\bar{x}$ in (6) is unconstrained. Therefore, in order for the p -Lagrangian dual function $L(\mu)$ to be bounded, we must impose the dual feasibility condition $\sum_{s \in S} \pi^s \mu^s = 0$. With this assumption in mind, the p -Lagrangian dual function (6) can be explicitly decomposed for each scenario $s \in S$

$$L(\mu) = \sum_{s \in S} \pi^s L^s(\mu^s), \quad (7)$$

where

$$L^s(\mu^s) = \left\{ \min_{x,y,w} (c^s + \mu^s)^\top x^s + q^{s^\top} y^s + \sum_{(i,j) \in B_Q} Q_{i,j}^s w_{i,j}^s : (x^s, y^s, \Gamma^s) \in G^s, \forall s \in S \right\}. \quad (8)$$

For any fixed value of $\mu = (\mu_1, \dots, \mu_{|S|})$, the p -Lagrangian dual function (7) provides a lower bound for the primal RNMDT_p problem (5) (Andrade et al., 2022). Our objective is to find the tightest (i.e., the uppermost) lower bound. Therefore, the dual bound can be obtained by solving the p -Lagrangian dual problem

$$z_{LD} = \max_{\mu} \left\{ L(\mu) : \sum_{s \in S} \pi^s \mu^s = 0 \right\}. \quad (9)$$

2.3. Solution method for p -Lagrangian dual function

In this section, we present adaptations of the proximal BM (Kim and Dandurand, 2022) and FWPH (Boland et al., 2018) for solving the p -Lagrangian dual problem (9). One should bear in mind that alternative nonsmooth (convex) optimisation algorithms can be potentially applied to solve p -Lagrangian dual function $L(\mu)$. The choice for proximal BM and FWPH was motivated by the literature on dual Lagrangian-based methods, including their reported efficiency (see, e.g., Feltenmark and Kiwiel, 2000; Boland et al., 2018; Palani et al., 2019; Bashiri et al., 2021), and our own experience with preliminary experiments involving both and other simpler nonsmooth optimisation methods (i.e., sub-gradient and cutting planes methods).

2.3.1. Proximal bundle method

We use an adaptation of a proximal BM utilised by Kim and Dandurand (2022) to solve the p -Lagrangian dual problem (9). The BM relies on an iterative approximation of the p -Lagrangian dual function $L(\mu)$ with piece-wise linear functions via cutting planes. Kim and Dandurand (2022) proposed a solution method for stochastic mixed-integer programming problems inspired by Carøe and Schultz (1999) and utilised the adaptation of the proximal BM presented in Kiwiel (1990) as a subroutine to solve dual problems. Kiwiel (1990) developed a new approach for updating the weights of proximal terms in BMs for minimising a convex function (Kiwiel, 1984, 1985, 1987). This technique can significantly reduce the number of cutting planes required to reach the desired convergence accuracy. The pseudo-code of the adaption of the proximal BM proposed by Kim and Dandurand (2022) is presented in Algorithm 1.

Suppose that in the k th iteration of the proximal BM, we have computed Lagrangian multipliers μ_l and centres of mass $\bar{\mu}_l$, for $l = 1, \dots, k-1$. In what follows, we present the adaptation of the proximal BM to update these parameters.

The Lagrangian multiplier μ_k is computed as follows:

$$\mu_k = \arg \max_{\mu} \left\{ m_k(\mu) - \frac{u_k}{2} \|\mu_k - \bar{\mu}_{k-1}\| \right\}, \quad (10)$$

where $m_k(\mu)$ is piece-wise linear approximation of $L(\mu)$ at iteration k given by

$$m_k(\mu) = \max_{\theta^s} \sum_{s \in S} \theta^s \quad (11)$$

$$\text{s.t.: } \theta^s \leq L^s(\mu_l) - \left(\frac{\partial L^s(\mu_l)}{\partial \mu_l} \right)^\top (\mu - \mu_l), \quad \forall s \in S, \quad l = 1, \dots, k-1. \quad (12)$$

The convergence of the proximal BM strongly relies on the update of the proximal parameter u_k and the centre of mass of $\bar{\mu}_k$. In line with the procedure developed in Kim and Dandurand (2022), the centre of mass $\bar{\mu}_k$ is updated as follows:

$$\bar{\mu}_k = \begin{cases} \mu_k, & \text{if } L(\mu_k) \geq L(\bar{\mu}_k) + a_L v_k \quad (\text{serious step}) \\ \bar{\mu}_{k-1}, & \text{otherwise} \quad (\text{null step}), \end{cases} \quad (13)$$

Algorithm 1. Proximal bundle method

initialise: $k = 0, k_{\max} \in \mathbb{N}, \mu_0, \bar{\mu}_0 = \mu_0, u_{\min}, u_1 > u_{\min}, C_{\min}^u, C_{\text{avg}}^u, C_{\max}^u, C^v, i_{\min}, i_{\max}$ and $i_1^u = 0$.

For each $s \in S$ solve $L^s(\mu)$ with $\mu = \mu_0$ and solve (11)–(12) with $l = 0$ to form m_1 .

repeat

$k = k + 1$.

From (10) obtain μ_k and the value of $m_k(\mu_k)$.

For each $s \in S$, solve $L^s(\mu)$ at the point $m = m_k$.

Compute $v_k, h_k, \bar{\delta}_k, g_k$ and δ_k as in (14), (17), (19), (20) and (21), respectively.

if $L(\mu_k) - L(\bar{\mu}_{k-1}) \geq a_L v_k$ **then**

$\bar{\mu}_k = \mu_k$

if $L(\mu_k) - L(\bar{\mu}_{k-1}) \geq a_R v_k$ and $i_k^u > 0$ **then**

$u_{k+1} = \max\{h_k, C_{\min}^u u_k, u_{\min}\}$

else if $i_k^u > i_{\max}$ **then**

$u_{k+1} = \max\{C_{\text{avg}}^u u_k, u_{\min}\}$

end if

$i_{k+1}^u = \begin{cases} \max\{i_k^u + 1, 1\}, & \text{if } u_{k+1} = u_k \\ 1, & \text{otherwise} \end{cases}$

else

$\bar{\mu}_k = \bar{\mu}_{k-1}$

if $\bar{\delta}_k > \max\{\delta_k + |g_k|, C^v v_k\}$ and $i_k^u < i_{\min}$ **then**

$u_{k+1} = \min\{h_k, C_{\max}^u u_k\}$

else

$u_{k+1} = C_{\max}^u u_k$

end if

$i_{k+1}^u = \begin{cases} \min\{i_k^u - 1, -1\}, & \text{if } u_{k+1} = u_k, \\ -1, & \text{otherwise.} \end{cases}$

end if

Formulate m_{k+1} as in (11)–(12).

until $v_k \leq \epsilon_{BM}$ **or** $k > k_{\max}$

return: $\bar{\mu}_k, L(\bar{\mu}_k), (x_{k_{\max}}^s, y_{k_{\max}}^s, w_{k_{\max}}^s)_{s \in S}$ solving $L(\bar{\mu}_k)$.

where we typically have $a_L \in (0, 0.5)$ and

$$v_k = m_k(\mu_k) - L(\bar{\mu}_{k-1}) \quad (14)$$

representing the predicted increase of p -Lagrangian function $L(\mu)$.

The proximal term u_k must be chosen carefully. To prevent the proximal BM from taking a serious step too frequently (after too little improvement in $L(\mu)$), u_k cannot be too large. On the other hand, if u_k value is too small, the method will take many null steps before it finds a good candidate for the new centre of mass. To accelerate the performance of the proximal BM, tests identifying whether the proximal parameter u_k value was too small or too large can be employed.

The case when u_k is too large can be identified by testing whether

$$L(\mu_k) \geq L(\bar{\mu}_{k-1}) + a_R v_k, \quad (15)$$

where $a_R \in (a_L, 1)$. If (15) holds the proximal term u_k is updated as

$$u_{k+1} = \max\{h_k, C_{\min}^u u_k, u_{\min}\}, \quad (16)$$

with

$$h_k = 2u_k \left(1 - \frac{L(\mu_k) - L(\bar{\mu}_{k-1})}{v_k} \right), \quad (17)$$

and $C_{\min}^u \in \mathbb{R}$. On the other hand, whether the proximal term u_k is too small is identified by the test

$$\bar{\delta}_k > \max\{\delta_k(\bar{\mu}_{k-1}) + |g_k|, C^v v_k\}, \quad (18)$$

where $C^v \in \mathbb{R}$ and

$$\bar{\delta}_k = L(\mu_k) - \left(\sum_{s \in S} \frac{\partial L^s(\mu_k)}{\partial \mu_k} (x_k^s) \right)^\top (\mu_k - \bar{\mu}_{k-1}) - L(\bar{\mu}_{k-1}), \quad (19)$$

in which $x_k^s, \forall s \in S$, is the optimal solution of the p -Lagrangian sub-problem $L^s(\mu)$ with $\mu = \mu_k$,

$$g_k \in \partial m_k(\mu_k), \quad (20)$$

and

$$\delta_k(\mu) = m_k(\mu_k) + (g_k)^\top (\mu - \mu_k) - L_\pi(\mu), \quad (21)$$

where ∂ denotes the sub-differential of m_k at u_k , making thus g_k a sub-gradient of m_k at u_k . If (18) holds, the proximal term u_k is updated as

$$u_{k+1} = \min\{h_k, C_{\max}^u u_k\}, \quad (22)$$

where $C_{\max}^u \in \mathbb{R}$. Algorithm 1 summarises the developed proximal BM, starting with a step $k = 0$ and the initialisation of the parameters.

Following the developments in Kim and Dandurand (2022), Algorithm 1 includes an additional parameter i_k^u that counts consecutive serious or null steps and enforces the tuning of the proximal term u_k , hoping to speed up the algorithm's convergence. The algorithm terminates when predicted increases v_k are within an arbitrary tolerance ϵ_{BM} . For proof of the convergence of the BM adaptation presented in Algorithm 1, one can refer to, for instance, Kiwiel (1990).

2.3.2. Frank–Wolfe progressive-hedging method

Alternatively, one can apply the FWPH method (Boland et al., 2018) to solve the p -Lagrangian dual problem (9). FWPH is applied to the primal characterisation of (9):

$$z_{LD} = \min_{x, \bar{x}, y, w} \left\{ \sum_{s \in S} \pi^s \left(c^\top x^s + q^{s^\top} y^s + \sum_{(i,j) \in B_Q} Q_{i,j}^s w_{i,j}^s \right) : (x^s, y^s, \Gamma^s) \in \text{conv}(G^s), x^s = \bar{x}, \forall s \in S \right\}, \quad (23)$$

Algorithm 2. Frank–Wolfe progressive hedging (FWPH) method

initialise: $(V_0^s)_{s \in S}, (x_0^s)_{s \in S}, \mu_0, \rho, \alpha, \epsilon_{FWPH}, k_{max}, t_{max}$ and ϵ_{SDM} .

Compute $\bar{x}_0 = \sum_{s \in S} \pi^s x_0^s$ and $\mu_1^s = \mu_0^s + \rho(x_0^s - \bar{x}_0)$.

for $k = 1, \dots, k_{max}$ **do**

for $s \in S$ **do**

$\tilde{x}^s = (1 - \alpha)\bar{x}_{k-1} + \alpha x_{k-1}^s$,

$[x_k^s, y_k^s, w_k^s, V_k^s, L^s(\mu_k^s)] = \text{SDM}(V_{k-1}^s, \tilde{x}^s, \mu_k^s, \bar{x}_{k-1}, t_{max}, \epsilon_{SDM})$

end for

Compute $L(\mu_k) = \sum_{s \in S} \pi^s L^s(\mu_k^s)$ and $\bar{x}_k = \sum_{s \in S} \pi^s x_k^s$.

if $\sqrt{\sum_{s \in S} \pi^s \|x_k^s - \bar{x}_{k-1}\|_2^2} \leq \epsilon_{FWPH}$ **then**

return $((x_k^s, y_k^s, w_k^s)_{s \in S}, \bar{x}_k, \mu_k, L(\mu_k))$

end if

 Compute $\mu_{k+1}^s = \mu_k^s + \rho(x_k^s - \bar{x}_k)$ for each $s \in S$.

end for

return $(x_{k_{max}}^s, y_{k_{max}}^s, w_{k_{max}}^s)_{s \in S}, \bar{x}_{k_{max}}, \mu_{k_{max}}, L(\mu_{k_{max}})$.

where $\text{conv}(G^s)$ denotes the convex hull of G^s for each $s \in S$.

The FWPH method primarily relies on the classical progressive hedging method (Rockafellar and Wets, 1991). However, unlike progressive hedging, FWPH can guarantee convergence in case the original problem is mixed-integer. Indeed, using the progressive hedging method as proposed in Rockafellar and Wets (1991) to solve 2SSMIP might result in sub-optimal bounds, cycling behaviour and poor convergence behaviour of Lagrangian dual bound for problem (9) as the presence of integer variables hinders its convergence guarantees. As a result, progressive hedging has typically been employed as a heuristics approach (see, e.g., Watson and Woodruff, 2011). FWPH integrates an extension of the Frank–Wolfe method called the simplicial decomposition method (SDM) to iteratively construct an inner approximation of $\text{conv}(G^s)$ for each $s \in S$. The composition of SDM and progressive hedging method allows for overcoming the aforementioned convergence issue. Additionally, it allows replacing the additional step of solving mixed-integer linear sub-problems with solving convex continuous quadratic sub-problems when calculating the Lagrangian dual bound. This, in turn, improves the computational performance of the FWPH method (Boland et al., 2018).

The FWPH method uses the augmented Lagrangian dual problem, that is, a modified Lagrangian dual problem in which the Lagrangian dual function is augmented by a penalty term that acts as a regularisation term. The augmented Lagrangian dual function based on relaxing the NAC constraints $x^s = \bar{x}, \forall s \in S$ in RNMDT_p problem (5) is

$$L_\rho(x, y, w, \bar{x}, \mu) = \sum_{s \in S} \pi^s L_\rho^s(x^s, y^s, w^s, \bar{x}, \mu^s), \quad (24)$$

where

$$L_\rho^s(x^s, y^s, w^s, \bar{x}, \mu^s) = c^\top x^s + q^{s\top} y^s + \sum_{(i,j) \in B_Q} Q_{i,j}^s w_{i,j}^s + \mu^{s\top} (x^s - \bar{x}) + \frac{\rho}{2} \|x^s - \bar{x}\|_2^2$$

and $\rho > 0$ is a penalty parameter.

The FWPH algorithm pseudo-code is stated in Algorithm 2. The parameter k_{max} defines the maximum number of iterations for the Frank–Wolfe method and ϵ_{FWPH} is a convergence tolerance

Algorithm 3. Simplicial decomposition method (SDM)

initialise: $V_0^s, x_0^s, \mu^s, \bar{x}, t_{max}$ and ϵ_{SDM} .
for $t = 1, \dots, t_{max}$ **do**
 $\hat{\mu}_t^s = \mu^s + \rho(x_{t-1}^s - \bar{x})$,
 $(\hat{x}^s, \hat{y}^s, \hat{w}^s) \in \operatorname{argmin}_{x,y,w} \{(c + \hat{\mu}_t^s)^\top x + q^{s\top} y + \sum_{(i,j) \in B_Q} Q_{i,j}^s w_{i,j} : (x, y, w) \in G^s\}$

if $t = 1$ **then**
 $L^s(\hat{\mu}_t^s) = (c + \hat{\mu}_t^s)^\top \hat{x}^s + q^{s\top} \hat{y}^s + \sum_{(i,j) \in B_Q} Q_{i,j}^s \hat{w}_{i,j}^s$
end if
Compute
 $\Gamma^t = -[(c + \hat{\mu}_t^s)^\top (\hat{x}^s - x_{t-1}^s) + q^{s\top} (\hat{y}^s - y_{t-1}^s) + \sum_{(i,j) \in B_Q} Q_{i,j}^s (\hat{w}_{i,j}^s - w_{t-1,i,j}^s)]$,

 $V_t^s = V_{t-1}^s \cup \{(\hat{x}^s, \hat{y}^s, \hat{w}^s)\}$ and
 $(x_t^s, y_t^s) \in \operatorname{argmin}_{x,y,w} \{L_\rho^s(x, y, w, \bar{x}, \hat{\mu}_t^s) : (x, y, w) \in \operatorname{conv}(V_t^s)\}$.
if $\Gamma^t \leq \epsilon_{SDM}$ **then**
 return $(x_t^s, y_t^s, w_t^s, V_t^s, L(\hat{\mu}_t^s))$
end if
end for
return $(x_{t_{max}}^s, y_{t_{max}}^s, w_{t_{max}}^s, V_{t_{max}}^s, L(\hat{\mu}_{t_{max}}^s))$.

parameter. The termination criterion involves the term $\sum_{s \in S} \pi^s \|x_k^s - \bar{x}_{k-1}\|$ that represents the sum of squared norms of primal and dual residuals associated with (23). These residuals evaluate how close the solution candidate $((x^s, y^s, w^s), \bar{x})$ is to satisfy the necessary and sufficient optimality conditions for (23).

As a subroutine, Algorithm 2 employs the SDM to minimise $L_\rho^s(x, y, w, \bar{x}, \mu^s)$ over $(x, y, w) \in \operatorname{conv}(G^s)$ for a given $s \in S$. The pseudo-code for SDM is stated in Algorithm 3. The precondition for the SDM algorithm is that $V_0^s \subset \operatorname{conv}(G^s)$ and $\bar{x} = \sum_{s \in S} \pi^s x_0^s$, where V_t^s are discrete sets of points such that $V_t^s \subset \operatorname{conv}(G^s)$. Parameter t_{max} defines the maximum number of iterations for SDM, and $\epsilon_{SDM} > 0$ is the convergence tolerance. The parameter α affects the initial linearisation point $\tilde{x}^s$ of the SDM method.

3. Dual decomposition

In this section, we present the branching approach we employ, which is inspired by the dual decomposition proposed in Carøe and Schultz (1999). The authors proposed a solution method for linear stochastic multi-stage problems that may involve integrality requirements at each stage. The solution method relies on dual decomposition combined with BnB strategies to ensure convergence. In what follows, we discuss our adaptation of the solution method proposed in Carøe and Schultz (1999) for the mixed-integer RNMDT relaxations of RDEM problems.

Let T be the set of unexplored nodes in the BnB search, in which each node is denoted by N . The key idea behind our approach is to extend the BnB procedure proposed in Carøe and Schultz (1999) for the RNMDT_p problem (5). Specifically, we perform branching on the first-stage variables and use the solution of the p -Lagrangian dual problem, as described in (9), as the

bounding procedure. To form candidates for feasible first-stage variables solution, the method uses an average $\bar{x}_N = \sum_{s \in S} \pi^s x_N^{*,s}$, combined with a rounding heuristic to fulfil the integrality requirements, where $x_N^{*,s}, \forall s \in S$, is obtained from solving the N node-corresponding dual problem (9).

If $\bar{x}_N$ violates integrality conditions for some integer index i , that is, $\lfloor \bar{x}_{N,i} \rfloor < \bar{x}_{N,i} < \lceil \bar{x}_{N,i} \rceil$, two nodes N^L and N^R with the correspondent sub-problems (9) are formed from parent node N , where feasibility sets $G_{N^L}^s$ and $G_{N^R}^s, \forall s \in S$, are formed, respectively, as

$$G_{N^L}^s = G_N^s \cap \{x_i^s \leq \lfloor \bar{x}_{N,i} \rfloor\} \text{ and} \quad (25)$$

$$G_{N^R}^s = G_N^s \cap \{x_i^s \geq \lceil \bar{x}_{N,i} \rceil\}. \quad (26)$$

There are multiple approaches for selecting the integer index i of a fractional-valued variable, that is, $\lfloor \bar{x}_{N,i} \rfloor < \bar{x}_{N,i} < \lceil \bar{x}_{N,i} \rceil$ to perform branching. Morrison et al. (2016) describe several common methods, of which we highlight the following:

- *Branching on the most (least) fractional or most (least) infeasible variable* (see, e.g., Ortega and Wolsey, 2003; Achterberg et al., 2005): This approach involves selecting the integer index i such that the fractional part of $\bar{x}_{N,i}$ is closest to (or furthest from) 0.5.
- *Pseudo-cost branching* (see, e.g., Bénichou et al., 1971): In this method, the integer index i is chosen based on the expected impact of branching on $\bar{x}_{N,i}$ on the objective function, leveraging the history of previous branching decisions in the tree.
- *Strong branching* (see, e.g., Achterberg et al., 2005; Achterberg, 2007): This technique recommends selecting the integer index i such that branching on $\bar{x}_{N,i}$ results in the greatest change to the objective function. This is typically evaluated by solving LP relaxations for the child nodes generated by branching on $\bar{x}_{N,i}$.

In addition to these strategies, Morrison et al. (2016) highlight several other methods, including combining strong and pseudocost branching (see, e.g., Linderoth and Savelsbergh, 1999), backdoor branching, which uses an auxiliary integer program to identify a subset of indices that reduce the search tree size (see, e.g., Fischetti and Monaci, 2011) and information-theoretic branching, where indices are chosen to minimise uncertainty in the sub-problems (see, e.g., Gilpin and Sandholm, 2011).

In our approach, a predominant feature is that we typically have a small search tree and that each node typically requires considerable computational work. The combination of these two characteristics led us to lean towards a simpler branching strategy.

If $\bar{x}_N$ satisfies integrality conditions but $x_N^{*,s}, \forall s \in S$, violates non-anticipativity conditions, two nodes N^L and N^R with the correspondent sub-problems (9) are formed from the parent node N , where feasibility sets $G_{N^L}^s$ and $G_{N^R}^s, \forall s \in S$, are formed, respectively, as

$$G_{N^L}^s = G_N^s \cap \{x_i^s \leq \bar{x}_{N,i} - \epsilon_{BB}\} \text{ and} \quad (27)$$

$$G_{N^R}^s = G_N^s \cap \{x_i^s \geq \bar{x}_{N,i} + \epsilon_{BB}\}, \quad (28)$$

where $\epsilon_{BB} > 0$. The branching index i is chosen based on the measure of the dispersion in the first-stage scenario solutions, for example, if the dispersion of the component i : $\sigma_i = \max_{s \in S} x_{N,i}^{*,s} - \min_{s \in S} x_{N,i}^{*,s}$ is zero, this should imply the non-anticipativity of this component

$$x_{N,i}^{*,1} = \dots = x_{N,i}^{*,|S|}.$$

Therefore, in case of violating non-anticipativity constraints, branching is performed on the index i with the largest dispersion.

The BnB search requires that any node not pruned or fathomed must be explored. Therefore, reducing the size of the search tree is directly tied to the efficiency of the pruning strategy. Morrison et al. (2016) highlight the following major groups of strategies:

- *Pruning by bound*: This strategy involves pruning nodes whose sub-problem objective value is worse (higher in the case of minimisation) than the incumbent solution objective value (see, e.g., Arora et al., 2002; Vilà and Pereira, 2014). The bounds of the sub-problems can be calculated either by solving a linear relaxation of the node sub-problems or by using duality techniques.
- *Pruning by the dominance relations*: this strategy involves pruning a node if another node's sub-problem dominates its sub-problem. A sub-problem is said to be dominated if another sub-problem guarantees a better solution or an equally good solution that is less computationally intensive (see, e.g., Fischetti and Salvagnin, 2010; Sewell et al., 2012). Dominance rules can be defined based on whether memory is used to store previously explored sub-problems. The *memory-based dominance* involves comparing unexplored nodes to sub-problems of already generated and stored nodes. If a stored sub-problem dominates a new one, the new sub-problem can be pruned. The *non-memory-based dominance* involves determining the existence of a dominating node without relying on stored sub-problems. It checks whether a dominance relation holds regardless of whether the dominating node has been explored.

In the proposed approach, we prune nodes based on the lower bounds generated by solving the dual sub-problems, for the same reasons that we chose simpler branching rules (i.e., small search trees with computationally expensive nodes).

Algorithm 4 summarises the adaptation of the BnB method presented by Carøe and Schultz (1999) that, hereafter, we refer to as p -BnB. For each BnB node $N \in T$, we generate node sub-problem (9) and compute its dual bound value z_N^* as well as corresponding optimal dual and primal variables values $(\mu_N^{*,s})_{s \in S}$ and $(x_N^{*,s}, y_N^{*,s}, w_N^{*,s})_{s \in S}$, respectively, by applying Algorithm 1 or 2. If the dual bound value $z_N^* > z_{UB}$ or one of the N sub-problems $s \in S$ is infeasible, the node N is fathomed. Otherwise, we check whether solution $x_N^{*,s}$ violates non-anticipativity or integrality conditions. If that is the case, we perform branching as described in (25)–(26) on the most fractional variable $\bar{x}_{N,i}$ if $x_N^{*,s}$ violates integrality conditions. Otherwise, we perform branching as described in (27)–(28) on the variable with the largest dispersion σ_i if $x_N^{*,s}$ violates non-anticipativity conditions. If $x_N^{*,s}$ satisfies both non-anticipativity and integrality conditions, we update the best upper bound value $z_{UB} = z_N^*$ and best solution value $x^* = \bar{x}_N$. Lastly, we update the best lower bound value z_{LB} by setting it to the smallest dual bound value z_N^* among the nodes N that are yet to be fathomed. The algorithm continues until the set T is empty.

Algorithm 4. *p*-branch-and-bound method (*p*-BnB)

initialise: $T = \emptyset$, $z_{UB} = \infty$, $z_{LB} = -\infty$, $x^* = \emptyset$, $\epsilon_{BB} > 0$ and $\epsilon_{NAC} \geq 0$.
 Create root node N_0 sub-problem (9), $T = T \cup \{N_0\}$.

repeat
 Choose a node $N \in T$.
 $T = T \setminus \{N\}$.
 Apply Algorithm 1 or 2 to the node N sub-problem (9) to obtain z_N^* , $(\mu_N^{*,s})_{s \in S}$ and $(x_N^{*,s}, y_N^{*,s}, w_N^{*,s})_{s \in S}$.
 if $z_N^* > z_{UB}$ or one of the N sub-problems is infeasible **then**
 fathom N
 else
 Compute $\bar{x}_N = \sum_{s \in S} \pi^s x_N^{*,s}$.
 Compute $\sigma_i = \max_{s \in S} \{x_{N,i}^{*,s}\} - \min_{s \in S} \{x_{N,i}^{*,s}\}$ for $i \in \{1, \dots, n_x\}$.
 if $\max_{i \in \{1, \dots, n_x\}} \{\sigma_i\} \leq \epsilon_{NAC}$ **then**
 if $\bar{x}_{N,i}$ is fractional for some integer index $i \in \{1, \dots, n_x\}$ **then**
 Choose integer variable index $i \in \{1, \dots, n_x\}$ such as $\lfloor \bar{x}_{N,i} \rfloor < \bar{x}_{N,i} < \lceil \bar{x}_{N,i} \rceil$.
 Create two new nodes N^L and N^R via (25) and (26), respectively.
 else if $z_{UB} > z_N^*$ **then**
 $z_{UB} = z_N^*$,
 $x^* = \bar{x}_N$
 end if
 else if $\max_{i \in \{1, \dots, n_x\}} \{\sigma_i\} > \epsilon_{NAC}$ and $z_{UB} > z_N^*$ **then**
 if $\bar{x}_{N,i}$ is fractional for some integer index $i \in \{1, \dots, n_x\}$ **then**
 Choose integer variable index $i \in \{1, \dots, n_x\}$ such as $\lfloor \bar{x}_{N,i} \rfloor < \bar{x}_{N,i} < \lceil \bar{x}_{N,i} \rceil$.
 Create two new nodes N^L and N^R via (25) and (26), respectively.
 else
 Choose continuous variable index $i \in \arg \max_i \sigma_i$.
 Create two nodes N^L and N^R via (27) and (28), respectively.
 end if
 $T = T \cup \{N^L, N^R\}$.
 end if
 end if
 Update z_{LB} .
until $T = \emptyset$

To clarify how the elements composing Algorithm 4 interact, we provide a graphical representation of the algorithm's structure in Fig. 1. Additionally, Section C presents an illustrative example, showcasing the procedures described in Algorithm 4 and demonstrating how they apply to a small problem instance.

In what follows, we provide a theoretical justification of Algorithm 4 convergence to the optimal solution of RNMDT_{*p*} relaxation (problem (5)). The convergence of Algorithm 4 to the solution set of problem (5) considering any fixed value of $p = \{-\infty, \dots, -1\}$ is stated in Theorem 1. Consequently, the solution set of problem (5) converges within a predefined tolerance level to the solution set of the primal RDEM (problem (4)) as the precision factor p approaches $-\infty$. Formally, the justification for convergence of the RNMDT relaxation (problem (5)) is stated in Theorem 2. It is worth highlighting that when discussing the convergence from (5) to (4) as p approaches $-\infty$ we take into consideration that, in practical applications, this corresponds to achieving a predetermined epsilon-accurate convergence rather than an absolute convergence.

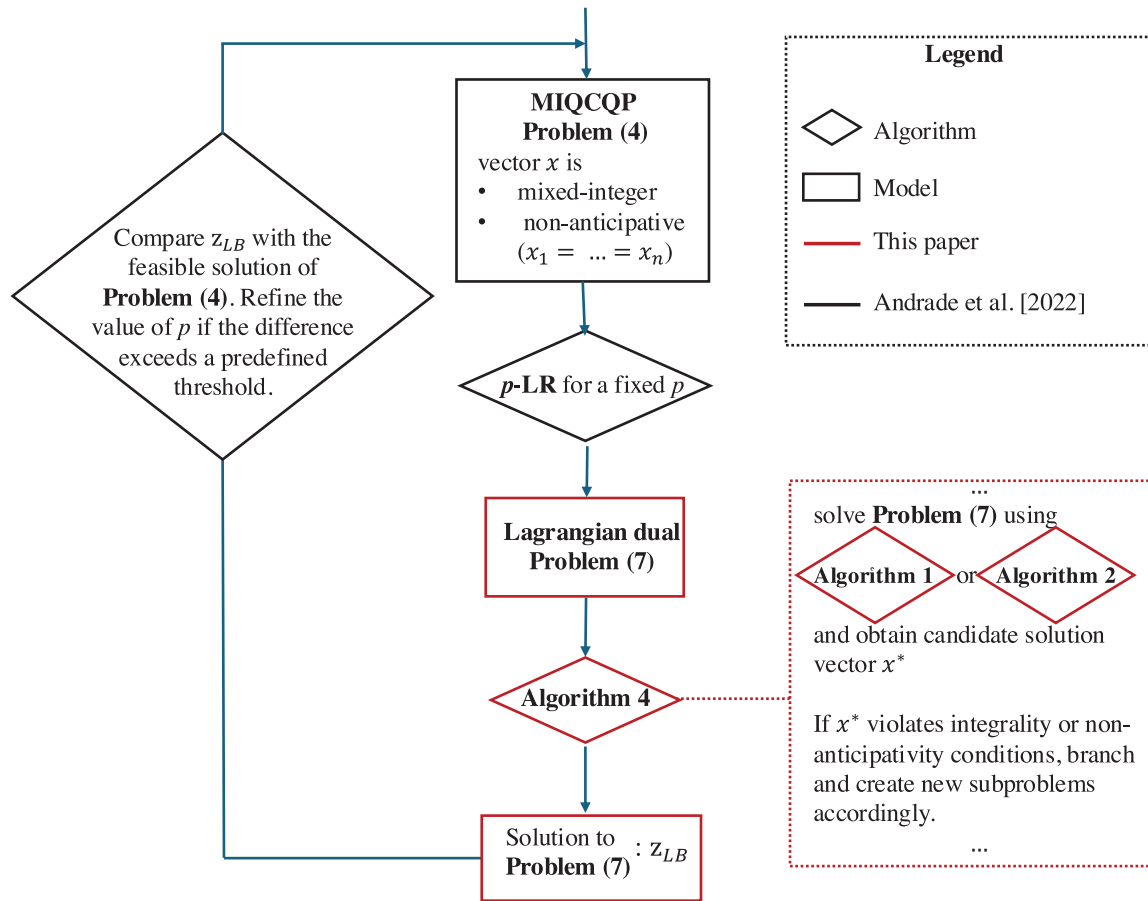


Fig. 1. Graphical representation of Algorithm 4 structure.

Theorem 1. Suppose we consider the RNMDT relaxation (problem (5)) with an arbitrary fixed value of the precision factor $p = \{-\infty, \dots, -1\}$. Then Algorithm 4 converges to the solution $(x_N^{*,s}, y_N^{*,s}, w_N^{*,s})_{s \in S}$ that is optimal for problem (5).

Proof. In Carøe and Schultz (1999), the authors demonstrate the termination in finitely many steps and convergence of Algorithm 4 to the optimal solution of problem (5) assuming that nodes p -Lagrangian dual sub-problem (9) are solved to optimality and, hence, yielding optimal dual bound. Employing either Algorithm 1 or 2 ensures the convergence to the optimal solution of the p -Lagrangian dual sub-problem (9). For the convergence of Algorithms 1 and 2 to the optimal solution of problem (6), please refer to Kim and Dandurand (2022) and Boland et al. (2018), respectively. $\square$

Theorem 2. Suppose we consider the RNMDT _{p} relaxation problem (5) with an arbitrary fixed value of the precision factor $p = \{-\infty, \dots, -1\}$. Then for any pair (p_1, p_2) such that $p_1 < p_2 \leq 0$ RNMDT _{p_1} is a tighter (or equal) relaxation of the original RDEM problem than RNMDT _{p_2} .

Table 1
Instance problem dimensions (per scenario)

Instance size	# of 1 st -stage variables	# of 2 nd -stage variables	# of constraints
Small (S)	100	100	100
Large (L)	200	200	200

Proof. See Andrade et al. (2019, Theorem 6). □

4. Computational experiments

This section presents numerical results for experiments performed using randomly generated non-convex 2SSMIP in the form of (4) or problems (4), as we refer to them hereinafter. All code and instances generated are available on the GitHub repository (Belyak, 2022). The experiments were designed using Julia (Version 1.7.3) language (Bezanson et al., 2017) and Gurobi (Version 9.1) solver (Gurobi Optimization, 2020). The code was run on Triton, Aalto University's high-performance computing cluster (Aalto Scientific Computing, 2022).

4.1. Design of experiments

We tested the efficiency of Algorithm 4 considering two alternative methods to solve sub-problems (9): the proximal BM presented in Section 2.3.1 and the FWPH method presented in Section 2.3.2. Algorithm 4 was implemented using parallel computing, meaning that the scenario-sub-problems (8) are solved in parallel in both solution methods (proximal BM and FWPH). For each instance, the number of processes utilised for parallel computing was equal to 30. The computational efficiency of Algorithm 4 was compared with Gurobi's (Gurobi Optimization, 2020) BnB algorithm with standard parameterisation.

We tested Algorithm 4 on 5 sets of randomly generated non-convex problem instances. Each set contained problems (4) with 50, 100 and 150 scenarios represented in two scales (small and large) as described in Table 1. More detailed information on the range of the coefficients for problems (4) is presented in Appendix A (see Table A1). Additionally, we assumed two different values of the precision factor $p \in \{-2, -1\}$. Hence, we considered 60 instances in total. It is important to note that smaller values of p were not explored, as the aforementioned values were found to be adequate for providing sufficient tolerance in terms of the RNMDT approximation. Furthermore, extensive numerical evidence demonstrating how good the RNMDT approximation is for different values of p has already been extensively discussed in Andrade et al. (2019, 2022).

For the sake of simplicity, for each instance, all the first-stage variables were assumed to be integer, and all the second-stage variables were assumed to be continuous. To make test instances similar to those available in the library of test problems for stochastic integer programming (Ahmed et al., 2015) (which are not MIQCQPs) in terms of the number of non-zero coefficients in the constraints and objective function, we assumed the quadratic matrices Q^s and $U_m^s \forall s \in S, \forall m \in M$ to be randomly generated with 1% density, which is of the same order of the density (i.e., proportion of non-zero elements in the problems' constraint matrices) of the test problems considered.

Table 2
Algorithm parameters

Proximal Bundle method	
u_{min}	10^{-3}
m_R	0.7
m_L	0.3
i_{max}	3
i_{min}	−3
C_{min}^u	0.1
C_{avg}^u	0.5
C_{max}^u	10
C^v	10
ϵ_{BM}	10^{-3}
Frank–Wolfe progressive hedging	
ρ	2
α	1
ϵ_{FWPH}	10^{-3}
ϵ_{SDM}	Not used

Therefore, problems (4) with 50, 100 and 150 scenarios would have in total 5100, 10,100 and 15,100 variables, respectively, in the case of small (S) instances and 10,200, 20,200 and 30,200 variables, respectively, in the case of large (L) instances.

Table 2 presents the parameter values used in the experiments for the proximal BM (Algorithm 1) and the FWPH (Algorithm 2). In addition to the parameters stated in Table 2, the maximum number of iterations for the proximal BM (k_{max}) was set to 1000. The maximum numbers of iterations for the FWPH algorithm (k_{max}) and SDM (t_{max}) were set to 1000 and 1, respectively. We considered $t_{max} = 1$ as Boland et al. (2019) suggest that considering higher values of the t_{max} parameter commonly increases the computational burden while not guaranteeing an adequate increase in the quality of the approximation of $\text{conv}(G^s)$ compared to the case when $t_{max} = 1$. It is also worth mentioning that since $t_{max} = 1$ for the SDM, the ϵ_{SDM} can be set to any arbitrary value as the condition $\Gamma^t \leq \epsilon_{SDM}$ is not going to be checked for the algorithm termination. Following Boland et al. (2018, Proposition 3.3), since we assume $t_{max} = 1$, to guarantee the convergence of Algorithm 2 one should generate initial sets $\{(V_0^s)_{s \in S}\}$ such that $\bigcap_{s \in S} \text{Proj}_x(\text{conv}(V_0^s)) \neq \emptyset$. Therefore, to initialise $(V_0^s)_{s \in S}$, we took one arbitrary scenario (in our case the first scenario in S , that is, $s = 1$) and set $V_0^1 = \{(x_0^1, y_0^1, w_0^1)\}$. Further, for each $s \in S, s \neq 1$, we initialised $V_0^s = \{(x_0^s, y_0^s, w_0^s), (x_0^1, \bar{y}^s, \bar{w}^s)\}$, where (x_0^s, y_0^s, w_0^s) solves $L^s(\mu_0^s)$ and $(\bar{y}^s, \bar{w}^s)$ solves,

$$\min_{y, w} \left\{ q^{s\top} y + \sum_{(i,j) \in B_Q} Q_{i,j}^s w_{i,j} : (x_0^1, y, w) \in G^s \right\}, \text{ for each } s \in S.$$

Starting dual multipliers values μ_0 for Algorithms 1 and 2 were set $\mu_0 = 0$. To set the first-stage variables $(x_0^s)_{s \in S}$ for Algorithm 2, we considered the solution of the p -Lagrangian dual function (6) for a fixed value of the dual variable $\mu = \mu_0$. The tolerances ϵ_{BB} and ϵ_{NAC} for the p -BnB

Table 3
Numerical results for the instances with low-density quadratic matrices

Instance parameters			Solution time (s)		
Size	$ S $	p	Full scale	p -BnB (FWPH)	p -BnB (BM)
S	50	−1	83.88	22.61	15.68
	50	−2	131.22	119.18	10.18
	100	−1	185.56	172.05	41.01
	100	−2	358.34	208.40	55.36
	150	−1	316.23	226.49	50.28
	150	−2	535.56	381.71	92.61
L	50	−1	687.88	630.49	119.81
	50	−2	866.44	420.63	122.50
	100	−1	1505.92	1637.15	367.48
	100	−2	2490.45	1708.13	284.36
	150	−1	2463.96	1372.53	523.98
	150	−2	3412.82	1031.00	369.48

(Algorithm 4) were set to 10^{-6} . As a time limit for solving each distinct instance, we considered one hour. In case multiple integer indices $i_1, \dots, i_{Int}$ are available for branching on the variable $\bar{x}_{N,i}$ in p -BnB, we simply choose the first index i_1 .

4.2. Numerical results

Table 3 presents averaged results of solving the small (S) and large (L) scale instances with the parameters as defined in Table 1 and Q and U matrices non-zero densities being set to 1%. We compared the time required to solve the instances with the proposed p -BnB method against solving them directly with the Gurobi solver (full scale). The columns ' p -BnB (FWPH)' and ' p -BnB (BM)' report the solution for the p -BnB method when employing FWPH and the proximal BM as a solution method for node sub-problems, respectively. Each cell in the 'Solution time' section represents the average solution time value for five instances generated using five different random seeds. It is worth mentioning that when calculating the average value for the column 'Full-scale' we have only considered the instances for which the Gurobi solver could generate a solution within one hour.

As the numerical results in Table 3 suggest, for small-scale instances, the proposed p -BnB method outperformed commercial solver Gurobi in terms of the solution time regardless of the method employed to solve the dual sub-problems. This conclusion also applies to the large-scale instances, except for the instance with 100 scenarios and precision factor $p = -1$. On average, applying p -BnB with FWPH allowed for saving up 31.41% and 32.76% of solution time for small- and large-scale instances, respectively, compared to solving the full-scale instances with Gurobi. The best improvement for the small-scale instances has been achieved for the instance with 50 scenarios and RNMDT precision factor $p = -1$, demonstrating a decrease in computational time by 73.05% compared to solving the instance directly with Gurobi. For the large-scale instances, the largest reduction in solution time was observed for the instance with 150 scenarios and RNMDT precision factor $p = -2$, allowing for reducing the solution time required by Gurobi by 69.79%. However, using p -BnB with the proximal BM instead has demonstrated even further improvements in

Table 4
Dimensions of instances with high-density Q matrices

Instance	# of scenarios	# of 1 st -stage variables	# of 2 nd -stage variables	# of constraints
1	15	30	25	25
2	20	30	30	20
3	20	40	15	15
4	30	30	20	15
5	40	20	10	15

computational solution time. Compared to solving the full-scale instances with Gurobi, p -BnB with the proximal BM demonstrated, on average, a decrease of 83.80% and 83.42% in solution time for the small- and large-scale instances, respectively. Moreover, the results suggest that the solution time improvement reached up to 92.24% for the small-scale instances, as in the case of the instances with 50 scenarios and an RNMDT precision factor of $p = -2$. For the large-scale instances, the maximum improvement in solution time was achieved for the instance with 150 scenarios and RNMDT precision factor being $p = -2$, allowing a reduction of 89.17% in the time required to solve that instance by Gurobi. However, it is important to highlight that in the case of utilising FWPH in the context of p -BnB we have observed a considerable portion of computational time spent by FWPH on generating the sets $\{(V_0^s)_{s \in S}\}$ at the beginning of Algorithm 2, particularly for the instances with a high number of scenarios. Hence, addressing this issue could potentially lead to an improvement in its computational time performance and possibly to approaching the efficiency of proximal BM.

Nevertheless, in all 60 instances, the p -BnB explored only one (root) node to identify the optimal solution. This effectively means that all of these instances were such that there was no duality gap when solving the p -Lagrangian duals and that bounds obtained by both methods were tight enough to find the optimal solution at the root node. This effect was also observed in Boland et al. (2018) where the authors reported convergence of the FWPH method to the optimal solution for most of the stochastic mixed-integer problem instances. Additionally, the usage of p -Lagrangian relaxation exploits the block-angular structure of the primal RDEM problem allowing one to obtain tighter bounds at the root if compared to linear programming (LP) relaxation. Such phenomena have been reported in Carøe and Schultz (1999) where the authors would obtain at a root node a duality gap of only 0.2–0.3% in case Lagrangian relaxation is explored while the LP relaxation, however, would provide a duality gap of 2.0–2.1%.

To demonstrate the convergence of the method in cases when the solution for the root node violates integrality or non-anticipativity conditions, we conducted another batch of experiments for somewhat less realistic instances in which the matrices Q and U densities are 90%. The increased densities of the matrices Q and U indicate a prevalence of non-zero coefficients of the bi-linear terms in the primal RDEM (problem (4)). This, in turn, implies that associated RNMDT relaxations (problem (5)) would have a significant number of auxiliary binary variables, thereby potentially increasing the likelihood of the existence of the duality gap when solving corresponding p -Lagrangian dual problems. However, to ensure convergence of p -BnB within one hour, we tested p -BnB on five instances (see Table 4) with RNMDT precision factor $p = -1$ only and remaining parameters as before. Table 5 demonstrates the results of solving instances 1–5 with the proposed p -BnB method employing the FWPH (p -BnB (FWPH)) and proximal BM (p -BnB (BM)) as a subroutine.

Table 5
Numerical results for the instances with high-density Q and U matrices

Instance	p -BnB (FWPH)			p -BnB (BM)		
	sol. time (s)	# nodes	# iter.	sol. time (s)	# nodes	# iter.
1	459.90	5	68	114.42	1	30
2	410.63	3	21	170.53	1	26
3	520.47	5	145	925.73	3	312
4	374.79	3	56	1439.22	5	268
5	427.94	9	191	3001.86	11	1525

The column ‘sol. time’ reports the time required by Algorithm 4 to converge to an optimal solution with a 0.00% gap, calculated as the relative difference between the upper bound (UB) and lower bound (LB) for the objective function generated by the corresponding method. The difference was calculated as $100\% \frac{UB-LB}{LB}$. It is worth highlighting that solving full-scale instances with Gurobi resulted in convergence within one hour only for Instance 1, taking in a total of 2064.55 seconds.

Interestingly, for the setting where the matrices are denser, FWPH outperforms BM as the Lagrangian dual solver in Instances 4 and 5, which is a consequence of FWPH leading the algorithm to explore fewer nodes than BM. In either case, the number of nodes explored is small. As can be seen in Table 5, the maximum number of nodes explored by p -BnB was only 11, for Instance 5 using the proximal BM, while the average number of nodes explored was five. This is because despite the very high density of the quadratic matrices (90%) in the instances, at the very first node, p -BnB was able to generate a solution with a tight dual bound, on average being 0.01%. In comparison, the average dual bound generated within one hour by solving Instances 1–5 with Gurobi was 4.98%. Further details on each individual instance are provided in Appendix B (see Table B1).

5. Conclusions

In this paper, we propose a novel method for solving two-stage stochastic programming problems whose deterministic equivalents are represented by non-convex MIQCQP models. Additionally, we assess the efficiency of this method by considering two alternative algorithms for solving dual sub-problems. The proposed method is named p -BnB and combines a BnB-based algorithm inspired by Carøe and Schultz (1999) with the p -Lagrangian decomposition proposed in Andrade et al. (2022). The p -Lagrangian decomposition method relies on the composition of the mixed-integer-based relaxation of the non-convex MIQCQP problem using the RNMDT (Andrade et al., 2019) and classic Lagrangian relaxation. The construction of a mixed-integer-based relaxation for the primal non-convex MIQCQP problem is essential to ensure the validity of the dual bound associated with the primal problem. The p -Lagrangian decomposition has been demonstrated to outperform the commercial solver Gurobi in terms of computational time required to generate the dual bounds for a primal non-convex MIQCQP problem, whose precision can be controlled by choice of parameters in RNMDT relaxation. However, p -Lagrangian decomposition could not tackle the duality gap arising from the mixed-integer nature of the primal non-convex MIQCQP problems. In contrast, the proposed p -BnB mitigates this issue by ensuring the integrality conditions of the optimal solution via a classic BnB approach. Additionally, following Carøe and

Schultz (1999), the BnB procedure takes place whenever the first-stage variable candidates violate the non-anticipativity constraints. We also evaluated the efficiency of p -BnB by considering two alternative solution methods for dual sub-problems in contrast to employing the classic bundle method as in the p -Lagrangian decomposition (Andrade et al., 2022). We utilised FWPH (Boland et al., 2018) and proximal BM (Kim and Dandurand, 2022) to solve the node sub-problems. The FWPH method is the enhancement of classic progressive hedging (Rockafellar and Wets, 1991) that guarantees convergence even for mixed-integer problems. The proximal BM is an enhancement of a classic BM (Lemar chal, 1974; Zhao and Luh, 2002) involving a new approach for updating the weights of proximal terms that can significantly reduce the number of iterations in the BM necessary to reach the desired convergence tolerance (Kiwiel, 1990). It is important to highlight that the p -BnB method ensures convergence to the solution of the non-convex MIQCQP problem given a predefined tolerance, as opposed to absolute convergence. Nonetheless, the former is generally sufficient for the majority of practical applications.

The p -BnB efficiency has been tested on a set of RNMDT relaxations of randomly generated non-convex MIQCQP instances. Numerical experiments demonstrated the superior performance of the proposed p -BnB method over attempts to solve full-scale RNMDT problems with the commercial solver Gurobi. Depending on the method utilised to solve dual sub-problems, the use of p -BnB allowed for saving on average about 32% of the time required by Gurobi to solve RNMDT problem in case p -BnB used FWPH as a subroutine or about 84% of the time if proximal BM has been used. These results lead us to conclude that, although FWPH does potentially provide better dual bounds, it pays the price for being a more computationally demanding solution method. This trade-off flips when we consider artificially denser instances, in which, we can see that the FWPH method leads to fewer nodes explored in instances 4 and 5, and thus potential computational savings.

It is worth highlighting that the p -BnB method implementation involves software engineering decisions that can greatly influence its performance. Nevertheless, our implementation still serves as a reliable proof of concept. Additionally, the p -BnB method as proposed only considers rudimentary heuristics to generate feasible solutions for the primal RNMDT relaxation and the implementation of more sophisticated heuristics would likely improve the performance of p -BnB, in a similar fashion as they are beneficial in mixed-integer programming solvers. Hence, one could further enhance p -BnB computational efficiency. In particular, one potential path for improvement involves enhancing the branching strategies considered (Chen, 2003; Cornu ejols et al., 2011). Another possible direction could be an improvement of the FWPH method implementation. Additionally, an improvement of the procedure for generating the sets $\{(V_0^s)_s\}_{inS}$ at the beginning of Algorithm 2 could bring new insight into p -BnB performance and convergence rate when using FWPH as a subroutine.

Acknowledgements

The authors gratefully acknowledge the computational resources provided by the Aalto Science-IT project. This research received financial support from the Research Council of Finland (decision number 348092).

Open access publishing facilitated by Aalto-yliopisto, as part of the Wiley - FinELib agreement. [Correction added on 27 May 2025, after first online publication: FinELib 2023 funding statement has been added.]

References

- Aalto Scientific Computing, 2022. Triton cluster, Available at <https://scicomp.aalto.fi/triton/#overview>.
- Achterberg, T., 2007. Constraint integer programming. <https://api-depositonce.tu-berlin.de/server/api/core/bitstreams/a301bdbc-d04d-4fd5-9e04-c0399908689c/content>
- Achterberg, T., Koch, T., Martin, A., 2005. Branching rules revisited. *Operations Research Letters* 33, 1, 42–54. <https://www.sciencedirect.com/science/article/pii/S0167637704000501>.
- Ahmed, S., Garcia, R., Kong, N., Ntamo, L., Parija, G., Qiu, F., Sen, S., 2015. Siplib: a stochastic integer programming test problem library. Available at <https://www2.isye.gatech.edu/~sahmed/siplib>.
- Amos, F., Rönnqvist, M., Gill, G., 1997. Modelling the pooling problem at the New Zealand refining company. *Journal of the Operational Research Society* 48, 8, 767–778.
- Andiappan, V., 2017. State-of-the-art review of mathematical optimisation approaches for synthesis of energy systems. *Process Integration and Optimization for Sustainability* 1, 3, 165–188.
- Andrade, T., Oliveira, F., Hamacher, S., Eberhard, A., 2019. Enhancing the normalized multiparametric disaggregation technique for mixed-integer quadratic programming. *Journal of Global Optimization* 73, 4, 701–722.
- Andrade, T., Belyak, N., Eberhard, A., Hamacher, S., Oliveira, F., 2022. The p-Lagrangian relaxation for separable nonconvex MIQCQP problems. *Journal of Global Optimization* 84, 1, 43–76.
- Arora, S., Bollobás, B., Lovász, L., 2002. Proving integrality gaps without knowing the linear program. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002*. IEEE, Piscataway, NJ, pp. 313–322.
- Bashiri, M., Nikzad, E., Eberhard, A., Hearne, J., Oliveira, F., 2021. A two-stage stochastic programming for asset protection routing and a solution algorithm based on the progressive hedging algorithm. *Omega* 104, 102480.
- Belotti, P., 2023. Couenne: a user's manual. Available at <https://www.coin-or.org>.
- Belyak, N., 2022. p-branch-and-bound method. Available at <https://github.com/gamma-opt/p-BnB>.
- Bénichou, M., Gauthier, J.-M., Girodet, P., Hentges, G., Ribière, G., Vincent, O., 1971. Experiments in mixed-integer linear programming. *Mathematical programming* 1, 76–94.
- Bergamini, M.L., Aguirre, P., Grossmann, I., 2005. Logic-based outer approximation for globally optimal synthesis of process networks. *Computers & Chemical Engineering* 29, 9, 1914–1933. <https://www.sciencedirect.com/science/article/pii/S0098135405001006>.
- Berthold, T., Heinz, S., Vigerske, S., 2012. Extending a CIP framework to solve MIQCPs. In *Mixed Integer Nonlinear Programming*. Springer, Berlin, pp. 427–444.
- Bezanson, J., Edelman, A., Karpinski, S., Shah, V.B., 2017. Julia: a fresh approach to numerical computing. *SIAM Review* 59, 1, 65–98. ISSN 0036-1445, 1095-7200.
- Boland, N., Christiansen, J., Dandurand, B., Eberhard, A., Linderoth, J., Luedtke, J., Oliveira, F., 2018. Combining progressive hedging with a Frank–Wolfe method to compute Lagrangian dual bounds in stochastic mixed-integer programming. *SIAM Journal of Optimization* 28, 2, 1312–1336.
- Boland, N., Christiansen, J., Dandurand, B., Eberhard, A., Oliveira, F., 2019. A parallelizable augmented Lagrangian method applied to large-scale non-convex-constrained optimization problems. *Mathematical Programming* 175, 503–536.
- Bragalli, C., D'Ambrosio, C., Lee, J., Lodi, A., Toth, P., 2012. On the optimal design of water distribution networks: a practical MINLP approach. *Optimization and Engineering* 13, 2, 219–246.
- Bush, R.R., Mosteller, F., 1953. A stochastic model with applications to learning. *The Annals of Mathematical Statistics* 24, 4, 559–585.
- Calvo, R.W., de Luigi, F., Haastrup, P., Maniezzo, V., 2004. A distributed geographic information system for the daily car pooling problem. *Computers & Operations Research* 31, 13, 2263–2278.
- Carøe, C.C., Schultz, R., 1999. Dual decomposition in stochastic integer programming. *Operations Research Letters* 24, 1–2, 37–45. <https://linkinghub.elsevier.com/retrieve/pii/S0167637798000509>.
- Castillo, I., Westerlund, J., Emet, S., Westerlund, T., 2005. Optimization of block layout design problems with unequal areas: a comparison of MILP and MINLP optimization methods. *Computers & Chemical Engineering* 30, 1, 54–69.
- Castro, P.M., 2016a. Normalized multiparametric disaggregation: an efficient relaxation for mixed-integer bilinear problems. *Journal of Global Optimization* 64, 4, 765–784.

- Castro, P.M., 2016b. Spatial branch and bound algorithm for the global optimization of MIQCPS. *Computer Aided Chemical Engineering* 38, 523–528.
- Chen, X.-W., 2003. An improved branch and bound algorithm for feature selection. *Pattern Recognition Letters* 24, 12, 1925–1933. ISSN 0167-8655. URL <https://www.sciencedirect.com/science/article/pii/S0167865503000205>.
- Cornuéjols, G., Liberti, L., Nannicini, G., 2011. Improved strategies for branching on general disjunctions. *Mathematical Programming* 130, 2, 225–247.
- Cui, X.T., Zheng, X.J., Zhu, S.S., Sun, X.L., 2013. Convex relaxations and MIQCQP reformulations for a class of cardinality-constrained portfolio selection problems. *Journal of Global Optimization* 56, 4, 1409–1423.
- de Oliveira, W., Sagastizábal, C., 2014. Bundle methods in the XXIst century: a bird's-eye view. *Pesquisa Operacional* 34, 647–670.
- Ding, T., Bo, R., Li, F., Sun, H., 2014. A bi-level branch and bound method for economic dispatch with disjoint prohibited zones considering network losses. *IEEE Transactions on Power Systems* 30, 6, 2841–2855.
- Feltenmark, S., Kiwiel, K.C., 2000. Dual applications of proximal bundle methods, including Lagrangian relaxation of nonconvex problems. *SIAM Journal on Optimization* 10, 3, 697–721.
- Fischetti, M., Monaci, M., 2011. Backdoor branching. In Günlük, O., Woeginger, G.J. (eds), *Integer Programming and Combinatorial Optimization*. Springer, Berlin, pp. 183–191.
- Fischetti, M., Salvagnin, D., 2010. Pruning moves. *INFORMS Journal on Computing* 22, 1, 108–119.
- Forrest, J.J.H., Hirst, J.P.H., Tomlin, J.A., 1974. Practical solution of large mixed integer programming problems with Umpire. *Management Science* 20, 5, 736–773.
- Gilpin, A., Sandholm, T., 2011. Information-theoretic approaches to branching in search. *Discrete Optimization* 8, 2, 147–159. <https://www.sciencedirect.com/science/article/pii/S1572528610000423>.
- Grossmann, I.E., Harjunkoski, I., 2019. Process systems engineering: academic and industrial perspectives. *Computers & Chemical Engineering* 126, 474–484. <https://www.sciencedirect.com/science/article/pii/S009813541930078X>.
- Jezowski, J., 2010. Review of water network design methods with literature annotations. *Industrial & Engineering Chemistry Research* 49, 10, 4475–4516.
- Kallrath, J., 2021. Optimization: using models, validating models, solutions, answers. In Kallrath, J. (ed.) *Business Optimization Using Mathematical Programming: An Introduction with Case Studies and Solutions in Various Algebraic Modeling Languages*, International Series in Operations Research & Management Science. Springer International Publishing, Cham, pp. 1–32.
- Kim, K., Dandurand, B., 2022. Scalable branching on dual decomposition of stochastic mixed-integer programming problems. *Mathematical Programming Computation* 14, 1, 1–41.
- Kiwiel, K.C., 1984. An algorithm for linearly constrained convex nondifferentiable minimization problems. *Journal of Mathematical Analysis and Applications* 105, 2, 452–465.
- Kiwiel, K.C., 1985. An exact penalty function algorithm for non-smooth convex-constrained minimization problems. *IMA Journal of Numerical Analysis* 5, 1, 111–119.
- Kiwiel, K.C., 1987. A constraint linearization method for nondifferentiable convex minimization. *Numerische Mathematik* 51, 395–414.
- Kiwiel, K.C., 1990. Proximity control in bundle methods for convex nondifferentiable minimization. *Mathematical Programming* 46, 1-3, 105–122.
- Kocis, G.R., Grossmann, I.E., 1987. Relaxation strategy for the structural optimization of process flow sheets. *Industrial & Engineering Chemistry Research* 26, 9, 1869–1880.
- Küçükyavuz, S., Sen, S., 2017. An introduction to two-stage stochastic mixed-integer programming. In *Leading Developments from INFORMS Communities*. INFORMS, Catonsville, MD, pp. 1–27.
- Lee, J., Leyffer, S., December, 2011. *Mixed Integer Nonlinear Programming*. Springer Science & Business Media, Berlin.
- Lemaréchal, C., 1974. An algorithm for minimizing convex functions. In Rosenfeld, J. L. (ed.) *Information Processing, Proceedings of the 6th IFIP Congress*. North-Holland, Stockholm, Sweden, pp. 552–556.
- LLC The Optimization Firm, 2019. Baron User Manual v. 2019.12.7. <https://www.minlp.com>.
- LLC Gurobi Optimization, 2020. Gurobi optimizer reference manual. Available at <http://www.gurobi.com>.
- Liao, S., Yao, W., Han, X., Fang, J., Ai, X., Wen, J., He, H., 2019. An improved two-stage optimization for network and load recovery during power system restoration. *Applied Energy* 249, 265–275. <https://www.sciencedirect.com/science/article/pii/S0306261919308359>.

- Linderroth, J.T., Savelsbergh, M.W.P., 1999. A computational study of search strategies for mixed integer programming. *INFORMS Journal on Computing* 11, 2, 173–187.
- McCormick, G.P., 1976. Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems. *Mathematical Programming* 10, 1, 147–175.
- Misener, R., Floudas, C.A., 2012. Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCQP) through piecewise-linear and edge-concave relaxations. *Mathematical Programming* 136, 1, 155–182.
- Morrison, D.R., Jacobson, S.H., Sauppe, J.J., Sewell, E.C., 2016. Branch-and-bound algorithms: a survey of recent advances in searching, branching, and pruning. *Discrete Optimization* 19, 79–102. <https://www.sciencedirect.com/science/article/pii/S1572528616000062>.
- Murtagh, B.A., Saunders, M.A., 1995. Minos 5.4 user's guide (revised). Technical Report SOL 83-20R, Department of Operations Research, Stanford University, Stanford, CA.
- Ortega, F., Wolsey, L.A., 2003. A branch-and-cut algorithm for the single-commodity, uncapacitated, fixed-charge network flow problem. *Networks: An International Journal* 41, 3, 143–158.
- Palani, A.M., Wu, H., Morcos, M.M., 2019. A Frank–Wolfe progressive hedging algorithm for improved lower bounds in stochastic SCUC. *IEEE Access* 7, 99398–99406.
- Rockafellar, R.T., Wets, R.J.-B., 1991. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research* 16, 1, 119–147.
- Sewell, E.C., Sauppe, J.J., Morrison, D.R., Jacobson, S.H., Kao, G.K., 2012. A BB&R algorithm for minimizing total tardiness on a single machine with sequence-dependent setup times. *Journal of Global Optimization* 54, 4, 791–812.
- Vilà, M., Pereira, J., 2014. A branch-and-bound algorithm for assembly line worker assignment and balancing problems. *Computers & Operations Research* 44, 105–114. <https://www.sciencedirect.com/science/article/pii/S0305054813003110>.
- Virasjoki, V., Siddiqui, A.S., Oliveira, F., Salo, A., 2020. Utility-scale energy storage in an imperfectly competitive power sector. *Energy Economics* 88, 104716.
- Wang, T., Meng, Q., Wang, S., Qu, X., 2021. A two-stage stochastic nonlinear integer-programming model for slot allocation of a liner container shipping service. *Transportation Research Part B: Methodological* 150, 143–160. <https://www.sciencedirect.com/science/article/pii/S0191261521000795>.
- Watson, J.-P., Woodruff, D.L., 2011. Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems. *Computational Management Science* 8, 4, 355–370.
- Zhao, X., Luh, P.B., 2002. New bundle methods for solving Lagrangian relaxation dual problems. *Journal of Optimization Theory and Applications* 113, 2, 373–397. <http://link.springer.com/10.1023/A:1014839227049>.

Appendix A

Table A1

Range of the parameters' values for RDEM problem. The values utilised are uniformly sampled from these ranges.

Parameter	Range
Q	[−100, 100]
c	[0, 100]
q	[−100, 100]
U	[0, 1000]
T	[0, 100]
W	[−100, 100]
h	[1000, 100000]
X	[0, 10]
Y	[0, 10]

Appendix B

Table B1

Average dual bound generated within one hour by solving the Instances 1-5 with Gurobi solver

Instance	Dual bound (%)
1	0.00
2	4.71
3	6.52
4	8.54
5	5.11

Appendix C

This section presents an illustrative example showcasing the application of Algorithm 4. Consider Problem (4) with two scenarios $S = \{s_1, s_2\}$, two first-stage variables x_1 and x_2 , one second-stage variable y and one constraint defined per scenario. Then, Problem (4) can be formulated as follows:

$$z^{\text{SMIP}} = \min_{x_1^1, x_1^2, x_2^1, x_2^2, y^1, y^2} -16.8y^1y^1 - 9.8y^1 \quad (\text{C1})$$

$$-4.5y^2y^2 - 7.9y^2 \quad (\text{C2})$$

$$-53.2x_1^1 - 23.2x_2^1 - 9.0x_1^2 - 3.9x_2^2 \quad (\text{C3})$$

s.t.:

$$78.1x_1^1 + 67.0x_2^1 + 16.8y^1 \leq 57108.7 \quad (\text{C4})$$

$$77.0y^2y^2 + 45.3x_1^2 + 30.2x_2^2 + 0.1y^2 \leq 56702.4 \quad (\text{C5})$$

$$x_1^1 - \hat{x}_1 = 0 \quad (\text{C6})$$

Table C1

Result of applying Algorithm 1 and Algorithm 2 to Problem (C17)

Number of iterations	1
Objective function value	$z_{LB} = -2299.62$
First-stage variables' values	$x_1^1 = 5.0, x_2^1 = 7.0, x_1^2 = 5.0, x_2^2 = 7.0$
Second-stage variables' values	$y^1 = 9.0, y^2 = 8.0$

$$x_1^2 - \hat{x}_1 = 0 \quad (C7)$$

$$x_2^1 - \hat{x}_2 = 0 \quad (C8)$$

$$x_2^2 - \hat{x}_2 = 0 \quad (C9)$$

$$0.0 \leq x_1^1, x_1^2 \leq 5.0 \quad (C10)$$

$$0.0 \leq x_2^1, x_2^2 \leq 7.0 \quad (C11)$$

$$0.0 \leq y^1 \leq 9.0 \quad (C12)$$

$$0.0 \leq y^2 \leq 8.0 \quad (C13)$$

$$x_1^1, x_1^2, x_2^1, x_2^2 \in \mathbb{Z} \quad (C14)$$

where (C6)–(C9) are non-anticipativity conditions.

Applying p -Lagrangian relaxation, we get Problem (6) with two sub-problems (C15) and (C16)

$$\begin{aligned} L^1(\mu^1) = \min_{\omega^1} & \frac{1}{\pi^1} (-16.8w^1 - 53.3x_1^1 - 23.2x_2^1 - 9.8y^1 + \mu_1^1(x_1^1 - \hat{x}_1) + \mu_2^1(x_2^1 - \hat{x}_2)) \\ \text{s.t.} & \\ & y^1 - 9\Delta y^1 - 4.5z_{-1}^1 = 0 \\ & -4.5\hat{y}_{-1}^1 + w^1 - 9\Delta w^1 = 0 \\ & 78.1x_1^1 + 67.0x_2^1 + 16.8y^1 \leq 57108.7 \\ & 0.5y^1 + 9\Delta y^1 - \Delta w^1 \leq 4.5 \\ & -0.5y^1 + \Delta w^1 \leq 0 \\ & -\Delta w^1 \leq 0 \\ & -9\Delta y^1 + \Delta w^1 \leq 0 \\ & -\hat{y}_{-1}^1 \leq 0 \\ & \hat{y}_{-1}^1 - 9z_{-1}^1 \leq 0 \\ & -y^1 + \hat{y}_{-1}^1 \leq 0 \\ & y^1 - \hat{y}_{-1}^1 + 9z_{-1}^1 \leq 9.0 \\ & 0.0 \leq x_1^1 \leq 5.0 \end{aligned}$$

$$\begin{aligned}
0.0 &\leq x_2^1 \leq 7.0 \\
0.0 &\leq y^1 \leq 9.0 \\
0.0 &\leq \Delta y^1 \leq 0.5 \\
x_1^1, x_2^1 &\in \mathbb{Z} \\
z_{-1}^1 &\in \{0, 1\}
\end{aligned} \tag{C15}$$

$$\begin{aligned}
L^2(\mu^2) &= \min_{\omega^2} \frac{1}{\pi^2} (-4.5w^2 - 9.0x_1^2 - 3.9x_2^2 - 7.9y^2 + \mu_1^2(x_1^2 - \hat{x}_1) + \mu_1^2(x_2^2 - \hat{x}_2)) \\
&\text{s.t.} \\
&y^2 - 8\Delta y^2 - 4z_{-1}^2 = 0 \\
&-4\hat{y}_{-1}^2 + w^2 - 8\Delta w^2 = 0 \\
&45.3x_1^2 + 30.2x_2^2 + 0.1y^2 + 77w^2 \leq 56702.4 \\
&0.5y^2 + 8\Delta y^2 - \Delta w^2 \leq 4.0 \\
&-0.5y^2 + \Delta w^2 \leq 0 \\
&-\Delta w^2 \leq 0 \\
&-8\Delta y^2 + \Delta w^2 \leq 0 \\
&-\hat{y}_{-1}^2 \leq 0 \\
&\hat{y}_{-1}^2 - 8z_{-1}^2 \leq 0 \\
&-y^2 + \hat{y}_{-1}^2 \leq 0 \\
&y^2 - \hat{y}_{-1}^2 + 8z_{-1}^2 \leq 8.0 \\
&0.0 \leq x_1^2 \leq 5.0 \\
&0.0 \leq x_2^2 \leq 7.0 \\
&0.0 \leq y^2 \leq 8.0 \\
&0.0 \leq \Delta y^2 \leq 0.5 \\
&x_1^2, x_2^2 \in \mathbb{Z} \\
&z_{-1}^2 \in \{0, 1\}.
\end{aligned} \tag{C16}$$

The terms π^1 and π^2 correspond to the probabilities of Problems (C15) and (C16), respectively, and

$$\omega^1 = \{x_1^1, x_2^1, \hat{x}_1, \hat{x}_2, y^1, \Delta y^1, \Delta w^1, w^1, \hat{y}_{-1}^1, z_{-1}^1\},$$

and

$$\omega^2 = \{x_1^2, x_2^2, \hat{x}_1, \hat{x}_2, y^2, \Delta y^2, \Delta w^2, w^2, \hat{y}_{-1}^2, z_{-1}^2\}.$$

Then the first root node of Algorithm 4 will be

$$z_{LD} = \max_{\mu^1, \mu^2} \{\pi^1 L^1(\mu^1) + \pi^2 L^2(\mu^2) : \pi^1 \mu^1 + \pi^2 \mu^2 = 0\}. \quad (C17)$$

Applying Algorithm 1 to solve Problem (C17) yields the solution in just one iteration due to the simple structure and small scale of the problem. A similar result occurs when applying Algorithm 2, which also converges to the solution z_{LB} in one iteration. The solution is presented in Table C1.

The average solution is given by $\hat{x} = \pi^1 x^1 + \pi^2 x^2$, resulting in $\{5, 7\}$. Due to the simplicity of the problem, both (1) and Algorithm 2 managed to yield a solution at the root node that satisfies both the non-anticipativity and integrality conditions ($x_1^1 = x_1^2 = 5.0$ and $x_2^1 = x_2^2 = 7.0$). However, in the general case, there is no guarantee that this will occur. Therefore, for the sake of illustration, let us assume that the solution to Problem (C17) violates one of the conditions, considering $\pi^1 = \pi^2 = 0.5$.

First, let us assume that $x_1^1 = 4$, $x_1^2 = 5$, and $x_2^1 = x_2^2 = 7$. In this case, the average solution is $\hat{x} = \{4.5, 7\}$. We can observe that the first component takes the value 4.5, which is not integer-valued. In this situation, following Algorithm 4, we would perform branching based on the violation of the integrality condition. Specifically, we would generate two child nodes N_1 and N_2 , such that N_1 contains Problem (C17) with the additional constraints $\{x_1^1, x_1^2 \leq 4\}$, while N_2 contains Problem (C17) with the constraints $\{x_1^1, x_1^2 \geq 5\}$.

Alternatively, let us assume that $x_1^1 = 4$, $x_1^2 = 6$, and $x_2^1 = x_2^2 = 7$. In this case, the average solution is $\hat{x} = \{5, 7\}$. We can see that the solution $\hat{x}$ satisfies the integrality condition but does not satisfy the non-anticipativity condition for the first component. Specifically, the dispersion is $\sigma_1 = x_1^2 - x_1^1 = 2$. In this case, following Algorithm 4, we would perform branching based on the violation of the non-anticipativity condition. We would create two child nodes N_1 and N_2 , where the subproblem for N_1 corresponds to Problem (C17) with the constraints $\{x_1^1, x_1^2 \leq 5 - \epsilon_{BB}\}$, and N_2 would correspond to Problem (C17) with the constraints $\{x_1^1, x_1^2 \geq 5 + \epsilon_{BB}\}$.