On Column Generation Methods for Nonlinear Optimization

Renaud Chicoisne

Mathematical Programming Series A manuscript No.
(will be inserted by the editor)

On Column Generation Methods for Nonlinear Optimization

Renaud Chicoisne

Received: date / Accepted: date

Abstract Solving large scale nonlinear optimization problems requires either significant computing resources or the development of specialized algorithms. In the case of Linear Programming (LP) problems there are decomposition methods that take advantage of problem structure, gradually constructing the full problem by generating variables or constraints. There are no direct adaptations of Column Generation (CG) methods for nonlinear optimization problems. In this work we present a generic CG method for nonlinear optimization problems, such that when optimizing over a structured set $\mathcal{X}$ plus a moderate number of complicating constraints, we solve a succession of 1) restricted *master* problems on a smaller set $\mathcal{S} \subset \mathcal{X}$ and 2) *pricing* problems that are Lagrangean relaxations with respect to the complicating constraints. The former provides feasible solutions and feeds dual information to the latter. In turn, the pricing problem identifies a variable of interest that is then taken into account into an updated subset $\mathcal{S}' \subset \mathcal{X}$. We show that our approach is valid not only when $\mathcal{S}$ is the convex hull of the generated variables as in CG for LPs, but also with a variety of subsets such as the cone hull, the linear span, and a special variable aggregation set. We discuss how the structure of the set we choose and the way it is updated influences the number of iterations required to reach optimality or near-optimality and the difficulty of solving the restricted master problems. We present linearized schemes that alleviate the computational burden of solving the pricing problem. We test our methods on Portfolio Optimization instances with up to 5 million variables including nonlinear objective functions and Second Order Cone (SOC) constraints. We show that some CGs with linearized pricing are 2-3 times faster than solving the complete problem directly. Further, for the larger instances the same CG methods are able to provide solutions within 1% of optimality in 6 hours whereas solving the complete problem runs out of memory.

R. Chicoisne
Université Libre de Bruxelles
Boulevard du Triomphe, Brussels, Belgium

E-mail: renaud.chicoisne@gmail.com

Keywords Nonlinear Optimization · Conic Optimization · Decomposition Methods · Lagrangean Duality · Dantzig-Wolfe · Portfolio Optimization

1 Introduction

Consider the following general problem $P(\mathcal{X})$

$$(P(\mathcal{X})) \quad \omega(\mathcal{X}) := \min_{x\in\mathcal{X},y} \{f(x,y) : -g(x,y) \in \mathcal{C}\}$$

where $\mathcal{C}$ is a cone in some Euclidean space. f and g are generic nonlinear mappings that satisfy some convexity properties in Section 3 only. Let us define n, n_0 and m as the respective dimensions of x, y and $\mathcal{C}$. We assume that the main issue with $P(\mathcal{X})$ is the presence of the constraints $-g(x,y) \in \mathcal{C}$ and the magnitude of $n \gg 1$. This means that we are considering a setting in which the unconstrained problem with a subset of variables x can be solved efficiently and we propose to use these simpler optimization problems to build a solution method for $P(\mathcal{X})$.

Decomposition methods are fundamental tools to solve difficult large scale problems. In this work, we focus on Column Generation (CG or primal decomposition) algorithms, where the number of variables is too large to allow a direct solution with an off-the-shelf optimization software. Similiar approaches for constraint generation [5] can be developed and are the topic of future research.

1.1 Preliminaries

We first introduce some important definitions and assumptions.

Definition 1 We call the *dual cone* of a given cone $\mathcal{K}$, the set $\mathcal{K}^*$ defined as

$$\mathcal{K}^* := \{u : \langle u, v\rangle \geqslant 0, \forall v \in \mathcal{K}\}.$$

For any penalization vector $\lambda \in \mathcal{C}^*$, let us define the following *Lagrangean relaxation*:

$$(L(\mathcal{X},\lambda)) \quad \omega(\mathcal{X},\lambda) := \min_{x\in\mathcal{X},y} \{f(x,y) + \langle \lambda, g(x,y)\rangle\}.$$

Notice that for any $\lambda \in \mathcal{C}^*$ we always have $\omega(\mathcal{X},\lambda) \leqslant \omega(\mathcal{X})$. We assume throughout this document that $L(\mathcal{X},\lambda)$ is easily solvable for any $\lambda \in \mathcal{C}^*$.

Assumption 1 *$P(\mathcal{X})$ has no Lagrangean duality gap, i.e. $P(\mathcal{X})$ has the same optimal value as its Lagrangean dual $D(\mathcal{X})$:*

$$(D(\mathcal{X})) \quad \omega(\mathcal{X}) := \max_{\lambda\in\mathcal{C}^*} \omega(\mathcal{X},\lambda).$$

Assumption 1 implies that if an optimal dual vector λ^* is available, so is the optimal value $\omega(\mathcal{X})$. Further, we can see that an optimal solution of $P(\mathcal{X})$ is also optimal for $L(\mathcal{X},\lambda^*)$. Now that we can bound $\omega(\mathcal{X})$ from below, we wish to bound it from above: Given a set $\mathcal{S} \subseteq \mathcal{X}$, let us define the following *restricted problem*:

$$(P(\mathcal{S})) \quad \omega(\mathcal{S}) := \min_{x\in\mathcal{S},y} \{f(x,y) : -g(x,y) \in \mathcal{C}\}.$$

Given that the original problem $P(\mathcal{X})$ is a relaxation of $P(\mathcal{S})$ for any $\mathcal{S} \subseteq \mathcal{X}$, we have $\omega(\mathcal{X}) \leqslant \omega(\mathcal{S})$. Remark that if $\mathcal{S}$ contains an optimal solution of $P(\mathcal{X})$ then we have $\omega(\mathcal{X}) = \omega(\mathcal{S})$ and that $P(\mathcal{S})$ returns feasible solutions for $P(\mathcal{X})$. We wish $\mathcal{S}$ to be small or structured enough so that $P(\mathcal{S})$ is computationally tractable.

Assumption 2 *Problem $P(\mathcal{S})$ maintains a zero Lagrangean duality gap and is solved by an algorithm that provides optimal Lagrange dual variables λ.*

Our objective is twofold: design an iterative search in terms of both λ and $\mathcal{S}$ that successively improves the lower and upper bounds with increasingly good feasible solutions as a byproduct. Our framework achieves this goal by feeding information from one problem to the other by updating respectively λ from $P(\mathcal{S})$ and $\mathcal{S}$ from $L(\mathcal{X}, \lambda)$.

1.2 Dantzig-Wolfe for LPs

To illustrate our point, let us take the following LP as a special case of $P(\mathcal{X})$:

$$\min_{x,y} c^\top x + d^\top y \tag{1a}$$
$$\text{s.t.: } x \in \mathcal{X} := \{x \geqslant 0 : Ax = a\} \tag{1b}$$
$$Xx + Yy \geqslant b, \tag{1c}$$

and its LP dual

$$\max_{\lambda,\pi} a^\top \pi + b^\top \lambda \tag{2a}$$
$$\text{s.t.: } \lambda \in \Lambda := \left\{\lambda \geqslant 0 : Y^\top \lambda = d\right\} \tag{2b}$$
$$A^\top \pi + X^\top \lambda \leqslant c, \tag{2c}$$

where π and $\lambda \geqslant 0$ are the dual variables associated to constraints (1b) and (1c) respectively. As pointed out before, if we were to replace $\mathcal{X}$ by some wisely chosen subset $\mathcal{S} \subset \mathcal{X}$ in (1), we would obtain a cheap upper bound for the optimal value of (1). In this LP case, there is a natural choice for $\mathcal{S}$ readily available [7]: Letting $\mathcal{V}$ be the set of vertices of $\mathcal{X}$ and $\mathcal{R}$ a complete set of extreme rays of $\mathcal{X}$ we have $\mathcal{X} := \text{conv.hull}(\mathcal{V}) + \text{cone.hull}(\mathcal{R})$. In other words:

$$x \in \mathcal{X} \Leftrightarrow \exists \theta \geqslant 0 : \begin{cases} \sum\limits_{l:\bar{x}^l \in \mathcal{V}} \theta_l = 1 \\ x = \sum\limits_{l:\bar{x}^l \in \mathcal{V}} \theta_l \bar{x}^l + \sum\limits_{l:\bar{x}^l \in \mathcal{R}} \theta_l \bar{x}^l. \end{cases}$$

Problem (1) can thus be rewritten as the following *extensive formulation*:

$$\min_{\theta,y} \sum_{l:\bar{x}^l \in \mathcal{V}} \theta_l c^\top \bar{x}^l + \sum_{l:\bar{x}^l \in \mathcal{R}} \theta_l c^\top \bar{x}^l + d^\top y \tag{3a}$$
$$\text{s.t.: } \sum_{l:\bar{x}^l \in \mathcal{V}} \theta_l X \bar{x}^l + \sum_{l:\bar{x}^l \in \mathcal{R}} \theta_l X \bar{x}^l + Yy \geqslant b \tag{3b}$$
$$\sum_{l:\bar{x}^l \in \mathcal{V}} \theta_l = 1 \tag{3c}$$
$$\theta \geqslant 0, \tag{3d}$$

whose LP dual is the following problem:

$$\max_{\lambda,\eta} \; b^\top \lambda + \eta \tag{4a}$$

$$\text{s.t.:} \; \left(X\bar{x}^l\right)^\top \lambda \leqslant c^\top \bar{x}^l - \eta \qquad ,\forall l : \bar{x}^l \in \mathcal{V} \tag{4b}$$

$$\left(X\bar{x}^l\right)^\top \lambda \leqslant c^\top \bar{x}^l \qquad ,\forall l : \bar{x}^l \in \mathcal{R} \tag{4c}$$

$$\lambda \in \Lambda. \tag{4d}$$

A direct solution of problem (3) is in general impractical as its number of variables can be exponential in (n_1, n_2, m). However, the *Dantzig-Wolfe* (DW) algorithm [13] offers a solution method successively generating vertices and extreme rays of the polyhedron $\mathcal{X}$. It starts with initial subsets $\overline{\mathcal{V}} \subset \mathcal{V}$ and $\overline{\mathcal{R}} \subset \mathcal{R}$ and solves a restricted master problem (3) with $\overline{\mathcal{V}}$ and $\overline{\mathcal{R}}$ instead of the full sets $\mathcal{V}$ and $\mathcal{R}$. With our notation, this restricted master problem is none other than $P(\mathcal{S})$ with $\mathcal{S} :=$ conv.hull $\left(\overline{\mathcal{V}}\right)$+cone.hull $\left(\overline{\mathcal{R}}\right)$. Making different choices for $\mathcal{S}$ and consider a broader class of optimization problems is the central idea of this paper. Obtaining the optimal dual variables λ associated with constraints (3b) in $P(\mathcal{S})$, the Lagrangean relaxation $L(\mathcal{X}, \lambda)$ is solved

$$\min_{x,y} \left\{ c^\top x + d^\top y + \lambda^\top (b - Xx - Yy) : x \in \mathcal{X} \right\}.$$

It can be rewritten

$$\begin{aligned} &\lambda^\top b + \min_{x\in\mathcal{X}} \left\{ \left(c - X^\top \lambda\right)^\top x \right\} + \min_{y} \left\{ \left(d - Y^\top \lambda\right)^\top y \right\} \\ =&\lambda^\top b + \min_{x\in\mathcal{X}} \left\{ \left(c - X^\top \lambda\right)^\top x \right\} \end{aligned} \tag{5}$$

by dual feasibility (4d) of λ. This means that we have $Y^\top \lambda = d$ and we can eliminate the variables y from the pricing problem. Discarding this dependency in y is however not always possible in a nonlinear setting. Let $\bar{x}$ be an optimal solution of the pricing problem (5). With a slight abuse of notation, we refer to an *optimal solution* to either a vertex of $\mathcal{X}$ if the pricing problem in x has a bounded optimal objective value, or an extreme ray of $\mathcal{X}$ if the pricing problem in x is unbounded. In the latter case, we increment $\bar{\mathcal{R}} \leftarrow \bar{\mathcal{R}} \cup \{\bar{x}\}$ and in the former case, we increment $\bar{\mathcal{V}} \leftarrow \bar{\mathcal{V}} \cup \{\bar{x}\}$, which defines the particular update mechanism used by DW. We iterate until an optimality tolerance criterion is satisfied or until we generated the complete sets $\mathcal{V}$ and $\mathcal{R}$.

1.3 Decomposition methods and previous work

CG algorithms were studied in depth for LPs [13,27] or Mixed Integer Linear Programming (MILP) problems [3,35] where they were able to successfully solve notoriously large MILPs [14,12]. The main idea of CG for LPs - i.e. when $\mathcal{X}$ is a polyhedron, f and g are linear mappings and $\mathcal{C}$ is the non-negative orthant - is to exploit the structure of $\mathcal{X}$ and solve smaller problems: 1) the master problem, that works over a reduced subset $\mathcal{S} \subset \mathcal{X}$ while keeping the side constraints; and 2)

a pricing problem that is still large but is computationally easy to solve because of the absence of the side constraints. They feed information to each other until some stopping criterion is reached and the current solution of the master problem is reasonably close to an optimal solution. The "Column Generation" terminology comes from the fact that the pricing problem generates a vector of variables $x \in \mathcal{X}$ that represents a column in an LP setting.

In a nonlinear setting, several algorithms such as the Alternating Direction Method of Multipliers (ADMM) [43], the Douglas-Rachford splitting operator [16] or Augmented Lagrangean algorithms [33] all make use of a special structure of $\mathcal{X}$. However, they all solve the so-called Lagrangean dual problem and do not always provide explicit solutions for $P(\mathcal{X})$. Further, proving optimality or near optimality can be tricky and a concrete stopping criterion is also not always available. Closer to a generalization of CG for nonlinear problems, the convex simplex method [44] minimizes a nonlinear objective over a polyhedron. It can be seen as solving a master problem over a *basis* of variables, and similar to the simplex algorithm, selects the entering variable by linearizing some penalization function. Akin to CG for LPs, the simplicial decomposition [40] solves a *linearized* master problem over a subset $\mathcal{S}$ that is the convex hull of a handful of points, and the pricing problems that generates such columns is the original problem with an objective linearized at an incumbent point. Other algorithms using inner approximations of $\mathcal{X}$ in a non linear setting were introduced in [18,19] and references therein.

In a Linear Conic Programming (LC) setting, however, more similar extensions of the CG principle have been developed: In [1], they present a decomposition procedure for Semidefinite Programming (SDP) problems. More precisely, they solve master problems that are inner approximations of a matrix set $\mathcal{X}$ that they update with the (matricial) "columns" generated by a separation problem tailored for SDPs. The approach has two drawbacks: 1) the master problem can be slow to reach a near optimal solution depending on the inner approximation chosen and 2) the pricing problem is a hand-made separation problem coming from problem-specific considerations and does not provide dual bounds. For SDPs, chordal sparsity patterns [39] are able to detect an underlying substructure which can be exploited by ADMM [46,45,38], but no CG approach has been attempted so far. The presence of a special substructure being crucial for decomposition, several works present automatic structure detection in LP [25,6,42] so that a decomposition method can make use of it. Previous decomposition methods for LC have focused on gradually building the set of variables considered with problem specific algorithms that are difficult to generalize. In this work we introduce a generic CG method that provides upper and lower bounds at every iteration.

Other works use a different kind of set $\mathcal{S}$ in LP. A subset $\mathcal{S}$ consisting on forcing clusters of variables to share the same value - thus aggregating the variables together is used in [29,8]. This variable aggregation principle has been successfully applied to Freight routing [34], general extended formulations [35], open pit mining scheduling [8] or pricing problems [2].

We list the contributions of this work as follows: 1) We provide generic CG algorithms that are valid for nonlinear optimization problems, provide feasible solutions, lower and upper bounds over time and include a wide range of subsets $\mathcal{S}$ as special cases. 2) As the Lagrangean relaxation of a nonlinear optimization problem can be as hard as the non-relaxed problem, we show that solving the Lagrangean relaxation with a linearized objective maintains the validity of the

general scheme while alleviating the burden of the pricing. 3) We provide numerical results on large scale nonlinear Portfolio Optimization problems showing that some of our linearized schemes run faster than solving the problem straightforwardly, and are able to find good quality solutions for the largest instances, whereas all other algorithms run out of memory.

1.4 Article outline

We first present in Section 2 a set-based primal decomposition algorithm to solve the generic *nonlinear conic* (NLC) problem $P(\mathcal{X})$ with a large number of variables admitting several existing schemes as special cases and discuss conditions to drop the optimization in the y variables for the pricing problem. Under some convexity assumptions, we present in Section 3 a linearized version of the methodology making $LR(\mathcal{X}, \lambda)$ easier to solve. Additionally, we also prove that $LR(\mathcal{X}, \lambda)$ can *always* be independent of the secondary variables y in the linearized algorithm. In Section 4, we point out the relationships of our generic schemes to existing frameworks. We present computational results in Section 5 and conclude with some remarks and the description of several ongoing works in Section 6.

2 A primal algorithm

let us define the Lagrangean relaxation of the restricted problem:

$$(L(\mathcal{S}, \lambda)) \quad \omega(\mathcal{S}, \lambda) := \min_{x \in \mathcal{S}, y} \{f(x, y) + \langle \lambda, g(x, y) \rangle\}.$$

By definition we have $\omega(\mathcal{X}, \lambda) \leqslant \omega(\mathcal{S}, \lambda) \leqslant \omega(\mathcal{S})$. Let us define the Lagrangean dual $D(\mathcal{S})$ of the restricted problem, whose optimal value is $\omega(\mathcal{S})$ from Assumption 2:

$$(D(\mathcal{S})) \quad \omega(\mathcal{S}) := \max_{\lambda \in \mathcal{C}^*} \quad \omega(\mathcal{S}, \lambda).$$

Instead of using specific forms of feeding the pricing information to the restricted problem, we use a generic set $\mathcal{S}$ that is updated at each iteration as described in Algorithm 1. We summarize in Figure 1 the relationships between the bounds of the problems involved in Algorithm 1. For the remaining of this paper, in all the bound relationship graphs an edge $a \to b$ means that $a \leqslant b$, the gray edges are the nontrivial relationships that apply at a stopping criterion and the gray nodes are the optimal values of the problems solved during the course of the algorithm.

Proposition 1 *At termination, Algorithm 1 returns an optimal solution for $P(\mathcal{X})$.*

Proof If Algorithm 1 terminates because of the stopping criterion at line 8, we have $\bar{x}^k \in \mathcal{S}^k$. Then, $(\bar{x}^k, \bar{y}^k)$ is also feasible and optimal for $L(\mathcal{S}^k, \lambda^k)$:

$$\omega\left(\mathcal{S}^k, \lambda^k\right) := \min_{x \in \mathcal{S}^k, y} \left\{ f(x, y) + \left\langle \lambda^k, g(x, y) \right\rangle \right\}$$

In consequence, we have that $\omega(\mathcal{X}, \lambda^k) = \omega(\mathcal{S}^k, \lambda^k)$. Recalling that (x^k, y^k) is an optimal solution for $P(\mathcal{S}^k)$ and λ^k is an optimal dual vector associated to

Algorithm 1: Primal-Gen

Data: A problem $P(\mathcal{X})$
Result: An optimal solution for $P(\mathcal{X})$
1 Set $\lambda^0 = 0$, $\mathcal{S}^1 \subseteq \mathcal{X}$ contains at least one feasible solution for $P(\mathcal{X})$ and $k = 1$;
2 **repeat**
3 Solve $P(\mathcal{S}^k)$. Let (x^k, y^k) be an optimal solution;
4 Let λ^k be an optimal dual vector corresponding to the constraints $-g(x, y) \in \mathcal{C}$;
5 **if** $\lambda^k = \lambda^{k-1}$ **then**
6 **return** (x^k, y^k);
7 Solve $L(\mathcal{X}, \lambda^k)$. Let $(\bar{x}^k, \bar{y}^k)$ be an optimal solution;
8 **if** $\bar{x}^k \in \mathcal{S}^k$ **then**
9 **return** (x^k, y^k);
10 Choose a set $\mathcal{S}^{k+1} \subseteq \mathcal{X}$ containing $\bar{x}^k$;
11 $k = k + 1$

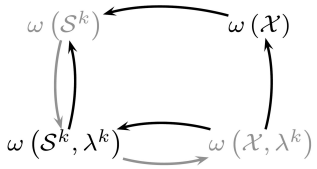

Fig. 1: Relationships between the optimal values of the problems involved in Algorithm 1.

$-g(x, y) \in \mathcal{C}$ in $P(\mathcal{S}^k)$, then (x^k, y^k) is also optimal for $L(\mathcal{S}^k, \lambda^k)$. From Assumption 2, $P(\mathcal{S}^k)$ has no duality gap so that $\omega(\mathcal{S}^k) = \omega(\mathcal{S}^k, \lambda^k)$, thus proving the gray edges in Figure 1. To summarize, we have:

$$\omega(\mathcal{X}) \leqslant \omega\left(\mathcal{S}^k\right) = \omega\left(\mathcal{S}^k, \lambda^k\right) = \omega\left(\mathcal{X}, \lambda^k\right) \leqslant \omega(\mathcal{X}),$$

making (x^k, y^k) optimal for $P(\mathcal{X})$. Now, if Algorithm 1 terminates because of the stopping criterion at line 5, we can choose $(\bar{x}^k, \bar{y}^k) = (\bar{x}^{k-1}, \bar{y}^{k-1})$ and have $\bar{x}^k \in \mathcal{S}^k$, which is the first stopping condition. □

2.1 General remarks

Algorithm 1 can be useful only if several conditions are met: 1) The Lagrangean problems $L(\mathcal{X}, \lambda^k)$ exhibit some exploitable structure, making them easy to solve even if they are very large, 2) the restricted problems $P(\mathcal{S}^k)$ can keep a complicated structure, but must be small enough so that they are solvable in a reasonable amount of time. In other words, either the dimension of the feasibility set of $P(\mathcal{S}^k)$ is significantly smaller than the feasibility set of $P(\mathcal{X})$, or $P(\mathcal{S}^k)$ exhibits a special structure or is sparser than in $P(\mathcal{X})$ and 3) the sets $\mathcal{S}^k$ are such that the problems $P(\mathcal{S}^k)$ are still feasible and satisfy Assumption 2. We only proved that if Algorithm 1 were to stop, it would return an optimal solution but not that it would necessarily stop, its finite termination generally depends on the way the sets $\mathcal{S}^k$

are generated in combination with the structure of the original problem. More precisely, the sequence $\mathcal{S}^k$ should shift towards at least one optimal solution of $P(\mathcal{X})$ by e.g. strictly growing in size or dimension until reaching $\mathcal{X}$ in the worst case. Ideally, the sets $\mathcal{S}^k$ should contain increasingly good solutions for $P(\mathcal{X})$ such that we can stop prematurely the algorithms and still obtain an approximately optimal solution. This can be achieved by e.g. forcing $\mathcal{S}^k$ to contain $\mathcal{S}^{k-1}$. Because we are maintaining lower and upper bounds over the optimal value of $P(\mathcal{X})$ at each iteration via the objective values of the Lagrangean relaxations $L\left(\mathcal{X}, \lambda^k\right)$ and the restricted problems $P\left(\mathcal{S}^k\right)$, early termination of the algorithms can be reasonably used and an optimality gap is provided.

2.2 y-independency

Notice that the pricing problem $L\left(\mathcal{X}, \lambda^k\right)$ 1) is optimized in x *and* the extra variables y and 2) it is still a nonlinear problem that can be as difficult to solve as the original problem. We partially address the former issue in the next Proposition and both issues in the next subsection. First, let us introduce the so-called *(P)-property* from Geoffrion [20] in the context of the *Generalized Benders Decomposition* that allows to generalize several results in this work.

Definition 2 (Adapted from [20]) Given $\varphi : \mathcal{U} \times \mathcal{V} \to \mathbb{R}$, the problem

$$\min_{(u,v)\in\mathcal{U}\times\mathcal{V}} \varphi(u,v) \tag{6}$$

satisfies the (P)-property wrt v if $\min_{v\in\mathcal{V}} \varphi(u,v)$ can be solved independently of the choice of $u \in \mathcal{U}$.

Even if this (P)-property appears to be overly restrictive, we now give several sufficient conditions on φ that make it hold:

Proposition 2 *Problem* (6) *satisfies the* (P)*-property wrt* v *if for some function* $\varphi_2 : \mathcal{V} \to \mathbb{R}$ *and some function* $\varphi_1 : \mathcal{U} \times \mathbb{R} \to \mathbb{R}$ *that is non-decreasing in its second argument,* $\varphi(u,v) = \varphi_1\left(u, \varphi_2(v)\right)$*. This includes a special cases*

1. φ *is separable in* u *and* v*, i.e.* $\varphi(u,v) = \varphi_1(u) + \varphi_2(v)$*,*
2. $\varphi(u,v) = \varphi_1(u)\varphi_2(v)$ *with* $\varphi_1(u) \geqslant 0$ *for any* $u \in \mathcal{U}$*,*

The proof of Proposition 2 is left to the reader. We are now ready to state a y-independency result for the Lagrangean relaxation $L\left(\mathcal{X}, \lambda^k\right)$:

Proposition 3 *If* $L\left(\mathcal{X}, \lambda^k\right)$ *satisfies the* (P)*-property wrt* y*, then* $y = y^k$ *is always an optimal choice in* $L\left(\mathcal{X}, \lambda^k\right)$ *and solving it is equivalent to solve:*

$$\left(L\left(\mathcal{X}, \lambda^k\right)\right) \quad \omega\left(\mathcal{X}, \lambda^k\right) := \min_{x\in\mathcal{X}} \left\{ f\left(x, y^k\right) + \left\langle \lambda^k, g\left(x, y^k\right)\right\rangle \right\}$$

Proof If $L\left(\mathcal{X}, \lambda^k\right)$ satisfies the (P)-property wrt y, then solving the problem in y given some $x \in \mathcal{X}$ is equivalent to solve the problem in y given $x = x^k$, i.e. for every $x \in \mathcal{X}$ we have

$$\begin{aligned} &\arg\min_y \left\{ f(x,y) + \left\langle \lambda^k, g(x,y)\right\rangle \right\} \\ =&\arg\min_y \left\{ f\left(x^k, y\right) + \left\langle \lambda^k, g\left(x^k, y\right)\right\rangle \right\}. \end{aligned} \tag{7}$$

From Assumption 1 there is no Lagrangean duality gap, implying that $\left(y^k, \lambda^k\right)$ is an optimal primal-dual solution for $\min_y \left\{f\left(x^k, y\right) : -g\left(x^k, y\right) \in \mathcal{C}\right\}$. In consequence, $\min_y \left\{f\left(x^k, y\right) + \left\langle \lambda^k, g\left(x^k, y\right)\right\rangle\right\}$ admits y^k as an optimal solution. Using equality (7), we obtain the result. □

3 A linearized primal algorithm

In practice it is common to have e.g. LPs with a nice structure - thus admitting tailored algorithms to solve them - getting *hardened* from replacing the linear objective function by a nonlinear objective f and/or adding possibly nonlinear side constraints $-g(x, y) \in \mathcal{C}$ to a polyhedral feasible set $\mathcal{X}$. This can happen in e.g. robust optimization [4] where uncertain problems are no easier than their deterministic counterparts.

Assumption 3 *In this subsection, we assume that given some cost vector $c \in \mathcal{A}$, the problem of minimizing $\langle c, x\rangle$ over $x \in \mathcal{X}$, i.e. $\min_{x\in\mathcal{X}} \langle c, x\rangle$, can be solved very efficiently.*

We now present some definitions and preliminary results that will allow us to use a linearized version of $L\left(\mathcal{X}, \lambda^k\right)$ as a pricing problem.

Definition 3 Consider a vector valued function $\varphi : \mathcal{U} \to \mathcal{V}$ and a cone $\mathcal{K} \subseteq \mathcal{V}$. φ is said to be $\mathcal{K}$-convex [9] if for any $t \in [0, 1]$ and any $\left(u^1, u^2\right) \in \mathcal{U} \times \mathcal{U}$,

$$t\varphi\left(u^1\right) + (1-t)\varphi\left(u^2\right) - \varphi\left(tu^1 + (1-t)u^2\right) \in \mathcal{K}.$$

Assumption 4 *In this subsection, we suppose that $\mathcal{X}$ and $\mathcal{C}$ are convex, f is convex and differentiable and g is $\mathcal{C}$-convex and differentiable.*

Proposition 4 *Let $\mathcal{K}$ be a cone, $\lambda \in \mathcal{K}^*$ and φ be a $\mathcal{K}$-convex function. Then*

$$\psi : u \to \langle \lambda, \varphi(u)\rangle \text{ is convex.}$$

Proof First, for any $t \in [0, 1]$ and any pair $\left(u^1, u^2\right) \in \mathcal{U} \times \mathcal{U}$ we have: $t\varphi\left(u^1\right) + (1-t)\varphi\left(u^2\right) - \varphi\left(tu^1 + (1-t)u^2\right) \in \mathcal{K}$. Given that $\lambda \in \mathcal{K}^*$ we have:

$$\begin{aligned} &\left\langle \lambda, t\varphi\left(u^1\right) + (1-t)\varphi\left(u^2\right) - \varphi\left(tu^1 + (1-t)u^2\right)\right\rangle \geqslant 0 \\ \Leftrightarrow &t\underbrace{\left\langle \lambda, \varphi\left(u^1\right)\right\rangle}_{\psi(u^1)} + (1-t)\underbrace{\left\langle \lambda, \varphi\left(u^2\right)\right\rangle}_{\psi(u^2)} \geqslant \underbrace{\left\langle \lambda, \varphi\left(tu^1 + (1-t)u^2\right)\right\rangle}_{\psi(tu^1+(1-t)u^2)} \end{aligned} \quad \square$$

Definition 4 Consider a differentiable function $\varphi : \mathcal{D} \to \mathbb{R}$. We call $\bar{\varphi}\left[\bar{u}\right]$ the linear approximation of φ at some $\bar{u} \in \mathcal{D}$, the following function:

$$\bar{\varphi}\left[\bar{u}\right](u) = \varphi\left(\bar{u}\right) + \left\langle \nabla\varphi\left(\bar{u}\right), u - \bar{u}\right\rangle.$$

Notice that $\bar{\varphi}\left[\bar{u}\right]$ is a global under estimator of φ if φ is convex. If φ is vector valued, its linear approximation is the component-wise linear approximation

$$\bar{\varphi}\left[\bar{u}\right](u) = \varphi\left(\bar{u}\right) + D\varphi\left(\bar{u}\right)\left(u - \bar{u}\right),$$

where $D\varphi\left(\bar{u}\right)$ is the Jacobian of φ at $\bar{u}$.

Proposition 5 *Given $\lambda \in \mathcal{V}$, $\varphi : \mathcal{U} \to \mathcal{V}$ and $\bar{u} \in \mathcal{U}$, the linear approximation of $\psi : u \to \langle \lambda, \varphi(u) \rangle$ at $\bar{u}$ is $\bar{\psi}[\bar{u}] : u \to \langle \lambda, \bar{\varphi}[\bar{u}](u) \rangle$.*

Proof The linear approximation of ψ at $\bar{u}$ is $\bar{\psi}[\bar{u}](u) := \psi(\bar{u}) + \langle \nabla\psi[\bar{u}], u - \bar{u} \rangle$. By definition:

$$\begin{aligned}
\langle \nabla\psi[\bar{u}], u - \bar{u} \rangle &= \lim_{\epsilon \to 0} \frac{\psi(\bar{u} + \epsilon(u - \bar{u})) - \psi(\bar{u})}{\epsilon} \\
&= \lim_{\epsilon \to 0} \frac{\langle \lambda, \varphi(\bar{u} + \epsilon(u - \bar{u})) \rangle - \langle \lambda, \varphi(\bar{u}) \rangle}{\epsilon} \\
&= \left\langle \lambda, \lim_{\epsilon \to 0} \frac{\varphi(\bar{u} + \epsilon(u - \bar{u})) - \varphi(\bar{u})}{\epsilon} \right\rangle = \langle \lambda, D\varphi(\bar{u})(u - \bar{u}) \rangle,
\end{aligned}$$

implying that $\bar{\psi}[\bar{u}](u) = \langle \lambda, \varphi(\bar{u}) \rangle + \langle \lambda, D\varphi(\bar{u})(u - \bar{u}) \rangle = \langle \lambda, \bar{\varphi}[\bar{u}](u) \rangle$, proving the result. □

Proposition 6 *Consider the optimization problem $\omega^* := \min_{u \in \mathcal{U}} \varphi(u)$, where $\mathcal{U}$ is convex and $\varphi : \mathcal{U} \to \mathbb{R}$ is differentiable. If u^* is one of its optimal solutions, it is also optimal for $\bar{\omega}[u^*] := \min_{u \in \mathcal{U}} \{\bar{\varphi}[u^*](u) := \varphi(u^*) + \langle \nabla\varphi(u^*), u - u^* \rangle\}$ and we have $\omega^* = \bar{\omega}[u^*]$.*

Proof First, by convexity of $\mathcal{U}$ and optimality of u^*, for any $\epsilon \in]0, 1]$ we have:

$$\begin{aligned}
&\varphi(u^*) \leqslant \varphi(u^* + \epsilon(u - u^*)) && ,\forall \epsilon \in]0,1], \forall u \in \mathcal{U} \\
\Leftrightarrow &\frac{\varphi(u^* + \epsilon(u - u^*)) - \varphi(u^*)}{\epsilon} \geqslant 0 && ,\forall \epsilon \in]0,1], \forall u \in \mathcal{U}.
\end{aligned}$$

The latter implies that when $\epsilon \to 0$ we obtain:

$$\begin{aligned}
&\langle \nabla\varphi(u^*), u - u^* \rangle \geqslant 0 && ,\forall u \in \mathcal{U} \\
\Leftrightarrow &\underbrace{\varphi(u^*) + \langle \nabla\varphi(u^*), u - u^* \rangle}_{\bar{\varphi}[u^*](u)} \geqslant \underbrace{\varphi(u^*) + \langle \nabla\varphi(u^*), u^* - u^* \rangle}_{\bar{\varphi}[u^*](u^*)} && ,\forall u \in \mathcal{U}
\end{aligned}$$

meaning that u^* is optimal for $\min_{u \in \mathcal{U}} \bar{\varphi}[u^*](u)$. As a byproduct, $\omega^* = \bar{\omega}[u^*]$. □

Let us define the following problems:

$$\begin{aligned}
\left(\overline{L}[\bar{x}, \bar{y}](\mathcal{X}, \lambda)\right) \quad & \bar{\omega}[\bar{x}, \bar{y}](\mathcal{X}, \lambda) := \min_{x \in \mathcal{X}, y} \left\{\bar{f}[\bar{x}, \bar{y}](x, y) + \langle \lambda, \bar{g}[\bar{x}, \bar{y}](x, y) \rangle\right\} \\
\left(\overline{L}[\bar{x}, \bar{y}](\mathcal{S}, \lambda)\right) \quad & \bar{\omega}[\bar{x}, \bar{y}](\mathcal{S}, \lambda) := \min_{x \in \mathcal{S}, y} \left\{\bar{f}[\bar{x}, \bar{y}](x, y) + \langle \lambda, \bar{g}[\bar{x}, \bar{y}](x, y) \rangle\right\}
\end{aligned}$$

that are respectively $L(\mathcal{X}, \lambda)$ and $L(\mathcal{S}, \lambda)$ with their objective functions linearized at $(\bar{x}, \bar{y})$.

Algorithm 2: Primal-Gen-Lin changes wrt Primal-Gen

5 **if** $\lambda^k = \lambda^{k-1}$ **and** $(x^k, y^k) = (x^{k-1}, y^{k-1})$ **then**
6 **return** (x^k, y^k);
7 Solve $\overline{L}\left[x^k, y^k\right](\mathcal{X}, \lambda^k)$. Let $(\bar{x}^k, \bar{y}^k)$ be an optimal solution;

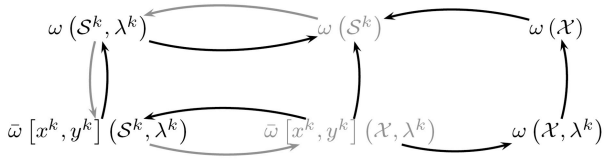

Fig. 2: Relationships between the optimal values of the problems involved in Algorithm 2.

3.1 A linearized algorithm

Now consider the following algorithm Primal-Gen-Lin: 1) instead of solving the pricing problem $L(\mathcal{X}, \lambda^k)$ we solve its linear approximation $\overline{L}\left[x^k, y^k\right](\mathcal{X}, \lambda^k)$ at the current incumbent (x^k, y^k) of the restricted problem $P(\mathcal{S}^k)$ and 2) the stopping criterion at line 5 of Algorithm 1 is replaced by a more restrictive condition. We describe the changes applied to Algorithm 1 in Algorithm 2 and illustrate the relationships between the bounds of the problems involved in it in Figure 2.

Proposition 7 *Algorithm 2 returns an optimal solution for $P(\mathcal{X})$ at termination.*

Proof If the algorithm terminates because the optimal solution $\bar{x}^k$ of the linearized pricing $\overline{L}\left[x^k, y^k\right](\mathcal{X}, \lambda^k)$ belongs to $\mathcal{S}^k$, it is also feasible and optimal for $\overline{L}\left[x^k, y^k\right](\mathcal{S}^k, \lambda^k)$. We then have:

$$\bar{\omega}\left[x^k, y^k\right]\left(\mathcal{S}^k, \lambda^k\right) = \bar{\omega}\left[x^k, y^k\right]\left(\mathcal{X}, \lambda^k\right).$$

Furthermore, we have that: $\bar{\omega}\left[x^k, y^k\right](\mathcal{X}, \lambda^k) \leqslant \omega(\mathcal{X}, \lambda^k) \leqslant \omega(\mathcal{X})$. The left inequality from the fact that f is convex and g is $\mathcal{C}$-convex so that from Proposition 5, $\bar{f}\left[x^k, y^k\right]$ and $\left\langle \lambda^k, \bar{g}\left[x^k, y^k\right](\cdot, \cdot)\right\rangle$ are global under estimators of f and $\left\langle \lambda^k, g(\cdot, \cdot)\right\rangle$ respectively. In consequence we have that $\bar{\omega}\left[x^k, y^k\right](\mathcal{S}^k, \lambda^k) \leqslant \omega(\mathcal{X})$. On another hand, x^k is also an optimal solution for $L(\mathcal{S}^k, \lambda^k)$ by definition . To finish the proof, we can interpret $\overline{L}\left[x^k, y^k\right](\mathcal{S}^k, \lambda^k)$ as the linearization of $L(\mathcal{S}^k, \lambda^k)$ at one of its optimal solutions (x^k, y^k). Recalling that:

$$\omega\left(\mathcal{S}^k, \lambda^k\right) := \min_{x \in \mathcal{S}^k, y} \left\{ f(x, y) + \left\langle \lambda^k, g(x, y)\right\rangle \right\}$$
$$\bar{\omega}\left[x^k, y^k\right]\left(\mathcal{S}^k, \lambda^k\right) := \min_{x \in \mathcal{S}^k, y} \left\{ \bar{f}\left[x^k, y^k\right](x, y) + \left\langle \lambda^k, \bar{g}\left[x^k, y^k\right](x, y)\right\rangle \right\},$$

Proposition 6 then tells us that (x^k, y^k) is also an optimal solution for the linearized pricing problem $\overline{L}\left[x^k, y^k\right](\mathcal{S}^k, \lambda^k)$. In consequence, we have $\omega(\mathcal{S}^k, \lambda^k) =$

$\bar{\omega}\left[x^k, y^k\right]\left(\mathcal{S}^k, \lambda^k\right)$. To summarize, we obtain

$$\begin{aligned}\omega\left(\mathcal{X}\right) &\leqslant \omega\left(\mathcal{S}^k\right) = \omega\left(\mathcal{S}^k, \lambda^k\right) = \bar{\omega}\left[x^k, y^k\right]\left(\mathcal{S}^k, \lambda^k\right) = \bar{\omega}\left[x^k, y^k\right]\left(\mathcal{X}, \lambda^k\right) \\ &\leqslant \omega\left(\mathcal{X}, \lambda^k\right) \leqslant \omega\left(\mathcal{X}\right),\end{aligned}$$

thus proving the optimality of x^k for $P\left(\mathcal{X}\right)$. It is not obvious that we can directly use the early stopping criterion at line 5 of Algorithm 1 in this linearized case. However, if Algorithm 2 stops because of its criterion at line 5, we can choose $\left(\bar{x}^k, \bar{y}^k\right) = \left(\bar{x}^{k-1}, \bar{y}^{k-1}\right)$, thus satisfying $\bar{x}^k \in \mathcal{S}^k$. □

3.2 y-independency

As opposed to Algorithm 1, the next proposition shows that we can always get rid of the variables y in the linearized Lagrangean relaxation $\overline{L}\left[x^k, y^k\right]\left(\mathcal{X}, \lambda^k\right)$, without needing the (P)-property:

Proposition 8 *Taking $y = y^k$ is always optimal for the linearization of the Lagrangean relaxation $\overline{L}\left[x^k, y^k\right]\left(\mathcal{X}, \lambda^k\right)$, which becomes a problem in x only, minimizing a linear objective function:*

$$\bar{\omega}\left[x^k, y^k\right]\left(\mathcal{X}, \lambda^k\right) := \min_{x \in \mathcal{X}} \left\{\bar{f}\left[x^k, y^k\right]\left(x, y^k\right) + \left\langle \lambda^k, \bar{g}\left[x^k, y^k\right]\left(x, y^k\right)\right\rangle\right\}$$

Proof First, notice that we have:

$$\begin{aligned}\bar{f}\left[x^k, y^k\right](x, y) =& f\left(x^k, y^k\right) \\ &+ \left\langle \nabla_x f\left(x^k, y^k\right), x - x^k\right\rangle + \left\langle \nabla_y f\left(x^k, y^k\right), y - y^k\right\rangle \\ \bar{g}\left[x^k, y^k\right](x, y) =& g\left(x^k, y^k\right) + D_x g\left(x^k, y^k\right)\left(x - x^k\right) + D_y g\left(x^k, y^k\right)\left(y - y^k\right)\end{aligned}$$

In consequence, we have that:

$$\begin{aligned}\bar{\omega}_P\left[x^k, y^k\right]\left(\lambda^k\right) :=& f\left(x^k, y^k\right) - \left\langle \nabla_x f\left(x^k, y^k\right), x^k\right\rangle - \left\langle \nabla_y f\left(x^k, y^k\right), y^k\right\rangle \\ &+ \left\langle \lambda^k, g\left(x^k, y^k\right) - D_x g\left(x^k, y^k\right) x^k - D_y g\left(x^k, y^k\right) y^k\right\rangle \\ &+ \min_{x \in \mathcal{X}} \left\{\left\langle \nabla_x f\left(x^k, y^k\right), x\right\rangle + \left\langle \lambda^k, D_x g\left(x^k, y^k\right) x\right\rangle\right\} \\ &+ \min_{y} \left\{\left\langle \nabla_y f\left(x^k, y^k\right), y\right\rangle + \left\langle \lambda^k, D_y g\left(x^k, y^k\right) y\right\rangle\right\}\end{aligned}$$

given that the linearized problem has a linear objective and that the variables x and y are not bound together, we can separate the optimization in x and y. Wlog we can assume that the problem in y attains its minimum, otherwise it would mean that the master problem is unbounded and we can stop. Under this assumption, the first order optimality conditions in y for the reduced problem $RP\left(\mathcal{S}^k\right)$ are: $\nabla_y f\left(x^k, y^k\right) + D_y g\left(x^k, y^k\right)^* \lambda^k = 0$, in turn implying that for any y: $\left\langle \nabla_y f\left(x^k, y^k\right), y\right\rangle + \left\langle \lambda^k, D_y g\left(x^k, y^k\right) y\right\rangle = 0$, meaning that the objective value of the problem in y is always zero, proving the result. □

3.3 "Reduced costs"

Recall that using DW for LPs, we can stop whenever the reduced costs are zero: the next Proposition shows that a similar criterion is also valid for the linearized scheme in our nonlinear setting.

Proposition 9 *Let* $\bar{c} := \nabla_x f\left(x^k, y^k\right) + D^* g\left(x^k, y^k\right) \lambda^k$ *be the cost vector of the pricing problem in the linearized scheme. If* $\bar{c} = 0$ *then* $\left(x^k, y^k\right)$ *is optimal for the full problem* $P(\mathcal{X})$.

Proof Recall that the linearized pricing problem in x is $\min_{x \in \mathcal{X}} \langle \bar{c}, x \rangle$. If $\bar{c} = 0$, then any $\bar{x}^k \in \mathcal{X}$ is optimal: choosing any $\bar{x}^k \in \mathcal{S}^k \subseteq \mathcal{X}$ we reach the stopping criterion, concluding the proof. □

This last result is particularly interesting in practice as it also provides a computationally cheap stopping criterion for the non-linearized scheme too. Indeed, the pricing problem - linearized or not - always provides a lower bound for ω^*: checking the "reduced costs" makes sure that the master problem cannot be improved.

3.4 General remarks

As we show in our experiments, the use of the linearized Algorithm 2 can be extremely useful when a dedicated algorithm is available for a linear objective pricing problem. In a ongoing work, we show its usefulness in discrete optimization, in terms of implementability, convergence and interpretation. Notice the use of the (P)-property in a primal setting whereas it was originally used in a dual setting for the generalized BD algorithm of Geoffrion [20]. The results in this section might possibly be extended to non-differentiable functions by working only with subgradients, we leave this idea for future work.

4 Relationship with existing schemes

4.1 Dantzig-Wolfe

Assuming that $\mathcal{X} \subseteq \mathbb{R}^n$ is a polyhedron, $y \in \mathbb{R}^{n_0}$ and the conic inequality is defined by $\mathcal{C} = \mathbb{R}^m_+$, $g(x, y) := b - Xx - Yy$ and the objective is linear $f(x, y) = c^\top x + d^\top y$. Using $\mathcal{S}^k := \text{conv.hull}\left(\left\{\bar{x}^l\right\}_{l \in \mathcal{V}^k}\right) + \text{cone.hull}\left(\left\{\bar{x}^l\right\}_{l \in \mathcal{R}^k}\right)$ we retrieve DW. In this example, the set $\mathcal{S}^k$ is the convex hull of the vertices plus the cone spanned by the extreme rays of $\mathcal{X}$ that we found via the Lagrangean relaxations until iteration k. Its finite convergence is ensured by the fact that $\mathcal{X}$ has a finite - although exponential in general - number of extreme rays $|\mathcal{R}|$ and vertices $|\mathcal{V}|$, thus making $\mathcal{S}^k$ finitely converge to $\mathcal{X}$. Notice that if $\mathcal{C}$ is a more general cone, our algorithm generalizes DW for LCs, where at each iteration, the master problem is an LC and the Lagrangean relaxation $L\left(\mathcal{X}, \lambda^k\right)$ is an LP, without the need of linearizing the objective. Applications of DW to LCs can be found in [1] and references therein. Furthermore, there is a straightforward extension of DW to a special class

of nonlinear problems: Let us go back to the general case for $\mathcal{C}$, f and g but keeping $\mathcal{X}$ polyhedral. $\overline{L}\left[x^k, y^k\right]\left(\mathcal{X}, \lambda^k\right)$ turns out to be an LP as from Proposition 8 it optimizes the linear function $\bar{f}\left[x^k, y^k\right](\cdot, y^k) + \left\langle \lambda^k, \bar{g}\left[x^k, y^k\right](\cdot, y^k)\right\rangle$ over the polyhedron $\mathcal{X}$. Because $\mathcal{X}$ is finitely generated, the finite convergence of the linearized algorithm is ensured with the same arguments as DW for LPs. Also, notice that if $\mathcal{X}$ were to be non-convex, in order to have $\mathcal{S}^k \subset \mathcal{X}$, we must use

$$\mathcal{S}^k := \mathcal{X} \cap \left(\text{conv.hull}\left(\left\{\bar{x}^l\right\}_{l\in\mathcal{V}^k}\right) + \text{cone.hull}\left(\left\{\bar{x}^l\right\}_{l\in\mathcal{R}^k}\right)\right),$$

thus potentially losing the advantage of dropping any $\mathcal{X}$-defining constraint in the master problem as in the LP/convex cases. In a similar way, if the conic hull was used instead and $\mathcal{X}$ was a cone, it would also be sufficient to use $\mathcal{S}^k := \text{cone.hull}\left(\left\{\bar{x}^l\right\}_{l\in\mathcal{R}^k}\right)$ instead of $\mathcal{S}^k := \mathcal{X} \cap \left(\text{cone.hull}\left(\left\{\bar{x}^l\right\}_{l\in\mathcal{R}^k}\right)\right)$.

4.2 Bienstock-Zuckerberg

We now link our framework with a decomposition scheme for LPs [8] where the choice of $\mathcal{S}$ differs substantially from DW.

Definition 5 For any vector $x \in \mathbb{R}^n$, we call a partition *induced* by x a partition $\mathcal{J}(x) = \{\mathcal{J}_1(x), ..., \mathcal{J}_L(x)\}$ of the indices $\{1, ..., n\}$ such that for every $l \in \{1, ..., L\}$ and any pair of indices $\left(j, j'\right) \in \mathcal{J}_l(x) \times \mathcal{J}_l(x)$ we have $x_j = x_{j'}$.

Definition 6 Given two partitions $\mathcal{J} = \{\mathcal{J}_1, ..., \mathcal{J}_L\}$ and $\mathcal{J}' = \left\{\mathcal{J}'_1, ..., \mathcal{J}'_{L'}\right\}$ of a finite set of indices, we define their *intersection* $\mathcal{J}\Delta\mathcal{J}'$ as follows:

$$\mathcal{J}\Delta\mathcal{J}' := \left\{\mathcal{J}_l \cap \mathcal{J}'_{l'}, \forall\left(l, l'\right) \in \{1, ..., L\} \times \{1, ..., L'\}\right\}$$

Given a current partition $\mathcal{J}^k$ and $\mathcal{J}^k\left(\bar{x}^k\right)$ a partition induced by $\bar{x}^k$, the new partition is given by $\mathcal{J}^{k+1} := \mathcal{J}^k\Delta\mathcal{J}^k\left(\bar{x}^k\right)$. Using

$$\mathcal{S}^k := \left\{x \in \mathcal{X} : \exists\theta \in \mathbb{R}^{\left|\mathcal{J}^k\right|} : x_j = \theta_l, \forall j \in \mathcal{J}^k_l, \forall l \in \left\{1, ..., \left|\mathcal{J}^k\right|\right\}\right\}$$

makes Algorithms 1 and 2 generalizations to nonlinear problems of the BZ algorithm [8]. This time the convergence is not ensured by some property of $P\left(\mathcal{X}\right)$ but rather thanks to the structure of the sets $\mathcal{S}^k$. In fact, either the partition is refined until eventually reach $\{\{1\}, ..., \{n\}\}$, meaning we reached the original problem $P\left(\mathcal{X}\right)$, or the partition is not refined and we found an optimal solution for $P\left(\mathcal{X}\right)$. The former implies that in the worst case, we will converge to an optimal solution in at most n iterations, due to the fact that the dimension of $\mathcal{S}^k$ is increased by at least one at each iteration. On another hand, the Lagrangean relaxation $L\left(\mathcal{X}, \lambda^k\right)$ may return an optimal solution $\bar{x}^k$ with a large number of different values, generating a high-cardinality induced partition $\mathcal{J}\left(\bar{x}^k\right)$, hence increasing rapidly the size of the partition $\mathcal{J}^{k+1}$ used in the restricted problem. This issue is partially addressed by the linearized Algorithm 2. For example, if $\mathcal{X}$ is polyhedral and possesses the integrality property, the Lagrangean problems $\overline{L}\left[x^k, y^k\right]\left(\mathcal{X}, \lambda^k\right)$ can return integer optimal solutions $\bar{x}^k$, thus increasing the probability of having a

reduced number of different values. This issue is completely circumvented in the next scheme.

Also, given that $\left\{x : \exists\theta \in \mathbb{R}^{|\mathcal{J}^k|} : x_j = \theta_l, \forall j \in \mathcal{J}_l^k, \forall l \in \{1, ..., |\mathcal{J}^k|\}\right\}$ is generally not contained in $\mathcal{X}$, it is necessary to maintain the $\mathcal{X}$-defining constraints. Notice however that this issue can be circumvented sometimes, e.g. for bound constraints $l \leqslant x \leqslant u$ eventually present in the definition of $\mathcal{X}$: they become equivalent to the $\left|\mathcal{J}^k\right| \ll n$ following bound constraints *for* θ*:*

$$\max_{j\in\mathcal{J}_l^k} l_j \leqslant \theta_l \leqslant \min_{j\in\mathcal{J}_l^k} u_j \quad , \forall l \in \left\{1, ..., \left|\mathcal{J}^k\right|\right\}. \tag{8}$$

4.3 Non-partitioned BZ

In [29] they describe a scheme that uses at each iteration k the subset of $\mathcal{X}$ spanned by lin.hull $\left(\left\{\bar{x}^l\right\}_{l\in\{1,...,k\}}\right)$, i.e.

$$\mathcal{S}^k := \left\{x \in \mathcal{X} : \exists\theta \in \mathbb{R}^k : x = \sum_{l=1}^{k} \theta_l \bar{x}^l\right\}.$$

As for the classic BZ, it converges to lin.hull $(\mathcal{X})$ in at most n iterations, What is not mentioned in [29] though, is that this is true independently of the structure of $P(\mathcal{X})$. Instead of using partitions of variables it uses the raw directions $\bar{x}^k$, thus avoiding an explosive increase in the number of variables. This comes at the cost of a less structured restricted problem $P(\mathcal{S})$: in fact, a variable aggregation is akin to a contraction operation in combinatorial optimization, which can eliminate a substantial amount of rows and columns when dealing with LPs.

Notice that even if we can maintain a reasonable number of variables in the master problem, the loss in structure in comparison to BZ prohibits the use of the trick in (8) in general, and all $\mathcal{X}$-defining constraints must be kept, bound constraints included.

4.4 Non-convex problems and reformulation-based relaxations

It is well known that DW for LPs is able to provide strong relaxation bounds for difficult MILPs. We now give some pointers about the fact that the choice of $\mathcal{S}$ defines the relaxation we choose and can alter the strength of the relaxation bounds.

Convexification Coming back to the LP world, DW comes in handy when facing problems having the following form:

$$\min_{x,y} c^\top x + d^\top y \tag{9a}$$

$$\text{s.t.: } x \in \widetilde{\mathcal{X}} \tag{9b}$$

$$Xx + Yy \geqslant b, \tag{9c}$$

where $\widetilde{\mathcal{X}}$ is eventually non-convex - say an integer lattice $\mathbb{Z}^n$ intersected with some polyhedron $\mathcal{P}$. Strong Lagrangean duality does not hold in general, and one can choose to relax the problem by replacing $\widetilde{\mathcal{X}}$ by $\mathcal{X} := \text{conv.hull}(\widetilde{\mathcal{X}})$. This implies that strong Lagrangean duality holds and DW can be rightfully used to solve the resulting problem. However, a description of conv.hull($\widetilde{\mathcal{X}}$) might be hard to obtain. To avoid this issue, let us consider the resulting Lagrangean relaxation:

$$\min_{x} \left\{ \left(c - X^\top \lambda\right)^\top x : x \in \text{conv.hull}(\widetilde{\mathcal{X}}) \right\}. \tag{10}$$

Problem (10) optimizes a linear function over a convex set whose extreme points are de facto members of $\widetilde{\mathcal{X}}$. In consequence, there is at least one optimal solution to (10) that belongs to $\widetilde{\mathcal{X}}$ implying that solving (10) is equivalent to solve $\min_{x \in \widetilde{\mathcal{X}}} \left(c - X^\top \lambda\right)^\top x$, which - as opposed to (10) - might be implementable in practice if an explicit description of $\widetilde{\mathcal{X}}$ is already available. Using DW on Problem (9) then solves

$$\begin{aligned} &\min_{x,y} \; c^\top x + d^\top y \\ &\text{s.t.: } x \in \text{conv.hull}(\widetilde{\mathcal{X}}), \quad Xx + Yy \geqslant b. \end{aligned}$$

This process is sometimes called *convexification*. In addition, if $\widetilde{\mathcal{X}} := \mathbb{Z}^n \cap \mathcal{P}$ and does not have the integrality property [21] - i.e. the LP relaxation of the pricing problem does not necessarily admit an integer optimal solution - the bound DW provides can be strictly better than the LP relaxation bound of (9). This is due to the fact that conv.hull($\widetilde{\mathcal{X}}$) $\subseteq \mathcal{P}$, with equality if and only if the integrality property holds. Notice that in this case, we do not know explicitly $\mathcal{X}$. Additionally, all the DW's assumptions are verified by strong duality in LP as $\mathcal{X}$ is a polyhedron.

Reformulation-based decomposition As described earlier, [8] proposes a method to *partition* the variables x into *clusters* of variables with a common value, thus constructing a different set $\mathcal{S}$ than DW. The restricted problem is now working on a reduced set of *aggregated* variables instead of variables corresponding to vertices and extreme rays of conv.hull($\widetilde{\mathcal{X}}$). As pointed in [29], doing so BZ solves in fact

$$\begin{aligned} &\min_{x,y} \; c^\top x + d^\top y \\ &\text{s.t.: } x \in \mathcal{X} := \text{lin.hull}(\widetilde{\mathcal{X}}) \cap \mathcal{P}, \quad Xx + Yy \geqslant b, \end{aligned}$$

Given that conv.hull($\widetilde{\mathcal{X}}$) $\subseteq \left(\text{lin.hull}(\widetilde{\mathcal{X}}) \cap \mathcal{P}\right) \subseteq \mathcal{P}$, the bound provided by BZ is no worse than the LP bound but is also no better than the one provided by DW, again with equality if $\widetilde{\mathcal{X}}$ has the integrality property. On one hand, DW relies on the fact that conv.hull($\widetilde{\mathcal{X}}$) $\cap \mathcal{P}$ = conv.hull($\widetilde{\mathcal{X}}$) to completely reformulate the master problem and eliminate the x variables, whereas BZ cannot avoid to keep them thus providing a potentially more difficult problem. On another hand, the convergence of BZ relies on dimension arguments that are independent of the structure of $P(\widetilde{\mathcal{X}})$.

5 Computational experience

The algorithms presented in this paper were coded in C programming language and run over Dell PowerEdge C6420 cluster nodes with Intel Xeon Gold 6152 CPUs at 2.10GHz with 32Gb RAM each. All the nonlinear programming problems are solved using the callable library of IPOPT [32, 41], using as a subroutine the linear solver Pardiso [30, 31].

5.1 Problem description

Consider a portfolio optimization where the aim is to allocate the resources of $T+1$ different clients interested in disjoint subsets of stocks and having different risk profiles. The administrator of the budgets getting a portion of the benefits, it also requires that the variance of the overall returns must be no greater than some threshold σ^2 or that the expected return on some stocks must be greater than some target μ. T clients have budgets b_t and are interested in n_t assets each while a separate client - the zeroth - has budget b_0 and is interested in $n_0 \ll n := \sum_{t=1}^{T} n_t$ assets. Client $t(0)$ has to pay a unitary cost a_j^t (a_i^0) per asset j. Each asset j has an uncertain future value c_j^t (c_i^0) and can be purchased in at most u_i^t (u_i^0) units. The variables x_j^t (y_j) represent the amount of each asset j purchased by client $t(0)$. Defining

$$\mathcal{X}_t := \left\{ x \in \left[0, u^t\right] : \left(a^t\right)^\top x \leqslant b_t \right\},$$

x and y must satisfy $x^t \in \mathcal{X}_t$, for each $t \in \{1, ..., T\}$ and $y \in \mathcal{X}_0$, the latter being part of the side constraints $g(x, y) \in \mathcal{C}$. The remaining side constraints are as follows: the variance over the same subset of assets must be no greater than a threshold σ^2:

$$\mathbb{V}\left(\left(d^0\right)^\top y + \sum_{t=1}^{T} \left(d^t\right)^\top x^t \right) \leqslant \sigma^2. \tag{11}$$

We see later on that we can interpret an approximation of (11) as a Second Order Cone (SOC) constraint. Each client wants to minimize a risk measure f_t (f_0) that depends on the uncertain return of the stocks. We consider that every client wants to minimize their entropic risk with some parameter α_t (α_0): $f_t\left(x^t\right) := \mathcal{E}_{\alpha_t}\left(-\left(c^t\right)^\top x^t\right)$, $f_0(y) := \mathcal{E}_{\alpha_0}\left(-\left(c^0\right)^\top y\right)$, where $\mathcal{E}_\alpha\left(Z\right) := \alpha \ln \mathbb{E}\left(e^{Z/\alpha}\right)$. The general problem can be cast as follows:

$$\min_{x,y} \left\{ f_0(y) + \sum_{t=1}^{T} f_t\left(x^t\right) \,:\, x^t \in \mathcal{X}_t, \forall t \in \{1, ..., T\},\, y \in \mathcal{X}_0,\, (11) \right\}$$

5.2 Sample Average Approximation

Using S samples $\left(c^{ts}, c^{0s}\right)$ of respective probabilities p_s, we approximate the variance and expectations as summarized in Table 1. For simplicity, we will use the

Original	SAA	type
$\mathcal{E}_\alpha\left(-c^\top x\right)$	$\alpha \ln \sum_{s=1}^{S} p_s e^{(-c^s)^\top x/\alpha}$	convex
$\mathbb{V}\left(d^\top x\right) \leqslant \sigma^2$	$\sum_{s=1}^{S} p_s \left((d^s)^\top x - \sum_{s'=1}^{S} p_{s'} \left(d^{s'}\right)^\top x\right)^2 \leqslant \sigma^2$	quadratic/SOC

Table 1: Sample Average Approximations

same names for the functions and their respective SAAs. Define V, V^t and V^0 such that $V^t_{sj} := \sqrt{p_s}\left(d^{ts}_j - d^t_j\right)$ and $Vx := \sum_{t=1}^{T} V^t x^t$, the variance constraint can be expressed as a classic nonlinear quadratic (CLA) convex constraint $\left|\left|V^0 y + Vx\right|\right|_2^2 \leqslant \sigma^2$ or the SOC constraint $\left(V^0 y + Vx, \sigma\right) \in \mathcal{L}_2^{S+1}$, where $\mathcal{L}_2^{S+1}$ is the Lorentz cone of dimension $S+1$. The full approximated problem becomes

$$
\begin{aligned}
\omega\left(\mathcal{X}\right) := \min_{x,y} \quad & \alpha_0 \ln \sum_{s=1}^{S} p_s e^{-\left(c^{0s}\right)^\top y/\alpha_0} + \sum_{t=1}^{T} \alpha_t \ln \sum_{s=1}^{S} p_s e^{-\left(c^{ts}\right)^\top x^t/\alpha_t} \\
\text{s.t.:} \quad & x^t \in \mathcal{X}_t \quad , \forall t \in \{1, ..., T\} \\
& y \in \mathcal{X}_0 \\
& \begin{cases} \left|\left|V^0 y + Vx\right|\right|_2^2 \leqslant \sigma^2 & \text{If (11) is seen as a CLA} \\ \left(V^0 y + Vx, \sigma\right) \in \mathcal{L}_2^{S+1} & \text{If (11) is seen as a SOC} \end{cases}
\end{aligned}
$$

5.3 Pricing problem

Depending on the coupling constraint considered, the pricing problem can have different structures: Given any dual vector $\left(\lambda^{1,0}, \lambda^{2,0}, \lambda^{3,0}\right) \in \mathbb{R}_+^{n_0} \times \mathbb{R}_+^{n_0} \times \mathbb{R}_+$ corresponding to the y-specific constraints $y \geqslant 0$, $y \leqslant u^0$ and $\left(a^0\right)^\top y \leqslant b_0$, the pricing problem is:

$$
\begin{aligned}
\omega\left(\lambda\right) := \min_{x,y} \quad & \alpha_0 \ln \sum_{s=1}^{S} p_s e^{-\left(c^{0s}\right)^\top y/\alpha_0} + \sum_{t=1}^{T} \alpha_t \ln \sum_{s=1}^{S} p_s e^{-\left(c^{ts}\right)^\top x^t/\alpha_t} \\
& - \left(\lambda^{1,0}\right)^\top y + \left(\lambda^{2,0}\right)^\top \left(y - u^0\right) + \lambda^{3,0}\left(\left(a^0\right)^\top y - b_0\right) \\
& + \begin{cases} \lambda^4 \left(\left|\left|V^0 y + Vx\right|\right|_2^2 - \sigma^2\right) \\ \qquad \text{If (11) is seen as a CLA} \qquad , \lambda^4 \in \mathbb{R}_+ \\ -\left(\lambda^4\right)^\top \left(V^0 y + Vx\right) - \lambda_0^4 \sigma \\ \qquad \text{If (11) is seen as a SOC, } \left(\lambda^4, \lambda_0^4\right) \in \mathcal{L}_2^{S+1} \end{cases} \\
\text{s.t.} \quad & x^t \in \mathcal{X}_t, \, \forall t \in \{1, ..., T\}
\end{aligned}
$$

Notice that considering (11) as a SOC makes the pricing problem separable in each x^t and y and also allows to use the y-independency result.

Using the linearized pricing, each sub-problem in x^t is solvable in $O(n_t \ln n_t)$ time: Indeed, the objective function becomes linear and we can solve each one of the T problems in x^t as follows:

Proposition 10 *The following LP can be reduced to a continuous Knapsack problem, and an optimal solution can be found in $O(n \ln n)$ time:*

$$\omega^* := \min_{x \in [0,u]} \left\{ c^\top x : a^\top x \leqslant b \right\}.$$

Proof Let us define the following subsets of $\{1, ..., n\}$:

$$\mathcal{J}_+ := \{j \in \{1, ..., n\} : c_j < 0, a_j > 0\} \quad \mathcal{J}_- := \{j \in \{1, ..., n\} : c_j > 0, a_j < 0\}$$
$$\mathcal{J}_0 := \{j \in \{1, ..., n\} : c_j, a_j \geqslant 0\} \quad \mathcal{J}_u := \{j \in \{1, ..., n\} : c_j, a_j \leqslant 0\}$$

First notice that we can fix beforehand the following variables

$$x_j^* = \begin{cases} u_j & \text{If } j \in \mathcal{J}_u \\ 0 & \text{If } j \in \mathcal{J}_0. \end{cases}$$

Next, for every $j \in \mathcal{J}_-$, we use the change of variable $x_j \leftarrow u_j - x_j$, obtaining the following problem:

$$\begin{aligned} \omega^* := \sum_{j \in \mathcal{J}_- \cup \mathcal{J}_u} c_j u_j + \min_x \quad & \sum_{j \in \mathcal{J}_+ \cup \mathcal{J}_-} \hat{c}_j x_j \\ \text{s.t.:} \quad & \sum_{j \in \mathcal{J}_+ \cup \mathcal{J}_-} \hat{a}_j x_j \leqslant \hat{b} \\ & x_j \in [0, u_j] \qquad\qquad , \forall j \in \mathcal{J}_+ \cup \mathcal{J}_- \end{aligned}$$

where $\hat{b} := b - \sum_{j \in \mathcal{J}_- \cup \mathcal{J}_u} a_j u_j$, $\hat{a}_j := a_j$ if $j \in \mathcal{J}_+$, $\hat{a}_j := -a_j$ if $j \in \mathcal{J}_-$, $\hat{c}_j := c_j$ if $j \in \mathcal{J}_+$ and $\hat{c}_j := -c_j$ if $j \in \mathcal{J}_-$. The latter is a knapsack problem with positive capacity $\hat{b}$ and weights $\hat{a}_j$ that can be solved by sorting the remaining indices $j \in \mathcal{J}_+ \cup \mathcal{J}_-$ in increasing *disutility* $\hat{c}_j / \hat{a}_j$ and filling the capacity constraint until no variable is available or the capacity is tight. □

5.4 Master problem

When solving the master, if we consider the side constraint as a SOC we need to solve an NLC that returns optimal dual variables. Unfortunately, even though nonlinear solvers or linear conic solvers do exist, we could not find any general purpose solver solving a problem having both features and returning conic multipliers. To circumvent this issue, the master is solved with an off the shelf nonlinear solver by considering the quadratic constraint as a nonlinear one and use the following result to obtain conic multipliers:

Proposition 11 *Given functions φ, ϕ and γ and some $\gamma_0 < 0$, consider the following NLC*

$$\omega^* := \min_u \{\varphi(u) : \phi(u) \leqslant 0, -(\gamma(u), \gamma_0) \in \mathcal{L}_2\}, \tag{15}$$

and its equivalent representation as a classic nonlinear optimization problem:

$$\omega^* := \min_u \left\{ \varphi(u) : \phi(u) \leqslant 0, ||\gamma(u)||_2^2 - \gamma_0^2 \leqslant 0 \right\}, \tag{16}$$

and assume they are both convex and neither has a Lagrangean duality gap. Given an optimal primal-dual pair $(u^*, (\pi^*, \lambda^*))$ *for* (16) *then* $\left(u^*, \left(\pi^*, \hat{\lambda}, \hat{\lambda}_0\right)\right)$ *is an optimal primal-dual pair for* (15)*, where:*

$$\left(\hat{\lambda}, \hat{\lambda}_0\right) := 2\lambda^* \left(\gamma\left(u^*\right), -\gamma_0\right).$$

Proof To prove our claim it is sufficient to show that $\left(u^*, \left(\pi^*, \hat{\lambda}, \hat{\lambda}_0\right)\right)$ satisfies the KKT conditions for (15), i.e.

$$\phi\left(u^*\right) \leqslant 0, \quad -\left(\gamma\left(u^*\right), \gamma_0\right) \in \mathcal{L}_2 \tag{17a}$$

$$\pi^* \geqslant 0, \quad \phi\left(u^*\right)^\top \pi^* = 0 \tag{17b}$$

$$\left(\hat{\lambda}, \hat{\lambda}_0\right) \in \mathcal{L}_2 \tag{17c}$$

$$\gamma\left(u^*\right)^\top \hat{\lambda} + \gamma_0 \hat{\lambda}_0 = 0 \tag{17d}$$

$$\nabla\varphi\left(u^*\right) + D\phi\left(u^*\right)^\top \pi^* + D\gamma\left(u^*\right)^\top \hat{\lambda} = 0 \tag{17e}$$

Conditions (17a) and (17b) are trivially satisfied, and given that $-\gamma_0, \lambda^* \geqslant 0$, the remaining conditions are equivalent to:

$$\left(\lambda^*\right)^2 \left(\left|\left|\gamma\left(u^*\right)\right|\right|_2^2 - \gamma_0^2\right) \leqslant 0$$

$$\lambda^* \left(\left|\left|\gamma\left(u^*\right)\right|\right|_2^2 - \gamma_0^2\right) = 0$$

$$\nabla\varphi\left(u^*\right) + D\phi\left(u^*\right)^\top \pi^* + 2\lambda^* D\gamma\left(u^*\right)^\top \gamma\left(u^*\right) = 0$$

which are all implied by the KKT conditions for (16) at $(u^*, (\pi^*, \lambda^*))$. □

As mentioned before, we can sometimes omit certain redundant constraints from the master problem. In consequence, we must have a way to find the dual variables associated to them when needed:

Proposition 12 *Given functions* φ*,* ϕ *and* γ *and two cones* $\mathcal{C}$ *and* $\mathcal{K}$ *of nonempty interior, consider the following NLC*

$$\omega^* := \min_u \left\{\varphi(u) : -\phi(u) \in \mathcal{C}, -\gamma(u) \in \mathcal{K}\right\}, \tag{18}$$

and assume that constraints $-\gamma(u) \in \mathcal{K}$ *are redundant, i.e.*

$$\omega^* := \min_u \left\{\varphi(u) : -\phi(u) \in \mathcal{C}\right\}. \tag{19}$$

Assume that both problems are convex and neither has a Lagrangean duality gap. Given an optimal primal-dual pair (u^*, λ^*) *for* (19) *then* $(u^*, (\lambda^*, \pi^* = 0))$ *is an optimal primal-dual pair for* (18)*.*

Proof To prove our claim it is sufficient to show that $(u^*, (\lambda^*, \pi^* = 0))$ satisfies the KKT conditions for (18): These conditions for (19) at (u^*, λ^*) are:

$$-\phi\left(u^*\right) \in \mathcal{C}, \quad \lambda^* \in \mathcal{C}^* \tag{20a}$$

$$\left\langle \lambda^*, \phi\left(u^*\right)\right\rangle = 0 \tag{20b}$$

$$\nabla\varphi\left(u^*\right) + D\phi\left(u^*\right)^* \lambda^* = 0 \tag{20c}$$

By assumption, u^* is also feasible (and optimal) for (18). Given that $\mathcal{K}$ has a non empty interior, then $\mathcal{K}^*$ is *pointed* - i.e. contains zero - meaning that $\pi^* = 0 \in \mathcal{K}^*$. Because $\pi^* = 0$ we have 1)$\langle \pi^*, \gamma(u^*) \rangle = 0$, and 2)$D\gamma(u^*)^* \pi^* = 0$: Using the latter point in (20c) proves that $(u^*, (\lambda^*, \pi^*))$ satisfies the last remaining KKT condition for the complete problem (18), the stationarity: $\nabla\varphi(u^*) + D\phi(u^*)^* \lambda^* + D\gamma(u^*)^* \pi^* = 0$.□

We summarize In Table 2 the types of pricing problem we encounter with our framework, and how to solve them.

Lin. LR	Coupling cone	LR type	Separable LR
yes	Any	T knapsacks	yes
no	$\mathbb{R}_+$	single NLP	no
	$\mathcal{L}_2^{S+1}$	T NLPs	yes

Table 2: Problem types

5.5 Checking if $\bar{x}^k \in \mathcal{S}^k$

We have now to determine if the current column generated $\bar{x}^k$ belongs or not to the current restricted set $\mathcal{S}^k$ used by the master. For the BZ scheme, it is enough to check if the size of the partition after refinement increased or not. For the lin.hull case, its is enough to check if $\bar{x}^k$ is a linear combination of the previous columns, which can be done by projection. However, in the conv.hull and cone.hull cases, we have to check whether a small LP is feasible. Let the polyhedron Θ^k be as follows

$$\Theta^k := \begin{cases} \left\{ \theta \in \mathbb{R}_+^k : \sum_{k'=0}^{k-1} \theta_{k'} = 1 \right\} & \text{If using conv.hull} \\ \mathbb{R}_+^k & \text{If using cone.hull} \end{cases}$$

In these cases, checking the membership of $\bar{x}^k$ is equivalent to check if the polyhedron $\left\{ \theta \in \Theta^k : \sum_{k'=0}^{k-1} \theta_{k'} \bar{x}^{k'} = \bar{x}^k \right\}$ is nonempty, which can be done by solving a small LP having k variables and n linear equations. We choose, however, to solve the following problem

$$\text{distance}_2^2\left(\bar{x}^k, \mathcal{S}^k\right) := \min_{\theta \in \Theta^k} \left|\left| \bar{x}^k - \sum_{k'=0}^{k-1} \theta_{k'} \bar{x}^{k'} \right|\right|_2^2.$$

This way, we are able to monitor the distance of the current column $\bar{x}^k$ to $\mathcal{S}^k$. We choose $||\cdot||_2$ instead of e.g. $||\cdot||_1$ because the full dimension $n \gg 1$ is not an issue during the execution of the interior point algorithm, whereas $||\cdot||_1$ would make appear $2n$ constraints.

5.6 Numerical Enhancement

When solving convex problems with an interior point method, solving the problems as they are rapidly exceeds the capabilities of an average workstation. In fact, the Hessian matrix of the penalized objective can have many nonzero coefficients because of the entropic risk measure and the variance constraint. This observation implies that the linear system that is solved during each Newton step of the interior point method can be overly demanding both in terms of memory and running time. To tackle this, we transform each objective by introducing new variables and constraints as follows:

$$\alpha_t \ln \sum_{s=1}^{S} p_s e^{\left(-c^{ts}\right)^\top x^t/\alpha_t} = \min_{v^t} \quad \alpha_t \ln \sum_{s=1}^{S} p_s e^{v_s^t/\alpha_t}$$
$$\text{s.t.} \quad v_s^t = -\left(c^{ts}\right)^\top x^t, \forall s \in \{1, ..., S\}.$$

In the same fashion, we replace the variance constraint with

$$||w||_2^2 \leqslant \sigma^2 \quad \text{and} \quad w = V^0 y + Vx.$$

This way, the Hessian of the Lagrangean function is a slightly larger matrix with way less nonzero coefficients, having at most $(T+1)S^2$ of them coming from the entropies, plus S from the variance constraint. For this reason and the fact that decomposition methods are typically useful when there is only a few side constraints, we purposefully kept S moderately small in our experimental design. The only remaining detail being how to catch the dual variables associated to the original constraints (depending only of x and y) from the dual variables of the constraints in the enhanced formulation (depending now on v and w). We address this in a general setting in the next Proposition, where we can basically ignore the extra constraints and use the multipliers *as is*:

Proposition 13 *Given functions φ, ϕ and γ and a proper cone $\mathcal{K}$ consider the following NLC*

$$\omega^* := \min_u \{\varphi(u) : \phi(u) \leqslant 0, -\gamma(u) \in \mathcal{K}\}. \tag{21}$$

Assume there exists a transformation with additional variables v and w such that for any $v = Vu$, $\varphi(u) = \tilde{\varphi}(v)$ and for any $w = Wu$, $\gamma(u) = \tilde{\gamma}(w)$, so that (21) can be equivalently rewritten as:

$$\omega^* := \min_{u,v,w} \{\tilde{\varphi}(v) \,:\, \phi(u) \leqslant 0,\, -\tilde{\gamma}(w) \in \mathcal{K},\, v = Vu,\, w = Wu\}. \tag{22}$$

Assuming that both are convex and neither has a Lagrangean duality gap, given an optimal primal-dual pair $\left((\tilde{u}, \tilde{v}, \tilde{w}), \left(\tilde{\pi}, \tilde{\lambda}, \tilde{\alpha}, \tilde{\beta}\right)\right)$ for (22) then $\left(\tilde{u}, \left(\tilde{\pi}, \tilde{\lambda}\right)\right)$ is an optimal pair for (21).

Proof The KKT conditions for (22) are

$$\phi(\tilde{u}) \leqslant 0, \quad -\tilde{\gamma}(\tilde{w}) \in \mathcal{K}, \quad \tilde{v} = V\tilde{u}, \quad \tilde{w} = W\tilde{u} \tag{23a}$$

$$\tilde{\pi} \geqslant 0, \quad \tilde{\lambda} \in \mathcal{K}^* \tag{23b}$$

$$\phi(\tilde{u})^\top \tilde{\pi} = 0, \quad \left\langle \tilde{\gamma}(\tilde{w}), \tilde{\lambda} \right\rangle = 0 \tag{23c}$$

$$\nabla\tilde{\varphi}(\tilde{v}) - \tilde{\alpha} = 0, \quad D\tilde{\gamma}(\tilde{w})^* \tilde{\lambda} - \tilde{\beta} = 0 \tag{23d}$$

$$D\phi(\tilde{u})^\top \tilde{\pi} + V^*\tilde{\alpha} + W^*\tilde{\beta} = 0 \tag{23e}$$

Conditions (23a)-(23b)-(23c) represent respectively primal and dual feasibility and complementary slackness for (21) at $\left(\tilde{u}, \left(\tilde{\pi}, \tilde{\lambda}\right)\right)$. We now prove that (23d)-(23e) imply the last remaining KKT condition for (21): stationarity. Replacing (23d) in (23e) we obtain:

$$D\phi(\tilde{u})^\top \tilde{\pi} + V^*\nabla\tilde{\varphi}(\tilde{v}) + W^*D\tilde{\gamma}(\tilde{w})^* \tilde{\lambda} = 0 \tag{24}$$

Next, for any (u, v, w) such that $Vu = v$ and $Wu = w$, we have $\varphi(u) = \tilde{\varphi}(v) = \tilde{\varphi}(Vu)$ and $\gamma(u) = \tilde{\gamma}(w) = \tilde{\gamma}(Wu)$, implying that

$$\nabla\varphi(u) = V^*\nabla\tilde{\varphi}(Vu) = V^*\nabla\tilde{\varphi}(v) \tag{25a}$$

$$D\gamma(u) = D\tilde{\gamma}(Wu)W = D\tilde{\gamma}(w)W \tag{25b}$$

Using (25) at $(u, v, w) = (\tilde{u}, \tilde{v}, \tilde{w})$ and replacing the expressions in (24) we obtain $D\phi(\tilde{u})^\top \tilde{\pi} + \nabla\varphi(\tilde{u}) + D\gamma(\tilde{u})^* \tilde{\lambda} = 0$, proving the result. □

5.7 Methods tested and nomenclature

We test our methods 1) On different sets $\mathcal{S}$ that can be conv.hull$(\mathcal{X})$ (V) or the intersection of $\mathcal{X}$ with cone.hull$(\mathcal{X})$ (C), lin.hull$(\mathcal{X})$ (LR) or the partitioned linear span (LP) 2) with (L) or without (NL) a linearized pricing problem 3) if considering the quadratic constraint as a linear SOC or as a classical nonlinear quadratic (CLA). The y-independency result or the tailored algorithm for Knapsack problems are always used whenever it applies. For example, the scheme using conv.hull$(\mathcal{X})$ with linearized pricing and seeing the variance constraint as a SOC will be named L-SOC-V. We summarize the different options tested in Table 3. We do not test any non-linearized scheme where the side constraint is considered

Parameter	Possibilities
$\mathcal{S}$	V, C, LR, LP
Linearized pricing	L, NL
$\mathbb{V}$ constraint	SOC, CLA

Table 3: Algorithms tested

CLA, as the pricing is still not separable and can be as hard as the original problem $P(\mathcal{X})$.

5.8 Instances generated

The algorithms are tested with $T \in \{1, 50\}$ blocks of equal sizes $n_t \in \{10^4, 10^5\}$ for every $t \in \{1, ..., T\}$. The number of auxiliary variables was fixed to $n_0 = 50$ and the number of scenarios generated to $S = 20$. The supplies u_j^t are uniformly drawn from $\{1, ..., 5\}$, and the costs and weights c_j^{ts} and a^t are uniformly drawn from $[1, 2]$. The budgets are set such that 5% of the assets can be bought, i.e. $b_t = 0.05 \cdot \left(u^t\right)^\top a^t$. For some scenario $s \in \{1, ..., S\}$, letting $\hat{x}^t$ and $\hat{y}$ be the respective optimal solutions of $\max_{x^t} \left\{ \left(c^{ts}\right)^\top x^t : x^t \in \mathcal{X}_t \right\}$ and $\max_{y} \left\{ \left(c^{0s}\right)^\top y : y \in \mathcal{X}_0 \right\}$, we set $\sigma^2 := 0.1 \cdot \left|\left|V^0\hat{y} + V\hat{x}\right|\right|_2^2$ so that the variance constraint is binding. Also, remark that the entropic risk measure has the following interpretation: for any value z and any utility random variable Z we have $E_\alpha(z - Z) = z + E_\alpha(-Z)$, meaning that we want to avoid outcomes of Z where $z - Z$ is greater than α. Given a reference random variable $\hat{Z}$ and setting $z = \mathbb{E}\left(\hat{Z}\right)$ and $\alpha = \beta \cdot \sqrt{\mathbb{V}\left(\hat{Z}\right)}$, the interpretation becomes clearer: in our case it translates into "avoid asset selections whose outcomes can make you lose more than β standard deviations, wrt to the expected solution's average return". With this observation in mind, we choose to set $\alpha_t := 0.7 \cdot \left|\left|V^t\hat{x}^t\right|\right|_2$ and $\alpha_0 := 0.7 \cdot \left|\left|V^0\hat{y}\right|\right|_2$. The vectors d^t are the returns c^t where all the components are zero, except for 60% of the assets bought in the referent $(\hat{y}, \hat{x})$, i.e. only those are considered "important" for the side constraint. Absolute and relative tolerances are respectively set to 10^{-6} and 0.1% and the runs are stopped after 6 hours.

5.9 Computational results

In Tables 4, 5, 6 and 7, we report the number of iterations (it), the total execution time (t), the master time (tmas), the pricing time (tpri), the best lower bound given by a pricing problem at any time (LB), the best upper bound given by the objective value of the last master (UB), the optimality gap (gap), and the number of variables defined by $\mathcal{S}$ ($|\mathcal{S}|$). The execution times are in seconds, the gaps in % and LB and UB are scaled wrt to the upper bound of L-CLA-V. The entries in bold font are the best of each columns, with the exception of the 't' column where it means the associated scheme went faster than the monolithic problem. If the time limit is hit during the last iteration, we report the execution time at the end of the previous one. If an algorithm stalls before giving any partial result, we mark the result line with "*".

Overall, every scheme is shown to always converge to solutions within the optimality tolerance for the small and mid-sized instances (10.000-500.000 main variables cf Tables 4, 5 and 6). Further, the execution time is mostly used to solve the master problems for the linearized schemes, and quite evenly split with the pricing time for the non-linearized schemes. We can see that using the convex hull with a linearized pricing (L-SOC-V and L-CLA-V) always performs 2-3 times faster than the monolithic model. Along with using the partitioned linear span with a linearized pricing (L-SOC-LP and L-CLA-LP), they are the only algorithms that were able to terminate successfully or return a good quality solution in the allotted time for the largest instance (cf Table 7, $T = 50$, $N = 100.000$) with 5

$\mathcal{S}$	lin	cone	it	t	tmas	tpri	LB	UB	gap	$\lvert\mathcal{S}\rvert$
V	NL	SOC	**2**	10	**0.49**	9.85	**0.9992**	**0.9992**	**0.0000**	**2**
	L	CLA	9	**5**	4.50	0.15	**0.9992**	1.0000	0.0845	9
		SOC	9	**5**	4.81	0.13	**0.9992**	1.0000	0.0845	9
C	NL	SOC	**2**	9	1.40	7.59	**0.9992**	**0.9992**	**0.0000**	**2**
	L	CLA	9	11	11.10	0.11	**0.9992**	1.0000	0.0845	9
		SOC	9	12	11.94	0.16	**0.9992**	1.0000	0.0845	9
LR	NL	SOC	**2**	11	2.92	8.34	**0.9992**	**0.9992**	**0.0000**	**2**
	L	CLA	9	18	18.27	**0.10**	**0.9992**	1.0000	0.0845	9
		SOC	9	21	20.55	0.11	**0.9992**	1.0000	0.0845	9
LP	NL	SOC	**2**	**5**	0.83	4.04	**0.9992**	**0.9992**	**0.0000**	33
	L	CLA	7	10	10.11	0.07	**0.9992**	0.9996	0.0418	208
		SOC	7	10	9.88	**0.10**	**0.9992**	0.9996	0.0418	208
Monolithic			-	8	-	-	-	**0.9992**	-	-

Table 4: Aggregated results for $T = 1$, $N = 10.000$

$\mathcal{S}$	lin	cone	it	t	tmas	tpri	LB	UB	gap	$\lvert\mathcal{S}\rvert$
V	NL	SOC	**2**	205	**3.60**	201.52	**0.9999**	**0.9999**	**0.0000**	**2**
	L	CLA	7	**29**	27.59	**0.93**	0.9998	1.0000	0.0208	7
		SOC	7	**29**	27.55	0.97	0.9998	1.0000	0.0208	7
C	NL	SOC	**2**	155	30.99	123.93	**0.9999**	**0.9999**	**0.0000**	**2**
	L	CLA	7	208	206.56	0.94	0.9998	1.0000	0.0208	7
		SOC	7	208	206.66	0.94	0.9998	1.0000	0.0208	7
LR	NL	SOC	**2**	278	105.41	172.22	**0.9999**	**0.9999**	**0.0000**	**2**
	L	CLA	7	527	525.53	0.96	0.9998	1.0000	0.0208	7
		SOC	7	527	526.32	**0.93**	0.9998	1.0000	0.0208	7
LP	NL	SOC	**2**	277	7.60	269.17	**0.9999**	**0.9999**	**0.0000**	33
	L	CLA	7	118	116.49	0.94	0.9998	1.0000	0.0158	203
		SOC	7	118	116.39	0.98	0.9998	1.0000	0.0158	203
Monolithic			-	111	-	-	-	**0.9999**	-	-

Table 5: Aggregated results for $T = 1$, $N = 100.000$

$\mathcal{S}$	lin	cone	it	t	tmas	tpri	LB	UB	gap	$\lvert\mathcal{S}\rvert$
V	NL	SOC	**2**	**481**	**22.08**	459.02	**0.9994**	**0.9994**	**0.0000**	**2**
	L	CLA	11	**339**	328.07	7.15	0.9991	1.0000	0.0858	11
		SOC	11	**345**	334.28	7.31	0.9991	1.0000	0.0858	11
C	NL	SOC	**2**	1192	719.59	471.89	**0.9994**	**0.9994**	**0.0000**	**2**
	L	CLA	11	5076	5065.19	6.66	0.9991	1.0000	0.0858	11
		SOC	11	5404	5391.71	7.12	0.9991	1.0000	0.0858	11
LR	NL	SOC	**2**	3176	2765.16	410.87	**0.9994**	**0.9994**	**0.0000**	**2**
	L	CLA	11	16221	16213.82	6.28	0.9991	1.0000	0.0858	11
		SOC	11	13514	13508.58	**5.39**	0.9991	1.0000	0.0858	11
LP	NL	SOC	**2**	2101	1561.09	538.02	**0.9994**	**0.9994**	**0.0000**	791
	L	CLA	9	6105	6094.70	5.62	0.9992	0.9996	0.0430	1763
		SOC	9	6108	6097.89	5.65	0.9992	0.9996	0.0430	1763
Monolithic			-	902	-	-	-	**0.9994**	-	-

Table 6: Aggregated results for $T = 50$, $N = 10.000$

million variables. This is due to the fact that - given the problem at hand - the two schemes are the only ones allowing to reduce substantially the size of the master problem, be it by making the $\mathcal{X}$ constraints redundant for the convex hull, or by shrinking the variable bounds into a small number of other variable bounds for the partitioned linear span.

We can see that - in our case - a linearized pricing is a crucial ingredient for a successful scheme as we can use a tailored algorithm to solve it. Also, considering the main side constraint as a SOC or a CLA does not make any difference when linearizing the pricing. Even though the non-linearized schemes are not competitive for large instances because they take longer to solve, we can see that their pricing problems provide excellent quality columns and bounds and make the schemes converge in a few iterations. Notice that using the variable aggregation scheme, the master problems are significantly bigger than the others, thus making their solution slower, although the bounds they return are also significantly better.

$\mathcal{S}$	lin	cone	it	t	tmas	tpri	LB	UB	gap	$\lvert\mathcal{S}\rvert$
V	NL	SOC	*	*	*	*	*	*	*	*
	L	CLA	7	**2303**	**2235.15**	47.09	**0.9996**	**1.0000**	**0.0444**	**7**
		SOC	7	**2304**	2236.46	46.78	**0.9996**	**1.0000**	**0.0444**	**7**
C	NL	SOC	*	*	*	*	*	*	*	*
	L	CLA	*	*	*	*	*	*	*	*
		SOC	*	*	*	*	*	*	*	*
LR	NL	SOC	*	*	*	*	*	*	*	*
	L	CLA	*	*	*	*	*	*	*	*
		SOC	*	*	*	*	*	*	*	*
LP	NL	SOC	*	*	*	*	*	*	*	*
	L	CLA	6	**20769**	20709.51	40.40	**0.9996**	1.0008	0.1153	454
		SOC	**4**	**8115**	8079.25	**25.38**	0.9978	1.0054	0.7593	228
Monolithic			-	*	-	-	-	*	-	-

Table 7: Aggregated results for $T = 50$, $N = 100.000$

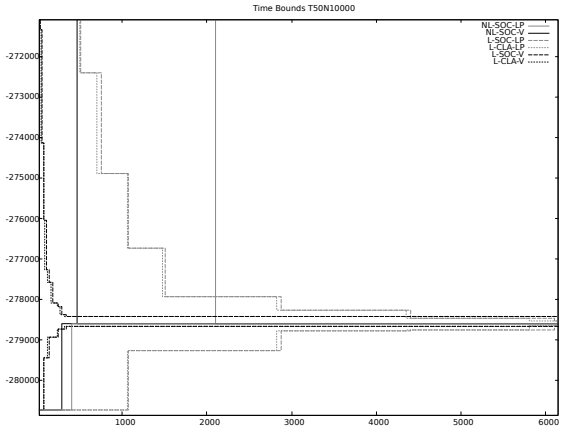

Fig. 3: Bounds Vs. Time for $T = 50$, $N = 10.000$.

In Figures 3 and 4, we show examples of progression of respectively the bounds and gaps over time of the schemes associated to V and LP on the mid-sized instance ($T = 50$, $N = 10.000$) . We can see that the bound/gap improvement is quite progressive for the linearized schemes, whereas it takes only a few large steps in the non-linearized schemes.

In Figures 5 and 6, we show examples of progression of respectively the distance between $\bar{x}^k$ and $\mathcal{S}^k$, and the largest "reduced costs" (cf Subsection 3.3) in absolute value over time of the schemes associated to V and LP on the mid-sized instance ($T = 50$, $N = 10.000$). It empirically confirms the theoretical results about the stopping criteria $\bar{x}^k \in \mathcal{S}^k$ and the zero-reduced cost one presented in Proposition 9. What is more interesting though, is the fact that to some extent these values can be used to give an idea of how close the current solution is from being optimal. We can make the same observation as for the bounds/gap, where the evolution is more progressive for the linearized schemes.

6 Conclusions and future work

We propose a generic primal decomposition method that unifies a broad range of existing schemes and opens the door for new exotic algorithmic frameworks. The convergence rate of the algorithms we present is not studied but can be heavily

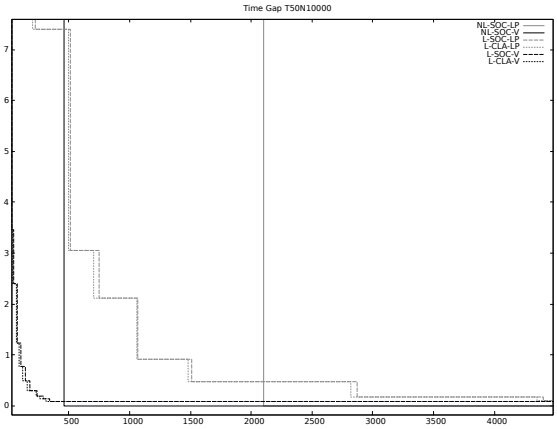

Fig. 4: Gap Vs. Time for $T = 50$, $N = 10.000$.

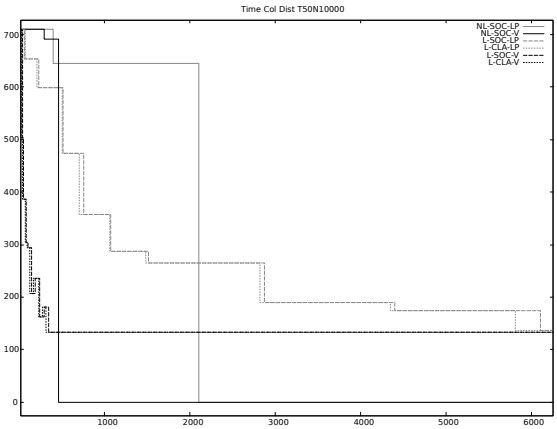

Fig. 5: Column distance to $\mathcal{S}$ Vs. Time for $T = 50$, $N = 10.000$.

problem dependent. Several special cases of our methods have been proved to converge under mild assumptions but more work is required to prove the convergence of broader classes of algorithms. Extensive computational experiments should be conducted on benchmark instances to gauge the advantages and disadvantages of each of those schemes.

Delayed column generation Through this paper, we always assumed that some structure $\mathcal{X}$ was exploitable: in an ongoing work, we explore the same kind of algorithm presented here but *relaxing all the constraints*, i.e. considering that $\mathcal{X}$ is the entire space the variables x live in. It leads to algorithms sharing similarities with delayed column generation [7] and the simplex algorithm for nonlinear problems [44]. In this case the pricing boils down to check if the reduced costs are zero and pick as an entering column the variables whose reduced cost is not, whereas the master problem can be significantly harder than in our current setting.

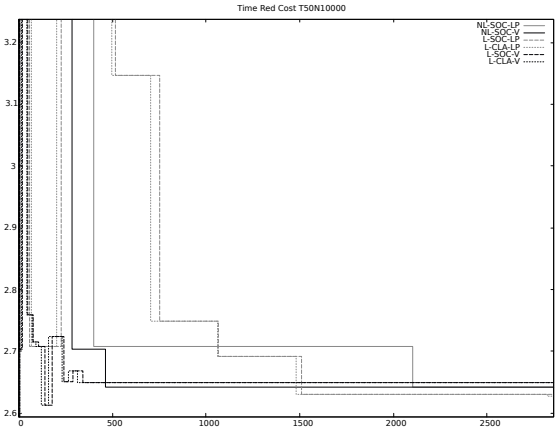

Fig. 6: Reduced Cost Vs. Time for $T = 50$, $N = 10.000$.

Non Lagrangean relaxations as pricing problems In this work, we extensively rely on Lagrangean duality, be it in the information we have access to when solving the master problem or the kind of pricing problem we solve. In an ongoing work, we link the schemes we present here with a class of stabilization techniques [27]. Different relaxations - hence dualities - can be used to provide stronger bounds and also attenuate unstable behaviors. We can use for example surrogate relaxations [24, 22] where instead of relaxing the side constraints in the objective, we aggregate all the side constraints in a single aggregated constraint $\langle \lambda, g(x, y) \rangle \geqslant 0$. The pricing problem becomes harder in general, however, it provides stronger dual bounds with weaker working hypothesis. Additionally, it becomes necessary to be able to solve the master problems *via* a *Surrogate algorithm*, i.e. that returns optimal surrogate multipliers λ. Recently, [28] provided the first general-purpose algorithm to do so.

Non-convex problems $P(\mathcal{X})$ As pointed out earlier, the set $\mathcal{X}$ can be a tractable relaxation of a combinatorial problem: the choice of the set $\mathcal{S}$ defines the kind of relaxation $\mathcal{X}$ of $\widetilde{\mathcal{X}}$ we are working on. The strength of the bounds and the difficulty to solve the problems used in our algorithms can vary greatly from one $\mathcal{S}$ to another. It was also shown that many NP-hard problems could be cast as *completely positive programs* [10], which admit several tractable relaxations on different matrix sets such as the cone of *doubly non-negative* matrices [15, 37]. Further, even though the idea is not new [1], our schemes allow to solve stronger but harder SDP relaxations of combinatorial problems [26, 23].

Dual decomposition Finally, in an ongoing work we provide *Dual decomposition schemes* where - using the tight relationship between DW and the Benders decomposition method - we present a generic constraint generation methodology where the dual variables are decomposed and generated on the fly. As in this paper, a variety of sets $\mathcal{S}$ can be used, yielding different master problemsm e.g. the Generalized Benders Decomposition [20] or the constraint aggregation schemes in [17, 36, 11].

Acknowledgements The author thanks Gonzalo Muñoz, Bernard Gendron, Fernando Ordóñez and Victor Bucarey for their valuable comments on an early version of this work. Powered@NLHPC: This research was partially supported by the supercomputing infrastructure of the NLHPC (ECM-02).

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Ahmadi, A.A., Dash, S., Hall, G.: Optimization over structured subsets of positive semidefinite matrices via column generation. Discrete Optimization **24**, 129–151 (2017)
2. Álvarez, C., Mancilla-David, F., Escalona, P., Angulo, A.: A bienstock–zuckerberg-based algorithm for solving a network-flow formulation of the convex hull pricing problem. IEEE Transactions on Power Systems **35**(3), 2108–2119 (2019)
3. Barnhart, C., Johnson, E.L., Nemhauser, G.L., Savelsbergh, M.W.P., Vance, P.H.: Branch-and-price: Column generation for solving huge integer programs. Operations research **46**(3), 316–329 (1998)
4. Ben-Tal, A., El Ghaoui, L., Nemirovski, A.: Robust optimization. Princeton University Press (2009)
5. Benders, J.F.: Partitioning procedures for solving mixed variables programming problems. Numerische mathematik **4**(1), 238–252 (1962)
6. Bergner, M., Caprara, A., Ceselli, A., Furini, F., Lübbecke, M.E., Malaguti, E., Traversi, E.: Automatic dantzig–wolfe reformulation of mixed integer programs. Mathematical Programming **149**(1-2), 391–424 (2015)
7. Bertsimas, D., Tsitsiklis, J.N.: Introduction to linear optimization, vol. 6. Athena Scientific Belmont, MA (1997)
8. Bienstock, D., Zuckerberg, M.: A new LP algorithm for precedence constrained production scheduling. Optimization Online pp. 1–33 (2009)
9. Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge university press (2004)
10. Burer, S.: On the copositive representation of binary and continuous nonconvex quadratic programs. Mathematical Programming **120**(2), 479–495 (2009)
11. Chicoisne, R., Ordoñez, F., Espinoza, D.: Risk averse shortest paths: A computational study. INFORMS Journal on Computing **30**(3), 539–553 (2018)
12. Choi, E., Tcha, D.W.: A column generation approach to the heterogeneous fleet vehicle routing problem. Computers & Operations Research **34**(7), 2080–2095 (2007)
13. Dantzig, G.B., Wolfe, P.: The decomposition algorithm for linear programs. Econometrica: Journal of the Econometric Society pp. 767–778 (1961)
14. Desaulniers, G., Desrosiers, J., Dumas, Y., Marc, S., Rioux, B., Solomon, M.M., Soumis, F.: Crew pairing at air france. European journal of operational research **97**(2), 245–259 (1997)
15. Dong, H., Anstreicher, K.: Separating doubly nonnegative and completely positive matrices. Mathematical Programming **137**(1-2), 131–153 (2013)
16. Eckstein, J., Bertsekas, D.P.: On the douglas-rachford splitting method and the proximal point algorithm for maximal monotone operators. Mathematical Programming **55**(1-3), 293–318 (1992)
17. Espinoza, D., Moreno, E.: A primal-dual aggregation algorithm for minimizing conditional value-at-risk in linear programs. Computational Optimization and Applications **59**(3), 617–638 (2014)
18. García, R., Marín, A., Patriksson, M.: Column generation algorithms for nonlinear optimization, I: Convergence analysis. Optimization **52**(2), 171–200 (2003)
19. García, R., Marín, A., Patriksson, M.: Column generation algorithms for nonlinear optimization, II: Numerical investigations. Computers & Operations Research **38**(3), 591–604 (2011)
20. Geoffrion, A.: Generalized benders decomposition. Journal of optimization theory and applications **10**(4), 237–260 (1972)

21. Giles, F.R., Pulleyblank, W.R.: Total dual integrality and integer polyhedra. Linear algebra and its applications **25**, 191–196 (1979)
22. Glover, F.: Surrogate constraint duality in mathematical programming. Operations Research **23**(3), 434–451 (1975)
23. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. Journal of the ACM (JACM) **42**(6), 1115–1145 (1995)
24. Greenberg, H., Pierskalla, W.: Surrogate mathematical programming. Operations Research **18**(5), 924–939 (1970)
25. Khaniyev, T., Elhedhli, S., Erenay, F.S.: Structure detection in mixed-integer programs. INFORMS Journal on Computing **30**(3), 570–587 (2018)
26. Lovász, L.: On the Shannon capacity of a graph. IEEE Transactions on Information theory **25**(1), 1–7 (1979)
27. Lübbecke, M., Desrosiers, J.: Selected topics in column generation. Operations research **53**(6), 1007–1023 (2005)
28. Müller, B., Muñoz, G., Gasse, M., Gleixner, A., Lodi, A., Serrano, F.: On generalized surrogate duality in mixed-integer nonlinear programming. In: International Conference on Integer Programming and Combinatorial Optimization, pp. 322–337. Springer (2020)
29. Muñoz, G., Espinoza, D., Goycoolea, M., Moreno, E., Queyranne, M., Rivera, O.: A study of the Bienstock–Zuckerberg algorithm: applications in mining and resource constrained project scheduling. Computational Optimization and Applications **69**(2), 501–534 (2018)
30. Petra, C.G., Schenk, O., Anitescu, M.: Real-time stochastic optimization of complex energy systems on high-performance computers. Computing in Science & Engineering **16**(5), 32–42 (2014)
31. Petra, C.G., Schenk, O., Lubin, M., Gärtner, K.: An augmented incomplete factorization approach for computing the schur complement in stochastic optimization. SIAM Journal on Scientific Computing **36**(2), C139–C162 (2014)
32. Pirnay, H., Lopez-Negrete, R., Biegler, L.: Optimal sensitivity based on ipopt. Mathematical Programming Computations **4**(4), 307–331 (2012)
33. Ruszczyński, A.: On convergence of an augmented lagrangian decomposition method for sparse convex optimization. Mathematics of Operations Research **20**(3), 634–656 (1995)
34. Sadykov, R., Lazarev, A., Shiryaev, V., Stratonnikov, A.: Solving a freight railcar flow problem arising in russia. In: ATMOS-13th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems-2013. Dagstuhl Open Access Series in Informatics (2013)
35. Sadykov, R., Vanderbeck, F.: Column generation for extended formulations. EURO Journal on Computational Optimization **1**(1-2), 81–115 (2013)
36. Song, Y., Luedtke, J.: An adaptive partition-based approach for solving two-stage stochastic programs with fixed recourse. SIAM Journal on Optimization **25**(3), 1344–1367 (2015)
37. Sponsel, J., Dür, M.: Factorization and cutting planes for completely positive matrices by copositive projection. Mathematical Programming **143**(1-2), 211–229 (2014)
38. Sun, Y., Andersen, M.S., Vandenberghe, L.: Decomposition in conic optimization with partially separable structure. SIAM Journal on Optimization **24**(2), 873–897 (2014)
39. Vandenberghe, L., Andersen, M.S.: Chordal graphs and semidefinite optimization. Foundations and Trends in Optimization **1**(4), 241–433 (2015)
40. Von Hohenbalken, B.: Simplicial decomposition in nonlinear programming algorithms. Mathematical Programming **13**(1), 49–68 (1977)
41. Wachter, A., Biegler, L.: On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. Mathematical Programming **106**(1), 25–57 (2006)
42. Wang, J., Ralphs, T.: Computational experience with hypergraph-based methods for automatic decomposition in discrete optimization. In: International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems, pp. 394–402. Springer (2013)
43. Wang, Y., Yin, W., Zeng, J.: Global convergence of admm in nonconvex nonsmooth optimization. Journal of Scientific Computing **78**(1), 29–63 (2019)
44. Zangwill, W.I.: The convex simplex method. Management Science **14**(3), 221–238 (1967)
45. Zheng, Y., Fantuzzi, G., Papachristodoulou, A., Goulart, P., Wynn, A.: Fast admm for semidefinite programs with chordal sparsity. In: 2017 American Control Conference (ACC), pp. 3335–3340. IEEE (2017)
46. Zheng, Y., Fantuzzi, G., Papachristodoulou, A., Goulart, P., Wynn, A.: Chordal decomposition in operator-splitting methods for sparse semidefinite programs. Mathematical Programming **180**(1), 489–532 (2020)