manuscript No.
(will be inserted by the editor)

A stochastic Lagrangian-based method for nonconvex optimization with nonlinear constraints

Dimitri Papadimitriou · **Bằng Công Vũ**

Received: date / Accepted: date

Abstract The Augmented Lagrangian Method (ALM) is one of the most common approaches for solving linear and nonlinear constrained problems. However, for non-convex objectives, handling nonlinear inequality constraints remains challenging. In this paper, we propose a stochastic ALM with Backtracking Line Search that performs on a subset (mini-batch) of randomly selected points for the solving of nonconvex problems. The considered class of problems include both nonlinear equality and inequality constraints. Together with the formal proof of the convergence properties (in expectation) of the proposed algorithm and its computational complexity, the performance of the proposed algorithm are then numerically compared against both exact and inexact state-of-the-art ALM methods. Further, we apply the proposed stochastic ALM method to solve a multi-constrained network design problem. We perform extensive numerical executions on a set of instances extracted from the SNDlib to study its behavior and performance, as well as potential improvements of this method. Then analysis and comparison of the results against those obtained by extending the method developed in [Contardo2021] to nonlinear constraints are provided for the approximation of separable nonconvex optimization programs.

Keywords Nonlinear optimization · Constrained optimization · Augmented Lagrangian · Nonconvex · Convex relaxation · Network design

Mathematics Subject Classification (2020) 65K05 · 68Q25 · 90C46 · 90C30 · 90C25

1 Introduction

Minimization problems involving both equality and inequality nonlinear constraints are of significant interest as shown by an abundant literature, e.g. [61] [36] [13] [58] to cite a few. The generic problem in nonlinear optimization is to minimize a smooth (possibly nonconvex) function $h\colon \mathbb{R}^K \to \mathbb{R}$ subject to nonlinear equality constraints and nonlinear inequality constraints. More formally,

$$\begin{aligned} &\text{minimize } h(x) \\ &\text{subject to } c_1(x) = b_1, c_2(x) \le b_2, x \in C, \end{aligned} \tag{1.1}$$

Dimitri Papadimitriou (Corresponding author)
3nLab & Universite Libre de Bruxelles (ULB)
& Belgium Research Center (BeRC), Huawei
3001 Leuven, Belgium
dpapadimitriou@3nlab.org

Bằng Công Vũ
Belgium Research Center (BeRC), Huawei
3001 Leuven, Belgium
bangcvvn@gmail.com

where c_1 and c_2 are smooth vector functions from $\mathbb{R}^K$ to $\mathbb{R}^m$, $(b_1, b_2) \in \mathbb{R}^m \times \mathbb{R}^m$ and C is a closed convex subset of $\mathbb{R}^K$. This typical problem finds applicability in mathematical optimization, in semidefinite programming and nonlinear split feasibility problems. It covers a wide class of applications in the domain of signal processing including image recovery problems [6] [58], in machine learning through various constrained problems in statistical learning as well as in operational research with, e.g., network design problems. In this context, the augmented Lagrangian-based methods (ALM) can be considered as a major breakthrough in constrained optimization providing the basis for fundamental algorithms that have been extensively studied for various classes of problems.

The main objective of this paper in this respect is to design the additional mechanisms and tools required to achieve a wider applicability of augmented Lagrangian-based methods (ALM) [23] [39] in the nonlinear setting described by the above model. Introduced by Powell and Hestenes in 1969 [49] [33], ALM alternates updates of the primal variable by minimizing the Augmented Lagrangian function and the Lagrangian multiplier by dual gradient ascent. Although the latter leads to the loss of the decomposability property, the resulting method shows improved convergence properties. Since then, this method has been subject to a vast amount of studies for the solving of both convex and nonconvex problems involving linear and nonlinear constraints. Indeed, in many of the applications described above, the optimization model turns out to include nonlinearities that the nonlinear composite problem (1) essentially captures. However, constraints are often assumed to be convex meaning that the feasible set is convex; in turn, this assumption implies that equality constraint functions must be affine and inequality constraint functions must be convex. With the proposed method, the minimization of the (possibly) nonconvex objective function h can be subject to nonlinear equality and inequality constraints without imposing convexity of its functions (or operators). Moreover, our method relies on line search that performs on a subset of randomly selected points only; hence, the stochastic ALM algorithm does not require the evaluation of all gradients (of objective function and constraints) at each iteration. This property enables, as long as the selected mini-batch verifies a well-defined minimum size criterion, the solving of larger scale nonconvex problems without compromising on convergence properties and computational complexity compared to its deterministic variant.

Following these formal developments, various numerical solving frameworks and methods based on ALM have been developed since the early 90's (and even before). As part of them, the ALGENCAN algorithmic scheme [1] [2] aims to provide a general method for solving smooth (non)convex optimization problems subject to both nonlinear equality and inequality constraints. That is, in ALGENCAN, the Augmented Lagrangian is defined not only with respect to equality constraints but also with respect to inequalities (without slack variables). Remember from this perspective that no ALM algorithm can solve such problem without assuming either the solving of nonconvex subproblems to their global minima or updating penalty sequence to remain bounded on the problem at hand. Hence, it aims at preserving the property of external penalty methods that global minimizers of the original problem can be obtained if each outer iteration computes a global minimizer of the subproblem. The general algorithm belongs to the Powell-Hestenes-Rockafellar (PHR) Augmented Lagrangian type. PHR-based Augmented Lagrangian methods are based on the iterative (approximate) minimization of the Lagrangian followed by the updating of the penalty parameter and the KKT multipliers approximations. It is a safeguarded Augmented Lagrangian method in the sense that approximations of the Lagrange multipliers are estimated at every iteration. Primal subproblems are solved using GENCAN [12]. GENCAN (that is included in ALGENCAN) is a Fortran code for minimizing a smooth function with a potentially large number of variables and box-constraints. The framework does not use matrix manipulations at all and, to enable solving large problems with moderate computer time.

More recently, several efforts have been dedicated to tackle composite nonconvex problems of the form $h(x) = f(x) + g(x)$, where f is continuously differentiable but possibly nonconvex and g is closed convex but possibly nonsmooth, subject to (possibly nonlinear) equality constraints vector function with continuously differentiable components $c(x) = 0$ [51] and (possibly nonlinear) inequality constraints $d(x) \leq 0$ [61]. For the latter, authors propose their equivalent reformulation as equality constraints $d(x) + s = 0$ by enforcing the nonnegativity of slack variable s. Moreover, ALM generally uses a sequence of penalty parameters $\{\rho_k\}$, which is nondecreasing and possibly unbounded. However, when the penalty parameter ρ_k becomes too large, the ALM subproblem can become ill-conditioned.

Therefore, instead using bounded ρ_k sequences is desirable, although for general nonconvex (and nonsmooth) problems, this condition might not be sufficient for the convergence of ALM [10, Section 2.1]. Further comparison against ALM methods is detailed in Section 5.

Alternatively, one could think of extending the applicability of the alternating direction method of multipliers (ADMM) [24] so that it can also solve Problem (1.1). This extension could be realized by adding non-negative slack variables s to the set of optimization variables. Now, it is fundamental to observe here that Problem (1.1) includes both nonlinear equality and inequality constraints. The usual trick of adding non-negative slack variables s does not transform the nature of the constraints and the complexity of the problem but only if the nonlinear constraints $c_2(x) \leq b_2$ are affine, that is $c_2(x) = Lx$. Hence, ADMM can straightforwardly deal with linear inequality constraints by adding non-negative slack variables. For nonlinear inequality constraints, the situation is completely different. Adding such equality constraints would transform the nature of the problem and the solving of its subproblems. Furthermore, transforming the constraints into indicator functions and adding them to the objective function implies in turn to compute (in every iteration) a projection onto the more complicated feasible set $\{x \mid c_2(x) \leq b_2\}$. Few papers in the literature deal with this specific issue and mostly in the convex setting [26]; therefore, we defer this study to a dedicated paper.

The generic formulation of the problem dealt with in this paper can be stated as follows.

Problem 1 Let M and K be strictly positive integers, let $(m_q)_{q=1}^M$ be a finite sequence of strictly positive integers with $\sum_{q=1}^M m_q = m < \infty$. Let $(\omega_q)_{1\leq q\leq M}$ be a sequence in $[0,1]^M$ with $\sum_{q=1}^M \omega_q = 1$. For every $q \in \{1,\dots,M\}$, let $h_q\colon \mathbb{R}^K \to \,]-\infty,+\infty]$ and $c_q\colon \mathbb{R}^K \to \mathbb{R}^{m_q}$ be smooth functions with Lipschitz continuous gradients. Let $b = (b_q)_{1\leq q\leq M} \in \oplus_{q=1}^M \mathbb{R}^{m_q}$, and S_q be a closed convex cone of $\mathbb{R}^{m_q}$. Let C be a closed convex subset of $\mathbb{R}^K$. The problem is to

$$\text{minimize } h(u) = \sum_{q=1}^M \omega_q h_q(u) \tag{1.2}$$

$$\text{subject to } (\forall q \in \{1,\dots,M\})\; c_q(u) - b_q \in S_q, u \in C. \tag{1.3}$$

In this paper, we further apply the proposed stochastic Lagrangian-based method on an operational research problem, namely, the network design problem. More precisely, given a set of finite-size point-to-point traffic demands and a network topology described by a graph, the problem is to minimize the cost of the bandwidth capacity that has to be provisioned on the individual arcs so that the network can accommodate all demands simultaneously. This problem is also referred to in the literature as the network capacity planning problem because the objective is to minimize the cost of provisioning arcs with minimum capacity compared to the traffic flow routing problem which consists of minimizing the total cost of transporting traffic units from source to destination, i.e., finding the set of individual (per arc) flows with the least total transport cost.

In its simplest form, the network design problem studied as a use case in this paper consists of minimizing the capacity provisioned on each of its arcs so that the network can simultaneously serve all incoming traffic demands at minimum cost. It refers to the situation where demand values are certain, i.e., they are not subject to fluctuations and variations, and flows defined by continuous real variables. The resulting cost minimization problem can be solved in polynomial time. However, minimizing the network design cost comes nowadays with additional constraints to the usual demand satisfaction, flow conservation, and capacity (linear and convex) constraints. Recent developments have motivated the need to also accommodate various nonlinear constraints such as delay constraints and other congestion constraints. Accommodating nonlinear constraints changes the very nature of the original problem. In turn, these requirements make the solving of these optimization problems more computationally challenging even when limited to the static case (i.e., with fixed traffic demand matrices). Similar reasoning can be drawn for the minimum cost multi-commodity network flow (MCF) problems. The variant involving continuous (real) flow variables is solvable in polynomial time by an LP solver, whereas its integer flow counterpart is NP-hard. Here again, the (transit) time-constrained

MCF problem is NP-hard and the complexity of fractional MCF over time is NP-hard.

Contribution: The main contribution of this paper is threefold.

- First, we propose a stochastic Augmented Lagrangian Method (ALM) method relying on Backtracking Line Search that performs on a subset (mini-batch) of randomly selected points to solve optimization problems involving the minimization of a smooth (possibly nonconvex) objective function subject to both nonlinear equality and inequality constraints. The convergence properties (in expectation) of the proposed algorithm are then thoroughly demonstrated under very general assumptions. The main features of the proposed algorithm compared to [61] [51] are the following. Firstly, it is structured as a *single-loop* algorithm; more precisely, it does not require calling a first-order method (such as proximal gradient descent) to compute inner iterates. For instance, [61] further involves the use of an intermediate interior Proximal Point (iPP) method to approximately solve the primal subproblems of the ALM. Secondly, since performing on a mini-batch whose size is $\ll M$, the proposed algorithm does not require the evaluation of all gradients (of the objective function and constraints) at each iteration. Thirdly, it uses the *backtracking line search* technique to find both primal and dual stepsize.
- Second, from the modeling perspective, we define a delay constrained network design problem, which has received relatively limited attention in the network optimization community due to its nonlinear nature. Let us cite, among others, [8] which approaches the problem from the MCF formulation perspective with the objective of minimizing the total routing cost while assuming that the routing path sets are given as input, and its robust variant [31]. We then propose a reformulation of this network optimization problem that can be solved by means of the stochastic ALM with Backtracking Line Search.
- Finally, we provide extensive numerical executions of the proposed algorithm and compare the results with those obtained by means of an extension to nonlinear constraints of the method developed in [19] for the approximation of separable nonconvex optimization programs. The executions are performed using real datasets extracted from the SNDlib library [44] to determine the performance and applicability but also the potential improvements to these methods.

Structure: The remainder of this paper is structured as follows. After introducing in Section 2 the preliminary notations and definitions used throughout this paper, the proposed algorithm, namely, the stochastic ALM with Backtracking Line Search is specified in Section 3.1. Its convergence properties are thoroughly detailed in Sections 3.2 and 3.3, which determine the conditions for local convergence (in expectation) of the sequences produced by the proposed Algorithm to a critical point of the augmented Lagrangian function. Section 4 characterizes the iteration complexity of the proposed Algorithm together with its formal proof. Next, the comparison against inexact ALM methods [61] [51] is fully documented in Section 5. This section also reports numerical experiments that corroborate the empirical performance of the proposed algorithm compared to the one developed in [61] and ALGENCAN [2]. Section 6 details the use case considered for its numerical evaluation, that is the multi-constrained network design problem, i.e., finding the capacity to provision on each of its arcs such that the network can serve all incoming traffic demands simultaneously at minimum cost, where each demand is specified with a maximum delay that can be incurred by the individual arcs on the traffic flows traversing the network (a.k.a. load-induced delay). Numerical experiments performed using the proposed algorithm are detailed in Section 6.3. The results are analyzed and compared against the piecewise linear relaxation method documented in Section 6.4. Finally, a concluding Section 7 summarizes the findings of this paper together with the research topics identified for future work.

2 Preliminaries

Notations. Denote by $\Gamma_0(\mathbb{R}^K)$ the class of all proper lower semicontinuous convex functions from $\mathbb{R}^K$ to $]-\infty,+\infty]$. The proximity operator of $f \in \Gamma_0(\mathbb{R}^K)$ is

$$\operatorname{prox}_f \colon \mathbb{R}^K \to \mathbb{R}^K \colon x \mapsto \underset{y\in\mathbb{R}^K}{\operatorname{argmin}}\, f(y) + \frac{1}{2}\|x-y\|^2.$$

The conjugate function of f is denoted by f^*. When f is the indicator function of some closed convex $S \subset \mathbb{R}^K$, which is denoted by ι_S,

$$\iota_S \colon x \mapsto \begin{cases} 0 & \text{if } x \in S \\ +\infty & \text{if } x \notin S, \end{cases}$$

the proximity operator of f reduces to the projection operator denoted by P_S. The distance from $x \in \mathbb{R}^K$ to S is $d_S(x) = \|x - P_S x\|$. Note that the conjugate function of ι_S is the support function of S and is denoted by σ_S. The normal cone operator of some closed convex set C is N_C. When S is a closed convex cone, the polar cone $S^\ominus$ of S is defined as $S^\ominus = \big\{u \mid \sup\langle S \mid u\rangle \leq 0\big\}$.

Let $g \colon \mathbb{R}^K \times \mathbb{R}^m \to]-\infty,+\infty]$ be a differentiable function. We denote by $\nabla_1 g$ the gradient of g with respect to the first variable when the second variable is fixed. The notation $\nabla_2 g$ is defined similarly. Let $c \colon \mathbb{R}^K \to \mathbb{R}^m$ be a differentiable (smooth) mapping, the Jacobian of c at $u \in \mathbb{R}^K$ is denoted by $J_c(u)$ and its conjugate is denoted by $J_c(u)^\intercal$. Let $\nu > 0$, the class of all smooth mappings $c \colon \mathbb{R}^K \to \mathbb{R}^m$ with ν-Lipschitzian Jacobian is denoted by $\mathcal{C}^1_\nu(\mathbb{R}^K, \mathbb{R}^m)$.
The development of this paper relies on the following definitions.

Definition 1 Let M be a strictly positive integer. Let $(\omega_q)_{1\leq q\leq M}$ be a sequence in $[0,1]^M$ with $\sum_{q=1}^M \omega_q = 1$. The weighted inner product on the Hilbert space V, maps each pairs of vectors $(y,v) =\in V \times V$ to the scalar $\langle \cdot \,||\, \cdot\rangle$ defined as

$$\langle \cdot \,||\, \cdot\rangle \colon (y,v) \mapsto \sum_{q=1}^M \omega_q \langle v_q \mid y_q\rangle \tag{2.1}$$

$$\text{with vector norm } |||\cdot||| \colon v \mapsto \sqrt{\langle v \,||\, v\rangle}, \tag{2.2}$$

where $y = (y_q)_{1\leq q\leq M}$ and $v = (v_q)_{1\leq q\leq M}$.

Definition 2 [16] Let $f \in \Gamma_0(\mathbb{R}^K)$, $g \in \Gamma_0(\mathbb{R}^m)$, $c \in \mathcal{C}^1_\nu(\mathbb{R}^K, \mathbb{R}^m)$, and $b \in \mathbb{R}^m$. A vector $d \in \mathbb{R}^K$ defines a descent direction of $\varphi \mapsto f(u) + g(c(u) - b)$ at u, if the difference $\Delta_0\varphi(u;d)$ verifies the strict inequality

$$\Delta_0\varphi(u;d) = f(u+d) + g\big(c(u) - b + J_c(u)d\big) - \varphi(x) < 0, \tag{2.3}$$

where $J_c(u)$ denotes the Jacobian of the function c at u. A method for which, at each iteration k, the descent direction d_k, at current point u_k, verifies the strict inequality $\Delta_0\varphi(u_k;d_k) < 0$ is referred to as a descent method.

Definition 3 Let $g \in \Gamma_0(\mathbb{R}^m)$, let $b \in \mathbb{R}^m$ and $\mathcal{C}^1_\nu(\mathbb{R}^K, \mathbb{R}^m) \ni c \colon u \mapsto c(u) - b$. For every $\rho \in]0,+\infty[$, and $(u,\lambda) \in \mathbb{R}^K \times \mathbb{R}^m$, the smooth approximation of $g(c(\cdot) - b)$ is defined by

$$g_\rho \colon (u,\lambda) \mapsto \sup_{y\in\mathbb{R}^m} \Big(\langle c(u) - b \,||\, y\rangle - g^\star(y) - \frac{1}{2\rho}|||y-\lambda|||^2\Big), \tag{2.4}$$

where ρ is referred to as the smoothing parameter and $g^\star$ denotes the Fenchel conjugate of the function g defined by $g^\star \colon u \mapsto \sup_{x\in\mathbb{R}^m} \big(\langle u \,||\, x\rangle - g(x)\big)$.

The function g_ρ is a smooth approximation of g, which is known as the smoothing technique. Various numerical methods have been developed by means of this technique; see, for instance, [43,47,7]. Several examples where g_β admits a closed-form expression can be found in [43,5]. We recall the following result concerning the differentiability of g_ρ.

Lemma 1 *For every $\rho > 0$, let the function g_ρ be defined by (2.4). Then, g_ρ is a differentiable function with respect to the variable u, and, for every $(u, \lambda) \in \mathbb{R}^K \times \mathbb{R}^m$,*

$$\nabla_1 g_\rho(u, \lambda) = (J_c(u))^\intercal \operatorname{prox}_{\rho^{-1}g^\star}(\rho^{-1}(c(u) - b) + \lambda), \tag{2.5}$$

where $(J_c(u))^\intercal$ is the (conjugate) transpose of the linear operator $J_c(u)$.

We extend this result to the case where the function g admits a separable structure. More precisely, we have the following result.

Lemma 2 *Let $g_q = \iota_{S_q}$, where ι_{S_q} denotes the indicator function of the closed convex subset S_q of $\mathbb{R}^{m_q}$, and define the function $g\colon (v_q)_{1\leq q\leq M} \mapsto \sum_{q=1}^{M} \omega_q g_q(v_q)$, where $(\omega_q)_{1\leq q\leq M}$ denotes a sequence in $[0,1]^M$ with $\sum_{q=1}^{M} \omega_q = 1$. Then, for every $\rho > 0$ and for every $(u, \lambda) \in \mathbb{R}^K \times \mathbb{R}^m$,*

$$g_\rho(u, \lambda) = \sum_{q=1}^{M} \omega_q g_{\rho,q}(u, \lambda_q), \tag{2.6}$$

$$\text{where } g_{q,\rho}(u, \lambda_q) = \sup_{y_q\in\mathbb{R}^{m_q}} \big(\langle c_q(u) - b_q \mid \lambda_q\rangle - g_q^\star(y_q) - \frac{1}{2\rho}\|y_q - \lambda_q\|^2\big), \tag{2.7}$$

is a differentiable function whose gradient with respect to the first variable u is given by

$$\nabla_1 g_\rho(u, \lambda) = \rho \sum_{q=1}^{M} \omega_q (J_{c_q}(u))^\intercal \Big(c_q(u) - b_q + \rho^{-1}\lambda_q - P_{S_q}\big(c_q(u) - b_q + \rho^{-1}\lambda_q\big)\Big). \tag{2.8}$$

Proof. Following Definition 3, the conjugate $g^\star$ of the function g can be expressed as

$$\begin{aligned} g^\star\colon v \mapsto \sup_{y\in\mathbb{R}^m} (\langle v \mid\mid y\rangle - g(y)) &= \sup_{(y_q)_{1\leq q\leq M}\in\mathbb{R}^m} \sum_{q=1}^{M} (\omega_q \langle v_q \mid y_q\rangle - \omega_q g_q(y_q)) \\ &= \sum_{q=1}^{M} \omega_q g_q^\star(v_q). \end{aligned} \tag{2.9}$$

Therefore, the smooth approximation of g with parameter ρ, $g_\rho(u, \lambda)$, is defined by

$$g_\rho(u, \lambda) = \sup_{y\in\mathbb{R}^m} \big(\langle c(u) - b \mid\mid y\rangle - g^\star(y) - \frac{1}{2\rho}|||y - \lambda|||^2\big) \tag{2.10}$$

$$\begin{aligned} &= \sum_{q=1}^{M} \omega_q \sup_{y_q\in\mathbb{R}^{m_q}} \big(\langle c_q(u) - b_q \mid y_q\rangle - g_q{}^\star(y_q) - \frac{1}{2\rho}\|y_q - \lambda_q\|^2\big) \\ &= \sum_{q=1}^{M} \omega_q g_{q,\rho}(u, \lambda_q), \end{aligned} \tag{2.11}$$

which proves (2.6). Next, it follows from (2.6) and Lemma 1 that

$$\begin{aligned} \nabla_1 g_\rho(u, \lambda) &= \sum_{q=1}^{M} \omega_q \nabla_1 g_{q,\rho}(u, \lambda_q) \\ &= \sum_{q=1}^{M} \omega_q (J_{c_q}(u))^\intercal \operatorname{prox}_{\rho g_q^\star}(\rho(c_q(u) - b_q) + \lambda_q) \\ &= \rho \sum_{q=1}^{M} \omega_q (J_{c_q}(u))^\intercal \Big(c_q(u) - b_q + \rho^{-1}\lambda_q - P_{S_q}\big(c_q(u) - b_q + \rho^{-1}\lambda_q\big)\Big), \end{aligned} \tag{2.12}$$

where the last equality follows from the Moreau's identity ($\mathrm{prox}_f(x)+\mathrm{prox}_{f^\star}(x)=x$) and the property $\mathrm{prox}_{\iota_{S_q}} = P_{S_q}$. □

Lemma 3 *Let $\lambda_q \in \mathbb{R}^{m_q}$ and $\rho \in]0,+\infty[$. Let $(g_q)_{1\leq q\leq M}$ be defined as Lemma 2. Let*

$$e_{q,\rho}\colon u \mapsto P_{S_q}(c_q(u)-b_q+\rho^{-1}\lambda_q). \tag{2.13}$$

Then, for every $(u,\lambda)\in\mathbb{R}^K\times\mathbb{R}^m$,

$$g_{q,\rho}(u,\lambda_q) = \langle c_q(u)-b_q-e_{q,\rho}(u) \mid \lambda_q\rangle + \frac{\rho}{2}\|c_q(u)-b_q-e_{q,\rho}(u)\|^2 \tag{2.14}$$

$$= \frac{\rho}{2}d^2_{S_q}(c_q(u)-b_q+\rho^{-1}\lambda_q) - \frac{1}{2\rho}\|\lambda_q\|^2, \tag{2.15}$$

where $d_{S_q}\colon v_q \mapsto \|v_q - P_{S_q}v_q\|$ defines the distance function d_{S_q}.

Proof. Let us define $\lambda_q^\dagger := \lambda_q + \rho(c_q(u)-b_q-e_{q,\rho}(u))$. Then, the Moreau's identity gives

$$\lambda_q^\dagger = \mathrm{prox}_{\rho\sigma_{S_q}}\big(\lambda_q+\rho(c_q(u)-b_q)\big) \text{ and } \sigma_{S_q}(\lambda_q^\dagger) = \big\langle \lambda_q^\dagger \mid e_{q,\rho}(u)\big\rangle\,. \tag{2.16}$$

Therefore, it follows from the definition of $g_{q,\rho}$ (see Definition 3) that

$$\begin{aligned}
g_{q,\rho}(u,\lambda_q) &= \sup_{v_q\in\mathbb{R}^{m_q}} \big(\langle c_q(u)-b_q \mid v_q\rangle - g_q^\star(v_q) - \frac{1}{2\rho}\|v_q-\lambda_q\|^2\big)\\
&= \big\langle c_q(u)-b_q \mid \lambda_q^\dagger\big\rangle - g_q^\star(\lambda_q^\dagger) - \frac{1}{2\rho}\|\lambda_q^\dagger-\lambda_q\|^2\\
&= \big\langle c_q(u)-b_q-e_{q,\rho}(u) \mid \lambda_q^\dagger\big\rangle - \frac{1}{2\rho}\|\lambda_q^\dagger-\lambda_q\|^2\\
&= \langle c_q(u)-b_q-e_{q,\rho}(u) \mid \lambda_q\rangle + \frac{\rho}{2}\|c_q(u)-b_q-e_{q,\rho}(u)\|^2,
\end{aligned} \tag{2.17}$$

which proves (2.14). Next, we have

$$\rho\big\langle c_q(u)-b_q-e_{q,\rho}(u) \mid \rho^{-1}\lambda_q\big\rangle = \frac{\rho}{2}\|c_q(u)-b_q-e_{q,\rho}(u)+\rho^{-1}\lambda_q\|^2 - \frac{\rho}{2}\|c_q(u)-b_q-e_{q,\rho}(u)\|^2 - \frac{\rho}{2}\|\rho^{-1}\lambda_q\|^2,$$

which implies that

$$g_{q,\rho}(u,\lambda_q) = \frac{\rho}{2}\|c_q(u)-b_q-e_{q,\rho}(u)+\rho^{-1}\lambda_q\|^2 - \frac{1}{2\rho}\|\lambda_q\|^2 \tag{2.18}$$

$$= \frac{\rho}{2}d^2_{S_q}(c_q(u)-b_q+\rho^{-1}\lambda_q) - \frac{1}{2\rho}\|\lambda_q\|^2, \tag{2.19}$$

where the last equality follows from the definition of d_{S_q}. Hence, (2.15) is verified. □
Let $(u,\lambda)\in\mathbb{R}^K\times\mathbb{R}^m$ and $\rho>0$. By using (2.11) and (2.19)

$$g_\rho(u,\lambda) = \sum_{q=1}^M \omega_q g_{q,\rho}(u,\lambda_q),$$

$$\text{where } g_{q,\rho}(u,\lambda_q) := \iota_{S_q,\rho}(u,\lambda_q) = \frac{\rho}{2}d^2_{S_q}(c_q(u)-b_q+\rho^{-1}\lambda_q) - \frac{1}{2\rho}\|\lambda_q\|^2,$$

we can define the smooth approximation of the augmented objective function $\mathcal{L}_\rho$ by involving the indicator functions $\iota_{S_q,\rho}(u,\lambda_q)$ as follows

$$\mathcal{L}_\rho\colon (u,\lambda)\mapsto \sum_{q=1}^M \Big(\omega_q h_q(u) + \frac{\rho\omega_q}{2}d^2_{S_q}(c_q(u)-b_q+\rho^{-1}\lambda_q)\Big) - \frac{1}{2\rho}|||\lambda|||^2. \tag{2.20}$$

Moreover, assuming the smoothing parameter ρ_k and multiplier λ_k are given at iteration k, one can define the function φ_k by

$$\varphi_k\colon u \mapsto \mathcal{L}_{\rho_k}(u, \lambda_k) = h(u) + \psi_k \circ c(u), \tag{2.21}$$

where $\circ$ denotes the function composition, and

$$\psi_k\colon (w_q)_{1\leq q\leq M} \mapsto \sum_{q=1}^{M} \frac{\rho\omega_q}{2} d^2_{S_q}(w_q - b_q + \rho^{-1}\lambda_{k,q}) - \frac{1}{2\rho}|||\lambda_k|||^2. \tag{2.22}$$

The following Lemma generalizes the definition of the descent direction d_k to nonconvex functions φ_k. This result is obtained by defining the function φ_k as the composition of a convex and a nonconvex function set as the argument of the former (convex) function.

Lemma 4 *Assume* $\overline{c}: \mathbb{R}^K \to \mathbb{R}^m \times\,]-\infty,+\infty] : u \mapsto \overline{c}(u) = (c(u), c_0(u))$ *together with* $c\colon \mathbb{R}^K \to \mathbb{R}^m\colon u \mapsto c(u)$ *and* $c_0 = h$*. Define the function* $\Psi_k\colon \mathbb{R}^K \times \mathbb{R} \to \mathbb{R}\colon (u,\xi) \mapsto \psi_k(u) + \mathrm{Id}_R(\xi)$*, where* $\mathrm{Id}_R : \mathbb{R} \ni \xi \mapsto \xi$*.*
If the function $\psi_k\colon \mathbb{R}^m \to\,]-\infty,+\infty]$ *is convex; then, the function* Ψ_k *is convex. The composition* $(\Psi_k \circ \overline{c})$ *verifies the identity*

$$(\Psi_k \circ \overline{c})(u) = \psi_k \circ c(u) + \mathrm{Id}_R \circ c_0(u) \equiv \varphi_k(u), \tag{2.23}$$

where φ_k *is defined by* (2.21)*. Moreover, by defining, for every* $u \in \mathrm{dom}(\varphi)$ *and* $d \in \mathbb{R}^K$*,*

$$\Delta\varphi_k(u;d) = \psi_k(c(u) + J_c(u)d) + \langle \nabla h(u) \mid d\rangle - \psi_k(c(u)), \tag{2.24}$$

the following identity is verified

$$\Delta_0(\Psi_k \circ \overline{c})(u;d) \equiv \Delta\varphi_k(u;d). \tag{2.25}$$

Proof. The proof follows the same reasoning as the one used when $h \equiv f + g$, see [46]. For the sake of completeness, we reproduce it here with this setting. The function $\Psi_k(u,\xi)$ defined by $\psi_k(u) + \mathrm{Id}_R(\xi)$ is convex since the identity function on $\mathbb{R}$ is convex, by assumption, the function $\psi_k(u)$ is convex, and the sum of two convex functions is again convex. The expression (2.23) follows from the definition of composition functions. Let us now prove (2.25). By definition of Δ_0 in (2.3), we get

$$\Delta_0(\Psi_k \circ \overline{c})(u;d) = \Psi_k(\overline{c}(u) + J_{\overline{c}}(u)d) - \Psi_k(\overline{c}(u))$$

By expanding the last equality using the definition of $(\Psi_k \circ \overline{c})(u;d)$ given by (2.23), we obtain

$$\begin{aligned}\Delta_0(\Psi_k \circ \overline{c})(u;d) &= \psi_k(c(u) + J_c(u)d) + \mathrm{Id}_R(c_0(u) + \langle \nabla c_0(u) \mid d\rangle) - \mathrm{Id}_R \circ c_0(u) - \psi_k \circ c(u)\\ &= \psi_k(c(u) + J_c(u)d) + \mathrm{Id}_R \circ \langle \nabla c_0(u) \mid d\rangle - \psi_k \circ c(u)\end{aligned}$$

Then, since $c_0\colon u \mapsto h(u)$ and the scalar product $\langle \nabla c_0(u) \mid d\rangle \in \mathbb{R}$, we deduce the expression

$$\Delta_0(\Psi_k \circ \overline{c})(u;d) = \psi_k(c(u) + J_c(u)d) + \langle \nabla h(u) \mid d\rangle - \psi_k(c(u)) \equiv \Delta\varphi_k(u;d), \tag{2.26}$$

which completes of the proof. □

Using Lemma 3, one can then prove that at each iteration k the descent direction computer at u_k verifies the strict inequality $\Delta\varphi(u_k; d_k) < 0$; hence, it can be referred to as defining a descent method.

We recall the basic properties of the projection operator onto the nonempty closed convex subset S_q denoted P_{S_q}, that will be used in Section 3.1.

Lemma 5 [5, Proposition 29.3, Theorem 3.16] *Let* $q \in \{1,\ldots,M\}$*, let* S_q *be a non-empty closed convex subset in* $\mathbb{R}^{m_q}$ *and* $S = \prod_{q=1}^M S_q$*. Then, the following hold.*

(i) *For any* $v = (v_q)_{q=1}^M \in \oplus_{q=1}^M \mathbb{R}^{m_q}$*,* $P_S v = (P_{S_q} v_q)_{1\leq q\leq M}$*.*

(ii) *For any* $v = (v_q)_{q=1}^{M} \in \oplus_{q=1}^{M} \mathbb{R}^{m_q}$,

$$p = P_S v \Longleftrightarrow (\forall w \in S)\ \langle v - p \mid\mid w - p \rangle \leq 0. \tag{2.27}$$

Let $(\Omega, \mathcal{F}, \mathrm{Prob})$ be a probability space and $\mathcal{H} = \mathbb{R}^K$. A $\mathcal{H}$-valued random variable is a measurable function $X : \Omega \to \mathcal{H}$, where $\mathcal{H}$ is endowed with the Borel σ-algebra. We denote by $\sigma(X)$ the σ-field generated by X. The expectation of a random variable X is denoted by $\mathbf{E}[X]$. The conditional expectation of X given a σ-field $\mathcal{A} \subset \mathcal{F}$ is denoted by $\mathbf{E}[X|\mathcal{A}]$. A $\mathcal{H}$-valued random process is a sequence $(x_k)_{k\in\mathbb{N}}$ of $\mathcal{H}$-valued random variables. The abbreviation a.s. stands for 'almost surely'.

Lemma 6 ([62, Theorem 1]) *Let $(\mathcal{F}_k)_{k\in\mathbb{N}}$ be an increasing sequence of sub-σ-algebras of $\mathcal{F}$, let $(z_k)_{k\in\mathbb{N}}$, $(\theta_k)_{k\in\mathbb{N}}$, $(\zeta_k)_{k\in\mathbb{N}}$ and $(t_k)_{k\in\mathbb{N}}$ be sequences of $[0,+\infty[$-valued random variables such that, for every $k \in \mathbb{N}$, z_k, θ_k, ζ_k and t_k are $\mathcal{F}_k$-measurable. Moreover, assume that $\sum_{k\in\mathbb{N}} t_k < +\infty$, $\sum_{k\in\mathbb{N}} \zeta_k < +\infty$ a.s. and*

$$(\forall k \in \mathbb{N})\ \mathbf{E}[z_{k+1}|\mathcal{F}_k] \leq (1 + t_k) z_k + \zeta_k - \theta_k \ \ a.s..$$

Then $(z_k)_{k\in\mathbb{N}}$ converges a.s. to a $[0,+\infty[$-valued random variable and $(\theta_k)_{k\in\mathbb{N}}$ is summable a.s..

Corollary 1 ([64, Corollary 2.6]) *Let $(\mathcal{F}_k)_{k\in\mathbb{N}}$ be an increasing sequence of sub-σ-algebras of $\mathcal{F}$, let $(x_k)_{k\in\mathbb{N}}$ be a $[0,+\infty[$-valued random sequence such that, for every $k \in \mathbb{N}$, x_{k-1} is $\mathcal{F}_k$-measurable and*

$$\sum_{k\in\mathbb{N}} \mathbf{E}[x_k|\mathcal{F}_k] < +\infty \quad a.s.. \tag{2.28}$$

Then, $\sum\limits_{k\in\mathbb{N}} x_k < +\infty$ *a.s..*

3 Algorithm and Convergence

In this section, we detail the specification of Algorithm 1 for solving Problem 1. The main design principles of this single-loop algorithm can be summarized as follows:

(i) Formulate a generalization of the augmented Lagrangian function by *smoothing the nonlinear constraints* $c_q(u) - b_q \in S_q$. This function is the sum of smoothed functions with respect to the primal variable u and the dual variable λ.

(ii) Then, given a point u and the Lagrangian multiplier λ, we apply the *projected stochastic mini-batch gradient* to update the primal variable as $u^+ = P_C(u - t_k d_k)$, where t_k is the primal stepsize and d_k is the mini-batch stochastic gradient; provided the size of the mini-batch satisfies a well-defined minimum size criteria.

(iii) We use the *backtracking technique* to find the stepsizes t_k and σ_k. Then, the update of the dual variable λ is performed as $\lambda^+ = \lambda + \sigma_k \nabla_2 \mathcal{L}_\rho(u^+, \lambda)$.

Thus, this algorithm does not involve any subsolver or auxiliary solver to compute the values of primal or dual variables; hence, it is referred to as a single-loop algorithm.

The stochastic gradient method was first introduced in [63]. This method as well as its extension, the stochastic proximal gradient method, have been widely adopted nowadays as optimization method in machine learning (statistical learning, deep learning, etc.), linear inverse problem, and game theory; see [4,14,15,42,32] for examples. A main feature of the stochastic gradient is that it uses only one sample point per iteration compared to the full gradient whose computational cost becomes prohibitive when the number of points of points is large. Nevertheless, the stochastic gradient does not guarantee convergence of the iterates without either ensuring the sequence of stepsizes decreases (leading to a decreasing stepsize method) or involving a variance reduction technique. A relaxation consists of using a minibatch approach where only a subset of samples is used per iteration. This idea leads to the minibatch stochastic gradient; see [15] for detailed development. The major advantage of the mini-batch stochastic gradient is the reduction of variance when the minibatch size increases [15,38,21].

Further comparison against inexact augmented Lagrangian methods such as [61] and [51] is provided in Section 5.

Further, we characterize the convergence properties of the sequences $(u_k, \lambda_k)_{k\in\mathbb{N}}$ generated by the proposed algorithm. For this purpose, we suppose that the Jacobian J_c of the constraints c verifies the following assumption.

Assumption 1 *Let C a closed convex subset of $\mathbb{R}^K$. Assume*

$$\mu_0 = \sup_{u\in C} \|J_c(u)^\intercal\| < +\infty \quad and \quad (\forall (u,\tilde{u}) \in C\times C)\ \|J_c(u) - J_c(\tilde{u})\| \leq \mu_c\|u-\tilde{u}\|, \tag{3.1}$$

where μ_c is a positive constant.

We further assume that the variance of d_{k,i_p} in (3.7) denoted by $\mathrm{Var}(d_{k,i_p})$ is bounded. More precisely, we need the following.

Assumption 2 *Let i_p be a random variable with probability* $\mathrm{Prob}(i_p = q) = \omega_q$*. Let d_{k,i_p} be defined by Step 2 of Algorithm 1. Assume that for all $k\in\mathbb{N}$,*

$$\mathrm{Var}(d_{k,i_p}) = \mathbf{E}_{i_p}[\|d_{k,i_p} - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2|\mathcal{E}_k] \leq \overline{\sigma}_k^2 < +\infty, \tag{3.2}$$

where $\mathcal{E}_k$ is the $\sigma-$algebra generated by $u_0, u_1, \ldots, u_k$.

Consequently the (sample) variance of the estimator of the descent direction $d_k \in \mathbb{R}^K$ is also bounded. More precisely,

$$\mathrm{Var}(d_k) \leq \frac{1}{\mathtt{m}_k^2}\sum_{p=1}^{\mathtt{m}_k} \mathrm{Var}(d_{k,i_p}), \tag{3.3}$$

where $\mathtt{m}_k$ denotes the size of the sample. Given $\lambda_k \in S^\ominus$ and $\xi_k = (i_p)_{1\leq p\leq \mathtt{m}_k}$. Define

$$f_{\lambda_k,\xi_k}(\cdot) = \frac{1}{\mathtt{m}_k}\sum_{p=1}^{\mathtt{m}_k}\Big(h_{i_p}(\cdot) + \frac{\rho_k}{2}d^2_{S_{i_p}}(c_{i_p}(\cdot) - b_{i_p} + \rho_k^{-1}\lambda_{k,i_p}) - \frac{1}{2\rho_k}\|\lambda_{k,i_p}\|^2\Big). \tag{3.4}$$

In the remainder of this paper, ℓ_{ξ_k} refers to the Lipschitz constant of $\nabla f_{\lambda_k,\xi_k}$. The Lipschitz constant of $\nabla\mathcal{L}_{\rho_k}(\cdot,\lambda_k)$ is denoted by ℓ_k. Recall also that $S = \prod_{q=1}^M S_q$.

3.1 Algorithm

Algorithm 1 ALM algorithm with backtracking

▷ **Initialization**

Set $u_0 \in C$, $u_{-1} \neq u_0$, $\lambda_0 \in S^{\ominus}$

Set $\sigma_{-1} \gg 1$, $\rho_{-1} \in]0,\infty[$, $1 \gg \varepsilon > 0$, $(\theta,\nu) \in]0,1[^2$, $n \in \mathbb{N}$

Compute μ_0 from (3.1)

▷ **Main Loop**

for $k \leftarrow 0 : n$ **do**

▷ **Step 1**

Select $\rho_k \in]0,\infty[$ such that

$$\begin{cases} \beta_k := 1 - \dfrac{\rho_k}{2}\|J_c(u_k)\|^2 > \varepsilon \\ \sqrt{\rho_k}\|\|c(u_k) - b - P_S\big(c(u_k) - b + \rho_k^{-1}\lambda_k\big)\|\| \leq \min_{1\leq i\leq k}\|u_i - u_{i-1}\| \\ \rho_k < \rho_{k-1} + \varepsilon\sigma_{k-1} \end{cases} \tag{3.5}$$

▷ **Step 2**

Select mini-batch size $\mathtt{m}_k \in \mathbb{N}$.

Generate $\mathtt{m}_k$ random variables $\xi_k = (i_p)_{1\leq p\leq \mathtt{m}_k}$ with $\mathrm{Prob}(i_p = q) = \omega_q$

Compute v_{k,i_p} and d_{k,i_p}

$$v_{k,i_p} = \lambda_{k,i_p} + \rho_k\big(c_{i_p}(u_k) - b_{i_p} - P_{S_{i_p}}(c_{i_p}(u_k) - b_{i_p} + \rho_k^{-1}\lambda_{k,i_p})\big) \tag{3.6}$$

$$d_{k,i_p} = \nabla h_{i_p}(u_k) + J_{c_{i_p}}(u_k)^{\intercal} v_{k,i_p} \tag{3.7}$$

Compute d_k

$$d_k = \frac{1}{\mathtt{m}_k}\sum_{p=1}^{\mathtt{m}_k} d_{k,i_p} \tag{3.8}$$

▷ **Step 3**: Find $\{t_k = \theta^j\ (j \in \mathbb{N}), \sigma_k \leq \rho_k\}$ such that

$$\nu\beta_k - \frac{1}{2}t_k(1 + t_k\ell_{\xi_k})^2 - t_k\big(1 + t_k\ell_k\big)^2 - 2(1+\varepsilon)t_k - \varsigma_{1,k} \geq \frac{\varepsilon}{2} \tag{3.9}$$

$$\text{where } \varsigma_{1,k} := (1+\varepsilon)4\sigma_k\Big[4\mu_0^2 + \mu_c^2 t_k^2\|d_k\|^2\Big]t_k \tag{3.10}$$

$$\text{and } f_{\lambda_k,\xi_k}\big(\bar{u}_{k+1}\big) < f_{\lambda_k,\xi_k}(u_k) + \nu t_k \mathit{\Delta} f_{\lambda_k,\xi_k}(u_k; -d_k) + \mathcal{O}\big(\frac{1}{(k+1)^{1+\varepsilon}}\big) \tag{3.11}$$

where $\bar{u}_{k+1} := P_C(u_k - t_k d_k)$ and f_{λ_k,ξ_k} is defined by (3.4)

▷ **Step 4**: Update

$$u_{k+1} = P_C(u_k - t_k d_k) \tag{3.12}$$

$$\lambda_{k+1} = \lambda_k + \sigma_k\Big(c(u_{k+1}) - b - P_S\big(c(u_{k+1}) - b + \rho_k^{-1}\lambda_k\big)\Big) \tag{3.13}$$

end for

3.2 Main Theorem

Before presenting our main convergence results, we summarize the general strategy followed. The main principle is to derive the descent property of the Lagrange function values $\big(\mathcal{L}_{\rho_k}(u_k,\lambda_k)\big)_{k\in\mathbb{N}}$ with respect to $(t_k\|d_k\|^2)_{k\in\mathbb{N}}$. To reach this goal, we consider the following steps:

(i) We first need to show that Step 1 and Step 3 are well defined. They are presented in Lemma 9 as well as in Lemma 10. In particular, we obtain the descent property of the stochastic function f_{ξ_k,λ_k} as in (3.11).

(ii) We further estimate $\varDelta f_{\lambda_k,\xi_k}(u_k;-d_k) \leq -\beta_k\|d_k\|^2$ as proved in Lemma 8. Combining this result to (3.11), we obtain the descent of f_{ξ_k,λ_k} with respect to d_k as written in (3.65).

(iii) Based on $\mathbf{E}_{\xi_k}[\mathcal{L}_{\rho_k,\xi_k}(u_k,\lambda_{k,\xi_k})] = \mathcal{L}_{\rho_k}(u_k,\lambda_k)$, we use the results obtained in Lemma 7 where we show that the Lagrange function satisfies a sufficient-decrease condition and Lemma 11 to derive the descent property of $\big(\mathcal{L}_{\rho_k}(u_k,\lambda_k)\big)_{k\in\mathbb{N}}$ from (3.65) as in (3.81).

(iv) From (3.81), it is easy to find the convergence property of the proposed method as in Theorem 3.

We first prove several auxiliary results, which will be used in the proof of the main Theorem part of this section.

Lemma 7 *Let $k\in\mathbb{N}$. Then,*

$$\mathcal{L}_{\rho_{k+1}}(u_{k+1},\lambda_{k+1}) \leq \mathcal{L}_{\rho_k}(u_{k+1},\lambda_k) + \frac{(\sigma_k+0.5(\rho_{k+1}-\rho_k))}{\sigma_k}|||\lambda_{k+1}-\lambda_k|||^2. \tag{3.14}$$

Proof. In view of Lemma 3, for $e_{k+1} = P_S(c(u_{k+1})-b+\rho_{k+1}^{-1}\lambda_{k+1})$, we have

$$\mathcal{L}_{\rho_{k+1}}(u_{k+1},\lambda_{k+1}) = h(u_{k+1}) + \langle c(u_{k+1})-b-e_{k+1} \mid\mid \lambda_{k+1}\rangle + \frac{\rho_{k+1}}{2}|||c(u_{k+1})-b-e_{k+1}|||^2. \tag{3.15}$$

By defining $p_{k+1} \coloneqq P_S(c(u_{k+1})-b+\rho_k^{-1}\lambda_k)$, we can express the third term in the right hand side of (3.15) as

$$\begin{aligned}\frac{\rho_{k+1}}{2}|||c(u_{k+1})-b-e_{k+1}|||^2 &= \frac{\rho_{k+1}}{2}|||c(u_{k+1})-b-p_{k+1}|||^2 + \frac{\rho_{k+1}}{2}|||e_{k+1}-p_{k+1}|||^2\\ &\quad + \rho_{k+1}\langle c(u_{k+1})-b-p_{k+1} \mid\mid p_{k+1}-e_{k+1}\rangle.\end{aligned} \tag{3.16}$$

The second term in the right hand side of (3.15) can be written as

$$\begin{aligned}\langle c(u_{k+1})-b-e_{k+1} \mid\mid \lambda_{k+1}\rangle &= \langle c(u_{k+1})-b-p_{k+1} \mid\mid \lambda_k\rangle + \langle c(u_{k+1})-b-e_{k+1} \mid\mid \lambda_{k+1}\rangle\\ &\quad - \langle c(u_{k+1})-b-p_{k+1} \mid\mid \lambda_k\rangle.\end{aligned} \tag{3.17}$$

Using the definition of p_{k+1}, the update rule of the dual variables can be written as

$$\lambda_{k+1} = \lambda_k + \sigma_k(c(u_{k+1}-b-p_{k+1})). \tag{3.18}$$

Thus, it follows that

$$\begin{aligned}\langle c(u_{k+1})-b-e_{k+1} \mid\mid \lambda_{k+1}\rangle &= \langle c(u_{k+1})-b-p_{k+1} \mid\mid \lambda_k\rangle + \langle c(u_{k+1})-b-e_{k+1} \mid\mid \lambda_{k+1}\rangle\\ &\quad - \langle c(u_{k+1})-b-p_{k+1} \mid\mid \lambda_{k+1}\rangle + \sigma_k|||c(u_{k+1})-b-p_{k+1}|||^2\\ &= \langle c(u_{k+1})-b-p_{k+1} \mid\mid \lambda_k\rangle + \langle p_{k+1}-e_{k+1} \mid\mid \lambda_{k+1}\rangle\\ &\quad + \sigma_k|||c(u_{k+1})-b-p_{k+1}|||^2.\end{aligned} \tag{3.19}$$

Therefore, (3.15) becomes

$$\mathcal{L}_{\rho_{k+1}}(u_{k+1},\lambda_{k+1}) = \mathcal{L}_{\rho_k}(u_{k+1},\lambda_k) + (\sigma_k + \frac{\rho_{k+1}-\rho_k}{2})|||c(u_{k+1})-b-p_{k+1}|||^2 + o_k, \tag{3.20}$$

where we set

$$\begin{aligned}
o_k &:= \langle p_{k+1} - e_{k+1} \mid\mid \lambda_{k+1}\rangle + \rho_{k+1}\langle c(u_{k+1}) - b - p_{k+1} \mid\mid p_{k+1} - e_{k+1}\rangle + \frac{\rho_{k+1}}{2}|||e_{k+1} - p_{k+1}|||^2 \\
&= \langle p_{k+1} - e_{k+1} \mid\mid \lambda_{k+1}\rangle + \rho_{k+1}\langle c(u_{k+1}) - b - e_{k+1} \mid\mid p_{k+1} - e_{k+1}\rangle - \frac{\rho_{k+1}}{2}|||p_{k+1} - e_{k+1}|||^2 \\
&\leq \rho_{k+1}\Big(\big\langle p_{k+1} - e_{k+1} \mid\mid \rho_{k+1}^{-1}\lambda_{k+1}\big\rangle + \big\langle c(u_{k+1}) - b - e_{k+1} \mid\mid p_{k+1} - e_{k+1}\big\rangle \Big) \\
&= \rho_{k+1}\Big(\big\langle p_{k+1} - e_{k+1} \mid\mid \rho_{k+1}^{-1}\lambda_{k+1} + c(u_{k+1}) - b - e_{k+1}\big\rangle \Big) \\
&\leq 0, \qquad (3.21)
\end{aligned}$$

where the last inequality follows from Lemma 5. Therefore, using the expression (3.18), the conclusion follows from (3.20). □

Lemma 8 *Let $k \in \mathbb{N}$ and let β_k, d_k and f_{λ_k,ξ_k} be defined, respectively, by Step 1, Step 2 and Step 3 of Algorithm 1. Then,*

$$\Delta f_{\lambda_k,\xi_k}(u_k; -d_k) \leq -\beta_k \|d_k\|^2. \qquad (3.22)$$

Proof. At each iteration $k \in \mathbb{N}$, define

$$\begin{cases}
c_{\xi_k} = (c_{i_p})_{1\leq p\leq \mathtt{m}_k} \\
v_{\xi_k} = (v_{k,i_p})_{1\leq p\leq \mathtt{m}_k} \\
b_{\xi_k} = (b_{i_p})_{1\leq p\leq \mathtt{m}_k} \\
\lambda_{k,\xi_k} = (\lambda_{k,i_p})_{1\leq p\leq \mathtt{m}_k} \\
S_{\xi_k} = (S_{i_p})_{1\leq p\leq \mathtt{m}_k} \\
h_{\xi_k}(\cdot) = \dfrac{1}{\mathtt{m}_k}\displaystyle\sum_{p=1}^{\mathtt{m}_k} h_{i_p}(\cdot) \\
\psi_{\xi_k}(\cdot) = \dfrac{1}{\mathtt{m}_k}\displaystyle\sum_{p=1}^{\mathtt{m}_k}\left(\frac{\rho_k}{2} d^2_{S_{i_p}}((\cdot) - b_{i_p} + \rho_k^{-1}\lambda_{k,i_p}) - \frac{1}{2\rho_k}\|\lambda_{k,i_p}\|^2\right) \\
\mathcal{L}_{\rho_k,\xi_k}(\cdot,\lambda_k) = h_{\xi_k}(\cdot) + (\psi_{\xi_k}\circ c_{\xi_k})(\cdot).
\end{cases}$$

Then, we obtain

$$f_{\lambda_k,\xi_k}(\cdot) = \mathcal{L}_{\rho_k,\xi_k}(\cdot,\lambda_k). \qquad (3.23)$$

The direction $d_k \in \mathbb{R}^K$ defined by (3.8) satisfies following (2.24),

$$\Delta f_{\lambda_k,\xi_k}(u_k; d_k) = \psi_{\xi_k}(c_{\xi_k}(u_k) + J_{c_{\xi_k}}(u_k)d_k) + \langle \nabla h_{\xi_k}(u_k) \mid d_k\rangle - \psi_{\xi_k}(c_{\xi_k}(u_k)). \qquad (3.24)$$

For the sake of clarity and conciseness, let us define the following

$$\begin{cases}
e_{\xi_k} := P_{S_{\xi_k}}(c_{\xi_k}(u_k) - b_{\xi_k} + \rho_k^{-1}\lambda_{\xi_k}), \\
z_{\xi_k} := P_{S_{\xi_k}}(c_{\xi_k}(u_k) + J_{c_{\xi_k}}(u_k)d_k - b_{\xi_k} + \rho_k^{-1}\lambda_k), \\
s_{\xi_k} := J_{c_{\xi_k}}(u_k)d_k - z_{\xi_k} + e_{\xi_k}.
\end{cases} \qquad (3.25)$$

We also use the following scalar product

$$\langle\langle\cdot \mid \cdot\rangle\rangle \colon (w_{\xi_k}, v_{\xi_k}) \mapsto \sum_{p=1}^{\mathtt{m}_k} \big\langle w_{i_p} \mid v_{i_p}\big\rangle$$

$$\text{with vector norm } |||\cdot|||\colon v_{\xi_k} \mapsto \sqrt{\langle\langle v_{\xi_k} \mid v_{\xi_k}\rangle\rangle}.$$

Using these notations, by Lemma 3, we have

$$\begin{aligned}
&\mathtt{m}_k\psi_{\xi_k}(c_{\xi_k}(u_k)+J_{c_{\xi_k}}(u_k)d_k)\\
&\quad=\langle\langle c_{\xi_k}(u_k)+J_{c_{\xi_k}}(u_k)d_k-b_{\xi_k}-z_{\xi_k}\mid\lambda_{\xi_k}\rangle\rangle+\frac{\rho_k}{2}|||c_{\xi_k}(u_k)+J_{c_{\xi_k}}(u_k)d_k-b_{\xi_k}-z_{\xi_k}|||^2\\
&\quad=\langle\langle c_{\xi_k}(u_k)-b_{\xi_k}-e_{\xi_k}+s_{\xi_k}\mid\lambda_{\xi_k}\rangle\rangle+\frac{\rho_k}{2}|||c_{\xi_k}(u_k)-b_{\xi_k}-e_{\xi_k}+s_{\xi_k}|||^2\\
&\quad=\langle\langle c_{\xi_k}(u_k)-b_{\xi_k}-e_{\xi_k}\mid\lambda_{\xi_k}\rangle\rangle+\langle\langle s_{\xi_k}\mid\lambda_{\xi_k}\rangle\rangle+\frac{\rho_k}{2}|||c_{\xi_k}(u_k)-b_{\xi_k}-e_{\xi_k}|||^2\\
&\qquad+\rho_k\langle\langle c_{\xi_k}(u_k)-b_{\xi_k}-e_{\xi_{\xi_k}}\mid s_{\xi_k}\rangle\rangle+\frac{\rho_k}{2}|||s_{\xi_k}|||^2, \qquad (3.26)
\end{aligned}$$

which implies, using the definition of e_{ξ_k}, that

$$\begin{aligned}
&\mathtt{m}_k\big(\psi_{\xi_k}(c_{\xi_k}(u_k)+J_{c_{\xi_k}}(u_k)d_k)-\psi_{\xi_k}(c_{\xi_k}(u_k))\big)\\
&\quad=\langle\langle s_{\xi_k}\mid\lambda_{\xi_k}\rangle\rangle+\rho_k\langle\langle c_{\xi_k}(u_k)-b_{\xi_k}-e_{\xi_k}\mid s_{\xi_k}\rangle\rangle+\frac{\rho_k}{2}|||s_{\xi_k}|||^2\\
&\quad=\langle\langle \lambda_{\xi_k}+\rho_k\big(c_{\xi_k}(u_k)-b_{\xi_k}-e_{\xi_k}\big)\mid s_{\xi_k}\rangle\rangle+\frac{\rho_k}{2}|||s_{\xi_k}|||^2\\
&\quad=\langle\langle v_{\xi_k}\mid s_{\xi_k}\rangle\rangle+\frac{\rho_k}{2}|||s_{\xi_k}|||^2. \qquad (3.27)
\end{aligned}$$

Note that

$$|||s_{\xi_k}|||^2=|||J_{c_{\xi_k}}(u_k)d_k|||^2+2\langle\langle J_{c_{\xi_k}}(u_k)d_k\mid e_{\xi_k}-z_{\xi_k}\rangle\rangle+|||e_{\xi_k}-z_{\xi_k}|||^2. \qquad (3.28)$$

Therefore,

$$\begin{aligned}
&\mathtt{m}_k\big(\psi_{\xi_k}(c_{\xi_k}(u_k)+J_{c_{\xi_k}}(u_k)d_k)-\psi_{\xi_k}(c_{\xi_k}(u_k))\big)\\
&\le\langle\langle v_{\xi_k}\mid s_{\xi_k}\rangle\rangle+\frac{\rho_k}{2}|||J_{c_{\xi_k}}(u_k)d_k|||^2+\rho_k\langle\langle J_{c_{\xi_k}}(u_k)d_k\mid e_{\xi_k}-z_{\xi_k}\rangle\rangle+\frac{\rho_k}{2}|||e_{\xi_k}-z_{\xi_k}|||^2\\
&=\langle\langle J_{c_{\xi_k}}(u_k)^\intercal v_{\xi_k}\mid d_k\rangle\rangle+\frac{\rho_k}{2}|||J_{\xi_k}(u_k)d_k|||^2+\langle\langle v_{\xi_k}+\rho_kJ_{\xi_k}(u_k)d_k\mid e_{\xi_k}-z_{\xi_k}\rangle\rangle+\frac{\rho_k}{2}|||e_{\xi_k}-z_{\xi_k}|||^2. \qquad (3.29)
\end{aligned}$$

The weighted inner product $\langle\langle v_{\xi_k}+\rho_kJ_{c_{\xi_k}}(u_k)d_k\mid e_{\xi_k}-z_{\xi_k}\rangle\rangle$ satisfies

$$\begin{aligned}
&\langle\langle v_{\xi_k}+\rho_kJ_{c_{\xi_k}}(u_k)d_k\mid e_{\xi_k}-z_{\xi_k}\rangle\rangle\\
&\quad=\rho_k\langle\langle c_{\xi_k}(u_k)-b_{\xi_k}+J_{c_{\xi_k}}(u_k)d_k+\rho_k^{-1}\lambda_{\xi_k}-e_{\xi_k}\mid e_{\xi_k}-z_{\xi_k}\rangle\rangle\\
&\quad=\rho_k\langle\langle c_{\xi_k}(u_k)-b_{\xi_k}+J_{c_{\xi_k}}(u_k)d_k+\rho_k^{-1}\lambda_{\xi_k}-z_{\xi_k}\mid e_{\xi_k}-z_{\xi_k}\rangle\rangle-\rho_k|||z_{\xi_k}-e_{\xi_k}|||^2\\
&\quad\le-\rho_k|||z_{\xi_k}-e_{\xi_k}|||^2, \qquad (3.30)
\end{aligned}$$

where the last inequality follows from Lemma 5. In turn,

$$\psi_{\xi_k}(c_{\xi_k}(u_k)+J_{c_{\xi_k}}(u_k)d_k)-\psi_{\xi_k}(c_{\xi_k}(u_k))\le\frac{1}{\mathtt{m}_k}\Big(\langle\langle J_{c_{\xi_k}}(u_k)^\intercal v_{\xi_k}\mid d_k\rangle\rangle+\frac{\rho_k}{2}|||J_{c_{\xi_k}}(u_k)d_k|||^2\Big). \qquad (3.31)$$

Adding $\langle\langle\nabla h_{\xi_k}(u_k)\mid d_k\rangle\rangle$ to both sides of (3.31) and using the definition of the descent direction d_k, we obtain

$$\Delta f_{\lambda_k,\xi_k}(u_k;d_k)\le\langle\langle\nabla h_{\xi_k}(u_k)+\frac{1}{\mathtt{m}_k}J_{c_{\xi_k}}(u_k)^\intercal v_{\xi_k}\mid d_k\rangle\rangle+\frac{\rho_k}{2\mathtt{m}_k}|||J_{c_{\xi_k}}(u_k)d_k|||^2 \qquad (3.32)$$

Observe that the stochastic direction d_k is the gradient of f_{λ_k,ξ_k} at the current point u_k, i.e $d_k=\nabla f_{\lambda_k,\xi_k}(u_k)$. Hence,

$$\begin{aligned}
\Delta f_{\lambda_k,\xi_k}(u_k;-d_k)&\le-\|d_k\|^2+\frac{\rho_k}{2\mathtt{m}_k}|||J_{c_{\xi_k}}(u_k)d_k|||^2\\
&\le-\big(1-\frac{\rho_k}{2\mathtt{m}_k}\|J_{c_{\xi_k}}(u_k)\|^2\big)\|d_k\|^2\\
&\le-\beta_k\|d_k\|^2\\
&<0, \qquad (3.33)
\end{aligned}$$

where the second inequality follows from (3.5). □

Lemma 9 *The sequence $(\lambda_k)_{k\in\mathbb{N}}$ belongs to the polar cone $S^{\ominus}$ when $\lambda_0 \in S^{\ominus}$ and Step 1 of Algorithm 1 is well defined.*

Proof. Suppose that $\lambda_k \in S^{\ominus}$. Let $u \in C$ and $\rho > 0$ and set $a = c(u) - b$. Then, it follows from [5, Theorem 6.30(i)] and [5, Proposition 29(ii)] that

$$\begin{aligned}\rho\Big(a - P_S\big(a + \rho^{-1}\lambda_k\big)\Big) &= \rho\Big(P_{S^{\ominus}}\big(a+\rho^{-1}\lambda_k\big) - \rho^{-1}\lambda_k\Big)\\ &= \rho P_{S^{\ominus}}\big((\rho a + \lambda_k)/\rho\big) - \lambda_k\\ &= \rho P_{S^{\ominus}/\rho}\big((\rho a + \lambda_k)/\rho\big) - \lambda_k\\ &= P_{S^{\ominus}}\big(\rho a + \lambda_k\big) - \lambda_k.\end{aligned} \tag{3.34}$$

The latter equality implies that for $u = u_k$, the following identity is verified

$$\lambda_k = P_{S^{\ominus}}\big(\rho(c(u_k) - b) + \lambda_k\big) - \rho\big(c(u_k) - b - P_S(c(u_k) - b + \rho^{-1}\lambda_k)\big). \tag{3.35}$$

Therefore,

$$\rho|||c(u_k) - b - P_S(c(u_k) - b + \rho^{-1}\lambda_k)||| = |||\lambda_k - P_{S^{\ominus}}\big(\rho(c(u_k)-b) + \lambda_k\big)||| \le \rho|||c(u_k) - b|||. \tag{3.36}$$

Hence, by choosing

$$\rho_k = \rho \le \min\left\{\Big(\min_{1\le i\le k}\|u_i - u_{i-1}\|\Big)^2 |||c(u_k)-b|||^{-2}, \rho_{k-1} + \varepsilon\sigma_{k-1}, 2(1-\varepsilon)\|J_c(u_k)\|^{-2}\right\}, \tag{3.37}$$

we get

$$\begin{cases} 1 - \dfrac{\rho_k}{2}\|J_c(u_k)\|^2 > \varepsilon \\ \sqrt{\rho_k}|||c(u_k) - b - P_S\big(c(u_k) - b + \rho_k^{-1}\lambda_k\big)||| \le \min_{1\le i\le k}\|u_i - u_{i-1}\| \\ \rho_k < \rho_{k-1} + \varepsilon\sigma_{k-1}. \end{cases} \tag{3.38}$$

Consequently, Step 1 is well defined when $\lambda_k \in S^{\ominus}$. We next prove $\lambda_{k+1} \in S^{\ominus}$. Indeed, we have

$$\lambda_k = P_{S^{\ominus}}\big(\rho_k(c(u_{k+1}) - b) + \lambda_k\big) - \rho_k\big(c(u_{k+1}) - b - P_S(c(u_{k+1}) - b + \rho_k^{-1}\lambda_k)\big). \tag{3.39}$$

Thus

$$\lambda_{k+1} = (1 - \sigma_k/\rho_k)\lambda_k + (\sigma_k/\rho_k)P_{S^{\ominus}}(c(u_{k+1}) - b + \rho_k^{-1}\lambda_k) \in S^{\ominus}, \tag{3.40}$$

where the last inclusion follows from $\lambda_k \in S^{\ominus}$ and $\sigma_k \le \rho_k$. Therefore, the lemma is proved by induction. □

Lemma 10 *The line search (Step 3 of Algorithm 1) terminates after a finite number of steps, i.e., there exists $t_k > 0$ such that*

$$f_{\lambda_k,\xi_k}\big(\bar{u}_{k+1}\big) < f_{\lambda_k,\xi_k}(u_k) + \nu t_k \Delta f_{\lambda_k,\xi_k}(u_k; -d_k) + \mathcal{O}\Big(\frac{1}{(k+1)^{1+\varepsilon}}\Big). \tag{3.41}$$

Proof. In view of [16, Lemma 5.1], for a fixed $\nu \in\,]0,1[$, there exists a finite upper limit $\overline{t_k} > 0$ of the primal stepsize interval such that for all primal stepsizes included in the open interval $]0, \overline{t_k}[$, the function φ_k verifies the following inequality

$$(\forall t \in \big]0, \overline{t_k}\big[)\ f_{\lambda_k,\xi_k}(u_k - td_k) \le f_{\lambda_k,\xi_k}(u_k) + t\nu\Delta f_{\lambda_k,\xi_k}(u_k; -d_k), \tag{3.42}$$

Since $\lim_{t\downarrow 0} P_C(u_k - td_k) = u_k$ and f_{λ_k,ξ_k} is continuous, we obtain

$$\lim_{t\downarrow 0}|f_{\lambda_k,\xi_k}(P_C(u_k - td_k)) - f_{\lambda_k,\xi_k}(u_k - td_k)| = 0. \tag{3.43}$$

Therefore, there exists

$$t_k \in \left]0, \bar{t}_k\right[\tag{3.44}$$

such that

$$f_{\lambda_k,\xi_k}(P_C(u_k - td_k)) \leq f_{\lambda_k,\xi_k}(u_k - td_k) + \mathcal{O}(1/(k+1)^{1+\varepsilon}), \tag{3.45}$$

which implies that the condition (3.11) is well-defined. □

Lemma 11 *Let d_k be defined by (3.8). Set*

$$\varsigma_{1,k} := (1+\varepsilon)4\sigma_k\left[4\mu_0^2 + \mu_c^2 t_k^2\|d_k\|^2\right]t_k. \tag{3.46}$$

Then

$$\mathbf{E}_{\xi_k}\left[\frac{1+\varepsilon}{\sigma_k}|||\lambda_{k+1} - \lambda_k|||^2|\mathcal{E}_k\right] \leq \mathbf{E}_{\xi_k}\left[\varsigma_{1,k}t_k\|d_k\|^2|\mathcal{E}_k\right] + 2(1+\varepsilon)t_{k-1}^2\|d_{k-1}\|^2. \tag{3.47}$$

Proof. Define

$$(\forall k \in \mathbb{N})\ e_k := P_S(c(u_k) - b + \rho_k^{-1}\lambda_k) \text{ and } q_k := c(u_k) - b - e_k. \tag{3.48}$$

and

$$(\forall k \in \mathbb{N})\ p_{k+1} := P_S(c(u_{k+1}) - b + \rho_k^{-1}\lambda_k) \text{ and } \bar{q}_{k+1} := c(u_{k+1}) - b - p_{k+1}. \tag{3.49}$$

Then, by the update rules

$$\begin{cases} v_k = \lambda_k + \rho_k q_k \\ \lambda_{k+1} = \lambda_k + \sigma_k \bar{q}_{k+1}, \end{cases} \tag{3.50}$$

we obtain the following inequality

$$\begin{aligned} |||\lambda_{k+1} - \lambda_k|||^2 &= \sigma_k^2\|\bar{q}_{k+1}\|^2 \\ &\leq 2\sigma_k^2(|||q_k|||^2 + |||q_k - \bar{q}_{k+1}|||^2) \\ &= 2\sigma_k^2(|||q_k|||^2 + |||c(u_{k+1}) - c(u_k) - p_{k+1} + e_k|||^2) \\ &\leq 2\sigma_k^2|||q_k|||^2 + 4\sigma_k^2|||c(u_{k+1}) - c(u_k)|||^2 + 4\sigma_k^2|||p_{k+1} - e_k|||^2. \end{aligned} \tag{3.51}$$

Using (3.49), the third term in the RHS of (3.51) becomes

$$\begin{aligned} |||p_{k+1} - e_k|||^2 &= |||P_S(c(u_{k+1}) - b + \rho_k^{-1}\lambda_k) - P_S(c(u_k) - b + \rho_k^{-1}\lambda_k)|||^2 \\ &\leq |||c(u_{k+1}) - c(u_k)|||^2. \end{aligned} \tag{3.52}$$

Therefore, inequality (3.51) can be written as

$$|||\lambda_{k+1} - \lambda_k|||^2 \leq 2\sigma_k^2|||q_k|||^2 + 8\sigma_k^2|||c(u_{k+1}) - c(u_k)|||^2. \tag{3.53}$$

By the Step 1 of Algorithm 1 and $\sigma_k \leq \rho_k$, the first term in the RHS of (3.53) verifies the inequality

$$2\sigma_k^2|||q_k|||^2 \leq 2\sigma_k|||u_k - u_{k-1}|||^2. \tag{3.54}$$

Since the Jacobian $J_c(u_k)$ of c is μ_c-Lipschitz continuous on the subset C of $\mathbb{R}^m$, the second term in the RHS of (3.53) satisfies the inequality

$$\begin{aligned} |||c(u_{k+1}) - c(u_k)|||^2 &\leq (|||J_c(u_k)(u_{k+1} - u_k)||| + (\mu_c/2)\|u_{k+1} - u_k\|^2)^2 \\ &\leq 2|||J_c(u_k)(u_{k+1} - u_k)|||^2 + (\mu_c^2/2)\|u_{k+1} - u_k\|^4 \\ &\leq 2\|J_c(u_k)\|^2\|(u_{k+1} - u_k)\|^2 + (\mu_c^2/2)\|u_{k+1} - u_k\|^4. \end{aligned} \tag{3.55}$$

By our assumption, $\sup_{k\in\mathbb{N}}\|J_c(u_k)\| \leq \mu_0$ is finite. It follows that

$$8\sigma_k^2|||c(u_{k+1}) - c(u_k)|||^2 \leq 16\mu_0^2\sigma_k^2\|(u_{k+1} - u_k)\|^2 + 4\mu_c^2\sigma_k^2\|u_{k+1} - u_k\|^4. \tag{3.56}$$

Summing the RHS of (3.54) and (3.56), we deduce from (3.53) that

$$|||\lambda_{k+1}-\lambda_k|||^2 \le 16\mu_0^2\sigma_k^2\|u_{k+1}-u_k\|^2 + 4\mu_c^2\sigma_k^2\|u_{k+1}-u_k\|^4 + 2\sigma_k\|u_k-u_{k-1}\|^2. \tag{3.57}$$

Since $\|u_{k+1}-u_k\| \le t_k\|d_k\|$, we further bound (3.57) as

$$\frac{1+\varepsilon}{\sigma_k}|||\lambda_{k+1}-\lambda_k|||^2 \le (1+\varepsilon)\Big[16\mu_0^2\sigma_k t_k^2\|d_k\|^2 + 4\mu_c^2\sigma_k t_k^4\|d_k\|^4 + 2t_{k-1}^2\|d_{k-1}\|^2\Big]. \tag{3.58}$$

$$= (1+\varepsilon)\Big[16\mu_0^2\sigma_k t_k + 4\mu_c^2\sigma_k t_k^3\|d_k\|^2\Big]t_k\|d_k\|^2 + 2(1+\varepsilon)t_{k-1}^2\|d_{k-1}\|^2$$

$$= \varsigma_{1,k}t_k\|d_k\|^2 + 2(1+\varepsilon)t_{k-1}^2\|d_{k-1}\|^2, \tag{3.59}$$

which proves (3.47) by taking the expectation with respect to ξ_k both sides of (3.59) and using $\mathbf{E}_{\xi_k}[t_{k-1}^2\|d_{k-1}\|^2] = t_{k-1}^2\|d_{k-1}\|^2$. □

Theorem 3 *Let $((u_k,\lambda_k))_{k\in\mathbb{N}}$ be the primal-dual sequence generated by Algorithm 1. Suppose that Assumptions 1 & 2 are satisfied and $(\mathcal{L}_{\rho_k}(u_k,\lambda_k))_{k\in\mathbb{N}}$ is bounded below. Further assume that, the size m_k of the mini-batch selected at each iteration k, verifies*

$$m_k \ge \mathcal{O}(\overline{\sigma}_k^2 t_{k,max}(k+1)^{1+\varepsilon}) \tag{3.60}$$

together with $(1+t_k^2\ell_{\xi_k}) \le t_{k,max} < +\infty$ a.s., and $(1+t_k^2\ell_k) \le t_{k,max} < +\infty$ a.s., where $t_{k,max}$ is independent of ξ_k.

Then, the following hold.

(i) *The sequence $(\mathbf{E}_{\xi_k}\Big[\|\frac{u_{k+1}-u_k}{\sqrt{t_k}}\|^2|\mathcal{E}_k\Big])_{k\in\mathbb{N}}$ is summmable, i.e.,*

$$\sum_{k\in\mathbb{N}}\mathbf{E}_{\xi_k}\Big[\|\frac{u_{k+1}-u_k}{\sqrt{t_k}}\|^2|\mathcal{E}_k\Big] < +\infty. \tag{3.61}$$

(ii) *The sequence $(\mathbf{E}_{\xi_k}\big[\sigma_k|||c(u_{k+1})-b-P_S(c(u_{k+1}-b+\rho_k^{-1}\lambda_k)|||^2|\mathcal{E}_k\big])_{k\in\mathbb{N}}$ is summable, i.e.,*

$$\sum_{k\in\mathbb{N}}\mathbf{E}_{\xi_k}\big[\sigma_k|||c(u_{k+1})-b-P_S(c(u_{k+1}-b+\rho_k^{-1}\lambda_k)|||^2|\mathcal{E}_k\big] < +\infty. \tag{3.62}$$

(iii) *Define $u_{k+1}^e = P_C(u_k - t_k\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k))$. Then, the sequence $(\mathbf{E}\Big[\|\frac{u_{k+1}^e-u_k}{\sqrt{t_k}}\|^2|\mathcal{E}_k\Big])_{k\in\mathbb{N}}$ is summmable, i.e.,*

$$\sum_{k\in\mathbb{N}}\mathbf{E}_{\xi_k}\Big[\|\frac{u_{k+1}^e-u_k}{\sqrt{t_k}}\|^2|\mathcal{E}_k\Big] < +\infty, \text{ a.s.} \tag{3.63}$$

(iv) *Choosing σ_k such that $\sup_{k\in\mathbb{N}}\sigma_k \le \sigma_\infty < +\infty$ where σ_∞ is independent of ξ_k. Then, the sequence $(\mathbf{E}_{\xi_k}\big[\sigma_k|||c(u_{k+1}^e)-b-P_S(c(u_{k+1}^e)-b+\rho_k^{-1}\lambda_k)|||^2|\mathcal{E}_k\big])_{k\in\mathbb{N}}$ is summable, i.e.,*

$$\sum_{k\in\mathbb{N}}\mathbf{E}_{\xi_k}\big[\sigma_k|||c(u_{k+1}^e)-b-P_S(c(u_{k+1}^e)-b+\rho_k^{-1}\lambda_k)|||^2|\mathcal{E}_k\big] < +\infty, \text{ a.s.} \tag{3.64}$$

Proof. In this proof, we denote by $\mathbf{E}_{\xi_k}[X] = \mathbf{E}_{\xi_k}[X|\mathcal{E}_k]$ the conditional expectation of X with respect to $\mathcal{E}_k$. Using Lemma 8 and Lemma 10, we obtain

$$f_{\lambda_k,\xi_k}(u_{k+1}) < \mathcal{L}_{\rho_k,\xi_k}(u_k,\lambda_{k,\xi_k}) - t_k\beta_k\nu\|d_k\|^2 + \mathcal{O}\Big(\frac{1}{(k+1)^{1+\varepsilon}}\Big). \tag{3.65}$$

Note that $u_{k+1}^e = P_C(u_k - t_k\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k))$. Then, it follows from the nonexpansiveness of the projection operator P_C that

$$\|u_{k+1}^e-u_{k+1}\|^2 = \|P_C(u_k-t_k\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)) - P_C(u_k-t_kd_k)\|^2 \le t_k^2\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2. \tag{3.66}$$

Let ℓ_{ξ_k} be the Lipschitz constant of $\nabla f_{\lambda_k,\xi_k}$. Then, it follows from the Descent Lemma [5, Lemma 2.64] that

$$\begin{aligned}
f_{\lambda_k,\xi_k}(u^e_{k+1}) - f_{\lambda_k,\xi_k}(u_{k+1}) &\leq \big\langle \nabla f_{\lambda_k,\xi_k}(u_{k+1}) \mid u^e_{k+1} - u_{k+1}\big\rangle + \frac{\ell_{\xi_k}}{2}\|u^e_{k+1} - u_{k+1}\|^2 \\
&\leq \|\nabla f_{\lambda_k,\xi_k}(u_{k+1})\|\|u^e_{k+1} - u_{k+1}\| + \frac{\ell_{\xi_k}}{2}\|u^e_{k+1} - u_{k+1}\|^2 \\
&\leq \Big(\|\nabla f_{\lambda_k,\xi_k}(u_k)\| + \|\nabla f_{\lambda_k,\xi_k}(u_{k+1}) - \nabla f_{\lambda_k,\xi_k}(u_k)\|\Big)\|u^e_{k+1} - u_{k+1}\| \\
&\quad + \frac{\ell_{\xi_k}}{2}\|u^e_{k+1} - u_{k+1}\|^2. \qquad (3.67)
\end{aligned}$$

Since $\|\nabla f_{\lambda_k,\xi_k}(u_{k+1}) - \nabla f_{\lambda_k,\xi_k}(u_k)\| \leq \ell_{\xi_k}\|u_{k+1} - u_k\|$, the RHS of (3.67) verifies

$$\begin{aligned}
&\Big(\|\nabla f_{\lambda_k,\xi_k}(u_k)\| + \|\nabla f_{\lambda_k,\xi_k}(u_{k+1}) - \nabla f_{\lambda_k,\xi_k}(u_k)\|\Big)\|u^e_{k+1} - u_{k+1}\| + \frac{\ell_{\xi_k}}{2}\|u^e_{k+1} - u_{k+1}\|^2 \\
&\qquad \leq (1 + t_k\ell_{\xi_k})\|d_k\|\|u^e_{k+1} - u_{k+1}\| + \frac{\ell_{\xi_k}}{2}\|u^e_{k+1} - u_{k+1}\|^2 \\
&\qquad \leq t_k(1 + t_k\ell_{\xi_k})\|d_k\|\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\| + \frac{\ell_{\xi_k}}{2}t_k^2\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2 \\
&\qquad \leq \frac{1}{2}t_k^2(1 + t_k\ell_{\xi_k})^2\|d_k\|^2 + \frac{1}{2}(1 + t_k^2\ell_{\xi_k})\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2. \qquad (3.68)
\end{aligned}$$

Combining (3.68) with (3.65), we deduce

$$\begin{aligned}
f_{\lambda_k,\xi_k}(u^e_{k+1}) &< \mathcal{L}_{\rho_k,\xi_k}(u_k,\lambda_{k,\xi_k}) - t_k\big(\nu\beta_k - \frac{1}{2}t_k(1 + t_k\ell_{\xi_k})^2\big)\|d_k\|^2 \\
&\quad + \frac{1}{2}(1 + t_k^2\ell_{\xi_k})\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2 + \mathcal{O}\Big(\frac{1}{(k+1)^{1+\varepsilon}}\Big). \qquad (3.69)
\end{aligned}$$

Let ℓ_k be the Lipschitz constant of $\nabla\mathcal{L}_{\rho_k}(\cdot,\lambda_k)$. Then, it follows from the Descent Lemma [5, Lemma 2.64] that

$$\mathcal{L}_{\rho_k}(u_{k+1},\lambda_k) - \mathcal{L}_{\rho_k}(u^e_{k+1},\lambda_k) \leq \big\langle \nabla\mathcal{L}_{\rho_k}(u^e_{k+1},\lambda_k) \mid u_{k+1} - u^e_{k+1}\big\rangle + \frac{\ell_k}{2}\|u^e_{k+1} - u_{k+1}\|^2 \qquad (3.70)$$

The RHS of (3.70) verifies

$$\begin{aligned}
&\big\langle \nabla\mathcal{L}_{\rho_k}(u^e_{k+1},\lambda_k) \mid u_{k+1} - u^e_{k+1}\big\rangle + \frac{\ell_k}{2}\|u^e_{k+1} - u_{k+1}\|^2 \\
&\quad \leq \|\nabla\mathcal{L}_{\rho_k}(u^e_{k+1},\lambda_k)\|\|u^e_{k+1} - u_{k+1}\| + \frac{\ell_k}{2}\|u^e_{k+1} - u_{k+1}\|^2 \\
&\quad \leq \Big(\|\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\| + \|\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k) - \nabla\mathcal{L}_{\rho_k}(u^e_{k+1},\lambda_k)\|\Big)\|u^e_{k+1} - u_{k+1}\| + \frac{\ell_k}{2}\|u^e_{k+1} - u_{k+1}\|^2 \\
&\quad \leq \Big(\|\mathbf{E}_{\xi_k}[d_k]\| + \ell_k\|u_k - u^e_{k+1}\|\Big)\|u^e_{k+1} - u_{k+1}\| + \frac{\ell_k}{2}\|u^e_{k+1} - u_{k+1}\|^2
\end{aligned}$$

Using (3.66), we obtain

$$\begin{aligned}
&\Big(\|\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\| + \ell_k\|u_k - u^e_{k+1}\|\Big)\|u^e_{k+1} - u_{k+1}\| + \frac{\ell_k}{2}\|u^e_{k+1} - u_{k+1}\|^2 \\
&\quad \leq t_k\Big(\|\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\| + t_k\ell_k\|\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|\Big)\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\| + \frac{\ell_k}{2}t_k^2\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2 \\
&\quad \leq t_k\|\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|\big(1 + t_k\ell_k\big)\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\| + \frac{\ell_k}{2}t_k^2\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2 \\
&\quad \leq \frac{1}{2}t_k^2\big(1 + t_k\ell_k\big)^2\|\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2 + \frac{1}{2}(1 + t_k^2\ell_k)\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2 \\
&\quad \leq t_k^2\big(1 + t_k\ell_k\big)^2\|d_k\|^2 + \frac{3}{2}(1 + t_k^2\ell_k)\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2 \qquad (3.71)
\end{aligned}$$

Combining (3.69) and (3.71), we deduce

$$\begin{aligned}&\mathcal{L}_{\rho_k}(u_{k+1},\lambda_k)-\mathcal{L}_{\rho_k}(u^e_{k+1},\lambda_k)+f_{\lambda_k,\xi_k}(u^e_{k+1})\\&\quad\le \mathcal{L}_{\rho_k,\xi_k}(u_k,\lambda_{k,\xi_k})-t_k\big[\big(\nu\beta_k-\frac{1}{2}t_k(1+t_k\ell_{\xi_k})^2\big)\|d_k\|^2-t_k\big(1+t_k\ell_k\big)^2\|d_k\|^2\big]\\&\qquad+\frac{1}{2}(1+t_k^2\ell_{\xi_k})\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2+\frac{3}{2}(1+t_k^2\ell_k)\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2+\mathcal{O}\Big(\frac{1}{(k+1)^{1+\varepsilon}}\Big).\end{aligned}\tag{3.72}$$

Taking the conditional expectation with respect to ξ_k, using Lemma 7, we derive from (3.72) that

$$\begin{aligned}\mathbf{E}_{\xi_k}[\mathcal{L}_{\rho_{k+1}}(u_{k+1},\lambda_{k+1})]&\le \mathcal{L}_{\rho_k}(u_k,\lambda_k)-\mathbf{E}_{\xi_k}\Big[t_k\Big(\nu\beta_k-\frac{1}{2}t_k(1+t_k\ell_{\xi_k})^2-t_k\big(1+t_k\ell_k\big)^2\Big)\|d_k\|^2\Big]\\&\quad+\mathbf{E}_{\xi_k}\Big[\frac{1}{2}(1+t_k^2\ell_{\xi_k})\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2+\frac{3}{2}(1+t_k^2\ell_k)\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2\\&\quad+\frac{1+\epsilon}{\sigma_k}\||\lambda_{k+1}-\lambda_k\||^2\Big]+\mathcal{O}\Big(\frac{1}{(k+1)^{1+\varepsilon}}\Big).\end{aligned}\tag{3.73}$$

Then, by Lemma 11,

$$\begin{aligned}\mathbf{E}_{\xi_k}[\mathcal{L}_{\rho_{k+1}}(u_{k+1},\lambda_{k+1})]&\le \mathcal{L}_{\rho_k}(u_k,\lambda_k)-\mathbf{E}_{\xi_k}\Big[t_k\Big(\nu\beta_k-\frac{1}{2}t_k(1+t_k\ell_{\xi_k})^2-t_k\big(1+t_k\ell_k\big)^2\Big)\|d_k\|^2\Big]\\&\quad+\mathbf{E}_{\xi_k}\Big[\frac{1}{2}(1+t_k^2\ell_{\xi_k})\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2\Big]+\mathbf{E}_{\xi_k}\Big[\frac{3}{2}(1+t_k^2\ell_k)\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2\Big]\\&\quad+\mathbf{E}_{\xi_k}\big[\varsigma_{1,k}t_k\|d_k\|^2\big]+2(1+\epsilon)t_{k-1}^2\|d_{k-1}\|^2+\mathcal{O}\Big(\frac{1}{(k+1)^{1+\varepsilon}}\Big).\end{aligned}\tag{3.74}$$

Next, by using Assumption 1, $(1+t_k^2\ell_k)\le t_{k,\max}$ and Assumption 2, we get for the third term of the RHS of (3.74)

$$\begin{aligned}\mathbf{E}_{\xi_k}\Big[\frac{1}{2}(1+t_k^2\ell_{\xi_k})\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2\Big]&\le\frac{1}{2}t_{k,\max}\mathbf{E}_{\xi_k}\big[\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2\big]\\&\le\frac{1}{2}\frac{t_{k,\max}\overline{\sigma}_k^2}{\mathtt{m}_k}\end{aligned}\tag{3.75}$$

and by the same manner, for the fourth term of the RHS of (3.74)

$$\begin{aligned}\mathbf{E}_{\xi_k}\Big[\frac{3}{2}(1+t_k^2\ell_k)\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2\Big]&\le\frac{3}{2}t_{k,\max}\mathbf{E}_{\xi_k}\big[\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2\big]\\&\le\frac{3}{2}\frac{t_{k,\max}\overline{\sigma}_k^2}{\mathtt{m}_k}.\end{aligned}\tag{3.76}$$

Adding (3.75) and (3.76), we obtain

$$\mathbf{E}_{\xi_k}\Big[\frac{1}{2}(1+t_k^2\ell_{\xi_k})\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2\Big]+\mathbf{E}_{\xi_k}\Big[\frac{3}{2}(1+t_k^2\ell_k)\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2\Big]\le 2\frac{t_{k,\max}\overline{\sigma}_k^2}{\mathtt{m}_k}.\tag{3.77}$$

Consequently, in order to satisfy (3.60), the following inequality must be verified

$$2\frac{t_{k,\max}\overline{\sigma}_k^2}{\mathtt{m}_k}\le\mathcal{O}\Big(\frac{1}{(k+1)^{1+\varepsilon}}\Big).\tag{3.78}$$

Then, by using the LHS of (3.77), we can simplify the RHS of (3.74) as

$$\begin{aligned}\mathbf{E}_{\xi_k}\big[\mathcal{L}_{\rho_{k+1}}(u_{k+1},\lambda_{k+1})\big]&\le\mathcal{L}_{\rho_k}(u_k,\lambda_k)-\mathbf{E}_{\xi_k}\Big[t_k\big(\nu\beta_k-\frac{1}{2}t_k(1+t_k\ell_{\xi_k})^2-t_k\big(1+t_k\ell_k\big)^2\big)\|d_k\|^2\Big]\\&\quad+\mathbf{E}_{\xi_k}\big[\varsigma_{1,k}t_k\|d_k\|^2\big]+2(1+\varepsilon)t_{k-1}^2\|d_{k-1}\|^2+\mathcal{O}\Big(\frac{1}{(k+1)^{1+\varepsilon}}\Big).\end{aligned}\tag{3.79}$$

Adding $\mathbf{E}_{\xi_k}\left[2(1+\varepsilon)t_k^2\|d_k\|^2\right]$ to both sides of (3.79), we get

$$\begin{aligned}
&\mathbf{E}_{\xi_k}\Big[\mathcal{L}_{\rho_{k+1}}(u_{k+1},\lambda_{k+1})+2(1+\varepsilon)t_k^2\|d_k\|^2\Big]\\
&\quad\leq \mathcal{L}_{\rho_k}(u_k,\lambda_k)-\mathbf{E}_{\xi_k}\Big[t_k\big(\nu\beta_k-\frac{1}{2}t_k(1+t_k\ell_{\xi_k})^2-t_k\big(1+t_k\ell_k)^2\big)\|d_k\|^2\Big]\\
&\qquad+\mathbf{E}_{\xi_k}\Big[\varsigma_{1,k}t_k\|d_k\|^2+2(1+\varepsilon)t_k^2\|d_k\|^2\Big]+2(1+\varepsilon)t_{k-1}^2\|d_{k-1}\|^2+\mathcal{O}\Big(\frac{1}{(k+1)^{1+\varepsilon}}\Big).
\end{aligned}\tag{3.80}$$

We can further obtain from (3.80) that

$$\begin{aligned}
&\mathbf{E}_{\xi_k}\Big[\mathcal{L}_{\rho_{k+1}}(u_{k+1},\lambda_{k+1})+2(1+\varepsilon)t_k^2\|d_k\|^2\Big]\\
&\quad\leq \mathcal{L}_{\rho_k}(u_k,\lambda_k)+2(1+\varepsilon)t_{k-1}^2\|d_{k-1}\|^2\\
&\qquad-\mathbf{E}_{\xi_k}\Big[t_k\big(\nu\beta_k-\frac{1}{2}t_k(1+t_k\ell_{\xi_k})^2-t_k(1+t_k\ell_k)^2-2(1+\varepsilon)t_k-\varsigma_{1,k}\big)\|d_k\|^2\Big]+\mathcal{O}\Big(\frac{1}{(k+1)^{1+\varepsilon}}\Big)\\
&\quad\leq \mathcal{L}_{\rho_k}(u_k,\lambda_k)+2(1+\varepsilon)t_{k-1}^2\|d_{k-1}\|^2-\frac{\varepsilon}{2}\mathbf{E}_{\xi_k}\left[t_k\|d_k\|^2\right]+\mathcal{O}\Big(\frac{1}{(k+1)^{1+\varepsilon}}\Big).
\end{aligned}\tag{3.81}$$

Since the $(\mathcal{L}_{\rho_k}(u_k,\lambda_k))_{k\in\mathbb{N}}$ is bounded below, by adding $-\inf_{k\in\mathbb{N}}\mathcal{L}_{\rho_k}(u_k,\lambda_k)$ to both sides of (3.81), we derive from Lemma 6 that

$$\sum_{k\in\mathbb{N}}\mathbf{E}_{\xi_k}[t_k\|d_k\|^2]<+\infty. \tag{3.82}$$

As a consequence, by Corollary 1, we obtain

$$\sum_{k\in\mathbb{N}}t_k\|d_k\|^2<+\infty\ \textbf{a.s}. \tag{3.83}$$

(i): This conclusion follows directly from (3.83) and $\|\frac{u_{k+1}-u_k}{\sqrt{t_k}}\|\leq\sqrt{t_k}\|d_k\|$.

(ii): Note that under the condition (3.9), $\varsigma_{1,k}<1$ and $t_{k-1}\leq 1$ a.s. Hence, we can derive from Lemma 11 that

$$\begin{aligned}
\sum_{k\in\mathbb{N}}\mathbf{E}_{\xi_k}\Big[\frac{1+\varepsilon}{\sigma_k}\||\lambda_{k+1}-\lambda_k\||^2\Big]&\leq\sum_{k\in\mathbb{N}}\mathbf{E}_{\xi_k}[\varsigma_{1,k}t_k\|d_k\|^2]+\sum_{k\in\mathbb{N}}2(1+\varepsilon)t_{k-1}^2\|d_{k-1}\|^2\\
&\leq\sum_{k\in\mathbb{N}}\mathbf{E}_{\xi_k}\left[t_k\|d_k\|^2+2(1+\varepsilon)t_{k-1}^2\|d_{k-1}\|^2\right]\\
&<+\infty,
\end{aligned}\tag{3.84}$$

where the last inequality follows from (i) and (3.81). Therefore, the conclusion follows from (3.84) and the update rule of λ_{k+1}.

(iii): We have

$$\begin{aligned}
\mathbf{E}_{\xi_k}\Big[\|\frac{u_{k+1}^e-u_k}{\sqrt{t_k}}\|^2\Big]&\leq 2\mathbf{E}_{\xi_k}\Big[\|\frac{u_{k+1}^e-u_{k+1}}{\sqrt{t_k}}\|^2\Big]+2\mathbf{E}_{\xi_k}\Big[\|\frac{u_{k+1}-u_k}{\sqrt{t_k}}\|^2\Big]\\
&\leq 2\mathbf{E}_{\xi_k}[t_k\|d_k-\nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2]+2\mathbf{E}_{\xi_k}\Big[\|\frac{u_{k+1}-u_k}{\sqrt{t_k}}\|^2\Big]
\end{aligned}\tag{3.85}$$

Since $t_k \in]0,1[$, the first term of the RHS of (3.85) verifies $\mathbf{E}_{\xi_k}[t_k\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2] \le \mathbf{E}_{\xi_k}[\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2]$; hence

$$
\begin{aligned}
2\mathbf{E}_{\xi_k}[t_k\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2] &+ 2\mathbf{E}_{\xi_k}\Big[\|\frac{u_{k+1}-u_k}{\sqrt{t_k}}\|^2\Big] \\
&\le 2\mathbf{E}_{\xi_k}\Big[\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2\Big] + 2\mathbf{E}_{\xi_k}\Big[\|\frac{u_{k+1}-u_k}{\sqrt{t_k}}\|^2\Big] \\
&\le 2\frac{\overline{\sigma}_k^2}{\mathtt{m}_k} + 2\mathbf{E}_{\xi_k}\Big[\|\frac{u_{k+1}-u_k}{\sqrt{t_k}}\|^2\Big] \\
&= 2\mathbf{E}_{\xi_k}\Big[\|\frac{u_{k+1}-u_k}{\sqrt{t_k}}\|^2\Big] + \mathcal{O}\Big(\frac{1}{(k+1)^{1+\epsilon}}\Big),
\end{aligned} \tag{3.86}
$$

which implies that the sequence $\Big(\mathbf{E}\Big[\|\frac{u^e_{k+1}-u_k}{\sqrt{t_k}}\|^2\Big]\Big)_{k\in\mathbb{N}}$ is summable.

(iv): Let us set

$$
\begin{cases}
r^e_{k+1} = c(u^e_{k+1}) - b - P_S(c(u^e_{k+1}) - b + \rho_k^{-1}\lambda_k) \\
r_{k+1} = c(u_{k+1}) - b - P_S(c(u_{k+1}) - b + \rho_k^{-1}\lambda_k).
\end{cases}
$$

Since P_S is nonexpansive,

$$
\begin{aligned}
\sigma_k|||r_{k+1} - r^e_{k+1}|||^2 &\le 2\sigma_k|||c(u^e_{k+1}) - c(u_{k+1})|||^2 \\
&\le 4\sigma_k|||J_c(u^e_{k+1})(u_{k+1} - u^e_{k+1})|||^2 + \sigma_k\mu_c^2\|u_{k+1} - u^e_{k+1}\|^4
\end{aligned} \tag{3.87}
$$

Then, using Assumption 1 & 2, and $t_k \le 1$, the RHS of (3.87) verifies

$$
\begin{aligned}
\mathbf{E}_{\xi_k}&\big[4\sigma_k|||J_c(u^e_{k+1})(u_{k+1} - u^e_{k+1})|||^2 + \sigma_k\mu_c^2\|u_{k+1} - u^e_{k+1}\|^4\big] \qquad (3.88)\\
&\le \mathbf{E}_{\xi_k}\Big[4\sigma_k\mu_0^2\|u_{k+1} - u^e_{k+1}\|^2 + \sigma_k\mu_c^2\|u_{k+1} - u^e_{k+1}\|^4\Big] \\
&\le \mathbf{E}_{\xi_k}\Big[4\sigma_k\mu_0^2 t_k^2\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2 + \sigma_k\mu_c^2 t_k^4\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^4\Big] \\
&\le \mathbf{E}_{\xi_k}\Big[4\mu_0^2\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^2 + \mu_c^2\|d_k - \nabla\mathcal{L}_{\rho_k}(u_k,\lambda_k)\|^4\Big]\sup_{k\in\mathbb{N}}(\sigma_k) \\
&\le \big(4\mu_0^2\frac{\overline{\sigma}_k^2}{\mathtt{m}_k} + \mu_c^2(\frac{\overline{\sigma}_k^2}{\mathtt{m}_k})^2\big)\sup_{k\in\mathbb{N}}(\sigma_k) \\
&= \mathcal{O}\Big(\frac{1}{(1+k)^{1+\epsilon}}\Big).
\end{aligned} \tag{3.89}
$$

Therefore, the sequence $(\mathbf{E}_{\xi_k}\big[4\sigma_k|||J_c(u^e_{k+1})(u_{k+1} - u^e_{k+1})|||^2 + \sigma_k\mu_c^2\|u_{k+1} - u^e_{k+1}\|^4)_{k\in\mathbb{N}}$ is summable. By (3.87), the sequence $(\sigma_k|||r_{k+1} - r^e_{k+1}|||^2)_{k\in\mathbb{N}}$ is also summable. Hence, in view of (ii) and

$$
|||r^e_{k+1}|||^2 \le 2|||r_{k+1} - r^e_{k+1}|||^2 + |||r_{k+1}|||^2,
$$

it follows that the sequence $(|||r^e_{k+1}|||^2)_{k\in\mathbb{N}}$ is summable and thus, the result is proved. □

Corollary 2 *Under the same conditions stated in Theorem 3. The followings hold almost surely,*

(i) *The sequence* $(\|\frac{u_{k+1}-u_k}{\sqrt{t_k}}\|)_{k\in\mathbb{N}}$ *is square summable.*

(ii) *The sequence* $\big(\sqrt{\sigma_k}|||c(u_{k+1}) - b - P_S(c_2(u_{k+1}) - b + \rho_k^{-1}\lambda_k)|||\big)_{k\in\mathbb{N}}$ *is square summable.*

(iii) *The sequence* $(\|\frac{u^e_{k+1}-u_k}{\sqrt{t_k}}\|)_{k\in\mathbb{N}}$ *is square summable.*

(iv) *The sequence* $\big(\sqrt{\sigma_k}|||c(u^e_{k+1}) - b - P_S(c_2(u^e_{k+1}) - b + \rho_k^{-1}\lambda_k)|||\big)_{k\in\mathbb{N}}$ *is square summable.*

Proof. The results follow from Theorem 3 as well as Corollary 1. □

3.3 Convergence to KKT points

In this section, we determine the conditions for the local convergence of the sequences $(u_k, \lambda_k)_{k\in\mathbb{N}}$ produced by the Algorithm 1 to a critical point of the augmented Lagrangian function $\mathcal{L}_\rho$ defined by (2.20).

Let us first recall the first-order KKT conditions for the constrained optimization problem at hand. If $N_C(u^\dagger)$ defines the normal cone of the set C at the point $u^\dagger$, a local minimum of Problem 1, that satisfies the regularity conditions stated here below; then, there exists a vector $\lambda \in \mathbb{R}^m$, where $m = \sum_{q=1}^M m_q$, such that the following conditions hold:

$$\begin{cases} \text{Stationarity: } -\Big(\nabla h(u^\dagger) + \displaystyle\sum_{i=1}^m \lambda_i J_{c_i}(u^\dagger)\Big) \in N_C(u^\dagger), \\ \text{Primal feasibility: } c(u^\dagger) - b \in S, \\ \text{Dual feasibility: } \lambda \in S^{\ominus 1}, \\ \text{Complementary slackness: } \big\langle \lambda \mid c(u^\dagger) - b\big\rangle = 0. \end{cases} \tag{3.90}$$

Throughout this paper, the set $\mathcal{K}$ of KKT points is non-empty. The first-order KKT conditions hold if some regularity conditions, called constraint qualification (CQ) conditions, are satisfied by feasible points. The constraint qualification of c on $Z \subseteq C$, a non-empty closed convex subset of $\mathcal{H}$, can be stated as follows: there exists a strictly positive constant $\zeta \in \,]0, +\infty[$ such that for all $v \in Y = c(Z) - b$ of $\mathbb{R}^m$, the following inequality is verified for all $u \in Z$

$$\zeta\|v\| \leq \|J_c(u)^\intercal v\|. \tag{3.91}$$

In nonlinear programming, see, e.g., [13] [51], the uniform regularity condition (3.91) is equivalent to the well-known Mangasarian-Fromovitz Constraint Qualification (MFCQ) of c on Z. Let $u^\dagger$ be a local minimizer for Problem 1. The MFCQ conditions holding at $u^\dagger$ guarantee the existence[2] and the boundedness -but not necessarily the uniqueness- of KKT multipliers (λ) at $u^\dagger$.

Throughout this paper, in addition to the lower boundedness of $\mathcal{L}_{\rho_k}(u_k, \lambda_k)$ for all $k \in \mathbb{N}$, the following conditions and properties are assumed to be verified. Let $(u_k)_{k\in\mathbb{N}} \subset Z \subseteq C$.

P1 Constraint c verifies the MFCQ conditions on Z with constant $\zeta \in \,]0, +\infty[$;
P2 The sequence $(\rho_k)_{k\in\mathbb{N}}$ is bounded from above.
P3 The primal sequence $(u_k)_{k\in\mathbb{N}}$ generated by Algorithm 1 is bounded.

The reasoning developed is to first demonstrate by means of Proposition 1 that the limit points of the subsequences produced by Algorithm 1 verify the first-order KKT conditions (3.90). The next step consists of proving that the set of limits points is non-empty (cf. Proposition 2). Knowing this property, the last step then requires to prove that, under certain conditions, the sequences produced by the algorithm converge to such limit point (cf. Corollary 3).

Proposition 1 *Assume that the conditions stated for Theorem 3 hold. Suppose, according to property* ***P2****, that $(\rho_k)_{k\in\mathbb{N}}$ is bounded from above. Let $((u_{n_k}, \lambda_{n_k}))_{k\in\mathbb{N}}$ be a subsequence of $((u_k, \lambda_k))_{k\in\mathbb{N}}$ such that*

$$\begin{cases} (u_{n_k}, \lambda_{n_k}) \to (u^\dagger, \lambda^\dagger), \\ (u_{n_k+1} - u_{n_k})/t_{n_k} \to 0, \\ c(u_{n_k+1}) - b - P_S(c(u_{n_k+1}) - b + \rho_{n_k}^{-1}\lambda_{n_k}) \to 0. \end{cases} \tag{3.92}$$

Then, the limit point $(u^\dagger, \lambda^\dagger)$ verifies the KKT conditions (3.90).

---
1 where $S^\ominus$ refers to the polar cone of S, see infra for its definition.
2 The set of KKT multipliers (λ) at $u^\dagger$ is nonempty.

Proof. (i). Primal feasibility. Since t_k $(= \theta^j, j \in \mathbb{N}) \le 1$,

$$\|u_{n_k+1} - u_{n_k}\| \le \|(u_{n_k+1} - u_{n_k})/t_{n_k}\|. \tag{3.93}$$

Hence, $u_{k+1} - u_k \to 0$ implies that

$$u_{n_k+1} \to u^\dagger. \tag{3.94}$$

Since P_S and c are continuous, it follows that

$$\lim_{k\to\infty} P_S(c(u_{n_k+1}) - b) = P_S(c(u^\dagger) - b). \tag{3.95}$$

By assumption $d_S(c(u_{n_k+1})-b) \le \||c(u_{n_k+1})-b-P_S(c(u_{n_k+1})-b+\rho_{n_k}^{-1}\lambda_{n_k})\|| \to 0$; hence, it follows that

$$c(u^\dagger) - b = P_S(c(u^\dagger) - b) \in S, \tag{3.96}$$

ii) Dual feasibility and Complementarity slackness: consider the negative of the dual cone of S[3], i.e., the polar cone $S^\ominus$ of S defined as $S^\ominus = \{u \mid \sup \langle S \mid u\rangle \le 0\}$. Then, by [5, Theorem 6.30], we have

$$(\forall a \in \mathbb{R}^m) \quad a = P_S a + P_{S^\ominus} a, \tag{3.97}$$

In turn, by using [5, Proposition 29(ii)], we obtain for the constraints $c(u) - b \in S$, and $a_{k+1} = c(u_{k+1}) - b$ that

$$\begin{aligned}
\rho_k\Big(a_{k+1} - P_S\big(a_{k+1} + \rho_k^{-1}\lambda_k\big)\Big) &= \rho_k\Big(P_{S^\ominus}\big(a_{k+1} + \rho_k^{-1}\lambda_k\big) - \rho_k^{-1}\lambda_k\Big)\\
&= \rho_k P_{S^\ominus}\big((\rho_k a_{k+1} + \lambda_k)/\rho_k\big) - \lambda_k\\
&= \rho_k P_{S^\ominus/\rho_k}\big((\rho_k a_{k+1} + \lambda_k)/\rho_k\big) - \lambda_k\\
&= P_{S^\ominus}\big(\rho_k a_{k+1} + \lambda_k\big) - \lambda_k.
\end{aligned} \tag{3.98}$$

The latter equality implies that for all k, the following identity is verified

$$\lambda_k = P_{S^\ominus}\big(\rho_k(c(u_{k+1}) - b) + \lambda_k\big) - \rho_k\big(c(u_{k+1}) - b - P_S(c(u_{k+1}) - b + \rho_k^{-1}\lambda_k)\big). \tag{3.99}$$

Following property **P2**, the sequence $(\rho_k)_{k\in\mathbb{N}}$ is bounded; hence, we derive from (3.99) that

$$\lambda^\dagger = P_{S^\ominus}\big(\rho^\dagger(c(u^\dagger) - b) + \lambda^\dagger\big) \in S^\ominus, \tag{3.100}$$

where $\rho^\dagger$ is a cluster point of $(\rho_{n_k})_{k\in\mathbb{N}}$. Moreover, since $\lambda^\dagger/2 \in S^\ominus$, by the Projection theorem [5, Theorem 3.16], we get

$$\big\langle c(u^\dagger) - b \mid\mid \lambda^\dagger\big\rangle = 0. \tag{3.101}$$

iii) Stationarity: We also have

$$\begin{aligned}
v_{n_k} &:= \lambda_{n_k} + \rho_{n_k}\big(c(u_{n_k}) - b - P_S(c(u_{n_k}) - b + \rho_{n_k}^{-1}\lambda_{n_k})\big) \to \lambda^\dagger\\
\nabla\mathcal{L}_{\rho_k}(u_k, \lambda_k) &= \big(\nabla h(u_{n_k}) + J_c(u_{n_k})^\intercal v_{n_k}\big) \to d^\dagger = \big(\nabla h(u^\dagger) + J_c(u^\dagger)^\intercal\lambda^\dagger\big).
\end{aligned} \tag{3.102}$$

Next, we deduce from the definition of u^e_{k+1} that

$$(u_{n_k} - u^e_{n_k+1})/t_{n_k} + \nabla\mathcal{L}_{\rho_k}(u_k, \lambda_k) \in N_C(u^e_{n_k+1}). \tag{3.103}$$

By Corollary 2(iii), $(u_{n_k} - u^e_{n_k+1})/t_{n_k} \to 0$ and $u^e_{n_k+1} \to u^\dagger$. Hence, it follows from (3.102) that

$$d^\dagger \in N_C(u^\dagger). \tag{3.104}$$

The expressions (3.96), (3.100), (3.101) and (3.104) are exactly the first-order KKT conditions (3.90). Consequently, the limit point $(u^\dagger, \lambda^\dagger) \in \mathcal{K}$. □

Note that when the set C is bounded, the primal sequence is bounded. In the general case, we have the following result.

[3] The dual cone $S^\star$ is always convex irrespective of the original set

Proposition 2 *Assume that the conditions stated for Theorem 3 hold, the sequence $(u_k)_{k\in\mathbb{N}}$ is bounded (property **P3**), and the sequence $(\rho_k)_{k\in\mathbb{N}}$ is bounded from above (property **P2**). We further suppose that the subsequences $(d_{n_k})_{k\in\mathbb{N}}$ is bounded. Then, the subsequence $(u_{n_k}, \lambda_{n_k})_{k\in\mathbb{N}}$ is bounded. Consequently, the set of cluster points of $(u_k, \lambda_k)_{k\in\mathbb{N}}$ is non-empty.*

Proof. It follows from our assumption that there exists a constant $\mathcal{O}(1)$ such that $(\forall k \in \mathbb{N})\ \|d_{n_k}\| \leq \mathcal{O}(1)$. Hence,

$$(\forall k \in \mathbb{N})\ \|\nabla \mathcal{L}_{\rho_{n_k}}(u_{n_k}, \lambda_{n_k})\| = \|\mathbf{E}_{\xi_{n_k}}[d_{n_k}]\| \leq \mathbf{E}_{\xi_{n_k}}[\|d_{n_k}\|] \leq \mathcal{O}(1). \tag{3.105}$$

Note that

$$\begin{aligned}\nabla \mathcal{L}_{\rho_k}(u_k, \lambda_k) &= \nabla h(u_{n_k}) + J_c(u_{n_k})^\intercal v_{n_k} \\ &\text{with } v_{n_k} = \lambda_{n_k} + \rho_{n_k}\big(c(u_{n_k}) - b - P_S(c(u_{n_k}) - b + \rho_{n_k}^{-1}\lambda_{n_k})\big),\end{aligned} \tag{3.106}$$

which implies that

$$\|J_c(u_{n_k})^\intercal v_{n_k}\| \leq \|\nabla \mathcal{L}_{\rho_{n_k}}(u_{n_k}, \lambda_{n_k})\| + \|\nabla h(u_{n_k})\|. \tag{3.107}$$

Since the primal sequence $(u_k)_{k\in\mathbb{N}}$ is bounded and since the function h is differentiable with μ_h-Lipschitz continuous gradient ∇h, it follows from (3.105) and (3.107) that

$$\|J_c(u_{n_k})^\intercal v_{n_k}\| \leq \mathcal{O}(1). \tag{3.108}$$

Now, using the Mangasarian-Fromowitz (MF) condition of c on Z (property **P1**), there exists a strictly positive constant ζ_c such that

$$\zeta_c \|\|(v_{n_k})\|\| \leq \|J_c(u_{n_k})^\intercal v_{n_k}\|. \tag{3.109}$$

Hence, $(v_{n_k})_{k\in\mathbb{N}}$ defines a sequence bounded by

$$\zeta_c \|\|(v_{n_k})\|\| \leq \|J_c(u_{n_k})^\intercal v_{n_k}\| \leq \mathcal{O}(1).$$

Since $(\rho_k)_{k\in\mathbb{N}}$ is bounded and $(u_k)_{k\in\mathbb{N}}$ is bounded, By Step 1 of Algorithm 1, the sequence $\Big(\rho_{n_k}\big(c(u_{n_k}) - b - P_S(c(u_{n_k}) - b + \rho_{n_k}^{-1}\lambda_{n_k})\big)\Big)_{k\in\mathbb{N}}$ is bounded. In turn, by the definition of v_{n_k} defined by (3.106), the sequence $(\lambda_{n_k})_{k\in\mathbb{N}}$ is also bounded. Consequently, the set of cluster points of $(u_k, \lambda_k)_{k\in\mathbb{N}}$ is non-empty. □

Corollary 3 *Assume the conditions stated for Theorem 3 hold. Suppose, the properties **P1**, **P2**, and **P3** are satisfied. If $\sum_{k\in\mathbb{N}} t_k = \infty$, and $\|d_k\| = \mathcal{O}(1)$, and $k\rho_k \to \infty$. Then, there exists a subsequence $(u_{n_k}, \lambda_{n_k})_{k\in\mathbb{N}}$ of $(u_k, \lambda_k)_{k\in\mathbb{N}}$ that converges to a limit point $(u^\dagger, \lambda^\dagger) \in \mathcal{K}$.*

Proof. Since $\sum_{k\in\mathbb{N}} t_k = \infty$, and by Theorem 3, $\sum_{k\in\mathbb{N}} t_k\|d_k\|^2 < +\infty$, we get $\inf_{k\in\mathbb{N}} \|d_k\| = 0$. Then there exists a subsequence $(d_{p_k})_{k\in\mathbb{N}}$ such that $\lim_{k\to\infty} d_{p_k} = 0$. By Proposition 2, there exists a subsequence $(u_{n_k}, \lambda_{n_k})_{k\in\mathbb{N}}$ of $(u_{p_k}, \lambda_{p_k})_{k\in\mathbb{N}}$ of such that $(u_{n_k}, \lambda_{n_k}) \to (u^\dagger, \lambda^\dagger)$. We first have

$$\|\frac{u_{n_k+1} - u_{n_k}}{t_{n_k}}\| \leq \|d_{n_k}\| \to 0. \tag{3.110}$$

Moreover, it follows Step 1 of Algorithm 1 that

$$\begin{aligned}\rho_k \|\|c(u_k) - b - P_S\big(c(u_k) - b + \rho_k^{-1}\lambda_k\big)\|\|^2 &\leq \min_{1\leq i\leq k} \|u_i - u_{i-1}\|^2 \\ &\leq \frac{1}{k}\sum_{i=1}^{k} \|u_i - u_{i-1}\|^2 \\ &\leq \frac{1}{k}\sum_{i=1}^{k} t_i^2\|d_i\|^2 \\ &\leq \mathcal{O}(1/k),\end{aligned} \tag{3.111}$$

where the last estimation follows from Corollary 2. Since $k\rho_k \to \infty$, we get $|||c(u_k) - b - P_S\big(c(u_k) - b + \rho_k^{-1}\lambda_k\big)||| \to 0$ and hence

$$|||c(u_{k+1}) - b - P_S\big(c(u_{k+1}) - b + \rho_k^{-1}\lambda_k\big)||| \to 0. \tag{3.112}$$

In view of Proposition 1, we get $(u^\dagger, \lambda^\dagger) \in \mathcal{K}$. □

4 Iteration Complexity

In this Section, we characterize the iteration complexity of the proposed Algorithm in terms of the difference $\Delta\mathcal{L}_k(\cdot, \lambda_k)$ and the feasibility. By iteration complexity, we refer here to the number of iterations required to obtain an approximate ε-KKT point of Problem 1 by means of the proposed algorithm (cf. Section 3.1).

Theorem 4 *Let $\mu_{c_{i_p}}$ and $\mu_{h_{i_p}}$ be, respectively, the Lipschitz constant of $J_{c_{i_p}}$ and ∇h_{i_p}. Set*

$$\mu_{c,\xi_k} = \frac{1}{2m_k}\sum_{p=1}^{m_k} \mu_{c_{i_p}} \text{ and } \mu_{h,\xi_k} = \frac{1}{2m_k}\sum_{p=1}^{m_k} \mu_{h_{i_p}} \tag{4.1}$$

Suppose that $C = \mathcal{H} \times \mathcal{G}$. Assume that the conditions stated in Theorem 3 are satisfied. Then, t_k and u_k verify the following

$$t_k \geq \frac{2\theta(1-\nu)\beta_{\xi_k}}{\beta_{\xi_k}\mu_{c,\xi_k}\rho_k + \mu_{h,\xi_k}} \text{ and } \min_{0\leq i\leq k} \frac{2\theta(1-\nu)\beta_{\xi_i}}{\beta_{\xi_i}\mu_{c,\xi_i}\rho_i + \mu_{h,\xi_i}} \|\frac{u_{i+1} - u_i}{t_i}\|^2 = \mathcal{O}(1/(k+1)), \tag{4.2}$$

where the constant of $\mathcal{O}$ is a random variable which is independent of k, and

$$\beta_{\xi_k} := \max_{0\leq t\leq 1, 1\leq p\leq m_k} \Big(d_{S_{i_p}}(c_{i_p}(u_k - td_k) - b_{i_p} + \rho_k^{-1}\lambda_{k,i_p}) + d_{S_{i_p}}(c_{i_p}(u_k) - J_{c_{i_p}}(u_k)(td_k) - b_{i_p} + \rho_k^{-1}\lambda_{k,i_p})\Big). \tag{4.3}$$

Suppose that there exists a positive constant β such that

$$\beta_{\xi_k} t_k \leq \beta \text{ and } 2\theta(1-\nu)\beta_{\xi_k} - t_k\mu_{h,\xi_k} \geq \epsilon_1 \tag{4.4}$$

Then, ρ_k is bounded below by $\rho_{\min} := \epsilon_1/(\beta\mu_c^e)$ with $\mu_c^e := \mathbf{E}_{\xi_k}[\mu_{c,\xi_k}]$. Moreover,

$$|||c(u_k) - b - P_S\big(c(u_k) - b + \rho_k^{-1}\lambda_k\big)||| \leq \mathcal{O}(1/\sqrt{k}). \tag{4.5}$$

Proof. Suppose that the line search step (3.11) does not yet terminate at a certain $t = \theta^j,\ \theta \in]0,1]$. Then, we have

$$\nu t\Delta f_{\lambda_k,\xi_k}(u_k; -d_k) \leq f_{\lambda_k,\xi_k}(u_k - td_k) - f_{\lambda_k,\xi_k}(u_k). \tag{4.6}$$

Following the definition of the function f_{λ_k,ξ_k}, the terms of the right-hand side in (4.6) can be written respectively as

$$f_{\lambda_k,\xi_k}(u_k - td_k) = \psi_{\xi_k}(c_{\xi_k}(u_k - td_k)) + h_{\xi_k}(u_k - td_k), \tag{4.7}$$

$$f_{\lambda_k,\xi_k}(u_k) = \psi_{\xi_k}(c_{\xi_k}(u_k)) + h_{\xi_k}(u_k). \tag{4.8}$$

Thus, the right-hand side of (4.6) becomes

$$f_{\lambda_k,\xi_k}(u_k - td_k) - f_{\lambda_k,\xi_k}(u_k) = [\psi_{\xi_k}(c_{\xi_k}(u_k - td_k)) - \psi_{\xi_k}\big(c_{\xi_k}(u_k)\big)] + [h_{\xi_k}(u_k - td_k) - h_{\xi_k}(u_k)]. \tag{4.9}$$

Using the definition of $\Delta f_{\lambda_k,\xi_k}(u_k; -td_k)$, the second term

$$-\psi_{\xi_k}(c_{\xi_k}(u_k)) = \Delta f_{\lambda_k,\xi_k}(u_k; -td_k) - \psi_{\xi_k}(c_{\xi_k}(u_k) - J_{c_{\xi_k}}(u_k)td_k) + \langle\nabla h_{\xi_k}(u_k) \mid td_k\rangle\,;$$

thus, the right-hand side of (4.6) can be expressed as

$$f_{\lambda_k,\xi_k}(u_k - td_k) - f_{\lambda_k,\xi_k}(u_k) = \Delta f_{\lambda_k,\xi_k}(u_k; -td_k) + \psi_{\xi_k}(c_{\xi_k}(u_k - td_k)) - \psi_{\xi_k}(c_{\xi_k}(u_k) - J_{c_{\xi_k}}(u_k)td_k) + h_{\xi_k}(u_k - td_k) - h_{\xi_k}(u_k) + \langle \nabla h_{\xi_k}(u_k) \mid td_k \rangle . \tag{4.10}$$

By Lemma 4, the function $\Psi_{\xi_k} : \mathbb{R}^K \times \mathbb{R} \to \mathbb{R}$: $(u, \xi) \mapsto \psi_{\xi_k}(u) + \mathrm{Id}_R(\xi)$ is convex. Hence, by defining $\overline{c}_{\xi_k} : \mathbb{R}^K \to \mathbb{R}^m \times]-\infty, +\infty] : u \mapsto \overline{c}_{\xi_k}(u) =(c_{\xi_k}(u), c_0(u))$, it follows from [16, Lemma 3.1] that

$$\begin{aligned} \Delta f_{\lambda_k,\xi_k}(u_k; td_k) &= \Delta_0(\Psi_{\xi_k} \circ \overline{c}_{\xi_k})(u_k; td_k) \\ &\le t\Delta_0(\Psi_{\xi_k} \circ \overline{c}_{\xi_k})(u_k; d_k) = t\Delta f_{\lambda_k,\xi_k}(u_k; d_k). \end{aligned} \tag{4.11}$$

In turn, simple calculations show that

$$\begin{aligned} &\Delta f_{\lambda_k,\xi_k}(u_k, -td_k) + \psi_{\xi_k}(c_{\xi_k}(u_k - td_k)) - \psi_{\xi_k}(c_{\xi_k}(u_k) - J_{c_{\xi_k}}(u_k)td_k) \\ &\quad \le t\Delta f_{\lambda_k,\xi_k}(u_k; -d_k) + \psi_{\xi_k}(c_{\xi_k}(u_k - td_k)) - \psi_{\xi_k}(c_{\xi_k}(u_k) - J_{c_{\xi_k}}(u_k)td_k). \end{aligned} \tag{4.12}$$

To determine the upper bound of the third term in the RHS of (4.12), we make use of the Assumption 1 and the $\mu_{c_{i_p}}$- Lipschitz continuity property of $J_{c_{i_p}}$ to obtain

$$\begin{aligned} &\psi_{\xi_k}(c_{\xi_k}(u_k - td_k)) - \psi_{\xi_k}(c_{\xi_k}(u_k) - J_{c_{\xi_k}}(u_k)td_k) \\ &= \frac{\rho_k}{2\mathtt{m}_k} \sum_{p=1}^{\mathtt{m}_k} \Big(d^2_{S_{i_p}}(c_{i_p}(u_k - td_k) - b_{i_p} + \rho_k^{-1}\lambda_{k,i_p}) - d^2_{S_{i_p}}(c_{i_p}(u_k) - J_{c_{i_p}}(u_k)td_k - b_{i_p} + \rho_k^{-1}\lambda_{k,i_p}) \Big) \\ &\le \frac{\beta_{\xi_k}\rho_k}{2\mathtt{m}_k} \sum_{p=1}^{\mathtt{m}_k} \Big(d_{S_{i_p}}(c_{i_p}(u_k - td_k) - b_{i_p} + \rho_k^{-1}\lambda_{i_p}) - d_{S_{i_p}}(c_{i_p}(u_k) - J_{c_{i_p}}(u_k)td_k - b_{i_p} + \rho_k^{-1}\lambda_{i_p}) \Big) \\ &\le \frac{\beta_{\xi_k}\rho_k}{2\mathtt{m}_k} \sum_{p=1}^{\mathtt{m}_k} \|c_{i_p}(u_k - td_k) - c_{i_p}(u_k) - J_{c_{i_p}}(u_k)td_k\| \\ &\le \frac{\beta_{\xi_k}\mu_{c,\xi_k}\rho_k}{2} \|td_k\|^2. \end{aligned} \tag{4.13}$$

Moreover, since the gradient of h_{ξ_k} is μ_{h,ξ_k}-Lipschitz continuous, we also have

$$h_{\xi_k}(u_k - td_k) - h_{\xi_k}(u_k) + \langle \nabla h_{\xi_k}(u_k) \mid td_k \rangle \le \frac{\mu_{h,\xi_k}}{2} t^2 \|d_k\|^2. \tag{4.14}$$

Therefore, we derive from (4.6), (4.12), (4.13) and (4.14) that

$$\nu t\Delta f_{\lambda_k,\xi_k}(u_k; -d_k) \le t\Delta f_{\lambda_k,\xi_k}(u_k; -d_k) + \frac{1}{2}(\beta_{\xi_k}\mu_{c,\xi_k}\rho_k + \mu_{h,\xi_k})t^2\|d_k\|^2, \tag{4.15}$$

which implies that

$$t \ge \frac{2(1-\nu)}{(\beta_{\xi_k}\mu_{c,\xi_k}\rho_k + \mu_{h,\xi_k})\|d_k\|^2} |\Delta f_{\lambda_k,\xi_k}(u_k; -d_k)|. \tag{4.16}$$

In turn, the line search step (3.11) terminates at $t_k > 0$ (since $\nu \in]0, 1[$) with

$$t_k \ge \frac{2\theta(1-\nu)}{(\beta_{\xi_k}\mu_{c,\xi_k}\rho_k + \mu_{h,\xi_k})\|d_k\|^2} |\Delta f_{\lambda_k,\xi_k}(u_k; -d_k)|. \tag{4.17}$$

In view of Lemma 8,

$$|\Delta f_{\lambda_k,\xi_k}(u_k; -d_k)| \ge \beta_k\|d_k\|^2. \tag{4.18}$$

It follows by combining (4.18) with (4.17) that

$$t_k \ge \frac{2\theta(1-\nu)\beta_k}{\beta_{\xi_k}\mu_{c,\xi_k}\rho_k + \mu_{h,\xi_k}}, \tag{4.19}$$

which is the first assertion in (4.2). Hence, by involving Corollary 2, we deduce

$$\sum_{k\in\mathbb{N}} \frac{2\theta(1-\nu)\beta_k\|d_k\|^2}{\beta_{\xi_k}\mu_{c,\xi_k}\rho_k+\mu_{h,\xi_k}} < +\infty, \tag{4.20}$$

which implies the second assertion in (4.2). Moreover, from (4.3) and (4.19), we also obtain

$$\beta\mu_{c,\xi_k}\rho_k \geq \beta_{\xi_k}\mu_{c,\xi_k}\rho_k t_k \geq 2\theta(1-\nu)\beta_{\xi_k} - t_k\mu_{h,\xi_k} \geq \epsilon_1. \tag{4.21}$$

This inequality implies that the sequence $(\rho_k)_{k\in\mathbb{N}}$ is bounded below by $\rho_{\min}$. Note also that

$$\sum_{k\in\mathbb{N}} \|u_{k+1}-u_k\|^2 \leq \sum_{k\in\mathbb{N}} t_k\|d_k\|^2 < +\infty. \tag{4.22}$$

Hence, using the Step 1 of Algorithm 1, it follows that

$$\sqrt{\rho_{\min}}^2 \||c(u_k)-b-P_S\big(c(u_k)-b+\rho_k^{-1}\lambda_k\big)\||^2 \leq \min_{1\leq i\leq k}\|u_k-u_{k-1}\|^2 = \mathcal{O}(1/k), \tag{4.23}$$

which proves the last conclusion. □

5 Comparison and Related Work

The augmented Lagrangian method (ALM) is one of the most common approaches for solving nonlinear constrained problems. However, as stated in the Introduction section, constraints are often assumed to be convex implying that equality constraint functions must be affine and inequality constraint functions must be convex. With the proposed method, the minimization of the (possibly) nonconvex objective function h can be subject to nonlinear equality and inequality constraints.

The handling of such problems has been the subject of significant efforts, including LANCELOT, GENCAN, and ALGENCAN due to the ability of ALM to solve large-scale problems. The latter (and most recent) algorithmic scheme iterates by approximately minimizing the so-called PHR-Augmented Lagrangian function subject to bound constraints as well as updating both the penalty parameter and the Lagrange multipliers. ALGENCAN includes a decision that takes into account improvements in both the feasibility and complementary conditions. If both feasibility and complementary conditions were improved, it is considered that the penalty parameter is sufficiently large; thus, it is not further increased. Otherwise, it is multiplied by factor large than 1. ALGENCAN imposes that the KKT multiplier estimates must be bounded by explicitly projecting the estimates on a compact box after each update. The main reason invoked is to preserve the property of external penalty methods such that global minimizers of the original problem are obtained if each outer iteration computes a global minimizer of the subproblem. The boundedness of penalty parameters imposes in turn to assume that the KKT multipliers are within the bounds imposed by the algorithm. For these purposes, ALGENCAN uses safeguarded KKT multipliers such that limit points converge to KKT points under the Constant Positive Linear Dependence (CPLD) constraint qualification –which is weaker than MFCQ– and exhibit good properties in terms of penalty parameter boundedness. Although insufficient to ensure convergence for general nonconvex problems, as shown, for instance, in [10, Section 2.1], the properties of this algorithmic scheme have been shown to be competitive against alternatives such as interior point methods.

Recently, several ALM-based methods have been proposed to deal with the minimization of nonconvex objective functions subject to nonconvex equality constraints [51] and even fewer with inequality constraints [61]. In [61], authors aim to minimize over $x \in \mathbb{R}^n$, the composite function $f(x) + g(x)$ subject to equality constraints $c(x) = 0$ and inequality constraints $d(x) \leq 0$ with f continuously differentiable but possibly nonconvex, g closed convex but possibly nonsmooth, and c, d being vector functions from $\mathbb{R}^n \to \mathbb{R}^l$. Further, in addition to uniform regularity conditions (to ensure near feasibility of a near-stationary point to the augmented Lagrangian function), their proposed method assumes weak convexity of both the function f and each component of the vector function c; these assumptions significantly restrict its applicability. Constraints are then handled by introducing slack variables

$s \geq 0$, leading the reformulation of the inequality constraints as $d(x) + s = 0$. Using the boundedness of the multipliers $\{y_k\}$, authors then show that their algorithm enables to reach an $\epsilon-$KKT point $(\bar{x}; \bar{s})$ with a corresponding multiplier $(\bar{y}, \bar{z})$. It turns out that $\bar{x}$ is an $O(\epsilon)-$KKT point of the original problem in terms of primal feasibility, dual feasibility, and the complementarity condition.

The former [51] applies the accelerated proximal gradient method as proposed by [25] to find an approximate primal solution to the ALM subproblems. The latter [61], referred to as Rate-Improved (RI)-iALM uses an inexact proximal point (iPP) method to approximately solve each ALM subproblem. The iPP procedure itself relies on the accelerated proximal gradient (APG) algorithm to solve each iPP subproblem. This combination yields a triple loop algorithm: each iteration k of the main ALM routine calls the iPP procedure to compute a x_{k+1} iterate that is itself the output obtained after running t iterations of the APG algorithm. This triple loop structure contrasts with the single-loop characterizing the proposed stochastic ALM algorithm. In [61], authors report that this change of subroutine for the solving of nonconvex subproblems enables to obtain order-reduced complexity by geometrically increasing the penalty parameter in ALM compared to [51] as well as more stable and efficient numerical results under the same assumptions. The complexity result of iPP has the best dependence on the smoothness and weak convexity constant (per iteration); however, for most problems, their explicit formula remains unknown and the corresponding parameters tuned. Table 5 compares the proposed stochastic ALM algorithm with inexact ALM (iALM) [51] and Rate-Improved ALM (RI-ALM) [61]. The complexity in the number of iterations (last column) is demonstrated in Section 4.

Table 1 Comparison of ALM methods for nonconvex nonlineary constrained problems

Method	Type	Objective	Constraints	Type	Regularity Condition	Complexity
iALM [37]	Inexact	Convex	Convex	Inequality		$\tilde{O}(\varepsilon^{-1})$
iALM [51]	Inexact	Nonconvex	Nonconvex	Equality	[51, Equation 18]	$\tilde{O}(\varepsilon^{-4})$
RI-iALM [61]	Inexact	Nonconvex	Convex Nonconvex	Equality Inequality†	[61, Assumption 3]	$\tilde{O}(\varepsilon^{-3})$
This paper	Inexact (Line Search)	Nonconvex	Convex Nonconvex	Equality Inequality	Assumption 1	$\tilde{O}(1/\sqrt{k})$ $\sim \tilde{O}(\varepsilon^{-2})$

5.1 Numerical Evaluation and Comparison

In this section, we detail the realization of numerical experiments to illustrate the empirical performance of the proposed algorithm compared to the one developed [61] and the ALGENCAN method as implemented in the `nlopt` framework [34]. For this purpose, we consider the Generalized Eigenvalue (GEV) problem in Section 5.1.1 and the max-cut problem in Section 5.1.2. All executions were performed on GNU `Octave` version 7.2 [22].

5.1.1 Generalized Eigenvalue Problem (GEV)

Let U, V be symmetric matrices in $\mathbb{R}^{d\times d}$. The generalized eigenvalue problem can be formulated as

$$\begin{aligned} &\text{minimize } \langle x \mid Ux\rangle \\ &\text{subject to } \langle x \mid Vx\rangle = 1. \end{aligned} \tag{5.1}$$

The problem (5.1) is a particular case of Problem (1.1) with $C = \mathbb{R}^d$, the objective function $h\colon x \mapsto \langle x \mid Ux\rangle$ whose gradient is Lipschitz continuous with constant $\mu_h = 2\|U\|$, $c_1\colon x \mapsto \langle x \mid Vx\rangle$ whose gradient is Lipschitz continuous with constant $\mu_c = 2\|V\|$, $b_1 = 1 \in \mathbb{R}$, and $c_2 \equiv 0$ with $b_2 \equiv 0$.

For the sake of our numerical experiments, we implement Algorithm 1 with the matrices U and V defined by $U = w\,\mathrm{diag}(1/i^2)w^\top$ and $V = z\,\mathrm{diag}(1/i)z^\top$, where w and z are orthogonal matrices. Since V is positive definite, we have $\|x\|^2 \leq \|V^{-1}\|$ which defines the set C.

The following executions have been performed. First, Algorithm 1 is executed up to $n = 15000$ iterations on matrices of dimension $d = 1000$ to 10000 by increments of 1000. We record the computation time (in seconds) as reported by the `Octave v7.2` solver and the relative gap to the optimal value $h^\star$

$$\frac{\mid h(x_n) - h^\star \mid}{\mid h^\star \mid}. \tag{5.2}$$

Then Algorithm 1 is executed on matrices of dimension $d \times d$, where d = 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000 and 10000. Executions are stopped when the relative gap -defined by (5.2)- reaches a value obtained by means of the alternative method, i.e., either RI-ALM or ALGENCAN. In Table 2 (and 4), we report the performance results obtained: Columns 2, 3 and 4 of these two tables list respectively the feasibility, the relative gap (to the optimal solution given by the opposite of the largest eigenvalue) and the computation time obtained when running this algorithm to reach this gap. The values of the relative gap reported in the Tables 2 and 4 show that the value obtained with Our Algorithm is always lower than the one obtained with both RI-ALM and ALGENCAN.

Table 2 compares these results against those collected by executing the RI-ALM algorithm [61]. The relative gap (Column 6 of Table 2) is obtained by running this triple loop algorithm until reaching the fixed computation time obtained with Our Algorithm (Column 4). Next, Column 7 lists the computation time required by RI-ALM to reach the relative gap value reported in Column 3. Column 7 also reports the number of ALM iterations, iPP iterations and APG iterations required by RI-ALM.

The following setting is considered for the execution of the RI-ALM algorithm: the parameter $\varepsilon = 0.001$ except for $d = 1000$ ($\varepsilon = 0.0001$), $\sigma = 2$, $\beta_0 = 0.5$ and since the general formula of the update of the dual step size [61, Equation 17] is used, $M = 0.001$ and $q = 1$. The initial values for the primal and dual variables are $x_0 = 0.5 \times \mathrm{randn}(n, r)$, where r is the rank set to 20, and $y_0 = 0$.

Importantly, in order to obtain these results, we need to define the matrix V as $V + \frac{1}{2} I_{d\times d}$ (to ensure that the regularity condition is met) but also to modify the smoothness $\hat{L}_k$ and weak convexity parameters $\hat{\rho}_k$ compared to their suggested setting in the original specification of the RI-ALM algorithm. Indeed, since both GEV and max-cut problems do not have an explicit compact constraint set, [61, Assumption 2] is not satisfied. Nevertheless, the feasibility region of (5.1) is bounded due to the positive definiteness of the matrix V.

Table 2 GEV problem: Comparison between Algorithm 1 and RI-ALM method

	Our Algorithm			RI-ALM		
d	Feasibility	Relative Gap	Computation Time (s)	Feasibility	Relative Gap	Comp. Time (s) Iterations
1000	2.96873e-13	2.98361e-13	17.9452	9.08386e-03	9.09256e-03	26.63848 27-464-13352
2000	6.29274e-13	1.49437e-12	62.3169	5.73541e-08	7.42032e-08	79.64402 63-360-15239
3000	2.12496e-13	3.77504e-12	331.3266	1.41369e-07	4.38408e-08	466.70770 62-249-19789
4000	1.40776e-13	4.70914e-12	1344.7046	1.52398e-07	3.61829e-09	1263.4407 73-276-24072
5000	7.26085e-14	1.04429e-12	3221.6046	1.08289e-07	1.67550e-08	2632.8747 88-293-28502
6000	6.10622e-14	7.29320e-12	4150.0458	3.76529e-08	3.30732e-08	4118.6702 89-284-30428
7000	3.44169e-14	3.29640e-12	6902.2255	2.21097e-08	1.45204e-08	4939.5622 215-417-34203
8000	1.37667e-14	2.62173e-12	8769.7175	8.59245e-10	8.59508e-10	8364.7827 1197-1421-51453
9000	2.33146e-14	3.88959e-12	10516.1510	2.86114e-08	7.71071e-09	11404.8650 104-333-41691
10000	1.73194e-14	3.27909e-12	13740.4867	1.23436e-08	4.69542e-09	13896.0151 106-321-43546

Table 3 lists for each matrix size d, the value of the parameters β_0, M, and q, as well as the update rule for the parameters $\hat{\rho}_k$ (weak convexity), $\hat{L}_k$ (smoothness) and ω_k (dual step size). The value of the parameter σ (in $\beta_k = \beta_0\sigma^k$) is set to 1 for all executions of the RI-ALM algorithm.

Table 4 GEV problem: Comparison between Our Algorithm 1 and ALGENCAN

	Our Algorithm			**ALGENCAN**		
d	Feasibility	Relative Gap	Computation Time (s)	Feasibility	Relative Gap	Computation Time (s)
1000	2.96873e-13	2.98361e-13	17.9452	1.79856e-14	4.30595e-13	31.2438
2000	6.29274e-13	1.49437e-12	62.3169	3.32068e-13	1.58337e-12	332.6893
3000	2.12496e-13	3.77504e-12	331.3266	1.23679e-13	3.91351e-12	1126.5961
4000	1.40776e-13	4.70914e-12	1344.7046	6.66134e-16	4.95375e-12	2809.7044
5000	7.26085e-14	1.04429e-12	3221.6046	4.31188e-12	1.12477e-12	4006.3148
6000	6.10622e-14	7.29320e-12	4150.0458	2.73531e-11	7.42048e-12	8191.0748
7000	3.44169e-14	3.29640e-12	6902.2255	4.32321e-13	3.34667e-12	15519.1857
8000	1.37667e-14	2.62173e-12	8769.7175	1.22125e-15	2.65238e-12	19020.3227
9000	2.33146e-14	3.88959e-12	10516.1510	5.93174e-12	3.99288e-12	21599.7243
10000	1.73194e-14	3.27909e-12	13740.4867	1.73195e-13	3.36005e-12	25306.0546

Table 3 RI-ALM Parameters

d	β_0	$\hat{\rho}_k$	$\hat{L}_k$	**M**	**q**	ω_k
1000	200	$4k^{0.1}$	$0.8n + \frac{\log(k+2)\lVert y_k\rVert}{100} + \hat{\rho}_k$	15	0.1	$\min\left\{\frac{15((k+1)^q)}{\lVert c(x^{k+1})\rVert}, 10^5\right\}$
2000	1500	$4k^{0.1}$	$0.9n + \frac{\log(k+2)\lVert y_k\rVert}{100} + \hat{\rho}_k$	20	0.1	$\min\left\{\frac{20((k+1)^q)}{\lVert c(x^{k+1})\rVert}, 10^5\right\}$
3000	1800	$2k^{0.1}$	$2800 + \frac{\log(k+2)\lVert y_k\rVert}{100} + 2\hat{\rho}_k$	25	0.1	$\min\left\{\frac{25((k+1)^q)}{\lVert c(x^{k+1})\rVert}, 10^5\right\}$
4000	2400	$2k^{0.1}$	$3500 + \frac{\log(k+2)\lVert y_k\rVert}{100} + 2\hat{\rho}_k$	25	0.1	$\min\left\{\frac{25((k+1)^q)}{\lVert c(x^{k+1})\rVert}, 10^5\right\}$
5000	3000	$2k^{0.1}$	$4500 + \frac{\log(k+2)\lVert y_k\rVert}{10} + 2\hat{\rho}_k$	30	0.1	$\min\left\{\frac{30((k+1)^q)}{\lVert c(x^{k+1})\rVert}, 10^5\right\}$
6000	5100	$2k^{0.1}$	$5500 + \frac{\log(k+2)\lVert y_k\rVert}{10} + 2\hat{\rho}_k$	40	0.1	$\min\left\{\frac{40((k+1)^q)}{\lVert c(x^{k+1})\rVert}, 10^5\right\}$
7000	5900	$2k^{0.1}$	$6000 + \frac{\log(k+2)\lVert y_k\rVert}{10} + 2\hat{\rho}_k$	40	0.1	$\min\left\{\frac{40((k+1)^q)}{\lVert c(x^{k+1})\rVert}, 10^5\right\}$
8000	6100	$2k^{0.1}$	$6200 + \frac{\log(k+2)\lVert y_k\rVert}{10} + 2\hat{\rho}_k$	40	0.1	$\min\left\{\frac{40((k+1)^q)}{\lVert c(x^{k+1})\rVert}, 10^5\right\}$
8000	6700	$2k^{0.1}$	$7200 + \frac{\log(k+2)\lVert y_k\rVert}{10} + 2\hat{\rho}_k$	45	0.1	$\min\left\{\frac{45((k+1)^q)}{\lVert c(x^{k+1})\rVert}, 10^5\right\}$
10000	9000	$2k^{0.1}$	$8500 + \frac{\log(k+2)\lVert y_k\rVert}{10} + 2\hat{\rho}_k$	45	0.1	$\min\left\{\frac{45((k+1)^q)}{\lVert c(x^{k+1})\rVert}, 10^5\right\}$

Table 4 compares these results against those collected by executing the Augmented Lagrangian method as specified by the ALGENCAN framework (using `nlopt`) with Limited-memory BFGS (L-BFGS) set as primal subproblem solver. The following setting is considered: the size of the storage window for the L-BFGS subsolver is set to 20 and the absolute tolerance on the objective function value is set to 1e-16. In this case, the absolute tolerance on the objective function value (set to 1e-16) is used as the stopping criterion. Feasibility, Relative gap to optimal value and Computation time are measured once this criterion is reached as reported in Columns 5, 6 and 7 of Table 4.

From Table 2, we can observe that for low dimension matrices (with $d = 1000, 2000$ and 3000), our algorithm remains competitive against RI-ALM. For larger values of d (i.e., from 4000 to 8000), the RI-ALM algorithm is up to $1.5\times$ faster when input data are customized so as to fit all required conditions of smoothness and weak convexity, including regularity conditions. Note also a factor 100 in favor of our method for what concerns feasibility. However, above a certain matrix dimension ($d > 8000$), our algorithm becomes again more computationally efficient. That is, beyond this threshold, the RI-ALM algorithm requires progressively less inner iterations (the number of outer iterations is nearly equal for $d = 9000$ and $d = 10000$) but each iteration takes more computation time. This behavior seems to be induced by the regularity condition (in addition to the well-controlled dual step size) that RI-ALM imposes to achieve best-known convergence rates.

Compared to ALGENCAN, our Algorithm 1 remains competitive in computation time. For all instances, the computation time is divided by a factor 2 (except for $d = 5000$) up to a factor 5 ($d = 2000$). ALGENCAN shows indeed higher computation time, independently of the size $d \times d$ of the matrices. Its computation time doubles as the size of the matrices increases by steps of 1000 from $d = 3000$ to $d = 7000$ and saturates for larger values of d (i.e., $d = 8000, 9000, 10000$).

5.1.2 Max-Cut Problem

Consider an undirected (edge-)weighted graph $G = (V, E, \omega)$ with vertex set V, edge set E and weight function ω that assigns to each edge $(i, j) \in E$ the weight ω_{ij} such that the matrix $\Omega = (\omega_{ij})$. The max-cut problem consists of partitioning the vertex set V into two disjoint complementary subsets to maximize the number of edges crossing the cut (unweighted case) or the sum of the weights of the edges crossing the cut (weighted case). The problem of finding a maximum cut is an NP-Hard combinatorial optimization problem [28]. The best known approximation to the max-cut problem is defined by its relaxation to a semidefinite program [29]. The optimal solution value of the SDP gives an upper bound of the weights of the max-cut.

Let X be a real symmetric matrix ($X = xx^\intercal$) and e the 1-vector $(1, 1, \dots, 1)^\intercal \in \mathbb{R}^m$, the SDP relaxation of the primal max-cut problem can be formulated as

$$\begin{aligned} &\text{maximize } \frac{1}{4}\langle L \mid X\rangle \\ &\text{subject to } \operatorname{diag}(X) = e, \quad X \succeq 0. \end{aligned} \tag{5.3}$$

The shorthand $X \succeq 0$ indicates that the matrix X is positive semidefinite. The operation $\langle L \mid X\rangle = Tr(L^\intercal X)$ represents the Frobenius (component-wise) inner product of the real symmetric matrices L and X of the same dimension. The matrix L denotes the graph Laplacian $L = (D - \Omega)$ where Ω is a symmetric weight matrix, such that individual edge weights $\omega_{ij} \in \mathbb{R}_0, \omega_{ij} = \omega_{ji}$ for all $(i, j) \in E,\ i, j \in \{1, \dots, m\}$, with $\omega_{ii} = 0$ for $i = 1, \dots, m$, and $D = Diag(d_1, \dots, d_m)$ is the degree (diagonal) matrix of the graph G with $d_i = \sum_{i=1}^m \omega_{ij}$. The Laplacian $L = D - \Omega$ is symmetric and positive semidefinite and can also be interpreted as the linear map from $\mathbb{R}^{\mathcal{V}}$ to itself, i.e., $(Lx)_i = \sum_{j:(i,j)\in E} \omega_{ij}(x_i - x_j),\ \forall x \in \mathbb{R}^V$. Since $x_i = \pm 1 \wedge x_i^2 = 1,\ \forall i$, the objective function $\frac{1}{4}\sum_i \sum_j w_{ij}(1 - x_i x_j)$ can be formulated as $\frac{1}{4}(\langle \Omega \mid J\rangle - \langle \Omega \mid X\rangle)$, where J is the matrix where every element is equal to one. The SDP relaxation thus aims to find a convex set $S := \{xx^\intercal : x \in \{\pm 1\}^m\}$, which contains all the rank-1 matrices X such that $X \in \mathcal{S}$. Note also that if the optimal solution X of the primal max-cut SDP has rank 1, then we may write matrix X in the form $X = xx^\intercal$ and therefore recover the optimal cut; in this case, the SDP relaxation is exact. Since any matrix in $\mathcal{S}$ is positive semidefinite ($X \succeq 0$) and the diagonal entries are equal to 1 ($X_{ii} = 1$), solving the primal max-cut SDP problem (5.3) is equivalent to

$$\begin{aligned} &-\text{minimize } \frac{1}{4}\langle \Omega \mid X\rangle \\ &\text{subject to } \operatorname{diag}(X) = e, \quad X \succeq 0. \end{aligned} \tag{5.4}$$

Therefore, the SDP relaxation (5.4) can be seen as a particular case of Problem (1.1) with i) $h : x \mapsto \frac{1}{4}\langle \Omega \mid X\rangle$; ii) $c_1 : x \mapsto \operatorname{diag}(X)$ and iii) $b_1 = e \in \mathbb{R}^m$. The SDP optimal value is then given by the sum of the weight matrix elements $\langle \Omega \mid J\rangle$ minus $Tr(\Omega X)$ divided by 4.

Table 5 Max-cut problem: numerical results for data sets 1) G1, 2) G30, 3) GD97b, 4) GD97c, 5) LFAT5t, 6) Sbbrail, 7) Shermann1, 8) Trefethen20b, 9) Trefethen200b, 10) Trefethen500

	Dim (n)	**Size** (Bytes)	$\eta_{\min}(\mathbf{C})$	$\eta_{\max}(\mathbf{C})$	**Computed value**	**r**	**r***	**Number of iterations**	**Computation Time** (sec)
1	800	70728	3.48e-17	17.738	12083.17247	10	10	2498	5.6367
2	2000	227719	1.2e-16	9.2	8382.54360	10	10	8702	560.6142
3	47	4142	0	1549.3	15340.10553	5	5	4082	0.9437
4	452	4039	0	85.2	384.00125	5	5	2657	1.1022
5	14	999	8.68e-12	4.71e6	189.88141	5	5	306170	69.0450
6	734	7775	2.66e-19	2.5842	834.04961	5	5	2882	0.8367
7	1000	10780	0	1.2441	279.53284	5	5	4584	1.5695
8	19	1336	1.51e-16	6.1897	48.66725	5	4	498	0.1242
9	199	5825	1.094e-16	5.2072	1006.60390	5	5	870	0.2164
10	500	15699	1.26e-16	2.6727	3014.46360	5	5	1749	0.6422

We execute Algorithm 1 on the following data sets: *G1*, *G30*, *GD97b*, *GD97c*, *LFAT5t*, *Sbbrail*, *Shermann1*, *Trefethen20b*, *Trefethen200b*, and *Trefethen500*. These datasets are openly available and accessible at https://networkrepository.com/networks.php. We consider the stop condition given by the relative gap (5.2) $< 10^{-5}$, where $h^\star$ corresponds to the value of the objective function as computed by the `cvx-Matlab` solver. This value is referred to as the optimal in the context of this study.

The results are presented in Table 5 listing the best computed value (in Column 6), the target rank r (in Column 7), the rank of the approximation solution $r^\star$ (in Column 8), the number of iterations (in Column 9) and the running time in seconds (in Column 10) as well as the smallest and the largest singular value of the set C denoted respectively by $\eta_{\min}(C)$ and $\eta_{\max}(C)$ (in Columns 3 and 4, respectively).

We then compare the results obtained for Algorithm 1 against the following methods: RI-ALM, and ALGENCAN. For the ALGENCAN method, we develop the model in C and execute it by tuning the AUGLAG algorithm provided in `nlopt`, a free/open-source library for nonlinear optimization. Limited-memory BFGS (L-BFGS) is set as the primal subproblem solver. The following setting is considered: the size of the storage window for the L-BFGS subsolver is set to 10 and the absolute tolerance on the objective function value is set to 1e-06. Concerning the RI-ALM method [61], we implement the main loop as well as the iPP and the APG subsolvers in `Octave 7.2`. The following setting is considered: the parameter $\varepsilon = 0.001$, $\sigma = 1.5$, $\beta_0 = 10$. The initial values for primal and dual variables are $x_0 = \mathrm{randn}(n, r)$ where r is the rank, and $y_0 = 0$.

For this purpose, we execute the RI-ALM and ALGENCAN algorithms on the following data sets: *G1*, *G30*, *GD97b*, *GD97c*, *LFAT5t*, *Sbbrail*, *Shermann1*, *Trefethen20b*, *Trefethen200b*, and *Trefethen500*. The results are presented in Table 6. Each algorithm is executed 20 times, and we take the average of the results obtained for the computation time (Column 5), the number of iterations (Column 4[4] and the computed value (Column 3). The second column of Table 6 gives the value of the objective function obtained by means of the Goemans-Williamson convex relaxation for the max-cut problem. This method, which provides an upper bound of the max-cut problem with the optimal value obtained by SDP, has been executed on the `cvx-Matlab` solver.

As it can be observed from Table 6, for every data set, the Algorithm 1 performs much better than RI-ALM in terms of computational time required to satisfy the stop condition set to $< 10^{-5}$. Among them only the third one meets the ε approximation threshold (set to 0.001) within 1000 iterations of the iPP method (while the number of inner APG loop ranges in the order of 1). For all data sets, the computation time significantly improves by at least a factor 10 except for the datasets *G30* and *GD97c*, for which the gain is more limited though still substantial for the latter (about a factor 2). These results emphasize thus the computational limitation of inexact ALM methods relying on triple embedded loops. Observe also that for the dataset *G30*, the computation time required by RI-ALM is about 2 third of the one needed by Algorithm 1. This result indicates that RI-ALM starts also to show

[4] where (o) indicates the number of outer iterations

Table 6 Max-cut problem: Comparison of Algorithm 1 against Rate-improved ALM (RI-ALM) and ALGENCAN

Algorithm	SDP Optimal Value (cvx)	Computed Value	Number of Iterations	Computation Time (s)
G1				
Our method	12083.19789	12083.17247	2498	5.6367
RI-ALM		12064.36612	5000	62.6936
ALGENCAN		10186.99315	149 (o)	2757.0824
G30				
Our method	8380.76452	8382.54360	8702	560.6142
RI-ALM		8189.02740	21214	844.2031
ALGENCAN		—	—	—
GD97b				
Our method	15340.05392	15340.10553	4082	0.9437
RI-ALM		15339.93391	36000	22.6120
ALGENCAN		15144.91984	91 (o)	0.3250
GD97c				
Our method	383.99999	384.0012	2657	1.1022
RI-ALM		384.0000	4512	2.3750
ALGENCAN		301.6125	14 (o)	45.6560
LFAT5t				
Our method	189.88246	189.88141	306170	69.4050
RI-ALM		189.38262	219394	9325.4687
ALGENCAN		—	—	—
Sbbrail				
Our method	834.05469	834.04961	2882	0.8367
RI-ALM		834.02713	86000	720.6418
ALGENCAN		582.82795	453 (o)	4905.9364
Sherman1				
Our method	279.53561	279.53284	4584	1.5695
RI-ALM		279.88156	54000	633.7705
ALGENCAN		272.17087	188 (o)	5452.4516
Trefethen20b				
Our method	48.66760	48.66725	498	0.1242
RI-ALM		48.66796	7179	3.5344
ALGENCAN		43.39074	41 (o)	0.1840
Trefethen200b				
Our method	1006.60980	1006.60390	870	0.2164
RI-ALM		1006.61334	21000	17.2612
ALGENCAN		914.41553	74 (o)	18.4548
Trefethen500				
Our method	3014.49374	3014.46360	1749	0.6422
RI-ALM		3014.51402	26000	36.1249
ALGENCAN		2399.19682	101 (o)	396.9699

some limits as the size of the problem increases since *G30* is the largest max-cut instance considered in our experiments. Note also from Table 6, that the values computed by ALGENCAN are quite far from the optimal; hence, performance of ALGENCAN are not comparable to Algorithm 1.

6 Use Case

In this section, we evaluate the performance of the proposed algorithm for the solving of multi-constrained network design problem, namely, the multi-commodity network design problem with load-induced delay constraints. The problem consists of finding the minimum capacity to be provisioned on each arc of the network topology such that the network can serve all demands simultaneously, i.e., all incoming demands can be routed simultaneously from their source to their destination. Compared to the usual multi-commodity network design problem, the problem considered in this paper assumes that each demand is specified together with a maximum delay that can be incurred on the traffic traversing the network (a.k.a. load-induced delay). The resulting constraints yield a problem involving nonlinear (equality and inequality) constraints. The numerical experiments performed by means of the proposed algorithm are then analyzed and compared against the convex relaxation method.

We are given a directed graph $G = (V, A)$ where V is the set of nodes ($|V| = n$) and A the set of possible arcs ($|A| = m$). Each arc is denoted by (i, j) where i is the head end and j the tail end of the arc. Each arc $(i, j) \in A$ from node i to node j provides a nominal maximum capacity $\kappa_{ij} > 0$. The cost per unit of capacity on arc $(i, j) \in A$ is given by the real number $\alpha_{ij}/\kappa_{ij} = \varsigma_{ij} \geq 0$, where α_{ij} is the installation cost for arc (i, j).

6.1 Aggregated demand matrix

This aggregated formulation assumes that the $n \times n$ fixed demand matrix D where $D(s, t)$ is the total amount of traffic sent from source s to destination t, for any pair of nodes $s, t \in V,\ s \neq t$, is given. Moreover, each (s, t) pair is specified with a time delay upper bound τ^{st} that represents the maximum amount of time that can elapse for traffic to flow from source s to destination t.

Let $\mathcal{D} \subset \mathbb{R}_+^{n\times n}$ be a bounded set, in the static case, the demand matrix $D \in \mathcal{D}$ is certain (not variable). In this setting, routing is defined as a function $f : \mathcal{D} \to \mathbb{R}^{m\times(n\times n)}$ that assigns an (s, t)-aggregated flow to the given realization of the demand matrix $D \in \mathcal{D}$. A capacity allocation $x \in \mathbb{R}_+^m$ is said to support the demand D if there exists a routing f serving D such that for every demand $D(s, t)$, the corresponding flow matrix $f^{st} \in \mathbb{R}^{n\times n}$ does not exceed the arc capacities given by x.

6.1.1 Variables

The following variables are defined.

- Continuous capacity allocation variables $x_{ij} \geq 0$ that represent the amount of capacity installed in the arc (i, j).
- Continuous flow variables $f_{ij}^{st} \geq 0$ that represent the amount of flow on arc (i, j) from source s to destination t. With this definition of the flow variables, the load on arc (i, j) is defined as $\sum_{s,t\in V} f_{ij}^{st}$ and the flow conservation and demand satisfaction constraints formulate as

$$\sum_{j:(i,j)\in A} f_{ij}^{st} - \sum_{j:(j,i)\in A} f_{ji}^{st} = \begin{cases} D(s,t) & \text{if } i = s \\ -D(s,t) & \text{if } i = t \\ 0 & \text{otherwise} \end{cases} \qquad \forall i, s, t \in V. \tag{6.1}$$

6.1.2 Formulation

For static routing, the initial formulation of the model involves the continuous variables defined in Section 6.1.1. The cost function $\varPhi$ is assumed (piecewise linear) convex in variable x_{ij}. Accounting for demand satisfaction and flow conservation (6.3), capacity allocation (6.4) and delay (6.6) constraints, yields the following formulation:

$$\min \sum_{(i,j)\in A} \Phi(x_{ij}) \tag{6.2}$$

s.t.

$$\sum_{j:(i,j)\in A} f_{ij}^{st} - \sum_{j:(j,i)\in A} f_{ji}^{st} = \begin{cases} D(s,t) & \text{if } i = s \\ -D(s,t) & \text{if } i = t \\ 0 & \text{otherwise} \end{cases} \qquad \forall i,s,t \in V \tag{6.3}$$

$$\sum_{s,t\in V} f_{ij}^{st} \le x_{ij} \qquad \forall (i,j) \in A \tag{6.4}$$

$$x_{ij} \le \kappa_{ij} \qquad \forall (i,j) \in A \tag{6.5}$$

$$\sum_{(i,j)\in A} \delta(f_{ij}^{st},(i,j)) \frac{1}{\kappa_{ij} - x_{ij}} \le \tau^{st} \qquad \forall s,t \in V \tag{6.6}$$

$$f_{ij}^{st} \ge 0 \qquad \forall (i,j) \in A, s,t \in V \tag{6.7}$$

$$x_{ij} \ge 0 \qquad \forall (i,j) \in A \tag{6.8}$$

The constraints (6.3) enforce flow conservation that ensures that the demand flow requirements given by the matrix elements $D(s,t)$ are appropriately routed for each (s,t) pair. Note that the flow conservation constraints are disaggregated per source. To account for the source-to-destination delay, one considers the sum over the average delay for passing through each intermediate node (i.e., channel) along the path. The capacity constraints (6.4) impose that the sum of the flows f_{ij}^{st} along each arc $(i,j) \in A$ does not exceed the capacity that would be allocated to this arc. Observing that the Little's Formula is valid for the steady state of any queueing process, the waiting time (delay) per channel/arc a modeled as a M/M/1 queuing system can be computed by applying the Little's formula: $1/(\kappa_{ij} - x_{ij})$. This expression can be generalized to M/G/1 queuing systems by means of the Pollaczek-Khinchin mean formula for the waiting time [48] [35]. The total delay to send a traffic flow unit from a source s to a destination t is then given by the sum of the load-induced delay incurred on the individual arcs from s to t. Consequently, the nonlinear delay constraints (6.6) are fractional in the capacity variables x_{ij}. They involve the binary indicator variable $\delta(f_{ij}^{st},(i,j)) = 1$ which equals 1 if the flow f_{ij}^{st} from source s to destination t along arc (i,j) is strictly greater than zero and 0 otherwise.

6.1.3 Reformulation

We reformulate Problem (6.2)-(6.8) as an instance of the nonlinear composition problem. For this purpose, we apply the following variable transformation

$$(\forall (i,j) \in A) \; y_{ij} = \frac{\kappa_{ij}}{\kappa_{ij} - x_{ij}}, \tag{6.9}$$

Then, the capacity allocation variable x_{ij} becomes

$$(\forall (i,j) \in A) \; x_{ij} = \kappa_{ij} - \frac{\kappa_{ij}}{y_{ij}}. \tag{6.10}$$

In turn, the original nonlinearly constrained minimization problem (6.2)-(6.8) can be reformulated as follows.

$$(CFP)\ \min \sum_{(i,j)\in A} \Phi\big(\kappa_{ij}(1-\frac{1}{y_{ij}})\big) \tag{6.11}$$

s.t.

$$\sum_{j:(i,j)\in A} f_{ij}^{st} - \sum_{j:(j,i)\in A} f_{ji}^{st} = \begin{cases} D(s,t) & \text{if } i=s \\ -D(s,t) & \text{if } i=t \\ 0 \text{ otherwise} \end{cases} \qquad \forall i,s,t\in V \tag{6.12}$$

$$\kappa_{ij} \le y_{ij}\big(\kappa_{ij} - \sum_{s,t\in V} f_{ij}^{st}\big) \qquad \forall (i,j)\in A \tag{6.13}$$

$$\sum_{(i,j)\in A} \delta(f_{ij}^{st},(i,j))\frac{y_{ij}}{\kappa_{ij}} \le \tau^{st} \qquad \forall s,t\in V \tag{6.14}$$

$$f_{ij}^{st} \ge 0 \qquad \forall (i,j)\in A, s,t\in V \tag{6.15}$$

$$y_{ij} \ge 1 \qquad \forall (i,j)\in A \tag{6.16}$$

Observe that the objective function of this model, referred to as (CFP), is now concave, which, in turn, requires approximation by piecewise linear functions for its solving. In the (CFP) formulation, the flow variables can be reformulated as

$$f_{ij}^{st} = z_{ij}^{st} D(s,t) \tag{6.17}$$

together with constraints of the form $0 \le z_{ij}^{st} \le z_{max}$ $(\forall (i,j)\in A,\ s,t\in V)$ and $z_{ij}^{st} \ge 0$ $(\forall (i,j)\in A,\ s,t\in V)$. Note that when $z_{max}=1$, one refers to fractional flow variables. The flow conservation constraints then become

$$\sum_{j:(i,j)\in A} z_{ij}^{st} - \sum_{j:(j,i)\in A} z_{ji}^{st} = \begin{cases} 1 & \text{if } i=s \\ -1 & \text{if } i=t \\ 0 \text{ otherwise} \end{cases} \qquad \forall i,s,t\in V \tag{6.18}$$

The combination of (6.18) together with (6.17) yields variables z_{ij}^{st} that define a source-destination percentage flow. In this case, the variable z^{st} determines for every demand from source s to destination t, the paths that are used to route the demand and what is the percentage splitting among these paths. This setting defines the static routing case since the flow for demand from source s to destination t can only change linearly with $D(s,t)$ on the paths described by the variables z^{st}. However, the restriction of the (CFP) to the static formulation doesn't facilitate the handling of the constraints (6.14).

The main alternative consists of defining the flow variables f_{ij}^{st} as $f_{ij}^{st} = b_{ij}^{st} D(s,t)$ $(\forall (i,j)\in A, s,t\in V)$ together with the binary variables $b_{ij}^{st}\in\{0,1\}$ $(\forall (i,j)\in A, s,t\in V)$. Consequently, one obtains the formulation referred to as (BFP):

$$(BFP)\ \min \sum_{(i,j)\in A} \Phi\big(\kappa_{ij}(1-\frac{1}{y_{ij}})\big) \tag{6.19}$$

s.t.

$$\sum_{j:(i,j)\in A} b_{ij}^{st} - \sum_{j:(j,i)\in A} b_{ji}^{st} = \begin{cases} 1 & \text{if } i=s \\ -1 & \text{if } i=t \\ 0 \text{ otherwise} \end{cases} \qquad \forall i,s,t\in V \tag{6.20}$$

$$\kappa_{ij} \le y_{ij}\big(\kappa_{ij} - \sum_{s,t\in V} D(s,t) b_{ij}^{st}\big) \qquad \forall (i,j)\in A \tag{6.21}$$

$$\sum_{(i,j)\in A} b_{ij}^{st}\frac{y_{ij}}{\kappa_{ij}} \le \tau^{st} \qquad \forall s,t\in V \tag{6.22}$$

$$b_{ij}^{st}\in\{0,1\} \qquad \forall (i,j)\in A, s,t\in V \tag{6.23}$$

$$y_{ij} \ge 1 \qquad \forall (i,j)\in A \tag{6.24}$$

With binary flow variables, traffic flow units are routed along a single path, yielding the unsplittable flow problem (UFP), which is NP-hard even without delay constraints. Also, since there now exists a single path from source s to destination t for every demand $D(s,t)$, static and dynamic routing coincide for all demands. Note that the UFP is modeled under the No-Bottleneck Assumption (NBA) since the largest demand is upper-bounded by the smallest arc capacity, i.e., $\max_{(s,t)} D(s,t) \leq \min_{(i,j)\in A} \kappa_{ij}$. The main advantage of this formulation stems from the expression of the constraints (6.22) as a sum of products of binary variables, which can be linearized through standard techniques [27].

6.2 Per-commodity flow

This formulation assumes a set of commodities K, where each commodity $k \in K$ is defined by its sources $s_k \in V$, destination $t_k \in V$, and demand size value $d_k \geq 0$. Let $\mathcal{D} \subset \mathbb{R}_+^{|K|}$ be a bounded set, in the static case, the demand vector $d = \{d_1, \ldots, d_{|K|}\} \in \mathcal{D}$ is certain (not variable).

Moreover, each commodity flow is specified with a time delay upper bound τ^k that represents the maximum amount that can elapse for traffic to flow from source s to destination t. In this case, a routing is a function $f : \mathcal{D} \to \mathbb{R}^{|A|\times|K|}$ that assigns a multi-commodity flow to the given realization of the demand vector $d \in \mathcal{D}$.

A capacity allocation $x \in \mathbb{R}_+^{|A|}$ is said to support the demand vector $d = \{d_1, \ldots, d_{|K|}\}$ if there exists a routing f serving d such that for every commodity $k \in K$ of demand size value d_k, the corresponding flow vector $f^k \in \mathbb{R}^{|A|}$ does not exceed the arc capacities given by x.

6.2.1 Variables

The following variables are defined.

- Continuous capacity allocation variables $x_{ij} \geq 0$ represent the amount of capacity installed on arc $(i,j) \in A$
- Continuous flow variables $f_{ij}^k \geq 0$ represent the amount of flow on arc (i,j) from source s to destination t for commodity k. The load on arc (i,j) is defined as $\sum_{k\in K} f_{ij}^k$.

In this case, the flow variables satisfy the flow conservation and demand satisfaction constraints.

$$\sum_{j:(i,j)\in A} f_{ij}^k - \sum_{j:(j,i)\in A} f_{ji}^k = \begin{cases} d_k & \text{if } i = s_k \\ -d_k & \text{if } i = t_k \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in V, k \in K \tag{6.25}$$

6.2.2 Formulation

Assuming per-commodity flow, the problem can be formulated as follows

$$\min \sum_{(i,j)\in A} \Phi(x_{ij}) \tag{6.26}$$

s.t.

$$\sum_{j:(i,j)\in A} f_{ij}^k - \sum_{j:(j,i)\in A} f_{ji}^k = \begin{cases} d_k & \text{if } i = s_k \\ -d_k & \text{if } i = t_k \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in V, k \in K \tag{6.27}$$

$$\sum_{k\in K} f_{ij}^k \le x_{ij} \qquad \forall (i,j) \in A \tag{6.28}$$

$$x_{ij} \le \kappa_{ij} \qquad \forall (i,j) \in A \tag{6.29}$$

$$\sum_{(i,j)\in A} \delta(f_{ij}^k, (i,j)) \frac{1}{\kappa_{ij} - x_{ij}} \le \tau^k \qquad \forall k \in K \tag{6.30}$$

$$f_{ij}^k \ge 0 \qquad \forall (i,j) \in A, k \in K \tag{6.31}$$

$$x_{ij} \ge 0 \qquad \forall (i,j) \in A \tag{6.32}$$

In this formulation, $\delta(f_{ij}^k, (i,j)) = 1$ if the value of the flow variable f_{ij}^k for commodity k along the arc (i,j) is strictly greater than zero ($f_{ij}^k > 0$) and 0 otherwise.

Applying the variable transformation (6.9), the original nonconvex minimization problem can be reformulated as follows.

$$(CFP) \min \sum_{(i,j)\in A} \Phi\big(\kappa_{ij}(1 - \frac{1}{y_{ij}})\big) \tag{6.33}$$

s.t.

$$\sum_{j:(i,j)\in A} f_{ij}^k - \sum_{j:(j,i)\in A} f_{ji}^k = \begin{cases} d_k & \text{if } i = s_k \\ -d_k & \text{if } i = t_k \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in V, k \in K \tag{6.34}$$

$$\kappa_{ij} \le y_{ij}\big(\kappa_{ij} - \sum_{k\in K} f_{ij}^k\big) \qquad \forall (i,j) \in A \tag{6.35}$$

$$\sum_{(i,j)\in A} \delta(f_{ij}^k, (i,j)) \frac{y_{ij}}{\kappa_{ij}} \le \tau^k \qquad \forall k \in K \tag{6.36}$$

$$f_{ij}^k \ge 0 \qquad \forall (i,j) \in A, k \in K \tag{6.37}$$

$$y_{ij} \ge 1 \qquad \forall (i,j) \in A \tag{6.38}$$

Here, also, one can define the flow variables as $f_{ij}^k = b_{ij}^k d_k$ $(\forall (i,j) \in A, k \in K)$ together with the binary variables $b_{ij}^k \in \{0,1\}$ $(\forall (i,j) \in A, k \in K)$. With this definition of the flow variables, one obtains the (BFP) formulation:

$$(BFP)\min \sum_{(i,j)\in A} \Phi\big(\kappa_{ij}(1-\frac{1}{y_{ij}})\big) \tag{6.39}$$

s.t.

$$\sum_{j:(i,j)\in A} b_{ij}^k - \sum_{j:(j,i)\in A} b_{ji}^k = \begin{cases} 1 & \text{if } i = s_k \\ -1 & \text{if } i = t_k \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in V, k \in K \tag{6.40}$$

$$\kappa_{ij} \le y_{ij}\big(\kappa_{ij} - \sum_{k\in K} d_k b_{ij}^k\big) \qquad \forall (i,j) \in A \tag{6.41}$$

$$\sum_{(i,j)\in A} b_{ij}^k \frac{y_{ij}}{\kappa_{ij}} \le \tau^k \qquad \forall k \in K \tag{6.42}$$

$$b_{ij}^k \in \{0,1\} \qquad \forall (i,j) \in A, k \in K \tag{6.43}$$

$$y_{ij} \ge 1 \qquad \forall (i,j) \in A \tag{6.44}$$

In the remainder, the transport cost function Φ is defined as a quadratic function of the load $x_{ij} = \sum_{k\in K} f_{ij}^k$ of the arc $(i,j) \in A$. More precisely, we set $\Phi(x_{ij}) = x_{ij}^2$. With this setting, the objective function becomes

$$\sum_{(i,j)\in A} \frac{\alpha_{ij}}{\kappa_{ij}^2} x_{ij}^2 = \sum_{(i,j)\in A} \frac{\alpha_{ij}}{\kappa_{ij}^2} \big(\sum_{k\in K} f_{ij}^k\big)^2, \tag{6.45}$$

where α_{ij} is the cost of the arc $(i,j) \in A$, κ_{ij} the capacity of the arc $(i,j) \in A$ and $f_{ij}^k \ge 0$. Similarly, when flows are defined as binary variables, i.e. $b_{ij}^k \in \{0,1\}$ the objective function reads as

$$\sum_{(i,j)\in A} \frac{\alpha_{ij}}{\kappa_{ij}^2} x_{ij}^2 = \sum_{(i,j)\in A} \frac{\alpha_{ij}}{\kappa_{ij}^2} \big(\sum_{k\in K} d_k b_{ij}^k\big)^2. \tag{6.46}$$

6.3 Numerical Experiments, Results and Analysis

6.3.1 Data Sets and Network Instances

To evaluate the formulations of the (CFP) and (BFP) models presented in Section 6.2.2, we consider a set of network topologies extracted from the SNDlib topology library [44]. From this database, the following topologies have been extracted (in alphabetical order): *austria*, *atlanta*, *cost266*, *france*, *germany50*, *india35*, *norway*, *pioro40*, *zib54*. All datasets are openly available and accessible at http://sndlib.zib.de/problems.overview.action. Their main properties are summarized in Table 7. The arc cost α_{ij} as well as the arc capacity κ_{ij} are provided as part of the dataset.

For each topology, a set K of n demands is generated, where each element $k \in K$ is a tuple that comprises a source s, a destination t, a positive size value d^{st} and a delay upper bound τ^{st}. Note that demand size and delay bounds are generated such that a feasible solution exists; for instance, the size of individual demands never exceeds the nominal arc capacity.

6.3.2 McCormick Envelope - PWL method

Reformulation Linearization Technique (RLT) relaxations [52] [53] can be used to obtain tight yet solvable convex relaxations of problems with quadratic terms that linearize constraints (6.13). This technique essentially consists of two main steps: a reformulation step in which certain additional nonlinear valid inequalities are (automatically) generated, and a linearization step in which each product term is replaced by a single continuous variable. In general, variables i are restricted to lie in

Table 7 Network Topologies and Properties

Topology	Nodes	Arcs	Min, Max, Avg Degree	Diameter	Demands
austria	65	216	1, 10, 3.32	9	934
atlanta	15	44	2, 4, 2.93	5	210
cost266	37	114	2, 5 3.08	8	1332
france	25	90	2, 10, 3.60	5	300
germany50	50	176	2, 5, 3.52	9	662
giul39	39	172	3, 8, 4.41	6	1471
india35	35	160	2, 9, 4.57	7	595
norway	27	102	2, 6, 3.78	7	702
pioro40	40	178	4, 5, 4.45	7	780
zib54	54	81	1, 10, 3.00	8	1501

the interval $[L_i, U_i]$, resulting in the so-called bound factors, that is, nonnegative expressions of the form $(x_i - L_i)$ and $(U_i - x_i)$.

Observe that in both the original and the reformulated problem, variables f_{ij}^{st} are naturally lower bounded by 0 and upper bounded by the minimum between the capacity κ_{ij} of arc (i,j) and the demand between (s,t); thus, $0 \leq f_{ij}^{st} \leq min\{\kappa_{ij}, d_{st}\}$. Instead, variables y_{ij} are lower bounded by $y_L = 1$ but require the setting of an arbitrary upper bound (since $y_{ij} \to \infty$ when $\sum_{s,t\in V} f_{ij}^{st} \to \kappa_{ij}$). In practice, due to the convex shape of the objective function, this upper bound can be selected in the range $[2/3\kappa_{ij}, \kappa_{ij}[$. Observe that when $y_{ij} \to \kappa_{ij}$, constraints (6.13) become $1 + \sum_{s,t\in V} f_{ij}^{st} \leq \kappa_{ij}, \forall (i,j) \in A$.

Note that (iteratively) decreasing this upper bound closer to its minimum produces tighter relaxations. Observe also that the product $f_{ij}^{st} y_{ij} \geq 0, \forall (s,t) \in K, (i,j) \in A$. Moreover, we further restrict the variables f_{ij}^{st} to be fractional (and multiply them by the demand size d_{st} in constraints (6.12)) in order to obtain constraints of the form $0 \leq f_{ij}^{st} \leq 1$.

– Reformulation step: since variables f_{ij}^{st} and y_{ij} are lower and upper bounded, that is, $0 \leq f_{ij}^{st} \leq 1$ and $1 \leq y_{ij} \leq y_U$, the following implied inequality constraints can be generated as product of (first-order) bound factors:

$$\begin{cases} 0 \leq (f_{ij}^{st} - 0)(y_{ij} - 1) & \Rightarrow f_{ij}^{st} y_{ij} \geq f_{ij}^{st} \\ 0 \leq (1 - f_{ij}^{st})(y_U - y_{ij}) & \Rightarrow f_{ij}^{st} y_{ij} \geq y_U f_{ij}^{st} + y_{ij} - y_U \\ 0 \leq (f_{ij}^{st} - 0)(y_U - y_{ij}) & \Rightarrow f_{ij}^{st} y_{ij} \leq y_U f_{ij}^{st} \\ 0 \leq (1 - f_{ij}^{st})(y_{ij} - 1) & \Rightarrow f_{ij}^{st} y_{ij} \leq f_{ij}^{st} + y_{ij} - 1 \end{cases} \tag{6.47}$$

The first two inequalities are referred to as the under-estimators of the product $y_{ij} f_{ij}^{st}$ and the last two inequalities as the over-estimators of this product. Note that since variables f_{ij}^{st} are fractional, higher-order polynomial factors could also be considered to strengthen under-estimators.

– Linearization (or convexification) step: the linearization step consists of replacing the bilinear products $y_{ij} f_{ij}^{st}$ appearing in the resulting constraints including (6.13) by the auxiliary variable z_{ij}^{st} together with the following set of constraints (under- and overestimators). This step enables finding the tightest possible convex approximation of the expression $z_{ij}^{st} = y_{ij} f_{ij}^{st}$.

$$\begin{cases} z_{ij}^{st} \geq f_{ij}^{st} \\ z_{ij}^{st} \geq y_U f_{ij}^{st} + y_{ij} - y_U \\ z_{ij}^{st} \leq y_U f_{ij}^{st} \\ z_{ij}^{st} \leq f_{ij}^{st} + y_{ij} - 1 \end{cases} \tag{6.48}$$

In the simplest form considered in this paper, due to the bilinear nature of the terms in constraints (6.13), this type of convex relaxation method is analogous to the McCormick envelopes [40]. The principle behind McCormick envelopes is to replace distinct products of variables, i.e. bilinear terms, by auxiliary variables together with bound constraints which form convex under-estimators (lower envelope) and over-estimators (upper envelope) of the bilinear terms. The essence of the McCormick envelopes technique is very similar to that of the RLT. However, the goal of the former is to replace all bilinear terms with appropriately bounded auxiliary variables whereas the goal of the latter is to produce additional, potentially redundant, constraints in order to minimize the set of candidate solutions to the relaxed problem. Convex envelopes represent the uniformly best convex under-estimators for nonconvex polynomials over some region.

Following the transformation of the capacity variables x_{ij}, the objective function itself becomes nonconvex. The developed technique replaces this nonlinear function by its piecewise linear (PWL) approximating functions. This replacement results in a formulation that can be tackled using state-of-the-art machinery from linear programming. By properly choosing and refining the piecewise linear approximations, this method can provide guarantees on the quality of the solutions computed. A similar method has been developed in [19] although it is limited to linearly constrained optimization problems.

6.3.3 Stochastic ALM with Backtracking: Setup

The following setup is considered for the solving of the static design problem (formulated per commodity $k = (s,t)$) by means of Algorithm 1:

$$\begin{cases} r &= |A| \quad \text{length of A} \\ n &= |K| \\ m &= |V| \\ q &= (i,j) \\ \kappa &= (\kappa_1,\dots,\kappa_r) \\ \varsigma &= (\varsigma_1,\dots,\varsigma_r) \\ d &= [d_1,\dots,d_n] \\ \tau &= [\tau_1,\dots\tau_n] \\ \Delta &= \delta(f^k_{ij},(i,j))_{k\in K,(i,j)\in A} = (\delta_{p,q}). \end{cases} \tag{6.49}$$

The following variables are defined

$$\begin{cases} x_k &= (f^k_{ij})_{(i,j)\in A} = (f^k_1,\dots f^k_r); \\ x &= (x_1,\dots,x_n) \\ y &= (y_{ij})_{(i,j)\in A} = (y_1,y_2,\dots,y_r) \\ u &= (x,y). \end{cases}$$

Define the set C by

$$C = \{u = (x,y) \mid x \geq 0, y \geq 1\}. \tag{6.50}$$

By setting

$$\begin{cases} \tau &= (\tau^{st})_{(s,t)\in V\times V} = (\tau^k)_{k\in K} \\ B_w &= \Big(\frac{\delta(f^{st}_{ij},(i,j))}{\kappa_{ij}}\Big)_{(i,j)\in A;(s,t)\in V\times V} = \Big(\frac{\delta(f^k_{ij},(i,j))}{\kappa_{ij}}\Big)_{(i,j)\in A;k\in K}, \end{cases}$$

the delay constraints can be formulated as

$$B_w y \leq \tau. \tag{6.51}$$

The flow conservation and demand satisfaction constraints (6.3), i.e.,

$$\sum_{j:(i,j)\in A} f^{st}_{ij} - \sum_{j:(j,i)\in A} f^{st}_{ji} = \begin{cases} D(s,t) & \text{if } i = s \\ -D(s,t) & \text{if } i = t, \\ 0 & \text{otherwise,} \end{cases} \tag{6.52}$$

can be rewritten as

$$(\forall k = (s,t) \in V \times V)\ B_0 w(:,k) = b_{0,k},\ b_0 = \ (b_{0,k})_{k \in K}, \tag{6.53}$$

where B_0 corresponds to the (per-) vertex-arc incidence matrix in the standard LP formulation ($\mathbf{Ax} = \mathbf{b}$) of the left hand side of (6.52), the matrix $w = (f_{ij}^k)_{(i,j)\in A;k\in K}$ (thus, the vector $w(:,k)$ denotes all arc-flow variables per commodity k), and $b_{0,k}$ corresponds to the size of the commodity indexed by $k = (s,t)$ in the right hand side of (6.52).

The capacity constraints can be formulated as follows

$$\kappa_{ij} \le y_{ij}\big(\kappa_{ij} - \sum_{k\in K} f_{ij}^k\big) \Longrightarrow y_q(\langle x \mid 1\rangle - \kappa_q) + \kappa_q \le 0. \tag{6.54}$$

Therefore,

$$c_q(x,y) = y_q(\langle x \mid 1\rangle - \kappa_q), \tag{6.55}$$

$$b_q = -\kappa_q, \tag{6.56}$$

$$S_q =]-\infty, 0]. \tag{6.57}$$

6.4 Numerical Results

In this section, we detail the results obtained with the proposed algorithm (see Section 3.1) and compare them against those obtained with the method presented in Section 6.3.2. The purpose being to identify and characterize tradeoffs (computation time vs. gap to (near-)optimality) as well as determine potential direction(s) of improvement.

6.4.1 Numerical Results: motivation

Since constraining the flow assignment to account for the load-induced delay reduces the nominal capacity available per arc, one could think of the following safety margin heuristic. Solve the network design, a.k.a capacity assignment problem, with reduced capacity per arc (e.g., 5, 10 or 20% of the nominal capacity) –without delay constraints. Under these conditions, Table 8 lists the objective function value obtained by solving, down to a relative gap of 1e-08, the capacity assignment problem (with binary flows) using `CPLEX` v12.9. Note that the goal here is to establish the limits of this heuristic in terms of the quality of the solution it produces (not its computational performance).

Table 8 Network design: reduced arc capacity

Topology	**Arc Capacity**					
	60%	80%	90%	95%	99%	100%
austria	542648	542648	542648	542648	542648	542648
atlanta	infeasible	18470139	18446781	18415925	18398292	18398292
cost266	814763	814763	814763	814763	814763	814763
france	infeasible	infeasible	7366	7288	7265	7261
germany50	infeasible	infeasible	132237	132007	131893	131882
giul39	infeasible	infeasible	4560	4524	4505	4501
india35	infeasible	12814	12625	12576	12549	12543
norway	infeasible	386964	386964	386964	386964	386964
pioro40	infeasible	infeasible	infeasible	31040	30862	30820
zib54	8897907	8626820	8580150	8570517	8562851	8560935

6.4.2 Stochastic ALM with Backtracking

The numerical results obtained by executing the Stochastic ALM algorithm with Backtracking are presented in Tables 9 and 10. The parameters used to execute the algorithm for each data set are displayed in Table 11. For each topology, Table 9 lists the initial objective value (Column 2) and feasibility (Column 3), the minimum objective value obtained at the end of the execution of the Stochastic ALM algorithm (Column 4), the feasibility measure obtained (Column 5) and the number of null load arcs, i.e., the number of unused arcs (Column 6). Next, Table 10 details in Column 4 the total time (i.e., the sum of the generation (Column 2) and computation time (Column 3) and number of iterations required to obtain the objective value listed in Column 4 of Table 9.

In Table 10, for the *austria* dataset, we have to tune the parameter ρ_k and increase ρ_k up to 3e+05 to reduce the feasibility (measure) down to 7.6310e-04. Moreover, the feasibility decreases slowly when it reaches 134 while the objective value remains unchanged when the feasibility sits below 134. Therefore, we terminate the execution of the algorithm when the feasibility measure reduces to 0.372 after 14092 iterations.

Table 9 Capacity cost minimization: Stochastic ALM algorithm

Topology	Initial Objective value	Feasibility	Minimum Objective value	Feasibility	Null Load Arcs
austria	7.0668e+05	135240.23	6.6594e+05	7.6310e-04	56
atlanta	3.3206e+07	2087.70	1.8895e+07	7.4047e-04	0
cost266	1.0001e+06	685.34	9.1343e+05	7.3247e-04	0
france	1.3459e+04	650.67	7.2643e+03	8.5600e-04	2
germany50	0.0000e+00	9.81	1.3467e+05	1.0340e-01	7
giul39	2.5867e+04	177.61	5.0210e+03	9.9534e-04	0
india35	4.6828e+03	122.56	1.3080e+04	8.7505e-04	2
norway	2.1717e+05	85.32	4.3771e+05	9.4864e-04	0
pioro40	1.2960e+04	654.76	3.5840e+04	5.4435e-04	3
zib54	6.9166e+04	1821.12	1.3129e+07	3.3126e-03	28

Table 10 Capacity cost minimization: Stochastic ALM algorithm

Topology	Solving Time (s)			Number of Iterations
	Generation	Computation	Total	
austria	19.3163	534615.01	< 5.3464e+05	135964
atlanta	0.2324	58.04	< 5.8273e+01	629
cost266	4.4124	2581.00	< 2.5855e+03	1032
france	1.0579	147.40	< 1.4846e+02	745
germany50	8.4489	11454.10	< 1.1463e+04	9199
giul39	5.7150	7887.41	< 7.8932e+03	1581
india35	3.7575	2553.23	< 2.5570e+03	989
norway	1.7206	542.01	< 5.4374e+02	821
pioro40	5.5732	16473.20	< 1.6479e+04	9491
zib54	12.0360	16827.05	< 1.6840e+04	3206

Table 12 lists the feasibility measure associated to the delay constraints (6.51), the flow conservation and demand satisfaction constraints (6.52), and the capacity constraints (6.54). The reported values

Table 11 Stochastic ALM: parameters and initial variables values

Topology	**Parameters** (s)				**Initial var. values**	
	$\rho_\mathbf{k}$	$\mathbf{t_k}$	$\sigma_\mathbf{0}$	$\sigma_\mathbf{k}$	$\mathbf{x_0}$	$\mathbf{y_0}$
austria	10	$0.015/\mu_h$	1/10	$(1+\frac{0.1}{k})\sigma_{k-1}$	10	1+rand
atlanta	70	$0.99/\mu_h$	1	$(1+\frac{0.5}{k})\sigma_{k-1}$	2	1+rand
cost266	10	$0.205/\mu_h$	2/5	$(1+\frac{0.2}{k})\sigma_{k-1}$	2	1+rand
france	4	$0.5/\mu_h$	1/5	$(1+\frac{0.2}{k})\sigma_{k-1}$	1/200	1+rand
germany50	2	$0.3143/\mu_h$	1	$(1+\frac{0.2}{k})\sigma_{k-1}$	1/200	1+rand
giul39	40	$0.24/\mu_h$	2	$(1+\frac{0.05}{k})\sigma_{k-1}$	1	1+rand
india35	6	$0.5/\mu_h$	1/5	$(1+\frac{0.2}{k})\sigma_{k-1}$	1/595	1+rand
norway	70	$0.05/\mu_h$	1/5	$(1+\frac{0.2}{k})\sigma_{k-1}$	1	1+rand
pioro40	50	$0.003/\mu_h$	1/20	$(1+\frac{0.1}{k})\sigma_{k-1}$	1	10+rand
zib54	2	0.0013461	1	$(1+\frac{0.1}{k})\sigma_{k-1}$	1	1+rand

Table 12 Stochastic ALM: Constraint validation

Topology	**Constraint** (6.54)	**Constraint** (6.52)	**Constraint** (6.51)
austria	1e-12	7.6310e-04	-1.0100
atlanta	1e-12	7.4047e-04	-1.0952
cost266	1e-12	4.5399e-04	-1.0865
france	1e-12	8.5600e-04	-0.9249
germany50	1e-12	9.3198e-04	+0.1034
giul39	1e-11	9.9534e-04	-0.3320
india35	1e-12	8.7505e-04	-0.1461
norway	1e-10	9.4864e-04	-1.0891
pioro40	1e-12	5.4435e-04	-1.0239
zib54	1e-12	3.3126e-03	-1.0110

show that, except for *germany50*, satisfying the flow conservation constraints is more demanding than the two others. Moreover, for all topologies considered in our numerical experiments satisfying the capacity constraints is more easily achievable.

Table 13 shows the $\lambda_{\min}$ and $\lambda_{\max}$ value for the delay constraints (6.51), the flow conservation and demand satisfaction constraints (6.52), and the capacity constraints (6.54). The main observation that can be drawn from this table is that the values of the dual variables remain below 1e+06. This result is a strong indication that model constraints verify the MF Constraint Qualification (MFCQ) condition.

Table 13 Stochastic ALM: $\lambda_{\min}$ and $\lambda_{\max}$

Topology	Constraint (6.51)		Constraint (6.52)		Constraint (6.54)	
	$\lambda_{\max}$	$\lambda_{\min}$	$\lambda_{\max}$	$\lambda_{\min}$	$\lambda_{\max}$	$\lambda_{\min}$
austria	-3.2153e-09	-3.5216e-06	9466.21	-8101.82	1.8271e-06	-6.0215e-05
atlanta	3.4227e+02	-4.1804e+01	1011.26	342.27	1.9623e+05	9.1665e+05
cost266	8.9867e+00	-8.0113e+00	4285.50	-4029.10	4.1249e+05	4.5484e+04
france	4.6253e-15	-5.6152e-15	4253.76	-4316.88	5.1550e+04	5.1550e+04
germany50	2.6877e+03	-5.9981e+00	197.20	-166.75	0.4251e+02	0.3313e+02
giul39	2.1029e+05	1.9474e+04	59.02	-52.07	0.13861e-01	1.2342e-03
india35	0.2230e+03	-5.1934e+01	389.51	-416.24	2.0500e+03	7.2353e+02
norway	1.1393e-15	-1.1371e-15	3359.40	-3348.72	8.2488e+04	0.4404e+04
pioro40	4.1716e+02	-1.6706e+01	1835.26	-1888.90	5.8916e+04	1.8818e+03
zib54	5.5709e-15	-5.5440e-15	598.45	-600.80	2.7488e+04	2.7488e+04

6.4.3 Convex Relaxation (CR) + PWL

The results of solving the nonlinearly constrained network design problem with McCormick envelopes and piecewise linear approximation of the objective function are reported in Table 14. For each topology (listed in the first column), this table records the minimum objective value obtained (Column 2), the relative gap to optimality as reported by the solver (Column 3), the number of null load arcs, i.e., the number of unused arcs (Column 4), the computation/solving time referred to as root + Branch&Cut (B&C) (Column 5) and the Total time, i.e., the sum of the computation and model generation time (Column 6).

Table 14 Convex Relaxation (CR) + PWL - MILP solver configuration: 128GB/8

Topology	Objective value	Rel. Gap Optimality	Null Load Arcs	Computation Time (s)	
				Root+B&C	Total
austria	656674	0.00%	0	467	489
atlanta	18737126	0.00%	0	5	6
cost266	893727	0.00%	0	396	406
france	7291	0.00%	2	678	687
germany50	135392	0.70%	0	47561	47578
giul39	4774	0.30%	0	119401	119410
india35	12728	0.33%	2	6487	6501
norway	401462	0.00%	0	747	753
pioro40	32540	0.34%	1	9176	9190
zib54	—	—	—	—	—

A clear cut can be observed between instances (such as *atlanta*, *austria*, *cost266*, *france*, *norway*) that can be efficiently solved to optimality (0.00% gap) in less than 1000s and others such as *germany50*, *giul39*, *india35* and *pioro40* that remain with a gap of less than 1% but require more computation time. Among those, *giul39* hits the configured maximum time limit configured and *germany50* requires about one order of magnitude longer to produce a near-optimal solution (gap of 0.70%). The *zib54* topology could not be solved with the proposed method.

6.4.4 Stochastic ALM with (backtracking and) LP start

Instead of using randomly generated flow variable values to initialize the stochastic ALM algorithm, we consider running the algorithm with the initial flow variable values obtained by solving the capacity design problem with continuous flows but without delay constraints. The network design problem is solved with the minimum capacity fraction that produces a feasible solution as detailed in Table 8 (cf. Section 5.4.1). The main motivation stems as follows: with a (piecewise linear) objective function that is convex in the capacity variables, the network design problem provides a lower bound to the nonlinearly/delay constrained problem. Consequently, the LP solver would find a feasible solution to the network design problem whereas the iterative algorithm handles the nonlinearly constrained problem.

The flow variable values are first projected on the set C and then used to produce the initial conditions (x_0, y_0) for the stochastic ALM algorithm. Also, this algorithm is executed with the same parameter setting for all network instances as indicated in Table 15.

Table 15 Stochastic ALM algortihm: parameters

Parameter	Initial Value	Iterate
ρ	$\rho_0 = 0.01\mu_h$	$\rho_k = \min(\rho_{k-1}, \sigma_{k-1})$
t	$t_0 = 1$	$t_k = \frac{1}{2}\sqrt{1 + 4t_{k-1}^2}$
σ	$\sigma_0 = 1.1$	$\sigma_k = 1.1$

The computation time required to obtain the flow variable values used as initial conditions for the Stochastic ALM algorithm is reported in Table 16 (Column 2). This table also lists the initial objective function value obtained (Column 4). The computation time required to reach a relative gap of 1e-09 is reported in (Column 5) together with the feasibility associated to the flow conservation constraints (Column 6), the delay constraints (Column 7) and capacity constraints (Column 8).

Table 16 Capacity cost minimization: Initialization of Stochastic ALM with LP start

Topology	Computation Time (s)	Relative Gap	Objective value	Computation time (s)	Feasibility		
					C.(6.52)	C.(6.51)	C.(6.54)
austria	28.25	0	542648.58	92.88	3.2977e-04	-1.0999	2.0118e-09
atlanta	1.89	0	18466583.53	0.26	1.3034e-04	-1.0922	9.0949e-13
cost266	93.07	0	814768.43	11.00	2.0919e-04	-1.0995	1.9213e-11
france	1.17	0	7364.96	1.54	2.5849e-04	-1.0499	4.5474e-13
germany50	18.89	0	132283.42	55.82	1.7468e-04	-0.3854	1.4210e-14
giul39	374.18	0	4559.70	38.42	2.8276e-04	-0.6316	9.0949e-13
india35	16.04	0	13017.02	15.18	1.2451e+02	-0.9374	2.2737e-13
norway	2.84	0	386968.21	4.77	2.1798e-04	-1.0846	5.9685e-13
pioro40	17.95	0	31037.86	16.29	2.2262e-04	-1.0425	8.2422e-13
zib54	136.36	0	8790313.59	97.83	3.0537e+02	-1.0937	3.6379e-12

For each topology, Table 17 reports the computational results obtained when executing the ALM algorithm initialized with the flow variable values obtained after solving the network design cost minimization problem without delay constraints. This table reports the number of iterations (Column 2) and the computation time (Column 3) required by the ALM algorithm to converge to a solution meeting both flow conservation and delay constraints down to a feasibility threshold of 1e-06 if not interrupted before the time limit (T.L.) set to 43200s. Note that for the flow conservation constraints, the L_∞ norm is used (instead of the L_2-norm) to measure feasibility.

Table 17 Capacity cost minimization: Stochastic ALM with LP start

Topology	**Objective value**	**Nbr of iterations**	**Computation time** (s)	**Feasibility**		
				C.(6.52)	**C.**(6.51)	**C.**(6.54)
austria	542648.69	585	20119	9.5726e-07	-1.0999	1.8923e-09
atlanta	18466588.81	421	113	9.4628e-07	-1.0947	9.0949e-13
cost266	814763.43	384	4192	9.5798e-07	-1.0992	1.8190e-11
france	7269.07	330	550	9.7837e-07	-1.0022	4.2632e-13
germany50	132233.43	614	26636	8.6057e-07	-0.3855	5.6843e-14
giul39	4559.67	1298	40125	5.7243e-08	-9.1393e-03	9.0949e-13
india35	18103.25	1479	T.L.	9.3674e-02	-0.4164	2.2737e-13
norway	386964.69	494	2354	9.9381e-07	-1.0927	4.5474e-13
pioro40	31037.89	332	5828	9.5056e-07	-0.9949	7.6738e-13
zib54	9110875.05	676	T.L.	5.1074e-02	-1.0932	4.5474e-13

6.5 Comparative Analysis

In this section, we compare, in addition to the solution quality, the performance in terms of the computation time required by the two proposed stochastic ALM methods (with and without LP start) against the conventional CR + PWL method. For this purpose, Tables 18 (ALM without LP start) and 19 (ALM with LP start) indicate in Columns 3 and 5, the total computation time required to obtain the objective value reported in Column 2 and 4, respectively.

Table 18 Capacity cost minimization: Comparison Stochastic ALM with CR + PWL method

Topology	**ALM**		**CR + PWL**		**Gain**	
	Minimum Obj. value	**Computation Time** (Iter.)	**Minimum Obj. value**	**Comp. Time**	**Abs.** (s)	**Rel.**
austria	1.1864e+06	40616 s (1246)	**6.56674e+05**	489 s	+40127	83.06
atlanta	1.8880e+07	119 s (580)	**1.8737e+07**	6 s	+113	19.86
cost266	9.14003e+05	8735 s (1165)	**8.9373e+05**	406 s	+8329	21.51
france	7.3650e+03	838 s (996)	**7.2910e+03**	687 s	+151	1.21
germany50	**1.3457e+05**	7971 s (327)	1.3539e+05	47578 s	-39607	**0.17**
giul39	5.9791e+03	39488 s (937)	**4.7740e+03**	119410 s	-79922	0.33
india35	1.3345e+04	7593 s (409)	**1.2728e+04**	6501 s	+1092	1.17
norway	4.3755e+05	9802 s (3724)	**4.0162e+05**	753 s	+9049	1.08
pioro40	3.6421e+04	11249 s (978)	**3.2540e+04**	9190 s	+2059	1.22
zib54	2.0437e+07	16065 s (150)	—	—	—	—

The performance of the stochastic ALM method strongly depends on its parameterization (initial values (x_0, y_0) and smoothing parameter ρ) but also on the finding of the primal t_k and dual σ_k stepsize in each iteration following conditions (3.11).

As shown in Table 18, for relatively small size instances such as *atlanta*, *cost266*, *france*, *norway* but also *india35* and *pioro40*, the computational time remains at the advantage of the CR+PWL method. The stochastic ALM method shows its advantages for larger and/or more complex instances such as *germany50*, *giul39*, and *zib54* by significantly decreasing for the two first instances the computational time. Observe also that, although the value of the objective function obtained with the ALM method is slightly better for *germany50*, its computation time shows a gain by a factor of about 6 in favor of the latter method. For *giul39*, the gain in computation time reaches a factor of about 3. Moreover, for the *zib54* instance, that can't be solved by the CR+PWL method, the objective value obtained with the ALM method can still be improved as indicated in Column 1 of Table 19. The same behaviour can be observed for the two other instances.

Table 19 Capacity cost minimization: Comparison Stochastic ALM + LP start with CR + PWL method

Topology	ALM + LP Start		CR + PWL		Gain	
	Minimum Obj. value	**Computation Time** (Iter.)	**Minimum Obj. value**	**Comp. Time**	**Abs.** (s)	**Rel.**
austria	**5.4265e+05**	20119 s (585)	6.5667e+05	489 s	+19630	41.14
atlanta	**1.8466e+07**	113 s (359)	1.8737e+07	6 s	+107	18.83
cost266	**8.1476e+05**	4192 s (384)	8.9373e+05	406 s	+3786	10.32
france	**7.2690e+03**	550 s (330)	7.2910e+03	687 s	-137	0.80
germany50	**1.3233e+05**	26636 s (614)	1.3539e+05	47578 s	**-20942**	**0.56**
giul39	**4.5597e+03**	40125 s (1298)	4.7740e+03	119410 s	**-79285**	**0.34**
india35	1.8103e+04	43200 s (1479)	**1.2728e+04**	6501 s	+36699	6.64
norway	**3.8696e+05**	2354 s (494)	4.0162e+05	753 s	+1601	3.13
pioro40	**3.1037e+04**	5828 s (332)	3.2540e+04	9190 s	**-3362**	**0.63**
zib54	9.1108e+06	43200 s (676)	—	—	—	—

As shown in Table 19, the stochastic ALM+LP start method competes with the CR+PWL method in terms of solution quality. In total, a smaller objective value is obtained for 9 of the 10 instances: *austria*, *atlanta*, *cost266*, *germany50*, *giul39*, *norway*, *pioro40* and *zib54* (for the latter, the CR+PWL method cannot produce a feasible solution). The main reason seems to find its root in the relaxation of the original problem and the gap this approximation yields in the solving of the resulting model.

Compared to the method without LP start, the stochastic ALM method with LP start also significantly improves for all instances (except for *india35* and *zib54*) the feasibility of flow conservation constraints (6.52) by orders of magnitude. Indeed, by comparing Column 2 of Table 12 to Column 5 of Table 17, we can observe an improvement of the order of 1e03 for all instances except the two cited. The plausible explanation being that the initial flow variable values, which are derived from the solving of the LP problem without nonlinear delay constraints, yield the iterative solver close to a local minimum of the problem with delay constraints. This observation is corroborated by Column C.(6.52) of Table 16, which shows that after one iteration, the feasibility of the flow conservation constraints reaches 1e-04 for all instances (except for *india35* and *zib54*). The latter two instances show that the combination of an LP start to the ALM method may yield a feasible point that is not necessarily suited as initial condition for the minimization of the augmented Lagrangian. Finally, for four of these instances, namely, *france*, *germany50*, *giul39* and *pioro40*, the improvement of the objective function value is accompanied by a decrease in the computation time, as indicated by the absolute and relative gain columns of Table 19.

Hence, although drawing final conclusion on the performance gain for the stochastic ALM method (thus NLP solvers over MIP solvers) remains premature, it is nevertheless evident that there is clear cut for the latter beyond which instances cannot be solved. This cut limit depends on resource factors (number of threads and working memory space) but also properties of the instance (number of vertices and arcs, arc capacity, and demand properties). Indeed *austria* is the largest instance in terms of the number of vertices and arcs but the nominal size of the demands is limited, as demonstrated by the 24 unused arcs and the total number of demands proportional to the number of vertices n (instead of n^2 compared to the other instances). A Lagrangian relaxation could also be investigated for comparative purposes; however, experience [45] shows that the resulting problem shall be decomposable per node to yield any potential advantage -which is not the case here due to the delay constraints.

On the other hand, beyond the potential improvements of the stochastic ALM method in terms of computational time (potentially due also to the experimental nature of the code developed), establishing the actual limits of the method remains open. One can observe though that as the size of the instance increases, the computation time tends nevertheless to become rather large as shown for *austria* although driving initial flow variable values significantly mitigates this increase (by a factor 10 in this case). At least we did not find any instance that could not be solved by means of the proposed ALM method. As part of our future work, experiments with more general nonconvex objective functions will be considered. Such experiments would also enable to further characterize and capture the gain of the proposed method.

7 Conclusion and future work

The performance analysis (computation time, number of iterations, etc.) performed in this paper requires to consider the full sample variant of the stochastic ALM algorithm in order to allow comparison against existing Lagrangian-based variants, including ALGENCAN and RI-ALM. In this setting, for the max-cut problem, the stochastic ALM algorithm nevertheless outperforms both of them in terms of computed value and performance (number of iterations and computation time). For the GEV problem, the results obtained are more contrasted. The stochastic ALM algorithm remains competitive against ALGENCAN by reaching a smaller relative gap and being twice less computation demanding. However, compared to RI-ALM, our algorithm shows similar performance in terms of computation time. The "full sample" experiments realized provide also the best objective value that could be achieved by the stochastic ALM algorithm.

From the use case under consideration, namely the multi-commodity network design problem with load-induced delay constraints, the numerical results obtained are promising. This observation can be drawn if we consider that the stochastic nature of the algorithm would reduce the computation time without necessarily significantly affecting the objective value. The latter being for 90% of the instances considered in our numerical experiments better than the value produced by means of customized LP solving methods.

Nevertheless, finding nonlinearly constrained programming models with nonconvex objectives where, compared to the deterministic variant, the stochastic ALM is competitive in terms of performance but also computes objective values that remain near to those produced by the deterministic variant remains an open research topic part of our future work.

Ethical Approval

Not applicable.

Competing interests

Authors have no competing interests as defined by Springer, or other interests that might be perceived to influence the results and/or discussion reported in this paper.

Funding

Not applicable.

Authors' contributions

Authors have contributed equally to this manuscript. All authors reviewed the manuscript.

Acknowledgments

Not applicable.

Availability of supporting data

For the max-cut problem, the data sets G1, G30, GD97b, GD97c, LFAT5t, Sherman1, Trefethen20b, Trefethen200b, Trefethen500 are openly available and accessible at https://networkrepository.com/networks.php.

The network topology data analysed during the current study are openly available in the SNDlib topology library repository [44] that is accessible at http://sndlib.zib.de/problems.overview.action.

The demand generated during the current study are available from the corresponding author on reasonable request.

References

1. R. Andreani, E. Birgin, J. Martınez and M. L. Schuverdt, Augmented Lagrangian methods under the Constant Positive Linear Dependence constraint qualification, *Math. Program.*, Vol. 111, pp. 5-32, 2008.
2. R. Andreani, E. G. Birgin, J. M. Martınez and M. L. Schuverdt, On Augmented Lagrangian Methods with General Lower-Level Constraints, *SIAM J. Optim.*, Vol. 18, pp. 1286-1309, 2008.
3. C. Audet, P. Hansen, B. Jaumard and G. Savard, A branch and cut algorithm for nonconvex quadratically constrained quadratic programming, *Math. Program.*, Series A, Vol. 87, pp. 131-152, 2000.
4. F. Bach and E. Moulines, Non-Asymptotic Analysis of Stochastic Approximation Algorithms for Machine Learning, *Advances in Neural Information Processing Systems*, pp. 451-459, 2011.
5. H. H. Bauschke and P. L. Combettes, Convex Analysis and Monotone Operator Theory in Hilbert Spaces, Springer, New York, 2nd ed., 2017.
6. S. Becker, J. Bobin and E. J. Candès, NESTA: A fast and accurate first-order method for sparse recovery, *SIAM J. Imag. Sci.*, Vol. 4, pp. 1-39, 2011.
7. A. Beck and M. Teboulle, Smoothing and first order methods: A unified framework, *SIAM J. Optim.*, Vol. 22, pp.557–580, 2012
8. W. Ben-Ameur and A. Ouorou, Mathematical models of the delay constrained routing problem, *Algorithmic Oper. Res.*, Vol.1, pp. 94-103, 2006.
9. A. Ben-Tal, A. Goryashko, E. Guslitzer and A. Nemirovski, Adjustable robust solutions of uncertain linear programs, *Math. Program.*, Vol. 99, pp. 351–376, 2004.
10. D. P. Bertsekas, Constrained Optimization and Lagrange Multiplier Methods, Academic Press, London, 2014.
11. J. R. Birge and J. K. Ho, Optimal Flows in Stochastic Dynamic Networks with Congestion, *Oper. Res.*, Vol. 41, pp. 203-216, 1993.
12. E. G. Birgin and J. M. Martınez, Large-scale active-set box-constrained optimization method with spectral projected gradients, *Comput. Optim. Appl.*, Vol. 23, pp.101-125, 2002.
13. J. Bolte, S. Sabach and M. Teboulle, Nonconvex lagrangian-based optimization: monitoring schemes and global convergence, *Math. Oper. Res.*, Vol. 43, pp. 1210-1232, 2018.
14. L. Bottou, Stochastic gradient learning in neural networks, *Proceedings of Neuro-Nîmes 91*, EC2, Nimes, France, 1991.
15. L. Bottou, F. E. Curtis and J. Nocedal, Optimization Methods for Large-Scale Machine Learning, *Siam Review*, Vol. 60, pp. 223-311, 2018.
16. J. V. Burke and A. Engle, Line search and trust-region methods for convex-composite optimization, https://arxiv.org/abs/1806.05218, 2018.
17. M. Carey, Optimal Time Varying Flows On Congested Networks, *Oper. Res.*, Vol. 35, pp.58-69, 1987.
18. C. Chekuri, G. Oriolo, M. G. Scutella and F. B. Shepherd, Hardness of robust network design, *Networks*, Vol. 50, pp. 50-54, 2007.

19. C. Contardo and S. U. Ngueveu, On the approximation of separable non-convex optimization programs to an arbitrary numerical precision, Rapport LAAS no.21222, https://hal.archives-ouvertes.fr/hal-03336022, Sep. 2021.
20. P. L. Combettes, Đ. Dũng and B. C. Vu, Proximity for sums of composite functions, *J. Math. Anal.*, Vol., 380, pp. 680-688, 2011.
21. S. Cui and U. V. Shanbhag, Variance-reduced splitting schemes for monotone stochastic generalized equations, *IEEE Trans. Autom. Control*, 2023.
22. J. W. Eaton, GNU Octave Manual, Network Theory Limited, 2022, Available at https://www.octave.org.
23. D. Gabay, Applications of the method of multipliers to variational inequalities, in: M. Fortinand R. Glowinski (Eds.), Augmented Lagrangian Methods: Applications to the Solution of Boundary-Value Problems, North-Holland, Amsterdam, 1983.
24. D. Gabay and B. Mercier, A dual algorithm for the solution of nonlinear variational problems via finite element approximation, *Comput. Math. with Appl.*, Vol. 2, pp. 17-40, 1976.
25. S. Ghadimi and G. Lan, Accelerated gradient methods for nonconvex nonlinear and stochastic programming, *Math. Program.*, Vol. 156, pp. 59-99, 2016.
26. J. Giesen and S. Laue, Distributed Convex Optimization with Many Non-Linear Constraints, https://arxiv.org/pdf/1610.02967.pdf, 2018.
27. F. Glover, Improved Linear Integer Programming Formulations of Nonlinear Integer Problems, *Manag. Sci.*, Vol. 22, pp. 455-460, 1975.
28. M. R. Garey, D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, ISBN 0-7167-1045-5, 1979.
29. M. X. Goemans and D. P. Williamson, Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, *J. ACM*, Vol 42, pp. 1115–1145, 1995.
30. A. Gupta, J. Kleinberg, A. Kumar, R. Rastogi and B. Yener, Provisioning a virtual private network: A network design problem for multicommodity flow, Proc 33rd annual ACM Symposium on Theory of Computing (STOC 2001), Heraklion, Crete, Greece, 2001, pp.389–398.
31. H. L. Hijazi, P. Bonami and A. Ouorou, Robust delay-constrained routing in telecommunications, *Ann. Oper. Res.*, Vol. 206, pp. 163-181, 2013.
32. E. Hazan and S. Kale, Beyond the regret minimization barrier: an optimal algorithm for stochastic strongly-convex optimization, *J. Mach. Learn. Res.*, Vol. 15, pp. 2489-2512, 2014.
33. M. R. Hestenes, Multiplier and gradient methods, *J. Optim. Theory Appl.*, Vol 4, pp. 303-320, 1969.
34. S. G. Johnson, The NLopt nonlinear-optimization package, Available at http://github.com/stevengj/nlopt.
35. A. Y. Khintchine, Mathematical theory of a stationary queue, *Mat. Sb.*, Vol. 39, pp. 73–84, 1932.
36. A. S. Lewis and S. J. Wright, A proximal method for composite minimization, *Math. Program.*, Vol. 158, pp. 501–546, 2016.
37. Z. Li and Y. Xu, Augmented Lagrangian based first-order methods for convex and nonconvex programs: nonergodic convergence and iteration complexity, 2021. Available at https://arxiv.org/abs/2003.08880
38. H. Xhang, S. Ghadimi and G. Lan, Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization, *Math. Program., Ser. A,* Vol. 155, pp. 267–305, 2016
39. David G. Luenberger, Yinyu Ye, et al., Linear and nonlinear programming, Volume 2, Springer, 2007, Third edition.
40. G. P. McCormick, Computability of global solutions to factorable nonconvex programs: Part I - convex underestimating problems, *Math. Program.* Vol. 10, pp. 147-175, 1976.
41. D. K. Merchant and G. L. Nemhauser, A Model and an Algorithm for the Dynamic Traffic Assignment Problems, *Transp. Sci.*, Vol. 12, pp. 183-199, 1978.
42. A. Nemirovski, A. Juditsky, G. Lan and A. Shapiro, Robust stochastic approximation approach to stochastic programming, *SIAM J. Optim.*, Vol. 19, pp. 1574–1609, 2008.
43. Y. Nesterov, Smooth minimization of non-smooth functions, *Math. Program.,* Vol. 103, pp. 127-152, 2005.
44. S. Orlowski, R. Wessäly, M. Pióro and A. Tomaszewski, SNDlib 1.0 - Survivable Network Design Library, *Networks*, Vol. 55, pp. 276-286, 2010.
45. D. Papadimitriou, B. Fortz and E. Gorgone, Lagrangian relaxation for the time-dependent combined network design and routing problem, IEEE International Conference on Communications (ICC), pp. 6030-6036, 2015.
46. D. Papadimitriou and B. Vu, An augmented Lagrangian method for nonconvex composite optimization problems with nonlinear constraints, *Optimization and Engineering*, Springer, Nov. 2023.
47. Q. Tran-Dinh, O. Fercoq and V. Cevher, A smooth primal-dual optimization framework for nonsmooth composite convex minimization, *SIAM J. Optim.*, Vol. 28, pp. 96-134,2018
48. F. Pollaczek, Über eine Aufgabe der Wahrscheinlichkeitstheorie, *Math. Z*, Vol. 32, pp. 64–100, 1930.
49. M. J. D. Powell, A method for non-linear constraints in minimization problems, Optimization, R. Fletcher Ed., Academic Press, New York, NY, pp. 283–298, 1969.
50. M. Poss and C. Raack, Affine Recourse for the Robust Network Design Problem: Between Static and Dynamic Routing, *Networks*, Vol. 61, pp. 180–198, 2013.
51. M. F. Sahin, A. Alacaoglu, F. Latorre and V. Cevher, An inexact augmented Lagrangian framework for nonconvex optimization with nonlinear constraints, *In Advances in Neural Information Processing Systems*, pp. 13943–13955, 2019.
52. H. D. Sherali and W. P. Adams, A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems, *SIAM J. Discret. Math.*, Vol. 3, pp. 411-430, 1990.
53. H. D. Sherali and C. H. Tuncbilek, A global optimization algorithm for polynomial programming problems using a reformulation-linearization technique, *J. Glob. Optim.*, Vol. 2, pp. 101-112, 1992.
54. H. D. Sherali and W. P. Adams, A hierarchy of relaxations and convex hull characterizations for mixed-integer zero-one programming problems, *Discrete. Appl. Math.*, Vol. 52, pp. 83-106, 1994.

55. H. D. Sherali and W. P. Adams, A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems, Kluwer Academic Publishers, Dordrecht, Boston, London, 1999.
56. H. D. Sherali and H. Wang, Global optimization of nonconvex factorable programming problems, *Math. Program.*, Vol 89, pp. 459-478, 2001.
57. W. P. Adams and H. D. Sherali, A hierarchy of relaxations leading to the convex hull representation for general discrete optimization problems, In M. Guignard, K. Spielberg (Eds.), State-of-the-Art in Integer Programming, *Ann. Oper. Res.*, Vol. 140, pp. 21-47, 2005.
58. T. Valkonen, A primal-dual hybrid gradient method for nonlinear operators with applications to MRI, *Inverse Probl.*, Vol.30, 055012, 2014.
59. http://proximity-operator.net/
60. https://sites.google.com/site/fomsolv
61. Z. Li, P. Y. Chen, S. Liu, S. Lu and Y. Xu, Rate-improved Inexact Augmented Lagrangian Method for Constrained Nonconvex Optimization, *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, San Diego, California, USA. PMLR, Vol. 130, 2021.
62. H. Robbins and D. Siegmund, A convergence theorem for non negative almost supermartingales and some applications, In: Rustagi JS, editor. *Optimizing methods in statistic*, New York (NY): Academic Press, pp. 233-257, 1971.
63. H. Robbins and S. Monro, A Stochastic Approximation Method, *Ann. Math. Statist.*, Vol. 22, pp. 400-407, 1951.
64. V. D. Nguyen and B. C. Vũ, Convergence analysis of the stochastic reflected forward–backward splitting algorithm, *Optim. Lett.*, Vol. 16, pp. 2649-2679, 2022.