GLOBAL OPTIMIZATION OF NONLINEAR NETWORK DESIGN

ARVIND U. RAGHUNATHAN *

Abstract. A novel approach for obtaining globally optimal solutions to design of networks with nonlinear resistances and potential driven flows is proposed. The approach is applicable to networks where the potential loss on an edge in the network is governed by a convex and strictly monotonically increasing function of flow rate. We introduce a relaxation of the potential loss constraint and formulate the design problem as a mixed-integer nonlinear program (MINLP). A linearization-based approach with tailored cuts is proposed that improves the computational efficiency over a standard implementation. We have also implemented a simple heuristic approach for finding feasible solutions at the root node and during the search process. The algorithm has been implemented with `IBM-ILOG CPLEX` and is shown to be computationally effective on a number of examples from literature.

Key words. nonlinear resistance networks mixed integer nonlinear program linearizations-based branch-and-cut

AMS subject classifications. 90C11, 90C26, 90C27, 90C57.

1. Introduction. Networks are employed for transporting energy, material, people or information between source and demand locations. The term nonlinear networks refers to the class of networks in which: (i) flow through the network is driven by a potential and (ii) potential loss on an edge in the network is a nonlinear function of the flow rate through that edge. The demand nodes typically represent a local distribution network. Consequently, a minimum potential is desired at the demand nodes in order to drive the flow further in the local distribution network without additional potential input. For a given network, the potential at the demand nodes can be affected by appropriately choosing the resistances on the network. Optimization of nonlinear network design refers to the minimum cost design of resistances on a nonlinear network so as to satisfy the minimum potential requirement at the demand nodes. Discrete choices for the resistances coupled with the nonlinear potential loss constraints puts the optimization problem in the class of Mixed Integer Nonlinear Programs with non-convex relaxations (non-convex MINLP for short).

Several approaches have been developed over the last 50 years for obtaining solutions on this problem. One of the popular approaches is the *split-pipe* approach proposed by Shamir and co-workers in the context of hydraulic networks. Alperovits and Shamir [5] relax the combinatorial aspect of discrete size choice by allowing an edge to be composed of multiple resistance sizes. The algorithm alternates between (i) solving a linear program given flows to determine the fractional lengths of different resistance sizes on each edge such that the potential requirements are satisfied and (ii) using the sensitivity of linear program's solution to update the set of flows. The procedure is continued until no further progress is observed. The same formulation was further studied by Goulter *et. al.* [23]. Fujiwara and Khang [21] proposed a two-phase procedure which can be considered an extension of the approach in [5]. Eiger *et. al.* [18] proposed an algorithm for the global optimization of the *split-pipe* formulation. A branch-and-bound type algorithm is proposed incorporating a bundle trust-region algorithm for obtaining local solutions and a dual procedure is used to estimate the global optimality gap. Zhang and Zhu [34] attempt to provide a theoretically sound version of the algorithm proposed in [5] using conjugate duality theory in

*United Technologies Research Center, East Hartford, CT 06118. Currently, with Mitsubishi Electric Research Laboratories, 201 Broadway, Cambridge, MA 02139. `raghunathan@merl.com`

conjunction with a trust-region algorithm. It must be mentioned that the *split-pipe* approach is not guaranteed to produce a single resistance size choice for each edge in the network. In a separate work, Hansen *et. al.* [24] employ a successive linear programming algorithm. No claims on the optimality of the solution are provided. Meta-heuristic approaches are also explored in [3, 30]. All of the approaches have been shown to be effective on small sized problems only.

Artina and Walker [6] proposed a mixed integer linear programming (MILP) formulation. The nonlinear potential loss functions are modeled using piece-wise linear inner approximations. The direction of flow on each edge is also modeled using binary variables. The resulting MILP is huge and results in poor linear relaxations. The computational experience with this approach has been unsatisfactory and limited to small problems. Further, the solution obtained from this approach is not guaranteed to satisfy the nonlinear potential loss constraints.

In an effort to rectify the shortcomings of the previous approaches, Bragalli *et. al.* [13] proposed a MINLP approach. The authors proposed a continuous reformulation of the cost function which results in a better nonlinear relaxation of the MINLP. Their formulation of the potential loss constraints is non-smooth. So they smoothed the potential drop constraints and the resulting MINLP is solved using a branch-and-bound algorithm implemented in open-source software `Bonmin` [9, 31]. The approach is only guaranteed to find a locally optimal solution. A variety of starting points are chosen in an attempt to approach the global optimal solution. The authors report successful optimization of several instances of hydraulic network problems.

In this paper, we propose a different method for handling the non-convexity. The network is expanded by cloning edges where each clone represents one of the allowable resistance size on that edge . We introduce separate flow variables for the two flow directions on an edge of particular resistance size. We utilize the convexity of the potential loss function to yield a convex relaxation of the optimization problem. The convex relaxation allows formulating the design problem as a MINLP with convex nonlinear constraints. We solve the optimization problem using the linearizations-based LP/NLP-BB framework of Quesada and Grossmann [29] and also implemented in `FilMINT`[4]. Specific cuts are derived for the class of problems that outperforms the standard linearizations employed in [4].

The paper is organized as follows. Description of the notation and the problem is provided in Section 2. Section 3 presents a convex formulation for the network analysis problem and presents results on uniqueness of solutions. Section 4 presents the convex relaxation of the potential loss constraints and the MINLP formulation. A linearization based algorithm is described in Section 5. Section 6 describes a heuristic method for computing an initial feasible solution and for finding feasible solutions during the search process. Computational results on a suite of test problem [1] are provided in Section 7.

2. Problem Description. The notation associated with problem data is described below:

$\boldsymbol{N}$	-	set of nodes in the network
$\boldsymbol{N}^{\text{src}}$	-	set of nodes that are fixed potential sources in the network, $\boldsymbol{N}^{\text{src}} \subset \boldsymbol{N}$
$\boldsymbol{E}$	-	set of edges in the network $e = (i, j)$ for $i, j \in \boldsymbol{N}$. We will employ e for brevity and where necessary make explicit the two nodes i, j that are connected by e. It is assumed without loss of generality that for the edges $e = (i, j)$ connected to a source node, $i \in \boldsymbol{N}^{\text{src}}$. All other edges are provided an arbitrary orientation.

$\boldsymbol{E}_1(i)$ - set of edges $e \in \boldsymbol{E}$ incident on the node i and $e = (i,j)$, that is $\boldsymbol{E}_1(i) = \{e | e = (i,j) \in \boldsymbol{E}\}$

$\boldsymbol{E}_2(i)$ - set of edges $e \in \boldsymbol{E}$ incident on the node i and $e = (j,i)$, that is $\boldsymbol{E}_2(i) = \{e | e = (j,i) \in \boldsymbol{E}\}$. Note that $\boldsymbol{E}_2(i) = \emptyset \; \forall \; i \in \boldsymbol{N}^{\text{src}}$.

π_i^{src} - specified potential at the source nodes of the network, $i \in \boldsymbol{N}^{\text{src}}$

$\pi_i^{\min}$ - minimum potential required at demand nodes i of the network, $i \in \boldsymbol{N} \setminus \boldsymbol{N}^{\text{src}}$

$\pi_i^{\max}$ - maximum potential tolerated at demand nodes i of the network, $i \in \boldsymbol{N} \setminus \boldsymbol{N}^{\text{src}}$

I_i^{dem} - flow rate demanded out of the node i of the network, $i \in \boldsymbol{N} \setminus \boldsymbol{N}^{\text{src}}$

L_e - length of the edge $e \in \boldsymbol{E}$ in the network

$\boldsymbol{\mathcal{R}}$ - set of available resistance sizes from which to choose for each edge in the network, $\boldsymbol{\mathcal{R}} = \{R_1, \ldots, R_{n_r}\}$

c_r - cost per unit length of resistance R_r, $r \in \{1, \ldots, n_r\}$

$\phi_r(I)$ - function relating the potential loss per unit length to the flow rate I through an edge with resistance $R_r \in \boldsymbol{\mathcal{R}}$. We assume without loss of generality that $\phi_r : \mathbb{R}_+ \to \mathbb{R}_+$, where $\mathbb{R}_+$ is the set of non-negative reals.

$I_{e,r}^{\max}$ - maximum flow rate allowed on edge $e \in \boldsymbol{E}$ with resistance $R_r \in \boldsymbol{\mathcal{R}}$.

r_e - index of resistance choice for edge $e \in \boldsymbol{E}$.

The notation used for the variables are:

I_e - flow rate on edge $e = (i,j) \in \boldsymbol{E}$. A positive value for the flow rate implies that the flow occurs from node i to j, while a negative value implies that the flow occurs from node j to i.

I_e^+ - flow rate on edge $e = (i,j) \in \boldsymbol{E}$ where the flow is from i to j

I_e^- - flow rate on edge $e = (i,j) \in \boldsymbol{E}$ where the flow is from j to i

π_i - potential at node $i \in \boldsymbol{N}$

$\Delta\pi_e^+$ - potential drop on edge $e = (i,j) \in \boldsymbol{E}$ when flow is from i to j

$\Delta\pi_e^-$ - potential drop on edge $e = (i,j) \in \boldsymbol{E}$ when flow is from j to i

$x_{e,r}$ - binary variable denoting the choice of resistance $r \in \{1, \ldots, n_r\}$ on edge $e \in \boldsymbol{E}$

x_e^{dir} - binary variable denoting the choice of flow direction on edge $e = (i,j) \in \boldsymbol{E}$. The flow is from i to j when $x_e^{\text{dir}} = 1$ and from j to i when $x_e^{\text{dir}} = 0$.

For ease of representation, we will use boldface notation to represent a set of quantities that are indexed over the set of nodes or edges. For instance, $\boldsymbol{I}$ will denote the set of all flows $I_e \; \forall \; e \in \boldsymbol{E}$ and $\boldsymbol{\pi}$ will denote the set of all potentials $\pi_i \; \forall \; i \in \boldsymbol{N}$. The set over which the quantities are indexed will be obvious from the context.

The following are the assumptions on the problem parameters that will be used through out the paper.

Assumptions:

(A1) The potential loss per unit length function $\phi_r : \mathbb{R}_+ \to \mathbb{R}_+$ for $r = 1, \ldots, n_r$ is at least once continuously differentiable and strictly monotonically increasing function of flow rate, I. In other words,

$$\phi_r(0) = 0 \text{ and } \frac{d\phi_r(I)}{dI} > 0 \; \forall \; I \geq 0, r = 1, \ldots, n_r. \tag{2.1}$$

(A2) The function $\phi_r(I)$ for $r = 1, \ldots, n_r$ is convex.

(A3) The allowable resistance choices in $\boldsymbol{\mathcal{R}}$ are assumed to satisfy $R_1 > R_2 \ldots > R_{n_r}$ and $c_1 < c_2 < \ldots < c_{n_r}$. The flow bounds satisfy $I^{\max}_{e,1} < \ldots < I^{\max}_{e,n_r} \; \forall \, e \in \boldsymbol{E}$.

Assumptions (A1) and (A2) are typically satisfied by networks arising from a number of application domains - electrical circuits, static structures and hydraulic circuits. The potential loss function $\phi_r(I)$ typically takes the form:

$$\phi_r(I) \propto I^{\alpha} R_r \text{ with } I \geq 0$$

where $\alpha > 1$. This implies satisfaction of Assumptions (A1) and (A2). Some of the common fluid networks of water, crude oil, natural gas satisfy the above assumptions. However, there exist fluids, called *non-newtonian fluids* [2], for which the assumptions (A1) and (A2) do not hold. Examples of non-newtonian fluids include lubricants, paints, corn starch and blood. Analogously in the electrical circuits there exist components such as *varistors* which maintain high resistance up to a certain applied potential and negligible resistance for higher voltages. The techniques described in this work cannot be applied to networks of such fluids or electrical components. Assumption (A3) states that cost per unit length and flow capacity $I^{\max}_{e,r}$ increase with decrease in resistance. This also holds for the above application domains. For instance, in electrical and hydraulic circuits the resistance function satisfies $R_r \propto D^{-\beta}$ where $\beta > 1$. The parameter D represents the thickness of the wire in the case of electrical circuits and inner diameter of pipe in hydraulic circuits. The increased use of material implies increased cost but decreased potential loss per unit length for the resistances. Further, the increased material also implies increased capacity and hence, more flow in the wire or pipe. We refer the interested reader to the paper by Hendrickson and Janson [25] for the form of the potential loss function corresponding to the different application domains.

2.1. Mathematical Formulation. Given a network with specified resistances $r_e \; \forall \, e \in \boldsymbol{E}$, *i.e.* $R_{r_e} \in \boldsymbol{\mathcal{R}}$ on the edges $e \in \boldsymbol{E}$, the network equations relating the potential at the nodes and the flow in the network can be represented as:

$$\begin{aligned} \pi_i - \pi_j = \text{sgn}(I_e)\phi_{r_e}(|I_e|)L_e \;\; \forall \, e \in \boldsymbol{E} \\ \pi_i = \pi_i^{\text{src}} \;\; \forall \, i \in \boldsymbol{N}^{\text{src}} \\ \sum_{e \in \boldsymbol{E}_2(i)} I_e - \sum_{e \in \boldsymbol{E}_1(i)} I_e = I_i^{\text{dem}} \;\; \forall \, i \in \boldsymbol{N} \setminus \boldsymbol{N}^{\text{src}} \end{aligned} \qquad (\text{NLNEQN}(\boldsymbol{r}))$$

where $\text{sgn}(\cdot)$ is the signum function and $\boldsymbol{r}$ $(:= \{r_e \in 1, \ldots, n_r | e \in \boldsymbol{E}\})$ represents the set of resistance choices for each edge. The equations in (NLNEQN($\boldsymbol{r}$)), commonly referred to as *network analysis equations*, form a set of nonlinear equations in π_i and I_e. The first equation in (NLNEQN($\boldsymbol{r}$)) refers to the loss in potential on each edge due to the flow, the second equation fixes the potential at some nodes designated as fixed potential sources and the last equation denotes the conservation of flow at non-source nodes. The potential loss equation has been incorrectly referred by Bragalli *et. al.* [13] as being non-smooth. This was brought to the authors notice individually by both Belotti [8] and Leyffer [27]. The right and left first derivatives of the function $\text{sgn}(I_e)\phi_{r_e}(|I_e|)$ are:

$$\begin{aligned} \left.\frac{d(\text{sgn}(I_e)\phi_{r_e}(|I_e|))}{dI_e}\right|_{0^+} &= \lim_{\delta > 0, \delta \to 0} \frac{\phi_{r_e}(\delta) - \phi_{r_e}(0)}{\delta} = \lim_{\delta > 0, \delta \to 0} \frac{\phi_{r_e}(\delta)}{\delta} \\ \left.\frac{d(\text{sgn}(I_e)\phi_{r_e}(|I_e|))}{dI_e}\right|_{0^-} &= \lim_{\delta < 0, \delta \to 0} \frac{(-\phi_{r_e}(-\delta)) - (\phi_{r_e}(0))}{\delta} = \lim_{\delta < 0, \delta \to 0} \frac{\phi_{r_e}(-\delta)}{-\delta} \end{aligned} \qquad (2.2)$$

which clearly converge to the same value in the limit. It is easy to show that the same holds for other derivatives as well. The non-smoothness arises when we attempt to express $\mathrm{sgn}(I_e)\phi_{r_e}(|I_e|)$ in modeling languages such as `AMPL` [20] using the standard operator such as the `power` function. The `power(a,b)` function requires the `a` to be non-negative whenever `b` is fractional. Solvers interfaced with `AMPL` rely on the automatic differentiation capability provided by `AMPL`. Clearly automatic differentiation will fail on this when it applies the chain-rule of differentiation to the combination of `sgn` and `power` functions. This necessitates the smoothing of the expression as was done by Bragalli *et. al.* Alternately, it is possible to define an user-defined function in `AMPL`, described in [22], which allows the user to supply derivatives and deal with this issue.

The nonlinear network design optimization problem aims to determine the resistance choices $r_r \ \forall\, e \in \boldsymbol{E}$ for the network so as to minimize the cost while satisfying the flow demands and bounds on the flows and potentials. We state the optimization problem as the following disjunctive program:

$$
\begin{aligned}
\min & \sum_{e\in\boldsymbol{E}} \tilde{c}_e L_e \\
\text{s.t.} & \sum_{e\in\boldsymbol{E}_2(i)} I_e - \sum_{e\in\boldsymbol{E}_1(i)} I_e = I_i^{\text{dem}} \ \forall\, i \in \boldsymbol{N}\setminus\boldsymbol{N}^{\text{src}} \\
& \bigvee_{r=1}^{n_r} \left[\begin{array}{c} \tilde{c}_e = c_r \\ -I_{e,r}^{\max} \le I_e \le I_{e,r}^{\max} \\ \left[\begin{array}{c} I_e \ge 0 \\ \pi_i - \pi_j = \phi_r(I_e) L_e \end{array}\right] \vee \left[\begin{array}{c} I_e < 0 \\ \pi_i - \pi_j = -\phi_r(-I_e) L_e \end{array}\right] \end{array} \right] \\
& \hspace{20em} \forall\, e \in \boldsymbol{E} \\
& \pi_i^{\min} \le \pi_i \le \pi_i^{\max} \ \forall\, i \in \boldsymbol{N}\setminus\boldsymbol{N}^{\text{src}} \\
& \pi_i = \pi_i^{\text{src}} \ \forall\, i \in \boldsymbol{N}^{\text{src}}.
\end{aligned}
\tag{2.3}
$$

We have resorted to the disjunctive formulation to avoid imposing a nonlinear formulation of the potential loss constraints. Observe that the signum function in potential loss constraints has been handled through the inner disjunction. The potential loss equations in (NLNEQN($\boldsymbol{r}$)) are non-convex constraints. In addition, the discrete resistance choices puts the nonlinear network design in the class of MINLP with non-convex relaxations. In the following section, we will consider the question of existence and uniqueness of solutions to the network analysis problem.

3. Convex Formulation for Network Analysis. In the nonlinear network optimization problem in (2.3), the solution of the network analysis problem that results from a particular choice of resistances needs to be checked for satisfaction of bounds on flows and potentials.

$$
\begin{aligned}
& (\boldsymbol{\pi}, \boldsymbol{I}) \text{ solves } (\text{NLNEQN}(\boldsymbol{r})) \\
& -I_{e,r_e}^{\max} \le I_e \le I_{e,r_e}^{\max} \ \forall\, e \in \boldsymbol{E} \\
& \pi_i^{\min} \le \pi_i \le \pi_i^{\max} \ \forall\, i \in \boldsymbol{N}\setminus\boldsymbol{N}^{\text{src}}
\end{aligned}
\tag{NLP($\boldsymbol{r}$)}
$$

It is easy to see that the satisfaction of the bounds on flows and potentials by the solution to (NLNEQN($\boldsymbol{r}$)) is necessary and sufficient for the feasibility of NLP($\boldsymbol{r}$). Hence, we consider the existence and uniqueness of solutions to (NLNEQN($\boldsymbol{r}$)). The material presented in this section is also of independent interest from the network analysis perspective. The equations in (NLNEQN($\boldsymbol{r}$)) are nonlinear which complicates the analysis. We present an alternate optimization-based formulation first presented by Cherry [15]. This formulation has been re-discovered in the context of different application domains as documented by Hendrickson and Janson [25].

Consider the following optimization problem,

$$\begin{aligned}
\min & \sum_{e\in \boldsymbol{E}} (\Phi_{r_e}(I_e^+) + \Phi_{r_e}(I_e^-))L_e - \sum_{i\in \boldsymbol{N}^{\text{src}}} \sum_{e\in \boldsymbol{E}} \pi_i^{\text{src}}(I_e^+ - I_e^-) \\
\text{s.t.} & \sum_{e\in \boldsymbol{E}_2(i)} (I_e^+ - I_e^-) - \sum_{e\in \boldsymbol{E}_1(i)} (I_e^+ - I_e^-) = I_i^{\text{dem}} \; \forall\, i \in \boldsymbol{N} \setminus \boldsymbol{N}^{\text{src}} \qquad (\text{CVXNLP}(\boldsymbol{r})) \\
& 0 \le I_e^+, I_e^- \; \forall\, e \in \boldsymbol{E}
\end{aligned}$$

where the function $\Phi_r(\cdot)$ is defined as

$$\Phi_r(I) = \int_0^I \phi_r(I')dI'. \tag{3.1}$$

Collins *et al.* [16] show that solution to (CVXNLP($\boldsymbol{r}$)) satisfies the potential driven flows (NLNEQN($\boldsymbol{r}$)). The proof in [16] assumes that the direction of flow on the edges is known a priori. We relax this assumption and include the result here for completeness.

Prior to presenting the result we state the first-order stationary conditions associated with the optimization problem (CVXNLP($\boldsymbol{r}$)). Suppose that $\hat{\boldsymbol{I}}^+, \hat{\boldsymbol{I}}^- \in \mathbb{R}^{|\boldsymbol{E}|}$ solves (CVXNLP($\boldsymbol{r}$)) with the associated multipliers: $\hat{\boldsymbol{\lambda}} \in \mathbb{R}^{|\boldsymbol{N}|\setminus|\boldsymbol{N}^{\text{src}}|}$ for flow conservation constraints and $\hat{\boldsymbol{\mu}}^+, \hat{\boldsymbol{\mu}}^- \in \mathbb{R}^{|\boldsymbol{E}|}$ for lower bound constraints on flow variables. From optimization theory, we have that the solution to (CVXNLP($\boldsymbol{r}$)) satisfies the following first-order stationary conditions [28]:

$$\hat{\lambda}_j - \pi_i^{\text{src}} + \phi_{r_e}(\hat{I}_e^+)L_e - \hat{\mu}_e^+ = 0 \; \forall\, i \in \boldsymbol{N}^{\text{src}}, e \in \boldsymbol{E} \tag{3.2}$$
$$\pi_i^{\text{src}} - \hat{\lambda}_j + \phi_{r_e}(\hat{I}_e^-)L_e - \hat{\mu}_e^- = 0 \; \forall\, i \in \boldsymbol{N}^{\text{src}}, e \in \boldsymbol{E} \tag{3.3}$$
$$\hat{\lambda}_j - \hat{\lambda}_i + \phi_{r_e}(\hat{I}_e^+)L_e - \hat{\mu}_e^+ = 0 \; \forall\, i \notin \boldsymbol{N}^{\text{src}}, e \in \boldsymbol{E} \tag{3.4}$$
$$\hat{\lambda}_i - \hat{\lambda}_j + \phi_{r_e}(\hat{I}_e^-)L_e - \hat{\mu}_e^- = 0 \; \forall\, i \notin \boldsymbol{N}^{\text{src}}, e \in \boldsymbol{E} \tag{3.5}$$
$$\hat{I}_e^+, \hat{\mu}_e^+ \ge 0, \hat{I}_e^+ \cdot \hat{\mu}_e^+ = 0 \; \forall\, e \in \boldsymbol{E} \tag{3.6}$$
$$\hat{I}_e^-, \hat{\mu}_e^- \ge 0, \hat{I}_e^- \cdot \hat{\mu}_e^- = 0 \; \forall\, e \in \boldsymbol{E} \tag{3.7}$$
$$\sum_{e\in \boldsymbol{E}_2(i)} (\hat{I}_e^+ - \hat{I}_e^-) - \sum_{e\in \boldsymbol{E}_1(i)} (\hat{I}_e^+ - \hat{I}_e^-) = I_i^{\text{dem}} \; \forall\, i \in \boldsymbol{N} \setminus \boldsymbol{N}^{\text{src}}. \tag{3.8}$$

THEOREM 3.1. *Suppose Assumption (A1) holds, then there exists a solution* $(\boldsymbol{\pi}, \boldsymbol{I})$ *for* (NLNEQN($\boldsymbol{r}$)) *if and only if there exists a solution* $(\hat{\boldsymbol{I}}^+, \hat{\boldsymbol{I}}^-, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\mu}}^+, \hat{\boldsymbol{\mu}}^-)$ *for* (CVXNLP($\boldsymbol{r}$)). *Further, the solution is unique.*

Proof. We will first consider the *if* part. Suppose $(\hat{\boldsymbol{I}}^+, \hat{\boldsymbol{I}}^-, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\mu}}^+, \hat{\boldsymbol{\mu}}^-)$ solves (CVXNLP($\boldsymbol{r}$)). Note that $\hat{I}_e^+, \hat{I}_e^- > 0$ cannot occur for any $e \in \boldsymbol{E}$. If this were not true then by re-defining

$$\tilde{I}_e^+ := \max(0, \hat{I}_e^+ - \hat{I}_e^-) \text{ and } \tilde{I}_e^- := \max(0, \hat{I}_e^- - \hat{I}_e^+) \tag{3.9}$$

we can obtain a set of flows that is feasible to (CVXNLP($\boldsymbol{r}$)) with $\tilde{I}_e^+ \le \hat{I}_e^+$, $\tilde{I}_e^- \le \hat{I}_e^-$. The flows $\tilde{\boldsymbol{I}}^+, \tilde{\boldsymbol{I}}^-$ are feasible for (CVXNLP($\boldsymbol{r}$)) and the strict monotonicity of $\phi_r(\cdot)$

(Assumption (A1)) implies that the flows defined in (3.9) have a lower objective which contradicts the assumption of optimality of $(\hat{\boldsymbol{I}}^{+}, \hat{\boldsymbol{I}}^{-})$. Further, the complementarity constraints in (3.6)-(3.7) impose the restriction that if $\hat{I}_e^+$(or $\hat{I}_e^-$) > 0 then $\hat{\mu}_e^+$(or $\hat{\mu}_e^-$) $= 0$. If $\hat{I}_e^+ = \hat{I}_e^- = 0$ for some $e \in \boldsymbol{E}$ then, substituting the flows in (3.4) and (3.5) we have that,

$$\left.\begin{aligned} \hat{\lambda}_j - \hat{\lambda}_i + \phi_{r_e}(0)L_e - \hat{\mu}_e^+ = 0 \\ \hat{\lambda}_i - \hat{\lambda}_j + \phi_{r_e}(0)L_e - \hat{\mu}_e^- = 0 \end{aligned}\right\} \implies \hat{\mu}_e^+ = \hat{\mu}_e^- = 0$$

which follows from adding the equalities and noting that $\hat{\mu}_e^+, \hat{\mu}_e^- \geq 0$. Further note that there exists no demand for source nodes $i \in \boldsymbol{N}^{\text{src}}$ and by our assumption all edges connected to source nodes are of the form $(i,j) \in \boldsymbol{E}$, we have that $\hat{I}_e^- = 0$. If this were not the case then we can set $\hat{I}_e^- = 0$ and obtain a feasible solution with a strictly reduced objective value which contradicts the assumption of optimality. Consequently, we have that equations (3.2)-(3.7) reduce to

$$\begin{aligned} \hat{\lambda}_j - \pi_i^{\text{src}} + \phi_{r_e}(\hat{I}_e^+)L_e = 0 \ \forall\, i \in \boldsymbol{N}^{\text{src}}, e \in \boldsymbol{E} \\ \hat{\lambda}_j - \hat{\lambda}_i + \phi_{r_e}(\hat{I}_e^+)L_e = 0 \ \forall\, e : \hat{I}_e^+ > 0 \\ \hat{\lambda}_i - \hat{\lambda}_j + \phi_{r_e}(\hat{I}_e^-)L_e = 0 \ \forall\, e : \hat{I}_e^- > 0 \\ \hat{\lambda}_i - \hat{\lambda}_j = 0 \ \forall\, e : \hat{I}_e^+ = \hat{I}_e^- = 0 \end{aligned}$$

which are precisely the potential loss equations in (NLNEQN($\boldsymbol{r}$)). Defining $(\boldsymbol{\pi}, \boldsymbol{I})$ as,

$$\begin{gathered} I_e = \hat{I}_e^+ - \hat{I}_e^- \ \forall\, e \in \boldsymbol{E} \\ \pi_i = \pi_i^{\text{src}} \ \forall\, i \in \boldsymbol{N}^{\text{src}} \quad \pi_i = \hat{\lambda}_i \ \forall\, i \in \boldsymbol{N} \setminus \boldsymbol{N}^{\text{src}}. \end{gathered} \tag{3.10}$$

it is easy to see from the above arguments and (3.8) that $(\boldsymbol{\pi}, \boldsymbol{I})$ satisfies (NLNEQN($\boldsymbol{r}$)).

Consider the *only if part* of the claim. Given a solution $(\boldsymbol{\pi}, \boldsymbol{I})$ to (NLNEQN($\boldsymbol{r}$)) define the following:

$$\begin{aligned} \hat{I}_e^+ = \max(0, I_e) \quad \hat{I}_e^- = \min(0, I_e) \ \forall\, e \in \boldsymbol{E} \\ \hat{\lambda}_i = \pi_i \ \forall\, i \ \in \boldsymbol{N} \setminus \boldsymbol{N}^{\text{src}} \\ \hat{\mu}_e^+ = \max(0, \pi_j - \pi_i + \phi_{r_e}(I_e^+)L_e) \ \forall\, e \in \boldsymbol{E} \\ \hat{\mu}_e^- = \max(0, \pi_i - \pi_j + \phi_{r_e}(I_e^-)L_e) \ \forall\, e \in \boldsymbol{E}. \end{aligned}$$

It can be verified that this choice satisfies the first order stationary conditions of (CVXNLP($\boldsymbol{r}$)). The strict monotonically increasing property of $\phi_r(\cdot)$ (Assumption (A1)) implies the strict convexity of $\Phi_r(\cdot)$. Since the constraints are linear we have that the optimization problem (CVXNLP($\boldsymbol{r}$)) is strictly convex. Consequently, the satisfaction of first-order stationary conditions are necessary and sufficient for $(\boldsymbol{\pi}, \boldsymbol{I})$ to be an optimal solution to (CVXNLP($\boldsymbol{r}$)). Further, the strict convexity of (CVXNLP($\boldsymbol{r}$)) implies that the solution is unique. The claim is proven. □

Theorem 3.1 shows that the network analysis problem (NLNEQN($\boldsymbol{r}$)) has a unique solution. The only assumption required is the strict monotonicity of the potential drop function (Assumption (A1)). As a consequence, at any integer feasible solution to the MINLP it is sufficient to solve the network analysis problem (NLNEQN($\boldsymbol{r}$)) and then, check for the satisfaction of bounds on the flows and potentials to verify feasibility of (NLP($\boldsymbol{r}$)). We state this in the following lemma.

LEMMA 3.2. *Suppose Assumption (A1) holds. Let,*

- *$\boldsymbol{r}$ be a choice of resistances for the edges*
- *$(\boldsymbol{\pi}, \boldsymbol{I})$ be the solution to* (NLNEQN($\boldsymbol{r}$)).

Then $(\boldsymbol{\pi}, \boldsymbol{I})$ is feasible for (NLP($\boldsymbol{r}$)) if it satisfies the bounds on flows and potentials.

Proof. From Theorem 3.1 we have that $(\boldsymbol{\pi}, \boldsymbol{I})$ is the unique solution to (NLNEQN($\boldsymbol{r}$)). The non-satisfaction of the bounds by this solution implies that (NLP($\boldsymbol{r}$)) is infeasible and the claim is proved. □

The convex program formulation (CVXNLP($\boldsymbol{r}$)) has a number of advantages over the nonlinear equation formulation:

1. If the potential drop function $\phi_r(\cdot)$ is sufficiently smooth the convex program is also smooth.
2. A unique solution exists since the optimization problem is strictly convex.
3. Further, an appropriate algorithm can guarantee that if (CVXNLP($\boldsymbol{r}$)) is feasible then a solution is found.

Given these advantages, it is surprising that this approach has not found favor over the conventional approach of solving (NLNEQN($\boldsymbol{r}$)). One of the drawbacks of the convex program is that it has a large number of bound constraints. The paper by Collins *et al.* [16] was published in the 1970's when there were few reliable algorithms for efficient solution of large-scale inequality constrained optimization problems. Significant advances have been made since then and has resulted in the development of interior-point algorithms for solving large-scale nonlinear non-convex programs [14, 33] that are both reliable and computationally efficient. Use of the convex programming approach in conjunction with the latest software can lead to more robust algorithms for network analysis. This needs to be explored further and is outside the scope of this paper.

4. MINLP Formulation with Convex Relaxation. The main obstacle in the global optimization of nonlinear networks are the non-convex potential loss constraints in (2.3). Observe also that these are the only nonlinear constraints in the problem. The formulation in (2.3) has two levels of disjunctions - the first level is over the set of resistance choices for each edge and the second is for the direction on that edge. One approach to deal with this is to distribute the inner disjunction which would result in $2n_r$ disjunctions at the same level for each of the edges. In this case, we will require $2n_r|\boldsymbol{E}|$ binary variables. We describe an alternate formulation that limits the number of binary variables of introduced to $(n_r+1)|\boldsymbol{E}|$.

We will introduce binary variables $x_{e,r} \in \{0,1\} \; \forall \; e \in \boldsymbol{E}, r = 1, \ldots, n_r$ to model the choice of resistance r on edge e (outer disjunction), where $x_{e,r} = 1$ implies choosing resistance r. Using these variables, the requirement that only one resistance be chosen on each edge can be posed as:

$$\left.\begin{array}{c} \sum\limits_{r=1}^{n_r} x_{e,r} = 1 \\ x_{e,r} \in \{0,1\} \; \forall \; r = 1, \ldots, n_r \end{array}\right\} \; \forall \; e \in \boldsymbol{E}. \tag{4.1}$$

The binary variables x_e^{dir} will denote the direction of flow on edge $e = (i,j)$ between nodes i and j, where $x_e^{\text{dir}} = 1$ represents the flow from i to j and $x_e^{\text{dir}} = 0$ vice-versa. The variables $I_{e,r}^+, I_{e,r}^-$ denote variables modeling the non-negative flow that occurs on edge e with resistance r contingent on flow direction. The variable $I_{e,r}^+ (I_{e,r}^-)$ models the flow from i to j (j to i). The relation between the flows and binary variables can be represented as:

$$\begin{array}{ccc} & x_e^{\text{dir}} \in \{0,1\} \; \forall \, e \in \boldsymbol{E} & \\ \left.\begin{array}{cc} I_{e,r}^{+} \geq 0 & I_{e,r}^{-} \geq 0 \\ I_{e,r}^{+} \leq I_{e,r}^{\max} x_{e,r} & I_{e,r}^{-} \leq I_{e,r}^{\max} x_{e,r} \\ I_{e,r}^{+} \leq I_{e,r}^{\max} x_e^{\text{dir}} & I_{e,r}^{-} \leq I_{e,r}^{\max}(1 - x_e^{\text{dir}}) \end{array}\right\} & \forall \begin{array}{c} r = 1, \ldots, n_r, \\ e \in \boldsymbol{E} \end{array} & . \end{array} \tag{4.2}$$

The constraints in (4.2) ensure that only the flow variables associated with chosen resistance and direction on that edge can be non-zero. We model the potential drop on each edge e based on the direction flow using variables, $\Delta\pi_e^+, \Delta\pi_e^-$. These variables can be related to the binary variables as follows:

$$\left.\begin{array}{l} 0 \leq \Delta\pi_e^+ \leq \Delta\pi_e^{\max,+} x_e^{\text{dir}} \\ 0 \leq \Delta\pi_e^- \leq \Delta\pi_e^{\max,-}(1 - x_e^{\text{dir}}) \\ \pi_i - \pi_j = \Delta\pi_e^+ - \Delta\pi_e^- \end{array}\right\} \forall \, e \in \boldsymbol{E} \tag{4.3}$$

where $\Delta\pi_e^{\max,+} := \pi_i^{\max} - \pi_j^{\min}$ and $\Delta\pi_e^{\max,-} := \pi_j^{\max} - \pi_i^{\min}$ are the maximum potential loss allowable in the appropriate flow direction on the edge e of the network. The potential loss constraints are relaxed as follows:

$$\left.\begin{array}{l} \dfrac{\Delta\pi_e^+}{L_e} \geq \phi_r(I_{e,r}^+) \\ \dfrac{\Delta\pi_e^-}{L_e} \geq \phi_e(I_{e,r}^-) \end{array}\right\} \forall \begin{array}{c} r = 1, \ldots, n_r, \\ e \in \boldsymbol{E} \end{array} \tag{4.4}$$

which requires that the potential loss on the edges be at least that specified by the potential loss function. From Assumption (A2) it follows that (4.4) is convex. This will be a key property that we will exploit in devising the solution algorithm. Finally, we add an additional constraint to limit the potential loss on the edge based on the flow as,

$$\begin{array}{c} \left.\begin{array}{l} \frac{\Delta\pi_e^+}{L_e} \leq \sum\limits_{r=1}^{n_r} (\text{slope}(\phi_r; 0, I_{e,r}^{\max})) I_{e,r}^+ \\ \frac{\Delta\pi_e^-}{L_e} \leq \sum\limits_{r=1}^{n_r} (\text{slope}(\phi_r; 0, I_{e,r}^{\max})) I_{e,r}^- \end{array}\right\} \forall \, e \in \boldsymbol{E} \\ \text{where } \text{slope}(\phi_r; I_1, I_2) := \dfrac{\phi_r(I_2) - \phi_r(I_1)}{I_2 - I_1}. \end{array} \tag{4.5}$$

Observe that when $x_{e,r} = x_e^{\text{dir}} = 1$ the constraints in (4.3) imply that $\Delta\pi_e^- \leq 0$ and $\Delta\pi_e^+/L_e \leq \text{slope}(\phi_r; 0, I_{e,r}^{\max}) I_{e,r}^+$. We show in Lemma 4.1 that $\text{slope}(\phi_r; 0, I_{e,r}^{\max}) I \geq \phi_r(I) \; \forall \, I \geq 0$ which implies that (4.5) is a *valid over-estimator* of the potential loss function.

Utilizing the convex relaxation of the potential loss constraints in (4.4) we can pose the relaxation of the nonlinear network design problem as a MINLP with convex nonlinear constraints

$$
\begin{aligned}
&\min \sum_{e\in \boldsymbol{E}} \sum_{r=1}^{n_r} c_r L_e x_{e,r} \\
&\text{s.t.} \sum_{e\in \boldsymbol{E}_2(i)} \sum_{r=1}^{n_r} (I^+_{e,r} - I^-_{e,r}) - \sum_{e\in \boldsymbol{E}_1(i)} \sum_{r=1}^{n_r} (I^+_{e,r} - I^-_{e,r}) = I^{\rm dem}_i \ \forall\, i \in \boldsymbol{N} \setminus \boldsymbol{N}^{\rm src} \\
&\quad \text{Eq. (4.1)} - (4.5) \\
&\quad \pi^{\min}_i \le \pi_i \le \pi^{\max}_i \ \forall\, i \in \boldsymbol{N} \setminus \boldsymbol{N}^{\rm src} \\
&\quad \pi_i = \pi^{\rm src}_i \ \forall\, i \in \boldsymbol{N}^{\rm src}
\end{aligned} \tag{4.6}
$$

The MINLP in (4.6) is a relaxation of the nonlinear network design problem. We state the following results about the MINLP in (4.6).

LEMMA 4.1. *Suppose Assumptions (A1) and (A2) hold. Let* $\mathcal{F}(2.3)$ *and* $\mathcal{F}(4.6)$ *denote respectively the feasible region of optimization problems* (2.3) *and* (4.6)*. Then,*

1. *The MINLP in* (4.6) *has a convex relaxation.*
2. $\phi_r(I) \le \mathrm{slope}(\phi_r; 0, I^{\max}_{e,r}) I \ \forall\, I \in [0, I^{\max}_{e,r}], e \in \boldsymbol{E}, r \in \{1, \ldots, n_r\}$
3. $\mathcal{F}(2.3) \subseteq \mathcal{F}(4.6)$.

Proof. Assumption (A2) implies that (4.4) is convex. Since the rest of the constraints are linear the MINLP in (4.6) has a convex relaxation. Consider the second claim. For any $e \in \boldsymbol{E}, r \in \{1, \ldots, n_r\}$ we have that

$$
\mathrm{slope}(\phi_r; 0, I^{\max}_{e,r}) I = \frac{\phi_r(I^{\max}_{e,r})}{I^{\max}_{e,r}} I = \begin{cases} 0 \text{ for } I = 0 \text{ (since } \phi_r(0) = 0) \\ \phi_r(I^{\max}_{e,r}) \text{ for } I = I^{\max}_{e,r}. \end{cases}
$$

In other words, the right hand side of $\mathrm{slope}(\phi_r; 0, I^{\max}_{e,r}) I$ coincides with $\phi_r(I)$ at the end points of the interval $[0, I^{\max}_{e,r}]$. Suppose there exists $\hat{I} \in (0, I^{\max}_{e,r})$ such that $\phi_r(\hat{I}) > \mathrm{slope}(\phi_r; 0, I^{\max}_{e,r}) \hat{I}$. Then,

$$
\begin{aligned}
\phi_r(\hat{I}) = \phi_r\left(\left(1 - \tfrac{\hat{I}}{I^{\max}_{e,r}}\right) 0 + \left(\tfrac{\hat{I}}{I^{\max}_{e,r}}\right) I^{\max}_{e,r} \right) &> \tfrac{\phi_r(I^{\max}_{e,r})}{I^{\max}_{e,r}} \hat{I} \\
\implies \phi_r\left(\left(1 - \tfrac{\hat{I}}{I^{\max}_{e,r}}\right) 0 + \left(\tfrac{\hat{I}}{I^{\max}_{e,r}}\right) I^{\max}_{e,r} \right) &> \left(1 - \tfrac{\hat{I}}{I^{\max}_{e,r}}\right) \phi_r(0) + \left(\tfrac{\hat{I}}{I^{\max}_{e,r}}\right) \phi_r(I^{\max}_{e,r}).
\end{aligned}
$$

The last inequality violates the assumption of convexity of $\phi_r(\cdot)$ (A2) and this proves the claim. Consider the third claim. Let $\boldsymbol{r}$ be a set of resistance choices on each edge of the network and $(\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{I}})$ be the solution of the corresponding network analysis problem (NLNEQN($\boldsymbol{r}$)). In other words, $(\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{I}})$ is a feasible solution of (2.3). Define for each $e \in \boldsymbol{E}$ and $r \in 1, \ldots, n_r$,

$$
\begin{aligned}
&\hat{x}_{e,r} = \begin{cases} 1 \text{ if } r = r_e \\ 0 \text{ otherwise.} \end{cases} \quad \hat{x}^{\rm dir}_e = \begin{cases} 1 & \text{if } \sum_{r=1}^{n_r} \hat{I}_{e,r} \ge 0 \\ 0 & \text{if } \sum_{r=1}^{n_r} \hat{I}_{e,r} < 0 \end{cases} \\
&\hat{I}^+_{e,r} = \max(0, \hat{I}_{e,r}) \quad \hat{I}^-_{e,r} = \max(0, -\hat{I}_{e,r}) \\
&\Delta\hat{\pi}^+_e = \max(0, \hat{\pi}_i - \hat{\pi}_j) \quad \Delta\hat{\pi}^-_e = \max(0, \hat{\pi}_j - \hat{\pi}_i)
\end{aligned} \tag{4.7}
$$

Denoting by $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{x}}^{\rm dir}$ respectively the set of all binary variables representing the resistance choices and flow directions on all edges we have that, ($\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{x}}^{\rm dir}$, $\hat{\boldsymbol{\pi}}$, $\Delta\hat{\boldsymbol{\pi}}^+$, $\Delta\hat{\boldsymbol{\pi}}^-$, $\hat{\boldsymbol{I}}^+$, $\hat{\boldsymbol{I}}^-$) is feasible to (4.6). This proves the claim. ☐

A standard MINLP algorithm applied to (4.6) will result in the algorithm solving a *relaxation* of network analysis problem at integer feasible nodes (Lemma 4.1). To rectify this, we will need to modify the problem that is solved at the integer feasible nodes of the search tree. Observe that in the optimization problem (2.3) the objective function does not depend on the flow and potential variables. Consequently, the relaxation can be addressed by solving (NLNEQN($\boldsymbol{r}$)) or (CVXNLP($\boldsymbol{r}$)) at the integer feasible nodes in the search tree of a branch-and-bound (BB) or linearizations based MILP (LP/NLP-BB) algorithm. We will restrict the focus of this paper to the case of the LP/NLP-BB algorithm.

5. Linearizations Based MINLP Algorithm. The LP/NLP-BB algorithm was first introduced by Quesada and Grossmann [29] and subsequently investigated by Abhishek *et al* [4]. The LP/NLP-BB algorithm uses a Mixed Integer Linear Program (MILP) which is derived by linearizing the convex nonlinear constraints at different values of the continuous variables. The linearizations of the nonlinear constraints are added to the problem as cuts during the search process. The LP/NLP-BB approach has been implemented in software such as `Bonmin` [10] and `FilMINT` [4]. The computational efficiency of the approach has been demonstrated on several test problems [4].

For ease of exposition, we will denote the set of all continuous and binary variables as $\boldsymbol{y} := (\boldsymbol{x}, \boldsymbol{x}^{\text{dir}}, \boldsymbol{\pi}, \Delta\boldsymbol{\pi}^+, \Delta\boldsymbol{\pi}^-, \boldsymbol{I}^+, \boldsymbol{I}^-,)$ and by $\boldsymbol{y}^l, \boldsymbol{y}^u$ the lower and upper bounds on the variables $\boldsymbol{y}$. Let J represent the set of indices of $\boldsymbol{y}$ that correspond to the binary variables, namely $J = \{1, \ldots, |\boldsymbol{E}|(1 + n_r)\}$. All the linear equality and linear inequality constraints including bounds in (4.6) will be represented collectively as:

$$\begin{aligned} A^{\text{eq}}(\boldsymbol{y}) &= b^{\text{eq}} \\ A^{\text{ineq}}(\boldsymbol{y}) &\leq b^{\text{ineq}}. \end{aligned} \tag{5.1}$$

The nonlinear inequality constraints in (4.4) will be represented compactly as:

$$\left.\begin{aligned} g^+_{e,r}(\boldsymbol{y}) &:= -\frac{\Delta\pi^+_e}{L_e} + \phi_r(I^+_{e,r}) \\ g^-_{e,r}(\boldsymbol{y}) &:= -\frac{\Delta\pi^-_e}{L_e} + \phi_r(I^-_{e,r}) \end{aligned}\right\} \forall \quad \begin{array}{c} r = 1, \ldots, n_r \\ e \in \boldsymbol{E}. \end{array} \tag{5.2}$$

Using the above notation, we will define the master MILP associated with (4.6) in the context of the LP/NLP-BB algorithm as:

$$\begin{array}{ll} \min & \eta \\ \text{s.t.} & \eta \geq \sum\limits_{e \in \boldsymbol{E}} \sum\limits_{r=1}^{n_r} c_r L_e x_{e,r} \\ & A^{\text{eq}}(\boldsymbol{y}) = b^{\text{eq}} \\ & A^{\text{ineq}}(\boldsymbol{y}) \leq b^{\text{ineq}} \\ & \boldsymbol{y}^l \leq \boldsymbol{y} \leq \boldsymbol{y}^u \\ & \boldsymbol{y} \in \{0,1\}^{|J|} \\ & \boldsymbol{L}g^+_{e,\hat{r}}(\boldsymbol{y}; \hat{I}) \leq 0 \ \forall \ (\hat{I}, \hat{r}) \in \mathcal{OA}^+_e, e \in \boldsymbol{E} \\ & \boldsymbol{L}g^-_{e,\hat{r}}(\boldsymbol{y}; \hat{I}) \leq 0 \ \forall \ (\hat{I}, \hat{r}) \in \mathcal{OA}^-_e, e \in \boldsymbol{E} \end{array} \tag{mMILP}$$

where η has been introduced to represent the value of objective function. Further, we will use the notation $LP(\boldsymbol{y}^l, \boldsymbol{y}^u)$ to denote the linear program obtain by relaxing the integral requirement on $\boldsymbol{y}_i \ \forall \ i \in J$. The branch & cut search process solves a sequence of LP relaxations at nodes of the search tree obtained by fixing a subset of the binary variables to either 0 or 1. When binary variables are fixed to a particular value it is assumed that $y_i^l = y_i^u$.

In (mMILP), $\boldsymbol{L}g^+_{e,\hat{r}}(\boldsymbol{y};\hat{I})$ represents a linearization of the convex nonlinear constraint $g^+_{e,\hat{r}}(\boldsymbol{y})$ at the point $\hat{I}$. We have parameterized the linearization by only the flow $\hat{I}$ since the potential loss function is nonlinear in the flow alone. We will postpone presenting the particular form of the linearization until Section 5.1. Though (mMILP) indicates only one type of cuts it is also possible to add other valid inequalities as also described in [4].

The set $\mathcal{OA}^+_e$ (set $\mathcal{OA}^-_e$) represent respectively the set of flow and resistance index, $(\hat{I},\hat{r})$ at which the inequality constraints $g^+_{e,\hat{r}}(\cdot)$ ($g^-_{e,\hat{r}}(\cdot)$) are linearized. For instance, one can a priori populate the sets $\mathcal{OA}^+_e, \mathcal{OA}^-_e$ with a number of points for each resistance choice, $r \in 1,\ldots,n_r$ and edge, $e \in \boldsymbol{E}$. This will amount to providing an approximation to the convex hull of each of the nonlinear constraints $g^+_{e,r}(\boldsymbol{y})$ and $g^-_{e,r}(\boldsymbol{y})$. Determining a choice of points *a priori* is a difficult task for general MINLPs. The LP/NLP-BB algorithm addresses this by initially populating the sets with cuts derived from linearizing at the solution of the continuous relaxation of (4.6). Namely, the algorithm solves

$$\begin{array}{ll} \min & \eta \\ \text{s.t.} & \text{all constraints in (4.6) with binary requirement} \\ & \text{replaced by } \boldsymbol{y}_i \in [0,1]\ \forall\, i \in J \\ & \boldsymbol{y}^l \le \boldsymbol{y} \le \boldsymbol{y}^u. \end{array} \qquad (\text{RNLP}(\boldsymbol{y}^l,\boldsymbol{y}^u))$$

and adds linearizations of the nonlinear constraints at the solution of $(\text{RNLP}(\boldsymbol{y}^l,\boldsymbol{y}^u))$. The set of points in $\mathcal{OA}^+_e, \mathcal{OA}^-_e$ are augmented with points that are obtained from the solution to (mMILP) at various nodes in the search tree. To enable this, the MILP solver implementation must allow adding cuts at intermediate nodes in the search tree by invoking a user-defined cut generation routine. Note that the cuts added in this manner may alter the feasible region as defined by the root node MILP problem. We will now outline a modification to the standard LP/NLP-BB framework which allows to obtain the globally optimal solution to (2.3). We make the following assumption on the MILP solver.

Assumption:

(S1) The MILP solver has the flexibility to call a *user-defined cut generation routine* after the solution of the *Linear Program (LP)* obtained by relaxing the integral constraints at every node in the search tree.

(S2) If cuts are added then the MILP solver will re-solve the LP at the node and return control back to the cut-generation routine. If no cuts are added the MILP solver proceeds uninterrupted.

We present the LP/NLP-BB algorithm in Algorithm 1. In the algorithm, we use $(\text{CVXNLP}(\boldsymbol{r}))$ to determine the set of potentials and flows $(\boldsymbol{\pi}^{\text{nlp}}, \boldsymbol{I}^{\text{nlp}})$ that solves the network analysis equations in $(\text{NLNEQN}(\boldsymbol{r}))$. Theorem 3.1 shows how the solution to $(\text{NLNEQN}(\boldsymbol{r}))$ can be derived from solution of $(\text{CVXNLP}(\boldsymbol{r}))$. Algorithm 1 differs from the standard LP/NLP-BB algorithm such as `FilMINT` [4] in the following aspects:

(i) In the standard LP/NLP-BB algorithm, the NLP solved at the integral node is derived by fixing the integer variables in MINLP to the appropriate values. Lemma 4.1 shows that such an approach will result in solving a relaxation to $(\text{CVXNLP}(\boldsymbol{r}))$ at the integral nodes. Consequently, we instead solve

Solve (RNLP($\boldsymbol{y}^l, \boldsymbol{y}^u$)). Let $(\hat{\eta}, \hat{\boldsymbol{y}})$ denote the objective and solution obtained.
for $e \in \boldsymbol{E}$ **do**
  Let $r^{\max} := \arg\max_r \phi_r(\hat{I}^+_{e,r})$. Set $\mathcal{OA}^+_e = \mathcal{OA}^+_e \cup (\hat{I}^+_{e,r^{\max}}, r^{\max})$.
  Let $r^{\max} := \arg\max_r \phi_r(\hat{I}^-_{e,r})$. Set $\mathcal{OA}^-_e = \mathcal{OA}^-_e \cup (\hat{I}^-_{e,r^{\max}}, r^{\max})$.
end for
Set **UserCuts** as user-defined cut generation routine in the **MILP** solver
Apply **MILP** solver to master problem (mMILP)

UserCuts routine:
Data: Let, s represent the node number,
d denote depth of the node in the search tree,
η^{best} denote the objective value of the best feasible solution found thus far.
$(\eta^s, \boldsymbol{y}^s)$ represent the objective value and solution of the *LP* at the node s
$iter^{\max}$ represent maximum number of iterations for **Repair** procedure
if $(\boldsymbol{x}^s, \boldsymbol{x}^{\text{dir},s})$ *is integral* **then** `/* integral node */`
  Define the set of resistances $\boldsymbol{r} \in \mathbb{R}^{|\boldsymbol{E}|}$ on the edges as $r_e : x^s_{e,r_e} = 1$.
  Solve (CVXNLP($\boldsymbol{r}$)) with $\boldsymbol{r} = \boldsymbol{r}^s$ to obtain $(\boldsymbol{\pi}^{\text{nlp}}, \boldsymbol{I}^{\text{nlp}})$, that solves network analysis (NLNEQN($\boldsymbol{r}$)).
  if $\boldsymbol{\pi}^{\min} \le \boldsymbol{\pi}^{\text{nlp}} \le \boldsymbol{\pi}^{\max}$ *and* $-\boldsymbol{I}^{\max} \le \boldsymbol{I}^{\text{nlp}} \le \boldsymbol{I}^{\max}$ **then**
    $feasSoln = true$
  else
    $feasSoln = false$
  end if
  if *not(feasSoln)* **then** `/* infeasible choice r */`
    Fathom the choice of resistances using the following cut

$$\sum_{e,r:x^s_{e,r}=1} x_{e,r} - \sum_{e,r:x^s_{e,r}=0} x_{e,r} \le |\boldsymbol{E}| - 1. \tag{5.3}$$

    $(repaired, \boldsymbol{r}^{\text{rep}}) =$ **Repair** $(\boldsymbol{r}, iter^{\max}, \eta^{\text{best}})$
    if *repaired* **then** `/* found better solution */`
      Solve (CVXNLP($\boldsymbol{r}$)) with $\boldsymbol{r} = \boldsymbol{r}^{rep}$ to obtain $(\boldsymbol{\pi}^{\text{nlp}}, \boldsymbol{I}^{\text{nlp}})$, that solves network analysis (NLNEQN($\boldsymbol{r}$)).
      Compute $\boldsymbol{y}$ from $(\boldsymbol{r}^{\text{rep}}, \boldsymbol{\pi}^{\text{nlp}}, \boldsymbol{I}^{\text{nlp}})$ as shown in (4.7).
      Set $\boldsymbol{y}$ as best solution and $\eta^{\text{best}} := \sum_{e \in \boldsymbol{E}} c_{r^{\text{rep}}_e} L_e$.
    end if
  end if
  if $feasSoln$ *or repaired* **then** `/* augment cuts */`
    for $e \in \boldsymbol{E}$ **do**
      $\mathcal{OA}^+_e = \mathcal{OA}^+_e \cup (I^{\text{nlp}}_e, r_e)\ \forall\ e : I^{\text{nlp}}_e \ge 0$
      $\mathcal{OA}^-_e = \mathcal{OA}^-_e \cup (|I^{\text{nlp}}_e|, r_e) \forall\ e : I^{\text{nlp}}_e < 0$
    end for
  end if
else `/* non-integral node */`
  Call **NodeCuts** $(\eta^{s-1}, \eta^s, d, \boldsymbol{y}^s)$
end if

Algorithm 1: Modified LP/NLP-BB algorithm for the global optimization of MINLP (2.3)

(CVXNLP($\boldsymbol{r}$)) using the integral values of the resistances determined at the node.

(ii) Additionally, at integral nodes we ignore the choice of the direction variables. Section 5.2 shows that this approach is correct and can improve the computational efficiency of the algorithm.

(iii) At an integral node s where the solution to the network analysis problem (NLNEQN($\boldsymbol{r}$)) $(\boldsymbol{\pi}^{\text{nlp}}, \boldsymbol{I}^{\text{nlp}})$ violates the bounds, the cut (5.3) invalidates the particular choice of resistances regardless of the choice of directions on the edges. The standard LP/NLP-BB algorithm invalidates the choice of resistances and direction variables at the node s,

$$\sum_{e,r:x^s_{e,r}=1} x_{e,r} - \sum_{e,r:x^s_{e,r}=0} x_{e,r} + \sum_{e:x^{\text{dir},s}_e=1} x^{\text{dir}}_e - \sum_{e:x^{\text{dir},s}_e=0} x^{\text{dir}}_e \leq \begin{array}{l} |\{e|x^{\text{dir},s}_e = 1\}| \\ +|\boldsymbol{E}| - 1. \end{array} \tag{5.4}$$

The cut in (5.3) is *stronger* and its validity will be shown in Lemma 5.5.

(iv) At an integral node s where the solution to the network analysis problem (NLNEQN($\boldsymbol{r}$)) $(\boldsymbol{\pi}^{\text{nlp}}, \boldsymbol{I}^{\text{nlp}})$ violates the bounds, we call **Repair** procedure to find a feasible solution in the neighborhood of the current resistance choices. If successful the procedure returns a solution with lower objective than the best feasible solution found till then. We postpone the description of the **Repair** procedure until Section 6.

(v) Another crucial distinction is that `FilMINT` manages the number of cuts in the search tree. The algorithm in [4] periodically removes linearizations that have not figured in the active set of the LP relaxations over a certain number of successive nodes. This can aid the algorithm in managing memory requirements when solving MINLPs. Our implementation does not perform this cut management.

We will assume for now that the subroutine **NodeCuts** provides linear cuts that are valid for (4.6). Algorithm 2 describes the linearization procedure. The implementation follows along the lines described in `FilMINT` [4]. The addition of cuts is controlled by three parameters: K_{oa}, β_{oa} and M_{oa}. For each node s we check for satisfaction of condition in step 5 of the algorithm. The first clause in the condition favors addition of cuts at the top of the search tree while branching is favored deeper in the tree. The parameter β_{oa} is the average number of nodes at which cuts are added for a given depth in the search tree. The second clause avoids adding cuts when the change in objective value between successive nodes is below K_{oa}. The number of rounds of cuts for each node is limited by the third clause using M_{oa}. In addition, the algorithm checks for each edge e the value of the direction variable x^{dir}_e at the current node. If $x^{\text{dir}}_e \geq 0.5$, this is an indication that the algorithm prefers the positive flow direction for this node. We add the linearization for the positive flow direction provided the maximal constraint violation, $\max_r g^+_{e,r}(I^+_{e,r})$ is greater than ϵ. If $x^{\text{dir}}_e < 0.5$ then linearization for the negative flow direction is added. We have found that this algorithm provides a good balance between adding cuts and branching in the search tree. The cuts that are added critically determine the efficiency of the LP/NLP-BB algorithm. Details on the form of the cuts are provided next.

5.1. Linearizations of Nonlinear Inequalities. The linearizations that are typically added in standard LP/NLP-BB algorithms for some choice of resistance

NodeCuts$(\eta^{\text{prev}}, \eta, d, \boldsymbol{y})$

Data: ϵ - minimum violation for constraint linearization,
β_{oa} - positive coefficient controlling number of cuts added at depth d,
K_{oa} - minimum improvement in objective over successive nodes,
M_{oa} - maximum number of rounds at each node.

Set $a \in [0, 1]$ be a uniformly generated random number.
if $a \leq \beta_{oa} 2^{-d}$ *and* $\frac{\eta - \eta^{\text{prev}}}{\eta^{\text{prev}}} \geq K_{oa}$ *and* $n \leq M_{oa}$ **then** `/* outer-approximation */`
    for $e \in \boldsymbol{E}$ **do**
        if $x_e^{\text{dir}} \geq 0.5$ **then** `/* flow from` i `to` j `*/`
            Set $r^{\max} := \arg\max_{r \in 1,\ldots,n_r} \phi_r(I^+_{e,r})$
            if $g^+_{e,r^{\max}}(I^+_{e,r^{\max}}) > \epsilon$ **then**
                Set $\mathcal{OA}^+_e = \mathcal{OA}^+_e \cup \{(I^+_{e,r^{\max}}, r^{\max})\}$
            end if
        else `/* flow from` j `to` i `*/`
            Set $r^{\max} := \arg\max_{r \in 1,\ldots,n_r} \phi_r(I^-_{e,r})$
            if $g^-_{e,r^{\max}}(I^-_{e,r^{\max}}) > \epsilon$ **then**
                Set $\mathcal{OA}^-_e = \mathcal{OA}^-_e \cup \{(I^-_{e,r^{\max}}, r^{\max})\}$
            end if
        end if
    end for
    Set $n = n + 1$.
end if

Algorithm 2: Algorithm for adding cuts at the nodes of the LP/NLP-BB search tree.

$\hat{r} \in \{1, \ldots, n_r\}$ and flow $\hat{I}$ take the form:

$$\begin{aligned} \tilde{\boldsymbol{L}}g^+_{e,\hat{r}}(\boldsymbol{y}; \hat{I}) &:= -\frac{\Delta\pi^+_e}{L_e} + \phi_{\hat{r}}(\hat{I}) + \phi'_{\hat{r}}(\hat{I})(I^+_{e,\hat{r}} - \hat{I}) \\ \tilde{\boldsymbol{L}}g^-_{e,\hat{r}}(\boldsymbol{y}; \hat{I}) &:= -\frac{\Delta\pi^-_e}{L_e} + \phi_{\hat{r}}(\hat{I}) + \phi'_{\hat{r}}(\hat{I})(I^-_{e,\hat{r}} - \hat{I}) \end{aligned} \tag{5.5}$$

where $\phi'_r(\cdot)$ denotes the first derivative of the potential loss function for resistance choice r. The above linearization can be strengthened using the binary variables for flow direction as:

$$\begin{aligned} \hat{\boldsymbol{L}}g^+_{e,\hat{r}}(\boldsymbol{y}; \hat{I}) &:= -\frac{\Delta\pi^+_e}{L_e} + \phi'_{\hat{r}}(\hat{I})I^+_{e,\hat{r}} + \left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right) x_e^{\text{dir}} \\ \hat{\boldsymbol{L}}g^-_{e,\hat{r}}(\boldsymbol{y}; \hat{I}) &:= -\frac{\Delta\pi^-_e}{L_e} + \phi'_{\hat{r}}(\hat{I})I^-_{e,\hat{r}} + \left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right)(1 - x_e^{\text{dir}}). \end{aligned} \tag{5.6}$$

The following lemma shows that (5.6) is indeed stronger than (5.5). We prove the result for a particular edge e, resistance r and the positive flow direction. A similar result can be proved for the case of negative flow direction.

LEMMA 5.1. *Suppose Assumptions (A1) and (A2) hold. Define,*

$$\begin{aligned} \mathcal{F}_0 &:= \left\{ \boldsymbol{y} \,\middle|\, \Delta\pi^+_e, I^+_{e,r} \geq 0, x_e^{\text{dir}} \in [0,1]\ \tilde{\boldsymbol{L}}g^+_{e,r}(\boldsymbol{y}; \hat{I}) \leq 0 \right\} \\ \mathcal{F}_1 &:= \left\{ \boldsymbol{y} \,\middle|\, \Delta\pi^+_e, I^+_{e,r}, \geq 0, x_e^{\text{dir}} \in [0,1]\ \hat{\boldsymbol{L}}g^+_{e,r}(\boldsymbol{y}; \hat{I}) \leq 0 \right\}. \end{aligned} \tag{5.7}$$

Then, $\mathcal{F}_1 \subset \mathcal{F}_0$.

Proof. From the convexity of $\phi_r(\cdot)$ we have that

$$\phi_r(\hat{I}) - \phi_r'(\hat{I})\hat{I} \le \phi_r(0) = 0 \tag{5.8}$$

where the equality follows from assumption (A1). Since $x^{\text{dir}} \in [0,1]$ we have from (5.8) that

$$\phi_r(\hat{I}) - \phi_r'(\hat{I})\hat{I} \le (\phi_r(\hat{I}) - \phi_r'(\hat{I})\hat{I})x_e^{\text{dir}}. \tag{5.9}$$

Suppose $\overline{\boldsymbol{y}} \in \mathcal{F}_1$ with potentials and flows $(\Delta\overline{\boldsymbol{\pi}}, \overline{\boldsymbol{I}})$ then,

$$\begin{aligned}\tilde{\boldsymbol{L}}g_{e,r}^{+}(\overline{\boldsymbol{y}};\hat{I}) &= -\frac{\Delta\overline{\pi}_e^{+}}{L_e} + \phi_r'(\hat{I})\overline{I}_{e,r}^{+} + \left(\phi_r(\hat{I}) - \phi_r'(\hat{I})\hat{I}\right) \\ &\le -\frac{\Delta\overline{\pi}_e^{+}}{L_e} + \phi_r'(\hat{I})\overline{I}_{e,r}^{+} + \left(\phi_r(\hat{I}) - \phi_r'(\hat{I})\hat{I}\right)x_e^{\text{dir}} \ \le \ 0\end{aligned} \tag{5.10}$$

where the last inequality follows from the assumption that $\boldsymbol{y} \in \mathcal{F}_1$. Hence, $\overline{\boldsymbol{y}} \in \mathcal{F}_0$. To show the strict inclusion, consider $\overline{\boldsymbol{y}}$ such that $\overline{I}_{e,r}^{+} > 0$, $(\Delta\overline{\pi}_e^{+}/L_e) = \phi_r'(\hat{I})\overline{I}_{e,r}^{+} + (\phi_r(\hat{I}) - \phi_r'(\hat{I})\hat{I})$ and $x_e^{\text{dir}} \in (0,1)$. For this choice, we have that

$$\begin{aligned}&-\frac{\Delta\overline{\pi}_e^{+}}{L_e} + \phi_r'(\hat{I})\overline{I}_{e,r}^{+} + \left(\phi_r(\hat{I}) - \phi_r'(\hat{I})\hat{I}\right)x_e^{\text{dir}} \\ &= -\left(\phi_r(\hat{I}) - \phi_r'(\hat{I})\hat{I}\right) + \left(\phi_r(\hat{I}) - \phi_r'(\hat{I})\hat{I}\right)x_e^{\text{dir}} \ge 0\end{aligned}$$

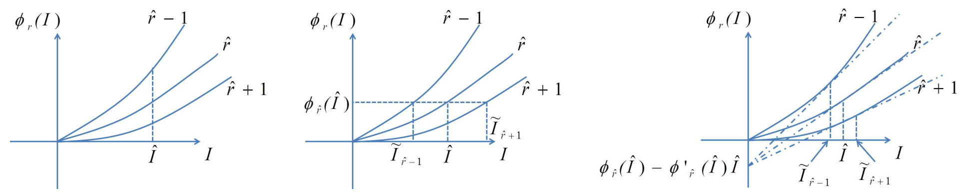
where the last inequality follows from (5.9). Hence, we have that $\overline{\boldsymbol{y}} \in \mathcal{F}_0, \notin \mathcal{F}_1$ for $x_e^{\text{dir}} \in (0,1)$. Hence, $\mathcal{F}_1 \subset \mathcal{F}_0$. ☐

The inequality in (5.6) is a linearization of the nonlinear potential loss constraint associated with one of the resistances. Typically a number of inequalities will be required to approximate the convex hull. Since this provides no linearization for other resistances choices on that edge, one can consider adding linearizations for each resistance choice on the edge. Suppose for the edge, the potential loss constraint $g_{e,\hat{r}}^{+}(\cdot)$ associated with $\hat{r}$ is linearized around $\hat{I}$, then consider adding the set of inequalities

$$\begin{aligned}\hat{\boldsymbol{L}}g_{e,\hat{r}}^{+}(\boldsymbol{y};\hat{I}) &\le 0 \\ \hat{\boldsymbol{L}}g_{e,r}^{+}(\boldsymbol{y};\tilde{I}_r) &\le 0 \ r \ne \hat{r}\end{aligned} \tag{5.11}$$

where $\tilde{I}_r$ is an appropriately chosen flow value. There exist several choices, three of which we list below: (i) same flow $\tilde{I}_r = \hat{I}$, (ii) flow at which potential loss is equal $\tilde{I}_r : \phi_r(\tilde{I}_r) = \phi_{\hat{r}}(\hat{I})$, (iii) flow at which linearization has same intercept $\tilde{I}_r$: $\phi_r(\tilde{I}_r) - \phi_r'(\tilde{I}_r)\tilde{I}_r = \phi_{\hat{r}}(\hat{I}) - \phi_{\hat{r}}'(\hat{I})\hat{I}$. A geometric depiction of the different choices are provided in Figure 5.1.

Addition of multiple cuts as in (5.11) can adversely affect the computational times. It will be desirable if a single cut can be derived which is stronger than the set of cuts in (5.11). To this end, consider a linear combination of the multiple cuts



(a) Equal flows for all resistance choices on an edge (b) Equal potential for all resistance choices on an edge (c) Equal intercept for linearization obtained for all resistance choices on edge

Fig. 5.1. *Geometric depiction of the possible choices for the flows at which the nonlinear constraints are linearized for resistance indices - $\hat{r}$ and $\hat{r}-1$, $\hat{r}+1$.*

where cut corresponding to each resistance is multiplied by $x_{e,r}$ as follows,

$$
\begin{aligned}
& x_{e,\hat{r}}\hat{\boldsymbol{L}}g^{+}_{e,\hat{r}}(y;\hat{I}) + \sum_{r\neq\hat{r}} x_{e,r}\hat{\boldsymbol{L}}g^{+}_{e,r}(\boldsymbol{y};\tilde{I}_r) \leq 0 \\
\implies & \left\{ \begin{aligned} & -\sum_{r=1}^{n_r} x_{e,r}\frac{\Delta\pi^{+}_{e}}{L_e} + x_{e,\hat{r}}\phi'_{\hat{r}}(\hat{I})I^{+}_{e,\hat{r}} + \sum_{r\neq\hat{r}} x_{e,r}\phi'_{r}(\tilde{I}_r)I^{+}_{e,r} \\ & + x_{e,\hat{r}}\left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right)x^{\mathrm{dir}}_{e} + \sum_{r\neq\hat{r}} x_{e,r}\left(\phi_{r}(\tilde{I}_r) - \phi'_{r}(\tilde{I}_r)\tilde{I}_r\right)x^{\mathrm{dir}}_{e} \leq 0. \end{aligned} \right.
\end{aligned} \tag{5.12}
$$

Since we have non-zero flow on an edge only when $x_{e,r}=1$ and since only one of the resistance choices can be 1 we can strengthen the cut to,

$$
\begin{aligned}
& -\sum_{r=1}^{n_r} x_{e,r}\frac{\Delta\pi^{+}_{e}}{L_e} + \phi'_{\hat{r}}(\hat{I})I^{+}_{e,\hat{r}} + \sum_{r\neq\hat{r}} \phi_{r}(\tilde{I}_r)I^{+}_{e,r} \\
& + x_{e,\hat{r}}\left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right)x^{\mathrm{dir}}_{e} + \sum_{r\neq\hat{r}} x_{e,r}\left(\phi_{r}(\tilde{I}_r) - \phi'_{r}(\tilde{I}_r)\tilde{I}_r\right)x^{\mathrm{dir}}_{e} \leq 0.
\end{aligned} \tag{5.13}
$$

If we use the equal intercept option (choice (iii)), the nonlinearity in (5.13) can be reduced to

$$
-\sum_{r=1}^{n_r} x_{e,r}\frac{\Delta\pi^{+}_{e}}{L_e} + \phi'_{\hat{r}}(\hat{I})I^{+}_{e,\hat{r}} + \sum_{r\neq\hat{r}} \phi'_{r}(\tilde{I}_r)I^{+}_{e,r} + \sum_{r=1}^{n_r} x_{e,r}\left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right)x^{\mathrm{dir}}_{e} \leq 0. \tag{5.14}
$$

Using the fact that the sum of resistance choices $x_{e,r}$ is 1 we can simplify cuts to,

$$
\begin{aligned}
\boldsymbol{L}g^{+}_{e,\hat{r}}(\boldsymbol{y};\hat{I}) :=\; & -\tfrac{\Delta\pi^{+}_{e}}{L_e} + \phi'_{\hat{r}}(\hat{I})I^{+}_{e,\hat{r}} + \left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right)x^{\mathrm{dir}}_{e} \\
& + \sum_{r\neq\hat{r}} \phi'_{r}(\tilde{I}_r)I^{+}_{e,r} \leq 0 \\
\boldsymbol{L}g^{-}_{e,\hat{r}}(\boldsymbol{y};\hat{I}) :=\; & -\tfrac{\Delta\pi^{-}_{e}}{L_e} + \phi'_{\hat{r}}(\hat{I})I^{-}_{e,\hat{r}} + \left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right)(1 - x^{\mathrm{dir}}_{e}) \\
& + \sum_{r\neq\hat{r}} \phi'_{r}(\tilde{I}_r)I^{-}_{e,r} \leq 0
\end{aligned} \tag{5.15}
$$

where for $r \neq \hat{r}, \tilde{I}_r$ satisfies $\phi_r(\tilde{I}_r) - \phi'_r(\tilde{I}_r)\tilde{I}_r = \phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})$.

On the other hand, using options (i) or (ii) we can derive a stronger cut than (5.5) as follows,

$$
\begin{aligned}
& x_{e,\hat{r}} \hat{\boldsymbol{L}} g^{+}_{e,\hat{r}}(y;\hat{I}) + \sum_{r\neq\hat{r}} x_{e,r} \hat{\boldsymbol{L}} g^{+}_{e,r}(\boldsymbol{y};\tilde{I}_r) \leq 0 \\
\Longrightarrow \quad & \begin{cases} -\dfrac{\Delta\pi^{+}_{e}}{L_e} + \phi'_{\hat{r}}(\hat{I}) I^{+}_{e,\hat{r}} + \displaystyle\sum_{r\neq\hat{r}} \phi'_{r}(\tilde{I}_r) I^{+}_{e,r} \\ + x_{e,\hat{r}} \left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right) + \displaystyle\sum_{r\neq\hat{r}} x_{e,r} \left(\phi_{r}(\tilde{I}_r) - \phi'_{r}(\tilde{I}_r)\tilde{I}_r\right) \leq 0. \end{cases}
\end{aligned}
\tag{5.16}
$$

Additionally, for the problems considered in [13] the potential loss function $\phi_r(\cdot)$ is a monomial. In this case, the flow $\tilde{I}$ computed using options (ii) and (iii) are identical. Consequently, the cut can be simplified similar to the approach used in deriving (5.15) from (5.14). The resulting cut can be strengthened as in (5.6) yielding the same cut in (5.15). For our class of problems, options (ii) and (iii) yield the same strengthened cut. It remains to be seen for general class of problems if one of them has a better computational performance. In our computational experience, option (i) has turned out to be inferior compared to options (ii) or (iii). In the following, we show that (5.15) are valid inequalities for (5.2).

Lemma 5.2. *Suppose Assumptions (A1) and (A2) hold. Then, the linearizations in* (5.15) *are well-defined.*

Proof. Assumptions (A1) and (A2) imply that $\phi_r(I)$ is strictly monotonically increasing and convex in I. Hence, for any resistance choice r we have that $\phi_r(\tilde{I}_r) - \phi'_r(\tilde{I}_r)\tilde{I}_r$ is strictly monotonically decreasing and non-positive (from (5.8)). Hence, there exists an unique $\tilde{I}_r$ for each $r \neq \hat{r}$ such that

$$\phi_r(\tilde{I}_r) - \phi'_r(\tilde{I}_r)\tilde{I}_r = \phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}. \tag{5.17}$$

This proves the claim. □

Lemma 5.3. *Suppose Assumptions (A1) and (A2) hold. Then, the linearization in* (5.15) *is a valid inequality for* (5.2).

Proof. Consider an integer feasible point $\overline{\boldsymbol{y}}$ of (mMILP). The constraints in (mMILP) require that at an integer feasible point, only one of $\overline{x}_{e,r} = 1$ for each $e \in \boldsymbol{E}$. Suppose without loss of generality that at the integer feasible point for an edge e, $\overline{x}_{e,\hat{r}} = \overline{x}^{\text{dir}}_{e} = 1$. Hence, only $\overline{I}^{+}_{e,\hat{r}}$ can be non-zero for edge e. Then, the linearization in (5.15) reduces to

$$-\frac{\Delta\overline{\pi}^{+}_{e}}{L_e} + \phi'_{\hat{r}}(\hat{I})\overline{I}^{+}_{e,\hat{r}} + \left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right) \leq 0 \tag{5.18}$$

which is the evaluation at $\overline{\boldsymbol{y}}$ of the outer-approximation to $g^{+}_{e,\hat{r}}(\boldsymbol{y})$ at the point $\hat{I}$. Consider the case that $x_{e,r} = x^{\text{dir}}_{e} = 1$ and $r \neq \hat{r}$. The linearization in (5.15) reduces to

$$
\begin{aligned}
& -\frac{\Delta\overline{\pi}^{+}_{e}}{L_e} + \phi'_{r}(\tilde{I}_r)\overline{I}^{+}_{e,r} + \left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right) \\
= & -\frac{\Delta\overline{\pi}^{+}_{e}}{L_e} + \phi'_{r}(\tilde{I}_r)\overline{I}^{+}_{e,r} + \left(\phi_{r}(\tilde{I}_r) - \phi'_{r}(\tilde{I}_r)\tilde{I}_r\right) \leq 0.
\end{aligned}
\tag{5.19}
$$

The second equation in (5.19) is obtained by substituting the definition of $\tilde{I}_r$ in (5.15). The inequality in (5.19) is evaluation at $\overline{\boldsymbol{y}}$ of the outer-approximation to $g^{+}_{e,r}(\boldsymbol{y})$ at the point $\tilde{I}_r$. This proves the claim. □

The following lemma shows that the linearization in (5.15) are indeed stronger cuts for (mMILP).

LEMMA 5.4. *Suppose Assumptions (A1) and (A2) hold. Let* $\hat{I} > 0$ *and* $\tilde{I}_r \; \forall\, r \neq \hat{r}$ *be as defined in* (5.15)*. Denote by* $\mathcal{F}_2$ *and* $\mathcal{F}_3$*,*

$$\mathcal{F}_2 = \{\boldsymbol{y}\,|\boldsymbol{y} \text{ satisfies } (5.15)\} \quad \mathcal{F}_3 = \{\boldsymbol{y}|\boldsymbol{y} \text{ satisfies } (5.11)\}\,. \tag{5.20}$$

Then, $\mathcal{F}_2 \subset \mathcal{F}_3$*.*

Proof. For any $\overline{\boldsymbol{y}} \in \mathcal{F}_2$

$$\frac{\Delta\overline{\pi}_e^+}{L_e} \geq \phi'_{\hat{r}}(\hat{I})\overline{I}_{e,\hat{r}}^+ + \left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right) x_e^{\text{dir}} + \sum_{r \neq \hat{r}} \phi'_r(\tilde{I}_r)\overline{I}_{e,r}^+ \tag{5.21}$$

From strict monotonicity of ϕ_r (Assumption (A1)) and nonnegativity of flows,

$$\phi'_{\hat{r}}(\hat{I})\overline{I}_{e,\hat{r}}^+ \geq 0 \text{ and } \phi'_r(\tilde{I}_r)\overline{I}_{e,r}^+ \geq 0 \;\forall\, r \neq \hat{r}. \tag{5.22}$$

The following can be inferred from (5.21) using (5.22),

$$\begin{array}{rcl} \frac{\Delta\overline{\pi}_e^+}{L_e} & \geq & \phi'_{\hat{r}}(\hat{I})\overline{I}_{e,\hat{r}}^+ + \left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right) x_e^{\text{dir}} \\ \frac{\Delta\overline{\pi}_e^+}{L_e} & \geq & \phi'_r(\tilde{I}_r)\overline{I}_{e,r}^+ + \left(\phi_{\hat{r}}(\hat{I}) - \phi'_{\hat{r}}(\hat{I})\hat{I}\right) x_e^{\text{dir}} \;\forall\, r \neq \hat{r} \\ & \geq & \phi'_r(\tilde{I}_r)\overline{I}_{e,r}^+ + \left(\phi_r(\tilde{I}_r) - \phi'_r(\tilde{I}_r)\tilde{I}_r\right) x_e^{\text{dir}} \;\forall\, r \neq \hat{r} \end{array} \tag{5.23}$$

Hence, $\mathcal{F}_2 \subseteq \mathcal{F}_3$. To prove the strict inclusion consider some set of flows $\overline{I}_{e,r}^+ > 0$ $r = 1, \ldots, n_r$ and $x_e^{\text{dir}} \in [0, 1]$. Choose $\Delta\overline{\pi}_e^+$ to be the maximum of the right-hand sides of (5.23). Clearly, this choice of $\Delta\overline{\pi}_e^+$ can be shown to not satisfy (5.15) due to nonnegativity property (5.22). The case of negative flow direction can be similarly shown. Hence, $\mathcal{F}_2 \subset \mathcal{F}_3$. □

5.2. Solving (CVXNLP($\boldsymbol{r}$)) **at Integral Nodes.** In the standard LP/NLP-BB algorithm, the nonlinear program that is solved at an integral node is derived by fixing the binary variables in the MINLP to $(\boldsymbol{x}^s, \boldsymbol{x}^{\text{dir},s})$. At a node s where $(\boldsymbol{x}^s, \boldsymbol{x}^{\text{dir},s})$ are integral, Algorithm 1 solves (CVXNLP($\boldsymbol{r}$)) with $\boldsymbol{r} = \boldsymbol{r}^s$ to determine the potentials and flows, $(\boldsymbol{\pi}^{\text{nlp}}, \boldsymbol{I}^{\text{nlp}})$ that satisfy the network analysis equations(NLNEQN($\boldsymbol{r}$)). The flow directions determined by the MILP algorithm $\boldsymbol{x}^{\text{dir},s}$ are ignored. The modification is motivated by the following observations:

- Given a set of resistance choices for the edges $\boldsymbol{r}^s$, the network analysis problem uniquely determines the flow directions on every edge in the network.
- The objective function is independent of the flow directions and only depends on the resistance choices.

If $(\pi^{\text{nlp}}, I^{\text{nlp}})$ satisfies the bounds on flows and potentials, the second observation guarantees that the MILP algorithm will not visit another integral node s' with $\boldsymbol{x}^s = \boldsymbol{x}^{s'}$ and $\boldsymbol{x}^{\text{dir},s} \neq \boldsymbol{x}^{\text{dir},s'}$, since the node s' will not yield a better objective function.

At an integral node, (5.3) invalidates the choice of resistances when the solution of network analysis problem (CVXNLP($\boldsymbol{r}$)) violates the bounds on flows and potentials. This again is a departure from the standard LP/NLP-BB algorithms. We show in the following that this is a valid cut for (2.3). This cut is called the *no-good* cut [7].

LEMMA 5.5. *Suppose Assumption (A1) holds. Further, suppose that at an integral node s the solution to the network analysis problem defined by the resistance choices*

$\boldsymbol{r}^s$, where $r^s_e = r : x^s_{e,r} = 1\ \forall\ e \in \boldsymbol{E}$, violates the bounds on flows and potentials. Then, the cut defined by (5.3) *is a valid cut for* (2.3)*.*

Proof. At a node s where $(\boldsymbol{x}^s, \boldsymbol{x}^{\text{dir},s})$ is integral, (5.3) invalidates the choice of resistances if the bounds on potentials and flows are violated by the solution to (CVXNLP($\boldsymbol{r}$)). By Theorem 3.1 we have that solution to the network analysis problem NLNEQN($\boldsymbol{r}^s$) is unique. Hence, the choice of edge resistances $\boldsymbol{r}^s$ can be invalidated regardless of the choice of direction variables, $\boldsymbol{x}^{\text{dir},s}$. This proves the claim. □

5.3. Convergence to Global Optimum. We will state the main result on the convergence of the algorithm to global optimum below.

THEOREM 5.6. *Suppose Assumptions (A1)-(A2) and (S1)-(S2) hold. Further, suppose* (2.3) *is feasible and cuts are added as described in* **NodeCuts**. *Then Algorithm 1 with the node linearization routine in Algorithm 2 converges to the globally optimal solution of MINLP in* (2.3)*.*

Proof. Lemma 4.1 shows that (4.6) is a convex relaxation of (2.3). Lemma 5.4 shows that the linearizations added by Algorithm 2 do not cut off the feasible region of (2.3). Lemma 5.5 proves the validity of the cut (5.3). From the above arguments, it is clear that Algorithm 1 does not cut off any choice of resistances $\boldsymbol{r}$ such that the solution of the network analysis (CVXNLP($\boldsymbol{r}$)) satisfies the bounds on potential and flow. Hence, algorithm converges to the globally optimal solution. □

6. Repair and Initilization Procedures. MILP algorithms incorporate a number of heuristics, such as the *feasibility pump* [19] or relaxation induced neighborhood search, for finding good feasible solutions at the root node as well as at intermediate nodes. The feasible solutions allow pruning the search tree and are critical to the computational efficiency. Similar algorithms have also been devised for MINLP problems [11, 17, 12] in recent years. These approaches typically require the solution of a related MILP which can be quite expensive. Exploiting the problem structure in nonlinear network design problems we developed a heuristic procedure that is computationally cheaper in obtaining feasible solutions. We will first describe the **Repair** procedure mentioned in Algorithm 1. The initialization procedure is based on calling **Repair** repeatedly in an attempt to yield a sequence of feasible resistance choices with decreasing objective values. The details of both procedures are given below.

6.1. Repair Procedure. In Algorithm 1 we call the procedure **Repair** (Line 25) when the network analysis solution for a given set of resistance choices $\boldsymbol{r}$ violates the bounds. The procedure attempts to find a feasible set of resistance choices in the neighborhood of $\boldsymbol{r}$. The details of this procedure are provided in Algorithm 3. In the following we will use $(\boldsymbol{\pi}^{\text{nlp}}, \boldsymbol{I}^{\text{nlp}})$ to denote the potential and flows that result from solving (CVXNLP($\boldsymbol{r}$)). In our experience we have found that the most common type of violations are: (i) maximum flow bound and (ii) minimum potential bound violations. The algorithm handles the violations as explained below.

- *Flow Bound Violation*: By Assumption (A3) we have that $I^{\max}_{e,r}$ increases with index r. Hence, for an edge that violates the flow upper bound we can expect to reduce bound violation by increasing r.
- *Minimum Potential Bound Violation*: In this case we also account for the flow direction on the edges connecting to the node that violates the bound. Specifically, we increment the resistance index r_e for an edge for which only the downstream node violates the minimum potential bound. This change

will amount to decreasing the potential loss on edge e if the flow direction is unchanged.

In both cases we address the infeasibility by increasing the index r_e (or equivalently, decreasing the value of the resistances) which results in increasing the objective (since $c_r < c_{r+1}$ by Assumption (A3)). So the procedure results in a sequence of resistance choices $\{\boldsymbol{r}^l\}$ in which the edge resistances are monotonically decreasing ($r_e^l \geq r_e^{l+1} \implies R_{r_e^l} \leq R_{r_e^{l+1}}$ by Assumption (A3)) and objective value is monotonically increasing with l. The choice of downstream node for potential violation is motivated by the following observation which holds for tree networks. In tree networks, the flow directions on edges can be prescribed a priori independent of the choice of $\boldsymbol{r}$. We will consider three different cases:

(i) Suppose $\exists\ e = (i,j) : I_e^{\text{nlp}} > 0$ and the upstream node violates the bound ($\pi_i^{\text{nlp}} < \pi_i^{\min}$) but not the downstream node ($\pi_j^{\text{nlp}} \geq \pi_j^{\min}$). Decreasing the resistance on edge e (incrementing the index r_e) will only serve to increase the potential at node j and nodes downstream of j while leaving node i's potential unchanged.

(ii) Suppose $\exists\ e = (i,j) : I_e^{\text{nlp}} > 0$ and the upstream and downstream nodes violate the bound ($\pi_i^{\text{nlp}} < \pi_i^{\min}$,$\pi_j^{\text{nlp}} < \pi_j^{\min}$). Similarly, decreasing the resistance on edge e (incrementing the index r_e) only have serve to increase the potential at node j and other nodes downstream of it while leaving node i's potential unchanged. Instead picking a node that is upstream of i will affect potential at both nodes.

(iii) Suppose $\exists\ e = (i,j) : I_e^{\text{nlp}} > 0$ and only the downstream node violate the bound ($\pi_i^{\text{nlp}} \geq \pi_i^{\min}$,$\pi_j^{\text{nlp}} < \pi_j^{\min}$). In this case, decreasing the resistance on edge e (incrementing the index r_e) will increase the potential at node j and decrease the bound violation.

For the case of potental bound violation, the approach takes a minimal approach to resistance changes. For general networks, it is difficult to predict the flow directions that will result from changing $\boldsymbol{r}$. A single resistance change on an edge can affect the flow directions in the entire network. The algorithm is structured so that flow bound violations are rectified before addressing potential bound violations. The algorithm proceeds to the next iteration if at least one of the edge resistances is modified. It is possible to find instances where edge resistances cannot be decreased further since $r_e^l = n_r$ for all edges e that are identified for modification. In such cases, the algorithm terminates without a feasible solution. Since resistance changes in our procedure always increase objective value we terminate the procedure with no solution if the resulting objective value exceeds the value of a previously known solution (η^{best}). Though the algorithm is not guaranteed to find a feasible solution even if one exists, our computational experience has been quite encouraging (see Section 7).

6.2. Initialization Procedure. We utilize the **Repair** procedure to find an initial set of resistances for the edges that can serve as a feasible solution for the optimization algorithm. The **InitialSoln** procedure is designed to execute a certain number of rounds, $rnds^{\max}$. In each round of the algorithm, the edges are randomly assigned a resistance choice. The **Repair** algorithm is called to obtain a feasible set of resistances as described in the previous section. Given a set of random assignments $\boldsymbol{r}$ the **Repair** procedure, if successful returns $\boldsymbol{r}'$ which satisfies potential and flow bounds but has equal or higher objective value. As part of the **InitialSoln** procedure it is desirable to produce feasible resistance choices in each round with decreasing objective values. In order to favor this we invoke a greedy hueristic, once a feasible set of resistances has been found, to modify the random assignment resistances until a set of resistances with objective value that is a fraction of the objective value of the best feasible solution found thus far (refer Steps $7-10$). In our experience, the procedure has been effective in finding good initial feasible solutions.

7. Computational Results. The algorithm presented in the previous section has been implemented using the MILP solver `IBM-ILOG CPLEX` [26] C-callable libraries. The user cut generation routine has been implemented using the routine `CPXcutcallback` available within `IBM-ILOG CPLEX` [26]. This routine satisfies the Assumption (S). The **Repair** procedure returns the feasible set of resistances (incumbent) to the solver using the `CPXheurcallback` routine. The network analysis problem solved at integral nodes have been solved using the convex formulation (CVXNLP($\boldsymbol{r}$)). The NLP solver `IPOPT` [32] was used for solving (CVXNLP($\boldsymbol{r}$)). All computations have been performed on Pentium Duo Core machine with 2.63 GHz processor and 3 GB RAM. The convergence criteria for all problems was 0.0% optimality gap. The time limit varied according to the problem sizes.

The problem instances considered are design of pipe sizes in hydraulic circuits [13]. It can be easily verified that the potential loss function defined in [13] satisfies Assumption (A). Table 7.1 provides information on the size for each of the problem instances.

7.1. Initialization Procedure. We will describe the results from using the initialization procedure **InitialSoln** on all the problems instances. In all cases the maximum number of rounds $rnds^{\max} = 20$, the maximum number of iterations $iter^{\max} = 20$ and fraction $\alpha = 0.85$ was used. Table 7.2 provides detailed information on the objective value of the best feasible solution returned by the **InitialSoln** procedure (column: Best Obj), total number of NLPs solved (column: # NLPs) and computational time (column: CPU time). We also provide in the table the root node objective (column: Root Obj) to indicate the initial gap at the root node. The computational effort required to generate the initial feasible solutions is small (< 3 mins) for all instances. The problem instance `newyork` is one in which all edges already have diameters and design problem is to find if edges are to be reinforced with another pipe. This is the reason why the root node objective value is 0.0.

7.2. Performance on Moderate Instances. We will describe the computational results for the moderate sized problems - `shamir`, `hanoi`, `newyork` and `blacksburg`. Table 7.3 presents results from solving a subset of the problems in Table 7.1 using Algorithm 1 with linearizations defined by (5.15). The parameters in the algorithm are chosen as:

$$\begin{array}{c} \beta_{oa} = 5, K_{oa} = 10^{-3}, M_{oa} = 5 \\ iter^{\max} = 50 \ \text{ (for } \textbf{Repair} \text{ algorithm)} \end{array} \tag{7.1}$$

Table 7.3 shows that the algorithm can solve all of the moderate instances to global optimality and within 5 minutes. The objective function values at the global optimal solution coincide with the best solutions reported in [13]. However, the algorithm in [13] was based on a non-convex MINLP algorithm and was unable to certify the global optimality of the reported solutions.

To emphasize the effectiveness of the approach described, consider the following two versions of the algorithm without the enhancements described herein.

- **V1**: Cut defined in (5.6) is used for node linearizations

- **V2**: The constraint (4.5), modeling the limit on the potential loss, is removed and cut defined in (5.15) is used for node linearizations.

Table 7.4 shows the computational results on the moderate instances for the two versions described above. The results show that **V1** is unable to solve 3 of the instances to global optimality. The instances `hanoi` and `newyork` terminate with out-of-memory error while `blacksburg` reaches time limit of 4 hours. This clearly shows that the stronger cut (5.15) is critical to the performance of Algorithm (1). On the other hand version **V2** has the stronger cut (5.15) but the inequality limiting the potential loss (4.5) is removed. Version **V2** is able to solve all problems to global optimality but does worse in terms of number of nodes explored on all problems except `shamir`. Also, the computational time is significantly higher for **V2** on `hanoi` and `newyork`. This emphasizes the importance of constraint (4.5). A refinement of these constraints are possible if the lower and upper bounds on flows can be refined as is done in a spatial branch and bound algorithm. This presents an opportunity to employ spatial brach and bound algorithms in a future investigation.

7.3. Performance on Larger Instances. Table 7.5 shows the results from solving the larger problem instances. The algorithm parameters are as given in (7.1). The computational time limit was set to 4 hours for these set of problems. We also limited the number of nodes explored to $165,000$ to ensure a recoverable termination. All instances other than `modena` terminate on reaching the node exploration limit. Relaxing this limit results in an unrecoverable out-of-memory error. In the larger instances, we have observed that in these larger problem instances the algorithm seldom reaches an integer feasible node. Consequently, the upper bound on the optimal objective value rarely improves and this affects the overall efficiency of the algorithm. We employ a heuristic procedure which is essentially a call to **Repair** procedure after every 500 nodes. The **Repair** procedure requires an initial set of resistance choices which is determined as follows. Denote by $\boldsymbol{y}$ the solution of the relaxed LP at the node at which the heuristic is invoked. The initial set of resistance choices $\boldsymbol{r}$ is chosen as

$$r_e := \min_{r=1,\ldots,n_r} \{x_{e,r} | x_{e,r} \geq \frac{1}{n_r}\}.$$

Further, we do not invoke the **Repair** procedure if the objective value of choices above is greater than the best feasible solution obtained thus far. We have observed that this procedure is effective in finding feasible solutions. The computational effort is also quite small compared with more general heuristic procedures such as feasibility pump [11, 17] or relaxation induced neighborhood search [12]. From the Table 7.5 it is clear that the algorithm was unable to solve any of the instances to global optimality. Table 7.6 compares the best objective values found by approach in [13] and variants of Algorithm 1. Without the heuristic procedure for finding additional solutions, `IBM-ILOG CPLEX` [26] has difficulty in finding good feasible solutions. In the instance of `foss_poly_0`, the algorithm terminates at a solution with objective value $71,741,922.90$, while [13] reports a solution with better objective value of $70,680,507.90$. Similarly in the cases of `foss_poly_1` and `modena` we terminate at solutions of $31,352.87$ and $4,191,445.38$ which are worse than the solutions reported in [13] $29,202.99$ and $2,580,379.53$. The objective value for the case of `foss_iron` matches that reported in [13]. For the instance `pescara` we converge to a better value of $1,814,271.91$ as compared with the solution reported in [13] of $1,830,440.4$.

RepairSoln$(\boldsymbol{r}, iter^{\max}, \eta^{\text{best}})$

Data: $\boldsymbol{r}$ - set of resistance that needs to be repaired
$iter^{\max}$ - maximum number of iterations
η^{best} - objective value of best feasible solution.

Set $l = 1, progress = true, repaired = false, \boldsymbol{r}^1 = \boldsymbol{r}$.
while $l \leq iter^{\max}$ *and not*$(repaired)$ *and* $progress$ **do**
    Solve CVXNLP$(\boldsymbol{r}^l)$ to obtain flows and potentials $\boldsymbol{I}^{\text{nlp}}, \boldsymbol{\pi}^{\text{nlp}}$.
    if $\boldsymbol{I}^{\text{nlp}}, \boldsymbol{\pi}^{\text{nlp}}$ *satisfies bounds* **then**
        Set $repaired = true$.
    else `/* increase resistances */`
        Set $progress = false$

        for $e \in \boldsymbol{E}$ **do** `/* max. flow violation */`
            if $I_e^{\text{nlp}} > I_{e,r_e}^{\max}$ *and* $r_e^l < n_r$ **then**
                Set $r_e^{l+1} = r_e^l + 1$
                Set $progress = true$
            else
                Set $r_e^{l+1} = r_e^l$
            end if
        end for

        for $e = (i,j) \in \boldsymbol{E}$ *and not*$(progress)$ **do** `/* min. potential violation */`
            if $(I_e^{\text{nlp}} > 0$ *and* $\pi_j < \pi_j^{\min}$ *and* $\pi_i \geq \pi_i^{\min})$ *or* $(I_e^{\text{nlp}} \leq 0$ *and* $\pi_i < \pi_i^{\min}$ *and* $\pi_j \geq \pi_j^{\min})$ **then** `/* downstream node violation */`
                if $r_e^l < n_r$ **then**
                    Set $r_e^{l+1} = r_e^l + 1$
                    Set $progress = true$
                else
                    Set $r_e^{l+1} = r_e^l$
                end if
            end if
        end for
    end if
    if $\sum_{e \in \boldsymbol{E}} c_{r_e^l} L_e \geq \eta^{\text{best}}$ **then** `/* objective worse than best */`
        Set $progress = false$
        Set $repaired = false$
    end if
    Set $l = l + 1$.
end while
return $(repaired, \boldsymbol{r}^l)$

Algorithm 3: Algorithm **Repair** for repairing a set of resistance choices whose network analysis solution violates bounds on flow or potentials.

InitialSoln

Data: $rnds^{\max}$ - maximum number of rounds
$iter^{\max}$ - maximum number of iterations per round
α - fractional decrease in objective value
$rand()$ - integer random generator
$a \; mod \; b$ - function returning the remainder when a is divided by b

Set $rnds = 1, bestObj = +\infty$
while $rnds \leq rnds^{\max}$ **do**
  Set $r_e = (rand() \; mod \; n_r) + 1 \; \forall \, e \in \boldsymbol{E}$
  while $\sum_{e \in \boldsymbol{E}} c_{r_e} L_e > \alpha \cdot bestObj$ **do** `/* Decrease resistances */`
    Set $e' = \arg\max\{(c_{r_e} - c_{r_e - 1}) L_e \mid e \in \boldsymbol{E}, r_e > 1\}$
    Set $r_{e'} = r_{e'} - 1$
  end while

  $(repaired, \boldsymbol{r}') =$ **Repaired**$(\boldsymbol{r}, iter^{\max})$
  if $repaired$ *and* $\sum_{e \in \boldsymbol{E}} c_{r'_e} L_e < bestObj$ **then** `/* Feasible solution */`
    Set $bestObj = \sum_{e \in \boldsymbol{E}} c_{r'_e} L_e$
    Set $\boldsymbol{r}^{\text{best}} = \boldsymbol{r}'$
  end if
  Set $rnds = rnds + 1$
end while
return $\boldsymbol{r}^{\text{best}}$

Algorithm 4: Algorithm for finding an initial feasible solution to the nonlinear network design problems.

Problem name	# nodes ($\lvert \boldsymbol{N} \rvert$)	# edges ($\lvert \boldsymbol{E} \rvert$)	# diameters (n_r)	# vars	# bin vars	# cons
shamir	8	8	14	367	120	502
hanoi	33	34	6	746	238	1051
new york	21	21	12	902	294	1237
blacksburg	32	25	11	1606	525	2212
foss_poly_0	38	58	7	1429	464	2008
foss_iron	38	58	13	2473	812	3400
foss_poly_1	38	58	22	4039	1334	5490
pescara	74	99	13	4229	1386	5811
modena	276	317	13	13586	4438	18656

TABLE 7.1

Description of the problem instances from [13]. # vars - number of variables, # bin vars - number of binary variables, # cons - number of constraints corresponding to the MINLP formulation in (4.6).

Problem name	Best Obj	Root Obj	# NLPs	CPU time (s)
shamir	592,000	306,598.80	217	3
hanoi	7,215,983.60	5,039,990.40	138	3
newyork	89,649,395.40	0.0	139	3
blacksburg	217,598.96	101,716.68	144	4
foss_poly_0	87,930,692.30	65,319,472.94	83	3
foss_iron	188,044.17	172,746.54	101	3
foss_poly_1	75,055.23	25,340.53	159	5
pescara	2,287,240.01	1,558,055.59	138	22
modena	6,016,670.90	2,088,449.27	204	152

TABLE 7.2

*Results from using **InitialSoln** procedure on the moderate problem instances. The parameters of the procedure were set as , $rnds^{\max} = 20$, $iter^{\max} = 20$ and $\alpha = 0.85$. Best Obj - objective value of the best feasible solution generated by the initalization algorithm, Root Obj - objective value at root node of NLP/BB search tree, # NLPs - number of NLPs solved by the initialization algorithm, CPU time (s) - computational time for the initialization algorithm in seconds.*

Problem name	Best Obj	Opt Gap(%)	# NLPs (# Repair NLPs)	# $\mathcal{OA}$ cuts	CPU time (s) (# Nodes)
shamir	419,000	0.0	30 (18)	162	9 (5011)
hanoi	6,109,620.09	0.0	165 (83)	424	251 (73,313)
newyork	39,307,799.72	0.0	154 (104)	491	267 (75,121)
blacksburg	118,251.09	0.0	24 (4)	239	139 (33,231)

TABLE 7.3

*Results from using Algorithm 1 with linearizations defined by (5.15). Best Obj - objective of best feasible solution returned on termination, Opt Gap (%) - optimality gap at termination of the algorithm defined as (Best Obj - LB Obj)/(Best Obj)*100 where LB Obj is the value of the lower bound at termination, # $\mathcal{OA}$ cuts - total number of cuts defined in (5.15) added to the search tree, # NLPs - total number of NLPs solved, # Repair NLPs - number of NLPs solved by **Repair** procedure, # $\mathcal{OA}$ cuts - number of cuts added, CPU time (s) - total computational time in seconds, # Nodes - number of nodes explored in the search tree.*

	V1			**V2**		
Problem name	Best Obj	Opt Gap(%)	CPU time (sec) (# Nodes)	Best Obj	Opt Gap(%)	CPU time (sec) (# Nodes)
shamir	419,000	0.0	45 (16,561)	419,000	0.0	6 (3541)
hanoi	6,309,721.40	5.12	3710^m (212,865)	6,109,620.09	0.0	984 (281,312)
newyork	50,941,900.72	51.85	7800^m (187,235)	39,307,799.72	0.0	324 (115,693)
blacksburg	118,251.09	1.34	14400^t (124,866)	118,251.09	0.0	126 (33,516)

TABLE 7.4

*Computational results for algorithms versions **V1** and **V2**. For an explanation of column names refer to caption in Table 7.3. ^{m} - terminates with out-of-memory message, ^{t} - terminates due to time limit of 4 hours.*

.

Problem name	Best Obj	LB Obj	Opt Gap (%)	# NLPs (# Repair NLPs)	# $\mathcal{OA}$ cuts	CPU time (h) (# nodes)
foss_poly_0	71, 741, 922.90	70, 0625, 097.18	2.34	347 (333)	943	0.6 (165, 000^n)
foss_iron	178, 494.14	177, 515.24	0.55	887 (586)	628	0.83 (165, 000^n)
foss_poly_1	31, 352.87	26, 236.17	16.32	1267 (1262)	355	0.9 (165, 000^n)
pescara	1, 814, 271.91	1, 700, 108.17	6.27	1435 (1398)	1519	2.1 (165, 000^n)
modena	4, 191, 445.38	2, 206, 137.43	47.37	4116 (4115)	2029	4.0^t (55, 439)

TABLE 7.5

Results from using Algorithm 1 with node linearizations defined by (5.15). *For an explanation of column names refer to caption in Table 7.3. LB Obj - objective value of the lower bound at termination,* ^{n} *- limit of number of nodes in search tree reached,* ^{t} *- time limit reached.*

	From [13]	**Algorithm 1 (w/ heuristic)**			**Algorithm 1**		
Problem name	Best Obj	Best Obj	Opt Gap(%)	CPU time (h) (# Nodes)	Best Obj	Opt Gap(%)	CPU time (h) (# Nodes)
foss_poly_0	70, 680, 507.90*	71, 741, 922.90	2.34	0.6 (165, 000^n)	87, 930, 692.30	20.95	0.97 (165, 000^n)
foss_iron	178, 494.14*	178, 494.14*	0.55	0.83 (165, 000^n)	178, 494.14*	0.55	0.92 (165, 000^n)
foss_poly_1	29, 202.99*	31, 352.87	16.32	0.9 (165, 000^n)	32, 928.65	20.31	0.97 (165, 000^n
pescara	1, 837, 440.40	1, 814, 271.91*	6.27	2.1 (165, 000^n)	2, 112, 583.71	20.36	1.92 (165, 000^n)
modena	2, 580, 379.53*	4, 191, 445.38	47.37	4.0^t (55, 439)	6, 016, 670.90	63.32	4.0^t (61, 102)

TABLE 7.6

*Comparison of the computational performance between Algorithm 1 with heuristic of calling **Repair** procedure every* 500 *nodes and Algorithm 1 without the heuristic. The first column provides the best objective value obtained by the approach described in [13]. In the above, * refers to the approach that has the lowest objective value among approach in [13] and our variants of Algorithm 1.*

.

8. Conclusions. Design optimization of nonlinear networks has been studied for over 50 years. This paper proposes a novel approach for global optimization of nonlinear network design with applications to variety of engineering disciplines - electrical, hydraulic, structures and transportation. A modified LP/NLP-BB algorithm is proposed with improved linearizations and is guaranteed to converge to a globally optimal solution. Computational studies on literature examples confirm the effectiveness of the algorithm.

Acknowledgements. The author would like to thank Sven Leyffer, Ashutosh Mahajan for their comments on an earlier draft of the paper. Special thanks are due to two anonymous referees whose comments served to improve the presentation and the results in the paper.

REFERENCES

[1] www.or.deis.unibo.it/research_pages/ORinstances/ORinstances.htm.
[2] *Transport Phenomena*, chapter 8, pages 231–242. John Wiley & Sons, 2007.
[3] A.J. Abebe and D.P. Solomantine. Application of global optimization to the design of pipe networks. In 3$^{\text{rd}}$ *International Conference on Hydroinformatics, Copenhagen*, 1998.
[4] K. Abhishek, S. Leyffer, and J.T. Linderoth. FilMINT: An outer-approximation-based solver for nonlinear mixed integer programs. Technical Report ANL/MCS-P1374-0906, Mathematics and Computer Science Division, Argonne National Laboratory, 2006.
[5] E. Alperovits and U. Shamir. Design of optimal water distribution systems. *Water Resources Research*, 13(6):885–900, 1977.
[6] S. Artina and J. Walker. Sull´uso della programmazione a valori misti nel dimensionamento di costo minimo di reti in pressione. *Atti dell´Accademia delle Scienze dell´Istituto di Bologna (in Italian)*, Anno 271(Serie III, Tomo X), 1983.
[7] E. Balas and R. Jeroslow. Canonical cuts on the unit hypercube. *SIAM Journal on Applied Mathematics*, 23:61–69, 1972.
[8] P. Belotti. Private communication, 2011.
[9] P. Bonami, L.T. Biegler, A.R. Conn, G. Cornuéjols, I.E. Grossmann, C.D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wächter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 2006.
[10] P. Bonami, T. Biegler, L., A. R. Conn, G. Cornuéjols, E. Grossmann, I., C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wächter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 2008.
[11] P. Bonami, G. Cornuéjols, A. Lodi, and F. Margot. A feasibility pump for mixed integer nonlinear programs. *Mathematical Programming*, 119:331–352, 2009.
[12] P. Bonami and J. P. M. Gon calvex. Heuristics for convex mixed integer nonlinear programs. *Computational Optimization and Applications*, in press.
[13] C. Bragalli, C. D´Ambrosio, J. Lee, A. Lodi, and P. Toth. Water network design by MINLP. Technical Report RC24495, IBM Research Report, 2008.
[14] R.H. Byrd, J. Nocedal, and R.A. Waltz. KNITRO: An integrated package for nonlinear optimization. In G. di Pillo and Roma M., editors, *Large-Scale Nonlinear Optimization*, pages 35–59. 2006.
[15] C. Cherry. Some general theorems for non-linear systems possessing reactanaces. *Proc. of Philosophical Society*, Oct. 1951.
[16] M. Collins, L. Cooper, R. Helgason, J. Kennington, and L. LeBlanc. Solving the pipe network analysis problem using optimization techniques. *Management Science*, 24(7):747–760, 1978.
[17] C. D'Ambrosio, A. Frangioni, L. Liberti, and A. Lodi. A Storm of Feasibility Pumps for Nonconvex MINLP. *Mathematical Programming (Ser. B)*, to appear.
[18] G. Eiger, U. Shamir, and A. Ben-Tal. Optimal design of water distribution networks. *Water Resources Research*, 30(9):2637–2646, 1994.
[19] M. Fischetti, F. Glover, and A. Lodi. The feasibility pump. *Mathematical Programming*, 104:91–104, 2005.
[20] R. Fourer, D. M. Gay, and B. W. Kerighan. *AMPL: A Modeling Language for Mathematical Programming.* Duxbury Press, second edition, 2002.

[21] O. Fujiwara and D.B. Khang. A two-phase decomposition method for optimal design of looped water distribution networks. *Water Resources Research*, 26(4):539–549, 1990.

[22] M. Gay, D. Hooking Your Solver to AMPL. Technical report, Bell Laboratories, Murray Hill, NJ, 1997.

[23] I.C. Goulter, B.M. Lussier, and D.R. Morgan. Implications of head loss choice in the optimization of water distribution networks. *Water Resources Research*, 22(5):819–822, 1986.

[24] C.T. Hansen, K. Madsen, and H.B. Nielsen. Optimization of pipe networks. *Mathematical Programming*, 52:45–58, 1991.

[25] C. T. Hendrickson and B. N. Janson. A Common Network Flow Formulation to Several Civil Engineering Problems. *Civil Engineerng Systems*, 1(2):195–203, 1984.

[26] IBM ILOG CPLEX. *v.12.2*. IBM Corporation.

[27] S. Leyffer. Private communication, 2011.

[28] O.L. Mangasarian. *Nonlinear Programming*. Classics in Applied Mathematics 10. SIAM, 1994.

[29] I. Quesada and I.E. Grossmann. An LP/NLP baed branch-and-bound algorithm for convex MINLP optimization problems. *Computers and Chemical Engineering*, 16:837–947, 1992.

[30] J. Tospornsampan, I. Kita, M. Ishii, and Y. Kitamura. Split-pipe design of water distribution networks using a combination of tabu search and genetic algorithm. *International J. Computer, Information and Systems Science and Engineering*, 2007.

[31] Bonmin. v. 0.1. http://projects.coin-or.org/bonmin.

[32] Ipopt. v. 3.8.1. http://projects.coin-or.org/ipopt.

[33] Wächter and Biegler. L.T. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[34] J. Zhang and D. Zhu. A bilevel programming method for pipe network optimization. *SIAM J. Optimization*, 6(3):838–857, 1996.