A Penalty Method Based on a Gauss-Newton Scheme for AC-OPF

Ilyes Mezghani, Anthony Papavasiliou
UCLouvain
ilyes.mezghani@uclouvain.be
anthony.papavasiliou@uclouvain.be

Quoc Tran-Dinh
University of North Carolina
quoctd@email.unc.edu

Ion Necoara
University Politehnica Bucharest
ion.necoara@acse.pub.ro

***Abstract*—We propose a globally convergent and robust Gauss-Newton algorithm for finding a (local) optimal solution of a non-convex and possibly non-smooth optimization problem arising from AC optimal power flow on meshed networks. The algorithm that we present is based on a Gauss-Newton-type iteration for an exact penalty reformulation of the power flow problem. We compare our algorithm with a state-of-the-art solver, IPOPT, on several representative problem instances in MATPOWER. We demonstrate the comparable performance of our method for a variety of the MATPOWER test cases.**

***Index Terms*—Nonlinear programming, Optimization with non-convex constraints, penalty reformulation, Gauss-Newton method, AC optimal power flow.**

I. Introduction

The optimal power flow (OPF) problem [1] consists in finding an optimal operating point of a power system while minimizing a certain objective (typically power generation cost), subject to the Kirchhoff's power flow equations and various network and control operating limits. We focus in this paper on the alternating current optimal power flow (AC-OPF) problem, which lies at the heart of short-term power system operations [2]. The increasing integration of distributed resources that are connected to medium and low-voltage networks has increased the relevance of AC-OPF as an appropriate framework of modeling operational constraints that affect the coordination of transmission and distribution system operations [3].

a) Related work: In recent years, there has been a great body of literature that has focused on convex relaxations of the AC-OPF problem, including semidefinite programming relaxations [4], conic relaxations [5], [6], and quadratic relaxations [7]. These works have established conditions under which these relaxations are exact, and understanding cases in which this is not so [8]. Instead, our interest in the present paper is to tackle directly this problem as a non-convex optimization problem with non-linear equality constraints. Such a formulation is sufficiently general to produce physically implementable solutions in the context of realistic system operations.

I. Mezghani and A. Papavasiliou acknowledge the financial support of ENGIE-Electrabel. I. Necoara would like to acknowledge the support from the Executive Agency for Higher Education, Research and Innovation Funding (UEFISCDI), Romania, PNIII-P4-PCE-2016-0731, project ScaleFreeNet, no. 39/2017. Q. Tran-Dinh's work was partly supported by NSF (DMS-1619884).

978-1-6654-3597-0/21/$31.00 ©2021 IEEE

The AC-OPF problem is usually formulated as a non-convex optimization problem due to non-convex constraints, which is often challenging to solve. Classical techniques such as interior-point, augmented Lagrangian, penalty, Gauss-Newton, and sequential quadratic programming methods can only aim at finding a stationary point, which is a candidate for a local minimum [9]. For an iterative method to identify a stationary point that is a local minimum, but not a saddle-point, more sophisticated techniques are required, such as cubic regularization [10] or random noise gradient [11]. However, these methods are often very difficult to implement and inefficient in large-scale problems with non-convex constraints. One of the most efficient and well-established nonlinear solvers for finding stationary points is IPOPT [12], which relies on a primal-dual interior-point method combined with other advanced techniques. We emphasize that this classical method is only guaranteed to converge to a stationary point, and often requires a strategy such as line-search, filter, or trust-region to achieve global convergence under certain restrictive assumptions. Moreover, each iteration of IPOPT requires solving a non-convex subproblem via linearization combined with a line-search or filter strategy.

b) Our approach and contributions: In this paper, we consider an AC-OPF problem over large-scale networks. We are interested in an approach that can tackle general meshed networks. We show that this problem can be posed in the framework of non-convex optimization with a particular structure on the constraints. Based on this structure we devise a provable convergent Gauss-Newton (GN)-type algorithm for solving this non-convex problem. Our algorithm converges globally to a stationary point of the problem from any starting point. In addition, it is also different from standard GN methods in the literature due to the use of a non-smooth penalty instead of a classical quadratic penalty term. This allows our algorithm to converge globally and also to be more robust to ill-conditioning [13]. Hence, we refer to this algorithm as a *global and robust* GN scheme. The main idea of our method is to keep the convex sub-structure of the original problem unchanged and to convexify the non-convex part by exploiting penalty theory and the GN framework. Hence, in contrast to IPOPT, each iteration of our algorithm requires solving a convex subproblem, which can efficiently be solved by many existing convex solvers.

The main contributions of the paper are the following:

(i) We consider a quadratic reformulation of the AC-OPF problem, as in [5], [14], and propose an exact penalty reformulation of the problem in order to handle the non-convex equality constraints and a novel global and robust GN algorithm for solving the corresponding problem.

(ii) For our optimization algorithm, we prove that its iterate sequence converges globally (i.e. from any starting point) to a stationary point of the underlying problem. We also estimate its best-known global sublinear convergence rate.

(iii) We show that the newly developed algorithm can be implemented efficiently on AC-OPF problems and test it on several numerical examples from the MATPOWER test cases [15], [16]. We observe competitive performance to IPOPT.

Our algorithm is simple to implement and can be incorporated flexibly with any available convex sub-solver that supports a warm start strategy in order to gain efficiency.

c) Content: The paper is organized as follows. In Section II, we introduce our Gauss-Newton algorithm. In Section III, we present the AC-OPF problem, its quadratic reformulation and the application of GN to AC-OPF, then illustrate through simulations the performance of our algorithm in Section IV. Finally, Section V concludes the paper. Note that this paper presents the main results of the work, but we will refer to an extended version for details [17].

II. A Gauss-Newton Algorithm for Non-Convex Optimization

In this section, we rely on the following non-convex optimization problem:

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) \;\; \text{s.t.} \;\; \Psi(\boldsymbol{x}) = 0,\; \boldsymbol{x}\in\Omega. \tag{1}$$

We present the main assumptions for (1), propose an exact penalty reformulation, and solve it using a Gauss-Newton-type algorithm. We further characterize the global and local convergence rates of our algorithm.

A. Exact penalty approach for non-convex programming

For the non-convex optimization problem (1) we assume that the objective function f is convex and differentiable and Ω is a compact convex set. Note that our method developed in the sequel can also be extended non-smooth convex function f or smooth non-convex function f whose gradient is Lipschitz continuous, but we make this assumption for simplicity of presentation. Furthermore, the non-convexity enters into the optimization problem through the non-linear equality constraints $\Psi(\boldsymbol{x}) = 0$ defined by $\Psi : \mathbb{R}^d \to \mathbb{R}^n$. We assume that Ψ is differentiable and its Jacobian Ψ' is Lipschitz continuous, i.e. there exists $L_\Psi > 0$ such that:

$$\|\Psi'(\boldsymbol{x}) - \Psi'(\hat{\boldsymbol{x}})\| \le L_\Psi \|\boldsymbol{x} - \hat{\boldsymbol{x}}\| \quad \forall \boldsymbol{x}, \hat{\boldsymbol{x}} \in \Omega,$$

where $\|\cdot\|$ is the ℓ_2-norm. Further, let $\mathcal{N}_\Omega$ denote the normal cone of the convex set Ω:

$$\mathcal{N}_\Omega(\boldsymbol{x}) := \begin{cases} \left\{\boldsymbol{w}\in\mathbb{R}^d \mid \boldsymbol{w}^\top(\boldsymbol{y}-\boldsymbol{x}) \ge 0, \forall \boldsymbol{y}\in\Omega\right\}, \text{if } \boldsymbol{x}\in\Omega \\ \emptyset, \quad \text{otherwise.} \end{cases}$$

Since problem (1) is non-convex, our goal is to search for a stationary point of this optimization problem that is a candidate for a local optimum in the following sense.

Definition II.1 ([9](Theorem 12.9)). *A point $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ is said to be a KKT point of* (1) *if it satisfies the following conditions:*

$$-\nabla f(\boldsymbol{x}^*) - \Psi'(\boldsymbol{x}^*)\boldsymbol{y}^* \in \mathcal{N}_\Omega(\boldsymbol{x}^*),\; \boldsymbol{x}^*\in\Omega,\; \Psi(\boldsymbol{x}^*) = 0.$$

Here, $\boldsymbol{x}^*$ is called a stationary point of (1)*, and $\boldsymbol{y}^*$ is the corresponding multiplier. Let $\mathcal{S}^*$ denote the set of these stationary points.*

Since Ω is compact, and Ψ and f are continuous, by the Weierstrass theorem, we have:

Proposition II.1. *If $\Omega \cap \{\boldsymbol{x} \mid \Psi(\boldsymbol{x}) = 0\} \neq \emptyset$, then* (1) *has global optimal solutions.*

B. Exact penalized formulation

Associated with (1), we consider its exact penalty form [9, Chapt. 17.3]:

$$\min_{\boldsymbol{x}\in\Omega} \Big\{ F(\boldsymbol{x}) := f(\boldsymbol{x}) + \beta|\Psi(\boldsymbol{x})| \Big\}, \tag{2}$$

where $\beta > 0$ is a penalty parameter, and $|\cdot|$ is the ℓ_1-norm. Two reasons for choosing an exact (non-smooth) penalty are as follows. First, for a certain finite choice of the parameter β, a single minimization in x of (2) can yield an exact solution of the original problem (1). Second, it does not square the condition number of Ψ as in the case of quadratic penalty methods, thus making our algorithm presented below more robust to ill-conditioning of the non-convex constraints. Now, we summarize the relationship between stationary points of (1) and of its penalty form (2). For this, let us define the directional derivative:

$$DF(\boldsymbol{x}^*)[\boldsymbol{d}] := \nabla f(\boldsymbol{x}^*)^\top \boldsymbol{d} + \beta\xi(\boldsymbol{x}^*)^\top \Psi'(\boldsymbol{x}^*)^\top \boldsymbol{d},$$

where $\xi(\boldsymbol{x}^*) \in \partial|\Psi(\boldsymbol{x}^*)|$ is one subgradient of $|\cdot|$ at $\Psi(\boldsymbol{x}^*)$, and $\partial|\cdot|$ denotes the subdifferential of $|\cdot|$, see [13]. Recall that the necessary optimality condition of (2) is

$$0 \in \nabla f(\boldsymbol{x}^*) + \beta\Psi'(\boldsymbol{x}^*)\partial|\Psi(\boldsymbol{x}^*)| + \mathcal{N}_\Omega(\boldsymbol{x}^*).$$

Then, this condition can be expressed equivalently as

$$DF(\boldsymbol{x}^*)[\boldsymbol{d}] \ge 0, \quad \forall \boldsymbol{d} \in \mathcal{F}_\Omega(\boldsymbol{x}^*), \tag{3}$$

where $\mathcal{F}_\Omega(\boldsymbol{x})$ is the set of feasible directions to Ω at $\boldsymbol{x}$:

$$\mathcal{F}_\Omega(\boldsymbol{x}) := \left\{\boldsymbol{d}\in\mathbb{R}^d \mid \boldsymbol{d} = t(\boldsymbol{y}-\boldsymbol{x}),\; \forall \boldsymbol{y}\in\Omega,\; t\ge 0\right\}.$$

Any point $\boldsymbol{x}^*$ satisfying (3) is called a stationary point of the penalized problem (2). Stationary points are candidates for local minima, local maxima, and saddle-points. If, in addition, $\boldsymbol{x}^*$ is feasible to (1), then we say that $\boldsymbol{x}^*$ is a feasible stationary point. Otherwise, we say that $\boldsymbol{x}^*$ is an infeasible stationary point. Proposition II.2 shows the relation between (1) and (2).

Proposition II.2 ([9], (Theorem 17.4.)). *Suppose that $\boldsymbol{x}^*$ is a feasible stationary point of* (2) *for β sufficiently large. Then, $\boldsymbol{x}^*$ is also stationary point of the original problem* (1).

Proposition II.2 requires $\boldsymbol{x}^*$ to be feasible for (1). When the feasible set $\Omega \cap \{\boldsymbol{x} \mid \Psi(\boldsymbol{x}) = 0\} \neq \emptyset$ of (1) is nonempty and bounded, according to [18, Proposition 2], if (1) satisfies the extended Mangarasian-Fromovitz constrained qualification condition (see [18, Proposition 2] for concrete definition), then there exists $\beta_* > 0$ such that for any $\beta > \beta_*$, every global or local solution of the penalized problem (2) is also a global or local optimal solution of (1), respectively. By [18, Proposition 3], β needs to be chosen such that $\beta > \beta_* := \|\boldsymbol{y}^*\|_\infty$, where $\boldsymbol{y}^*$ is any optimal Lagrange multiplier of (1).

C. Global Gauss-Newton method

We first develop our GN algorithm. Then, we investigate its global convergence rate.

1) The derivation of the Gauss-Newton scheme and the full algorithm: Our GN method aims at solving the penalized problem (2) using the following convex subproblem:

$$\begin{aligned}\min_{\boldsymbol{x}\in\Omega} \Big\{ &\mathcal{Q}_L(\boldsymbol{x};\boldsymbol{x}^k) := f(\boldsymbol{x}) \\ &+\beta|\Psi(\boldsymbol{x}^k) + \Psi'(\boldsymbol{x}^k)(\boldsymbol{x}-\boldsymbol{x}^k)| + \tfrac{L}{2}\|\boldsymbol{x}-\boldsymbol{x}^k\|^2 \Big\}\end{aligned} \tag{4}$$

where $\boldsymbol{x}^k$ is a given point in Ω for linearization, $\Psi'(\cdot)$ is the Jacobian of Ψ, and $L > 0$ is a regularization parameter.

Note that our subproblem (4) differs from those used in classical penalty methods [9], since we linearize the constraints and we also add a regularization term. Thus, the objective function of (4) is strongly convex. Hence, if Ω is nonempty and even if the problem is non-differentiable, this problem admits a unique optimal solution, and can be solved efficiently by several convex methods and solvers. For instance, alternating direction methods of multipliers (ADMM) [19] and primal-dual schemes [20] can be efficient for solving (4). Note that the convergence guarantees of ADMM and primal-dual schemes often depends on the distance between the initial point $\boldsymbol{x}^{k,0}$ of the algorithm and the exact optimal solution of $\bar{\boldsymbol{x}}^{k+1}$ of (4), see, e.g, [20, Theorem 2]. Hence, if we warm-start $\boldsymbol{x}^{k,0}$ at the previous approximate solution $\boldsymbol{x}^k$ obtained at the $(k-1)$-th iteration, then the distance $\|\boldsymbol{x}^0 - \bar{\boldsymbol{x}}^{k+1}\|$ is small. This allows the algorithm to converge faster to a desired approximate solution $\boldsymbol{x}^{k+1}$ of (4) at the k-th iteration.

Let us define: $\boldsymbol{V}_L(\boldsymbol{x}^k) := \operatorname{argmin}_{x\in\Omega} \Big\{ \mathcal{Q}_L(\boldsymbol{x};\boldsymbol{x}^k) \Big\}$. And the following quantities:

$$\begin{aligned}&\boldsymbol{G}_L(\boldsymbol{x}^k) := L(\boldsymbol{x}^k - \boldsymbol{V}_L(\boldsymbol{x}^k)),\\ &\boldsymbol{d}_L(\boldsymbol{x}^k) := \boldsymbol{V}_L(\boldsymbol{x}^k) - \boldsymbol{x}^k,\ r_L(\boldsymbol{x}^k) := \|\boldsymbol{d}_L(\boldsymbol{x}^k)\|.\end{aligned}$$

The necessary and sufficient optimality condition for subproblem (4) becomes

$$\begin{aligned}&\big[\nabla f(\boldsymbol{V}_L(\boldsymbol{x}^k)) - \boldsymbol{G}_L(\boldsymbol{x}^k) + \beta\Psi'(\boldsymbol{x}^k)\xi(\boldsymbol{x}^k)\big]^\top (\hat{\boldsymbol{x}}\\ &-\boldsymbol{V}_L(\boldsymbol{x}^k)) \geq 0, \quad \forall \hat{\boldsymbol{x}} \in \Omega,\end{aligned}$$

where $\xi(\boldsymbol{x}^k) \in \partial|\Psi(\boldsymbol{x}^k) + \Psi'(\boldsymbol{x}^k)(\boldsymbol{V}_L(\boldsymbol{x}^k) - \boldsymbol{x}^k)|$. $\boldsymbol{G}_L(\cdot)$ can be considered as a gradient mapping of F in (2) [13], and $\boldsymbol{d}_L(\boldsymbol{x}^k)$ is a search direction for Algorithm 1. Now, using the subproblem (4) as a main component, we describe our GN scheme in Algorithm 1.

Algorithm 1 *The Basic Gauss-Newton Algorithm*

1: **Initialization:** Choose $\boldsymbol{x}^0 \in \Omega$ and a penalty parameter $\beta > 0$ sufficiently large (ideally, $\beta > \|\boldsymbol{y}^*\|_\infty$).
2: Choose a lower bound $L_{\min} \in (0, \beta L_\Psi]$.
3: **For** $k := 0$ **to** $k_{\max}$ **perform**
4: Find $L_k \in [L_{\min}, \beta L_\Psi]$ such that $F(\boldsymbol{V}_{L_k}(\boldsymbol{x}^k)) \leq \mathcal{Q}_{L_k}(\boldsymbol{V}_{L_k}(\boldsymbol{x}^k);\boldsymbol{x}^k)$.
5: Update $\boldsymbol{x}^{k+1} := \boldsymbol{V}_{L_k}(\boldsymbol{x}^k)$.
6: Update β if necessary.
7: **End for**

The main step of Algorithm 1 is the solution of the convex subproblem (4) at Step 4. As mentioned, this problem is strongly convex, and can be solved by several methods that converge linearly. If we choose $L_k \equiv L \geq \beta L_\Psi$, then we do not need to perform a line-search on L at Step 4, and only need to solve (4) once per iteration. However, L_Ψ may not be known or if it is known, the global upper bound βL_Ψ may be too conservative, i.e. it does not take into account the local structures of non-linear functions in (2). Therefore, following the algorithm in [13], we propose performing a line-search in order to find an appropriate L_k. If we perform a line-search by doubling L_k at each step starting from $L_{\min}$, (i.e., $L_k \to 2L_k$), then after i_k line-search steps, we have $L_k = 2^{i_k} L_{\min}$, and the number of line-search iterations i_k is at most $\lfloor \log_2(\beta L_\Psi / L_{\min}) \rfloor + 1$. Note that it is rather straightforward to estimate $L_{\min}$. For example, we can set $L_{\min} := \frac{c\beta\|\Psi'(\hat{\boldsymbol{x}}^0) - \Psi'(\boldsymbol{x}^0)\|}{\|\hat{\boldsymbol{x}}^0 - \boldsymbol{x}^0\|} \leq \beta L_\Psi$ for some $\hat{\boldsymbol{x}}^0 \neq \boldsymbol{x}^0$ and $c \in (0, 1]$. A detailed discussion on tuning β and L, as well as convergence analysis and proofs are provided in [17].

III. Applying the GN Algorithm to AC-OPF

In this section, we present the OPF problem and its reformulation in a form that obeys the structure presented in the previous section. We then perform numerical experiments to validate the algorithm and compare it with IPOPT.

A. Problem settings

a) Original AC-OPF: Consider a directed electric power network with a set of nodes $\mathcal{B}$ and a set of branches $\mathcal{L}$. The network consists of a set $\mathcal{G}$ of generators, with $\mathcal{G}_i$ denoting the set of generators at bus i. Denote $Y = G + jB$ as the system admittance matrix, G being the conductance and B the susceptance ($j^2 = -1$) [21]. The decision variables of AC-OPF are the real and reactive power outputs of generators, which are denoted as $\boldsymbol{p} \in \mathbb{R}^{|\mathcal{G}|}$ and $\boldsymbol{q} \in \mathbb{R}^{|\mathcal{G}|}$, the voltage magnitudes $\boldsymbol{v} \in \mathbb{R}^{|\mathcal{B}|}$ and phase angles $\boldsymbol{\theta} \in \mathbb{R}^{|\mathcal{B}|}$. We will consider a fixed real and reactive power demand at every node i, which we denote as P_i^d and Q_i^d, respectively. The constraints of the AC-OPF problem can be described as follows [6]:

$$\begin{aligned}\sum_{j\in\mathcal{G}_i} p_j - P_i^d - \sum_{j\in\mathcal{B}} v_i v_j \Big(&G_{ij}\cos(\theta_i - \theta_j)\\ &+ B_{ij}\sin(\theta_i - \theta_j)\Big) = 0, \qquad \forall i \in \mathcal{B}, \quad (5a)\end{aligned}$$

$$\sum_{j\in\mathcal{G}_i} q_j - Q_i^d - \sum_{j\in\mathcal{B}} v_i v_j \Big(G_{ij}\sin(\theta_i-\theta_j) - B_{ij}\cos(\theta_i-\theta_j)\Big) = 0, \quad \forall i\in\mathcal{B}, \quad (5b)$$

$$(\boldsymbol{p},\boldsymbol{q},\boldsymbol{v},\boldsymbol{\theta})\in\mathcal{C}. \quad (5c)$$

Constraints (5a) and (5b) correspond to the real and reactive power balance equations of node i. Constraints (5c) gathers the set of classical inequality constraints on power systems such as generator capacity limits, line flow limits or voltage limits. It is common to assume that the set $\mathcal{C}$ is convex and can be casted as a second order cone set of constraints.

We will consider the objective of minimizing real power generation costs modeled as a convex quadratic function f:

$$f(\boldsymbol{p}) = \boldsymbol{p}^\top \mathrm{diag}(\boldsymbol{C_2})\boldsymbol{p} + \boldsymbol{C_1}^\top \boldsymbol{p},$$

where $\boldsymbol{C_2} \geq 0$ and $\boldsymbol{C_1}$ are given coefficients of the cost function. The AC OPF problem then reads as the following non-convex optimization problem:

$$\mathcal{P}_{opt}: \quad \min_{(\boldsymbol{p},\boldsymbol{q},\boldsymbol{v},\boldsymbol{\theta})} f(\boldsymbol{p}) \quad \text{subject to (5)}.$$

b) Quadratic reformulation: The starting point of our proposed GN method for solving problem $\mathcal{P}_{opt}$ is the quadratic reformulation of AC-OPF [14]. In this reformulation, we replace the voltage magnitudes and angles $v_i\angle\theta_i = v_i e^{j\theta_i} = v_i(\cos(\theta_i)+j\sin(\theta_i))$ by a new set of variables c_{ij} and s_{ij}. These new variables are defined for $i\in\mathcal{B}$ and $(i,j)\in\mathcal{L}$ as:

$$c_{ii} = v_i^2,\ c_{ij} = v_i v_j \cos(\theta_i-\theta_j), \quad (6a)$$

$$s_{ii} = 0,\ s_{ij} = -v_i v_j \sin(\theta_i-\theta_j), \quad (6b)$$

where we will denote the vectors $\boldsymbol{c}$ and $\boldsymbol{s}$ as the collection of the c_{ij} and s_{ij} variables, respectively.

Assuming $-\pi/2 \leq \theta_i \leq \pi/2, \forall i\in\mathcal{B}$ (which is common practice [**?**]), the mapping from $(\boldsymbol{v},\boldsymbol{\theta})$ to $(\boldsymbol{c},\boldsymbol{s})$ defined by (6) can be inverted as follows:

$$v_i = \sqrt{c_{ii}}, \quad \theta_i - \theta_j = \mathrm{atan2}(s_{ij}, c_{ij}),$$

thereby defining a bijection in $(\boldsymbol{c},\boldsymbol{s})$ and $(\boldsymbol{v},\boldsymbol{\theta})$ [6]. The set of $(\boldsymbol{c},\boldsymbol{s})$ and $(\boldsymbol{v},\boldsymbol{\theta})$ that define this bijection is further equivalent to the following set of non-linear non-convex constraints [6]:

$$c_{ij}^2 + s_{ij}^2 = c_{ii}c_{jj}, \quad \forall (i,j)\in\mathcal{L}, \quad (7a)$$

$$\sin(\theta_i-\theta_j)c_{ij} + \cos(\theta_i-\theta_j)s_{ij} = 0, \quad \forall (i,j)\in\mathcal{L}. \quad (7b)$$

Now, we will substitute the voltage magnitude variables into the problem $\mathcal{P}_{opt}$, and consider the problem on the variables $(\boldsymbol{c},\boldsymbol{s},\boldsymbol{\theta})$. This reformulation has been commonly employed in the literature in order to arrive at an SOCP (Second-Order Cone Programming) relaxation of the problem [5], [14]. Through numerical experiments, we demonstrate that this reformulation results in highly effective starting points for our algorithm based on the SOCP relaxation of the AC-OPF. Moreover, the reformulation preserves the power balance constraints in linear form. Thus, the power balance constraints are not penalized in our scheme, which implies that they are respected at every iteration of the algorithm. For all these reasons, we pursue the quadratic reformulation of the present section, despite the fact that it requires the introduction of the new variables $\boldsymbol{c}$ and $\boldsymbol{s}$. Concretely, the AC OPF constraints (5) are substituted using $\boldsymbol{c}$ and $\boldsymbol{s}$ by:

$$\sum_{j\in\mathcal{G}_i} p_j - P_i^d - \sum_{j\in\mathcal{B}} (G_{ij}c_{ij} - B_{ij}s_{ij}) = 0, \quad \forall i\in\mathcal{B}, \quad (8a)$$

$$\sum_{j\in\mathcal{G}_i} q_j - Q_i^d + \sum_{j\in\mathcal{B}} (G_{ij}s_{ij} + B_{ij}c_{ij}) = 0, \quad \forall i\in\mathcal{B}, \quad (8b)$$

$$(\boldsymbol{c},\boldsymbol{s},\boldsymbol{\theta},\boldsymbol{p},\boldsymbol{q})\in\mathcal{C}^Q, . \quad (8c)$$

where $\mathcal{C}^Q$ is the still the set of inequality constraints, adapted to the quadratic formulation. (8) defines a convex set of constraints. As a result, an equivalent formulation for the AC-OPF model $\mathcal{P}_{opt}$ is:

$$\mathcal{P}_{opt}^{cs\theta}: \min_{(\boldsymbol{p},\boldsymbol{q},\boldsymbol{c},\boldsymbol{s},\boldsymbol{\theta})} f(\boldsymbol{p}) \quad \text{s.t. (7), (8)}.$$

We then have two non-convex equality constraints:

- Constraints (7a): $\Psi_q^{ij}(\boldsymbol{c},\boldsymbol{s}) := c_{ij}^2 + s_{ij}^2 - c_{ii}c_{jj} = 0,\ \forall (i,j)\in\mathcal{L}$. We refer to them as *quadratic* constraints.
- Constraints (7b): $\Psi_t^{ij}(\boldsymbol{c},\boldsymbol{s},\boldsymbol{\theta}) := \sin(\theta_i-\theta_j)c_{ij} + \cos(\theta_i-\theta_j)s_{ij} = 0,\ \forall (i,j)\in\mathcal{L}$. We refer to them as *trigonometric* constraints.

Now, AC-OPF is written in the format of (1), where $\boldsymbol{x} := (\boldsymbol{p},\boldsymbol{q},\boldsymbol{c},\boldsymbol{s},\boldsymbol{\theta})$, $f(\boldsymbol{x}) := f(\boldsymbol{p})$, $\Psi(\boldsymbol{x}) := (\Psi_q^{ij}(\boldsymbol{c},\boldsymbol{s}), \Psi_t^{ij}(\boldsymbol{c},\boldsymbol{s},\boldsymbol{\theta}))$ and $\Omega := \{\boldsymbol{x}\in\mathbb{R}^d \mid \boldsymbol{x} \text{ satisfies (8)}\}$.

B. A practical implementation of the GN algorithm for OPF

In this section, the goal is to optimize the settings of the GN method. We will demonstrate that the choice of β and L is crucial. This will allow us to derive a practical version of the GN algorithm, which we compare to IPOPT.

a) Stopping criteria: We terminate Algorithm 1 in two occasions, which have been validated through experimental results: (i) if the maximum number of iterations $k_{\max} := 100$ is reached; (ii) if the quadratic and trigonometric constraints are satisfied with a tolerance of ϵ_2, where $\epsilon_2 := 1e^{-5}$ (i.e. if $\max(\|\Psi_q(c^k,s^k)\|_\infty, \|\Psi_t(c^k,s^k,\theta^k)\|_\infty) \leq \epsilon_2$). If the difference $\|x^{k+1}-x^k\|_\infty < \epsilon_1$ ($\epsilon_1 := 1e^{-6}$ in the numerical experiment), then Algorithm 1 has reached an approximate stationary point of the exact penalized formulation (2). In this case, the last iterate might not be feasible for $\mathcal{P}_{opt}^{cs\theta}$. We then use the run-and-inspect strategy [22]: the last iterate becomes the starting point of GN and β is doubled.

b) SOCP relaxation for initialization: As mentioned previously, the quadratic formulation is also used to derive the SOCP relaxation. In this relaxation, the angles $\boldsymbol{\theta}$ are not modeled and the trigonometric constraints (7b) are removed. Moreover, the non-convex constraints (7a) are relaxed:

$$c_{ij}^2 + s_{ij}^2 \leq c_{ii}c_{jj}, \quad \forall (i,j)\in\mathcal{L}. \quad (9)$$

Then, $\mathcal{P}_{socp}$ is defined such that:

$$\mathcal{P}_{socp}: \min_{(\boldsymbol{p},\boldsymbol{q},\boldsymbol{c},\boldsymbol{s})} f(\boldsymbol{p}) \quad \text{s.t. (8), (9)}.$$

Solving this relaxation will provide a partial initial point $(\boldsymbol{p}^0,\boldsymbol{q}^0,\boldsymbol{c}^0,\boldsymbol{s}^0)$ (and $\boldsymbol{\theta}^0 = 0$).

c) Parameter tuning strategies: The convergence theory presented above does not require tuning the β and L parameters. In practice, tuning is crucial for improving the performance of algorithms for constrained non-convex optimization, including Algorithm 1. Several observations allow us to decrease the number of iterations that are required for convergence: (i) according to Proposition II.2, large values of β ensure the equivalence between (1) and (2); (ii) quadratic constraints and trigonometric constraints scale up differently; (iii) a careful update of L influences the number of times that subproblem (4) is solved. These observations guide a detailed investigation concerning the choices of β and L parameters [17].

d) Acceleration through warmstart: We observe that the subproblems (4) are based on the same formulation, and only differ by slight changes of certain parameters along the iterations. This motivates us to warm-start the subproblem (4) with a previous primal-dual iterate. In other words, we initialize the solver for solving the subproblem (4) at the k-th iteration at the final solution $\boldsymbol{x}_{k-1}$ and its corresponding multiplier $\boldsymbol{y}_{k-1}$ obtained from the previous iteration $k-1$. Warm-starting is indeed a key step in iterative methods, including our GN scheme, and will be further analyzed in Section IV-2.

IV. Numerical Experiments

In order to validate the proposed GN algorithm, our numerical experiments are conducted in 2 steps: first, we launch simulations on several test cases of a classical library (MATPOWER) and compare the GN algorithm with a state-of-the-art non-convex solver (IPOPT); second, we show the potential benefit of warm-start for our approach.

1) Illustration on MATPOWER instances: We use the MATPOWER [15] library to have access to a wide range of AC-OPF test systems that have been investigated in the literature. We test our approach on instances whose size ranges between 1,354 and 25,000 nodes (`1354pegase` has 11,192 variables and 27,911 constraints while `ACTIVS25k` has 186,021 variables and 431,222 constraints). We benchmark our approach against IPOPT, a non-linear solver based on the interior-point method. To do so, we make use of `PowerModels.jl` [23], a Julia package that can be used to solve AC-OPF instances of different libraries with different formulations. In order to make a fair comparison, we initialize GN and IPOPT using the SOCP solution. The experiments are conducted in Julia (version 1.1.1) using JuMP on a MacBook Pro 2016, with a 2.9 GHz Dual-Core Intel Core i5 processors.

The results of our analysis are presented in Table I. In Table I, for each test case (first column), we report the objective value and the execution time (in seconds). For GN, we also report the number of iterations. The last column provides the gap between the GN solution and the IPOPT solution.

The first notable observation is that GN finds a stationary point (i.e. feasible) of the original AC-OPF problem for all 23 test cases. The stationary point obtained by GN attains the same objective function value as the one returned by IPOPT for most instances (a difference of 0.05% in the objective value may be attributed to numerical precision) and the proposed method outperforms IPOPT in some instances (e.g. `2737sop`).

TABLE I
Comparison of the GN algorithm against IPOPT

	Gauss-Newton			IPOPT		
Test Case	# It	Objective	Time	Objective	Time	Gap
`1354pegase`	13	$7.407e^4$	15.1 s	$7.407e^4$	**6.00** s	0.0 %
`1888rte`	14	$5.981e^4$	**25.9** s	$\mathbf{5.980e^4}$	59.0 s	0.0 %
`1951rte`	4	$8.174e^4$	**6.94** s	$8.174e^4$	7.67 s	0.0 %
`ACTIVSg2000`	5	$1.229e^6$	**8.39** s	$1.229e^6$	14.8 s	0.0 %
`2383wp`	19	$1.868e^6$	51.0 s	$1.868e^6$	**21.6** s	0.0 %
`2736sp`	4	$\mathbf{1.307e^6}$	**10.9** s	$1.308e^6$	13.3 s	-0.1 %
`2737sop`	3	$\mathbf{7.767e^5}$	**7.77** s	$7.778e^5$	9.98 s	-0.1 %
`2746wop`	17	$1.208e^6$	38.1 s	$1.208e^6$	**12.8** s	0.0 %
`2746wp`	3	$\mathbf{1.631e^6}$	**6.89** s	$1.632e^6$	16.0 s	-0.1 %
`2848rte`	17	$5.303e^4$	**52.0** s	$\mathbf{5.302e^4}$	74.3 s	0.0 %
`2868rte`	4	$7.980e^4$	**9.11** s	$\mathbf{7.979e^4}$	29.3 s	0.0 %
`2869pegase`	12	$1.340e^5$	41.9 s	$1.340e^5$	**15.5** s	0.0 %
`3012wp`	8	$2.593e^6$	25.2 s	$\mathbf{2.592e^6}$	**18.8** s	0.0 %
`3120sp`	13	$\mathbf{2.142e^6}$	41.2 s	$2.143e^6$	**19.2** s	0.0 %
`3375wp`	8	$7.413e^6$	34.7 s	$\mathbf{7.412e^6}$	**21.3** s	0.0 %
`6468rte`	19	$8.685e^4$	187 s	$\mathbf{8.683e^4}$	**103** s	0.0 %
`6470rte`	11	$9.835e^4$	**89.3** s	$9.835e^4$	144 s	0.0 %
`6495rte`	12	$1.063e^5$	180 s	$1.063e^5$	**52.4** s	0.0 %
`6515rte`	18	$1.098e^5$	204 s	$1.098e^5$	**86.9** s	0.0 %
`9241pegase`	17	$3.167e^6$	894 s	$\mathbf{3.159e^6}$	**368** s	0.3 %
`ACTIVSg10k`	6	$2.488e^6$	117 s	$\mathbf{2.486e^6}$	**93.4** s	0.1 %
`13659pegase`	19	$3.885e^5$	**137** s	$\mathbf{3.861e^5}$	685 s	0.6 %
`ACTIVSg25k`	16	$6.033e^6$	1,740 s	$\mathbf{6.018e^6}$	**544** s	0.3 %

Our experiments further demonstrate that the GN method consistently requires a small number of iterations (less than 20) in a wide range of instances. This is critically important to further accelerate the performance of our method if we appropriately exploit warm-start strategies and efficient solvers for the strongly convex subproblem.In terms of computational time, the performance is shared between the two approaches. Nevertheless, some instances reveal limitations of the GN algorithm, compared to IPOPT (`6495rte`, `9241pegase` and `ACTIVSg25k` for example): when the solution of a subproblem becomes time-consuming because of the size of the subproblem, GN might require a larger execution time. We use Gurobi for solving subproblem (4), because it is one of the most stable QCP solvers that are available.

Unfortunately, Gurobi (and IPMs for QCPs in general) does not support warm-start, which would have significantly decreased the computational time. One alternative is to use an ADMM solver that supports warm-start. However, ADMM solvers are not mature enough to test large-scale problems. Implementing an efficient subsolver is out of the scope of this work, however we are able to analyze the effect of warm start on these solvers which is the subject of the next section.

2) The effect of warm-starting strategy: We consider using OSQP [24] as an ADMM solver. We consider one small test case (`118_ieee`) since we observed numerical instability for larger test cases. We examine each GN iteration individually, and highlight the impact of warm-start on the number of ADMM iterations in Fig. 1. Note that we warm-start dual and primal variables only after iteration 1. Warm-start decreases substantially the number of ADMM iterations in two cases: (i) When L is updated. Indeed, updating L only results in slightly changing the objective function. One expects the previous

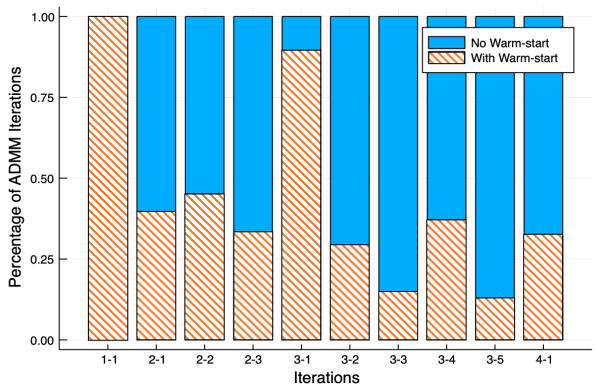

Fig. 1. Evolution of the percentage of ADMM iterations along the iterations for `118_ieee`. The percentage of ADMM iterations 'With Warm-start' is measured relatively to 'No Warm-start'. GN iterations are shown on the x axis in an $a-b$ format: a is the actual GN iteration and b represents the bth subproblem that had to be solved at iteration a because of an update of L.

iterates to provide a good warm-start. This is confirmed in Fig. 1, where we observe that the number of ADMM iterations is divided by at least a factor of 2 every time L is updated;(ii) when the last iterates are computed. Intuitively, one does not expects iterates to change substantially when approaching the optimal solution. This intuition is confirmed by Fig. 1. For the particular case of the last iterate, the required number of ADMM iterations is less than 30%. Globally, we observe that warm start divides the total number of ADMM iterations by almost 3 for `118_ieee`. This investigation suggests that, with a mature ADMM solver, warm-starting is a promising feature for improving the performance of GN on large test cases.

V. Conclusion

We propose a novel Gauss-Newton algorithm for solving a general class of optimization problems with non-convex constraints. We utilize an exact non-smooth penalty reformulation of the original problem and suggest an iterative scheme for solving this penalized problem which relies on the non-squared Gauss-Newton method. The subproblems of our proposed scheme are strongly convex programs, which can be efficiently solved by numerous third-party convex optimization solvers.

We apply our proposed approach to solve AC-OPF, which is a fundamental and ubiquitous problem in power systems engineering. We apply our proposed algorithm to a reformulation of the AC-OPF, and we propose numerous strategies for tuning our proposed GN scheme, initializing the algorithm, and warm-starting the resolution of the subproblems that are treated by our proposed method. We perform extensive numerical experiments on a large set of instances from the MATPOWER library, and demonstrate the competitive performance of our method to IPOPT, which is a state of the art non-linear non-convex solver. This analysis validates the theoretical analysis of our proposed GN scheme, and proves its effectiveness in practical applications.

References

[1] J. Carpentier, "Contribution a letude du dispatching economique," Bulletin de la Societe Francaise des Electriciens, vol. 3, no. 1, 1962.

[2] S. Frank, I. Steponavice, and S. Rebennack, "Optimal power flow: a bibliographic survey i," Energy Systems, vol. 3, no. 3, 2012.

[3] H. Le Cadre, I. Mezghani, and A. Papavasiliou, "A game-theoretic analysis of transmission-distribution system operator coordination," European Journal of Operational Research, vol. 274, no. 1, pp. 317–339, 2019.

[4] J. Lavaei and S. H. Low, "Zero duality gap in optimal power flow problem," IEEE Transactions on Power Systems, vol. 27, no. 1, pp. 92–107, 2011.

[5] R. A. Jabr, "Optimal power flow using an extended conic quadratic formulation," IEEE transactions on power systems, vol. 23, no. 3, pp. 1000–1008, 2008.

[6] B. Kocuk, S. S. Dey, and X. A. Sun, "Strong socp relaxations for the optimal power flow problem," Operations Research, vol. 64, no. 6, 2016.

[7] C. Coffrin, H. L. Hijazi, and P. Van Hentenryck, "The qc relaxation: A theoretical and computational study on optimal power flow," IEEE Transactions on Power Systems, vol. 31, no. 4, pp. 3008–3018, 2015.

[8] D. K. Molzahn, J. T. Holzer, B. C. Lesieutre, and C. L. DeMarco, "Implementation of a large-scale optimal power flow solver based on semidefinite programming," IEEE Transactions on Power Systems, vol. 28, no. 4, pp. 3987–3998, 2013.

[9] J. Nocedal and S. Wright, Numerical Optimization, 2nd ed., ser. Springer Series in Operations Research and Financial Engineering. Springer, 2006.

[10] Y. Nesterov and B. T. Polyak, "Cubic regularization of newton method and its global performance," Mathematical Programming, vol. 108, no. 1, pp. 177–205, 2006.

[11] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," in Advances in neural information processing systems, 2014, pp. 2933–2941.

[12] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," Mathematical programming, vol. 106, no. 1, pp. 25–57, 2006.

[13] Y. Nesterov, "Modified gauss–newton scheme with worst case guarantees for global performance," Optimisation Methods and Software, vol. 22, no. 3, pp. 469–483, 2007.

[14] A. G. Expósito and E. R. Ramos, "Reliable load flow technique for radial distribution networks," IEEE Transactions on Power Systems, vol. 14, no. 3, pp. 1063–1069, 1999.

[15] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, "Matpower: Steady-state operations, planning, and analysis tools for power systems research and education," IEEE Transactions on power systems, vol. 26, no. 1, pp. 12–19, 2010.

[16] A. Eltved, J. Dahl, and M. S. Andersen, "On the robustness and scalability of semidefinite relaxation for optimal power flow problems," Optimization and Engineering, pp. 1–18, 2018.

[17] I. Mezghani, Q. Tran-Dinh, I. Necoara, and A. Papavasiliou, "A globally convergent gauss-newton algorithm for ac optimal power flow," arXiv preprint arXiv:1905.08588, 2019.

[18] G. D. Pillo, "Exact penalty methods," in Algorithms for Continuous Optimization. Springer, 1994, pp. 209–253.

[19] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," Foundations and Trends® in Machine learning, vol. 3, no. 1, pp. 1–122, 2011.

[20] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," J. Math. Imaging Vis., vol. 40, no. 1, pp. 120–145, 2011.

[21] J. A. Taylor, Convex optimization of power systems. Cambridge University Press, 2015.

[22] Y. Chen, Y. Sun, and W. Yin, "Run-and-inspect method for nonconvex optimization and global optimality bounds for r-local minimizers," Mathematical Programming, vol. 176, no. 1-2, pp. 39–67, 2019.

[23] C. Coffrin, R. Bent, K. Sundar, Y. Ng, and M. Lubin, "Powermodels.jl: An open-source framework for exploring power flow formulations," in 2018 Power Systems Computation Conference (PSCC), June 2018.

[24] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," ArXiv e-prints, Nov. 2017.