A twist on SLP algorithms for NLP and MINLP problems: An application to gas transmission networks

Ángel M. González Rueda[1,3], Julio González Díaz[1,2,3], and María P. Fernández de Córdoba[1]

[1] *Department of Statistics, Mathematical Analysis and Optimization. University of Santiago de Compostela.*
[2] *IMAT and ITMATI.*
[3] *MODESTYA Research Group.*

Published in Optimization and Engineering 20, 349–395 (2019)
Published version available at https://link.springer.com/journal/11081
DOI 10.1007/s11081-018-9407-4

Abstract

This paper presents a modification of classic SLP algorithms for the resolution of NLP and MINLP problems, and does it with a clear application in mind: optimization of gas transmission networks.

The SLP-NTR and 2-step SLP algorithms we present have been developed within the collaboration with a company of the gas industry and thoroughly tested with real problems in this field. Here we present a comparison of their performance with that of classic SLP algorithms and state of the art solvers.

Importantly, to provide some foundations for the potential applicability of these new algorithms to general NLP and MINLP problems, we present a theoretical analysis of their properties.

Keywords. Gas transmission networks, Optimization, Sequential linear programming, NLP problems, MINLP problems

MSC2010 classification codes. 90C30, 90C11, 90C90, 76N25

1 Introduction

Energy networks are becoming more and more prevalent worldwide and, therefore, an efficient management of these infrastructures is crucial to get the most out of the different sources of energy: gas, oil, electricity,... In particular, gas transmission networks have already been studied for a long time and two recent references on the optimization problems that arise in this field are Ríos-Mercado and Borraz-Sánchez (2015) and Koch et al. (2015).

This paper stems from the collaboration between ITMATI (Technological Institute for Industrial Mathematics) and a Spanish company operating in the gas industry: Reganosa, a Transport System Operator who owns part of the Spanish gas transmission network. One of the main problems tackled during this collaboration is precisely the optimization of gas transmission networks in steady state, and the algorithms we present in Section 3 are routinely used by the company.

This version: May 6, 2019

In particular, these algorithms are at the core of GANESO$^{\text{TM}}$ software, which stands for GAs NEtwork Simulation and Optimization, developed for and owned by Reganosa Company.[1]

The contribution of this paper to the literature is twofold: a first one dealing with some new ideas in the modeling of gas networks and a second one is more methodological, dealing with the resolution technique of the mixed integer nonlinear optimization problems associated to the gas network models.

In Section 2 we start by introducing a relatively standard baseline model for the optimization of gas transmission networks. Then, our contribution consists of some novel approaches in the optimization-driven modeling of gas transmission networks, such as the explicit modeling of boil-off costs at regasification plants and some novel graph-representations for compressor stations. These and other elements have been successfully incorporated and tested as part of the functionalities of GANESO$^{\text{TM}}$.

The second part of the contribution is methodological, consisting of a twist to classic sequential linear programming (SLP) techniques that has delivered very competitive results when compared to state of the art solvers, as reported in Section 5. The optimization problems associated to gas transmission networks are highly nonlinear and nonconvex, mainly because of the nature of the pressure loss constraints that account for the effect of the friction of the gas with the walls of the pipes. Classic SLP algorithms solve, at each iteration, a linearization of the underlying NLP problem, obtaining a solution that is then taken as the basis for the linearization in the next iteration. If such an algorithm converges, the limit is a KKT point of the original NLP problem (see, for instance, Kim et al. (1985)). In order to improve the convergence properties of these algorithms, a trust region is included in the linear subproblems. What we propose in this paper is a 2-step SLP algorithm in which the trust region is removed in the first step, in which what we call SLP-NTR algorithm is run, and brought back in the second one, where a more classic SLP algorithm is used.

Our practical experience has shown that, for the optimization NLP problems defined on gas transmission networks, the 2-step SLP algorithm has very good behavior, delivering solutions that outperform not only the one-step SLP algorithms, but also state of the art solvers such as Knitro, Ipopt, and BARON.

Importantly, some of the resulting problems associated to gas network optimization problems may contain binary and integer variables (to model elements such as control valves, compressors, boil-off costs, operational ranges,...). Then, to solve the resulting MINLP problems, classic SLP algorithms do not work. An important advantage of the SLP-NTR algorithm is that it can be readily adapted as a heuristic for MINLP problems, and so does the 2-step SLP algorithm. For the latter, the binary and integer variables are fixed in the second step to the values obtained by the SLP-NTR algorithm. This two-step heuristic for MINLP has also been satisfactorily tested on different problems associated to gas transmission networks and, given our experience with it, we conjecture that it can be especially useful for problems in which the integer variables represent a relatively small proportion of the total number of variables.

As part of the above methodological contribution we develop, in Section 4, a theoretical analysis of the properties of the SLP-NTR algorithm run in the first step of the 2-step SLP algorithm. We prove that limit points of the sequence generated by the algorithm are KKT points of NLP and, further, even if the sequence does not converge, if two points in the sequence are sufficiently close, one of them is almost a KKT point in a sense that is properly formalized in Section 4.

We have chosen to start our analysis in Section 2 with the discussion regarding the modeling of gas transmission networks and then move to the methodological contribution on SLP algorithms in Section 3 and Section 4. Yet, the latter two sections are self-contained, so those readers

[1] See http://www.reganosa.com/en/ganeso.

mainly interested on the mathematical programming aspects behind these algorithms may jump directly to Section 3. Section 5 presents a computational analysis of the performance of these new SLP algorithms on gas transmission networks, bringing together the contents of the preceding sections. The paper concludes with some conclusions and future work.

2 Optimization of gas transmission networks

As we have mentioned in the Introduction, there is already a large body of literature on the physical modeling of gas networks, both in the stationary case and in the transient one, and there are also relatively standard simplifications to obtain realistic optimization models that can be handled with state of the art techniques. In this context, given that the main contribution of this paper lies on the resolution approach, we won't delve deeply on the formal derivation of the mathematical constraints but the interested reader may refer, for instance, to González-Rueda (2017), González-Diéguez (2017), and Möller (2004) for three PhD thesis that thoroughly discuss the physics behind each constraint.[2]

There are three important goals that we want to accomplish in this section: i) to introduce the (standard) elements of the baseline optimization model for the stationary case, ii) to present some novel elements in the modeling of gas transmission networks, arising as a result of the interaction with the Spanish company Reganosa and that have delivered satisfactory results within that collaboration, and iii) to discuss the resulting optimization model and motivate the chosen algorithmic approach, which is then formally developed in Section 3.

2.1 Baseline model

The baseline model we present in this section can be summarized as follows. Given a gas network in which consumers have some demands and suppliers some capacities, provide the feasible flow configuration that satisfies the demand, delivering the gas within some pre-specified pressure bounds, and that has a minimal cost, measured by the energy consumption at compressors.

Preliminary concepts and notations

We model a gas transmission network as a directed graph $\mathcal{G} = (N, E)$, where N is the set of nodes and $E \subset N \times N$ is the set of edges. The set E is partitioned into three subsets, E^{P}, E^{C}, and E^{V}, denoting the sets of pipes, compressors, and valves, respectively. The elements of N represent supply nodes, demand nodes, and structural nodes (intersections between pipes at which no gas is exchanged with the outside of the network).

Given a node $i \in N$, we denote by $E_i^{\text{INI}} \subset E$ and $E_i^{\text{FIN}} \subset E$ the sets of edges having i as initial and final node, respectively.

Variables

The main variables of the problem are the following:

- For each edge $k = (i, j) \in E$, $q_k \in \mathbb{R}$ denotes the (mass) flow rate [kg/s] through edge k. A positive flow represents gas flowing from (i, j), whereas a negative flow represents gas going in the opposite direction.[3]

[2] Like this paper, the first two references have been developed within the collaboration between Reganosa company and ITMATI.

[3] In some networks one can know in advance the flow direction in all the edges of the network, which greatly simplifies the analysis (in particular, the need of binary variables is notably reduced). In the case of the Spanish

- For each node $i \in N$, $u_i \geq 0$ denotes the square of the gas pressure [Pa^2] at node i. Alternatively, one could use the pressure itself, $p_i \geq 0$, but for our modeling it turns out that the squared pressure is more convenient (particularly for the pressure loss constraints, see Equation (PL).

Auxiliary variables introduced to facilitate the presentation of the optimization model:

- For each pipe $k \in E^{\text{P}}$, $p_k >= 0$ denotes the average pressure [Pa] at pipe k.
- For each compressor $k \in E^{\text{C}}$, g_k denotes the self-consumption [kg/s] of gas at compressor k. This self-consumption will be the cost to minimize in our model (see Equation (Obj)).

Binary variables needed for the controllable elements of the network:

- For each valve $k \in E^{\text{V}}$, we have a binary variable y_k^{V}.
- It will be discussed later that, depending on the modeling approach, we may also have a binary variable y_k^{C} associated to each compressor $k \in E^{\text{C}}$.

Main constraints

We start with the *box constraints*, which provide bounds for the variables:

$$\begin{aligned} u_i^{\text{LB}} \leq u_i \leq u_i^{\text{UB}}, \qquad & \forall i \in N, \\ q_k^{\text{LB}} \leq q_k \leq q_k^{\text{UB}}, \qquad & \forall k \in E. \end{aligned} \tag{BC}$$

The *average pressure* in a pipe k, p_k, can be computed in different ways. The main one that we use in our modeling is[4]

$$p_k = \frac{2}{3}\Big(\sqrt{u_i u_j} - \frac{\sqrt{u_i u_j}}{\sqrt{u_i} + \sqrt{u_j}}\Big), \quad k \in E^{\text{P}}. \tag{AP$_{\text{COM}}$}$$

Alternatively, one can rely on the less precise but simpler formula given by

$$p_k = \sqrt{(u_i + u_j)/2}, \quad k \in E^{\text{P}}. \tag{AP$_{\text{SIM}}$}$$

At the core of a gas transmission problem we have the standard *flow conservation constraints*:

$$c_i^{\text{LB}} \leq \sum_{k \in E^{\text{INI}}} q_k - \sum_{k \in E^{\text{FIN}}} q_k \leq c_i^{\text{UB}}, \quad \forall i \in N. \tag{FC}$$

A typical demand node will have $c_i^{\text{LB}} = c_i^{\text{UB}} < 0$, a typical supply node will have $c_i^{\text{LB}} = 0$ and $c_i^{\text{UB}} > 0$, whereas structural nodes have $c_i^{\text{LB}} = c_i^{\text{UB}} = 0$.

One of the main difficulties in the optimization problems associated to gas transmission networks arises from the nonlinearities associated to the *pressure loss* constraints:

$$u_i - u_j = \frac{16 L_k}{\pi^2 D_k^5} \lambda_k Z(p_k, \theta)\, R\, \theta |q_k| q_k + \frac{2g}{R\theta} \frac{p_k^2}{Z(p_k, \theta)}(h_j - h_i), \quad \forall k = (i,j) \in E^{\text{P}}. \tag{PL}$$

The following parameters are involved in the above constraint: L_k and D_k are the length [m] and the diameter [m] of the section of the pipe, R is the ideal gas constant [J/kg K], h_i and h_j

gas network, which is the one for which the methodologies in this paper were developed, the direction of the flow in most edges is not known in advance.

[4]This equation reflects the fact that the pressure does not decrease linearly along the pipe, since the pressure drop depends on the flow in a nonlinear way.

are the heights [m] at the endpoints of the pipe, g is the acceleration of the gravity [m/s^2], and θ is the average of the gas temperature in the pipe [K].[5] There are two elements in Equation (PL) that require additional discussion:

- *Compressibility factor.* We follow AGA8 model for the computation of $Z(p_k, \theta)$ (see Starling and Savidge (1992)):
$$Z(p, \theta) = 1 + 0.257\frac{p}{p_c} - 0.533\frac{p}{p_c}\frac{\theta_c}{\theta}, \tag{ZF}$$
where p_c [Pa] and θ_c [K] are the critical pressure and the critical temperature, respectively, of the gas.[6]

- *Friction factor.* We follow Weymouth equation (see Weymouth (1912)) for the computation of λ_k:
$$\lambda_k = 0.009427/(e_k D_k^{1/3}), \tag{FF}$$
where e_k [dimensionless] is a parameter representing the efficiency coefficient of pipe k and D_k denotes its diameter [m]. There are more precise formulas to compute the friction factor, such as those based on the so called Colebrook equation. Yet, since in these formulations λ_k depends on q_k and does it in a complex nonlinear way (indeed, it is typically computed through Newton-type algorithms), we restrict to Weymouth formulation for the analysis in this paper.[7]

Constraints associated to controllable elements

We move now to the main controllable elements typically associated to a gas transmission network: *control valves* and *compressors*. The role of the latter is to counterbalance the pressure loss at the pipes while the former are important to add flexibility to the gas network operation, increasing the gas routing possibilities. As we discuss below, the modeling of these elements usually requires the use of binary variables.

Control valves allow to reduce the pressure of the gas in the network. The constraint associated to a valve $k = (i, j) \in E^{\text{v}}$ would be $u_i - u_j \geq 0$ when $q_k > 0$ and $u_i - u_j \leq 0$ otherwise. This "conditional" constraint is modeled in Equation (CV) using a binary variable $y_k^{\text{v}} \in \{0, 1\}$ and adding four constraints:

$$\left.\begin{array}{rcl} q_k & \leq & q_k^{\text{UB}} y_k^{\text{v}} \\ q_k & \geq & q_k^{\text{LB}}(1 - y_k^{\text{v}}), \\ u_i - u_j & \leq & M_1 y_k^{\text{v}}, \\ u_i - u_j & \geq & M_2(y_k^{\text{v}} - 1). \end{array}\right. \quad \forall k = (i, j) \in E^{\text{v}}. \tag{CV}$$

The first two constraints ensure that $y_k^{\text{v}} = 1$ if $q_k > 0$ and $y_k^{\text{v}} = 0$ if $q_k < 0$. Constraints three and four ensure that control valves only decrease the pressure in the direction of the flow. Therefore, if $q_k > 0$ ($y_k^{\text{v}} = 1$), gas flow goes from i to j and the fourth constraint imposes $u_i \geq u_j$. Analogously, if $q_k < 0$ ($y_k^{\text{v}} = 1$), gas flow goes from j to i and the third constraint imposes $u_i \leq u_j$. Suitable values for the "big M" parameters are $M_1 = u_i^{\text{UB}} - u_j^{\text{LB}}$ and $M_2 = u_j^{\text{UB}} - u_i^{\text{LB}}$.

[5]We assume throughout that the temperature is constant. This is a standard assumption, given that gas networks are typically underground and soil has very good insulating properties.

[6]We are implicitly assuming that the gas flowing through the network is homogeneous and, therefore, p_c and θ_c are known parameters. Also, it is worth mentioning that GANESO™ software can compute the compressibility factor via the more sophisticated SGERG88 model, as described in ISO-12213-3 (2006).

[7]GANESO™ software has also been tested with the formulation based on Colebrook formulation.

Compressors allow to increase the pressure of the gas in the direction of the flow. This is implied by the following constraint, which assumes that compressors are *unidirectional*:

$$u_i \leq u_j, \quad \forall k = (i, j) \in E^{\mathrm{c}}. \tag{UC}$$

Thus, compressor $k = (i, j)$ can compress gas only when it goes from i to j. Note that, when $q_k < 0$ the above constraint would also allow a compressor to "decompress" from j to i, *i.e.*, the compressor would be acting as a valve. This is not an issue, since compressors are normally part of more complex structures containing control valves which could also do that job. Alternatively, one could use binary variables to control that compressors can only compress (with a modeling similar to the one presented for valves), but this would increase the complexity of the resulting optimization model (see González-Rueda (2017)). In Section 2.2 we discuss why the constraints given by Equation (UC) are suitable given the modeling approach chosen for compressor stations.

Gas consumption associated to a compressor station is given by the following constraint:

$$g_k = \frac{Z(\sqrt{u_i}, \theta)}{LCV} \frac{R\theta}{\eta_k} \frac{\gamma}{\gamma - 1} \Big((\frac{u_j}{u_i})^{\frac{\gamma-1}{2\gamma}} - 1 \Big) |q_k|, \quad \forall k = (i, j) \in E^{\mathrm{c}}, \tag{GC}$$

where LCV [J/kg] denotes the lower calorific value of the gas, η_k [dimensionless] represents the efficiency of the compressor, and γ [dimensionless] the ratio of specific heats (at constant volume and constant pressure).

We present now some final comments regarding compressors:

- The gas consumed by the compressors is taken from the gas flowing through the network, so a completely rigorous model should take this into account in the corresponding flow conservation constraints. However, since this consumption is usually quite small it is customary to work under the simplified model in which this gas is assumed to remain in the network.[8]
- For a compressor $k = (i, j) \in E^{\mathrm{c}}$ with $q_k < 0$ and $u_i < u_j$, Equation (GC) would imply that a cost is paid for decompressing gas. However, as we discuss in Section 2.2, this won't be a problem for our approach.
- Actual compressors must operate within some operational ranges, which are briefly discussed in Section 2.2.

Objective function

The objective function consists in minimizing the gas consumption at compressors:

$$\min \sum_{k \in E^{\mathrm{c}}} g_k. \tag{OBJ}$$

2.2 Novel ingredients

We briefly outline now some novel elements and novel approaches to the modeling of gas transmission networks that are the result of the feedback we got from the collaboration with our partner in the gas industry and that have been incorporated to GANESO™ software. Again, the reader interested on a deeper analysis may refer to González-Rueda (2017).

[8] González-Rueda (2017) reports that actual consumption is around 0.4% of the gas flowing through the compressors, with similar figures having been reported in the literature.

Enlarging the feasible region

It turns out that, for the final user of a tool as GANESO™, it is crucial to get some useful information even for infeasible optimization problems. In particular, since most of the infeasibilities observed in practice could be solved by relaxing the pressure bounds, we implemented a variation of the model consisting in adding slack variables to pressure upper bound constraints (BC):

$$\begin{aligned} u_i^{\text{LB}} \leq u_i \leq u_i^{\text{UB}} + u_i^{\text{EXTRA}}, &\qquad \forall i \in N, \\ u_i^{\text{EXTRA}} \geq 0, &\qquad \forall i \in N, \\ q_k^{\text{LB}} \leq q_k \leq q_k^{\text{UB}}, &\qquad \forall k \in E, \end{aligned} \tag{BC$_{\text{EXTRA}}$}$$

where the extra slack variables u_i^{EXTRA} would be appropriately penalized in the objective function, obtaining

$$\min \sum_{k \in E^{\text{c}}} g_k + M^{\text{EXTRA}} \sum_{i \in N} u_i^{\text{EXTRA}}. \tag{OBJ$_{\text{EXTRA}}$}$$

Then, if $M > 0$ is sufficiently large, whenever the original problem is infeasible and the problem with constraints (BC$_{\text{EXTRA}}$) is not, those nodes for which $u_i^{\text{EXTRA}} > 0$ in a solution of the enlarged problem deliver useful information regarding the reason for the infeasibility of the original problem.[9]

Modeling of compressor stations

Compressors are a crucial element of the optimization of gas transmission networks and, as such, they do not appear in isolation, but as part of compressor stations. Figure 1 represents a typical compressor station of the Spanish gas network (taken from Enagás GTS (2010)). This compressor station appears in the connection of three pipes (from/to Montesa, Getafe, and Córdoba).

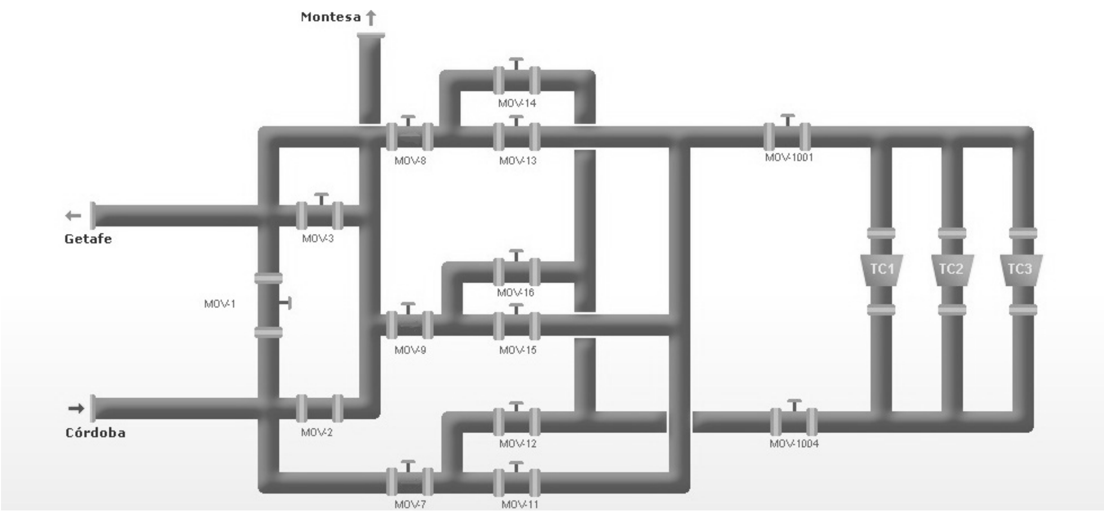
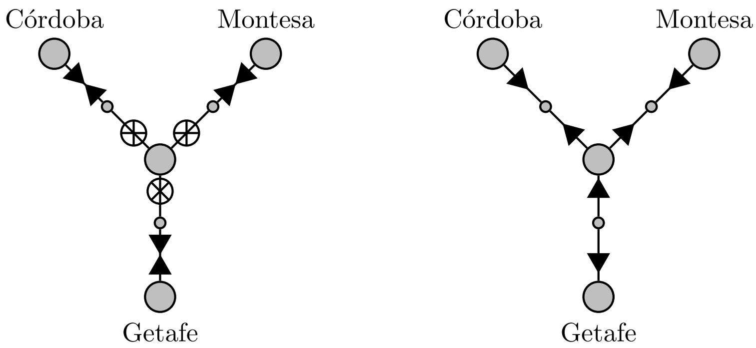


Figure 1: Hydraulic scheme of the compressor station in Alcázar de San Juan (Spain).

[9]Interestingly, as we briefly discuss in Section 5.1.1, when applying a sequential linear programming technique to solve the gas network optimization problem, the modeling with Equation (BC$_{\text{EXTRA}}$) is useful to sidestep potential infeasibility issues in the resolution of the linearized subproblems.

The elements TC1, TC2 and TC3 represent three unidirectional compressors working in parallel (the gas to be compressed must enter through the upper part of the compressors). The elements denoted by MOV-XXX represent control valves, allowing to configure the compressor station so that any compressing possibility is possible. For instance, one could get gas from Córdoba and send part of it to Getafe after compressing it, while the other part of the gas could go to Montesa with no compression. Another possibility could be to take gas from Getafe and Córdoba at different pressures, use the valves to bring them to the same pressure and then compress the resulting gas to send it to Montesa.

The downside of the richness allowed by the scheme of the figure is that it requires 14 binary variables for a single compressor station, one for each pressure control valve. This kind of modeling for the Spanish gas network would result in the inclusion of more than 200 binary variables in our MINLP problem just to model the compressor stations. In Figure 2 we present an alternative modeling, an approximation that is at the core of GANESO™ and that has led to good results in practice.



(a) Triplication. (b) Duplication.

Figure 2: Y-shaped modeling of a compressor station.

The elements of the Y-shaped modeling in Figure 2 are the following. Each triangle represents a unidirectional compressor, modeled as in Equation (UC); ▶ represents compression from left to right and ◀ compression from right to left. Each $\otimes$ represents a control valve, modeled as in Equation (CV). We refer to the representation in Figure 2(a) as *triplication*, since three edges are used to represent the connection of each pipe to the compressor station: two compressors and a valve. It turns out that the triplication modeling is quite flexible and allows essentially all possibilities that can be obtained for a compressor station like the one in Figure 1, but with only three control valves. Further, and we consider this a relevant feature of this novel Y-shaped modeling, is that these three valves can be removed altogether leading to a modeling that we call *duplication* (see Figure 2(b)). Interestingly, in terms of feasible solutions not much is lost since, as we mentioned while discussing unidirectional compressors, they can be used to "decompress", effectively acting as a control valve if needed. The only problem with this simplified modeling is that this decompression would come at a cost in the model while in reality would be for free.[10]

Operating costs at regasification plants: Boil-off costs

The tanks at a regasification plant are designed to stay cool, but they cannot provide perfect insulation against warming. Heat slowly affects the tanks, which can cause the gas inside to

[10]In GANESO™ software this effect is corrected at a later stage.

evaporate and produce the so-called boil-off gas. This increases the pressure in the tanks, and must be cooled down again, burned, or injected into the network. This process entails non-negligible costs, which can also be included into the mathematical programming problem. To the best of our knowledge this is the first paper in which boil-off costs are explicitly incorporated into the modeling of the optimization of gas transmission networks.

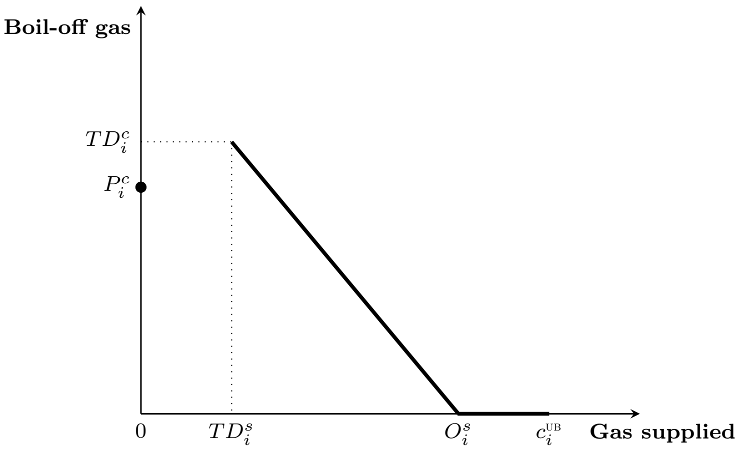


Figure 3: Boil-off consumption of a regasification plant.

Let N^{RP} be the set of nodes representing regasification plants. Then, for each $i \in N^{\text{RP}}$, the following elements are needed to model the boil-off costs (see Figure 3):

- P_i^c, the boil-off gas consumption of the plant when it is not supplying gas to the network.
- $TD_i = (TD_i^s, TD_i^c)$, the turn down point of the plant, which provides the minimum amount of gas it can supply, TD_i^s, and the boil-off consumption at this regime, TD_i^c.
- O_i^s, the supply from which the boil-off consumption of the plant is negligible.
- D_i^c: it is the discharge consumption by unit of gas supplied from the plant.

Figure 3 represents the boil-off consumption as a function of the gas that the plant is supplying, which in our model is represented, for each $i \in N^{\text{RP}}$, by $Q_i = \sum_{k \in E_i^{\text{INI}}} q_k - \sum_{k \in E_i^{\text{FIN}}} q_k$. Then, the cost function associated to regasification plant i, $B_i(\cdot)$, is given by

$$B_i(Q) = \begin{cases} P_i^c & \text{if} \quad Q_i = 0 \\ TD_i^c - \frac{TD_i^c}{O_i^s - TD_i^s}(Q_i - TD_i^s) + D_i^c Q_i & \text{if} \quad TD_i^s \leq Q_i \leq O_i^s \\ D_i^c Q_i & \text{if} \quad Q_i \geq O_i^s. \end{cases}$$

The inclusion of this function in our mathematical programming model requires to use, for each $i \in N^{\text{RP}}$, a binary variable y_i^{BO} that represents whether or not the plant is supplying gas to the network:

$$\begin{array}{rcl} Q_i & \geq & TD_i^s y_i^{\text{BO}}, \\ Q_i & \leq & c_i^{\text{UB}} y_i^{\text{BO}} \end{array} \tag{BO-1}$$

If $y_i^{\text{BO}} = 1$ the plant is injecting gas into the network and $TD_i^s \leq Q_i \leq c_i^{\text{UB}}$. If $y_i^{\text{BO}} = 0$ then there is no gas supplied by the plant, $Q_i = 0$. Now, function $B_i(\cdot)$ is represented with a variable b_i, by means of the following constraints, which include an auxiliary variable r_i:

$$\begin{array}{rcl} r_i & \geq & TD_i^c - \frac{TD_i^c}{O_i^s - TD_i^s}(Q_i - TD_i^s), \\ r_i & \geq & 0, \\ b_i & = & (1-y_i^{\text{BO}})P_i^c + D_i^c Q_i + r_i - (1-y_i^{\text{BO}})(TD_i^c + \frac{TD_i^c}{O_i^s - TD_i^s} TD_i^s). \end{array} \tag{BO-2}$$

The modification of the original cost function to include the costs associated to the plants on top of the costs associated to the compressors results in

$$\min \sum_{k \in E^{\text{C}}} g_k + \alpha \sum_{i \in N^{\text{RP}}} b_i, \tag{OBJ$_{\text{BOIL-OFF}}$}$$

where parameter α can be used to adjust the relative weight of boil-off costs relative to those associated to compressors.

Simplified modeling of the compressibility factor

In practice, the compressibility factor in natural gas networks does not have a lot of variability. In particular, for high pressure networks where pressures vary between $30 \cdot 10^5$ and $90 \cdot 10^5$ pascals, the range for the compressibility factor is typically between 0.75 and 0.95. Because of this, a natural simplification of the model is to assume that the compressibility factor is constant (see, for instance, Martin et al. (2006), Koch et al. (2015), or De Wolfe and Smeers (2000)).

This approach is especially useful when applying a sequential programming algorithm to solve the optimization problem. The reason is that, at iteration t of the algorithm, one can replace the terms $Z(p_k, \theta)$ with the values associated to the current solution $\mathbf{p}^t$, computed using pressure p_k^t in Equation (ZF). Then, if the algorithm converges, these approximations also converge to their true values.[11]

Additional elements

We now briefly mention some additional elements included in the full model that is routinely solved by the users in Reganosa Company but which, for the sake of exposition, have not been included in the batteries of test problems discussed in Section 5. The interested reader may refer to González-Rueda (2017) for more detailed descriptions.

- A completely rigorous model of a compressor station requires to include the operational ranges associated to each compressor station. For instance, in the case of the compressor station represented in Figure 1, one should ensure that the rate of compression given for the flow entering the station can be achieved using three compressors working in parallel. This requires to know the technical specifications of each compressor and, when translated to the optimization problem, requires the use of additional binary variables.
- One can also model additional types of valves such as i) open/closed valves, which can fully close the access of the gas through them and ii) regulation valves, which can be used to automatically bring the pressure down to a pre-specified value before entering a given part of the network.
- Additional cost drivers can be included into the model such as i) maximize/minimize the total amount of gas present in the pipes, known as line-pack and ii) maximize imports/exports from/to a given zone of the network.

[11] This iterative approach has also been implemented in GANESO$^{\text{TM}}$ to model the friction factor via Colebrook equation, taking it as a constant for the linearized subproblems, but updating it after every iteration.

2.3 Resulting optimization models and state of the art

The elements discussed so far in this section allow to define different optimization problems over gas transmission networks, depending not only on the relevant cost drivers on which one may desire to focus, but also on the desired trade-off between the precision on the modeling of different elements and the difficulty of the resulting problem. To give a rough idea about the typical complexity of these problems, in the particular case of the Spanish gas transmission network, the resulting model has around 1000 variables and 1000 constraints, half of which are nonlinear (and nonconvex). The number of binary variables can vary between 10 and 200, depending on the number of valves considered, and the inclusion or not of elements such as boil-off costs and triplications or operational ranges at compressor stations.

Interestingly, setting aside the binary variables, we have a problem where the number of variables is similar to the number of constraints and a large number of these constraints are equality constraints. This structure is similar to that of the problems in the oil industry where sequential linear programming algorithms appeared for the first time starting with Griffith and Stewart (1961) as a research project for Shell Development Company and with significant activity in the seventies (Ali et al., 1978; Beale, 1978; Boddington and Randall, 1979). One of the first papers to develop a mathematical analysis of SLP techniques is Palacios-Gomez et al. (1982), where the authors already identify that "It appears that SLP will be most successful when applied to large problems with low degrees of freedom". To the best of our knowledge this assessment is still valid nowadays, and this is one of the reasons why we have chosen to apply these techniques to the optimization of gas transmission networks.

Before delving into the details of our approach, we present a brief overview of the state of the art in the field. A recent survey of the different solution procedures to deal with this problem can be found in Ríos-Mercado and Borraz-Sánchez (2015). A more general introduction to the field of gas transmission networks can be found in the book Koch et al. (2015).

Part of the past literature has focused on NLP formulations of the problem, neglecting the combinatorial aspects of the transmission networks (mainly valves and compressors). Dynamic programming has been one of the first techniques to be applied to the optimization of gas networks, although restricted to networks without cycles (Wong and Larson, 1968) and, more recently, these techniques were applied to cyclic networks (Carter, 1998; Ríos-Mercado et al., 2006). Sequential linear/quadratic programming ideas have also been seen in practice (Ehrhardt and Steinbach, 2002). A different approach is taken in De Wolfe and Smeers (2000), where the nonlinear pressure loss constraints are replaced by piecewise linear approximations and the resulting problem is solved by an extension of the simplex method.

When the gas network is modeled using MINLP formulations, it is common to use two-step procedures that fix the behavior of the combinatorial elements by solving a simplified MINLP problem in the first step and then solve the resulting NLP problem in the second one (see, for instance, Pfetsch et al. (2015). Two exceptions are Cobos-Zaleta and Ríos-Mercado (2002) and Martin et al. (2006). In Cobos-Zaleta and Ríos-Mercado (2002) the solution technique is based on an outer approximations combined with an augmented penalty algorithm. In Martin et al. (2006), the authors present some novel techniques to obtain piecewise linear approximations of the nonlinearities of the model, resulting in large MILP problems. This approach has the advantage that if the piecewise linear approximations are accurate enough, globally optimal solutions of the approximated problem are guaranteed. On the down side, the high number of binary variables needed to get accurate approximations limits the size of the problems that can be tackled with this approach.

In this paper we propose a 2-step SLP algorithm, whose first step is a natural modification of the classic SLP approaches, which we call SLP-NTR, that handles the combinatorial features of the problem (no simplification is made on the optimization problem in this step). Then, once the combinatorial elements are fixed, we run a more classic SLP algorithm that has better convergence properties than the SLP-NTR used in the first step. This approach is relatively close to the one used in Pratt and Wilson (1984), where the authors propose a successive mixed-integer linear programming method in which, at each iteration, the pressure loss constraints are linearized and a MILP is solved. As the algorithm progresses, the values of those binary variables that have not changed for a given number of iterations are fixed. Pratt and Wilson (1984) is mainly an applied contribution, containing no theoretical results. Differently, one of the main contributions of this paper, included in the following section, is to study to what extent the SLP-NTR preserves the theoretical properties of the classic SLP algorithms for general NLP problems (and not only for gas network problems). Furthermore, the 2-step SLP described above can be applied to general MINLP problems as a heuristic.[12] The performance of the SLP and 2-step SLP algorithms for the optimization of gas transmission networks is studied in Section 5.

3 SLP without trust region for NLP and MINLP problems

We move now to the main methodological and theoretical contribution of this paper: a modification of the classic sequential linear programming algorithms. More precisely, this modification works with *No Trust Region* and, hence, the name SLP-NTR. The SLP-NTR is an algorithm for general NLP problems which, moreover, will be the basis for the 2-step SLP algorithm, an algorithm that can also be applied as a heuristic for MINLP problems.

3.1 Preliminaries and notations

Consider the standard formulation of a nonlinear programming problem NLP:

$$\begin{array}{llll} \text{(NLP)} & \text{Minimize} & f(\boldsymbol{x}) & \\ & \text{subject to} & g_i(\boldsymbol{x}) \leq 0 & \forall i \in I^m = \{1,\ldots,m\} \\ & & h_j(\boldsymbol{x}) = 0 & \forall j \in J^l = \{m+1,\ldots,m+l\} \\ & & \boldsymbol{x} \in X = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\}, & \end{array}$$

where X denotes the set of linear constraints and the functions $f : \mathbb{R}^n \to \mathbb{R}$, $g_i : \mathbb{R}^n \to \mathbb{R}$ for $i \in I^m$ and $h_j : \mathbb{R}^n \to \mathbb{R}$ for $j \in J^l$ are assumed to be continuously differentiable. The next definition associates, to each NLP problem, a linearization of it that is at the core of any sequential linear programming algorithm.

Definition 1. The *first-order Taylor linear programming approximation* of the NLP problem around $\bar{\boldsymbol{x}} \in \mathbb{R}^n$ with step $\boldsymbol{s} \in \mathbb{R}^n$, TNLP$(\bar{\boldsymbol{x}}, \boldsymbol{s})$, is the following linear programming problem:

$$\begin{array}{llll} (\text{TNLP}(\bar{\boldsymbol{x}}, \boldsymbol{s})) & \text{Minimize} & f(\bar{\boldsymbol{x}}) + \nabla f(\bar{\boldsymbol{x}})^{\intercal}(\boldsymbol{x} - \bar{\boldsymbol{x}}) & \\ & \text{subject to} & g_i(\bar{\boldsymbol{x}}) + \nabla g_i(\bar{\boldsymbol{x}})^{\intercal}(\boldsymbol{x} - \bar{\boldsymbol{x}}) \leq 0 & \forall i \in I^m \\ & & h_j(\bar{\boldsymbol{x}}) + \nabla h_j(\bar{\boldsymbol{x}})^{\intercal}(\boldsymbol{x} - \bar{\boldsymbol{x}}) = 0 & \forall j \in J^l \\ & & \boldsymbol{x} \in X = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\} & \\ & & -\boldsymbol{s} \leq \boldsymbol{x} - \bar{\boldsymbol{x}} \leq \boldsymbol{s} & \text{(trust region).} \end{array}$$

If we let $\boldsymbol{x} = \bar{\boldsymbol{x}} + \boldsymbol{d}$ and think of TNLP$(\bar{\boldsymbol{x}}, \boldsymbol{s})$ in terms of the variables $\boldsymbol{d} = \boldsymbol{x} - \bar{\boldsymbol{x}}$, then this problem can be seen as a direction-finding problem. This linear problem yields an optimal solution $\boldsymbol{d}$, which denotes the optimal descent direction from $\bar{\boldsymbol{x}}$. How far to proceed along this direction is restricted by the trust region ($-\boldsymbol{s} \leq \bar{\boldsymbol{x}} + \boldsymbol{d} \leq \boldsymbol{s}$). We slightly abuse notation and use

[12] Given our experience, we conjecture that this 2-step SLP approach is especially suitable for “large problems with low degrees of freedom” where the proportion of binary variables is relatively small.

$\text{TNLP}(\bar{\boldsymbol{x}}) = \text{TNLP}(\bar{\boldsymbol{x}}, \infty)$ to denote the linear approximation of NLP in which there is no trust region.

A SLP algorithm works as follows: given an initial point $\bar{\boldsymbol{x}}$ (not necessary feasible for NLP) and a step bound $\boldsymbol{s}$, it solves the linearized problem $\text{TNLP}(\bar{\boldsymbol{x}}, \boldsymbol{s})$. If the solution of the linearized problem satisfies some criteria (a trade-off between feasibility and improvement of the objective function), the solution is taken as the basis of a new linearization, the step bounds are (possibly) increased and the procedure is repeated. Otherwise, the step bounds are reduced to $\boldsymbol{s}'$ and the problem $\text{TNLP}(\bar{\boldsymbol{x}}, \boldsymbol{s}')$ is solved. The main mathematical property of these algorithms (see (Bazaraa et al., 2006, Chapter 10.2)) is that if a point $\bar{\boldsymbol{x}}$ solves $\text{TNLP}(\bar{\boldsymbol{x}}, \boldsymbol{s})$, then $\bar{\boldsymbol{x}}$ is a KKT point for NLP, so typical stopping criteria are based on $\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|$ being sufficient small.

One of the main limitations of the SLP algorithms as we have described them is that, for nonconvex problems, the linear problems $\text{TNLP}(\bar{\boldsymbol{x}}, \boldsymbol{s}')$ may be infeasible, even if the original NLP problem is not. When the NLP under consideration may suffer from this problem, it is natural to work with penalized versions of the linearized problems, which ensure feasibility after every iteration. Next, we present one algorithm implementing these ideas and that will be the benchmark for both the theoretical and numerical analysis developed in this paper.

3.2 Penalty Sequential Linear Programming

We present now the Penalty Sequential Linear Programming algorithm (PSLP), introduced in Kim et al. (1985) and tested with real nonlinear problems at Exxon Company in Baker and Lasdon (1985). We need some additional preliminaries before presenting the scheme of the PSLP algorithm.

Definition 2. Given problem NLP, the penalized problem $\text{PNP}^{\boldsymbol{\mu}}$ with penalty parameter $\boldsymbol{\mu} \in \mathbb{R}^{m+l}$ is defined as:

$$(\text{PNP}^{\boldsymbol{\mu}}) \quad \begin{array}{ll} \text{Minimize} & p(\boldsymbol{x}) = f(\boldsymbol{x}) + \displaystyle\sum_{i \in I^m} \mu_i \max\{g_i(\boldsymbol{x}), 0\} + \sum_{j \in J^l} \mu_j |h_j(\boldsymbol{x})| \\ \text{subject to} & \boldsymbol{x} \in X = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\}. \end{array}$$

The following result, whose proof can be seen in (Bazaraa et al., 2006, Theorem 10.3.1), establishes the relationship between the original NLP problem and problem $\text{PNP}^{\boldsymbol{\mu}}$.

Theorem 1. *Let* NLP *be a nonlinear problem and let* $\text{PNP}^{\boldsymbol{\mu}}$ *be the associated penalized problem with penalty parameter* $\boldsymbol{\mu}$. *The following statements hold true:*

i) If $\bar{\boldsymbol{x}}$ is a KKT point for problem NLP *with Lagrange multipliers (u_i, v_j) for all $i \in I^m$, $j \in J^l$ such that $\mu_i > |u_i|$ for all $i \in I^m$ and $\mu_j > |v_j|$ for all $j \in J^l$, then $\bar{\boldsymbol{x}}$ is a KKT point for problem* $\text{PNP}^{\boldsymbol{\mu}}$.

ii) If $\bar{\boldsymbol{x}}$ is a KKT point for problem $\text{PNP}^{\boldsymbol{\mu}}$ *and $\bar{\boldsymbol{x}}$ is feasible for* NLP, *then $\bar{\boldsymbol{x}}$ is a KKT point for* NLP.

PSLP follows the idea of SLP algorithms but using linearizations of $\text{PNP}^{\boldsymbol{\mu}}$ instead of linearizations of NLP. We now define the first-order Taylor linear approximation of the objective function of $\text{PNP}^{\boldsymbol{\mu}}$ around a point $\bar{\boldsymbol{x}}$:

$$\begin{aligned} p_L(\boldsymbol{x}) = f(\bar{\boldsymbol{x}}) + \nabla f(\bar{\boldsymbol{x}})^\intercal(\boldsymbol{x} - \bar{\boldsymbol{x}}) \quad &+ \quad \sum_{i \in I^m} \mu_i \max\{g_i(\bar{\boldsymbol{x}}) + \nabla g_i(\bar{\boldsymbol{x}})^\intercal(\boldsymbol{x} - \bar{\boldsymbol{x}}), 0\} \\ &+ \quad \sum_{j \in J^l} \mu_j |h_j(\bar{\boldsymbol{x}}) + \nabla h_j(\bar{\boldsymbol{x}})^\intercal(\boldsymbol{x} - \bar{\boldsymbol{x}})|. \end{aligned}$$

Definition 3. Given the problem PNP$^{\boldsymbol{\mu}}$, its first-order Taylor linear approximation around $\overline{\boldsymbol{x}} \in \mathbb{R}^n$ with step $\boldsymbol{s}$ is defined as the problem TPNP$^{\boldsymbol{\mu}}(\overline{\boldsymbol{x}}, \boldsymbol{s})$, given by

$$\begin{array}{lll} (\text{TPNP}^{\boldsymbol{\mu}}(\overline{\boldsymbol{x}}, \boldsymbol{s})) & \text{Minimize} & p_L(\boldsymbol{x}) \\ & \text{subject to} & \boldsymbol{x} \in X = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\} \\ & & -\boldsymbol{s} \leq \boldsymbol{x} - \overline{\boldsymbol{x}} \leq \boldsymbol{s}. \end{array}$$

We can equivalently rewrite TPNP$^{\boldsymbol{\mu}}(\overline{\boldsymbol{x}}, \boldsymbol{s})$ without the non-differentiable terms (absolute value and maximum function) as follows:

$$\begin{array}{llll} (\text{TPNP}^{\boldsymbol{\mu}}(\overline{\boldsymbol{x}}, \boldsymbol{s})) & \text{Minimize} & p_L(\boldsymbol{x}) = f(\overline{\boldsymbol{x}}) + \nabla f(\overline{\boldsymbol{x}})^{\intercal}(\boldsymbol{x} - \overline{\boldsymbol{x}}) + \sum_{i \in I^m} \mu_i y_i + \sum_{j \in J^l} \mu_j (p_j^+ + p_j^-) & \\ & \text{subject to} & g_i(\overline{\boldsymbol{x}}) + \nabla g_i(\overline{\boldsymbol{x}})^{\intercal}(\boldsymbol{x} - \overline{\boldsymbol{x}}) \leq y_i & \forall i \in I^m \\ & & h_j(\overline{\boldsymbol{x}}) + \nabla h_j(\overline{\boldsymbol{x}})^{\intercal}(\boldsymbol{x} - \overline{\boldsymbol{x}}) = p_j^+ - p_j^- & \forall j \in J^l \\ & & \boldsymbol{x} \in X = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\} & \\ & & -\boldsymbol{s} \leq \boldsymbol{x} - \overline{\boldsymbol{x}} \leq \boldsymbol{s} & \\ & & y_i, p_j^+, p_j^- \geq 0 & \forall i \in I^m, \forall j \in J^l. \end{array}$$

Under this reformulation, TPNP$^{\boldsymbol{\mu}}(\overline{\boldsymbol{x}}, \boldsymbol{s})$ is a linear problem. Finally, using the substitution $\boldsymbol{x} = \overline{\boldsymbol{x}} + \boldsymbol{d}$ and removing the constant term of the objective function, TPNP$^{\boldsymbol{\mu}}(\overline{\boldsymbol{x}}, \boldsymbol{s})$ can be equivalently rewritten as:

$$\begin{array}{llll} (\text{TPNP}^{\boldsymbol{\mu}}(\overline{\boldsymbol{x}}, \boldsymbol{s})) & \text{Minimize} & p_L(\boldsymbol{x}) = \nabla f(\overline{\boldsymbol{x}})^{\intercal}\boldsymbol{d} + \sum_{i \in I^m} \mu_i y_i + \sum_{j \in J^l} \mu_j (p_j^+ + p_j^-) & \\ & \text{subject to} & g_i(\overline{\boldsymbol{x}}) + \nabla g_i(\overline{\boldsymbol{x}})^{\intercal}\boldsymbol{d} \leq y_i & \forall i \in I^m \\ & & h_j(\overline{\boldsymbol{x}}) + \nabla h_j(\overline{\boldsymbol{x}})^{\intercal}\boldsymbol{d} = p_j^+ - p_j^- & \forall j \in J^l \\ & & \boldsymbol{A}(\overline{\boldsymbol{x}} + \boldsymbol{d}) \leq \boldsymbol{b} & \\ & & -\boldsymbol{s} \leq \boldsymbol{d} \leq \boldsymbol{s} & \\ & & y_i, p_j^+, p_j^- \geq 0 & \forall i \in I^m, \forall j \in J^l. \end{array}$$

This linear problem is the *direction-finding subproblem.*

The PSLP works as follows. Given a current iterate $\boldsymbol{x}^k \in X$ and a trust region $\boldsymbol{s}^k$, it solves the direction-finding subproblem TPNP$^{\boldsymbol{\mu}}(\boldsymbol{x}^k, \boldsymbol{s}^k)$ obtaining a solution $\boldsymbol{d}^k$. Then, the decision whether to accept or reject the new iterate $\boldsymbol{x}^k + \boldsymbol{d}^k$ and the modifications of the step bounds $\boldsymbol{s}^k$ are made based on the ratio r^k of the actual decrease Δp^k of the objective function of the problem PNP$^{\boldsymbol{\mu}}$, and the decrease Δp_L^k of the linearized objective function used in TPNP$^{\boldsymbol{\mu}}$, provided that the latter is non-zero.[13] These quantities are computed as follows:

$$\Delta p^k = p(\boldsymbol{x}^k) - p(\boldsymbol{x}^k + \boldsymbol{d}^k) \text{ and } \Delta p_L^k = p_L(\boldsymbol{x}^k) - p_L(\boldsymbol{x}^k + \boldsymbol{d}^k).$$

Suppose that, at some iteration of the algorithm, we get that $\boldsymbol{d}^k = \boldsymbol{0}$. Then, the optimality conditions for TPNP$^{\boldsymbol{\mu}}$ coincide with those for PNP$^{\boldsymbol{\mu}}$, obtaining that $\boldsymbol{x}^k$ is a KKT point for PNP$^{\boldsymbol{\mu}}$. Thus, if $\boldsymbol{x}^k$ is feasible for NLP, by item ii) of Theorem 1, $\boldsymbol{x}^k$ is a KKT point for NLP. On the other hand, if $\boldsymbol{x}^k$ is not feasible for NLP, the penalty parameters $\boldsymbol{\mu}$ may need to be increased. The full description of PSLP is illustrated in Algorithm 1.

[13]The idea of the update of the trust region in Algorithm 1 is as follows. If the ratio r^k is negative or close to zero, the penalty function has either worsened or its improvement is very small. In this case, the algorithm rejects the current solution and reduces the step bound. "Kim et al. (1985)" showed that within a finite number of such reductions, a positive value of r^k will be obtained. Otherwise, the current solution is accepted and, depending on the improvement of the penalty function the step bound may be reduced if the improvement is below a threshold ρ_1 or increased if it is above another threshold ρ_2.

Algorithm 1 Penalty Sequential Linear Programming (PSLP)

1: Initialize $\boldsymbol{x}^0 \in \mathbb{R}^n$ satisfying the bound constraints of the problem, and $\boldsymbol{s}^0 \in \mathbb{R}^n$. Fix the parameters $\boldsymbol{\mu}$, $0 < \rho_0 < \rho_1 < \rho_2 < 1$, $0 < \beta < 1$, $\alpha > 0$ and $\varepsilon > 0$. Let $k = 0$.
2: Solve TPNP$^{\boldsymbol{\mu}}(\boldsymbol{x}^k, \boldsymbol{s}^k)$. Let $\bar{\boldsymbol{x}}^k$ be an optimal solution of TPNP$^{\boldsymbol{\mu}}(\boldsymbol{x}^k, \boldsymbol{s}^k)$.
3: Compute $\Delta p = p(\boldsymbol{x}^k) - p(\bar{\boldsymbol{x}}^k)$ and $\Delta p_L = p_L(\boldsymbol{x}^k) - p_L(\bar{\boldsymbol{x}}^k)$.
4: **if** $\Delta p_L < \varepsilon$ **then**
5: STOP: Return $\bar{\boldsymbol{x}}^k$.
6: **else**
7: Compute $r^k = \frac{\Delta p}{\Delta p_L}$.
8: **end if**
9: **if** $r^k < \rho_0$ **then**
10: $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k$ and $\boldsymbol{s}^{k+1} = \beta \boldsymbol{s}_k$.
11: **else**
12: $\boldsymbol{x}^{k+1} = \bar{\boldsymbol{x}}^k$ and update $s_{k+1} = \begin{cases} \beta \boldsymbol{s}^k & \text{if } r^k < \rho_1 \\ \dfrac{\boldsymbol{s}^k}{\beta} & \text{if } r^k > \rho_2 \\ \boldsymbol{s}^k & \text{otherwise.} \end{cases}$
13: **end if**
14: Take $s_i^{k+1} = \max\{s_i^{k+1}, \alpha\}$ for all $i \in \{1, \ldots, n\}$, let $k = k + 1 \rightarrow$ go to 2.

Regarding the convergence of PSLP, it cannot be ensured that the sequence $\{\boldsymbol{x}^l\}$ converges. However, the following theorem, whose proof can be seen in Kim et al. (1985), ensures that any convergent subsequence converges to a KKT point for problem PNP$^{\boldsymbol{\mu}}$ which, if feasible for NLP, is also a KKT point for NLP.

Theorem 2. *Let $\{\boldsymbol{x}^l\}$ be the sequence generated by the* PSLP *algorithm applied to the problem* PNP$^{\boldsymbol{\mu}}$. *If the set $C = \{\boldsymbol{x} \in X : p(\boldsymbol{x}) \leq p(\boldsymbol{x}^1)\}$ is bounded, then every accumulation point of $\{\boldsymbol{x}^l\}$ is a KKT point for* PNP$^{\boldsymbol{\mu}}$. *Furthermore, if the accumulation point is also feasible for* NLP, *then it is a KKT point for* NLP.

3.3 Sequential linear programming with no trust region

We now define a natural modification of the SLP ideas that we call *Sequential Linear Programming with No Trust Region*: SLP-NTR. The modification consists in removing the trust region, so there is no control about how far from each other are two consecutive elements of the generated sequence. More precisely, given a solution $\boldsymbol{x}^k$, problem TNLP$(\boldsymbol{x}^k)$ is solved, obtaining $\boldsymbol{x}^{k+1}$. The algorithm finishes when $\|\boldsymbol{x}^k - \boldsymbol{x}^{k+1}\|$ is sufficiently small. The scheme of SLP-NTR is represented in Algorithm 2.

Algorithm 2 Sequential Linear Programming with No Trust Region (SLP-NTR)

1: Initialize $\boldsymbol{x}^0 \in \mathbb{R}^n$. Fix $\varepsilon > 0$. Let $k = 0$.
2: Solve TNLP$(\boldsymbol{x}^k)$. Let $\boldsymbol{x}^{k+1}$ be a solution of TNLP$(\boldsymbol{x}^k)$.
3: **if** $\|\boldsymbol{x}^k - \boldsymbol{x}^{k+1}\| < \varepsilon$ **then**
4: STOP: Return $\boldsymbol{x}^{k+1}$.
5: **else**
6: Let $k = k + 1 \rightarrow$ go to 2.
7: **end if**

Note that a limitation of SLP-NTR is that, for a given NLP problem, some of the TNLP$(\boldsymbol{x}^k)$

subproblems may be infeasible. When working with problems in which this may be a concern, one can apply the penalized version of the algorithm: PSLP-NTR, described in Algorithm 3. The idea is to work with the linearizations of problem PNP$^{\boldsymbol{\mu}}$, solving subproblems TPNP$^{\boldsymbol{\mu}}(\boldsymbol{x}^k)$, whose feasibility is guaranteed.

Algorithm 3 Penalty Sequential Linear Programming with No Trust Region (PSLP-NTR)

1: Initialize $\boldsymbol{x}^0 \in \mathbb{R}^n$. Fix $\varepsilon > 0$ and $\boldsymbol{\mu} \in \mathbb{R}^{m+l}$. Let $k = 0$.
2: Solve TPNP$^{\boldsymbol{\mu}}(\boldsymbol{x}^k)$. Let $\boldsymbol{x}^{k+1}$ be a solution of TPNP$^{\boldsymbol{\mu}}(\boldsymbol{x}^k)$.
3: **if** $\|\boldsymbol{x}^k - \boldsymbol{x}^{k+1}\| < \varepsilon$ **then**
4: STOP: Return $\boldsymbol{x}^{k+1}$.
5: **else**
6: Let $k = k + 1 \rightarrow$ go to 2.
7: **end if**

3.3.1 Strengths of SLP-NTR and PSLP-NTR algorithms

- The implementation of these algorithms is straightforward and, differently from classic PSLP, there are no parameters to be tuned. The only exception is the penalization $\boldsymbol{\mu}$ in PSLP-NTR, which has to be chosen large enough to ensure that the algorithm returns feasible solutions of NLP.
- One of the main contributions of this paper comes from the theoretical results obtained for SLP-NTR and PSLP-NTR algorithms, developed in Section 4, and that can be summarized as follows:
  - If the sequence $\{\boldsymbol{x}^l\}$ generated by SLP-NTR converges to a point $\bar{\boldsymbol{x}}$, then $\bar{\boldsymbol{x}}$ is a KKT point for NLP (Theorem 4).
  - If the sequence $\{\boldsymbol{x}^l\}$ generated by PSLP-NTR converges to a point $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{x}}$ is feasible for NLP, then $\bar{\boldsymbol{x}}$ is a KKT point for NLP (Proposition 5).
  - If two consecutive points $\bar{\boldsymbol{x}}^{k-1}$ and $\bar{\boldsymbol{x}}^k$ in the sequence $\{\boldsymbol{x}^l\}$ generated by SLP-NTR are sufficiently close, then $\bar{\boldsymbol{x}}^k$ is an *almost-KKT point* of NLP (Theorem 9).
  - If two consecutive points $\bar{\boldsymbol{x}}^{k-1}$ and $\bar{\boldsymbol{x}}^k$ in the sequence $\{\boldsymbol{x}^l\}$ generated by PSLP-NTR are sufficiently close and $\bar{\boldsymbol{x}}^k$ is *almost feasible*, then $\bar{\boldsymbol{x}}^k$ is an *almost*-KKT point for NLP (Proposition 10).
- One advantage of SLP-NTR and PSLP-NTR algorithms with respect to classic SLP algorithms is that they can be readily applied, as a heuristic, to MINLP problems. Actually, this is the main reason why we have chosen this approach to tackle optimization problems in gas transmission networks.[14]
- The above heuristic is applied as a 2-step SLP procedure, formally defined in Section 3.4. In the first step either SLP-NTR or PSLP-NTR are used to fix the binary variables and then the classic PSLP is run on the resulting NLP problem.

[14]Adapting a classic PSLP algorithm to deal with integer variables is not straightforward, since it is hard to reconcile the trust region philosophy of the algorithm with the discrete nature of the integer variables. A 0-1 change in a binary variable may have a big impact on the feasible region, so to get feasible solutions one may need a large trust region for the continuous variables. For a recent exception where a trust region approach is used in a convex MINLP problem refer to Kronqvist et al. (2018).

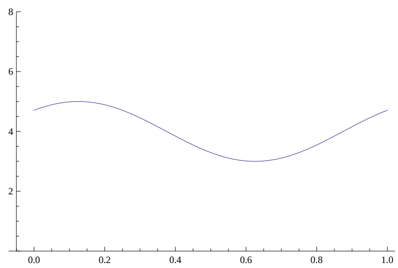
- An important part of the role of SLP-NTR (or PSLP-NTR) in the first step of the 2-step SLP is to provide a good starting point to PSLP, which has better convergence properties.

- Remarkably, as we discuss in Section 5, 2-step SLP exhibits a very good performance on NLP and MINLP problems associated to gas transmission networks.

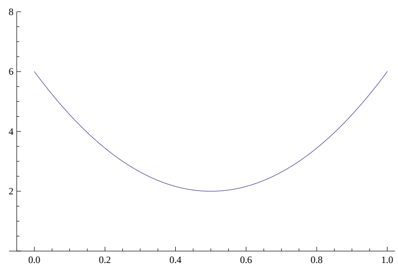
3.3.2 Weaknesses of SLP-NTR and PSLP-NTR algorithms

SLP-NTR and PSLP-NTR algorithms are just simplified versions of the classic SLP and PSLP algorithms, and the introduction of the trust region probably came to overcome some limitations of the "raw" versions, such as the ones we discuss below:

- SLP-NTR and PSLP-NTR algorithms rarely converge to interior points so, in particular, they will rarely converge in problems with no KKT points in the boundary. The main reason is that, since there is no trust region, the solution at every iteration will be an extreme point of the linearization of the feasible region around the current point.[15] To illustrate, suppose that we want to minimize the function in Figure 4(a) over the $[0, 1]$ interval. The global minimum is at 0.625, with $f(0.625) = 3$. Yet, the SLP-NTR applied to this function converges to one of the local minima on the boundary, being $f(0) = f(1) = 4.7071$. Arguably, the situation is even worse to minimize the function in Figure 4(b) over the $[0, 1]$ interval. The global minimum is at 0.5, with $f(0.5) = 2$. In this case, the SLP-NTR does not converge, oscillating between 0 and 1.



(a) $f(x) = 4 + sin(2\pi x + \pi/4)$.



(b) $f(x) = (4x - 2)^2 + 2$.

Figure 4: Weaknesses of SLP-NTR and PSLP algorithms.

- The removal of the trust region results in algorithms that are less stable in terms of convergence, as illustrated with the problem in Figure 4(b). This suggests that some form of stabilization (trust region, regularization, or line search) is needed to improve the convergence.[16]

- Although convergence is not guaranteed either for classic SLP algorithms, theoretical results normally ensure that any accumulation point of the sequence generated by the algorithm is a KKT point for NLP (see Theorem 2). This is not true for SLP-NTR and PSLP-NTR algorithms. Indeed, in the problem in Figure 4(b), the two accumulation

[15] Judging from the numerical results reported in Section 5, this severe limitation of SLP-NTR and PSLP-NTR algorithms does not seem critical in problems as the ones studied in this paper, with many equality constraints and, thus, feasible regions with low degrees of freedom and "small" interiors (and relative interiors).

[16] The reader interested in different stabilization methods may refer to the book (Bazaraa et al., 2006, Chapter 8-10) and references therein.

points of the sequence would be the two global maxima of the problem. This limitation of the algorithms is partially mitigated by the results obtained in Section 4, already outlined above.

Despite of the above weaknesses, since our proposal is to run a 2-step SLP algorithm, without trust region in the first step and with it in the second one, these theoretical limitations are not so problematic in practice.

3.4 A 2-step SLP algorithm for NLP and MINLP problems

Before moving to the theoretical analysis of the properties of SLP-NTR and PSLP-NTR algorithms for NLP problems, we formally present the 2-step SLP algorithm that, to some extent, combines the advantages of the algorithms with and without trust region. The use of an algorithm without trust region in the first step allows to heuristically tackle MINLP problems, finding candidate values for the integer variables, and the use of an algorithm with trust region in the second one allows to get better convergence properties for the resulting NLP problem.

Consider the standard formulation of a MINLP problem:

$$
\begin{array}{llll}
\text{(MINLP)} & \text{Minimize} & f(\boldsymbol{x}) & \\
& \text{subject to} & g_i(\boldsymbol{x}) \leq 0 & \forall i \in I^m = \{1, \ldots, m\} \\
& & h_j(\boldsymbol{x}) = 0 & \forall j \in J^l = \{m+1, \ldots, m+l\} \\
& & \boldsymbol{x} \in X = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\} & \\
& & x_i \in \mathbb{Z} & \forall i \in Z,
\end{array}
$$

where Z denotes the index set of integer variables. In the first step of the 2-step SLP algorithm one uses SLP-NTR or PSLP-NTR to solve the MINLP problem. The main goal in this first step is to find suitable values for the integer variables. The latter are taken as fixed in the second step, in which the classic PSLP algorithm is run.[17] The scheme of 2-step SLP is represented in Algorithm 4.

Algorithm 4 2-step SLP algorithm (2SLP)

Let $P^{\text{MINLP}}(\boldsymbol{x})$ be a MINLP problem.
Step 1:
  Initialize $\boldsymbol{x}^{\text{s1}}$.
  Apply SLP-NTR or PSLP-NTR to $P^{\text{MINLP}}(\boldsymbol{x})$ taking $\boldsymbol{x}^{\text{s1}}$ as initial solution.
  Let $\bar{\boldsymbol{x}}^{\text{s1}}$ be the solution.
Fixed the integer variables: $\boldsymbol{x}_i = \bar{\boldsymbol{x}}_i^{\text{s1}}$ for all $i \in Z$.
Let $P^{\text{NLP}}(\boldsymbol{x})$ be the resulting NLP problem.
Step 2:
  Initialize $\boldsymbol{x}^{\text{s2}} = \bar{\boldsymbol{x}}^{\text{s1}}$.
  Apply PSLP to $P^{\text{NLP}}(\boldsymbol{x})$ taking $\boldsymbol{x}^{\text{s2}}$ as initial solution.
  Let $\bar{\boldsymbol{x}}^{\text{s2}}$ be the solution.
Return $\bar{\boldsymbol{x}}^{\text{s2}}$.

We conclude with some final considerations regarding the 2-step SLP approach, based on our experience with it:

[17] It is worth noting that, when the algorithm run in the first step does not converge, one must identify the "best solution" and use it to fix the integer variables. In Section 5 we explain the approach we have followed for our computational tests.

- Since the algorithm is just based on SLP approaches it is easy to implement, relying only on general LP and MILP solvers.
- It does not require any simplification of the original problem in first step, which is a departure from what is usually done in practice (see, for instance, the two-stage approach in Pfetsch et al. (2015)).
- It is suitable for problems that have nearly as many constraints as variables and that have relatively small number of integer variables with respect to the total number of variables. This is the case of the optimization model for gas transmission networks presented in Section 2.
- Although in practice we often observe a poor convergence of the algorithm without trust region used in first step, this might even be an advantage, since it allows for large jumps in the search space, and can therefore explore the search space to find good starting points. Yet, it is important that this poor convergence is addressed in the second step when running the PSLP algorithm.
- Since the 2-step SLP algorithm is at the core of GANESO™ software, it has been widely tested on real instances of the gas industry (with over a thousand variables).

4 Theoretical analysis of SLP-NTR for NLP problems

This section is devoted to the study of the theoretical properties associated to the convergence of the SLP-NTR and the PSLP-NTR algorithms for general NLP problems . Recall that all the defining functions of NLP are assumed to be continuously differentiable. Hereafter we also assume that the set X is bounded, so the feasible region of problem NLP is contained in a compact set $K \subset \mathbb{R}^n$. Boundedness is a relatively weak assumption that holds in most real life applications. For the sake of notation, in this section we write TNLP($\boldsymbol{x}$) and TPNP$^{\boldsymbol{\mu}}$($\boldsymbol{x}$) instead of TNLP($\boldsymbol{x}, \infty$) and TPNP$^{\boldsymbol{\mu}}$($\boldsymbol{x}, \infty$).

Next, we introduce an auxiliary result useful to proof next theorem of this section related to the convergence properties of the SLP-NTR algorithm.

Lemma 3. *Given the problem* NLP *and* $\bar{\boldsymbol{x}} \in \mathbb{R}^n$*, suppose that functions* f*,* g_i *for all* $i \in I^m$*, and* h_j *for all* $j \in J^l$ *are continuously differentiable* $\mathcal{C}^1(\mathbb{R}^n)$*. Let* TNLP($\bar{\boldsymbol{x}}$) *be the first order Taylor approximation of* NLP *around* $\bar{\boldsymbol{x}}$*. Then,* $\bar{\boldsymbol{x}}$ *is a KKT point of* NLP *if and only if* $\bar{\boldsymbol{x}}$ *is an optimal solution of* TNLP($\bar{\boldsymbol{x}}$)*.*

Proof. Let $\bar{\boldsymbol{x}}$ be an (global) optimal point of TNLP($\bar{\boldsymbol{x}}$). Recall that for a linear programming problem optimality conditions are equivalent to KKT conditions, therefore we have,

$$\begin{aligned} \nabla f(\bar{\boldsymbol{x}}) + \sum_{i \in I^m} u_i \nabla g_i(\bar{\boldsymbol{x}}) + \sum_{j \in J^l} v_j \nabla h_j(\bar{\boldsymbol{x}}) &= \mathbf{0}, \\ u_i g_i(\bar{\boldsymbol{x}}) &= 0, \quad \text{for all } i \in I^m, \\ u_i &\geq 0, \quad \text{for all } i \in I^m. \end{aligned}$$

These properties are exactly the conditions ensuring that $\bar{\boldsymbol{x}}$ is a KKT point of NLP, so the theorem is proven. □

We start with a simple result, establishing some properties of the limit of the sequence generated by the SLP-NTR algorithm, in case of convergence. The result relies on some basic results on stability of linear programming problems, which can be seen, for instance, in López-Cerdá (2012) and Section 3.A.6 in González-Rueda (2017). Let $\mathcal{F}$ be the feasible set mapping associated to the sequence of linear problems $\{\text{TNLP}(\boldsymbol{x}^l)\}$.

Theorem 4. *Let $\{\boldsymbol{x}^l\}$ be the sequence generated by the* SLP-NTR *algorithm. Suppose that $\{\boldsymbol{x}^l\}$ converges to a point $\bar{\boldsymbol{x}}$ and $\mathcal{F}$ is* lower semicontinuous *at* TNLP$(\bar{\boldsymbol{x}})$*. The following sentences hold true:*

i) $\bar{\boldsymbol{x}}$ *is an optimal solution for* TNLP$(\bar{\boldsymbol{x}})$*.*

ii) $\bar{\boldsymbol{x}}$ *is a feasible point for* NLP*.*

iii) $\bar{\boldsymbol{x}}$ *is a KKT point for* NLP*.*

Proof. By the convergence of $\{\boldsymbol{x}^l\}$ to $\bar{\boldsymbol{x}}$, we have that the parameters defining each of the linear problems in the sequence $\{\text{TNLP}(\boldsymbol{x}^l)\}$ converge to those defining TNLP$(\bar{\boldsymbol{x}})$. Moreover, since $\mathcal{F}$ is lower semicontinuous at TNLP$(\bar{\boldsymbol{x}})$, we have that $\mathcal{F}$ is closed at TNLP$(\bar{\boldsymbol{x}})$. Therefore, the limit point $\bar{\boldsymbol{x}}$ of the sequence of optimal points $\{\boldsymbol{x}^l\}$ is an optimal solution of TNLP$(\bar{\boldsymbol{x}})$, and statement i) is proved.
Statement ii) is directly deduced from statement i) and the fact that a point $\bar{\boldsymbol{x}}$ is a feasible point of TNLP$(\bar{\boldsymbol{x}})$ if and only if $\bar{\boldsymbol{x}}$ is a feasible point of NLP. Statement iii) follows from statement i) and Lemma 3. □

Remark 1. The reader interested in the different approaches to establish the lower semicontinuity of the feasible set mapping may refer, for instance, to the survey López-Cerdá (2012). ◁

We present below an analogous result for PSLP-NTR.

Proposition 5. *Let $\{\boldsymbol{x}^l\}$ be the sequence generated by the* PSLP-NTR *algorithm. Suppose that $\{\boldsymbol{x}^l\}$ converges to a point $\bar{\boldsymbol{x}}$ and $\mathcal{F}$ is* lower semicontinuous *at* TPNP$^{\boldsymbol{\mu}}(\bar{\boldsymbol{x}})$*. Then, if $\bar{\boldsymbol{x}}$ is feasible for problem* NLP*, $\bar{\boldsymbol{x}}$ is a KKT point for* NLP*.*

Proof. Applying PSLP-NTR to solve NLP is equivalent to apply SLP-NTR to solve the penalized problem PNP$^{\boldsymbol{\mu}}$. Thus, by Theorem 4, $\bar{\boldsymbol{x}}$ is an optimal solution for TPNP$^{\boldsymbol{\mu}}(\bar{\boldsymbol{x}})$ and a KKT point for PNP$^{\boldsymbol{\mu}}$. Furthermore, since $\bar{\boldsymbol{x}}$ is feasible for problem NLP, by statement ii) of Theorem 1 we have that $\bar{\boldsymbol{x}}$ is a KKT point for NLP. □

The above two results imply that, if the sequences generated by SLP-NTR or PSLP-NTR converge, then the resulting solutions are, from a theoretical point of view, as good as the one given by the classic PSLP. However, when convergence is not achieved, any accumulation point of the sequence generated by PSLP is still a KKT point for problem PNP$^{\boldsymbol{\mu}}$, whereas we have seen that this may not be true for the methods without trust region. The rest of this section is devoted to show that the stopping criterion $\|\boldsymbol{x}^k - \boldsymbol{x}^{k+1}\| < \varepsilon$ for algorithms SLP-NTR and PSLP-NTR is sound even when the algorithms do not converge. To do so we prove that, whenever two consecutive points $\boldsymbol{x}^k$ and $\boldsymbol{x}^{k+1}$ in the sequence are close enough to each other, the point $\boldsymbol{x}^k$ is an almost feasible and almost KKT point for NLP in a sense we formally describe below.

4.1 Almost feasible points

Hereafter, we denote the feasible region of NLP by:

$$F^{NL} = \{\boldsymbol{x} \in X \subseteq \mathbb{R}^n : g_i(\boldsymbol{x}) \leq 0 \text{ for all } i \in I^m \text{ and } h_j(\boldsymbol{x}) = 0 \text{ for all } j \in J^l\} \subseteq K.$$

We denote the set of *active (binding) constraints* at $\boldsymbol{x}$ as $I_{\boldsymbol{x}} = \{i \in I^m : g_i(\boldsymbol{x}) = 0\}$. Below we formally introduce the definitions of *almost feasible* point and the set of *almost active constraints*.

Definition 4. Let $\boldsymbol{x} \in K$ and $\varepsilon > 0$. Then, $\boldsymbol{x}$ is an ε-*feasible* point if:

$$g_i(\boldsymbol{x}) \leq \varepsilon \text{ for all } i \in I^m \quad \text{and} \quad |h_j(\boldsymbol{x})| \leq \varepsilon \text{ for all } j \in J^l.$$

The set of ε-*active constraints* at $\boldsymbol{x}$ is defined as $I_{\boldsymbol{x}}^{\varepsilon} = \{i \in I^m : |g_i(\boldsymbol{x})| \leq \varepsilon\}$.

Let $\mathcal{I}$ be the set of *infinitesimal functions* at 0, that is:

$$\mathcal{I} = \{\, t : \mathbb{R}^+ \to \mathbb{R}^+ \text{ such that } \lim_{x \to 0} t(x) = 0\}.$$

We now present two intuitive results regarding ε-feasible points, whose proofs have been relegated to the Appendix. We denote by $d(\boldsymbol{x}, F)$ the euclidean distance between point $\boldsymbol{x}$ and set F.

Proposition 6. *Consider problem* NLP*. There is a non-decreasing function* $t^F \in \mathcal{I}$ *such that, if* $\boldsymbol{x} \in K$ *is an* ε*-feasible point, then* $d(\boldsymbol{x}, F^{NL}) \leq t^F(\varepsilon)$*.*

Proof. See Section A.1 in the Appendix. □

Theorem 7 below is the main instrumental result of our analysis. Informally, it says that if a point $\boldsymbol{x}$ is close enough to the feasible set, then there is a nearby feasible point $\bar{\boldsymbol{x}}$ such that all ε-active constraints at $\boldsymbol{x}$ are also active at $\bar{\boldsymbol{x}}$.

Theorem 7. *Consider problem* NLP*. There are* $\hat{t} \in \mathcal{I}$ *and* $\bar{\varepsilon} > 0$ *such that, if* $\boldsymbol{x}$ *is an* ε*-feasible point for* $0 < \varepsilon < \bar{\varepsilon}$*, then there is a feasible point* $\bar{\boldsymbol{x}} \in F^{NL}$ *such that:*

a) $\|\boldsymbol{x} - \bar{\boldsymbol{x}}\| \leq \hat{t}(\varepsilon)$*,*

b) $I_{\boldsymbol{x}}^{\varepsilon} \subseteq I_{\bar{\boldsymbol{x}}}$*.*

Proof. See Section A.2 in the Appendix. □

4.2 Almost KKT points

An *almost-KKT point* $\boldsymbol{x}$ or, more precisely, an $(\varepsilon_1, \varepsilon_2)$*-KKT point* $\boldsymbol{x}$ is, intuitively, a point that is almost feasible and, at the same time, it almost satisfies the KKT conditions. We refer the reader to Han et al. (2010), Haeser (2010), Haeser and Schuverdt (2011), and Andreani et al. (2011) for some references where related definitions of the notion of approximated-KKT are introduced but in different settings.

Definition 5. Given $\varepsilon_1, \varepsilon_2 \geq 0$, a point $\boldsymbol{x} \in K$ is an $(\varepsilon_1, \varepsilon_2)$*-KKT point* of NLP if

i) $\boldsymbol{x}$ is ε_1-feasible and

ii) for each $i \in I_{\boldsymbol{x}}^{\varepsilon_1}$, there is $u_i \geq 0$ and, for each $j \in J^l$, there is $v_j \in \mathbb{R}$, such that:

$$\|\,\nabla f(\boldsymbol{x}) + \sum_{i \in I_{\boldsymbol{x}}^{\varepsilon_1}} u_i \,\nabla g_i(\boldsymbol{x}) + \sum_{j \in J^l} v_j \,\nabla h_j(\boldsymbol{x})\| \leq \varepsilon_2.$$

Figure 5 illustrates the concept of $(\varepsilon_1, \varepsilon_2)$-KKT point and also illustrates the statement of Theorem 8 below, which says that if ε_1 and ε_2 are sufficiently small, then there is a feasible point $\bar{\boldsymbol{x}}$ that is very close to $\boldsymbol{x}$ and almost satisfies the KKT conditions. A bit more formally, $-\nabla f(\bar{\boldsymbol{x}})$ almost belongs to the cone of the gradients of the binding constraints at $\bar{\boldsymbol{x}}$, *i.e.*, either $-\nabla f(\bar{\boldsymbol{x}})$ belongs to the cone or it forms a very small angle with a vector in it.

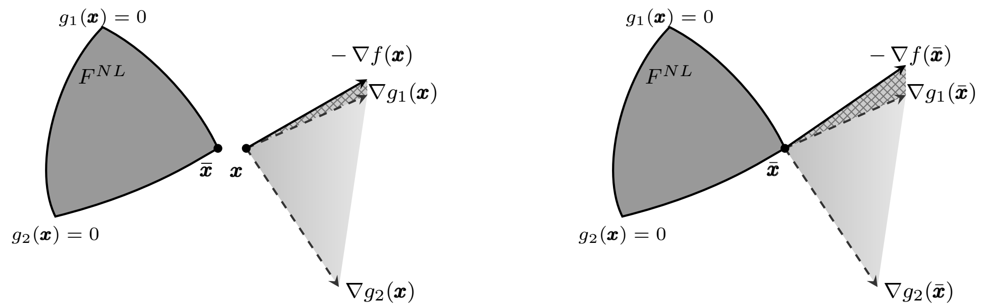


Figure 5: Illustration of the notion of almost-KKT point.

Remark 2. Recall that we are assuming that ∇f, ∇g_i for all $i \in I^m$ and ∇h_j for all $j \in J^l$ are continuous functions in K. Thus, since K is compact, these functions are uniformly continuous and there is $t' \in \mathcal{I}$ such that, for each $\delta > 0$, if $\|\boldsymbol{x} - \boldsymbol{y}\| < \delta$, then:

$$\begin{aligned} \|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| &\leq t'(\delta), \\ \|\nabla g_i(\boldsymbol{x}) - \nabla g_i(\boldsymbol{y})\| &\leq t'(\delta), \quad \forall i \in I^m, \text{ and} \\ \|\nabla h_j(\boldsymbol{x}) - \nabla h_j(\boldsymbol{y})\| &\leq t'(\delta), \quad \forall j \in J^l. \end{aligned} \qquad \triangleleft$$

Theorem 8. *There are $t_1, t_2 \in \mathcal{I}$ and $\bar{\varepsilon} > 0$ such that, if $\boldsymbol{x}$ is an $(\varepsilon_1, \varepsilon_2)$-KKT point with $0 \leq \varepsilon_1 < \bar{\varepsilon}$ and $\varepsilon_2 \geq 0$, then there is $\bar{\boldsymbol{x}} \in F^{NL}$ satisfying*

i) $\|\boldsymbol{x} - \bar{\boldsymbol{x}}\| \leq t_1(\varepsilon_1)$, *and*

ii) for each $i \in I_{\bar{\boldsymbol{x}}}$, there is $\bar{u}_i \geq 0$ and, for each $j \in J^l$, there is $\bar{v}_j \in \mathbb{R}$, such that:

$$\|\nabla f(\bar{\boldsymbol{x}}) + \sum_{i \in I_{\bar{\boldsymbol{x}}}} \bar{u}_i \nabla g_i(\bar{\boldsymbol{x}}) + \sum_{j \in J^l} \bar{v}_j \nabla h_j(\bar{\boldsymbol{x}})\| \leq \varepsilon^*,$$

where $\varepsilon^* = \varepsilon_2 + \big(1 + \sum_{i \in I_{\bar{\boldsymbol{x}}}} \bar{u}_i + \sum_{j \in J^l} \bar{v}_j\big) t_2(\varepsilon_1)$.

Proof. By definition, the $(\varepsilon_1, \varepsilon_2)$-KKT point $\boldsymbol{x}$ is ε_1-feasible. By Theorem 7, there are $\bar{\varepsilon} > 0$ and $t_1 \in \mathcal{I}$ such that, if $\varepsilon_1 < \bar{\varepsilon}$, then there is $\bar{\boldsymbol{x}} \in F^{NL}$ with $\|\boldsymbol{x} - \bar{\boldsymbol{x}}\| \leq t_1(\varepsilon_1)$ and $I_{\boldsymbol{x}}^{\varepsilon_1} \subseteq I_{\bar{\boldsymbol{x}}}$. On the other hand, by Definition 5, there are $u_i \geq 0$ for $i \in I_{\boldsymbol{x}}^{\varepsilon_1}$ and $v_j \in \mathbb{R}$ for $j \in J^l$ such that:

$$\|\nabla f(\boldsymbol{x}) + \sum_{i \in I_{\boldsymbol{x}}^{\varepsilon_1}} u_i \nabla g_i(\boldsymbol{x}) + \sum_{j \in J^l} v_j \nabla h_j(\boldsymbol{x})\| \leq \varepsilon_2.$$

Then, if we take $\bar{u}_i = u_i$ for $i \in I_{\boldsymbol{x}}^{\varepsilon_1}$, $\bar{u}_i = 0$ for $i \in I_{\bar{\boldsymbol{x}}} \setminus I_{\boldsymbol{x}}^{\varepsilon_1}$ and $\overline{v_j} = v_j$ for $j \in J^l$, we have that:

$$\begin{aligned} &\|\nabla f(\bar{\boldsymbol{x}}) + \sum_{i \in I_{\bar{\boldsymbol{x}}}} \bar{u}_i \nabla g_i(\bar{\boldsymbol{x}}) + \sum_{j \in J^l} \bar{v}_j \nabla h_j(\bar{\boldsymbol{x}})\| \leq \\ &\|\nabla f(\boldsymbol{x}) + \sum_{i \in I_{\boldsymbol{x}}^{\varepsilon_1}} u_i \nabla g_i(\boldsymbol{x}) + \sum_{j \in J^l} v_j \nabla h_j(\boldsymbol{x})\| + \|\nabla f(\boldsymbol{x}) - \nabla f(\bar{\boldsymbol{x}})\| + \\ &\sum_{i \in I_{\boldsymbol{x}}^{\varepsilon_1}} u_i \|\nabla g_i(\boldsymbol{x}) - \nabla g_i(\bar{\boldsymbol{x}})\| + \sum_{j \in J^l} v_j \|\nabla h_j(\boldsymbol{x}) - \nabla h_j(\bar{\boldsymbol{x}})\|. \end{aligned}$$

Now we use the assumption that the functions ∇f, ∇g_i for all $i \in I^m$ and ∇h_j for all $j \in J^l$ are uniformly continuous. Since $\|\boldsymbol{x} - \bar{\boldsymbol{x}}\| \leq t_1(\varepsilon_1)$, by Remark 2, we can take $t_2 = t' \circ t_1 \in \mathcal{I}$ so that:

$$\|\nabla f(\bar{\boldsymbol{x}}) + \sum_{i \in I_{\bar{\boldsymbol{x}}}} u_i \nabla g_i(\bar{\boldsymbol{x}}) + \sum_{j \in J^l} \bar{v}_j \nabla h_j(\bar{\boldsymbol{x}})\| < \varepsilon_2 + (1 + \sum_{i \in I_{\bar{\boldsymbol{x}}}} \bar{u}_i + \sum_{j \in J^l} \bar{v}_j) t_2(\varepsilon_1) = \varepsilon^*. \qquad \square$$

We now go back to the sequence arising from the application of the SLP-NTR algorithm.

Remark 3. Given $\boldsymbol{x}^0 \in K$, let $\{\boldsymbol{x}^l\}_{l\in\mathbb{N}}$ be the sequence resulting from the application of SLP-NTR. Then, for each $k \in \mathbb{N}$, $\boldsymbol{x}^k$ is a KKT point for the linearized problem TNLP($\boldsymbol{x}^{k-1}$), that is,

$$\nabla f(\boldsymbol{x}^{k-1}) + \sum_{i\in I^L_{\boldsymbol{x}^k}} u_i \nabla g_i(\boldsymbol{x}^{k-1}) + \sum_{j\in J^l} v_j \nabla h_j(\boldsymbol{x}^{k-1}) = \mathbf{0},$$

where $I^L_{\boldsymbol{x}^k} = \{\, i \in I^m : g_i(\boldsymbol{x}^{k-1}) + \nabla g_i(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) = 0 \,\}$ and $u_i \geq 0$ for all $i \in I^L_{\boldsymbol{x}^k}$ and $v_j \in \mathbb{R}$ for all $j \in J^l$. The scalars u_i and v_j are the Lagrange multipliers associated with $\boldsymbol{x}^k$ in the linear problem TNLP($\boldsymbol{x}^{k-1}$). $\lhd$

Remark 4. For the next result we need an additional regularity assumption: $g_i, h_j \in \mathcal{C}^2(K)$ for all $i \in I^m$ and all $j \in J^l$, *i.e.*, these functions are twice continuously differentiable in K. Because of the compactness of K, all second derivatives of g_i and h_j are then bounded and there is $R > 0$ such that the Hessian matrix satisfies $\|H(g_i)(\boldsymbol{x})\| < R$ and $\|H(h_j)(\boldsymbol{x})\| < R$ for all $\boldsymbol{x} \in K$, all $i \in I^m$, and all $j \in J^l$. By the first-order Taylor's formula with Lagrange remainder at a point $\boldsymbol{x}^0 \in K$, (see Theorem 5.6.2 in Cartan (1971)), we have

$$g_i(\boldsymbol{x}) = g_i(\boldsymbol{x}^0) + \nabla g_i(\boldsymbol{x}^0)^\intercal(\boldsymbol{x} - \boldsymbol{x}^0) + R^i_{\boldsymbol{x}^0}(\boldsymbol{x}), \quad \forall i \in I^m,$$

$$h_j(\boldsymbol{x}) = h_j(\boldsymbol{x}^0) + \nabla h_j(\boldsymbol{x}^0)^\intercal(\boldsymbol{x} - \boldsymbol{x}^0) + R^j_{\boldsymbol{x}^0}(\boldsymbol{x}), \quad \forall j \in J^l,$$

where the remainders $|R^i_{\boldsymbol{x}^0}(\boldsymbol{x})|$ and $|R^j_{\boldsymbol{x}^0}(\boldsymbol{x})|$ are bounded by $\frac{\|\boldsymbol{x}-\boldsymbol{x}^0\|^2}{2}R$. $\lhd$

The next result formally establishes that, if two consecutive points in the SLP-NTR sequence are at distance δ, then the second one is an $(\varepsilon_1, \varepsilon_2)$-KKT, where ε_1 and ε_2 go to zero as δ goes to zero.

Theorem 9. *Suppose that g_i and h_j belong to $\mathcal{C}^2(K)$ for all $i \in I^m$ and all $j \in J^l$. Let $\{\boldsymbol{x}^l\}$ be the sequence generated by* SLP-NTR *for problem* NLP*. There are $t_1, t_2 \in \mathcal{I}$ such that, if $\boldsymbol{x}^k$ satisfies that $\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\| \leq \delta$, then $\boldsymbol{x}^k$ is an $(\varepsilon_1, \varepsilon_2)$-KKT point for* NLP *with $\varepsilon_1 = t_1(\delta)$ and $\varepsilon_2 = (1 + \sum_{i\in I^L_{\boldsymbol{x}^k}} u_i + \sum_{j\in J^l} v_j)t_2(\delta)$, where u_i and v_j are the Lagrange multipliers associated with $\boldsymbol{x}^k$ in* TNLP($\boldsymbol{x}^{k-1}$).

Proof. First, we show that there is $t_1 \in \mathcal{I}$ such that, if $\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\| \leq \delta$ then $\boldsymbol{x}^k$ is ε_1-feasible with $0 < \varepsilon_1 = t_1(\delta)$. Note that $\boldsymbol{x}^k$ belongs to the feasible region of TNLP($\boldsymbol{x}^{k-1}$), *i.e.*,

$$\begin{aligned} g_i(\boldsymbol{x}^{k-1}) + \nabla g_i(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) \leq 0 \quad \forall i \in I^m,\\ h_j(\boldsymbol{x}^{k-1}) + \nabla h_j(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) = 0 \quad \forall j \in J^l. \end{aligned}$$

Combining the above equations with Remark 4, we have

$$\begin{aligned} g_i(\boldsymbol{x}^k) = g_i(\boldsymbol{x}^{k-1}) + \nabla g_i(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) + R^i_{\boldsymbol{x}^{k-1}}(\boldsymbol{x}^k) \leq R^i_{\boldsymbol{x}^{k-1}}(\boldsymbol{x}^k) \quad \forall i \in I^m,\\ h_j(\boldsymbol{x}^k) = h_j(\boldsymbol{x}^{k-1}) + \nabla h_j(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) + R^j_{\boldsymbol{x}^{k-1}}(\boldsymbol{x}^k) = R^j_{\boldsymbol{x}^{k-1}}(\boldsymbol{x}^k) \quad \forall j \in J^l, \end{aligned}$$

and hence we have

$$\begin{aligned} g_i(\boldsymbol{x}^k) \leq |R^i_{\boldsymbol{x}^{k-1}}(\boldsymbol{x}^k)| \leq \delta^2 R/2 = t_1(\delta), \quad \forall i \in I^m,\\ |h_j(\boldsymbol{x}^k)| \leq |R^j_{\boldsymbol{x}^{k-1}}(\boldsymbol{x}^k)| \leq \delta^2 R/2 = t_1(\delta), \quad \forall j \in J^l. \end{aligned}$$

Thus, $\boldsymbol{x}^k$ is an ε_1-feasible point with $\varepsilon_1 = t_1(\delta)$.

Second, we show that there are $t_2 \in \mathcal{I}$ and scalars $\bar{u}_i \geq 0$ for all $i \in I^{\varepsilon_1}_{\boldsymbol{x}^k}$ and $\bar{v}_j \in \mathbb{R}$ for all $j \in J^l$ such that:

$$\|\nabla f(\boldsymbol{x}^k) + \sum_{i \in I^{\varepsilon_1}_{\boldsymbol{x}^k}} \bar{u}_i \nabla g_i(\boldsymbol{x}^k) + \sum_{j \in J^l} \bar{v}_j \nabla h_j(\boldsymbol{x}^k)\| < (1 + \sum_{i \in I^L_k} u_i + \sum_{j \in J^l} v_j) t_2(\delta).$$

By construction, $I^L_{\boldsymbol{x}^k} \subseteq I^{\varepsilon_1}_{\boldsymbol{x}^k}$. Now, since $\boldsymbol{x}^k$ is a KKT point for TNLP$(\boldsymbol{x}^{k-1})$:

$$\nabla f(\boldsymbol{x}^{k-1}) + \sum_{i \in I^L_{\boldsymbol{x}^k}} u_i \nabla g_i(\boldsymbol{x}^{k-1}) + \sum_{j \in J^l} v_j \nabla h_j(\boldsymbol{x}^{k-1}) = \mathbf{0},$$

with $u_i \geq 0$ for all $i \in I^L_{\boldsymbol{x}^k}$ and $v_j \in \mathbb{R}$ for all $j \in J^l$. Then, we take $\bar{u}_i = u_i$ for all $i \in I^L_{\boldsymbol{x}^k}$, $\bar{u}_i = 0$ for all $i \in I^{\varepsilon_1}_{\boldsymbol{x}^k} \setminus I^L_{\boldsymbol{x}^k}$ and $\bar{v}_j = v_j \in \mathbb{R}$ for all $j \in J^l$. By the uniform continuity of the functions ∇f, ∇g_i for all $i \in I^m$ and ∇h_j for all $j \in J^l$ in K, using the notation in Remark 2, we can define $t_2 = t' \in \mathcal{I}$ and we have:

$$\begin{aligned}
&\|\nabla f(\boldsymbol{x}^k) + \sum_{i \in I^{\varepsilon_1}_{\boldsymbol{x}^k}} \bar{u}_i \nabla g_i(\boldsymbol{x}^k) + \sum_{j \in J^l} \bar{v}_j \nabla h_j(\boldsymbol{x}^k)\| && \leq \\
&\|\nabla f(\boldsymbol{x}^{k-1}) + \sum_{i \in I^L_{\boldsymbol{x}^k}} u_i \nabla g_i(\boldsymbol{x}^{k-1}) + \sum_{j \in J^l} v_j \nabla h_j(\boldsymbol{x}^{k-1})\| + \|\nabla f(\boldsymbol{x}^{k-1}) - \nabla f(\boldsymbol{x}^k)\| && + \\
&+ \sum_{i \in I^L_{\boldsymbol{x}^k}} u_i \|\nabla g_i(\boldsymbol{x}^{k-1}) - \nabla g_i(\boldsymbol{x}^k)\| + \sum_{j \in J^l} v_j \|\nabla h_j(\boldsymbol{x}^{k-1}) - \nabla h_j(\boldsymbol{x}^k)\| && \leq \\
&(1 + \sum_{i \in I^L_{\boldsymbol{x}^k}} u_i + \sum_{j \in J^l} v_j) t_2(\delta) = \varepsilon_2. && \square
\end{aligned}$$

Next we show an analogous result but for the PSLP-NTR algorithm.

Proposition 10. *Suppose that g_i and h_j belong to $\mathcal{C}^2(K)$ for all $i \in I^m$ and all $j \in J^l$. Let $\{\boldsymbol{x}^l\}$ be the sequence generated by* PSLP-NTR *for problem* NLP*. There are $\varepsilon_1, \varepsilon_2$ such that, if $\boldsymbol{x}^k$ is ε-feasible for problem* NLP *and it satisfies that $\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\| < \delta$, then $\boldsymbol{x}^k$ is an $(\varepsilon_1, \varepsilon_2)$-KKT point for* NLP *with $\varepsilon_1 = t_1(\delta) + \varepsilon$ and $\varepsilon_2 = (1 + \sum_{i \in I^{PL}_{\boldsymbol{x}^k}} u_i + \sum_{j \in J^l} v_j) t_2(\delta)$, where u_i and v_j are part of the Lagrange multipliers associated with $\boldsymbol{x}^k$ in* TPNP$^{\boldsymbol{\mu}}(\boldsymbol{x}^{k-1})$.

Proof. The PSLP-NTR algorithm consists in solving iteratively the problem TPNP$^{\boldsymbol{\mu}}$. Let us recall the definition of the problem TPNP$^{\boldsymbol{\mu}}(\boldsymbol{x}^{k-1})$:

$$\begin{array}{lll}
\min & p_L(\boldsymbol{x}) = f(\boldsymbol{x}^{k-1}) + \nabla f(\boldsymbol{x}^{k-1})^\intercal \boldsymbol{x} + \left[\sum_{i \in I^m} \mu_i y_i + \sum_{j \in J^l} \mu_j (p_j^+ + p_j^-) \right] & \\
\text{subject to} & g_i(\boldsymbol{x}^{k-1}) + \nabla g_i(\boldsymbol{x}^{k-1})^\intercal (\boldsymbol{x} - \boldsymbol{x}^{k-1}) \leq y_i & \forall i \in I^m \\
& y_i \geq 0 & \forall i \in I^m \\
& h_j(\boldsymbol{x}^{k-1}) + \nabla h_j(\boldsymbol{x}^{k-1})^\intercal (\boldsymbol{x} - \boldsymbol{x}^{k-1}) = p_j^+ - p_j^- & \forall j \in J^l \\
& p_j^+ \geq 0 & \forall j \in J^l \\
& p_j^- \geq 0 & \forall j \in J^l \\
& \boldsymbol{x} \in X = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\}. &
\end{array}$$

Note that given $\boldsymbol{x}^k \in X$, the minimum value of the objective function of the previous problem is realized by taking

$$\begin{aligned}
y_i &= \max\{0, g_i(\boldsymbol{x}^{k-1}) + \nabla g_i(\boldsymbol{x}^{k-1})^\intercal (\boldsymbol{x} - \boldsymbol{x}^{k-1})\}, \quad i \in I^m, \\
p_j^+ + p_j^- &= |h_j(\boldsymbol{x}^{k-1}) + \nabla h_j(\boldsymbol{x}^{k-1})^\intercal (\boldsymbol{x} - \boldsymbol{x}^{k-1})|, \quad j \in J^l.
\end{aligned}$$

Let $I^{PL}_{\boldsymbol{x}^k} = \{\, i \in I^m : g_i(\boldsymbol{x}^{k-1}) + \nabla g_i(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) - y_i = 0 \,\}$.

First note that if $\boldsymbol{x}^k$ is ε-feasible for problem NLP, then $\boldsymbol{x}^{k-1}$ is also ε^*-feasible for $\varepsilon^* = t(\delta)+\varepsilon$ with $t \in \mathcal{I}$. Indeed, by the continuity of g_i and h_j for all $i \in I^m$ and all $j \in J^l$ in K, we have

$$|g_i(\boldsymbol{x}^{k-1})| \leq |g_i(\boldsymbol{x}^k) - g_i(\boldsymbol{x}^{k-1})| + |g_i(\boldsymbol{x}^k)| \leq t(\delta) + \varepsilon = \varepsilon^*, \quad i \in I^m,$$

$$|h_j(\boldsymbol{x}^{k-1})| \leq |h_j(\boldsymbol{x}^k) - h_j(\boldsymbol{x}^{k-1})| + |h_j(\boldsymbol{x}^k)| \leq t(\delta) + \varepsilon = \varepsilon^*, \quad j \in J^l.$$

We prove now that there is $\varepsilon_1 > 0$ such that $I^{PL}_{\boldsymbol{x}^k} \subseteq I^{\varepsilon_1}_{\boldsymbol{x}^k}$. Given $i \in I^{PL}_{\boldsymbol{x}^k}$, we have $y_i = g_i(\boldsymbol{x}^{k-1}) + \nabla g_i(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1})$. Therefore, by the sub-additivity absolute value and the Cauchy-Schwarz inequality, we have

$$|y_i| = |g_i(\boldsymbol{x}^{k-1}) + \nabla g_i(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1})| \leq \varepsilon^* + \|\nabla g_i(\boldsymbol{x}^{k-1})\|\delta.$$

Then, by Remark 4 we have

$$\begin{aligned} |g_i(\boldsymbol{x}^k)| &= |g_i(\boldsymbol{x}^{k-1}) + \nabla g_i(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) + R^i_{\boldsymbol{x}^{k-1}}(\boldsymbol{x}^k)| \leq |y_i| + |R^i_{\boldsymbol{x}^{k-1}}(\boldsymbol{x}^k)| \\ &\leq \varepsilon^* + ||\,\nabla g_i(\boldsymbol{x}^{k-1})||\delta + \frac{\delta^2}{2}R = t_1(\delta) + \varepsilon, \end{aligned}$$

being $t_1(\delta) = t(\delta) + ||\,\nabla g_i(\boldsymbol{x}^{k-1})||\delta + \frac{\delta^2}{2}R$. Let $\varepsilon_1 = t_1(\delta) + \varepsilon$, then $i \in I^{\varepsilon_1}_{\boldsymbol{x}^k}$ and we have showed that $I^{PL}_{\boldsymbol{x}^k} \subseteq I^{\varepsilon_1}_{\boldsymbol{x}^k}$. Besides, $\varepsilon_1 > \varepsilon$, so $\boldsymbol{x}^k$ is also ε_1-feasible for problem NLP.

Finally, we show that there are ε_2 and scalars $u_i \geq 0$ for all $i \in I^{\varepsilon_1}_{\boldsymbol{x}^k}$ and $v_j \in \mathbb{R}$ for all $j \in J^l$ such that

$$\|\,\nabla f(\boldsymbol{x}^k) + \sum_{i \in I^{\varepsilon_1}_{\boldsymbol{x}^k}} u_i\, \nabla g_i(\boldsymbol{x}^k) + \sum_{j \in J^l} v_j\, \nabla h_j(\boldsymbol{x}^k)\| < \varepsilon_2.$$

Since $\boldsymbol{x}^k \in X$ is a KKT point for problem TPNP$^{\boldsymbol{\mu}}(\boldsymbol{x}^{k-1})$, there are Lagrange multipliers u^1_i, $u^2_i > 0$ for all $i \in I^m$ and v^1_j, v^2_j and $v^3_j \in \mathbb{R}$ for all $j \in J^l$, associated to the different sets of constraints, respectively, such that

$$\nabla f(\boldsymbol{x}^{k-1}) + \sum_{i \in I^{PL}_{\boldsymbol{x}^k}} u^1_i\, \nabla g_i(\boldsymbol{x}^{k-1}) + \sum_{j \in J^l} v^1_j\, \nabla h_j(\boldsymbol{x}^{k-1}) = \mathbf{0}, \tag{1}$$

$$\begin{aligned} &\mu_i - u^1_i - u^2_i = 0, \quad i \in \{i \in I^m : g_i(\boldsymbol{x}^{k-1}) + \nabla g_i(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) = 0\}, \\ &\mu_j - v^1_j - v^2_j = 0, \quad j \in \{j \in J^l : h_j(\boldsymbol{x}^{k-1}) + \nabla h_j(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) > 0\}, \\ &\mu_j + v^1_j - v^3_j = 0, \quad j \in \{j \in J^l : h_j(\boldsymbol{x}^{k-1}) + \nabla h_j(\boldsymbol{x}^{k-1})^\intercal(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) < 0\}. \end{aligned}$$

Then, we take $u_i = u^1_i$ for $i \in I^{PL}_{\boldsymbol{x}^k}$, $u_i = 0$ for $i \in I^{\varepsilon_1}_{\boldsymbol{x}^k} \setminus I^{PL}_{\boldsymbol{x}^k}$ and $v_j = v^1_j \in \mathbb{R}$ for $j \in J^l$. By the uniform continuity of ∇f, ∇g_i and ∇h_j in K, using the notation in Remark 2, and taking into account Equation (1), we have

$$\begin{aligned} &\|\nabla f(\boldsymbol{x}^k) + \sum_{i \in I^{\varepsilon_1}_{\boldsymbol{x}^k}} u_i\, \nabla g_i(\boldsymbol{x}^k) + \sum_{j \in J^l} v_j\, \nabla h_j(\boldsymbol{x}^k)\| \leq \\ &\|\nabla f(\boldsymbol{x}^{k-1}) + \sum_{i \in I^{PL}_{\boldsymbol{x}^k}} u^1_i\, \nabla g_i(\boldsymbol{x}^{k-1}) + \sum_{j \in J^l} v^1_j\, \nabla h_j(\boldsymbol{x}^{k-1})\| + \|\nabla f(\boldsymbol{x}^{k-1}) - \nabla f(\boldsymbol{x}^k)\| + \\ &+ \sum_{i \in I^{PL}_{\boldsymbol{x}^k}} u^1_i \|\nabla g_i(\boldsymbol{x}^{k-1}) - \nabla g_i(\boldsymbol{x}^k)\| + \sum_{j \in J^l} v^1_j \|\nabla h_j(\boldsymbol{x}^{k-1}) - \nabla h_j(\boldsymbol{x}^k)\| \leq \\ &(1 + \sum_{i \in I^{PL}_{\boldsymbol{x}^k}} u^1_i + \sum_{j \in J^l} v^1_j) t_2(\delta) = (1 + \sum_{i \in I^{PL}_{\boldsymbol{x}^k}} u_i + \sum_{j \in J^l} v_j) t_2(\delta) \varepsilon_2. \end{aligned}$$

□

5 Computational results of the 2-step SLP algorithm

In this section we present the results of a series of tests in which the performance of the 2-step SLP algorithm is compared with that of classic PSLP and state of the art solvers for NLP and MINLP problems such as local solvers Knitro 10.3.0 (Byrd et al., 2006) and Ipopt 3.12.8 (Wächter and Biegler, 2006) and global solver BARON 17.10.13 (Tawarmalani and Sahinidis, 2005). These solvers have been interfaced through AMPL modeling language (Fourer et al., 1990), version 20180423, and the corresponding problem instances can be downloaded from `https://goo.gl/1We8yh`. The LP and MILP problems associated with the SLP algorithms have been solved with Gurobi 7.5.0 (Gurobi Optimization Inc., 2018).

Next, we briefly describe the parameters used for the SLP algorithms and the above solvers. For the SLP algorithms, we take $\varepsilon = 10^{-4}$ as stopping criteria, which is the unique parameter needed by SLP-NTR. The additional parameters that we use to configure the PSLP algorithm are the ones suggested in Bazaraa et al. (2006): $\rho_0 = 10^{-6}$, $\rho_1 = 0.25$, $\rho_2 = 0.75$, $\beta = 0.5$ and $\Delta_{LB} = 0$. With respect to the penalty parameter, in our experience $\mu = 10$ has been sufficiently large for all problems considered. We take 200 as the limit number of iterations of SLP-NTR and PSLP algorithms.[18] Nonlinear solvers use their default parameters except for the feasibility tolerance that is set to 10^{-4}, which is also imposed as feasibility threshold for the solutions reported by SLP algorithms. The following time limits were used in the different tests sets: 300 seconds for Test 1 (NLP), 600 seconds for Test 2 (MINLP), and no time limit for Test 3 (NLP) and Test 4 (MINLP), given that the problems in the latter two sets are relatively small. All computations were performed on a Linux cluster, whose nodes have Intel(R) Xeon(R) 2.40GHz quad core processors and 36GB of RAM. In order to deliver comparable results across solvers, no parallelization was allowed to any of them.

It is worth noting that, through the course of our research, we have observed that the computational results presented in this section seem robust to different variations of instances and modeling choices. For instance, the results reported in González-Rueda (2017) were run on a slight variation of the model presented here and are very similar qualitatively. Interestingly, Section 4.5.2 in González-Rueda (2017) also presents results on the performance of 2-step SLP on a different class of problems: multicommodity flow problems. The analysis there is run on a set of problems taken from Babonneau and Vial (2009) and the reason for not reporting the associated results in this paper is that the performance of all solution techniques is very similar: PSLP, SLP-NTR, 2-step SLP, and state of the art NLP solvers deliver very similar results (although the SLP algorithms required significantly shorter running times).

5.1 Computational tests on the Spanish gas transmission network

In this section we want to show the practical relevance of the 2-step SLP approach introduced in this work over a set of instances of the Spanish gas transmission network. We also give details regarding the performance of the SLP-NTR in the first step of the algorithm.[19]

[18] In case the algorithm SLP-NTR do not converge, we select the $\boldsymbol{x}^k$ such that the distance $\|\boldsymbol{x}^k - \boldsymbol{x}^{k+1}\|$ is the smallest in the sequence generated by the algorithm.

[19] In our analysis here we do not enter into the specific results of each of the instances being solved, but the interested reader may refer to Bermúdez et al. (2015), where a detailed case study is developed. In particular, the results obtained after the application of 2-step SLP represent savings of around 82% with respect to the costs associated to the operation reported by the Spanish Technical System Manager. Yet, this operation was reported in a study about security of supply, with no mention to an optimized operation.

5.1.1 Test sets for the Spanish gas transmission network

We employ a set of 120 different instances of the Spanish gas network to test the algorithms. The process to generate the instances is as follows:

- We consider 12 representative instances of the Spanish gas network corresponding to different gas consumption periods over the year (winter, summer, peak days,...). Thus, the distribution of the gas demand over the different regions of the network varies and the total gas demand is also different for every instance.
- Then, for each of the representative instances, we generate 9 new instances rescaling the demand of the consumption points of the original instance. In particular:
  - We generate 3 new instances with lower demands: 70%, 80% and 90% of the original one.
  - We generate 6 new instances with higher demands. In this case, if we denote by max_{cap} the maximum amount of gas that can be sent to consumers from the supply points and by ini_{cons} the total consumption of the original instance, we compute:

    $$\Delta_{cons} = \frac{max_{cap} - ini_{cons}}{10}.$$

    Then, the total demand of these six scenarios is: $ini_{cons} + i \cdot \Delta_{cons}$ for $i = 1 \ldots, 6$.
- Note that this process does not ensure feasibility of the resulting instances and, indeed, for some of them no solver was able to find a feasible solution.

Compressor stations are modeled with the duplication approach described in Section 2.2. The resulting instances have around 500 nodes and 500 edges, leading to optimization models with around 1000 variables, with over 500 linear and 500 nonlinear (and nonconvex) constraints. Based on these instances, we define two test sets, one for NLP problems and one for MINLP ones.

- Test 1 (NLP): The NLP problems associated to the instances described above contain the following objective function and constraints:
  - *Solvers interfaced through AMPL.* The model includes the constraints given by Equations (BC), (AP_{COM}), (FC), (PL), and (ZF) (See Section 2.1). The friction factor is computed with Equation (FF). To get an NLP problem, no control valves are considered and Equations (UC) and (GC) are used to model compressors. The objective function is given by Equation (OBJ).
  - SLP *algorithms.* For the PSLP run in the second step of the 2-step SLP algorithm, we use exactly the same model we have just described. For the SLP-NTR model run in step one, we make the following minor variations/simplifications:
    - To ensure feasibility after every iteration, SLP-NTR uses the enlarged feasible region as described in Section 2.2. Thus, Equation (BC_{EXTRA}) is used instead of (BC), and the corresponding penalized term is added to the function, now given by Equation ($\text{OBJ}_{\text{EXTRA}}$).
    - We rely on the iterative nature of the algorithm to use the simplified modeling to iteratively update the compressibility factor as described in Section 2.2. Similarly, to simplify the linearizations at each iteration, the average pressure in Equation (AP_{COM}) is taken as a constant that is updated after every iteration.

It is worth noting that the above variations of the model are not strictly needed. Instead of enlarging the feasible region one could just apply PSLP-NTR in the first step. Further, the simplifications regarding the average pressure and the computation of the compressibility factor are quite minor, improve the speed of the algorithm, and will anyway be adjusted in the second step of the algorithm.

- Test 2 (MINLP): The only difference with respect to the NLP model defined in Test 1 is that we additionally include operating costs at supply points, including boil-off costs, so we obtain MINLP problems with 10 binary variables. The main reason for using MINLP problems with such a small number of binary variables is that they can be solved by enumeration, which is very convenient for the analysis. Thus, with respect to the model used for Test 1, we add Equations (BO-1) and (BO-2) (See Section 2.2). Accordingly, for the objective function, Equation ($\text{O}\textsc{bj}_{\text{boil-off}}$) is used instead of Equation ($\text{O}\textsc{bj}_{\text{boil-off}}$). The same considerations in Test 1 apply to the model variations associated to SLP-NTR so, in particular, the objective function includes both the penalization associated to the enlargement of the feasible region and the operating costs at supply points.

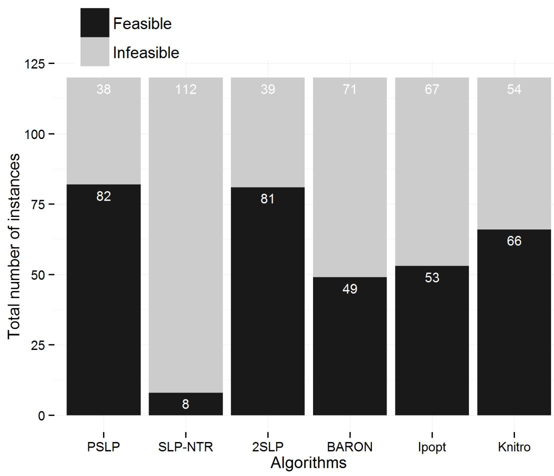
All solvers and SLP algorithms have been given the same initial solution. In particular, we have set up an initial solution in which the flow through a pipe is proportional to its diameter, except for those edges whose flows can be pre-computed explicitly given the demands and the topology of the network, which have been intitialized to these values. The initial value for the pressure at a node is set at the average of its lower and upper bounds. The initial solutions obtained with such a naive approach are typically far from being feasible.

5.1.2 Computational results on NLP problems (Test 1)

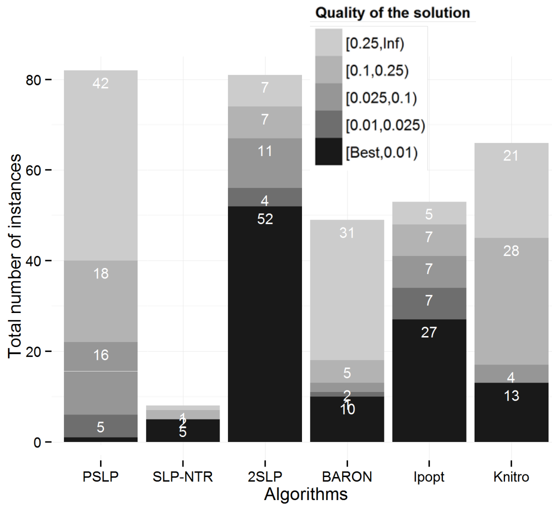
Figure 6(a) shows the number of instances solved to feasibility by the different algorithms/solvers. The behavior of PSLP and 2-step SLP is very similar, finding a feasible solution for 82 and 81 instances, respectively (they differ only in one instance, for which the 2-step SLP finds a solution with a violation of $2.7 \cdot 10^{-4}$, quite close to the feasibility threshold 10^{-4}). On the other hand, the state of the art solvers fail to recognize the feasibility of a considerable number of instances, where Knitro is the solver reaching feasibility in more instances (66). SLP-NTR exhibits the worst performance, finding a feasible solution in only 8 cases. The main reason for this behavior is that SLP-NTR did not converge for most of the instances, as we had anticipated in Section 3.3. More precisely, SLP-NTR did not converge for 108 instances, PSLP always converges, and 2-step SLP does not converge for 2 instances (for which none of the approaches found a feasible solution).

Figure 6(b) represents the quality of the feasible solutions. Given a feasible instance, we take the best objective function found by the algorithms that solved it to feasibility and we compute the relative difference of the objective function of each algorithm with respect to it. Then, we represent, for each solver, the number of instances that fall within each "quality interval": worsening of at most 1%, between 1% and 2.5%, and so on. Figure 6(b) shows a very good performance of the 2-step SLP algorithm, which will be a recurrent feature in all the test instances discussed in this section. Note that the quality of the solutions of 2-step SLP is clearly superior to the ones found by PSLP, which suggests that the solution provided by the SLP-NTR in the first step (despite not having converged) is a good starting point that helps the PSLP to find better solutions in second step.

Figure 7 represents box plots with the computational times. As we can see BARON is the slowest solver, which is natural since it is a global optimization solver. In particular, in Figure 7(a), where only instances solved to feasibility are considered, BARON always reaches the time limit imposed of 300 seconds (note that this does not mean that BARON needs 300

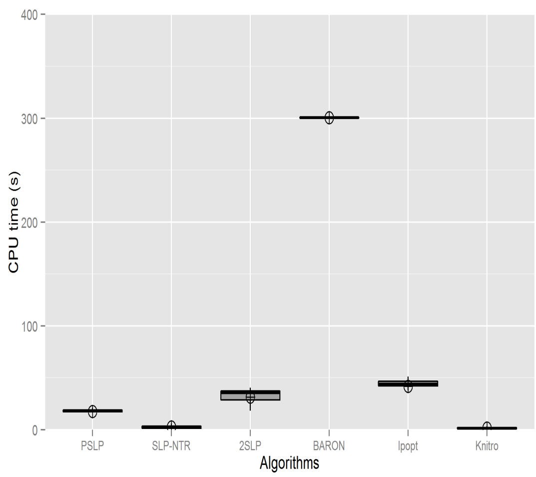


(a) Feasibility results

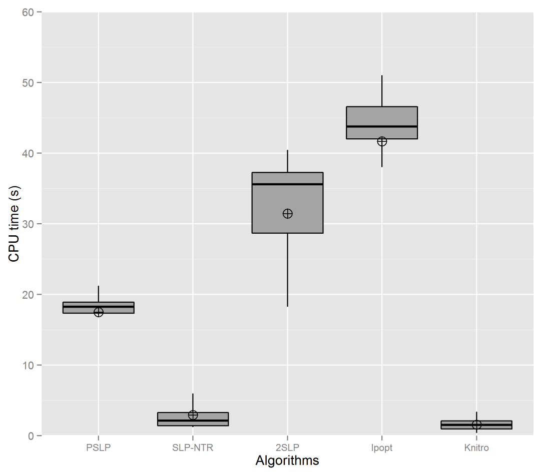


(b) Quality of the feasible solutions.

Figure 6: Summary of results for Test 1 (NLP).



(a) CPU times in instances solved to feasibility by each solver.



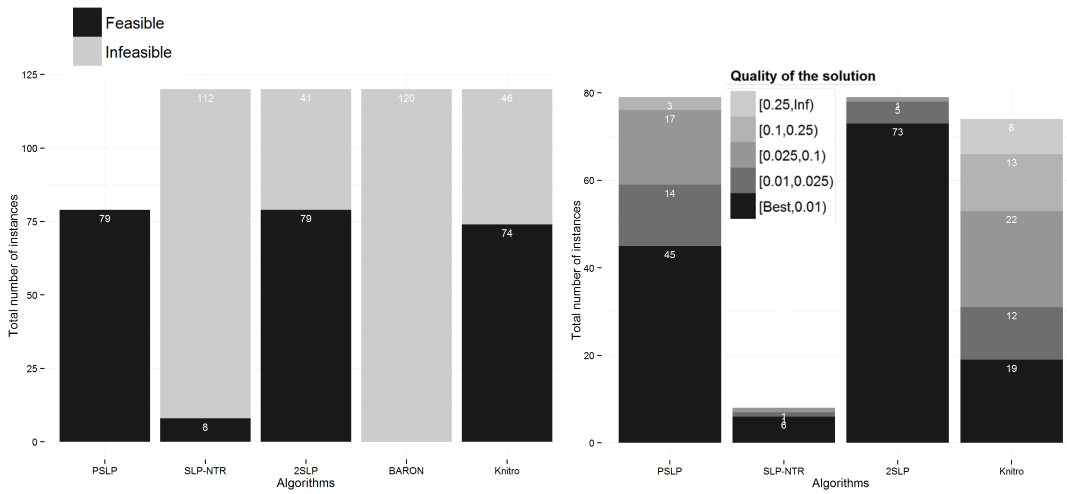
(b) CPU times in instances solved to feasibility (without BARON).

Figure 7: Computational times (in seconds) for Test 1 (NLP).

seconds to find a feasible solution, but trying to establish its global optimality). Interestingly, Figure 7(b) shows that the best computational times are achieved by SLP-NTR and Knitro. Yet, overall, it seems that 2-step SLP outperforms the rest of the algorithms/solvers in Test 1.

5.1.3 Computational results on MINLP problems (Test 2)

We move now to Test 2, where for each instance we have to solve a MINLP problem. Although we cannot apply PSLP to these problems, we still denote by PSLP the solution obtained by solving, with PSLP, each instance of the problem by enumeration of the combinations of the binary variables. Furthermore, Ipopt is not included in the analysis since it does not solve MINLP problems.



(a) Feasibility results

(b) Quality of the feasible solutions.

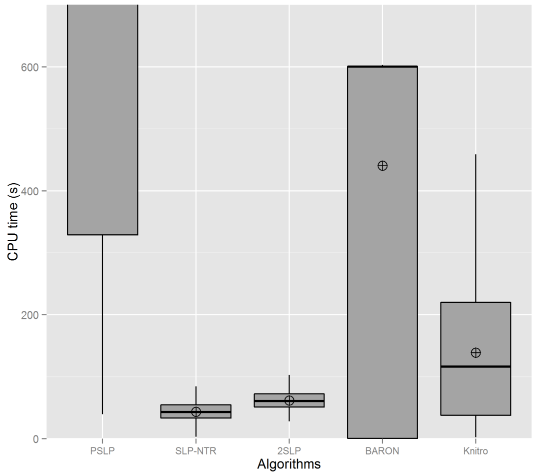
Figure 8: Summary of results for Test 2 (MINLP).

As Figure 8 shows,[20] the behavior of SLP-NTR and 2-step SLP algorithms is similar to the one in Test 1 (NLP) regarding feasibility. However, now BARON is not able to find any feasible solution while Knitro improves its performance. We must take into account that we impose a time limit of 600 seconds for the state of the art solvers and BARON often reached this time limit; see Figure 9(a).[21] As expected, the computational time of the PSLP enumeration is the largest one. Regarding the convergence of the SLP algorithms, the behavior is similar to the one discussed in Test 1 (NLP).

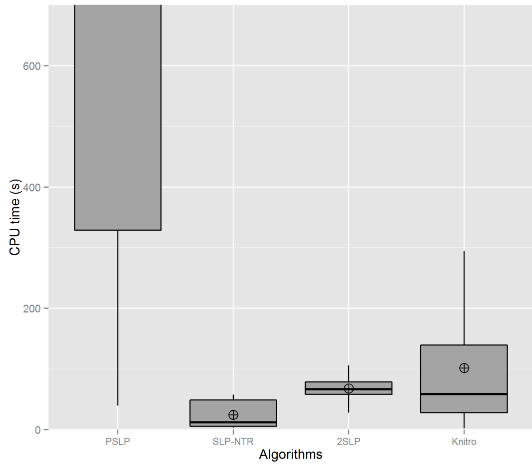
Finally, Figure 8(b) shows that the quality of the solution provided by the 2-step SLP outperforms both the PSLP enumeration and Knitro. In particular, the fact that 2-step SLP often finds a better solution that the best one found by the PSLP enumeration probably means that local solutions were found for some of the NLP subproblems.

[20] We only ran the PSLP by enumeration for those instances solved to feasibility by the 2-step SLP.

[21] This bad behavior of BARON had already been observed, although not so dramatically, in the analysis in González-Rueda (2017) (which was ran on the same instances, but with some minor modeling differences).



(a) CPU times considering all the instances.



(b) CPU times in instances solved to feasibility by each algorithm.

Figure 9: Computational times (in seconds) for Test 2 (MINLP).

5.2 Computational tests on the Belgian gas transmission network

In order to get a broader view of the performance of 2-step SLP on optimization problems in gas transmission networks, we also study its performance on the optimization model developed in De Wolfe and Smeers (2000). The model is similar to the one used in Test 1, but some of the nonlinear expressions have been simplified. For the sake of exposition we do not present here the detailed description of the differences between the two models, but the interested reader may refer to Section 4.5.1 in González-Rueda (2017).

5.2.1 Test sets for the Belgian gas transmission network

We consider 300 different instances of the Belgian gas transmission network, wich are generated by randomly modifying some parameters associated to a reference instance taken from data for the Belgian gas transmission network provided in Appendix A of De Wolfe and Smeers (2000). The reference instance contains 20 nodes and 24 edges (including 3 compressors), being the resulting problems much smaller than those of Test 1 and Test 2.

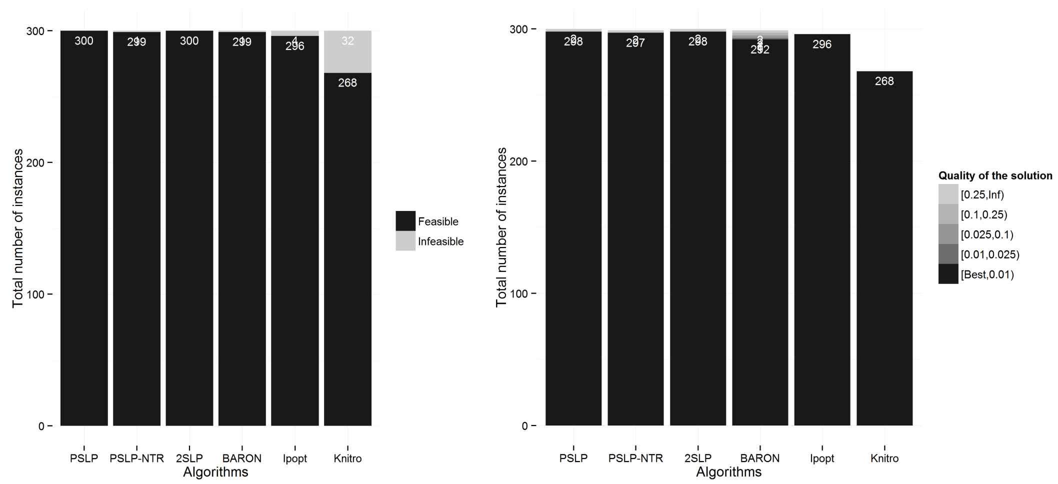
Based on the above instances, we define two test sets, one for NLP problems (with around 60 continuous variables, 20 linear constraints and 25 nonlinear constraints) and one for MINLP ones (with around 60 continuous variables, 20 integer variables, 125 linear constraints, and 25 nonlinear ones).

- Test 3 (NLP): All solvers and SLP algorithms run on the same model: the NLP model in De Wolfe and Smeers (2000).
- Test 4 (MINLP): All solvers and SLP algorithms run on the same model. To obtain a MINLP problems, the approach has been to replace the absolute value in the pressure loss constraints, $|q|$, with a binary variable to account for the sign of q. The resulting MINLP problems are mathematically equivalent to the original NLP ones.

In these tests PSLP-NTR is run in the first step of 2-step SLP instead of SLP-NTR. The initial solution has been defined as follows: flows have been set to the solution of solving the model using only the linear constraints and pressures to the averages of the bounds.

5.2.2 Computational results on NLP problems (Test 3)

In Figure 10(a) we can see the feasibility analysis related to Test 3 (NLP). We can see that the SLP algorithms, BARON, and Ipopt exhibit a good performance, but Knitro fails to find a feasible solution for 32 instances. Remarkably, for this model the PSLP-NTR only fails to recognize the feasibility of 1 instance, being the only one for which it does not converge, while the PSLP and the 2-step SLP converge for all the instances. Regarding the quality of the solution, given in Figure 10(b), all the algorithms/solvers find the best solution for almost all the instances they solve to feasibility.



(a) Feasibility results

(b) Quality of the feasible solutions.

Figure 10: Summary of results for Test 3 (NLP).

Concerning the computational time, given that most instances were identified as feasible, all graphics with computational times are very similar and we only represent the one considering all instances. We can see in Figure 11 that all solvers deliver running times under 1 second. Ipopt and Knitro are especially fast, with PSLP-NTR being close to them.

5.2.3 Computational results on MINLP problems (Test 4)

First, recall that the optimal objective functions for problems in Test 3 (NLP) and Test 4 (MINLP) coincide, since the latter are just equivalent reformulations of the former. Thus, similar results could be expected.

Regarding feasibility, depicted in Figure 12(a), the behavior is indeed similar to the one in Test 3. Interestingly, the number of feasible instances found by Knitro increases from 268 to 289. With respect to the quality of the solutions (see Figure 12(b)) the results are also similar, with BARON now finding the best solution for all the instances. Finally, regarding computational times, Figure 13 shows that the PSLP-NTR is the fastest algorithm and Knitro the slowest one (the result would not change if we looked only at the solved instances).

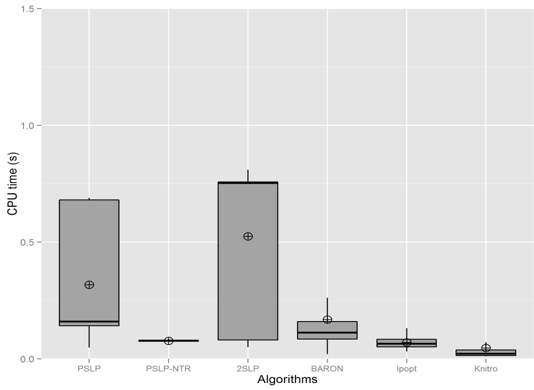
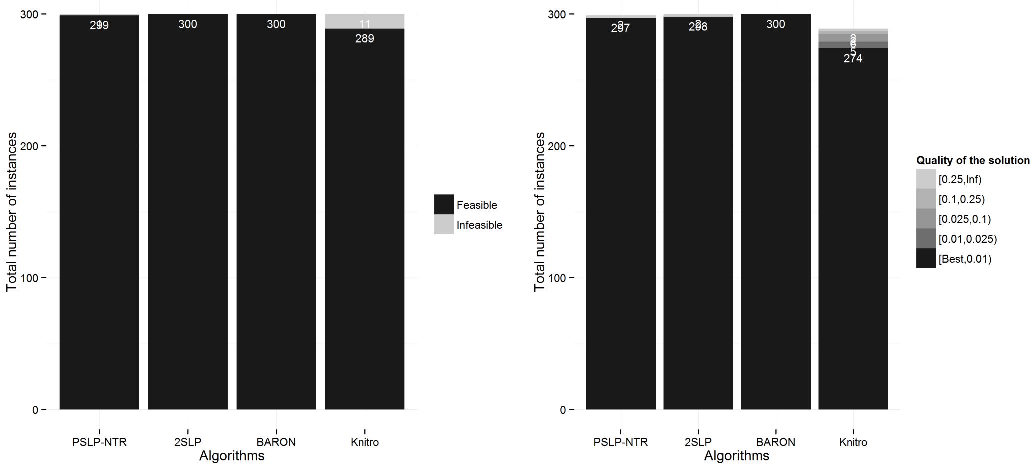


Figure 11: Computational times (in seconds) for Test 3 (NLP).



(a) Feasibility results

(b) Quality of the feasible solutions.

Figure 12: Summary of results for Test 4 (MINLP).

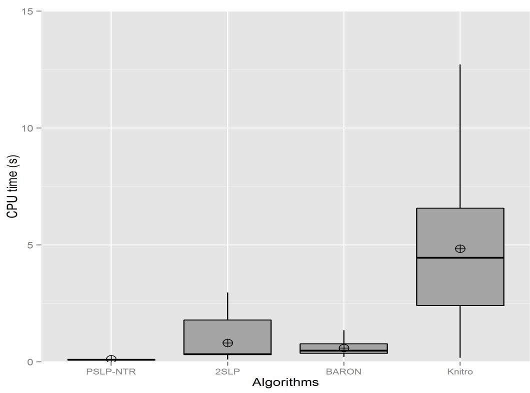


Figure 13: Computational times (in seconds) for Test 4 (MINLP).

6 Conclusions and future research

We have presented a new algorithm, 2SLP, that is a modification of classic SLP algorithms with the additional feature that it can be applied not only to find local solutions to NLP problems, but also as a heuristic for MINLP problems.

The 2SLP algorithm was born as part of a collaboration with a partner in the gas industry, to solve optimization problems in gas transmission networks.

In Section 4 we presented some theoretical results underlying the SLP-NTR algorithm, which is run in the first step of 2SLP and discussed its main limitations (such as poor convergence properties) and strengths (ease of implementation, straightforward application to MINLP problems).

The performance of the 2SLP algorithm was then studied in a series of tests on different NLP and MILP instances of optimization problems in gas networks. In these tests, the 2SLP algorithm outperformed the classic SLP approaches and, moreover, also exhibited results in most cases superior to those of state of the art solvers.

The future lines of research within this project are mostly driven by the requirements of our industrial partner, and nowadays focus on two extensions of the models under consideration:

i) Inclusion of pooling constraints, that allow to keep track of different properties of the gas in the network in the case in which there is a heteregeneous mix of gases flowing through it. The pooling problem is a classic problem in energy related industries and has been widely studied (see Misener and Floudas (2009)).

ii) Inclusion of uncertainty on prices and demand, so that GANESO$^{\text{TM}}$ can be used to help to take decisions regarding mid and long-term infrastructure planning. This extension requires the combination of the NLP and MINLP techniques discussed in this paper with those of stochastic optimization and algorithms such as Progressive Hedging (see Birge and Louveaux (2011) and Rockafellar and Wets (1991)).

Acknowledgements. This project has been partially funded by Reganosa company under a contract with ITMATI. The authors also acknowledge support from Ministerio de Economía y Competitividad and FEDER through project MTM2014-60191-JIN and from Xunta de Galicia through project ED431C-2017/38. Ángel M. González-Rueda acknowledges support from Ministerio de Educación through Grant FPU13/01130.

References

Ali, H., A. Batchelor, E. M. L. Beale, J. Beasley. 1978. Mathematical models to help manage the oil resources of Kuwait. *Internal report, Scientific Control Systems* .

Andreani, R., G. Haeser, J. M. Martínez. 2011. On sequential optimality conditions for smooth constrained optimization. *Optimization* **60** 627–641.

Babonneau, F., J.-P. Vial. 2009. ACCPM with a nonlinear constraint and an active set strategy to solve nonlinear multicommodity flow problems. *Mathematical Programming* **120**, no. 1 170–210.

Baker, T. E., L. S. Lasdon. 1985. Successive linear programming at Exxon. *Management Science* **31** 264–274.

Bazaraa, M. S., H. D. Sherali, C. M. Shetty. 2006. *Nonlinear programming: Theory and algorithms.* John Wiley and Sons.

Beale, E. M. L. 1978. Nonlinear programming using a general mathematical programming system. *Design and Implementation of Optimization Software* 259–279.

Bermúdez, A., J. González-Díaz, F. J. González-Diéguez, A. M. González-Rueda, M. P. F. de Córdoba. 2015. Simulation and optimization models of steady-state gas transmission networks. *Energy Procedia* **64** 130–139.

Birge, J. R., F. Louveaux. 2011. *Introduction to Stochastic Programming*. Springer-Verlag New York.

Boddington, C. E., W. C. Randall. 1979. Nonlinear programs for product blending. *Joint National TIMS/ORSA Meeting, New Orleans, April/May,* 105.

Byrd, R. H., J. Nocedal, R. A. Waltz. 2006. *Large-scale nonlinear optimization*, vol. 83 of *Nonconvex optimization and its applications book series*, chap. KNITRO: An integrated package for nonlinear optimization, 35–59. Springer US, Boston, MA.

Cartan, H. 1971. *Differential Calculus*. Hermann, Paris.

Carter, R. G. 1998. Pipeline optimization: dynamic programming after 30 years. In *Proceedings of the 30th PSIG annual meeting*.

Cobos-Zaleta, D., R. Z. Ríos-Mercado. 2002. A MINLP model for minimizing fuel consumption on natural gas pipeline networks. In *Proceedings of the XI Latin-Ibero-American Conference on Operations Research*. Concepción, Chile.

De Wolfe, D., Y. Smeers. 2000. The gas transmission problem solved by an extension of the simplex algorithm. *Management Science* **46**, no. 11 1454–1465.

Ehrhardt, K., M. Steinbach. 2002. Betriebskostenminimierung für Erdgas-Transportnetze. Tech. rep., Konrad Zuse Zentrum für Informationstechnik, Berlin.

Enagás GTS. 2010. *Infraestructuras en operación a junio de 2010*. Enagás GTS.

Fourer, R., D. M. Gay, B. W. Kernighan. 1990. A Modeling Language for Mathematical Programming. *Management Science* **36** 519–554.

González-Diéguez, F. J. 2017. *Modeling, simulation and optimization of gas transport networks*. Ph.D. thesis, University of Santiago de Compostela.

González-Rueda, A. M. 2017. *Gas transmission networks: optimization algorithms and cost allocation methodologies*. Ph.D. thesis, University of Santiago de Compostela.

Griffith, R., R. A. Stewart. 1961. A nonlinear programming technique for the optimization of continuous processing systems. *Management Science* **7** 379–392.

Gurobi Optimization Inc. 2018. Gurobi Optimizer Reference Manual. http://www.gurobi.com.

Haeser, G. 2010. On the global convergence of interior-point nonlinear programming algorithms. *Computational & Applied Mathematics* **29** 125–138.

Haeser, G., M. L. Schuverdt. 2011. On approximate KKT condition and its extension to continuous variational inequalities. *Journal of Optimization Theory and Applications* **149** 528–539.

Han, D., J. Jian, Q. Xu. 2010. A norm-relaxed method of quasi-multiplier-strongly sub-feasible direction for general constrained optimization with nonlinear equality and inequality constraints. In *Proceedings of the International Conference on Computer and Computational Intelligence*.

ISO-12213-3. 2006. *Natural gas - Calculation of compression factor - Part 3: Calculation using physical properties*. International Organization for Standardization.

Kim, N. H., J. Zhang, L. S. Lasdon. 1985. An improved successive linear programming algorithm. *Management Science* **31 (10)** 1312–1331.

Koch, T., B. Hiller, M. E. Pfetsch, L. Schewe. 2015. *Evaluating gas network capacities*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Philadelphia.

Kronqvist, J., D. E. Bernal, I. E. Grossmann. 2018. Using Regularization and Second Order Information in Outer Approximation for Convex MINLP. *Eprints for the optimization community* .

López-Cerdá, M. A. 2012. Stability in linear optimization and related topics. A personal tour. *TOP* **20** 217–244.

Martin, A., M. Möller, S. Moritz. 2006. Mixed integer models for the stationary case of gas network optimization. *Mathematical Programming* **105**, no. 2 563–582.

Misener, R., C. A. Floudas. 2009. Advances for the Pooling Problem: Modeling, Global Optimization, and Computational Studies. *Applied and Computational Mathematics* **8** 3–22.

Möller, M. 2004. *Mixed integer models for the optimisation of gas networks in the stationary case*. Ph.D. thesis, Darmstadt University of Technology.

Palacios-Gomez, F., L. Lasdon, M. Engquist. 1982. Nonlinear optimization by successive linear programming. *Management Science* **10** 1106–1120.

Pfetsch, M. E., A. Fügenschuh, B. Geissler, N. Geissler, R. Gollmer, B. Hiller, J. Humpola, T. Koch, T. Lehmann, A. Martin, A. Morsi, J. Rövekamp, L. Schewe, M. Schmidt, R. Schultz, R. Schwarz, J. Schweiger, C. Stangl, M. C. Steinbach, S. Vigerske, B. M. Willert. 2015. Validation of nominations in gas network optimization: Models, methods, and solutions. *Optimization Methods and Software* **30**, no. 1 15–53.

Pratt, K. F., J. G. Wilson. 1984. Optimisation of the operation of gas transmission systems. *Transactions of the Institute of Measurement and Control* **6**, no. 5 261–269.

Ríos-Mercado, R. Z., C. Borraz-Sánchez. 2015. Optimization problems in natural gas transportation systems: A state of the art review. *Applied Energy* **147**, no. 1 536–555.

Ríos-Mercado, R. Z., S. Kimb, E. A. Boyd. 2006. Efficient operation of natural gas transmission systems: A network-based heuristic for cyclic structures. *Computers & Operations Research* **33** 2323–2351.

Rockafellar, R. T., R. J.-B. Wets. 1991. Scenarios and Policy Aggregation in Optimization Under Uncertainty. *Mathematics of Operations Research* **16**, no. 1 119–147. doi:10.1287/moor.16.1.119.

Starling, K. E., J. L. Savidge. 1992. *Compressibility factors of natural gas and other related hydrocarbon gases*. Transmission Measurement Committee Report No. 8. American Gas Association, Virginia, 2nd edn.

Tawarmalani, M., N. V. Sahinidis. 2005. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming* **103** 225–249.

Wächter, A., L. T. Biegler. 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming* **106**, no. 1 25–57. ISSN 1436-4646. doi:10.1007/s10107-004-0559-y.

Weymouth, T. R. 1912. Problems in natural gas engineering. *Transactions of the American Society of Mechanical Engineers* **34**, no. 1349 185–231.

Wong, P. J., R. E. Larson. 1968. Optimization of tree-structured natural-gas transmission networks. *Journal of Mathematical Analysis and Applications* **24** 613–626.

A Proofs of auxiliary results in Section 4

A.1 Proof of Proposition 6

For each $\varepsilon > 0$, define K_ε as the set of ε-feasible points of K. By the continuity of all the g_i and h_j functions, the K_ε sets are closed.[22] Thus, by the compactness of K, they are also compact.

Let $t^F : \mathbb{R} \to \mathbb{R}$ be defined, for each $\varepsilon > 0$, by $t^F(\varepsilon) = \max_{\boldsymbol{x} \in K_\varepsilon} d(\boldsymbol{x}, F^{NL})$. The compactness of K_ε ensures that this function is well defined. Moreover, since $\varepsilon < \varepsilon'$ implies that $K_\varepsilon \subseteq K_{\varepsilon'}$, the function t^F is non-decreasing.

We now show that $t^F \in \mathcal{I}$, *i.e.*, $\lim_{\varepsilon \to 0} t^F(\varepsilon) = 0$. Suppose, on the contrary, that there is a sequence $\{\boldsymbol{x}^n\} \subset K$ of ε_n-feasible points with $\varepsilon_n \to 0$ and a real number $r > 0$ such that, for each $n \in \mathbb{N}$, $d(\boldsymbol{x}^n, F^{NL}) > r$.

By the compactness of K, there is a convergent subsequence $\{\boldsymbol{x}^{n_k}\}$ of $\{\boldsymbol{x}^n\}$ on K. The limit $\bar{\boldsymbol{x}}$ of $\boldsymbol{x}^{n_k}$ belongs to F^{NL} because it satisfies the constraints of the NLP problem:

$$g_i(\bar{\boldsymbol{x}}) = \lim_{n_k \to \infty} g_i(\boldsymbol{x}^{n_k}) \leq \lim_{n_k \to \infty} \varepsilon_{n_k} = 0,$$

$$0 \leq |h_j(\bar{\boldsymbol{x}})| = \lim_{n_k \to \infty} |h_j(\boldsymbol{x}^{n_k})| \leq \lim_{n_k \to \infty} \varepsilon_{n_k} = 0 \implies h_j(\bar{\boldsymbol{x}}) = 0,$$

which contradicts that $d(\bar{\boldsymbol{x}}, F^{NL}) = \lim_{n_k \to \infty} d(\boldsymbol{x}^{n_k}, F^{NL}) \geq r > 0$.

A.2 Proof of Theorem 7

We need some auxiliary notations and results. For each $I \subseteq I^m$ we define:

$$B_I^{NL} = \{\boldsymbol{x} \in F^{NL} : g_i(\boldsymbol{x}) = 0 \text{ for all } i \in I \text{ and } g_i(\boldsymbol{x}) \neq 0 \text{ for all } i \notin I\} = \{\boldsymbol{x} \in F^{NL} : I_{\boldsymbol{x}} = I\}.$$

Informally, the set B_I^{NL} contains the feasible points whose active constraints are only those belonging to I.

Remark 5. Note that $B_I^{NL} \cap B_J^{NL} = \emptyset$ if $I \neq J$, so thse sets form a partition of the feasible region:

$$F^{NL} = \bigcup_{I \subseteq I^m} B_I^{NL}.$$

[22] They are the intersections of finitely many inverse images of closed sets.

Moreover, when $I = I^m$ the set $B_I^{NL} = \left(\bigcap_{i\in I^m} g_i^{-1}(\{\mathbf{0}\})\right) \cap \left(\bigcap_{j\in J^l} h_j^{-1}(\{\mathbf{0}\})\right)$ is a compact set (possibly empty). On the other hand, if $I \subsetneq I^m$, then either $\mathrm{Cl}(B_I^{NL}) = B_I^{NL}$ and so B_I^{NL} is compact or:[23]

$$\mathrm{Cl}(B_I^{NL}) \setminus B_I^{NL} \subseteq \bigcup_{J\supsetneq I} B_J^{NL}. \tag{2}$$

Indeed, if $\boldsymbol{x} \in \mathrm{Cl}(B_I^{NL}) \setminus B_I^{NL}$, there exists a sequence $\{\boldsymbol{x}^n\} \in B_I^{NL}$ that converges to $\boldsymbol{x}$. Since $\boldsymbol{x} \notin B_I^{NL}$ and, for each $i \in I$, $g_i(\boldsymbol{x}) = \lim_{n\to\infty} g_i(\boldsymbol{x}^n) = 0$, so $i \in I_{\boldsymbol{x}}$ and we have $I \subsetneq I_{\boldsymbol{x}}$. Thus, $\boldsymbol{x} \in B_J^{NL}$ for $J = I_{\boldsymbol{x}} \supsetneq I$. ◁

Finally, given a set of inequality constraints $I \subsetneq I^m$ and a feasible point $\boldsymbol{x} \in F^{NL}$, we define a function that computes from those inequality constraints not belonging to set I, the minimum distance about how far they are to be active constraints at point $\boldsymbol{x}$. Formally, $g^I_{\min} : F^{NL} \to \mathbb{R}$ is the function defined, for each $\boldsymbol{x} \in F^{NL}$, by:

$$g^I_{\min}(\boldsymbol{x}) = \frac{1}{2}\min\{|g_i(\boldsymbol{x})| : i \notin I\}.$$

For the sake of completeness, for I^m, we define $g^{I^m}_{\min} : F^{NL} \to \mathbb{R}$ as the constant function with value $\max\{|g_i(\boldsymbol{x})| : i \in I^m,\ \boldsymbol{x} \in F^{NL}\}$. Clearly, the continuity of the g_i functions implies the continuity of the $g^I_{\min}(\boldsymbol{x})$ ones.

Remark 6. Since all the g_i functions are uniformly continuous functions on K, there is a non-decreasing function $t^U \in \mathcal{I}$ such that, for each $\gamma > 0$, if $\boldsymbol{x}, \boldsymbol{y} \in K$ with $\|\boldsymbol{x} - \boldsymbol{y}\| \leq t^U(\gamma)$, then $|g_i(\boldsymbol{x}) - g_i(\boldsymbol{y})| \leq \gamma$, for all $i \in I^m$. ◁

Lemma 11. *Take $t^U \in \mathcal{I}$ as in Remark 6 and let $\boldsymbol{x} \in F^{NL}$. Then, we have that, for each $0 < \gamma \leq g^{I_{\boldsymbol{x}}}_{\min}(\boldsymbol{x})$ and each $\boldsymbol{y} \in K$ such that $\|\boldsymbol{x} - \boldsymbol{y}\| < t^U(\gamma)$ the following conditions hold:*

i) $|g_i(\boldsymbol{y})| \leq \gamma$ *for each* $i \in I_{\boldsymbol{x}}$,

ii) $|g_i(\boldsymbol{y})| \geq \gamma$ *for each* $i \notin I_{\boldsymbol{x}}$.

Proof. Let $\boldsymbol{x} \in F^{NL}$. By Remark 6, for each $0 < \gamma \leq g^{I_{\boldsymbol{x}}}_{\min}(\boldsymbol{x})$, if $\|\boldsymbol{x} - \boldsymbol{y}\| \leq t^U(\gamma)$ then $|g_i(\boldsymbol{x}) - g_i(\boldsymbol{y})| \leq \gamma$ for all $i \in I^m$. Now, if $i \in I_{\boldsymbol{x}}$, $g_i(\boldsymbol{x}) = 0$, so statement i) is proved. Otherwise, if $i \notin I_{\boldsymbol{x}}$, we have $\big||g_i(\boldsymbol{x})| - |g_i(\boldsymbol{y})|\big| \leq |g_i(\boldsymbol{x}) - g_i(\boldsymbol{y})| \leq \gamma$. Then,

$$|g_i(\boldsymbol{y})| \geq |g_i(\boldsymbol{x})| - \gamma \geq 2g^{I_{\boldsymbol{x}}}_{\min}(\boldsymbol{x}) - \gamma \geq \gamma. \qquad \square$$

The next lemma provides the key ingredients for the proof of Theorem 7.

Lemma 12. *Take $t^U \in \mathcal{I}$ as in Remark 6. For each $\gamma > 0$, there are $\underline{\gamma}$ and $\overline{\gamma}$, with $0 < \underline{\gamma} \leq \overline{\gamma} \leq \gamma$, with the following property: for each $\boldsymbol{x} \in K \setminus F^{NL}$ with $d(\boldsymbol{x}, F^{NL}) \leq t^U(\underline{\gamma})$, there is $\overline{\boldsymbol{x}} \in F^{NL}$ such that:*

i) $\|\boldsymbol{x} - \overline{\boldsymbol{x}}\| \leq t^U(\overline{\gamma})$,

ii) $|g_i(\boldsymbol{x})| \leq \overline{\gamma}$ *for each* $i \in I_{\overline{\boldsymbol{x}}}$, *and*

iii) $|g_i(\boldsymbol{x})| \geq \underline{\gamma}$ *for each* $i \notin I_{\overline{\boldsymbol{x}}}$.

Proof. Given $\gamma > 0$, using an inductive process we define, for each $I \subseteq I^m$ with $B_I^{NL} \neq \emptyset$, a real number $\gamma_I \leq \gamma$ and a set $C_I \subseteq \mathrm{Cl}\,(B_I^{NL})$ as follows:

[23] $\mathrm{Cl}(S)$ denotes the topological closure of set S.

- *Case 1.* $I = I^m$ and $B_I^{NL} \neq \emptyset$. Define $\gamma_I = \gamma$ and $C_I = B_I^{NL}$.
- *Case 2.* $I \neq I^m$ and $B_I^{NL} \neq \emptyset$. Suppose that γ_J is defined whenever $|J| > |I|$. Define:

$$C_I = \mathrm{Cl}\,(B_I^{NL}) \setminus \bigcup_{J \supsetneq I} \Big(\bigcup_{\boldsymbol{z} \in \mathrm{Cl}(B_I^{NL}) \cap B_J^{NL}} B\big(\boldsymbol{z}, t^U(\gamma_J)\big) \Big),$$

  which is a compact set. Furthermore note that, when B_I^{NL} is compact, then $C_I = B_I^{NL}$. By the compactness of C_I we can define:

$$\gamma_I = \min \Big\{ \{ g_{\min}^I(\boldsymbol{z}) : \boldsymbol{z} \in C_I \}, \{ \gamma_J : |J| > |I| \} \Big\} > 0.$$

  It is important to remark that $C_I \subseteq F^{NL}$. Indeed, given $\boldsymbol{x} \in C_I$, if $\boldsymbol{x} \in B_I^{NL}$ it is trivial because $B_I^{NL} \subseteq F^{NL}$ by definition. Otherwise, since $C_I \subseteq \mathrm{Cl}\,(B_I^{NL})$, $\boldsymbol{x} \in \mathrm{Cl}\,(B_I^{NL}) \setminus B_I^{NL}$, and we know by Equation (2) that $\boldsymbol{x} \in B_{I_{\boldsymbol{x}}}^{NL}$ with $I_{\boldsymbol{x}} \supsetneq I$, so $\boldsymbol{x} \in F^{NL}$.

Informally, the set C_I contains the points in the closure of B_I^{NL} that are not too close to a point in F^{NL} with more active constraints than those in I. This ensures that, when restricted to C_I, the function $g_{\min}^I$ is bounded away from zero. Now, let $H = \{ I \subseteq I^m : B_I^{NL} \neq \emptyset \}$. We define:

$$\underline{\gamma} = \min_{I \in H} \{\gamma_I\} \quad \text{and} \quad \overline{\gamma} = \max_{I \in H} \{\gamma_I\}.$$

Note that, for each $\boldsymbol{y} \in K$ such that $d(\boldsymbol{y}, C_I) \leq t^U(\gamma_I)$, there exists $\boldsymbol{x} \in C_I \subseteq F^{NL}$ with $I = I_{\boldsymbol{x}}$ and by Lemma 11 we have that $|g_i(\boldsymbol{y})| \leq \gamma_I$ for all $i \in I$ and $|g_i(\boldsymbol{y})| \geq \gamma_I$ for all $i \notin I$.

Finally, we revise statements i), ii), and iii). Take $\boldsymbol{x} \in K \setminus F^{NL}$ with $d(\boldsymbol{x}, F^{NL}) \leq t^U(\underline{\gamma})$. Let $J \subseteq I^m$ the largest subset of indices such that $d(\boldsymbol{x}, C_J) \leq t^U(\gamma_J)$. Then, there exists $\bar{\boldsymbol{x}} \in C_J \subseteq F^{NL}$ such that:

i) $\|\boldsymbol{x} - \bar{\boldsymbol{x}}\| \leq t^U(\gamma_J) \leq t^U(\overline{\gamma})$.

Now, by definition of γ_J and by Lemma 11 taking into account that $J = I_{\bar{\boldsymbol{x}}}$:

ii) $|g_i(\boldsymbol{x})| \leq \gamma_J \leq \overline{\gamma}$ for all $i \in J$

iii) $|g_i(\boldsymbol{x})| \geq \gamma_J \geq \underline{\gamma}$ for all $i \notin J$. □

We are finally equipped to prove Theorem 7.

Proof of Theorem 7. First, take t^F as in the statement of Proposition 6 and t^U as in Remark 6. Lemma 12 associates, to each $\gamma > 0$, real numbers $\underline{\gamma}$ and $\overline{\gamma}$ with $0 < \underline{\gamma} \leq \overline{\gamma} \leq \gamma$.[24] Clearly, we can find $\varepsilon > 0$ small enough so that there is $\gamma > 0$ satisfying that $\varepsilon < \underline{\gamma}$ and $t^F(\varepsilon) < t^U(\underline{\gamma})$. Moreover, the monotonicity of t^F ensures that the same γ would work for all $\varepsilon' < \varepsilon$. Then, define $\bar{\varepsilon}$ as:

$$\bar{\varepsilon} = \sup\{\varepsilon > 0 : \text{there is } \gamma > 0 \text{ satisfying that } \varepsilon < \underline{\gamma} \text{ and } t^F(\varepsilon) < t^U(\underline{\gamma})\}.$$

Now, for each $\varepsilon > 0$, with $\varepsilon < \bar{\varepsilon}$, define:

$$\gamma^\varepsilon = \inf\{\gamma : \varepsilon < \underline{\gamma} \text{ and } t^F(\varepsilon) < t^U(\underline{\gamma})\},$$

and let $\hat{t} \in \mathcal{I}$ be defined, for each $\varepsilon < \bar{\varepsilon}$, by $\hat{t}(\varepsilon) = t^U(\overline{\gamma^\varepsilon})$.

Finally, let $\boldsymbol{x}$ be an ε-feasible point, with $\varepsilon < \bar{\varepsilon}$. By Proposition 6, we have that $d(\boldsymbol{x}, F^{NL}) \leq t^F(\varepsilon) < t^U(\underline{\gamma^\varepsilon})$. Then, take $\bar{\boldsymbol{x}} \in F^{NL}$ with the properties given in Lemma 12. We conclude by showing that $\bar{\boldsymbol{x}}$ satisfies a) and b).

[24] With some abuse of notation, in this proof we think of $\underline{\gamma}$ and $\overline{\gamma}$ as functions of γ.

a) By statement i) in Lemma 12, $\|\boldsymbol{x} - \bar{\boldsymbol{x}}\| \leq t^U(\overline{\gamma^\varepsilon}) = \hat{t}(\varepsilon)$, so a) is satisfied.

b) By statement iii) in Lemma 12, $|g_i(\boldsymbol{x})| \geq \underline{\gamma}^\varepsilon$ for each $i \notin I_{\bar{\boldsymbol{x}}}$. Then, since for each $i \in I_{\boldsymbol{x}}^\varepsilon$ we have that $|g_i(\boldsymbol{x})| \leq \varepsilon < \underline{\gamma}^\varepsilon$, we deduce that $I_{\boldsymbol{x}}^\varepsilon \subseteq I_{\bar{\boldsymbol{x}}}$. $\square$