Computers and Chemical Engineering 101 (2017) 243–258

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

Constrained NLP via gradient flow penalty continuation: Towards self-tuning robust penalty schemes

Felipe Scott[a], Raúl Conejeros[b], Vassilios S. Vassiliadis[c,*]

[a] Green Technology Research Group, Facultad de Ingeniería y Ciencias Aplicadas, Universidad de los Andes, Chile, Mons. Álvaro del Portillo 12455, Las Condes, Santiago 7620001, Chile
[b] School of Biochemical Engineering, Pontificia Universidad Católica de Valparaíso, Av. Brasil 2085, Valparaíso, Chile
[c] Department of Chemical Engineering and Biotechnology, University of Cambridge, Cambridge CB2 3RA, UK

ARTICLE INFO

Article history:
Received 28 July 2016
Received in revised form 16 January 2017
Accepted 18 January 2017
Available online 3 March 2017

Keywords:
Gradient flow
Nonlinear programming problem
Convergence analysis

ABSTRACT

This work presents a new numerical solution approach to nonlinear constrained optimization problems based on a gradient flow reformulation. The proposed solution schemes use self-tuning penalty parameters where the updating of the penalty parameter is directly embedded in the system of ODEs used in the reformulation, and its growth rate is linked to the violation of the constraints and variable bounds. The convergence properties of these schemes are analyzed, and it is shown that they converge to a local minimum asymptotically. Numerical experiments using a set of test problems, ranging from a few to several hundred variables, show that the proposed schemes are robust and converge to feasible points and local minima. Moreover, results suggest that the GF formulations were able to find the optimal solution to problems where conventional NLP solvers fail, and in less integration steps and time compared to a previously reported GF formulation.

1. Introduction

Optimization problems arise in many areas of chemical engineering practice, from component and systems design (Grossmann, 2012) to operation and control (Amrit et al., 2013). Due to an increasing concern of legislators and the general public in environmental sustainability, optimization has been recently used to aid in the design of supply chains and products considering their life cycle (Yue et al., 2013), and as a tool for the design of new sustainable energy conversion systems (Baliban et al., 2013; Scott et al., 2013). Applications encompass formulations ranging from linear programming problems (LP) to mixed-integer non-linear programming problems (MINLP) and dynamic optimization problems (optimal control problems, OCP). A common feature of many of these classes of problems, is that at a certain point one or several non-linear constrained programming problems (NLP) need to be solved. The solution of large-scale NLP problems was made possible by breakthroughs in non-linear programming during the previous decades. In particular, the development of modern barrier methods (Byrd et al., 1999; Vanderbei and Shanno, 1999; Wächter and Biegler, 2006), sequential quadratic programming (Gill et al., 2005) and reduced gradient methods (Drud, 1994), led to implementations (solvers) such as IPOPT (Wächter and Biegler, 2006), SNOPT (Gill et al., 2005) and CONOPT (Drud, 1994) that can be used in user-friendly modeling and optimization environments such as GAMS (Bussieck and Meeraus, 2004), AMPL (Fourer et al., 2002) and AIMMS (Bisschop and Roelofs, 2006).

Most algorithms used to compute a local optimum of constrained NLP problems rely on Taylor series expansions truncated after the linear or quadratic term; according to this, constraints are linearized and large steps towards the local minimum are allowed. For this reason, in highly nonlinear problems intermediate iterations might prove infeasible and frequent failures to converge to a local optimum may arise. Alternatively to the Taylor expansion based methods, gradient flow (GF) methods have been proposed for the solution of unconstrained and constrained nonlinear programming problems. In its most simple version, the solution of an unconstrained problem $\min_x f(x)$ can be obtained by solving the following set of coupled ordinary differential equations (ODEs):

$$\frac{dx}{dt} = -\nabla_x f(x); \quad x(0) = x_0 \tag{1}$$

where $x \in \mathbb{R}^n, f(x) : \mathbb{R}^n \mapsto \mathbb{R}^1$. This approach creates a smooth trajectory that might offer an advantage for highly nonlinear problems compared with the conventional optimization techniques which take finite steps along line-search directions. For the latter, finding

* Corresponding author.
E-mail address: vsv20@cam.ac.uk (V.S. Vassiliadis).

http://dx.doi.org/10.1016/j.compchemeng.2017.01.034
0098-1354/

a suitable step-size can be difficult when the optimization function is non-quadratic and has large third derivatives, resulting in a slow progress towards the solution due to the smalls steps required (Brown and Bartholomew-Biggs, 1989a).

Another interesting feature of GF methods is the possibility of using state-of-the-art integration software to find the solution of optimization problems. This approximation for the solution of unconstrained problems can be traced to the work of Botsaris (1978). In the following decades, efforts were made to reach a competitive level in terms of computational time and iterations compared to conventional methods, with a summary found in Brown and Bartholomew-Biggs (1989a). The application of GF methods was further extended by introducing new formulations that were able to cope with constrained nonlinear problems (Brown and Bartholomew-Biggs, 1989b; Evtushenko and Zhadan, 1994; Smirnov, 1994; Orsi, 1999). The constraints of the NLP problem ($h(x)$) are incorporated to the objective function ($f(x)$) with a penalty scheme in order for GF methods to be employed, with one of the major issues being the updating of the penalty parameters utilized. For an optimization problem with equality constraints only, Tanabe (1974) proposed the following Gradient Flow formulation:

$$\frac{dx}{dt} = -Q(x)\nabla_x f(x); \quad x(0) = x_0 \tag{2}$$

$$Q(x) = [I - J^T(x)(J(x)J^T(x))^{-1}J(x)]$$

where $\nabla_x f(x)$ and $J(x)$ represents the gradient of the objective function with respect to the optimization variables and the Jacobian matrix, respectively. Eq. (2) is a direct generalization of the gradient projection method proposed by Rosen (1961) to a differential form, which is based on projecting the search direction given by $\nabla_x f(x)$ into the subspace tangent to the active constraints. The method proposed by Tanabe (1974) was further modified by Yamashita (1980) and Evtushenko and Zhadan (1994) to yield

$$\frac{dx}{dt} = -sQ(x)\nabla_x f(x) - J(x)(J(x)J^T(x))^{-1}h(x); \quad x(0) = x_0 \tag{3}$$

with s a positive constant. Following the work of Evtushenko and Zhadan (1994), Wang et al. (2003) proposed an approach to include inequality constraints and bounds where a pseudo-inverse of the square matrix $J(x)J^T(x)$ acts as a penalizer (Eq. (4)), with this approach requiring a non-singular Jacobian:

$$\frac{dx}{dt} = -[\nabla f(x) + J^T(x)(\tau h(x) - (J(x)J^T(x))^{-1}J(x)\nabla f(x))]; \quad x(0) = x_0 \tag{4}$$

In their work, they avoid the use of slacks to account for variable bounds by using the state-space transformation technique proposed by Evtushenko and Zhadan (1994), otherwise the use of quadratic slacks would result in singular Jacobians. As proved by the above mentioned authors, their GF formulations have the property that once the equality constraints are satisfied, the trajectory of the solution will stay on the manifold determined by the constraints. However, as analyzed by Brown and Bartholomew-Biggs (1989b), the ODE system that allows following a path with those characteristics needs to be solved quite accurately. This and the fact that inverses of large matrices need to be calculated, produce a heavy numerical overhead. Moreover, in the formulations represented by Eqs. (2)–(4), authors do not present an approach to select the values of the parameters required in their formulations (such as τ in Eq. (4)). In practice, the value of these parameters need to be adjusted to each problem. Finally, Schropp and Singer (2000) compare SQP methods and GF methods for the solution of nonlinear problems from a theoretical point of view and using two case studies. They concluded that SQP methods are efficient tools whereas the ODE approach may be more reliable, with the ODE approach being more efficient for problems with only a small number of highly nonlinear constraints. Moreover, they propose an approach combining differential and algebraic equations that, according to the authors, combines efficiency and reliability.

In this work, a self-tuning penalty scheme is presented for the solution of constrained NLPs. The approach does not require the calculation of an inverse (or pseudo-inverse) of the Jacobian matrix, and the penalty parameters updating is directly embedded into the system of ODEs. The performance of the GF formulations presented in this work are compared to the formulation presented by Wang et al. (2003) and also against several state-of-the-art NLP solvers.

2. New formulations using the gradient flow approach for solving NLP problems

2.1. Problem definition

The minimization of the following standard constrained NLP is considered:

$$\begin{aligned} &\min_{x_s} f(x_s) \\ &\text{subject to :} \\ &h_s(x_s) = 0 \\ &g(x_s) \le 0 \\ &x_s^L \le x_s \le x_s^U \end{aligned} \tag{5}$$

where $x_s \in \mathbb{R}^{n_s}$, $f(x_s) : \mathbb{R}^{n_s} \mapsto \mathbb{R}^1$, $h_s(x_s) : \mathbb{R}^{n_s} \mapsto \mathbb{R}^{m_1}$ and $g(x_s) : \mathbb{R}^{n_s} \mapsto \mathbb{R}^{m_2}$. The subscript s stands for standard, as this problem will be converted to a penalized version were the subscripts will be dropped to simplify the notation. This problem is converted to a penalty function minimization, using a quadratic penalty scheme and standard transformations. Inequality constraints are converted to equalities via the use of squared slack variables as follows. First, inequality constraints are converted to equality constraints using the following transformation:

$$g(x_s) + w^2 = 0 \tag{6}$$

where

$$w \in \mathbb{R}^{m_2}$$

.

Variables bounds are transformed to equalities, by using the following equations:

$$x_s + \left(s^U\right)^2 = x_s^U \tag{7}$$

$$x_s - \left(s^L\right)^2 = x_s^L \tag{8}$$

where $s^L, s^U \in \mathbb{R}^{n_s}$. The equality constraints defined by Eqs. (6)–(8) and the original constraints, $h_s(x)$, will be appended in the vector $h(x) : \mathbb{R}^n \mapsto \mathbb{R}^m$ with $n = 3n_s + m_2$, $m = 2n_s + m_1 + m_2$ and $x = \{x_s, s^U, s^L, w\}$. Using this new defined set of constraints and variables, the original problem posed in Eq. (5) can be redefined as the following (higher dimensionality) optimization problem:

$$\begin{aligned} &\min_{x} f(x) \\ &\text{subject to :} \\ &h(x) = 0 \end{aligned} \tag{9}$$

with the Lagrangian of the problem defined by:

$$L(x, \mu) = f(x) + \lambda^T h(x) \tag{10}$$