PENALTY METHODS FOR THE SOLUTION OF GENERALIZED NASH EQUILIBRIUM PROBLEMS (WITH COMPLETE TEST PROBLEMS)

Francisco Facchinei[1] and Christian Kanzow[2]

[1] Sapienza University of Rome
Department of Computer and System Sciences “A. Ruberti”
Via Ariosto 25, 00185 Roma, Italy
e-mail: facchinei@dis.uniroma1.it

[2] University of Würzburg
Institute of Mathematics
Am Hubland, 97074 Würzburg, Germany
e-mail: kanzow@mathematik.uni-wuerzburg.de

February 11, 2009

Abstract

The generalized Nash equilibrium problem (GNEP) is an extension of the classical Nash equilibrium problem where both the objective functions *and* the constraints of each player may depend on the rivals' strategies. This class of problems has a multitude of important engineering applications and yet solution algorithms are extremely scarce. In this paper, we analyze in detail a globally convergent penalty method that has favorable theoretical properties. We also consider strengthened results for a particular subclass of problems very often considered in the literature. Basically our method reduces the GNEP to a single penalized (and nonsmooth) Nash equilibrium problem. We suggest a suitable method for the solution of the latter penalized problem and present extensive numerical results.

Key Words: Nash equilibrium problem, Generalized Nash equilibrium problem, Jointly convex problem, Exact penalty function, Global convergence.

1 Introduction

In this paper, we consider the Generalized Nash Equilibrium Problem (GNEP) and describe new globally convergent methods for its solution. The GNEP is an extension of the classical Nash Equilibrium Problem (NEP) where the feasible sets of each player may depend on the rivals' strategies. We assume there are N players and denote by $x^\nu \in \mathbb{R}^{n_\nu}$ the vector representing the ν-th player's strategy. We write

$$\mathbf{x} := \begin{pmatrix} x^1 \\ \vdots \\ x^N \end{pmatrix}, \qquad \mathbf{x}^{-\nu} := \begin{pmatrix} x^1 \\ \vdots \\ x^{\nu-1} \\ x^{\nu+1} \\ \vdots \\ x^N \end{pmatrix}$$

to denote the vector of all players' strategies and the vector of all players's strategies except that of player ν. With this terminology in mind, we also write $\mathbb{R}^n \ni \mathbf{x} = (x^\nu, \mathbf{x}^{-\nu})$, where $n := n_1 + \cdots + n_N$.

The aim of player ν, given the other players' strategies $\mathbf{x}^{-\nu}$, is to choose a vector x^ν that solves the minimization problem

$$\begin{aligned} &\text{minimize}_{x^\nu} && \theta_\nu(x^\nu, \mathbf{x}^{-\nu}) \\ &\text{subject to} && x^\nu \in X_\nu(\mathbf{x}^{-\nu}). \end{aligned} \tag{1}$$

Throughout this paper, we make the blanket assumption that the objective functions $\theta_\nu : \mathbb{R}^n \to \mathbb{R}$ are continuously differentiable and, as a function of x^ν alone, convex, while the point-to-set mapping $X_\nu : \mathbb{R}^{n-n_\nu} \rightrightarrows \mathbb{R}^{n_\nu}$ is assumed to be convex valued. A point $\bar{\mathbf{x}}$ is feasible for the GNEP if $\bar{x}^\nu \in X_\nu(\bar{\mathbf{x}}^{-\nu})$ for all players ν. A generalized Nash equilibrium, or simply a solution of the GNEP, is a feasible point $\bar{\mathbf{x}}$ such that

$$\theta_\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu}) \leq \theta_\nu(x^\nu, \bar{\mathbf{x}}^{-\nu}), \qquad \forall x^\nu \in X_\nu(\bar{\mathbf{x}}^{-\nu}) \tag{2}$$

holds for each player $\nu = 1, \ldots, N$. If the feasible sets $X_\nu(\mathbf{x}^{-\nu})$ do not actually depend on the rivals' strategies $\mathbf{x}^{-\nu}$ and are therefore constant sets, we have the classical NEP. There is a further case which has been often considered in the literature and that to some extent can be seen as an intermediate case between the NEP and the GNEP. This intermediate case is known as the *jointly convex* GNEP and it arises when the feasible sets $X_\nu(\mathbf{x}^{-\nu})$ are given by

$$X_\nu(\mathbf{x}^{-\nu}) := \{x^\nu \in \mathbb{R}^{n_\nu} : (x^\nu, \mathbf{x}^{-\nu}) \in \mathbf{X}, \text{ for some suitable } \mathbf{x}^{-\nu}\},$$

where $\mathbf{X} \subseteq \mathbb{R}^n$ is a non-empty closed convex set.

In this paper, we shall consider the most common case where the feasible sets $X_\nu(\mathbf{x}^{-\nu})$ are defined by parametric inequalities. More specifically, let $g^\nu : \mathbb{R}^n \to \mathbb{R}^{m_\nu}$ be given; we assume

$$X_\nu(\mathbf{x}^{-\nu}) := \left\{x^\nu \in \mathbb{R}^{n_\nu} \,|\, g^\nu(x^\nu, \mathbf{x}^{-\nu}) \leq 0\right\}, \tag{3}$$

where each g_i^ν, $i = 1, \ldots, m_\nu$, is continuously differentiable and, as a function of x^ν only, convex. Usually, the constraints defining the set $X_\nu(\mathbf{x}^{-\nu})$ are divided into two groups: those that actually depend on x^ν only and those that depend also on the other players' variables $\mathbf{x}^{-\nu}$. From a practical point of view, it might be useful to distinguish between these two groups of constraints; however, we avoid this distinction for the sake of notational simplicity, since their consideration would not add much to our theoretical analysis, see, however, Remark 2.9. When such practical setting is considered, it is easy to see that a problem is jointly convex if $g^1 = g^2 = \cdots = g^N =: g$ and g is convex with respect to the whole vector $\mathbf{x}$.

The GNEP was formally introduced by Debreu [7] as early as 1952, it lies at the heart of modern mathematical economics and, as such, its importance cannot be underestimated. However, it is only from the mid-1990s that interesting engineering applications started to arise along with the real need for the computation of their solutions. These applications range from structural engineering to telecommunications, from computer science to the analysis of deregulated markets and of pollution scenarios. We refer the interested reader to [13] for a detailed history of the GNEP and of its many applications.

While the role of the GNEP as an important modelling tool is well established, the situation regarding solution algorithms is rather bleak. Rosen [43] popularized the jointly convex GNEP and this setting has dominated the literature until very recently. He proposed the first algorithm (a projected gradient-type approach) for the solution of jointly convex GNEPs, and some other algorithms were later developed for this class of problems; prominent among these is the *relaxation method* [31, 44], relying on the Nikaido-Isoda function. Other approaches are possible to the solution of jointly convex problems; among them we may cite the recently proposed methods from [24, 25, 26, 27], which are still based on the Nikaido-Isoda function and therefore computationally quite intensive, or the variational inequality approaches from [11, 34] which are also restricted to the jointly convex case. There also exist a number of other proposals, and we refer the interested reader to the survey paper [13] and the references therein.

If we consider the general GNEP, the situation becomes still more complex. It is well-known [3, 13, 23] that a GNEP can be reduced to a Quasi Variational Inequality (QVI). However, since the development of globally convergent algorithms for the solution of a QVI is, in turn, still a challenging field, this reduction is of little algorithmic use, even if one may conceptually use some gap function to reduce the GNEP to an optimization problem. Reduction to an optimization problem can also be achieved through the use of the Nikaido-Isoda function, but the computational overhead is high and the conditions for establishing convergence are not easy to understand. Other approaches are also possible, and we again refer the reader to [13] for a more detailed survey of existing possibilities. However, it is safe to say that with the exception of penalty algorithms, that we discuss below, and algorithms for very specific applications, see for example [39, 40, 41], the study of algorithms (and, especially, of their global convergence) for general GNEPs is still in its infancy.

Penalty approaches to the solution of GNEPs are based on the usual penalization idea: eliminate the difficult "coupling" constraints in a GNEP thus reducing it to a somewhat simpler NEP. To this latter problem, we can then apply a host of methods based mainly on the optimization or variational inequalities techniques, see [15], although conditions under which the algorithms for VIs are guaranteed to work for NEPs are not well discussed so far. Nevertheless, the VI formulation provides a strong theoretical and numerical framework for the solution of the classical NEP. The penalization approach to GNEPs was initiated very recently by Fukushima and Pang in [21], where they proposed a sequential penalty approach to the solution of GNEPs in which an infinite sequence of differentiable penalized problems is solved. The idea of using an exact penalty approach whereby a single nondifferentiable NEP has to be solved to obtain a solution was first put forward in [16] even if in a rather sketchy way. This topic has recently been considered also by Fukushima [20] who, among other things, gives some conditions under which a penalty approach can be used to find a solution of a GNEP. Although many issues have still to be fully addressed (among them the practical solution of subproblems and a sound understanding of the theoretical conditions under which useful results can be established for a penalty approach), we believe that penalty based methods are a very promising approach to the solution of GNEPs and currently the *only* practical class of methods available for their solution.

This paper is a contribution to this line of research. One of the main difficulties in an (exact) penalization approach is how to properly choose the penalty parameters. In this respect, we provide a very simple algorithm that automatically settles the penalty parameters to the "right" values. This approach relies heavily on the fact that, unlike in most approaches, we use the γ-norm with some $\gamma \in (1, \infty)$ (see below) to define the penalization. We analyze in detail conditions which guarantee that the penalty parameters will settle to a finite value after a finite number of iterations thus guaranteeing convergence to a solution of the GNEP. We also consider in detail the jointly convex case for which stronger results can be obtained. We finally provide an extensive numerical testing to show that the approach is viable in practice.

Throughout this paper, for some vector $\mathbf{x}$ of appropriate dimension,

$$\|\mathbf{x}\|_\gamma := \left(\sum_i |x_i|^\gamma \right)^{1/\gamma} \tag{4}$$

denotes the γ-norm for some fixed $\gamma \in [1, \infty]$ (as we will see, in this paper we will use mainly γ-norms with $\gamma \in (1, \infty)$). For the Euclidean norm we use no subscript and simply write $\|\mathbf{x}\|$. We further write $g_+(\mathbf{x}) := \max\{0, g(\mathbf{x})\}$ for a given function $g : \mathbb{R}^n \to \mathbb{R}^m$, with the maximum taken componentwise, $\nabla g(\mathbf{x})$ denotes the transposed Jacobian of this mapping g (provided that g is differentiable), and $\nabla_{x^\nu} g(\mathbf{x})$ is the partial transposed Jacobian of g at $\mathbf{x}$, where the partial derivatives are taken only with respect to the components i belonging to the subvector x^ν of $\mathbf{x}$. When no confusion is possible, we will also write $(v^1, \dots, v^N)$ instead of $((v^1)^T, \dots, (v^N)^T)^T$.

2 Exact Penalization Algorithm

In this section, we present an algorithmic scheme based on exact penalization for general GNEPs of the form (1) where the constraints are defined as in (3):

$$\begin{array}{ll} \min_{x^\nu} & \theta_\nu(x^\nu, \mathbf{x}^{-\nu}) \\ & g^\nu(x^\nu, \mathbf{x}^{-\nu}) \leq 0. \end{array} \tag{5}$$

Our aim is to transform the GNEP problem into a(n unconstrained) nondifferentiable Nash problem by using a penalty approach. To this end, we consider a penalization of the GNEP where each player's minimization problem is given by

$$\text{minimize}_{x^\nu} \; \theta_\nu(x^\nu, \mathbf{x}^{-\nu}) + \rho_\nu \, \|g^\nu_+(x^\nu, \mathbf{x}^{-\nu})\|_\gamma, \tag{6}$$

where the ρ_ν are positive penalty parameters and $\|\cdot\|_\gamma$ denotes the γ-norm (4) for some fixed $\gamma \in (1, \infty)$. The choice of this norm in order to define a penalty problem is somewhat unusual. Traditionally, the norms $\|\cdot\|_1$ and $\|\cdot\|_\infty$ have been preferred. However, the γ-norm taken here is differentiable at any infeasible point. We believe that this is a clear advantage and will put this peculiarity to fruitful use. By setting

$$P_\nu(\mathbf{x}, \rho_\nu) := \theta_\nu(x^\nu, \mathbf{x}^{-\nu}) + \rho_\nu \, \|g_+^\nu(x^\nu, \mathbf{x}^{-\nu})\|_\gamma,$$

problem (6) can be rewritten as a (standard) Nash equilibrium problem, where each player's problem is given by

$$\min_{x^\nu} P_\nu(\mathbf{x}, \rho_\nu). \tag{7}$$

We refer to problem (6) or, equivalently, (7) as the Penalized Nash Equilibrium Problem (PNEP for short). Note that, for every fixed $\mathbf{x}^{-\nu}$, the ν-th player's subproblem in (6) is convex, although nondifferentiable. In principle, the PNEP is easier than the original generalized Nash problem (1), even if the nondifferentiability of the players' objective functions is somewhat problematic from the numerical point of view.

To make this idea viable in practice, we must tackle two main issues: (a) (How) can we choose the penalty parameters ρ_ν so that a solution of the PNEP is also a solution of the original GNEP? (b) How can we solve the PNEP for a given choice of penalty parameters? In order to make the analysis more general and also simpler, we discuss these two issues separately. This section is devoted to the analysis of (a), while we shall discuss (b) in Section 3. Therefore, in what follows, we will simply assume momentarily that we have an algorithm that solves the PNEP for given *fixed* values of the penalty parameters. To be more precise, we suppose an iterative algorithm $\mathcal{A}$ is available that, given a point $\mathbf{x}^k$, generates a new point $\mathbf{x}^{k+1} := \mathcal{A}[\mathbf{x}^k]$. We make the following absolutely natural blanket assumption on $\mathcal{A}$.

Assumption 2.1 For every $\mathbf{x}^0$, the sequence $\{\mathbf{x}^k\}$ obtained by setting $\mathbf{x}^{k+1} = \mathcal{A}[\mathbf{x}^k]$ is such that every limit point (if any) is a solution of the PNEP.

Note that all the results of this section hold independently of the algorithm $\mathcal{A}$ so that, on the one hand, we have total freedom in the choice of $\mathcal{A}$ while, on the other hand, we need not worry about replicating results whenever the algorithm $\mathcal{A}$ changes. Based on the algorithm $\mathcal{A}$ we show next how we can iteratively update the values of the ρ_ν. The idea behind the scheme below is that if, for any value of the penalty parameters ρ_ν, we find a solution of (7) that is also feasible for (5), then this is easily seen to be a solution of (5). We therefore try to force feasibility by increasing the penalty parameters if this seems to be necessary. On the other hand, if a solution $\bar{\mathbf{x}}$ of (7) is not feasible for at least one player ν of the game (5), then the function $P_\nu(\mathbf{x}, \rho_\nu)$ is continuously differentiable at $\bar{\mathbf{x}}$ and this entails

$$\|\nabla_{x^\nu}\theta_\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu})\| = \rho_\nu \left\| \nabla_{x^\nu}\|g_+^\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu})\|_\gamma \right\|. \tag{8}$$

The idea of the updating scheme below is then to detect when this "dangerous" situation occurs (see the test at Step 2), and to increase the value of the penalty parameter in this case. Here is the detailed realization of this idea.

Algorithm 2.2 *(Penalty Updating Scheme for General GNEPs)*

Data: $\mathbf{x}^0 \in \mathbb{R}^n$, $\rho_\nu > 0$ *and* $c_\nu \in (0,1)$ *for all* $\nu = 1, \ldots, N$. *Set* $k := 0$.

Step 1: *If* $\mathbf{x}^k$ *is a solution of the GNEP* (1)*: STOP.*

Step 2: *Let* $I^k := \{\nu \,|\, (x^k)^\nu \notin X_\nu((\mathbf{x}^k)^{-\nu})\}$. *For every* $\nu \in I^k$, *if*

$$\|\nabla_{x^\nu}\theta_\nu((x^k)^\nu, (\mathbf{x}^k)^{-\nu})\| > c_\nu \left[\rho_\nu \left\| \nabla_{x^\nu}\|g_+^\nu((x^k)^\nu, (\mathbf{x}^k)^{-\nu})\|_\gamma \right\|\right], \tag{9}$$

then double the penalty parameters ρ_ν.

Step 3: *Compute* $\mathbf{x}^{k+1} = \mathcal{A}[\mathbf{x}^k]$, *set* $k \leftarrow k+1$, *and go to Step 1.*

Regarding Step 2 of Algorithm 2.2, note that, for all $\nu \in I^k$, the current iterate $\mathbf{x}^k$ is not feasible for the ν-th player's optimization problem. Hence $g_+^\nu(\mathbf{x}^k)$ is positive, and the γ-norm at this point is therefore differentiable. The rationale behind the test (9) is very simple: whenever the gradient of the penalty term is not sufficiently larger than the gradient of the objective function, we increase the penalty parameter. The effect of this increase is to correspondingly increase the norm of the penalty term and so avoid the possibility that (8) occurs.

Remark 2.3 Algorithm 2.2 is simple and it uses a very rough updating rule. We will discuss many possible variants, which are probably more algorithmically attractive, later on. □

Remark 2.4 Fukushima [20] proposed an updating scheme for the value of penalty parameters in a penalty approach to GNEPs. This is based, more traditionally, on the use of some suitable Lagrange multipliers to decide whether to update the penalty parameters. The scheme in [20] and our one are substantially distinct. One key difference that we believe is important from the practical point of view is that in [20], in order to decide whether to update the penalty parameters, a nondifferentiable penalized Nash problem has to be *solved exactly.* In our approach instead, a *single* iteration of a solution method has to be performed. Note that there is not even need that the penalized problem have a solution for all values of the penalty parameters for this to be accomplished. □

The following theorem gives a detailed picture of what we can expect from Algorithm 2.2. Although somewhat technical, the theorem moves along lines that have proven extremely useful in optimization and is the cornerstone on which we will elaborate to obtain more practical results. In what follows, we indicate by I_∞ the set of indices of players for which the updating test (9) is satisfied infinitely many times. Note that $I_\infty = \emptyset$ if and only if all the penalty parameters are updated a finite number of times only.

Theorem 2.5 *Let* $\{\mathbf{x}^k\}$ *be the sequence generated by Algorithm 2.2. If* $I_\infty = \emptyset$ *(i.e. if the penalty parameters are updated a finite number of times only), then every limit point* $\bar{\mathbf{x}}$ *of this sequence is a solution of the GNEP.*

If instead $I_\infty \neq \emptyset$ *(i.e. if some penalty parameters grow to infinity) and the sequence* $\{\mathbf{x}^k\}$ *is bounded, then, for each* $\nu \in I_\infty$, *there is a limit point* $\bar{\mathbf{x}}$ *for which one of the following assertions is true:*

(a) $\bar{x}^\nu$ *is a global minimizer of the constraint violation* $\|g_+^\nu(\cdot, \bar{\mathbf{x}}^{-\nu})\|_\gamma$ *with* $\|g_+^\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu})\|_\gamma > 0$;

(b) $\bar{x}^\nu$ *is Fritz John point for the player's problem (5) (when the rivals play* $\bar{\mathbf{x}}^{-\nu}$*), but not a solution of (5);*

(c) $\bar{x}^\nu$ *is an optimal solution for the player's problem (5) (when the rivals play* $\bar{\mathbf{x}}^{-\nu}$*).*

Before proving the theorem, let us comment on it. The first part of the theorem deals with the case in which the penalty parameters are updated a finite number of times only, that can be interpreted as "exact penalization is working". In this case, we have a neat result: every limit point of $\{\mathbf{x}^k\}$ is a solution of the GNEP. The second part, instead, deals with the case in which the penalty parameters blow-up to infinity, that is with the case in which "exact penalization is not working". We show that in this case the algorithm does as well as possible, in that one of the three possibilities (a), (b), or (c) must necessarily hold for some players. In nowadays standard optimization jargon introduced in [4], this can be restated by saying that $\bar{x}^\nu$ (for $\nu \in I_\infty$) is a *generalized stationary point.* At first sight, this may seem a weak result, but a moment thought immediately reveals that this result actually shows that Algorithm 2.2 achieves the maximum that can be expected. In fact, observe that we made no regularity assumptions whatsoever on problem (5) and, therefore, it could happen that, for a given $\bar{\mathbf{x}}^{-\nu}$, a player's optimization problem is infeasible (case (a)) or, if feasible, does not satisfy any constraint qualification, so that Fritz John points are present (case (b)). Obviously, the problem can also admit a solution (case (c)). In any event, the second part of the theorem shows that the algorithm finds "meaningful" results and that, roughly speaking, if it "does not work" it is because some problems of the players are lacking in feasibility or constraint qualification.

Proof of Theorem 2.5: We begin by showing that if the penalty parameters are updated a finite number of times only, then any limit point $\bar{\mathbf{x}}$ is a solution of the GNEP. Since the penalty parameters are updated a finite number of times only, after a finite number of steps, Algorithm 2.2 can be viewed as the application of Algorithm $\mathcal{A}$ to the solution of the PNEP (for fixed values of the penalty parameters ρ_ν). Therefore, by Assumption 2.1, every limit point $\bar{\mathbf{x}}$ of the sequence $\{\mathbf{x}^k\}$ is a Nash equilibrium of the PNEP. In order to show that $\bar{\mathbf{x}}$ is also a Nash equilibrium of the original GNEP, it is enough to show that $\bar{\mathbf{x}}$ is feasible. In fact, if $\bar{\mathbf{x}}$ is a Nash equilibrium of the PNEP, this means that $\bar{x}^\nu$ is the optimal unconstrained solution of the problem $\min_{x^\nu} P_\nu(x^\nu, \bar{\mathbf{x}}^{-\nu}, \rho_\nu)$. But, on the feasible set $X_\nu(\bar{\mathbf{x}}^{-\nu})$, we have $\theta_\nu(x^\nu, \bar{\mathbf{x}}^{-\nu}) = P_\nu(x^\nu, \bar{\mathbf{x}}^{-\nu}, \rho_\nu)$. Therefore, $\bar{\mathbf{x}}$ is also a solution of the GNEP.

Suppose then, by contradiction, that $\bar{\mathbf{x}}$ is not feasible. Subsequencing if necessary, we may assume that the entire sequence $\{\mathbf{x}^k\}$ converges to $\bar{\mathbf{x}}$. Suppose also, without loss of generality, that the penalty parameters are not updated for any k. Since $\bar{\mathbf{x}}$ is not feasible, the set $\bar{I} := \{\nu \,|\, \bar{x}^\nu \notin X_\nu(\bar{\mathbf{x}}^{-\nu})\}$ is nonempty. For each $\nu \in \bar{I}$, P_ν is continuously differentiable in a neighborhood of $\bar{\mathbf{x}}$ and, therefore, since $\bar{\mathbf{x}}$ is a Nash equilibrium of the unconstrained penalized problem (6), we have

$$0 = \nabla_{x^\nu} P_\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu}, \rho_\nu) = \nabla_{x^\nu} \theta_\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu}) + \rho_\nu \nabla_{x^\nu} \|g_+^\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu})\|_\gamma,$$

from which we deduce $\|\nabla_{x^\nu} \theta_\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu})\| = \rho_\nu \big\| \nabla_{x^\nu} \|g_+^\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu})\|_\gamma \big\|$. Using $c_\nu \in (0,1)$ and simple continuity arguments, this shows that the test (9) must be satisfied eventually for all $\nu \in \bar{I}$ and for all k sufficiently large. Hence the corresponding penalty parameters ρ_ν are updated. This contradiction shows that $\bar{\mathbf{x}}$ is feasible, and thus we have shown the first part of the theorem.

Consider now the case in which $I_\infty \neq \emptyset$. Choose an index $\nu \in I_\infty$. Since $\nu \in I_\infty$, subsequencing if necessary, we can assume, without loss of generality, that $\{\mathbf{x}^k\} \to \bar{\mathbf{x}}$ and that for all k, the penalty parameter ρ^ν is updated. According to Step 2 of Algorithm 2.2, we then have

$$\|\nabla_{x^\nu}\theta_\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})\| > c_\nu\left[\rho_\nu\left\|\nabla_{x^\nu}\|g_+^\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})\|_\gamma\right\|\right]\quad\forall k.$$

By our continuity assumptions on the functions defining the problem and the fact that the penalty parameters go to infinity, this immediately implies that

$$\nabla_{x^\nu}\|g_+^\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})\|_\gamma \;\to\; 0. \tag{10}$$

We now distinguish two cases, namely whether $\bar{\mathbf{x}}$ is feasible or infeasible for player ν.

First consider the case where $\bar{\mathbf{x}}$ is not feasible for player ν. Then the mapping $\|g_+^\nu(\cdot,\bar{\mathbf{x}}^{-\nu})\|_\gamma$ is continuously differentiable at $\bar{x}^\nu$, and we get

$$\nabla_{x^\nu}\|g_+^\nu(\bar{x}^\nu,\bar{\mathbf{x}}^{-\nu})\|_\gamma \;=\; 0 \tag{11}$$

from (10). Since $g^\nu(\cdot,\bar{\mathbf{x}}^{-\nu})$ is convex by assumption, $\|g_+^\nu(\cdot,\bar{\mathbf{x}}^{-\nu})\|_\gamma$ is also convex. Then (11) shows that $\bar{x}^\nu$ is a global minimizer of the function $\|g_+^\nu(\cdot,\bar{\mathbf{x}}^{-\nu})\|_\gamma$ with $\|g_+^\nu(\bar{x}^\nu,\bar{\mathbf{x}}^{-\nu})\|_\gamma > 0$ (due to the infeasibility of $\bar{\mathbf{x}}$). In other words, case (a) occurs.

Now consider the case where $\bar{\mathbf{x}}$ is feasible for player ν. We first rewrite (10) explicitly as

$$\nabla_{x^\nu}g^\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})(\lambda^k)^\nu \to 0, \qquad \text{where} \qquad (\lambda^k)^\nu := \frac{g_+^\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})^{\gamma-1}}{\|g_+^\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})\|_\gamma^{\gamma-1}},$$

where $g_+^\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})^{\gamma-1}$ means that we apply the exponent $\gamma-1$ to each component of the vector $g_+^\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})$ (the same notation will also be used in our subsequent analysis). Now it is not difficult to see that there are constants $0<\alpha_1\le\alpha_2$ such that $\alpha_1\le\|(\lambda^k)^\nu\|_\gamma\le\alpha_2$ for all k. Hence we may assume that there is a limiting vector $\bar{\lambda}^\nu\neq 0$ such that

$$\nabla_{x^\nu}g^\nu(\bar{x}^\nu,\bar{\mathbf{x}}^{-\nu})\bar{\lambda}^\nu \;=\; 0, \qquad \bar{\lambda}^\nu \;\geq\; 0, \qquad (\bar{\lambda}^\nu)^T g^\nu(\bar{x}^\nu,\bar{\mathbf{x}}^{-\nu}) \;=\; 0,$$

where the last relation follows from the fact that, by definition of $(\lambda^\nu)^k$, we have $(\lambda_i^\nu)^k = 0$ for all indices i with $g_i^\nu(\bar{\mathbf{x}}) < 0$. Hence $\bar{x}^\nu$ is a Fritz John point for problem (5). Therefore, in this case, $\bar{x}^\nu$ either satisfies (b) or (c). □

The previous theorem describes in detail the behavior of Algorithm 2.2 and, in certain sense, clarifies that it is the presence of generalized stationary points that are not solutions of the GNEP that causes troubles. However, it gives no clear hint on what conditions on the problem can guarantee *a priori* that no troubles will arise. Our line of attack is clearly suggested by Theorem 2.5, and we therefore proceed to give sufficient conditions on the problem which ensure that $I_\infty=\emptyset$. This will guarantee that every limit point of the sequence generated by the algorithm is a solution of the GNEP.

In order to do this, we first introduce a nonstandard constraint qualification that, again, is suggested by (a) and (b) of Theorem 2.5. We denote by $\partial^*_{x^\nu}\|g_+^\nu(x^\nu,\mathbf{x}^{-\nu})\|_\gamma$ the set

$$\begin{aligned}\partial^*_{x^\nu}\|g_+^\nu(x^\nu,\mathbf{x}^{-\nu})\|_\gamma \quad := \quad \big\{\xi\in\mathbb{R}^{n_\nu}\;\big|\;&\exists\{\mathbf{y}^k\}\,\text{with}\,(y^k)^\nu\notin X_\nu((\mathbf{y}^k)^{-\nu})\,\text{such that}\\ &\{\mathbf{y}^k\}\to\mathbf{x}\,\text{and}\,\nabla_{x^\nu}\|g_+^\nu((y^k)^\nu,(\mathbf{y}^k)^{-\nu})\|_\gamma\to\xi\,\big\}.\end{aligned}$$

Note that if $x^\nu\notin X_\nu(\mathbf{x}^{-\nu})$, then $\|g_+^\nu(x^\nu,\mathbf{x}^{-\nu})\|_\gamma$ is continuously differentiable around $\mathbf{x}$ and

$$\partial^*_{x^\nu}\|g_+^\nu(x^\nu,\mathbf{x}^{-\nu})\|_\gamma \;=\; \big\{\nabla_{x^\nu}\|g_+^\nu(x^\nu,\mathbf{x}^{-\nu})\|_\gamma\big\}.$$

If, instead, x^ν belongs to the interior of $X_\nu(\mathbf{x}^{-\nu})$, then $\partial^*_{x^\nu}\|g^\nu_+(x^\nu,\mathbf{x}^{-\nu})\|_\gamma$ is obviously the empty set because no sequence $\{\mathbf{x}^k\}$ such that $(x^k)^\nu \notin X_\nu\big((\mathbf{x}^k)^{-\nu}\big)$ converging to $\mathbf{x}$ exists. Finally, if x^ν belongs to the boundary of $X_\nu(\mathbf{x}^{-\nu})$, the set $\partial^*_{x^\nu}\|g^\nu_+(x^\nu,\mathbf{x}^{-\nu})\|_\gamma$ can be interpreted, roughly speaking, as the set of subgradients in $\partial_{x^\nu}\|g^\nu_+(x^\nu,\mathbf{x}^{-\nu})\|_\gamma$ "arising" from infeasible points, cf. the corresponding subgradient definition from [5].

Exploiting the definition of the set $\partial^*_{x^\nu}\|g^\nu_+(x^\nu,\mathbf{x}^{-\nu})\|_\gamma$, we give the following definition.

Definition 2.6 We say that the GNEP (5) satisfies the constraint qualification CQ$_\gamma$ at a point $\bar{\mathbf{x}}$ if

$$0 \notin \partial^*_{x^\nu}\|g^\nu_+(\bar{x}^\nu,\bar{\mathbf{x}}^{-\nu})\|_\gamma, \qquad \forall \nu = 1,\dots,N. \tag{12}$$

In the case of optimization problems, this kind of constraint qualification was probably introduced in [8], see also [9, 10]. The one above is the natural extension of this optimization constraint qualification to GNEPs.

In order to put this constraint qualification in connection with more usual constraint qualifications, we also introduce the Extended Mangasarian-Fromovitz Constraint Qualification (EMFCQ) for GNEPs. This condition has already been used in penalty schemes for GNEPs in [20, 21].

Definition 2.7 We say that the GNEP (5) satisfies the EMFCQ at a point $\bar{\mathbf{x}}$ if, for every player $\nu = 1,\dots,N$, there exists a vector d^ν such that

$$\nabla_{x^\nu} g^\nu_i(\bar{x}^\nu,\bar{\mathbf{x}}^{-\nu})^T d^\nu < 0 \quad \forall i \in I^\nu_+(\bar{\mathbf{x}}),$$

where $I^\nu_+(\bar{\mathbf{x}}) := \big\{ i \in \{1,\dots,m_\nu\} \,\big|\, g^\nu_i(\bar{x}^\nu,\bar{\mathbf{x}}^{-\nu}) \geq 0 \big\}$ is the index set of all active and violated constraints at the point $\bar{\mathbf{x}}$.

Clearly, this is nothing else than the extension to GNEPs of a well-known corresponding condition for optimization problems.

Theorem 2.8 *Assume that the sequence $\{\mathbf{x}^k\}$ generated by Algorithm 2.2 applied to the GNEP (5) is bounded. Consider the following assertions:*

(a) *The EMFCQ holds at every limit point $\bar{\mathbf{x}}$ of $\{\mathbf{x}^k\}$;*

(b) *The CQ$_\gamma$ condition holds at every limit point $\bar{\mathbf{x}}$ of $\{\mathbf{x}^k\}$;*

(c) *The penalty parameters are updated a finite number of times only, i.e. $I_\infty = \emptyset$.*

Then the following implications hold:

$$(a) \quad \Rightarrow \quad (b) \quad \Rightarrow \quad (c).$$

Proof. $(a) \Rightarrow (b)$. Suppose that EMFCQ holds at a limit point $\bar{\mathbf{x}}$, but (12) is violated. Then there is a player ν such that $0 \in \partial^*_{x^\nu}\|g^\nu_+(\bar{x}^\nu,\bar{\mathbf{x}}^{-\nu})\|_\gamma$. Hence there is a sequence $\{\mathbf{y}^k\}$ with $(y^k)^\nu \notin X_\nu((\mathbf{y}^k)^{-\nu})$ such that $\{\mathbf{y}^k\} \to \bar{\mathbf{x}}$ and $\nabla_{x^\nu}\|g^\nu_+((y^k)^\nu,(\mathbf{y}^k)^{-\nu})\|_\gamma \to 0$. Taking into account the differentiability of the norm at the infeasible point $\mathbf{y}^k$, this implies

$$\sum_{i=1}^{m_\nu} (\lambda^k_i)^\nu \nabla_{x^\nu} g^\nu_i((y^k)^\nu,(\mathbf{y}^k)^{-\nu}) \to 0, \qquad \text{where} \qquad (\lambda^k)^\nu := \frac{g^\nu_+((y^k)^\nu,(\mathbf{y}^k)^{-\nu})^{\gamma-1}}{\|g^\nu_+((y^k)^\nu,(\mathbf{y}^k)^{-\nu})\|^{\gamma-1}_\gamma}. \tag{13}$$

Similar to the proof of Theorem 2.5, we may assume that both $\mathbf{y}^k \to \bar{\mathbf{x}}$ and $(\lambda^k)^\nu \to \bar{\lambda}^\nu$ on a suitable subsequence, where $\bar{\lambda}^\nu \in \mathbb{R}^{m_\nu}$ is a nonzero vector. Then, (13) gives

$$\sum_{i=1}^{m_\nu} \bar{\lambda}_i^\nu \nabla_{x^\nu} g_i^\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu}) = 0.$$

Since we obviously have $\bar{\lambda}_i^\nu = 0$ for all $i \notin I_+^\nu(\bar{\mathbf{x}})$ (this being the index set from Definition 2.7), this becomes

$$\sum_{i \in I_+^\nu(\bar{\mathbf{x}})} \bar{\lambda}_i^\nu \nabla_{x^\nu} g_i^\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu}) = 0.$$

Note that $\bar{\lambda}_i \geq 0$ for all $i \in I_+^\nu(\bar{\mathbf{x}})$, and that at least one component is strictly positive. Therefore, if we multiply the above equation by d^ν, where d^ν denotes the vector from the EMFCQ, we obtain

$$0 = \sum_{i \in I_+^\nu(\bar{\mathbf{x}})} \bar{\lambda}_i^\nu \nabla_{x^\nu} g_i^\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu})^T d^\nu < 0,$$

a contradiction. (Note that we assume $I_+^\nu(\bar{\mathbf{x}}) \neq \emptyset$ here since otherwise the statement is rather trivially satisfied.)

$(b) \Rightarrow (c)$. Suppose by contradiction that $I_\infty \neq \emptyset$ and let ν be an index in this set. Subsequencing if necessary, we may assume that ρ_ν is updated at every iteration and $\{\mathbf{x}^k\} \to \bar{\mathbf{x}}$ with $(x^k)^\nu \notin X_\nu((x^k)^{-\nu})$. By Step 2 of Algorithm 2.2, we then have $\nabla_{x^\nu} \|g_+^\nu((x^k)^\nu, (\mathbf{x}^k)^{-\nu})\|_\gamma \to 0$. The definition of $\partial^*_{x^\nu} \|g_+^\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu})\|_\gamma$ therefore gives $0 \in \partial^*_{x^\nu} \|g_+^\nu(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu})\|_\gamma$, a contradiction to (12). Hence all penalty parameters are updated a finite number of times only. □

The following example shows that indeed condition CQ$_\gamma$ is weaker than EMFCQ. Consider one player whose feasible set (in $\mathbb{R}$) is defined by $x \leq 0$ and $-x \leq 0$. It is obvious that the EMFCQ cannot hold at the origin (which is also the only feasible point). On the other hand, CQ$_\gamma$ holds at the origin. In fact, we have $\partial^*_x \|(x_+, (-x)_+)\|_\gamma = \{-1, 1\}$.

From a practical point of view, Algorithm 2.2 should be regarded as a starting point, and it is easy to verify that it is possible to define several variants without affecting the properties described in the two previous theorems. Below we describe some of these modifications. First of all, note that at Step 2 it is not essential to double the penalty parameter. If an updating occurs, we can well set $\rho_\nu \leftarrow \alpha_\nu \rho_\nu$, where α_ν is any constant greater than 1. Obviously, α_ν need not be the same for all iterations, what is crucial is that, if α_ν^k denotes the updating factor used at iteration k, then $\alpha_\nu^k \in [1 + \underline{\alpha}, 1 + \bar{\alpha}]$, where $\underline{\alpha} \leq \bar{\alpha}$ are positive constants. Another logical updating strategy could be that of choosing the new penalty parameter so that the test at Step 2 is "safely failed", i.e. to take the new ρ_ν so that

$$\|\nabla_{x^\nu} \theta_\nu((x^k)^\nu, (\mathbf{x}^k)^{-\nu})\| + \textit{constant} \leq c_\nu \left[\rho_\nu \left\| \nabla_{x^\nu} \|g_+^\nu((x^k)^\nu, (\mathbf{x}^k)^{-\nu})\|_\gamma \right\| \right],$$

where *constant* is a positive number.

We could also think to modify in a different way the test at Step 2. It should be clear from the discussion in this section that we want the penalty parameters to be updated as little as possible, because good convergence properties are related to the penalty parameters staying finite. Therefore, it seems reasonable to look for more stringent updating rules that, however, do not impair convergence. From the analysis carried out in this section, it is rather obvious that we really need to update the ν-th penalty parameter if we are converging to a stationary point of the ν-th penalized problem which is not feasible for the corresponding constrained problem. From this point of view, a natural variant of the algorithm is to replace the test in Step 2 by

$$\|\nabla_{x^\nu}\theta_\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})\| > c_\nu \left[\rho_\nu \left\| \nabla_{x^\nu}\|g_+^\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})\|_\gamma \right\|\right]$$

and

$$\left\| \nabla_{x^\nu} \left[\theta_\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu}) + \rho_\nu \|g_+^\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})\|_\gamma\right] \right\| \le \textit{constant},$$

where, again, *constant* is a positive number. Following this line of thought, we could even think of *reducing* the penalty parameters if this situation seems favorable. A possible test indicating that we are in the position to reduce the penalty parameter is, when the test at step 2 is failed by a large margin, for example

$$\|\nabla_{x^\nu}\theta_\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})\| \le 0.01c_\nu \left[\rho_\nu \left\| \nabla_{x^\nu}\|g_+^\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})\|_\gamma \right\|\right].$$

It is trivial to show that reducing the penalty parameters does not change any of the convergence properties of Algorithm 2.2 as long as only a finite number of reductions are permitted.

We finally note that, from a practical point of view, it might be convenient to perform more than one step of algorithm $\mathcal{A}$ in Step 3. That is, it is not necessary to perform just one step of algorithm $\mathcal{A}$ before checking whether to update the penalty parameters. It can be checked that all the results in this section still hold if we perform any number of iterations of algorithm $\mathcal{A}$ in Step 3, provided that there is an upper bound on the number of iterations performed between two successive penalty parameter updating tests.

Remark 2.9 As we mentioned in the introduction, it is often the case that we can distinguish two groups of constraints for each player ν: those depending on x^ν only and those depending on the other players' variables as well:

$$\begin{array}{ll} \min\limits_{x^\nu} & \theta_\nu(x^\nu,\mathbf{x}^{-\nu}) \\ \text{s.t.} & h^\nu(x^\nu) \le 0, \\ & g^\nu(x^\nu,\mathbf{x}^{-\nu}) \le 0 \end{array}$$

for certain functions $h^\nu : \mathbb{R}^{n_\nu} \to \mathbb{R}^{p_\nu}$ and $g^\nu : \mathbb{R}^{n_\nu} \to \mathbb{R}^{m_\nu}$.

If this is the case, we could consider a partial penalization scheme where only the "difficult" mixed constraints g^ν are penalized so that the penalized problem becomes

$$\begin{array}{ll} \min\limits_{x^\nu} & \theta_\nu(x^\nu,\mathbf{x}^{-\nu}) + \rho_\nu\|g_+^\nu(x^\nu,\mathbf{x}^{-\nu})\|_\gamma \\ \text{s.t.} & h^\nu(x^\nu) \le 0. \end{array}$$

Note that, in this case, the penalized problem is a constrained Nash game, as opposed to the unconstrained one we considered in this paper. All the developments presented so far go through for this partial penalization approach, with minor technical adjustments, especially in the regularity conditions.

In this paper, we favored the "total" penalization approach for sake of simplicity of presentation, and also because it appears simpler to develop algorithms $\mathcal{A}$ for the solution of the penalized problem in case all constraints have been penalized and "moved" to the objective function, see Section 3. One must note, however, that keeping the h as constraints in the penalized problem has its own advantages. Foremost among them is the fact that it is (theoretically) easy to develop algorithms that only produce iterates $\mathbf{x}^k$ that remain feasible for these constraints. The main advantage of this is that, in case the set defined by the constraints h all together (i.e. the set $\{\mathbf{x} \in \mathbb{R}^n : h^\nu(x^\nu) \leq 0, \nu = 1, \ldots, N\}$) is bounded, one can be sure that the sequence $\{\mathbf{x}^k\}$ produced by the penalty algorithm is also bounded. □

2.1 The Jointly Convex Case

We now consider the case of a Nash game with jointly convex constraints; in particular, we consider the case where the ν-th player's problem is

$$\begin{array}{ll} \min_{x^\nu} & \theta_\nu(x^\nu, \mathbf{x}^{-\nu}) \\ & g(x^\nu, \mathbf{x}^{-\nu}) \leq 0 \end{array} \tag{14}$$

for a continuously differentiable function $g : \mathbb{R}^n \to \mathbb{R}^m$ that is convex as a function of $\mathbf{x}$ (recall that the constraint function g is the same for all players in the jointly convex case, in contrast to the more general case discussed in the first part of this section).

This is a much more structured problem than the one considered in the previous subsection and, obviously, all the results established there are applicable to a jointly convex GNEP. However, due to the special structure, we expect to be able to get further results for GNEPs with jointly convex constraints.

To this end, however, we consider the following algorithm which differs from Algorithm 2.2 in the way the penalty parameters are updated.

Algorithm 2.10 *(Penalty Updating Scheme for Jointly Convex Constraints)*

Data: $\mathbf{x}^0 \in \mathbb{R}^n$, *$\rho_\nu > 0$ and $c_\nu \in (0,1)$ for all $\nu = 1, \ldots, N$. Set $k := 0$.*

Step 1: *If $\mathbf{x}^k$ is a solution of the GNEP* (1)*: STOP.*

Step 2: *If $\mathbf{x}^k$ is not feasible, i.e. if $g(\mathbf{x}^k) \not\leq 0$, and*

$$\|\nabla_{x^\nu}\theta_\nu((x^k)^\nu, (\mathbf{x}^k)^{-\nu})\| > c_\nu \left[\rho_\nu \left\| \nabla_{x^\nu}\|g_+((x^k)^\nu, (\mathbf{x}^k)^{-\nu})\|_\gamma \right\|\right] \quad \forall\nu, \tag{15}$$

then double ρ_ν for all ν.

Step 3: *Compute $\mathbf{x}^{k+1} = \mathcal{A}[\mathbf{x}^k]$, set $k \leftarrow k+1$, and go to Step 1.*

To better appreciate the differences between this algorithm and Algorithm 2.2, we first observe that, in the present case, if a point $\mathbf{x}$ is feasible, i.e. if $g(\mathbf{x}) \leq 0$, then the set I^k considered at Step 2 of Algorithm 2.2 is empty. On the other hand, if $\mathbf{x}$ is not feasible, i.e. if $g(\mathbf{x}) \not\leq 0$, then the set I^k includes all the indices $\nu = 1, \ldots, N$. Having this difference in mind, we see that Algorithm 2.10 updates *all* the penalty parameters if the updating test (29) holds for *all* players, while in Algorithm 2.2 the test and the updating occur individually for each player. Among other things, this implies that in Algorithm 2.10 either all the penalty parameters are updated a finite number of times or they all grow to infinity. For simplicity, we could therefore use in this case a single penalty parameter for all players. We prefer to keep different penalty parameters ρ_ν both because this may be useful numerically (we can still use different initial penalty parameters for each player; furthermore see the comments after Theorem 2.8) and to parallel as much as possible the notation used previously.

Theorem 2.5 can be easily extended to the current setting.

Theorem 2.11 *Consider the GNEP* (14) *with jointly convex constraints. Let* $\{\mathbf{x}^k\}$ *be the sequence generated by Algorithm 2.10. If the penalty parameters are updated only a finite number of times, then every limit point* $\bar{\mathbf{x}}$ *of this sequence is a solution of the GNEP.*

If, instead, the penalty parameters are updated infinitely many times (and therefore grow to infinity) and the sequence $\{\mathbf{x}^k\}$ *is bounded, then there is a limit point* $\bar{\mathbf{x}}$ *such that, for each* $\nu = 1, \ldots, N$*, the block component* $\bar{x}^\nu$ *is such that one of the following assertions is true:*

(a) $\bar{x}^\nu$ *is a global minimizer of the constraint violation* $\|g_+(\cdot, \bar{\mathbf{x}}^{-\nu})\|_\gamma$ *with* $\|g_+(\bar{x}^\nu, \bar{\mathbf{x}}^{-\nu})\|_\gamma > 0$*;*

(b) $\bar{x}^\nu$ *is Fritz John point for the player's problem (14) (when the rivals play* $\bar{\mathbf{x}}^{-\nu}$*), but not a solution of (14);*

(c) $\bar{x}^\nu$ *is an optimal solution for the player's problem (14) (when the rivals play* $\bar{\mathbf{x}}^{-\nu}$*).*

Proof. Let $\bar{\mathbf{x}}$ be a limit point of the sequence $\{\mathbf{x}^k\}$ and assume, without loss of generality, that the whole sequence actually converges to $\bar{\mathbf{x}}$. Similar to the proof of Theorem 2.5, it suffices to show that $\bar{\mathbf{x}}$ is feasible for the GNEP (14). This can be shown exactly as in the proof of Theorem 2.5.

Assume now that the penalty parameters are updated infinitely many times (for all players $\nu = 1, \ldots, N$) in Algorithm 2.10. Since $\{\mathbf{x}^k\}$ is bounded, we may assume without loss of generality that $\{\mathbf{x}^k\}$ converges to some point $\bar{\mathbf{x}}$ and that the penalty parameters are updated at every iteration k. In view of the updating rule (15) in Algorithm 2.10, we have

$$\|\nabla_{x^\nu}\theta_\nu((x^k)^\nu, (\mathbf{x}^k)^{-\nu})\| > c_\nu \left[\rho_\nu \left\| \nabla_{x^\nu}\|g_+((x^k)^\nu, (\mathbf{x}^k)^{-\nu})\|_\gamma \right\|\right] \quad \forall k \;\; \forall \nu = 1, \ldots, N.$$

By continuity and the fact that all penalty parameters go to infinity, this implies

$$\nabla_{x^\nu}\|g_+((x^k)^\nu, (\mathbf{x}^k)^{-\nu})\|_\gamma \to 0 \quad \forall \nu = 1, \ldots, N. \tag{16}$$

The remaining part of the proof is essentially the same as the one of Theorem 2.5. The only thing one has to take into account is the fact that $\bar{x}^\nu$ is feasible for one player ν if and only if it is feasible for all players $\nu = 1, \ldots, N$ (due to the joint convexity of all players' strategy sets). □

Note the difference in the statements of the second part of Theorem 2.5 and of the theorem above: In Theorem 2.11 we state that there is a limit point $\bar{\mathbf{x}}$ of the sequence $\{\mathbf{x}^k\}$ such that, for all $\nu = 1, \ldots, N$, the block component $\bar{x}^\nu$ is a generalized stationary point of player ν-th optimization problem, i.e., we have *one* limit point whose components yield generalized stationary points for *all* players $\nu = 1, \ldots, N$. On the other hand, in Theorem 2.5, we showed that, for each player ν, there exists a limit point $\bar{\mathbf{x}}$ (possibly depending on ν) such that $\bar{x}^\nu$ solves the optimization problem of player ν.

Similar to the analysis carried out after Theorem 2.5, our next concern is to understand when the penalty parameters are updated a finite number of times only. The conditions given in the next theorem are quite natural and closer to the standard conditions used in optimization.

Theorem 2.12 *Assume that the sequence* $\{\mathbf{x}^k\}$ *generated by Algorithm 2.10 applied at the GNEP* (14) *is bounded. Consider the following assertions:*

(a) *the constraints $g(\mathbf{x})$ are linear, i.e. $g(\mathbf{x}) = A\mathbf{x} - b$;*

(b) *Slater's condition holds, i.e. there exists a point $\hat{\mathbf{x}}$ such that $g(\hat{\mathbf{x}}) < 0$;*

(c) *In Algorithm 2.10, the penalty parameters are updated a finite number of times only.*

Then the following implications hold:

$$(a) \quad \Rightarrow \quad (c) \quad \Leftarrow \quad (b)$$

Proof. $(a) \Rightarrow (c)$. Suppose that the penalty parameters ρ_ν are unbounded. Since the sequence $\{\mathbf{x}^k\}$ is bounded, we can assume without loss of generality that it converges to a point $\bar{\mathbf{x}}$ and that the penalty parameters ρ_ν are updated for all k. In view of Step 2 in Algorithm 2.10, the test (15) is therefore satisfied for all k and all $\nu = 1, \ldots, N$. Using continuity arguments, this implies $\left\{\nabla_{x^\nu} \|g_+(\mathbf{x}^k)\|_\gamma\right\} \to 0$, $\forall \nu = 1, \ldots, N$. This immediately gives

$$\left\{\nabla_{\mathbf{x}} \|g_+(\mathbf{x}^k)\|_\gamma\right\} \to 0. \tag{17}$$

We first claim that $\bar{\mathbf{x}}$ is feasible for all players ν. In fact, otherwise g_+ would be continuously differentiable at $\bar{\mathbf{x}}$, and (17) would imply $\nabla_{\mathbf{x}} \|g_+(\bar{\mathbf{x}})\|_\gamma = 0$. However, the mapping $\|g_+(\cdot)\|$ is convex due to the assumed convexity of $g(\cdot)$, and therefore $\nabla_{\mathbf{x}} \|g_+(\bar{\mathbf{x}})\|_\gamma = 0$ would imply that $\bar{\mathbf{x}}$ is a global minimizer of this function. But the global minimizers of this mapping are precisely the feasible points, a contradiction to the assumption that $\bar{\mathbf{x}}$ is not feasible.

Hence we know that the limit point $\bar{\mathbf{x}}$ is feasible for each player $\nu = 1, \ldots, N$. On the other hand, the iterates $\mathbf{x}^k$ are infeasible for all k since otherwise by the rules of Step 2, ρ_ν would not be updated at iteration k. Hence the mapping $v(\mathbf{x}) := \|g_+(\mathbf{x})\|_\gamma = \|\max\{0, A\mathbf{x} - b\}\|_\gamma$ is continuously differentiable at each point $\mathbf{x}^k$. Its gradient is given by

$$\nabla v(\mathbf{x}^k) = \nabla g(\mathbf{x}^k) \frac{g_+(\mathbf{x}^k)^{\gamma-1}}{\|g_+(\mathbf{x}^k)\|_\gamma^{\gamma-1}} = \sum_{i=1}^m \lambda_i^k a_i^T, \quad \text{where} \quad \lambda_i^k = \frac{\max\{0, a_i\mathbf{x}^k - b_i\}^{\gamma-1}}{\|\max\{0, a_i\mathbf{x}^k - b_i\}\|_\gamma^{\gamma-1}} \quad \forall i \tag{18}$$

and where a_i denotes the i-th row of the matrix A. Note that the definition of v and (17) together imply

$$\nabla v(\mathbf{x}^k) = \sum_{i=1}^m \lambda_i^k a_i^T \to 0 \quad \text{for } k \to \infty. \tag{19}$$

Let us denote by $I(\mathbf{x}^k) := \{i \,|\, a_i\mathbf{x}^k > b_i\}$ the set of violated constraints at the iterate $\mathbf{x}^k$ (recall that this index set is nonempty for each k). Subsequencing if necessary, we can assume without loss of generality that the set $I(\mathbf{x}^k)$ is independent of k. We denote this set by I. Obviously, we have $I \subseteq I(\bar{\mathbf{x}}) := \{i \,|\, a_i\bar{\mathbf{x}} = b_i\}$. We next show that the vectors a_i $(i \in I)$ are positively linearly independent. Assume this is not true. Then we can write

$$a_{i_0} + \sum_{i \in I \setminus \{i_0\}} \gamma_i a_i = 0$$

for some numbers $\gamma_i \geq 0$ and an index $i_0 \in I$. Hence we have

$$a_{i_0} = -\sum_{i \in I \setminus \{i_0\}} \gamma_i a_i. \tag{20}$$

Furthermore, since $i_0 \in I$ and $I \subseteq I(\bar{\mathbf{x}})$, we have

$$b_{i_0} = a_{i_0}\bar{\mathbf{x}} = -\sum_{i \in I \setminus \{i_0\}} \gamma_i a_i \bar{\mathbf{x}} = -\sum_{i \in I \setminus \{i_0\}} \gamma_i b_i. \tag{21}$$

By construction of the index set I, however, all constraints belonging to I are violated at each iterate $\mathbf{x}^k$. Using (20) and $\gamma_i \geq 0$, we can therefore write

$$b_{i_0} < a_{i_0}\mathbf{x}^k = -\sum_{i \in I \setminus \{i_0\}} \gamma_i a_i \mathbf{x}^k \leq -\sum_{i \in I \setminus \{i_0\}} \gamma_i b_i.$$

This, however, contradicts (21). Consequently, the vectors a_i $(i \in I)$ are positively linearly independent.

We now turn to the final contradiction: The definition of λ_i^k in (18) shows that, similar to the reasoning in the proof of Theorem 2.5, that we can assume without loss of generality that the sequence $\{\lambda^k\}$ converges to a vector $\bar{\lambda} \neq 0$. Note also that $\bar{\lambda}$ is nonnegative. Taking the limit $k \to \infty$, we then get from (19) that

$$0 = \sum_{i=1}^m \bar{\lambda}_i a_i^T = \sum_{i \in I} \bar{\lambda}_i a_i^T,$$

with the second equality being a consequence of the fact that $\bar{\lambda}_i = 0$ for all $i \notin I$. This shows that the vectors a_i $(i \in I)$ are positively linearly dependent, a contradiction.

$(b) \Rightarrow (c)$. Suppose that the penalty parameters ρ_ν are unbounded. Since the sequence $\{\mathbf{x}^k\}$ is bounded, we can assume without loss of generality, that it converges to a point $\bar{\mathbf{x}}$ and that the penalty parameters ρ_ν are updated for all k. Using the convexity of g, we can show as we did for the case of linear constraints that the limit point $\bar{\mathbf{x}}$ is feasible for all players $\nu = 1, \ldots, N$. On the other hand, the iterates $\mathbf{x}^k$ are infeasible for all k. Hence the mapping $v(\mathbf{x}) := \|g_+(\mathbf{x})\|_\gamma = \|\max\{0, g(\mathbf{x})\}\|_\gamma$ is continuously differentiable at each $\mathbf{x}^k$ with gradient

$$\nabla v(\mathbf{x}^k) = \sum_{i=1}^m \lambda_i^k \nabla g_i(\mathbf{x}^k) \qquad \text{where} \qquad \lambda_i^k := \frac{\max\{0, g_i(\mathbf{x}^k)\}^{\gamma-1}}{\|\max\{0, g_i(\mathbf{x}^k)\}\|_\gamma^{\gamma-1}} \quad \forall i = 1, \ldots, m. \tag{22}$$

Similar to the proof of the previous case, we may deduce from the fact that ρ_ν is updated for all k that

$$\nabla v(\mathbf{x}^k) \to 0, \tag{23}$$

cf. (17), (19). Let us denote by $I(\mathbf{x}^k) := \{i \,|\, g_i(\mathbf{x}^k) > 0\}$ the indices of violated constraints at the current iterate $\mathbf{x}^k$. Without loss of generality, we can assume that these sets are independent of k and denote this set by I. Similar to the proof of Theorem 2.5, we can also assume that the sequence $\{\lambda^k\}$ converges to a vector $\bar{\lambda} \in \mathbb{R}^m$ satisfying

$$\bar{\lambda} \geq 0 \quad \text{and} \quad \bar{\lambda} \neq 0. \tag{24}$$

Taking the limit for $k \to \infty$ and using (22) and (23), we therefore obtain from the definition of I that

$$0 = \sum_{i \in I} \bar{\lambda}_i \nabla g_i(\bar{\mathbf{x}}), \tag{25}$$

since we obviously have $\bar{\lambda}_i = 0$ for all $i \notin I$. Since $I \subseteq I(\bar{\mathbf{x}}) := \{i \,|\, g_i(\bar{\mathbf{x}}) = 0\}$, it follows from the convexity of g_i that

$$\nabla g_i(\bar{\mathbf{x}})^T(\hat{\mathbf{x}} - \bar{\mathbf{x}}) \leq g_i(\hat{\mathbf{x}}) - g_i(\bar{\mathbf{x}}) = g_i(\hat{\mathbf{x}}) < 0 \quad \forall i \in I,$$

where $\hat{\mathbf{x}}$ denotes the Slater point from our assumption. Premultiplication of (25) by $(\hat{\mathbf{x}} - \bar{\mathbf{x}})^T$ therefore gives

$$0 = \sum_{i \in I} \bar{\lambda}_i \nabla g_i(\bar{\mathbf{x}})^T(\hat{\mathbf{x}} - \bar{\mathbf{x}}) \leq \sum_{i \in I} \bar{\lambda}_i g_i(\hat{\mathbf{x}}).$$

This, in turn, yields $\bar{\lambda}_i = 0$ for all $i \in I$. Since we also have $\bar{\lambda}_i = 0$ for all $i \notin I$, this is a contradiction to $\bar{\lambda} \neq 0$. □

It may be interesting to compare the conditions in Theorem 2.8 (when applied to a jointly convex problem) and those in Theorem 2.12. Since in the case of optimization problems with convex and differentiable (inequality) constraints, the Mangasarian-Fromovitz constraint qualification and Slater's one are equivalent, one could think this goes through to GNEPs: this is not so. The EMFCQ (at feasible points) is stronger than the Slater condition in the theorem above. In fact, consider a feasible point $\mathbf{x}$ for a jointly convex GNEP and assume that it satisfies the EMFCQ. By setting $d := (d^1, \ldots, d^N)$ (the vectors d^1, …, d^N being those in the definition of the EMFCQ), it is clear that we have $\nabla_{\mathbf{x}} g(\mathbf{x})^T d < 0$ and in turn, it is classical (and easy) to show that this implies Slater's condition in Theorem 2.12. The converse does not hold. In fact, consider a jointly convex GNEP in which there are two players, each one controlling one variable, and assume $m = 1$ with $g(\mathbf{x}) = (\mathbf{x}^1)^2 + (\mathbf{x}^2)^2 - 1$. It is obvious that Slater's condition holds (for example take $\hat{\mathbf{x}}$ to be the origin). On the other hand, consider the point $(1, 0)$. At this point we have $\nabla_{\mathbf{x}^2} g(\mathbf{x}) = 0$ and the EMFCQ therefore cannot hold.

Passing to the CQ$_\gamma$, we can see that this condition neither implies the Slater's condition nor is implied by it. In fact, consider again the example we used above to show that the EMFCQ may not be satisfied while Slater's condition holds. It is easy to check that $0 \in \partial^*_{\mathbf{x}^2} \|((\mathbf{x}^1)^2 + (\mathbf{x}^2)^2 - 1)_+\|_\gamma$ at $(1, 0)$ by using the sequence of infeasible points $\mathbf{x}^k = \{(1, 1/k)\}$. On the other hand, consider a game with one player (i.e. an optimization problem) with feasible set (in $\mathbb{R}$) given by $x \leq 0$ and $-x \leq 0$. As discussed after Theorem 2.8, CQ$_\gamma$ holds at the origin, but it is clear that the feasible region is just the origin, hence Slater's condition does not hold.

Finally, the following example shows that CQ$_\gamma$ may not be satisfied for linear constraints, a case which is, however, covered by Theorem 2.12.

Example 2.13 Consider a game with $N = 2$ players given by

$$\begin{array}{ll|ll} \min_{x_1} & x_1^2 & \min_{x_2, x_3} & x_2^2 + x_3^2 \\ \text{s.t.} & x_1 + x_2 \leq 0, & \text{s.t.} & x_1 + x_2 \leq 0, \\ & x_3 \leq 0, & & x_3 \leq 0. \end{array}$$

Player $\nu = 1$ has x_1 as its only decision variable, whereas player $\nu = 2$ has the two variables x_2 and x_3. Obviously, $\bar{\mathbf{x}} := (0, 0, 0)$ is a Nash equilibrium of this game. Now have a look at the first player, whose constraints can be written as

$$g(\mathbf{x}) \leq 0 \qquad \text{with} \qquad g(\mathbf{x}) := \begin{pmatrix} x_1 + x_2 \\ x_3 \end{pmatrix}.$$

Using the sequence $\{\mathbf{x}^k\} := \{(0,0,1/k)\} \to \bar{\mathbf{x}}$, we can easily see that condition (12) is violated for the first player. Obviously, EMFCQ is also violated since the gradient (with respect to x_1) of the second constraint is zero. $\Diamond$

3 Implementation and Numerical Results

What we described in the previous section is not a single algorithm, but a family of algorithms. Independently of the many variants we mentioned after Theorem 2.8, the main point is that we still did not specify the algorithm $\mathcal{A}$: for every single choice of $\mathcal{A}$, we get a different algorithm. Although we believe that an appropriate choice of $\mathcal{A}$ is crucial to the numerical success of the penalty scheme described in the previous section, it is outside the scope of this paper to analyze and compare different algorithms $\mathcal{A}$. In this section, we therefore consider a single choice for $\mathcal{A}$, a choice that we think is effective in practice, and then proceed to a rather extensive numerical testing in order to have a feel for the efficiency of our approach.

We recall that algorithm $\mathcal{A}$ must be able to solve for arbitrary, but fixed, values of the penalty parameters ρ_ν, the penalized Nash game PNEP. The PNEP is a standard (unconstrained) Nash equilibrium problem whose solution is not straightforward because of the inherent nondifferentiability of the objective functions $P_\nu(\mathbf{x}, \rho_\nu)$. In principle, there exist methods for the solution of standard NEPs which do not require differentiability assumptions, see, for example, the relaxation method from [44], its modification from [26], or the proximal-like methods from [18]. However, these methods assume that certain constrained optimization problems can be solved, and, in practice, this is only possible for sufficiently smooth data. In this section, we therefore give an alternative approach for the solution of the PNEP that exploits the special structure of the nonsmooth penalty functions.

We first recall that the objective function of player ν in the PNEP is given by

$$\begin{aligned} P_\nu(\mathbf{x}, \rho_\nu) &= P_\nu(x^\nu, \mathbf{x}^{-\nu}, \rho_\nu) \\ &= \theta_\nu(x^\nu, \mathbf{x}^{-\nu}) + \rho_\nu \|g_+^\nu(x^\nu, \mathbf{x}^{-\nu})\|_\gamma \\ &= \theta_\nu(x^\nu, \mathbf{x}^{-\nu}) + \rho_\nu \left(\sum_{i=1}^{m_\nu} \max\{0, g_i^\nu(x^\nu, \mathbf{x}^{-\nu})\}^\gamma \right)^{1/\gamma}. \end{aligned}$$

We now approximate these functions by the smooth mappings

$$\begin{aligned} \tilde{P}_\nu(\mathbf{x}, \rho_\nu, \varepsilon) &:= \tilde{P}_\nu(x^\nu, \mathbf{x}^{-\nu}, \rho_\nu, \varepsilon) \\ &:= \theta_\nu(x^\nu, \mathbf{x}^{-\nu}) + \rho_\nu \left(\sum_{i=1}^{m_\nu} \max\{0, g_i^\nu(x^\nu, \mathbf{x}^{-\nu})\}^\gamma + \varepsilon \right)^{1/\gamma} + \frac{\varepsilon}{2}\|x^\nu\|^2, \end{aligned}$$

where $\varepsilon > 0$ is a given parameter which makes the root mapping smooth. The choice of γ is more delicate. So far, γ was taken arbitrarily from the open interval $(1, \infty)$. From now on we assume that $\gamma > 2$ holds. This makes the max-term twice continuously differentiable so that, assuming that θ_ν and g^ν are twice continuously differentiable, $\tilde{P}_\nu$ itself is twice continuously differentiable (note that the "obvious" choice $\gamma = 2$ only ensures that $\tilde{P}_\nu$ be a C^1-mapping). This observation will be exploited in a short while. We are therefore naturally led to define a smooth "approximation" of the PNEP, namely the PNEP(ε) where the problem of player ν is minimizing the function $\tilde{P}_\nu$.

The presence of the regularization term $(\varepsilon/2)\|x^\nu\|^2$ guarantees that $\tilde{P}_\nu(\mathbf{x}, \rho_\nu, \varepsilon)$ is uniformly convex as a function of x^ν. While this is a nice property, it is not strictly needed, our subsequent discussions also hold without this regularization term. We could also consider a different ϵ or γ for each player; moreover, we could use two different constants for the ε under the root and the one in the regularization term. We refrain from doing this in order to keep things as simple as possible.

We observe that the PNEP(ε) is a game where each player's problem is a continuously differentiable, unconstrained, convex optimization problem. Its solutions are therefore the solutions of the following system of equations:

$$F_\varepsilon(\mathbf{x}) := \begin{pmatrix} \nabla_{x^1} \tilde{P}_1(x^1, \mathbf{x}^{-1}, \rho_1, \varepsilon) \\ \vdots \\ \nabla_{x^N} \tilde{P}_N(x^N, \mathbf{x}^{-N}, \rho_N, \varepsilon) \end{pmatrix} = 0. \tag{26}$$

Since we assumed that $\gamma > 2$, this system is C^1 and, therefore, we can solve it by using a host of widely available, powerful and efficient methods.

Proposition 3.2 below is the basis for our approach that consists in attempting to solve the PNEP by solving inaccurately a sequence of PNEP(ε) for $\varepsilon \to 0$. In order to prove this proposition, we need a technical result from [17] whose proof we report here for sake of clarity and completeness.

Lemma 3.1 *Let $f : \mathbb{R}^s \times \mathbb{R}^t \to \mathbb{R}$ be given and assume that f is locally Lipschitz continuous around a point $(\bar{u}, \bar{v}) \in \mathbb{R}^s \times \mathbb{R}^t$ and such that $f(\cdot, v)$ is convex for every v in a neighborhood of $\bar{v}$. Let $\{(u^k, v^k)\}$ be a sequence of points converging to $(\bar{u}, \bar{v})$ and let $\{\xi^k\}$, with $\xi^k \in \partial_u f(u^k, v^k)$, be a sequence of (Clarke's) partial generalized gradients. Then, every limit point $\bar{\xi}$ of this sequence (and there is at least one such limit point) belongs to $\partial_u f(\bar{u}, \bar{v})$.*

Proof. By [5, Proposition 2.3.16] we know that $\xi^k \in \pi_u \partial f(u^k, v^k)$ (where π_u denotes projection on the u space). Therefore we can find a sequence $\{\eta^k\}$ such that $(\xi^k, \eta^k) \in \partial f(u^k, v^k)$ for all $k \in \mathbb{N}$. By the local boundedness and upper-semicontinuity of the generalized gradient of a locally Lipschitz function, we then see that we must have (renumbering if necessary) $\{(\xi^k, \eta^k)\} \to (\bar{\xi}, \bar{\eta}) \in \partial f(\bar{u}, \bar{v})$. But then, by the convexity assumption and [5, Proposition 2.5.3], we conclude that $\bar{\xi} \in \partial_u f(\bar{u}, \bar{v})$. □

Proposition 3.2 *Let $\{\varepsilon_k\}$ and $\{\eta_k\}$ be two sequences of positive numbers converging to 0 and, for every k, let $\mathbf{x}(\varepsilon_k)$ be a point such that*

$$\| F_{\varepsilon_k}(\mathbf{x}(\varepsilon_k)) \| \leq \eta_k. \tag{27}$$

Then every limit point of the sequence $\mathbf{x}(\varepsilon_k)$ is a solution of the PNEP (6).

Proof. Subsequencing if necessary, we may assume without loss of generality that $\mathbf{x}(\varepsilon_k) \to \bar{\mathbf{x}}$ with $\bar{\mathbf{x}} \in \mathbb{R}^n$. In order to show that $\bar{\mathbf{x}}$ is a solution of the PNEP, it is sufficient to show that, for all ν

$$0 \in \partial_{x^\nu} P_\nu(\mathbf{x}, \rho_\nu). \tag{28}$$

To this end, and for a fixed ν, define the function

$$\tilde{\tilde{P}}_\nu(\mathbf{x},\delta) := \theta_\nu(x^\nu,\mathbf{x}^{-\nu}) + \rho_\nu \left(\sum_{i=1}^{m_\nu} \max\{0, g_i^\nu(x^\nu,\mathbf{x}^{-\nu})\}^\gamma + \delta^\gamma\right)^{1/\gamma} + \frac{\delta^\gamma}{2}\|x^\nu\|^2$$

(remember that ρ_ν is a fixed quantity in the present setting), and note that $\tilde{\tilde{P}}_\nu$ is everywhere locally Lipschitz as a function of $(\bar{x},\delta)$ (note that, for this observation to be correct, we had to replace ε by δ^γ in the definition of $\tilde{\tilde{P}}_\nu$). Note furthermore that $P_\nu(\mathbf{x},\rho_\nu) = \tilde{P}_\nu(\mathbf{x},\rho_\nu,0) = \tilde{\tilde{P}}(\mathbf{x},0)$ and that $\partial_{x^\nu}\tilde{P}_\nu(\mathbf{x},\rho_\nu,\varepsilon_k) = \partial_{x^\nu}\tilde{\tilde{P}}(\mathbf{x},(\varepsilon_k)^{1/\gamma})$. We now set

$$f(\mathbf{x},\delta) := \tilde{\tilde{P}}_\nu(\mathbf{x},\delta)$$

and

$$\delta_k := (\varepsilon_k)^{1/\gamma}, \quad u^k := x^\nu(\varepsilon_k), \quad v^k := (\mathbf{x}^{-\nu}(\varepsilon_k),\delta_k), \quad \xi^k := \nabla_{x^\nu}\tilde{\tilde{P}}_\nu((x^\nu)^k,(\mathbf{x}^{-\nu})^k,\delta_k).$$

With these identifications in place and taking (26) and (27) into account, it is easy to see that Lemma 3.1 implies (28). □

The discussion above suggests to realize algorithm $\mathcal{A}$ by approximately solving a sequence of equations $F_\varepsilon(\mathbf{x}) = 0$ for $\varepsilon \downarrow 0$. In fact, since our main Algorithm 2.2 does not require the exact solution of the PNEP (just one iteration, where we are free to define what we view as one iteration), our implementation of algorithm $\mathcal{A}$ uses (for each outer iteration k) a fixed value of $\varepsilon = \varepsilon_k$ and applies a certain number of steps to the single nonlinear system of equations $F_\varepsilon(\mathbf{x}) = 0$ in order to compute the new iterate $\mathbf{x}^{k+1}$. Then $\varepsilon = \varepsilon_k$ is updated in the outer iteration depending on whether or not the PNEP was solved with a sufficiently high accuracy.

The following is a precise statement of our algorithm for solving general GNEPs (cf. Algorithm 2.2) using the ideas outlined in our previous discussion.

Algorithm 3.3 *(Algorithm for the solution of general GNEPs)*

Data: $\mathbf{x}^0 \in \mathbb{R}^n$, $\gamma > 2$, $\rho_\nu > 0$, *and* $c_\nu \in (0,1)$ *for all* $\nu = 1,\ldots,N$, $\varepsilon_0 > 0$, *term*$_1 > 0$, *term*$_2 > 0$, *term*$_3 > 0$, $S > 0$ *and integer. Set* $k := 0$.

Step 1: *If* $\mathbf{x}^k$ *is such that*

1. $\|\max\{0, g(\mathbf{x}^k\}\| \le$ *term*$_1$;
2. $\varepsilon^k \le$ *term*$_2$;
3. $\|F_{\varepsilon_k}(\mathbf{x})\| \le$ *term*$_3$;

then STOP.

Step 2: *Let* $I^k := \{\nu \,|\, (x^k)^\nu \notin X_\nu((\mathbf{x}^k)^{-\nu})\}$. *For every* $\nu \in I^k$, *if*

$$\|\nabla_{x^\nu}\theta_\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})\| > c_\nu \left[\rho_\nu \left\| \nabla_{x^\nu}\|g_+^\nu((x^k)^\nu,(\mathbf{x}^k)^{-\nu})\|_\gamma \right\|\right], \tag{29}$$

then multiply by 10 the penalty parameter ρ_ν.

Step 3: *Perform at most S steps of an equation solver to the nonlinear system of equations $F_{\varepsilon_k}(\mathbf{x}) = 0$. Let $\mathbf{x}^{k+1}$ be the final iterate of this equation solver.*

If $\|F_{\varepsilon_k}(\mathbf{x}^{k+1})\| \leq 1000\varepsilon_k$, set $\varepsilon_{k+1} = 0.1\varepsilon_k$, otherwise let $\varepsilon_{k+1} := \varepsilon_k$.

Set $k \leftarrow k+1$, and go to Step 1.

Step 1 is our termination criterion. Recall that a solution $\bar{\mathbf{x}}$ of the PNEP is a solution of the GNEP provided that $\bar{\mathbf{x}}$ is feasible for the GNEP. Therefore, based on the discussion in this section, at each iteration we check whether the current point is sufficiently feasible (test 1) and whether the point is sufficiently close to being a solution of the PNEP (tests 2 and 3.).

Step 2 is our penalty updating scheme for the case of general GNEPs and corresponds exactly to Step 2 of Algorithm 2.2, the only difference being that we found more effective to increase the penalty parameters by multiplying them by 10 instead of 2. In particular, note that we did not implement any of the other variants discussed after Theorem 2.8.

Finally, Step 3 contains our realization of algorithm $\mathcal{A}$ and is based on the discussion in this section. By saying that we apply at most S steps of an equation solver to the system $F_{\varepsilon_k}(\mathbf{x}) = 0$, we mean that we either solve this system within a given accuracy in less than S iterations, or we stop after S iterations without having solved the nonlinear equation within a required accuracy. Step 3 also reduces the current smoothing parameter ε_k if we were able to solve the system $F_{\varepsilon_k}(\mathbf{x}) = 0$ within a given tolerance. Note that, as required by Proposition 3.2, the accuracy with which the system $F_{\varepsilon_k}(\mathbf{x}) = 0$ is solved increases as the iterations progress.

Note that Algorithm 2.10 for jointly convex problems can also be implemented in a way similar to our realization of Algorithm 2.2 simply by replacing the penalty updating rule from Step 2 of Algorithm 3.3 by the corresponding updating rule from Algorithm 2.10.

3.1 Numerical Results

We implemented Algorithm 3.3 (as well as its variant for the jointly convex case) in MATLAB® using the parameters

$$\begin{aligned} &\text{term}_1 = 10^{-4}n, \quad \text{term}_1 = 10^{-5}, \quad \text{term}_3 = 10^{-4}n, \\ &\gamma = 3, \quad S = 20/50/150 \quad \text{and} \quad c_\nu = 0.1 \text{ for all } \nu \end{aligned}$$

as well as the initial values

$$\varepsilon_0 = 10^{-3} \quad \text{and} \quad \rho_\nu = 1 \text{ for all } \nu = 1, \ldots, N.$$

The parameter S should clearly depend on the dimension of the problem. We set $S = 20$ for all problems with a number of variables less or equal to 20, $S = 50$ for problems with a number of variables between 21 and 150 and 150 for the larger problems. We used the MATLAB® function `fsolve` from the Optimization Toolbox in order to (approximately) solve the nonlinear system of equations $F_{\varepsilon_k}(\mathbf{x}) = 0$ in Step 3. In this routine, and with our choice of options, only function values are used; first order information is approximated numerically. For this reason, and also because our method is globally convergent but not superlinearly convergent, we did not insist on a very stringent termination criterion (i.e. the parameters term_i are not too small). Nevertheless, each time we could compare our solutions with known ones, we found an accuracy that is at least of order 10^{-4}.

Example	N	n	Start. point	k	i_{total}	ε_f	$\|F_{\varepsilon_f}(\mathbf{x}^f)\|$	$\|g_+(\mathbf{x}^f)\|$	$\rho^f_{\max}$
A.1	10	10	0.01	5	34	1e-7	3.6e-13	7.6e-4	10
			0.1	5	33	1e-7	3.6e-13	7.6e-4	10
			1	5	30	1e-7	3.6e-13	7.6e-4	10
A.2	10	10	0.01	6	46	1e-8	2.4e-12	6.2e-4	10
			0.1	6	44	1e-08	7.6e-13	9.1e-4	10
			1	F					
A.3	3	7	0	3	4	1e-5	1.7e-11	0	1
			1	3	5	1e-5	2.3e-11	0	1
			10	3	8	1e-5	1.5e-11	0	10
A.4	3	7	0	16	247	1e-10	1.4e-10	5.6e-4	1e+5
			1	29	510	1e-08	1.4e-10	5.6e-4	1e+5
			10	22	351	1e-9	1.6e-10	3.8e-4	1e+4
A.5	3	7	0	3	43	1e-5	6.2e-6	4.3e-4	100
			1	3	43	1e-5	6.2e-6	4.3e-4	100
			10	5	62	1e-5	8.9e-16	4.3e-4	100
A.6	3	7	0	10	88	1e-10	5.3e-9	3.5e-4	1e+4
			1	9	121	1e-9	4.7e-10	6.1e-4	1e+4
			10	10	99	1e-10	5.4e-9	3.5e-4	1e+4
A.7	4	20	0	12	148	1e-9	3.6-10	0.001	1e+4
			1	12	151	1e-9	3.6e-10	0.001	1e+4
			10	F					
A.8	3	3	0	5	81	1e-5	4.6e-13	6.3e-5	100
			1	4	31	1e-6	9.4e-14	2.0e-4	10
			10	F					
A.9a	7	56	0	6	169	1e-8	2.8e-11	3.7e-3	100
A.9b	7	112	0	22	1034	1e-10	5.9e-8	9.6e-4	100
A.10a	8	24	see [14]	11	246	1e-9	6.4e-10	1.5e-3	1000
A.10b	25	125	see [14]	8	203	1e-8	3.1e-11	8.4e-3	100
A.10c	37	222	see [14]	101	14637	1e-8	4.8e-9	0.01	1e+4
A.10d	37	370	see [14]	11	1282	1e-7	2.3e-11	0.03	1000
A.10e	48	576	see [14]	12	1392	1e-7	1.6e-10	0.04	1000
A.11 [12]	2	2	0	9	60	1e-11	9.9e-12	9.6e-5	10
A.12 [31]	2	2	(2, 0)	3	5	1e-5	3.6e-12	0	1
A.13 [31]	3	3	0	10	84	1e-11	1.7e-9	1.5e-4	10
A.14 [29, 12]	10	10	0	3	16	1e-5	2.1e-10	0	100
A.15 [6]	3	6	0	3	9	1e-5	4.7e-10	0	1
A.16(P=75)[36]	5	5	10	10	102	1e-10	1.3e-7	3.9e-4	1000
A.16(P=100)	5	5	10	9	74	1e-10	1.7e-8	3.1e-4	1000
A.16(P=150)	5	5	10	10	101	1e-11	3.3e-8	2.9e-4	100
A.16(P=200)	5	5	10	10	73	1e-11	2.8e-8	2.3e-4	10
A.17 [33]	2	3	0	9	71	1e-10	1.5e-10	1.9e-4	100
A.18 [34]	2	12	0	19	346	1e-7	1.1e-7	0.001	1e+4
			1	12	182	1e-7	4.1e-7	5.2e-4	10e+4
			10	11	16	1e-7	4.4e-8	5.7e-4	1e+4

Table 1: Summary of the numerical results

We applied our method to two classes of problems, namely to general GNEPs where each player might have different constraints (problems from A.1 to A.10, these are essentially new problems) and to jointly convex problems (problems from A.11 to A.18, taken from the literature). Note that basically all the test problems for general GNEPs are new while those for jointly convex problems are taken from the literature (this corresponds, quite obviously, to a lack of reported numerical results for the solution of non jointly convex problems).

The numerical results are summarized in Table 1, where the columns have the following meaning:

Example Number of the test example taken from the appendix; for some problems, we also report a bibliographical reference;
N Number of players involved in the GNEP;
n Total number of variables $n = n_1 + \ldots + n_N$;
Start. point Represents the starting point. If a number is reported here, this means that all variables have been set to that value; otherwise the starting vector is reported;
k Number of outer iterations; an F here indicates failure;
i_{total} Total (cumulated) number of inner iterations (in `fsolve`) ;
ε_f Final value of the smoothing parameter ε;
$\|F_{\varepsilon_f}(\mathbf{x}^f)\|$: Measures, together with ε_f, the accuracy with which the PNEP is solved
$\|g_+(\mathbf{x}^f)\|$: Constraint violation at the final iterate;
$\rho^f_{\max}$ Final (maximal) value of the penalty parameter(s).

Recall that ρ_ν might be different for each player, hence we present the largest (final) penalty parameter in our table. Furthermore, the function g consists of all the constraint functions $g^1, \ldots, g^N$ stacked together to one vector. Therefore $\|\max\{0, g(\mathbf{x}^f)\}\|$ gives a fairly accurate estimate of the feasibility of our solution for the general GNEP, while for jointly convex problems, the same quantity actually underestimates feasibility because some constraints are repeated for all players (hence, in the jointly convex case, the final point is closer to feasibility than what the number $\|\max\{0, g(\mathbf{x}^f)\}\|$ may suggest).

The results above show that the algorithm seems able to solve rather efficiently a reasonable array of problems. A few observations are in order:

1. The algorithm is usually able to get easily close to a solution while reaching a high accuracy might be more problematic on some difficult problems. This had to be expected since when we are close to a solution and ε is small, we are trying to solve a(n almost) nondifferentiable unconstrained NEP. Note also that we use the MATLAB® routine `fsolve` to solve all the equations involved, and we use numerical differentiation to estimate first order information. It is rather obvious that, under these circumstances, a high accuracy cannot be expected in general. However, as we already mentioned, and in spite of all these caveats, the accuracy we get is usually at least in the order of 10^{-4}.

2. A point that is very important in the penalty approach is the handling of the penalty parameters. A proper initial choice and suitable updating rules seem crucial to the success of the algorithm. In this respect, we believe that our choices are reasonable, but there is certainly room for improvement here. For example, on single problems we could obtain substantially better results by appropriately choosing the initial value of ρ_ν (and ε), and even avoid some of the Failures.

3. Overall we have only three failures. In all cases, the failures were caused by the algorithm generating unbounded sequences.

4. Related to the point above, we note that another factor that seems to influence the behavior of the algorithm is the feasibility of the starting point. When the initial point is not feasible, the algorithm can have more difficulties in avoiding the generation of unbounded sequences; this is obviously connected also to the fact that, to keep the sequence generated by the algorithm bounded, we might have to rely on high penalty parameters, and if we start from an infeasible point, appropriate tuning may require a few iterations, thus, in unfortunate cases, giving opportunity to the algorithm to drive the sequence of points x^k to infinity.

5. It is interesting to observe that, also when the problem has an infinite number of solutions, the algorithm seems to converge always to the same solution, independent from the starting point. This suggests that the penalty approach implicitly operates a selection among the solutions. It would be extremely valuable to understand better this phenomenon. Fukushima in [20] addressed similar issues with respect to a different penalty scheme and got some interesting insights.

6. We also implemented a version of Algorithm 2.10 that we run on the jointly convex problems. The numerical results we obtained are very similar to those illustrated above and we therefore do not report them. At the moment, the main advantage of the specific updating rule for jointly convex problems analyzed in Section 2.1 seems to be theoretical in that convergence can be guaranteed in some interesting cases for which convergence cannot be guaranteed by the general scheme from Algorithm 2.2.

7. It is known (see e.g. [13]) that, for the most successful algorithms for the solution of jointly convex problems, an important feature of the problem that seems to differentiate easy from difficult ones is the positive (semi-)definiteness of the Jacobian of the function F defined by
$$F(\mathbf{x}) := \begin{pmatrix} \nabla_{x^1}\theta_1(\mathbf{x}) \\ \vdots \\ \nabla_{x^N}\theta_N(\mathbf{x}) \end{pmatrix}.$$
We propose to call the Jacobian of F the *Jaco-Hessian* of the underlying GNEP. Although there is currently no result showing that positive (semi-) definiteness of the Jaco-Hessian plays, for the non-jointly convex problems, a role similar to that it has in jointly convex problems, our impression is that this property also plays a crucial role for general GNEPs.

8. The value of the parameter S can have a considerable influence on the performance of the algorithm. We set S to the values mentioned at the beginning of this subsection without performing any refined analysis on the behavior of the algorithm when varying S. However, just to show what could be expected, we mention that on Problem A.18, the results reported are obtained with $S = 20$. If we set $S = 50$ and start the algorithm from the origin, we get convergence in 6 iterations (and 138 inner iterations) and convergence occurs to a different point from the one obtained with $S = 20$ when starting from the origin. The opposite situation can also occur and convergence can be slower. These changes are obviously due to the fact that, when S varies, different "trajectories" can

be obtained since the penalty parameters can be updated differently. In many cases though, a variation of S only produces a variation of the outer iterations, while the number of inner iterations remains more or less constant.

In order to get a better feeling of how the algorithm works, we report in Table 2 the detailed numerical results for problem A.17.

k	ρ_k	$(x_1^1)^k$	$(x_2^1)^k$	$(x_1^2)^k$	$\|\max\{0, g(\mathbf{x}^k)\}\|$	ε_k	$\|F_{\varepsilon_k}(\mathbf{x}^k)\|$	i_{total}
0	1	0.00000000	0.00000000	0.00000000	0.00000000	10^{-3}		
1	1	1.94821944	14.57766447	4.41381390	31.6005704818	10^{-3}	0.00000000	7
2	10	0.31474723	10.65786894	7.59786199	0.0652462332	10^{-4}	3.60734452	27
3	100	0.00121253	11.00196386	7.99708624	0.0186015391	10^{-4}	0.00000000	36
4	100	0.00038976	11.00108474	7.99882026	0.0086351352	10^{-5}	0.00000000	41
5	100	0.00016359	11.00052084	7.99946966	0.0040083084	10^{-6}	0.00000000	47
6	100	0.00007420	11.00024349	7.99975556	0.0018605494	10^{-7}	0.00000000	53
7	100	0.00003427	11.00011320	7.99988671	0.0008636065	10^{-8}	0.00000000	59
8	100	0.00001589	11.00005256	7.99994743	0.0004008557	10^{-9}	0.00000000	65
9	100	0.00000737	11.00002440	7.99997560	0.0001860626	10^{-10}	0.00000000	71

Table 2: Detailed numerical results for Example A.17

We can see that, in the first iteration, we jump from an initial feasible point to a “highly” infeasible one. After this, the penalty parameter is increased to 10 and thus, the feasibility violation having more weight, in the following iterations we easily progress towards feasibility. A further increase of the penalty parameter is needed in order to reach a higher accuracy. Obviously, things might not always be so “clean” and readily interpretable, and the rule for the updating of the penalty parameters may not always detect so easily when an increase is necessary, but in most cases the behavior is of the kind just described.

Regarding the iterates, it is easy to verify the phenomenon discussed above (Point 1): from iteration 3, we are clearly very close to the solution we are converging to, but a lot of effort is still needed to reach the required accuracy. As discussed above, this is due to the numerical difficulties of solving problems that are closer and closer to being nondifferentiable and to our method not being superlinearly convergent.

It is also interesting to look at the last column, reporting the (total) number of (inner) iterations needed by the equation solver `fsolve.` We recall that we set a limit of 20 (inner) iterations. So if, in a certain iteration k, the limit of 20 iterations is reached, this means that the system was not solved (and remember, we do not need to solve the system, this is one of the advantages of our approach!). For this problem, the solution of the system $F_{\varepsilon_k}(\mathbf{x}) = 0$ seems rather easy (and this is certainly not always the case). Note that, in the second iteration, we pass from 7 to 27 inner iterations, and in fact the residual of the system is approximatively 3.6. In the following iterations we always stop before the limit of 20 inner iterations, meaning that the system is solved exactly (correspondingly, we have a residual of 0 from the third iteration onwards). Note that, by the updating rule for ε at Step 3 of Algorithm 3.3, this implies that ε is reduced at each iteration from the third iteration onwards; and this can be checked in the corresponding column.

4 Conclusions

We presented an exact penalty approach for the solution of GNEPs, both with general constraints and with jointly convex constraints. We tested the algorithm on an array of problems considerably larger than those usually employed in the literature, and the numerical results indicate that the method seems to work quite well in practice. We therefore believe that our method is at least an interesting alternative to the very limited number of globally convergent methods for the solution of (general) GNEPs.

We believe that, at this stage of research, penalty methods are the only viable approach to the solution of general GNEPs, and intend to continue our research on this topic. There are many points that may lead to substantial numerical improvements. An important issue that certainly deserves central stage is the way used to solve the penalized Nash problem. In this paper, we adopted a smoothing method, but other choices are possible and we intend to investigate this point in detail. We think it is important that our analysis does not rely on any specific algorithm $\mathcal{A}$, thus making the analysis of alternatives relatively straightforward. Furthermore, we believe that, in order to obtain a high accuracy, the penalty method could be embedded in a Phase I - Phase II method where a switch is made, when appropriate, to some local Newton methods like those described in [11, 26, 27]. On a more theoretical level, a detailed analysis of boundedness issues is certainly paramount. With respect to this, we want to mention that partial penalization, whereby simple constraints (for example box constraints) are not penalized, is certainly an interesting issue to consider.

References

[1] K.J. Arrow and G. Debreu: *Existence of an equilibrium for a competitive economy.* Econometrica 22, 1954, pp. 265–290.

[2] T. Basar and G.J. Olsder: *Dynamic Noncooperative Game Theory.* Academic Press, New York, NY, second edition 1995.

[3] A. Bensoussan: *Points de Nash dans le cas de fonctionelles quadratiques et jeux differentiels lineaires a N personnes.* SIAM Journal on Control 12, 1974, pp. 460–499.

[4] J.V. Burke: *A robust trust region method for constrained nonlinear programming problems.* SIAM Journal on Optimization 2, 1992, pp. 325–347.

[5] F.H. Clarke: *Optimization and Nonsmooth Analysis.* John Wiley, New York, 1983.

[6] J. Contreras, M. Klusch, and J.B. Krawczyk: *Numerical solutions to Nash-Cournot equilibria in coupled constraint electricity markets.* IEEE Transactions on Power Systems 19, 2004, pp. 195–206.

[7] G. Debreu: *A social equilibrium existence theorem.* Proceedings of the National Academy of Sciences 38, 1952, pp. 886–893.

[8] G. Di Pillo and F. Facchinei: *Exact penalty functions for nondifferentiable programming problems.* In Nonsmooth Optimization and Related Topics, F.H. Clarke, V.F.Dem'yanov, and F. Giannessi editors, pp. 89–107. Plenum Press, New York, 1989.

[9] G. Di Pillo and F. Facchinei: *Exact barrier methods for Lipschitz programs.* Applied Mathematics and Optimization 32, 1995, pp. 1–31.

[10] F. Facchinei: *Refinements of necessary conditions for optimality in nonlinear programming.* Journal of Optimization Theory and Applications 73, 1992, pp. 65–74.

[11] F. Facchinei, A. Fischer, and V. Piccialli: *On generalized Nash games and VIs.* Operations Research Letters 35, 2007, pp. 159–164.

[12] F. Facchinei, A. Fischer, and V. Piccialli: *Generalized Nash equilibrium problems and Newton methods.* Mathematical Programming 117, 2009, pp. 163–194.

[13] F. Facchinei and C. Kanzow: *Generalized Nash equilibrium problems.* 4OR — A Quarterly Journal of Operations Research 5, 2007, pp. 173–210.

[14] F. Facchinei and C. Kanzow: *Penalty methods for the solution of generalized Nash equilibrium problems (with complete test problems).* Technical Report, Institute of Mathematics, University of Würzburg, Germany, February 2009.

[15] F. Facchinei and J.-S. Pang: *Finite-Dimensional Variational Inequalities and Complementarity Problems, Volume I.* Springer, New York, 2003.

[16] F. Facchinei and J.-S. Pang: *Exact penalty functions for generalized Nash problems.* In G. Di Pillo and M. Roma (eds): *Large Scale Nonlinear Optimization.* Springer, 2006, pp. 115–126.

[17] F. Facchinei, V. Piccialli, and M. Sciandrone: *A decomposition algorithm for generalized potential games.* DIS Technical Report, Sapienza University of Rome, forthcoming.

[18] S.D. Flåm and A.S. Antipin: *Equilibrium programming using proximal-like algorithms.* Mathematical Programming 78, 1997, pp. 29–41.

[19] S.D. Flåm and A. Ruszczyński: *Noncooperative convex games: Computing equilibria by partial regularization.* IIASA Working Paper 94-42, Laxenburg, Austria, May 1994.

[20] M. Fukushima: *Restricted generalized Nash equilibria and controlled penalty algorithm.* Technical Report 2008-007, Department of Applied Mathematics and Physics, Kyoto University, July 2008.

[21] M. Fukushima and J.-S. Pang: *Quasi-variational inequalities, generalized Nash equilibria, and multi-leader-follower games.* Computational Management Science 2, 2005, pp. 21–56.

[22] G. Gürkan and J.-S. Pang: *Approximations of Nash equilibria.* Technical Report, Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY, January 2006.

[23] P.T. Harker: *Generalized Nash games and quasi-variational inequalities.* European Journal of Operational Research 54, 1991, pp. 81–94.

[24] A. von Heusinger and C. Kanzow: *Optimization reformulations of the generalized Nash equilibrium problem using Nikaido-Isoda-type functions.* Computational Optimization and Applications, to appear.

[25] A. von Heusinger and C. Kanzow: *SC^1 optimization reformulations of the generalized Nash equilibrium problem.* Optimization Methods and Software 23, 2008, pp. 953–973.

[26] A. von Heusinger and C. Kanzow: *Relaxation methods for generalized Nash equilibrium problems with inexact line search.* Journal of Optimization Theory and Applications, to appear.

[27] A. von Heusinger, C. Kanzow, and M. Fukushima: *Newton's method for computing a normalized Nash equilibrium in the generalized Nash game through fixed point formulation.* Technical Report, Institute of Mathematics, University of Würzburg, Germany, January 2009.

[28] B.F. Hobbs: *Linear complementarity models of Nash-Cournot competition in bilateral and POOLCO power markets.* IEEE Transactions on Power Systems 16, 2001, pp. 194–202.

[29] A. Kesselman, S. Leonardi, and V. Bonifaci: *Game-theoretic analysis of internet switching with selfish users.* Lecture Notes in Computer Science 3828, 2005, pp. 236–245.

[30] J.B. Krawczyk: *Coupled constraint Nash equilibria in environmental games.* Resource and Energy Economics 27, 2005, pp. 157–181.

[31] J.B. Krawczyk and S. Uryasev: *Relaxation algorithms to find Nash equilibria with economic applications.* Environmental Modeling and Assessment 5, 2000, pp. 63–73.

[32] F.H. Murphy, H.D. Sherali, and A.L. Soyster: *A mathematical programming approach for determining oligopolistic market equilibrium.* Mathematical Programming 24, 1982, pp. 92–106.

[33] K. Nabetani: *Variational inequality approaches to generalized Nash equilibrium problems.* Master Thesis, Department of Applied Mathematics and Physics, Kyoto University, February 2008.

[34] K. Nabetani, P. Tseng, and M. Fukushima: *Parametrized variational inequality approaches to generalized Nash equilibrium problems with shared constraints.* Technical Report, Department of Applied Mathematics and Physics, Graduate School of Informatics, Kyoto University, Japan, October 2008.

[35] H. Nikaido and K. Isoda: *Note on noncooperative convex games.* Pacific Journal of Mathematics 5, 1955, pp. 807–815.

[36] J.V. Outrata, M. Kočvara, and J. Zowe: *Nonsmooth Approach to Optimization Problems with Equilibrium Constraints.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[37] J.-S. Pang: *Computing generalized Nash equilibria.* Technical Report, Department of Mathematical Sciences, The Johns Hopkins University, Baltimore, MD, October 2002.

[38] J.-S. Pang and M. Fukushima: *Quasi-variational inequalities, generalized Nash equilibria, and multi-leader-follower games.* Computational Management Science 2, 2005, pp. 21–56.

[39] J.-S. Pang, G. Scutari, F. Facchinei, and C. Wang: *Distributed power allocation with rate constraints in gaussian parallel interference channels.* IEEE Transactions on Information Theory 54, 2008, pp. 3471–3489.

[40] S.M. Robinson: *Shadow prices for measures of effectiveness, I: Linear model.* Operations Research 41, 1993, pp. 518–535.

[41] S.M. Robinson: *Shadow prices for measures of effectiveness, II: General model.* Operations Research 41, 1993, pp. 536–548.

[42] R.T. Rockafellar: *Convex Analysis.* Princeton University Press, Princeton, NJ, 1970.

[43] J.B. Rosen: *Existence and uniqueness of equilibrium points for concave N-person games.* Econometrica 33, 1965, pp. 520–533.

[44] S. Uryas'ev and Y. Rubinstein: *On relaxation algorithms in computation of noncooperative equilibria.* IEEE Transactions on Automatic Control 39, 1994, pp. 1263–1267.

A Test Examples

In this appendix, we report a detailed description of the test problems we used along with a few additional information. We divide this section into two subsections, with Subsection A.1 containing examples of general (not jointly convex) GNEPs, and Subsection A.2 presenting the jointly convex examples that we used in our numerical test runs.

A.1 General Problems

The majority of the problems presented here are new and were developed in order to test our code.

Example A.1 This test problem is a variant of the internet switching model introduced by Kesselman et al. [29] and further analyzed by Facchinei et al. [12], see also Example A.14 below. There are N players, each player having a single variable $x^\nu \in \mathbb{R}$. The objective functions are given by

$$\theta_\nu(\mathbf{x}) := \frac{-x^\nu}{x^1 + \ldots + x^N}\left(1 - \frac{x^1 + \ldots + x^N}{B}\right) \quad \forall \nu = 1, \ldots, N$$

for some constant B. We set $N = 10$ and $B = 1$. The constraints of the first player are $0.3 \leq x^1 \leq 0.5$, while the remaining players' constraints are

$$x^1 + \ldots + x^N \leq B, \quad x^\nu \geq 0.01.$$

Note that the objective functions are not continuous at $\mathbf{x} = 0$ which, however, is an infeasible point. This variant of the basic problem, described in Example A.14, gives the first player a "privileged status".

Convergence of Algorithm 3.3 from the three starting points used in Table 1 always occurs to exactly the same point, given by

```
0.29923815223336
0.06951127617805
0.06951127617805
0.06951127617805
0.06951127617805
0.06951127617805
0.06951127617805
0.06951127617805
0.06951127617805
0.06951127617805
```

Example A.2 This is a variant of the previous problem. We only report differences and changes. The objective functions of players 2, 3, 4, and 5 are

$$\theta_\nu(\mathbf{x}) := \frac{-x^\nu}{x^1+\ldots+x^N}\left(1-\frac{x^1+\ldots+x^N}{B}\right)^2.$$

Players 5 and 6 have the additional constraint

$$0.99 \,\le\, x^1+\ldots+x^N,$$

while the upper bounds of players 9 and 10 are 0.06 and 0.05, respectively.

By construction, the feasible region of this problem is very "narrow", since the sum of the (nonnegative) variables must be between 0.99 and 1. Convergence of Algorithm 3.3 for the first two starting points occurs to the following points (recall that we have a failure for the third starting point):

```
0.29962894677774     0.29962898846513
0.00997828224734     0.00997828313762
0.00997828224734     0.00997828313762
0.00997828224734     0.00997828313762
0.59852469355630     0.59745624992082
0.02187270661760     0.02220301920403
0.00999093169361     0.01013441012117
0.00999093169361     0.01013441012117
0.00999093169361     0.01013441012117
0.00999093169361     0.01013441012117
```

Example A.3 There are three players ($N = 3$) having 3, 2, and 2 variables, respectively. Each player has a quadratic strongly convex objective function given by

$$\theta_\nu(\mathbf{x}) \,=\, \frac{1}{2}(x^\nu)^T A^\nu x^\nu + (x^\nu)^T(B^\nu x^{-\nu} + b^\nu),$$

where the matrices and vectors involved are

$$A^1 = \begin{pmatrix} 20 & 5 & 3 \\ 5 & 5 & -5 \\ 3 & -5 & 15 \end{pmatrix}, \quad A^2 = \begin{pmatrix} 11 & -1 \\ -1 & 9 \end{pmatrix}, \quad A^3 = \begin{pmatrix} 48 & 39 \\ 39 & 53 \end{pmatrix},$$

$$B^1 = \begin{pmatrix} -6 & 10 & 11 & 20 \\ 10 & -4 & -17 & 9 \\ 15 & 8 & -22 & 21 \end{pmatrix}, \quad B^2 = \begin{pmatrix} 20 & 1 & -3 & 12 & 1 \\ 10 & -4 & 8 & 16 & 21 \end{pmatrix}, \quad B^3 = \begin{pmatrix} 10 & -2 & 22 & 12 & 16 \\ 9 & 19 & 21 & -4 & 20 \end{pmatrix},$$

$$b^1 = \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}, \quad b^2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad b^3 = \begin{pmatrix} -1 \\ 2 \end{pmatrix}.$$

The variables all have bound constraints: $-10 \leq \mathbf{x} \leq 10$. In addition, the first player has the linear constraints

$$x_1^1 + x_2^1 + x_3^1 \leq 20, \qquad x_1^1 + x_2^1 - x_3^1 \leq x_1^2 - x_2^3 + 5,$$

the second player has the single constraint

$$x_1^2 - x_2^2 \leq x_2^1 + x_3^1 - x_1^3 + 7,$$

and the third player has the constraint

$$x_2^3 \leq x_1^1 + x_3^1 - x_1^2 + 4.$$

The Jaco-Hessian of this example is constant, but indefinite.

Using the three starting points from Table 1, our method converges to the following points:

```
-0.38046562696258    -0.38046562696294    -0.38046562696275
-0.12266997083581    -0.12266997083590    -0.12266997083484
-0.99322817120517    -0.99322817120634    -0.99322817120582
 0.39034789080544     0.39034789080558     0.39034789080555
 1.16385412687962     1.16385412688026     1.16385412688162
 0.05039533464000     0.05039533464023     0.05039533463988
 0.01757740533460     0.01757740533476     0.01757740533435
```

Example A.4 This is a variant of the previous problem, where some terms depending on the other players' variables have been added to the matrices A^ν:

$$A^1 = \begin{pmatrix} 20 + (x_1^2)^2 & 5 & 3 \\ 5 & 5 + (x_2^2)^2 & -5 \\ 3 & -5 & 15 \end{pmatrix}, \quad A^2 = \begin{pmatrix} 11 + (x_1^3)^2 & -1 \\ -1 & 9 \end{pmatrix}, \quad A^3 = \begin{pmatrix} 48 & 39 \\ 39 & 53 + (x_1^1)^2 \end{pmatrix}.$$

Furthermore, the lower bounds have all been set to 1 (instead of -10). The Jaco-Hessian is neither constant nor positive (semi-) definite in general. Convergence of our method, based on the three starting points given in Table 1, occurs to the following points:

```
 0.99982626069210     0.99982626069210     0.99991935573409
 0.99996267821636     0.99996267821636     0.99998267343165
 0.99987070414176     0.99987070414176     0.99993998316707
 0.99985869062731     0.99985869062731     0.99993440803353
 0.99983447394048     0.99983447394048     0.99992316858615
 0.99991824127925     0.99991824127925     0.99987984742324
 0.99991381820076     0.99991381820076     0.99987334759435
```

Example A.5 This example has the same structure as Problem A.3, but with the following data:

$$A^1 = \begin{pmatrix} 20 & 6 & 0 \\ 6 & 6 & -1 \\ 0 & -1 & 8 \end{pmatrix}, \quad A^2 = \begin{pmatrix} 11 & 1 \\ 1 & 7 \end{pmatrix}, \quad A^3 = \begin{pmatrix} 28 & 14 \\ 14 & 29 \end{pmatrix},$$

$$B^1 = \begin{pmatrix} -1 & -2 & -4 & -3 \\ 0 & -3 & 0 & -4 \\ 0 & 1 & 9 & 6 \end{pmatrix}, \quad B^2 = \begin{pmatrix} -1 & 0 & 0 & -7 & 4 \\ -2 & -3 & 1 & 4 & 11 \end{pmatrix}, \quad B^3 = \begin{pmatrix} -4 & 0 & 9 & -7 & 4 \\ -3 & -4 & 6 & 4 & 11 \end{pmatrix},$$

$$b^1 = \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}, \quad b^2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad b^3 = \begin{pmatrix} -1 \\ 2 \end{pmatrix}.$$

Lower bounds on all variables are all set to 0, while the upper bounds and the linear constraints are the same used in Problem A.3. Here, the Jaco-Hessian is constant and positive definite. Convergence of our method using the three starting points from Table 1 occurs to the following points:

```
-0.00006229891126      -0.00006229891126      -0.00006229910314
 0.20279012064850       0.20279012064850       0.20279011130836
-0.00003469558295      -0.00003469558295      -0.00003469562269
-0.00028322020054      -0.00028322020054      -0.00028322027018
 0.07258934064261       0.07258934064261       0.07258933626181
 0.02531280162415       0.02531280162415       0.02531280221816
-0.00007396699835      -0.00007396699835      -0.00007396699882
```

Example A.6 This is another variant of Problem A.3 with the following data:

$$A^1 = \begin{pmatrix} 20 + (x_1^2)^2 & 5 & 3 \\ 5 & 5 + (x_2^2)^2 & -5 \\ 3 & -5 & 15 \end{pmatrix}, \quad A^2 = \begin{pmatrix} 11 + (x_1^3)^2 & -1 \\ -1 & 9 \end{pmatrix}, \quad A^3 = \begin{pmatrix} 48 & 39 \\ 39 & 53 + (x_1^1)^2 \end{pmatrix},$$

$$B^1 = \begin{pmatrix} -2 & 0 & 1 & 2 \\ 1 & -4 & -7 & 9 \\ -3 & 8 & 22 & 21 \end{pmatrix}, \quad B^2 = \begin{pmatrix} -2 & 1 & -3 & -12 & -1 \\ 0 & -4 & 8 & 16 & 21 \end{pmatrix}, \quad B^3 = \begin{pmatrix} 1 & -7 & 22 & -12 & 16 \\ 2 & -9 & 21 & -1 & 21 \end{pmatrix},$$

$$b^1 = \begin{pmatrix} 1 \\ -2 \\ -3 \end{pmatrix}, \quad b^2 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad b^3 = \begin{pmatrix} 1 \\ -2 \end{pmatrix}.$$

All variables have a lower bound of 1 and an upper bound of 10. The non-bound constraints include also nonlinear constraints. The first player has the following non-bound constraints,

$$x_1^1 + x_2^1 + x_3^1 \le 20, \qquad x_1^1 + x_2^1 - x_3^1 \le x_1^2 - x_2^3 + 3.7, \qquad (x_1^1)^4 + x_1^3 x_2^1 \le x_1^2 + 2,$$

the second player

$$x_1^2 - x_2^2 \le x_2^1 + x_3^1 - x_1^3 + 7, \qquad (x_1^2 - 2)^2 + (x_2^2)^2 \le 0.75 + (x_1^1)^2,$$

while the third player has the constraints

$$x_2^3 \le x_1^1 + x_3^1 - x_1^2 + 4, \qquad 2(x_1^3)^2 - (x_2^3 - 2)^2 \le x_1^2 x_1^3 + 1.5.$$

The Jaco-Hessian of this test problem is in general neither constant nor positive (semi-) definite. Convergence of our method, using the three starting points from Table 1, occurs to the following points:

0.99987722673822	0.99973555394222	0.99987722673822
2.31570964703584	2.31634992067271	2.31570964703584
0.99989251930167	0.99976846015730	0.99989251930167
1.31499923583926	1.31481981480565	1.31499923583926
0.99989852480755	0.99993110204166	0.99989852480755
0.99992298465841	0.99983409362034	0.99992298465841
1.09709158271764	1.09703474801283	1.09709158271764

Example A.7 This problem has 4 players, each controlling 5 variables. The structure of the objective function is

$$\theta_\nu = \frac{1}{2}(x^\nu)^T A^\nu x^\nu + (x^\nu)^T B^\nu x^{-\nu}.$$

Reporting each matrix separately is very long, so we give directly the Jaco-Hessian of the problem, which is

$$\begin{array}{cccccccccccccccccccc}
110 & -3 & 22 & -14 & -27 & 1 & 9 & 19 & -2 & 23 & -7 & -20 & -4 & 22 & -19 & 22 & 3 & 13 & -12 & 18 \\
-3 & 79 & -9 & -21 & 18 & 61 & 0 & 14 & 58 & -11 & 4 & -16 & 20 & -19 & 13 & -17 & -1 & 24 & 22 & 5 \\
22 & -9 & 90 & 28 & 22 & -9 & -21 & -1 & -5 & 29 & 15 & -7 & 4 & 30 & 2 & 9 & -1 & -19 & -60 & 4 \\
-14 & -21 & 28 & 106 & 11 & -33 & -42 & 14 & 28 & -10 & 3 & 6 & 13 & 22 & -8 & 6 & -3 & 15 & -3 & 0 \\
-27 & 18 & 22 & 11 & 134 & 4 & -4 & -29 & 39 & -62 & 74 & 2 & 4 & -34 & -1 & 13 & 8 & 18 & 12 & 35 \\
1 & 61 & -9 & -33 & 4 & 119 & -14 & 12 & 12 & -6 & -23 & -14 & 16 & -4 & 15 & -2 & 8 & 16 & 9 & -9 \\
9 & 0 & -21 & -42 & -4 & -14 & 72 & -14 & 6 & -9 & 12 & 2 & -24 & 13 & 29 & 17 & 13 & -1 & 19 & 21 \\
19 & 14 & -1 & 14 & -29 & 12 & -14 & 92 & -10 & 5 & 8 & 0 & -4 & 23 & 8 & -50 & -11 & 48 & -8 & 3 \\
-2 & 58 & -5 & 28 & 39 & 12 & 6 & -10 & 124 & -39 & -4 & -16 & 24 & -18 & 26 & 4 & 13 & 29 & 43 & 23 \\
23 & -11 & 29 & -10 & -62 & -6 & -9 & 5 & -39 & 130 & -42 & -21 & 21 & 68 & -24 & -21 & -30 & -54 & -23 & 9 \\
-7 & 4 & 15 & 3 & 74 & -23 & 12 & 8 & -4 & -42 & 138 & -4 & -24 & -12 & -27 & 24 & 21 & 2 & -10 & 18 \\
-20 & -16 & -7 & 6 & 2 & -14 & 2 & 0 & -16 & -21 & -4 & 89 & -11 & -14 & -16 & -32 & -7 & -5 & 13 & -4 \\
-4 & 20 & 4 & 13 & 4 & 16 & -24 & -4 & 24 & 21 & -24 & -11 & 107 & 31 & -3 & -2 & -22 & 17 & 4 & 22 \\
22 & -19 & 30 & 22 & -34 & -4 & 13 & 23 & -18 & 68 & -12 & -14 & 31 & 116 & -1 & 5 & -18 & -16 & -43 & 27 \\
-19 & 13 & 2 & -8 & -1 & 15 & 29 & 8 & 26 & -24 & -27 & -16 & -3 & -1 & 98 & -4 & -2 & 50 & 23 & 8 \\
22 & -17 & 9 & 6 & 13 & -2 & 17 & -50 & 4 & -21 & 24 & -32 & -2 & 5 & -4 & 102 & 46 & -29 & -17 & -1 \\
3 & -1 & -1 & -3 & 8 & 8 & 13 & -11 & 13 & -30 & 21 & -7 & -22 & -18 & -2 & 46 & 110 & -16 & 24 & 12 \\
13 & 24 & -19 & 15 & 18 & 16 & -1 & 48 & 29 & -54 & 2 & -5 & 17 & -16 & 50 & -29 & -16 & 102 & 45 & 14 \\
-12 & 22 & -60 & -3 & 12 & 9 & 19 & -8 & 43 & -23 & -10 & 13 & 4 & -43 & 23 & -17 & 24 & 45 & 119 & 21 \\
18 & 5 & 4 & 0 & 35 & -9 & 21 & 3 & 23 & 9 & 18 & -4 & 22 & 27 & 8 & -1 & 12 & 14 & 21 & 59
\end{array}$$

Note that it is easy to reconstruct from this matrix all the A^ν and B^ν. For example, A^1 is the submatrix formed by the first 5 rows and columns, B^1 is the submatrix formd by the first 5 rows and columns from 6 to 20, A^2 is the submatrix formed by rows 6 to 10 and columns 6 to 10 and so on. All players have lower and uppers bounds of 1 and 5, respectively. In addition, each player has one additional linear coupling constraint given, respectively, by

$$\begin{aligned}
x_1^1 + 2x_2^1 - x_3^1 + 3x_4^1 - 4x_5^1 &\le 2 - x_2^2 + 3x_3^2, \\
-x_1^2 + 3x_2^2 - 2x_3^2 + x_4^2 + 3x_5^2 &\le 4 - x_1^3 + 3x_5^3 - 2x_3^4,
\end{aligned}$$

$$\begin{array}{rcl} -2x_1^3+3x_2^3+x_3^3-x_4^3-2x_5^3 & \leq & 4-x_1^1+4x_5^4, \\ 4x_1^4-2x_2^4-3x_3^4-6x_4^4+5x_5^4 & \leq & 3-x_1^1-x_2^1+x_1^2+x_2^2. \end{array}$$

The Jaco-Hessian of this example is a constant and positive definite matrix. Convergence of our method always occurs to the following point:

```
0.99988245735506
0.99985542095046
0.99989138444537
0.99988866261891
0.99984494662577
0.99986703246906
0.99986897052169
0.99992059068103
0.99981225576918
1.00013812006334
0.99987211313045
1.84253230021096
0.99986555230493
0.99987070302597
0.99987574778109
0.99993185140789
0.99988068741824
0.99984157413000
0.99986193178624
0.99983143496263
```

Example A.8 This is the GNEP considered in [12]. There are $N = 3$ players, each player ν controls a single variable x^ν. The GNEP is defined by the following optimization problems of the three players:

$$\begin{array}{ll|ll|ll} \min_{x^1} & -x^1 & \min_{x^2} & (x^2-0.5)^2 & \min_{x^3} & (x^3-1.5x^1)^2 \\ \text{s.t.} & x^3 \leq x^1+x^2 \leq 1, & \text{s.t.} & x^3 \leq x^1+x^2 \leq 1, & \text{s.t.} & 0 \leq x^3 \leq 2. \\ & x^1 \geq 0, & & x^2 \geq 0, & & \end{array}$$

The problem has infinitely many solutions which can be calculated analytically. They are given by

$$\big\{(\alpha, 1-\alpha, 1.5\alpha)^T \mid \alpha \in [\frac{1}{2}, \frac{2}{3}]\big\}.$$

The Jaco-Hessian of this example is constant and negative semi-definite. Convergence of our method occurs to the following points (recall that we have a failure for the third starting point):

```
0.62503131162143      0.62510047245551
0.37500031253875      0.37500003126256
```

```
0.93754579549990      0.93765059147694
```

Example A.9 This a GNEP arising in a problem of power allocation in telecommunications. The model is described in detail in [39] and represents a realistic communication system with Quality of Service constraints. There are seven players transmitting to seven different Base Stations by using K different channels. The GNEP is defined by the following optimization problems for each of the players:

$$\begin{aligned} \underset{x^\nu}{\text{minimize}} \quad & \sum_{i=1}^{K} x_i^\nu \\ \text{subject to} \quad & x^\nu \geq 0, \\ & \sum_{i=1}^{K} \log_2 \left(1 + \frac{h_i^{\nu\nu} x_i^\nu}{(\sigma_i^\nu)^2 + \sum\limits_{\mu \neq \nu} h_i^{\nu\mu} x_i^\mu} \right) \geq L^\nu. \end{aligned}$$

In both instances we considered, we set $\sigma_i^\nu = 0.3162$ for all ν and i, and in both cases the starting point was the origin.
(a) For this problem $K = 8$, $L^\nu = 8$ for all players, and the values of the coefficient h are

0.0362	0.0008	0.0018	0.0022	0.0085	0.0008	0.0060
0.2211	0.0003	0.0014	0.0074	0.0005	0.0037	0.0006
0.3356	0.0032	0.0027	0.0043	0.0007	0.0012	0.0003
0.0077	0.0039	0.0032	0.0073	0.0089	0.0006	0.0002
0.0036	0.0019	0.0032	0.0014	0.0017	0.0038	0.0046
0.1811	0.0013	0.0014	0.0024	0.0010	0.0000	0.0121
0.0442	0.0093	0.0002	0.0036	0.0096	0.0031	0.0042
0.3507	0.0017	0.0004	0.0042	0.0034	0.0061	0.0129
0.0145	0.1487	0.0001	0.0013	0.0001	0.0005	0.0008
0.0035	0.1341	0.0021	0.0001	0.0000	0.0001	0.0000
0.0042	0.3074	0.0024	0.0002	0.0001	0.0004	0.0004
0.0006	0.3358	0.0017	0.0008	0.0002	0.0004	0.0022
0.0007	0.0083	0.0045	0.0003	0.0002	0.0000	0.0019
0.0024	0.2214	0.0011	0.0000	0.0001	0.0006	0.0016
0.0034	0.1261	0.0079	0.0002	0.0002	0.0007	0.0001
0.0032	0.0288	0.0059	0.0004	0.0001	0.0001	0.0014
0.0040	0.0051	0.1475	0.0042	0.0002	0.0000	0.0005
0.0030	0.0002	0.0300	0.0003	0.0000	0.0001	0.0001
0.0046	0.0014	0.2207	0.0089	0.0000	0.0001	0.0003
0.0059	0.0005	0.3028	0.0024	0.0002	0.0001	0.0002
0.0037	0.0002	0.0828	0.0008	0.0003	0.0001	0.0001
0.0023	0.0003	0.0167	0.0009	0.0014	0.0001	0.0002
0.0009	0.0001	0.0956	0.0001	0.0005	0.0000	0.0003
0.0006	0.0028	0.0566	0.0018	0.0000	0.0001	0.0002
0.0005	0.0002	0.0004	0.0991	0.0057	0.0000	0.0003
0.0000	0.0000	0.0016	0.0377	0.0003	0.0002	0.0003
0.0014	0.0001	0.0002	0.8100	0.0035	0.0001	0.0000
0.0019	0.0002	0.0016	0.3608	0.0141	0.0000	0.0000
0.0013	0.0005	0.0014	0.0617	0.0014	0.0006	0.0000
0.0016	0.0003	0.0004	0.0723	0.0016	0.0005	0.0001
0.0007	0.0000	0.0028	0.3314	0.0126	0.0002	0.0000
0.0011	0.0001	0.0002	0.0953	0.0026	0.0000	0.0001
0.0001	0.0000	0.0001	0.0009	0.0993	0.0015	0.0001
0.0002	0.0003	0.0003	0.0013	0.1702	0.0210	0.0000
0.0007	0.0000	0.0002	0.0024	1.1062	0.0033	0.0004
0.0012	0.0002	0.0006	0.0014	0.2817	0.0091	0.0004
0.0003	0.0003	0.0004	0.0008	0.5208	0.0024	0.0001
0.0001	0.0001	0.0001	0.0006	0.3959	0.0010	0.0001
0.0004	0.0001	0.0011	0.0009	0.1057	0.0029	0.0004
0.0000	0.0001	0.0000	0.0020	0.0932	0.0058	0.0006
0.0002	0.0010	0.0006	0.0001	0.0001	0.3905	0.0028
0.0004	0.0001	0.0000	0.0001	0.0009	0.1322	0.0032
0.0001	0.0009	0.0001	0.0003	0.0018	0.3691	0.0130
0.0004	0.0007	0.0003	0.0001	0.0018	0.1142	0.0022
0.0009	0.0010	0.0002	0.0001	0.0019	0.1127	0.0046
0.0017	0.0014	0.0000	0.0001	0.0022	0.1665	0.0015
0.0009	0.0001	0.0000	0.0002	0.0015	0.0066	0.0042
0.0004	0.0006	0.0005	0.0001	0.0022	0.2389	0.0026
0.0008	0.0054	0.0004	0.0001	0.0001	0.0040	0.1110
0.0005	0.0013	0.0008	0.0004	0.0000	0.0006	0.0834

0.0033	0.0063	0.0001	0.0004	0.0000	0.0013	0.4948
0.0012	0.0079	0.0000	0.0000	0.0001	0.0029	0.0355
0.0002	0.0095	0.0002	0.0002	0.0001	0.0003	0.3045
0.0010	0.0018	0.0000	0.0002	0.0003	0.0009	0.5359
0.0003	0.0045	0.0002	0.0001	0.0002	0.0007	0.0328
0.0005	0.0035	0.0006	0.0000	0.0008	0.0008	0.2950

where the value of $h_i^{\nu\mu}$ is reported in the ν-th column, in position $(\mu-1)K+i$.
(b) For this problem $K = 16$, $L^\nu = 16$ for all players, and the values of the coefficient h are

0.0129	0.0010	0.0015	0.0008	0.0005	0.0088	0.0048
0.0037	0.0062	0.0020	0.0044	0.0043	0.0040	0.0029
0.0514	0.0114	0.0024	0.0094	0.0063	0.0057	0.0045
0.1382	0.0087	0.0096	0.0113	0.0040	0.0144	0.0114
0.1824	0.0026	0.0180	0.0068	0.0054	0.0140	0.0157
0.1193	0.0002	0.0135	0.0007	0.0094	0.0043	0.0132
0.0290	0.0002	0.0033	0.0019	0.0116	0.0016	0.0089
0.0188	0.0020	0.0016	0.0075	0.0192	0.0065	0.0105
0.0550	0.0074	0.0038	0.0070	0.0270	0.0076	0.0157
0.0642	0.0134	0.0024	0.0020	0.0194	0.0056	0.0136
0.0495	0.0166	0.0004	0.0019	0.0052	0.0058	0.0041
0.0318	0.0160	0.0009	0.0042	0.0017	0.0067	0.0001
0.0287	0.0098	0.0050	0.0027	0.0030	0.0050	0.0027
0.0585	0.0025	0.0102	0.0004	0.0010	0.0020	0.0021
0.0843	0.0001	0.0092	0.0002	0.0002	0.0027	0.0000
0.0586	0.0002	0.0032	0.0001	0.0002	0.0077	0.0027
0.0028	0.0288	0.0208	0.0004	0.0000	0.0002	0.0011
0.0066	0.0136	0.0127	0.0000	0.0001	0.0000	0.0006
0.0088	0.0500	0.0094	0.0001	0.0002	0.0003	0.0017
0.0039	0.0251	0.0125	0.0000	0.0002	0.0006	0.0010
0.0016	0.0091	0.0147	0.0000	0.0001	0.0003	0.0003
0.0061	0.1168	0.0130	0.0005	0.0000	0.0001	0.0005
0.0098	0.1155	0.0088	0.0009	0.0001	0.0002	0.0002
0.0121	0.0029	0.0044	0.0006	0.0001	0.0001	0.0008
0.0142	0.1913	0.0019	0.0001	0.0003	0.0000	0.0021
0.0106	0.4811	0.0009	0.0001	0.0004	0.0001	0.0011
0.0035	0.3142	0.0017	0.0002	0.0004	0.0003	0.0002
0.0004	0.0131	0.0044	0.0000	0.0003	0.0006	0.0016
0.0002	0.1289	0.0049	0.0002	0.0002	0.0007	0.0014
0.0009	0.3551	0.0038	0.0009	0.0001	0.0004	0.0002
0.0031	0.2986	0.0099	0.0014	0.0001	0.0001	0.0017
0.0034	0.1287	0.0202	0.0010	0.0000	0.0003	0.0029
0.0023	0.0011	0.0609	0.0017	0.0002	0.0001	0.0002
0.0028	0.0021	0.0588	0.0012	0.0001	0.0004	0.0002
0.0025	0.0071	0.0554	0.0028	0.0001	0.0004	0.0001
0.0008	0.0096	0.1574	0.0099	0.0000	0.0002	0.0000
0.0007	0.0045	0.2024	0.0192	0.0002	0.0001	0.0000
0.0036	0.0003	0.0726	0.0165	0.0005	0.0000	0.0001
0.0051	0.0025	0.0497	0.0032	0.0004	0.0000	0.0004
0.0024	0.0038	0.1986	0.0029	0.0004	0.0001	0.0004
0.0000	0.0016	0.2281	0.0205	0.0005	0.0002	0.0002
0.0008	0.0005	0.1334	0.0312	0.0004	0.0004	0.0000
0.0012	0.0007	0.0762	0.0231	0.0003	0.0006	0.0001
0.0006	0.0005	0.0209	0.0103	0.0004	0.0004	0.0001
0.0021	0.0005	0.0115	0.0042	0.0002	0.0001	0.0001
0.0043	0.0004	0.1096	0.0032	0.0000	0.0003	0.0001
0.0041	0.0005	0.1429	0.0034	0.0002	0.0004	0.0001
0.0026	0.0012	0.0759	0.0028	0.0004	0.0001	0.0002
0.0006	0.0001	0.0016	0.0492	0.0066	0.0011	0.0002
0.0002	0.0002	0.0014	0.0128	0.0003	0.0008	0.0002
0.0007	0.0002	0.0013	0.1154	0.0011	0.0000	0.0001
0.0005	0.0001	0.0018	0.1459	0.0010	0.0003	0.0001
0.0011	0.0000	0.0010	0.0509	0.0017	0.0005	0.0005
0.0016	0.0000	0.0001	0.0136	0.0077	0.0001	0.0006
0.0017	0.0000	0.0000	0.0487	0.0095	0.0001	0.0002
0.0014	0.0000	0.0003	0.0591	0.0041	0.0004	0.0000
0.0003	0.0002	0.0017	0.0434	0.0016	0.0002	0.0003
0.0003	0.0004	0.0030	0.0570	0.0015	0.0001	0.0005
0.0027	0.0004	0.0016	0.1731	0.0001	0.0004	0.0003
0.0030	0.0001	0.0005	0.3232	0.0019	0.0003	0.0000
0.0007	0.0000	0.0025	0.2794	0.0055	0.0001	0.0001
0.0019	0.0001	0.0030	0.0914	0.0076	0.0002	0.0002
0.0051	0.0002	0.0008	0.0519	0.0119	0.0000	0.0001
0.0039	0.0001	0.0004	0.1091	0.0139	0.0004	0.0000
0.0000	0.0001	0.0000	0.0003	0.0301	0.0023	0.0003
0.0003	0.0001	0.0000	0.0010	0.0167	0.0070	0.0002
0.0008	0.0000	0.0001	0.0018	0.0398	0.0089	0.0009
0.0007	0.0001	0.0003	0.0021	0.0606	0.0049	0.0021
0.0004	0.0003	0.0005	0.0014	0.0857	0.0010	0.0015
0.0002	0.0001	0.0002	0.0003	0.1207	0.0012	0.0003
0.0002	0.0001	0.0001	0.0002	0.0936	0.0021	0.0005
0.0005	0.0008	0.0003	0.0007	0.0217	0.0026	0.0007
0.0004	0.0011	0.0005	0.0008	0.0188	0.0035	0.0002
0.0001	0.0005	0.0003	0.0003	0.0808	0.0027	0.0003
0.0004	0.0000	0.0001	0.0002	0.1234	0.0007	0.0003
0.0007	0.0001	0.0000	0.0006	0.1248	0.0013	0.0000
0.0004	0.0000	0.0002	0.0007	0.0850	0.0029	0.0011
0.0002	0.0000	0.0003	0.0003	0.0287	0.0024	0.0022

0.0005	0.0001	0.0002	0.0000	0.0210	0.0009	0.0015
0.0003	0.0001	0.0000	0.0000	0.0432	0.0002	0.0005
0.0001	0.0002	0.0001	0.0002	0.0008	0.1341	0.0150
0.0004	0.0003	0.0001	0.0002	0.0027	0.0507	0.0035
0.0006	0.0002	0.0000	0.0001	0.0016	0.0187	0.0001
0.0004	0.0001	0.0000	0.0001	0.0001	0.0037	0.0008
0.0003	0.0003	0.0001	0.0001	0.0020	0.0046	0.0019
0.0007	0.0005	0.0000	0.0001	0.0032	0.0153	0.0121
0.0005	0.0006	0.0000	0.0000	0.0014	0.0351	0.0198
0.0001	0.0005	0.0000	0.0000	0.0001	0.0956	0.0113
0.0006	0.0005	0.0001	0.0003	0.0001	0.1673	0.0011
0.0011	0.0007	0.0002	0.0003	0.0006	0.1522	0.0008
0.0008	0.0009	0.0002	0.0002	0.0020	0.0527	0.0034
0.0006	0.0007	0.0002	0.0002	0.0022	0.0005	0.0057
0.0006	0.0003	0.0001	0.0004	0.0004	0.0801	0.0051
0.0003	0.0000	0.0000	0.0003	0.0003	0.2297	0.0004
0.0000	0.0000	0.0000	0.0001	0.0011	0.3127	0.0043
0.0001	0.0000	0.0001	0.0001	0.0003	0.2574	0.0162
0.0009	0.0038	0.0000	0.0000	0.0003	0.0018	0.1630
0.0024	0.0009	0.0000	0.0001	0.0006	0.0052	0.3698
0.0035	0.0016	0.0001	0.0001	0.0007	0.0053	0.2515
0.0021	0.0019	0.0002	0.0001	0.0003	0.0016	0.0571
0.0003	0.0017	0.0002	0.0002	0.0000	0.0005	0.0774
0.0004	0.0016	0.0003	0.0003	0.0001	0.0019	0.0623
0.0013	0.0018	0.0007	0.0001	0.0001	0.0012	0.0244
0.0015	0.0054	0.0012	0.0001	0.0001	0.0000	0.2083
0.0015	0.0094	0.0010	0.0002	0.0000	0.0004	0.3246
0.0009	0.0066	0.0004	0.0003	0.0001	0.0007	0.1571
0.0001	0.0011	0.0000	0.0003	0.0007	0.0005	0.0440
0.0004	0.0003	0.0000	0.0003	0.0012	0.0004	0.0953
0.0007	0.0022	0.0000	0.0002	0.0009	0.0003	0.1190
0.0002	0.0045	0.0000	0.0001	0.0004	0.0009	0.1166
0.0001	0.0082	0.0001	0.0001	0.0001	0.0015	0.0703
0.0004	0.0085	0.0001	0.0000	0.0001	0.0008	0.0079

For these two problems, we do not report the solutions found; they are available on request from the authors.

Example A.10 This is a series of problems based on the famous Arrow and Debreu model of a competitive economy, see [1]. In this game there are F firms, C consumers and one market player (hence there is a total of $F + C + 1$ players). The economy is based on the production and exchange of P goods. The market player sets the (normalized) prices p with a "market clearing" problem, the firms maximize their profit deciding how much to produce while the consumers decide how much to own of each product in order to maximize their utility functions. In our setting, the problem of the firms, controlling the variables $y^j \in \mathbb{R}^P$ is, for $j = 1, \dots, F$,

$$\begin{array}{ll} \max_{y^j} & p^T y^j \\ \text{s.t.} & y^j \geq 0, \\ & \sum_{k=1}^{P} (y_k^j)^2 \leq j * 10. \end{array}$$

The i-th consumption player controls the variables $x^i \in \mathbb{R}^P$, and his problem is

$$\begin{array}{ll} \max_{x^i} & u_i(x^i) \\ \text{s.t.} & x^i \geq 0, \\ & p^T x^i \leq p^T \xi^i, \end{array}$$

where $\xi^i \in \mathbb{R}_+^P$ is an initial endowment of goods. Finally, the market player problem is

$$\begin{array}{ll} \max_{p} & p^T \left(\sum_{i=1}^{C} x^i - \sum_{j=1}^{F} y^j - \sum_{i=1}^{C} \xi^i \right) \\ \text{s.t.} & p \geq 0, \\ & \sum_{h=1}^{P} p_h = 1. \end{array}$$

In all cases, we started the algorithm from the feasible point $y^j = 0$, $x^i = 0$ and $p_k = 1/P$. In order to define the problems, we must specify F, C, P, the utility functions u_i and the initial endowments ξ^i.
(a) $F = 2$, $C = 5$, $P = 3$. The utility functions are quadratic and convex and have the form

$$u_i(x^i) = -\frac{1}{2}(x^i)^T Q^i x^i + (b^i)^T x^i,$$

where

$$Q^i = \begin{pmatrix} 6 & -2 & 5 \\ -2 & 6 & -7 \\ 5 & -7 & 20 \end{pmatrix}, \qquad b^i = \begin{pmatrix} 30+i+F \\ 30+i+F \\ 30+i+F \end{pmatrix}, \qquad i = 1, 2,$$

while

$$Q^i = \begin{pmatrix} 6 & 1 & 0 \\ 1 & 7 & -5 \\ 0 & -5 & 7 \end{pmatrix}, \qquad b^i = \begin{pmatrix} 30+(i+F)*2 \\ 30+(i+F)*2 \\ 30+(i+F)*2 \end{pmatrix}, \qquad i = 3, 4, 5.$$

The initial endowments are

$$\xi^i = \begin{pmatrix} 2 \\ 3 \\ 4 \end{pmatrix}, \quad i = 1, 2, \qquad \xi^i = \begin{pmatrix} 6 \\ 5 \\ 4 \end{pmatrix}, \quad i = 3, 4, 5.$$

(b) $F = 4$, $C = 20$, $P = 5$. The utility functions are of logarithmic type and given by:

$$u^i(x^i) = \sum_{k=1}^{P} (a_k + i + F) * \log(x^i_k + b_k + 2 * (i + F)), \qquad i = 1, \ldots, C/2,$$

and

$$u^i(x^i) = \sum_{k=1}^{P} (c_k + i + F) * \log(x^i_k + d_k + i + F), \qquad i = C/2 + 1, \ldots, C,$$

with

$$a = \begin{pmatrix} 1 \\ 2 \\ 4 \\ 6 \\ 8 \end{pmatrix}, \quad b = \begin{pmatrix} 20 \\ 30 \\ 30 \\ 40 \\ 50 \end{pmatrix}, \quad c = \begin{pmatrix} 10 \\ 6 \\ 4 \\ 10 \\ 1 \end{pmatrix}, \quad d = \begin{pmatrix} 50 \\ 40 \\ 30 \\ 20 \\ 20 \end{pmatrix},$$

while the initial endowments are

$$\xi^i = \begin{pmatrix} 2 \\ 3 \\ 4 \\ 1 \\ 6 \end{pmatrix}, \quad i = 1, \ldots, C/2, \qquad \xi^i = \begin{pmatrix} 6 \\ 5 \\ 4 \\ 3 \\ 2 \end{pmatrix}, \quad i = C/2 + 1, \ldots, C.$$

Note that the logarithms are not defined if the variables become too negative. However, we took the terms b and d with positive component large enough to make this possibility unlikely.

(c) $F = 6$, $C = 30$, $P = 6$. The utility functions are again convex and quadratic. We first set A equal to

```
 68.22249416536778  12.12481199690621  -8.35496210217478  -6.81177486915109  -4.66752803051747  3.64100170417482
 12.12481199690621  53.51450780426463 -21.77618227261339 -15.00376305863444  -0.11788350473544  2.03354709400720
 -8.35496210217478 -21.77618227261339  35.44033408387684   4.35160649036518  19.17472558234163 -3.40090742729160
 -6.81177486915109 -15.00376305863444   4.35160649036518  52.25155022199242  -5.99490328518247 20.40443259092577
 -4.66752803051747  -0.11788350473544  19.17472558234163  -5.99490328518247  23.32798561358070 -3.58535668529727
  3.64100170417482   2.03354709400720  -3.40090742729160  20.40443259092577  -3.58535668529727 10.21258119890765
```

and B equal to

```
 61.74633559943146 -23.83006225091380  16.78581949473039  14.42073900860500  -2.75188745616575  13.44307656650567
-23.83006225091380  37.64246654306209  -3.76510322128227  16.32022449045404 -39.90743633716275  11.38657250296817
 16.78581949473039  -3.76510322128227  53.34843665848310   4.60388415537161 -23.04611587657949 -25.31392346426841
 14.42073900860500  16.32022449045404   4.60388415537161  40.69699687713468 -30.78019133996427  17.08866411420883
 -2.75188745616575 -39.90743633716275 -23.04611587657949 -30.78019133996427  66.22678445157413 -12.28091080313848
 13.44307656650567  11.38657250296817 -25.31392346426841  17.08866411420883 -12.28091080313848  41.37849544246254
```

With reference to the notation in (a), we then have

$$Q^i = A,\ b^i = \begin{pmatrix} 50+i+F \\ 60+i+F \\ 70+i+F \\ 60+i+F \\ 60+i+F \\ 50+i+F \end{pmatrix},\ i = 1,\dots,C/2,\ \ Q^i = B, \quad b^i = \begin{pmatrix} 50+2*(i+F) \\ 60+2*(i+F) \\ 50+2*(i+F) \\ 70+2*(i+F) \\ 70+2*(i+F) \\ 60+2*(i+F) \end{pmatrix},\ i = C/2{+}1,\dots,C.$$

The initial endowments are

$$\xi^i = \begin{pmatrix} 2 \\ 3 \\ 4 \\ 1 \\ 6 \\ 1 \end{pmatrix}, \quad i = 1,\dots,C/2, \qquad \xi^i = \begin{pmatrix} 6 \\ 5 \\ 4 \\ 3 \\ 2 \\ 8 \end{pmatrix}, \quad i = C/2+1,\dots,C.$$

(d) $F = 6$, $C = 30$, $P = 10$. The structure of the problem is similar to that in case (b), but with the following data:

$$a = \begin{pmatrix} 1 \\ 2 \\ 4 \\ 6 \\ 8 \\ 7 \\ 8 \\ 10 \\ 1 \\ 5 \end{pmatrix}, \quad b = \begin{pmatrix} 50 \\ 60 \\ 70 \\ 60 \\ 60 \\ 50 \\ 50 \\ 80 \\ 60 \\ 70 \end{pmatrix}, \quad c = \begin{pmatrix} 10 \\ 6 \\ 4 \\ 10 \\ 1 \\ 2 \\ 6 \\ 4 \\ 9 \\ 4 \end{pmatrix}, \quad d = \begin{pmatrix} 50 \\ 60 \\ 50 \\ 70 \\ 70 \\ 60 \\ 50 \\ 50 \\ 80 \\ 50 \end{pmatrix};$$

while the initial endowments are

$$\xi^i = \begin{pmatrix} 2 \\ 3 \\ 4 \\ 1 \\ 6 \\ 1 \\ 3 \\ 6 \\ 2 \\ 10 \end{pmatrix}, \quad i = 1,\dots,C/2, \qquad \xi^i = \begin{pmatrix} 6 \\ 5 \\ 4 \\ 3 \\ 2 \\ 8 \\ 4 \\ 6 \\ 2 \\ 0 \end{pmatrix}, \quad i = C/2+1,\dots,C.$$

(e) $F = 7$, $C = 40$, $P = 12$. The structure of the problem is similar to that in case (b), but with the following data:

$$a = \begin{pmatrix} 1 \\ 2 \\ 4 \\ 6 \\ 8 \\ 7 \\ 8 \\ 10 \\ 1 \\ 5 \\ 2 \\ 4 \end{pmatrix}, \quad b = \begin{pmatrix} 50 \\ 60 \\ 70 \\ 60 \\ 60 \\ 50 \\ 50 \\ 80 \\ 60 \\ 70 \\ 70 \\ 80 \end{pmatrix}, \quad c = \begin{pmatrix} 10 \\ 6 \\ 4 \\ 10 \\ 1 \\ 2 \\ 6 \\ 4 \\ 9 \\ 4 \\ 5 \\ 1 \end{pmatrix}, \quad d = \begin{pmatrix} 50 \\ 60 \\ 50 \\ 70 \\ 70 \\ 60 \\ 50 \\ 50 \\ 80 \\ 50 \\ 60 \\ 70 \end{pmatrix};$$

while the initial endowments are

$$\xi^i = \begin{pmatrix} 2 \\ 3 \\ 4 \\ 1 \\ 6 \\ 1 \\ 3 \\ 6 \\ 2 \\ 10 \\ 3 \\ 4 \end{pmatrix}, \quad i = 1, \ldots, C/2, \qquad \xi^i = \begin{pmatrix} 6 \\ 5 \\ 4 \\ 3 \\ 2 \\ 8 \\ 4 \\ 6 \\ 2 \\ 0 \\ 6 \\ 0 \end{pmatrix}, \quad i = C/2+1, \ldots, C.$$

For these five problems, we do not report the solutions found; they are available on request from the authors.

A.2 Jointly Convex Problems

Here we give the details of the test examples for the jointly convex GNEPs. All examples are taken from the literature. Some examples have known solutions which we then state in the description of the corresponding problems. For those examples which have no known (analytical) solution, we provide an approximate solution that has been computed by the locally quadratically convergent method from [27], hence the numerical solution should have a high accuracy. For a jointly convex GNEP, there also exists the notion of a *normalized equilibrium*, cf. [43, 31, 24]. The normalized equilibria form a subset of all solutions and are often the solutions which can be computed by existing methods.

Example A.11 This test problem is taken from [12]. There are two players, each player ν has a one-dimensional decision variable $x^\nu \in \mathbb{R}$. The optimization problems of the two players are given by

$$\begin{array}{rc|rc} \min_{x^1} & (x^1-1)^2 & \min_{x^2} & \left(x^2 - \frac{1}{2}\right)^2 \\ \text{s.t.} & x^1 + x^2 \leq 1 & \text{s.t.} & x^1 + x^2 \leq 1. \end{array}$$

This GNEP has infinitely many solutions $\{(\alpha, 1-\alpha) \mid \alpha \in [1/2, 1]\}$, but only one normalized equilibrium at $\bar{x} := \left(\frac{3}{4}, \frac{1}{4}\right)^T$. The Jaco–Hessian is symmetric positive definite everywhere. Our method converges to the following point which is (numerically) the normalized equilibrium:

```
0.75002417863494
0.25002417863744
```

Example A.12 This is a duopoly model with $N = 2$ players taken from [31]. Each player ν controls one variable $x^\nu \in \mathbb{R}$. The objective functions are given by

$$\theta_\nu(x) := x^\nu\big(\rho(x^1 + x^2) + \lambda - d\big) \quad \text{for } \nu = 1, 2,$$

and the constraints are

$$x^\nu \in [-10, +10] \quad \text{for } \nu = 1, 2.$$

Note that this is a standard NEP. For the parameter values $d = 20, \lambda = 4, \rho = 1$, this example has the (unconstrained) solution

$$\bar{x} := \big(\frac{16}{3}, \frac{16}{3}\big)^T.$$

The Jaco-Hessian of this example is constant and symmetric positive definite. Algorithm 3.3 converges to the following point:

```
5.33331555561568
5.33331555561568
```

Example A.13 This is the river basin pollution example, also taken from [31]. There are three players ν, each of them having control of a single decision variable $x^\nu \in \mathbb{R}$. The objective functions are defined by

$$\theta_\nu(x) := x^\nu\big(c_{1\nu} + c_{2\nu}x^\nu - d_1 + d_2(x^1 + x^2 + x^3)\big) \quad \forall \nu = 1, 2, 3.$$

The joint constraints are given by

$$\begin{aligned} u_{11}e_1x^1 + u_{21}e_2x^2 + u_{31}e_3x^3 &\leq K_1, \\ u_{12}e_1x^1 + u_{22}e_2x^2 + u_{32}e_3x^3 &\leq K_2. \end{aligned}$$

We use the parameters $K_1 = K_2 = 100, d_1 = 3, d_2 = 0.01$ and $c_{1\nu}, c_{2\nu}, e_\nu, u_{\nu1}, u_{\nu2}$ as specified in Table 3.

player ν	1	2	3	4	5
1	0.10	0.01	0.50	6.5	4.583
2	0.12	0.05	0.25	5.0	6.250
3	0.15	0.01	0.75	5.5	3.750

Table 3: Parameters for the river basin pollution game from Example A.13

A solution of this problem is approximately given by the vector

$$\bar{x} = \big(21.14671036, 16.02782075, 2.7242447250\big)^T.$$

The Jaco-Hessian is constant and symmetric positive definite. Algorithm 3.3 applied to this example converges to the following vector:

```
21.14480155732168
16.02785326538717
 2.72597096564384
```

Example A.14 This test problem is an internet switching model introduced by Kesselman et al. [29], also analyzed by Facchinei et al. [12]. There are N players, each player having a single variable $x^\nu \in \mathbb{R}$. The utility functions are given by

$$\theta_\nu(x) := \frac{-x^\nu}{x^1 + \ldots + x^N}\left(1 - \frac{x^1 + \ldots + x^N}{B}\right) \quad \forall \nu = 1, \ldots, N$$

for some constant B. The constraints are

$$x^1 + \ldots + x^N \leq B, \quad x^\nu \geq l_\nu$$

for some lower bounds $l_\nu \geq 0$. Note that [29] uses the lower bound $l_\nu = 0$, however, in order to avoid possible divisions by zero, we take the lower bounds $l_\nu := 0.01$ for all $\nu = 1, \ldots, N$ which does not change the solution for our particular instance of this test example. This particular instance uses $N = 10$ players and the parameter $B = 1$. The problem has a known solution at

$$\bar{x} = \left(\frac{9}{100}, \ldots, \frac{9}{100}\right)^T.$$

The Jaco-Hessian is nonsingular according to [12] and non symmetric; furthermore, it is also positive definite at the solution. Starting with $x^0 := (0, \ldots, 0)^T$, Algorithm 3.3 converges to the following point:

```
0.08999991899425
0.08999991899426
0.08999991899425
0.08999991899425
0.08999991899425
0.08999991899425
0.08999991899425
0.08999991899426
0.08999991899425
0.08999991899425
```

Example A.15 Here we describe the electricity market problem suggested by Contreras et al. [6]. This model involves three players with player 1 having a single variable $x^1 \in \mathbb{R}$, player 2 controlling a two-dimensional vector $x^2 = (x_1^2, x_2^2) \in \mathbb{R}^2$, and player 3 having a three-dimensional decision variable $x^3 = (x_1^3, x_2^3, x_3^3) \in \mathbb{R}^3$. For simplicity of notation, we write

$$x = (x_1^1, x_1^2, x_2^2, x_1^3, x_2^3, x_3^3)^T =: (x_1, x_2, x_3, x_4, x_5, x_6).$$

Then the objective functions can be written as

$$\begin{aligned}
\theta_1(x) &:= \psi(x)x_1 + \big(\frac{1}{2}c_1x_1^2 + d_1x_1 + e_1\big), \\
\theta_2(x) &:= \psi(x)(x_2 + x_3) + \sum_{i=2}^{3} \big(\frac{1}{2}c_ix_i^2 + d_ix_i + e_i\big), \\
\theta_3(x) &:= \psi(x)(x_4 + x_5 + x_6) + \sum_{i=4}^{6} \big(\frac{1}{2}c_ix_i^2 + d_ix_i + e_i\big)
\end{aligned}$$

with $\psi(x) := 2(x_1 + \ldots + x_6) - 378.4$ and the parameters c_i, d_i, e_i as given in Table 4.

component i	1	2	3	4	5	6
c_i	0.04	0.035	0.125	0.0166	0.05	0.05
d_i	2	1.75	1	3.25	3	3
e_i	0	0	0	0	0	0

Table 4: Parameters corresponding to the electricity market model from Example A.15

The feasible set is given by

$$x_1 \leq 80,\ x_2 \leq 80,\ x_3 \leq 50,\ x_4 \leq 55,\ x_5 \leq 30,\ x_6 \leq 40,\ x_i \geq 0\ \forall i = 1, \ldots, 6,$$

hence this GNEP is actually a standard NEP. An approximate solution of this problem is

$$\bar{x} = (46.661622, 32.154050, 15.003109, 22.107198, 12.339584, 12.339584)^T.$$

The Jaco-Hessian of this example is a constant and symmetric positive definite matrix. Algorithm 3.3 being applied to this example using the starting point $x^0 := (0, \ldots, 0)^T$ converges to the following vector:

```
46.66150692423980
32.15293850189938
15.00419467998705
22.10485810522063
12.34076570922471
12.34076570922471
```

Example A.16 Here we describe the details of an oligopoly model as used in [36]. It is a GNEP-variant of a standard NEP considered in [32]. There are N players. Each player ν

controls a single variable $x^\nu \in \mathbb{R}$, the objective functions are given by

$$\theta_\nu(x) := f_\nu(x^\nu) - 5000^{1/\gamma} x^\nu (x^1 + \ldots + x^N)^{-1/\gamma} \quad \forall \nu = 1, \ldots, N$$

with

$$f_\nu(x^\nu) := c_\nu x^\nu + \frac{\delta_\nu}{1+\delta_\nu} K_\nu^{-1/\delta_\nu} (x^\nu)^{(1+\delta_\nu)/\delta_\nu} \quad \forall \nu = 1, \ldots, N.$$

The constraints are

$$x^1 + \ldots + x^N \leq P, \quad x^\nu \geq 0 \ \forall \nu = 1, \ldots, N.$$

Here $\gamma, P, c_\nu, K_\nu, \delta_\nu$ are given parameters. For the concrete test runs, we follow [36] and use $N = 5$ players, set $\gamma = 1.1$ and take the parameters c_ν, K_ν, δ_ν as specified in Table 5.

ν	1	2	3	4	5
c_ν	10	8	6	4	2
K_ν	5	5	5	5	5
δ_ν	1.2	1.1	1.0	0.9	0.8

Table 5: Parameters corresponding to the oligopoly model from Example A.16

For the parameter P (representing an upper bound for the total production activity of all players), different values are used. The standard values together with the corresponding (approximate) solutions are given in Table 6 which correspond to a normalized equilibrium. Note that these solutions are somewhat different from those reported in [36]. In fact, the problem seems to have many solutions, but those from Table 6 might be the only normalized equilibria.

P	$\bar{x}$
75	$(10.403965, 13.035817, 15.407354, 17.381556, 18.771308)^T$
100	$(14.050088, 17.798379, 20.907187, 23.111429, 24.132916)^T$
150	$(23.588779, 28.684248, 32.021533, 33.287258, 32.418182)^T$
200	$(35.785329, 40.748959, 42.802485, 41.966381, 38.696846)^T$

Table 6: Approximate solutions of the oligopoly model from Example A.16 for different values of P

The Jaco-Hessian of this example is non-constant. Evaluated at the four different solution points, the eigenvalues of its symmetric part all positive so that we can conclude that the Jaco-Hessian is positive definite at the solutions. Application of Algorithm 3.3 with the starting point $x^0 := (10, \ldots, 10)^T$ suggested in [36] using the four different values $P = 75, 100, 150, 200$, respectively, gives the following approximate solutions:

```
10.40385838022815  14.05009569254498  23.58870405660235  35.78534478322666
13.03589735757035  17.79839784112484  28.68433662443821  40.74896955849369
15.40740737819953  20.90720382090469  32.02151725264314  42.80249145810686
17.38156802485601  23.11144740620916  33.28727604396901  41.96639063661255
```

```
18.77134655563816   24.13291790632350   32.41822391278223   38.69685028772376
```

Example A.17 This GNEP is taken from [33] and consists of $N = 2$ players. The first player controls the two-dimensional decision variable $x^1 = (x^1_1, x^1_2)^T =: (x_1, x_2) \in \mathbb{R}^2$, whereas the second player has a one-dimensional decision variable $x^2 =: x_3 \in \mathbb{R}$. The optimization problems of both players are given by

$$\begin{array}{ll|ll}
\min_{x_1,x_2} & x_1^2 + x_1x_2 + x_2^2 + (x_1 + x_2)x_3 - 25x_1 - 38x_2 & \min_{x_3} & x_3^2 + (x_1 + x_2)x_3 - 25x_3 \\
\text{s.t.} & x_1, x_2 \geq 0, & \text{s.t.} & x_3 \geq 0, \\
 & x_1 + 2x_2 - x_3 \leq 14, & & x_1 + 2x_2 - x_3 \leq 14, \\
 & 3x_1 + 2x_2 + x_3 \leq 30, & & 3x_1 + 2x_2 + x_3 \leq 30.
\end{array}$$

This problem has an infinite number of solutions given by

$$\big\{ (\alpha, 11 - \alpha, 8 - \alpha)^T \mid \alpha \in [0, 2] \big\}.$$

For $\alpha = 0$, we get a normalized equilibrium. The Jaco-Hessian is constant and (everywhere) symmetric and positive definite. We use the starting point $x^0 := (0, 0, 0)^T$ for initializing Algorithm 3.3, and obtain convergence to the following vector:

```
 0.00000737
11.00002440
 7.99997560
```

Example A.18 Here we give the details of an electricity market model originally proposed in [38] and further discussed in [34]. The formulation here is taken from the latter reference. There are two players (companies). Each player has an electricity plant in two out of three possible regions which are represented by the nodes of a graph. The goal is to maximize the profit of each company. The overall model has 18 variables, however, below we only present the reduced formulation with 12 variables. The reduction comes from the fact that both companies have plants on only 2 of the 3 nodes, so that the constraints of the overall model immediately imply that six variables are equal to zero.

Let us use the abbreviations

$$\begin{aligned}
S_1 &:= 40 - \frac{40}{500}\big(x_1 + x_4 + x_7 + x_{10}\big), \\
S_2 &:= 35 - \frac{35}{400}\big(x_2 + x_5 + x_8 + x_{11}\big), \\
S_3 &:= 32 - \frac{32}{600}\big(x_3 + x_6 + x_9 + x_{12}\big).
\end{aligned}$$

The problem for player 1 is then as follows: Player 1 has the 6 variables $(x_1, \dots, x_6) := (x^1_1, \dots, x^1_6)$ and minimizes the following objective function:

$$\theta_1(x) := (15 - S_1)(x_1 + x_4) + (15 - S_2)(x_2 + x_5) + (15 - S_3)(x_3 + x_6).$$

Furthermore, player 1 has the following constraints: Nonnegativity constraints

$$x_1, \dots, x_6 \geq 0,$$

capacity constraints

$$\begin{aligned} x_1 + x_2 + x_3 &\leq 100, \\ x_4 + x_5 + x_6 &\leq 50, \end{aligned}$$

and the joint constraints

$$S_j - S_i \leq 1 \quad \forall i, j = 1, 2, 3, \quad \text{with} \quad i \neq j.$$

Player 2 has a similar problem: His variables are $(x_7, \dots, x_{12}) := (x_1^2, \dots, x_6^2)$, he minimizes his objective function

$$\theta_2(x) := (15 - S_1)(x_7 + x_{10}) + (15 - S_2)(x_8 + x_{11}) + (15 - S_3)(x_9 + x_{12}).$$

Furthermore, player 2 has the following constraints: Nonnegativity constraints

$$x_7, \dots, x_{12} \geq 0,$$

capacity constraints

$$\begin{aligned} x_7 + x_8 + x_9 &\leq 100, \\ x_{10} + x_{11} + x_{12} &\leq 50, \end{aligned}$$

and the joint constraints (as for player 1)

$$S_j - S_i \leq 1 \quad \forall i, j = 1, 2, 3, \quad \text{with} \quad i \neq j.$$