Chapter 4

Improved Nonmonotone Adaptive Trust-region Method to solve Generalized Nash Equilibrium Problems

4.1 Introduction

Generalized Nash Equilibrium Problems (GNEPs) are non-cooperative games. In these problems, every player's strategy set may depend on the rival player's strategies. Early in the 1950s, Nash [12] introduced a concept of equilibrium for non-cooperative games with N players, where each player's payoff function may be dependent on the strategies of other players, named Nash equilibrium. Arrow et al. [3] developed this concept and extended it to the GNEPs, where both the set of feasible strategies and the payoff function may be dependent on the strategies of other players. The last two decades have seen GNEPs as a major area of research, which has several applications to the real world in the areas of computer science, engineering, and economics, for instance, the abstract economy model [3], a power allocation problem in telecommunications [4],

energy problems [16], wireless communication [7], Economic models and transportation [79, 80], cyber security [81], etc. Robinson [11] discussed the problem of evaluating performance in combat models based on optimization and gave a number of mathematical formulations.

4.2 Motivation

The main challenge in solving GNEPs is that the solution sets are mostly local and nonunique. Therefore, the standard Newtonian methods rarely converge very well. Some reliable techniques have attractive global convergence properties as well, for example, the augmented Lagrangian-type method [46] and the interior-point-type scheme [47], but they are not locally fast convergent. To develop a method for GNEPs, which is both locally and globally convergent, we apply some appropriate methods that also work for nonunique solutions. There are some methods for optimization problems that have locally and globally convergent properties under an error-bound condition; see [31, 50]. These error bounds are dependent on the specific GNEP reformulation. A well-known reformulation takes the KKT conditions of the players, concatenates all KKT conditions, and reformulates them into a semismooth system with the help of the Fischer-Burmeister complementarity function. Unfortunately, it is challenging to find locally superlinearly convergent Newton-type methods for semismooth systems of equations under an error-bound condition without any further assumptions [82].

To overcome these difficulties, Tong et al. [48] have proposed a monotone trust region method for constrained optimization problems. This method is globally and quadratically convergent under the local error-bound assumption. Further, Galli et al. [49] modified this method using a nonmonotone strategy and obtained a nice local convergence as well as global convergence under mild error-bound conditions. We develop this method using a new nonmonotone technique and adaptive trust region radius. Also, with some mild error bound constraints [50], we use a smooth GNEP reformulation,

and using the proposed method, we solve a dataset of 35 different GNEPs.

4.3 Contributions

Neatly novelty and contribution of this chapter are as follows:

- By using the Fischer-Burmeister C-function [42], we formulate a smooth merit function for solving GNEPs under consideration.
- We use an improved nonmonotone term, which helps the INATR method for faster local and global convergence compared to the NTR method [49], and MTR method [48]. Due to its nonmonotone counterpart, the INATR method acquires the local convergence properties. So, we prove the global convergence of the INATR method.
- We take an adaptive trust region radius in place of an ordinary trust region radius which improves the trust region method.
- Step-wise algorithm of the proposed improved nonmonotone adaptive trust region (INATR) method is provided.
- Well-definedness and global convergence of the proposed method are given.
- Numerical performance comparison of INATR method with MTR [48] and NTR [49] for solving GNEPs is given.

4.4 Trust-region framework

Here, we discuss the trust region method so that its generalization to an improved nonmonotone adaptive trust region (INATR) method can be easy to state. In this chapter, we consider the problem of finding a solution to the following constrained

nonlinear system of equations:

$$\left.\begin{aligned} \phi(z) &= 0 \\ \text{subject to } z &\in \Omega \end{aligned}\right\}, \tag{4.1}$$

where $\Omega \subseteq \mathbb{R}^n$ is a nonempty set, and $\phi : \mathbb{R}^n \to \mathbb{R}^n$ is a given function.

We consider some assumptions which are supposed to hold for problem (4.1) to obtain convergence (both local and global) of the introduced trust region method.

Assumptions:

1. $\phi \in C^1$ and $\nabla\phi(z)$ is locally Lipschitzian.
2. The solution set Z^* is nonempty.
3. Ω is a nonempty closed and convex set.
4. $\|\phi(z)\|$ gives a local error bound in the neighborhood of a solution $z^* \in Z^*$.

Here, in notation $\phi \in C^1$, we mean that ϕ is a continuously differentiable function. To illustrate the trust region method, we consider the following merit function

$$\Psi(z) = \tfrac{1}{2}\|\phi(z)\|^2, \tag{4.2}$$

where $\phi \in C^1$, and therefore we have $\nabla\Psi(z) = \nabla\phi(z)^\top\phi(z)$. Since the solution set Z^* is nonempty, z^* solves (4.1) if and only if z^* is a solution of the optimization problem

$$\min_z \Psi(z) \quad \text{subject to } z \in \Omega. \tag{4.3}$$

At the current iterate z_k, the regularized trust-region subproblem is given by

$$\begin{aligned} &\min_d \tfrac{1}{2}\|\phi(z_k) + \nabla\phi(z_k)^\top d\|^2 + \tfrac{1}{2}\mu_k\|d\|^2 \\ &\text{subject to } \|d\| \le \Delta_k, \end{aligned} \tag{4.4}$$

where Δ_k is trust region radius, and μ_k is an appropriate constant that depends on the iteration index k.

4.5 INATR method

This section presents an improved nonmonotone trust region method with an adaptive trust region radius. To achieve global convergence, the trust region methods replace the line search techniques and handle the difficulties due to ill-conditioned problems and nonsmooth problems. The updated approach of the new trust region radius affects the number of iterations as well as the convergence behavior of our algorithm. Therefore, we update the trust region radius using a gradient and Hessian.

Shi et al. [83] have presented a new trust region radius using the gradient information. They use a vector $q_k \in \mathbb{R}^n$ such that q_k satisfies the angle condition:

$$-\frac{\nabla\Psi(z_k)^\top q_k}{\|\nabla\Psi(z_k)\|\|q_k\|} \geq \zeta, \tag{4.5}$$

where $0 < \zeta < 1$. Kamandi et al. [84] have improved the vector $q_k \in \mathbb{R}^n$ by

$$q_k = \begin{cases} -\nabla\Psi(z_k), & \text{if } k = 0 \text{ or } \frac{-(\nabla\Psi(z_k)^\top d_{k-1})}{\|\nabla\Psi(z_k)\|\|d_{k-1}\|} \leq \zeta, \\ d_{k-1}, & \text{if } k > 0, \end{cases} \tag{4.6}$$

where $0 < \zeta < 1$ and the vector d_{k-1} is obtained from solving the subproblem (4.4). To avoid getting a very small trust region radius, a scalar $s'_k \in \mathbb{R}$ is determined by

$$s'_k = \begin{cases} -\frac{\nabla\Psi(z_k)^\top q_k}{q_k^\top B_k q_k}\|q_k\|, & \text{if } k = 0, \\ \max\left(-\frac{\nabla\Psi(z_k)^\top q_k}{q_k^\top B_k q_k}\|q_k\|, \gamma\Delta_{k-1}\right), & \text{if } k > 0, \end{cases} \tag{4.7}$$

where $\gamma > 1$, q_k is calculated using (4.6) and B_k is the Hessian approximation matrix

updated by BFGS update formula, given by

$$B_{k+1} = \begin{cases} \theta_k I - \frac{\theta_k d_k d_k^\top}{d_k^\top d_k} + \frac{y_k y_k^\top}{d_k^\top y_k}, & \text{if } d_k^\top y_k > 0, \\ B_k - \frac{B_k d_k d_k^\top B_k}{d_k^\top d_k} + \frac{y_k^* y_k^{*\top}}{d_k^\top y_k^*}, & \text{otherwise.} \end{cases} \tag{4.8}$$

Then, B_{k+1} is always positive definite. Here, $\theta_k = \frac{d_k^\top y_k}{\|d_k\|^2}$, $y_k = \nabla\Psi(z_{k+1}) - \nabla\Psi(z_k)$, $d_k = z_{k+1} - z_k$, $y_k^* = y_k + t_k d_k$ with $t_k \in [0, C]$, $C > 0$, a constant and t_k is selected in a manner that $\{t_k\} \to 0$.

The trust region radius is calculated by

$$\Delta_k = t^p \min\{s_k', \Delta_{max}\}, \tag{4.9}$$

where p is a nonnegative integer, $\Delta_{max} > 0$ is a constant, and $t \in (0, 1)$. The computational experiments show that iterative algorithms with proper nonmonotone techniques provide stronger convergence behavior [85]. For Newton's method, Grippo et al. [86] have given a nonmonotone technique search technique such that $\Psi(z_k + \alpha d_k) \leq \Psi_{l(k)} + \tau\alpha\nabla\Psi(z_k)^\top d_k$, where scalar α is the largest value of the set $\{l^k : l \in (0, 1),\ k = 0, 1, 2, \ldots\}$, scalar $\tau \in (0, 1)$, and for a given positive integer M_1, the nonmonotone term $\Psi_{l(k)}$ is given by

$$\Psi_{l(k)} = \max_{0 \leq j \leq m(k)} \{\Psi(z_{k-j})\}, \tag{4.10}$$

with $m(0) = 0$, and $0 \leq m(k) \leq \min\{m(k-1) + 1, M_1\}$ for $k \geq 1$. This nonmonotone line search technique is used to improve the numerical performance of the proposed method in this chapter. However, the nonmonotone term (4.10) has some disadvantages. For example, a better functional value obtained at any iteration may be omitted, and the numerical performance depends on the selection of the initially chosen number M_1 [87].

Ahookhosh et al. [87] proposed an improved nonmonotone term that is a convex

combination of the functional value obtained in current iterate Ψ_k and the maximum of functional values $\Psi_{l(k)}$ obtained from some prior successful iterations. Therefore, the new nonmonotone term R_k is given by

$$R_k = \beta_k \Psi_{l(k)} + (1 - \beta_k)\Psi_k, \tag{4.11}$$

where $\beta_k \in [\beta_{min}, \beta_{max}]$, $\beta_{min} \in [0, 1)$ and $\beta_{max} \in [\beta_{min}, 1]$ are two prefixed constants and $\Psi_{l(k)}$ is given by (4.10).

In the proposed algorithm, at k-th iteration, the actual reduction is given by

$$\text{Ared}_k = R_k - \Psi(z_k + d_k). \tag{4.12}$$

Let us define a model function $m_k(d_k)$ at k-th iteration by $m_k(d_k) = \frac{1}{2}\|\phi(z_k) + \nabla\phi(z_k)^\top d_k\|^2$. Then, the predicted reduction is defined by

$$\begin{aligned} \text{Pred}_k &= m_k(0) - m_k(d_k) \\ &= \tfrac{1}{2}\|\phi(z_k)\|^2 - \tfrac{1}{2}\|\phi(z_k) + \nabla\phi(z_k)^\top d_k\|^2. \end{aligned} \tag{4.13}$$

We use the actual and predicted reductions for computing the modified ratio r_k. Therefore, the modified ratio r_k is given by

$$r_k = \frac{\text{Ared}_k}{\text{Pred}_k} = \frac{R_k - \Psi(z_k + d_k)}{m_k(0) - m_k(d_k)}. \tag{4.14}$$

We choose a parameter $u \in (0, 1)$, and accept the trial step d_k if $r_k \geq u$ and increase the trust-region radius Δ_k; otherwise, we decrease the trust region radius Δ_k by a constant fraction and recompute d_k.

Now, we will provide a brief algorithm to solve the main problem (4.3) using the INATR method.

Algorithm 5 Computing z_k such that $\|\Psi(z_k)\| < \epsilon$ to solve (4.1)

Step 0. (Initialization step).

Initially choose $z_0 \in \Omega$, $B_0 = I_{n\times n}$, $R_0 = \Psi(z_0)$, and a positive integer M_1.

Choose constants $0 < \zeta < 1$, $u \in (0,1)$, $t \in (0,1)$, $\gamma_0 > 0$, $\beta_{min} \in [0,1)$, $\beta_{max} \in [\beta_{min}, 1]$, $\Delta_{max} > 0$ and set iteration counter $k = 0$.

Step 1. (Terminating condition).

If $\|\Psi(z_k)\| < \epsilon$, then stop, and give the output as z_k, an ϵ-precision solution to (4.1)

Step 2. (Main steps).

Step 2.1. (Computation of vectors required for trust region radius)

Compute vectors q_k by (4.6), s'_k by (4.7), and set $p = 0$ (p is used for (4.9))

Step 2.2. (Computation of trust region radius Δ_k)

Compute the trust region radius Δ_k by (4.9).

Step 2.3. (Computation of projected-gradient direction)

$$\begin{aligned} \text{Compute } d_k^G &= -\frac{\Delta_k}{\Delta_{\max}}\gamma_k \nabla\Psi(z_k), \\ \text{and } \overline{d}_k^G &= P_\Omega[z_k + d_k^G] - z_k, \end{aligned} \tag{4.15}$$

where $\gamma_k = \min\left\{1, \frac{\Delta_{\max}}{\|\Psi(z_k)\|}, \frac{\beta\Psi(z_k)}{\|\Psi(z_k)\|^2}\right\}$.

Step 2.4. (Computation of projected trust region direction)

Compute the trust region direction d_k^{TR} by solving (4.4). Then, compute the projected direction

$$\overline{d}_k^{TR} = P_\Omega[z_k + d_k^{TR}] - z_k.$$

Step 2.5. (Computation of optimal combined direction)

Compute a convex combination of the projected trust region direction and the projected-gradient direction by

$$\overline{d}_k = t^*\overline{d}_k^G + (1-t^*)\overline{d}_k^{TR},$$

where $0 < t^* < 1$ is obtained by solving the problem

$$\min_{t\in[0,1]} \tfrac{1}{2}\left\|\phi(z_k) + \nabla\phi(z_k)^\top(t\overline{d}_k^G + (1-t)\overline{d}_k^{TR})\right\|$$

Step 2.6. (Acceptance and rejection of trial step d_k)

Compute actual reduction by (4.12) and predicted reduction by (4.13) to compute r_k by (4.14). If $r_k < u$, then set $p = p + 1$, and go to Step 2.2.

Step 2.7. (Increase the iteration counter k and update z_k).

Set $z_{k+1} = z_k + d_k$. Update B_{k+1} by (4.8), choose $\beta_k \in [\beta_{min}, \beta_{max}]$. Set $k = k + 1$, and go to **Step 1**.

In Algorithm 5, the loop between Step 2.2. and Step 2.6. is known as the inner cycle. It is called an unsuccessful iteration if $r_k < u$. In the acceptance and rejection of d_k, the parameter u in Step 2.6. plays a crucial role. We implicitly assume throughout our convergence analysis that after finitely many iterations, the termination criterion (**Step 1.**) does not hold. Therefore, in our theoretical analysis of Algorithm 5, we suppose that none of the iterates z_k is an exact stationary point, so that $P_\Omega[z_k - \nabla\Psi(z_k)] - z_k \neq 0$ for all k.

Note that the sequence $\{z_k\}$ produced by Algorithm 5 contained in feasible set Ω. Also, from [48], we can see that Algorithm 5 with r_k from (4.14) is well-defined, i.e., the number of inner iterations between (Step 2.2.) and (Step 2.6.) is finite for every outer iteration k.

4.6 Convergence Analysis

In this section, we prove the convergence of Algorithm 5. To prove the convergence, we first present the following lemmas and propositions.

Lemma 4.1 *Let $\{z_k\}$ be a sequence, which is generated by Algorithm 5. Then, the sequence of nonmonotone terms $\{\Psi_{l(k)}\}$ is a monotonic decreasing sequence.*

Proof: The proof is the same as Lemma 4 in [87]. □

Lemma 4.2 *Following inequality*

$$\nabla\Psi(z_k)^\top \overline{d}_k^G(\Delta) \leq -\left(\frac{\Delta}{\Delta_{\max}\gamma_k}\right)\left\|\overline{d}_k^G(\Delta_{\max})\right\|^2 \tag{4.16}$$

holds under the consideration of Algorithm 5.

Proof: The proof is the same as Lemma 1 in [49]. □

The following proposition establishes a relation between predicted reduction and projected gradient direction, which is used to prove the convergence of Algorithm 5.

Proposition 4.1 *Let $\{z_k\}$ be a sequence generated by the Algorithm 5. Let z^* be an accumulation point of a subsequence $\{z_k\}_{k\in K}$. If z^* is not a stationary point, then there exists an index $k' > 0$ and $\Delta_{max} > 0$ such that the following inequality*

$$-t\nabla\Psi(z_k)^\top \overline{d}_k^G(\Delta) \leq Pred_k(\Delta) \tag{4.17}$$

holds for all $\Delta \in (0, \Delta_{max})$, and for all $k \in K$ with $k \geq k'$.

Proof: We start the proof from the following useful observation. From the assumption 1, it is easy to see that there exists a constant c_1 such that

$$\|\nabla\phi(z_k)\| \leq c_1 \text{ for all } k \in K. \tag{4.18}$$

Note that for all $k \in K$,

$$\begin{aligned}
\|\nabla\phi(z_k)^\top \overline{d}_k^G(\Delta)\| &= \|\nabla\phi(z_k)^\top (P_\Omega[z_k + d_k^G] - z_k)\| \text{ from (4.15)} \\
&\leq \|\nabla\phi(z_k)\|\|(z_k + d_k^G) - z_k\| \\
&\leq \tfrac{\Delta\gamma_k}{\Delta_{\max}}\|\nabla\phi(z_k)\|\|\nabla\Psi(z_k))\| \text{ from (4.15)} \\
&\leq c_1\Delta \text{ using definition of } \gamma_k \text{ in Step 2.3. and (4.18).}
\end{aligned} \tag{4.19}$$

It is given that z^* is not a stationary point. Therefore, we have a positive number γ^* such that

$$\gamma^* = \lim_{k\in K, k\to\infty} \gamma_k = \min\left\{1, \tfrac{\Delta_{\max}}{\|\nabla\Psi(z^*)\|}, \tfrac{\beta\Psi(z^*)}{\|\nabla\Psi(z^*)\|^2}\right\}.$$

Thus, using (4.15) and $\gamma^* > 0$, we have

$$\|\overline{d}_k^G(\Delta_{\max})\| \to \|P_\Omega[z^* - \gamma^*\nabla\Psi(z^*)] - z^*\| > 0 \text{ as } k \to \infty.$$

Hence, there exist a constant $c_2 > 0$ and an index $k' > 0$ such that

$$\|\overline{d}_k^G(\Delta_{\max})\| \geq c_2 \text{ for all } k \in K, k \geq k'. \tag{4.20}$$

Consider

$$\Delta' = \min\left\{\Delta_{\max}, \tfrac{(1-t)c_2^2}{c_1^2\Delta_{\max}}\right\}. \tag{4.21}$$

Now we prove (4.17). From (4.13), we have

$$\begin{aligned}
Pred_k(\Delta) &= m_k(0) - m_k(\overline{d}_k) \\
&= \tfrac{1}{2}\|\phi(z_k)\|^2 - \tfrac{1}{2}\|\phi(z_k) + \nabla\phi(z_k)^\top \overline{d}_k(\Delta)\|^2 \\
&\geq \tfrac{1}{2}\|\phi(z_k)\|^2 - \tfrac{1}{2}\|\phi(z_k) + \nabla\phi(z_k)^\top \overline{d}_k^G(\Delta)\|^2 \\
&= \tfrac{1}{2}\|\phi(z_k)\|^2 - \tfrac{1}{2}\|\phi(z_k)\|^2 - \nabla\Psi(z_k)^\top \overline{d}_k^G(\Delta) - \tfrac{1}{2}\|\nabla\phi(z_k)^\top \overline{d}_k^G(\Delta)\|^2 \\
&= -\nabla\Psi(z_k)^\top \overline{d}_k^G(\Delta) - \tfrac{1}{2}\|\nabla\phi(z_k)^\top \overline{d}_k^G(\Delta)\|^2 \\
&= -t\nabla\Psi(z_k)^\top \overline{d}_k^G(\Delta) - (1-t)\nabla\Psi(z_k)^\top \overline{d}_k^G(\Delta) - \tfrac{1}{2}\|\nabla\phi(z_k)^\top \overline{d}_k^G(\Delta)\|^2 \\
&\geq -t\nabla\Psi(z_k)^\top \overline{d}_k^G(\Delta) + (1-t)\left(\tfrac{\Delta}{\Delta_{\max}\gamma_k}\right)\left\|\overline{d}_k^G(\Delta_{\max})\right\|^2 - \tfrac{1}{2}c_1^2\Delta^2 \\
&\qquad\qquad \text{using Lemma 4.2 and (4.19)} \\
&\geq -t\nabla\Psi(z_k)^\top \overline{d}_k^G(\Delta) + c_1^2\Delta\Delta' - \tfrac{1}{2}c_1^2\Delta^2 \text{ using (4.20), (4.21) and } 0 < \gamma_k \leq 1 \\
&\geq -t\nabla\Psi(z_k)^\top \overline{d}_k^G(\Delta) \text{ because } \Delta \leq \Delta',
\end{aligned}$$

which is the required inequality. □

Now, using Lemma 4.1 and Proposition 4.1, we will prove the convergence of Algorithm 5, which is as follows.

Theorem 4.1 *Suppose $\{z_k\}$ is the sequence generated by the Algorithm 5. Then, every accumulation point of $\{z_k\}$ is a stationary point for Ψ.*

Proof: Let z^* be an accumulation point of $\{z_k\}$ and let $\lim_{k\in K, k\to\infty} z_k = z^*$ be a convergent subsequence. We shall use contradiction to prove this theorem. Assume that

z^* is not a stationary point. Then, using the calculation made in proof of Proposition 4.1, we have a positive number γ^* such that

$$\gamma^* = \lim_{k \in K, k \to \infty} \gamma_k = \min \left\{ 1, \frac{\Delta_{\max}}{\|\nabla \Psi(z^*)\|}, \frac{\beta \Psi(z^*)}{\|\nabla \Psi(z^*)\|^2} \right\}$$

and there exist an index $k' > 0$ and a constant $c_3 > 0$ such that

$$\|\overline{d}_k^G(\Delta_{\max})\| \geq c_3 \text{ for all } k \in K, k \geq k'. \tag{4.22}$$

Further, with the help of Proposition 4.1, there exist k' and Δ_{max} such that (4.17) is satisfied. Moreover, using (4.9), for $k \in K$, $k \geq k'$ and a positive constant $c, \Delta_k^* > c\Delta$. It means that Δ_k^* has a lower bound for sufficiently large $k \in K$.

Since z^* is not a stationary point, therefore $r_k \geq u$ for all $k \in K$, $k \geq k'$. Thus, from (4.14), we have

$$R_k - \Psi(z_k + d_k) \geq u(m_k(0) - m_k(d_k)). \tag{4.23}$$

Using (4.11), we have

$$R_k = \beta_k \Psi_{l(k)} + (1 - \beta_k)\Psi_k \leq \beta_k \Psi_{l(k)} + (1 - \beta_k)\Psi_{l(k)} = \Psi_{l(k)}.$$

Therefore, from (4.23), $\Psi_{l(k)} - \Psi(z_k + d_k) \geq u(m_k(0) - m_k(d_k))$. Let $k = l(k) - 1$. Then, we have

$$\begin{aligned}
\Psi_{l(l(k)-1)} - \Psi_{l(k)} &\geq u(m_{l(k)-1}(0) - m_{l(k)-1}(d_{l(k)-1}) \\
&= uPred_{l(k)-1}(\Delta^*_{l(k)-1}) \\
&\geq -u\left(t\nabla\Psi(z_{l(k)-1})^\top \overline{d}^G_{l(k)-1}(\Delta^*_{l(k)-1})\right) \text{ using (4.17)} \\
&\geq ut\left(\frac{\Delta^*}{\Delta_{\max}\gamma_{l(k)-1}}\right)\left\|\overline{d}^G_{l(k)-1}(\Delta_{\max})\right\|^2 \text{ using (4.16)} \\
&\geq 0.
\end{aligned} \tag{4.24}$$

Now we have $\Psi(z_k) \leq \Psi_{l(k)}$, and $\Psi(z_k)$ is bounded from below. Also, from Lemma 4.1, $\{\Psi_{l(k)}\}$ is a decreasing sequence. Therefore, we have

$$|\Psi_{l(l(k)-1)} - \Psi_{l(k)}| \to 0 \text{ as } k \to \infty.$$

It implies, from (4.24), that $\left\|\overline{d}^G_{l(k)-1}(\Delta_{\max})\right\|^2 \to 0$ as $k \to \infty$, which is a contradiction. Hence, the proof is completed.

□

4.7 Application to generalized Nash equilibrium problems

Consider the player convex GNEP as described in Chapter 1. Therefore, we have the reformulated system (1.8) for player convex GNEP.

Therefore, for computing the solution to the original problem (1.2), we concentrate on solving (1.5). To solve system (1.5), we reformulate (1.5) into a smooth system of equations using Hadamard product (componentwise product). Using the slack variables $s \in \mathbb{R}^m_+$ and Hadamard product $(so\lambda)_i = s_i\lambda_i$ for $i = 1, 2, \ldots, m$, we have

$$F(z) = \begin{pmatrix} L(x,\lambda) \\ g(x) + s \\ so\lambda \end{pmatrix} \text{ such that } z = (x, s, \lambda) \in \Omega = \mathbb{R}^n \times \mathbb{R}^m_+ \times \mathbb{R}^m_+, \tag{4.25}$$

which is a box-constrained system of equations. Suppose that the corresponding solution set

$$\{z \in \Omega : F(z) = 0\} \tag{4.26}$$

is nonempty. We first define a merit function to solve the system (4.25). Consider the merit function $\psi(z) = \frac{1}{2}\|F(z)\|^2$. Then, the system (4.25) will be reduced into a

constrained optimization problem similar to optimization problem (4.3):

$$\min_{z\in\Omega} \psi(z), \tag{4.27}$$

where $\Omega = \mathbb{R}^n \times \mathbb{R}^m_+ \times \mathbb{R}^m_+$. Therefore, the problem (4.27) can be solved using Algorithm 5.

4.8 Numerical results

In this section, we provide some numerical examples of generalized Nash equilibrium problems and performance measures of Algorithm 5 to analyze the efficiency of the INATR method 5. Galli et al. [49] have given a detailed comparison of "A nonmonotone trust region method" with original monotone trust region method and a solver for nonlinear, box-constrained systems of equations in [49]. Then, numerical performances in [49] indicate that the NTR method [49] outperforms the other numerical methods compared in [49]. Here, we analyze the numerical performance of our approach using a larger collection of examples. We have compared the following three algorithms:

1. Algorithm 5 (INATR),
2. MTR method by Tong et al. [48], and
3. NTR method by Galli et al. [49].

MTR, NTR, and INATR have been coded in MATLAB software (version: 9.12.0.2009381 (R2022a)) on a CPU of i5-10th generation. During the compilation of algorithms, we use a stopping condition $\|\nabla\Psi(z_k)\| < \epsilon$, with $\epsilon = 10^{-4}$. Other algorithmic parameter values are as follows:

- $\zeta = 10^{-2}$, $t = 0.3$, $u = 0.0001$, $\gamma = 2$, $\beta = 0.99$,
- In subproblem (4.4), we use $\mu = C\min(\|\phi(z_k)\|^2, 1)$, $C = 0.0001$,

- $\alpha = C/2$, $\beta_0 = 0.5$, $B_0 = I_{n\times n,}$
- Termination condition $\epsilon = 10^{-4}$,
- $\Delta_{\min} = 0.0001$, $\Delta_{\max} = \max(1000, \|(\text{initial point})\|)$, $\Delta_0 = 5$,
- To obtain nonmonotone term, we use $M_1 = 20$,
- For the improved nonmonotone term R_k, parameter β_k is updated by

$$\beta_k = \begin{cases} \frac{1}{2}\beta_0, & \text{if } k = 1, \\ \frac{1}{2}(\beta_{k-1} + \beta_{k-2}), & \text{if } k \geq 2, \end{cases} \tag{4.28}$$

- $\rho_1 = 0.0001$, $\rho_2 = 0.75$, $\sigma = 0.5$.

A dataset of 35 different GNEPs [47] has been tested using each of three methods: INATR, NTR, and MTR. We use *fmincon* (built-in function of Matlab) for solving the subproblem (4.4) to maintain consistency in each of the three algorithms. To present the performance of the methods, we use the performance profiles given by Dolan et al. [88]. We use some parameters in measuring the performance profiles: Let n_s be the number of solvers, and n_p represent the number of test problems. Let $\mathbb{S}$ be the set of all algorithms, and P be the set of test problems considered here. We are interested in using computation time as a performance measure. Therefore, for every problem p and solver s, we use $t'_{p,s}$ as the CPU time consumed by solver s to solve problem p. Therefore, the performance ratio is given as

$$r'_{p,s} = \frac{t'_{p,s}}{\min\{t'_{p,s} : s \in \mathbb{S}\}}.$$

It is clear that for all problems p and solvers s, $r'_{p,s} \geq 1$. Therefore, the performance

profile for every solver s is given by

$$P_s(\zeta) = \frac{\text{cardinality of set } \{p \in P : r'_{p,s} \leq \zeta\}}{n_p}.$$

Thus, for the solver, $s \in \mathbb{S}$, the ratio $P_s(\zeta)$ is the probability that a performance ratio $r'_{p,s}$ is within a factor $\zeta \in \mathbb{R}$ of the best possible ratio. We have given performance profiles based on CPU-time in Figure 4.1 and based on the number of iterations in Figure 4.2. The method with the highest percentage of problems solved within a time that was within the best time factor *zeta* is represented by the top curve.

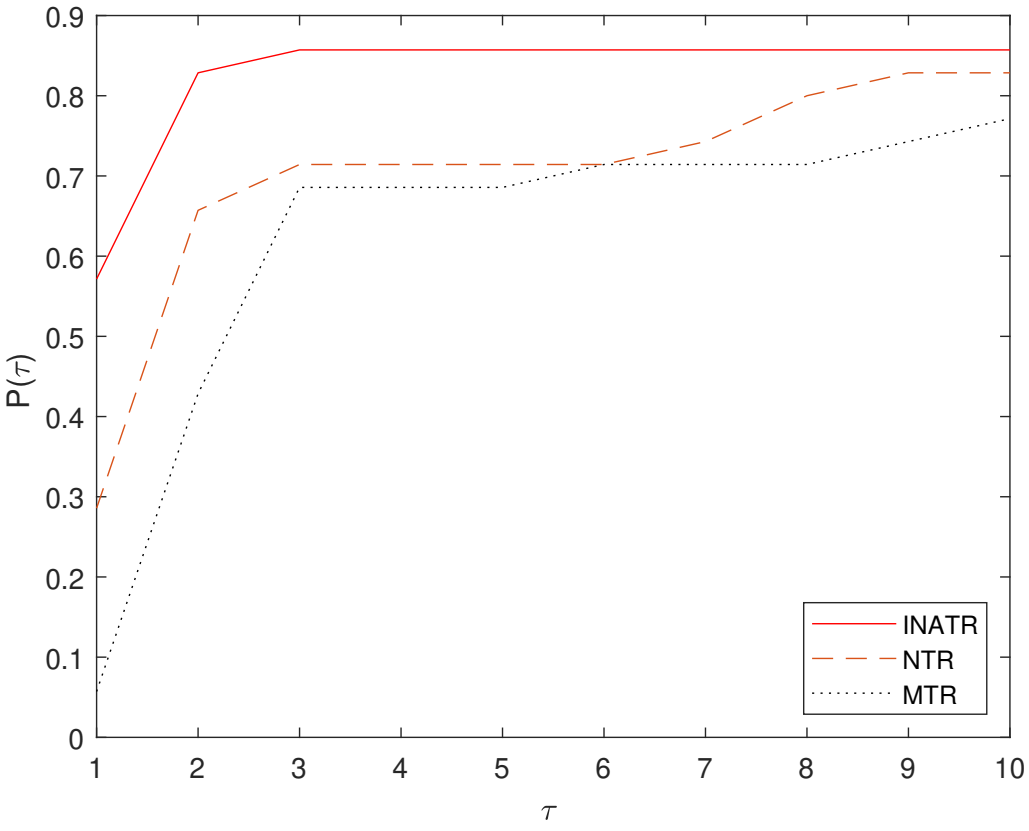

Figure 4.1: Performance profile based on CPU-time

From Fig. 4.1, it is clear that the best-performing method is the INATR method, while the worst-performing method is the MTR method out of all three methods based

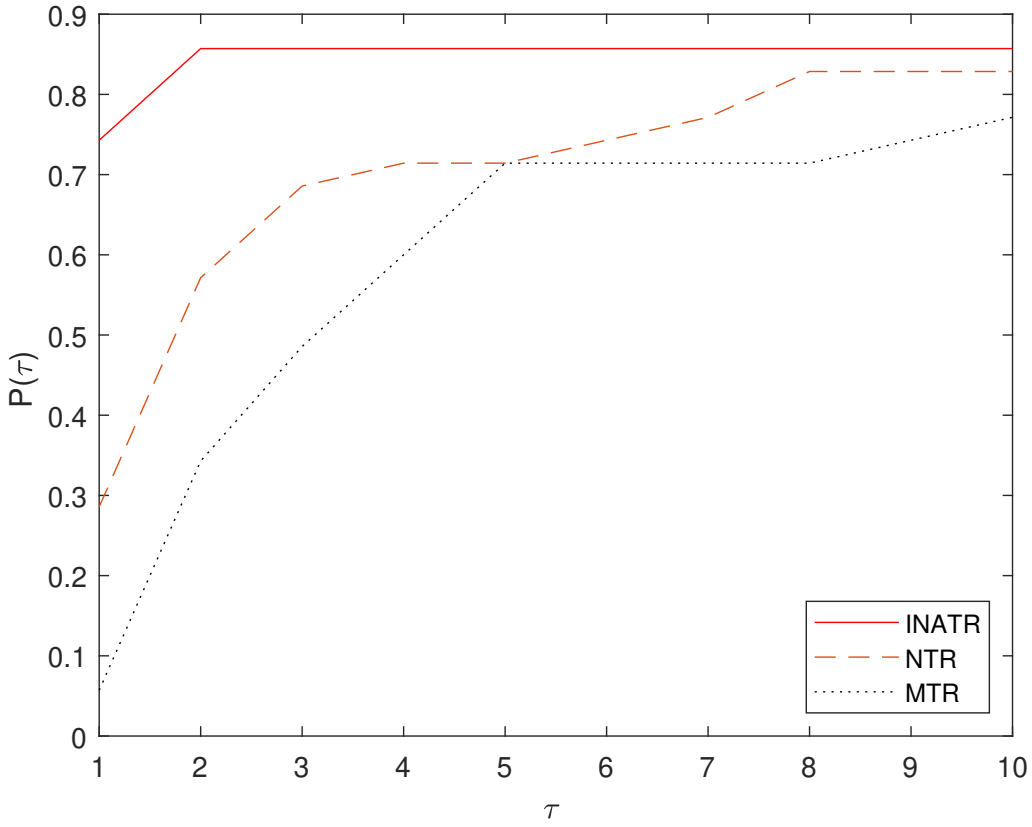

Figure 4.2: Performance profile based on number of iterations

on the CPU time. Also, from Fig. 4.2, it is clear that the INATR method performs better than the rest of the three methods based on the number of iterations.

4.8.1 Some illustrative examples

Example 4.1 *Consider the following GNEP that has two players and one shared constraint:*

$$\min_{x_1} x_1^2 + \tfrac{8}{3}x_1x_2 - 34x_1 \qquad \min_{x_2} x_2^2 + \tfrac{5}{4}x_1x_2 - \tfrac{97}{4}x_2$$

$$\text{subject to } x_1 + x_2 \leq 15, \qquad \text{and} \qquad \text{subject to } x_1 + x_2 \leq 15,$$

$$0 \leq x_1 \leq 10, \qquad 0 \leq x_2 \leq 10.$$

This game was introduced by Harker [63]. For this problem, Algorithm 5 converges to $\overline{x}_1 = 5, \overline{x}_2 = 9$ *starting from any feasible point. We have compared the two algo-*

rithms: INATR and NTR [49]. We have specified some regions for starting points. We have randomly taken 100 points from each specified region and presented the minimum, median, and maximum of the number of iterations and CPU time consumed by the two algorithms: INATR and NTR. In the comparison Table 4.1, we can see that the proposed Algorithm 5 performs better than NTR [49]. Also, we have given a graph showing the convergence of Problem 4.1 starting from a specific initial point in Figure 4.3.

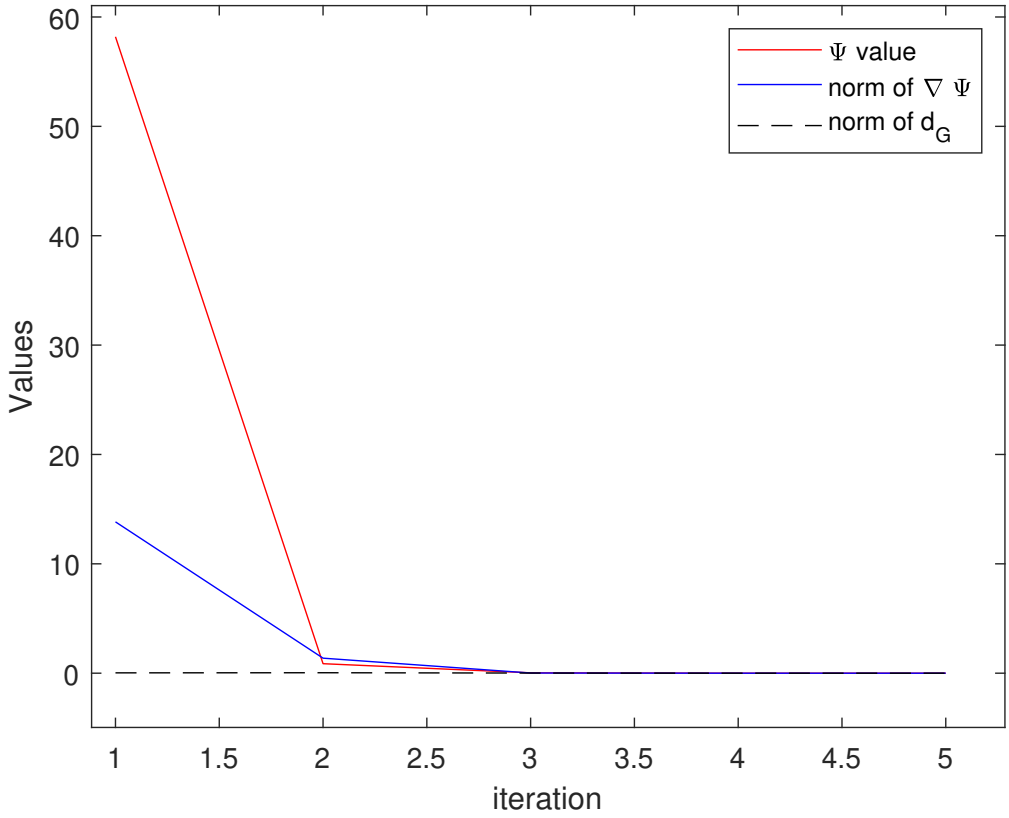

Figure 4.3: Convergence of INATR method for Problem 4.1

Example 4.2 *Kesselman et al. [65] have introduced a model of internet switching, where selfish users have produced the traffic. The model deals with the behaviour of users sharing the limited ability of the first in, first out buffer. Every user's utility depends on their congestion level and transmission rate. To be more precise, we assume that the buffer capacity is B and the total number of users is N. The user v controls*

Table 4.1: Performances of INATR and NTR methods on Problem 4.2

Region of initial point	INATR method						NTR method					
	Iteration number			Computation time			Iteration number			Computation time		
	Min	Median	Max	Min	Median	Max	Min	Median	Max	Min	Median	Max
$\|x_0\| \leq 1$	42	59	70	260.78	299.88	341.60	315	378	493	2849.67	3107.01	4052.93
$1 < \|x_0\| \leq 5$	85	119	150	478.27	614.43	772.31	307	382	450	2503.14	3155.58	3770.24
$5 < \|x_0\| \leq 15$	4	5.5	6	20.09	25.41	27.82	4	5	7	19.90	25.69	30.20
$15 < \|x_0\| \leq 50$	93	121	181	490.41	626.18	935.54	298	377	431	2397.78	2958.76	3406.41
$50 < \|x_0\| \leq 100$	95	104	123	765.68	834.11	984.42	781	852	958	7038.80	7686.25	8631.36

the amount of his "packets" in the buffer, denoted by $x^v \in [0, \infty)$. *The utility function for the player* v $(v = 1, 2, 3, \ldots, N)$ *is given by*

$$\theta_v(x^v, x^{-v}) = -\frac{x^v}{x^1 + x^2 + \cdots + x^N}\left(1 - \frac{x^1 + x^2 + \cdots + x^N}{B}\right), \ (x^v, x^{-v}) \in \mathbb{R}^N \tag{4.29}$$

and the constraints are $x^1 + x^2 + \cdots + x^N \leq B$ *and* $x^v \geq l_v$, *where* $l_v \geq 0$. *Kesselman et al. [65] have shown that the model (4.29) has a unique solution* $\bar{x}^v = B(N-1)/N^2$, $v = 1, 2, \ldots, N$.

In this model, we take $N = 20$, *i.e.,* 20 *players,* $l_v = 0.01$, *for each* $v = 1, 2, \ldots, N$ *and* $B = 1$ *for numerical computation. The problem has a total of* 20 *variables, and the system (4.2) pertaining to solving this GNEP involves* 41 *variables. The GNEP problem (4.29) for twenty players has a unique solution* $\bar{x}^v = 0.05$, *for each* $v = 1, 2, \ldots, N$. *In employing Algorithms 5 and NTR [49] on this problem, we randomly take the initial points from the region indicated in Table 4.2. The numerical performance of Algorithms INATR and NTR on this GNEP is provided in Table 4.2, which clearly indicates that the INATR method is cost-efficient compared to NTR corresponding to each specified region for initial points. Also, we have given a graph showing the convergence of Problem 4.2, starting from a specific initial point in Figure 4.4.*

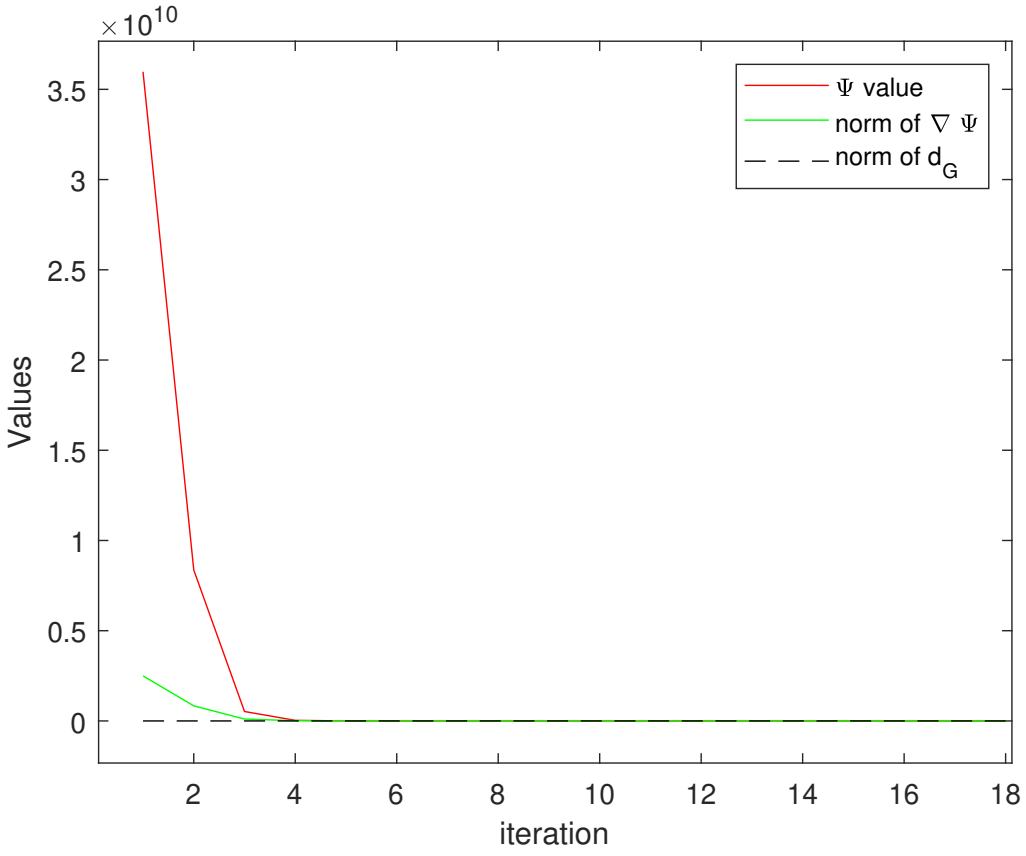

Figure 4.4: Convergence INATR method for Problem 4.1

Table 4.2: Performances of INATR method and NTR-method on Problem 4.2

Region of initial point	INATR method						NTR method					
	Iteration number			Computation time			Iteration number			Computation time		
	Min	Median	Max	Min	Median	Max	Min	Median	Max	Min	Median	Max
$\|x_0\| \leq 1$	2	3	5	5.61	6.99	10.36	2	3	5	22.01	23.68	25.83
$1 < \|x_0\| \leq 5$	13	16	27	48.73	52.95	62.40	15	18	27	52.69	59.98	82.17
$5 < \|x_0\| \leq 15$	13	15	21	43.59	50.84	92.67	13	20	22	44.09	68.61	105.83
$15 < \|x_0\| \leq 50$	14	22	30	48.82	95.77	128.40	15	25	69	55.27	83.82	204.17
$50 < \|x_0\| \leq 100$	17	18	22	58.77	64.17	71.73	18	25	67	57.11	108.62	198.27

4.9 Conclusion

In this chapter, we have proposed an INATR method (Algorithm 5) for constrained optimization problems and have shown its application to solve GNEPs (Section 4.7). In this method, we have computed an adaptive trust region radius ((4.5)-(4.9)) using gradient and Hessian matrix information. It affects the convergence of the algorithm

as well as the number of iterations. Also, we have used a new nonmonotone technique (4.11), a convex combination of the functional value obtained in the current iteration and the maximum of the functional values obtained from some prior successful iterations. Subsequently, we have given a global convergence (Theorem 4.1) of the proposed algorithm. Further, we have solved a dataset of 35 GNEPs using the INATR method and have provided a comparison of the performance profiles of the INATR method with the existing two methods: NTR and MTR.

****