Interior Point Method for Nonlinear Optimization

A Dissertation report submitted in partial fulfillment of the requirements for the degree

of

M.Sc in Mathematics

Submitted by

Sakshi Daksh
2K19/MSCMAT/16

Under the supervision of

Dr. L N Das

DEPARTMENT OF APPLIED MATHEMATICS

DELHI TECHNOLOGICAL UNIVERSITY
(FORMERLY DELHI COLLEGE OF ENGINEERING), BAWANA ROAD,
DELHI – 110042

APRIL 2021

Department of Applied Mathematics
Delhi Technological University
(Formerly Delhi College of Engineering), Bawana Road, Delhi – 110042

CANDIDATE'S DECLARATION

I, Sakshi Daksh, Roll No. (2K19/MSCMAT/16) of M.Sc (Mathematics), hereby declare that the project Dissertation titled **Interior Point Method for Nonlinear Optimization**,which is submitted by me to the **Department of Applied Mathematics, Delhi Technological University Delhi** , in partial fulfillment of the requirement for the award of the degree of Master of Science, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Associateship, Fellowship or other similar title or recognition.

Sakshi Daksh

(....................)

Place: Delhi **Sakshi Daksh**

Date: 24.05.2021 **2K19/MSCMAT/16**

Department of Applied Mathematics
Delhi Technological University
(Formerly Delhi College of Engineering), Bawana Road, Delhi – 110042

CERTIFICATE

I hereby certify that the Project dissertation titled **Interior Point Method for Nonlinear Optimization**, which is submitted by **Sakshi Daksh** (2K19/MSCMAT/16) an postgraduate student of the **Department of Applied Mathematics**, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Science, is a record of the project work carried out by the students under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree to this University or elsewhere.

Dr. L.N. Das
Professor. Applied Mathematics, DTU.

(...................)

Place: Delhi **Dr. L N Das**

Date: 24.05.2021 **Department of Applied Mathematics, Delhi**

ACKNOWLEDGEMENTS

It is an incredible pleasure for me to communicate my regard and profound feeling of appreciation to my M.Sc supervisor **Dr. L N Das**, Professor, Department of Applied Mathematics, Delhi Technological University, Delhi, for his wisdom, vision, ability, direction, eager contribution and tireless consolation during the arranging and improvement of this research work. I also thankfully acknowledge his meticulous endeavors in completely going through and improving the original copies without which this work couldn't have been completed.

I am exceptionally obliged to **Prof. S. Shivprasad Kumar**, Head of the Department, Applied Mathematics for providing all the facilities, help and support for doing this research work.

I'm obliged to my parents for their ethical help, love, consolation and favors to finish this task.

I likewise might want to communicate my friends and all other peoples whose names don't show up here, for helping me either straightforwardly or in a roundabout way in all even and odd times.

Finally, I am obliged and thankful to the Almighty for helping me in this endeavor.

(Sakshi Daksh)

ABSTRACT

The line search methods are effective tool to solve nonlinear optimization problems. A variant line search method namely interior point estimation method has been effectively efficient to solve nonlinear constrained optimization problems. In this report we will present an interior point estimation method that solves perturbed Karush Kuhn Tucker conditions in a primal-dual optimization problem. At each iteration of the interior point estimation method, the algorithmic process computes the direction in which to be proceeded, and then calculates the suitable step length along the search direction. In order to compute the search direction, interior point estimation method utilizes Newton method and a merit function to decide a step length that balances the conflicting situation of reducing the objective function with satisfying the constraints. The proposed computation method is investigated on some test problems and real world problems. Further numerical comparison with existing methods shows that the computation process is efficient.

Keywords: Interior Point Method, Newton Method, Merit Function

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Line search methods start with a starting point and go through a sequence of iterations to find the optimal point. The next point of iteration is calculated by the addition of previous point and the direction in which to move multiply by a suitable step length. Thus, line search methods follows the following iterative sequence to obtain the optimal point of unconstrained optimization problem $\min\{g(t) : t \in \mathbb{R}^n\}$

$$t^{(k+1)} = t^{(k)} + \alpha^{(k)} d^{(k)}, \;\; k \in \{0, 1, 2 \ldots\},$$

where $d^{(k)}$ is a search direction and $\alpha^{(k)} \in (0, 1]$ is the step length along the direction d_k. Moreover, the step length $\alpha^{(k)}$ is selected such that the function value at current iteration should be less than or equal to the previous point. It may be possible that if we are moving towards the search direction, where the function value does not reduce then we shorten the step length until the following inequality satisfies

$$g(t^{(k)} + \alpha d^{(k)}) \leq g(t^{(k)}) \;\; \text{for all } k \in \{0, 1, 2, \ldots\},$$

where α is reduced step length.

Constrained nonlinear optimization problems can be found in a broad variety of optimization problems. A line search technique known as interior point methods (IPM) is commonly used to solve constrained nonlinear optimization problems. Interior-point algorithms have been the main and most promising area of study for optimization techniques because of its polynomial-time complexity. In 1884, Karmarkar published a new polynomial-time algorithm after Khachiyan's ellipsoid method. IPM outperformed the ellipsoid process in terms of performance. Karmarkar also argued that his method outperformed the simplex method. Interior-point techniques were originally used to solve problems where feasible regions have nonempty interiors and the starting point was a feasible region's interior point. However, computing an interior-point of the feasible region is a difficult process, and the feasible region's interior can be an empty set.

The infeasible interior-point approach was introduced by Lustig, and it is an interior-point algorithm in which the initial point is not a feasible point (IIPM). Vanderbei introduced LOQO, a software package that applies a primal-dual interior-point approach for general quadratic programming, in 1999. The function of Vanderbei and Shanno covers both convex and nonconvex optimization problems.

The paper is organised as follows. In Section 3.1.1 , we give some basic definitions. The description of the proposed algorithm is detailed in Section 4.1. Section contains the details of the merit function 4.2. In Section 5.1, numerical results of the proposed algorithm are shown and also, compare the results with the existing methods.

1.1 Preliminaries and Notations

The following lists of notations used throughout the article.

- ∇_x denotes the gradient operator.

- ∇_{xx} denotes the Hessian operator.

- $\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2\right)^{\frac{1}{2}}$.

The following optimization problem is taken into consideration:

$$\left.\begin{array}{ll} \min & f(x) \\ \text{s.t.} & c_i(x) \geq 0, \;\; i = 1, 2, \cdots m, \end{array}\right\} \tag{1.1}$$

where $x = (x_1, x_2, \ldots, x_n)^\top$ is a vector of decision variables and the objective function f ($f : \mathbb{R}^n \to \mathbb{R}$) and the constraint functions c_i ($c_i : \mathbb{R}^n \to \mathbb{R}$) are twice continuously differentiable for all $i = 1, 2, \ldots, m$.

1.1.1 Basic Definitions

1.1.1.1 Local minimizer

Let $\Omega \subset \mathbb{R}^n$ and $a^* \in \Omega$. A point a^* is local minimizer of the function $f : \Omega \to \mathbb{R}$ if there exists $\epsilon > 0$ such that whenever $\|a - a^*\| < \epsilon$ then

$$f(a) \geq f(a^*) \text{ for every } a \in \Omega \backslash \{a^*\}.$$

1.1.1.2 Global minimizer

Let $\Omega \subset \mathbb{R}^n$ and $a^* \in \Omega$. A point a^* is global minimizer of the function $f : \Omega \to \mathbb{R}$ if there exists $\epsilon > 0$ such that

$$f(a) \geq f(a^*) \text{ for every } a \in \Omega \backslash \{a^*\}.$$

1.1.1.3 Descent Direction

Let $\hat{y} \in \mathbb{R}^n$ and $d \in \mathbb{R}^n$. If there exists $\delta > 0$ such that $f(\hat{y} + \alpha d) < f(\hat{y})$ for all $\alpha \in (0, \delta)$, then d is a descent direction of f at $\hat{y}$. Alternatively,
Let $\hat{y} \in \mathbb{R}^n$ be a point and $d \in \mathbb{R}^n$ satisfying

$$\langle d, \nabla_y f(\hat{y}) \rangle < 0.$$

Then d is a descent direction of function f at $\hat{y}$.

1.1.1.4 Convex Function

Consider C as a convex subset of $\mathbb{R}^n$. Then the function $f : C \to \mathbb{R}$ is convex if the following inequality satisfy:

$$f(\nu y_1 + (1 - \nu) y_2) \leq \nu f(y_1) + (1 - \nu) f(y_2), \text{ for any } y_1, y_2 \in C \text{ and } \nu \in [0, 1].$$

Alternatively,
A twice differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is convex, if and only if the hession $\nabla_{yy} f(y)$ is positive semi-definite for all $y \in \mathbb{R}^n$.

1.1.1.5 Feasible point

A point y is said to be feasible point for problem (2.1) if $y \in \mathcal{F}$, where

$$\mathcal{F} = \{y : c_i(y) \geq 0, \ i \in \{1, 2, \cdots, m\}\}.$$

The set $\mathcal{F}$ is known as feasible set.

1.1.1.6 Karush-Kuhn-Tucker conditions

If y^* is a local minimum for problem (2.1), then the following necessary conditions hold :

- $\nabla_y f(y^*) + \sum_{i=1}^{m} \lambda_i \nabla_y c_i(y^*) = 0, \qquad$ (Stationarity);
- $c_i(y) \geq 0, \qquad i \in \{1, 2, \cdots, m\}, \qquad$ (Feasibility);
- $\lambda_i \geq 0, \qquad i \in \{1, 2, \cdots, m\}, \qquad$ (Nonnegativity);
- $\lambda_i c_i(y) = 0, \qquad i \in \{1, 2, \cdots, m\}. \qquad$ (Complementarity slackness);

Chapter 2

Interior Point Method

2.1 Description of Interior Point Method

In this section, we first convert the inequality constraints into equality by introducing slack variables. Thereafter, a log-barrier problem is formulated and corresponding to this barrier problem KKT conditions are derived. In order to solve KKT conditions, Newton method is applied to find the search directions towards the solution of KKT conditions.

Recall the optimization problem (2.1)

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & c(x) \geq 0, \end{aligned}$$

where $c(x) = (c_1(x), c_2(x), \cdots, c_m(x))^\top$.

Introducing the slack variables vector $t = (t_1, t_2, \cdots, t_m)$ to make the inequality constraints into the equality constraints

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & c(x) - t = 0, \\ & t \geq 0. \end{aligned} \tag{2.1}$$

A log-barrier problem corresponding (2.1) is formulated in which nonnegative slack variables are kept inside the log term

$$\begin{aligned} \min \quad & \mathcal{B}(x, t; \mu) \\ \text{s.t.} \quad & c(x) - t = 0, \end{aligned} \tag{2.2}$$

where $\mathcal{B}(x,t;\mu) = f(x) - \mu \sum_{i=1}^{m} \log(t_i)$ and $\mu > 0$ is the barrier parameter.

The Lagrangian function for barrier problem (2.2) is

$$\mathcal{L}(x,t,\lambda;\mu) = \mathcal{B}(x,t;\mu) - \lambda^\top (c(x) - t)\,. \tag{2.3}$$

For $\mu > 0$, the first order KKT conditions associated to the problem (2.2) are as follows:

$$\left.\begin{aligned} \nabla_x \mathcal{L} &= \nabla f(x) - \nabla c(x)^\top \lambda = 0 \\ \nabla_t \mathcal{L} &= -\mu T^{-1} e + \lambda = 0 \\ \nabla_\lambda \mathcal{L} &= c(x) - t = 0, \end{aligned}\right\} \tag{2.4}$$

where $T = \operatorname{diag}(t_1, t_2 \ldots, t_m)$, $e = (1, 1, \ldots, 1)^\top$ and $\nabla c(x)$ denotes the Jacobian matrix of the vector $c(x)$.

Now, multiplying second equation of (2.4) by T, we obtain the following reduced KKT conditions.

$$\left.\begin{aligned} \nabla f(x) - \nabla c(x)^\top \lambda = 0 \\ -\mu e + T\Lambda e = 0 \\ c(x) - t = 0, \end{aligned}\right\} \tag{2.5}$$

where Λ is again a diagonal matrix with $\lambda_i,\ i = 1, 2, \cdots, m$.

In order to solve (2.5), interior point method utilizes Newton method. Hence, for a given $\mu > 0$, Newton direction $(\Delta x, \Delta t, \Delta\Lambda)$ at point (x, t, λ) is determined by solving the following system for (2.5)

$$\begin{bmatrix} H(x,t) & 0 & -(A(x))^\top \\ 0 & \Lambda & T \\ A(x) & -I & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta t \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} \nabla c(x)^\top \lambda - \nabla f(x) \\ \mu e - T\Lambda e \\ -c(x) + t \end{bmatrix}, \tag{2.6}$$

where

$$H(x,t) = \nabla_{xx} f(x) - \sum_{i=1}^{m} \lambda_i \nabla_{xx} c_i(x) \text{ and } A(x) = \nabla_x c(x).$$

On multiply the first equation of (2.6) by -1 and the second equation by $-T^{-1}$, we get the following system

$$\begin{bmatrix} -H(x,t) & 0 & (A(x))^\top \\ 0 & -T^{-1}\Lambda & -I \\ A(x) & -I & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta t \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} \sigma \\ \gamma \\ \rho \end{bmatrix}, \tag{2.7}$$

where

$$\left.\begin{aligned} \sigma &= \nabla c(x)^\top \lambda - \nabla f(x), \\ \gamma &= \mu T^{-1} e - \lambda, \\ \rho &= c(x) - t. \end{aligned}\right\} \tag{2.8}$$

We denote ρ and σ as the primal infeasibility and dual infeasibility respectively. If ρ is zero at a point, then the point is primal feasible. Also, we refer $p = (x, t)$ as the primal variables. Consider the following measure

$$\nu(p, \lambda) = \max\left\{\|\sigma\|_2, \|\rho\|_2, \|T\Lambda e\|_2\right\}. \tag{2.9}$$

A point (p, λ) is said to KKT point if $\nu(p, \lambda) = 0$. Also, for a predefined accuracy parameter ϵ if $\nu(p, \lambda) < \epsilon$, then we declare that point as a approximated KKT point.

From (2.7), we can eliminate Δt by using the following expression

$$\Delta t = T\Lambda^{-1}(\gamma - \Delta\lambda).$$

Now resulting Newton system is

$$\begin{bmatrix} -H(x,t) & (A(x))^\top \\ A(x) & T\Lambda^{-1} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} \sigma \\ \rho + T\Lambda^{-1}\gamma \end{bmatrix}. \tag{2.10}$$

The system (2.10) has unique solution, when the matrix $H(x, t)$ is positive definite. In case of non positive definite, we perturb the Hessian matrix as $\hat{H} = H + rI$, where $r > 0$ is chosen so that the matrix $\hat{H}$ is positive definite. By solving the system (2.10), we can easily find the direction Δx, $\Delta\lambda$ and Δt. The explicit formulas of the search direction are

$$\left.\begin{aligned} \Delta x &= N^{-1}\left((A(x))^\top (T^{-1}\Lambda\rho + \gamma) - \sigma\right) \\ \Delta t &= -\rho + A(x)\Delta x, \\ \Delta\lambda &= \gamma + T^{-1}\Lambda(-A(x)\Delta x + \rho), \end{aligned}\right\} \tag{2.11}$$

where $N = H(x,t) + (A(x))^\top T^{-1}\Lambda A(x)$. If H is not positive definite, we replace it by $\hat{H}$.

To find a solution of 2.4, the algorithm that we propose here starts from an initial point $\left(x^{(0)}, t^{(0)}, \lambda^{(0)}\right)$; then, at the k-th iteration, it determines a search direction $\left(\Delta x^{(k)}, \Delta t^{(k)}, \Delta\lambda^{(k)}\right)$ by 2.11 at $\left(x^{(k)}, t^{(k)}, \lambda^{(k)}\right)$; lastly, it chooses a step length $\alpha^{(k)}$ and then finds the next iterate by $x^{(k+1)} = x^{(k)} + \alpha^{(k)}\Delta x^{(k)}$, $t^{(k+1)} = t^{(k)} + \alpha^{(k)}\Delta t^{(k)}$ and $\lambda^{(k+1)} = \lambda^{(k)} + \alpha^{(k)}\Delta\lambda^{(k)}$, where the step length $\alpha^{(k)}$ is detailed in the next subsection.

2.1.1 Selection of step length

The proposed algorithm updates the iteration point at the end of each iteration by **??**. When choosing the step length at every iteration, attention must be given so that the vectors t and λ, stay positive across the iterations. For this positivity, we choose the step length α at every iteration by the following standard ratio formula:

$$\alpha \;=\; \min\left\{\delta\left(\max_i\left\{-\frac{\Delta t_i}{t_i}, -\frac{\Delta\lambda_i}{\lambda_i}\right\}\right)^{-1}, 1\right\}, \tag{2.12}$$

where $0 < \delta \leq 1$.

2.2 Merit Function

Over the last two decades, there has been a lot of study on merit functions for constrained nonlinear programming. A merit function ensures that progress toward a local minimizer and feasibility is made in tandem. This progress is accomplished by shortening the steplength along the search directions specified by (8) as required to decrease the merit function sufficiently. One possibility of the merit function can be

$$\Psi_1(x,\beta) = f(x) + \beta\|\rho(x,t)\|_1 \tag{2.13}$$

This merit function is exact, which implies that there exists a β_0 such that, for all $\beta \geq \beta_0$, a minimizer of (6) is guaranteed to be feasible and, under general conditions, a local minimizer of the problem (2.1). Though exactness is a desirable property, the nondifferentiability of ℓ_1-norms can make mathematical calculations challenging. The variety of the smooth merit function is defined as

$$\Psi_2(x, \beta) = f(x) + \frac{\beta}{2}\|\rho(x,t)\|_2^2 \tag{2.14}$$

The merit function Ψ_2 is studied by Fiacco and McCormick.

The ℓ_2 merit function (2.14) for problem (2.2) is

$$\Psi_{\beta,\mu}(x, \omega) = f(x) - \sum_{i=1}^{m}\log(t_i) + \frac{\beta}{2}\|\rho(x,t)\|_2^2, \tag{2.15}$$

where $\rho(x,t) = t - h(x)$.

Theorem 1 shows that for large enough $\beta's$ the search directions defined by (2.11) are descent directions for $\Psi_{\beta,\mu}$ whenever the problem is $H(x,t)$ is positive definite.

Theorem 1 Let the matrix N is positive definite, Then there exist $\beta_{\min} \geq 0$ such that, for each $\beta > \beta_{\min}$ the search directions $(\Delta x, \Delta t)$ are descent for the merit function $\Psi_{\beta,\mu}$ i.,e.,

$$\begin{bmatrix}\nabla_x \Psi_{\beta,\mu} \\ \nabla_t \Psi_{\beta,\mu}\end{bmatrix}^\top \begin{bmatrix}\Delta x \\ \Delta t\end{bmatrix} < 0.$$

Proof The gradient of merit function with respect to x and t are

$$\begin{bmatrix}\nabla_x \Psi_{\beta,\mu} \\ \nabla_t \Psi_{\beta,\mu}\end{bmatrix} = \begin{bmatrix}\nabla_x f(x) - \beta(A(x))^\top \rho \\ -\mu T^{-1}e + \beta\rho\end{bmatrix}.$$

Now,

$$\begin{aligned}\begin{bmatrix}\nabla_x \Psi_{\beta,\mu} \\ \nabla_t \Psi_{\beta,\mu}\end{bmatrix}^\top \begin{bmatrix}\Delta x \\ \Delta t\end{bmatrix} =& - \left(\nabla_x f(x) - \mu(A(x))^\top T^{-1}e\right)^\top N^{-1}\left(\nabla_x f(x) - \mu(A(x))^\top T^{-1}e\right) \\ &+ \mu e^\top T^{-1}\rho + \left(\nabla_x f(x) - \mu(A(x))^\top T^{-1}e\right)^\top N^{-1}(A(x))^\top T^{-1}\Lambda\rho \\ &- \beta\|\rho\|^2.\end{aligned}$$

Now, two cases arise

1. When the term of the last expression

$$\Gamma = \mu e^\top T^{-1}\rho + \left(\nabla_x f(x) - \mu(A(x))^\top T^{-1}e\right)^\top N^{-1}(A(x))^\top T^{-1}\Lambda\rho$$

is negative then

$$\begin{bmatrix} \nabla_x \Psi_{\beta,\mu} \\ \nabla_t \Psi_{\beta,\mu} \end{bmatrix}^\top \begin{bmatrix} \Delta x \\ \Delta t \end{bmatrix} < 0.$$

Hence $(\Delta x, \Delta t)$ is descent direction for the merit function $\Psi_{\beta,\mu}$.

2. If the term

$$\Gamma = \mu e^\top T^{-1}\rho + \left(\nabla_x f(x) - \mu(A(x))^\top T^{-1} e\right)^\top N^{-1}(A(x))^\top T^{-1}\Lambda\rho > 0,$$

then we set

$$\beta_{\min} = \frac{-\left(\nabla_x f(x) - \mu(A(x))^\top T^{-1}e\right)^\top N^{-1}\left(\nabla_x f(x) - \mu(A(x))^\top T^{-1}e\right) + \Gamma}{\|\rho\|^2}.$$

Hence, in this case, we can choose a $\beta > \beta_{\min}$ such that

$$\begin{bmatrix} \nabla_x \Psi_{\beta,\mu} \\ \nabla_t \Psi_{\beta,\mu} \end{bmatrix}^\top \begin{bmatrix} \Delta x \\ \Delta t \end{bmatrix} < 0.$$

.

2.2.1 Selection of Barrier Parameter

Interior point method keeps changing the value of barrier parameter at every point of iteration. Traditionally the value of barrier parameter is chosen as

$$\mu = r\frac{T^\top \lambda}{m}, \tag{2.16}$$

where $r \in (0, 1)$.

We propose the following interior-point algorithm with a backtracking line-search algorithm for the nonlinear optimization problem:

2.2.2 Algoritm 1

• Initialization

Give an initial point $(\mathrm{x}^{(0)}, t^{(0)}, \lambda^{(0)})$ such that $t^{(0)}, \lambda^{(0)} > 0$.
Give the values of parameters $r \in (0, 1)$ and $0 < \kappa < 1$.

Give a value of the precision parameter $\epsilon > 0$. Set $k = 0$.

• **Main Steps**

while $\nu(p^{(k)}, \lambda^{(k)}) \geq \epsilon$

$\mu^{(k)}$ according to (2.16).

Calculate the search directions $(\Delta x^{(k)}, \Delta t^{(k)}, \Delta\lambda^{(k)})$ by (2.11).

Set $\beta = 10\beta_{\min}$ to guarantee that $(\Delta x^{(k)}, \Delta t^{(k)}, \Delta\lambda^{(k)})$ are descent for Ψ.

Choose step length α by the formula (2.12)

Set $p^{(k+1)} = p^{(k)} + \alpha\Delta p^{(k)}$, $\lambda^{(k+1)} = \lambda^{(k)} + \alpha\Delta\lambda^{(k)}$

Find $\alpha^{(k)} \in (0, \alpha)$ such that the following Armijo condition satisfied

$$\Psi_{\eta,\mu^{(k)}}(p^{(k+1)}) \leq \Psi_{\eta,\mu^{(k)}}(p^{(k)}) + \alpha^{(k)}\kappa\left(\nabla\Psi_{\eta,\mu^{(k)}}\right)^{\top}\Delta p^{(k)}.$$

Set $p^{(k+1)} = p^{(k)} + \alpha^{(k)}\Delta p^{(k)}$, $\lambda^{(k+1)} = \lambda^{(k)} + \alpha^{(k)}\Delta\lambda^{(k)}$

• **end while**

• **return** *minimum point* x^*.

Chapter 3

Numerical Experiments

3.1 Numerical Implementation

In this part, we report some mathematical investigations for the proposed Algorithm 1(2.2.2). The programs are written in MATLAB R2020a and run on a machine with an Intel Core i3 7020 2.30GHz CPU and 3.00GB RAM. We used the following value of the parameters $\epsilon = 10^{-6}$, $\kappa = 0.1$, $r = 0.01$.

In the next part, we take some test problems to test the exhibition of the Algorithm 1(2.2.2). Details of these test problems are described in Table 3.1.

3.1.1 Test Problems

3.1.1.1 Test 1

$$\begin{aligned} \min \quad & (x_1 + x_2^2 - 7)^2 + (x_1^2 + x_2 - 11)^2 \\ \text{s.t.} \quad & (x_1 - 0.05)^2 + (x_2 - 2.5)^2 - 4.84 \leq 0, \\ & 4.84 - (x_1)^2 - (x_2 - 2.5)^2 \leq 0, \\ & x_1,\ x_2 \in [0, 6]. \end{aligned}$$

3.1.1.2 Test 2

$$\begin{aligned} \text{Minimize} \quad & 6(x_1 - 10)^2 + 4(x_2 - 12.5)^2, \\ \text{Subject to} \quad & x_1^2 + (x_2 - 5)^2 \leq 50, \\ & x_1^2 + 3x^2 \leq 200, \\ & (x_1 - 6)^2 + x_2^2 \leq 37. \end{aligned}$$

Table 3.1: Data for the test problems performed by the Algorithm 1.

Problem	no. of variables (n)	no. of constraints (m)	iterations	x^*	$f(x^*)$
Test 1	2	2	9	$(2.2468, 2.3818)^\top$	13.5898
Test 2	2	3	8	$(6.9999, 5.9999)^\top$	223.0011
Test 3	2	1	13	$(0.9999, 0.9999)^\top$	0.0000
Test 4	2	1	10	$(0.9999, 0.9999)^\top$	0.0000

3.1.1.3 Test 3

$$\begin{aligned} \min \quad & (1-x_1)^2 + 100(x_2 - x_1^2)^2, \\ \text{s.t.} \quad & (x_1-1)^3 - x_2 + 1 \leq 0, \\ & -1.5 \leq x_1 \leq 1.5, \\ & -0.5 \leq x_2 \leq 2.5 \end{aligned}$$

3.1.1.4 Test 4

$$\begin{aligned} \min \quad & (1-x_1)^2 + 100(x_2 - x_1^2)^2, \\ \text{s.t.} \quad & x_1^2 + x_2^2 \leq 2, \\ & -1.5 \leq x_1 \leq 1.5, \\ & -1.5 \leq x_2 \leq 1.5 \end{aligned}$$

3.1.2 Real World Problems

3.1.2.1 *Welded Beam Design Problem (WBDP) [13]*

The aim of this problem is to minimize the expense of the welded beam when taking into account constraints such as end deflection of the beam (δ), shear stress (θ), bucking load on the bar (P_c), bending stress in the beam (σ), and side constraints. Mathematically, this problem can be written as:

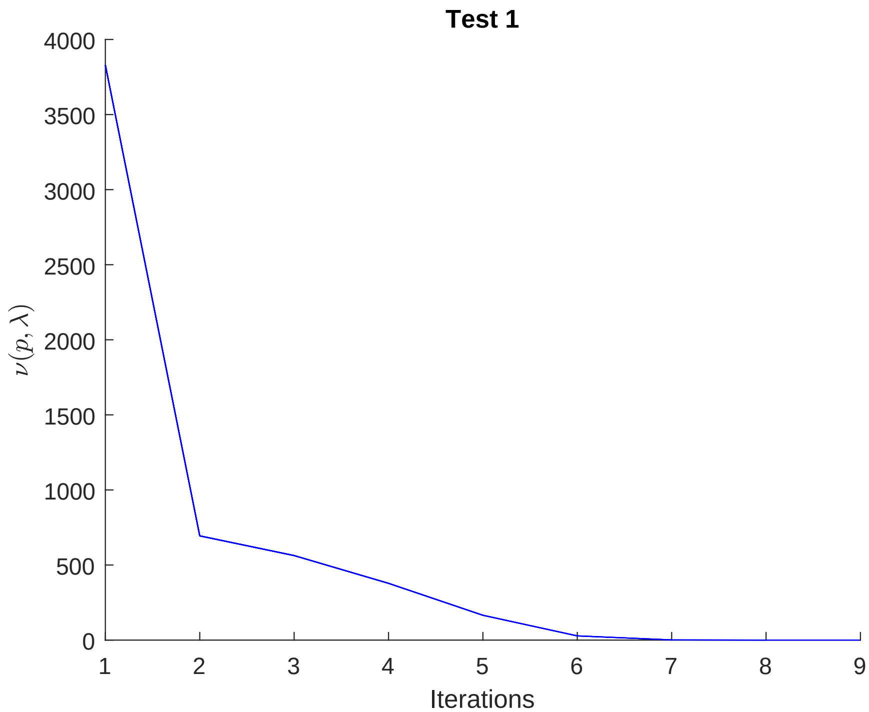

Figure 3.1: Convergence of Algorithm 1 for Test1

$$\left.\begin{array}{ll} \min & 0.04811x_3x_4(14.0+x_2)1.1047x_1^2x_2 \\ \text{s.t.} & \theta(x) \le \theta_{\max} \\ & \sigma(x) \le \sigma_{\max} \\ & x_1 - x_4 \le 0 \\ & P \le P_c(x) \\ & \delta(x) \le \delta_{\max} \\ & x_1,\ x_4 \in [0.1, 2] \ \text{ and } \ x_2,\ x_3 \in [0.1, 10], \end{array}\right\}$$

where

$$\theta(x) = \sqrt{((\theta'(x))^2 + ((\theta''(x))^2 + \frac{x_2(\theta'(x)(\theta''(x))}{\sqrt{0.25[x_2^2 + (x_1 + x_3)^2]}}},$$

$$\sigma(x) = \frac{6PL}{x_3^2x_4}, \delta(x) = \frac{4PL^3}{Ex_3^2x_4},$$

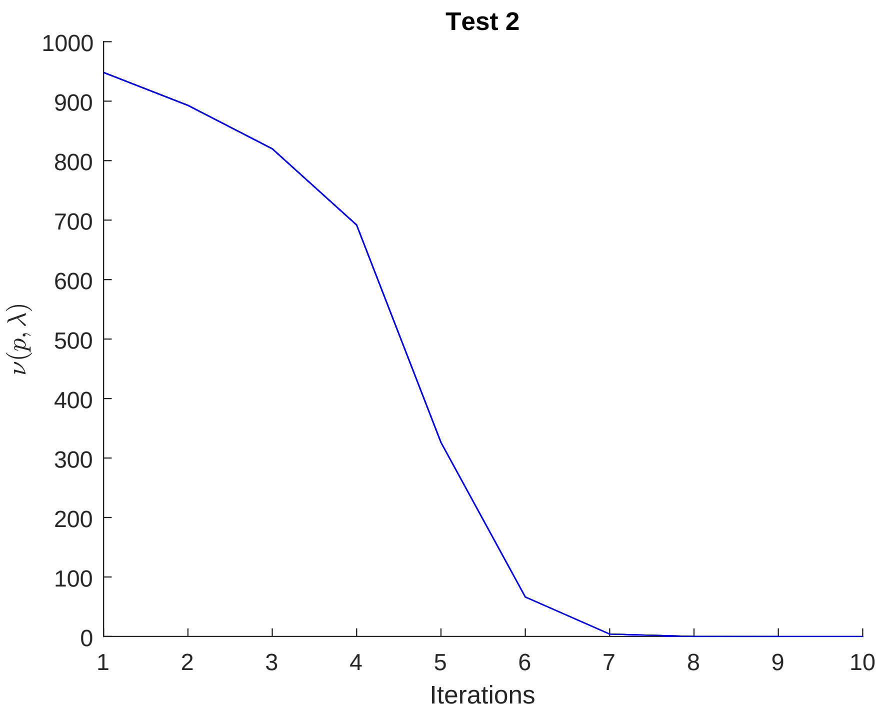

Figure 3.2: Convergence of Algorithm 1 for Test 2

$$P_c(\mathrm{x}) = \frac{4.013E\sqrt{\frac{\mathrm{x}_3^2\mathrm{x}_4^6}{36}}}{L^2}\left[1 - \frac{\mathrm{x}_3}{2L}\sqrt{\frac{E}{4G}}\right]$$

where

$$\theta'(\mathrm{x}) = \frac{P}{\sqrt{2}\mathrm{x}_1\mathrm{x}_2}, \quad \theta''(\mathrm{x}) = \frac{MR}{J}$$

$$M = [L + 0.5\mathrm{x}_2]\,P, \;\; R = \sqrt{0.25\mathrm{x}_2^2 + 0.25(\mathrm{x}_1 + \mathrm{x}_3)^2}$$

$$J = \; 2\sqrt{2}\,\mathrm{x}_2\mathrm{x}_1\left(\mathrm{x}_2^2/12 + 0.25(\mathrm{x}_1 + \mathrm{x}_3)^2\right)$$

$P = 6000\text{lb}$, $E = 30106\text{psi}$, $L = 14\text{in}$, $G = 1210^6\text{psi}$, $\theta_{\max} = 13600\text{psi}$, $\delta_{\max} = 0.25\text{in}$ $\sigma_{\max} = 30000\text{psi}$.

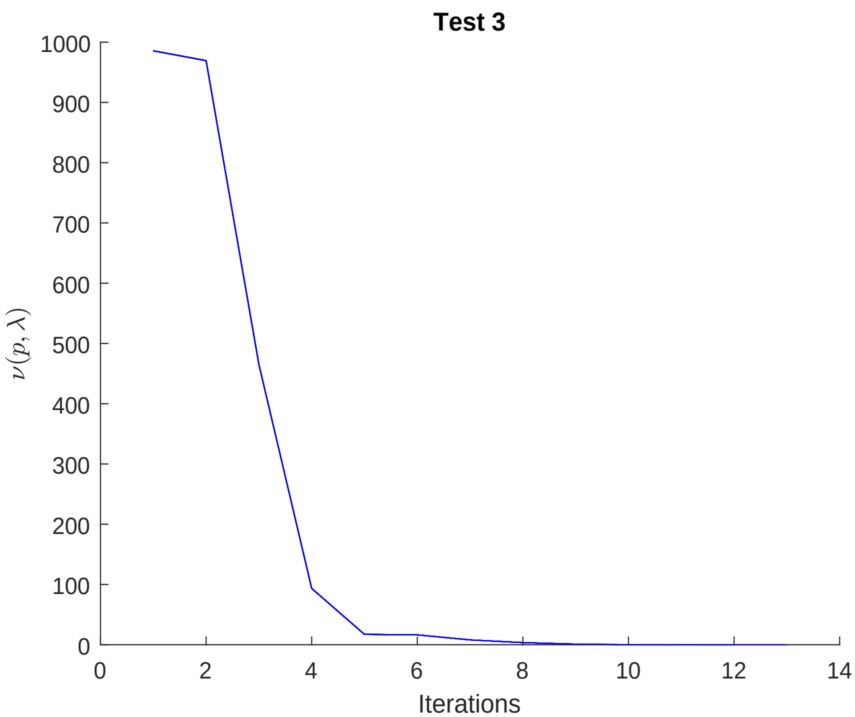

Figure 3.3: Convergence of Algorithm 1 for Test 3

3.1.2.2 *Weight of a Tension /Compression Spring Problem (WSP) [14]*

The aim of this problem is to reduce the weight of a tension as much as possible with the restrictions: minimum deflection, limits on outside, surge frequency, diameter, shear stress (see Figure 5.7). The problem formulation is as follows:

$$
\begin{aligned}
\text{min} \quad & x_1^2 x_2 (2 + x_3), \\
\text{s.t.} \quad & \frac{x_2^3 x_3}{(71785 x_1^4)} - 1 \geq 0, \\
& 1 - \frac{4x_2^2 - x_1 x_2}{12566 x_1^3 (x_2 - x_1)} - \frac{1}{5108 x_1^2} \frac{x_2^3 x_3}{(71785 x_1^4)} - 1 \geq 0, \\
& \frac{140.45 x_1}{x x_3 x_2^2} - 1 \frac{x_2^3 x_3}{(71785 x_1^4)} - 1 \geq 0, \\
& 1 - \frac{x_1 + x_2}{1.5 - 1} \frac{x_2^3 x_3}{(71785 x_1^4)} - 1 \geq 0,
\end{aligned}
$$

and $x_1 \in [0.05, 2], \;\; x_2 \in [0.25, 1.3], \;\; x_3 \in [2, 15]$.

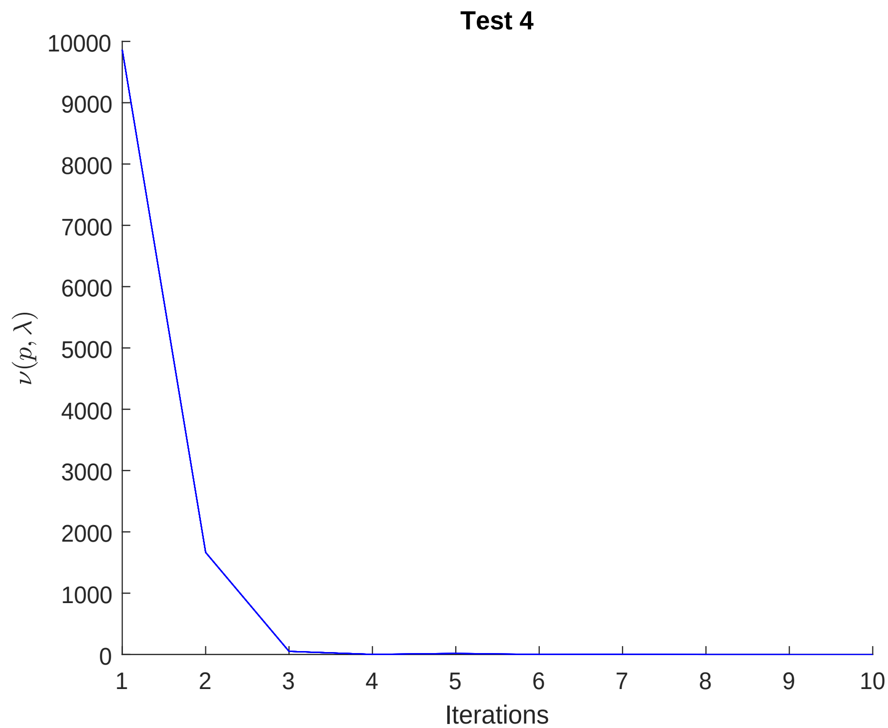

Figure 3.4: Convergence of Algorithm 1 for Test 4

3.1.2.3 *Three Bar Truss Problem (TBTB) [14]*

In this problem, three bars should be put as seen in Figure 5.9. The goal is to keep the weight of the bars in this position as low as possible. This dilemma has the following mathematical expression:

$$\begin{aligned} \min \quad & (2\sqrt{2}x_1 + x_2)L \\ \text{s.t.} \quad & \sigma - \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P \geq 0 \\ & \sigma - \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P \geq 0 \\ & \sigma - \frac{1}{x_1 + \sqrt{2}x_2}P \geq 0 \\ & 0 \leq x_1, x_2 \leq 1. \end{aligned}$$

The constants are $L = 100cm$, $\sigma = 2KN/cm^2$ and $P = 2KN/cm^2$.

Table 3.2: Optimal value for WBDP of different algorithm and Algorithm 1

Method	Author	$f^*(x)$
Interior-point method	This paper	1.72485
Self-adaptive penalty approach	Coello	1.74830
Constraint correction at constant cost	Arora	2.43311
CPSO	He and Wang	1.728024
Geometric programming	Ragsdell and Phillips	2.38593
GA	Deb	2.43311
Feasibility-based tournament selection	Coello and Montes	1.72822
Modified PSO	Ebehart	1.72485

Table 3.3: Optimal value for WSB of different algorithm and Algorithm 1.

Author	Method	$f(x^*)$
This paper	Interior-point method	0.01266
He and Wang	CPSO	0.01267
Ebehart	Modified PSO	0.01266
Coello	Self-adaptive penalty approach	0.01270
Belegundu	Numerical optimization technique	0.01283
Arora	Constraint correction at constant cost	0.12730
Coello and Montes	Feasibility-based tournament selection	0.01268

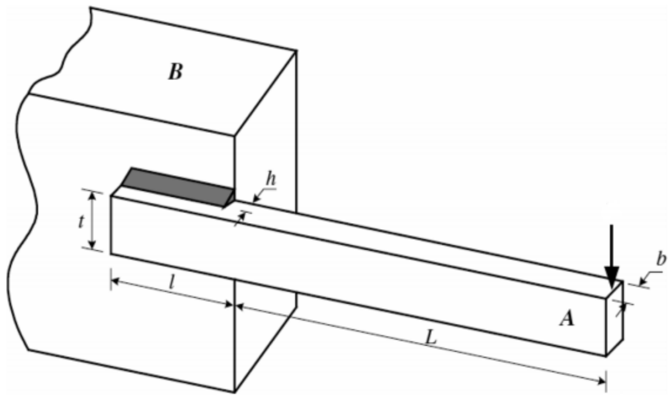

Figure 3.5: Welded Beam Design

Table 3.4: Optimal value for TBTB of different algorithm and Algorithm 1.

Algorithm name	optimal point	optimal value
Cricket algorithm	(0.7886, 0.4083)	263.8958
Bat algorithm	(0.7886, 0.4082)	263.8958
Swarm optimization approach	(0.7950, 0.3950)	264.3000
Mine blast algorithm	(0.7885, 0.4082)	263.8958
Cuckoo search algorithm	(0.7886, 0.4090)	263.9716
Artificial atom algorithm (A3)	(0.7887, 0.4080)	263.8958
Interior-point method	(0.7886, 0.4082)	263.8956

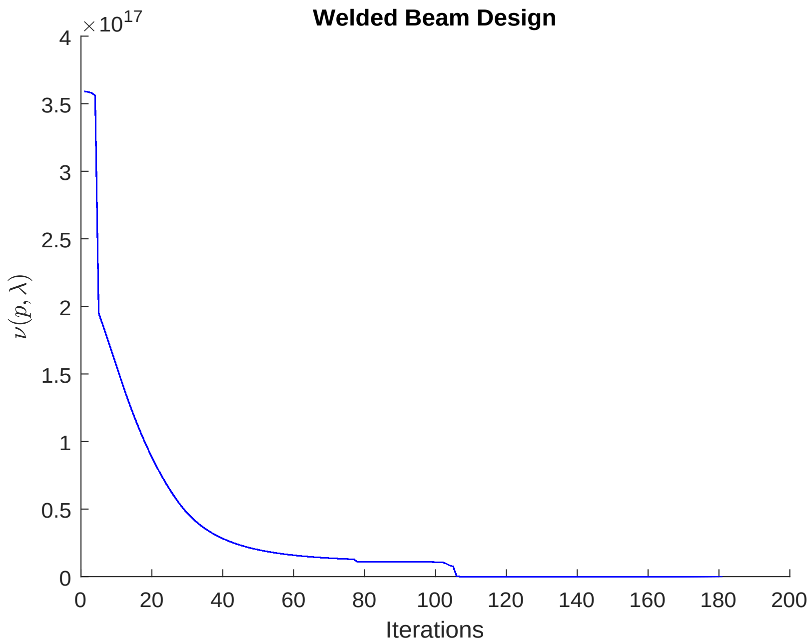

Figure 3.6: Convergence of Algorithm 1 for Welded Beam Design Problem

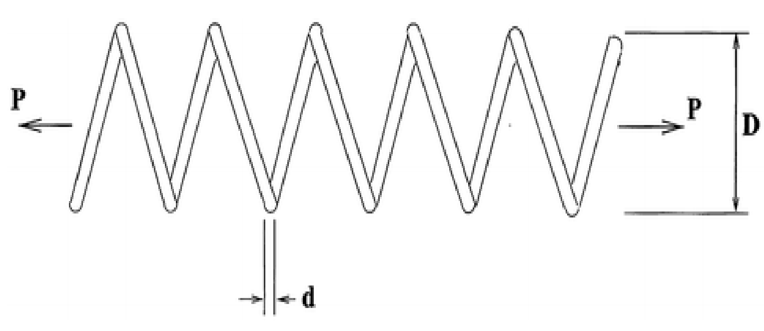

Figure 3.7: Tension/compression string problem

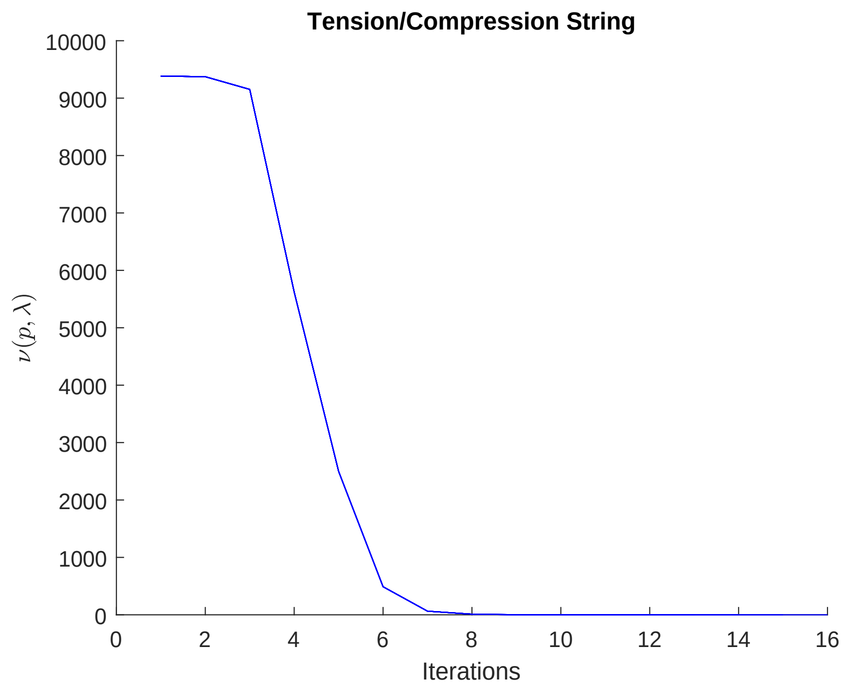

Figure 3.8: Convergence of Algorithm 1 for Tension string problem

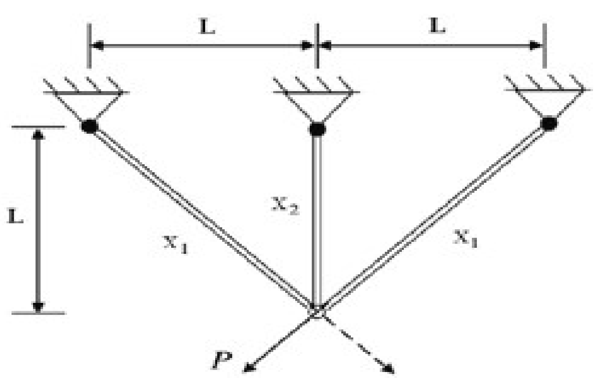

Figure 3.9: Three Bar Truss Problem

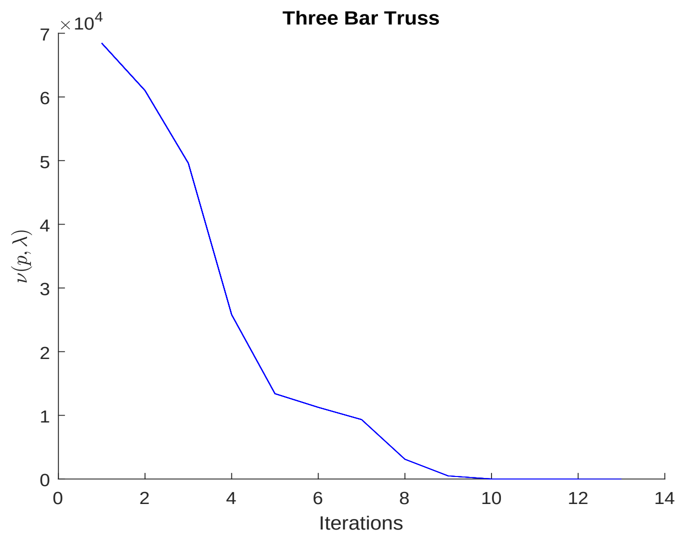

Figure 3.10: Convergence of Algorithm 1 for Three Bar Truss Problem

Chapter 4

Conclusion

In this project, a system of perturbed KKT systems are solved with the help of IPM. IPM utilizes Newton method to find the direction along which to proceed. A merit function is used to take appropriate steplength along the search direction. Hence, proposed algorithm gradually reduces $\nu(p, \lambda)$ as the iterations increase. The efficiency of the proposed algorithims tested on some test problems and real worlds problems. The numerical outcomes have shown that the proposed algorithm is able to solve constrained problems and real-world problems efficiently.

Chapter 5

MATLAB CODE (Test 1)

x = [x1; x2];

X = diag(x);

w = [w1;w2;w3;w4;w5;w6];

W = diag(w);

y = [y1;y2;y3;y4;y5;y6];

Y = diag(y);

e = [1;1;1;1;1;1];

I = diag(e);

I1=diag([1,1]);

f = (x1*x1 + x2 -11)*(x1*x1 + x2 -11) + (x1+x2*x2 -7)*(x1+x2*x2 -7);

x0 = [1;1];

w0 = ones(6,1);

y0 = 1./w0;

h1 = 4.84 - (x1 - 0.05)$^2 - (x2 - 2.5)^2$;

$h2 = x1 * x1 + (x2 - 2.5) * (x2 - 2.5) - 4.84$;

$h3 = x1$;

$h4 = x2$;

$h5 = 6 - x1$;

$h6 = 6 - x2$;

$h = [h1; h2; h3; h4; h5; h6]$;

$gradf = gradient(f, x)$;

$b = f - k * (sum(log(w(1:6))))$;

$grad_b x = gradient(b, x)$;

$grad_b w = gradient(b, w)$;

$H1 = 0;$

$fori = 1 : 6$

$H1 = H1 - y(i). * hessian(h(i), x);$

end

$H = hessian(f, x) + H1;$

$H0 = vpa(subs(H, [x; y], [x0; y0]));$

$A = vpa(jacobian(h, x));$

$sigma = gradf - transpose(A) * y;$

$sgm = gradf - k * transpose(A) * inv(W) * e;$

$sigma_1 = subs(sigma, [x; y], [x0; y0])$

$gamma = k * inv(W) * e - y;$

$game = W * Y * e;$

$game_1 = subs(game, [w; y], [w0; y0])$

$rho = w - h;$

$rho_1 = subs(rho, [x; w], [x0; w0])$

$S = b + (t/2) * transpose(rho) * rho;$

$grad_S x = gradient(S, x);$

$grad_S w = gradient(S, w);$

$N = H + transpose(A) * inv(W) * Y * A;$

$d1 = vpa(transpose(rho_1) * rho_1);$

$d2 = transpose(game_1) * game_1;$

$d3 = transpose(sigma_1) * sigma_1;$

$mer_1 = max([d1, d2, d3]);$

$count = 0;$

$fori = 1 : 80$

$disp('enteredinforloop(numberofiteration)i =')$

$disp(i)$

$count = count + 1$

$xk = x0;$

$wk = w0;$

$yk = y0;$

$Wk = diag(wk);$

$Yk = diag(yk);$

$A1 = subs(A, x, xk);$

$rho1 = subs(rho, [x; w], [xk; wk]);$
$sigma1 = subs(sigma, [x; y], [xk; yk]);$
$k1 = 0.01 * (transpose(wk) * yk)/(10)$
$sgm1 = subs(sgm, [x; w; k], [xk; wk; k1]);$
$grad_b x1 = subs(grad_b x, [x; k], [xk; k1]);$
$grad_b w1 = subs(grad_b w, [w; k], [wk; k1]);$
$grad_S x1 = subs(grad_S x, [x; w; k], [xk; wk; k1]);$
$grad_S w1 = subs(grad_S w, [x; w; k], [xk; wk; k1]);$
$H1 = subs(H, [x; y], [xk; yk]);$
$N1 = subs(N, [x; w; y], [xk; wk; yk]);$
$lamb_N = eig(N1);$
$pos = min(lamb_N);$
$if(pos < 0)$
$N1 = N1 + (abs(pos) + 1) * I1;$
end
$delx = k * inv(N1) * transpose(A) * inv(W) * e - inv(N1) * gradf + inv(N1) * transpose(A) * inv(W) * Y * rho;$
$delw = k * A * inv(N1) * transpose(A) * inv(W) * e - A * inv(N1) * gradf - (I - A * inv(N1) * transpose(A) * inv(W) * Y) * rho;$
$dely = -inv(W) * Y * delw + gamma;$
$delx1 = subs(delx, [x; w; y; k], [xk; wk; yk; k1]);$
$delw1 = subs(delw, [x; w; y; k], [xk; wk; yk; k1]);$
$dely1 = subs(dely, [x; w; y; k], [xk; wk; yk; k1]);$
$d1 = [delx1; delw1];$
$db1 = [grad_b x1; grad_b w1];$
$ppp = transpose(db1) * d1$
$pp3 = -transpose(sgm1) * inv(N1) * sgm1 + k1 * transpose(e) * inv(Wk) * rho1 + transpose(sgm1) * inv(N1) * transpose(A1) * inv(Wk) * Yk * rho1;$
$if(transpose(db1) * d1 < 0)$
$disp('umhhbetaiszerohereandxkisinfeasible')$
$t1 = 0;$
$a1 = -(delw1./wk);$

a2=-(dely1./yk);
a3=max([a1 ;a2]);
a4=min([1,0.95/a3]) ;
x$_new = vpa(x0 + a4 * delx1);$
$w_new = vpa(w0 + a4 * delw1);$
$y_new = vpa(y0 + a4 * dely1);$
$bk = subs(b, [x; w; k], [xk; wk; k1])$
$b_new = subs(b, [x; w; k], [x_new; w_new; k1])$
$while(b_new > bk + a4 * 0.01 * ppp)$
$disp('shorteningthestep')$
$a_11 = a4;$
$a11_1 = 0.05 * a_11;$
$x_new = xk + a11_1 * delx1$
$w_new = wk + a11_1 * delw1$
$y_new = yk + a11_1 * dely1;$
$bk_new = subs(b, [x; w; k], [x_new; w_new; k1])$
$b_new = vpa(bk_new);$
$a4 = a11_1;$
end
$x0 = x_new;$
$w0 = w_new;$
$y0 = y_new;$
$else$

if (pp3¡0)
t1=0;
a1=-(delw1./wk);

a2=-(dely1./yk);
a3=max([a1 ;a2]);
a4=min([1,0.95/a3]) ;
x$_new = vpa(x0 + a4 * delx1);$
$w_new = vpa(w0 + a4 * delw1);$
$y_new = vpa(y0 + a4 * dely1);$

$bk = subs(b, [x; w; k], [xk; wk; k1]);$
$b_new = subs(b, [x; w; k], [x_new; w_new; k1]);$
$while(b_new > bk + a4 * 0.01 * ppp)$
$disp('shorteningthestep')$
$a_11 = a4;$
$a11_1 = 0.05 * a_11;$
$x_new = xk + a11_1 * delx1;$
$w_new = wk + a11_1 * delw1;$
$y_new = yk + a11_1 * dely1;$
$bk_new = subs(b, [x; w; k], [x_new; w_new; k1]);$
$b_new = vpa(bk_new);$
$a4 = a11_1;$
end
$x0 = x_new;$
$w0 = w_new;$
$y0 = y_new;$
$else$
$disp('nowcalculatebetaandxkisinfeasible')$
$t2 = pp3/(transpose(rho1) * rho1);$
$t1 = 10 * t2;$
$S = b + (t/2) * transpose(rho) * rho;$
$grad_Sx = gradient(S, x);$
$grad_Sw = gradient(S, w);$
$grad_Sx1 = subs(grad_Sx, [x; w; k; t], [xk; wk; k1; t1]);$
$grad_Sw1 = subs(grad_Sw, [x; w; k; t], [xk; wk; k1; t1]);$
$db2 = [grad_Sx1; grad_Sw1];$
$kkk4 = transpose(db2) * d1$
$khg = pp3 - t1 * transpose(rho1) * rho1$
$if(transpose(db2) * d1 < 0)$
$disp('calculatedbeta')$
$a1 = -(delw1./wk);$

a2=-(dely1./yk);
a3=max([a1 ;a2]);

a4=min([1,0.95/a3]);

$x_n ew = vpa(x0 + a4 * delx1);$

$w_n ew = vpa(w0 + a4 * delw1);$

$y_n ew = vpa(y0 + a4 * dely1);$

$Sk = subs(S, [x; w; t; k], [xk; wk; t1; k1]);$

$S_n ew = subs(S, [x; w; t; k], [x_n ew; w_n ew; t1; k1]);$

$while(S_n ew > Sk + a4 * 0.01 * kkk4)$

$disp('shorteningthestep')$

$a_1 1 = a4;$

$a11_1 = 0.05 * a_1 1;$

$w_n ew = wk + a11_1 * delw1;$

$y_n ew = yk + a11_1 * dely1;$

$S_n ew1 = subs(S, [x; w; k; t], [x_n ew; w_n ew; k1; t1]);$

$S_n ew = vpa(S_n ew1);$

$a4 = a11_1;$

end

$x0 = x_n ew;$

$w0 = w_n ew;$

$y0 = y_n ew;$

$else$

end

end

end

$rho2 = subs(rho, [x; w], [x0; w0]);$

$sigma2 = subs(sigma, [x; y], [x0; y0]);$

$gamma2 = subs(game, [w; y; k], [w0; y0; k1]);$

$norm = transpose(rho2) * rho2 + transpose(sigma2) * sigma2 + transpose(gamma2) * gamma2$

$SAI1 = vpa(subs(S, [x; w; k; t], [x0; w0; k1; t1]));$

$bar = subs(b, [x; w; k; t], [x0; w0; k1; t1]);$

$holdon$

$xlabel('Iterations')$

$ylabel('\nu(p, \lambda)'$,'interpreter','latex')

title('Test 1')

Dh(i) = norm;
plot(1:i,Dh(1:i),’b-’)
if (norm¡=0.00001)
disp(xk)
break
end
x0=x$_{n}ew;$
$w0 = w_{n}ew;$
$y0 = y_{n}ew;$
end
$disp(x0)$
$fun_{v}al = subs(f, x, x0)$

Bibliography

[1] Khachiyan, Leonid Genrikhovich. ”A polynomial algorithm in linear programming.” Doklady Akademii Nauk. Vol. 244. No. 5. Russian Academy of Sciences, 1979.

[2] Karmarkar, Narendra. ”A new polynomial-time algorithm for linear programming.” Proceedings of the sixteenth annual ACM symposium on Theory of computing. 1984.

[3] Lustig, Irvin J. ”Feasibility issues in a primal-dual interior-point method for linear programming.” Mathematical Programming 49.1 (1990): 145-162.

[4] Wright, Stephen J. Primal-dual interior-point methods. Society for Industrial and Applied Mathematics, 1997.

[5] Yuan, Gonglin, and Zengxin Wei. ”New line search methods for unconstrained optimization.” Journal of the Korean Statistical Society 38.1 (2009): 29-39.

[6] Gertz, Edward Michael. Combination trust-region line-search methods for unconstrained optimization. University of California, San Diego, 1999.

[7] Yuan, Ya-xiang. ”A new stepsize for the steepest descent method.” Journal of Computational Mathematics (2006): 149-156.

[8] Yuan, Gonglin, Xiwen Lu, and Zengxin Wei. ”A conjugate gradient method with descent direction for unconstrained optimization.” Journal of Computational and Applied Mathematics 233.2 (2009): 519-530.

[9] Fischer, Andreas. ”A special Newton-type optimization method.” Optimization 24.3-4 (1992): 269-284.

[10] Yin Zhang. On the convergence of a class of infeasible interior-point methods for the horizontallinear complementarity problem.SIAMJournalonOptimization, 4(1):208–227, 1994.

[11] R J Vanderbei. An interior point code for quadratic programming.PrincetonUniversity,Princeton,NJ,USA, 1994

[12] Robert J Vanderbei and David F Shanno. An interior-point algorithm for nonconvex nonlinear programming Computational Optimization and Applications, 13(1-3):231–252, 1999.

[13] Yokota, Takao, Takeaki Taguchi, and Mitsuo Gen. "A solution method for optimal cost problem of welded beam by using genetic algorithms." Computers & industrial engineering 37.1-2 (1999): 379-382.

[14] Amer, Noor Hafizah, Nurhidayati Ahmad, and Amar Faiz Zainal Abidin. "Weight Minimization of Helical Compression Spring Using Gravitational Search Algorithm (GSA)." Applied Mechanics and Materials. Vol. 773. Trans Tech Publications Ltd, 2015.

[15] Yildirim, Ayşe Erdoan, and Ali Karci. "Application of Three Bar Truss Problem among Engineering Design Optimization Problems using Artificial Atom Algorithm." 2018 International Conference on Artificial Intelligence and Data Processing (IDAP). IEEE, 2018.