

A Stochastic Trust Region Method for Stochastic Optimization Problem with Equality Constraints

Youmeng He¹, Dan Xue*, Donglei Liu

¹School of Mathematics and Statistics, Qingdao University, Qingdao 266071, China

Abstract: A stochastic trust region algorithm is proposed to solve constrained minimization problems with stochastic objectives. This method can be used to deal with nonconvex problems. The new iterate is generated by minimizing an exact penalty function, whose size is controlled by a trust-region type parameter. The stochastic gradient is used to take the place of deterministic gradient for the determination of descent directions of the penalty function. The convergence of the method is discussed under some reasonable conditions. Some preliminary numerical results show that our method can effectively solve the stochastic nonlinear programming problems.

Keywords: Stochastic Programming; Nonlinear Programming; Stochastic Gradient; Trust Region Methods; Penalty Function

Received 15 Jan 2021, Revised 20 May 2021, Accepted 22 June 2021

*Corresponding Author: Dan Xue, wxuedan@126.com

1. Introduction

In this paper, we consider the following stochastic nonlinear programming problem:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} f(x) \\ \text{s.t. } c(x) := (c_1(x), \dots, c_q(x))^T = 0, \end{aligned} \quad (1.1)$$

where both $f: \mathbb{R}^n \rightarrow \mathbb{R}$ and $c: \mathbb{R}^n \rightarrow \mathbb{R}^m$ are continuous differentiable but possibly nonconvex. We assume that the function values and gradients of $c_i(x), i = 1, \dots, m$ can be accurately obtained. The problem (1.1) appears in many practical applications, such as mixed logit modeling problems in economics and transportation [2,4,17], machine learning [18], simulation-based optimization [10]. Many problems in these fields have the following objective functions:

$$f(x) = \mathbb{E}_\xi [F(x, \xi)] \text{ or } f(x) = \int_{\Theta} F(x, \xi) dP(\xi), \quad (1.2)$$

where ξ denotes the random variable whose distribution P is supported on Θ and $\mathbb{E}_\xi [\cdot]$ means that the expectation is taken with respect to ξ . Due to the fact that the integral is difficult to evaluate, or function $F(\cdot, \xi)$ is not given explicitly, the function values and gradients of f are not easily obtainable and only noisy information is available.

Stochastic programming has been studied for several decades. Dantzig and Ferguson proposed a stochastic programming algorithm for solving maximize airline profits. In recent years, the constraint optimization

problems have attracted more and more attention. Conn et al.[15] and Curtis et al.[16] proposed a trust region algorithm, it can escape saddle points and find a local minimum by iterating the variable along the direction related to the eigenvector of the Hessian with the most negative eigenvalue. There has been extensive research when $f(x)$ is convex [8,11,12]. Due to the increasing popularity of deep learning, the nonconvex case has attracted significant attention. See e.g., [5,7,9] for results on the smooth nonconvex case (i.e., $h(x) \equiv 0$). For the more general nonsmooth nonconvex case, the research is still somewhat limited. There are lots of nonconvex optimization problems in practical applications. So, in this paper, we would like to exploit an effective method for solving more general stochastic optimization problems. The objective functions can be nonconvex.

Ghadimi et al. [5] analyzed the deterministic proximal gradient method for nonconvex nonsmooth problems. Wang, Yuan et al. [6] proposed a penalty method with stochastic first-order information for solving stochastic nonconvex problem. Curtis et al. [13] proposed a trust-region method (first-order) for solving nonconvex problem. The trust region methods are deemed to be invaluable tools for solving nonlinear and nonconvex optimization problems [6]. We shall propose a stochastic trust region method. In our methods, we minimize a penalty function of the objective function in the trust region.

A trust region algorithm with stochastic first-order information could help for solving equation constraint(1.1). Similar to classical trust region methods, we used a suitable model to replace the complex objective function. Due to the equality constraints of our problem, we minimized a penalty function in the trust region. The stochastic gradient was used to take place of

deterministic gradient for the determination of descent directions of the penalty function. Stochastic gradients were computable at manageable cost. Stochastic trust region method will be feasible in practice. Convergence theory and numerical results verified the reliability of our method to solve stochastic optimization problems with general constraints.

2. Stochastic Trust Region Algorithm

We adopt the following notation throughout the paper. $\nabla f(x)$ denotes the gradient of f and

$$J(x) := \nabla c(x) = (\nabla c_1(x), \dots, \nabla c_q(x))^T \tag{2.1}$$

denotes the Jacobian matrix of c . Without specification, $\|\cdot\|$ represents the Euclidean norm $\|\cdot\|_2$ in R^n .

Before we present the stochastic trust region algorithm for solving the problem (1.1), we consider the following penalty function:

$$\Phi_h = f(x) + h(c(x)), \tag{2.2}$$

where h is convex but may be nonsmooth function, and where f and c are continuously differentiable throughout the domain of interest but may be nonconvex.

Similar as the deterministic penalty method in references [6], let $h(c(x)) = \rho \|c(x)\|$. Hence, h is convex and Lipschitz continuous with Lipschitz constant $L_h = \rho$. It means, $\Phi_h = \Phi(\cdot, \rho)$, where $\Phi(\cdot, \rho)$ is defined in $\Phi(x, \rho) = f(x) + \rho \|c(x)\|$, with the penalty parameter $\rho > 0$ and associated to the problem (1.1). Notice that $\Phi(x, \rho)$ is a special case of Φ_h .

Assuming the instantaneous functions have finite gradients that follows the gradients of $f(x)$ are given by

$$g(x) := \nabla f(x) := E_\theta [\nabla F(x, \theta)], \tag{2.3}$$

when the number of functions $F(x, \theta)$ is large, exact evaluation of the gradient $g(x)$ is impractical. This motivates the use of stochastic gradients in lieu of exact gradients. For more precision, consider a given set of N realizations $\tilde{\theta} = (\theta_1, \theta_2, \dots, \theta_n)$ and define the stochastic gradient of $f(x)$ at given samples $\tilde{\theta}$ as

We consider linearizing the argument of Φ_h around x

$$\hat{g}(x, \tilde{\theta}) := \frac{1}{N} \sum_{n=1}^N \nabla f(x, \theta_n). \tag{2.4}$$

to obtain the approximation

At each iterate k , the trial step s_k is computed as the solution of the trust-region subproblem

$$\hat{q}(x, s) \stackrel{def}{=} f(x) + \hat{g}^T(x) s + h(c(x) + J(x) s). \tag{2.5}$$

Adjusting the radius h_k , and building a new iteration based on the value of the ratio r_k of the actual

$$\min_{x \in R^n} \hat{q}(x_k, s) = f(x_k) + \hat{g}(x_k, \tilde{\theta})^T s + h(c(x) + J(x_k) s) \tag{2.6}$$

s.t. $\|s\|_2 \leq h_k$.

function decrease $\Phi_h(x_k) - \Phi_h(x_k + s_k)$ to the optimal model decrease, according to the trust-region rules, similarly,

where $s = x - x_k, h_k \in (0, \bar{h})$ is the trust-region radius.

The model $\hat{q}(x_k, s)$ is minimized approximately to

$$\Psi(x_k, h_k) \stackrel{def}{=} \hat{q}(x_k, 0) - \min_{\|s\|_2 \leq h_k} \hat{q}(x_k, s) = \hat{q}(x_k, 0) - \hat{q}(x_k, s_k), \tag{2.7}$$

produce a step s_k which subject to having the norm less than or equal to h_k . Note that the connection between the (2.5) and (2.7), we could know $\Phi_h(x_k) = \hat{q}(x_k, 0)$. The radio \hat{r}_k of the actual reduction of the objective function $Ared_k = \Phi_h(x_k) - \Phi_h(x_k + s_k)$ to the predicted reduction $Pred_k = \hat{q}(x_k, 0) - \hat{q}(x_k, s_k)$ was calculated.

3. Algorithm (Stochastic Trust Region) Initialization

Variable x_0 . Initial radius $h_0 \in (0, \bar{h})$ where $\bar{h} > 0$,

choose constant, $0 < \eta_1 \leq \eta_2 \leq 1, \forall \epsilon > 0$. Set $k = 0$.

While $\Psi(x_k, h_k) \geq \epsilon$, do.

Step 1.

Acquire N independent samples $\tilde{\theta}_k = [\theta_{k1}; \dots; \theta_{kN}]$, calculate

Step 2.

Solve the trust region subproblem (2.6), giving a trial

$$\hat{g}(x_k, \tilde{\theta}_k) := \frac{1}{N} \sum_{n=1}^N \nabla f(x_k, \theta_{kn}). \tag{2.8}$$

step s_k .

Step 3. Compute

$$\hat{r}_k = \frac{Ared_k}{Pred_k} \tag{2.9}$$

Where $Ared_k = \Phi_h(x_k) - \Phi_h(x_k + s_k)$, $Pred_k = \hat{q}^{(k)}(0) - \hat{q}^{(k)}(s_k)$. If

$\hat{r}_k \geq \eta_1$, set $x_{k+1} = x_k + s_k$; otherwise, set $x_{k+1} = x_k$.

Step 4.

Update the trust region radius.

$$h_{k+1} \in \begin{cases} [h_k, \gamma_3 h_k], & \hat{h}_k \geq \eta_2 \quad k \text{ very successful}; \\ [\gamma_1 h_k, \gamma_3 h_k], & \hat{h}_k \in (\eta_1, \eta_2) \quad k \text{ successful}; \\ [\gamma_1 \min\{|s_k|_2, h_k\}, \gamma_2 h_k], & \hat{h}_k \leq \eta_1 \quad k \text{ unsuccessful}; \end{cases} \tag{2.10}$$

Step 5.

Set $k := k + 1$; go to step 2.

3. Convergence Analysis

In order to prove the convergence we need the following assumptions.

A1. The level set $L = \{x | f(x) + h(c(x)) \leq f(x_0) + h(c(x_0))\}$ was bounded. Moreover, the function $f(x) + h(c(x))$ was bounded below in L .

A2. Define \hat{g} as the stochastic gradient of $f(x)$, and $J(x)$ as the Jacobi of $c(x)$ at x , for any k , we have $E[\hat{g}(x_k, \bar{\theta}_k)] = \nabla f(x_k)$

A3. \hat{g} and J are Lipschitz continuous, with Lipschitz constants $L_g \geq 1$ and L_j respectively.

A4. We assume that h is convex and Lipschitz continuous, with Lipschitz constant L_h . Note that h is convex implies that h is Lipschitz continuous at all required points, provided that the iteration is in a bound set or h is bound above and below on R^n .

Let us relate the minimizers of (2.2) to the solutions of problem (1.1). It is not difficult to find that the advantage of the penalty function is that for a sufficiently large ρ , the local minimizers of satisfying

the MFCQ are minimizers of $\Phi(\cdot, \rho)$. For some $\rho > 0$, x^* is the critical point of $\Phi(\cdot, \rho)$, and it is feasible, then x^* is a KKT point of $\Phi(\cdot, \rho)$.

Definition: for approximate optimality for problem (2.1), which was defined as follows [13]. x^* was called a KKT point of (1.1), if there exists $\lambda^* \in R^q$ such that

$$\nabla f(x^*) + J(x^*)\lambda = 0, \text{ and, } c(x^*) = 0$$

therefore, Cartis, Gould and Toint [13] introduced the following definition of critical point of (2.2). x^* is called a critical point of (2.2), if there exists $\lambda \in R^q$, we have:

$$\begin{aligned} \|\nabla f(x) + J(x)\lambda\| &\leq \varepsilon, \text{ and, } \Phi_\rho(x) \leq \varepsilon \\ \Psi_\rho(x) &\stackrel{def}{=} q^\rho(x_k, 0) - \min_{\|s\| \leq h_k} q^\rho(x_k, s) \\ q(x, s) &\stackrel{def}{=} f(x) + g^T(x)s + \rho\|c(x) + J(x)s\| \end{aligned}$$

In stochastic settings, X. Wang, Y. X. Yuan et al [6] introduced the following definition of critical point of (2.2), any step is a random process, and the output is a random variable, according to the above conditions,

$$E[\|\nabla f(x) + J(x)\lambda\|] \leq \varepsilon, \text{ and, } E[\Phi_\rho(x)] \leq \varepsilon.$$

In this Definition, when the optimality measure $\Phi_\rho(x)$ is small, we describe the similar result at the approximate critical point of $\Phi(\cdot, \rho)$.

Lemma 3.1. Assume that A1, A2 and A4 hold, then we have

$$E[\Psi(x_k, h_k)] \geq \min\{h_k, 1\} E[\Psi_k] \tag{2.11}$$

Proof 3.1. Assume first that $h_k \geq 1$, then

$$\min_{\|s\| \leq 1} \hat{q}(x_k, s) \geq \min_{\|s\| \leq h_k} \hat{q}(x_k, s) \tag{2.12}$$

and so $E[\Psi_k] \leq E[\Psi(x_k, h_k)]$, because

$\min\{h_k, 1\} = 1$, we could deduce (2.11) in the

above case. Let $h_k < 1$, $s_k^* \stackrel{def}{=} \operatorname{argmin}_{\|s\| \leq 1} \hat{q}(x_k, s)$. Then

$\|h_k s_k^*\|_2 \leq h_k$ and so $\hat{q}(x_k, s_k) \leq \hat{q}(x_k, h_k s_k^*)$, implying

$$\begin{aligned} \text{The next lemmas deduce a lower bound on } h_k. \\ E[\Psi(x_k, h_k)] \geq E[\hat{q}(x_k, 0) - \hat{q}(x_k, h_k s_k^*)] \geq h_k E[(\hat{g}(x_k, 0) - \hat{g}(x_k, s_k^*))] = h_k E[\Psi_k] \end{aligned} \tag{2.13}$$

Lemma 3.2. Assume that A1, A2, A3 and A4 hold, then $\Psi_k \neq 0$, where

$$h_k \leq \kappa_L \sqrt{\Psi_k} \min\{1, \sqrt{\Psi_k}\} \tag{2.14}$$

we have that

$$\kappa_L \stackrel{def}{=} \frac{1 - \eta_2}{L_g + \frac{1}{2} L_h L_J} \quad (2.15)$$

k is very successful in the sense of (2.10).

Proof 3.2. From $\Phi_h := f(x) + h(c(x))$, (2.9) and (2.10), we can know

$$\begin{aligned} E[\hat{r}_k - 1] &= E\left[\frac{1}{\Psi(x_k, h_k)} |\Phi_h(x_k + s_k) - \hat{g}(x_k, s_k)|\right] \\ &= E\left[\frac{1}{\Psi(x_k, h_k)} |f(x_k + s_k) - f(x_k) - \hat{g}_k^T s_k + h(c(x_k + h_k)) - h(c(x_k) + J_k s_k)|\right] \\ &\leq E\left[\frac{1}{\Psi(x_k, h_k)} (|f(x_k + s_k) - f(x_k) - \hat{g}_k^T s_k| + |h(c(x_k + h_k)) - h(c(x_k) + J_k s_k)|)\right] \end{aligned} \quad (2.16)$$

From Taylor expansions $f(x_k + s_k) = f(x_k) + \hat{g}_k^T s_k$, there are $\varepsilon_k \in [x_k, x_k + s_k]$ and $c(x_k + s_k) = c(x_k) + \int_0^1 J(x_k + ts_k) s_k dt$ together with A3 and A4, that

$$\begin{aligned} |f(x_k + s_k) - f(x_k) - \hat{g}_k^T s_k| &\leq L_g \|s_k\|^2 \\ |h(c(x_k + h_k)) - h(c(x_k) + J_k s_k)| &\leq \frac{1}{2} L_h L_J \|s_k\|^2 \end{aligned} \quad (2.17)$$

From equation (2.14), it follows that

$$E[|\hat{r}_k - 1|] \leq \frac{(1 - \eta_2) \|s_k\|^2}{\kappa_L E[\Psi(x_k, h_k)]} \leq \frac{1 - \eta_2}{\kappa_L} \frac{h_k^2}{\min\{h_k, 1\} E[\Psi(x_k)]}.$$

From equation (2.15) and $\|s_k\| \leq h_k$ we can deduce the second inequality above. The implication (2.16) now follows from (2.11) and $\kappa \leq 1$, the latter being provided by $L_g \geq 1$ and $0 < \eta_1 < \eta_2 < 1$.

Lemma 3.3. Assume that A1, A2, A3 and A4 hold, let $\epsilon \in (0, 1]$ such that

$$E[\Psi_k] > \epsilon \text{ for all } k = 0, \dots, j \quad (2.18)$$

Where $j \leq \infty$ then

$$h_k \geq \min\{h_0, \gamma_1 \kappa_L \epsilon\} \text{ for all } k = 0, \dots, j, \quad (2.19)$$

where κ_L is defined in (2.14).

proof 3.3. For any $\epsilon \in (0, 1]$, $k \in 0, \dots, j$ and (2.18), we have

$$\kappa_L \epsilon = \kappa_L \sqrt{\epsilon} \min\{1, \sqrt{\epsilon}\} \leq \kappa_L \sqrt{E[\Psi_k]} \min\{1, \sqrt{E[\Psi_k]}\}$$

from (2.10) and Lemma 3.1, we can deduce

$$h_k \leq \kappa_L \epsilon \Rightarrow h_k \leq h_{k+1}. \quad (2.20)$$

Thus when $h_k \geq \gamma_1 \kappa_L \epsilon$, (2.20) implies that $h_k \geq \gamma_1 \kappa_L \epsilon$ for all $k \in 0, \dots, j$, where the factor γ_1 is introduced for the case when h_k is greater than $\kappa_L \epsilon$ and iteration k is not very successful. Letting $k=0$ in (2.20) gives (2.19) when $h_0 < \gamma_1 \kappa_L \epsilon$ since $\gamma_1 \in (0, 1)$.

We are now ready to give the main result of this section.

Lemma 3.4. Assume that A1, A2, A3 and A4 hold, and let $\{\Phi_h(x_k)\}$ be bounded below by Φ_h^{low} . Given any $\epsilon \in (0, 1]$, assume that $\Psi_0 > \epsilon$, and let $j_1 \leq \infty$ be the first iteration such that $\Psi_{j_1+1} \leq \epsilon$. Then the stochastic trust-region algorithm, Algorithm, takes at most

$$J_1^s \stackrel{def}{=} \lceil \kappa_{TR}^s \epsilon^{-2} \rceil \quad (2.21)$$

successful iteration, or equivalently, problem stochastic gradient evaluations, to generate $\Psi_{j_1+1} \leq \epsilon$, where

$$\kappa_{TR}^s \stackrel{def}{=} \frac{\Phi_h(x_0) - \Phi_h^{low}}{\eta_1 \min\{h_0, \gamma_1 \kappa_L\}} \quad (2.22)$$

where κ_L is defined in (2.14). Additionally, assume that on each very successful iteration k , h_k is chosen

$$h_{k+1} \leq \gamma_3 h_k \quad (2.23)$$

such that

for some $\gamma_3 > 1$. Then

$$j_1 \leq \lceil \kappa_{TR}^s \epsilon^{-2} \rceil = J_1, \quad (2.24)$$

and so Algorithm takes at most J_1 (successful and unsuccessful) iterations, or, equivalently, problem evaluations, to generate $E[\Psi_{j_1+1}] \leq \epsilon$, where

$$\kappa_{TR} \stackrel{def}{=} \kappa_{TR}^s \left(1 - \frac{\log \gamma_3}{\log \gamma_2}\right) + \frac{1}{|\log \gamma_2|} \cdot \frac{h_0}{\gamma_1 \kappa_L}. \quad (2.25)$$

Proof 3.4. According to the definition of j_1 in the theorem, it is equivalent to

$$E[\Psi_k] > \epsilon \text{ for all } k = 0, \dots, j_1 \text{ and } \Psi_{j_1+1} \leq \epsilon.$$

Thus Lemma 3.3. applies with $j = j_1$. From $E[\Psi(x_k, h_k)] \geq \min\{h_k, 1\} E[\Psi_k]$ and $h_k \geq \min\{h_0, \gamma_1 \kappa_L \epsilon\}$, $k = 0, \dots, j$ we can deduce

$$E[\Psi\{x_k, h_x\}] \geq \min\{1, h_0, \gamma_1 \kappa_L \epsilon\} E[\Psi_k], k = 0, \dots, j_1,$$

due to $\kappa_L \in (0, 1], \gamma_1 \in (0, 1]$, and again

$$E[\Psi_k] > \epsilon, k = 0, \dots, j_1 \text{ and } E[\Psi_{j_1+1}] \leq \epsilon,$$

$$E[\Psi\{x_k, h_x\}] \geq \min\{h_0, \gamma_1 \kappa_L\} \epsilon^2, k = 0, \dots, j_1 \quad (2.26)$$

can be derived from the above formula. Define S as the set of all successful and very successful iterations in (2.10), let $k \in S \cap \{0, \dots, j_1\}$, due to

$$E[\hat{r}_k] = \frac{\Phi_h(x_k) - \Phi_h(x_k + s_k)}{E[\Psi(x_k, h_k)]} = \frac{\Phi_h(x_k) - \Phi_h(x_k + s_k)}{E[\hat{q}(x_k, 0) - \hat{q}(x_k, s_k)]} \quad (2.27)$$

by Trust-region radius update set (2.10), and (2.16) imply

$$\begin{aligned} \Phi_h(x_k) - \Phi_h(x_k + s_k) &= E[\hat{r}_k \Psi(x_k, h_k)] \geq \eta_1 E[\Psi(x_k, h_k)] \\ &\geq \eta_1 \min\{h_0, \gamma_1 \kappa_L\} \epsilon^2, k = 0, \dots, j_1. \end{aligned} \quad (2.28)$$

Define k_s as the number of successful iterations up to j_1 , because $\Phi_h(x_{j_1}) \geq \Phi_h^{low}$, and keep the function value unchanged during unsuccessful iterations, we get

$$\Phi_h(x_0) - \Phi_h^{low} \geq k_s \eta_1 \min\{h_0, \gamma_1 \kappa_L\} \epsilon^2$$

Given J_1^s as the upper bound of iteration, in order to prove the bound J_1 , suppose the upper bound of the number of unsuccessful iterations is j_1 . From (2.10) and (2.23) we can get

$$\begin{aligned} h_{k+1} &\leq \gamma_3 h_k, k \in \{0, \dots, j_1\} \cap S \\ h_{i+1} &\leq \gamma_2 h_i, i \in \{0, \dots, j_1\} \setminus S \end{aligned}$$

Define k_u as the number of unsuccessful iterations up to j_1 , we can conclude that

$$h_{j_1} \leq h_0 \gamma_2^{k_u} \gamma_3^{k_s},$$

according to (2.19), this further becomes

$$\gamma_2^{k_u} \gamma_3^{k_s} \geq \frac{h_{j_1}}{h_0} = \min\{1, \frac{\gamma_1 \kappa_L \epsilon}{h_0}\},$$

given $\gamma_2 \in (0, 1)$, taking the logarithm of both sides of the above inequality, we get

$$k_u \leq \frac{1}{\log \gamma_2} \log \frac{\gamma_1 \kappa_L \epsilon}{h_0} - k_s \frac{\log \gamma_3}{\log \gamma_2}$$

therefore, due to $\log x \leq x$, this further becomes

$$j_1 = k_s + k_u \leq k_s \left(1 - \frac{\log \gamma_3}{\log \gamma_2}\right) + \frac{h_0}{\gamma_1 \kappa_L |\log \gamma_2|} \frac{1}{\epsilon} \quad (2.29)$$

which together with the bound J_1^s on k_s and $\epsilon \in (0, 1]$ yields (2.25).

4. Numerical Experiments

In this section, the proposed stochastic trust region method to solve nonconvex problems with stochastic objectives is applied.

Example 1: Extended Rosenbrock Function

In mathematical optimization, the Rosenbrock function is frequently used in nonconvex optimization problems as a performance test problem for optimization algorithms, which introduced by Howard H. Rosenbrock in 1960.

Here we use a stochastic version of the Extended Rosenbrock function as a test case [14]. Specifically consider a random vector $\theta \in R^{n/2}$, a variable $x \in R^n$, the function $f(x)$ is defined as

$$\begin{aligned} f(x) &:= E_\theta[F(x, \theta)] \\ &:= E_\theta\left[\sum_{i=1}^{n/2} \{(1 + \theta_i)((100(x_{2i-1}^2 - x_{2i})^2 + (1 - x_{2i-1})^2)\}\right] \end{aligned} \quad (3.1)$$

we chose θ uniformly at random from the box $\Theta = [-\theta_0, \theta_0]^n$

and set $\theta_0 = 0.5$. In fact from the observation

$E_\theta[\theta] = 0$, we can write the average function as

$$\begin{aligned} f(x) &= \sum_{i=1}^{n/2} ((100(x_{2i-1}^2 - x_{2i})^2 + (1 - x_{2i-1})^2) \\ c(x) &= \sum_{i=1}^{n/2} x_{2i-1} + x_{2i} - 2i \end{aligned} \quad (3.2)$$

The function in (3.2) has a minimizer $x^* = [1, 1, \dots, 1]^T$ with $f(x^*) = 0$ and $c(x^*) = 0$

T h e

numerical results of the above examples are shown in the table below

Table 1 Output for test problem

Example simulation	Initial point	Epsilon	<u>Itre</u>	Val ^e
$n = 2$	$x = (2, 1)^T$	$1e - 6$	15	$3.8942e$
$n = 4$	$x = (2, \dots, 2)^T$	$1e - 6$	13	$4.0324e$
$n = 10$	$x = (2, \dots, 2)^T$	$1e - 6$	398	$1.3410e$
$n = 20$	$x = (2, \dots, 2)^T$	$1e - 6$	277	$1.1253e$

Example 2: Booth Function

In mathematical optimization, the Booth function is frequently used in optimization problems as a performance test problem for optimization algorithms.

Here we consider a random vector $\theta \in R^n$, a variable $x \in R^n$, the function $f(x)$ is defined as

$$f(x) := E_{\theta}[F(x, \theta)]$$

$$:= E_{\theta}\left[0.5 \sum_{i=1}^n (1 + \theta_i) (x_i^4 - 16x_i^2 + 5x_i)\right] \quad (3.3)$$

we chose θ uniformly at random from the box $\Theta = [-\theta_0, \theta_0]^n$ and set $\theta_0 = 0.5$. In fact from the observation $E_{\theta}[\theta] = 0$, we can write the average function as

$$f(x) = 0.5 \sum_{i=1}^n (x_i^4 - 16x_i^2 + 5x_i)$$

$$c(x) = \sum_{i=1}^n ((-1/2.903534x_i) - n) \quad (3.4)$$

The function in (3.4) has a minimizer $x^* = [-2.903534, \dots, -2.903534]^T$ with $f(x^*) = -39.16599n$ and $c(x^*) = 0$.

The numerical results of the above examples are shown in the table below

Table 2 Output for test problem

Example simulation	Initial point	Epsilon	<u>Itre</u>	Val ^e
$n = 2$	$x = (-2, -2)^T$	$1e - 6$	3	-78.3260
$n = 4$	$x = (-2, \dots, -2)^T$	$1e - 6$	32	-156.6614
$n = 10$	$x = (-2, \dots, -2)^T$	$1e - 6$	99	-391.6610
$n = 30$	$x = (-2, \dots, -2)^T$	$1e - 6$	216	-1174.9680
$n = 50$	$x = (-2, \dots, -2)^T$	$1e - 6$	212	-1958.2983

Example 3: Extended Three-Hump Camel Function

In mathematical optimization, the Three-Hump Camel function is frequently used in optimization problems as a performance test problem for optimization

algorithms, Here we consider a random vector $\theta \in R^{n/2}$, a variable $x \in R^n$, the function $f(x)$ is defined as

$$f(x) := E_{\theta}[F(x, \theta)]$$

$$:= E_{\theta}\left[\sum_{i=1}^{n/2} (1 + \theta_i) (2x_{2i}^2 - 1.05x_{2i-1}^2 + x_{2i-1}^6/6 + x_{2i-1}x_{2i} + x_{2i}^2)\right] \quad (3.5)$$

we chose θ uniformly at random from the box $\Theta = [-\theta_0, \theta_0]^{n/2}$ and set $\theta_0 = 0.5$. In fact from the observation $E_{\theta}[\theta] = 0$, we can write the average function as

$$f(x) = \sum_{i=1}^{n/2} (2x_{2i}^2 - 1.05x_{2i-1}^2 + x_{2i-1}^6/6 + x_{2i-1}x_{2i} + x_{2i}^2)$$

$$c(x) = \sum_{i=1}^{n/2} x_{2i-1} + x_{2i} \quad (3.6)$$

The function in (3.6) has a minimizer $x^* = [0, 0, \dots, 0]^T$ with $f(x^*) = 0$ and $c(x^*) = 0$. The numerical results of the above examples are shown in the table below

Table 3 Output for test problem

Example simulation	Initial point	Epsilon	<u>Itre</u>	Val ^e
$n = 2$	$x = (-2, -2)^T$	$1e - 6$	10	$5.879e - 10$
$n = 10$	$x = (-2, \dots, -2)^T$	$1e - 6$	15	$4.0027e - 10$
$n = 20$	$x = (-2, \dots, -2)^T$	$1e - 6$	23	$7.3555e - 13$
$n = 30$	$x = (-2, \dots, -2)^T$	$1e - 6$	17	$1.1663e - 09$
$n = 50$	$x = (-2, \dots, -2)^T$	$1e - 6$	237	$6.3476e - 08$

5. Conclusions

In this paper we proposed a stochastic trust region method and showed that the new algorithm was convergent for solving constrained minimization problems with stochastic objectives. Based on the penalty method and the trust region framework, our method could deal with convex optimization problems with ill-conditioned objective functions as well as nonconvex optimization problems. With careful analysis, we were able to show that our method was convergent. Numerical results illustrated that the method could efficiently solve the given test problems. Therefore the new method is potentially efficient, and thus paves the way towards developing concrete algorithms for specific applications.

References

- [1] Apt K. Cambridge University Press, 2003 9-10.
- [2] F. Bastin, C. Cirillo, and P. L. Toint. C. Mathematical Programming, 108(2006) 207-234.
- [3] P. Hententryck. ACM Computing Surveys, 28(1996) 701-726.
- [4] D. Brownstone, D. S. Bunch, and K. Train. Transportation Research B, 34(2000) 315-338.
- [5] S. Ghadimi, G. H. Lan, and H. C. Zhang. Mathe-

- mathematical Programming, 155(2016) 267-305.
- [6] X. Wang, S. Q. Ma, Y. X. Yuan. *Mathematics of Computation*, 86(2016) 1793-1820.
 - [7] L. H. Lei, C. Ju, J. B. Chen, and M. I. Jordan. In *Advances in Neural Information Processing Systems*, (2017) 2345-2355.
 - [8] L. Xiao, T. Zhang. *SIAM Journal on Optimization*, 24(2014) 2057-2075.
 - [9] S. J. Reddi, A. Hefny, S. Sra, B. Póczos, and A. Smola, *Computer Science Bibliography*, a(2016) 314-323.
 - [10] M. Fu. *INFORMS Journal on Computing*, 14(2002) 192-215.
 - [11] Z.Y. A. Zhu. *Theory of Computing*, a(2017) 1200-1205.
 - [12] A. Defazio, F. Bach and S. L. Julien. In *Advance in Neural Information Processing Systems*, (2014) 1646-1645.
 - [13] C. Cartis, N. I. M. Gould and P. L. Toint. *SIAM Journal on Optimization*, 21(2011) 1721-1739.
 - [14] J. Nocedal, S. Wright. *Numerical Optimization*, (1999).
 - [15] A. R. Conn, N. I. M. Gould, and P. L. Toint. *SIAM Journal on Optimization*, 45(2003) 128-131.
 - [16] F. E. Curits, D. P. Robinson, and M. Samadi. *Mathematical Programming*, 162(2017) 1-32.
 - [17] D. A. Hensher, W. H. Greene. *Transportation*, 30(2003) 133-176.
 - [18] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. *International Conference On Machine Learning*, (2009) 689-696.