symmetry

MDPI

Article

Solving the Adaptive Cubic Regularization Sub-Problem Using the Lanczos Method

Zhi Zhu and Jingya Chang *

School of Mathematics and Statistics, Guangdong University of Technology, Guangzhou 510520, China; zhuzhixixi@163.com

* Correspondence: jychang@gdut.edu.cn

Abstract: The adaptive cubic regularization method solves an unconstrained optimization model by using a three-order regularization term to approximate the objective function at each iteration. Similar to the trust-region method, the calculation of the sub-problem highly affects the computing efficiency. The Lanczos method is an useful tool for simplifying the objective function in the sub-problem. In this paper, we implement the adaptive cubic regularization method with the aid of the Lanczos method, and analyze the error of Lanczos approximation. We show that both the error between the Lanczos objective function and the original cubic term, and the error between the solution of the Lanczos approximation and the solution of the original cubic sub-problem are bounded up by the condition number of the optimal Hessian matrix. Furthermore, we compare the numerical performances of the adaptive cubic regularization algorithm when using the Lanczos approximation method and the adaptive cubic regularization algorithm without using the Lanczos approximation for unconstrained optimization problems. Numerical experiments show that the Lanczos method improves the computation efficiency of the adaptive cubic method remarkably.

Keywords: adaptive cubic regularization sub-problem; Lanczos method; large-scale optimization; Krylov subspaces

Citation: Zhu, Z.; Chang, J. Solving the Adaptive Cubic Regularization Sub-Problem Using the Lanczos Method. *Symmetry* **2022**, *14*, 2191. https://doi.org/10.3390/sym14102191

Academic Editors: Wen Li, Yannan Chen and Zhigang Jia

Received: 14 September 2022
Accepted: 9 October 2022
Published: 18 October 2022

1. Introduction

For the unconstrained optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}),$$

Cartis et al. [1] proposed an adaptive cubic regularization (ACR) algorithm. It is an alternative to classical globalization techniques, which uses a cubic over-estimator of the objective function as a regularization technique, and uses an adaptive parameter σ to replace the Lipschitz constant in the cubic Taylor-series model. At each iteration, the objective function is approximated by a cubic function. Numerical experiments in [1] show that the ACR is comparable with trust-region method for small-scale problems. Despite the fact that the method has been shown to have powerful local and global convergence properties, the practicality and efficiency of the adaptive cubic regularization method depend critically on the efficiency of solving its sub-problem at each iteration.

For solving the trust-region sub-problem, many efficient algorithms have been proposed. These algorithms can be grouped into three broad categories: the accurate methods for dense problems, the accurate methods for large-sparse problems, and the approximation methods for large-scale problems. The first category are the accurate methods for dense problems, such as the classical algorithm proposed by Moré and Sorensen [2], which used Newton's method to iteratively solve symmetric positive definite linear systems via the Cholesky factorization. The second category are the accurate methods for large-sparse problems. For instance, the Lanczos method was employed to solve the large-scale trust-region sub-problem through a parameterized eigenvalue problem [3,4]. Another accurate

approach [5], is based on a parametric eigenvalue problem within a semi-definite framework, which employed the Lanczos method for the smallest eigenvalue as a black box. Hager [6] and Erway et al. [7] utilized the subspace projection algorithms for accurate methods. The third category are the approximation methods for large-scale problems. The generalized Lanczos trust-region method (GLTR) [8,9] was proposed as an improved Steihaug [10]-Toint [11] conjugate-gradient method. For the GLTR method, Zhang et al. established prior upper bounds [12] and posterior error bounds [13] for the optimal objective value and the optimal solution between the original trust-region sub-problem and their projected counterparts.

For solving cubic models sub-problems, many algorithms are extensions of trust-region algorithms. Cartis et al. [1] provided the Newton's method to solve the sub-problem of ACR, which employs Cholesky factorization at each iteration. This method usually applies to small-scale problems. Moreover, Cartis et al. briefly described the process of using the Lanczos method for the ACR sub-problem in [1]. Carmon and Duchi [14] provided the gradient descent method to approximate the cubic-regularized Newton step, and gave the convergence rate. However, the convergence rate of the gradient descent method is worse than that of the Krylov subspace method. Birgin et al. [15] proposed a Newton-like method for unconstrained optimization, whose sub-problem is similar to but different from that of ACR. They introduced a mixed factorization, which is a cheaper factorization than the Cholesky factorization. Brás et al. [16] used the Lanczos method efficiently to solve the sub-problems associated with a special type of cubic models, and also embedded the Lanczos method in a large-scale trust-region strategy. Furthermore, an accelerated first-order method for the ACR sub-problem was developed by Jiang et al. [17].

In this paper, we employ the Lanczos method to solve the sub-problem of the adaptive cubic regularization method (ACRL) for large-scale problems. The ACRL algorithm mainly includes the following three steps. Firstly, the ACRL generates the jth Krylov subspace using the Lanczos method. Next, we project the original sub-problem onto the jth Krylov subspace to obtain a smaller-sized sub-problem. Finally, we solve the resulting smaller-sized sub-problem to get an approximate solution. Such procedures are based on the minimization of the local model of the objective function over a sequence of small-sized sub-spaces. As a result, the ACRL is applicable for large-scale problems. Moreover, we analyze the error of the Lanczos approximation. For unconstrained optimization problems, we perform numerical experiments and compare our method with the method of not using the Lanczos approximation (ACRN).

The outline of this paper is as follows. In Section 2, we introduce the adaptive cubic regularization method and its optimality condition. The method using the Lanczos algorithm to solve the ACR sub-problem is introduced in Section 3. In Section 4, we show the error bounds of the approximate solution and approximate objective value obtained using the ACRL method. Numerical experiments demonstrating the efficiency of the algorithm are given in Section 5. Finally, we give some concluding remarks in Section 6.

2. Preliminaries

Throughout the paper, a matrix is represented by a capital letter, while a lower case bold letter is used for a vector and a lower case letter for a scalar.

The adaptive cubic regularization method [1,18] is proposed by Cartis et al. for unconstrained optimization problems. It mainly uses a cubic over-estimator of the objective function as a regularization technique to calculate the step at each iteration. Assuming that $\mathbf{x}_k$ is the current iteration point, the objective function $f(\mathbf{x})$ is second-order continuously differentiable, and its Hessian matrix $H(\mathbf{x}) = \nabla_{\mathbf{xx}} f(\mathbf{x})$ is globally Lipschitz continuous. For any $\mathbf{p} \in \mathbb{R}^n$, by expressing the Taylor expansion of $f(\mathbf{x}_k + \mathbf{p})$ at the point $\mathbf{x}_k$, we obtain

$$\begin{aligned} f(\mathbf{x}_k + \mathbf{p}) &= f(\mathbf{x}_k) + \mathbf{p}^T \mathbf{g}(\mathbf{x}_k) + \frac{1}{2}\mathbf{p}^T H(\mathbf{x}_k)\mathbf{p} + \int_0^1 (1-t)\mathbf{p}^T [H(\mathbf{x}_k + t\mathbf{p}) - H(\mathbf{x}_k)]\mathbf{p} \mathrm{d}t \\ &\leq f(\mathbf{x}_k) + \mathbf{p}^T \mathbf{g}(\mathbf{x}_k) + \frac{1}{2}\mathbf{p}^T H_k \mathbf{p} + \frac{1}{6} L \|\mathbf{p}\|^3, \end{aligned} \tag{1}$$

where $\mathbf{g}(\mathbf{x}) = \nabla_{\mathbf{x}} f(\mathbf{x})$, $H(\mathbf{x}) = \nabla_{\mathbf{xx}} f(\mathbf{x})$, and L is the Lipschitz constant. Here, and for the remainder of this paper, $\|\cdot\|$ denotes an l_2 norm. The inequality is obtained by using the Lipschitz property of $\nabla_{\mathbf{xx}} f(\mathbf{x})$. In [1], Cartis et al. proposed to replace the constant $\frac{1}{2}L$ in Equation (1) with a dynamic positive parameter σ_k. In the cubic regularization model, the matrix $H(\mathbf{x})$ needs not to be globally or locally continuous in general. Furthermore, the approximation of $H(\mathbf{x})$ by a symmetric matrix B_k is employed at each iteration. Therefore, the model

$$\min \ m_k(\mathbf{p}) := f(\mathbf{x}_k) + \mathbf{p}^T \mathbf{g}(\mathbf{x}_k) + \frac{1}{2}\mathbf{p}^T B_k \mathbf{p} + \frac{1}{3}\sigma_k \|\mathbf{p}\|^3 \tag{2}$$

is used to estimate $f(\mathbf{x}_k)$ at each iteration. Then, the adaptive cubic regularization method sub-problem aims to compute a descent direction vector $\mathbf{p}$. Finally, the sub-problem is given with the form of

$$\min \ q_k(\mathbf{p}) := \mathbf{p}^T \mathbf{g_k} + \frac{1}{2}\mathbf{p}^T B_k \mathbf{p} + \frac{1}{3}\sigma_k \|\mathbf{p}\|^3, \tag{3}$$

in which $\mathbf{g_k}$ is short for $\mathbf{g}(\mathbf{x}_k)$.

Cartis et al. introduced the following global optimality result of ACR, which is similar to the optimality conditions of the trust-region method.

Theorem 1 ([1], Theorem 3.1)**.** *The vector $\mathbf{p}_{opt}$ is a global minimizer of the sub-problem (3) if and only if there is a scalar $\lambda_{opt} \geq 0$ satisfying the following system of equations:*

$$(B_k + \lambda_{opt} I)\mathbf{p}_{opt} = -\mathbf{g}_k, \tag{4}$$

where $\lambda_{opt} = \sigma_k \|\mathbf{p}_{opt}\|$, and $B_k + \lambda_{opt} I$ is a positive semi-definite matrix. If $B_k + \lambda_{opt} I$ is positive definite, then $\mathbf{p}_{opt}$ is unique.

The optimality condition of the trust-region sub-problem [19] aims to minimize $\mathbf{g}_k^T \mathbf{p} + \frac{1}{2}\mathbf{p}^T B_k \mathbf{p}$ within an l_2-norm trust region $\|\mathbf{p}\| \leq \triangle_k$, where $\triangle_k > 0$ is the trust-region radius. For a trust-region sub-problem, the vector $\mathbf{p}_{opt}$ satisfies $\lambda_{opt}(\triangle_k - \|\mathbf{p}_{opt}\|) = 0$, which means either $\lambda_{opt} = 0$ or $\|\mathbf{p}_{opt}\| = \triangle_k$. When both the trust-region sub-problem and the cubic regularization sub-problem approximate the original objective function precisely enough, we get $\triangle_k = \lambda_{opt}/\sigma_k$ from Theorem 1. Therefore, the parameter σ_k in the ACR algorithm is inversely proportional to the trust-region radius, and it plays the same role as the trust region-radius, while we adjust the estimation accuracy of the sub-problem.

3. Computation of the ACR Sub-Problem with the Lanczos Method

The Lanczos algorithm [20] was proposed to solve sparse linear systems and to find the eigenvalues of sparse matrices. It builds up an orthogonal basis $Q_j = \{\mathbf{q_0}, \mathbf{q_1}, \ldots, \mathbf{q_j}\}$ for the Krylov space $\mathcal{K}_j(B, \mathbf{g}) := \{\mathbf{g}, B\mathbf{g}, B^2\mathbf{g}, \ldots, B^j\mathbf{g}\}$. By utilizing the orthogonal basis Q_j, the original symmetric matrix B is transformed into a tridiagonal matrix.

Normally, the dimension of the $\mathcal{K}_j(B, \mathbf{g})$ increases by 1 as j increases by 1. However, the Lanczos process may break down and the dimension of $\mathcal{K}_j(B, \mathbf{g})$ stops increasing at a certain j. We define j_{max} as the smallest nonnegative integer, such that the Lanczos process breaks down. If the dimension of the Krylov space is much less than the size of the matrix, it greatly saves the storage space and highly improves the calculation speed by projecting B onto a $j+1$ subspace. Specially, we find a proper Q_j using the Lanczos method, such that $Q_j^T B Q_j = T_j$ is tridiagonal. We state the procedure in the following algorithm.

Algorithm 1 computes an orthogonal matrix $Q_j = [\mathbf{q}_0, \mathbf{q}_1, \ldots, \mathbf{q}_j] \in \mathbb{R}^{n\times(j+1)}$, where

$$T_j = \begin{pmatrix} \alpha_0 & \beta_1 & & & \\ \beta_1 & \alpha_1 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \beta_{j-1} \\ & & & \beta_{j-1} & \alpha_j \end{pmatrix}$$

is tridiagonal. Moreover, it follows directly from Algorithm 1 that

$$Q_j^T Q_j = I, T_j = Q_j^T B Q_j, Q_j^T \mathbf{g} = \beta_0 \mathbf{e}_1, \tag{5}$$

where $\mathbf{e}_1$ is the first unit vector of $j+1$ in length.

Algorithm 1 Lanczos algorithm

1: $j = 0, \beta_0 = \|\mathbf{g}\|, \mathbf{r}_0 = \mathbf{g}, \mathbf{q}_0 = \mathbf{r}_0/\|\mathbf{r}_0\|$
2: **while** $j = 0$ *or* $\beta_j \neq 0$ **do**
3: $\quad \mathbf{q}_{j+1} = r_j/\beta_j$
4: $\quad j = j + 1$
5: $\quad \alpha_j = \mathbf{q}_j^T B \mathbf{q}_j$
6: $\quad \mathbf{r}_j = (B - \alpha_j I)\mathbf{q}_j - \beta_{j-1}\mathbf{q}_{j-1}$
7: $\quad \beta_j = \|\mathbf{r}_j\|$
8: **end while**

For a large-scale trust-region sub-problem, an effective solution is to approximately calculate it using the Krylov subspace methods. The Lanczos algorithm, as one of the Kryolv subspace methods, was first introduced in [8] for the trust-region method. Similar to the trust-region method, the Lanczos algorithm is also suitable for solving the cubic regularization sub-problem. By employing Algorithm 1, we find

$$\mathbf{p}_j^k = Q_j^k \mathbf{u}_j^k := \underset{\mathbf{p}\in\mathcal{K}_j(B_k,\mathbf{g}_k)}{\arg\min}\ q_k(\mathbf{p}) \in \mathcal{K}_j(B_k, \mathbf{g}_k), \tag{6}$$

where $q_k(\mathbf{p})$ is defined by (3). The original sub-problem (3) is transformed into the following sub-problem

$$\min\ q_k(\mathbf{u}) := \beta_0 \mathbf{u}^T e_1 + \frac{1}{2}\mathbf{u}^T T_j \mathbf{u} + \frac{1}{3}\sigma_k \|\mathbf{u}\|_2^3. \tag{7}$$

Theorem 1 illustrates that $\mathbf{u}$ is a global minimizer of the above sub-problem, if and only if a pair of $(\mathbf{u}, \lambda)$ satisfies

$$(T_j + \lambda I)\mathbf{u} = -\beta_0 \mathbf{e}_1 \text{ and } \lambda^2 = \sigma_k^2 \mathbf{u}^T \mathbf{u}, \tag{8}$$

where $T_j + \lambda I$ is positive semi-definite. Equation (8) can finally be solved by Newton's method ([1], Algorithm 6.1). Newton's method for solving the sub-problem requires the eigenvalue decomposition of $B + \lambda I$ for various λ. When the scale of the original problem is large, it is very expensive to directly use the iterative method.

In summary, an approximation $\mathbf{p}_j^k$ of the solution of the ACR sub-problem (3) can be obtained in the following steps. First, we apply j steps of the Lanczos method to the cubic function appearing in (3) to obtain a tridiagonal matrix T_j. Then, we use the Newton's method for a small-size sub-problem with matrix T_j to compute the Lagrange multiplier λ_k and $\mathbf{u}_j^k$. Finally, the matrix Q_j is used to recover $\mathbf{p}_j^k$. Thus, it should be noted that the Lanczos vectors need to be saved. We sketch the algorithm as follows.

In the GLTR algorithm, ([8], Theorem 5.8) discussed a restarting strategy for the degenerate case, which means that multiple global solutions $\mathbf{p}_{opt}$ exist. Similar to the

GLTR, a restarting strategy also applies to the ACRL, although this is just discussed from a theoretical perspective. Therefore, we mainly consider the nondegenerate case in the following analysis.

4. Convergence Analysis

Theorem 1 shows that we aim to seek a pair of $(\mathbf{p}_{opt}, \lambda_{opt})$ satisfying

$$(B_k + \lambda_{opt} I)\mathbf{p}_{opt} = -\mathbf{g}_k \text{ and } \lambda_{opt} = \sigma_k \|\mathbf{p}_{opt}\|. \tag{9}$$

Then, we have $\|\mathbf{p}_{opt}\| = \lambda_{opt}/\sigma_k$. In this section, we will analyze the error between the optimal objective function value of the original sub-problem $q_k(\mathbf{p}_{opt})$ and the the optimal objective function value $q_k(\mathbf{p}_j^k)$ of the sub-problem in the subspace $\mathcal{K}_j(B_k, \mathbf{g}_k)$ generated by the Algorithm 2, as well as the distance between $\mathbf{p}_{opt}$ and $\mathbf{p}_j^k$ under the assumption $\sigma_k > 0$ when the Equation (9) was satisfied.

Algorithm 2 The ACRL method

1: **for** $j = 0 \ldots$ **do**
2: Obtain T_j and Q_j from Algorithm 1
3: Solve the tridiagonal sub-problem (7) to get $\mathbf{u}_j^k$ by Newton's method
4: $\mathbf{p}_j^k = Q_j \mathbf{u}_j^k$
5: $j = j + 1$
6: **end for**

We set

$$B_k^{opt} := B_k + \lambda_{opt} I, \tag{10}$$

which is positive definite in the nondegenerate case. The spectral condition number of B_k^{opt} is

$$\kappa(B_k^{opt}) = \frac{\theta_n + \lambda_{opt}}{\theta_1 + \lambda_{opt}}, \tag{11}$$

where $\theta_1 \leq \theta_2 \leq ... \leq \theta_n$ are the eigenvalues of B_k. We define

$$q_k^{opt}(\mathbf{p}) := \frac{1}{2}\mathbf{p}^T B_k^{opt}\mathbf{p} + \mathbf{p}^T\mathbf{g}_k + \frac{1}{3}\sigma_k\|\mathbf{p}\|^3 = q_k(\mathbf{p}) + \frac{1}{2}\lambda_{opt}\|\mathbf{p}\|^2, \tag{12}$$

in which $q_k(\mathbf{p})$ is defined by (3).

Next, for the vector $\mathbf{p}_j^k$ defined in (6), we analyze the errors

$$\|\mathbf{p}_j^k - \mathbf{p}_{opt}\| \text{ and } |q_k(\mathbf{p}_j^k) - q_k(\mathbf{p}_{opt})|.$$

Theorem 2. *Suppose* (3) *is nondegenerate;* $\|\mathbf{p}_{opt}\| = \lambda_{opt}/\sigma_k$ *and* $\mathbf{p}_j^k$ *is the jth approximation of* $\mathbf{p}_{opt}$ *generated by ACRL satisfying* $\|\mathbf{p}_j^k\| = \lambda_{opt}/\sigma_k$*, then for any nonzero* $\widetilde{\mathbf{p}} \in \mathcal{K}_j(B_k, \mathbf{g}_k)$*, we have*

$$0 \leq q_k(\mathbf{p}_j^k) - q_k(\mathbf{p}_{opt}) \leq 2\|B_k^{opt}\|\|\widetilde{\mathbf{p}} - \mathbf{p}_{opt}\|^2 \tag{13}$$

and

$$\|\mathbf{p}_j^k - \mathbf{p}_{opt}\| \leq 2\kappa(B_k^{opt})\|\widetilde{\mathbf{p}} - \mathbf{p}_{opt}\|. \tag{14}$$

Proof. It can be seen that $|\|\widetilde{\mathbf{p}}\| - \frac{\lambda_{opt}}{\sigma_k}| = |\|\widetilde{\mathbf{p}}\| - \|\mathbf{p}_{opt}\|| \leq \|\widetilde{\mathbf{p}} - \mathbf{p}_{opt}\|$. Then, we obtain

$$\left|1 - \frac{\lambda_{opt}}{\sigma_k\|\widetilde{\mathbf{p}}\|}\right| \leq \frac{\|\widetilde{\mathbf{p}} - \mathbf{p}_{opt}\|}{\|\widetilde{\mathbf{p}}\|}. \tag{15}$$

Let $\mathbf{s} = \mathbf{v} - \mathbf{p}_{opt}$, where

$$\mathbf{v} = \frac{\lambda_{opt}}{\sigma_k} \cdot \frac{\widetilde{\mathbf{p}}}{\|\widetilde{\mathbf{p}}\|}. \tag{16}$$

Based on (16), we obtain

$$\|\mathbf{v}\| = \|\mathbf{p}_{opt}\| = \frac{\lambda_{opt}}{\sigma_k}. \tag{17}$$

We immediately have

$$\begin{aligned}\|\mathbf{s}\| = \|\mathbf{p}_{opt} - \mathbf{v}\| &\le \|\mathbf{p}_{opt} - \widetilde{\mathbf{p}}\| + \|\widetilde{\mathbf{p}} - \mathbf{v}\| \\ &= \|\mathbf{p}_{opt} - \widetilde{\mathbf{p}}\| + \left\|\widetilde{\mathbf{p}} - \frac{\lambda_{opt}}{\sigma_k} \cdot \frac{\widetilde{\mathbf{p}}}{\|\widetilde{\mathbf{p}}\|}\right\| \\ &\le \|\mathbf{p}_{opt} - \widetilde{\mathbf{p}}\| + \|\widetilde{\mathbf{p}}\|\left|1 - \frac{\lambda_{opt}}{\sigma_k\|\widetilde{\mathbf{p}}\|}\right| \\ &\le 2\|\mathbf{p}_{opt} - \widetilde{\mathbf{p}}\|, \end{aligned} \tag{18}$$

where the last equality follows from (15).

Furthermore, for any $0 \le i \le j_{max} - 1$,

$$q_k(\mathbf{p}_i^k) = \min_{\mathbf{p}\in\mathcal{K}_i(B_k,\mathbf{g}_k)} q_k(\mathbf{p}) \ge q_k(\mathbf{p}_{i+1}^k) = \min_{\mathbf{p}\in\mathcal{K}_{i+1}(B_k,\mathbf{g}_k)} q_k(\mathbf{p}) \ge q_k(\mathbf{p}_{opt}) = \min q_k(\mathbf{p}).$$

Therefore, we have

$$\begin{aligned} 0 \le q_k(\mathbf{p}_j^k) - q_k(\mathbf{p}_{opt}) &\le q_k(\mathbf{v}) - q_k(\mathbf{p}_{opt}) = q_k(\mathbf{s} + \mathbf{p}_{opt}) - q_k(\mathbf{p}_{opt}) \\ &= \frac{1}{2}\mathbf{s}^T B_k \mathbf{s} + \mathbf{s}^T(B_k\mathbf{p}_{opt} + \mathbf{g}_k) + \frac{1}{3}\sigma_k(\|\mathbf{v}\|^3 - \|\mathbf{p}_{opt}\|^3) \\ &= \frac{1}{2}\mathbf{s}^T B_k \mathbf{s} - \lambda_{opt}\mathbf{s}^T\mathbf{p}_{opt} \quad \text{(by (17) and (9))} \\ &= \frac{1}{2}\mathbf{s}^T(B_k + \lambda_{opt} I)\mathbf{s} \qquad (19) \\ &\le \frac{\|B_k^{opt}\|}{2}\|\mathbf{s}\|^2 \le 2\|B_k^{opt}\|\|\widetilde{\mathbf{p}} - \mathbf{p}_{opt}\|^2 \quad \text{(by (18))}. \qquad (20) \end{aligned}$$

From

$$(\frac{\lambda_{opt}}{\sigma_k})^2 = \|\mathbf{v}\|^2 = \|\mathbf{s}\|^2 + \|\mathbf{p}_{opt}\|^2 + 2\mathbf{s}^T\mathbf{p}_{opt}$$

and (17), we get $\mathbf{s}^T\mathbf{p}_{opt} = -\|\mathbf{s}\|^2/2 = -\mathbf{s}^T\mathbf{s}/2$. Then, the equality (19) holds. The conclusion in (13) is given based on the above analysis.

Next, we prove the inequality (14). From the definition of q_k^{opt} in (12), for any $\mathbf{p}_j^k$, by $\|\mathbf{p}_{opt}\| = \|\mathbf{p}_j^k\|$, we have

$$\begin{aligned} q_k^{opt}(\mathbf{p}_j^k) - q_k^{opt}(\mathbf{p}_{opt}) &= \left(q_k(\mathbf{p}_j^k) + \frac{1}{2}\lambda_{opt}\|\mathbf{p}_j^k\|^2\right) - \left(q_k(\mathbf{p}_{opt}) + \frac{1}{2}\lambda_{opt}\|\mathbf{p}_{opt}\|^2\right) \\ &= q_k(\mathbf{p}_j^k) - q_k(\mathbf{p}_{opt}). \end{aligned} \tag{21}$$

Furthermore, we obtain

$$\begin{aligned} q_k^{opt}(\mathbf{p}_j^k) - q_k^{opt}(\mathbf{p}_{opt}) &= \frac{1}{2}(\mathbf{p}_j^k)^T B_k^{opt}\mathbf{p}_j^k + (\mathbf{p}_j^k)^T\mathbf{g}_k - \frac{1}{2}\mathbf{p}_{opt}^T B_k^{opt}\mathbf{p}_{opt} - \mathbf{p}_{opt}^T\mathbf{g}_k \\ &= \frac{1}{2}(\mathbf{p}_j^k - \mathbf{p}_{opt})^T B_k^{opt}(\mathbf{p}_j^k - \mathbf{p}_{opt}), \end{aligned}$$

where the last equality follows from (9) and (10). Then,

$$q_k^{opt}(\mathbf{p}_j^k) - q_k^{opt}(\mathbf{p}_{opt}) \geq \frac{1}{2}(\theta_1 + \lambda_{opt})\|\mathbf{p}_j^k - \mathbf{p}_{opt}\|^2. \tag{22}$$

Combining (13), (21), and (22), we get

$$\frac{1}{2}(\theta_1 + \lambda_{opt})\|\mathbf{p}_j^k - \mathbf{p}_{opt}\|^2 \leq q_k(\mathbf{p}_j^k) - q_k(\mathbf{p}_{opt}) \leq 2\|B_k^{opt}\|\|\widetilde{\mathbf{p}} - \mathbf{p}_{opt}\|^2.$$

The inequality (14) holds. □

5. Numerical Experiments

In order to show the efficiency of the Lanczos for improving the adaptive cubic regularization algorithm, we perform the following two numerical experiments. In this section, we compare the numerical performances of the adaptive cubic regularization algorithm when using the Lanczos approximation method (ACRL) and the adaptive cubic regularization algorithm by just using Newton's method (ACRN) for unconstrained optimization problems.

The ACRL and ACRN algorithms are implemented with the following parameters

$$\eta_1 = 0.1, \eta_2 = 0.8, \gamma_1 = 0.25, \gamma_2 = 1.2, \text{and } \gamma_3 = 2.$$

Convergence in both algorithms for the sub-problem occurs as soon as

$$\|\nabla q_k(\mathbf{p}_j^k)\| \leq \min\left(0.0001, \frac{\|\mathbf{p}_j^k\|}{\max(1, \sigma_k)}\right)\|\nabla q_k(0)\|$$

or if more than the maximum number of iterations has been performed, which we set to 2000. All numerical experiments in this paper were performed on a laptop with i5-10210U CPU at 1.60 GHz and 16.0 GB of RAM.

Example 1 (Generalized Rosenbrock function [21])**.** *The Generalized Rosenbrock function is a non-convex function, introduced by Howard H. Rosenbrock in 1960, which is defined as follows:*

$$f(\mathbf{x}) = \sum_{i=1}^{n-1} c(\mathbf{x}_{i+1} - \mathbf{x}_i^2)^2 + (1 - \mathbf{x}_i)^2, c = 100. \tag{23}$$

From the (23), the solution $\mathbf{x}^\star$ is obviously $(1, 1, \ldots, 1)^T$, and the minimum $f(\mathbf{x}^\star) = 0$.

In Table 1, we show the results of the ACRL and the ACRN for computing the minima of the Generalized Rosenbrock function, with variables from 10 to 2000. In addition to the dimensions of the Generalized Rosenbrock function, we give the number of iterations ("Iter."), the total CPU time required in seconds and the relative error between the computational result and the exact minimum ("Err."). It can be seen that, using the Lanczos method to solve the adaptive cubic regularization sub-problem of Generalized Rosenbrock function is much more efficient than not using the Lanczos method. Moreover, it is not only faster, but also more accurate to calculate, especially when the scale is relatively large.

Example 2 (Eigenvalues of tensors arising from hypergraphs)**.** *Next, we consider the problem of computing extreme eigenvalues of sparse tensors arising from a hypergraph. An adaptive cubic regularization method on a Stiefel manifold named ACRCET is proposed to solve the eigenvalues of tensors [22]. We compare the numerical performances of the ACRL and the ACRN method when applying to the sub-problem of ACRCET. Before going to the experiment part, we first introduce the concepts of tensor eigenvalues and hypergraphs.*

Table 1. Results for computing the minima of the Generalized Rosenbrock function.

n	ACRL			ACRN		
	Iter.	Time (s)	Err.	Iter.	Time (s)	Err.
10	11	0.01	4.05×10^{-14}	8	0.01	4.85×10^{-11}
500	17	0.05	8.33×10^{-13}	12	0.88	1.10×10^{-13}
1000	21	0.27	7.35×10^{-13}	12	4.56	5.53×10^{-11}
5000	23	6.24	1.47×10^{-15}	9	191.95	1.41×10^{-12}
10,000	21	20.33	2.42×10^{-15}	11	1524.88	6.29×10^{-13}
20,000	21	77.8	6.90×10^{-13}	14	16,166.62	3.03×10^{-11}

A real mth order n-dimensional tensor $\mathcal{A} \in \mathbb{R}^{[m,n]}$ has n^m entries:

$$\{a_{i_1 i_2 \cdots i_m}\}$$

for $i_j \in \{1, 2, \cdots, n\}$ and $j \in \{1, 2, \cdots, m\}$. If the value of $\{a_{i_1 i_2 \cdots i_m}\}$ is invariable under any permutation of its indices, $\mathcal{A}$ is a symmetric tensor.

Qi [23] defined a scalar $\Lambda \in \mathbb{R}$ as a Z-eigenvalue of $\mathcal{A}$ and a nonzero vector $\mathbf{x} \in \mathbb{R}^n$ as its associated Z-eigenvector if they satisfy

$$\mathcal{A}\mathbf{x}^{m-1} = \Lambda \mathbf{x} \quad \text{and} \quad \mathbf{x}^T \mathbf{x} = 1.$$

Definition 1 (Hypergraph)**.** *A hypergraph is defined as $G = (V, E)$, where $V = \{1, 2, \cdots, n\}$ is the vertex set and $E = \{e_1, e_2, \cdots, e_m\}$ is the edge set for $e_p \subset V$, $p = 1, 2, \cdots, m$. If $|e_p| = r \geq 2$ for $p = 1, 2, \cdots, m$ and $e_i \neq e_j$ when $i \neq j$, we call G an r-uniform hypergraph.*

For each vertex $i \in V$, the degree $d(i)$ is defined as

$$d(i) = |\{e_p : i \in e_p, e_p \in E\}|.$$

Definition 2 (adjacency tensor and Laplacian tensor)**.** *The adjacency tensor $\mathcal{A} \in \mathbb{R}^{[m,n]}$ of a m-uniform hypergraph G is a symmetric tensor with entries*

$$a_{i_1 \cdots i_m} = \begin{cases} \frac{1}{(m-1)!} & \text{if } \{i_1, \cdots, i_m\} \in E, \\ 0 & \text{otherwise.} \end{cases}$$

For an m-uniform hypergraph G, the degree tensor $\mathcal{D}$ is a diagonal tensor whose ith diagonal element is $d(i)$. Then, the Laplacian tensor $\mathcal{L}$ is defined as

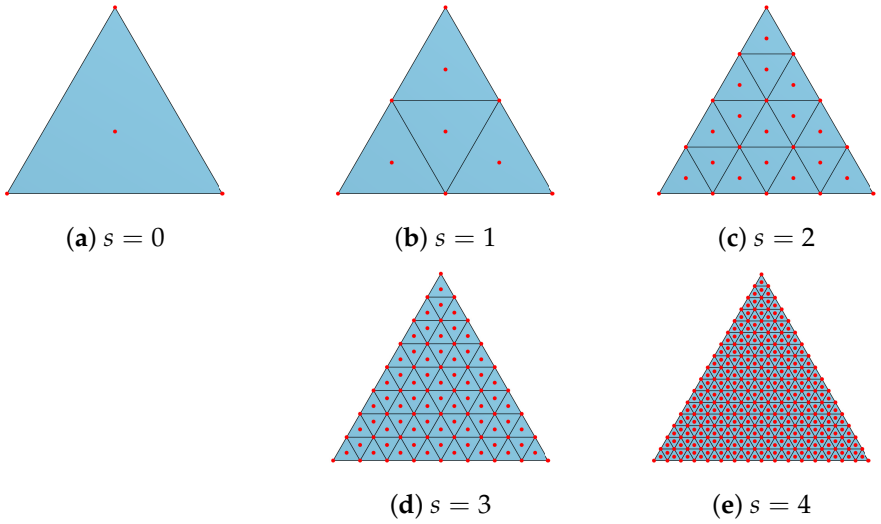
$$\mathcal{L} = \mathcal{D} - \mathcal{A}.$$

A triangle has three vertices and three edges. In this example, we subdivide the triangles by connecting the midpoints of each edge of the triangles. Then, the s-order subdivision of a triangle has 4^s faces, and each face is a triangle. As shown in Figure 1, three vertices as well as the center of the triangles are regarded as an edge of a 4-uniform graph G_T^s.

We compute the largest Z-eigenvalue of the Laplacian tensor $\mathcal{L}(G_T^s)$ via the ACRCET method, using ACRL and ACRN, respectively. In each run, 10 points on the unit sphere are randomly chosen, and 10 estimated eigenvalues are calculated. Then, we take the best one as the estimated largest eigenvalue. For different subdivision order s, the computation results, including the estimated largest Z-eigenvalue, the total number of iterations, and the total CPU time (in seconds) of the 10 runs are reported in Table 2.

It can be seen that both the ACRL and the ACRN find all the largest eigenvalues. However, the ACRL takes almost no time compared to the ACRN. When $s = 6$, the ACRL method only costs 236 s, while the ACRN needs 103,900 s. The numerical comparison

between the ACRL and the ACRN verifies that the Lanczos method dramatically accelerates the running speed when solving the ACR sub-problem (3), and is powerful for large-scale problems.



(**a**) $s = 0$ (**b**) $s = 1$ (**c**) $s = 2$

(**d**) $s = 3$ (**e**) $s = 4$

Figure 1. Four-uniform hypergraphs: subdivision of a triangle.

Table 2. Results for finding the largest Z-eigenvalues of $\mathcal{L}(G_T^s)$.

				ACRL			ACRN	
s	**m**	**n**	**Iter.**	**Time (s)**	$\lambda_{\max}^Z(\mathcal{L}(G_T^s))$	**Iter.**	**Time (s)**	$\lambda_{\max}^Z(\mathcal{L}(G_T^s))$
1	10	4	53	0.04	3	53	0.05	3
2	31	16	57	0.05	6	56	0.17	6
3	109	64	55	0.08	6	54	0.79	6
4	409	256	56	0.78	6	54	13.05	6
5	1585	1024	67	13.02	6	66	727.77	6
6	6241	4096	81	236.35	6	85	103,900	6

6. Conclusions

In this paper, we have used the Lanczos method to solve the adaptive cubic regularization method sub-problem (ACRL). The ACRL method first projects a large-scale ACR sub-problem (3) into a much smaller sub-problem (7) using the Lanczos method, and then solves the smaller sub-problem (7) using the Newton's method. For the convergence analysis, we also established prior error bounds on the differences between the approximate objective value $q_k(\mathbf{p}_j^k)$ and the approximate solution $\mathbf{p}_j^k$ with its corresponding optimal ones. Numerical experiments illustrate that the ACRL method greatly improves the computing efficiency and performs well, even for large-scale problems.

Author Contributions: Methodology, Z.Z. and J.C.; writing—original draft preparation, Z.Z.; writing—review and editing, J.C.; supervision, J.C.; funding acquisition, J.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant No. 11901118 and No. 62073087.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cartis, C.; Gould, N.I.; Toint, P.L. Adaptive cubic regularisation methods for unconstrained optimization. Part I: Motivation, convergence and numerical results. *Math. Program.* **2011**, *127*, 245–295. [CrossRef]
2. Moré, J.J.; Sorensen, D.C. Computing a trust region step. *SIAM J. Sci. Stat. Comput.* **1983**, *4*, 553–572. [CrossRef]
3. Sorensen, D.C. Minimization of a large-scale quadratic functionsubject to a spherical constraint. *SIAM J. Optim.* **1997**, *7*, 141–161. [CrossRef]
4. Rojas, M.; Santos, S.A.; Sorensen, D.C. A new matrix-free algorithm for the large-scale trust-region subproblem. *SIAM J. Optim.* **2000**, *11*, 611–646. [CrossRef]
5. Rendl, F.; Wolkowicz, H. A semidefinite framework for trust region subproblems with applications to large scale minimization. *Math. Program.* **1997**, *77*, 273–299. [CrossRef]
6. Hager, W.W. Minimizing a quadratic over a sphere. *SIAM J. Optim.* **2001**, *12*, 188–208. [CrossRef]
7. Erway, J.B.; Gill, P.E.; Griffin, J.D. Iterative methods for finding a trust-region step. *SIAM J. Optim.* **2009**, *20*, 1110–1131. [CrossRef]
8. Gould, N.I.; Lucidi, S.; Roma, M.; Toint, P.L. Solving the trust-region subproblem using the Lanczos method. *SIAM J. Optim.* **1999**, *9*, 504–525. [CrossRef]
9. Conn, A.R.; Gould, N.I.; Toint, P.L. *Trust Region Methods*; SIAM: Philadelphia, PA, USA, 2000; pp. 91–105.
10. Steihaug, T. The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Numer. Anal.* **1983**, *20*, 626–637. [CrossRef]
11. Toint, P. Towards an efficient sparsity exploiting Newton method for minimization. In *Sparse Matrices and Their Uses*; Academic Press: Cambridge, MA, USA, 1981; pp. 57–88.
12. Zhang, L.H.; Shen, C.; Li, R.C. On the generalized Lanczos trust-region method. *SIAM J. Optim.* **2017**, *27*, 2110–2142. [CrossRef]
13. Zhang, L.; Yang, W.; Shen, C.; Feng, J. Error bounds of Lanczos approach for trust-region subproblem. *Front. Math. China* **2018**, *13*, 459–481. [CrossRef]
14. Carmon, Y.; Duchi, J. Gradient descent finds the cubic-regularized nonconvex Newton step. *SIAM J. Optim.* **2019**, *29*, 2146–2178. [CrossRef]
15. Birgin, E.G.; Martínez, J.M. A Newton-like method with mixed factorizations and cubic regularization for unconstrained minimization. *Comput. Optim. Appl.* **2019**, *73*, 707–753. [CrossRef]
16. Brás, C.P.; Martínez, J.M.; Raydan, M. Large-scale unconstrained optimization using separable cubic modeling and matrix-free subspace minimization. *Comput. Optim. Appl.* **2020**, *75*, 169–205. [CrossRef]
17. Jiang, R.; Yue, M.C.; Zhou, Z. An accelerated first-order method with complexity analysis for solving cubic regularization subproblems. *Comput. Optim. Appl.* **2021**, *79*, 471–506. [CrossRef]
18. Cartis, C.; Gould, N.I.; Toint, P.L. Adaptive cubic regularisation methods for unconstrained optimization. Part II: Worst-case function-and derivative-evaluation complexity. *Math. Program.* **2011**, *130*, 295–319. [CrossRef]
19. Nocedal, J.; Wright, S.J. *Numerical Optimization*; Springer: Berlin/Heidelberg, Germany, 1999; pp. 69–71.
20. Parlett, B.N.; Reid, J.K. Tracking the Progress of the Lanczos Algorithm for Large Symmetric Eigenproblems. *IMA J. Numer. Anal.* **1981**, *1*, 135–155. [CrossRef]
21. Andrei, N. An unconstrained optimization test functions collection. *Adv. Model. Optim.* **2008**, *10*, 147–161. [CrossRef].
22. Chang, J.; Zhu, Z. An adaptive cubic regularization method for computing extreme eigenvalues of tensors. *arXiv* **2022**, arXiv:2209.04971.
23. Qi, L. Eigenvalues of a real supersymmetric tensor. *J. Symb. Comput.* **2005**, *40*, 1302–1324. [CrossRef]