Master's Thesis

Practical Implementations of the Adaptive Regularized Newton Method

Guidance

Associate Professor Nobuo Yamashita

Masataka Nishimori

Department of Applied Mathematics and Physics

Graduate School of Informatics

Kyoto University

February 2012

Abstract

In this paper, we consider the Adaptive Regularized Newton Method (ARNM), which is a method designed to solve the unconstrained minimization problem. ARNM is based on the Newton method, and ensures the global convergence by adjusting a regularized parameter without line search. It has nice convergence properties under some reasonable conditions. However, the current implementation of ARNM uses the minimum eigenvalue of the Hessian of the objective function at each iteration. Furthermore, it requires many iterations for solving some ill-conditioned problems.

In this paper, we propose practical implementations of ARNM that overcome the above problems. First, we implement ARNM with a nonmonotone search technique in order to reduce the number of iterations for some ill-conditioned problems. The technique allows us to update an iterate even if the objective function value is increasing. Moreover, we determine the search direction by using the modified Cholesky factorization proposed by Cheng and Higham. By using it, we do not need to calculate the minimum eigenvalue of Hessian directly. Finally, we present some numerical results and investigate the validity of the proposed algorithm as compared to the conventional ARNM. As a result, we have confirmed that the proposed implementation is practical because it can find a solution in less iterations than the conventional ARNM.

目 次

1 Introduction

In this paper, we consider the unconstrained minimization problem

$$\min_{x\in\mathbb{R}^n} \quad f(x), \tag{1.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a twice continuously differentiable function. Problem (1.1) has various applications in management science, engineering and economics. Here, we propose practical implementations of the regularized Newton method [15] for the problem (1.1) and evaluate their performance by means of several numerical experiments.

For solving problem (1.1), many solution methods, such as steepest descent methods and Newton-type methods, have been developed [2, 11]. Among them, the trust region Newton method has good theoretical convergence properties, such as global convergence and superlinear convergence. However, it has to solve a nonconvex subproblem at each iteration in order to get a search direction. Thus, its total computational time may become too large when solving the subproblem exactly. In order to overcome this difficulty, Ueda and Yamashita [14] proposed the regularized Newton method using the the trust region method technique. They called their method the Adaptive Regularized Newton Method (ARNM). For a given x_k at the k-th iteration, the ARNM defines the following quadratic model function $m_k : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ of f.

$$m_k(d,\nu) := f(x_k) + \nabla f(x_k)^T d + \frac{1}{2} d^T \left(\nabla^2 f(x_k) + E_k(\nu)\right) d,$$

where ^{T} denotes transposition of a vector, $E_k(\nu)$ is a symmetric matrix with $\nabla^2 f(x_k) + E_k(\nu)$ being symmetric positive definite, and ν is a positive parameter which represents the magnitude of the minimum eigenvalue of $\nabla^2 f(x_k) + E_k(\nu)$. When $E_k(\nu) \equiv 0$, the model function becomes a quadratic approximation of f. When $\nabla^2 f(x_k)$ is positive definite, the pure Newton method adopts a minimum of m_k with $E_k(\nu) = 0$ as search direction. However, $\nabla^2 f(x_k)$ is not necessarily positive definite. For such cases, the regularized Newton method adds the regularized matrix $E_k(\nu)$ in order to make $\nabla^2 f(x_k) + E_k(\nu)$ positive definite. As a concrete example of $E_k(\nu)$, Ueda and Yamashita [15] proposed the following matrix.

$$\bar{E}_k(\nu) = \left(c \max\left(0, -\lambda_{\min}\left(\nabla^2 f(x_k)\right)\right) + \nu \|\nabla f(x_k)\|^\delta\right) I, \tag{1.2}$$

where $c > 1$ and $\delta > 0$ are given constants, $\lambda_{\min}\left(\nabla^2 f(x_k)\right)$ is the minimum eigenvalue of $\nabla^2 f(x_k)$ and $I \in \mathbb{R}^{n\times n}$ is the identity matrix. Note that the minimum eigenvalue of $\nabla^2 f(x_k) + \bar{E}_k(\nu)$ is $\nu\|\nabla f(x_k)\|^\delta$. Therefore $\nu\|\nabla f(x_k)\|^\delta$ is regarded as a regularized parameter of the model function m_k.

Since $\nabla^2 f(x_k) + E_k(\nu)$ is symmetric positive definite, the model function $m_k(\cdot,\nu)$ is strongly convex. Therefore, we can get the global minimum of $m_k(\cdot,\nu)$ by solving linear equations. For some parameter ν_k, ARNM adopts the global minimizer of m_k as the search direction $d_k(\nu_k)$, that is,

$$d_k(\nu_k) = \operatorname*{argmin}_{d\in\mathbb{R}^n} m_k(d,\nu_k) = -\left(\nabla^2 f(x_k) + E_k(\nu_k)\right)^{-1} \nabla f(x_k). \tag{1.3}$$

The usual regularized Newton methods with line search update the next iterate by $x_{k+1} := x_k + t_k d_k(\nu_k)$, where t_k is a step size. ARNM does not use a line search, instead it controls the regularized

matrix $E_k(\nu_k)$. If $f(x_k + d_k(\nu_k))$ sufficiently decreases as compared to $f(x_k)$, ARNM sets $x_{k+1} := x_k + d_k(\nu_k)$. Otherwise, it increases the regularized parameter ν_k, that is, the minimum eigenvalue of $\nabla^2 f(x_k) + E_k(\nu)$, and gets the new search direction (1.3) with the renewed $E_k(\nu_k)$. If ν_k is sufficiently large and matrix $E_k(\nu)$ is given by (1.2), the search direction $d_k(\nu_k)$ tends to be the steepest descent direction, and hence ARNM behaves like a steepest descent method. Therefore, $f(x_{k+1}) < f(x_k)$ holds at each iteration, and hence the global convergence of ARNM is guaranteed. Moreover, ARNM has the following two nice theoretical convergence properties. First, ARNM has superlinear convergence under some reasonable assumptions such as a local error bound condition. Second, its upper bound of the number of iterations so that $\|\nabla f(x_k)\| < \epsilon$ is $O(\epsilon^{-2})$. In [14], the author proved that the same convergence properties hold even if we adopt more general matrix $E_k(\nu)$. The details are given in Assumption 2.1 of the next section. In this paper, we do not use $\bar{E}_k(\nu)$ given by (1.2), and we will use a more general matrix $E_k(\nu)$ which satisfies the assumption.

When we can calculate the minimum eigenvalue of $\nabla^2 f(x_k)$ exactly, we can get the search direction $d_k(\nu_k)$ with $\bar{E}_k(\nu)$ of (1.2). Ueda and Yamashita [15] presented some numerical results for such cases. The results seem promising if we ignore the calculation time of the minimum eigenvalues. However, we cannot ignore it for some large-scale problems. Moreover, ARNM takes many iterations to obtain a solution for some test problems [15]. Thus, we must consider the following three issues in order to make ARNM more practical.

Difficulty 1. ARNM controls ν_k in order to decrease the objective value at each iteration. Therefore, ν_k tends to become too large for some ill-conditioned problems, which causes numerical difficulties.

Difficulty 2. To ensure the good convergence properties, $E_k(\nu)$ must satisfy some conditions (Assumption 1.2) related to the minimum eigenvalue of $\nabla^2 f(x_k)$. Therefore, we must evaluate the minimum eigenvalue in some ways. In fact, Ueda and Yamashita [15] used the minimum eigenvalue itself. It may take much time.

Difficulty 3. If the objective value $f(x_k + d_k(\nu_k))$ does not decrease sufficiently, then ARNM must solve the linear equation (1.3) again with a new matrix $E_k(\nu_k)$. As compared to a line search where t_k is chosen by evaluation of the function value only, it may take much time.

In this paper, in order to overcome Difficulty 1, we propose a new algorithm implemented with a nonmonotone technique that is used in the trust region method. We also propose a method that computes the search direction $d_k(\nu_k)$ and $E_k(\nu_k)$ by using a modified Cholesky factorization so that $E_k(\nu_k)$ satisfies the conditions (Assumption 2.1) required in [14] for the nice convergence. Moreover, we investigate the validity of the proposed algorithm by conducting several numerical experiments and comparisons with the conventional ARNM.

The paper is organized as follows. In the first half of Section 2, we introduce the details of the ARNM algorithm proposed in [15], and explain its theoretical properties. In the second half of Section 2, we describe the modified Cholesky factorization algorithm [3] used in our proposed algorithms. In Section 3, we propose a new method that uses the nonmonotone technique and the modified Cholesky factorization. In particular, we introduce the nonmonotone technique to reduce the number of iterations, and describe how to implement it with ARNM. In Section 4, we present some numerical results, comparing the ARNM [15] and the proposed method. We discuss the effect

of the nonmonotone approach, and show that the proposed algorithm actually reduces the number of iterations as compared to the conventional ARNM. Section 5 concludes the paper with some remarks.

The following notations will be used in the subsequent sections. For a vector $x \in \mathbb{R}^n$, $\|x\|$ denotes its Euclidean norm defined by

$$\|x\| := \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}.$$

Given a matrix $A \in \mathbb{R}^{m \times n}$, A_{ij} denotes its (i,j)th element. For a given matrix $A \in \mathbb{R}^{m \times n}$, $\|A\|_F$ denotes the Frobenius norm defined by

$$\|A\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij}^2 \right)^{1/2}.$$

Moreover, for a symmetric matrix $M \in \mathbb{R}^{n \times n}$, we denote the maximum and the minimum eigenvalues of M as $\lambda_{\max}(M)$ and $\lambda_{\min}(M)$, respectively.

2 Preliminaries

In this section, we first describe the detail of ARNM, then we introduce the modified Cholesky factorization that is used in the subsequent sections.

2.1 Adaptive Regularized Newton Method

ARNM is one of the Newton-type methods designed to solve problem (1.1). ARNM defines the quadratic model function $m_k : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ of the objective function f at the k-th iterate x_k,

$$m_k(d, \nu) \quad := \quad f(x_k) + \nabla f(x_k)^T d + \frac{1}{2} d^T \left(\nabla^2 f(x_k) + E_k(\nu) \right) d,$$

where $E_k(\nu)$ is a symmetric matrix which satisfies the following conditions [14]:

Assumption 2.1. *Let δ be a positive constant. We define $\Lambda_k := \max\left(0, -\lambda_{\min}\left(\nabla^2 f(x_k)\right)\right)$. Then, there exists positive constants $\kappa_1, \kappa_2, \kappa_3$ and κ_4 satisfying the following conditions (a),(b) and (c) for any $\nu \in [\nu_{\min}, \infty)$ and k.*

(a) *$\nabla^2 f(x_k) + E_k(\nu)$ is positive definite.*

(b) $\lambda_{\min}\left(\nabla^2 f(x_k) + E_k(\nu)\right) \geq \kappa_1 \Lambda_k + \kappa_2 \nu \|\nabla f(x_k)\|^\delta$.

(c) $\|E_k(\nu)\| \leq \kappa_3 \Lambda_k + \kappa_4 \nu \|\nabla f(x_k)\|^\delta$.

We call the parameter ν, which defines $E_k(\nu)$, a regularized parameter. Ueda and Yamashita [15] use the matrix $\bar{E}_k(\nu)$ defined by equation (1.2) in order to satisfy Assumption 2.1. In fact, the matrix $\bar{E}_k(\nu)$ satisfies the conditions (a),(b) and (c) of the Assumption 2.1. ARNM minimizes the model function m_k for some regularized parameter $\nu_k > 0$ and obtains a search direction, that is,

$$d_k(\nu_k) := \operatorname*{argmin}_{d \in \mathbb{R}^n} m_k(d, \nu_k) = -\left(\nabla^2 f(x_k) + E_k(\nu_k)\right)^{-1} \nabla f(x_k). \tag{2.1}$$

Then, ARNM updates the iterates with $x_{k+1} := x_k + d_k(\nu_k)$ if $f(x_k + d_k(\nu_k))$ sufficiently decreases. Otherwise, it controls ν_k so that $f(x_k + d_k(\nu_k)) < f(x_k)$. The regularized parameter ν_k is controlled in a way similar to the updating of the trust-region radius in the trust region method. If the value of the model function m_k at $x_k + d_k(\nu_k)$ is sufficiently close to that of the objective function, then $f(x_k) - m_k(d_k(\nu_k), \nu_k) \approx f(x_k) - f(x_k + d_k(\nu_k))$ holds. Therefore, we define the ratio related to the validity of the model function at point $x_k + d_k(\nu)$ by

$$\rho_k := \frac{f(x_k) - f(x_k + d_k(\nu))}{f(x_k) - m_k(d_k(\nu), \nu)}.$$

If the ratio ρ_k is near to 1, then we regard the model function value at $x_k + d_k(\nu_k)$ as a good approximation of $f(x_k + d_k(\nu_k))$, and update the iterate with $x_{k+1} := x_k + d_k(\nu_k)$. If the ratio ρ_k is small or negative, then we consider that the accuracy of the approximation is poor, and we increase ν_k. We note that if $\rho_k > 0$, then the inequality $f(x_k + d_k(\nu)) < f(x_k)$ always holds.

Now we describe ARNM more formally.

Adaptive Regularized Newton Method (ARNM)

Step 0: (**Initialization**) Choose parameters $\nu_0, \nu_{\min}, \eta_1, \eta_2, \gamma_1, \gamma_2, \delta, c$ such that

$$0 < \nu_{\min} \leq \nu_0, \quad 0 < \eta_1 \leq \eta_2 \leq 1, \quad 0 < \gamma_1 < 1 \leq \gamma_2, \quad 0 < \delta, \quad 1 < c.$$

Choose a starting point $x_0 \in \mathbb{R}^n$. Set $k := 0$.

Step 1: (**Termination criteria**) If x_k satisfies the termination criteria, then stop. Otherwise go to Step 2.

Step 2: (**Calculate the search direction and the regularized parameter**)

Step 2.0: Set $l_k := 0, \bar{\nu}_{l_k} := \nu_k$.

Step 2.1: (**Calculate the search direction**) Compute the search direction $\bar{d}_{l_k}$ by solving the following system of linear equations:

$$\left(\nabla^2 f(x_k) + E_k(\bar{\nu}_{l_k})\right) d \quad = \quad -\nabla f(x_k),$$

where $E_k(\bar{\nu}_{l_k})$ is a matrix that satisfies Assumption 2.1.

Step 2.2: (**Calculate the ratio** ρ_{l_k}) Calculate the ratio ρ_{l_k} by

$$\rho_{l_k} := \frac{f(x_k) - f(x_k + \bar{d}_{l_k})}{f(x_k) - m_k(\bar{d}_{l_k}, \bar{\nu}_{l_k})}.$$

Step 2.3: (**Update the regularized parameter**) Update the regularized parameter $\bar{\nu}_{l_k}$.

$$\bar{\nu}_{l_k+1} := \begin{cases} \max(\gamma_1 \bar{\nu}_{l_k}, \nu_{\min}) & \text{if } (\rho_{l_k} \geq \eta_2) \\ \bar{\nu}_{l_k} & \text{if } (\rho_{l_k} \in [\eta_1, \eta_2) \\ \gamma_2 \bar{\nu}_{l_k} & \text{if } (\rho_{l_k} < \eta_1) \end{cases}$$

Step 2.4: If $\rho_{l_k} > \eta_1$, then go to Step 3. Otherwise update $l_k := l_k + 1$ and go to Step 2.1.

Step 3: (**Update the iterate**) Set $x_{k+1} := x_k + \bar{d}_{l_k}$, $\nu_{k+1} := \bar{\nu}_{l_k+1}$. Set $k := k + 1$, and go to Step 1.

The following theorems show the convergence properties of ARNM, given in [15].

Theorem 2.1. ***(Global convergence)*** *Suppose that Assumption 2.1 holds. Suppose also that the sequence $\{x_k\}$ generated by ARNM is bounded. Then the accumulation point of $\{x_k\}$ is a stationary point of f.* □

Theorem 2.2. ***(Local convergence)*** *Suppose that Assumption 2.1 holds. Suppose also that the following assumptions (a),(b),(c) and (d) hold.*

(a). $0 < \delta \leq 1$

(b). *There exists a local optimal solution x^* of the problem (1.1).*

(c). $\nabla^2 f(x)$ *is locally Lipschitz continuous, that is, there exist constants* $b \in (0,1)$ *and* $L > 0$ *such that*

$$\|\nabla^2 f(y) - \nabla^2 f(x)\| \leq L\|x - y\| \quad \textit{for all } x, y \in N(x^*, b).$$

(d). $\|\nabla f(x_k)\|$ *provides a local error bound for the problem (1.1), that is, there exists a constant* $c > 0$ *such that*

$$c \ \mathrm{dist}(x, X^*) \leq \|\nabla f(x_k)\| \quad \textit{for all } x \in N(x^*, b).$$

If we choose the starting point x_0 *sufficiently close to* x^**, then* $\{x_k\}$ *converges at the rate of* $1 + \delta$*.* □

Theorem 2.3. ***(Global complexity bound)*** *Suppose that Assumption 2.1 holds. Suppose also that the following assumptions (a),(b) and (c) hold.*

(a). $\delta \leq \frac{1}{2}$

(b). *The sequence* $\{x_k\}$ *generated by ARNM is bounded.*

(c). $\nabla^2 f(x_k)$ *is Lipschitz continuous.*

Then for a given constant $\epsilon > 0$*, an upper bound of the number of iteration so that* $\|\nabla f(x_k)\| \leq \epsilon$ *is* $O(\epsilon^{-2})$*.* □

2.2 Modified Cholesky Factorizations

Here, we describe the modified Cholesky factorization that is used to overcome Difficulties 2 and 3 mentioned in Introduction. At first, we begin with the classical Cholesky factorization which is the base of the modified Cholesky factorizations. The Cholesky factorization aims to factorize a positive symmetric matrix $A \in \mathbb{R}^{n \times n}$ into

$$A = \ L \ D \ L^T, \tag{2.2}$$

where L is a lower triangular matrix with unit diagonal elements and D is a diagonal matrix with positive elements. In the following, we denote d_i as the i-th diagonal element of the diagonal matrix D. The algorithm of the Cholesky factorization is given in detail below.

Cholesky Factorization with LDL^T Form

Let $A \in \mathbb{R}^{n\times n}$ be a given symmetric positive definite matrix.

for $j = 1, 2, \ldots, n$

$$d_j = A_{jj} - \sum_{s=1}^{j-1} d_s L_{js}^2 \tag{2.3}$$

for $i = j+1, \ldots, n$

$$L_{ij} = \left(A_{ij} - \sum_{s=1}^{j-1} d_s L_{is} L_{js} \right) / d_j \tag{2.4}$$

end

end

If and only if all the elements d_j in D, presented in the Cholesky factorization, are positive, then the symmetric matrix $A \in \mathbb{R}^{n\times n}$ is positive definite. If A is indefinite, the Cholesky factorization may not exist. Even if A is positive definite, d_j may become too small, and the factorization is numerically unstable. In such cases, some elements of L and D can become too large. The modified Cholesky factorization executes the Cholesky factorization of $A + E$ by adding a matrix E, so that $A + E$ is positive definite. Note that the matrix E is not necessarily computed in advance. It is constructed during iterations of the Cholesky factorization. It is desirable that the norm of the matrix E is small from practical viewpoints. In the following, we introduce GMW81, proposed by Gill, Murray and Wright [9], and CH98, proposed by Cheng and Higham [3], which compute a small E.

GMW81 first chooses two positive parameters β and γ that control the magnitude of d_j. Then it modifies d_j during the computation of the j-th columns of L and D ((2.3) and (2.4)) so that:

$$d_j \geq \gamma, \quad L_{ij}\sqrt{d_j} \leq \beta, \quad \text{for } i = j+1, j+2, \ldots, n.$$

In order to satisfy the above conditions, we replace the diagonal elements d_j in the Cholesky factorization as follows.

$$d_j = \max\left(A_{jj} - \sum_{s=1}^{j-1} d_s L_{js}^2 \ , \ \left(\frac{\theta_j}{\beta}\right)^2 \ , \ \gamma \right) \quad \text{with } \theta_j = \max_{j<i\leq n} |L_{ij} d_j|.$$

These observations of the modified Cholesky algorithm are described in detail in Gill, Murray, and Wright [9]. Paper [9] introduces symmetric interchanges of rows and columns in an attempt to reduce the size of the modification $\|E\|$. If P denotes the permutation matrix associated with the row and column interchanges, then,

$$P\ A\ P^T + E = \ L\ D\ L^T,$$

where E is a nonnegative diagonal matrix equal to zero if A is sufficiently positive definite. In the literature, many Cholesky decompositions based on this idea were considered [1, 3, 7, 8, 12], and can be applied to modified Newton methods. However, the norm of E relies on the process of the algorithm, and in general we cannot control the size of such norm by using the parameters γ and β. The reason is that GMW81 cannot ensure the boundedness of L. Therefore, ARNM cannot use the $E_k(\nu)$ generated by GMW81 to satisfy Assumption 2.1.

Algorithm CH98 ensures the boundedness of the matrix L. CH98 uses the bounded Bunch-Kaufman pivoting strategy which factorizes the indefinite symmetric factorization. In the following, we describe the factorization approach. Any symmetric matrix A, positive definite or not, can be written as

$$P\ A\ P^T = \ L\ B\ L^T, \tag{2.5}$$

where L is a unit lower triangular, B is a block diagonal with blocks of dimension 1 or 2, and P is a permutation matrix. The symmetric indefinite factorization allows us to determine the *inertia* of a matrix, that is, the number of positive, zero, and negative eigenvalues. The *inertia* of B equals the *inertia* of A. We can achieve the decomposition by using a bounded Bunch-Kaufman pivoting strategy.

Afterwards, we modify the block diagonal matrix B to calculate E such that $PAP+E$ is positive definite. First, the algorithm computes the spectral decomposition $B = Q\Lambda Q^T$, where Q is an orthogonal matrix, and $\Lambda = \text{diag}(\lambda_i)$ with λ_i being the i-th eigenvalue of the matrix B. We denote $\text{diag}(\lambda_i)$ as the diagonal matrix such that the i-th diagonal element is λ_i. This spectral decomposition is not expensive to compute, because B is a block diagonal matrix such that the dimension of the block is at most 2. By using the spectral decomposition, we compute ΔB such that $B + \Delta B$ is sufficiently positive definite. To achieve this, we define ΔB as

$$\Delta B := Q\ \text{diag}(\tau_i)\ Q^T, \quad \tau_i = \begin{cases} 0, & \lambda_i \geq \xi \\ \xi - \lambda_i, & \lambda_i < \xi \end{cases} \quad i = 1, 2, ..., n,$$

where λ_i is the i-th eigenvalue of B. The matrix ΔB is thus a modification of the minimum Frobenius norm that ensures that all eigenvalues of the modified matrix $B + \Delta B$ are no less than ξ. The strategy therefore modifies the factorization (2.5) as follows:

$$P\ \ (A + E)\ \ P^T = L\ \ (B + \Delta B)\ \ L^T, \tag{2.6}$$

where $E = P^T L F L^T P$ is a symmetric matrix such that $A + E$ is positive definite, and is not a diagonal matrix in general. With this modification, we can change the matrix A in order to satisfy $\lambda_{\min}(A + E) \approx \xi$ whenever the original matrix A has $\lambda_{\min}(A) < \xi$.

In the following, we describe the CH98 algorithm more formally.

Algorithm CH98

Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix and ξ be a positive parameter.

Step 1. By using the bounded Bunch-Kaufman pivoting strategy [1], we factorize A as

$$PAP^T = LBL^T, \tag{2.7}$$

where P is a permutation matrix, L is a unit lower triangular matrix, and B is a block diagonal matrix with diagonal blocks of dimension 1 or 2.

Step 2. We define $B_{11}, B_{22}, \ldots, B_{mm}$ as the blocks of the block diagonal matrix B in order from upper left. For the block B_{ii} $(i = 1, 2, \ldots, m)$,

(a). If the dimension of the block is 1, we have:

$$\tilde{B}_{ii} := \max\left(B_{ii}, \xi\right)$$

(b). If the dimension of the block is 2, we have: we denote λ_1^i, λ_2^i as eigenvalues of block B_{ii}. We compute a spectral decomposition of $\tilde{B}_{ii}$ as

$$\tilde{B}_{ii} := V \begin{bmatrix} \lambda_1^i & 0 \\ 0 & \lambda_2^i \end{bmatrix} V^T,$$

where V is an orthogonal matrix $(V \cdot V^T = I)$. By adding the following value to the λ_1^i, λ_2^i, we regularize block $\tilde{B_{ii}}$.

$$\tilde{\lambda}_j^i = \max\left(\lambda_j^i, \xi\right), \quad \text{for } j = 0, 1$$

By using these λ_1^i and λ_2^i, we compute the block $\tilde{B}_i$ as

$$\tilde{B}_i = V \begin{bmatrix} \tilde{\lambda}_1^i & 0 \\ 0 & \tilde{\lambda}_2^i \end{bmatrix} V^T$$

We denote $\tilde{B}$ as the block diagonal matrix with blocks $\tilde{B}_{11}, \ldots, \tilde{B}_{mm}$ in order from upper left.

Thus, P, L and $B + \Delta B$ are obtained.

By using CH98, L is bounded regardless of the value of A, that is,

$$\begin{aligned} \lambda_{\max}\left(LL^T\right) &\leq 4n^2 - 3n, \\ \lambda_{\min}\left(LL^T\right) &\geq (3.781)^{2-2n}. \end{aligned}$$

We note that indeed P is not a permutation matrix for maintaining the sparsity of L, P is used for maintaining the boundedness of L. Therefore, L is not a sparse matrix in general, and this algorithm is not suitable for large-scale problem. Nevertheless, as benefits of this approach, once we obtain the decomposition of A, we can calculate the decomposition (2.6) in order to perform Step 2 for different ξ. By changing ξ, there is not need to compute the Step 1 any more.

3 The practical implementations of ARNM

In this section, we propose ARNM, implemented with a nonmonotone search technique. Moreover, we describe an efficient way to compute the search direction by using the modified Cholesky factorization.

3.1 Nonmonotone ARNM

As mentioned in the introduction, ARNM suffers from Difficulty 1, where the convergence speed of ARNM may become too slow. The reason for why this difficulty occurs is that ARNM must reduce the objective value at each iteration. Therefore, it can be avoided by changing the updating rules of ARNM so that it becomes able to update iterations even if the objective value does not decrease. One of such approaches to is to use the nonmonotone algorithm [4]. The nonmonotone algorithm adopts the search direction even if the objective value does not reduce.

We consider that we implement the similar method used in the nonmonotone trust region [4], within ARNM. First, we change the definition of the ratio ρ_{l_k} in Step 2.2 of ARNM as follows.

$$\rho_{l_k} := \frac{f(x_{r(k)}) - f(x_k + \bar{d}_{l_k})}{f(x_k) - m_k(\bar{d}_{l_k}, \bar{\nu}_{l_k})}, \tag{3.1}$$

where $f(x_{r(k)})$ is the objective value of iteration $x_{r(k)}$ at $r(k)$ as

$$r(k) \leq k, \quad f(x_k) \leq f(x_{r(k)}). \tag{3.2}$$

Various choices of $r(k)$ for the above condition are proposed [4, 5, 13, 17]. In this paper, we use a "sliding window" technique which selects the reference iteration $r(k)$ by requiring that

$$r(k) := \underset{i=0,\ldots,\min(k,m)}{\operatorname{argmax}} f(x_{k-i}).$$

Then the the condition $\rho_{l_k} > \eta_1$ can be satisfied even if $f(x_{k+1}) > f(x_k)$. Thus, the ARNM it updates the iteration. Meanwhile, the objective values of the sequence generated by the proposed algorithm does not decrease monotonely. However, when this nonmonotone algorithm is used with the trust region method, it ensures the global convergence of the trust region method. By using the same analysis, we can also ensure the global convergence in ARNM. Meanwhile, the best value of m is known to be around 25 in the trust region algorithm. We call the algorithm that implements the nonmonotone approach into ARNM as NM-ARNM. In the following, we describe the difference when compared to ARNM.

NM-ARNM Step 2.2 (Calculation of the ratio ρ_{l_k})

Step 2.2: (Calculation of the ratio ρ_{l_k}) Select the reference iteration

$$r(k) = \underset{i=0,\ldots,\min(k,m)}{\operatorname{argmax}} f(x_{k-i}), \tag{3.3}$$

and compute

$$\rho_{l_k} = \frac{f(x_{r(k)}) - f(x_k + \bar{d}_{l_k})}{f(x_k) - m_k(\bar{d}_{l_k}, \bar{\nu}_{l_k})}.$$

The overhead in computing $r(k)$ by (3.3) compared to Step 2.2 of ARNM is of order of m operations, which is considered negligible compared to the cost of computing $d_k(\nu_k)$, except possibly for very small problems.

3.2 How to Calculate the Search Direction with Modified Cholesky Factorization

Next, we describe the way to calculate the concrete search direction of ARNM in order to overcome Difficulties 2 and 3 of the conventional ARNM. Ueda and Yamashita [15] adopt

$$\bar{E}_k(\nu) := \left(c\Lambda_k + \nu\|\nabla f(x_k)\|^\delta\right) I,$$

as $E_k(\nu)$ in Step 2.1. Here, we note again that $\Lambda_k := \max\left(0, -\lambda_{\min}\left(\nabla^2 f(x_k)\right)\right)$. This is the reasonable way when we can compute $\lambda_{\min}\left(\nabla^2 f(x_k)\right)$ easily.

However, if we can not calculate $\lambda_{\min}\left(\nabla^2 f(x_k)\right)$ exactly, ARNM can not ensure the theoretical convergence. This is the Difficulty 2 of the conventional ARNM. To overcome this difficulty, we propose a way to calculate the search direction by using the algorithm CH98, which is introduced in Section 2.2. We denote this proposed algorithm as ARNM-MC where MC stands for Modified Cholesky. ARNM-MC changes Step 2.1 of ARNM as follows.

ARNM-MC Step 2.1 (Calculate the search direction)

Step 2.1(a) Execute Step 1 of CH98 Algorithm for $A = \nabla^2 f(x_k)$, and factorize $\nabla^2 f(x_k)$ to the following format.

$$P_k \nabla^2 f(x_k) P_k^T = L_k B_k L_k^T, \tag{3.4}$$

where L_k is unit lower triangular, B_k is block diagonal with blocks of dimension 1 or 2, and P_k is a permutation matrix.

Step 2.1(b) Substituting $B = B_k$, $\xi = c\ \max(0, -\lambda_{\min}(B_k)) + \bar{\nu}_{l_k}\|\nabla f(x_k)\|^\delta$, and execute Step 2 of CH98 Algorithm and calculate $\Delta B_k(\bar{\nu}_{l_k})$. As a result, $\nabla^2 f(x_k)$ is factorized to the following format.

$$P_k\left(\nabla^2 f(x_k) + E_k(\bar{\nu}_{l_k})\right) P_k^T = L_k\left(B_k + \Delta B_k(\bar{\nu}_{l_k})\right) L_k^T. \tag{3.5}$$

Step 2.1(c) Calculate the search direction $\bar{d}_{l_k}$ by solving the following linear equations.

$$\left(\nabla^2 f(x_k) + E_k(\bar{\nu}_{l_k})\right) d = -\nabla f(x_k).$$

We note that this proposed algorithm does not calculate the $E_k(\bar{\nu}_{l_k})$ explicitly, and it only calculates the matrices $L_k, B_k + \Delta B_k(\bar{\nu}_{l_k})$ and P_k.

In the following, we discuss the reason for why the Difficulties 2 and 3 of ARNM can be overcomed by using the search direction $d_k(\nu_k)$ which is calculated in Step 2.1 of ARNM-MC.

We begin with Difficulty 2. ARNM uses $\lambda_{\min}\left(\nabla^2 f(x_k)\right)$, which makes difficult to calculate $E_k(\nu)$ exactly and to satisfy the Assumption 2.1. ARNM-MC calculates the matrix $E_k(\nu)$ by

using $\lambda_{\min}(B_k)$ instead of $\lambda_{\min}\left(\nabla^2 f(x_k)\right)$. Since the matrix L is bounded, we can see that the Assumption 2.1 holds, and hence overcome Difficulty 2 by using ARNM-MC.

Next, we describe the reason for why we can overcome Difficulty 3. First, we evaluate the time complexity bound at each step of Step 2.1. In Step 2.1(a), the complexity bound is $O(n^3)$. In Step 2.2(b), the complexity bound is $O(n)$. Then, in Step 2.1(c), we calculate the search direction $d_k(\nu_k)$ by using $P_k^T L_k(B_k + \Delta B_k(\bar{\nu}_{l_k}))L_k^T P_k$ without $E_k(\nu)$. Since L_k is unit lower triangular and $B_k + \Delta B_k(\nu)$ is block diagonal with diagonal blocks of dimension 1 or 2, we can calculate the following linear equations effectively.

$$\left(\nabla^2 f(x_k) + E_k(\nu)\right) d = P_k^T L_k \left(B_k + \Delta B_k(\nu)\right) L_k^T P_k d = -\nabla f(x_k)$$

More specifically, we use the following 5-step process:

$$\begin{aligned} P_k^T z_1 &= -g \\ L_k z_2 &= z_1 \\ \left(B_k + \Delta B_k(\nu)\right) z_3 &= z_2 \\ L_k^T z_4 &= z_3 \\ P_k d &= z_4. \end{aligned}$$

By calculating the search direction like this, the complexity bound of Step 2.1(a)-(c) is $O(n^2)$ when we calculate the search direction with ν increasing and x_k fixed. Since the point x_k has not been updated yet, the Hessian $\nabla^2 f(x_k)$ is the same as if we had changed ν. Therefore, when we update $\bar{\nu}_{l_k}$, we need not to calculate L_k and B_k again. Thus, we can calculate the direction again only to calculate Step 2.1(b) and Step 2.1(c). Hence, the complexity bound is at most $O(n^2)$. Therefore, we can overcome Difficulty 3.

In the following, we call the algorithm constructed by changing Step 2.1 of ARNM to ARNM-MC and Step 2.2 of ARNM to NM-ARNM, as NM-ARNM-MC.

4 Numerical Results

In the previous section, we described our proposed algorithms, whose aim is to overcome the difficulties of ARNM. In this section, we report some numerical results for the proposed algorithms and ARNM.

4.1 Test Environment

All the algorithms were coded in Python 2.7, and run on a machine equiped with a 1.60 GHz Intel Core i7 CPU and 4.00 GB memory. We set the termination criterion as $\|\nabla f(x_k)\| \leq 10^{-5}$. If the number of the iterations exceeds 10^4, then we consider the run as a failure.

For our test, first, we investigate the behavior of the parameters in the proposed algorithms and ARNM. Then, we compare the proposed algorithms with ARNM using the suitable parameters determined. In each experiment, a benchmark of 120 problems were chosen from the collection of test problems CUTEr [10]. We used the initial point x_0 suggested by CUTEr. We note that the objective value of CUTEr problem **STRATEC** is equal to $-\infty$ for a specific finite point. For this reason, some algorithms do not work approximately for **STRATEC**. Therefore, we ignore **STRATEC** from the numerical results. We also note that ARNM computes $\lambda_{\min}\left(\nabla^2 f(x_k)\right)$ not exactly, but only approximately by function numpy.linalg.eigvalsh of Python.

4.1.1 Performance Evaluation

For the comparisons, we use a performance profile [6] which is a tool for evaluating and comparing the efficiency and the robustness of several algorithms. This comparison technique is expressed by the distribution function proposed in [6]. We denote a set of solvers as S, and a set of problems that can be solved by all methods in S as P_S. We also denote a measure for evaluation required to solve a problem p by a solver s as $v_{p,s}$, and the best $v_{p,s}$ for each p as v_p^*, that is, $v_p^* = \min\{v_{p,s} | s \in S\}$. The distribution function $F_s^S(t)$ for a method s is defined by

$$F_s^S(t) = \frac{|\{p \in P_S \mid v_{p,s} \leq t v_p^*\}|}{|P_S|}, \quad t \geq 1,$$

where $|\cdot|$ means the size of a set.

4.2 Choosing the Algorithmic Parameters

Here, we investigate the parameters in the proposed algorithms and ARNM. First, we investigate the influence of the parameters γ_1 and γ_2 in each algorithm. In all numerical experiments, except for γ_1 and γ_2, the parameters of all the algorithms are fixed as follows:

$$\eta_1 = 0.01,\ \eta_2 = 0.8,\ \nu_0 = 1,\ \nu_{\min} = 10^{-5},\ \delta = 2,\ c = 2.$$

Moreover, we adopt the nonmonotone parameter $m = 20$ for NM-ARNM and NM-ARNM-MC.

In order to examine the influence of (γ_1, γ_2), we tested the proposed algorithm for each (γ_1, γ_2) in $\{0.05, 0.1, 0.2, 0.5\} \times \{2, 5, 10, 20, 50, 100, 200, 500\}$. Figures 1, 2, 3 and 4 show the performance profile in terms of function evaluations for ARNM, NM-ARNM, ARNM-MC and NM-ARNM-MC, respectively. From these figures, we see that suitable parameters (γ_1^*, γ_2^*) are

- ARNM: $(\gamma_1^*, \gamma_2^*) = (0.1, 20)$,
- NM-ARNM: $(\gamma_1^*, \gamma_2^*) = (0.1, 100)$,
- ARNM-MC: $(\gamma_1^*, \gamma_2^*) = (0.2, 10)$,
- NM-ARNM-MC: $(\gamma_1^*, \gamma_2^*) = (0.2, 10)$.

Afterwards, we investigate the influence of the parameter m which determines $f(x_{r(k)})$ for NM-ARNM and NM-ARNM-MC. We tested NM-ARNM and NM-ARNM-MC for each m in $\{1, 2, 5, 10, 20, 50, 100\}$. Figures 5 and 6 show the comparisons of m in terms of number of function evaluations for NM-ARNM and NM-ARNM-MC, respectively. Tables 1 and 2 show their details. From Table 1, we see that NM-ARNM for $m = 100$ could not solve problems **AKIVA** and **3PK**. Furthermore, from Table 2, NM-ARNM-MC for $m = 100$ could not solve problems **NONCVXUN** and **NONCVXU2**. Tables 3 and 4 show the dimension of x (n), the number of function evaluations (N_f), the number of iterations (N_{iter}), the factorization count (N_{fac}) and the function value (f) for each problem. From Figures 5 and 6, we see that each optimal value m^* for each m in $\{1, 2, 5, 10, 20, 50, 100\}$ is

- NM-ARNM: $m^* = 20$,
- NM-ARNM-MC: $m^* = 20$.

In the following, we compare ARNM with the proposed algorithms.

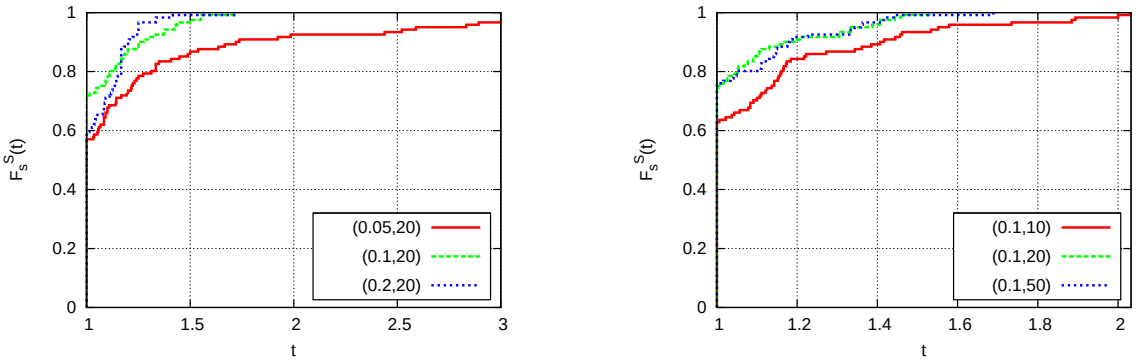

図 1: Influence of (γ_1, γ_2) for ARNM

4.3 Comparing NM-ARNM with ARNM

We show that the numerical results give an indication on how NM-ARNM improves the performance when compared to the conventional ARNM. Figures 7 and 8 show the performance profile for function evaluations and number of iterations. From these figures, we see that NM-ARNM performs better than ARNM, when we compare function evaluations or number of iterations. We see that NM-ARNM requires less iterations for some ill-conditioned problems. Therefore, we can confirm that Difficulty 1 of ARNM, described in Section 1, is overcome.

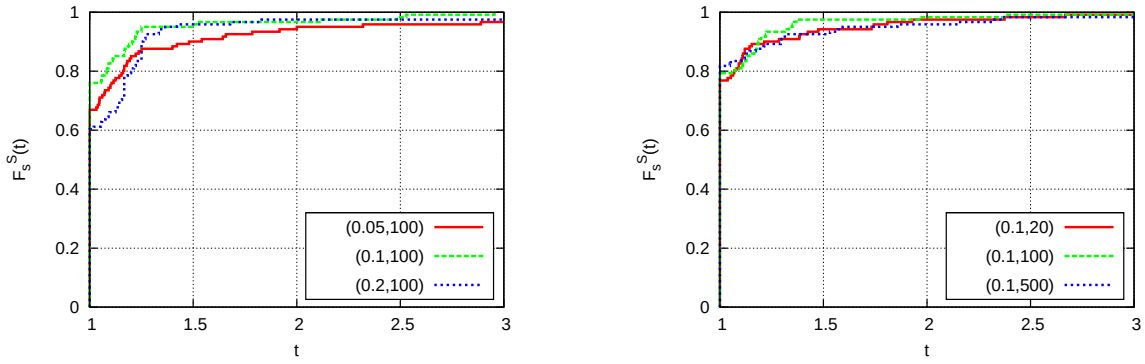

図 2: Influence of (γ_1, γ_2) for NM-ARNM

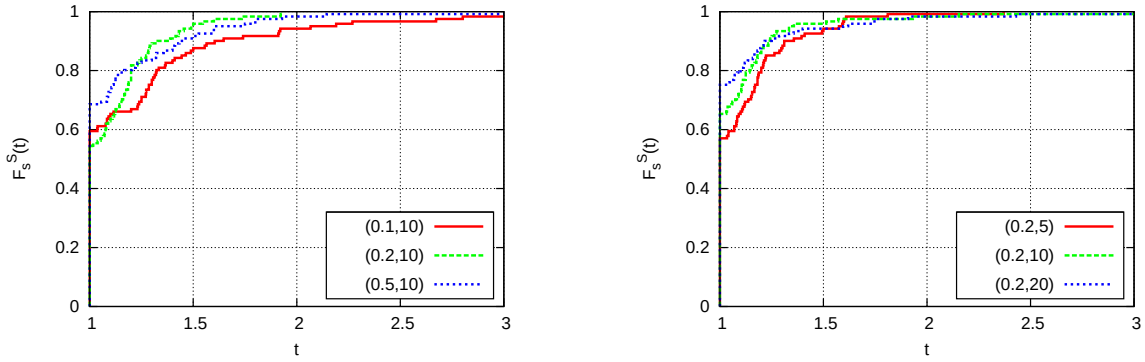

図 3: Influence of (γ_1, γ_2) for ARNM-MC

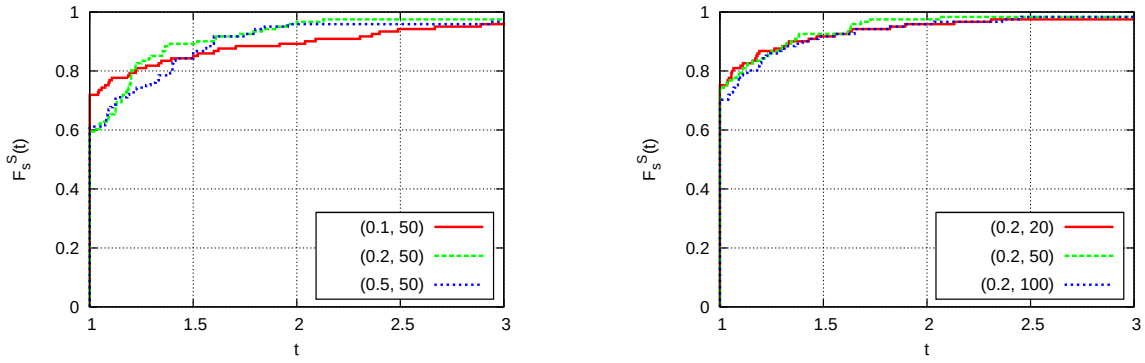

図 4: Influence of (γ_1, γ_2) for NM-ARNM-MC

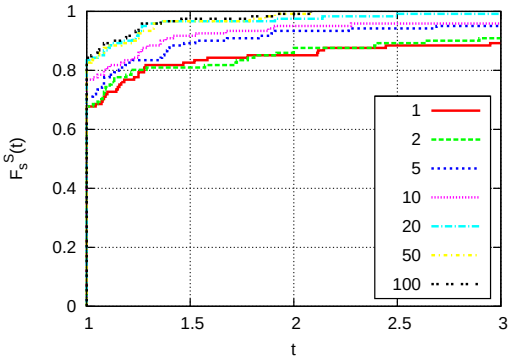

図 5: Influence of m for NM-ARNM

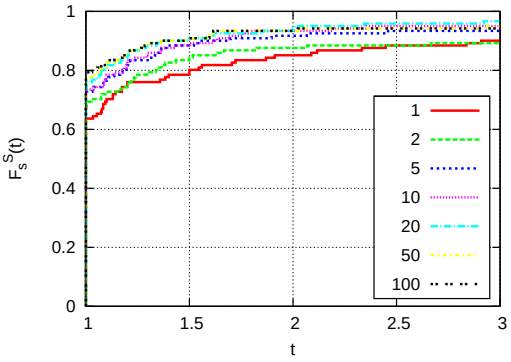

図 6: Influence of m for NM-ARNM-MC

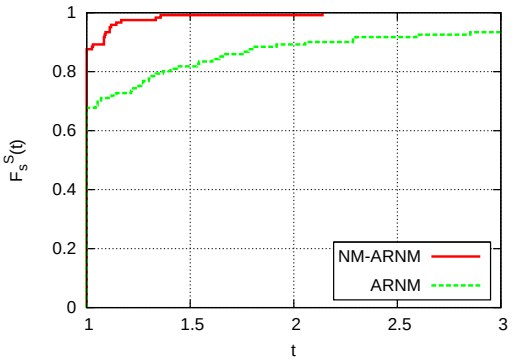

図 7: Function evaluation profile

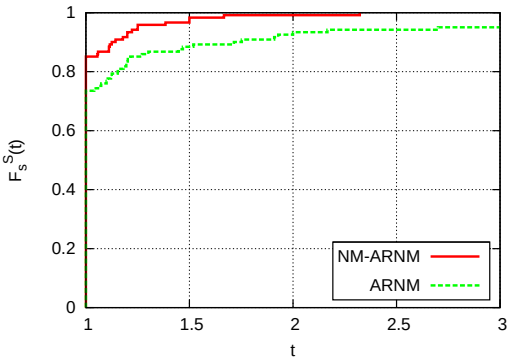

図 8: Number of iterations profile

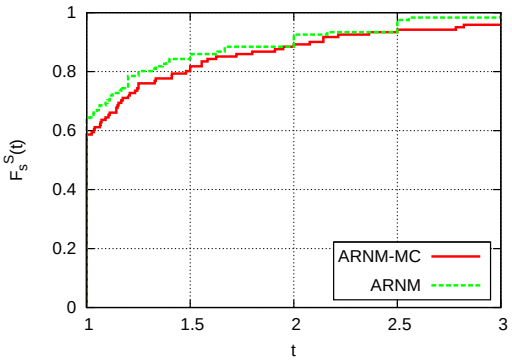

図 9: Function evaluation profile

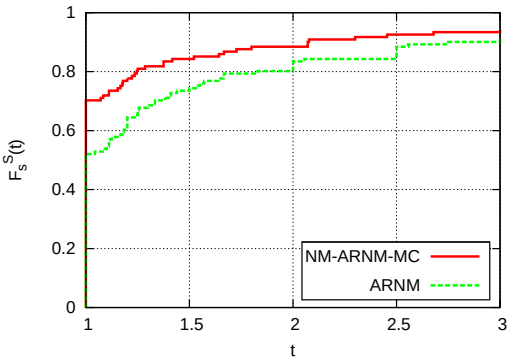

図 10: Function evaluation profile

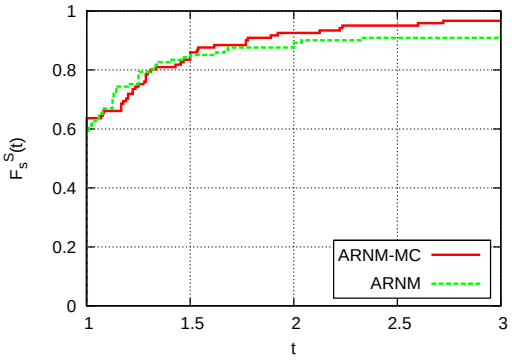

図 11: Number of iterations profile

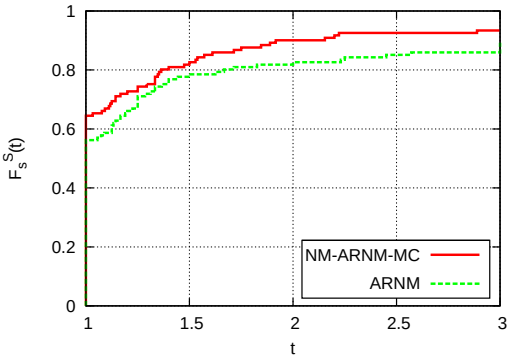

図 12: Number of iterations profile

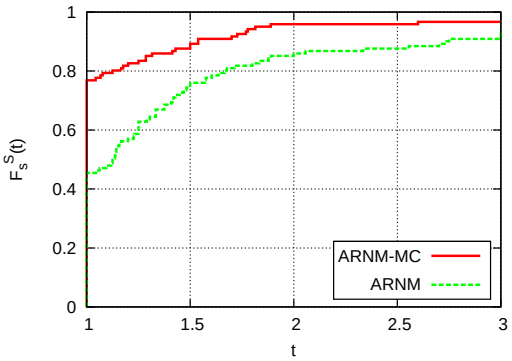

図 13: Factorization count profile

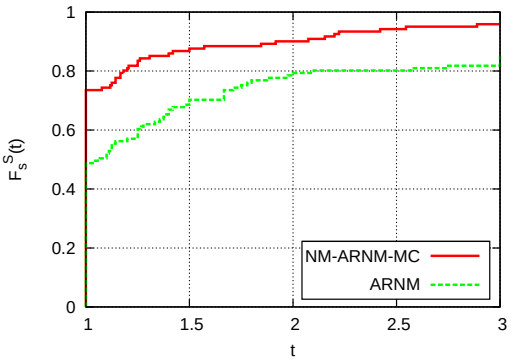

図 14: Factorization count profile

4.4 Comparing ARNM-MC with ARNM

Here, we investigate the efficiency of ARNM-MC when compared to the conventional ARNM. As performance measures, we use the function evaluations, the number of iterations and the factorization count. The factorization count is the number of factorizations to the format $\nabla^2 f(x_k) = L_k B_k L_k^T$ with modified Cholesky factorization in ARNM-MC. In ARNM, we consider the number of times to solve linear equations as the factorization count. Furthermore, we point out again that we do not need to factorize $\nabla^2 f(x_k)$ when we calculate the search direction by increasing ν. Indeed, $\nabla^2 f(x_k)$ has already been factorized to L_k and B_k. Figures 9, 11 and 13 show the performance profiles for the function evaluations, the number of iterations and the factorization count, respectively. Furthermore, Figures 9 and 11 show that the function evaluations and the number of iterations for ARNM-MC and ARNM are the almost same. For these reasons, we can confirm that there is no differences between $\lambda_{\min}\left(\nabla^2 f(x_k)\right)$ and the modify Cholesky factorization in order to determine $E_k(\nu)$. Consequently, we can overcome Difficulty 2 of ARNM. Furthermore, Figure 13 shows that ARNM-MC is superior to ARNM in terms of factorization count. Therefore, we can confirm that ARNM-MC can better calculate the search direction than ARNM when calculating the search direction again. Thus, ARNM-MC overcomes Difficulty 3 of ARNM.

4.5 Comparing NM-ARNM-MC with ARNM

Finally, we compare NM-ARNM-MC with ARNM. We use the function evaluations, the number of iterations and the factorization count as performance measures in the same way as in Section 2. From Figures 10 and 12, we confirm that NM-ARNM-MC can reduce function evaluations and number of iterations compared to ARNM. Furthermore, from Figure 14, we observe that the number of iterations is reduced.

Given these facts, we can say that NM-ARNM-MC is an efficient way to implement ARNM while overcoming the Difficulties 1, 2 and 3 of ARNM.

5 Concluding Remarks

In this paper, we proposed two efficient implementation ways to overcome the difficulties of ARNM. First, we proposed NM-ARNM which is implemented with a nonmonotone search technique in order to overcome Difficulty 1, refers to the fact that ARNM becomes too slow for some ill-conditioned problems. Then, we proposed ARNM-MC, which calculates the search direction by using the modified Cholesky factorization proposed by Cheng and Higham [3] in order to overcome Difficulty 2, which is that ARNM must calculate the minimum eigenvalue of $\nabla^2 f(x_k)$. It also overcomes Difficulty 3, which is that ARNM must solve the linear equation (1.3) again if the objective value $f(x_k + d_k(\nu_k))$ does not decrease sufficiently. Finally, we proposed NM-ARNM-MC which is a combination of the above two algorithms.

We have presented some numerical results and investigated the validity of our proposed algorithm when compared to the conventional ARNM. First, we confirmed that NM-ARNM can obtain an optimal solution with fewer function evaluations and number of iterations than ARNM. As a result, we could overcome Difficulty 1. Then, we compared ARNM-MC with ARNM. As a result, we confirmed that the function evaluations and iteration count of ARNM-MC were the same as these of the conventional ARNM. Finally, we analyzed numerical results of NM-ARNM-MC. As a result, we confirmed that we could compute more efficiently than the conventional ARNM. Thus, we could confirm that NM-ARNM-MC can overcome the difficulties of the conventional ARNM.

Although, the triangular matrix L_k given by the modified Cholesky factorization in ARNM-MC is bounded, it is not sparse in general. Thus we can not apply it for the large-scale problem. In the future, it is worth to develop a modified Cholesky factorization that would ensure the sparsity of $\nabla^2 f(x_k)$ and boundedness of L_k.

Acknowledgment

I would like to express my deepest gratitude to Associate Professor Nobuo Yamashita whose enormous support and insightful comments were invaluable during the course of my study. Special thanks also go to Professor Masao Fukushima and Assistant Professor Shunsuke Hayashi who gave me invaluable comments and warm encouragements. I would also like to express my gratitude to researcher Kenji Ueda, researcher Ellen Hidemi Fukuda, my labmate Ong Bun Theang and all members of Fukushima Laboratory for their moral support and warm encouragements. Finally, I really appreciate to my parents Mikiko Nishimori and Shoichi Nishimori for sincere support.

参考文献

[1] G. Ashcraft, R. G. Grimes, and J. G. Lewis, Accurate Symmetric Indefinite Linear Equation Solvers, SIAM Journal on Matrix Analysis and Applications, 20 (1998), pp. 513–561.

[2] D. P. Bertsekas, Nonlinear Programming, Athena Scientific, New York, 1995.

[3] S. H. Cheng and N. J. Higam, A Modified Cholesky Algorithm Based on a Symmetric Indefinite Factorization, SIAM Journal on Matrix Analysis and Applications, 19 (1998), pp. 1097–1110.

[4] A. R. Conn, N. I. M. Gould and P. L. Toint, Trust-Region Methods, SIAM, Philadelphia, 2000.

[5] N. Y. Deng, Y. Xiao and F. J. Zhou, Nonmonotonic Trust Region Algorithm, Journal of Optimization Theory and Applications, 90 (1993), pp. 259–285.

[6] E. D. Dolan and J. J. Moré, Benchmarking Optimization Software with Performance Profiles, Mathematical Programming, 91 (2002), pp. 201–213.

[7] I. S. Duff, MA57–A Code for the Solution of Sparse Symmetric Definite and Indefinite Systems, ACM Transactions on Mathematical Software, 30 (2004), pp. 118–144.

[8] H. Fang and D. P. O'leary, Modified Cholesky Algorithms: A Catalog with New Approaches, Mathematical Programming, 115 (2006), pp. 319–349.

[9] P. E. Gill, W. Murray and M. H. Wright, Practical Optimization, Academic Press, London, 1981.

[10] N. I. M. Gould, D. Oreban and P. L. Toint, CUTEr (and SifDec), a Constrained and Unconstrained Testing Environment, Revised, ACM Transactions on Mathematical Software, 29 (2003), pp. 373–394.

[11] J. Nocedal and S. J. Wright, Numerical Optimization, Springer-Verlag New York, Inc., New York, 1999.

[12] R. B. Schnable and E. Eskow, A Revised Modified Cholesky Factorization Algorithm, SIAM Journal on Optimization, 9 (1999), pp. 1135–1148.

[13] P. L. Toint, Non-monotone Trust-Region Algorithms for Nonlinear Optimization Subject to Convex Constraints, Mathematical Programming, 77 (1997), pp. 69–94.

[14] K. Ueda, Studies on Regularized Newton-Type Methods for Unconstrained Minimization Problems and Their Global Complexity Bounds, Doctor Thesis, Kyoto University, 2011.

[15] K. Ueda and N. Yamashita, A Regularized Newton Method without Line Search for Unconstrained Optimization, Technical Report 2009-007, Department of Applied Mathematics and Physics, Graduate School of Informatics, Kyoto University, 2009.

[16] G. L. Yuan, C. L. Chen and Z. X. Wei, A Nonmonotone Adaptive Trust-Region Algorithm for Symmetric Nonlinear Equations, 230 (2010), pp. 44–58.

[17] J. Z. Zhang and L. H. Chen, Nonmonotone Levenberg–Marquardt Algorithms and Their Convergence Analysis, Journal of Optimization Theory and Applications, 92 (1997), pp. 393–418.

表 1: Influence of m for NM-ARNM

Name	n	$m=1$	$m=2$	$m=5$	$m=10$	$m=20$	$m=50$	$m=100$
3PK	30	6	6	6	6	6	6	6
AKIVA	2	7	7	7	7	7	7	7
ALLINITU	4	9	9	9	9	9	9	9
ARGLINA	200	5	5	5	5	5	5	5
ARWHEAD	100	6	6	6	6	6	6	6
BARD	3	8	8	8	8	8	8	8
BDQRTIC	100	10	10	10	10	10	10	10
BEALE	2	9	10	10	10	10	10	10
BIGGS6	6	102	107	90	83	83	88	88
BOX3	3	8	8	8	8	8	8	8
BRKMCC	2	4	4	4	4	4	4	4
BROWNAL	200	5	5	5	5	5	5	5
BROWNBS	2	12	13	13	13	13	13	13
BROWNDEN	4	9	9	9	9	9	9	9
BROYDN7D	100	29	29	29	29	29	29	29
BRYBND	100	12	13	13	13	13	13	13
CHNROSNB	50	143	128	64	58	53	42	42
CLIFF	2	28	28	28	28	28	28	28
COSINE	100	10	10	10	10	10	10	10
CRAGGLVY	100	14	14	14	14	14	14	14
CUBE	2	70	43	10	10	10	10	10
CURLY10	100	23	18	18	18	18	18	18
CURLY20	100	22	19	19	19	19	19	19
DECONVU	61	19	23	12	12	12	12	12
DENSCHNA	2	6	6	6	6	6	6	6
DENSCHNB	2	8	8	8	8	8	8	8
DENSCHNC	2	11	11	11	11	11	11	11
DENSCHND	3	33	34	34	34	34	34	34
DENSCHNE	3	14	14	14	14	14	14	14
DENSCHNF	2	7	7	7	7	7	7	7
DIXMAANA	300	9	11	11	11	11	11	11
DIXMAANB	300	15	17	17	19	19	19	19
DIXMAANC	300	10	10	10	10	10	10	10
DIXMAAND	300	10	10	10	10	10	10	10
DIXMAANE	300	9	9	9	9	9	9	9
DIXMAANF	300	13	13	13	13	13	13	13
DIXMAANG	300	14	14	14	14	14	14	14
DIXMAANH	300	15	15	15	15	15	15	15
DIXMAANI	300	10	10	10	10	10	10	10
DIXMAANJ	300	19	20	20	20	20	20	20

表 1: Influence of m for NM-ARNM

Name	n	$m=1$	$m=2$	$m=5$	$m=10$	$m=20$	$m=50$	$m=100$
DIXMAANK	15	14	22	19	19	19	19	19
DIXMAANL	300	24	25	23	23	23	23	23
DIXON3DQ	100	5	5	5	5	5	5	5
DQDRTIC	100	4	4	4	4	4	4	4
EDENSCH	36	13	13	13	13	13	13	13
ENGVAL1	100	8	8	8	8	8	8	8
ENGVAL2	3	18	18	18	18	18	18	18
ERRINROS	50	92	100	21	21	21	21	21
EXPFIT	2	14	12	12	12	12	12	12
FLETCBV2	100	2	2	2	2	2	2	2
FREUROTH	100	13	12	12	12	12	12	12
GENROSE	100	131	124	121	112	102	102	102
GROWTHLS	3	171	183	111	75	49	49	49
GULF	3	30	24	24	28	28	28	28
HAIRY	2	60	94	86	107	47	84	84
HATFLDD	3	19	19	19	19	19	19	19
HATFLDE	3	19	19	19	19	19	19	19
HEART6LS	6	3353	3220	2288	2948	2166	417	224
HEART8LS	8	171	168	156	97	64	58	58
HELIX	3	11	11	11	11	11	11	11
HIELOW	3	10	8	8	8	8	8	8
HILBERTA	2	5	5	5	5	5	5	5
HILBERTB	10	4	4	4	4	4	4	4
HIMMELBB	2	13	13	13	13	13	13	13
HIMMELBF	4	160	160	151	90	90	90	90
HIMMELBG	2	8	7	7	7	7	7	7
HIMMELBH	2	8	8	8	8	8	8	8
HUMPS	2	256	305	297	323	331	339	373
KOWOSB	4	15	14	14	14	14	14	14
LIARWHD	100	11	11	11	11	11	11	11
LOGHAIRY	2	2929	3334	1924	2602	1386	1835	1650
MARATOSB	2	2088	1999	1221	952	215	113	86
MEXHAT	2	38	28	25	25	25	25	25
MOREBV	100	3	3	3	3	3	3	3
NONCVXU2	100	36	36	39	51	77	75	75
NONCVXUN	100	28	28	28	28	28	28	28
NONDIA	100	11	10	10	10	10	10	10
OSBORNEA	5	93	225	29	27	27	27	27
OSBORNEB	11	22	17	18	15	15	15	15
PALMER1C	8	7	7	7	7	7	7	7

表 1: Influence of m for NM-ARNM

Name	n	$m=1$	$m=2$	$m=5$	$m=10$	$m=20$	$m=50$	$m=100$
PALMER1D	7	6	6	6	6	6	6	6
PALMER2C	8	6	6	6	6	6	6	6
PALMER3C	8	6	6	6	6	6	6	6
PALMER4C	8	6	6	6	6	6	6	6
PALMER5C	6	5	5	5	5	5	5	5
PALMER6C	8	6	6	6	6	6	6	6
PALMER7C	8	6	6	6	6	6	6	6
PALMER8C	8	6	6	6	6	6	6	6
PFIT1LS	3	1799	1855	1532	1329	61	61	61
PFIT2LS	3	497	699	915	769	46	46	46
PFIT3LS	3	837	742	290	38	47	47	47
PFIT4LS	3	1733	1783	1688	1465	50	50	50
POWELLSG	4	16	16	16	16	16	16	16
ROSENBR	2	44	43	25	18	18	18	18
QUARTC	100	25	25	25	25	25	25	25
S308	2	12	11	11	11	11	11	11
SBRYBND	100	25	25	25	25	25	25	25
SCHMVETT	100	5	5	5	5	5	5	5
SINEVAL	2	108	88	75	66	57	51	51
SINQUAD	100	14	14	14	14	14	14	14
SISSER	2	13	13	13	13	13	13	13
SNAIL	2	191	161	93	105	89	91	91
SPARSINE	100	7	7	7	7	7	7	7
SPARSQUR	100	17	17	17	17	17	17	17
SPMSRTLS	100	11	11	11	11	11	11	11
SROSENBR	100	8	8	8	8	8	8	8
TESTQUAD	1000	5	5	5	5	5	5	5
TOINTGOR	50	6	6	6	6	6	6	6
TOINTGSS	100	6	6	6	6	6	6	6
TOINTPSP	50	26	24	30	22	22	22	22
TOINTQOR	50	5	5	5	5	5	5	5
TQUARTIC	100	14	14	13	13	13	13	13
TRIDIA	100	4	4	4	4	4	4	4
VARDIM	200	30	30	30	30	30	30	30
VAREIGVL	50	11	22	21	21	21	21	21
VIBRBEAM	8	32	34	37	40	40	40	40
WATSON	12	13	13	13	13	13	13	13
WOODS	4	97	74	53	38	28	28	28
YFITU	3	115	55	44	32	32	32	32
ZANGWIL2	2	4	4	4	4	4	4	4

表 2: Influence of m for NM-ARNM-MC

Name	n	$m=1$	$m=2$	$m=5$	$m=10$	$m=20$	$m=50$	$m=100$
3PK	30	6	6	6	6	6	6	6
AKIVA	2	7	7	7	7	7	7	7
ALLINITU	4	9	9	9	9	9	9	9
ARGLINA	200	2	2	2	2	2	2	2
ARWHEAD	100	6	6	6	6	6	6	6
BARD	3	10	10	10	10	10	10	10
BDQRTIC	100	10	10	10	10	10	10	10
BEALE	2	11	18	15	15	15	15	15
BIGGS6	6	96	120	33	47	40	40	40
BOX3	3	8	8	8	8	8	8	8
BRKMCC	2	3	3	3	3	3	3	3
BROWNAL	200	6	6	6	6	6	6	6
BROWNBS	2	10	12	12	12	12	12	12
BROWNDEN	4	9	9	9	9	9	9	9
BROYDN7D	100	30	30	20	22	22	22	22
BRYBND	100	11	11	11	11	11	11	11
CHNROSNB	50	86	62	70	55	49	46	46
CLIFF	2	28	28	28	28	28	28	28
COSINE	100	9	9	9	9	9	9	9
CRAGGLVY	100	15	15	15	15	15	15	15
CUBE	2	51	40	31	15	15	15	15
CURLY10	100	26	24	24	24	24	24	24
CURLY20	100	26	23	23	23	23	23	23
DECONVU	61	9	9	9	9	9	9	9
DENSCHNA	2	6	6	6	6	6	6	6
DENSCHNB	2	10	11	11	11	11	11	11
DENSCHNC	2	11	11	11	11	11	11	11
DENSCHND	3	33	33	33	33	33	33	33
DENSCHNE	3	27	27	27	27	29	29	29
DENSCHNF	2	7	7	7	7	7	7	7
DIXMAANA	300	8	7	7	7	7	7	7
DIXMAANB	300	27	27	23	23	23	23	23
DIXMAANC	300	27	23	23	23	23	23	23
DIXMAAND	300	24	24	24	31	31	31	31
DIXMAANE	300	8	8	8	8	8	8	8
DIXMAANF	300	29	29	29	27	27	27	27
DIXMAANG	300	30	29	29	29	29	29	29
DIXMAANH	300	27	27	27	27	27	27	27
DIXMAANI	300	9	9	9	9	9	9	9
DIXMAANJ	300	21	21	21	21	21	21	21

表 2: Influence of m for NM-ARNM-MC

Name	n	$m=1$	$m=2$	$m=5$	$m=10$	$m=20$	$m=50$	$m=100$
DIXMAANK	15	13	13	17	21	21	21	21
DIXMAANL	300	29	29	33	33	33	33	33
DIXON3DQ	100	2	2	2	2	2	2	2
DQDRTIC	100	2	2	2	2	2	2	2
EDENSCH	36	13	13	13	13	13	13	13
ENGVAL1	100	8	8	8	8	8	8	8
ENGVAL2	3	24	20	20	20	20	20	20
ERRINROS	50	142	157	30	22	22	22	22
EXPFIT	2	11	11	11	11	11	11	11
FLETCBV2	100	2	2	2	2	2	2	2
FREUROTH	100	20	19	19	19	19	19	19
GENROSE	100	191	155	134	122	120	120	112
GROWTHLS	3	170	117	91	81	37	37	37
GULF	3	45	43	30	36	38	38	38
HAIRY	2	91	43	52	47	86	87	87
HATFLDD	3	16	16	16	16	16	16	16
HATFLDE	3	17	17	17	17	17	17	17
HEART6LS	6	3092	15	15	15	15	15	15
HEART8LS	8	2413	2185	2140	1433	141	183	140
HELIX	3	18	21	24	27	27	27	27
HIELOW	3	6	6	6	6	6	6	6
HILBERTA	2	3	3	3	3	3	3	3
HILBERTB	10	2	2	2	2	2	2	2
HIMMELBB	2	15	15	15	15	15	15	15
HIMMELBF	4	230	231	231	231	231	231	231
HIMMELBG	2	5	5	5	5	5	5	5
HIMMELBH	2	9	9	9	9	9	9	9
HUMPS	2	430	575	446	532	502	502	552
KOWOSB	4	14	14	13	13	13	13	13
LIARWHD	100	11	11	11	11	11	11	11
LOGHAIRY	2	477	1079	903	743	152	152	152
MARATOSB	2	2035	1598	1116	826	126	94	94
MEXHAT	2	39	31	25	25	25	25	25
MOREBV	100	3	3	3	3	3	3	3
NONCVXU2	100	501	478	262	471	1127	4349	13809
NONCVXUN	100	208	184	204	433	962	4782	13752
NONDIA	100	12	10	10	10	10	10	10
OSBORNEA	5	94	94	34	28	28	28	28
OSBORNEB	11	71	41	34	37	35	35	35
PALMER1C	8	6	6	6	6	6	6	6

表 2: Influence of m for NM-ARNM-MC

Name	n	$m=1$	$m=2$	$m=5$	$m=10$	$m=20$	$m=50$	$m=100$
PALMER1D	7	4	4	4	4	4	4	4
PALMER2C	8	5	5	5	5	5	5	5
PALMER3C	8	5	5	5	5	5	5	5
PALMER4C	8	5	5	5	5	5	5	5
PALMER5C	6	2	2	2	2	2	2	2
PALMER6C	8	5	5	5	5	5	5	5
PALMER7C	8	6	6	6	6	6	6	6
PALMER8C	8	6	6	6	6	6	6	6
PFIT1LS	3	1699	1495	1500	1141	87	55	55
PFIT2LS	3	591	639	596	33	33	33	33
PFIT3LS	3	1045	1183	899	742	40	40	40
PFIT4LS	3	1159	1019	725	44	328	71	71
POWELLSG	4	16	16	16	16	16	16	16
ROSENBR	2	31	30	18	18	18	18	18
QUARTC	100	25	25	25	25	25	25	25
S308	2	14	10	10	10	10	10	10
SBRYBND	100	51	50	33	33	33	33	33
SCHMVETT	100	4	4	4	4	4	4	4
SINEVAL	2	83	74	65	60	60	60	60
SINQUAD	100	22	20	20	20	20	20	20
SISSER	2	13	13	13	13	13	13	13
SNAIL	2	148	133	127	113	109	174	174
SPARSINE	100	116	117	161	158	233	430	744
SPARSQUR	100	17	17	17	17	17	17	17
SPMSRTLS	100	38	39	59	45	111	111	221
SROSENBR	100	14	6	6	6	6	6	6
TESTQUAD	1000	2	2	2	2	2	2	2
TOINTGOR	50	7	7	7	7	7	7	7
TOINTGSS	100	2	2	2	2	2	2	2
TOINTPSP	50	35	32	33	54	75	126	149
TOINTQOR	50	2	2	2	2	2	2	2
TQUARTIC	100	39	36	36	36	38	38	38
TRIDIA	100	2	2	2	2	2	2	2
VARDIM	200	30	30	30	30	30	30	30
VAREIGVL	50	17	22	20	20	20	20	20
VIBRBEAM	8	63	86	107	103	117	657	449
WATSON	12	13	13	13	13	13	13	13
WOODS	4	71	60	55	51	35	29	29
YFITU	3	105	126	50	48	37	37	37
ZANGWIL2	2	2	2	2	2	2	2	2

表 3: Comparison with ARNM and NM-ARNM

		ARNM				NM-ARNM			
Name	n	N_f	N_{iter}	N_{fac}	f	N_f	N_{iter}	N_{fac}	f
3PK	30	6	5	5	1.72E+00	6	5	5	1.72E+00
AKIVA	2	7	6	6	6.17E+00	7	6	6	6.17E+00
ALLINITU	4	9	8	8	5.74E+00	9	8	8	5.74E+00
ARGLINA	200	5	4	4	2.00E+02	5	4	4	2.00E+02
ARWHEAD	100	6	5	5	8.79E-14	6	5	5	8.79E-14
BARD	3	8	7	7	8.21E-03	8	7	7	8.21E-03
BDQRTIC	100	10	9	9	3.79E+02	10	9	9	3.79E+02
BEALE	2	9	7	8	2.68E-13	10	8	9	6.04E-13
BIGGS6	6	128	94	127	7.35E-07	83	74	82	3.78E-09
BOX3	3	8	7	7	3.49E-12	8	7	7	3.49E-12
BRKMCC	2	4	3	3	1.69E-01	4	3	3	1.69E-01
BROWNAL	200	5	4	4	4.64E-14	5	4	4	4.64E-14
BROWNBS	2	12	9	11	6.74E-16	13	10	12	1.97E-31
BROWNDEN	4	9	8	8	8.58E+04	9	8	8	8.58E+04
BROYDN7D	100	31	26	30	3.28E+01	29	25	28	3.28E+01
BRYBND	100	12	9	11	3.65E-22	13	10	12	3.38E-22
CHNROSNB	50	76	49	75	2.17E-13	53	41	52	2.28E-13
CLIFF	2	28	27	27	2.00E-01	28	27	27	2.00E-01
COSINE	100	10	9	9	-9.90E+01	10	9	9	-9.90E+01
CRAGGLVY	100	14	13	13	3.23E+01	14	13	13	3.23E+01
CUBE	2	45	29	44	1.85E-17	10	9	9	1.16E-18
CURLY10	100	24	17	23	-1.00E+04	18	17	17	-1.00E+04
CURLY20	100	24	17	23	-1.00E+04	19	18	18	-1.00E+04
DECONVU	61	23	13	22	2.02E-07	12	11	11	1.40E-07
DENSCHNA	2	6	5	5	1.34E-12	6	5	5	1.34E-12
DENSCHNB	2	8	6	7	2.15E-13	8	6	7	4.05E-19
DENSCHNC	2	11	10	10	4.87E-20	11	10	10	4.87E-20
DENSCHND	3	33	28	32	1.82E-08	34	33	33	1.14E-08
DENSCHNE	3	14	13	13	3.94E-13	14	13	13	3.94E-13
DENSCHNF	2	7	6	6	6.85E-22	7	6	6	6.85E-22
DIXMAANA	300	11	8	10	1.00E+00	11	8	10	1.00E+00
DIXMAANB	300	14	9	13	1.00E+00	19	15	18	1.00E+00
DIXMAANC	300	10	9	9	1.00E+00	10	9	9	1.00E+00
DIXMAAND	300	10	9	9	1.00E+00	10	9	9	1.00E+00
DIXMAANE	300	9	8	8	1.00E+00	9	8	8	1.00E+00
DIXMAANF	300	13	12	12	1.00E+00	13	12	12	1.00E+00
DIXMAANG	300	14	13	13	1.00E+00	14	13	13	1.00E+00
DIXMAANH	300	15	14	14	1.00E+00	15	14	14	1.00E+00
DIXMAANI	300	10	9	9	1.00E+00	10	9	9	1.00E+00

表 3: Comparison with ARNM and NM-ARNM

		ARNM				NM-ARNM			
Name	n	N_f	N_{iter}	N_{fac}	f	N_f	N_{iter}	N_{fac}	f
DIXMAANJ	300	26	18	25	1.00E+00	20	19	19	1.00E+00
DIXMAANK	15	17	13	16	1.00E+00	19	18	18	1.00E+00
DIXMAANL	300	24	18	23	1.00E+00	23	22	22	1.00E+00
DIXON3DQ	100	5	4	4	4.14E-11	5	4	4	4.14E-11
DQDRTIC	100	4	3	3	4.72E-12	4	3	3	4.72E-12
EDENSCH	36	13	12	12	2.19E+02	13	12	12	2.19E+02
ENGVAL1	100	8	7	7	1.09E+02	8	7	7	1.09E+02
ENGVAL2	3	18	17	17	1.28E-14	18	17	17	1.28E-14
ERRINROS	50	90	54	89	3.99E+01	21	20	20	4.04E+01
EXPFIT	2	9	6	8	2.41E-01	12	9	11	2.41E-01
FLETCBV2	100	2	1	1	-5.14E-01	2	1	1	-5.14E-01
FREUROTH	100	11	8	10	1.20E+04	12	9	11	1.20E+04
GENROSE	100	134	87	133	1.00E+00	102	81	101	1.00E+00
GROWTHLS	3	127	81	126	1.00E+00	49	42	48	1.00E+00
GULF	3	45	30	44	7.93E-16	28	26	27	9.98E-19
HAIRY	2	108	76	107	2.00E+01	47	38	46	2.00E+01
HATFLDD	3	20	18	19	6.62E-08	19	17	18	6.62E-08
HATFLDE	3	20	17	19	5.12E-07	19	17	18	5.12E-07
HEART6LS	6	3043	1753	3042	1.03E-07	2166	1458	2165	1.40E-24
HEART8LS	8	195	117	194	1.42E-18	64	54	63	5.04E-16
HELIX	3	11	10	10	3.74E-23	11	10	10	3.74E-23
HIELOW	3	10	7	9	8.74E+02	8	7	7	8.74E+02
HILBERTA	2	5	4	4	1.42E-25	5	4	4	1.42E-25
HILBERTB	10	4	3	3	7.22E-15	4	3	3	7.22E-15
HIMMELBB	2	13	12	12	2.00E-18	13	12	12	2.00E-18
HIMMELBF	4	163	156	162	3.19E+02	90	89	89	3.19E+02
HIMMELBG	2	7	5	6	5.66E-15	7	6	6	3.67E-12
HIMMELBH	2	7	5	6	-1.00E+00	8	6	7	-1.00E+00
HUMPS	2	403	360	402	3.40E-10	331	321	330	1.03E-13
KOWOSB	4	12	8	11	3.08E-04	14	10	13	3.08E-04
LIARWHD	100	11	10	10	1.52E-13	11	10	10	1.52E-13
LOGHAIRY	2	2298	1709	2297	1.82E-01	1386	1313	1385	1.82E-01
MARATOSB	2	1555	929	1554	-1.00E+00	215	159	214	-1.00E+00
MEXHAT	2	41	28	40	-4.00E-02	25	24	24	-4.00E-02
MOREBV	100	3	2	2	8.67E-10	3	2	2	8.67E-10
NONCVXU2	100	36	31	35	2.32E+02	77	72	76	2.34E+02
NONCVXUN	100	28	27	27	2.34E+02	28	27	27	2.34E+02
NONDIA	100	9	6	8	3.22E-18	10	9	9	1.00E-17
OSBORNEA	5	77	42	76	5.52E-05	27	22	26	5.46E-05

表 3: Comparison with ARNM and NM-ARNM

		ARNM				NM-ARNM			
Name	n	N_f	N_{iter}	N_{fac}	f	N_f	N_{iter}	N_{fac}	f
OSBORNEB	11	23	17	22	4.01E-02	15	14	14	4.01E-02
PALMER1C	8	7	6	6	9.76E-02	7	6	6	9.76E-02
PALMER1D	7	6	5	5	6.53E-01	6	5	5	6.53E-01
PALMER2C	8	6	5	5	1.44E-02	6	5	5	1.44E-02
PALMER3C	8	6	5	5	1.95E-02	6	5	5	1.95E-02
PALMER4C	8	6	5	5	5.03E-02	6	5	5	5.03E-02
PALMER5C	6	5	4	4	2.13E+00	5	4	4	2.13E+00
PALMER6C	8	6	5	5	1.64E-02	6	5	5	1.64E-02
PALMER7C	8	6	5	5	6.02E-01	6	5	5	6.02E-01
PALMER8C	8	6	5	5	1.60E-01	6	5	5	1.60E-01
PFIT1LS	3	1113	636	1112	4.58E-07	61	53	60	5.54E-10
PFIT2LS	3	371	212	370	3.59E-08	46	41	45	2.52E-15
PFIT3LS	3	461	268	460	8.10E-09	47	41	46	2.99E-21
PFIT4LS	3	991	571	990	1.94E-16	50	45	49	1.48E-21
POWELLSG	4	16	15	15	4.43E-09	16	15	15	4.43E-09
ROSENBR	2	37	24	36	2.51E-23	18	14	17	4.74E-12
QUARTC	100	25	24	24	2.31E-08	25	24	24	2.31E-08
S308	2	11	8	10	7.73E-01	11	10	10	7.73E-01
SBRYBND	100	28	18	27	1.36E-20	25	18	24	1.49E-12
SCHMVETT	100	5	4	4	-2.94E+02	5	4	4	-2.94E+02
SINEVAL	2	78	49	77	2.09E-28	57	41	56	4.23E-30
SINQUAD	100	17	11	16	-4.01E+03	14	10	13	-4.01E+03
SISSER	2	13	12	12	1.14E-08	13	12	12	1.14E-08
SNAIL	2	157	94	156	1.30E-16	89	64	88	2.17E-11
SPARSINE	100	7	6	6	1.04E-21	7	6	6	1.04E-21
SPARSQUR	100	17	16	16	7.66E-09	17	16	16	7.66E-09
SPMSRTLS	100	11	10	10	4.05E-11	11	10	10	3.50E-11
SROSENBR	100	8	7	7	7.65E-15	8	7	7	3.27E-15
TESTQUAD	1000	5	4	4	1.01E-29	5	4	4	1.01E-29
TOINTGOR	50	6	5	5	1.37E+03	6	5	5	1.37E+03
TOINTGSS	100	6	5	5	1.01E+01	6	5	5	1.01E+01
TOINTPSP	50	28	18	27	2.26E+02	22	16	21	2.26E+02
TOINTQOR	50	5	4	4	1.18E+03	5	4	4	1.18E+03
TQUARTIC	100	12	9	11	3.22E-22	13	9	12	7.80E-15
TRIDIA	100	4	3	3	2.16E-14	4	3	3	2.16E-14
VARDIM	200	30	29	29	2.91E-24	30	29	29	2.91E-24
VAREIGVL	50	24	21	23	1.23E-15	21	19	20	1.73E-15
VIBRBEAM	8	39	26	38	1.56E-01	40	29	39	3.32E-01
WATSON	12	13	12	12	1.30E-07	13	12	12	1.30E-07

表 3: Comparison with ARNM and NM-ARNM

		ARNM				NM-ARNM			
Name	n	N_f	N_{iter}	N_{fac}	f	N_f	N_{iter}	N_{fac}	f
WOODS	4	64	41	63	5.36E-13	28	27	27	1.14E-14
YFITU	3	57	42	56	6.67E-13	32	29	31	6.82E-13
ZANGWIL2	2	4	3	3	-1.82E+01	4	3	3	-1.82E+01

表 4: Comparison with ARNM-MC and NM-ARNM-MC

		ARNM-MC				NM-ARNM-MC			
Name	n	N_f	N_{iter}	N_{fac}	f	N_f	N_{iter}	N_{fac}	f
3PK	30	6	5	5	1.72E+00	6	5	5	1.72E+00
AKIVA	2	7	6	6	6.17E+00	7	6	6	6.17E+00
ALLINITU	4	9	8	8	5.74E+00	9	8	8	5.74E+00
ARGLINA	200	2	1	1	2.00E+02	2	1	1	2.00E+02
ARWHEAD	100	6	5	5	6.59E-14	6	5	5	6.59E-14
BARD	3	13	10	10	8.21E-03	10	8	8	8.21E-03
BDQRTIC	100	10	9	9	3.79E+02	10	9	9	3.79E+02
BEALE	2	10	7	7	1.54E-12	15	12	12	6.54E-15
BIGGS6	6	85	47	47	1.91E-06	40	31	31	1.65E-05
BOX3	3	8	7	7	4.62E-13	8	7	7	4.62E-13
BRKMCC	2	3	2	2	1.69E-01	3	2	2	1.69E-01
BROWNAL	200	6	5	5	1.12E-18	6	5	5	1.12E-18
BROWNBS	2	11	8	8	0.00E+00	12	10	10	0.00E+00
BROWNDEN	4	9	8	8	8.58E+04	9	8	8	8.58E+04
BROYDN7D	100	33	23	23	4.90E+01	22	18	18	4.88E+01
BRYBND	100	12	8	8	1.31E-16	11	8	8	1.73E-17
CHNROSNB	50	84	53	53	5.98E-16	49	40	40	4.50E-13
CLIFF	2	28	27	27	2.00E-01	28	27	27	2.00E-01
COSINE	100	9	8	8	-9.90E+01	9	8	8	-9.90E+01
CRAGGLVY	100	15	14	14	3.23E+01	15	14	14	3.23E+01
CUBE	2	48	31	31	1.72E-15	15	13	13	3.73E-18
CURLY10	100	30	22	22	-1.00E+04	24	23	23	-1.00E+04
CURLY20	100	29	21	21	-1.00E+04	23	22	22	-1.00E+04
DECONVU	61	9	8	8	1.20E-13	9	8	8	1.20E-13
DENSCHNA	2	6	5	5	2.21E-12	6	5	5	2.21E-12
DENSCHNB	2	10	7	7	1.37E-14	11	8	8	1.04E-13
DENSCHNC	2	11	10	10	2.18E-20	11	10	10	2.18E-20
DENSCHND	3	33	26	26	3.51E-08	33	32	32	5.64E-08
DENSCHNE	3	31	23	23	8.17E-12	29	28	28	3.85E-16
DENSCHNF	2	7	6	6	6.51E-22	7	6	6	6.51E-22
DIXMAANA	300	10	7	7	1.00E+00	7	6	6	1.00E+00
DIXMAANB	300	30	20	20	1.00E+00	23	17	17	1.00E+00
DIXMAANC	300	25	17	17	1.00E+00	23	20	20	1.00E+00
DIXMAAND	300	20	16	16	1.00E+00	31	26	26	1.00E+00
DIXMAANE	300	8	7	7	1.00E+00	8	7	7	1.00E+00
DIXMAANF	300	27	18	18	1.00E+00	27	23	23	1.00E+00
DIXMAANG	300	30	20	20	1.00E+00	29	24	24	1.00E+00
DIXMAANH	300	27	18	18	1.00E+00	27	22	22	1.00E+00
DIXMAANI	300	9	8	8	1.00E+00	9	8	8	1.00E+00

表 4: Comparison with ARNM-MC and NM-ARNM-MC

		ARNM-MC				NM-ARNM-MC			
Name	n	N_f	N_{iter}	N_{fac}	f	N_f	N_{iter}	N_{fac}	f
DIXMAANJ	300	27	22	22	1.00E+00	21	18	18	1.00E+00
DIXMAANK	15	16	12	12	1.00E+00	21	20	20	1.00E+00
DIXMAANL	300	36	24	24	1.00E+00	33	29	29	1.00E+00
DIXON3DQ	100	2	1	1	0.00E+00	2	1	1	0.00E+00
DQDRTIC	100	2	1	1	0.00E+00	2	1	1	0.00E+00
EDENSCH	36	13	12	12	2.19E+02	13	12	12	2.19E+02
ENGVAL1	100	8	7	7	1.09E+02	8	7	7	1.09E+02
ENGVAL2	3	28	20	20	7.78E-25	20	19	19	6.60E-19
ERRINROS	50	112	69	69	3.99E+01	22	21	21	3.99E+01
EXPFIT	2	14	9	9	2.41E-01	11	8	8	2.41E-01
FLETCBV2	100	2	1	1	-5.14E-01	2	1	1	-5.14E-01
FREUROTH	100	26	17	17	1.20E+04	19	14	14	1.20E+04
GENROSE	100	178	114	114	1.00E+00	120	95	95	1.00E+00
GROWTHLS	3	120	77	77	1.00E+00	37	36	36	1.00E+00
GULF	3	44	28	28	4.65E-08	38	31	31	5.28E-11
HAIRY	2	79	52	52	2.00E+01	86	72	72	2.00E+01
HATFLDD	3	20	17	17	6.62E-08	16	14	14	6.62E-08
HATFLDE	3	16	13	13	5.12E-07	17	15	15	5.12E-07
HEART6LS	6	22	14	14	4.94E+01	15	14	14	4.94E+01
HEART8LS	8	336	207	207	2.56E-23	141	109	109	1.09E-19
HELIX	3	21	15	15	1.21E-21	27	22	22	1.33E-16
HIELOW	3	6	5	5	8.74E+02	6	5	5	8.74E+02
HILBERTA	2	3	2	2	8.22E-33	3	2	2	8.22E-33
HILBERTB	10	2	1	1	4.04E-29	2	1	1	4.04E-29
HIMMELBB	2	15	14	14	8.51E-20	15	14	14	8.51E-20
HIMMELBF	4	230	229	229	3.19E+02	231	230	230	3.19E+02
HIMMELBG	2	5	4	4	1.66E-12	5	4	4	1.66E-12
HIMMELBH	2	8	6	6	-1.00E+00	9	7	7	-1.00E+00
HUMPS	2	247	214	214	3.34E-11	502	485	485	8.44E-12
KOWOSB	4	10	6	6	3.08E-04	13	10	10	3.08E-04
LIARWHD	100	11	10	10	1.42E-13	11	10	10	1.42E-13
LOGHAIRY	2	1063	839	839	1.82E-01	152	151	151	6.18E+00
MARATOSB	2	1340	822	822	-1.00E+00	126	104	104	-1.00E+00
MEXHAT	2	42	27	27	-4.00E-02	25	24	24	-4.00E-02
MOREBV	100	3	2	2	5.49E-10	3	2	2	5.49E-10
NONCVXU2	100	257	167	167	2.33E+02	1127	812	812	2.32E+02
NONCVXUN	100	200	133	133	2.37E+02	962	699	699	2.33E+02
NONDIA	100	12	7	7	1.97E-19	10	9	9	1.79E-17
OSBORNEA	5	95	54	54	5.50E-05	28	23	23	5.46E-05

表 4: Comparison with ARNM-MC and NM-ARNM-MC

		ARNM-MC				NM-ARNM-MC			
Name	n	N_f	N_{iter}	N_{fac}	f	N_f	N_{iter}	N_{fac}	f
OSBORNEB	11	64	38	38	8.76E-02	35	26	26	8.76E-02
PALMER1C	8	6	5	5	9.76E-02	6	5	5	9.76E-02
PALMER1D	7	4	3	3	6.53E-01	4	3	3	6.53E-01
PALMER2C	8	5	4	4	1.44E-02	5	4	4	1.44E-02
PALMER3C	8	5	4	4	1.95E-02	5	4	4	1.95E-02
PALMER4C	8	5	4	4	5.03E-02	5	4	4	5.03E-02
PALMER5C	6	2	1	1	2.13E+00	2	1	1	2.13E+00
PALMER6C	8	5	4	4	1.64E-02	5	4	4	1.64E-02
PALMER7C	8	6	5	5	6.02E-01	6	5	5	6.02E-01
PALMER8C	8	6	5	5	1.60E-01	6	5	5	1.60E-01
PFIT1LS	3	1279	757	757	3.94E-07	87	63	63	8.33E-15
PFIT2LS	3	549	325	325	3.52E-09	33	32	32	2.57E-09
PFIT3LS	3	730	433	433	1.26E-12	40	34	34	2.82E-12
PFIT4LS	3	716	424	424	9.34E-09	328	233	233	6.50E-17
POWELLSG	4	16	15	15	4.38E-09	16	15	15	4.38E-09
ROSENBR	2	28	21	21	2.62E-16	18	14	14	1.19E-23
QUARTC	100	25	24	24	2.31E-08	25	24	24	2.31E-08
S308	2	12	8	8	7.73E-01	10	9	9	7.73E-01
SBRYBND	100	79	49	49	1.15E-14	33	24	24	3.82E-15
SCHMVETT	100	4	3	3	-2.94E+02	4	3	3	-2.94E+02
SINEVAL	2	77	48	48	2.02E-17	60	45	45	7.89E-29
SINQUAD	100	24	16	16	-4.01E+03	20	15	15	-4.01E+03
SISSER	2	13	12	12	1.07E-08	13	12	12	1.07E-08
SNAIL	2	152	93	93	1.07E-11	109	79	79	8.44E-13
SPARSINE	100	135	83	83	1.88E-17	233	200	200	1.05E-09
SPARSQUR	100	17	16	16	7.63E-09	17	16	16	7.63E-09
SPMSRTLS	100	41	26	26	2.51E-16	111	83	83	4.02E-16
SROSENBR	100	12	9	9	1.06E-15	6	5	5	2.49E-28
TESTQUAD	1000	2	1	1	0.00E+00	2	1	1	0.00E+00
TOINTGOR	50	7	6	6	1.37E+03	7	6	6	1.37E+03
TOINTGSS	100	2	1	1	1.00E+01	2	1	1	1.00E+01
TOINTPSP	50	32	22	22	2.26E+02	75	56	56	2.26E+02
TOINTQOR	50	2	1	1	1.18E+03	2	1	1	1.18E+03
TQUARTIC	100	83	52	52	4.64E-17	38	28	28	5.02E-19
TRIDIA	100	2	1	1	2.13E-28	2	1	1	2.13E-28
VARDIM	200	30	29	29	2.62E-24	30	29	29	2.62E-24
VAREIGVL	50	12	9	9	2.33E-12	20	18	18	5.25E-14
VIBRBEAM	8	76	50	50	7.10E+00	117	92	92	6.79E+00
WATSON	12	13	12	12	3.24E-08	13	12	12	3.24E-08

表 4: Comparison with ARNM-MC and NM-ARNM-MC

		ARNM-MC				NM-ARNM-MC			
Name	n	N_f	N_{iter}	N_{fac}	f	N_f	N_{iter}	N_{fac}	f
WOODS	4	61	40	40	2.83E-16	35	30	30	3.24E-15
YFITU	3	59	42	42	6.67E-13	37	32	32	6.67E-13
ZANGWIL2	2	2	1	1	-1.82E+01	2	1	1	-1.82E+01