Noname manuscript No.
(will be inserted by the editor)

A Second-Order Method for Strongly Convex ℓ_1-Regularization Problems

Kimon Fountoulakis and Jacek Gondzio

Technical Report ERGO-13-011

June 20, 2013

Abstract In this paper a second-order method for solving large-scale strongly convex ℓ_1-regularized problems is developed. The proposed method is a Newton-CG (Conjugate Gradients) algorithm with backtracking line-search embedded in a doubly-continuation scheme. Worst-case iteration complexity of the proposed Newton-CG is established. Based on the analysis of Newton-CG, worst-case iteration complexity of the doubly-continuation scheme is obtained.

Numerical results are presented on large-scale problems for the doubly-continuation Newton-CG algorithm, which show that the proposed second-order method competes favourably with state-of-the-art first-order methods. In addition, ℓ_1-regularized Sparse Least-Squares problems are discussed for which a parallel block coordinate descent method stagnates.

Keywords ℓ_1-regularization · Strongly convex optimization · Second-order methods · Iteration Complexity · Continuation · Newton Conjugate-Gradients method

Mathematics Subject Classification (2000) 68W40, 65K05, 90C06, 90C25, 90C30, 90C51

J. Gondzio is supported by EPSRC Grant EP/I017127/1

Kimon Fountoulakis
School of Mathematics and Maxwell Institute, The University of Edinburgh, Mayfield Road, Edinburgh EH9 3JZ, United Kingdom.
E-mail: K.Fountoulakis@sms.ed.ac.uk
Tel.: +44 131 650 5083

Jacek Gondzio
School of Mathematics and Maxwell Institute, The University of Edinburgh, Mayfield Road, Edinburgh EH9 3JZ, United Kingdom.
E-mail: J.Gondzio@ed.ac.uk
Tel.: +44 131 650 8574, Fax: +44 131 650 6553

1 Introduction

We are concerned with the solution of the following optimization problem

$$\text{minimize } f_\tau(x) := \tau\|x\|_1 + \varphi(x), \tag{1}$$

where $x \in \mathbb{R}^m$, $\tau > 0$ and $\|\cdot\|_1$ is the ℓ_1-norm. The following three assumptions are made:

- The function $\varphi(x)$ is twice differentiable, and
- strongly convex, which implies that at any x its second derivative $\nabla^2\varphi(x)$ is uniformly bounded
$$\lambda_m I \preceq \nabla^2\varphi(x) \preceq \lambda_1 I, \tag{2}$$
with $0 < \lambda_m \leq \lambda_1$, where I is the identity matrix in appropriate dimension.
- The second derivative of $\varphi(x)$ is Lipschitz continuous
$$\|\nabla^2\varphi(y) - \nabla^2\varphi(x)\| \leq L_\varphi\|y - x\|, \tag{3}$$
for any x, y, where $L_\varphi \geq 0$ is the Lipschitz constant of $\nabla^2\varphi(x)$, $\|\cdot\|$ is the ℓ_2-norm.

Recently there has been an increased interest in the solution of strongly convex problems such as (1). Such problems can be found in well-established scientific fields, for example, Regression [39] and Machine Learning [49]. Additionally, there has been a number of efficient methods in the optimization field [6,19,35,37,40–42,45,47] for the solution of problem (1). Observe, that all approaches proposed in the previously cited papers are first-order methods. There has been also a series of interesting papers which describe the adaptation of second-order methods to such problems [4,12,13,16,21–24,36] but despite their author's efforts they do not seem to compete favourably with state-of-the-art first-order methods [48].

First-order methods rely on properties of the ℓ_1-norm to obtain the new direction at each iteration. In particular, very often the direction of the solver is obtained by minimizing exactly an upper bound of the objective function in problem (1),

$$d := \arg\min_p \quad \|x + p\|_1 + \varphi(x) + \nabla\varphi(x)^\intercal p + \frac{L_\varphi}{2}\|p\|^2,$$

where x is the current iteration and d is the direction [27]. Other first-order methods use the decomposability of the former problem and solve it only for some chosen coordinates [35]. In this case, the Lipschitz constant is replaced by partial Lipschitz constants for each chosen coordinate. Another approach exploits the replacement of the ℓ_1-norm by a first-order differentiable parameterized function which approximates the former [28]. These efficient techniques of manipulating the non-smooth part, the ℓ_1-norm, in combination with the strongly convex properties of the problem have settled first-order methods as the prevalent approach. On the contrary, for second-order methods, examples of the most commonly used techniques are

– [11,36] in which the ℓ_1-norm, $\|x\|_1$, is replaced with

$$\sum_{i=1}^{m} u_i + v_i, \text{ subject to: } u_i, v_i \geq 0 \quad \forall\, i = 1, 2, \ldots, m$$

and then the optimal x is recovered by $x = u - v$.
– [12] in which the direction at every iteration is obtained by approximately solving

$$d := \arg\min_p \quad \|x + p\|_1 + \varphi(x) + \nabla\varphi(x)^\intercal p + \frac{1}{2} d^T \nabla^2 \varphi(x) d,$$

using a coordinate descent algorithm.

The drawback of these approaches is that in the first case the dimension of the problem is doubled and additional simple constraints are introduced. Whilst, in the second case, the convergence of the coordinate descent algorithm is sensitive to the spectral properties of matrix $\nabla^2\varphi(x)$. In this paper, our goals are:

1. Keep the unconstrained nature of the problem, maintain the size of it unchanged, while the consequences of the non-smoothness of the ℓ_1-norm are weakened.
2. Employ a Newton-CG algorithm with backtracking line-search which is embedded in a doubly-continuation scheme for further acceleration.
3. Give a complete analysis of Newton-CG with backtracking line-search, i.e. proof of global convergence, global and local convergence rates results, local region of fast convergence rate and worst-case iteration complexity, as in standard analysis of the Newton algorithm with backtracking line-search in [2]. *Based on a range of published papers and a book [3, 7, 8, 26, 31, 32], from various fields in which Newton-CG with backtracking line-search is applied, there has been no similar analysis.* To be precise, the analysis in the previous citations provides global convergence guarantees for various globalization techniques of Newton-CG. Additionally, local convergence rate results are obtained assuming that the method is initialized in a neighbourhood of the optimal solution which is not explicitly defined,
4. Provide worst-case iteration complexity of the doubly-continuation scheme based on the iteration complexity of Newton-CG with backtracking line-search. Continuation schemes are generally well-known among optimizers [1,10,14,15,25,43,44,46], they can speed-up a good solver, unfortunately, they are difficult to analyze. In particular, in the previously cited papers there has been no iteration complexity results.

In order to meet the first goal, the ℓ_1-norm is approximated by a smooth function which has derivatives of all degrees. Hence, problem (1) is replaced by

$$\text{minimize } f_\tau^\mu(x) := \tau\psi_\mu(x) + \varphi(x).$$

where $\psi_\mu(x)$ denotes the smooth function which replaces the ℓ_1-norm and μ is a parameter which controls the quality of approximation. Then, a Newton-CG algorithm embedded in a doubly-continuation scheme is applied to solve approximately a sequence of the above approximate problems.

In what follows in this section we give a brief introduction of the proposed approach. In Section 2, necessary basic results are given which will be used to support theoretical results in Sections 4 and 5. In Section 3, the proposed doubly-continuation scheme is discussed in details. In Section 4, the convergence analysis of Newton-CG with backtracking line-search is studied. In Section 5, worst-case iteration complexity of doubly-continuation Newton-CG with backtracking line-search is presented. Finally, in Section 6, numerical results are presented, in which we compare the proposed method with state-of-the-art first-order methods on large-scale Sparse Least-Squares and Logistic Regression problems.

1.1 Pseudo-Huber regularization

The non-smoothness of the ℓ_1-norm makes a straightforward application of the second-order method to problem (1) impossible. In this subsection, we focus on approximating the non-smooth ℓ_1-norm by a smooth function. To meet such a goal, the first-order methods community replaces the ℓ_1-norm with the so-called Huber penalty function $\sum_{i=1}^m \phi_\mu(x_i)$ [1], where

$$\phi_\mu(x_i) = \begin{cases} \frac{1}{2}\frac{x_i^2}{\mu}, & \text{if } |x_i| \leq \mu \\ |x_i| - \frac{1}{2}\mu, & \text{if } |x_i| \geq \mu \end{cases} \quad i = 1, 2, \ldots, m$$

and $\mu > 0$. The smaller the parameter μ of the Huber function is, the better the function approximates the ℓ_1-norm. Observe that the Huber function is only first-order differentiable, therefore, this approximation trick is not applicable to second-order methods. Fortunately, there is a smooth version of the Huber function, the pseudo-Huber function which has derivatives of all degrees [17]. The pseudo-Huber function parameterized with $\mu > 0$ is

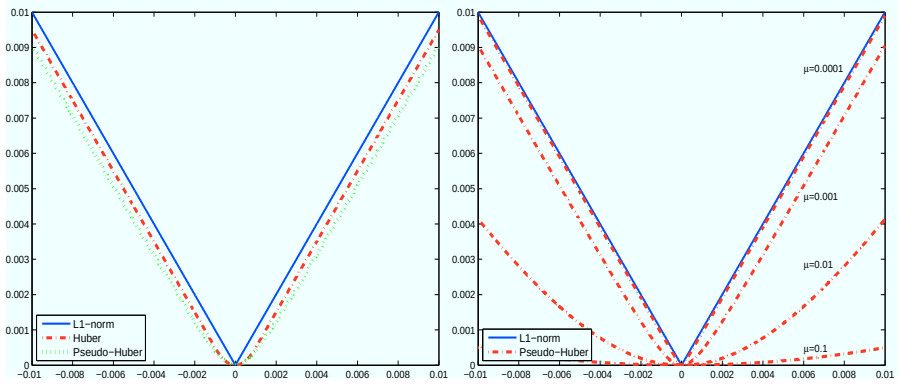
$$\psi_\mu(x) = \mu \sum_{i=1}^m \Big(\sqrt{1 + \frac{x_i^2}{\mu^2}} - 1 \Big). \tag{4}$$

A comparison of the three functions ℓ_1-norm, Huber and Pseudo-Huber function can be seen in Figure 1.

The pseudo-Huber function is employed in design of an efficient second-order method in this paper. In particular, the ℓ_1-regularization problem in (1) is replaced with the following approximation

$$\text{minimize } f_\tau^\mu(x) := \tau\psi_\mu(x) + \varphi(x). \tag{5}$$

The advantages of such an approach are listed below.



(a) ℓ_1-norm, Huber, Pseudo-Huber functions **(b)** Pseudo-Huber function for $\mu \to 0$

Fig. 1 Comparison of the approximation functions, Huber and pseudo-Huber, with the ℓ_1-norm in one dimensional space. **Fig.1a** shows the quality of approximation for the Huber and pseudo-Huber functions. **Fig.1b** shows how pseudo-Huber function converges to the ℓ_1-norm as $\mu \to 0$

- Availability of second-order information owed to the differentiability of the pseudo-Huber function.
- Avoiding the need to double problem dimensions.
- Opening the door to using iterative methods to compute descent directions.

There is an obvious cost which comes along with the above benefits, and that is the approximate nature of the pseudo-Huber function. There is a concern that in case that a very accurate solution is required, the pseudo-Huber function may be unable to deliver it. In theory, since the quality of the approximation is controlled by parameter μ in (4), see Figure 1, then the pseudo-Huber function can recover any level of accuracy under the condition that sufficiently small μ is chosen. In practise a very small parameter μ might cause instabilities in the linear algebra of the solver. However, we shall show in Section 6 that even when μ is set to small values, i.e. 1.0e–8, the proposed method is very efficient.

By replacing the initial problem (1) with the approximation (5) we obtain a smooth strongly convex problem. Since problem (5) is unconstrained the application of a Newton-CG algorithm [9,26,33] is a good fit for purpose. In particular, a Newton-CG direction is calculated by solving the Newton linear system

$$\nabla^2 f_\tau^\mu(x)d = -\nabla f_\tau^\mu(x) \tag{6}$$

using the CG algorithm [18,20,38]. The CG algorithm is terminated prematurely, therefore, an approximation of the actual Newton direction is obtained. Moreover, a backtracking line-search approach is used which guarantees global convergence of the Newton-CG algorithm. To achieve a maximum practical efficiency the Newton-CG algorithm is embedded into a doubly-continuation scheme. By the term doubly-continuation, it is meant that continuation is ap-

plied on the parameters τ and μ of problem (5) simultaneously. Details for this method will be discussed in Section 3.

2 Preliminaries

Throughout the paper the standard dot product, i.e. $\langle y, x\rangle = y^\intercal x$ is used. Occasionally, the local norm $\|\cdot\|_x := \sqrt{\langle \cdot, \nabla^2 f_\tau^\mu(x)\cdot\rangle}$ will be employed. The $\|\cdot\|_\infty$ denotes the infinity norm. The operator $diag(\cdot)$ returns a diagonal matrix with the diagonal components of the input square matrix or takes as input a vector and creates a diagonal matrix with the input vector on the diagonal. The operator $[\cdot]_{ij}$ returns the element at row i and column j of the input matrix, for vectors, the operator $[\cdot]_j$ is used instead. Finally, $\lceil\cdot\rceil$ is the ceil function.

2.1 Properties of pseudo-Huber function

The gradient of the pseudo-Huber function $\psi_\mu(x)$ in (4) is given by

$$\nabla\psi_\mu(x) = \frac{1}{\mu}\Big[x_1\Big(1+\frac{x_1^2}{\mu^2}\Big)^{-\frac{1}{2}}, \ldots, x_m\Big(1+\frac{x_m^2}{\mu^2}\Big)^{-\frac{1}{2}}\Big], \tag{7}$$

and the Hessian is given by

$$\nabla^2\psi_\mu(x) = diag\Big(\frac{1}{\mu}\Big[(1+\frac{x_1^2}{\mu^2})^{-\frac{3}{2}}, \ldots, (1+\frac{x_m^2}{\mu^2})^{-\frac{3}{2}}\Big]\Big). \tag{8}$$

The following lemma shows that the gradient of the function $\psi_\mu(x)$ is bounded.

Lemma 1 *The gradient $\nabla\psi_\mu(x)$ satisfies*

$$-1_m \preceq \nabla\psi_\mu(x) \preceq 1_m.$$

where 1_m is a vector of ones of length m.

Proof Since $-\sqrt{\mu^2+x_i^2} \leq x_i \leq \sqrt{\mu^2+x_i^2}\ \forall\, i = 1, 2, \ldots, m$, we get $-1 \leq x_i\Big(\mu^2+x_i^2\Big)^{-\frac{1}{2}} \leq 1$. The proof is complete.

The next lemma guarantees that the Hessian of the pseudo-Huber function $\psi_\mu(x)$ is bounded.

Lemma 2 *The Hessian matrix $\nabla^2\psi_\mu(x)$ satisfies*

$$0I \prec \nabla^2\psi_\mu(x) \preceq \frac{1}{\mu}I$$

where I is the identity matrix in appropriate dimension.

Proof The result follows easily by observing that $0 < \left(1 + \frac{x_i^2}{\mu^2}\right)^{-\frac{3}{2}} \leq 1$ for any x_i, $i = 1, 2 \ldots, m$. The proof is complete.

According to Lemma 2 all eigenvalues of the Hessian matrix of the pseudo-Huber function are positive, thus, the function is strictly convex. The next lemma shows that the Hessian matrix of the pseudo-Huber function is Lipschitz continuous.

Lemma 3 *The Hessian matrix $\nabla^2\psi_\mu(x)$ is Lipschitz continuous*

$$\|\nabla^2\psi_\mu(y) - \nabla^2\psi_\mu(x)\| \leq \frac{1}{\mu^2}\|y - x\|.$$

Proof

$$\begin{aligned}\|\nabla^2\psi_\mu(y) - \nabla^2\psi_\mu(x)\| &= \Big\| \int_0^1 \frac{d\nabla^2\psi_\mu(x + s(y-x))}{ds} ds \Big\| \\ &\leq \int_0^1 \Big\| \frac{d\nabla^2\psi_\mu(x + s(y-x))}{ds} \Big\| ds \end{aligned} \tag{9}$$

where $\frac{d\nabla^2\psi_\mu(x+s(y-x))}{ds}$ is a diagonal matrix with each diagonal component, $i = 1, 2, \ldots m$, given by

$$\Big[\frac{d\nabla^2\psi_\mu(x + s(y-x))}{ds}\Big]_{ii} = \frac{-3(x_i + s(y_i - x_i))(y_i - x_i)}{\mu^3(1 + \frac{(x_i+s(y_i-x_i))^2}{\mu^2})^{\frac{5}{2}}}.$$

Using the previous observation we have that

$$\Big\| \frac{d\nabla^2\psi_\mu(x + s(y-x))}{ds} \Big\| = \max_{i=1,2,\ldots,m} \Big| \Big[\frac{d\nabla^2\psi_\mu(x + s(y-x))}{ds}\Big]_{ii} \Big| \tag{10}$$

Moreover, we have

$$\begin{aligned}\Big| \Big[\frac{d\nabla^2\psi_\mu(x + s(y-x))}{ds}\Big]_{ii} \Big| &= \Big| \frac{-3(x_i + s(y_i - x_i))(y_i - x_i)}{\mu^3(1 + \frac{(x_i+s(y_i-x_i))^2}{\mu^2})^{\frac{5}{2}}} \Big| \\ &= \Big| \frac{-3(x_i + s(y_i - x_i))}{\mu^3(1 + \frac{(x_i+s(y_i-x_i))^2}{\mu^2})^{\frac{5}{2}}} \Big| |(y_i - x_i)| \end{aligned} \tag{11}$$

where the first absolute value in (11) has a maximum at $\frac{\mu-2x_i}{2(y_i-x_i)}$, which gives

$$\Big| \frac{-3(x_i + s(y_i - x_i))}{\mu^3(1 + \frac{(x_i+s(y_i-x_i))^2}{\mu^2})^{\frac{5}{2}}} \Big| \leq \frac{48}{25\sqrt{5}\mu^2} < \frac{1}{\mu^2}. \tag{12}$$

Combining (11) and (12) we get

$$\Big| \Big[\frac{d\nabla^2\psi_\mu(x + s(y-x))}{ds}\Big]_{ii} \Big| \leq \frac{1}{\mu^2}|y_i - x_i|. \tag{13}$$

Replacing (13) in (10) and using the fact that $\|\cdot\|_\infty \leq \|\cdot\|$ we get

$$\left\|\frac{d\nabla^2\psi_\mu(x+s(y-x))}{ds}\right\| \leq \frac{1}{\mu^2}\|y-x\|.$$

Replacing the above expression in (9) and calculating the integral we arrive at the desired result. The proof is complete.

Now consider the gradient of pseudo-Huber function of a particular (fixed) point x and consider a function of two parameters τ an μ defined as follows $\xi_x(\tau,\mu) = \tau\nabla\psi_\mu(x)$.

Lemma 4 *Let $\xi_x(\tau,\mu) = \tau\nabla\psi_\mu(x)$, then if $\tau \geq \tilde{\tau}$ and $\mu > \tilde{\mu}$, the following bound holds*

$$\|\xi_x(\tilde{\tau},\tilde{\mu}) - \xi_x(\tau,\mu)\| \leq \omega(\tilde{\tau},\tilde{\mu},\tau,\mu)\sqrt{m}$$

where

$$\omega(\tilde{\tau},\tilde{\mu},\tau,\mu) = 2(\tau-\tilde{\tau}) + \frac{|\tau\tilde{\mu}-\tilde{\tau}\mu|}{\mu-\tilde{\mu}}\log\frac{\mu}{\tilde{\mu}}.$$

In case that $\mu = \tilde{\mu}$, the following holds

$$\|\xi_x(\tilde{\tau},\mu) - \xi_x(\tau,\mu)\| \leq (\tau-\tilde{\tau})\sqrt{m}.$$

Proof For simplicity of notation, let us define variables $\zeta = (\tau,\mu)$ and $\tilde{\zeta} = (\tilde{\tau},\tilde{\mu})$. Then

$$\begin{aligned}\|\xi_x(\tilde{\zeta}) - \xi_x(\zeta)\| &= \left\|\int_0^1 \frac{d\xi_x(\zeta+s(\tilde{\zeta}-\zeta))}{ds}ds\right\| \\ &\leq \int_0^1 \left\|\frac{d\xi_x(\zeta+s(\tilde{\zeta}-\zeta))}{ds}\right\|ds \qquad (14)\end{aligned}$$

where

$$\frac{d\xi_x(\zeta+s(\tilde{\zeta}-\zeta))}{ds} = \frac{d\xi_x(\zeta+s(\tilde{\zeta}-\zeta))}{d(\zeta+s(\tilde{\zeta}-\zeta))}\frac{d(\zeta+s(\tilde{\zeta}-\zeta))}{ds} =$$

$$\begin{bmatrix} & -\gamma_1(s)[\nabla\psi_{\mu+s(\tilde{\mu}-\mu)}(x)]_1 \\ \nabla\psi_{\mu+s(\tilde{\mu}-\mu)}(x) & -\gamma_2(s)[\nabla\psi_{\mu+s(\tilde{\mu}-\mu)}(x)]_2 \\ & \vdots \\ & -\gamma_m(s)[\nabla\psi_{\mu+s(\tilde{\mu}-\mu)}(x)]_m \end{bmatrix}\begin{bmatrix}\tilde{\tau}-\tau \\ \tilde{\mu}-\mu\end{bmatrix} \qquad (15)$$

where

$$\gamma_i(s) = \frac{(\tau+s(\tilde{\tau}-\tau))(\mu+s(\tilde{\mu}-\mu))}{(\mu+s(\tilde{\mu}-\mu))^2+x_i^2}$$

Using Lemma 1 and (15) we get that

$$\left\|\frac{d\xi_x(\zeta+s(\tilde{\zeta}-\zeta))}{ds}\right\| \leq (|\tilde{\tau}-\tau| + \bar{\gamma}(s)|\tilde{\mu}-\mu|)\sqrt{m}, \qquad (16)$$

where

$$\bar{\gamma}(s) = \frac{\tau + s(\tilde{\tau} - \tau)}{\mu + s(\tilde{\mu} - \mu)}$$

By replacing (16) in (14) and using $\tau \geq \tilde{\tau}$ and $\mu > \tilde{\mu}$, we get the first result. For the case that $\mu = \tilde{\mu}$, using Lemma 1 we get

$$\|\xi_x(\tilde{\tau}, \mu) - \xi_x(\tau, \mu)\| = (\tau - \tilde{\tau})\|\nabla\psi_\mu(x)\| \leq (\tau - \tilde{\tau})\sqrt{m}.$$

The proof is complete.

2.2 Properties of function $f_\tau^\mu(x)$

Having all necessary properties of pseudo-Huber function (4) we can present basic properties of function $f_\tau^\mu(x)$ defined in (5). The gradient of $f_\tau^\mu(x)$ is given by

$$\nabla f_\tau^\mu(x) = \tau\nabla\psi_\mu(x) + \nabla\varphi(x)$$

where $\nabla\psi_\mu(x)$ has been defined in (7). The Hessian matrix of $f_\tau^\mu(x)$ is

$$\nabla^2 f_\tau^\mu(x) = \tau\nabla^2\psi_\mu(x) + \nabla^2\varphi(x).$$

where $\nabla^2\psi_\mu(x)$ has been defined in (8). Using (2) and Lemma 2 we get the following bounds on the Hessian matrix of $f_\tau^\mu(x)$

$$\lambda_m I \prec \nabla^2 f_\tau^\mu(x) \preceq \Big(\frac{\tau}{\mu} + \lambda_1\Big) I, \tag{17}$$

where I is the identity matrix in appropriate dimension. Hence, the equivalence of the Euclidean and the local norm is given by the following inequality

$$\lambda_m^{\frac{1}{2}}\|d\| \leq \|d\|_x \leq \Big(\frac{\tau}{\mu} + \lambda_1\Big)^{\frac{1}{2}}\|d\|. \tag{18}$$

Lemma 5 *For any x and x^*, the minimizer of $f_\tau^\mu(x)$, the following hold*

$$\frac{1}{2\Big(\frac{\tau}{\mu} + \lambda_1\Big)}\|\nabla f_\tau^\mu(x)\|^2 \leq f_\tau^\mu(x) - f_\tau^\mu(x^*) \leq \frac{1}{2\lambda_m}\|\nabla f_\tau^\mu(x)\|^2$$

and

$$\|x - x^*\| \leq \frac{2}{\lambda_m}\|\nabla f_\tau^\mu(x)\|.$$

Proof The right hand side of the first inequality is proved in page 460 of [2]. The left hand side of the first inequality is proved by using strong convexity of $f_\tau^\mu(x)$,

$$f_\tau^\mu(y) \leq f_\tau^\mu(x) + \nabla f_\tau^\mu(x)^\intercal(y - x) + \frac{\frac{\tau}{\mu} + \lambda_1}{2}\|y - x\|^2$$

and defining $\tilde{y} = x - \frac{1}{\frac{\tau}{\mu}+\lambda_1}\nabla f_\tau^\mu(x)$, we get

$$f_\tau^\mu(x) - f_\tau^\mu(x^*) \geq f_\tau^\mu(x) - f_\tau^\mu(\tilde{y}) \geq \frac{1}{2\Big(\frac{\tau}{\mu}+\lambda_1\Big)}\|\nabla f_\tau^\mu(x)\|^2.$$

The last inequality is proved in page 460 of [2]. The proof is complete.

The following lemma guarantees that the Hessian matrix $\nabla^2 f_\tau^\mu(x)$ is Lipschitz continuous. In this lemma, L_φ is defined in (3).

Lemma 6 *The function $\nabla^2 f_\tau^\mu(x)$ is Lipschitz continuous*

$$\|\nabla^2 f_\tau^\mu(y) - \nabla^2 f_\tau^\mu(x)\| \leq L_{f_\tau^\mu}\|y - x\|,$$

where $L_{f_\tau^\mu} := \frac{\tau}{\mu^2} + L_\varphi$.

Proof Using Lemma 3 and (3) we have

$$\begin{aligned}\|\nabla^2 f_\tau^\mu(y) - \nabla^2 f_\tau^\mu(x)\| &\leq \tau\|\nabla^2\psi_\mu(y) - \nabla^2\psi_\mu(x)\| + \|\nabla^2\varphi(y) - \nabla^2\varphi(x)\| \\ &\leq \Big(\frac{\tau}{\mu^2} + L_\varphi\Big)\|y - x\|.\end{aligned}$$

The Lipschitz constant of $\nabla^2 f_\tau^\mu(x)$ is therefore $L_{f_\tau^\mu} := \frac{\tau}{\mu^2} + L_\varphi$.

The next lemma gives a useful bound for the function $\nabla f_\tau^\mu(x)$ where its independent variables are the parameters τ, μ and x is a constant.

Lemma 7 *If $\tau \geq \tilde{\tau}$ and $\mu > \tilde{\mu}$, then the gradient $\nabla f_\tau^\mu(x)$ satisfies*

$$\|\nabla f_{\tilde{\tau}}^{\tilde{\mu}}(x) - \nabla f_\tau^\mu(x)\| \leq \omega(\tilde{\tau}, \tilde{\mu}, \tau, \mu)\sqrt{m}$$

with $\omega(\tilde{\tau}, \tilde{\mu}, \tau, \mu)$ as in Lemma 4. In case that $\mu = \tilde{\mu}$, the following holds

$$\|\nabla f_{\tilde{\tau}}^{\mu}(x) - \nabla f_\tau^\mu(x)\| \leq (\tau - \tilde{\tau})\sqrt{m}.$$

Proof The proof follows easily from Lemma 4 because $\|\nabla f_{\tilde{\tau}}^{\tilde{\mu}}(x) - \nabla f_\tau^\mu(x)\| = \|\xi_x(\tilde{\tau}, \tilde{\mu}) - \xi_x(\tau, \mu)\|$. The proof is complete.

Using the reverse triangular inequality on $\nabla f_\tau^\mu(x)$ as a function of parameters τ and μ we get

$$\|\nabla f_{\tilde{\tau}}^{\tilde{\mu}}(x)\| - \|\nabla f_\tau^\mu(x)\| \leq \|\nabla f_{\tilde{\tau}}^{\tilde{\mu}}(x) - \nabla f_\tau^\mu(x)\|$$

and by applying Lemma 7 in the former we get

$$\|\nabla f_{\tilde{\tau}}^{\tilde{\mu}}(x)\| \leq \|\nabla f_\tau^\mu(x)\| + \omega(\tilde{\tau}, \tilde{\mu}, \tau, \mu)\sqrt{m}, \tag{19}$$

and

$$\|\nabla f_{\tilde{\tau}}^{\mu}(x)\| \leq \|\nabla f_\tau^\mu(x)\| + (\tau - \tilde{\tau})\sqrt{m}, \tag{20}$$

respectively. The next lemma shows how well the second-order Taylor expansion of $f_\tau^\mu(x)$ approximates the function $f_\tau^\mu(x)$.

Lemma 8 *If $q_\tau^\mu(y)$ is a quadratic approximation of the function $f_\tau^\mu(x)$ at x*

$$q_\tau^\mu(y) := f_\tau^\mu(x) + \nabla f_\tau^\mu(x)^\intercal(y-x) + \frac{1}{2}(y-x)^\intercal \nabla^2 f_\tau^\mu(x)(y-x),$$

then

$$|f_\tau^\mu(y) - q_\tau^\mu(y)| \leq \frac{1}{6} L_{f_\tau^\mu} \|y-x\|^3.$$

Proof Using corollary 1.5.3 in [34] and Lemma 6 we have

$$\begin{aligned} |f_\tau^\mu(y) - q_\tau^\mu(y)| &\leq \|y-x\|^2 \int_0^1 \int_0^t \|\nabla^2 f_\tau^\mu(x + s(y-x)) - \nabla^2 f_\tau^\mu(x)\| ds dt \\ &\leq \|y-x\|^2 \int_0^1 \int_0^t s L_{f_\tau^\mu} \|y-x\| ds dt \\ &= \frac{1}{6} L_{f_\tau^\mu} \|y-x\|^3. \end{aligned}$$

The proof is complete.

2.3 Approximate Newton directions via Conjugate Gradients algorithm

In the case that pseudo-Huber regularization is performed, the direction at every iteration is calculated by a Newton-CG algorithm. This implies that the Newton system (6) is solved iteratively using the CG algorithm and the process is truncated when a required accuracy is obtained. The result is an approximate solution of the Newton system, which means that an inexact Newton direction is calculated. For Newton-CG, the CG algorithm is always initialized with the zero solution and the termination criterion is set to

$$\|r_\tau^\mu(x)\| \leq \eta \|\nabla f_\tau^\mu(x)\|, \tag{21}$$

where $r_\tau^\mu(x) = \nabla^2 f_\tau^\mu(x) d + \nabla f_\tau^\mu(x)$ is the residual in equation (6), $0 \leq \eta < 1$ is a user-defined constant and $d \in \mathbb{R}^m$ is the Newton-CG direction. In the literature there has been extensive theoretical work regarding the termination parameter η. The most efficient of the proposed approaches of setting η suggest that the parameter should be varied from iteration to iteration starting from a well-defined large value and steadily decreasing. According to [31], page 167, if k is the iteration index of Newton-CG, then by setting

$$\eta^k = \min\{\frac{1}{2}, \|\nabla f_\tau^\mu(x^k)\|^{c_0}\}, \tag{22}$$

local super-linear convergence of Newton-CG algorithm is obtained for $c_0 = \frac{1}{2}$, whilst, local quadratic convergence is obtained for $c_0 = 1$. Although in practice we have observed that by setting η^k =1.0e–1 results in very fast convergence of the Newton-CG algorithm, it will be analyzed for η^k set as in (22) with $c_0 = 1$. The next lemma determines bounds on the norm of the Newton-CG direction d as a function of $\|\nabla f_\tau^\mu(x)\|$.

Lemma 9 *Let $d \in \mathbb{R}^m$ be the Newton-CG direction calculated by CG algorithm which is terminated according to criterion (21) with $0 \leq \eta < 1$. Then the following holds*

$$\frac{1-\eta^2}{2(\frac{\tau}{\mu}+\lambda_1)}\|\nabla f_\tau^\mu(x)\| \leq \|d\| \leq \frac{2}{\lambda_m}\|\nabla f_\tau^\mu(x)\|$$

Proof By squaring (21) and making simple rearrangements of it we get

$$d^\intercal(\nabla^2 f_\tau^\mu(x))^2 d + 2\nabla f_\tau^\mu(x)^\intercal \nabla^2 f_\tau^\mu(x) d + (1-\eta^2)\|\nabla f_\tau^\mu(x)\|^2 \leq 0.$$

Using (17) in the above inequality we get

$$\lambda_m^2\|d\|^2 - 2\Big(\frac{\tau}{\mu}+\lambda_1\Big)\|\nabla f_\tau^\mu(x)\|\|d\| + (1-\eta^2)\|\nabla f_\tau^\mu(x)\|^2 \leq 0.$$

By dropping the quadratic term $\lambda_m^2\|d\|^2$ from the previous inequality and making appropriate rearrangements we get

$$\|d\| \geq \frac{1-\eta^2}{2(\frac{\tau}{\mu}+\lambda_1)}\|\nabla f_\tau^\mu(x)\|.$$

Which proves the left hand side of the result. For the right hand side, we simply use the equality

$$d = \nabla^2 f_\tau^\mu(x)^{-1}(-\nabla f_\tau^\mu(x) + r_\tau^\mu(x)).$$

By taking norms and using (21) and $\|\nabla^2 f_\tau^\mu(x)^{-1}\| = \frac{1}{\lambda_m}$ we obtain the right hand side of the result. The proof is complete.

The following property is shown in the proof of Lemma 2.4.1 in [20]. For a CG algorithm that is initialised with the zero solution $p_0 = 0$ the returned direction d_i satisfies

$$d_i := \arg\min_p\{\nabla f_\tau^\mu(x)^\intercal p + \frac{1}{2}p^\intercal \nabla^2 f_\tau^\mu(x) p \mid p \in \mathcal{E}_i\},$$

where

$$\mathcal{E}_i := \text{span}(\nabla f_\tau^\mu(x), \nabla^2 f_\tau^\mu(x)\nabla f_\tau^\mu(x), \ldots, \nabla^2 f_\tau^\mu(x)^{i-1}\nabla f_\tau^\mu(x)).$$

Therefore for every $p \in \mathcal{E}_i$, at $t = 0$, we get

$$\frac{d(\nabla f_\tau^\mu(x)^\intercal(d_i+tp) + \frac{1}{2}(d_i+tp)^\intercal \nabla^2 f_\tau^\mu(x)(d_i+tp))}{dt} = (\nabla f_\tau^\mu(x)^\intercal + \nabla^2 f_\tau^\mu(x) d_i)^\intercal p = 0.$$

Since, $d_i \in \mathcal{E}_i$, then

$$(\nabla f_\tau^\mu(x)^\intercal + \nabla^2 f_\tau^\mu(x) d_i)^\intercal d_i = 0 \iff d_i^\intercal \nabla^2 f_\tau^\mu(x) d_i = -\nabla f_\tau^\mu(x)^\intercal d_i. \tag{23}$$

As a termination criterion of the Newton-CG algorithm we will use an upper bound of the Newton decrement $\|d_N\|_x$, where d_N is the Newton direction at a given point x,

$$\|d_N\|_x = \inf\{c_1 \mid |\nabla f_\tau^\mu(x)^\intercal h| \leq c_1 \|h\|_x \ \forall h \in \mathbb{R}^m\}.$$

It is shown in [30], pages 15-16, that the optimal $c_1 = \|d_N\|_x$ of the previous minimization problem is given when $h = d_N$. Therefore, for $h = d$, given by CG, and due to (23) we have $\|d_N\|_x \leq \|d\|_x$. The measure $\|d\|_x$ is inexpensive to be calculated, because of the relation (23) and will be used as a termination criterion of Newton-CG algorithm. For details about the Newton decrement see page 15 in [30].

3 Doubly-Continuation Newton-CG

The pseudo-Huber regularization problem (5) to be solved is a smooth and strongly convex problem. Therefore, one could directly apply an efficient and simple Newton-CG algorithm. In this paper, we make a step further and embed a Newton-CG algorithm with backtracking line-search in a doubly-continuation scheme. Continuation schemes define a sequence of minimizers as a function of a parameter. For ℓ_1-regularization problems as in (1), it is common that this sequence depends on parameter τ. In such cases the continuation path, is constructed for decreasing values of τ to a user defined constant. For the proposed method, the continuation path, consists of minimizers of problem (5) for simultaneously decreasing parameters τ and μ, till $\tau \to \bar{\tau}$ and $\mu \to \bar{\mu}$, where $\bar{\tau}$ and $\bar{\mu}$ are pre-defined parameters. This explains the term "doubly-continuation". The following two observations justify the purpose of the doubly-continuation scheme.

- It is shown in [22] that problem (1) has the zero solution for any $\tau \geq \|\nabla\varphi(0)\|_\infty$. In case of problem (5), by setting μ to a relatively large value, i.e. $\mu \approx 1$, the regularization effect of the ℓ_1-norm is weakened, see Figure 1b. Therefore, $\tau \geq \|\nabla\varphi(0)\|_\infty$ is not a sufficient condition for the subproblem (5) to have the zero solution. However, our empirical experience shows that Newton-CG is warm-started, i.e. Newton-CG converges in few iterations, when the initial solution is set to zero and $\tau \gg \|\nabla\varphi(0)\|_\infty$.
- The linear algebra is favoured when $\tau = \mathcal{O}(\mu)$, since in this case the Hessian matrices $\nabla^2 f_\tau^\mu(x)$ are tightly bounded, see (17).

Briefly, large values of τ favour warm-starting, while a small ratio $\frac{\tau}{\mu}$ assures good numerical properties of linear systems to be solved. However, the parameters τ and μ have to be decreased to their pre-defined values eventually. We propose a continuation approach which consists of gradual decreasing τ and μ from their large initial values to the ones which determine an accurate enough solution to the original problem (1). We expect that solving a sequence of easier problems is much more efficient than giving one-shot to the actual

problem itself. The process of decreasing the τ, μ parameters follows the rule

$$\tau^{l+1} = \max\{\beta_1 \tau^l,\ \bar{\tau}\} \quad \text{and} \quad \mu^{l+1} = \max\{\beta_2 \mu^l,\ \bar{\mu}\}, \tag{24}$$

where $0 < \beta_1, \beta_2 < 1$. Observe in (24) that the parameters τ and μ can be decreased at every iteration arbitrarily. The former implies that the proposed method is long-step depending on these parameters. Long-step, in the sense that the reduction parameters β_1 and β_2 is not limited in order to guarantee convergence of the doubly-continuation scheme.

The pseudo-code of the proposed doubly-continuation Newton-CG and the backtracking line-search algorithms follow. In these pseudo-codes we redefine the notation of the local inner product to distinguish among continuation iterations, $\|\cdot\|_{x,l} = \sqrt{\langle \cdot, \nabla^2 f_{\tau^l}^{\mu^l}(x) \cdot \rangle}$.

Algorithm dcNCG

1: **Input** Choose x^0 and $\tau^0, \mu^0, \bar{\tau}, \bar{\mu}, \epsilon > 0$, $0 < \beta_1, \beta_2 < 1$.

2: Set flag$= 0$.

3: For $l = 0, 1, 2, \ldots$ and $k = 0, 1, 2, \ldots$ generate τ^{l+1}, μ^{l+1} from τ^l, μ^l and x^{k+1} from x^k, respectively, according to the iteration:

4: **while** flag< 2 **do**

5: **for** maximum Newton-CG iterations **do**

6: Use CG algorithm to approximately solve $\nabla^2 f_{\tau^l}^{\mu^l}(x^k) d^k = -\nabla f_{\tau^l}^{\mu^l}(x^k)$, till (21) is satisfied with η as in (22) and $c^0 = 1$.

7: If $\|d^k\|_{x^k,l} \leq \epsilon^l$ break for-loop.

8: Calculate $0 < \alpha^k \leq 1$ which satisfies $f_{\tau^l}^{\mu^l}(x^k + \alpha^k d^k) \leq f_{\tau^l}^{\mu^l}(x^k) - c_2 \alpha^k \|d^k\|_{x^k,l}^2$, where $c_2 \in (0, \frac{1}{2})$, by using backtracking line-search.

9: set $x^{k+1} = x^k + \alpha^k d^k$ and $k = k + 1$

10: **end for**

11: $\tau^{l+1} = \max\{\beta_1 \tau^l, \bar{\tau}\}$ and $\mu^{l+1} = \max\{\beta_2 \mu^l, \bar{\mu}\}$

12: If $(\tau^{l+1}, \mu^{l+1}) = (\bar{\tau}, \bar{\mu})$, then set $\epsilon^l = \epsilon$ and flag=flag+1

13: l $=$ l $+ 1$

14: **end while**

The outer loop of the above algorithm is the doubly-continuation scheme on the parameters τ, μ. The inner loop is the Newton-CG algorithm with backtracking line-search that minimizes $f_{\tau^l}^{\mu^l}(x)$ for fixed parameters τ^l, μ^l given from the l^{th} outer loop. Although, the termination criterion of CG algorithm is set as in (22), in practise it can be relaxed to any $0 \leq \eta < 1$. Step 8 is the backtracking line-search algorithm given below.

Algorithm backtracking line-search

1: **Input** Given τ^l, μ^l and an inexact Newton direction d for point x, calculated by CG in step 6 of algorithm dcNCG and $c_2 \in (0, \frac{1}{2})$, $c_3 \in (0, 1)$, then calculate a step-size α:

2: set $\alpha = 1$

3: **while** $f_{\tau^l}^{\mu^l}(x + \alpha d) > f_{\tau^l}^{\mu^l}(x) - c_2 \alpha \|d\|_{x,l}^2$ **do**

4: $\quad \alpha = c_3 \alpha$

5: **end while**

For more details about backtracking line-search the reader is referred to Chapter 9 in [2]. The termination criteria of the inner loop in step 7 can be relaxed, for example, $\epsilon^1 \leq \epsilon^0$ and $\epsilon^l \leq \epsilon^{l-1}$ $\forall l > 0$, since the subproblems (5) for given $\tau^l \neq \bar{\tau}$ and $\mu^l \neq \bar{\mu}$ do not have to be solved in high accuracy.

The analysis of Newton-CG algorithm, steps 5-10, without the continuation scheme is presented in the next section. This analysis will provide us with useful results that will be used for the proof of the worst-case iteration complexity of the doubly-continuation scheme.

4 Convergence analysis of Newton-CG with backtracking line-search

In this section we analyze the Newton-CG with backtracking line-search, steps 5-10, of algorithm dcNCG. In particular, we prove global convergence, we study the global and local convergence rates and we explicitly define a region in which Newton-CG has fast convergence rate. Additionally, worst-case iteration complexity result of this Newton-CG is presented.

Since Newton-CG in this section does not depend on the continuation scheme, the indexes τ and μ from function $f_\tau^\mu(x)$ are dropped. Moreover, the upper bound of the largest eigenvalue of $\nabla^2 f_\tau^\mu(x)$, shown in (17), is denoted by $\tilde{\lambda}_1 = \left(\frac{\tau}{\mu} + \lambda_1\right)$, and the Lipschitz constant $L_{f_\tau^\mu}$ defined in Lemma 6, is denoted by L. Finally, for this section, an upper bound of the condition number of matrix $\nabla^2 f(x)$ for any x will be used, which is denoted by $\kappa = \frac{\tilde{\lambda}_1}{\lambda_m}$.

4.1 Global convergence rate of Newton-CG with backtracking line-search

In the following lemma the decrease of the objective function at every iteration of Newton-CG with backtracking line-search is calculated. In this lemma the constants c_2 and c_3 are defined in the backtracking line-search algorithm given in Section 3.

Lemma 10 *Let $x \in \mathbb{R}^m$ be the current iteration of Newton-CG, $d \in \mathbb{R}^m$ be the Newton-CG direction calculated using the truncated CG algorithm described in subsection 2.3. The parameter η of the termination criterion (21) of the CG*

algorithm is set to $0 \leq \eta < 1$. *If* x *is not the minimizer of problem* (5), *i.e.* $\|\nabla f(x)\| \neq 0$, *then, the backtracking line-search algorithm will calculate a step-size* $\bar{\alpha}$ *such that*

$$\bar{\alpha} \geq c_3 \frac{\lambda_m}{\tilde{\lambda}_1}.$$

For this step-size $\bar{\alpha}$ *the following holds*

$$f(x) - f(x(\bar{\alpha})) > c_4 \|d\|_x^2,$$

where $c_4 = c_2 c_3 \frac{1}{\kappa}$ *and* $x(\bar{\alpha}) = x + \bar{\alpha} d$.

Proof For $x(\alpha) = x + \alpha d$ and from strong convexity of $f(x)$ we have

$$f(x(\alpha)) \leq f(x) + \alpha \nabla f(x)^\intercal d + \frac{\alpha^2}{2} \tilde{\lambda}_1 \|d\|^2.$$

If $\|\nabla f(x)\| \neq 0$, due to positive definiteness of $\nabla^2 f(x)$, see (17), the CG algorithm terminated at the i^{th} iteration returns the vector $d_i \neq 0$ which according to (23) satisfies

$$d_i^\intercal \nabla^2 f(x) d_i = -\nabla f(x)^\intercal d_i.$$

Therefore, by setting $d := d_i$ we get

$$f(x(\alpha)) \leq f(x) - \alpha \|d\|_x^2 + \frac{\alpha^2}{2} \tilde{\lambda}_1 \|d\|^2.$$

Using (18) we get

$$f(x(\alpha)) \leq f(x) - \alpha \|d\|_x^2 + \frac{\alpha^2}{2} \frac{\tilde{\lambda}_1}{\lambda_m} \|d\|_x^2.$$

The right hand side of the above inequality is minimized for $\bar{\alpha} = \frac{\lambda_m}{\tilde{\lambda}_1}$, which gives

$$f(x(\bar{\alpha})) \leq f(x) - \frac{1}{2} \frac{\lambda_m}{\tilde{\lambda}_1} \|d\|_x^2.$$

Observe, that for this step-size the exit condition of the backtracking line-search algorithm is satisfied, since

$$f(x(\bar{\alpha})) \leq f(x) - \frac{1}{2} \frac{\lambda_m}{\tilde{\lambda}_1} \|d\|_x^2 < f(x) - c_2 \frac{\lambda_m}{\tilde{\lambda}_1} \|d\|_x^2.$$

Therefore, the step-size $\bar{\alpha}$ returned by the backtracking line-search algorithm is in worst-case bounded by

$$\bar{\alpha} \geq c_3 \frac{\lambda_m}{\tilde{\lambda}_1},$$

which results in the following decrease of the objective function

$$f(x) - f(x(\bar{\alpha})) > c_2 c_3 \frac{\lambda_m}{\tilde{\lambda}_1} \|d\|_x^2 = c_2 c_3 \frac{1}{\kappa} \|d\|_x^2.$$

The proof is complete.

Standard convergence analysis treats the Newton algorithm with backtracking line-search as a two phase method. For the first phase, which is of our interest in this subsection, linear convergence rate is proved. In particular, convergence is measured according to the decrease of the objective function at every iteration, which is proved to be $c_4\|d_N\|_x^2$, where d_N is the Newton direction, see page 490 in [2]. For the Newton-CG algorithm with backtracking line-search, which is studied in this paper, it is proved in Lemma 10 that the decrease of the objective function at every iteration of Newton-CG for the first phase is $c_4\|d\|_x^2$, where d is the inexact Newton direction returned by CG. Hence, *Newton-CG with backtracking line-search enjoys analogous global linear convergence rate to that of Newton algorithm with backtracking line-search.*

4.2 Region of fast convergence rate of Newton-CG

In this section we define a region based on $\|d\|_x$, in which, by setting parameter η as in (22) with $c_0 = 1$, the Newton-CG algorithm converges with fast rate. By fast rate it is meant that when Newton-CG is initialized in this region, then the worst-case iteration complexity result is of the form $\log_2 \log_2 \frac{\text{constant}}{\text{required accuracy}}$. We will show in Subsection 4.4 that the rate of convergence which is proved in this subsection for the explicitly defined region satisfies the previously stated condition. The lemma below is crucial for the analysis in this subsection, it shows the behaviour of the function $f(x)$ when a step along the Newton-CG direction is made.

Lemma 11 *Let $x \in \mathbb{R}^m$ be the current iteration of Newton-CG, $d \in \mathbb{R}^m$ be the Newton-CG direction calculated using the CG algorithm described in subsection 2.3. The parameter η of the termination criterion (21) of the CG algorithm is set to $0 \leq \eta < 1$. Then*

$$f(x) - f(x(\alpha)) \geq \alpha\|d\|_x^2 - \frac{\alpha^2}{2}\|d\|_x^2 - \frac{\alpha^3}{6}\frac{L}{\lambda_m^{\frac{3}{2}}}\|d\|_x^3,$$

where $x(\alpha) = x + \alpha d$ and $\alpha > 0$.

Proof Using Lemma 8 and setting $y = x(\alpha) = x + \alpha d$ we get

$$f(x(\alpha)) \leq f(x) + \alpha\nabla f(x)^\intercal d + \frac{\alpha^2}{2}d^\intercal\nabla^2 f(x)d + \frac{\alpha^3}{6}L\|d\|^3.$$

Using (18) and (23) we get

$$f(x(\alpha)) \leq f(x) - \alpha\|d\|_x^2 + \frac{\alpha^2}{2}\|d\|_x^2 + \frac{\alpha^3}{6}\frac{L}{\lambda_m^{\frac{3}{2}}}\|d\|_x^3.$$

The result is obtained by rearrangement of terms. The proof is complete.

Based on Lemma 11, a region is defined in the following lemma, in which unit-step sizes are calculated by the backtracking line-search algorithm. Additionally, for this region, $\|d^{k+1}\|_{x^{k+1}}$ is bounded as a function of $\|d^k\|_{x^k}$. In this lemma the constants c_2 and c_3 have been defined in the backtracking line-search algorithm, given in Section 3, moreover, $x^{k+1} = x^k + d^k$.

Lemma 12 *If $\|d^k\|_{x^k} \leq 3(1-2c_2)\frac{\lambda_m^{\frac{3}{2}}}{L}$, then the backtracking line-search algorithm calculates unit step-sizes. Moreover, if the parameter η^k of the termination criterion (21) of the CG algorithm is set as in (22) with $c_0 = 1$, then for two consequent directions d^k and d^{k+1} and points x^k and x^{k+1} of Newton-CG algorithm, the following holds*

$$\frac{1}{2}\frac{16\tilde{\lambda}_1^2 + L}{\lambda_m^{\frac{3}{2}}}\|d^{k+1}\|_{x^{k+1}} \leq \Big(\frac{1}{2}\frac{16\tilde{\lambda}_1^2 + L}{\lambda_m^{\frac{3}{2}}}\|d^k\|_{x^k}\Big)^2.$$

Proof By setting $\bar{\alpha} = 1$ in Lemma 11 we get

$$f(x^k) - f(x^{k+1}) \geq \frac{1}{2}\|d^k\|_{x^k}^2 - \frac{1}{6}\frac{L}{\lambda_m^{\frac{3}{2}}}\|d^k\|_{x^k}^3 = \Big(\frac{1}{2} - \frac{1}{6}\frac{L}{\lambda_m^{\frac{3}{2}}}\|d^k\|_{x^k}\Big)\|d^k\|_{x^k}^2.$$

if $\|d^k\|_{x^k} \leq 3(1-2c_2)\frac{\lambda_m^{\frac{3}{2}}}{L}$ we get

$$f(x^k) - f(x^{k+1}) \geq c_2\|d^k\|_{x^k}^2,$$

which implies that $\bar{\alpha} = 1$ satisfies the exist condition of the backtracking line-search algorithm. Let us define the quantity $\nabla f(x(t))^\intercal h$, where $x(t) = x^k + td^k$ and $h \in \mathbb{R}^m$, then we have

$$\begin{aligned}
\nabla f(x(t))^\intercal h &= \nabla f(x^k)^\intercal h + t(d^k)^\intercal \nabla^2 f(x^k)h \\
&\quad + \int_0^t \int_0^u \nabla^3 f(x(\delta))[d^k, d^k, h]d\delta du \\
&\leq \nabla f(x^k)^\intercal h + t(d^k)^\intercal \nabla^2 f(x^k)h \\
&\quad + \int_0^t \int_0^u \Big|\nabla^3 f(x(\delta))[d^k, d^k, h]\Big|d\delta du \\
&= \nabla f(x^k)^\intercal h + t(d^k)^\intercal \nabla^2 f(x^k)h \\
&\quad + \int_0^t \int_0^u \lim_{\delta\to 0}\Big|\frac{(d^k)^\intercal(\nabla^2 f(x(\delta)) - \nabla^2 f(x^k))h}{\delta}\Big|d\delta du \\
&\leq \nabla f(x^k)^\intercal h + t(d^k)^\intercal \nabla^2 f(x^k)h \\
&\quad + \|d^k\|\|h\| \int_0^t \int_0^u \lim_{\delta\to 0}\Big\|\frac{1}{\delta}(\nabla^2 f(x(\delta)) - \nabla^2 f(x^k))\Big\|d\delta du \\
&\leq \nabla f(x^k)^\intercal h + t(d^k)^\intercal \nabla^2 f(x^k)h + \|d^k\|\|h\| \int_0^t \int_0^u L\|d^k\|d\delta du \\
&= \nabla f(x^k)^\intercal h + t(d^k)^\intercal \nabla^2 f(x^k)h + \frac{t^2}{2}L\|d^k\|^2\|h\|.
\end{aligned}$$

By taking absolute values and setting $t = 1$ we get

$$|\nabla f(x^{k+1})^\intercal h| \leq |\nabla f(x^k)^\intercal h + (d^k)^\intercal \nabla^2 f(x^k) h| + \frac{1}{2} L \|d^k\|^2 \|h\|$$

From (18) and (21), by setting $r^k = \nabla^2 f(x^k) d^k + \nabla f(x^k)$, we get

$$|\nabla f(x^{k+1})^\intercal h| \leq \Big(\frac{\eta^k}{\lambda_m^{\frac{1}{2}}} \|\nabla f(x^k)\| + \frac{1}{2} \frac{L}{\lambda_m^{\frac{3}{2}}} \|d^k\|_{x^k}^2 \Big) \|h\|_{x^{k+1}}.$$

Since, η^k is set according to (22), by using (18) and Lemma 9 we get

$$\begin{aligned} |\nabla f(x^{k+1})^\intercal h| &\leq \Big(\frac{4\tilde{\lambda}_1^2}{\lambda_m^{\frac{3}{2}} (1 - (\eta^k)^2)^2} + \frac{1}{2} \frac{L}{\lambda_m^{\frac{3}{2}}} \Big) \|d^k\|_{x^k}^2 \|h\|_{x^{k+1}} \\ &\leq \frac{1}{2} \Big(\frac{16\tilde{\lambda}_1^2 + L}{\lambda_m^{\frac{3}{2}}} \Big) \|d^k\|_{x^k}^2 \|h\|_{x^{k+1}}. \end{aligned}$$

The previous result holds for every $h \in \mathbb{R}^m$, hence, by setting $h = d^{k+1}$ and by using (23) we prove the second part of this lemma. The proof is complete.

The following corollary states the region of fast convergence rate of Newton-CG.

Corollary 1 *If the parameter η^k in the termination criterion (21) of the CG algorithm is set as in* (22) *with $c_0 = 1$ and $\|d^k\|_{x^k} < \varpi$, $0 < \varpi \leq c_5$, where*

$$c_5 = \min \Big\{ 3(1 - 2c_2) \frac{\lambda_m^{\frac{3}{2}}}{L}, \frac{\lambda_m^{\frac{3}{2}}}{16\tilde{\lambda}_1^2 + L} \Big\},$$

then according to Lemma 12 Newton-CG convergences with fast rate.

4.3 Global Convergence of Newton-CG with backtracking line-search

The global convergence result of Newton-CG algorithm with backtracking line-search, steps 5-10 of dcNCG algorithm, follows.

Theorem 1 *Let $\{x^k\}$ be a sequence generated by Newton-CG algorithm with step-sizes calculated by backtracking line-search. The parameter η of the termination criterion* (21) *of the CG algorithm is set to a value $0 \leq \eta < 1$. The sequence $\{x^k\}$ converges to x^*, which is the minimizer of $f(x)$ in problem (5).*

Proof Let $\{x^k\}$ be a sequence generated by Newton-CG algorithm with step-sizes $\bar{\alpha}^k$ calculated by the backtracking line-search algorithm presented in Section 3. The positive definiteness of $\nabla^2 f(x)$ at any x, see (17), and the fact that $0 \leq \eta < 1$ in (21), imply that CG returns $d^k = 0$ at a point x^k if and only if $\nabla f(x^k) = 0$. Hence, only at optimality CG will return a zero direction. Moreover, from Lemma 10 we get that if $\|\nabla f(x^k)\| \neq 0$, then $\bar{\alpha}^k$ is bounded away from zero and the function $f(x)$ is monotonically decreasing

when the step $\bar{\alpha}^k d^k$ is applied. The monotonic decrease of the objective function implies that $\{f(x^k)\}$ converges to a limit, thus, $\{f(x^k) - f(x^{k+1})\} \to 0$. Therefore, the sequence $\{x^k\}$ converges to a point x^*. Using Lemma 10 and $\{f(x^k) - f(x^{k+1})\} \to 0$ we get that $\|d^k\|_x \to 0$, hence, due to strong convexity of $f(x)$, $\|d^k\| \to 0$. Moreover, from Lemma 9 we deduce that if $\|d^k\| \to 0$, then $\|\nabla f(x^k)\| \to 0$, therefore, x^* is a stationary point of function $f(x)$. Strong convexity of $f(x)$ guarantees that a stationary point must be a minimizer. The proof is complete.

4.4 Worst-case iteration complexity of Newton-CG with backtracking line-search

The following theorem shows the worst-case iteration complexity of Newton-CG with backtracking line-search in order to enter the region of fast convergence rate, i.e. $\|d\|_x < \varpi$, where $0 < \varpi \leq c_5$ and c_5 has been defined in Corollary 1. In this theorem the constant c_4 has been defined in Lemma 10, c_2 and c_3 are constants of the backtracking line-search algorithm given in Section 3. Moreover, x^* denotes the minimizer of $f(x)$.

Theorem 2 *Starting from an initial point x^0, such that $\|d^0\|_{x^0} \geq \varpi$, Newton-CG with backtracking line-search and $0 \leq \eta < 1$ in the termination criterion* (21) *of CG, requires at most*

$$K_1 = c_6 \log\Big(\frac{f(x^0) - f(x^*)}{c_7 \varpi^2}\Big),$$

iterations to obtain a solution x^k, $k > 0$, such that $\|d^k\|_{x^k} < \varpi$, where

$$c_6 = \frac{2\kappa^3}{(1-\eta^2)^2 c_2 c_3} \quad \textit{and} \quad c_7 = \frac{1}{8\kappa^2}.$$

Proof Let us assume an iteration index $k > 0$, then from (18), Lemmas 5 and 9 we get

$$f(x^k) - f(x^*) \geq \frac{1}{8\kappa^2}\|d^k\|^2_{x^k}, \tag{25}$$

and

$$f(x^{k-1}) - f(x^*) \leq \frac{2\kappa^2}{(1-\eta^2)^2}\|d^{k-1}\|^2_{x^{k-1}}. \tag{26}$$

From Lemma 10 we have

$$f(x^k) < f(x^{k-1}) - c_4\|d^{k-1}\|^2_{x^{k-1}}. \tag{27}$$

Combining (26), (27) and subtracting $f(x^*)$ from both sides we get

$$\begin{aligned} f(x^k) - f(x^*) &< \Big(1 - \frac{(1-\eta^2)^2 c_4}{2\kappa^2}\Big)(f(x^{k-1}) - f(x^*)) \\ &< \Big(1 - \frac{(1-\eta^2)^2 c_4}{2\kappa^2}\Big)^k (f(x^0) - f(x^*)) \\ &= \Big(1 - \frac{(1-\eta^2)^2 c_2 c_3}{2\kappa^3}\Big)^k (f(x^0) - f(x^*)) \end{aligned}$$

From the last inequality and (25) we get

$$\frac{1}{8\kappa^2}\|d^k\|_{x^k}^2 < \Big(1 - \frac{(1-\eta^2)^2 c_2 c_3}{2\kappa^3}\Big)^k (f(x^0) - f(x^*)).$$

Using the definitions of constants c_6 and c_7 we have

$$\|d^k\|_{x^k}^2 < \Big(1 - \frac{1}{c_6}\Big)^k \frac{1}{c_7}(f(x^0) - f(x^*)).$$

Hence, we conclude that after at most K_1 iterations as defined in the preamble of this theorem, the algorithm produces $\|d^k\|_{x^k} < \varpi$. The proof is complete.

The following theorem presents the worst-case iteration complexity result of Newton-CG to obtain a solution x^l, of accuracy $f(x^l) - f(x^*) < \epsilon$, when initialized at a point inside the region of fast convergence. The index l has been used as an iteration counter of the doubly-continuation loop, however, only for the next theorem it will be used as an iteration counter of Newton-CG algorithm.

Theorem 3 *Suppose that there is an iteration index k, such that $\|d^k\|_{x^k} < \varpi$. If η in (21) is set as in (22) with $c_0 = 1$, then Newton-CG needs at most*

$$K_2 = \log_2 \log_2 \Big(\frac{c_8}{\epsilon}\Big)$$

iterations to obtain a solution x^l, $l > k$, such that $f(x^l) - f(x^*) < \epsilon$, where

$$c_8 = \frac{32\kappa^2\lambda_m^3}{(16\tilde{\lambda}_1^2 + L)^2}.$$

Proof Suppose that there is an iteration index k such that $\|d^k\|_{x^k} < \varpi$, then for an index $l > k$, by applying Lemma 12 recursively we get

$$\frac{1}{2}\frac{16\tilde{\lambda}_1^2 + L}{\lambda_m^{\frac{3}{2}}}\|d^l\|_{x^l} \le \Big(\frac{1}{2}\frac{16\tilde{\lambda}_1^2 + L}{\lambda_m^{\frac{3}{2}}}\|d^k\|_{x^k}\Big)^{2^{l-k}} < \Big(\frac{1}{2}\Big)^{2^{l-k}}. \tag{28}$$

From (18), Lemmas 5 and 9 and η^k in (21) we get

$$f(x^l) - f(x^*) \le \frac{2\kappa^2}{(1-\eta^2)^2}\|d^l\|_{x^l}^2 \le 8\kappa^2\|d^l\|_{x^l}^2,$$

By replacing (28) in the above inequality we get

$$f(x^l) - f(x^*) < \frac{32\kappa^2\lambda_m^3}{(16\tilde{\lambda}_1^2 + L)^2}\Big(\frac{1}{2}\Big)^{2^{l-k+1}}.$$

Hence, in order to obtain a solution x^l, such that $f(x^l) - f(x^*) < \epsilon$, Newton-CG requires at most as many iterations as in the preamble of this theorem. The proof is complete.

The following theorem summarizes the complexity result of Newton-CG with backtracking line-search. The constants c_6, c_7 and c_8 in this theorem are defined in Theorems 2 and 3, respectively. *A worst-case iteration complexity result of the form in Theorem 4 below has been first obtained in [5], for an inexact Newton method based on cubic regularization.*

Theorem 4 *Starting from an initial point x^0, such that $\|d^0\|_{x^0} \geq \varpi$, Newton-CG with backtracking line-search requires at most*

$$K_3 = c_6 \log\left(\frac{f(x^0) - f(x^*)}{c_7 \varpi^2}\right) + \log_2 \log_2\left(\frac{c_8}{\epsilon}\right)$$

iterations to converge to a solution x^k, $k > 0$, of accuracy

$$f(x^k) - f(x^*) < \epsilon.$$

5 Worst-case iteration complexity of dcNCG with backtracking line-search

First, let us remind the reader that in Section 4 the Newton-CG algorithm has been analyzed independently of the doubly-continuation scheme, hence, for simplicity of notation, the parameters τ and μ have been dropped. In this section, it is necessary to introduce back the dependence on the parameters τ^l and μ^l for each continuation iteration l. In particular, when a reference to results from Section 4 is made, then the usual notation which includes these parameters will be used. Moreover, k denotes the iteration index of dcNCG algorithm, x^l is used to denote the initial solution of the l^{th} inner loop or an approximate minimizer of $f_{\tau^{l-1}}^{\mu^{l-1}}(x)$ for $l > 0$. For example, if the termination criterion in step 7 is satisfied for x^k and d^k at the $(l-1)^{th}$ continuation iteration, then $x^l := x^k$. Finally, z^l is the minimizer of $f_{\tau^l}^{\mu^l}(x)$ and $\bar{z}$ is the minimizer of $f_{\bar{\tau}}^{\bar{\mu}}(\bar{z})$.

We start by an intuitive explanation of the worst-case iteration complexity result. For this, the following summary of doubly-continuation scheme is helpful.

Summary of algorithm dcNCG

1: **Outer loop:** For $l = 0, 1, 2, \ldots, \vartheta$, produce sequences $\{\tau^l\}$ and $\{\mu^l\}$, where ϑ is the number of continuation iterations.
2: **Inner loop:** Approximately solve the subproblem

$$\text{minimize } f_{\tau^l}^{\mu^l}(x)$$

using Newton-CG with backtracking line-search.

From the summary of the continuation scheme it is clear that the total number of required steps is the sum of steps required for the approximate

solution of each subproblem. In the previous section, we established the worst-case iteration complexity of the inner loop, i.e. the approximate solution of the subproblem by using Newton-CG with backtracking line-search. In particular, to obtain $f_{\tau^l}^{\mu^l}(x^{l+1}) - f_{\tau^l}^{\mu^l}(z^l) < \epsilon$ for the l^{th} problem, then from Theorem 4 we have that at most K_3^l iterations of Newton-CG with backtracking line-search are required. Hence, if every l^{th} problem is solved to accuracy $f_{\tau^l}^{\mu^l}(x^{l+1}) - f_{\tau^l}^{\mu^l}(z^l) < \epsilon$ then the worst-case iteration complexity of the continuation scheme is $\sum_{l=1}^{\vartheta} K_3^l$. However, it would be preferable to have a bound of this result independent of the indexing l. For this reason, in this section we will show that

$$\sum_{l=1}^{\vartheta} K_3^l \leq \vartheta \max_{0\leq l\leq \vartheta} K_3^l < \vartheta K_4,$$

where K_4 is an upper bound of the worst-case K_3^l $\forall l$. For convenience, let us explicitly state K_3^l,

$$K_3^l = c_6^l \log\Big(\frac{f_{\tau^l}^{\mu^l}(x^l) - f_{\tau^l}^{\mu^l}(z^l)}{c_7^l(\varpi^l)^2}\Big) + \log_2\log_2\Big(\frac{c_8^l}{\epsilon}\Big), \tag{29}$$

where

$$c_6^l = \frac{2\Big(\frac{\tau^l}{\mu^l} + \lambda_1\Big)^3}{(1-\eta^2)^2\lambda_m^3 c_2 c_3}, \quad c_7^l = \frac{\lambda_m^2}{8\Big(\frac{\tau^l}{\mu^l} + \lambda_1\Big)^2},$$

$$c_8^l = \frac{32\Big(\frac{\tau^l}{\mu^l} + \lambda_1\Big)^2\lambda_m}{\Big(16\Big(\frac{\tau^l}{\mu^l} + \lambda_1\Big)^2 + \Big(\frac{\tau^l}{(\mu^l)^2} + L_\varphi\Big)\Big)^2}$$

and c_2 and c_3 are constants of the backtracking line-search algorithm, which do not depend on the continuation iteration index l. Moreover, according to Corollary 1, the region of fast convergence of Newton-CG for the l^{th} subproblem is defined for $\|d^k\|_{x^k,l} < \varpi^l$, where $\varpi^l \leq c_5^l$ and

$$c_5^l = \min\Big\{3(1-2c_2)\frac{\lambda_m^{\frac{3}{2}}}{\Big(\frac{\tau^l}{(\mu^l)^2} + L_\varphi\Big)}, \frac{\lambda_m^{\frac{3}{2}}}{16\Big(\frac{\tau^l}{\mu^l} + \lambda_1\Big)^2 + \Big(\frac{\tau^l}{(\mu^l)^2} + L_\varphi\Big)}\Big\}.$$

In order to obtain the upper bound K_4 one needs to calculate upper bounds for the quantities c_6^l, c_8^l and $f_{\tau^l}^{\mu^l}(x^l) - f_{\tau^l}^{\mu^l}(z^l)$, and a lower bound for c_7^l over all l. Additionally, a lower bound of the quantity c_5^l is required, which will be used to obtain a tighter upper bound for ϖ^l over all l, as required by Corollary 1. The following remark will be used for this purpose.

Remark 1 The parameters β_1, β_2 and τ^0, μ^0 are set such that $0 < \beta_1 \leq \beta_2 < 1$ and $\lceil \log_{\beta_2} \frac{\bar{\mu}}{\mu^0} \rceil < \lceil \log_{\beta_1} \frac{\bar{\tau}}{\tau^0} \rceil$. Briefly, we require that the sequence of iterates $\{\mu^l\}$ produced by (24) converges to $\bar{\mu}$ before the sequence $\{\tau^l\}$ converges to $\bar{\tau}$,

although, $\{\tau^l\}$ has a faster or similar rate of convergence. Then, according to rule (24) the sequences $\{\beta_1^l\}$ and $\{\beta_2^l\}$ are generated for which $0 < \beta_1^l \leq \beta_2^l \leq 1$ and

$$\tau^{l+1} = \beta_1^l \tau^l \quad \text{and} \quad \mu^{l+1} = \beta_2^l \mu^l. \tag{30}$$

Even though this assumption is not necessary for convergence of the doubly-continuation scheme, it facilitates the proof of monotonic decrease of $f_{\tau^l}^{\mu^l}(x)$ $\forall l$*, and it simplifies the calculation of some non-essential results.*

Using Remark 1 it is easy to find upper bounds for the quantities c_6^l, c_8^l and lower bounds for c_5^l and c_7^l. In particular, from Remark 1 we have $\frac{\bar{\tau}}{\bar{\mu}} \leq \frac{\tau^l}{\mu^l} \leq \frac{\tau^0}{\mu^0}$ $\forall l$. Hence, the quantities c_5^l, c_6^l, c_7^l, c_8^l are replaced with

$$\tilde{c}_5 = \min\Big\{3(1-2c_2)\frac{\lambda_m^{\frac{3}{2}}}{\Big(\frac{\tau^0}{\mu^0\bar{\mu}} + L_\varphi\Big)}, \frac{\lambda_m^{\frac{3}{2}}}{16\Big(\frac{\tau^0}{\mu^0} + \lambda_1\Big)^2 + \Big(\frac{\tau^0}{\mu^0\bar{\mu}} + L_\varphi\Big)}\Big\},$$

$$\tilde{c}_6 = \frac{2\Big(\frac{\tau^0}{\mu^0} + \lambda_1\Big)^3}{(1-\eta^2)^2\lambda_m^3 c_2 c_3}, \quad \tilde{c}_7 = \frac{\lambda_m^2}{8\Big(\frac{\tau^0}{\mu^0} + \lambda_1\Big)^2},$$

and

$$\tilde{c}_8 = \frac{32\Big(\frac{\tau^0}{\mu^0} + \lambda_1\Big)^2 \lambda_m}{\Big(16\Big(\frac{\bar{\tau}}{\bar{\mu}} + \lambda_1\Big)^2 + \Big(\frac{\bar{\tau}}{\bar{\mu}\mu^0} + L_\varphi\Big)\Big)^2}.$$

Having a lower bound of c_5^l $\forall l$, we can tighten the conditions of Corollary 1, which are replaced with

$$\|d^k\|_{x^k,l} < \tilde{\varpi} \quad \text{and} \quad \tilde{\varpi} \leq \tilde{c}_5 \quad \forall l. \tag{31}$$

Another consequence of Remark 1 is the following lemma.

Lemma 13 *For any* x*, if Remark 1 is satisfied, then*

$$f_{\tau^0}^{\mu^0}(x) > f_{\tau^1}^{\mu^1}(x) \quad \textit{and} \quad f_{\tau^{l-1}}^{\mu^{l-1}}(x) > f_{\tau^l}^{\mu^l}(x) \quad \forall l > 0.$$

Proof For (30) we can rewrite functions $f_{\tau^1}^{\mu^1}(x)$ and $f_{\tau^l}^{\mu^l}(x)$ as $f_{\beta_1^0\tau^0}^{\beta_2^0\mu^0}(x)$ and $f_{\beta_1^{l-1}\tau^{l-1}}^{\beta_2^{l-1}\mu^{l-1}}(x)$, respectively. It is easy to show that $f_{\tau^0}^{\mu^0}(x) > f_{\beta_1^0\tau^0}^{\beta_2^0\mu^0}(x)$ and $f_{\tau^l}^{\mu^l}(x) > f_{\beta_1^{l-1}\tau^{l-1}}^{\beta_2^{l-1}\mu^{l-1}}(x)$. The proof is complete.

Therefore using Lemma 13, the fact that z^l is the minimizer of function $f_{\tau^l}^{\mu^l}(x)$ and x^0 is the initial point of the continuation scheme we get

$$f_{\tau^0}^{\mu^0}(x^0) > f_{\tau^0}^{\mu^0}(z^0) > f_{\tau^1}^{\mu^1}(z^0) > f_{\tau^1}^{\mu^1}(z^1) > \ldots > f_{\bar{\tau}}^{\bar{\mu}}(\bar{z}). \tag{32}$$

It remains now to find a bound for the distance $\rho^l = f_{\tau^l}^{\mu^l}(x^l) - f_{\tau^l}^{\mu^l}(z^l)$. For $l = 0$ using (32) we get

$$\rho^0 < f_{\tau^0}^{\mu^0}(x^0) - f_{\bar{\tau}}^{\bar{\mu}}(\bar{z}). \tag{33}$$

For $l > 0$ we proceed by adding and subtracting the term $f_{\tau^l}^{\mu^l}(z^{l-1})$ in ρ^l,

$$\rho^l = f_{\tau^l}^{\mu^l}(x^l) - f_{\tau^l}^{\mu^l}(z^{l-1}) + f_{\tau^l}^{\mu^l}(z^{l-1}) - f_{\tau^l}^{\mu^l}(z^l).$$

We shall analyze independently the following two quantities

$$\rho_1^l = f_{\tau^l}^{\mu^l}(x^l) - f_{\tau^l}^{\mu^l}(z^{l-1}) \text{ and } \rho_2^l = f_{\tau^l}^{\mu^l}(z^{l-1}) - f_{\tau^l}^{\mu^l}(z^l)$$

First we start with the bound on ρ_1^l. From strong convexity of $f_{\tau^l}^{\mu^l}(x)$, we have that

$$\begin{aligned}\rho_1^l &\leq \nabla f_{\tau^l}^{\mu^l}(x^l)^\intercal (x^l - z^{l-1}) - \frac{\lambda_m}{2}\|x^l - z^{l-1}\|^2 \\ &\leq \|\nabla f_{\tau^l}^{\mu^l}(x^l)\|\|x^l - z^{l-1}\| - \frac{\lambda_m}{2}\|x^l - z^{l-1}\|^2. \end{aligned} \tag{34}$$

It is obvious that we need tight bounds for $\|\nabla f_{\tau^l}^{\mu^l}(x^l)\|$ and $\|x^l - z^{l-1}\|$. From Remark 1, we distinguish two scenarios for the sequences $\{\beta_1^l\}$ and $\{\beta_2^l\}$, first $0 < \beta_1^l \leq \beta_2^l < 1$, second $0 < \beta_1^l \leq 1$ and $\beta_2^l = 1$. For the first case by using (30) and (19) in (34) we get

$$\rho_1^l \leq (\|\nabla f_{\tau^{l-1}}^{\mu^{l-1}}(x^l)\| + \tau^{l-1}\Big(2 - \frac{\log(\beta_2^l)}{1-\beta_2^l}\Big)\sqrt{m})\|x^l - z^{l-1}\| - \frac{\lambda_m}{2}\|x^l - z^{l-1}\|^2.$$

While if $0 < \beta_1^l \leq 1$ and $\beta_2^l = 1$, then using (30) and (20) in (34) we get

$$\rho_1^l \leq (\|\nabla f_{\tau^{l-1}}^{\mu^{l-1}}(x^l)\| + 2\tau^{l-1}\sqrt{m})\|x^l - z^{l-1}\| - \frac{\lambda_m}{2}\|x^l - z^{l-1}\|^2.$$

The upper bound of ρ^l based on these two cases is given by the first bound and by replacing τ^l with τ^0 and β_2^l with $\beta_2 = \min_l \beta_2^l$,

$$\rho_1^l \leq (\|\nabla f_{\tau^{l-1}}^{\mu^{l-1}}(x^l)\| + \tilde{\omega}\sqrt{m})\|x^l - z^{l-1}\| - \frac{\lambda_m}{2}\|x^l - z^{l-1}\|^2, \tag{35}$$

where $\tilde{\omega} = \tau^0\Big(2 - \frac{\log(\beta_2)}{1-\beta_2}\Big)$. Using (18), Lemmas 5 and 9 and setting $0 \leq \eta < 1$ we have that

$$\|\nabla f_{\tau^{l-1}}^{\mu^{l-1}}(x^l)\| \leq \frac{2\Big(\frac{\tau^{l-1}}{\mu^{l-1}} + \lambda_1\Big)}{1-\eta^2}\|d^l\| \leq \frac{2\Big(\frac{\tau^{l-1}}{\mu^{l-1}} + \lambda_1\Big)}{(1-\eta^2)\lambda_m^{\frac{1}{2}}}\|d^l\|_{x^l,l-1} \tag{36}$$

and

$$\|x^l - z^{l-1}\| \leq \frac{2}{\lambda_m}\|\nabla f_{\tau^{l-1}}^{\mu^{l-1}}(x^l)\| \leq \frac{4\Big(\frac{\tau^{l-1}}{\mu^{l-1}} + \lambda_1\Big)}{(1-\eta^2)\lambda_m^{\frac{3}{2}}}\|d^l\|_{x^l,l-1}. \tag{37}$$

By replacing (36) and (37) in (35) we get

$$\begin{aligned}\rho_1^l &\leq \Big(\frac{2\Big(\frac{\tau^{l-1}}{\mu^{l-1}}+\lambda_1\Big)}{(1-\eta^2)\lambda_m^{\frac{1}{2}}}\|d^l\|_{x^l,l-1}+\tilde{\omega}\sqrt{m}\Big)\frac{4\Big(\frac{\tau^{l-1}}{\mu^{l-1}}+\lambda_1\Big)}{(1-\eta^2)\lambda_m^{\frac{3}{2}}}\|d^l\|_{x^l,l-1}\\ &- \frac{8\Big(\frac{\tau^{l-1}}{\mu^{l-1}}+\lambda_1\Big)^2}{(1-\eta^2)\lambda_m^2}\|d^l\|_{x^l,l-1}^2\\ &= \frac{4\tilde{\omega}\sqrt{m}\Big(\frac{\tau^{l-1}}{\mu^{l-1}}+\lambda_1\Big)}{(1-\eta^2)\lambda_m^{\frac{3}{2}}}\|d^l\|_{x^l,l-1} \leq \frac{4\tilde{\omega}\sqrt{m}\Big(\frac{\tau^0}{\mu^0}+\lambda_1\Big)}{(1-\eta^2)\lambda_m^{\frac{3}{2}}}\|d^l\|_{x^l,l-1}.\end{aligned}$$

Notice that at termination of the $(l-1)^{th}$ inner loop of dcNCG algorithm, step 7, we have that $\|d^l\|_{x^l,l-1} \leq \epsilon^{l-1}$. By considering a sequence of termination tolerances $\epsilon^1 \leq \epsilon^0$ and $\epsilon^l \leq \epsilon^{l-1}$ $\forall l > 0$, then $\|d^l\|_{x^l,l-1} \leq \epsilon^0$, hence, we get

$$\rho_1^l \leq \frac{4\tilde{\omega}\sqrt{m}\Big(\frac{\tau^0}{\mu^0}+\lambda_1\Big)}{(1-\eta^2)\lambda_m^{\frac{3}{2}}}\epsilon^0. \tag{38}$$

Regarding ρ_2^l, using (32) we get

$$\rho_2^l < f_{\tau^0}^{\mu^0}(x^0) - f_{\bar{\tau}}^{\bar{\mu}}(\bar{z}), \tag{39}$$

which is a finite constant. We can now calculate the bound of $\rho^l = \rho_1^l + \rho_2^l$, for $l > 0$, through the bounds of ρ_1^l and ρ_2^l, (38) and (39), respectively,

$$\rho^l = \rho_1^l + \rho_2^l < \frac{4\tilde{\omega}\sqrt{m}\Big(\frac{\tau^0}{\mu^0}+\lambda_1\Big)}{(1-\eta^2)\lambda_m^{\frac{3}{2}}}\epsilon^0 + f_{\tau^0}^{\mu^0}(x^0) - f_{\bar{\tau}}^{\bar{\mu}}(\bar{z}) = \tilde{\rho}, \tag{40}$$

which holds if every x^l point satisfies $\|d^l\|_{x^l,l-1} \leq \epsilon^0$.

In the following theorem the worst-case iteration complexity of the doubly-continuation scheme is given. Let us remind the reader that in this theorem, the constants $\tilde{c}_5$, $\tilde{c}_6$, $\tilde{c}_7$, $\tilde{c}_8$ and $\tilde{\rho}$ have been defined in the beginning of this section, the constant $\tilde{\varpi}$ satisfies (31), $0 \leq \eta < 1$ is the tolerance of the termination criterion (21) of the CG algorithm.

Theorem 5 *If β_1, τ^0 are set according to Remark 1 and $\epsilon^1 \leq \epsilon^0$, $\epsilon^l \leq \epsilon^{l-1}$ $\forall l > 0$ in step 7 of dcNCG algorithm, then the dcNCG algorithm initialized at x^0 requires at most*

$$K_5 = \Big(\tilde{c}_6 \log\Big(\frac{\tilde{\rho}}{\tilde{c}_7\tilde{\varpi}^2}\Big) + \log_2\log_2\Big(\frac{\tilde{c}_8}{\tilde{\epsilon}}\Big)\Big)\Big\lceil \log_{\beta_1}\frac{\bar{\tau}}{\tau^0}\Big\rceil$$

iterations to converge at a solution $\bar{x}$ of accuracy

$$f_{\bar{\tau}}^{\bar{\mu}}(\bar{x}) - f_{\bar{\tau}}^{\bar{\mu}}(\bar{z}) < \tilde{\epsilon}, \quad \textit{where} \quad \tilde{\epsilon} = \frac{\lambda_m^2}{8\Big(\frac{\tau^0}{\mu^0}+\lambda_1\Big)^2}(\epsilon^0)^2.$$

Proof According to Remark 1 the number of continuation iterations is $\vartheta = \lceil \log_{\beta_1} \frac{\bar{\tau}}{\tau^0} \rceil$. A worst-case K_3^l over all l, denoted at the beginning of this section with K_4, is given by replacing the quantities ϖ^l, c_5^l, c_6^l, c_7^l, c_8^l and $f_{\tau^l}^{\mu^l}(x^l) - f_{\tau^l}^{\mu^l}(z^l)$ in (29) with appropriate bounds over all continuation iterations l, as it has been shown previously in this section. First, c_5^l, c_7^l and c_6^l, c_8^l are replaced with their lower and upper bounds $\tilde{c}_5$, $\tilde{c}_7$ and $\tilde{c}_6$, $\tilde{c}_8$, respectively, over all l. The conditions $\|d^k\|_{x^k,l} < \varpi^l$ and $\varpi^l \leq c_5^l$ $\forall l$, of Corollary 1, are replaced by the tighter conditions (31). For $l = 0$ we bound $f_{\tau^l}^{\mu^l}(x^l) - f_{\tau^l}^{\mu^l}(z^l)$ by (33), while for $l > 0$ we bound this distance by (40), which holds if $\|d^k\|_{x^k,l-1} \leq \epsilon^0$ in step 7 of dcNCG. To guarantee the previous condition, sufficient iterations of Newton-CG are performed at the continuation step $l-1$. In order to calculate a worst-case iteration complexity such that $\|d^k\|_{x^k,l-1} \leq \epsilon^0$ two cases are distinguished for the magnitude of ϵ^0, first $\epsilon^0 < \tilde{\varpi}$, second $\epsilon^0 \geq \tilde{\varpi}$. The worst case is given for $\epsilon^0 < \tilde{\varpi}$. If $l = 0$, according to Theorem 4, $\tilde{c}_6 \log\left(\frac{f_{\tau^0}^{\mu^0}(x^0) - f_{\bar{\tau}}^{\bar{\mu}}(\bar{z})}{\tilde{c}_7 \tilde{\varpi}^2}\right) + \log_2 \log_2 \left(\frac{\tilde{c}_8}{\tilde{\epsilon}^l}\right)$ iterations are needed for Newton-CG to converge to a solution x^{l+1} such that $f_{\tau^l}^{\mu^l}(x^{l+1}) - f_{\tau^l}^{\mu^l}(z^l) < \tilde{\epsilon}^l$ and $\|d^{l+1}\|_{x^{l+1},l} < \epsilon^0$, where $\tilde{\epsilon}^l = \frac{\lambda_m^2}{8\left(\frac{\tau^l}{\mu^l} + \lambda_1\right)^2}(\epsilon^0)^2$. The previous bound on $\|d^{l+1}\|_{x^{l+1},l}$ is obtained by using (18), Lemmas 5 and 9, which give us

$$\frac{\lambda_m^2}{8\left(\frac{\tau^l}{\mu^l} + \lambda_1\right)^2}\|d^{l+1}\|_{x^{l+1},l}^2 \leq f_{\tau^l}^{\mu^l}(x^{l+1}) - f_{\tau^l}^{\mu^l}(z^l) < \tilde{\epsilon}^l \quad \forall l.$$

Hence from (40) we get that $\rho^1 < \tilde{\rho}$. If the previous statement holds, then for $l = 1$, according to Theorem 4 and (40) by performing $\tilde{c}_6 \log\left(\frac{\tilde{\rho}}{\tilde{c}_7 \tilde{\varpi}^2}\right) + \log_2 \log_2 \left(\frac{\tilde{c}_8}{\tilde{\epsilon}^l}\right)$ iterations of Newton-CG, it is guaranteed that $\|d^{l+1}\|_{x^{l+1},l} < \epsilon^0$ and $\rho^2 < \tilde{\rho}$. In a similar way it is guaranteed that $\rho^l < \tilde{\rho}$ $\forall l > 0$. Additionally, $\rho^0 < \tilde{\rho}$, because the upper bound of ρ^0 in (33) is smaller than $\tilde{\rho}$, therefore $\rho^l < \tilde{\rho}$ $\forall l$. From Remark 1 we have $\frac{\bar{\tau}}{\bar{\mu}} \leq \frac{\tau^l}{\mu^l} \leq \frac{\tau^0}{\mu^0}$ $\forall l$, hence, $\tilde{\epsilon} \leq \tilde{\epsilon}^l$ $\forall l$. Notice that the condition $\|d^{l+1}\|_{x^{l+1},l} < \epsilon^0$ is preserved if we require accuracy $\tilde{\epsilon}$ for all l continuation iterations instead of $\tilde{\epsilon}^l$. We conclude that

$$\max_{0 \leq l \leq \vartheta} K_3^l \leq K_4 = \tilde{c}_6 \log\left(\frac{\tilde{\rho}}{\tilde{c}_7 \tilde{\varpi}^2}\right) + \log_2 \log_2 \left(\frac{\tilde{c}_8}{\tilde{\epsilon}}\right).$$

If K_4 iterations of Newton-CG are performed for every l^{th} continuation iteration, then $\|d^{l+1}\|_{x^{l+1},l} < \epsilon^0$ and $f_{\tau^l}^{\mu^l}(x^{l+1}) - f_{\tau^l}^{\mu^l}(z^l) < \tilde{\epsilon}$ $\forall l$. By multiplying K_4 with ϑ we obtain the result. The proof is complete.

6 Numerical experience

We illustrate the efficiency of dcNCG on synthetic ℓ_1-regularized Sparse Least-Squares (S-LS) problems and ℓ_1-regularized Logistic Regression (LR) problems. The algorithm dcNCG is compared with the following state-of-the-art first order methods.

- CDN (Coordinate Descent Newton) [48] is a coordinate descent method which calculates one-dimensional Newton directions. It has been shown in [48] that CDN is one of the most efficient implementations when the required accuracy is not very high, which is the case for the problems of our interest. This solver can be found as part of the LIBLINEAR package at `http://www.csie.ntu.edu.tw/~cjlin/liblinear/`. In this paper the version 1.7 of the LIBLINEAR package has been used. This is because the newest version 1.93 of LIBLINEAR package, does not offer the option to solve Logistic Regression problems with CDN algorithm. The version 1.7 can be downloaded from `http://www.csie.ntu.edu.tw/~cjlin/liblinear/oldfiles/`.
- RCDC (Randomized (block) Coordinate Descent Method) [35] is a random parallel coordinate descent method. The published implementation performs parallel coordinate updates, where the coordinates are chosen randomly. This method is well-known for exploiting separability of the problems, and its ability to solve huge-scale problems. This code can be downloaded at `http://code.google.com/p/ac-dc/`.

6.1 Implementation details

All solvers are C or $C++$ implementations. All experiments are performed on a Dell PowerEdge C6220 running Redhat Enterprise Linux with Quad 8 Core Intel Xeon (Sandybridge) processors running in 64bit mode. For ℓ_1-regularized LR problems the solvers CDN and dcNCG are serial implementations. For ℓ_1-regularized S-LS problems, RCDC as a parallel solver exploits 24 cores, whilst, for these problems we also exploit 24 cores by implementing a version of the dcNCG with parallel linear algebra, i.e $x^\intercal y$, Ax and $A^\intercal y$ operations are performed in parallel. Both parallel solvers are implemented using the openMP protocol, for details see `http://openmp.org/wp/`. Finally, for dcNCG a diagonal preconditioner of the form $P = diag(\nabla^2 f(x))$ is used for all experiments and preconditioned CG (PCG) is employed.

6.2 ℓ_1-Regularized Sparse Least-Squares

In this subsection we compare the dcNCG solver with RCDC. The comparison is made on problems for which

$$\varphi(x) = \frac{1}{2}\|Ax - b\|^2$$

in (1), where $x \in \mathbb{R}^m$, $b \in \mathbb{R}^n$, $A \in \mathbb{R}^{n \times m}$ with $n \geq m$.

6.2.1 Benchmark Generators

The data A and b are synthetically generated using two different generators. The first generator was introduced in [35] and published online at `http://code.google.com/p/ac-dc/`. This generator has been originally proposed in [29] for producing ℓ_1-regularized dense Least-Squares problems. The advantage of this generator is that given $\tau = 1$ in problem (1), the generator produces such A and b with a known minimizer x^* of problem (1), hence, with known optimal objective function value. Additionally, one can control the number of non-zero elements per column of matrix A. Let us state an important observation about this generator. It produces problems that on average the diagonal of the matrix $A^\intercal A$ has 99% of the mass of the whole matrix. The published implementation of RCDC algorithm exploits this fact by biasing each coordinate direction using information from the diagonal of matrix $A^\intercal A$. In particular every coordinate direction is obtained by solving exactly

$$d_i := \arg\min_p \quad |x_i + p_i| + \nabla\varphi_i(x)p_i + \frac{\sigma a_i^\intercal a_i}{2} p_i^2,$$

where $d_i, p_i \in \mathbb{R}$, $\varphi_i(x)$ is the i^{th} partial derivative of $\varphi(x)$ and a_i is the i^{th} column of matrix A, all other coordinates of the direction are set to zero, see Algorithm 4 in [35] for details. The constant $\sigma \approx 1$ will be defined later. It is clear that if the generated problem has massive diagonal elements in the Hessian matrix $A^\intercal A$, then, these directions are biased by nearly perfect partial second-order information. One could argue that for dcNCG this observed property could be exploited by using PCG with a diagonal preconditioner, $P = diag(A^\intercal A)$. However, this is not entirely correct. In particular, in our small scale experiments, which will be discussed later, we have observed that the condition number of the matrix $P^{-1}A^\intercal A$ is of order 1.0e+5 to 1.0e+7. Hence, diagonal preconditioning does not favour that much the dcNCG solver. Fortunately, we will show that dcNCG converges fast even when the termination criterion of PCG is set to $\eta = 0.9$. Problems which have Hessian with massive diagonal are trivial, therefore, we modify the generator such that the mass of the matrix $A^\intercal A$ is spread evenly on each elements. For such problems the approximation of the matrix $A^\intercal A$ with its diagonal or any diagonal preconditioner will not be as efficient. For simplicity we will refer to the first version of this generator as BG1.1 and to the second version as BG1.2.

The second generator, which we will denote by BG2, can be implemented by using standard MATLAB commands. The MATLAB command

$$A = \texttt{sprand}(m, n, \text{density}, \text{rc}),$$

which returns a sparse uniformly distributed random matrix A, with approximately density $\cdot\, m \cdot n$ nonzero elements randomly positioned, where $0 \leq$ density ≤ 1, and rc is a vector of length m which determines the singular

values of matrix A when $n \geq m$. One can create any optimal sparse solution x^* and obtain the vector $b = Ax^*$. Given A and b the optimal solution x^* can be reconstructed by solving problem (1) with τ set to a very small value, i.e. 1.0e–8. However, because the current implementation of RCDC only solves problem (1) with $\tau = 1$ we will set τ as such for both solvers, RCDC and dcNCG.

6.2.2 Termination criteria and parameter tuning

Regarding the generators BG1.1 and BG1.2, given $\tau = 1$ the optimal objective value f_τ^* of problem (1) is known. First the dcNCG solver is employed and an approximate optimal value of the objective function is found, denoted by $f_\tau(x)$. The optimal objective value is subtracted from the one obtained by dcNCG to get an accuracy measure, i.e. $f_\tau(x) - f_\tau^*$. For these problems, at the termination of dcNCG, the solver reported $f_\tau(x) - f_\tau^* = \mathcal{O}(1.0\text{e–}6)$. Then the RCDC algorithm is employed till the required accuracy drops below the one of the dcNCG solver. Since, for RCDC algorithm making function evaluations is prohibited, because it is considered by its authors as a very expensive operation, we do not include the CPU time of making such operations in the total CPU time.

The parameter tuning of dcNCG is as follows. Parameters $\bar{\mu}$ and μ^0 are set to 1.0e–6 and 1, respectively. Parameter τ^0 is set to $\|A^\intercal b\|_\infty$ and the initial solution x^0 is set to the zero vector; this setting warm-starts Newton-CG algorithm. Parameters β_1 and β_2 are set to 1.0e–1 and 3.0e–1, for BG1.1 and BG1.2, respectively. Parameter η is set to 0.9. The maximum number of backtracking line-search iterations is fixed to 20. The tolerance is set to ϵ^l =1.0e+6 for each subproblem (5). By setting the tolerance ϵ^l to such a large value, the dcNCG algorithm performs one inexact Newton direction for each l^{th} continuation iteration; as shown in the numerical results in this section this setting is sufficient for dcNCG to produce a highly accurate solution.

For problems generated by BG2, dcNCG is employed first. At termination of dcNCG, the solver returned a solution of accuracy $\|d\|_x = \mathcal{O}(1.0\text{e–}8)$ for the last sub-problem (5) of the algorithm. However, for these problems, given $\tau = 1$ the optimal objective function value is unknown. Hence, only the obtained approximate optimal objective value $f_\tau(x)$ of dcNCG is calculated. The RCDC algorithm is employed till the objective value at a particular iteration drops below $f_\tau(x)$.

For these problems, parameters $\bar{\mu}$ and μ^0 are set to 1.0e–6 and 1, respectively. Parameter τ^0 is set to $\|A^\intercal b\|_\infty$ and x^0 is set to the zero vector. Parameters β_1 and β_2 are set to the save values, 0.9 or 0.99 for well-conditioned and ill-conditioned problems, respectively. Parameter η is set to 1.0e–1. The maximum number of backtracking line-search iterations is fixed to 20. The tolerance is set to ϵ^l =1.0e+6 for each subproblem.

The parameter σ of the RCDC algorithm is set to $1 + 23\frac{\omega-1}{m-1}$, where ω is the maximum number of non-zero elements over all rows of matrix A.

6.2.3 Three Experiments

For the generators BG1.1 and BG1.2, two classes of experiments are presented, two experiments are performed per class, one for each generator. In the first class of experiments our aim is to gradually increase the density of the generated matrix A. By gradually, increasing the density of matrix A we expect that the sparsity of the Hessian matrix $A^\intercal A$ will be affected. It is interesting to find out how both solvers compared react to increased density of the Hessian matrix. For this experiment, we fix $m = 10^3$, $n = 2m$. The number of non-zeros per column of matrix A is varied from 1 to n with a step of plus 100. Ten trials are performed per generated experiment and average results are reported. Since, these problems can be dense, we were not able to solve large scale problems as will be done in the next experiment. The results are shown in Figure 2. In these figures, "sparsity ratio" is the number of non-zeros per column of matrix A over the number of rows n. Observe in Figures 2a and 2b that dcNCG was faster for both generators when the "sparsity ratio" was more than 15%. However, we will show in the second experiment that this is a very pessimistic scenario and dcNCG can be very competitive for smaller "sparsity ratio" when the size n is very large. Notice, in Figures 2c and 2e that generator BG1.1 indeed produces problems where the matrix $A^\intercal A$ has a massive diagonal, while, the condition number of $P^{-1}A^\intercal A$ is relatively high. In Figure 2f we can see that the property of massive diagonal of matrix $A^\intercal A$ has been relaxed, and in Figure 2d it is shown that the condition number of $P^{-1}A^\intercal A$ for BG1.2 is the same as BG1.1. It is important to mention that the performance of dcNCG was unaffected by the "sparsity ratio" of the problem or any diagonal property of matrix $A^\intercal A$, while, RCDC seems to be efficient only for problems that are very sparse and have matrix $A^\intercal A$ with massive diagonal.

In the second class of experiments for generators BG1.1 and BG1.2 our aim is to observe the efficiency of the solvers as the size of problem m increases. For this experiment the size m is varied from 2^{12} to 2^{26} with a step of times two. The number of rows is set to $n = 2m$ and the number of non-zeros per column of matrix A is set to 20. It is worth mentioning that this experiment favours the RCDC solver, since the "sparsity ratio" is extremely small. Hence, we expect that the problems are well-conditioned and have matrix $A^\intercal A$ with massive diagonal. As a consequence of this very small "sparsity ratio" these problems tend to be separable, a property which is exploited by RCDC, since the experiment is run on 24 cores. Despite this fact, it is shown that dcNCG is more efficient for the large-scale realistic problems generated by BG1.2. The results of these experiments are shown in Figure 3.

In the third experiment our aim is to observe the performance of the solvers on problems where the mass of the diagonal of matrix $A^\intercal A$ is small, the non-zero components of x^* are uniformly distributed random variables in (–1.0e+3, 1.0e+3) and the singular values of matrix A are uniformly distributed random variables in (0.1, 1.0e+χ), where $\chi = 1, 2, 3$, or one hundred singular values of matrix A are set to 1.0e+5 and the rest to 1.0e–1. For this purpose, it is nec-

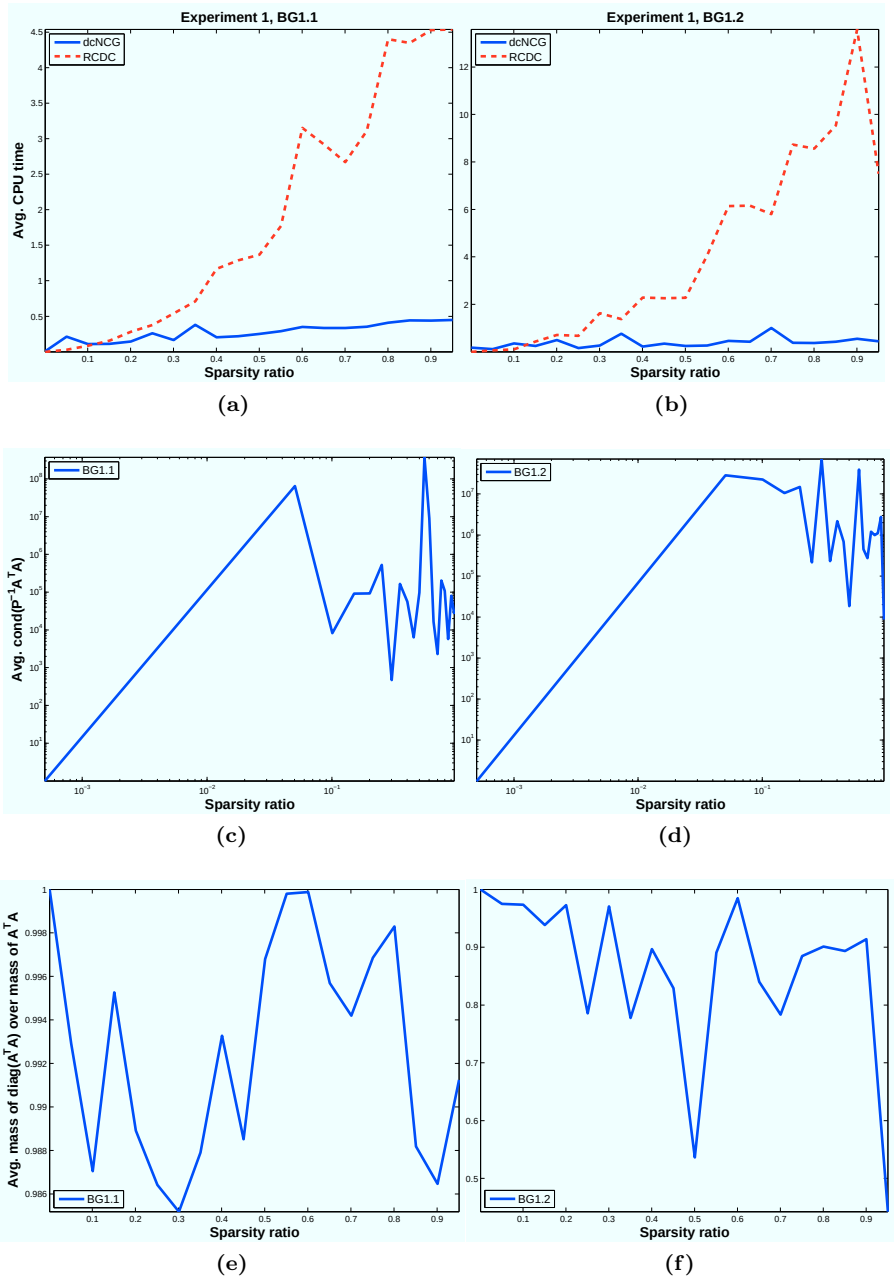


Fig. 2 Experiment 1: Increasing density of matrix A. The first column, from the left, of figures shows average results for BG1.1 and the second column shows average results for BG1.2. The first row of figures, from the top, shows the comparison of the solvers in terms of average CPU time against "sparsity ratio". The second row of figures, shows the average condition number of matrix $P^{-1}A^\intercal A$ against "sparsity ratio". The third row of figures, shows the average ratio of mass of the diagonal of matrix $A^\intercal A$ over the mass of the whole matrix against "sparsity ratio"

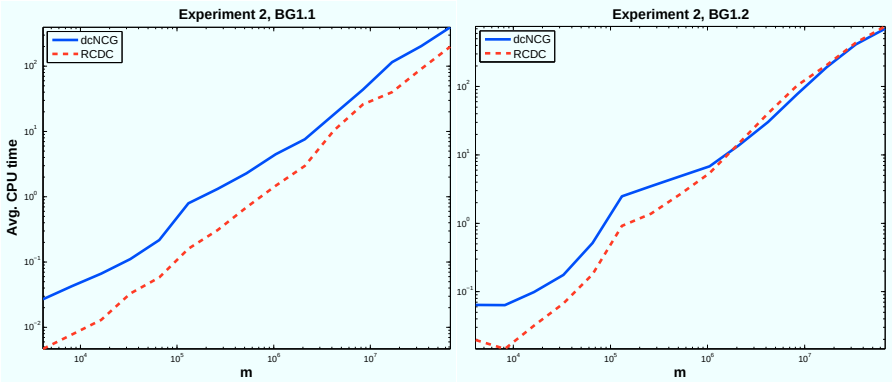


Fig. 3 Experiment 2, comparison of solvers for increasing dimension m, using both generators BG1.1 and BG1.2

essary to employ BG2, as it allows a direct control of all previously mentioned quantities. The parameters m and n are set to 2^{12} and $2m$, respectively, x^* has only 50 non-zero elements and density=5.0e–3. The results of these experiments are shown in Figures 4a, 4b, 4c and 4d, following the order of the different settings of singular values that were mentioned previously. Figure 4a corresponds to a problem with condition number of matrix $A^\intercal A$ of order 1.0e+4. Figure 4b corresponds to a problem with condition number of matrix $A^\intercal A$ of order 1.0e+8. Figure 4c corresponds to a problem with condition number of matrix $A^\intercal A$ of order 1.0e+12. Figure 4d corresponds to a problem with condition number of matrix $A^\intercal A$ of order 1.0e+12. As a last part of this experiment, corresponding to Figure 4e, we set one hundred singular values of matrix A to 1.0e+6 and the rest to 1.0e–1, the non-zero components of x^* are uniformly distributed random variables in (–1.0e+4, 1.0e+4), the parameters m and n are set to 2^{10} and $2m$, respectively, x^* has only 12 non-zero elements and density is kept to 5.0e–3. For this problem the condition number of matrix $A^\intercal A$ is of order 1.0e+14. For all problems, the density of matrix $A^\intercal A$ was approximately between 5.0e–3 and 1.0e–2, the mass of the diagonal of $A^\intercal A$ over the mass of the whole matrix was approximately between 3.0e–1 and 5.0e–1. Observe that despite the sparse nature of the problem, which is a property exploited by the parallelism of RCDC, the dcNCG solver was faster and more robust. For the experiments in Figures 4c and 4d the RCDC algorithm was terminated after ten million iterations, while for the experiment in Figure 4e the RCDC algorithm was terminated after one billion iterations which corresponds to more than 31 hours of CPU time.

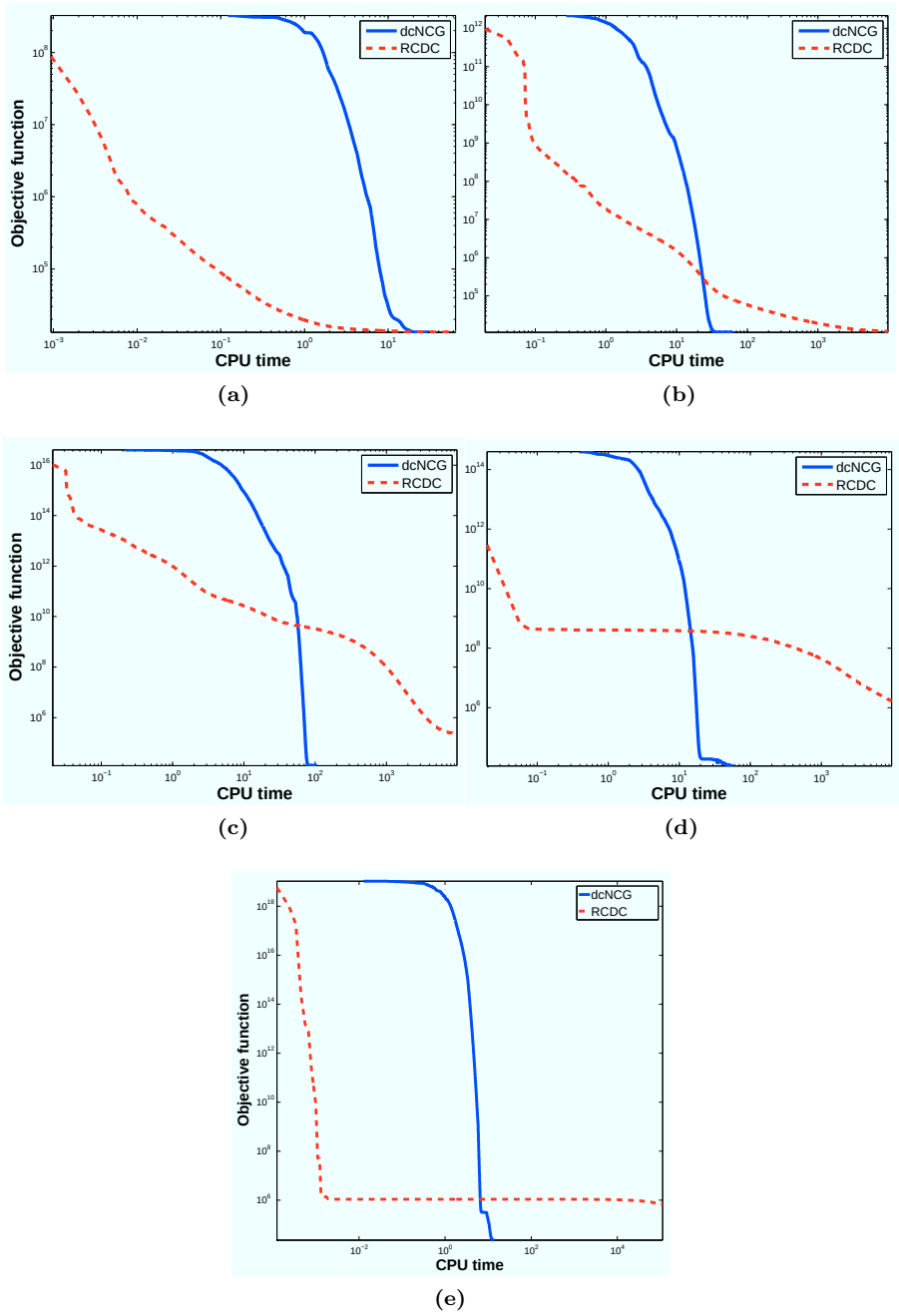


Fig. 4 Experiment 3: Increasing condition number of Hessian matrix $A^\intercal A$ and different clustering of eigenvalues

6.3 ℓ_1-Regularized Logistic Regression

In this subsection we compare the dcNCG solver with CDN on ℓ_1-regularized LR problems. For ℓ_1-regularized LR the function $\varphi(x)$ in (1) is set to

$$\varphi(x) = \sum_{i=1}^{n} \log(1 + e^{-y_i w^\intercal x_i}),$$

where $x_i \in \mathbb{R}^m \; \forall i = 1, 2, \ldots, n$ are called feature vectors and $y_i \in \{-1, +1\}$ are the corresponding labels. Such problems are used for training a linear classifier $w \in \mathbb{R}^m$. Although in Linear Support Vector Machine (LSVM) literature there are more alternatives for function $\varphi(x)$, in this section we choose LR because it is second-order differentiable. For more details about Support Vector Machine (SVM) problems we refer the reader to [49]. For ℓ_1-regularized LR a collection of LSVM problems can be downloaded from `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/` as part of the LIBSVM Data package. From this collection of problems we choose all 45 problems with binary labels. No normalization is performed on the feature vectors x_i, except if they have been normalized by default. Since many of the problems do not provide testing samples, we use the training data to measure the accuracy level of the obtained linear classifier w for all problems. We do not perform any cross-validation for the selection of parameter τ in (1), this parameter is set always to 1.

First, the problems are solved using CDN solver with default termination criteria. Then the same problem is solved using the dcNCG algorithm until approximately the same objective value to the one of CDN algorithm is obtained. All other parameters of the CDN algorithm are set to their default values. The parameter tuning of dcNCG is as follows. Parameter $\bar{\mu}$ is set to 1.0e–1 or 1.0e–2, parameter μ^0 is set to 1. Parameter τ^0 is set to $\|\varphi(0)\|_\infty$. Parameters β_1 and β_2 are set between 2.0e–1 and 7.0e–1. Parameter η is set to 1.0e–1. The maximum number of backtracking line-search iterations is fixed to 20. The tolerance is set to ϵ^l =1.0e+3 or 1.0e+4 for each subproblem (5).

The results for these 45 problems are shown in Table 1. The dcNCG solver was faster on 37 out of 45 problems. The problems for which dcNCG was faster are denoted by *italic* font. Both solvers had the same level of accuracy.

7 Conclusion

We propose and implement a new second-order method for strongly convex ℓ_1-regularized problems arising in many fields of optimization, i.e. Sparse Least-Squares and Machine Learning. The proposed method is a Newton Conjugate Gradients algorithm with backtracking line-search accelerated by a doubly-continuation scheme. The continuation scheme allows warm-start of Newton Conjugate Gradients and some control of the spectral properties of the Newton systems which are solved approximately at every iteration. Worst-case iteration complexity of Newton Conjugate Gradients with backtracking line-search

Table 1 Results on ℓ_1-regularized LR. The second and forth columns show the percentage of the training data that the obtained linear classifier w separated correctly. The third and fith columns show the CPU time, in seconds, that was required by the solvers for convergence. If the CPU time is less than 1.00e-02 sec. then zero sec. is reported. The problems for which dcNCG was faster are denoted by *italic* font

	dcNCG		CDN	
Problem	Accuracy	CPU time	Accuracy	CPU time
a1a	85.05 %	*2.00e−02*	85.05 %	9.00e−02
a2a	83.22 %	*3.00e−02*	83.40 %	1.80e−01
a3a	84.58 %	*3.00e−02*	84.71 %	2.20e−01
a4a	84.96 %	*6.00e−02*	84.96 %	2.00e−01
a5a	84.89 %	*8.00e−02*	84.91 %	2.00e−01
a6a	84.61 %	*1.70e−01*	84.65 %	2.90e−01
a7a	84.82 %	*2.50e−01*	84.82 %	3.20e−01
a8a	84.75 %	*4.10e−01*	84.75 %	4.90e−01
a9a	84.89 %	*6.10e−01*	84.91 %	1.01e+00
australian	86.23 %	1.00e−02	85.51 %	0.00e+00
breast-cancer	94.88 %	3.00e−02	94.73 %	1.00e−02
cod-rna	93.37 %	*3.80e−01*	93.39 %	8.58e+00
colon-cancer	100.0 %	2.00e−02	100.0 %	1.00e−02
covtype	75.61 %	*3.24e+01*	75.35 %	3.80e+02
diabetes	69.66 %	*0.00e+00*	69.66 %	4.00e−02
duke	100.0 %	*2.00e−02*	100.0 %	3.00e−02
epsilon	90.04 %	*2.34e+02*	90.06 %	1.59e+03
fourclass	73.55 %	*0.00e+00*	73.55 %	1.00e−02
german	77.80 %	*1.00e−02*	77.50 %	4.00e−02
gisette	100.0 %	*6.17e+00*	100.0 %	1.12e+01
heart	86.30 %	*0.00e+00*	85.93 %	3.00e−02
ijcnn1	92.46 %	*2.10e−01*	92.46 %	1.81e+00
ionosphere	84.90 %	*0.00e+00*	86.61 %	2.00e−02
kdda	91.16 %	2.66e+03	90.44 %	2.51e+03
kddb	91.84 %	4.99e+03	91.18 %	8.22e+02
leu	100.0 %	*2.00e−02*	100.0 %	3.00e−02
liver-disorders	69.57 %	*0.00e+00*	69.57 %	1.00e−02
mushrooms	99.95 %	*3.00e−02*	100.0 %	7.00e−02
news20	92.46 %	9.67e+00	92.96 %	3.25e+00
rcv1	96.62 %	*4.60e−01*	96.93 %	7.30e−01
real-sim	96.93 %	*1.08e+00*	97.42 %	1.83e+00
sonar	84.13 %	*0.00e+00*	85.10 %	4.00e−02
splice	83.00 %	*1.00e−02*	83.10 %	2.00e−01
svmguide1	84.40 %	*1.00e−02*	84.33 %	3.00e−02
svmguide3	78.44 %	*1.00e−02*	79.08 %	7.00e−02
url	99.49 %	1.62e+03	99.49 %	2.09e+02
w1a	90.03 %	*1.00e−02*	90.03 %	2.00e−02
w2a	90.00 %	*1.00e−02*	90.32 %	2.00e−02
w3a	90.68 %	*2.00e−02*	90.61 %	6.00e−02
w4a	90.58 %	*2.00e−02*	90.56 %	1.20e−01
w5a	90.25 %	*3.00e−02*	90.54 %	1.40e−01
w6a	90.33 %	*5.00e−02*	90.57 %	3.70e−01
w7a	90.59 %	*8.00e−02*	90.74 %	5.50e−01
w8a	90.55 %	*2.00e−01*	90.63 %	2.52e+00
webspam	98.29 %	3.02e+03	99.17 %	5.79e+02

and worst-case iteration complexity of the doubly continuation scheme are established.

Computational experience presented in this paper shows that although for ℓ_1-regularized problems the research community seems to favour first-order methods, a specialized second-order method is very competitive.

References

1. S. R. Becker, J. Bobin, and E. J. Candés. Nesta: A fast and accurate first-order method for sparse recovery. *SIAM J. Imaging Sciences*, 4(1):1–39, 2011.
2. S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press New York, NY, USA, 2004.
3. P. N. Brown and Y. Saad. Convergence theory of nonlinear Newton-krylov algorithms. *SIAM J. Optimization*, 4(2):297–330, 1994.
4. R. H. Byrd, G. M. Chin, J. Nocedal, and F. Oztoprak. A family of second-order methods for convex ℓ_1-regularized optimization. Unpublished: Optimization Center: Northwestern University, Tech Report, June 2012.
5. C. Cartis, N. I. M. Gould, and P. Toint. Evaluation complexity of adaptive cubic regularization methods for convex unconstrained optimization. Technical report, School of Mathematics, Edinburgh University, 2010.
6. K.-W. Chang, C.-J. Hsieh, and C.-J. Lin. Coordinate descent method for large-scale ℓ_2-loss linear support vector machines. *Journal of Machine Learning Research*, 9:1369–1398, 2008.
7. R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact Newton methods. *SIAM Journal on Numerical Analysis*, 19:400–408, 1982.
8. S. C. Eisenstat and H. F. Walker. Globally convergent inexact Newton methods. *SIAM J. Optimization*, 4(2):393–422, 1994.
9. S. C. Eisenstat and H. F. Walker. Choosing the forcing terms in an inexact Newton method. *SIAM J. Sci. Comput.*, 17(1):16–32, 1996.
10. M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse Problems. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):586–597, 2007.
11. K. Fountoulakis, J. Gondzio, and P. Zhlobich. Matrix-free interior point method for compressed sensing problems. *Technical Report ERGO 12-006*, 2012.
12. J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Machine Learning Research*, 9:627–650, 2008.
13. A. Galen and J. Gao. Scalable training of ℓ_1-regularization log-linear models. In *ICML*, 2007.
14. E. T. Hale, W. Yin, and Y. Zhang. A fixed-point continuation method for ℓ_1-regularized minimization with applications to compressed sensing. Technical Report TR07-07, 2007.
15. E. T. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for ℓ_1-minimization: methodology and convergence. *SIAM J. Optim.*, 19:1107–1130, 2008.
16. S. Hansen and J. Nocedal. Second-order methods for ℓ_1-regularized problems in machine learning. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2012*, pages 5237 –5240, march 2012.
17. R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.
18. M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6):409–436, 1952.
19. C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. *Proceedings of the 25th international conference on Machine Learning, ICML 2008*, pages 408–415, 2008.
20. C. T. Kelly. *Iterative Methods for Linear and Nonlinear Equations*. SIAM, Philadelphia, PA., 1995.
21. J. Kim and H. Park. Fast active-set-type algorithms for ℓ_1-regularized linear regression. In *In Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 2010.
22. S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale ℓ_1-regularized least squares. *IEEE Journal on Selected Topics in Signal Processing*, 1(4):606–617, 2007.
23. S.-I. Lee, H. Lee, P. Abbeel, and A. Y. Ng. Efficient ℓ_1-regularized logistic regression. In *In AAAI*, 2006.
24. C.-J. Lin, R. C. Weng, and S. S. Keerthi. Trust region Newton method for large-scale logistic regression. *Journal of Machine Learning Research*, 9:627–650, 2008.

25. D. M. Malioutov, M. Setin, and A. S. Willsky. Homotopy continuation for sparse signal representation. *IEEE International Conference on Acoustics, Speech and Signal Processing. Proceedings (ICASSP '05).*, 5:733–736, 2005.
26. S. G. Nash. A survey of truncated-Newton methods. *Journal of Computational and Applied Mathematics*, 124(1-2):45–59, 2000.
27. Y. Nesterov. *Introductory Lecture Notes On Convex Optimization. A Basic Course.* Kluver, Boston, 2004.
28. Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.
29. Y. Nesterov. Gradient methods for minimizing composite objective function. CORE Discussion Papers 2007076, September 2007.
30. Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming.* SIAM Studies in Applied Mathematics, 1994.
31. J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer, New York, 2006.
32. R. P. Pawlowski, J. N. Shadid, J. P. Simonis, and H. F. Walker. Globalization techniques for Newton-krylov methods and applications to the fully coupled solution of the navier-stokes equations. *Journal of Machine Learning Research*, 48(4):700–721, 2006.
33. D. Pu and W. Tian. Globally convergent inexact generalized Newton's method for nonsmooth equations. *Journal of Computational and Applied Mathematics*, 138(1):37–39, 2002.
34. J. Renegar. *A Mathematical View of Interior-Point Methods in Convex Optimization.* MOS-SIAM Series on Optimization, Cornell University, Ithaca, New York, 2001.
35. P. Richtárik and M. Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 2012.
36. M. Schmidt, G. Fung, and R. Rosales. Fast optimization methods for ℓ_1-regularization: A comparative study and two new approaches. In *In Proceedings of European Conference on Machine Learning*, pages 286–297, 2007.
37. S. Shalev-Shwartz and A. Tewari. Stochastic methods for ℓ_1-regularized loss minimization. *Journal of Machine Learning Research*, 12(4):1865–1892, 2011.
38. J. R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical report, Carnegie Mellon University Pittsburgh, PA, USA, 1994.
39. R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Roy. Statist. Soc.*, 58(1):267–288, 1996.
40. P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.
41. P. Tseng. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM J. Optim.*, 22:341–362, 2012.
42. P. Tseng and S. Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Math. Program., Ser. B*, 117:387–423, 2009.
43. Z. Wen, W. Yin, D. Goldfarb, and Y. Zhang. A fast algorithm for sparse reconstruction based on shrinkage, subspace optimization and continuation. *SIAM J. Sci. Comput.*, 32(4):1809–1831, 2010.
44. Z. Wen, W. Yin, H. Zhang, and D. Goldfarb. On the convergence of an active set method for ℓ_1-minimization. *Optimization Methods and Software*, 27(6):1127–1146, 2012.
45. S. J. Wright. Accelerated block-coordinate relaxation for regularized optimization. *SIAM Journal on Optimization*, 22(1):159–186, 2012.
46. S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.
47. T. T. Wu and K. Lange. Coordinate descent algorithms for lasso penalized regression. *The Annals of Applied Statistics*, 2(1):224–244, 2008.
48. G.-X. Yuan, K.-W. Chang, C.-J. Hsieh, and C.-J. Lin. A comparison of optimization methods and software for large-scale ℓ_1-regularized linear classification. *Journal of Machine Learning Research*, 11:3183–3234, 2010.
49. G. X. Yuan, C. H. Ho, and C. J. Lin. Recent advances of large-scale linear classification. *Proceedings of the IEEE*, 100(9):2584–2603, 2012.