Kanzow, Christian; Lechner, Theresa

Article — Published Version

Globalized inexact proximal Newton-type methods for nonconvex composite functions

Computational Optimization and Applications

Provided in Cooperation with:
Springer Nature

Suggested Citation: Kanzow, Christian; Lechner, Theresa (2020) : Globalized inexact proximal Newton-type methods for nonconvex composite functions, Computational Optimization and Applications, ISSN 1573-2894, Springer US, New York, NY, Vol. 78, Iss. 2, pp. 377-410, https://doi.org/10.1007/s10589-020-00243-6

This Version is available at:
https://hdl.handle.net/10419/288442

Computational Optimization and Applications (2021) 78:377–410
https://doi.org/10.1007/s10589-020-00243-6

Globalized inexact proximal Newton-type methods for nonconvex composite functions

Christian Kanzow[1] · **Theresa Lechner**[1]

Received: 27 February 2020 / Accepted: 26 October 2020 / Published online: 16 November 2020

Abstract
Optimization problems with composite functions consist of an objective function which is the sum of a smooth and a (convex) nonsmooth term. This particular structure is exploited by the class of proximal gradient methods and some of their generalizations like proximal Newton and quasi-Newton methods. The current literature on these classes of methods almost exclusively considers the case where also the smooth term is convex. Here we present a globalized proximal Newton-type method which allows the smooth term to be nonconvex. The method is shown to have nice global and local convergence properties, and some numerical results indicate that this method is very promising also from a practical point of view.

1 Introduction

In this paper, we deal with the composite problem

$$\min_{x\in\mathbb{R}^n} \psi(x) := f(x) + \varphi(x), \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is (twice) continuously differentiable and $\varphi : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is convex, proper, and lower semicontinuous (lsc). In this formulation, the objective function ψ is neither convex nor smooth, so it covers a wide class of applications as described below. Since φ is allowed to take the value $+\infty$, (1) also comprises constrained problems on convex sets.

✉ Christian Kanzow
kanzow@mathematik.uni-wuerzburg.de

Theresa Lechner
theresa.lechner2@mathematik.uni-wuerzburg.de

[1] University of Würzburg, Institute of Mathematics, Emil-Fischer-Str. 30, 97074 Würzburg, Germany

1.1 Background

Optimization problems in the form (1) arise in many applications in statistics, machine learning, compressed sensing, and signal processing.

Common applications are the lasso [42] and related problems, where the function f represents a smooth loss function such as the quadratic loss $f(x) := \|Ax - b\|_2^2$ or the logistic loss $f(x) := \frac{1}{m}\sum_{i=1}^m \log\left(1 + \exp(a_i^T x)\right)$ for some given data $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, and $a_i \in \mathbb{R}^n$ for $i = 1, \ldots, m$. A convex regularizer φ is added to involve some additional constraints or to control some sparsity. Typical regularizers are the ℓ_1- and ℓ_2-norm, a weighted ℓ_1-norm $\varphi(x) := \sum_{i=1}^n \omega_i |x_i|$ for some weights $\omega_i > 0$, or the total variation $\varphi(x) = \|\nabla x\| := \sum_{i=1}^{n-1} |x_{i+1} - x_i|$. Loss problems are typically used to reconstruct blurred or incomplete data or to classify data.

Another type of application are inverse covariance estimation problems [3, 46]. The aim of this problem class is to find the (sparse) inverse covariance matrix of a probability distribution of identically and independently distributed samples. For further applications, where the function f is assumed to be convex, we refer to the list given by Combettes and Ways [17] and references therein. Further problems in the form (1) are constrained problems [10] arising in the above mentioned fields.

Nonconvex applications occur, e.g., in inverse problems, where given data are not related linearly or are perturbed with non Gaussian errors as student's t-regression [1], see also Sects. 5.2 and 5.4; cf. the list by Bonettini et al. [9] for more examples of problems of this type.

1.2 Description of the Method

In every step of the proximal Newton-type method, we (inexactly) solve the problem

$$\arg\min_y \left\{ f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T H (y - x) + \varphi(y) \right\} \tag{2}$$

for some $x \in \mathbb{R}^n$ and a given matrix H which is either equal to the Hessian $\nabla^2 f(x)$ or represents a suitable approximation of the exact Hessian. The advantage of using proximal Newton-type steps that take into account second order information of f is that, similar to smooth Newton-type methods, one can prove fast local convergence. However, they are only well-defined for convex f and the convergence theorems typically require some strong convexity assumption.

In contrast, proximal gradient methods perform a backward step using only first order information of f. This means that (2) is solved for some positive definite $H \in \mathbb{R}^{n\times n}$, which is usually a fixed multiple of the identity matrix. The method can therefore be shown to converge globally in the sense that every accumulation point of a sequence generated by this method is a stationary point of ψ, but it is not possible to achieve fast local convergence results.

In this paper, we take into account the advantages of both methods and combine them to get a globalized proximal Newton-type method. Since the proximal Newton-type update is preferable, we try to solve the corresponding subproblem and use

a novel descent condition to control whether the current iterate is updated with its solution or a proximal gradient step is performed. To achieve global convergence, we further add an Armijo-type line search.

As the computation of the Newton-type step defined in (2) can be expensive, our convergence theory allows some freedom in the choice of the matrices H, in particular, one can use quasi-Newton or limited memory quasi-Newton matrices.

1.3 Related work

The original proximal gradient method was introduced by Fukushima and Mine [22]. It may be viewed as a special instance of the method described in Tseng and Yun [44], which utilizes a block separable structure of φ and performs block wise descent. Numerous authors [24, 35, 45] deal with acceleration techniques whereby all of them require the Lipschitz continuity of the gradient ∇f. Further methods [6, 39] also assume that f is convex.

In an intermediate approach between proximal Newton and proximal gradient methods, referred to as variable metric proximal gradient methods, the matrix H in (2) does not need to be a multiple of the identity matrix, but is still positive definite, uniformly bounded, and does not necessarily contain second order information of f. Various line search techniques and inexactness conditions on the subproblem solution can be applied [7–9, 13, 21, 23, 26, 27, 40, 41] to prove global convergence. These references include fast local convergence results for the case that H is replaced by the Hessian of f or some approximation and a suitable boundedness condition holds.

In Lee, Sun, and Saunders [27] a generic version of the proximal Newton method is presented and several convergence results based on the exactness of the subproblem solutions and the Hessian approximation are stated. For the local convergence theory, they need strong convexity of f. In Yue, Zhou, and So [47], an inexact proximal Newton method with regularized Hessian is presented which assumes f to be convex, but not strongly convex, and an error bound condition. Their inexactness criterion is similar to ours. The authors in [28, 43] assume that f is convex and self-concordant and apply a damped proximal Newton method.

Bonettini et al. [8, 9] consider an inexact proximal gradient method with variable metric and an Armijo-type line search to solve problem (1). The structure of the method in [9] is similar to ours, but they use a different inexactness criterion, have no globalization and add an overrelaxation step to ensure convergence. The convergence theory covers global convergence and local convergence under the assumption that ∇f is Lipschitz continuous and ψ satisfies the Kurdyka-Łojasiewicz property.

A similar method with various line search criteria is introduced by Lee and Wright [26]. Their inexactness criterion is related to the one from Bonettini et al. Furthermore, they use a line search technique to update the matrix H in (2), if suitable descent is not achieved. Here, convergence rates are proven for nonconvex as well as for convex problems.

Further methods exist for the case where we can write $\varphi = \tilde{\varphi} \circ B$ for a linear mapping $B : \mathbb{R}^n \to \mathbb{R}^p$ and a convex function $\tilde{\varphi} : \mathbb{R}^p \to \mathbb{R}$. This formulation is

used if the proximity operator of $\tilde{\varphi}$ is easy to compute whereas the one of φ is not. In [15, 16, 29] fixed point methods are used to solve the problems under different assumptions, the reformulation into a constrained problem is applied in [2, 48].

Another class of methods to solve (1) are semismooth Newton methods. Patrinos, Stella, and Bemporad assume in [37] that f is convex and apply a semismooth Newton method combined with a line search strategy. The method MINFBE of Stella, Themelis, and Patrinos [41] is based on the same idea, but uses a different line search strategy, for which they can prove convergence under the assumption that ∇f is Lipschitz continuous. Furthermore, they state linear convergence for convex problems.

For strongly convex f with Lipschitz continuous gradient, Patrinos and Bemporad [36] state a semismooth Newton method that uses a globalization strategy similar to our method and applies a proximal gradient step if the given descent criterion does not hold. A semismooth Newton method with filter globalization is introduced by Milzarek and Ulbrich [32] for $\varphi(x) = \lambda\|x\|_1$ with some $\lambda > 0$ and adapted for arbitrary convex φ by Milzarek [31]. For the semismooth Newton update, they check a filter condition and, if it does not hold, a proximal gradient step with Armijo-type line search is performed.

1.4 Outline of the paper

This paper is organized as follows. First, we introduce the proximity operator with some properties, formulate the proximal gradient method, and state a convergence result in Sect. 2. The globalization of the proximal Newton-type method and its inexact variant is deduced in Sect. 3, where we also state some preliminary observations. In Sect. 4, we first prove global convergence under fairly mild assumptions, and then provide a fast local convergence result. We then consider the numerical behaviour of our method(s) on different classes of problems in Sect. 5, also including a comparison with several state-of-the-art solvers. We conclude with some final remarks in Sect. 6.

1.5 Notation

For $x = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$ and $J \subset \{1, \ldots, n\}$, the subvector $x_J \in \mathbb{R}^{|J|}$ consists of all elements x_i of x with $i \in J$. Furthermore, $\overline{\mathbb{R}} := \mathbb{R} \cup \{\infty\}$ is the set of extended real numbers. The set of all symmetric matrices in $\mathbb{R}^{n\times n}$ is denoted by $\mathbb{S}^n$, and the set of all symmetric positive definite matrices is abbreviated by $\mathbb{S}^n_{++}$. We write $H \succ 0$ or $H \succeq 0$ for $H \in \mathbb{R}^{n\times n}$ if H is positive definite or positive semidefinite, respectively. Analogously, we write $H \succ G$ or $H \succeq G$ for $G, H \in \mathbb{R}^{n\times n}$ if $H - G$ is positive (semi)definite. The standard inner product of $x, y \in \mathbb{R}^n$ is denoted by $\langle x, y\rangle := x^T y$. Finally, we write $\|x\|_H := \sqrt{x^T H x}$ for the norm induced by a given matrix $H \succ 0$.

2 The proximal gradient method

This section first recalls the definition and some elementary properties of the proximity operator, and then describes a version of the proximal gradient method which is applicable to possibly nonconvex composite optimization problems. Throughout this section, we assume that f is continuously differentiable and φ is proper, lsc, and convex.

2.1 The proximity operator

The proximity operator was introduced by Moreau [34] and turned out to be a very useful tool both from a theoretical and an algorithmic point of view. Here we restate only some of its properties, and refer to the monograph [4] by Bauschke and Combettes for more details.

For a positive definite matrix $H \in \mathbb{R}^{n\times n}$ and a convex, proper, and lsc function $\varphi : \mathbb{R}^n \to \overline{\mathbb{R}}$, the mapping

$$x \mapsto \operatorname{prox}_{\varphi}^{H}(x) := \arg\min_{y} \left\{ \varphi(y) + \frac{1}{2}\|y - x\|_H^2 \right\}$$

is called the *proximity operator* of φ with respect to H. Here, the minimizer $\operatorname{prox}_{\varphi}^{H}(x)$ is uniquely defined for all $x \in \mathbb{R}^n$ since the expression inside the arg min is a strongly convex function. If H is the identity matrix, we simply write $\operatorname{prox}_{\varphi}(x)$ instead of $\operatorname{prox}_{\varphi}^{I}(x)$.

Using Fermat's rule and the sum rule for subdifferentials, the definition of the proximity operator gives $p = \operatorname{prox}_{\varphi}^{H}(x)$ if and only if $0 \in \partial\varphi(p) + H(p - x)$, or equivalently

$$p \in x - H^{-1}\partial\varphi(p). \tag{3}$$

We next restate a result on the continuity of the proximity operator due to Milzarek [31, Corollary 3.1.4], which states that the proximity operator is continuous not only with respect to the argument, but also with respect to the positive definite matrix.

Lemma 2.1 *The proximity operator $(x, H) \mapsto \operatorname{prox}_{\varphi}^{H}(x)$ is Lipschitz continuous on every compact subset of $\mathbb{R}^n \times \mathbb{S}^n_{++}$, and continuous on $\mathbb{R}^n \times \mathbb{S}^n_{++}$.*

We call $x^* \in \operatorname{dom}\varphi$ a *stationary point* of the program (1) if $0 \in \nabla f(x^*) + \partial\varphi(x^*)$. Using [4, Proposition 17.14] and (3), we obtain the characterizations

$$\begin{aligned} x^* \text{ stationary point of (1)} &\iff -\nabla f(x^*) \in \partial\varphi(x^*) \\ &\iff \psi'(x^*;d) \geq 0 \text{ for all } d \in \mathbb{R}^n \\ &\iff x^* = \operatorname{prox}_{\varphi}^{H}\big(x^* - H^{-1}\nabla f(x^*)\big), \end{aligned} \tag{4}$$

where the last reformulation turns out to be independent of the particular matrix H.

2.2 Proximal gradient method

The proximal gradient method was introduced by Fukushima and Mine [22] as a generalization of the proximal point algorithm, which, in turn, was established by Rockafellar [38]. Note that the existing literature on the proximal gradient method usually assumes f to be smooth with a (globally) Lipschitz continuous gradient. In order to obtain complexity and rate of convergence results, additional assumptions, e.g. the convexity of f, are required, cf. Beck [5] for more details.

Here we present a version of the proximal gradient method which still has nice global convergence properties also in the case where f is only continuously differentiable (not necessarily convex and without assuming any Lipschitz continuity of the corresponding gradient mapping). The method itself is essentially known and may be viewed as a special instance of the method described in Tseng and Yun [44], see also the PhD Thesis by Milzarek [31]. This version differs from the original one in [22] and its variants considered for convex problems by using a different line search globalization strategy. The proximal gradient method described here plays a central role in the globalization of our proximal Newton-type method.

To motivate the proximal gradient method, let us first recall that the classical (weighted) gradient method for the minimization of a smooth objective function f first computes a minimizer d^k of the quadratic subproblem

$$\min_d f(x^k) + \nabla f(x^k)^T d + \frac{1}{2} d^T H_k d \tag{5}$$

for some $H_k \succ 0$, and then takes $x^{k+1} = x^k + t_k d^k$ for some suitable stepsize $t_k > 0$. Usually, H_k is chosen as a positive multiple of the identity matrix. For $H_k = I$, we get the method of steepest descent, hence d^k is given by $-\nabla f(x^k)$ in this case.

Next consider the composite optimization problem from (1). To solve this nonsmooth problem, we simply add the nonsmooth function to the argument of (5) and obtain the subproblem

$$\min_d f(x^k) + \nabla f(x^k)^T d + \frac{1}{2} d^T H_k d + \varphi(x^k + d). \tag{6}$$

Let $d^k = d_{H_k}(x^k)$ be a solution of this subproblem. The next iterate is then defined by $x^{k+1} := x^k + t_k d^k$ for a suitable stepsize $t_k > 0$. A simple calculation shows that the solution d^k of (6) is given by

$$d^k = \operatorname{prox}_{\varphi}^{H_k}\left(x^k - H_k^{-1} \nabla f(x^k)\right) - x^k. \tag{7}$$

We now state our proximal gradient method explicitly. The stepsize rule uses the expression

$$\Delta_k := \nabla f(x^k)^T d^k + \varphi(x^k + d^k) - \varphi(x^k) \tag{8}$$

for $k \in \mathbb{N}_0$, which is an upper bound of the directional derivative $\psi'(x^k, d^k)$, see Lemma 2.3. Occasionally, we write Δ instead of Δ_k, if it is computed in some variables x and d instead of x^k and d^k, respectively.

Algorithm 2.2 (Proximal Gradient Method)

(S.0) *Choose* $x^0 \in \mathrm{dom}\varphi$, $\beta, \sigma \in (0,1)$, *and set* $k := 0$.
(S.1) *Choose* $H_k \succ 0$ *and determine* d^k *as the solution of*

$$\min_d \nabla f(x^k)^T d + \frac{1}{2} d^T H_k d + \varphi(x^k + d).$$

(S.2) *If* $d^k = 0$: *STOP.*
(S.3) *Compute* $t_k = \max\{\beta^l : l = 0, 1, 2, \dots\}$ *such that* $\psi(x^k + t_k d^k) \le \psi(x^k) + t_k \sigma \Delta_k$.
(S.4) *Set* $x^{k+1} := x^k + t_k d^k$, $k \leftarrow k+1$, *and go to (S.1).*

The algorithm allows H_k to be any positive definite matrix. In general, it is chosen independently of the iteration and as a positive multiple of the identity matrix, because in that case the computation of the proximity operator is less costly, in some cases (depending on the mapping φ) even an explicit expression is known.

We now want to prove that Algorithm 2.2 is well-defined and justify the termination criterion. The analysis is mainly based on [31, 44]. Note that we assume implicitly that the algorithm does not terminate after finitely many steps.

We first give an estimate for the value of Δ, which is essentially [32, Lemma 3.5].

Lemma 2.3 *Let* $x \in \mathrm{dom}\varphi$, $H \in \mathbb{S}^n_{++}$ *be given, and set* $d := \mathrm{prox}^H_\varphi\left(x - H^{-1}\nabla f(x)\right) - x$, *cf.* (7). *Then the inequalities* $\psi'(x;d) \le \Delta \le -d^T H d$ *hold.*

Note that this result implies that Δ_k is always a negative number as long as d^k is nonzero.

The termination criterion in (S.2) is justified by (4). Thus, it ensures that the algorithm terminates in a stationary point of ψ. Together with the next result, it follows that Algorithm 2.2 is well-defined, which means, in particular, that the line search procedure in (S.3) always terminates after finitely many steps.

Corollary 2.4 *Algorithm 2.2 is well-defined, and we have* $\psi(x^{k+1}) < \psi(x^k)$ *for all* k.

Proof Consider a fixed iteration index k. Since, by assumption, the algorithm generates an infinite sequence, (S.2) yields $d^k \neq 0$ for all k. Thus, by Lemma 2.3, we have $\Delta_k < 0$. Using the first inequality in Lemma 2.3, we therefore obtain

$$\frac{\psi(x^k + t d^k) - \psi(x^k)}{t} \le \sigma \Delta_k$$

for all sufficiently small $t > 0$. Rearranging this inequality, we see that the step size rule (S.3) and, consequently, the whole algorithm is well-defined. Furthermore, using $\Delta_k < 0$ in (S.3) yields $\psi(x^{k+1}) = \psi(x^k + t_k d^k) \le \psi(x^k) + t_k \sigma \Delta_k < \psi(x^k)$, and this completes the proof. □

The following convergence result is a special case of [44, Theorem 1(e)].

Theorem 2.5 *Let $\{H_k\}_k \subset \mathbb{S}^n_{++}$ be a sequence such that there exist $0 < m < M$ with $mI \preceq H_k \preceq MI$ for all $k \in \mathbb{N}_0$. Then any accumulation point of a sequence generated by Algorithm 2.2 is a stationary point of ψ.*

Theorem 2.5 cannot be applied directly in order to verify global convergence of our inexact proximal Newton-type method since only some of the search directions d^k are computed by a proximal gradient method, whereas other directions correspond to an inexact proximal Newton-type step. However, a closer inspection of the proof of [44, Theorem 1] yields that the following slightly stronger convergence result holds.

Remark 2.6 An easy consequence of the proof of Theorem 2.5, cf. [44], is the following more general result: Let $\{x^k\}$ be a sequence such that $x^{k+1} = x^k + t_k d^k$ holds for all k with some search directions $d^k \in \mathbb{R}^n$ (not necessarily generated by a proximal gradient step) and a stepsize $t_k > 0$. Assume further that $\psi(x^{k+1}) \le \psi(x^k)$ holds for all k. Let $\{x^k\}_K$ be a convergent subsequence of the given sequence such that the search directions $d^k = d_{H_k}(x^k)$ are obtained by proximal gradient steps for all $k \in K$, where $mI \preceq H_k \preceq MI$ $(0 < m \le M)$, and the corresponding step sizes $t_k > 0$ are determined by the Armijo-type rule from (S.3). Then the limit point of the subsequence $\{x^k\}_K$ is still a stationary point of ψ. ◊

3 Globalized inexact proximal Newton-type method

Let us start with the derivation of our globalized inexact proximal Newton-type method. To this end, let us first assume that H_k stands for the exact Hessian $\nabla^2 f(x^k)$ (later H_k will be allowed to be an approximation of the Hessian only).

In smooth optimization, one step of the classical version of Newton's method for minimizing a function $f : \mathbb{R}^n \to \mathbb{R}$ consists in finding a solution of $H_k(x - x^k) = -\nabla f(x^k)$. This is equivalent (assuming H_k being positive definite for the moment) to solve the problem $\min_x f_k(x)$, where

$$f_k(x) := f(x^k) + \nabla f(x^k)^T(x - x^k) + \tfrac{1}{2}(x - x^k)^T H_k(x - x^k) \tag{9}$$

is a quadratic approximation of f at the current iterate x^k. To solve this problem inexactly, one often uses the criterion

$$\|\nabla f_k(x)\| \le \eta_k \|\nabla f(x^k)\| \tag{10}$$

for some $\eta_k \in (0, 1)$.

Now we adapt this strategy to the nonsmooth problem (1). In this case, the objective function is $f + \varphi$, and the corresponding approximation we use is

$$\psi_k(x) := f_k(x) + \varphi(x) = f(x^k) + \nabla f(x^k)^T(x - x^k) + \tfrac{1}{2}(x - x^k)^T H_k(x - x^k) + \varphi(x). \tag{11}$$

In view of (4), we may view

$$F(x) := x - \operatorname{prox}_{\varphi}(x - \nabla f(x)) \tag{12}$$

as a replacement for the derivative of the objective function since $F(x) = 0$ if and only if x is a stationary point of ψ.

Since ψ_k is another function of the form (1), one can use the same idea to replace the derivative of ψ_k by

$$F^k(x) := x - \operatorname{prox}_{\varphi}\big(x - \nabla f_k(x)\big) = x - \operatorname{prox}_{\varphi}\big(x - (\nabla f(x^k) + H_k(x - x^k))\big).$$

This observation motivates to replace the inexactness criterion (10) by a condition like $\|F^k(x)\| \le \eta_k \|F(x^k)\|$ for some $\tau > 0$ and $\eta_k \ge 0$, see [13, 27].

Note that the methods of Bonettini et al. [9] and Lee and Wright [26] use a different inexactness criterion considering the value of the difference $\psi_k(x) - \psi_k(x^{*,k})$ of the function values of ψ_k, where $x^{*,k}$ is an exact minimizer of ψ_k. In contrast, our criterion originates directly from the smooth Newton method and considers a different optimality criterion based on the distance of the point x itself from being a solution of the subproblem, not the distance of the function values.

The main idea of our globalized proximal Newton-type method is now similar to a standard globalization of the classical Newton method for smooth unconstrained optimization problems: Whenever the proximal Newton-type direction exists and satisfies a suitable sufficient decrease condition, the proximal Newton-type direction is accepted and followed by a line search. Otherwise, a proximal gradient step is taken which always exists and guarantees suitable global convergence properties. The descent criterion used here is motivated by the condition in [18, 36]. The line search is based on the Armijo-type condition already used in the proximal gradient method and makes use of the same Δ_k that was already defined in (8). The exact statement of our method is as follows, where, now, we allow H_k to be an approximation of the Hessian of f at x^k.

Algorithm 3.1 (Globalized Inexact Proximal Newton-type Method (GIPN))

(S.0) *Choose initial parameters:* $x^0 \in \operatorname{dom}\varphi$, $\rho > 0$, $p > 2$, $\beta, \eta \in (0, 1)$, $\sigma \in (0, \frac{1}{2})$, $\zeta \in (\sigma, \frac{1}{2})$, $0 < c_{\min} \le c_{\max}$, *and set* $k := 0$.

(S.1) *Choose* $H_k \in \mathbb{R}^{n \times n}$ *symmetric*, $\eta_k \in [0, \eta)$ *and compute an inexact solution* $\hat{x}^k$ *of the subproblem* $\min_x \psi_k(x)$ *satisfying*

$$\|F^k(\hat{x}^k)\| \le \eta_k \|F(x^k)\| \qquad \text{and} \qquad \psi_k(\hat{x}^k) - \psi_k(x^k) \le \zeta \Delta_k, \tag{13}$$

and set $d^k := \hat{x}^k - x^k$. *If this is not possible or the condition*

$$\Delta_k \le -\rho \|d^k\|^p \tag{14}$$

is not satisfied, choose $c_k \in [c_{\min}, c_{\max}]$ *and determine* d^k *as the (unique) solution of*

$$\min_d \nabla f(x^k)^T d + \frac{1}{2} c_k \|d\|^2 + \varphi(x^k + d). \tag{15}$$

(S.2) *If* $d^k = 0$: *STOP.*
(S.3) *Compute* $t_k = \max\{\beta^l \mid l = 0, 1, 2, \dots\}$ *such that* $\psi(x^k + t_k d^k) \le \psi(x^k) + \sigma t_k \Delta_k$.
(S.4) *Set* $x^{k+1} := x^k + t_k d^k$, $k \leftarrow k + 1$ *and go to (S.1).*

Before we start to analyse the convergence properties of Algorithm 3.1, let us add a few comments regarding the proximal subproblems that we try to solve inexactly in (S.1). Since H_k is not necessarily positive definite, these subproblems are not guaranteed to have a solution. The same difficulty arises within the classical Newton method since, in the indefinite case, the quadratic subproblem (9) certainly has no minimizer. Nevertheless, the classical Newton method is often quite successful even if H_k is indefinite (at least during some intermediate iterations), and the Newton direction is usually well-defined because it just computes a stationary point of the subproblem (9) which exists also for indefinite matrices H_k. Here, the situation is similar since the conditions (13) only check whether we have an (inexact) stationary point (note that these conditions certainly hold for the exact solution of the corresponding subproblem, cf. [27, Proposition 2.4] for the second condition and note that $\zeta < \frac{1}{2}$). Moreover, the situation here is even better than in the classical case since the additional function φ may guarantee the existence of a minimum even for indefinite H_k (e.g. if φ has compact support as this occurs when φ is the indicator function of a bounded feasible set). We therefore believe that our proximal Newton-type direction does exist in many situations (otherwise we switch to the proximal gradient direction).

The properties of Algorithm 3.1 obviously depend on the choice of the matrices H_k and the degree of inexactness that is used to compute the inexact proximal Newton-type direction in (S.1). This degree is specified by the test in (13). The local convergence analysis requires some additional conditions regarding the choice of the sequence η_k, whereas the global convergence analysis depends only on the choice $\eta_k \in [0, \eta)$ for some given $\eta \in (0, 1)$ and does not need the second condition in (13). The condition in (14) is a sufficient decrease condition, with $\rho > 0$ typically being a small constant.

For our subsequent analysis, we set

$$\mathcal{K}_G := \{k : x^{k+1} \text{ was generated by the proximal gradient method}\},$$
$$\mathcal{K}_N := \{k : x^{k+1} \text{ was generated by the inexact proximal Newton-type method}\}.$$

The following result shows that the step size rule in (S.3) is well-defined and Algorithm 3.1 is a descent method.

Proposition 3.2 *Consider a fixed iteration k and suppose that $d^k \neq 0$. Then the line search in (S.3) is well-defined and yields a new iterate x^{k+1} satisfying $\psi(x^{k+1}) < \psi(x^k)$.*

Proof Since the proximal gradient method is well-defined by Corollary 2.4, the claim holds for $k \in \mathcal{K}_G$. Now, assume $k \in \mathcal{K}_N$, in which case (14) holds. Then $\Delta_k < 0$ and, therefore, the remaining part of the proof is identical to the one of Corollary 2.4. □

Proposition 3.2 requires $d^k \neq 0$. In view of the following result, this assumption can be stated without loss of generality. In particular, this result justifies our termination criterion in (S.2).

Lemma 3.3 *An iterate x^k generated by* GIPN *is a stationary point of ψ if and only if $d^k = 0$.*

Proof For $k \in \mathcal{K}_G$, the result follows from (4). Hence assume $k \in \mathcal{K}_N$, and let $d^k = 0$. This yields $\hat{x}^k = x^k$. Since $F^k(x^k) = F(x^k)$, condition (13) yields $\|F(x^k)\| \leq \eta_k \cdot \|F(x^k)\|$. As $\eta_k \in [0, 1)$, we get $F(x^k) = 0$ and x^k is a stationary point of ψ, using again (4). Conversely, assume that $d^k \neq 0$ for $k \in \mathcal{K}_N$. Then, analogous to Lemma 2.3, we get $\psi'(x^k; d^k) \leq \Delta_k \leq -\rho \|d^k\|^p < 0$. Hence x^k is not a stationary point of ψ. □

Altogether, the previous results show that Algorithm 3.1 is well-defined.

4 Convergence theory

In the following, we will prove global and local convergence results for algorithm GIPN. For this purpose, we assume that GIPN generates an infinite sequence and $d^k \neq 0$ holds for all $k \in \mathbb{N}$. The latter is motivated by Lemma 3.3.

4.1 Global convergence

The following is the main global convergence result for Algorithm 3.1. It guarantees stationarity of any accumulation point. Hence, if f is also convex, this implies that any accumulation point is a solution of the composite optimization problem from (1).

Theorem 4.1 *Consider Algorithm* GIPN *with a bounded sequence of matrices $\{H_k\}$. Then every accumulation point of a sequence generated by this method is a stationary point of ψ.*

Proof Let $\{x^k\}$ be a sequence generated by GIPN and $\{x^k\}_K$ a subsequence of $\{x^k\}$ converging to some x^*. If there are infinitely many indices $k \in K$ with $k \in \mathcal{K}_G$, i.e. the subsequence contains infinitely many iterates x^k such that x^{k+1} is generated by the proximal gradient method, Proposition 3.2 and the statement of Remark 2.6 yield that x^* is a stationary point of ψ.

Hence consider the case where all elements of the subsequence $\{x^{k+1}\}_K$ are generated by inexact Newton-type steps. Since $\{\psi(x^k)\}$ is monotonically decreasing by Proposition 3.2, $\{x^k\}_K$ converges to x^*, and since ψ is lsc, we get the convergence of the entire sequence $\{\psi(x^k)\}$ to some finite number ψ^*. The line search rule therefore yields

$$0 \leftarrow \psi(x^{k+1}) - \psi(x^k) \leq \sigma t_k \Delta_k < 0$$

and, hence, $t_k\Delta_k \to 0$ for $k \to \infty$. We claim that this implies $\{\|d^k\|\}_K \to 0$ (possibly after taking another subsequence). To verify this statement, we distinguish two cases:

Case 1: $\liminf_{k\in K} t_k > 0$. Then $\{\Delta_k\}_K \to 0$, and we therefore obtain $\{\|d^k\|\}_K \to 0$ in view of (14).

Case 2: $\liminf_{k\in K} t_k = 0$. Without loss of generality, assume $\lim_{k\in K} t_k = 0$. Then, for all $k \in K$ sufficiently large, the line search test is violated for the stepsize $\tau_k := t_k/\beta$. Using the monotonicity of the difference quotient of convex functions, cf. [4, Proposition 9.27], and the definition of Δ_k, we therefore obtain

$$\begin{aligned}\sigma\Delta_k &< \frac{\psi(x^k + \tau_k d^k) - \psi(x^k)}{\tau_k} \leq \frac{f(x^k + \tau_k d^k) - f(x^k)}{\tau_k} + \varphi(x^k + d^k) - \varphi(x^k) \\ &= \frac{f(x^k + \tau_k d^k) - f(x^k)}{\tau_k} - \nabla f(x^k)^T d^k + \Delta_k = \left(\nabla f(\xi^k) - \nabla f(x^k)\right)^T d^k + \Delta_k\end{aligned}$$

for all $k \in K$ sufficiently large, where the last expression uses the mean value theorem with some $\xi^k \in (x^k, x^k + \tau_k d^k)$. Reordering these expressions, we obtain

$$0 < -(1-\sigma)\Delta_k < \left(\nabla f(\xi^k) - \nabla f(x^k)\right)^T d^k.$$

Using (14) we get

$$(1-\sigma)\rho\|d^k\|^{p-1} \leq \|\nabla f(\xi^k) - \nabla f(x^k)\| \tag{16}$$

for all $k \in K$. Since $\{t_k\Delta_k\}_K \to 0$, it follows that $t_k\|d^k\|^p \to_K 0$ in view of (14). Using $p > 1$, this implies $\tau_k\|d^k\| \to_K 0$. Hence the right hand side of (16) converges to zero due to the uniform continuity of ∇f on compact sets. Consequently, (16) shows that $\|d^k\| \to_K 0$.

Therefore, $d^k \to_K 0$ holds in both cases. Since $x^k \to_K x^*$, the definition of d^k also implies $\hat{x}^k \to_K x^*$. Using the continuity of the proximity operator, we therefore get

$$F(x^k) \to_K x^* - \operatorname{prox}_\varphi(x^* - \nabla f(x^*))$$

and, since $\{H_k\}$ is bounded by assumption,

$$F^k(\hat{x}^k) \to_K x^* - \operatorname{prox}_\varphi(x^* - \nabla f(x^*)).$$

Since $\|F^k(\hat{x}^k)\| \leq \eta\|F(x^k)\|$ for all $k \in K$ in view of (13) and $\eta \in (0,1)$, taking the limit $k \to_K \infty$ therefore implies $x^* = \operatorname{prox}_\varphi(x^* - \nabla f(x^*))$, which is equivalent to x^* being a stationary point of ψ. □

Remark 4.2 Note that the proof of Theorem 4.1 only requires $p > 1$ and the first condition from (13). The second condition from (13) is only needed in the local convergence theory. ◊

4.2 Local convergence

We now turn to the local convergence properties of Algorithm 3.1. To this end, we assume that ψ is locally strongly convex in a neighbourhood of an accumulation point of a sequence of iterates and the sequence $\{H_k\}$ is bounded. Under these assumptions, we first prove the convergence of the complete sequence.

Theorem 4.3 *Consider Algorithm 3.1 with $\{H_k\}$ satisfying $MI \geq H_k \geq mI$ for all $k \in \mathbb{N}_0$ with suitable $M \geq m > 0$. Let x^* be an accumulation point of the generated sequence $\{x^k\}$ such that ψ is locally strongly convex in a neighbourhood of x^*. Then the whole sequence $\{x^k\}$ converges to x^*, and x^* is a strict local minimum of ψ.*

Proof In view of Theorem 4.1, every accumulation point of the sequence $\{x^k\}$ is a stationary point of ψ. Since ψ is locally strongly convex, x^* is the only stationary point in a suitable neighbourhood. Hence x^* is necessarily the only accumulation point of the sequence $\{x^k\}$ in this neighbourhood, and a strict local minimum of ψ. In order to verify the convergence of $\{x^k\}$, we therefore have to verify only the condition $\{\|x^{k+1} - x^k\|\}_K \to 0$ for any subsequence $\{x^k\}_K \to x^*$, cf. [33, Lemma 4.10].

Hence let $\{x^k\}_K$ denote an arbitrary subsequence converging to x^*. Since $\|x^{k+1} - x^k\| = t_k\|d^k\|$ for all $k \in \mathbb{N}$, it suffices to show $\{t_k\|d^k\|\}_K \to 0$ for $K \subset \mathcal{K}_G$ and $K \subset \mathcal{K}_N$. First, let $K \subset \mathcal{K}_N$. Then the statement is already shown in the proof of Theorem 4.1. On the other hand, if $K \subset \mathcal{K}_G$, the continuity of the solution operator in the proximal gradient method, see Lemma 2.1, yields $\{\|d^k\|\}_K \to 0$. The claim follows from $0 \leq t_k\|d^k\| \leq \|d^k\|$. □

Note that the assumption regarding the local strong convexity of ψ in a neighbourhood of x^* certainly holds if the Hessian $\nabla^2 f(x^*)$ is positive definite.

For the following analysis, we assume, in addition, that f is twice continuously differentiable and the sequence $\{H_k\}$ satisfies the Dennis-Moré condition [19]

$$\lim_{k\to\infty} \frac{\left\|\left(H_k - \nabla^2 f(x^*)\right)(\hat{x}^k - x^k)\right\|}{\|\hat{x}^k - x^k\|} = 0.$$

Under suitable assumptions, we expect the method to be locally superlinearly or quadratically convergent. The main steps into this direction are summarized in the following observations, which are partly taken from [47].

Proposition 4.4 *Consider Algorithm 3.1 with $\{H_k\}$ satisfying the Dennis-Moré condition and $MI \geq H_k \geq mI$ for all $k \in \mathbb{N}_0$ with suitable $M \geq m > 0$. Let x^* be a stationary point of ψ such that ψ is locally strongly convex in a neighbourhood of x^*. Then there exist constants $\varepsilon > 0$ as well as $C_1, C_2, \kappa_1, \kappa_2, \mu > 0$ such that, for any*

iterate $x^k \in B_\varepsilon(x^*)$, the following statements hold, where $\hat{x}^k_{ex}$ is the exact solution of the corresponding subproblem in (S.1) of Algorithm 3.1:

(a) $\left\|\hat{x}^k - \hat{x}^k_{ex}\right\| \le C_1 \eta_k \|F(x^k)\|$.
(b) $\left\|\hat{x}^k_{ex} - x^k\right\| \le \kappa_1 \|x^k - x^*\|$.
(c) $\mu\left\|\hat{x}^k_{ex} - x^*\right\| \le C_2 \eta_k \|F(x^k)\| + \left\|\left(H_k - \nabla^2 f(x^*)\right)(\hat{x}^k - x^k)\right\|$
$+ \left\|\nabla f(x^k) - \nabla f(x^*) - \nabla^2 f(x^*)(x^k - x^*)\right\|$.

Proof We verify each of the three statements separately, using possibly different values of ε.

(a) First, note that the function ψ_k is strongly convex and, therefore, has a unique minimizer. Thus, the exact solution $\hat{x}^k_{ex} = \operatorname{prox}_\varphi^{H_k}\left(x^k - H_k^{-1}\nabla f(x^k)\right)$ of the subproblem exists and hence guarantees that there is an inexact solution $\hat{x}^k$.

Since $F^k(\hat{x}^k) = \hat{x}^k - \operatorname{prox}_\varphi\left(\hat{x}^k - \nabla f_k(\hat{x}^k)\right)$, we obtain from (3) that

$$F^k(\hat{x}^k) - \nabla f_k(\hat{x}^k) \in \partial\varphi(\hat{x}^k - F^k(\hat{x}^k)).$$

The definition of ψ_k together with the subdifferential sum rule therefore implies

$$F^k(\hat{x}^k) + \nabla f_k\left(\hat{x}^k - F^k(\hat{x}^k)\right) - \nabla f_k(\hat{x}^k) \in \partial\psi_k\left(\hat{x}^k - F^k(\hat{x}^k)\right),$$

which is equivalent to

$$(I - H_k)F^k(\hat{x}^k) \in \partial\psi_k\left(\hat{x}^k - F^k(\hat{x}^k)\right). \tag{17}$$

Since ψ_k is strongly convex with modulus $m > 0$, its subdifferential is strongly monotone in this neighbourhood with the same modulus. Hence, using (17) together with $0 \in \partial\psi_k(\hat{x}^k_{ex})$, we get

$$\left\langle (I - H_k)F^k(\hat{x}^k), \hat{x}^k - F^k(\hat{x}^k) - \hat{x}^k_{ex}\right\rangle \ge m\left\|\hat{x}^k - F^k(\hat{x}^k) - \hat{x}^k_{ex}\right\|^2.$$

Applying the Cauchy-Schwarz inequality, this implies

$$\left\|\hat{x}^k - F^k(\hat{x}^k) - \hat{x}^k_{ex}\right\| \le \frac{1}{m}\left\|(I - H_k)F^k(\hat{x}^k)\right\| \le \frac{1}{m}(1 + M)\|F^k(\hat{x}^k)\|.$$

Using the inexactness criterion (13), we finally get, with $C_1 := (1 + M + m)/m$,

$$\begin{aligned}\|\hat{x}^k - \hat{x}^k_{ex}\| &\le \|\hat{x}^k - F^k(\hat{x}^k) - \hat{x}^k_{ex}\| + \|F^k(\hat{x}^k)\| \\ &\le \frac{1}{m}(1 + M)\|F^k(\hat{x}^k)\| + \|F^k(\hat{x}^k)\| \le C_1\eta_k\|F(x^k)\|.\end{aligned}$$

(b) Let $G(x, H) := x - \operatorname{prox}_\varphi^H\left(x - H^{-1}\nabla f(x)\right)$. By Lemma 2.1, G is Lipschitz continuous for $x \in B_\varepsilon(x^*)$ for some $\varepsilon > 0$ and $H \in \mathbb{S}^n_{++}$ with $mI \preceq H \preceq MI$ and $G(x^*, H) = 0$ for all such H by (4). Thus, there exists $\kappa_1 > 0$ (not depending on H_k) such that

$$\|\hat{x}^k_{ex} - x^k\| = \|G(x^k, H_k)\| = \|G(x^k, H_k) - G(x^*, H_k)\| \le \kappa_1\|x^k - x^*\|.$$

(c) The inequality holds trivially for $\hat{x}_{ex}^k = x^*$. Thus, assume $\hat{x}_{ex}^k \neq x^*$. First, note that (a) implies

$$\begin{aligned}
&\|(H_k - \nabla^2 f(x^*))(\hat{x}_{ex}^k - x^k)\| \\
&\quad \leq (M + \|\nabla^2 f(x^*)\|)\|\hat{x}_{ex}^k - \hat{x}^k\| + \|(H_k - \nabla^2 f(x^*))(\hat{x}^k - x^k)\| \\
&\quad \leq C_1(M + \|\nabla^2 f(x^*)\|)\eta_k\|F(x^k)\| + \|(H_k - \nabla^2 f(x^*))(\hat{x}^k - x^k)\|.
\end{aligned} \tag{18}$$

Since ψ is locally strongly convex in a neighbourhood of x^*, the subdifferential is strongly monotone, i.e. there exist $\varepsilon > 0$ and $\mu > 0$ such that

$$\langle x - y, \nabla f(x) + s(x) - \nabla f(y) - s(y)\rangle \geq 2\mu\|x - y\|^2$$

holds for all $x, y \in B_\varepsilon(x^*)$ and $s(x) \in \partial\varphi(x), s(y) \in \partial\varphi(y)$. Using the stationarity of x^* and $\hat{x}_{ex}^k$, we have $0 \in \nabla f(x^*) + \partial\varphi(x^*)$ and $0 \in \nabla f(x^k) + H_k(\hat{x}_{ex}^k - x^k) + \partial\varphi(\hat{x}_{ex}^k)$. Thus, also noting that $\hat{x}_{ex}^k$ eventually belongs to $B_\varepsilon(x^*)$ in view of part (b), we get

$$\begin{aligned}
2\mu\|\hat{x}_{ex}^k - x^*\|^2 &\leq \langle \nabla f(\hat{x}_{ex}^k) - \nabla f(x^k) - H_k(\hat{x}_{ex}^k - x^k), \hat{x}_{ex}^k - x^*\rangle \\
&= \langle \big(\nabla^2 f(x^*) - H_k\big)(x^k - \hat{x}_{ex}^k), x^* - \hat{x}_{ex}^k\rangle \\
&\quad + \langle \nabla f(x^k) - \nabla f(\hat{x}_{ex}^k) - \nabla^2 f(x^*)(x^k - \hat{x}_{ex}^k), x^* - \hat{x}_{ex}^k\rangle \\
&\leq \Big\|\big(\nabla^2 f(x^*) - H_k\big)(x^k - \hat{x}_{ex}^k)\Big\| \cdot \Big\|x^* - \hat{x}_{ex}^k\Big\| \\
&\quad + \|\nabla f(x^k) - \nabla f(x^*) - \nabla^2 f(x^*)(x^k - x^*)\| \cdot \|x^* - \hat{x}_{ex}^k\| \\
&\quad + \|\nabla f(x^*) - \nabla f(\hat{x}_{ex}^k) - \nabla^2 f(x^*)(x^* - \hat{x}_{ex}^k)\| \cdot \|x^* - \hat{x}_{ex}^k\|.
\end{aligned}$$

From (b) we get $\{\hat{x}_{ex}^k\} \to x^*$. Thus, by reducing $\varepsilon > 0$, if necessary, we get

$$\|\nabla f(x^*) - \nabla f(\hat{x}_{ex}^k) - \nabla^2 f(x^*)(x^* - \hat{x}_{ex}^k)\| \leq \mu\|x^* - \hat{x}_{ex}^k\|$$

from the twice differentiability of f. The assertion follows from dividing by $\|x^* - \hat{x}_{ex}^k\|$ and using (18). □

A suitable combination of the previous results leads to the following (global and) local convergence result for Algorithm 3.1.

Theorem 4.5 *Consider Algorithm 3.1 and assume that the sequence $\{H_k\}$ satisfies the assumptions from Proposition 4.4. Let x^* be an accumulation point of a sequence $\{x^k\}$ generated by Algorithm 3.1 such that ψ is locally strongly convex in a neighbourhood of x^*. Then the following statements hold:*

(a) *For all sufficiently large k, the search direction is attained by the inexact proximal Newton-type direction.*
(b) *For all sufficiently large k, the full step size $t_k = 1$ is accepted.*
(c) *If $\eta < \overline{\eta}$, the sequence $\{x^k\}$ converges linearly to x^*, where $\overline{\eta} = 1/((C_1 + \frac{1}{\mu}C_2)(L + 2))$ with C_1, C_2, μ from Proposition 4.4 and a local Lipschitz constant $L > 0$ of ∇f in a neighbourhood of x^*.*

(d) *If* $\{\eta_k\} \to 0$, *the sequence* $\{x^k\}$ *converges superlinearly to* x^*.

Proof Note that we know from Theorem 4.3 that x^* is both a stationary point and a strict local minimum of ψ, and that the whole sequence $\{x^k\}$ converges to x^*.

(a) Similar to the proof of Proposition 4.4, there exists a solution $\hat{x}^k$ of the subproblem $\min_x \psi_k(x)$ for all $k \in \mathbb{N}$. Let $\Delta_{k,N}$ be the Δ-function corresponding to the search direction $d_N^k := \hat{x}^k - x^k$, i.e. $\Delta_{k,N} := \nabla f(x^k)^T d_N^k + \varphi(x^k + d_N^k) - \varphi(x^k)$. Then the second condition in (13) is equivalent to

$$(1-\zeta)\Delta_{k,N} \le -\frac{1}{2}(d_N^k)^T H_k d_N^k,$$

which yields

$$\Delta_{k,N} \le -\tilde{c}\|d_N^k\|^2 \quad \text{for} \quad \tilde{c} := m/(2(1-\zeta)). \tag{19}$$

Since x^* is a stationary point of ψ, hence $F(x^*) = 0$, it follows from the continuity of F and the results in Proposition 4.4 (a) and (b) that

$$\|\hat{x}^k - \hat{x}_{ex}^k\| \le \frac{1}{2}\Big(\frac{\rho}{\tilde{c}}\Big)^{1/(2-p)}, \quad \|\hat{x}_{ex}^k - x^k\| \le \frac{1}{2}\Big(\frac{\rho}{\tilde{c}}\Big)^{1/(2-p)}$$

holds for all sufficiently large $k \in \mathbb{N}$. Combining these inequalities yields $\|d_N^k\| = \|\hat{x}^k - x^k\| \le (\rho/\tilde{c})^{1/(2-p)}$. We therefore get

$$\Delta_{k,N} \le -\tilde{c}\|d_N^k\|^2 = -\tilde{c}\|d_N^k\|^p \|d_N^k\|^{2-p} \le -\rho\|d_N^k\|^p.$$

Thus, the sufficient descent condition (14) is fulfilled and the search direction $d^k = d_N^k$ is obtained by the inexact proximal Newton-type method.

(b) Taylor expansion yields

$$\begin{aligned} f(\hat{x}^k) - f(x^k) &= \nabla f(x^k)^T(\hat{x}^k - x^k) + \frac{1}{2}(\hat{x}^k - x^k)^T \nabla^2 f(x^k)(\hat{x}^k - x^k) \\ &\quad + \frac{1}{2}(\hat{x}^k - x^k)^T\big(\nabla^2 f(\xi^k) - \nabla^2 f(x^k)\big)(\hat{x}^k - x^k) \end{aligned}$$

for some $\xi^k \in (x^k, \hat{x}^k)$. Hence, we get

$$\begin{aligned} &\psi(\hat{x}^k) - \psi(x^k) + \psi_k(x^k) - \psi_k(\hat{x}^k) \\ &= f(\hat{x}^k) - f(x^k) - \nabla f(x^k)^T(\hat{x}^k - x^k) - \frac{1}{2}(\hat{x}^k - x^k)^T H_k(\hat{x}^k - x^k) \\ &\le \frac{1}{2}\Big\|\nabla^2 f(\xi^k) - \nabla^2 f(x^k)\Big\| \cdot \|\hat{x}^k - x^k\|^2 + \frac{1}{2}\Big\|\nabla^2 f(x^k) - \nabla^2 f(x^*)\Big\| \cdot \|\hat{x}^k - x^k\|^2 \\ &\quad + \frac{1}{2}\Big\|\big(H_k - \nabla^2 f(x^*)\big)(\hat{x}^k - x^k)\Big\| \cdot \|\hat{x}^k - x^k\|. \end{aligned}$$

By the Dennis-Moré criterion, this is $o(\|\hat{x}^k - x^k\|^2)$ for $x^k \to x^*$. As before, it follows from the continuity of F and the results in Proposition 4.4 (a) and (b) that $\|\hat{x}^k - x^k\| \to 0$. Thus, using (13), we obtain

$$\begin{aligned}\psi(\hat{x}^k) - \psi(x^k) &= \big(\psi(\hat{x}^k) - \psi(x^k) + \psi_k(x^k) - \psi_k(\hat{x}^k)\big) + \psi_k(\hat{x}^k) - \psi_k(x^k)\\ &\le (\zeta - \sigma)\tilde{c}\|\hat{x}^k - x^k\|^2 + \zeta\Delta_k\\ &= (\zeta - \sigma)\tilde{c}\|\hat{x}^k - x^k\|^2 + \sigma\Delta_k + (\zeta - \sigma)\Delta_k\\ &\le (\zeta - \sigma)\tilde{c}\|\hat{x}^k - x^k\|^2 + \sigma\Delta_k - (\zeta - \sigma)\tilde{c}\|\hat{x}^k - x^k\|^2 = \sigma\Delta_k,\end{aligned}$$

for all sufficiently large k, where the last inequality follows from (19) (note that $\Delta_k = \Delta_{k,N}$ in the current situation). This proves that in this case the full step length is attained.

For the remaining part choose $\varepsilon > 0$ such that Proposition 4.4 holds for $x^k \in B_\varepsilon(x^*)$ and ∇f is Lipschitz continuous with constant $L > 0$ in $B_\varepsilon(x^*)$. Let k_0 be sufficiently large such that all iterates x^k for $k \ge k_0$ are in this neighbourhood. Note that

$$\begin{aligned}\|F(x^k)\| &= \|x^k - \operatorname{prox}_\varphi(x^k - \nabla f(x^k))\|\\ &= \|x^k - \operatorname{prox}_\varphi(x^k - \nabla f(x^k)) - x^* + \operatorname{prox}_\varphi(x^* - \nabla f(x^*))\|\\ &\le 2\|x^k - x^*\| + \|\nabla f(x^k) - \nabla f(x^*)\| \le (2 + L)\|x^k - x^*\|,\end{aligned}$$

where the inequality uses the nonexpansivity of the proximity operator, cf. [17, Lemma 2.4]. Using parts (a) and (b) yields $x^{k+1} = \hat{x}^k$. Thus, by Proposition 4.4 (a) and (c), we get

$$\begin{aligned}\|x^{k+1} - x^*\| = \|\hat{x}^k - x^*\| \;&\le\; \|\hat{x}^k - \hat{x}^k_{ex}\| + \|\hat{x}^k_{ex} - x^*\|\\ &\le \Big(C_1 + \frac{1}{\mu}C_2\Big)\eta_k\|F(x^k)\| + \frac{1}{\mu}\|\nabla f(x^k) - \nabla f(x^*) - \nabla^2 f(x^*)(x^* - x^k)\|\\ &\quad + \frac{1}{\mu}\Big\|\big(H_k - \nabla^2 f(x^*)\big)(\hat{x}^k - x^k)\Big\|.\end{aligned}$$

The twice continuous differentiability of f yields that the second term is $o(\|x^k - x^*\|)$. The Dennis-Moré condition implies that the third term is $o(\|x^k - x^*\|)$. Thus, the above yields part (c) for $\overline{\eta} = 1/((C_1 + \frac{1}{\mu}C_2)(L + 2))$. Finally, under the assumptions of part (d), also the first term is $o(\|x^k - x^*\|)$, which completes the proof. □

Note that one can also verify local quadratic convergence under slightly stronger assumption as in Theorem 4.5 (d), in particular, using a stronger version of the Dennis-Moré condition. The details are left to the reader.

5 Numerical results

In this section, we report some numerical results for solving problem (1) and show the competitiveness compared to several state-of-the-art methods. All numerical results have been obtained in MATLAB R2018b using a machine running Open SuSE Leap 15.1 with an Intel Core i5 processor 3.2 GHz and 16 GB RAM.

In the following, GPN denotes the globalized (inexact) proximal Newton method, whereas QGPN denotes a globalized (inexact) proximal quasi-Newton method, where the exact Hessian is replaced by a limited memory BFGS-update.

5.1 logistic regression with ℓ_1-Penalty

In this example, we consider the logistic regression problem

$$\min_{y,v} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 + \exp\left(-b_i(a_i^T y + v)\right)\right) + \lambda \|y\|_1, \tag{20}$$

where $a_i \in \mathbb{R}^n$ $(i = 1, \ldots, m)$ are given feature vectors and $b_i \in \{\pm 1\}$ the corresponding labels, $\lambda > 0$, $y \in \mathbb{R}^n, v \in \mathbb{R}$. Usually, we have $m \gg n$. Logistic regression is used to separate data by a hyperplane, see [25] for further information.

With $\phi : \mathbb{R} \to \mathbb{R}$, $\phi(u) := \log(1 + \exp(-u))$, $x := (y^T, v)^T$ and $A \in \mathbb{R}^{m \times (n+1)}$, where the i-th row of A is $(b_i a_i^T, b_i)$ for $i = 1, \ldots, m$, we can write (20) equivalently as

$$\min_x \psi(x) := \frac{1}{m} \sum_{i=1}^{m} \phi\big((Ax)_i\big) + \lambda \|x_{\{1,\ldots,n\}}\|_1. \tag{21}$$

The function ϕ is convex, but not strictly convex, and its derivative is globally Lipschitz continuous. Thus, this holds also for the smooth part of ψ.

5.1.1 Algorithmic details

Subproblem solvers The crucial part of the implementation is the solution of the subproblem (13). We use two methods for this aim, which are described below: The fast iterative shrinkage thresholding algorithm (FISTA) [6] and the globalized semismooth Newton method (SNF) [32].

FISTA by Beck and Teboulle [6] is an accelerated first order method for the solution of problems of type (1), where f is convex and has a Lipschitz continuous gradient. In every step a problem of type (6) is solved for $H_k = L_k I$, where L_k is an approximation to the Lipschitz constant of ∇f, which is updated by backtracking. After that, a step size is computed and the next iterate is a convex combination of the old iterate and the computed solution. For the approximation of the Lipschitz constant of f_k, we start with $L_0 := 1$ and use the increasing factor $\eta := 2$. The globalized proximal Newton-type method with this subproblem solver is denoted by GPN-F.

SNF by Milzarek and Ulbrich [32] is a semismooth Newton method with filter globalization. Since the subproblems in this example are convex, we use the convex variant of the method. The semismooth Newton method is essentially applied to the equation $F(x) = 0$ with $F(x)$ defined in (12). After computing a search direction, a filter decides if the update is applied or a proximal gradient step is performed. All

constants are chosen as in [32]. We denote the globalized proximal Newton method with SNF subproblem solver by GPN-S.

In both cases, the initial point for the subproblem solvers is the current iterate x^k.

Choice of parameters We use the parameters $p = 2.1$ and $\rho = 10^{-8}$ for the acceptance criterion (14). The line search is performed with $\beta = 0.1$ and $\sigma = 10^{-4}$. The constant c_k for the proximal gradient step is initialized with $c_0 = 1/6$, and in each step adapted to reach the Lipschitz constant of the gradient of f.

Variant with quasi-Newton-update Assuming that ψ is locally strongly convex in a neighbourhood of an accumulation point of a sequence generated by GPN, the sequence of matrices $\{H_k\}$ is generated using BFGS-updates and the subproblems in (13) are solved exactly, i.e. $\eta = 0$. Then, similar to [49] one can prove that the sequence $\{H_k\}$ satisfies the Dennis-Moré-condition.

Motivated by this idea, we implemented a variant of the algorithm, where the exact Hessian in the quadratic approximation (11) is replaced by a limited memory BFGS-update with a memory of 10. The implementation follows [14]. We skip the update, if $(s^k)^T y^k < 10^{-9}$ for $s^k = x^k - x^{k-1}$ and $y^k = \nabla f(x^k) - \nabla f(x^{k-1})$. Like before, we denote these methods by QGPN-F and QGPN-S, respectively.

5.1.2 State-of-the-art methods

We check the above described variants of GPN against each other, but also compare them with several state-of-the-art methods, which are listed below.

PG The proximal gradient method is described in Algorithm 2.2. It is a first order method to solve problem (1). We set $\beta = 0.1$, $\sigma = 10^{-4}$ and $H_k = c_k I$, where c_k is updated as before.

FISTA [6] The fast iterative shrinkage thresholding algorithm is an accelerated variant of the proximal gradient method. Details were already given in Sect. 5.1.1.

SpaRSA [45] SpaRSA (Sparse reconstruction by separable approximation) is another accelerated first order method to solve problem (1). The main difference to FISTA is the update of the factor c_k, which is done by a Barzilai-Borwein approach.

SNF [32] The semismooth Newton method with filter globalization is described in 5.1.1. Similar to the subproblem solver, we apply the convex version of the method.

5.1.3 Numerical comparison

We follow the example in [12] and generate test problems with $n = 10^4$ features and $m = 10^6$ training sets. Each feature vector a_i has approximately 10 nonzero entries, which are generated independently from a standard normal distribution. We choose $y^{\text{true}} \in \mathbb{R}^n$ with 100 nonzero entries and $v^{\text{true}} \in \mathbb{R}$, which are independently generated from standard normal distribution and define the labels as $b_i = \text{sign}\left(a_i^T y^{\text{true}} + v^{\text{true}} + v_i\right)$, where the v_i $(i = 1, \ldots, m)$ are also chosen independently from a normal distribution with variance 0.1. The regularization parameter λ is set to $0.1\lambda_{\max}$, where $\lambda_{\max}$ is the smallest value such that $y^* = 0$ is a solution of (20). The derivation of this value can be found in [25]. For all methods, we start with the initial value $x^0 = 0$.

Due to the differences of the methods, the standard termination criteria of them are not a suitable choice to compare the performance. Thus, we compute the approximate minimizer ψ^* of (20) using GPN-F with very high accuracy. We terminate each of the algorithms above when the value $\psi(x^k)$ in the current iterate x^k satisfies

$$\frac{\psi(x^k) - \psi^*}{|\psi^*|} \leq \texttt{tol} \tag{22}$$

for $\texttt{tol} = 10^{-6}$.

Termination of the subproblems We start with an investigation of the termination of the subproblems (13). As a consequence of Theorem 4.5, we can choose the sequence $\{\eta_k\}$ to be constant (const.). For our experiments, we computed an upper bound for $\overline{\eta}$ using the constants in the convergence theorem and set $\eta_k = 0.9\overline{\eta}$. A second possibility is to use a diminishing (dim.) sequence $\{\eta_k\}$. Here we investigated the sequence $\eta_k = 1/(k+1)$. Since the inexact termination criterion (13) is not practicable without significant additional computation costs, we also use a third variant: We minimize (11) using the standard termination criterion for the used solvers with a low maximal number of iterations, more precisely, 80 iterations for FISTA and 10 iterations for SNF, which resulted in the best performance. The tolerance is adapted in each step such that the subproblems are solved more exactly when the current iterate is near the solution.

The averaged results of 100 runs for the described variants of our method are listed in Table 1. It can be seen that for the variants with subproblem solver SNF, the computation costs using the diminishing or constant sequence $\{\eta_k\}$ are much higher than the costs using a maximum of 10 iterations, although, as expected, the number of total iterations is lower. Especially the number of evaluations of the proximity operator illustrates the difference in computation costs using the inexactness criterion in (13) and the approximation of the criterion by limiting the inner iterations. This is reasonable since there is one extra computation of the proximity operator in every inner iteration to check the inexactness condition. In contrast, the numbers of iterations are within the same range. For the variants using FISTA to solve the subproblems, we observe a similar behaviour, although it is less marked here.

To draw a conclusion from these observations, in the following we restrict the experiments and only investigate solving the subproblems with a maximum of 10 iterations (SNF) and 80 iterations (FISTA), which have the lowest computation costs. To accomplish comparability for these experiments, we look at the runtime of 100 test examples and document the results using the performance profiles introduced by Dolan and Moré [20]. The results are shown in Fig. 1, the averaged values for some counters are given in Table 2.

Comparison of GPN-variants We start with a comparison of the variants of the globalized proximal Newton-type methods, namely GPN-F, GPN-S, QGPN-F, and QGPN-S. At first, it can be observed that the iterations obtained by the inexact proximal Newton step are almost always accepted. We see that the semismooth Newton subproblem solver performs much better than the FISTA solver. One reason for this is that we can terminate the subproblem solvers in (Q)GPN-S after only 10 iterations to get reasonable results, whereas test runs show that (Q)

Table 1 Averaged values of 100 runs for the example in Sect. 5.1 with tolerance 10^{-6}

Method	Term.-crit.	Iter	Newton-iter	Sub-iter	Function eval	Proximity eval	Matrix-vector products
GPN-F	max.	11.7	11.7	994	12.7	1 016	3 923
	dim.	16.6	12.6	631	14.2	1 305	2 636
	const.	9.4	9.4	935	10.4	1 965	3 948
GPN-S	max.	16.9	16.9	33.4	118.0	50.8	296
	dim.	10.8	10.6	821	12.5	2 040	$3.21 \cdot 10^6$
	const.	8.9	7.5	849	10.5	2 081	$2.95 \cdot 10^6$
QGPN-F	max.	29.1	29.1	2 015	30.2	2 471	58.3
	dim.	27.5	27.5	1 522	28.6	3 369	55.1
	const.	28.8	28.8	2 778	30.0	6 234	57.8
QGPN-S	max.	21.6	21.6	36.2	22.7	58.9	43.3
	dim.	24.6	24.6	115	25.7	278.8	49.2
	const.	28.6	28.6	1684	29.8	4049	57.4

Abbreviations: term.-crit. (method to terminate the solver for subproblems), iter (total number of (outer) iterations), Newton-iter (number of Newton-iterations—only for GPN-variants and SNF), sub-iter (number of inner iterations), function eval (number of evaluations of the function f or its gradient), proximity eval (number of evaluations of the proximity operator), matrix-vector-products (number of evaluations of products $A \cdot x$ or $A^T \cdot x$)

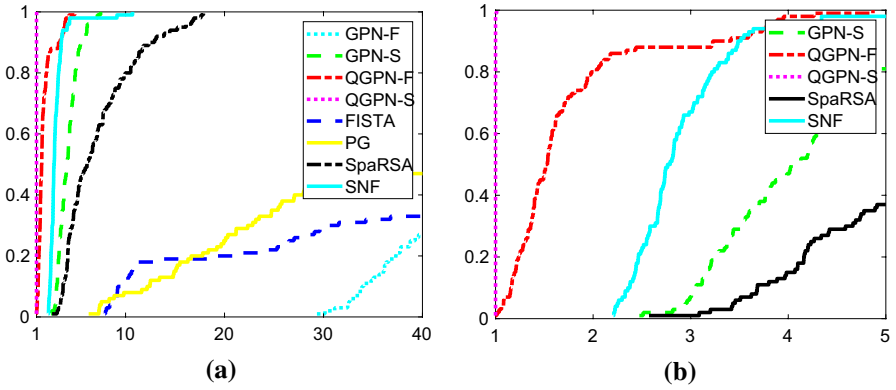


Fig. 1 Performance profiles showing the runtime for 100 random test examples as described in Sect. 5.1.3. Figure **a** shows a range from 1 to 40 times the best method, whereas Figure **b** is scaled from 1 to 5 times the best method

GPN-F performs best with a maximum of 80 iterations in each subproblem. Nevertheless, note that every iteration of SNF itself needs to solve a linear system by the CG method, but both, FISTA and SNF, need to evaluate the product $\nabla^2 f(x^k)z$ for some $z \in \mathbb{R}^n$ in every iteration, which is the most expensive part of the algorithm since it involves two multiplications with A or A^T.

Table 2 Averaged values of 100 runs for the example in Sect. 5.1 with tolerance 10^{-6}

Method	Iter	Newton-iter	Sub-iter	Function eval	Proximity eval	Matrix-vector products
GPN-F	11.7	11.7	994	12.7	1 016	3 923
GPN-S	16.9	16.9	33.4	18.0	50.8	296.0
QGPN-F	29.1	29.1	2 015	30.2	2 471	58.3
QGPN-S	21.6	21.6	36.2	22.7	58.9	43.3
FISTA	1 269	–	1 466	4 005	1 466	6 544
SpaRSA	133	–	221	223	222	446
PG	1 520	–	–	3 131	1 520	4 642
SNF	15.2	14.0	31.2	15.7	15.4	90.5

The columns have the same meaning as in Table 1

Furthermore, the performance of the variants with limited memory BFGS-update for the Hessian of the smooth part is significantly better than the use of the exact Hessian, although we need more outer and inner iterations to reach the termination accuracy. Again, this is due to the number of Hessian-vector-multiplications, which appear in QGPN only once in every iteration to compute the function value and the BFGS-update, whereas in GPN they are needed in every inner iteration.

Both arguments together verify why QGPN-S is the best variant tested, whereas the performance of GPN-F is not competitive.

We see in Table 2 that almost all solutions of the subproblems satisfy the descent condition (14) and, since the number of function evaluations is approximately the number of outer iterations, almost all search directions are applied with full step length. Thus, for this example, the globalization is not necessary in practice. Since problem (21) is globally strongly convex if A has full range, a slight adaption of our local convergence theory shows that one can prove convergence also without globalization. The details are left to the reader.

Comparison to other methods Since FISTA and the proximal gradient method are first order methods, it is not surprising that they need considerable more iterations to reach the termination tolerance. Thus, with the same arguments as above, they are not competitive due to the huge number of matrix-vector-products involving the matrices A or A^T, although they do not need to evaluate the Hessians. The third first order method, SpaRSA, is far better, because the number of iterations and therefore the number of matrix-vector-products is much smaller, but it is still not able to compete with the second order methods.

The semismooth Newton method with filter globalization is the only second order method we compare our method to. As before, we see a correlation between the runtime and the number of matrix-vector-products with one of the matrices A or A^T. As this number is higher than the one of QGPN, the runtime is still larger than the one of QGPN-S for most of the examples.

In contrast to our method, we did not implement SNF with a limited memory BFGS-update. The low number of matrix-vector-products given in Table 2 recommends that this would not yield a significantly better performance.

Comparing FISTA with GPN-F and QGPN-F, where FISTA is used to solve the subproblems, we see that GPN-F is not competitive for the mentioned reasons, whereas QGPN-F is far better than FISTA on its own. A similar observation is true for the comparison of SNF with GPN-S and QGPN-S, where the GPN method is still the slowest method but not significantly. Thus, the globalized proximal Newton-type method with limited memory BFGS-update for the Hessian accelerates the performance of the underlying subproblem solver.

5.2 Student's *t*-regression with ℓ_1-Penalty

In many applications of inverse problems, the aim is to find a sparse solution $x^* \in \mathbb{R}^n$ of the problem $Ax = b$ with $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. Often, b is not known exactly but only a perturbed vector $\hat{b}$. A widespread solution is to consider the penalized problem

$$\min_x \frac{1}{2}\|Ax - \hat{b}\|_2^2 + \lambda\|x\|_1$$

for some $\lambda > 0$. This works well if we have Gaussian errors in the entries of $\hat{b}$. Particularly, the influence of large errors is large. In problems, where the influence of large errors should be weighted less, but the influence of errors in a specific domain should be weighted more, it is reasonable to replace the quadratic loss by the student loss. We obtain the problem

$$\min_x \psi(x) := \sum_{i=1}^m \phi\big((Ax-b)_i\big) + \lambda\|x\|_1 = \sum_{i=1}^m \log\left(1 + \frac{(Ax-b)_i^2}{\nu}\right) + \lambda\|x\|_1, \tag{23}$$

with $\phi : \mathbb{R} \to \mathbb{R}, \phi(u) = \log\left(1 + \frac{u^2}{\nu}\right)$ for some $\nu > 0$. For more information on student's t-distribution, we refer to [1, 32] and references therein. It is easy to see that the derivative of ϕ is still Lipschitz continuous and ϕ is coercive, but not convex. Thus, many state-of-the-art methods are not applicable to this problem.

We expect a solution of (23) to solve the linear system $Ax = b$, at least approximately. Since ϕ is locally strongly convex in $B_{\sqrt{\nu}}(0)$, we expect that in a solution of (23) the local convergence theory is applicable.

5.2.1 Algorithmic details

Subproblem solvers As seen in Sect. 5.1, the SNF subproblem solver performed much better than the FISTA subproblem solver. Thus, we use again the semismooth Newton method with filter globalization [32] for the solution of the subproblems, apply at most 10 inner iterations per outer iteration and adapt the tolerance to get

more exact solutions, if the current iterate is close to the solution of the main problem. We denote this method by GPN.

Since the problem in this section is nonconvex, the subproblems might be not bounded from below. To circumvent this problem, we also implemented a variant with regularized Hessians. As the second derivative of ϕ is easy to compute and the Hessian of the objective function is of the form $A^T DA$ for some diagonal matrix $D \in \mathbb{R}^{m\times m}$, we replace all diagonal entries d_i of D by the maximum of d_i and a small positive constant. The subproblem solver remains unchanged and we denote this regularized method by GPN+.

Choice of parameters As above, we set $p = 2.1$, $\rho = 10^{-8}$, $\beta = 0.1$, and $\sigma = 10^{-4}$. In this case, we start with $c_0 = 100$ and again adapt c_k to approximate the Lipschitz constant of the gradient of the smooth part in (23).

Quasi-Newton-update In the second of the following test examples we use again a variant of the globalized proximal Newton method, where the Hessian of f is replaced by a limited memory BFGS-update with a memory of 10. We denote the method by QGPN. As before, we skip the update and use the previous approximation, if $(s^k)^T y^k < 10^{-9}$ for $s^k = x^k - x^{k-1}$ and $y^k = \nabla f(x^k) - \nabla f(x^{k-1})$. Since this problem is not convex, one could expect that skipping of updates happens occasionally. However, our experiments show that this happens in less than 10% and, if so, especially in the first iterations. Thus, the limited memory BFGS-updates are reasonably practicable.

5.2.2 State-of-the-art methods

Since problem (23) is nonconvex, most of the methods in Sect. 5.1 do not apply in this case. We therefore compare our algorithm to the following methods.

PG The proximal gradient method as described in Algorithm 2.2 has no convexity requirement. Again, we set $\beta = 0.1$, $\sigma = 10^{-4}$, and $H_k = c_k I$, where c_k is initialized with $c_0 = 100$ and adapted to reach a Lipschitz constant of ∇f.

SNF [32] The semismooth Newton method with filter globalization, as described in 5.1.1, has also a nonconvex variant with additional descent conditions, which are checked for the semismooth Newton update. We choose all constants as described in [32].

5.2.3 Numerical comparison

As mentioned above, we test two sets of examples. We start with the test setting described in [32]. Let $n = 512^2$ and $m = n/8 = 32768$. The matrix $A \in \mathbb{R}^{m\times n}$ takes m random cosine measurements, i.e. for a random subset $I \subset \{1, \dots, n\}$ with m elements, we set $Ax = (\mathrm{dct}(c))_I$, where dct is the discrete cosine transform.

We generate a true sparse vector $x^{\text{true}} \in \mathbb{R}^n$ with $k = \lfloor n/40 \rfloor = 6553$ nonzero entries, whose indices are chosen randomly. The nonzero components are computed via $x_i^{\text{true}} = \eta_1(i)10^{\eta_2(i)}$ with $\eta_1(i) \in \{\pm 1\}$ is a random sign and $\eta_2(i)$ is chosen independently from a uniform distribution in [0, 1]. The image $b \in \mathbb{R}^m$ is generated by adding Student's t-noise with degree of freedom 4 and rescaled by 0.1 to Ax^{true}. We set $\nu = 0.25$ and set $\lambda = 0.1\lambda_{\max}$, where $\lambda_{\max}$ is the critical value, for which the zero

vector is already a critical point of (23). Using Fermat's rule for the generalized Jacobian of (23), we obtain by a short calculation $\lambda_{\max} = 2\left\| \sum_{i=1}^{m} b_i/(\nu + b_i^2) \cdot a_i \right\|_\infty$, where a_i^T is the i-th row of A.

We start with the initial point $x^0 = A^T b$ and, again, terminate each of the algorithms above, when the value $\psi(x^k)$ in the current iterate x^k satisfies (22) for $\texttt{tol} = 10^{-6}$, where ψ^* is computed by GPN with a very high accuracy. It is important to mention that all stationary points of problem (23), if there is more than one, have the same function value. Thus, this termination criterion makes sense although the problem is nonconvex.

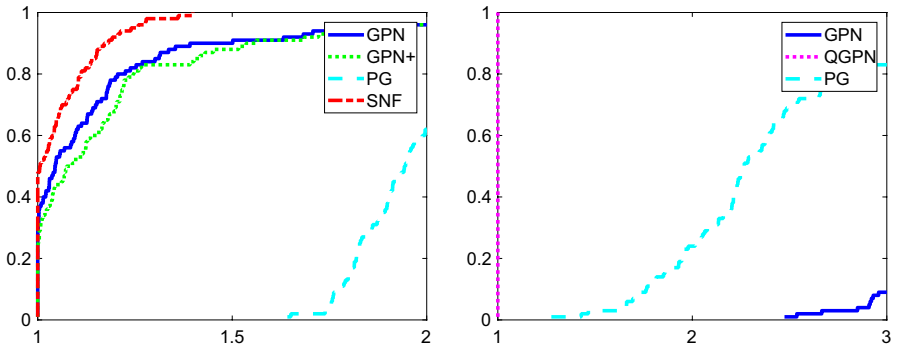
For this example, we do not use QGPN since test runs have shown that QGPN is significantly slower than GPN here. The reason is that, in contrast to the example in 5.1.3, the computation of matrix-vector-products involving the matrix A are cheaper than the product with the BFGS-matrix, as the discrete cosine transform is a predefined Matlab-function.

To accomplish comparability, we look at the runtime of 100 test examples and document the performance using the performance profiles introduced by Dolan and Moré [20]. The results are shown in Fig. 2a, the averaged values for some counters are given in Table 3.

The first observation is that there is no significant difference between the globalized proximal Newton method GPN and the regularized version GPN+. In both methods, almost all updates are performed by proximal Newton steps. Thus, in the following we refer only to GPN.

The proximal gradient method is in all examples significantly slower than the second order methods. As mentioned above, this is not due to the number of matrix-vector-products, which has the same magnitude as the one for GPN. In contrast, the numbers of function evaluations and evaluations of the proximity operator are much higher.

To demonstrate the performance of the limited memory BFGS proximal Newton-type method QGPN, we construct a second test example with higher computation costs for the matrix-vector-products with the matrices A or A^T. In the above test



(a) Example 1 **(b)** Example 2

Fig. 2 Performance profiles showing the runtime for 100 random test examples described in Sect. 5.2. Figures **a** and **b** correspond to Examples 1 and 2, respectively

Table 3 Averaged values of 100 runs for the first example in Sect. 5.2 with tolerance 10^{-6}

Method	Iter	Newton-iter	Sub-iter	Function eval	Proximity eval	Matrix-vector products
GPN	11.5	11.4	57.4	13.6	76.2	1 475
GPN+	11.6	11.5	58.0	13.7	77.2	1 530
PG	460	–	–	956	460	1 417
SNF	51.0	21.0	231	96.4	66.0	532

The columns have the same meaning as in Table 1

setting, we change n, m and use A as defined in Sect. 5.1, this is $n = 10^4$, $m = 10^6$, and $A \in \mathbb{R}^{m \times n}$ with approximately 10 nonzero entries in every row. Everything else remains unchanged.

As there was no significant difference in the performance of GPN and GPN+, we apply GPN, QGPN, SNF and the proximal gradient method PG to this setting. The results are shown in Fig. 2b and Table 4.

First, we observe that SNF did not converge at all within 1 000 iterations for this problem class. A look at the function value shows that it increases in every step. Since SNF is not a descent method regarding the function value and there is no result guaranteeing the convergence in the nonconvex case, this is not unreasonable.

Comparing the remaining methods, we find that the results confirm the observations of the example in Sect. 5.1. The performance of QGPN is far the best, whereas GPN is not competitive, though it is not as bad as for the ℓ_1-regularized logistic regression.

5.3 Logistic regression with overlapping group penalty

The main advantage of the globalized proximal Newton method over semismooth Newton methods is that it is also able to solve problems of type (1), where the nonsmooth function φ is not the ℓ_1-norm and there is no known formula to

Table 4 Averaged values of 100 runs for the second example in Sect. 5.2

Method	iter	Newton-iter	Sub-iter	Function eval	Proximity eval	Matrix-vector products
GPN	49.2	49.2	246	83.3	330	2 169
GPN+	29.6	29.6	148	68.1	184	3 547
QGPN	125	125	837	211	994	336
PG	156	–	–	572	156	728
SNF	DNC	DNC	DNC	DNC	DNC	DNC

The columns have the same meaning as in Table 1. The abbreviation DNC stands for: did not converge within 1 000 iterations

compute the proximity operator to this function. An example is the group penalty function

$$\varphi(x) = \lambda \sum_{j=1}^{s} \mu_j \|x_{G_j}\|_2,$$

where $\mu_j > 0$ are positive weights, $\lambda > 0$ and $G_j \subset \{1, \dots, n\}$ are nonempty sets. When the sets G_j $(j = 1, \dots, s)$ form a partition of $\{1, \dots, n\}$ or are at least pairwise disjoint, the proximity operator can be computed explicitly. Here we are interested in the case of overlapping groups, i.e. the sets G_j are not pairwise disjoint. In this case, no explicit formula for the proximity operator is known.

Like in Sect. 5.1 we consider a logistic regression problem

$$\min_x \frac{1}{m} \sum_{i=1}^{m} \phi\big((Ax)_i\big) + \lambda \sum_{j=1}^{s} \mu_j \|x_{G_j}\|_2, \tag{24}$$

where $A \in \mathbb{R}^{m \times n}$ contains the information on feature vectors and corresponding labels and $\phi : \mathbb{R} \to \mathbb{R}$ is defined by $\phi(u) := \log(1 + \exp(-u))$. A group penalty makes sense in many applications here, since some features are related to others. For more information on logistic regression with group penalty, we refer to [30].

5.3.1 Algorithmic details

Subproblem solver As there is no formula to compute the proximity operator of φ, the subproblem solvers of the previous sections are not directly applicable. We can write φ as $\tilde{\varphi} \circ B$, where B is a linear mapping and $\tilde{\varphi}$ is a group penalty without overlapping. Thus, we can compute the proximity operator of $\tilde{\varphi}$. Both, the proximal Newton subproblem as well as the proximity operator, can be written as

$$\min_x \frac{1}{2} x^T Q x + c^T x + \tilde{\varphi}(Bx)$$

with a positive definite matrix $Q \in \mathbb{R}^{n \times n}$ and $c \in \mathbb{R}^n$. We solve both problems with fixed point methods described by Chen et al. in [16]. For the computation of the proximity operator, we use the fixed point algorithm based on the proximity operator (FP^{2}O) and for solving the proximal Newton subproblem the primal-dual fixed point algorithm based on the proximity operator (PDFP^{2}O).

For both methods, we use a stopping tolerance of 10^{-9} and apply at most 10 iterations for each problem. For the method we also need the largest eigenvalue of BB^T, which can be shown to be equal to the largest integer k such that there exists an index $i \in \{1, \dots, n\}$ that is contained in k groups G_j.

Choice of parameters As before, we set the parameters to $p = 2.1$, $\rho = 10^{-8}$, $\beta = 0.1$, and $\sigma = 10^{-4}$. Here, we start with $c_0 = 1$ and again adapt c_k to approximate the Lipschitz constant of the gradient of the smooth part in (24).

Other methods We make a comparison between our method with the above mentioned subproblem-solvers, FISTA [6] with the parameters as in 5.1.1. For the

computation of the proximity operators, we also use FP^{2}O. Furthermore, we apply PDFP^{2}O directly to problem (24).

5.3.2 Numerical comparison

We follow an example in [2] and generate $A \in \mathbb{R}^{n\times m}$ with $n = 1000$, $m = 700$ from a uniform distribution and normalize the columns of A. The groups G_j are

$$\{1,\dots,5\},\ \{5,\dots,9\},\ \{9\dots,13\},\ \{13,\dots,17\},\ \{17,\dots,21\},$$
$$\{4,22,\dots,30\},\ \{8,31,\dots,40\},\ \{12,41,\dots,50\},\ \{16,51,\dots,60\},\ \{20,61,\dots,70\},$$
$$\{71,\dots,80\},\ \{81,\dots,90\},\ \dots,\ \{991,\dots 1000\}.$$

The first five groups contain five consecutive numbers and the last element of one group is, at the same time, the first element of the next group. Each of the next five groups contain one element of one of the first groups. The remaining groups have no overlap and contain always 10 elements. The coefficients μ_j are chosen to be $1/\sqrt{|G_j|}$, where $|G_j|$ is the number of indices in that group.

The parameter λ is again chosen as $0.1\lambda_{\max}$, where $\lambda_{\max}$ is the critical value such that 0 is a solution of (24) for all $\lambda \geq \lambda_{\max}$. Let a_i^T be the rows of A. Then a short computation shows $\lambda_{\max} = \sqrt{5}/(2m)\left\|\sum_{i=1}^m a_i\right\|_2$. As before, we start with the initial value $x^0 = 0$.

We terminate each of the algorithm as soon as the current iterate satisfies (22) for $\texttt{tol} = 10^{-6}$, where ψ^* is the function value computed by GPN using a very high accuracy. Again, we document the results using the performance profiles on the runtime of 100 test examples. The results are shown in Fig. 3, the averaged values for some counters are given in Table 5.

We see that there are about 15% of the examples, where FISTA performs better than GPN, but in most examples GPN shows by far the best performance. This can be seen by looking at the number of inner iterations of both methods. In this

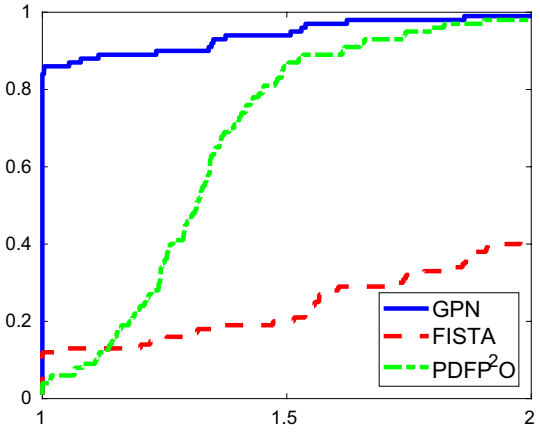


Fig. 3 Performance profile showing the runtime for 100 random test examples from Sect. 5.3 with tolerance 10^{-6}

Table 5 Averaged values of 100 runs for the example in Sect. 5.3 using the tolerance 10^{-6} and three different methods

Method	Iter	Newton-iter	Sub-iter	Matrix-vector products
GPN	9.5	9.5	95.1	221
PDFP^{2}O	76.9	–	–	156
FISTA	23.4	–	234	119

case, the costs of inner iterations is almost equal for both methods. Since the average number of inner iterations in FISTA is more then twice as big as the one of GPN, this illustrates the difference in performance.

5.4 Nonconvex image restoration

We demonstrate the performance of our method for nonconvex image restoration. Given a noisy blurred image $b \in \mathbb{R}^n$ and a blur operator $A \in \mathbb{R}^{n\times n}$, the aim is to find an approximation x to the original image satisfying $Ax = b$. Note that, for simplicity, we assume that the images x, b are vectors in $\mathbb{R}^n$. For this purpose, we use again the student loss from Sect. 5.2 and get the problem

$$\min_x \psi(x) := \sum_{i=1}^{n} \phi\big((Ax-b)_i\big) + \lambda\|Bx\|_1,$$

where $\phi : \mathbb{R} \to \mathbb{R}, \phi(u) = \log\left(1 + \frac{u^2}{\nu}\right)$ for some $\nu > 0$, and $B : \mathbb{R}^n \to \mathbb{R}^n$ is a two-dimensional discrete Haar wavelet transform, which guarantees antialiasing.

Since B is orthogonal, we get

$$\operatorname{prox}^{\tau}_{\lambda\|B\cdot\|_1}(u) = B^T \operatorname{prox}^{\tau}_{\lambda\|\cdot\|_1}(Bu).$$

Thus, the proximity operator can be computed exactly.

Similar to Sect. 5.2, we expect that ψ is strongly convex in a neighbourhood of a solution such that our local convergence theory applies here.

5.4.1 Algorithmic details

We solve the subproblems using FISTA with a maximum of 50 iterations and a tolerance of 10^{-6}. We do not use the SNF-solver here since the occurring linear systems of equations are not separable and we would need to solve a full dimensional system of equations several times, see below for details. The parameters are chosen as in Sect. 5.3.1.

We compare our methods GPN and the limited memory BFGS variant QGPN, where the updating of the BFGS-matrix follows the description in Sect. 5.2.1, to the proximal gradient method PG and the semismooth Newton method with filter globalization SNF [32] with the parameters mentioned in that paper. In this case, the matrix $M(x^k)$ occurring in the linear systems $M(x^k) = -F(x^k)$ has the form

$$M(x^k) = (B^T D_k B - I)H_k - B^T D_k B,$$

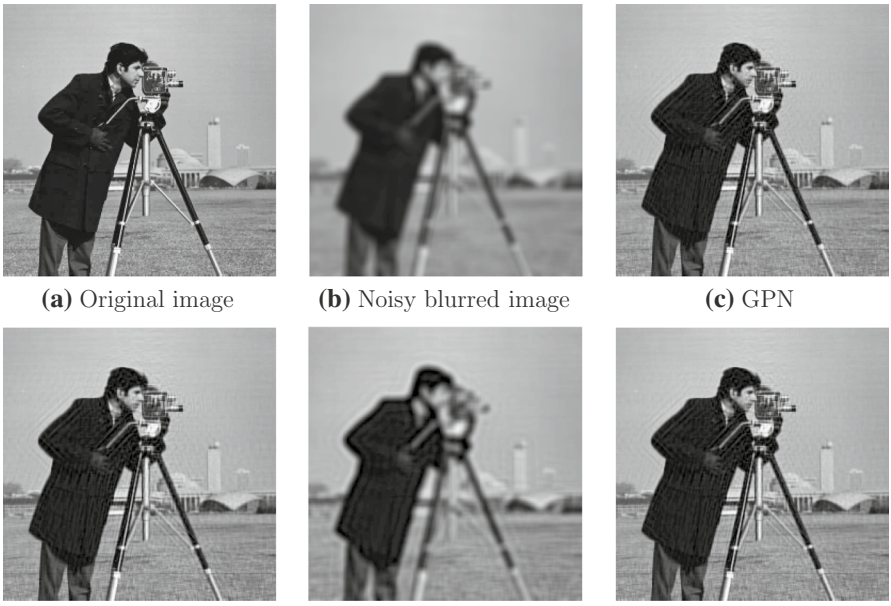
where H_k is an approximation to the Hessian of the smooth part of ψ and D_k is a diagonal matrix depending on the iterate x^k. This matrix does not have a block structure or is separable, so this is a full dimensional linear system of equations, which impairs the performance of this method. We solve each of these systems using GMRES(m) with 100 iterations and restart every $m = 10$ iterations.

5.4.2 Numerical comparison

We follow the example in [41], see also [11]. In detail, A is a Gaussian blur operator with standard deviation 4 and a filter size of 9, $\nu = 1$ and B is the discrete Haar wavelet transform of level four. Furthermore, we choose $\lambda = 10^{-4}$. The blurred noisy image b is created by applying A to the test image *cameraman* of size 256×256 and adding Student's t-noise with degree of freedom 1 and rescaled by 10^{-3}. For all methods, the initial point is $x^0 = b$.

Since the most expensive computations are the applications of A, B and their transposes, we stop each of the algorithms if, after an outer iteration, the sum of these applications exceeds $2 \cdot 10^4$. The results are shown in Fig. 4 and Table 6.

The reason why the restored images are minimal lighter than the original is that we used the Hair wavelet transform with four levels and not the maximal possible level $\log_2(256) = 8$. Furthermore, we mention that for GPN and QGPN almost all



(a) Original image **(b)** Noisy blurred image **(c)** GPN

(d) QGPN **(e)** SNF **(f)** PG

Fig. 4 Nonconvex image restoration: Original and blurred image and recovered images using the stated algorithms and terminating after $2 \cdot 10^4$ calls of A and B

Table 6 Values of the example in Sect. 5.3 for the four tested algorithms

Method	Time	Iter	Optim	Subiter	A-calls	B-calls
GPN	25.7	99	0.27	4 950	10 149	10 049
QGPN	62.2	194	0.99	9 650	413	19 655
PG	98.7	4 839	0.58	–	10 324	9 679
SNF	23.6	52	3.17	5 200	9 868	10 246

Abbreviations: time (CPU-time in seconds), iter (total number of (outer) iterations), optim (optimality criterion $\frac{\psi(x)-\psi^*}{\psi^*}$), subiter (number of inner iterations), A-calls, B-calls (applications of the mapping A and B and transposed mapping, resp.)

iterations are Newton steps, whereas for SNF only half of the iterations are Newton steps. As expected, the performance of the semismooth Newton method with filter globalization is not satisfying here, since the solution of the linear systems is expensive. In contrast, the proximal methods show good restorations. The difference in the corresponding images in Fig. 4 are hard to see, so we study the values in Table 6.

The relative error $(\psi(x) - \psi^*)/\psi^*$, where x is the image provided by the algorithm and ψ^* is the value of ψ in the original image, is best for GPN, so the corresponding image best approximates the original one. Comparing the inner iterations of GPN and QGPN with the iterations of the proximal gradient method, the ones of GPN and PG are within the same range, whereas the ones of QGPN have almost the double value. This and the number of calls of B and B^T explain why the CPU-time used by QGPN is approximately twice as much as the one of GPN. In this case, the avoidence of calls of A and A^T does not yield a better performance, since the price to pay is the higher number of calls of the Haar transform.

Comparing GPN and PG, the numbers of (inner) iterations and applications of A, B and their transposes are almost the same, but the superiority of the second order method GPN over PG can be seen in the values of the CPU-time and the relative error of the function value.

6 Conclusion

We introduced a globalization of the proximal Newton-type method to solve structured optimization problems consisting of a smooth and a convex function. For this purpose the proximal Newton-type method was combined with a proximal gradient method using a novel descent criterion. We also gave an inexactness approach and the possibility to replace the Hessian of the smooth part by quasi-Newton matrices. We proved global convergence in the convex and nonconvex case and, under suitable conditions, local superlinear convergence.

The numerical part shows that the proposed method is competitive for convex and nonconvex problems, especially when the computation of the Hessian is expensive and we can use limited memory quasi-Newton updates. Furthermore, when there is

no efficient way to compute the proximity operator for the nonsmooth function, the globalized proximal Newton-type method outperforms the methods compared to.

Acknowledgements The authors would like to thank the two referees for the very detailed comments which helped a lot to improve the paper significantly.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

1. Aravkin, A., Friedlander, M.P., Herrmann, F.J., Van Leeuwen, T.: Robust inversion, dimensionality reduction, and randomized sampling. Math. Program **134**, 101–125 (2012)
2. Argyriou, A., Micchelli, C.A., Pontil, M., Shen, L., Xu, Y.: Efficient first order methods for linear composite regularizers, arXiv preprint arXiv:1104.1436, (2011)
3. Banerjee, O., Ghaoui, L.E., d'Aspremont, A., Natsoulis, G.: Convex optimization techniques for fitting sparse gaussian graphical models. In: Proceedings of the 23rd international conference on Machine learning, pp. 89–96 (2006)
4. Bauschke, H., Combettes, P.: Convex Analysis and Monotone Operator Theory in Hilbert Spaces. CMS Books in Mathematics, 2nd edn. Springer, Berlin (2017)
5. Beck, A.: First-Order Methods in Optimization. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Philadelphia (2017)
6. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imag. Sci. **2**, 183–202 (2009)
7. Becker, S., Fadili, J., Ochs, P.: On quasi-newton forward-backward splitting: Proximal calculus and convergence. SIAM J. Optim. **29**, 2445–2481 (2019)
8. Bonettini, S., Loris, I., Porta, F., Prato, M.: Variable metric inexact line-search-based methods for nonsmooth optimization. SIAM J. Optim. **26**, 891–921 (2016)
9. Bonettini, S., Loris, I., Porta, F., Prato, M., Rebegoldi, S.: On the convergence of a linesearch based proximal-gradient method for nonconvex optimization. Inv. Prob. **33**, 055005 (2017)
10. Bonettini, S., Prato, M.: New convergence results for the scaled gradient projection method. Inv. Prob. **31**, 095008 (2015)
11. Boţ, R.I., Csetnek, E.R., László, S.C.: An inertial forward-backward algorithm for the minimization of the sum of two nonconvex functions. EURO J. Comput. Optim. **4**, 3–25 (2016)
12. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. **3**, 1–122 (2011)
13. Byrd, R.H., Nocedal, J., Oztoprak, F.: An inexact successive quadratic approximation method for l-1 regularized optimization. Math. Program. **157**, 375–396 (2016)
14. Byrd, R.H., Nocedal, J., Schnabel, R.B.: Representations of quasi-Newton matrices and their use in limited memory methods. Math. Program. **63**, 129–156 (1994)
15. Chen, D.-Q., Zhou, Y., Song, L.-J.: Fixed point algorithm based on adapted metric method for convex minimization problem with application to image deblurring. Adv. Comput. Math. **42**, 1287–1310 (2016)
16. Chen, P., Huang, J., Zhang, X.: A primal-dual fixed point algorithm for convex separable minimization with applications to image restoration. Inv. Prob. **29**, 025011 (2013)

17. Combettes, P.L., Wajs, V.R.: Signal recovery by proximal forward-backward splitting. Multiscale Model. Simul. **4**, 1168–1200 (2005)
18. De Luca, T., Facchinei, F., Kanzow, C.: A semismooth equation approach to the solution of nonlinear complementarity problems. Math. Program. **75**, 407–439 (1996)
19. Dennis, J.E., Moré, J.J.: A characterization of superlinear convergence and its application to quasi-Newton methods. Math. Comput. **28**, 549–560 (1974)
20. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. program. **91**, 201–213 (2002)
21. Fountoulakis, K., Tappenden, R.: A flexible coordinate descent method. Comput. Optim. Appl. **70**, 351–394 (2018)
22. Fukushima, M., Mine, H.: A generalized proximal point algorithm for certain non-convex minimization problems. Int. J. Syst. Sci. **12**, 989–1000 (1981)
23. Ghanbari, H., Scheinberg, K.: Proximal quasi-Newton methods for regularized convex optimization with linear and accelerated sublinear convergence rates. Comput. Optim. Appl. **69**, 597–627 (2018)
24. Gu, B., Huo, Z., Huang, H.: Inexact proximal gradient methods for non-convex and non-smooth optimization, arXiv preprint arXiv:1612.06003, (2016)
25. Koh, K., Kim, S.-J., Boyd, S.: An interior-point method for large-scale l1-regularized logistic regression. J. Mach. Learn. Res. **8**, 1519–1555 (2007)
26. Lee, C.-P., Wright, S.J.: Inexact successive quadratic approximation for regularized optimization. Comput. Optim. Appl. **72**, 641–674 (2019)
27. Lee, J.D., Sun, Y., Saunders, M.A.: Proximal Newton-type methods for minimizing composite functions. SIAM J. Optim. **24**, 1420–1443 (2014)
28. Li, J., Andersen, M.S., Vandenberghe, L.: Inexact proximal Newton methods for self-concordant functions. Math. Methods Oper. Res. **85**, 1–23 (2016)
29. Li, Q., Shen, L., Xu, Y., Zhang, N.: Multi-step fixed-point proximity algorithms for solving a class of optimization problems arising from image processing. Adv. Comput. Math. **41**, 387–422 (2015)
30. Meier, L., Van De Geer, S., Bühlmann, P.: The group lasso for logistic regression. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **70**, 53–71 (2008)
31. Milzarek, A.: Numerical methods and second order theory for nonsmooth problems. PhD thesis, Technische Universität München (2016)
32. Milzarek, A., Ulbrich, M.: A semismooth Newton method with multidimensional filter globalization for l_1-optimization. SIAM J. Optim. **24**, 298–333 (2014)
33. Moré, J.J., Sorensen, D.C.: Computing a trust region step. SIAM J. Sci. Stat. Comput. **4**, 553–572 (1983)
34. Moreau, J.-J.: Proximité et dualité dans un espace hilbertien. Bulletin de la Société mathématique de France **93**, 273–299 (1965)
35. Nesterov, Y.: Gradient methods for minimizing composite functions. Math. Program. **140**, 125–161 (2013)
36. Patrinos, P., Bemporad, A.: Proximal Newton methods for convex composite optimization. In: 52nd IEEE Conference on Decision and Control, IEEE, pp. 2358–2363 (2013)
37. Patrinos, P., Stella, L., Bemporad, A.: Forward-backward truncated Newton methods for convex composite optimization, arXiv preprint arXiv:1402.6655, (2014)
38. Rockafellar, R.T.: Monotone operators and the proximal point algorithm. SIAM J. Control Optim. **14**, 877–898 (1976)
39. Scheinberg, K., Goldfarb, D., Bai, X.: Fast first-order methods for composite convex optimization with backtracking. Found. Comput. Math. **14**, 389–417 (2014)
40. Scheinberg, K., Tang, X.: Practical inexact proximal quasi-Newton method with global complexity analysis. Math. Program. **160**, 495–529 (2016)
41. Stella, L., Themelis, A., Patrinos, P.: Forward-backward quasi-Newton methods for nonsmooth optimization problems. Comput. Optim. Appl. **67**, 443–487 (2017)
42. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. R. Stat. Soc. Ser. B (Methodol.) **58**, 267–288 (1996)
43. Tran-Dinh, Q., Kyrillidis, A., Cevher, V.: A proximal Newton framework for composite minimization: Graph learning without Cholesky decompositions and matrix inversions. In: International Conference on Machine Learning, pp. 271–279 (2013)
44. Tseng, P., Yun, S.: A coordinate gradient descent method for nonsmooth separable minimization. Math. Program. **117**, 387–423 (2009)

45. Wright, S.J., Nowak, R.D., Figueiredo, M.A.: Sparse reconstruction by separable approximation. IEEE Trans. Sig. Process. **57**, 2479–2493 (2009)
46. Yuan, M., Lin, Y.: Model selection and estimation in regression with grouped variables. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **68**, 49–67 (2006)
47. Yue, M.-C., Zhou, Z., So, A.M.-C.: A family of inexact SQA methods for non-smooth convex minimization with provable convergence guarantees based on the Luo-Tseng error bound property. Math. Program. **174**, 327–358 (2019)
48. Zhang, S., Qian, H., Gong, X.: An alternating proximal splitting method with global convergence for nonconvex structured sparsity optimization. In: 30. AAAI Conference on Artificial Intelligence, pp. 2330–2336 (2016)
49. Zhong, K., Yen, I.E.-H., Dhillon, I.S., Ravikumar, P.K.: Proximal quasi-Newton for computationally intensive l1-regularized M-estimators. In: Advances in Neural Information Processing Systems 27, pp. 2375–2383 (2014)