Projected-Search Methods for Bound-Constrained Optimization

Michael W. Ferry* Philip E. Gill† Elizabeth Wong† Minxin Zhang†

UCSD Center for Computational Mathematics
Technical Report CCoM-1-20
March 25, 2020

Abstract

Projected-search methods for bound-constrained minimization are based on performing a line search along a continuous piecewise-linear path obtained by projecting a search direction onto the feasible region. A potential benefit of a projected-search method is that many changes to the active set can be made at the cost of computing a single search direction.

As the objective function is not differentiable along the piecewise-linear path, it is not possible to use a line-search based on satisfying the Wolfe conditions, which involve the derivatives at two points on the search path. This means that the step must be computed using a simpler backtracking method. Methods for conventional unconstrained minimization that use the Wolfe conditions are more reliable and efficient than methods based on simple backtracking. For example, if the search direction is generated using a quasi-Newton method, the Wolfe conditions impose a restriction on the directional derivative that guarantees the satisfaction of a necessary condition for the quasi-Newton update to give a positive-definite approximate Hessian. This paper concerns projected-search methods based on a new *quasi-Wolfe* line search that is appropriate for piecewise differentiable functions. The behavior of the line search is similar to that of a conventional Wolfe line search, except that a step is accepted under a wider range of conditions. These conditions take into consideration steps at which the restriction of the objective function on the search path is not differentiable. Standard existence and convergence results associated with a conventional Wolfe line search are extended to the quasi-Wolfe case.

The quasi-Wolfe line search is considered in conjunction with two projected search methods: a limited-memory quasi-Newton method and a new method that combines a primal-dual interior method with a projected search. Computational results show that in these contexts, a quasi-Wolfe line search is substantially more efficient and reliable than an Armijo line search.

Key words. Bound constrained optimization, projected-search methods, line search methods, quasi-Newton methods, primal-dual interior methods.

AMS subject classifications. 49M37, 65K05, 90C30, 90C53

*NVIDIA Corporation, Hillsboro, Oregon (michael@mwferry.com).

†Department of Mathematics, University of California, San Diego, La Jolla, CA 92093-0112 (pgill@ucsd.edu, elwong@ucsd.edu, miz151@ucsd.edu) Research supported in part by National Science Foundation grants DMS-0915220 and DMS-1318480. The content is solely the responsibility of the authors and does not necessarily represent the official views of the funding agencies.

1. Introduction

This paper concerns the formulation of line-search methods for the bound-constrained problem

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\;\; f(x) \quad \text{subject to} \quad x \in \Omega, \tag{BC}$$

where f is a twice-continuously differentiable function with gradient ∇f, and $\Omega = \{\, x : \ell \le x \le u \,\}$ for vectors of lower and upper bounds such that $\ell \le u$ (with all inequalities defined componentwise).

Projected-search line-search methods for problem (BC) generate a sequence of feasible iterates $\{\, x_k \,\}_{k=0}^{\infty}$ such that $x_{k+1} = \mathbf{proj}_{\Omega}(x_k + \alpha p_k)$, where p_k is a descent direction and $\mathbf{proj}_{\Omega}(x)$ is the projection of x onto the feasible region, i.e.,

$$[\,\mathbf{proj}_{\Omega}(x)\,]_i = \begin{cases} \ell_i & \text{if } \; x_i < \ell_i, \\ u_i & \text{if } \; x_i > u_i, \\ x_i & \text{otherwise.} \end{cases}$$

A potential benefit of a projected-search method is that many changes to the active set can be made at the cost of computing a single search direction. The projected-search methods of Goldstein [21], Levitin and Polyak [27], and Bertsekas [2] are based on using the steepest-descent direction $p_k = -\nabla f(x_k)$. Bertsekas [4] and Calamai and Moré [8] propose methods that identify the optimal active set using a projected-search method and then switch to a Newton method. Projected-search methods based computing p_k using a quasi-Newton method are proposed by Ni and Yuan [31], Kim, Sra and Dhillon [25], Ferry [11], and Ferry et al. [12].

Many methods for minimizing f with no constraints generate a sequence of iterates $\{\, x_k \,\}_{k=0}^{\infty}$ such that x_{k+1} is chosen to give a decrease in f that is at least as good as a fixed fraction η_A $(0 < \eta_A < 1)$ of the decrease in the local affine model $f(x_k) + \nabla f(x_k)^T(x - x_k)$. If x_{k+1} is computed as $x_{k+1} = x_k + \alpha_k p_k$, where p_k is a vector and α_k is a scalar step, then the sufficient-decrease condition may be written as

$$f(x_k + \alpha_k p_k) \le f(x_k) + \alpha_k \eta_A \nabla f(x_k)^T p_k, \tag{1.1}$$

(see, e.g., Armijo [1], Ortega and Rheinboldt [33]). Many practical methods satisfy the Armijo condition in conjunction with a condition on the directional derivative $\nabla f(x_k + \alpha_k p_k)^T p_k$. In particular, the strong Wolfe conditions require that α_k satisfy both (1.1) and

$$|\nabla f(x_k + \alpha_k p_k)^T p_k| \le \eta_W |\nabla f(x_k)^T p_k|, \tag{1.2}$$

where η_W is a preassigned scalar such that $\eta_W \in (\eta_A, 1)$. (See, e.g., Wolfe [36], Moré and Thuente [30], and Gill et al. [20]). Alternatively, the weak Wolfe conditions involve the Armijo condition (1.1) in conjunction with the one-sided condition

$$\nabla f(x_k + \alpha_k p_k)^T p_k \ge \eta_W \nabla f(x_k)^T p_k \tag{1.3}$$

instead of (1.2). The strong Wolfe conditions allow η_W to be chosen to vary the accuracy of the step. If η_A is fixed at a value close to zero (e.g., 10^{-4}), then a value of η_W close to η_A gives a "tighter" or more accurate step with respect to closeness to a critical point of $\nabla f(x_k + \alpha p_k)^T p_k$. A value of η_W close to one results in a "looser" or more approximate step. If the Wolfe conditions are used in conjunction with a quasi-Newton method, there is the additional benefit that the conditions (1.2) and (1.3) guarantee a positive-definite approximate Hessian H_k can be updated to give a positive definite H_{k+1} for the next iterate. If α_k satisfies either the strong Wolfe condition (1.2) or the weak Wolfe condition (1.3), then $w_k^T d_k > 0$, where $w_k = \nabla f(x_{k+1}) - \nabla f(x_k)$, and $d_k = x_{k+1} - x_k$. The inequality $w_k^T d_k > 0$ is a necessary condition for the updated quasi-Newton Hessian to be positive definite.

In a projected-search method the new iterate may be written as $x_{k+1} = x_k(\alpha_k)$, where $x_k(\alpha)$ denotes the vector $x_k(\alpha) = \mathbf{proj}_{\Omega}(x_k + \alpha p_k)$. The function $x_k(\alpha)$ defines a piecewise linear continuous path, and the line-search function $f\big(x_k(\alpha)\big)$ is not necessarily differentiable along $x_k(\alpha)$. In particular, $f\big(x_k(\alpha)\big)$ has "kinks" where $[\,x_k + \alpha p_k\,]_i = \ell_i$ or $[\,x_k + \alpha p_k\,]_i = u_i$. This implies that it is not possible to use a line search based on the conventional Wolfe conditions, and the benefit of guaranteeing a positive-definite quasi-Newton update is lost. Thus, existing projected-search methods are restricted to using a *quasi-Armijo* line search based on satisfying the Armijo condition along the path $x_k(\alpha)$. Let γ, σ, and η_A denote fixed parameters such that $\gamma > 0$, $\sigma \in (0,1)$, and $\eta_A \in (0,1)$. A quasi-Armijo step has the form $\alpha_k = \gamma\sigma^{t_k}$, where t_k is the smallest nonnegative integer such that

$$f\big(x_k(\alpha_k)\big) \le f(x_k) + \alpha_k \eta_A \nabla f(x_k)^T p_k \tag{1.4}$$

(see Bertsekas [2, 3]).

In this paper we propose methods based on a new *quasi-Wolfe* line search that is specifically designed for use with a projected search. The behavior of the line search is similar to that of a conventional Wolfe line search, except that a step is accepted under a wider range of conditions that take into account steps at which f is not differentiable. The paper is organized in seven sections. The standard results associated with a conventional Wolfe line search are reviewed in Section 2. Analogous results are established for the quasi-Wolfe line search in Section 3. Section 4 describes the use of the quasi-Wolfe line search in conjunction with two projected search methods: a limited-memory quasi-Newton method and a new method that combines a primal-dual interior method with a projected search. The convergence properties of a projected-search method with a quasi-Wolfe line search are established in Section 5. Finally, the numerical performance of the quasi-Wolfe line search in conjunction with a limited-memory quasi-Newton projected-search method are considered in Section 6.

Notation. Given vectors x and y, the vector consisting of x augmented by y is denoted by (x, y). The subscript i is appended to vectors to denote the ith component of that vector, whereas the subscript k is appended to a vector to denote its value during the kth iteration of an algorithm, e.g., x_k represents the value for x during the kth iteration, whereas $[\,x_k\,]_i$ denotes the ith component of the vector

x_k. The ith component of the gradient of the scalar-valued function f is denoted by $\nabla_i f(x)$. Given vectors a and b with the same dimension, vectors with ith component $a_i b_i$ and a_i/b_i are denoted by $a \cdot b$ and $a \cdot\!/\, b$ respectively. Similarly, $\min(a,b)$ is a vector with components $\min(a_i, b_i)$. The vector e denotes the column vector of ones, and I denotes the identity matrix. The dimensions of e and I are defined by the context. The vector two-norm or its induced matrix norm are denoted by $\|\cdot\|$.

2. The Wolfe line search

A typical Wolfe line search may be viewed as a two-stage process. The first stage involves the determination of an interval containing a Wolfe step, if one exists. The second stage is to locate a Wolfe step in this interval using safeguarded polynomial interpolation. If the first stage fails, then the objective function is necessarily unbounded below. The key principle that drives the first stage is that certain conditions may be formulated that determine if an interval contains a Wolfe step. Much of the discussion in this section is based on the work of Morè and Thuente [30]. More information may be found in Wolfe [37] and Nocedal and Wright [32]. In order to simplify the notation we omit the suffix k and consider the univariate function $\phi(\alpha) = f(x + \alpha p)$. With this notation the Wolfe conditions (1.1) and (1.2) may be written in the form

$$\phi(\alpha) \le \phi(0) + \alpha\eta_A \phi'(0), \quad \text{and} \quad |\phi'(\alpha)| \le \eta_W |\phi'(0)|.$$

Much of the theory associated with a Wolfe line search is based on the properties of the auxiliary function

$$\omega(\alpha) = \phi(\alpha) - \big(\phi(0) + \alpha\eta_A\phi'(0)\big), \quad \text{with} \quad \omega'(\alpha) = \phi'(\alpha) - \eta_A\phi'(0).$$

Moré and Sorensen [29] show that a minimizer of this function at which ω is negative satisfies the Wolfe conditions. The function ω and its relationship to ϕ are depicted in Figure 1.

The first stage of a Wolfe line search is motivated by the following proposition.

Proposition 2.1. *Let $\{\,\alpha_i\,\}_{i=0}^{\infty}$ be a strictly monotonically increasing sequence with $\alpha_0 = 0$. Let ϕ and ω be continuously differentiable univariate functions such that $\phi'(0) < 0$ and $\omega(\alpha) = \phi(\alpha) - \big(\phi(0) + \alpha\eta_A\phi'(0)\big)$ with $0 < \eta_A < 1$. If there exists a least bounded index j such that at least one of the following conditions is true:*

(a) *α_j is a Wolfe step;*

(b) *α_j is not an Armijo step, i.e., $\omega(\alpha_j) > 0$;*

(c) *$\omega(\alpha_j) \ge \omega(\alpha_{j-1})$; or*

(d) *$\omega'(\alpha_j) \ge 0$,*

then there exists a Wolfe step $\alpha^* \in [\alpha_{j-1}, \alpha_j]$. Collectively, (a)–(d) are called the stage-one conditions.

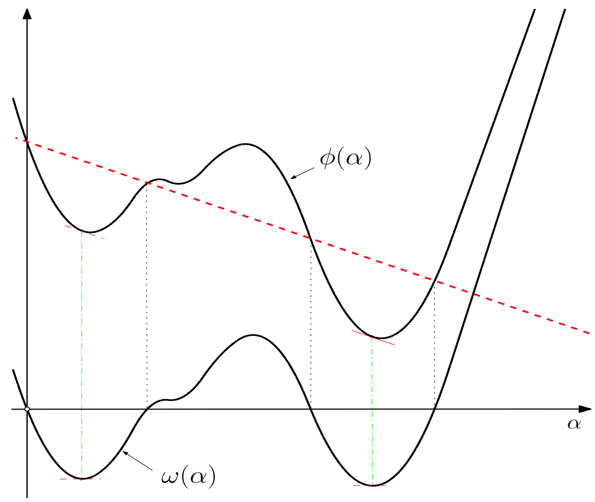

Figure 1: The graph depicts $\phi(\alpha) = f(x + \alpha p)$ as a function of positive α, with the shifted function $\omega(\alpha) = \phi(\alpha) - \big(\phi(0) + \alpha\eta_A\phi'(0)\big)$ superimposed. The dashed line represents the affine function $\phi(0) + \alpha\eta_A\phi'(0)$.

Proof. Observe that if $j > 1$ then α_{j-1} must satisfy none of the conditions (a)–(d), otherwise j would not be the least index. This implies that $\omega(\alpha_{j-1}) \le 0$ from (b), and $\omega'(\alpha_{j-1}) < 0$ from (d). The following simple argument shows that

$$\phi'(\alpha_{j-1}) \le \eta_W\phi'(0) < 0, \tag{2.1}$$

for all $j \ge 1$. First, the definition of ω gives $\omega'(\alpha_{j-1}) = \phi'(\alpha_{j-1}) - \eta_A\phi'(0) < 0$, which implies that $\phi'(\alpha_{j-1}) < 0$. As (a) cannot hold at α_{j-1}, it must be the case that at least one of the Wolfe conditions $\phi'(\alpha_{j-1}) \ge \eta_W\phi'(0)$ and $\phi'(\alpha_{j-1}) \le -\eta_W\phi'(0)$ is not satisfied. It has already been shown that $\phi'(\alpha_{j-1}) < 0$, which means the second inequality is satisfied and so (2.1) must hold. Finally, if $j = 1$ then $\omega'(\alpha_{j-1}) = \phi'(0) < \eta_W\phi'(0)$. The inequality (2.1) is used in the proofs that follow.

Case 1. If (a) is true, the proposition is true trivially.

Case 2. If (b) is true, then $\omega(\alpha_j) > 0$. This, and the fact that $\omega(\alpha_{j-1}) \le 0$ implies that the scalar α_m such that

$$\alpha_m = \sup\ \big\{\, \alpha \in [\alpha_{j-1}, \alpha_j] : \omega(\beta) \le 0 \ \text{ for all } \ \beta \in [\alpha_{j-1}, \alpha] \,\big\}$$

is well-defined. Inequality (2.1) gives

$$\omega'(\alpha_{j-1}) = \phi'(\alpha_{j-1}) - \eta_A\phi'(0) < \phi'(\alpha_{j-1}) - \eta_W\phi'(0) < 0,$$

and it must be that $\alpha_{j-1} < \alpha_m < \alpha_j$ and $\omega(\alpha_m) = 0$. As f is differentiable, $\omega(\alpha_{j-1}) \le \omega(\alpha_m)$ and $\omega'(\alpha_{j-1}) < 0$, then, by the mean-value theorem, there exists $\alpha^* \in [\alpha_{j-1}, \alpha_m]$ such that $\omega'(\alpha^*) = 0$ and $\omega(\alpha^*) < 0$. Then

$$\eta_W\phi'(0) < \eta_A\phi'(0) = \phi'(\alpha^*) < 0,$$

and α^* is a Wolfe step.

Case 3. If (c) is true and (b) is false, assume first that $\omega(\alpha) \leq 0$ for all $\alpha \in [\alpha_{j-1}, \alpha_j]$. As $\omega(\alpha_j) \geq \omega(\alpha_{j-1})$ by assumption, and $\omega'(\alpha_{j-1}) < 0$, there must exist a step $\alpha^* \in [\alpha_{j-1}, \alpha_j]$ such that $\omega'(\alpha^*) = 0$. As $\omega(\alpha^*) \leq 0$, α^* is a Wolfe step. If $\omega(\alpha) > 0$ for some $\alpha \in [\alpha_{j-1}, \alpha_j]$, the preceding argument may be used with α in place of α_j.

Case 4. If (d) is true and (b) is false, assume without loss of generality that $\omega(\alpha) \leq 0$ for all $\alpha \in [\alpha_{j-1}, \alpha_j]$, as in the preceding case. As $\omega'(\alpha_{j-1}) < 0$ and $\omega'(\alpha_j) \geq 0$, the continuity of ω' implies that there exists $\alpha^* \in [\alpha_{j-1}, \alpha_j]$ such that $\omega'(\alpha^*) = 0$. As $\omega(\alpha^*) \leq 0$, α^* is a Wolfe step. ∎

Note that the converse result is not true, i.e., there may be a Wolfe step in the interval $[0, \alpha_1]$ even though none of the stage-one conditions are satisfied for $j = 1$. The second proposition follows.

Proposition 2.2. *Let $\mathcal{I}$ be an interval with distinct endpoints α_{low} and α_{high} in arbitrary order. Let ϕ and ω be continuously differentiable univariate functions such that $\omega(\alpha) = \phi(\alpha) - \big(\phi(0) + \alpha\eta_A\phi'(0)\big)$ with $0 < \eta_A < 1$. If α_{low} and α_{high} satisfy all of the following conditions (collectively called the stage-two conditions):*

(a) $\omega(\alpha_{\text{low}}) \leq 0$ *(i.e., α_{low} is an Armijo step);*

(b) $\omega(\alpha_{\text{low}}) \leq \omega(\alpha_{\text{high}})$ *if α_{high} is an Armijo step; and*

(c) $\omega'(\alpha_{\text{low}})(\alpha_{\text{high}} - \alpha_{\text{low}}) < 0$*,*

then there exists a Wolfe step $\alpha^* \in \mathcal{I}$.

Proof. The proof is similar to that of Proposition 2.1. ∎

If the initial step α_1 is not a Wolfe step—here subscripts will refer to iterations within the first stage of the line search algorithm and $\alpha_0 = 0$ by definition—successively larger step lengths are computed, α_2, α_3, $\dots$, α_{j-1}, α_j, α_{j+1}, $\dots$, $\alpha_{j_{\max}} = \alpha_{\max}$ until one of the stage-one conditions is satisfied.

By Proposition 2.1, if one of the stage-one conditions is satisfied during iteration j, then the interval $[\alpha_{j-1}, \alpha_j]$ must contain a Wolfe step. At this point, the line search algorithm moves on to the second stage (or terminates successfully if stage-one condition (a) is satisfied).

If, after a finite number of iterations, the algorithm reaches $\alpha_{j_{\max}} = \alpha_{\max}$ and none of the conditions have been satisfied, it terminates with the Armijo step associated with the least computed function value. An Armijo step can always be found because not satisfying the conditions at j implies that α_j is an Armijo step.

The computations of a Wolfe line search may be organized into two "functions" associated with the stage-one and stage-two conditions. If the first stage finds an interval that contains a Wolfe step, the first-stage function passes the endpoints α_{j-1} and α_j to the second-stage function, which labels them α_{low} and α_{high}. Next, the second-stage function interpolates the endpoints to calculate a best-guess step in the interval, α_{new}. The second-stage function is called recursively using α_{new} and an existing endpoint, labeling them so that the stage-two conditions hold again. This is repeated until α_{new} is a Wolfe step or until the function has been called a fixed number of times. In practice, it rarely takes more than 1 or 2 interpolations to find a Wolfe step.

Algorithm 1 gives a schematic outline of a Wolfe line search.

Algorithm 1 Schematic outline of a Wolfe line search.

```
function WOLFE_LINE_SEARCH(α)
    Choose η_A ∈ (0, 1), η_W ∈ (η_A, 1), α_max ∈ (0, +∞);
    α ← min{α, α_max};   α_old ← 0;
    while α is not a Wolfe step and α ≠ α_max do
        if ω(α) > 0 or ω(α) ≥ ω(α_old) then
            α ← Stage_Two(α_old, α); break;
        end if
        if ω′(α) ≥ 0 then
            α ← Stage_Two(α, α_old); break;
        end if
        α_old ← α;   Choose α ∈ (α_old, α_max];        [Increase α towards α_max]
    end while
    return α;
end function
function STAGE_TWO(α_low, α_high)
    Choose α_new in the interval defined by α_low and α_high;
    if α_new is a Wolfe step then
        return α_new;
    end if
    if ω(α_new) > 0 or ω(α_new) ≥ ω(α_low) then
        return Stage_Two(α_low, α_new);
    end if
    if ω′(α_new)(α_high − α_low) < 0 then
        return Stage_Two(α_new, α_high);
    else
        return Stage_Two(α_new, α_low);
    end if
end function
```

A practical implementation of a Wolfe line search is very complex. There are many ways to interpolate to obtain a new point in the second stage. Finite precision usually forces some sort of safeguarding during interpolation and gives rise to a whole host of issues, including how to handle cases when the function or step length are changing by a value near or less than machine precision. See, e.g., Brent [6], Hager [24], Ghosh and Hager [17], and Moré and Thuente [30] for more details.

3. The Quasi-Wolfe Line Search

As projected-search methods perform a line search on the piecewise-differentiable function $\psi_k(\alpha) = f\big(x_k(\alpha)\big) = f\big(\mathbf{proj}_{\Omega}(x_k + \alpha p_k)\big)$, it is not possible for such methods to use a conventional Wolfe line search. In this section we consider a new step type, called a *quasi-Wolfe step*, that is designed to extend the benefits of a Wolfe line search to projected-search methods.

3.1. Piecewise-differentiable functions: the quasi-Wolfe step

Performing a line search on the univariate function

$$\psi_k(\alpha) = f\big(x_k(\alpha)\big) = f\big(\mathbf{proj}_{\Omega}(x_k + \alpha p_k)\big),$$

instead of $\phi_k(\alpha) = f(x_k + \alpha p_k)$, is a substantially more difficult task because ψ_k is only piecewise differentiable, with a finite number of jump discontinuities in the derivative. Propositions 2.1 and 2.2 established in the preceding section cannot be used to guarantee a Wolfe step in the nondifferentiable case because they use the mean-value theorem and require the line-search function to be differentiable.

In the following discussion, the suffix k is omitted if the iteration index is not relevant to the discussion. The definition of a quasi-Wolfe step involves the left and right derivatives $\psi'_-(\alpha)$ and $\psi'_+(\alpha)$ of ψ at α, which are defined as

$$\psi'_-(\alpha) = \lim_{\beta\to\alpha_-} \psi'(\beta) \quad \text{and} \quad \psi'_+(\alpha) = \lim_{\beta\to\alpha_+} \psi'(\beta).$$

The following lemma used in Proposition 3.2 below is stated here without proof.

Lemma 3.1. *Let a, $b \in \mathbb{R}$ be such that $0 \le a < b$, and assume that θ is a univariate, continuous, piecewise-differentiable function with a finite number of jump discontinuities in the derivative.*

(a) *If $\theta'_+(a) \le 0$ and $\theta(a) \le \theta(b)$, then there exists an $x \in [a, b]$ such that*

$$\theta'_-(x) \le 0 \le \theta'_+(x).$$

(b) *If $\theta'_+(a) \le 0$ and $\theta'_-(b) \ge 0$ then there exists an $x \in [a, b]$ such that*

$$\theta'_-(x) \le 0 \le \theta'_+(x).$$

If θ is differentiable at x then (a) and (b) hold with equality throughout. ∎

A step α is called a *quasi-Wolfe step* if it satisfies the Armijo condition

$$\psi(\alpha) \le \psi(0) + \alpha\eta_A \psi'_+(0),$$

and at least one of the following conditions:

(C$_1$) $|\psi'_-(\alpha)| \le \eta_W |\psi'_+(0)|$;

($\mathbf{C}_2$) $|\psi'_+(\alpha)| \le \eta_W |\psi'_+(0)|$;

($\mathbf{C}_3$) ψ is not differentiable at α and $\psi'_-(\alpha) \le 0 \le \psi'_+(\alpha)$.

Figure 2 depicts three examples of a kink point that satisfies the quasi-Wolfe conditions. A weak quasi-Wolfe step is defined in a similar manner by modifying ($\mathbf{C}_1$) and ($\mathbf{C}_2$).

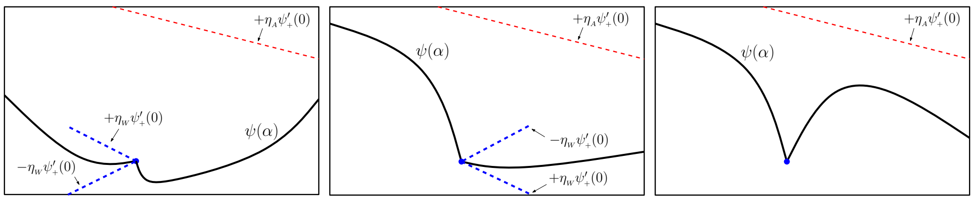

Figure 2: Three examples of a kink point satisfying the quasi-Wolfe conditions. The left, center and right figures depict kink points satisfying conditions **($\mathbf{C}_1$)**, **($\mathbf{C}_2$)** and **($\mathbf{C}_3$)** respectively. The slope of each dashed line is marked.

The properties of the new line search are characterized by extending the framework for the differentiable case. In particular, the discussion makes extensive use of the auxiliary function

$$\omega(\alpha) = \psi(\alpha) - \big(\psi(0) + \alpha\eta_A\psi'_+(0)\big), \quad \text{with} \quad \omega'_\pm(\alpha) = \psi'_\pm(\alpha) - \eta_A\psi'_+(0). \tag{3.1}$$

Proposition 3.1. *Let f be a scalar-valued continuously differentiable function defined on $\Omega = \{\, x \in \mathbb{R}^n : \ell \le x \le u \,\}$. Assume that $x_0 \in \Omega$ is chosen such that the level set $\mathcal{L}\big(f(x_0)\big)$ is closed and bounded, and assume that $\{\, p_k \,\}$ is a sequence of feasible descent directions. If $0 < \eta_A < \eta_W < 1$, then at every iteration k either there exists an $\alpha_L > 0$ and an interval (α_L, α_U) such that every $\alpha \in (\alpha_L, \alpha_U)$ is a quasi-Wolfe step, or there exists a quasi-Wolfe step that satisfies the condition ($\mathbf{C}_3$).*

Proof. First, we show that there exists a positive scalar σ such that the function ω of (3.1) satisfies $\omega(\alpha) < 0$ for all $\alpha \in (0, \sigma)$. As $\psi'_+(0) = \nabla f(x_k)^T p_k < 0$ and $\eta_A < 1$, it must hold that

$$\omega'_+(0) = (1 - \eta_A)\psi'_+(0) < 0,$$

in which case there must be a scalar σ ($\sigma > 0$) such that $\omega(\alpha) < 0$ for all $\alpha \in (0, \sigma)$. It follows that there exists a $\sigma_1 \in (0, \sigma)$ such that $\omega(\sigma_1) < 0$.

From the compactness of the level set $\mathcal{L}(f(x_0))$, $\psi(\alpha)$ is bounded below by some constant ψ_{low}, i.e., $\psi(\alpha) \ge \psi_{\text{low}}$ for all $\alpha \in [0, \infty)$. As $\psi(0) + \alpha\eta_A\psi'_+(0) \to -\infty$ as $\alpha \to \infty$, there must exist a positive σ_2 such that $\psi(0) + \sigma_2\eta_A\psi'_+(0) = \psi_{\text{low}}$, and we have

$$\omega(\sigma_2) = \psi(\sigma_2) - \psi(0) - \sigma_2\eta_A\psi'_+(0) = \psi(\sigma_2) - \psi_{\text{low}} \ge 0.$$

Given scalars σ_1 and σ_2 ($0 \le \sigma_1 < \sigma_2$) such that $\omega(\sigma_1) < 0$ and $\omega(\sigma_2) \ge 0$, the intermediate-value theorem states that there must exist at least one positive α such that $\omega(\alpha) = 0$. Let β denote the least positive root of $\omega(\alpha) = 0$, then $\omega(\alpha) < 0$ for all $\alpha \in (0, \beta)$. As $\omega(0) = 0$, $\omega(\beta) = 0$, and $\omega'_+(0) < 0$, by Lemma 3.1 (a), there exists an $\xi \in (0, \beta)$ such that

$$\omega'_-(\xi) \leq 0 \leq \omega'_+(\xi), \quad \text{or, equivalently,} \quad \psi'_-(\xi) \leq \eta_A \psi'_+(0) \leq \psi'_+(\xi).$$

By construction, $\xi \in (0, \beta)$, which implies that $\omega(\xi) < 0$, or equivalently, $\psi(\xi) \leq \psi(0) + \xi\eta_A\psi'_+(0)$, and ξ satisfies the Armijo condition. If $\psi'_+(\xi) \leq 0$, then the inequality $\eta_A < \eta_W$ implies that ξ is a quasi-Wolfe step that satisfies the derivative condition ($\mathbf{C}_2$). By the piecewise continuity of $\psi'_+(\alpha)$, there exists an $\alpha_L > 0$ and an interval (α_L, α_U) such that every $\alpha \in (\alpha_L, \alpha_U)$ is a quasi-Wolfe step. Otherwise, if $\psi'_+(\xi) > 0$, then ξ is a quasi-Wolfe step that satisfies the condition ($\mathbf{C}_3$). ∎

Proposition 3.2. *Let $\{\,\alpha_i\,\}_{i=0}^{\infty}$ be a strictly monotonically increasing sequence with $\alpha_0 = 0$. Let ψ be a continuous piecewise-differentiable univariate function whose derivative has a finite number of jump discontinuities. Assume that $\psi'_+(0) < 0$ and define $\omega(\alpha) = \psi(\alpha) - \big(\psi(0) + \alpha\eta_A\psi'_+(0)\big)$ with $0 < \eta_A < 1$. If there exists a least bounded index j such that at least one of the following "stage-one" conditions is true:*

(a) *α_j is a quasi-Wolfe step;*

(b) *α_j is not an Armijo step, i.e., $\omega(\alpha_j) > 0$;*

(c) *$\omega(\alpha_j) \geq \omega(\alpha_{j-1})$; or*

(d) *$\omega'_-(\alpha_j) \geq 0$,*

then there exists a quasi-Wolfe step $\alpha^* \in [\alpha_{j-1}, \alpha_j]$.

Proof. Observe that if $j > 1$, then α_{j-1} must satisfy none of the conditions (a)–(d), otherwise j would not be the least index. This implies that $\omega(\alpha_{j-1}) \leq 0$ from (b), and $\omega'_-(\alpha_{j-1}) < 0$ from (d).

The first step is to show that

$$\psi'_+(\alpha_{j-1}) < \eta_W\psi'_+(0) < 0, \tag{3.2}$$

for all $j \geq 1$. If $\psi'(\alpha_{j-1})$ exists, then $\psi'_+(\alpha_{j-1}) = \psi'_-(\alpha_{j-1})$ and $\omega'_+(\alpha_{j-1}) = \psi'_+(\alpha_{j-1}) - \eta_A\psi'_+(0) < 0$, which means that $\psi'_-(\alpha_{j-1}) = \psi'_+(\alpha_{j-1}) < 0$. As (a) cannot hold at α_{j-1}, it must be the case that at least one of the ($\mathbf{C}_2$) conditions $\psi'_+(\alpha_{j-1}) \geq \eta_W\psi'_+(0)$ and $\psi'_+(\alpha_{j-1}) \leq -\eta_W\psi'_+(0)$ cannot hold. It has already been shown that $\psi'_+(\alpha_{j-1}) < 0$ so that the second inequality is satisfied, which implies that (3.2) holds. If $\psi'(\alpha_{j-1})$ does not exist, then (d) implies that $\omega'_-(\alpha_{j-1}) = \psi'_-(\alpha_{j-1}) - \eta_A\psi'_+(0) < 0$, in which case $\psi'_-(\alpha_{j-1}) < 0$ because $\psi'_+(0) < 0$ by assumption. As ($\mathbf{C}_3$) cannot hold at α_{j-1}, at least one of the inequalities $\psi'_-(\alpha_{j-1}) > 0$ or $\psi'_+(\alpha_{j-1}) < 0$ must apply, which means that $\psi'_+(\alpha_{j-1}) < 0$. Now, if ($\mathbf{C}_2$) does not hold at α_{j-1} then $\psi'_+(\alpha_{j-1}) < \eta_W\psi'_+(0) < 0$ or $\psi'_+(\alpha_{j-1}) > -\eta_W\psi'_+(0)$, which again implies that (3.2) holds because $\psi'_+(\alpha_{j-1}) < 0$. Finally, if $j = 1$ then $\psi'_+(\alpha_{j-1}) = \psi'_+(0) < \eta_W \psi'_+(0)$. It follows that (3.2) holds for all $j \geq 1$. The inequality (3.2) is used in the proofs that follow.

Case 1. If (a) is true, the proposition holds trivially.

Case 2. If (b) is true, then $\omega(\alpha_j) > 0$. This, and the fact that $\omega(\alpha_{j-1}) \leq 0$ from (b) implies that the scalar α_m such that

$$\alpha_m = \sup \; \big\{ \, \alpha \in [\alpha_{j-1}, \alpha_j] : \omega(\beta) \leq 0 \;\text{ for all }\; \beta \in [\alpha_{j-1}, \alpha] \, \big\}$$

is well-defined. Inequality (3.2) gives

$$\omega'_+(\alpha_{j-1}) = \psi'_+(\alpha_{j-1}) - \eta_A \psi'_+(0) < \psi'_+(\alpha_{j-1}) - \eta_W \psi'_+(0) < 0,$$

and it must be that $\alpha_{j-1} < \alpha_m < \alpha_j$ and $\omega(\alpha_m) = 0$. As $\omega(\alpha_{j-1}) \leq \omega(\alpha_m)$ and $\omega'_+(\alpha_{j-1}) < 0$, part (a) of Lemma 3.1 implies that there exists an $\alpha^* \in [\alpha_{j-1}, \alpha_m]$ such that

$$\omega'_-(\alpha^*) \leq 0 \leq \omega'_+(\alpha^*).$$

This implies that

$$\psi'_-(\alpha^*) \leq \eta_A \psi'_+(0) \leq \psi'_+(\alpha^*).$$

By the definition of α_m, α^* is an Armijo step. Observe that $\psi'_-(\alpha^*) < 0$. Therefore, if $\psi'_+(\alpha^*) \geq 0$, then α^* must be a quasi-Wolfe step. Alternatively, if $\psi'_+(\alpha^*) < 0$, then

$$\eta_W \psi'_+(0) < \eta_A \psi'_+(0) \leq \psi'_+(\alpha^*) < 0,$$

and α^* is a quasi-Wolfe step.

Case 3. If (c) is true and (b) is false, then $\omega(\alpha_j) \geq \omega(\alpha_{j-1})$ and $\omega(\alpha_j) \leq 0$. Without loss of generality, assume that $\omega(\alpha) \leq 0$ for all $\alpha \in [\alpha_{j-1}, \alpha_j]$. By (3.2),

$$\omega'_+(\alpha_{j-1}) = \psi'_+(\alpha_{j-1}) - \eta_A \psi'_+(0) < \psi'_+(\alpha_{j-1}) - \eta_W \psi'_+(0) < 0.$$

This last inequality and $\omega(\alpha_j) \geq \omega(\alpha_{j-1})$ imply that part (a) of Lemma 3.1 can be applied to establish the existence of an Armijo step $\alpha^* \in [\alpha_{j-1}, \alpha_j]$ such that

$$\omega'_-(\alpha^*) \leq 0 \leq \omega'_+(\alpha^*).$$

This implies that

$$\psi'_-(\alpha^*) \leq \eta_A \psi'_+(0) \leq \psi'_+(\alpha^*).$$

The same argument used for the preceding case shows that α^* is a quasi-Wolfe step.

Case 4. Finally, consider the case where (d) is true and (b) is false, i.e., $\omega'_-(\alpha_j) \geq 0$ and $\omega(\alpha_j) \leq 0$. Assume without loss of generality that $\omega(\alpha) \leq 0$ for all $\alpha \in [\alpha_{j-1}, \alpha_j]$. Part (b) of Lemma 3.1 establishes the existence of a step $\alpha^* \in [\alpha_{j-1}, \alpha_j]$ such that

$$\omega'_-(\alpha^*) \leq 0 \leq \omega'_+(\alpha^*).$$

As $\omega(\alpha) \leq 0$ for all $\alpha \in [\alpha_{j-1}, \alpha_j]$ it must be the case that α^* is an Armijo step. It follows the same argument from the preceding case that α^* is a quasi-Wolfe step. ∎

Proposition 3.3. *Let $\mathcal{I}$ be an interval with distinct endpoints $\alpha_{\rm low}$ and $\alpha_{\rm high}$ in arbitrary order. Let ψ and ω be scalar functions such that $\omega(\alpha) = \psi(\alpha) - \big(\psi(0) + \alpha\eta_A\psi'_+(0)\big)$ with ψ a continuous, piecewise-differentiable univariate function with a finite number of jump discontinuities in its derivative. If $\alpha_{\rm low}$ and $\alpha_{\rm high}$ satisfy the conditions:*

(a) *$\alpha_{\rm low}$ is an Armijo step;*

(b) *$\omega(\alpha_{\rm low}) \le \omega(\alpha_{\rm high})$ if $\alpha_{\rm high}$ is an Armijo step; and*

(c) *$\omega'_+(\alpha_{\rm low}) < 0$ if $\alpha_{\rm low} < \alpha_{\rm high}$ or $\omega'_-(\alpha_{\rm low}) > 0$ if $\alpha_{\rm low} > \alpha_{\rm high}$,*

then there exists a quasi-Wolfe step $\alpha^* \in \mathcal{I}$.

Proof. The proof is similar to that of Proposition 3.2. ∎

Although the implementation of a quasi-Wolfe line search is similar to that of a Wolfe line search, there are a number of crucial practical issues associated with the potential nondifferentiability of the line-search function. These issues include the definition of the derivatives of the line-search function, the computation of a new estimate of a quasi-Wolfe step and the properties of the quasi-Newton update.

3.2. Derivatives of the line-search function

For the differentiable case, the slope of ϕ is the directional derivative of f at $x + \alpha p$ in direction p, i.e., $\phi'(\alpha) = \nabla f(x+\alpha p)^T p$. Consider the function $\psi(\alpha) = f\big(x(\alpha)\big) = f(\mathbf{proj}_\Omega(x+\alpha p))$ at a point $x(\alpha)$ at which $\psi(\alpha)$ is differentiable. The chain rule gives

$$\psi'(\alpha) = \frac{d}{d\alpha} f\big(x(\alpha)\big) = \nabla f\big(x(\alpha)\big)^T \frac{d}{d\alpha} x(\alpha) = \nabla f\big(x(\alpha)\big)^T x'(\alpha). \tag{3.3}$$

By definition, $x(\alpha) = \mathbf{proj}_\Omega(x+\alpha p)$, with

$$x_i(\alpha) = \begin{cases} \ell_i & \text{if } \ x_i + \alpha p_i < \ell_i, \\ u_i & \text{if } \ x_i + \alpha p_i > u_i, \\ x_i + \alpha p_i & \text{if } \ \ell_i \le x_i + \alpha p_i \le u_i. \end{cases} \tag{3.4}$$

Differentiating with respect to α gives

$$x_i'(\alpha) = \begin{cases} 0 & \text{if } \ x_i + \alpha p_i < \ell_i, \\ 0 & \text{if } \ x_i + \alpha p_i > u_i, \\ p_i & \text{otherwise.} \end{cases} \tag{3.5}$$

The vector $x'(\alpha)$ may be expressed in terms of the *projected direction of p at x*, which is defined as

$$[\,P_x(p)\,]_i = \begin{cases} 0 & \text{if } \ x_i = \ell_i \ \text{ and } \ p_i < 0, \\ 0 & \text{if } \ x_i = u_i \ \text{ and } \ p_i > 0, \\ p_i & \text{otherwise.} \end{cases} \tag{3.6}$$

This vector represents the projection of p onto the closure of the set of feasible directions at $x(\alpha)$. By assumption, x is feasible, α is positive and p is a feasible direction. It follows that if $x_i + \alpha p_i < \ell_i$ then $p_i < 0$. Similarly, if $x_i + \alpha p_i > u_i$, then $p_i > 0$. This implies $x'(\alpha)$ may be written in the form

$$x_i'(\alpha) = \begin{cases} 0 & \text{if } \ x_i(\alpha) = \ell_i \ \text{ and } \ p_i < 0, \\ 0 & \text{if } \ x_i(\alpha) = u_i \ \text{ and } \ p_i > 0, \\ p_i & \text{otherwise.} \end{cases} \tag{3.7}$$

Using this expression for $x_i'(\alpha)$ in (3.3) in conjunction with the definition (3.6) gives

$$\psi'(\alpha) = \nabla f\big(x(\alpha)\big)^T P_{x(\alpha)}(p).$$

If $\psi(\alpha)$ is not differentiable at α then there must be at least one index i such that

$$(x_i + \alpha p_i = \ell_i \ \text{ and } \ p_i < 0) \quad \text{or} \quad (x_i + \alpha p_i = u_i \ \text{ and } \ p_i > 0).$$

An α satisfying one of these conditions is called a *kink step with respect to* i. If $\psi'(\alpha)$ does not exist, then α is necessarily a kink step; conversely, if α is a kink step, it is almost always the case that $\psi'_-(\alpha) \neq \psi'_+(\alpha)$.

If α is a kink step with respect to i then $x_i + \beta p_i$ is infeasible for all $\beta > \alpha$ and $x_i'(\beta) = 0$ (cf. (3.4) and (3.5)). This implies that for *any* α, it must hold that $\lim_{\beta \to \alpha+} x_i'(\beta) = P_{x(\alpha)}(p)$, and

$$\psi'_+(\alpha) = \nabla f\big(x(\alpha)\big)^T P_{x(\alpha)}(p)$$

from (3.7). It follows that the slope of ψ' going forward from a kink step α is the directional derivative of f at $x(\alpha)$ in the projected direction $P_{x(\alpha)}(p)$. If α is a kink step, the projected direction $P_{x(\alpha)}(p)$ is denoted by $P^+_{x(\alpha)}(p)$ to emphasize the discontinuity of $x'(\alpha)$ at α.

In order to compute $\psi'_-(\alpha)$, it is necessary to consider the values of $x'(\beta)$ as β approaches α from below. If α is a kink step with respect to i then $x_i + \beta p_i$ is feasible for all β sufficiently close to α and if follows from (3.5) and (3.7) that $x_i'(\beta) = p_i$. If this value is combined with the components of $x_i'(\beta)$ associated with the differentiable case, we obtain

$$\psi'_-(\alpha) = \nabla f\big(x(\alpha)\big)^T P^-_{x(\alpha)}(p),$$

where

$$\Big[\, P^-_{x(\alpha)}(p) \,\Big]_i = \begin{cases} p_i & \text{if } \ \alpha \ \text{ is a kink step with respect to } \ i, \\ \big[\, P_{x(\alpha)}(p) \,\big]_i & \text{otherwise.} \end{cases}$$

This implies that there is a jump of $\big|p_i \nabla_i f\big(x(\alpha)\big)\big|$ in the derivative of ψ at a kink step with respect to i.

3.3. Computing a quasi-Wolfe step

As in the Wolfe line search [32], the implementation of the quasi-Wolfe line search implicitly consists of two stages. The first stage begins with an initial step length α_0 and continues with steps of increasing magnitude until one of three things occurs: (i) an acceptable step length is found; (ii) an interval that contains a quasi-Wolfe step is found; or (iii) the step is considered to be unbounded. If the first step terminates with a bounded step, the second stage repeatedly calls a function `Stage_Two`(α_{low}, α_{high}), where

(a) the interval bounded by α_{low} and α_{high} contains a quasi-Wolfe step;

(b) among all the step lengths generated so far, α_{low} gives the least value of ω;

(c) α_{high} is chosen so that $\omega'_+(\alpha_{\text{low}}) < 0$ if $\alpha_{\text{low}} < \alpha_{\text{high}}$, or $\omega'_-(\alpha_{\text{low}}) > 0$ if $\alpha_{\text{low}} > \alpha_{\text{high}}$.

An interval with end points α_{low} and α_{high} is known as an *interval of uncertainty*. Similarly, α_{low} and α_{high} are said to *bracket* a quasi-Wolfe step. In practice, an upper bound $\alpha_{\max}$ is imposed on the value of α_j and the search is terminated if this bound is exceeded during the Stage-one iterations. If the line search terminates at $\alpha_{\max}$ without finding an interval containing a quasi-Wolfe step, then all of the steps computed up to that point are Armijo steps. Algorithm 2 gives a schematic outline of a quasi-Wolfe line search.

A major difference between a Wolfe and quasi-Wolfe line search concerns how interpolation is used to find new steps in the second stage. Each time `Stage_Two`(α_{low}, α_{high}) is invoked, a new trial step α_{new} is generated. In the differentiable case, α_{new} is usually obtained by polynomial interpolation using the value of ϕ and its derivatives at α_{low} and α_{high}. If the line-search function is only piecewise differentiable, there may be kink points between α_{low} and α_{high} in which case a conventional interpolation approach may not provide a good estimate of a quasi-Wolfe step. One strategy to speed convergence in this situation is to search for the kink step (if it exists) between α_{low} and α_{high} that is closest to α_{low}. This approach is justified by the following argument. If a new point α_{new} is not a quasi-Wolfe step, then based on Proposition 2.2, the end points α_{low} and α_{high} are updated to α_{low} and α_{new} in three cases:

Case (1). α_{new} is not an Armijo step;

Case (2). $\omega(\alpha_{\text{new}}) \geq \omega(\alpha_{\text{low}})$;

Case (3). $\omega'_+(\alpha_{\text{new}}) < 0$ if $\alpha_{\text{high}} < \alpha_{\text{low}}$, or $\omega'_-(\alpha_{\text{new}}) > 0$ if $\alpha_{\text{high}} > \alpha_{\text{low}}$.

In these cases, the new interval bounded by α_{low} and α_{new} will not contain a kink step. In the remaining case:

Case (4). $\omega'_+(\alpha_{\text{new}}) \geq 0$ if $\alpha_{\text{high}} < \alpha_{\text{low}}$, or $\omega'_-(\alpha_{\text{new}}) \leq 0$ if $\alpha_{\text{high}} > \alpha_{\text{low}}$,

Algorithm 2 Schematic outline of a quasi-Wolfe line search.

function QUASI_WOLFE_LINE_SEARCH(α)
    Choose $\eta_A \in (0, 1)$, $\eta_W \in (\eta_A, 1)$, $\alpha_{\max} \in (0, +\infty)$;
    $\alpha \leftarrow 1$; $\alpha_{\text{old}} \leftarrow 0$;
    while α is not a quasi-Wolfe step and $\alpha \neq \alpha_{\max}$ **do**
        if $\omega(\alpha) > 0$ or $\omega(\alpha) \geq \omega(\alpha_{\text{old}})$ **then**
            $\alpha \leftarrow$ `Stage_Two`$(\alpha_{\text{old}}, \alpha)$; break;
        end if
        if $\omega'_-(\alpha) \geq 0$ **then**
            $\alpha \leftarrow$ `Stage_Two`$(\alpha, \alpha_{\text{old}})$; break;
        end if
        $\alpha_{\text{old}} \leftarrow \alpha$;   Choose $\alpha \in (\alpha_{\text{old}}, \alpha_{\max}]$;   [Increase α towards $\alpha_{\max}$]
    end while
    return α;
end function
function STAGE_TWO($\alpha_{\text{low}}, \alpha_{\text{high}}$)
    Choose α_{new} in the interval defined by α_{low} and α_{high};
    if α_{new} is a quasi-Wolfe step **then**
        return α_{new};
    end if
    if α_{new} is not an Armijo step or $\omega(\alpha_{\text{new}}) \geq \omega(\alpha_{\text{low}})$ **then**
        return `Stage_Two`$(\alpha_{\text{low}}, \alpha_{\text{new}})$;
    end if
    if $\omega'_+(\alpha_{\text{new}}) < 0$ and $\alpha_{\text{low}} < \alpha_{\text{high}}$ **then**
        return `Stage_Two`$(\alpha_{\text{new}}, \alpha_{\text{high}})$;
    else if $\omega'_-(\alpha_{\text{new}}) > 0$ and $\alpha_{\text{low}} > \alpha_{\text{high}}$ **then**
        return `Stage_Two`$(\alpha_{\text{new}}, \alpha_{\text{high}})$;
    else
        return `Stage_Two`$(\alpha_{\text{new}}, \alpha_{\text{low}})$;
    end if
end function

the new interval will be bounded by α_{high} and α_{new}, but may contain kink points. However, the new interval must contain at least one fewer kink point.

The search for the kink points proceeds as follows. The first time the function `Stage_Two`(α_{low}, α_{high}) is invoked, the kink steps are computed in $O(n)$ flops from

$$\kappa_i = \begin{cases} (u_i - x_i)/p_i & \text{if } p_i > 0, \\ (\ell_i - x_i)/p_i & \text{if } p_i < 0, \\ \infty & \text{if } p_i = 0. \end{cases}$$

As the interval bounded by α_{low} and α_{high} brackets a quasi-Wolfe step, only the kink steps within that interval need be stored. These steps are then sorted in decreasing order within $O(n \log n)$ operations using a heapsort algorithm (see, e.g., Williams [35], Knuth [26, Section 5.2.3]). The kink step closest to α_{low}, say κ_1^*, is either the smallest or the largest kink step within the interval of uncertainty, depending on whether α_{low} is smaller or greater than α_{high}. Once κ_1^* has been found, the search for κ_l^* $(l > 1)$ is made towards α_{low} starting at the kink step κ_{l-1}^* from the preceding iteration. To prevent the iterations from lingering at **Case (4)** for too long, an upper limit is imposed on the number of consecutive kink steps as trial steps. If this limit is reached, a new trial step is generated by bisection.

Once all the kinks in the interval of uncertainty have been eliminated, conventional polynomial interpolation may be used to generate a new step length. However, some care is necessary to choose the appropriate left or right derivative for use in the interpolation (see Section 3.2).

If there is just one kink step in the interval of uncertainty, α_{new} is set to be that kink step. As the number of kink steps in an interval increases, it becomes more difficult to strike a balance between making effective use of the knowledge they exist and efficiency; for example, if an interval contains 10^6 kink steps, it is not practical to jump to the middle one and repeat on each subinterval.

4. Projected-Search Methods

In this section, we focus on two projected search methods for problem (BC) that use the quasi-Wolfe line search. The first is a limited-memory quasi-Newton method that uses function and gradient values; the second uses second derivatives and combines a primal-dual interior method with a projected search.

4.1. A limited-memory quasi-Newton method

Given an initial $x_0 \in \Omega$, the sequence of iterates $\{\, x_k \,\}$ satisfies $x_{k+1} = x_k(\alpha_k) = \mathbf{proj}_{\Omega}(x_k + \alpha_k p_k)$, where p_k is computed in terms of a direction d_k such that $\nabla f(x_k)^T d_k < 0$. The vector d_k is the unique solution of the subproblem

$$\underset{d}{\text{minimize}}\ \nabla f(x_k)^T d + \tfrac{1}{2} d^T H_k d \quad \text{subject to } d_i = 0 \ \text{ for all } \ i \in \mathcal{W}_k(x_k), \tag{4.1}$$

where H_k a positive-definite limited memory BFGS approximation of $\nabla^2 f(x_k)$, and $\mathcal{W}_k(x)$ is a *working set* of indices of x. The working set is defined as

$$\mathcal{W}_k(x) = \{\, i : x_i < \ell_i + \epsilon_k \ \text{ and } \ \nabla_i f(x) > 0 \ \text{ or } \ x_i > u_i - \epsilon_k \ \text{ and } \ \nabla_i f(x) < 0 \,\},$$

where $\epsilon_k = \min\{\epsilon, \|P_{x_k}\big(-\nabla f(x_k)\big)\|\}$, with ϵ a fixed positive parameter. The matrix H_k is maintained in reduced-Hessian form and is not stored explicitly (for more details, see Gill and Leonard [18], Ferry [11], and Ferry et al. [12]). Once the subproblem (4.1) has been solved, the components of d_k are modified if necessary to give a line-search direction p_k such that $[\,p_k\,]_i \geq 0$ if $[\,x_k\,]_i < \ell_i + \epsilon_k$, and $[\,p_k\,]_i \leq 0$ if $[\,x_k\,]_i > u_i - \epsilon_k$. This additional step guarantees convergence in the situation where iterates approach a boundary point from the interior of the feasible region—a phenomenon known as zigzagging or jamming (see Bertsekas [3]). The vector p_k retains the descent property of d_k. For example, if the solution of (4.1) has $[\,d_k\,]_i \neq 0$ and $[\,x_k\,]_i < \ell_i + \epsilon_k$, then the definition of $\mathcal{W}_k(x)$ implies that $\nabla_i f(x_k) \leq 0$. If $[\,p_k\,]_i > 0$ then $[\,p_k\,]_i = [\,d_k\,]_i$. If $[\,d_k\,]_i < 0$ then $\nabla_i f(x_k)\,[\,d_k\,]_i \geq 0$, and setting $[\,p_k\,]_i = 0$ makes the directional derivative more negative.

An important benefit of the conventional Wolfe conditions in the unconstrained case is that the restriction on the directional derivative guarantees the satisfaction of a necessary condition for the quasi-Newton update to give a positive-definite approximate Hessian. Unfortunately it is not possible to completely guarantee this property in the bound-constrained case, although the likelihood of a skipped update is significantly less than that for a method using an Armijo step. If the next iterate is given by $x_{k+1} = \mathbf{proj}_{\,\Omega}(x_k + \alpha_k p)$, where α_k is a quasi-Wolfe step, then the approximate curvature $\big(\nabla f(x_{k+1}) - \nabla f(x_k)\big)^T(x_{k+1} - x_k)$ need not be greater than zero. It is worth pointing out that this situation is possible only if the path $\mathbf{proj}_{\,\Omega}(x_k + \alpha_k p_k)$ changes direction for some $\alpha \in (0, \alpha_k)$. If it does change direction, $\psi'_+(0)$ and $\psi'_-(\alpha_k)$ may be directional derivatives of f in a direction other than $x_{k+1} - x_k$. For example, using Figure 3, which has lower bounds $x_1 = 0$ and $x_2 = 0$, $\psi'_+(0)$ is a directional derivative of f in direction $[\,p_k\,]_1$ and $\psi'_-(\alpha_k)$ is a directional derivative of f in direction $[\,p_k\,]_2$.

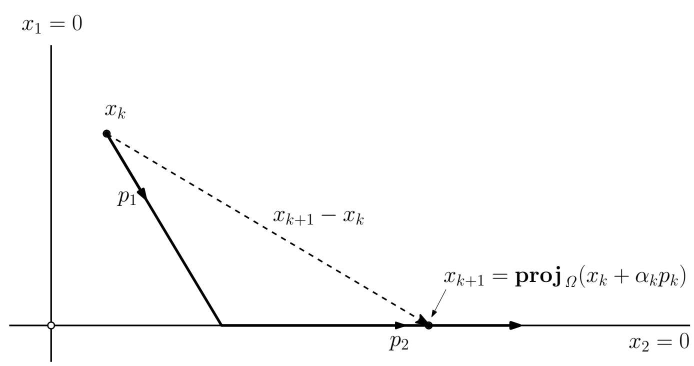

Figure 3: Example with no guarantee of an update for the approximate Hessian.

As a result, if the path changes direction for $\alpha \in (0, \alpha_k)$, then it suffers from the same theoretical downside that an Armijo line search suffers from: there is no guarantee that the approximate Hessian can be updated.

In practice, it seems to be uncommon for an algorithm using a quasi-Wolfe line search to skip the approximate Hessian update. The application of a quasi-Newton method with a quasi-Wolfe line search to 106 bound constrained examples [1]resulted in 53 of the 32440 updates being skipped ($\approx 0.16\%$). This can be compared to 247 of the 37298 updates being skipped ($\approx 0.66\%$) for the Armijo step. (The number of updates reflects the number of iterations needed for convergence.) In all cases, the quasi-Wolfe line and Armijo line searches found a step satisfying the required criteria. For more details of the runs, see Section 4.1.

If it can be shown that an algorithm using a quasi-Wolfe line search correctly identifies the active set at the solution in a finite number of iterations, then, after the active set stabilizes, a quasi-Wolfe line search behaves exactly like a Wolfe line search in the sense that updates to the approximate Hessian are guaranteed. Under such circumstances, and given suitable conditions, the convergence rate of a quasi-Newton method using a quasi-Wolfe line search is q-superlinear.

4.2. A primal-dual interior projected-search method

The primal-dual interior method of Forsgren and Gill [14] is based on minimizing the unconstrained function $M(x, z_1, z_2\,; \mu)$ given by

$$f(x) - \sum_{j=1}^{n} \left\{ \mu \ln\left(x_j - \ell_j\right) + \mu \ln\left([z_1]_j (x_j - \ell_j)\right) - [z_1]_j (x_j - \ell_j) \right\}$$
$$- \sum_{j=1}^{n} \left\{ \mu \ln\left(u_j - x_j\right) + \mu \ln\left([z_2]_j (u_j - x_j)\right) - [z_2]_j (u_j - x_j) \right\}$$

for a sequence of μ-values such that $\mu \to 0$. In order to guarantee that $M(x, z_1, z_2\,; \mu)$ is well-defined, the variables are subject to implicit bound constraints of the form $\ell < x < u$, $z_1 > 0$ and $z_2 > 0$. This implies minimizing $M(x, z_1, z_2\,; \mu)$ is equivalent to solving the bound-constrained problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; M(v\,; \mu) \quad \text{subject to} \quad \ell_v < v < u_v, \tag{IPBC}$$

with $M(v\,; \mu) = M(x, z_1, z_2\,; \mu)$, where

$$v = \begin{pmatrix} x \\ z_1 \\ z_2 \end{pmatrix}, \qquad \ell_v = \begin{pmatrix} \ell \\ 0 \\ 0 \end{pmatrix}, \quad \text{and} \quad u_v = \begin{pmatrix} u \\ +\infty \\ +\infty \end{pmatrix}.$$

At a given point (x, z_1, z_2) such that $\ell < x < u$, $z_1 > 0$ and $z_2 > 0$, let $X_1 = \operatorname{diag}\bigl(x_j - \ell_j\bigr)$, $X_2 = \operatorname{diag}\bigl(u_j - x_j\bigr)$, $Z_1 = \operatorname{diag}([z_1]_j)$, and $Z_2 = \operatorname{diag}([z_2]_j)$. One iteration of Newton's method for minimizing $M(x, z_1, z_2\,; \mu)$ requires solving the equations $\nabla^2 M(v\,; \mu) \Delta v = -\nabla M(v\,; \mu)$. If the diagonal matrices μX_1^{-1} and μX_2^{-1} in the expression for $\nabla^2 M(v\,; \mu)$ are replaced by Z_1 and Z_2, we obtain an approximate Hessian with $n \times n$ principal minor $H_k = \nabla^2 f(x) + X_1^{-1} Z_1 + X_2^{-1} Z_2$. It follows that one iteration of an *approximate* Newton method for minimizing $M(x, z_1, z_2\,; \mu)$ gives

[1]Described in Section 6.

the estimate $(x+\Delta x,\, z_1+\Delta z_1,\, z_2+\Delta z_2)$, where $\Delta z_1 = -X_1^{-1}\big(z_1 \cdot (x+\Delta x-\ell)-\mu e\big)$, $\Delta z_2 = -X_2^{-1}\big(z_2 \cdot (u - x - \Delta x) - \mu e\big)$, and Δx satisfies the equations

$$H_k \Delta x = -\big(\nabla f(x) - \mu X_1^{-1} e + \mu X_2^{-1} e\big). \tag{4.2}$$

The equations for $(\Delta x,\, \Delta z_1,\, \Delta z_2)$ are equivalent to the primal-dual path-following equations for problem (BC). Let v_k denote a point such that $\ell_v < v_k < u_v$ and let Δv_k denote the solution of the approximate Newton equations at v_k. The proposed projected-search line-search method for problem (IPBC) generates a sequence of feasible iterates $\{\, v_k \,\}_{k=0}^{\infty}$ such that $v_{k+1} = \mathbf{proj}_{\Omega_k}(v_k + \alpha_k \Delta v_k)$, where $\mathbf{proj}_{\Omega_k}(x)$ is the projection of v onto the set

$$\Omega_k = \big\{\, v : v_k - \sigma(v_k - \ell_v) \le v \le v_k + \sigma(u_v - v_k) \,\big\}, \tag{4.3}$$

with σ a fixed positive scalar such that $0 < \sigma < 1$. (It is assumed implicitly that the upper bounds are infinite for the z_1 and z_2 components of v.) The quantity σ may be interpreted as the "fraction to the boundary" parameter used in many conventional interior-point methods.

If the matrix H_k of (4.2) is positive definite, then Δv is a descent direction for $M(x, z_1, z_2\,; \mu)$. Otherwise a positive-definite modified matrix $\widehat{H}_k \approx H_k$ must be used. There are a number of alternative approaches for choosing $\widehat{H}_k$ based on computing a factorization of H_k (see, e.g., Greenstadt [23], Gill and Murray [19], Fletcher and Freeman [13], Moré and Sorensen [28], Schnabel and Eskow [34], Forsgren, Gill and Murray [15], and Cheng and Higham [9]. All of these methods compute factors such that $\widehat{H}_k = H_k$ if H_k is sufficiently positive-definite.

5. Convergence Analysis

In this section we consider the convergence properties of projected search methods. First, we consider the convergence of a projected-search method based on a quasi-Armijo line search, which gives a step satisfying the condition (1.4).

Theorem 5.1. (Convergence of the quasi-Armijo line search) *Let f be a scalar-valued continuously differentiable function defined on $\Omega = \{\, x \in \mathbb{R}^n : \ell \le x \le u \,\}$. Assume that $x_0 \in \Omega$ is chosen such that the level set $\mathcal{L}(f(x_0))$ is closed and bounded, and $\{\, x_k \,\}$ is defined by $x_{k+1} = x_k(\alpha_k)$, where α_k is a quasi-Armijo step. Also assume that $\{\, p_k \,\}$ is a sequence of feasible descent directions with $\|p_k\| \le \theta$ for some constant θ independent of k. For any fixed $\epsilon > 0$, define*

$$\epsilon_k = \min\{\epsilon, \|P_{x_k}\big(-\nabla f(x_k)\big)\|\}.$$

If the components of p_k satisfy $[\,p_k\,]_i \ge 0$ if $[\,x_k\,]_i < \ell_i + \epsilon_k$, and $[\,p_k\,]_i \le 0$ if $[\,x_k\,]_i > u_i - \epsilon_k$, then either

$$\lim_{k\to\infty} \big|\nabla f(x_k)^T p_k\big| = 0, \quad \textit{or} \quad \liminf_{k\to\infty} \|P_{x_k}\big(-\nabla f(x_k)\big)\| = 0.$$

Proof. If $\liminf_{k\to\infty} \|P_{x_k}\big(-\nabla f(x_k)\big)\| \neq 0$, then $\liminf_{k\to\infty} \epsilon_k > 0$. We show that

$$\lim_{k\to\infty} |\nabla f(x_k)^T p_k| = 0.$$

Observe that the quasi-Armijo condition (1.4) implies that $\{\, f(x_k)\,\}$ is a strictly decreasing sequence. The set $\mathcal{L}(f(x_0))$ is closed and bounded, which implies that $\{\, f(x_k)\,\}$ converges. It follows that

$$0 = \lim_{k\to\infty} f(x_k) - f(x_{k+1}) \geq \lim_{k\to\infty} \alpha_k \eta_A |\nabla f(x_k)^T p_k| = 0.$$

The proof is by contradiction. Suppose that $|\nabla f(x_k)^T p_k| \not\to 0$ as $k \to \infty$, then there must exist some $\bar\epsilon > 0$ such that $|\nabla f(x_k)^T p_k| > \bar\epsilon$ infinitely often. Let $\mathcal{G} = \{\, k : |\nabla f(x_k)^T p_k| > \bar\epsilon\,\}$, then it must be that $\alpha_k \to 0$ for $k \in \mathcal{G}$. for all $k \in \mathcal{G}$, define the step $\beta_k = \alpha_k/\sigma$. As $\{\, \|p_k\|\,\}$ is uniformly bounded by θ and $\liminf_{k\to\infty} \epsilon_k > 0$, there exists $\bar k$ such that each component of $\beta_k p_k$ satisfies $|\,[\,\beta_k p_k\,]_i\,| < \epsilon_k$ for all $k \geq \bar k$ in $\mathcal{G}$. The assumptions on components of p_k imply that $[\,p_k\,]_i > 0$ only if $u_i - [\,x_k\,]_i \geq \epsilon_k$, and $[\,p_k\,]_i < 0$ only if $[\,x_k\,]_i - \ell_i \geq \epsilon_k$. It follows that for all $k \geq \bar k$ in $\mathcal{G}$, $\ell_i \leq [\,x_k + \beta_k p_k\,]_i \leq u_i$ and $\mathbf{proj}_{\,\Omega}(x_k + \beta_k p_k) = x_k + \beta_k p_k$.

Let $\bar{\mathcal{G}}$ denote the indices $k \geq \bar k$ of iterations at which a reduction in the initial step length was necessary, i.e., $\bar{\mathcal{G}} = \{\, k : t_k > 0,\ k \in \mathcal{G},\ k \geq \bar k\,\}$. Since α_k converges to zero, $\bar{\mathcal{G}}$ must be an infinite set. By definition,

$$f(x_k + \beta_k p_k) = f(\mathbf{proj}_{\,\Omega}(x_k + \beta_k p_k)) > f(x_k) + \beta_k \eta_A \nabla f(x_k)^T p_k, \quad \text{for all } \ k \in \bar{\mathcal{G}}.$$

Adding $-\beta_k \nabla f(x_k)^T p_k$ to both sides and rearranging gives

$$\begin{aligned} f(x_k + \beta_k p_k) - f(x_k) - \beta_k \nabla f(x_k)^T p_k &> -\beta_k (1-\eta_A) \nabla f(x_k)^T p_k \\ &> \beta_k (1-\eta_A)\bar\epsilon, \quad \text{for all } \ k \in \bar{\mathcal{G}}. \end{aligned} \tag{5.1}$$

The Taylor expansion of $f(x_k + \beta_k p_k)$ gives

$$f(x_k+\beta_k p_k) - f(x_k) - \beta_k \nabla f(x_k)^T p_k = \beta_k \int_0^1 (\nabla f(x_k + \tau\beta_k p_k) - \nabla f(x_k))^T p_k \, d\tau. \tag{5.2}$$

If $\|\cdot\|_D$ denotes the norm dual to $\|\cdot\|$, i.e., $\|x\|_D = \max_{v\neq 0} |x^T v|/\|v\|$, then

$$|(\nabla f(x_k + \tau\beta_k p_k) - \nabla f(x_k))^T p_k| \leq \|\nabla f(x_k + \tau\beta_k p_k) - \nabla f(x_k)\|_D \|p_k\|.$$

If this inequality is substituted in (5.2), it then follows from (5.1) that

$$\begin{aligned} (1-\eta_A)\bar\epsilon &< \int_0^1 (\nabla f(x_k + \tau\beta_k p_k) - \nabla f(x_k))^T p_k \, d\tau \\ &\leq \max_{0\leq\tau\leq 1} \|\nabla f(x_k + \tau\beta_k p_k) - \nabla f(x_k)\|_D \|p_k\|, \quad \text{for all } \ k \in \bar{\mathcal{G}}. \end{aligned}$$

The continuity of ∇f implies that there exists some $\tau_k \in [0, \beta_k]$ such that

$$\max_{0\leq\tau\leq 1} \|\nabla f(x_k + \tau\beta_k p_k) - \nabla f(x_k)\|_D = \|\nabla f(x_k + \tau_k p_k) - \nabla f(x_k)\|_D.$$

Then

$$(1 - \eta_A)\bar{\epsilon} < \|\nabla f(x_k + \tau_k p_k) - \nabla f(x_k)\|_D \|p_k\|. \tag{5.3}$$

However, $\alpha_k p_k \to 0$ implies $\tau_k p_k \to 0$ for $k \in \mathcal{G}$, and the continuity of ∇f gives

$$\|\nabla f(x_k + \tau_k p_k) - \nabla f(x_k)\|_D \to 0.$$

As $\{ \|p_k\| \}$ is uniformly bounded above by θ, the right-hand side of (5.3) converges to zero, which gives the required contradiction. ∎

Theorem 5.2. (Convergence of quasi-Wolfe line search) *Let f be a scalar-valued continuously differentiable function defined on $\Omega = \{ x \in \mathbb{R}^n : \ell \le x \le u \}$. Assume that $x_0 \in \Omega$ is chosen such that the level set $\mathcal{L}(f(x_0))$ is closed and bounded, and $\{ x_k \}$ is given by $x_{k+1} = x_k(\alpha_k)$, where α_k is a quasi-Wolfe step. Also assume that $\{ p_k \}$ is a sequence of feasible descent directions with $\|p_k\| \le \theta$ for some constant θ independent of k. For an arbitraily fixed $\epsilon > 0$, define*

$$\epsilon_k = \min\{\epsilon, \|P_{x_k}\big(-\nabla f(x_k)\big)\|\}.$$

If the components of p_k satisfy $[\,p_k\,]_i \ge 0$ if $[\,x_k\,]_i < \ell_i + \epsilon_k$, and $[\,p_k\,]_i \le 0$ if $[\,x_k\,]_i > u_i - \epsilon_k$, then either

$$\lim_{k\to\infty} |\nabla f(x_k)^T p_k| = 0, \quad \text{or} \quad \liminf_{k\to\infty} \|P_{x_k}\big(-\nabla f(x_k)\big)\| = 0.$$

Proof. If $\liminf_{k\to\infty} \|P_{x_k}\big(-\nabla f(x_k)\big)\| \ne 0$, then $\liminf_{k\to\infty} \epsilon_k > 0$. We show that

$$\lim_{k\to\infty} |\nabla f(x_k)^T p_k| = 0.$$

The first quasi-Wolfe condition is equivalent to the quasi-Armijo condition, and the arguments in the proof of Theorem 5.1 may be used to show that $\{ f(x_k) \}$ is a convergent sequence. This implies that

$$\lim_{k\to\infty} \alpha_k \nabla f(x_k)^T p_k = 0.$$

The proof is by contradiction. Suppose that $|\nabla f(x_k)^T p_k| \not\to 0$ as $k \to \infty$, then there exists some $\bar{\epsilon} > 0$ such that $|\nabla f(x_k)^T p_k| > \bar{\epsilon}$ infinitely often. Let $\mathcal{G} = \{ k : |\nabla f(x_k)^T p_k| > \bar{\epsilon} \}$, then it must be that $\alpha_k \to 0$ for $k \in \mathcal{G}$. As $\{ \|p_k\| \}$ is uniformly bounded above by θ, $\alpha_k p_k \to 0$ for $k \in \mathcal{G}$.

If the quasi-Wolfe condition $(\mathbf{C}_1)$ is satisfied, then

$$\nabla f\big(x_k(\alpha_k)\big)^T P_{x_k(\alpha_k)}(p_k) \ge -\eta_W |\nabla f(x_k)^T p_k|.$$

Also, if the quasi-Wolfe condition $(\mathbf{C}_3)$ is satisfied, then

$$\nabla f\big(x_k(\alpha_k)\big)^T P_{x_k(\alpha_k)}(p_k) \ge 0 \ge -\eta_W |\nabla f(x_k)^T p_k|.$$

In either case, as $\nabla f(x_k)^T p_k < 0$, it must hold that

$$\nabla f\big(x_k(\alpha_k)\big)^T P_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T p_k \ge (1-\eta_W)|\nabla f(x_k)^T p_k| > (1-\eta_W)\bar{\epsilon}, \quad \text{for } k \in \mathcal{G}.$$

The application of the triangle inequality yields

$$
\begin{aligned}
0 < (1-\eta_W)\bar{\epsilon} < |\nabla f\big(x_k(\alpha_k)\big)^T P_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T p_k| \\
\le |\nabla f\big(x_k(\alpha_k)\big)^T P_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T P_{x_k(\alpha_k)}(p_k)| \\
+ |\nabla f(x_k)^T P_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T p_k|. \qquad (5.4)
\end{aligned}
$$

Let $\|\cdot\|_D$ denote the norm dual to $\|\cdot\|$, then

$$
\begin{aligned}
&|\nabla f\big(x_k(\alpha_k)\big)^T P_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T P_{x_k(\alpha_k)}(p_k)| \\
&\quad \le \|\nabla f\big(x_k(\alpha_k)\big) - \nabla f(x_k)\|_D \|P_{x_k(\alpha_k)}(p_k)\| \le \|\nabla f\big(x_k(\alpha_k)\big) - \nabla f(x_k)\|_D \|p_k\|.
\end{aligned}
$$

As ∇f is continuous and $\|p_k\|$ is uniformly bounded, the right-hand side of this inequality must converge to zero for $k \in \mathcal{G}$, which implies that

$$
|\nabla f\big(x_k(\alpha_k)\big)^T P_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T P_{x_k(\alpha_k)}(p_k)| \to 0, \ \text{ for } k \in \mathcal{G}.
$$

Basic norm inequalities imply

$$
\begin{aligned}
|\nabla f(x_k)^T P_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T p_k| &\le \|\nabla f(x_k)\|_D \|P_{x_k(\alpha_k)}(p_k) - p_k\| \\
&= \|\nabla f(x_k)\|_D \|P_{x_k(\alpha_k)}(p_k) - P_{x_k}(p_k)\|.
\end{aligned}
$$

As the level set $\mathcal{L}(f(x_0))$ is closed and bounded, and ∇f is continuous, the sequence of dual norms $\{\, \|\nabla f(x_k)\|_D \,\}$ is uniformly bounded. Also, because

$$
\|x_k(\alpha_k) - x_k\| \le \|\alpha_k p_k\| \to 0, \ \text{ for } k \in \mathcal{G},
$$

and $\liminf_{k\to\infty} \epsilon_k > 0$, there must exist an $\bar{k}$ such that for all $k \ge \bar{k}$ in $\mathcal{G}$,

$$
[\, x_k(\alpha_k) - x_k \,]_i < \epsilon_k.
$$

From the assumptions on the components of p_k, it must hold that for all $k \ge \bar{k}$ in $\mathcal{G}$, $[\, p_k \,]_i < 0$ only if $[\, x_k \,]_i > \ell_i + \epsilon_k$, in which case $[\, x_k(\alpha_k) \,]_i > \ell_i$; and $[\, p_k \,]_i > 0$ only if $[\, x_k \,]_i < u_i - \epsilon_k$, in which case $[\, x_k(\alpha_k) \,]_i < u_i$. It follows that, for $k \in \mathcal{G}$ sufficiently large,

$$
P_{x_k(\alpha_k)}(p_k) = P_{x_k}(p_k) = p_k.
$$

Therefore,

$$
\|\nabla f(x_k)\|_D \|P_{x_k(\alpha_k)}(p_k) - P_{x_k}(p_k)\| \to 0, \ \text{ for } k \in \mathcal{G},
$$

and consequently

$$
|\nabla f(x_k)^T P_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T p_k| \to 0, \ \text{ for } k \in \mathcal{G}.
$$

It follows that the right-hand side of (5.4) converges to zero for $k \in \mathcal{G}$, which gives the required contradiction.

It remains to consider the case where the quasi-Wolfe condition ($\mathbf{C}_2$) is satisfied, i.e.,

$$
\nabla f\big(x_k(\alpha_k)\big)^T P^-_{x_k(\alpha_k)}(p_k) \ge -\eta_W |\nabla f(x_k)^T p_k|.
$$

The assumption that $\nabla f(x_k)^T p_k < 0$ gives

$$\nabla f\big(x_k(\alpha_k)\big)^T P^-_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T p_k \ge (1-\eta_W)|\nabla f(x_k)^T p_k| > (1-\eta_W)\bar{\epsilon}, \;\; \text{for } k \in \mathcal{G},$$

which implies that

$$\begin{aligned} 0 < (1-\eta_W)\bar{\epsilon} &< |\nabla f\big(x_k(\alpha_k)\big)^T P^-_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T p_k| \\ &\le |\nabla f\big(x_k(\alpha_k)\big)^T P^-_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T P^-_{x_k(\alpha_k)}(p_k)| \\ &\quad + |\nabla f(x_k)^T P^-_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T p_k|. \end{aligned} \tag{5.5}$$

By the definition of the dual norm,

$$\begin{aligned} &|\nabla f\big(x_k(\alpha_k)\big)^T P^-_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T P^-_{x_k(\alpha_k)}(p_k)| \\ &\quad \le \|\nabla f\big(x_k(\alpha_k)\big) - \nabla f(x_k)\|_D \|P^-_{x_k(\alpha_k)}(p_k)\| \le \|\nabla f\big(x_k(\alpha_k)\big) - \nabla f(x_k)\|_D \|p_k\|. \end{aligned}$$

From the continuity of ∇f and uniform boundedness of $\|p_k\|$, the right-hand side of the above inequality converges to zero for $k \in \mathcal{G}$, which means that

$$|\nabla f\big(x_k(\alpha_k)\big)^T P^-_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T P^-_{x_k(\alpha_k)}(p_k)| \to 0, \;\; \text{for } k \in \mathcal{G}.$$

Also,

$$\begin{aligned} |\nabla f(x_k)^T P^-_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T p_k| &\le \|\nabla f(x_k)\|_D \|P^-_{x_k(\alpha_k)}(p_k) - p_k\| \\ &= \|\nabla f(x_k)\|_D \|P^-_{x_k(\alpha_k)}(p_k) - P_{x_k}(p_k)\|. \end{aligned}$$

As the level set $\mathcal{L}(f(x_0))$ is closed and bounded, and ∇f is continuous, it must hold that the sequence of dual norms $\big\{ \|\nabla f(x_k)\|_D \big\}$ is uniformly bounded. Also, as

$$\|x_k(\alpha_k) - x_k\| \le \|\alpha_k p_k\| \to 0, \;\; \text{for } k \in \mathcal{G},$$

arguments analogous to those used to establish convergence in cases $(\mathbf{C}_1)$ and $(\mathbf{C}_3)$ give

$$P^-_{x_k(\alpha_k)}(p_k) = P_{x_k}(p_k) = p_k \;\; \text{for } k \in \mathcal{G} \text{ sufficiently large},$$

in which case

$$\|\nabla f(x_k)\|_D \|P^-_{x_k(\alpha_k)}(p_k) - P_{x_k}(p_k)\| \to 0, \;\; \text{for } k \in \mathcal{G}.$$

This implies that

$$|\nabla f(x_k)^T P^-_{x_k(\alpha_k)}(p_k) - \nabla f(x_k)^T p_k| \to 0, \;\; \text{for } k \in \mathcal{G}.$$

It follows that the right-hand side of (5.5) converges to zero for $k \in \mathcal{G}$, which gives the required contradiction. ∎

6. Numerical Experiments

In this section we illustrate the numerical performance of the projected-search methods described in Sections 4.1 and 4.2. All testing was done on problems taken from the CUTEst test collection (see Bongartz et al. [5] and Gould, Orban and Toint [22]). As of January 1, 2020, the CUTEst test set contains 147 bound-constrained problems.

6.1. Numerical results for the limited-memory quasi-Newton method

In order to assess the effectiveness of the quasi-Wolfe line search, the limited-memory quasi-Newton method described in Section 4.1 was used in conjunction with a quasi-Wolfe and a quasi-Armijo line search. The resulting implementations, `LRHB-qWolfe`, and `LRHB-Armijo` are based on the Fortran package `LRHB` (see Ferry et al. [12]). The kink steps are sorted in decreasing order in $O(n \log n)$ operations using a heapsort algorithm (see, e.g., Williams [35], Knuth [26, Section 5.2.3]). In all the runs, the Fortran version of the heapsort given by Byrd et al. [7] was used. In the implementation of `LRHB-qWolfe`, the Armijo tolerance η_A was chosen to be 10^{-4}, the Wolfe tolerance $\eta_W = 0.9$. In the implementation of `LRHB-Armijo`, $\eta_A = 0.3$. The value of ϵ_k for the definition of the working set used the machine precision for ϵ.

The results are given in Table 2. The top row describes the algorithms being tested, and the first three columns of Table 2 list the properties of all 147 problems in the set. The second column lists the dimension of the problem. The third column lists the number of indices in the active set at the solution. If the active sets differed for the methods tested, the average was taken. If only one method converged, the entry gives the number of active-set indices associated with the successful method. An algorithm was considered to have successfully solved a problem if $\|P_{x_k}\big(-\nabla f(x_k)\big)\|_\infty < 10^{-5}$. If an algorithm reached iteration 5000 without meeting this condition, it was terminated and was considered to have failed to converge. If a method did not converge on a particular problem, the symbol "`--`" is displayed.

A problem name (located in the left-most column) is marked with an asterisk if the two algorithms converged, but converged with different objective values. Objective values f_1 and f_2 are classified as being different if $|f_1 - f_2| > 10^{-4}$. As shown in Table 2, 10 problems `EXPLIN`, `EXPQUAD`, `GRIDGENA`, `HIMMELP1`, `HS2`, `NCVXBQP3`, `PALMER7A`, `PALMER8A`, `POWELLBC`, and `QRTQUAD` converged to different solutions. Problems for which the two algorithms converged to different solutions are not shown in the performance profiles (see below) unless otherwise noted.

Table 2 shows that the solvers converged to the same solution for 106 problems. Summing over all these problems, gives the overall comparison of Table 1. It should be noted that the run-time totals in Table 1 mainly reflect the overhead associated with cost of an iteration—with the exception of `BLEACHNG`, none of the problems tested have computationally expensive functions.

Table 1: Comparison of `LRHB-qWolfe` and `LRHB-Armijo` (Sum Total)

Algorithm	Nf	Time (sec)	Failed
LRHB-QWOLFE	36407	70.5340	28
LRHB-ARMIJO	51667	63.6327	31

Table 2: Comparison of algorithms on 147 problems.

			LRHB-qWolfe		LRHB-Armijo	
Problem	n	$\mathcal{A}(x^*)$	Nf	Time	Nf	Time
3PK	30	0	--	--	--	--
AIRCRFTB	8	3	63	0.0020	72	0.0020
ALLINIT	4	1	22	0.0010	22	0.0010
BDEXP	5000	0	20	0.0540	20	0.0550
BIGGS3	6	3	24	0.0010	29	0.0010
BIGGS5	6	1	90	0.0030	150	0.0050
BIGGSB1	5000	2	--	--	--	--
BLEACHNG	17	11	8	6.6290	--	--
BOX2	3	1	19	0.0001	19	0.0001
BQP1VAR	1	1	3	0.0001	8	0.0010
BQPGABIM	50	14	30	0.0020	32	0.0010
BQPGASIM	50	10	30	0.0010	35	0.0020
BQPGAUSS	2003	90	--	--	--	--
BRATU1D	5003	2	--	--	--	--
CAMEL6	2	0	16	0.0010	14	0.0010
CHARDIS0	2000	0	5	0.2560	5	0.3050
CHEBYQAD	100	0	1183	2.3950	1134	2.2890
CHENHARK	5000	2000	--	--	--	--
CLPLATEA	5041	71	1380	1.6810	1312	1.6110
CLPLATEB	5041	71	663	0.8270	490	0.6150
CLPLATEC	5041	71	--	--	--	--
CVXBQP1	10000	10000	11	0.0880	62	0.1350
DECONVB	63	17	131	0.0090	191	0.0110
DECONVU	63	12	1260	0.0730	470	0.0260
DEGDIAG	100001	100001	3	0.6990	3	0.7090
DEGTRID	100001	0	45	1.3650	42	1.2660
DEGTRID2	100001	100001	3	0.7060	9	0.8140
DRCAV1LQ	4489	520	--	--	--	--
DRCAV2LQ	4489	520	--	--	--	--
DRCAV3LQ	4489	520	--	--	--	--
EG1	3	1	21	0.0010	16	0.0001
EXPLIN*	1200	1149	219	0.0350	403	0.0530
EXPLIN2	1200	1181	148	0.0280	182	0.0300
EXPQUAD*	1200	77	56	0.0180	125	0.0250
FBRAIN2LS	4	1	37	0.0830	68	0.1500
FBRAINLS	2	0	16	0.0190	17	0.0200
GENROSEB	500	498	160	0.0240	--	--
GRIDGENA*	6218	658	101	0.1620	331	0.4280
HADAMALS	400	39	32	0.0090	24	0.0090
HARKERP2	1000	999	53	0.1480	50	0.1390

Table 2: Comparison of algorithms on 147 problems. (Continued)

			LRHB-qWolfe		LRHB-Armijo	
Problem	n	$\mathcal{A}(x^*)$	Nf	Time	Nf	Time
HART6	6	0	26	0.0010	26	0.0001
HATFLDA	4	1	88	0.0020	--	--
HATFLDB	4	2	82	0.0020	--	--
HATFLDC	25	0	30	0.0010	39	0.0020
HIMMELP1*	2	1	15	0.0010	32	0.0010
HOLMES	180	176	72	0.1340	99	0.1820
HS1	2	0	25	0.0010	114	0.0020
HS2*	2	0	12	0.0001	16	0.0010
HS3	2	1	22	0.0001	18	0.0010
HS3MOD	2	1	2	0.0001	2	0.0001
HS4	2	1	10	0.0001	11	0.0001
HS5	2	0	48	0.0020	229	0.0040
HS25	3	1	11	0.0001	27	0.0010
HS38	4	2	3	0.0001	6	0.0010
HS45	5	5	5	0.0010	24	0.0010
HS110	10	0	10	0.0010	14	0.0010
JNLBRNG1	10000	3507	509	1.4950	482	1.4230
JNLBRNG2	10000	4246	632	1.8020	684	1.9170
JNLBRNGA	10000	3641	366	0.9920	384	1.0360
JNLBRNGB	10000	4617	2231	5.5820	2408	5.9520
KOEBHELB	3	0	207	0.0110	453	0.0190
LINVERSE	1999	569	482	0.2980	1146	0.6300
LMINSURF	5625	296	547	0.7790	552	0.7740
LOGROS	2	0	105	0.0030	352	0.0060
MAXLIKA	8	1	849	0.1270	812	0.1200
MCCORMCK	5000	1	26	0.0640	30	0.0700
MDHOLE	2	1	87	0.0020	200	0.0030
MINSURF	64	28	21	0.0010	45	0.0020
MINSURFO	5306	1762	484	0.9630	506	1.0240
NCVXBQP1	10000	4003	5	0.0800	5	0.0810
NCVXBQP2	10000	5606	5	0.0820	5	0.0810
NCVXBQP3*	10000	9861	121	0.2310	183	0.2810
NLMSURF	5625	296	4682	6.3490	4264	5.6660
NOBNDTOR	5476	1158	176	0.2970	192	0.3140
NONSCOMP	5000	2491	41	0.0840	48	0.0780
OBSTCLAE	10000	4869	195	0.5940	217	0.6500
OBSTCLAL	10000	4869	145	0.4530	144	0.4490
OBSTCLBL	10000	2943	143	0.4470	145	0.4570
OBSTCLBM	10000	2943	157	0.4840	150	0.4600
OBSTCLBU	10000	2943	135	0.4270	153	0.4740
ODC	5184	284	238	0.5760	220	0.5290
ODNAMUR	11130	4916	--	--	--	--
OSLBQP	8	7	3	0.0010	3	0.0010
PALMER1	4	0	84	0.0020	68	0.0010
PALMER1A	6	0	830	0.0250	1517	0.0530
PALMER1B	4	0	98	0.0020	71	0.0020
PALMER1E	8	0	--	--	--	--
PALMER2	4	1	--	--	--	--
PALMER2A	6	0	325	0.0100	667	0.0310
PALMER2B	4	0	74	0.0020	58	0.0010

Table 2: Comparison of algorithms on 147 problems.
(Continued)

			LRHB-qWolfe		LRHB-Armijo	
Problem	n	$\mathcal{A}(x^*)$	Nf	Time	Nf	Time
PALMER2E	8	0	--	--	--	--
PALMER3	4	1	29481	0.2540	--	--
PALMER3A	6	0	578	0.0200	795	0.0230
PALMER3B	4	0	122	0.0020	34	0.0010
PALMER3E	8	0	--	--	--	--
PALMER4	4	0	48	0.0010	85	0.0010
PALMER4A	6	0	385	0.0130	913	0.0400
PALMER4B	4	0	90	0.0020	41	0.0010
PALMER4E	8	0	--	--	--	--
PALMER5A	8	1	--	--	--	--
PALMER5B	9	0	--	--	--	--
PALMER5E	8	0	--	--	3051	0.0820
PALMER6A	6	0	1198	0.0350	1052	0.0280
PALMER6E	8	0	--	--	--	--
PALMER7A*	6	0	5505	0.1620	6725	0.1660
PALMER7E	8	0	--	--	5564	0.1640
PALMER8A*	6	0	380	0.0100	627	0.0170
PALMER8E	8	0	--	--	--	--
PENTDI	5000	4998	5	0.0360	5	0.0370
PFIT1LS	3	0	587	0.0140	1653	0.0280
PFIT2LS	3	0	2052	0.0480	6366	0.1060
PFIT3LS	3	0	2291	0.0520	8969	0.1510
PFIT4LS	3	0	3285	0.0770	4688	0.0790
POWELLBC*	1000	205	2076	16.8130	5152	41.2730
PROBPENL	500	0	5	0.0040	5	0.0040
PSPDOC	4	1	13	0.0001	19	0.0001
QR3DLS	610	0	--	--	--	--
QRTQUAD*	5000	1	13	0.0440	12	0.0450
QUDLIN	5000	4999	12	0.0430	63	0.0580
RAYBENDL	2050	4	--	--	--	--
RAYBENDS	2050	4	4054	36.2590	3218	28.7400
S368	8	2	13	0.0010	16	0.0010
SANTALS	21	0	349	0.0150	391	0.0150
SCOND1LS	5002	2	--	--	--	--
SIM2BQP	2	2	3	0.0001	5	0.0001
SIMBQP	2	1	6	0.0001	10	0.0010
SINEALI	1000	0	52	0.0190	52	0.0190
SPECAN	9	0	154	0.3270	262	0.5560
SSC	5184	284	231	0.5380	206	0.4770
TORSION1	5476	1916	165	0.2730	190	0.3060
TORSION2	5476	1916	198	0.3300	197	0.3240
TORSION3	5476	3676	92	0.1680	83	0.1520
TORSION4	5476	3676	110	0.1990	124	0.2150
TORSION5	5476	4524	43	0.0960	49	0.1020
TORSION6	5476	4524	62	0.1270	66	0.1300
TORSIONA	5476	1852	141	0.2630	182	0.3180
TORSIONB	5476	1852	208	0.3730	200	0.3590
TORSIONC	5476	3644	77	0.1570	96	0.1840
TORSIOND	5476	3644	92	0.1860	118	0.2290
TORSIONE	5476	4508	38	0.0960	47	0.1090

Table 2: Comparison of algorithms on 147 problems. (Continued)

			LRHB-qWolfe		LRHB-Armijo	
Problem	n	$\mathcal{A}(x^*)$	Nf	Time	Nf	Time
TORSIONF	5476	4508	58	0.1290	61	0.1330
WALL10	1461	294	--	--	--	--
WALL20	5924	1238	--	--	--	--
WALL50	37311	7820	--	--	--	--
WALL100	149624	38895	--	--	--	--
WEEDS	3	0	61	0.0010	81	0.0020
YFIT	3	0	85	0.0020	261	0.0060

The results of Table 2 are summarized using performance profiles (in $\log_2$ scale) proposed by Dolan and Moré [10]. If $\mathcal{P}$ denotes a set of n_p problems used for a given numerical experiment. For each method s we define the function $\pi_s : [0, r_M] \mapsto \mathbb{R}^+$ such that

$$\pi_s(\tau) = \frac{1}{n_p} \left|\left\{\, p \in \mathcal{P} : \log_2(r_{p,s}) \le \tau \,\right\}\right|,$$

where n_p is the number of problems in the test set and $r_{p,s}$ denotes the ratio of the number of function evaluations needed to solve problem p with method s and the least number of function evaluations needed to solve problem p. The number r_M is the maximum value of $\log_2(r_{p,s})$. Similarly, the profile can be defined with $r_{p,s}$ being the ratio of the time needed to solve problem p with method s and the least time needed to solve problem p.

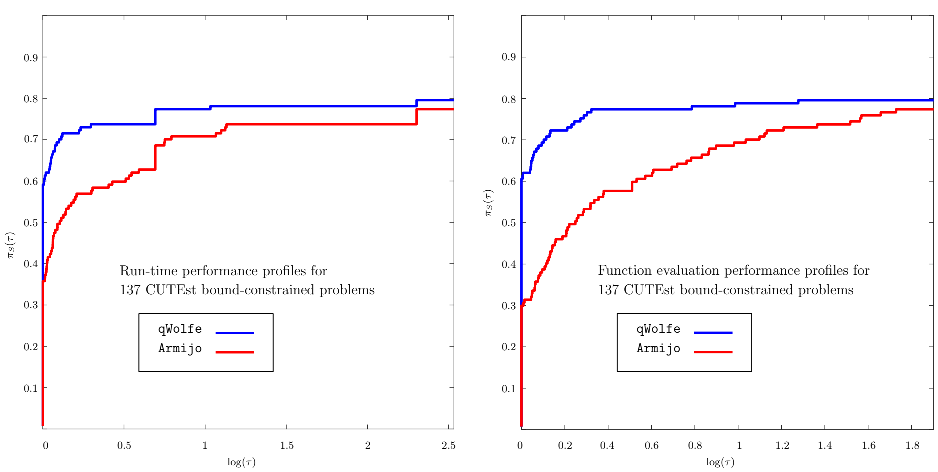

Figure 4: The figures give the performance profiles on the 137 bound-constrained problems from the CUTEst test set for which the quasi-Wolfe and Armijo methods terminated at the same solution. The profile for the total run-time is given on the left. The profile for the total number of function evaluations is given on the right.

Figure 4 gives the performance profiles for the 137 problems for which the quasi-Wolfe and Armijo methods terminated at the same solution. The profile for the total run-time is given on the left. The profile for the total number of function evaluations is given on the right. Tables 1–2 and Figure 4 indicate that using a quasi-Wolfe line search in LRHB resulted in better performance with respect to time and function calls than using a quasi-Armijo line search.

6.2. Numerical results for the projected-search interior method

Numerical results are given for a MATLAB implementation of the interior-point method described in Section 4.2. All testing was done using MATLAB version R2019a on an iMac with a 3.0 GHz Intel Zeon W processor and 128 GB of 800 MHz DDR4 RAM running macOS, version 10.14.6 (64 bit). Results were obtained for a subset of the 147 bound-constrained problems in standard interface format (SIF). The test set includes all bound-constrained problems for which n is of the order of 600 or less, and problems for which the dimension may be set at less than 1000. This gave 137 problems ranging in size from `BQP1VAR` (one variable) to `POWELLBC` (1000 variables). Exact second derivatives were used for all the runs.

Results are presented from two interior methods based on the primal-dual interior method of Forsgren and Gill [14]. The first, `PD-Wolfe`, is the conventional primal-dual method implemented with a Wolfe line search. The second method, `PDproj-qWolfe`, is the method proposed in Section 4.2. As underlying interior method is the same in both cases, the results show the benefits of formulating the method as a projected search method.

The algorithms were considered to have solved a problem successfully if

$$\max\left\{\|\max(0, g(x)\ \boldsymbol{\cdot}\ x_\ell)\|_\infty, \|\max(0, -g(x)\ \boldsymbol{\cdot}\ x_u)\|_\infty\right\} \le 10^{-5},$$

where $x_\ell = \min\{1, (x-\ell)\ \boldsymbol{\cdot}/\ (1+|\ell|)\}$, $x_u = \min\{1, (u-x)\ \boldsymbol{\cdot}/\ (1+|u|)\}$, and $g(x) = \nabla f(x)/(\max\{1, \|\nabla f(x)\|_\infty\})$. A limit of 500 was placed on the number of iterations. The strategy for choosing the barrier parameter μ was that used in the method of Gertz and Gill [16]. If necessary, the block-diagonal matrix associated with the symmetric indefinite factorization of H_k (4.2) was modified to give a positive definite $\widehat{H}_k$ (see Moré and Sorensen [28]). The factors were computed using the MATLAB routine `LDL`. The fraction-to-the-boundary parameter σ of (4.3) was set at 0.9.

The full list of results is given in Table 3. An entry "`--`" indicate that the convergence criterion was not satisfied in 500 iterations. An "i" implies that the method was terminated because the line search was unable to find a sufficiently improved point. (In all cases, this occurred on iterations at which the Hessian H_k (4.2) was modified.)

Table 3: Interior-point methods on 137 problems.

		PD-Wolfe		PD-qWolfe	
Problem	n	Itns	Nf	Itns	Nf
3PK	30	10	11	10	7
AIRCRFTB	8	10	15	10	15
ALLINIT	4	10	14	10	14
BDEXP	500	13	14	13	12

Table 3: Comparison of interior methods on 137 problems. (Continued)

		PD-Wolfe		PD-qWolfe	
Problem	n	Itns	Nf	Itns	Nf
BIGGS3	6	8	14	8	14
BIGGS5	6	47	71	47	71
BIGGSB1	25	15	16	15	11
BLEACHNG	17	20	22	20	6
BOX2	3	7	11	7	11
BQP1VAR	1	11	14	11	11
BQPGABIM	50	16	17	16	21
BQPGASIM	50	16	17	16	21
BRATU1D	503	4	5	4	5
CAMEL6	2	8	10	8	9
CHARDIS0	400	10	11	10	4
CHEBYQAD	100	51	58	51	54
CLPLATEA	529	5	6	5	6
CLPLATEB	529	2	3	2	3
CLPLATEC	529	1	2	1	2
CVXBQP1	100	10	11	10	9
DECONVB	63	27	29	27	27
DRCAV1LQ	196	31	38	31	38
DRCAV2LQ	196	20	30	20	30
DRCAV3LQ	196	21	35	21	35
EG1	3	16	21	16	18
EXPLIN	120	64	66	64	16
EXPLIN2	120	68	70	68	16
EXPQUAD	120	20	21	20	16
FBRAIN2LS	4	16	20	16	22
FBRAINLS	2	10	11	10	10
GENROSEB	500	140	143	140	19
GRIDGENA	500	6	7	6	5
HADAMALS	400	--	--	129	129
HARKERP2	500	29	30	29	15
HART6	6	8	10	8	9
HATFLDA	4	13	20	13	7
HATFLDB	4	11	12	11	13
HATFLDC	25	7	8	7	6
HIMMELP1	2	12	16	12	14
HOLMES	180	20	24	20	19
HS1	2	27	33	27	36
HS2	2	15	20	15	22
HS3	2	12	21	12	7
HS3MOD	2	13	21	13	27
HS4	2	15	23	15	13
HS5	2	8	9	8	9
HS25	3	0	1	0	1
HS38	4	42	54	42	62
HS45	5	11	12	11	12
HS110	10	7	8	7	7
JNLBRNG1	529	10	14	10	12
JNLBRNG2	529	9	13	9	11
JNLBRNGA	529	11	15	11	12
JNLBRNGB	529	12	16	12	11

Table 3: Comparison of interior methods on 137 problems. (Continued)

		PD-Wolfe		PD-qWolfe	
Problem	n	Itns	Nf	Itns	Nf
KOEBHELB	3	80	115	80	116
LINVERSE	199	35	40	35	27
LMINSURF	121	8	12	8	12
LOGROS	2	35	53	35	51
MAXLIKA	8	49	54	49	42
MCCORMCK	500	11	12	11	7
MDHOLE	2	43	65	43	84
MINSURF	64	4	8	4	8
MINSURFO	731	16	20	16	19
NCVXBQP1	100	439	442	439	17
NCVXBQP2	100	426	431	426	20
NCVXBQP3	100	284	292	284	15
NLMSURF	961	9	11	9	11
NOBNDTOR	484	10	11	10	11
NONSCOMP	500	12	13	12	9
OBSTCLAE	529	14	17	14	14
OBSTCLAL	529	15	19	15	13
OBSTCLBL	529	62	63	62	10
OBSTCLBM	529	11	12	11	8
OBSTCLBU	529	82	83	82	10
ODC	144	15	16	15	16
OSLBQP	8	13	16	13	14
PALMER1	4	22	29	22	15
PALMER1A	6	48	59	48	55
PALMER1B	4	21	28	21	22
PALMER1E	8	85	108	85	70
PALMER2	4	15	20	15	13
PALMER2A	6	76	96	76	101
PALMER2B	4	17	20	17	19
PALMER2E	8	63	81	63	82
PALMER3	4	22	29	22	29
PALMER3A	6	69	87	69	111
PALMER3B	4	16	17	16	17
PALMER3E	8	51	62	51	67
PALMER4	4	17	24	17	27
PALMER4A	6	48	58	48	68
PALMER4B	4	17	19	17	14
PALMER4E	8	39	45	39	47
PALMER5A	8	--	--	--	--
PALMER5B	9	437	688	437	791
PALMER5D	4	2	3	2	3
PALMER5E	8	--	--	--	--
PALMER6A	6	114	146	114	161
PALMER6E	8	46	59	46	59
PALMER7A	6	--	--	--	--
PALMER7E	8	i	i	i	i
PALMER8A	6	35	44	35	43
PALMER8E	8	28	30	28	29
PENTDI	500	9	14	9	12
PFIT1LS	3	235	344	235	368

Table 3: Comparison of interior methods on 137 problems. (Continued)

		PD-Wolfe		PD-qWolfe	
Problem	n	Itns	Nf	Itns	Nf
PFIT2LS	3	36	50	36	54
PFIT3LS	3	122	171	122	240
PFIT4LS	3	222	325	222	397
POWELLBC	1000	--	--	--	--
PROBPENL	500	5	6	5	3
PSPDOC	4	10	16	10	15
QR3DLS	610	198	369	198	361
QRTQUAD	120	46	50	46	43
QUDLIN	120	178	179	178	6
RAYBENDL	130	i	i	i	i
RAYBENDS	130	i	i	i	i
S368	100	50	56	50	30
SANTALS	21	37	51	37	51
SCOND1LS	502	377	555	377	419
SIM2BQP	2	12	13	12	13
SIMBQP	2	9	10	9	10
SINEALI	100	11	12	11	9
SPECAN	9	12	13	12	10
SSC	1122	2	3	2	3
TORSION1	484	9	10	9	11
TORSION2	484	9	10	9	9
TORSION3	484	10	11	10	10
TORSION4	484	10	11	10	9
TORSION5	484	11	12	11	9
TORSION6	484	10	11	10	11
TORSIONA	484	9	10	9	11
TORSIONB	484	8	9	8	8
TORSIONC	484	10	11	10	10
TORSIOND	484	9	10	9	9
TORSIONE	484	10	11	10	10
TORSIONF	484	9	10	9	8
WEEDS	3	33	39	33	42
YFIT	3	45	54	45	63

Figure 5 gives the performance profiles for the total number of function evaluations required to solve the 137 problems. The profiles compare the primal-dual interior method `PD-Wolfe` implemented with a Wolfe line search and a projected-search interior method `PDproj-qWolfe` with a quasi-Wolfe line search (i.e., the method described in Section 4.2). Figure 5 and the results of Table 3 indicate that a projected search interior method with a quasi-Wolfe line search can provide substantial improvements in robustness and performance compared to a conventional interior method.

7. Conclusions

Projected-search methods for bound-constrained minimization are based on performing a line search along a continuous piecewise-linear path obtained by projecting

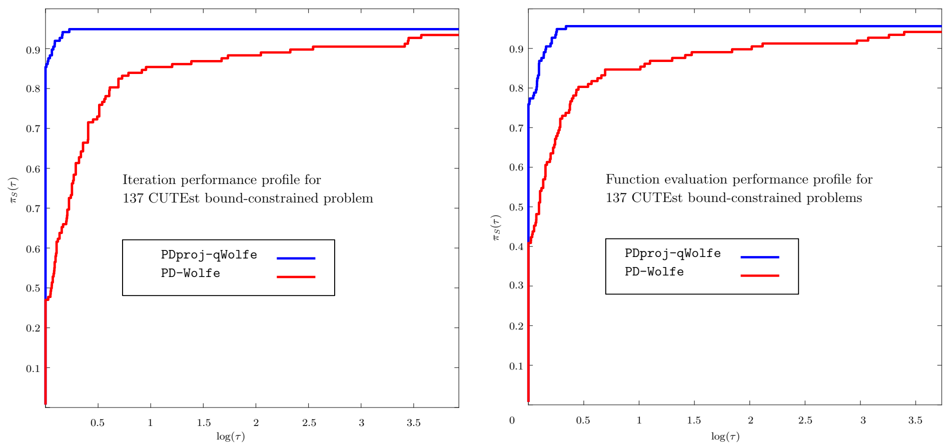

Figure 5: Performance profiles for two interior-point methods: a conventional primal-dual method with a Wolfe line search (`PD-Wolfe`) and a projected-search interior method with a quasi-Wolfe line search (`PDproj-qWolfe`). The figure gives the performance profiles for the total number of iterations and function evaluations required to solve 137 bound-constrained problems from the CUTEst test set.

a search direction onto the feasible region. A potential benefit of a projected-search method is that far from the solution, an estimate of the active set can be changed significantly at the cost of computing a single search direction. Two projected-search methods are considered: a limited-memory quasi-Newton method, and a new method that combines a primal-dual interior method with a projected search. Both methods utilize a new *quasi-Wolfe* line search that is appropriate for a function defined on the piecewise-linear path. The behavior of the line search is similar to that of a conventional Wolfe line search, except that a step is accepted under a wider range of conditions. These conditions take into consideration steps at which the restriction of the objective function on the search path is not differentiable. In a quasi-Newton projected search method, the quasi-Wolfe line search has the benefit that it is more likely that the quasi-Newton update can be applied. Standard existence and convergence results associated with a conventional Wolfe line search are extended to the quasi-Wolfe case.

Numerical results indicate that a projected search method with a quasi-Wolfe line search can require fewer iterations and function evaluations compared to the same method implemented with a conventional Armijo line search. Moreover, a projected search method interior method using a quasi-Wolfe line search can provide substantial improvements in robustness and performance compared to a conventional interior method.

References

[1] L. Armijo. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics*, 16:1–3, 1966. 2

[2] D. P. Bertsekas. On the Goldstein-Levitin-Polyak gradient projection method. *IEEE Trans. Automatic Control*, AC-21(2):174–184, 1976. 2, 3

[3] D. P. Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Computer Science and Applied Mathematics. Academic Press Inc. [Harcourt Brace Jovanovich Publishers], New York, 1982. 3, 17

[4] D. P. Bertsekas. Projected Newton methods for optimization problems with simple constraints. *SIAM J. Control Optim.*, 20(2):221–246, 1982. 2

[5] I. Bongartz, A. R. Conn, N. I. M. Gould, and Ph. L. Toint. CUTE: Constrained and unconstrained testing environment. *ACM Trans. Math. Software*, 21(1):123–160, 1995. 24

[6] R. P. Brent. *Algorithms for minimization without derivatives*. Prentice-Hall Inc., Englewood Cliffs, N.J., 1973. Prentice-Hall Series in Automatic Computation. 7

[7] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.*, 16:1190–1208, 1995. 24

[8] P. H. Calamai and J. J. Moré. Projected gradient methods for linearly constrained problems. *Math. Program.*, 39:93–116, 1987. 2

[9] S. H. Cheng and N. J. Higham. A modified Cholesky algorithm based on a symmetric indefinite factorization. *SIAM J. Matrix Anal. Appl.*, 19(4):1097–1110, 1998. 19

[10] E. D. Dolan and J. J. Moré. Benchmarking optimization software with COPS. Technical Memorandum ANL/MCS-TM-246, Argonne National Laboratory, Argonne, IL, 2000. 28

[11] M. W. Ferry. *Projected-Search Methods for Box-Constrained Optimization*. PhD thesis, Department of Mathematics, University of California, San Diego, May 2011. 2, 17

[12] M. W. Ferry, P. E. Gill, E. Wong, and M. Zhang. A limited-memory reduced-Hessian method for bound-constrained optimization. Center for Computational Mathematics Report CCoM 20-02, Center for Computational Mathematics, University of California, San Diego, La Jolla, CA, 2020. 2, 17, 24

[13] R. Fletcher and T. L. Freeman. A modified Newton method for minimization. *J. Optim. Theory Appl.*, 23:357–372, 1977. 19

[14] A. Forsgren and P. E. Gill. Primal-dual interior methods for nonconvex nonlinear programming. *SIAM J. Optim.*, 8:1132–1152, 1998. 18, 29

[15] A. Forsgren, P. E. Gill, and W. Murray. Computing modified Newton directions using a partial Cholesky factorization. *SIAM J. Sci. Comput.*, 16:139–150, 1995. 19

[16] E. M. Gertz and P. E. Gill. A primal-dual trust-region algorithm for nonlinear programming. *Math. Program., Ser. B*, 100:49–94, 2004. 29

[17] N. Ghosh and W. W. Hager. A derivative-free bracketing scheme for univariate minimization. *Computers Math. Applic.*, 20(2):23–24, 1990. 7

[18] P. E. Gill and M. W. Leonard. Limited-memory reduced-Hessian methods for large-scale unconstrained optimization. *SIAM J. Optim.*, 14:380–401, 2003. 17

[19] P. E. Gill and W. Murray. Newton-type methods for unconstrained and linearly constrained optimization. *Math. Program.*, 7:311–350, 1974. 19

[20] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. A note on a sufficient-decrease criterion for a nonderivative step-length procedure. *Math. Programming*, 23(3):349–352, 1982. 2

[21] A. A. Goldstein. Convex programming in Hilbert space. *Bulletin of the American Mathematical Society*, 70(5):709–710, 1964. 2

[22] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr and SifDec: A constrained and unconstrained testing environment, revisited. *ACM Trans. Math. Software*, 29(4):373–394, 2003. 24

[23] J. Greenstadt. On the relative efficiencies of gradient methods. *Math. Comput.*, 21:360–367, 1967. 19

[24] W. W. Hager. A derivative-based bracketing scheme for univariate minimization and the conjugate gradient method. *Computers Math. Applic.*, 18(9):779–795, 1989. 7

[25] D. Kim, S. Sra, and I. S. Dhillon. Tackling box-constrained optimization via a new projected quasi-Newton approach. *SIAM J. Sci. Comput.*, 32(6):3548–3563, December 2010. 2

[26] D. Knuth. *The Art of Computer Programming, 3.* Addison-Wesley Publishing Company, Redwood City, third edition, 1997. 16, 24

[27] E. S. Levitin and B. T. Polyak. Constrained minimization methods. *U.S.S.R. Comput. Math. and Math. Physics*, 6(5):1–50, 1966. 2

[28] J. J. Moré and D. C. Sorensen. On the use of directions of negative curvature in a modified Newton method. *Math. Program.*, 16:1–20, 1979. 19, 29

[29] J. J. Moré and D. C. Sorensen. Newton's method. In G. H. Golub, editor, *Studies in Mathematics, Volume 24. MAA Studies in Numerical Analysis*, pages 29–82. Math. Assoc. America, Washington, DC, 1984. 4

[30] J. J. Moré and D. J. Thuente. Line search algorithms with guaranteed sufficient decrease. *ACM Trans. Math. Software*, 20(3):286–307, 1994. 2, 4, 7

[31] Q. Ni and Y.-x. Yuan. A subspace limited memory quasi-Newton algorithm for large-scale nonlinear bound constrained optimization. *Math. Comput.*, 66:1509–1520, 10 1997. 2

[32] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer-Verlag, New York, 1999. 4, 14

[33] J. M. Ortega and W. C. Rheinboldt. *Iterative solution of nonlinear equations in several variables.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000. Reprint of the 1970 original. 2

[34] R. B. Schnabel and E. Eskow. A new modified Cholesky factorization. *SIAM J. Sci. and Statist. Comput.*, 11:1136–1158, 1990. 19

[35] J. W. J. Williams. Algorithm 232 - Heapsort. *Communications of the Association for Computing Machinery*, 7:347–348, 1964. 16, 24

[36] P. Wolfe. Convergence conditions for ascent methods. *SIAM Rev.*, 11:226–235, 1969. 2

[37] P. Wolfe. On the convergence of gradient methods under constraint. *IBM J. Res. Dev.*, 16:407–411, 1972. 4