# COMBINING SEARCH DIRECTIONS USING GRADIENT FLOWS

**Javier M. Moguerza**
**Francisco J., Prieto**

**00-79**

WORKING PAPERS

Universidad Carlos III de Madrid

Working Paper 00-79
Statistics and Econometrics Series 38
November 2000

Departamento de Estadística y Econometría
Universidad Carlos III de Madrid
Calle Madrid, 126
28903 Getafe (Spain)
Fax (34) 91 624-98-49

# COMBINING SEARCH DIRECTIONS USING GRADIENT FLOWS

Javier M. Moguerza y Francisco J. Prieto*

**Abstract**

The efficient combination of directions is a significant problem in line search methods that either use negative curvature, or wish to include additional information such as the gradient or different approximations to the Newton direction.

In this paper we describe a new procedure to combine several of these directions within an interior-point primal-dual algorithm. Basically, we combine in an efficient manner a modified Newton direction with the gradient of a merit function and a direction of negative curvature, is it exists. We also show that the procedure is well-defined, and it has reasonable theoretical properties regarding the convergence of the method.

We also present numerical results from an implementation of the proposed algorithm on a set of small test problems from the CUTE collection.

**Keywords:** Negative curvature; Primal-dual methods; Interior-point methods; Nonconvex optimization; Line searches.

*Moguerza, School of Engineering, Universidad Rey Juan Carlos, Spain, e-mail: j.moguerza@escet.urcj.es; Prieto, Dept. of Statistics and Econometrics; Universidad Carlos III de Madrid; e-mail: fjp@est-econ.uc3m.es

# Combining search directions using gradient flows

Javier M. Moguerza[1]
School of Engineering
Univ. Rey Juan Carlos, Spain
E-mail: j.moguerza@escet.urjc.es

Francisco J. Prieto[1]
Dept. of Statistics and Econometrics
Univ. Carlos III de Madrid, Spain
E-mail: fjp@est-econ.uc3m.es

ABSTRACT

The efficient combination of directions is a significant problem in line search methods that either use negative curvature, or wish to include additional information such as the gradient or different approximations to the Newton direction.
In this paper we describe a new procedure to combine several of these directions within an interior-point primal-dual algorithm. Basically, we combine in an efficient manner a modified Newton direction with the gradient of a merit function and a direction of negative curvature, if it exists. We also show that the procedure is well-defined, and it has reasonable theoretical properties regarding the convergence of the method.
We also present numerical results from an implementation of the proposed algorithm on a set of small test problems from the CUTE collection.

**Keywords:** Negative curvature; Primal-dual methods; Interior-point methods; Nonconvex optimization; Line searches

**AMS:** 49M37, 65K05, 90C30

## 1 Introduction

Our goal is to describe a procedure to combine search directions in algorithms that compute local solutions for nonlinear optimization problems of the form

$$\begin{array}{ll} \min_x & f(x) \\ \text{s.t.} & c(x) = 0 \\ & x \geq 0, \end{array} \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $c : \mathbb{R}^n \to \mathbb{R}^m$, and we assume all functions to be at least twice continuously differentiable.

Standard procedures for computing these solutions are based on solving local approximations to the problem at successive iterates $x_k$, and using the solutions for these approximations as the next iterates. This basic scheme is complicated by the need to ensure global convergence for the resulting algorithm, attained basically through the use of trust-region schemes or searches along a parametrized line or curve (linesearch methods); we will concentrate on procedures that are based on this second alternative. Standard algorithms in this class (see for example Fletcher [13], Gill et al. [17]) rely on some modified version of the Newton direction and compute the next iterate on the unidimensional subspace generated by this direction. In some cases, it may be useful or efficient to take into account additional information to determine this next iterate. For example, the use of directions of negative curvature is needed to ensure the convergence of the algorithm to second-order KKT points, while perhaps improving the efficiency of the algorithm. The use of additional descent information, obtained from the gradient, for example, may provide more robust algorithms and may yield better iterates, particularly away from the solution. Our goal is to derive a procedure that, by taking into account additional search information in an efficient manner, requires a reduced number of iterations to obtain a solution for problem (1), with a consequently reduced computational cost.

Although these directions might be used in the line searches independently of each other to generate the sequence of iterates, it seems more efficient to combine them before computing the next iterate. Many proposals based on these ideas can be found in the literature. For example, the dogleg method

[1]This research was supported by DGICYT grant PB96-0111.

(see Dennis and Schnabel [9], for example) combines the gradient and Newton direction in an attempt to mimic the behavior of trust-region methods. Moré and Sorensen [21] proposed a procedure to find the next iterate from a direction of descent and one of negative curvature by following a quadratic curve on the subspace spanned by both directions.

In this paper we present a proposal based on the approximate solution of a system of ordinary differential equations. This idea was first proposed in Courant [8]; other related proposals can be found in Behrman [1], Botsaris [4], Schropp [22] and Zang [27] for the unconstrained case, and Evtushenko and Zhadan [11] for the constrained case. Our proposal is most closely related to that in [1] for the unconstrained case, where the iterates were found by constructing a Krylov subspace in each iteration, performing a standard univariate search on the steepest descent curve defined on this subspace. We apply similar ideas to the combination of the search directions in a constrained optimization setting.

The main difficulty when combining the different search directions arises from the differences in the scales of these directions. While the Newton direction is in general well scaled (a step of one is reasonable in many cases), this is not true either for directions of negative curvature or for the gradient direction, our choices of additional search directions. One alternative to overcome this difficulty would be to carry out a search on the reduced-dimension subspace spanned by these search directions. Byrd et al. [6] compute the next iterate from a linear combination of a direction of negative curvature and a gradient direction, and these coefficients are obtained as the solution of a two-dimensional trust-region problem. Nevertheless, most proposals in the literature reduce first the search to a univariate one, to attain greater computational efficiency. This will also be our approach; we will construct a curve in the subspace generated by the directions of interest: descent direction, negative curvature and gradient. A reasonable curve in this subspace would be the one that corresponds to the trajectory having the steepest descent at each point; this trajectory would lead to a local solution at the fastest rate, measured in terms of the objective function. Unfortunately, this curve is in general very expensive to compute, and we will satisfy ourselves with constructing a steepest descent curve based on a simple (quadratic) local model of the problem. The next iterate will be obtained as a point on the curve providing sufficient descent for an appropriate merit function.

This proposal will be introduced as part of a complete algorithm for the solution of problem (1), based on a primal-dual interior point method and the use of an augmented Lagrangian merit function. The method computes approximate solutions for a sequence of barrier problems of the form

$$\begin{array}{ll} \min_x & f(x) - \sum_i (\mu_k)_i \log x_i \\ \text{s.t.} & c(x) = 0, \end{array} \tag{2}$$

where $\mu_k \to 0$, see Fiacco and McCormick [12], Wright [24] for a theoretical analysis of these procedures. Note also that we use a vector of barrier parameters $\mu \in \mathbb{R}^n$.

In each iteration, the algorithm computes a descent direction and a direction of negative curvature for problem (2), if it exists. These directions and the gradient of an augmented Lagrangian merit function (see Bertsekas [2])

$$LA(x, \lambda; \rho) = f(x) - \sum_i \mu_i \log x_i - \lambda^T c(x) + \frac{\rho}{2} \|c(x)\|^2, \tag{3}$$

are then combined to generate a new iterate that provides sufficient decrease for this merit function. This merit function has been extensively used in optimization packages, see for example Conn et al. [7].

The rest of the paper is organized as follows: In Section 2 we describe the general algorithm to compute a local solution for problem (1). In Section 3 we motivate and describe the proposal to combine the directions generated by the algorithm to obtain the next iterate. In Section 4 we justify some basic properties of this procedure, such as for example that it is well-defined, and that sufficient descent can be achieved in each iteration. Finally, Section 5 gives the general structure of the algorithm, discusses some implementation issues and presents and comments some computational results on a set of small test problems.

# 2 The interior-point algorithm

Our main goal is to explore an alternative procedure for the combination of search directions in a line-search based algorithm. In this regard, we are interested in determining the impact this approach may

have in the practical behavior of a nonlinear optimization algorithm. As a consequence, we will introduce an algorithm that uses the combination procedure of interest, but that also computes efficiently the required search directions. This algorithm is based on a primal-dual interior point approach to generate the search directions, and uses an augmented Lagrangian merit function to ensure global convergence. An iterative algorithm of this sort carries out three main tasks in each iteration: i) Compute search directions at the current iterate. In our case we will obtain a descent direction and a direction of negative curvature from the KKT system of linear equations, in addition to the gradient of the merit function. ii) Combine the directions to obtain the next iterate. iii) Update the parameters in the algorithm.

Section 3 will be devoted to our main goal, the description of the combination of directions, while in this section we will describe those issues related to the first and third items, providing only the basic details of the procedures implemented in the algorithm. Additional information can be found in Moguerza and Prieto [20].

## 2.1 Computing the search directions

In the proposed algorithm we solve a sequence of problems (2) such that $\mu_i \rightarrow 0$ for all $i$, following [12]. The search directions are obtained from the application of Newton's method to the primal-dual equations for problem (2),

$$\begin{aligned} \nabla f(x) - \nabla c^T(x)\lambda - \sigma &= 0, \\ c(x) &= 0, \\ \Sigma x &= \mu, \end{aligned} \tag{4}$$

where $\Sigma = \mathrm{diag}(\sigma)$.

Newton's method provides search directions $d_x$, $d_\lambda$ and $d_\sigma$, corresponding to update directions for the variables $x$, $\lambda$ and $\sigma$ respectively. From the first-order Taylor series expansion for the primal-dual KKT conditions (4) about the current values $x$, $\lambda$ and $\sigma$, the resulting system of linear equations defining the search directions is (we omit the dependence on the variables to simplify the notation):

$$\begin{pmatrix} H & -\nabla c^T & -I \\ \nabla c & & \\ \Sigma & & X \end{pmatrix} \begin{pmatrix} d_x \\ d_\lambda \\ d_\sigma \end{pmatrix} = \begin{pmatrix} -\nabla f + \nabla c^T \lambda + \sigma \\ -c \\ \mu - \Sigma x \end{pmatrix}, \tag{5}$$

where $H = \nabla^2_{xx} L(x, \lambda)$, $L(x, \lambda)$ is the Lagrangian function, that is, $L(x, \lambda) = f(x) - \lambda^T c(x)$, $X = \mathrm{diag}(x)$ and $I$ denotes the identity matrix. From the last set of equations in (5), we have

$$d_\sigma = X^{-1}\mu - \sigma - X^{-1}\Sigma d_x. \tag{6}$$

Replacing (6) into the first two sets of equations in (5), the movement direction $d_x$ can be computed as the solution of the symmetric system

$$K \begin{pmatrix} d_x \\ -d_\lambda \end{pmatrix} = \begin{pmatrix} -\nabla f + \nabla c^T \lambda + X^{-1}\mu \\ -c \end{pmatrix}, \tag{7}$$

where $K$ is defined as

$$K = \begin{pmatrix} G & \nabla c^T \\ \nabla c & 0 \end{pmatrix}, \tag{8}$$

for $G = H + X^{-1}\Sigma$. Any ill-conditioning that might arise from the diagonal terms in $G$ is benign, see Wright [25] for example.

The direction obtained from (7) may fail to provide descent for any reasonable merit function, for example when the iterates are close to a stationary point that is not a minimizer. We adapt system (7) to ensure that the direction $d_x$ is a sufficient descent direction. The modified system that we use to define these search directions is

$$\begin{pmatrix} \bar{G}_\rho & \nabla c^T \\ \nabla c & 0 \end{pmatrix} \begin{pmatrix} d_x \\ -d_\lambda \end{pmatrix} = \begin{pmatrix} -\nabla f + \nabla c^T \lambda + X^{-1}\mu \\ -c \end{pmatrix}, \tag{9}$$

where its coefficient matrix (and $\bar{G}_\rho$ in particular) is computed from a modification of

$$K_\rho = \begin{pmatrix} G_\rho & \nabla c^T \\ \nabla c & 0 \end{pmatrix}, \tag{10}$$

for $G_\rho = \nabla^2_{xx} L(x, \lambda - \rho c) + X^{-1}\Sigma$.

An appropriate matrix $\bar{G}_\rho$ for (9), such that $Z_A^T \bar{G}_\rho Z_A$ is positive definite, can be generated in the process of factorizing $K_\rho$, where $Z_A$ has columns that form a basis for the null-space of $\nabla c(x)$. A modified Choleski factorization of the reduced Hessian $Z_A^T G_\rho Z_A$ could be used, as in Gay et al. [15]. This approach requires forming explicitly the reduced Hessian, and as a consequence it is only useful for problems in which this reduced Hessian is not too large. We have chosen to use a version of the symmetric indefinite factorization, see Bunch et al. [5] for example, incorporating the modifications proposed in Forsgren and Murray [14]. This alternative is able to obtain the desired modification for the reduced Hessian directly from system (9), it allows the computation of appropriate directions of negative curvature, as we will indicate below, and it can be applied to medium-sized and large problems. Additional details of the computation of these directions and the factorization used in the algorithm can be found in [14, 20].

We would also want to satisfy the necessary second-order condition at any limit point. For problem (2) this condition requires that

$$Z_A^T \left( \nabla^2_{xx} L(x, \lambda) + M X^{-2} \right) Z_A \quad \text{is p.s.d.}, \tag{11}$$

where $M$ is a diagonal matrix with entries those of $\mu$, that is, $M = \text{diag}(\mu)$. We will use directions of negative curvature to avoid converging to points that do not satisfy (11), but as the direction of negative curvature will be used to obtain iterates that decrease the merit function (3), we also need to ensure that such a direction will be appropriate for our merit function. A direction of negative curvature $d_n$ for our algorithm should lie in the subspace spanned by the columns of $Z_A$ and should satisfy

$$d_n^T (\nabla^2_{xx} L(x, \lambda - \rho c) + M X^{-2}) d_n < 0. \tag{12}$$

As we will justify in Section 3, in our case the choice of an appropriate sign for $d_n$ (to ensure descent, for example) is not relevant, as the search for the next iterate will be performed on a subspace spanned by a combination of directions including $d_n$, and the combination chosen by the algorithm will take the best sign into account automatically.

This direction $d_n$ (assuming that it exists) is computed from the same symmetric indefinite factorization used to obtain the descent direction $d_x$ from (9). Let $K_\rho$ be the matrix defined in (10), and assume that its symmetric indefinite factorization $K_\rho = U^T D U$ has been computed using the algorithm in [14]. Assume also that from the factorization it has been determined that this matrix has more than $m$ negative eigenvalues, implying that $Z_A^T G_\rho Z_A$ has at least one negative eigenvalue. Let $P$ be the permutation matrix associated with the pivoting choices in the factorization algorithm and define $w = P\tilde{w}$, where $\tilde{w}$ satisfies

$$\begin{pmatrix} U_{11} & U_{21} \\ 0 & U_{22} \end{pmatrix} \begin{pmatrix} \tilde{w}_1 \\ \tilde{w}_2 \end{pmatrix} = \pm \sqrt{-\lambda_{\min}(D_2)} \begin{pmatrix} 0 \\ u_\lambda \end{pmatrix}, \tag{13}$$

for a partition of $U$ and $D$ such that $D_1$ and $U_{11}$ correspond to all the pivots taken from elements of $\nabla c$, $\lambda_{\min}(D_2)$ denotes the most negative eigenvalue of $D_2$ and $u_\lambda$ is a unit eigenvector corresponding to this smallest eigenvalue. The direction of negative curvature $d_n$ is defined as the first $n$ components of $w$. Additional details can be found in [14]; in particular, it is shown there that $\nabla c(x) d_n = 0$, and consequently $d_n$ lies in the correct subspace. Also, there exist positive constants $k_1$ and $k_2$ such that

$$d_n^T G_\rho d_n \leq -k_1 \lambda^2_{\min}(Z_A^T G_\rho Z_A) \qquad \text{and} \qquad d_n^T d_n \leq -k_2 \lambda_{\min}(Z_A^T G_\rho Z_A).$$

The direction of negative curvature computed from (13) will satisfy

$$d_n^T (\nabla^2_{xx} L(x, \lambda - \rho c) + X^{-1}\Sigma) d_n < 0$$

If $\sigma - X^{-1}\mu$ is sufficiently small and the constraints are close to zero, $G_\rho$ will be close to the Hessian of the Lagrangian of (2) and the direction $d_n$ computed using the preceding procedure will be a direction

of negative curvature for the merit function $LA(x, \lambda; \rho)$, given that $\nabla c d_n = 0$. As in general $\sigma - X^{-1}\mu$ may not be small, each time a direction of negative curvature is computed we will also check if (12) is satisfied. If this is not the case, the direction of negative curvature $d_n$ will not be used.

A third search direction that will be used to generate the next iterate will be the gradient of the merit function, defined as

$$d_g = \nabla f - X^{-1}\mu - \nabla c^T(\bar{\lambda} - \bar{\rho}c), \tag{14}$$

where $\bar{\lambda}$ and $\bar{\rho}$ will be defined later on.

## 2.2 Updating the parameters

In each iteration the algorithm must update the different parameters involved in the specification of the barrier subproblems (2) and the merit function (3). In the following paragraphs we describe the procedures used to change the multiplier estimates and the barrier parameters.

### 2.2.1 The multipliers

Two sets of dual variables are generated by the algorithm, the equality constraint multipliers $\lambda$ and the approximations to the multipliers for the bound constraints $\sigma$. The multipliers $\lambda$ will be updated using $d_\lambda$ from (9), as described in Section 3.

The solution of the Newton equations (5) provides a search direction for the multipliers $\sigma$, $d_\sigma$, defined in (6). These dual variables will be updated from

$$\sigma_{k+1} = \sigma_k + \alpha_d d_\sigma.$$

The only restriction on the values of the dual variables is their non-negativity. The scalar $\alpha_d$ is chosen as the largest reasonable value that satisfies this condition, as follows. Let

$$\alpha_d = \min\left(\tau \min\left(\left.\frac{-(\sigma_k)_i}{(d_\sigma)_i}\right| (d_\sigma)_i < 0\right), 1\right), \tag{15}$$

where $\tau$ is defined as

$$\tau = \max(0.995, 1 - \|\mu\|_2). \tag{16}$$

This definition is introduced to ensure reasonable local convergence properties for the algorithm, see Yamashita and Yabe [26].

### 2.2.2 The barrier parameters

The vector of barrier parameters in (2) is also updated in each iteration. The updating rule is based on the relationship between the satisfaction of the first-order conditions, the complementarity conditions and the previous values of the barrier parameters. Let $F(x, \lambda, \sigma; \rho)$ be a measure of the satisfaction of the first-order KKT conditions for problem (1) at the current iterate, that is,

$$F(x, \lambda, \sigma; \rho) = \begin{pmatrix} \nabla f(x) - \nabla c(x)^T(\lambda - \rho c) - \sigma \\ c(x) \\ \Sigma x \end{pmatrix}, \tag{17}$$

set

$$\theta = \begin{cases} \|F(x, \lambda, \sigma; \rho)\|_2 & \text{if } \|F(x, \lambda, \sigma; \rho)\|_2 \geq 1, \\ \|F(x, \lambda, \sigma; \rho)\|_2^2 & \text{otherwise,} \end{cases} \tag{18}$$

and define $y = X\sigma$.

The new value for $\mu$ is chosen to ensure a reasonably uniform allocation of the distance from optimality taking into account each complementarity gap. These new values are obtained, in a manner similar to the procedure in [20], from the solution of the problem

$$\begin{array}{ll} \min_\mu & \frac{1}{2}\mu^T\mu \\ \text{s.t.} & y^T\mu = \theta \\ & \mu \geq 0. \end{array} \tag{19}$$

This solution is given by $\mu^* = \omega y$, where $\omega = \theta/(y^T y)$. Definition (18) has been introduced to prevent $\mu_i^*$ from becoming too large when far from a KKT point. On the other hand, if $y_i$ is small then $\mu_i^*$ may become too small. To avoid this situation we compute a reference value $\hat{\mu}$, similar to that in El-Bakry et al. [10],

$$\hat{\mu} = \frac{x^T \sigma}{n},$$

and define the new value of $\mu$ at iteration $k$ as

$$(\mu_{k+1})_i = \min(\delta_k \max(\mu_i^*, \hat{\mu}), (\mu_k)_i), \tag{20}$$

where $\delta_k = \min(0.25, \exp(-(1/\theta_k)))$ and $\theta_k$ is given by (18). Note that $\mu_i$ will not be decreased in every iteration, but only when a sufficient reduction in the satisfaction of the KKT conditions has been achieved. This definition of $\mu$ ensures that $\mu \to 0$ if problem (2) has a solution.

# 3 The computation of a new iterate

We now describe how to combine in an efficient manner our search directions: descent $d_x$, negative curvature $d_n$ (if it exists) and gradient $d_g$. Classical line search methods compute a direction of movement $(d_x, d_\lambda)$ and a scalar $\alpha$ such that the next iterate $(x + \alpha d_x, \lambda + \alpha d_\lambda)$ provides sufficient decrease for an appropriate merit function. This approach works quite well in practice whenever there is a single search direction $d_x$. In our case we may have up to three search directions at a given iteration, and the preceding procedure must be modified to take into account that we search for the next iterate on a subspace of dimension three, as opposed to the univariate classical approach. Following our previous discussion, we proceed first by combining these directions into a trajectory of points of interest, and we then perform a conventional univariate search (a backtracking search) on this trajectory.

## 3.1 Combining the search directions

In the unconstrained case, and given our three search directions, it would seem reasonable to select the new iterate as a point on the trajectory defined by the steepest descent of the objective function from the current iterate, see [1] for example. For our constrained problem (2), we have chosen to apply these ideas to our merit function (3). In particular, we construct this trajectory from the gradient field of the merit function starting from a given iterate $x_k$; it will be given by the solution of the system of ordinary differential equations (we omit the iteration subscript to simplify the notation)

$$\dot{\gamma}(t) = -\nabla LA\left(x + \gamma(t), \bar{\lambda}; \bar{\rho}\right), \quad \gamma(0) = 0, \tag{21}$$

where the reference value for the multipliers, $\bar{\lambda}$, and the penalty parameter value, $\bar{\rho}$, will be defined later on.

Computing this trajectory is too expensive for most practical cases; we will restrict ourselves to solving an approximation to it, in the following two senses:

- We will approximate locally the right-hand side of (21) by a linear function, to obtain a linear system of ODEs, having a closed-form solution.
- We will also restrict ourselves to those points lying on the subspace spanned by our three search directions, to reduce the dimension of the problem and to limit the computational cost.

The ODE that defines the modified trajectory $\beta(t)$ on the two- or three-dimensional subspace of interest will have the following form:

$$\dot{\beta}(t) = -B^T d_g - B^T(W + \bar{\rho}\nabla c^T \nabla c) B\beta(t), \quad \beta(0) = 0,$$

where $B$ denotes an orthonormal basis for the subspace spanned by the search directions and $d_g = \nabla LA(x, \bar{\lambda}; \bar{\rho})$. The matrix $W$ will be either $G_\rho$ as defined in (10), whenever a direction of negative curvature is available, or $\tilde{G}_\rho$ if it is not or has been discarded.

If we introduce the notation $\tilde{H} = B^T(W + \bar{\rho}\nabla c^T \nabla c)B$ and $\tilde{g} = B^T d_g$, we can obtain a closed-form solution for this ODE. If this solution is transformed back to the full space, we obtain the trajectory of interest, $\bar{\gamma}(t)$, given by:

$$\bar{\gamma}(t) = B\beta(t) = B\tilde{H}^{-1}\left(\exp(-\tilde{H}t) - I\right)\tilde{g}. \tag{22}$$

Note that the computation of this trajectory requires only the determination of the exponential of a square matrix of dimension two or at most three.

Interior-point methods ensure the positivity of all iterates, to guarantee that the objective function in (2), and in particular its barrier term, is well defined in each iteration. As a consequence, the trajectory defined by $\bar{\gamma}(t)$ must be transformed into another trajectory that lies within the strict interior of the positive orthant. In our algorithm, this is achieved by projecting each point on the trajectory (22) onto the simple bounds,

$$\hat{\gamma}(t) = \alpha(t)\bar{\gamma}(t),$$

where the scalar $\alpha(t)$ is chosen for each $t$ as

$$\alpha(t) = \min\left\{1, \tau \min\left\{ \frac{x_i}{-\bar{\gamma}_i(t)} \,\middle|\, \bar{\gamma}_i(t) < 0 \right\}\right\}, \tag{23}$$

and $\tau$ is defined as in (16).

The next step would be to determine an acceptable value for the parameter $t$ in $\hat{\gamma}(t)$. As we will show in Section 4, when $\tilde{H}$ is positive definite we have $\bar{\gamma}(t) \to -B\tilde{H}^{-1}\tilde{g} = d_x$, a reasonable step, as $t \to \infty$. As a consequence, we may need to handle infinite values of the parameter $t$ to determine the next iterate. To avoid the complications associated with these values, the curve is reparametrized so that points of interest, such as this Newton step, can be found by moving a finite distance along $\hat{\gamma}$.

We have chosen to use the following reparametrization, see for instance [1, 19],

$$s = \begin{cases} \dfrac{-1}{\delta_m}\left(e^{-\delta_m t} - 1\right) & \text{if } \delta_m \neq 0, \\ t & \text{if } \delta_m = 0, \end{cases} \tag{24}$$

where $\delta_m \leq \ldots \leq \delta_1$ are the eigenvalues of $\tilde{H}$. Under this reparametrization, if $\delta_m > 0$ then $s \in [0, 1/\delta_m]$.

## 3.2 Computing the step

Once the trajectory $\hat{\gamma}(s)$ has been computed, we obtain the next iterate in the variables $x$ from an appropriate step along the curve, $x_{k+1} = x_k + \hat{\gamma}(s)$. The value of $s$ is chosen to ensure sufficient descent for our merit function, and it is found by performing a backtracking search starting at $s_0 = 1/\delta_m$. We determine the step $s_i$ as the first value in the sequence $\{s_0/2^i\}_{i=0}^{\infty}$ that satisfies the following sufficient descent condition:

$$LA(x + \hat{\gamma}(s), \bar{\lambda}; \bar{\rho}) \leq LA(x, \bar{\lambda}; \bar{\rho}) - \sigma s_i \|\nabla LA(x, \bar{\lambda}; \bar{\rho})\|^2. \tag{25}$$

The scalar $\sigma$ has been chosen as a small value $\sigma \in (0, 1)$.

## 3.3 The multiplier estimates

The value of $\bar{\lambda}$ in (21) should be defined to ensure that the sequence of iterates in the algorithm is associated to a decreasing sequence of values for the merit function, to guarantee the global convergence of the algorithm. This value is kept fixed at all trial points in the trajectory. For the search of the new iterate we define

$$\bar{\lambda} = \begin{cases} \lambda + d_\lambda & \text{if } d_n \neq 0 \\ \lambda & \text{otherwise,} \end{cases} \tag{26}$$

In practice, this approach may not be satisfactory for all iterations. At the end of the search procedure, the next iterate $\lambda_{k+1}$ is defined as $\bar{\lambda}$, if there is no negative curvature, the step $s_0$ is accepted and $\alpha(s_0) \geq 0.95$. Otherwise, we use an approach similar to [15]: the value of $\lambda_{k+1}$ is chosen as the least-squares estimate at the accepted step.

### 3.4 Adjusting the penalty parameter

The traditional role of the penalty parameter in a merit function that includes penalty terms, such as (3), is to enforce convergence to points satisfying the constraints $c(x) = 0$. Although the use of the Newton direction should generate iterates that satisfy feasibility in the limit, if the penalty parameter is not chosen to be sufficiently large, the Newton direction may not be a descent direction for the merit function and no valid step will be found.

In the proposed algorithm, the combination of directions to define the trajectory $\bar{\gamma}(s)$ automatically ensures that sufficient descent will be available at all iterates. As a consequence, the penalty parameter is used to attain other reasonable properties for the search trajectory. In particular, we wish to ensure that the Newton step will belong to the trajectory whenever there is no negative curvature in the null-space of the constraints at the current iterate. Note that this may not be true in all cases; a sufficient condition will be that the matrix $\tilde{H}$ is positive definite, as we will show in Section 4.

The absence of negative curvature in the null-space of the constraints $c$ implies that the matrix $W$ (that is, $G_\rho$ in (9) or $\bar{G}_\rho$ if the existing negative curvature was discarded) is positive definite on the subspace spanned by $Z_A$. For the trajectory (22) we would need $\tilde{H}$ to be positive definite, that is, the Hessian $W + \bar{\rho}\nabla c^T \nabla c$ should be positive definite on the subspace spanned by the columns of $B$, for some value of the penalty parameter $\bar{\rho}$ chosen to enforce this property.

If we assume that $\nabla c$ has full row rank, $\tilde{H}$ will be positive definite for large enough values of $\bar{\rho}$. Note that $W$ will be positive definite in the subspace spanned by $Z_A$, while $\nabla c^T \nabla c$ will be positive definite on the orthogonal subspace. If we denote by $\bar{W} = B^T W B$ and $\tilde{J} = B^T \nabla c^T \nabla c B$, we need $\bar{W} + \bar{\rho}\tilde{J}$ to be positive definite. As in this case there is no direction of negative curvature, the matrices $\bar{W}$ and $\tilde{J}$ have dimension 2, and the eigenvalues of $\bar{W} + \rho\tilde{J}$ can be found as the roots of a low-degree polynomial in $\rho$. The value of $\bar{\rho}$ is chosen to be larger than the largest root of this polynomial, or zero if it is negative, and $\rho_{k+1} = \bar{\rho}$.

## 4 Properties of the search

In this section we present some basic properties of the procedure to compute the next iterate. Our aim is not to provide any convergence proof for the algorithm; a detailed proof of this sort will be a matter for a different paper. We only wish to establish that the procedure to combine the search directions is well defined, and has reasonable properties regarding the global convergence of an algorithm that uses appropriate search directions and parameter updates.

We will assume that certain properties are satisfied by the functions defining problem (1) and the iterates generated by the algorithm. A global convergence proof would be the subject of a separate paper, but it should include some of these assumptions and prove that the algorithm satisfies the others.

*A*.1 The iterates $x_k$ generated by the algorithm remain in a compact set, $C \subset \mathbb{R}^n_+$.

*A*.2 The functions $f$ and $c$ have continuous second derivatives in $C$.

*A*.3 For a given value of the barrier parameter $\mu_k$, the iterates $x_k$ are bounded away from zero, $x_k \geq \beta(\mu_k) > 0$, where $\beta^2(\mu_k)/\|\mu_k\| \geq \delta > 0$.

*A*.4 The multiplier estimates $\lambda$ remain bounded in norm at all iterates.

We will ignore the iteration subscript in what follows, whenever the context is clear. We start by establishing that the algorithm is well-defined.

**Lemma 1** *At any iteration $k$, the search described in Section 3 finds a step satisfying (25) in a finite number of iterations.*

**Proof.** From the continuity of $\bar{\gamma}(t)$, $\bar{\gamma}(0) = 0$ and assumption *A*.3 there will exist a value $\bar{t}(\mu_k) > 0$ such that $\alpha(t)$ defined in (23) takes the value one for all $t \in [0, \bar{t}(\mu_k))$. From the reparametrization in (24) and this property, there will exist a value $\bar{s}(\mu_k) > 0$ such that $\hat{\gamma}(s) = \bar{\gamma}(s)$ for all $s \in [0, \bar{s}(\mu_k))$. We will only consider these values of $s$ in what follows. We will also omit the iteration subscript $k$ to simplify the notation.

From the definition of the function $LA$ in (3), the definition of the curve (22), the reparametrization (24) and the Taylor series expansion around $s = 0$ we have

$$LA(x + \hat{\gamma}(s), \bar{\lambda}; \bar{\rho}) - LA(x, \bar{\lambda}; \bar{\rho}) = s\nabla LA(x, \bar{\lambda}; \bar{\rho})^T \frac{d}{ds}\hat{\gamma}(0) + \frac{s^2}{2}\left(\frac{d}{ds}\hat{\gamma}(0)\right)^T \nabla^2 LA(\tilde{x}, \bar{\lambda}; \bar{\rho})\frac{d}{ds}\hat{\gamma}(0),$$

where $\tilde{x} = x + \zeta\hat{\gamma}(s)$ for some $\zeta \in [0, 1]$ and

$$\frac{d}{ds}\hat{\gamma}(0) = \frac{d}{dt}\tilde{\gamma}(0)\frac{dt}{ds} = -B\tilde{g} = -BB^T\nabla LA(x, \bar{\lambda}; \bar{\rho}).$$

Note that $B$ has columns that form an orthonormal basis for a subspace spanned by $\nabla LA(x, \bar{\lambda}; \bar{\rho})$ and other directions. This implies $BB^T\nabla LA(x, \bar{\lambda}; \bar{\rho}) = \nabla LA(x, \bar{\lambda}; \bar{\rho})$. As a consequence,

$$\begin{aligned} & LA(x + \hat{\gamma}(s), \bar{\lambda}; \bar{\rho}) - LA(x, \bar{\lambda}; \bar{\rho}) + \sigma s\|\nabla LA(x, \bar{\lambda}; \bar{\rho})\|^2 \\ = \; & -(1 - \sigma)s\|\nabla LA(x, \bar{\lambda}; \bar{\rho})\|^2 + \frac{s^2}{2}\nabla LA(x, \bar{\lambda}; \bar{\rho})^T\nabla^2 LA(\tilde{x}, \bar{\lambda}; \bar{\rho})\nabla LA(x, \bar{\lambda}; \bar{\rho}). \end{aligned} \tag{27}$$

If $\sigma < 1$, the desired result follows from this relationship. □

To prove global convergence for an algorithm based on this search we should have sufficient descent on the merit function in every iteration, this function should be bounded below and we would also need the value of the parameter $s$ to be bounded away from zero. As we show in Lemma 2, this last property follows from the same arguments as the preceding derivation.

**Lemma 2** *The step $s$ along the curve in each iteration is bounded away from zero by a positive value, $s \geq \hat{s} > 0$.*

**Proof.** From the fact that $B$ has orthonormal columns and assumptions $A.1$ to $A.4$, it follows that there will exist a positive constant $\bar{\beta}$ such that

$$\nabla LA(x, \bar{\lambda}; \bar{\rho})^T\nabla^2 LA(\tilde{x}, \bar{\lambda}; \bar{\rho})\nabla LA(x, \bar{\lambda}; \bar{\rho}) \leq \left(\bar{\beta} + \frac{\bar{\beta}\|\mu\|}{\beta^2(\mu)}\right)\|\nabla LA(x, \bar{\lambda}; \bar{\rho})\|^2 \leq 2\bar{\beta}\|\nabla LA(x, \bar{\lambda}; \bar{\rho})\|^2.$$

The dependence on $\mu$ is a consequence of the terms $MX^{-2}$ in $\nabla^2 LA(\tilde{x}, \bar{\lambda}; \bar{\rho})$ and assumption $A.3$.

Given this bound, from (27) it will follow that

$$LA(x + \hat{\gamma}(s), \bar{\lambda}; \bar{\rho}) - LA(x, \bar{\lambda}; \bar{\rho}) - \sigma s\|\nabla LA(x, \bar{\lambda}; \bar{\rho})\|^2 < 0$$

for all $s \in (0, \bar{s})$, where

$$\bar{s} = \frac{(1 - \sigma)}{\bar{\beta}}.$$

As a consequence of this result and the backtracking search implemented in the algorithm, the computed step will satisfy $s \geq \hat{s} \equiv (1 - \sigma)/(2\bar{\beta})$. □

We will prove a last result in this section, related to the desirable local convergence properties of the algorithm. We show that if the initial trial value $s_0$ is accepted, in that iteration we update the variables using the Newton direction. As a consequence, the superlinear convergence of the algorithm will follow from this result if we are able to accept this step and we update $\mu$ appropriately. Lemma 3 is an extension of a similar result for unconstrained problems in [1].

**Lemma 3** *In those iterations where no negative curvature is used and the multiplier estimate is taken as $\lambda + d_\lambda$, if $\lim_{t\to\infty} \alpha(t) = 1$ and $\bar{\rho}$ has been chosen as indicated in Section 3.4, we have*

$$\lim_{t\to\infty} \hat{\gamma}(t) = \hat{\gamma}(s_0) = d_x.$$

**Proof.** From the condition $\lim_{t\to\infty}\alpha(t)=1$, it will be enough to show that if no negative curvature is used then $\lim\bar{\gamma}(t)=d_x$. In this case we have $W=\bar{G}_\rho$, a positive definite matrix, and $\bar{\lambda}=\lambda+d_\lambda$. From (22), as $\bar{\rho}$ has been chosen to ensure that $\tilde{H}$ is positive definite,

$$\lim_{t\to\infty}\bar{\gamma}(t)=\lim_{t\to\infty}B\tilde{H}^{-1}\left(\exp(-\tilde{H}t)-I\right)\tilde{g}=-B\tilde{H}^{-1}\tilde{g}.$$

From (9) we have

$$(W+\bar{\rho}\nabla c^T\nabla c)d_x=-\nabla LA(x,\lambda+d_\lambda;\bar{\rho}),$$

and using the definitions of $\tilde{H}$ and $\tilde{g}$, as $B$ has columns that form an orthonormal basis for a subspace containing $d_x$, implying $BB^Td_x=d_x$, we obtain

$$\begin{aligned}\lim_{t\to\infty}\bar{\gamma}(t) &= -B(B^T(W+\bar{\rho}\nabla c^T\nabla c)B)^{-1}B^T\nabla LA(x,\lambda+d_\lambda;\bar{\rho})\\ &= B(B^T(W+\bar{\rho}\nabla c^T\nabla c)B)^{-1}B^T(W+\bar{\rho}\nabla c^T\nabla c)d_x\\ &= BB^Td_x=d_x.\end{aligned}$$

□

# 5 Implementation and numerical results

## 5.1 The algorithm

We present a scheme of the proposed interior point algorithm (**G**radient **F**low **I**nterior **P**oint **M**ethod - **GFIPM**), summarizing those aspects described in the previous sections.

***Algorithm GFIPM***

*Choose initial values for* $x_0$, $\lambda_0$ *and* $\sigma_0$.
*Choose initial values for the scalar* $\rho_0$ *and the vector* $\mu_0$
*Set* $k=0$
***repeat***
  *Compute* $d_x$ *and* $d_\lambda$ *from (9) using the factorization described in [14], and* $d_\sigma$ *from (6)*
  *Compute, if it exists,* $d_n$, *a direction of negative curvature from (13)*
  *Set* $d_n=0$ *if (12) is not satisfied*
  *Compute* $\bar{\rho}$ *from the procedure in 3.4*
  *Compute* $\bar{\lambda}$ *from (26)*
  *Compute* $s$ *using a backtracking search until (25) is satisfied*
  $x_{k+1}=x_k+\hat{\gamma}(s)$
  *Update* $\lambda_{k+1}$ *from* $\lambda_k$ *and* $d_\lambda$ *using the procedure in 3.3*
  *Compute* $\alpha_d$ *from (15)*
  $\sigma_{k+1}=\sigma_k+\alpha_d d_\sigma$
  *Compute the updated barrier vector* $\mu_{k+1}$ *from (20)*
  $\rho_{k+1}=\bar{\rho}$
  $k=k+1$
***until*** *convergence*

## 5.2 Numerical results

We have conducted a set numerical experiments on a collection of test problems using algorithm GFIPM. The algorithm has been implemented and the tests have been carried out in MATLAB. The test set we have considered is composed of 140 small problems from the CUTE collection, see Bongartz et al. [3]. selected from those nonlinear constrained problems having less than 100 variables and continuous derivatives (note that exact first and second derivatives have been used). The algorithm has been implemented to include both lower and upper bounds in the barrier terms.

Whenever possible, the initial points given in CUTE have been used. Sometimes these initial points do not satisfy the bound constraints. Such points have been transformed following a strategy similar to that described in Vanderbei and Shanno [23]. Table 1 shows the results obtained by GFIPM for these problems. The termination criterion used has been

$$\|F(x, \lambda, \sigma; \rho)\| \leq \epsilon(1 + \|\nabla f(x)\|),$$

where $\epsilon = 10^{-8}$, except for problems `DISC2` and `HS91`, where $\epsilon = 10^{-7}$.

The columns in the table correspond to:

- `Prob.`: problem name.
- `Const.`: norm of the constraint vector, $\|c(x)\|$, at the solution, including slacks.
- `KKT`: norm of the first-order KKT conditions at the solution, $\|F(x, \lambda, \sigma; \rho)\|$.
- `Iter.`: iteration count (number of factorizations of the primal-dual system).
- `Eval.`: number of evaluations of the objective function and the constraints.
- `NC`: number of iterations in which directions of negative curvature were used.

In those cases where negative curvature was detected the problem was solved a second time, setting the direction of negative curvature to zero. Table 1 includes two lines for those problems, one for the results from each of the two versions of the algorithm.

## 5.3 Analysis of the results

The algorithm was able to solve all problems but one, problem HS13 (that does not satisfy a constraint qualification at the solution). For some of the problems the code finds better local minimizers than those given in [18] (this happened for problems HS105, HS106, HS107, HS112 and HS116), while for other problems these local minimizers are worse (HS59, HS70, HS97, HS98 and HS108). Problem HS99 is an example of a badly scaled problem. The termination tolerance is satisfied when the norm of the first-order KKT conditions is 0.4994. Introducing a more demanding stopping criterion (a tolerance of $10^{-14}$), the norm of the KKT conditions goes down to $10^{-6}$ after 3 additional iterations, but the value of the merit function remains basically unaltered.

In general, the number of iterations required to solve the problems is fairly small. The number of function evaluations is higher, but no particular care was taken when implementing a strategy to find a value of the parameter $s$ that satisfied (25); a standard backtracking search was used. It is also interesting to note the large number of cases in which a step $s_0$ (the equivalent to a unit step) was accepted.

Table 2 presents a brief summary of the results, both iteration counts and function evaluations, for all problems that make use of negative curvature, as well as the size of these problems.

For the whole test set, negative curvature was used in only 8% of the cases. The gradient direction would seem to take care of some negative curvature information; in [20], where a standard line search procedure is used (without any gradient information), negative curvature was used for 23% of the problems in a similar test set.

The preceding table also includes a certain number of cases in which using negative curvature was worse than ignoring it. Globally, the reductions in iterations and function evaluations seem to be more significant than the increases. The largest deterioration in the number of iterations amounted to 9 iterations (39%) for problem `PRODPL1` and 25 function evaluations (56%) for problem `HS24`, while the largest improvement was 41 iterations (87%) and 63 function evaluations (91%) for problem `POLAK5`. Nevertheless, from the observation of the different behavior in the numbers of iterations and function evaluations, special care should be taken when computing the parameter $s$ in the search, in order to reduce the number of function evaluations whenever negative curvature is used. For example, a procedure based on polynomial models for the univariate search would be likely to contribute to the improvement in the behavior of the algorithm.

Table 1: Results for small-size problems

| Prob. | Obj. | Const. | KKT | Iter. | Eval. | NC |
|---|---|---|---|---|---|---|
| AIRPORT | 47952.7017 | 3.1e-14 | 9.9e-12 | 15 | 15 | 0 |
| ALJAZZAF | 75.005 | 2.5e-09 | 8.9e-07 | 20 | 37 | 0 |
| ALSOTAME | 0.08208499 | 0 | 7.1e-11 | 8 | 8 | 0 |
| BIGGSC4 | -24.499999 | 1.5e-15 | 5.2e-08 | 21 | 26 | 2 |
| | -24.5 | 1.8e-15 | 1.9e-15 | 21 | 21 | 0 |
| CANTILVR | 1.33995636 | 3.1e-11 | 3.3e-11 | 16 | 58 | 0 |
| CB2 | 1.95222449 | 6.9e-12 | 7.6e-12 | 11 | 14 | 0 |
| CB3 | 2.0 | 2.5e-12 | 2.5e-12 | 10 | 23 | 0 |
| CHACONN1 | 1.95222449 | 1.6e-11 | 2.8e-11 | 8 | 9 | 0 |
| CHACONN2 | 2.0 | 2.5e-11 | 4.3e-11 | 10 | 11 | 0 |
| CONGIGMZ | 28.0 | 8.5e-12 | 8.9e-12 | 21 | 34 | 0 |
| CSFI1 | -49.0752 | 1.3e-10 | 1.5e-09 | 11 | 13 | 0 |
| CSFI2 | 55.0176056 | 1.2e-13 | 1.7e-13 | 14 | 17 | 0 |
| DEMYMALO | -3.0 | 2.3e-11 | 2.5e-11 | 11 | 13 | 0 |
| DIPIGRI | 680.63006 | 1.6e-08 | 3.6e-08 | 11 | 26 | 1 |
| | 680.63006 | 1.7e-11 | 4.4e-11 | 12 | 18 | 0 |
| DISC2 | 1.5624999 | 2.4e-08 | 2.4e-08 | 63 | 208 | 0 |
| DUAL1 | 0.035012968 | 1.9e-16 | 7.0e-12 | 21 | 21 | 0 |
| DUAL2 | 0.033733671 | 6.1e-16 | 9.0e-09 | 13 | 13 | 0 |
| DUAL4 | 0.746090649 | 2.1e-16 | 2.3e-08 | 14 | 14 | 0 |
| EXPFITA | 0.0011366117 | 1.8e-14 | 1.0e-09 | 32 | 32 | 0 |
| FCCU | 11.14910914 | 4.4e-15 | 4.2e-14 | 8 | 8 | 0 |
| GIGOMEZ1 | -3.0 | 1.9e-14 | 2.0e-14 | 11 | 20 | 0 |
| HATFLDH | -24.5 | 2.7e-15 | 3.5e-15 | 14 | 20 | 1 |
| | -24.5 | 2.9e-15 | 3.9e-15 | 13 | 14 | 0 |
| HIMMELBI | -1735.569579 | 8.0e-14 | 3.2e-11 | 29 | 29 | 0 |
| HIMMELBK | 0.0518143 | 3.5e-12 | 3.5e-12 | 18 | 18 | 0 |
| HIMMELP2 | -8.19803189 | 3.8e-14 | 3.8e-14 | 11 | 15 | 0 |
| HIMMELP3 | -59.0131239 | 5.2e-10 | 4.9e-09 | 8 | 23 | 0 |
| HIMMELP4 | -59.0131239 | 9.7e-13 | 9.8e-13 | 11 | 12 | 0 |
| HIMMELP5 | -59.0131239 | 3.5e-09 | 3.6e-09 | 44 | 148 | 0 |
| HIMMELP6 | -59.0131239 | 5.7e-12 | 5.9e-12 | 16 | 39 | 0 |
| HONG | 22.57108736 | 0 | 6.4e-13 | 7 | 7 | 0 |
| HS10 | -0.9999999 | 1.4e-08 | 1.4e-08 | 13 | 46 | 0 |
| HS11 | -8.49846422 | 1.4e-14 | 2.4e-14 | 7 | 7 | 0 |
| HS12 | -30.0 | 1.2e-08 | 1.2e-08 | 8 | 8 | 0 |
| HS13 | -- | -- | -- | -- | -- | -- |
| HS14 | 1.39346498 | 5.1e-12 | 2.2e-11 | 9 | 38 | 0 |
| HS15 | 306.50 | 8.1e-13 | 3.1e-10 | 16 | 34 | 0 |
| HS16 | 0.25 | 1.1e-16 | 2.5e-16 | 13 | 14 | 0 |

Table 1: (cont.) Results for small-size problems

| Prob. | Obj. | Const. | KKT | Iter. | Eval. | NC |
|---|---|---|---|---|---|---|
| HS17 | 1.0 | 1.7e-12 | 5.5e-10 | 17 | 79 | 0 |
| HS18 | 5.0 | 0 | 5.6e-17 | 28 | 188 | 0 |
| HS19 | -6961.81388 | 2.1e-11 | 2.0e-08 | 14 | 18 | 0 |
| HS20 | 40.19873021 | 2.1e-09 | 1.6e-06 | 6 | 16 | 0 |
| HS21 | -99.9599999 | 3.6e-15 | 2.9e-14 | 5 | 5 | 0 |
| HS21MOD | -99.9599999 | 0 | 9.7e-16 | 11 | 11 | 0 |
| HS22 | 1.0 | 3.3e-09 | 1.5e-08 | 5 | 5 | 0 |
| HS23 | 2.0 | 1.8e-12 | 1.8e-12 | 8 | 9 | 0 |
| HS24 | -4.0e-97 | 0 | 6.4e-19 | 13 | 31 | 1 |
| | -1.0 | 7.1e-19 | 7.0e-11 | 6 | 6 | 0 |
| HS29 | -22.62741699 | 7.4e-10 | 8.7e-10 | 7 | 8 | 0 |
| HS30 | 1.0 | 2.4e-09 | 5.0e-09 | 5 | 5 | 0 |
| HS31 | 5.999999 | 9.4e-12 | 3.9e-09 | 5 | 5 | 0 |
| HS32 | 1.0 | 7.8e-11 | 4.5e-10 | 8 | 9 | 0 |
| HS33 | -4.5857864 | 3.2e-11 | 3.2e-11 | 8 | 8 | 0 |
| HS34 | -0.83403244 | 4.3e-12 | 4.3e-12 | 8 | 8 | 0 |
| HS35 | 0.11111111 | 1.1e-17 | 1.9e-10 | 7 | 7 | 0 |
| HS36 | -3299.9999 | 3.5e-15 | 9.8e-12 | 8 | 8 | 1 |
| | -3299.9999 | 2.9e-27 | 9.7e-12 | 8 | 8 | 0 |
| HS37 | -3456 | 2.8e-21 | 9.7e-14 | 6 | 6 | 0 |
| HS41 | 1.92592592 | 0 | 1.4e-12 | 7 | 7 | 0 |
| HS43 | -44.0 | 6.5e-12 | 3.2e-11 | 9 | 9 | 0 |
| HS44 | -13.0 | 1.0e-15 | 2.2e-15 | 9 | 9 | 0 |
| HS44NEW | -13.0 | 1.0e-15 | 2.2e-15 | 9 | 9 | 0 |
| HS53 | 4.0930232 | 1.8e-15 | 6.2e-14 | 4 | 4 | 0 |
| HS59 | -6.749505 | 1.0e-14 | 1.2e-14 | 66 | 202 | 0 |
| HS60 | 0.03256682 | 6.5e-12 | 1.8e-11 | 7 | 7 | 0 |
| HS63 | 961.7151721 | 1.2e-08 | 2.4e-08 | 6 | 9 | 0 |
| HS64 | 6299.84243 | 1.1e-16 | 9.1e-13 | 17 | 22 | 0 |
| HS65 | 0.95352886 | 3.5e-15 | 4.4e-15 | 10 | 14 | 1 |
| | 0.95352886 | 7.1e-15 | 7.2e-15 | 11 | 12 | 0 |
| HS66 | 0.518163274 | 5.1e-15 | 5.3e-15 | 10 | 10 | 0 |
| HS67 | -1162.119226 | 2.3e-12 | 2.3e-12 | 8 | 10 | 0 |
| HS68 | -0.920425 | 1.2e-16 | 1.2e-14 | 27 | 72 | 0 |
| HS69 | -956.712887 | 1.1e-10 | 1.6e-07 | 12 | 12 | 0 |
| HS70 | 0.1870436431 | 2.9e-11 | 1.2e-09 | 22 | 39 | 0 |
| HS71 | 17.0140173 | 4.1e-08 | 4.1e-08 | 8 | 8 | 0 |
| HS72 | 727.67936 | 3.4e-16 | 1.2e-13 | 22 | 42 | 0 |
| HS73 | 29.894378 | 1.0e-08 | 1.1e-08 | 11 | 11 | 0 |
| HS74 | 5126.4981 | 1.4e-12 | 4.6e-10 | 8 | 8 | 0 |
| HS75 | 5174.4127 | 6.8e-13 | 2.0e-09 | 8 | 8 | 0 |

Table 1: (cont.) Results for small-size problems

| Prob. | Obj. | Const. | KKT | Iter. | Eval. | NC |
|---|---|---|---|---|---|---|
| HS76 | -4.681818181 | 8.3e-16 | 1.3e-09 | 7 | 7 | 0 |
| HS80 | 0.0539498 | 3.0e-10 | 3.1e-10 | 7 | 9 | 0 |
| HS81 | 0.0539498 | 1.5e-10 | 1.5e-10 | 8 | 8 | 0 |
| HS83 | -30665.539 | 3.2e-14 | 7.7e-13 | 18 | 18 | 0 |
| HS84 | -5280335.13 | 5.6e-08 | 1.6e-04 | 19 | 23 | 0 |
| HS86 | -32.348679 | 1.0e-14 | 3.6e-08 | 14 | 14 | 0 |
| HS88 | 1.362656815 | 3.2e-14 | 5.0e-10 | 24 | 314 | 0 |
| HS91 | 1.36265681 | 4.4e-11 | 4.8e-08 | 18 | 167 | 0 |
| HS93 | 135.075963 | 3.2e-15 | 1.5e-07 | 9 | 9 | 0 |
| HS95 | 0.0156195 | 3.4e-12 | 3.4e-12 | 11 | 11 | 0 |
| HS96 | 0.0156195 | 1.7e-12 | 1.7e-12 | 11 | 11 | 0 |
| HS97 | 4.0712463 | 2.2e-10 | 4.4e-08 | 12 | 34 | 0 |
| HS98 | 4.0712463 | 6.8e-14 | 6.7e-11 | 15 | 33 | 0 |
| HS99 | -8.3108e+08 | 2.9e-11 | 0.49945443 | 6 | 6 | 0 |
| HS100 | 680.630057 | 1.6e-08 | 3.6e-08 | 11 | 26 | 1 |
| | 680.630057 | 1.7e-11 | 4.4e-11 | 12 | 18 | 0 |
| HS104 | 3.9511634 | 4.4e-10 | 2.4e-09 | 9 | 12 | 0 |
| HS105 | 1044.725129 | 2.0e-17 | 1.1e-10 | 16 | 19 | 1 |
| | 1044.725129 | 2.6e-18 | 5.0e-11 | 16 | 19 | 0 |
| HS106 | 7049.24802 | 3.9e-10 | 3.9e-10 | 10 | 54 | 0 |
| HS107 | 4797.98188 | 2.6e-10 | 1.0e-05 | 10 | 78 | 0 |
| HS108 | -0.6749814 | 1.5e-14 | 1.9e-14 | 12 | 15 | 0 |
| HS109 | 5362.06918 | 4.8e-08 | 4.9e-08 | 12 | 32 | 0 |
| HS110 | -45.7784697 | -- | 4.8e-13 | 5 | 5 | 0 |
| HS111 | -47.7610913 | 2.7e-08 | 4.9e-08 | 12 | 32 | 0 |
| HS112 | -47.7610908 | 2.5e-06 | 1.8e-08 | 11 | 11 | 0 |
| HS113 | 24.306209 | 1.6e-11 | 2.9e-11 | 33 | 55 | 0 |
| HS114 | -1768.80696 | 2.1e-11 | 7.9e-11 | 16 | 16 | 0 |
| HS116 | 97.5875096 | 5.6e-09 | 7.6e-09 | 33 | 40 | 0 |
| HS117 | 32.3486790 | 3.4e-10 | 1.0e-09 | 17 | 19 | 0 |
| HS118 | 664.820450 | 2.1e-14 | 1.1e-12 | 14 | 14 | 0 |
| HS119 | 244.899697 | 6.3e-16 | 2.9e-07 | 11 | 11 | 0 |
| HS268 | 4.9e-9 | 9.7e-15 | 9.8e-09 | 17 | 19 | 0 |
| HUBFIT | 0.016893495 | 2.9e-17 | 2.8e-09 | 7 | 7 | 0 |
| KIWCRESC | 1.2e-09 | 3.3e-09 | 3.8e-09 | 11 | 16 | 0 |
| LAUNCH | 9.004903149 | 6.8e-08 | 6.8e-08 | 22 | 24 | 1 |
| | 9.004903149 | 5.1e-10 | 3.8e-07 | 15 | 15 | 0 |
| LIN | -0.020198312 | 4.4e-17 | 7.5e-15 | 15 | 16 | 0 |
| LOADBAL | 0.4528510391 | 1.3e-13 | 2.0e-10 | 13 | 13 | 0 |
| MADSEN | 0.616432435 | 9.7e-12 | 4.7e-11 | 15 | 33 | 0 |
| MAKELA1 | -1.414213564 | 1.1e-13 | 1.8e-11 | 19 | 24 | 0 |
| MAKELA2 | 7.1999999 | 6.4e-11 | 8.5e-11 | 7 | 7 | 0 |

Table 1: (cont.) Results for small-size problems

| Prob. | Obj. | Const. | KKT | Iter. | Eval. | NC |
|---|---|---|---|---|---|---|
| MIFFLIN1 | -1.0 | 3.2e-09 | 1.5e-08 | 5 | 5 | 0 |
| MIFFLIN2 | -0.9999999 | 1.4e-10 | 1.8e-10 | 13 | 28 | 0 |
| MINMAXBD | 115.7064397 | 6.4e-11 | 6.4e-11 | 25 | 36 | 0 |
| MINMAXRB | 3.5e-17 | 1.3e-11 | 1.3e-11 | 7 | 13 | 0 |
| MISTAKE | -1.0 | 5.6e-09 | 6.3e-09 | 10 | 10 | 0 |
| ODFITS | -2380.026775 | 8.0e-13 | 8.4e-13 | 8 | 8 | 0 |
| POLAK1 | 2.718281833 | 3.5e-14 | 6.1e-14 | 8 | 8 | 0 |
| POLAK3 | 5.9330033 | 2.2e-09 | 4.7e-09 | 15 | 50 | 0 |
| POLAK4 | 6.0e-17 | 3.9e-14 | 3.9e-14 | 20 | 22 | 0 |
| POLAK5 | 49.99999 | 1.4e-08 | 1.7e-08 | 6 | 6 | 1 |
| | 49.99999 | 2.0e-08 | 2.0e-08 | 47 | 69 | 0 |
| PRODPL0 | 58.79009997 | 2.3e-09 | 4.5e-08 | 16 | 16 | 0 |
| PRODPL1 | 35.73896744 | 2.4e-12 | 1.7e-11 | 23 | 37 | 2 |
| | 35.73896744 | 2.1e-14 | 3.4e-13 | 14 | 16 | 0 |
| QPCBLEND | -0.0078425 | 5.3e-15 | 4.1e-11 | 43 | 43 | 0 |
| QPNBLEND | -0.00913614 | 1.4e-14 | 5.8e-08 | 23 | 23 | 0 |
| ROSENMMX | -44.0 | 1.7e-13 | 1.7e-13 | 31 | 87 | 0 |
| S268 | 4.9e-09 | 9.7e-15 | 9.9e-15 | 17 | 19 | 0 |
| TAME | 3.1e-33 | 0 | 2.0e-15 | 3 | 4 | 0 |
| TENBARS4 | 368.4931619 | 9.9e-12 | 9.9e-12 | 15 | 18 | 0 |
| TRUSPYR1 | 11.22874087 | 2.4e-12 | 1.1e-11 | 9 | 9 | 0 |
| TRUSPYR2 | 11.22874090 | 5.9e-12 | 6.0e-12 | 12 | 12 | 0 |
| TRY-B | 1.2e-25 | 1.9e-10 | 1.9e-10 | 10 | 11 | 0 |
| TWOBARS | 1.508652417 | 2.3e-15 | 4.0e-15 | 13 | 170 | 0 |
| WOMFLET | 6.6e-13 | 5.6e-10 | 5.6e-10 | 11 | 23 | 0 |
| ZECEVIC2 | -4.125 | 6.3e-16 | 9.0e-09 | 8 | 40 | 0 |
| ZECEVIC3 | 97.30945002 | 7.0e-09 | 3.1e-08 | 10 | 10 | 0 |
| ZECEVIC4 | 7.557507769 | 4.8e-16 | 7.3e-15 | 9 | 13 | 0 |
| ZY2 | 2.0 | 3.8e-09 | 3.8e-09 | 6 | 6 | 0 |

Table 2: Problems using directions of negative curvature.

| Prob. | Var. | Cons. | Iter.cn | Iter. | Eval.cn | Eval. |
|---|---|---|---|---|---|---|
| HS24 | 2 | 3 | 13 | 6 | 31 | 6 |
| HS36 | 3 | 1 | 8 | 8 | 8 | 8 |
| HS65 | 3 | 1 | 10 | 11 | 14 | 12 |
| POLAK5 | 3 | 2 | 6 | 47 | 6 | 69 |
| BIGGSC4 | 4 | 7 | 21 | 21 | 26 | 21 |
| HATFLDH | 4 | 7 | 14 | 13 | 20 | 14 |
| DIPIGRI | 7 | 4 | 11 | 12 | 26 | 18 |
| HS100 | 7 | 4 | 11 | 12 | 26 | 18 |
| HS105 | 8 | 1 | 16 | 16 | 19 | 19 |
| LAUNCH | 25 | 28 | 22 | 15 | 24 | 15 |
| PRODPL1 | 60 | 29 | 23 | 14 | 37 | 16 |
| TOTAL | | | 155 | 175 | 237 | 216 |

# 6 Conclusions

We have described a procedure to combine different search directions, including directions of negative curvature if they exist, and an algorithm to solve general nonlinear optimization problems, based on a primal-dual approach, that uses the combination procedure. The algorithm has been shown to be efficient on a set of small test problems. The combination of the directions is also very efficient, as shown in the reduced number of iterations required by the algorithm. A clear advantage is that the scaling of the different directions is done in a natural way.

Although this procedure has been applied to cases in which we had either two or three directions to combine, it would be straightforward to extend it to additional directions, such as for example additional directions of negative curvature if they are available.

The impact of the negative curvature is not very significant on these small problems (it is used in only 8% of them), possibly due to the use of the gradient in the search, but it can be quite important in some cases. Given the limited cost of computing a direction of negative curvature whenever an appropriate factorization is used to obtain the movement directions, we think it is reasonable for nonconvex problems to take into account this second-order information whenever it is available.

# References


[1] Behrman, W. *An efficient gradient flow method for unconstrained optimization.* Ph. D. Thesis. Stanford University, June 1998.

[2] Bertsekas, D. P. *Constrained optimization and Lagrange multiplier methods.* Academic Press, New York, 1982.

[3] Bongartz, I., Conn, A. R., Gould, N. I. M. and Toint, Ph. L. "CUTE: Constrained and Unconstrained Testing Environment", *Transactions of the ACM on Mathematical Software* **21**, pp. 123–160. 1995.

[4] Botsaris, C. A. "A curvilinear optimization method based upon iterative estimation of the eigensystem of the Hessian matrix", *J. Math. Anal. Appl.* **63**, pp. 396-411, 1978.

[5] Bunch, J. R., Kaufman, L. and Parlett, B. N. "Decomposition of a symmetric matrix", *Numer. Math.* **27**, pp. 95–109, 1976.

[6] Byrd, R.H., Schnabel, R.B. and Shultz, G.A. "Approximate solution of the trust region problem by minimization over two-dimensional subspaces", *Mathematical Programming* **40**, pp. 247–263, 1988.

[7] Conn, A.R., Gould, N. and Toint, Ph.L. "A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds", *SIAM Journal on Numerical Analysis* **28**, pp. 545-572, 1991.

[8] Courant, R. "Variational methods for the solution of problems of equilibrium and vibrations", *Bull. Amer. Math. Soc.* **49**, pp. 1-23, 1943.

[9] Dennis, J.E. and Schnabel, R.B. (1983) *Numerical methods for unconstrained optimization and nonlinear equations.* Prentice Hall: Englewood Cliffs.

[10] El-Bakry, A. S., Tapia, R. A., Tsuchiya, T. and Zhang, Y. "On the formulation and theory of the Newton interior-point method for nonlinear programming", *Journal Optim. Theory Applications* **89**, pp. 507–541, 1996.

[11] Evtushenko, Y. E. and Zhadan, V. G. "Stable barrier-projection and barrier-Newton methods in nonlinear programming", In: Optimization Methods and Software, Vol. 3, pp. 237-256, 1994.

[12] Fiacco, A. V. and McCormick, G. P. *Nonlinear programming: Sequential unconstrained minimization techniques.* Society for Industrial And Applied Mathematics, Philadelphia, 1990 (Originally published by Research Analysis Corporation, McLean, Virginia).

[13] Fletcher, R. *Practical Methods of Optimization.* John Wiley, Chichester, 1991.

[14] Forsgren, A. and Murray, W. "Newton methods for large-scale linear equality-constrained minimization", *SIAM J. on Matrix Analysis and Applications* **14**, pp. 560–587, 1993.

[15] Gay, D. M., Overton, M. L. and Wright, M. H. *A primal-dual interior method for nonconvex nonlinear programming.* Technical Report 97-4-08. Computing Science Research Center, Bell Laboratories, Murray Hill, New Jersey, 1997.

[16] Gill, P. E., Murray, W., Saunders, M. A. and Wright, M. H. *User's guide for NPSOL (version 4.0): A FORTRAN package for nonlinear programming.* Technical Report SOL 86-2. Stanford University, 1986.

[17] Gill, P. E., Murray, W. and Wright, M. H. *Practical optimization.* Academic Press, London/New York, 1981.

[18] Hock, W. and Schittkowski, K. *Test examples for nonlinear programming codes.* Springer Verlag. Berlin, 1981.

[19] Moguerza, J.M. "Interior point methods for nonconvex optimization", Ph.D. Thesis, Department of Statistics and Econometrics, Universidad Carlos III de Madrid, 2000 (in Spanish).

[20] Moguerza, J.M. and Prieto, F.J. "An augmented Lagrangian interior-point method using directions of negative curvature", Working Paper 00-36. Department of Statistics and Econometrics, Universidad Carlos III de Madrid, 2000.

[21] Moré, J.J. and Sorensen, D.C. "On the use of directions of negative curvature in a modified Newton method", *Math. Programming* **16**, pp. 1–20, 1979.

[22] Schropp, J. "Using dynamical systems to solve minimization problems", *Applied Numerical Mathematics* **18**, pp. 321-335, 1995.

[23] Vanderbei, R. J. and Shanno, D. F. *An interior-point algorithm for nonconvex nonlinear programming.* Technical Report SOR-97-21, Princeton University, 1997.

[24] Wright, M.H. "Interior methods for constrained optimization", In: *Acta Numerica 1992*, A. Iserles, ed. Cambridge University Press, N.Y., pp. 341-402, 1992.

[25] Wright, M.H. *Ill-conditioning and computational error in primal-dual methods for nonlinear programming.* Technical Report 97-4-04. Computing Science Research Center, Bell Laboratories, Murray Hill, New Jersey, 1997.

[26] Yamashita, H and Yabe, H. "Superlinear and quadratic convergence of some primal-dual interior point methods for constrained optimization", *Math. Programming* **75**, pp. 377–397, 1996.

[27] Zang, I. "A new arc algorithm for unconstrained optimization", *Math. Programming* **15**, pp. 36-52, 1978.