JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS: Vol. 47, No. 1, SEPTEMBER 1985

Numerical Decomposition of a Convex Function[1]

M. LAMOUREUX[2] AND H. WOLKOWICZ[3]

Communicated by A. V. Fiacco

Abstract. Given the $n \times p$ orthogonal matrix A and the convex function $f: R^n \to R$, we find two orthogonal matrices P and Q such that f is almost constant on the convex hull of $\pm$ the columns of P, f is sufficiently nonconstant on the column space of Q, and the column spaces of P and Q provide an orthogonal direct sum decomposition of the column space of A. This provides a numerically stable algorithm for calculating the cone of directions of constancy, at a point x, of a convex function. Applications to convex programming are discussed.

Key Words. Convex functions, convex programming, cone of directions of constancy, numerical stability.

1. Introduction

The *cone of directions of constancy* at x of a convex function $f: R^n \to R$ is defined as

$$D_f(x) = \{d \in R^n\text{: there exists } \bar{\alpha} > 0 \text{ such that } f(x+\alpha d) = f(x),$$
$$\text{for all } 0 < \alpha \leq \bar{\alpha}\}. \tag{1}$$

It has recently been used in various characterizations of optimality in convex programming (see, e.g., Refs. 1 and 2). It has proven to be a key ingredient in solving ill-posed and ill-conditioned convex programs, i.e., convex programs for which no constraint qualification may hold (see, e.g., Refs. 1

[1] This work was supported by the National Science and Engineering Research Council of Canada (Grant No. A3388 and Summer Grant).

[2] Graduate Student, Department of Mathematics, University of California at Berkeley, Berkeley, California.

[3] Associate Professor, Department of Mathematics, University of Alberta, Edmonton, Alberta, Canada.

and 3). An algorithm for calculating $D_f(x)$ when f is faithfully convex has been proposed in Ref. 4.

Stability in convex programs is directly related to constraint qualifications. Programs for which no constraint qualification holds are not stable with respect to perturbations in the data. They can also be ill-posed and so discontinuous with respect to perturbations in the data. The correct calculation of $D_f(x)$ proves to be vital in solving these programs (see Ref. 3 for a discussion). However, the calculation of $D_f(x)$ is itself an ill-posed problem, i.e., it is discontinuous with respect to small perturbations in the data.

In this paper, we reformulate the problem of calculating $D_f(x)$ into the problem of calculating the cone of directions of almost constancy and derive a stable algorithm for the calculation. Moreover, this reformulation allows us to remove the restriction to faithfully convex functions needed in Ref. 4. More precisely, given an orthogonal $n \times p$ matrix A, we find orthogonal matrices P and Q such that f is almost constant on the convex hull of $\pm$ the columns of P, is sufficiently nonconstant on the column space of Q, and the column spaces of P and Q provide an orthogonal direct sum decomposition of the column space of A (see Theorem 2.1 and Corollary 2.1 for more precise statements). P and Q are found by postmultiplying A by a finite number of orthogonal matrices (Householder transformations). This guarantees the backward stability of the algorithm. The algorithm is given in Section 2, while backward stability is proved in Section 3.

The algorithm can be compared to finding the singular values of a matrix using Householder transformations. In fact, if f is a linear function [i.e., $f(x) = c^t x$, where the superscript t denotes transpose], then the algorithm finds the $n \times (p-1)$ matrix P and the $n \times 1$ matrix Q, such that the columns of P are the right singular vectors of $d = c^t A$ corresponding to the $p-1$ zero singular values, while Q is the right singular vector corresponding to the single nonzero singular value. If we choose our tolerance badly, then we might have P being $n \times p$; i.e., the nonzero singular value is considered to be numerically zero. Thus, we are making a numerical decision on the rank of d.

In a forthcoming study, we plan to apply this algorithm to the numerical decomposition of convex functions f: $R^n \to R^m$, which are convex with respect to some partial order on R^m. This numerical decomposition can be compared to the decomposition of a matrix using the singular value decomposition and associated singular vectors corresponding to nonzero and numerically zero singular values. Thus, we are making a numerical decision on the rank of the function f. For discussions of pseudo-rank or numerical rank of matrices, see, e.g., Refs. 5 and 6. The decomposition of this vector function f will be used to regularize convex programs.

2. Algorithm

We consider the differentiable convex function $f: R^n \to R$. The *cone of directions of constancy* of f at x is

$$D_f(x) = \{d \in R^n: \text{there exists } \bar{\alpha} > 0 \text{ such that } f(x+\alpha d) = f(x), \text{ for all } 0 < \alpha \leq \bar{\alpha}\}.$$

If f is differentiable, then $D_f(x)$ is a convex cone (e.g., Ref. 1). A convex function is called *faithfully convex* (see Ref. 7), if it is affine on a line segment only if it is affine on the whole line containing that segment. Analytic convex, as well as strictly convex, functions are examples. As in Ref. 4, the algorithm is based on the fact that $D_f(x) \subset \mathcal{N}(\nabla f(x))$, for all x, where $\mathcal{N}$ denotes null space and ∇ denotes gradient. In addition, if both

$$d^t \nabla f(0) < \epsilon \quad \text{and} \quad d^t \nabla f(d) < \epsilon,$$

then f is almost constant, in the direction d, at 0. Though we restrict ourselves to differentiable functions, the results can be easily extended to nondifferentiable convex functions by using subgradients. Moreover, this algorithm is not restricted to faithfully convex functions as was the case in Ref. 4.

Without loss of generality, we restrict ourselves to $D_f(0)$, which we denote by D_f. The algorithm does not find D_f exactly. Given the orthogonal (orthonormal columns) $n \times p$ matrix A, the algorithm decomposes the column space of A into two orthogonal complements, given by the range spaces of P and Q, such that the function f is almost constant on the convex hull of $\pm$ the columns of P and effects a sufficient nonconstant behavior on $R(Q)$, the range space of Q. This is accomplished by postmultiplying A by a finite number of orthogonal matrices A_i, found by deleting a column of a certain elementary Hermitian matrix (a Householder transformation). Thus, the error analysis is similar to that of Householder's method for finding the eigenvalues of a matrix, and the algorithm is backward stable; see Section 3.

We first present a lemma for the computation of an orthonormal basis for the null space of a nonzero vector.

Lemma 2.1. Suppose that $0 \neq d \in R^k$, $k \geq 2$, and that d_{i_0} is a component of d with largest absolute value. Let

$$u = d + \text{sign}(d_{i_0}) \|d\| e_{i_0},$$

where e_{i_0} is the i_0th unit vector in R^k. Form the $k \times (k-1)$ matrix A by deleting the i_0th column from the elementary Hermitian matrix

$$H = I_k - \theta u u^t, \tag{2}$$

where I_k is the $k \times k$ identity matrix, $\theta = 2/u^t u$, and the superscript t denotes transpose. Then,

$$A^t A = I_{k-1}, \qquad \mathcal{R}(A) = \mathcal{N}(d). \tag{3}$$

Proof. The matrix H is the Householder transformation (or elementary Hermitian) which transforms the vector $\|d\| e_{i_0}$ into the vector $-\text{sign}(d_{i_0}) d$. Thus, $HH^t = I_k$ and the i_0th column of H is $-(1/\|d\|)\, \text{sign}(d_{i_0}) d$. □

We use the component of largest absolute value to enhance stability and reduce roundoff error. In fact (see Ref. 8), the matrix H is continuous with respect to the data d if and only if we use a nonzero component of d. If d is not a negative multiple of e_{i_0}, we could have chosen

$$u = d - \text{sign}(d_{i_0}) \|d\| e_{i_0}.$$

The only arithmetic step involved in computing u is $d_{i_0} \pm \text{sign}(d_{i_0})\|d\|$. We choose + rather than − to guarantee a low relative error, no cancellation error. We could more simply use $i_0 = 1$, regardless of the sign of magnitude, and then set

$$u = d + \|d\| e_1,$$

provided d is not a negative multiple of e_1. Then, we would delete the first column of H. This has an added advantage in that we need never explicitly form H, but can leave it as $I_k - \theta u u^t$. For, it is much cheaper to perform matrix multiplications with H in this form. Since we will be post multiplying by a finite number of these H's, we can wait and delete the first column after performing all the multiplications. The formation of H with this rule is still stable (see Refs. 9 and 10).

Let the matrix $A \in R^{n \times p}$, with orthonormal columns, and let the scalars ϵ_0, $\epsilon_1 > 0$ be given.

Algorithm 2.1

Initialization. Set $P_0 = A$ and $i = 1$. With $x = 0$, if

$$\|P_0^t \nabla f(x)\| > \epsilon_0, \tag{4}$$

then denote $x_1 = x = 0$, set

$$q_1 = \frac{1}{\|P_0 P_0^t \nabla f(x_1)\|} P_0 P_0^t \nabla f(x_1),$$

and proceed to case (I); while, if (4) fails, then proceed to Step $i = 1$.

Step i. $i=1,\dots,s\le p$. Find a point x in the set of $2(p-i+1)$ vectors $\{\pm$ columns of $P_{i-1}\}$ such that

$$|x^t\nabla f(x)|=\max\{|y^t\nabla f(y)|: y=\pm \text{ a column of } P_{i-1}\}>\epsilon_1. \tag{5}$$

If the point x exists, denote it by x_i and set

$$q_i=\frac{1}{\|P_{i-1}P_{i-1}^t\nabla f(x_i)\|}P_{i-1}P_{i-1}^t\nabla f(x_i).$$

Case (I). If such an x exists and $i<p$, then, using Lemma 2.1, determine

$$A_i\in R^{(p-i+1)\times(p-i)}, \tag{6}$$

such that

$$\mathcal{R}(A_i)=\mathcal{N}(P_{i-1}^t\nabla f(x)). \tag{7}$$

Set

$$P_i=P_{i-1}A_i, \tag{8}$$

and proceed to Step $i+1$.

Case (II). If such an x exists, but $i=p$, then set $s=i$, $k=p-s=0$, $P_s=0$, and stop. In conclusion,

$$D_f^\epsilon\cap\mathcal{R}(A)=0. \tag{9}$$

Case (III). If such an x does not exist, then set $s=i-1$, $k=p-s$, and stop. In conclusion,

$$D_f^\epsilon\cap\mathcal{R}(A)=\mathcal{R}(P_{i-1}). \tag{10}$$

The above algorithm finds the numerical cone of directions of constancy with precision $\epsilon=(\epsilon_0,\epsilon_1)$. We make this precise in the following theorem and corollary. Numerical stability is proved in the next section.

Theorem 2.1. Let $p_1,\dots,p_k$ be the columns of P_s. The algorithm finds the p orthonormal direction vectors

$$p_1,\dots,p_k,q_1,\dots,q_s, \tag{11}$$

such that the span of the p_j and q_j equals $\mathcal{R}(A)$ and

$$|p_j^t\nabla f(0)|\le\epsilon_0,\qquad j=1,\dots,k, \tag{12}$$

$$|p_j^t\nabla f(\pm p_j)|\le\epsilon_1,\qquad j=1,\dots,k, \tag{13}$$

$$f((1+t)q_j)-f(0)\ge-\epsilon_0+t\epsilon_1,\qquad j=1,\dots,s,\ t\in R, \tag{14}$$

$$|q_j^t\nabla f(0)|\le\epsilon_0,\qquad j=2,\dots,s. \tag{15}$$

Proof. Let f_i denote the composite function

$$f_i(y) = f(P_i y).$$

When the algorithm stops,

$$f_s(y) = f(P_s y).$$

Note that

$$f_{i+1}(y) = f_i(A_{i+1} y),$$

and so f_{i+1} is the restriction of f_i to $\mathscr{R}(A_{i+1})$. Also,

$$\nabla f_i(y) = P_i^t \nabla f(P_i y). \tag{16}$$

Thus,

$$\nabla f(x_i) = \nabla f(\pm P_i e_j) = \nabla f_i(\pm e_j),$$

where e_j is the particular jth unit vector in R^{p-i+1} corresponding to the $x = x_i$ chosen at the ith step. At the ith step, the algorithm uses the function $f_{i-1}(y) = f(P_{i-1} y)$. q_i is then the normalized projection of $\nabla f(x_i)$ onto the subspace $\mathscr{R}(P_{i-1})$, while P_i is constructed so that $\mathscr{R}(P_i)$ is a proper subspace of $\mathscr{R}(P_{i-1})$ and is also a subspace of $\mathscr{N}(P_{i-1}^t \nabla f(x_i))$.

Now, if (4) fails, then, for $j = 0, \ldots, s$,

$$\begin{aligned} \|\nabla f_j(0)\| &= \|P_j^t \nabla f(0)\| \\ &= \|A_j^t \cdots A_1^t P_0^t \nabla f(0)\| \\ &= \|P_0^t \nabla f(0)\| \leq \epsilon_0. \end{aligned}$$

On the other hand, if (4) holds, then, for $j = 1, \ldots, s$,

$$\|\nabla f_j(0)\| = \|A_1^t P_0^t \nabla f(0)\| = 0, \tag{17}$$

by the choice of A_1 in (7). In either case, we have

$$\|\nabla f_j(0)\| = \|P_j^t \nabla f(0)\| \leq \epsilon_0, \qquad j = 1, \ldots, s. \tag{18}$$

This proves (13), since p_j is the jth column of P_s. Moreover, if

$$u_i = P_{i-1}^t q_i, \qquad i = 2, \ldots, s,$$

then $q_i = P_{i-1} u_i$ and

$$|q_i^t \nabla f(0)| \leq \|u_i\| \, \|P_{i-1}^t \nabla f(0)\| \leq \epsilon_0. \tag{19}$$

This prove (15). That (13) holds follows from the test at the final step.

It remains to prove (14). Suppose that $\|d\| \leq 1$ and $d = P_i u$, for some u and $i \geq 1$. Then,

$$\|u\| = \|P_i^t d\| \leq 1.$$

Thus,

$$|d^t\nabla f(0)| \leq \|P_i^t \nabla f(0)\| \leq \epsilon_0, \tag{20}$$

for any d of norm 1 in $\mathcal{R}(P_i)$, $i \geq 1$. By convexity of f, this implies that

$$f(x_i) - f(0) \geq -\epsilon_0, \qquad i = 1, \ldots, s. \tag{21}$$

Also, by convexity, we now conclude that

$$\begin{aligned} f((1+t)q_i) &\geq f(x_i) + \nabla f(x_i)^t((1+t)q_i - x_i) \\ &\geq f(0) - \epsilon_0 + (1+t)\frac{\nabla f(x_i)^t P_{i-1} P_{i-1}^t \nabla f(x_i)}{\|P_{i-1} P_{i-1}^t \nabla f(x_i)\|} - \nabla f(x_i)^t x_i \\ &= f(0) - \epsilon_0 + (1+t)\|P_{i-1} P_{i-1}^t \nabla f(P_{i-1} e_j)\| - \nabla f(P_{i-1} e_j)^t P_{i-1} e_j \\ &\geq f(0) - \epsilon_0 + t\epsilon_1, \end{aligned}$$

since

$$|\nabla f(P_{i-1} e_j)^t P_{i-1} e_j| \leq \|P_{i-1} P_{i-1}^t \nabla f(P_{i-1} e_j)\| \, \|P_{i-1} e_j\|$$

and

$$\|P_{i-1} P_{i-1}^t \nabla f(P_{i-1} e_j)\| > \epsilon_1,$$

by the test (5). □

The theorem provides an orthogonal direct sum decomposition of $\mathcal{R}(A)$ with respect to f. We can now see the behavior of f on $\mathcal{R}(P)$ and $\mathcal{R}(Q)$, where $P = P_s$ and $Q = [q_1, \ldots, q_s]$. Note that conv denotes convex hull.

Corollary 2.1. Let A, P, Q be as above, and construct $\bar{Q}$ by deleting the first column q_1 of Q. Let fA denote the composite function $fA(x) = f(Ax)$, and define fP, fQ, $f\bar{Q}$ similarly. For $d = (d_i) \in R^s$, let

$$\bar{d} = \max_i \{d_i\}.$$

The algorithm decomposes the composite function fA so that

$$\|\nabla fP(0)\| \leq \epsilon_0, \tag{22}$$

$$|fP(d) - fP(0)| \leq \max\{\epsilon_0, \epsilon_1\}, \qquad \text{for all } d \in \text{conv}\{\pm e_j\}_{j=1}^k \subset R^k, \tag{23}$$

$$fQ(td) - fQ(0) \geq -\epsilon_0 + (t\bar{d} - 1)\epsilon_1, \qquad \text{for all } d = (d_i) \in R^s,\ t \in R, \tag{24}$$

$$\|\nabla f\bar{Q}(0)\| \leq \epsilon_0. \tag{25}$$

Proof. The bounds on the gradients in (22) and (25) follow from (18) and (19), respectively, and correspond to (12) and (15). From (12), (13), and the convexity of f, we see that

$$|fP(d)-fP(0)|\leq \max\{\epsilon_0, \epsilon_1\}, \qquad \text{for all } d\in\{\pm e_j\}_{j=1}^k,$$

which yields (23). To prove (24), we use a similar argument to the proof of (14). We have

$$\begin{aligned} fQ(td) &\geq f(x_i)-\nabla f(x_i)'x_i+t\nabla f(x_i)'Qd \\ &\geq f(0)+\nabla f(0)'x_i-\nabla f(x_i)'x_i+t\nabla f(x_i)'Qd \\ &\geq f(0)-\epsilon_0+td_i\|P_{i-1}P_{i-1}^t\nabla f(x_i)\|-\nabla f(x_i)'x_i \\ &\geq fQ(0)-\epsilon_0+(td_i-1)\epsilon_1. \end{aligned} \tag{26}$$

□

Note that we are finding a subspace of the cone of almost directions of constancy. For example, if we set

$$f=\begin{cases} x^2, & \text{if } x\geq 0, \\ 0, & \text{otherwise}, \end{cases}$$

then we find that

$$D_f=\{0\}, \qquad Q=[1],$$

rather than

$$D_f=\{x: x\leq 0\}.$$

Note that (24) guarantees a nonconstant behavior for all d not in the negative orthant. Thus, f can be considered nonconstant on $\mathcal{R}(Q)$.

3. Error Analysis

Calculating the exact cone of directions of constancy of a convex function is an ill-posed problem. This can be seen easily when we consider the functions

$$f_\epsilon(x)=\epsilon x^2, \qquad \epsilon>0.$$

At $\epsilon=0$, $D_{f_\epsilon}=R$, while, for $\epsilon>0$, $D_{f_\epsilon}=0$. Instead of trying to find D_f exactly, the above algorithm finds the cone of directions of almost constancy. Reformulating the problem this way changes it to a well-posed problem if we choose our parameters ϵ_0, ϵ_1 correctly.

As in all algorithms, the accuracy of the solution is limited by the accuracy of the input data. Errors introduced by numerical calculations may be recast as a perturbation of the input data. If this perturbation is within the bounds of uncertainty of the input data, we say that the algorithm is backward stable (see, e.g., Refs. 10 and 11). In this section, we show that our algorithm is backward stable.

The input data is the $n \times p$ matrix A and the function $\nabla f(x)$ at a finite number of points. The solution is given as p vectors $p_1, \ldots, p_k, q_1, \ldots, q_s$, which we write as the matrix

$$S_s = [p_1, \ldots, p_k, q_s, \ldots, q_1] = [P_s \vdots Q_s]. \tag{27}$$

We let

$$S_0 = P_0,$$

$$S_i = [p_1, p_2, \ldots, p_{p-1} \vdots q_i, q_{i-1}, \ldots, q_1] = [P_i \vdots Q_i] \tag{28}$$

be, respectively, the initial and intermediate matrices. At the ith step, we define the transformation matrix[4]

$$H_i = \begin{bmatrix} I_{p-i+1} - \theta u u^t & \vdots & 0 \\ \cdots\cdots & & \cdots \\ 0 & \vdots & I_{i-1} \end{bmatrix}, \tag{29}$$

where θ and u are as in Lemma 2.1, with

$$d = P_{i-1}^t \nabla f(x_i).$$

The index j_i indicates the component of $P_{i-1}^t \nabla f(x_i)$ with maximum value, and e_{j_i} denotes the j_ith unit vector in R^{p-i+1}. Thus (see Lemma 2.1), H_i is the transformation obtained by moving the j_ith column of the Householder transformation which transforms e_{j_i} into $P_{i-1}^t q_i$, i.e.,

$$H_i = \begin{bmatrix} A_i & \vdots & P_{i-1}^t q_i & \vdots & 0 \\ \cdots & & \cdots & & \cdots \\ & 0 & & \vdots & I_{i-1} \end{bmatrix},$$

where A_i satisfies (7). The transformation yields $S_i = S_{i-1} H_i$, and so the solution satisfies

$$S_s = A J_1, \tag{30}$$

where

$$J_1 = H_1 H_2 \cdots H_s.$$

Suppose that ϵ is the unit roundoff of our arithmetic unit, and suppose that, due to roundoff, we find at Step i that

$$\tilde{H}_i = H_i + \tilde{F}_i.$$

[4] With the j_ith column moved to position $p-i$.

If our arithmetic unit is good, we can assume that

$$\|\tilde{F}_i\| \leq \epsilon; \tag{31}$$

see Ref. 10, p. 96. We can now recast the numerical error as a perturbation in A.

Theorem 3.1. Suppose that the $n \times p$ orthonormal matrix A is given and that the arithmetic unit is good, so that (31) holds. Then, the algorithm finds

$$\tilde{S}_s = (A + M_1)J_1, \tag{32}$$

where

$$\|M_1\| \leq sn\epsilon, \qquad J_1 = H_1 H_2 \cdots H_s.$$

Proof. The proof is similar to the argument in Ref. 10, p. 94. Due to roundoff error, we do not find H_i precisely, but instead we find

$$\tilde{H}_i = H_i + \tilde{F}_i. \tag{33}$$

By our assumption, (31) holds. We then find

$$\tilde{S}_i = \tilde{S}_{i-1}\tilde{H}_i + \bar{W}_i,$$

where $\bar{W}_i$ is the result of roundoff error in the product. This yields

$$\tilde{S}_i = \tilde{S}_{i-1}(H_i + \tilde{F}_i) + \bar{W}_i = \tilde{S}_{i-1}H_i + W_i,$$

where

$$W_i = \tilde{S}_{i-1}\tilde{F}_i + \bar{W}_i.$$

Then,

$$\begin{aligned}\tilde{S}_s &= \tilde{S}_{s-1}H_s + W_s\\ &= (\tilde{S}_{s-2}H_{s-1} + W_{s-1})H_s + W_s\\ &\quad \cdots\cdots\cdots\cdots\cdots\cdots\\ &= S_0 J_1 + \sum_{i=1}^{s} W_i J_{i+1},\end{aligned}$$

where $J_{s+1} = I$ and

$$J_i = H_i H_{i+1} \cdots H_s. \tag{34}$$

Simplifying, we have

$$\tilde{S}_s = (P_0 + M_1)J_1,$$

where

$$M_1 = \left[\sum_{i=1}^{s} W_i J_{i+1} \right] J_1^t \tag{35}$$

is called the *equivalent perturbation matrix* induced by the computation (see Ref. 10). The perturbation is bounded since

$$\|M_1\| \le \sum_{i=1}^{s} \|W_i\|, \tag{36a}$$

$$\|W_i\| \le \|\bar{W}_i\| + \|S_i\| \, \|\tilde{F}_i\|. \tag{36b}$$

Since the arithmetic unit is good, we may assume that

$$\|\tilde{F}_i\| \le \epsilon \qquad \text{(roundoff unit);}$$

and, since $\|H_i\| = 1$, we may assume that

$$\|\bar{W}_i\| \le n\epsilon \|\tilde{S}_i\|, \tag{37}$$

$$\|\tilde{S}_{i+1}\| \le (1+\epsilon)^3 \|\tilde{S}_i\|. \tag{38}$$

From (36a), this implies that the perturbation satisfies

$$\|M_1\| \le sn\epsilon \|A\| = sn\epsilon. \tag{39}$$

Hence, the backward error analysis tells us that, given the initial matrix $P_0 = A$, the algorithm calculates a solution matrix

$$\tilde{S}_s = [p_1, \ldots, p_k, q_s, \ldots, q_1] = (A + M_1) J_1,$$

where the columns of $\tilde{S}_s$ satisfy the conclusions in Theorem 3.1. J_1 is the orthogonal transformation matrix for the solution and M_1 is the equivalent perturbation matrix induced by the algorithm. By (39), we see that we have obtained the exact solution of a nearby problem. □

The tests (4) and (5) cannot be performed exactly all the time. One approach could be to choose ϵ_0, ϵ_1, but then modify it appropriately if the test (4) or (5) is ambiguous. Since there are only a finite number of tests performed, we would end up with appropriate scalars ϵ_0, ϵ_1 dependent on the specific function f.

If exact calculations were possible, we would find, with $\epsilon_0 = \epsilon_1 = 0$, that

$$\mathcal{R}(S_s) = \mathcal{R}(A), \tag{40}$$

$$\mathcal{R}(Q_s) = \mathcal{R}(A) \cap \operatorname{span}\{P_{i-1} P_{i-1}^t \nabla f(x_i)\}_{i=1}^{s}; \tag{41}$$

therefore, since D_f is orthogonal to $\nabla f(x)$ for all x,

$$\mathcal{R}(P_s) = \mathcal{R}(A) \cap D_f. \tag{42}$$

The error $\tilde{F}_i$ in (32) arises from calculating H_i using $\nabla f(x_i)$. We have seen that this is small; see (31). We can reflect this error back as well into an uncertainty in the data $\nabla f(x_i)$, rather than just reflecting the total error back into an uncertainty in A.

4. Example

We now illustrate the above algorithm with the following example. We assume three decimal places of accuracy and also that we have an exact accumulator. Thus, arithmetic between two numbers is done exactly and then is rounded to three decimals. We choose $\epsilon_0 = \epsilon_1 = 0.01$. The example uses the same functions as in Ref. 4.

Example 4.1. Consider the convex function

$$f(x) = -[4 + (x_1 + x_2)^2]^{1/2} + x_1 + x_2 + x_3^2.$$

Initialization. Set

$$P_0 = A_0 = I_3, \qquad i = 1.$$

Since

$$\nabla f(x) = \left(\left[1 - \frac{x_1 + x_2}{4 + (x_1 + x_2)^2} \right]^{1/2}, \left[1 - \frac{x_1 + x_2}{4 + (x_1 + x_2)^2} \right]^{1/2}, 2x_3 \right),$$

we see that

$$\|P_0' \nabla f(0)\| = \|(1, 1, 0)\| = \sqrt{2} > \epsilon_0.$$

Thus, (4) holds and we proceed to Step 1, Case (I).

Step 1. Using Lemma 2.1, with

$$d = \nabla f(0) = (1, 1, 0),$$

we get

$$P_1 = P_0 A_1 = \begin{bmatrix} -0.708 & 0 \\ 0.706 & 0 \\ 0.000 & 1 \end{bmatrix}.$$

Step 2. With $x = P_2 = (0, 0, 1)$ being the second column of P_1, we have

$$|x' \nabla f(x)| = 2 > \epsilon_1.$$

Thus, (5) holds and we are in Case (I) again. Lemma 2.1, with

$$d = P_1^t \nabla f(x) = (0, 2),$$

yields

$$A_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

and so

$$P_2 = P_1 A_2 = \begin{bmatrix} -0.708 \\ 0.706 \\ 0.000 \end{bmatrix}.$$

Step 3. With

$$x = \pm(-0.708, 0.706, 0.000),$$

we see that

$$|x^t \nabla f(x)| = 0.002 < \epsilon_1.$$

Therefore, Case (III) implies that

$$D_f^\epsilon = \mathscr{R}\left(\begin{bmatrix} -0.708 \\ 0.706 \\ 0.000 \end{bmatrix}\right).$$

Now, suppose that

$$g(x) = -x_1 - x_2 + x_3^2.$$

Let us find $D_f^\epsilon \cap D_g^\epsilon$. Note that both f and g are convex and analytic and so faithfully convex. This implies that D_f and D_g are subspaces independent of x.

Initialization. Set

$$P_0 = A_0 = \begin{bmatrix} -0.708 \\ 0.706 \\ 0.000 \end{bmatrix}.$$

Let

$$p_1^t = (-0.708, 0.706, 0.000).$$

Then,

$$\|P_0^t \nabla f(0)\| = \|\pm p_1^t \nabla f(\pm p_1^t\}\| = 0.002 < \epsilon_0 = \epsilon_1.$$

Therefore,

$$D_f^\epsilon \cap D_g^\epsilon = \mathcal{R}(P_o).$$

Note that the exact solution, as found in Ref. 4, is that the above holds with

$$P_0 = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}.$$

The critical accuracy was 0.002. In fact, if we choose our numerical zeros ϵ_0, $\epsilon_1 < 0.002$, then we would have obtained $\{0\}$ as the solution, rather than the one-dimensional subspaces $\mathcal{R}(P_0)$.

References

1. BEN-ISRAEL, A., BEN-TAL, A., and ZLOBEC, S., *Optimality in Nonlinear Programming: A Feasible Directions Approach*, Wiley, New York, New York, 1981.
2. WOLKOWICZ, H., *Geometry of Optimality Conditions and Constraint Qualifications: The Convex Case*, Mathematical Programming, Vol. 19, pp. 32–60, 1980.
3. WOLKOWICZ, H., *The Method of Reduction in Convex Programming*, Journal of Optimization Theory and Applications, Vol. 40, pp. 349–378, 1983.
4. WOLKOWICZ, H., *Calculating the Cone of Directions of Constancy*, Journal of Optimization Theory and Applications, Vol. 25, pp. 451–457, 1978.
5. BAART, M. L., *The Use of Autocorrelation for Pseudo-Rank Determination in Noisy Ill-Conditioned Linear Least-Squares Problems*, IMA Journal of Numerical Analysis, Vol. 2, pp. 241–247, 1982.
6. GOLUB, G., KLEMA, V., and STEWART, G. W., *Rank Degeneracy and Least Squares Problems*, Stanford University, Research Report, 1976.
7. ROCKAFELLAR, R. T., *Some Convex Programs Where Duals Are Linearly Constrained*, Nonlinear Programming, Edited by J. B. Rosen, O. L. Mangasarian, and K. Ritter, Academic Press, New York, New York, pp. 293–322, 1970.
8. COLEMAN, T. F., and SORENSEN, D. C., *A Note on the Computation of an Orthonormal Basis for the Null Space of a Matrix*, Cornell University, Department of Computer Science, Technical Report No. 82-510, 1982.
9. PARLETT, B. N., *Analysis of Algorithms for Reflections in Bisectors*, SIAM Review, Vol. 13, pp. 197–208, 1971.
10. PARLETT, B. N., *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, New Jersey, 1980.
11. WILKINSON, J. H., *Rounding Errors in Algebraic Processes*, Prentice-Hall, Englewood Cliffs, New Jersey, 1963.