# On Proving Existence of Feasible Points in Equality Constrained Optimization Problems

R. Baker Kearfott*
University of Southwestern Louisiana

October 21, 1996

**Abstract**

Various algorithms can compute approximate feasible points or approximate solutions to equality and bound constrained optimization problems. In exhaustive search algorithms for global optimizers and other contexts, it is of interest to construct bounds around such approximate feasible points, then to verify (computationally but rigorously) that an actual feasible point exists within these bounds. Hansen and others have proposed techniques for proving the existence of feasible points within given bounds, but practical implementations have not, to our knowledge, previously been described. Various alternatives are possible in such an implementation, and details must be carefully considered. Also, in addition to Hansen's technique for handling the underdetermined case, it is important to handle the overdetermined case, when the approximate feasible point corresponds to a point with many active bound constraints. The basic ideas, along with experimental results from an actual implementation, are summarized here.

**Key words.** constrained global optimization, verified computations, interval computations, bound constraints.

**AMS subject classifications.** 65K05, 90C26

*This work was supported in part by National Science Foundation grant CCR-9203730.

## 1 Introduction and Motivation

The context of our present study is the optimization problem

$$\boxed{\begin{array}{rrcll} \text{minimize} & \phi(X) & & & \\ \text{subject to} & c_i(X) & = & 0, & i = 1, \ldots, m \\ & a_{i_j} \leq & x_{i_j} & \leq b_{i_j}, & j = 1, \ldots, q, \end{array}} \tag{1}$$

where $X = (x_1, \ldots, x_n)^T$. A general constrained optimization problem, including inequality constraints $g(X) \leq 0$ can be put into this form by introducing slack variables $s + g(X) = 0$ along with the bound constraint $0 \leq s \leq \infty$. See [10] for a practical discussion of this.

The specific problem addressed here is:

$$\boxed{\begin{array}{l} \text{Given an approximate feasible point } \check{X} \in \mathbb{R}^n\text{, con-} \\ \text{struct bounds} \\ \quad \boldsymbol{X} = \left\{ (x_1, \ldots, x_n)^T \in \mathbb{R}^n \mid \check{x}_i - \epsilon_i \leq x_i \leq \check{x}_i + \epsilon_i \right\} \\ \text{such that there exists at least one feasible point of} \\ \text{Problem (1) in } \boldsymbol{X}\text{, i.e. a point } X \in \boldsymbol{X} \text{ with } C(X) = 0, \\ \text{such that } X \text{ also satisfies the bound constraints of} \\ \text{Problem (1).} \end{array}} \tag{2}$$

Problem (2) can be approached with *interval Newton methods,* which, at the most fundamental level, are computational versions of Brouwer's fixed point theorem. An introduction to interval Newton methods in the context of this paper is in [5]. Additional explanation in the present context will appear in [12]. A related, relaxed problem, i.e. that in which $\boldsymbol{X}$ are constructed or found such that every point $X \in \boldsymbol{X}$ has $\|C(X)\| \leq \epsilon$ for some $\epsilon$, has also been considered [17, 20]. Generally, if an accurate approximation to a solution (or feasible point) is known, interval Newton methods can prove that an actual solution exists within specified bounds, at a small fraction of the cost of the total cost of an exhaustive search algorithm that finds feasible points or solutions. The technique is thus of general utility.

In exhaustive search algorithms for global optimization, an upper bound $\overline{\phi}$ to the global minimum of the objective function $\phi$ is invaluable in eliminating regions over which the range of $\phi$ lies above $\overline{\phi}$. (See [6] or [2] for recent effective algorithms, as well as [17] or [5] for examples in unconstrained optimization.) A rigorous upper bound $\overline{\phi}$ can be obtained with interval arithmetic by evaluating $\phi$ over a small region $\boldsymbol{X}$ containing an approximate minimizer. However, in equality-constrained problems such as Problem 1, a rigorous upper bound is obtained with this process only if it is certain that $\boldsymbol{X}$ contains a feasible point.

Hansen proposes a technique in [5, §12.3 ff.] for determining whether a particular box (i.e. rectangular parallelepiped) $\boldsymbol{X}$ contains a feasible point. Hansen's technique involves selection of some of the variables to form a square subsystem of $C(X) = 0$. Variants of this technique are one of the alternatives summarized in this paper. (However, we recommend using it to verify *small* boxes around an approximate feasible point that has been found by a conventional floating-point method.) Additionally, a technique involving preconditioning and direct analysis of the rectangular constraint system $C(X) = 0$ is tried. A third alternative tried here is to use the Fritz John system to simultaneously verify feasibility of the constraints and existence of a critical point.

Optima in practical bound-constrained problems often occur on lower-dimensional boundaries, with many active bound constraints. In search algorithms for global constrained optima, local optimization software is first used to obtain an approximation to such an optimum, from which an upper bound $\overline{\phi}$ is to be constructed as above. However, use of an interval Newton method to prove that there exists a point at which the equality constraints hold requires the dimension of the space in which the point occurs be greater than or equal to the number of such constraints. In fact, verification that a box $\boldsymbol{X}$ constructed about an approximate feasible point contains a feasible point succeeds most often when the approximate feasible point is accurately near a feasible point, and when the approximation is at the center of the box (coordinate bounds) used for verification. For this reason, approximate feasible points obtained from conventional floating point optimizers sometimes should first be perturbed away from bound constraints, then adjusted. Two schemes for this are summarized.

The next section gives notation, while §3 summarizes the algorithm variants and alternatives for replacing the underdetermined system by a square system. Techniques for perturbing approximate optima off bound constraints are summarized in §4. The problems used in the experiments are listed in §5. The actual experiments are reported in §6. Overall conclusions can be found in §7. More complete details appear in [9].

## 2 Notation

Here, boldface will be used to denote intervals, lower case to denote scalar quantities, and upper case to denote vectors and matrices. Underscores will denote lower bounds of intervals and overscores will denote upper bounds of intervals. For components of vectors, corresponding lower case letters will be used. For example, we may have:

$$\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]^T,$$

where $\boldsymbol{x}_i = [\underline{x}_i, \overline{x}_i]$.

As above, $C(X) = (c_1(X), \ldots c_m(X))^T = 0$, $C : \mathbb{R}^n \to \mathbb{R}^m$, will denote the set of equality constraints, $\mathbf{C}(\boldsymbol{X})$ will denote the set of interval residuals of the equality constraints over $\boldsymbol{X}$, and $\boldsymbol{\nabla C}$ will denote either a componentwise interval extension of the Jacobi matrix of the constraints $C$ or a slope matrix corresponding to the constraints [16, 12].

Interval arithmetic will not be reviewed here. The reader may consult any of the numerous introductions, such as those in [1], [15], [16], [17], or [12].

## 3 A Summary of the Techniques

The techniques for proving existence of a feasible point depend on an *interval Newton method*. Interval Newton methods operate on nonlinear systems of equations $F(X) = 0$, $F : \mathbb{R}^n \to \mathbb{R}^n$. Most interval Newton methods can be put into the general form

$$\boldsymbol{N}(F; \boldsymbol{X}, \check{X}) = \check{X} + \boldsymbol{V}, \tag{3}$$

where $\boldsymbol{V}$ is an interval vector that bounds the exact solution set to the square interval linear system

$$\boldsymbol{A}(X - \check{X}) = -F(\check{X}), \tag{4}$$

where $\boldsymbol{A}$ is either an interval extension to the Jacobi matrix or a slope matrix [5, 12, 16]. For example, $\boldsymbol{V}$ can be obtained with preconditioned interval Gaussian elimination or with the preconditioned interval Gauss–Seidel method. (See [1, 16], and see [5, 12] for details in the present context.)

For understanding in this paper, an interval Newton method may be viewed abstractly as an operator on boxes $\boldsymbol{X} \subset \mathbb{R}^n$, such that $\boldsymbol{N}(F; \boldsymbol{X}, \check{X}) \subseteq$

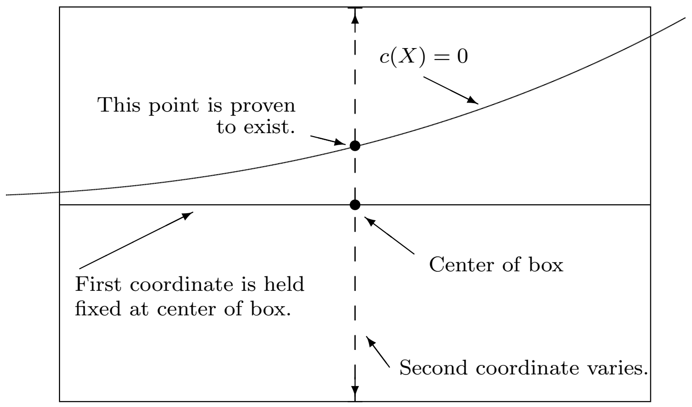
$c(X) = 0$
This point is proven to exist.
First coordinate is held fixed at center of box.
Center of box
Second coordinate varies.

Figure 1: Proving that there exists a feasible point of an underdetermined constraint system

$\boldsymbol{X}$ implies that there exists a solution of $F(X) = 0$ in $\boldsymbol{N}(F; \boldsymbol{X}, \check{X}) \subseteq \boldsymbol{X}$. Thus, to prove that there exists a feasible point, i.e. a point $X$ with $C(X) = 0$, in some box $\boldsymbol{X}$, it is necessary to construct a square system $F(X) = 0$ upon which to apply the interval Newton method.

However, systems of equality constraints $C(X) = 0$ $C : \mathbb{R}^n \to \mathbb{R}^m$ are typically underdetermined, with $m < n$. In [5, §12.3 ff.], Hansen suggests holding $(n - m)$ of the coordinates fixed. The basic idea is to choose those coordinates to which the system is least sensitive to be held fixed. For example, if $n = 2$ and there is just one constraint (i.e. $m = 1$), then $C(X) = 0$ is a curve in $\mathbb{R}^2$, and an interval Newton method can prove the existence of a feasible point along a line parallel to one of the coordinate directions; see Figure 1. Heuristics for choosing which coordinates to hold fixed can depend on the interval constraint matrix $\boldsymbol{\nabla C}(\boldsymbol{X})$. Hansen's heuristic for the choice of coordinates to be held fixed depends on Gaussian elimination with complete pivoting:

**Method 1** (Hansen's technique for choosing the fixed coordinates)

1. *Compute the midpoint matrix $A \in \mathbb{R}^{m \times n}$ of $\boldsymbol{\nabla C}(\boldsymbol{X})$.*
2. *Perform Gaussian elimination with complete pivoting on the rectangular matrix $A$.*
3. *Choose the original indices of the columns of $A$ that have been permuted into the last $n - m$ columns during the elimination process to be the indices of those variables to be held fixed (i.e. to be replaced by points) in the interval Newton method.*

There are a number of possible alternative schemes. One is to try to choose the coordinate directions to be held fixed to be nearly orthogonal to the null space of the Jacobi matrix of $C$ at a point:

**Method 2** (Alternate method of choosing the fixed coordinates)

1. *Compute an approximate basis* $V^{(1)}$, $V^{(2)}$, $V^{(i)} = (v_1^{(i)}, \ldots v_n^{(i)})^T$, $V^{(n-m)}$, *for the null space to* $\nabla C(\check{X})$, *where* $\check{X}$ *is a floating point approximation to a feasible point.*
2. *Order the* $n$ *coordinate directions* $j$, $j = 1, \ldots, n$ *in order of increasing* $\sum_{i=1}^{n-m} |v_j^{(i)}|$.
3. *Choose the last* $(n-m)$ *coordinate directions from Step 2 (in a sense, those coordinate directions most nearly tangent to the null space of the Jacobi matrix of the constraints) to be held fixed in the interval Newton method.*

It is also possible to use preconditioning to *implicitly* choose the square subspace in which to apply the interval Newton method. In particular, if the $m$ by $n$ interval linear system

$$\boldsymbol{\nabla C}(\boldsymbol{X})(X - \check{X}) = -C(\check{X})$$

is preconditioned by a point $m$ by $m$ matrix $Y$, the resulting system

$$Y\boldsymbol{\nabla C}(\boldsymbol{X})(X - \check{X}) = -YC(\check{X}), \tag{5}$$

is a square $m \times m$ interval linear system that corresponds to Equation (4). In fact, techniques from [7, 10, 12] can be used to produce a larger preconditioner $Y = (Y_1, \ldots, Y_n)^T$, where $Y_i$ is the $i$-th row of $Y$, from which $m$ rows as in Equation 5 can sometimes be selected. The preconditioner rows as in [7, 10, 12] are such that the width of the $i$-th coordinate of each coordinate image $\tilde{\boldsymbol{x}}_i$, where

$$\tilde{\boldsymbol{X}} = (\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_n) = \boldsymbol{N}(YC; \boldsymbol{X}, \check{X}),$$

is minimal over all possible preconditioner rows $Y_i$, when the interval Gauss–Seidel method is used to bound the solution set. (These preconditioners are called optimal LP preconditioners, since they can be computed approximately by approximately solving linear programming (LP) problems with floating point arithmetic.) If $\tilde{\boldsymbol{x}}_i \subseteq \boldsymbol{x}_i$ for $m$ coordinates $\{i_1, \ldots, i_m\}$, then, for each choice of the $x_j \in \boldsymbol{x}_j$, $j \notin \{i_1, \ldots, i_m\}$, there is a unique solution to $C(X) = 0$ within $\boldsymbol{X}$. (For details, see [12].) This leads to the following:

**Method 3** (Using preconditioners as in [7, 10, 12] to prove existence of a feasible point in $\boldsymbol{X}$)

1. *Compute the preconditioner row $Y_i$ and the interval Gauss–Seidel image $\tilde{\boldsymbol{x}}_i$, as described above, one row at a time, for $i = 1, \ldots, n$.*
2. *If there is a set $\{i_1, \ldots, i_m\}$ such that $\tilde{\boldsymbol{x}}_{i_k} \subseteq \boldsymbol{x}_{i_k}$ for $k = 1, dots, m$, then there exists a feasible point of $C(X) = 0$ in $\boldsymbol{X}$, for each selection of coordinate values of the coordinates other than $x_{i_1}, \ldots, x_{i_m}$.*

A possible practical advantage of methods 1 and 2 over Method 3 is that some of the coordinates are set to points, which could make the widths smaller in the interval Newton image, thus making it more likely to prove the existence of feasible points. On the other hand, the heuristic choices in methods 1 and 2 could could give inappropriate coordinates for proving the existence of a feasible point.

A final aspect of the process is construction of the box $\boldsymbol{X}$ within which a feasible point is to be proven to exist. An initial box can be constructed, followed by $\epsilon$-*inflation* as explained in [14, 18, 19]. In $\epsilon$-inflation a small box $\boldsymbol{X}$ is initially constructed, centered at an approximate feasible point $\check{X}$, and an interval Newton method attempts to prove existence of a feasible point within $\boldsymbol{X}$. If existence cannot be proven, one or more coordinate directions of $\boldsymbol{X}$ are made wider, then the interval Newton method is repeated. Eventual success of this process depends on how close $\check{X}$ actually is to a feasible point. Also, the size of the initial box $\boldsymbol{X}$ should be related to the accuracy to which $\check{X}$ has been computed. For details, see [9, 12].

## 4 On Bound Constraints

In bound-constrained problems, it is often the rule, rather than the exception, that bound constraints are active at the optimizers. This is also true of inequality-constrained problems, which for simplicity and other reasons, we convert to equality- and bound-constrained problems; cf. [10]. An extreme case of many active bound constraints is linear programming, in which a maximal number of bound constraints must be active.

Consequentially, if the box $\boldsymbol{X}$ is to be constructed about the approximate feasible point $\check{X}$ and lying within the region defined by the bound constraints, then $\check{X}$ must be near the boundary of $\boldsymbol{X}$. When that happens, the feasible manifold (such as the curve $c(X) = 0$ in Figure 1) can lie near a corner of $\boldsymbol{X}$, and the techniques summarized in §3 may fail to prove that a feasible point exists in $\boldsymbol{X}$. (The image of the interval Newton method tends to be centered at a solution of $F(X) = 0$, and if the solution is near the boundary of $\boldsymbol{X}$, the image $\boldsymbol{N}(F; \boldsymbol{X}, \check{X})$ may overlap with points outside $\boldsymbol{X}$.) If there are sufficient degrees of freedom left when the variables corresponding to bound-constraints are held fixed, then the interval Newton method can be applied in a subspace. Otherwise, (if $\check{X}$ is within a certain tolerance of too many bound constraints) $\check{X}$ can be perturbed into the interior of the bounds, then the resulting point can somehow be projected back onto the manifold $C(X) = 0$ (e.g. by using local optimization software in a hyperplane parallel to the bound constraints). This is done in the experiments below. For additional explanation of the procedure, see [9, 12].

Instead of trying to identify active bound constraints and adjust the number of degrees of freedom, it is possible to include the bound constraints in the Fritz John conditions [5, 10, 12], i.e. as part of a generalized Lagrange function, and to apply an interval Newton method to the gradient of the generalized Lagrange function. However, an interval Jacobi matrix or an interval slope matrix for the system of equations corresponding to the gradient of the Fritz John conditions often contains singular matrices when there are many active bound constraints. Thus, the practicality of this alternative is not certain.

## 5 The Test Set

Most of the problems in the test set are taken from [4]. We selected these to be non-trivial problems with a variety of constraint types, as well as differing numbers of variables and constraints. We also tried the problems from [20]. Although the latter are relatively simple, [20] contains one of the few published experimental results for general interval constrained optimization algorithms. Also, inclusion of these problems allows contrasting the relative ease of *verifying*, as done in this paper, with global search algorithms such as that of [20].

Each problem is identified with a mnemonic, given below.

Basic attributes of the test problems appear in Table 1. In each problem, each non-trivial inequality constraint in the original formulation was replaced by an equality constraint and a bound constraint on a slack variable. The numbers of variables in the table reflect these added slack variables; the bound constraints reflect original bound constraints and lower bounds on the added slack variables. The column labelled "# active" gives the number of

Table 1: Summary attributes of the test problems

| Problem name | # vars. | # equality constr. | # bound constr. | # active | pert? | type of constraints |
|---|---|---|---|---|---|---|
| fpnlp3 | 6 | 3 | 8 | 3 | no | linear |
| fpqp3 | 23 | 9 | 32 | 14 | yes | linear |
| fplnp6 | 4 | 2 | 6 | 2 | no | degree 4 |
| fppb1 | 9 | 6 | 13 | 7 | yes | bilinear |
| fphe1 | 16 | 13 | 28 | 10 | yes | bilinear |
| gould | 4 | 2 | 4 | 2 | no | quadratic |
| bracken | 3 | 2 | 1 | 1 | no | quadratic |
| wolfe3 | 3 | 2 | 2 | 0 | no | quadratic |

active bound constraints at the solution near which a feasible point is to be proven to exist. The column labelled "pert?" indicates whether perturbation (as mentioned in §4) is required to obtain a square system for the interval Newton method. (Such perturbation is required if the number of equality constraints exceeds the number of variables minus the number of active bound constraints.)

Each problem is identified below, along with the coordinates of the solution near which a feasible point is to be proven to exist. Additional details can be found in [9].

**fpnlp3** is the third nonlinear programming test problem, [4, p. 28]. The approximate solution of interest is

$$(\frac{4}{3}, 4, 0, 0, \frac{8}{3}, 0).$$

**fpqp3** is the third quadratic programming test problem, [4, p. 8]. The approximate solution of interest is

$$(1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1, 0, 0, 0, 5, 5, 0, 6, 6).$$

**fpnlp6** is the sixth nonlinear programming test problem, [4, p. 30]. The approximate solution of interest is

$$(2.3295, 3.1783, 0, 0).$$

**fppb1** is the first pooling-blending test problem, [4, p. 59]. The approximate solution of interest is

$$(0, 200, 0, 100, 0, 100, 0, 100, 1, 0, 0).$$

**fphe1** is the first heat exchanger network test problem, [4, pp. 63–66]. The approximate solution of interest is

$$(0, 10, 10, 10, 10, 0, 210, 150, 310, 210, 10, 0, 190, 140, 40, 50).$$

**gould** is the first test problem in [20]. The approximate solution of interest is

$$(14.095, .842960788, 0, 0).$$

**bracken** is the second test problem in [20]. The approximate solution of interest is

$$(0.822875653899075, 0.911437827385507, 0).$$

**wolfe3** is the third test problem in [20]. The approximate solution of interest is

$$(1.2247448866122757, 1.2247448567842063, 1.7320508069462872).$$

Although this is not an extensive set, these problems seem representative, particularly from the point of view of varying numbers of active bound constraints and parameter space dimensions. Later, it would be of interest to try the techniques on problems with highly nonlinear, transcendental constraints. Appearance of transcendental functions *per se* should not affect verification, but complicated expressions conceivably could.

## 6 Experimental Results

### 6.1 Implementation and Environment

The methods in §3 were programmed in the Fortran 90 environment developed for that purpose and described in [11]. Similarly, the functions described in §5 were programmed using the same Fortran 90 system, and an internal symbolic representation of the objective function, constraints and gradient of the objective function was generated prior to execution of the numerical tests. In the actual tests, generic routines then interpreted this single internal representation to obtain both floating point and interval values and derivative matrices.

The Lancelot package routine "DAUGLG", described in [3], was used to obtain the approximate feasible points $\check{X}$. The objective functions were included when this routine was called, since a primary use of verified feasible points in global optimization algorithms is to obtain good upper bounds on the global minimum. For convenience, a generic interface to DAUGLG was used.

The NAG Fortran 90 compiler, version 2 was used on a Sparc 20. Execution times were measured using the routine "DSECND".

A good initial guess was handed to DAUGLG, which then corrected it. Afterwards, the active bound constraints were determined. If the number of inactive bound constraints was then less than the number of equality constraints, then the approximate solution was perturbed, as indicated in §4. After the perturbed feasible point was produced, each of Method 2, Method 3 and and Method 1 was tried. Success of verification, along with CPU times and other performance information, was recorded.

Additional implementation details appear in [9].

## 6.2 Output and conclusions

Table 2 lists the success of each of the three methods (column "verified"), as well as the number of times the box was expanded in the $\epsilon$-inflation procedure (column "ninfl") before either it was proven that the box contained a feasible point, or until there was failure, with "1" indicating no expansions. The columns of Table 2 labelled "null space" correspond to Method 2, the columns labelled "LP" correspond to Method 3, and the columns labelled "elimination" correspond to Method 1.

Failure to prove existence of a feasible point was always due either to failure to compute a preconditioner for the interval Newton method or due to $\boldsymbol{X}$ becoming larger than a preset size. Since Method 1 *always* succeeded without $\epsilon$-inflation (and since no method succeeded after one or more expansions), it is clear that failure was due to intrinsic properties of the method, and not to an inaccurate approximate solution before or after the perturbation process or a poor choice of heuristics in the $\epsilon$-inflation process.

A second conclusion from Table 2 is that choosing the coordinates to be held fixed by Gaussian elimination (advocated by Hansen) is the most reliable method.

Table 2: Verification success of the three schemes

| | null space | | LP | | elimination | |
|---|---|---|---|---|---|---|
| problem | verified | ninfl | verified | ninfl | verified | ninfl |
| fpnlp3 | yes | 1 | yes | 1 | yes | 1 |
| fpqp3 | – | 1 | – | 4 | yes | 1 |
| fpnlp6 | yes | 1 | yes | 1 | yes | 1 |
| fppb1 | yes | 1 | – | 10 | yes | 1 |
| fphe1 | – | 4 | yes | 1 | yes | 1 |
| gould | yes | 1 | yes | 1 | yes | 1 |
| bracken | yes | 1 | yes | 1 | yes | 1 |
| wolfe3 | yes | 1 | – | 7 | yes | 1 |

In Table 3, numbers of interval evaluations of the constraints and constraint gradients are given for each of the three methods and each of the problems. This can be used for comparison with other results and tasks, such as the global search algorithm in [20].

In Table 4, CPU times in seconds for the three verification algorithms are listed.

Tables 2, 3 and 4 indicate that the Hansen variant is both more reliable and less costly, although most of the difference in cost is attributable to the fact that the Hansen variant is more reliable, and failure to prove feasibility cost more (because of repeated but ineffective $\epsilon$-inflation steps). The smaller times in Table 4 are inaccurate, since 0.01 sec. is the smallest time unit that can be measured with "DSECND."

Execution times to perturb the approximate feasible point away from bound constraints are given in [9]. An astounding result is that it took far more effort to perturb the approximate feasible point than to verify that a feasible point existed, once the approximate feasible point was perturbed. Most of the CPU time in such perturbation steps was spent in the constrained optimizer DAUGLG. See [9] for details. Also, see [8, 12] for a discussion of efficiency in the Fortran 90 system [11].

The alternative to selecting square subsystems and perturbing approximate feasible points, using the Fritz John system, was unsuccessful in preliminary experiments, and is not reported in these tables.

Table 3: Interval constraint and constraint gradient evaluations

| | null space | | LP | | elimination | |
|---|---|---|---|---|---|---|
| problem | $C$ | $\boldsymbol{\nabla C}$ | $C$ | $\boldsymbol{\nabla C}$ | $C$ | $\boldsymbol{\nabla C}$ |
| fpnlp3 | 9 | 9 | 6 | 9 | 9 | 9 |
| fpqp3 | 27 | 396 | 72 | 792 | 27 | 396 |
| fpnlp6 | 6 | 4 | 4 | 4 | 6 | 4 |
| fppb1 | 18 | 84 | 120 | 420 | 18 | 84 |
| fphe1 | 156 | 832 | 26 | 208 | 39 | 416 |
| gould | 6 | 4 | 4 | 4 | 6 | 4 |
| bracken | 6 | 4 | 4 | 4 | 6 | 4 |
| wolfe3 | 6 | 12 | 28 | 42 | 6 | 12 |
| Totals | 234 | 1345 | 264 | 1483 | 117 | 929 |

Table 4: CPU times for the verification steps

| problem | null space | LP | elimination |
|---|---|---|---|
| fpnlp3 | 0.02 | 0.01 | 0.01 |
| fpqp3 | 0.05 | 0.43 | 0.05 |
| fpnlp6 | 0.01 | 0.01 | 0.01 |
| fppb1 | 0.03 | 0.26 | 0.03 |
| fphe1 | 0.33 | 0.20 | 0.11 |
| gould | 0.01 | 0.01 | 0.00 |
| bracken | 0.01 | 0.01 | 0.01 |
| wolfe3 | 0.02 | 0.07 | 0.01 |
| Totals: | 0.48 | 1.00 | 0.23 |

## 7 Summary

Several techniques for proving feasibility of a point in a neighborhood of an approximate solution of an optimization problem posed in terms of equality constraints and bound constraints have been outlined. These techniques have been tested with a small but significant set of test problems. The techniques appear to be reliable and inexpensive, relative to the local floating point optimizers used in conjunction with them. The technique should prove valuable in global search algorithms and other applications.

Although verifying existence of feasible points has been discussed in the literature, to our knowledge, there has been little previous thorough development and empirical evaluation of techniques. A notable exception is [13]. There, as large a box as possible was constructed within which inequalities of the form $g(x) < 0$ can be rigorously verified; the algorithm was applied to a significant engineering design problem (of composite laminates).

Other considerations, such as handling linearly dependent constraints, appear in [9, 12].

## 8 Acknowledgement

I wish to acknowledge Shiying Ning for programming and checking the more complicated test problems used here. I also wish to apologize to the referees and the associate editor for the rather unpolished first draft of the paper, and I wish to thank them for their work and patience.

## References

[1] G. Alefeld and J. Herzberger. *Introduction to Interval Computations.* Academic Press, New York, 1983.

[2] O. Caprani, B. Godthaab, and K. Madsen. Use of a real-valued local minimum in parallel interval global optimization. *Interval Computations*, 1993(2):71–82, 1993.

[3] A. R. Conn, N. Gould, and Ph.L. Toint. *LANCELOT: A Fortran Package for Large-Scale Nonlinear Optimization.* Series in Computational Mathematics, vol. 17. Springer-Verlag, New York, 1992.

[4] C. A. Floudas and P. M. Pardalos. *A Collection of Test Problems for Constrained Global Optimization Algorithms.* Lecture Notes in Computer Science no. 455. Springer-Verlag, New York, 1990.

[5] E. R. Hansen. *Global Optimization Using Interval Analysis.* Marcel Dekker, Inc., New York, 1992.

[6] C. Jansson and O. Knüppel. A global minimization method: The multidimensional case. Technical Report 92.1, Informathinstechnik, Technische Uni. Hamburg–Harburg, 1992.

[7] R. B. Kearfott. Preconditioners for the interval Gauss–Seidel method. *SIAM J. Numer. Anal.*, 27(3):804–822, June 1990.

[8] R. B. Kearfott. Empirical evaluation of innovations in interval branch and bound algorithms for nonlinear algebraic systems, 1994. Accepted for publication in *SIAM J. Sci. Comput.*

[9] R. B. Kearfott. On verifying feasibility in equality constrained optimization problems. Technical report, University of Southwestern Louisiana, 1994.

[10] R. B. Kearfott. A review of techniques in the verified solution of constrained global optimization problems, 1994. Preprint, Department of Mathematics, Univ. of Southwestern Louisiana, U.S.L. Box 4-1010, Lafayette, LA 70504.

[11] R. B. Kearfott. A Fortran 90 environment for research and prototyping of enclosure algorithms for nonlinear equations and global optimization. *ACM Trans. Math. Software*, 21(1):63–78, March 1995.

[12] R. B. Kearfott. *Rigorous Global Search Methods for Continuous Problems.* Kluwer, Dordrecht, Netherlands, 1996.

[13] B. P. Kristinsdottir, Z. B. Zabinsky, T. Csendes, and M. E. Tuttle. Methodologies for tolerance intervals. *Interval Computations*, 1993(3):133–147, 1993.

[14] G. Mayer. Epsilon–inflation in verification algorithms, 1993. Preprint, Fachbereich Mathematik, Universität Rostock, Postfach 6980, D-18051 Rostock, Germany.

[15] R. E. Moore. *Methods and Applications of Interval Analysis.* SIAM, Philadelphia, 1979.

[16] A. Neumaier. *Interval Methods for Systems of Equations.* Cambridge University Press, Cambridge, England, 1990.

[17] H. Ratschek and J. Rokne. *New Computer Methods for Global Optimization.* Wiley, New York, 1988.

[18] S. M. Rump. *Kleine Fehlerschranken bei Matrixproblemen.* PhD thesis, Universität Karlsruhe, 1980.

[19] S. M. Rump. Verification methods for dense and sparse systems of equations. In J. Herzberger, editor, *Topics in Validated Computations*, pages 63–135, Amsterdam, 1994. Elsevier Science Publishers.

[20] M. A. Wolfe. An interval algorithm for constrained global optimization. *J. Comput. Appl. Math.*, 50:605–612, 1994.