Mathematical Programming manuscript No.
(will be inserted by the editor)

R. Baker Kearfott

Improved and Simplified Validation of Feasible Points

Inequality and Equality Constrained Problems

Received: date / Revised version: date

Abstract. In validated branch and bound algorithms for global optimization, upper bounds on the global optimum are obtained by evaluating the objective at an approximate optimizer; the upper bounds are then used to eliminate subregions of the search space. For constrained optimization, in general, a small region must be constructed within which existence of a feasible point can be proven, and an upper bound on the objective over that region is obtained. We had previously proposed a perturbation technique for constructing such a region. In this work, we propose a much simplified and improved technique, based on an orthogonal decomposition of the normal space to the constraints. In purely inequality constrained problems, a point, rather than a region, can be used, and, for equality and inequality constrained problems, the region lies in a smaller-dimensional subspace, giving rise to sharper upper bounds. Numerical experiments on published test sets for global optimization provide evidence of the superiority of the new approach within our GlobSol environment.

1. Introduction

The overall problem being considered is

$$\begin{array}{l}\text{minimize } \varphi(x)\\ \text{subject to } c_i(x) = 0,\ i = 1,\dots,m_1,\\ \qquad\qquad g_i(x) \le 0,\ i = 1,\dots,m_2,\\ \text{where } \varphi : \boldsymbol{x} \to \mathbb{R} \text{ and } c_i, g_i : \boldsymbol{x} \to \mathbb{R}, \text{ and where } \boldsymbol{x} \subset \mathbb{R}^n \text{ is}\\ \text{the hyperrectangle (box) defined by}\\ \qquad\qquad \underline{x}_i \le x_i \le \overline{x}_i,\ 1 \le i \le n,\\ \text{where the } \underline{x}_i \text{ and } \overline{x}_i \text{ are constant bounds.}\end{array} \tag{1}$$

We will call this problem a *general nonlinear programming problem*, abbreviated "general NLP" or "NLP".

In deterministic algorithms for solving NLP, the search region (defined by the box $\boldsymbol{x}$) is recursively subdivided, and a branch and bound process is used to eliminate subregions that cannot contain the global optimum. A fundamental technique[1] in this branch and bound process is obtaining an upper bound $\overline{\varphi}$ for the global optimum by evaluating the objective φ at a feasible point, then

R. Baker Kearfott: Department of Mathematics, University of Louisiana, U.L. Box 4-1010, Lafayette, LA 70504-1010 USA

Mathematics Subject Classification (1991): 90C26, 90-08, 65G20

[1] This is not the only technique used in such algorithms, since "acceleration" procedures are necessary to make the algorithms practical.

eliminating subregions $\tilde{\boldsymbol{x}}$ by obtaining a lower bound $\underline{\varphi}(\tilde{\boldsymbol{x}})$ on the objective over $\tilde{\boldsymbol{x}}$, then eliminating $\tilde{\boldsymbol{x}}$ provided $\underline{\varphi}(\tilde{\boldsymbol{x}}) > \overline{\varphi}$.

In *validated* algorithms for solving an NLP, the algorithm returns bounds on all possible optimizers and on the global optimum, and roundoff error is taken into account in such a way that completion of the algorithm is a mathematical proof that the optimum and optimizers must be exactly within the bounds that are output. When there are inequality and equality constraints in the NLP, a valid algorithm cannot obtain the upper bound $\overline{\varphi}$ by merely evaluating the objective $\varphi(\check{x})$ at an approximate optimizing point[2], but φ must be evaluated either at a point that is proven to be feasible or over a small region, such as a box $\boldsymbol{x}$, $\check{x} \in \boldsymbol{x}$, and then computing a rigorous upper bound for φ over $\boldsymbol{x}$, such as can be done using interval arithmetic.

One is tempted to use the Kuhn–Tucker equations (or Fritz–John equations) in an interval Newton method to prove existence of a constrained critical point (and hence, of a feasible point) within a box $\boldsymbol{x}$ constructed about the approximate optimizer $\check{x}$. However, because of likely singularity of this system, this technique is likely to fail; see the discussion in [5, §2] to gain insight into the case where the NLP (1) is linear.

In [3], we proposed and provided test results for a technique for perturbing the approximate optimizing point $\check{x}$ and by constructing a box about the perturbed feasible point within which an interval Newton method can prove existence of feasible points. Since then, we have obtained significant additional experience with this technique within our GlobSol (see [4, §2], [6] and [1]) algorithmic environment. In particular, we have found that, for a significant number of problems, the interval Newton method fails to prove existence of a feasible point within $\boldsymbol{x}$, or the widths of the box $\boldsymbol{x}$ are such that the upper bound $\overline{\varphi}$ so obtained is not sharp enough to be as effective as it could be.

In this paper, we describe alternate techniques for validating feasibility. Not only are these techniques significantly simpler than that in [3], but they also are more reliable and gives sharper $\overline{\varphi}$. When there are only inequality and bound constraints, our new technique uses a QR factorization of the matrix of gradients of the active constraints to perturb $\check{x}$ to a point $\tilde{x}$ in the interior of the feasible region, such that an interval evaluation of the constraints at $\hat{x}$ can prove that $\hat{x}$ is feasible. In contrast, using the technique in [3] to handle inequality constraints, the active inequality constraints are treated as equality constraints, and an attempt is made to prove existence of a point at which all constraints are simultaneously active, a much stronger but unnecessary condition.

In the case of both inequality constraints and equality constraints, we perform a QR factorization of the matrix of both equality and inequality constraints, thus obtaining vectors orthogonal to the gradients of the equality constraints, in which we can perturb $\check{x}$ to obtain a point $\hat{x}$. We then apply an interval Newton method in a subspace of $\mathbb{R}^n$ parallel to the gradients of the equality constraints and in

[2] This is true even if roundoff is taken into account when φ is evaluated, such as if interval arithmetic is used and the upper bound of $\varphi(\check{x})$ is taken as the value.

the tangent space of the inequality constraints. Experimental results confirm that this technique is reliable and effective.

We present the basic technique for inequality-constrained problems in §2. We describe our experimental environment and test set in §3, we present experimental results with the basic technique for inequality-constrained problems in §4. We explain and analyze our technique for both equality and inequality constraints in §5, and we present experimental results in the presence of both types of constraints in §6. Conclusions appear in §7.

2. Perturbations for Inequality Constraints

Let $\check{x} \in \mathbb{R}^n$ be an approximately obtained optimizing point of the NLP (1), and assume that $m_1 = 0$ (that is, assume there are no inequality constraints). Suppose further that there are m_i approximately active inequality constraints at $\check{x}$, including possibly active bound constraints, with $m_i \leq n$, and denote these active inequality and bound constraints by $g_{i_j} \leq 0$, $1 \leq j \leq m_i$.

Note 1. Approximate (i.e. "floating point") optimizers typically return approximate dual variables or Lagrange multipliers for the inequality and bound constraints. The mathematically "usual" situation is that there are n or less such constraints active, and these constraints can be identified by non-zero values of the approximate Lagrange multipliers. However, it sometimes occurs[3] that more than n inequality constraints are simultaneously approximately active at $\check{x}$. If only n or less of the corresponding Lagrange multipliers are non-zero, we can still attempt to apply our techniques.

Let $A^{(\iota)} \in \mathbb{R}^{n \times m_i}$ be that matrix whose j-th column is the approximate gradient $\nabla g_{i_j}(\check{x})$, $1 \leq j \leq m_i$, and form a QR factorization

$$A^{(\iota)} = Q^{(\iota)} R^{(\iota)}. \tag{2}$$

We then use the following algorithm to produce a direction v at $\check{x}$ that is likely to point into the feasible region.

Algorithm 1 *(Producing a direction into the feasible region)*
INPUT: The matrix $Q^{(\iota)}$ as in Equation (2).
OUTPUT: A vector v likely to point from $\check{x}$ into the interior of the feasible region of the NLP (1).

1. IF $Q^{(\iota)}_{:,1} \circ A^{(\iota)}_{:,1} < 0$ THEN $Q^{(\iota)}_{:,1} \leftarrow -Q^{(\iota)}_{:,1}$.
2. $v_1 \leftarrow -Q^{(\iota)}_{:,1} sgn(R^{(\iota)}_{1,1})$
3. DO *for* $j = 2$ *to* m_i*:*
   (a) $\delta_k \leftarrow Q^{(\iota)}_{:,k} \circ A^{(\iota)}_{:,k}$.
   (b) IF $\delta_k < 0$ THEN
      i. $Q^{(\iota)}_{:,k} \leftarrow -Q^{(\iota)}_{:,k}$

[3] due, for example to redundancy in the model formulation

ii. $\delta_k \leftarrow -\delta_k$.
END IF
(c) $\nu_k \leftarrow v_{k-1} \circ A^{(\iota)}_{:,k}$.
(d) $\alpha_k \leftarrow 2\nu_k/\delta_k$.
(e) IF $\alpha_k > 0$ THEN
i. $u_k \leftarrow v_{k-1} - \alpha_k Q^{(\iota)}_{:,k}$.
ii. $v_k \leftarrow u_k/\|u_k\|_2$.
ELSE
$v_k \leftarrow v_{k-1}$.
END IF
END DO
4. $v \leftarrow v_{m_i}$.

Note 2. The vectors u_k and v_k need not be stored separately, but may be accumulated in a single vector v.

Note 3. The factor "2" in Step 3d of Algorithm 1 can, in principle, be any number greater than 1, as we shall see in the proof of the following theorem.

Theorem 1. *Suppose the NLP (1) contains only equality constraints (i.e. $m_1 = 0$), and suppose that the bound constraints are included in the m_2 inequality constraints. Suppose that $\check{x} \in \mathbb{R}^n$ is a point at which $m_i \leq n$ constraints $g_{i_j} \leq 0$, $1 \leq j \leq m_i$ are exactly active, and no other constraints are active at $\check{x}$. Suppose all constraints (both active and inactive) are continuous at $\check{x}$, and suppose the constraint functions g_{i_j} are differentiable at $\check{x}$. Suppose $\nabla g_{i_j}(\check{x}) \neq 0$, let $A^{(\iota)} \in \mathbb{R}^{n \times m_i}$ be the matrix whose j-th column is exactly the gradient $\nabla g_{i_j}(\check{x})$, $1 \leq j \leq m_i$, and suppose v is produced by Algorithm 1. Then, for all sufficiently small ϵ, $\hat{x} = \check{x} + \epsilon v$ is strictly feasible with respect to the constraints of NLP.*

Proof. The proof will proceed by induction on the number m_i of active inequality constraints. If $m_i = 1$, then

$$v = -\nabla g_{i_1}(\check{x})/\|\nabla g_{i_1}(\check{x})\|$$

by construction. Thus, since g_{i_1} is differentiable and with nonzero gradient at $\check{x}$, there is a sufficiently small ϵ_1 such that for all $\epsilon > 0$, $\epsilon < \epsilon_1$, $g_{i_1}(\check{x} + \epsilon v) < 0$. Now, denote an arbitrary remaining inequality constraint or bound constraint for the NLP by $\tilde{g}(x) \leq 0$. Since $\tilde{g}(\check{x}) < 0$, there is an $\tilde{\epsilon}$ such that $\epsilon < \tilde{\epsilon}$ implies $\tilde{g}(\check{x} + \epsilon v) < 0$. Thus, if ϵ is less than the minimum of ϵ_1 and all of the $\tilde{\epsilon}$, then all of the constraints of the NLP are strictly feasible at $\check{x} + \epsilon v$.

Now assume that Algorithm 1 produces a vector v such that $v \circ \nabla g_{i_j}(\check{x}) < 0$ for each active inequality constraint $g_{i_j}(\check{x}) = 0$ when the number of active inequality constraints is $m_i = k$; suppose the NLP has $m_i = k + 1$ active inequality constraints $\left\{g_{i_j}\right\}_{j=1}^{m_i}$ at $\check{x}$, and suppose Algorithm 1 produces v_{k+1} for NLP. Then, by construction,

$$u_{k+1} = v_k - 2\frac{\min\left\{v_k \circ A^{(\iota)}_{k+1}, 0\right\}}{Q^{(\iota)}_{k+1} \circ A^{(\iota)}_{k+1}} Q^{(\iota)}_{k+1},$$

where $Q^{(\iota)}_{k+1}$ has possibly been negated, so that $Q^{(\iota)}_{k+1} \circ A^{(\iota)}_{k+1} > 0$, and $v_{k+1} = \beta u_{k+1}$ for some $\beta > 0$. Furthermore:

1. $\nabla g_{i_j}(\check{x})$ for $j \le k$ is in the column space of the first k columns of $A^{(\iota)}$, and hence is orthogonal to $Q^{(\iota)}_{k+1}$;
2. v_k can be thought of as being formed from the relaxed problem obtained from the NLP (1) by removing the constraint $g_{i_{k+1}}(x) \le 0$. Thus, from the induction hypothesis, $v_k \circ \nabla g_{i_j}(\check{x}) < 0$ for $1 \le j \le k$.

To complete the induction step (namely, to show that $v_{k+1} \circ \nabla g_{i_j}(\check{x}) < 0$ for $1 \le j \le k+1$), we thus need only show that $v_{k+1} \circ \nabla g_{i_{k+1}}(\check{x}) < 0$. This is equivalent to showing $u_{k+1} \circ \nabla g_{i_{k+1}}(\check{x}) < 0$. If $v_k \circ A^{(\iota)}_{k+1} \le 0$, then $v_{k+1} = v_k$ by construction, and the induction step is complete, so assume $v_k \circ A^{(\iota)}_{k+1} > 0$. Then:

$$\begin{aligned} u_{k+1} \circ \nabla g_{i_{k+1}}(\check{x}) &= v_k \circ \nabla g_{i_{k+1}}(\check{x}) - 2\frac{v_k \circ A^{(\iota)}_{k+1}}{Q^{(\iota)}_{k+1} \circ A^{(\iota)}_{k+1}} \left(Q^{(\iota)}_{k+1} \circ \nabla g_{i_{k+1}}(\check{x})\right) \\ &= v_k \circ \nabla g_{i_{k+1}}(\check{x}) - 2v_k \circ \nabla g_{i_{k+1}}(\check{x}) \\ &= -v_k \circ \nabla g_{i_{k+1}}(\check{x}) \\ &< 0, \end{aligned}$$

thus completing the induction step.

Now, choose ϵ_1 sufficiently small so that $g_{i_j}(\check{x}+\epsilon v_{k+1}) = \epsilon \nabla \boldsymbol{g}_{i_j}(\check{x}) \circ v_{k+1} + \mathrm{o}(\epsilon)$ leads to $g_{i_j}(\check{x} + \epsilon v_{k+1}) < 0$ for each j, $1 \le j \le m_i = k+1$ and every $\epsilon < \epsilon_1$. Now, as in the case of $m_i = 1$, choose $\tilde{\epsilon}$ such that $\epsilon < \tilde{\epsilon}$ implies $\tilde{g}(\check{x} + \epsilon v) < 0$ for each inactive constraint $\tilde{g}(\check{x}) < 0$. To complete the proof, choose $\epsilon < \min\{\epsilon_1, \tilde{\epsilon}\}$. □

Note 4. If $\check{x}$ is only approximately feasible with respect to the constraints of the NLP, but the approximation is sufficiently good that $\check{x}$ is much closer to a point where the m_i constraints are active than to points where the inactive constraints become active, then there is a range $[\underline{\epsilon}, \overline{\epsilon}]$ such that, if $\epsilon \in [\underline{\epsilon}, \overline{\epsilon}]$, then $\hat{x} = \check{x} + \epsilon v$ is strictly feasible with respect to the constraints of NLP.

To use Algorithm 1, we use the following algorithm.

Algorithm 2 *(Finding and validating an approximate optimum of the NLP (1), when only inequality constraints exist)*
INPUT: A tolerance ϵ_d
OUTPUT: Either a point $\hat{x}$ that has been proven to be feasible and that is likely to be near an optimizing point of the NLP (1), or a "failure" message

1. *Use a high-quality floating point approximate optimizer to compute a point $\check{x}$ that is an approximate local optimizer of the NLP(1). Supply a tolerance of $\epsilon_d/100$ to the approximate optimizer so its heuristic that determines the accuracy of the solution likely will result in an approximate feasible point that is significantly closer than ϵ_d to a Kuhn–Tucker point.*

2. *Determine the active inequality and bound constraints to be those corresponding to non-zero dual variables as returned*[4] *by the approximate optimizer.*
3. IF *the number of constraints identified as active exceeds* n, THEN *exit, signalling failure.*
4. *Compute a QR factorization as in Equation (2) of the matrix* $A^{(\iota)}$ *whose columns are the gradients of the constraints identified as active.*
5. *Use the QR factorization from Step 4 to execute Algorithm 1 to produce a vector* v *likely to point into the feasible region at* $\check{x}$.
6. *Compute* $\hat{x} = \check{x} + \epsilon_d v$.
7. *Using interval arithmetic to bound rounding errors, compute* $g_i(\hat{x})$ *for every inequality and bound constraint* $g_i(x) \leq 0$ *of the NLP (1).*
   IF *the upper bound of each* g_i *is non-positive* THEN
   Return with $\hat{x}$ *as the validated feasible point for the NLP 1.*
   ELSE
   Return, indicating "failure".
   END IF

3. Experimental Environment and Test Problems

We performed our experiments within the GlobSol environment, on a dual 3.2GHz Pentium-4 based machine[5] running Windows XP. We used an experimental version of GlobSol's algorithm that uses linear relaxations, essentially[6] as we presented in [4]. In that algorithm (essentially Algorithm 2 in [4] with an LP filtering step inserted between constraint propagation (step (iii)) and the interval Newton method (step (iv))), the feasible point verification scheme from [3] was attempted

1. within the initial box, before the start of Algorithm 2 of [4],
2. when the LP solver produced an optimizing point[7] to the linear relaxation[8], such that a validated upper bound on the optimum is smaller than any previously computed validated upper bound,
3. with a starting point within each box that is "sufficiently small" (i.e. step (v)(a) of Algorithm 2 in [4]), and
4. optionally, after the interval Newton step and before bisection (i.e. immediately prior to step (vi) of Algorithm 2 in [4]), with starting point equal to the midpoint of the current box.

[4] A tolerance τ is used, such that a Lagrange multiplier λ is deemed to be non-zero provided $|\lambda| < \tau$.

[5] The machine had 2 gigabytes of memory, but this was irrelevant, since at no time did GlobSol require more than several megabytes when running the reported experiments.

[6] In addition to the modifications to include our simplified improved validation of feasible points, another possibly significant modification over the algorithm in [4] was ordering the coordinates in the "complementation" process we presented in [2, §4.3.1] from widest to narrowest.

[7] to be used as a starting point for the approximate optimizer

[8] This step occurs between steps (iii) and (iv) of Algorithm 2 in [4].

In our experiments, we inserted our simplified scheme, i.e. Algorithm 2 above, both after successful computation of an upper bound from an LP relaxation and immediately before bisection. In the experiments, we kept the scheme from [3], to be tried only if Algorithm 2 above failed. For comparison, we ran GlobSol twice on all problems in the test sets, with identical parameters and configuration, except in the first set of runs, we did not use Algorithm 2 whereas we did in the second set of runs.

As an initial test, we used the problem designated as "OET5" in [10] and which we formulated as Equation 4 in [4] using Lemaréchal's conditions. Although approximate optimizers easily find Kuhn–Tucker points for this problem and although the BARON global optimization software [8] readily produces an optimizing point, our GlobSol software has found it challenging to validate that all solutions have been found, and part of the difficulty has been in validation of feasible points.

For a more systematic test, we used those problems from the "Tiny 1" Neumaier–Shcherbina test set [7] for which we both had correct Fortran 90 input files[9] and such that we had already implemented rigorous linear relaxations in GlobSol for the standard functions that occurred. For testing Algorithm 2 above, we used only those problems that did not contain equality constraints. We also eliminated several problems in which there appeared to be issues with operation exceptions in our installation of the floating point constrained optimization routine[10]. We also eliminated all of the unconstrained problems.

In each case we attempted to run the branch and bound algorithm (in GlobSol) to completion, but we allowed at most 10,000 seconds of processor time, except for OET5, where we allowed 25,000 seconds of processor time.

4. Experimental Results: Only Inequality-Constrained Problems

The following abbreviations are used in the tables.

OK? indicates whether the branch and bound algorithm successfully validated all global optimizing points (Y) or not (N) within 10,000 processor seconds.

N_{box} is the total number of sub-boxes processed in the branch and bound process.

T_{tot} is the total processor time, in seconds used in the branch and bound algorithm.

N_{better} is the number of times a smaller upper bound on the global optimum was obtained from a feasible point obtained with our simplified scheme (Algorithm 2).

$N_{\text{too many}}$ is the number of times there were too many active inequality constraints, as identified in step 3 of Algorithm 2

N_{v} is the number of times a perturbed point $\hat{x}$ from Algorithm 2 was proven to be feasible.

[9] There were some difficulties with the conversion process from AMPL format, for some files.

[10] We used a version of IPOPT [9] dating from early 2003 as a floating point optimizer.

$N_{\neg \mathrm{v}}$ is the number of times a point $\hat{x}$ was found in Algorithm 1 but could not be proven to be feasible in the interval evaluation in step 7 of Algorithm 2.

$N_{\mathrm{v:\ old}}$ is the number of times existence of a point at which the m_i identified active constraints are validated through our older technique in [3] to be simultaneously active.

Columns of the tables below labelled "w/o" represent statistics for the branch and bound algorithm without the simplified validation scheme of Algorithm 2 above, whereas columns labelled "w" represent statistics for the branch and bound algorithm when Algorithm 2 above is included.

4.1. Results for OET5

The experimental results for OET5 appear in Table 1. This table displays that, although the branch and bound process successfully completed both with and without Algorithm 2 above, it completed significantly more quickly (in roughly 2/3 the time and with roughly 2/3 the number of boxes processed) when Algorithm 2 was included. It also revealed that in no case were too many active constraints identified; this is interesting, since our Lemaréchal formulation of OET5 had only $n = 5$ but 42 inequality constraints. We also see that in no case did Algorithm 2 above produce a perturbed point $\hat{x}$ that the interval arithmetic could not verify was strictly feasible, while 17736 feasible points were produced, of which 24 resulted in better lower bounds on the objective. In contrast, the old scheme could validate existence of a feasible point only 1062 times[11].

Table 1. Results for OET5. (See text for column headings.)

	w/o	w	Ratio
OK?	Y	Y	
T_{tot}	15062	9335	0.62
N_{box}	23791	15141	0.64
$N_{\mathrm{too\ many}}$	—	0	
$N_{\neg \mathrm{v}}$	—	0	
N_{v}	—	17736	
N_{better}	—	24	
$N_{\mathrm{v:\ old}}$	1062	2	

4.2. Results for the Neumaier–Shcherbina Test Set

Information related to improvements in efficiency of the branch and bound process when Algorithm 2 above is used appears in Table 2, while information

[11] In this case, the 2 times that the old scheme validated existence of a feasible point when the new scheme was also used correspond to application of that process before and after the branch and bound algorithm, at points other than those described above.

related to how reliably Algorithm 2 finds points that can be proven to be feasible and that lead to a better validated upper bound on the global optimum appears in Table 3.

Table 2. Efficiency study for the test set from [7]. (See text for column headings.)

Problem Name	OK?		T_{tot}		N_{box}	
	w/o	w	w/o	w	w/o	w
ex14.1.1	Y	Y	2.20	2.78	87	87
ex14.1.3	Y	Y	9.70	17.06	856	856
ex14.1.9	Y	Y	1.50	0.41	211	36
ex2.1.1	Y	Y	2.03	2.05	259	259
ex2.1.2	Y	Y	2.03	2.05	173	173
ex2.1.4	Y	Y	3.33	3.36	228	228
ex3.1.2	Y	Y	1.86	1.86	84	84
ex3.1.3	Y	Y	3.25	3.23	291	291
ex3.1.4	Y	Y	0.77	0.84	35	35
ex4.1.9	Y	Y	0.34	0.38	36	32
ex7.2.5	Y	Y	5.25	5.88	110	110
ex7.2.6	Y	Y	0.36	0.36	32	32
ex7.3.2	Y	Y	0.05	0.05	8	8
sample	Y	Y	38.47	8.53	141	82
Totals			71.14	48.84	2551	2313
Ratios			0.69		0.91	

As can be seen in Table 2, the branch and bound algorithm completed successfully on all of these problems, but there were significant differences in efficiency for some of the problems. Algorithm 2 had a small but significant impact on the total number of boxes for the entire set (with 10% less boxes) and a significant impact on the total processor time (30% less time). Furthermore, the total execution time was skewed by "ex14.1.3", an unusual problem in which there were many instances when more than n constraints were identified as active at the solution (as is seen from Table 2); the only other problem in which this occurred was "ex14.1.1". (We had used a heuristic in the branch and bound algorithm to decide whether to attempt the scheme from [3]; we did not use that heuristic before trying Algorithm 2, since we judged the time to actually do Algorithm 2 to be significantly less.)

The process resulted in significant decreases in the total number of boxes processed in the branch and bound in "ex14.1.9" and "sample,", but resulted in no change in the number of boxes processed in any of the other problems except "ex4.1.9". In "ex14.1.9" and "sample", our older technique from [3] was also able to verify feasibility, but our new technique apparently led to better upper bounds earlier in the algorithm, allowing rejection of larger boxes not containing optima. Finally, Algorithm 2 requires significantly less work than the scheme from [3], a fact that accounts for the bigger difference in processor time than in number of boxes processed.

Examining Table 3, we see that in no case did Algorithm 2 above produce a point that could not be proven to be feasible by evaluation of the constraints with interval arithmetic.

Table 3. Reliability study for the test set from [7]. (See text for column headings.)

Problem Name	$N_{\text{too many}}$	$N_{\neg \text{v}}$	N_{v}	N_{better}	$N_{\text{v: old}}$	
					w	w/o
ex14.1.1	133	0	0	0	7	7
ex14.1.3	1271	0	0	0	1	1
ex14.1.9	0	0	36	6	0	117
ex2.1.1	0	0	4	0	51	52
ex2.1.2	0	0	0	0	0	0
ex2.1.4	0	0	0	0	0	0
ex3.1.2	0	0	1	0	3	3
ex3.1.3	1	0	0	0	0	0
ex3.1.4	5	0	7	0	9	9
ex4.1.9	0	0	15	5	10	29
ex7.2.5	0	0	1	1	9	9
ex7.2.6	0	0	33	0	0	27
ex7.3.2	0	0	0	0	0	0
sample	0	0	82	15	0	72
Totals	1410	0	179	27	90	326

5. Handling Equality Constraints

Equality constraints can be efficiently evaluated using a combination of the techniques from §2 above and [3], as follows.

Suppose that there are m_i inequality constraints and $m_e \leq n - m_i$ active equality constraints at a point $\check{x}$, and let $A^{(\text{eq})} \in \mathbb{R}^{n \times m_e}$ be that matrix whose i-th column is the gradient of the i-th active equality constraint at $\check{x}$. Similarly, let $A^{(\iota)} \in \mathbb{R}^{n \times m_i}$ be the matrix whose i-th column is the gradient of the i-th active inequality constraint, and form the combined matrix $A = [A^{(\text{eq})}, A^{(\iota)}] \in \mathbb{R}^{n \times (m_e + m_i)}$ whose first $\neq$ columns are the columns of $A^{(\text{eq})}$ and whose last m_i columns are the columns of $A^{(\iota)}$. Compute a QR factorization:

$$A = [A^{(\text{eq})}, A^{(\iota)}] = QR = [Q^{(\text{eq})}, Q^{(\iota)}, Q^{(\text{null})}]R, \tag{3}$$

where the columns of $Q^{(\text{eq})}$ form a basis for the space spanned by the columns of $A^{(\text{eq})}$, the columns of $Q^{(\iota)}$ form a basis for the space spanned by the columns of $A^{(\iota)}$, and the columns of $Q^{(\text{null})}$ form a basis for the null space of A. The matrix $Q^{(\iota)}$ in Equation 3 will give us directions tangent to the equality constraints, in which we can perturb $\check{x}$ into the region that is feasible with respect to the inequality constraints.

Assuming that we have a point that is sufficiently inside the region defined by the inequality constraints, we need to prove existence of a nearby point at which all of the equality constraints are satisfied. To do this, we construct a box, as in [3], in an m_e-dimensional subspace of $\mathbb{R}^n$; however, in contrast to using coordinate directions as a basis for this space as in [3], we use a subspace orthogonal to the inequality constraint gradients, to maintain feasibility of the inequality constraints. To this end, we form a matrix $\tilde{A}$ as the matrix A above, except that we place the columns of $A^{(\iota)}$ first, then form a QR decomposition of the resulting matrix. That is, we form

$$\tilde{A} = [A^{(\iota)}, A^{(\text{eq})}] = \tilde{Q}\tilde{R} = [\tilde{Q}^{(\iota)}, \tilde{Q}^{(\text{eq})}, \tilde{Q}^{(\text{null})}]\tilde{R}. \tag{4}$$

As with $Q^{(\mathrm{eq})}$ in Equation 3, the columns of $\tilde{Q}^{(\mathrm{eq})}$ form a basis for the space spanned by the columns of $A^{(\mathrm{eq})}$, except that, here, the columns of $\tilde{Q}^{(\mathrm{eq})}$ are within the tangent space of the inequality constraints.

Now, define $\tilde{c} : \mathbb{R}^n \to \mathbb{R}^{m_e}$ to be the function whose i-th component is the i-th equality constraint that has been identified as active, and form $f : \mathbb{R}^{m_i} \to \mathbb{R}^{m_i}$ by

$$f(u) = \tilde{c}\left(\hat{x} + \sum_{i=1}^{m_e} u_i \tilde{Q}^{(\mathrm{eq})}_{:,i}\right), \tag{5}$$

where $\hat{x}$ will have been previously obtained by perturbing an approximate optimizing point using Algorithm 1, but with $Q^{(\iota)}$ computed from Equation (3).

The algorithm for validating existence of a feasible point in the presence of both equality and inequality constraints then proceeds as follows:

Algorithm 3 *(Finding and validating an approximate optimum of the NLP (1), when both inequality and equality constraints exist)*
INPUT: A tolerance ϵ_d
OUTPUT: Either a small box $\hat{\boldsymbol{x}}$ that has been proven to be feasible and that is likely to be near an optimizing point of the NLP (1) or a "failure" message

1. *Produce a perturbed point $\hat{x}$ as in Algorithm 1, but use*[12] *the matrix $Q^{(\iota)} = Q_{:,m_e+1:m_e+m_i}$ as in Equation (3).*
2. IF *the perturbed point from step 1 is not proven to be feasible*[13] THEN EXIT*, returning "failure".*
3. *Compute a QR factorization of the matrix $\tilde{A}$ as in Equation 4.*
4. *Form a box $\boldsymbol{u} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{m_e})$ such that $\boldsymbol{u}_i = [-\epsilon_d/10, \epsilon_d/10]$.*
5. *Use an interval Newton method with system of equations $f(u) = 0$, f as in Equation 5, and initial box $\boldsymbol{u}$ as in step 4. Let $\tilde{\boldsymbol{u}}$ be the image under the interval Newton method.*
   IF $\tilde{\boldsymbol{u}} \not\subseteq \boldsymbol{u}$ THEN
   EXIT*, returning "failure".*
   ELSE
   (a) *Iterate the interval Newton method with $\boldsymbol{u} \leftarrow \boldsymbol{u} \cap \tilde{\boldsymbol{u}}$ to obtain as narrow a set of parameters as possible.*
   (b) *Form a box $\hat{\boldsymbol{x}} = \hat{x} + \sum_{i=1}^{m_e} \boldsymbol{u}_i \tilde{Q}^{(eq)}_{:,i}$, where the $\boldsymbol{u}_i$ are the components of the parameter box that has been narrowed by the interval Newton method.*
   END IF
6. *Use interval arithmetic to bound the range of each inequality constraint $g_i(x) \leq 0$, $1 \leq i \leq m_2$, over $\hat{\boldsymbol{x}}$, that is, compute interval values $\boldsymbol{g}_i(\boldsymbol{x})$ and thus compute upper bounds $\overline{g}_i(\boldsymbol{x})$ for each $\boldsymbol{g}_i$ over $\boldsymbol{x}$.*
   IF $\overline{g}_i(\boldsymbol{x}) > 0$ *for any i* THEN

[12] Note that, when this is done, the vector $\hat{x} - \check{x}$ is orthogonal to the gradients of the equality constraints, and hence in the direction of the tangent space of the equality constraints. More precisely speaking, $\hat{x} - \check{x}$ is only approximately orthogonal to these gradients, since the QR decomposition, etc. have been done in floating point arithmetic. However, the fact that the computations are only approximate at this point do not affect the rigor in the validation.

[13] e.g. by evaluating the inequality constraints with interval arithmetic

EXIT, *returning "failure".*
ELSE
EXIT, *returning* $\hat{\boldsymbol{x}}$ *as narrow bounds within which there must exist a feasible point.*
END IF

Note 5. The widths of the box in step 4 of Algorithm 3 should be related to the tolerances used in the approximate optimizer and in the perturbation for the inequality constraints. In particular, we have suggested using $\epsilon_d/100$ as a tolerance for the approximate optimizer, ϵ_d as the length of the perturbation, and $\epsilon_d/10$ as the size of the box over which validation is attempted for the equality constraints. In any case, the perturbation for the inequality constraints should be larger than the tolerance for the approximate optimizer, to take account of inaccuracies in the approximate optimizing point. Similarly, the box widths should be larger than the tolerance of the approximate optimizer, but these box widths should also be smaller than the length of the step to perturb into the interior of the region defined by the inequality constraints; this latter condition makes it likely that the entire box lies in the interior of the region defined by the inequality constraints.

6. Experimental Results: Both Inequality and Equality Constraints

For tests of Algorithm 3, we used only those problems from the "Tiny 1" Neumaier–Shcherbina test set of [7] for which there was at least one equality constraint. As in the tests of Algorithm 2, we eliminated problems for which GlobSol had no implementation for rigorous linear relaxations and for which there were operation exceptions in the approximate solver. We also removed `ex8.1.8` because our copy of it was identical to `ex7.2.2`. Finally, we removed the problem "house". This problem did not complete successfully in 10,000 seconds with either the new or just the old method of calculating feasible points, and a study of the partial results hints that there is a continuum of optimizers in the version available to us, something that GlobSol can handle, but not efficiently.

The results appear in Table 4 and Table 5. As can be seen, the results are similar to the case with only inequality constraints. An interesting aspect is that our new technique, Algorithm 3 above, never failed to prove feasibility of a nearby point for any of the approximate optimizers passed to it. (That is, in Table 5, $N_{\neg \mathrm{v}} = 0$ for all of the problems tried.) This provides evidence that the approximate solver IPOPT is giving high-quality approximate optimizers, that the parametrization (5) defining the function for the interval Newton method is reasonable, and that the heuristics that we used to determine the relative sizes of the accuracy request passed to the approximate solver, the size of the step into the region defined by the inequality constraints, and the widths of the box constructed for the interval Newton method are adequate.

Table 4. Efficiency study for equality-constrained problems from the test set from [7]. (See text for column headings.)

Problem Name	OK?		T_{tot}		N_{box}	
	w/o	w	w/o	w	w/o	w
ex14.1.2	0	0	80.53	54.62	505	478
ex14.1.5	0	0	3.70	2.12	124	102
ex14.2.5	0	0	27.09	12.84	352	266
ex4.1.8	0	0	0.06	0.06	11	11
ex6.1.2	0	0	5.98	3.62	116	120
ex7.2.2	0	0	7.16	7.25	96	109
ex7.3.3	0	0	0.61	0.62	17	17
mhw4d	0	0	6.64	10.94	196	196
Totals			131.77	92.07	14171	1299
Ratios			0.70		0.92	

Table 5. Reliability study for equality-constrained problems from the test set from [7]. (See text for column headings.)

Problem Name	$N_{\text{too many}}$	$N_{\neg\text{v}}$	N_{v}	N_{better}	$N_{\text{v: old}}$	
					w	w/o
ex14.1.2	573	0	0	0	29	21
ex14.1.5	0	0	27	1	0	9
ex14.2.5	199	0	0	0	10	18
ex4.1.8	0	0	8	8	0	3
ex6.1.2	0	0	123	1	0	68
ex7.2.2	0	0	106	7	0	55
ex7.3.3	0	0	0	0	0	0
mhw4d	0	0	212	4	1	67
Totals	772	0	476	21	40	241

7. Conclusions

We have presented much simplified methods for computationally but rigorously proving existence of feasible points within small bounds of an approximate optimizer of an inequality and equality-constrained nonlinear program. Numerical results on a small previously published test set indicate the methods are reliable and effective.

The basic methods presented above are not applicable when more than n equality and inequality constraints have been identified as active, where n is the number of variables, although we are hopeful that progress can be made on this case, too.

References

1. G. F. Corliss and R. B. Kearfott. Rigorous global search: Industrial applications. In *Developments in Reliable Computing*, pages 1–16, Dordrecht, Netherlands, 2000. Kluwer.
2. R. B. Kearfott. *Rigorous Global Search: Continuous Problems*. Kluwer, Dordrecht, Netherlands, 1996.
3. R. B. Kearfott. On proving existence of feasible points in equality constrained optimization problems. *Math. Prog.*, 83(1):89–100, September 1998.

4. R. B. Kearfott. Empirical comparisons of linear relaxations and alternate techniques in validated deterministic global optimization, 2004, accepted for publication in *Optimization Methods and Software*, `http://interval.louisiana.edu/preprints/validated_global_optimization_search_comparisons.pdf`.
5. R. B. Kearfott. Validated probing with linear relaxations, 2004. submitted to the *Journal of Global Optimization*, `http://interval.louisiana.edu/preprints/2004_probing_and_Lagrange_multipliers.pdf`.
6. R. B. Kearfott, M. Neher, S. Oishi, and F. Rico. Libraries, tools, and interactive systems for verified computations: Four case studies. In R. Alt, A. Frommer, R. B. Kearfott, and W. Luther, editors, *Numerical Software with Result Verification, Lecture Notes in Computer Science no. 2991*, pages 36–63, New York, 2004. Springer-Verlag.
7. A. Neumaier, O. Shcherbina, W. Huyer, and T. Vinkó. A comparison of complete global optimization solvers, 2004. preprint, `http://www.mat.univie.ac.at/ neum/glopt/coconut/tests/figures/test.pdf`.
8. M. Tawarmalani and N. V. Sahinidis. *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*. Kluwer, Dordrecht, Netherlands, 2002.
9. A. Wächter. Homepage of IPOPT, 2002. `http://www.coin-or.org/Ipopt/index.html`.
10. J. L. Zhou and A. L. Titts. An SQP algorithm for finely discretized continuous minimax problems and other minimax problems with many objective functions. *SIAM J. Optim.*, 6(2):461–487, 1996.