Verified Branch and Bound for Singular Linear and Nonlinear Programs – An Epsilon-Inflation Process

Ralph Baker Kearfott
and
Julie Roy

Received: date / Accepted: date

Abstract

Linear and nonlinear programs are often ill-posed, or approximately ill-posed. However, current commercial software often is constructed with implicit assumptions of well-posedness built in. The result is that such software returns just one of many possible solutions, without any indication that singularity or approximate singularity exists. This is illustrated by applying several commercial packages, including a commercial global optimization package with complete search, to a simple example problem. However, on that same problem, a global optimization package that does not include nonsingularity assumptions does return the entire solution set, but very inefficiently. An algorithm is therefore proposed for efficiently depicting such singular solution sets and for efficiently but rigorously verifying that all such solution sets have been enclosed. This algorithm is implemented, and computations with it are illustrated.

Keywords: ill-posed nonlinear programs, branch and bound algorithms, epsilon-inflation, interval analysis, complete search

1 Introduction

In recent years, commercial high-quality global optimization problems, such as BARON [11], has become available for "complete search" algorithms for global optimization; in such complete search algorithms, completion of the algorithm finds the global optimum with certainty, provided roundoff error is not a factor. (See, for example, [10] for a further discussion of "complete search" and related concepts.) However, there has been little focus on uniqueness of optimizing points, even if it would be useful to the modeler who posed the problem to know of multiple instances of optimizing points. For example, if the problem is detected to be linear, it may be passed to a linear program solver, which might return an approximate optimizing point with no errors, even though an

entire hyperplane of optimizing points exists. As an example of this, consider the following classroom problem.

Example 1 *An investment company needs to decide how to invest* $\$200,000$ *in a mix of four stocks, with the following expected rates of return and measures of risk.*

	STOCKS			
	A	*B*	*C*	*D*
Price per share	*\$100*	*\$50*	*\$80*	*\$40*
Return per share	*0.12*	*0.08*	*0.06*	*0.10*
Risk measure per \$	*0.10*	*0.07*	*0.05*	*0.08*

Furthermore, we require that

- *The annual rate of return must be at least 9%.*
- *No one stock can account for more than 50% of the total investment.*
- *We wish to minimize the risk subject to these conditions.*

This leads to the following linear program.

Minimize $10A + 3.5B + 4C + 3.2D$

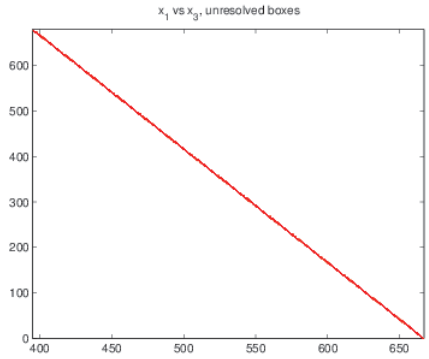
Subject to:

$$\begin{aligned} 100A + 50B + 80C + 40D &\le 200,000, \\ 12A + 4B + 4.8C + 4D &\ge 18,000, \\ 0 \le 100A \le 100000, &\qquad 0 \le 50B \le 100000, \\ 0 \le 80C \le 100000, &\qquad 0 \le 40D \le 100000. \end{aligned}$$

This simple linear problem was not designed to be special, and there is no reason to think that it should be. However, a crucial property of this problem, possibly of interest in the application area, can be overlooked when a given solver is used. (Although this problem is simple, the overlooked property and its interest to the modeler will be present to an even greater degree in larger-scale and nonlinear problems.) To illustrate, we tried the MINOS [8] solver as distributed with AMPL [2], CONOPT [1] with GAMS [3], BARON [12] through NEOS [9], and GlobSol [5]. The general MINOS solver and CONOPT are designed for large-scale constrained nonlinear optimization, and do not pretend to be global optimizers, while BARON, although not claiming to be mathematically rigorous, claims to offer complete search for global optimization. GlobSol is an interval-arithmetic-based branch and bound code for small problems in constrained and unconstrained global optimization, that claims to output mathematically rigorous enclosures to all global optimizing points, provided its search completes. The results are as follows:

Solver	A	B	C	D	min
MINOS (ampl)	$666.\overline{6}$	0	0	2500	14666
CONOPT (gams)	$333.\overline{3}$	0	$833.\overline{3}$	2500	14666
BARON (neos)	$666.\overline{6}$	0	0	2500	14666
GlobSol	See below				14666

MINOS, CONOPT, and BARON completed almost immediately, with normal termination and no warnings, but GlobSol did not complete its search, even after hours. However, GlobSol's partial output included a list of boxes that contained both of the solutions returned by MINOS, BARON, and CONOPT. In fact, GlobSol's list of unresolved boxes, plotted quantity A versus quantity C, is as follows:



In all of the unresolved boxes represented in this figure, the quantity A is within approximately 0.06 of 0, while the quantity D is very near 2500. Also, there was a severe "clustering effect" in the sense that the line in the figure depicted above represents 256,903 boxes.

A brief analysis of the problem shows that the objective gradient and the gradient of the constraint on the total amount invested are linearly dependent, resulting in a line in parameter space upon which the objective is minimum. A random perturbation will remove this singularity. For example, if we perturb Example 1 as follows

Old / New

	A	B	C	D
Price	\$100	\$50	\$80	\$40
Return	$\frac{0.12}{0.124}$	0.08	$\frac{0.06}{0.061}$	0.10
Risk	$\frac{0.10}{0.101}$	0.07	$\frac{0.05}{0.051}$	0.08

then all four solvers give $A \approx 645.161$, $B = C = 0$, $D = 2500$, and a minimum of $14,516$, values that are close to one of the particular solutions to the unperturbed problem; in fact, an experimental version of GlobSol, using LP relaxations, completed this perturbed problem with a search tree consisting of only 163 boxes.

One might be tempted to accept the answer to the perturbed problem as an approximate answer to the singular problem, and to not attempt to describe the solution set to the original problem. However, the original application, namely, minimizing the risk of a particular portfolio, indicates that the investor might be interested in the entire choice of possibilities, or at least in knowing that

the choice exists. For example, although there is a continuum of portfolios that minimize the risk, some might have a larger potential profit, or be more desirable according to other criteria. Furthermore, even if the problem is only approximately ill-posed, it may be desirable for the same reasons to compute and describe approximate solution sets. This leads to the following questions:

1. How can we efficiently characterize and represent solution sets of ill-posed problems?
2. How can we efficiently locate such solution sets and eliminate the remaining portions of the search region in a branch and bound algorithm?
3. Can we design such techniques for approximately singular problems, in addition to exactly singular problems?
4. Can our techniques be applicable generally, to constrained nonlinear programs?

This work addresses these questions.

2 Representation of Singular Solution Sets

The "clustering" problem referenced in the introduction is largely due to the fact that a solution whose orientation is skewed with respect to the original coordinate axes requires a large number of boxes with sides parallel to the coordinate axes to cover even a small portion of it tightly. This has been observed many times, such as in [7, Fig. 12]; we illustrate the phenomenon for a hypothetical tessellation resulting from a branch and bound process in Figure 1. In this figure, it is assumed that boxes 1 through 6 are identified, in that order, as containing solutions; after taking the complement of these boxes in the list of unfathomed boxes (as in [4, Algorithm 11, p. 156]), we obtain the other narrow boxes depicted in the figure.

Reasoning intuitively, we conclude that representation of a box in terms of coordinates that are locally tangent and locally orthogonal to the solution set will be advantageous. We take this general approach.

2.1 Context and Notation

Although Example 1 is a simple linear program, we view it as a canonical case of a phenomenon that can occur in general constrained nonlinear programs, and we discuss our methods within that general context. The general problem we

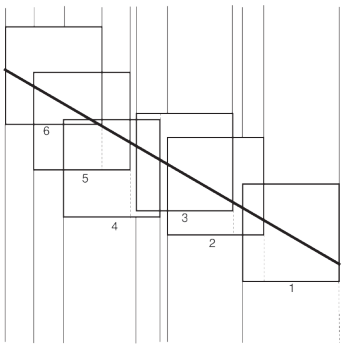


Figure 1: Hypothetical tessellation in a branch and bound algorithm over a singular set

consider will be stated as

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

We distinguish between possible bound constraints, which here we assume will be included among the inequality constraints $g_i(x) \leq 0$, and simple limits $x \in \boldsymbol{x}$ on the search region.

2.2 Directions for the Solution Set

Suppose $\check{x} \in \boldsymbol{x}$ is an approximately feasible point point at which φ is approximately optimal, and suppose φ, the c_i, and the g_i have continuous partial derivatives throughout $\boldsymbol{x}$. (The point $\check{x}$ may be obtained through any efficient local optimizer.) Also, suppose that the inequality constraints $\left\{g_{i_j}\right\}_{j=1}^{n_\mathrm{a}}$ are active at $\check{x}$. Then, locally, directions that are perpendicular to both $\nabla\varphi$, all of the $\nabla \boldsymbol{c}_i$, and the $\nabla \boldsymbol{g}_{i_j}$ will be directions v such that $\check{x} + \epsilon v$ remains approximately feasible and approximately optimal, for sufficiently small ϵ. (There may be additional directions pointing into the feasible region defined by the inequality constraints, but we will ignore these for now.) We then form a matrix G whose columns consist of the gradients $\nabla\varphi(\check{x})$, $\nabla c_i(\check{x})$, and $\nabla g_{i_j}(\check{x})$ of the objective, equality constraints, and active inequality constraints:

$$G = \left[\nabla\varphi(\check{x}), \nabla c_1(\check{x}), \ldots, \nabla c_{m_1}(\check{x}), \nabla g_{i_1}(\check{x}), \ldots, \nabla g_{i_{n_\mathrm{a}}}(\check{x})\right]. \tag{2}$$

Through standard linear algebra techniques (such as a singular value decomposition) we then obtain an orthonormal basis $\{v_i\}_{i=1}^{n_\mathrm{w}}$ for the null space of G, and

we obtain an orthonormal basis $\{v_i\}_{i=n_{\rm w}+1}^{n_{\rm s}}$ for the orthogonal complement of the null space of G. (Here, we may look at approximate null spaces in the sense that corresponding singular values are small.) We denote the space spanned by the vectors $\{v_i\}_{i=1}^{n_{\rm w}}$ (directions in which the point remains approximately optimal) by $\mathcal{W}$, and we denote the space spanned by the vectors $\{v_i\}_{i=n_{\rm w}+1}^{n_{\rm s}}$ (directions in which the objective value increases or the problem becomes infeasible) by $\mathcal{S}$. (More generally, rather than divide the basis into two sets, we may order the v_i in order of increasing singular values of G, representing directions of increasing importance in determining optimality.)

2.3 Construction in the New Coordinates

We represent points z near our approximate optimizing point $\check{x}$ in terms of the coordinates v_i and w_i:

$$z = \check{x} + \sum_{i=1}^{n_{\rm w}} \alpha_i v_i + \sum_{i=n_{\rm w}+1}^{n} \alpha_i v_i \tag{3}$$

The key to the success of our techniques is expanding the objective and constraints in Taylor series in terms of the new basis $\{v_i\}_{i=1}^{n}$. (Automatic differentiation technology enables us to do this in a practical way.) In particular, if

$$f \in \left\{\varphi, \quad \{c_i\}_{i=1}^{m_1}, \quad \{g_{i_j}\}_{j=1}^{n_{\rm a}}\right\},$$

then we write

$$f(z) \in f(\check{x}) + \sum_{i=1}^{n_{\rm w}} \alpha_i D_{v_i}(f)(\boldsymbol{x}^{(\rm B)}) + \sum_{i=n_{\rm w}1}^{n} \alpha_i D_{v_i}(f)(\boldsymbol{x}^{(\rm B)}), \tag{4}$$

where $\boldsymbol{x}^{(\rm B)}$ is some *bounding box* centered on $\check{x}$ and $D_{v_i}(f)(\boldsymbol{x}^{(\rm B)})$ is an interval extension of the directional derivative of f in the direction v_i over $\boldsymbol{x}^{(\rm B)}$. We then construct new coordinates $\boldsymbol{\alpha}_i = [-\alpha_i, \alpha_i]$ for a *skewed box* $\boldsymbol{\alpha}$

$$\boldsymbol{\alpha} = \left\{\check{x} + \sum_{i=1}^{n} \alpha_i v_i, \quad \alpha_i \in \boldsymbol{\alpha}_i, \quad 1 \le i \le n.\right\} \tag{5}$$

In (5), the extents $|\alpha_i|$ in the directions v_i are determined according to target tolerances ϵ_φ, ϵ_c, and ϵ_g, such that φ be within ϵ_φ of optimal, the c_i be within ϵ_c of feasible, and the g_{i_j} be within ϵ_g of feasible. In particular, we set a *bounding box* $\boldsymbol{x}^{(\rm B)}$, then choose the $|\alpha_i|$ so that

$$\begin{aligned}
\sum_{i=1}^{n} |\alpha_i||D_{v_i}(\varphi)(\boldsymbol{x}^{(\rm B)})| &\le \epsilon_\varphi, \\
\sum_{i=1}^{n} |\alpha_i||D_{v_i}(c_j)(\boldsymbol{x}^{(\rm B)})| &\le \epsilon_c, \qquad 1 \le j \le m_1 \\
\sum_{i=1}^{n} |\alpha_i||D_{v_i}(g_{j_k})(\boldsymbol{x}^{(\rm B)})| &\le \epsilon_g, \qquad 1 \le k \le n_{\rm a}.
\end{aligned} \tag{6}$$

Equations (6) will be true if

$$r_i = |\alpha_i| = \{\text{radius of the skewed box in the direction of } v_i\}$$

obeys

$$\begin{aligned} r_i \quad = \min_{j,k} \big\{ \quad & \epsilon_\varphi / \left(n|D_{v_i}(\varphi)(\boldsymbol{x}^{(\mathrm{B})})|\right), \\ & \epsilon_c / \left(n|D_{v_i}(\boldsymbol{c}_j)(\boldsymbol{x}^{(\mathrm{B})})|\right), \\ & \epsilon_g / \left(n|D_{v_i}(\boldsymbol{g}_k)(\boldsymbol{x}^{(\mathrm{B})})|\right) \big\}, \end{aligned} \tag{7}$$

where the minimum is over all all equality constraints and active inequality constraints, and where a term is not included if the corresponding $|D_{v_i}|$ is less than the machine epsilon ϵ_{M}. Additionally, we require

$$r_i \leq \min_{\substack{1 \leq \imath \leq n, \\ |v_{i,j}| > \epsilon_{\mathrm{M}}}} \frac{w(\boldsymbol{x}_i^{(\mathrm{B})})}{2}, \tag{8}$$

where $v_{i,j}$ is the j-th component of the i-th basis vector v_i and $w(\boldsymbol{x}_i^{(\mathrm{B})})/2$ is the radius of the i-th coordinate of the bounding box. The conditions (8) are essentially necessary conditions for $\boldsymbol{\alpha} \subseteq \boldsymbol{x}^{(\mathrm{B})}$.

Once the skewed box is initially determined according to (7), a *containing box*, that is, a box $\boldsymbol{x}^{(\mathrm{C})}$ whose sides are parallel to the coordinate axes and that tightly encloses the skewed box to within roundout error, is determined by

$$\boldsymbol{x}^{(\mathrm{C})} = \check{x} + \sum_{i=1}^{n} \boldsymbol{\alpha}_i v_i = \check{x} + V\boldsymbol{\alpha}, \tag{9}$$

where V is the $n \times n$ matrix whose i-th column is v_i.

Finally, observe that the interval extensions (4) are valid only if $\boldsymbol{\alpha} \subset \boldsymbol{x}^{(\mathrm{B})}$, so that it is guaranteed that $|\varphi(x) - \varphi(\check{x})| \leq \epsilon_\varphi$, $|c_i(x)| \leq \epsilon_c$, and $g_{i_j}(x) \leq \epsilon_g$ for every $x \in \boldsymbol{\alpha}$ only if $\boldsymbol{\alpha} \subset \boldsymbol{x}^{(\mathrm{B})}$. For this reason, after our initial computation of $\boldsymbol{\alpha}$ and $\boldsymbol{x}^{(\mathrm{C})}$ using (7), the bounding box widths are individually and iteratively adjusted larger and smaller, along with recomputation of $\boldsymbol{\alpha}$ and $\boldsymbol{x}^{(\mathrm{C})}$, until $\boldsymbol{x}^{(\mathrm{C})}$ and $\boldsymbol{x}^{(\mathrm{B})}$ have widths that are roughly the same. Actual bounds on φ and the c_i and g_{i_j} are then obtained by evaluating the expressions (4) using interval arithmetic.

3 Elimination Near Singular Solution Sets

The skewed box $\boldsymbol{\alpha}$ of the previous section represents a set of bounds, wide in some directions and narrow in others, enclosing the solution set. To avoid the clustering effect, we want to be able to efficiently reject the region around $\boldsymbol{\alpha}$ as not containing approximate optimizers. To achieve this, it makes sense to represent the adjacent regions as boxes with sides parallel to the v_i, rather than parallel to the standard coordinate axes. Furthermore, if we proceed outward from the sides of $\boldsymbol{\alpha}$ in directions $\{v_i\}_{i=1}^{n_{\mathrm{w}}}$, the objective and constraints will not

change very much, and it will be unlikely we can reject corresponding portions of the search region. On the other hand, if we proceed outward from $\boldsymbol{\alpha}$ in directions $\{v_i\}_{i=n_{\mathrm{w}}+1}^{n_{\mathrm{w}}+n_{\mathrm{s}}}$, the objective or constraints will be changing, and we will be more likely to be able to reject a corresponding portion of the search space. This is illustrated in Figure 2.

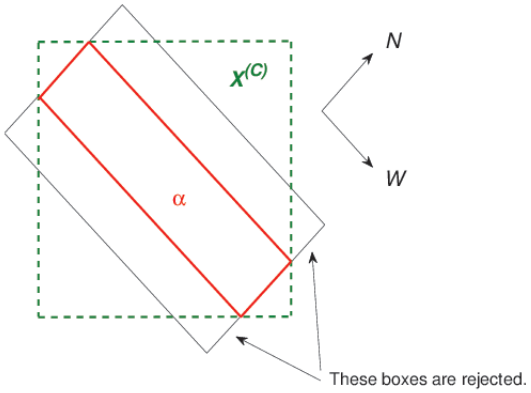


Figure 2: Illustration of the skewed box, the containing box, and elimination around the skewed box

With these considerations, we construct boxes proceeding outward from $\boldsymbol{\alpha}$, adjacent to the $2n_{\mathrm{s}}$ sides of $\boldsymbol{\alpha}$ corresponding to $\alpha_i = \pm r_i$, $i = n_{\mathrm{w}} + 1$ to $i = n_{\mathrm{w}} + n_{\mathrm{s}}$. This is done with a type of "epsilon inflation" process (akin to classical epsilon inflation as in [6]):

1. An initial box "sliver" $\boldsymbol{\alpha}^{\mathrm{S}}$, represented in terms of the skewed coordinates, is constructed, with all coordinate bounds except the i-th bound equal to corresponding bounds on the skewed box $\boldsymbol{\alpha}$, and with the i-th coordinate bounds having one bound equal to either $-\alpha_i$ or α_i (depending on which side of $\boldsymbol{\alpha}$ is being considered), and the other bound having value $\pm(\alpha_i + \epsilon)$, where ϵ is appropriately chosen according to accuracy tolerances in the computation.

2. It is determined that $\boldsymbol{\alpha}^{\mathrm{S}}$ can be rejected by first computing a containing box for $\boldsymbol{\alpha}^{\mathrm{S}}$, then evaluating φ, the c_i, and the g_{i_j} over $\boldsymbol{\alpha}^{\mathrm{S}}$ using the representation (4), where the containing box for $\boldsymbol{\alpha}^{\mathrm{S}}$ is used in place of the bounding box $\boldsymbol{x}^{(\mathrm{B})}$. (If the lower bound on φ is greater than a verified upper bound on the global optimum, if the range bounds on at least one of the c_i do not contain zero, or if the lower bound on at least one of the g_i is greater than zero, then $\boldsymbol{\alpha}^{\mathrm{S}}$ can be rejected.)

3. ϵ is increased, $\boldsymbol{\alpha}^{\mathrm{S}}$ is re-constructed (i.e. $\boldsymbol{\alpha}^{\mathrm{S}}$ is expanded outward in the direction of v_i), the containing box for $\boldsymbol{\alpha}^{\mathrm{S}}$ is recomputed, and it is re-

checked that $\boldsymbol{\alpha}^{\mathrm{S}}$ can be rejected. This is done iteratively until a maximally wide $\boldsymbol{\alpha}^{\mathrm{S}}$ that can be rejected is found.

Once all $2n_{\mathrm{s}}$ box slivers $\boldsymbol{\alpha}^{\mathrm{S}}$ are so constructed, the union of these with the original skewed box $\boldsymbol{\alpha}$ is another skewed box $\boldsymbol{\Lambda}$ that can be eliminated from the search region in the branch and bound process. Although the branch and bound process works with boxes with sides parallel to the original coordinate axes, a box $\boldsymbol{x}$ in such a branch and bound process has corresponding skewed coordinate bounds

$$\boldsymbol{z} = V^T(\boldsymbol{x} - \check{x});$$

any boxes produced during the branch and bound process whose corresponding skewed coordinate bounds lie completely within the bounds $\boldsymbol{\Lambda}$ can be rejected. Because the sensitivities of the objective and constraints to various directions have been used, there is less interval dependency in construction of $\boldsymbol{\Lambda}$, and the result is a more powerful fathoming device in the branch and bound process than simply using the standard coordinate directions.

4 Some Numerical Results

These techniques were implemented, utilizing the interval arithmetic and automatic differentiation capabilities within our GlobSol environment. We then tried the techniques on both the exactly singular example and the approximately singular perturbation of it from the introduction to this paper. For Example 1, with upper bound on the global optimum $\overline{\varphi} = 14667.14175$, $\epsilon_\varphi = \epsilon_c = \epsilon_g = 10^{-2}$, initial approximate optimizing point $\check{x} \approx (666.7, 0, 0, 2500)$, and initial bounding box radii equal to 0.1. This resulted in $n_{\mathrm{w}} = 1$, $n_{\mathrm{s}} = 3$, and the basis of v_i approximately as in the following table.

v_1	v_2	v_3	v_4
−0.3714	0.0604	0.8833	−0.2796
0.0000	0.0205	0.3005	0.9536
0.9285	0.0242	0.3533	−0.1119
0.0000	0.9777	0.0000	0.0000

In the process, we used a small box $\boldsymbol{x}^{\mathrm{V}}$, given in terms of the original coordinates, within which it is assumed that a feasible point has been verified to exist. We obtained the following coordinates for the epsilon-inflated skewed box $\boldsymbol{\Lambda}$, the containing box $\boldsymbol{X}^{(\mathrm{C})}$ for $\boldsymbol{\Lambda}$, along with corresponding interval evaluations of the active constraints; here, all bounds are rounded outward into the precision displayed.

Box:	$\boldsymbol{x}^{\mathrm{V}}$	$\boldsymbol{\Lambda}$	$\boldsymbol{X}^{(\mathrm{C})}$
coords.	$[653, 681]$	$[-2 \times 10^{11}, 2 \times 10^{11}]$	$[-5 \times 10^{10}, 5 \times 10^{10}]$
	$[0, 1.97]$	$[-47.7, 0.63]$	$[-4.9, 7.6]$
	$[0, 1.97]$	$[-12.4, 18.8]$	$[-2 \times 10^{11}, 2 \times 10^{11}]$
	$[2450, 2500]$	$[-.135, 1.96]$	$[-2452, 2502]$
obj.	$[14369, 14811]$	$[14139, 14681]$	$[-9 \times 10^{11}, 9 \times 10^{11}]$
act. con.	$[-178, 361]$	$[-178, 361]$	$[-1 \times 10^{12}, 1 \times 10^{12}]$
	$[-2000, \approx 0]$	$[-2000, \approx 0]$	$[-1955, 58.91]$

This table makes the advantages of dealing with the skewed box clear.

In a second test, we applied the same methods to the perturbation of the singular problem we presented in the introduction to this work. For this approximately singular problem, we used $\check{x} \approx (645.2, 0, 0, 2500)$, $\overline{\varphi} \approx 14516.2$, and all of the other parameters and tolerances the same as for the singular problem. As with the exactly singular problem, a constraint was identified to be active if it is within 1 unit of 0. With those criteria, the same constraints were identified as active as in the exactly singular problem, and $n_{\mathrm{w}} = 1$. Here, we compare v_1 for the exactly singular problem to v_1 for the approximately singular problem:

exact. sing.	approx. sing.
-0.3714	-0.2263
-0.0000	-0.3883
0.9285	0.8933
0.0000	0.0000

Thus, even when the problem is only approximately singular, we obtain a direction in which the problem remains approximately feasible and in which the objective value does not change much. A corresponding table, for the approximately singular problem, with coordinates for the $\boldsymbol{x}^{\mathrm{V}}$ we used, $\boldsymbol{\Lambda}$, $\boldsymbol{X}^{(\mathrm{C})}$, and corresponding bounds on the objective and active constraints, is:

Box:	$\boldsymbol{x}^{\mathrm{V}}$	$\boldsymbol{\Lambda}$	$\boldsymbol{X}^{(\mathrm{C})}$
coords.	$[640, 650]$	$[-2 \times 10^{11}, 2 \times 10^{11}]$	$[-3 \times 10^{10}, 3 \times 10^{10}$
	$[0, 1]$	$[-2.8, 2.4]$	$[-5 \times 10^{10}, 5 \times 10^{10}]$
	$[0, 1]$	$[-4.9, 4.8]$	$[-1 \times 10^{11}, 1 \times 10^{11}]$
	$[2499, 2500]$	$[-.071, .998]$	$[2496, 2504]$
obj.	$[14450, 14580]$	$[14470, 14580]$	$[-8 \times 10^{11}, 8 \times 10^{11}]$
act. con.	$[-69, 69]$	$[-69, 69]$	$[-1 \times 10^{12}, 1 \times 10^{12}]$
	$[-40, \approx 0]$	$[-40, \approx 0]$	$[-124, 107]$

This illustrates that the methods are insensitive to perturbations, and are useful even when the singularity is only approximate.

5 Improvements and Future Work

The methods described here use only first-order information, and will be advantageous only for problems that are locally approximately linear and that have a significant number of active constraints at optimizing points. To see this, observe that, for an unconstrained problem, $\nabla\varphi(\check{x}) \approx 0$, so G in (2) is the zero

matrix, $n_{\mathrm{w}} = n$, and the skewed coordinate system can taken to be the original coordinate system. However, in unconstrained problems with non-isolated sets of optimizing points, the Jacobi matrix of the gradient, that is, the second-order derivative tensor of φ is typically singular at the approximate optimizing points. If we denote this second-order tensor by $\nabla^2\varphi$, then directions v in which the skewed box can be wide are defined by those directions v for which

$$(\nabla\varphi + \nabla^2\varphi(v)) \circ v \approx 0.$$

Similar second-order expansions can be used for active constraints whose gradients vanish at approximate optimizing points. If, among the objective and constraints, there are various such functions whose gradients vanish, we obtain a system of quadratic equations that the components of v must satisfy. Additional analysis should yield efficient ways of determining such directions v, and hence of determining appropriate bases for skewed boxes. In this context, the representations (4) should possibly be replaced by second-order representations, with the first order directional derivatives $D_{v_i}(f)$ evaluated at the point $\check{x}$ and with interval extensions for the second-order directional derivative matrices $D_{v_i,v_j}(\boldsymbol{x}^{(\mathrm{B})})$. For more bizarre types of singularities, where the second derivative tensors are singular, higher-order tensors could possibly be employed.

Acknowledgements

I wish to thank Arnold Neumaier for suggesting that I represent singular solutions as in Example 1 in terms of the null space of the objective and constraints. I also wish to thank Shin'ich Oishi, Michael Plum, and Siegfried Rump for inviting me to the International Workshop on Numerical Verification and its Applications, Waseda University, March, 2007, where the initial investigations leading to this work were first presented. Finally, I wish to acknowledge Anthony Holmes, who supplied the example problem, who did the initial computations with it, and who pointed out the usefulness of knowing the entire solution set.

References

[1] Arne Drud. CONOPT: A GRG code for large sparse dynamic nonlinear optimization problems. *Mathematical Programming*, 31:153–191, 1985.

[2] R. Fourer, D. Gay, and B. Kernighan. *AMPL A Modeling Language for Mathematical Programming.* Boyd and Frazer, Danvers, Massachusetts, 1993.

[3] GAMS.

[4] R. Baker Kearfott. *Rigorous Global Search: Continuous Problems.* Number 13 in Nonconvex optimization and its applications. Kluwer Academic Publishers Group, Norwell, MA, USA, and Dordrecht, The Netherlands,

1996. Includes a module on interval arithmetic, as well as Fortran 90 code for automatic differentiation, nonlinear systems code, and constrained and unconstrained optimization.

[5] R. Baker Kearfott, Markus Neher, Shin'ichi Oishi, and Fabien Rico. Libraries, tools, and interactive systems for verified computations four case studies. *Lecture Notes in Computer Science*, 2991:36–63, 2004.

[6] Günter Mayer. Epsilon-inflation in verification algorithms. *J. Comput. Appl. Math.*, 60(1-2):147–169, 1995.

[7] R. E. Moore. *Interval Arithmetic and Automatic Error Analysis in Digital Computing.* Ph.D. dissertation, Department of Mathematics, Stanford University, Stanford, CA, USA, November 1962. Also published as Applied Mathematics and Statistics Laboratories Technical Report No. 25.

[8] B. A. Murtagh and M. A. Saunders. Minos 5.5 user's guide(rev). Technical Report SOL 83-20R, Department of Operation Research, Stanford University, Stanford CA, 1998.

[9] NEOS Server for Optimization Problems.

[10] Arnold Neumaier. Complete search in continuous global optimization and constraint satisfaction. In A. Iserles, editor, *Acta Numerica 2004*, pages 271–369. Cambridge University Press, 2004.

[11] Nick Sahinidis and Mohit Tawarmalani. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming, Series B*, 103(??):225–249, ???? 2005. Winner of the Beale-Orchard-Hays Prize administered by the Mathematical Programming Society.

[12] Nikolaos V. Sahinidis. BARON: A general purpose global optimization software package. *J. Global Optim.*, 8(2):201–205, 1996.