Detecting Unboundedness in Nonlinear Models

Arne Stolbjerg Drud
ARKI Consulting & Development A/S
Bagsvaerd
Denmark
adrud@arki.dk

Overview:

- Introduction and Background
- The Difference between LP and NLP
- Some Initial Results
- Definitions of Unboundedness
- What Goes Wrong? Examples of Unbounded Model that do not terminate “Unbounded”.
- The Test Models
- Results with an updated CONOPT3
- Conclusion and Summary of Difficulties

Introduction and Background

GAMS uses a two-part classification of a solution:

Model Status (a description of the numerical solution that has been returned):

- (Locally) Optimal
- (Locally) Infeasible
- Unbounded
- Intermediate Infeasible or Non-optimal
- Error / Failure

Solver Status (a description of how the algorithm behaved):

- Normal Completion
- Resource Interrupt (Iterations, CPU Time, User-Interrupt)
- Interrupt due to too many Function Evaluation Errors
- Cannot Proceed, intermediate Solution exist
- Error or Failure with No solution

Introduction and Background (cont)

The combinations of Model Status and Solver Status form a matrix where the entries can be classified as follows:

	Normal	Resource Interrupt	Function Errors	Cannot Proceed	Error
Optimal	Good	Illegal	Illegal	Illegal	Illegal
Infeasible	Good	Illegal	Illegal	Illegal	Illegal
Unbounded	Good	Illegal	Illegal	Illegal	Illegal
Intermediate	Illegal	Acceptable	Acceptable	Acceptable	Illegal
Error	Illegal	Illegal	Illegal	Illegal	Bad

A Performance Program tries to solve models more quickly, thereby minimizing the number of blue combinations.

A Quality Control Program must among other things assure that a solver never returns any of the red combinations. In addition, the model status for the green combinations must be correct.

The purple combinations usually indicate a solver problem. They can also be caused by non-smooth models being solved by a smooth solver. The yellow combinations will usually indicate a modeling problem.

We will later show that unbounded models in practice may return all kinds of combinations of model/solver status.

The Difference Between LP and NLP

An unbounded LP model has an unbounded ray in a direction with improving objective.

Example:

Max	$x2$
Subject to	$x2 <= 0.5*x1$
	$x1 \geq 0, x2 \geq 0$

An unbounded NLP model may not have an unbounded ray in a direction with improving objective.

Example:

Max	$x2$
Subject to	$x2 <= \mathrm{sqrt}(x1)$
	$x1 > 0, x2 > 0$

The Difference Between LP and NLP (cont)

An **LP model** can be proved unbounded with a pair of n-vectors: A feasible point, x, and an unbounded feasible ray, d, emerging from this point. The numerical work involved in defining x and d can all be done close to the origin, i.e. without actually moving towards infinity and without working with very large numbers.

How is an **NLP model** proven to be unbounded? For a general non-convex NLP we must go all the way towards infinity, and all kinds of numerical problems may pop up on the way.

Some Initial Results

8 families of clearly unbounded models with a total of 38 instances have been defined. These instances have been solved using MINOS, SNOPT, CONOPT2, and CONOPT3 under GAMS distribution 21.1 (from June 2003).

The classification of the 158 solves (based on default tolerances and options) were:

	Normal	Resource Interrupt	Function Errors	Cannot Proceed	Error
Optimal	25				
Infeasible	2				
Unbounded	41				
Intermediate		31	8	45	
Error					

With only 41 = 26% correct classifications we must conclude that detection of unboundedness is not a strong side of the selected solvers.

Some Initial Results (cont)

We have translated the same 38 instances into AMPL using GAMS/CONVERT and solved them using KNITRO, LOQO, FILTER, and LANCELOT via the NEOS Server and using default options and tolerances (version August 2003).

AMPL and these solvers use different classifications of the solution so we have performed a manual translation into the GAMS classification. Not all cases were clear: A solution declared optimal by a solver but with an objective value of 1.e20 is here classified as optimal.

The classification of the 158 solves were:

	Normal	Resource Interrupt	Function Errors	Cannot Proceed	Error
Optimal	41				
Infeasible	5				
Unbounded	15				
Intermediate		74	4	12	
Error					1

These results with these solvers are not better than with the GAMS solvers, and we must again conclude that detection of unboundedness is not a strong side of these solvers.

Some Initial Results (cont)

The classification of the 38 solves for CONOPT3 (again based on default tolerances) were:

	Normal	Resource Interrupt	Function Errors	Cannot Proceed	Error
Optimal	10				
Infeasible	0				
Unbounded	6				
Intermediate		8	2	12	
Error					

Many of the 'Cannot Proceed' cases are caused by violation of a default upper limit on the size of Jacobian elements (around 1.e7), intended to prevent users from trying to solve very poorly scaled models.

Since unbounded models can have very large Jacobian elements, we have tried to increase this limit is to 1.e30. The results were only slightly better:

	Normal	Resource Interrupt	Function Errors	Cannot Proceed	Error
Optimal	11				
Infeasible	0				
Unbounded	13				
Intermediate		7	2	5	
Error					

Definitions of NLP Unboundedness

Mathematical Definitions:

1: An NLP model is unbounded if, for any given large number M there exist a feasible solution, x, such that $x_{obj} \geq M$.

2: An NLP model is unbounded if there exist a feasible path, x(?), such that for any M there exist a ? such that $x_{obj}(?) \geq M$.

3: An NLP model is unbounded if there exist a feasible path, x(?), such that the objective increases with ? and for any M there exis t a ? such that $|x(?)| \geq M$.

Definition 2 and 3 are fairly different since 2 require the objective to become very large while 3 just need very large variable(s). The difference is shown in the following model:

Max x_2, s.t. $x_2 = -1/x_1, x_1 \geq 0$

It is unbounded according to definition 3 but not according to definition 2.

The definitions are not operational because they have the 'for any large M' clause.

Definitions of NLP Unboundedness (cont)

Numerical Definition: An NLP model is unbounded if, given a fixed large M, there exist a feasible point, x with $|x| \geq M$, and a feasible tangent direction in x, d, such that $d_{obj} > 0$.

This definition is natural for a feasible-path NLP solver (and many other types of NLP solvers):

Replace all ‘Infinity’ bounds with a large value (M) and solve the model. If any of the M-bounds become binding with a nonzero marginal then the model considered is unbounded. The test can be applied to the final solution or to intermediate solutions.

This definition is used in CONOPT (based on intermediate solutions). It corresponds to the mathematical definition 3 above.

Note that in a Modeling System like GAMS the direction vector d cannot be returned to the user (without hiding it in the vector of marginals). Only part of the “Proof of unboundesness”, namely x, is made available.

What Went Wrong with the Unbounded Models?

Problem 1: The solver stops and declares the final solution **“Locally Optimal”**.

Example:	Max	x_2
	s.t.	$x_2 = x_1$**p, $x_1 \geq 0$

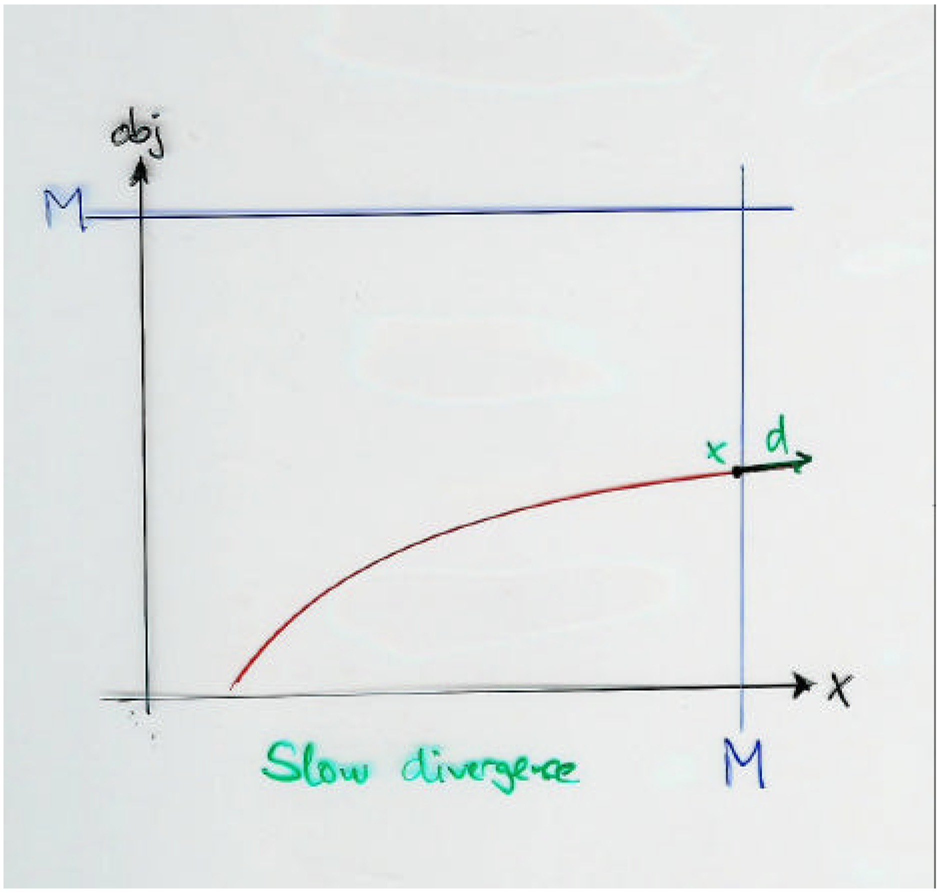
with p a small positive number.

With M = 1.e15 and optimality tolerance = 1.e-7 we get the following critical x_1 values, x_{cr}, above which a numerical solver will declare the solution optimal:

p	x_{cr}	$x_2(x_{cr})$	$x_2(M)$
0.50	2.5e13	5.0e6	3.1e7
0.25	3.4e8	136	5623
0.10	4.6e7	4.64	31.6

The only way to reduce the number of false “Locally Optimal” classifications is to reduce the optimality tolerance which seems unrealistic.

I do not see much hope that we can “fix” this problem.


obj
M
x
d
X
M
Slow divergence

What Went Wrong with the Unbounded Models (cont)?

Problem 2: The solver stops and declares the final solution **“Locally Infeasible”**.

This can be caused by very large intermediate values in the calculation of the constraints combined with small absolute tolerances.

Other problems come from very small derivatives that are ignored or create very large duals.

Example: Max x_2

s.t. $x_2 * x_1**(-p) = 1, x_1 \geq 0$

The model is mathematically equivalent to the one above, but x_2 is not an implicitly defined function of x_1. With p=4, SNOPT stops with $x_1 = 614$ and $x_2 = 2.95e7$. The left hand side is 2.e-4 and the point is therefore infeasible. The Jacobian of the constraint is (-0.51, 7.e-12). The derivative seems to be so small that it is ignored, making the model feasible.

Aggressive scaling without ignoring small Jacobian elements seems to be the only solution to these problems.

What Went Wrong with the Unbounded Models (cont)?

Problem 3: The solver stops on **“Iteration Interrupt”.**

The following model is clearly unbounded:

$$\begin{array}{ll} \text{Max} & x_3 = x_1 + x_2 \\ \text{s.t.} & x_2 = x_1**1.5 \\ & x_1 \geq 0,\ x_2 \geq 0,\ x_3 \geq 0 \end{array}$$

Now change the constraint to the mathematically equivalent

$$x_2**(2*p) = x_1**(3*p)$$

for some integer values of p (1, 2, 3, and 4) and the model is of course still unbounded.

The constraints become increasingly nonlinear with larger p values. In addition, x_2 is implicitly defined and progress can be very slow.

The model can also terminate **“Locally Infeasible”** if the solver uses an absolute feasibility tolerance. The nonlinear constraints have very large terms that are supposed to cancel each other.

Problem 3: The solver stops on **“Iteration Interrupt” (continued).**

For p=1, the most mildly nonlinear version of the model, we get the following results:

MINOS seems to be doing very well with 2 minor iterations per major iteration and step=1 between major iterations. However, the change in objective is only 4-8 per iteration.

SNOPT solves the SQP sub-problems in 1 to 3 minor iterations, but the step is always between 1.e-5 and 1.e-12 so progress is very slow.

CONOPT2 and CONOPT3 have a common bug that prevents large steps for this model. The change in objective is exactly 1.0 in each iteration.

There does not seem to be any obvious fix to these problems, except to remove bugs and to use models like the above as test models.

What Went Wrong with the Unbounded Models (cont)?

Problem 4: The solver stops and reports **"Cannot Proceed"** or **"Terminated by Solver".**

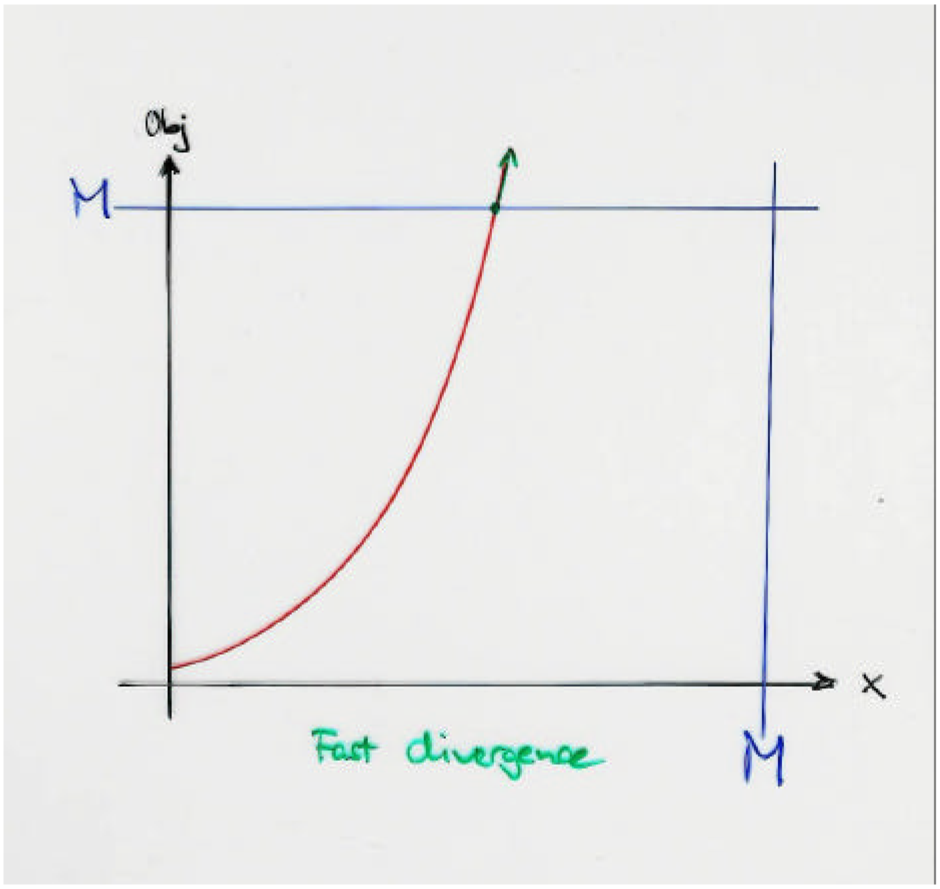
This can happen for a large number of reasons and we will provide a few examples:

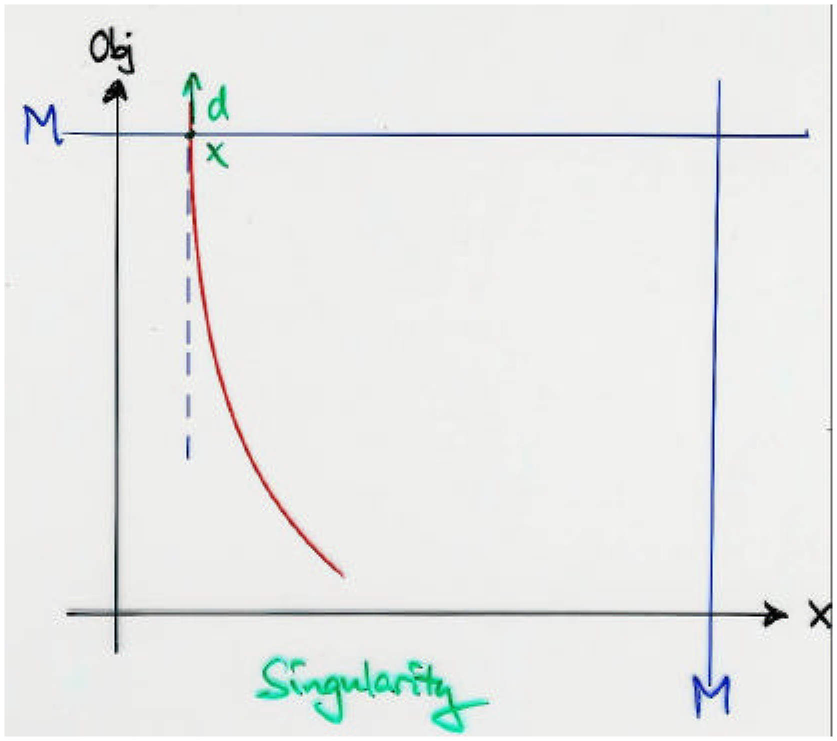
$$\begin{array}{ll} \max & x_2 \\ \text{s.t.} & x_2 = -\log(x_1) \end{array}$$

CONOPT3 terminates with the feasible solution $(x_1,x_2) = (7.3e\text{-}32, 71.688)$. In this point there is a derivative of 1.4e31 which is above an absolute internal limit included to prevent numerical problems. CONOPT3 returns **"Terminated by Solver".**

Mathematically the model is unbounded, but x_2 can not exceed 1000 if we use ordinary floating point arithmetic. So is the model really unbounded?

Independent of the answer to this question, we would like to classify the point better than with **"Terminated by Solver".**


Obj
M
X
M
Fast divergence


Obj
M
d
x
x
Singularity
M

Problem 4: The solver stops and reports **“Cannot Proceed”** or **“Terminated by Solver”** (cont)**.**

This model is similar to the one above:

$$\begin{array}{ll} \max & x_2 \\ \text{s.t.} & x_2 = -\log(x_1+1) \end{array}$$

CONOPT3 returns $(x_1,x_2) = (-1+eps, 36.74)$ where eps is the relative machine precision. The message is **“Slow Convergence”.**

Due to truncation errors in the evaluation of (x_1+1) the objective is piecewise constant just above -1 and the function values are not consistent with the derivatives.

Problem 4: The solver stops and reports **“Cannot Proceed”** or **“Terminated by Solver”** (cont)**.**

Max x_4

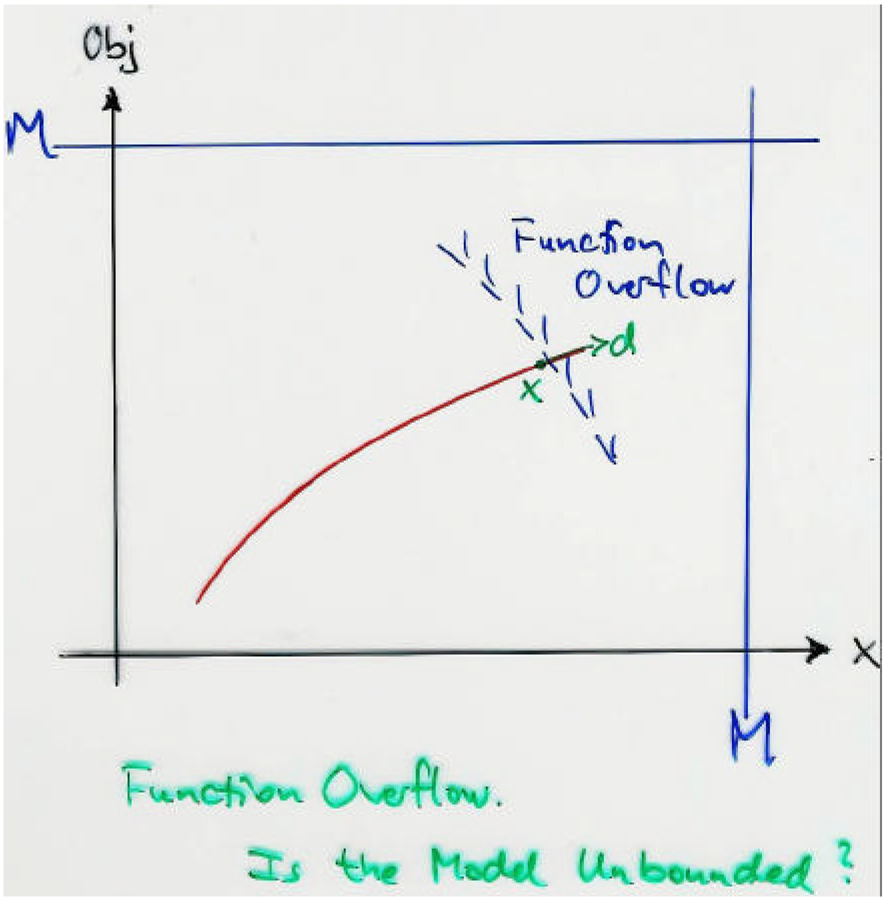
s.t. $x_4 = x_3 – 0.99*x_1 – 0.01*x_2$

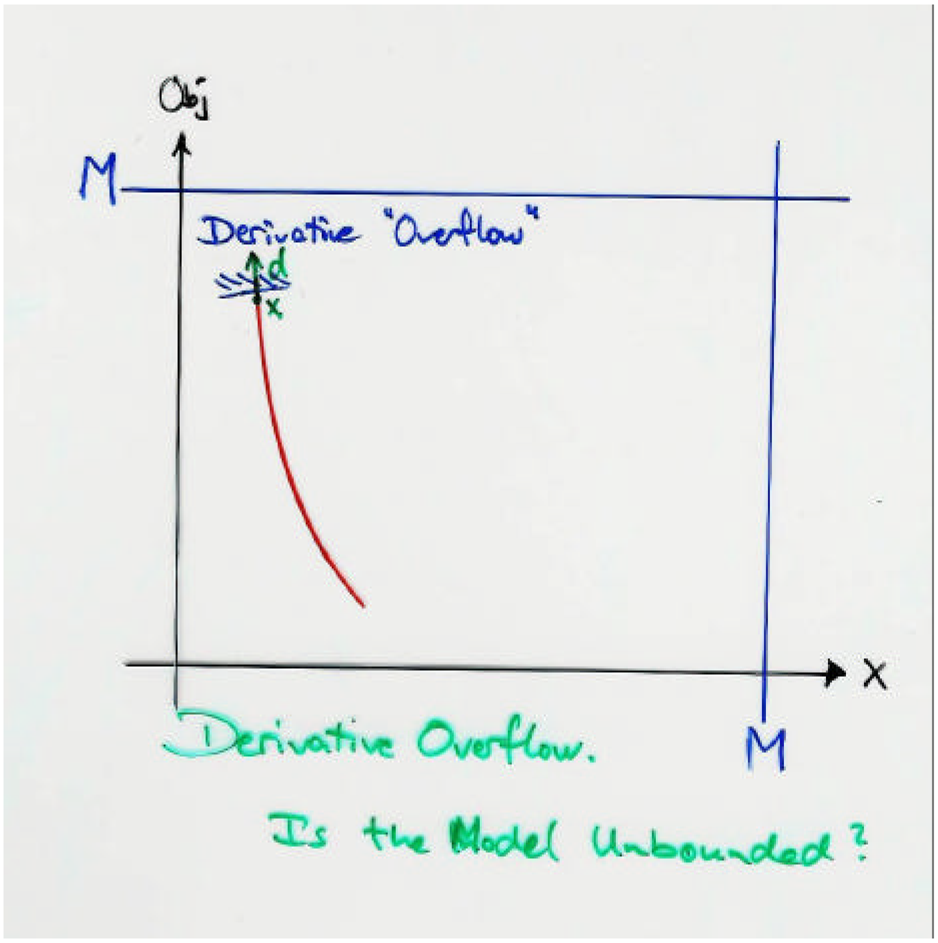
$x_3 = 1.01 *$
$(0.5*x_1**Rho+0.5*x_2**Rho)**(1/Rho)$

The model represents a production function with increasing returns to scale and unbounded inputs available at fixed prices.

For Rho = 5 CONOPT3 stops with (x_1,x_2,x_3,x_4) = (1.0, 1.1e13, 9.4e12, 9.3e12) and the message is **“Slow Progress”.**

The problem is hidden in a message saying “671 cases of x**c overflow”. A small increase in x_2 results in a function evaluation error return code from GAMS’ evaluator instead of the numerical value of the nonlinear function. CONOPT3 cannot improve the objective even though gradients suggest that better solutions exist. But x is still far from ‘Infinity’.


Obj
M
Function Overflow
d
x
X
M
Function Overflow.
Is the Model Unbounded ?


Obj
M
Derivative "Overflow"
d
x
X
M
Derivative Overflow.
Is the Model Unbounded?

What Went Wrong with the Unbounded Models (cont)?

Problem 5: The solver stops on **"Function Evaluation Error Limit"** or something similar.

$$\begin{array}{ll} \max & x_2 \\ \text{s.t.} & x_2 = x_1**(-0.5) \\ & x_1 \geq 0 \end{array}$$

The current distribution of CONOPT3 finds the intermediate feasible point $(x_1,x_2) = (8.7e\text{-}19,1.07e\text{+}9)$. Since x_1 is very close to the lower bound it is set to zero. When CONOPT3 tries to compute the derivative, it is not defined and it stops. The solution that is returned is not even feasible.

This is of course a bug that is being fixed, but it may not be an unusual one.

The Test Models

Family 1: Explicit Power Functions:

$$\begin{array}{ll} \max & x_2 \\ \text{s.t.} & x_2 = x_1{**}p \end{array}$$

for p=-4, -2, -1, -0.5, -0.25, +0.25, +0.5, +1, +2, and +4.

A lower bound of zero on x_1 can be added. It will in general change the behavior of the solver.

Family 1a: Positive Power Functions and Division:

$$\begin{array}{ll} \max & x_2 \\ \text{s.t.} & x_2 = 1/x_3 \\ & x_3 = x_1{**}p \end{array}$$

for p= +0.25, +0.5, +1, +2, and +4.

These models are similar to those in Family 1 with negative p, but the power is well behaved and the ‘problem’ is moved to the division.

The Test Models (cont).

Family 1b: Power Functions with Log and Exp:

$$\begin{array}{ll} \max & x_2 \\ \text{s.t.} & x_2 = \exp(p*\log(x_1)) \end{array}$$

for p = -4, -2, -1, -0.5, -0.25.

These models are again similar to those in Family 1 with negative p, but the 'problem' is this time moved to the log function.

Family 2: Implicitly Defined Power Functions:

$$\begin{array}{ll} \max & x_2 \\ \text{s.t.} & x_2 * x_1{**}(-p) = 1 \end{array}$$

for p=-4, -2, -1, -0.5, -0.25, +0.25, +0.5, +1, +2, and +4.

Family 2 is mathematically equivalent to family 1 but much harder to solve due to the implicit definition of the objective.

The Test Models (cont).

Family 3: Log and Exp:

$$
\begin{array}{lll}
\max & x_2 & \\
\text{s.t.} & x_2 = \log(x_1) & \text{(case 1)} \\
 & x_2 = \exp(x_1) & \text{(case 2)} \\
 & \exp(x_2) = x_1 & \text{(case 3)} \\
 & \log(x_2) = x_1 & \text{(case 4)}
\end{array}
$$

Case 3 is mathematically equivalent to case 1 and case 4 to 2, but 3 and 4 are much harder to solve due to the implicit definition of the objective.

The Test Models (cont).

Family 4: Implicit Objective with Large Hidden Intermediate Terms:

$$\begin{array}{ll} \text{max} & x_3 \\ \text{s.t.} & x_3 = x_1 + x_2 \\ & x_2**(2*p) = x_1**(3*p) \end{array}$$

for p = 1, 2, 3, and 4.

Family 4a: Implicit Objective with Large Explicit Intermediate Terms:

$$\begin{array}{ll} \text{max} & x_3 \\ \text{s.t.} & x_3 = x_1 + x_2 \\ & x_2**(2*p) = x_4 \\ & x_1**(3*p) = x_4 \end{array}$$

for p = 1, 2, 3, and 4.

Family 4a is mathematically equivalent to Family 4, but the large terms inside the nonlinear function is revealed in the variable x4. Many solvers will conclude that the model is unbounded.

The Test Models (cont).

Family 5: Nonbinding Constraint with Overflow:

Max x_2,
s.t. x_2 = -sqr(x_1-100)
exp(x_1) =g= 1

As x_1 approach 100 a slack becomes unbounded and the exp function may overflow. The unboundedness depends on the definition.

Family 6: Production Function with Increasing Returns to Scale and Free Inputs:

Max x_4
s.t. $x_4 = x_3 – 0.99*x_1 – 0.01*x_2$
x_3 = 1.01 *
(0.5*x_1**Rho+0.5*x_2**Rho)**(1/Rho)

for Rho = 1, 3, 5, 7, and 9.

The Test Models (cont)

Family 7: Log with Truncation Problems:

$$\begin{array}{ll} \max & x_2 \\ \text{s.t.} & x_2 = -\log(x_1+p) \end{array}$$

started in $(x_1,x_2) = (1-p,0)$ and solved for $p = 0$ and 1.

<u>Family 8: Logarithmic Spiral:</u>

$$\begin{array}{lll} \text{Max} & x_4 & \\ \text{s.t.} & x_1 = \exp(-x_3) * \cos(x_3) & \\ & x_2 = \exp(-x_3) * \sin(x_3) & \\ & x_4 = 1/\text{sqrt}(\ \text{sqr}(x_1) + \text{sqr}(x_2)\ ) & \text{(case 1)} \\ & x_4 = \quad \text{sqrt}(\ \text{sqr}(x_1) + \text{sqr}(x_2)\ ) & \text{(case 2)} \end{array}$$

(x_1, x_2) follows a logarithmic spiral and x_4 measures the distance from the origin or its inverse. Case 1 has a singularity hidden inside a larger system.

The families 1a, 1b, and 4a are not included in the reported tests.

Results with an Updated CONOPT3

Based on work with these models we have made a number of changes in CONOPT3, mainly:

- Many Bugs have been identified and removed
- More aggressive scaling
- Adjustment of Filtering for small values

The classification of the 38 solves has changed from:

	Normal	Resource Interrupt	Function Errors	Cannot Proceed	Error
Optimal	11				
Infeasible	0				
Unbounded	13				
Intermediate		7	2	5	
Error					

to

	Normal	Resource Interrupt	Function Errors	Cannot Proceed	Error
Optimal	2				
Infeasible	0				
Unbounded	25				
Intermediate		0	1	10	
Error					

Some of the "Cannot Proceed" cases have derivatives above 1.e30. Others stop close to points where functions overflow. The Function Error most be caused by an unidentified bug.

Classification of Unboundedness

So far we have identified at least three types of Unbounded models:

1. Variables outside the objective grow without bound while the objective function grows slowly. Examples: Log(x) or x**p for 0<p<1. May terminate “Locally Optimal”.

2. Variables outside the objective grow without bound while the objective function grows as fast or faster. Examples: exp(x) or x**p for p>1. Are undoubtedly Unbounded and should terminate as such.

3. Variables outside the objective are bounded. The objective function grows close to a singularity. Examples: 1/x or –Log(x).

Models from group 3 (with singularities) are very difficult because the derivatives become extremely large, but the steps are very small and the objective may not grow fast enough to reach large values before numerical problems, including truncation problems, appear. Models with singularities may hide inside models that appear as models from group 2. Example: Family 8.

Summary and Conclusion

- Unboundedness is an overlooked area in NLP and the current state is **BAD**.
- Definitions need more work. Do we also need other classifications than Optimal, Infeasible, and Unbounded? For example: “Boundary of Computable Region” (with a better short name).
- The return information should include the “Proof of unboundedness” (or an Unboundedness Certificate) which include a tangent direction, d.
- Work with unbounded models can help detect bugs that influence ordinary models. The models are good stress tests and it is not a luxury area.
- Filtering of small values or values close to bounds may have to be revised (including interface code to Modeling Systems).

Summary and Conclusion (cont)

- Unboundedness due to singularities is very challenging. Can extremely large gradient be used as indication or proof of unboundedness? Compare Log(x) and Sqrt(x).

- Interface to Models may have to be revised. Domain Errors, as in Log(-1), and Overflow, as in Exp(1000), should not be treated in the same way. Can we classify and systematize Error Return Code?

- Can Overflow be used to classify models as Unbounded? There is overflow both in exp(x) and in sigmoid(x) = exp(x)/(1+exp(x)) for large x, but one is unbounded and the other is not.

- There is a lot of work ahead!