Mathematical Programming Glossary Supplement: Tolerances

Harvey J. Greenberg
University of Colorado at Denver
http://www.cudenver.edu/~hgreenbe/

May 26, 2003

This is to identify tolerances used in mathematical programming and how some relate to each other.

Whenever we base some decision on a comparison of numerical values, we must consider the impurity of those values. Thus, if a decision depends on whether $v = 0$, we instead test if $|v| \leq \tau$, where τ is some small value called a *tolerance*. More generally, for a particular comparison of the form $v = V$, we define two tolerances: *absolute* and *relative*, denoted τ_a and τ_r, respectively. Then, we test if

$$|v - V| \leq \tau_r |V| + \tau_a,$$

where we call V the *referent* value. (Some prefer the symmetry of using $\tau_r(|V| + |v|)$ instead of $\tau_r|V|$.)

This extends to inequality testing by removing the absolute values: $v \leq V$ becomes

$$v - V \leq \tau_r |V| + \tau_a.$$

For example, to test if $v \leq 0$, we test if $v \leq \tau_a$, and to test if $v \leq -1000$, we test if $v + 1000 \leq \tau_r 1000 + \tau_a$.

The use of absolute tolerance, relative tolerance and referent value stems from classical error analysis [5, 3]. Many pairs of tolerances are in ANALYZE [4] for the different decisions made in its procedures, notably its REDUCE command. Other systems use tolerances differently. For example, PCx [1] (Argonne's interior point method) uses a single tolerance for each test. Their tests are of the form:

$$||r|| \leq (1 + ||V||)\tau,$$

where r is a residual and V is a related (referent) value. In particular, to test primal feasibility for the system $Ax = b$, $0 \le x \le U$,

$$r = \begin{pmatrix} Ax - b \\ x - U \end{pmatrix} \text{ and } V = \begin{pmatrix} b \\ U \end{pmatrix}.$$

In relation to the ANALYZE tests, PCx sets absolute and relative tolerances equal, and they test the sum, rather than individual constraints.

There is also a very wide range of default values across systems: from 10^{-3} to 10^{-15}, with the most typical being 10^{-6} to 10^{-8}. Most systems have $\tau_r \le \tau_a$, and many solvers use either $\tau_r = 0$ or $\tau_r = \tau_a$ (some have no separate τ_r). Setting their values requires an understanding of how they are used, particularly in relation to each other. To illustrate, consider the following MILP:

$$\min Z^I = cx + Kz : \; y = Ax + Bz, \; L^x \le x \le U^x, \; L^y \le y \le U^y, \; z \in \{0,1\}^n.$$

Let Z^P be the objective value of the LP relaxation, whose dual is the following:

$$\max Z^D = \lambda^x L^x - \mu^x U^x + \lambda^y L^y - \mu^u U^y - \mu^z \mathbf{1} :$$

$$\begin{array}{rcll} \chi & = & \pi A + \lambda^x - \mu^x & \le \; c \\ \gamma & = & -\pi + \lambda^y - \mu^y & \le \; 0 \\ \kappa & = & \pi B - \mu^z & \le \; K \end{array}$$

$$\lambda^x, \mu^x, \lambda^y, \mu^y \ge 0,$$

where it is understood that a dual variable is absent if a bound is infinite (viz., $\mu^x_j U^x_j \equiv 0$ if $U^x_j = \infty$ and $\lambda^y_i L^y_i \equiv 0$ if $L^y_i = -\infty$).

Table 1 gives a list of decisions and tolerance tests that arise in mathematical programming solvers, but note that tolerances can also depend upon when and why they are being used. For example, primal feasibility tests only the continuous values against their bounds. The tolerance values could be different when executing the simplex method or some preprocessor because the consequences of a "wrong" decision differ. In fact, some algorithms have *dynamic tolerances* to allow more freedom to choose pivot exchanges during an early stage and tightening in the later stages of computation. Also, optimization algorithms, like the simplex method, are fault tolerant — they usually recover if feasibility is lost due to a noisy basis inverse representation. On the other hand, a preprocessing decision could permanently change a status that has a chance of being incorrect.

Decision	Condition	Tolerances	Referent
Primal feasibility	$L^x \le x \le U^x$	τ_r^{xf}, τ_a^{xf}	$L^x,\ U^x$
	$L^y \le y \le U^y$	τ_r^{yf}, τ_a^{yf}	$L^y,\ U^y$
	$0 \le z \le 1$	τ_a^{zf}	n/a
	$z = 0, 1$	τ_i	n/a
Dual feasibility	$\chi \le c$	$\tau_r^{\chi f}, \tau_a^{\chi f}$	c
	$\kappa \le K$	$\tau_r^{\kappa f}, \tau_a^{\kappa f}$	K
	$\gamma \le 0$	τ_0	n/a
	$\lambda, \mu \ge 0$	τ_0	n/a
Primal optimality	$\hat{Z}^P \le Z^P$	$\tau_r^{opt}, \tau_a^{opt}$	β
	$Z^i \le Z^I$	$\tau_r^{opt}, \tau_a^{opt}$	β^i
Dual optimality	$\hat{Z}^D \ge Z^D$	$\tau_r^{opt}, \tau_a^{opt}$	δ
Duality gap	$Z^P - Z^D \le g$	τ_r^g, τ_a^g	g
	$Z^i - Z^I \le G$	τ_r^{Gi}, τ_a^{Gi}	G

Table 1: Standard Tolerances for an Optimizer

A *feasible solution* is an assignment of values to variables (primal or dual) that passes the feasibility test (see table 1). An *optimal solution* is a feasible solution that passes the optimality test. A *near optimal solution* is a feasible solution that passes the duality gap test. These terms apply to the relaxation or the MILP, taken in context.

Special tests: τ_0 is a tolerance for comparing v to 0, regardless of what v is. The test for $v \le 0$ is $v \le \tau_0$, and the test for $v \ge 0$ is $v \ge -\tau_0$. The test for whether v is an integer value also has just one tolerance, τ_i, and the test is whether $v = \lfloor v + \tau_i \rfloor$. Testing for $v = 0$ depends on context. Two tests apply: $v = \lfloor v + \tau_i \rfloor$ and $|v| \le \tau_0$. (Generally, only one test is used.) They could produce different results.

In systems where v could be large, a relative tolerance is used for rounding. For example, $1,000,000.1$ is close enough to $1,000,000$ to round it, but 1.1 is not close enough to 1 to round it. The different decisions result from the relative magnitude of the fractional part:

$$|v - \lfloor v + .5 \rfloor| \le \tau_r |v|.$$

For $\tau_r = 10^{-6}$, $v = 1,000,000.1$ would pass, and v would be rounded to 1,000,000, but $v = 1.1$ would fail, so v would be classified as non-integer.

Added notation: Z^* is the optimal objective value (without error) for the LP relaxation; $\hat{Z}^P$ is the computed primal objective value for some feasible solution to the LP relaxation; $\hat{Z}^i$ is the computed primal objective value for some feasible solution, which also satisfies $z \in \{0,1\}^n$; $\hat{Z}^D$ is the computed dual objective value for some dual feasible solution to the LP relaxation; β is a computed bound for the LP relaxation (so $Z^P \geq \beta$ if $beta$ has no error); and β^i is a computed bound for the MILP (so $Z^I \geq \beta^i$ if $beta^i$ has no error); δ is a computed bound for the dual of the LP relaxation (so $Z^D \leq \delta$ if δ has no error).

Now we shall discuss how some tolerances are related. Near optimality is related to the duality gap:

$$Z^P - Z^D \leq \varepsilon \rightarrow (Z^P \leq Z^* + \varepsilon \text{ and } Z^D \geq Z^* - \varepsilon).$$

For notational convenience, having made the point that $x \leq U^x$ might be tested differently from whether $y \leq U^y$, let us simplify the MILP to have fewer data objects:

$$\min cx + Kz\colon\ Ax + Bz = b, x \geq 0, z \in \{0,1\}^n.$$

Now suppose we have a termination rule for LP relaxation that consists of three tests:

Primal feasibility: $|A_i x + B_i z - b_i| \leq \tau_r |b_i| + \tau_a$ for all i, and $z_j \leq 1 + \tau_a$ for all j.

Primal optimality: $\hat{Z}^P - \beta \leq \varepsilon$.

Duality gap: $\hat{Z}^P - \hat{Z}^D \leq g$.

In general, primal optimality and duality gap tests can be redundant, or they can reinforce each other to reduce occurrence of incorrect conclusions. In a numerically pure world, $\beta \geq Z^*$, but suppose $\beta = Z^* + \beta^{Error}$. Passing the primal optimality test implies

$$\hat{Z}^P \leq Z^* + \beta^{Error} + \varepsilon.$$

This test is thus designed to suggest that we are within ε of optimality (corresponds to setting $\tau_r^{opt} = 0$, $\tau_a = \varepsilon$, and assuming $\beta^{Error} = 0$). The duality gap test has a similar implication if we assume $\hat{Z}^D = Z^* + Z^{Error}$:

$$\hat{Z}^P \leq Z^* + (1 + \tau_r^g)|g| + \tau_a^g + Z^{Error}.$$

If we set $g = 0$ (asking for no duality gap), the relation becomes:

$$\hat{Z}^P \leq Z^* + \tau_a^g + Z^{Error}.$$

We reach the following conclusions:

$$\tau_a^g + \beta^{Error} > \varepsilon + Z^{Error} \rightarrow \text{ the primal optimality test dominates;}$$

$$\tau_a^g + \beta^{Error} < \varepsilon + Z^{Error} \rightarrow \text{ the duality gap test dominates;}$$

$$\tau_a^g + \beta^{Error} = \varepsilon + Z^{Error} \rightarrow \text{ the tests are redundant.}$$

We do not know the errors, but we can control the tolerance settings. Suppose we set $\tau_a^g = \tau_a^{opt} = \varepsilon$. Then, if $\beta = \hat{Z}^D$, the tests are redundant, but if we have a bound computed independently of the dual, the two tests combine to give us a stronger test:

$$\text{Terminate if feasible and } \hat{Z}^D \leq Z^* + \varepsilon + \min\{Z^{Error}, \beta^{Error}\}.$$

Now consider some pre-processing tests. Table 2 lists decisions that arise in some tests, which use different tolerances. Each decision shown has an absolute and relative tolerance associated with the tests.

Decision
Levels have changed
Prices have changed
Primal is infeasible
Dual is infeasible
Fix variable
Fix price
Original coefficient is not zero
Computed coefficient is not zero
Pivot coefficient is acceptable

Table 2: Some ANALYZE Decisions that Use Tolerances

The following example is from the ANALYZE *User Guide* [4] to illustrate an important pitfall to avoid. Consider the 2×2 system:

$$\begin{array}{rcrcl} \frac{1}{2}x_1 & + & x_2 & = & 1 \\ x_1 & + & x_2 & = & 2 \end{array}$$

This has the unique solution, $x = (2, 0)$, and it is this uniqueness that causes a problem with greater implications.

In successive bound reduction, the first order tests (i.e., the inexpensive ones) evaluate rows (with matrix stored in column major form) to see if just one row alone can tighten a bound on a variable. Initially, the bounds are the original ones: $L^0 = L = (0, 0)$ and $U^0 = U = (\infty, \infty)$. The first iteration results in the inference that $x_1 \le 2$, from the first equation and the fact that $x_1 \ge 0$. It similarly produces an upper bound, $x_2 \le 1$, so $U^1 = (2, 1)$. Still in iteration 1, the second equation causes the inference, $x_1 \ge 1$, because we already have $x_2 \le 1$ when we get there. Thus, $L^1 = (1, 0)$.

At a general iteration, we have inferred $L_1^k \le x_1 \le 2$ and $0 \le x_2 \le U_2^k$, where $L_1^k < 2$ and $U_2^k > 0$. It is not difficult to show that at the end of iteration k, the inferred bounds are:

$$L^k = (2 - (\tfrac{1}{2})^{k-1},\ 0) \text{ and } U^k = (2,\ (\tfrac{1}{2})^k) \ \text{ for } k = 1, \ldots$$

This converges to the unique solution, but it does not reach it finitely. If the iterations go far enough, the bounds become within a tolerance such that the action is to fix the levels. Specifically, we have:

$$\begin{array}{lll} \text{Fix } x_1 \in [2 - (\tfrac{1}{2})^{k-1},\ 2] & \text{if} \ \ (\tfrac{1}{2})^{k-1} & \le \ \ \tau_r^{\mathit{fix}} 2 + \tau_a^{\mathit{fix}}. \\ \text{Fix } x_2 \in [0,\ (\tfrac{1}{2})^k] & \text{if} \ \ (\tfrac{1}{2})^k & \le \ \ \tau_a^{\mathit{fix}}. \end{array}$$

Let $\tau_r^{\mathit{fix}} = 0$, and take logarithms, so these tests are equivalent to the following:

$$\begin{array}{lll} \text{Fix } x_1 \in [2 - (\tfrac{1}{2})^{k-1},\ 2] & \text{if} \ \ k - 1 & \ge \ \ -\log_2 \tau_a^{\mathit{fix}}. \\ \text{Fix } x_2 \in [0,\ (\tfrac{1}{2})^k] & \text{if} \ \ k & \ge \ \ -\log_2 \tau_a^{\mathit{fix}}. \end{array}$$

Here are some iterations to help visualize the sequences:

k	L_1^k	U_1^k	L_2^k	U_2^k
1	$2 - 1$	2	0	$\frac{1}{2}$
2	$2 - \frac{1}{2}$	2	0	$\frac{1}{4}$
3	$2 - \frac{1}{4}$	2	0	$\frac{1}{8}$
4	$2 - \frac{1}{8}$	2	0	$\frac{1}{16}$

Suppose we iterate to $k = \lceil -\log_2 \tau_a^{\mathit{fix}} \rceil$, at which time we fix x_2. (This occurs before we would fix x_1 because of interval widths: $2 - L_2^k = (\frac{1}{2})^{k-1} < U_1^k = (\frac{1}{2})^k$.) A natural choice is to fix a variable at the midpoint of its interval, so fix

$$x_2 = \tfrac{1}{2}(2 + 2 - (\tfrac{1}{2})^{k-1}) = 2 - (\tfrac{1}{2})^k.$$

Now the first equation has the range $[2 - (\frac{1}{2})^k + (\frac{1}{2})^{k-1}, 2 + (\frac{1}{2})^{k-1}]$. This follows from having $0 \le x_1 \le 2^k$ and $x_2 = 2 - (\frac{1}{2})^k$.

Using the inferred bounds, the minimum value of $(\frac{1}{2}x_1+x_2)$ is $2+(\frac{1}{2})^k$, and a feasibility test compares this with its given lower bound, 2:

$$2 + (\tfrac{1}{2})^k \; - \; 2 \geq \tau_r^{\mathit{inf}} \, 2 + \tau_a^{\mathit{inf}}$$

Suppose $\tau_r^{\mathit{inf}} = 0$, so we infer:

$$(\tfrac{1}{2})^k \geq \tau_a^{\mathit{inf}} \rightarrow \text{ Primal infeasible.}$$

Equivalently,

$$k \geq -\log_2 \, \tau_a^{\mathit{inf}} \rightarrow \text{ Primal infeasible.}$$

This happens if $\tau_a^{\mathit{fix}} \leq \tau_a^{\mathit{inf}}$!

This example highlights two things:

Tolerances are related. The tolerance to fix a variable should strictly exceed the infeasibility tolerance.

Fix a variable judiciously. When having inferred $x_j \in [L_j, U_j]$, such that $U_j - L_j$ is within tolerance of fixing x_j, do so in the following order of choice:

1. If L_j is an original bound, fix $x_j = L_j$;
2. If U_j is an original bound, fix $x_j = U_j$;
3. If $[L_j, U_j]$ contains an integer, p, fix $x_j = p$.
4. If all of the above fail, fix $x_j = \frac{1}{2}(L_j + U_j)$.

Nonlinear solvers use tolerances similarly, but with different default values. For example, CONOPT [2] uses six tolerances, and feasibility uses an absolute tolerance of 4×10^{-10}. Nonlinear functions have properties that introduce additional tolerances, such as computing gradients and functional values iteratively.

References

[1] J. Czycyk, S. Mehrotra and S.J. Wright, *PCx User Guide*, Technical Report OTC 96/01, Argonne National Laboratories, Argonne, IL, 1997.

[2] A. Drud, CONOPT User Manual.

[3] G. Forsythe and C.R. Moler, *Computer Solution of Linear Algebraic Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1967.

[4] H.J. Greenberg, *A Computer-Assisted Analysis System for Mathematical Programming Models and Solutions: A User's Guide for ANALYZE*, Kluwer Academic Publishers, Boston, MA, 1993.

[5] J.H. Wilkinson, *Rounding Errors in Algebraic Processes*, Prentice-Hall, Englewood Cliffs, NJ, 1963.