Testing Linear Algebra Software

Nick Higham
Department of Mathematics
University of Manchester

higham@ma.man.ac.uk

http://www.ma.man.ac.uk/~higham/

THE UNIVERSITY of MANCHESTER

Observations on Testing

Breakdown of lines of code in LAPACK 3.0 (1997):

SRC	353,393
TESTING	308,770
TIMING	104,790
Total	766,953

Dongarra & Stewart (1984):

> In the development of LINPACK, considerable time and effort were spent in designing and implementing a test package. In some cases, the test programs were harder to design than the programs they tested.

Outline

- ▷ What is meant by correctness of a code?
- ▷ Tests based on rounding error analysis.
- ▷ Test matrices.
- ▷ Case histories.

What is Correctness?

In linear algebra

- Does the code compute the right answer in exact arithmetic?
- Does the code compute an "acceptable" answer in floating point arithmetic?

Possible definition:

> "A code is correct if it faithfully represents in some high level language the algorithm to be encoded".

- How is the algorithm represented?
- Is the algorithm an unambiguous specification? E.g., matrix multiplication can be done in many ways.

Testing Methodology

General approach: define an a posteriori measure of success of the computation and evaluate it on selected problems.

- Are we testing the implementation or the underlying algorithm?
- Evaluate error statistics on battery of test problems.
- Attempt to maximize the success measure as a function of the data using an optimization routine (direct search).
  - Much more likely to reveal bugs than random tests.
  - Hard to automate and can be very computationally expensive.

Tests Based on A Priori Error Analysis

- For many methods we have existing error analysis (usual backward).
- Can compare the actual backward error or forward error with the theoretical bounds.

Note: some error bounds are ordering dependent.

Example: recursive summation in natural order,

$$s_n = \sum_{i=1}^{n} x_i.$$

We have

$$|s_n - \widehat{s}_n| \le (|x_1| + |x_2|)(n-1)u + \sum_{i=3}^{n} |x_i|(n-i+1)u + O(u^2).$$

An ordering-independent bound is

$$|s_n - \widehat{s}_n| \le nu \sum_{i=1}^{n} |x_i| + O(u^2).$$

LU Factorization

For LU factorization with partial pivoting

$$\widehat{L}\widehat{U} = A + \Delta A, \qquad \|\Delta A\| \le p(n)\rho_n u\|A\|.$$

Note that p depends on the norm and the method:

- p is cubic for the ∞-norm but quadratic for the M-norm, $\|A\|_M := \max_{i,j} |a_{ij}|$.
- For Cholesky factorization, p is quadratic for the 2-norm but linear for the M-norm.

- $p \to \sqrt{p}$ is often argued to be more realistic.
- Experiments in the *LINPACK Users' Guide* found $\|A - \widehat{L}\widehat{U}\|_1 / \|A\|_1$ behaved like a linear function of n.

A posteriori stability test:

$$\|\Delta A\| \le f(n)u\|A\|,$$

for some "modest" function f.

- LAPACK: 1-norm, $f(n) = 30n$.
- LINPACK: 1-norm, $f(n) = n$.

Linear Systems

Now residual $\neq$ backward error.

Normwise relative backward error of approximate solution y to $Ax = b$, $A \in \mathbb{R}^{n\times n}$:

$$\min\{\,\epsilon : (A + \Delta A)y = b + \Delta b,\ \ \|\Delta A\| \le \epsilon\|A\|, \\ \|\Delta b\| \le \epsilon\|b\|\,\}$$

$$= \frac{\|r\|}{\|A\|\,\|y\| + \|b\|},$$

where $r = b - Ay$.

Componentwise relative backward error:

$$\min\{\,\epsilon : (A + \Delta A)y = b + \Delta b,\ \ |\Delta A| \le \epsilon|A|, \\ |\Delta b| \le \epsilon|b|\,\}$$

$$= \max_i \frac{|r_i|}{(|A||y| + |b|)_i}.$$

LS Problem

LS problem $\min_x \|Ax - b\|_2$, $A \in \mathbb{R}^{m\times n}$, $m \geq n$.

Define

$$\eta(y) := \min\{ \|[\Delta A, \theta\Delta b]\|_F : \|(A + \Delta A)y - (b + \Delta b)\|_2 = \min \}$$

Theorem 1 (Waldén, Karlson & Sun, 1992) *With $r = b - Ay$,*

$$\eta(y) = \begin{cases} \dfrac{\|r\|_2}{\|y\|_2}\sqrt{\mu}, & \lambda_* \geq 0, \\ \left(\dfrac{\|r\|_2^2}{\|y\|_2^2}\mu + \lambda_*\right)^{1/2}, & \lambda_* < 0, \end{cases}$$

where

$$\lambda_* = \lambda_{\min}\Big(AA^T - \mu\frac{rr^T}{\|y\|_2^2}\Big), \quad \mu = \frac{\theta^2\|y\|_2^2}{1 + \theta^2\|y\|_2^2}.$$

Minimum 2-Norm Solution

$A \in \mathbb{R}^{m\times n}$, $m \le n$, problem $\min\{\, \|x\|_2 : Ax = b \,\}$.

The standard method for solution based on QR factorization is normwise backward stable (H, 1996).

Define

$$\mu(y) \;:=\; \min\{\, \|[\Delta A, \theta \Delta b]\|_F : y \text{ is the solution of min. 2-norm to } (A+\Delta A)y = b + \Delta b.$$

Sun & Sun (1995) show that

$$\mu(y) = \sqrt{\frac{\theta^2\|y\|_2^2}{1+\theta^2\|y\|_2^2}\cdot\frac{\|r\|_2^2}{\|y\|_2^2} + \sigma_m^2\left(\left(I - \frac{yy^T}{y^Ty}\right)A\right)}.$$

Eigenproblem

Definitions of backward error less clear-cut.

Consider Schur decomposition of $A = QTQ^* \in \mathbb{R}^{n \times n}$. Natural tests are that

$$\|\widehat{Q}^*\widehat{Q} - I\|, \quad \frac{\|A - \widehat{Q}\widehat{T}\widehat{Q}^*\|}{\|A\|}$$

are small. (LAPACK requires them $\leq 20nu$, for the 1-norm)

Note: standard error analysis says there exists unitary Q such that $\|A - Q\widehat{T}Q^*\| \leq p(n)u\|A\|$.

A definition of backward error for $\widehat{T}$ alone is

$$\min\{\, \|A - Q\widehat{T}Q^*\| : Q^*Q = I \,\}.$$

Open problem to evaluate this.

Forward Error Tests

For a linear system $Ax = b$, we could compute $\|x - \widehat{x}\|/\|x\|$. Requires

- computation of an exact, or "sufficiently accurate", solution x,
- computation or estimation of a condition number, to obtain theoretical error bound.

Forward error tests appropriate for

- Algorithms that are forward stable but not backward stable.
- Testing forward error bounds computed by a code.

LAPACK tests both backward error and forward error of LU factorization routines.

“Known Solution” Trick

Old technique:

- Choose x, let $b := Ax$.
- Solve $Ax = b$.
- Compute the “error” $\|x - \widehat{x}\|/\|x\|$.

We actually compute

$$\widehat{b} = fl(Ax) = Ax + \Delta b, \qquad |\Delta b| \le \gamma_n |A||x|,$$

where $\gamma_n = nu/(1 - nu)$. Thus

$$|x - A^{-1}\widehat{b}| \le \gamma_n |A^{-1}||A||x|.$$

Conclude: this approach is safe.

Test Matrices

- Many nonrandom test matrices devised and explored since the 1950s.
- Classic example is the Hilbert matrix, $h_{ij} = 1/(i+j-1)$. Actually a poor test matrix!
- Of more general use for the testing are random matrices and matrices from applications.

To form matrices with known SVD (say): choose random orthogonal U and V, singular values σ_i, and form $A := U \operatorname{diag}(\sigma_i) V^T$. Set `[U, R] = qr(randn(n))` for genuinely random U.

Sources of test matrices:

- MATLAB: Test Matrix Toolbox (Higham, 1989–).
- Harwell–Boeing collection (Duff, Grimes and Lewis, 1989–).
- Matrix Market (Boisvert, 1995–): `http://math.nist.gov/MatrixMarket/`

Generating Random Orthogonal

Stewart (1980): Let the $x_i \in R^i$ be independent vectors from the Normal (0,1) distribution. Form

$$Q = DP_1P_2 \dots P_{n-1},$$

where $P_i = \text{diag}(I_{n-i}, \overline{P}_i)$, where $\overline{P}_i$ is the Householder transformation that reduces x_i to $r_{ii}e_1$ and $D = \text{diag}(\text{sign}(r_{ii}))$.

Then Q is from the Haar distribution.

LAPACK Test Matrix Generators

LAPACK has test matrix generators in

`LAPACK/TESTING/MATGEN`

They are *not* documented in the Users' Guide.

`xLATME` (21 parameters) generates random nonsymmetric square matrix with specified eigenvalues.

- User may specify $D = \mathrm{diag}(\lambda_i)$ or it can be computed from one of six different distributions.
- The upper triangle of D can optionally be filled with random numbers.
- A is formed as XDX^{-1}, where $X = U\Sigma V^T$, with U, V random orthogonal matrices and Σ a diagonal matrix of singular values chosen in an analogous way to D.

xLATMS, xLATMR

`xLATMS` (16 parameters) generates random matrix with specified singular values.

- Matrix of singular values specified by user or chosen from one of six different distributions.
- A is formed as UDV^T, where V is random orthogonal.
- Bandwidth optionally reduced to specified value by further unitary transformations (possibly combined with previous step).

`xLATMR` (28 parameters) generates random matrix.

- Elements of A formed from one of three distributions.
- Symmetric or nonsymmetric.
- Optionally graded (by row and/or column), banded, "sparsified".
- Bandwidth optionally reduced to specified value.

Case History: MATLAB's `rcond`

The problem: approximate $\kappa(A) = \|A\|\|A^{-1}\|$ cheaply, given some factorization of the matrix.

Two types of condition number estimator:

- statistically based estimators that use random numbers,
- non-statistical estimators, as in LINPACK and LAPACK.

Difficult to test the latter, as they can fail. For the former we can test distribution of estimates.

MATLAB's `rcond` is an upper bound for $\kappa_1(A)^{-1}$; direct translation of the LINPACK estimator.

Release Notes for MATLAB 4.1: "This release of MATLAB fixes a bug in the `rcond` function. Previously, `rcond` returned a larger than expected estimate for some matrices ... `rcond` now returns an estimate that matches the value returned by the Fortran LINPACK library."

Case History: LAPACK's Symmetric Indef. Solver

To solve a symmetric indefinite linear system $Ax = b$, $A \in \mathbb{R}^{n\times n}$, compute

$$PAP^T = LDL^T,$$

where P is a permutation matrix, L is unit lower triangular and $D = \mathrm{diag}(D_{ii})$, D_{ii} 1×1 or 2×2.

LAPACK uses the partial pivoting strategy of Bunch and Kaufman. For this strategy, $\|L\|_\infty/\|A\|_\infty$ can be *arbitrarily large*, even though the factorization itself is backward stable.

The LAPACK 2.0 implementation can be unstable when $\|L\|_\infty$ is large (Ashcraft, Grimes & Lewis, 1995). Due to replacing a symmetric rank-2 update by two rank-1 updates, via the use of an eigendecomposition.

Instability not detected by LAPACK test software; so far seen only on examples specially constructed.

Case History: LAPACK/LINPACK Matrix Inversion

LAPACK 2.0 tests the general matrix inversion routine by evaluating the ratio

$$\frac{\|A\widehat{X} - I\|_1}{n\, u\, \kappa_1(A)}.$$

However, it is the ratio

$$\frac{\|\widehat{X}A - I\|_1}{n\, u\, \kappa_1(A)}$$

that is bounded independently of A.

The incorrect test has apparently never been failed.

Precisely the same error is present in the LINPACK test software.

Left Residual versus Right Residual