Linear systems stability: condition number of a matrix

Numerical Linear Algebra 200919

When solving linear systems, roundoff will play a role. We want to investigate the accuracy of the computed solution.

A matrix $A \in \mathbb{R}^{n\times n}$ is **ill-conditioned** if relatively small changes in A (if is rounded for example) can cause large changes in the solution of $Ax = b$ (for a fixed $b \in \mathbb{R}^n$). Otherwise it is well-conditioned. For a given matrix norm $\|\cdot\|$, the **accuracy** of the solution depends on the **condition number** $k(A) = \|A\|\|A^{-1}\|$.

In general, we have approximations of A and b, say $A+\delta A$ and $b+\delta B$, and we obtain a solution $\hat{x} = x + \delta x$. Recall that (backward error analysis formula):

$$\frac{\|\delta x\|}{\|\hat{x}\|} \leq k(A)\left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|A\|\|\hat{x}\|}\right).$$

---

1. There are ill-conditioned matrices of relatively small dimensions.

   (a) Consider the Vandermonde matrix of order n

   $$V_n = \begin{pmatrix} 1 & 1 & \dots & 1 \\ \alpha_1 & \alpha_2 & \dots & \alpha_n \\ \vdots & \vdots & \dots & \vdots \\ \alpha_1^{n-1} & \alpha_2^{n-1} & \dots & \alpha_n^{n-1} \end{pmatrix}.$$

   Let $\alpha_j = 1 - 2(j-1)/(n-1)$, $j = 1, ..., n$. Investigate how $\kappa_2(V_n)$ behaves as a function of n (consider values of $n \leq 50$).

   (b) Check numerically Szegö's result $k_2(H_n) \sim e^{3.5n}$ for the Hilbert matrices $H_n(i,j) = 1/(i+j-1)$, $1 \leq i, j \leq n$.

2. Investigate the relation between the condition number and the precision of the solution when solving a linear system.

   Consider the n-dimensional Frank matrix

   $$A_{i,j} = \begin{cases} 0 & \text{for } j < i-2, \\ n+1-i & \text{for } j = i-1, \\ n+1-j & \text{for } j \geq i. \end{cases}$$

   Assume that $A_{i,j}$ are stored as float (double precision) numbers. Let $x_0 = (1, \dots, 1)$ and $b = Ax_0$. Note that b contains roundoff errors of the order of the precision machine ($\epsilon_{\text{machine}} \sim 10^{-16}$). For $n = 2, \dots, 24$ compute $k_2(A)$ and compare the values of $\epsilon_{\text{machine}} k_2(A)$ with those of the relative error of the solution, that is

   $$e_{\text{rel}}(x) = \frac{\|x - x_0\|_2}{\|x_0\|_2}.$$

   Explain the results using the backward error analysis formula.

3. Moral: It is inadvisable to actually compute A^{-1}

Imagine we want to solve $Ax = b$, $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$. We compare two ways: 1) $x = A^{-1}b$ and 2) using GEPP (Gauss elimination with partial pivoting). Assume that for 1) we do the things properly, that is, for example we compute the inverse A^{-1} solving n-linear systems using GEPP. We want to see that, even if A^{-1} is computed exactly,

- Cost: 1) $2n^3$ flops, 2) $2n^3/3$ flops (3 times faster!).
- Stability: 1) is less numerically stable than 2)

To this end, perform the following numerical experiment. Consider fifty 25×25 linear systems $Ax = b$ (and repeat for 10^4 systems of dimension 50 for example) of the following form:

i) A is random with $k_2(A) = \epsilon_{\text{machine}}^{-1/2} \approx 9 \times 10^7$, and
ii) the elements of x are taken from the standard Gaussian distribution.

For each system

(a) compute the solution using the 1) "matrix inverse method" and 2) the LU method using GEPP,
(b) compute the norm-wise relative backward error

$$\eta_{A,b} = \frac{\|r\|}{\|A\|\|x\| + \|b\|}, \qquad r = Ax - b \text{ (the residual)}$$

Write a program to compute the minimum and the maximum of $\eta_{A,b}$ obtained for each method. Also compare both methods in terms of the computation time as a function of the dimension (e.g. up to $n = 100$ and solving 10^3 systems for each n).

Remark: To generate random matrices with $k_2(A) = \alpha > 1$ do the following:

i) generate a random matrix B (e.g. with standard Gaussian coefficients),
ii) perform the so-called QR decomposition of B, then $B = QR$,
iii) consider $A = Q^t DQ$, where D is diagonal matrix $D = [\alpha, 1, ..., 1]$.

The matrix A has the desirable condition number $k_2(A)$.

4. A method to estimate the condition number $k_1(A)$

We have seen that it is inadvisable (and usually unnecessary!) to actually compute A^{-1}. In particular, the computation of $k(A)$ for large matrices involves a large number of operations. Then, how can we effectively compute the condition number of a matrix? Techniques have been developed to approximate $\|A^{-1}\|_1$ without computing the inverse of the matrix A, that is, the condition number of a nonsingular matrix is estimated by $k_1(A) = \|A\|_1\omega$ where ω is an estimate of $\|A\|_1^{-1}$.

(a) Write a code that implements Hager's algorithm (from Demmel's book, p.53) to estimate the 1-norm of a matrix (computes ω such that $\omega \leq \|B\|_1$, and typically a) $\omega \approx \|B\|_1$ and b) it stops after 2 iterations)

ALGORITHM 2.5. *Hager's condition estimator returns a lower bound* $\|w\|_1$ *on* $\|B\|_1$*:*

choose any x *such that* $\|x\|_1 = 1$ /* *e.g.* $x_i = \frac{1}{n}$ */
repeat
  $w = Bx$, $\zeta = \text{sign}(w)$, $z = B^T\zeta$ /* $z^T = \nabla f$ */
  if $\|z\|_\infty \leq z^T x$ *then*
    return $\|w\|_1$
  else
    $x = e_j$ *where* $|z_j| = \|z\|_\infty$
  endif
end repeat

Remark. $\text{sign}(x) = (s_1, ...s_{,n})$ where $s_i = 1$ if $x_i \geq 0$ and $s_i = -1$ otherwise.

(b) Modify the previous algorithm to obtain an estimate of $\|A^{-1}\|_1$. Observe that if $B = A^{-1}$ then $w = Bx$ and $z = B^T\eta$ can be rewritten as the linear systems $Aw = x$ and $A^T z = \eta$, then apply GEPP/LU to solve these systems. [1]

Check the algorithm by recording the ratio between the estimated condition number and the direct computation of the condition number (inverting the matrix A and computing the norms) for 1000 random matrices A having $10^3 \leq k_2(A) \leq 10^{16}$ of dimensions $10 \leq n \leq 90$.

[1] An improvement by Higham of this algorithm is the subroutine **slacon** in LAPACK. Also **condest** in MATLAB performs a similar computation. There are other methods for very large systems.