Institutionen för datavetenskap
Umeå universitet

18 november 2010
Numeriska metoder för civilingenjörer

Solutions to review exercises for quiz

1 Theme 1

1. Machine epsilon ϵ_M is the distance between the number 1 and the next floating point number. (*Warning:* in this course, we use Matlab's definition of machine epsilon. Another common definition is the one used in Wikipedia's machine epsilon article. Wikipedia's definition yields a machine epsilon that is $\frac{1}{2}\epsilon_M$).

2. $|x - fl(x)| \le \frac{1}{2}\epsilon_M |x|$. For, $\epsilon_M = 2^{-52}$, this estimate yields the bound $|\pi - fl(\pi)| \le 2^{-53}\pi \approx 3.5 \times 10^{-16}$. (Tighter bounds can be given!)

3. The appropriate test is to check whether $|f(x) - a| \le \tau$ (in Matlab, `abs(f - a) <= tau`), where $\tau > 0$ is a small number.

4. The discretization error usually dominates.

5. (i) Problems that are sensitive to changes in input data, for instance the solution of linear systems with almost singular (ill-conditioned) matrices. (ii) When numerically unstable algorithms are used, for instance Gaussian elimination without row pivoting.

6. Underflow occurs when the result of a floating point calculation yields a number of magnitude less than what is representable as a normalized number in the floating point system. Attempting the operations `0/0` and `Inf-Inf`, for instance, will result in `NaN`.

7. Cancellation of significant digits occur when subtracting two numbers that almost are the same. Examples: (i) the calculation $\sqrt{1+x^2} - \sqrt{1-x^2}$, (ii) the calculation of a finite difference $\big(f(x+h) - f(x)\big)/h$

8. The *discretization error* will dominate for large values of h, whereas the *rounding error* will dominate for small values of h.

9. *Short explanation (sufficient!)*: the partial sums $S_N = \sum_{k=1}^{N} 1/k$ eventually become large enough so that next term $1/(N+1)$ vanishes in the roundoff.

   A little longer explanation:

   $$S_{N+1} = S_N + \frac{1}{N+1} = S_N\left(1 + \frac{1}{S_N(N+1)}\right)$$

   By definition, the next larger floating point number after 1 is $1 + \epsilon_M$. Thus, the above right-hand side will be rounded to S_N when N is so large that

   $$\frac{1}{S_N(N+1)} < \frac{1}{2}\epsilon_M,$$

   and the sum will stall at S_N.

2 Theme 2

1. `x = B\(2*A + eye(n))*(C\b + A*b);`

2. LU factorization takes about $\frac{2}{3}n^3$ flops. Thus, the time per floating point operation is $t_f = T/(\frac{2}{3}n^3)$, where T is the elapsed time. Forward and backward substitution takes n^2 flops each, which yields the elapsed time

$$T_{\text{fb}} = n^2 t_f = n^2 \frac{T}{\frac{2}{3}n^3} = \frac{3T}{2n} = \frac{3\cdot 11}{2\cdot 5000} = 3.3 \text{ ms},$$

for either of the operations. (In reality, they will take slightly longer time due to startup times.) Note how much faster the forward and back substitutions are compared to the factorization!

3. When writing `A\b`, the linear system will be solved using Gaussian elimination, which takes less floating point operations than to explicitly compute the inverse matrix and then perform the matrix–vector multiplication `inv(A)*b`.

4. (i) LU factorize the matrix once and for all (takes about $\frac{2}{3}n^3$ floating point operations, where n is the order of the matrix). (ii) Perform forward and back substitutions for each right hand side. These require $2n^2$ floating point operations per right-hand side. (The costly factorization step will be performed only once when using this strategy; compare the timings from question 4!)

5. $A = LU$ yields $A^T = U^T L^T$. The equation $A^T x = b$ can thus be written $U^T L^T x = b$ and be solved by solving the two following triangular system in sequence:

$$\begin{aligned} U^T y &= b, \\ L^T x &= y. \end{aligned}$$

6. No. The condition number ($\kappa(A) = \|A^{-1}\|\|A\|$) is a property of the matrix itself, independent of which algorithm that is used to factorize it.

7. $\|A\|_\infty = 8.5$ (largest 1 norm of the row vectors), $\|A\|_1 = 6.5$ (largest 1 norm of the column vectors). To compute $\|A\|_2$, form matrix $S = A^T A$, which will be symmetric (and positive semidefinite). Then $\|A\|_2$ will be the square root of the largest eigenvalue of S.

8. Matrix 1: ill conditioned ($\kappa = 10^{20}$). Matrices 2 and 3: well conditioned ($\kappa = 1$). Matrix 4: ill conditioned (the columns are linearly dependent, so the matrix is singular with $\kappa = +\infty$).

9. (a) Making the ansatz $A = LU$ with

$$L = \begin{pmatrix} 1 & 0 \\ l_{11} & 1 \end{pmatrix}, \quad U = \begin{pmatrix} u_{11} & u_{12} \\ 0 & u_{22} \end{pmatrix},$$

yields that

$$LU = \begin{pmatrix} u_{11} & u_{12} \\ l_{11}u_{11} & l_{11}u_{12} + u_{22}. \end{pmatrix}$$

Identification of the $(1,1)$- and $(2,1)$ elements in A and LU yields the equations $u_{11} = 0$ and $l_{11}u_{11} = 1$, which have no solution.

   (b) Row pivoting.

10. (a)

$$A = \begin{pmatrix} 1 & 0.5 & 1.5 & -1 \\ 2 & 3 & 2 & -2 \\ 0 & 2 & 1 & 0 \\ 0 & 4 & 2 & 2 \end{pmatrix} \begin{matrix} \textcircled{\scriptsize -2} \\ \hookleftarrow \\ \\ \\ \end{matrix} \sim \begin{pmatrix} 1 & 0.5 & 1.5 & -1 \\ 0 & 2 & -1 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 4 & 2 & 2 \end{pmatrix} \begin{matrix} \\ \textcircled{\scriptsize -1} \\ \hookleftarrow \; \textcircled{\scriptsize -2} \\ \hookleftarrow \end{matrix}$$

$$\sim \begin{pmatrix} 1 & 0.5 & 1.5 & -1 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 4 & 2 \end{pmatrix} \begin{matrix} \\ \\ \textcircled{\scriptsize -2} \\ \hookleftarrow \end{matrix} \sim \begin{pmatrix} 1 & 0.5 & 1.5 & -1 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}.$$

The coefficients used in the elementary row operations, with the opposite sign, form the elements in the under triangle of the L matrix. Thus,

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 2 & 2 & 1 \end{pmatrix}, \qquad U = \begin{pmatrix} 1 & 0.5 & 1.5 & -1 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}.$$

Check:

$$LU = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 2 & 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0.5 & 1.5 & -1 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 0.5 & 1.5 & -1 \\ 2 & 3 & 2 & -2 \\ 0 & 2 & 1 & 0 \\ 0 & 4 & 2 & 2 \end{pmatrix} = A$$

(b) The L factors can be greater than 1 if pivoting is not performed (like in the example above!), which can cause numerical instability through successive amplification of rounding errors. There is also a risk for division by zero if row pivoting is not performed.

11. (a)

$$A = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 10 \end{pmatrix} \begin{matrix} \textcircled{\scriptsize -2}\,\textcircled{\scriptsize -3} \\ \hookleftarrow \quad \\ \quad \hookleftarrow \end{matrix} \sim \begin{pmatrix} 1 & 4 & 7 \\ 0 & -3 & -6 \\ 0 & -6 & -11 \end{pmatrix} \begin{matrix} \\ \textcircled{\scriptsize -2} \\ \hookleftarrow \end{matrix},$$

$$\sim \begin{pmatrix} 1 & 4 & 7 \\ 0 & -3 & -6 \\ 0 & 0 & 1 \end{pmatrix}$$

which yields

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \end{pmatrix}, \qquad U = \begin{pmatrix} 1 & 4 & 7 \\ 0 & -3 & -6 \\ 0 & 0 & 1 \end{pmatrix}.$$

$Ax = LUx = b$ with $b = (1,1,1)^T$. First solve $Ly = b$:

$$\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \Rightarrow \begin{matrix} y_1 = 1 \\ y_2 = 1 - 2y_1 = -1 \\ y_3 = 1 - 3y_1 - 2y_2 = 0 \end{matrix} \quad ,$$

and then $Ux = y$:

$$\begin{pmatrix} 1 & 4 & 7 \\ 0 & -3 & -6 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} \Rightarrow \begin{matrix} x_1 = 1 - 4x_2 - 7x_3 = -1/3 \\ x_2 = (-1 + 6x_3)/(-3) = 1/3, \\ x_3 = 0, \end{matrix}$$

that is $x = (-1/3, 1/3, 0)^T$.

(b) The error estimate

$$\frac{\|x-\tilde{x}\|}{\|x\|} \leq \kappa(A)\frac{\|b-\tilde{b}\|}{\|b\|}$$

holds for systems $Ax = b$ and $A\tilde{x} = \tilde{b}$, with arbitrary vector norm and associated matrix norm. We know that $\|A^{-1}\|_\infty = 7$ and we can read off $\|A\|_\infty = 19$ ($\|A\|_\infty$ is the largest 1 norm of any row vector in the matrix), which yields $\kappa_\infty(A) = 133$. We are also given that $\|b-\tilde{b}\|_\infty = 5\times 10^{-4}$, and it holds that $\|b\|_\infty = 1$, $\|x\|_\infty = 1/3$. Thus,

$$\|x-\tilde{x}\|_\infty \leq \kappa_\infty(A)\frac{\|b-\tilde{b}\|_\infty}{\|b\|_\infty}\|x\|_\infty = 133\cdot 0.0005\cdot 1/3 \approx 0.0222,$$

(that is, an error in the second decimal!).

3 Theme 3

1. Let $\mathbf{e}_k = \mathbf{x}_k - \hat{\mathbf{x}}$, where $\hat{\mathbf{x}}$ is the exact solution. If

$$||\mathbf{e}_{k+1}|| \sim C||\mathbf{e}_k||,$$

where $0 < C < 1$, then the sequence $\mathbf{x}_k$ is said to converge linearly with convergence rate C. (The precise definition is that the convergence is linear if there is a constant $0 < C < 1$ such that $\lim_{k\to\infty}\|\mathbf{e}_{k+1}\|/\|\mathbf{e}_k\| = C$.)

2. (a) quadratic; (b) linear with rate constant 10^{-2}.

3. Newton's method for solving equation $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ can be written $\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{J}(\mathbf{x}_k)^{-1}\mathbf{f}(\mathbf{x}_k)$. For linear $\mathbf{f}(\mathbf{x})$, it holds that $\mathbf{f}(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$ for a matrix $\mathbf{A}$ and a vector $\mathbf{b}$. Thus, the Jocobian is here is $\mathbf{J} = \mathbf{A}$ (independent of $\mathbf{x}$). Newton's metod then becomes

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{A}^{-1}(\mathbf{A}\mathbf{x}_k - \mathbf{b}) = \mathbf{A}^{-1}\mathbf{b},$$

so Newton's metod finds the solution to the equation $\mathbf{A}\mathbf{x} = \mathbf{b}$ in one step, regardless of starting guess.

4. Advantage, fixed-point iterations: no linear system to solve, no Jacobian calculation needed. Advantage, Newton's method: fast (quadratic) local convergence.

5. At local minima $\mathbf{x}_*$ of f, it holds that all partial derivatives of f vanish,

$$\frac{\partial f}{\partial x_i} = 0, \text{for } i = 1, \ldots, n,$$

that is, the gradient of f vanishes at $\mathbf{x}_*$, $\nabla f(\mathbf{x}_*) = \mathbf{0}$. The condition $\nabla f(\mathbf{x}_*) = \mathbf{0}$ is a nonlinear system of equations in $\mathbf{x}_*$. The Jacobian of the gradient ∇f is the Hessian matrix $\mathbf{H}$ with components

$$H_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}.$$

Newton's method then becomes

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \mathbf{H}(\mathbf{x}_n)^{-1}\nabla f(\mathbf{x}_n).$$