Fall 2017: Numerical Methods I
Assignment 2 (due Oct. 5, 2017)

The same rules as for the previous assignment hold. For this assignment, you are asked to hand in code for some of the problems. Please try to produce well-written code following the suggestions on the previous assignment. This assignment has (and future assignments will have) an extra credit problem. I recommend that you only work on those once you are done with the regular problems. Extra credit problems are supposed to encourage exploration and are usually worth less than regular problems.

1. **[Convergence order, 1+2+2pt]** We estimate the convergence order of a value u_h that depends on $h > 0$, as $h \to 0$. Often, h is a step size parameter, and we expect that $u_h \to u$ as $h \to 0$.

   (a) If we know the limit value u, we can estimate the order p using the ansatz

   $$u_h - u = Ch^p + O(h^{p+1}). \tag{1}$$

   Use (1) to show that p can be estimated from

   $$\log_2 \left| \frac{u_h - u}{u_{h/2} - u} \right| = p + O(h), \tag{2}$$

   where $\log_2$ denotes the 2-based logarithm.

   (b) If one does not know the limit u, one way to estimate the convergence rate is to replace u by the best available approximation and then use (2). As an alternative, one can use approximations with successively halved values for h. Show that:

   $$\left| \frac{u_h - u_{h/2}}{u_{h/2} - u_{h/4}} \right| = 2^p + O(h).$$

   (c) Try both methods to estimate the convergence rate of the trapezoidal rule to approximate the integral $\int_a^b f(t)\,dt$. The trapezoidal rule approximation is given by:

   $$u_h = \frac{h}{2} f(a) + h \sum_{j=1}^{N-1} f(a + jh) + \frac{h}{2} f(b),$$

   where $h = (b-a)/N$, for $f = \exp(x)/(1 + 4x^2)$ and interval bounds $a = 0, b = 4$.

2. **[Floating points, O/o-notation, convergence, 1+2+2+1pt]**

   (a) How many IEEE double precision numbers are between an adjacent pair of nonzero single precission floating point numbers?

   (b) Consider

   $$\alpha_n := \frac{2n^2 + 4n}{n^2 + 2n + 1} \quad \text{for } n = 1, 2, \ldots$$

   Show that $\alpha_n = \alpha + O(\frac{1}{n^2})$ as $n \to \infty$, where $\alpha = \lim_{n\to\infty} \alpha_n$. What is the convergence order of

   $$\alpha'_n := \frac{2n^2 + 3n}{n^2 + 2n + 1}?$$

(c) Some of the following statements are true, some false. Prove them or give a counterexample.

- $\sin(x) = O(x^2)$ as $x \to 0$.
- $1 - \cos(x) = O(x)$ as $x \to 0$.
- If $\lim_{x\to a} |g(x)| < \infty$, then $O(f(x)g(x)) = O(f(x))$ as $x \to a$.
- $O(n^2) = o(n^2)$ as $n \to \infty$.

(d) At what rate does the sequence

$$x_n := \sin\left(\frac{5}{n}\right)$$

converge to zero as $n \to \infty$?

3. **[Orthogonal matrices and matrix condition numbers, 2+1+2pt]**

(a) Let $Q \in \mathbb{R}^{n\times n}$ be orthogonal, i.e., $Q^TQ = I$. Show that $\|Q\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|_2$. Does the same hold for the norms $\|\cdot\|_1$ and $\|\cdot\|_\infty$ as well? Prove or give a counterexample.

(b) Show that for an orthogonal matrix Q holds $\|Q\|_2 = 1$ and $\kappa_2(Q) = 1$, where κ_2 is the condition number with respect to the $\|\cdot\|_2$-norm.

(c) Let $A \in \mathbb{R}^{n\times n}$ be invertible. Let $\boldsymbol{b} \in \mathbb{R}^n\backslash\{\boldsymbol{0}\}$, and $A\boldsymbol{x} = \boldsymbol{b}$, $A\boldsymbol{x}' = \boldsymbol{b}'$ and denote the perturbations by $\Delta\boldsymbol{b} = \boldsymbol{b}' - \boldsymbol{b}$ and $\Delta\boldsymbol{x} = \boldsymbol{x}' - \boldsymbol{x}$. We have shown that

$$\frac{\|\Delta\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} \le \kappa_2(A)\frac{\|\Delta\boldsymbol{b}\|_2}{\|\boldsymbol{b}\|_2}\,.$$

Show that this inequality is *sharp*, that is, find vectors $\boldsymbol{b}, \Delta\boldsymbol{b}$ for which equality holds above.[1]

4. **[Properties of LU decomposition, 1+2pt]**

(a) Give an example of an invertible 3×3 matrix that does not have zero entries, for which the LU decomposition without pivoting fails.

(b) Show that the LU factorization of an invertible matrix $A \in \mathbb{R}^{n\times n}$ is unique. That is,

$$A = LU = L_1U_1$$

with upper triangular matrices U, U_1 and lower triangular matrices L, L_1 that have 1's in the diagonal necessarily implies that $L = L_1$ and $U = U_1$. *Hint:* Show first that the set $\mathcal{L}$ of lower triangular matrices with 1's on the diagonal is a subgroup of the group of invertible matrices, i.e., $\mathcal{L}$ contains the identity matrix, the product of two elements in $\mathcal{L}$ is again in $\mathcal{L}$, and the inverse of an element in $\mathcal{L}$ is also in $\mathcal{L}$.

5. **[Forward substitution, 5pt]** We implement forward substitution to solve the triangular system

$$L\boldsymbol{x} = \boldsymbol{b},$$

where $L \in \mathbb{R}^{n\times n}$ is an invertible, lower triangular matrix, and $\boldsymbol{x}, \boldsymbol{b} \in \mathbb{R}^n$. Write functions that take L and $\boldsymbol{b}$ as input arguments and return the solution $\boldsymbol{x}$, and hand in code listings with your assignment. The programs should check if L is square and if its size is compatible with the size of b.

- Implement a naive forward substitution `x = forward1(L,b)` by looping over rows (outer loop) and over columns (inner loop).

[1] Hint: consider the eigenvectors of A^TA.

- Implement forward substitution `x = forward2(L,b)` by looping over the rows and using vector operations in each row[2], i.e.:

$$x_1 = b_1/L_{11},$$
$$x_i = \left(b_i - L_{i,1:i-1}\boldsymbol{x}_{1:i-1}\right)/L_{ii} \text{ for } i = 2, 3, \ldots, n,$$

  where, following MATLAB's notation, $k : l = (k, k+1, \ldots, l)$ such that $L_{i,1:i-1}\boldsymbol{x}_{1:i-1}$ denotes the multiplication of a row with a column vector.

- Implement forward substitution `x = forward3(L,b)` by looping over the columns and using vector operations in each row.[3] Note that the following steps will overwrite $\boldsymbol{b}$ with the solution $\boldsymbol{x}$:

$$b_j = b_j/L_{jj};\ \boldsymbol{b}_{j+1:n} = \boldsymbol{b}_{j+1:n} - b_j L_{j+1:n,j} \text{ for } j = 1, 2, \ldots, n-1,$$
$$b_n = b_n/L_{nn}.$$

Choose a lower triangular matrix with random entries[4] to verify that your implementations are correct by comparing solutions with the build-in solver. Report timings for large systems ($n = 5,000$ and larger) for each of your implementation as well as the build-in implementation for solving linear systems (\ in MATLAB).[5]

6. **[Implementing Choleski, 3+2pt]** Implement the Choleski decomposition for a symmetric and positive definite (spd) matrix A. The function `C = mychol(A)` should terminate if it encounters a zero or negative diagonal element, and otherwise return the upper triangular Choleski factor C such that $A = C'C$.

   (a) Verify your implementation by comparing with the build-in Choleski factorization. Document that comparison and hand in a code listing of `mychol()`.

   (b) Report timings for large (e.g., $n = 500, 1000, 5000$) spd matrices for your own as well as the build-in Choleski factorization.[6] Do the timings grow as expected from floating point operation counts (i.e., with a factor of n^3)?

7. **[Sherman-Morrison formula, 6pt]** Sometimes, one has to solve linear systems that are rank-1 modifications of other linear systems, for which a factorization is already available. Let us derive a solution algorithm for the modified system. Let $A \in \mathbb{R}^{n\times n}$ be invertible and $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$ be column vectors.

   (a) Show that the matrix $A + \boldsymbol{u}\boldsymbol{v}^T$ is invertible if and only if $\boldsymbol{v}^T A^{-1}\boldsymbol{u} \neq -1$, and that in this case the inverse is

$$(A + \boldsymbol{u}\boldsymbol{v}^T)^{-1} = A^{-1} - \frac{1}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}} A^{-1}\boldsymbol{u}\boldsymbol{v}^T A^{-1}.$$

[2] Compare with Program 1 in Section 3.2.1 in Quarteroni/Sacco/Saleri.

[3] Compare with Program 2 in Section 3.2.1 in Quarteroni/Sacco/Saleri.

[4] To avoid roundoff problems resulting in NaN's due to poorly conditioned matrices, you can choose a triangular matrix with all 1's in the diagonal in this tests.

[5] Despite the fact that each method requires roughly the same number of floating point operations, timings should differ significantly. This is mainly due to how matrices are stored in memory and how this memory is accessed. MATLAB stores matrices as one-dimensional arrays, column by column. You can verify that by accessing matrices with only one index, i.e., using $A(k)$ for an $n \times n$ matrix returns the entry $A(m, l)$, where $k = m + (l-1)n$, $1 \leq l, 1 \leq m \leq n$. Numbers that are next to each other in memory can be read from memory much faster than numbers that are stored further away from each other.

[6] You can generate a random spd matrix by multiplying a matrix with random entries with its transpose. The resulting matrix is symmetric and positive semidefinite, and almost always positive definite.

(b) Let $\boldsymbol{v}^T A^{-1}\boldsymbol{u} \neq -1$ and assume given the LU decomposition of A. Specify an efficient algorithms based on the Sherman-Morrison formula to solve the rank-1 modified system $(A + \boldsymbol{u}\boldsymbol{v}^T)\boldsymbol{x} = \boldsymbol{b}$ for a given right hand side $\boldsymbol{b} \in \mathbb{R}^n$.

(c) Use the above algorithm to solve the system

$$\begin{bmatrix} 2 & 1 & 2 & 2 & 3 \\ 2 & 3 & 4 & 4 & 6 \\ 3 & 3 & 7 & 6 & 9 \\ 4 & 4 & 8 & 9 & 12 \\ 5 & 5 & 10 & 10 & 16 \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} 8 \\ 15 \\ 21 \\ 29 \\ 36 \end{bmatrix}$$

by choosing a suitable diagonal matrix for A and vectors $\boldsymbol{v}, \boldsymbol{u}$.

8. **[Spectral radius, diagonally dominant matrices, 1+2+1pt]** Let $A \in \mathbb{R}^{n\times n}$, $n \geq 1$.

(a) Show that the spectral radius $\rho(A)$ is a lower bound to all induced matrix norms, i.e.,

$$\rho(A) := \max_{1\leq i\leq n} |\lambda_i(A)| \leq \|A\|,$$

where $\|A\|$ is the matrix norm induced by an arbitrary vector norm $\|\cdot\|$.

(b) Recall that a matrix is called strictly (row) diagonally dominant if

$$|a_{ii}| > \sum_{j\neq i} |a_{ij}|, \quad \text{for } i = 1, \ldots, n.$$

Show that the LU-factorization without pivoting of an invertible matrix A can be performed if A^T is strictly diagonally dominant.

(c) Show that strictly diagonal dominant matrices are invertible.

9. **[Numerical study of random matrices, Extra credit, 2pt]** The goal of this problem is to numerically explore some properties of random matrices. We will try to come up with conjectures based on experiments and plots of our results. A random matrix is an $n \times n$ matrix where each entry is a draw from a real normal distribution with zero mean and standard deviation $1/\sqrt{n}$.[7] The factor $\sqrt{n}$ is introduced to obtain a well-behaved limit as $n \to \infty$.

(a) What do the eigenvalues of a random matrix look like? Take 100 random matrices and superimpose all their eigenvalues in a single plot.[8] If you do this for $n = 8, 16, 32, 64, 128 \ldots$, what do you observe? How does the spectral radius $\rho(A)$ of a random matrix behave as $n \to \infty$?

(b) Let us look at the matrix norms $\|A\|_2$ of these random matrices. As we showed above, $\rho(A) \leq \|A\|_2$. Does the inequality appear to approach an equality as $n \to \infty$?

(c) Repeat with random triangular matrices, i.e., upper triangular matrices with entries drawn from the same distribution as above. How do your answers change?

[7] In MATLAB, `A=randn(n,n)/sqrt(n);`

[8] The eigenvalues will be complex and can be plotted in the complex plain. Turn off the connecting lines between them to obtain a clean plot—in MATLAB you can use something like `plot(eig(A),'o');`