Spring 2019: Numerical Analysis
Assignment 2 (due March 7, 2019)

1. **[1+1+2+2pt]** Let $f : \mathbb{R}^2 \mapsto \mathbb{R}^2$ defined by $f(x, y) = (f_1(x, y), f_2(x, y))^T$, where

$$f_1(x, y) = x^2 + 4y^2 - 4, \quad f_2(x, y) = 2y - \sqrt{3}x^2.$$

We want to find the roots of f, i.e., all pairs $(x, y) \in \mathbb{R}^2$ such that $f(x, y) = (0, 0)^T$.

(a) Sketch or plot the sets $\mathcal{S}_i = \{(x, y) \in \mathbb{R}^2 : f_i(x, y) = 0\}$, $i = 1, 2$, i.e., the set of all zeros of f_1 and f_2. What geometrical shapes do these sets have?

(b) Calculate analytically the roots of f, i.e., the intersection of the sets $\mathcal{S}_1$ and $\mathcal{S}_2$.

(c) Calculate the Jacobian of f, defined by

$$J_f(x, y) = \begin{pmatrix} \partial_x f_1(x, y) & \partial_y f_1(x, y) \\ \partial_x f_2(x, y) & \partial_y f_2(x, y) \end{pmatrix} \in \mathbb{R}^{2\times 2}.$$

Here, $\partial_x f_i(x, y)$ and $\partial_y f_i(x, y)$, $i = 1, 2$ denote the partial derivatives of f_i with respect to x and y, respectively.

(d) The Newton method in 2D is as follows: Starting from an initial value $(x_0, y_0)^T \in \mathbb{R}^2$, compute the iterates

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} x_k \\ y_k \end{pmatrix} - [J_f(x_k, y_k)]^{-1} f(x_k, y_k), \text{ for } k = 0, 1, \dots,$$

where $[J_f(x_k, y_k)]^{-1}$ is the inverse of the Jacobi matrix of f evaluated at (x_k, y_k). Implement the Newton method in 2D and use it to calculate the first 5 iterates for the starting values $(x_0, y_0) = (2, 3)$ and $(x_0, y_0) = (-1.5, 2)$. Plot these iterates in the xy-plane together with the curves $\mathcal{S}_1$ and $\mathcal{S}_2$. Please also hand in your code.[1]

2. **[1+2pt]** We study basic properties of the LU-factorization.

(a) Give an example of an invertible 3×3 matrix that does not have any 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, if

$$A = LU = L_1 U_1$$

with upper triangular matrices U, U_1 and unit lower triangular matrices L, L_1, then necessarily $L = L_1$ and $U = U_1$. You can use the results we discussed in class about products of lower/upper triangular matrices, and their inverses.

[1] Some useful syntax: The MATLAB commands `b=[1;2]` and `A=[1, 2; 3, 4]` create the column vector $\boldsymbol{b} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$ and the 2-by-2 matrix $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$. Moreover, `A*b` is a simple matrix multiplication and to obtain $A^{-1}\boldsymbol{b}$, you can use either `inv(A)*b`, which inverts the matrix A, or (much better!) the command `A\b`, which solves the linear system $A\boldsymbol{x} = \boldsymbol{b}$. You can use the command `surf` to make surface plots.

3. **[4pt]** Let $n \geq 2$. Consider a matrix $A \in \mathbb{R}^{n\times n}$ for which every leading principal submatrix of order less than n is non-singular.

   (a) Show that A can be factored in the form $A = LDU$, where $L \in \mathbb{R}^{n\times n}$ is unit lower triangular, $D \in \mathbb{R}^{n\times n}$ is diagonal and $U \in \mathbb{R}^{n\times n}$ is unit upper triangular.

   (b) If the factorization $A = LU$ is known, where L is unit lower triangular and U is upper triangular, show how to find the LU-factors of the transpose A^T. Note that our requirement for an LU-factorization is that L is *unit* lower triangular, and U is upper triangular.

4. **[3+2pt]** LU factorization without pivoting.

   (a) Implement the LU factorization using $(2.18), (2.19)$ from the textbook (hence assuming no permutations are required), and apply it to the matrix

$$A = \begin{bmatrix} 6 & 2 & 1 & -1 \\ 2 & 4 & 1 & 0 \\ 1 & 1 & 4 & -1 \\ -1 & 0 & -1 & 3 \end{bmatrix}.$$

   (b) Generalize your code to handle input matrices A of any size $n \geq 2$. To avoid division by very small numbers or zero, check at each step that the absolute value of u_{jj} in (2.18) is not smaller than 10^{-8}. If it is, display an error message[2] and stop the code. Please also hand in your code.

5. **[2+2+2pt]** Let us use the LU-decomposition to compute the inverse of a matrix[3].

   (a) Describe an algorithm that uses the LU-decomposition of an $n \times n$ matrix A for computing A^{-1} by solving n systems of equations (one for each unit vector).

   (b) Calculate the floating point operation count of this algorithm.

   (c) Improve the algorithm by taking advantage of the structure (i.e., the zero entries—see question 5a) of the right-hand side. What is the new algorithm's floating point operation count?

6. **[4pt]** Bisection and Newton's method.

   In this problem we consider the function $f : [0, 10] \to \mathbb{R}$ defined by $f(x) = x^2 - 1$.

   (a) Verify that f has a root ξ in the interval $[0, 10]$. Although this may be obvious by inspection, justify this by citing a theorem, and identify the root ξ.

   (b) Show that in the interval $[\xi - \frac{1}{2}, \xi + \frac{1}{2}]$, the function f satisfies:

$$\frac{|f''(x)|}{|f'(y)|} \leq A \quad \forall x, y \in [\xi - \frac{1}{2}, \xi + \frac{1}{2}]$$

   Determine the smallest such value of A for the above statement to hold.

[2]MATLAB has the command `error('message')` for doing that.

[3]This also illustrates that computing a matrix inverse is significantly more expensive than solving a linear system.

(c) Write down a sufficient condition on x_0, the initial guess for Newton's method, for quadratic convergence. Justify your answer by considering Theorem 1.8 in Sulï and Mayers and using part (b).

(d) Explain why you can apply the bisection iteration, using the initial interval $[a_0, b_0] = [0, 10]$, to compute the root ξ. Write down the first four iterations (i.e. intervals) of the bisection iteration.

(e) If the midpoints of the intervals from the bisection iteration with initial interval $[a_0, b_0] = [0, 10]$ are used as initial guesses for Newton's method, how many bisection iterations must you do to guarantee quadratic convergence of Newton's method?

7. **[3pt]** Cholesky factorization of a 2×2 matrix.

This problem is concerned with the factorization of a symmetric positive definite matrix. For some with $b \neq 0$, we consider a symmetric 2×2 matrix of the form

$$\boldsymbol{A} = \begin{bmatrix} a & b \\ b & a \end{bmatrix}.$$

(a) Derive a condition under which $\boldsymbol{A}$ is positive definite.

(b) Assuming the condition derived in part (a), perform a step of Gaussian elimination to reduce $\boldsymbol{A}$ to upper triangular form.

(c) Working from the result in part (b), express $\boldsymbol{A}$ in the form $\boldsymbol{L}\boldsymbol{L}^T$, for some lower triangular matrix $\boldsymbol{L}$. Note that $\boldsymbol{L}$ is not necessarily unit lower triangular. This factorization is called the *Cholesky factorization*, and $\boldsymbol{L}$ is sometimes called the "square root" of $\boldsymbol{A}$. This type of factorization exists in general for $n \times n$ symmetric positive definite matrices!

8. **[4pt]** Deflation

We will now consider how to find all the roots of a polynomial on an interval. Let $f_5(x) = 2x^5 + 2x^4 - 16x^3 - 16x^2 + 32x + 32$; we will search for roots of f_5 on $I = [-4, 4]$.

(a) Prove that there exists a root on I. Use whichever iterative method you prefer to obtain the first root of f_5, called ξ_1.

(b) Define a deflated function $f_4(x) = \frac{f_5(x)}{x-\xi_1}$, note that $f_4(x)$ as written is undefined at ξ_1. Provided that the exact value of ξ_1 has been determined, how are the roots of $f_4(x)$ related to those of f? Using an iterative method, find a second root, ξ_2, of $f(x)$.

(c) Find the remaining roots of $f(x)$, ξ_3, ξ_4, ξ_5, with the deflated polynomials $f_i(x) = \frac{f_{i+1}(x)}{x-\xi_{5-i}}$, for $i = 3, 2, 1$.

(d) If f is a high degree polynomial, what problems can you envision when this algorithm is implemented on a computer? Explain.

9. **[4pt]** Stability of the Gaussian elimination algorithm.

Consider the system

$$Ax = b, \tag{1}$$

where $A, L, E \in \mathbb{R}^{n\times n}$, $x, b \in \mathbb{R}^n$, with $A = L + E$. L is unit lower triangular, where all the subdiagonal elements are -1, i.e., $l_{i,j} = -1$ for $i < j$, $l_{i,i} = 1$ for $i = 1 \dots n$, and $l_{i,j} = 0$ otherwise. Additionally, E is a matrix of such that $e_{i,n} = 1$ for $i = 1 \dots n-1$ and $e_{i,j} = 0$ otherwise. For example, when $n = 5$, we have

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & 1 \\ -1 & -1 & 1 & 0 & 1 \\ -1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 \end{pmatrix}.$$

(a) Prove that A is invertible.

(b) When $n = 5$ and $b = (1, \frac{1}{4}, \frac{1}{9}, \frac{1}{16}, \frac{1}{25})^T$, solve for x in (1) using Gaussian elimination, then backward substitution.

(c) Now consider A, x, b, where n is not specified. Put (1) into the form $Ux = c$, where U is upper triangular using Gaussian elimination (you do not have write what c is since b is not given). What is $\max_{i,j} |u_{i,j}|$?

(d) For large n, e.g. $n = 2000$, what problems can you envision if you try to solve (1) using Gaussian elimination on a computer? Explain.