Homework No.1, 553.480/680, Due September 8, 2023.

1. Construct a Taylor series around \( x = 0 \) for the function \( f(x) = (1 - x)^{-1} \) and establish an exact formula for the remainder when truncating after the first \( n \) terms.

2. (a) Consider the matrix \( A = \begin{pmatrix} \alpha & \beta \\ 0 & \beta \end{pmatrix} \) for real numbers \( \alpha, \beta \). Show by induction that \( A^n = \begin{pmatrix} \alpha^n & \beta^n \\ 0 & \beta^n (\alpha/\beta - 1) \end{pmatrix} \) for integers \( n \geq 1 \).

   (b) Using your result in part (a) show that \( B = \sum_{n=0}^{\infty} A^n \) exists if \( |\alpha|, |\beta| < 1 \) and show further that \( B = (I - A)^{-1} \).

   We shall discuss later in the course a generalization valid for any square matrix \( A \).

3. Convert the following numbers to their decimal equivalent:

   (a) \((f1e2d3)_{16}\)
   (b) \((8a9b.4c5)_{16}\)
   (c) \((3210.11)_{4}\)
   (d) \((0.1\overline{3})_{4}\)

4. Show that to convert from hexadecimal representation to binary, each hecadecimal digit \( b \) is replaced, in order, by the four binary digits \( a, a', a'', a''' \) that satisfy

\[
b = 2^3a + 2^2a' + 2a'' + a'''.
\]

Use this fact to construct a simple algorithm to convert a hexadecimal representation to a binary representation for both integers and fractions. Write a pseudocode to make the conversion \( b \rightarrow (a, a', a'', a''') \), using the function \( \text{mod}(x, 2) \), which gives the remainder of \( x \) when divided by 2.

*Hint: \( a''' = \text{mod}(b, 2) \) and \( b' \equiv (b - a''')/2 = 2^2a + 2a' + a'' \).*

5. In IEEE Standard Floating-Point representation for numbers, a 12-bit binary number is used first to encode the sign \( \sigma \) and exponent \( E \), followed by a 52-bit binary to encode the fraction \( F \). This can also be represented by a 3-digit hexadecimal number, followed by a 13-digit hexadecimal number, as in MATLAB. Consider the following IEEE Standard Floating-Point number in hexadecimal form:

\[x=c023bd3cc9be45de\]

(a) Use the first 3 hexadecimal digits to determine the sign \( \sigma \) and exponent \( E \) of \( x \) in decimal format.
(b) Convert the remaining 13 hexadecimal digits to a decimal representation of the fraction $F$ for $x$.
(c) Calculate the decimal representation of the double-precision number $x$.

6. Suppose that $f_l(x)$ is a $k$-digit rounding approximation to $x$ in base $\beta$. Show that
\[
\left| \frac{x - f_l(x)}{x} \right| \leq \frac{1}{2} \beta^{-k+1}
\]
Assume that the base $\beta$ is even, as it is for the most familiar cases ($\beta = 2, 10, 16$).

**Hint:** See BF Exercise 1.2.24.

7. For the following numbers $x_T$, the true value, and $x_A$, the approximation, how many significant digits are there in $x_A$ with respect to $x_T$?
   (a) $x_A = -1$, $x_T = \sec(22)$
   (b) $x_A = -1.0457429987$, $x_T = \sec(23)$
   (c) $x_A = 2.357$, $x_T = \sec(24)$

8. Show that
\[
\text{Rel}(\hat{x}/\hat{y}) = \frac{\text{Rel}(\hat{x}) - \text{Rel}(\hat{y})}{1 + \text{Rel}(\hat{y})} \approx \text{Rel}(\hat{x}) - \text{Rel}(\hat{y})
\]
for $|\text{Rel}(\hat{x})|, |\text{Rel}(\hat{y})| \ll 1$.

9. (a) For each of the following expressions
   (i) $y^3 - x^3$
   (ii) $\ln(y) - \ln(x)$
   (iii) $\tan(y) - \tan(x)$
find a method to avoid loss of precision when $y \approx x$.
(b) Consider in MATLAB single precision numbers $x = \text{single}(2)$, $y = \text{single}(2.00001)$. Evaluate the expressions in (a) using both the original definition and also your improved formulation. How many significant figures do you obtain using the original definition and the improved formulation in single precision arithmetic?

10*. (a) What is the condition number for evaluating the function $f_n(x) = x^n$?
(b) Show that the following algorithm can be used to calculate $f_n(x)$: generating the sequence $y_n$ via $y_n = (x - \frac{1}{3}) y_{n-1} + y_{n-2}$, $n \geq 2$; $y_1 = x$, $y_0 = 1$, then $f_n(x) = y_n$.
(c) Implement the algorithm of part (b) numerically in Matlab for $x = \frac{1}{3}$ and for $n = 2, \ldots, 20$ and calculate the relative error in $y_n$ with respect to the true value $f_n(\frac{1}{3}) = (\frac{1}{3})^n$ and use these results to estimate the condition number of the algorithm. Plot the log of the condition number versus $n$. Is the condition number of the algorithm comparable to the true condition number?
(d) Explain the poor performance of the algorithm. *Hint:* Find the general solution of the iteration $y_n = (x - \frac{1}{x}) y_{n-1} + y_{n-2}$, $n \geq 2$ for arbitrary $y_0, y_1$.

(e) Repeat part (c) for $x = \frac{1}{2}$ and for $n = 2, .., 20$ and calculate the relative error in $y_n$ with respect to the true value $f_n(\frac{1}{2}) = (\frac{1}{2})^n$. Explain why the results differ so radically from those in part (c).