Homework 7

Ch En 263 – Numerical Tools

Due: 4 Mar. 2024

Instructions

- Complete the problems below and submit the following files to Learning Suite:
  - Handwritten portion: scan each page (or take a picture) and combine them into a single pdf named: `LastName_FirstName_HW7.pdf`
  - Excel portion: submit a workbook named `LastName_FirstName_HW7.xlsx` where each worksheet tab is named "Problem_1", "Problem_2", etc.
  - Python portion: submit a separate file for each problem named `LastName_FirstName_HW7_ProblemXX.py` where XX is the problem number.

Problems

1. In this problem you will write a Python program to do back substitution. Consider the upper triangular system of linear equations

$$
\begin{aligned}
x_0 + 2x_1 + 3x_2 &= 13 \\
x_1 - x_2 &= 2 \\
-2x_2 &= -4
\end{aligned}
$$

   (a) Write the matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$ for this system of equations and solve for $\boldsymbol{x}$ by hand.

   (b) Define numpy arrays for $\boldsymbol{A}$ and $\boldsymbol{b}$.

   (c) Write a loop which performs the sum $\sum_{j=i+1}^{n-1} a_{i,j}x_j$ for $i = 0$ assuming $x = [0, 1, -3]$. Print the sum to the console.

   (d) Write the full back substitution algorithm using a nested loop and print the solution, x, to the console. Use your hand-written solution to check your steps as necessary.

2. In this problem you will write a code to do a complete Gauss elimination algorithm in Python. *Hint: Re-use the forward elimination and backward substitution codes from above to help you do this.*

   (a) Write a function called `Gauss` that takes two arguments, a 2D Numpy array `A` and a 1D numpy array `b` and returns the solution `x` obtained via the Gauss elimination algorithm.

   (b) Import the data in `A.csv` into a 2D array `A` and the data in `b.csv` into a 1D array `b`. Use the function you defined in part (a) to find the solution `x` and print it to the console.

3. Answer the following questions using the function below which performs the back substitution algorithm.

```
1  def back_sub(A, b):
2      n = len(b)
3      x = np.zeros(n)
4      x[n-1] = b[n-1]/A[n-1,n-1]
5      for i in range(n-2, -1, -1):
6          xi_sum = 0
7          for j in range(i+1, n):
8              xi_sum += A[i,j]*x[j]
9          x[i] = (b[i] - xi_sum)/A[i,i]
10     return x
```

(a) If an integer is 4 bytes, a float is 8 bytes and $n = 100$, how many bytes of memory does the function need? *Hint: Include A and b in your calculation.*

(b) If c_1 is the time it takes to execute the first line of code, c_2 is the time it takes to execute the second line, etc., write an expression $T(n)$ that describes the time it takes to execute the entire function. *Hint: The inner-most loop runs* $(n-1)(n)/2$ *times.*

(c) What is the asymptotic behavior of $T(n)$ at large n? Using this asymptotic limit, estimate how long it will take for the function to execute when $n = 10^4$ if it takes 2 minutes using $n = 10^3$.

4. Use the tools in the Numpy linear algebra library to do the following.

(a) Find the norm of the matrix

$$\boldsymbol{A} = \begin{bmatrix} 0 & -2 & 1 & 0 & 0 \\ 2 & 0 & -2 & 1 & 0 \\ -1 & 2 & 0 & -2 & -1 \\ 0 & -1 & 2 & 0 & -2 \\ 0 & 0 & -1 & 2 & 0 \end{bmatrix}$$

and the vector

$$\boldsymbol{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 4 \\ 1 \end{bmatrix}.$$

Print the value of both to the console.

(b) Solve the equation $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ using $\boldsymbol{A}$ and $\boldsymbol{b}$ from Part (a) with Numpy's linear algebra solver. Print $\boldsymbol{x}$ to the console and verify your solution by evaluating the residual $|\boldsymbol{A} \cdot \boldsymbol{x} - \boldsymbol{b}|$.

(c) Solve the equation $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ using $\boldsymbol{A}$ and $\boldsymbol{b}$ from Part (a) by finding $\boldsymbol{A}^{-1}$ using Numpy. Print $\boldsymbol{x}$ to the console and verify your solution by evaluating the residual $|\boldsymbol{A} \cdot \boldsymbol{x} - \boldsymbol{b}|$.