

University of Toronto Mississauga  
Department of Mathematical and Computational Sciences  
**CSC 338 - Numerical Methods, Spring 2009**

**Assignment 3**

Due date: Tuesday March 24, 12:10pm, at the start of tutorial.  
No late assignments will be accepted.

Note: The material you hand in should be legible (either typed or *neatly* hand-written), well-organized and easy to mark, including the use of good English. All computer programs should be handed in and should be well commented. In general, short, simple answers are worth more than long, complicated ones. Unless stated otherwise, all answers should be justified.

**If you do not know the answer to a question, and you write “I don’t know”, you will receive 20% of the marks of that question. If you just leave a question blank with no such statement, you get 0 marks for that question.**

1. The following exercises are to be done by hand.
  - (a) Question **3.23** on page 147.
  - (b) Question **3.2** on page 149.
  - (c) Question **3.5** on page 149.
  - (d) Question **3.17** on page 150.
  - (e) Question **3.23** on page 151.
  - (f) Question **3.24** on page 151.

2. (37 points total) The following computer problems are to be done using Matlab or Octave. For each question, hand in your program code and a transcript of a terminal session demonstrating that your programs work correctly. Be sure to indicate clearly which questions the programs and the transcripts refer to. (Note: some paper-and-pencil exercises are included.)
- (a) Question **3.8** on page 154. You should use the Matlab functions `chol` and `qr` to compute the Cholesky factorizations and QR factorizations, respectively, and then use the Matlab backslash operator, `\`, to solve the resulting triangular systems. (For more details, see the Matlab help pages for these functions, and search for “Cholesky” using the Matlab help facility.)
- (b) Do the following:
- Write a Matlab function `measure(n)` that does the following: It first generates two random  $n \times n$  matrices,  $A$  and  $B$ , and a random column vector,  $x$ , of length  $n$ . It then uses `tic` and `toc` to measure how long it takes to compute  $A(Bx)$  and how long it takes to compute  $(AB)x$ . It then returns these two time measurements. Test the function with  $n = 3000$ .
  - Write a Matlab function `medianMeasure(n, M)` that executes `measure(n)`  $M$  times. You will get  $M$  different pairs of time measurements. Compute the median time to execute  $A(Bx)$ , and the median time to execute  $(AB)x$ . (Use the Matlab `median` function.) Return the two median time measurements. Test the function with  $n = 2000$  and  $M = 10$ .
  - Execute `medianMeasure(n, M)` with  $M = 50$  and  $n = 50, 100, 150, \dots, 500$ . Plot the median time to execute  $A(Bx)$  vs.  $n$ . On the same set of axes, plot the median time to execute  $(AB)x$  vs.  $n$ . (Use the Matlab `plot` function.) Explain the difference in the two plots.
  - In terms of  $n$ , how many multiplications are needed to compute  $A(Bx)$ . How many are needed to compute  $(AB)x$ ?
  - Suppose  $A_i$  is an  $n \times n$  matrix and  $x$  is a column vector of length  $n$ . How would you efficiently execute the expression  $A_1 A_2 \cdots A_n x$ ? How many multiplications does this require?

- (c) Do the following:
- i. Write a simple program that, given a rectangular matrix, computes the Householder matrices  $H_1$ ,  $H_2$  and  $H_3$ . You do not need to worry about the efficiency of this program. Just keep it simple. (Use the Matlab function `size` to get the dimensions of the given matrix.)
  - ii. Generate a random  $10 \times 6$  matrix,  $A$ , and use your program to generate  $H_1$ ,  $H_2$  and  $H_3$ . Verify that each  $H_i$  is orthogonal.
  - iii. Use  $H_1$ ,  $H_2$  and  $H_3$  to annihilate the first three subdiagonal columns of  $A$ . Use ordinary matrix multiplication.
  - iv. Write a faster program to annihilate the first three subdiagonal columns of a rectangular matrix, without generating  $H_1$ ,  $H_2$  and  $H_3$  and without using matrix multiplication. (Use the results on slide 32 of chapter 3.) Try to minimize the number of arithmetic operations performed and the amount of storage used, but do not overwrite the matrix  $A$ . The use of brackets in matrix expressions will be important. Also, use the Matlab `:` operator to minimize the number of loops in your program, especially nested loops (since loops in Matlab are slow).
  - v. Apply your faster program to matrix  $A$  and verify that it gives the same results as in part iii.
  - vi. Generate a random  $8 \times 3$  matrix,  $A$ , and a random column vector,  $b$ , of length 8. Use your program from part i to triangularize  $A$ , and then use backward substitution to find the vector  $x$  that minimizes  $\|Ax - b\|_2$ . (Use the Matlab backslash operator, `\`, to perform the backward substitution.) Compare your answer to the true answer by using the Matlab expression  $A \setminus b$ , which solves the same problem.
  - vii. Modify your program from part iv to efficiently find the vector  $x$  that minimizes  $\|Ax - b\|_2$ . Verify that it gives the same result as part vi.
  - viii. Generate random  $m \times n$  matrices for  $n = 30, 100, 300, 1000, 2000, 3000$  and  $m = 2n$ . Use the Matlab functions `tic` and `toc` to measure how much time your program in part iv takes to execute on each of these matrices. Also measure the amount of time used by your program in part i. (If your computer cannot handle the larger matrices, don't worry about it.)

**No more questions will be added**

University of Toronto Mississauga  
Department of Mathematical and Computational Sciences  
CSC 338 - Numerical Methods, Spring 2009

## Cover sheet for Assignment 3

---

Complete this page and hand it in with your assignment.

**Name:** \_\_\_\_\_  
(Underline your last name)

**Student number:** \_\_\_\_\_

I declare that the solutions to Assignment 3 that I have handed in are solely my own work, and they are in accordance with the University of Toronto Code of Behavior on Academic Matters.

**Signature:** \_\_\_\_\_