University of Toronto Mississauga

CSC 338 - Numerical Methods, Spring 2010

Assignment 2

Due date: Thursday Feb 25, 4:10pm, at the start of tutorial.
No late assignments with 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.

1. The following exercises from Chapter 2 in the text are to be done by hand.

   (a) (8 points total) Questions 2.61 (4 points) and 2.62 (4 points) on page 95.

   (b) (1 point) Question 2.66 on page 96.

   (c) (4 points total) Questions 2.75 (1 point) and 2.76 (3 points) on page 96.

   (d) Questions 2.16 and 2.17 on page 97.

   (e) Question 2.21 on page 98.

   (f) Question 2.22 on page 98. (Hint: $\sum_{i=1}^{n} i^{k-1} \approx n^k/k$)

   The following computer problems are to be done using Matlab or Octave. For each question, hand in your program code, any requested output, 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.

2. (45 points total) This question illustrates the effect of the condition number of a matrix on numerical error. The question first focuses on matrix inversion, and then looks at the difference (in terms of error) of using matrix inversion v.s. Cholesky factorization.

   (a) Download the data file `mnist_all.mat` from the course web page and load it into Matlab. This should create a number of variables in your Matlab workspace called test0, test1, ... test9 and train0, train1, ... train9. Each of these variables is a matrix having 784 columns. Each row of each matrix represents a hand-written digit. For example, each row of the matrix test7 represents the digit 7. Each of the train variables represents about 6,000 digits, and each of the test variables represents about 1,000 digits. You will be using these variables in future assignments to train and test a program that learns to do optical character

recognition. For now, we will be looking at a small number of these digits to familiarize you with the data and to visualize numerical error.

Although each digit is stored as a row vector with 784 components, it actually represents an array of pixels with 28 rows and 28 columns ($784 = 28 \times 28$). To view a digit, you must first convert it to a 28×28 array using the Matlab function `reshape`. (You can check the dimensions of any Matlab variable using the `size` function.) To display an array as a 2-dimensional image, use the Matlab function `imagesc`. If an image comes out sideways, try displaying its transpose. To convert an image to black-and-white, give the Matlab command `colormap(gray)`.

(b) (2 points) Display the second digit in `test5` and the 53^{rd} digit in `train3` as 2-dimensional black-and-white images.

(c) (2 points) The 784 components of each digit are not floating-point numbers. Instead, they are 8-bit unsigned integers (`uint8` in industry-standard terminology). Before we can perform matrix operations on them, we must convert them to double-precision floating-point numbers using the Matlab function `double`. It will also be convenient to represent a digit as a column vector instead of a row vector. Convert the first digit in `test7` to a floating-point column vector and store it in the variable x. Check the dimensions of x using the `size` function. Display x as a 2-dimensional black-and-white image. (It should look like a 7.) You will be using x throughout the rest of this question.

(d) (2 points) Download the data file `matrixA.mat` from the course web page and load it into Matlab. This should create a variable called A containing a 784×784 floating-point matrix in double precision. (A was randomly generated using the Matlab `rand` function.) Let B be the matrix $A \times A^T$. Use the Matlab function `cond` to compute the condition numbers of A and B.

(e) (4 points) Using the Matlab function `inv` to compute matrix inverses, let $y = A\,A^{-1}x$, and let $z = B\,B^{-1}x$. Display y and z as 2-dimensional black-and-white images. If you have done everything correctly, y should look exactly like x, and z should look like noise (2 points). Why do y and z look so different (2 points)?

(f) (3 points) This question illustrates the gradual increase in error as condition number increases. For various values of α, let $C = B + I/\alpha$, where I is the identity matrix. (Use the Matlab function `eye` to generate the identity matrix of a given dimension.) Try $\alpha = 10^6, 10^{6.5}, 10^7, 10^{7.5}$ and 10^8. For each value of α, let $y = C\,C^{-1}x$, and display y as a 2-dimensional black-and-white image (2 points). If you have done everything correctly, y should become noisier as α increases. What happens to the condition number of C as α increases (1 point)?

(g) (8 points) This question quantifies the observations made in part (f). Let $r = x - C\,C^{-1}x$ be the error (or residual). Produce a log-log plot of $\|r\|$ v.s. α (2 points). You should plot points at $\alpha = 10^n$ for 40 equally spaced values of n between 4 and 8. Also produce a log-log plot of the condition number of C v.s. α for the same values of α (2 points). You can use the Matlab function `norm` to compute $\|r\|$, and the Matlab function `loglog` to generate the log-log plots. Use

the Matlab functions `xlabel` and `ylabel` to label the axes of the plots. Explain why each plot tends to increase from left to right (4 points).

(h) (9 points) If you have done everything correctly, the log-log plot of condition number in part (g) should have a linear section. Here we explore why this is. In terms of B and α, the condition number of matrix C is given by the following formula:

$$\frac{s_1 + 1/\alpha}{s_2 + 1/\alpha}$$

where $s_1 = \|B\|$ and $s_2 = 1/\|B^{-1}\|$. Give a simple, intuitive interpretation of this formula (3 points). Show that the formula is approximately linear for the particular values of B and α used in part (g) (2 points). To verify that the formula is correct, plot it's values on top of the log-log plot of condition number in part (g) (2 points). (You may find the Matlab command `hold` useful for plotting one curve on top of another.) If you have done everything correctly, the two plots should coincide in the linear section. Why does this section of the plots appear to have a slope of 1 (1 point)? Explain any differences in the two curves (1 point). Hint: Reread Section 2.3.3 of Heath.

(i) (5 points) Prove that matrix C in part (f) is positive definite if $\alpha > 0$.

(j) (6 points) In this question, you will use backward error analysis to compare the accuracy of matrix inversion and Cholesky factorization. In particular, you will solve the equation $Cy = x$ in two different ways: (*i*) letting $y_1 = C^{-1}x$, and (*ii*) letting y_2 be the solution derived through Cholesky factorization followed by forward and back substitution. (Since matrix C is positive definite, we can apply Cholesky factorization to it.) Here, x is the column vector generated in part (c). To compute the backward errors (or residuals), let $r_1 = x - Cy_1$ and $r_2 = x - Cy_2$. On a single set of axes, generate log-log plots of $\|r_1\|$ v.s. α and of $\|r_2\|$ v.s. α (4 points). You should plot points at $\alpha = 10^n$ for 40 equally spaced values of n between 4 and 7 (not 8). If you have done everything correctly, the plots should show that Cholesky factorization is almost always more accurate than matrix inversion. Based on your plots, how much more accurate is Cholesky factorization in this case (2 points)? You can use the Matlab function `chol` to do Cholesky factorization, and the backslash operator, \, to do substitution. (See the Matlab function `mldivide` for more information on the backslash operator).

(k) (4 points) Using $\alpha = 10^7$, generate 2-dimensional black-and-white images of $|Cy_1|$ and $|Cy_2|$ (2 points). (You can use the Matlab function `abs` to compute absolute values.) If you have done everything correctly, you should find that Cholesky factorization gives an image with much less background noise than does matrix inversion. What is the reason for using absolute value when generating these images? (2 points) (Hint: read the Matlab help page for `imagesc`.)

3. (10 points total) Question **2.2** on page 100 (3 points for part a, 2 for b, 5 for c). You should use the Matlab function `lu` to compute the LU factorization of the matrix A, and then use the Matlab backslash operator, \, to solve the resulting triangular systems. (See the Matlab function `mldivide` for more information on the backslash operator).

4. (5 points) Question **2.3** on page 100.

No more questions will be added

University of Toronto Mississauga
CSC 338 - Numerical Methods, Spring 2010

Cover sheet for Assignment 2

Complete this page and hand it in with your assignment.

Name: _____
(Underline your last name)

Student number: _____

I declare that the solutions to Assignment 1 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: _____