University of Toronto Mississauga

CSC 338 - Numerical Methods, Spring 2010

Assignment 3

Due date: Thursday March 18, 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 3 in the text are to be done by hand.

   (a) Question 3.28 on page 148.

   (b) Question 3.29 on page 148.

   (c) Question 3.1 on page 149.

   (d) Question 3.6 on page 149.

   (e) Question 3.17 on page 150.

   (f) Question 3.18(b), (b) and (c) on page 150.

   (g) Question 3.19(a), (b) and (d) on page 150.

   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. Question 3.5(a) on page 153.

3. (Optical Character Recognition: Part I) In this problem, you will write and test some Matlab functions for use in optical character recognition in the next assignment. The strategy is to model each character as a multivariate normal distribution. Recall that in such a distribution, the probability density of an m-dimensional vector, x, is given by

$$\frac{exp[-(x-\mu)^T\Sigma^{-1}(x-\mu)/2]}{(\sqrt{2\pi}\,|\Sigma|)^m} \tag{1}$$

   Here, μ is the mean of the distribution, Σ is it's covariance matrix, and $|\Sigma|$ is the determinant of Σ. In optical character recognition, each character has a different μ

and a different Σ, and one problem is to determine what these are. We shall do this using techniques from Artificial Intelligence, and specifically from the subfield of Machine Learning. The idea is to write and test a Matlab program that learns μ and Σ from a sample of training data. That is one of the problems below.

Unless otherwise specified, you may not use any of Matlab's statistical functions, such as `mean`, `var`, `cov` or `mvnpdf`. Also, for full marks, *you should not use any loops.* Instead, you should use Matlab's vector and matrix operations, which are much faster and can be executed in parallel. To this end, you may find the following functions useful: `sum`, `diag`, `dot` and `repmat`.

(a) Suppose $x_1, x_2, ...x_n$ are vectors. Their mean is the vector μ, where $\mu_j = \sum_{i=1}^{n} x_{ij}/n$ and x_{ij} is the j^{th} component of vector x_i. The covariance matrix of the vectors is the matrix Σ, where

$$\Sigma_{kj} = \sum_{i=1}^{n} (x_{ik} - \mu_k)(x_{ij} - \mu_j)/n$$

We shall store the vectors in a matrix, X, where the i^{th} row of the matrix is the vector x_i. Prove that $\Sigma = X_c^T X_c/n$, where X_c is a matrix whose i^{th} row is the vector $x_i - \mu$. Also prove that Σ is positive semi-definite (This is the same as positive definite with $>$ replaced by $\geq$). (In standard parlance, X is called the data matrix, and X_c is called the centered data matrix, since the row vectors of X_c have a mean of 0.)

(b) Write a Matlab function `fitNormal(data)` that fits a multivariate normal distribution to a data set. Here, `data` is a data matrix as described above. The function should return the mean and covariance matrix of the data.

(c) Write a Matlab function `MVNinv(X,mu,sigma)` that computes probabilities for a multivariate normal distribution with mean `mu` and covariance matrix `sigma`. Specifically, the function should evaluate expression (1) for each row vector in matrix `X` and should return the probabilities as a column vector. For this question, your function may use loops and matrix inversion.

(d) Write a Matlab function `MVNchol(X,mu,sigma)` that does the same thing as `MVNinv` but without using any loops and by using Cholesky factorization and substitution instead of matrix inversion. For full marks, you should do substitution only once (i.e., you should use the Matlab backslash operator only once). Explain your solution.

(e) In the next several questions, you will test your functions on a sample of 2-dimensional data. The first step is to download the datafile `data2D.mat` from the course website and load it into Matlab. This should create a variable in your Matlab workspace called `Data` whose value is a 100×2 matrix of double-precision floating-point numbers. Each row of the matrix represents a 2-dimensional point. Plot all the points, using a dot for each point.

(f) Use `fitNormal` to fit a multivariate normal distribution to `Data`. Print out the mean and covariance matrix. If you have done everything correctly, σ should be symmetric and its determinant should be about 4.

(g) If you have written the functions `MVNinv` and `MVNchol` correctly, then they should produce the same answers (or almost exactly the same answers). Test this by running them on `Data` and on the mean and covariance matrix fitted to `Data` in part (f).

(h) Using the Matlab function `contour`, plot a 2-dimensional contour map of the multivariate density function fitted to `Data` in part (f). You should plot the function on a rectangular window that is just big enough to include all the points in `Data`. To generate a contour map, you will need to divide the window into a rectangular grid, and evaluate the density function at each of the grid points. For this question, you should use a 100×100 grid of equally spaced points, for a total of 10,000 grid points.

As you can see in the Matlab help page, there are a variety of ways to call the `contour` function. Perhaps the most convenient for this question is the call `contour(X1,X2,Z)`, where `X1` and `X2` are vectors that specify the grid points, and `Z` is a matrix of function values at the grid points. In our case, `X1` and `X2` each have length 100, and `Z` is a 100×100 matrix. Here, `X1` and `X2` specify the division of each of the two axes into 100 equal-sized intervals. For full marks, you should use a single call to the function `MVNinv` to evaluate the density function at the grid points. You may find the Matlab functions `linspec`, `meshgrid` and `reshape` useful. If you have done everything correctly, the contour plot should be a set of concentric ellipses. The points on an ellipse all have the same probability density.

(i) Add the points in `Data` to your contour plot in part (h). If you have done everything correctly, the ellipses in the contour plot should be centred on the middle of the data set and should be aligned with the data points. Most of the data should lie inside the largest ellipse.

(j) Repeat parts (h) and (i) using `MVNchol` instead of `MVNinv` to evaluate the density function at the grid points. If you have done everything correctly, you should get exactly the same plot.

(k) Repeat parts (h) and (i) using the function `pcolor` instead of `contour` to produce a pseudo-color plot of the probability density function. Use the Matlab command `shading interp` to specify the color shading of the plot. Regions of similar color in the plot have similar probability density.

(l) Repeat part (h) using the function `surf` instead of `contour` to plot a 3-dimensional surface of the probability density function. Use faceted shading.

(m) Repeat parts (i), (k) and (l) using a 10×10 grid, instead of a 100×100 grid. Describe and explain any differences in the plots.

No more questions will be added

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

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 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: _____