Quiz 2 Self-Study Guide 2.086/2.088/2.087 Spring 2013

Quiz 2 will focus on the material of Unit III and Unit IV: the material addressed in the lectures or the lecture notes; and the material addressed in Assignments 3 and 4 (and associated Matlab tutorials). Quiz 2 will not include material from Unit V or Assignment 5[1] but will include the linear algebra and related Matlab concepts developed in Unit III.

In this guide we propose a few questions which might help you review your understanding of the material and focus your study effort accordingly. Note in most cases we do not state the questions overly precisely, nor do we provide any specific answers; we ask you to decide, as part of the study process, how best to pose the question, and how best to interpret and challenge your answers. We will spend some time in lecture (on Tuesday, 30 April) working through the questions and answers together.

1. This question relates to vector and matrix operations. Note in all cases we refer to linear algebra operations.

   Define two 4×1 vectors v and w. Define two 4×4 matrices A and B. Also define the 4×1 vector $z = (0 \;\; 0 \;\; -1 \;\; 0)^{\mathrm{T}}$ (in Matlab, `z = [0;0;-1;0]`).

   Calculate the inner product $v^{\mathrm{T}} w$ "by hand"; confirm your result with Matlab.

   Define $p = Aw$. Calculate p_2 (the second element of the vector p) "by hand"; confirm your result with Matlab. This is a good opportunity for two-handed multiplication: recall the i^{th} element of p is the inner product of the i^{th} row of A with the vector w.

   Define $q = Az$. Calculate all entries of q "by hand"; confirm your result with Matlab. This is a good opportunity for one-handed multiplication: recall that q is the sum over j of the j^{th} column of A weighted by z_j.

   Define $C = AB$. Calculate $C_{1\,2}$ (the element of C in the first row and second column) "by hand"; confirm your result with Matlab. Recall that the entry $C_{i\,j}$ is given by the inner product of the i^{th} row of A and the j^{th} column of B.

   Define two matrices R and S for which RS is defined but SR is not defined. Confirm your result with Matlab — Matlab should give an error message if the matrices do not have the correct "size" for matrix-matrix multiplication.

2. This question relates to least squares, regression analysis, and joint confidence intervals. (Note that, on the quiz, we will not ask you to derive the formulas for joint confidence intervals, though we may well ask you to apply the formulas provided in the lecture notes.)

   We consider here measurements which take the form

   $$Y_i = Y_{\text{model}}(x_i; \beta^{\text{true}}) + \epsilon_i \;, \;\; 1 \le i \le m \;, \tag{1}$$

   for

   $$Y_{\text{model}}(x; \beta) = \beta_0 + \beta_1 x \;; \tag{2}$$

[1] Since you are not responsible for the material of Assignment 5 on Quiz 2 you can also safely assume that there will be no questions related to power iteration or inverse power iteration even though this material is nominally part of Unit IV.

here $\beta^{\text{true}} = (\beta_0^{\text{true}}\ \ \beta_1^{\text{true}})^{\text{T}}$ and $\beta = (\beta_0\ \ \beta_1)^{\text{T}}$. Note that x is our independent variable: the x_i, $1 \leq i \leq m$, are the values of x at which we take the respective measurements Y_i, $1 \leq i \leq m$; there are m measurements in total. We assume that the noise satisfies our assumptions N1, N2, and N3 with standard deviation σ. In short, we are fitting a straight line to noisy data.

You can create "synthetic" data in order to exercise your least squares and regression skills. In particular, the MATLAB code

```
clear
m = ?; % for you to specify, and ultimately vary
xpts = linspace(0,1,m)'; % for simplicity let x vary from 0 to 1
beta_0_true = ? ; % set to any value you choose
beta_1_true = ?; % set to any value you choose
sigma_true = ?; % set to any value you choose, but presumably not too large
Y = beta_0_true + xpts*beta_1_true + sigma_true*randn(m,1);
```

creates for you a dataset and in particular a dataset for which you know the truth. Here m corresponds to m; xpts(i) corresponds to $x_{\texttt{i}}$, $1 \leq \texttt{i} \leq m$; beta_0_true and beta_1_true correspond to β_0^{true} and β_1^{true}, respectively; sigma_true corresponds to σ; and Y(i) $= Y_{\texttt{i}}$, $1 \leq \texttt{i} \leq m$, is our (synthetic) data. Note each time you run the script you generate a new dataset since randn will generate new random variates (with zero mean and unit standard deviation).

Now add to the script above code which computes the least squares approximation to β^{true}, which we typically denote $\hat{\beta}$; our estimate for σ, which we typically denote $\hat{\sigma}_m$; and joint confidence intervals (at the 95% confidence level) for β_0^{true} and β_1^{true}. Is your estimate for β^{true}, $\hat{\beta}$, close to the anticipated value? Is your estimate for σ, $\hat{\sigma}_m$, close to the anticipated value? Do your confidence intervals contain β^{true} — *should* they (always) contain β^{true}?

What happens as you increase m, the number of (synthetic) measurements? Do your estimates $\hat{\beta}$ and $\hat{\sigma}_m$ improve? Do your confidence intervals become tighter? What quantity in the confidence interval expressions changes most significantly as you increase m?

3. This question relates to ordinary differential equations and in particular initial value problems.

   Consider the equation

$$\frac{du}{dt} = -a\,t\,u \quad , \quad 0 \leq t \leq t_f \ , \tag{3}$$

   with initial condition

$$u(t = 0) = u_0 \ , \tag{4}$$

   for the particular case of $a = 1$, $t_f = 10$, and $u_0 = 1$. Note this is similar to the model problem studied in class except that here λ is replaced by $-a\,t$ and hence is time-dependent; however, our schemes are defined generally for any right-hand-side "dynamics" $g(t, u)$, and hence we can readily treat this new equation.

   Can you find the exact solution to this equation? (This is not so crucial for 2.086. As a hint: in a first step, you can derive the relationship $\ln(u/u_0) = -at^2/2$.)

   Consider the Euler Forward scheme applied to Equations (3), (4): denote the approximation $\tilde{u}^j_{\text{EF}}$, $0 \leq j \leq J$, as in the lecture notes and textbook (but here we add the subscript EF). Recall that $\tilde{u}^0_{\text{EF}} = u_0$. Implement the scheme in MATLAB. Find $\tilde{u}^{j=1}_{\text{EF}}$ for $J = 100$ "by hand."

Compare your MATLAB code result with your "by hand" prediction. Compare your MATLAB code result with the exact solution. Now run your MATLAB code for $J = 5$ — can you explain the (poor) results for $J = 5$?

Consider the Euler Backward scheme applied to Equations (3), (4): denote the approximation $\tilde{u}^j_{\mathrm{EB}}$, $0 \leq j \leq J$, as in the lecture notes and textbook (except here we add the subscript EB). Recall that $\tilde{u}^0_{\mathrm{EB}} = u_0$. Implement the scheme in MATLAB. Find $\tilde{u}^{j=1}_{\mathrm{EB}}$ for $J = 100$ "by hand." Compare your MATLAB code result with your "by hand" prediction. Compare your MATLAB code result with the exact solution. Now run your MATLAB code for $J = 5$ — can you explain why Euler Backward performs better than Euler Forward for $J = 5$?

Apply `ode45` to the problem Equations (3), (4) but now for general a, t_f, and u_0. This is a good opportunity to exercise your MATLAB anonymous function skills. Compare the `ode45` solution to the exact solution or to your Euler Forward or Euler Backward approximations.

4. This question relates to eigenproblems and linear stability analysis.

   A simple predator prey model is given by

$$\frac{dw_1}{dt} = w_1 - 2w_1 w_2 \ , \tag{5}$$

$$\frac{dw_2}{dt} = -w_2 + w_1 w_2 \ , \tag{6}$$

   where w_1 represents the size of the prey population and w_2 represents the size of the predator population. We also provide initial conditions, though these are not important for our purposes here. (In actual practice, we would only be interested in positive solutions to these equations, however you can ignore this issue in your investigation below.)

   This solution has two equilibria given by $\overline{w}_{\mathrm{ex}} = (0 \quad 0)^{\mathrm{T}}$ and $\overline{w}_{\mathrm{co}} = (1 \quad 1/2)^{\mathrm{T}}$ (where "ex" refers to extinction and "co" refers to co-existence). Determine the linear stability of these two equilibria "by hand" and with the MATLAB built-in function `eig`.

   We review the steps of linear stability analysis (for any given equilibrium): linearize the equations about the equilibrium; assume exponential behavior in time to derive the eigenproblem; find the eigenvalues; inspect the eigenvalues to conclude "unstable," "stable," or "marginally/neutrally stable." (Recall that "marginally/neutrally stable" indicates that we are unable to conclude from linear stability analysis whether the equilibrium is stable or unstable.)

   You may wish to also find, say in the case of the extinction equilibrium, the eigenvector associated with the eigenvalue with the maximum (in the sense of most positive) real part.