CS 383C
CAM 383C/M 383E **Numerical Analysis: Linear Algebra** **Fall 2008**

Homework 2

Lecturer: Inderjit Dhillon Date Due: Sep 17, 2008

Keywords: *Matrix Norm, Singular Value Decomposition*

1. Prove that for any $A \in \mathbb{C}^{m\times n}$,
$$\|A\|_\infty = \max_i \|a_i^*\|_1,$$
where a_i^* is the i-th row of A.

2. Assume $A \in \mathbb{C}^{n\times n}$ and $\exists\ p \geq 1$, s.t. $\|A\|_p < 1$, where $||.||_p$ is a vector-induced matrix norm.
   (a) Prove that $I - A$ is invertible.
   (b) Assuming that the series $\sum_{k=0}^{\infty} A^k$ converges, prove that:
   $$(I-A)^{-1} = \sum_{k=0}^{\infty} A^k.$$
   (c) Prove that:
   $$\|A\|_q \|A^{-1}\|_q \geq 1, \quad \forall 1 \leq q < \infty.$$
   (d) Prove that:
   $$\frac{1}{1+\|A\|_p} \leq \|(I-A)^{-1}\|_p \leq \frac{1}{1-\|A\|_p}.$$

3. Consider the following procedure to approximate the SVD of a given square matrix $A = U\Sigma V^T$, where $A, U, \Sigma, V \in \Re^{n\times n}$:

   (i) Initialize $U,\ \Sigma,\ V$ to I.

   (ii) Assuming $U,\ \Sigma$ fixed, compute V and orthogonalize it.

   (iii) Assume $U,\ V$ fixed, compute Σ. Ensure that Σ is diagonal and positive.

   (iv) Assuming $\Sigma,\ V$ fixed, compute U and orthogonalize it.

   (v) If $\|A - U\Sigma V^T\|_F \geq tol$, repeat steps (ii)-(iv).

   For simplicity, assume that A is an invertible matrix.

   (a) Implement the above given procedure using Matlab and verify that it converges to the SVD given by Matlab's `svd` command. For verification, run the above given procedure with $tol = 1e-5$ for 10 different 50x50 random matrices. Compress all your matlab code in one file and email it to the TA with the subject "NLA:HW2". You can use Matlab's `qr` function for orthogonalization. YOU CAN USE MATLAB's STANDARD FUNCTIONS, BUT DO NOT USE ANY CODE FROM THE WEB.
   (b) Compare the time required by this procedure to that of Matlab's `svd` command. Generate a plot of the time required by your implementation to that of Matlab's `svd` command while varying size of input matrix from 10 to 100. Average your results over 10 different runs. Use matlab's `tic` and `toc` command to measure the elapsed time.