MA398 MATRIX ANALYSIS AND ALGORITHMS: ASSIGNMENT 2

Please submit your solutions to this assignment via Moodle by **noon on Thursday November 19th**. Make sure that your submission is clearly marked with your name, university number, course and year of study.

- The written part of the solutions may be typed in LaTeX, or written on paper and subsequently scanned/photographed provided the images are clearly legible. The result should be a single multi-page document entitled *MA398_Assignment2_FirstnameLastname.pdf*.
- The Matlab code scripts relevant to each question should be submitted as *MA398_ Assignment2_ ExerciseN.m*, where you should be careful to implement any functions having the same name, input and output format as indicated in the questions below.

⚠Only in an emergency or if the Moodle submission is unavailable because of a general outage, the assignment should in the respective case be submitted by email to radu.cimpeanu@warwick.ac.uk and n.shkeir@warwick.ac.uk.

---

1. (Warm-up - singular value decompositions). Let

$$A := \begin{pmatrix} 5 & 3 & 4 \\ 3 & 5 & -4 \end{pmatrix}.$$

and

$$B := \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Calculate by hand, showing your working, the SVDs of A and B.

2. (Image compression using SVD). We have recently briefly investigated how the low-rank approximations to rectangular matrices that arise from singular value decomposition can be used for image compression.

(a) Suppose that $A \in \mathbb{R}^{m\times n}$, let $p := \min\{m, n\}$, and let $A = U\Sigma V^T = \sum_{j=1}^{p} \sigma_j u_j v_j^T$ be an SVD of A, with the usual ordering $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$. For $k \in \{0, 1, \ldots, p\}$, let

$$A_k = \sum_{j=1}^{k} \sigma_j u_j v_j^T \in \mathbb{R}^{m\times n}.$$

How many real coefficients are required to store A? How about A_k?

(b) Prove that, for any $A \in \mathbb{R}^{m\times n}$, A_k defined as above is the optimal rank-k approximation to A in the matrix 2-norm, with the size of the error being exactly the singular value σ_{k+1}. That is, show that

$$\sigma_{k+1} = \|A - A_k\|_2 = \min\{\, \|A - X\|_2 \mid X \in \mathbb{R}^{m\times n} \text{ and } \mathrm{Rank}(X) \leq k\}.$$

Hints: To show that $\sigma_{k+1} \leq$ the claimed minimum, use the min-max theorem which states that the singular values σ_k of A satisfy

$$\sigma_k = \min_{\substack{S:\\ \dim(S)=n-k+1}} \max_{\substack{x\in S,\\ \|x\|_2=1}} \|Ax\|_2.$$

No proof of the min-max theorem is required.
To show that $\sigma_{k+1} \geq$ the claimed minimum, consider the action of A and A_k on any $x \in \mathbb{R}^n$, where x is written in the basis $\{v_i\}_{i=1}^n$ of $\mathbb{R}^n$.

(c) Save the three *.jpg* images accompanying this assignment to the working folder on your machine. Create a code which, in the case of each of the above (one at a time), will load an image file into Matlab and convert it to a matrix A with values in $[0,1]$ representing the grayscale values of the pixels in the image. Note that the details of the rescaling are up to you. It is not a mandatory step if you feel it has a detrimental impact on your solution.

Hint: You may look at the Matlab file on Moodle which accompanies Lecture 14 for inspiration in finding certain useful commands for this.

(d) After having familiarised yourselves with the built-in `svd`-function, use it in combination with the `image` command (or any alternative) to plot and save some of the images corresponding to the low-rank approximations A_k for $k = 1,\ p$, and two other values of k between 1 and p. Use values of k that are roughly evenly spaced and that correspond to important visual features of the image being resolved.

(e) Why does `LinesVertical` allow for much better SVD compression than `LinesTilted`?

(f) From your personal perspective, what is the smallest value of k/p for which the image A_k is recognisably A? What is the smallest value of k/p for which A_k is, for you, indistinguishable from A? (There is no right or wrong answer to this question, but you should nevertheless try to - briefly - justify your answer.)

(g) **(Bonus)** The `BlackCat` image represents a true test of the method given its size and complexity. In the spirit of (f) above, but now quantitatively using a suitable norm-based approach (which you should formulate), how would you characterise the overall change in quality of the resulting figure as k is increased? Make a suitable visualisation (a simple plot) of how your metric encapsulates this idea. When do you reach an area of 'diminishing returns'? (in that the image appears to have almost converged, making further steps superfluous).

(h) **(Unmarked)** You can read more about the people and history behind the method in this article about Stanford computer science professor Gene Golub *('Professor SVD')* written by Mathworks CEO Cleve Moler. There is also some detailed extra material on the computation of SVD (how the Matlab `svd` is actually computed) as a *.pdf* in the assignment folder. New friends such as the QR factorisation or the Householder transform will be present there.

3. (Polynomial (Over-)Fitting).

(a) Let $x_0, \dots, x_n \in \mathbb{R}$ be any $n+1$ points, which we shall call *nodes*.

$$V := \begin{pmatrix} 1 & x_0 & \dots & x_0^d \\ 1 & x_1 & \dots & x_1^d \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^d \end{pmatrix} \in \mathbb{R}^{(n+1)\times(d+1)}$$

is called the Vandermonde matrix of the points $x_0, ..., x_n$. In first year linear algebra, you probably proved the qualitative result that V is invertible if and only if $n = d$ and all the x_i are distinct. Now we study its invertibility more quantitatively:

- Write a Matlab helper function `[V] = vandermonde(x,d)` which assembles the above matrix, where `d = size(x)-1`, resulting in a square matrix.
- Generate equidistant points in $[-1,1]$, and numerically calculate the condition number $\kappa(V)$ of V for several representative values $d = n \le 100$ in some matrix norm.

(b) Denote by $\mathscr{P}_n$ the space of polynomials of degree $\le n$, equipped with the norm

$$\|p\|_{[-1,1]} := \max_{x\in[-1,1]} |p(x)|.$$

We consider the following *Polynomial Approximation Problem*. Given distinct nodes $x_0, \dots, x_n \in \mathbb{R}$ and (not necessarily distinct) values $y_0, \dots, y_n \in \mathbb{R}$, find a polynomial $p(x) \in \mathscr{P}_n$ such that $p(x_i) = y_i$.

Show that $p(x) = \sum_{j=0}^{n} c_j x^j$ solves this problem if and only if the coefficients c_j satisfy

$$V \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}.$$

(c) Plot the function

$$f(x) = \frac{1}{1 + 25x^2},$$

as well as its polynomial interpolant $p(x) = \sum_{j=0}^{100} c_j x^j$ in 101 equispaced points on $[-1, 1]$. Use the `ylim` specification to ensure reasonable limits for the y-axis. Repeat the same procedure using 256 and 512 points, respectively. You should find that p behaves very differently from f, particularly near the endpoints of the interval, before the approximation becomes inaccurate altogether.

(d) Since interpolation does not yield satisfying results, try a least squares approximation, that is we aim to find a polynomial $q(x) = \sum_{j=0}^{d} c_j x^j$ of degree d that minimizes

$$\sum_{i=0}^{100} |q(x_i) - y_i|^2 = \|Ac - b\|_2^2.$$

Determine the matrix A and vector b in the above equation. Plot $f(x)$, as well as the least squares approximation $q(x)$, for $d = 20$. Then try $d = 32$, 64 and 128, commenting on your findings.

Hint: You may find the in-built `polyfit` and `polyval` functions useful for this task, but there are a number of alternative solutions as well.