Numerical Analysis 1 – Class 4

Thursday, February 3rd, 2023

Subjects covered

- The SVD and dimensionality reduction
- Applications of the SVD: Image compression, latent semantic indexing
- Solving a system of linear equations: Gaussian elimination.
- Matrix decompositions and linear systems: LU, Cholesky, QR.

Reading

- N. Kutz, Chapter 2.1 (Direct solvers for linear systems).
- C. Moler, Chapter 2 (linked on Canvas).
- "A Singularly Valuable Decomposition -- The SVD of a Matrix", D. Kalman (available on Canvas).
- “Using Linear Algebra for Intelligent Information Retrieval”, M. Berry, et al. (linked on Canvas).

Problems

Most of the following problems require you to write a program. For each program you write, please make sure you also write a test which validates your program.

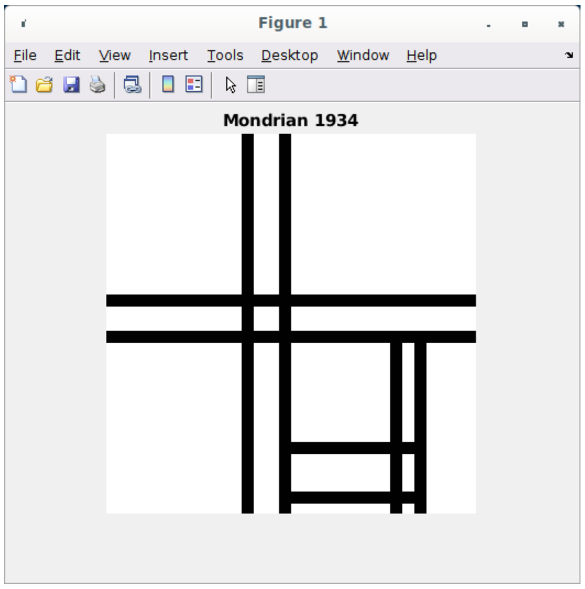
Problem 1

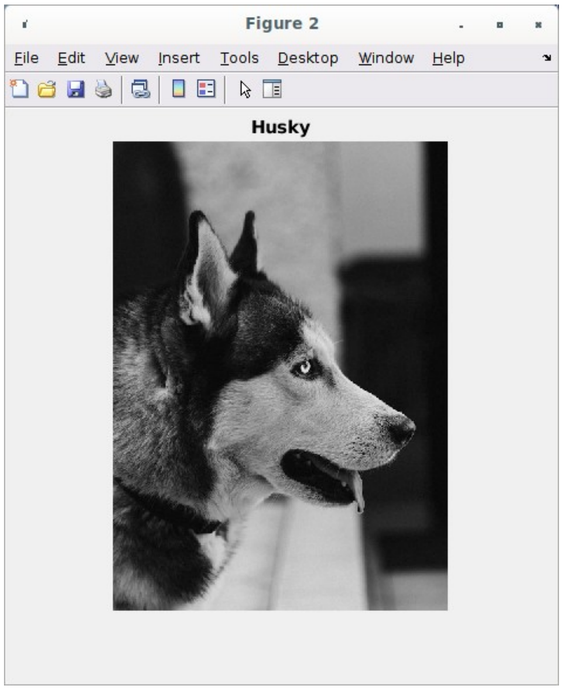
In 1934 the Dutch modernist artist Piet Mondrian created a painting called “Composition in black and white with double lines”. I show the painting below. I also show a black and white image of a husky. I have placed the two black & white images onto Canvas as text (csv) files so you can see the numbers corresponding to each pixel. The files are called “mondrian_1934.csv” and “side_husky.csv”.

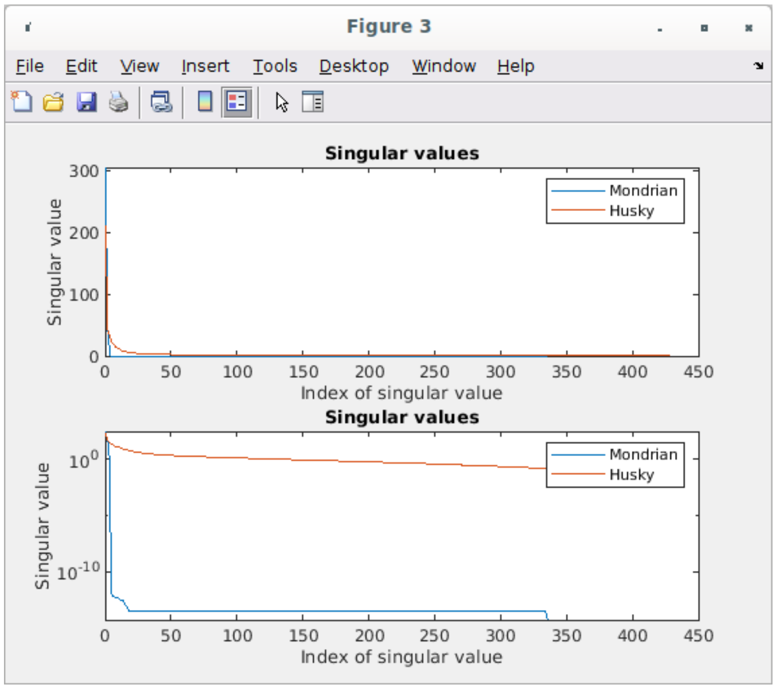
Each image may be interpreted as a matrix whose pixel values are the elements of the matrix. Please write a program which does the following:

- Read in each image as a matrix. The Matlab function “readmatrix()” will do this.
- Show the image. The Matlab function “imshow()” will do this. My results are shown below.
- Compute the SVD of each matrix. Then plot the singular values vs. index. My results are shown below. Note that I plot the singular values twice – once on a linear scale then again on a log scale. I do this so one can see all the singular values.
- Now answer the questions: 1. How many non-zero singular values does the Mondrian matrix have? (Note that many of the values may be very, very small. They are round-off error so you can consider these values to be zero.) 2. What is the rank of the Mondrian matrix? 3. Why does the Mondrian matrix have this rank? 4. How many non-zero singular values does the husky matrix have? 5. What is the rank of the husky matrix? 6. Why does the husky matrix

have this rank?







Regarding testing, the requested function is simple enough that you don't need a test. Just supply a runner which will generate the plots.

Problem 2

On Canvas you will find a Matlab implementation of Gaussian elimination with no pivoting (naive Gauss). Please take that implementation and modify it so it implements partial pivoting (row pivoting).

Assume the solver takes as input the known matrix *A* and vector *b*, and returns the unknown *x* which satisfies $A x = b$. Test your code against the following test systems:

$$A = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

(The matrix *A* may be created in Matlab using `A = ones(5) - eye(5)`.)

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

(The matrix *A* may be created in Matlab using `A = hankel([1; 1; 1; 1; 1]).`)

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 0 \\ 3 & 4 & 5 & 0 & 0 \\ 4 & 5 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix}$$

(The matrix *A* may be created in Matlab using `A = hankel([1; 2; 3; 4; 5]).`)

Also test against random matrices and vectors created using randn() the usual way. Feel free to use Matlab's mldivide (solve) operator "\" to compute the reference answer for use in your test function.

Problem 3

Many problems of engineering interest can be reduced to the problem of solving a tridiagonal matrix equation $A x = b$, where A is of form

$$A = \begin{pmatrix} a_1 & c_1 & 0 & 0 & 0 \\ b_1 & a_2 & c_2 & 0 & 0 \\ 0 & b_2 & a_3 & c_3 & 0 \\ 0 & 0 & b_3 & a_4 & c_4 \\ 0 & 0 & 0 & b_4 & a_5 \end{pmatrix}$$

That is, the matrix elements are non-zero only on the main diagonal and the two adjacent sub-diagonals. Solving this equation may proceed using the so-called "Thomas algorithm", which is similar to Gaussian elimination. A good description of the algorithm is contained in a tutorial by W. Lee (linked on Canvas). Many other descriptions are also available elsewhere on the web.

Your assignment: write a Matlab program to solve an arbitrary tridiagonal system $Ax = b$ using the Thomas algorithm. (Don't worry about pivoting – assume inputs which don't need pivoting or other pre-processing.) Your program should accept the matrix A and vector b as input, and return the vector x. To test your program you can simply verify that the x you compute satisfies the original problem statement $A\,x = b$ (within a testing tolerance). Use any method you can think of to generate random tridiagonal matrices.

Note: There are several implementations of the Thomas algorithm on the web. Feel free to read about them and use them for ideas, but do not directly copy the online implementations. Please write your own code.