DN2222 - Autumn 2011

Programming assignment 3

Singular Value Decomposition, SVD.

Exercise 1 and 2 of this assignment should be reported by the usual mini-report as in the earlier assignments. Exercise 3 in this assignment (pattern recognition) should be reported with a full report.

The scope of this assignment is to

– Understand simple properties of SVD and use them, such as varying the rank.
– Use SVD for Least Squares Fitting.
– Use SVD in classification and data mining.

Study the relevant topics before preparing solutions to the assignments. **D:3.1,2,5** and **L:3**. In the reading advice **D** means the book *Applied Numerical Linear Algebra* by **JW Demmel** and **L** (and sometimes **R**) means the lecture notes *Topics in Numerical Linear Algebra* by **A Ruhe**.

Extra reading: **Lars Eldén:** *Numerical Linear Algebra in Data Mining*, Acta Numerica (2006), pp. 327-384. (download-able from the course home page).

You are encouraged to work in groups of two and you may work at any time. The course assistant Ashraful Kadir will answer questions concerning these assignments at the scheduled lab sessions.

Hand in a report by December 9 at the Student expedition.

For the first two exercises the report should be short and show your conclusions. It may be hand written and should preferably include MATLAB-plots, etc. Do not include all of your MATLAB code or printout. Only include a few lines showing the most interesting parts/results. I do not want to see long MATLAB programs or diary files!

For the last exercise (pattern recognition) it should be a full report. (Complete with description, conclusions, plots, code etc.) It should not be hand written.

All three exercises should be put togther in a single hand-in. Put a staple through it, or put all in one envelope or binder or ... Do not use paper clips as they might fall off. Do not forget the official first page.

Exercise 1. Singular Value Decomposition

Formulate the least squares problem you get when you are fitting a polynomial to a set of measurement data. With m data points, $(t_i, y_i), i = 1 : m$, and a polynomial of degree $n-1$ you will get an $m \times n$ matrix A.

Study the singular value decomposition of the matrix A when we have chosen equidistant t from $[-1, 1]$ with $m = 101$ (hint: use `t=linspace(-1,1,m)`) and $n = 20$.

Exercise 1a. Singular values and rank

The singular values σ_k decrease rather fast in size for larger k. Use `semilogy` to plot them. If data are given with moderate accuracy, say 4 decimals, there is no great benefit to use any singular values σ_k smaller than 10^{-4}. How large numerical rank k should we choose then?

Exercise 1b. Columns of A

The columns of A are the first powers of t. Plot them as functions of t. See that they start to look very similar for higher degrees. Is this good for the requirement of linear in-dependency of columns of a matrix? Are they orthogonal?

Exercise 1c. Columns of U

Plot the first columns of U, the matrix of left singular vectors. They are orthogonal to each other. How does that show up? (Hint: look at the sign changes).

Exercise 1d. Compare with QR

As a comparison, do a QR factorization of A. Now each column q_k of the orthogonal factor Q is a polynomial of degree $k-1$. Plot and compare them with the columns of U.

Note: The k^{th} column q_k of Q is a polynomial of degree $k-1$. All columns u_k of U are polynomials of degree $m-1$ but they should oscillate like the trigonometric functions $sin(k\pi t)$.

Exercise 2. Least Squares Curve Fitting

Use your insights from the previous exercises to compute polynomials to fit the following data series:

1) $y(t) = exp(-t)$
2) $y = |t|$
3) $y = exp(t) + e * randn(m, 1)$
Choose a series of values of $e = 10^{-16}, 10^{-12}, 10^{-8}, \ldots$ until
you no longer recognize the exponential function.

a) Check how/if the approximation is improved by a higher degree of the polynomial.

b) Plot the function y as well as the approximating polynomial.

c) In all three cases, look at the transformed right hand side $b = U^T y$. Its elements b_k should get smaller and smaller for larger k.

d) Compare to the singular values of the matrix A. When an element b_k is significantly larger in absolute value than the singular value σ_k, we get a large component in the solution which does not decrease the residual by a significant amount. For which k does this happen?

e) The case with a random perturbation is of special interest. Here it is not useful to include any singular values σ_k smaller than the perturbation e. Then fix e to a well chosen value for this exercise.

f) Plot the residual $r_k(t) = y(t) - p_k(t)$ in some interesting cases. Here k is the chosen rank/degree.

g) Plot the Euclidean norm of the residual $r_k(t) = y(t) - p_k(t) = y - A * x_k$ as a function of the Euclidean norm of x_k. Here x_k is the solution one gets by using the first k elements in b and a rank k approximation of the matrix A. There is a break even point when increasing rank k only gives marginally smaller residual norms at the cost of a very large solution norm.

Exercise 3. Use SVD for pattern recognition

This part should be reported with a full report!

We borrow the following exercise from a NGSSC (National Graduate School of Scientific Computation) course given by Lars Eldén. If you want to know more, read his paper. Algorithms of this kind are used to recognize hand written digits in postal codes, zip codes in US Postal service terminology. The data is from their test of prospective algorithms.

Construct an algorithm in Matlab for character recognition of handwritten digits. Using a training set, compute an SVD of each matrix of digits of one kind. Use the first few (5-10) singular vectors as basis and classify unknown test digits according to how well they can be represented in terms of the respective bases (use the residual vector in the least squares approximation as a measure). Try to tune the algorithm for accuracy of classification (ie, vary the number of basis vectors). Check if all digits $0, 1, \ldots, 9$ are equally easy to classify or if some digits are easy to confuse.

Report the number of incorrectly classified digits in a table. Also look at some of those, and see that in many cases they are very badly written.

Check the singular values of different digits. See if it is motivated to use different numbers of basis vector for different digits. Does the risk of confusion change the choice?

The training and test data can be downloaded from the course home page. Read the data into Matlab with the command `load('zipdata')` The following files are then loaded:

- `dtrain` and `atrain`: The first is a 1×1707 vector that holds the digits (the number) and the second is a matrix of dimension 256×1707 that holds the training images. The images are of dimension 1×256 that have been constructed from 16×16 images.

- `dtest` and `atest` holding the test data. Use your algorithm on the columns of the matrix `atest` to see if if can guess the corresponding value in the vector `dtest`. There are 2007 digits in the test set, so the first is a 1×2007 vector and the latter a 256×2007 matrix.

- There is a function `ima2.m` that takes an image vector as input and displays the image.

Note: The zipfile is prepared for Matlab versions on Solaris and/or Unix. Some Windows systems may have trouble with un-packing it. Should you run into such trouble, please un-pack the files on a Unix computer here at KTH and then (if you want) bring them to your own computer.

Good luck! /Ninni