Computer Assignment 3

1. Write a function with header `[B] = myMakeLinInd(A)`, where `A` and `B` are matrices. Let the `rank(A) = n`, then `B` should be a matrix containing the first `n` columns of `A` that are all linearly independent.

2. Write a function `alpha = myPolyfit(n,p,x)` that finds the coefficients of a polynomial $p(x)$ of degree `n` that fits the data in `p` and `x`. Your function should solve this problem as a linear system of equations and show an error if there is either no solution or an infinite number of solutions.

3. Repeat the question above but using the least square method instead. Note that now there is always a unique solution, independently of the length `p` and `x`. You can check your results with the MATLAB built-in function `polyfit`.

4. Using the bisection method, write a function `r = myRoots(alpha)` that outputs the (real) roots of a polynomial whose coefficients are the elements of the (real-valued) array `alpha`. You can check your method with the MATLAB built-in function `roots`.

   Hint: Find the monotonicity intervals by finding the roots of the derivative of the polynomial.

5. The eigenvalues λ of a (square) matrix A correspond to the roots of the function $p(\lambda) = \det(A - \lambda I)$, where I denotes the identity matrix. Explain why if A is of size n, then $p(\lambda)$ is a polynomial of degree n. Next, using question 3 and question 4, code a function that finds the real eigenvalues A and their corresponding eigenvectors.

6. The singular value decomposition of a matrix A of size `n`×`m`, is a factorisation of A in the form $A = USV^t$, where both U and V are (full rank) (orthonormal) square matrices and S is a non-necessarily-square diagonal matrix whit non-negative elements. The non-zero elements of the diagonal of S, called singular values of A, correspond to the square root of the non-zero eigenvalues of AA^t (or A^tA). The matrix V is formed by the eigenvectors of A^tA and the matrix U is formed by the eigenvectors of AA^t. Using `eig`, implement a function `[U,S,V] = mySVD(A)` which computes the SVD decomposition of a matrix `A`.

7. Note that the rank of a matrix A is given by the number of non-zero singular values of A (why?). Write a function that take as input a matrix A, and outputs a new matrix A_k, which is k-rank version of A, computed by keeping the k-largest singular values of A. Use this function to show a low rank version of the image of question 10 of Assignment 1.

8. Find regression curves for the average runtime data $T_1(n)$ and $T_2(n)$, corresponding to the runtime of the code of question 10 of Assignment 2, and its efficient version, respectively, where n is the size of the input matrix `M`. Plot your regression curves along with the runtime data. Can you quantify now how faster is the efficient implementation with respect to the inefficient one?

9. Implement a MATLAB function that take as input two arrays `f` and `x`, representing the values of a real valued function $f(x)$; the array `x` should be evenly spaced. Your function should:

   (a) create a new array `f_s` which replace each element of `f` with the average of its `k` nearest neighbours (`k` should also be an input of your function) to the left and to the right. The function `f_s` is a way of regularising a noisy or irregular function.

   (b) returns the numerical derivative of f_s using a centred first order finite difference scheme that you should also implement.

   Test your code with `x = linspace(0,2*pi,1000)` and `f = sin(x) + 0.1*randn(size(x))`, for different values of k.

10. Write a function `I = myTrapez(f, a, b, n)`, which computes the approximation of $\int_a^b f(x)\,dx$ by a trapezoidal rule: $\int_a^b f(x)\,dx \approx h\left[\frac{f(a)+f(b)}{2} + \sum_{k=1}^{n-1} f(x_k)\right]$, where $x_k = a + hk$, and $h = \frac{b-a}{n}$.Your function should not use any built-in Matlab functions. Test your function by computing $\int_0^1 \sqrt{1-x^2}\,dx$, with $n = 10, 20$, and 40. Given that the exact value of the integral is $\pi/4$, how does the error of the approximateresult scale with n?