EXERCISES OF CHAPTER 1

Course: Introduction to Machine Learning for Scientific Computing

Prof. Antonio Marquina

Facultat de Matemàtiques, Aula 1.1

Schedule: Tuesdays, 9:30-12:30

Next session 2019.03.05

2019.02.28

[1]. **Fibonacci sequence generation using matrices and vectors**

Consider the Fibonacci sequence

$$0, 1, 2, 3, 5, 8, \ldots,$$

which can be defined recursively as $F_{n+1} = F_n + F_{n-1}$, where $F_0 = 0$ and $F_1 = 1$. We can also define the sequence in terms of matrices and vectors as follows. Define $\mathbf{v}_k = [F_k, F_{k-1}]^T$ and observe that

$$\mathbf{v}_{k+1} = A\,\mathbf{v}_k, \tag{1}$$

where

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

To find the $k-$th number in the Fibonacci sequence, use the fact that $\mathbf{v}_{k+1} = A\,\mathbf{v}_k = A^2\,\mathbf{v}_{k-1} = \cdots = A^k\,\mathbf{v}_1$.

- 1.1.- Write a Python program to calculate the $k-$th number in the Fibonacci sequence using the `NumPy linalg module` for matrix multiplication.
- 1.2.- Write a Python program to compute the diagonalization of the matrix A. To this end the program must calculate the eigenvalues of A and its eigenvectors such that the matrix D is a diagonal matrix containing in the main diagonal the calculated eigenvalues and the invertible matrix P of the corresponding eigenvectors, such that $A = P\,D\,P^{-1}$.

- 1.3.- Using the Python function written in the part 1.2 compute $A^k = P\,D^k\,P^{-1}$ as direct formula for matrix A to the power k. Then, use this to get an alternative formula for the $k-$th number in the Fibonaci sequence.

Note: The analytical calculation of the above matrices justifies the formula displayed in the Python code `fibonacci_direct.py`.

[2]. Bernstein polynomials: Linear algebra and graphic representation of functions

The Bernstein polynomials of degree $n = 4$ are

$$
\begin{aligned}
B_0^4(x) &= (1-x)^4 \\
B_1^4(x) &= 4x(1-x)^3 \\
B_2^4(x) &= 6x^2(1-x)^2 \\
B_3^4(x) &= 4x^3(1-x) \\
B_4^4(x) &= x^4
\end{aligned}
$$

- 2.1.- Using the library `Matplotlib` write a Python program to plot the above functions taking as domain the closed interval $[0, 1]$.

- 2.2.- Calculate explicitely the transition 5×5 matrix P_{SB} from the Bernstein basis

  $$B = [B_0^4(x), B_1^4(x), B_2^4(x), B_3^4(x), B_4^4(x)]$$

  into the standard polynomial basis $S = [1, x, x^2, x^3, x^4]$, and its inverse, Q_{BS}.

- 2.3.- Consider the polynomial $p(x) = 3x^4 - 8x^3 + 6x^2 - 12x + 4$. Write a Python program to compute the polynomial $p(x)$ in the Bernstein basis B, by using the matrix computation $[p(x)]_B = Q_{BS}[p(x)]_S$. The resulting polynomial should be:

  $$p(x) = 4B_0^4(x) + B_1^4(x) - B_2^4(x) - 4B_3^4(x) - 7B_4^4(x)$$

  .

[3]. **Taylor and Tchebyshev polynomials: approximate dimensionality reduction**

Consider the function $\exp(x)$ defined on the closed interval $[-1, 1]$. Let us consider the Taylor polynomial of degree 20 of the given exponential function:

$$P_{20}(x) := 1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots + \frac{x^{20}}{20!}$$

This polynomial belongs to the vector space of polynomials of degree no larger than 20, generated by the standard basis $1, x, x^2, \cdots, x^{20}$. Let us consider an alternative basis, consisting of the Tchebyshev polynomials, defined as

$$T_0(x) = 1, T_1(x) = x, T_{n+1}(x) = 2\,x\,T_n(x) - T_{n-1}(x), n = 1, 2, \cdots$$

Then, we ask the following questions:

- 3.1.- Obtain an analytic expression of the matrix P_T^n that change the standard basis to the Tchebyshev basis, for any n.
- 3.2.- Write a Python function that computes the coefficients of the linear combination in the Tchebyshev basis for the Taylor polynomial $P_{20}(x)$ defined above.
- 3.3.- Write a table with two columns displaying the coefficients of both polynomials from $n = 0$ to $n = 20$. Discuss the values of the table.