Serie di MacLaurin dell'esponenziale

$$\boxed{e^x = \exp(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{3!} + \cdots + \frac{x^k}{k!} + \dots}$$

Scrivere una `function y = expmac(x, n)` che calcola la somma dei primi n termini di questa serie.

Quanto accurato è `expmac(20, 500)`? Quanto accurato è `expmac(-20, 500)`?

Sapreste spiegare questa differenza di accuratezza?

$y = 1$

for $\rightarrow$ $\left[\begin{array}{l} y = y + x \\ y = y + \frac{x^2}{2} \\ \vdots \end{array}\right.$

Scrivere una function `A = matrice(n)` che crea la versione $n \times n$ di questa matrice.

Matrice di esempio

$$(A_n)_{i,j} = i + j$$

$$A_n = \begin{bmatrix} 1+1 & 1+2 & 1+3 & \dots & 1+n \\ 2+1 & 2+2 & 2+3 & \dots & 2+n \\ \vdots & \vdots & \vdots & \vdots & \\ n+1 & n+2 & n+3 & \dots & n+n \end{bmatrix}$$

$$\text{ad esempio, } A_3 = \begin{bmatrix} 2 & 3 & 4 \\ 3 & 4 & 5 \\ 4 & 5 & 6 \end{bmatrix}$$

Per casa: sapreste creare matrici del tipo $\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$?

Scrivere una funzione `C = matrice2(n)` che crea la versione $n \times n$ di questa matrice.

Matrice di esempio

$$(C_n)_{i,j} = \begin{cases} 0 & \text{se } i = j, \\ \frac{1}{i-j} & \text{se } i \neq j. \end{cases}$$

$$C_n = \begin{bmatrix} 0 & \frac{1}{1-2} & \frac{1}{1-3} & \cdots & \frac{1}{1-n} \\ \frac{1}{2-1} & 0 & \frac{1}{3-2} & \cdots & \frac{1}{2-n} \\ \vdots & \vdots & \vdots & \vdots & \\ \frac{1}{n-1} & \frac{1}{n-2} & \frac{1}{n-3} & \cdots & 0 \end{bmatrix}$$

$$\text{ad esempio, } C_3 = \begin{bmatrix} 0 & -1 & -\frac{1}{2} \\ 1 & 0 & -1 \\ \frac{1}{2} & 1 & 0 \end{bmatrix}$$

Scrivere una funzione `L = matrice3(n)` che crea questa matrice.

Matrice di esempio

$$(L_n)_{i,j} = \begin{cases} 2 & \text{se } i = j, \\ -1 & \text{se } |i-j| = 1, \\ 0 & \text{altrimenti.} \end{cases}$$

$$L_n = \begin{bmatrix} 2 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -1 \\ 0 & \dots & 0 & -1 & 2 \end{bmatrix}$$

$$\text{ad esempio, } L_4 = \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix}$$

Scrivere una funzione `w = prodottoL(v)` che calcola il prodotto $w = L_n v$, dato un vettore $v \in \mathbb{R}^n$.

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} = \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}$$

Utilizzando il comando di Matlab `L * v`, il prodotto costa $\mathcal{O}(n^2)$ operazioni aritmetiche (perché?). È possibile scrivere questa funzione in modo che utilizzi $\mathcal{O}(n)$ operazioni aritmetiche?