Esame di Programmazione Informatica

16 giugno 2023

Esercizio 1 (14/30)

Data la seguente successione:

$$\underbrace{\sqrt{6}}_{x_1}, \underbrace{\sqrt{6+\sqrt{6}}}_{x_2}, \underbrace{\sqrt{6+\sqrt{6+\sqrt{6}}}}_{x_3}, \ldots$$

scrivere una funzione MATLAB che prenda un intero n come argomento in ingresso e restituisca come argomento in uscita il vettore $[x_1, x_2, \ldots, x_n]$ dei primi n termini della successione.

Sulla base della precedente funzione e sapendo che il limite della successione è $l = 3$, determinare, mostrandone il procedimento, il minimo numero n di termini della successione tale per cui $|x_n - l| < \delta$, con $\delta = 10^{-6}$. Per fare ciò vi sono due possibilità:

- utilizzare in maniera diretta la funzione scritta precedentemente (più semplice ma valutato di meno);
- scrivere una nuova funzione, simile alla precedente, che prenda in ingresso il valore δ e restituisca il minimo numero n di termini (più complesso ma valutato di più).

Soluzione: utilizziamo un ciclo `for` per calcolare gli elementi della successione, dal momento che sappiamo a priori il numero n di iterazioni. Memorizzeremo quindi tali elementi in un vettore `x` di lunghezza n dove il primo termine lo possiamo calcolare direttamente. I termini dal secondo all'ultimo (n-èsimo) li calcoliamo dalla successione nella forma $x_i = \sqrt{6 + x_{i-1}}$:

`successione.m`

```
function x = successione(n)
  x = zeros(n,1) ; % inizializzazione
  x(1) = sqrt(6) ; % primo termine
  for i = 2:n
    x(i) = sqrt( 6 + x(i-1) ) ;
  end
end
```

Possiamo anche evitare di inizializzare il vettore `x` sfruttando la possibilità offerta da MATLAB di assegnare nuovi elementi in un vettore:

successione.m

```
function x = successione(n)
  x = 0 ;
  for i = 1:n
    x(i) = sqrt( 6 + x(end) ) ; % dimensione di x aumenta ogni iteraz.
  end
end
```

Determinazione del numero minimo di termini, versione "semplice":

```
delta = 1e-6 ;
l = 3 ;

% Funzione di comodo per ottenere l'ultimo elemento di un vettore
ultimo_termine = @(v) v(end) ;

% Determinazione n
n = 1 ;
while abs( ultimo_termine( successione(n) ) - l ) >= delta
  n = n + 1 ;
end
```

ottenendo $n = 9$. Alternativamente si poteva procedere in maniera più "diretta" provando manualmente per quale valore di n si ottiene una differenza minore di δ.

Determinazione del numero minimo di termini, versione con funzione:

n_successione.m

```
function n = n_successione(delta)
  x = 0 ;
  n = 0 ;
  l = 3 ;
  while abs( x - l ) >= delta
    n = n + 1 ;
    x = sqrt(6+x) ;
  end
end
```

che richiamata con `delta = 1e-6` fornisce $n = 9$.

Esercizio 2 (14/30)

Scrivere una funzione MATLAB che prenda come argomenti in ingresso due interi m ed n e restituisca come argomento in uscita una matrice $\mathbf{A}$, di dimensione $m \times n$ (ossia m righe e n colonne), i cui elementi a_{ij} sono definiti da:

$$a_{ij} = \frac{\sin\left(4\pi \frac{i \cdot j}{m \cdot n}\right)}{i}$$

La funzione dovrà inoltre fare il plot grafico delle m righe della matrice $\mathbf{A}$ nella stessa finestra grafica (una curva per ogni riga). Ogni riga è quindi un vettore (riga) di n elementi.

Mostrare poi come utilizzare la funzione nel caso $m = 10$ ed $n = 50$.

Soluzione: possiamo procedere con un doppio ciclo `for` che calcola tutti gli elementi della matrice in maniera scalare e facciamo anche il plot di ciascuna riga:

matrix_and_plot.m

```
function A = matrix_and_plot(m,n)
  A = zeros(m,n) ; % inizializzazione
  hold on ;        % per mantenere tutti i plot nella stessa finestra
  for i = 1:m
    for j = 1:n
      A(i,j) = sin(4*pi*(i*j)/(m*n)) / i ;
    end
    plot(A(i,:)) ;
  end
end
```

Alternativamente, si poteva operare in maniera più compatta attraverso l'uso di operazioni matriciali per mezzo della funzione `meshgrid`:

matrix_and_plot.m

```
function A = matrix_and_plot(m,n)
  [i,j] = meshgrid(1:m,1:n) ;
  A = sin(4*pi*(i .* j)/(m*n)) ./ i ;
  plot(A) ;
  A = A' ; % indici i,j ottenuti da meshgrid necessitano trasposta
end
```

oppure anche senza l'uso di `meshgrid` ma sempre attraverso l'uso di operazioni vettoriali/-matriciali elemento per elemento (`.*` e `./`):

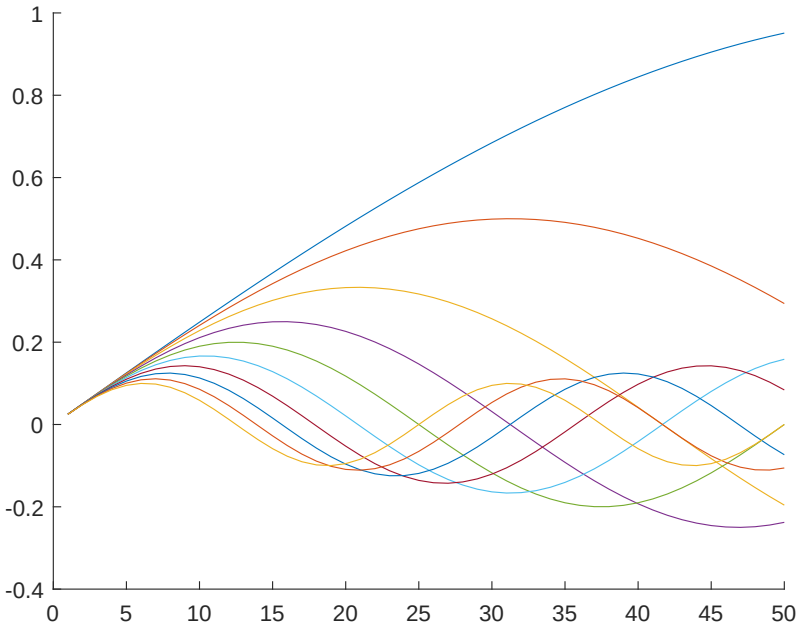
matrix_and_plot.m

```
function A = matrix_and_plot(m,n)
  i = (1:m)' ; % vettore colonna degli indici riga
  j =  1:n   ; % vettore riga degli indici colonna
  A = sin(4*pi*(i .* j)/(m*n)) ./ i ;
  plot(A') ;   % plot fa il plot delle colonne
end
```

Utilizzo nel caso $m = 10$ ed $n = 50$:

```
A = matrix_and_plot(10,50) ;
```

ottenendo:



Domande a risposta multipla (5/30)

Domanda 1 (3/30)

Data la seguente funzione MATLAB:

```
function x = f(x,y,z)
   x = sin(y) ;
   y = sin(z) ;
   z = sin(x) ;
   if cos(z)>0
      y = sin(x) ;
   else
      y = cos(x) ;
   end
end
```

che funzione matematica $f(x, y, z)$ essa implementa?

- $\sin y$
- $\sin x$ se $\cos z > 0$, $\cos x$ altrimenti
- $\cos z$
- Nessuna delle precedenti

Soluzione: $\sin y$.

Domanda 2 (2/30)

E' possibile implementare mediante funzione anonima la funzione che restituisce l'ultimo elemento di un vettore?

- No, perchè è necessaria più di un'operazione
- No, perchè è necessaria più di una riga di istruzioni
- Sì, perchè è sufficiente un'istruzione
- Sì, ma solo se il vettore è colonna

Soluzione: Sì, perchè è sufficiente un'istruzione: `ultimo_elemento = @(v) v(end)`