Esame di Programmazione Informatica

4 febbraio 2021

Informazioni generali

- 2 esercizi di script MATLAB e 2 domande a risposta multipla.
- La durata dell'esame è di 2 ore.
- È consigliato l'uso di MATLAB nello svolgimento di tutti i punti.
- Le slide e gli script del corso sono ovviamente consultabili liberamente.
- Il punteggio assegnato ad ogni esercizio ed alle domande a risposta multipla è indicato tra parentesi.

Esercizio 1 (14/30)

Dato un generico polinomio $p(x)$ di grado n

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n,$$

esso può essere rappresentato dal vettore riga $\mathbf{p}$ dei suoi coefficienti:

$$\mathbf{p} = [a_0, a_1, a_2, \ldots, a_n].$$

La primitiva (o integrale indefinito) di $p(x)$, in particolare quella che si annulla in $x = 0$, è

$$P(x) = \int p(x)\,dx = a_0 x + \frac{a_1}{2} x^2 + \frac{a_2}{3} x^3 + \cdots + \frac{a_n}{n+1} x^{n+1},$$

che essendo sempre un polinomio (di grado $n+1$), può essere nuovamente rappresentata dal vettore riga $\mathbf{P}$ dei suoi coefficienti:

$$\mathbf{P} = \left[0, a_0, \frac{a_1}{2}, \frac{a_2}{3}, \ldots, \frac{a_n}{n+1}\right].$$

Scrivere una funzione che prenda in ingresso il vettore riga $\mathbf{p}$ dei coefficienti di $p(x)$ e restituisca in uscita il vettore riga $\mathbf{P}$ dei coefficienti della sua primitiva $P(x)$.

Considerando poi l'integrale definito

$$I = \int_a^b p(x)\,dx = P(b) - P(a),$$

si utilizzi la funzione scritta precedentemente per calcolare l'integrale definito I di $p(x) = 1 + x$ tra gli estremi $a = 0$ e $b = 1$, utilizzando la seguente funzione `valuta_polinomio` che calcola un generico polinomio $q(x)$, definito dal vettore riga `q` dei suoi coefficienti, per i valori di x dati in `x`:

valuta_polinomio.m

```
function y = valuta_polinomio( q , x )
   y = 0 * x ;
   n = length( q ) - 1 ;
   for i = n : -1 : 0
      y = q(i+1) + x .* y ;
   end
end
```

Soluzione: è sufficiente scrivere esplicitamente la formula del vettore riga $\mathbf{P}$ utilizzando la divisione elemento per elemento e ricordandosi di aggiungere lo zero all'inizio:

primitiva_polinomio.m

```
function P = primitiva_polinomio( p )
   n = length( p ) - 1 ;
   P = [ 0 , p ./ (1:n+1) ] ;
end
```

Per calcolare l'integrale definito di $p(x) = 1 + x$, cioè $\mathbf{p} = [1, 1]$, è sufficiente calcolarne la primitiva mediante la precedente funzione `primitiva_polinomio`:

```
p = [1,1] ;
P = primitiva_polinomio( p ) ;
```

ottenendo correttamente $\mathbf{P} = [0, 1, 1/2]$, cioè $P(x) = x + x^2/2$. L'integrale definito si calcola quindi facendo la differenza tra i valori di $P(x)$ agli estremi $a = 0$ e $b = 1$, calcolati con `valuta_polinomio`:

```
a = 0 ;
b = 1 ;
I = valuta_polinomio( P , b ) - valuta_polinomio( P , a ) ;
```

ottenendo correttamente $I = 1.5$.

Esercizio 2 (14/30)

Data una funzione $f(x)$ sviluppabile in serie di Taylor nell'intorno di x_0, lo sviluppo di Taylor al secondo grado è

$$f(x) = \underbrace{f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2}(x - x_0)^2}_{t(x)} + r(x)$$

dove il resto $r(x)$ è quindi dato da $r(x) = f(x) - t(x)$.

Scrivere una funzione che prenda in ingresso:

- il *function handle* della funzione f,
- il *function handle* della derivata prima f',
- il *function handle* della derivata seconda f'',
- il punto x_0,
- due valori scalari a e b,

e faccia il plot grafico dell'errore relativo percentuale $e(x)$, definito da

$$e(x) = 100\frac{r(x)}{f(x)} = 100\left(1 - \frac{t(x)}{f(x)}\right),$$

nell'intervallo $[a, b]$ suddiviso uniformemente in 1000 punti (la funzione non fornisce alcuna uscita).

Si utilizzi quindi la precedente funzione per fare il plot dell'errore relativo percentuale della funzione $f(x) = f'(x) = f''(x) = e^x$ nell'intervallo $[-1, 1]$ con $x_0 = 0$. Si utilizzino tre (oppure anche una sola, visto che sono uguali) funzioni anonime per definire $f(x)$, $f'(x)$ e $f''(x)$. Non è necessario personalizzare il grafico.

Soluzione: calcoliamo lo sviluppo $t(x)$ al secondo grado utilizzando solo operazioni vettoriali, per poi calolare l'errore relativo percentuale come da formula ricordando di utilizzare la divisione elemento per elemento. Plottiamo infine l'errore:

plot_errore.m

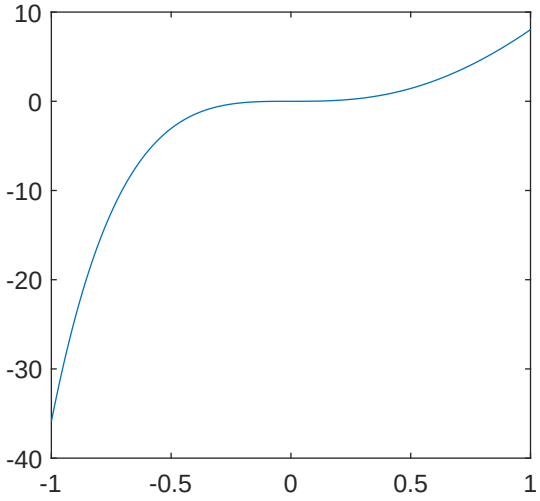
```
function plot_errore(f, df, ddf, x0, a, b)
   x = linspace(a, b, 1000) ;
   t = f(x0) + df(x0)*(x-x0) + 0.5*ddf(x0)*(x-x0) .^ 2 ;
   e = 100 * ( 1 - t ./ f(x) ) ;
   plot(x, e) ;
end
```

Utilizziamo la precedente funzione nel caso di $f(x) = f'(x) = f''(x) = e^x$ nell'intervallo $[-1, 1]$ con $x_0 = 0$. Utilizziamo tre funzioni anonime:

```
% Definizione di f(x) = f'(x) = f''(x) = e^x mediante funzioni anonime
f   = @(x) exp(x) ;
df  = @(x) exp(x) ;
ddf = @(x) exp(x) ;
```

```
% Utilizzo nell'intervallo [-1,1] con centro x0=0
x0 =  0 ;
a  = -1 ;
b  =  1 ;
plot_errore(f, df, ddf, x0, a, b) ;
```

ottenendo correttamente il seguente grafico dell'errore percentuale:



Domande a risposta multipla (5/30)

Domanda 1 (2/30)

Quanto spazio è necessario per rappresentare in un calcolatore, senza alcuna compressione, un'immagine raster in scala di grigi di dimensione $m \times n$ utilizzando una quantizzazione con 2^k livelli di grigio?

- 1000×1000 con $2^8 = 256$ livelli di grigio: 8 Mbit = 1 MB.
- 1000×1000 con $2^6 = 64$ livelli di grigio: 6 Mbit = 750 kB.
- 1000×1000 con $2^4 = 16$ livelli di grigio: 4 Mbit = 500 kB.

Domanda 2 (3/30)

Caso A. Data la seguente funzione MATLAB:

```
function y = f(a, n)
   y = sum( a(1) ) ;
end
```

con $\mathbf{a} = [a_1, \ldots, a_n]$ vettore e $n \geq 1$ intero, che funzione matematica $f(\mathbf{a}, n)$ essa implementa?

- $f(\mathbf{a}, n) = \sum_{i=1}^{n} a_i$
- $f(\mathbf{a}, n) = a_1$
- $f(\mathbf{a}, n) = na_1$
- $f(\mathbf{a}, n) = \sum_{i=1}^{n} a_1$

Soluzione: $f(\mathbf{a}, n) = a_1$. `a(1)` è uno scalare e la somma estesa ad un singolo scalare è lo scalare stesso.

Caso B. Data la seguente funzione MATLAB:

```
function y = f(a, n)
   a = n * a(1) ;
   y = sum( a ) ;
end
```

con $\mathbf{a} = [a_1, \ldots, a_n]$ vettore e $n \geq 1$ intero, che funzione matematica $f(\mathbf{a}, n)$ essa implementa?

- $f(\mathbf{a}, n) = \sum_{i=1}^{n} a_i$
- $f(\mathbf{a}, n) = a_1$
- $f(\mathbf{a}, n) = na_1$
- $f(\mathbf{a}, n) = \sum_{i=1}^{n} na_1$

Soluzione: $f(\mathbf{a}, n) = na_1$. `a(1)` è uno scalare e la somma estesa ad un singolo scalare, `n*a(1)`, è lo scalare stesso.