Corso di Laurea in Informatica – Università di Ferrara
Calcolo Numerico
Prova scritta e pratica del 27/01/2021 – A.A. 2020–2021
Tempo a disposizione: 4 ore

Esercizio 1

1) (T) (2 punti) Determinare la rappresentazione del numero $y = (-7034.9065)_{10}$ secondo il formato ANSI standard IEEE in semplice precisione.

2) (M) (2 punti) Siano dati i seguenti coefficienti:

$$c_6 = \log_{10}\big(\pi^{7/4}\max\{\mathrm{e}^2 - 5,\, \tan(1/2)\}\big), \quad c_5 = \cos\Big(\sqrt[3]{\big|-0.7\mathrm{e}^{-0.2}\big|}\Big), \quad c_3 = -\sin\big(0.2 + \mathrm{e}^{-1.6}\big),$$
$$c_1 = \arccos(3.7 \cdot 10^{-1}) + 1/4\,, \quad c_0 = \ln\big(\sqrt{\pi^3} + 2/3\big),$$

del polinomio $p_6(x) = c_6x^6 + c_5x^5 + \ldots + c_1x + c_0$. Realizzare un M-script file Matlab che, usando la function `ruffiniHorner` in allegato, calcoli e stampi il valore del polinomio $p_6(x)$ e delle sue derivate prima $p_6'(x)$ e seconda $p_6''(x)$ in un prefissato punto x_0 accettato in input. Lo script visualizzi anche il grafico del polinomio nell'intervallo $[-1.5, 1.3]$. Provare lo script con il valore $x = 0.7$.

Esercizio 2

1) (T) (3 punti) Determinare il dominio Ω_φ, l'errore inerente e l'errore algoritmico nel calcolo dell'espressione:

$$\varphi(x) = \sqrt[4]{x\big(\ln(5 + x) - 1\big)}$$

Valutare inoltre l'indice di condizionamento e l'indice algoritmico. Determinare gli eventuali valori di x per i quali il problema è mal condizionato e/o il calcolo è instabile. Fornire, se esiste, una diversa formulazione dell'espressione $\varphi(x)$, che sia matematicamente equivalente a quella data.

2) (M) (3 punti) Realizzare una M-function che accetti in input un vettore `x` di ascisse e due parametri scalari `s`, `w` $\neq 0$.

   La M-function determini, <u>senza usare cicli</u>, le componenti x_i di $\boldsymbol{x}$ che soddisfano le due condizioni $x_i + s > 0$ e $x_i\big(\ln(x_i + s) - w\big) > 0$. *Solo* per tali componenti e <u>senza usare cicli</u>, la M-function calcoli il valore della funzione $\varphi(x)$ del punto precedente, nella quale s sostituisce il numero "5" nell'argomento del logaritmo e w sostituisce il numero "1" dentro le parentesi. La M-function ripeta questo calcolo per `N` $= 10000$ volte.

   La M-function restituisca in output come primo parametro `z` tutti i valori così calcolati e come secondo parametro gli indici delle componenti alle quali si riferiscono i valori. Mediante le funzioni predefinite `tic` e `toc`, la M-function determini il tempo medio impiegato per la valutazione della funzione $\varphi(x)$ e lo restituisca come terzo parametro di uscita. Dotare la M-function di opportuni controlli sugli input e di corrispondenti messaggi per l'utente in caso di errore.

   Scrivere infine un M-script che provi la M-function usando per s e w sia i valori del punto precedente, che due altri valori scelti dall'utente, diversi dai precedenti. Lo script stampi a console il tempo per entrambe le coppie di parametri e disegni in un unico grafico, in scala logaritmica sulle ordinate, i due profili della funzione $\varphi(x)$. Dotare il grafico di opportune etichette degli assi, di titolo e di legenda, posizionata all'interno del grafico.

Appendice: codici forniti

```
function [r, q] = ruffiniHorner(p, a)
% RUFFINIHORNER - Schema di Ruffini-Horner
%  Calcola il valore di un polinomio p(x) nel punto x = a e i coefficienti
%  del polinomio quoziente q(x) = p(x) / (x - a)
% SYNOPSIS
%   [r, q] = ruffiniHorner(p, a)
% INPUT
%   p (double array) - Vettore dei coefficienti del polinomio, ordinati
%                      da quello di grado piu' alto a quello di grado zero
%   a (double)       - Punto (numero reale) in cui calcolare il polinomio
% OUTPUT
%   r (double)       - Valore del polinomio nel punto x = a
%   q (double array) - Vettore dei coefficienti del polinomio quoziente
%                      q(x) = p(x) / (x - a)
%
r = [];
if ( isempty(p) )
    q = [];
    warning('Il vettore p dei coefficienti e'' vuoto');
    return
elseif ( isempty(a) )
    q = p;
    warning('Il punto ''a'' in cui valutare il polinomio e'' vuoto');
    return
else
    n = numel(p) - 1;  % grado del polinomio
    q = zeros(n, 1);
    q(1) = p(1);
    for i = 2 : n+1
        q(i) = q(i-1)*a + p(i);
    end
    r = q(n+1);
    q = q(1:n);
end
end % fine della function ruffiniHorner
```

Soluzioni e commenti

Nel seguito sono riportate le soluzioni degli esercizi del compito, includendo in generale solo quei commenti che risultino fondamentali per la comprensione della soluzione. Qualora uno stesso esercizio possa eventualmente essere risolto in più modi, in alcuni casi si riportano le descrizioni anche di qualche modo alternativo. Si invitano gli studenti a contattare il docente per eventuali dubbi o chiarimenti.

Nella trascrizione delle soluzioni è stata posta la massima cura. Tuttavia, nel caso si rilevino sviste e/o errori di battitura, si invitano gli studenti a comunicarle via e-mail al docente.

Soluzione esercizio 1

Teoria

7034	/2
3517	0
1758	1
879	0
439	1
219	1
109	1
54	1
27	0
13	1
6	1
3	0
1	1
0	1

0.9065	×2
0.8130	1
0.6260	1
0.2520	1
0.5040	0
0.0080	1
0.0160	0
0.0320	0
0.0640	0
0.1280	0
0.2560	0
0.5120	0
⋮	⋮

$$
\begin{aligned}
(7034)_{10} &= (1101101111010)_2 \\
(0.9065)_{10} &= (0.11101000000\ldots)_2 \\
y &= (-7034.9065)_{10} \\
&= (-1101101111010.11101000000\ldots)_2 \\
&= (-1.101101111010111101000000\ldots)_2 \cdot 2^{(+1100)_2} \\
\\
y < 0 &\Rightarrow s = 1 \\
p &= (+1100)_2 = (+12)_{10} \\
&\Rightarrow \widetilde{p} = p + bias = 12 + 127 = 139 = 128 + 8 + 2 + 1 = (10001011)_2 \\
m &= (1.10110111101011110100000\overline{0})_2 \\
&\Rightarrow \widetilde{m} = 10110111101011110100000 \quad \text{(attenzione: \underline{troncamento} della mantissa alla 24-esima cifra binaria)} \\
\mathrm{IEEE}_{32}(y) &= 1\,\underbrace{10001011}_{\widetilde{p}}\,\underbrace{10110111101011110100000}_{\widetilde{m}}
\end{aligned}
$$

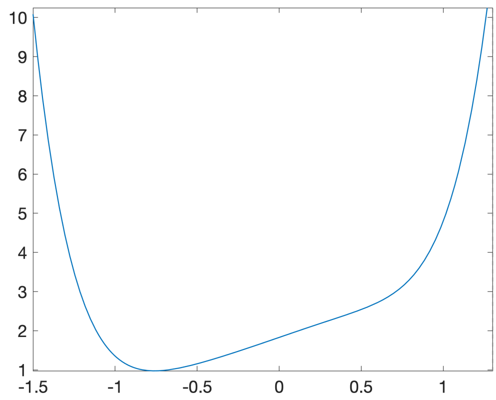
Matlab

Contenuto dell'M-function file `esercizio1.m`:

```
% Cognome Nome
% Matricola
%-----------------------------
%  esercizio 1 - 27/01/2021
%-----------------------------
close all; clear all; clc;
disp('Esecizio 1');

p6 = zeros(7,1);

p6(1) = log10( pi^(7/4) * max(exp(2)-5, tan(1/2)) ); % c6
p6(2) = cos( abs(-0.7 * exp(-0.2))^(1/3) );          % c5
p6(4) = -sin( 0.2 + exp(-1.6) );                     % c3
p6(6) = acos( 3.7e-1 ) + 1/4;                        % c1
p6(7) = log( sqrt(pi^3) + 2/3 );                     % c0

x0 = input('Inserire il punto nel quale valutare il polinomio (double): x0 = ');
[r,    q ] = ruffiniHorner( p6, x0 );
[derp, q1] = ruffiniHorner( q,  x0 );
[ders, q2] = ruffiniHorner( q1, x0 );
fprintf('\nValore del polinomio in x0: p(%g) = %g',x0,r);
fprintf('\nDerivata prima del polinomio in x0: p''(%g) = %g',x0,derp);
fprintf('\nDerivata seconda del polinomio in x0: p''''(%g) = %g\n\n',x0,2*ders);
fh = fplot(@(x)(polyval(p6,x)), [-1.5, 1.2]);
```

Eseguendo lo script si ottengono il seguente output a console e una figura simile alla seguente:

```
Esercizio 1
Inserire il punto nel quale valutare il polinomio (double): x0 = 0.7

Valore del polinomio in x0: p(0.7) = 2.96546
Derivata prima del polinomio in x0: p'(0.7) = 2.93515
Derivata seconda del polinomio in x0: p''(0.7) = 11.9746
```


10
9
8
7
6
5
4
3
2
1
-1.5
-1
-0.5
0
0.5
1