Corso di Laurea in Informatica – Università di Ferrara
Calcolo Numerico
Esercizi dalla prova scritta e pratica del 12/6/2019

Esercizio 1

1. (T) (2 punti) Determinare la rappresentazione del numero $y = 1024 + 2^{-6}$ secondo il formato ANSI standard IEEE in semplice precisione.
2. (M) (2 punti) Realizzare un M-script file Matlab che, usando la function `ruffiniHorner.m` in allegato, calcoli e stampi il valore del polinomio $p_6(x) = -\mathrm{e}^{-\pi}x^6 - 1.4x^5 + x^4 - \ln(207.13)x^2 + 3\pi$ 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 $[0.8, 1.7]$. Provare lo script con il valore $x_0 = 0.2$, riportando nel foglio delle risposte i valori ottenuti.

Esercizio 2

1. (T) (3 punti) Valutare l'errore inerente e l'errore algoritmico nel calcolo dell'espressione:

$$\varphi(x) = \frac{\cos(x)}{5 - \mathrm{e}^{3.7x}} \qquad x \in \Omega \subset \mathbb{R}$$

   determinando l'insieme Ω di definizione. Valutare l'indice di condizionamento e l'indice algoritmico. Determinare inoltre gli eventuali valori di x per i quali il problema è mal condizionato e quelli per i quali il calcolo è instabile.

2. (M) (3 punti) Realizzare un M-function file per la valutazione della funzione $\varphi_{\alpha,\beta}(x) = \cos(x)/\big(\alpha - \mathrm{e}^{\beta x}\big)$, generalizzazione della $\varphi(x)$ del punto precedente, nella quale $\alpha, \beta \in \mathbb{R}$ sono parametri passati dall'utente. Per ogni fissato vettore $\boldsymbol{x}$ di valori, la M-function esegua in modo efficiente il calcolo del valore della funzione per tutte le componenti x_i di $\boldsymbol{x}$ e restituisca in output il vettore $\boldsymbol{y}$ con i risultati delle valutazioni di $\varphi_{\alpha,\beta}(x)$.

   In un M-script di prova, si consideri un vettore $\boldsymbol{x}$ di N componenti equispaziate nell'intervallo $[a, b]$. Lo script invochi la funzione in un ciclo per ciascuna componente x_i di $\boldsymbol{x}$, salvando i risultati nel vettore `fx`. Poi invochi nuovamente la M-function passandole l'intero vettore $\boldsymbol{x}$, salvando i risultati nel vettore `z`. Il ciclo e la seconda invocazione della M-function siano temporizzati separatamente. Lo script visualizzi a console i due tempi di calcolo `tempoCiclo` e `tempoVett`, insieme alla loro differenza relativa rispetto al tempo del ciclo e alla percentuale di `tempoVett` rispetto a `tempoCiclo`. Per le visualizzazioni, si utilizzino opportune istruzioni di stampa formattata. Lo script disegni infine il grafico di $\varphi_{\alpha,\beta}(x)$ nell'intervallo $[a, b]$ usando le ascisse $\boldsymbol{x}$ ed i valori in `z`.

   Provare lo script con $[a, b] = [-1.5, 2.3]$, $\alpha = 2$, $\beta = 1.2$ ed $N = 2001$.

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 commenti che aiutino il più possibile 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. In generale, questo rende il testo delle soluzioni degli esercizi inevitabilmente molto più lungo di quanto non sia effettivamente richiesto allo studente nel momento della prova scritta, pertanto **la lunghezza delle soluzioni qui riportate è da considerarsi NON RAPPRESENTATIVA della lunghezza del compito**. 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

Per la parte intera, è ben noto che $1024 = 2^{10}$: è dunque **superfluo** applicare l'algoritmo delle divisioni successive. Inoltre, la parte frazionaria è già fornita come potenza negativa di 2, quindi è **del tutto superfluo** anche applicare l'algoritmo delle moltiplicazioni successive, dovendosi anche calcolare esplicitamente il valore decimale di 2^{-6}.

1024	/2
512	0
256	0
128	0
64	0
32	0
16	0
8	0
4	0
2	0
1	0
0	1

0.015625	×2
0.031250	0
0.062500	0
0.125000	0
0.250000	0
0.500000	0
0.000000	1

$$
\begin{aligned}
(1024)_{10} &= (10000000000)_2 \\
\left(2^{-6}\right)_{10} &= (0.000001)_2 \\
y &= \left(1024 + 2^{-6}\right)_{10} \\
&= (10000000000.000001)_2 \\
&= (1.0000000000000001)_2 \cdot 2^{+1010}
\end{aligned}
$$

$$
\begin{aligned}
&y > 0 \Rightarrow s = 0 \\
&p = (+1010)_2 = (+10)_{10} \\
&\Rightarrow \widetilde{p} = p + bias = 10 + 127 = 137 = 128 + 8 + 1 = (10001001)_2 \\
&m = (1.00000000000000010000000)_2 \\
&\Rightarrow \widetilde{m} = 00000000000000010000000
\end{aligned}
$$

(attenzione al troncamento della 23-esima cifra binaria della mantissa!)

$$
\mathrm{IEEE}_{32}(y) = 0\,\underbrace{10001001}_{\widetilde{p}}\,\underbrace{00000000000000010000000}_{\widetilde{m}}
$$

Matlab

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

```
% Cognome Nome
% Matricola
close all
clear all
%----------------------------
%  esercizio 1 - 12/06/2019
%----------------------------
disp('Esecizio 1');
p = [-exp(-pi) -1.4 1 0 -log(207.13) 0 3*pi]';
x0 = input('Inserire il punto nel quale valutare il polinomio (double): x0 = ');
[r, q] = ruffiniHorner( p, x0 );
fprintf('\nValore del polinomio in x0 = %g: p(x0) = %g',x0,r);
[derp, q1] = ruffiniHorner( q, x0 );
[ders, q2] = ruffiniHorner( q1, x0 );
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);
fplot(@(x)(polyval(p,x)),[0.8,1.7]);
```

Eseguendo lo script si ottengono i seguenti risultati in output e la seguente figura:

Punto di valutazione: $x_0 = 0.2$
Valori calcolati in x_0:
$p_6(x_0) = \quad 9.21259$
$p_6'(x_0) = \quad -2.11262$
$p_6''(x_0) = -10.4128$