Laboratorio di Analisi Numerica
Lezione 1

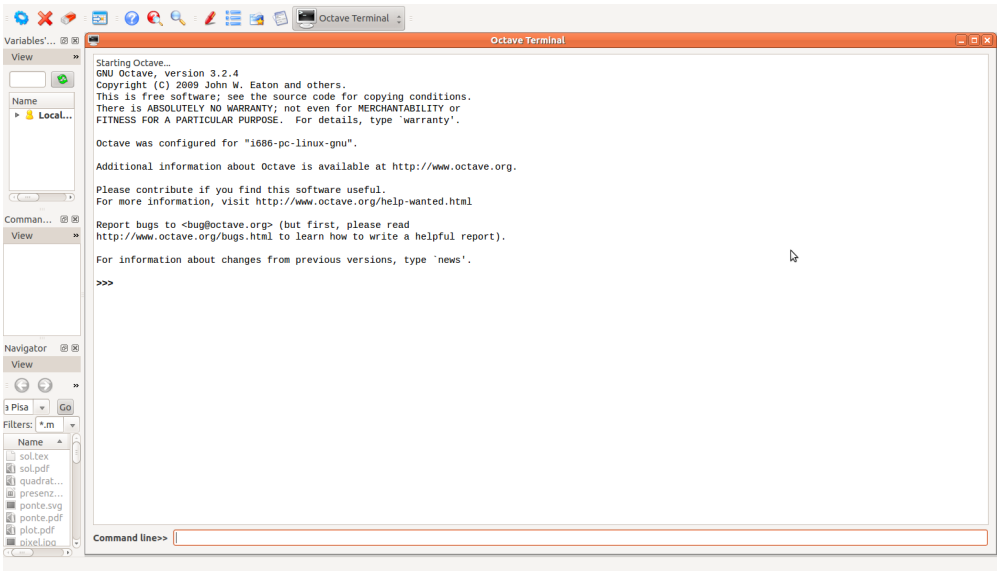
Gianna Del Corso <delcorso@di.unipi.it>
Federico Poloni <fpoloni@di.unipi.it>

2 ottobre 2012

Quantità di esercizi: in questa dispensa ci sono *più esercizi* di quanti uno studente medio riesce a farne durante una lezione di laboratorio, specialmente tenendo conto anche degli esercizi facoltativi. Questo è perché sono pensate per "tenere impegnati" per tutta la lezione anche quegli studenti che già hanno un solido background di programmazione.
Quindi fate gli esercizi che riuscite, partendo da quelli *non* segnati come facoltativi, e non preoccupatevi se non li finite tutti!

1 Primo programma

Lanciamo QTOctave con il comando `qtoctave`. Per ora ci interessa solo la finestra intitolata *Octave Terminal*, quindi possiamo ingrandire quella e ignorare le altre.



Possiamo dare comandi inserendoli nella riga in basso e ottenere dell'output.

```
octave:1> 'Hello, world'
ans = Hello, world
```

2 Primi calcoli in virgola mobile

Octave utilizza la doppia precisione (8 byte per ogni numero).

```
octave:1> realmin
ans = 2.2251e-308
octave:2> realmax
ans = 1.7977e+308
octave:3> eps
ans = 2.2204e-16
```

Octave come una calcolatrice:

```
octave:1> 1+1
ans = 2
octave:2> 10^10
ans = 1.0000e+10
octave:3> 1e10
ans = 1.0000e+10
octave:4> (1e10)^2
ans = 1.0000e+20
```

Se c'è un punto e virgola alla fine della linea, Octave esegue il calcolo ma non scrive il risultato

```
octave:1> 1+1;
octave:2>
```

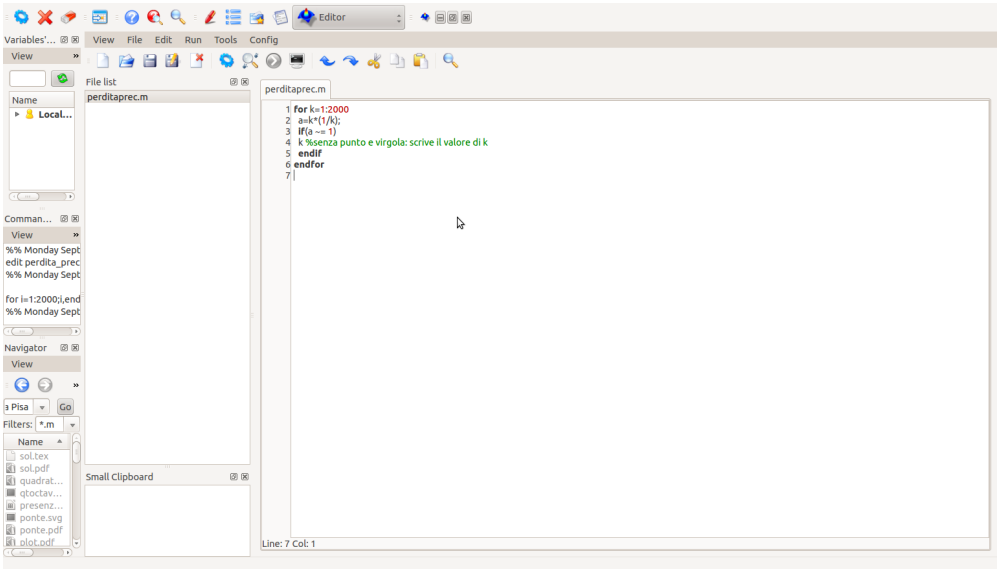
Perdita di precisione da alcuni calcoli:

```
octave:1> a=1e10
a = 1.0000e+10
octave:2> b=1e4
b = 10000
octave:3> c=(a+b)^2
c = 1.0000e+20
octave:4> format long % scrive piu' cifre
octave:5> c
c = 1.00000200000100e+20
octave:6> c - a^2 - 2*a*b - b^2
ans = 7936
```

Principalmente da sottrazioni tra due numeri grossi e molto vicini, (*errori di cancellazione*), ma anche da moltiplicazioni:

```
octave:1> a=98
a = 98
octave:2> 1 - a*(1/a)
ans = 1.1102e-16
octave:3> a=97
a = 97
octave:4> 1 - a*(1/a)
ans = 0
```

Quando succede? Controlliamo con un breve programma. Creiamo uno *script*, cioè un file di testo con estensione `.m` contenente una sequenza di comandi. QTOctave contiene un semplice editor che possiamo richiamare premendo l'icona rossa con una penna. Trovo più comodo avere l'editor in una finestra separata (menu `view/Show outside of main window`).




```
% scrivete queste istruzioni in nel file perditaprec.m
% le righe che cominciano con "%" sono commenti e vengono ignorate

for k=1:300
  a=k*(1/k);
  if(a ~= 1)
    k %senza punto e virgola: scrive il valore di k
  endif
endfor
```

Se il file si chiama `perditaprec.m` *e si trova nella cartella da cui avete lanciato Octave*[1], possiamo eseguirlo semplicemente digitando `perditaprec` dal prompt di Octave.

```
octave:1> perditaprec
k = 49
k = 98
k = 103
k = 107
k = 161
k = 187
k = 196
k = 197
k = 206
k = 214
k = 237
k = 239
k = 249
k = 253
```

Possiamo controllare che per i valori stampati a schermo il valore calcolato di `a*(1/a)` è diverso da 1.

3 Funzioni e accumulatori

```
function f=fact(n);
% calcola il fattoriale di n
% n dev'essere un intero
f=1;
% f fa da accumulatore: parte da 1,
% a ogni passo, lo moltiplico per k
for k=1:n
  f=f*k;
endfor
% ora f vale n!
endfunction
```

Va scritto in un file chiamato `fact.m` e messo *nella cartella da cui abbiamo lanciato Octave*; poi possiamo lanciarlo:

```
octave:1> fact(10)
ans = 3628800
```

Esercizio 1. Scrivete una funzione `pow(x,n)` che calcoli x^n

[1] Altrimenti, potete controllare e cambiare la "cartella di lavoro" di Octave con il menu `file/change directory` o i soliti comandi `cd`, `pwd`, `ls`, che funzionano anche all'interno di Octave.

4 Calcolo dell'esponenziale

```
function a=myexp(x,n)
% calcola exp(x) con la serie di Taylor troncata all'n-esimo termine
  a=1; %accumulatore
  for k=1:n
    a=a+pow(x,k)/fact(k);
  endfor
endfunction
```

Qualche esperimento su quanti termini servono per approssimare bene — confrontiamo con la funzione `exp(x)` di Octave, che calcola l'esponenziale meglio di noi.

```
octave:1> myexp(1,5)
ans = 2.7167
octave:2> myexp(10,50)
ans = 2.2026e+04
octave:3> exp(10)
ans = 2.2026e+04
octave:4> format long
octave:5> myexp(10,50)
ans = 22026.4657948067
octave:6> exp(10)
ans = 22026.4657948067
```

Due problemi:

- Inaccurato: prova `exp(-20,500)`
- Lento: con n termini, il numero di operazioni da eseguire cresce come n^2 (perché?)

Risolviamo (2) introducendo un altro accumulatore:

```
function a=myexp2(x,n)
  %calcola e^x con Taylor troncato
  %ma usa solo O(n) operazioni
  t=1; %accumulatore che contiene il termine generico della sommatoria
  a=1; %accumulatore che contiene le somme parziali
  for k=1:n
    t=t*x/k;
    a=a+t;
  endfor
endfunction
```

Ora va meglio:

```
octave:1> myexp(-20,500)
ans = NaN
```

```
octave:2> myexp2(-20,500)
ans = 5.62188447213042e-09
```

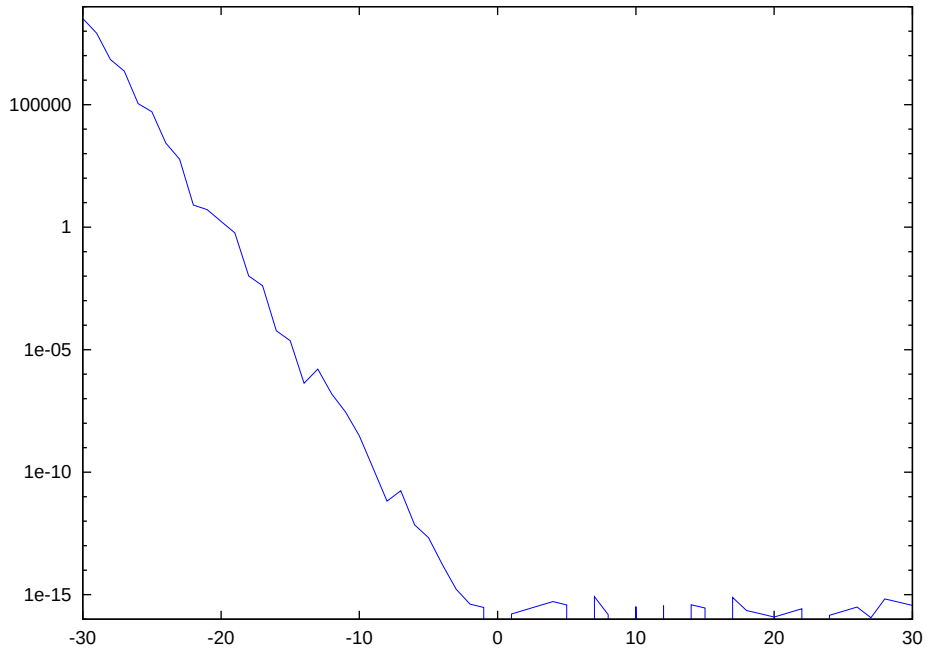
Cosa succedeva?

```
octave:27> fact(500)
ans = Inf
octave:28> pow(-20,500)
ans = Inf
octave:29> Inf/Inf
ans = NaN
```

Ci sono ancora pesanti accuratezze sui numeri negativi:

```
octave:36> myexp2(-30,500)
ans = -3.06681235635622e-05
```

Un esponenziale dovrebbe sempre essere positivo... Ci sono pesanti errori di cancellazione nella formula che abbiamo usato (perché?).



Errore relativo: $\frac{|\text{exptaylor2}(x)-e^x|}{e^x}$

La soluzione: cambiare algoritmo e sceglierne uno che non porti a errori di cancellazione

```
octave:2> exp(-30)
ans = 9.3576e-14
octave:3> myexp2(-30,500)
```

```
ans = -3.0668e-05
octave:4> 1/myexp2(30,500)
ans = 9.3576e-14
octave:5> format long
octave:6> exp(-30)
ans = 9.35762296884017e-14
octave:7> 1/myexp2(30,500)
ans = 9.35762296884017e-14
```

Esercizio 2. Scrivere una funzione `myexp(x)` che controlla se x è negativo o positivo, e a seconda del segno calcola l'esponenziale come $1/e^{-x}$ oppure e^x con la serie di Taylor troncata a $n = 500$.

Esercizio 3 (facoltativo). Si consideri la funzione $f(x) = \log(1+x)/x$ nell'intervallo $[10^{-15}, 10^{-14}]$.

a) Si confrontino i valori ottenuti su punti dell'intervallo calcolando $f(x)$ o

$$g(x) = \frac{\log(1+x)}{(1+x)-1}.$$

b)Si faccia l'analisi teorica dell'errore allo scopo di capire il motivo di questo comportamento.

Esercizio 4 (facoltativo). Scrivere una funzione `solve2(a,b,c)` che risolve l'equazione di secondo grado $ax^2+bx+c = 0$ in nel modo più stabile possibile (hint: ci sono al massimo sottrazioni. Una è necessaria (perché?); l'altra no). Provare su $x^2-(10^{10}+10^{-10})x+1 = 0$.

Esercizio 5 (facoltativo). Come si può calcolare $\frac{1-\cos x}{x^2}$ in modo accurato per valori di x piccoli? Testate il risultato su $x = 10^{-10}$.

Esercizio 6 (facoltativo). Scrivere una funzione che calcola un'approssimazione della costante di Eulero–Mascheroni

$$\gamma = \lim_{n\to\infty} \left(\sum_{k=1}^{n} \frac{1}{k} - \log n \right)$$

troncando la somma all'n-esimo termine. La convergenza di questo limite è molto lenta — non stupitevi se servono molti termini.