Calcolo Numerico : Laboratorio
Lezione 1-2

Gianna Del Corso <gianna.delcorso@unipi.it>

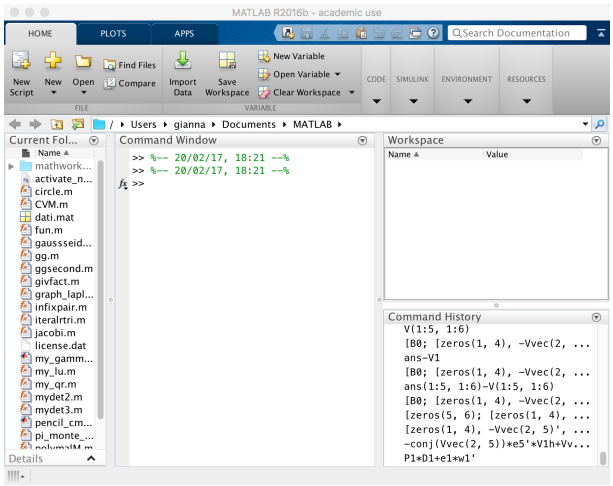
1 Matlab

L'università ha stipulato una convenzione con Mathworks per la licenza Total Academic Headcount (TAH) Student che permette agli studenti dell'Università di Pisa di installare Matlab sui propri computer personali (fino ad un massimo di 4 computer per studente). Le istruzioni per l'installazione sono disponibili sulla piattaforma di e-learning. Lo studente dovrà associare all'account Matlab un indirizzo email dell'università di Pisa, cioè del tipo "studenti.unipi.it".

Provate ad installare sulla vostra macchina personale il software, per il nostro corso basta Matlab senza ulteriori pacchetti. Chi riscontrasse problemi può contattarmi.

2 Primo programma

Lanciamo Matlab.



Possiamo dare comandi inserendoli nella Command Window e ottenere dell'output.

```
>> 'Hello, world'
ans = Hello, world
```

3 Primi calcoli in virgola mobile

Matlab utilizza la doppia precisione (8 byte per ogni numero). Il numero positivo più piccolo rappresentabile si chiama `realmin`, il più grande si chiama `realmax`

```
>> realmin
ans = 2.2251e-308
>> realmax
ans = 1.7977e+308
>> eps
ans = 2.2204e-16
```

Si provino a confrontare questi valori con quelli teorici di $\omega = 2^{-m-1}$ e $\Omega = 2^M(1-2^{-t})$ ... attenzione, occorre porre un po' di attenzione per calcolare Ω, ricordando che per lo standard IEEE, $m = 1021, M = 1024, t = 53$.

Esercizio 1. `realmin` è davvero il più piccolo numero rappresentabile? verificare che esistono i numeri "denormalizzati" che si trovano tra 0 e `realmin`.

Matlab come una calcolatrice:

```
>> 1+1
ans = 2
>> 10^10
ans = 1.0000e+10
>> 1e10
ans = 1.0000e+10
>> (1e10)^2
ans = 1.0000e+20
```

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

```
>> 1+1;
>>
```

Perdita di precisione da alcuni calcoli:

```
>> a=1e10
a = 1.0000e+10
>> b=1e4
b = 10000
>> c=(a+b)^2
c = 1.0000e+20
>> format long % scrive piu' cifre
>> c
```

```
c = 1.00000200000100e+20
>> c - a^2 - 2*a*b - b^2
ans = 7936
```

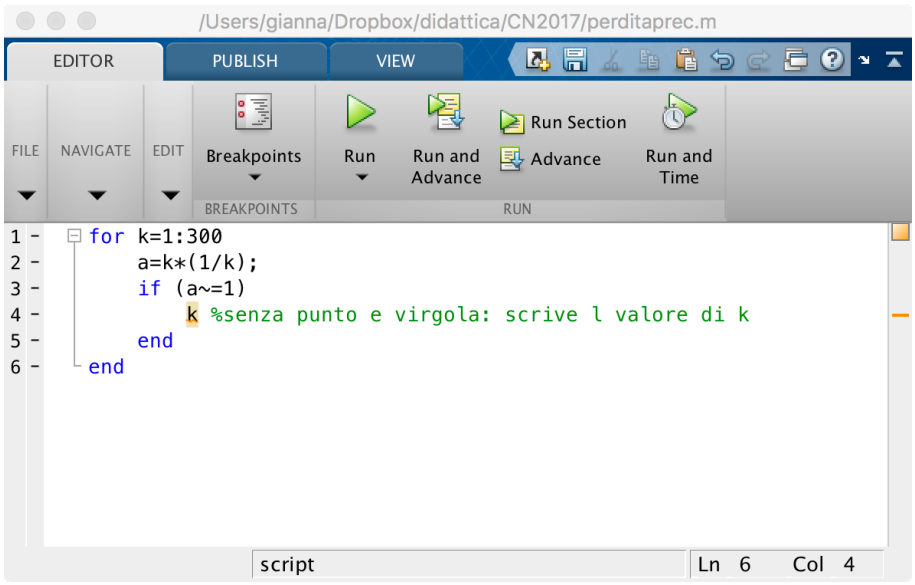
Cosa succede se sottraiamo da c prima il termine b^2?

Principalmente l'errore è dovuto alla sottrazioni di due numeri grossi e molto vicini, (*errori di cancellazione*), ma anche da moltiplicazioni:

```
>> a=98
a = 98
>> 1 - a*(1/a)
ans = 1.1102e-16
>> a=97
a = 97
>> 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. Per fare questo apriamo il nostro editor di testo dal menù e creiamo un file vuoto di nome `perditaprec.m` nella nostra home.

Attenzione: È importante che salviamo il file nella stessa cartella dove abbiamo lanciato il comando `Matlab`. Se non abbiamo eseguito il comando `cd` cartella nel terminale all'inizio di questa lezione (dove cartella è il nome di qualche cartella già esistente nella nostra home) allora questa sarà la nostra home, ovvero la cartella `/home/nomeutente`. Possiamo verificare in ogni momento in quale cartella ci troviamo (sia nel terminale che all'interno di matlab) con il comando `pwd`.



```
% scrivete queste istruzioni 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
  end
end
```

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

```
>> 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.

4 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
```

[1] Altrimenti, potete controllare e cambiare la "cartella di lavoro" di Matlab con il menu `Open` o navigando a partire dalla finestra "Current Folder" i soliti comandi `cd`, `pwd`, `ls`, che funzionano anche all'interno di Matlab.

```
  f=f*k;
end
% ora f vale n!
end
```

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

```
>> fact(10)
ans = 3628800
```

Esercizio 2. Scrivete una funzione `pow(x,n)` che attraverso un ciclo for e l'uso di un accumulatore, calcoli x^n, per $n \geq 1$.

5 Calcolo dell'esponenziale

La formula di Taylor permette di esprimere una funzione come un polinomio con infiniti termini, e troncando in modo opportuno questa serie è possibile approssimare una funzione con un polinomio.

In particolare, data una funzione $f \in C^n$, cioè derivabile n-volte con derivata n-esima continua, tale che $f : (a, b) \to \mathbf{R}$. Preso un punto $x_0 \in (a, b)$, abbiamo che

$$f(x) = f(x_0) + f'(x_0)(x - x_0)| + \frac{f''(x_0)}{2!}(x - x_0)^2 + \cdots + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n + R_n(x)$$

dove $R_n(x)$ è detto resto di Taylor e nella sua forma di Lagrange è dato da

$$R_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)^{n+1},$$

sotto l'ipotesi che $f^{(n+1)}(x)$ esista per ogni $x \in (a, b), x \neq x_0$, e con $|\xi - x_0| < |x - x_0|$.

Si può semplicemente vedere che il polinomio di Taylor per $x_0 = 0$, arrestato al termine n-esimo della funzione esponenziale è

$$1 + x + \frac{x^2}{2} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!}$$

da cui possiamo scrivere la seguente funzione per approssimare il valore dell'esponenziale in un punto x.

```
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);
  end
end
```

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

```
>> myexp(1,5)
ans = 2.7167
>> myexp(10,50)
ans = 2.2026e+04
>> exp(10)
ans = 2.2026e+04
>> format long
>> myexp(10,50)
ans = 22026.4657948067
>> exp(10)
ans = 22026.4657948067
```

Due problemi:

(1) Lento: con n termini, il numero di operazioni da eseguire cresce come n^2 (perché?)

(2) Inaccurato: prova `myexp(-20,500)`

Risolviamo il problema (1) 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;
  end
end
```

Ora va meglio per quanto riguarda il tempo. Si provi a calcolare il tempo di calcolo facendo `tic; myexp(1, 500); toc` e `tic; myexp2(1, 500); toc`.

Questa modifica risolve il problema (2)?

```
>> myexp(-20,500)
ans = NaN
>> myexp2(-20,500)
ans = 5.62188447213042e-09
```

Cosa succedeva?

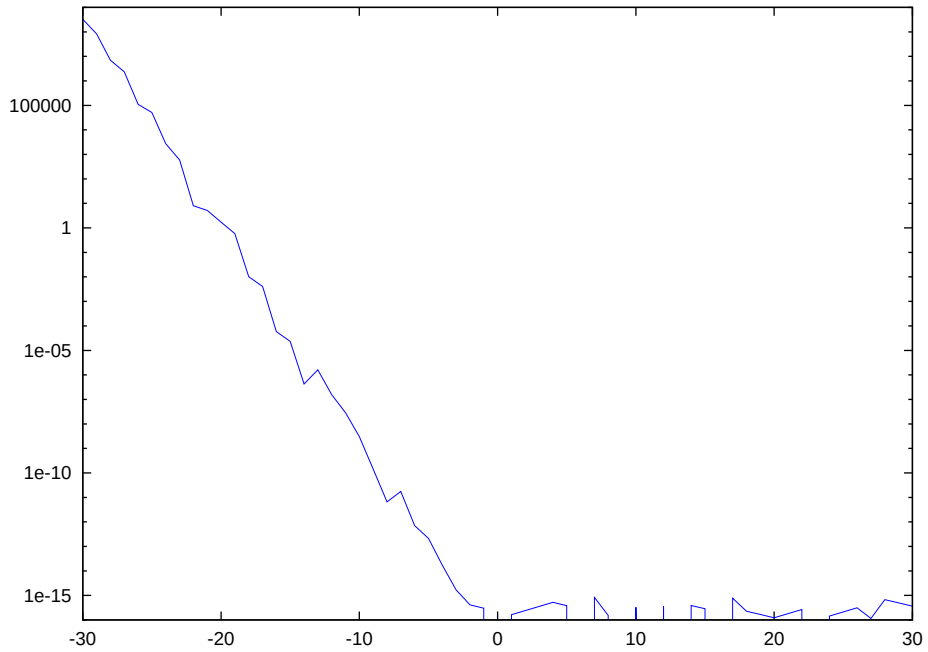
```
>> fact(500)
ans = Inf
>> pow(-20,500)
```

```
ans = Inf
>> Inf/Inf
ans = NaN
```

Ci sono ancora pesanti perdite di accuratezza sui numeri negativi:

```
>> 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: $\frac{|\text{myexp2}(x)-e^x|}{e^x}$

La soluzione: cambiare algoritmo e sceglierne uno che non porti a errori di cancellazione che sono dovuti al fatto che per $x < 0$ la serie esponenziale è a segni alterni.

```
>> exp(-30)
ans = 9.3576e-14
>> myexp2(-30,500)
ans = -3.0668e-05
>> 1/myexp2(30,500)
ans = 9.3576e-14
>> format long
>> exp(-30)
ans = 9.35762296884017e-14
```

```
>> 1/myexp2(30,500)
ans = 9.35762296884017e-14
```

Esercizio 3. 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.

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

Esercizio 5 (facoltativo). Supponimo di voler calcolare il limite della funzione $f(x) = x(\sqrt{x^2+1} - x)$ per $x \to \infty$. Dopo aver dimostrato che $\lim_{x\to\infty} f(x) = 1/2$,

- Si generi un vettore di 100 punti nell'intervallo $[5 \cdot 10^7, 10^8]$, con il comando `x=linspace(5*10^7, 10^8)`.
- Si definisca la funzione con il comando `f=@(x) x.*(sqrt(x.^2+1)-x)`, dove gli operatori preceduti dal . permettono di valutare in parallelo la funzione su tutte le componenti di un vettore, cioè $f(x_1, x_2, \cdots, x_n) = [f(x_1), f(x_2), \cdots, f(x_n)]$.
- Si valuti f sui 100 punti del vettore `x`, con il compando $y = f(x)$.
- Si plottino questi valori con il comando `plot(x, y)`. Il grafico dovrebbe in teoria mostrare la convergenza della funzione a $1/2$. Cosa succede invece?

Si ripeta lo stesso ragionamento riscrivendo $f(x)$ come $f_1(x) = x/(\sqrt{x^2+1} + x)$.