Laboratorio didattico di matematica computazionale

Beatrice Meini

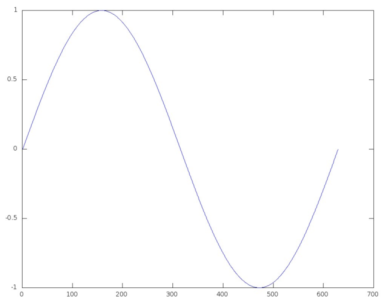
Lezione 2 - 12/3/2014

1 Richiami sui grafici

La funzione `plot` permette di disegnare grafici sul piano. L'uso più semplice é `plot(x)`, dove `x` è un vettore ; in questo caso il comando `plot` collega con dei segmenti i punti di coordinate `k, x(k)`, dove `k` va da 1 alla lunghezza di `x`. Ad esempio:

```
octave:6> x=sin([0:0.01:2*pi]);
octave:7> plot(x)
```

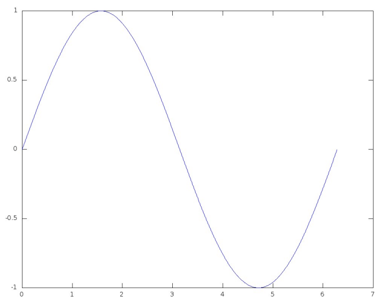
produce il grafico:



Invece i comandi:

```
octave:10> t=[0:0.01:2*pi]; x=sin(t);
octave:11> plot(t,x)
```

producono il grafico:



Talvolta è comodo tracciare grafici in cui la scala dell'asse delle ordinate è logaritmica invece che lineare: per questo possiamo usare il comando `semilogy`. Proviamo a tracciare il grafico della funzione $f(x) = 3^x$ nell'intervallo [0, 10] utilizzando il comando `plot` e `semilogy` (il comando `figure(2)` indirizza il grafico successivo al comando sulla finestra grafica numero 2)

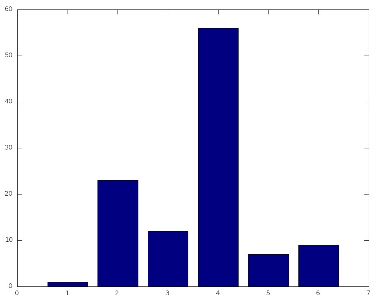
```
octave:13> x=[0:0.01:10];
octave:14> y=3.^x;
octave:15> plot(x,y)
octave:16> figure(2)
octave:17> semilogy(x,y)
```

Altri comandi analoghi sono `semilogx`, che genera un grafico con scala logaritmica sulle ascisse, e il comando `loglog` che usa la scala logaritmica sui due assi.

Il comando `bar(v)`, dove `v` è un vettore, produce un grafico a barre corrispondente ai valori degli elementi di `v`. Ad esempio:

```
octave:21> v=[1 23 12 56 7 9];
octave:22> bar(v)
```

produce il grafico:



2 La successione di Collatz

Sia n un numero intero positivo fissato. La successione di Collatz è la successione $\{a_k\}_{k\geq 1}$ di numeri interi definita nel seguente modo:

$$a_1 = n,$$
$$\text{per} \quad k \geq 1, \quad a_{k+1} = \begin{cases} 1 & \text{se} \;\; a_k = 1 \\ a_k/2 & \text{se} \;\; a_k \;\; \text{pari} \\ 3a_k + 1 & \text{altrimenti} \end{cases}$$

La congettura di Collatz afferma che, comunque si scelga n, esiste sempre un intero h tale che $a_h = 1$.

Esercizio 1. Si definisca in un file la function `a = collatz(n)` che prende in input n e dà in output il vettore a, che contiene i primi h elementi della successione $\{a_k\}_k$, dove h è minimo intero tale che $a_h = 1$.

Tips. Utilizzare i comandi `while`, `if`, `rem` e "allungare" il vettore `a` ad ogni iterazione, inzializzando il vettore il vettore `a` con il vettore vuoto `a=[ ]`.

Dovreste ottenere i seguenti risultati:

```
octave:1> a= collatz(3)
a =
   3   10    5   16    8    4    2    1
octave:2> a = collatz(1)
a = 1
octave:3> a = collatz(10)
a =
  10    5   16    8    4    2    1
octave:4> a = collatz(7)
a =
   7   22   11   34   17   52   26   13   40   20   10    5   16    8    4
2    1
```

Per "disegnare" la successione possiamo dare i comandi:

```
octave:1> close all, plot(a,'.-')
octave:4> title ( [ 'n = ' num2str( a ( 1 ) ) ] );
```

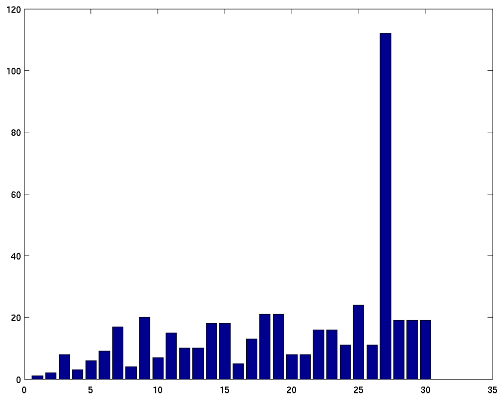
È interessante vedere, al variare del numero iniziale n, dopo quante iterazioni viene raggiunto il valore 1:

Esercizio 2. Si scriva la function `u = collatz_count (m)` che prende come input il numero intero positivo m, e restituisce in output il vettore u di dimensione m, tale che u_j è il numero di elementi del vettore `a= collatz(j)`, per $j = 1, 2, ..., m$.

Si fissi m e si dia il comando `bar(u)`. Ad esempio

```
octave:5> close all
octave:6> u=collatz_count(30);
octave:7> bar(u)
```

Dovreste ottenere la figura



Si provi con altri valori di m. Riuscite a trovare un valore inziale che richieda più iterazioni di quante ne servano per $n = 27$?

3 Successione di Fibonacci

La successione di Fibonacci è la successione $\{f_n\}_n$ definita nel seguente modo

$$f_1 = 1, \quad f_2 = 1$$
$$f_n = f_{n-1} + f_{n-2}, \quad n \geq 3$$

Esercizio 3. Si scriva in un file la function `f=fibonacci (m)` che, preso in input il numero intero positivo `m`, restituisce in output il vettore `f` che contiene i primi m elementi della successione $\{f_n\}_n$.

Si calcoli e si disegni il rapporto f_{n+1}/f_n per $n = 1, \ldots, m$, con m abbastanza grande (ma non troppo, per evitare overflow). Ad esempio:

```
octave:20> m=50;
octave:21> f=fibonacci(50);
octave:22> r=f(2:m)./f(1:m-1);
octave:23> plot(r)
octave:24> r(end)
ans = 1.6180
octave:25> format long e
octave:26> r(end)
ans =  1.61803398874989e+00
```

Si provi ora questo giochetto: si scelga un reale positivo `x`, ad esempio `x=10`, e si sostituisca alla variabile `x` il valore $(1+x)^{1/2}$ per "tante volte". Questo si fa eseguendo successivamente il comando `x=sqrt(1+x)`, che possiamo richiamare con la freccia in alto:

```
octave:38> x=10;
octave:39> x=sqrt(1+x)
x =  3.31662479035540e+00
octave:40> x=sqrt(1+x)
x =  2.07764886117828e+00
octave:41> x=sqrt(1+x)
x =  1.75432290675870e+00
octave:42> x=sqrt(1+x)
x =  1.65961528878192e+00
octave:43> x=sqrt(1+x)
x =  1.63083269797423e+00
octave:44> x=sqrt(1+x)
x =  1.62198418548833e+00
octave:45> x=sqrt(1+x)
x =  1.61925420656805e+00
octave:46> x=sqrt(1+x)
x =  1.61841101286665e+00
octave:47> x=sqrt(1+x)
x =  1.61815049141501e+00
octave:48> x=sqrt(1+x)
x =  1.61806998965280e+00
octave:49> x=sqrt(1+x)
x =  1.61804511360246e+00
octave:50> x=sqrt(1+x)
x =  1.61803742651475e+00
octave:51> x=sqrt(1+x)
```

```
x =   1.61803505107731e+00
octave:52> x=sqrt(1+x)
x =   1.61803431702709e+00
octave:53> x=sqrt(1+x)
x =   1.61803409019312e+00
octave:54> x=sqrt(1+x)
x =   1.61803402009758e+00
octave:55> x=sqrt(1+x)
x =   1.61803399843686e+00
octave:56> x=sqrt(1+x)
x =   1.61803399174333e+00
octave:57> x=sqrt(1+x)
x =   1.61803398967492e+00
```

Confrontate `x` con `r(end)`. Utilizzando il comando `help roots` guardate il funzionamento dell'istruzione `roots`.

Dare i comandi

```
octave:58> z=roots([1 -1 -1])
z =

   -6.18033988749895e-01
    1.61803398874989e+00

octave:59> q=max(z)
q =   1.61803398874989e+00
```

Che cosa rappresenta il vettore `[1 -1 -1]`? Come è legato alla successione di Fibonacci e alla funzione $x \to (1+x)^{1/2}$? Perché `q` è "vicino" a `x` e a `r(end)`?

Disegnare in scala semilogaritmica (utilizzando il comando `semilogy`) i primi m termini della successione di fibonacci e della successione $\{q^n\}$:

```
octave:70> semilogy(f,"r-;Fibonacci;")
octave:71> hold on
octave:72> semilogy(q.^[1:m],"g-;Exp;")
octave:73> hold off
```

Perché si ottengono due rette? Perché sono parallele?

Esercizio 4. Si consideri la successione $\{g_n\}_n$ definita nel seguente modo

$$\begin{aligned} &g_1 = 1, \quad g_2 = 1, \quad g_3 = 2 \\ &g_n = g_{n-1} + g_{n-3}, \quad n \geq 4 \end{aligned}$$

Lavorando come per la successione di Fibonacci, si determini ρ tale che $g_n \approx \sigma\rho^n$.

4 Successione di Fibonacci "randomizzata"

La successione di Fibonacci randomizzata è così definita:

$$\begin{aligned} &f_1 = 1, \quad f_2 = 1 \\ &f_n = f_{n-1} + p_n \cdot f_{n-2}, \quad n \geq 3 \end{aligned}$$

dove, per ogni intero n, p_n vale 1 con probabilità 1/2, vale -1 con probabilità 1/2. Si cerci su Wikipedia "Random Fibonacci sequence".

Esercizio 5. Si scriva la function `rf = rfibonacci (m)` che, preso in input il numero intero positivo `m`, dia in output il vettore `rf` che contiene i primi m elementi della successione $\{f_n\}$. Per calcolare p_n si possono utilizzare le funzioni `sign` e `rand`.

È stato dimostrato da D. Viswanath (Random Fibonacci sequences and the number c=1.13198824. Math. Comp., 69:1131–1155, 2000) che, con probabilità 1, la successione $\{f_n\}_n$ cresce come c^n dove $c = 1.13198824$....

Esercizio 6. Visualizzare graficamente questa proprietà disegnando in scala semilogaritmica il grafico della successione di Fibonacci randomizzata e quello di c^n, con $c = 1.13198824$

Dato un numero reale positivo β, definire la successione

$$x_n = x_{n-1} + p_n \cdot \beta \cdot x_{n-2}, \quad n \geq 3,$$

con $x_1 = 1$, $x_2 = 1$, dove p_n vale 1 con probabilità 1/2, vale -1 con probabilità 1/2. È stato dimostrato che la successione cresce esponenzialmente con probabilità 1 e il tasso di crescita esponenziale è minore di 1 se $0 < \beta < \beta^* = 0.70258\ldots$ (circa), maggiore di 1 se $\beta > \beta^*$ (si cerchi su Wikipedia "Embree–Trefethen constant").

Esercizio 7. *(Facolativo: solo se avete finito tutto il resto).* Disegnando in scala semilogaritmica un certo numero di elementi della successione $\{x_n\}_n$, si verifichi sperimentalmente questa proprietà. Si provi il caso particolare $\beta = \beta^*$.