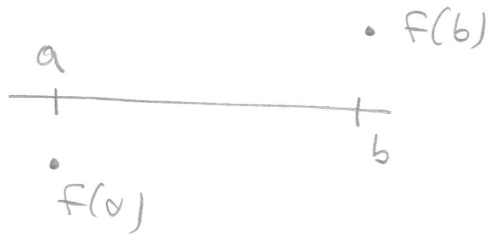
Misurando gli errori in senso assoluto...

- ϕ è un'approx <u>STABILE</u> di F (BISEZIONE si comporta $\approx$ bisezione)
- il <u>condizionamento</u> della funzione bisezione NON è buono (cambiando un pochino i dati cambia molto il risultato).

<u>TEO</u> $f : [a,b] \to \mathbb{R}$, $\delta > 0$



t.c.

- $f(a) < -\delta$, $f(b) > \delta$
- $f \in \mathcal{C}^1$ su $[a,b]$ con $f' > 0$

$\Rightarrow \exists!\, \alpha \in [a,b] : f(\alpha) = 0$

e $g : [a,b] \to \mathbb{R}$ continua

t.c. $\forall x \in [a,b]$, $|f(x) - g(x)| < \delta$

<u>Allora</u>:

- $\exists \beta \in [a,b]$ t.c. $g(\beta) = 0$
- $f(\beta) - \cancel{f(\alpha)} = f'(y)(\beta - \alpha)$

con y tra α e β (Teo di Lagrange)

$$\Rightarrow |\beta-\alpha| = \left|\frac{f(\beta)}{f'(y)}\right| < \frac{\delta}{m} \qquad m = \min_{[a,b]} |f'(x)|$$

$$\left[\, |f(\beta)| = |f(\beta)-g(\beta)| < \delta \,\right]$$

---

ovvero: distanza (zero f, zero g)

$< \frac{1}{m}$ distanza (f, g)

"numero di condizionam"

<u>Es</u>: Se ip non verificate...

- $f(x) = x^2$, $\alpha = 0$: $\forall \delta > 0$, $f+\delta$ NON ha zeri!
- $f(x) = (x-2)^{13}$, $\alpha = 2$, $f'(2) = 0$

  $\delta = 10^{-13}$, $(f-\delta)(x) = 0$ per $x = 2 + \frac{1}{10}$

ovvero: $\boxed{\text{dist (zero f, zero g)} = 10^{12} \text{ dist } (f, g)}$

Considerazioni conclusive sul METODO di BISEZIONE

☺ – è robusto ...

– funziona anche su f non molto regolari (f continua)

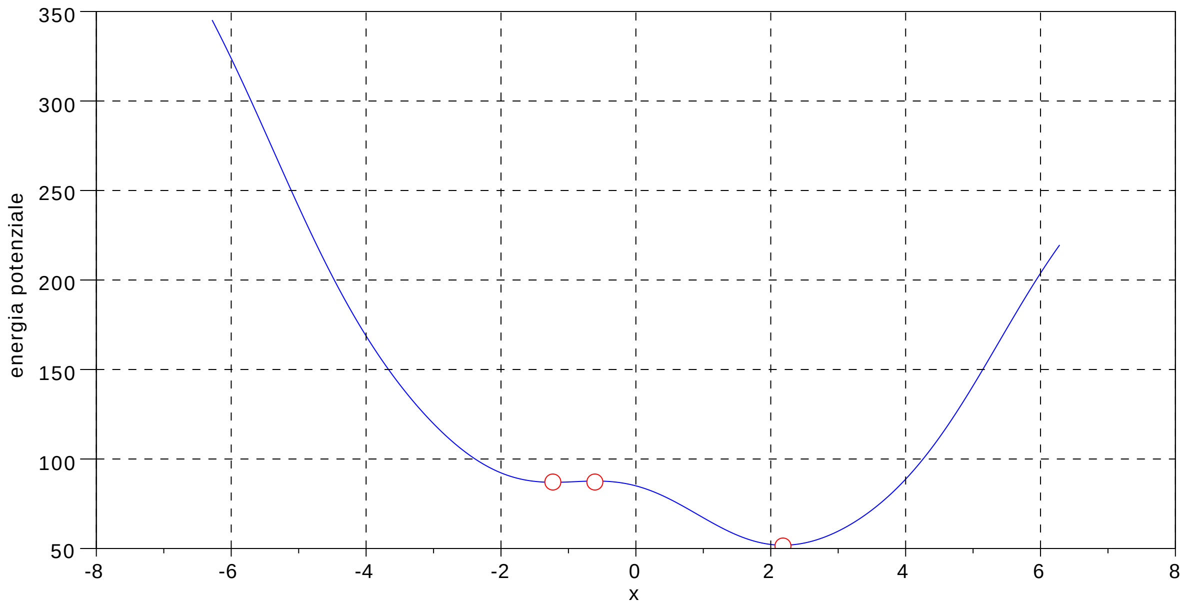
☹ – è necess trovare $[a,b]$ t.c. ...

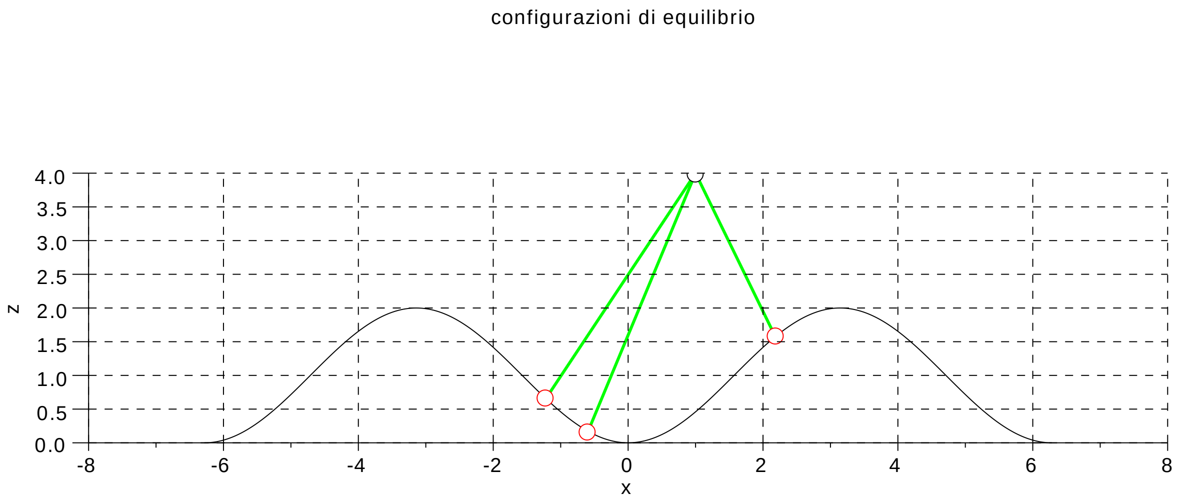
– come generalizz per $f: \mathbb{R}^n \to \mathbb{R}^n$?

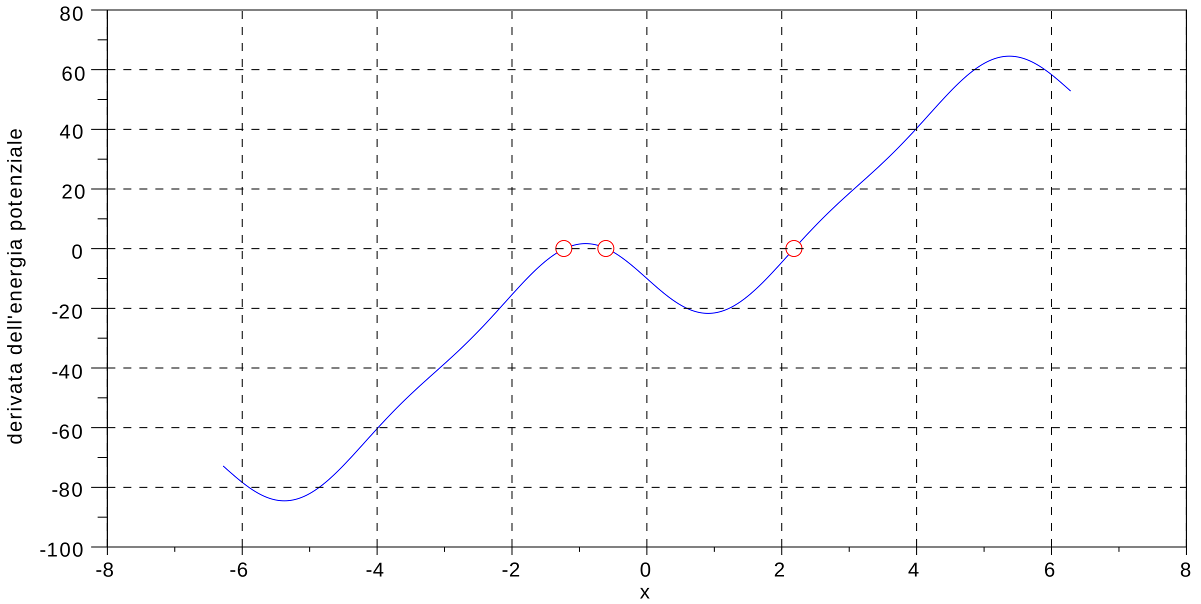
– "lento" ...

```
//
// Studio delle configurazioni di equilibrio di un punto materiale pesante P, mobile
// su una guida liscia contenuta in un piano verticale e soggetto all'azione di una
// molla. La molla, per la quale si suppone valida la Legge di Hooke con lunghezza a
// riposo zero, ha l'estremo non collegato a P fisso in un punto A del piano.
//
// La massa m del punto e la costante elastica K della molla sono assegnate, come anche
// le coordinate del punto A e l'equazione della guida rispetto ad un fissato sistema di
// riferimento nel piano con asse x orizzontale verso destra e z verticale verso l'alto.
//
// Incognita del problema: ascissa delle configurazioni di equilibrio del punto P.
//
// Essendo il vincolo liscio e le forze attive conservative, le configurazioni di
// equilibrio sono punti stazionari della funzione energia potenziale
//
//                    1
//   V(x) = mg f(x) + --- K ( (x - x(A))^2 + (f(x) - z(A))^2 )
//                    2
//
// La funzione f che descrive la guida deve essere derivabile.
//
// ***************************************************** Costanti:
m = 1; // kg
g = 9.8; // m/s^2
K = 10; // N/m
coord_A = [1,4]; // m, coordinate di A
// ***************************************************** Fine costanti
//
// Funzione che descrive la guida e sua derivata
//
function z=f_guida(x)
    z = 1 - cos(x)
endfunction
//
function j=Jf_guida(x)
    j = sin(x)
endfunction
//
// Energia potenziale del punto e sua derivata
//
function ep=V_punto(x)
    ep = m*g*f_guida(x) + K*((x-coord_A(1)).^2 + (f_guida(x)-coord_A(2)).^2)/2
endfunction
//
function y=JV_punto(x)
    y = m*g*Jf_guida(x) + K*(2*(x-coord_A(1)) - 2*(coord_A(2)-f_guida(x)).*Jf_guida(x))/2
endfunction
//
// Ricerca intervalli che includono una configurazione di equilibrio. La ricerca avviene
// utilizzando il Teorema di esistenza degli zeri sulla funzione V'.
//
PuntiEq = []; // Colonna dei punti di equilibrio trovati (in m)
Intervalli = []; // Matrice le cui colonne sono gli intervalli (in
                 // teoria, ciascun intervallo include un punto di eq)
//
// *** Qui si sceglie in quale intervallo globale ricercare i punti di equilibrio!
xx = linspace(-2*%pi,2*%pi,401); // I potenziali estremi di intervalli:
                         // si cercano, tra questi, zeri di V' oppure punti adiacenti nei
                         // quali V' assume valori di segno opposto.
// ***
SegnoJV = sign(JV_punto(xx)); // sign(x) = 0 se x = 0 ecc...
if SegnoJV(1) == 0 then PuntiEq(1) = xx(1); end;
for i = 2:length(xx),
    if SegnoJV(i) == 0 then PuntiEq($+1) = xx(i);
       elseif SegnoJV(i-1) ~= 0 then
           if SegnoJV(i) ~= SegnoJV(i-1) then
              Intervalli(1:2,$+1) = [xx(i-1);xx(i)]; end;
    end;
end;
// Adesso ho qualche configurazione di eq (quelle trovate tra le xx) e gli intervalli che
// ne racchiudono altre.
```

```
//
// **************************** Ricerca configurazioni di eq
//
//Percorso = "/home/maurizio/Documents/Scienza/Scilab/Zeri/Metodi/";
Percorso = "/home/ciampa/Lavoro/Scilab/Lavoro/Zeri/Metodi/";
exec(Percorso+'bisezione.sci');
tol = 2d-6; // Individua ciascun punto di eq con errore assoluto inferiore ad 1d-6 m
            // (un micron)
kmax = 55;
//
for Intervallo = Intervalli,
    a = Intervallo(1); b = Intervallo(2);
    [eq,v,info] = bisezione(JV_punto,a,b,"assoluto",tol,kmax,"zitto");
    PuntiEq($+1) = eq;
    printf("\nnuova ascissa di equilibrio = "),disp(eq);
    printf("v = "),disp(v);
    printf("info = "),disp(info);
end;
// **************************** Fine ricerca configurazioni di eq
//
printf("\nConfigurazioni di equilibrio trovate: %d", length(PuntiEq));
//
// Grafico energia potenziale e derivata, con configurazioni eq trovate
//

...omesso...

//
// Grafico configurazioni di equilibrio
//

...omesso...

//
```


energia potenziale
x


configurazioni di equilibrio
z
x


derivata dell'energia potenziale
x