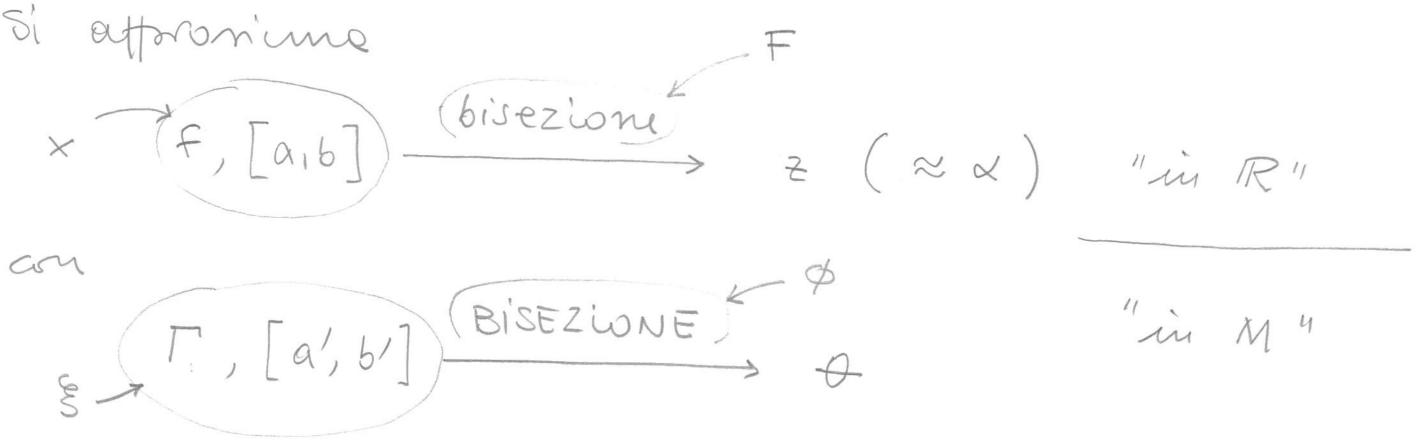
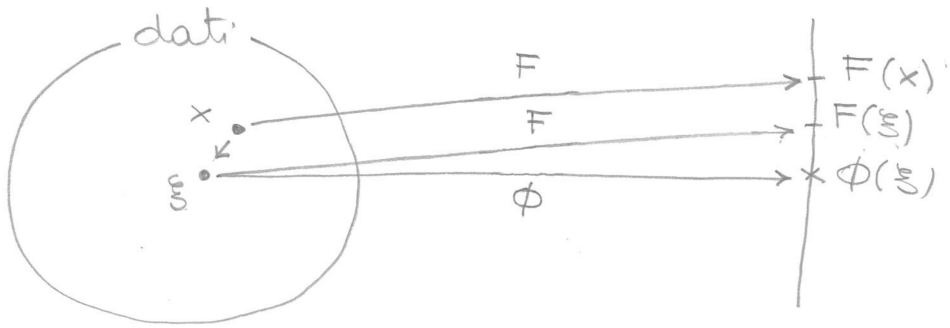
Es 2: $f:[a,b]\to\mathbb{R}$ continua, $f(a)\,f(b)<0$, un solo zero in $[a,b]$: α.

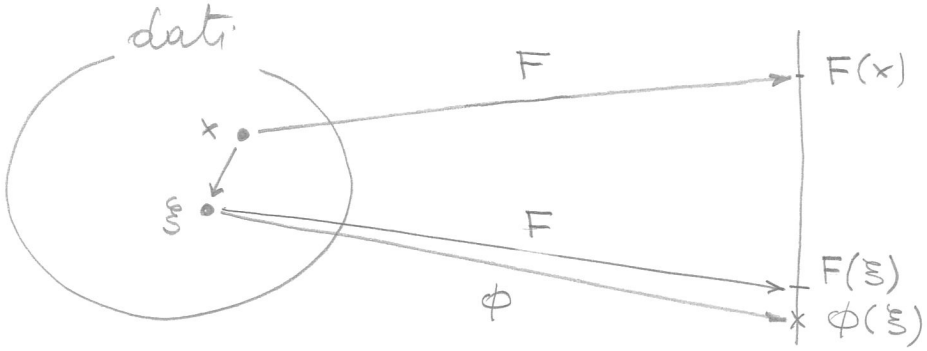
Si approssima





$\Gamma = \Phi$,

$\Gamma = \phi_2$



Misurando gli errori in senso assoluto...

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

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

t.c.

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

e $g:[a,b]\to\mathbb{R}$ continua t.c. $\forall x\in[a,b]$, $|f(x)-g(x)|<\delta$

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

Allora:

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

con y tra α e β (Teo di Lagrange)

$$\Rightarrow |\beta - \alpha| = \left| \frac{f(\beta)}{f'(\gamma)} \right| < \frac{\delta}{m}$$

$$m = \min_{[a,b]} |f'(x)|$$

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

---

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

"numero di condizionamento"

Es: 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: dist(zero f, zero g) $= 10^{12}$ dist(f, g)

Considerazioni conclusive sul METODO di BISEZIONE

- è robusto...
- funziona anche su f non molto regolari (f continua)

- è necessario trovare $[a,b]$ t.c. ...
- come generalizzo per $f: \mathbb{R}^n \to \mathbb{R}^n$?
- "lento"...

```
0001  //
0002  // Studio delle configurazioni di equilibrio di un punto materiale
0003  // pesante P, mobile su una guida liscia contenuta in un piano verticale
0004  // e soggetto all'azione di una molla. La molla, per la quale si suppone
0005  // valida la Legge di Hooke con lunghezza a riposo zero, ha l'estremo
0006  // non collegato a P fisso in un punto A del piano.
0007  //
0008  // La massa m del punto e la costante elastica K della molla sono
0009  // assegnate, come anche le coordinate del punto A e l'equazione della
0010  // guida rispetto ad un fissato sistema di riferimento nel piano con
0011  // asse x orizzontale verso destra e z verticale verso l'alto.
0012  //
0013  // Incognita del problema: ascissa delle configurazioni di equilibrio
0014  // del punto P.
0015  //
0016  // Essendo il vincolo liscio e le forze attive conservative, le
0017  // configurazioni di equilibrio sono punti stazionari della funzione
0018  // energia potenziale
0019  //
0020  //                      1
0021  //   V(x) = mg f(x) + --- K ( (x - x(A))^2 + (f(x) - z(A))^2 )
0022  //                      2
0023  //
0024  // La funzione f che descrive la guida deve essere derivabile.
0025  //
0026  // ************************************************* Costanti:
0027  m = 1; // kg
0028  g = 9.8; // m/s^2
0029  K = 10; // N/m
0030  coord_A = [0,4]; // m, coordinate di A
0031  // ********************************************** Fine costanti
0032  //
0033  protezione = funcprot();
0034  funcprot(0);
0035  //
0036  // Funzione che descrive la guida e sua derivata
0037  //
0001  function z=f_guida(x)
0002      //z = (x.^2)/2;  // (1)
0003      z = 1 - cos(x)   // (2)
0004  endfunction
0042  //
0001  function d1=d1f_guida(x)
0002      //d1 = x    // (1)
0003      d1 = sin(x) // (2)
0004  endfunction
0047  //
0048  // Energia potenziale del punto
0049  //
0001  function ep=V_punto(x)
0002      ep = m*g*f_guida(x) +...
0003           K*((x-coord_A(1)).^2 + (f_guida(x)-coord_A(2)).^2)/2
0004  endfunction
0054  //
0055  // Derivata della funzione energia potenziale
0056  //
0001  function y=d1V_punto(x)
0002      y = m*g*d1f_guida(x) +...
0003           K*((x-coord_A(1)) + (f_guida(x)-coord_A(2)).*d1f_guida(x))
0004  endfunction
0061  //
0062  // Ricerca intervalli che includono una configurazione di equilibrio.
0063  // La ricerca avviene utilizzando il Teorema di esistenza degli zeri
0064  // sulla funzione V'.
0065  //
0066  PuntiEq = []; // Colonna dei punti di equilibrio trovati (in m)
0067  Intervalli = []; // Matrice le cui colonne sono gli intervalli (in
0068                   // teoria, ciascun intervallo include un punto di eq)
0069  //
0070  // *** Qui si sceglie in quale intervallo globale ricercare i punti
0071  //     di equilibrio!
0072  xx = linspace(-2*%pi,2*%pi,401); // I potenziali estremi di intervalli:
0073                                   // si cercano tra questi zeri di V' oppure
0074                                   // punti adiacenti nei quali V' assume valori
0075                                   // di segno opposto.
0076  // ***
0077  Segnod1V = sign(d1V_punto(xx)); // sign(x) = 0 se x = 0 ecc...
0078  if Segnod1V(1) == 0 then PuntiEq(1) = xx(1); end;
```

```
for i = 2:length(xx),
    if Segnod1V(i) == 0 then PuntiEq($+1) = xx(i);
       elseif Segnod1V(i-1) ~= 0 then
           if Segnod1V(i) ~= Segnod1V(i-1) then
              Intervalli(1:2,$+1) = [xx(i-1);xx(i)]; end;
    end;
end;
// Adesso ho qualche configurazione di eq (quelle trovate tra le xx)
// e intervalli che ne racchiudono altre
//
if PuntiEq ~= [] then
    for PE = PuntiEq,
            printf("\nx_eq = %3.2e\n",PE);
    end
end
//
// *************************** Ricerca configurazioni di eq
//
exec('..\Metodi\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(d1V_punto,a,b,"assoluto",tol,kmax,"zitto");
    PuntiEq($+1) = eq;
    printf("\nx_eq = %3.2e",eq);
    printf("\nd1V(x_eq) = %3.2e",v);
    printf("\ninfo = %d\n",info);
end;
// *************************** Fine ricerca configurazioni di eq
//
printf("\nConfigurazioni di equilibrio trovate: %d", length(PuntiEq));
//
// Grafico energia potenziale e derivata, con configurazioni eq trovate
//
scf(0); clf(0);
subplot(221);
plot(xx,V_punto(xx),"b");
for eq = PuntiEq',
    plot(eq,V_punto(eq),"ro");
end;
xlabel("x");
ylabel("energia potenziale");
xgrid();
subplot(223);
plot(xx,d1V_punto(xx),"b");
for eq = PuntiEq',
    plot(eq,d1V_punto(eq),"ro");
end;
xlabel("x");
ylabel("derivata dell''energia potenziale");
xgrid();
//
// Grafico configurazioni di equilibrio
//
subplot(222);
plot2d(xx,f_guida(xx),frameflag=4);
for eq = PuntiEq',
    plot2d([eq,coord_A(1)],[f_guida(eq),coord_A(2)],3);
    DisegnoMolla = gce(); // il compound...
    LineaDisegnoMolla = DisegnoMolla.children; // ...la linea.
    LineaDisegnoMolla.thickness = 3;
    plot(eq,f_guida(eq),"ro");
end;
plot(coord_A(1),coord_A(2),"ko");
xlabel("x");
ylabel("z");
xtitle("configurazioni di equilibrio");
xgrid();
//
funcprot(protezione);
//
```

energia potenziale
x

derivata dell'energia potenziale
x

configurazioni di equilibrio
N
x