

Equazioni nonlineari.

Ángeles Martínez Calomardo e Alvisé Sommariva

Università degli Studi di Padova
Dipartimento di Matematica Pura e Applicata

7 novembre 2012

Equazioni non lineari: esempi.

- ▶ Risoluzione $f(x) = 0$ con $x \in [a, b] \subseteq \mathbb{R}$, $f \in C([a, b])$.
- ▶ Esempio 1: equazioni polinomiali $p_N(x) = 0$ con p_N polinomio di grado N . Possibili problemi (nessuna soluzione in \mathbb{R} , soluzioni multiple).
- ▶ Esempio 2: $f(x) = \sin(x) - x$. Soluzione unica poichè f decrescente (vedi derivata).

Equazioni non lineari: convergenza e ordine convergenza

- ▶ Metodo iterativo: genera successione $\{x_n\}_{n \in \mathbb{N}}$ che si desidera convergere a x^* tale che $f(x^*) = 0$.
- ▶ **Ordine di convergenza** del metodo iterativo: sia $\{x_k\}$ una successione convergente ad x^* e sia $e_k = x_k - x^*$ l'errore al passo k . Se esiste un numero $p > 0$ e una costante $C \neq 0$ tale che

$$\lim_{k \rightarrow \infty} \frac{|e_{k+1}|}{|e_k|^p} = C$$

allora p è chiamato *ordine di convergenza* della successione e C è la *costante asintotica di errore*. Per $p = 1$ la convergenza si dice **lineare**, per $p = 2$ si dice **quadratica**.

Metodo bisezione: definizione

Sia $f : [a, b] \rightarrow \mathbb{R}$ una funzione continua e supponiamo $f(a) \cdot f(b) < 0$. Il **metodo di bisezione** genera una successione di intervalli (a_k, b_k) con

- ▶ $f(a_k) \cdot f(b_k) < 0$,
- ▶ $[a_k, b_k] \subset [a_{k-1}, b_{k-1}]$,
- ▶ $|b_k - a_k| = \frac{1}{2}|b_{k-1} - a_{k-1}|$.

Fissate due tolleranze ϵ_1, ϵ_2 si arresta l'algoritmo quando

$$|b_k - a_k| \leq \epsilon_1 \text{ oppure } |f((a_k + b_k)/2)| \leq \epsilon_2.$$

Metodo bisezione: esempio

Operativamente dati $a \leq b$ con $f(a) \cdot f(b) < 0$ calcola $c = (a + b)/2$. Se $f(a) \cdot f(c) > 0$ sostituisce c ad a , viceversa sostituisce c a b , fermandosi se le condizioni d'arresto sono verificate.

Studiamo $f(x) = 0$ con $f(x) = \sin(x) - x$. Osserviamo che se $x < 0$ allora $f(x) > 0$ altrimenti $f(x) \leq 0$.

1. $\mathbf{a} = -0.30000000$, $\mathbf{b} = 0.40000000 \rightarrow \mathbf{c} = 0.05000000$.
2. $\mathbf{a} = -0.30000000$, $\mathbf{b} = 0.05000000 \rightarrow \mathbf{c} = -0.12500000$
3. $\mathbf{a} = -0.12500000$, $\mathbf{b} = 0.05000000 \rightarrow \mathbf{c} = -0.03750000$
4. $\mathbf{a} = -0.03750000$, $\mathbf{b} = 0.05000000 \rightarrow \mathbf{c} = 0.00625000$
5. ...

Metodo bisezione: alcuni fatti

- ▶ Non ha una velocità di convergenza nel senso della definizione sopra definita. Esempio: se applico bisezione per risolvere $\sin(x) - x = 0$, in $[a, b] = [-3, 2]$ la successione $|e_{n+1}/e_n|$ alterna valori 1.5 e $1/6$ e quindi non converge con ordine 1.
- ▶ Usa solo il segno della funzione.
- ▶ Se $f(a) \cdot f(b) < 0$ e $f \in C([a, b])$ allora converge (sempre!!) a un x^* tale che $f(x^*) = 0$.
- ▶ Il test del residuo $|f(x_k)| \leq \text{tol}$, con *tol* tolleranza prefissata, può essere non adatto a funzioni *piatte* o con *picchi* intorno al punto cui converge.
- ▶ Fissata una tolleranza ϵ , e due punti iniziali a, b tali che $f(a) \cdot f(b) < 0$ (con $f \in C([a, b])$) per avere un'errore assoluto $|x_n - x^*|$ sulla soluzione x^* inferiore ad ϵ necessitano al più

$$n = \text{ceil}\left(\frac{\log(b - a) - \log \epsilon}{\log 2}\right)$$

iterazioni del metodo.

Sia c un'approssimazione dello zero e si supponga di doverlo determinare a meno di una tolleranza tol .

Il criterio del residuo $|f(c)| < tol$ non è spesso accettabile:

- ▶ funzione **piatta**: si pensi a risolvere l'equazione $f(x) = 0$ con $f(x) = 10^{-50} \cdot x$; il residuo $|f(1)| = 10^{-50}$ ma 1 è molto distante da $x^* = 0$.
- ▶ funzione **ripida**: si pensi a risolvere l'equazione $f(x) = 10^{50} \cdot x = 0$; il residuo $|f(10^{-20})| = 10^{30}$ seppure 10^{-20} sia molto vicino a $x^* = 0$.

Bisezione in Matlab/Octave: criterio arresto

Siano $a < b$ e $c = (a + b)/2$. Diciamo **residuo pesato** $|f(c) \cdot w|$ con

$$w := \left(\frac{f(b) - f(a)}{b - a} \right)^{-1}.$$

Si vede subito che w^{-1} è un rapporto incrementale.

- ▶ funzione **piatta**: si pensi a risolvere l'equazione $f(x) = 0$ con $f(x) = 10^{-50} \cdot x$; w è grande ($w = 10^{50}$) e quindi $|f(1)| = 10^{-50}$ ma $|f(1)w| = 1$;
- ▶ funzione **ripida**: si pensi a risolvere l'equazione $f(x) = 0$ con $f(x) = 10^{50} \cdot x$; w è piccolo ($w = 10^{-50}$) e quindi $|f(10^{-20})| = 10^{30}$ ma $|f(10^{-20})w| = 10^{30} \cdot 10^{-50} = 10^{-20}$

Per f più generali, il test del residuo pesato $|f(c)w| < \text{tol}$ prova ad adattare situazioni in cui il test del residuo $|f(c)| < \text{tol}$ non sia affidabile.

Metodo bisezione: implementazione

```
function [c,k,semilunghezza,residuopesato]=bisezione(a,b,  
    tolintv,tolres,maxit,f)  
  
if b < a  
    s=b; b=a; a=s;  
end  
fa=fval(f,a); fb=fval(f,b);  
if fa == 0  
    c=a; k=0; semilunghezza=(b-a)/2; residuopesato=0;  
    return;  
end  
if fb == 0  
    c=b; k=0; semilunghezza=(b-a)/2; residuopesato=0;  
    return;  
end
```

Metodo bisezione: implementazione

```
for index=1:maxit
    c(index)=(a+b)/2; fc=feval(f,c(index));
    if sign(fc) == sign(fa)
        a=c(index); fa=fc;
    else
        b=c(index); fb=fc;
    end
    semilunghezza(index)=(b-a)/2; den=(fb-fa);
    if den == 0
        den=eps;
    end
    w=(b-a)/den; residuopesato(index)=abs(fc*w);
    if (residuopesato(index) < tolres) | ...
        (semilunghezza(index) < tolintv) | (fc == 0)
        k=index; fprintf('\n'); return;
    end
    fprintf('\n \t [IT]:%3.0f [c]: %5.5f [a]:%3.3f', ...
        index,c(index),a)
    fprintf('[AMP]: %2.2e [WRES]:%2.2e', ...
        semilunghezza(index),residuopesato(index));
end
k=maxit; fprintf('\n')
```

Metodo bisezione: descrizione dell'implementazione

- ▶ Supposto $I_0 = [a_0, b_0] \equiv [a, b]$ con $f(a) \cdot f(b) < 0$, calcoliamo c_1 punto medio di I_0 . Se $f(c_1) = 0$ si esce, determinando i parametri richiesti dal metodo. Altrimenti si determina l'intervallo $I_1 = [a_1, b_1]$ con $a_1 = c_1$ e $b_1 = b_0$ oppure $b_1 = c_1$ e $a_1 = a_0$ in modo che sia $f(a_1) \cdot f(b_1) < 0$.
- ▶ Registrata la semi-ampiezza di I_1 nella prima componente del vettore semilunghezza e il residuo pesato

$$\rho_1 := |f(c_1) \cdot (b_1 - a_1) / (f(b_1) - f(a_1))|$$

nella prima componente del vettore residuopesato, si testa se i criteri di arresto sono verificati, cioè se la semi-ampiezza dell'intervallo è inferiore a `tolintv` o il residuo pesato è minore di `tolres` o si è raggiunto un numero massimo di iterazioni `maxit`. Se uno solo di questi test viene soddisfatto si esce dalla routine altrimenti si continua.

Metodo bisezione: descrizione dell'implementazione

- ▶ Supposto $I_k = [a_k, b_k]$ con $f(a_k) \cdot f(b_k) < 0$, calcoliamo c_{k+1} punto medio di I_k . Se $f(c_{k+1}) = 0$ si esce, determinando i parametri richiesti dal metodo. Altrimenti si considera l'intervallo $I_{k+1} = [a_{k+1}, b_{k+1}]$ con $a_{k+1} = c_{k+1}$ e $b_{k+1} = b_k$ oppure $b_{k+1} = c_{k+1}$ e $a_{k+1} = a_k$ in modo che sia $f(a_{k+1}) \cdot f(b_{k+1}) < 0$.
- ▶ Registrata la semi-ampiezza di I_{k+1} nella $k + 1$ -sima componente del vettore `semiamp` e il residuo pesato

$$\rho_{k+1} := |f(c_{k+1}) \cdot (b_{k+1} - a_{k+1}) / (f(b_{k+1}) - f(a_{k+1}))|$$

nella $k + 1$ -sima componente del vettore `wres`, si testa se i criteri di arresto sono verificati, cioè se la semi-ampiezza dell'intervallo I_{k+1} è inferiore a `tolintv` o il residuo pesato è minore di `tolresiduo` o si è raggiunto un numero massimo di iterazioni `maxit`. Se uno solo di questi test viene soddisfatto si esce dalla funzione altrimenti si continua.

Salviamo in demobisezione.m il seguente file

```
% f=inline('(x-1).^3'); a=0; b=1.5;  
f=inline('x.^2-2'); a=1; b=2;  
tolres=10^(-15);tolintv=10^(-15);maxit=10000;  
[aa,bb]=bisezione(a,b,tolintv,tolres,maxit,f);  
format long e; [aa bb]
```

- ▶ **inline**: definisce funzione da valutare;
- ▶ **bisezione**: funzione bisezione (da fare);
- ▶ **a,b**: intervallo iniziale bisezione (input);
- ▶ **tolres,tolintv,maxit**: tolleranze bisezione (input);
- ▶ **f**: passaggio di funzione come variabile (input);
- ▶ **aa,bb**: vettori intervalli analizzati da bisez. (output);

Il metodo di Newton richiede che

- ▶ f sia derivabile con continuità su un intervallo $[a, b]$ di \mathbb{R} ;
- ▶ $f^{(1)}(x) \neq 0$ per ogni $x \in [a, b]$.

Se ben definito, il metodo di Newton genera una successione $\{x_k\}$ definita da

$$x_{k+1} := x_k - f(x_k)/f^{(1)}(x_k).$$

Metodo Newton: convergenza locale

Nel caso del metodo di Newton la convergenza non è in generale garantita. Esistono degli esempi in cui il metodo produce una successione non convergente. Ciò nonostante esistono molti teoremi che illustrano quando si è certi che il metodo converga.

Uno zero x^* si dice **semplice** se $f(x^*) = 0$ e $f^{(1)}(x^*) \neq 0$.

Teorema. Sia $x^* \in (a, b)$ uno zero semplice di $f : [a, b] \rightarrow \mathbb{R}$. Si supponga inoltre $f \in C^2([a, b])$. Allora per $x_0 \in [a, b]$ sufficientemente vicino a x^* le iterazioni del metodo di Newton

$$x_{k+1} = x_k - f(x_k)/f^{(1)}(x_k), \quad k = 0, 1, 2, \dots$$

sono ben definite e convergono quadraticamente a x^* .

1. Il metodo di Newton non è sempre convergente.
2. Se converge, non è detto che l'ordine di convergenza sia $p = 2$ (conv. quadratica).
3. Se uno zero x^* di f non è semplice allora la convergenza non è quadratica.

Metodo Newton: implementazione

```
function [x,ko]=newton(x0,tol,kmax,f,df)
x_old=x0; ko=0; x=[x0];
step=realmax; k=0;
while (abs(step) > tol) & (k < kmax)
    k=k+1; fx=feval(f,x_old);
    if fx == 0 return; end
    dfx=feval(df,x_old);
    if dfx == 0 ko=1; return; end
    step=-fx/dfx; x_new=x_old+step;
    x=[x; x_new]; x_old=x_new;
end
ko=1;
```

Metodo Newton: implementazione, esempio

Quale esempio con il file `demonewton.m` usiamo il metodo di Newton per il calcolo di $\sqrt{2}$, cioè l'unica radice positiva di $f(x) = x^2 - 2$.

```
% f=inline('(x-1).^3'); x0=1.5;  
f=inline('x.^2-2'); df=inline('2*x'); x0=1;  
tolstep=10^(-15); maxit=1000;  
[x,ko]=newton(x0,tolstep,maxit,f,df);  
format long e; x
```

Metodo Newton: implementazione, esempio

- ▶ Il comando **inline** permette di definire le funzioni $f(x) = x^2 - 2$ e $f^{(1)}(x) = 2x$, in forma vettoriale (si noti il `.`).
- ▶ Il valore da cui parte il metodo di Newton è **x0** (vicino a x^*).
- ▶ **tolstep** definisce la tolleranza del criterio di arresto dello step

$$|x_{k+1} - x_k| \leq \text{tolstep}$$

uscendo al massimo dopo **maxit** iterazioni.

- ▶ **newton** è la routine che esegue il metodo di Newton. In output vettore iterazioni `x`, flag `ko` (0: ok, 1: ko).

Esercizi per casa. Calcolare con i metodi di bisezione e Newton un'approssimazione dello zero α di

$$5x = \exp(-x) \quad (1)$$

$$x^3 - 2 = 0 \quad (2)$$

$$x \log(x) - 1 = 0 \quad (3)$$

Verificare che per le approssimazioni $x^* \approx \alpha$ fornite sia effettivamente $f(x^*) \approx 0$. Quante iterazioni necessitano perché i metodi forniscano tale x^* , supponendo di utilizzare una tolleranza di 10^{-15} ?