▷ Lezione II

Equazioni non-lineari

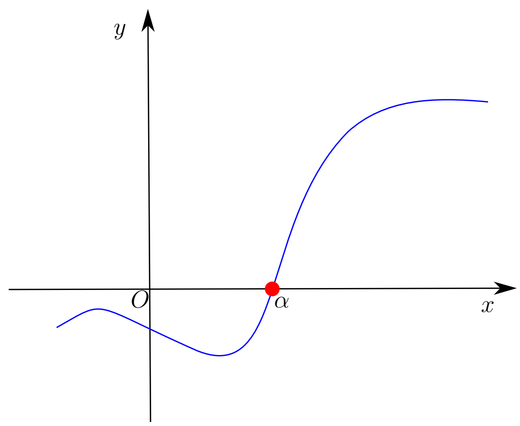
Libro Capitolo 2, sezione 2.1-2.2-2.3

Metodi numerici per equazioni non-lineari

Consideriamo una funzione scalare continua f, il problema che vogliamo risolvere è il seguente

$$f(x) = 0, \tag{1.1}$$

ovvero siamo interessati a trovare gli zeri (o radici) α di f, come rappresentato nel seguente grafico.



In generale non è possibile trovare un'espressione analitica di α, tranne che in casi semplici come ad esempio le equazioni lineari, quadratiche, alcuni casi di equazioni trigonometriche ecc. Dobbiamo quindi utilizzare uno schema numerico che fornisca una sua approssimazione. Un metodo numerico per trovare un'approssimazione di α parte da un valore o *guess* iniziale x^0 e costruisce una sequenza $\{x^i\}_{i=0}^k$ tale che per $k \to \infty$ otteniamo che $x^i \to \alpha$:

$$\text{dato } x^0 \xrightarrow{\text{calcolo}} x^1 \xrightarrow{\text{calcolo}} x^2 \xrightarrow{\text{calcolo}} \dots \xrightarrow{\text{calcolo}} x^k \xrightarrow[k\to\infty]{\text{calcolo}} \alpha.$$

Tale approccio è detto iterativo in quanto genera una sequenza di soluzioni "tentativo" che possono convergere alla soluzione esatta. Se infatti abbiamo che

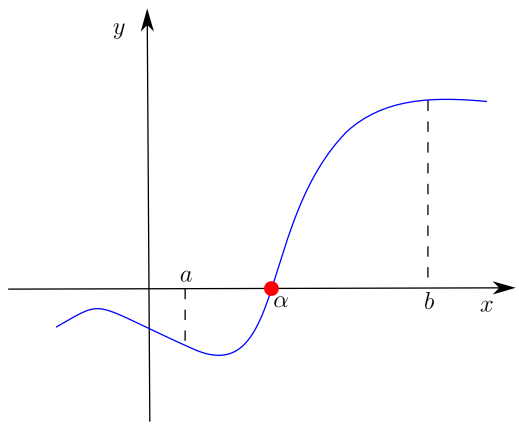
$$\lim_{k\to\infty} x^k = \alpha,$$

allora il metodo si dice convergente. Vediamo di seguito alcuni esempi di metodi numerici per la ricerca degli zeri di funzioni nonlineari.

Algoritmo di bisezione

Teorema 2.1 - Teorema degli zeri

Data una funzione $f : [a, b] \to \mathbb{R}$ continua sull'intervallo $[a, b]$ se $f(a) \cdot f(b) < 0$, allora esiste uno zero o radice di f in $[a, b]$.

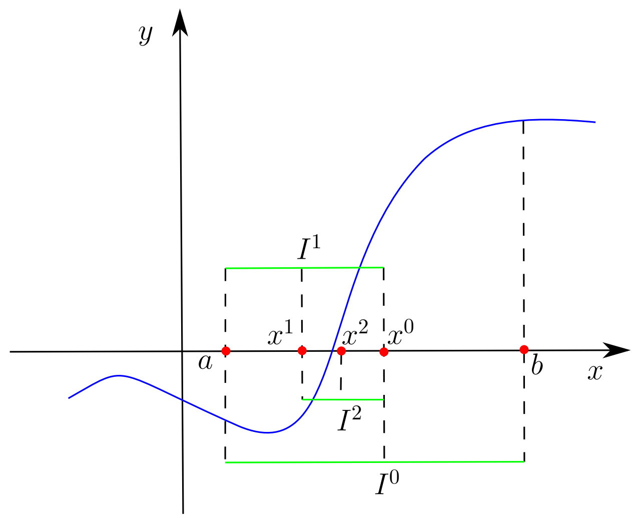


Sotto le ipotesi del Teorema 1.1.1, possiamo introdurre, nell'Algoritmo 1.1.1 lo pseudo-codice dell'algoritmo di bisezione. Lo pseudo-codice è una possibile implementazione di un algoritmo in un meta-linguaggio che risulta simile a molti linguaggi di programmazione, per cui la traduzione successiva in uno specifico linguaggio (es Python, C++) risulta molto semplice. Osserviamo che al posto di usare l'uguale come simbolo matematico, usiamo la freccia che punta a sinistra $\leftarrow$ che indica l'assegnazione di un valore ad una variabile.

Algoritmo 2.1 - Algoritmo di bisezione

Data: Per $k = 0$ impostiamo $a^0 = a$, $b^0 = b$ e conseguentemente $I^0 = [a^0, b^0]$;
for $k = 0, 1, 2, \dots$ **do**
  $x^k \leftarrow \dfrac{a^k + b^k}{2}$;
  if $f(x^k) = 0$ **then**
    $\alpha \leftarrow x^k$;
    `Stop`;
  end
  if $f(x^k) \cdot f(a^k) < 0$ **then**
    $a^{k+1} \leftarrow a^k$;
    $b^{k+1} \leftarrow x^k$;
  end
  if $f(x^k) \cdot f(b^k) < 0$ **then**
    $a^{k+1} \leftarrow x^k$;
    $b^{k+1} \leftarrow b^k$;
  end
  $I^{k+1} \leftarrow [a^{k+1}, b^{k+1}]$;
  $k \leftarrow k + 1$;
end

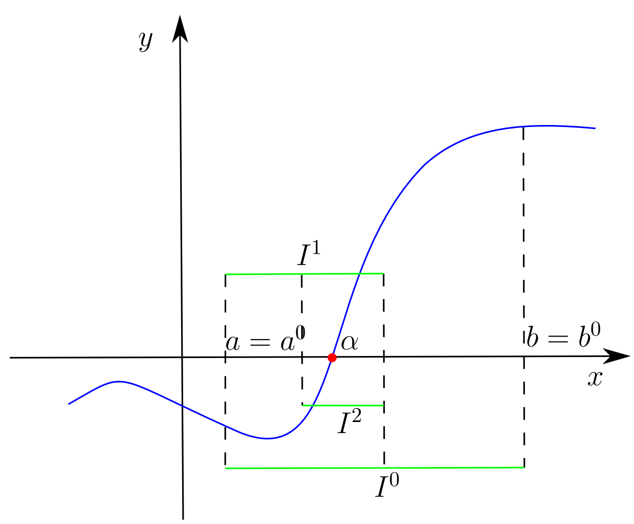
L'idea è che, dati a e b tali per cui la funzione ha segno opposto, $f(a)f(b) < 0$, allora fra a e b è necessariamente compreso uno zero. Per ridurre l'incertezza si divide l'intervallo a metà e si valuta il segno della funzione nel punto medio, quindi si sceglie la metà dell'intervallo in cui la funzione cambia segno e si ripete il procedimento. La rappresentazione grafica è data dalla figura seguente



Quando l'Algoritmo 1.1.1 è implementato in un calcolatore siamo in presenza di aritmetica finita: invece di risolvere esattamente il problema (1.1) $f(x) = 0$ arriviamo iterativamente ad avere

$$|f(x)| < toll,$$

dove *toll* è una tolleranza pre-impostata e solitamente è un numero molto piccolo. Ricordiamo che non è mai una buona strategia fare test di uguaglianza tra numeri macchina, è più sicuro richiedere che due numeri siano uguali a meno di una tolleranza. Analizziamo quindi le iterazioni che vengono generate dall'Algoritmo 1.1.1 nel seguente grafico



Ad ogni iterazione abbiamo che l'intervallo e la sua dimensione in cui ricerchiamo la radice al passo k sono dati da

$$I^k = (a^k, b^k) \quad \Rightarrow \quad \left|I^k\right| = \left|b^k - a^k\right| = \frac{1}{2}\left|I^{k-1}\right| = \frac{1}{2}\left|b^{k-1} - a^{k-1}\right| = \ldots = \frac{1}{2^k}\left|b - a\right|.$$

Il secondo passaggio è dovuto al fatto che dall'iterazione $k-1$ all'iterazione k l'intervallo si dimezza.

Possiamo quindi calcolare l'errore e^k che viene commesso dallo schema di bisezione al passo k

$$\left|e^k\right| = \left|x^k - \alpha\right| \leq \frac{1}{2}\left|b^k - a^k\right| = \frac{1}{2}\frac{1}{2^k}\left|b-a\right| = \frac{1}{2^{k+1}}\left|b-a\right|.$$

Siamo quindi in grado di stimare l'errore commesso senza dover sapere il valore della soluzione esatta α, tale risulta

$$\left|e^k\right| \leq \frac{1}{2^{k+1}}\left|b-a\right|. \tag{1.2}$$

Dato che $\left|e^k\right| \to 0$ per $k \to \infty$, allora il metodo è convergente ed avremo che

$$\lim_{k\to\infty} x^k = \alpha.$$

Si noti che (1.2) è una stima dall'alto dell'errore al passo k, è possibile che l'errore effettivo sia minore della stima ma sicuramente non sarà maggiore.

Definzione 2.1 - Metodo iterativo convergente

Dato un metodo iterativo che a partire da un dato iniziale x^0 costruisce una successione $\{x^k\}$, tale si dice convergente se

$$\lim_{k\to\infty} x^k = \alpha,$$

dove α è la soluzione esatta del problema.

Per il metodo di bisezione, se vogliamo trovare un'approssimazione di α, a meno di una tolleranza ϵ prefissata, possiamo sapere quante iterazioni saranno necessarie. Infatti, considerando la disequazione (1.2) abbiamo

$$\left|e^k\right| \leq \frac{1}{2^{k+1}}\left|b-a\right| < \epsilon \quad \Rightarrow \quad \log_2 \frac{1}{2^{k+1}} < \log_2 \frac{\epsilon}{\left|b-a\right|} \quad \Rightarrow \quad -(k+1) < \log_2 \frac{\epsilon}{\left|b-a\right|} \quad \Rightarrow$$
$$k+1 > -\log_2 \frac{\epsilon}{\left|b-a\right|} \quad \Rightarrow \quad k+1 > \log_2 \frac{\left|b-a\right|}{\epsilon} \quad \Rightarrow \quad k > \log_2 \frac{\left|b-a\right|}{\epsilon} - 1,$$

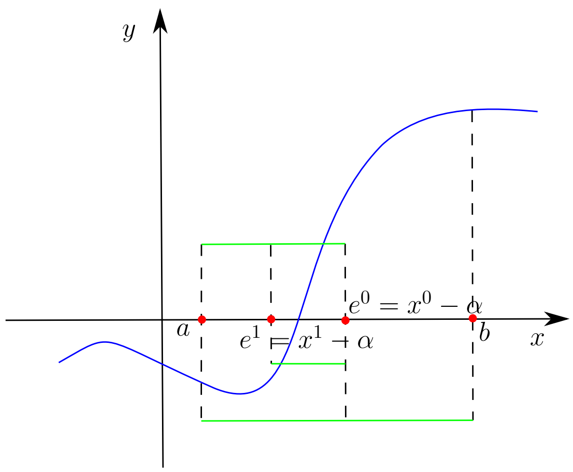
ovvero il numero di iterazioni necessarie per il raggiungimento della soluzione a meno della tolleranza ϵ è data da

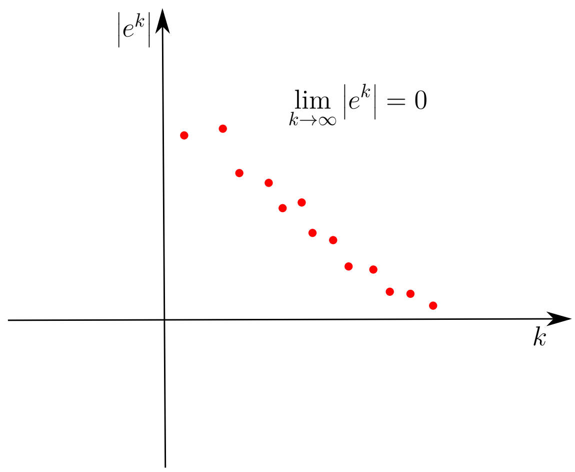
$$k > \log_2 \frac{\left|b-a\right|}{\epsilon} - 1.$$

Nel caso in cui si ottenga un numero non intero nell'espressione a destra allora k va arrotondato al numero intero superiore. Per riassumere, il metodo di bisezione converge a uno degli zeri della funzione f compresi nell'intervallo $[a, b]$. La convergenza non è tuttavia monotona, ovvero non possiamo affermare se

$$\left|e^{k+1}\right| \leq \left|e^k\right| \quad \text{oppure} \quad \left|e^{k+1}\right| \geq \left|e^k\right| \quad \text{oppure} \quad \left|e^{k+1}\right| = \left|e^k\right|.$$

Infine la convergenza del metodo di bisezione è generalmente lenta, tale concetto verrà sviluppato meglio nel seguito. La non-monotonicità dell'errore risulta evidente se rappresentiamo l'andamento dell'errore nell'esempio considerato:





Teorema 2.2

Se una funzione ha molteplicità algebrica dispari allora esiste un intorno di α tale che le ipotesi del teorema degli zeri sono verificate. Ad esempio uno zero ha

$$\begin{array}{ll} \text{molteplicità 1 se} & f(\alpha)=0 \quad \text{e} \quad f'(\alpha)\neq 0 \\ \text{molteplicità 3 se} & f(\alpha)=0 \quad \text{e} \quad f'(\alpha)=0 \quad \text{e} \quad f''(\alpha)=0 \quad \text{e} \quad f'''(\alpha)\neq 0 \end{array}$$

Non è invece assicurata l'esistenza nel caso di molteplicità algebrica pari, ovvero

$$\text{molteplicità 2 se} \quad f(\alpha)=0 \quad \text{e} \quad f'(\alpha)=0 \quad \text{e} \quad f''(\alpha)\neq 0$$

La molteplicità algebrica può essere anche studiata sfruttando l'espansione in serie di Taylor di f nell'intorno di α. Ottengo infatti un'espressione del tipo

$$f(x) = f(\alpha) + f'(\alpha)(x-\alpha) + \frac{f''(\alpha)}{2}(x-\alpha)^2 + \frac{f'''(\alpha)}{6}(x-\alpha)^3 + \dots$$

Quindi nell'intorno dello zero α abbiamo la seguente approssimazione locale di f

$$\begin{array}{lll} \text{molteplicità 1:} & f(\alpha)=0 \quad \text{e} \quad f'(\alpha)\neq 0 & \Rightarrow f(x)\sim x-\alpha \\ \text{molteplicità 2:} & f(\alpha)=0 \quad \text{e} \quad f'(\alpha)=0 \quad \text{e} \quad f''(\alpha)\neq 0 & \Rightarrow f(x)\sim (x-\alpha)^2 \\ \text{molteplicità 3:} & f(\alpha)=0 \quad \text{e} \quad f'(\alpha)=0 \quad \text{e} \quad f''(\alpha)=0 \quad \text{e} \quad f'''(\alpha)\neq 0 & \Rightarrow f(x)\sim (x-\alpha)^3 \end{array}$$

Esempio 2.1

Consideriamo la seguente funzione

$$f(x) = \left(e^{x^2-2} - 1\right)\left(x + \sqrt{2}\right)^2$$

dove $\alpha = -\sqrt{2}$ è radice del problema $f(x) = 0$. Calcoliamone la molteplicità

$$f(\alpha) = 0$$

$$f'(x) = \left(2\,x + 2\sqrt{2}\right)\left(\mathrm{e}^{x^2-2} - 1\right) + 2\,x\,\mathrm{e}^{x^2-2}\left(x+\sqrt{2}\right)^2 \quad \Rightarrow \quad f'(\alpha) = 0$$

La radice è di molteplicità almeno 2, calcoliamo la derivata seconda

$$f''(x) = 2\,e^{x^2-2} + 2\,e^{x^2-2}\left(x+\sqrt{2}\right)^2 + 4\,x\,e^{x^2-2}\left(2\,x+2\sqrt{2}\right) + 4\,x^2\,e^{x^2-2}\left(x+\sqrt{2}\right)^2 - 2$$

che calcolata in α fornisce ancora una volta 0: $f''(\alpha) = 0$. Quindi α è di molteplicità almeno 3. Calcoliamo infine la derivata terza di f che risulta

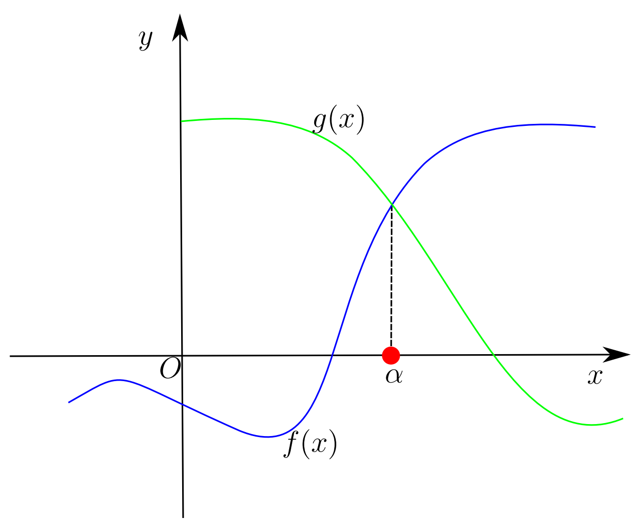
$$f'''(x) = 4\,e^{x^2-2}\left(2\,x^5 + 4\sqrt{2}\,x^4 + 13\,x^3 + 12\sqrt{2}\,x^2 + 12\,x + 3\sqrt{2}\right)$$

che valutata in α risulta $f'''(\alpha) = -12\sqrt{2} \neq 0$, quindi la molteplicità di α risulta pari a 3.

Nel caso in cui volessi risolvere il problema di determinare α in modo che due funzioni abbiano lo stesso valore, ossia se volessi risolvere

$$f(\alpha) = g(\alpha),$$

come rappresentato in figura



Tale problema può essere riscritto come un problema di ricerca degli zeri di una funzione ausiliaria h, data da

$$h(x) = f(x) - g(x) \quad \Rightarrow \quad h(\alpha) = f(\alpha) - g(\alpha) = 0,$$

come visto in precedenza.

Metodi di punto fisso

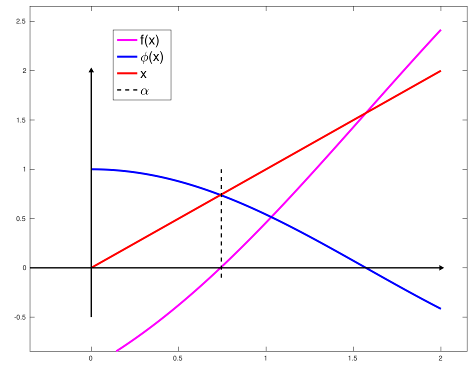
In alcuni casi è conveniente riscrivere il problema di ricerca degli zeri in questa forma:

$$x = \phi(x), \tag{1.3}$$

ad esempio, nel caso in cui $f(x) = x - \cos(x)$ è evidente che possiamo individuare $\phi(x) = \cos(x)$. Come mostrato in figura il punto cercato, lo zero di $f(x)$, coincide con l'intersezione fra i grafici $y = \phi(x)$ e $y = x$.

La funzione ϕ è detta *funzione di iterazione* e possiamo costruire in modo naturale una sequenza di soluzioni in questo modo: data una *guess* iniziale x^0,

$$x^{k+1} = \phi(x^k).$$



Questo metodo è per costruzione *fortemente consistente* nel senso che la soluzione esatta x soddisfa sempre l'uguaglianza $x = \phi(x)$. Notiamo anche che la scelta della funzione di iterazione, data $f(x)$, non è unica. Il metodo di punto fisso può essere molto semplice da implementare, si veda l'algoritmo seguente.

Algoritmo 2.2 - Algoritmo di punto fisso

Data: Per $k = 0$ impostiamo $x^0 = x_0$;
for $k = 0, 1, 2, \ldots$ **do**
  $x^{k+1} \leftarrow \phi(x^k)$;
  $k \leftarrow k + 1$;
end

Tuttavia, il metodo non è sempre convergente, in particolare abbiamo il seguente risultato.

Teorema 2.3

Si consideri la sequenza $x^{k+1} = \phi(x^k)$, con $k \geq 0$, e x^0 dato. Supponiamo che:

1. $\phi : [a, b] \to [a, b]$
2. $\phi \in C^1([a, b])$
3. $|\phi'(x)| < 1 \; \forall x \in [a, b]$.

Allora ϕ ha un unico punto fisso $\alpha \in [a, b]$ e la sequenza $\{x^k\}$ converge ad α per qualsiasi scelta di $x^0 \in [a, b]$. Inoltre, abbiamo che

$$\lim_{k \to +\infty} \frac{x^{k+1} - \alpha}{x^k - \alpha} = \phi'(\alpha).$$

Approfondimento 2.1

Illustriamo brevemente come si può dimostrare il teorema enunciato. Innanzitutto, l'esistenza di un punto fisso è garantita dalla condizione (1), ossia se $\phi(x)$ è continua in $[a, b]$ con valori in $[a, b]$

allora il suo grafico interseca necessariamente la bisettrice. Dimostriamo per assurdo l'unicità dello zero α. Supponiamo esistano α_1, α_2:

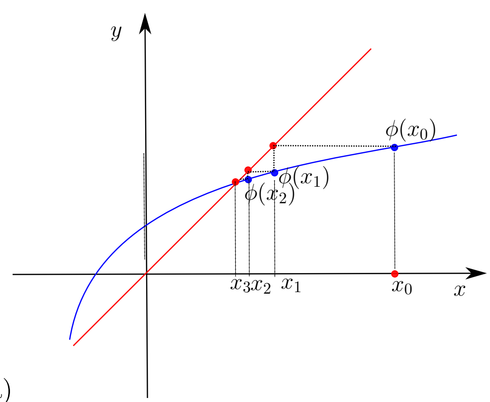
$$
\begin{gathered}
\alpha_1 = \phi(\alpha_1) \\
\alpha_2 = \phi(\alpha_2) \\
|\alpha_1 - \alpha_2| = |\phi(\alpha_1) - \phi(\alpha_2)| = |\phi'(\eta)(\alpha_1 - \alpha_2)| \leq |\phi'(\eta)||(\alpha_1 - \alpha_2)|
\end{gathered}
$$

dove, nel penultimo passaggio, abbiamo usato il Teorema di Lagrange e η è un punto compreso fra α_1, α_2. Infine, sappiamo che $|\phi'(x)| < 1$ otteniamo $|\alpha_1 - \alpha_2| \leq |\alpha_1 - \alpha_2|$ che implica l'uguaglianza delle due radici. Infine, abbiamo che

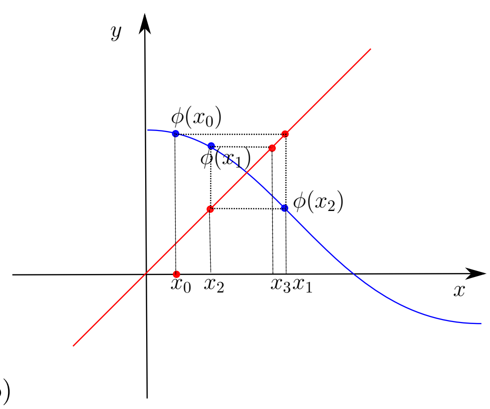
$$
\begin{gathered}
x^{k+1} - \alpha = \phi(x^k) - \alpha \\
x^{k+1} - \alpha = \phi(x^k) - \phi(\alpha) \\
x^{k+1} - \alpha = \phi'(\eta)(x^k - \alpha).
\end{gathered}
$$

Per $k \to \infty$ si ottiene il risultato di convergenza enunciato.

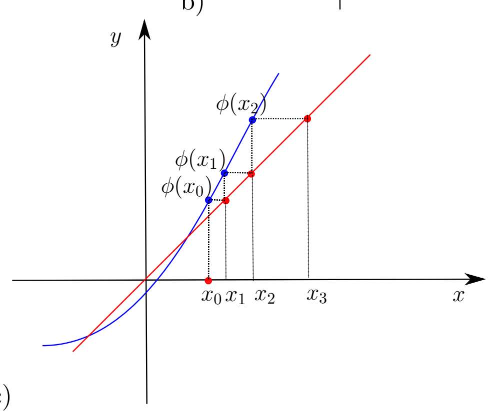
Sappiamo dunque che il metodo converge se, in un intervallo che contiene la soluzione α la funzione di iterazione ha derivata minore di 1 (in modulo); inoltre, il metodo converge più velocemente se la derivata ϕ' è prossima allo zero. Nelle figure seguenti osserviamo tre possibili casi: a) $0 < \phi'(x) < 1$ nell'intervallo considerato: la sequenza converge e il segno di $x_k - \alpha$ è sempre lo stesso (positivo in questo caso); b) $-1 < \phi'(x) < 0$, la sequenza converge e il segno di $x_k - \alpha$ cambia nel corso delle iterazioni; c) $\phi'(x) > 1$ e la sequenza diverge.



a)



b)



c)

Il teorema enunciato garantisce la convergenza *globale* nell'intervallo $[a, b]$ che, tuttavia, non è sempre determinabile a priori. In alternativa, può essere utile considerare il Teorema di Ostrowski, che fornisce un risultato locale:

Teorema 2.4

Sia α un punto fisso della funzione ϕ, che è continua e derivabile in un intorno di α. Se $|\phi'(\alpha)| < 1$ allora esiste un $\delta > 0$ tale che la sequenza $\{x^k\}$ converge ad α per ogni x^0 tale che $|x^0 - \alpha| < \delta$.

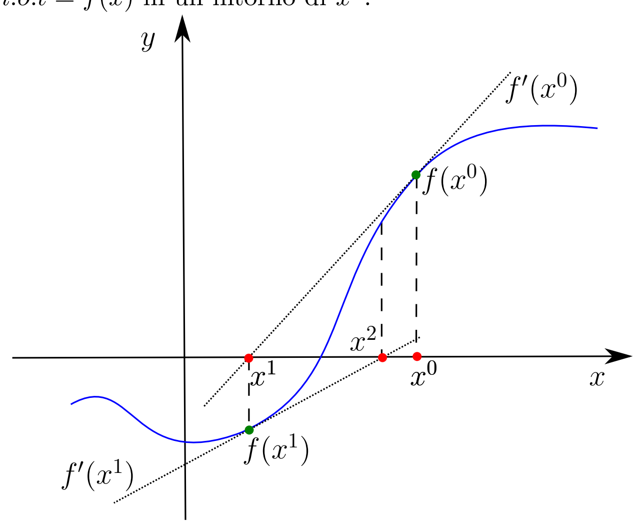
Quando arrestare un metodo iterativo di punto fisso? Con l'algoritmo di bisezione l'ampiezza dell'intervallo considerato è un indicatore dell'errore massimo che stiamo commettendo. Per i metodi di punto fisso e il metodo di Newton, presentato nella prossima sezione, sono necessari altri *criteri di arresto*, discussi in dettaglio nella sezione dedicata.

Metodo di Newton

Il metodo di Newton è alternativo al metodo di bisezione per la ricerca degli zeri di una funzione. Dato, all'iterazione k, un valore di tentativo per lo zero esatto x^k, la retta tangente a f in x^k è data da

$$\frac{r(x) - f(x^k)}{x - x^k} = f'(x^k) \quad \Rightarrow \quad r(x) = f(x^k) + f'(x^k)(x - x^k).$$

Notiamo che $r(x) + h.o.t = f(x)$ in un intorno di x^k.



Dato x^k, il punto x^{k+1} è trovato come punto di intersezione della retta tangente con l'asse x ovvero lo zero della retta tangente r approssimante f. Abbiamo che

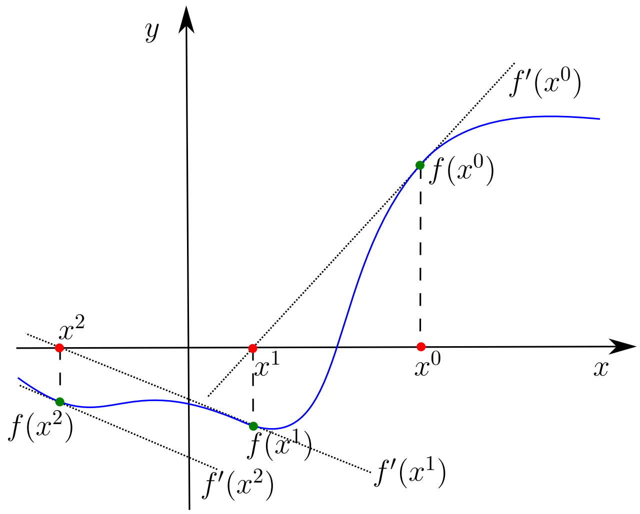
$$r(x^{k+1}) = f(x^k) + f'(x^k)(x^{k+1} - x^k) = 0 \quad \Rightarrow \quad x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)}$$

Possiamo quindi generare una sequenza $\{x^k\}$ e sperare che converga verso lo zero di f sfruttando non solo il valore della funzione, ma anche le informazioni sulla sua derivata. Introduciamo quindi il metodo di Newton

Algoritmo 2.3 - Algoritmo di Newton

Data: Dato x^0;
for $k = 0, 1, 2, \ldots$ **do**
  $x^{k+1} \leftarrow x^k - \dfrac{f(x^k)}{f'(x^k)}$;
  $k \leftarrow k + 1$;
end

Il metodo di Newton, a differenza del metodo di bisezione, è solamente localmente convergente. Significa che la scelta di x^0 ha un impatto sulla convergenza o mancata tale verso α. Inoltre, il valore di f' al denominatore potrebbe diventare arbitrariamente piccolo, se non nullo, andando a creare problemi a livello numerico.



Teorema 2.5

Sia $f : [a,b] \to \mathbb{R}$ una funzione di classe C^2 in $[a,b]$. Sia α tale che $f(\alpha) = 0$ e $f'(\alpha) \neq 0$, ovvero di molteplicità algebrica pari a 1. Allora esiste $\eta > 0$ tale che, se scelgo x^0 in modo che $\left|x^0 - \alpha\right| < \eta$, allora si ha

$$\forall k \in \mathbb{N} \quad \left|x^k - \alpha\right| < \eta,$$

inoltre il metodo risulta convergente, ovvero

$$\lim_{k\to\infty} x^k = \alpha.$$

Infine il metodo di Newton ha una convergenza quadratica, abbiamo

$$\lim_{k\to\infty} \frac{x^{k+1} - \alpha}{(x^k - \alpha)^2} = C = \frac{f''(\alpha)}{2f'(\alpha)}.$$

In teoria il valore esatto, nel caso in cui l'Algoritmo 1.1.3 converge, viene raggiunto solo al limite, ovvero

$$\text{dato } x^0 \xrightarrow{\text{calcolo}} x^1 \xrightarrow{\text{calcolo}} x^2 \xrightarrow{\text{calcolo}} \dots \xrightarrow{\text{calcolo}} x^k \xrightarrow[k\to\infty]{\text{calcolo}} \alpha.$$

Tuttavia, in pratica, definiremo criteri di arresto per determinare quando arrestare l'algoritmo quando l'approssimazione è sufficientemente accurata.

Il teorema che garantisce la convergenza locale di Newton richiede che la radice α abbia molteplicità 1. In questo caso il metodo di Newton se converge, allora converge quadraticamente, ossia

$$\lim_{k\to\infty} \frac{\left|x^{k+1} - \alpha\right|}{\left|x^k - \alpha\right|^2} = C \quad \text{con} \quad C = \frac{f''(\alpha)}{2f'(\alpha)}. \tag{1.4}$$

Nel caso in cui la molteplicità di α risulti maggiore di 1, come ad esempio per $f(x) = x^2$, abbiamo il seguente risultato.

Teorema 2.6

Sia $f \in C^2([a, b])$ e x^0 sufficientemente vicino ad α, ed α radice di f con molteplicità maggiore di 1, il metodo di Newton converge ma solo linearmente. Ovvero

$$\lim_{k\to\infty} \frac{|x^{k+1} - \alpha|}{|x^k - \alpha|} = C \quad \text{con} \quad 0 < C < 1.$$

Quindi se la radice α ha molteplicità 1 allora il metodo di Newton converge quadraticamente, mentre se la molteplicità è maggiore di 1 allora la convergenza è solo lineare. In ogni caso il metodo di Newton converge solo localmente. Per ripristinare la convergenza quadratica nel caso di molteplicità maggiore di 1, introduciamo il metodo di Newton modificato.

Metodo di Newton modificato

Introducendo m come molteplicità algebrica dello zero α, possiamo presentare l'algoritmo associato al metodo di Newton modificato come segue.

Algoritmo 2.4 - Algoritmo di Newton modificato

Data: Dato x^0;
while *(criterio di convergenza)* **do**
  $x^{k+1} \leftarrow x^k - m\dfrac{f(x^k)}{f'(x^k)}$;
  $k \leftarrow k + 1$;
end

In questo caso, se il metodo converge allora converge quadraticamente come in (1.4). Come nel caso del metodo di Newton la convergenza è solo locale, cioè per un x^0 sufficientemente vicino al valore di α. Chiaramente il valore di m potrebbe non essere facile da determinare, può essere tuttavia stimato. La dimostrazione della convergenza e del fattore moltiplicativo m si basano sulla formulazione dell'algorimo come iterazioni di punto fisso.

Comparazione metodi per equazioni non-lineari

Per la determinazione degli zeri α di una funzione f, abbiamo introdotto tre algoritmi: il metodo di bisezione, il metodo di punto fisso e il metodo di Newton. Innanzitutto possiamo osservare che il metodo di punto fisso, poichè la definizione della funzione di iterazione ϕ non è unica, definisce in realtà un insieme di metodi, di cui il metodo di Newton è un caso particolare. Infatti Newton può essere interpretato come un punto fisso con

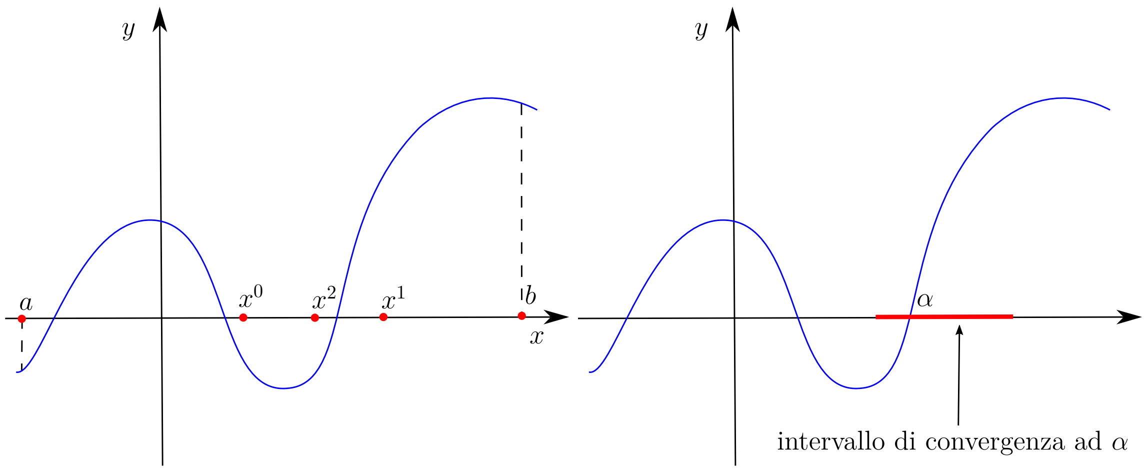
$$\phi(x) = x - \frac{f(x)}{f'(x)}$$

In questo caso

$$\phi'(x) = 1 - \frac{(f'(x))^2 - f(x)f''(x)}{(f'(x))^2} = -\frac{f(x)f''(x)}{(f'(x))^2}$$

La derivata della funzione di iterazione è piccola nell'intorno della soluzione α, perchè $f(\alpha) = 0$, a patto che la derivata f' non sia nulla. Il metodo di Newton converge quadraticamente se α ha molteplicità 1. Se invece la radice α ha molteplicità maggiore di 1 allora il metodo converge solo linearmente. Per recuperare la convergenza quadratica è possibile utilizzare il metodo di Newton modificato.

Il metodo di Newton inoltre richiede il calcolo della derivata di f, che potrebbe non essere immediato da calcolare se la funzione f non è analiticamente nota, come nei casi in cui essa descrive un sistema fisico complesso.



Infine, la convergenza dei metodi di punto fisso e di Newton dipende dalla buona scelta della *guess* iniziale x^0, mentre per il metodo di bisezione è sufficiente identificare un intervallo $[a, b]$. Una strategia pratica può quindi essere quella di effettuare alcune iterazioni del metodo di bisezione per "avvicinarsi" alla soluzione e quindi utilizzare il metodo di Newton per migliorare l'approssimazione sfruttandone la convergenza veloce, ma locale.

Criteri di arresto

Gli algoritmi presentati generano una sequenza $\{x^k\}$ che assumiamo converga allo zero α della funzione f. In pratica però dobbiamo arrestare l'algoritmo quando un certo criterio di convergenza è stato soddisfatto, cioè quando il valore x^k è sufficientemente vicino al valore reale α. Data una tolleranza ϵ, un numero piccolo e positivo, l'idea è di arrestare l'algoritmo quando

$$\left|e^k\right| = \left|x^k - \alpha\right| < \epsilon. \tag{1.5}$$

Tale criterio, seppure ideale, risulta non utilizzabile dato che richiede la conoscenza di α. Dobbiamo quindi utilizzare uno stimatore dell'errore, ovvero una quantità calcolabile che mi garantisce che l'errore è minore o uguale allo stimatore stesso. In questo modo posso terminare l'algoritmo con la certezza che (1.5) sia soddisfatta.

Una scelta comune consiste nel calcolare la differenza tra due iterate successive, ossia verificare se

$$\left|x^{k+1} - x^k\right| < \epsilon.$$

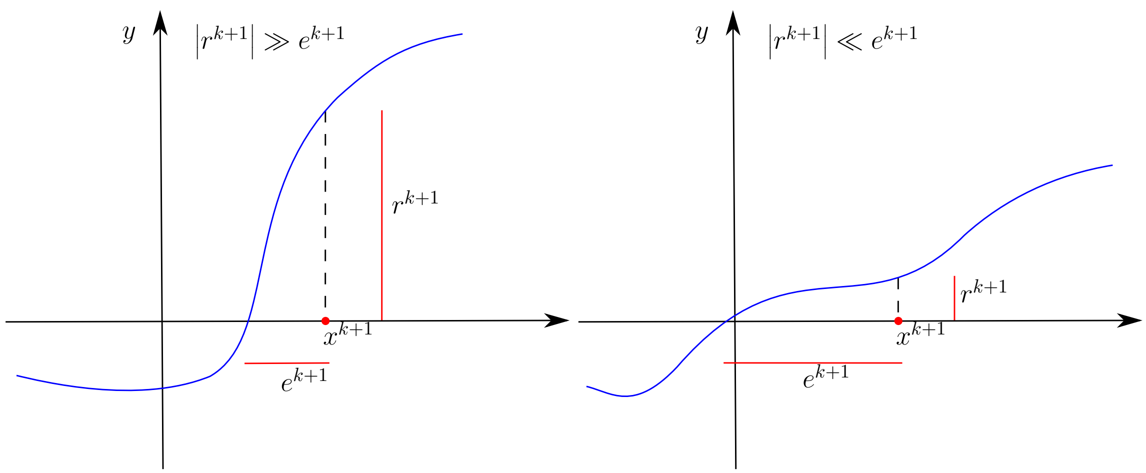
L'idea è che, a convergenza, i punti x^k, x^{k+1} siano sempre più vicini e quindi possiamo dedurre che x^{k+1} fornisce una buona approssimazione di α. Alternativamente, possiamo calcolare il residuo r del problema dato da

$$r^{k+1} = f(x^{k+1}).$$

Poichè vogliamo risolvere il problema $f(\alpha) = 0$ allora il valore del residuo fornisce una stima di quando x^{k+1} risulta lontano rispetto al vero valore di α. L'algoritmo verrà quindi arrestato se

$$\left|r^{k+1}\right| < \epsilon$$

e il valore di α verrà approssimato con x^{k+1}. Osserviamo che non sempre il residuo r fornisce un buon indicatore dell'errore. In particolare è un indicatore affidabile quando $|f'(\alpha)| \approx 1$, infatti considerando la figura seguente



a sinistra abbiamo una curva con $|f'(\alpha)| \gg 1$, ovvero con $|f'(\alpha)|$ molto maggiore di 1, e conseguentemente con $\left|r^{k+1}\right| \gg e^{k+1}$: in questo caso il criterio basato sul residuo è "troppo severo". Nella figura a destra invece otteniamo che $|f'(\alpha)| \approx 0$ che mi fornisce $\left|r^{k+1}\right| \ll e^{k+1}$, ovvero $\left|r^{k+1}\right|$ molto minore di e^{k+1}: in questo caso un residuo basso non implica un errore basso.

Sistemi di equazioni non-lineari

Consideriamo il problema della ricerca degli zeri per un sistema di equazioni non-lineari, ovvero date n incognite vogliamo risolvere n equazioni:

$$\begin{cases} f_1(x_1, x_2, \ldots, x_n) = 0 \\ f_2(x_1, x_2, \ldots, x_n) = 0 \\ \ldots \\ f_n(x_1, x_2, \ldots, x_n) = 0 \end{cases}$$

tale problema può essere scritto in forma compatta introducendo

$$\boldsymbol{f} = [f_1, f_2, \ldots, f_n]^\top \quad \text{e} \quad \boldsymbol{x} = [x_1, x_2, \ldots, x_n]^\top \quad \text{e} \quad \boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_n]^\top.$$

Il problema può essere scritto quindi come: determinare $\boldsymbol{\alpha}$ tale che

$$\boldsymbol{f}(\boldsymbol{\alpha}) = \boldsymbol{0}. \tag{1.6}$$

Possiamo utilizzare il metodo di Newton per la risoluzione di (1.6), adattandolo rispetto al caso di una singola funzione, in cui abbiamo che dato x^k, l'iterazione successiva si calcola come

$$x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)}.$$

Alternativamente possiamo scrivere il calcolo di x^{k+1} come

$$x^{k+1} - x^k = \delta x = -\frac{f(x^k)}{f'(x^k)} \quad \Rightarrow \quad x^{k+1} = x^k + \delta x,$$

dove δx è l'incremento al passo $k+1$ del vettore soluzione. Nel caso si volesse risolvere il problema (1.6), possiamo introdurre il metodo di Newton per sistemi non-lineari. Dato $\boldsymbol{x}^k$ calcoliamo

$$\boldsymbol{x}^{k+1} - \boldsymbol{x}^k = \delta \boldsymbol{x} = -\left[J_f(\boldsymbol{x}^k)\right]^{-1} \boldsymbol{f}(\boldsymbol{x}^k),$$

dove $J_f \in \mathbb{R}^{n\times n}$ è la matrice Jacobiana associata alla funzione $\boldsymbol{f}$, data da

$$J_f = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \dots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \dots & \frac{\partial f_n}{\partial x_n} \end{bmatrix},$$

solitamente indichiamo con $[J_f]_{i,j} = \frac{\partial f_i}{\partial x_j}$, l'elemento (i,j) della matrice Jacobiana. Se volessimo scrivere il metodo in termini di incremento $\delta\boldsymbol{x}$, otteniamo che per il passo k dobbiamo risolvere

$$J_f(\boldsymbol{x}^k)\delta\boldsymbol{x} = -\boldsymbol{f}(\boldsymbol{x}^k) \quad \Rightarrow \quad \delta\boldsymbol{x} = -\left[J_f(\boldsymbol{x}^k)\right]^{-1}\boldsymbol{f}(\boldsymbol{x}^k) \quad \text{e} \quad \boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \delta\boldsymbol{x},$$

che è un sistema lineare in $n\times n$ equazioni e con $\delta\boldsymbol{x}$ come incognita. La soluzione di sistemi lineari, in particolare di grandi sistemi lineari, sarà l'oggetto delle prossime lezioni.

Per avere una convergenza quadratica, dobbiamo richiedere che $f_i \in C^2$ per $i = 1,\dots,n$, inoltre che $\boldsymbol{\alpha}$ sia una radice semplice ovvero che

$$J_f(\boldsymbol{\alpha}) \quad \text{sia invertibile} \quad \Longleftrightarrow \quad \det J_f(\boldsymbol{\alpha}) \neq 0.$$

Infine, la convergenza del metodo è solamente locale come nel caso scalare, cioè la sequenza $\{\boldsymbol{x}^k\}$ generata converge a $\boldsymbol{\alpha}$ se $\boldsymbol{x}^0$ è sufficientemente vicino ad $\boldsymbol{\alpha}$.

Esempio 2.2

Consideriamo un sistema di due equazioni in due incognite:

$$\begin{cases} 1 - x_1^2 - x_2^2 = 0 \\ (x_1 + 2x_2)^2 - \dfrac{3}{2} = 0 \end{cases}$$

Le due equazioni si possono interpretare come le isolinee di livello zero di due superfici, rispettivamente $x_3 = 1 - x_1^2 - x_2^2$, $x_3 = (x_1 + 2x_2)^2 - \dfrac{3}{2}$, e la/le soluzione/i sono i punti di intersezione fra queste due linee di livello, come in figura.

Abbiamo quindi $\boldsymbol{f} = [1 - x_1^2 + x_2^2,\ (x_1 + x_2)^2 - \frac{3}{2}]^\top$. Lo jacobiano è

$$J = \begin{bmatrix} -2x_1 & -2x_2 \\ 2(x_1 + 2x_2) & 4(x_1 + 2x_2) \end{bmatrix}.$$

Se scegliamo $\boldsymbol{x}_0 = [1,\ 1]^\top$ otteniamo la seguente sequenza di soluzioni:

$$\boldsymbol{x}_1 = [1.25,\ 0.75]^\top$$
$$\boldsymbol{x}_2 = [1.0218,\ 0.1409]^\top$$
$$\boldsymbol{x}_3 = [0.9939,\ 0.1166]^\top$$
$$\boldsymbol{x}_4 = [0.9933,\ 0.1157]^\top$$