Esercitazione 4
Soluzione di Equazioni Non Lineari

Metodo delle iterazioni di punto fisso

Consideriamo il problema: data una funzione $\phi : [a, b] \to \mathbb{R}$, trovare $\alpha \in [a, b]$ tale che:

$$\alpha = \phi(\alpha).$$

Se un tale α esiste, viene detto *punto fisso* di ϕ. Se la funzione di iterazione è sufficientemente regolare e $|\phi'(\alpha)| < 1$ la successione definita da

$$x^{(k+1)} = \phi(x^{(k)}), \quad k \geq 0 \tag{1}$$

converge ad α per ogni scelta del dato iniziale $x^{(0)}$ in un intorno opportuno di α. Tale successione soddisfa la condizione di convergenza

$$\lim_{k\to\infty} \frac{x^{(k+1)} - \alpha}{x^{(k)} - \alpha} = \phi'(\alpha), \tag{2}$$

ovvero (per k grande) l'errore al passo $(k+1)$ è uguale all'errore al passo k moltiplicato per una costante $\phi'(\alpha)$ il cui valore assoluto è minore di 1 (ordine di convergenza uguale almeno ad 1, o convergenza almeno *lineare*).

In maniera del tutto generale si può dimostrare che se le derivate i-esime della funzione ϕ valutate in α si annullano per $i = 1, \cdots, p$ con $p \geq 1$ e $\phi^{(p+1)}(\alpha) \neq 0$, allora il metodo di punto fisso ha ordine $(p + 1)$ e vale:

$$\lim_{k\to\infty} \frac{x^{(k+1)} - \alpha}{(x^{(k)} - \alpha)^{p+1}} = \frac{\phi^{(p+1)}(\alpha)}{(p + 1)!}. \tag{3}$$

Ad esempio, se $\phi'(\alpha) = 0$, (ma $\phi''(\alpha) \neq 0$) il metodo di punto fisso è convergente di ordine 2.

Le iterazioni di punto fisso possono servire anche per il calcolo degli zeri di una funzione $f(x)$. In generale, la funzione di iterazione ϕ deve essere scelta in modo tale che

$$\phi(\alpha) = \alpha \text{ ogni volta che } f(\alpha) = 0.$$

La scelta di ϕ ovviamente non è unica. Si può ricorrere infatti a manipolazioni algebriche differenti di f per ottenere delle possibili funzioni di iterazione ϕ. Ad esempio, ogni funzione della forma $\phi(x) = x + F(f(x))$ è una funzione di iterazione ammissibile, purchè F sia una funzione continua tale che $F(0) = 0$.

Una semplice implementazione dell'algoritmo di punto fisso è la seguente:

```
function [succ, it] = ptofis(x0, phi, nmax, toll)
%
% [succ, it] = ptofis(x0, phi, nmax, toll)
% Metodo di punto fisso x = phi(x)
%
% --------Parametri di ingresso:
% x0      Punto di partenza
% phi     Funzione di punto fisso (definita inline o anonimous)
% nmax    Numero massimo di iterazioni
% toll    Tolleranza sul test d'arresto
%
```

```
% --------Parametri di uscita:
% succ   Vett. contenente tutte le iterate calcolate
%         (l'ultima componente e' la soluzione)
% it      Iterazioni effettuate
err   = 1 + toll;
it    = 0;
succ  = x0;
xv    = x0;
while (it < nmax && err > toll)
   xn    = phi(xv);
   err   = abs(xn - xv);
   succ = [succ; xn];
   it    = it + 1;
   xv    = xn;
end
fprintf(' \n Numero di Iterazioni    : %d \n',it);
fprintf(' Punto fisso calcolato   : %12.13f \n',succ(end));
```

Spesso può essere interessante visualizzare le iterate successive dell'algoritmo di punto fisso; anzichè visualizzare i valori generati $x^{(k)}$ al variare dell'indice k, si preferisce costruire un grafico formato da una linea spezzata che congiunge i punti di coordinate:

$$\left(x^{(k)}, \phi(x^{(k)})\right) \longrightarrow \left(x^{(k+1)}, x^{(k+1)}\right) \longrightarrow \left(x^{(k+1)}, \phi(x^{(k+1)})\right) \cdots, \ k \geq 0.$$

Il grafico così ottenuto consente di distinguere immediatamente i punti fissi attrattori dai punti fissi repulsori.

Esercizio 1

Si consideri la funzione $f(x) = \cos^2(2x) - x^2$.

1. Si disegni la funzione e si individuino graficamente i punti in cui $f(x) = 0$.
2. Si verifichi teoricamente per quale intervallo di valori della costante A il metodo di punto fisso per la ricerca degli zeri di $f(x)$ con funzione di iterazione:

   $$\phi(x) = x + Af(x)$$

   può convergere allo zero $\alpha > 0$ per una scelta opportuna del dato iniziale (criterio di convergenza locale).
3. Si utilizzi la `function ptofis.m` con $A = 0.1$ e $x^{(0)} = 0.1$ per ottenere un valore dello zero α, scegliendo come tolleranza 10^{-10}. Si usi il valore dello zero ottenuta per verificare numericamente il range di valori ammissibili per A ottenuto teoricamente al punto precedente. Si studi la sensibilità della convergenza del metodo al variare di A e $x^{(0)}$.
4. Si trovi lo zero della funzione f usando in successione le `function bisection.m` e `ptofisso.m` per
   (a) stimare la guess iniziale $x^{(0)}$ tramite alcune iterazioni dell'algoritmo di bisezione;
   (b) applicare il metodo di punto fisso con la scelta conveniente di $x^{(0)}$ trovata nel punto precedente.
5. Stimare l'ordine di convergenza e il fattore di convergenza del metodo di punto fisso al variare di A utilizzando la funzione `stimap.m`, verificando l'affidabilità delle stime teoriche.
6. Fornire un valore di A tale da ottenere un metodo del secondo ordine.
7. Ricordando che il metodo di Newton può essere riletto come metodo di punto fisso, implementarlo utilizzando la `function ptofis.m`. Determinare sperimentalmente gli intervalli in cui scegliere il dato iniziale in modo che il metodo converga allo zero α positiva.

Metodo di Newton per equazioni non lineari

Il metodo di Newton per la ricerca dello zero α di un'equazione non lineare $f(x) = 0$ è un metodo iterativo che necessita della conoscenza della derivata prima $f'(x)$ della funzione $f(x)$.

Dato $x^{(0)}$, dato iniziale, la formula della generica iterazione k del metodo di Newton si esprime come:

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})} \quad \text{se } f'(x^{(k)}) \neq 0,\ k \geq 0. \tag{4}$$

Nel caso in cui α sia uno zero semplice di f ($f'(\alpha) \neq 0$) il metodo di Newton converge quadraticamente. Se invece la molteplicità è maggiore di uno, il metodo di Newton converge linearmente.

Detta m la molteplicità dello zero α, la convergenza quadratica può essere recuperata modificando la formula generale del metodo di Newton nel modo seguente:

$$x^{(k+1)} = x^{(k)} - m\frac{f(x^{(k)})}{f'(x^{(k)})} \quad \text{se } f'(x^{(k)}) \neq 0,\ k \geq 0. \tag{5}$$

Il metodo (5) si chiama *Newton modificato*.

Per arrestare il metodo di Newton si utilizza generalmente il criterio seguente: ad ogni iterazione k si valuta se la differenza tra due iterate successive è inferiore ad una certa soglia ε:

$$|x^{(k)} - x^{(k-1)}| < \varepsilon \tag{6}$$

Esercizio 2

Vogliamo risolvere il problema della ricerca degli zeri dell'equazione non lineare $f(x) = 0$, dove f è definita da:

$$f(x) = x^3 - (2+e)x^2 + (2e+1)x + (1-e) - \cosh(x-1), \quad x \in [0.5, 6.5] \tag{7}$$

1. Disegnare i grafici della funzione f e f' nell'intervallo $[0.5, 6.5]$ ed evidenziare le radici della funzione. (Suggerimento: utilizzare il comando `grid on`).

2. Dai grafici di f e di f' discutere le proprietà di convergenza del metodo di Newton per tutti gli zeri, valutando l'opportunità di applicare il metodo di Newton modificato. Noto il valore esatto dello zero $\alpha_1 = 1$ si valuti la molteplicità dello zero.

3. Implementare il metodo di Newton per la risoluzione del problema della ricerca degli zeri di una funzione non lineare f.
   L'intestazione della funzione `newton.m` deve essere:

   ```
   function [xvect,it]=newton(x0,nmax,toll,fun,dfun,mol)
   ```

   La funzione prende in ingresso il dato iniziale $x^{(0)}$, il numero massimo di iterazioni, il valore della tolleranza ε necessario per il criterio d'arresto, la funzione f di cui si stanno ricercando gli zeri, la sua derivata f' e per ultimo il valore della molteplicitá dello zero.
   In uscita la funzione restituisce il vettore `xvect` contenente tutte le iterate calcolate (l'ultima componente sará perció la soluzione) e il numero di iterazioni effettuate.

4. Risolvere il problema della ricerca degli zeri della funzione $f(x)$ definita in (7), utilizzando la funzione `newton.m` implementata al punto precedente. Si ponga la tolleranza pari a 10^{-6} e si fissi un numero massimo di iterazioni a scelta. Si utilizzi per il calcolo di ogni zero un opportuno dato iniziale `x0`, sulla base delle osservazioni fatte ai punti precedenti, eventualmente ricorrendo all'algoritmo di bisezione.
   Nel caso dello zero $\alpha_1 = 1$ si risolva il problema sia con il metodo di Newton classico (`mol`$= 1$), sia con il metodo di Newton modificato (`mol` fissato opportunamente). Si riportino su un grafico in scala semilogaritmica gli andamenti degli errori in funzione del numero di iterazioni in entrambi i casi.

5. Stimare l'ordine di convergenza dei due metodi (Newton classico e Newton modificato nel caso $\alpha_1 = 1$) mediante la funzione Matlab® `stimap.m` fornita.

Metodo di Newton per sistemi di equazioni non lineari

Si consideri il sistema di equazioni non lineari composto da n funzioni non lineari in n incognite:

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

Chiamando $\mathbf{F} \equiv (f_1, \ldots, f_n)^T$ e $\mathbf{x} \equiv (x_1, \ldots, x_n)^T$, il problema può essere riscritto nella forma:

$$\text{data } \mathbf{F} : \mathbb{R}^n \to \mathbb{R}^n, \text{ trovare } \mathbf{x}^\star \in \mathbb{R}^n \text{ tale che } \mathbf{F}(\mathbf{x}^\star) = \mathbf{0}. \tag{8}$$

Al fine di estendere il metodo di Newton al caso di un sistema, si costruisca la *matrice Jacobiana* $\mathbf{J_F}$ della funzione vettoriale $\mathbf{F}$, le cui componenti sono

$$(\mathbf{J_F})_{ij} \equiv \frac{\partial f_i}{\partial x_j}, \quad i, j = 1, \ldots, n.$$

Con questa notazione, il metodo di Newton per sistemi di equazioni non lineari diventa:

Dato	$\mathbf{x}^{(0)} \in \mathbb{R}^n$, per $k = 0, \ldots,$ fino a convergenza
risolvere	$\mathbf{J_F}(\mathbf{x}^{(k)})\delta\mathbf{x}^{(k)} = -\mathbf{F}(\mathbf{x}^{(k)})$
porre	$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \delta\mathbf{x}^{(k)}$

Per quanto riguarda l'aspetto implementativo del metodo, si noti che, ad ogni passo, esso richiede:

1. la costruzione di un vettore colonna di dimensione n, tramite valutazione di $\mathbf{F}$ nel punto $\mathbf{x}^{(k)}$;
2. la costruzione di una matrice $n \times n$, tramite valutazione della matrice jacobiana $\mathbf{J_F}$ nel punto $\mathbf{x}^{(k)}$;
3. la soluzione di un sistema lineare di matrice $\mathbf{J_F}(\mathbf{x}^{(k)})$ e termine noto $-\mathbf{F}(\mathbf{x}^{(k)})$;
4. l'aggiornamento della soluzione.

L'algoritmo giunge a convergenza quando la norma euclidea dell'incremento $\delta\mathbf{x}^{(k)}$ è minore di una tolleranza ε fissata a priori:

$$\|\delta\mathbf{x}^{(k)}\| < \varepsilon$$

Come nel caso scalare, la convergenza del metodo dipende dalla scelta del dato iniziale $\mathbf{x}^{(0)}$. Inoltre, ad ogni passo, la matrice $\mathbf{J_F}(\mathbf{x}^{(k)})$ deve essere non singolare. Si osservi anche che il costo richiesto per la risoluzione del sistema lineare può essere eccessivamente elevato per n grande, e che $\mathbf{J_F}(\mathbf{x}^{(k)})$ può essere mal condizionata, rendendo difficile ottenere una soluzione accurata.

Esercizio 3

Consideriamo il sistema non lineare seguente nella forma $\mathbf{F}(\mathbf{x}) = \mathbf{0}$, dove

$$\begin{cases} F_1(x_1, x_2) = x_1^2 + x_2^2 - 1 = 0 \\ F_2(x_1, x_2) = \sin(x_1\pi/2) + x_2^3 = 0 \end{cases}$$

1. Scrivere la anonymous function Matlab® che preso in input un generico vettore `x`, restituisce un vettore contenente la valutazione di $\mathbf{F}$ in `x`.
   La sintassi sarà: `F = @(x) [ ... ; ... ]`.

2. Scrivere la anonymous function Matlab® che preso in input un generico vettore `x`, restituisce la matrice Jacobiana $\mathbf{J_F}$ contente la valutazione dello Jacobiano della funzione $\mathbf{F}$ in `x`.
   La sintassi sarà `JF = @(x) [ ... , ... ; ... , ... ]`.

3. Implementare il metodo di Newton per la risoluzione della ricerca degli zeri di un sistema non lineare $\mathbf{F}(\mathbf{x}) = \mathbf{0}$.
   L'intestazione della funzione `newtonsys.m` deve essere:

   ```
   function [x, R, niter] = newtonsys(F, JF, x0, tol, nmax)
   ```

   La funzione prende in ingresso la funzione `F` che restituisce la valutazione di $\mathbf{F}$, la funzione `JF` che restituisce la valutazione dello Jacobiano della funzione $\mathbf{F}$, il dato iniziale `x0`, il valore della tolleranza necessario per valutare il criterio d'arresto e il numero massimo di iterazioni.
   In uscita la funzione restituisce il vettore `x` contenente la soluzione del sistema non lineare, il valore del residuo valutato in corrispondenza della soluzione finale e il numero delle iterazioni effettuate.

4. Partendo dal dato iniziale `x0 = [1; 1]`, prendendo come tolleranza `tol = 1e-5` e numero di iterazioni massime `nmax = 10`, risolvere il problema $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ usando la funzione `newtonsys.m` implementata al punto precedente.

Esercizio 4

Si consideri l'equazione non lineare $f(x) = 0$, in cui f è definita come:

$$f(x) = \arctan\left(7\left(x - \frac{\pi}{2}\right)\right) + \sin\left(\left(x - \frac{\pi}{2}\right)^3\right), \quad x \in [-1, 6]. \tag{9}$$

1. Si disegni il grafico della funzione f nell'intervallo $[-1, 6]$.

2. Si verifichi che l'unico zero α della funzione f è semplice e si utilizzi la funzione `newton.m` per approssimarlo. Si assuma una tolleranza pari a 10^{-10} e dato iniziale `x0` $= 1.5$. Successivamente si ripeta tale calcolo partendo da `x0` $= 4$. Sapendo che $\alpha = \pi/2$, si calcolino gli errori assoluti nei due casi e si motivino i risultati ottenuti. Si stimino inoltre gli ordini di convergenza mediante la funzione `stimap.m` giá fornita.

Esercizio 5

Ripetere l'Esercizio 3 considerando come sistema non lineare

$$\begin{cases} F_1(x_1, x_2, x_3) = -\dfrac{1}{81}\cos x_1 - x_1 + \dfrac{1}{9}x_2^2 + \dfrac{1}{3}\sin x_3 = 0 \\ F_2(x_1, x_2, x_3) = \dfrac{1}{3}\sin x_1 - x_2 + \dfrac{1}{3}\cos x_3 = 0 \\ F_3(x_1, x_2, x_3) = -\dfrac{1}{9}\cos x_1 + \dfrac{1}{3}x_2 + \dfrac{1}{6}\sin x_3 - x_3 = 0\,. \end{cases}$$

avente come soluzione il vettore `x=[0;1/3;0]`, partendo dal dato iniziale `x0 = [1;1;1]`, considerando come tolleranza `tol = 1e-10` e numero di iterazioni massime `nmax = 10`.