Serie 5
Ottimizzazione Numerica

Ottimizzazione Non Vincolata

Si consideri una funzione $\Phi(\mathbf{y})$ definita per $\mathbf{y} \in \mathbb{R}^n$, ovvero $\Phi : \mathbb{R}^n \to \mathbb{R}$, detta *funzione obiettivo* (o funzione costo). L'ottimizzazione non vincolata della funzione obiettivo Φ consiste nel trovare il *punto di minimo* $\mathbf{x} \in \mathbb{R}^n$ tale che

$$\mathbf{x} = \arg\min_{\mathbf{y}\in\mathbb{R}^n} \Phi(\mathbf{y}).$$

Condizioni di ottimalità

Le condizioni di ottimalità sono le condizioni matematiche che un punto di minimo deve soddisfare affinché sia una soluzione del problema di minimo (e dunque del problema di ottimizzazione). In particolare, sotto opportune ipotesi di differenziabilità di $\Phi(\mathbf{x})$, il gradiente $\nabla\Phi(\mathbf{x})$ della funzione obiettivo $\Phi(\mathbf{x})$ corrisponde ad un sistema di equazioni non lineari per la funzione vettoriale

$$\mathbf{F}(\mathbf{x}) = \nabla\Phi(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^n.$$

Si ha quindi una corrispondenza tra problema di minimo e soluzione di un sistema di equazioni non lineari:

$$\begin{array}{ccccc} \min_{\mathbf{x}\in\mathbb{R}^n} \Phi(\mathbf{x}) & \leftrightarrow & \text{trovare } \mathbf{x} \in \mathbb{R}^n : \nabla\Phi(\mathbf{x}) = \mathbf{0} & \leftrightarrow & \text{Hessiana di } \Phi : H_\Phi(\mathbf{y}) \in \mathbb{R}^{n\times n} \\ & & \updownarrow & & \updownarrow \\ & & \text{trovare } \mathbf{x} \in \mathbb{R}^n : \mathbf{F}(\mathbf{x}) = \mathbf{0} & \leftrightarrow & \text{Jacobiana di } \mathbf{F} : J_{\mathbf{F}}(\mathbf{y}) \in \mathbb{R}^{n\times n} \end{array}$$

1 Metodi di tipo *Derivative-Free*

Per problemi di ottimizzazione in cui non si può o si vuole evitare il calcolo esplicito del gradiente della funzione obiettivo Φ, la ricerca del minimo è eseguita solamente confrontando i valori della funzione Φ.

1.1 Metodo della sezione aurea

Si consideri il caso monodimensionale, in cui è data una funzione obiettivo

$$\Phi : [a, b] \to \mathbb{R} .$$

Assumiamo Φ continua in $[a, b]$ e dotata di un unico punto di minimo $x \in (a, b)$.

Il metodo della sezione aurea è così detto in quanto sfrutta il *rapporto aureo* (o sezione aurea) $\varphi = \frac{1+\sqrt{5}}{2} \simeq 1.6180339887$ per suddividere iterativamente l'intervallo di partenza (a, b) in sotto-intervalli in cui è localizzato il minimo.

In particolare, si suddivide (a, b) determinando

$$c = a + \frac{b-a}{\varphi+1} \qquad \text{e} \qquad d = a + \frac{b-a}{\varphi},$$

per cui $c < d$.

- Se $\Phi(c) > \Phi(d)$, il punto di minimo si trova nell'intervallo $[c, b]$, quindi si aggiorna $a = c$;
- se $\Phi(c) \leq \Phi(d)$, il punto di minimo si trova nell'intervallo $[a, d]$, quindi si aggiorna $b = d$.

Procedendo iterativamente, si generano sottointervalli $I^{(k)} = (a^{(k)}, b^{(k)})$ di lunghezza sempre minore e il punto di minimo x della funzione obiettivo viene ottenuto come limite della successione dei punti medi $x^{(k)}$.

Si pone $a^{(0)} = a$, $b^{(0)} = b$ e $x^{(0)} = \dfrac{a^{(0)} + b^{(0)}}{2}$ e per $k = 0, 1, \ldots$

$$c^{(k)} = a^{(k)} + \frac{b^{(k)} - a^{(k)}}{\varphi+1}$$
$$d^{(k)} = a^{(k)} + \frac{b^{(k)} - a^{(k)}}{\varphi}$$

Se $\Phi(c^{(k)}) > \Phi(d^{(k)})$

$$a^{(k+1)} = c^{(k)} \text{ e } b^{(k+1)} = b^{(k)}$$

Altrimenti

$$a^{(k+1)} = a^{(k)} \text{ e } b^{(k+1)} = d^{(k)}$$

Infine

$$x^{(k+1)} = \frac{a^{(k+1)} + b^{(k+1)}}{2}$$

Si termina l'algoritmo quando è soddisfatto un opportuno criterio di arresto. In questo laboratorio scegliamo di arrestare le iterazioni quando $\widetilde{e}^{(k)} < tol$, dove $\widetilde{e}^{(k)}$ è la metà della misura del sottointervallo $I^{(k)}$.

In particolare, vale la seguente stima dell'errore:

$$e^{(k)} := |x^{(k)} - x| \leq \widetilde{e}^{(k)} := \frac{|I^{(k)}|}{2} = \frac{b-a}{2\,\varphi^k} \qquad \text{per ogni } k \geq 0.$$

Esercizio 1.1

Si consideri la funzione obiettivo $\Phi : [a, b] \to \mathbb{R}$

$$\Phi(y) = -\frac{y^4}{2} + 4y^3 - 7y^2 - 4y + 10$$

con $a = -1$ e $b = 3$.

1. Si rappresenti graficamente la funzione e si verifichi che esiste un minimo $x = 2$.
2. Si implementi una funzione `sezione_aurea.m` per il metodo della sezione aurea. La funzione deve avere la seguente intestazione:

```
[xv, k, errv] = sezione_aurea(f, a, b, tol, nmax),
```

dove `f` è la funzione obiettivo da minimizzare, definita come anonymous function, `a` e `b` sono gli estremi dell'intervallo di definizione della funzione, `tol` è la tolleranza (per il criterio di arresto basato sullo stimatore dell'errore), `nmax` è il numero massimo di iterazioni; in uscita la funzione restituisce la successione `xv` delle approssimazioni del minimo della funzione obiettivo, il numero di iterazioni svolte `k` e il vettore `errv` contenente la stima dell'errore ad ogni iterazione.

3. Si applichi la funzione appena implementata per la ricerca del punto di minimo di Φ, utilizzando una tolleranza pari a 10^{-6} e un numero massimo di iterazioni pari a 100.

4. Si valuti l'errore commesso rispetto alla soluzione esatta e lo si confronti con la stima teorica. In particolare, si rappresenti graficamente l'andamento di entrambe le quantità rispetto al numero di iterazioni in un grafico in scala semilogaritmica sull'asse delle ordinate.

2 Metodi di Discesa o Line-Search

I metodi di discesa, o *line-search* methods, sono una famiglia di metodi iterativi la cui idea di base è muoversi nella direzione che riduce maggiormente il valore della funzione, fino ad approssimare un punto minimo locale.

La generica iterata di un metodo di discesa è

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \, \mathbf{d}^{(k)} \qquad \text{per } k = 0, 1, \ldots,$$

dove $\mathbf{x}^{(k)} \in \mathbb{R}^n$ è il punto di partenza, $\mathbf{d}^{(k)}$ è la direzione di discesa e $\alpha_k \in \mathbb{R}$ determina l'ampiezza del passo da compiere lungo $\mathbf{d}^{(k)}$.

Dato che i metodi di discesa sfruttano tutti il calcolo del gradiente della funzione obiettivo, un possibile criterio d'arresto per il metodo iterativo consiste nel considerare lo stimatore dell'errore $\widetilde{e}^{(k)} = \left\|\nabla\Phi(\mathbf{x}^{(k)}\right\|$ e fermare le iterazioni dell'algoritmo all'iterata k-esima tale per cui risulta $\widetilde{e}^{(k)} < tol$.

2.1 Condizioni di Wolfe

Date l'iterata $\mathbf{x}^{(k)}$ e la direzione di discesa $\mathbf{d}^{(k)}$, il metodo di line-search cerca un valore scalare $\alpha_k \in \mathbb{R}$ tale per cui la nuova iterata $\mathbf{x}^{(k+1)}$ riduca la funzione obiettivo Φ il più possibile. Abbiamo dunque:

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}} \Phi(\mathbf{x}^{(k)} + \alpha \, \mathbf{d}^{(k)}) \qquad \text{per } k = 0, 1, \ldots,$$

che – a sua volta – può essere visto come un problema di ottimizzazione (da ripetersi per ogni iterata k). Questo problema può essere a sua volta risolto in maniera approssimata tramite uno dei metodi di tipo *derivative-free*.

Un possibile criterio di scelta del parametro α_k in una direzione di discesa $\mathbf{d}^{(k)}$ è rappresentato dalle condizioni di Wolfe (*forti*):

$(CW1)$ $\quad \Phi(\mathbf{x}^{(k)} + \alpha_k\,\mathbf{d}^{(k)}) \leq \Phi(\mathbf{x}^{(k)}) + c_1\,\alpha_k\,\nabla\Phi(\mathbf{x}^{(k)})^T\,\mathbf{d}^{(k)} \qquad \text{con } 0 < c_1 < 1,$

$(CW2)$ $\quad \left|\nabla\Phi(\mathbf{x}^{(k)} + \alpha_k\,\mathbf{d}^{(k)})^T\,\mathbf{d}^{(k)}\right| \leq c_2\,\left|\nabla\Phi(\mathbf{x}^{(k)})^T\,\mathbf{d}^{(k)}\right| \qquad \text{con } 0 < c_1 < c_2 < 1.$

Un algoritmo spesso utilizzato per trovare un valore adatto per α_k nel metodo di discesa (da applicarsi per ciascuna iterazione k-esima dell'algoritmo) è l'algoritmo (iterativo) di *backtracking*. La condizione $(CW1)$ è anche nota come regola di Armijo.

2.2 Metodi del gradiente

Il *metodo del gradiente* consiste nello scegliere la k-esima direzione di discesa in come:

$$\mathbf{d}^{(k)} = -\nabla\Phi(\mathbf{x}^{(k)}) \qquad \text{per } k = 0, 1, \ldots,$$

per poi selezionare α_k per esempio tramite l'algoritmo di backtracking o comunque in modo che le condizioni di Wolfe vengano soddisfatte.

Il *metodo del gradiente coniugato* consiste nello scegliere la k-esima direzione di discesa come segue

$$\begin{aligned}\mathbf{d}^{(0)} &= -\nabla\Phi(\mathbf{x}^{(0)})\\ \mathbf{d}^{(k+1)} &= -\nabla\Phi(\mathbf{x}^{(k+1)}) + \beta_k\,\mathbf{d}^{(k)} \qquad \text{per } k = 0, 1, \ldots,\end{aligned}$$

dove la scelta del parametro $\beta_k \in \mathbb{R}$ viene determinata secondo diversi criteri (il parametro α_k viene poi determinato in modo da soddisfare le condizioni di Wolfe). In questo laboratorio consideriamo solo la seguente scelta del parametro β_k per $k = 0, 1, \ldots$:

$$\beta_k^{FR} = \frac{\|\nabla\Phi(\mathbf{x}^{(k+1)})\|^2}{\|\nabla\Phi(\mathbf{x}^{(k)})\|^2} \qquad \text{Fletcher–Reeves}$$

Osservando che nel caso $\beta_k = 0$, il metodo del gradiente coniugato coincide con il metodo del gradiente, possiamo riassumere i metodi del gradiente nel seguente algoritmo.

Data l'iterata iniziale $\mathbf{x}^{(0)}$ e posta $\mathbf{d}^{(0)} = -\nabla\Phi(\mathbf{x}^{(0)})$; per ogni iterata $k \geq 0$:

(i) Determiniamo il valore di α_k

(ii) Consideriamo la generica iterata del metodo del gradiente

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\,\mathbf{d}^{(k)}$$

(iii) Determiniamo il parametro β_k e aggiorniamo la direzione di discesa

$$\mathbf{d}^{(k+1)} = -\nabla\Phi(\mathbf{x}^{(k+1)}) + \beta_k\,\mathbf{d}^{(k)}$$

.

Si termina l'algoritmo quando è soddisfatto un opportuno criterio di arresto. In questo laboratorio scegliamo di arrestare le iterazioni quando $\widetilde{e}^{(k)} < tol$, dove $\widetilde{e}^{(k)} = \|\nabla\Phi(\mathbf{x}^{(k)})\|_2$.

Considerata una generica funzione obiettivo $\Phi : \mathbb{R}^n \to \mathbb{R}$ tale che $\Phi \in C^2\left(B_r(\mathbf{x})\right)$ (per qualche $r > 0$), in generale, il metodo del gradiente è convergente con ordine p inferiore o al più uguale a 1 (convergenza lineare). Il metodo del gradiente coniugato invece tende a convergere perlomeno linearmente (ordine $p = 1$), o superlinearmente.

Esercizio 2.1

Si consideri il problema di ottimizzazione applicato alla meccanica del volo già presentato nell'Esempio 5.1.1 delle note del corso. Si vuole massimizzare l'efficienza aerodinamica in condizioni di crociera, ossia massimizzare il rapporto *portanza/resistenza* che supponiamo dipendere da soli due parametri: l'angolo di attacco α e la deflessione della superficie di controllo δ. Date le seguenti espressioni per la portanza L e la resistenza D

$$C_L(\alpha, \delta) = C_{L0} + C_{L\alpha}\alpha + C_{L\delta}\delta$$
$$C_D(\alpha, \delta) = C_{D0} + C_{D\alpha}\alpha^2 + C_{D\delta}\delta^2$$

si definisce il funzionale $\Phi(\mathbf{y}) : \mathbb{R}^2 \to \mathbb{R}$ con $\mathbf{y} = (y_1, y_2) = (\alpha, \delta)$ come l'efficienza aerodinamica cambiata di segno:

$$\Phi(\mathbf{y}) = -\frac{C_L(\mathbf{y})}{C_D(\mathbf{y})}$$

L'obiettivo è trovare la coppia di parametri $\mathbf{x} = (\alpha^*, \delta^*)$ tali che

$$\Phi(\mathbf{x}) = \min_{\mathbf{y}\in\mathbb{R}^2} \Phi(\mathbf{y})$$

Sono assegnati i coefficienti di portanza e resistenza $C_{L0} = 0$, $C_{L\alpha} = 5\,\mathrm{rad}^{-1}$, $C_{L\delta} = 0.3\,\mathrm{rad}^{-1}$, $C_{D0} = 0.02$, $C_{D\alpha} = 0.5\,\mathrm{rad}^{-2}$ e $C_{D\delta} = 0.05\,\mathrm{rad}^{-2}$. Dati questi valori, il gradiente $\nabla\Phi$ assume la forma

$$\nabla\Phi(y_1,\, y_2) = \begin{pmatrix} 500\,(50\,y_1^2 + 6\,y_1\,y_2 - 5\,y_2^2 - 2)/(50\,y_1^2 + 5\,y_2^2 + 2)^2 \\ 10\,(-150\,y_1^2 + 500\,y_1\,y_2 + 15\,y_2^2 - 6)/(50\,y_1^2 + 5\,y_2^2 + 2)^2 \end{pmatrix}.$$

Analiticamente si ottiene che il punto di minimo della funzione obiettivo Φ si trova in $\mathbf{x} = (0.196494373584908, 0.117896623783779)^T$.

1. Si definisca il funzionale su Matlab® e lo si rappresenti graficamente, insieme alla sue curve di livello, negli intervalli $y_1 \in [0, 0.4]$ e $y_2 \in [-0.1, 0.3]$ (*Suggerimento*: si usino i comandi `meshgrid`, `surf` e `contourf`). Si verifichi che esiste un minimo in $\mathbf{x}$.

2. Viene fornito un template della function `gradiente_coniguato` con intestazione

   ```
   [xvect, it] = gradiente_coniugato(Phi, GradPhi, flag_metodo, x0, toll, nmax)
   ```

   dove `Phi` è la funzione obiettivo da minimizzare, definita come anonymous function che prende in input due variabili, `GradPhi` è il gradiente di Φ (anonymous function con due variabili in input) e `flag_metodo` è una stringa che permette di scegliere il metodo per assegnare β_k (gradiente `'G'`, Fletcher-Reeves `'FR'`), `x0` è la guess iniziale, `toll` è la tolleranza (per il criterio di arresto basato sulla norma del gradiente), `nmax` è il numero massimo di iterazioni; in uscita la funzione restituisce la successione `xvect` delle approssimazioni del minimo della funzione obiettivo e il numero di iterazioni svolte `it`. Osserviamo che gli algoritmi del gradiente e gradiente coniugato non necessitano dell'espressione di Φ, tuttavia questa è rischiesta per la verifica della condizione di Wolfe $(CW1)$ nell'algoritmo di backtracking.

   Completare l'implementazione della function per i metodi sopra-citati.

   Suggerimento: All'interno della funzione `gradiente_coniugato.m` si richiami la seguente funzione `backtracking.m` già fornita su WeBeep

```
[alphak, it] = backtracking(Phi, GradPhi, xk, dk, c1, rho, nmax )
```

da applicare a ogni iterazione k-esima dell'algoritmo e che restituisce il valore di step-size α_k. Si noti che nel template di `gradiente_coniugato.m` sono fissati dei parametri di default da fornire in input a `backtracking.m`, ma possono essere calibrati nel caso fosse necessario migliorare le proprietà di convergenza del metodo.

3. Risolvere il problema di minimizzazione del funzionale tramite il metodo del gradiente (`flag_method='G'`) e il metodo del gradiente coniugato con β_k di Fletcher-Reeves (`flag_method='FR'`) e commentare i risultati ottenuti. Si considerino $\mathbf{x}^{(0)} = (0.1, 0.05)^T$, `toll` $= 10^{-5}$ e `nmax` $= 100$. In input alla function `backtracking.m` si tengano i parametri di default forniti (`c1_bt = 1e-4`, `rho_bt = 0.3` `nmax_bt = 10`).

4. Si considerino ora $\mathbf{x}^{(0)} = (0, 0.1)^T$, `toll` $= 10^{-5}$ e `nmax` $= 100$ e si risolva il problema di ottimizzazione con i due metodi del gradiente. In caso di mancata convergenza, provare un set di parametri alternativo per il backtracking: `c1_bt = 2e-1`, `rho_bt = 0.45`. Commentare i risultati ottenuti.

5. Si riporti in scala semilogaritmica rispetto all'asse y l'andamento degli errori per il caso $\mathbf{x}^{(0)} = (0.1, 0.05)^T$ in funzione del numero di iterazioni per tutti i metodi testati.

2.3 Metodi di Newton e quasi-Newton

I metodi del gradiente visti nella sezione precedente sfruttano solo la conoscenza del gradiente della funzione obiettivo, ovvero $\nabla\Phi : \mathbb{R}^n \to \mathbb{R}^n$. Tuttavia, non incorporano informazioni aggiuntive sulla curvatura della funzione obiettivo $\Phi : \mathbb{R}^n \to \mathbb{R}$, ovvero non utilizzano l'informazione fornita dalla matrice Hessiana $H_\Phi : \mathbb{R}^n \to \mathbb{R}^{n\times n}$.

Metodo di Newton

Il *metodo di Newton* visto nella Serie 04 per la ricerca degli zeri di sistemi di equazioni non lineari, può essere convenientemente applicato al problema di ricerca del punto di minimo $\mathbf{x}$ di Φ ponendo

$$\mathbf{d}^{(k)} = -\left(H_\Phi(\mathbf{x}^{(k)})\right)^{-1} \nabla\Phi(\mathbf{x}^{(k)}) \qquad \text{per } k = 0, 1, \ldots,$$

e scegliendo poi il passo $\alpha_k = 1$. Il metodo di Newton (esatto) richiede che $\alpha_k = 1$. Tuttavia, dato che il metodo di Newton converge per $\mathbf{x}^{(0)}$ *sufficientemente* vicino a $\mathbf{x}$, la robustezza del metodo può essere migliorata selezionando α_k tramite il metodo di *backtracking*.

Dato $\mathbf{x}^{(0)}$, la generica iterata k del metodo di Newton si esprime come:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\, \mathbf{d}^{(k)}, \qquad k \geq 0.$$

Si termina l'algoritmo quando è soddisfatto un opportuno criterio di arresto. In questo laboratorio scegliamo di arrestare le iterazioni quando $\widetilde{e}^{(k)} < tol$, dove $\widetilde{e}^{(k)} = \|\nabla\Phi(\mathbf{x}^{(k)})\|_2$.

Se Φ è *sufficientemente* regolare, $\mathbf{x}^{(0)}$ è *sufficientemente* vicino a $\mathbf{x}$ e $\det(H_\Phi(\mathbf{x})) \neq 0$, allora il metodo di Newton (per $\alpha_k = 1$) converge con ordine $p = 2$ (*quadraticamente*), ovvero

$$\lim_{k\to+\infty} \frac{\|\mathbf{x} - \mathbf{x}^{(k+1)}\|}{\|\mathbf{x} - \mathbf{x}^{(k)}\|^2} = \mu.$$

Osserviamo che il metodo è particolarmente efficace, ma converge in generale a punti stazionari di Φ, non necessariamente a punti di minimo.

Metodi quasi-Newton

Quando consideriamo problemi di larga scala, la gestione della matrice Hessiana e soluzione del sistema lineare possono diventare computazionalmente inefficienti. In queste situazioni, sono preferibili i cosiddetti metodi di *quasi-Newton*, tra cui consideriamo il metodo BFGS. Tale metodo costruisce, ad ogni iterazione k, un'approssimazione della matrice inversa dell'Hessiano, ovvero $B^{(k)} \in \mathbb{R}^{n\times n}$ tale per cui $B^{(k)} \simeq \left(H_\Phi(\mathbf{x}^{(k)})\right)^{-1}$. Dato $\mathbf{x}^{(0)}$, la direzione di discesa del metodo BFGS è quindi:

$$\mathbf{d}^{(k)} = -B^{(k)}\,\nabla\Phi(\mathbf{x}^{(k)}) \qquad \text{per } k = 0, 1, \ldots,$$

dove per $k = 0, 1, \ldots$

$$B^{(k+1)} = \left(I - \rho_k\,\boldsymbol{\delta}^{(k)}\,\left(\mathbf{s}^{(k)}\right)^T\right)\,B^{(k)}\,\left(I - \rho_k\,\mathbf{s}^{(k)}\,\left(\boldsymbol{\delta}^{(k)}\right)^T\right) + \rho_k\,\boldsymbol{\delta}^{(k)}\,\left(\boldsymbol{\delta}^{(k)}\right)^T$$

con

$$\boldsymbol{\delta}^{(k)} = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}, \quad \mathbf{s}^{(k)} = \nabla\Phi(\mathbf{x}^{(k+1)}) - \nabla\Phi(\mathbf{x}^{(k)}), \quad \rho_k = \frac{1}{\left(\boldsymbol{\delta}^{(k)}\right)^T\,\mathbf{s}^{(k)}}.$$

Dati $\mathbf{x}^{(0)}$ e $B^{(0)}$ (tipicamente viene scelta $B^{(0)} = I$), la generica iterata k del metodo BFGS si esprime come:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\,\mathbf{d}^{(k)},$$

dove α_k è determinata tramite l'algoritmo di backtracking. Si termina l'algoritmo quando è soddisfatto un opportuno criterio di arresto. Come per il metodo di Newton, in questo laboratorio scegliamo di arrestare le iterazioni quando $\widetilde{e}^{(k)} < tol$, dove $\widetilde{e}^{(k)} = \|\nabla\Phi(\mathbf{x}^{(k)})\|_2$.

Esercizio 2.2

Si consideri la funzione $\Phi : \mathbb{R}^2 \to \mathbb{R}$ dove

$$\Phi(\mathbf{y}) = \Phi(y_1, y_2) = 3\,y_1^2 + y_2^2 - \frac{y_1}{4} - \frac{y_2}{6} + e^{-2\,y_1\,y_2},$$

dotata del punto di minimo $\mathbf{x} = (0.099299729019640, 0.179161952163217)^T$.

1. Si rappresentino le curve di livello di Φ in Matlab® per $y_1 \in [-0.15, 0.25]$ e $y_2 \in [-0.05, 0.35]$ (*hint*: si usino i comandi `meshgrid`, `surf` e `contourf`). Si verifichi che esiste un minimo nel punto $\mathbf{x}$.

2. Si implementi una funzione `newton_opt.m` per il metodo di Newton. La funzione deve avere la seguente intestazione:

```
[xv, it] = newton_opt(GradPhi, HessPhi, x0, toll, nmax ),
```

dove `GradPhi` è il gradiente della funzione obiettivo da minimizzare (definito come anonymous function con due input), `HessPhi` è l'Hessiano della funzione obiettivo da minimizzare (definito come anonymous function con due input), `x0` è la guess iniziale (definita come un vettore), `toll` è la tolleranza (per il criterio di arresto basato sulla stima dell'errore), `nmax` è il numero massimo di iterazioni; in uscita la funzione restituisce la successione `xv` delle approssimazioni del minimo della funzione obiettivo (definita come un matrice sulle cui colonne troviamo le approssimazioni successive) e il numero di iterazioni svolte `it`.

3. Si risolva il problema di approssimazione dello zero $\mathbf{x}$ di $\nabla\Phi$ (punto stazionario di Φ) tramite il metodo di Newton. Si consideri $\mathbf{x}^{(0)} = (-0.14,\, 0.14)^T$, `toll` $= 10^{-8}$ e `nmax` $=$ 200.

4. Si implementi una funzione `bfgs.m` per il metodo BFGS. La funzione deve avere la seguente intestazione:

```
[xv, it] = bfgs(Phi, GradPhi, x0, toll, nmax ),
```

   dove `Phi` è la funzione obiettivo da minimizzare (definita come anonymous function con due input) e gli altri input hanno lo stesso significato riportato al punto 2 di questo esercizio. In uscita la funzione restituisce la successione `xv` delle approssimazioni del punto di minimo della funzione obiettivo (definita come un matrice sulle cui colonne troviamo le approssimazioni successive) e il numero di iterazioni svolte `it`. All'interno della funzione `bfgs.m` si usi la funzione `backtracking.m` (fornita su WeBeep) la cui intestazione è

```
[alphak, it] = backtracking(Phi, GradPhi, xk, dk, c1, rho, nmax )
```

   da applicare a ogni iterazione k-esima dell'algoritmo e che restituisce il valore di step-size α_k. Osserviamo che l'algoritmo BFGS non necessita dell'espressione di Φ, tuttavia questa è rischiesta per la verifica della condizione di Wolfe ($CW1$) nell'algoritmo di backtracking.

5. Si risolva il problema di approssimazione del punto stazionario $\mathbf{x}$ di Φ tramite il metodo BFGS. Si considerino $\mathbf{x}^{(0)} = (-0.14,\, 0.14)^T$, `toll` $= 10^{-8}$ e `nmax` $= 200$. Per il metodo di backtracking si usino $c1 = 10^{-4}$, $\rho = 0.5$ e `nmax` $= 10$.

6. Si riporti in scala semilogaritmica l'andamento degli errori in funzione del numero di iterazioni per il metodo di Newton e per il metodo BFGS. Si commentino i risultati ottenuti.

3 Esercizi addizionali

Esercizio 3.1

Si consideri il problema di ottimizzazione presentato nell'Esercizio 2.1.

1. A partire dall'implementazione di `gradiente_coniguato.m`, scrivere una nuova funzione che permetta di scegliere il metodo per assegnare il parametro di accelerazione (passo) α_k, aggiungendo la possibilità di utilizzare il metodo della sezione aurea oltre al backtracking.

   La nuova funzione avrà la seguente intestazione

   ```
   [xvect, it] = gradiente_coniugato_BT_SA(Phi, GradPhi, flag_metodo, flag_alpha, ...
                   x0, toll, nmax)
   ```

   in cui tra gli input si aggiunge la stringa `flag_alpha` (backtracking `'BT'` o sezione aurea `'SA'`).

   Suggerimento: All'interno della funzione `gradiente_coniugato_BT_SA`, a seconda del caso scelto tramite la variabile `flag_alpha`, si usino le funzioni

   ```
   [xv, k, errv] = sezione_aurea(f, a, b, tol, nmax),
   ```

   per `flag_alpha = "SA"`, oppure

   ```
   [alphak, it] = backtracking(Phi, GradPhi, xk, dk, c1, rho, nmax )
   ```

   per `flag_alpha = "BT"`, da applicare a ogni iterazione k-esima dell'algoritmo e che restituiscono il valore di step-size α_k.

2. Risolvere il problema di minimizzazione del funzionale tramite il metodo del gradiente (`flag_method='G'`) e il metodo del gradiente coniugato con β_k di Fletcher-Reeves (`flag_method='FR'`) utilizzando l'algoritmo della sezione aurea per determinare il parametro α_k e commentare i risultati ottenuti. Si considerino $\mathbf{x}^{(0)} = (0.1, 0.05)^T$, `toll` $= 10^{-5}$ e `nmax` $= 100$. In input alla funzione `sezione_aurea.m` si impostino i parametri `a = 0`, `b = 1`, `tol_sa = 1e-5` e `nmax_sa = 20`.

3. Si riporti in scala semilogaritmica rispetto all'asse y l'andamento degli errori in funzione del numero di iterazioni per i metodi testati.

Esercizio 3.2

Si consideri la funzione obiettivo di Esercizio 2.2 $\Phi : \mathbb{R}^2 \to \mathbb{R}$:

$$\Phi(\mathbf{y}) = \Phi(y_1, y_2) = 3\,y_1^2 + y_2^2 - \frac{y_1}{4} - \frac{y_2}{6} + e^{-2\,y_1\,y_2},$$

dotata del punto di minimo $\mathbf{x} = (0.099299729019640, 0.179161952163217)^T$.

Si risolva il problema di approssimazione del punto di minimo $\mathbf{x}$ tramite i metodi di Newton, BFGS, gradiente e gradiente coniugato considerando: $\mathbf{x}^{(0)} = (-0.14, 0.14)^T$, `toll` $= 10^{-8}$ e `nmax` $= 200$. Per i metodi BFGS, gradiente e gradiente coniugato, si determini il valore di α_k tramite il metodo di *backtracking* usando come parametri $c1 = 10^{-4}$, $\rho = 0.5$ e `nmax`$= 10$. Si stimino gli ordini di convergenza dei quattro metodi al punto di minimo $\mathbf{x}$.