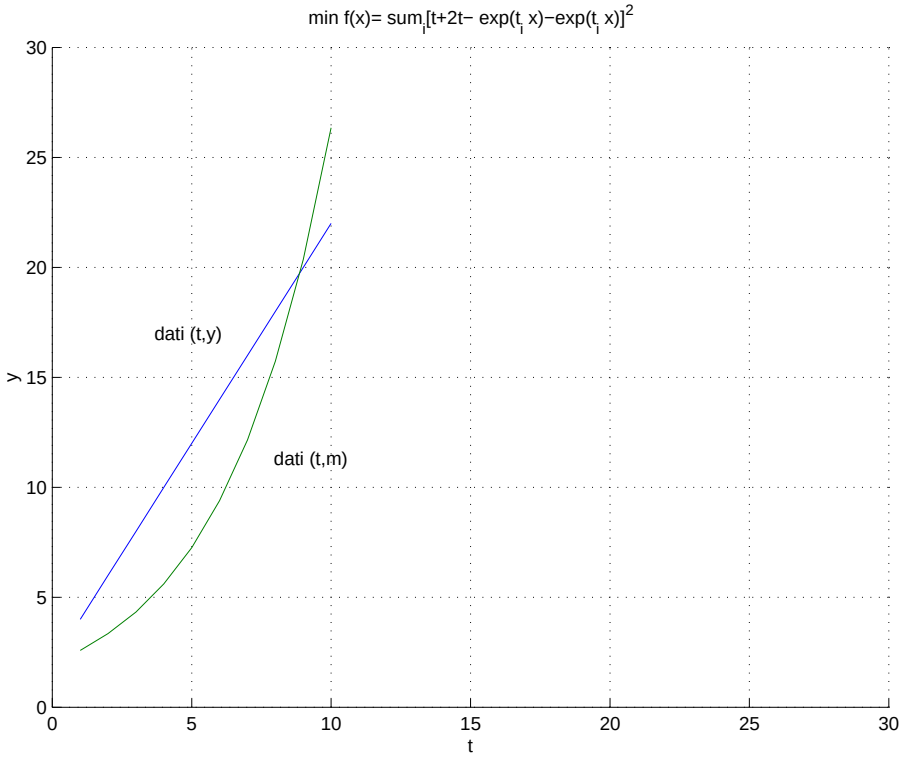
Giovanni Zilli
Luca Bergamaschi **Manolo Venturin**
Dipartimento di Metodi e Modelli Matematici
Università di Padova

METODI DI OTTIMIZZAZIONE



DMMMSA

(EDIZIONE PROVVISORIA)

Prefazione

Questa Dispensa, assieme ai due capitoli contenuti nella dispensa *Metodi Variazionali per Equazioni Differenziali* [44], origina dalle Lezioni di *Metodi Numerici e Calcolo Parallelo* da me tenute al corso di Perfezionamento in *Matematica Applicata e Programmazione* dell'Università di Padova, nell'A.A: 1998/99. Per le nozioni di Calcolo Numerico necessarie si rimanda al Testo [45] citato in Bibliografia. Ispirandosi alle lezioni, devono intendersi come un'edizione provvisoria (e incompleta) di un futuro testo.

Nei due capitoli sono trattati i principali e più usati metodi di Ottimizzazione (detta anche Programmazione Matematica) non vincolata e vincolata, rispettivamente.

Padova, gennaio 1999

Giovanni Zilli

Prefazione

Questa edizione esce con varie correzioni e aggiunte rispetto alla precedente. In particolare, a cura di L. Bergamaschi, sono state aggiunte l'Appendice A e l'Appendice B sui metodi variazionali per sistemi lineari.
La Dispensa è parte integrante del corso di Metodi Numerici (Ingegneria Aerospaziale, Laurea specialistica, IV anno).

Padova, gennaio 2005

Giovanni Zilli Luca Bergamaschi

Prefazione

Questa edizione esce con varie correzioni e aggiunte rispetto alla precedente. In particolare, è stato ampliato il capitolo 2 e aggiunto il capitolo 3. I programmi al capitolo 2 (metodo interior point) sono di M. Venturin.

Padova, novembre 2008

Giovanni Zilli Luca Bergamaschi Manolo Venturin

Indice

Elenco delle figure

Capitolo 1

Ottimizzazione non vincolata

Notazioni

Il problema della *Ottimizzazione non vincolata* si può così formulare:

Determinare un punto di minimo (minimizzatore) $\overline{x}$ tale che

$$f(\overline{x}) = \min\{f(x);\ x \in R^n\}, \quad f : R^n \longrightarrow R\,.$$

Si scrive anche

$$\overline{x} = \underbrace{\arg\min}_{x \in R^n} f(x)\,.$$

Prendendo $-f$ in luogo di f, si ha un problema di massimo, avendosi $\min f = -\max(-f)$.

Se $x \in \Omega \subset R^n$ si ha un problema di estremo *vincolato,* capitolo 2, sezione 2.1. Ω è l'insieme dei vincoli.
Ovviamente, in questo capitolo si ha $\boxed{\Omega = R^n}$: problema di estremo non vincolato (o libero).
f si chiama funzione obiettivo. Si ha:

$$f : R^n \longrightarrow R; \quad \Longrightarrow \quad f' = \nabla f = g \quad \text{(gradiente di } f\text{)}; \quad \Longrightarrow$$
$$f'' = \nabla^2 f = H\ (= G) \quad \text{(matrice hessiana di } f\text{)}; \quad B = \hat{H} \approx H,$$

con $B = \hat{H} \approx H$ (approssimazione della hessiana H).

Precisamente, il vettore gradiente è dato da:

$$\nabla f(x) = \left[\frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x)\right]^T.$$

La matrice hessiana $H(x)$ è la matrice di elementi:

$$\nabla^2 f(x)_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j} \qquad 1 \leq i,\ j \leq n.$$

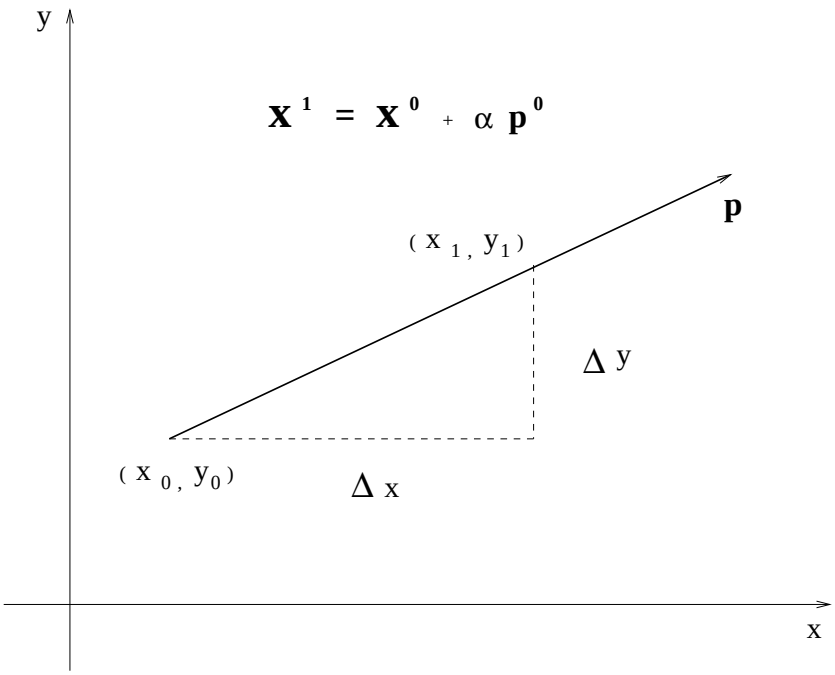


Figura 1.1: Direzione di ricerca p e successione di punti di minimo (in R^2).

Si suppongono noti i risultati sulle *funzioni convesse* f e il loro studio tramite la hessiana[1]. Ad esempio:

$f : \Omega \in R^n \longrightarrow R$ di classe C^2 è *strettamente convessa* nel convesso $\Omega \in R^n$ se la matrice hessiana f'' è > 0, cioè definita positiva (condizione sufficiente).

Attenzione: le due funzioni $f = x^2$, $f = x^4$ sono strettamente convesse in R e $Dx^2 = 2 > 0$, ma $Dx^4 = 12x^2$ è nulla in $x = 0$.
Si ha la seguente caratterizzazione:
$f : \Omega \in R^n \longrightarrow R$ di classe C^2 è *convessa* nel convesso $\Omega \in R^n$ se e solo se la matrice hessiana f'' è ≥ 0, cioè semidefinita positiva.

Vettore *direzione di ricerca* p lungo una linea (in generale $\|p\| \neq 1$):

$$\boxed{x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}, \qquad \alpha^{(k)} > 0, \qquad k \geq 0} \tag{1.1}$$

dove $\alpha^{(k)} \in R^+$ (scalare positivo).
Altre notazioni usate in letteratura (si veda la Figura 1.1):

$$\boxed{x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)} = x^{(k)} + \Delta x^{(k)} = x^{(k)} + \delta^{(k)} = x^{(k)} + s^{(k)}}$$

[1]Cfr. [19, pag. 413] , [13, pag. 476], [27, pag. 176].

$p^{(k)}$ ($\equiv$ $d^{(k)}$), direzione di ricerca k-esima, $\alpha^{(k)}$, lunghezza del passo;

$$s^{(k)} = x^{(k+1)} - x^{(k)} = \alpha^{(k)} p^{(k)}$$
$$\alpha^{(k)} p^{(k)} = s^{(k)} = \Delta x^{(k)} = \delta^{(k)}, \quad \text{passo } k-\text{esimo} \quad (s \equiv \text{ step}).$$

A volte si scriverà, con abuso di scrittura: f^k, $f^{(k)}$, f_k per $f(x^{(k)})$ e simili.

Esempio (di minimizzazione): f funzione quadratica (con $A > 0$, definita positiva)[2]

$$f(x) = \frac{1}{2} x^T A x - b^T x, \ (x, b) \in R^n, \tag{1.2}$$
$$\nabla f(x) = f'(x) = Ax - b; \quad f''(x) = H = A \in R^{n \times n}; \quad A = A^T, \quad A > 0.$$

Per questa funzione, si ha la seguente equivalenza tra i due problemi P_1 e P_2 :

$$P_1: \ F(x) = \nabla f = Ax - b = 0 \quad \Longleftrightarrow \quad P_2: \ \min_{x \in R^n} \{f\},$$

in quanto, come è noto, essendo ogni punto di R^n un punto interno, l'equazione $\nabla f = 0$ caratterizza[3] i *punti critici* della f.
Cioè, i due problemi (si veda [45, cap. 10], [44, cap. 2]):

- P_1: soluzione sistema lineare $F(x) = Ax - b = 0$ e
- P_2: minimizzazione non vincolata del funzionale associato $f(x)$,

sono equivalenti in quanto il minimo esiste ed è unico (e globale). Si vedano gli esempi in sezione 1.4.

Per funzioni generali (sistemi non lineari $F(x) = (f_1, f_2, \ldots, f_n)^T = 0$) questa equivalenza non c'è. Precisamente, i due problemi:

- P_1: soluzione sistema non lineare $F(x) = 0$ e
- P_2: minimizzazione non vincolata della norma-2 (metodo dei minimi quadrati, sezione 1.9)

$$\min\{\|F(x)\|_2^2\} = \min\{F(x)^T F(x)\} = \min\{f_1^2 + \cdots + f_n^2\}$$

[2]Questa funzione (più precisamente è un funzionale) è un buon modello di una funzione qualunque, in quanto, se $f \in C^2(I(\overline{x}))$, la formula di Taylor si scrive

$$f(\overline{x} + s) = f(\overline{x}) + s^T f'(\overline{x}) + \frac{1}{2} s^T f''(\overline{x} + ts) \ s, \quad 0 < t < 1.$$

[3]Attenzione: il punto $x = 0$ è un punto critico, ma non è un punto di minimo, per la $f = x^3$ essendo $f'' = 6x = 0$ per $x = 0$, mentre lo è per la $f = x^2$ in quanto $f'' = 2 > 0$.

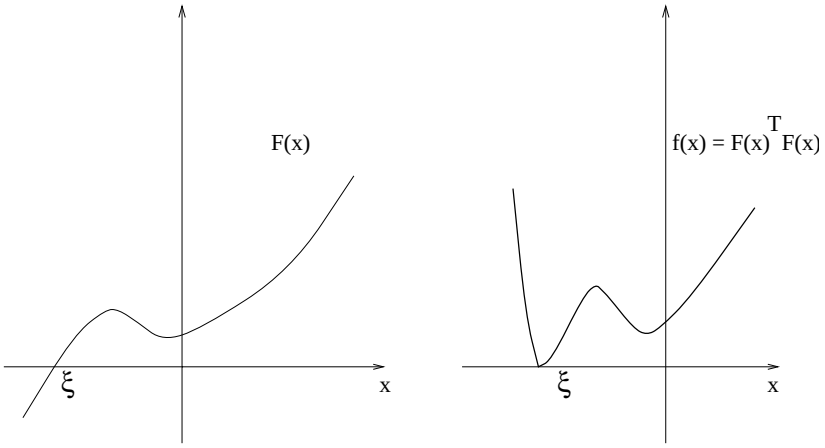


Figura 1.2: Sistema non lineare e corrispondente problema di minimo (in R^1).

non (sempre) sono equivalenti, nel senso che il secondo può introdurre dei minimi locali che non sono soluzioni del sistema; si veda la Figura 1.2, relativa a funzioni in R^1.

I metodi di ottimizzazione (che vedremo) si possono distinguere in:

- *metodi diretti* (senza calcolo di derivate).

Esempio: metodo di Powell (poco usato per $n > 2$ variabili), che qui non trattiamo[4];

- metodi *indiretti* (con calcolo anche di derivate).

Esempi: metodi di tipo gradiente (convergenza del primo ordine) alle sezioni 1.4 e 1.5, metodo di Newton (convergenza del secondo ordine) in sezione 1.6, metodi di tipo secante (di Broyden) o quasi-Newton (a convergenza superlineare) in sezione 1.8.
Ovviamente, se si usano differenze finite per approssimare le derivate, i metodi diretti possono considerarsi metodi diretti.

[4]Per esso si veda [13, pag. 539-541], [19, pag. 468-421], [17, pag. 197-201] dove è risolto un esempio: la classica funzione test "banana" di Rosenbrock (minimizzatore $\overline{x} = (1, 1)^T$):

$$f(x) = f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

In Matlab esiste la routine *fminsearch.m* che usa il metodo diretto detto del simplesso (di Nelder-Mead).

1.1 Metodi variazionali per sistemi lineari

Come *Premessa* ai metodi variazionali di questo capitolo, introduciamo i metodi variazionali (di tipo gradiente) applicati alla funzione quadratica (1.2). Essi conducono alla soluzione dei *sistemi lineari simmetrici* $Ax = b,\ x \in R^n$, con A definita positiva, anche con *precondizionamento*, e più in generale ai metodi di tipo Krilov per sistemi *non simmetrici*.

Si veda l'**Appendice B** di questa Dispensa (e le **Note** di sezioni 1.2 e 1.5).

1.2 Condizione di discesa

È ragionevole assumere che in questi metodi di minimizzazione la direzione di ricerca $p^{(k)}$ in (1.1) sia una direzione di discesa, ossia che si abbia la seguente *condizione di discesa* (descent condition)[5]

$$\boxed{f(x^{(k+1)}) < f(x^{(k)}) \qquad \text{per} \quad k \geq 0\ .} \tag{1.3}$$

Poniamo

$$\varphi(\alpha) = f(x^{(k+1)}) = f(x^{(k)} + \alpha p^{(k)}). \tag{1.4}$$

Applicando la *formula di Taylor* a $f \in C^2(I(x^{(k)}))$, si ha

$$\varphi(\alpha) = f(x^{(k+1)}) = f(x^{(k)}) + (\alpha p^{(k)})^T\ \nabla f(x^{(k)}) + O(\alpha^2) \tag{1.5}$$

e quindi per soddisfare la (1.3) deve aversi

$$\boxed{(p^{(k)})^T\ \nabla f(x^{(k)}) = \langle \nabla f(x^{(k)}), p^{(k)} \rangle < 0} \tag{1.6}$$

per $\boldsymbol{\alpha > 0}$, sufficientemente piccolo.

Al solito, con $\langle x, y \rangle = y^T x$ si indica il prodotto scalare dei due vettori x, y.

In altre parole, la retta $x = x^{(k)} + \alpha p^{(k)}$ deve formare un angolo ottuso con la direzione del gradiente (ossia, derivata direzionale $\nabla f_k^T p_k$ negativa), si veda la Figura 1.3.

[5] Anche se essa da sola **non** garantisce che $x^{(k)}$ converga a un minimizzatore: $\alpha^{(k)}$ deve soddisfare alle due condizioni di Wolfe (1.13) e (1.14). Si vedano i due (contro)esempi in [15, pag. 117-118], in cui o la f non decresce abbastanza rispetto al passo, o i passi sono troppo piccoli rispetto alla diminuzione della f. Come si vedrà in sezione 1.3, al primo inconveniente si può ovviare con la prima condizione di Wolfe (1.13), detta anche regola di Armijo, mentre la seconda condizione (1.14) evita passi troppo piccoli.
Avvertiamo inoltre che esistono algoritmi *non monotoni* in cui si richiede che la condizione di discesa sia soddisfatta non ad ogni passo k, ma dopo un numero $m, m \geq 1$, di passi prefissato, cioè: $f(x^{(k+1)}) < f(x^{(k-m)})$.

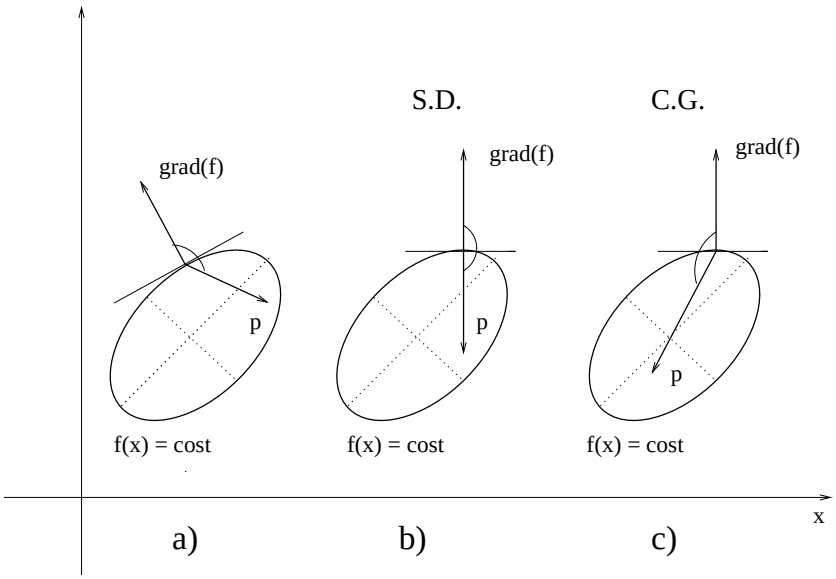


Figura 1.3: Condizione di discesa: a) caso generale, b) S.D., c) C.G.

Anticipiamo i due casi notevoli, Figura 1.3, casi b) e c):

• Steepest descent (S.D.), metodo della discesa più ripida o del gradiente (Cauchy 1847), sezione 1.4:

$$\boxed{p^{(k)} = -\nabla f(x^{(k)})} \tag{1.7}$$

È noto che lungo la direzione del (-) gradiente (che è ortogonale[6] alle curve di livello $f = c$) si ha massima riduzione della f. Questo metodo ha convergenza lineare, ma spesso molto lenta (in pratica 'infinite' iterazioni), con tipico andamento a zig zag, si controlli la formula (1.10) (e la Figura 1.4).

• Gradiente coniugato (C.G.), sezione 1.5, generalizzazione, nel caso di minimizzazione di una f qualunque, del ben noto algoritmo nel caso di f quadratica usato per risolvere i sistemi lineari (Hestenes e Stiefel, 1952), sezione 1.1.

In generale, in tutti i metodi di *ricerca monodimensionale* detti anche metodi di minimizzazione lungo una linea[7]) sussiste una condizione di *ortogonalità* fra due gradienti consecutivi. Nel metodo del gradiente S.D. si ha

[6]Sia $f(x, y) = c, \implies df = f_x dx + f_y dy = \langle \nabla f, dP \rangle = 0$.

[7]Questi metodi consistono dei seguenti passi:
- determinare una direzione di ricerca $p^{(k)}$
- trovare $\alpha^{(k)}$ che minimizzi la $f(x^{(k)} + \alpha^{(k)} p^{(k)})$ rispetto ad $\alpha = \alpha^{(k)}$ (sezione 1.3)
- porre $x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$.

la I-ortogonalita:

$$\boxed{\langle\ \nabla f(x^{(k+1)}), \nabla f(x^{(k)})\ \rangle = 0} \tag{1.8}$$

Infatti: considerando la funzione composta $\varphi(\alpha) \equiv \varphi(\alpha^{(k)})$ in (1.4) e *derivando rispetto ad* $\alpha^{(k)}$, tramite la (1.7), si ha subito (condizione necessaria di minimo)

$$\varphi'(\alpha) = \nabla f_{k+1}^T\ p_k = -\nabla f_{k+1}^T\ \nabla f_k = 0,$$

ossia, scritto più precisamente

$$\begin{aligned} 0 = \varphi'(\alpha^{(k)}) &= \langle\ f'(x^{(k)} + \alpha^{(k)} p^{(k)}), p^{(k)}\ \rangle = -\langle\ \nabla f(x^{(k+1)}), \nabla f(x^{(k)})\ \rangle \\ &= -\nabla^T f(x^{(k+1)})\ \nabla f(x^{(k)}). \end{aligned}$$

Tramite la condizione di ortogonalità (1.8) si può determinare il valore dello scalare $\alpha^{(k)}$, ma come si è detto, esso si calcola, in maniera inesatta, con una tecnica di *ricerca monodimensionale* (line search, **sezione 1.3**).

Se la f è quadratica (formula (1.2), caso dei sistemi lineari) il calcolo è esplicito, e come visto in sezione 1.1 e formula (B.3), si trova:

$$\alpha^{(k)} = \frac{\langle r^k, p^k\ \rangle}{\langle Ap^k, p^k\ \rangle} = \frac{\langle p^k, p^k\ \rangle}{\langle Ap^k, p^k\ \rangle} = \frac{\langle r^k, r^k\ \rangle}{\langle Ar^k, r^k\ \rangle} = \frac{(r^k)^T r^k}{(r^k)^T A r^k}, \tag{1.9}$$

dove: $p^k = -\nabla f^k = r^k = b - Ax^k$ (residuo k-esimo). Essendo

$$\langle r^k, p^k\ \rangle = -\langle \nabla f^k, p^{(k)} \rangle,$$

dalla (1.6) segue che $\alpha^{(k)} > 0$.

NOTA

Rivediamo il calcolo (come) fatto in sezione 1.1. Detta h la soluzione vera, definiamo l'errore assoluto come $e = h - x$, il residuo come $r = b - Ax$ e sia $\Phi(x)$la funzione quadratica (errore) da minimizzare:

$$\Phi(e(x)) = \frac{1}{2} e^T A e = \frac{1}{2} r^T A^{-1} r = \frac{1}{2}\|e\|_{2,A}^2 = \frac{1}{2}\|r\|_{2,A}^2\ .$$

Si noti che, presa la funzione quadratica $f(x) = \frac{1}{2}x^T Ax - b^T x$, risulta:

$$\nabla f(x) = \nabla\Phi(e(x)) = Ax - b = -r, \quad \nabla^2 f(x) = \nabla^2\Phi(e(x)) = A.$$

In generale la minimizzazione rispetto ad α è *inesatta*. Metodi di minimizzazione diversi corrispondono a scelte diverse di $p^{(k)}$.

Si ha:

$$
\begin{aligned}
&2\Phi(e(x^{(k+1)})) = \\
&= (e^{(k+1)})^T A\, e^{(k+1)} = \langle\, A(h - x^{(k+1)}), h - x^{(k+1)} \,\rangle \\
&= \langle\, A(h - x^{(k)}) - A\alpha^{(k)}p^k, (h - x^{(k)}) - \alpha^{(k)}p^k \,\rangle \\
&= \langle\, A(h - x^{(k)}), h - x^{(k)} \,\rangle + \langle\, A(h - x^{(k)}), -\alpha^{(k)}p^{(k)} \,\rangle \\
&+ \langle\, -A\alpha^{(k)}p^{(k)}, h - x^{(k)} \,\rangle + \langle\, -A\alpha^{(k)}p^{(k)}, -\alpha^{(k)}p^{(k)} \,\rangle \\
&= \langle\, Ap^{(k)}, p^{(k)} \,\rangle(\alpha^{(k)})^2 - 2\langle\, A(h - x^{(k)}), p^{(k)} \,\rangle\alpha^{(k)} + \langle\, A(h - x^{(k)}), h - x^{(k)} \,\rangle\ .
\end{aligned}
$$

Possiamo trovare il coefficiente $\alpha^{(k)}$ imponendo la condizione di minimo (gradiente di $\Phi = 0$):

$$\Phi' = \nabla_{\alpha^{(k)}}\Phi = \langle\, Ap^{(k)}, p^{(k)} \,\rangle\alpha^{(k)} - \langle\, A(h - x^{(k)}), p^{(k)} \,\rangle = 0,$$

da cui

$$\alpha^{(k)} = \frac{\langle Ae^k, p^k \,\rangle}{\langle Ap^k, p^k \,\rangle} = \frac{\langle r^k, p^k \,\rangle}{\langle Ap^k, p^k \,\rangle} = \frac{\langle r^k, r^k \,\rangle}{\langle Ar^k, r^k \,\rangle} = \frac{(r^k)^T r^k}{(r^k)^T Ar^k}\ .$$

c.v.d.

(Riverifichiamo che) presa $\alpha^{(k)}$ data dalla (1.9), si ha la condizione di I-ortogonalità (1.8)

$$\langle\, r^{(k+1)}, r^{(k)} \,\rangle = 0$$

Infatti:

$$
\begin{aligned}
r^{(k+1)} &= b - Ax^{(k+1)} \\
&= b - A(x^{(k)} + \alpha^{(k)}p^{(k)}) = b - Ax^{(k)} - \alpha^{(k)}Ap^{(k)} = r^{(k)} - \alpha^{(k)}Ap^{(k)}.
\end{aligned}
$$

Quindi, si avrà che

$$\langle\, r^{(k+1)}, r^{(k)} \,\rangle = \langle\, r^{(k)} - \alpha^{(k)}Ap^{(k)}, r^{(k)} \,\rangle = \langle\, r^{(k)}, r^{(k)} \,\rangle - \alpha^{(k)}\langle\, Ap^{(k)}, r^{(k)} \,\rangle = 0,$$

a patto che sia

$$\alpha^{(k)} = \frac{\langle r^k, r^k \,\rangle}{\langle Ap^k, r^k \,\rangle} = \frac{\langle r^k, p^k \,\rangle}{\langle Ap^k, p^k \,\rangle}\ .$$

c.v.d.

Come visto in sezione 1.1, nel caso dei *sistemi lineari* $Ax = b$, l'algoritmo del gradiente si scrive:

Algoritmo del gradiente (S.D.) $(Ax = b,\ A = A^T > 0)$

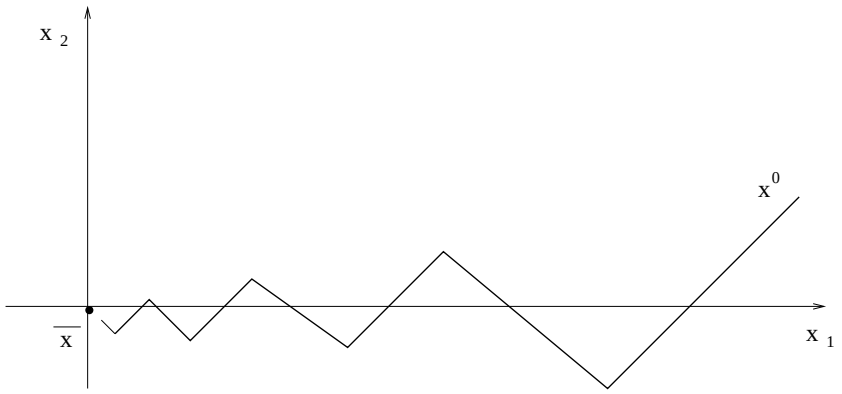


Figura 1.4: S.D. in un caso malcondizionato (in R^2).

In Input: $x^{(0)}, A, b, k_{\max}$, Toll

- **1.** Per $k = 0, 1, \ldots$ fino a convergenza
- $r^{(k)} = (\ p^{(k)} = -\nabla f^{(k)}) = b - Ax^{(k)}$
- $\alpha^{(k)} = \dfrac{\langle r^{(k)}, r^{(k)} \rangle}{\langle Ar^{(k)}, r^{(k)} \rangle}$
- $x^{(k+1)} = x^{(k)} + \alpha^{(k)} r^{(k)}$
- se $\|x^{(k+1)} - x^{(k)}\| = \|\alpha^{(k)} r^{(k)}\| \leq$ Toll, allora Stop
- altrimenti, vai a 1 (fino a $k \leq k_{\max}$).

Il residuo al passo k può essere calcolato in funzione del residuo al passo $k-1$ (cfr. B.2).

A causa della condizione di ortogonalità (1.8), si ha il tipico comportamento a zig-zag del S.D., Figura 1.4.

Lo S.D. converge linearmente, $\forall\ x^{(0)} \in R^n$, all'unico punto di minimo minimo $\overline{x}$. Si dimostra infatti[8], posto $\|x\|_{H,2} = \sqrt{x^T H x}$, che:

$$\|\overline{x}\ - x^{(k)}\|_{H,2} = \|e^{(k)}\|_{H,2} \leq \left(\frac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n}\right)^k \|e^{(0)}\|_{H,2} \tag{1.10}$$

dove $\overline{x}$ è il minimizzatore e $\lambda_1,\ \lambda_n$ sono il massimo e il minimo autovalore della matrice hessiana $A = H(\overline{x}) = \nabla^2 f(\overline{x})$, calcolata dunque nel punto di

[8]Si veda [8], [27, pag.220], [19] e [15, pag. 115]. La (1.10) vale anche nel caso generale (sezione 1.4) nell'ipotesi che il metodo sia convergente e che i due autovalori estremi della matrice hessiana H, calcolata nel punto di minimo, $\lambda_1,\ \lambda_n$ siano positivi.

minimo.
Se $\lambda_1 \gg \lambda_n$ la matrice A è malcondizionata e la convergenza può essere troppo lenta, si veda la Figura 1.4. Si controllino gli esempi in sezione 1.4.

Nella (1.10), $M = \dfrac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n}$ è la costante asintotica dell'errore, e il numero $R = -\log M$ è la velocità (asintotica) di convergenza (cfr. [45, cap.2]).
Fissata una tolleranza ϵ $(= 10^{-r})$, si ha

$$\frac{\|e^{(k)}\|_{H,2}}{\|e^{(0)}\|_{H,2}} \leq M^k \leq \epsilon \implies k \geq \frac{\log \epsilon}{\log M} = \frac{-r}{\log M},$$

cioè k fornisce una stima del numero di iterazioni da effettuarsi per avere una riduzione, in norma H, dell'errore iniziale $e^{(0)}$ di una quantità fissata $\epsilon = 10^{-r}$.

Ricordiamo infine che (cfr. [45, sezione 3.3.2]) il metodo iterativo

$$x^{(k+1)} = x^{(k)} + Cr^{(k)}, \quad C = I, \text{ matrice identità},$$

è il metodo di Richardson, caso particolare di quello di Jacobi ($C = D^{-1}$). Dunque, il metodo S.D. coincide con il metodo di Richardson *non* stazionario, ove si ponga $C^{(k)} = \alpha^{(k)} I$.

(Fine NOTA).

1.3 Minimizzazione monodimensionale (linesearch)

Come accennato, nei metodi che stiamo studiando occorre risolvere, a ogni passo k, un problema (detto anch'esso) di minimizzazione monodimensionale lungo la direzione $p^{(k)}$ (line search). I metodi di minimizzazione monodimensionali (in una variabile) più noti sono i metodi (diretti, ossia senza calcolo di derivate) della *bisezione sequenziale,* di *Fibonacci* e della *sezione aurea,* o (con calcolo di derivate) metodi di *interpolazione polinomiale* (quadratica, cubica,...).
Per questi ultimi **si veda la Tesina [40]**.

Essi possono essere molto sofisticati, essendo necessariamente metodi *inesatti* di ricerca lungo una linea. Inoltre (come già avvertito alla nota n.ro 5, a piè di pagina), per garantire la convergenza (che sia *anche globale*) è necessario avere una *sufficiente diminuzione* della f lungo la linea di ricerca.

Nei metodi[9] in esame, questa si ottiene se il parametro α che dà la lunghezza del passo soddisfa le *condizioni di Wolfe* (1969, 1971) con le quali si può soddisfare anche alla *regola di Armijo* (1966)[10].

Riprendiamo la (1.5):

$$\begin{aligned}\varphi(\alpha) = f(x^{(k)} + \alpha p^{(k)}) &= f(x^{(k)}) + \alpha\ (p^{(k)})^T\ \nabla f(x^{(k)}) &+ O(\alpha^2)\\ &= \varphi(0) \quad + \quad \alpha\ \varphi'(0) &+ O(\alpha^2).\end{aligned}$$

Secondo la *regola di Armijo,* α è accettabile se

$$\varphi(\alpha)\ \leq \underbrace{\varphi(0) + \lambda\ \alpha\ \varphi'(0)}_{l(\alpha)}, \qquad 0 < \lambda < 1 \tag{1.11}$$

Cioè

$$f(x^{(k+1)})\ \leq\ f(x^{(k)}) + \lambda\ \alpha^{(k)}\ (p^{(k)})^T\ \nabla f(x^{(k)}) \tag{1.12}$$

Si noti che $\alpha^{(k)}\ p^{(k)} = x^{(k+1)} - x^{(k)}$.

Nelle implementazioni, si sceglie λ piccolo, in genere $\lambda = 10^{-4}$.
Il valore iniziale di α viene ridotto fino a soddisfare la regola di Armijo.

Se la derivata $\varphi'(\alpha)$ si può calcolare facilmente, α è accettabile se soddisfa alle *due condizioni di Wolfe* (la prima coincide con la regola di Armijo (1.11))

$$\varphi(\alpha) \leq \varphi(0) + \lambda\ \alpha\ \varphi'(0), \tag{1.13}$$

$$\varphi'(\alpha) \geq \beta\ \varphi'(0), \tag{1.14}$$

$$0 < \lambda < \beta < 1 \tag{1.15}$$

Ad esempio $\lambda = 0.1,\ \beta = 0.5$.

La (1.14) si scrive anche:

$$(p^{(k)})^T\ \nabla f(x^{(k+1)}) \geq \beta\ p^{(k)})^T\ \nabla f(x^{(k)})\ .$$

La condizione (1.15) si impone perchè le precedenti due valgano contemporaneamente, e i più piccoli valori di α che soddisfano contemporaneamente la (1.13) e la (1.14) determinano l'estremo sinistro a e l'estremo destro b, rispettivamente, dell'intervallo di accettabilità $[a, b]$ di α, come è illustrato nelle Figure 1.5, 1.6 e 1.7 (dove $c_2 = \beta$).

[9]Esiste un'altra strategia (trust region) di convergenza globale, cfr. alla Nota 4 di sezione 1.6.

[10]Si veda [15, cap. 6], [18, formule (1.11) e (1.12)], [27, pag. 212].

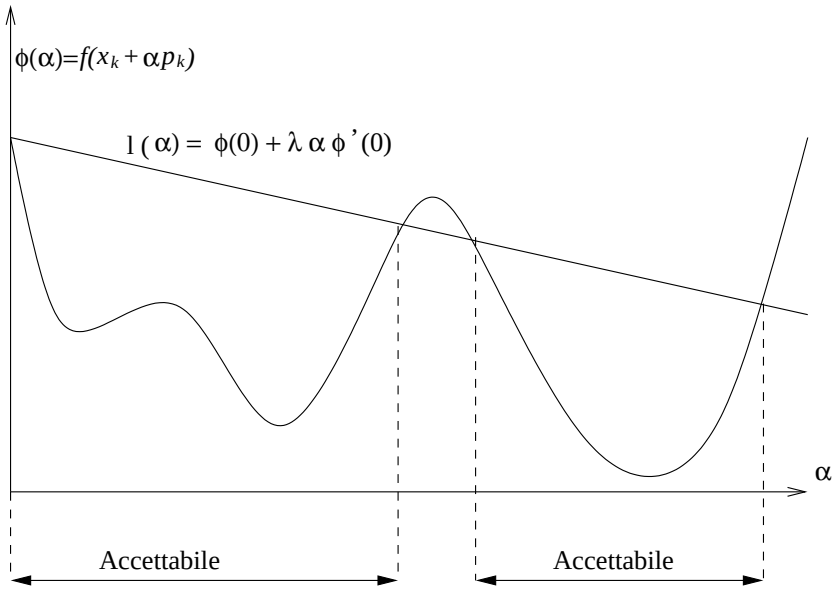


Figura 1.5: Prima condizione di Wolfe (Armijo).

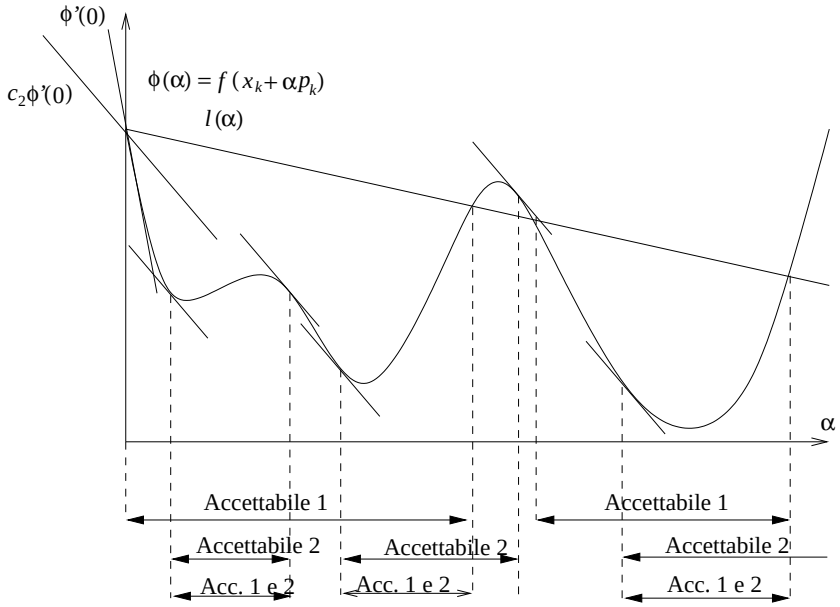


Figura 1.6: Prima e seconda condizione di Wolfe.

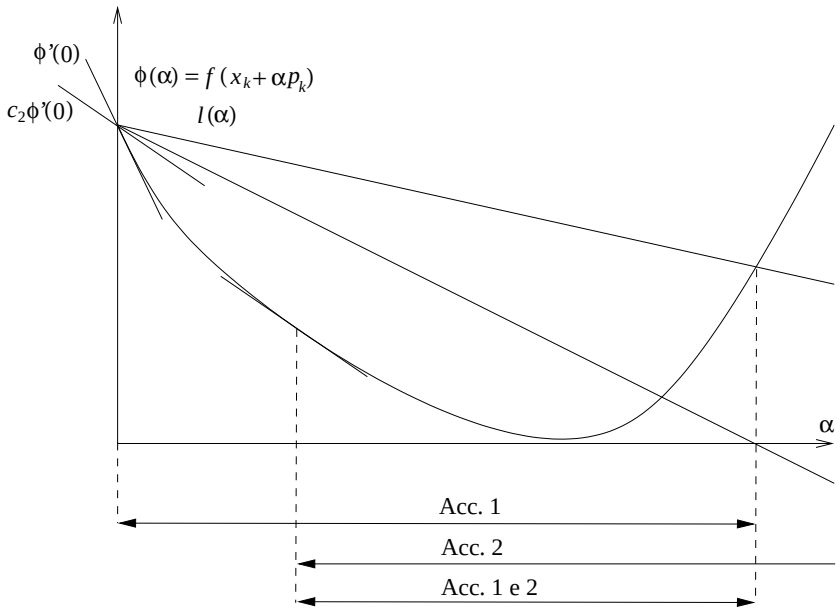


Figura 1.7: Prima e seconda condizione di Wolfe nel caso di una funzione convessa.

Nelle implementazioni, si utilizza (solo) la condizione di sufficiente decrescenza di Armijo (1.12), associata ad una tecnica di *"backtraking"* che riduce α[11]. Per esempio, con una riduzione costante $\alpha^{(j)} = 2^{-j}, j \geq 0$.

ESEMPI (linesarch con regola di Armijo e riduzione del passo)

1.

$$f(x_1, x_2) = x_1^4 + x_1^2 + x_2^2; \quad \Longrightarrow \quad \nabla f = (4x_1^3 + 2x_1, 2x_2)^T.$$

Sia:

$$x^0 = (1, 1)^T; \quad p^{(0)} = (-3, \ -1)^T; \quad \boxed{\lambda = 0.1};$$

da cui

$$f(x^0) = 3; \quad \nabla f(x^0) = (6, \ 2)^T; \quad (p^{(0)})^T \ \nabla f(x^{(0)}) = -20 < 0.$$

Scegliamo $\boxed{\alpha^{(0)} = 2^0 = 1}$. Si ottiene:

$$x^{(1)} = x^{(0)} + \alpha^{(0)} p^{(0)} = (-2, \ 0)^T \quad \Longrightarrow \quad f(x^{(1)}) = 20,$$

[11]Cfr. [15, pag. 126], [34, pag. 41-42], [35, pag. 377]. Il lettore, oltre che implementare la regola di Armijo descritta per trovare α, può anche utilizzare la routine matlab di linesearch polinomiale quadratica-cubica *polymod.m* del software in [25] .

mentre la regola di Armijo fornisce

$$f(x^{(0)}) + \lambda\ \alpha^{(0)}\ (p^{(0)})^T\ \nabla f(x^{(0)}) = 3 + 0.1(-3,\ -1)\begin{pmatrix} 6 \\ 2 \end{pmatrix} = 1\ .$$

Essendo $f(x^{(1)}) = 20 > 1$, $\alpha^{(0)} = 1$ *non è* accettabile.

Proviamo $\alpha^{(0)} = 2^{-1} = 0.5$. Si ottiene:

$$x^{(1)} = x^{(0)} + \alpha^{(0)} p^{(0)} = (-0.5,\ 0.5)^T \Longrightarrow f(x^{(1)}) = 9/16 = 0.5625,$$

mentre la regola di Armijo fornisce

$$f(x^{(0)}) + \lambda\ \alpha^{(0)}\ (p^{(0)})^T\ \nabla f(x^{(0)}) = 3 + 0.1 \times 0.5(-3,\ -1)\begin{pmatrix} 6 \\ 2 \end{pmatrix} = 2\ .$$

Essendo $f(x^{(1)}) = 0.5625 < 2$, $\alpha^{(0)} = 0.5$ è accettabile.

Verifichiamo che per $\alpha^{(0)} = 0.5$, anche la seconda condizione di Wolfe è soddisfatta, per ogni $\beta > \alpha$:

$$(p^{(0)})^T\ \nabla f(x^{(1)}) =\ (-3,\ -1)\begin{pmatrix} -1.5 \\ 1 \end{pmatrix} = 3.5 > \beta\ (p^{(0)})^T\ \nabla f(x^{(0)}) = -20\beta$$

Si può inoltre verificare che, preso $\beta = 0.5$, anche per $\alpha^{(0)} = 1$ la seconda condizione di Wolfe è soddisfatta:

$$(p^{(0)})^T\ \nabla f(x^{(1)}) =\ (-3,\ -1)\begin{pmatrix} -36 \\ 0 \end{pmatrix} = 108 > \beta\ (p^{(0)})^T\ \nabla f(x^{(0)}) = -10.$$

Per *esercizio* si provi che per $\alpha^{(0)} = 0.1$ e $\beta = 0.5$, la condizione di Armijo è soddisfatta, ma non lo è la seconda condizione di Wolfe.

2. (Vedi Esempi N. 7 di S.D. e N. 7, 8 di Newton)

1.4 Steepest Descent (S.D. o metodo del gradiente)

A parte il calcolo di $\alpha^{(k)}$, l'algoritmo del gradiente per funzioni generiche è analogo all'algoritmo visto in sezione 1.1 (per funzioni quadratiche e quindi) per i sistemi lineari simmetrici (come anche ricordato alla **Nota** di sezione 1.2):

Algoritmo S.D (caso generale)

In Input: $x^{(0)}, f, \nabla f, k_{\max}$, Toll

- **1.** Per $k = 0, 1, \ldots$ fino a convergenza
- $p^{(k)} = -\nabla f(x^{(k)})$
- **calcolare** $\alpha^{(k)} : f(x^{(k)} + \alpha^{(k)} p^{(k)}) = \min_{\alpha>0}\{f(x^{(k)} + \alpha p^{(k)})\}$ (con *linesearch*, vedi sezione 1.3)
- $x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$.
- se $\|x^{(k+1)} - x^{(k)}\| = \|\alpha^{(k)} p^{(k)}\| \leq$ Toll, allora Stop
- altrimenti, vai a 1 (fino a $k \leq k_{\max}$).

Altri criteri di arresto possono essere usati, quale: $|f^{k+1} - f^k| \leq$ Toll.

Ovviamente, nel caso dei sistemi lineari $Ax = b$ già descritto, il valore di α^k è dato dalla (1.9), cioè :

$$\alpha^k = \frac{\langle r^k, r^k \rangle}{\langle Ar^k, r^k \rangle} .$$

ESEMPI[12] (da risolvere con S.D. $p^{(k)} = r^{(k)} = -\nabla f(x^{(k)})$)

1. (funzione quadratica: somma di quadrati; hessiana diagonale)

$$f(x_1, x_2) = x_1^2 + x_2^2 = \frac{1}{2} x^T H x - b^T x, \quad x^0 = (1, 2)^T;$$

$$\nabla f = (2x_1, 2x_2)^T, \; H = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}, \; b = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Il minimizzatore[13] è (banalmente) $\overline{x} = (0, 0)^T$, $f(\overline{x}) = 0$ (minimo). Infatti:

$$\nabla f(1, 2) = (2, 4)^T \qquad \Longrightarrow \qquad r^{(0)} = -\nabla f(1, 2) = -(2, 4)^T$$

$$x^{(1)} = x^{(0)} + \alpha^{(0)} r^{(0)} = x^{(0)} - \alpha^{(0)} \nabla f(x^{(0)}) = (1 - 2\alpha^{(0)}, 2 - 4\alpha^{(0)})^T$$

$$f(x^{(1)}) = \varphi(\alpha^{(0)}) = (1 - 2\alpha)^2 + (2 - 4\alpha)^2, \qquad (\alpha^{(0)} = \alpha)$$

[12] Si consiglia di risolvere gli esempi di questa e delle altre sezioni con **tutti** i metodi.

[13] In questo e negli altri esempi, si trova subito risolvendo analiticamente il sistema $\nabla f(x) = 0$.

$$\varphi(\alpha) = 5 - 20\alpha + 20\alpha^2, \quad \varphi'(\alpha) = -20 + 40\alpha = 0 \quad \Longrightarrow \quad \alpha = \frac{20}{40} = \frac{1}{2}.$$

Dunque:

$$x^{(1)} = (1,2)^T - \frac{1}{2}(2,4)^T = (0,0)^T$$

Osservazioni.

a. In questo semplice esempio, il valore di α si è ottenuto derivando la $\varphi(\alpha)$ e non ricorrendo a una line search inesatta.

b. Ovviamente, per raggiungere il minimo basta 1 sola iterazione, essendo f = *cost* una famiglia di cerchi concentrici (fare la figura!). La velocità di convergenza è massima, essendo la costante asintotica dell'errore $M = \dfrac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n} = \dfrac{2-2}{2+2} = 0$.
Inoltre, essendo la f una funzione quadratica, α^0 si può calcolare teoricamente[14] con la formula (1.9):

$$\alpha^0 = \frac{\langle p^0, p^0 \rangle}{\langle Hp^0, p^0 \rangle} = \frac{{r^0}^T r^0}{{r^0}^T H r^0} = \frac{2^2 + 4^2}{(-2,-4)\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}\begin{pmatrix} -2 \\ -4 \end{pmatrix}} = \frac{1}{2},$$

essendo: $p^0 = r^0 = b - Hx^0 = -\nabla f^0 = (-2,-4)^T$ (residuo).

c. S.D. (come C.G. di sezione 1.5) è "sensibile" ai cambi di variabile (cioè non è un metodo "scale-invariant". Al contrario, "scale-invariant" sono i metodi Newton di sezione 1.6 e quasi-Newton di sezione 1.8). Riprendiamo l'esempio e operiamo il seguente cambio di variabili:

$$y_1 = x_1; \qquad y_2 = \frac{x_2}{2}$$

Il problema trasformato è (fare la figura!)

$$f(y_1, y_2) = y_1^2 + 4y_2^2 = (y_1)^2 + (2y_2)^2, \quad y^0 = (1,1)^T.$$

$$\nabla f = (2y_1, 8y_2)^T, \; H = \begin{pmatrix} 2 & 0 \\ 0 & 8 \end{pmatrix}, \; M = 0.6$$

Essendo: $p^0 = r^0 = -\nabla f^0 = (-2,-8)^T$, si ha

$$\alpha^0 = \frac{{r^0}^T r^0}{{r^0}^T H r^0} = \frac{2^2 + 8^2}{(-2,-8)\begin{pmatrix} 2 & 0 \\ 0 & 8 \end{pmatrix}\begin{pmatrix} -2 \\ -8 \end{pmatrix}} = \frac{68}{520} = 0.1308,$$

[14] Cioè risolvendo il sistema lineare $Hx = b$, $b = (0,0)^T$. Cfr. la sezioni 1.1 e B.2. Per esercizio, risolvere i sistemi proposti anche con valori di $b \neq (0,0)^T$.

$$y^{(1)} = y^{(0)} + \alpha^{(0)} r^{(0)} = [1 - 2\alpha^0, 1 - 8\alpha^0]^T = [0.7385, -0.04615] \neq [0, 0]^T.$$

Nota. Il cambio di variabile si puo' utilizzare al "contrario", per accelerare la convergenza del metodo di minimizzazione. Questa tecnica è nota come *precondizionamento*, ed è particolarmente utilizzata nel metodo del gradiente coniugato (cfr. la sezione B.4).

2. (funzione quadratica: somma di quadrati; hessiana diagonale)

$$f(x_1, x_2) = \frac{1}{2}x_1^2 + \frac{9}{2}x_2^2,\ x^0 = (9, 1)^T;\ \nabla f = (x_1, 9x_2)^T,\ H = \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix},\ b = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Il minimizzatore è (banalmente) $\overline{x} = (0, 0)^T$, $f(\overline{x}) = 0$ (minimo).

La convergenza è lenta (a zig zag), essendo la costante asintotica $M = \dfrac{9-1}{9+1} = 0.8 \sim 1$ e la velocità di convergenza $R = -\log M = 0.097$. Per esempio, occorrono circa $k = \dfrac{\log 10^{-6}}{\log M} = 62$ iterazioni per avere: $\dfrac{\|e^{(k)}\|_{H,2}}{\|e^{(0)}\|_{H,2}} < 10^{-6}$.

Procedendo come nell'esercizio precedente si ha:

$$\nabla f(9, 1) = (9, 9)^T = -p^{(0)} = -r^{(0)}$$

$$\alpha^{(0)} = \frac{r^{0T} r^0}{r^{0T} H r^0} = \frac{(-9, -9)\begin{pmatrix} -9 \\ -9 \end{pmatrix}}{(-9, -9)\begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix}\begin{pmatrix} -9 \\ -9 \end{pmatrix}} = \frac{1}{5} = 0.2$$

$$x^{(1)} = x^{(0)} + \alpha^{(0)} r^{(0)}$$

$$= (9, 1)^T + \frac{1}{5}(-9, -9)^T = (9 \times 0.8, -0.8)^T = (7.2, -0.8)^T.$$

Calcoliamo la seconda iterata $x^{(2)}$.

$r^{(1)} = -\nabla f(x^{(1)}) = (-7.2,\ 7.2)^T$

(Equivalentemente, si può usare la formula iterativa $r^{(1)} = r^{(0)} - \alpha^{(0)} H r^{(0)}$)

$$\alpha^{(1)} = \frac{r^{1T} r^1}{r^{1T} H p^1} = \frac{103.68}{(-7.2,\ 7.2)\begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix}\begin{pmatrix} -7.2 \\ 7.2 \end{pmatrix}} = \frac{1}{5} = 0.2$$

$$x^{(2)} = x^{(1)} + \alpha^{(1)} r^{(1)}$$

$$= (7.2, -0.8)^T + \frac{1}{5}(-7.2, 7.2)^T = (9 \times 0.8^2, (-0.8)^2)^T = (5.76, 0.64)^T.$$

Ecc... Si trova: $x^{(k)} = \left(9 \times 0.8^k \ , \ (-0.8)^k\right)^T, k \geq 1,$ e quindi $\lim_{k\to\infty} x^{(k)} = 0.$

3. (funzione quadratica)

$$f(x) = f(x_1, x_2) = 2x_1^2 + x_2^2 - 3 = \frac{1}{2}x^T Hx - b^T x + c, \quad x^0 = (1,1)^T,$$
$$\nabla f = (4x_1, 2x_2)^T, \quad H = \begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix}, \quad b = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad c = -3$$

Il minimizzatore è (banalmente) $\overline{x} = (0,0)^T$, $f(\overline{x}) = -3$ (minimo).
Si osservi che: $f + 3 = (\sqrt{2}x_1)^2 + (x_2)^2$.
La costante asintotica è $M = \dfrac{4-2}{4+2} = 1/3 = 0.3333$ e quindi la velocità di convergenza è $R = -\log_{10} M \approx 0.5$, e $1/R = 2$. Per esempio, occorrono circa $k = \dfrac{\log 10^{-6}}{\log M} = 13$ iterazioni per avere: $\dfrac{\|e^{(k)}\|_{H,2}}{\|e^{(0)}\|_{H,2}} < 10^{-6}$.
Procedendo come nell'esercizio precedente si ha:

$$\nabla f(1,1) = (4,2)^T = -p^{(0)} = -r^{(0)}$$
$$\alpha^{(0)} = \frac{r^{0T} r^0}{r^{0T} H r^0} = \frac{4^2 + 2^2}{(-4,-2)\begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix}\begin{pmatrix} -4 \\ -2 \end{pmatrix}} = \frac{20}{72} = \frac{5}{18} = 0.27778$$
$$x^{(1)} = x^{(0)} + \alpha^{(0)} r^{(0)}$$
$$= (1,1)^T + \frac{5}{18}(-4,-2)^T = (-1/9, 4/9)^T = (-0.1111, 0.4444)^T.$$

Calcoliamo la seconda iterata $x^{(2)}$.

$$r^{(1)} = -\nabla f(x^{(1)}) = (4/9, \ -8/9)^T = (0.4444, \ -0.8889)^T$$
(Equivalentemente, si può usare la formula iterativa $r^{(1)} = r^{(0)} - \alpha^{(0)} H r^{(0)}$)
$$\alpha^{(1)} = \frac{r^{1T} r^1}{r^{1T} H r^1} = \frac{80/81}{(4/9, \ -8/9)\begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix}\begin{pmatrix} 4/9 \\ -8/9 \end{pmatrix}} = \frac{5}{12} = 0.41667$$
$$x^{(2)} = x^{(1)} + \alpha^{(1)} r^{(1)}$$
$$= \left(\frac{-1}{9} + \frac{5}{12}\,\frac{4}{9} \ , \ \frac{4}{9} + \frac{5}{12}\,\frac{-8}{9}\right)^T = \left(\frac{2}{27} \ , \frac{2}{27}\right)^T = (0.074074, \ 0.074074)^T.$$

Ecc...

4. (funzione quadratica, con hessiana definita positiva)

$$f(x_1,x_2,x_3)=\frac{1}{2}x^THx-b^Tx,\ x^0=(0,0,0)^T,\quad H=\begin{pmatrix}2&1&1\\1&2&1\\1&1&2\end{pmatrix},\ b=\begin{pmatrix}1\\2\\3\end{pmatrix}.$$

Si trova: eig(H) =$\{1\ \ 1\ \ 4\ \}$.

In questo caso, come già segnalato in sezione 1, minimizzare la funzione quadratica f *equivale* a risolvere il sistema lineare

$$F(x)=\nabla f=Hx-b=0,\ \nabla^2 f=H=H^T,\ H>0,$$

in quanto il minimo esiste ed è unico (e globale).
La funzione (quadratica) da minimizzare, per disteso si scrive:

$$f(x_1,x_2,x_3)=x_1^2+x_2^2+x_3^2+x_1x_2+x_1x_3+x_2x_3-x_1-2x_2-3x_3.$$

Il minimizzatore è $\overline{x}=(-1/2,1/2,3/2)^T$, $f(\overline{x})=-5/2$ (minimo).

Nota. Gli esempi precedenti (e i due successivi) sono dello stesso tipo.

5. (funzione quadratica)

$$f(x)=f(x_1,x_2)=\frac{1}{2}x^THx-b^Tx=\frac{1}{2}[5x_1^2+2x_1x_2+2x_2^2]-3x_1,\ x^0=(0,0)^T.$$

$$\nabla f(x)=(5x_1+x_2-3,x_1+2x_2)^T,\quad H=\begin{pmatrix}5&1\\1&2\end{pmatrix},\quad b=\begin{pmatrix}3\\0\end{pmatrix}.$$

Il minimizzatore, come subito si verifica, è $\overline{x}=(2/3,-1/3)^T=(0.666667,-0.333333)$, e $f(\overline{x})=-1$ (minimo).
Poichè $\lambda(H)=\dfrac{7\pm\sqrt{13}}{2}$, la costante asintotica è $M=\dfrac{\sqrt{13}}{7}=0.52$ e quindi la velocità di convergenza è $R=-\log_{10}M\approx 0.29$. Per esempio, occorrono circa $k=\dfrac{\log 10^{-6}}{\log M}=21$ iterazioni per avere: $\dfrac{\|e^{(k)}\|_{H,2}}{\|e^{(0)}\|_{H,2}}<10^{-6}$.
Procedendo come negli esercizi precedenti si ha:

$$\nabla f(0,0)=(-3,0)^T=-p^{(0)}=-r^{(0)}$$

$$\alpha^{(0)}=\frac{r^{0T}r^0}{r^{0T}Hr^0}=\frac{3^2+0^2}{(3,0)\begin{pmatrix}5&1\\1&2\end{pmatrix}\begin{pmatrix}3\\0\end{pmatrix}}=\frac{1}{5}=0.2$$

$$x^{(1)}=x^{(0)}+\alpha^{(0)}r^{(0)}$$

$$=(0,0)^T+\frac{1}{5}(3,0)^T=(3/5,0)^T=(0.6,0)^T.$$

Volendo *derivare* per trovare $\alpha^{(0)}$, si ha:

$$x^{(1)} = x^{(0)} + \alpha^{(0)} r^{(0)} = x^{(0)} - \alpha^{(0)} \nabla f(x^{(0)}) = (3\alpha^{(0)}, 0)^T$$

$$f(x^{(1)}) = \varphi(\alpha) = \frac{1}{2}\, 45\alpha^2 - 9\alpha, \qquad (\alpha^{(0)} = \alpha)$$

$$\varphi'(\alpha) = 45\alpha - 9 = 0 \quad \Longrightarrow \quad \alpha^{(0)} = \frac{1}{5}.$$

Calcoliamo la seconda iterata $x^{(2)}$.

$$r^{(1)} = -\nabla f(x^{(1)}) = -(5 \times 3/5 - 3,\ 3/5)^T = (0,\ -3/5)^T$$

$$\alpha^{(1)} = \frac{{r^1}^T r^1}{{r^1}^T H r^1} = \frac{9/25}{(0,\ -3/5) \begin{pmatrix} 5 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} 0 \\ -3/5 \end{pmatrix}} = \frac{1}{2} = 0.5$$

$$x^{(2)} = x^{(1)} + \alpha^{(1)} r^{(1)}$$

$$= (3/5\ , 0)^T + \frac{1}{2}\ (0,\ -3/5)^T = (3/5,\ -3/10)^T = (0.6,\ -0.3)^T\,.$$

Ecc...

6. (funzione quadratica)

$$f(x_1, x_2) = 5x_1^2 + 2x_2^2 - 2x_1x_2 - 2x_1 - 2x_2 + 1,\ x^0 = (0,0)^T,$$

$$\nabla f = (10x_1 - 2x_2 - 2, 4x_2 - 2x_1 - 2,)^T, \quad H = \begin{pmatrix} 10 & -2 \\ -2 & 4 \end{pmatrix},\ b = \begin{pmatrix} 2 \\ 2 \end{pmatrix},\ c = 1.$$

Il minimizzatore, come subito si verifica, è $\overline{x} = (1/3, 2/3)^T$, con $f(\overline{x}) = 1/5$ (minimo).
Si trova $x^{(1)} = (2/5, 2/5)^T$, ecc...

Questo problema (cfr. in sezione 1) deriva dal seguente problema

(P1): risolvere il sistema[15]

$$F(x) = 0 \quad \Longleftrightarrow \quad \begin{cases} f_1(x_1, x_2) = \quad x_1 + x_2 - 1 = 0 \\ f_2(x_1, x_2) = -2x_1 + x_2 \quad\ = 0 \end{cases}$$

con il metodo dei *minimi quadrati non lineari* (cfr. in sezione 1.9), cioè

$$\min\{\|F(x)\|_2^2\} = \min\{F(x)^T F(x)\} = \min\{\sum_{i=1}^{n} f_i^2\}.$$

[15]In questo esempio, essendo la $f(x_1, x_2)$ quadratica, il sistema $F(x) = 0$ è lineare e facilmente risolubile, ma in generale esso è non lineare.

In questo esempio, (n=2) e:

$$\min\ f(x_1, x_2) = (x_1 + x_2 - 1)^2 + (-2x_1 + x_2)^2.$$

Abbiamo già avvertito in sezione 1 che in generale i due problemi:

P1: soluzione sistema non lineare e

P2: minimizzazione non vincolata,

non sono equivalenti, nel senso che il secondo può introdurre dei minimi locali che non sono soluzioni del sistema non lineare, (si riveda la Figura 1.2). In questo esempio lo sono.

NOTA. Per risolvere questi problemi, piuttosto che i metodi di tipo gradiente o più in generale qualsiasi metodo di minimizzazione non vincolata (per esempio il metodo quasi-Newton *BFGS,* sezione 1.8), si usano metodi (ad hoc) quali (il metodo di Newton, da cui i metodi di) *Gauss-Newton* e *Levenberg-Marquardt*, in quanto essi, come si vedrà in *sezione 1.9*, tengono conto della speciale forma della funzione da minimizzare (somma di quadrati).

7. Trovare il minimo locale della funzione obiettivo (funzione cubica)

$$f(x_1, x_2) = 4x_1^2 + x_2^2 - x_1^2 x_2, \qquad x^{(0)} = (1,\ 1)^T$$

Si ha:

$$\nabla f = (8x_1 - 2x_1x_2, 2x_2 - x_1^2)^T, \quad H = \begin{pmatrix} 8 - 2x_2 & -2x_1 \\ -2x_1 & 2 \end{pmatrix}$$

(a) Da $\nabla f = 0$, si trova facilmente che il minimizzatore (locale) cercato è $\overline{x} = (0, 0)^T$, con $f(\overline{x}) = 0$.

Osservazione. $H(0,0) = \begin{pmatrix} 8 & 0 \\ 0 & 2 \end{pmatrix}$ è definita positiva (> 0). Si ha $M = \dfrac{8-2}{8+2} = 0.6$ e quindi occorrono circa $k \geq \dfrac{\log_{10} 10^{-6}}{\log_{10} M} \approx 28$ iterazioni per avere $\dfrac{\|e^{(k)}\|_{H,2}}{\|e^{(0)}\|_{H,2}} \leq 10^{-6}$. Più precisamente, essendo $\|e^{(0)}\|_{H,2} = \sqrt{(1 \quad 1)\begin{pmatrix} 8 & 0 \\ 0 & 2 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix}} = \sqrt{10}$, da $\|e^{(k)}\|_{H,2} \leq M^k \|e^{(0)}\|_{H,2} \leq 10^{-6}$, si ottiene $k \approx 30$ iterazioni.

(b) Minimizziamo la funzione composta

$$f(x^{(1)}) = f(x^{(0)} + \alpha p^{(0)}) = \varphi(\alpha), \quad \text{con} \quad p^{(0)} = -\nabla f_0 = (-6,\ -1)^T$$

cioè $\varphi(\alpha) = 4(1-6\alpha)^2 + (1-\alpha)^2 - (1-6\alpha)^2(1-\alpha)$. Derivando si trova $\alpha^{(0)} \approx 0.17389$. Quindi, alla prima iterazione di S.D. si ottiene:

$$\alpha^{(0)} \approx 0.17389 \quad \Longrightarrow \quad x^{(1)} = x^{(0)} + \alpha^{(0)} p^{(0)} = [-0.04334,\ 0.82611]^T.$$

(c) Verifichiamo se $\alpha^{(0)} \approx 0.17389$ e quindi la prima iterata $x^{(1)}$ calcolata in (b) soddisfa la condizione (di sufficiente diminuzione) di Armijo, con $\lambda = 0.1$.

Si ha:

$$\begin{aligned} x^{(0)} &= (1,\ 1)^T \quad \Longrightarrow \quad f(x^{(0)}) = 4; \\ x^{(1)} &= (-0.04334,\ 0.82611)^T \quad \Longrightarrow \quad f(x^{(1)}) = 0.68841943 \end{aligned}$$

Poichè: $\alpha^{(0)} = 0.17389$, $p^{(0)} = -\nabla f_0 = (-6,\ -1)^T$ e $\lambda = 0.1$, si ottiene (condizione di Armijo):

$$\begin{aligned} f(x^{(0)}) + \lambda\ \alpha^{(0)}\ (p^{(0)})^T\ \nabla f(x^{(0)}) &= f(x^{(0)}) - \lambda\ \alpha^{(0)} \|\nabla f(x^{(0)})\|^2 \\ &= 4 - (0.1)(0.17389) \times (6,\ 1) \begin{pmatrix} 6 \\ 1 \end{pmatrix} = 3.356607. \end{aligned}$$

Quindi: $f(x^{(1)}) = 0.068841943 < 3.356607$ e la regola di Armijo è soddisfatta.

1.5 Metodo del gradiente coniugato (C.G.)

Come già accennato[16], esso è un metodo di ricerca monodimensionale che (generalmente) si applica al caso di *n molto grande (e per problemi sparsi).* Esso è un esempio importante di *metodi vettore,* così detti in quanto si richiede la memorizzazione di soli vettori con una occupazione di memoria dell'ordine di n. La direzione di ricerca p (Figura 1.3, caso c) è data da:

$$\begin{aligned} p^{(0)} &= -\nabla f^{(0)} \\ p^{(k)} &= -\nabla f^{(k)} + \beta^{(k-1)}\ p^{(k-1)}, \end{aligned}$$

dove (Fletcher-Reeves, 1964):

$$\beta^{(k-1)} = \frac{\langle \nabla f^{(k)}, \nabla f^{(k)}\ \rangle}{\langle \nabla f^{(k-1)}, \nabla f^{(k-1)}\ \rangle}. \tag{1.16}$$

Questa formula coincide con quella del caso di f quadratica (sezione 1.1):

$$\beta^{(k-1)} = \frac{\langle \nabla f^{(k)}, \nabla f^{(k)}\ \rangle}{\langle \nabla f^{(k-1)}, \nabla f^{(k-1)}\ \rangle} = \frac{\langle \nabla f^{(k)}, Ap^{(k-1)}\ \rangle}{\langle Ap^{(k-1)}, p^{(k-1)}\ \rangle} = -\frac{\langle r^{(k)}, Ap^{(k-1)}\ \rangle}{\langle Ap^{(k-1)}, p^{(k-1)}\ \rangle}.$$

[16] Cfr. il caso dei sistemi lineari simmetrici (Hestenes e Stiefel, 1952) sezione 1.1.

Computazionalmente più efficiente si è dimostrata la formula (Polak-Ribiere, 1969), cfr. [19, pag. 468]:

$$\beta^{(k-1)} = \frac{\langle \nabla f^{(k)}, \nabla f^{(k)} - \nabla f^{(k-1)} \rangle}{\langle \nabla f^{(k-1)}, \nabla f^{(k-1)} \rangle}. \tag{1.17}$$

Nel caso quadratico le due formule coincidono. Al solito, $\alpha^{(k)}$ si calcola tramite minimizzazione monodimensionale (linesearch, sezione 1.3).

NOTA

Richiamiamo l'algoritmo del gradiente coniugato per i sistemi lineari ($Ax = b,\ A = A^T > 0,\ x \in R^n$) visto in sezione 1.1. In esso devono essere soddisfatte le due condizioni **a)** e **b)** seguenti.

a) $$x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$$

Poichè si richiede:

$$\langle\ r^{(k+1)}, r^{(k)}\ \rangle = 0$$

(per la minimizzazione dell'errore) deve essere, come visto in sezione 1.2

$$\alpha^{(k)} = \frac{\langle p^{(k)}, r^{(k)}\ \rangle}{\langle Ap^{(k)}, p^{(k)}\ \rangle}\ .$$

b) $$p^{(k+1)} = r^{(k+1)} + \beta^{(k)} p^{(k)},$$

per la quale si richiede la proprietà di A-ortogonalità:

$$\langle\ p^{(k+1)}, Ap^{(k)}\ \rangle = 0$$

la quale fornisce:

$$\langle r^{(k+1)} + \beta^{(k)} p^{(k)}, Ap^{(k)} \rangle = \langle r^{(k+1)}, Ap^{(k)} \rangle + \beta^{(k)} \langle p^{(k)}, Ap^{(k)} \rangle = 0,$$

ossia

$$\beta^{(k)} = -\frac{\langle r^{(k+1)}, Ap^{(k)}\ \rangle}{\langle Ap^{(k)}, p^{(k)}\ \rangle}\ .$$

Si noti che la direzione di ricerca p è una combinazione lineare del residuo r e della direzione p al passo precedente.
In conclusione, l'algoritmo del gradiente coniugato (C.G) nel caso dei sistemi lineari $Ax = b$ si scrive:

Algoritmo del gradiente coniugato $(Ax = b,\ A = A^T > 0)$

In Input: $x^{(0)}, A, b, k_{\max}$, $r^{(0)} = (\ p^{(0)} = -\nabla f^{(0)}) = b - Ax^{(0)}$, Toll

- **1.** Per $k = 0, 1, \ldots$ fino a convergenza
- $\alpha^{(k)} = \dfrac{\langle p^{(k)}, r^{(k)}\ \rangle}{\langle Ap^{(k)}, p^{(k)}\ \rangle}$
- $x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$
- $r^{(k+1)} = \ r^{(k)} - \alpha^{(k)} Ap^{(k)}$
- $\beta^{(k)} = -\dfrac{\langle r^{(k+1)}, Ap^{(k)}\ \rangle}{\langle Ap^{(k)}, p^{(k)}\ \rangle}$
- $p^{(k+1)} = r^{(k+1)} + \beta^{(k)} p^{(k)}$
- se $\|x^{(k+1)} - x^{(k)}\| = \|\alpha^{(k)} p^{(k)}\| \leq$ Toll, allora Stop
- altrimenti, vai a 1 (fino a $k \leq k_{\max}$).

I due coefficienti $\alpha^{(k)}$ e $\beta^{(k)}$ si possono scrivere in modi diversi, ma matematicamente equivalenti (si veda in sezione B.3.1).

Proprietà del C.G. Si dimostra che (cfr. [8, teorema 5.32]):

1) $$\langle\ p^{(k+1)}, Ap^{(i)}\ \rangle = 0, \qquad \text{per} \quad 0 \leq i \leq k,$$

cioè $p^{(n)}$ è I-ortogonale alla varietà $span\langle Ap^{(0)}, Ap^{(1)}, \ldots, Ap^{(n-1)}\rangle$, e quindi $p^{(n)} \equiv 0$, essendo ortogonale a n vettori linearmente indipendenti, e quindi $x^{(n+1)} = x^{(n)}$ (soluzione).

2) $$\langle\ r^{(n+1)}, r^{(i)}\ \rangle = 0, \qquad \text{per} \quad 0 \leq i \leq k,$$

cioè $r^{(n)}$ è I-ortogonale alla varietà $span\langle r^{(0)}, r^{(1)}, \ldots, r^{(n-1)}\rangle$, e quindi $r^{(n)} = b - Ax^{(n)} \equiv 0$. Perciò $x^{(n)}$ è la soluzione, trovata dopo n passi (*terminazione quadratica*).
In pratica questo *non* si verifica a causa degli inevitabili errori di arrotondamento, cioè $r^{(n)} \neq 0$. Per accelerare la convergenza si ricorre a una trasformazione di variabili (precondizionamento, cfr. B.4).

(Fine NOTA).

A parte il calcolo di $\alpha^{(k)}$ e la sostituzione del residuo con il gradiente della funzione obiettivo f, l'algoritmo del gradiente coniugato per funzioni generiche[17] è analogo all'algoritmo visto in sezione 1.1 per i sistemi lineari simmetrici e può anche essere *precondizionato*.
Ovviamente si ha *terminazione quadratica,* cioè convergenza in n passi (precisamente in n passi se la minimizzazione monodimensionale è esatta).

Il metodo C.G. converge linearmente, ma la costante asintotica dell'errore è più piccola del caso S.D. Si dimostra[18] infatti che (caso quadratico):

$$\|\overline{x} - x^k\|_{H,2} = \|e^k\|_{H,2} \leq 2\left(\frac{\sqrt{\lambda_1} - \sqrt{\lambda_n}}{\sqrt{\lambda_1} + \sqrt{\lambda_n}}\right)^k \|e^0\|_{H,2}, \qquad (1.18)$$

dove $\lambda_1,\ \lambda_n$ sono il massimo e il minimo autovalore di $H(\overline{x}) = \nabla^2 f(\overline{x})$.

Algoritmo C.G. (caso generale)

In Input: $f, x^{(0)}, k_{\max}, p^{(0)} = -\nabla f^{(0)}$, Toll

- **1.** Per $k = 0, 1, \ldots$ fino a $\|\nabla f(k)\| \leq$ Toll
- **calcolare** $\alpha^{(k)} : f(x^{(k)} + \alpha^{(k)} p^{(k)}) = \min_{\alpha>0}\{f(x^{(k)} + \alpha p^{(k)})\}$
  (con *linesearch*, vedi sezione 1.3)
- $x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$
- calcolare $\nabla f^{(k+1)}$
- $\beta^{(k)} = \dfrac{\langle \nabla f^{(k+1)}, \nabla f^{(k+1)} \rangle}{\langle \nabla f^{(k}, \nabla f^{(k)} \rangle}$ formula (1.16)
- $p^{(k+1)} = -\nabla f^{(k+1)} + \beta^{(k)} p^{(k)}$
- vai a 1 (fino a $k \leq k_{\max}$).

ESEMPI (con C.G.)

1. (si veda l'esercizio 2 in sezione 1.4)

$$f(x_1, x_2) = \frac{1}{2}x_1^2 + \frac{9}{2}x_2^2,\ x^0 = (9, 1)^T,\ \nabla f = (x_1, 9x_2)^T,\ H = \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix}$$

[17] Cfr. [19, (10.7.84), pag. 466], [34, sez. 5.2].
[18] Si veda [8], [19], [24, pag. 17]. La stima del numero di iterazioni k per ridurre l'errore iniziale vale se non si ha terminazione quadratica in n iterazioni.

Il minimizzatore è $\overline{x} = (0,0)^T$, e $f(\overline{x}) = 0$ (minimo).
Essendo la f una funzione quadratica, si ha convergenza (in aritmetica esatta) in 2 iterazioni (l'equazione $f = cost$ è un'ellisse).

La prima iterata: $x^{(1)} = (7.2,\ -0.8)^T$ coincide con quella, già calcolata, dello S.D.

Calcoliamo la seconda iterata $x^{(2)}$.
Troviamo la direzione $p^{(1)}$ H-coniugata a $p^{(0)} = (-9,\ -9)^T$.

$$p^{(0)^T} H\ p^{(1)} = (-9,-9) \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix} \begin{pmatrix} p_1^{(1)} \\ p_2^{(1)} \end{pmatrix} = -9p_1^{(1)} - 81p_2^{(1)} = 0,$$

da cui (per esempio) $\begin{cases} p_1^{(1)} = -1 \\ p_2^{(1)} = 1/9 \end{cases}$, (ecc...).

Essendo: $r^{(1)} = \begin{pmatrix} -7.2 \\ 7.2 \end{pmatrix}$, (già calcolato), si ha:

$$\alpha^{(1)} = \frac{\langle r^1, p^1 \rangle}{\langle Hp^1, p^1 \rangle} = \frac{{r^1}^T p^1}{{p^1}^T H p^1} = \frac{(-7.2, 7.2) \begin{pmatrix} -1 \\ 1/9 \end{pmatrix}}{(-1, 1/9) \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix} \begin{pmatrix} -1 \\ 1/9 \end{pmatrix}} = \frac{72}{10} = 7.2.$$

Infine: $x^{(2)} = x^{(1)} + \alpha^{(1)} p^{(1)} = \begin{pmatrix} 7.2 \\ -0.8 \end{pmatrix} + 7.2 \begin{pmatrix} -1 \\ 1/9 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$

Si noti che $\alpha^{(1)} p^{(1)} = (-7.2, 0.8)^T$.

In alternativa, possiamo calcolare l'iterata $x^{(2)}$ usando le formule iterative viste (cfr. anche in sezione B.3.1), con $\alpha^{(0)} = 0.2$ (già calcolato):

$$x^{(1)} = \begin{pmatrix} 7.2 \\ -0.8 \end{pmatrix}$$

$$r^{(1)} = r^{(0)} - \alpha^{(0)} H p^{(0)} = \begin{pmatrix} -9 \\ -9 \end{pmatrix} - 0.2 \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix} \begin{pmatrix} -9 \\ -9 \end{pmatrix} = \begin{pmatrix} -7.2 \\ 7.2 \end{pmatrix}$$

$$\beta^{(0)} = \frac{{r^1}^T r^1}{{r^0}^T r^0} = \frac{(-7.2, 7.2) \begin{pmatrix} -7.2 \\ 7.2 \end{pmatrix}}{(-9,-9) \begin{pmatrix} -9 \\ -9 \end{pmatrix}} = \frac{103.68}{162} = 0.64$$

$$p^{(1)} = r^{(1)} + \beta^{(0)} p^{(0)} = \begin{pmatrix} -7.2 \\ 7.2 \end{pmatrix} + 0.64 \begin{pmatrix} -9 \\ -9 \end{pmatrix} = \begin{pmatrix} -12.96 \\ 1.44 \end{pmatrix}$$

$$\alpha^{(1)} = \frac{{r^1}^T p^1}{{p^1}^T H p^1} = \frac{(-7.2, 7.2) \begin{pmatrix} -12.96 \\ 1.44 \end{pmatrix}}{(-12.96, 1.44) \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix} \begin{pmatrix} -12.96 \\ 1.44 \end{pmatrix}} = \frac{103.68}{186.624}$$

$$= \frac{5}{9} = 0.555556.$$

Si noti che $\alpha^{(1)} p^{(1)} = (-7.2, 0.8)^T$. Quindi

$$x^{(2)} = x^{(1)} + \alpha^{(1)} p^{(1)} = \begin{pmatrix} 7.2 \\ -0.8 \end{pmatrix} + \frac{5}{9} \begin{pmatrix} -12.96 \\ 1.44 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

2. (si veda l'esercizio 3 in sezione 1.4)

$$f(x) = f(x_1, x_2) = 2x_1^2 + x_2^2 - 3, \quad x^0 = (1, 1)^T,$$
$$\nabla f(x) = (4x_1, 2x_2)^T, \quad H = \begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix}$$

Il minimizzatore è $\overline{x} = (0, 0)^T$.
Essendo la f una funzione quadratica, si ha convergenza (in aritmetica esatta) in 2 iterazioni (l'equazione $f = cost$ è un'ellisse).

La prima iterata: $x^{(1)} = (-1/9,\ 4/9)^T$ coincide con quella, già calcolata, dello S.D.

Calcoliamo la seconda iterata $x^{(2)}$.
Troviamo la direzione $p^{(1)}$ H-coniugata a $p^{(0)} = (-4,\ -2)^T$

$${p^{(0)}}^T H p^{(1)} = (-4, -2)^T \begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} p_1^{(1)} \\ p_2^{(1)} \end{pmatrix} = -16 p_1^{(1)} - 4 p_2^{(1)} = 0,$$

da cui (per esempio) $\begin{cases} p_1^{(1)} = 1 \\ p_2^{(1)} = -4 \end{cases}$, (ecc...).

Essendo: $r^{(1)} = \begin{pmatrix} 4/9 \\ -8/9 \end{pmatrix}$, (già calcolato), si ha:

$$\alpha^{(1)} = \frac{\langle r^1, p^1 \rangle}{\langle Hp^1, p^1 \rangle} = \frac{{r^1}^T p^1}{{p^1}^T H p^1} = \frac{(4/9\ ,\ -8/9) \begin{pmatrix} 1 \\ -4 \end{pmatrix}}{(1, -4) \begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ -4 \end{pmatrix}} = \frac{1}{9} = 0.111111.$$

Infine:

$$x^{(2)} = x^{(1)} + \alpha^{(1)} p^{(1)} = \begin{pmatrix} -1/9 \\ 4/9 \end{pmatrix} + \frac{1}{9} \begin{pmatrix} 1 \\ -4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Si noti che $\alpha^{(1)} p^{(1)} = (1/9, -4/9)^T$.

In alternativa, possiamo calcolare l'iterata $x^{(2)}$ usando le formule iterative viste (cfr. anche in sezione B.3.1):

$$\begin{aligned} x^{(1)} &= \begin{pmatrix} -1/9 \\ 4/9 \end{pmatrix} \\ r^{(1)} &= r^{(0)} - \alpha^{(0)} H p^{(0)} = \begin{pmatrix} -4 \\ -2 \end{pmatrix} - \frac{5}{18} \begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} -4 \\ -2 \end{pmatrix} = \begin{pmatrix} 4/9 \\ -8/9 \end{pmatrix} \\ &= -\nabla f(x^{(1)}) \end{aligned}$$

$$\beta^{(0)} = \frac{{r^1}^T r^1}{{r^0}^T r^0} = \frac{(4/9, -8/9) \begin{pmatrix} 4/9 \\ -8/9 \end{pmatrix}}{(-4, -2) \begin{pmatrix} -4 \\ -2 \end{pmatrix}} = \frac{4}{81} = 0.049383$$

$$p^{(1)} = r^{(1)} + \beta^{(0)} p^{(0)} = \begin{pmatrix} 4/9 \\ -8/9 \end{pmatrix} + \frac{4}{81} \begin{pmatrix} -4 \\ -2 \end{pmatrix} = \begin{pmatrix} 20/81 \\ -80/81 \end{pmatrix}$$

$$\alpha^{(1)} = \frac{{p^1}^T r^1}{{p^1}^T H p^1} = \frac{(20/81, -80/81) \begin{pmatrix} 4/9 \\ -8/9 \end{pmatrix}}{(20/81, -80/81) \begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} 20/81 \\ -80/81 \end{pmatrix}} = \frac{9}{20} = 0.45.$$

Si noti che $\alpha^{(1)} p^{(1)} = (1/9, -4/9)^T$. Quindi

$$x^{(2)} = x^{(1)} + \alpha^{(1)} p^{(1)} = \begin{pmatrix} -1/9 \\ 4/9 \end{pmatrix} + \frac{9}{20} \begin{pmatrix} 20/81 \\ -80/81 \end{pmatrix} = \begin{pmatrix} 1/9 - 1/9 \\ 4/9 - 4/9 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

3. (si veda l'esercizio 5 in sezione 1.4)

$$f(x) = f(x_1, x_2) = \frac{1}{2} x^T H x - b^T x = \frac{1}{2}[5x_1^2 + 2x_1 x_2 + 2x_2^2] - 3x_1, \; x^0 = (0,0)^T.$$

$$\nabla f(x) = (5x_1 + x_2 - 3, x_1 + 2x_2)^T, \quad H = \begin{pmatrix} 5 & 1 \\ 1 & 2 \end{pmatrix}, \quad b = \begin{pmatrix} 3 \\ 0 \end{pmatrix}.$$

Il minimizzatore, come subito si verifica, è $\overline{x} = (2/3, -1/3)^T$, e $f(\overline{x}) = -1$ (minimo).

Essendo la f una funzione quadratica, si ha convergenza (in aritmetica esatta) in 2 iterazioni (l'equazione $f = cost$ è un'ellisse).

La prima iterata: $x^{(1)} = (3/5, 0)^T$ coincide con quella, già calcolata, dello S.D.

Calcoliamo la seconda iterata $x^{(2)}$. Troviamo la direzione $p^{(1)}$ H-coniugata a $p^{(0)} = (3,\ 0)^T$

$$p^{(0)^T} H\ p^{(1)} = (3,0) \begin{pmatrix} 5 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} p_1^{(1)} \\ p_2^{(1)} \end{pmatrix} = 3[5p_1^{(1)} + p_2^{(1)}] = 0,$$

$$\text{da cui (per esempio)} \qquad \begin{cases} p_1^{(1)} = 1 \\ p_2^{(1)} = -5 \end{cases}, \quad \text{(ecc...)}.$$

Essendo: $r^{(1)} = -\nabla f(x^{(1)}) = -\ (5 \times 3/5 - 3, 3/5)^T = \begin{pmatrix} 0 \\ -3/5 \end{pmatrix}$, si ha:

$$\alpha^{(1)} = \frac{\langle r^{(1)}, p^{(1)}\ \rangle}{\langle Hp^{(1)}, p^{(1)}\ \rangle} = \frac{{r^1}^T p^1}{{p^1}^T H p^1} = \frac{(0\ ,\ -3/5) \begin{pmatrix} 1 \\ -5 \end{pmatrix}}{(1,-5) \begin{pmatrix} 5 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ -5 \end{pmatrix}} = \frac{1}{15} = 0.0666667.$$

Infine:

$$x^{(2)} = x^{(1)} + \alpha^{(1)} p^{(1)} = \begin{pmatrix} 3/5 \\ 0 \end{pmatrix} + \frac{1}{15} \begin{pmatrix} 1 \\ -5 \end{pmatrix} = \begin{pmatrix} 2/3 \\ -1/3 \end{pmatrix}.$$

Si noti che $\alpha^{(1)} p^{(1)} = (1/15, -1/3)^T$.

In alternativa, possiamo calcolare l'iterata $x^{(2)}$ usando le formule iterative viste (cfr. anche in sezione B.3.1), con $\alpha^{(0)} = 0.2$ (già calcolato):

$$x^{(1)} = \begin{pmatrix} 3/5 \\ 0 \end{pmatrix}$$

$$r^{(1)} = r^{(0)} - \alpha^{(0)} H p^{(0)} = \begin{pmatrix} 3 \\ 0 \end{pmatrix} - 0.2 \begin{pmatrix} 5 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} 3 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ -3/5 \end{pmatrix}$$

$$= \begin{pmatrix} 0 \\ -0.6 \end{pmatrix} = -\nabla f(x^{(1)})$$

$$\beta^{(0)} = \frac{{r^1}^T r^1}{{r^0}^T r^0} = \frac{(0, -0.6) \begin{pmatrix} 0 \\ -0.6 \end{pmatrix}}{(3,0) \begin{pmatrix} 3 \\ 0 \end{pmatrix}} = \frac{1}{25} = 0.04$$

$$p^{(1)} = r^{(1)} + \beta^{(0)} p^{(0)} = \begin{pmatrix} 0 \\ -3/5 \end{pmatrix} + \frac{1}{25}\begin{pmatrix} 3 \\ 0 \end{pmatrix} = \begin{pmatrix} 3/25 \\ -3/5 \end{pmatrix} = \begin{pmatrix} 0.12 \\ -0.6 \end{pmatrix}$$

$$\alpha^{(1)} = \frac{{p^1}^T r^1}{{p^1}^T H p^1} = \frac{(3/25, -3/5)\begin{pmatrix} 0 \\ -3/5 \end{pmatrix}}{(3/25, -3/5)\begin{pmatrix} 5 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 3/25 \\ -3/5 \end{pmatrix}} = \frac{9}{25}\frac{125}{81} = \frac{5}{9} \ .$$

Si noti che $\alpha^{(1)} p^{(1)} = (1/15, -1/3)^T$. Quindi

$$x^{(2)} = x^{(1)} + \alpha^{(1)} p^{(1)} = \begin{pmatrix} 3/5 \\ 0 \end{pmatrix} + \frac{5}{9}\begin{pmatrix} 3/25 \\ -3/5 \end{pmatrix} = \begin{pmatrix} 2/3 \\ -1/3 \end{pmatrix} \ .$$

4. (Cfr. [17, pag. 207]) funzione "banana" di Rosenbrock (somma di quadrati)

$$f(x) = f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2, \quad x^0 = (-1.2, 1.0)^T \text{ (usuale)}$$

Il minimizzatore è $\overline{x} = (1, 1)^T$.

1.6 Metodo di Newton (per l'ottimizzazione)

In (1.1) si pone

$$\boxed{p^{(k)} = -H^{-1}(x^{(k)})\ \nabla f(x^{(k)})} \tag{1.19}$$

($p^{(k)} = -H^{-1}(x^{(k)})\ \nabla f(x^{(k)}$, direzione di Newton), e $\alpha^{(k)} = 1$.

Il metodo di Newton si scrive

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)} = x^{(k)} - H^{-1}(x^{(k)})\ \nabla f(x^{(k)}) \tag{1.20}$$

cioè:

$$H(x^{(k)})\ (x^{(k+1)} - x^{(k)}) = -\nabla f(x^{(k)}).$$

Se $\alpha^{(k)} \neq 1$ si ha il metodo di *Newton rilassato* (o smorzato), da usare se la funzione obiettivo f non diminuisce abbastanza (per esempio se si discosta troppo da una funzione quadratica). In tal caso, la costante $\alpha^{(k)}$ (> 0) si calcola al solito con una minimizzazione monodimensionale (line search).

Note

1. Il metodo altro non è se non il classico metodo di Newton per la soluzione dell'equazione non lineare (sistema) $F(x) = \nabla f = 0$ caratterizzante i punti critici della f, da cui $\nabla(\nabla f)s = -\nabla f$ e quindi $s = -(\nabla^2 f)^{-1}(\nabla f)$.

Se la f è la funzione quadratica (1.2), si ha ovviamente convergenza in una sola iterazione:

$$x^1 = x^0 + s^0 = x^0 - A^{-1}(Ax^0 - b) = x^0 - x^0 + A^{-1}b = A^{-1}b.$$

2. Casi particolari di (1.19):

(i) $H = H^{-1} = I$, da cui S.D. di sezione 1.4.

(ii) $H \approx \hat{H} = B$ (B una approssimazione della hessiana o della sua inversa), da cui i *metodi secanti* di sezione 1.8.

3. La direzione (1.19) di Newton è una direzione di discesa?
La risposta è affermativa *se e solo se* (cfr. la condizione (1.6))

$$\langle \nabla f, p \rangle = \langle \nabla f, -H^{-1}\nabla f \rangle \ = (\nabla f)^T(-H^{-1})\ (\nabla f) < 0$$

cioè se e solo se la hessiana $H = \nabla^2 f$ (ovvero la sua inversa) è definita positiva: $(\nabla f)^T H\ (\nabla f) > 0$. In tal caso la direzione di Newton è ideale.

Ma il costo del metodo di Newton è alto, in quanto, oltre a dover valutare l'hessiana, occorre "invertirla", ossia risolvere, a ogni iterazione, il sistema $Hs = -\nabla f$ (con $O(n^3)$ operazioni).

Per conservare la definitezza positiva dell'hessiana, si può modificare la direzione di ricerca orientandola di più verso la direzione del gradiente. Ciò si ottiene risolvendo il sistema[19]

$$(H + \mu I)s = -\nabla f$$

invece del sistema $Hs = -\nabla f$, dove $\mu \geq 0$ è un parametro (di shift) *opportunamente* scelto. Si può, per esempio, porre $\mu > -\min\{\lambda_i\}$, dove λ_i è un autovalore di H. Una stima di $-\min\{\lambda_i\}$, si può ottenere utilizzando il teorema di Gershgorin (cfr. [45, Teorema 3.1.3]) alla matrice H.

Come vedremo, questi inconvenienti possono essere superati con i *metodi secanti* di sezione 1.8.

4. (Cfr. [15, pag. 147]). È il più "veloce" di tutti gli altri metodi visti nel capitolo, ovviamente, *ma*: la convergenza è locale, ecc....

5. (Cfr. [15, pag. 147]).
È ben noto che il metodo di Newton per la soluzione dei sistemi non lineari $F(x) = (f_1, f_2, \ldots, f_n)^T = 0$ ha convergenza quadratica, ma *locale.* La convergenza *globale* si può ottenere adottando una strategia di convergenza

[19] Cfr. il metodo di di Levenberg-Marquardt in sezione 1.9.

globale applicata al problema di minimizzare la funzione $\|F(x)\|_2$ (minimi quadrati non lineari, sezione 1.9; (si riveda l'esempio 6 in sezione 1.4).
Infatti, ciò corrisponde ad avere

$$\|F(x^{(k+1)})\|_2 < \|F(x^{(k)})\|_2$$

con una *sufficiente* diminuzione, tramite una strategia di *backtracking* (riduzione del passo), accennata in sezione 1.3.
Vedi [15] e [24, pag. 137] (per il caso dei sistemi).

Avvertiamo che, oltre alla strategia di ricerca monodimensionale con il parametro di ricerca $\alpha^{(k)}$ scelto in modo da avere una sufficiente diminuzione della f, un'altra importante strategia di convergenza *globale* è quella della *trust region*. **Si veda la Tesina [6]**.
Cfr. per esempio [15, cap. 6], [18, pag. 4], [25].

Circa gli inconvenienti (e i rimedi) nel metodo di Newton e un algoritmo generale si veda [17, pag. 213] e [19, pag. 456–458].

Nella sua forma più semplice, l'algoritmo di Newton si scrive:

Algoritmo (Newton)

In Input: $x^{(0)}, f, \nabla f, \nabla^2 f, k_{\max}$, Toll

1. Per $k = 0, 1, \ldots$ fino a convergenza

2. $\nabla^2 f(x^{(k)}) s^{(k)} = -\nabla f(x^{(k)})$ % (Da risolvere con fattorizzazione LL^T)

3. $x^{(k+1)} = x^{(k)} + s^{(k)}$

4. se $\|s^k\| = \|x^{(k+1)} - x^{(k)}\| \leq Toll$, allora Stop

5. altrimenti, vai a 1 (fino a $k \leq k_{\max}$).

Se si vuole usare un metodo di Newton smorzato, sostituire il punto **3** con:

- **calcolare** $\alpha^{(k)}$: $f(x^{(k)} + \alpha^{(k)} p^{(k)}) = \min_{\alpha>0}\{f(x^{(k)} + \alpha p^{(k)})\}$
  ($p^{(k)} = s^{(k)}$, calcolato al punto **2**)
- $x^{(k+1)} = x^{(k)} + \alpha^{(k)} s^{(k)}$

ESEMPI (da risolvere con Newton)

1.

$$f(x) = f(x_1, x_2) = 4x_1^2 + x_2^2 - 2x_1x_2, \quad x^0 = (1,1)^T,$$
$$\nabla f(x) = (8x_1 - 2x_2, 2x_2 - 2x_1)^T, \quad H = \begin{pmatrix} 8 & -2 \\ -2 & 2 \end{pmatrix}, \quad b = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Il minimizzatore è $\overline{x} = (0,0)^T$. Essendo f quadratica, si ha convergenza in una iterazione (in aritmetica esatta). Infatti:

$$Hs^0 = -\nabla f(x^0). \qquad \begin{pmatrix} 8 & -2 \\ -2 & 2 \end{pmatrix} \begin{pmatrix} s_1^0 \\ s_2^0 \end{pmatrix} = - \begin{pmatrix} 6 \\ 0 \end{pmatrix}$$

da cui

$$\begin{pmatrix} s_1^0 \\ s_2^0 \end{pmatrix} = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$$

e dunque

$$x^1 = x^0 + s^0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} + \begin{pmatrix} -1 \\ -1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \equiv \overline{x}.$$

2. (si veda l'esercizio 1 in sezione 1.4)

$$f(x) = f(x_1, x_2) = x_1^2 + 4x_2^2, \quad x^0 = (1,1)^T$$

Il minimizzatore è $\overline{x} = (0,0)^T$. Essendo f quadratica, si ha convergenza in una iterazione.

3.

$$f(x) = f(x_1, x_2) = x_1^2 + x_1x_2 + x_2^2 + 3x_1, \quad x^0 = (0,0)^T$$

$$\nabla f(x) = (2x_1 + x_2 + 3,\ x_1 + 2x_2)^T, \quad H = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}, \quad b = \begin{pmatrix} -3 \\ 0 \end{pmatrix}$$

Il minimizzatore è $\overline{x} = (-2,1)^T$. Essendo f quadratica, si ha convergenza in una iterazione.

4.

$$f(x) = f(x_1, x_2) = (x_1 - 2)^4 + (x_1 - 2)^2x_2^2 + (x_2 + 1)^2, \quad x^0 = (1,1)^T$$

Si noti che f non è quadratica (è una somma di quadrati). Il minimizzatore è $\overline{x} = (2,-1)^T$, raggiunto in sei iterazioni (con otto cifre esatte) .

5.

$$f(x) = f(x_1, x_2) = \frac{1}{2}x_1^2x_2^2 + x_1^2 + x_2^2 + 2x_1 + x_2, \quad x^0 = (1,1)^T$$
$$\nabla f(x) = (x_1x_2^2 + 2x_1 + 2, x_1^2x_2 + 2x_2 + 1)^T, \quad H(x) = \begin{pmatrix} x_2^2+2 & 2x_1x_2 \\ 2x_1x_2 & x_1^2+2 \end{pmatrix}$$

Si noti che f non è quadratica.
Il minimizzatore è $\overline{x} \approx (-0.9435 - 0.3460)^T$, con $f(\overline{x}) = -1.16981$.

Si ottiene: $s^0 = \left[-\frac{7}{5}, -\frac{2}{5}\right]$; $\|s^0\| = \|p^0\| = \sqrt{\frac{53}{25}} = 1.456$ e
$x^1 = [-0.4,\ 0.6]$, con $f(x^1) = 0.3488 < f(x^0) = 5.5$.

Non essendo f quadratica, può essere conveniente provare un Newton rilassato ($\alpha^{(k)} \neq 1$).
Minimizzando la $f(x^1) = f(x^0 + \alpha s^0) = \varphi(\alpha)$, si trova (derivando rispetto ad α):
$x_r^1 = (-1.24, 0.36)^T$ per $\alpha^0 = 1.601$, e
$f(x_r^1) = -0.3532 < f(x^1) = 0.3488 < f(x^0) = 5.5$.

Si noti che $\|s_r^0\| = |\alpha^0| \cdot \|s^0\| = 2.334 > \|s^0\| = 1.456$.

6.

$$f(x) = f(x_1, x_2) = x_1^4 + x_1x_2 + (1 + x_2)^2, \quad x^0 = (0,0)^T$$
$$\nabla f(x) = (4x_1^3 + x_2, x_1 + 2(1 + x_2))^T, \quad H(x) = \begin{pmatrix} 12x_1^2 & 1 \\ 1 & 2 \end{pmatrix}$$

Il minimizzatore è $\overline{x} \approx (0.6959, -1.3479)^T$.
Se $x^0 = (0,0)^T$ (lontano da $\overline{x}$), il metodo di Newton si muove nella direzione sbagliata. Infatti: $\nabla f(0) = (0,2)^T$, $H(0) = \begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix} < 0$ (definita negativa), e quindi $x^1 = s^0 = (-2,0)^T$ e il punto si allontana.
Se invece $x^0 = (0.7, -0.8)^T$ (vicino a $\overline{x}$), la convergenza è molto rapida. Infatti: $\nabla f(x^0) = (0.572, 1.1)^T$, $H(x^0) = \begin{pmatrix} 5.88 & 1 \\ 1 & 2 \end{pmatrix} > 0$, per cui $s^0 = (-0.004, -0.548)$ e quindi $x^1 = x^0 + s^0 = (0.696, -1.348)^T \approx \overline{x}$.

7.

$$f(x) = f(x_1, x_2) = \frac{1}{3}x_1^3 - x_1x_2 + x_2^3, \quad x^0 = (0.5,\ 0.5)^T$$
$$\nabla f(x) = (x_1^2 - x_2,\ -x_1 + 3x_2^2)^T, \quad H(x) = \begin{pmatrix} 2x_1 & -1 \\ -1 & 6x_2 \end{pmatrix}$$

Si noti che f non è quadratica. Il minimizzatore è $\overline{x} = (3^{-1/3},\ 3^{-2/3})^T \approx (0.6934,\ 0.4807)^T$, con $f(\overline{x}) = -1/9 = -0.111111$.

Si ha:

$$f(x^{(0)}) = -0.083333; \quad \nabla f^{(0)}) = (-0.25,\ 0.25)^T.$$

(a) Calcoliamo le prime 3 iterate con il metodo di Newton:

$$H^{(k)}s^{(k)} = -\nabla f^{(k)} \qquad s^{(k)} = \alpha^{(k)}p^{(k)},\ \alpha^{(k)} = 1, \quad k \geq 0.$$

Si trova

$$s^{(0)} = \begin{pmatrix} 0.25 \\ 0.0 \end{pmatrix}; \quad s^{(1)} = \begin{pmatrix} -0.5357143 \\ -0.1785714 \end{pmatrix}; \quad s^{(2)} = \begin{pmatrix} -0.003056 \\ -0.001387 \end{pmatrix}$$

$$x^{(1)} = \begin{pmatrix} 0.75 \\ 0.5 \end{pmatrix}; \quad x^{(2)} = \begin{pmatrix} 0.696429 \\ 0.482143 \end{pmatrix}; \quad x^{(3)} = \begin{pmatrix} 0.693372 \\ 0.480756 \end{pmatrix}$$

(b) Per esercizio, verifichiamo se nel calcolo della prima iterata $x^{(1)}$ si è ottenuta una *sufficiente* diminuzione della f secondo la regola di Armijo (1.12), con $\lambda = 0.01$.
Si ha:

$$\begin{aligned} x^{(0)} &= (0.5,\ 0.5)^T \quad \Longrightarrow \quad f(x^{(0)}) = -0.083333; \\ x^{(1)} &= (0.75,\ 0.5)^T \quad \Longrightarrow \quad f(x^{(1)}) = -0.109375 \end{aligned}$$

Poichè: $\alpha^{(0)} = 1, \quad p^{(0)} = (0.25,\ 0)^T$ e $\lambda = 0.01,$ si ottiene

$$\begin{aligned} f(x^{(0)}) + \lambda\ \alpha^{(0)}\ (p^{(0)})^T\ \nabla f(x^{(0)}) &= -0.083333 + 0.01 \times (0.25,\ 0) \begin{pmatrix} -0.25 \\ 0.25 \end{pmatrix} \\ &= -0.083958 \end{aligned}$$

Quindi $f(x^{(1)}) = -0.109375 < -0.083958$ e la regola di Armijo è soddisfatta.

(c) Per *esercizio*, calcoliamo la prima iterata $x^{(1')}$ con un Newton rilassato ($\alpha^{(0)} \neq 1$). Minimizziamo la $f(x^{(1')}) = x^{(0)} + \alpha s^{(0)}) = \varphi(\alpha)$, derivando rispetto ad α.

$$x^{(1')} = x^{(0)} + \alpha\ s^{(0)} = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} + \alpha \begin{pmatrix} 0.25 \\ 0 \end{pmatrix} = \begin{pmatrix} 0.5 + 0.25\ \alpha \\ 0.5 \end{pmatrix}$$

$$\varphi(\alpha) = f(x^{(1')}) = \frac{1}{3}(0.5 + 0.25\ \alpha)^3 - \frac{1}{2}(0.5 + 0.25\ \alpha) + \frac{1}{8}$$

Derivando si ottiene il valore ottimo di α (positivo):

$$\varphi'(\alpha) = \frac{1}{4}(0.5 + 0.25\ \alpha)^2 - \frac{1}{8} = 0, \quad \Longrightarrow \quad \alpha = 2(\sqrt{2} - 1) = 0.828427$$

Si noti che:

$$x^{(1')} = (0.707107,\ 0.5)^T \implies f(x^{(1')}) = -0.1107025 < -0.109375 = f(x^{(1)}).$$

8. (Si veda l'esercizio 7 in sezione 1.4 (SD).) Minimo locale della funzione

$$f(x_1, x_2) = 4x_1^2 + x_2^2 - x_1^2 x_2, \qquad x^{(0)} = (1,\ 1)^T.$$

Si verifichi se la prima iterata $x^{(1)}$ soddisfa la condizione di (sufficiente diminuzione) di Armijo, con $\lambda = 0.1$. In caso contrario, si dimezzi lo scalare di discesa $\alpha^{(0)}$, fino a soddisfare la condizione.

Si verifica facilmente che il minimizzatore è $\overline{x} = (0, 0)^T$, con $f(\overline{x}) = 0$.

Partendo dal vettore iniziale $x^{(0)} = (1,\ 1)^T$ calcoliamo la prima iterata $x^{(1)}$, usando il metodo di Newton:

$$H^{(0)} s^{(0)} = -\nabla f^{(0)}; \qquad s^{(0)} = \alpha^{(0)} p^{(0)}, \quad \alpha^{(0)} = 1.$$

Si noti che $H(x^{(0)})$ è definita positiva e quindi $s^{(0)}$ è una direzione di discesa. Si ha:

$$\begin{pmatrix} 6 & -2 \\ -2 & 2 \end{pmatrix} \begin{pmatrix} s_1^0 \\ s_2^0 \end{pmatrix} = - \begin{pmatrix} 6 \\ 1 \end{pmatrix};\ s^{(0)} = \begin{pmatrix} -1.75 \\ -2.25 \end{pmatrix} \text{ e } x^{(1)} = \begin{pmatrix} -0.75 \\ -1.25 \end{pmatrix}.$$

Si verifica che $f(x^{(1)}) = 4.515625 > f(x^{(0)}) = 4$, cioè il passo $s^{(0)}$ è troppo lungo: $\|s^{(0)}\|_2 = 2.85$.

Verifichiamo allora se nel calcolo di $x^{(1)}$ si è ottenuta una *sufficiente* diminuzione della f secondo la regola di Armijo (1.12), con ($\alpha^{(0)} = 1$ e) $\lambda = 0.1$.

Si ha:

$$\begin{aligned} x^{(0)} &= (1,\ 1)^T \quad \Longrightarrow \quad f(x^{(0)}) = 4; \\ x^{(1)} &= (-0.75,\ -1.25)^T \quad \Longrightarrow \quad f(x^{(1)}) = 4.515625. \end{aligned}$$

Poichè: $\alpha^{(0)} = 1$, $p^{(0)} = s^{(0)} = (-1.75,\ -2.25)^T$ e $\lambda = 0.1$, si ottiene

$$\begin{aligned} f(x^{(0)}) + \lambda\ \alpha^{(0)}\ (p^{(0)})^T\ \nabla f(x^{(0)}) &= 4 + 0.1 \times (-1.75,\ -2.25) \begin{pmatrix} 6 \\ 1 \end{pmatrix} \\ &= 4 - (0.1)(12.75) = 2.725. \end{aligned}$$

Quindi $f(x^{(1)}) = 4.515625 > 2.725$ e la regola di Armijo non è soddisfatta (ovviamente).

Prendiamo ora $\alpha^{(0)} = 2^{-1} = 0.5$ (backtracking). Si ha:

$$x^{(1)} = x^{(0)} + \alpha^{(0)}\ s^{(0)} = (0.125,\ -0.125)^T \quad \Longrightarrow \quad f(x^{(1)}) = 0.080078125$$

$$f(x^{(0)}) + \lambda\ \alpha^{(0)}\ (p^{(0)})^T\ \nabla f(x^{(0)}) = 4 - (0.1)(0.5)(12.75) = 3.3625.$$

Dunque $f(x^{(1)}) = 0.080078125 < 3.3625$, la regola di Armijo è soddisfatta e $\alpha^{(0)} = 2^{-1} = 0.5$ è accettabile.

1.7 Metodo di Broyden per sistemi non lineari

Come *Premessa* ai metodi *quasi-Newton* di ottimizzazione non vincolata (in sezione 1.8), vediamo il metodo quasi-Newton di Broyden[20] per risolvere i *sistemi non lineari*.

Si debba risolvere (ad esempio con il metodo di Newton [45, cap.2]) il sistema non lineare

$$F(x) = (f_1(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_n))^T = 0,$$

con $F : R^n \longrightarrow R^m$. Sia $F'(x) = J(x) \in R^{m\times n}$ la matrice jacobiana della F :

$$J(x)_{ij} = \frac{\partial f_i}{\partial x_j}(x) \qquad i = 1, \ldots, m,\ j = 1, \ldots, n.$$

Qui consideriamo il caso $m = n$.

Sia la matrice B un'approssimazione della jacobiana J: $B \approx J$. Per esempio, una approssimazione[21] della $J(x)$ si può ottenere usando le *differenze finite in avanti* è:

$$(b_{ij}(x)) = \frac{f_i(x + h_j e_j) - f_i(x)}{h_j}, \tag{1.21}$$

dove e_j è il j-esimo vettore unitario. In una variabile, questa approssimazione fornisce il classico metodo (quasi-Newton) della *secante variabile* [45, cap.2].

[20] Cfr. [15, cap. 8] , [24] e la tesina [14].

[21] Si noti che in realtà interessa non la jacobiana $J(x)$, ma il prodotto matrice-vettore $J(x)u = \lim_{h\to 0} \dfrac{F(x + hu) - F(x)}{h}$.

In queste approssimazioni, è cruciale la scelta del parametro h_j che eviti disastrose cancellazioni numeriche. Una scelta ragionevole è $h_j = \sqrt{\varepsilon}$, dove ε è la precisione di macchina. Per $\varepsilon \approx 10^{-15},\ h_j \approx 10^{-7}$. Si può tener conto della grandezza della variabile x prendendo $h_j = \sqrt{\varepsilon}|x_j|$. Una ulteriore scelta pratica può essere: $h_j = 100\sqrt{\varepsilon}|x_j|$ per $x_j \neq 0$, $h_j = 100\sqrt{\varepsilon}$ per $x_j = 0$, cfr. [15, pag 97].
Per esercizio, scrivere un programma che calcoli $f'(1),\ f = x^3 + cost$, prendendo $h = 10^{-i},\ i \geq 0$.

Invece, nel metodo quasi-Newton (o metodo secante) di Broyden, l'approssimazione della matrice jacobiana J calcolata alla iterazione k è tale da essere utilizzata per calcolare l'approssimazione della jacobiana alla iterazione successiva.

Il vantaggio di questo metodo rispetto al classico *metodo di Newton*:

$$J_k s_k = -F(x_k), \qquad x_{k+1} = x_k + s_k, \tag{1.22}$$

risiede nel fatto che esso richiede la sola valutazione della F (e non anche la valutazione della jacobiana ottenuta analiticamente o, come in (1.21), con differenze finite[22], che può diventare proibitiva se n è grande).

Pertanto, se la successione delle approssimanti B_k è costruita a partire da una buona approssimazione iniziale B_0 della $J(x_0)$ e $J(\overline{x}) = F'(\overline{x})$ è non singolare, questo metodo presenta degli indubbi vantaggi in termini di costo computazionale rispetto al metodo di Newton (1.22).

Tutti i metodi quasi-Newton (di Broyden) obbediscono alla formula:

$$\boxed{B_k s_k = -F(x_k), \qquad x_{k+1} = x_k + s_k,}$$

dove la matrice $B_k = B(x_k)$ è una approssimazione della matrice jacobiana $J(x_k)$. Può accadere che la B_k (B, da Broyden) non sempre sia una buona approssimazione della J_k, ad ogni iterazione k, ma questo fatto non pregiudica una rapida convergenza del metodo.
In genere B_k è scelta in modo da soddisfare alla cosiddetta *equazione della secante*[23]

[22] O con i moderni metodi di *derivazione automatica AD*; cfr. per esempio il recente toolbox di Matlab ADMAT [12]. Per aumentare l'efficienza del metodo di Newton, ma a scapito del numero di iterazioni, si può valutare la jacobiana non ad ogni passo, ma ad ogni m passi.

[23] In una variabile (dal teorema del valor medio di Lagrange):

$$m_{k+1} = \frac{f_{k+1} - f_k}{x_{k+1} - x_k} = f'(\xi),\ \xi \in (x_k, x_{k+1}),\ m_{k+1} \approx f'_{k+1} \implies f'_{k+1} s_k = f_{k+1} - f_k (= \Delta f_k).$$

$$\boxed{B_{k+1}s_k = y_k, \qquad \text{dove} \qquad y_k = F(x_{k+1}) - F(x_k).} \tag{1.23}$$

L'aggiornamento (update) suggerito da Broyden (1965)[24], consiste in una *correzione di rango uno* di B_k per ottenere B_{k+1} cioè:

$$B_{k+1} = B_k + auv^T,$$

con a uno scalare e u, v vettori. Precisamente:

$$\boxed{B_{k+1} = B_k + \frac{(y_k - B_k s_k)s_k^T}{s_k^T s_k} = B_k + \frac{F(x_{k+1})s_k^T}{s_k^T s_k}} \tag{1.24}$$

essendo $y_k - B_k s_k = y_k + F(x_k) = F(x_{k+1})$. Ovviamente la (1.24) sodisfa alla equazione della secante (1.23).

Spesso $B_0 = J(x_0)$ (non singolare), oppure una sua approssimazione ottenuta con differenze finite in avanti

$$b_{ij}(x_0) = \frac{f_i(x_0 + h_j e_j) - f_i(x_0)}{h_j}.$$

Si noti che in una dimensione l'equazione della secante (1.23) $B_{k+1}s_k = y_k$ specifica *univocamente* il metodo della secante varabile, ma in più dimensioni essa *non* definisce *un'unica* matrice B_k. Infatti, abbiamo un sistema di (solo) n equazioni lineari ma in n^2 incognite che sono gli elementi della matrice (da cui infinite soluzioni).
Questo problema, della unicità della update, fu risolto da Broyden (1965) scegliendo al passo $k+1$ la update B_{k+1} più vicina alla B_k nella *norma di Frobenius* (cfr. [45, sez. 3.1.3]), sotto la condizione che valga l'equazione secante (1.23). In altre parole si ha la:

Proposizione 1.7.1 (Broyden) *Sia* $A \in R^{n\times n}$, $s_k, y_k \in R^n, s_k \neq 0$. *La matrice (1.24)* B_{k+1} *è la soluzione del problema*

$$\min_{B_k \in Q(y,s)} \|A - B_k\|_F, \qquad \textit{sotto la condizione} \quad B_{k+1}s_k = y_k,$$

dove $Q(y,s) = \{B \in R^{n\times n} |\ Bs = y\}$.

Dimostrazione ([15, pag 171, cap.8]).
Sia $B \in Q(y,s)$, con $Q(y,s)$ insieme di matrici (affine e quindi) convesso.

[24] La più usata per problemi densi di piccola o media dimensione. Si noti però che se la B_k è sparsa, la B_{k+1} non lo è.

Allora, per la (1.24) (prendendo le norme di Frobenius ed essendo $Bs_k = y_k$) si ha:

$$\|B_{k+1} - B_k\| = \left\| \frac{(y_k - B_k s_k) s_k^T}{s_k^T s_k} \right\| = \left\| \frac{(B - B_k) s_k s_k^T}{s_k^T s_k} \right\|$$
$$\leq \|(B - B_k)\| \left\| \frac{s_k s_k^T}{s_k^T s_k} \right\| \leq \|(B - B_k)\| ,$$

in virtù delle due condizioni

$$\|AB\| \leq \ \|A\| \cdot \|B\|, \qquad \left\| \frac{s_k s_k^T}{s_k^T s_k} \right\| = 1$$

che discendono dalle proprietà delle norme di matrici (cfr. [15, pag. 45]). La dimostrazione dell'unicità della B_{k+1} si conclude notando che la norma di Frobenius è strettamente convessa (segue immediatamente dalla sua definizione come norma euclidea della matrice pensata come un vettore di $R^{n \times n}$) nell'insieme convesso $Q(y, s)$. c.v.d.

Si dimostra[25] che se x_0 è sufficientemente vicino alla soluzione $\overline{x}$, B_0 sufficientemente vicino a J_0 e $J(\overline{x})$ non singolare, il metodo di Broyden ha convergenza (locale) superlineare, cioè $\|\varepsilon_{k+1}\| = o(\|\varepsilon_k\|)$.

Esso può essere applicato anche ai sistemi lineari $Ax = b$, in cui è la $B \approx A$ che viene aggiornata[26].

Algoritmo (Broyden)

In Input: $F(x),\ x^{(0)},\ B^{(0)} \in R^{n \times n}$ $(= J(x^{(0)})$, Toll

- **1.** Per $k = 0, 1, \ldots$ fino a convergenza
- Risolvere per $s^{(k)}$ il sistema lineare: $B^{(k)} s^{(k)} = -F(x^{(k)})$
  (ad esempio) con fattorizzazione LU o QR, [45, cap. 3.2]
- **2.** $x^{(k+1)} = x^{(k)} + s^{(k)}$
- se $\|s^k\| = \|x^{(k+1)} - x^{(k)}\| \leq Toll$, allora Stop
  (Oppure: se $\|F(x^{(k+1)})\| \leq Toll$, allora Stop)
- altrimenti

[25] Cfr. [15, sezione 2, cap. 8].

[26] Nel caso non simmetrico, recenti versioni si sono dimostrate competitive con i metodi di Krilov (GMRES, ecc....), cfr. [42] e tesina [7].

- $y^{(k)} = F(x^{(k+1)}) - F(x^{(k)})$
- $B^{(k+1)} = B^{(k)} + \dfrac{(y^{(k)} - B^{(k)}s^{(k)})(s^{(k)})^T}{(s^{(k)})^T s^{(k)}} = B^{(k)} + \dfrac{F(x^{k+1})(s^{(k)})^T}{(s^{(k)})^T s^{(k)}}$
- vai a **1.** (fino a $k \leq k_{\max}$)

La soluzione del sistema lineare $Bs = -F$ al punto 2 dell'algoritmo richiede in genere $O(n^3)$ operazioni (se fatta con fattorizzazione LU), ma si può ottenere *anche* con $O(n^2)$ operazioni, usando opportunamente la fattorizzazione QR, cfr. [15, pag.187] e l'implementazione in [35, pag. 382-386] (broydn.for).

Trattandosi di una matrice ottenuta con una correzione di rango uno, è noto che la sua inversa si può calcolare tramite la *formula di Sherman-Morrison* [27], cfr. [15, pag. 188], [13, pag. 940], e quindi *ancora* con $O(n^2)$ operazioni.

Posto $A = B^{(k)},\ u = \dfrac{(y^{(k)} - B^{(k)})s^{(k)})}{(s^{(k)})^T s^{(k)}},\ v = s^{(k)}$, dalla (1.24) si ha:

$$(B^{k+1})^{-1} = (B^k)^{-1} + \frac{(s^k - (B^k)^{-1}y^k)(s^k)^T(B^k)^{-1}}{(s^k)^T(B^k)^{-1}y^k}.$$

Tuttavia l'uso della inversa[28] mal rivela un possibile malcondizionamento. Ma si vedano le osservazioni in [24, pag. 127] (codice: brsol.m).

ESEMPI (da risolvere con Broyden, e anche con Newton)

1.

$$F(x) = \begin{cases} f_1(x_1, x_2) = x_1^2 + x_2^2 - 1 \quad = 0 \\ f_2(x_1, x_2) = x_1^2 - x_2^2 + 0.5 = 0 \end{cases}, \quad J(x) = \begin{pmatrix} 2x_1 & 2x_2 \\ 2x_1 & -2x_2 \end{pmatrix}$$

$$x^0 = (1, 3)^T, \quad B_0 = J(x^0) = \begin{pmatrix} 2 & 6 \\ 2 & -6 \end{pmatrix}.$$

La soluzione del sistema (nel primo quadrante) è $\overline{x} = (0.5, \sqrt{3}/2)^T$, raggiunta in 10 iterazioni (Newton converge in 6 iterazioni). Si trova $x^1 = (0.625, 1.625)^T$, ecc...

[27] Siano: $u,\ v \in R^n,\ A \in R^{n\times n}$ non singolare. Allora $(A + uv^T)^{-1}$ esiste se e solo se $\sigma \equiv 1 + v^T A^{-1} u \neq 0$. Inoltre

$$(A + uv^T)^{-1} = A^{-1} - \frac{1}{\sigma} A^{-1} u v^T A^{-1}.$$

[28] Spesso in letteratura si adotta la notazione $H_k = (B^k)^{-1}$, ma in questa Dispensa si farebbe confusione con la hessiana.

```
Metodo di Broyden per sistemi (risultati ottenuti con Matlab)
Esempio 1:   F:R^2--> R^2
F= (x^2 + y^2 - 1; x^2 - y^2 + 0.5)'
J(x)=(2x  2y; 2x -2y)'
x0 =  [1 3]'
F0 =
    9.0000
   -7.5000

B0=J(x0)
B0 =
     2     6
     2    -6

-----------------          1.o passo:

Risolviamo il sistema:   B0*s0= -F0 ;  s0=-B0\F0
s0 =
   -0.3750
   -1.3750

>> x1=x0+s0
x1 =
    0.6250
    1.6250
------------------------- Valutiamo F1= F(x_1):
>> F1=[2.03125 -1.75]'
F1 =
    2.0313
   -1.7500

Calcolo della norma_2 di s0=x1-x0:
>> t=norm((s0))^2
t =
    2.0313

>> Aggiornamento (update) della 'jacobiana': da B0 a B1
>> B1=B0+(F1*s0')/t
B1 =
    1.6250    4.6250
    2.3231   -4.8154
```

```
-----------------          2.o passo:

Risolviamo il sistema lineare B1*s1= -F1;   s1=-B1\F1
s1 =
   -0.0909
   -0.4073
>> x2=x1+s1
x2 =
    0.5341
    1.2177

Valutiamo F2, calcoliamo la norma di s1 e quindi l'update B2.
Risolviamo poi il sistema lineare B2*s2= -F2: s2=-B2\F2
e quindi: x3=x2+s2.
Ecc..., per   x4,...,x10= (0.5  0.8666)'=(1/2, sqrt(3)/2)'.
(Test di arresto verificato. Newton converge in 6 iterazioni).
```

2.

$$F(x) = \begin{cases} f_1(x_1, x_2) = x_1 + x_2 - 3 = 0 \\ f_2(x_1, x_2) = x_1^2 + x_2^2 - 9 = 0 \end{cases}, \; J(x) = \begin{pmatrix} 1 & 1 \\ 2x_1 & 2x_2 \end{pmatrix}$$

$$x^0 = (1, 5)^T, \quad B_0 = J(x^0) = \begin{pmatrix} 1 & 1 \\ 2 & 5 \end{pmatrix}.$$

Il sistema, come subito si verifica, ha le due le radici $\overline{x} = (0, 3)^T$, e $\overline{x} = (3, 0)^T$. Il metodo converge a $x^1 = (0, 3)^T$, in 7 iterazioni (Newton in 5). Si trova $x^1 = (-0.625, 3.625)^T$, $x^2 \approx (-0.076, 3.076)^T$, ecc...

3.

$$F(x) = \begin{cases} f_1(x_1, x_2) = x_1^2 + x_2^2 - 2 \quad = 0 \\ f_2(x_1, x_2) = e^{x_1 - 1} + x_2^3 - 2 = 0 \end{cases}, \; J(x) = \begin{pmatrix} 2x_1 & 2x_2 \\ e^{x_1 - 1} & 3x_2^2 \end{pmatrix}$$

$$x^0 = (1.5, 2)^T, \quad B_0 = J(x^0) = \begin{pmatrix} 3 & 4 \\ \sqrt{e} & 12 \end{pmatrix}.$$

Il sistema ha la radice $\overline{x} = (1, 1)^T$. Il metodo converge in 11 iterazioni (Newton in 7). Si trova $x^1 \approx (0.8060692, 1.457948)^T$, $x^2 \approx (0.7410741, 1.277067)^T$, ecc...

4.

$$F(x) = \begin{cases} f_1(x) = x_1 + e^{x_1 - 1} + (x_2 + x_3)^2 - 27 = 0 \\ f_2(x) = \dfrac{e^{x_2 - 2}}{x_1} + x_3^2 - 10 = 0 \\ f_3(x) = x_3 + \operatorname{sen}(x_2 - 2) + x_2^2 - 7 = 0 \end{cases}$$
$$x^0 = (4, 4, 4)^T$$

(B_0 è ottenuta con differenze in avanti). Il sistema ha la radice $\overline{x} = (1, 2, 3)^T$. Il metodo converge in 8 iterazioni, con uno scarto (in norma) pari a 1.32×10^{-6}. Si trova:
$x^1 \approx (2.85780400, 2.19477200, 3.35123400)^T, \ldots,$
$x^8 \approx (1.00073800, 1.99976900, 3.0019900)^T.$

5.

$$F(x) = \begin{cases} f_1(x) = 3x_1 - \cos(x_2 x_3) - 0.5 = 0 \\ f_2(x) = x_1^2 - 81(x_2 + 0.1)^2 + \operatorname{sen} x_3 + 1.06 = 0 \\ f_3(x) = e^{-x_1 x_2} + 20x_3 + \dfrac{10\pi - 3}{3} = 0 \end{cases}$$
$$x^0 = (0.1, 0.1, -0.1)^T, \qquad B_0 = J(x^0)$$

Il sistema ha la radice $\overline{x} = (0.5, 0, -0.52359877)^T$. Il metodo converge in 5 iterazioni con uno scarto (in norma) pari a 6.24×10^{-5} (Newton in 4). Si trova:
$x^1 \approx (0.4998693, 1.946693 \times 10^{-2}, -0.5215209)^T, \ldots,$
$x^5 \approx (0.5000002, -1.445223 \times 10^{-6}, -0.5235989)^T.$

1.8 Metodi quasi-Newton (o metodi secanti)

Questi metodi[29] sono anche detti (seguendo la scuola inglese di Powell) metodi a *metrica variabile*.

Poichè

$$\min \{f(x) : x \in R^n\} \implies F(x) = \nabla(f) = 0,$$

si potrebbe procedere come visto per i sistemi non lineari in sezione 1.7, semplicemente "slittando" di una derivata, operando una correzione di rango uno:

$$B^{k+1} = B^k + auv^T,$$

[29] Cfr. [15, cap. 9], [11], ecc...

dove a è uno scalare e u, v vettori $\in R^n$. Ma questa approssimazione:

$$B^{k+1} = B^k + \frac{(y^k - B^k s^k)(s^k)^T}{(s^k)^T s^k},$$

dove $y^k = \nabla f(x^{k+1}) - \nabla f(x^k)$, *non* sarebbe simmetrica, pur essendolo la matrice approssimante B^k ($B \approx H$, H matrice hessiana), a meno che $(y^k - B^k s^k)$ non sia multiplo di s^k.

Per conservare la simmetria, si effettua un processo di simmetrizzazione, e a convergenza si ottiene la update (aggiornamento) di Powell-Broyden (PSB, 1970), [15, formula (9.1.3)], che è una *correzione di rango due*, ossia del tipo:

$$B^{k+1} = B^k + auu^T + bvv^T.$$

La formula PSB obbedisce all'equazione della secante:

$$B^{k+1}s^k = y^k, \quad \text{dove:} \quad s^k = x^{k+1} - x^k, \quad y^k = \nabla f(x^{k+1}) - \nabla f(x^k),$$

ma può *non* essere definita positiva (e quindi la direzione scelta non sarebbe una direzione di discesa).

A tutt'oggi, la formula più efficiente, tra le varie altre proposte[30], e che risolve completamente il problema, è la famosa **BFGS** (1970), cfr. [15, formula (9.2.10)], di rango (al più) due, *simmetrica e definita positiva* in forza dell'ipotesi $y^T s > 0$ (e B^0 simmetrica e definita positiva):

$$B^{k+1} = B^k + \frac{y^k (y^k)^T}{(y^k)^T s^k} - \frac{B^k s^k (s^k)^T B^k}{(s^k)^T B^k s^k} \qquad (1.26)$$

$$= B^k + \frac{y^k (y^k)^T}{(y^k)^T s^k} - \frac{(B^k s^k)(B^k s^k)^T}{(s^k)^T B^k s^k}$$

[30] Ad esempio, la storica DFP (Davidon, 1959), che è in un certo senso la duale della BFGS in quanto è riferita alla approssimazione C della inversa H^{-1} della hessiana, ottenuta tramite la formula di Sherman-Morrison. Le due formule si possono anche ottenere l'una dall'altra per *complementarietà* scambiando y con s e B con C, per cui dalla formula BFGS (1.26) si ottiene la formula DFP (1.25):

$$C^{k+1} = C^k + \frac{s^k (s^k)^T}{(s^k)^T y^k} - \frac{C^k y^k (y^k)^T C^k}{(y^k)^T C^k y^k}, \qquad C^k \approx H^{-1}. \qquad (1.25)$$

dove $B^k \approx H$, matrice hessiana, con le notazioni standard:

$s^k = x^{k+1} - x^k, \quad B^{k+1}s^k = y^k$ (equazione della secante), $y^k = \nabla f(x^{k+1}) - \nabla f(x^k)$.

Come si è detto, la update B^{k+1} è definita positiva:

$$0 < (y^k)^T s^k = (B^{k+1}s^k)^T s^k = (s^k)^T B^{k+1} s^k$$

Anche la formula BFGS ha convergenza superlineare.

Nella sua forma più semplice l'algoritmo BFGS si scrive:

Algoritmo (metodo quasi-Newton BFGS)

In Input: $\epsilon > 0,\ x^0,\ f,\ \nabla f,\ B^0 = I$.

1. $k \longleftarrow 0$
2. while $\|\nabla f(x^k)\| > \epsilon$
   - $B^k s^k = -\nabla f(x^k)$
   - $x^{k+1} = x^k + \alpha^k s^k, \qquad \%\ \ \alpha^k$ calcolato con line-search sulla f
   - $s^k = x^{k+1} - x^k, \qquad y^k = \nabla f(x^{k+1}) - \nabla f(x^k)$
   - $B^{k+1} = B^k + \dfrac{y^k (y^k)^T}{(y^k)^T s^k} - \dfrac{B^k s^k (s^k)^T B^k}{(s^k)^T B^k s^k}$
   - $k \longleftarrow k + 1$
3. fine (while)

In questo contesto, cfr. anche:
Metodi Newton-Inesatti (Dembo et alias,1982); conservazione della sparsità delle update, metodi Inesatti Quasi-Newton in ambiente di calcolo parallelo [43], [46].

ESEMPI (da risolvere con BFGS)

1. (Cfr. esempio n. 6 con Newton)

$$f(x) = f(x_1, x_2) = x_1^4 + x_1 x_2 + (1 + x_2^2), \quad x^0 = (0,0)^T,\ B^0 = I$$

2. (Cfr. esempio n. 4 con Newton)

$$f(x) = f(x_1, x_2) = (x_1 - 2)^4 + (x_1 - 2)^2 x_2^2 + (x_2 + 1)^2,$$
$$x^0 = (1,1)^T, \quad B^0 = H^0.$$

Il punto di minimo $\overline{x} = (2, -1)^T$, è raggiunto in 10 iterazioni (con otto cifre esatte). Per esempio: $x^1 = (1, -0.5)^T$.

3. ("banana" di Rosenbrock, funzione somma di due quadrati)

$$f(x) = f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2, \quad x^0 = (-1.2, 1.0)^T$$

Il punto di minimo $\overline{x} = (1, 1)^T$, è raggiunto in 37 iterazioni (con cinque cifre esatte). Per esempio: $x^1 = (-1.1750, 1.3814)^T$, $x^2 = (-1.1505, 1.3230)^T$.

1.9 Minimi quadrati non lineari

Problema *(best data-fitting)*[31]:

Si cerca di approssimare i dati sperimentali

$$(\mathbf{t_i}, \mathbf{y_i}), \quad \mathbf{1} \leq \mathbf{i} \leq \mathbf{m},$$

tramite una funzione modello $\mathbf{m}(\mathbf{x}, \mathbf{t})$ che è una opportuna funzione *non* lineare nel vettore dei parametri $x = (x_1, \ldots, x_n)^T$, $m \gg n$ (esempio, $m = 100$, $n = 5$).

Si sceglie il vettore dei parametri x in modo che la funzione $f(x)$, somma dei quadrati dei residui (o scarti)[32]

$$f(x) = S(x) = \sum_{i=1}^{m} r_i^2(x) = \sum_{i=1}^{m} [m(x, t_i) - y_i]^2 \tag{1.27}$$

sia minima (in norma l_2; norma di Gauss).

Dunque:

$$\min_{x \in R^n} S(x) = R(x)^T R(x) = \|R(x)\|_2^2 = \sum_{i=1}^{m} r_i^2(x), \tag{1.28}$$

dove $R(x) = (r_1(x), \ldots, r_m(x))^T$ è una funzione da R^n in R^m, $m \geq n$.

Il vettore $x \in R^n$ è il *vettore dei parametri* rispetto ai quali viene fatta la minimizzazione della funzione somma dei residui (1.27)

$$S(x) = \sum_{i=1}^{m} r_i^2(x), \qquad r_i = [m(x, t_i) - y_i],$$

[31] Cfr. [15, cap.10], [11, pag. 2-16], [13, cap.8], [29], ...

[32] Per questa funzione $f(x)$ si adotta la stessa notazione $S(x)$ del capitolo 5 di [45].

con $m(x,t)$ funzione modello non lineare nei parametri x, e $y \in R^m$ è il vettore delle osservazioni.

Nota. Per il *caso lineare* cfr. [45, cap.6]: approssimazione polinomiale ai minimi quadrati, dove i parametri sono indicati con $a_r,\ 1 \le r \le n$, e $t \equiv x$, e le funzioni base sono i monomi $\{x^r\},\ 1 \le r \le n$.
Segue che il modello, lineare nel vettore dei parametri $a = (a_0, a_1, \cdots, a_n)$ è:

$$m(a, x_i) = [a_0 + a_1 x_i + \cdots + a_n x_i^n]$$

.

Sia $J(x) \in R^{m \times n}$

$$J(x)_{ij} = \frac{\partial r_i(x)}{\partial x_j} \qquad 1 \le i \le m, \quad 1 \le j \le n,$$

la matrice jacobiana (derivata prima) della $R(x)$.

Si **noti** che se $m = n$, per ogni soluzione del sistema non lineare

$$F(x) = (f_1, f_2, \ldots, f_n) = 0$$

si ottiene $S(x) = F^T F = \sum_{i=1}^n f_i^2(x) = 0$. Allora (cfr.in sezione 1 e nell'esempio n. 6 di sezione 1.4) la soluzione di un *sistema non lineare* rientra in questo contesto.

Si era inoltre osservato che, tenendo conto della speciale forma della funzione da minimizzare (somma di quadrati), piuttosto che un generico metodo di minimizzazione, è conveniente utilizzare il metodo di Newton.

Calcoliamo dunque il gradiente e l'hessiana della funzione somma dei residui (1.27) $S(x)$.
(Per un esempio, si veda l'**Esempio 1** alla fine di questa sezione).

Il vettore gradiente, $\nabla S(x) \in R^n$ (derivata prima) della funzione somma dei residui $S(x)$ è:

$$S'(x) = \nabla S(x) = 2J(x)^T R(x) = \left(2 \sum_{i=1}^m r_i(x) \nabla r_i(x) \right)_j$$

(In modo impreciso, l'indice in basso sta ad indicare la componente j-esima del vettore, $1 \le j \le n$).

La matrice hessiana, $\nabla^2 S(x) \in R^{n \times n}$ (derivata seconda) della funzione dei residui $S(x)$ è :

$$S''(x) = \nabla^2 S(x) = 2J(x)^T J(x) + 2\left(J^T(x)\right)' R(x)$$
$$= \left(2 \sum_{i=1}^m \nabla r_i(x) \nabla r_i(x)^T + 2 \sum_{i=1}^m r_i(x) \nabla^2 r_i(x) \right)_{jk}$$

(In modo impreciso, gli indici in basso stanno ad indicare l'elemento (j,k) della matrice, $1 \le j,k \le n$).

La *condizione necessaria* per il problema di minimo (1.28) è :

$$S'(x) = \nabla S(x) = 0 \quad \Longrightarrow \quad 2\ J(x)^T R(x) = 0.$$

Utilizzando il **metodo di Newton** (1.22) per risolvere questo sistema non lineare, per il passo (direzione) di Newton $s^k = x^{k+1} - x^k$ si ottiene:

$$\nabla^2 S(x^k)\ s^k = -\nabla S(x^k); \qquad (J^T J + (J^T)' R) s^k = -J^T R\ . \tag{1.29}$$

Se i residui sono piccoli, in (1.29) si può trascurare il termine (costoso da calcolare) delle derivate seconde $(J^T)'R$. Si noti che esso è uguale a zero nel caso lineare.
Si ottiene in tal modo il metodo di *Gauss-Newton* (ancora con convergenza quadratica):

$$\boxed{J(x^k)^T J(x^k) s^k = -J(x^k)^T R(x^k)} \tag{1.30}$$

In questo metodo il passo (direzione) s^k è la soluzione del sistema $(n \times n)$ delle *equazioni normali* (1.30), da risolversi usando la fattorizzazione LL^T di Cholesky della $J^T J$ (o, meglio, la fattorizzazione QR o SVD, cfr. [45, sezione 6.1.3]).

Se la jacobiana J ha rango massimo n (cioè le n funzioni base del modello linearmente indipendenti), si vede subito che il metodo di Gauss-Newton è un metodo di discesa, essendo la hessiana $\nabla^2 S(x) = J(x)^T J(x)$ definita positiva (vale la la condizione(1.6)):

$$(\nabla S)^T s = (J^T R)^T [-(J^T J)^{-1} J^T R] = -(J^T R)^T (J^T J)^{-1})(J^T R) = -y^T M y < 0,$$

avendo posto $y = J^T R,\ M = (J^T J)^{-1}$.

La convergenza può quindi essere accelerata usando una strategia di ricerca monodimensionale, oppure[33] una di "trust region".

Nel primo caso si ottiene il metodo di Gauss-Newton rilassato (*damped*):

$$J(x^k)^T J(x^k)\ s^k = -J(x^k)^T R(x^k), \qquad x^{k+1} = x^k + \alpha^k s^k \tag{1.31}$$

dove α^k è scelto con un procedimento di minimizzazione monodimensionale (linesearch, sezione 1.3):

$$\alpha^{(k)};\ f(x^{(k)} + \alpha^{(k)} s^{(k)}) = \min_{\alpha > 0} \{ f(x^{(k)} + \alpha s^{(k)}) \}.$$

[33] Morè (1977), [15], [29].

Nel secondo caso si ottiene direttamente il classico metodo di *Levenberg (1944)-Marquardt (1963)*:

$$\boxed{\left[J(x^k)^T J(x^k) + \mu^k I\right] s^k = -J(x^k)^T R(x^k)} \tag{1.32}$$

dove $\mu^k \geq 0$ è un parametro *opportunamente* scelto (per esempio con tecniche adattive, cfr. alla Nota 3 a pag. 31 di sezione 1.6).
Se $\mu^k = 0$ si riottiene il metodo di Gauss-Newton, mentre se μ^k è abbastanza grande si ottiene una direzione di massima discesa e quindi s^k si ottiene come una combinazione tra i metodi di Gauss-Newton e del gradiente ($s = -J^T R = -\nabla S$).

NOTA
Il metodo di Levenberg-Marquardt (1.32) può essere visto come un antesignano dei metodi trust-region, si veda [6], [22]. Si tratta infatti di un problema di minimo condizionato (del modello quadratico della S):

$$\min S(\overline{x} + s) = S(\overline{x}) + s^T \nabla S(\overline{x}) + \frac{1}{2} s^T H(\overline{x})\ s, \qquad \frac{1}{2} s^T s \leq h^2.$$

Le equazioni K.K.T. (si veda la sezione 2.2) danno la (1.32).

Infatti, scriviamo la lagrangiana:

$$L(s) = \left\{ s^T \nabla S(\overline{x}) + \frac{1}{2} s^T H(\overline{x})\ s \right\} + \mu \left(\frac{1}{2} s^T s - h^2 \right), \quad \mu \geq 0$$

μ, parametro lagrangiano.

Condizioni KKT:

$$L'(s) = Hs + \nabla S + \mu s = 0$$
$$\mu(\frac{1}{2} s^T s - h^2) = 0; \quad \mu \geq 0 \quad \text{(complementarietà)}$$

Cioè

$$(H + \mu I)s = -\nabla S$$

Ossia

$$(J_k^T J_k + \mu_k I)s_k = -J_k^T R_k$$

(Fine NOTA).

Esempio 1.

Siano $(t_i, y_i),\ 1 \le i \le m = 4$ i dati sperimentali. Sia

$$m(x, t_i) = e^{t_i x_1} + e^{t_i x_2}$$

il modello non lineare nei due parametri x_1, x_2 $(n = 2)$.

Siano inoltre:
$r_i(t_i, x) = [e^{t_i x_1} + e^{t_i x_2} - y_i]$ il residuo i-esimo, e $R(x) = (r_1(x), \ldots, r_m(x))^T$ la funzione (vettore) dei residui.

La matrice jacobiana $J(x) \in R^{m \times n} = R^{4 \times 2}$ della funzione $R(x)$ è :

$$J(x) = \frac{\partial r_i(x)}{\partial x_j} = \begin{pmatrix} t_1 e^{t_1 x_1} & t_1 e^{t_1 x_2} \\ t_2 e^{t_2 x_1} & t_2 e^{t_2 x_2} \\ t_3 e^{t_3 x_1} & t_3 e^{t_3 x_2} \\ t_4 e^{t_4 x_1} & t_4 e^{t_4 x_2} \end{pmatrix}.$$

Il vettore gradiente $\nabla S(x) \in R^n = R^2$ della funzione da minimizzare $S(x) = \|R(x)\|_2^2 = \sum_{i=1}^m r_i^2(x)$ è :

$$\begin{aligned} \nabla S(x) &= 2J(x)^T R(x) = 2 \sum_{i=1}^{4} r_i(x) \nabla r_i(x) \\ &= \begin{pmatrix} 2 \sum_{i=1}^4 r_i(x) \frac{\partial r_i}{\partial x_1} \\ \\ 2 \sum_{i=1}^4 r_i(x) \frac{\partial r_i}{\partial x_2} \end{pmatrix} = \begin{pmatrix} 2 \sum_{i=1}^4 r_i(x)\ t_i e^{t_i x_1} \\ \\ 2 \sum_{i=1}^4 r_i(x)\ t_i e^{t_i x_2} \end{pmatrix}. \end{aligned}$$

Poichè il vettore gradiente e la matrice hessiana del residuo i-esimo sono:

$$\nabla r_i(x) = \begin{pmatrix} t_i e^{t_i x_1} \\ t_i e^{t_i x_2} \end{pmatrix}, \quad \nabla^2 r_i(x) = \begin{pmatrix} t_i^2 e^{t_i x_1} & 0 \\ 0 & t_i^2 e^{t_i x_2} \end{pmatrix},$$

la matrice hessiana $\nabla^2 S(x) \in R^{n \times n} = R^{2 \times 2}$ della funzione $S(x)$ è :

$$\nabla^2 S(x) = 2 \begin{pmatrix} \sum_{i=1}^4 \left(t_i e^{t_i x_1}\right)^2 + \sum_{i=1}^4 t_i^2 r_i(x) e^{t_i x_1} & \sum_{i=1}^4 t_i^2 e^{t_i (x_1 + x_2)} \\ \\ \sum_{i=1}^4 t_i^2 e^{t_i (x_1 + x_2)} & \sum_{i=1}^4 \left(t_i e^{t_i x_2}\right)^2 + t_i^2 r_i(x) e^{t_i x_2} \end{pmatrix}$$

ossia

$$\nabla^2 S(x) = 2 \begin{pmatrix} \sum_{i=1}^4 t_i^2 e^{t_i x_1} (r_i(x) + e^{t_i x_1}) & \sum_{i=1}^4 t_i^2 e^{t_i (x_1 + x_2)} \\ \\ \sum_{i=1}^4 t_i^2 e^{t_i (x_1 + x_2)} & \sum_{i=1}^4 t_i^2 e^{t_i x_2} (r_i(x) + e^{t_i x_2}) \end{pmatrix}.$$

Siamo ora in grado di applicare il metodo di Newton o di Gauss-Newton.

Risolviamo il seguente caso particolare. Siano $(t_i, y_i) = (i, 2+2i), 1 \leq i \leq 10$, i dati.

t_i	1	2	3	4	5	6	7	8	9	10
y_i	4	6	8	10	12	14	16	18	20	22

La routine *lsqnonlin.m* di Matlab [11] dopo 23 iterazioni fornisce:

$$x = (0.2578 \quad 0.2578)^T, \qquad S(x) = \sum_{i=1}^{10} r_i^2(x) = 124.3622.$$

```
------------------------
 function f=funz(x)
 t=[1:10];
 f=(2 + 2*t) - (exp(t*x(1)) + exp(t*x(2)));
------------------------
 Per eseguire:
 x0=[0.3 0.4];
 [x,resnorm,residual,exitflag,output]=lsqnonlin('funz',x0)
```

In Figura 1.8 sono riportati i dati $(t_i, y_i = 2 + 2i)$, contro i risultati (t_i, m_i) ottenuti.

1.10 Esercitazione

APPLICAZIONE DEL METODO DEI MINIMI QUADRATI NON LINEARI AI RISULTATI SPERIMENTALI IN UNA GALLERIA DEL VENTO

I risultati degli esperimenti in una galleria del vento sul flusso d'aria sull'ala di un aeroplano sono i seguenti:

```
R/C:
0.73 0.780  0.81  0.86  0.875  0.89   0.95   1.02  1.03  1.055 1.135
1.14 1.245  1.32  1.385 1.43   1.445  1.535  1.57  1.63  1.755

V_teta/V_inf:
0.0788 0.0788 0.0640 0.0788 0.0681 0.0703 0.0703 0.0681 0.0681 0.079 0.0575
0.0681 0.0575 0.0511 0.0575 0.0490 0.0532 0.0511 0.0490 0.0532 0.0426
```

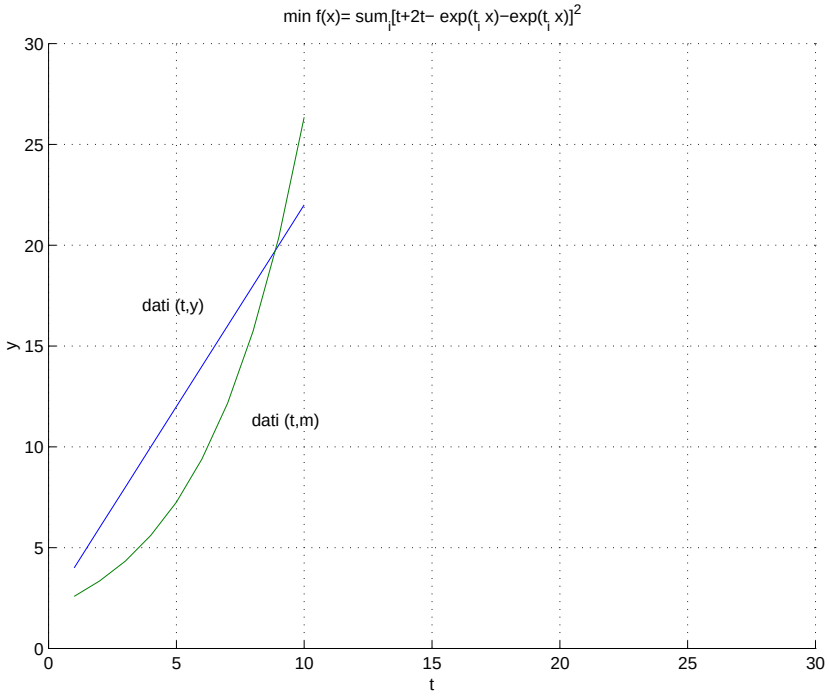


Figura 1.8: Minimi quadrati non lineari (Esempio 1).

dove R è la distanza dal centro del vortice, C è la corda dell'ala, V_θ è la velocità tangenziale del vortice e V_∞ è la velocità del flusso indisturbato.

Poniamo:

$$t = R/C \qquad\qquad y = V_\theta/V_\infty.$$

Si approssimino i dati sperimentali con il seguente modello non lineare nei due parametri $(A,\ \lambda)$, con 21 osservazioni (t_i, y_i) :

$$m(A, \lambda; t_i) = \frac{A}{t_i}\left(1 - \exp\left(-\lambda t_i^2\right)\right), \qquad\qquad 1 \ \leq i \ \leq 21$$

SOLUZIONE

Funzione (dei residui) da minimizzare:

$$S(A,\lambda) = \sum_{i=1}^{21} r_i^2(A,\lambda) = \sum_{i=1}^{21} \left(y_i - m(A,\lambda;t_i)\right)^2 = \sum_{i=1}^{21} \left(y_i \ - \ \frac{A}{t_i}\left(1 - \exp\left(-\lambda t_i^2\right)\right)\right)^2$$

Condizioni (necessarie) di minimo:

$$\nabla S(A,\lambda) = 0 \quad \Longleftrightarrow \quad \begin{cases} S'(A,\lambda)_A = 0 \\ S'(A,\lambda)_\lambda = 0 \end{cases}$$

Cioè:

$$\sum_{i=1}^{21} \left(\frac{1}{t_i}\right) \left(1 - \exp\left(-\lambda t_i^2\right)\right) \left(y_i - \frac{A}{t_i}(1 - \exp\left(-\lambda t_i^2\right))\right) = 0$$

$$\sum_{i=1}^{21} (t_i)(\exp\left(-\lambda t_i^2\right)) \left(y_i - \frac{A}{t_i}(1 - \exp\left(-\lambda t_i^2\right))\right) = 0$$

Sistema non lineare da risolvere (con Newton, Gauss-Newton, Levenberg-Marquardt...).

Si ottiene la soluzione: $A = 7.67175E - 002, \quad \lambda = 2.33896E + 00$

Capitolo 2

Ottimizzazione vincolata

2.1 Vincoli di uguaglianza (Lagrange)

Il classico problema della *Ottimizzazione vincolata* con vincoli di *uguaglianza,* si può così formulare:

$$\min\ f(x), \qquad x \in \Omega \subset R^n \qquad \Omega = \{x \in R^n;\ \varphi(x) = 0\} \tag{2.1}$$

dove $f : R^n \longrightarrow R$ e $\varphi = (\varphi_1(x), \ldots, \varphi_m(x))^T$ è una funzione da R^n in R^m, $m < n$.

Come è noto, sotto ipotesi di differenziabilità della f e dei vincoli φ_i, il problema si risolve tramite la regola dei *moltiplicatori di Lagrange* (1761) che trasforma il problema vincolato in uno non vincolato (libero).

Precisamente, se la matrice jacobiana

$$J(\overline{x}) = \frac{\partial(\varphi_1, \ldots, \varphi_m)}{\partial(x_1, \ldots, x_n)} = \begin{pmatrix} (\varphi_1)_{x_1} \ldots (\varphi_1)_{x_n} \\ \vdots \\ (\varphi_m)_{x_1} \ldots (\varphi_m)_{x_n} \end{pmatrix}$$

ha rango m (in $x = \overline{x}$), ossia se i gradienti degli m vincoli sono linearmente indipendenti (in tal caso esiste il piano tangente e il punto $\overline{x}$ si dice *regolare*), tramite il teorema (di Dini) sulle funzioni implicite si ottiene [1] che (condizione necessaria):

in un punto di estremo (minimo o massimo) $\overline{x}$ il gradiente $\nabla f(\overline{x})$ è ortogonale al sottospazio tangente in $\overline{x}$ ad Ω (Ω è una varietà) e dunque esiste

[1] Cfr. [20, sezione 18.5, Vol. 1].

un vettore (dei moltiplicatori) $\overline{\lambda} = (\overline{\lambda_1}, \ldots, \overline{\lambda_m})^T \in R^m$ *tale che*

$$\nabla f(\overline{x}) = -(\overline{\lambda})^T \nabla\varphi(\overline{x}) = 0, \quad \overline{\lambda} = (\overline{\lambda_1}, \ldots, \overline{\lambda_m})^T \neq 0 \quad (2.2)$$

$$\text{(cioè)} \qquad \nabla f(\overline{x}) + \sum_{i=1}^{m} \overline{\lambda_i}\, \nabla\varphi_i(\overline{x}) = 0, \quad \overline{\lambda} = (\overline{\lambda_1}, \ldots, \overline{\lambda_m})^T \neq 0.$$

Ossia, $\nabla f(\overline{x})$ è una combinazione lineare dei gradienti $\varphi(x)$ in $\overline{x}$.
Nel caso di un solo vincolo, si veda in Figura 2.1 l'interpretazione geometrica dell'Esempio 1 ($n = 2, m = 1$): parallelismo dei due gradienti ∇f e $\nabla\varphi$.

Equivalentemente, introdotta la *funzione lagrangiana*

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i\, \varphi_i(x), \quad (2.3)$$

si ha che la coppia $(\overline{x}, \overline{\lambda})$ soddisfa il sistema (in generale non lineare) di $n+m$ equazioni nelle n incognite x_i e nelle m incognite λ_i

$$\nabla_x L(x, \lambda) = \nabla f(x) + \sum_{i=1}^{m} \lambda_i\, \nabla\varphi_i(x) = 0, \quad (2.4)$$

$$\nabla_\lambda L(x, \lambda) = \varphi(x) = 0 \quad (2.5)$$

Si noti che l'equazione (2.4) corrisponde alla (2.2) e la (2.5) coincide con la condizione vincolare $\varphi(x) = 0$.
Queste condizioni (del primo ordine) sono in generale solo *necessarie*. Si veda il controesempio $f = x(y-1) - y^3$ sotto il vincolo ($m = 1$) $\Omega = \{(x, y) \in R^2 : x = 0\}$. Nell'origine $(0, 0)$ la regola dei moltiplicatori è soddisfatta (con $\lambda = 1$), ma essa non è un minimo per la f, come si può facilmente verificare (la $f/_{(x=0)} = -y^3$ ha un flesso nell'origine).

Condizioni (necessarie) sufficienti del *secondo ordine* per l'esistenza di un minimo si possono ottenere se[2] le funzioni sono di classe C^2, sotto ipotesi di (semidefinitezza) definitezza positiva della matrice hessiana $\nabla^2 L_{xx}$ della lagrangiana sul sottospazio tangente $M = \{h : \nabla\varphi(\overline{x})\ h = 0\}$. Si veda il successivo Esempio 2.

In forma estesa, il sistema (2.4) di $n + m$ equazioni si scrive

$$\begin{cases} f_{x_1}(x) + \lambda_1(\varphi_1(x))_{x_1} + \cdots + \lambda_m(\varphi_m(x))_{x_1} = 0 \\ \qquad\qquad\qquad\qquad \vdots \qquad\qquad\qquad\qquad \vdots \\ f_{x_n}(x) + \lambda_1(\varphi_1(x))_{x_n} + \cdots + \lambda_m(\varphi_m(x))_{x_n} = 0 \\ \\ \qquad\qquad \varphi_1(x) = 0, \ldots, \varphi_m(x) = 0 \end{cases}$$

[2]Cfr. [27, sezione 10.5].

Riscriviamo le prime n equazioni sotto forma matriciale

$$\nabla f + J^T\lambda = \begin{pmatrix} f_{x_1}(x) \\ \vdots \\ f_{x_n}(x) \end{pmatrix} + \begin{pmatrix} (\varphi_1)_{x_1} \dots (\varphi_m)_{x_1} \\ \vdots \\ (\varphi_1)_{x_n} \dots (\varphi_m)_{x_n} \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{pmatrix} = 0.$$

Si noti che ad intervenire è la matrice *trasposta* J^T della matrice jacobiana J della funzione vettoriale dei vincoli $\varphi(x)$.

Esempio 1 (in R^2):

$$\min\ f = x + y \quad \text{sotto il vincolo} \quad \varphi(x,y) = x^2 + y^2 - 1 = 0,$$

$$\text{ossia:} \quad \Omega = \{(x,y) \in R^2;\ x^2 + y^2 - 1 = 0\} \quad (m = 1).$$

un punto di minimo e di massimo (assoluti).

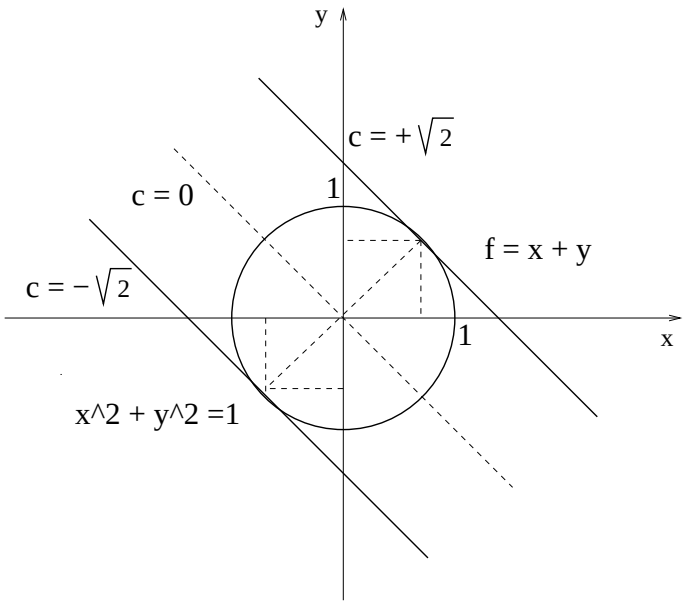


Figura 2.1: Moltiplicatori di Lagrange (in R^2).

Lagrangiana: $L(x,y;\lambda) = x + y + \lambda(x^2 + y^2 - 1)$.
Le condizioni necessarie di estremo vincolato (2.4) sono:

$$L_x = 1 + 2\lambda x = 0,\ L_y = 1 + 2\lambda y = 0,\ L_\lambda = x^2 + y^2 - 1 = 0.$$

Si trovano le due soluzioni

$$(\overline{x},\ \overline{y},\ \overline{\lambda}) : \quad (\frac{-1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}) \quad \text{e} \quad (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}),$$

con $f(\overline{x},\overline{y}) = -\sqrt{2}$ e $\sqrt{2}$, rispettivamente. Si può facilmente constatare (calcolare l'hessiana della lagrangiana L) che le due soluzioni corrispondono a un punto di minimo e di massimo (assoluti).
Si noti in Figura 2.1, nei due punti, la tangenza fra vincolo φ e curve di (livello) $f = c$ ($c =$ costante), per $c = \pm\sqrt{2}$, e quindi il parallelismo dei loro gradienti. Cioè:

$$\nabla f = -\lambda\nabla\varphi, \quad \nabla f + \lambda\nabla\varphi = \nabla\left(f + \lambda\nabla\varphi\right) = 0, \qquad L(x,y,\lambda) = f + \lambda\varphi.$$

$$\nabla f = \begin{pmatrix}1\\1\end{pmatrix}, \quad \nabla\varphi = \begin{pmatrix}2\overline{x}\\2\overline{y}\end{pmatrix} = \begin{pmatrix}-\sqrt{2}\\-\sqrt{2}\end{pmatrix} \implies \begin{pmatrix}1\\1\end{pmatrix} = -\overline{\lambda}\begin{pmatrix}-\sqrt{2}\\-\sqrt{2}\end{pmatrix},\ \overline{\lambda} = \frac{1}{\sqrt{2}}.$$

$$\nabla f = \begin{pmatrix}1\\1\end{pmatrix}, \quad \nabla\varphi = \begin{pmatrix}2\overline{x}\\2\overline{y}\end{pmatrix} = \begin{pmatrix}\sqrt{2}\\\sqrt{2}\end{pmatrix} \implies \begin{pmatrix}1\\1\end{pmatrix} = -\overline{\lambda}\begin{pmatrix}\sqrt{2}\\\sqrt{2}\end{pmatrix},\ \overline{\lambda} = -\frac{1}{\sqrt{2}}.$$

Esempio 2 (in R^3):

$$\min\ f = xy + yz + xz \quad \text{sotto il vincolo} \quad x + y + z = 3.$$

Lagrangiana: $L(x,y,z;\lambda) = (xy + xz + yz) + \lambda(x + y + z - 3)$
Le condizioni necessarie di estremo vincolato (2.4) sono

$$\begin{array}{rrrr} & y & +z & +\lambda = 0 \\ x & & +z & +\lambda = 0 \\ x & +y & & +\lambda = 0 \\ x & +y & +z & -3 = 0 \end{array}$$

La soluzione è $x = y = z = 1,\ \lambda = -2$. Considerando la matrice hessiana della lagrangiana L, con derivate seconde calcolate *rispetto* a $x,\ y$ e z, si trova

$$\nabla^2 L = \begin{pmatrix}0 & 1 & 1\\ 1 & 0 & 1\\ 1 & 1 & 0\end{pmatrix}$$

che è indefinita (gli autovalori sono $\{-1,-1,2\}$). Considerando il sottospazio (piano tangente)

$$\{h : \nabla\varphi(\overline{x})\ h = (1\ 1\ 1)(h_1\ h_2\ h_3)^T = h_1 + h_2 + h_3 = 0\}, \quad (h_1,\ h_2,\ h_3)^T \neq 0,$$

da cui $(h_2 + h_3) = -h_1,\ (h_1 + h_3) = -h_2,\ (h_1 + h_2) = -h_3$, si ottiene

$$h^T\ \nabla^2 L\ h = h_1(h_2 + h_3) + h_2(h_1 + h_3) + h_3(h_1 + h_2) = -(h_1^2 + h_2^2 + h_3^2).$$

Si può allora concludere che, almeno sul sottospazio tangente visto, l'hessiana $\nabla^2 L$ è definita negativa e quindi la soluzione trovata è un massimo locale.

Esempio 3 (Programmazione quadratica):
Trovare i punti di massimo e di minimo della funzione *quadratica*

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c,$$

sotto il vincolo *lineare*

$$\Omega = \{x \in R^n;\ Cx - d = 0\},$$

dove C è una matrice $m \times n$ di rango m e $d \in R^m, m < n$.

Lagrangiana:

$$L(x, \lambda) = \left(\frac{1}{2}x^T Ax - b^T x + c\right) + \lambda(Cx - d),\ x \in R^n,\ \lambda \in R^m.$$

Le condizioni di Lagrange danno il sistema *lineare* (aumentato)

$$L_x = Ax - b + C^T\lambda = 0,\ L_\lambda = Cx - d = 0,$$

che in forma matriciale compatta si scrive

$$\begin{cases} Ax + C^T\lambda = b \\ Cx \quad\ = d \end{cases} \Longleftrightarrow \begin{pmatrix} A & C^T \\ C & 0 \end{pmatrix}\begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} b \\ d \end{pmatrix}, \tag{2.6}$$

con matrice (simmetrica ma) *indefinita*.
Il problema vincolato (vincolo: $Cx - d = 0$) è stato trasformato in uno non vincolato ma a costo di un *aumento* (di m equazioni) del sistema $\nabla f(x) = 0$.

Si dimostra[3]che il sistema è unicamente risolubile se la matrice jacobiana dei vincoli C ha rango massimo m e la matrice A è definita positiva sullo spazio tangente dei vincoli, cioè: $x^T A\ x > 0$ per ogni $x \neq 0$ tale che $Cx = 0$.

Per risolvere in modo numericamente efficientemente il sistema (2.6) di Lagrange, si usano sia *metodi diretti*, con fattorizzazione triangolare di Gauss e sostituzioni in avanti e indietro (Cholesky non è applicabile), sia *metodi iterativi*, opportunamente precondizionati[4].

Il precedente esempio 2 è un esempio di programmazione quadratica dove $A = \nabla^2 L(x)$, C la matrice (1 1 1), $b = c = 0,\ d = 3$.

[3]Cfr. [34, Lemma 16.1].
[4]Cfr. [34, cap. 16] e [4].

2.2 Vincoli di disuguaglianza (Kuhn-Tucker)

Il problema della *Ottimizzazione vincolata* con vincoli di *disuguaglianza,* si può così formulare:

$$\min\ f(x), \qquad x \in \Omega \subset R^n \qquad \Omega = \{x \in R^n;\ \varphi(x) \leq 0\} \tag{2.7}$$

dove $f : R^n \longrightarrow R$ e $\varphi = (\varphi_1(x), \ldots, \varphi_m(x))^T$ è una funzione da R^n in R^m. Qui non si richiede più $m < n$.

Si possono distinguere i due tipi di vincoli (uguaglianza e disuguaglianza)

$$\varphi(x) = (\varphi_1(x), \ldots, \varphi_p(x))^T = 0, \qquad \varphi(x) = (\varphi_{p+1}(x), \ldots, \varphi_m(x))^T \leq 0.$$

Si noti che in bibliografia i vincoli vengono espressi anche nella forma $\Omega = \{x \in R^n;\ \varphi(x) \geq 0\}$. Senza cambiare notazioni, per riportarsi alla forma in (2.7), basta porre ovunque $[-\varphi(x)]$ al posto di $\varphi(x)$.

Premettiamo il seguente risultato di analisi convessa sulle disequazioni lineari.

Definizioni: Siano: $A \in R^{m\times n},\ x \in R^n$.
Cono convesso C: $C = \{x; \quad Ax \leq 0\}$.
Cono convesso duale di C: $C' = \{y; \quad y = A^T u, \quad u \in R^m, u \geq 0\}$.

Lemma di Farkas (1901)[13, pag. 483]:

$$\begin{aligned} &\text{Sia}: \quad A \in R^{m\times n}, \quad x \in R^n, \quad p \in R^n. \qquad \textit{Allora}: \\ &\{p^T x \ \leq 0, \quad \forall x, \quad Ax \leq 0\} \quad \Longleftrightarrow \quad \{p \in C'\} \quad \textit{cioè} \\ &\{\exists\ u \in R^m, \quad u \geq 0; \quad p = A^T u\}. \end{aligned}$$

Interpretazione geometrica del Lemma (Figura 2.2):

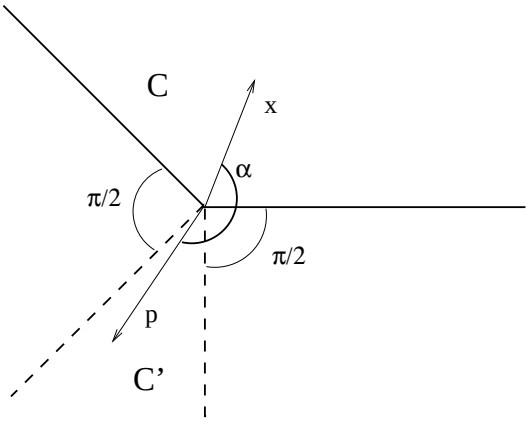


Figura 2.2: Lemma di Farkas: interpretazione geometrica (in R^2).

perchè un vettore qualunque $p \in R^n$ non formi un angolo acuto con alcun vettore x del cono convesso C, bisogna che p appartenga al cono convesso C' duale di C.

Nota. Il Lemma di Farkas è un'estensione alle disuguaglianze del classico risultato (*di alternativa*) sui sistemi lineari. Sotto forma di alternativa, il Lemma di Farkas si enuncia così:

$$\begin{aligned} &\text{o} \qquad \exists\, u \in R^m; \quad p = A^T u, \quad u \geq 0 \\ &\text{o} \qquad \exists\, x \in R^n; \quad Ax \leq 0, \quad p^T x > 0. \end{aligned}$$

In analogia con il problema (2.1) associamo al problema (2.7) la funzione *lagrangiana*

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i \, \varphi_i(x), \quad x \in R^n, \quad \lambda \in R^m_+ \tag{2.8}$$

(dove λ è il vettore dei moltiplicatori).

In ipotesi di differenziabilità della f e delle φ_i, come conseguenza del lemma di separabilità di Farkas [13, pag. 514], in un punto di minimo vincolato $\overline{x}$, *oltre ai vincoli* $\varphi_i(x) \leq 0,\ i = 1, \ldots, m$, sussiste la *relazione*

$$\boxed{\nabla f(\overline{x}) = -\overline{\lambda}^T \, \nabla\varphi(\overline{x})} = -\sum_{i=1}^{m} \overline{\lambda_i} \, \nabla\varphi_i(\overline{x}), \tag{2.9}$$

$$\text{dove:} \quad \overline{\lambda} = (\overline{\lambda_1}, \ldots, \overline{\lambda_m})^T \geq 0,$$

insieme con la *condizione di complementarietà*: $\overline{\lambda}^T \varphi(\overline{x}) = 0$, cioè

$$\overline{\lambda}^T \nabla_\lambda L(\overline{x}, \overline{\lambda}) = \overline{\lambda}^T \varphi(\overline{x}) = \sum_{i=1}^{m} (\overline{\lambda_i}) \ \varphi_i(\overline{x}) = 0 \tag{2.10}$$

dove $\overline{\lambda_i} = 0$ per $i \notin I(x)$, essendo $I(x) = \{i : 1 \leq i \leq m;\ \varphi_i(\overline{x}) = 0\}$.

Premettiamo le **definizioni** seguenti:

- Un punto $\overline{x}$ si dice ammissibile[5] se verifica tutti i vincoli, cioè $\overline{x} \in \Omega$.
- Un vincolo di disuguaglianza $\varphi_i(x) \leq 0$ è detto *attivo* nel punto ammissibile $\overline{x}$ se $\varphi_i(\overline{x}) = 0$ ed *inattivo* nel punto $\overline{x}$ se $\varphi_i(\overline{x}) < 0$.

Per convenzione, un vincolo di uguaglianza $\varphi_i(x) = 0$ è *attivo* in ogni x.

[5] Feasible, in inglese.

- Un punto ammissibile $\overline{x}$ è detto *regolare* se i gradienti $\nabla\varphi_i(\overline{x})$ dei vincoli *attivi* sono *linearmente indipendenti* (*ipotesi dei vincoli qualificati*).

Se distinguiamo i vincoli φ_i fra vincoli (attivi) di uguaglianza $h_i(x),\ h_i(x) = 0$ per $1 \leq i \leq p$, e di disuguaglianza $g_j(x)$, $j \in J$, dove J è l'insieme degli indici j per cui $g_j(x) = 0$, allora si ha che:

- i vincoli sono *qualificati* se i gradienti $\nabla h_i(\overline{x})$, per $1 \leq i \leq p$, e $\nabla g_j(\overline{x})$, per $j \in J$, sono linearmente indipendenti.

Per la definizione di vincoli attivi, la relazione (2.9) vale per *tutti* gli $i, 1 \leq i \leq m$, considerando vincoli attivi e non attivi, ma (può essere) $\lambda_i \neq 0$ *solo* se il corrispondente vincolo è attivo.

Per vincoli qualificati, in un punto di minimo valgono dunque le **condizioni**

$$\nabla_x L(\overline{x}, \overline{\lambda}) = 0 \tag{2.11}$$

$$\nabla_\lambda L(\overline{x}, \overline{\lambda}) = \varphi(\overline{x}) \leq 0 \tag{2.12}$$

$$\overline{\lambda} \geq 0 \tag{2.13}$$

$$\overline{\lambda}^T \varphi(\overline{x}) = 0 \tag{2.14}$$

Le (2.11)—(2.14) esprimono le condizioni KKT (necessarie, del primo ordine) di *Kuhn-Tucker* (1951, dette anche di Karush-Kuhn-Tucker).

Si osservi che la (2.9) coincide con la (2.11):

$$\nabla_x L(x, \lambda) = \nabla f(x) + \lambda^T \nabla\varphi(x) = \nabla f(x) + \sum_{i=1}^{m} \lambda_i\ \nabla\varphi_i(x) = 0,$$

e la condizione di complementarietà (2.10) coincide con la (2.14).

Inoltre, tenuto conto del segno dei parametri λ e dei vincoli φ (condizioni (2.12) e (2.13)), essa equivale alle m condizioni

$$\overline{\lambda_i}\ \varphi_i(\overline{x}) = 0, \qquad 1 \leq i \leq m.$$

Esse implicano che o $\overline{\lambda_i} = 0$ o $\varphi_i(\overline{x}) = 0$.
Cioè: le condizioni (di complementarietà) (2.10) equivalgono al fatto che, come si è detto, λ_i può essere positivo *solo se* il corrispondente vincolo è attivo. Banalmente, essendo $\lambda \geq 0$ e $\varphi \leq 0$, allora λ può essere diverso da zero *solo se* il corrispondente vincolo è uguale a zero.

Richiamiamo le **Definizioni**:

- *Programmazione convessa*: f convessa, vincoli di uguaglianza lineari, vincoli di disuguaglianza convessi (da cui segue che Ω è convesso)

- (In particolare, la importante)
  Programmazione quadratica: f quadratica e vincoli lineari.
  Nella forma standard[6] essa si scrive come:

$$\begin{aligned} &\min \ \tfrac{1}{2}\, x^T Q x + c^T x \\ &\text{tale che} \quad Ax = b, \\ &\qquad x \geq 0, \end{aligned} \tag{2.15}$$

  dove $Q \in \mathcal{R}^{n\times n}$ è una matrice semidefinita positiva, $A \in \mathcal{R}^{m\times n}$ è la matrice, di rango massimo m, dei vincoli lineari e i vettori x, $c \in R^n$ e $b \in R^m$.

  Inoltre si ha che, essendo i vincoli *lineari,* l'ipotesi di qualificazione dei vincoli (loro lineare indipendenza) è automaticamente soddisfatta.

- *Programmazione lineare*: f lineare e vincoli lineari (caso particolare di programmazione quadratica-convessa con matrice $Q = 0$).

Sotto ipotesi di definitezza della matrice hessiana $\nabla^2 L_{xx}$, cfr. [34, teoremi 12.5 e 12.6], le (2.11)—(2.14) diventano condizioni *necessarie* e condizioni *sufficienti* del *secondo ordine* affinchè il punto $\overline{x}$ sia di minimo vincolato.

Se il problema è un problema di programmazione convessa, le condizioni KKT diventano condizioni necessarie e sufficienti, tutti i punti di minimo locali sono di minimo globale e l'insieme di questi minimi è un insieme convesso.

In modo approssimativo, si può dire che le relazioni di Kuhn-Tucker sono, rispetto ai vincoli di disuguaglianza, quello che i moltiplicatori di Lagrange sono rispetto ai vincoli di uguaglianza.

[6] Ad essa ci si può sempre ricondurrre. Per esempio, i vincoli $Cx \leq d$ o $Cx \geq d$ si trasformano in vincoli di uguaglianza introducendo le *variabili slack* (ausiliarie) z, non negative, ottenendo:

$$Cx + z = d, \qquad \text{oppure} \qquad Cx - z = d, \qquad z \geq 0.$$

Se per qualche componente risulta $x_i \geq l_i$, l_i = costante, basta porre $x_i{}' = x_i - l_i \geq 0$. Se qualche componente x_i è illimitata, basta considerare la parte positiva e negativa, in quanto:

$$x_i = x_i^+ - x_i^-, \quad x_i^+ \geq 0,\ x_i^- \geq 0; \qquad x^+ = \frac{|x| + x}{2}\ , \quad x^- = \frac{|x| - x}{2}\ .$$

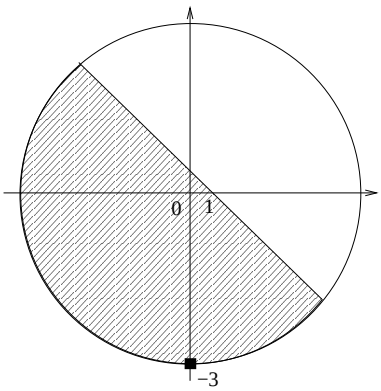


Figura 2.3: Esempio 1.

I punti di estremo si possono anche caratterizzare come punti di *sella* della lagrangiana e ricorrendo alla teoria della *dualità*[7] (dei due problemi, *primale* che opera nello spazio $(n-m)$-dimensionale e *duale* che opera nello spazio m-dimensionale dei moltiplicatori di Lagrange); ma su questo non insistiamo.

Esempio 1 (in R^2) (Tema: 08/04/2005)

$$\min\ f(x,y) = x^2+y, \qquad \Omega = \{(x,y) \in R^2;\ x^2+y^2-9 \le 0; \quad x+y-1 \le 0\}.$$

Nota 1.
Per questo e per gli altri esempi, si consiglia di fare le figure.
Si trovi anche il massimo, ricordando che $\max(f) = -\min(-f)$. Si noti che il minimo e il massimo esistono (in Ω compatto) in forza del teorema di Weierstrass sulle funzioni continue.
Si controllino inoltre le direzioni dei gradienti della funzione obiettivo f e dei vincoli φ nei punti ottimali, verificando che siano soddisfatte la condizione di discesa (1.6) e la relazione (2.9)[8]: $-\nabla f(\overline{x}) = \sum_{i=1}^{m} \overline{\lambda_i}\ \nabla\varphi_i(\overline{x})$.
(Fine Nota 1).

Lagrangiana:

$$L(x,y,\lambda_1,\lambda_2) = f(x)+\lambda_1\varphi_1+\lambda_2\varphi_2 = x^2+y+\lambda_1(x^2+y^2-9)+\lambda_2(x+y-1)$$

[7]Si veda anche in programmazione lineare, sezione 2.5.

[8]Geometricamente: il vettore $\overline{x}$ è punto KKT se e solo se $-\nabla f(\overline{x})$ appartiene al cono convesso generato dai gradienti dei vincoli attivi (bindings).

Le condizioni necessarie del primo ordine di K.-Kuhn-Tucker sono:

$$\begin{array}{rrrr} L_x = 2x & +2\lambda_1 x & +\lambda_2 & = 0 \\ L_y = 1 & +2\lambda_1 y & +\lambda_2 & = 0 \\ & & \varphi_1 = x^2 + y^2 - 9 & \leq 0 \\ & & \varphi_2 = x + y - 1 & \leq 0 \\ & & \lambda_1 \quad \geq 0, \quad \lambda_2 & \geq 0 \\ & & \lambda_1\varphi_1 = \lambda_1(x^2 + y^2 - 9) & = 0 \\ & & \lambda_2\varphi_2 = \lambda_2(x + y - 1) & = 0 \end{array}$$

Nota 2 .
Cercheremo la soluzione per le varie combinazioni[9] di *vincoli attivi*, controllando quindi il segno dei moltiplicatori di Lagrange λ_i ottenuti. In tal modo, nel sistema aumentato compaiono solo delle *uguaglianze*, ed esso può essere risolto con un qualunque metodo numericamente efficiente.
Nei seguenti semplici esempi, troveremo soluzioni analitiche; inoltre, essendo al più $m = 2$ i vincoli, è facile controllare *tutte* le combinazioni.

Per m grande sono stati proposti vari algoritmi[10], tra cui l'importante *active set*, particolarmente adatto a problemi quadratici con hessiana definita positiva. Ad illustrazione del metodo, si veda il successivo **esempio 5**, a pag. 70.

1. caso:
$\varphi_1(x) = 0$ (attivo; λ_1 può essere $\neq 0$) ; $\varphi_2(x) < 0$ (inattivo) $\implies \lambda_2 = 0$.

Si ottiene il sistema non lineare

$$\begin{array}{rrr} 2x & +2\lambda_1 x & = 0 \\ 1 & +2\lambda_1 y & = 0 \\ & x^2 + y^2 - 9 & = 0 \end{array}$$

Dalla prima equazione si ha $2(1 + \lambda_1)x = 0$, e quindi $1 + \lambda_1 = 0$ e $x = 0$.
La prima condizione fornisce $\lambda_1 = -1 < 0$ (caso di non ottimalità).
Viceversa, dalla seconda condizione si ottiene facilmente la soluzione (ottimale)

$$(\overline{x},\ \overline{y},\ \overline{\lambda_1}, \overline{\lambda_2}) = (0,\ -3,\ \frac{1}{6},\ 0), \quad f(\overline{x}, \overline{y}) = -3.$$

Dunque, il punto $(\overline{x},\ \overline{y}) = (0,\ -3)$ è ottimale.

[9] Per un problema con m vincoli, ve ne sono 2^m.
[10] Tra essi, il recente metodo interior point che si vedrà alla sezione 2.6.

Si noti che il caso di non ottimalità fornisce la soluzione

$$(\overline{x},\ \overline{y},\ \overline{\lambda_1}, \overline{\lambda_2}) = (-\frac{\sqrt{35}}{2},\ \frac{1}{2},\ -1,\ 0), \quad f(\overline{x}, \overline{y}) = \frac{37}{4}\ .$$

Osservando che, posto

$$L(x, y, \lambda_1, \lambda_2) = -f + \lambda_1\ \varphi_1 + \lambda_2\ \varphi_2$$

da cui

$$-L(x, y, \lambda_1, \lambda_2) = f + (-\lambda_1)\ \varphi_1 + (-\lambda_2)\ \varphi_2,$$

non è difficile convincersi che essa corrisponde a un punto di ottimalità per la funzione obiettivo $-f$, ricordando che $\max\ f = -\min\ (-f)$.
Si è infatti trovato il massimo della f (cfr. Esempio 1').

2. caso:
$\varphi_1(x) < 0$ (inattivo) $\implies \lambda_1 = 0$; $\varphi_2(x) = 0$ (attivo; λ_2 può essere $\neq 0$).
Le condizioni KKT forniscono:

$$1 + \lambda_2 = 0 \implies \lambda_2 = -1 < 0 \quad \text{(impossibile)}.$$

3. caso: $\varphi_1(x) < 0$ e $\varphi_2(x) < 0$ (inattivi) $\implies \lambda_1 = \lambda_2 = 0$.
Le condizioni KKT forniscono: $2x = 0;\ 1 = 0$ (impossibile).
(Non vi sono estremanti interni al dominio).

4. caso: $\varphi_1(x) = \varphi_2(x) = 0$ (attivi; $\lambda_1 \neq 0,\ \lambda_2 \neq 0$).
Le condizioni KKT forniscono i 2 punti (sulla frontiera):

$$(x_1, y_1) = (\frac{1-\sqrt{17}}{2}, \frac{1+\sqrt{17}}{2}), \quad (x_2, y_2) = \frac{1+\sqrt{17}}{2}, \frac{1-\sqrt{17}}{2}),$$

con $f(x_1, y_1) = f(x_2, y_2) = 5\ (> -3)$.
In questo 4. caso, come si verifica facilmente, non si può avere $\lambda_1 > 0$ e $\lambda_2 > 0$.

In conclusione (cfr. la Figura 2.3) si è ottenuto il minimo assoluto, uguale a -3, raggiunto in $(\overline{x},\ \overline{y}) = (0,\ -3)$, come si può anche verificare calcolando l'hessiana della lagrangiana.

Per *esercizio,* ricordando che $\max\ f = -\min\ (-f)$, risolvere

Esempio 1' (Tema: 07/09/2005)

$\max\ f = x^2 + y, \qquad \Omega = \{(x, y) \in R^2;\ x^2 + y^2 - 9 \leq 0; \quad x + y - 1 \leq 0\}.$

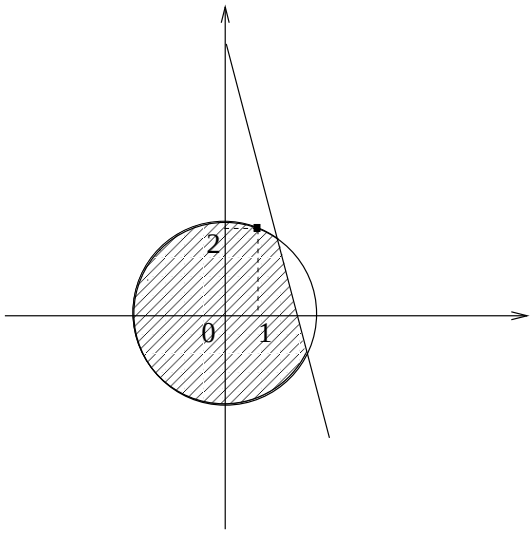


Figura 2.4: Esempio 2.

Risolvendo

$\min\ -f = -(x^2+y), \qquad \Omega = \{(x,y) \in R^2;\ x^2+y^2-9 \leq 0; \quad x+y-1 \leq 0\},$

si trova: $\max f = f(-\frac{\sqrt{35}}{2}, \frac{1}{2}) = \frac{37}{4} = 9.25.$

Esempio 2 (in R^2)

$$\min\ f = 2x^2 + 2xy + y^2 - 10x - 10y$$

sotto i vincoli $(m = 2) \quad x^2 + y^2 \leq 5, \quad 3x + y \leq 6, \quad$ ossia

$$\Omega = \{(x,y) \in R^2\ : x^2 + y^2 \leq 5;\ 3x + y \leq 6\}.$$

Lagrangiana:

$L(x, y, \lambda_1, \lambda_2) = 2x^2 + 2xy + y^2 - 10x - 10y + \lambda_1(x^2 + y^2 - 5) + \lambda_2(3x + y - 6)$

Le condizioni necessarie del primo ordine di K.Kuhn-Tucker sono:

$$\begin{array}{rrrl} 4x & +2y & -10 & +2\lambda_1 x + 3\lambda_2 = 0 \\ 2x & +2y & -10 & +2\lambda_1 y + \lambda_2 = 0 \\ & & & x^2 + y^2 - 5 \leq 0 \\ & & & 3x + y - 6 \leq 0 \\ & & & \lambda_1 \geq 0, \quad \lambda_2 \geq 0 \\ & & & \lambda_1(x^2 + y^2 - 5) = 0 \\ & & & \lambda_2(3x + y - 6) = 0 \end{array}$$

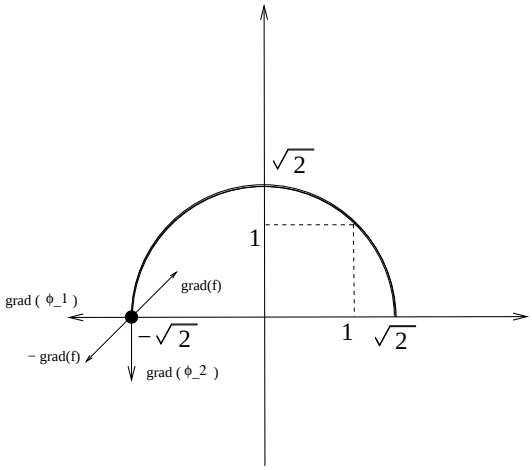


Figura 2.5: Esempio 3.

Cominciamo supponendo che il primo vincolo sia attivo ($\varphi_1 = 0$, da cui possibilmente $\lambda_1 \neq 0$) ed il secondo inattivo ($\varphi_2 < 0$, da cui $\lambda_2 = 0$). Si ottiene il sistema

$$\begin{aligned} 4x\ +2y\ -10\ +2\lambda_1 x\ &= 0 \\ 2x\ +2y\ -10\ +2\lambda_1 y\ &= 0 \\ x^2 + y^2 &= 5, \end{aligned}$$

che ha per soluzione $x = 1,\ y = 2,\ \lambda_1 = 1\ (> 0)$, da cui $3x + y = 5$ e perciò il secondo vincolo è soddisfatto. La soluzione trovata soddisfa dunque le condizioni del primo ordine (si veda la Figura 2.4).

Come si può verificare, le altre combinazioni (primo vincolo inattivo e secondo attivo, primo e secondo vincolo inattivi, primo e secondo vincoli attivi) non forniscono altri punti ottimali.

Esempio 3 (in R^2) (Tema: 29/03/2006)

$$\min\ f = x + y \quad \text{sotto i vincoli} \quad x^2 + y^2 - 2 \leq 0\ , \quad -y \leq 0 \quad (y \geq 0).$$

Lagrangiana:

$$L(x, y, \lambda_1, \lambda_2) = x + y + \lambda_1(x^2 + y^2 - 2) + \lambda_2(-y)$$

Le condizioni necessarie del primo ordine di K.-Kuhn-Tucker sono:

$$\begin{array}{rrr} 1 & +2\lambda_1 x & = 0 \\ 1 & +2\lambda_1 y \quad -\lambda_2 & = 0 \\ & x^2+y^2-2 & \leq 0 \\ & -y & \leq 0 \\ \lambda_1 & \geq 0, \quad \lambda_2 & \geq 0 \\ & \lambda_1(x^2+y^2-2) & = 0 \\ & \lambda_2(-y) & = 0 \end{array}$$

Cominciamo supponendo i due vincoli attivi: $\varphi_1 = 0$, $\varphi_2 = 0$, da cui possibilmente λ_1 e $\lambda_2 \neq 0$.

Si ottiene il sistema

$$\begin{array}{rrr} 1 & +2\lambda_1 x & = 0 \\ 1 & -\lambda_2 & = 0 \\ & x^2-2 & = 0 \end{array}$$

Si trovano subito le due soluzioni:

$$(\overline{x}, \overline{y}, \overline{\lambda_1}, \overline{\lambda_2}) = (-\sqrt{2},\ 0,\ \frac{1}{2\sqrt{2}},\ 1), \qquad f(\overline{x}, \overline{y}) = -\sqrt{2}.$$

$$(\overline{x}, \overline{y}, \overline{\lambda_1}, \overline{\lambda_2}) = (-\sqrt{2},\ 0,\ \frac{1}{2\sqrt{2}},\ 1), \qquad f(\overline{x}, \overline{y}) = \sqrt{2}.$$

Solo la prima è ottimale (cfr. la Figura 2.5).
Inoltre, l'hessiana della lagrangiana in $(\overline{x}, \overline{y}, \overline{\lambda}_1, \overline{\lambda}_2)^T$ è

$$\nabla^2 L = \begin{pmatrix} 2\overline{\lambda}_1 & 0 \\ 0 & 2\overline{\lambda}_1 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} & 0 \\ 0 & 1/\sqrt{2} \end{pmatrix} .$$

Questa matrice è definita positiva, e quindi (condizione sufficiente) il punto trovato è proprio un minimo locale, anzi globale, trattandosi di un problema di programmazione convessa.

Si noti in Figura 2.5 come il vettore - grad(f) $= -\nabla f(\overline{x}, \overline{y})$ appartiene al cono generato dai gradienti dei due vincoli in $(\overline{x}, \overline{y})$, cioè

$$-(1,\ 1)^T = -\nabla f(\overline{x})] = \overline{\lambda_1}\,\nabla\varphi_1(\overline{x}) + \overline{\lambda_2}\,\nabla\varphi_2(\overline{x}) = \frac{1}{2\sqrt{2}}(-2\sqrt{2},\ 0)^T + 1(0,\ -1)^T.$$

Si può verificare che le altre tre combinazioni (primo vincolo attivo e secondo inattivo[11] e viceversa, primo e secondo vincolo inattivi) non forniscono altri

[11] Questo caso di non ottimalità fornisce la soluzione

$$(\overline{x},\ \overline{y},\ \overline{\lambda_1}, \overline{\lambda_2}) = (1,\ 1,\ -\frac{1}{2},\ 0), \quad f(\overline{x}, \overline{y}) = 2.$$

punti ottimali.

Esempio 3'

$$\max f = x + y \quad \text{sotto i vincoli} \quad x^2 + y^2 - 2 \leq 0 \ , \quad -y \leq 0 \quad (y \geq 0).$$

Si trova: $\max(f) = f(1,1) = 2$, in corrispondenza della soluzione di non ottimalità per la ricerca del minimo della f:

$$(\overline{x},\ \overline{y},\ \overline{\lambda_1}, \overline{\lambda_2}) = (1,\ 1,\ -\frac{1}{2},\ 0) \ .$$

Esempio 4 (in R^2) (Si veda l'"analogo" esempio 1 in sezione 2.1)

$$\min\ f = x + y \quad \text{sotto il vincolo} \quad x^2 + y^2 - 1 \leq 0.$$

Lagrangiana:

$$L(x, y, \lambda_1) = x + y + \lambda_1(x^2 + y^2 - 1)$$

Si trova la soluzione ottimale:

$$(\overline{x}, \overline{y}, \overline{\lambda_1}) = (\frac{-1}{\sqrt{2}}, \frac{-1}{\sqrt{2}},\ \frac{1}{2\sqrt{2}}), \qquad f(\overline{x}, \overline{y}) = -\sqrt{2}.$$

Inoltre, l'hessiana della lagrangiana in $(\overline{x}, \overline{y}, \overline{\lambda}_1)^T$ è

$$\nabla^2 L = \begin{pmatrix} 2\overline{\lambda}_1 & 0 \\ 0 & 2\overline{\lambda}_1 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} & 0 \\ 0 & 1/\sqrt{2} \end{pmatrix} \ .$$

Questa matrice è definita positiva, e quindi (condizione sufficiente) il punto trovato è proprio un minimo locale, anzi globale, trattandosi di un problema di programmazione convessa.

Esempio 4'

$$\max f = x + y \quad \text{sotto il vincolo} \quad x^2 + y^2 - 1 \leq 0.$$

Si trova: $\max(f) = f(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}) = \sqrt{2}$.

Esempio 5 (Active Set)

Esso corrisponde a un punto di ottimalità per la funzione obiettivo $-f$. Difatto si è ottenuto il massimo della f.

Per risolvere un problema generale di programmazione convessa, senza entrare nella teoria, accenniamo al metodo detto *active set*, limitandoci a qualche considerazione di carattere generale[12]. L'esempio illustrerà l'algoritmo.

L'idea del metodo è di definire ad ogni passo un insieme di vincoli, detto *working set*, da considerare come attivo. A ogni passo si risolve un sottoproblema di programmazione quadratica con solo vincoli di uguaglianza, eliminando tutti gli altri vincoli. Il *working set* può cambiare scartando qualche vincolo ed aggiungendone un altro, facendo in modo che la funzione obiettivo decresca. Questo si ottiene vedendo il segno dei moltiplicatori di Lagrange, che, come è noto, per l'ottimalità devono essere non negativi. Se, viceversa, uno dei moltiplicatori è negativo, la funzione obiettivo decresce se si scarta il vincolo corrispondente. Se ve ne sono più di uno, è più conveniente scartare quello corrispondente al moltiplicatore più negativo.
Essendovi un numero finito di working sets, il processo ha termine dopo un numero finito di passi.

Esempio. Consideriamo il problema (quadratico)

$$\begin{aligned} &\min \quad f(x,y) = 2x^2 + xy + y^2 - 12x - 10y \qquad (2.16)\\ &= \frac{1}{2}x^T H x - b^T x = \frac{1}{2}(x,\ y)\begin{pmatrix} 4 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} - (12,\ 10)\begin{pmatrix} x \\ y \end{pmatrix} \end{aligned}$$

$$\text{sotto i vincoli} \quad (m=3) \qquad x + y \le 4, \quad -x \le 0, \quad -y \le 0.$$

Prendendo come punto iniziale $x_0 = (0,\ 0)^T$, come *working set* definiamo ovviamente quello formato dal secondo e terzo vincolo (vincoli attivi in x_0): $-x = 0,\ -y = 0$, Si veda la Figura 2.6. Ovviamente si ha la direzione di ricerca $p_0 = (0,\ 0)^T$. Inoltre, λ_1 e $\lambda_2 \neq 0,\ \lambda_3 = 0$, da cui la Lagrangiana:

$$L(x, y, \lambda_1, \lambda_2) = f(x) + \lambda_1(-x) + \lambda_2(-y)$$

Le condizioni KKT forniscono il sistema aumentato (n+m=2+2):

$$\begin{aligned} L_x = 4x + y - 12 - \lambda_1 &= 0 \\ L_y = x + 2y - 10 - \lambda_2 &= 0 \\ -\lambda_1 x &= 0 \\ -\lambda_2 y &= 0 \\ \lambda_1 \neq 0, \quad \lambda_2 &\neq 0 \end{aligned}$$

Si ottiene facilmente: $x = y = 0$ e i due moltiplicatori $\lambda_1 = -12,\ \lambda_2 = -10$, entrambi *negativi*. Essendo λ_1 più negativo di λ_2, il primo vincolo $-x \le 0$

[12] Si veda [34, cap. 16].

esce dal working set, e come vincolo attivo rimane solo il secondo $-y \leq 0$. La Lagrangiana è:

$$L(x, y, \lambda_2) = f(x) + \lambda_2(-y),$$

e le condizioni KKT forniscono il sistema:

$$\begin{array}{rr} L_x = 4x + y - 12 & = 0 \\ L_y = x + 2y - 10 & -\lambda_2 = 0 \\ -\lambda_2 y = 0, \qquad \lambda_2 \neq 0 & \end{array}$$

Si ottiene facilmente: $x_1 = (3, 0)^T$ e $\lambda_2 = -7$, ancora negativo. Dunque, anche il secondo vincolo viene rimosso e il working set è ora l'insieme vuoto. Abbiamo perciò un problema di minimo libero:

$$\min \ f = 2x^2 + xy + y^2 - 12x - 10y.$$

Le condizioni di Lagrange (KKT) diventano $\nabla f = 0$:

$$\begin{aligned} L_x &= 4x + y - 12 = 0 \\ L_y &= x + 2y - 10 = 0 \end{aligned}$$

Si ottiene: $x = 2, \ y = 4$ (centro delle ellissi $f = cost$).

La direzione p_1 di minimo è quella che va dal punto $x_1 = (3, \ 0)^T$ al punto $(2, \ 4)^T$: $p_1 = (2 - 3, \ 4 - 0)^T = (-1, \ 4)^T$, cioè lungo la retta di coefficiente angolare $m = \operatorname{tg} \alpha = -4$, ossia $y = -4(x - 3)$.
Ci muoviamo lungo questa retta fino ad incontrare un nuovo vincolo. Cerchiamo dunque l'intersezione di questa retta con il primo vincolo $x + y \leq 4$. Si ottiene subito il punto $x_2 = (8/3, \ 4/3)^T$.
Come si può verificare, risulta $x_2 = x_1 + \alpha p_1$, con $\alpha = 1/3$ che è minore di 1, e il working set è ora costituito dal primo vincolo $x + y \leq 4$, che così diventa attivo (si dice anche che è un *vincolo bloccante*).

Nota .
Il valore di α si può calcolare con la formula

$$\alpha_k = \min_{a_i^T p_k > 0} \left\{ 1, \frac{b_i - a_i^T x_k}{a_i^T p_k} \right\}$$

Nel nostro caso:

$$k = 2, \ i = 1, \ a_i^T = (1, \ 1)^T, \ b_i = 4, \ x_k = (3, \ 0)^T, \ p_k = (-1, \ 4)^T$$

$$\alpha_2 = \frac{4 - (1, \ 1) \begin{pmatrix} 3 \\ 0 \end{pmatrix}}{(1, \ 1) \begin{pmatrix} -1 \\ 4 \end{pmatrix}} = \frac{1}{3} \ .$$

Siamo ora in grado di trovare il punto di minimo cercato x_3, minimizzando la f lungo la direzione $p_2 = (-1,\ 1)^T$ del vincolo, tramite una line search:

$$x_3 = x_2 + \alpha p_2 = \left(\frac{8}{3},\ \frac{4}{3}\right)^T + \alpha \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \left(\frac{8}{3} - \alpha,\ \frac{4}{3} + \alpha\right)^T,$$

ottenendo

$$f(x_3) = \phi(\alpha) = 2(\frac{8}{3}-\alpha)^2 + (\frac{8}{3}-\alpha)(\frac{4}{3}+\alpha) + (\frac{4}{3}+\alpha)^2 - 12(\frac{8}{3}-\alpha) - 10(\frac{4}{3}+\alpha).$$

La condizione di minimo $\phi'(\alpha) = 0$ fornisce $\alpha = 7/6$ che è positivo (e < 1). Perciò, il punto di minimo cercato è

$$x_3 = \left(\frac{8}{3} - \frac{7}{6},\ \frac{4}{3} + \frac{7}{6}\right)^T = (3/2,\ 5/2)^T,$$

con $f(x_3) = -28.5$.

Osservazione.
Si può verificare che, essendo l'hessiana H definita positiva, la $f = cost$ rappresenta una famiglia di ellissi confocali, di centro $C = (2, 4)$ e di assi le due rette $y - 4 = (-1 \pm \sqrt{2})(x - 2)$, corrispondenti ai due coefficienti angolari m, di angoli $\pi/8$ e $5/8\pi$, rispettivamente[13]. La f cresce procedendo dal centro verso l'origine.
Si ha: $f(1.5,\ 2.5) = -28.5,\ f(3, 0) = -18,\ f(0, 0) = 0$, ecc...
E' geometricamente evidente che il minimo vincolato è raggiunto in x_3, dove si ha la tangenza fra la f e il vincolo $x + y = 4$.

Si ha infatti il seguente problema di minimo con 1 vincolo di uguaglianza

$$\min\ f = 2x^2 + xy + y^2 - 12x - 10y, \quad \text{sotto il vincolo} \quad x + y = 4.$$

Lagrangiana: $L(x, y; \lambda) = f + \lambda(x + y + z - 4)$
Le condizioni di Lagrange (KKT) di estremo vincolato (2.4) sono

$$\begin{aligned} L_x &= 4x + y - 12 + \lambda = 0 \\ L_y &= x + 2y - 10 + \lambda = 0 \\ x &+ y \qquad\quad = 4 \end{aligned}$$

Si trova la soluzione: $x = 3/2,\ y = 5/2,\ \lambda = 7/6$.

[13]Infatti: $\nabla f = 0$, da cui $C = (2, 4)$. Inoltre: $h_{12}m^2 + (h_{11} - h_{22})m - h_{12} = m^2 + 2m - 1 = 0$, da cui $m = -1 \pm \sqrt{2}$.

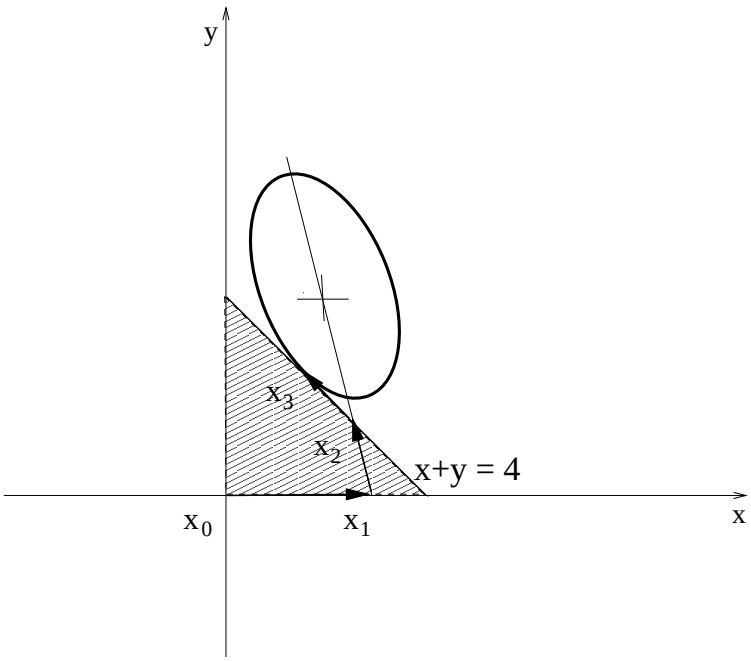


Figura 2.6: Programmazione quadratica: active set (in R^2).

2.3 Programmazione Quadratica Sequenziale (SQP)

Vediamo ora come si può risolvere *numericamente* un generale problema di ottimizzazione vincolata con vincoli di uguaglianza e/o di disuguaglianza (2.7)

$$\min\ f(x), \quad x \in \Omega \subset R^n \qquad \Omega = \{x \in R^n;\ \varphi(x) \leq 0\}.$$

Tralasciamo per il momento il caso della *Programmazione Lineare* (funzione obiettivo lineare e vincoli lineari), sezione 2.5, per il quale esistono consolidati metodi di risoluzione: metodo del *simplesso* di Dantzig (1947), o il più recente (e più efficiente per problemi, *anche non* lineari, di grandi dimensioni) metodo *interior point*, introdotto per primo da Karmarkar (1984), sezione 2.6.

In sezione 2.4 vedremo il metodo di *Penalizzazione* che risolve il problema senza ricorrere ai metodi visti dei moltiplicatori di Lagrange (2.2) o alle "analoghe" condizioni (2.9) KKT. Come si vedrà, per un problema con n variabili e m vincoli, i metodi di penalizzazione operano direttamente nello spazio n-dimensionale delle variabili.

Invece, i più moderni metodi di ottimizzazione vincolata (per esempio il metodo del punto interno, sezione 2.6), risolvono *direttamente* le predette

equazioni KKT del problema originale, tramite una opportuna iterazione di Newton.

In questa sezione descriviamo l'importante metodo detto della *Programmazione Quadratica Sequenziale* (SQP)[14]. Per sua natura, questo metodo fornisce sia le variabili primali che le duali (moltiplicatori di Lagrange).

Il metodo SQP (anche detto metodo di Lagrange proiettato, o metodo di Newton-Lagrange), allo scopo di ottenere una convergenza quadratica o almeno superlineare, usa metodi di Newton o quasi-Newton, superando in tal modo la difficoltà che si opera in uno spazio di dimensione $n + m$, con uguaglianze e disuguaglianze.
Precisamente, si risolve una successione di problemi di *programmazione quadratica* (2.15), in cui la *hessiana* della funzione obiettivo da minimizzare è la hessiana della lagrangiana ed i *vincoli* sono l'approssimazione del primo ordine dei vincoli effettivi.
La convergenza globale si ottiene minimizzando simultaneamente una opportuna *funzione di merito*.

SQP con vincoli di uguaglianza

Consideriamo dapprima il problema con vincoli di uguaglianza, che facilmente si potrà estendere al caso con vincoli di disuguaglianza:

$$\min\ f(x), \qquad x \in \Omega \subset R^n \qquad \Omega = \{x \in R^n;\ \varphi(x) = 0\}, \tag{2.17}$$

dove $f : R^n \longrightarrow R$ e $\varphi = (\varphi_1(x), \ldots, \varphi_m(x))^T$ è una funzione da R^n in R^m. Tutte le funzioni si assumono differenziabili con continuità (almeno) due volte.

Introdotta la *funzione lagrangiana* (2.3)

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i\ \varphi_i(x) = f(x) + \lambda^T\ \varphi(x),$$

si ottiene il sistema (2.4)–(2.5) di $n+m$ equazioni nelle n incognite x_i e nelle m incognite λ_i

$$\nabla_x L(x, \lambda) = \nabla f(x) + \sum_{i=1}^{m} \lambda_i\ \nabla\varphi_i(x) = \nabla f(x) + \lambda^T \nabla\varphi(x) = 0,$$
$$\nabla_\lambda L(x, \lambda) = \varphi(x) = 0$$

[14]In inglese: Sequential Quadratic Programming. Si veda [34, cap. 18], [1, cap. 10] e [11] pag. 2.23 e segg.

Per risolvere questo sistema non lineare usiamo il *metodo di Newton*. Poichè la matrice jacobiana del sistema è

$$\begin{pmatrix} H(x,\lambda) & \nabla\varphi(x)^T \\ \nabla\varphi(x) & 0 \end{pmatrix}$$

dove H è la matrice hessiana della lagrangiana L

$$H(x,\lambda) = \nabla^2_{xx} L(x,\lambda) = \nabla^2 f(x) + \lambda^T\ \nabla^2\varphi(x) = \nabla^2 f(x) + \sum_{i=1}^{m} \lambda_i\ \nabla^2\varphi_i(x),$$

si ottiene, alla iterazione k-esima (indicata dal pedice k), il *sistema lineare*:

$$\begin{pmatrix} H_k & \nabla\varphi_k^T \\ \nabla\varphi_k & 0 \end{pmatrix} \begin{pmatrix} p_k \\ p_\lambda \end{pmatrix} = - \begin{pmatrix} \nabla f_k + \nabla\varphi_k^T\ \lambda_k \\ \varphi_k \end{pmatrix}, \tag{2.18}$$

dove

$$p_k = x_{k+1} - x_k, \qquad p_\lambda = \lambda_{k+1} - \lambda_k.$$

Come già visto nell'Esempio 3 di sezione 2.1, si ha[15]che il sistema è unicamente risolubile se la matrice jacobiana dei vincoli $J_k = \nabla\varphi_k$ ha rango massimo e la matrice H_k è definita positiva sullo spazio tangente dei vincoli, cioè, $d^T H d > 0$ per ogni $d \neq 0$ tale che $J_k\ d = 0$.

SQP: paradigma

Possiamo vedere come l'iterazione di Newton (2.18) sia *equivalente al seguente problema di ottimizzazione quadratica*. All'iterata (x_k, λ_k) definiamo il problema

$$\min\ \frac{1}{2}\ p^T\ H_k\ p + \nabla f_k^T\ p + f_k \tag{2.19}$$

$$\text{tale che} \qquad \nabla\varphi_k\ p + \varphi_k = 0. \tag{2.20}$$

Nota 1.
Tenuto conto del vincolo (2.20), la funzione obiettivo (2.19) rappresenta l'approssimazione del secondo ordine della lagrangiana $L(x,\lambda) = f(x) + \lambda^T\ \varphi(x)$. *Infatti*, fissato λ e posto $p = x - x_k$, lo sviluppo di Taylor della L, rispetto a x, è:

$$\begin{aligned} L(x,\lambda) &= [f_k + \lambda^T\ \varphi_k]\ +\ \nabla L_k^T\ p\ +\ \frac{1}{2}\ p^T\ \nabla^2 L_k\ p \\ &= [f_k + \lambda^T\ \varphi_k] + [\nabla f_k +\ \lambda^T\ \nabla\varphi_k]\ p + \frac{1}{2}\ p^T\ [\nabla^2 f_k + \lambda^T\ \nabla^2\varphi_k]\ p \\ &= (\text{eliminando la condizione vincolare (2.20)}\ ) \\ &= f_k + \nabla f_k^T\ p + \frac{1}{2}\ p^T\ [\nabla^2 f_k + \lambda^T\ \nabla^2\varphi_k]\ p. \end{aligned}$$

[15]Cfr. [34, Assumption 18.1].

(Fine Nota 1).

Nelle solite ipotesi di regolarità, la soluzione del problema di minimo esiste ed è unica.
Risolviamolo. Introdotta la Lagrangiana:

$$\mathcal{L}(p,\mu) = (\frac{1}{2}\ p^T H_k p + \nabla f_k^T\ p + f_k) + \mu\ (\nabla\varphi_k\ p + \varphi_k),$$

le condizioni (KKT) di minimo sono:

$$\begin{aligned} \nabla_p\mathcal{L}(p,\mu) &= H_k\ p + \nabla f_k + \ \nabla\varphi_k^T\ \mu = 0, \\ \nabla_\mu\mathcal{L}(p,\mu) &= \nabla\varphi_k\ p + \varphi_k = 0 \end{aligned} \tag{2.21}$$

ossia, in forma matriciale compatta,

$$\begin{pmatrix} H_k & \nabla\varphi_k^T \\ \nabla\varphi_k & 0 \end{pmatrix} \begin{pmatrix} p_k \\ \mu_k \end{pmatrix} = \begin{pmatrix} -\nabla f_k \\ -\varphi_k \end{pmatrix}.$$

Osserviamo ora che la prima equazione del sistema (2.18) si può riscrivere come:

$$H_k\ p_k + \ \nabla\varphi_k^T\ \lambda_{k+1} - \nabla\varphi_k^T\ \lambda_k = -\nabla f_k - \nabla\varphi_k^T\ \lambda_k,$$

che dunque *coincide* con la prima equazione del sistema (2.21), e quindi si ha $p = p_k$ e $\lambda_{k+1} = \mu_k$ in virtù dell'unicità della soluzione.

SQP con vincoli di disuguaglianza

Includiamo ora nel problema di minimo (2.17) m vincoli di disuguaglianza $\varphi_i(x)$ (e siano l quelli di uguaglianza $h_i(x)$):

$$\begin{aligned} &\min\ f(x) \\ \text{tale che} \qquad &\varphi_i(x) \le 0 \qquad i = 1,\dots,m \\ &h_i(x) = 0 \qquad i = 1,\dots,l \end{aligned} \tag{2.22}$$

Linearizzando i vincoli, otteniamo il **sottoproblema quadratico (QP)**:

$$\begin{aligned} &\min\ \tfrac{1}{2}\ p^T\ H_k\ p + \nabla f_k^T\ p + f_k \\ \text{tale che} \qquad &\nabla\varphi_i(x_k)^T\ p + \varphi_i(x_k) \le 0 \qquad i = 1,\dots,m \\ &\nabla h_i(x_k)^T\ p + h_i(x_k) = 0 \qquad i = 1,\dots,l \end{aligned} \tag{2.23}$$

dove H è la hessiana della lagrangiana $L(x,\lambda,\nu) = f(x) + \lambda^T\ \varphi(x) + \nu^T\ h(x)$:

$$\begin{aligned} H(x,\lambda,\nu) &= \nabla^2_{xx} L(x,\lambda,\nu) = \nabla^2 f(x) + \lambda^T\ \nabla^2\varphi(x) + \nu^T\ \nabla^2 h(x) \\ &= \nabla^2 f(x) + \sum_{i=1}^{m} \lambda_i\ \nabla^2\varphi_i(x) + \sum_{i=1}^{l} \nu_i\ \nabla^2 h_i(x). \end{aligned}$$

Le condizioni KKT sono:

$$\begin{aligned} \nabla^2_{xx} H_k\ p + \nabla f(x_k) + \sum_{i=1}^{m} \lambda_i\ \nabla\varphi_i(x_k) + \sum_{i=1}^{l} \nu_i\ \nabla h_i(x_k) = 0 \\ \nabla\varphi_i(x_k)^T\ p + \varphi_i(x_k) \le 0 \qquad i = 1, \dots, m \\ \nabla h_i(x_k)^T\ p + h_i(x_k) = 0 \qquad i = 1, \dots, l \\ \lambda_i[\nabla\varphi_i(x_k)^T\ p + \varphi_i(x_k)] = 0 \qquad i = 1, \dots, m \\ \lambda \ge 0, \qquad \nu \quad \text{libero} \end{aligned} \tag{2.24}$$

Nota 2 .
Per risolvere il problema quadratico (2.23), eventualmente ricondotto alla forma standard (2.15), vari algoritmi sono stati proposti (cfr. [34, cap. 16]): active set, gradiente proiettato, interior point. Come già accennato, per quest'ultimo metodo si veda la sezione 2.6. Un esempio del metodo active set è stato visto in sezione 2.1, esempio (2.16).

Convergenza globale e funzione di merito

Una difficoltà del metodo SQP consiste nel richiedere l'esistenza delle derivate seconde (nella hessiana), e inoltre che la stessa hessiana sia definita positiva. Per ovviare a queste difficoltà si preferisce usare un metodo *quasi-Newton*, introducendo un'approssimazione B_k della hessiana che sia definita positiva. Una buona scelta è la *formula BFGS* (1.26).
Ovviamente, il metodo diventa localmente superlineare, e se la soluzione iniziale non è abbastaza prossima al minimo cercato, il metodo non converge.

Per ovviare a questo inconveniente e assicurare una convergenza globale, si introduce una *funzione di merito*. Questa è una funzione che, assieme alla funzione obiettivo, viene simultaneamente minimizzata. Si deve inoltre garantire una condizione di discesa, ottenuta con una opportuna line search. Una funzione di merito facile da implementare, ma che ha l'inconveniente di non essere differenziabile per la presenza del valore assoluto (ma dotata tuttavia di derivate direzionali), è la *funzione esatta* l_1 :

$$F_m(x) = f(x) + \mu \left[\sum_{i=1}^{m} \max\{0, \varphi_i(x)\} + \sum_{i=1}^{l} |h_i(x)| \right] \tag{2.25}$$

dove $\mu > 0$ è chiamato parametro di penalizzazione.
Sfortunatamente, essa può incorrere nel cosidetto *effetto Maratos* (1978); accade cioè che la funzione, pur in vicinanza della soluzione, non decresca. Un semplice esempio, con vincoli di sola uguaglianza, di questo effetto è

dovuto a Powell (1986) [34, sezione 18.11].
Si può, in alternativa, usare la *funzione di merito l_2 di Fletcher* (in pratica una lagrangiana aumentata), che è differenziabile. Per vincoli di sola uguaglianza $\varphi(x)$ essa è:

$$F_m(x) = f(x) + \lambda(x)^T \varphi(x) + \frac{1}{2}\mu \|\varphi(x)\|_2^2 \ .$$

In presenza di vincoli di disuguaglianza, si possono utilizzare delle *variabili slack* s_i tali che: $\varphi_i + s_i = 0, \quad s_i \geq 0, \quad i = 1, \ldots, m$, dove $\mu > 0$ è il parametro di penalizzazione e

$$\lambda(x) = [\nabla\varphi(x)\ \nabla\varphi(x)^T]^{-1}\ \nabla\varphi(x)\ \nabla f(x)$$

sono i moltiplicatori stimati ai minimi quadrati:

$$\nabla\varphi(x)^T\lambda(x) = \nabla f(x) \Longrightarrow \nabla\varphi(x)\nabla\varphi(x)^T\lambda(x) = \nabla\varphi(x)\nabla f(x).$$

Per un pratico algoritmo implementativo della SQP fin qui descritta, si veda [34, sezione 18.6].

Esempio (in R^2)

$$\begin{aligned} \min \quad & f(x) = 2x_1^2 + 2x_2^2 - 2x_1x_2 - 4x_1 - 6x_2 \\ \text{tale che} \quad & \varphi_1(x) = 2x_1^2 - x_2 \leq 0 \\ & \varphi_2(x) = x_1 + 5x_2 - 5 \leq 0 \\ & \varphi_3(x) = -x_1 \leq 0 \\ & \varphi_4(x) = -x_2 \leq 0 \end{aligned}$$

La lagrangiana è:

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m=4} \lambda_i\ \varphi_i(x) = f(x) + \lambda^T\ \varphi(x).$$

Si ha perciò il problema di ottimizzazione quadratica (2.19)—(2.20):

$$\begin{aligned} \min\ & \frac{1}{2}\ p^T\ H_k\ p + \nabla f_k^T\ p + f_k \\ \text{tale che} \quad & \nabla\varphi_i(x_k)^T\ p + \varphi_i(x_k)^T \leq 0, \qquad 0 \leq i \leq 4 \end{aligned}$$

L'hessiana è:

$$H(x, \lambda) = \nabla^2_{xx} L(x, \lambda) = \nabla^2 f(x) + \lambda^T\ \nabla^2\varphi(x) = \nabla^2 f(x) + \sum_{i=1}^{m} \lambda_i\ \nabla^2\varphi_i(x).$$

Risolviamo il problema con le seguenti condizioni iniziali:

$$B = \nabla^2 L, \qquad x_0 = (0,1)^T \qquad \lambda = (0,0,0,0)^T.$$

Si ha:

$$\nabla f = (4x_1 - 2x_2 - 4,\ 4x_2 - 2x_1 - 6)^T = (-6,\ -2)^T \Longrightarrow \nabla f^T d = -6p_1 - 2p_2$$

$$\nabla^2 L(x,\lambda) = \nabla^2 f + 0 = \begin{pmatrix} 4 & -2 \\ -2 & 4 \end{pmatrix} \Longrightarrow \frac{1}{2}\ p^T\ H_k\ p = \frac{1}{2}(4p_1^2 - 4p_1p_2 + 4p_2^2)$$

La matrice jacobiana $J(x) \in R^{m\times n} = R^{4\times 2}$ dei vincoli

$$J(x) = \frac{\partial \varphi_i(x)}{\partial x_j} = \begin{pmatrix} 4x_1 & -1 \\ 1 & 5 \\ -1 & 0 \\ 0 & -1 \end{pmatrix}$$

da cui $\quad \nabla\varphi_i(x_0)^T\ p + \varphi_i(x_0)\ \leq 0, \quad$ cioè

$$J(x_0)\ p = \begin{pmatrix} 0 & -1 \\ 1 & 5 \\ -1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \leq 0, \qquad \begin{array}{rr} -p_2 & -1 \leq 0 \\ p_1 & +5p_2 \leq 0 \\ -p_1 & \leq 0 \\ -p_2 & -1 \leq 0 \end{array} \tag{2.26}$$

Raccogliendo, si ha infine il (sotto)*problema di programmazione quadratica* (QP):

$$\begin{aligned} \min \quad & F(p) = \frac{1}{2}(4p_1^2 - 4p_1p_2 + 4p_2^2)\ +\ (-6p_1 - 2p_2) \\ & \text{sotto i vincoli lineari (2.26).} \end{aligned}$$

Dalla Figura 2.7 è evidente che all'ottimalità, *solo il secondo vincolo* (linearizzato) $\varphi_2(p) = p_1 + 5p_2 \leq 0$ è attivo e quindi la lagrangiana diventa semplicemente:

$$\mathcal{L}(p,\mu) = F(p) + \nu_2\,\varphi_2(p),$$

con ν_2 parametro lagrangiano. Le condizioni KKT associate sono:

$$\begin{aligned} -6 + \frac{1}{2}(8p_1 - 4p_2)\ +\ \nu_2\ &= 0 \\ -2 + \frac{1}{2}(8p_2 - 4p_1)\ +\ 5\nu_2 &= 0 \\ p_1 + 5p_2 &\leq 0 \\ \nu_2(p_1 + 5p_2) &= 0 \qquad\qquad \nu_2 \geq 0\,. \end{aligned}$$

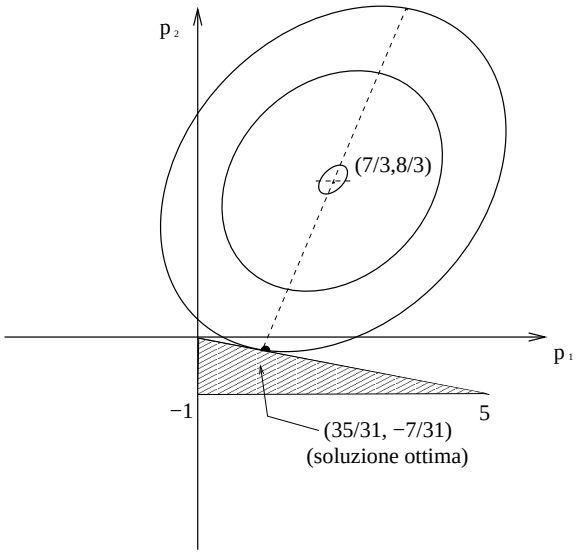


Figura 2.7: Soluzione del sottoproblema quadratico (QP).

Risolvendo, si trova

$$p_0 = \left(\frac{35}{31},\ -\frac{7}{31}\right)^T = (1.129,\ 1.032)^T, \quad \nu_2 = \frac{32}{31} = 1.0323\ (>0)\,.$$

Quindi, la nuova iterata è

$$x_1 = x_0 + p_0 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} \frac{35}{31} \\ \\ -\frac{7}{31} \end{pmatrix} = \begin{pmatrix} 1.1290322 \\ 0.4771936 \end{pmatrix}.$$

Introduciamo ora la *funzione di merito* (2.25), da minimizzarsi, tramite una qualunque line search (cfr. la sezione 1.3), lungo la direzione p_0.

Introdotto un parametro reale $\alpha \geq 0$ e scelto $\mu = 10$, si ha:

$$\begin{aligned} \min\ F_m(x_0 + \lambda p_0) &= f(x_0 + \lambda p_0) + \mu \left[\sum_{i=1}^{4} \max\{0, \varphi_i(x_0 + \lambda p_0)\}\right] \\ &= [3.1612897\ \lambda^2 - 6.3225804\ \lambda - 4] \\ &+10\ [\ \max\{0, 2.5494274\ \lambda^2 + 0.2258064\ \lambda - 1\} \\ &\quad + \max\{0, 0\} + \max\{0, -1.1290322\ \lambda\} \\ &\quad + \max\{0, -1 + 0.2258064\ \lambda\}\ ]\,. \end{aligned}$$

Usando, per esempio, il metodo della sezione aurea si trova $\lambda = 0.5835726$, e quindi

$$x_1 = x_0 + \lambda\, p_0 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} + 0.5835726 \begin{pmatrix} \frac{35}{31} \\ -\frac{7}{31} \end{pmatrix} = \begin{pmatrix} 0.6588722 \\ 0.8682256 \end{pmatrix} .$$

2.4 Metodi di Penalizzazione

Tra i vari altri metodi di risoluzione dei problemi di ottimizzazione vincolata, in questa sezione ci limitiamo a descrivere il classico (da un'idea di Courant, 1943, 1956) metodo[16] di *Penalizzazione* (penalty), che trasforma in modo naturale il problema vincolato

$$\min\ f(x), \quad x \in \Omega \subset R^n \qquad \Omega = \{x \in R^n;\ \varphi(x) \leq 0\}, \tag{2.27}$$

in uno non vincolato o in una *successione* di problemi non vincolati, da risolversi con uno dei metodi di discesa visti nelle sezioni del capitolo 1.
Esso opera direttamente nello spazio n-dimensionale delle variabili.

Per illustrare il concetto di penalizzazione, consideriamo il caso di un solo vincolo di uguaglianza:

$$\min\ f(x), \qquad x \in \Omega \subset R^n \qquad \Omega = \{x \in R^n;\ h(x) = 0\}\,.$$

Rimpiazziamo il problema con il seguente non vincolato:

$$\min\ f(x) + \mu\, h^2(x) \qquad x \in R^n$$

dove $\mu > 0$ è un numero grande.
Si intuisce chiaramente che per ottenere una soluzione ottimale, $h^2(x)$ deve essere piccolo, altrimenti si ha una grande penalizzazione $\mu\, h^2(x)$ che viola il vincolo.

Se invece si ha un vincolo di disuguaglianza:

$$\min\ f(x), \quad x \in \Omega \subset R^n \qquad \Omega = \{x \in R^n;\ \varphi(x) \leq 0\}$$

la precedente penalizzazione non è appropriata, in quanto si avrebbe penalizzazione sia per $\varphi \geq 0$ (punto non ammissibile), che va bene, ma anche per $\varphi(x) \leq 0$. Conviene invece rimpiazzare il problema con il seguente:

$$\min\ f(x) + \mu\, \max\{0, \varphi(x)\} \qquad x \in R^n.$$

[16] Come si fa in programmazione lineare, quando non esiste una base ammissibile (feasible).

È chiaro che la penalizzazione si ha solo per $\varphi(x) > 0$;
Se si vuole una funzione obiettivo differenziabile, allora si può ricorrere a una penalizzazione del tipo $\mu\,[\max\{0, \varphi(x)\}^2]$, cfr. la successiva (2.29).

Vediamo ora la procedura da adottare nel caso generale. Per ogni i si sostituisce il problema originario (2.27) con un problema non vincolato della forma

$$\min\ F_i(c_i, x) = \min\ f(x) + c_i\, P(x) \tag{2.28}$$

dove:
1. c_i è una successione di numeri monotona divergente così definita:
per ogni i, $c_i \geq 0,\ c_{i+1} > c_i$. Ad esempio $c_i = 10^i,\ i \geq 0$.
2. $P(x)$ è una funzione continua non negativa, nulla se e solo se $x \in \Omega$.

Una funzione di penalizzazione, detta *esterna* in quanto la successione di minimizzatori x_i tende al punto di minimo dall'esterno di Ω, molto utile (ed usata) è

$$P(x) = \sum_{i=1}^{m} [\max\{0,\ \varphi\}]^2 \tag{2.29}$$

La funzione $c\,P(x)$ è illustrata in Figura 2.8 nel caso monodimensionale con i due vincoli $\varphi_1(x) = a - x,\ \varphi_2(x) = x - b,\ a < b$.

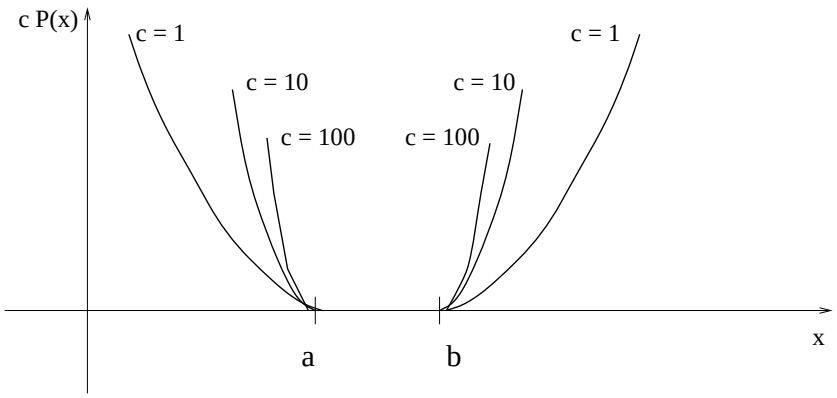


Figura 2.8: Funzione di penalizzazione $cP(x)$ con due vincoli (in R^1).

È chiaro che per c_i molto grande il punto di minimo del problema non vincolato (2.29) si troverà in un insieme dove P è piccola. Idealmente, per $c_i \to \infty$ la successione di minimizzatori x_i generata dal metodo di penalizzazione convergerà al punto di minimo $\overline{x}$ del problema vincolato, come effettivamente *si dimostra*, cfr. [27, sezione 12.1].

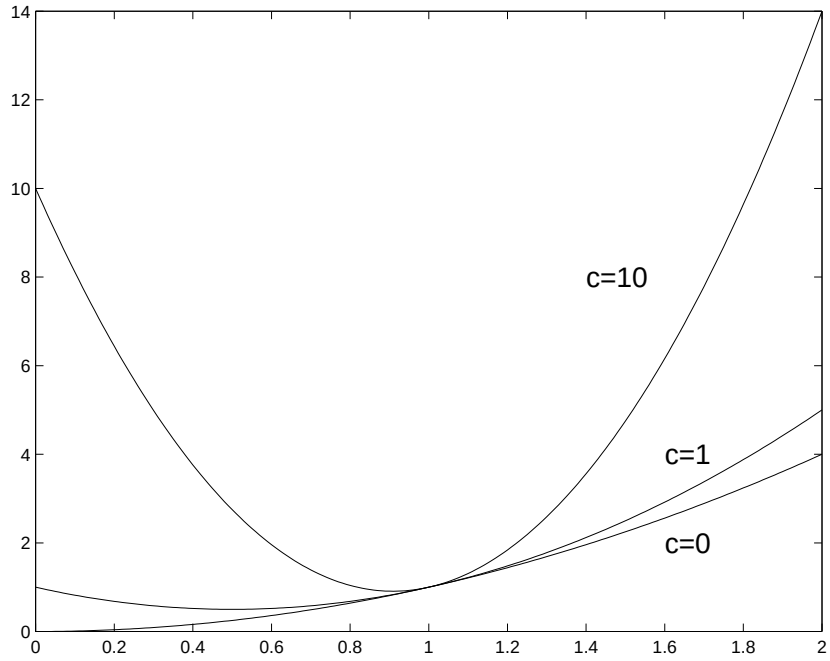


Figura 2.9: Penalizzazione esterna: funzione $F_i(c_i, x)$ (in R^1).

Consideriamo il seguente semplice *esempio* in R^1

$$\min\ x^2, \quad \Omega = \{x \in R^1 : 1 - x \leq 0\}$$

Si ha

$$F_i(c_i, x) = x^2 + 10^i[\max\{1 - x, 0\}]^2.$$

È facile verificare che $F_i(c_i, x)$ ha un minimo in $\overline{x_i} = 10^i/(1 + 10^i) \approx 1$.

Come si vede dalla Figura 2.9, al crescere di c_i l'effetto della penalizzazione è la creazione in prossimità del minimo di una valle sempre più ripida (come nella funzione di Rosenbrock).

ESEMPIO (di programmazione quadratica, [27, pag 423, 439]) $f = 2x^2 + 2xy + y^2 - 2y$ sotto il vincolo $x = 0$; il minimo è $x = 0,\ y = 1$.

Un'altra funzione di penalizzazione, detta *interna* in quanto si tende al punto di minimo dall'interno[17] di Ω che deve essere dunque ad interno $\overset{\circ}{\Omega}$ non vuoto, è la funzione *barriera inversa* $B(x)$

$$B(x) = -\sum_{i=1}^{m} \frac{1}{\varphi_i(x)} \tag{2.30}$$

[17]Da cui i più recenti metodi interior point, sezione 2.6.

oppure la funzione *barriera logaritmica* $B(x)$

$$B(x) = -\sum_{i=1}^{m} \log(-\varphi_i(x)) \tag{2.31}$$

Con le stesse notazioni del metodo di penalizzazione esterna, si risolve la successione di problemi vincolati[18]

$$\min\ G_i(c_i, x) = \min \quad f(x) + \frac{1}{c_i} B(x), \quad x\ \in \overset{\circ}{\Omega} \tag{2.32}$$

dove:
1. c_i è una successione di numeri monotona divergente così definita:
per ogni i, $c_i > 0,\ c_{i+1} > c_i$. Ad esempio $c_i = 10^i,\ i \geq 0$.
2. $B(x)$ è una funzione continua non negativa, e tende all'infinito se x tende alla frontiera di Ω.

La funzione $(1/c)B(x)$ è illustrata in Figura 2.10 nel caso monodimensionale con i due vincoli $\varphi_1(x) = a - x,\ \varphi_2(x) = x - b,\ a < b$.

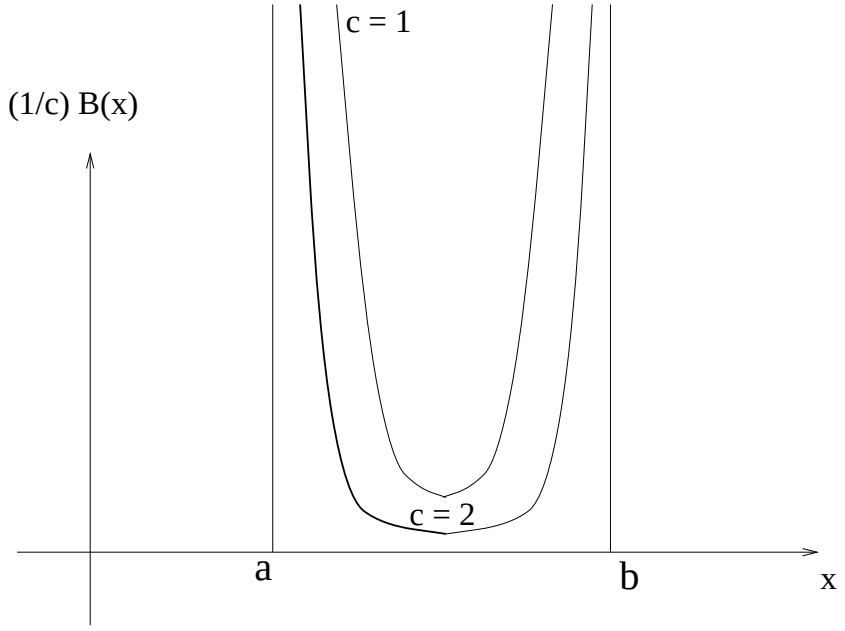


Figura 2.10: Funzione di penalizzazione $(1/c)B(x)$ con due vincoli (in R^1).

[18]Che, se risolti con un metodo di discesa (esempio, di Newton), diventano in pratica dei problemi non vincolati in quanto, essendo la funzione obiettivo G infinita in prossimità del vincolo, la successione dei minimizzatori automaticamente rimane all'interno di Ω.

Come si vede dalla Figura 2.10, al crescere di c_i l'effetto della barriera (penalizzazione interna) è la creazione in prossimità del vincolo di una parete quasi verticale. Il metodo non permette, durante la procedura di ricerca del minimo, di uscire da Ω.

Esempi:
[13, pag. 550], [23].

2.5 Programmazione lineare: Metodo del Simplesso

IDEA (Dantzig): essendo il minimo nei vertici del simplesso, si passa da un vertice ad un altro adiacente in modo tale che la funzione obiettivo diminuisca[19].

Teorema di dualità (cfr. [27, pag. 86 formula (2)], [17, pag. 284], forma asimmetrica):

Problema **primale (P):**

$$\begin{aligned} \min \quad & c^T x \\ \text{tale che} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

Problema **duale (D):**

$$\begin{aligned} \max \quad & b^T y \\ \text{tale che} \quad & A^T y \leq c \quad (\implies \quad A^T y + s = c,\ s \geq 0) \end{aligned}$$

dove $s \in R^n$, vettore delle variabili slack (slack=artificiale).

Definiamo come insieme primale-duale ammissibile:

$$\mathcal{F} = \{(x, y, s) \in R^n \times R^m \times R^n;\ Ax = b,\ A^T y + s = c,\ (x, s) \geq 0\}$$

Se $(x, s) > 0$, chiamiamo $\mathcal{F}^0$ l'insieme strettamente ammissibile.

Per variabili $x,\ y$ ammissibili, si ha:

$$\inf\ c^T x\ \geq\ \sup\ b^T y \quad \text{(Teorema debole della dualità).}$$

Se uno dei due problemi è risolubile, allora:

$$\min\ c^T x\ =\ \max\ b^T y \quad \text{(Teorema forte della dualità).}$$

Esempio: [17, cap. 7, pag. 285]

[19] Si vedano gli APPUNTI (1985/86) e un qualsiasi testo tra quelli citati in bibliografia: [13], [10], [17], [27], [28], [30], [34], [19].

2.6 Metodo Interior Point (IP)

Riferimenti bibliografici[20].

2.6.1 Caso della Programmazione Lineare

1.

Primal problem (P)

$$\min \{c^T x; Ax = b, x \geq 0\},$$

dove $x \in R^n, c \in R^n, A \in R^{m \times n}$. Scriviamo la Lagrangiana (y ed s: parametri lagrangiani):

$$L(x, y, s) = c^T x - y^T(Ax - b) - s^T x,$$

Le condizioni KKT (2.11)—(2.14) forniscono

$$\begin{array}{lcl} (\nabla_x L =)\ c - A^T y - s = 0 & & A^T y + s = c \\ (\nabla_y L =)\ Ax - b = 0 & & Ax - b = 0 \\ (\nabla_s L =)\ x \geq 0 & \Longleftrightarrow & \\ s \geq 0 & & x_i s_i = 0,\ 1 \leq i \leq n \quad (XSe = 0) \\ x^T s = 0 & & (x, s) \geq 0 \end{array} \tag{2.33}$$

dove:

$$S = diag(s_1, s_2, \ldots, s_n), \quad X = diag(x_1, x_2, \ldots, x_n), \quad e = (1, \ldots, 1)^T,$$

l'equazione $XSe = 0$ è equivalente a $x_i s_i = 0,\ 1 \leq i \leq n$, e $(x, s) \geq 0$ vuol dire $x_i \geq 0, s_i \geq 0,\ \forall i$. Ovviamente:

$$X = diag(x_1, x_2, \ldots, x_n) = \begin{pmatrix} x_1 & & & \\ & x_2 & & \bigcirc \\ & & \ddots & \\ \bigcirc & & & x_n \end{pmatrix}.$$

Come sappiamo, le (2.33) sono condizioni necessarie[21] affinchè la tripletta (x^*, y^*, s^*) sia un minimizzatore.

[20] G.Zilli: Corso di: *Metodi di Ottimizzazione e di Punto Interno*, Dottorato in Matematica Computazionale, Padova, 1997-98 e segg.
Si veda: [Appunti J. Gondzio, 2000], l'importante monografia [41], [34], e inoltre [32, cap. 5, pag. 31-33] (caso della programmazione lineare), [2] e bibliografia ivi riportata.
Cfr. anche [33], [37], [13, pag. 528], [17, pag. 287], e la tesina [23] come introduzione ai metodi interior point secondo Fiacco e McCormick (1968, 1990).

[21] In generale non sufficienti per problemi qualunque, ma nel caso lineare anche sufficienti e così per problemi di programmazione convessa (e quadratica) con vincoli lineari [41, pag. 163)] . Nel caso generale bisogna assumere una condizione di qualificazione dei vincoli, cfr. la sezione 2.2.

2.

Primal-Dual problem (P-D)

$$\min \{c^T x; Ax = b, x \geq 0\};$$
$$\max \{b^T y; A^T y \leq c\} \iff \max \{b^T y; s = c - A^T y \geq 0;\ A^T y + s = c\}$$

dove $s \in R^n$, vettore slack (≥ 0) (slack=artificiale), introdotte per convertire le disuguaglianze in uguaglianze.

Cioè, i due problemi **1.** e **2.** si "corrispondono" nel senso che la coppia $(y^*, s^*) \in R^{m+n}$ è soluzione del problema **2.** se e solo se esiste $x^* \in R^n$ tale che il sistema (2.33) delle KKT è verificato per $(x, y, s) = (x^*, y^*, s^*)$.

3.

In realtà la relazione di *complementarietà* [41, pag. 4, pag. 158)]

$$\boxed{XSe = 0} \tag{2.34}$$

(equazione quadratica) diventa (nella teoria *interior point*)

$$\boxed{XSe = \mu e, \qquad \mu > 0} \tag{2.35}$$

cioè:

$$x_i s_i = \mu,\ \forall i, \quad \text{da cui} \quad \mu = \frac{\sum_{i=1}^n x_i s_i}{n} = \frac{x^T s}{n}, \quad x^T s = n\mu.$$

Il parametro μ è il (duality gap), cfr. il punto **4'.** sotto.

Si definisce il *central path*

$$S = \{x(\mu), y(\mu), s(\mu); \mu > 0\},$$

che è la curva (path) lungo la quale si ha $XSe = \mu e$ (da cui il metodo di *path-following primale-duale*, Kojima et al. 1989).

Quanto detto in precedenza si può *giustificare* se si scrivono le condizioni di stazionarietà (del primo ordine) relative al problema primale **1.**, incorporando nella lagrangiana la (classica) *barriera logaritmica*, cfr. la (2.31),

$$L(x, \mu, y) = c^T x \ - y^T(Ax - b) \ - \mu \sum_{i=1}^n \ln x_i,$$

con $\mu > 0$ (parametro barriera), da cui, derivando:

$$\begin{aligned} \nabla_x L &= c - A^T y - \mu X^{-1} e = 0 \\ \nabla_y L &= Ax - b = 0 \\ x &\geq 0 \ , \end{aligned}$$

dove al solito $X = diag(x_1, x_2, \ldots, x_n)$ ed $e = (1, \ldots, 1)^T$.

Ponendo ora $\mu X^{-1} e = s$, cioè

$$s_i = \frac{\mu}{x_i}, \ \forall i \quad \Longrightarrow \quad XSe = \mu e,$$

si ottiene il sistema (2.33), ma con $x_i s_i = \mu$.

Si noti che:

$$\min(-\sum_{i=1}^{n} \ln x_i) = \min \mathbf{e}^{-\sum_{i=1}^{n} \ln x_i} = \min \prod_{i=1}^{n} \frac{1}{x_i} = \max \prod_{i=1}^{n} x_i.$$

Perciò, la barriera logaritmica rimpiazza la condizione di non negatività $x_i \geq 0$, in quanto minimizzare la $(-\sum_{i=1}^{n} \ln x_i)$ equivale a massimizzare il prodotto delle distanze dagli iperpiani che definiscono l'ortante positivo $\{x \in R^n; \quad x_i \geq 0\}$.

4.

Per risolvere questo sistema (mildly=moderatamente) non lineare (delle equazioni KKT), si applica ora il metodo di Newton con un parametro di line search α, cfr. la sezione 1.6. Si noti che esso non è il Newton originale, ma un Newton "perturbato" per la presenza di μ, anzi di $\sigma\mu$ (vedi il sistema (2.36) al punto **4'**.).

4'.

Il parametro μ si chiama *duality measure* (o duality gap) [41, (1.11)].
Nella relazione di complementarietà si introduce il *centering parameter* σ, $0 \leq \sigma \leq 1$, [41, (1.12)]. Casi estremi (vedi **4".**):
$\sigma = 0$ (Newton puro o affine-scaling direction), $\sigma = 1$ (direzione di centramento).
Nella maggior parte degli algoritmi si ha $\sigma \in (0, \ 1)$.

In conclusione, si applica il metodo di Newton al sistema non lineare

$$\begin{aligned} Ax &= b \\ A^T y + s &= c \\ XSe &= \sigma\mu e. \\ (x, s) &\geq 0 \end{aligned} \tag{2.36}$$

Per $\mu \to 0$ il sistema (2.36) coincide con (2.33).
Si noti che il parametro μ controlla la distanza all'ottimo:

$$c^T x - b^T y = c^T x - x^T A^T y = x^T (c - A^T y) = x^T s = n\mu.$$

Tenendo conto della non ammissibilità (infeasibility) delle soluzioni, che in pratica vuol dire (primi due, primale e duale,) residui k-esimi $\neq 0$, il sistema non lineare "interior-point" è (al passo k, dove x^k sta per $x^{(k)}$, ecc...):

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{pmatrix} \begin{pmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{pmatrix} = - \begin{pmatrix} Ax^k - b \\ A^T y^k + s^k - c \\ X^k S^k e - \sigma^k \mu^k e \end{pmatrix} = \begin{pmatrix} \xi_p \\ \xi_d \\ \xi_\mu \end{pmatrix} \tag{2.37}$$

con $(x^0, y^0, s^0) > 0$, $k \geq 0$ e $0 \leq \sigma \leq 1$, $\mu^k = (x^k)^T s^k / n$.

Si pone quindi

$$(x^{k+1}, y^{k+1}, s^{k+1}) \longleftarrow (x^k, y^k, s^k) + (\alpha_P^k \Delta x^k, \alpha_D^k \Delta y^k, \alpha_D^k \Delta s^k)$$

con i parametri di line search α_P^k e α_D^k scelti *in modo che* $(x^k, s^k) > 0$ (da cui metodo del punto interno).

Il metodo del punto interno (2.37) può dunque *essere interpretato* come il metodo di Newton applicato alle condizioni KKT perturbate. In questo modo si creano delle iterate (x^k, y^k, s^k) tali che (x^k, s^k) rimangono strettamente positive. Per $k \longrightarrow \infty$ il *duality gap* μ^k va a zero (linearmente).

Si può **dimostrare** [34, sez. 14.4], [41, cap. 5] che:

- tutte le nuove iterate calcolate con un passo di Newton sono ammissibili e stanno in un (piccolo) θ-intorno $\mathcal{N}_2(\theta)$ (ex: $\theta = 0.1$) del central path:

  $$\mathcal{N}_2(\theta) = \{(x, y, s) \in \mathcal{F}_0;\ \|XSe - \mu e\|_2 \leq \theta\mu\},$$

  e convergono verso la soluzione ottima

- si ha una sistematica, anche se lenta, riduzione del duality gap

  $$\mu^{(k+1)} = \sigma\mu^{(k)}, \quad \sigma < 1$$

  Per esempio: $\sigma = 1 - \dfrac{\beta}{\sqrt{n}}, \quad 0 < \beta < 1 \quad (\beta = 0.1)$

Segue che dopo $\sqrt{n}$ iterazioni si ha una riduzione pari a

$$\left(1-\frac{\beta}{\sqrt{n}}\right)^{\sqrt{n}} \approx \left(e^{-\frac{\beta}{\sqrt{n}}}\right)^{\sqrt{n}} \approx e^{-\beta}.$$

Dopo *cost* $\times \sqrt{n} = \mathcal{O}(\sqrt{n})$, per *cost* abbastanza grande, μ può diventare abbastanza piccolo; quindi l'algoritmo ha una complessità computazionale pari a $\mathcal{O}(\sqrt{n})$.

4".

Una variante[22] dell'algoritmo, invece di $\mu = \dfrac{x^T s}{n}$ richiede $\mu = \gamma \dfrac{x^T s}{n}$, $0 < \gamma < 1$ (ex: $\gamma = 10^{-3}$), in quanto deve valere la *centrality condition* e la coppia (x, s) sta nell'intorno $N_{-\infty}(\gamma)$ del central path. Quindi non $x_i s_i = \mu$, ma, cfr. [41, (1.16)],

$$x_i s_i \geq \gamma\mu,\ 0 < \gamma < 1, \quad \forall i \ \ 1 \leq i \leq n.$$

Soluzione del sistema IP

Il sistema lineare (2.37) può essere ricondotto a una forma ridotta a blocchi (2×2) detta **sistema aumentato** di dimensione $(n+m)$[23]. Infatti, essendo la matrice diagonale X invertibile, usando la terza equazione, si può eliminare il vettore

$$\Delta s = X^{-1}(\xi_\mu - S\Delta x) = -X^{-1}S\Delta x + X^{-1}\xi_\mu$$

dalla seconda equazione, ottenendo il sistema aumentato, che è *simmetrico e indefinito*[24] (per semplicità, è stato soppresso l'indice k della iterazione):

$$\begin{pmatrix} -\Theta^{-1} & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} \xi_d - X^{-1}\xi_\mu \\ \xi_p \end{pmatrix} = \begin{pmatrix} r \\ h \end{pmatrix} \tag{2.38}$$

avendo introdotto la matrice diagonale

$$\Theta = XS^{-1} = S^{-1}X \quad \Longrightarrow \quad \Theta^{-1} = X^{-1}S.$$

[22]Detta 'long step path following', diversamente del precedente che è detto 'short step', [41, cap5, pag. 86 e 96].

[23]Aumentato della variabile $y \in R^m$, rispetto al problema originale nella sola variabile $x \in R^n$.

[24]Se si usano metodi diretti di soluzione, la fattorizzazione (sparsa) di Cholesky non può essere usata. Vari algoritmi sono stati proposti, ex: Bunch-Parlett, [41, cap11, pag. 222].

Usando la seconda equazione, una ulteriore eliminazione del vettore

$$\Delta x = \Theta A^T \ \Delta y - \Theta \ r$$

dalla prima equazione, conduce al sistema delle **equazioni normali** di dimensione m che è *simmetrico e definito positivo*[25]

$$(A \ \Theta \ A^T) \ \Delta y = A\Theta \ r + h = g.$$

Infatti: (per $\xi_p = h = 0$)

$$A\Theta A^T \Delta y = A\Theta \ r$$

$$\underbrace{A\Theta^{1/2}}_{B^T}\underbrace{\Theta^{1/2}A^T}_{B}\Delta y = \underbrace{A\Theta^{1/2}}_{B^T}\underbrace{\Theta^{1/2}r}_{d}$$

da cui

$$B^T B \ \Delta y = B^T d,$$

che sono le classiche equazioni normali per i sistema sovradeterminato $(n \times m)$ $B\Delta y = d$.

Algoritmo IP: Caso lineare

Inizializzazione
k=0
$(x^0, y^0, s^0) > 0$
$\mu^0 = \mu_0 = \frac{1}{n}(x^0)^T s^0$
$\alpha_0 = 0.9995$

Repeat until optimality
$\mu^k = \sigma \mu^{k-1}, \qquad 0 < \sigma < 1$
Δ = Calcolo della direzione di Newton verso il μ–centro

Line search
$\alpha_P = \max\{\alpha > 0 : x + \alpha \Delta x \geq 0\}$
$\alpha_D = \max\{\alpha > 0 : s + \alpha \Delta s \geq 0\}$

Calcolo delle nuove variabili Primale-Duale:
$x^{k+1} = x^k + \alpha_0 \alpha_P \Delta x$
$y^{k+1} = y^k + \alpha_0 \alpha_P \Delta y$
$s^{k+1} = s^k + \alpha_0 \alpha_D \Delta s$

[25]La fattorizzazione (sparsa) di Cholesky $A\Theta A^T = LDL^T$ o il metodo del gradiente coniugato sono entrambi applicabili.

4"'.

Soluzione del sistema IP (2.37) o (2.38) per eliminazione (gaussiana), **metodi diretti** (i più usati[26]):
[41, (1.23) e (1.25)], sistema aumentato ed equazioni normali con il metodo di Mehrotra *primal-dual predictor corrector*: *predictor,* $\sigma = 0$ caso affine, e *corrector,* $\sigma = 1$ caso della direzione di centramento, cfr. [37, (2.5)-(2.6)], [41, pag. 92 e cap. 10].

Nei metodi path-following, la direzione utilizzata è una combinazione convessa di questi due casi estremi.

4"".

Soluzione iterativa ddel sistema IP: **metodi iterativi**[27]**,** metodi Newton-Inesatti[28] applicati a problemi più generali (extensions) [41, pag. 13 e cap. 8]. In particolare: al *problema di complementarietà non lineare misto* [2, pag. 10], Ferris sezioni 1 e 2, dove esso è preso a *prototipo* di tutti gli altri problemi:

$$\begin{aligned} f(v) &= w - u \\ v_l \le v &\le v_u \\ (v - v_l)^T w &= 0 \\ (v_u - v)^T u &= 0 \\ v \in R^N,\ w, u &\in R^N_+ \ . \end{aligned} \tag{2.39}$$

Posto $x = (w, u)^T \ge 0$, $s = (v - v_l, v_u - v)^T \ge 0$, il problema si può scrivere nella forma standard

$$\begin{aligned} F(v, s, x) = f(v) - w + u &= 0 \\ XSe &= 0 \\ s \ge 0,\ x &\ge 0, \end{aligned} \tag{2.40}$$

che (in questo caso) è un sistema non lineare di $5N$ equazioni:
le $N + 2N = 3N$ equazioni $F(v, s, x) = 0$ (in generale non lineari) e le $2N$ equazioni di complementarietà $XSe = 0$. Si veda l'esempio (Bratu) in 2.6.4.

[26]Cfr., tra gli altri, il codice HOPDM [21].

[27]Per soluzione con metodi iterativi, opportunamente precondizionati, applicati a problemi quadratici di grandi dimensioni cfr. i recenti [4], [5] e [3], dove di dimostra che essi possono essere molto più vantaggiosi del classico approccio con metodi diretti.

[28]Vedi le Tesi di dottorato [2] e [16] e [9].

2.6.2 Estensioni

Il metodo interior point primale-duale si estende facilmente a problemi di Programmazione convessa-quadratica, cfr. [34, sez. 16.7]. L'estensione a problemi convessi e più generalmente non convessi costituisce attuale tema di ricerca[29].

Problema quadratico in forma standard (2.15)

$$\begin{aligned} \min\ & c^T x + \tfrac{1}{2}\ x^T Q x \\ \text{tale che}\quad & Ax = b, \\ & x \geq 0, \end{aligned}$$

dove $Q \in \mathcal{R}^{n\times n}$ è una matrice semidefinita positiva, $A \in \mathcal{R}^{m\times n}$ è la matrice, di rango massimo m, dei vincoli lineari e i vettori x, $c \in R^n$ e $b \in R^m$.

• Rimpiazziamo i vincoli di disuguaglianza con la barriera logaritmica

$$\min\ \frac{1}{2}\ c^T x + x^T Q x - \mu \sum_{i=1}^{n} \ln x_i.$$

• La lagrangiana associata è:

$$L(x, \mu, y) = c^T x\ + \frac{1}{2}\ x^T Q x\ -\ y^T(Ax - b)\ - \mu \sum_{i=1}^{n} \ln x_i,$$

con $\mu > 0$ (parametro barriera).

• Derivando, si ottengono le condizioni di stazionarietà:

$$\begin{aligned} \nabla_x L &= c - A^T y - \mu X^{-1} e + Qx = 0 \\ \nabla_y L &= Ax - b = 0, \end{aligned}$$

dove al solito $X = diag(x_1, x_2, \ldots, x_n)$ ed $e = (1, \ldots, 1)^T$.
Ponendo $\mu X^{-1} e = s$, cioè

$$s_i = \frac{\mu}{x_i},\ \forall i \quad \Longrightarrow \quad XSe = \mu e,$$

si ottiene il sistema quadratico interior point:

$$\begin{aligned} Ax &= b \\ A^T y + s - Qx &= c \\ XSe &= \mu e. \\ (x, s) &\geq 0 \end{aligned} \tag{2.41}$$

[29] Cfr. per esempio il talk [38] e segg.

• Si applica ora il metodo di Newton al sistema (2.41), riducendo gradualmente il parametro barriera μ, per garantire la convergenza. Si ottiene:

$$\begin{pmatrix} A & 0 & 0 \\ -Q & A^T & I \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta s \end{pmatrix} = \begin{pmatrix} \xi_p \\ \xi_d \\ \xi_\mu \end{pmatrix} \tag{2.42}$$

Come nel caso lineare, usando la terza equazione, si può eliminare il vettore

$$\Delta s = X^{-1}(\xi_\mu - S\Delta x) = -X^{-1}S\Delta x + X^{-1}\xi_\mu$$

dalla seconda equazione, ottenendo il **sistema aumentato** $(n + m)$, che è simmetrico e indefinito:

$$\begin{pmatrix} -Q - \Theta^{-1} & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} \xi_d - X^{-1}\xi_\mu \\ \xi_p \end{pmatrix} \tag{2.43}$$

Nel caso quadratico in esame, *non conviene* utilizzare le equazioni normali

$$(A\ (Q + \ \Theta^{-1})\ A^T)\ \Delta y = b_{qp}$$

che richiederebbero l'inversione della matrice hessiana Q.

5.

L'algoritmo IP per il caso quadratico è analogo all'algoritmo visto nel caso lineare.

2.6.3 Implementazione ed Esempi

Alla fine della sezione 2.6.4 è riportato un codice scritto in MatLab che implementa l'algoritmo IP visto in sezione 2.6, per problemi di programmazione lineare e quadratica.
In esso sono presentati cinque esempi (bidimensionali) di programmazione lineare, 1–5, e cinque esempi (bidimensionali) di programmazione quadratica, 11-15.
Come esempio di default lineare è stato scelto l'esempio n. 4 e come esempi di default quadratici sono stati scelti il n. 14 e il n. 15.

Un esempio lineare (esempio 4 nel file MatLab)

In questo esempio si applica l'algoritmo visto al seguente problema di ottimizzazione lineare al fine di visualizzare il funzionamento del metodo IP.

$$\begin{aligned} \max \quad & x_1 + 2x_2 \\ s.t. \quad & x_1 + x_2 \leq 8 \\ & 2x_1 + x_2 \leq 12 \\ & 0 \leq x_1 \leq 5 \\ & 0 \leq x_2 \leq 5\,. \end{aligned}$$

Come prima fase dell'algoritmo, si deve riscrivere il problema di ottimizzazione nella forma standard.
Trasformando il problema di massimo in un problema di minimo e introducendo le variabili slack x_3, x_4, x_5 e x_6 per convertire i vincoli di disuguaglianza in vincoli di uguaglianza, possiamo riscrivere il problema precedente come:

$$\begin{aligned} \min \quad & -x_1 - 2x_2 = c^T x \\ s.t. \quad & x_1 + x_2 + x_3 = 8 \\ & 2x_1 + x_2 + x_4 = 12 \\ & x_1 + x_5 = 5 \\ & x_2 + x_6 = 5 \\ & x_1, x_2, x_3, x_4, x_5, x_6, \geq 0\,. \end{aligned}$$

Da questa formulazione riconosciamo le varie matrici coinvolte nel metodo di punto interno. In particolare si ha

$$c = \begin{bmatrix} -1 & -2 & 0 & 0 & 0 & 0 \end{bmatrix}$$

e

$$A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 8 \\ 12 \\ 5 \\ 5 \end{bmatrix}.$$

In Figura 2.11 è riportato l'andamento del metodo per diversi valori del parametro σ (parametro di centramento). Accanto alla soluzione trovata ad ogni passo (cerchi neri) è sovrapposta in curva continua la soluzione del central path.

Dalla figura risulta chiaro che il parametro σ gioca un ruolo fondamentale nel passo di avanzamento dell'algoritmo. Se il valore è grande, cioè vicino

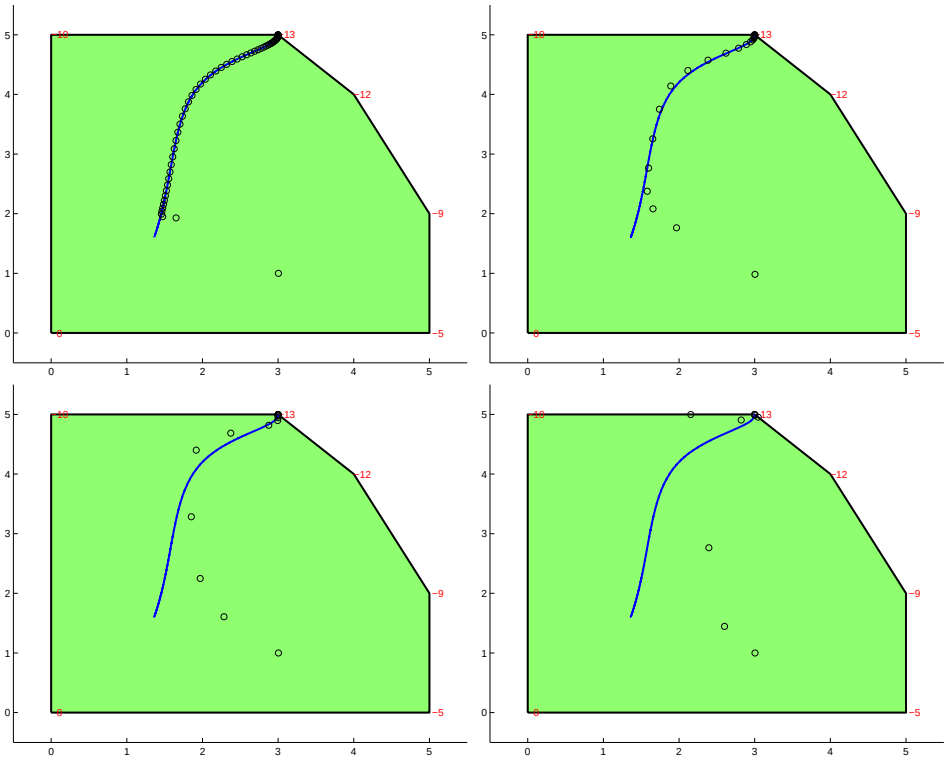


Figura 2.11: Metodo del punto interno (caso lineare) per diversi valori di $\sigma = (0.9, 0.7, 0.5, 0.3)$.

all'unità, l'algoritmo produrrà piccoli (e più numerosi) passi vicino al central path (direzione di centramento), mentre, se il parametro σ è troppo piccolo, cioè vicino allo zero, allora l'algoritmo produrrà sì grandi (e meno numerosi) passi di avanzamento (direzione di Newton) ma rischia di andare vicino alla frontiera del dominio con possibile perdita della convergenza.

Un esempio quadratico (esempio 14 nel file MatLab)

In questo esempio si applica l'algoritmo visto al seguente problema di ottimizzazione quadratico al fine di visualizzare il funzionamento del metodo IP.

$$\begin{aligned} \max \quad & -[x_1^2 + 0.2x_1x_2 + x_2^2] + [10x_1 + 10x_2] \\ s.t. \quad & x_1 + x_2 \leq 8 \\ & 2x_1 + x_2 \leq 12 \\ & 0 \leq x_1 \leq 5 \\ & 0 \leq x_2 \leq 5\,. \end{aligned}$$

Come prima fase dell'algoritmo, si deve riscrivere il problema di ottimizzazione nella forma standard.
Trasformando il problema di massimo in un problema di minimo e introducendo le variabili slack x_3, x_4, x_5 e x_6 per convertire i vincoli di disuguaglianza in vincoli di uguaglianza, possiamo riscrivere il problema precedente come:

$$\begin{aligned} \min \quad & \frac{1}{2}[2x_1^2 + 0.4x_1x_2 + 2x_2^2] - [10x_1 + 10x_2] = \frac{1}{2}\, x^T Q x + c^T x \\ s.t. \quad & x_1 + x_2 + x_3 = 8 \\ & 2x_1 + x_2 + x_4 = 12 \\ & x_1 + x_5 = 5 \\ & x_2 + x_6 = 5 \\ & x_1, x_2, x_3, x_4, x_5, x_6, \geq 0\,. \end{aligned}$$

Da questa formulazione riconosciamo le varie matrici coinvolte nel metodo di punto interno. In particolare si ha

$$Q = \begin{bmatrix} 1 & 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0.1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad c = \begin{bmatrix} -1 \\ -2 \\ 0 \\ 0 \\ 0 \\ 5 \end{bmatrix}$$

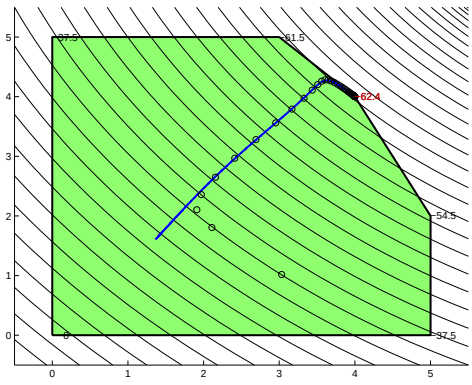


Figura 2.12: Metodo del punto interno (caso quadratico n. 14) per $\sigma = 0.7$.

e

$$A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 8 \\ 12 \\ 5 \\ 5 \end{bmatrix}$$

In Figura 2.12 è riportato l'andamento del metodo per $\sigma = 0.7$ (parametro di centramento). Accanto alla soluzione trovata ad ogni passo (cerchi neri) è sovrapposta in curva continua la soluzione del central path.

Un esempio quadratico (esempio 15 nel file MatLab)

In questo esempio si applica l'algoritmo visto al seguente problema di ottimizzazione quadratico al fine di visualizzare il funzionamento del metodo IP.

$$\begin{aligned} \min \quad & \frac{1}{2}[x_1^2 - 2x_1x_2 + 2x_2^2] + [-2x_1 - 6x_2] = \frac{1}{2}\, x^T Q x + c^T x \\ s.t. \quad & x_1 + x_2 \leq 2 \\ & -x_1 + 2x_2 \leq 2 \\ & 2x_1 + x_2 \leq 3 \\ & 0 \leq x_1 \\ & 0 \leq x_2 \end{aligned}$$

Introducendo le variabili slack x_3, x_4, x_5 per convertire i vincoli di disuguaglianza in vincoli di uguaglianza, possiamo riscrivere il problema precedente

come:

$$\begin{aligned} \min \quad & \frac{1}{2}[x_1^2 - 2x_1x_2 + 2x_2^2] + [-2x_1 - 6x_2] = \frac{1}{2}\, x^T Q x + c^T x \\ & x_1 + x_2 + x_3 = 2 \\ & -x_1 + 2x_2 + x_4 = 2 \\ & 2x_1 + x_2 + x_5 = 3 \\ & x_1, x_2, x_3, x_4, x_5, \geq 0\,. \end{aligned}$$

Da questa formulazione riconosciamo le varie matrici coinvolte nel metodo di punto interno. In particolare si ha

$$Q = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad c = \begin{bmatrix} -2 \\ -6 \\ 0 \\ 0 \end{bmatrix}$$

e

$$A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ -1 & 2 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ 2 \\ 3 \end{bmatrix}$$

In Figura 2.13 è riportato l'andamento del metodo per diversi valori del parametro σ (parametro di centramento). Accanto alla soluzione trovata ad ogni passo (cerchi neri) è sovrapposta in curva continua la soluzione del central path.

Dalla Figura 2.13 risulta chiaro, come già visto nel caso lineare, che il parametro σ gioca un ruolo fondamentale nel passo di avanzamento dell'algoritmo. Se il valore è grande, cioè vicino all'unità, l'algoritmo produrrà piccoli passi vicino al central path (direzione di centramento), mentre, se il parametro σ è troppo piccolo, cioè vicino allo zero, allora l'algoritmo produrrà sì grandi passi di avanzamento (direzione di Newton) ma rischia di andare vicino alla frontiera del dominio con possibile perdita della convergenza.

2.6.4 Problema Bratu

Come **Esempio** di programmazione quadratica vediamo il classico *Problema Bratu* (ostacle problem con forza $=\lambda e^v$)[30].

[30] Ferris pag. 21, [2].

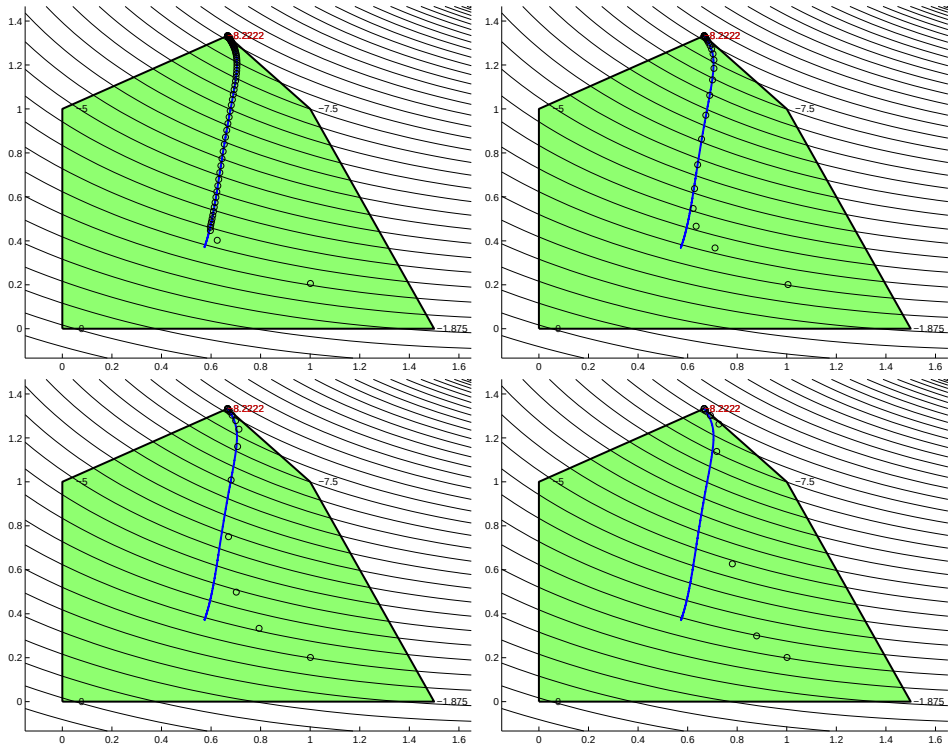


Figura 2.13: Metodo del punto interno (caso quadratico n. 15) per diversi valori di $\sigma = (0.9, 0.7, 0.5, 0.3)$.

Inizialmente, esso è un problema di programmazione quadratica (2.15) (funzione obiettivo quadratica e vincoli lineari) ma con due vincoli sulla variabile v (ostacoli v_l, v_u), che una volta discretizzato[31] si scrive

$$\min q(v) = \frac{1}{2} v^T M v - q^T v, \quad v_l \leq v \leq v_u, \ v \in H_0^1(D), \tag{2.44}$$

dove M è la solita matrice a 5 elementi non nulli per riga $[4, -1, -1, -1 - 1]$ (pentadiagonale!) ottenuta discretizzando il laplaciano con differenze finite a 5 punti di passo $h = 1/(N+1)$ lungo i due assi, con $n = N^2$ e $D = [0,1] \times [0,1]$. In generale $q = q(v)$; esempio $q = h^2 \times cost$, oppure $q = h^2 \times \lambda e^v$ (Bratu).

Essendo la lagrangiana relativa al problema (2.44)

$$L(v, u, w) = q(v) - w^T(v - v_l) - u^T(v_u - v),$$

scrivendo le condizioni KKT, si arriva al problema di complementarietà non lineare misto, cfr. (2.39)

$$\begin{aligned} \nabla_v L = f(v) - w + u = (Mv - q) - w + u = 0, \quad (q = h^2 \lambda e^v) \\ (v - v_l) \geq 0, \ (v_u - v) \geq 0 \\ (v - v_l)^T w = 0, \\ (v_u - v)^T u = 0 \\ v \in R^n; w, u \ \in R^n_+, \ w \geq 0, u \geq 0. \end{aligned} \tag{2.45}$$

Nel caso del Bratu problem chiaramente si ha

$$f(v_{i,j}) = (4v_{i,j} - v_{i+1,j} - v_{i-1,j} - v_{i,j+1} - v_{i,j-1}) - h^2 \lambda e^{v_{i,j}}, \quad 1 \leq i, j \leq N.$$

In totale si hanno 5 vettori (v, w, u, v_l, v_u) incogniti di dimensione n ciascuno. Tre diversi tipi di ostacoli sono proposti in Ferris e [2]

Scriviamo per disteso le equazioni (2.45) nel caso particolare $n = 2$.

Poniamo $x = (w, u)^T$, $s = (v - v^l, v^u - v)^T$, con i due ostacoli (che qui indichiamo con) $v^l \leq v \leq v^u$. Sia

$$v = (v_1, v_2)^T, \quad w = (w_1, w_2)^T, \quad u = (u_1, u_2)^T,$$

e ovviamente

$$v - v^l = (v_1 - v_1^l, v_2 - v_2^l)^T, \quad v^u - v = (v_1^u - v_1, v_2^u - v_2)^T.$$

[31] Si veda Ferris pag. 21 e [44].

Sia inoltre

$$W = diag(w_1, w_2), \quad U = diag(u_1, u_2),$$
$$V^l = diag(v_1 - v_1^l, v_2 - v_2^l), \quad V^u = diag(v_1^u - v_1, v_2^u - v_2).$$

Allora il sistema non lineare (di Newton non perturbato) è

$$\begin{aligned} Mv - h^2\lambda I e^v - w + u &= 0 \\ w^T(v - v^l) &= 0 \\ u^T(v^u - v) &= 0 \end{aligned}$$

da cui (I.P.)

$$\begin{aligned} Mv - h^2\lambda I e^v - w + u &= 0 \\ WV^l e &= \sigma\mu e \\ UV^u e &= \sigma\mu e. \end{aligned}$$

Essendo $\Delta(v - v^l) = \Delta v$, $\Delta(v^u - v) = -\Delta v$, il sistema Inexact Interior Point (IIP) per il problema Bratu è (al passo k):

$$\begin{pmatrix} M - h^2\lambda I e^v & -I & I \\ W & V^l & 0 \\ -U & 0 & V^u \end{pmatrix}_k \begin{pmatrix} \Delta v \\ \Delta w \\ \Delta u \end{pmatrix} = - \begin{pmatrix} r_v \\ r_w \\ r_u \end{pmatrix}_k$$

dove r_v, r_w, r_u sono i residui. Per disteso

$$\begin{pmatrix} m_{11} - h^2\lambda e^{v_1} & m_{12} & -1 & 0 & 1 & 0 \\ m_{12} & m_{22} - h^2\lambda e^{v_2} & 0 & -1 & 0 & 1 \\ w_1 & 0 & v_1 - v_1^l & 0 & 0 & 0 \\ 0 & w_2 & 0 & v_2 - v_2^l & 0 & 0 \\ -u_1 & 0 & 0 & 0 & v_1^u - v_1 & 0 \\ 0 & -u_2 & 0 & 0 & 0 & v_2^u - v_2 \end{pmatrix}_k \begin{pmatrix} \Delta v_1 \\ \Delta v_2 \\ \Delta w_1 \\ \Delta w_2 \\ \Delta u_1 \\ \Delta u_2 \end{pmatrix}$$
$$= - \begin{pmatrix} m_{11}v_1 - h^2\lambda e^{v_1} + m_{12}v_2 - w_1 + u_1 \\ m_{21}v_1 - h^2\lambda e^{v_2} + m_{22}v_2 - w_2 + u_2 \\ w_1(v_1 - v_1^l) - \sigma\mu \\ w_2(v_2 - v_2^l) - \sigma\mu \\ u_1(v_1^u - v_1) - \sigma\mu \\ u_2(v_2^u - v_2) - \sigma\mu \end{pmatrix}_k .$$

```
% ESEMPIO METODO INTERIOR POINT (lineare e quadratico)
% ----------------------------------------------------------------
% Questo file *.m contiene vari esempi che mostrano il
% funzionamento del metodo interior point (Dispensa,sez. 2.6)
% per problemi bi-dimensionali lineari e quadratici .
%
% Gli esempi lineari vanno dal numero 1 al numero 5,
% quelli quadratici vanno dal numero 11 al numero 15.
% Gli esempi si possono scegliere cambiando il numero di scelta
% nello switch iniziale del file (esempio = numero).
%
% Per usare il programma con un altro esempio bisogna aggiungere
% un nuovo case allo switch con le opportune definizioni delle
% nuove variabili (f.ne obiettivo, vincoli, p.to iniziale,...).
%
% Si richiede che il problema sia definito in forma standard:
%   - Problemi lineari    : min c'x, Ax=b, x >=0.
%   - Problemi quadratici : min 1/2 x'Qx + c'x, Ax=b, x >=0.
%
% La scelta del punto iniziale viene effettuata in modo
% interattivo (da tastiera), se non  viene specificato nessun
% punto iniziale nello switch.
%
% Alla fine viene stampata la soluzione ottima (output)
%   - xopt (soluzione)
%   - yopt (parametri lagrangiani).
%
% Riferimento bibliografico:
% G. Zilli, L. Bergamaschi, M. Venturin
% Metodi di Ottimizzazione (sez. 2.6)
% DMMMSA (2008)

% Pulizia dello schermo e delle variabili
% ---------------------------------------
close all
clear all
clc
format long g
format compact

% Alcune inizializzazione standard
% --------------------------------
% Inizializzazione matrici interior point vuote
Q = []; c = []; A = []; b = [];
% Punto iniziale vuoto (scelta visuale)
xinit = []; yinit = []; sinit = [];

% Identificativo dell'esempio da provare
% --------------------------------------
```

```
% Casi lineari    : da  1 a  5
% Casi quadratici : da 11 a 15
esempio = 15;
switch esempio
    case 1
        % Titolo dell'esempio e descrizione del problema
        disp('massimo su un triangolo');
        disp('max  x1 + 2x2');
        disp('s.t. x1 +  x2 ≤ 3');
        disp('     x1 ,  x2 ≥ 0');
        % Funzione obiettivo in forma standard: forma lineare
        c = [−1 −2 0]';
        % Vincoli del dominio: A x = b
        A = [1 1 1];
        b = [3]';
        % Coordinate dei vertici del dominio (senso antiorario)
        xv = [0 3 0];
        yv = [0 0 3];
        % Scelta visuale del punto iniziale (visuale)
        %
        % Soluzione trovata con le routine di MatLab
        disp('Soluzione trovata con MatLab');
        [XSOL,FVALSOL,EXITFLAGSOL,OUTPUTSOL,LAMBDASOL] = ...
        linprog([−1 −2],[1 1],[3],[],[],[0 0],[Inf Inf])

    case 2
        % Titolo dell'esempio e descrizione del problema
        disp('massimo su un quadrato');
        disp('max  x1 + x2');
        disp('s.t. x1      ≤ 1');
        disp('          x2 ≤ 1');
        disp('     x1 , x2 ≥ 0');
        % Funzione obiettivo in forma standard: forma lineare
        c = [−1 −1 0 0]';
        % Vincoli del domnio: A x = b
        A = [1 0 1 0; 0 1 0 1];
        b = [1 1]';
        % Coordinate dei vertici del dominio (senso antiorario)
        xv = [0 1 1 0];
        yv = [0 0 1 1];
        % Scelta visuale del punto iniziale (visuale)
        %
        % Soluzione trovata con le routine di MatLab
        disp('Soluzione trovata con MatLab');
        [XSOL,FVALSOL,EXITFLAGSOL,OUTPUTSOL,LAMBDASOL] = ...
        linprog([−1 −1],[],[],[],[],[0 0],[1 1])

    case 3
        % Titolo dell'esempio e descrizione del problema
```

```
99          disp('minimo su un lato di un dominio generico');
100         disp('min  −x1 − 2x2');
101         disp('s.t.  x1 + x2 ≤ 8');
102         disp('     2x1 + x2 ≤ 12');
103         disp('      x1      ≤ 5');
104         disp('           x2 ≤ 5');
105         disp('      x1 , x2 ≥ 0');
106         % Funzione obiettivo in forma standard: forma lineare
107         c = [−1 −2 0 0 0 0]';
108         % Vincoli del domnio: A x = b
109         A=[1 1 1 0 0 0; 2 1 0 1 0 0; 1 0 0 0 1 0; 0 1 0 0 0 1];
110         b = [8 12 5 5]';
111         % Coordinate dei vertici del dominio (senso antiorario)
112         xv = [0 5 5 4 3 0];
113         yv = [0 0 2 4 5 5];
114         % Scelta visuale del punto iniziale (visuale)
115         %
116         % Soluzione trovata con le routine di MatLab
117         disp('Soluzione trovata con MatLab');
118         [XSOL,FVALSOL,EXITFLAGSOL,OUTPUTSOL,LAMBDASOL] = ...
119         linprog([−1 −2],[1 1;2 1],[8;12],[],[],[0 0],[5 5])
120
121     case 4
122         % Titolo dell'esempio e descrizione del problema
123         disp('minimo su un lato di un dominio generico');
124         disp('min −10x1 − 10x2');
125         disp('s.t.   x1 +   x2 ≤ 8');
126         disp('      2x1 +   x2 ≤ 12');
127         disp('       x1       ≤ 5');
128         disp('              x2 ≤ 5');
129         disp('       x1 ,   x2 ≥ 0');
130         % Funzione obiettivo in forma standard: forma lineare
131         c = [−10 −10 0 0 0 0]';
132         % Vincoli del domnio: A x = b
133         A = [1 1 1 0 0 0; 2 1 0 1 0 0; 1 0 0 0 1 0; 0 1 0 0 0 1];
134         b = [8 12 5 5]';
135         % Coordinate dei vertici del dominio (senso antiorario)
136         xv = [0 5 5 4 3 0];
137         yv = [0 0 2 4 5 5];
138         % Scelta visuale del punto iniziale (visuale)
139         %
140         % Soluzione trovata con le routine di MatLab
141         disp('Soluzione trovata con MatLab');
142         [XSOL,FVALSOL,EXITFLAGSOL,OUTPUTSOL,LAMBDASOL] = ...
143         linprog([−10 −10],[1 1;2 1],[8;12],[],[],[0 0],[5 5])
144     case 5
145         % Titolo dell'esempio e descrizione del problema
146             disp('minimo su un dominio generico');
147         disp('min  −2x1 − 6x2');
```

```
148         disp('s.t.  x1  +  x2 ≤ 2');
149         disp('     −x1  + 2x2 ≤ 2');
150         disp('     2x1  +  x2 ≤ 3');
151         disp('      x1 , x2 ≥ 0');
152         % Funzione obiettivo in forma standard: forma lineare
153         c = [−2 −6 0 0 0]';
154         % Vincoli del domnio: A x = b
155         A = [1 1 1 0 0 ; −1 2 0 1 0; 2 1 0 0 1];
156         b = [2 2 3]';
157         % Coordinate dei vertici del dominio (senso antiorario)
158         xv = [0 3/2 1 2/3 0];
159         yv = [0 0   1 4/3 1];
160         % Scelta visuale del punto iniziale (visuale)
161         %
162         % Soluzione trovata con le routine di MatLab
163         disp('Soluzione trovata con MatLab');
164         [XSOL,FVALSOL,EXITFLAGSOL,OUTPUTSOL,LAMBDASOL] = ...
165         linprog([−2 −6],[1 1;−1 2;2 1],[2;2;3],[],[],[0 0],[Inf Inf])
166
167     case 11
168         % Titolo dell'esempio e descrizione del problema
169         disp('minimo su un triangolo−forma quadratica diagonale');
170         disp('min  1/2 * [ 10x1^2 + 2x2^2 ] −x1−2x2');
171         disp('s.t. x1 +  x2 ≤ 3');
172         disp('     x1 ,  x2 ≥ 0');
173         % Funzione obiettivo in forma standard: forma quadratica
174         Q = [10,0,0;0,2,0;0,0,0];
175         % Funzione obiettivo in forma standard: forma lineare
176         c = [−1 −2 0]';
177         % Vincoli del dominio: A x = b
178         A = [1 1 1];
179         b = [3]';
180         % Coordinate dei vertici del dominio (senso antiorario)
181         xv = [0 3 0];
182         yv = [0 0 3];
183         % Scelta visuale del punto iniziale (visuale)
184         %
185         % Soluzione trovata con le routine di MatLab
186         disp('Soluzione trovata con MatLab');
187         [XSOL,FVALSOL,EXITFLAGSOL,OUTPUTSOL,LAMBDASOL] = ...
188         quadprog([10 0;0 2],[−1,−2],[1 1],[3],[],[],[0 0],[Inf Inf])
189
190     case 12
191         % Titolo dell'esempio e descrizione del problema
192         disp('minimo su un lato del quadrato−forma quadratica diagonale');
193         disp('min  1/2 * [ .1x1^2 + 30x2^2 ] −x1 −x2');
194         disp('s.t. x1      ≤ 1');
195         disp('          x2 ≤ 1');
196         disp('     x1 , x2 ≥ 0');
```

```
197         % Funzione obiettivo in forma standard: forma quadratica
198         Q = [.1,0,0,0;0,30,0,0;0,0,0,0;0,0,0,0];
199         % Funzione obiettivo in forma standard: forma lineare
200         c = [-1 -1 0 0]';
201         % Vincoli del dominio: A x = b
202         A = [1 0 1 0; 0 1 0 1];
203         b = [1 1]';
204         % Coordinate dei vertici del dominio (senso antiorario)
205         xv = [0 1 1 0];
206         yv = [0 0 1 1];
207         % Scelta visuale del punto iniziale (visuale)
208         %
209         % Soluzione trovata con le routine di MatLab
210         disp('Soluzione trovata con MatLab');
211         [XSOL,FVALSOL,EXITFLAGSOL,OUTPUTSOL,LAMBDASOL] = ...
212         quadprog([.1 0;0 30],[-1 -1]',[],[],[],[],[0 0],[1 1])
213
214     case 13
215       % Titolo dell'esempio e descrizione del problema
216       disp('minimo su un dominio generico-forma quadratica generale');
217       disp('min  1/2 * [ .1x1^2 +x1*x2 +30x2^2 ] -x1 -2x2');
218       disp('s.t.  x1 + x2 ≤ 8');
219       disp('     2x1 + x2 ≤ 12');
220       disp('      x1      ≤ 5');
221       disp('           x2 ≤ 5');
222       disp('      x1 , x2 ≥ 0');
223        % Funzione obiettivo in forma standard: forma quadratica
224        Q = [.1,1,0,0,0;1,30,0,0,0;zeros(3,5)];
225        % Funzione obiettivo in forma standard: forma lineare
226         c = [-1 -1 0 0 0]';
227         % Vincoli del dominio: A x = b
228         A = [1 1 1 0 0; 1 0 0 1 0; 0 1 0 0 1];
229         b = [3 2 2]';
230         % Coordinate dei vertici del dominio (senso antiorario)
231         xv = [0 2 2 1 0];
232         yv = [0 0 1 2 2];
233         % Scelta visuale del punto iniziale (visuale)
234         %
235         % Soluzione trovata con le routine di MatLab
236         disp('Soluzione trovata con MatLab');
237         [XSOL,FVALSOL,EXITFLAGSOL,OUTPUTSOL,LAMBDASOL] = ...
238         quadprog([.1 1;1 2],[-1 -2],[1 1;2 1],[8;12],[],[],[0 0],[5 5])
239
240     case 14
241         % Titolo dell'esempio e descrizione del problema
242         disp('minimo su un dominio generico-forma quadratica generale');
243         disp('min  1/2 * [ x1^2 +0.2x1*x2 +x2^2 ] -10x1 - 10x2');
244         disp('s.t.   x1 +   x2 ≤ 8');
245         disp('      2x1 +   x2 ≤ 12');
```

```
246         disp('       x1        ≤ 5');
247         disp('              x2 ≤ 5');
248         disp('       x1 ,   x2 ≥ 0');
249         % Funzione obiettivo in forma standard: forma quadratica
250         Q = [1,.1,0,0,0,0;.1,1,0,0,0,0;zeros(4,6)];
251         % Funzione obiettivo in forma standard: forma lineare
252         c = [-10 -10 0 0 0 0]';
253         % Vincoli del dominio: A x = b
254         A = [1 1 1 0 0 0; 2 1 0 1 0 0; 1 0 0 0 1 0; 0 1 0 0 0 1];
255         b = [8 12 5 5]';
256         % Coordinate dei vertici del dominio (senso antiorario)
257         xv = [0 5 5 4 3 0];
258         yv = [0 0 2 4 5 5];
259         % Scelta visuale del punto iniziale (visuale)
260         %
261         % Soluzione trovata con le routine di MatLab
262         disp('Soluzione trovata con MatLab');
263         [XSOL,FVALSOL,EXITFLAGSOL,OUTPUTSOL,LAMBDASOL] = ...
264         quadprog([1 .1;.1 1],[-10 -10],[1 1;2 1],[8;12],[],[],[0 0],[5 5])
265
266      case 15
267         % Titolo dell'esempio e descrizione del problema
268         disp('minimo Matlab pag. 4-134');
269         disp('min  1/2 * [ x1^2 -2x1*x2 +2x2^2 ] -2x1 - 6x2');
270         disp('s.t.  x1  +  x2 ≤ 2');
271         disp('     -x1  + 2x2 ≤ 2');
272         disp('     2x1  +  x2 ≤ 3');
273         disp('      x1 , x2 ≥ 0');
274         % Funzione obiettivo in forma standard: forma quadratica
275         Q = [1,-1,0,0,0;-1,2,0,0,0;zeros(3,5)];
276         % Funzione obiettivo in forma standard: forma lineare
277         c = [-2 -6 0 0 0]';
278         % Vincoli del dominio: A x = b
279         A = [1 1 1 0 0 ; -1 2 0 1 0; 2 1 0 0 1];
280         b = [2 2 3]';
281         % Coordinate dei vertici del dominio (senso antiorario)
282         xv = [0 3/2 1 2/3 0];
283         yv = [0 0   1 4/3 1];
284         % Scelta visuale del punto iniziale (visuale)
285         %
286         % Soluzione trovata con le routine di MatLab
287         disp('Soluzione trovata con MatLab');
288  [XSOL,FVALSOL,EXITFLAGSOL,OUTPUTSOL,LAMBDASOL] = ...
289  quadprog([1 -1;-1 2],[-2 -6],[1 1;-1 2;2 1],[2;2;3],[],[],[0 0],[Inf Inf])
290
291      otherwise
292         error('Esempio non trovato!');
293 end
294
```

```
% Informazioni sul problema
% ---------------------------
% Numero delle variabili
n = length(c);
% Numero di vincoli
m = length(b);
% Tipo di problema (lineare: ptype==0, quadratico: ptype==1)
ptype = 1;
if isempty(Q)
    ptype = 0;
end

% Disegno del dominio con curve di livello
% ----------------------------------------
% Disegno del dominio
h = patch(xv,yv,[0.5,1,0.5]);
set(h,'LineWidth',2);
%
% Valutazione della funzione sui vertici del dominio
hold on;
% parte lineare
% lineare
c1 = c(1); c2 = c(2);
fval = c1*xv+c2*yv;
% parte quadratica
if ptype
    q11 = Q(1,1); q12 = Q(1,2);
    q21 = Q(2,1); q22 = Q(2,2);
    fval = fval+1/2*(xv.*(q11*xv+q12*yv)+yv.*(q21*xv+q22*yv));
end
% Disegno valori
hold on;
h = text(xv,yv,num2str(fval','%5.2f'));
set(h,'color','k','FontSize',14);
hold off
%
% Linee di livello
% Assi del dominio
axmin = min(xv); axmax = max(xv);
aymin = min(yv); aymax = max(yv);
xoffset = (axmax-axmin)/10; yoffset = (aymax-aymin)/10;
axmin = axmin-xoffset; axmax = axmax+xoffset;
aymin = aymin-yoffset; aymax = aymax+yoffset;
% Risoluzione per le curve di livello
resx = (axmax-axmin)/40; resy = (aymax-aymin)/40;
% Valore della funzione nel dominio
[xm,ym] = meshgrid(axmin:resx:axmax,aymin:resy:aymax);
if ptype
  % quadratica
```

```
344   optexpr=1/2*(xm.*(q11*xm+q12*ym)+ym.*(q21*xm+q22*ym))+(c1*xm+c2*ym);
345 else
346   % lineare
347   optexpr = c1*xm+c2*ym;
348 end
349 % Valori massimi e minimi della funzione
350 minopt = min(min(optexpr));
351 maxopt = max(max(optexpr));
352 % Disegno delle curve di livello con curve piu'
353 % fitte attorno al minimo della funzione
354 level = (linspace(0,sqrt(maxopt-minopt),40)).^2+minopt;
355 hold on;
356 contour(xm,ym,optexpr,level,'k');
357 hold off;
358 axis([axmin axmax aymin aymax]);
359 % axis square
360
361 % Scelta visuale del punto iniziale
362 % ----------------------------------------
363 if isempty(xinit)
364     disp('Scegli punto iniziale interno al dominio.')
365     % punto iniziale
366     x12 = ginput(1)
367     % punto rimanenti del dominio
368     xr = A(:,3:end)\(b-A(:,1:2)*x12');
369     % creazione del vettore iniziale
370     xinit = [x12,xr']';
371 end
372 if isempty(yinit)
373     yinit = ones(m,1)*(min(c)-1);
374 end
375 if isempty(sinit)
376     sinit = -A'*yinit+c;
377 end
378 % copia dati
379 x = xinit; y = yinit; s = sinit;
380
381 % Verifica della scelta del punto iniziale
382 % ----------------------------------------
383 if any(x<=0)
384     error('x deve essere > 0');
385 end
386 if any(s<=0)
387     error('s deve essere > 0');
388 end
389
390 % Parametri algoritmo interior point
391 % ----------------------------------
392 mu     = 1/n*(s'*x);         % mu
```

```
393 alpha0 = 0.9995;              % alpha0
394 sigma  = 0.7;                 % parametro di centramento
395 k      = 1;                   % iterazioni
396 tol    = 1e-08;               % tolleranza
397 kmax   = 1000;                % numero massimo di iterazioni
398 e      = ones(n,1);           % vettore unitario
399
400 % Ciclo principale algoritmo interior point
401 % ---------------------------------------------
402 % salvataggio della soluzione
403 xhist = x; yhist = y; shist = s;
404 while s'*x > tol & k<kmax
405
406     % disp(['iterazione ip: ',num2str(k)]);
407     % calcolo di mu
408     mu = sigma*mu;
409     % Matrice del metodo interior point
410     J = spalloc(2*n+m,2*n+m,2*nnz(A)+3*n);
411     J(1:m,1:n)                  = A;
412     J(m+1:m+n,n+1:n+m)          = A.';
413     J(m+1:m+n,n+m+1:2*n+m)      = speye(n,n);
414     J(n+m+1:2*n+m,1:n)          = spdiags(s,0,n,n);
415     J(n+m+1:2*n+m,n+m+1:2*n+m) = spdiags(x,0,n,n);
416     if ptype
417         J(m+1:m+n,1:n)          = -Q;
418     end
419
420     % vettore termine noto del sistema lineare
421     if ptype
422         F = [b-A*x; c-(A'*y)-s+Q*x; mu-x.*s];
423     else
424         F = [b-A*x; c-(A'*y)-s; mu-x.*s];
425     end
426
427     % calcolo della soluzione
428     xvect = J\F;
429     DELTAx = xvect(1:n);
430     DELTAy = xvect(n+1:n+m);
431     DELTAs = xvect(n+m+1:end);
432
433     % scelta di alphap
434     ind=find(DELTAx<0);
435     alphap = min(1,alpha0*min(-x(ind)./DELTAx(ind)));
436     if isempty(alphap)
437         alphap=1;
438     end
439
440     % scelta di alphad
441     ind=find(DELTAs<0);
```

```
442     alphad = min(1,alpha0*min(-s(ind)./DELTAs(ind)));
443     if isempty(alphad)
444         alphad=1;
445     end
446 
447     % aggiornamento dei vettori del sistema
448     x = x+alphap*DELTAx;
449     y = y+alphad*DELTAy;
450     s = s+alphad*DELTAs;
451 
452     % salvataggio del punto corrente per eventuale disegno
453     xhist = [xhist,x]; yhist = [yhist,y]; shist = [shist,s];
454 
455     % aggiornamento delle iterazioni
456     k = k+1;
457 end
458 
459 % Soluzione ottima del problema
460 % ---------------------------------
461 % numero iterazioni eseguite
462 disp(['Numero iterazioni nonlineari : ',num2str(k)]);
463 xopt = x;
464 yopt = y;
465 sopt = s;
466 % Terna interior point
467 % xopt,yopt,sopt
468 
469 % Calcolo del cammino centrale
470 % ---------------------------------
471 % Disabilito un warning di MatLab
472 warning off MATLAB:nearlySingularMatrix
473 % Soluzione al passo corrente
474 xsol = xopt; ysol = yopt; ssol = sopt;
475 % Numero punti della soluzione
476 np = 100;
477 inp = 1;
478 % Inizializzazione strutture
479 xchist = zeros(n,np);
480 ychist = zeros(m,np);
481 schist = zeros(n,np);
482 % salvataggio della soluzione
483 xchist(1:n,1) = xsol; ychist(1:m,1) = ysol; schist(1:n,1) = ssol;
484 % cicla su tutti i possibili mu applicando il metodo di Newton
485 %for muval=linspace(mu,10,500)
486 for muval=logspace(log10(mu),log10(100),np)
487     % Visualizza iterazione
488     % disp(['muval(',num2str(inp),')=',num2str(muval)]);
489     % Iterazioni massimi
490     iter = 0; itmax = 100;
```

```
491     % Soluzione al passo precedente
492     xold = xsol; sold = ssol; yold = ysol;
493
494     % Ciclo principale di Newton
495     while (iter≤itmax)
496         % incrementa contatore iterazioni
497         iter = iter+1;
498
499         % Matrice IP
500         J = zeros(n+m+n,n+m+n);
501         J(1:m,1:n)                 = A;
502         J(m+1:m+n,n+1:n+m)         = A.';
503         J(m+1:m+n,n+m+1:2*n+m)     = eye(n,n);
504         J(n+m+1:2*n+m,1:n)         = diag(sold);
505         J(n+m+1:2*n+m,n+m+1:2*n+m) = diag(xold);
506         if ptype
507             J(m+1:m+n,1:n)             = -Q;
508         end
509
510         % Vettore IP
511         if ptype
512           F=[b-A*xold; c-(A'*yold)-sold+Q*xold; muval-xold.*sold];
513         else
514           F=[b-A*xold; c-(A'*yold)-sold; muval-xold.*sold];
515         end
516
517         % Risolvi sistema lineare
518         xvect = J\F;
519         DELTAx = xvect(1:n);
520         DELTAy = xvect(n+1:n+m);
521         DELTAs = xvect(n+m+1:end);
522
523         % aggiorna soluzione
524          xsol = xold+DELTAx;
525          ysol = yold+DELTAy;
526          ssol = sold+DELTAs;
527
528         % verifico convergenza
529         if norm(xsol-xold,Inf)≤1e-8
530             break;
531         end
532     end
533
534     % salvataggio della soluzione
535     inp = inp+1;
536     xchist(1:n,inp) = xsol;
537     ychist(1:m,inp) = ysol;
538     schist(1:n,inp) = ssol;
539
```

```
540     % soluzione al passo precedente
541     xold = xsol; sold = ssol; yold = ysol;
542 end
543 % Riabilito un warning di MatLab
544 warning on
545 
546 % Disegno del cammino centrale ed andamento interior point
547 % ------------------------------------------------------------
548 hold on;
549 % Cammino centrale
550 h = plot(xchist(1,:),xchist(2,:),'b-');
551 set(h,'LineWidth',2);
552 % Andamento delle iterate ip
553 h=plot(xhist(1,:),xhist(2,:),'ko');
554 hold off;
555 % valore ottimo
556 xm = xopt(1); ym = xopt(2);
557 optval = (c1*xm+c2*ym);
558 if ptype
559   optval=optval + 1/2*(xm.*(q11*xm+q12*ym)+ym.*(q21*xm+q22*ym));
560 end
561 hold on;
562 % valore della soluzione
563 h = text(xm,ym,num2str(optval','%5.2f'));
564 set(h,'color','r','FontSize',14,'EdgeColor','red');
565 % punto di ottimo
566 text(axmax-(axmax-axmin)/20,aymax-(aymax-aymin)/20,...
567     [['x_1^* = ',num2str([xopt(1)],'%5.2f')];...
568     ['x_2^* = ',num2str([xopt(2)],'%5.2f')]],...
569     'color','k','HorizontalAlignment','right',...
570     'VerticalAlignment','top',...
571     'FontSize',10,'EdgeColor','k','BackgroundColor','w');
572 hold off;
573 
574 % Visualizzazione soluzione ottima
575 disp('Soluzione ottima :');
576 disp(['xopt(=XSOL) : ',num2str(xopt','%0.5f ')]);
577 disp(['yopt(=LAMBDASOL.ineqlin): ',num2str(-yopt','%0.5f ')]);
578 disp(['fopt(=FVALSOL) : ',num2str(optval','%0.5f ')]);
579 
580 format
```

Capitolo 3

Ottimizzazione multiobiettivo

In questo capitolo tratteremo nelle sue linee essenziali il problema della ricerca del minimo di più funzioni obiettivo, generalizzazione del caso scalare visto al capitolo 2.
Per i riferimenti teorici e la bibliografia si rimanda a [26] e [31]. In questo capitolo ci limitiamo ad esporre le principali definizioni e i metodi di soluzione più comuni, corredati da qualche semplice esempio.

3.1 Definizioni

Il problema della *ottimizzazione vincolata multiobiettivo,* si può così formulare:

$$\begin{aligned} \min\ & f(x) = (f_1(x), f_2(x), \ldots, f_k(x))^T, \qquad k \geq 2 \\ & x \in \mathcal{F} \end{aligned} \tag{3.1}$$

dove; $f_i : R^n \longrightarrow R,\ \text{per } 1 \leq i \leq k,\ \mathcal{F} \subset R^n$.

Chiameremo R^n *spazio delle variabili di decisione* e R^k *spazio degli obiettivi.* Con $\mathcal{Z} = f(\mathcal{F})$ indicheremo l'immagine della *regione ammissibile* $\mathcal{F} \subset R^n$ delle variabili $x \in R^n$ nello spazio degli obiettivi $f \in R^k$ (si veda la Figura 3.1), cioè:

$$\mathcal{Z} = f(\mathcal{F}) = \{z \in R^k;\ \exists\ x \in \mathcal{F},\ z = f(x)\}.$$

Un vettore degli obiettivi $z \in R^k$ è ammissibile quando $z \in \mathcal{Z} \subset R^k$.

Nota.
Nel seguito (sezione 3.2) considereremo un problema multiobiettivo in cui la regione ammissibile $\mathcal{F}$ è definita dai (soliti) m vincoli di disuguaglianza

$$\begin{aligned} \min\quad & f(x) \\ & g(x) \leq 0 \end{aligned} \tag{3.2}$$

dove: $f : R^n \longrightarrow R^k$ $(k \geq 2)$ e $g : R^n \longrightarrow R^m$ sono funzioni differenziabili con continuità.
Introdotta la lagrangiana L e gli m parametri λ_i

$$L(x, \lambda) = (f_1(x) + f_2(x) + \ldots + f_k(x)) + (\lambda_1 g_1(x) + \lambda_2 g_2(x) + \ldots + \lambda_m f_m(x))$$

al problema (3.2) si potranno estendere le classiche condizioni KKT di ottimalità del primo ordine (2.11)—(2.14).

(Fine Nota).

Si definisce vettore *ideale* z^{id} degli obiettivi, il vettore di componenti

$$z_i^{id} = \min_{x \in \mathcal{F}} f_i(x), \qquad 1 \leq i \leq k \tag{3.3}$$

Se non ci fossero *conflitti* tra le k funzioni obiettivo f_i, una soluzione sarebbe quella ottenibile minimizzando separatamente le k funzioni, ottenendo proprio il vettore z^{id} (soluzione banale).
In realtà, siccome le funzioni obiettivo sono, almeno in parte, in conflitto tra loro, si suppone che $z^{id} \notin \mathcal{Z}$.

Per giungere al concetto di ottimo secondo Pareto (1896)[1], introduciamo nello spazio $R^N, N \geq 2$ la seguente relazione binaria *parziale* $\leq_P$

Definizione 3.1.1 *Dati due vettori $z^1, z^2 \in R^N$, diciamo che z^1 domina z^2 secondo Pareto ($z^1 \leq_P z^2$) se si ha:*

$$\begin{aligned} &z_i^1 \leq z_i^2, && \forall\ i = 1, 2, \ldots, k \\ &z_j^1 < z_j^2 \ \ \textit{per almeno un indice } j \in \{1, \ldots, k\}. \end{aligned}$$

Tramite la relazione d'ordine parziale $\leq_P$ introdotta è possibile dare la seguente definizione di ottimalità secondo Pareto

Definizione 3.1.2 (Ottimo secondo Pareto) *Un vettore di decisioni $x^* \in \mathcal{F}$ è un ottimo di Pareto se non esiste un altro vettore $x \in \mathcal{F}$ tale che $f(x) \leq_P f(x^*)$, ossia*

$$\begin{aligned} &f_i(x) \leq f_i(x^*), && \forall\ i = 1, 2, \ldots, k \\ &f_j(x) < f_j(x^*), \ \ \textit{per almeno un indice } j \in \{1, \ldots, k\}. \end{aligned}$$

[1]Vilfredo Pareto (1848-1923), ingegnere, economista e sociologo (franco-)italiano.

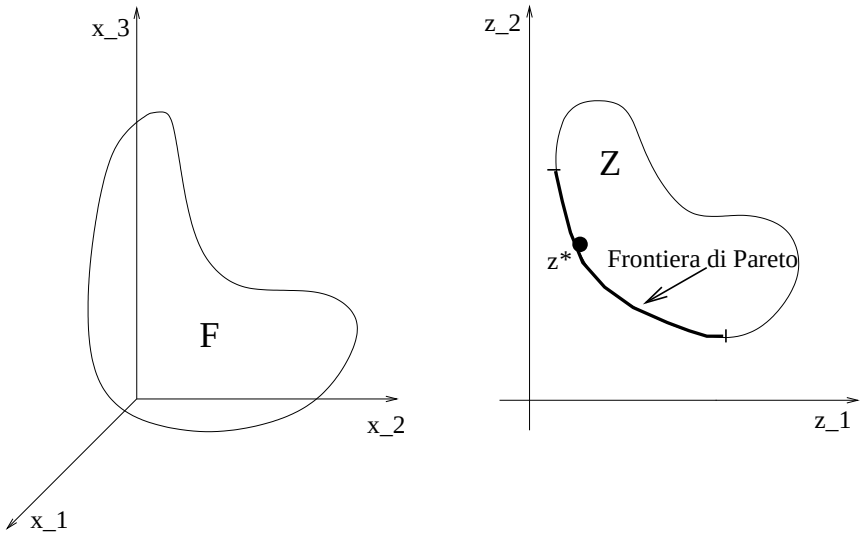


Figura 3.1: Insiemi delle variabili $\mathcal{F}$, degli obiettivi $\mathcal{Z}$ e insieme (frontiera) ottimo di Pareto

Corrispondentemente, *un vettore obiettivo* $z^* \in \mathcal{Z}$ *è ottimo secondo Pareto se non esiste un altro vettore* $z \in \mathcal{Z}$ *tale che* $z \leq_P z^*$, ossia se il vettore di decisioni x^* corrispondente è ottimo secondo Pareto.

In altre parole, *non è possibile* migliorare una componente di z^* senza peggiorarne almeno una delle rimanenti. Si può dire che i vettori ottimi di Pareto sono punti di *equilibrio* che stanno sulla *frontiera* di $\mathcal{Z}$ (si veda la Figura 3.1).

La definizione 3.1.2 è una definizione globale, valida in tutto $\mathcal{F}$. Si ha un ottimo *locale* se essa vale (solo) in un opportuno intorno di $x \in \mathcal{F}$. Ovviamente, un ottimo globale è anche ottimo locale. Il viceversa è vero per problemi di programmazione convessa (insieme $\mathcal{F}$ e funzioni obiettivo convessi)..

Un vettore di decisione x^* si dice *ottimo debole* secondo Pareto se non esiste un altro vettore $x \in \mathcal{F}$ tale che:

$$f_i(x) < f_i(x^*), \quad \forall\ i = 1, 2, \ldots, k.$$

Gli ottimi di Pareto son anche ottimi deboli (non viceversa); si veda la Figura 3.2.

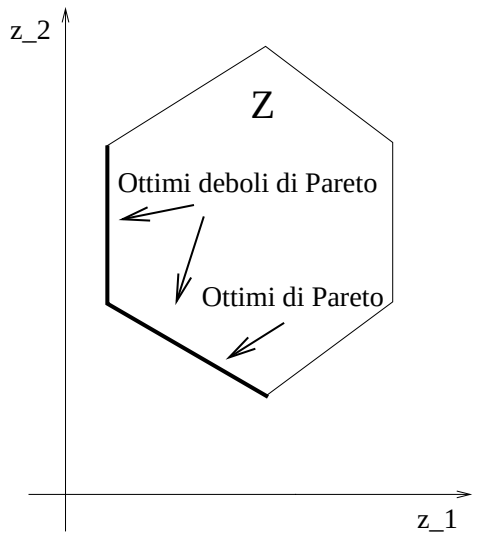


Figura 3.2: Insiemi degli ottimi (forti) di Pareto e degli ottimi deboli di Pareto

3.1.1 Esempio

Risolvere il seguente problema multi-obiettivo:

$$\begin{array}{rl} \min & f(x) = (f_1(x) = x_1 + x_2,\ f_2(x) = x_1 - x_2)^T \\ \text{tale che} & g_1(x) = x_1^2 + x_2^2 - 2 \leq 0 \\ & g_2(x) = x_1^2 - x_2 \leq 0 \\ & g_3(x) = -x_1 \leq 0 \end{array} \tag{3.4}$$

L'insieme delle variabili $\mathcal{F}$ è compreso fra il cerchio di raggio $\sqrt{2}$ e la parabola $x_1^2 - x_2 = 0$ (si veda la Figura 3.3).

Troviamo l'insieme, trasformato di $\mathcal{F}$, degli obiettivi $\mathcal{Z} = f(\mathcal{F})$, applicando la seguente trasformazione $f : \mathcal{F} \longrightarrow \mathcal{Z}$

$$\begin{cases} z_1 = f_1(x_1, x_2) = x_1 + x_2 \\ z_2 = z_2(x_1, x_2) = x_1 - x_2 \end{cases} \quad \Longrightarrow \quad \begin{cases} x_1 = \dfrac{z_1 + z_2}{2} \\ x_2 = \dfrac{z_1 - z_2}{2} \end{cases}$$

L'insieme $\mathcal{Z}$ è perciò così definito (si veda la Figura 3.3):

$$\begin{aligned} &T(g_1) = z_1^2 + z_2 - 4 \leq 0 \\ &T(g_2) = z_1^2 + z_2^2 + 2z_1z_2 - 2z_1 + 2z_2 = (z_1 + z_2)^2 - 2z_1 - 2z_2 \leq 0 \\ &T(g_3) = -(z_1 + z_2) \leq 0 \end{aligned}$$

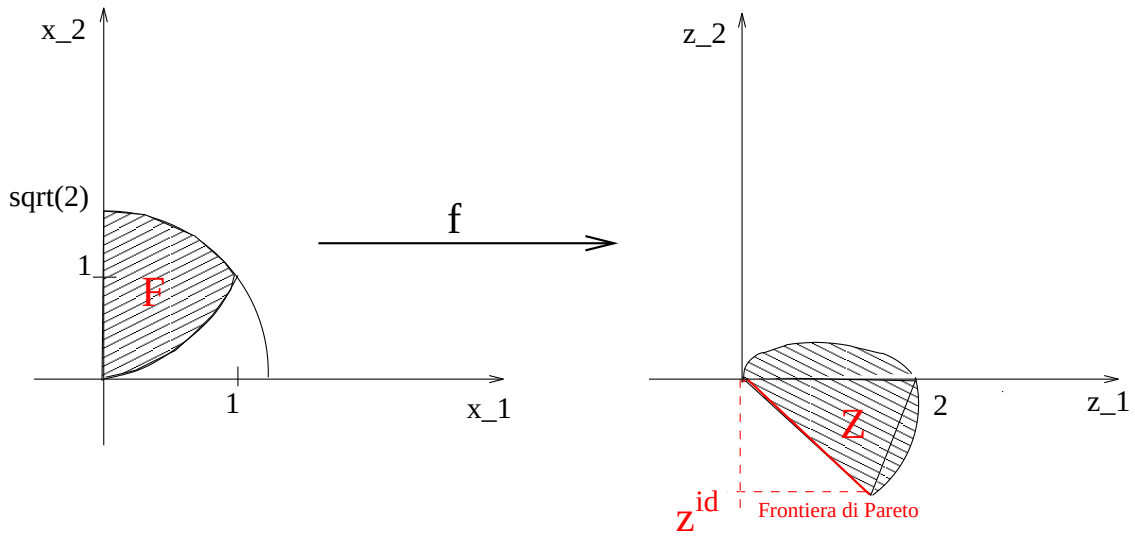


Figura 3.3: Insiemi $\mathcal{F}$ e $\mathcal{Z}$, z^{id} e frontiera di Pareto

Ossia, $\mathcal{Z}$ è compreso fra la parabola $(z_1 + z_2)^2 - 2z_1 - 2z_2 = 0$ di vertice l'origine e di asse $z_1 + z_2 = 0$ e il cerchio di raggio 2.

Possiamo trovare il vettore z^{id} del problema (3.4), risolvendo *separatamente* i due problemi mono-obiettivo:

$$\begin{array}{lll} \min & z_1 = x_1 + x_2 \\ g_1(x) & = x_1^2 + x_2^2 \leq 2 \\ g_2(x) & = x_1^2 - x_2^2 \leq 0 \\ g_4(x) & = -x_1 \leq 0 \end{array} \quad \text{e} \quad \begin{array}{lll} \min & z_2 = x_1 - x_2 \\ g_1(x) & = x_1^2 + x_2^2 \leq 2 \\ g_2(x) & = x_1^2 - x_2^2 \leq 0 \\ g_4(x) & = -x_1 \leq 0 \end{array}$$

È facile trovare i due minimi x^{1*}, x^{2*} per via grafica, tracciando le curve di livello $z_1 = c$, $z_2 = c$ (si veda la Figura 3.4). Si trova:

$$x^{1*} = (0,0)^T \implies z_1(x^{1*}) = 0, \qquad x^{2*} = (0,\sqrt{2})^T \implies z_2(x^{2*}) = -\sqrt{2}.$$

Perciò: $z^{id} = (0, -\sqrt{2})^T$. Si noti che $z^{id} \notin \mathcal{Z} = f(\mathcal{F})$, a conferma del fatto che il problema non si risolve minimizzando separatamente le due funzioni obiettivo. In Figura 3.3 è anche evidenziato l'insieme (frontiera) ottimo di Pareto. Essendo il problema convesso, i minimi sono (locali e anche) globali.

3.2 Metodi di Soluzione

Osservazione.
Tutti i metodi di soluzione si basano sulla medesima idea: trasformare il pro-

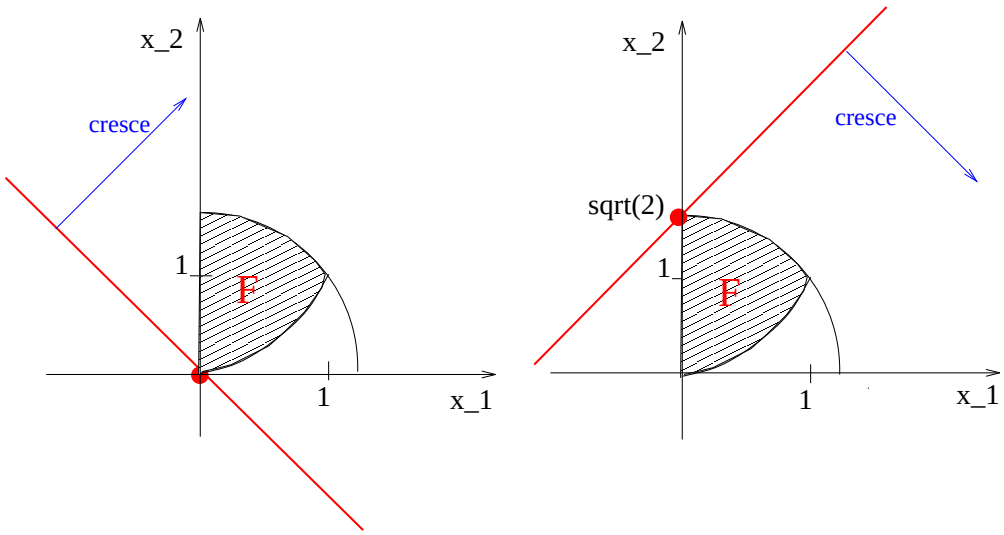


Figura 3.4: Problemi-monobiettivo

blema originario in uno con *una sola* funzione obiettivo ('scalarizzazione' del problema vettoriale). Si applicano poi i metodi classici della programmazione mono-obiettivo visti al capitolo 2.

In base alle scelte del decisore, i metodi vengono suddivisi in varie grandi categorie, sulle quali non insistiamo, quali ad esempio[2]:
metodi senza preferenze, metodi a posteriori, metodi a priori, ecc...

3.2.1 Metodo della Distanza (senza preferenze)

Detto anche *metodo Global o Goal* [3]
Strategia: si cerca la soluzione che minimizza, nello spazio degli obiettivi, la distanza tra la regione ($\mathcal{Z}$) ammissibile e un qualunque punto di riferimento $z^{ref} \notin \mathcal{Z} = f(\mathcal{F})$. Ad esempio $z^{ref} = z^{id}$. In tal caso, si risolve dunque il problema:

$$\min \; d(x) = \|f(x) - z^{id}\|_p \qquad \mathcal{F} = \{x \in R^n;\; g(x) \leq 0\} \tag{3.5}$$

dove

$$\|x\|_p = \left(\sum_{i=1}^{k} |x_i|^p \right)^{1/p}, \qquad 1 \leq p \leq \infty$$

[2]Cfr. [31, Part II].
[3]Per il metodo Goal propriamente detto, cfr. la funzione *fgoalattain* di [11].

è la classica norma p di un vettore.
Per $p = \infty$ si ha un classico problema di $\min - \max$ (di Tchebycheff).

Esempio 1 Vedi [26, 5.1.1], pag. 14-15.

Varianti del metodo Global utilizzano funzioni equivalenti alla (3.5) da minimizzare, per esempio le due funzioni seguenti (per $p > 1$):

$$d_1(x) = \left(\|f(x) - z^{id}\|_p\right)^p = \sum_{i=1}^{k} \left(f_i(x) - z_i^{id}\right)^p, \quad d_2(x) = \sum_{i=1}^{k} \left(\frac{f_i(x) - z_i^{id}}{z_i^{id}}\right)^p.$$

Esempio 2: (uso della funzione $d_2(x)$)
Ricerca del $\min f(x)$, per un problema di *production planning* (Azarm, 1996)

$$\begin{aligned} \min \quad & f(x) = (f_1(x) = -4x_1 - 5x_2,\ f_2(x) = -x_1)^T \\ \text{tale che} \quad & g_1(x) = x_1 + x_2 - 200 \leq 0 \\ & g_2(x) = (5/4)x_1 + (3/4)x_2 - 200 \leq 0 \\ & g_3(x) = x_2 - 150 \leq 0 \\ & g_4(x) = -x_1 \leq 0 \\ & g_5(x) = -x_2 \leq 0 \end{aligned}$$

Applichiamo la seguente trasformazione $T:\ \mathcal{F} \longrightarrow \mathcal{Z}$, dove, per un più facile confronto, i due assi coordinati sono in realtà $-f_1$, $-f_2$:

$$\begin{cases} z_1 = f_1(x_1, x_2) = 4x_1 + 5x_2 \\ z_2 = f_2(x_1, x_2) = x_1 \end{cases} \quad \Longrightarrow \quad \begin{cases} x_1 = -f_2 \\ x_2 = \dfrac{-f_1 + 4f_2}{5} \end{cases}$$

L'insieme trasformato degli obiettivi $\mathcal{Z}$ è perciò così definito:

$$\begin{aligned} T(g_1) &= f_2 + f_1 \leq 1000 \\ T(g_2) &= 3f_2 + f_1 \leq 4000 \\ T(g_3) &= f_1 - 4f_2 \leq 750 \\ T(g_4) &= -f_2 \leq 0 \\ T(g_5) &= -f_1 + 4f_2 \leq 0 \end{aligned}$$

Nelle due Figure 3.5 e 3.6 sono disegnati, rispettivamente, l'insieme delle variabili di decisione $\mathcal{F} \subset R^2$ e il suo trasformato $\mathcal{Z} \subset R^2$, insieme delle funzioni obiettivo. Come avvertito, in Figura 3.6 i due assi coordinati sono $-f_1$, $-f_2$ e la vera figura risulta così ribaltata.

Figura 3.5: Insieme $\mathcal{F} = S$ delle variabili di decisione (x_1, x_2) del problema production planning (linea azzurra); (Esempio 2).

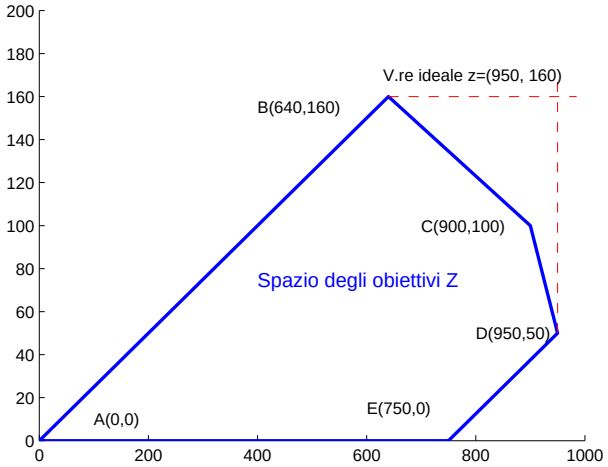


Figura 3.6: Insieme $\mathcal{Z}$ degli obiettivi (z_1, z_2) del problema production planning (linea azzurra); (Esempio 2).

Se si riguarda il problema come un problema di massimo: $\max f = -\min(-f)$, dalla Figura 3.6 risulta evidente[4] che (si notino i segmenti in rosso)

$$\max\ -f_1(x) = \max\ 4x_1 + 5x_2 = 950, \quad \text{e} \quad \max\ -f_2(x) = \max\ x_1 = 160$$

e quindi $z^{id} = (950,\ 160)^T$. Infatti:

$f_1^{id} = \max -f_1 = 950$ in $x_1^* = (50,\ 150)^T$ e $f_2^{id} = \max -f_2 = 160$ in $x_1^* = (160,\ 0)^T$.

Scegliendo $p = 2$, si deve quindi risolvere il problema scalare di programmazione quadratica

$$\min\ d_2(x) = \left(\frac{4x_1 + 5x_2 - 950}{950}\right)^2 + \left(\frac{x_1 - 160}{160}\right)^2$$

sotto i vincoli $g(x) \leq 0$. Applicando un qualsiasi metodo di ottimizzazione mono-obiettivo[5], si trova

$$x^* = (135.1\ ,\ 41.4)^T \quad \Longrightarrow \quad f(x^*) = (f_1, f_2)^T = (747.4\ ,\ 135.1)^T\ .$$

3.2.2 Metodo dei Pesi (a posteriori)

Strategia: si risolve il seguente problema scalare

$$\begin{aligned} \min\ & \sum_{i=1}^{k} \omega_i\ f_i(x) \\ & g(x) \leq 0 \end{aligned}$$

dove $\omega = (\omega_1, \omega_2, \ldots, \omega_k)^T \in R_+^k\ (\geq 0)$ è il vettore dei *pesi*, che si intendono normalizzati: $\sum_{i=1}^{k} \omega_i = 1$. Essi sono scelti dal decisore.

Se il problema è convesso, modificando l'insieme dei nodi si può ottenere l'intero insieme ottimo di Pareto.

Esempio 1: tornio (lathe) (Azarm, 1996)

$$\begin{aligned} \min \quad & f_1(x) \quad = 3x^2 \\ \min \quad & f_2(x) \quad = (5 - x)^2 \\ & (x_1,\ x_2) \geq 0 \end{aligned}$$

[4] Si controlli anche la Figura 3.7, riferita allo stesso problema.

[5] Per esempio, la funzione *fmincon* di [11].

dove f_1 è la misura dell'inverso della percentuale di rimozione del metallo (metal removal rate), f_2 l'inverso del periodo di vita del tornio (tool life) e la variabile di decisione x è la percentuale di alimentazione (feed rate).

Il problema (scalare) da risolvere è:

$$\begin{aligned} \min\ f(x) = \omega_1 f_1 + \omega_2 f_2 = \omega_1(3x^2) + \omega_2(5-x)^2 \\ \omega_1 + \omega_2 = 1 \\ (\omega_1,\ \omega_2) \geq 0 \end{aligned}$$

Derivando, si ottiene subito la soluzione analitica:

$$f'(x) = 6\ \omega_1 x - 10\omega_2 + \omega_2 x = 0, \quad x^* = \frac{5\omega_2}{3\ \omega_1 + \omega_2}$$

Si verifica subito che si tratta di un minimo, in quanto:

$$f''(x) = 6\omega_1 + 2\omega_2 > 0, \qquad \forall x.$$

Nella seguente Tabella è riportata la soluzione x^* secondo la scelta dei pesi corrispondente:

ω_1	0.2	0.4	0.6	0.8
ω_2	0.8	0.6	0.4	0.2
x^*	2.86	1.67	0.91	0.38

3.2.3 Metodo degli ϵ-vincoli (a posteriori)

Detto anche *metodo Trade-off.*

Strategia: si seleziona una funzione obiettivo $f_l(x)$ e poi si trasformano tutte le altre funzioni obiettivo $f_i(x)$ (con $i = 1, 2, \ldots, k,\ i \neq l$) in vincoli, imponendo degli *upper bound* ϵ_i sui loro valori.

Il problema scalare da risolvere è allora il seguente:

$$\begin{aligned} \min \quad & f_l(x) && l \in \{1, 2, \ldots, k\} \\ & f_i(x) \leq \epsilon_i, && \forall i = 1, 2, \ldots, k,\ i \neq l \\ & g(x) \leq 0 \end{aligned}$$

Si può dimostrare che, se il punto x^* è l'unica soluzione del problema per qualche $l \in \{1, 2, \ldots, k\}$ e con $\epsilon_i = f_i(x^*)$ per ogni $i \neq l$, allora esso è ottimo di Pareto per il problema originale (3.2).

Esempio 1: tornio (lathe) (Azarm, 1996)
Prese le due variabili di decisione x_1 = velocità di taglio (cutting speed),

e x_2 = percentuale di alimentazione per giro (feed rate per revolution), si chiede di massimizzare:

1. la percentuale di rimozione del metallo (metal remove rate) $f_1(x)$
2. la vita dell'utensile (tool life) $f_2(x)$.

Tenuto conto dei vincoli fisici (...omissis...), il problema (di minimo) da risolvere è

$$\begin{aligned}
\min \quad & -f(x) = \left(-f_1(x) = -\ 6000\ x_1 x_2,\ -f_2(x) = -\ \frac{1.28 \times 10^7}{x_1^{3.33}\ x_2^{2.22}} \right)^T \\
\text{tale che} \quad & g_1(x) = -x_1 + 21.1 \leq 0 \\
& g_2(x) = x_1 - 263.9 \leq 0 \\
& g_3(x) = -x_2 + 0.05 \leq 0 \\
& g_4(x) = x_2 - 1.0 \leq 0 \\
& g_5(x) = x_1 x_2^{0.75} - 95.7 \leq 0 \\
& g_6(x) = -x_1^2 x_2 + 1600 \leq 0
\end{aligned}$$

Scegliendo $-f_1$ come funzione da minimizzare, il problema scalare da risolvere è il seguente

$$\begin{aligned}
\min & -f_1(x) = -6000\ x_1 x_2 \\
\text{tale che} \quad & g_i(x) \leq 0 \qquad 1 \leq i \leq 6 \\
& g_7(x) = -f_2(x) = \quad -\ \frac{1.28 \times 10^7}{x_1^{3.33}\ x_2^{2.22}} \leq -\ \epsilon_2
\end{aligned}$$

Nella seguente Tabella è riportata la soluzione $x^* = (x_1^*, x_2^*)^T$ per differenti valori del 'upper bound' $-\epsilon_2$ della funzione obiettivo f_2:

ϵ_2	$(x_1^*,\ x_2^*)^T$	$-f_1(x_1^*,\ x_2^*)$	$-f_2(x_1^*,\ x_2^*)$
45	$[43.3,\ 1.0]^T$	-2.60×10^5	-45.12
60	$[40.0,\ 1.0]^T$	-2.40×10^5	-59.20
75	$[48.9,\ 0.67]^T$	-1.96×10^5	-73.77
90	$[57.6,\ 0.48]^T$	-1.65×10^5	-89.67

3.2.4 Metodo Lessicografico (a priori)

Detto anche *metodo Gerarchico.*

Strategia: il decisore ordina le funzioni obiettivo in base alla loro importanza relativa. Una volta che le funzioni obiettivo siano state ordinate, il processo

risolutivo inizia con la minimizzazione della prima funzione obiettivo $f_1(x)$ sull'insieme ammissibile originario $\mathcal{F}$. Si risolve cioè il problema scalare

$$\begin{aligned} \min \quad & f_1(x) \\ & g(x) \leq 0. \end{aligned} \tag{3.6}$$

Se il problema (3.6) ha un'unica soluzione allora questa è anche soluzione di (3.2) e l'algoritmo termina. Altrimenti si minimizza la seconda funzione obiettivo $f_2(x)$ nell'ordinamento lessicografico scelto, ma *aggiungendo* ai vincoli originari un ulteriore vincolo il cui scopo è quello di garantire che all'ottimo non peggiori il valore della prima funzione obiettivo. Il problema è quindi il seguente

$$\begin{aligned} \min \quad & f_2(x) \\ & f_1(x) \leq \ f_1(x_1^*) \\ & g(x) \leq 0 \end{aligned} \tag{3.7}$$

dove x_1^* è la soluzione di (3.6). Se (3.7) ha un'unica soluzione ci si ferma, altrimenti si procede come prima, selezionando la successiva funzione obiettivo nell'ordinamento scelto.
Al generico passo $h \leq k$ avremo il seguente problema

$$\begin{aligned} \min \quad & f_h(x) \\ & f_i(x) \leq \ f_i(x_i^*) \qquad i = 1, 2, \ldots, h-1 \\ & g(x) \leq 0 \end{aligned} \tag{3.8}$$

dove x_i^* è la soluzione del problema $i-$esimo.

Si può dimostrare che ogni soluzione ottenuta con il metodo lessicografico è ottima secondo Pareto per il problema (3.2).

Nota.
Gli upper bound dei vincoli aggiuntivi possono essere una frazione del valori ottimi $f_i(x_i^*)$ via via trovati, cioè:

$$f_i(x) \leq \left(1 - \frac{\varepsilon_i}{100}\right) f_i(x_i^*), \qquad 0 \leq \varepsilon_i \leq 100.$$

Esempio 1
Riprendiamo il problema della ricerca del $\max f = -\min(-f)$ per il proble-

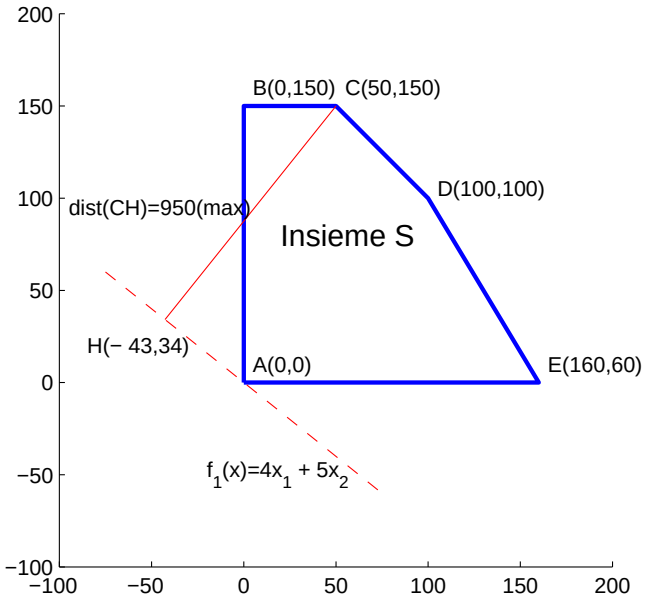


Figura 3.7: Metodo gerarchico: max di f_1 nell'insieme $\mathcal{F} = S$

ma di *production planning*, esempio 2 di sezione 3.2.1:

$$\begin{aligned} \max \quad & f(x) = (f_1(x) = 4x_1 + 5x_2,\ f_2(x) = x_1)^T \\ \text{tale che} \quad & g_1(x) = x_1 + x_2 - 200 \leq 0 \\ & g_2(x) = (5/4)x_1 + (3/4)x_2 - 200 \leq 0 \\ & g_3(x) = x_2 - 150 \leq 0 \\ & g_4(x) = -x_1 \leq 0 \\ & g_5(x) = -x_2 \leq 0 \end{aligned}$$

Ritenendo la funzione f_1 la prima in ordine di importanza, risolviamo il problema, ignorando la funzione f_2. Dalla Figura 3.7 risulta evidente, come già è stato detto, che la soluzione è

$$x^* = (50\ ,\ 150)^T \quad \Longrightarrow \quad f_1(x^*) = 950\ .$$

Il passo successivo del metodo è quello di trovare il massimo della $f_2 = x_1$ nell'insieme $F1 = (A, B, C1, D1, E)$ formato dai vincoli originali e da un ulteriore vincolo $g_6(x)$ fornito dal valore ottimo della funzione f_1 appena trovato (si veda la Figura 3.8):

$$g_6(x) = 4x_1 + 5x_2 - 950 \leq 0.$$

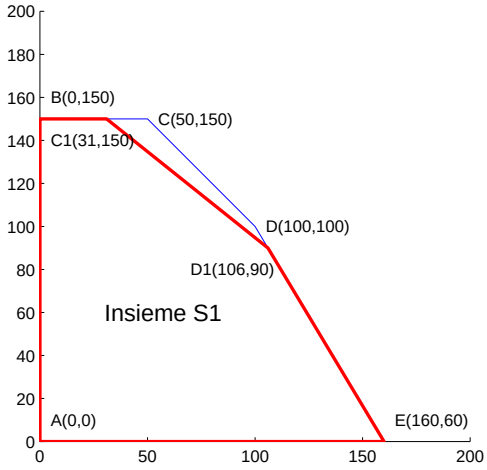


Figura 3.8: Metodo gerarchico: max di f_2 nell'insieme $\mathcal{F}1 = S1$

Si trova (è anche geometricamente evidente):

$$x^* = (160\ , 0)^T \quad \Longrightarrow \quad f_1(x^*) = 640,\ f_2(x^*) = 160\ .$$

Appendice A

Sistemi lineari sparsi e di grandi dimensioni

A.1 Introduzione

Data una matrice quadrata non singolare A $n \times n$ diciamo che essa è **sparsa** se il numero di elementi diversi da zero (nonzeri) si può esprimere come $k \cdot n$ dove k è una costante, generalmente molto piccola (ordine delle unità o delle decine). Matrici sparse sono comuni in molti problemi della matematica applicata quali ad esempio la discretizzazione di equazioni differenziali alle derivate parziali. Siamo interessati a risolvere un sistema lineare della forma $Ax = b$, dove A è una matrice sparsa possibilmente di grandi dimensioni. Se A è sparsa, si può memorizzare su un calcolatore in forma **compatta,** ovvero memorizzando solo gli elementi diversi da zero.

Esempio. Una matrice di dimensione $n = 10\,000$ in cui il numero dei nonzeri sia 50 000, può essere memorizzata tramite tre vettori (2 interi e uno reale) di dimensione 50 000 anzichè con una matrice reale di dimensione $10\,000^2 = 10^8$.

Tale memorizzazione si chiama formato a **coordinate**: per ciascun elemento diverso da zero, si memorizzano *gli indici di riga e di colonna e il valore dell'elemento.*

La moltiplicazione di una tale matrice per un vettore ha un costo computazionale dell'ordine del numero dei nonzeri della matrice (non più dell'ordine di n^2).

A.2 Metodi diretti (Cenni)

I metodi diretti per la soluzione di sistemi lineari hanno in comune la caratteristica di produrre una fattorizzazione della matrice che si scrive come prodotto di due matrici triangolari: $A = LU$. Una volta effettuata la fattorizzazione (cfr. [45, cap. 3.2]), la soluzione del sistema originale si ottiene risolvendo i due sistemi triangolari

$$Ly = b, \qquad Ux = y$$

Se A è sparsa, non c'è garanzia che i due fattori lo siano. Nella pratica succede che le matrici L ed U sono **molto meno** sparse della matrice A. Questo fatto produce:

- Occupazione di memoria molto superiore al sistema originale.
- Soluzione dei sistemi lineari triangolari molto costosa dal punto di vista computazionale.

Per questo motivo, la formulazione classica dei metodi diretti (per esempio della eliminazione di Gauss) è stata modificata allo scopo di ridurre:

- Il tempo di calcolo della fattorizzazione.
- La "densità" dei fattori triangolari prodotti dalla fattorizzazione e quindi il tempo impiegato per la soluzione dei sistemi triangolari.

Sono stati sviluppati i metodi **frontali** che si basano su un riordinamento preliminare della matrice A allo scopo di minimizzare il *riempimento* dei fattori triangolari. I metodi frontali, inoltre, sfruttano in maniera ottimale gli accessi in memoria riformulando la fattorizzazione di una matrice sparsa in tante fattorizzazioni di matrici dense ma di piccole dimensioni. Si noti che la semplice istruzione MatLab che permette di risolvere un sistema lineare: `x = A \ b`, implementa implicitamente il metodo frontale.

L'implementazione, pur ottimale di tali metodi, non sempre produce un algoritmo efficiente per la soluzione di sistemi lineari sparsi. In particolare, se le dimensioni sono molto grandi o se la larghezza di banda della matrice, ovvero $\max\{|i-j| : a_{ij} \neq 0\}$ è grande, i fattori triangolari rimangono troppo densi per poter essere memorizzati e risolti in maniera efficiente.

Una valida alternativa ai metodi diretti è rappresentata in questo caso dai metodi iterativi. Nel corso di Calcolo Numerico si sono studiati studiati alcuni metodi (stazionari): Jacobi, Gauss-Seidel, SOR, (cfr. [45, cap. 3.3]. I metodi iterativi (variazionali) che vedremo in Appendice B sono generalmente più rapidi e robusti dei metodi precedenti, almeno per una classe importante di matrici: le matrici simmetriche definite positive.

Appendice B

Metodi variazionali per la soluzione di sistemi lineari

B.1 Introduzione al problema

Definizione. Si dice che una matrice quadrata A $n \times n$ è *definita positiva* se per ogni $x \in \mathbb{R}^n, x \neq 0$ vale

$$x^T Ax > 0$$

È noto che tutti gli autovalori di una matrice simmetrica e definita positiva sono positivi. Nel seguito denoteremo gli autovalori di A con $0 < \lambda_n < \cdots < \lambda_1$.
Sia da risolvere il sistema lineare

$$Ax = b \tag{B.1}$$

dove A è una matrice quadrata simmetrica e definita positiva, b è il vettore termine noto, e x è il vettore soluzione.

B.2 Metodo della discesa più ripida (del gradiente)

Sia h la soluzione "esatta" (unica!) del sistema (B.1), ed $e = x - h$ il vettore errore. Definiamo inoltre il vettore *residuo* $r = b - Ax$.
Definiamo una funzione $\mathbb{R}^n \to \mathbb{R}$

$$\phi(x) = \frac{1}{2}(x-h)^T A(x-h) = \frac{1}{2}e^T Ae$$

Essendo la fuzione ϕ positiva per qualunque $e \neq 0$ (e dunque $x \neq h$), e poiché vale $\phi(h) = 0$, risulta che l'unico minimo della funzione ϕ si ha per $x = h$. Risolvere il sistema (B.1) equivale dunque a *minimizzare* la funzione $\phi(x)$. Assegnata una soluzione approssimata $x \neq h$, $\phi(x)$ assumerà un valore positivo (incognito perché non si conosce h). Se teniamo fisso questo valore, l'equazione $\phi(x) =$ costante rappresenta una forma quadratica in n dimensioni, cioè un *iperellissoide* nello spazio n-dimensionale il cui centro geometrico coincide con la soluzione h.
Sia ora

$$A = UDU^T$$

dove $D = \text{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$ è la matrice diagonale degli autovalori di A e U è la matrice le cui colonne sono i corrisponenti autovettori. Tale diagonalizzazione è possibile grazie al fatto che A è simmetrica e dunque *normale*. Scriviamo ora la funzione $\phi(x)$ effettuando un cambio di variabile

$$x' = U^T x, \qquad e' = U^T e$$

Si ha

$$\phi(x) = \frac{1}{2}(e')^T U^T A U e' = \frac{1}{2}(e')^T D e' = \frac{1}{2}\sum_{i=1}^{n}(e'_i)^2\lambda_i$$

L'equazione dell'iperellissoide con il cambio di variabile diventa

$$\frac{1}{2}(e')^T D e' = \text{costante} \tag{B.2}$$

Ora i semi-assi dell'iperellissoide coincidono con gli assi cartesiani.
Trovare la soluzione del sistema coincide con il minimizzare la funzione ϕ e dunque coincide con il trovare il centro dell'iperellissoide dato un punto iniziale di partenza. Nello schema classico chiamato della *discesa più ripida* o *steepest descent* l'approssimazione $(k+1)$-esima[1] della soluzione è ottenuta dalla precedente con l'equazione

$$x_{k+1} = x_k + \alpha_k p_k$$

dove α_k è un opportuno scalare e p_k è la direzione *normale* all'iperellissoide nel punto $x = x_k$. La normale alla superficie, diretta verso il centro dell'iperellissoide (B.2), è il vettore *gradiente* (cambiato di segno) del campo scalare $\phi =$ costante. Nel nuovo sistema di riferimento è semplice calcolare il gradiente:

$$p'_k = -De'_k$$

[1] In questa Appendice, la notazione x_k e simili equivale ovviamente alla notazione $x^{(k)}$.

Ora dobbiamo ricavare p_k ritornando al vecchio sistema di riferimento

$$p_k = Up'_k = -UDe'_k = -UDU^T e_k = -Ae_k = -A(x_k - h) = b - Ax_k = r_k$$

cioè la direzione del gradiente coincide con il vettore residuo r_k.
Si tratta ora di determinare lo scalare α_k. Per questo richiediamo che $\phi(x_{k+1}) = \phi(x_k + tp_k)$ sia minima al variare di $t \in \mathbb{R}$. Consideriamo ora la ϕ come funzione del parametro t e imponiamo l'annullamento della sua derivata prima.

$$\begin{aligned}\frac{\partial \phi}{\partial t} &= \frac{1}{2}\frac{\partial\left[(h - x_k - tp_k)^T A\ (h - x_k - tp_k)\right]}{\partial t} = \\ &= \frac{1}{2}\left(-p_k^T A(h - x_k - tp_k) - (h - x_k - tp_k)^T Ap_k\right) \\ &= -p_k^T Ah + p_k^T Ax_k + tp_k^T Ap_k \\ &= -p_k^T b + p_k^T Ax_k + tp_k^T Ap_k = -p_k^T r_k + tp_k^T Ap_k\end{aligned}$$

La condizione $\dfrac{\partial \phi}{\partial t} = 0$ equivale a

$$\alpha_k = t = \frac{p_k^T r_k}{p_k^T Ap_k} = \frac{r_k^T r_k}{r_k^T Ar_k} \tag{B.3}$$

La generica iterazione $k+1$ dell'algoritmo della discesa ripida si può schematizzare come segue:

$$\begin{aligned} r_k &= b - Ax_k \\ \alpha_k &= \frac{r_k^T r_k}{r_k^T Ar_k} \\ x_{k+1} &= x_k + \alpha_k r_k \end{aligned}$$

Si noti che è possibile ricavare il residuo al passo $k+1$ in funzione del residuo precedente, risparmiando un prodotto matrice vettore:
da $x_{k+1} = x_k + \alpha_k r_k$ si ottiene $b - Ax_{k+1} = b - Ax_k - A\alpha_k r_k$ cioè

$$r_{k+1} = r_k - \alpha_k Ar_k\ .$$

Il valore di α_k, scelto in modo da minimizzare localmente la ϕ, garantisce l'**ortogonalità** di due residui consecutivi, ovvero (cfr. anche la (1.8)) :

$$r_{k+1}^T r_k = 0\ . \tag{B.4}$$

Il metodo iterativo termina quando è verificata la condizione (sul residuo relativo)

$$\frac{\|r_k\|}{\|b\|} < \text{toll}\ .$$

ALGORITMO STEEPEST DESCENT
Input: $x_0, A, b, k_{\max}$, toll

- $r_0 = b - Ax_0,\ k = 0$
- WHILE $\|r_k\| > \text{toll}\ \|b\|$ AND $k < k_{\max}$ DO
  1. $z_k = Ar_k$
  2. $\alpha_k = \dfrac{r_k^T r_k}{z_k^T r_k}$
  3. $x_{k+1} = x_k + \alpha_k r_k$
  4. $r_{k+1} = r_k - \alpha_k z_k$
  5. $k = k + 1$
- END WHILE

Convergenza

Si definisca (cfr. [45, sez. 3.2.8]) il numero di condizionamento spettrale di A $\kappa_2(A)$ come

$$\kappa_2(A) = \frac{\lambda_1}{\lambda_n}. \qquad \text{Inoltre:} \quad \mu = \frac{\kappa_2(A) - 1}{\kappa_2(A) + 1}$$

Si può dimostrare che (cfr. la (1.10), dove $M = \mu$)

$$\sqrt{\phi(x_k)} = \sqrt{e_k^T A e_k} = \|e_k\|_{A,2} \leq \mu^k \|e_0\|_{A,2} \tag{B.5}$$

Nei casi realistici il numero $\kappa_2(A)$ è molto grande e dunque il fattore di riduzione dell'errore (costante asintotica dell'errore) μ molto vicino ad 1. In questi casi si può dare una stima (pessimistica) del numero di iterazioni necessario per ridurre la norma A dell'errore di una tolleranza ε.

$$k \approx \frac{1}{2} \log \left(\frac{2}{\varepsilon} \right) (\kappa_2(A) + 1)$$

L'espressione appena scritta indica che il numero di iterazioni è proporzionale a $\kappa_2(A)$.

B.3 Il metodo del Gradiente Coniugato (CG)

Cerchiamo ora una relazione di ricorrenza del tipo

$$x_{k+1} = x_k + \alpha_k p_k$$

Rispetto al caso della discesa più ripida cambia la scelta dei vettori p_k che sono scelti in modo da verificare la ricorrenza

$$\begin{aligned} p_0 &= r_0 \\ p_{k+1} &= r_{k+1} + \beta_k p_k, \qquad k \geq 0 \end{aligned}$$

I coefficienti β_k sono scelti in modo da rendere l'insieme dei vettori p_k A-ortogonale cioè tale da soddisfare

$$p_{k+1}^T A p_k = 0$$

Sostituendo si ha

$$p_{k+1}^T A p_k = r_{k+1}^T A p_k + \beta_k p_k^T A p_k$$

da cui

$$\beta_k = -\frac{r_{k+1}^T A p_k}{p_k^T A p_k}$$

Come nel caso della SD – formula (B.3) – richiediamo che α_k sia tale da minimizzare la funzione $\phi(x_k + \alpha_k p_k)$

$$\alpha_k = \frac{r_k^T p_k}{p_k^T A p_k}$$

B.3.1 Proprietà del Gradiente Coniugato

1. $p_k^T r_k = r_k^T r_k \qquad (\implies \quad \alpha_k = \dfrac{r_k^T p_k}{p_k^T A p_k} = \dfrac{r_k^T r_k}{p_k^T A p_k})$

   Infatti

   $$\begin{aligned} p_{k-1}^T r_k &= p_{k-1}^T \left(r_{k-1} - \alpha_{k-1} A p_{k-1}\right) \\ &= p_{k-1}^T r_{k-1} - \alpha_{k-1} p_{k-1}^T A p_{k-1} = 0 \end{aligned} \tag{B.6}$$

   dove l'ultima uguaglianza si deve alla definizione di α_{k-1}.

   $$\begin{aligned} p_k^T r_k &= \left(r_k + \beta_{k-1} p_{k-1}\right)^T r_k = \\ &= r_k^T r_k + \beta_{k-1} p_{k-1}^T r_k = (\text{per la proprietà (B.6)}) = r_k^T r_k \end{aligned} \tag{B.7}$$

2. Dalla proprietà (B.6) segue anche che la direzione p_k è una direzione di discesa (cfr. la (1.6)):
$$p_k^T \nabla f_k = -(r_k + \beta_{k-1} p_{k-1})^T r_k = -r_k^T r_k - \beta_{k-1} p_{k-1}^T r_k < 0 \ .$$

3. $r_k^T A p_k = p_k^T A p_k$
Segue dalla relazione $r_k = p_k - \beta_{k-1} p_{k-1}$ e dalla A-ortogonalità di p_k e p_{k-1}.

4. Combinando le proprietà 1. e 3. si ottiene
$$\alpha_k = \frac{r_k^T p_k}{p_k^T A p_k} = \frac{r_k^T r_k}{r_k^T A p_k}$$
Da quest'ultima definizione per α_k si deriva inoltre l'ortogonalità di due residui consecutivi.
$$r_k^T r_{k+1} = r_k^T (r_k - \alpha_k A p_k) = r_k^T r_k - \alpha_k r_k^T A p_k = 0$$

5. L'insieme $\{p_k\}$ è A-ortogonale ovvero
$$p_{k+1}^T A p_i = 0, \qquad \text{per} \quad 0 \le i \le k$$
Da tale proprietà segue che p_n, dovendo essere A-ortogonale ad un insieme linearmente indipendente di n vettori, è necessariamente il vettore nullo. Dunque
$$x_{n+1} = x_n + p_n = x_n$$
pertanto x_n è soluzione del sistema lineare (B.1).

6. L'insieme $\{r_k\}$ è ortogonale (I-ortogonale) ovvero
$$r_{k+1}^T r_i = 0, \qquad \text{per} \quad 0 \le i \le k$$
Come per la proprietà 5. necessariamente deve aversi $r_n = 0$

7. Formulazioni alternative per il coefficiente β_k
$$p_k^T A p_k = p_k^T \frac{1}{\alpha_k}(r_k - r_{k+1}) = (\text{proprietà 1. e 2.}) = \frac{1}{\alpha_k} r_k^T r_k$$
$$r_{k+1}^T A p_k = r_{k+1}^T \frac{1}{\alpha_k}(r_k - r_{k+1}) = -\frac{1}{\alpha_k} r_{k+1}^T r_{k+1}$$
Dunque si può scrivere
$$\beta_k = -\frac{r_{k+1}^T A p_k}{p_k^T A p_k} = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$$

ALGORITMO DEL GRADIENTE CONIUGATO

In Input: $x_0, A, b, k_{\max}$, toll

- $r_0 = p_0 = b - Ax_0,\ k = 0$
- WHILE $\|r_k\| > \text{toll}\ \|b\|$ AND $k < k_{\max}$ DO
  1. $z_k = Ap_k$
  2. $\alpha_k = \dfrac{p_k^T r_k}{z_k^T p_k}$
  3. $x_{k+1} = x_k + \alpha_k p_k$
  4. $r_{k+1} = r_k - \alpha_k z_k$
  5. $\beta_k = \dfrac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$
  6. $p_{k+1} = r_{k+1} + \beta_k p_k$
  7. $k = k + 1$
- END WHILE

Convergenza

Le proprietà 4. e 5. prima enunciate mostrano che, in aritmetica infinita, il metodo del gradiente coniugato converge alla soluzione esatta in al più n passi.

Quando si implementa il metodo C.G. su un calcolatore e lo si usa per risolvere il sistema (B.1) per n sufficientemente grande, si osserva che non sono sufficienti n iterazioni per ottenere la soluzione. Ciò è dovuto ad errori di arrotondamento che distruggono a poco a poco l'ortogonalità delle successioni $\{p_k\}$ e $\{r_k\}$. D'altra parte, dovendo risolvere sistemi lineari di grandi dimensioni ($n \approx 10^5$) anche un numero n di iterazioni sarebbe eccessivo e renderebbe poco competitivo il metodo iterativo del CG a confronto con i metodi diretti.

Si può dimostrare che anche il metodo del CG converge linearmente, ma la costante asintotica dell'errore è più piccola rispetto alla SD. Vale infatti che:

$$\sqrt{\phi(x_k)} = \sqrt{e_k^T A e_k} = \|e_k\|_{A,2} \leq 2(\mu')^k \|e_0\|_{A,2} \tag{B.8}$$

dove la costante asintotica dell'errore μ' vale:

$$\mu' = \frac{\sqrt{\kappa_2(A)} - 1}{\sqrt{\kappa_2(A)} + 1}$$

Come per la SD si può dare una stima (pessimistica) del numero di iterazioni necessario per ridurre la norma A dell'errore di una tolleranza ε.

$$k \approx \frac{1}{2} \log \left(\frac{2}{\varepsilon} \right) (\sqrt{\kappa_2(A)} + 1)$$

L'espressione appena scritta indica che il numero di iterazioni è proporzionale a $\sqrt{\kappa_2(A)}$, laddove nel caso della SD il numero di iterazioni è proporzionale a $\kappa_2(A)$.

B.4 Precondizionamento

Il metodo del gradiente coniugato converge velocemente a patto che il numero $\kappa_2(A)$ sia sufficientemente piccolo. Nella pratica, tuttavia, $\kappa_2(A)$ può essere molto grande anche per matrici di piccole dimensioni.
Come esemplificazione, consideriamo il caso di una matrice che deriva dalla discretizzazione con le differenze finite o con gli elementi finiti di un problema di diffusione su una dominio quadrato unitario con una griglia regolare. In questo caso si dimostra che il numero di condizionamento è proporzionale a $(\Delta x)^2$ dove Δx è il passo di discretizzazione spaziale. Ancora più semplicemente $\kappa_2(A)$ è direttamente proporzionale alla dimensione n del sistema lineare.
Se $n \approx 10^5$ possiamo avere matrici il cui numero di condizionamento è dell'ordine di 10^6 da cui $\sqrt{\kappa_2(A)} \approx 10^3$. In tale caso il numero di iterazioni richieste dal CG può essere di svariate migliaia.
Per ottenere una convergenza più rapida, e quindi un metodo efficiente, l'idea del **precondizionamento** consiste nel cercare di ridurre il numero di condizionamento del sistema premoltiplicando il sistema (B.1) per una matrice che indicheremo con M^{-1}.

$$M^{-1}Ax = M^{-1}b \tag{B.9}$$

Il precondizionamento è efficiente innanzitutto se $\kappa_2(M^{-1}A) \ll \kappa_2(A)$. Questa proprietà si verifica in particolare quando $M^{-1} \approx A^{-1}$.

Precondizionare, ovvero premoltiplicare per M^{-1} richiede in ogni caso il

calcolo di $w = M^{-1}z$ per un qualche vettore z, operazione che si esegue risolvendo il sistema

$$Mw = z \tag{B.10}$$

Dunque la seconda condizione per il precondizionatore M^{-1} è che sia computazionalmente poco costoso risolvere un sistema della forma (B.10).

Come abbiamo visto nei paragrafi precedenti il metodo del gradiente coniugato è applicabile solo a matrici simmetriche definite positive. Mostreremo ora come il sistema (B.9) è equivalente ad un sistema la cui matrice è simmetrica definita positiva a patto che lo sia la matrice M.

Se M è definita positiva si può diagonalizzare nel prodotto $Q^T DQ$, con D matrice diagonale e $d_{ii} > 0$. Dunque esiste una matrice simmetrica e definita positiva $X = Q^T D^{1/2} Q$ tale che $M = XX$. Il sistema (B.9) si può scrivere

$$X^{-1}X^{-1}Ax = X^{-1}X^{-1}b$$

o anche, premoltiplicando per X

$$X^{-1}Ax = X^{-1}b.$$

Sia ora $x' = Xx$, $b' = X^{-1}b$, $B = X^{-1}AX^{-1}$.
Notando che $X^{-1}AX^{-1}X^{-1}Xx = X^{-1}b$, possiamo riscrivere il sistema come

$$Bx' = b'. \tag{B.11}$$

La matrice di questo nuovo sistema è ora simmetrica (banale) e definita positiva ($y^T X^{-1}AX^{-1}X^{-1}y = (X^{-1}y)^T A(X^{-1}y) = z^T Az > 0, \forall z \neq 0$).
A tale sistema possiamo applicare il metodo del gradiente coniugato. Si noti che gli autovalori, e quindi il numero di condizionamento, di $X^{-1}AX^{-1}$ sono gli stessi della matrice $M^{-1}A$ per la similitudine tra le due matrici.

B.4.1 CG precondizionato

Partiamo ora dal sistema (B.11) per riscrivere il metodo del gradiente coniugato nelle variabile originarie. Abbiamo le seguenti relazioni:

$$\begin{aligned}
x'_k &= Xx_k, \\
p'_k &= Xp_k, \\
r'_k &= b' - Bx'_k = X^{-1}b - X^{-1}AX^{-1}Xx_k = X^{-1}(b - Ax_k) = X^{-1}r_k
\end{aligned}$$

$$\alpha'_k = -\frac{{r'_k}^T p'_k}{{p'_k}^T B p'_k} = -\frac{r_k^T X^{-1} X p_k}{p_k^T X X^{-1} A X^{-1} X p_k} = \alpha_k$$
$$\beta'_k = \frac{{r'}^T_{k+1} r'_{k+1}}{{r'}^T_k r'_k} = \frac{r_{k+1}^T M^{-1} r_{k+1}}{r_k^T M^{-1} r_k}$$

$$\begin{aligned}
x'_{k+1} = x'_k + \alpha_k p'_k \quad &\Longrightarrow \quad X x_{k+1} = X x_k + \alpha_k X p_k \quad \Longrightarrow \quad x_{k+1} = x_k + \alpha_k p_k \\
r'_{k+1} = r'_k - \alpha_k B p'_k \quad &\Longrightarrow \quad X^{-1} r_{k+1} = X^{-1} r_k - \alpha_k X^{-1} A X^{-1} X p_k \\
&\Longrightarrow \quad r_{k+1} = r_k - \alpha_k A p_k \\
p'_{k+1} = r'_{k+1} + \beta' p'_k \quad &\Longrightarrow \quad X p_{k+1} = X^{-1} r_{k+1} + \beta' X p_k \\
&\Longrightarrow \quad p_{k+1} = M^{-1} r_{k+1} + \beta' p_k
\end{aligned}$$

Siamo ora in grado di scrivere l'algoritmo del gradiente coniugato precondizionato:

ALGORITMO DEL GRADIENTE CONIUGATO PRECONDIZIONATO

In Input: $x_0, A, M, b, k_{\max}$, toll

- $r_0 = b - Ax_0,\ p_0 = M^{-1} r_0,\ k = 0,\ \rho_0 = r_0^T p_0$
- WHILE $\|r_k\| > \text{toll}\ \|b\|$ AND $k < k_{\max}$ DO
  1. $z_k = A p_k$
  2. $\alpha_k = \dfrac{p_k^T r_k}{z_k^T p_k}$
  3. $x_{k+1} = x_k + \alpha_k p_k$
  4. $r_{k+1} = r_k - \alpha_k z_k$
  5. $g_{k+1} = M^{-1} r_{k+1},\ \rho_{k+1} = r_{k+1}^T g_{k+1}$
  6. $\beta_k = \dfrac{r_{k+1}^T M^{-1} r_{k+1}}{r_k^T M^{-1} r_k} = \dfrac{\rho_{k+1}}{\rho_k}$
  7. $p_{k+1} = g_{k+1} + \beta_k p_k$
  8. $k = k + 1$
- END WHILE

B.4.2 Scelta del precondizionatore

Confrontando gli algoritmi del CG si osserva che nel corso di una iterazione il precondizionamento introduce il costo addizionale dovuto al calcolo di $g_{k+1} = M^{-1}r_{k+1}$.
Considereremo ora due possibili scelte di M^{-1}, tra le più popolari nella soluzione di problemi reali di grandi dimensioni.

Precondizionatore di Jacobi

$M = D$, dove D è la matrice diagonale tale che

$$d_{ii} = a_{ii}$$

Il calcolo di $M^{-1}r_{k+1}$ ha un costo computazionale dell'ordine di n operazioni e dunque molto 'economico'. Spesso l'utilizzo di questo semplice precondizionatore è sufficiente per abbattere il numero di iterazioni del gradiente coniugato.

Fattorizzazione incompleta di Cholesky

$M = LL^T$, dove L è il fattore triangolare inferiore della fattorizzazione incompleta di Cholesky. La fattorizzazione incompleta si ottiene a partire dal noto algoritmo della fattorizzazione di Cholesky di una matrice definita positiva, azzerando però nel fattore tutti quegli elementi che corrispondono a uno zero nella matrice A. Formalmente:

$$a_{ij} = 0 \Longrightarrow l_{ij} \equiv 0$$

In tale modo si tiene sotto controllo il numero di coefficienti diversi da zero del fattore stesso.
Al termine di questa fattorizzazione incompleta si ha $A = LL^T + E$, e dunque si può supporre che M^{-1} sia una buona approssimazione di A^{-1}.
Il calcolo di $g_{k+1} = M^{-1}r_{k+1}$ si effettua mediante la soluzione di due sistemi triangolari sparsi: da $g_{k+1} = (LL^T)^{-1}r_{k+1}$ si ottiene $LL^T g_{k+1} = r_{k+1}$ da cui

$$\begin{aligned} Ly &= r_{k+1} \\ L^T g_{k+1} &= y \end{aligned}$$

Il costo per iterazione grosso modo raddoppia (la soluzione di due sistemi triangolari costa poco più di un prodotto matrice per vettore), ma l'accceleraZione che ne deriva produce una riduzione notevole del numero di iterazioni anche per problemi malcondizionati.

Esempio numerico

Il seguente esempio numerico evidenzia l'efficienza del precondizionamento rispetto al CG non precondizionato. Si tratta di risolvere un sistema lineare in cui la matrice A deriva dalla discretizzazione mediante elementi finiti di una equazione di diffusione su un dominio bidimensionale. La dimensione è $n = 77120$, il numero di elementi diversi da zero è 384320 (matrice sparsa) e infine $\kappa_2(A) = 4 \times 10^6$. Implementando il metodo del CG su un pentium 1GHz nel linguaggio Fortran si ottengono i seguenti risultati con una tolleranza toll=10^{-7}:

precondizionatore	iterazioni	tempo di cpu	cpu per iter
-	6022	60.78	0.010
Jacobi	1299	14.00	0.011
Cholesky	246	6.76	0.027

B.5 Soluzione di sistemi lineari non simmetrici

Circa 20 anni fa i metodi basati sulla minimizzazione di funzioni in sottospazi di Krylov sono diventati popolari in particolare per la risoluzione numerica di sistemi lineari, e di problemi agli autovalori, sparsi e di grandi dimensioni. Dato un vettore arbitrario v_1, si definisce spazio di Krylov di dimensione m lo spazio generato dalla seguente successione di vettori:

$$K_m(v_1) = \text{span}\left(v_1, Av_1, A^2v_1, \ldots, A^{m-1}v_1\right)$$

Definito uno spazio di Krylov, tali metodi cercano la soluzione approssimata del sistema lineare

$$Ax = b$$

minimizzando il residuo $b - Ax_m$ tra tutti i vettori x_m della forma:

$$x_m = x_0 + y, \qquad y \in K_m$$

Tutti i metodi di questo tipo, hanno *terminazione finita* in aritmetica esatta, cioè convergono alla soluzione esatta in al più n iterazioni dove n è la dimensione del sistema. Infatti, nel caso $m = n - 1$, la minimizzazione del residuo è effettuata su uno spazio di dimensione n e dunque il minimo vale 0 e si ottiene per $x_n = A^{-1}b$.
Vediamo più in dettaglio come si effettua la minimizzazione, per ogni singolo passo m. Innanzitutto occorre sottolineare che la base

$$\{v_1, Av_1, A^2v_1, \ldots, A^{m-1}v_1\}$$

di K_m ha il difetto di essere 'poco linearmente indipendente'. Per m grande i vettori di tale base tendono a dei multipli dell'autovettore relativo all'autovalore massimo. Una base più opportuna dello spazio di Krylov K_m è rappresentata da una base ortonormale, che si costruisce a partire dal vettore v_1 tramite il processo di ortogonalizzazione di Gram-Schmidt. Chiameremo $v_1, v_2, \ldots, v_m$ i vettori che compongono tale base.
Una volta nota la base dello spazio di Krylov, occorre effettuare una minimizzazione locale del residuo, cioè tra tutte le x_m del tipo

$$x_m = x_0 + \sum_{i=1}^{m} z_i v_i \tag{B.12}$$

trovare quella per cui la norma del residuo è minore.

B.5.1 Il metodo GMRES

Il metodo GMRES è stato proposto nel 1986 da Saad e Schultz [36] come un metodo di Krylov per sistemi nonsimmetrici. GMRES costruisce una successione di vettori $x_0, x_1, \ldots, x_m$ in modo tale da minimizzare la norma euclidea del residuo $\|b - Ax\|$ tra tutti i vettori della forma

$$x = x_0 + y, \qquad y \in K_m$$

Lo spazio di Krylov K_m è generato dal residuo iniziale $r_0 = b - Ax_0$. Dunque abbiamo

$$K_m(r_0) = \text{span}\left(r_0, Ar_0, A^2 r_0, \ldots, A^{m-1} r_0\right)$$

Il primo passo come si è detto è di trovare una base ortonormale di tale spazio, dunque costruiamo la nuova base dello spazio di Krylov applicando la ortonormalizzazione di Gram-Schmidt come segue: $\beta = \|r_0\|$, $v_1 = \dfrac{r_0}{\beta}$
DO $k = 1, m$

1. $w_{k+1} = Av_k$
2. $h_{jk} = w_{k+1}^T v_j, \qquad j = 1, \ldots k$
3. $v_{k+1} = w_{k+1} - \sum_{j=1}^{k} h_{jk} v_j$
4. $h_{k+1,k} = \|v_{k+1}\|$
5. $v_{k+1} = v_{k+1}/h_{k+1,k}$

END DO

I vettori generati da questo procedimento verificano la relazione seguente

$$Av_k = \sum_{j=1}^{k+1} h_{jk} v_j, \qquad k = 1, \ldots, m$$

che può essere riscritta in forma matriciale:

$$AV_m = V_{m+1} H_m$$

dove si è posto

$$V_m = [v_1, v_2, \ldots v_m]$$

e H_m è la matrice che ha come elementi gli h_{jk} definiti nell'algoritmo. Tale matrice (rettangolare, di dimensioni $m+1 \times m$) è detta di Hessenberg superiore ed ha la seguente forma:

$$H_m = \begin{pmatrix} h_{11} & h_{12} & \ldots & & \\ h_{21} & h_{22} & h_{23} & \ldots & \\ 0 & h_{32} & h_{33} & h_{34} & \\ & \ldots & \ldots & \ldots & \\ & & 0 & h_{m,m-1} & h_{mm} \\ & & & 0 & h_{m+1,m} \end{pmatrix} \tag{B.13}$$

Il secondo passo consiste nell'effettuare la minimizzazione del residuo. Ricordiamo l'espressione generica per $x_m = x_0 + y, \quad y \in K_m$. Ora un vettore generico di K_m può essere espresso come combinazione lineare dei vettori di una sua base pertanto:

$$y = \sum_{j=1}^{m} z_i v_i$$

e quindi

$$x_m = x_0 + \sum_{j=1}^{m} z_i v_i$$

Le incognite sono ora i coefficienti z_i, che formano il vettore z. Ricaviamo un'espressione più compatta per x_m:

$$x_m = x_0 + V_m z$$

Il nostro problema è di trovare il vettore x_m tra tutti quelli esprimibile come sopra, per il quale la norma euclidea del residuo sia minima. Vogliamo dunque minimizzare la funzione

$$J(z) = \|b - Ax_m\| = \|b - A\,(x_0 + V_m z)\,\|$$

$$\begin{aligned} r_m = b - Ax_m &= b - A\left(x_0 + V_m z\right) = \\ &= r_0 - AV_m z = \\ &= \beta v_1 - V_{m+1} H_m z = \\ &= V_{m+1}\left(\beta e_1 - H_m z\right) \end{aligned} \tag{B.14}$$

dove con e_1 si intende il vettore $[1, 0, \ldots, 0]^T$. Pertanto, per la ortonormalità della matrice V_{m+1} si ha

$$\begin{aligned} J(z) &= \sqrt{r_m^T r_m} = \\ &= \sqrt{\left(\beta e_1 - H_m z\right)^T V_{m+1}^T V_{m+1}\left(\beta e_1 - H_m z\right)} = \\ &= \sqrt{\left(\beta e_1 - H_m z\right)^T \left(\beta e_1 - H_m z\right)} = \\ &= \| \left(\beta e_1 - H_m z\right) \| \end{aligned} \tag{B.15}$$

e dunque $z = \{z : \|\beta e_1 - H_m z\| \ \text{ minimo}\}$.
Si noti che tale problema ha una dimensione molto piccola rispetto alla dimensione del sistema originale (al massimo qualche centinaio). Se H_m fosse una matrice quadrata, il problema sarebbe equivalente a risolvere il sistema $H_m z = \beta e_1$. Essendo invece H_m una matrice rettangolare $(m+1 \times m)$, il sistema è sovradeterminato, cioè, in generale, non ha soluzioni (cfr. [45, sez. 6.1.3]). Per risolvere il problema di minimo occorre pertanto fattorizzare la matrice H_m in un prodotto di una matrice ortogonale per una matrice triangolare superiore: $H_m = QR$ (fattorizzazione QR, cfr. [45, sez. 3.2.9]) .

$$\min \|\beta e_1 - H_m z\| = \min \|Q\left(\beta Q^T e_1 - Rz\right)\| = \min \|g - Rz\|$$

dove $g = \beta Q^T e_1$. La matrice R ha la forma seguente:

$$R = \begin{pmatrix} r_{11} & r_{12} & \ldots & & \\ 0 & r_{22} & r_{23} & \ldots & \\ & \ldots & \ldots & \ldots & \\ & & & 0 & r_{mm} \\ & & & & 0 \end{pmatrix} \tag{B.16}$$

la soluzione del nuovo problema di minimo è effettuata risolvendo il sistema $\tilde{R}z = \tilde{g}$ dove $\tilde{R}$ è la matrice ottenuta da R togliendo l'ultima riga di zeri e $\tilde{g}$ il vettore che si ottiene da g togliendo la sua ultima componente. Essendo $\tilde{R}$ una matrice triangolare superiore, tale sistema si risolve facilmente con le "sostituzione all'indietro" (cfr. [45, sez. 3.2.2]). Si noti che la funzione $J(z) = \|b - Ax_m\|$ assume come valore minimo $\|g_{m+1}\|$.

ALGORITMO GMRES

In Input: $x_0, A, b, k_{\max}$, toll

- $r_0 = b - Ax_0,\ k = 0,\ \rho_0 = \|r_0\|,\ \beta = \rho_0,\ v_1 = \dfrac{r_0}{\beta}$
- WHILE $\rho_k >$ toll $\|b\|$ AND $k < k_{\max}$ DO
  1. $k = k + 1$
  2. $v_{k+1} = Av_k$
  3. DO $j = 1, k$
     $h_{jk} = v_{k+1}^T v_j$
     $v_{k+1} = v_{k+1} - h_{jk} v_j$
     END DO
  4. $h_{k+1,k} = \|v_{k+1}\|$
  5. $v_{k+1} = v_{k+1}/h_{k+1,k}$
  6. $z_k = \arg\min\|\beta e_1 - H_k z\|$ con $z \in \mathbb{R}^{k+1}$
  7. $\rho_k = \|\beta e_1 - H_k z_k\|$
- END WHILE
- $x_k = x_0 + V_k z_k$

B.5.2 GMRES con restart

Lo svantaggio principale del metodo GMRES è costituito dal fatto che occorre tenere in memoria tutti i vettori della base dello spazio di Krylov. Quando il numero di iterazioni raggiunge valori di qualche centinaio (cosa che avviene nei casi reali) questa occupazione di memoria può diventare eccessivamente gravosa (si noti che nel caso di matrici sparse l'occupazione di memoria dovuta alla matrice è dell'ordine di $10n$). Tale problema viene risolto nelle applicazioni definendo a priori un numero massimo di vettori che possono essere memorizzati, diciamo p. Dopo p iterazioni si calcola l'approssimazione x_p, si cancellano tutti i vettori dello spazio di Krylov e si riparte con un nuovo spazio di Krylov avente come vettore generatore $r_p = b - Ax_p$. Tale modifica risolve il problema della memoria ma rallenta l'algoritmo dal punto di vista della sua convergenza alla soluzione.

B.5.3 Precondizionamento

Come nel caso del CG, anche il metodo GMRES deve essere opportunamente precondizionato per garantire una convergenza veloce. Analogamente ai sistemi simmetrici, si possono impiegare il precondizionatore di Jacobi e il precondizionatore basato questa volta sulla fattorizzazione incompleta LU (precondizionatore ILU = Incomplete LU). L'unica variazione all'algoritmo originale avviene al punto 2. dove, invece di calcolare il prodotto $v_{k+1} = Av_k$ si calcola $v_{k+1} = M^{-1}Av_k$:

$$t = Av_k, \qquad v_{k+1} = M^{-1}t$$

dove l'ultima operazione si esegue come descritto in sezione B.4.2.
La figura riporta il profilo di convergenza del metodo GMRES applicate ad una matrice "piccola" ($n = 27$), utilizzando i due precondizionatori descritti e diversi valori del parametro di restart p (nrest nella figura).

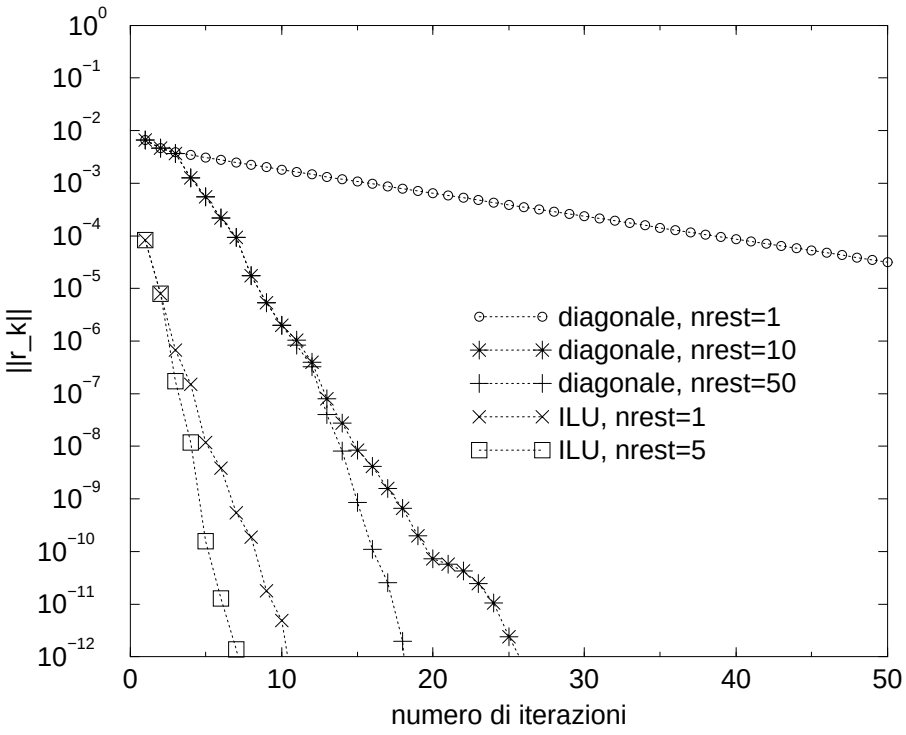


Figura B.1: Profilo di convergenza del metodo GMRES applicato ad una matrice non simmetrica di ordine $n = 27$.

B.5.4 Altri metodi per sistemi nonsimmetrici

Diversamente dal metodo GMRES, vi sono altri metodi che non soddisfano un principio di minimizzazione; essi sono perlopiù basati su una formula di *bi-ortogonalizzazione* e richiedono una occupazione di memoria inferiore a quella richiesta dal GMRES. Tra di essi citiamo il metodo Bi-CG per il quale il residuo soddisfa:

$$r_k^T w = 0, \qquad \forall w \in \hat{K}_k$$

dove $\hat{K}_k$ è lo spazio generato da

$$\{\hat{r}_0, A^T\hat{r}_0, (A^T)^2\hat{r}_0, \ldots, (A^T)^{m-1}\hat{r}_0\}$$

e $\hat{r}_0$ è un vettore iniziale che spesso si pone uguale a r_0. Il metodo Bi-CG costruisce 4 successioni di vettori: $\{\hat{r}_k\}, \{r_k\}, \{\hat{p}_k\}, \{r_k\}$, che verificano

$$\hat{r}_{k+1}^T r_i = 0, \qquad \hat{p}_k^T A p_i = 0, \qquad i \leq k$$

Se A è simmetrica definita positiva Bi-CG coincide con CG.

La variante attualmente più nota del Bi-CG è il metodo BiCGstab (Bi-CG stabilizzato) [39, 25] che risolve il problema della convergenza instabile che si osserva nel Bi-CG.

ALGORITMO BiCGSTAB

Input: $x_0, A, b, k_{\max}$, toll

- $r_0 = b - Ax_0,\ k = 0,$ si scelga $\hat{r}_0$ in modo che $\hat{r}_0^T \hat{r}_0 \neq 0$
- $\rho_{-1} = \alpha_{-1} = \eta_0 = 1;$
- WHILE $\|r_k\| > \text{toll}\ \|b\|$ AND $k < k_{\max}$ DO
  1. $\rho_k = \hat{r}_0^T r_k$
  2. $\beta_k = (\rho_k/\rho_{k-1})(\alpha_{k-1}/\eta_k)$
  3. $p_k = r_k + \beta_k(p_{k-1} - \eta_k v_{k-1})$
  4. $v_k = Ap_k$
  5. $\alpha_k = \rho_k/\hat{r}_0^T v_k$
  6. $s = r_k - \alpha_k v_k$
  7. $t = As$
  8. $\eta_{k+1} = s^T t/t^T t$
  9. $c_{k+1} = c_k + \alpha_k p_k + \eta_{k+1} s$
  10. $r_{k+1} = s - \eta_{k+1} t$
  11. $k = k + 1$
- END WHILE

Bibliografia

[1] M. Bazaraa, H. Sherali, and C. Shetty. *Nonlinear Programming, Theory and Applications*. J. Wiley & Sons, New York, (Third Edition) 2006.

[2] S. Bellavia. An inexact interior point method. *Journal of Optimization Theory and Applications*, 96, 1998.

[3] L. Bergamaschi, J. Gondzio, M. Venturin, and G. Zilli. Inexact constraint preconditioners for linear systems arising in interior point methods. *Computational Optimization and Applications*, pages 137–147, 2007.

[4] L. Bergamaschi, J. Gondzio, and G. Zilli. Preconditioning indefinite systems in interior point methods for optimization. *Computational Optimization and Applications*, pages 149–171, 2004.

[5] L. Bergamaschi, J. Gondzio, and G. Zilli. Coap 2004 best paper award. *Computational Optimization and Applications*, pages 363–365, 2005.

[6] R. Bertelle. *Metodi di Trust Region*. Tesina per il corso di metodi di ottimizzazione, Dottorato in Matematica Computazionale , A.A. 2003/04, Padova, 2004.

[7] F. Bettin. *Uso di metodi secanti per sistemi di equazioni non lineari nella risoluzione di sistemi di equazioni non lineari e lineari*. Tesi per il corso di metodi numerici e calcolo parallelo, a.a. 1994/95, Corso di Perfezionamento in Matematica Applicata e Programmazione, Padova, 1995.

[8] D. Bini, M. Capovani, and O. Menchi. *Metodi Numerici per l'Algebra Lineare*. Zanichelli, Bologna, 1988.

[9] S. Bonettini. An nonmonotone inexact newton method. *Optimization Methods and Software*, 20:475–491, 2005.

[10] P. G. Ciarlet. *Introduzione all'analisi numerica matriciale e all'ottimizzazione*. Masson, 1989.

[11] T. Coleman, M. A. Branch, and A. Grace. *Optimization Toolbox User's Guide*. The Math Works Inc, Natick MA, 1999.

[12] T. F. Coleman and A. Verna. *Automatic differentiation and Matlab interface toolbox*. Masson, Cornell University, 1998.

[13] V. Comincioli. *Analisi Numerica*. McGraw-Hill, Milano, 1990.

[14] I. Cunegato. *Soluzioni di sistemi non lineari con metodi di Newton e quasi Newton: Tests numerici in ambiente MATLAB*. Tesina per il corso di metodi numerici e calcolo parallelo, a.a. 1997/98, Corso di Perfezionamento in Matematica Applicata e Programmazione, Padova, 1998.

[15] J. E. Dennis, Jr., and S. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, NJ, 1983 and 1996.

[16] C. Durazzi. On the newton interior-point method for large scaling non linear programming problems. *Journal of Optimization Theory and Applications*, 104:333–357, 2000.

[17] T. F. Edgar and D. M. Himmelblau. *Optimization of chemical processes*. Mc Graw Hill, Boston, 1988.

[18] R. Fletcher. An overview of uncostrained optimization. In E. Spedicato, editor, *Algorithms for Continous Optimization*, pages 109–143. Kluwer Academic Publishers, (1993)-94.

[19] F. Fontanella and A. Pasquali. *Calcolo Numerico. Metodi e Algoritmi*. Pitagora Editrice, Bologna, 1982.

[20] A. Ghizzetti. *Lezioni di Analisi Matematica*. Veschi, Roma, 1971-72.

[21] J. Gondzio. HOPDM (version 2.12) – a fast LP solver based on a primal-dual interior point method. *European Journal of Operational Research*, 85:221–225, 1995.

[22] N. Gould and S. Leyffer. *An introduction to algorithms for nonlinear optimization*. RAL-TR, 2002-031.

[23] F. Granziero. *Introduzione ai metodi interior point.* Tesi per il corso di metodi numerici e calcolo parallelo, a.a. 1997/98, Corso di Perfezionamento in Matematica Applicata e Programmazione, Padova, 1998.

[24] C. T. Kelley. *Iterative Methods for Linear and non Linear Equations.* Frontiers in Applied Mathematics, Siam, Philadelphia, 1995.

[25] C. T. Kelley. *Iterative Methods for Optimization.* Frontiers in Applied Mathematics, Siam, Philadelphia, 1999.

[26] G. Liuzzi. *Appunti sulla Programmazione Multiobiettivo.* www.dis.uniroma1.it/∼liuzzi, Università di Roma, 2000.

[27] D. G. Luenberger. *Linear and non linear programming.* Addison-Wesley, Reading Mass, 1984.

[28] F. Maffioli. *Elementi di Programmazione Matematica vol. 2.* Masson, Milano, 1991.

[29] A. Mancini. *Approssimazione non lineare ai minimi quadrati. Metodo di Levenberg-Marquardt.* Tesi per il corso di metodi numerici e calcolo parallelo, a.a. 1995/96, Corso di Perfezionamento in Matematica Applicata e Programmazione, Padova, 1996.

[30] O. L. Mangasarian. *Nonlinear programming.* McGraw–Hill, New York, 1969, 1969.

[31] K. M. Miettinen. *Nonlinear Multiobjective Optimization.* Kluwer Academic Publishers, Boston, (Fourth Edition) 2004.

[32] J. J. Morè and S. J. Wright. *Optimization Software Guide.* SIAM, Philadelphia., 1993.

[33] J. Nocedal. Large scale uncostrained optimization. In *The State of the Art of N.A*, pages 311–338. I.N.A, 1997.

[34] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer-Verlag, New York, 1999.

[35] W. Press et al. *Numerical Recipes in Fortran.* Cambridge U.P.Springer, 1992.

[36] Y. Saad and M. H. Schultz. Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, pages 856–869, (7) 1996.

[37] D. F. Shanno and E. M. Simantiraki. Interior point methods and non linear programming. In *The State of the Art of N.A*, pages 339–362. I.N.A, 1997.

[38] D. F. Shanno and R. J. Vanderbei. Interior point methods for nonconvex nonlinear programming. In *IFIP Conference*, page 50. Cambridge, England, 1999.

[39] H. A. van der Vorst. Bi-cgstab: A fast and smoothly converging variant of BI-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, pages 631–644, (13) 1992.

[40] M. Venturin. *Metodi di Line Search.* Tesina per il corso di metodi di ottimizzazione, Dottorato in Matematica Computazionale , A.A. 2002/03, Padova, 2003.

[41] S. J. Wright. *Primal-Dual Interior-point methods.* SIAM, Philadelphia, 1997.

[42] U. Meier Yang and K. A. Gallivan. A new family of preconditioned iterative solvers for nonsymmetric linear systems. *Applied Numerical Mathematics*, pages 287–317, 1993.

[43] G. Zilli. Parallel method for sparse nonsymmetric linear and non linear systems of equations on a transputer network,. *Supercomputer 66-XII-4*, pages 4–15, 1996.

[44] G. Zilli. *Metodi Variazionali per Equazioni Differenziali.* Imprimitur, Padova, 1998.

[45] G. Zilli. *Calcolo Numerico: Lezioni ed Esercizi.* Edizioni Libreria Progetto, Padova, Edizione 2008.

[46] G. Zilli and L. Bergamaschi. Parallel newton methods for sparse systems of nonlinear equations. *Rend. Circ. Matem. Palermo, Serie II*, pages 247–257, 1999.

Indice analitico