Metodi per la soluzioni di problemi non vincolati che non fanno uso di derivate

Veronica Piccialli

Dottorato in Ingegneria dei Sistemi, corso di Ottimizzazione Continua

Indice

1 Introduzione ai metodi per problemi non vincolati **3**
1.1 Metodi alle differenze finite 5
1.2 Metodi di Ricerca diretta 6
1.3 Metodi di modellizzazione 7

2 Metodi euristici di ricerca diretta **9**
2.1 Algoritmo di Nelder e Mead 9

3 Metodi di ricerca diretta convergenti **17**
3.1 Teoria generale di convergenza 17
3.2 Metodi di tipo Pattern Search 23
3.2.1 Metodo delle coordinate 28
3.2.2 Algoritmo di Hooke e Jeeves 32
3.3 Metodi di tipo linesearch 33
3.4 Metodi pattern-line 38

Capitolo 1

Introduzione ai metodi per problemi non vincolati

Un problema di *programmazione matematica* assume la forma generale

$$\min_{x \in X} f(x), \tag{1.1}$$

dove

- $f : R^n \to R$ è la *funzione obiettivo*;
- $X \subseteq R^n$ è l'*insieme ammissibile delle soluzioni*.

Nei problemi *vincolati* risulta $X \subset R^n$, in quelli *non vincolati* si ha $X = R^n$.
Molti problemi di ottimizzazione derivanti da applicazioni reali sono caratterizzati dal fatto che l'espressione analitica della funzione obiettivo e/o dei vincoli non è nota. Tale situazione si verifica, ad esempio, quando sistemi fisici complessi vengono descritti, analizzati e controllati ottimizzando risultati di simulazioni al computer. In questo contesto le grandezze necessarie al processo di ottimizzazione vengono calcolate mediante simulazioni ripetute ed ogni simulazione può coinvolgere diversi programmi tra loro indipendenti. I dati così ottenuti vengono poi ulteriormente processati per calcolare la funzione obiettivo e/o i vincoli. E` chiaro che in questo caso non è possibile (o comunque richiede un costo troppo elevato) calcolare le derivate, anche quando i fenomeni fisici considerati potrebbero essere rappresentati per loro natura tramite funzioni "smooth" (in generale i fenomeni naturali sono continui). L'interesse applicativo ha motivato quindi lo sviluppo di metodi di ottimizzazione che non richiedano la conoscenza delle derivate.

Storicamente i primi metodi senza derivate sono stati introdotti già negli anni '50, ma sono poi stati abbandonati nei primi anni '70 per mancanza di un'analisi teorica rigorosa e per la bassa velocità di convergenza dimostrata. Solo ultimamente ([19]) l'interesse della comunità scientifica si è risvegliato grazie ad una serie di articoli che dimostrano proprietà teoriche di convergenza globale per algoritmi senza derivate.
Considereremo inizialmente problemi di ottimizzazione non vincolata, cioè problemi del tipo

$$\min_{x \in R^n} f(x), \tag{1.2}$$

e assumeremo che la funzione obiettivo sia *continuamente differenziabile*.
In generale, gli algoritmi (con e senza derivate) proposti in letteratura consentono soltanto la determinazione di punti stazionari di f, cioè di punti che soddisfano le condizioni di ottimalità del primo ordine e che quindi appartengono all'insieme

$$\Omega = \{x \in R^n : \nabla f(x) = 0\}$$

Notiamo che non avendo a disposizione le derivate non è possibile verificare direttamente l'appartenenza di un punto all'insieme Ω tramite la valutazione del gradiente. Si deve quindi utilizzare un criterio diverso per stabilire se un punto appartenga all'insieme Ω o meno. La potenza dei metodi senza derivate è proprio quella di riuscire a garantire la convergenza a punti stazionari senza fare esplicitamente uso del valore del gradiente.
Anche nel caso di algoritmi senza derivate lo schema generale di un algoritmo di minimizzazione è il seguente.

1. Si fissa un punto iniziale $x_0 \in R^n$.

2. Se $x_k \in \Omega$ stop.

3. Si calcola una *direzione di ricerca* $d_k \in R^n$.

4. Si calcola un *passo* $\alpha_k \in R$ lungo d_k.

5. Si determina un nuovo punto $x_{k+1} = x_k + \alpha_k d_k$, si pone $k = k + 1$ e si ritorna al Passo 2.

Quello che caratterizza il singolo metodo è la scelta della direzione di ricerca $d_k \in R^n$ e la scelta del passo $\alpha_k \in R$.
In generale, per i metodi che utilizzano le derivate, la conoscenza del gradiente permette

- di definire e calcolare una la direzione d_k che sia una direzione di discesa in x_k, cioè sia tale che

$$\nabla f(x_k)^T d_k < 0, \tag{1.3}$$

- di scegliere lungo la direzione scelta un passo α_k che soddisfi (almeno) una condizione tipo Armijo

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \gamma \alpha_k \nabla f(x_k)^T d_k \tag{1.4}$$

che garantisce di sfruttare al meglio la proprietà di discesa della direzione ottenendo un sufficiente decremento del valore della funzione obiettivo.

Le condizioni (1.3) e (1.4) non sono ovviamente utilizzabili quando il gradiente della funzione obiettivo non è noto, e ciò evidenzia la difficoltà che caratterizza il progetto di metodi senza derivate. Questi metodi possono essere raggruppati in tre classi:

- metodi che fanno uso di *approssimazioni alle differenze finite*;
- metodi di *ricerca diretta*.
- metodi di *modellizzazione*.

1.1 Metodi alle differenze finite

L'idea alla base dei metodi di questa classe è di approssimare, ad ogni iterazione k, le componenti del gradiente con formule *alle differenze in avanti* del tipo

$$\tilde{\nabla}_j f(x_k) = \frac{f(x_k + h_j e_j) - f(x_k)}{h_j}$$

in cui h_j è il *passo di discretizzazione* e e_j rappresenta il $j - esimo$ asse coordinato. Il gradiente così approssimato può essere utilizzato all'interno di efficienti metodi *standard* di ottimizzazione, quali metodi quasi-Newton o di tipo gradiente coniugato. Osserviamo però che il calcolo numerico delle derivate è affetto da

- *un errore di troncamento* proporzionale al passo di discretizzazione. Infatti se consideriamo lo sviluppo in serie di Taylor della funzione $f(x)$ arrestato al primo ordine

abbiamo

$$f(x_k + h_j e_j) = f(x_k) + \nabla f(x_k)^T (h_j e_j) + o(h_j) = f(x_k) + h_j \nabla_j f(x_k) + o(h_j).$$

da cui si ricava

$$\nabla_j f(x_k) = \frac{f(x_k + h_j e_j) - f(x_k)}{h_j} - \frac{o(h_j)}{h_j}.$$

- *un errore nel calcolo dei valori della funzione*, nel senso che se indichiamo con $\hat{f}(x_k + h_j e_j)$, $\hat{f}(x_k)$ i valori calcolati di $f(x_k + h_j e_j)$ e $f(x_k)$ si può porre

$$f(x_k + h_j e_j) = \hat{f}(x_k + h_j e_j) + \epsilon_1 \qquad f(x_k) = \hat{f}(x_k) + \epsilon_2$$

in cui ϵ_1, ϵ_2 sono gli errori nel calcolo dei valori di f, dovuti, tipicamente, alla presenza di rumore. Si avrà allora un errore sulla componente del gradiente dato da

$$\frac{\epsilon_1 - \epsilon_2}{h}$$

L'effetto dell'errore di troncamento richiederebbe un passo di discretizzazione "piccolo", che amplificherebbe però l'errore nel calcolo dei valori della funzione obiettivo. In pratica quindi, i metodi basati sulle approssimazioni alle differenza finite presentano un comportamento soddisfacente in problemi di ottimizzazione in cui la presenza di rumore è trascurabile. Nei problemi (molto frequenti) in cui questa ipotesi non si verifica, i metodi alle differenze finite non sono in grado di fornire soluzioni accettabili, ed è quindi preferibile l'uso di metodi di ricerca diretta o di modellizzazione.

1.2 Metodi di Ricerca diretta

Con il nome di metodi di ricerca diretta ("direct search") si indica un insieme di metodi accomunati dall'idea di basare la minimizzazione sul confronto diretto dei valori della funzione obiettivo nei punti generati dall'algoritmo.
All'interno di questa classe di metodi si trovano algoritmi euristici, che non hanno cioè proprietà teoriche, e algoritmi per cui invece si può dimostrare la convergenza a punti stazionari. In particolare, si possono distinguere tre classi di metodi di ricerca diretta:

- metodi di tipo *simplesso*

- metodi di tipo *pattern search*

- metodi di tipo *line search*

L'algoritmo base dei metodi di tipo simplesso è l'algoritmo di Nelder e Mead, che consideriamo in dettaglio, pur essendo un'euristica, perchè su molti problemi funziona bene ed è in pratica molto utilizzato.

1.3 Metodi di modellizzazione

Questi metodi ([1],[2][15],[16],[12]) si basano sull'idea di definire un modello analitico della funzione obiettivo, che può essere lineare o quadratico, e di applicare a questo modello una tecnica di minimizzazione che utilizza informazioni sulle derivate. In particolare, il modello utilizzato è della forma

$$m_k(x_k + s) = f(x_k) + g_k^T s + \frac{1}{2} s^T H_k s \tag{1.5}$$

dove $g_k \in \Re^n$, $H_k \in \Re^{n \times n}$ sono tali che il modello $m_k(x_k + s)$ è un interpolante della funzione su un opportuno insieme di punti Y che viene aggiornato durante l'avanzare dell'algoritmo. Si ha quindi

$$m_k(y) = f(y), \quad \forall y \in Y. \tag{1.6}$$

Il modello $m_k(x_k+s)$ viene minimizzato all'interno di una sfera centrata nel punto corrente di raggio Δ_k, dove Δ_k rappresenta una stima del raggio della regione in cui il modello è affidabile. Sul punto prodotto dalla minimizzazione viene valutata la funzione obiettivo reale $f(x)$ e in base al valore ottenuto si aggiorna l'insieme di punti Y e il raggio della sfera Δ_k. Nella definizione e nell'aggiornamento dell'insieme Y si deve far si che i punti $y \in Y$ soddisfino delle relazioni geometriche che garantiscano che Y sia tal da rappresentare adeguatamente la funzione obiettivo nell'intorno del punto corrente.
Questi algoritmi sono stati essenzialmente progettati per la soluzione di problemi in cui la valutazione della funzione obiettivo è particolarmente onerosa, e prevedono l'impiego di tecniche sofisticate di algebra lineare. L'analisi teorica delle proprietà di convergenza di algoritmi di questo tipo richiede l'introduzione di opportune ipotesi ed è piuttosto complessa.
Nell'ambito dei metodi di modellizzazione, una strategia alternativa è quella di definire "fuori linea" una funzione differenziabile approssimante (ad esempio, un modello neurale)

e di validare l'affidabilità del modello ottenuto. Una volta certificato che il modello approssima con un grado di accuratezza sufficiente la funzione obiettivo, si può pensare di applicare un algoritmo standard (tipo gradiente, gradiente coniugato, quasi-Newton) per la minimizzazione del modello ottenuto. Questo tipo di strategia è stata vantaggiosamente utilizzata per la soluzione di specifici problemi applicativi.

Capitolo 2

Metodi euristici di ricerca diretta

2.1 Algoritmo di Nelder e Mead

L'algoritmo di Nelder e Mead è stato introdotto nel 1965 in [14] ed è rimasto uno dei più popolari algoritmi senza derivate per l'efficienza dimostrata soprattutto per problemi di piccole dimensioni. Questo metodo si sposta nello spazio delle soluzioni tramite simplessi, cioè figure geometriche che soddisfano la seguente definizione:

Definizione 2.1 *Si definisce* simplesso *S l'involucro convesso di $n+1$ punti $\{x_i \in \Re^n\}_{i=1}^{n+1}$ (detti* vertici *del simplesso), cioè*

$$S = \{y \in \Re^n : \ y = \sum_{i=1}^{n+1} \lambda_i x_i, \ \lambda_i \geq 0, \ \sum_{i=1}^{n} \lambda_i = 1\}.$$

Un simplesso S si dice non singolare *se gli n vettori $\{x_2 - x_1, \ldots, x_{n+1} - x_1\}$ sono tra loro linearmente indipendenti.*

In figura 2.1 (a) e (b) sono riportati rispettivamente un esempio di simplesso non singolare e uno di simplesso singolare.
La scelta della figura geometrica del simplesso è dovuta principalmente a due motivi: la capacità del simplesso di adattare la sua forma all'andamento nello spazio della funzione obiettivo deformandosi (allungandosi o schiacciandosi), e il fatto che richiede la memorizzazione di soli $n + 1$ punti.
L'algoritmo tiene in memoria gli $n + 1$ vertici del simplesso corrente con i rispettivi valori della funzione obiettivo e ad ogni iterazione tenta di generare un nuovo simplesso sostituendo il punto a cui corrisponde il valore massimo di f con un nuovo punto, scelto in

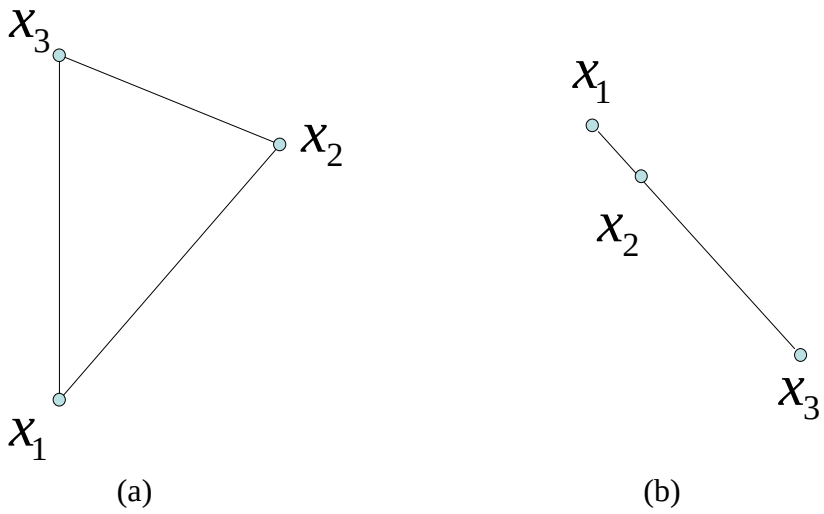

Figura 2.1:

maniera opportuna, in cui la funzione obiettivo abbia valore inferiore. In particolare, gli $n+1$ vertici $\{x_j\}_{j=1}^{n+1}$ del simplesso sono ordinati in base al valore della funzione obiettivo

$$f(x_1) \leq \ldots \leq f(x_{n+1}).$$

L'algoritmo cerca di sostituire il peggior vertice x_{n+1} con un nuovo punto della forma:

$$x(\mu) = (1+\mu)x_c - \mu x_{n+1}, \tag{2.1}$$

dove x_c è il *centroide* degli n punti rimanenti, ossia il punto

$$x_c = \frac{1}{n}\sum_{j=1}^{n} x_j. \tag{2.2}$$

L'idea è quella quindi di operare una *riflessione* di x_{n+1} rispetto a x_c, dove $\mu > 0$ è un opportuno coefficiente di riflessione. Tipicamente il valore di μ è selezionato da una sequenza di valori così scelti:

$$-1 \leq \mu_{ic} < 0 < \mu_{oc} < \mu_r < \mu_e.$$

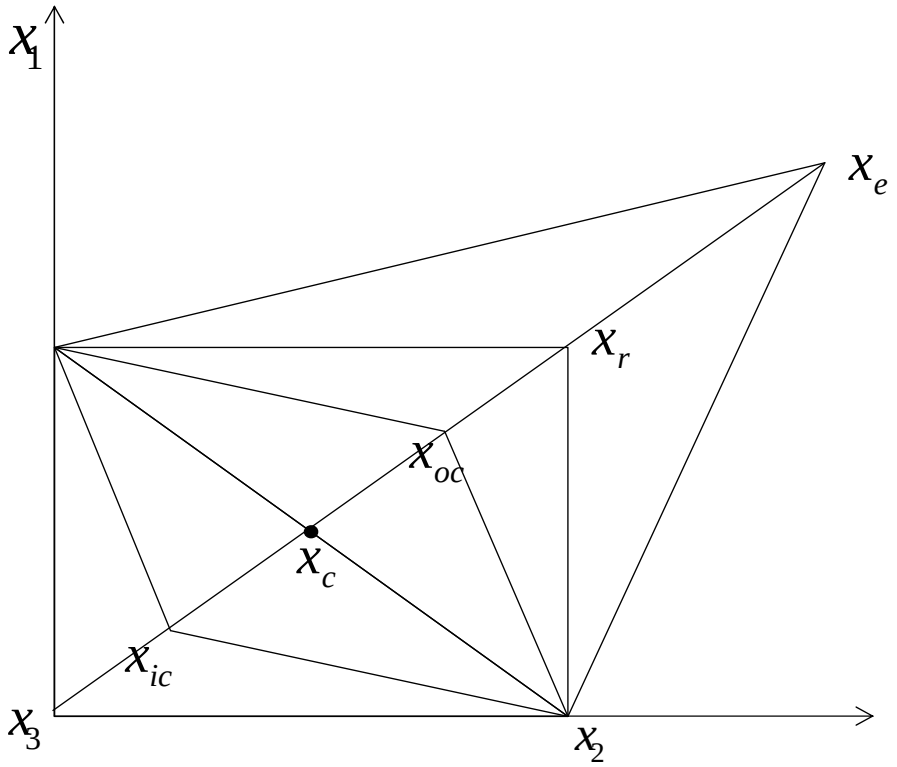

Figura 2.2:

Una sequenza tipica è

$$\{\mu_r, \mu_e, \mu_{oc}, \mu_{ic}\} = \{1, 2, \frac{1}{2}, -\frac{1}{2}\}$$

e in figura 2.2 è riportato un esempio di punti corrispondenti a questi valori di μ supponendo che il simplesso corrente abbia come vertici i punti x_1, x_2 e x_3.
Se il nuovo punto soddisfa $f(x(\mu)) \leq f(x_{n+1})$ si sostituisce nel simplesso corrente x_{n+1} con $x(\mu)$. Se $f(x(\mu)) > f(x_n)$ si cerca di determinare un nuovo punto riflettendo gli altri vertici, oppure facendo contrarre il simplesso.
Un esempio di algoritmo è il seguente:

Algoritmo di Nelder e Mead

Dati: S, $-1 \leq \mu_{ic} < 0 < \mu_{oc} < \mu_r < \mu_e$.

Passo 1: Calcola f nei vertici di S e ordinali in base al valore della funzione obiettivo in modo da avere

$$f(x_1) \leq \ldots \leq f(x_{n+1}).$$

Passo 2: Se $f(x_{n+1}) - f(x_1) \leq \tau$ stop. Altrimenti calcola x_c secondo la formula (2.2), $x(\mu_r)$ e $f_r = f(x(\mu_r))$.

Passo 3 (Riflessione): Se $f(x_1) \leq f_r < f(x_n)$ sostituisci x_{n+1} con $x(\mu_r)$ e vai al passo 1.

Passo 4 (Espansione): Se $f_r < f(x_1)$, calcola $f_e = f(x(\mu_e))$. Se $f_e < f_r$ sostituisci x_{n+1} con $x(\mu_e)$; altrimenti sostituisci x_{n+1} con $x(\mu_r)$ e vai al passo 1.

Passo 5 (Contrazione esterna): Se $f(x_n) \leq f_r < f(x_{n+1})$, calcola $f_{oc} = f(x(\mu_{oc}))$. Se $f_{oc} < f_r$ sostituisci x_{n+1} con $x(\mu_{oc})$ e vai al passo 1; altrimenti vai al passo 7.

Passo 6 (Contrazione interna): Se $f_r \geq f(x_{n+1})$, calcola $f_{ic} = f(x(\mu_{ic}))$. Se $f_{ic} < f(x_{n+1})$ sostituisci x_{n+1} con $x(\mu_{ic})$ e vai al passo 1; altrimenti vai al passo 7.

Passo 7 (Riduzione): Per ogni $2 \leq i \leq n+1$ poni $x_i = x_1 - (x_i - 1)/2$, calcola $f(x_i)$ e vai al passo 1.

Si vede dall'algoritmo che l'idea è quella di cercare di espandere il simplesso se si trovano valori buoni della funzione obiettivo e contrarlo se non se ne trovano. Questo algoritmo in generale è una tecnica euristica, nel senso che non è possibile assicurare la convergenza globale della sequenza prodotta, salvo alcuni casi specifici. Infatti in [8] si dimostra che

si ha convergenza ad un punto stazionario per funzioni strettamente convesse con una sola variabile, mentre, nel caso di funzioni strettamente convesse con due variabili, si dimostrano risultati di convergenza più deboli. In particolare, sono noti controesempi di problemi a 2 variabili, in cui la sequenza generata dal metodo converge ad un punto che non è un punto stazionario. Una famiglia di funzioni per cui questo accade è stata introdotta da Mc Kinnon in [13] ed ha la seguente espressione:

$$f(x,y) = \begin{cases} \theta\phi|x|^\tau + y + y^2, & x \leq 0 \\ \theta|x|^\tau + y + y^2, & x \geq 0 \end{cases} \tag{2.3}$$

dove θ e ϕ sono costanti positive. Per diversi valori dei parametri θ, ϕ e τ si ha, in corrispondenza di un particolare simplesso iniziale, convergenza dell'algoritmo di Nelder e Mead al punto di coordinate $(0,0)$, che non è un punto stazionario del problema. In figura 2.3 sono disegnate le curve di livello della funzione (2.3) per i valori dei parametri $(\theta,\phi,\tau) = (6,60,2)$ e il simplesso iniziale che ha come vertici (già ordinati per valori crescenti della funzione obiettivo):

$$x_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad x_2 = \begin{pmatrix} \frac{1+\sqrt{33}}{8} \\ \frac{1-\sqrt{33}}{8} \end{pmatrix} \quad x_3 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Come si vede dalle figure 2.4 e 2.5 ad ogni passo l'algoritmo si trova ad eseguire il passo 6 contraendo internamente il simplesso corrente senza mai cambiare il miglior vertice che rimane x_1 . L'algoritmo prosegue in questo modo fino a convergere proprio sul punto x_1, che, come già detto, non è stazionario ($\nabla f(0,0) = (1 \quad 1)^T$). In particolare, una direzione di discesa in questo punto è costituita dal secondo asse coordinato. Si vede quindi come su una funzione semplice in due variabili, l'algoritmo di Nelder e Mead converge ad un punto non stazionario.

Tuttavia, questo algoritmo, sebbene non caratterizzato da proprietà teoriche di convergenza, si è rivelato in pratica molto efficiente, particolarmente per la soluzione di problemi di dimensioni non superiori alle dieci variabili. Tale metodo è perciò presente in varie librerie standard di ottimizzazione.

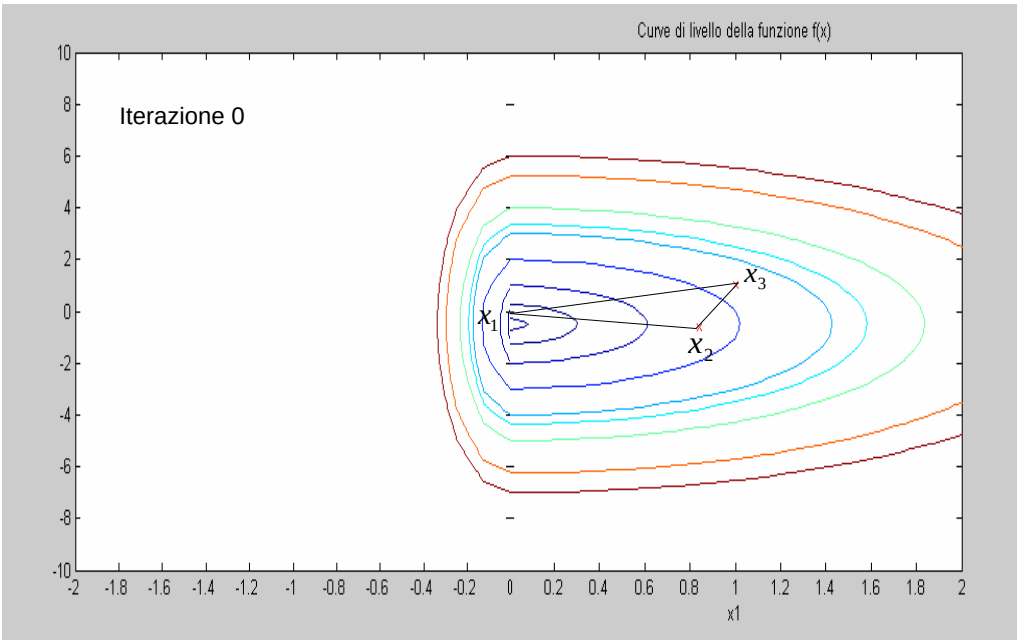

Figura 2.3:

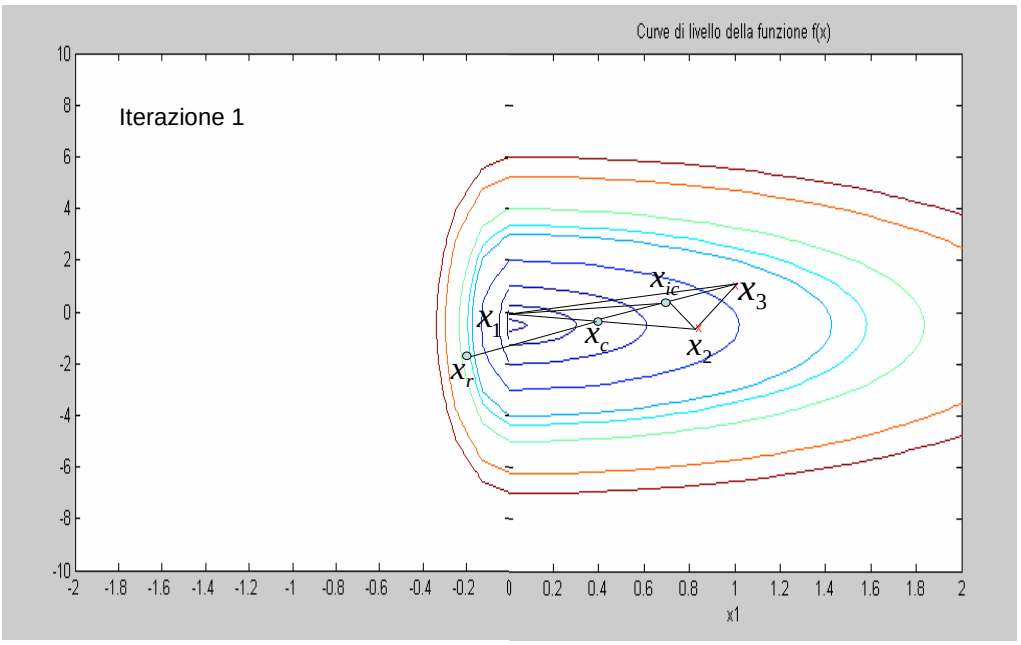

Figura 2.4:

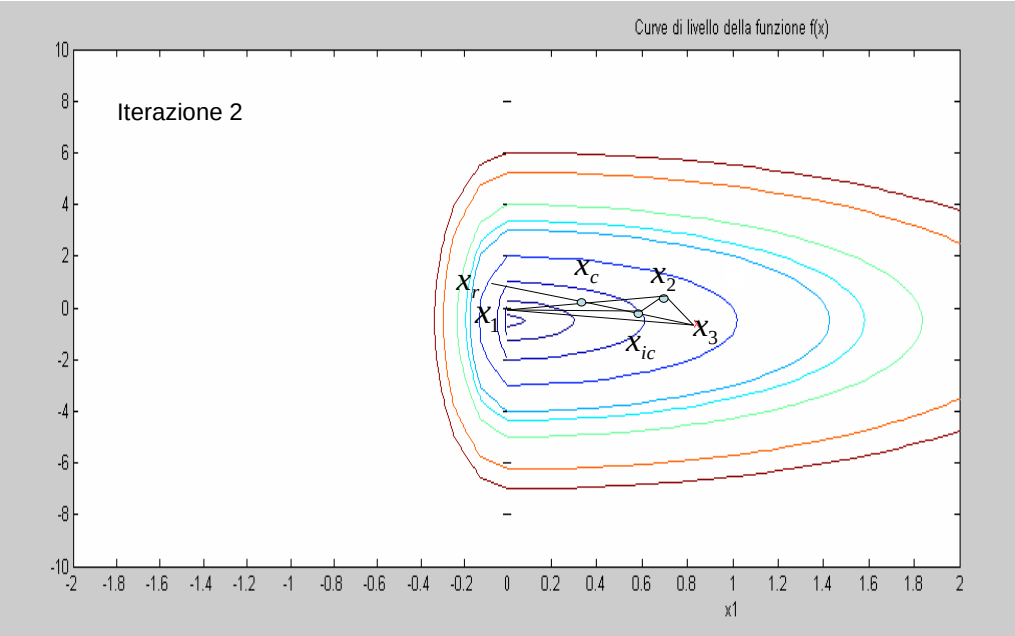
Curve di livello della funzione f(x)
Iterazione 2
x_r
x_c
x_2
x_1
x_3
x_{ic}
x1

Figura 2.5:

Capitolo 3

Metodi di ricerca diretta convergenti

3.1 Teoria generale di convergenza

Assumiamo che sia verificata la seguente assunzione:

Assunzione 3.1 *La funzione $f : \Re^n \to \Re$ è continuamente differenziabile.*

Come già detto in precedenza, questa ipotesi è spesso verificata in pratica perchè i fenomeni che si vogliono rappresentare sono per loro natura "smooth".
Il primo passo per definire un metodo di minimizzazione è quello di definire un opportuno insieme di direzioni di ricerca $\{p_k^i\}$, $i = 1, \ldots, r$, associato al punto corrente x_k. Questo insieme di direzioni deve essere tale da fornire informazione comparabile a quella che sarebbe data dal gradiente se fosse calcolabile, cioè la conoscenza dell'andamento della funzione lungo $\{p_k^i\}$, $i = 1, \ldots, r$ deve essere sufficiente a caratterizzare l'andamento locale della funzione. Questa proprietà può essere formalizzata tramite la seguente condizione ([11]):

Condizione C1

Data una sequenza di punti $\{x_k\}$, le sequenze di direzioni $\{p_k^i\}$, $i = 1, \ldots, r$ sono limitate e tali che

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0 \quad \text{se e solo se} \quad \lim_{k\to\infty} \sum_{i=1}^{r} \min\{0, \nabla f(x_k)^T p_k^i\} = 0. \tag{3.1}$$

Questa condizione equivale a richiedere che le derivate direzionali della funzione obiettivo rispetto all'insieme delle direzioni di ricerca tendano ad assumere valori non negativi se e

soltanto se l'algoritmo si sta avvicinando ad un punto stazionario. L'interesse di insiemi di direzioni che soddisfino questa condizione è reso evidente dal seguente teorema ([11]):

Proposizione 3.2 ([11]) *Sia $\{x_k\}$ una sequenza di punti limitata e siano $\{p_k^i\}$, $i = 1, \ldots, r$ delle sequenze di direzioni che soddisfano la condizione C1. Per ogni $\eta > 0$, esistono due parametri $\gamma > 0$ e $\delta > 0$ tali che, per k sufficientemente grande, se x_k soddisfa $\|\nabla f(x_k)\| \geq \eta$, allora esiste una direzione $p_k^{i_k}$, $i_k = 1, \ldots, r$, tale che*

$$f(x_k + \alpha p_k^{i_k}) \leq f(x_k) - \gamma \|\nabla f(x_k)\| \|p_k^{i_k}\|, \tag{3.2}$$

per ogni $\alpha \in (0, \delta]$.

Dim. La dimostrazione procede per assurdo. Supponiamo che la tesi sia falsa, cioè supponiamo che esista uno scalare $\eta > 0$ tale che per ogni coppia γ_t, δ_t per k sufficientemente grande esistono degli scalari α_k^i, $i = 1, \ldots, r$ tali che

$$\|\nabla f(x_k)\| \geq \eta \tag{3.3}$$

$$f(x_k + \alpha_k^i p_k^i) > f(x_k) - \gamma_t \|\nabla f(x_k)\| \|p_k^i\| \tag{3.4}$$

$$\alpha_k^i \leq \delta_t \tag{3.5}$$

per ogni $i = 1, \ldots, r$. Sappiamo per ipotesi che la sequenza $\{x_k\}$ è limitata e quindi ammette almeno un punto di accumulazione $\bar{x}$. Rinominiamo $\{x_k\}$ la sequenza convergente al punto $\bar{x}$. Inoltre, poichè la condizione (3.3) è vera per ogni coppia γ_t, δ_t possiamo considerare una sequenza γ_k, δ_k tale che

$$\{x_k\} \to \bar{x} \tag{3.6}$$

$$\gamma_k \to 0 \tag{3.7}$$

$$\delta_k \to 0 \tag{3.8}$$

$$\alpha_k^i \leq \delta_k \tag{3.9}$$

$$f(x_k + \alpha_k^i p_k^i) > f(x_k) - \gamma_k \alpha_k^i \|\nabla f(x_k)\| \|p_k^i\|. \tag{3.10}$$

Poichè il gradiente è continuo per l'Assunzione 3.1, segue che $\|\nabla f(\bar{x})\| \geq \eta$, quindi la condizione C1 implica che, per k sufficientemente grande, esiste un indice $i \in \{1, \ldots, r\}$ tale che

$$\nabla f(x_k)^T p_k^i \leq \rho < 0. \tag{3.11}$$

Le due condizioni (3.8) e (3.9) implicano $\{\alpha_k^i\} \to 0$. Poichè le direzioni p_k^i soddisfano la condizione C1, sono limitate e quindi si ha

$$\lim_{k\to\infty} \alpha_k^i \|p_k^i\| = 0 \tag{3.12}$$

per ogni $i \in \{1, \ldots, r\}$. Dalla relazione (3.10) abbiamo, applicando il teorema della media,

$$f(x_k + \alpha_k^i p_k^i) - f(x_k) = \alpha_k^i \nabla f(x_k + \theta_k^i \alpha_k^i p_k^i)^T p_k^i > -\gamma_k \alpha_k^i \|\nabla f(x_k)\| \|p_k^i\| \tag{3.13}$$

dove $\theta_k^i \in (0, 1)$ da cui segue

$$\nabla f(x_k)^T p_k^i + (\nabla f(x_k + \theta_k^i \alpha_k^i p_k^i) - \nabla f(x_k))^T p_k^i > -\gamma_k \|\nabla f(x_k)\| \|p_k^i\|. \tag{3.14}$$

Il termine a destra di quest'ultima relazione tende a zero in quanto γ_k tende a zero per la (3.7), $\nabla f(x_k)$ tende a $\|\nabla f(\bar{x})\|$ e $\|p_k^i\|$ ha un valore limitato in quanto per la condizione C1 le direzioni $\{p_k^i\}$ sono limitate. Quindi, passando al limite, e tenendo conto della (3.12), abbiamo

$$\lim_{k\to\infty} \nabla f(x_k)^T p_k^i \geq 0 \tag{3.15}$$

che contraddice la (3.11) e quindi abbiamo un assurdo. □

Il teorema precedente giustifica l'interesse teorico della condizione C1, che garantisce sempre la presenza di una "buona" direzione di discesa. L'interesse pratico di questa condizione deriva inoltre dal fatto che numerose classi di direzioni la soddisfano. In particolare, un esempio di classe di direzioni che soddisfa la condizione C1 è data da direzioni che spannano positivamente $\Re^n$, che soddisfano cioè la seguente definizione.

Definizione 3.3 *Le direzioni $\{p^1, \ldots, p^r\}$ spannano positivamente $\Re^n$ se, dato un qualunque punto $y \in \Re^n$, questo può essere espresso come combinazione a coefficienti non negativi di $\{p^1, \ldots, p^r\}$, cioè*

$$y = \sum_{i=1}^{r} \beta^i p^i, \quad \beta^i \geq 0, \ i = 1, \ldots, r.$$

Un insieme di direzioni $\{p^1, \ldots, p^r\}$ è una base positiva se spanna positivamente $\Re^n$ e nessun suo sottoinsieme proprio spanna $\Re^n$.

Si dimostra ([4]) che la cardinalità di una base positiva è compresa tra $n + 1$ e $2n$. Un esempio di base positiva è dato dalle colonne delle matrici $[I \quad -I]$ e $[I \quad -e]$, dove I

è la matrice identità (che contiene quindi gli assi coordinati) e $e = (1 \dots 1)^T$. L'interesse verso queste direzioni deriva dal seguente risultato.

Proposizione 3.4 ([11]) *Sia $\{x_k\}$ una sequenza di punti limitata e siano $\{p_k^i\}$, $i = 1, \dots, r$ delle sequenze di direzioni limitate e tali che ogni punto limite $(\bar{p}^1, \dots, \bar{p}^r)$ delle sequenze $\{p_k^i\}$, $i = 1, \dots, r$ spanna positivamente $\Re^n$. Allora $\{p_k^i\}$, $i = 1, \dots, r$ soddisfano la condizione C1.*

Dim. Per far vedere che le direzioni $\{p_k^i\}$, $i = 1, \dots, r$ soddisfano la condizione C1 dobbiamo far vedere che

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0 \quad \text{se e solo se } \lim_{k\to\infty} \sum_{i=1}^{r} \min\{0, \nabla f(x_k)^T p_k^i\} = 0. \tag{3.16}$$

Prima di tutto notiamo che, se $\lim_{k\to\infty} \|\nabla f(x_k)\| = 0$, allora, poichè la direzioni $\{p_k^i\}$, $i = 1, \dots, r$ sono limitate per ipotesi, si ha $\lim_{k\to\infty} \sum_{i=1}^{r} \min\{0, \nabla f(x_k)^T p_k^i\} = 0$.
Rimane quindi da dimostrare che se

$$\lim_{k\to\infty} \sum_{i=1}^{r} \min\{0, \nabla f(x_k)^T p_k^i\} = 0, \tag{3.17}$$

allora si ha

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0. \tag{3.18}$$

Supponiamo per assurdo che la (3.18) non sia verificata. Poichè la sequenza $\{x_k\}$ è limitata, ammette almeno un punto di accumulazione. Possiamo quindi scegliere un insieme di indici K_1 tale che

$$\lim_{k\to\infty, k\in K_1} x_k = \bar{x}, \tag{3.19}$$

$$\|\nabla f(\bar{x})\| \geq \eta > 0. \tag{3.20}$$

Poichè per ipotesi le direzioni $\{p_k^i\}$, $i = 1, \dots, r$ sono limitate, si può scegliere un insieme di indici $K_2 \subseteq K_1$ tale che

$$\lim_{k\to\infty, k\in K_2} p_k^i = \bar{p}^i, \quad i = 1, \dots, r \tag{3.21}$$

dove $\{\bar{p}^1, \ldots, \bar{p}^r\}$ spannano positivamente $\Re^n$. Si può quindi scrivere

$$-\nabla f(\bar{x}) = \sum_{i=1}^{r} \beta^i \bar{p}^i, \tag{3.22}$$

con $\beta^i \geq 0$, $i = 1, \ldots, r$. Dalla (3.20) e dalla (3.22) segue

$$-\eta^2 \geq \sum_{i=1}^{r} \beta^i \nabla f(\bar{x})^T \bar{p}^i. \tag{3.23}$$

Dalla continuità del gradiente e dalla (3.23) si ha

$$\lim_{k\to\infty, k\in K_2} \sum_{i=1}^{r} \min\{0, \nabla f(x_k)^T p_k^i\} = \sum_{i=1}^{r} \min\{0, \nabla f(\bar{x})^T \bar{p}^i\} < 0, \tag{3.24}$$

che contraddice la (3.17). □

Una volta introdotta la condizione C1 e definito degli insieme di direzioni che le soddisfano possiamo enunciare il teorema di convergenza che specifica un insieme di condizioni che, se soddisfatte, garantiscono la convergenza globale ad un punto stazionario. Per far questo è necessaria la seguente assunzione che è standard nei problemi di ottimizzazione non vincolati.

Assunzione 3.5 *Dato un punto $x_0 \in \Re^n$, la curva di livello $\mathcal{L}_0 = \{x \in \Re^n : \ f(x) \leq f(x_0)\}$ della funzione obiettivo è compatta.*

Proposizione 3.6 ([11]) *Supponiamo che siano verificate le assunzioni 3.1 e 3.5. Sia $\{x_k\}$ una sequenza di punti e siano $\{p_k^i\}$, $i = 1, \ldots, r$ delle sequenze di direzioni. Supponiamo verificate le seguenti condizioni:*

(a) $f(x_{k+1}) \leq f(x_k)$,

(b) $\{p_k^i\}$, $i = 1, \ldots, r$ *soddisfano la condizione C1,*

(c) esistono delle sequenze di punti $\{y_k^i\}$ e delle sequenze di scalari positivi $\{\xi_k^i\}$ tali che:

$$f(y_k^i + \xi_k^i p_k^i) \geq f(y_k^i) - o(\xi_k^i) \tag{3.25}$$

$$\lim_{k\to\infty} \xi_k^i = 0 \tag{3.26}$$

$$\lim_{k\to\infty} \|x_k - y_k^i\| = 0. \tag{3.27}$$

Allora

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0 \tag{3.28}$$

Dim. Dalla condizione (a) segue che la sequenza $\{x_k\}$ appartiene all'insieme $\mathcal{L}_0$ che è compatto per ipotesi. Per il teorema di Weierstrass la funzione f ammette un minimo globale x^* su $\mathcal{L}_0$. Abbiamo quindi una sequenza $\{f(x_k)\}$ non crescente e limitata inferiormente, e quindi convergente. Di conseguenza la sequenza $\{x_k\}$ ammette almeno un punto di accumulazione $\bar{x}$. Inoltre le sequenze direzioni $\{p_k^i\}$, $i = 1, \ldots, r$ sono limitate poichè soddisfano la condizione C1 e di conseguenza ammettono punti di accumulazione. Possiamo allora selezionare un insieme di indici K tale che

$$\lim_{k\to\infty, k\in K} x_k = \bar{x} \tag{3.29}$$

$$\lim_{k\to\infty, k\in K} p_k^i = \bar{p}^i, \quad i = 1, \ldots, r. \tag{3.30}$$

Inoltre la (3.29) e la (3.27) implicano

$$\lim_{k\to\infty, k\in K} y_k^i = \bar{x}, \quad i = 1, \ldots, r. \tag{3.31}$$

Dalla (3.25), applicando il teorema della media, si ha

$$f(y_k^i + \xi_k^i p_k^i) - f(y_k^i) = \xi_k^i \nabla f(y_k^i + \theta \xi_k^i p_k^i)^T p_k^i \geq -o(\xi_k^i), \tag{3.32}$$

da cui segue

$$\nabla f(y_k^i + \theta \xi_k^i p_k^i)^T p_k^i \geq -\frac{o(\xi_k^i)}{\xi_k^i}. \tag{3.33}$$

Passando al limite per $k \to \infty$ la continuità del gradiente, la (3.26), la (3.32), e la (3.33) implicano

$$\nabla f(\bar{x})^T \bar{p}^i \geq 0. \tag{3.34}$$

Allora, poichè le direzioni $\{p_k^i\}$, $i = 1, \ldots, r$ soddisfano la condizione C1, si ha

$$\nabla f(\bar{x}) = 0, \tag{3.35}$$

cioè $\bar{x}$ è un punto stazionario. Poichè $\bar{x}$ è un qualsiasi punto di accumulazione della sequenza $\{x_k\}$, si ha

$$\lim_{k\to\infty} \nabla f(x_k) = 0. \tag{3.36}$$

□

Definito il teorema generale di convergenza, si possono considerare tre classi di metodi che soddisfano le ipotesi della Proposizione 3.6: metodi di tipo pattern search, metodi di tipo line search e metodi di tipo pattern-line.

3.2 Metodi di tipo Pattern Search

I metodi di tipo pattern search procedono valutando la funzione obiettivo su punti appartenenti ad una griglia. Questa griglia è indipendente dalla funzione obiettivo ed è definita da un insieme di direzioni e da uno scalare Δ che indica la distanza tra i punti della griglia. L'idea è quella di esplorare la griglia centrata nel punto corrente alla ricerca di un punto con funzione obiettivo migliore del punto corrente.
Appartengono alla classe di algoritmi pattern search il metodo delle coordinate, il metodo di Hooke and Jeeves introdotto in [7] e il metodo di ricerca multidirezionale ([17]). Nella sua forma più generale l'algoritmo pattern search è il seguente ([18]):

Algoritmo Pattern Search

Dati: $x_0 \in \Re^n$, $\Delta_0 > 0$, $D \in \Re^{n\times p}$, $p > 2n$, $\tau > 1$, $m_k^+ \geq 1$, $m_k^- \leq -1$

Passo 0: Poni $k = 0$.

Passo 1: Determina uno spostamento $s_k = \Delta_k d_k$ tramite la procedura Mossa Esplorativa(Δ_k, D_k).

Passo 2: Se $f(x_k + s_k) < f(x_k)$ poni $x_{k+1} = x_k + s_k$, altrimenti poni $x_{k+1} = x_k$.

Passo 3: Aggiorna Δ_k e D_k, poni $k = k + 1$ e vai al Passo 1.

Per definire un determinato metodo pattern, occorre specificare la matrice delle direzioni D_k, la procedura di campionamento Mossa Esplorativa utilizzata per produrre lo spostamento s_k, e le regole di aggiornamento di D_k e del passo Δ_k, che rappresentano rispettivamente l'insieme di direzioni e il passo che definiscono la griglia all'iterazione corrente. In particolare, le colonne della matrice D_k rappresentano le direzioni di ricerca lungo le quali avviene il campionamento della funzione obiettivo. Affinchè questo campionamento sia significativo, tali direzioni devono soddisfare opportuni requisiti. In particolare, la matrice D_k è definita tramite una *matrice di base* $B \in \Re^{n\times n}$ e una *matrice generatrice* intera $C_k \in \mathcal{Z}^{n\times p}$, $p > 2n$. La matrice di base B è una qualunque matrice non singolare $n \times n$. La matrice generatrice deve invece avere una struttura del seguente tipo

$$C_k = [M_k \ -M_k \ L_k] = [\Gamma_k \ L_k]$$

in cui $L_k \in \mathcal{Z}^{n\times(2n-p)}$ contiene almeno una colonna costituita da elementi nulli, $M_k \in M$, dove $M \subset \mathcal{Z}^{n\times n}$ è un insieme di matrici non singolari con elementi interi. La matrice delle direzioni è definita come

$$D_k = BC_k. \tag{3.37}$$

La procedura Exploratory Moves deve soddisfare due condizioni:

(a) Lo spostamento s_k deve essere un elemento di $\Delta_k D_k$;

(b) Se $\min\{f(x_k + y),\ y \in \Delta_k B\Gamma_k\} < f(x_k)$, allora $f(x_k + s_k) < f(x_k)$.

In pratica la mossa esplorativa deve produrre uno spostamento di lunghezza Δ_k lungo una delle direzioni $d \in D_k$ e deve garantire che se uno spostamento Δ_k lungo una delle $2n$ direzioni appartenenti a $B\Gamma_k$ produce un decremento della funzione obiettivo, allora la mossa deve avere successo, cioè deve produrre un decremento della funzione obiettivo. Per quanto riguarda l'aggiornamento di Δ_k, tipicamente avviene secondo la regola

$$\Delta_{k+1} = \begin{cases} \theta\Delta_k & \text{se } s_k = 0 \\ \lambda\Delta_k & \text{altrimenti} \end{cases} \tag{3.38}$$

dove $\theta = \tau^{m_k^-}$, $\lambda = \tau^{m_k^+}$, essendo $\tau > 1$ un numero razionale, $m_k^- < 0$ e $m_k^+ \geq 0$ interi. Questa regola di aggiornamento fa sì che $\theta \in (0, 1)$ e $\lambda \geq 1$. Quindi, se un'iterazione ha successo, riesce cioè a produrre un nuovo punto migliore del precedente, è possibile

aumentare il passo (o lasciarlo invariato), mentre, se un'iterazione fallisce, il passo viene diminuito.
Con queste ipotesi sulla mossa esplorativa e con questa regola di aggiornamento del parametro Δ_k si dimostra il seguente risultato:

Proposizione 3.7 ([18]) *Supponiamo verificata l'Assunzione 3.5. Allora, se la procedura Mossa Esplorativa verifica le proprietà (a) e (b) e se il passo Δ_k viene aggiornato tramite la regola (3.38), esiste un insieme infinito di indici K tali che*

$$\lim_{k\to\infty, k\in K} \Delta_k = 0. \tag{3.39}$$

Dim. Osserviamo innanzi tutto che, ad ogni iterazione k, per come è definita la regola di aggiornamento (3.38) del passo corrente Δ_k, esiste sempre un intero r_k tale che

$$\Delta_k = \tau^{r_k}\Delta_0. \tag{3.40}$$

Inoltre, poichè l'algoritmo produce una sequenza di punti $\{x_k\}$ tali che $f(x_{k+1}) \leq f(x_k)$, abbiamo che la sequenza di punti appartiene all'insieme $\mathcal{L}_0$, che è compatto. Questo implica che esiste un $\Delta = \max_{x,y\in\mathcal{L}_0} \|x - y\|$ tale che $\Delta_k \leq \Delta$, e quindi, scelto un intero r_u tale che $\Delta_0\tau^{r_u} \geq \Delta$, avremo

$$\Delta_k \leq \Delta_0\tau^{r_u}. \tag{3.41}$$

Ora supponiamo per assurdo che la (3.39) non sia verificata. Questo implica che esiste un intero r_l tale che

$$\Delta_k \geq \Delta_0\tau^{r_l} > 0. \tag{3.42}$$

Quindi le equazioni (3.40), (3.41) e (3.42) implicano che per ogni $k \geq 0$ l'intero r_k e di conseguenza il parametro Δ_k possono assumere solo un numero finito di valori. Inoltre per ogni k il punto x_{k+1} può essere scritto come $x_k + \Delta_k d_k$, con $d_k \in D_k$ (notiamo che se $x_{k+1} = x_k$ questa relazione è ancora valida perchè la matrice L_k contiene la colonna

nulla). Avremo quindi, dato un qualunque intero J:

$$
\begin{aligned}
x_J &= x_0 + \sum_{k=1}^{J-1} \Delta_k d_k = x_0 + \Delta_0 B \sum_{k=1}^{J-1} \tau^{r_k} c_k = \\
&= x_0 + \frac{p^{r_l}}{q^{r_u}} \Delta_0 B \sum_{k=1}^{J-1} p^{r_k - r_l} q^{r_u - r_k} c_k = x_0 + \frac{p^{r_l}}{q^{r_u}} \Delta_0 B \sum_{k=1}^{J-1} z_k,
\end{aligned} \tag{3.43}
$$

dove p e q sono due interi primi tra di loro tali che $\tau = \frac{p}{q}$ e dove $z_k \in \mathcal{Z}^n$. Quindi, per ogni k, il punto corrente appartiene al reticolo intero traslato centrato in x_0 e definito dalle colonne di $\frac{p^{r_l}}{q^{r_u}} \Delta_0 B$. L'intersezione di questo reticolo con l'insieme compatto $\mathcal{L}_0$ è un insieme di punti finiti. Quindi la (3.43) implica che deve esistere almeno un punto x_* nel reticolo tale che $x_k = x_*$ un numero infinito di volte. Ma le istruzioni dell'algoritmo Pattern Search implicano che uno stesso punto non può essere visitato infinite volte perchè si accetta uno spostamento s_k se e solo se $f(x_k + s_k) < f(x_k)$. Questo vuol dire che non posso tornare su un punto già visitato in iterazioni precedenti. Quello che invece può succedere è che $x_{k+1} = x_k$. Quindi esiste una sottosequenza $\{x_k\}_K$ tale che, per ogni $k \in K$, $x_{k+1} = x_k$, $x_k = x_*$ e $s_k = 0$. Ma la regola di aggiornamento di Δ_k implica che ogni volta che non si riesce a produrre un decremento della funzione obiettivo si pone $\Delta_{k+1} = \theta \Delta_k$ e quindi, poichè $\theta \in (0, 1)$,

$$\lim_{k \to \infty, k \in K} \Delta_k = 0.$$

□

Imponendo delle ipotesi più restrittive sulla procedura Mossa Esplorativa e sull'aggiornamento del parametro Δ_k si può dimostrare un risultato più forte. In particolare, si può dimostrare il seguente risultato:

Proposizione 3.8 ([18]) *Supponiamo verificata l'Assunzione 3.5. Allora, se la procedura Mossa Esplorativa verifica le seguenti proprietà*

(a) lo spostamento s_k deve essere un elemento di $\Delta_k D_k$;

(b1) se $\min\{f(x_k + y),\ y \in \Delta_k B \Gamma_k\} < f(x_k)$, *allora* $f(x_k + s_k) \leq \min\{f(x_k + y),\ y \in \Delta_k B \Gamma_k\}$.

e se la regola di aggiornamento di Δ_k è la seguente

$$\begin{aligned} \Delta_{k+1} &= \theta\Delta_k \qquad \text{se } s_k = 0 \\ \Delta_{k+1} &= \Delta_k \qquad \text{altrimenti} \end{aligned} \tag{3.44}$$

si ha

$$\lim_{k\to\infty} \Delta_k = 0. \tag{3.45}$$

La prossimo proposizione stabilisce le proprietà di convergenza dell'algoritmo Pattern Search nella sua versione più debole. Questo risultato è stato dimostrato in [18], ma qui è riportata la dimostrazione descritta in [10] che sfrutta la Proposizione 3.6.

Proposizione 3.9 ([18]) *Sia $\{x_k\}$ la successione prodotta dall'algoritmo Pattern Search. Supponiamo che le direzioni p_k^i che compongono $B\Gamma_k$, $i = 1, \ldots, 2n$ soddisfino la Condizione C1. Allora, se la procedura Mossa Esplorativa verifica le proprietà (a) e (b) e se il passo Δ_k viene aggiornato tramite la regola (3.38), esiste un insieme infinito di indici K tali che*

$$\lim_{k\to\infty, k\in K} \|\nabla f(x_k)\| = 0. \tag{3.46}$$

Dim.([10]) Per dimostrare il risultato facciamo vedere che esiste una sottosequenza $\{x_k\}_K$ tale che le condizioni (a), (b) e (c) della Proposizione 3.6 sono soddisfatte. Le istruzioni dell'algoritmo implicano $f(x_{k+1}) \leq f(x_k)$ per ogni k e quindi la condizione (a) è verificata. Inoltre la condizione (b) è verificata per ipotesi. Per quel che riguarda la condizione (c), la Proposizione 3.7 implica che esiste un insieme infinito di indici K tale che

$$\lim_{k\to\infty, k\in K} \Delta_k = 0.$$

Questo implica che esiste una sottosequenza $\{\Delta_k\}_{K_1}$, con $K_1 \subseteq K$ tale che per ogni $k \in K_1$ $\Delta_k < \Delta_{k-1}$. Questo implica che per ogni $k \in K_1$, l'iterazione precedente è stata fallimentare, e quindi, dall'aggiornamento di Δ_k, si ha che, per ogni $k \in K_1$,

$$f(x_k + \frac{\Delta_k}{\theta} p_k^i) \geq f(x_k), \quad i = 1, \ldots, 2n.$$

Quindi, scegliendo

$$\xi_k^i = \frac{\Delta_k}{\theta}, \quad \forall k \in K, \tag{3.47}$$

$$o(\xi_k^i) = 0, \quad \forall k \in K \tag{3.48}$$

$$y_k^i = x_k, \quad \forall k \in K \tag{3.49}$$

abbiamo che la condizione (c) è verificata. □

Sotto ipotesi più restrittive si può dimostrare il seguente teorema di convergenza più forte

Proposizione 3.10 ([18]) *Sia $\{x_k\}$ la successione prodotta dall'algoritmo Pattern Search. Supponiamo che le direzioni p_k^i che compongono $B\Gamma_k$, $i = 1, \ldots, 2n$ soddisfino la Condizione C1. Allora, se la procedura Mossa Esplorativa verifica le proprietà (a) e (b1) e se il passo Δ_k viene aggiornato tramite la regola (3.44),*

$$\lim_{k \to \infty} \|\nabla f(x_k)\| = 0. \tag{3.50}$$

Per definire un metodo pattern search specifico bisogna definire la procedura Mossa Esplorativa, la matrice delle direzioni D_k e i parametri dell'aggiornamento del passo Δ_k. Consideriamo come esempi di metodi Pattern Search due algoritmi storicamente piuttosto importanti: il metodo delle coordinate e l'algoritmo di Hooke e Jeeves.

3.2.1 Metodo delle coordinate

Il metodo delle coordinate è il più semplice e il più ovvio dei metodi Pattern. Il metodo è descritto in [3], ma è molto antecedente (**un vero riferimento?Io non l'ho trovato**). L'idea è quella di spostarsi lungo gli assi coordinati, cambiando (aumentando o diminuendo) il valore di una variabile per volta di una quantità fissa (passo Δ_k). Se così facendo si ottiene un decremento della funzione obiettivo ci si sposta sul punto ottenuto e si continua con la variabile successiva. Se non si riesce a produrre un decremento lungo nessun asse coordinato si dimezza il passo e si ricomincia. In figura 3.1 sono riportati tutti i possibili spostamenti a partire dal punto corrente x_k in $\Re^2$. I punti x_k^i e $x_k^{i'}$ sono i punti in cui viene valutata la funzione obiettivo durante l'iterazione. La linea è tratteggiata quando la funzione obiettivo non diminuisce nel passare da un punto all'altro, mentre è piena se si ha un decremento. Nel caso peggiore (l'ultimo nella figura 3.1) la funzione viene valutata in $2n$ punti senza riuscire a produrre un nuovo punto.

Questo semplice algoritmo può essere rappresentato come un metodo pattern ([18]). In

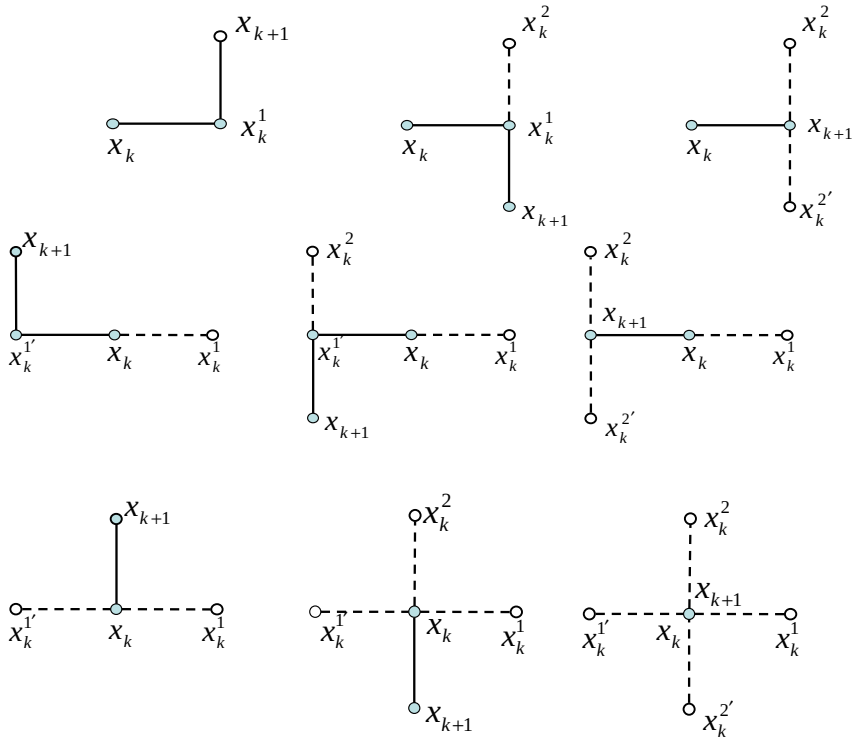

Figura 3.1: Tutti i possibili passi effettuati da un iterazione del metodo delle coordinate in $\Re^2$

particolare, la matrice delle direzioni $D_k = BC_k$ è ottenuta ponendo $B = I$, dove I è la matrice identità (oppure si può utilizzare un'opportuna matrice di scalatura). La matrice delle direzioni C_k è costante in tutte le iterazioni e deve contenere nelle sue colonne tutte le possibili combinazioni di $\{-1, 0, 1\}$. Quindi, la matrice C ha 3^n colonne. In particolare, le colonne di C contengono sia I che $-I$ e una colonna tutta nulla. Definiamo quindi $M_k = M = I$, mentre $L_k = L$ è costituita dalle rimanenti $3^n - 2n$ colonne di C. Poichè C è costante durante tutte le iterazioni non è necessario definire una regola di aggiornamento per la matrice delle direzioni.
Nel caso $n = 2$ si ha

$$C = \begin{bmatrix} 1 & 0 & -1 & 0 & 1 & 1 & -1 & -1 & 0 \\ 0 & 1 & 0 & -1 & 1 & -1 & -1 & 1 & 0 \end{bmatrix}.$$

Quindi, per $n = 2$, tutti i possibili punti di prova definiti dalla matrice $D = BC$, per un passo fissato Δ_k, possono essere visti in figura 3.2 La procedura Mossa Esplorativa in

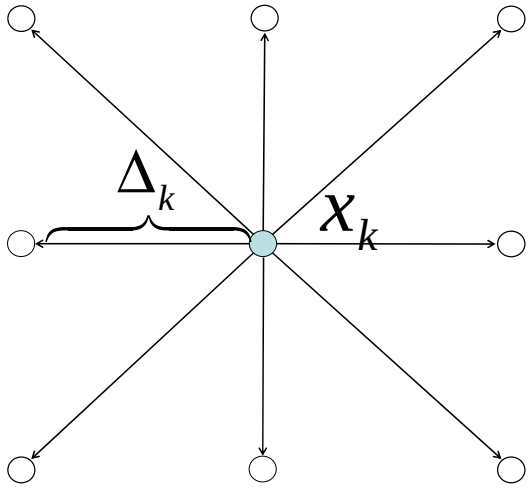

Figura 3.2: La griglia per il metodo delle coordinate in $\Re^2$ con passo Δ_k

questo caso particolare è la seguente, dove e_i rappresenta l'i-esimo asse coordinato:

Mossa Esplorativa per il metodo delle coordinate

Dati: $x_k \in \Re^n$, $\Delta_k > 0$, $B \in \Re^{n \times n}$

Passo 0: Poni $s_k = 0$ e $f_{min} = f(x_k)$. Poni $i = 1$.

Passo 1: Poni $s_k^i = s_k + \Delta_k B e_i$ e $x_k^i = x_k + s_k^i$. Calcola $f(x_k^i)$.

Passo 2: Se $f(x_k^i) < f_{min}$ poni $f_{min} = f(x_k^i)$, $s_k = s_k^i$ e vai al passo 5.

Passo 3: Poni $s_k^i = s_k - \Delta_k B e_i$ e $x_k^i = x_k + s_k^i$. Calcola $f(x_k^i)$.

Passo 4: Se $f(x_k^i) < f_{min}$ poni $f_{min} = f(x_k^i)$, $s_k = s_k^i$ e vai al passo 5.

Passo 5: Se $i = n$ stop. Altrimenti poni $i = i + 1$ e vai al Passo 1.

Gli spostamenti dal punto corrente sono fatti in sequenza, nel senso che la selezione del successivo punto di prova dipende dal fallimento o dal successo del tentativo precedente. Quindi, benchè siano 3^n i punti della griglia corrente (corrispondenti alle colonne della matrice C_k), se ne visitano al minimo n (nel caso in cui si ha successo lungo tutti gli assi coordinati presi con il segno positivo) e al massimo $2n$ (nel caso in cui si ha fallimento lungo tutti gli assi coordinati sia con il segno positivo che con il segno negativo). La procedura così definita soddisfa le ipotesi (a) della sezione precedente, in quanto produce uno spostamento $s_k = \Delta_k c_k$, con $c_k \in C$. Inoltre, anche se non è detto che questo tipo di Mossa esplorativa produca lo spostamento migliore (cioè quello che porta al maggior decremento della funzione obiettivo), nel caso peggiore di fallimento valuta la funzione obiettivo nei $2n$ punti $x_k \pm \Delta_k e_i$, $i = 1, \ldots, n$, che corrispondono ai punti $\{x_k + y,\ y \in \Delta_k B \Gamma_k\}$, e quindi verifica anche la proprietà (b).
L'ultimo aspetto da specificare è la regola di aggiornamento del parametro Δ_k, che avviene secondo la legge (3.44) con $\theta = 0.5$. Abbiamo quindi che il metodo delle coordinate rientra nella classe dei metodi pattern search e soddisfa le condizioni della Proposizione 3.9.

3.2.2 Algoritmo di Hooke e Jeeves

L'algoritmo di Hooke e Jeeves introdotto in [7] è una variante del metodo delle coordinate che incorpora un passo "pattern" nel tentativo di velocizzare la convergenza dell'algoritmo sfruttando le informazioni ottenute durante l'ultima iterazione in caso di successo. In particolare, se l'ultima iterazione ha prodotto un nuovo punto effettuando una ricerca lungo gli assi coordinati, allora l'iterazione successiva inizia effettuando un'iterazione del metodo delle coordinate a partire dal punto $x_k + (x_k - x_{k-1})$ (passo "pattern"). L'idea è quella di tentare di sfruttare il più possibile la direzione $x_k - x_{k-1}$ che è una direzione di discesa (poichè $x_k \neq x_{k-1}$ e $f(x_k) < f(x_{k-1})$).

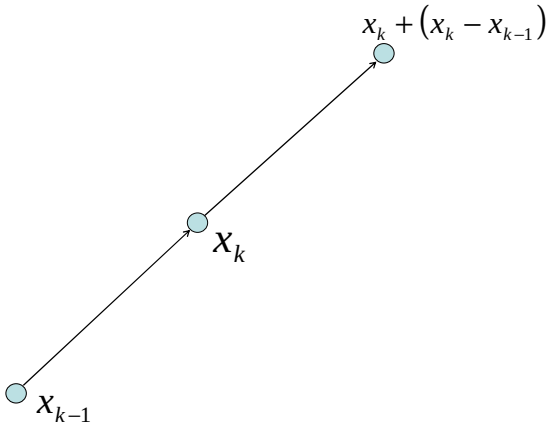

Figura 3.3: Passo "pattern" in $\Re^2$

In figura 3.3 è riportato un esempio di passo "pattern" in $\Re^2$. In particolare, la funzione viene calcolata nel punto $x_k+(x_k-x_{k-1})$ e, anche se $f(x_k+(x_k-x_{k-1})) \geq f(x_k)$, questo punto viene temporaneamente accettato e si effettua un'iterazione del metodo delle coordinate a partire da questo punto (vedi figura 3.4). Se questa iterazione ha successo, allora il punto prodotto diventa il nuovo punto x_{k+1}, altrimenti, se $f(x_k + (x_k - x_{k-1}) + s_k) \geq f(x_k)$, si prende come punto corrente x_k. In pratica, se il passo "pattern" fallisce, un'iterazione dell'algoritmo di Hooke e Jevees si riduce ad un'iterazione del metodo delle coordinate. Anche nel caso in cui l'iterazione precedente non ha avuto successo ($x_k = x_{k-1}$), allora l'iterazione si riduce ad un'iterazione del metodo delle coordinate intorno al punto x_k con passo ridotto di un fattore $\theta \in (0, 1)$ ($\Delta_k = \theta\Delta_{k-1}$). Chiaramente il passo "pattern" non viene eseguito quando $k = 0$.

Anche questo algoritmo rientra nella classe di algoritmi Pattern Search ([18]). In particolare, la matrice delle direzioni $D_k = BC_k$ è ottenuta ponendo $B = I$ anche in questo

caso. La matrice C_k questa volta varia al variare delle iterazioni per rappresentare il passo "pattern". In particolare, C_k è costituita da $2 \cdot 3^n$ colonne, dove le prime 3^n sono fisse e sono identiche alla matrice C del metodo delle coordinate, mentre le seconde 3^n variano e servono a rappresentare gli effetti del passo "pattern" rispetto al punto x_k. Anche in questo caso si pone $M = I$, mentre L_k varia al variare delle iterazioni.
Anche in questo caso sono soddisfatte le ipotesi della Proposizione 3.9.

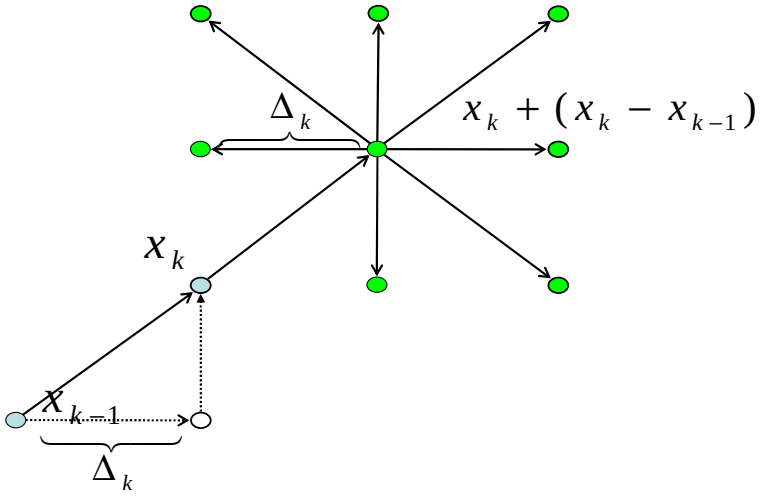

Figura 3.4: Tutti i punti che può visitare l'iterazione k-esima dell'algoritmo di Hooke e Jeeves se l'iterazione $(k-1)$-esima ha avuto successo

3.3 Metodi di tipo linesearch

I metodi di tipo line search che utilizzano le derivate sono descritti da un'iterazione del tipo

$$x_{k+1} = x_k + \alpha_k d_k$$

in cui la direzione $d_k \in R^n$ soddisfa una *condizione d'angolo*

$$\nabla f(x_k)^T d_k \leq -c_1 \|d_k\| \|\nabla f(x_k)\|$$

con $c_1 > 0$, e il passo $\alpha_k \in R$ è determinato con un tecnica di ricerca unidimensionale tale da assicurare il soddisfacimento delle seguenti condizioni

$$f(x_{k+1}) \leq f(x_k)$$

$$\lim_{k \to \infty} \frac{\nabla f(x_k)^T d_k}{\|d_k\|} = 0.$$

Le tecniche di ricerca unidimensionale usualmente adottate nel contesto dei metodi con derivate sono basate su condizioni di sufficiente decremento della funzione obiettivo tipo Armijo della forma

$$f(x_k + \alpha d_k) \leq f(x_k) - \gamma \alpha_k \nabla f(x_k)^T d_k \tag{3.51}$$

in cui compare esplicitamente la derivata direzionale.
Per trasferire la strategia di tipo line search nell'ambito di metodi senza derivate occorre quindi definire opportune tecniche di ricerca unidimensionale che non richiedano la conoscenza del gradiente della funzione obiettivo. In letteratura ([6],[9],[5]) sono state proposte tecniche basate su una condizione di sufficiente decremento del tipo

$$f(x_k \pm \alpha d_k) \leq f(x_k) - \gamma \alpha_k^2 \|d_k\|^2, \tag{3.52}$$

che sostituisce la (3.51). Osserviamo che il test di sufficiente decremento deve essere effettuato lungo le direzioni $\pm d_k$, perchè senza informazioni sulle derivate non è possibile stabilire a priori se una direzione d_k è di discesa.
I vari metodi di tipo line search si differenziano sostanzialmente per la regola di accettazione del passo lungo la direzione di ricerca. Riassumendo, tali metodi presentano i seguenti aspetti:

- sono basati su condizioni di "sufficiente decremento della funzione obiettivo;
- la proprietà di convergenza globale può essere assicurata senza imporre ipotesi particolarmente restrittive sul passo e sulle direzioni di ricerca (in tal senso quindi sono più flessibili rispetto ai metodi di tipo pattern).

Un esempio di algoritmo Line Search è il seguente.

Algoritmo Line Search

Dati: $x_0 \in \Re^n$, α^0

Passo 0: Poni $k = 0$.

Passo 1: Determina uno spostamento α_k lungo la direzione d_k tramite la procedura Line Search$(x_k, \alpha^0, d_k, \beta_k)$.

Passo 2: Poni $x_{k+1} = x_k + \alpha_k d_k$.

Passo 3: Poni $k = k + 1$ e vai al Passo 1.

Line Search$(x_k, \alpha_0, d_k, \beta_k)$

Dati: $\gamma > 0$, $\delta \in (0,1)$.

Passo 0: Poni $\alpha = \alpha_0$.

Passo 1: Se esiste $u \in \{-1, 1\}$ tale che

$$f(x_k + u\alpha d_k) \leq f(x_k) - \gamma\alpha^2\|d_k\|^2, \tag{3.53}$$

allora poni $\alpha_k = \min\{\delta^{-j}\alpha,\ j \in 0, 1, \ldots\}$ tale che

$$\begin{aligned} f(x_k + u\alpha_k d_k) &\leq f(x_k) - \gamma\left(\alpha_k\right)^2\|d_k\|^2, \\ f(x_k + u\frac{\alpha_k}{\delta}d_k) &\geq \max\left[f(x_k + u\alpha_k d_k), f(x_k) - \gamma\left(\frac{\alpha_k}{\delta}\right)^2\|d_k\|^2\right] \end{aligned} \tag{3.54}$$

ed esci.

Passo 2: Calcola $\alpha = \max\{\delta^j\alpha,\ j = 0, 1, \ldots\}$ tale che

- o esiste $u \in \{-1, 1\}$ tale che

$$f(x_k + u\alpha d_k) \leq f(x_k) - \gamma\alpha^2\|d_k\|^2 \tag{3.55}$$

  In questo caso poni $\alpha_k = u\alpha$ ed esci
- oppure $\alpha\|d_k\| \leq \beta_k$. In questo caso poni $\alpha_k = 0$ ed esci.

Le direzioni d_k devono essere scelte in modo tale che esista uno scalare $\eta > 0$ tale che, per ogni k, si abbia

$$|\det D_k| \geq \eta \tag{3.56}$$

dove D_k è la matrice $n \times n$ le cui colonne sono i vettori $d_{k+j}/\|d_{k+j}\|$, $j = 0, 1, \ldots, n-1$. La scelta del parametro β_k deve invece soddisfare la seguente condizione

$$\lim_{k\to\infty} \beta_k = 0 \tag{3.57}$$

Si può dimostrare che la procedura Line Search è ben definita ([9],[5]):

Proposizione 3.11 *Supponiamo verificata l'Assunzione 3.5. Allora la procedura Line Search è ben definita.*

Dim. Supponiamo per assurdo che non sia ben definita. Abbiamo due possibilità :

- o esiste $u \in \{-1, 1\}$ tale che $f(x_k + u\alpha_0 d_k) \leq f(x_k) - \gamma\alpha_0^2\|d_k\|^2$,
- o, per ogni $u \in \{-1, 1\}$, $f(x_k + u\alpha_0 d_k) > f(x_k) - \gamma\alpha_0^2\|d_k\|^2$

Nel primo caso la nostra ipotesi assurda equivale a supporre che per ogni $j = 0, 1, \ldots$, si ha

$$f(x_k + u\alpha_0\delta^{-j}d_k) \leq f(x_k) - \gamma\left(\frac{\alpha_0}{\delta^j}\right)^2\|d_k\|^2, \tag{3.58}$$

$$f(x_k + u\frac{\alpha_0}{\delta^{j+1}}d_k) < \max\left[f(x_k + u\frac{\alpha_0}{\delta^j}d), f(x_k) - \gamma\left(\frac{\alpha_0}{\delta^{j+1}}\right)^2\|d_k\|^2\right]. \tag{3.59}$$

Dall'Assunzione 3.5 abbiamo che f ammette un minimo globale su $\mathcal{L}_0$ e quindi è inferiormente limitata. Ma, al tendere di j all'infinito, le condizioni (3.58) e (3.59) implicano che la funzione è illimitata inferiormente e questo è assurdo.
Nel secondo caso, invece, l'ipotesi che la procedura non sia ben definita si traduce nel supporre che per ogni $j = 0, 1, \ldots$

$$f(x_k + u\alpha_0\delta^j d_k) > f(x_k) - \gamma(\delta^j\alpha)^2\|d_k\|^2 \tag{3.60}$$

$$\alpha_0\delta^j\|d_k\| > \beta_k, \tag{3.61}$$

ma al tendere di j all'infinito, $\alpha_0\delta^j$ tende a zero e quindi la (3.61) non può valere, poichè β_k è una quantità positiva. □

Si può dimostrare il seguente risultato di convergenza ([9],[5])

Proposizione 3.12 *Sia $\{x_k\}$ la successione prodotta dall'algoritmo Line Search. Supponiamo che le direzioni D_k soddisfino la condizione (3.56) e che il parametro β_k soddisfi la relazione 3.57. Allora*

$$\lim_{k\to\infty}\|\nabla f(x_k)\| = 0. \tag{3.62}$$

3.4 Metodi pattern-line

I metodi con proprietà teoriche fin qui considerati presentano vantaggi e svantaggi. In particolare, per quel che riguarda i metodi di tipo Pattern Search, si possono fare le seguenti considerazioni:

- hanno una buona capacità di individuare una "buona" direzione e approssimano in maniera accurata l'andamento della funzione nell'intorno del punto corrente perchè ad ogni iterazione utilizzano un insieme di direzioni ricco;

- d'altro canto le condizioni che le direzioni devono soddisfare sono piuttosto stringenti e questo limita la scelta delle direzioni di ricerca;

- le regole di aggiornamento del passo Δ_k sono rigide e questo può portare ad una convergenza lenta dell'algoritmo;

- ad ogni iterazione, la funzione obiettivo viene valutata lungo tutte le r direzioni p_k^i, $i = 1, \ldots, r$ e questo può essere costoso dal punto di vista computazionale.

Per quel che riguarda gli algoritmi di tipo Line Search, si hanno i seguenti vantaggi e svantaggi

- si possono effettuare passi lunghi, che permettono di sfruttare al meglio una "buona" direzione;

- di contro, ad ogni iterazione viene presa in considerazione un'unica direzione, che può non essere sufficiente a caratterizzare l'andamento della funzione obiettivo nell'intorno del punto corrente;

- la linesearch viene effettuata sempre anche se il passo iniziale non soddisfa la condizione (3.52) e questo può portare a sprecare un gran numero di calcoli di funzione.

Partendo da queste considerazioni in [11] sono stati proposti dei nuovi algoritmi che sono basati su un approccio ibrido tra quello Pattern Search e quello Line Search. In particolare, l'idea è quella di valutare ad ogni iterazione la funzione obiettivo su un insieme di direzioni $\{p_k^i\}$, $i = 1, \ldots, r$ alla ricerca di una "buona" direzione di discesa, cioè una direzione che garantisca un "sufficiente" decremento. Ogni volta che si individua una "buona" direzione viene effettuata una line search lungo questa direzione in modo da sfruttarne la proprietà il più possibile e da individuare un passo "sufficientemente" grande. Un esempio di algoritmo Pattern-Line è il seguente:

Algoritmo Pattern-Line

Dati: $x_0 \in \Re^n$, $\tilde{\alpha}_0^i > 0$, $i = 1, \ldots, r$, $\gamma > 0$, $\delta, \theta \in (0,1)$.

Passo 0: Poni $k = 0$.

Passo 1: Poni $i = 1$, $y_k^i = x_k$.

Passo 2: Se

$$f(y_k^i + \tilde{\alpha}_k^i p_k^i) \leq f(y_k^i) - \gamma(\tilde{\alpha}_k^i)^2$$

allora calcola α_k^i tramite la procedura LS$(\tilde{\alpha}_k^i, y_k^i, p_k^i, \gamma, \delta)$ e poni $\tilde{\alpha}_{k+1}^i = \alpha_k^i$. Altrimenti poni $\tilde{\alpha}_{k+1}^i = \theta \tilde{\alpha}_k^i$ e $\alpha_k^i = 0$. Poni $y_k^{i+1} = y_k^i + \alpha_k^i p_k^i$.

Passo 3: Se $i < r$, poni $i = i + 1$ e vai al Passo 2. Altrimenti poni $x_{k+1} = y_k^{r+1}$, $k = k + 1$ e vai al Passo 1.

Procedura LS$(\tilde{\alpha}_k^i, y_k^i, p_k^i, \gamma, \delta)$

Calcola $\alpha_k^i = \min\{\delta^{-j} : \ j = 0, 1, \ldots\}$ tale che

$$f(y_k^i + \alpha_k^i p_k^i) \leq f(y_k^i) - \gamma(\alpha_k^i)^2, \tag{3.63}$$

$$f(y_k^i + \frac{\alpha_k^i}{\delta} p_k^i) \geq \max\left[f(y_k^i + \alpha_k^i p_k^i), f(y_k^i) - \gamma(\frac{\alpha_k^i}{\delta})^2 \right] \tag{3.64}$$

Questo algoritmo ad ogni iterazione esamina l'andamento della funzione obiettivo lungo tutte le direzioni di ricerca p_k^i, $i = 1, \ldots, r$. Tuttavia, appena trova una direzione promettente lungo cui è verificata la condizione di sufficiente decremento $f(y_k^i + \tilde{\alpha}_k^i p_k^i) \leq f(y_k^i) - \gamma(\tilde{\alpha}_k^i)^2$, l'algoritmo produce un nuovo punto effettuando una line search lungo p_k^i in modo da avere un passo sufficentemente grande. Inoltre in questo algoritmo si ha un passo diverso $\tilde{\alpha}_k^i$ per ogni direzione che viene aggiornato in base alle informazioni raccolte durante l'iterazione. Questo aspetto è particolarmente utile quando le direzioni rimangono costanti, si ha cioè $p_k^i = \bar{p}^i$ per ogni k, in quanto permette di tener conto dell'andamento

della funzione obiettivo lungo la singola direzione. Si può dimostrare il seguente risultato di convergenza:

Proposizione 3.13 ([11]) *Sia $\{x_k\}$ la sequenza di punti prodotta dall'algoritmo Pattern-Line. Le direzioni p_k^i, $i = 1, \ldots, r$ siano tali da soddisfare la Condizione C1. Allora l'algoritmo Pattern-Line è ben definito e si ha*

$$\lim_{k\to\infty} \nabla f(x_k) = 0. \tag{3.65}$$

Dim. Prima di tutto dimostriamo che l'algoritmo è ben definito. Dobbiamo far vedere che, dato un intero $i \leq r$ tale che il test al Passo 2 be soddisfatto, esiste un intero j per cui valgono le due relazioni (3.63) e (3.64) con $\alpha_k^i = \delta^{-j}\tilde{\alpha}_k^i$. Supponiamo per assurdo che questo non sia vero, cioè che per ogni $j = 0, 1, \ldots$ si abbia

$$f(y_k^i + \delta^{-j}\tilde{\alpha}_k^i p_k^i) < f(y_k^i) - \gamma(\delta^{-j}\tilde{\alpha}_k^i)^2, \quad \text{per ogni } j, \tag{3.66}$$

oppure

$$f(y_k^i + \delta^{-j-1}\tilde{\alpha}_k^i p_k^i) < f(y_k^i + \delta^{-j}\tilde{\alpha}_k^i p_k^i) \leq f(y_k^i) - \gamma(\delta^{-j}\tilde{\alpha}_k^i)^2, \quad \text{per ogni } j. \tag{3.67}$$

Prendendo il limite per $j \to \infty$, si ottiene in entrambi i casi che f è illimitata inferiormente, che contraddice l'Assunzione 3.5.
Ora dimostriamo la (3.65) dimostrando che sono soddisfatte le ipotesi (a), (b) e (c) della Proposizione 3.6. Le istruzioni dell'algoritmo implicano $f(x_{k+1}) \leq f(x_k)$ e quindi la condizione (a) è soddisfatta. La condizione (b) è vera per ipotesi. Rimane da dimostrare l'ipotesi (c).
Per far questo si dimostra prima di tutto che per $i = 1, \ldots, r$, si ha

$$\lim_{k\to\infty} \alpha_k^i = 0 \tag{3.68}$$

e

$$\lim_{k\to\infty} \tilde{\alpha}_k^i = 0. \tag{3.69}$$

Dalle istruzioni dell'algoritmo si ha

$$f(x_{k+1}) = f(y_k^{r+1}) \leq f(x_k) - \gamma \sum_{i=1}^{r} (\alpha_k^i)^2,$$

da cui segue

$$f(x_{k+1}) - f(x_k) \leq -\gamma \sum_{i=1}^{r} (\alpha_k^i)^2. \tag{3.70}$$

Poichè la sequenza $\{x_k\}$ appartiene tutta all'insieme $\mathcal{L}_0$ che è compatto ed è quindi convergente. Di conseguenza abbiamo che $\{f(x_k)\}$ è convergente, e quindi la (3.70) implica che $\sum_{i=1}^{r} (\alpha_k^i)^2$ tende a zero, da cui segue la (3.68).
Consideriamo ora gli $\tilde{\alpha}_k^i$, $i = 1, \ldots, r$. Dato un indice $i \in \{1, \ldots, r\}$, dividiamo la sequenza di iterazioni $\{k\}$ in due sottoinsiemi, K e $\bar{K}$:

$$K = \{k : \ \alpha_k^i > 0\}, \quad \bar{K} = \{k : \ \alpha_k^i = 0\}.$$

Per ogni $k \in K$, abbiamo $\tilde{\alpha}_k^i \leq \alpha_k^i$, da cui segue che se K è infinito si ha

$$\lim_{k \to \infty, k \in K} \tilde{\alpha}_k^i = 0. \tag{3.71}$$

Per ogni $k \in \bar{K}$, sia m_k l'indice più grande tale che $m_k \in K$, $m_k < k$ (poniamo $m_k = 0$ se questo indice non esiste). Avremo dalle istruzioni dell'algoritmo

$$\tilde{\alpha}_k^i = (\theta)^{k - m_k} \tilde{\alpha}_{m_k}^i. \tag{3.72}$$

Al tendere di k all'infinito per $k \in \bar{K}$, avremo che o tende all'infinito m_k (se K è un insieme infinito), o tende all'infinito k (se $\bar{K}$ è un insieme infinito). Nel primo caso, il fatto che $\theta \in (0, 1)$ implica

$$\lim_{k \to \infty, k \in \bar{K}} \tilde{\alpha}_k^i = 0. \tag{3.73}$$

Nel secondo la (3.73) è implicata dalla (3.71). Abbiamo quindi che dalla (3.71) e dalla (3.73) segue la (3.69). Se poniamo quindi

$$\xi_k^i = \begin{cases} \frac{\alpha_k^i}{\delta} & \text{se } k \in K \\ \tilde{\alpha}_k^i, & \text{se } k \in \bar{K} \end{cases}, \tag{3.74}$$

dalle istruzioni dell'algoritmo otteniamo

$$f(y_k^i + \xi_k^i p_k^i) \geq f(y_k^i) - \gamma (\xi_k^i)^2.$$

Inoltre le due condizioni (3.68) e (3.69) implicano $\lim_{k\to\infty} \xi_k^i = 0$. Le istruzioni dell'algoritmo implicano

$$\|y_k^i - x_k\| \leq \sum_{j=1}^{i-1} \alpha_k^j \|p_k^j\|,$$

che combinata con la (3.68) implica

$$\lim_{k\to\infty} \|x_k - y_k^i\| = 0,$$

e questo completa la dimostrazione della condizione (c) della Proposizione 3.6. □

Questo teorema dimostra le proprietà di convergenza dell'algoritmo Pattern-Line. Per tentare di accelerare la convergenza di questo algoritmo si può tentare di sfruttare ulteriormente le informazioni raccolte durante un'iterazione traendo ispirazione dai metodi di modellizzazione. Infatti, al termine di ogni iterazione di questo algoritmo, si hanno a disposizione dei punti su cui si è valutata la funzione obiettivo. In particolare si hanno per ogni $i = 1, \ldots, r$ i punti $y_k^i + \tilde{\alpha}_k^i p_k^i$ e, in caso di successo lungo l'i-esima direzione, anche tutti i punti di prova utilizzati all'interno della procedura LS. Questi punti sono "vicini" al punto corrente x_k e possono essere utilizzati insieme ai corrispondenti valori della funzione obiettivo per costruire un interpolante dei f del tipo (1.5) che soddisfi la (1.6) con l'insieme Y costituito dai punti visitati durante l'iterazione. Si minimizza poi su un'opportuna sfera il modello così ottenuto e se il punto

$$x_k^* = \min_{\|s\| \leq \Delta_k} m_k(x_k + s)$$

soddisfa $f(x_k^*) \leq f(y_k^{r+1})$, al passo 3 si pone $x_{k+1} = x_k^*$. Le proprietà di convergenza dell'algoritmo rimangono invariate, ma in questo modo si sfruttano ulteriormente le informazioni raccolte durante l'iterazione.

Bibliografia

[1] K. S. A. R. Conn and P. L. Toint, *Recent progress in unconstrained nonlinear optimization without derivatives*, Mathematical Programming, 79 (1997), pp. 397–414.

[2] K. S. A.R. Conn and P. Toint, *Approximation Theory and Optimization: tributes to M.J.D. Powell*, Cambridge University Press, Cambridge, UK, 1997, ch. On the convergence of derivative-free methods for unconstrained optimization, pp. 83–108.

[3] W. C. Davidon, *Variable metric method for minimization*, SIAM Journal on Optimization, 1 (1991), pp. 1–17. with a belated preface for ANL 5990.

[4] C. Davis, *Theory of positive linear dependence*, American Journal of Mathematics, 76 (1954), pp. 733–746.

[5] L. Grippo, *A class of unconstrained minimization methods for neural network training*, Optimization Methods and Software, 4 (1994), pp. 135–150.

[6] L. Grippo, F. Lampariello, and S. Lucidi, *Global convergence and stabilization of unconstrained minimization methods without derivatives*, Journal of Optimization Theory and Applications, 56 (1988), pp. 385–406.

[7] R. Hooke and T. A. Jeeves, *Direct search solution of numerical and statistical problems*, Journal of ACM, 8 (1961), pp. 212–229.

[8] M. W. J.C. Lagarias, J.A. Reeds and P. E. Wright, *Convergence properties of the nelderMead simplex method in low dimensions*, SIAM Journal on Optimization, 9 (1998), pp. 112–147.

[9] R. D. Leone, M. Gaudioso, and L. Grippo, *Stopping criteria for linesearch methods without derivatives*, Mathematical Programming, 30 (1984), pp. 285–300.

[10] S. Lucidi and M. Sciandrone, *On the global convergence of derivative free methods for unconstrained optimization*, TR 18-96, Department of Computer and System Sciences "A. Ruberti", University of Rome "La Sapienza", Rome, Italy, 1996.

[11] ——, *On the global convergence of derivative free methods for unconstrained optimization*, SIAM J. Optimization, 13 (2002), pp. 97–116.

[12] M. Marazzi and J. Nocedal, *Wedge trust region methods for derivative free optimization*, Mathematical Programming, 91 (2002), pp. 289–305.

[13] K. I. M. Mckinnon, *Convergence of the nelder-mead simplex method to a nonstationary point*, SIAM Journal on Optimization, 9 (1998), pp. 148–158.

[14] J. Nelder and R. Mead, *A simplex method for function minimization*, The Computer Journal, 8 (1965), pp. 308–313.

[15] M. Powell, *UOBYQA: unconstrained optimization by quadratic approximation*, Mathematical Programming, 95 (2002), pp. 555–582.

[16] M. D. Powell, *On trust region methods for unconstrained minimization without derivatives*, Mathematical Programming Ser. B, 97 (2003), pp. 605–623.

[17] V. Torczon, *On the convergence of the multidirectional search algorithms*, SIAM Journal on Optimization, 1 (1991), pp. 123–145.

[18] ——, *On the convergence of pattern search algorithms*, SIAM Journal on Optimization, 7 (1997), pp. 1–25.

[19] M. H. Wright, *Direct search methods: once scorned, now respectable*, in Numerical Analysis 1995 (Proceedings of the 1995 Dundee Biennal Conference in Numerical Analysis), D. F. Griffiths and G. A. Watson, eds., Pitman Res. Notes Math. Ser. 344, Boca Raton, FL, 1996, CRC Press, pp. 191–208.