UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA DELL'INFORMAZIONE

"FONDAMENTI DI OTTIMIZZAZIONE CONVESSA"

Relatore: Prof. Sandro Zampieri

Laureando: Leonardo Beltrame

ANNO ACCADEMICO 2021 – 2022

Data di laurea 20/09/2022

Ai miei genitori, alla mia famiglia, ai miei amici e a Padova

Fondamenti di Ottimizzazione Convessa

Abstract

L'ottimizzazione è una branca della matematica che si occupa di metodi per la ricerca di massimi/minimi locali e/o globali di funzioni che rappresentano soprattutto costi legati all'economia o all'efficienza di un algoritmo. In questa tesi descriverò gli aspetti teorici dell'ottimizzazione convessa, ovvero prendendo in considerazione funzioni che hanno al massimo un punto critico, definite su domini numerici continui; inoltre approfondirò il problema dell'ottimizzazione vincolata, quindi i risultati ammissibili vivono in un intervallo definito e limitato, usando il concetto di dualità. Confronterò i risultati principali del metodo di Newton e del gradiente.

Indice

Capitolo 1

Introduzione

1.1 Ottimizzazione matematica

Un problema di ottimizzazione può essere presentato nel seguente modo:

$$\begin{array}{ll} min & f_0(x) \\ soggetta \quad a & f_i(x) \leq a_i \quad i = 1, \ldots, n \\ & h_j(x) = b_j \quad j = 0, \ldots, m \end{array}$$

La funzione $f_0 : \mathbf{R}^n \to \mathbf{R}$ è chiamata funzione Obiettivo o funzione Costo, mentre le diverse famiglie $f_i : \mathbf{R}^n \to \mathbf{R}$ e $h_j : \mathbf{R}^n \to \mathbf{R}$ sono funzioni che rappresentano condizioni o vincoli al problema.

Risolvere un tale problema implica quindi ricercare uno o più valori x^* che soddisfino le specifiche del problema; ovvero x^* deve essere tale che:

- $x^* \in D$, dove $D = dom(f_0) \cap (\cap_{i=1}^{n} dom(f_i)) \cap (\cap_{j=0}^{m} dom(h_j))$
- $f_0(x^*) = p^*$, dove $p^* = inf\{\ f_0(x) \mid f_i(x) \leq 0 \quad i = 1, \ldots, n, \quad h_j(x) = 0 \quad j = 0, \ldots, m\ \}$ è chiamato valore ottimale.

Questa soluzione potrebbe non esistere nel caso in cui non appartenesse al dominio della funzione Obiettivo o delle funzioni Vincolo. In questo caso la soluzione si dice non ammissibile.

Oppure la funzione Obiettivo potrebbe essere tale da non avere un valore ottimale.

Le prime semplici applicazioni di problemi di ottimizzazione ci vengono dall'Analisi Matematica in una variabile reale; infatti se consideriamo come esempio il seguente:

$$min \quad f_0(x) = x^2 + 1$$

Notiamo intanto che si tratta di un problema privo di vincoli, ovvero si ha $m = n = 0$, la cui soluzione viene immediatamente dal Teorema di Fermat

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(f_0(x)\right) = 0 \iff 2x = 0 \iff x^* = 0$$

e quindi

$$p^* = f_0(x^*) = 1$$

In questo caso, ma ciò non vale in generale, x^* è soluzione globale e non solo locale (che è ciò che il Teorema di Fermat ci assicura in $\mathbf{R}$).

Questa tesi ruota attorno all'importante concetto di *convessit*, insieme all'ottimizzazione convessa, ovvero un problema di ottimizzazione definito su funzioni e domini convessi, che ha l'importante caratteristica di poter essere risolto globalmente.

1.2 Applicazioni

La branca dell'ottimizzazione trova molte applicazioni in diversi settori quali informatica, economia, statistica, elettronica, meccanica ed è in grado persino di essere rilevante in ambiti legati alla vita di tutti i giorni.

Prendiamo come esempio un problema di dieta.

Sappiamo che una persona, per avere una dieta sana, ha bisogno almeno di m differenti sostanze nutrienti (es. carboidrati, proteine, vitamine...), che possiamo rappresentare tramite il vettore $\mathbf{b} = \{b_1, b_2, \ldots, b_m\}$; da notare che tutte le componenti di $\mathbf{b}$ devono avere la stessa unità di misura affinché possano essere poste in relazione tra loro (es. $Joule, calorie, grammi \ldots$).

Consideriamo successivamente un secondo vettore $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$ che rappresenti n cibi diversi, ognuno esplicabile tramite combinazione lineare dei nutrienti b_i, ed ognuno con un certo costo c_k.

Allora il problema di trovare la dieta sana più economica si può modellare come:

$$\begin{aligned} &min && c^T x \\ &soggetta \quad a && Ax \succeq b \\ & && x \succeq 0 \end{aligned}$$

dove

- $c^T x : \mathbf{R}^n \to \mathbf{R}$ è la funzione obiettivo da minimizzare;
- il primo vincolo di disuguaglianza, che si può riscrivere come $Ax - b \succeq 0$, indica come i vari cibi, pesati con i vari a_{ij}, ovvero scalari che rappresentano la quantità di nutriente i contenuta in un cibo j organizzati nella matrice A, debbano essere sufficienti per soddisfare il fabbisogno giornaliero;
- l'ultimo vincolo impone banalmente che i cibi introdotti devono essere a valori positivi.

Questo è un classico problema di ottimizzazione convessa, in quanto la funzione costo è una funzione lineare, che vedremo essere contemporaneamente concava e convessa, mentre il vincolo di disuguaglianza è una funzione affine, ovvero differisce da una funzione lineare per una costante.

Da notare che se ad ulteriore esempio i pesi c_k rappresentassero quanto piace il rispettivo cibo x_k il problema sarebbe equivalente a questo, ma varrebbe per un singolo soggetto o per un insieme limitato di persone.

Un possibile metodo di risoluzione del problema

$$\begin{aligned} &min && f_0(x) \\ &soggetta \quad a && f_i(x) \leq a_i \quad i = 1, \ldots, n \\ & && h_j(x) = b_j \quad j = 0, \ldots, m \end{aligned}$$

è il metodo dei moltiplicatori di Lagrange, che vedremo fornire un lower bound al valore ottimale p^*. Quest'ultimo è molto apprezzato per la sua relativa semplicità di implementazione, e il suo principale problema (ovvero il fatto che restituisca un lower bound e non il valore cercato) può essere risolto quando il problema di ottimizzazione è convesso.

Al fine di comprendere al meglio gli argomenti trattati in seguito è necessario definire il concetto di segno per una matrice. Di particolare interesse per questa tesi è la seguente definizione di matrice semidefinita positiva:

Una matrice A $\in \mathbf{R}^{n\times n}$ simmetrica a coefficienti reali

si dice semidefinita positiva se $\forall x \in \mathbf{R}^n$ vale

$$xAx^T \geq 0, \text{ con } x \neq 0.$$

Si ricordi inoltre che A è una matrice simmetrica se e solo se vale $A = A^T$ e che la matrice Hessiana, sotto ipotesi di continuità delle derivate seconde, risulta essere una matrice simmetrica.

Un'altra definizione valida è la seguente e si basa sulla regola dei segni di Cartesio applicata al polinomio caratteristico:

Una matrice A $\in \mathbf{R}^{n\times n}$ simmetrica a coefficienti reali

si dice semidefinita positiva se e solo se tutti i suoi autovalori sono non negativi.

Questa proprietà si indica con la simbologia $A \succeq 0$.

1.3 Altri metodi

Non di rado accade di dover avere a che fare con funzioni non convesse o non lineari, per le quali risulti difficile ridurre il problema in sottoproblemi equivalenti di più semplice risoluzione.

In particolare se il problema si presenta come:

$$min \quad f(x)$$

dove $f : \mathbf{R}^n \to \mathbf{R}$ è una funzione continua e due volte differenziabile, allora sappiamo che i minimi (massimi) locali (globali) si trovano ponendo $\nabla f(x^*) = 0$, dove $\nabla f(x) = (\frac{\mathrm{d}f(x)}{\mathrm{d}x_1}, \frac{\mathrm{d}f(x)}{\mathrm{d}x_2}, \ldots, \frac{\mathrm{d}f(x)}{\mathrm{d}x_n})$.

Questo perché il problema è privo di vincoli.

In qualche caso è possibile risolvere l'equazione per via analitica, ma spesso c'è bisogno di algoritmi iterativi che minimizzino la funzione "spostandosi" lungo la direzione dell'antigradiente, ovvero $-\nabla f(x_0)$, che rappresenta la direzione di massima decrescita partendo da una certa condizione iniziale x_0.

In questa tesi tratteremo due dei principali algoritmi iterativi che permettono di trovare efficientemente una soluzione, per lo meno locale, e in altri casi globale, al problema, ovvero il Metodo della discesa del gradiente e il Metodo di Newton, anche conosciuto come metodo delle tangenti.

Il primo è un metodo del primo ordine, ovvero sfrutta la definizione di gradiente per iterare il movimento lungo i sottolivelli della funzione.

Il secondo è del secondo ordine, quindi vi sarà il bisogno esplicito di calcolare la matrice Hessiana $\nabla^2 f(x_0)$ di $f(x)$.

Gli pseudocodici sono i seguenti:

1. Metodo della discesa del gradiente

$$x_0 \in \mathbf{R}^n, \quad k = 0$$

$$while\ (\nabla f(x_k) \neq 0)\ \{$$

$$d_k = -\nabla f(x_k);$$

$$\text{calcolo il passo di discesa } \alpha_k;$$

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

$$k++; \quad \}$$

2. Metodo di Newton

$$x_0 \in \mathbf{R}^n, \quad k = 0$$
$$while\ (\nabla f(x_k) \neq 0)\ \{$$
$$x_{k+1} = x_k - \nabla^2 f(x_k)^{-1} \nabla f(x_k);$$
$$k++; \quad \}$$

Vedremo che questi due algoritmi, sebbene portino allo stesso risultato, hanno caratteristiche diverse sia nell'idea alla base sia nella complessità computazionale.

Nella figura sottostante è rappresentato il grafico di una superficie con le sue curve di livello. Una curva di livello (nel disegno ne sono rappresentate 8) è l'insieme dei punti che hanno la stessa immagine nella funzione, ovvero $L_y = \{x \mid f(x) = y\}$. In questo caso, col metodo della discesa del gradiente è possibile risalire al minimo, che si trova al centro.

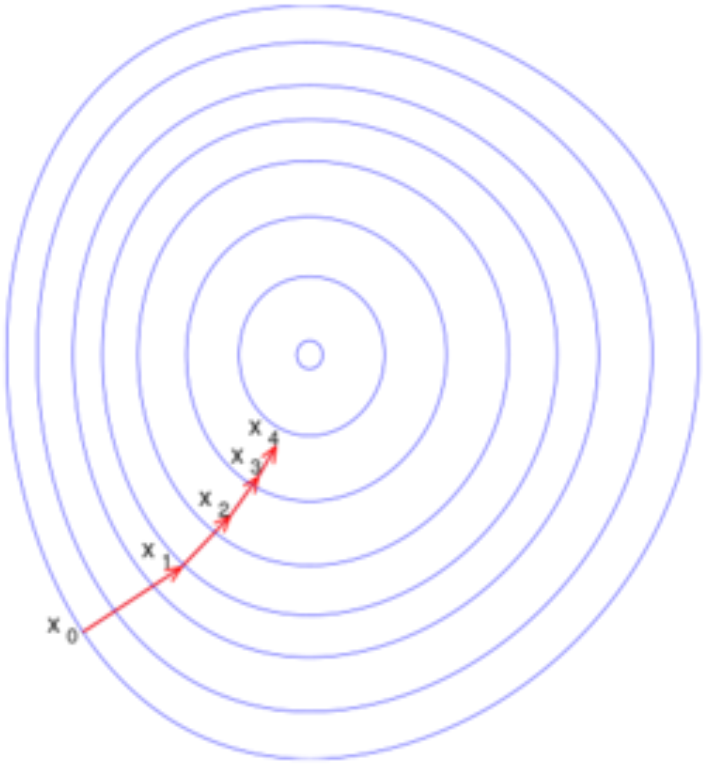


Figura 1.1

Si noti come il passo di discesa, che sarebbe la lunghezza dello spostamento lungo le curve nell'iterazione k, diminuisce man mano che ci si avvicina al minimo. Questo intuitivamente perché più siamo vicini al punto ottimale (che sarebbe poi x^*, soluzione al problema di ottimizzazione senza vincoli), meno informazione abbiamo per muoverci.

Capitolo 2

Convessità e Dualità

2.1 Insiemi convessi

Un importante concetto che può aiutare a risolvere in maniera efficiente molti problemi di ottimizzazione è quello della convessità.

Non basta però che la funzione obiettivo sia convessa affinché il problema di ottimizzazione sia convesso. Infatti un problema del tipo:

$$\begin{array}{ll} min & f_0(x) \\ soggetta \quad a & f_i(x) \leq a_i \quad i = 1, \ldots, n \\ & h_j(x) = b_j \quad j = 0, \ldots, m \end{array}$$

risulta convesso se e solo se tutti vincoli di disuguaglianza sono convessi e quelli di uguaglianza affini, oltre alla convessità della funzione $f_0(x)$.

Per quanto riguarda il dominio degli elementi ammissibili, questo è sicuramente un insieme convesso, in quanto

$$D = (\cap_{i=0}^{n} dom(f_i)) \cap (\cap_{j=0}^{m} dom(h_i))$$

è una intersezione multipla di insieme convessi e iperpiani, in quanto l'intersezione di insiemi convessi è un insieme convesso.

Le definizione di insieme convesso è:

(1) un insieme A si dice convesso se e solo se $\forall (x, y) \in A$ si ha $\{(1-\theta)x + \theta y \mid \theta \in [0; 1]\} \subset A$.

Risulta importante allo scopo di comprendere al meglio la definizione appoggiarsi al concetto di segmento nel piano (o di segmento in un iperspazio se in n variabili).

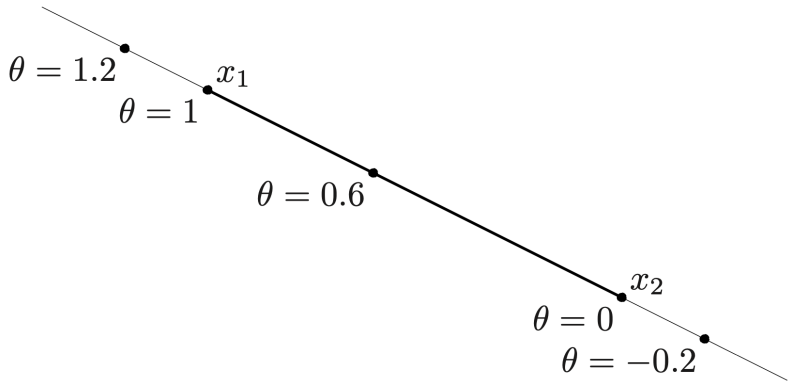


Figura 2.1

Come si vede dalla figura 2.1, il valore di θ individua la posizione occupata dal punto $((1-\theta)x;\ \theta y)$ lungo il segmento che congiunge x e y. In particolare se $\theta = 0$ ritroviamo il punto x, se $\theta = 1$ abbiamo il punto y. Per $\theta > 1$ e $\theta < 0$, è possibile descrivere tutti i punti della retta passante per x e y.

Da questo si deduce il significato della definizione (1), questa infatti afferma che affinché l'insieme sia convesso occorra che, dati due punti qualsiasi del piano, il segmento che li congiunge sia interamente contenuto nell'insieme.

L'esempio più banale di insieme convesso è $\emptyset$, ovvero l'insieme vuoto. Un altro è $\mathbf{R}^n$. Un insieme convesso meno banale è il segmento stesso, che per definizione risulta inglobare ogni segmento congiungente punti appartenenti ad esso.

Quando si lavora in più dimensioni è utile generalizzare i concetti che appartengono ad $\mathbf{R}$ e $\mathbf{R}^2$. Questo per riconoscere con immediatezza spazi convessi o meno.

Un buon punto di partenza è l'*iperpiano*, che generalizza il concetto di retta. Un iperpiano è così definito:

$$\{x \mid a^T x = b\},$$

con $a \in \mathbf{R}^n$, $a \neq 0$ e $b \in \mathbf{R}$.

Come si vede dalla figura sottostante, un iperpiano in n variabili divide $\mathbf{R}^n$ in due insiemi, $\{x \mid a^T x \geq b\}$ e $\{x \mid a^T x \leq b\}$. Questi due insiemi si possono dimostrare essere entrambi convessi, ma non affini.

Da questo concetto appare molto chiaro anche il motivo per cui i vincoli di uguaglianza, affinché il problema sia convesso, debbano essere affini:

$$h_j(x) = b_i \iff \begin{cases} a_j^T x \leq b_i \\ a_j^T x \geq b_i \end{cases}$$

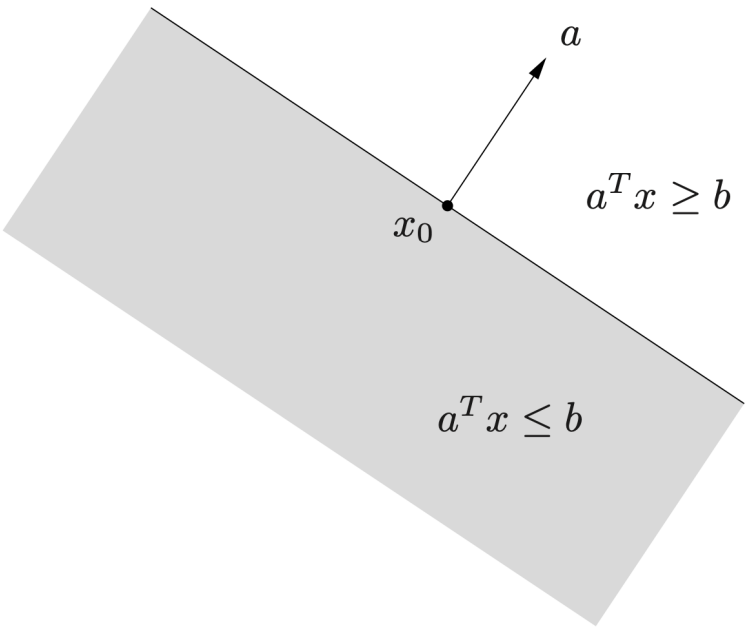
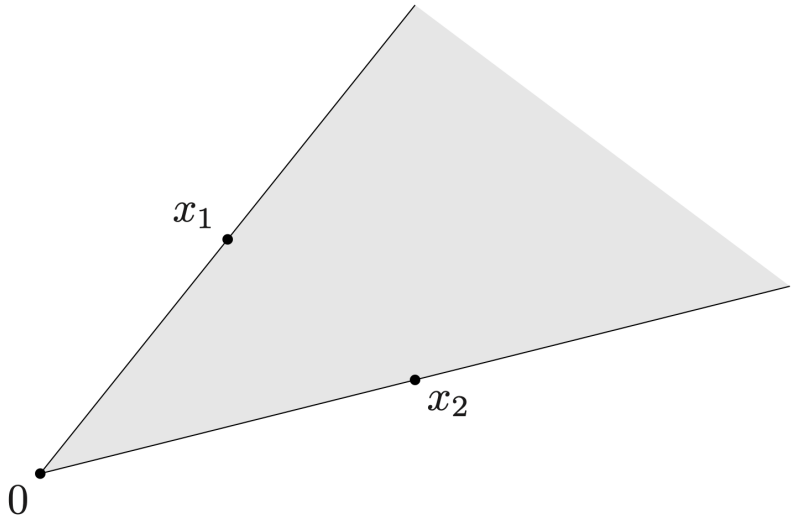


Figura 2.2

Un altro esempio non banale è quello del cono convesso.

Un insieme C è chiamato *cono* se $\forall x \in C$ e $\theta \geq 0$ si ha $\theta x \in C$.

La figura in basso mostra il più semplice esempio di cono convesso, in $\mathbf{R}^2$. In questo caso se prendiamo due punti x_1 e x_2 all'interno dell'insieme C e consideriamo una combinazione lineare del tipo $\theta_1 x_1 + \theta_2 x_2$, essa deve stare all'interno dell'insieme di partenza.



Un oggetto del tipo $\theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_k x_k$, con $\theta_1, \theta_2, \ldots, \theta_k > 0$, è chiamato combinazione conica ed è la generalizzazione di quanto visto sopra. Un cono C è convesso quindi in $\mathbf{R}^n$ se avendo i vari $x_i \in C$, tutte le combinazioni lineari a coefficienti positivi appartengono a C.

2.2 Funzioni convesse

La definizione formale di funzione convessa è la seguente:

(2) una funzione $f(x) : \mathbf{I} \to \mathbf{R}$

si dice convessa in $\mathbf{I}$ se e solo se $\mathbf{I}$ è convesso e $\forall (x, y)$ vale

$$f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y), \text{ limitatamente per } \lambda \in [0, 1].$$

Anche qui il concetto di segmento generalizzato gioca un ruolo cardine. La seconda definizione impone che l'immagine del segmento, ovvero tutti i punti del segmento mappati tramite f in $\mathbf{R}$, rimanga sempre minore o al più uguale al segmento che congiunge le immagini dei due punti considerati, questo $\forall \lambda \in [0, 1]$.

Con questa definizione risulta immediato dimostrare la convessità di semplici funzioni convesse. Un esempio può essere quello del paraboloide in $\mathbf{R}^3$ di equazione $f(x, y) = x^2 + y^2$.

Come prima cosa si scelgono due punti arbitrari (x_1, y_1) e (x_2, y_2). Dopodiché partendo dalla definizione si ha:

$[(1-\lambda)x_1 + \lambda x_2]^2 + [(1-\lambda)y_1 + \lambda y_2]^2 \leq (1-\lambda)(x_1^2 + x_2)^2 + \lambda(y_1^2 + y_2^2)$

$[x_1 - \lambda(x_1 - x_2)]^2 + [x_1 - \lambda(y_1 - y_2)]^2 \leq x_1^2 + y_1^2 - \lambda(x_1^2 + y_1^2) + \lambda(x_2^2 + y_2^2)$

...

dopo una serie di passaggi algebrici si ottiene

$\lambda[(x_1 - x_2)^2 + (y_1 - y_2)^2] \leq (x_1 - x_2)^2 + (y_1 - y_2)^2$

Ricordando che $0 \leq \lambda \leq 1$, la disuguaglianza è sempre rispettata $\implies f(x, y)$ è convessa.

Altre due importanti caratteristiche delle funzioni convesse, che hanno poi portato all'introduzione di due ulteriori condizioni di convessità (nessuna di queste ovviamente è in contraddizione con l'altra), sono legate alle approssimazioni di Taylor del primo e secondo ordine della funzione.

(3) una funzione $f(x) : \mathbf{R}^n \to \mathbf{R}, f \in C^1$

è convessa se e solo se $dom(f)$ è un insieme convesso e vale

$$f(x) \geq f(x_0) + \nabla f(x_0)^T (x - x_0),$$

questo $\forall x, x_0 \in dom(f)$.

In particolare quindi l'approssimazione lineare di Taylor in ogni punto della funzione deve essere minore o al più uguale alla funzione stessa, ovvero stare sotto quest'ultima. Per $x = x_0$ la disequazione diventa un'uguaglianza in quanto il valore della retta coincide col valore assunto dalla funzione nel punto in cui si approssima.

(4) una funzione $f(x) : \mathbf{R}^n \to \mathbf{R}, f \in C^2$

è convessa se e solo se $dom(f)$ è un insieme convesso e vale

$$\nabla^2 f(x) \succeq 0,$$

ovvero se la matrice Hessiana della funzione risulta semidefinita positiva.

Consideriamo come esempio la funzione $f : \mathbf{R}^n \to \mathbf{R}$, $dom(f) = \mathbf{R}^n$, insieme banalmente convesso, definita come $f(x) = \frac{1}{2} x^T A x + b^T x + c$

Notiamo come non si possa dire nulla sulla convessità o meno di questa funzione, in quanto è quadratica e per $n = 1$ vale $f(x) = \frac{1}{2} a x^2 + bx + c$, dove sappiamo essere

il coefficiente a a determinare il verso di curvatura della parabola.

Si ha $\nabla f = Ax + b$ e quindi $\nabla^2 f = A$.

Quindi

$$A \succeq 0 \implies f(x) \quad convessa$$
$$A \preceq 0 \implies f(x) \quad concava.$$

Il vantaggio più grande che si può trarre dal riconoscere e maneggiare funzioni obiettivo convesse nei problemi di ottimizzazione è quello che la soluzione $\{x^* \in D \mid f(x^*) = p^*\}$, dove nel caso in cui $D = dom(f)$ allora $f'(x^*) = 0$, sarà globale in D.

Saremmo quindi sicuri del fatto che $p^* = min(f(x)),$ con $x \in D$.

A volte è possibile, e consigliabile, trasformare un problema di ottimizzazione non convesso in uno convesso. Questo porta ad un problema equivalente la cui soluzione coincide con quella del precedente, ma che potrà essere risolto con tecniche di ottimizzazione convessa quali la Programmazione Lineare, che opera su vincoli di disuguaglianza affini.

La LP (Linear Programmation) inizia verso la metà del Novecento e si presenta nella forma:

$$\begin{aligned} &min && c^T x + d \\ &soggetta \quad a && Ax \preceq b \\ & && Fx = h, \end{aligned}$$

dove $A \in \mathbf{R}^{m \times n}$ e $F \in \mathbf{R}^{p \times n}$.

Tralasciando il vincolo di uguaglianza, l'insieme D degli elementi ammissibili risulta essere un poliedro in $\mathbf{R}^n$, ovvero l'insieme convesso mostrato in figura sottostante per $n = 2$.

Il numero dei lati del poliedro e la sua forma sono rappresentati dalle m righe di A, ovvero dal numero di disequazioni lineari presenti nel problema. Risulta importante osservare il problema nella sua forma canonica, in modo da individuare il poliedro di riferimento correttamente; per questo spesso è necessario cambiare il ver-

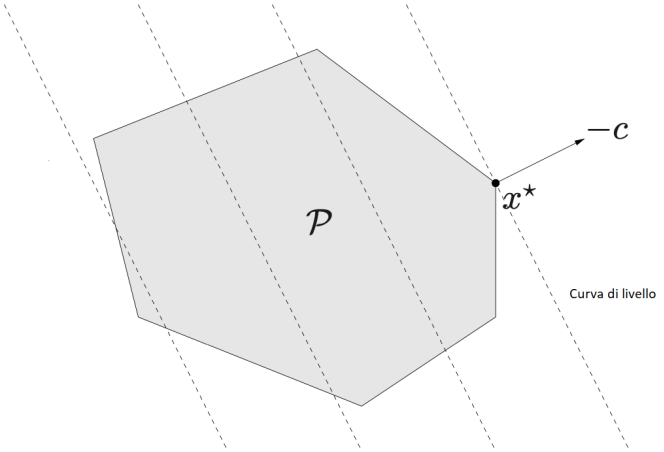


Figura 2.3

so di disequazione cambiando il segno o aggiungere costanti per mantenere i vincoli utili.

Anche il problema della dieta visto nell'introduzione risulta essere di Programmazione Lineare.

In quel caso il vincolo $Ax \succeq b$ diventa $-Ax + b \preceq 0$ rendendo chiara l'idea del poliedro come insieme dei valori ammissibili.

2.3 Dualità

La dualità è un concetto utile in matematica, oltre che nella logica e nell'informatica: permette di osservare un problema da un altro punto di vista rispetto agli approcci "standard".

Nell'ambito dell'analisi dei segnali in frequenza la Trasformata di Fourier è uno strumento del tutto duale alle operazioni fatte nel dominio del tempo.

In ottimizzazione la Funzione Lagrangiana e il rispettivo Metodo dei Moltiplicatori di Lagrange sono stati introdotti con la stessa logica, ovvero per ottenere un valore p^*, soluzione del problema di ottimizzazione, in maniera relativamente semplice ed affidabile.

La Funzione Lagrangiana si ottiene partendo da un problema di ottimizzazione del tipo

$$\begin{aligned} &min && f_0(x) \\ &soggetta \quad a && f_i(x) \leq a_i \quad i = 1, \ldots, n \\ & && h_j(x) = b_j \quad j = 0, \ldots, m, \end{aligned}$$

dove assumiamo come al solito $f_i(x) : \mathbf{R}^n \to \mathbf{R}$, $h_j(x) : \mathbf{R}^n \to \mathbf{R}$ e $D = dom(f_0) \cap (\cap_{i=1}^n dom(f_i)) \cap (\cap_{j=0}^m dom(h_j))$.

Si definisce allora la funzione Lagrangiana $L(x, \lambda, \mu) : \mathbf{R}^n \times \mathbf{R}^d \times \mathbf{R}^p$ come:

$$L(x, \lambda, \mu) = f_0(x) + \sum_{i=1}^{n} \lambda_i f_i(x) + \sum_{j=0}^{m} \mu_j h_j(x).$$

Questa funzione duale prevede l'introduzione dei coefficienti λ_i e μ_j per pesare al meglio i vincoli del problema.

Senza perdere in generalità, inglobando i parametri a_i e b_j nelle rispettive funzioni vincolo, si ha che sicuramente $f_i(x) - a_i \leq 0$ e $h_j(x) - b_j = 0$; si ha quindi che

$$L(x, \lambda, \mu) \leq f_0(x) \quad \forall x \in D.$$

Si definisce ora la funzione duale di Lagrange che permette, insieme alla Lagrangiana, di ottenere un lower bound al valore ottimale p^*.

$$g(\lambda, \mu) = inf_{x \in D}(L(x, \lambda, \mu)),$$

L'idea alla base di questa funzione duale è quella di, dato l'insieme D dei punti ammissibili rispettanti i vincoli del problema, trovare il valore minimo tra tutti questi e trovare i pesi ottimi per completare il problema equivalente.

Ipotizzando esista $\{x^* \mid f_0(x^*) = p^*\}$, soluzione del problema di ottimizzazione, si avrà

$$g(\lambda, \mu) = L(x^*, \lambda, \mu) \leq f_0(x^*) = p^*,$$

che porta a

$$g(\lambda, \mu) \leq p^*,$$

per tutti i $\lambda \in \mathbf{R}^d$ e i $\mu \in \mathbf{R}^p$.

La figura 2.5 mostra un esempio di utilizzo di una funzione Lagrangiana del tipo $L(x, \lambda)$, dove per semplicità assumiamo l'assenza di vincoli di uguaglianza.

In linea continua è rappresentata la funzione costo $f_0(x)$ mentre in tratteggiata una funzione vincolo $f_1(x)$ che possiamo pensare di disuguaglianza del tipo $f_1(x) \leq 0$. Le curve tratteggiate rappresentano invece la funzione Lagrangiana pesata al variare di λ.

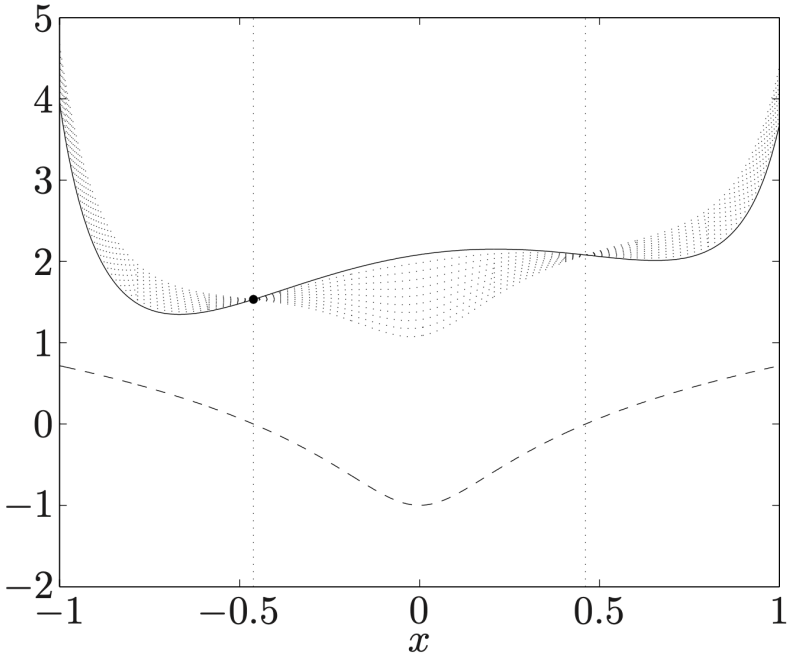


Figura 2.4

Possiamo notare come l'insieme dei punti ammissibili sia in questo caso l'intervallo $[-0.46, 0.46]$, in quanto è l'intervallo in cui la funzione Lagrangiana (l'insieme dei punti che contornano la funzione obiettivo, che è rappresentata in linea continua) sta al di sotto di $f_0(x)$, con $x^* = -0.46$ e $p^* = 1.54$. Se fosse presente un ulteriore

vincolo del tipo $x \geq 0$, come comunemente accade nei problemi di questo tipo dove gli input raramente sono valori negativi, il valore di x^* si intuirebbe essere dal grafico 0.46 con un corrispondente $p^* \approx 2$.

La figura 2.6 è molto interessante, in quanto mostra l'andamento della funzione duale di Lagrange $g(\lambda)$ al variare di $\lambda \in [0, 1]$; in linea tratteggiata è rappresentato il valore ottimale p^*. Vediamo come la funzione sta sempre sotto al punto ottimale e in particolare il suo estremo superiore fornisce un lower bound spesso accettabile e differisce da p^* di una quantità che chiamiamo gap ottimale di dualità.

Per il gap di dualità si può dimostrare che se il problema

$$\begin{aligned} &\textit{min} && f_0(x) \\ &\textit{soggetta} \quad \textit{a} && f_i(x) \leq 0 \quad i = 1, \ldots, n \\ & && Ax = b \end{aligned}$$

è convesso e vale la condizione di Slater, ovvero $\exists x \in dom(f)$ tale che $f_i(\hat{x}) < 0,\ i = 1, \ldots, n$ e $A\hat{x} = b$, allora la dualità è forte ovvero si ha $g(\lambda, \mu) = L(x^*, \lambda, \mu) = f_0(x^*) = p^*$.

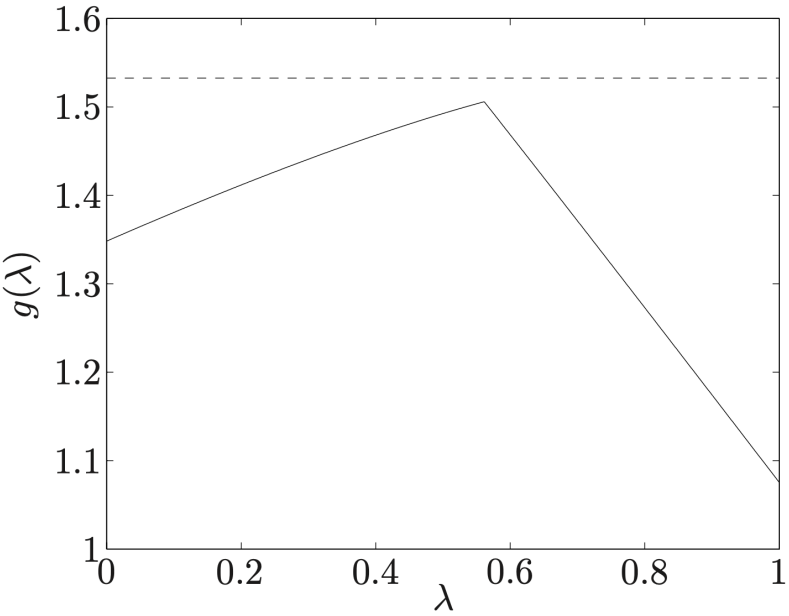


Figura 2.5

Altra cosa da notare è la concavità di $g(\lambda, \mu)$ in quanto l'estremo inferiore di una famiglia di funzioni affini di (λ, μ) risulta concava anche senza assumere che le

varie funzioni $f_i(x)$ siano convesse. Massimizzare quindi rispetto a (λ, μ) la funzione duale di Lagrange corrisponderà a minimizzare la sua opposta: $-g(\lambda, \mu)$.

Consideriamo il seguente esempio in $\mathbf{R}^3$.

$$\begin{aligned} &min && f(x,y) = x^2 + y^2 - 4y \\ &soggetta \quad a && x + 2y - 2 = 0, \end{aligned}$$

notiamo subito essere la funzione obiettivo convessa e il vincolo di uguaglianza affine, per cui il gap di dualità sarà nullo e il valore trovato col metodo dei moltiplicatori di Lagrange corrisponderà alla soluzione del problema di ottimizzazione vincolata.

Come prima cosa costruiamo la funzione Lagrangiana $L(x, y, \lambda) = x^2 + y^2 - 4y + \lambda(x + 2y - 2)$. Poi calcoliamo le derivate parziali rispetto a tutte e tre le variabili (notare che la derivata parziale rispetto a λ coincide con l'equazione del vincolo) e poniamole contemporaneamente uguali a 0.

Otterremo in questo caso un sistema lineare di tre equazioni che porterà alla terna di valori $(x = -\frac{2}{5}, y = \frac{6}{5}, \lambda = \frac{4}{5})$. Si trova quindi il punto $x^* = (-\frac{2}{5}, \frac{6}{5})$ e conseguentemente $f(x^*) = -\frac{16}{5}$.

Data la convessità del problema, non è necessario calcolare l'Hessiana per scoprire l'entità del punto trovato, poiché si sa trattarsi di un minimo globale.

Capitolo 3

Algoritmi di ricerca iterativa

3.1 Ottimizzazione non vincolata

Quando il problema di ottimizzazione che dobbiamo risolvere è privo di vincoli, ovvero si presenta nella forma

$$min \quad f(x),$$

dove $f : \mathbf{R}^n \to \mathbf{R}$ è convessa e due volte differenziabile allora, a patto di assumere che il problema sia risolvibile e quindi $\exists x^* : f(x^*) = p^* = inf_{x \in dom(f)} f(x)$, allora una condizione necessaria e sufficiente affinché x^* sia punto ottimale è

$$\nabla f(x^*) = 0.$$

Risolvere questo sistema di equazioni è possibile per via analitica solo se la funzione possiede forme e simmetrie particolari, ma nella maggior parte dei casi la soluzione passa attraverso l'introduzione di una sequenza di punti $x^{(0)}, x^{(1)}, \ldots, x^{(k)}$, con $x^{(i)} \in dom(f) \;\; \forall i = 0, 1, \ldots, k$, e tale che $f(x^{(k)}) \to p^*$ per $k \to +\infty$.

Il punto da cui si comincia ad iterare viene scelto in maniera spesso randomica o comunque euristica, facendo considerazioni più o meno ragionate su un possibile intervallo nel quale si stima debba trovarsi il minimo, ma questo deve rispettare la condizione di appartenere al dominio della funzione e l'insieme
$S = \{x \in dom(f) \mid f(x) \leq f(x^{(0)})\}$, che rappresenta tutti i sottolivelli della funzione a partire da quello comprendente il punto $x^{(0)}$, deve essere chiuso.

Per ragioni pratiche di implementazione, gli algoritmi che trovano il punto ottimale iterando uno spostamento lungo punti sempre più vicini al valore ricercato, non si interrompono quando il gradiente è esattamente uguale a zero, ma piuttosto quando questo è molto piccolo o quando $f(x^{(k)}) - p^* \leq \epsilon$, con $\epsilon > 0$ molto piccolo. Tuttavia quest'ultima condizione non è quasi mai usata in quanto non si conosce a priori il valore di p^*. Assumeremo quindi che, ad ogni iterazione, il gradiente possa essere molto piccolo ma mai uguale a 0. Inoltre assumeremo che la funzione obiettivo considerata sia strettamente convessa, ovvero che

$$(1) \quad \nabla^2 f(x) \succeq mI,$$

con $m > 0$ e $I \in \mathbf{R}^{n \times n}$ matrice identità; imponiamo quindi che la funzione abbia sempre una curvatura minima. Si noti che per $m = 0$ si ottiene la definizione 2.2.(4) di funzione convessa.

Questa condizione ci permette di essere sicuri della convergenza al valore ottimale e ci fornisce un upper bound del gradiente in funzione di ϵ arbitrario. Infatti operando l'espansione di Taylor del secondo ordine nel punto $x^{(i)}$ della generica funzione strettamente convessa $f(x) : \mathbf{R}^n \to \mathbf{R}$ si ha

$$f(x) \approx f(x^{(i)}) + \nabla f(x^{(i)})^T (x - x^{(i)}) + \frac{1}{2}(x - x^{(i)})^T \nabla^2 f(x^{(i)})(x - x^{(i)}).$$

Ora, ricordando che stiamo cercando il minimo della funzione obiettivo $f(x)$, deriviamo la parte destra della disequazione rispetto a x e poniamo il tutto uguale a 0. Otteniamo il valore
$\tilde{x} = x^{(i)} - \frac{\nabla f(x^{(i)})}{m}$ che minimizza l'espressione. Sostituendo quindi ad x il valore $\tilde{x}$ trovato si arriva a

$$f(x) \geq f(x^{(i)}) - \frac{1}{2m} \|\nabla f(x^{(i)})\|_2^2.$$

Ipotizzando che il problema sia risolvibile, possiamo porre $f(x) = p^*$, presupponendo di trovarci quindi all'ultima iterazione ($i = k$). Allora /

$$f(x^{(k)}) - p^* \leq \frac{1}{2m} \|\nabla f(x^{(k)})\|_2^2.$$

Ponendo ora arbitrariamente in questa ultima iterazione $\|\nabla f(x^{(k)})\|_2 \leq \sqrt{2m\epsilon}$ si ritorna al punto dal quale eravamo partiti ovvero

$$f(x^{(k)}) - p^* \leq \epsilon,$$

ottenendo quindi una relazione diretta tra gradiente e soluzione del problema di ottimizzazione, con m unica incognita in quanto ϵ può essere scelto piccolo a piacere.

Come anticipato nell'introduzione, nei paragrafi successivi saranno illustrati due dei più importanti metodi iterativi di ricerca di punti ottimali, entrambi appartenenti alla famiglia degli *Steepest descent methods,* ovvero il Metodo del Gradiente e il Metodo di Newton. Prima di presentarli nel dettaglio vediamo le loro caratteristiche comuni.

Il passo iterativo della sequenza minimizzante viene effettuato nel seguente modo:

$$x^{(i+1)} = x^{(i)} + t^{(i)}\Delta x^{(i)} \qquad \forall\ i = 0, \ldots, k,$$

dove $\Delta x^{(i)}$ è chiamata direzione di ricerca e $t^{(i)} \geq 0$ è chiamato passo di discesa ed è uno scalare che indica di quanto ci si sposterà lungo la linea $\{x + t\Delta x\}$.

Quando $\|\nabla f(x^{(k)})\|_2 \leq \epsilon'$, con $\epsilon' > 0$ scelto a priori, allora $t^{(k)} = 0$ e il processo si interrompe.

La direzione di ricerca è logicamente basata sulle informazioni che si possiedono localmente in un intorno di $x^{(i)}$.

La condizione primaria affinché si parli di metodo di discesa è che si stia effettivamente scendendo ad ogni iterazione lungo la funzione. Quindi occorre avere

$$\nabla f(x^{(k)})^T \Delta x^{(k)} < 0.$$

Il passo di discesa risulta essere importante poiché andare troppo "avanti" lungo la funzione potrebbe significare mancare il valore ottimale o, nel caso in cui la funzione obiettivo non sia convessa, finire in un altro minimo locale; d'altra parte spostarsi di poco comporta perdite in termini di efficienza computazionale. Dal passo dipende la velocità di convergenza dell'algoritmo.

Vedremo che i due metodi differiscono proprio per il calcolo del passo di discesa.

3.2 Metodo della Discesa del Gradiente

Dato l'algoritmo iterativo

$$x^{(i+1)} = x^{(i)} + t^{(i)}\Delta x^{(i)} \qquad i = 0, \ldots, k,$$

si ponga $\Delta x^{(i)} = -\nabla f(x^{(i)})$ e si noti che $-\nabla f(x^{(i)})^T \nabla f(x^{(i)}) = -\|\nabla f(x^{(i)})\|_2^2 \leq 0$; quindi sicuramente la condizione di discesa è rispettata.

Il passo di discesa può essere calcolato in due modi, e in ogni caso va effettuata una ricerca lineare lungo la semiretta $\{x^{(i)} - t\nabla f(x^{(i)})\}$, con $x \in \mathbf{R}^n$ e $t \in \mathbf{R}_+$.

Il primo è chiamato Ricerca Lineare Esatta e calcola il passo t ad ogni iterazione nel seguente modo:

$$(2) \qquad t = argmin_{s \geq 0} f(x^{(i)} - s\nabla f(x^{(i)})).$$

Viene usato quando minimizzare la funzione in una variabile costa molto meno che calcolare la direzione di ricerca.

Un esempio è il seguente:

$$min \quad f(x, y) = x^2 - xy + y^2,$$

con condizione di uscita dal ciclo $\|\nabla f(x, y)\|_2 \leq 0,5$ e punto iniziale $x^{(0)} = (1, \frac{1}{2})$.

Calcoliamo il gradiente della funzione $\nabla f(x, y) = (2x - y,\ 2y - x)$ e, notando che si tratta di equazioni lineari. Calcoliamo la matrice Hessiana per avere informazioni sulla convessità della funzione.

$$\nabla^2 f(x, y) = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix},$$

che ha due autovalori positivi $\lambda_1 = 1$ e $\lambda_2 = 3$ ed è quindi definita positiva. Si deduce che la funzione è convessa per la 2.2.(4).

Alla prima iterazione si ha $\nabla f(x^{(0)}) = (\frac{3}{2}, 0) \implies \|\nabla f(x^{(0)}\|_2 = 1,5 < 0,5$.

Iterando si ottiene

$$x^{(1)} = x^{(0)} - t\nabla f(x^{(0)})$$
$$x^{(1)} = (1 - \frac{3}{2}t,\ \frac{1}{2}).$$

Usando la 3.1.(2) si ha

$$f(x^{(0)} + t\Delta x^{(0)}) = f(x^{(1)}) = (1 - \frac{3}{2}t)^2 - \frac{1}{2}(1 - \frac{3}{2}t) + (\frac{1}{2})^2$$
$$\frac{\mathrm{d}f}{\mathrm{d}t}(t) = 0 \implies t = \frac{1}{2}.$$

Il punto conclusivo della prima iterazione, ovvero $x^{(1)} = x^{(0)} - \frac{1}{2}\nabla f(x^{(0)}) = (\frac{1}{4}, \frac{1}{2})$, sarà il punto da cui partirà la seconda iterazione a meno che non sia soddisfatta la condizione di break. Il calcolatore eseguirà quindi il controllo $\|\nabla f(x^{(1)})\|_2 = 0,75 > 0,5$.

Dopo una sola altra iterazione si arriverà a $x^{(2)} = (\frac{1}{4}, \frac{1}{8})$ e , individuando in $x^{(2)}$ la soluzione il ciclo si chiuderà dopo sole 3 iterazioni.

Il secondo è chiamato Ricerca Lineare a Ritroso, in inglese backtracking line search. La ricerca viene eseguita previa l'introduzione di due costanti α, β, con $0 < \alpha < 0.5$ e $0 < \beta < 1$ e l'idea alla base è quella di scegliere $t = \beta^k$, dove $k = min\{i \in \mathbf{N} \mid f(x + \beta^i\Delta x) \leq f(x) + \beta^i\alpha\nabla f(x)^T\Delta x\}$.

L'algoritmo è il seguente

$$\Delta x = -\nabla f(x), \quad x \in dom(f)$$
$$\text{choose } \alpha \in (0, 0.5), \beta \in (0, 1)$$
$$t := 1$$
$$while(f(x + t\Delta x) > f(x) + \alpha t\nabla f(x)^T\Delta x) \text{ do}$$
$$t := \beta t.$$

Come si può vedere dalla figura sottostante, la pendenza della retta considerata diminuisce sempre di più scalata del fattore β ad ogni iterazione fino a che l'immagine della funzione lungo il segmento rimane al di sotto del segmento stesso.

Per rendere l'idea è stata introdotta la funzione $g(t) = f(x + t\Delta x)$; il metodo

sceglie t come il primo β^k che sta a destra del punto P.

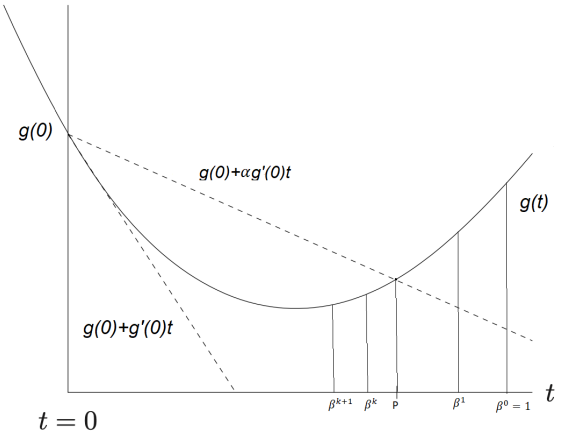


Figura 3.1

Valori tipici di α sono compresi tra 0.01 e 0.3, mentre per β si varia da 0.1 che corrisponde a una ricerca molto "veloce" fino a 0.8 che corrisponde a iterazioni più precise.

Scegliendo come direzione di ricerca l'antigradiente e combinando la 3.1.(1) con l'upper bound $\nabla^2 f(x) \preceq MI$, con $M > m > 0$, si arriva a $mI \preceq \nabla^2 f(x) \preceq MI$. Assumendo quindi stretta convessità della funzione è possibile dimostrare la convergenza dell'algoritmo nel caso di utilizzo di entrambi i metodi di ricerca lineare.

Infatti se consideriamo l'uso della ricerca esatta e considerando la funzione $\hat{f} : \mathbf{R} \to \mathbf{R}$, $\hat{f}(t) = f(x - t\nabla f(x))$ allora sicuramente si parte da

$$\hat{f}(t) \leq f(x) - t\|\nabla f(x)\|_2^2 + \frac{Mt^2}{2}\|\nabla^2 f(x)\|_2^2.$$

Ipotizziamo di trovare t^*, ovvero il passo che minimizza la funzione obiettivo lungo il segmento, e conseguentemente $\hat{f}(t^*)$; successivamente si trova $t^{**} = argmin_{t^*}(f(x) - t^*\|\nabla f(x)\|_2^2 + \frac{M(t^*)^2}{2}\|\nabla^2 f(x)\|_2^2) = \frac{1}{M}$ e $f(t^{**}) = f(x) - \frac{1}{2M}\|\nabla f(x)\|_2^2$, derivando la parte destra della disequazione rispetto a t, ponendo uguale a 0 e sostituendo.

Si ottiene quindi

$$f(x^{(i+1)}) = f(t^*) \leq f(x^{(i)}) - \frac{1}{2M}\|\nabla f(x)\|_2^2;$$

Sottraendo il valore ottimale p^* a entrambi i membri e aggiungendo il fatto che $\|\nabla f(x)\|_2 \geq \sqrt{2m\ \epsilon^{(i)}}$, con $\epsilon^{(i)} = f(x^{(i)}) - p^*$, ovvero che vi sia un lower bound del gradiente in funzione del lower bound dell'Hessiana si arriva a

$$f(x^{(i+1)}) - p^* \leq \frac{M-m}{M}(f(x^{(i)}) - p^*).$$

Iterando la disequazione rispetto alla sequenza minimizzante $x^{(0)}, x^{(1)}, \ldots, x^{(k)}$ si ha

$$(3) \qquad f(x^{(k)}) - p^* \leq c^k(f(x^{(0)}) - p^*),$$

con $c = \frac{M-m}{M} < 1$. Questo implica che $f(x^{(k)}) \to p^*$ per $k \to +\infty$.

Dalla 3.2.(3) si ottiene anche un upper bound al numero di iterazioni ovvero $k \leq log(\frac{f(x^{(0)})-p^*}{\epsilon})/log(\frac{1}{c})$.

La figura 3.2 mostra un possibile percorso iterativo dell'algoritmo della discesa del gradiente con la tecnica di backtracking line search della funzione $f : \mathbf{R}^2 \to \mathbf{R} \quad f(x,y) = e^{x+3y-0.1} + e^{x-3y-0.1} + e^{-x-0.1}$. Si noti che la funzione è sicuramente strettamente convessa in quanto somma di funzioni strettamente convesse.

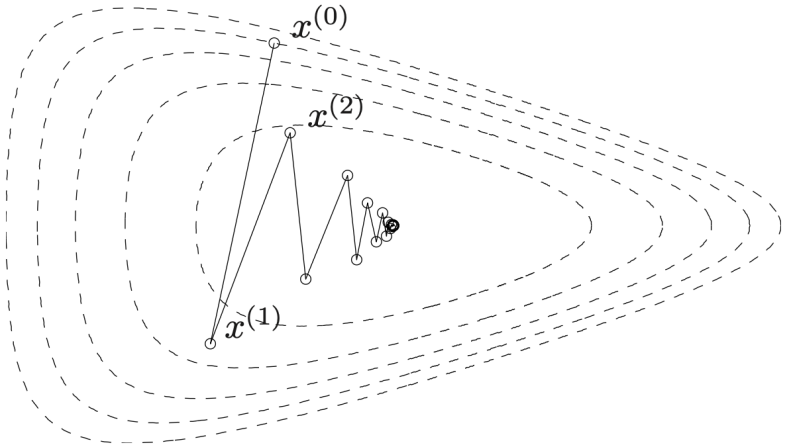


Figura 3.2

La figura sottostante mostra invece, per la stessa funzione, un confronto tra il numero di iterazioni svolte da un algoritmo di ricerca esatta e uno di backtracking; nell'asse delle ordinate è mostrata la variazione dell'errore.

Notiamo che per entrambe la decrescita è all'incirca lineare rispetto alla scala logaritmica dell'errore, ma la ricerca esatta risulta avere una pendenza doppia rispetto al backtracking e un conseguente numero di iterazioni dimezzato.

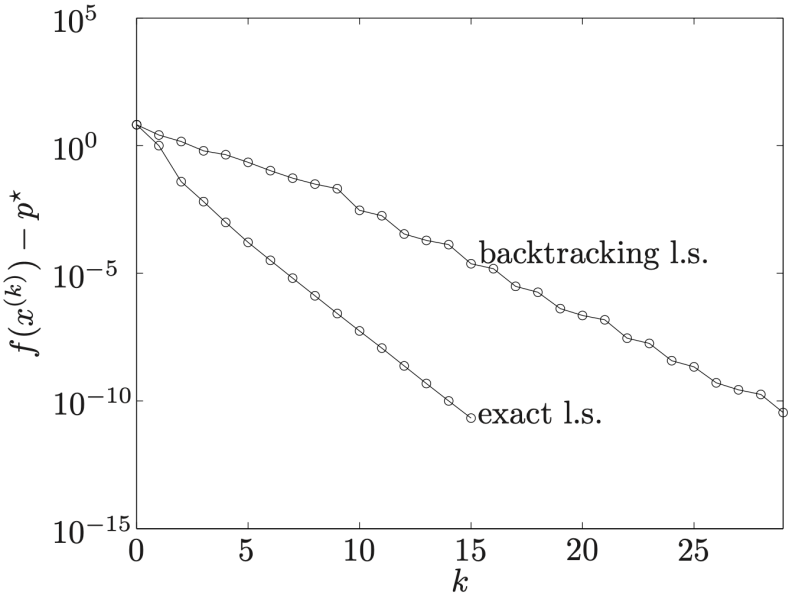


Figura 3.3

La ricerca lineare esatta è sicuramente più efficiente quando il numero di variabili è basso, ma vedremo che la backtracking search può essere conveniente quando le variabili in gioco sono molte poiché il calcolo del passo è meno pesante in termini di complessità.

3.3 Metodo di Newton

Il metodo di Newton, nato come metodo di ricerca degli zeri in un polinomio di grado 3 o superiore, fu introdotto da Isaac Newton nel 1669 e successivamente nominato Metodo delle tangenti in quanto l'idea alla base è quella di approssimare in ogni punto la funzione con la retta tangente a essa in quel punto sfruttando l'espansione di Taylor del prim'ordine, per trovare lo zero della linearizzata dal quale partirà l'iterazione successiva.

Il suo utilizzo nel campo dell'ottimizzazione è più recente e prevede di andare a scovare gli zeri del gradiente della funzione con la stessa idea di base; il risultato sarà un approssimazione di Taylor troncata al termine di grado 2 della funzione in ogni

punto, e l'utilizzo quindi della matrice Hessiana per iterare lo spostamento verso punti $x^{(i)}$ con immagini $f(x^{(i)})$ sempre minori delle precedenti.

Vediamo la formalizzazione di questa idea.

Dato la classica relazione iterativa,

$$x^{(i+1)} = x^{(i)} + t^{(i)}\Delta x^{(i)} \qquad i = 0, \ldots, k,$$

è chiamato Passo di Newton il valore

$$\Delta x_{ns} = \Delta x^{(i)} = -\nabla^2 f(x^{(i)})^{-1}\nabla f(x^{(i)}),$$

questo $\forall x^{(i)} \in dom(f)$.

Ipotizzando che la funzione sia convessa, per la 2.2.(4) si ottiene

$$\nabla f(x)^T \Delta x_{ns} = -\nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x) < 0,$$

che soddisfa la condizione di discesa.

Considerata la funzione obiettivo $f : \mathbf{R}^n \to \mathbf{R}$ e il punto generico $x \in dom(f) \subseteq \mathbf{R}^n$, operando l'espansione di Taylor del secondo ordine all'i-esima iterazione e ipotizzando momentaneamente $t = 1$ si ha

$$f(x + \Delta x) \approx f(x) + \nabla f(x)^T \Delta x + \frac{1}{2}\Delta x^T \nabla^2 f(x) \Delta x.$$

Volendo ora minimizzare l'espressione del paraboloide appena ottenuta rispetto a Δx si ottiene proprio $\Delta x = \Delta x_{ns}$.

Lo stesso risultato si ottiene, banalmente, se linearizziamo

$$\nabla f(x + \Delta x) \approx \nabla f(x) + \nabla^2 f(x) \Delta x$$

e poniamo il tutto uguale a 0.

La figura sottostante mostra chiaramente l'idea alla base dell'algoritmo. Si parte infatti dal punto $(x, f(x))$, grazie all'espansione di Taylor si ottiene $\hat{f}$, funzione che approssima f all'ordine 2, e si trova $x + \Delta x_{nt}$ tale che $\nabla \hat{f}(x + \Delta x_{nt}) = 0$.

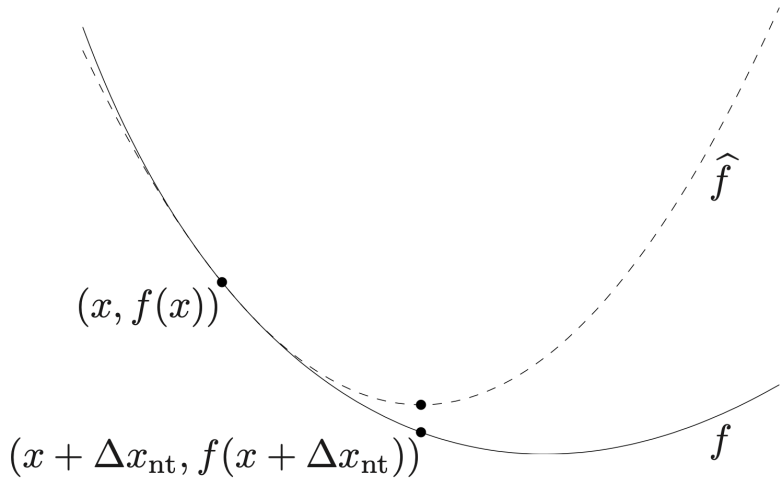


Figura 3.4

L'iterazione successiva partirà dunque da $x' = x + \Delta x_{nt}$.

Allo scopo di costruire una buona condizione di uscita dal loop e di analizzare la convergenza del metodo, si introduce la funzione ausiliaria

$$\lambda(x) = (\nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x))^{1/2},$$

che possiamo mettere in relazione con la quantità $f(x) - inf_{\Delta x} f(x + \Delta x) = f(x) - f(x + \Delta x_{ns}) = \frac{1}{2}\lambda(x)^2$, che indica quindi il decremento subito dalla funzione nel passo iterativo $x \to x + \Delta x$. Questa informazione ci fornisce una ottima condizione di uscita in quanto, ipotizzando di essere sempre più vicini a x^*, soluzione del problema di ottimizzazione, il decremento sarà sempre minore e quindi, scegliendo $\epsilon > 0$ piccolo a piacere, si deduce che l'algoritmo dovrà interrompersi quando $\frac{1}{2}\lambda^2 \leq \epsilon$.

Da notare è anche che $\nabla f(x)^T \Delta x_{ns} = -\lambda(x)^2$, e quindi la funzione ausiliaria può darci informazioni anche circa la velocità di convergenza del metodo.

Lo pseudocodice risulta essere quindi

$$x_0 \in dom(f) \subseteq \mathbf{R}^n, \quad i = 0, \quad \epsilon > 0$$
$$\lambda^2 = \nabla f(x_i)^T \nabla^2 f(x_i)^{-1} \nabla f(x_i);$$
$$while\ (\lambda^2/2 > \epsilon)\ \{$$
$$\Delta x_{ns} = -\nabla^2 f(x_i)^{-1} \nabla f(x_i);$$
$$\lambda^2 = \nabla f(x_i)^T \nabla^2 f(x_i)^{-1} \nabla f(x_i);$$
$$x_{i+1} = x_i - \nabla^2 f(x_i)^{-1} \nabla f(x_i);$$
$$i++; \quad \}.$$

Da notare che è stato posto $t = 1$, per tutta la durata dell'algoritmo. In effetti non vi sono grossi problemi a non effettuare il calcolo del passo ad ogni iterazione con il Metodo di Newton. L'efficienza non risente molto di questa scelta.

Tuttavia è utile distinguere tra due periodi di convergenza all'interno dell'algoritmo, che ora andremo ad esplorare tramite il seguente esempio.

$$min \quad -\sum_{i=1}^{n} log(1 - x_i^2) - \sum_{i=1}^{m} log(b_i - a_i^T x),$$

con $n = 10^4$, $m = 10^5$ e a_i vettori sparsi, ovvero dove gran parte delle sue componenti sono nulle. Il problema si svolge quindi in $\mathbf{R}^{10000}$.

La figura 3.5 mostra come in appena 18 iterazioni l'algoritmo raggiunge la soluzione con un'accuratezza dell'ordine di 10^{-7}.

Risulta evidente un primo periodo, in cui la convergenza è lineare per le prime 13 iterazioni, e dove nell'esempio preso in esame viene usata una backtracking line search con $\alpha = 0.01$ e $\beta = 0.5$, che corrisponde ad una ricerca ad accuratezza media. Quando l'errore diventa piccolo, e si è quindi vicini al valore ottimale x^*, la convergenza diventa quadratica e da quel momento bastano 4 o 5 iterazioni per arrivare al risultato desiderato.

La figura sottostante mostra invece un confronto tra l'utilizzo della ricerca esatta

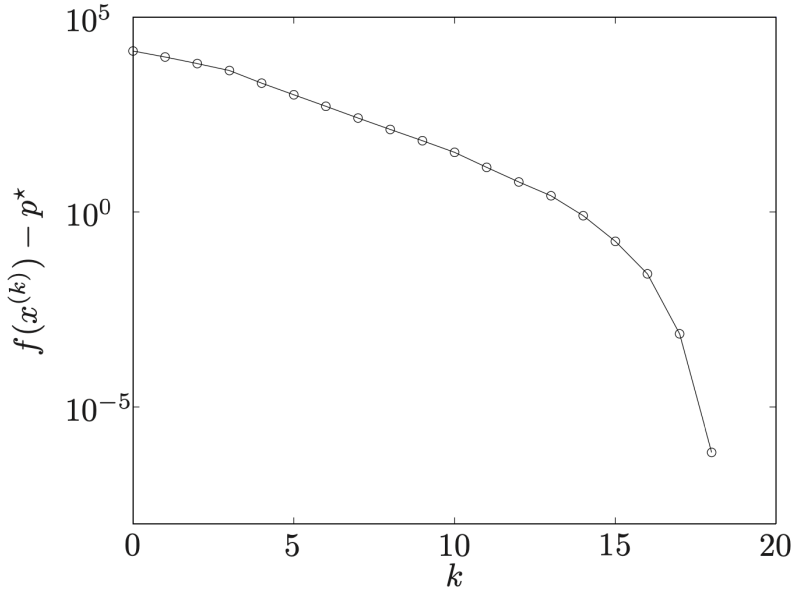


Figura 3.5

e della ricerca a ritroso in un problema in $\mathbf{R}^{100}$ nella forma

$$min \quad f(x) = c^T x - \sum_{i=1}^{m} log(b_i - a_i^T x),$$

con $m = 500$.

La backtracking line search è settata ai parametri $\alpha = 0.01$ e $\beta = 0.5$.

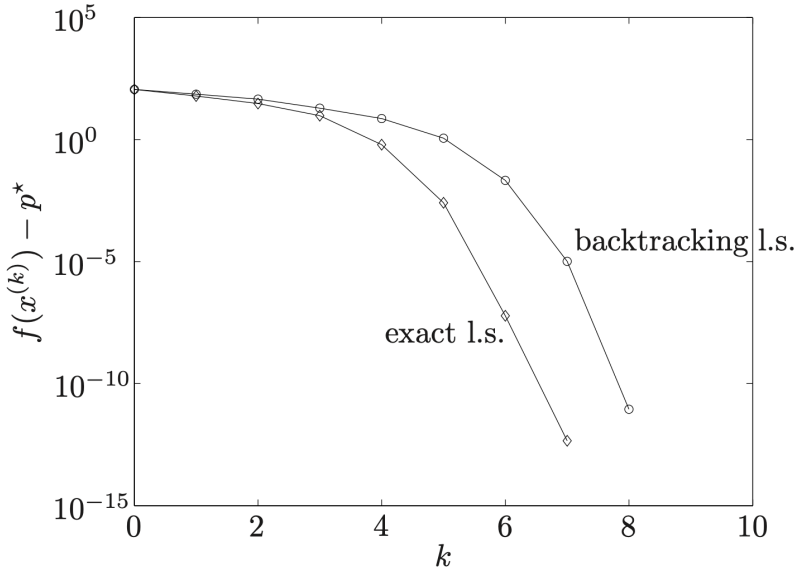


Figura 3.6

Vediamo come si abbia inizialmente, come nell'esempio precedente, convergenza lineare per poi all'incirca dimezzare il numero di iterazioni necessarie per raggiungere

il risultato. Si noti inoltre come i due metodi di ricerca lineare differiscono di una sola iterazione, con una leggera convenienza nell'uso della ricerca esatta.

Nel periodo di convergenza quadratica il passo t diviene quasi sempre unitario, per questo motivo spesso la ricerca lineare non viene adottata quando si implementa il metodo di Newton.

Capitolo 4

Conclusioni

In questa tesi ho presentato una introduzione all'ottimizzazione convessa, che è un ramo molto importante dell'ottimizzazione matematica.

I suoi principali vantaggi sono la certezza dell'esistenza di un unico punto di minimo globale.

Inoltre anche se il problema iniziale risulta non convesso, è spesso possibile considerare sottoproblemi convessi, trovare una direzione di ricerca e quindi cercare una soluzione con approccio bottom-up.

La dualità di Lagrange è un ottimo strumento per affrontare un problema di ottimizzazione vincolata. Infatti grazie a questa è sempre possibile arrivare ad un lower bound per il valore ottimale. In particolare se la funzione obiettivo è convessa porta alla soluzione esatta.

Risulta cruciale quindi trovare tracce di convessità in un problema di ottimizzazione prima di passare ad implementazioni più rischiose e meno efficienti.

Per concludere il percorso fatto, ritengo indispensabile un confronto diretto tra il Metodo della Discesa del Gradiente e il Metodo di Newton, dal punto di vista del numero di iterazioni e del costo computazionale, allo scopo di fornire al lettore delle linee guide su quando risulta meglio utilizzare uno piuttosto dell'altro.

Se consideriamo l'ultimo esempio del capitolo 3, ovvero

$$minimizzo \quad f(x) = c^T x - \sum_{i=1}^{m} log(b_i - a_i^T x),$$

con $m = 500$, ebbene, se provassimo ad implementare la soluzione con il metodo del gradiente combinato ad una ricerca lineare a ritroso con parametri $\alpha = 0.1$ e $\beta = 0.5$, otterremmo una convergenza lineare inizialmente più rapida e poi più lenta man mano che ci si avvicina al valore di x^*, per un totale di circa 175 iterazioni eseguite per raggiungere un errore dell'ordine di 10^{-4}, a fronte però di un costo computazionale ridotto in quanto l'algoritmo risulta essere molto semplice essendo l'unico step critico il calcolo del gradiente ad ogni iterazione, oltre alla ricerca lineare che solitamente è molto veloce soprattutto se si utilizza la backtracking line search.

Come si vede dalla figura 3.6, invece, il Metodo di Newton raggiunge la soluzione approssimata all'ordine di 10^{-10} in circa 8 iterazioni. Infatti l'efficienza del problema varia poco al variare del numero delle variabili, poiché una volta raggiunto il punto in cui si inizia ad avere convergenza quadratica spesso sono sufficienti 6 o 7 iterazioni per produrre una soluzione desiderata.

Questo è il motivo per cui solitamente il Metodo di Newton è utilizzato quando il numero delle variabili in gioco è elevato. Occorre però prestare attenzione ai problemi che si hanno quando si incorre nel calcolo della matrice Hessiana, soprattutto in quanto questa va poi invertita, rendendo l'algoritmo non solo costoso ma anche rischioso nel qual caso siano coinvolti numeri con parti decimali importanti.

Questo problema può essere però bypassato quando la matrice Hessiana possiede forme e simmetrie particolari che rendono più leggero il calcolo e con l'aiuto di strumenti quali la fattorizzazione matriciale.

Spesso nell'ambito dell'ottimizzazione si ha a che fare con problemi la cui soluzione porta a dover risolvere un sistema lineare del tipo $Ax = b$. Questo accade nel metodo di Newton e anche nella programmazione lineare, dove lo scopo è appunto ridursi ad un sistema lineare maneggiando i vincoli del problema iniziale.

Questo si fa perché anche se le variabili in gioco sono innumerevoli, esistono spesso algoritmi efficienti e di facile implementazione che risolvono il problema con un costo dell'ordine di n^2.

Ecco perché rendere più efficiente il calcolo di un sistema lineare può rivelarsi fondamentale, in quanto diminuisce il costo dell'algoritmo ed evita di dover invertire le matrici per trovare la soluzione.

Se consideriamo infatti il calcolo della soluzione x dell'equazione $Ax = b$, con

$A \in \mathbf{R}^{n \times m}$, $x \in \mathbf{R}^n$ e $b \in \mathbf{R}^m$, allora si può dimostrare che la matrice A può essere sempre fattorizzata, a patto che sia non singolare, nel seguente modo

$$PA = LU,$$

con P matrice di permutazione, L matrice lower triangular, ovvero con tutti zeri al di sopra della diagonale, e U upper triangular, con tutti zeri al di sotto della diagonale.

Questa fattorizzazione si ricava avere un costo di $\frac{2}{3}n^3$ operazioni, ma rende molto più semplice il calcolo, che diventa $LUx = Pb$, effettuabile per ricorsione calcolando prima $Ly = Pb$, e poi $Ux = y$. Questi sistemi lineari sono abbastanza semplici in quanto le matrici coinvolte sono semplici.

Ancora migliore risulta essere la fattorizzazione di Cholesky, introdotta intorno al 1900, che richiede che la matrice A sia definita positiva. In tal caso una possibile fattorizzazione è

$$A = PLL^T P^T,$$

con un costo minore di $\frac{1}{3}n^3$ operazioni.

Se la matrice A è sparsa si può evitare la permutazione ed ottenere un efficiente risultato per una implementazione.

Per finire l'ottimizzazione, in quanto branca della matematica relativamente moderna, è in continua evoluzione nell'intelligenza artificiale, dove viene usata per lo studio delle reti neurali programmate con l'apprendimento automatico allo scopo che una macchina prenda le decisioni migliori in ogni momento.

Un ulteriore esempio per me particolarmente interessante è quello di AlphaZero, motore specializzato in apprendimento di giochi da tavoli quali scacchi, dama, shogi ecc...

Venuto alla ribalta nel 2017 grazie a Google DeepMind. I suoi risultati di gioco hanno stupito il mondo scacchistico nel momento in cui si è saputo che su 100 partite contro Stockfish, il motore scacchistico tradizionale, era stato battuto 25 volte coi pezzi neri, 3 con i bianchi e aveva pareggiato nel resto delle partite.

Il codice di AlphaZero non è mai stato divulgato.

Bibliografia

[1] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.