Università di Pisa

Master degree in Computer Science
AI and BDT curricula

Computational Mathematics for Learning and Data Analisys

Anno Accademico 2022/2023

Basate sulle lezioni dei professori Antonio Frangioni e Federico Poloni

Indice

Introduzione

Ho deciso di riassumere in questa dispensa i principali contenuti delle lezioni del professor Antonio Frangioni e del professor Federico Poloni del corso di Computational Mathematics dell' a.a. 2022/2023 del corso di laurea in Artificial Intelligence.

Queste dispense sono intese prima di tutto come uno strumento per aiutare me stesso a studiare, perciò sono possibili e anzi probabili eventuali errori, imprecisioni o typo. Nel caso decida un giorno di pubblicare tali dispense, sono gradite correzioni di tali errori, potete scrivermi alla mail g.lagomarsini@studenti.unipi.it o su telegram boh.

Probabilmente per capire al meglio tutto serve un po' di familiarità dei concetti di un corso di Analisi I, come continuità e derivabilità, e qualche nozione di algebra lineare (comunque nelle loro lezioni i professori ridefiniscono praticamente ogni concetto, soprattuto la parte di Algebra Lineare). Saranno presenti qualche nozione di continuità, derivabilità ecc... in più variabili visto che nella triennale di informatica almeno fino a qualche anno fa non si toccavano tali argomenti.

Siccome il corso è diviso in due sezioni interconnesse ma distinte, ho deciso anch'io di dividere le dispense nelle due parti principali, ovvero quella di Ottimizzazione e quella di Analisi Numerica. Le varie figure presenti nel documento sono state prese per la maggior parte dei casi direttamente dalle slide dei professori. In alcuni casi ho preso alcune figure da internet evitando di citare le fonti tanto non è che lo vendo sto robo.

Grazie a Irene per tutte le correzioni.

Giacomo Lagomarsini

Parte I

Ottimizzazione matematica

Capitolo 1

Ottimizzazione in una dimensione

1.1 Introduzione ai problemi di ottimizzazione

La prima parte delle dispense sarà dedicata ad una trattazione di alcune tecniche di ottimizzazione, ovvero della ricerca di minimi o massimi di funzioni.

Definizione. Dato un insieme X, detto *regione ammissibile* (del problema) e una funzione $f : X \to \mathbb{R}$, un problema di ottimizzazione è trovare

$$(P) \qquad f_* = \min_{x \in X} f(x). \tag{1.1}$$

Osserviamo che se f_* esiste, è unico, ma in generale non è unico x_* tale per cui $f(x_*) = f_*$. In generale possiamo formulare in maniera equivalente il problema come la ricerca del valore x_* che minimizza f, ovvero di

$$(P) \qquad x_* = \arg\min_{x \in X} f(x). \tag{1.2}$$

Osserviamo anche che non è restrittivo supporre di voler cercare il minimo anzichè il massimo: infatti

$$\min_{x \in X} f(x) = -\max_{x \in X} -f(x),$$

perciò se dobbiamo massimizzare un'altra funzione $g(x)$, possiamo minimizzare $-g(x)$. Se f è definita su un insieme più grande F con $X \subset F$, chiamiamo *inammissibili* i punti di $F \setminus X$.

Studieremo funzioni reali in una o più variabili, ovvero funzioni $f : X \subseteq \mathbb{R}^d \to \mathbb{R}$, con $d \geq 1$. Iniziamo a trattare il caso unidimensionale, ovvero il caso in cui $d = 1$. Tale caso è interessante perché come vedremo spesso potremo ridurre singoli passaggi in algoritmi per la risoluzione di casi più complicati, ovvero casi di dimensione maggiore, a problemi di ottimizzazione unidimensionale (detta anche line search).

1.2 Ottimizzazione in una dimensione

1.2.1 Ok I'm gonna optimize these functions... Damn this hard

Anche nel caso più semplice possibile, quello di una funzione $f : \mathbb{R} \to \mathbb{R}$, non è sempre facile trovare un minimo locale. Pensiamo ad una funzione costante in $\mathbb{R} \setminus x_*$, con valore $f(x) = 1$ e con $f(x_*) = 0$: in tale caso se non conosciamo nulla su dove possa trovarsi x_*, è praticamente impossibile trovare tale minimo. La funzione appena descritta ha una discontinuità in x_*, perciò si potrebbe supporre che per funzioni continue la situazione migliori: così non è in alcune situazioni. Immaginiamo per esempio una funzione che " scende" con velocità arbitraria, come nella figura 1.1.

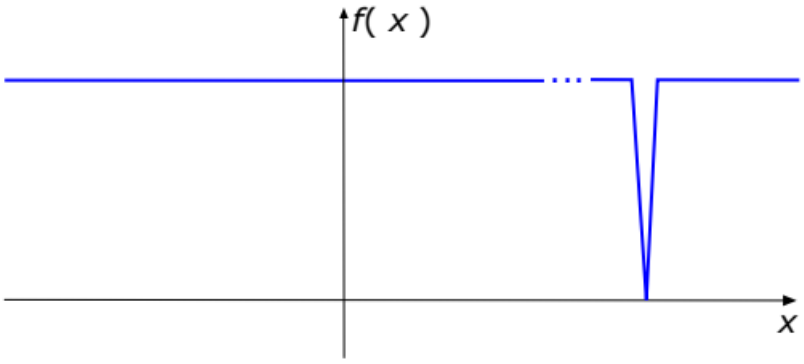

Figura 1.1: Funzione continua molto difficile da pttimizzare :(

Facciamo perciò delle assunzioni sul nostro problema:

- Scegliamo una regione ammissibile chiusa e limitata $X = [x_1, x_2]$: questa ipotesi ci garantisce che esista un minimo, se f è continua (teorema di Weierstrass).

- Cerchiamo una soluzione ϵ-approssimata per qualche $\epsilon > 0$, cioè una soluzione $f(x_\epsilon) \leq f_* + \epsilon$. A volte si sceglie $\epsilon \approx 10^{-16}$, cioè dell'ordine di grandezza della precisione di macchina di un computer, ma spesso ϵ può essere anche molto più grande.

- Assumiamo che la funzione da ottimizzare sia *L-lipschitziana*:

Definizione. Diciamo che una funzione $f : X \to \mathbb{R}$ è lipschitziana con costante di Lipschitz $L > 0$, o in breve L-lipschitz, se per ogni $x, y \in X$ si ha che

$$| f(x) - f(y) | \leq L |x - y|.$$

Questa definizione formalizza il fatto che se due punti sono molto vicini anche le loro immagini saranno ragionevolmente vicine.

Con queste ipotesi è facile trovare un algoritmo per trovare una soluzione ϵ-approssimata: basta campionare uniformemente l'intervallo prendendo punti a distanza sempre uguale e minore o uguale a ϵ/L.

Osserviamo che in questo modo ci sono circa DL/ϵ punti da estrarre, dove D è la lunghezza dell'intervallo X iniziale. Perciò dobbiamo calcolare f in $O\left(\frac{DL}{\epsilon}\right)$ punti. In pratica perciò l'algoritmo non è per niente efficiente: è infatti esponenziale in $\log(1/\epsilon)$, cioè nel numero di cifre significative dell'approssimazione Si può dimostrare però che, se assumiamo solamente queste ipotesi, nessun altro algoritmo può fare meglio di così su ogni istanza.

1.2.2 Minimi locali

Fin'ora abbiamo cercato di trovare il minimo *globale* di una funzione. Abbassiamo un po' le nostre pretese e cerchiamo un punto di minimo *locale* di f:

Definizione. Un punto di minimo locale di f è x_* tale che esiste $\delta > 0$ tale per cui per ogni altro x in $(x_* - \delta, x_* + \delta)$, $f(x_*) \leq f(x)$.

Se x_* è un punto di minimo locale, "tipicamente" (ma non sempre!) esiste un δ' (in generale non lo stesso nella definizione di minimo locale) tale per cui f è *unimodale* in $(x_* - \delta', x_* + \delta')$:

Definizione. f è unimodale attorno a x_* se esistono due punti $x_- \leq x_* \leq x_+$ tale per cui f è decrescente in $[x_-, x_*]$ e crescente in $[x_*, x_+]$.

L'intervallo $[x_-, x_+]$ della definizione di funzione unimodale è detto *bacino di attrazione*. In un bacino di attrazione abbiamo un algoritmo per trovare un minimo approssimato molto semplice: dato $X = [x_-, x_+]$,

1. Scegliamo due punti x'_- e x'_+, con $x_- < x'_- < x'_+ < x_+$.
2. Se $f(x'_-) < f(x'_+)$ scartiamo $[x'_+, x_+]$, ovvero reiteriamo su $[x_-, x'_+]$.
3. Altrimenti reiteriamo su $[x'_-, x_+]$.

La domanda che ci poniamo subito è come scegliere x'_- e x'_+? Se D è la lunghezza dell'intervallo, possiamo scegliere $r \in (\frac{1}{2}, 1)$ e prendere $x'_- = x'_- + (1-r)D$, $x'_+ = x_+ - (1-r)D$. In tal modo ad ogni passo la lunghezza dell'intervallo passa da D a rD. Perciò l'intuizione ci dice di prendere r molto vicino a $\frac{1}{2}$ per andare il più veloce possibile: in questo modo, ad ogni iterazione, la lunghezza dell'intervallo viene dimezzata.

Golden ratio search

Se vogliamo minimizzare il numero di valutazioni della funzione f, possiamo altrimenti prendere come uno dei due x'_- e x'_+ il rimanente tra i due punti. In questo caso deve valere che, nel nuovo intervallo di lunghezza rD, il punto rimanente tra i due diventi uno dei due nuovi punti mediani scelti (se abbiamo scartato $[x_-, x'_-]$, il vecchio x_+ diventerà il nuovo x'_- e viceversa) (Figura 1.2). Vogliamo cioè che valga la proporzione

$$1 : r = r : 1 - r,$$

perciò si trova che r deve risolvere l'equazione

$$r^2 + r - 1 = 0$$

Otteniamo perciò

$$r = \frac{\sqrt{5} - 1}{2} = \frac{1}{g} = 1 - g$$

dove $g = \frac{\sqrt{5}+1}{2}$ è la sezione aurea. Tale metodo di ricerca si chiama *golden ratio search*.

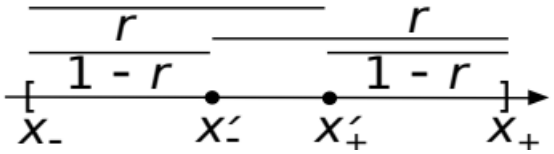

Figura 1.2: Golden Ratio Search

Quando l'intervallo è più piccolo di $\delta = \frac{\epsilon}{L}$, dove L è la costante di Lipschitz della funzione, siamo sicuri che un qualsiasi punto nell'intervallo ci fornisca una soluzione ϵ- approssimata. Questo algoritmo fa un numero $O(\log \frac{1}{\epsilon})$ di iterazioni: ha cioè complessità lineare nel numero di cifre significative che richiediamo all'approssimazione.

1.2.3 Modello del primo ordine

Analizziamo ora un metodo che tiene conto della derivata. Ricordiamo preliminarmente il teorema di Taylor:

Teorema 1.2.1. (Polinomio di Taylor) *Sia $f : X \to \mathbb{R}$ una funzione derivabile n volte in $x_0 \in X$. Chiamiamo $f^{(k)}$ la k-esima derivata di f in x_0, e denotiamo $f^{(0)}(x_0) = f(x_0)$. Allora vale per ogni $x \in X$:*

$$f(x) = \sum_{k=0}^{n} f^{(k)}(x_0)\frac{(x - x_0)^k}{k!} + o((x - x_0)^n)$$

In pratica, possiamo approssimare la funzione in un punto x vicino a x_0 con un polinomio i cui coefficienti sono dati dalle derivate successive di f, e l'errore che commettiamo in tale approssimazione è più piccolo della distanza tra x e x_0 elevata alla n.

Per $n = 1$ si ha che

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + o(x - x_o),$$

perciò la funzione lineare $f(x_0)+f'(x_0)(x-x_0)$ ci fornisce un *modello* (al primo ordine) della funzione f da minimizzare. La derivata in un punto esprime il coefficiente angolare della retta tangente in quel punto: ci dice cioè in quale direzione la funzione sta crescendo e in quale sta decrescendo, e con quale velocità (Figura 1.3). Nei minimi locali in particolare la derivata è sempre nulla (Figura 1.4) , per il noto Teorema di Fermat sui punti stazionari. La

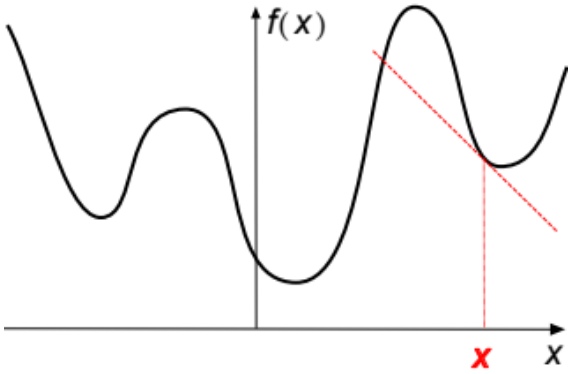

Figura 1.3: La derivata di una funzione in un punto ci dice la pendenza della retta tangente in quel punto

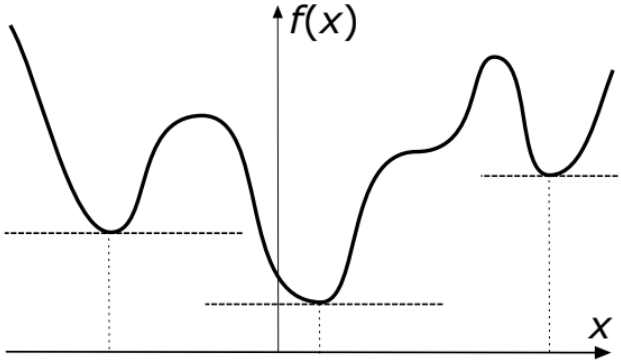

Figura 1.4: La derivata si annulla nei minimi locali

condizione che la derivata si annulli è solo necessaria, ma non sufficiente: la derivata può essere nulla anche in altri punti, come ad esempio nei massimi locali.

Ricerca dicotomica:

L'ultima osservazione ci porta a sviluppare un nuovo algoritmo per minimizzare una funzione:

- Supponiamo innanzitutto che f' sia continua in X, ovvero che $f \in C^1$ (cioè f è derivabile una volta e la sua derivata è continua).

- Supponiamo inoltre di trovare due punti $x_- < x_+$ con $f'(x_-) < -\epsilon$ e $f'(x_+) > \epsilon$.

Allora possiamo procedere in questo modo:

1. Ad ogni passo scegliamo un punto $x_c \in (x_-, x_+)$. Se $-\epsilon < f'(x_c) < \epsilon$, fermiamo il procedimento.
2. Altrimenti se $f'(x_c) < -\epsilon$ ricominciamo con $x_- = x_c$
3. Altrimenti ($f'(x_c) > \epsilon$)ricominciamo con $x_+ = x_c$

Il vantaggio di questo algoritmo è che ad ogni passo dobbiamo fare solo una valutazione di funzione (in questo caso f'). Possiamo inoltre scegliere un passo di esattamente $\frac{1}{2}$, cioè $x_c = (x_- + x_+)/2$, perciò questo metodo ad ogni passo dimezziamo esattamente la lunghezza dell'intervallo considerato

Osservazione 1. Il criterio di stop che usiamo è $|f'(x_c) < \epsilon|$, che ci garantisce che $|f(x_*) - f(x_c)| \leq \epsilon|x_* - x_c| + o(|x_* - x_c|)$, ovvero $f(x_c)$ è ragionevolmente vicino al minimo locale.

1.2.4 Modello del secondo ordine e metodo di Newton

Idea: Scegliere x_c esattamente al centro di $[x_-, x_+]$ non è la scelta più furba da fare: notiamo che stiamo usando infatti solo il segno della derivata nel passo 1. (o meglio il fatto che sia maggiore di ϵ in valore assoluto) e non il suo valore effettivo.

Interpolazione quadratica

Invece che un modello lineare vogliamo ora un modello quadratico p (un polinomio di secondo grado) che approssimi la nostra funzione nel nostro intervallo. Idealmente vorremmo un modello che coincida con f negli estremi x_- e x_+: vorremmo cioè $p(x) = ax^2 + bx + c$, dove a, b, c, sono le nostre incognite, tale che

$$\begin{cases} p(x_-) = f(x_-) \\ p(x_+) = f(x_+) \\ p'(x_-) = f'(x_-) \\ p'(x_+) = f'(x_+). \end{cases}$$

Visto che le incognite sono 3 e le equazioni sono 4, il sistema è "sovradeterminato" (anche se non è un sistema lineare) perciò dobbiamo lasciar cadere qualche ipotesi. In particolare richiediamo solo che $p'(x_-) = f'(x_-)$ e $p'(x_+) = f'(x_+)$. In questo caso quindi

$$\begin{aligned} p'(x_-) &= 2ax_- + b = f'(x_-) \\ p'(x_+) &= 2ax_+ + b = f'(x_+) \end{aligned}$$

da cui otteniamo svolgendo i sistemi che

$$\begin{aligned} a =& \frac{f'(x_+) - f'(x_-)}{2(x_+ - x_-)} \\ b =& \frac{x_+ f'(x_-) - x_- f'(x_-)}{x_+ - x_-}. \end{aligned}$$

Per minimizzare p basta porre la derivata $ax + b = 0$. ciò accade in

$$x = \frac{x_+ f'(x_-) - x_- f'(x_-)}{f(x_+) - f(x_-)}.$$

Se $f'(x)$ è abbastanza piccola ci fermiamo, se no come al solito se $f'(x) < \epsilon$ sostituiamo x a x_- e viceversa se $f'(x) > \epsilon$ sostituiamo x a x_+. Iterando questo procedimento troviamo un punto in cui $f'(x) \approx 0$, e quindi un minimo locale.

Osservazione 2. Il procedimento descritto equivale all'applicare il metodo delle secanti alla *derivata* di f (Figura 1.5).

Metodo di Newton

Un altro modo di trovare un modello del secondo ordine è applicando il Teorema 1.2.1 con $n = 2$:

$$p_{x_0}(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)(x - x_0)^2}{2}$$

per qualche $x_0 \in X$. Questo nuovo modello ci porta al metodo di Newton (o delle tangenti): Partiamo da $x_0 \in X$ e per ogni $i = 0, 1, 2, ...$

- Se $|f'(x_i)| < \epsilon$ ci fermiamo

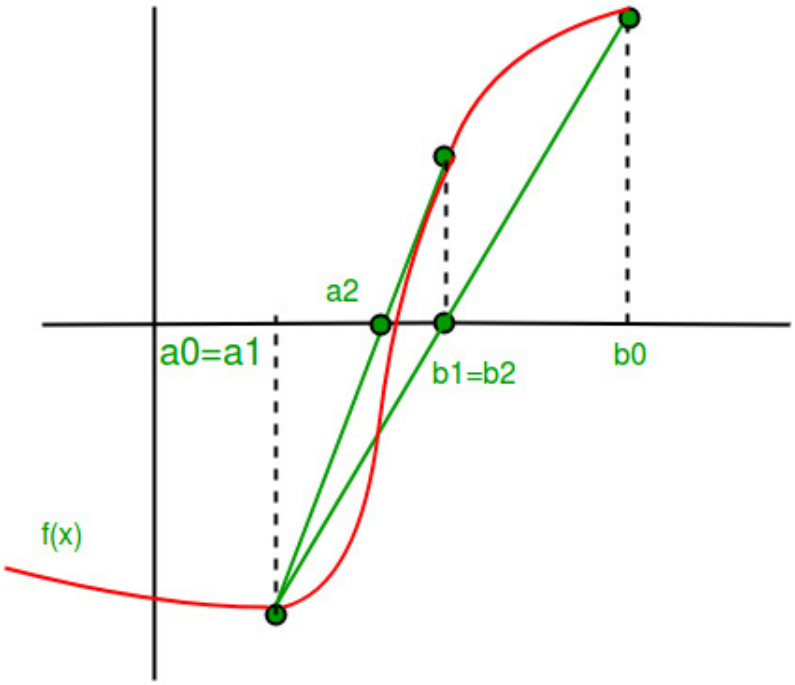

Figura 1.5: Il metodo delle secanti è un semplice metodo per risolvere un'equazione omogenea $g(x) = 0$. Nel nostro caso $g = f'$.

- Altrimenti minimizziamo

$$p_{x_i}(x) = f(x_i) + f'(x_i)(x - x_i) + \frac{1}{2}f''(x_i)(x - x_i)^2$$

.

Tale minimo sarà assunto quando $p'_{x_i}(x) = f'(x_i) + f''(x_i)(x - x_i) = 0$ cioè in

$$x_{i+1} = \frac{x_i - f'(x_i)}{f''(x_i)}.$$

Possiamo quindi ripetere il procedimento finché $|f'(x_i)|$ non è abbastanza piccolo. Questo metodo è detto *metodo di Newton* o *delle tangenti*, poiché stiamo cercando di risolvere l'equazione $f'(x) = 0$ andando a vedere ad ogni passo dove si annulla la retta tangente alla funzione nel punto x_i (Figura 1.6). Il metodo di Newton converge se x_0 è abbastanza vicino al punto di minimo x_*.

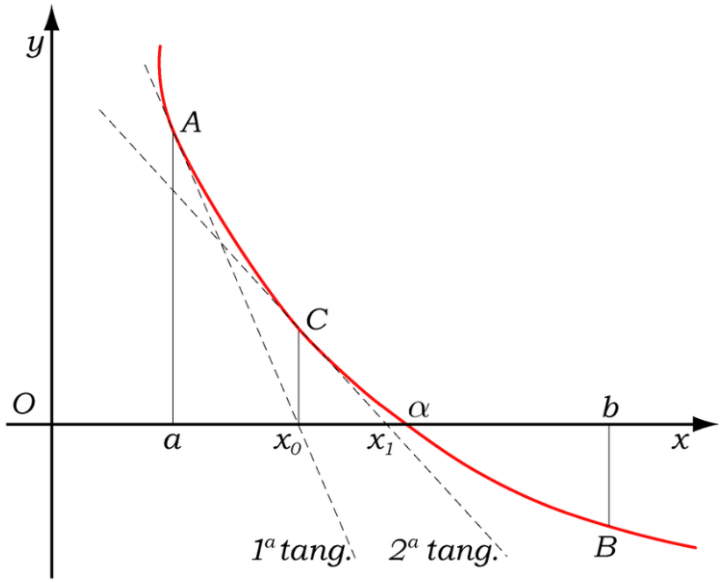

Figura 1.6: Metodo delle tangenti applicato ad una funzione g

1.2.5 Misurare la velocità di convergenza

Iniziamo una breve discussione su come misurare la velocità di convergenza di algoritmi iterativi (tutti gli algoritmi che abbiamo visto fin'ora).

Definizione. (Tasso di convergenza): Sia $f_i = f(x_i)$ la sequenza generata da un algoritmo iterativo e sia f_* il minimo locale della funzione a cui f_i converge (cioè $\lim_{i\to\infty} f_i = f_*$; supponiamo inoltre che esistano delle costanti $0 < r \leq 1$, $t \geq 1$ tali che

$$\lim_{i\to\infty} \frac{f_{i+1} - f_*}{(f_i - f_*)^t} = r$$

Possiamo distinguere fondamentalmente 3 casi:

$t = 1$, $r = 1$**:** In questo caso parliamo di tasso di convergenza *sublineare*. In tal caso la sequenza $f_i - f_*$ converge a 0 molto lentamente: più passi facciamo meno miglioriamo in proporzione la nostra stima.

$t = 1$, $0 < r < 1$**:** In questo caso parliamo di convergenza *lineare*: il numero di cifre significative cresce in modo lineare. Questo caso può essere buono se $r << 1$, mentre se $r \approx 1$ la convergenza può risultare molto lenta.

$t = 1$, $r = 0$**:** In questo caso parliamo di convergenza *superlineare*; in particolare se vale anche che per $t = 2$ il raporto converge a un $r > 0$, si parla di convergenza quadratica; è il caso in cui il metodo converge più rapidamente.

In figura vediamo come si comportano (in scala semilogaritmica) diverse funzioni che convergono a 0 con tasso sublineare, lineare e superlineare.

Con questa definizione si può vedere che la golden ratio search e la ricerca dicotomica sono lineari con $r = \frac{\sqrt{5}-1}{2} \approx 0.618$ e $r = 0.5$ rispettivamente. Il metodo delle secanti è anch'esso lineare al caso pessimo, per esempio quando la funzione è molto inclinata (skewed) verso uno dei due estremi. Vediamo invece che in determinate ipotesi il metodo di Newton è quadratico:

Teorema 1.2.2. *Supponiamo che f sia derivabile 3 volte e che la sua derivata terza sia continua ($f \in C^3$) e che x_* sia un minimo locale con $f''(x_*) \neq 0$. Allora esiste un $\delta > 0$ tale per cui se scegliamo un qualsiasi $x_0 \in [x_* - \delta, x_* + \delta]$ il metodo di Newton converge con tasso quadratico partendo da x_0.*

Dimostrazione. Siano $\{x_i\}$ i passi generati dal metodo di Newton.
$x_{i+1} = x_i - f'(x_i)/f''(x_i)$, quindi, ricordando che $f'(x_*) = 0$,

$$x_{i+1} - x_* = x_i - x_* + \frac{f'(x_*) - f'(x_i)}{f''(x_i)} = \frac{f'(x_*) - f(x_i) - f''(x_i)(x_* - x_i)}{f''(x_i)}$$

Per la formula di Taylor (con resto di Lagrange) applicata a f', esiste un y tra x_i e x_* tale che

$$0 = f'(x_*) = f'(x_i) + f''(x_i)(x_* - x_i) + \frac{f'''(y)(x_* - x_i)}{2},$$

dunque $x_{i+1} - x_* = (f'''(y)/(2f''(x_i)))(x_i - x_*)^2$. Poiché x_* è un minimo locale, $f''(x_*) > 0$, e dunque esistono un $\delta > 0$ e due costanti k_1 per cui $k_1 \leq f''(x)$ e $|f'''(x)| \leq k_2$ (per continuità di f''') per ogni $x \in [x_* - \delta, x_* + \delta]$. Dunque

$$|x_{i+1} - x_*| \leq \frac{k_2}{2k_1}(x_i - x_*)^2.$$

Ciò prova il tasso di convergenza quadratico.

□

Capitolo 2

Ottimizzazione non vincolata

Passiamo ora al caso in cui la funzione da ottimizzare è una funzione $f : \mathbb{R}^n \to \mathbb{R}$. Chiameremo $\mathbf{x}$ in grassetto un vettore di $\mathbb{R}^n$ e $(x_1, ..., x_n)$ le sue componenti (a differenza di prima dove x_i erano le iterazioni di un metodo iterativo)

2.1 Richiami di analisi in più variabili

2.1.1 Prodotti scalari, norme, distanze

Definizione. Siano $\mathbf{x}, \mathbf{y}$ due vettori in $\mathbb{R}^n$. Il loro *prodotto scalare* (standard) è

$$\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x} \cdot \mathbf{y} = \mathbf{x}^T \mathbf{y} = \sum_{i=1}^{n} x_i y_i.$$

La *norma euclidea* di $\mathbf{x}$ è

$$||\mathbf{x}||_2 = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle} = \sqrt{\sum_{i=1}^{n} x_i^2}$$

Il concetto di norma si può generalizzare:

Definizione. Una *norma* su $\mathbb{R}^n$ è una funzione $||\cdot|| : \mathbb{R}^n \to \mathbb{R}$ con le seguenti proprietà: per ogni $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$

- $||\mathbf{x}|| \geq 0$ e $||\mathbf{x}|| = 0$ se e solo se $\mathbf{x} = \mathbf{0} = (0, ..., 0)$

- Per ogni $\alpha \in \mathbb{R}$ $||\alpha \mathbf{x}|| = |\alpha| \, ||\mathbf{x}||$
- (disuguaglianza triangolare) $||\mathbf{x} + \mathbf{y}|| \leq ||\mathbf{x}|| + ||\mathbf{y}||$

È facile vedere che la norma euclidea è in effetti una norma. Più precisamente fa parte della famiglia di norme L_p, cioè della forma

$$||\mathbf{x}||_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}}$$

Queste funzioni sono vere e proprie norme per ogni $p \geq 1$, mentre per $p < 1$ manca la proprietà di disuguaglianza triangolare. La norma euclidea corrisponde alla norma L_2. Per $p \to \infty$ definiamo la norma infinito come il limite di funzioni delle norme L_p. È facile dimostrare che

$$||\mathbf{x}||_\infty = \max_{i=1,\ldots,n} |x_i|$$

Definizione. Una *distanza* è una qualsiasi funzione $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ con le seguenti proprietà:

- $d(\mathbf{x}, \mathbf{y}) \geq 0$ e $d(\mathbf{x}, \mathbf{y}) = 0$ se e solo se $\mathbf{x} = \mathbf{y}$
- $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$ (Simmetria)
- $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z})$ (Disuguaglianza triangolare per le distanze)

Definizione. Data una qualsiasi norma $|| \cdot ||$, la *distanza indotta* da tale norma è la funzione $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ così definita:

$$d(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}||$$

Si verifica facilmente che le proprietà della norma rendono d effettivamente una distanza, cioè d soddisfa le 3 proprietà della definizione di distanza.

Definizione. Data una distanza d una palla (aperta) di centro $\mathbf{x}$ e raggio $R > 0$ è l'insieme $\mathcal{B}(\mathbf{x}, R) \subseteq \mathbb{R}^n$ definito da

$$\mathcal{B}(\mathbf{x}, R) = \{\mathbf{y} \ : \ d(\mathbf{x}, \mathbf{y}) < R\}$$

Nel seguito, quando useremo la notazione $|| \cdot ||$, intenderemo la norma euclidea, se non specificato altrimenti. Anche la distanza che useremo sarà quella euclidea,

$$d(\mathbf{x}, \mathbf{y}) = ||x - y||_2 = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}.$$

2.1.2 Gradiente, Hessiana, Jacobiana

Definizione. (Tomografia di una funzione) Data una funzione $f : \mathbb{R}^n \to \mathbb{R}$ la tomografia di f data un'origine $\mathbf{x}$ e una direzione $\mathbf{d}$ è

$$\varphi_{\mathbf{x},\mathbf{d}}(t) = f(\mathbf{x} + t\mathbf{d})$$

Come per funzioni in più variabili possiamo definire il concetto di derivabilità e differenziabilità per funzioni di più variabili:

Definizione. (Derivata direzionale)
Sia $\mathbf{d}$ un vettore in $\mathbb{R}^n$. La derivata di f in $\mathbf{x}$ lungo la direzione $\mathbf{d}$ è

$$\frac{\partial f(\mathbf{x})}{\partial \mathbf{d}} = \lim_{t\to 0} \frac{f(\mathbf{x} + t\mathbf{d}) - f(\mathbf{x})}{t} = \varphi'_{\mathbf{x},\mathbf{d}}(0)$$

se tale limite esiste ed è finito.

In particolare se $\mathbf{d} = \mathbf{e}^i$ è un vettore della base canonica di $\mathbb{R}^n$, ovvero $\mathbf{e}^i_j = 0$ per ogni $j \neq i$ e $\mathbf{e}^i_i = 1$, calcolare la derivata direzionale rispetto a $\mathbf{e}^i$, equivale a derivare f solo rispetto alla variabile x_i. Perciò denotiamo la derivata direzionale lungo $\mathbf{e}^i$ come

$$\frac{\partial f(\mathbf{x})}{\partial x_i}$$

e la chiamiamo *derivata parziale (i-esima)*

Nella pratica per calcolare la i-esima derivata parziale basta calcolare

$$f'(x_1, ..., x_{i-1}, x, x_{i+1}, ..., x_n)$$

rispetto a x, trattando le altre variabili come costanti.
Esempio. $\mathbf{x} = (x, y)$, $f(x, y) = x^2 + xy \implies \partial f/\partial x = 2x + y$, $\partial f/\partial y = x$

Definizione. Il gradiente di f è il vettore delle derivate parziali:

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f(\mathbf{x})}{\partial x_1}, ..., \frac{\partial f(\mathbf{x})}{\partial x_n} \right).$$

Definizione. Diciamo che una funzione è *differenziabile* in un punto $\mathbf{x}$ se esiste una funzione lineare ϕ tale che per ogni $\mathbf{y}$

$$f(\mathbf{y}) = f(\mathbf{x}) + \phi(\mathbf{y} - \mathbf{x}) + R(\mathbf{y}),$$

con $R(\mathbf{y})$ (il resto) che tende a 0 più velocemente di $||\mathbf{y} - \mathbf{x}||_2$ se $\mathbf{y} \to \mathbf{x}$, cioè $\frac{R(\mathbf{y})}{||\mathbf{y}-\mathbf{x}||} \xrightarrow{\mathbf{y}\to\mathbf{x}} 0$.

Se f è differenziabile, allora $\phi(\mathbf{y}) = \nabla f(\mathbf{x}) \cdot \mathbf{y}$ Perciò se f è differenziabile in $\mathbf{x}$ allora esistono tutte le derivate parziali $\frac{\partial f(\mathbf{x})}{\partial x_i}$, e $f(\mathbf{y}) + \nabla f(\mathbf{x}) \cdot \mathbf{y}$ è un modello al primo ordine di f.

Proposizione 2.1.1. *Se f è differenziabile in $\mathbf{x}$, per ogni direzione $\mathbf{d}$ si ha che la derivata direzionale lungo $\mathbf{d}$ è data dal prodotto scalare tra il gradiente di f in $\mathbf{x}$ e $\mathbf{d}$:*

$$\frac{\partial f(\mathbf{x})}{\partial \mathbf{d}} = \nabla f(\mathbf{x}) \cdot \mathbf{d}$$

Dimostrazione. (non necessaria) Si usa la chain rule:

$$\begin{aligned}\frac{\partial f(\mathbf{x})}{\partial \mathbf{d}} &= \lim_{t \to 0} \frac{f(\mathbf{x} + t\mathbf{d}) - f(\mathbf{x})}{t} = \frac{df(\mathbf{x} + t\mathbf{d})}{dt}(t = 0) \\ &= \nabla f(\mathbf{x}) \frac{d(\mathbf{x} + t\mathbf{d})}{dt} = \nabla f(\mathbf{x}) \cdot \mathbf{d}\end{aligned}$$

□

L'esistenza di tutte le derivate parziali non indica in generale che la funzione sia differenziabile: in $\mathbb{R}^n$, con $n > 2$, "derivabilità $\neq$. differenziabilità". Tuttavia vale il teorema seguente:

Teorema 2.1.1. (differenziale totale) *Se esistono le derivate parziali in un punto $\mathbf{x}$ e esiste un $\delta > 0$ tale che le derivate parziali sono tutte continue nella palla $\mathcal{B}(\mathbf{x}, \delta)$, allora f è differenziabile in $\mathbf{x}$*

Cioè se le derivate parziali sono continue, allora f è differenziabile.

Definizione. Se F è una funzione a valori vettoriali, cioè
$F = (F_1, ..., F_m) : \mathbb{R}^n \to \mathbb{R}^m$, possiamo definire la matrice Jacobiana come la matrice $n \times m$ delle derivate parziali di ciascuna f_i :

$$J_F(\mathbf{x}) = \begin{pmatrix} \frac{\partial F_1(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial F_1(\mathbf{x})}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial F_m(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial F_m(\mathbf{x})}{\partial x_n} \end{pmatrix}$$

Un particolare tipo di matrice jacobiana è la *matrice hessiana* di una funzione $f : \mathbb{R}^n \to \mathbb{R}$, ovvero la matrice delle derivate seconde:

$$\nabla^2(\mathbf{x}) = J_{\nabla f}(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_1} \\ \vdots & & \vdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_n} & \cdots & \frac{\partial^2 f_(\mathbf{x})}{\partial x_n \partial x_n} \end{pmatrix}$$

In generale le derivate parziali non commutano: $\frac{\partial^2 f}{\partial x_i \partial x_j} \neq \frac{\partial^2 f}{\partial x_j \partial x_i}$. Tuttavia l'uguaglianza vale se $f \in C^2$. Perciò se $f \in C^2$, la matrice hessiana è *simmetrica*.

2.1.3 Condizioni di ottimalità al primo e secondo ordine

Analogamente al caso unidimensionale, se $\mathbf{x}$ è un minimo locale, allora il gradiente in $\mathbf{x}$ si annulla. Più precisamente:

Teorema 2.1.2. *Se f è differenziabile in $\mathbf{x}$ e $\mathbf{x}$ è un minimo locale, allora $\nabla f(\mathbf{x}) = 0$.*

Dimostrazione. Supponiamo per assurdo che $\mathbf{x}$ sia minimo locale e $\nabla f(\mathbf{x}) \neq 0$. Allora la derivata direzionale lungo la direzione $\mathbf{d} = \frac{-\nabla f(\mathbf{x})}{||\nabla f(\mathbf{x})||}$ è negativa: infatti per la Proposizione 2.1.1,

$$\frac{\partial f(\mathbf{x})}{\partial \mathbf{d}} = \nabla f(\mathbf{x}) \cdot \mathbf{d} = \nabla f(\mathbf{x}) \cdot \frac{-\nabla f(\mathbf{x})}{||\nabla f(\mathbf{x})||} = \frac{-\|\nabla f(\mathbf{x})||^2}{||\nabla f(\mathbf{x})||} = -||\nabla f(\mathbf{x})||.$$

Dunque per la differenziabilità di f si ha che per qualsiasi ϵ abbastanza piccolo, se $\mathbf{y} = \mathbf{x} + \epsilon\mathbf{d}$

$$f(\mathbf{y}) = f(\mathbf{x}) + \nabla f(\mathbf{x})\epsilon\mathbf{d} + R(\mathbf{y}) = f(\mathbf{x}) - \epsilon||\nabla f(\mathbf{x})|| + R(\mathbf{y})$$

Ma abbiamo detto che

$$\frac{R(\mathbf{y})}{||\mathbf{y} - \mathbf{x}||} = \frac{R(\mathbf{y})}{\epsilon} \xrightarrow{\epsilon \to 0} 0$$

perciò esiste ϵ' tale che, per ogni $\epsilon < \epsilon'$, $R(\mathbf{y}) \leq \frac{\nabla f(\mathbf{x})\epsilon}{2}$ e dunque

$$f(\mathbf{y}) = f(\mathbf{x}) - \nabla f(\mathbf{x})\epsilon\mathbf{d} + R(\mathbf{y}) \leq f(\mathbf{x}) - \nabla f(\mathbf{x})\frac{\epsilon}{2} < f(\mathbf{x}).$$

Ciò contraddice però che $\mathbf{x}$ sia un minimo locale.

□

Osservazione 3. $\mathbf{d}$ definita prima è la direzione di massima decrescita della funzione f, e $\nabla f(\mathbf{x})/\|\nabla f(\mathbf{x})\|$ è la direzione di massima crescita: infatti per ogni direzione $\mathbf{d}'$ con $||\mathbf{d}'|| = 1$ si ha che

$$\frac{\partial f(\mathbf{x})}{\partial \mathbf{d}'} = \mathbf{d}' \cdot \nabla f(\mathbf{x}) = \|\mathbf{d}'\|\|\nabla f(\mathbf{x})\| \cos\theta = \|\nabla f(\mathbf{x})\| \cos\theta.$$

Il massimo di questa funzione è assunto quando $\cos\theta = 1$, cioè quando $\mathbf{d}'$ è parallelo a $\nabla f(\mathbf{x})$ e con lo stesso verso, cioè quando $\mathbf{d}' = \nabla f(\mathbf{x})/\|\nabla f(\mathbf{x})\|$. Viceversa il minimo è assunto quando $\mathbf{d}' = -\nabla f(\mathbf{x})/\|\nabla f(\mathbf{x})\| = \mathbf{d}$.

Il Teorema 2.1.2 ci dice solo che se $\mathbf{x}_*$ è un punto di minimo locale allora $\nabla f(\mathbf{x}_*) = 0$. Esistono però altri punti stazionari, cioè tali che $\nabla f(\mathbf{x}) = 0$ che non sono minimi locali: in particolare abbiamo massimi locali e punti di sella. Andando a studiare l'hessiana della funzione si possono tuttavia trovare delle condizioni più precise per assicurarci che il punto stazionario sia effettivamente un minimo:

Teorema 2.1.3. *Se $f \in C^2$*

1. *Se x_* è un minimo locale di f, allora $\nabla^2 f(\mathbf{x}_*)$ è semidefinita positiva.*
2. *Viceversa vale che se $\mathbf{x}_*$ è un punto stazionario e $\nabla^2 f(\mathbf{x}_*)$ è definita positiva, allora x_* è un minimo locale*

.

Dimostrazione. Dimostriamo la prima delle due tesi. Abbiamo che vale la formula di Taylor del secondo ordine

$$f(\mathbf{y}) = f(\mathbf{x}) + \nabla f(\mathbf{x})(\mathbf{y}-\mathbf{x}) + (\mathbf{y}-\mathbf{x})^T \nabla^2 f(\mathbf{x})\,(\mathbf{y}-\mathbf{x}) + o(||\mathbf{y}-\mathbf{x}||^2)$$

Se $\mathbf{x} = \mathbf{x}_*$ è minimo, in particolare è stazionario e vale quindi che

$$f(\mathbf{y}) = f(\mathbf{x}) + \frac{1}{2}(\mathbf{y}-\mathbf{x})^T \nabla^2 f(\mathbf{x})(\mathbf{y}-\mathbf{x}) + o(||\mathbf{y}-\mathbf{x}||^2) \tag{2.1}$$

Se $\nabla^2 f(\mathbf{x})$ non fosse semidefinita positiva, esisterebbero uno scalre $\lambda > 0$ e una direzione $\mathbf{d}$ tale per cui

$$\mathbf{d}^T \nabla^2 f(\mathbf{x})\,\mathbf{d} = -\lambda\,||\mathbf{d}||^2 < 0$$

Prendendo allora $\mathbf{y} = \mathbf{x} + \epsilon\mathbf{d}$ per ogni $\epsilon > 0$ abbastanza piccolo, si ha che

$$\begin{aligned} f(\mathbf{y}) &= f(\mathbf{x}) - \frac{1}{2}\,\epsilon\,\lambda\,||d^2|| + o(\epsilon^2) \\ &\leq f(\mathbf{x}) - \frac{1}{4}\,\epsilon\,\lambda\,||\mathbf{d}||^2 < f(\mathbf{x}) \end{aligned}$$

Ma ciò è assurdo perchè $\mathbf{x}$ è un minimo.

Proviamo ora la seconda delle due tesi. Se $\mathbf{x}$ è un punto stazionario vale ancora l'equazione (2.1). Poiché $\nabla^2 f(\mathbf{x})$ è definita positiva, esiste una costante $c \geq 0$ tale per cui per ogni direzione $\mathbf{d}$

$$\mathbf{d}^T \nabla^2 f(\mathbf{x})\, \mathbf{d} \geq c > 0.$$

Prendendo $\mathbf{y} = \mathbf{x} + \epsilon \mathbf{d}$ per ogni $\mathbf{d}$ e per ogni $\epsilon > 0$ abbastanza piccolo, si ha che

$$f(\mathbf{y}) = f(\mathbf{x}) + \frac{1}{2}\,\epsilon\, c + o(\epsilon^2) \geq f(\mathbf{x}) + \frac{1}{4}\,\epsilon\, c > f(\mathbf{x})$$

Abbiamo perciò dimostrato che esiste un ϵ tale per cui $f(\mathbf{x}) < f(\mathbf{y})$ per ogni $\mathbf{y} \in \mathcal{B}(\mathbf{x}, \epsilon)$. □

2.1.4 Funzioni convesse

Se vogliamo garantire che quando troviamo un punto stazionario, questo sia un punto di minimo locale, dobbiamo verificare che l'hessiana in tale punto sia semidefinita positiva. Nel caso delle funzioni convesse questa convinzione è sempre verificata.

Definizione. Una funzione $f : \mathbb{R}^n \to \mathbb{R}$ [1] è *convessa* se per ogni $\mathbf{x}, \mathbf{y} \in X$ e per ogni $t \in [0, 1]$

$$f\left(t\mathbf{x} + (1-t)\mathbf{y}\right) \leq t\, f(\mathbf{x}) + (1-t) f(\mathbf{y})$$

Se vale la disuguaglianza opposta allora la funzione si dice *concava*. Enunciamo ora alcune proprietà importanti delle funzioni convesse:

Proposizione 2.1.2. *f convessa* $\iff$ $(\nabla f(\mathbf{x}) - \nabla f(\mathbf{z})) \cdot (\mathbf{x} - \mathbf{z}) \geq 0$

Proposizione 2.1.3. *Se $f \in C^2$, allora f è convessa se e solo se la sua matrice hessiana è semidefinita positiva in ogni punto.*

In generale non vale che l'hessiana di f sia definita positiva in ogni punto, ma si può comunque dimostrare che ogni punto stazionario è un minimo *globale*:

[1] In teoria potremmo definire f anche solo su un insieme $X \subseteq \mathbb{R}^n$ convesso, ovvero tale per cui, dati due suoi punti qualsiasi, l'intero segmento tra essi è contenuto in X. La definizione formale di insieme convesso e le relative proprietà sono date nella sezione sull'ottimizzazione vincolata.

Proposizione 2.1.4. $\mathbf{x}$ *è un punto stazionario di* f *se e solo se per ogni* $\mathbf{z}$ $f(\mathbf{z}) \geq f(\mathbf{x})$

Possiamo anche rinforzare la definizione di convessità richiedendo qualcosa di più

Definizione. Una funzione f è *strettamente convessa* se è convessa e vale per ogni $t \notin \{0, 1\}$

$$f\left(t\mathbf{x} + (1-t)\mathbf{y}\right) < t\, f(\mathbf{x}) + (1-t) f(\mathbf{y})$$

Esempio. Ogni funzione f lineare è convessa ma non strettamente convessa: vale infatti l'= nella definizione di convessità per ogni t.

Esempio. Una funzione quadratica $f(x) = \mathbf{x}^T Q\mathbf{x}$ è strettamente convessa se e solo se Q è definita positiva.

Ovviamente se f è strettamente convessa allora è anche convessa, ma abbiamo visto con l'esempio delle funzioni lineari che non vale il viceversa. Inoltre se f è strettamente convessa, allora ha un unico punto stazionario, e tale punto è dunque *l'unico minimo globale*. Una definizione ancora più forte è la seguente.

Definizione. Una funzione f convessa si dice *fortemente convessa* o τ-*convessa* con modulo τ se $f(\mathbf{x}) - \frac{\tau}{2}||\mathbf{x}||^2$ è convessa.

Esempio. Se Q è definita positiva, $f(\mathbf{x}) = \mathbf{x}^T Q\mathbf{x}$ è τ-convessa con

$$\tau = \min_{i=1,\dots,n} \lambda_i,$$

dove i λ_i sono gli autovalori di Q.

Esempio. $-\log(x)$ è una funzione strettamente convessa, ma non τ-convessa.

Proposizione 2.1.5. $f \in C^2$ *è* τ*-convessa se e solo se gli autovalori di* $\nabla^2(f\mathbf{x})$ *sono sempre* $\geq \tau$ *al variare di* $\mathbf{x}$

Dimostrazione. $g(\mathbf{x}) = f(\mathbf{x}) - \frac{\tau}{2}\|\mathbf{x}\|^2$ è convessa e quindi $\nabla^2 g(\mathbf{x}) = \nabla^2 f(\mathbf{x}) - \tau I$ è semi-definita positiva, perciò detto λ l'autovalore minimo di $\nabla^2 f(\mathbf{x})$, $\lambda - \tau \geq 0$, cioè $\lambda \geq \tau$. □

2.2 Algoritmo della discesa del gradiente

Abbiamo visto che dato un punto $\mathbf{x}$ in cui f è differenziabile, abbiamo un modello del primo ordine di f dato da

$$f(\mathbf{x}) + \nabla f(\mathbf{x})(\mathbf{y} - \mathbf{x})$$

Sottolineiamo che questo modello per la maggior parte delle volte è buono solo se siamo molto vicini a $\mathbf{x}$ (in una palla $\mathcal{B}(\mathbf{x}, \epsilon)$ con ϵ molto piccolo).

L'algoritmo della discesa del gradiente si basa su questo modello, ed è il seguente:

- Scegliamo un vettore iniziale $\mathbf{x}^0 \in \mathbb{R}^n$., poi per ogni $i = 0, 1, 2, ...,$
- se $||\nabla f(\mathbf{x}^i)|| < \epsilon$ ci fermiamo,
- altrimenti ad ogni iterazione scegliamo un' ampiezza del passo α_i e aggiorniamo il vettore in questo modo:

  $$\mathbf{x}^{i+1} = \mathbf{x}^i - \alpha_i \nabla f(\mathbf{x}^i) = \mathbf{x}^i + \alpha_i \mathbf{d}^i.$$

  Come abbiamo già osservato prima questo metodo ha senso perché $\mathbf{d} = -\nabla f(\mathbf{x})$ è la direzione in cui la funzione decresce di più in $\mathbf{x}$.

Algoritmo 1 Algoritmo della Discesa del gradiente

```
procedure x = SDQ ( f , x , ε )
  while( ‖ ∇f( x ) ‖ > ε ) do
    d ← −∇f( x ); α ← stepsize( f , x , d ); x ← x + αd;
```

La domanda che ci poniamo subito è come scegliere l'ampiezza del passo α_i, ovvero come implementare `stepsize`$(f, \mathbf{x}, \mathbf{d})$. Questo problema si ritrova anche nel machine learning, dove la scelta del learning rate è molto importante. Durante l'esecuzione di questo algoritmo infatti, se scegliamo male α possiamo imbatterci in due problemi opposti [2] :

Scilla: Se α_i è troppo grande, rischiamo che $f(\mathbf{x}^{i+1}) >> f(\mathbf{x}^i)$ e l'algoritmo può anche divergere senza trovare il minimo;

[2] `https://www.elicriso.it/it/mitologia_ambiente/scilla_cariddi/`

Cariddi: Viceversa se α^i è troppo piccolo, i passi fatti dall'algoritmo saranno molto piccoli e rischiamo di non arrivare al minimo locale a cui stiamo puntando.

In linea di massima α_i dovrebbero decrescere ma non troppo velocemente:

Esempio. Se $f(\mathbf{x}) = -\mathbf{x}$, $\mathbf{x}^0 = 0$, $\alpha_i = \frac{1}{i^2}$, per ogni passo si ha $\mathbf{d}^i = -f'(\mathbf{x}) = 1$ e quindi [3]

$$\mathbf{x}^i = \sum_{j=1}^{i} \frac{1}{i^2} \xrightarrow{i\to\infty} \frac{\pi^2}{6} \approx 1.645,$$

mentre evidentemente $f_* = -\infty$. Se invece fosse $\alpha_i = \frac{1}{i}$, allora

$$\mathbf{x}^i = \sum_{j=1}^{i} \frac{1}{i} \xrightarrow{i\to\infty} +\infty$$

e perciò effettivamente si ha che $f(\mathbf{x}^i) \to -\infty$

2.2.1 Metodo del gradiente con passo costante

Spesso nel Machine Learning, quando si implementa la discesa del gradiente, si fissa un *learning rate* $\alpha_i = \alpha$, come iperparametro .

Iniziamo con un' osservazione: se vogliamo che la successione $\mathbf{x}^i$ tenda a un punto $\mathbf{x}_*$, dev'essere necessariamente $\|x^{i+1} - \mathbf{x}^i\| \to 0$ (ovvero la successione $\{\|\mathbf{x}^i\|\}$ è *di Cauchy*). Se prendessimo come direzione $\mathbf{d}^i = -\frac{\nabla f(\mathbf{x}^i)}{\|\nabla f(\mathbf{x}^i)\|}$, ovviamente $\|\mathbf{d}^i\| = 1$ e allora

$$\|\mathbf{x}^{i+1} - \mathbf{x}^i\| = \|\alpha \mathbf{d}^i\| = |\alpha| \|\mathbf{d}^i\| = |\alpha|, \tag{2.2}$$

cioè la differenza in norma tra i vettori dell'iterazione è costante.

Prendiamo perciò $\mathbf{d}^i = -\nabla f(\mathbf{x}^i)$ senza normalizzare il vettore. In questo modo cambiando l'ultimo passaggio di (2.2) si ha che $\|x^{i+1} - \mathbf{x}^i\| = |\alpha| \, \|\mathbf{d}^i\|$, che tende a 0 se $\|\mathbf{d}^i\| = \|\nabla f(\mathbf{x}^i)\| \to 0$, cioè se ci siamo avvicinando ad un punto stazionario.

Ci chiediamo ora quanto piccolo debba essere α per garantire che il metodo converga a un punto stazionario. Intuitivamente, se f varia molto rapidamente, α dovrà essere molto piccola. Assumiamo perciò come prima cosa che f sia L-lipchitz.

[3] `https://it.wikipedia.org/wiki/Problema_di_Basilea`

Osservazione 4. Se f è differenziabile, allora è L-lipschitz se e solo se $||\nabla f|| \leq L$.

In realtà per controllare che f non vari troppo rapidamente ci serve controllare che ∇f sia anch'esso lipschitz:

Definizione. f è L-smooth se ∇f è L-lipschitz.

In particolare se $f \in C^2$, similmente all'osservazione precedente f è L-smooth se gli autovalori di $\nabla^2 f(\mathbf{x})$ al variare di $\mathbf{x}$ sono compresi in $[-L, L]$ (ciò corrisponde al fatto che la norma di $\nabla^2 f(\mathbf{x})$ sia limitata) .

Ricordiamo che dati due vettori $\mathbf{x}$ e $\mathbf{d}$ la tomografia di f è la funzione in una variabile $\varphi_{\mathbf{x},\mathbf{d}}(t) = f(\mathbf{x} + t\mathbf{d})$. Ci interessa in particolare la tomografia $\varphi_{\mathbf{x}^i,\mathbf{d}^i}$: nell'algoritmo del gradiente stiamo infatti variando f, a partire da $\mathbf{x}^i$, lungo la direzione $\mathbf{d}^i = -\nabla f(\mathbf{x}^i)$, di un passo α:

$$f(\mathbf{x}^i) = \varphi_{\mathbf{x}^i,\mathbf{d}^i}(0),\ f(\mathbf{x}^{i+1}) = \varphi_{\mathbf{x}^i,\mathbf{d}^i}(\alpha).$$

Se f non varia molto velocemente, ci aspettiamo che anche φ non vari molto velocemente. Vale infatti la seguente proposizione:

Proposizione 2.2.1. *Se f è L-smooth, $\varphi_{\mathbf{x},\mathbf{d}}(t)$ è $L\|\mathbf{d}\|^2$- smooth.*
In particolare quindi $\varphi_{\mathbf{x}^i,\mathbf{d}^i}(t)$ è $L\|\nabla f(\mathbf{x}^i)\|^2$- smooth.

Per semplificare la notazione indichiamo con $\varphi = \varphi_{\mathbf{x}^i,\mathbf{d}^i}$. Dalla Proposizione 2.1.1 si ha che

$$\varphi'(0) = \frac{\partial f}{\partial \mathbf{d}^i}(\mathbf{x}^i) = \mathbf{d}^i \cdot \nabla f(\mathbf{x}^i) = -\|\nabla f(\mathbf{x}^i)\|^2.$$

Perciò, per ogni $\alpha > 0$, dal fatto che φ è $L\|\nabla f(\mathbf{x}^i)\|^2$- smooth, si ottiene che

$$\varphi'(\alpha) \leq \varphi'(0) + L||\nabla f(\mathbf{x}^i)||^2\alpha = ||\nabla f(\mathbf{x}^i)||^2(L\alpha - 1)$$

Dunque, sicuramente $\varphi'(\alpha) < 0$ per ogni $\alpha \in [0, 1/L)$, cioè φ è decrescente nell'intervallo $[0, 1/L]$ φ. Prendiamo dunque α il più grande possibile, ovvero $\alpha = \frac{1}{L}$.

Abbiamo dunque che $\varphi(\alpha) = f(\mathbf{x}^{i+1}) < f(\mathbf{x}^i) = \varphi(0)$. Ci chiediamo ora se possiamo in qualche modo stimare $f(\mathbf{x}^i) - f(\mathbf{x}^{i+1})$.

Teorema 2.2.1. *Per ogni $\alpha \in [0, \frac{1}{L}]$,*

$$\varphi(\alpha) \leq \varphi(0) + ||\nabla f(\mathbf{x})||^2 \left(\frac{L\alpha^2}{2} - \alpha\right)$$

Dimostrazione. (non necessaria) Sia $g : [0, \frac{1}{L}] \to \mathbb{R}$ la funzione così definita:

$$g(t) = \varphi'(0) + L||\nabla f(\mathbf{x}^i)||^2 t$$

Perciò per ogni t, $\varphi'(t) \leq g(t)$. Per la monotonia dell'integrale (e per il Teorema Fondamentale del Calcolo Integrale) si ha quindi che

$$\varphi(\alpha) - \varphi(0) = \int_0^\alpha \varphi'(t)\, dt \leq \int_0^\alpha g(t)\, dt. \tag{2.3}$$

Calcoliamo allora il secondo integrale, che è facile perché g è lineare in t:

$$\begin{aligned}
\int_0^\alpha g(t)\, dt &= \left[\varphi'(0)t + \frac{L||\nabla f(\mathbf{x}^i)||^2 t^2}{2}\right]_0^\alpha \\
&= \varphi'(0)\alpha + \frac{L||\nabla f(\mathbf{x}^i)||^2 \alpha^2}{2} \\
&= -||\nabla f(\mathbf{x}^i)||^2 \alpha + ||\nabla f(\mathbf{x}^i)||^2 \frac{L\alpha^2}{2} \\
&= ||\nabla f(\mathbf{x}^i)||^2 \left(\frac{L\alpha^2}{2} - \alpha\right).
\end{aligned}$$

Mettendo l'ultima equazione e la disequazione (2.3) assieme arriviamo alla tesi. □

In particolare se $\alpha = \frac{1}{L}$ si ha che

$$f(\mathbf{x}^{i+1}) = \varphi\left(\frac{1}{L}\right) \leq \varphi(0) - ||\nabla f(\mathbf{x}^i)||^2 \frac{1}{2L} = f(\mathbf{x}^i) - ||\nabla f(\mathbf{x}^i)||^2 \frac{1}{2L}$$

In conclusione, il metodo del gradiente con passo costante è sub-lineare su funzioni L-smooth, se "crediamo" che per ogni i, $f(\mathbf{x}^{i+1})$ sia abbastanza vicino alla stima dall'alto appena dimostrata. In particolare si può dimostrare che per ottenere $f(\mathbf{x}^i) \leq f(\mathbf{x}_*) + \varepsilon$ servono $O(1/\varepsilon)$ passi. La buona notizia è che la stima non dipende dalla dimensione del problema, ossia dal numero di variabili di f.

In alcuni casi particolari si può tuttavia dimostrare una maggior velocità dell'algoritmo: nel caso in cui la funzione sia L-smooth, e anche *fortemente convessa* con modulo τ si dimostra che l'algoritmo converge linearmente, con tasso di convergenza

$$r \leq 1 - \frac{\tau}{L}. \tag{2.4}$$

Dimostrazione. Iniziamo con una importante proprietà delle funzioni τ-convesse, che è una generalizzazione della Proposizione 2.1.2: detto $\mathbf{x}_i$ il punto trovato al passo i-esimo del metodo del gradiente, per ogni altro $\mathbf{x}$ vale

$$f(\mathbf{x}) \geq f(\mathbf{x}^i) + \nabla f(\mathbf{x}^i)(\mathbf{x} - \mathbf{x}^i) + \tau\|\mathbf{x} - \mathbf{x}^i\|^2 \tag{2.5}$$

La disuguaglianza continua a valere prendendo il minimo rispetto alla variabile $\mathbf{x}$ delle due espressioni:

$$\min_{\mathbf{x}} f(\mathbf{x}) \geq \min_{\mathbf{x}} f(\mathbf{x}^i) + \nabla f(\mathbf{x}^i)(\mathbf{x} - \mathbf{x}^i) + \tau\|\mathbf{x} - \mathbf{x}^i\|^2$$

A sinistra il minimo viene assunto quando $\mathbf{x} = \mathbf{x}_*$, a destra dobbiamo derivare l'espressione in $\mathbf{x}$, e porre la derivata uguale a 0. Otteniamo

$$\arg\min_{\mathbf{x}} f(\mathbf{x}^i) + \nabla f(\mathbf{x}^i)(\mathbf{x} - \mathbf{x}^i) + \tau\|\mathbf{x} - \mathbf{x}^i\|^2 = \mathbf{x}^i - \nabla f(\mathbf{x}^i)/(2\tau)$$

Sostituendo il valore trovato per $\mathbf{x}$ in (2.5) otteniamo che

$$f(\mathbf{x}_*) \geq f(\mathbf{x}^i) - \frac{\|\nabla f(\mathbf{x}^i)\|}{2\tau},$$

cioè

$$\|\nabla f(\mathbf{x}_*)\|^2 \geq 2\tau\left(f(\mathbf{x}^i) - f(\mathbf{x}_*)\right) = 2\tau\mathbf{a}^i \tag{2.6}$$

con $\mathbf{a}^i = f(\mathbf{x}^i) - f(\mathbf{x}_*)$ (il gap al passo i). Per la L-smoothness di f vale inoltre che $\mathbf{a}^{i+1} - \mathbf{a}^i \leq \|\nabla f(\mathbf{x}_*)\|^2/(2L)$ (è il Teorema 2.2.1 con $\alpha = 1/L$). Mettendo insieme questa disuguaglianza con la (2.6), otteniamo che

$$\mathbf{a}^{i+1} \leq \mathbf{a}^i\left(1 - \frac{\tau}{L}\right).$$

Abbiamo dimostrato la convergenza lineare con tasso $r = 1 - \tau/L$. □

2.2.2 Metodo del gradiente con line-search "esatta"

Vogliamo ora vedere le proprietà del gradiente se scegliamo α_i come il passo che minimizza φ. In altre parole in questo caso per ogni i vogliamo trovare

$$\alpha^{i+1} = \arg\min_{\alpha\in\mathbb{R}^+} \varphi(\alpha) = \arg\min_{\alpha\in\mathbb{R}^+} f(\mathbf{x}^i + \alpha\mathbf{d}^i)$$

Ad ogni passo dobbiamo perciò risolvere un problema di line-search (LS) cioè di ottimizzazione in una variabile. Poiché abbiamo visto che la line-search può essere molto difficile, analizziamo il caso particolari di funzioni quadratiche.

Prendiamo dunque $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q\mathbf{x} + \mathbf{q}^T\mathbf{x}$, e chiamiamo $\mathbf{g} = \nabla f(\mathbf{x}) = Q\mathbf{x}+\mathbf{q}$. Per minimizzare $\varphi(\alpha) = f(\mathbf{x}-\alpha\mathbf{g})$ dobbiamo trovare dove si annulla φ':

$$\varphi'(\alpha) = \mathbf{g}\cdot\nabla f(\mathbf{x}-\alpha\mathbf{g}) = \mathbf{g}\cdot(Q\mathbf{x}-\alpha Q\mathbf{g}+\mathbf{q}) = \mathbf{g}\cdot(-\alpha Qg+\mathbf{g})$$

che si annulla quindi se

$$\alpha = \frac{||\mathbf{g}||^2}{\mathbf{g}^T Q\mathbf{g}} \tag{2.7}$$

Osservazione 5. Il calcolo di α richiede una moltiplicazione matrice per vettore, perciò costa $O(n^2)$ per ogni iterazione. Notiamo che anche il calcolo di $\nabla f(\mathbf{x})$ costa $O(n^2)$ operazioni.

Velocità di convergenza del metodo

Vogliamo analizzare il tasso di convergenza del metodo per funzioni quadratiche: siano $\lambda_1 \geq \cdots \geq \lambda_n$ gli autovalori di Q. Notiamo che possiamo assumere che Q sia semi-definita positiva, poiché se esiste un autovalore negativo, il problema è illimitato inferiormente. Per dimostrare il risultato di convergenza ci serve in realtà che Q sia strettamente definita positiva, cioè $\lambda_n > 0$. Ciò equivale a dire che Q è L-smooth e τ-convessa, con $L = 2\lambda_1$ e $\tau = 2\lambda_n$. In questo caso Q è anche invertibile, e gli autovalori di Q^{-1} sono $1/\lambda_n \geq \cdots \geq 1/\lambda_1$.

Lemma 2.2.1. *Chiamiamo $E(\mathbf{x}^i) = f(\mathbf{x}^i) - f(\mathbf{x}_*)$. Allora la sequenza $\{\mathbf{x}^i\}_i$ generata dal metodo del gradiente con α_i scelto come in (2.7) soddisfa*

$$E(\mathbf{x}^{i+1}) = \left(1 - \frac{\|\mathbf{d}^i\|^4}{((\mathbf{d}^i)^T Q\mathbf{d}^i)\,((\mathbf{d}^i)^T Q^{-1}\mathbf{d}^i)}\right) E(\mathbf{x}^i) \tag{2.8}$$

Perciò, visto che per ogni $\mathbf{x}$ vale $\mathbf{x}^T Q\mathbf{x} \leq \lambda_1\|\mathbf{x}\|^2$ e $\mathbf{x}^T Q^{-1}\mathbf{x} \leq \|\mathbf{x}\|^2/\lambda_n$ vale

$$\frac{\|\mathbf{x}\|^2}{\mathbf{x}^T Q\mathbf{x}} \geq \frac{1}{\lambda_1}, \quad \frac{\|\mathbf{x}\|^2}{\mathbf{x}^T Q^{-1}\mathbf{x}} \geq \lambda_n,$$

da cui prendendo $\mathbf{x} = \mathbf{d}^i$ e sostituendo in (2.8), si ottiene

$$E(\mathbf{x}^{i+1}) \leq \left(1 - \frac{\lambda_n}{\lambda_1}\right) E(\mathbf{x}^i).$$

Il tasso r soddisfa perciò lo stesso bound che si trova nel caso di passo costante, ma in realtà usando la disuguaglianza di Kantorovich si può dimostrare una limitazione più forte:

$$r \leq \left(\frac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n}\right)^2.$$

Il metodo dunque converge con velocità lineare, ma tasso migliore di quello che abbiamo visto scegliendo un passo costante.

Metodo nel caso di funzioni generali

Nel caso generale, abbiamo già visto nel capitolo 1 che una line-search esatta è spesso troppo costosa o addirittura impossibile, perciò in realtà possiamo usare solo una ricerca approssimativamente esatta.

2.2.3 Condizioni di Armijo e Wolfe

Il metodo del gradiente funziona persino con un passo costante, perciò a maggior ragione dovrebbe funzionare con una line search, anche se non esatta. Per farlo diamo delle condizioni su questa ricerca: vogliamo innanzitutto che $f^i = f(\mathbf{x}^i)$ decresca abbastanza:

Definizione. (Condizione di Armijo) Dato un valore $0 < m_1 << 1$, vogliamo che

$$\varphi(\alpha) \leq \varphi(0) + \alpha m_1 \varphi'(0) \tag{2.9}$$

Il problema è che se α è molto vicino a 0, la condizione è soddisfatta (vedasi la Figura (2.1)), infatti sviluppando φ con Taylor vicino a 0, se α è piccolo otteniamo $\varphi(\alpha) \approx \varphi(0) + \alpha\varphi'(0) > \varphi(0) + \alpha m_1 \varphi'(0)$. Tuttavia se vale la condizione di Armijo e evitiamo che i passi siano troppo piccoli, allora abbiamo convergenza:

Proposizione 2.2.2. *Se esiste un $\bar{\alpha} > 0$ tale che per ogni i $\alpha_i \geq \bar{\alpha}$, e vale la condizione di Armijo (2.9) per ogni i, allora*

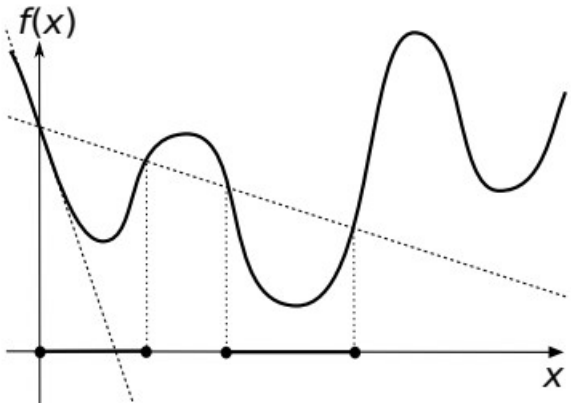

Figura 2.1: I segmenti più spessi sono le aree in cui vale la condizione di Armijo

- *O f è inferiormente illimitata ($f_* = -\infty$)*
- *oppure $||\nabla f(\mathbf{x}^i)|| \to 0$, ovvero $\{\mathbf{x}^i\}$ converge ad un punto stazionario.*

Dimostrazione. Se vale la condizione (2.9) per ogni i e $||\nabla f(\mathbf{x}^i)||^2 > \epsilon > 0$ per ogni i [4], vale allora

$$f^{i+1} \leq f^i + m_1 \alpha_i \varphi_i'(0) \leq f^i - m_1 \bar{\alpha} \epsilon$$

e per induzione si ha che

$$f^i \leq f^0 - m_i \bar{\alpha} \varepsilon i \to -\infty$$

□

Un modo per evitare di scegliere α troppo piccolo è imporre anche la cosiddetta condzione di Goldstein: dato $m_2 > m_1$, imponiamo

$$\varphi(\alpha) \geq \varphi(0) + \alpha m_2 \varphi'(0)$$

Il problema di questa condizione è che, insieme alla condizione di Armijo può escludere molti/ tutti i minimi locali (Figura 2.2)

Introduciamo perciò una terza condizione che non esclude di per sé nessun punto stazionario:

[4] In realtà sarebbe più corretto dire per infiniti i, visto che stiamo negando che $\nabla f(\mathbf{x}^i) \to 0$, ma la dimostrazione funziona ugualmente a meno di ridursi ad una sottosuccessione $\{i_j\}_{j=0}^{\infty}$

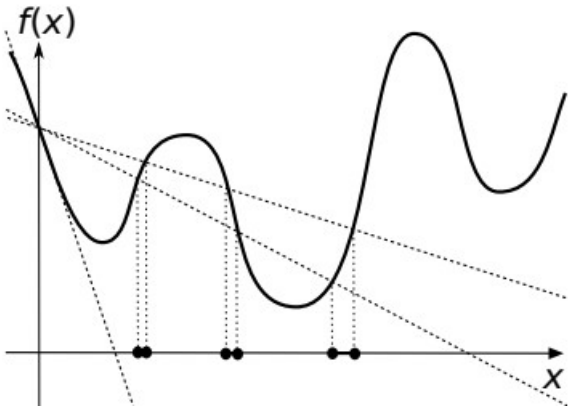

Figura 2.2: I brevissimi segmenti tra le coppie di punti sono le aree in cui valgono le condizioni di Armijo e di Goldstein. Si può notare come in questo caso tutti i minimi siano esclusi.

Definizione. (Condizione di Wolfe) Dato $m_1 < m_3 < 1$, vogliamo che sia soddisfatta la condizione

$$\varphi'(\alpha) \geq m_3\varphi'(0) \tag{2.10}$$

Vogliamo cioè che la derivata lungo la direzione di ricerca di f sia più vicina allo 0 dopo ogni passo, oppure sia positiva (anche molto più grande di 0) Per evitare che $\varphi'(\alpha) >> 0$ possiamo rafforzare la condizione in

$$|\varphi'(\alpha)| \leq m_3|\varphi'(0)|. \tag{2.11}$$

Quest'ultima condizione è detta di Wolfe forte.

Line Search di Armijo e Wolfe

Le condizioni di Armijo e Wolfe combinate forniscono un modo ragionevole di scegliere il passo per ogni iterazione, senza rischiare di tagliare troppi minimi locali: i minimi locali potrebbero essere tagliati fuori solo dalla condizione di Armijo, ma se m_1 non è troppo vicino a 1 ciò non succede. Nella pratica si sceglie infatti m_1 pari a circa 10^{-4}

Proposizione 2.2.3. *Se φ è derivabile con continuità e $\varphi(\alpha)$ è inferiormente limitato (non va a $-\infty$ se $\alpha \to +\infty$), allora esiste un α che soddisfa le condizioni di Armijo e di Wolfe.*

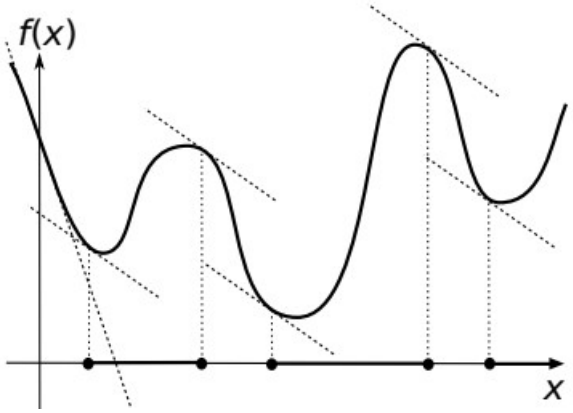

Figura 2.3: I segmenti in grassetto sono le aree in cui vale la condizione di Wolfe

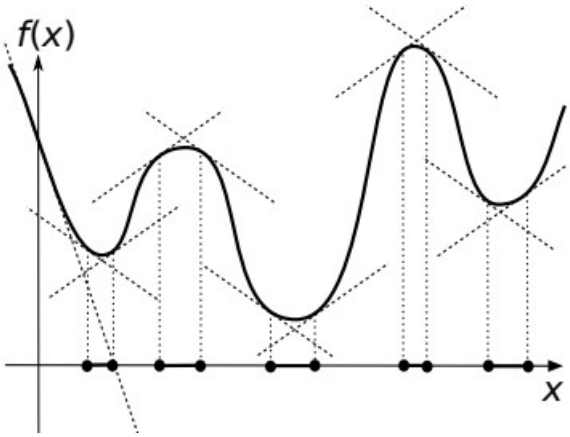

Figura 2.4: I segmenti in grassetto sono le aree in cui vale la condizione di Wolfe forte

Dimostrazione. Definiamo la funzione $g(\alpha) = \varphi(0) + m_1\alpha\varphi'(0)$.

Definiamo $d(\alpha) = g(\alpha) - \varphi(\alpha)$. Questa funzione è la distanza (con segno) tra g e φ in α. Osserviamo che $d(0) = 0$ e

$$d'(\alpha) = m_1\varphi'(0) - \varphi'(\alpha),$$

perciò in particolare

$$d'(0) = (m_1 - 1)\varphi'(0) > 0$$

Se non esistesse $\bar{\alpha} > 0$ tale che $d(\bar{\alpha}) = 0$, φ sarebbe illimitata interiormente: sarebbe infatti per ogni α, $d(\alpha) > 0$ e quindi $g(\alpha) > \varphi(\alpha)$. Ma $g(\alpha) \xrightarrow{\alpha\to\infty} -\infty$, perciò anche $\varphi(\alpha) \xrightarrow{\alpha\to\infty} -\infty$.

Esiste perciò $\bar{\alpha}$ tale che $d(\bar{\alpha}) = 0$. Prendiamo in particolare il più piccolo.

Vediamo ora che vale la condizione di Armijo per ogni $\alpha < \bar{\alpha}$: infatti $\bar{\alpha}$ è il più piccolo $\alpha > 0$ per cui $d(\alpha) = 0$, per cui per ogni $\alpha < \bar{\alpha}$, abbiamo che

$$d(\alpha) > 0,$$

ovvero vale la condizione di Armijo.

Per il teorema di Rolle abbiamo inoltre che esiste $0 < \alpha' < \bar{\alpha}$ tale per cui $d'(\alpha') = 0$, ovvero

$$m_1\varphi'(0) = \varphi'(\alpha').$$

Poiché $m_3 < m_1$,

$$\varphi'(\alpha') > m_3\varphi'(0),$$

cioè α' soddisfa anche la condizione di Wolfe. □

Dato che α esiste, tutto sta nel trovarlo: per farlo basta usare un qualsiasi metodo di ottimizzazione in una variabile, usando come nuova condizione di stop che valgano (2.9) e (2.10), oppure (2.9) e (2.11) se vogliamo la versione forte.

Backtracking Line Search

Una versione ancora più semplice, che controlla solo se è soddisfatta la condizione di Armijo, è la Backtracking Line Search (BLS). Per ogni iterazione della discesa del gradiente ripetiamo la seguente procedura:

1. Prendiamo un passo α iniziale e un numero reale $0 < t < 1$
2. Se vale $\varphi(\alpha) \leq \varphi(0) + m_1\alpha\varphi(\alpha)$ finiamo l'iterazione, altrimenti

3. Ripetiamo con un nuovo $\alpha' = t\alpha$

Ripetendo 2. e 3. finché non è soddisfatta la (2.9), garantiamo di trovare un α che soddisfi la condizione: infatti la successione $\{t^k\alpha\}$ tende a 0, e per α che tende a 0 la condizione di Armijo è sempre soddisfatta. Evitiamo inoltre "Cariddi" cioè di muoverci troppo poco, perché nei primi passi di ogni iterazione possiamo provare un α anche molto grande

Algoritmo 2 Backtracking Line Search

```
procedure α = BLS ( φ , α , m1 , τ )        // τ < 1
  while( φ( α ) > φ( 0 ) + m1 α φ'( 0 ) ) do α ← τ α;
```

In che condizioni possiamo garantire che la BLS funzioni? Ricordiamo che per ogni i esiste un α_i tale che per ogni $\alpha < \alpha_i$, la condizione di Armijo è soddisfatta all'iterazione i della discesa del gradiente.

Assumiamo per semplicità che per ogni iterazione $\alpha = 1$ (intendiamo l'α iniziale dato in input). Allora sicuramente il passo all'iterazione i sarà $t^{k_i} < 1$, con $k_i > \min\{k \mid t^k < \alpha_i\}$. Se garantiamo che esista un $\bar{\alpha}$ tale per cui $\alpha_i > \bar{\alpha}$ per ogni i, allora si ha anche che $k_i > \bar{k} = \min\{k \mid t^k < \bar{\alpha}\}$. Dunque per la Proposizione 2.2.2, si ha o la convergenza del metodo ad un punto stazionario o che f è illimitata inferiormente.

Per garantire che un tale $\bar{\alpha}$ esista, possiamo assumere che f sia L-smooth: in questo caso ricordiamo che φ è $(L||\nabla f(\mathbf{x})||^2)$- smooth. Ricordiamo anche che esiste un $\alpha' < \bar{\alpha}$ che soddisfa anche la condizione di Wolfe (visto nella dimostrazione della Proposizione 2.2.2). Dunque vale

$$L||\nabla f(\mathbf{x})||^2\alpha' \geq \varphi'(\alpha') - \varphi'(0) > (1 - m_3)(-\varphi'(0)) = (1 - m_3)||\nabla f(\mathbf{x})||^2$$

Semplificando si ottiene allora che

$$\alpha' > \frac{1 - m_3}{L}.$$

Perciò $\bar{\alpha} > \alpha'$ esiste ed è maggiore di 0.

Se f è anche τ-convessa si può dimostrare che l'ordine di convergenza è lineare, con $r \approx 1 - \frac{\tau}{L}$, dipendente da m_1 e m_3.

2.3 Metodo di Newton in più variabili

Abbiamo finora assunto che la direzione $\mathbf{d}^i$ sia l'antigradiente in $\mathbf{x}^i$. Tuttavia esistono infinite direzioni di decrescita in $\mathbf{x}^i$ Abbiamo usato l'antigradiente per due motivi:

1. Lontano da un punto stazionario $\varphi_i'(0) = -||\nabla f(\mathbf{x}^i)||^2$ è molto negativa $\Rightarrow$ la decrescita è veloce.

2. Se i passi α_i non tendono a 0 troppo velocemente, abbiamo una decrscita significativa ad ogni passo, a meno che $||\nabla f(\mathbf{x}^i)|| \to 0$ (che è comunque ciò che vogliamo, stiamo cercando un punto stazionario).

2. Non dipende sostanzialmente dalla direzione scelta, e ci sono molte altre direzioni in cui vale 1. a meno di un fattore moltiplicativo.

Esempio. Se $\mathbf{d}^i = R\left(-\nabla f(\mathbf{x}^i)\right)$ con R una matrice di rotazione di $\frac{\pi}{4}$,

$$\varphi_i'(0) = -||\nabla f(\mathbf{x}^i)||^2 \cos\left(\frac{\pi}{4}\right) = -||\nabla f(\mathbf{x}^i)||^2 \frac{\sqrt{2}}{2} < 0,$$

e la dimostrazione di convergenza funziona allo stesso modo.

In effetti ogni direzione tale per cui $\nabla f(\mathbf{x}^i) \cdot \mathbf{d}^i < 0$ è di decrescita. Esiste cioè un intero semispazio di direzioni di decrescita. Per garantire la convergenza al minimo locale, dobbiamo però accertarci che $\nabla f(\mathbf{x}^i) \cdot \mathbf{d}^i$ non converga a 0 troppo velocemente:

Teorema 2.3.1 (Zoutendijk)**.** *Sia $f \in C^1$, L-smooth e con $f_* > -\infty$, se $\{\mathbf{d}^i\}$ sono le direzioni di decrescita con $\varphi_i'(0) = -||\nabla f(\mathbf{x}^i)||^2 \cos\theta^i$, allora se valgono le condizioni di Armijo e Wolfe vale anche*

$$\sum_{i=0}^{\infty} \cos^2\theta^i ||\nabla f(\mathbf{x}^i)||^2 < \infty$$

Come conseguenza abbiamo che se $\sum_{i=0}^{\infty} \cos^2\theta^i \to \infty$, allora per forza $||\nabla f(\mathbf{x}^i)|| \to 0$. Ciò vuol dire che, se $\mathbf{d}^i$ non diventa ortogonale a $\nabla f(\mathbf{x}^i)$ troppo velocemente, la convergenza è garantita.

Ci chiediamo perciò come scegliere delle direzioni $\mathbf{d}^i$ migliori dell'antigradiente. Per farlo dobbiamo guardare al modello del secondo ordine.

2.3.1 Newton per funzioni convesse

Usiamo il modello del secondo ordine

$$T^{(2)}_{\mathbf{x}^0}(\mathbf{x}) = f(\mathbf{x}^0) + \nabla f(\mathbf{x})^T(\mathbf{x}-\mathbf{x}^0) + \frac{1}{2}(\mathbf{x}-\mathbf{x}^0)^T\nabla^2 f(\mathbf{x}^0)(\mathbf{x}-\mathbf{x}^0)$$

Nel caso localmente convesso il modello è definito positivo, perciò esiste un minimo: tale minimo è in $\bar{\mathbf{x}} = \mathbf{x}^0 + \mathbf{d}$ con

$$\mathbf{d} = -\left(\nabla^2 f(\mathbf{x}^0)\right)^{-1}\nabla f(\mathbf{x}^0)$$

Il metodo di Newton non è globalmente convergente, esattamente come in una variabile, perciò va globalizzato: per farlo cominciamo con una LS Backtracking o con le condizioni di Armijo-Wolfe con $\alpha = 1$ iniziale. Per dimostrare che un algoritmo del genere funziona abbiamo 3 Teoremi da dimostrare:

Teorema 2.3.2. *Sia $f \in C^2$ e L-smooth e τ-convessa. Chiamiamo θ_i l'angolo tra $\mathbf{d}^i = -\left(\nabla^2 f(\mathbf{x}^i)\right)^{-1}\nabla f(\mathbf{x}^i)$ e $-\nabla f(\mathbf{x}^i)$. Si ha allora che*

$$\cos\theta^i \leq -\frac{\tau}{L}$$

Corollario 2.3.1. *Grazie al Teorema di Zoutendijk, il teorema 2.3.2 ci garantisce che il metodo di Newton con le condizioni di Armijo-Wolfe o con BLS converga.*

Dimostrazione. Si ha che $\nabla f(\mathbf{x}^i)^2\,\mathbf{d}^i = -\nabla f(\mathbf{x}^i)$, perciò

$$\nabla f(\mathbf{x}^i)^T\mathbf{d}^i = (\mathbf{d}^i)^T\left(\nabla^2 f(\mathbf{x}^i)\right)\mathbf{d}^i \leq -\tau||\mathbf{d}^i||^2$$

e

$$||\nabla f(\mathbf{x}^i)|| = ||\nabla f(\mathbf{x}^i)^2\,\mathbf{d}^i|| \leq \|\nabla f(\mathbf{x}^i)^2\|\|\mathbf{d}^i\| \leq L\|\mathbf{d}^i\|.$$

Dunque

$$\cos\theta^i = \frac{\nabla f(\mathbf{x}^i)^T\mathbf{d}^i}{\|\nabla f(\mathbf{x}^i)\|\|\mathbf{d}^i\|} \leq -\frac{\tau}{L}$$

□

Teorema 2.3.3. *Se $f \in C^3$, e esiste un punto x_* tale che $\nabla f(\mathbf{x}_*) = 0$ e $\nabla^2 f(\mathbf{x}_*)$ è definita positiva, allora esiste un $\delta > 0$ tale che iniziando il metodo di Newton da $\mathbf{x}^0 \in \mathcal{B}(\mathbf{x}*, \delta)$, il metodo converge con tasso di convergenza quadratico*

Dimostrazione. Lasciata al lettore, è del tutto simile al caso unidimensionale (Teorema 1.2.2) □

Teorema 2.3.4. *Sia f come nel Teorema 2.3.2. Se $\mathbf{x}^i \to \mathbf{x}_*$, allora esiste un $h \in \mathbb{N}$ tale che, per ogni $i \geq h$, la condizione di Armijo (2.9) è sempre soddisfatta per $\alpha^i = 1$ (richiedendo $m_1 \leq \frac{1}{2}$: $m_1 > \frac{1}{2}$ taglia via il minimo se f è quadratica)*

Dimostrazione. $\mathbf{x}^i \to \mathbf{x}_*$ implica che $\|\nabla f(\mathbf{x}^i)\| \to 0$ e quindi anche $\|\mathbf{d}^i\| \to 0$. Perciò per la formula di Taylor al secondo ordine

$$f(\mathbf{x}^{i+1}) = f(\mathbf{x}^i + \mathbf{d}^i) = f(\mathbf{x}^i) + \nabla f(\mathbf{x}^i)^T \mathbf{d}^i + \frac{1}{2} \left(\mathbf{d}^i\right)^T \nabla^2 f(\mathbf{x}^i) \mathbf{d}^i + R(\mathbf{d}^i)$$

$\mathbf{d}^i = -\nabla^2 f(\mathbf{x}^i)^{-1} \nabla f(\mathbf{x}^i)$ perciò

$$\begin{aligned} f(\mathbf{x}^{i+1}) = f(\mathbf{x}^i) \quad & - \nabla f(\mathbf{x}^i)^T \nabla^2 f(\mathbf{x}^i)^{-1} \nabla f(\mathbf{x}^i) \\ & + \frac{1}{2} \left(-\nabla^2 f(\mathbf{x}^i)^{-1} \nabla f(\mathbf{x}^i)\right)^T \nabla^2 f(\mathbf{x}^i) \left(-\nabla^2 f(\mathbf{x}^i)^{-1} \nabla f(\mathbf{x}^i)\right) \\ & + R(\mathbf{d}^i) \\ = f(\mathbf{x}^i) \quad & - \nabla f(\mathbf{x}^i)^T \nabla^2 f(\mathbf{x}^i)^{-1} \nabla f(\mathbf{x}^i) \\ & + \frac{1}{2} \nabla f(\mathbf{x}^i)^T \nabla^2 f(\mathbf{x}^i)^{-1} \nabla f(\mathbf{x}^i) + R(\mathbf{d}^i) \\ = f(\mathbf{x}^i) \quad & + \frac{1}{2} \nabla f(\mathbf{x}^i)^T \mathbf{d}^i + R(\mathbf{d}^i). \end{aligned}$$

Abbiamo inoltre assunto che f sia τ-convessa, perciò

$$-\nabla f(\mathbf{x}^i)^T \mathbf{d}^i = (\mathbf{d}^i)^T \nabla^2 f(\mathbf{x}^i) \mathbf{d}^i \geq \tau \|\mathbf{d}^i\|^2,$$

Inoltre $R(\mathbf{d}^i) = o\left(\|\mathbf{d}^i\|\right)$, dunque dato $m_1 < \frac{1}{2}$ esiste un $\bar{i}$ tale per cui, per ogni $i > \bar{i}$,

$$f(\mathbf{x}^{i+1}) \leq f(\mathbf{x}^i) + m_1 \nabla f(\mathbf{x}^i)^t \mathbf{d}^i,$$

cioè vale la condizione di Armijo con $\alpha^i = 1$. □

Il metodo di Newton si divide perciò in una prima fase "globale", dove α varia per soddisfare le condizioni di Armijo e Wolfe, e in una seconda fase di "Newton puro", in cui si ha convergenza quadratica e $\alpha = 1$.

Metodo di Newton come metodo del gradiente in uno spazio modificato

Una possibile interpretazione del metodo di Newton è quella che vede l'algoritmo come un metodo del gradiente in uno spazio modificato. Prendiamo per esempio una funzione quadratica $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q\mathbf{x} + \mathbf{q}^T\mathbf{x}$. In questo caso il modello del secondo ordine è f stesso, perciò compiendo un passo dell'algoritmo di Newton ci troviamo subito in $\mathbf{x}_* = -Q^{-1}\mathbf{q}$, con $\nabla f(\mathbf{x}_*) = 0$. Quindi $\mathbf{x}_*$ è un punto stazionario e l'algoritmo impiega un solo passo prima di terminare. Abbiamo visto che l'algoritmo del gradiente è molto efficiente per funzioni quadratiche in cui l'autovalore più grande e quello più piccolo di Q, rispettivamente $\lambda_1(Q)$, e $\lambda_n(Q)$, sono molto vicini (Q è ben condizionata).

Il metodo di Newton applicato ad una funzione quadratica corrisponde ad un metodo del gradiente su un'altra funzione quadratica

$$h(\mathbf{z}) = \frac{1}{2}\mathbf{z}^t Q'\mathbf{z} + q^T\mathbf{z}$$

in cui la matrice Q' è molto ben condizionata:

Definizione. Definiamo la *"radice quadrata"* di Q come la matrice R tale che $R^2 = Q$. In generale tale matrice può non esistere (come matrice reale) o non essere unica.

Se Q è semidefinita positiva possiamo calcolare la decomposizione spettrale $Q = H\Lambda H^T$ e prendere $R = H\sqrt{\Lambda}H^T$.

Se inoltre Q è non singolare, anche R è non singolare perciò esiste un'inversa R^{-1} e le funzioni

$$\mathbf{x} \to \mathbf{z} = R\mathbf{x} \quad \mathbf{z} \to \mathbf{x} = R^{-1}\mathbf{z}$$

sono bigezioni (più precisamente isomorfismi lineari) dello spazio $\mathbb{R}^n$. In questo modo possiamo definire

$$h(\mathbf{z}) = f(R^{-1}\mathbf{z}) = \frac{1}{2}\mathbf{z}^T I\mathbf{z} + \mathbf{q}R^{-1}\mathbf{z}.$$

notiamo che $\nabla^2 h(\mathbf{z}) = I$, dunque "nello spazio $\mathbf{z}$ l'hessiana di f è l'identità". In questo spazio il metodo del gradiente per minimizzare h è molto efficiente: detto $\mathbf{g} = \nabla h(\mathbf{z}) = -z - R^{-1}\mathbf{q}$, si ha

$$\nabla h(\mathbf{z} + \mathbf{g}) = \nabla h(-R^{-1}\mathbf{q}) = -R^{-1}\mathbf{q} + R^{-1}\mathbf{q} = 0.$$

Lo svantaggio di questa trattazione è che calcolare R è molto costoso, tuttavia $z = R\mathbf{x}$ non è l'unica scelta, soprattutto se Q non è definita positiva.

2.3.2 Newton per funzioni non convesse

Modificare l'hessiana

Abbandoniamo ora l'ipotesi che ci ha accompagnato nelle sezioni precedenti, la ($\tau-$)convessità di f, e vediamo come garantire comunque la convergenza del metodo di Newton. Abbiamo visto che il metodo di Newton corrisponde ad una deformazione dello spazio che rende $\nabla^2 f(\mathbf{x})$ meglio condizionata, e tale dilatazione è appunto prodotta dalla moltiplicazione per $-\nabla^2(\mathbf{x})^{-1}$. Specialmente se $\nabla^2 f(\mathbf{x})$ non è definita positiva, tale scelta può non essere la migliore. In generale si può pensare di prendere per ogni i, $\mathbf{d}^i = -H_i \nabla f(\mathbf{x}^i)$, per qualche matrici H_i. I metodi in cui l'iterazione è eseguita in questo modo vengono detti metodi *quasi-Newton*.

Teorema 2.3.5. *Se scegliamo ad ogni passo una matrice H_i tale che gli autovalori di H_i sono contenuti in $[\tau, L]$ per due valori $\tau < L$ (cioè $\tau\, I \preceq H_i \preceq L\, I$), e prendiamo $\mathbf{d}^i = -H_i^{-1}\nabla f(\mathbf{x}^i)$, allora se valgono le condizioni di Armijo-Wolfe si ha convergenza globale.*

Dimostrazione. È uguale al Teorema 2.3.2 con H_i al posto di $\nabla f(\mathbf{x}^i)$. Avevamo infatti usato le ipotesi di L-smoothness e τ- convessità per garantire che $\tau \preceq \nabla^2 f(\mathbf{x}^i) \preceq L$. □

Come possiamo trovare H_i come nel teorema precedente? Se $\nabla^2 f(\mathbf{x}^i)$ non è semi-definita positiva, esistono degli autovalori minori di 0 di $\nabla^2 f(\mathbf{x}^i)$.

In teoria se $\lambda_1 > \cdots > \lambda_n$, e $\lambda_n < 0$ è il più piccolo di essi,basta prendere $H_i = \nabla^2 f(\mathbf{x}^i) + \epsilon_i I$ con $\epsilon_i > -\lambda_n$ per garantire che H_i sia definita positiva. Tuttavia incontriamo due problemi:

Problema numerico: In precisione di macchina valori troppo piccoli ($\approx 10^{-16}$ ma anche più grandi) potrebbero essere considerati 0.

Problema algoritmico: Se $\lambda_n + \epsilon_i$ è troppo piccolo, si ha come nella discussione sull'efficienza del metodo del gradiente, linee di livello molto allungate (nel nuovo spazio) e il metodo è poco efficiente.

Soluzione: Prendiamo $\epsilon_i = \max\{0, \delta - \lambda_n\}$ con $\delta > 0$ un parametro piccolo fissato. In questo modo se già $\nabla^2 f(\mathbf{x}^i)$ ha tutti autovalori maggiori di δ, si fa un passo normale del metodo di Newton, in caso contrario modifichiamo l'hessiana come sopra.

Si dimostra che questa scelta di H_i risolve il problema di ottimizzazione

$$\min \left\{ \|H_i - \nabla^2 f(\mathbf{x}^i)\|_2 \ : \ H_i \succeq \delta \right\}.$$

Possiamo anche puntare a minimizzare una norma di matrice diversa da $\|\|_2$: ad esempio usando la norma di Frobenius e risolvendo

$$\min \left\{ \|H_i - \nabla^2 f(\mathbf{x}^i)\|_F \ : \ H_i \succeq \delta \right\}$$

si ottiene che $H_i = H\bar{\Lambda}H^T$ con

$$\bar{\Lambda} = \begin{pmatrix} \bar{\lambda_1} & & \\ & \ddots & \\ & & \bar{\lambda_n} \end{pmatrix}$$

e $\bar{\lambda_i} = \max\{\lambda_i, \delta\}$. In ogni caso per trovare H_i appropriato abbiamo bisogno di $O(n^3)$ operazioni, che sono troppe per ottimizzazione per larga scala, ad esempio $n \geq 10^4$.

Regioni di fiducia

> Round balls are better than kinky balls
>
> Frangioni ti fa cock-rating

Un'altra tecnica per far funzionare il metodo di Newton senza che $\nabla^2 f(\mathbf{x}^i)$ sia definita positiva è cercare delle regioni di fiducia. Se l'Hessiana non è semi-definita positiva, il modello del secondo ordine non ha minimo in $\mathbb{R}^n$, perchè ci sono delle direzioni di curvatura negativa. Invece che escludere tali regioni come abbiamo fatto nella sezione precedente, vogliamo sfruttare tali direzioni di decrescita. Minimizziamo allora il modello del secondo ordine attorno a $\mathbf{x}^i$, $T^{(2)}_{\mathbf{x}^i}(\mathbf{y})$, in un insieme *compatto* $\mathcal{T}_i$ detto *regione di fiducia* attorno a $\mathbf{x}^i$. Cerchiamo cioè

$$\mathbf{x}^{i+1} \in \arg\min_{\mathbf{y} \in \mathcal{T}_i} T^{(2)}_{\mathbf{x}^i}(\mathbf{y})$$

Il problema diventa un problema di ottimizzazione vincolato, che è NP-hard per molte scelte di $\mathcal{T}_i$, ad esempio prendendo le palle (chiuse) in norma 1 o in norma ∞. Se però $\mathcal{T}_i = \bar{\mathcal{B}}_2(\mathbf{x}^i, r)$ è la palla chiusa in norma 2, tale problema è risolvibile. Abbiamo incontrato uno dei primi casi di ottimizzazione vincolata, che studieremo nel prossimo capitolo.

2.4 Metodi quasi-Newton

Vediamo come ricavare dei metodi in qualche modo simili a quello di Newton senza calcolare però l'Hessiana. Tale calcolo è infatti molto dispendioso se n è molto grande. Prendiamo perciò un metodo del tipo

$$\mathbf{x}^{i+1} = \mathbf{x}^i + \alpha_i \mathbf{d}^i, \quad \mathbf{d}^i = -H^i \nabla f(\mathbf{x}^i). \tag{2.12}$$

Lo spazio delle H_i per cui si ha convergenza è grande:

Teorema 2.4.1. *Se $f \in C^2$ e*

$$\lim_{i\to\infty} \frac{\| \left(H_i - \nabla^2 f(\mathbf{x}^i)\right) d^i \|}{\|\mathbf{d}^i\|} = 0,$$

si ha convergenza superlineare del metodo

$$\mathbf{x}^{i+1} = x^i + \mathbf{d}^i, \quad \mathbf{d}^i = -H_i^{-1} \nabla f(\mathbf{x}^i).$$

Con un metodo quasi-Newton, per ogni i stiamo implicitamente costruendo, e minimizzando, il modello

$$m_i(\mathbf{x}) = \nabla f(\mathbf{x}^i)^T (\mathbf{x} - \mathbf{x}^i) + (\mathbf{x} - \mathbf{x}^i)^T H_i (\mathbf{x} - \mathbf{x}^i)$$

Alcune proprietà di m_i e H_i che vorremmo sono:

1. Per ogni i, H_i è definita positiva (i modelli sono fortemente convessi);
2. $\nabla m_{i+1}(\mathbf{x}^i) = \nabla f(\mathbf{x}^i)$ (il nuovo modello conviene con i precedenti sulle vecchie informazioni: infatti anche $\nabla m_i(\mathbf{x}^i) = \nabla f(\mathbf{x}^i)$);
3. $\|H_{i+1} - H_i\|$ è piccolo (il nuovo modello è simile a quello precedente) .

La proprietà 2. è equivalente a

2'. $H_{i+1}(\mathbf{x}^{i+1} - \mathbf{x}^i) = \nabla f(\mathbf{x}^{i+1}) - \nabla f(\mathbf{x}^i)$ (equazione della secante)

Notazione: denotiamo con $\mathbf{s}^i = \mathbf{x}^{i+1} - \mathbf{x}^i = \alpha_i \mathbf{d}^i$, $\mathbf{y}^i = \nabla f(\mathbf{x}^{i+1}) - \nabla f(\mathbf{x}^i)$ Si nota immediatamente che con questa nuova notazione, l'equazione delle secanti 2'. si può riscrivere come

$$H_{i+1}\mathbf{s}^i = \mathbf{y}^i \tag{2.13}$$

L'equazione 2.13 implica che $\mathbf{s}^i \cdot \mathbf{y}^i = (\mathbf{s}^i)^T H_{i+1} s^i$. Se vale anche la proprietà 1. vale dunque $\mathbf{s}^i \cdot \mathbf{y}^i > 0$. Tale proprietà è detta *condizione di curvatura* ed è spesso scritta come

$$\rho_i = \frac{1}{\mathbf{s}^i \cdot \mathbf{y}^i} > 0 \tag{2.14}$$

Vogliamo perciò garantire che la condizione sulla curvatura (2.14) sia soddisfatta.

Proposizione 2.4.1. *La condizione di Wolfe (2.10) implica la condizione sulla curvatura.*

Dimostrazione.

$$\varphi'(\alpha_i) = \nabla f(\mathbf{x}^{i+1}) \cdot \mathbf{d}^i \geq m_3 \varphi'(0) = m_3 \left(\nabla f(\mathbf{x}^i) \cdot \mathbf{d}^i\right)$$

dunque

$$\left(\nabla f(\mathbf{x}^{i+1}) - \nabla f(\mathbf{x}^i)\right) \cdot \mathbf{d}^i = \mathbf{y}^i \cdot \mathbf{d}^i \geq (m_3 - 1)\varphi'(0) > 0.$$

Ma $\mathbf{s}^i = \alpha_i \mathbf{d}^i$ perciò

$$\mathbf{y}^i \cdot \mathbf{s}^i \geq \alpha(m_3 - 1)\phi'(0) > 0$$

□

Dunque con una Armijo-Wolfe Line Search la condizione (2.14) è sempre soddisfatta.

2.4.1 DFP

Come detto nella proprietà 3. , vorremmo che $H_{i+1} - H_i$ sia piccolo. Una soluzione è prendere

$$H_{i+1} = \arg\min\{\|H - H_i\| : H \succeq 0, H \text{ soddisfa 2'.}\} \tag{2.15}$$

per una qualche norma a nostra scelta.

Definizione. [5] Data una matrice W, la norma di Frobenius pesata è $\|A\|_W = \|W^{\frac{1}{2}} A W^{\frac{1}{2}}\|$.

[5] Questa definizione e le seguenti sono solo per dare un'Idea da dove vengano fuori le formule DFP e BFGS. Non sono necessarie ai fini dell'implementazione dell'algoritmmo

Possiamo scegliere qualsiasi W tale che vale $W\mathbf{y}^i = \mathbf{s}^i$, e in particolare possiamo prendere, per ogni i, $W = \bar{G}_i^{-1}$, dove $\bar{G}_i$ è l'Hessiana media, definita da

$$\bar{G}_i = \int_0^1 \nabla^2 f(\mathbf{x}^i + t\alpha_i \mathbf{d}^i) dt$$

Scegliendo questa norma e questa matrice, H_{i+1} è facilmente calcolabile attraverso la formula di Davidon-Fletcher-Powell:

Proposizione 2.4.2. *La soluzione del problema (2.15) con la norma di Frobenius indotta da $\bar{G}^{-1}$ è*

$$\text{(DFP-H)} \qquad H_{i+1} = \left(I - \rho_i \mathbf{y}^i (\mathbf{s}^i)^T\right) H_i \left(I - \rho_i \mathbf{s}^i (\mathbf{y}^i)^T\right) + \rho_i \mathbf{y}^i (\mathbf{y}^i)^T \quad (2.16)$$

Per applicare il metodo non ci interessa in realtà H_i, ma la sua inversa $B_i = H_i^{-1}$. B_i si può calcolare altrettanto facilmente come segue:

$$\text{(DFP-B)} \qquad B_{i+1} = B_i + \rho_i \mathbf{s}^i (\mathbf{s}^i)^T - \frac{\left(B_i \mathbf{y}^i (\mathbf{y}^i)^T B_i\right)}{(\mathbf{y}^i)^T B_i \mathbf{y}^i} \quad (2.17)$$

Ad ogni passo quindi correggiamo B_i sommandogli due matrici di rango 1. Calcolare tali matrici è poco costoso: il costo è dominato dallo svolgimento di 3 prodotti matrice-vettore, perciò il metodo DFP richiede $O(n^2)$ operazioni per iterazione.

2.4.2 BFGS

Invece che risolvere il problema (2.15), possiamo risolvere il problema equivalente per B_i, che si ottiene scambiando i ruoli di $\mathbf{s}^i$ e $\mathbf{y}^i$:

$$B'_{i+1} = \arg\min\{\|B - B_i\| : B \succeq 0, B\mathbf{y}^i = \mathbf{s}^i\} \quad (2.18)$$

Usando la norma definita in precedenza con $\bar{G}_i$ (senza l'inversa), si ottengono in modo simmetrico scambiando $H \leftrightarrow B$, $\mathbf{y} \leftrightarrow \mathbf{s}$, le formule di Broyden-Fletcher-Goldfarb-Shanno:

$$\text{(BFGS-H)} \qquad H'_{i+1} = H'_i - \rho_i \mathbf{y}^i (\mathbf{y}^i)^T - \frac{H_i \mathbf{s}^i (\mathbf{s}^i)^T H'_i}{(\mathbf{s}^i)^T H_i \mathbf{s}^i} \quad (2.19)$$

$$\text{(BFGS-B)} \qquad B'_{i+1} = \left(I - \rho_i \mathbf{s}^i (\mathbf{y}^i)^T\right) B'_i \left(I - \rho_i \mathbf{y}^i (\mathbf{s}^i)^T\right) - \rho_i \mathbf{s}^i (\mathbf{s}^i)^T \quad (2.20)$$

In che modo scegliere B^0? una prima idea è γI per qualche γ. Un'alternativa è calcolare un'approssimazione di $\nabla^2 f(\mathbf{x}^0)$ attraverso il metodo delle differenze finite. Si può in qualche modo pensare ai metodi DFP e BFGS come metodi per imparare l'hessiana $\nabla^2 f(\mathbf{x})$ tramite esempi dati da $\nabla f(\mathbf{x})$. Lo studio della convergenza del metodo è complicata ma nella pratica non è molto distante alla velocità del metodo di Newton, ed è a volte anche meno incline a fermarsi in minimi locali: all'inizio il metodo è $\approx$ gradiente, quindi riesce a sfuggire meglio a minimi locali, mentre nelle ultime iterazioni, avendo imparato meglio $\nabla^2 f(\mathbf{x})$, si possono vedere le proprietà di convergenza veloce del metodo di Newton.

2.4.3 Limited-memory BFGS

Possiamo essere ancora più efficienti ad ogni iterazione: notiamo che sviluppando (BFGS-B) e chiamando $V^i = I - \rho_i \mathbf{y}^i (\mathbf{s}^i)^T$, otteniamo

$$B^{i+1} = (V^i)^T B^i V^i + \rho^i \mathbf{s}^i (\mathbf{s}^i)^T \tag{2.21}$$

Possiamo allora provare a sviluppare la ricorsione, ottenendo

$$\begin{aligned} B^{i+1} =& (V^0 \cdot V^1 \cdot \ldots \cdot V^i)^T B^0 (V^0 \cdot V^1 \cdot \ldots \cdot V^i) \\ &+ \rho_0 (V^1 \cdot \ldots \cdot V^i)^T \mathbf{s}^0 \cdot (\mathbf{s}^0)^T (V^1 \cdot \ldots \cdot V^i) \\ &+ \rho_1 (V^2 \cdot \ldots \cdot V^i)^T \mathbf{s}^1 \cdot (\mathbf{s}^1)^T (V^2 \cdot \ldots \cdot V^i) \\ &+ \ldots \\ &+ \rho_i \mathbf{s}^i (\mathbf{s}^i)^T \end{aligned} \tag{2.22}$$

In questo modo è facile vedere che B^{i+1} può essere calcolato solo tramite prodotti scalari tra vettori, che richiedono $O(n)$ operazioni, se svolgiamo i prodotti in un ordine "furbo". Se lasciamo la formula così com'è ci servono comunque $O(n^2)$ operazioni totali. Possiamo perciò troncare la formula dopo k passi:

$$\begin{aligned} B^{i+1} =& (V^{i-k} \cdot V^{i-k+1} \cdot \ldots \cdot V^i)^T B^{i-k} (V^{i-k} \cdot V^{i-k+1} \cdot \ldots \cdot V^i) \\ &+ \rho_{i-k} (V^{i-k+1} \cdot \ldots \cdot V^i)^T \mathbf{s}^{i-k} \cdot (\mathbf{s}^{i-k})^T (V^{i-k+1} \cdot \ldots \cdot V^i) \\ &+ \rho_{i-k+1} (V^{i-k+2} \cdot \ldots \cdot V^i)^T \mathbf{s}^{i-k+1} \cdot (\mathbf{s}^{i-k+1})^T (V^{i-k+2} \cdot \ldots \cdot V^i) \\ &+ \ldots \\ &+ \rho_i \mathbf{s}^i (\mathbf{s}^i)^T \end{aligned} \tag{2.23}$$

Osservazione 6. In pratica, non serve che in (2.23) B^{i-k} sia davvero la B calcolata all'iterazione $i-k$, ma la matrice può essere scelta in modo arbitrario per ogni i, in modo sparso: ad esempio $B^{i-k} = \gamma_i I$ con $\gamma_i = \mathbf{s}^i \cdot y^{i-1}/\|y^{i-1}\|^2$. In questo modo si può dimostrare che servono $O(kn)$ operazioni totali.

Se n è molto grande, scegliendo un $k \ll n$ si può ottenere un vantaggio computazionale significativo, ma l'efficienza del metodo in termini di operazioni se k è molto piccolo è paragonabile al metodo del gradiente. Se invece k cresce l'efficienza comincia ad essere simile a quella della BFGS, e quindi si avvicina al metodo di Newton.

2.5 Gradiente coniugato e Heavy Ball

Se vogliamo dei metodi ancora meno costosi, invece che modificare la direzione di decrescita d_i con una matrice H, possiamo deviarla sommando un vettore $\mathbf{v}^i$, ottenendo $\mathbf{d}^i = -\nabla f(\mathbf{x}^i) + \mathbf{v}^i$. Scegliamo $\mathbf{v}^i = \beta_i \mathbf{d}^{i-1}$, con $\beta_i > 0$: in questo modo, assumendo $\mathbf{v}^0 = 0$, otteniamo che

$$\mathbf{d}^i = -\sum_{j=0}^{i} \gamma_j \nabla f(\mathbf{x}^j), \quad \gamma_j = \sum_{k=j+1}^{i} \beta_k$$

cioè ad ogni passo stiamo tenendo conto di tutta la storia precedente.

Tuttavia il problema di scegliere i β_i tali per cui le direzioni sono di discesa è non banale: se per esempio $\beta_i \to 0$, la direzione sarà di discesa per i abbastanza grande, tuttavia $\mathbf{d}^i \to \nabla f(x^i)$, dunque ci possiamo aspettare una convergenza lenta come quella della discesa del gradiente.

2.5.1 Metodo del gradiente coniugato

Se applichiamo il metodo del gradiente con Line Search esatta ad una funzione quadratica, abbiamo visto che $\mathbf{d}^{i+1} \perp \mathbf{d}^i$: questo accade perché $\mathbf{x}^{i+1}$ è il minimo di f lungo $\mathbf{d}^i$, ovvero nel sottospazio vettoriale (unidimensionale) generato da $\mathbf{d}^i$ che denotiamo con $\text{span}\{\mathbf{d}^i\}$. Tale proprietà si perde al passo successivo: niente ci garantisce che $\mathbf{x}^{i+2}$ sia ancora il minimo di f lungo $\mathbf{d}^i$ L'idea è quindi richiedere che $\mathbf{x}^{i+1}$ sia il minimo in tutto il sottospazio $\text{span}\{\mathbf{d}^1, \dots, \mathbf{d}^i\}$ Ciò è possibile prendendo $\mathbf{d}^i = -\nabla f(\mathbf{x}^i) + \beta_i \mathbf{d}^{i-1}$, con

$$\beta_i = \frac{\nabla f(\mathbf{x}^i)^T Q\, \mathbf{d}^i}{(\mathbf{d}^{i-1})^T Q\, \mathbf{d}^{i-1}}. \tag{2.24}$$

La formula (2.24) si può riscrivere più semplicemente come

$$\beta_i = \frac{\|\nabla f(\mathbf{x}^i)\|^2}{\|\nabla f(\mathbf{x}^{i-1})\|^2} \tag{2.25}$$

Questa formula è detta di *Fletcher-Reeves*.

Queste considerazioni portano al seguente algoritmo:

Algoritmo 3 Algoritmo del gradiente coniugato per funzioni quadratiche

```
procedure x = NCG ( f , x , ε )
  ∇f⁻ = 0;
  while( ‖ ∇f( x ) ‖ > ε ) do
    if( ∇f⁻ = 0 ) then d ← −∇f( x );
      else { β = ‖ ∇f( x ) ‖² / ‖ ∇f⁻ ‖²; d ← −∇f( x ) + βd⁻; }
    α ← LS( f , x , d ); x ← x + αd; d⁻ ← d; ∇f⁻ ← ∇f( x );
```

Per f non quadratica la Line Search esatta non è possibile, o comunque non conveniente, quindi possiamo escludere che $\mathbf{x}^{i+1}$ sia il minimo nella direzione $\mathbf{d}^i$, e a maggior ragione nel sottospazio generato da tutte le direzioni precedenti. Ci sono tuttavia alcuni modi di scegliere β_i in modo che $\mathbf{x}^i$ si avvicini a tale minimo. Tali formule coincidono con la (2.25) se f è quadratica:

Polak-Ribière:

$$\beta_i^{PR} = \frac{(\nabla f(\mathbf{x}^i) - \nabla f(\mathbf{x}^{i-1}) \cdot \nabla f(\mathbf{x}^i)}{\|\nabla f(\mathbf{x}^{i-1})\|^2}$$

Hestenes-Stiefel:

$$\beta_i^{HS} = \frac{(\nabla f(\mathbf{x}^i) - \nabla f(\mathbf{x}^{i-1})) \cdot \nabla f(\mathbf{x}^i)}{(\nabla f(\mathbf{x}^i) - \nabla f(\mathbf{x}^{i-1})) \cdot \mathbf{d}^{i-1}}$$

Day-Yuan:

$$\beta_i^{DY} = \frac{\|\nabla f(\mathbf{x}^i)\|^2}{(\nabla f(\mathbf{x}^i) - \nabla f(\mathbf{x}^{i-1})) \cdot \mathbf{d}^{i-1}}$$

Tale modo di scegliere $\mathbf{d}^i$ viene applicato assieme ad una Armijo-Wolfe Line Search.

In generale la convergenza dipende dal tipo di β_i scelto. Assumendo ragionevoli ipotesi su f, si può dimostrare che scegliendo β_i come nella formula di Fletcher-Reeves (2.25), se la line search che soddisfa la condizione di Wolfe forte, con $m_1 < m_3 < \frac{1}{2}$.

Usando β_i^{PR} o β_i^{HS}, una buona idea è il *restart*, ovvero scegliere

$$\beta_i = \max\{\beta_i^*, 0\}$$

dove $* = PR, HS$, ovvero prendere di tanto in tanto $\mathbf{d}^i = -\nabla f(\mathbf{x}^i)$.

Per quanto riguarda la convergenza, si rimanda alla parte di Analisi Numerica per l'analisi.

2.5.2 Heavy Ball

Un'alternativa ancora meno costosa rispetto al metodo del gradiente coniugato è il metodo Heavy Ball, in cui prendiamo β costante:

$$\mathbf{x}^{i+1} = \mathbf{x}^i - \alpha_i \nabla f(\mathbf{x}^i) + \beta(\mathbf{x}^i - \mathbf{x}^{i-1}).$$

- $\beta(\mathbf{x}^i - \mathbf{x}^{i-1})$ è un termine che di dà il momento della discesa: grazie a questo termine $\mathbf{x}^{i+1}$ si mantiene circa nella stessa direzione;
- l'antigradiente è la forza che devia la traiettoria verso un punto di discesa.

Un momento β molto grande ci permette di avere meno zig-zag nella discesa e di ottenere perciò una convergenza migliore. Si può dimostrare che con la giusta scelta di α_i e β si ha convergenza lineare con tasso migliore di quello della discesa del gradiente classica: si può dimostrare che se f è L-smooth e τ-convessa,

$$\|\mathbf{x}^{i+1} - \mathbf{x}_*\| \leq r\|\mathbf{x}^i - \mathbf{x}_*\|$$

con $r = \dfrac{\sqrt{k}-1}{\sqrt{k}+1}$ e $k = \dfrac{L}{\tau}$. La convergenza è perciò almeno lineare.

2.6 Ottimizzazione di funzioni non differenziabili

In questa sezione vediamo come si possono ottimizzare funzioni continue ma non C^1. La motivazione di questa trattazione è per esempio la minimizza-

zione della regolarizzazione L1: la norma $\|\cdot\|_1$ non è differenziabile in punti $\mathbf{x}$ tale che esiste un indice i con $x_i = 0$.

2.6.1 Subgradiente e subdifferenziale

Per tutta la trattazione assumeremo la funzione f almeno convessa. Dunque anche se f non è differenziabile, esiste un unico minimo locale.

Definizione. Data una funzione $f : X \subseteq \mathbb{R}^n \to \mathbb{R}$ convessa, diciamo che un vettore $\mathbf{s}$ è un suo subgradiente in $\mathbf{x}$ se per ogni $\mathbf{z} \in X$

$$f(\mathbf{z}) \geq f(\mathbf{x}) + \mathbf{s} \cdot (\mathbf{z} - \mathbf{x})$$

L'utilità di questa definizione è che se 0 è un subgradiente di f in $\mathbf{x}$, allora $\mathbf{x}$ è chiaramente il minimo globale: per ogni $\mathbf{z}$, $f(\mathbf{z}) \geq f(\mathbf{x})$. Il problema è che 0 potrebbe essere solo uno dei tanti subgradienti in $\mathbf{x}$:

Definizione. Il *subdifferenziale* di f in $\mathbf{x}$ è l'insieme

$$\partial f(\mathbf{x}) = \{\mathbf{s} \,|\, \mathbf{s} \text{ è subgradiente di } f \text{ in } \mathbf{x} \}$$

In particolare, se f è differenziabile in $\mathbf{x}$, allora $\partial f(\mathbf{x}) = \{\nabla f(\mathbf{x})\}$, cioè l'unico subgradiente è il gradiente stesso. Notiamo che per ogni direzione $\mathbf{d}$ e per ogni $\mathbf{s} \in \partial f(\mathbf{x})$,

$$\frac{\partial f(\mathbf{x})}{\partial \mathbf{d}} \geq \mathbf{s} \cdot \mathbf{d}$$

Ciò implica che $\mathbf{d}$ è una direzione di decrescita se e solo se $\mathbf{s} \cdot \mathbf{d} \leq 0$ per ogni subgradiente $\mathbf{s}$, quindi la direzione

$$\mathbf{s}^* = -\arg\min_{\mathbf{s} \in \partial f(\mathbf{x})} \|\mathbf{s}\|$$

è la direzione di massima decrescita.

Notiamo inoltre che $\mathbf{x}$ è minimo locale se e solo se $0 \in \partial f(\mathbf{x})$.

2.6.2 Metodi del subgradiente

Chiamiamo metodo del subgradiente un metodo del tipo $\mathbf{x}^{i+1} = \mathbf{x}^i + \alpha_i \mathbf{d}^i$, con $\mathbf{d}^i \in -\partial f(\mathbf{x}^i)$. A differenza del metodo del gradiente, ci sono più direzioni possibili se la funzione non è differenziabile in $\mathbf{x}^i$. In particolare quindi se

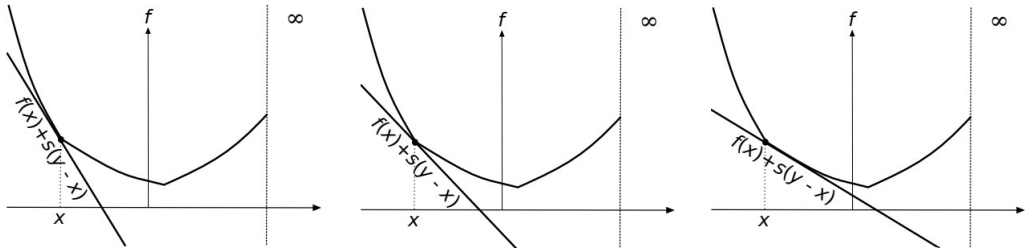

Figura 2.5: Subgradienti in $\mathbf{x}$: f non è differenziabile in $\mathbf{x}$, perciò abbiamo tanti subgradienti che corrispondono ad altrettanti modelli di f attorno ad $\mathbf{x}$

$f(\mathbf{x}^i)$ è vicino ad un minimo locale f_*, non è detto che la direzione $\mathbf{d}^i$ sia vicina a 0, perciò la condizione $\|\mathbf{d}^i\| < \varepsilon$ non è una condizione di stop efficace come nel caso del metodo del gradiente.

Non possiamo poi usare un passo di grandezza fissa $\alpha_i = \alpha$: prendiamo per esempio come funzione da ottimizzare $f(x) = L|x|$. Scelto un qualsiasi passo $\alpha > 0$, se partiamo da $x^0 = -\frac{-\alpha L}{2}$, essendo $f'(x^0) = -L$,

$$x^1 = x^0 + \alpha L = -\frac{\alpha L}{2} + \alpha L = \frac{\alpha L}{2}.$$

Continuando, $f'(x^1) = L$, dunque

$$x^2 = x^1 - \alpha L = -\frac{\alpha L}{2} = x^0.$$

L'algoritmo va dunque in loop. Osserviamo che entrambi questi problemi derivano appunto dal fatto che il subgradiente, anche se siamo molto vicini al minimo, potrebbe potrebbe non essere piccolo: nell'esempio precedente il gradiente è $\pm L$ in ogni punto diverso da 0.

DSS

Una soluzione che funziona è prendere α_i corti ma non troppo, cioè tali che

$$\sum_{i=1}^{\infty} \alpha_i = \infty,$$

ma

$$\sum_{i=1}^{\infty} \alpha_i^2 < \infty.$$

Questo tipo di passi si dice *Diminishing–Square Summable (DSS)*. Un esempio è $\alpha_i = \frac{1}{i}$. Questa scelta funziona perché se denotiamo con $\mathbf{g}^i$ il subgradiente scelto al passo i, possiamo notare che

$$\begin{aligned}\|\mathbf{x}^{i+1} - \mathbf{x}_*\|^2 &= \|\mathbf{x}^i - \alpha_i \mathbf{g}^i - \mathbf{x}_*\|^2 \\ &= \|\mathbf{x}^i - \mathbf{x}_*\|^2 + 2\alpha^i \mathbf{g}^i \cdot (\mathbf{x}_* - \mathbf{x}^i) + \alpha_i^2 \|\mathbf{g}^i\|^2 \\ &\leq \|\mathbf{x}^i - \mathbf{x}_*\|^2 + 2\alpha^i (f(\mathbf{x}_*) - f(\mathbf{x}^i)) + \alpha_i^2 \|\mathbf{g}^i\|^2,\end{aligned}$$

dove la disuguaglianza segue dal fatto che $\mathbf{g}^i$ è un subgradiente. Assumiamo ora inoltre che esista un L per cui $\|g^i\| \leq L$,

$$\|\mathbf{x}^{i+1} - \mathbf{x}_*\|^2 \leq \|\mathbf{x}^i - \mathbf{x}_*\|^2 + 2\alpha^i (f(\mathbf{x}_*) - f(\mathbf{x}^i)) + \alpha_i^2 L^2$$

e per induzione otteniamo che

$$\|\mathbf{x}^{i+1} - \mathbf{x}_*\| \leq \|\mathbf{x}^1 - \mathbf{x}_*\|^2 + \sum_{k=1}^{i} 2\alpha_k (f_* - f(\mathbf{x}^k)) + \sum_{k=i}^{i} \alpha_k^2 L^2. \tag{2.26}$$

Sia $\bar{x}$ un punto limite di $\mathbf{x}^i$ (la successione è limitata poichè in (2.26) il secondo termine è negativo e il terzo è $< +\infty$ per (DSS), quindi esiste un punto limite). Supponiamo allora per assurdo che $\mathbf{x}^i \nrightarrow \mathbf{x}_*$, cioè esiste un $\delta > 0$ per cui $f(x^i) - f_* \geq \delta$. Prendendo il limite per $i \to \infty$ nella disequazione (2.26), otteniamo

$$\|\bar{\mathbf{x}} - \mathbf{x}_*\| \leq \|\mathbf{x}^1 - \mathbf{x}_*\|^2 - \delta \sum_{k=1}^{\infty} 2\alpha_k + \sum_{k=i}^{\infty} \alpha_k^2 L^2 = -\infty,$$

che è assurdo.

Dal punto di vista pratico però la velocità di convergenza del metodo è molto lenta, perciò ci serve un metodo più veloce per scegliere i passi.

Passi di Polyak

Abbiamo precedentemente dimostrato che $\|\mathbf{x}^{i+1} - \mathbf{x}_*\|^2 \leq \|\mathbf{x}^i - \mathbf{x}_*\|^2 + 2\alpha^i (f(\mathbf{x}_*) - f(\mathbf{x}^i)) + \alpha_i^2 \|\mathbf{g}^i\|^2$. Notiamo allora che

$$\min 2\alpha^i (f(\mathbf{x}_*) - f(\mathbf{x}^i)) + \alpha_i^2 \|\mathbf{g}^i\|^2 = \frac{f(\mathbf{x}^i) - f_*}{\|\mathbf{g}^i\|^2}$$

Un passo della forma $\alpha^i = \beta^i \frac{(f(\mathbf{x}^i)-f_*)}{\|\mathbf{g}^i\|^2}$ è detto di Polyak. Abbiamo appena visto che $\beta^i = 1$ è ottimale, ma anche $\beta^i \in (0, 2)$ ci garantisce che $\|x^{i+1} - x_*\|^2 \leq \|x^i - x_*\|^2$.

Con questa scelta possiamo dimostrare che la velocità di convergenza è $O(1/\varepsilon^2)$, perciò il metodo è comunque molto lento: ad esempio per passare da $\varepsilon = 10^{-3}$ a $\varepsilon = 10^{-4}$ servono circa 100 volte il numero di iterazioni usate per arrivare a 10^{-3}. In ogni caso la sceta del passo è ottimale se conoscessimo f_*, che non conosciamo. Possiamo provare a stimarlo con il target level stepsize.

Target Level Stepsize

TODO

Deflected Subgradient

Se vogliamo una velocità di convergenza migliore possiamo provare a cambiare la direzione di decrescita, deviando il gradiente. Prendiamo $\mathbf{x}^{i+1} = \mathbf{x}^i - \alpha^i \mathbf{d}^i$ dove

$$\mathbf{d}^i = \gamma^i \mathbf{g}^i + (1 - \gamma^i)\mathbf{d}^{i-1}$$

con $0 < \gamma^i < 1$ e $\mathbf{g}^i$ un subgradiente in $\mathbf{x}^i$. Se vogliamo qualche risultato di convergenza teorica, dobbiamo usare qualche accorgimento.

2.6.3 Smoothed Gradient

Supponiamo che f sia della forma $f(x) = \max\{\mathbf{x}^T A\mathbf{z} : \mathbf{z} \in Z\}$, cioè per ogni $\mathbf{x}$, $f(\mathbf{x})$ è un problema di massimizzazione con funzione obbiettivo lineare. In generale una funzione del genere è continua ma non differenziabile: si può dimostrare che se $\bar{\mathbf{z}}$ è ottimale per $\mathbf{x}$ (nel problema di massimizzazione), allora $\bar{\mathbf{z}} \in \partial f(\mathbf{x})$, dunque se per un qualche $\mathbf{x}$ esiste più di un z ottimale, $|\partial f(\mathbf{x})| > 1$, e quindi f non è differenziabile in $\mathbf{x}$. Prendiamo allora una modificazione di f "smoothed", cioè

$$f_\mu(\mathbf{x}) = \max\{x^T Az + -\mu\|\mathbf{z}\|^2 : \mathbf{z} \in Z\}. \tag{2.27}$$

f_μ così definito è differenziabile, e possiamo perciò applicare uno dei metodi che abbiamo già descritto per funzioni differenziabili. Scegliendo μ piccolo ($\mu = O(\varepsilon)$), la velocità di convergenza risulta essere $O(1/\varepsilon)$, che è molto migliore di quella del metodo subgradiente, ma comunque piuttosto lenta.

2.6.4 Bundle Methods

L'idea fondamentale dei bundle methods è spostarsi solo quando è davvero conveniente, e altrimenti raccogliere informazioni per provare una nuova direzione. Se f è convessa, abbiamo la garanzia che ogni modello costruito a partire da un qualsiasi subgradiente stia sotto la funzione. L'idea è quindi di raccogliere più modelli possibili e utilizzare il massimo tra tutti per "tagliare lo spazio di ricerca" (da cui l'altro nome del metodo, cutting plane algorithm).

Supponiamo dunque di essere nel punto $\mathbf{x}^i$, e sia $f^i = f(\mathbf{x}^i)$. Il *bundle* in $\mathbf{x}^i$ è

$$\mathcal{B} = \{(\mathbf{x}^i, f^i, \mathbf{g}^i) : \mathbf{g}^i \in \partial f(\mathbf{x}^i)\}$$

Il modello che consideriamo sarà quindi

$$f_{\mathcal{B}}(\mathbf{x}) = \max\{f^i + \langle \mathbf{g}^i, \mathbf{x} - \mathbf{x}^i \rangle : (\mathbf{x}^i, f^i, \mathbf{g}^i) \in \mathcal{B}\},$$

cioè il modello migliore possibile con le informazioni disponibili. Ad ogni passo calcoliamo $x^*_{\mathcal{B}} = \min f_{\mathcal{B}}$. Così com'è, il metodo non funziona bene, poiché non ci sono garanzie che $\mathbf{x}^{i+1}$ sia vicino a $\mathbf{x}^i$, e anzi $f_{\mathcal{B}}$ potrebbe anche essere illimitata inferiormente. La soluzione è stabilizzare il problema con un termine quadratico, cioè risolvere

$$\min\{f_{\mathcal{B}} + \mu\|\mathbf{x} - \bar{\mathbf{x}}\|\},$$

dove $\bar{\mathbf{x}}$ è la minore soluzione precedentemente trovata, e$\mu > 0$ è un parametro che va stimato di volta in volta, e può cambiare da un'iterazione all'altra. Ci muoviamo solo se il nuovo $\mathbf{x}^*_{\mathcal{B}}$ è tale che $f(\mathbf{x}_{\mathcal{B}}) << f(\bar{\mathbf{x}})$, dove $<<$ indica una regola simile a quella di Armijo, se consideriamo che il modello invece che dal gradiente è dato da $f_{\mathcal{B}}$ (Algoritmo 4). In questo caso possiamo possibilmente aumentare μ (passi più lunghi) , in caso contrario μ va diminuito (passi più corti)

Algoritmo 4 Proximal Bundle Method

procedure $x = PBM\,(\,f\,,\,x\,,\,m_1\,,\,\varepsilon\,,\,\mu\,)$
  $\mathscr{B} \leftarrow \{\,(\,x\,,\,f(\,x\,)\,,\,g \in \partial f(\,x\,)\,)\,\}$;
  while (**true**) **do**
    $d^* \leftarrow \text{argmin}\,\{\,f_{\mathscr{B}}(\,x + d\,) + \mu \| \,d\,\|^2 / 2\,\}$;
    if($\mu \| \,d^*\,\| \leq \varepsilon$) **then break**;
    if($f(\,x + d^*\,) - f(\,x\,) \leq m_1[\,f_{\mathscr{B}}(\,x + d^*\,) - f(\,x\,)\,\,]$)
      then { $x \leftarrow x + d^*$; possibly $\mu \searrow$; } **else** possibly $\mu \nearrow$;
    $\mathscr{B} \leftarrow \mathscr{B} \cup \{\,(\,x + d^*\,,\,f(\,x + d^*\,)\,,\,g \in \partial f(\,x + d^*\,)\,)\,\}$;

Capitolo 3

Ottimizzazione vincolata

3.1 Introduzione all'ottimizzazione vincolata

Torniamo al problema generale descritto nell'introduzione, cioè risolvere

$$(P) \qquad \min f(x) \quad x \in X \subseteq \mathbb{R}^n \tag{3.1}$$

Dove X è la regione ammissibile per le soluzioni. Se $x \in X$ diciamo che è una soluzione ammissibile, altrimenti diciamo che x non è ammissibile. x_* che soddisfa il problema di minimo viene spesso detto *ottimo*, e indicheremo il valore ottimo $f_* = f(x_*)$ con $v(P)$. Notiamo che in teoria potremmo nascondere i vincoli nella funzione obbiettivo, e passare così ad un problema non vincolato, in questo modo:

$$(P) \qquad \min f(x) + \mathbb{1}_X(x) \tag{3.2}$$

con

$$\mathbb{1}_X(x) = \begin{cases} 0 & x \in X \\ +\infty & x \notin X \end{cases}.$$

Questa idea non è quasi mai buona, perchè la funzione indicatrice $\mathbb{1}_X$ è una funzione difficile da ottimizzare, in quanto non continua.

Ci sono alcuni problemi che possono nascere quando imponiamo dei vincoli sulle soluzioni:

- Se $X = \emptyset$, $v(P) = +\infty = \inf\{\emptyset\}$. In questo caso non possiamo risolvere (P) trovando una soluzione (perché non ce ne sono), ma dobbiamo dimostrare che X è vuoto.

- Anche se X non è vuoto, il minimo potrebbe comunque non appartenere a X: prendiamo per esempio $\min x$ con $x \in (0, 1)$. Chiaramente vorremo prendere come soluzione $x = 0$, che tuttavia non appartiene alla regione ammissibile.

In ogni caso questi problemi non sono molto importanti nel Machine learning, perciò non daremo loro molto peso nella trattazione.

Come nel caso non vincolato, l'ottimizzazione globale è difficile, perciò ci restringiamo a trovare un minimo locale cioè $x^* = \arg\min\{f(x) : x \in B(x^*, \varepsilon) \cap X\}$ per qualche $\epsilon > 0$.

3.1.1 Cenni di topologia

Definizione. Diciamo che un insieme $X \subseteq \mathbb{R}^n$ è *aperto* se per ogni punto $x \in X$ esiste un $\varepsilon > 0$ tale che la palla $B(x, \varepsilon)$ è contenuta in X Diciamo che X è *chiuso* se il suo complementare $Y = \mathbb{R}^n \setminus X$ è aperto.

Una caratterizzazione equivalente delle due definizioni è la seguente: Chiamiamo $\mathring{X}$ la parte interna di X, ovvero tutti i punti $x \in X$ per cui vale che esiste un $\varepsilon > 0$ tale che la palla $B(x, \varepsilon)$ è contenuta in X. È evidente che X è aperto se e solo se $X = \mathring{X}$.

Chiamiamo poi ∂X la frontiera di X ovvero i punti $x \in \mathbb{R}^n$ per cui vale che per ogni $\epsilon > 0$ esistono $y_1, y_2 \in B(x, \epsilon)$ tali che $y_1 \in X$ e $y_2 \notin X$. La chiusura di X, denotata con $\bar{X}$, è $\mathring{X} \cup \partial X$ Si può dimostrare facilmente che X è chiuso se e solo se $X = \bar{X}$.

Esempio. Alcuni esempi di insiemi aperti e chiusi sono:

- Dati $a < b \in \mathbb{R}$, l'intervallo (a, b) è aperto, mentre l'intervallo $[a, b]$ è chiuso.
- Gli intervalli $[a, b)$ e $(a, b]$ non sono né aperti né chiusi.
- La frontiera di tutti questi insiemi è $\{a, b\}$, mentre la parte interna è (a, b).
- La palla aperta $B(x, \varepsilon) = \{y \in \mathbb{R}^n : \|x - y\| < \varepsilon\}$ è chiaramente aperta, mentre la palla chiusa $\bar{B}(x, \epsilon) = \{y \in \mathbb{R}^n : \|x - y\| \leq \varepsilon\}$ è chiusa.

Un'altra caratterizzazione utile degli insiemi chiusi è la seguente:

X è chiuso se e solo se, per ogni successione $(x_i)_{i=1}^{\infty}$ di punti appartenenti a X, se $x_i \to x$, allora $x \in X$. Nel seguito assumeremo sempre X chiuso. Così facendo, ci assicuriamo che se $x \in \partial X$, allora x è ammissibile. Nella prossima sezione infatti daremo delle condizioni per cui possiamo assicurarci che $x \in \partial X$ sia ottimale.

3.2 Condizioni di ottimalità

3.2.1 Cono Tangente

Se $x_* \in \mathring{X}$ è un ottimo locale, allora è ovvio che $\nabla f(x_*) = 0$. Tuttavia così non è se $x \in \partial X$: un esempio banale è $\min\{x : x \in [0,1]\}$: in questo caso 0 è un minimo, ma $f'(x) = (x)' = 1$ per ogni x, dunque $f'(0) \neq 0$. Ci servono perciò delle diverse condizioni di ottimalità per punti appartenenti alla frontiera. Introduciamo per questo motivo il cono tangente a X in x:

Definizione. Il cono tangente è l'insieme

$$T_X(x) = \left\{ d : \exists\, (z_i) \subseteq X, z_i \to x, \text{ e } (t_i) > 0, t_i \to 0 \text{ t.c. } \lim_{i \to +\infty} \frac{z_i - x}{t_i} = d \right\}$$

Definizione. Un insieme $\mathcal{C}$ è un *cono* se vale $x \in \mathcal{C} \Rightarrow tx \in \mathcal{C}$ per ogni $t \geq 0$.

Osservazione 7. Si verifica facilmente che $T_X(x)$ è effettivamente un cono.

Proposizione 3.2.1. *Se $x \in X$ è un minimo locale, vale*

$$(TCC) \qquad \nabla f(x) \cdot d \geq 0 \; \forall\, d \in T_X(x)$$

Dimostrazione. Supponiamo per assurdo che x sia un ottimo locale, ma esista $d \in T_X(x)$ tale che $\nabla f(x) \cdot d < 0$. $T_X(x)$ è un cono e $d \neq 0$, quindi possiamo assumere che $\|d\| = 1$. Dalla definizione di $d \in T_X(x)$ si vede che esiste una successione (z_i) e una successione $(t_i) > 0$ tale per cui

$$\lim_{i \to \infty} \frac{\|z_i - x\|}{t_i} = 1,$$

cioè $\|z_i - x\|$ e t_i convergono a 0 con la stessa velocità. Per la formula di Taylor al primo ordine,

$$f(z_i) - f(x) = \nabla f(x) \cdot (z_i - x) + o(\|z_i - x\|)$$

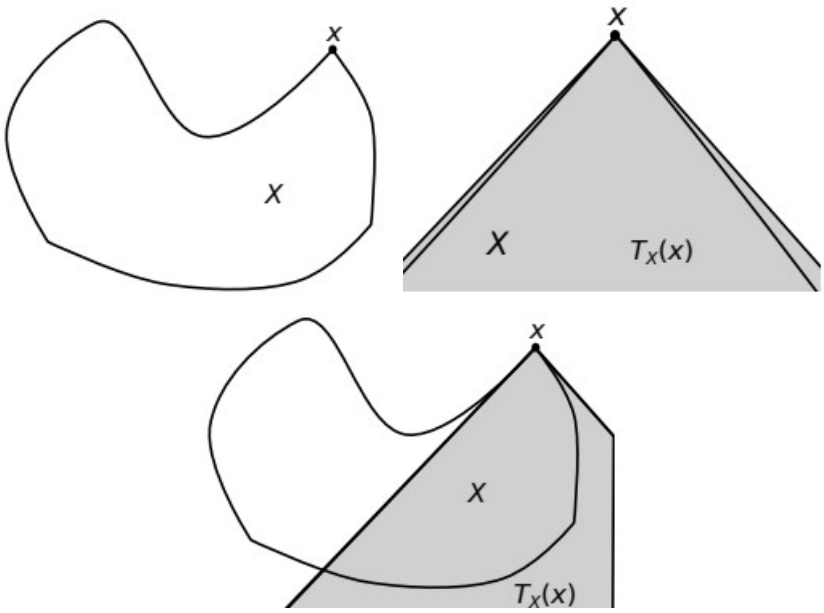

Figura 3.1: Se zoomiamo la regione X in x, localmente X assomiglia ad un cono. Tale cono approssima X molto vicino a x. Nella terza figura vediamo $T_X(x)$ e X nella scala originaria.

Per ipotesi $\nabla f(x) \cdot d < 0$, dunque poiché $t_i > 0$ e $\frac{z_i - x}{t_i} \to d$, da un certo i in poi $\nabla f(x_i) \cdot (z_i - x) < 0$. Inoltre

$$\lim_i \frac{\nabla f(x) \cdot (z_i - x)}{t_i} = \nabla f(x) \cdot d = costante,$$

Perciò $\nabla f(x) \cdot (z_i - x)$ va a 0 velocemente quanto t_i, e quindi quanto $\|z_i - x\|$. Dunque da un certo i in poi, $f(z_i) - f(x) < 0$. Poiché $z_i \to x$ si ottiene che per ogni ϵ esiste $z_i \in B(x, \epsilon)$ con $f(z_i) < f(x)$, contraddicendo che x sia un minimo locale. □

3.2.2 Insiemi convessi

Abbiamo appena dimostrato che x minimo locale $\Rightarrow$ (TCC). L'implicazione inversa non vale in generale, tuttavia vale se $T_X(x) + x \supset X$, cioè se l'insieme è contenuto nel cono tangente al minimo (traslato in modo che abbia il vertice nel minimo), è possibile dimostrare. La condizione $T_X(x) + x \supset X$ vale sempre se X è convesso:

Definizione. Diciamo che un insieme C è *convesso* se, per ogni $x, z \in X$, il segmento tra x e z, $conv(x,z) = \{y : y = tx + (1-t)z, t \in [0,1]\}$, è tutto contenuto in C.

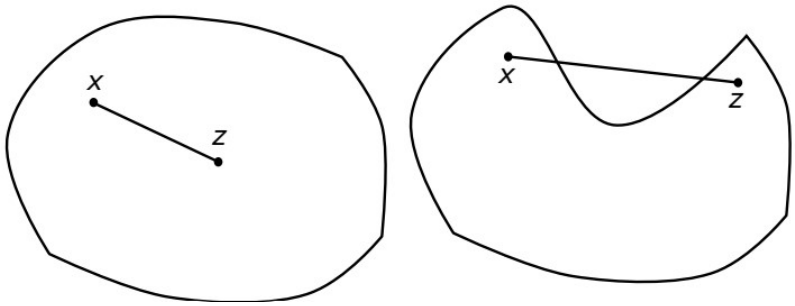

Figura 3.2: A sinistra un insieme convesso, a destra un insieme non convesso.

Dato un qualsiasi insieme $X \subseteq \mathbb{R}^n$, possiamo definire l'inviluppo convesso di X come

$$conv(X) = \bigcap \{C : C \text{ è convesso, } X \subseteq C\} = \bigcup_{x,y \in X} conv(x,y)$$

cioè il più piccolo insieme convesso contenente X o, equivalentemente, l'unione di tutti i segmenti che hanno entrambi gli estremi in X.

Enunciamo alcune proprietà degli insiemi convessi che ci saranno utili in seguito:

Proposizione 3.2.2. *Se $f : \mathbb{R}^n \to \mathbb{R}$ è una funzione, f è convessa se e solo se il suo epigrafico $epi(f) = \{(x,z) \in \mathbb{R}^{n+1} : z \geq f(x)\}$ è convesso.*

Proposizione 3.2.3. *Se $f : \mathbb{R}^n \to \mathbb{R}$ è convessa, i suoi insiemi di sottolivello $S(f,z) = \{x \in \mathbb{R}^n : f(x) \leq z\}$ sono convessi per ogni z. Il viceversa non è vero. Vale infatti: $S(f,z)$ convesso $\Leftrightarrow$ f quasi-convessa.*

Un buon numero di insiemi che già abbiamo visto sono convessi:

- l'inviluppo convesso di un numero finito di punti, detto anche politopo:

$$conv(\{x_1, \ldots, x_k\}) = \{x = \sum_{i=1}^{k} \alpha_i x_i : \sum \alpha_i = 1, \alpha_i \geq 0\}.$$

- gli iperpiani affini $\mathcal{H} = \{x \in \mathbb{R}^n : ax = b\}$, che sono insiemi di livello di funzioni lineari affini.
- i semispazi affini $\mathcal{S} = \{x \in \mathbb{R}^n : ax \leq b\}$, che sono insiemi di sottolivello di funzioni lineari affini.
- Gli elissoidi $\mathcal{E}(Q, x, r) = \{z \in \mathbb{R}^n : (z-x)^T Q(z-x) \leq r$, che sono insiemi di sottolivello di funzioni quadratiche.
- Le palle in norma p per ogni $p \geq 1$.

Come abbiamo già accennato, se X è convesso allora se in x vale (TCC), ciò è sufficiente perché x sia ottimale.

Definizione. Il cono delle direzioni ammissibili in x è

$$F_X(x) = \{d : \exists\, \bar{\varepsilon} > 0 \text{ t.c. } x + \bar{\varepsilon} d \in X\}.$$

Dimostrare che $F_X(x)$ è un cono è semplice: $d \in F_X(x)$ se e solo se $x + \varepsilon d \in X$, se e solo se $x + td\varepsilon/t \in X$, se e solo se $td \in F_X(x)$. Con questa nuova definizione possiamo dimostrare

Proposizione 3.2.4. *Se f è una funzione convessa e X è un insieme convesso, allora x_* soddisfa (TCC) $\iff$ x_* è un minimo globale*

Dimostrazione. Abbiamo già dimostrato ($\Leftarrow$) per insiemi qualsiasi. Dimostriamo ($\Rightarrow$).

- Si dimostra facilmente che dato un insieme X qualsiasi (anche non convesso), $X \subseteq F_X(x) + x$ per ogni x. Infatti dato $y \in X$, $(y - x) \in F_X(x)$: $x + (y - x) = y \in X$ (definizione di $F_X(x)$ con $\varepsilon = 1$). Dunque $y = (y - x) + x \in F_X(x) + x$.
- Inoltre se X è convesso, $F_X(x) \subseteq T_X(x)$ (in particolare T_X è la chiusura di F_X): sia $d \in F_X(x)$ Allora esiste un ϵ tale per cui $x + \varepsilon d \in X$. Data una qualunque successione $(t_i)_i$ con $t_i < 1$ per ogni i, e $t_i \to 0$, basta prendere $z_i = x + \epsilon t_i d$.

  Tutti gli z_i stanno nel segmento tra x e $x + \varepsilon d$, perciò, per la convessità dell'insieme, appartengono a X.

  Allora $\lim_{i\to\infty} \dfrac{\varepsilon t_i d}{t_i} = \varepsilon d \in T_X(x)$. Poiché $T_X(x)$ è un cono, anche $d \in T_X(x)$. Dunque $T_X(x) \supseteq F_X(x)$.

- $T_X(x)+x \supseteq F_X(x) \supseteq X$, e quindi (TCC) implica che nessuna direzione ammissibile sia anche di decrescita, dunque se x_* soddisfa (TCC), è anche un minimo locale. Inoltre, se f è convessa, ogni minimo locale è un minimo globale in X. Abbiamo dimostrato perciò la nostra tesi.

□

Osserviamo che se $x \in \mathring{X}$, $T_X(x) = F_X(x) = \mathbb{R}^n$, perciò la (TCC) si traduce in $\nabla f(x) \cdot d \geq 0$ per ogni d, e quindi $\nabla f(x) = 0$. Perciò in generale possiamo vedere (TCC) come una generalizzazione del fatto che x sia un punto stazionario: è solo necessaria e non sufficiente nel caso ,generale, ma se (P) è convesso siamo fortunati, perché esiste un unico punto stazionario, ed è un minimo globale.

Non è ovvio come assicurarci che un punto x soddisfi (TCC): in principio, dovremmo fare infiniti controlli, uno per ogni direzione. Nella prossima sezione perciò vogliamo trasformare questa condizione in un'altra più utilizzabile in pratica.

3.2.3 Lemma di Farkas

Fino ad ora abbiamo assunto X come un generico sottoinsieme (chiuso) di $\mathbb{R}^n$. Nella pratica però ci serve una descrizione algebrica di X: in particolare lo pensiamo come descritto da un insieme di disequazioni e di equazioni. Una forma standard che useremo è la seguente, dati due insiemi di indici $\mathcal{I}$ e $\mathcal{J}$, e delle funzioni g_i per ogni $i \in \mathcal{I}$ e h_j per ogni $j \in \mathcal{J}$:

$$X = \{x \in \mathbb{R}^n : g_i(x) \leq 0 \, \forall i \in \mathcal{I}, \, h_j(x) = 0 \, \forall j \in \mathcal{J}\} \tag{3.3}$$

equivalentemente possiamo scrivere X con vincoli in forma vettoriale: dette $G(x) = [g_i(x)]_{i \in \mathcal{I}} : \mathbb{R}^n \to \mathbb{R}^{|\mathcal{I}|}$, $H(x) = [h_j(x)]_{j \in \mathcal{J}} : \mathbb{R}^n \to \mathbb{R}^{|\mathcal{J}|}$,

$$X = \{x : G(x) \leq 0, H(x) = 0\} \tag{3.4}$$

Osserviamo che, almeno dal punto di vista della trattazione teorica (dal punto di vista algoritmico spesso non conviene) possiamo assumere che non ci siano vincoli di uguaglianza: infatti $h_j = 0$ corrisponde a $h_j \leq 0, -h_j \leq 0$. Notiamo che X definito come in 3.3 è convesso se e solo se le g_i sono convesse e sia h_j che $-h_j$ sono convesse, cioè le h_j sono lineari. Molto spesso ci restringeremo a questo caso o ad altri ancora più semplici.

Esempio. Cominciamo dal caso più semplice di tutti, quello di vincoli di uguaglianza lineari:

$$(P) \qquad \min\{f(x) : Ax = b\},$$

dove $A \in \mathbb{R}^{m\times n}$ (m è il numero di vincoli, n la dimensione del problema). In questo caso notiamo che le direzioni ammissibili e quelle del cono tangente coincidono, e corrispondono alle d tali per cui $Ad = 0$: infatti se $Ax = b$, $A(x + d) = Ax + Ad = Ax = b$. In particolare se $d \in F_X$, anche $-d \in F_X$.

Quindi la condizione (TCC) diventa $\nabla f(x) \cdot d = 0$ per ogni $d \in F_X(x)$. Poiché $F_X(x) = \ker A$, si ha che $\nabla f(x) \in (\ker A)^\perp = \mathrm{im}(A^T)$, cioè esiste un $\mu \in \mathbb{R}^m$ tale per cui $\nabla f(x) = \mu A$. Le condizioni di ottimalità diventano perciò:

$$Ax = b, \text{ e esiste } \mu \in \mathbb{R}^m \text{ t.c. } \nabla f(x) = \mu^T A \tag{3.5}$$

μ è il primo esempio di *variabile duale*: siamo passati da un "per ogni" nel (TCC) originale, ad un "esiste": per verificare che x sia ottimale basta trovare μ.

Passiamo ora al caso di generiche equazioni non lineari $g_i(x) \leq 0$, $i \in \mathcal{I}$. Chiamiamo *vincoli attivi* di $x \in X$ i vincoli per cui $g_i(x) = 0$, cioè

$$\mathcal{A}(x) = \{i \in \mathcal{I} : g_i(x) = 0\}$$

Questi vincoli sono gli unici che davvero ci interessano quando ci troviamo in x: il cono tangente è un concetto locale, cioè $T_X(x) = T_{X\cap B(x,r)}(x)$, e se, per qualche i, $g_i(x) < 0$, tale vincolo non ha impatto su $T_X(x)$. Si veda la Figura 3.1: il cono dipende solo dai vincoli che definiscono il punto angoloso in cui si trova x. Chiamiamo inoltre cono delle direzioni ammissibili al primo ordine

$$D_X(x) = \{d : \nabla g_i(x) \cdot d \leq 0 \; i \in \mathcal{A}(x), \nabla h_j(x) \cdot d = 0 \; j \in \mathcal{J}\}$$

Si può dimostrare che $D_X \supseteq T_X$ per ogni x. In generale non vale l'uguaglianza, ma in molti casi sì: si parla di *qualificazione dei vincoli*, cioè di trovare condizioni per cui vale $D_X = T_X$. Alcuni esempi tipici sono:

Vincoli affini: se le g_i e le h_j sono affini, $D_X(x) = T_X(x)$ per ogni $x \in X$

Slater: se le g_i sono convesse, le h_j sono affini e vale che esiste $\bar{x}$ tale che $g_i(\bar{x}) < 0$, allora $D_X(x) = T_X(x)$ per ogni $x \in X$

Lineare indipendenza dei gradienti: se per un $\bar{x} \in X$ vale che i vettori $\{(\nabla g_i(\bar{x}))_{i\in\mathcal{A}(\bar{x}}, (\nabla h_j(\bar{x}))_{j\in\mathcal{J}}\}$ sono linearmente indipendenti, allora $D_X(\bar{x}) = T_X(\bar{x})$

Nel caso in cui valga $D_X = T_X$ possiamo quindi controllare (TCC) usando la definizione di D_X. Per fare ciò useremo il lemma di Farkas. Prima di enunciarlo diamo una definizione:

Definizione. $\mathcal{C}$ è un cono poliedrico se si può scrivere come

$$\mathcal{C} = \{d : Ad \leq 0\}$$

per qualche $A \in \mathbb{R}^{k\times n}$

Si vede facilmente che $D_X(x)$ è un cono poliedrico: basta prendere la matrice

$$A = \begin{pmatrix} (\nabla g_i(x))_{i\in\mathcal{A}(x)} \\ (\nabla h_j(x))_{j\in\mathcal{J}} \\ (-\nabla h_j(x))_{j\in\mathcal{J}} \end{pmatrix}. \tag{3.6}$$

Dato un qualunque cono poliedrico C possiamo definire il cono duale $\mathcal{C}^*$ come

$$\mathcal{C}^* = \{c = \sum_{i=1}^{k} \lambda_i A_i : \lambda \geq 0\}$$

Possiamo enunciare il lemma di Farkas:

Lemma. (di Farkas): *Per ogni $c \in \mathbb{R}^n$ vale sempre una e una sola delle seguenti condizioni:*

- *o esiste $\lambda \in \mathbb{R}^k$, $\lambda \geq 0$, tale che $c = \sum_{i=1}^{k} \lambda_i A_i$*
- *oppure esiste d tale che $Ad \leq 0$ e $c \cdot d > 0$*

In altre parole dato un cono poliedrico $\mathcal{C}$, o $c \in \mathcal{C}^*$, oppure il prodotto scalare con uno degli elementi di $\mathcal{C}$ è positivo, e le due condizioni sono mutualmente esclusive.

Nella figura 3.4 si dà un'idea della dimostrazione: possiamo definire il *cono polare*

$$\mathcal{C}^\circ = \{c : \forall\, d \in \mathcal{C}\ c \cdot d \leq 0\},$$

e vedere che $\mathcal{C}^\circ = \mathcal{C}^*$. Il lemma di Farkas dice quindi " $c \in \mathcal{C}^*$ oppure $c \notin \mathcal{C}^*$"

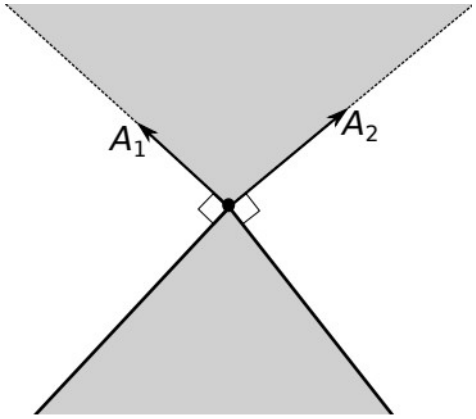

Figura 3.3: Il cono $\mathcal{C}$ (in basso) e il suo duale $\mathcal{C}^*$ (in alto). $\mathcal{C}^*$ è la combinazione lineare con coefficienti non negativi di A_1 e A_2, che sono i vettori che definiscono C

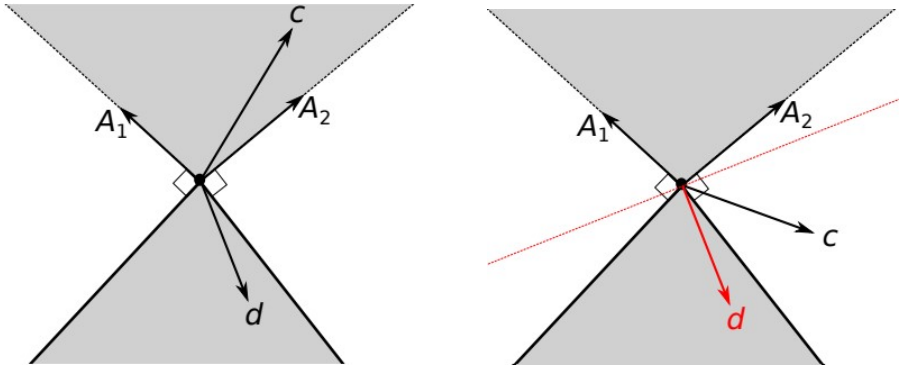

Figura 3.4: Il cono $\mathcal{C}^*$ è composto da tutte e sole le direzioni che formano un angolo maggiore di 90 gradi con ogni vettore del cono $\mathcal{C}$; a sinistra siamo nel primo caso dell'enunciato del Lemma di Farkas, cioè $c \in \mathcal{C}^*$; a destra nel secondo caso, cioè $c \notin \mathcal{C}^*$ e $c \cdot d > 0$.

3.2.4 Condizioni di Karush–Kuhn–Tucker

Dal Lemma di Farkas è facile arrivare a delle condizioni che ci permettano di verificare l'ottimalità. Supponiamo che (P) sia convesso e che $D_X(x) = T_X(x)$. In queste ipotesi per verificare l'ottimalità globale di x, basta verificare (TCC) su D_X, ovvero

$$\nabla f(x) \cdot d \geq 0 \text{ per ogni } d \text{ t.c. } \nabla g_i(x) \cdot d \leq 0 \; i \in \mathcal{A}(x), \; \nabla h_j(x) \cdot d = 0 \; j \in \mathcal{J} \tag{3.7}$$

Per verificare (3.7) usiamo il Lemma di Farkas con $\mathcal{C} = D_X$, A come definita in (3.6) e $c = -\nabla f(x)$: per il lemma, (3.7) vale se e solo se esistono $\lambda \in \mathbb{R}_+^{|\mathcal{A}(x)|}$ e $\mu', \mu'' \in \mathbb{R}_+^{|\mathcal{J}|}$ tali che

$$\nabla f(x) + \sum_{i \in \mathcal{A}(x)} \lambda_i \nabla g_i(x) + \sum_{j \in \mathcal{J}} \mu'_j \nabla h_j(x) \sum_{j \in \mathcal{J}} -\mu''_j \nabla h_j(x) = 0$$

chiamando $\mu = \mu' - \mu''$, otteniamo che x è ottimale se esistono $\lambda \in \mathbb{R}_+^{|\mathcal{A}(x)|}$ e $\mu \in \mathbb{R}^{|\mathcal{J}|}$ (μ non necessariamente positivo) tali che

$$\nabla f(x) + \sum_{i \in \mathcal{A}(x)} \lambda_i \nabla g_i(x) + \sum_{j \in \mathcal{J}} \mu_j \nabla h_j(x) = 0 \tag{3.8}$$

Le equazioni di Karush–Kuhn–Tucker sono molto simili all'equazione (3.8) e sono ad essa equivalenti:

Condizioni di Karush-Kuhn-Tucker (KKT): Esistono $\lambda \in \mathbb{R}^{|\mathcal{I}|}$ e $\mu \in \mathbb{R}^{|\mathcal{J}|}$ tali che

$$g_i(x) \leq 0 \; i \in \mathcal{I}, \quad h_j(x) = 0 \; j \in \mathcal{J} \tag{3.9}$$

$$\nabla f(x) + \sum_{i \in \mathcal{I}} \lambda_i \nabla g_i(x) + \sum_{j \in \mathcal{J}} \mu_j \nabla h_j(x) = 0 \tag{3.10}$$

$$\sum_{i \in \mathcal{I}} \lambda_i \left(-g_i(x)\right) = 0 \tag{3.11}$$

Analizziamo le 3 equazioni:

- La (3.9) ci dice semplicemente che x è una soluzione ammissibile.
- La (3.10) corrisponde alla (3.8), ma con tutto $\mathcal{I}$ al posto dell'insieme dei vincoli attivi $\mathcal{A}(x)$. Perciò per rendere equivalenti le due equazioni ci serve la terza equazione.

- la (3.11), detta *complementary slackness*, ci garantisce che se $i \notin \mathcal{A}(x)$, allora $\lambda_i = 0$: infatti la sommatoria è composta di termini tutti maggiori o uguali a 0, perciò, se vogliamo che sia nulla, ogni termine dev'essere esattamente 0. Se $i \notin \mathcal{A}(x)$, allora $-g_i(x) > 0$, e dunque per forza $\lambda_i = 0$.

Osservazione 8. La riscrittura dell'equazione (3.8) nelle equazioni KKT è utile dal punto di vista pratico perché nella KKT il vettore λ ha sempre la stessa dimensione, mentre nella (3.8) dipende da quanti vincoli sono attivi e quindi da x.

Esattamente come (TCC), le (KKT) sono solo necessarie perché x sia ottimo, ma in generale non sufficienti (non riescono a distinguere massimi da minimi da punti di sella).

Tuttavia, se (P) è convesso, allora sono necessarie e sufficienti per l'ottimalità: se x_* soddisfa (KKT), $\nabla f(x) \cdot d \geq 0$ per ogni $d \in D_X(x_*)$, e $D_X \supseteq T_X \supseteq F_X$, quindi $\nabla f(x) \cdot d \geq 0$ per ogni $d \in F_X(x_*)$, cioè x_* è ottimo globale.

3.3 Dualità Lagrangiana

3.3.1 Funzione Lagrangiana e dualità

Riprendiamo la condizione (KKT):

$$\nabla f(x) + \sum_{i \in |\mathcal{A}(x)} \lambda_i \nabla g_i(x) + \sum_{j \in \mathcal{J}} \mu_j \nabla h_j(x) = 0$$

Notiamo che l'equazione è equivalente a $\nabla_x L(x, \lambda, \mu) = 0$, dove

$$L(x, \lambda, \mu) = f(x) + \sum_{i \in \mathcal{I}} \lambda_i g_i(x) + \sum_{j \in \mathcal{J}} \mu_j h_j(x) \tag{3.12}$$

e ∇_x indica il gradiente ottenuto derivando solo nella variabile x.

Definizione. $L(\cdot)$ è detta *funzione lagrangiana* del problema (P).

Cioè x_* è ottimo se è un punto stazionario di L per i giusti λ e μ fissati. Ci chiediamo perciò come trovare x_* conoscendo i giusti λ e μ: l'idea è di risolvere il *rilassamento lagrangiano*

$$(R_{\lambda,\mu}) \qquad \psi(\lambda, \mu) = \min_{x \in \mathbb{R}^n} L(x, \lambda, \mu)$$

In generale chiamiamo un rilassamento $(\underline{P})$ di (P) è un altro problema di ottimizzazione

$$(\underline{P}) \qquad \min\{\underline{f}(x), x \in \underline{X}\}$$

con

i) $\underline{f}(x) \leq f(x)$ per ogni $x \in X$

ii) $\underline{X} \supseteq X$.

In questo modo è ovvio che $v(\underline{P}) \leq v(P)$, cioè $(\underline{P})$ ci fornisce un lower bound al valore ottimo di (P).

Lemma 3.3.1. *Per ogni $\lambda \in \mathbb{R}_+^{|\mathcal{I}|}, \mu \in \mathbb{R}^{|\mathcal{J}|}$, $(R_{\lambda,\mu})$ è un rilassamento di (P).*

Dimostrazione. La regione ammissibile di $(R_{\lambda,\mu})$ è tutto $\mathbb{R}^n$, dunque vale ii).

Vediamo che vale i): se x è ammissibile per (P), $h_j(x) = 0$ per ogni $j \in \mathcal{J}$ e $g_i(x) \leq 0$ per ogni $i \in \mathcal{I}$. Inoltre $\lambda \geq 0$, perciò $\sum_j \mu_j h_j(x) = 0$ e $\sum_i \lambda_i g_i(x) \leq 0$, perciò $L(x, \lambda, \mu) \leq f(x)$ qualsiasi siano $\lambda \geq 0$ e μ. □

Il Lemma 3.3.1 ci porta al concetto di dualità debole:

Dualità debole: Per ogni $\lambda \geq 0, \mu$ e x vale $\psi(\lambda, \mu) \leq v(P) \leq f(x)$.

In altre parole, possiamo definire il *problema duale*

$$(D) \qquad \max \psi(\lambda, \mu) \quad \lambda \in \mathbb{R}_+^{|\mathcal{I}|}, \mu \in \mathbb{R}^{|\mathcal{J}|} \tag{3.13}$$

La dualità debole ci dice che vale sempre $v(D) \leq v(P)$.

(D) è spesso un problema non troppo difficile da risolvere, anche se (P) fosse difficile:

- Per calcolare $\psi(\lambda, \mu)$ bisogna risolvere $(R_{\lambda,\mu})$ che è un problema di ottimizzazione non vincolato.
- ψ è concava: è il minimo di (infinite) funzioni lineari in λ e μ. (la "non linearità" di L è solo in x). Trovare il massimo di una funzione concava è facile (equivale a trovare il minimo di una funzione convessa). Tuttavia in alcuni casi $\psi = -\infty$
- In generale $\psi \notin C^1$, anche se $f, g_i, h_j \in C^1$, ma vale che se $\bar{x}$ è una soluzione ottima di $R_{\lambda,\mu}$, allora il vettore $(G(\bar{x}), H(\bar{x})) \in \partial\psi(\lambda, \mu)$

- In particolare se $\bar{x}$ nel punto sopra è l'unica soluzione ottima di $(R_{\lambda,\mu})$, si dimostra che ψ è differenziabile in (λ, μ), e $\nabla\psi(\lambda, \mu) = (G(\bar{x}), H(\bar{x}))$

Queste proprietà ci portano a concludere che ψ è spesso facile da massimizzare, dunque abbiamo trasformato un problema (P) in un altro problema (D) quasi non vincolato (il vincolo $\lambda \geq 0$ è facile) che ci dà un valore $v(D) \leq v(P)$ Allo stesso tempo, per calcolare $\psi(\lambda, \mu)$ bisogna risolvere un problema di minimizzazione non necessariamente convesso, che può essere in alcuni casi difficile.

Idealmente vorremmo che valga la **dualità forte**, ovvero

$$v(D) = v(P)$$

In generale questa proprietà non vale, ma vale se (P) è convesso.

3.3.2 Duali speciali

Problemi lineari

Nel caso di problemi lineari

$$(P) \qquad \min c^T x \quad Ax \geq b,$$

la funzione lagrangiana diventa

$$L(x, \mu) = c^T x + \lambda^T (b - Ax) = \lambda^T b + (c^T - \lambda^T A)x.$$

Perciò il rilassamento lagrangiano è

$$(R_\lambda) \qquad \lambda^T b + \min_{x \in \mathbb{R}^n} (c^T - \lambda^T A)x$$

che può essere risolto in formula chiusa:

$$\psi(\lambda) = \begin{cases} -\infty & \text{se } c^T - \lambda^T A \neq 0 \\ \lambda^T b & \text{altrimenti} \end{cases}$$

Il duale è quindi

$$(D) \qquad \max \psi \quad \lambda^T A = c^T, \lambda \geq 0.$$

Vale la dualità forte, a meno che entrambe le regioni ammissibili non siano vuote (in questo caso $v(D) = -\infty < v(P) = +\infty$).

Problemi quadratici

Iniziamo da un problema quadratico molto semplice:

$$(P) \qquad \frac{1}{2}\min \|x\|^2 \quad Ax = b.$$

In questo caso

$$L(\mu) = [\frac{1}{2}\|x\|^2 + \mu^T Ax] - \mu^T b. \tag{3.14}$$

Il minimo in x della lagrangiana si ha quando $\nabla_x L(x,\mu) = x + (\mu^T A)^T = 0$. Se sostituiamo $x = -(\mu^T A)^T = -A^T\mu$ in (3.14) otteniamo che

$$\psi(\mu) = -\frac{1}{2}\mu^T(AA^T)\mu - \mu^T b,$$

dunque il problema duale diventa

$$(D) \qquad \max -\frac{1}{2}\mu^T(AA^T)\mu - \mu b \quad \mu \in \mathbb{R}^m$$

Che è un problema quadratico non vincolato.

Analizziamo ora un caso più generale:

$$(P) \qquad \min \frac{1}{2}x^T Qx + q^T x \quad Ax \geq b$$

con Q definita positiva. La lagrangiana è

$$L(x,\lambda) = \frac{1}{2}x^T Qx + q^T x + \lambda^T(b - Ax) = \lambda^T b + \left[\frac{1}{2}x^T Qx + (q^T - \lambda^T A)x\right],$$

il cui minimo è quando $\nabla_x L = Qx + q - (\lambda^T A)^T = 0$ e quindi quando $x = Q^{-1}((\lambda^T A)^T - q)$. Definiamo $v = (\lambda^T A)^T - q$, e quindi $x = Q^{-1}v$, e otteniamo che

$$\psi(\lambda) = \frac{1}{2}v^T Q^{-1}v - v^T Q^{-1}v + \lambda^T b = -\frac{1}{2}v^T Q^{-1}v + \lambda^T b.$$

Dunque

$$(D) \qquad \max -\frac{1}{2}v^T Q^{-1}v + \lambda^T b \qquad \lambda^T A - v = q, \quad \lambda \geq 0$$

3.4 Algoritmi per l'ottimizzazione vincolata

Per la trattazione di questi algoritmi, ci restringeremo al caso di problemi quadratici, ovvero dove $f(x) = \frac{1}{2}x^TQx + q^Tx$, e l'insieme delle regioni ammissibili è $X = \{x : Ax \leq b\}$.

Un'importante notazione che useremo è la seguente: dati $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, e un sottoinsieme $B \subseteq 1, \ldots, m$, denotiamo con A_B la matrice $(A_{ij})_{i\in B}$, e con b_B il vettore $(b_i)_{i\in B}$. Con questa notazione il cono tangente in un punto x è

$$T_X(x) = F_X(x) = \{d \in \mathbb{R}^n : A_{\mathcal{A}(x)}d \leq 0\}$$

Prima di passare al primo algoritmo vero e proprio, trattiamo un caso più semplice.

Problemi quadratici con vincoli di uguaglianza

$$(P) \qquad \min\{\frac{1}{2}x^tQx + qx : Ax = b\}$$

In questo caso, se $Q \succeq 0$, il problema è convesso, e per risolverlo basta risolvere il sistema dato dalle (KKT)

$$Qx + A^T\mu + q = 0 \tag{3.15}$$
$$Ax = b \tag{3.16}$$

che in forma matriciale è

$$\begin{pmatrix} Q & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ \mu \end{pmatrix} = \begin{pmatrix} -q \\ b \end{pmatrix}.$$

Questo sistema appare in molti contesti diversi e va risolto efficientemente (vedasi Poloni).

3.4.1 Metodo Active Set per problemi quadratici

Come visto nella sezione precedente, se sapessimo l'insieme di vincoli attivi della soluzione ottimale x_*, basterebbe risolvere un problema lineare. Ovviamente non sapendo qual è x_*, non conosciamo neanche $\mathcal{A}(x_*)$, perciò un'idea è quella di stimarlo:

- Dato x, prendiamo $B = \mathcal{A}(x)$, e risolviamo il problema quadratico ristretto a B
$$\min\{\frac{1}{2}x^TQx + q^Tx : A_Bx = b_B\}$$
Così facendo otteniamo una soluzione $(\bar{x}, \bar{\mu_B})$

- Non sappiamo se $\bar{x}$ soddisfi tutti gli altri vincoli, cioè se dato il complementare B^C di B, valga $A_{B^C} \leq b_{B^C}$:
  - Se vale $A_{B^C}\bar{x} \leq b_{B^C}$
    - se vale anche $\mu_B \geq 0$, allora $\bar{x}$ è ottimale, perché soddisfa le condizioni KKT;
    - se invece $\bar{\mu_B} \ngeq 0$, prendiamo il minimo indice h tale per cui $(\bar{\mu}_B)_h < 0$, e ricominciamo con $B \setminus \{h\}$
  - Se invece $\bar{x}$ non soddisfa tutti gli altri vincoli, comunque $d = \bar{x} - x$ è una direzione di decrescita: prendiamo perciò
$$\alpha = \min\left\{\alpha_i = \frac{b_i - A_ix}{A_id} : A_id > 0, i \notin B\right\}$$
E ricominciamo con $x + \alpha d$ come nuovo punto, aggiornando B di conseguenza.

Lemma 3.4.1. *α appena definito è il massimo passo che possiamo fare per cui $x + \alpha d$ è ancora una soluzione ammissibile. Inoltre $\alpha < +\infty$.*

Dimostrazione. Ricordiamo che $x + \alpha d$ è ammissibile se e solo se, per ogni i, $A_i(x + \alpha d) \leq b_i$.

Se $i \in B$, $A_i(x + \alpha d) = A_i((1 - \alpha)x + \alpha\bar{x}) = (1 - \alpha)b_i + \alpha b_i = b_i$.

Se $Ad \leq 0$, chiaramente non c'è alcun problema.

Se invece $A_id > 0$ ha che se $x+\alpha' d$ è ammissibile, dev'essere $b_i - A_ix - \alpha' A_id \geq 0$, ovvero $\alpha' \leq \alpha_i = \frac{b_i - A_ix}{A_id}$. Ciò deve valere per ogni i per cui vale $A_id > 0$, perciò α' dev'essere minore o uguale del minimo tra gli α_i.

Si ha inoltre che $\alpha < 1$, perché $x + d = \bar{x}$ non è ammissibile, dunque viola un qualche vincolo $\bar{i}$, e in particolare $\alpha_{\bar{i}} < 1$. □

Si dimostra che il metodo Active Set finisce in un numero finito di passi: si può vedere infatti che la funzione obbiettivo diminuisce strettamente per ogni iterazione, quindi non viene mai ripetuto l'insieme dei vincoli attivi B. Il problema è che nel caso sfortunato potremmo analizzare ogni B possibile, e i sottoinsiemi possibili di $\{1, \ldots, m\}$ sono 2^m.

Metodo nel caso di box costraint

. Un caso particolare in cui il calcolo di α è particolarmente semplice è quello di *box costraint* ovvero vincoli della forma $\underline{x} \leq x \leq \bar{x}$. Chiamiamo $N = \{1, \dots n\}$, dove n è la dimensione di x. In questo caso l'insieme dei vincoli attivi B si può dividere in una partizione $U \cup L$, dove $l \in L$ se $x_l = \underline{x}_l$, e $u \in U$ se $x_u = \bar{x}_u$. Chiamiamo $F = N \setminus (L \cup U)$, l'insieme dei vincoli liberi. La condizione $A_B x = b_B$ ci dice che $x = (x_L, x_F, x_U) = (\underline{x}_L, x_F, \bar{x}_U)$, cioè il solo vettore x_F è libero, mentre gli altri due sono vincolati a essere uguali a $\underline{x}_L$ e $\bar{x}_U$.

Se assumiamo $\underline{x} = 0$ (possiamo prendendo $x \leftarrow x + \underline{x}$), $x = (0, x_F, \bar{x}_U)$ e quindi il problema da risolvere nell'iterazione dell'algoritmo diventa

$$\min \left\{ \frac{1}{2} x_F^T Q_{FF} x_F + (q_F + \bar{x}_U^T Q_{UF}) x_F \right\},$$

che è un problema non vincolato in dimensione minore. Q_{UF} denota la sottomatrice di Q ottenuta prendendo le righe e con indici in U e le colonne con indici in F. Analogamente Q_{FF}.

3.4.2 Gradiente proiettato

Mettiamoci ora nel caso di un problema dove la funzione obbiettivo $f \in C^1$ è generica e i vincoli sono lineari di disuguaglianza. Dato $x \in X$, una direzione che sappiamo essere di decrescita per f è $-\nabla f(x)$. Se però $-\nabla f(x)$ non è una direzione ammissibile, vogliamo prendere d ammissibile "simile" a $-\nabla f(x)$: la soluzione è prendere la *proiezione* dell'antigradiente sul cono delle soluzioni ammissibili $F(x)$. Cerchiamo dunque

$$p_{F(x)}\left(-\nabla f(x)\right) = \min\{h(d) = \frac{1}{2}\|d + \nabla f(x)\|^2 : A_{\mathcal{A}(x)} d \leq 0\}. \tag{3.17}$$

Per ogni iterazione dobbiamo dunque risolvere un problema quadratico su un costraint di cono poliedrico, ad esempio usando il metodo active set che abbiamo visto nella sezione precedente.

Lemma 3.4.2. *Se $p_{F(x)}\left(-\nabla f(x)\right) = 0$, allora $\lambda A + \nabla f(x) = 0$, cioè valgono le (KKT) e quindi x è ottimale*

Dimostrazione. $\frac{1}{2}\nabla h(d) = d + \nabla f(x)$, e quindi in particolare $\nabla h(0) = \nabla f(x)$. Dunque, se $p_{F(x)}(-\nabla f(x)) = 0$, usando la (KKT) (3.10) applicata al problema (3.17), si ottiene che esiste $\lambda' \in \mathbb{R}^{|\mathcal{A}(x)|}$ tale per cui

$$\nabla h(0) + \lambda' A_{\mathcal{A}(x)} = \nabla f(x) + \lambda' A_{\mathcal{A}(x)} = 0.$$

Prendendo $\lambda = \lambda'$ negli indici di $\mathcal{A}(x)$ e $\lambda = 0$ negli altri indici, otteniamo che $\nabla f(x) + \lambda A = 0$, cioè vale la condizione (3.10) per il problema originario. Per costruzione valgono ovviamente anche le altre due condizioni (KKT), perciò x è ottimale. □

Grazie al lemma 3.4.2, siamo sicuri che la condizione $\|d\| < \epsilon$ sia un buon criterio di stop per il metodo iterativo

Metodo nel caso di box costraint

Il problema (3.17) può essere costoso da risolvere, ma nel caso di box costraint $\underline{x} \leq x \leq \bar{x}$, è molto semplice. Possiamo decomporre $X = X_1 \times \dots X_n$, con $X_i = [\underline{x}, \bar{x}]$, e di conseguenza decomporre anche $F(x) = F(x)_1 \times \dots F(x)_n$, con

$$F(x)_i = \begin{cases} \mathbb{R}_{\geq 0} & x_i = \underline{x}_i \\ \mathbb{R}_{\leq 0} & x_i = \bar{x}_i \\ \mathbb{R} & x_i \in (\underline{x}_i, \bar{x}_i) \end{cases}$$

Cioè se $x_i = \underline{x}_i$, x_i può solo aumentare, e analogamente se $x_i = \bar{x}_i$, può solo diminuire, mentre se x_i è "libero" possiamo farlo variare in entrambe le direzioni. Possiamo perciò minimizzare $\|d_i + \nabla f(x)_i\|$ con $d_i \in F(x)_i$ uno a uno ottenendo n problemi molto più semplici e parallelizzabili-.

Gradiente proiettato di Goldstein

Una variante del gradiente proiettato è quella in cui "prima muoviamo, poi proiettiamo". Se al passo i ci troviamo in x^i, calcoliamo

$$y^i = x^i + \alpha \nabla f(x^i)$$
$$x^{i+1} = p_X(y^i)$$

Questo metodo non ci garantisce che $f(x^{i+1}) < f(x^i)$, e in generale per la convergenza serve un passo appropriato, ad esempio, se f è L-lipschitz, $\alpha = \frac{1}{L}$. Tuttavia in alcuni casi si può garantire un risultato di convergenza simile a quello del metodo del gradiente nel caso non vincolato.

Gradiente proiettato di Rosen

Se la proiezione su F_X è troppo costosa, possiamo proiettare sul bordo del cono ammissibile $\partial F_X = \{d \in \mathbb{R}^n : A_{\mathcal{A}(x)} d = 0\}$. La proiezione su questo insieme è molto semplice: è un problema quadratico con vincoli di uguaglianza, e se $\bar{A} = A_{\mathcal{A}(x)}$ ha rango massimo, abbiamo una formula chiusa per d e μ:

$$\mu = -[\bar{A}\bar{A}^T]^{-1}\bar{A}\nabla f(x), \tag{3.18}$$

$$d = (I - \bar{A}^T[\bar{A}\bar{A}^T]^{-1}\bar{A})(-\nabla f(x)) \tag{3.19}$$

3.4.3 Metodo di Frank-Wolfe

Nel gradiente proiettato dobbiamo ad ogni passo risolvere un problema quadratico con vincoli rappresentati da F_X. Nel metodo di Frank-Wolfe (o del gradiente condizionale), ad ogni passo risolviamo un sotto-problema in cui la funzione obbiettivo è lineare:

$$\bar{x}_i = \arg\min\{\nabla f(x_i) \cdot x : Ax \leq b\}$$

Ad ogni passo minimizziamo quindi il prodotto scalare tra x e $\nabla f(x_i)$: in questo modo minimizziamo il modello del primo ordine in x_i all'interno di X. Detta $d_i = \bar{x}_i - x_i$, prendiamo $x_{i+1} \leftarrow x_i + \alpha_i d_i$ (con $\alpha_i \in [0, 1]$ risultato di una line-search lungo d_i). Notiamo che, per convessità di X, $x_{i+1} \in X$.

Lemma 3.4.3. [1] *Se $\nabla f(x) \cdot d = 0$, allora x è un ottimo locale.*

Lemma 3.4.4. *Se $\nabla f(x) \cdot d \neq 0$, allora $\nabla f(x) \cdot d < 0$, cioè d è una direzione di discesa.*

Dimostrazione. $\bar{x}$ è il minimo di $L_x(y) = \nabla f(x) \cdot y$, perciò in particolare $\nabla f(x) \cdot \bar{x} \leq \nabla f(x) \cdot x$, e dunque $\nabla f(x) \cdot (\bar{x} - x) = \nabla f(x) \cdot d \leq 0$. Poiché non vale l'uguaglianza, otteniamo la tesi. □

Abbiamo quindi il classico criterio di stop $\nabla f(x) \cdot d \geq -\varepsilon$. Possiamo però dire di più sul metodo nel caso di funzioni convesse: chiamiamo

$$v^* = f(x) + \nabla f(x) \cdot (\bar{x} - x) = L_x(\bar{x}).$$

[1] Da qui in poi ometto gli indici i per semplicità di notazione.

Se f è convessa, il modello del primo ordine approssima dal basso f, cioè $L_x(y) \leq f(y)$ per ogni x, y. Ciò vale in particolare se $y = x^*$ è l'ottimo: per ogni x vale dunque

$$\begin{aligned} v(P) =& \min\{f(y) : y \in X\} \geq \min\{L_x(y) : y \in X\} \\ =& f(x) + \min\{\nabla f(x) \cdot y : y \in X\} - \nabla f(\mathbf{x}) \cdot x = f(x) + \nabla f(x) \cdot (\bar{x} - x) = v^*. \end{aligned}$$

Ciò ci dice che v^* fornisce un *lower bound* del valore ottimo della funzione. Possiamo perciò per la prima volta stimare il gap tra $f(x)$ e $v(P)$:

$$f(x) - v^* \geq f(x) - v(P).$$

In particolare, se teniamo in memoria il miglior lower bound l (quindi il *massimo* v^* per ogni iterazione) abbiamo un nuovo criterio di stop: $f(x_i) - l \leq \varepsilon$

Metodo nel caso di box-costraint

Nel caso di Box costraint $X = \{x : \underline{x} \leq x \leq \bar{x}\}$, il problema lineare è risolvibile in modo immediato: basta prendere per ogni componente k

$$x_k = \begin{cases} \underline{x}_k & \text{se } \nabla f(x)_k \geq 0 \\ \bar{x}_k & \text{se } \nabla f(x)_k < 0 \end{cases}$$

Stabilizzazione del metodo di Frank Wolfe

La convergenza del metodo di Frank Wolfe risulta essere piuttosto lenta: ad ogni passo cerchiamo $\bar{x}$ che minimizza il modello del primo ordine L_x dentro tutto l'insieme ammissibile X. In altre parole ci fidiamo del modello anche molto lontano da x, dove non abbiamo garanzie che il modello stesso sia buono. Per ovviare a questo problema abbiamo possiamo provare a regolarizzare il problema lineare, aggiungendo un termine $\gamma\|x - y\|_2^2$ a $L_x(y)$. In questo modo però trasformiamo il sottoproblema in un altro problema quadratico.

Un'altra soluzione è quella della stabilizzazione tramite regioni di fiducia: possiamo aggiungere un termine $\|x - y\|_\infty \leq \tau$ ai vincoli del problema, o in altre parole dei box costraint $x_i - \tau \leq y_i \leq x_i + \tau$. Il problema rimane così lineare, e i vincoli che aggiungiamo non rendono quasi mai il problema più difficile.

3.4.4 Metodo duale

Fino ad ora abbiamo analizzato metodi che si basavano sul problema primale, ottenendo ad ogni passo soluzioni primali x^k ammissibili, e trovando le variabili duali $\lambda \geq 0$ ammissibili solo una volta raggiunto l'ottimo. Un approccio alternativo è invece considerare direttamente il duale: prendiamo [2]

$$\psi(\lambda) = \min_{x \in \mathbb{R}^n} \{\frac{1}{2} x^T Q x + q^T x + \lambda(b - Ax)\},$$

Abbiamo visto che se Q è definita positiva, l'ottimo si ha in $x(\lambda) = -Q^{-1}(A^T\lambda + q)$. Possiamo allora risolvere il problema duale

$$(D) \qquad \max\{\psi(\lambda) : \lambda \geq 0\}.$$

In generale, $\psi \in C^1$, ma $\psi \notin C^2$, perciò (D) può essere risolto con qualsiasi metodo adatto a funzioni C^1. Il problema di questo metodo è che, dato λ_i, $x(\lambda_i)$ è ammissibile per (P) se e solo se sia $x(\lambda_i)$ che λ_i sono ottimali per i rispettivi problemi, quindi non abbiamo a disposizione nessuna soluzione ammissibile per (P), finchè non troviamo l'ottimo. Tuttavia se i vincoli di (P) sono abbastanza semplici, possiamo calcolare *l'euristica lagrangiana*, ovvero la proiezione

$$x_i = p_X(x(\lambda_i)) = \min\{\|x - x(\lambda_i)\| : x \in X\}$$

Rilassamento Lagrangiano parziale

In alcuni casi, invece che far passare tutti i vincoli nella funzione obbiettivo tramite i moltiplicatori di Lagrange, può essere utile risolvere un rilassamento parziale: se il nostro problema è della forma $\min_x f(x)$, con x soggetto ai vincoli $Ax \leq b$ e $Ex \leq d$ (dove stiamo dividendo a nostro piacimento i vincoli di uguaglianza in due parti), risolviamo

$$(R_\lambda) \qquad \min_x\{f(x) - \lambda(b - Ax) : Ex \leq d\}$$

Questo approccio è molto utile quando la funzione è separabile, cioè non lega le variabili tra di loro, e ci sono pochi vincoli che legano le variabili, mentre gli altri sono anch'essi vincoli separabili. In questo modo la $\psi(\lambda)$ risultante si

[2] Consideriamo il vincolo$Ax \leq b$ come abbiamo supposto fin'ora, anziché $Ax \geq b$, come nella sezione sul duale di funzioni quadratiche

può riscrivere come una somma di funzioni di meno variabili, e il problema di massimo risultante si decompone in sotto-problemi più piccoli, che possono anche essere parallelizzati.

3.4.5 Barrier Methods (acenno)

L'approccio duale ha molti pro, come il fatto che è pressochè non vincolato, e alcuni contro, come il fatto che non abbiamo una soluzione fino alla fine. Inoltre ψ è C^1 ma non C^2 anche se il problema primale è differenziabile infinite volte. Se vogliamo tutti i pro dell'approccio duale senza i contro, possiamo risolvere un problema leggermente diverso:

$$(P_\gamma) \qquad \min_x \{f_\gamma(x) = f(x) - \gamma \sum_{i=}^{m} \log(b_i - A_i x)\}$$

Il termine logaritmico produce una barriera ai bordi della regione ammissibile: infatti $-\log(x) \to +\infty$ se $x \to 0$. Il problema perciò è molto simile a risolvere il problema con le variabili indicatrici (3.2). (P_γ) è, al variare di γ, strettamente convesso, perciò per ogni γ esiste un'unica soluzione x_γ . Chiamiamo *central path* la curva formata dagli x_γ per γ che va da $+\infty$ a 0. Per $\gamma \to 0$, $x_\gamma \to x_*$, soluzione ottimale del problema iniziale.

L'implementazione in pratica usa una tecnica primale-duale, che trova una soluzione primale ammissibile e una duale ammissibile, risolvendo un sistema lineare ad ogni passo (simile al metodo di Newton). Il sistema deriva dalle condizioni KKT del problema, riscritte in modo da tenere conto del fattore γ (cfr. bibliografia sulle slide per quanto possibile, qua Frangioni ha glissato completamente e non si è capito nulla). Il sistema risultate è simmetrico e di grosse dimensioni, perciò un metodo basato sui sottospazi di Krylov può risultare adatto per la risoluzione.

Il metodo in pratica è il più veloce di tutti in termine di numero di iterazioni, ma le iterazioni sono molto costose (risoluzione di un sistema lineare $\implies O(n^3)$ ad iterazione)

Parte II

Analisi numerica

Capitolo 4

Introduzione

Questa sezione coprirà alcune tecniche per la risoluzione di alcuni problemi importanti di algebra lineare numerica. In particolare i due problemi problemi principali che verranno analizzati saranno:

- La risoluzione di **sistemi lineari**: ovvero Data la matrice *quadrata* $A \in \mathbb{R}^{m\times m}$, e il vettore $b \in \mathbb{R}^m$, trovare il vettore x (che è unico sotto determinate condizioni) tale per cui $Ax = b$
- La risoluzione di problemi ai **minimi quadrati**: ovvero data la matrice (non necessariamente quadrata) $A \in \mathbb{R}^{m\times n}$, trovare $\min_x \|Ax - b\|$

Entrambi i problemi hanno svariate applicazioni in ottimizzazione e nel machine learning

Esempio. Le condizioni KKT per funzioni quadratiche (3.15) si traducono in un sistema lineare con matrice simmetrica. Vedremo come risolverlo in modo efficiente, sfruttando la simmetria

Esempio. (*regressione lineare*) La regressione lineare è il più semplice algoritmo che si studia nel corso di Machine Learning. Data una funzione f (sconosciuta), e dei dati $(x_i, y_i = f(x_i))_{i=1}^n$, vogliamo trovare la funzione lineare $h(x) = a^T x + b$ che approssimi f. Usando la MSE (mean squaded error), otteniamo che vogliamo minimizzare [1]

$$\sum_{i=1}^{n}(h(x_i) - y_i)^2 = \sum_{i=1}^{n}(a^T x_i - y_i)^2 = \|Xa - y\|_2^2,$$

[1] Possiamo includere b nella parte lineare, basta aggiungere una componente ad a e agli x_i, ponendo l'ultima componente di ciascun x_i uguale a 1.

dove

$$X = \left(\begin{array}{c} x_1^T \\ \hline \dots \\ \hline x_n^T \end{array}\right), y = (y_1, ..., y_n)^T$$

La regressione lineare si può dunque ricondurre con facilità ad un problema ai minimi quadrati (dove X è la matrice, e a il vettore, n.b.).

4.1 Richiami di Algebra lineare

Prodotto matrice-vettore

Data una matrice $A \in \mathbb{R}^{m \times n}$, e un vettore $x \in \mathbb{R}^n$, il prodotto matrice vettore Ax è dato dal prodotto riga-colonna di A e x, ovvero, se $Ax = b$, $b_i = \sum_j A_{ij} x_j$. Osserviamo che ciò equivale al fatto che b è una combinazione lineare delle righe di A:

$$b = \begin{pmatrix} A_{11} \\ \vdots \\ A_{m1} \end{pmatrix} x_1 + \begin{pmatrix} A_{12} \\ \vdots \\ A_{m2} \end{pmatrix} x_2 + \cdots + \begin{pmatrix} A_{1n} \\ \vdots \\ A_{mn} \end{pmatrix} x_n$$

Diciamo che b appartiene all'immagine di A.

Base

Una base di uno spazio vettoriale (nel nostro caso sarà sempre $\mathbb{R}^m$), è un'insieme ordinato di vettori $\{v_1, \dots, v_m\}$ tali che ogni vettore è esprimibile *in maniera unica* come combinazione lineare di tali vettori. La base più semplice è la base canonica, ovvero

$$e_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \dots, e_m = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

Prodotto matrice-matrice

Il prodotto matrice-matrice si esegue facendo il prodotto scalare tra le righe della prima matrice e le colonne della seconda. Se $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{n\times k}$, $AB = C \in \mathbb{R}^{m\times k}$ (il trucco per memorizzarlo è che le dimensioni interne devono coincidere, e si cancellano, mentre restano quelle esterne). Il prodotto matrice-matrice è piuttosto costoso: sono $m(n-1)k$ operazioni, o $O(mnk)$. È quindi cubico se le tre dimensioni coincidono (vedremo che la maggior parte degli algoritmi che analizzeremo sono anche cubici).

Rango

Il *rango* di una matrice $A \in \mathbb{R}^{m\times n}$ è il numero di colonne linearmente indipendenti, o, equivalentemente, il numero di righe linearmente indipendenti (si può dimostrare che le due definizioni coincidono). Il rango di A non può quindi essere più grande della dimensione minore di A: $\text{rank}(A) \leq \min\{m, n\}$

4.2 Ortogonalità

Introduciamo un concetto, e una classe di matrici, che compariranno spesso durante i prossimi capitoli. Ricordiamo che dati due vettori della stessa dimensione $x, y \in \mathbb{R}^m$, il prodotto scalare tra x e y è

$$x^T y = x_1 y_1 + \cdots + x_m y_m.$$

Questo prodotto coincide con il prodotto matrice-vettore dove la matrice è $x^T \in \mathbb{R}^{1\times m}$. Il prodotto scalare è ovviamente simmetrico, cioè $x^T y = y^T x$. Definiamo la norma 2 di un vettore x come la radice quadrata del prodotto scalare di x con se stesso:

$$\|x\|_2 = \sqrt{x^T x} = \sqrt{x_1^2 + \cdots + x_m^2}$$

Definizione. Diciamo che due vettori sono *ortogonali* se il loro prodotto scalare è uguale a 0. Due vettori sono *ortonormali* se il loro prodotto scalare è nullo, cioè sono ortogonali, e inoltre la loro norma 2 è uguale a 1:

$$x^y = 0, \quad \|x\|_2 = \|y\|_2 = 1$$

4.2.1 Matrici ortogonali

Definizione. Una matrice quadrata $U \in \mathbb{R}^{m\times m}$ si dice *ortogonale* se $U^TU = UU^T = I$, o equivalentemente, $U^{-1} = U^T$.

Osservazione 9. Se $U = (u_1|\dots|u_m)$, dove gli u_i sono le colonne di U, gli u_i sono ortonormali (non solo ortogonali!): infatti

$$(U^TU)_{ij} = u_i^Tu_j = I_{ij} = \begin{cases} 1 & \text{se } i = j \\ 0 & \text{se } i \neq j \end{cases}$$

Le matrici ortogonali sono importanti perché preservano la norma 2:

Proposizione 4.2.1. *Se U è ortogonale e x è un vettore, $\|Ux\|_2 = \|x\|_2$*

Dimostrazione. $\|Ux\|_2^2 = (Ux)^T(Ux) = x^TU^TUx = x^T(U^TU)x = x^TIx = x^Tx = \|x\|_2^2$ □

Dal punto di vista geometrico, la trasformazione dello spazio associata a U non distorce gli angoli e le grandezze in nessuna direzione.

Matrici con colonne ortonormali

Notiamo che se un matrice $U_1 \in \mathbb{R}^{m\times n}$ ha colonne ortonormali e $m > n$, allora $U_1^TU_1 = I_n$, usando la stessa dimostrazione che abbiamo usato prima, ma $U_1U_1^T \neq I_m$: infatti $U_1U_1^T$ può avere rango al massimo uguale a quello di U_1, e il rango di U_1 è esattamente n (le n colonne sono ortonormali, quindi sono necessariamente linearmente indipendenti). Dunque $U_1U_1^T$ non può essere uguale a I_m, che ha rango $m > n$.

Autovalori e Teorema spettrale

Data $A \in \mathbb{R}^{m\times m}$, se vale per un certo vettore x e scalare λ che $Ax = \lambda x$, chiamiamo λ autovalore di A, e x autovettore di A. Spesso possiamo decomporre A nella seguente maniera: $A = V\Lambda V^{-1}$, dove $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_m)$ è la matrice diagonale contenenti tutti gli autovettori di A. Non sempre questa decomposizione esiste: ad esempio una matrice potrebbe non avere autovettori reali, ma solo autovettori complessi. Anche eliminando questo caso (lavorando per esempio in uno spazio vettoriale complesso $\mathbf{C}^m$, non è comunque detto che la decomposizione spettrale appena descritta esiste. Tuttavia le cose vanno bene se A è simmetrica: la decomposizione spettrale esiste, e la matrice di cambio di base può essere scelta ortogonale:

Teorema 4.2.1. ***(Teorema spettrale reale)***
Se $A = A^T$, allora esiste una matrice ortogonale *U tale che*

$$A = U\Lambda U^{-1} = U\Lambda U^T \tag{4.1}$$

4.2.2 Matrici definite positive

Prendiamo una matrice simmetrica $\in \mathbb{R}^{m\times m}$. A Q è associata la funzione $f(x) = x^TQx$, detta *forma quadratica*. Questa funzione è l'estensione di una parabola in m dimensioni, e si incontra molto spesso nella parte di ottimizzazione. Vale un Teorema importante:

Teorema 4.2.2. *Sia λ_{min} l' autovvalore minimo di Q, e λ_max l'autovalore massimo. Allora*

$$\lambda_{min}\|x\|_2^2 \leq x^TQx \leq \lambda_{max}\|x\|_2^2$$

Dimostrazione. Se Q è diagonale, $x^TQx = \lambda_1x_1^2 + \cdots + \lambda_mx_m^2$.
$\lambda_{min}(x_1^2 + \cdots + x_m^2) \leq \lambda_1x_1^2 + \cdots + \lambda_mx_m^2 \leq \lambda_{max}(x_1^2 + \cdots + x_m^2)$
Se Q non è diagonale, usiamo la decomposizione spettrale (4.1), e poiché U è ortogonale, la norma di x resta uguale:
$x^TU\Lambda U^Tx = y^T\Lambda y$ con $y = U^Tx$. □

Definizione. Q è *definita positiva* se il suo autovalore minimo è positivo: $\lambda_{min} > 0$. Q è *semidefinita positiva* se $\lambda_{min} \geq 0$.

Se Q è semidefinita positiva, quindi, $x^TQx \geq 0$ per ogni x. Se Q è anche definita positva, invece, $x^TQx > 0$ per ogni $x \neq 0$.

Capitolo 5

SVD

5.1 Introduzione alla SVD

Sia $A \in \mathbb{R}^{m\times m}$. Possiamo definire la sua *decomposizioni ai valori singolari (SVD)* come

$$A = U\Sigma V^T, \tag{5.1}$$

dove

- U e $V \in \mathbb{R}^{m\times m}$ sono matrici ortogonali,

$$U = \begin{pmatrix} u_1 & u_2 & \dots & u_k \end{pmatrix} \qquad V^T = \begin{pmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_m^T \end{pmatrix}$$

- $\Sigma \in \mathbb{R}^{m\times m}$ è una matrice diagonale, $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_m)$, con $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$. Gli scalari σ_i sono detto valori singolari di A

Se $A \in \mathbb{R}^{m\times n}$ è u a matrice non quadrata, possiamo definire similmente la SVD $A = U\Sigma V^T$, dove

- $U \in \mathbb{R}^{m\times m}$ matrice ortogonale
- $V \in \mathbb{R}^{n\times n}$ matrice ortogonale
- $\Sigma \in \mathbb{R}^{m\times n}$ è "diagonale" nel senso che, detto $k = \min\{m.n\}$, le uniche entrate non nulle di Σ sono $(\Sigma_{i,i})_{i=1}^k$

Teorema 5.1.1. *Per ogni $A \in \mathbb{R}^{m\times n}$, esiste sempre una SVD come abbiamo definito sopra*

Thin SVD

Usando la SVD, dato $A \in \mathbb{R}^{m\times n}$, possiamo scrivere $A = U\Sigma V^T = u_1\sigma_1 v_1^T + \cdots + u_k\sigma_k v_k^T$, dove $k = \min m, n$. Dunque non ci serve davvero Tutta la matrice Σ e entrambe le matrici quadrate, ma possiamo conservare in memoria solo i primi k vettori riga per U e colonna per V. Se $m < n$, quindi, U rimane uguale, mentre

$$V^T = \begin{matrix} V_1^T \\ V_2^T \end{matrix} \begin{matrix} m \\ n-m \end{matrix},$$

e possiamo calcolare solo V_1.

Al contrario, se $m > n$, V rimane uguale, mentre

$$U = \begin{matrix} n & m-n \\ \big(U_1 & U_2 \big) \end{matrix},$$

e calcoliamo solo U_1.

5.1.1 Proprietà della SVD

Elenchiamo ora alcune proprietà importanti della SVD:

1. $\text{rank}(A) = |\{\sigma_i \neq 0\}|$. Infatti, se esiste un r tale che $\sigma_1 > \sigma_2 > \cdots > \sigma_r > \sigma_{r+1} = \cdots = \sigma_k = 0$, allora $A = u_1\sigma_1 v_1^T + \cdots + u_r\sigma_r v_r^T$
2. $\text{im}(A) = \text{span}\{u_1, \ldots, u_r\}$
3. $\ker(A) = \text{span}\{v_{r+1}, \ldots, v_n\}$: infatti, se $i > r$, $Av_i = u_1\sigma_1(v_1^T v_i) + \cdots + u_r\sigma_r(v_r^T v_i)$, e ciascun $v_j^T v_i$ è uguale a 0, poiché V è ortogonale.
4. Se $A \in \mathbb{R}^{m\times m}$ è invertibile, allora la SVD di A^{-1} è $(V^T)^{-1}\Sigma^{-1}U^{-1} = V\Sigma^{-1}U^T$. Notiamo che $\Sigma^{-1} = \text{diag}(1/\sigma_1, \ldots, 1/\sigma_n)$

La SVD è inoltre legata alla decomposizione spettrale della matrice $A^T A$. Notiamo innanzitutto che, qualsiasi sia A, tale matrice è simmetrica. Prendiamo una generica $A \in \mathbb{R}^{m\times n}$, e notiamo che

$$A^T A = (U\Sigma V^T)^T U\Sigma V^T = V\Sigma^T U^T U\Sigma V^T = V\Sigma^T\Sigma V^T$$

Dunque calcolando la SVD di A, possiamo facilmente ricavare anche la decomposizione spettrale di $A^T A$, utilizzando le stesse matrici prodotte dalla SVD. Ora, nel caso quadrato, $\Sigma^T\Sigma = \Sigma^2 = \text{diag}(\sigma_1^2, \ldots, \sigma_m^2)$, ma nel

caso non-quadrato bisogna fare un po' più di attenzione: se $m \geq n$, vale effettivamente

$$\Sigma^T\Sigma = \left(\begin{array}{ccc|c} \sigma_1 & & & \\ & \ddots & & 0 \\ & & \sigma_n & \end{array}\right) \left(\begin{array}{c} \sigma_1 \\ & \ddots \\ & & \sigma_n \\ \hline & 0 & \end{array}\right) = \begin{pmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_n^2 \end{pmatrix}$$

Se $m < n$, invece

$$\Sigma^T\Sigma = \left(\begin{array}{c} \sigma_1 \\ & \ddots \\ & & \sigma_n \\ \hline & 0 & \end{array}\right) \left(\begin{array}{ccc|c} \sigma_1 & & & \\ & \ddots & & 0 \\ & & \sigma_n & \end{array}\right) = \left(\begin{array}{ccc|c} \sigma_1^2 & & & \\ & \ddots & & 0 \\ & & \sigma_n^2 & \\ \hline & 0 & & 0 \end{array}\right)$$

5.1.2 SVD e norme di matrici

Definizione. Una norma di matrici è una funzione $\|\cdot\| : \mathbb{R}^{m\times n} \to \mathbb{R}$ che soddisfa:

- $\|A\| \geq 0$, con $\|A\| = 0$ solo se $A = 0$.
- omogeneità: $\|\alpha A\| = |\alpha|\|A\|$ per ogni $\alpha \in \mathbb{R}$
- subadditività: $\|A + B\| \leq \|A\| + \|B\|$
- submoltiplicatività: $\|A \cdot B\| \leq \|A\| \cdot \|B\|$

Definizione. Data una norma vettoriale $\|\dot{\|}$ su $\mathbb{R}^n$ e una matrice $A \in \mathbb{R}^{m\times n}$, definiamo la norma di matrice indotta $\|A\|$ come

$$\|A\| = \max_{v\in\mathbb{R}^n, v\neq 0} \frac{\|Av\|}{\|v\|}$$

Tale funzione soddisfa le proprietà delle norme appena definite, e inoltre vale

- $\|Av\| \leq \|A\| \cdot \|v\|$ per ogni vettore v

Norma euclidea

Se $\|\cdot\| = \|\cdot\|_2$ è la norma euclidea, $U \in \mathbb{R}^{m\times m}$ è una matrice ortogonale, e $A \in \mathbb{R}^{m\times n}$ è una matrice qualsiasi, allora $\|UA\| = \|A\|$: infatti per ogni vettore w, $\|Uw\| = \|w\|$, dunque, per ogni vettore v, $\|UAv\|/\|v\| = \|U(Av)\|/\|v\| = \|Av\|/\|v\|$.

Come conseguenza, $\|A\| = \|U\Sigma V^T\| = \|\Sigma\| = \sigma_1$: la seconda uguaglianza segue dal fatto che U e V sono ortogonali, la terza segue da

$$\begin{aligned}\Sigma v &= \left\| \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_m \end{pmatrix} \begin{pmatrix} v_1 \\ \vdots \\ v_m \end{pmatrix} \right\| = \left\| \begin{pmatrix} \sigma_1 v_1 \\ \vdots \\ \sigma_m v_m \end{pmatrix} \right\| \\ &= \sqrt{\sigma_1^2 u_1^2 + \cdots + \sigma_m^2 u_m^2} \leq \sqrt{\sigma_1^2 (u_1^2 + \cdots + u_m^2)} = \sigma_1,\end{aligned}$$

con l'uguaglianza che vale quando $v = (1, 0, \ldots, 0)^T$.

Norma di Frobenius

In generale quindi, $\|A\|_2 \neq \sqrt{\sum_{i,j} a_{ij}^2}$. Tale espressione è una norma, chiamata norma di Frobenius, che indicheremo con $\|A\|_F$. La norma di Frobenius non è una norma indotta, ma ha tutte le proprietà delle norme appena elencate, compresa quella sulle matrici ortogonali: $\|UA\|_F = \|A\|_F$. Dunque vale anche che $\|A\|_F = \sqrt{\sum_{i,j} a_{ij}^2} = \sqrt{\sum_{i=1}^{\min\{m.n\}} \sigma_i^2}$

5.2 Low Rank Approximation e PCA

La SVD ci permette di risolvere facilmente un problema molto interessante in vari campi: Data una matrice $A \in \mathbb{R}^{m\times n}$, e un intero $k < \min\{m, n\}$ qual è la matrice $X \in \mathbb{R}^{m\times n}$ di rango $\text{rank}(X) \leq k$ che la approssima meglio? Con approssimare meglio intendiamo che, scelta una certa norma $\|\cdot\|$, la norma di $A - X$ è minima:

$$\min_{\substack{X\in\mathbb{R}^{m\times n},\\ \text{rank}(X)\leq k}} \|A - X\| \tag{5.2}$$

Teorema 5.2.1. ***(Eckart-Young)*** *Se $\|\cdot\| = \|\cdot\|_2$ o $\|\cdot\| = \|\cdot\|_F$, allora la matrice che risolve il problema (5.2) è $X_k = u_1\sigma_1 v_1^T + \cdots + u_k\sigma_k v_k^T$*

Per semplicità supponiamo $m > n$.

Notiamo che $A - X = u_{k+1}\sigma_{k+1}v_{k+1}^T + \cdots + u_n\sigma_n v_n^T$, e questa equivale a una SVD, dunque $\|A - X\|_2 = \sigma_{k+1}$, $\|A - X\|_F = \sqrt{\sum_{k+1}^n \sigma_i^2}$.

Esempio. Perché ci interessa questo problema? Pensiamo ad esempio ad $A \in \mathbb{R}^{256\times256}$, cioè A è un'immagine di $256\times256 = 65.536$ pixel. Se prendiamo $k = 10$, dato `[U,S,V]=svd(A)`, `X = U(1:256, 1:10)* S(1:10,1:10)*V(1:10,1:256)`, cioè per tenere in memoria l'approssimazione ci bastano $256 \times 10 + 10 \times 10 + 10 \times 256 \approx 5.000$ numeri reali. L'approssimazione con 10 valori singolari non è ovviamente molto buona, a parte rari casi [1].

5.2.1 Analisi delle Componenti Principali

L'applicazione più nota dell'svd e dell'approssimazione di rango basso è la analisi delle componenti principali o PCA. LA PCA è una tecnica che ci consente di proiettare dati di alta dimensionalità in spazi di dimensione molto più bassa. Supponiamo di avere dei dati $\{x_1, x_n\}$ con ciascun x_i un vettore di $\mathbb{R}^m$. Costruiamo la matrice $A \in \mathbb{R}^{m\times n}$, $A = (x_1|, \ldots, |x_n)$. Possiamo supporre che non tutte le features di questi vettori siano indipendenti: per esempio possiamo pensare che A sia una certa matrice di rango k + una matrice di errore $\mathcal{E}$, con ε_{ij} errori random indipendenti, distribuiti con densità gaussiana con una qualche varianza. Con questa assunzione, $\min \sum_{i,j} \epsilon_{i,j} = \min \|\mathcal{E}\|_F = \min_{\text{rank}(X)=k} \|A - X\|_F$. I valori singolari assumono allora un significato piuttosto chiaro:

- Se $\sigma_1 >> \sigma_2, \ldots$, l'errore della approssimazione di rango 1 sarà basso, cioè tutte le feature sono dipendenti.
- Se $\sigma_1, \sigma_2 >> \sigma_3, \ldots$, l'errore di $\|A - X_1\|_F$ è alto, ma quello di $\|A - X_2\|_F$ è basso, cioè due tipi di feature approssimano bene i nostri dati.
- ecc...

L'analisi delle componenti principali è solitamente eseguita non su A, ma su $\hat{A} = (\hat{x}_1|...|\hat{x}_n)$, con $\hat{x}_i = x_i - \mu$, dove μ è la media degli x_i, $\mu = \frac{1}{n}\sum x_i$. A questo punto, calcoliamo la matrice di covarianza $C = \frac{1}{n}(\hat{x}_1^T\hat{x}_1 + \cdots + \hat{x}_n^T\hat{x}_n) = \frac{1}{n}\hat{A}\hat{A}^T$. La decomposizione spettrale di C è data da $C = U\Lambda U^T$, dove U

[1] `http://timbaumann.info/svd-image-compression-demo/`

è la matrice della SVD di A, e $\Lambda_{ii} = \frac{1}{n}\sigma_i^2$. $\{u_1, \ldots, u_n\}$ ci dà una base del sottospazio span$\{\hat{x}_1, \ldots, \hat{x}_n\}$, cioè per ogni j, $\hat{x}_j = \alpha_{1j}u_1 + \cdots + \alpha_{nj}u_n$ Infatti:

$$\hat{x}_j = \hat{A}e_j = u_1(\sigma_1 v_1^T e_j) + \cdots + u_n(\sigma_n v_1^T e_n) = u_1\alpha_{1j} + \cdots + u_n\alpha_{nj}$$

in cui si ha la proprietà che gli $\hat{x}_j$ variano di più nella direzione u_1, poi nella direzione u_2, e così via. La PCA può essere quindi usata per diminuire la dimensionalità dei dati, e prendendo solo le prime due componenti principali, possiamo visualizzare il nostro dataset in due dimensioni. La PCA ha alcune limitazioni:

- Considera solamente la distanza euclidea, poichè la SVD è ottimale solo per tale norma.
- E‘ un approccio non supervisionato: le u_i non hanno etichette che descrivono specifiche features.
- I dati vengono ovviamente appiattiti e alcune informazioni vengono perse.

Capitolo 6

Problemi ai Minimi Quadrati

Un altro problema di approssimazione che vogliamo risolvere è il seguente: dati dei vettori $a_1, \ldots, a_n \in \mathbb{R}^m$, e un vettore $b \in \mathbb{R}^m$, vogliamo trovare i coefficienti $x_1, \ldots, x_n \in \mathbb{R}$ tali per cui $x_i a_i + \ldots x_n a_n$ approssimi al meglio b. In forma matriciale, vogliamo trovare

$$\min \|Ax - b\|. \tag{6.1}$$

Tale problema è detto dei minimi quadrati (Linear Least Squares) Se A è una matrice quadrata e invertibile, esiste una sola soluzione che corrisponde alla soluzione del sistema lineare $Ax = b$. Vedremo molte tecniche per questo problema successivamente. Se $m \neq n$, invece, siamo interessati in particolare a $\min \|Ax-b\|_2$, dunque la norma che useremo è quella euclidea. Nel seguito assumeremo $m \geq n$.

Definizione. A ha full-column-rank (rango massimo) se $\text{rank}(A) = n$. Ciò implica che $\ker(A) = \{\underline{0}\}$.

In questo caso, il problema dei minimi quadrati ha esattamente una soluzione; nel caso contrario, le soluzioni possono essere multiple: se $w \neq 0 \in \ker(A)$, e x è una soluzione di (6.1), allora $\|A(x+z)-b\| = \|Ax+Az-b\| = \|Ax - b\|$, dunque anche $(x + z)$ è ottimale.

Teorema 6.0.1. *A ha rango massimo $\iff$ A^TA è definita positiva*

Dimostrazione. A ha rango massimo $\iff$ $Az \neq 0$ per ogni $z \neq 0$ $\iff$ $\|Az\|_2 \neq 0$ $\iff$ $(Az)^T(Az) \neq 0$ $\iff$ $z^T(A^TA)z \neq 0$ $\iff$ A^TA è definita positiva $\square$

6.1 Equazioni Normali

Per adesso assumeremo A di rango massimo. In questo caso, possiamo calcolare una soluzione esplicita del problema attraverso la *pseudoinversa* di A. Notiamo infatti che (6.1) è ottenuto quando otteniamo anche $\min \frac{1}{2}\|Ax-b\|^2$. Svolgendo il prodotto scalare che definisce la norma 2 otteniamo

$$\begin{aligned}\min \frac{1}{2}\left(x^T A A^T x - x^T A^T b - b^T A x + b^T b\right) = \\ = \min \frac{1}{2} x^T (A^T A) x - (b^T A) x + \frac{1}{2} b^T b\end{aligned}$$

L'ultimo termine è costante, quindi non conta nella ricerca di x. Se chiamiamo $Q = A^T A$, e $q = -A^T b$, otteniamo che il problema è equivalente a

$$\min_x x^T Q x + q^T x.$$

Cioè ogni problema ai minimi quadrati è equivalente a un problema di ottimizzazione quadratica. Possiamo risolvere in forma chiusa il problema, se Q è definita positiva, cioè, come abbiamo appena mostrato, se A ha rango massimo: il minimo è in

$$x = (A^T A)^{-1} A^T b = A^+ b \tag{6.2}$$

$A^+ = (A^T A)^{-1} A^T$ è detta *pseudoinversa* di A: il nome è giustificato dal fatto che $A^+ A = (A^T A)^{-1}(A^T A) = I_n$. Non è vero però che $AA^+ = I_m$, a meno che non sia $m = n$: infatti, se A ha rango $n < m$, qualunque matrice $C = AB$ avrà rango $r \leq n < m$, mentre I_m ha rango m. La soluzione diretta del problema è detto *metodo delle equazioni normali*, ed è composto dai seguenti passi:

1. Calcolare $A^T A$: $2mn^2$ operazioni in teoria, ma per la simmetria di $A^T A$, ne bastano mn^2

2. Calcolare $A^T b$: $2mn$ operazioni

3. Risolvere il sistema lineare $(A^T A)x = A^T b$ Sarebbero $\frac{2}{3}n^3$ operazioni usando la fattorizzazione LU /l'eliminazione gaussiana, ma poiché $A^T A$ è simmetrica, il sistema può essere risolto in $\frac{1}{3}n^3$, usando la fattorizzazione di Cholevsky (che vedremo).

6.2 Metodo QR per minimi quadrati

Un'altra tecnica per risolvere il problema ai minimi quadrati è utilizzare una fattorizzazione di A, ottenuta applicando trasformazioni ortogonali. Abbiamo infatti visto che le matrici ortogonali lasciano invariata la norma di un vettore: $\|Qx\|_2 = \|x\|_2$.

6.2.1 Matrici di Householder

Un particolare tipo di matrice ortogonale che useremo è la matrice di Householder: Dato un vettore $v \in \mathbb{R}?n$, definiamo

$$H = I_n - \frac{2}{v^T v} v v^T$$

H è simmetrica e ortogonale: infatti $H^T = I^T - \frac{2}{v^T v}(vv^T)^T = I - \frac{2}{v^T v} vv^T = H$. Inoltre

$$\begin{aligned} H^T H = H^2 &= \left(I_- \frac{2}{v^T v} v v^T \right)^T \left(I_n - \frac{2}{v^T v} v v^T \right) \\ &= I - \frac{4}{v^T v} v v^T + \frac{4}{(v^T v)\cancel{(v^T v)}} v \cancel{(v^T v)} v^T = I \end{aligned}$$

Un modo alternativo di scrivere queste matrici è $I - uu^T$ con $u = v/\|v\|$. Un vantaggio di queste matrici è che possiamo calcolare il prodotto matrice vettore con $O(n)$ operazioni: $Hw = w - 2u(u^T w)$, che sono un prodotto scalare, una moltiplicazione scalare-vettore e una somma di vettori.

Lemma 6.2.1. *Dati due vettori x e $y \in \mathbb{R}^n$, chiamiamo $v = x - y$. Allora la matrice di Householder $H = I_n - \frac{2}{v^T v} vv^T$ è tale che $Hx = y$*

Possiamo cioè definire una matrice di Householder che manda un qualsiasi vettore in un altro vettore.

Teorema 6.2.1. *Per ogni $A \in \mathbb{R}^{m \times n}$, esistono $Q \in \mathbb{R}^{m \times m}$ e $R \in \mathbb{R}^{m \times n}$ tali che $A = QR$, con Q ortogonale, e R triangolare superiore, cioè $R_{ij} \neq 0$ solo se $j \geq i$. Tali matrici sono dette* fattorizzazione QR *di A*

Dimostrazione. La dimostrazione è l'algoritmo per costruire la fattorizzazione QR.

- Al passo 1, Prendiamo H_1 di Householder tale che H_1 manda la prima riga di A in un vettore multiplo di e_1:

$$H_1 \begin{pmatrix} A_{11} \\ A_{21} \\ \vdots \\ A_{m1} \end{pmatrix} = \begin{pmatrix} s_1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

  Dunque

$$H_1 A = \begin{pmatrix} s_1 & * & \dots & * \\ 0 & * & \dots & * \\ \vdots & \vdots & & \vdots \\ 0 & * & \dots & * \end{pmatrix}$$

  Chiamiamo $Q_1 = H_1$.

- Al secondo passo non possiamo prendere H_2 come H_1 ma per la seconda riga di A, perchè riempirebbe di nuovo la prima riga, e saremmo al punto di partenza. Prendiamo invece

$$Q_2 = \left(\begin{array}{c|c} 1 & 0^T \\ \hline 0 & H_2 \end{array}\right)$$

  Con $H_2 \in \mathbb{R}^{m-1 \times m-1}$ che manda $A_{2:end,2}$ in $(s_2, 0, ..., 0)^T$. In questo modo,

$$Q_2 Q_1 A = \begin{pmatrix} s_1 & * & * & \dots & * \\ 0 & s_2 & * & \dots & * \\ \vdots & 0 & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & * & \dots & * \end{pmatrix}$$

- Al passo k, procediamo allo stesso modo: supponendo che

$$Q_{k-1}Q_{k-2}\cdot\cdot\cdot\cdot\cdot Q_1A = \begin{pmatrix} s_1 & * & * & * & \dots & * \\ 0 & \ddots & * & * & \dots & * \\ 0 & \ddots & s_k & * & \dots & * \\ \vdots & 0 & 0 & \vdots & & \vdots \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & * & \dots & * \end{pmatrix}$$

  prendiamo

$$Q_2 = \left(\begin{array}{c|c} I_{k-1} & 0 \\ \hline 0 & H_k \end{array}\right)$$

  con $H_K A_{k:end,k} = (s_k, 0, ..., 0)^T$.

- Alla fine otteniamo

$$Q_n\cdot\cdot\cdot\cdot\cdot Q_1A = \left(\begin{array}{ccc} s_1 & * & * \\ & \ddots & * \\ & & s_n \\ \hline & 0 & \end{array}\right) = R.$$

  Quindi $Q_nQ_{n-1}\dots Q_1A = R \Rightarrow A = Q_1Q_2\dots Q_nR$

□

L'algoritmo come nella dimostrazione impiega $O(m^2n)$ operazioni a passo, date dalla moltiplicazione di una matrice $n \times n$ e una $m \times n$, dunque un totale di $O(m^2n^2)$ operazioni. Per la forma speciale delle Q_i però, possiamo ridurre le operazioni della moltiplicazione a $O(mn)$: infatti se H è una matrice di Householder, e M una matrice qualsiasi, $HM = M - (2u)(u^TM)$: l'operazione più costosa è quindi u^TM, che è un'operazione matrice-vettore è impiega $O(mn)$ operazioni.

6.2.2 Algoritmo QR per minimi quadrati

Se abbiamo calcolato la fattorizzazione QR di A, il problema dei minimi quadrati $\min \|Ax - b\|$ si può risolvere facilmente:

$$\|Ax - b\| = \|QRx - b\| = \|Q^T(QRx - b)\| = \|Rx - Q^T b\| =$$
$$= \left\| \left(\begin{array}{c} R_0 \\ \hline 0 \end{array}\right) x - \left(\begin{array}{c} Q_0^T \\ \hline Q_1^T \end{array}\right) b \right\| = \left\| \left(\begin{array}{c} R_0 x - Q_0^T b \\ \hline Q_1^T b \end{array}\right) \right\|$$

La parte sotto non contiene x, quindi non ci possiamo fare niente. La parte sopra è ora un sistema lineare quadrato $R_0 x = Q_0^T b$, che si può risolvere per sostituzione all'indietro in $O(n^2)$ operazioni, perché R_0 è triangolare superiore.

Lemma 6.2.2. *R_0 è invertibile se e solo se A ha rango massimo.*

Dimostrazione. A ha rango massimo $\iff$ $A^T A$ è invertibile $\iff$ $R^T Q^T Q R$ è invertibile $\iff$ $(R_0^T | 0) \left(\frac{R_0}{0}\right) = R_0^T R_0$ è invertibile $\iff$ R ha rango massimo, cioè è invertibile. □

Algoritmo completo

- Calcolare $A = QR$ in modo thin: $2mn^2$ operazioni.
- Calcolare $c = Q_0^T b$: $2mn$ operazioni.
- Risolvere $R_0 x = c$: n^2 operazioni.

Il costo dominante perciò è $2mn^2$, lo stesso delle equazioni normali. Il metodo QR ha però il vantaggio di essere numericamente più stabile, come vedremo.

6.3 SVD per minimi quadrati

Vediamo ora come possiamo usare la decomposizione ai valori singolari per risolvere il problema dei minimi quadrati. Come sempre assumiamo $m \geq n$,

quindi $A = U\Sigma V^T = U_0\Sigma_0 V^T$. Possiamo scrivere

$$\|Ax-b\| = \|U\Sigma V^T x - b\| = \|(U^T(U\Sigma(V^T x) - b)\| = \|\Sigma y - U^T b\|$$
$$= \left\| \left(\begin{array}{c} \sigma_1 y_1 \\ \vdots \\ \sigma_n y_n \\ \hline 0 \\ \vdots \\ 0 \end{array}\right) - \left(\begin{array}{c} u_1^T b \\ \vdots \\ u_n^T b \\ \hline u_{n+1}^T b \\ \vdots \\ u_m^T b \end{array}\right) \right\|$$

Per ottenere il minimo possibile, basta imporre $\sigma_i y_i = u_i^T b_i$, cioè $y = \dfrac{u_i^T b_i}{\sigma_i}$. Perciò

$$x = Vy = v_1 \frac{u_1^T b}{\sigma_1} + \cdots + \frac{u_n^T b}{\sigma_n} = V\Sigma^{-1}U_0^T$$

Con un semplice conto, si verifica che, in effetti, $V\Sigma^{-1}U_0^T = A^+$, la pseudoinversa di A.

6.3.1 Minimi quadrati per matrici non a rango massimo

A differenza degli altri metodi, con la SVD possiamo risolvere il problema anche se A non ha rango massimo: se il rango di A è $r < n$, la soluzione è

$$v_1 \frac{u_1^T b}{\sigma_1} + \cdots + \frac{u_r^T b}{\sigma_r}.$$

Se A non ha rango massimo, possiamo definire la sua pseaudoinversa in termini della sua SVD, come

$$A^+ = V \begin{pmatrix} \frac{1}{\sigma_1} & & & & & \\ & \ddots & & & & \\ & & \frac{1}{\sigma_r} & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{pmatrix} U^T$$

Dal punto di vista teorico questa soluzione è ottimale, ma dal punto di vista numerico c'è un grosso problema: poiché i valori singolari sono al denominatore in questo caso, i valori singolari più piccoli contano di più. In pratica, quando calcoliamo al computer la SVD di A, i valori singolari $\sigma_{r+1}, \dots \sigma_n$ di A in aritmetica floating point non saranno mai esattamente 0, e questo può portare a grossi errori nel calcolo della pseudoinversa e della soluzione dei minimi quadrati.

Una soluzione può essere allora troncare la SVD quando si ha che $\sigma_{i+1} << \sigma_i$. Questo approccio ha problemi quando tutti i valori singolari sono equamente distribuiti.

Possiamo perciò semplicemente fissare un certo k a priori, e prendere

$$x_{reg} = v_1 \frac{u_1^T b}{\sigma_1} + \dots + \frac{u_k^T b}{\sigma_k}.$$

6.3.2 Regolarizzazione di Tikhonov

Una soluzione più sofisticata, e molto usata anche in vari campi, è quella della regolarizzazione di Tikhonov, o ridge regression: al problema dei minimi quadrati aggiungiamo un termine di regolarizzazione in norma 2,

$$min_x \|Ax - b\|^2 + \alpha \|x\|^2.$$

dove $\alpha > 0$ è un parametro che scegliamo noi, e regola quanto è forte la regolarizzazione. Così facendo ci assicuriamo che la norma di x non sia troppo grande.

Per risolvere questo problema, notiamo che vale l'uguaglianza

$$\|Ax - b\| + \alpha \|x\|^2 = \left\| \begin{pmatrix} A \\ \alpha I_n \end{pmatrix} x - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|^2 = \left\| \begin{pmatrix} Ax - b \\ \alpha x \end{pmatrix} \right\|^2$$

Ora la matrice $\left(\begin{smallmatrix} A \\ \alpha I \end{smallmatrix}\right)$ è sicuramente di rango massimo, quindi non ci dobbiamo preoccupare di valori singolari nulli. La soluzione sta nel prendere la pseudoinversa di questa matrice, che è

$$(A^T A + \alpha^2 I)^{-1} (A^T | \alpha I),$$

e dunque $x_{ridge} = (A^T A + \alpha^2 I)^{-1} (A^T | \alpha I) b = (A^T A + \alpha^2 I)^{-1} A^T b$ (soluzione mediante le equazioni normali).

In termini della SVD,

$$x_{ridge} = + = V \begin{pmatrix} \frac{\sigma_1}{\sigma_1^2+\alpha^2} & & \\ & \ddots & \\ & & \frac{\sigma_n}{\sigma_n^2+\alpha^2} \end{pmatrix} U^T b$$

Osserviamo che

$$\frac{\sigma_1}{\sigma_1^2+\alpha^2} \approx \begin{cases} \frac{1}{\sigma_i} & \text{se } \sigma_i >> \alpha \\ 0 & \text{se } \sigma_i << \alpha \end{cases}.$$

Quindi, la regolarizzazione di Tikhonov approssima la SVD troncata per i valori singolari più piccoli di α.

Capitolo 7

Analisi degli errori

Non ne ho proprio voglia scusate

Capitolo 8

Sistemi Lineari

8.1 Metodi diretti

L'ultimo problema che vogliamo affrontare è la risoluzione di sistemi lineari. Prendiamo cioè una matrice $A \in \mathbb{R}^{m \times m}$ invertibile, e un vettore $b \in \mathbb{R}^m$, e risolviamo $Ax = b$.

Partiremo dagli approcci diretti basati su fattorizzazione: questi metodi si basano sullo scomporre A come un prodotto di matrici $A = B_1 B_2 ... B_n$, tali che risolvere $B_i c = d$ sia più facile di risolvere il sistema lineare con A. A questo punto, risolvere $Ax = b$ si traduce nel risolvere

$$\begin{cases} B_n x = y_{n-1} \\ B_{n-1} y_{n-1} = y_{n-2} \\ \vdots \\ B_1 y_1 = b \end{cases}$$

Analizzeremo poi il caso di matrici sono molto grandi e sparse. I sistemi lineari si ritrovano in moltissime aree nella matematica, nell'ottimizzazione, e nella ricerca operative. Un esempio visto in questo corso sono le equazioni di Karush-Kuhn-Tucker, che si traducono in molti casi nel risolvere un sistema lineare. In MATLAB, le matrici sparse sono memorizzate come insieme di triple (i, j, c), a significare che $A_{ij} = c$. Vengono memorizzate perciò solo le entrate non vuote della matrice. Il vantaggio di questo metodo di memorizzare si traduce in un vantaggio in spazio e in tempo: ad esempio, per fare il prodotto scalare di due vettori sparsi ci basta eseguire un numero di

moltiplicazioni pari al numero di entrate in cui entrambi i vettori sono non vuoti. Vediamo ora il primo metodo per risolvere i sistemi lineari.

8.1.1 Fattorizzazione LU

Un primo modo per risolvere il sistema $Ax = b$ è adoperando la fattorizzazione $A = LU$, dove L è una matrice triangolare inferiore, e U è una matrice triangolare superiore. La creazione della fattorizzazione LU segue lo stesso processo dell'eliminazione gaussiana:

- Sia a_{11} il primo elemento di A, che supponiamo essere diverso da 0, e siano $A_1, \ldots, A_m$ le righe di A. Ad ogni passo aggiorniamo l'i-esima riga nel seguente modo:

  $$A_i = A_i - \frac{a_{i1}}{a_{11}} A_1$$

  In questo modo, la prima riga di A diventa nulla in tutte le entrate diverse da a_{11}.

  $$A_1 = A, \quad A_2 = \begin{pmatrix} a_{11} & * & \ldots & * \\ 0 & * & \ldots & * \\ \vdots & \vdots & & \vdots \\ 0 & * & \ldots & * \end{pmatrix}$$

  Questa operazione corrisponde a una moltiplicazione per una matrice triangolare inferiore che ha solo 1 sulla diagonale:

  $$A_2 = \begin{pmatrix} 1 & 0 & \ldots & 0 \\ -a_{21}/a_{11} & 1 & 0 & 0 \\ \vdots & 0 & \ddots & 0 \\ -a_{n1}/a_{11} & 0 & \ldots & 1 \end{pmatrix} A_1$$

- Al passo k, aggiorniamo $U_{k+1:end,k:end} = L^T_{k+1:end,k} U_{k,k+1:end}$

Il costo totale è $2(m-1)^2 + 2(m-2)^2 + \cdots + 2 \cdot 2^2 + 2 \cdot 1^2 = \frac{2}{3}m^3$. La fattorizzazione LU costa la metà della fattorizzazione QR. Una volta ottenuta $A = LU$, è facile risolvere il sistema lineare:

$$Ax = b \equiv LUx = b \equiv \begin{cases} Ux = y \\ Ly = b \end{cases}$$

Entrambi i sistemi lineari possono essere risolti con m^2 operazioni tramite la sostituzione in avanti/all'indietro.

Il problema è che nell'eseguire questo processo, potremmo incappare in una divisione per 0, o per un numero molto vicino a 0. Perciò così com'è la fattorizzazione LU non è stabile.

Esempio.

$$A = \begin{pmatrix} 10^{-30} & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & -1 \end{pmatrix}$$

Applicando un passo della fattorizzazione LU, otteniamo

$$A_2 = \begin{pmatrix} 10^{-30} & 1 & 1 \\ 0 & 1 - 10^{-30} & 1 - 10^{-30} \\ 0 & 1 - 10^{-30} & -1 - 10^{-30} \end{pmatrix}$$

Ora il blocco 2×2 in basso a destra è quasi singolare, e in aritmetica FP A_2 è indistinguibile da una matrice singolare, nonostante A fosse molto chiaramente non singolare.

Dal punto di vista della stabilità all'indietro, $\tilde{L}\tilde{U} = A + E$, con $\|E\| = O(u)\|L\| \cdot \|U\|$. Ma $\|L\|$ e $\|U\|$ possono essere molto più grandi di $\|A\|$. Dunque l'algoritmo non è stabile.

Pivoting

Un modo per riparare questo problema è il pivoting: Ad ogni passo, portiamo l'elemento più grande della riga sulla diagonale. In questo modo, garantiamo che $\|L\| \leq 1$. In alcuni casi, $\|U\|/\|A\|$ può crescere in modo esponenziale, ma estremamente improbabile, e l'algoritmo LU con pivoting è generalmente considerato stabile.

8.1.2 Fattorizzazione LDL^T

Prendiamo adesso una matrice $M \in \mathbb{R}^{m \times m}$ simmetrica, cioè $M = M^T$. Possiamo sfruttare la simmetria di M per facillitare la fattorizzazione. Per mantenere M simmetrica, se moltiplichiamo a sinistra per una matrice L_i, dobbiamo moltiplicare a destra per la sua traposta L_i^T. Vediamo passo per passo il processo.

- Al primo passo, se moltiplichiamo per la matrice L_1 che corrisponde al primo passo dell'eliminazione gaussiana, moltiplichiamo a destra per L_1^T, ottenendo
$$L_1 M L_1^T = \begin{pmatrix} * & 0 & \dots & 0 \\ 0 & * & \dots & * \\ \vdots & \vdots & & \vdots \\ 0 & * & \dots & * \end{pmatrix}$$

- Dopo un alatro passo,
$$L_2 L_1 M L_1^T L_2^T = \begin{pmatrix} * & 0 & 0 & \dots & 0 \\ 0 & * & 0 & \dots & 0 \\ 0 & 0 & * & \dots & * \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & * & \dots & * \end{pmatrix}$$

- Così via, finche al passo $m-1$ non otteniamo
$$L_{n-1} \cdot \dots \cdot L_1 M L_1^T \cdot \dots \cdot L_{n-1}^T = D,$$
dove D è una matrice diagonale. Se chiamiamo $L_{n-1} \cdot \dots \cdot L_1 = L^{-1}$, otteniamo $M = LDL^T$, con L una matrice triangolare inferiore con tutti 1 sulla diagonale.

Per simmetria, quindi, non ci serve calcolare davvero L^T, che è semplicemente la trasposta di L. Bastano quindi la metà delle operazioni rispetto alla fattorizzazione LU, ovvero $\frac{1}{3}m^3$ operazioni.

Anche in questo caso per garantire la stabilità serve il pivoting. Per mantenere la simmetria, ogni volta che scambiamo due righe dobbiamo anche scambiare due colonne. A volte questo non basta: ad esempio se
$$M = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 3 \\ 2 & 3 & 0 \end{pmatrix},$$

non c'è alcuno scambio riga e colonna per cui $M_{11} \neq 0$. Possiamo dunque scambiare blocchi da 2 in questo modo:

$$\begin{aligned} M &\leftarrow M - M_{:,1:2} M_{1:2,1:2} M_{1:2,:} \\ &= \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 3 \\ 2 & 3 & 0 \end{pmatrix} - \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}^{-1} \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 3 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 3 \\ 2 & 3 & 0 \end{pmatrix} - \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 3 & 2 \end{pmatrix} \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 3 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 3 \\ 2 & 3 & 0 \end{pmatrix} - \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 3 \\ 2 & 3 & 12 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -12 \end{pmatrix} \end{aligned}$$

8.1.3 Fattorizzazione di Cholesky

Supponiamo adesso che M sia simmetrica e definita positiva

Teorema 8.1.1. *Se M è definita positiva, allora non ci sono pivot nulli, e quindi la fattorizzazione è stabile*

Dimostrazione. Segue dal fatto che

- M definita positiva $\implies$ $M_{kk} > 0$ per ogni k: infatti $e_k^T M e_k = M_{kk} > 0$.
- Se M è definita positiva e B è invertibile, BMB^T è definita positiva: infatti per ogni vettore $v \neq 0$, $v^T BMB^T v = (B^T v)^T M (B^T v)$. Chiamiamo $w = B^T v$, $w \neq 0$, poiché B^T è invertibile. Allora, $v^T BMB^T v = w^T M w > 0$.

L_i è invertibile per ogni i, poiché è triangolare e con elementi diagonali non nulli (sono tutti uguali a 1), e dunque ad ogni passo la matrice resta definita positiva. □

In questo caso, possiamo riscrivere la fattorizzazione LDL^T in modo equivalente: Abbiamo appena dimostrato che $D = \operatorname{diag}(d_1, \ldots, d_m)$, con ciascun $d_m > 0$, e dunque $D = \sqrt{D}\sqrt{D}$, con $\sqrt{D} = \operatorname{diag}(\sqrt{d_1}, \ldots, \sqrt{d_m})$. Chiamiamo $R^T = L\sqrt{D}$, e otteniamo

$$M = LDL^T = L\sqrt{D}\sqrt{D}^T L^T = R^T R$$

Questa fattorizzazione è detta *fattorizzazione di Cholesky*. La fattorizzazione è stabile: $\|R\| = \|M\|^{1/2}$. Infatti, M definita positiva $\implies$ esiste A tale che $R = A^T A$. Se prendiamo la fattorizzazione QR di A, otteniamo $A = Q_0 R_0$, e $M = A^T A = R_0^T R_0$, con l'i-esimo valore singolare di R_0, $\sigma_i(R_0) = \sigma_i(A^T A)^{1/2} = \sigma_i(M)^{1/2}$.

8.1.4 Recap degli algoritmi diretti per sistemi lineari

Facciamo un recap degli algoritmi diretti per sistemi lineari visti fin'ora:

- Se A è simmetrica e definita positiva $\implies$ Fattorizzazione di Cholesky, impiega $\frac{1}{3}m^3$ operazioni, ed è stabile.
- Se A è simmetrica e non definita positiva $\implies$ Fattorizzazione LDL^T, impiega $\frac{1}{3}m^3$ operazioni, può richiedere pivoting su righe e colonne per mantenere stabilità.
- Se A non è simmetrica $\implies$ Fattorizzazione LU, impiega $\frac{2}{3}m^3$ operazioni, e pivoting sulle righe per mantenere stabilità.

8.1.5 Limitazioni degli algoritmi basati su fattorizzazione

Se la matrice A è sparsa e di grosse dimensioni, la principale limitazioni degli algoritmi diretti basati sulle fattorizzazioni, è il cosiddetto fill-in: Dopo un po' di iterazioni dell'algoritmo, la matrice risultante ha molti più elementi non nulli della matrice di partenza. Questo porta a problemi di memoria, poiché la memoria del PC potrebbe semplicemente non essere abbastanza grande per contenere una matrice molto grande con tanti elementi non nulli. Inoltre, poiché il costo di operazioni su matrici sparse dipende dal numero di elementi diversi da 0, il fill-in porta anche a problemi di prestazioni. Introduciamo perciò un'altra classe di algoritmi, che invece di calcolare x in modo esatto (sempre in relazione alla aritmetica di macchina che è persua natura non esatta), producono ad ogni iterazione una approssimazione sempre migliore della soluzione.

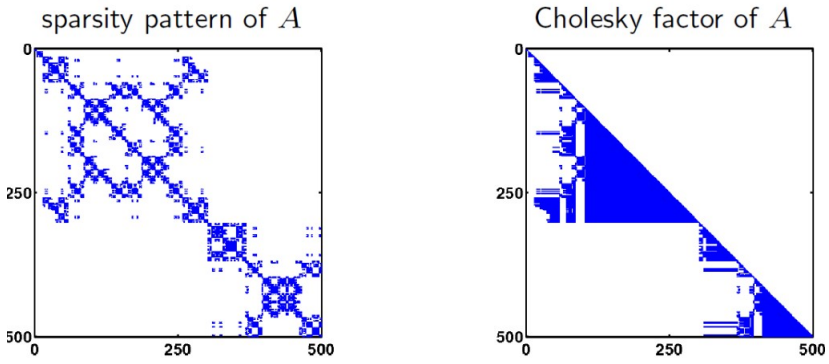

Figura 8.1: Fenomeno del fill-in nella fattorizzazione di Cholesky di una matrice simmetrica.

8.2 Metodi iterativi

I metodi iterativi sono una classe di metodi per risolvere sistemi lineari ispirati alle tecniche di ottimizzazione.

Iniziamo dal caso di una matrice A definita positiva. Se definiamo il problema quadratico

$$\min_{x \in \mathbb{R}^m} f(x) = \frac{1}{2} x^T A x - b^T x,$$

l'ottimo del problema è quando

$$\nabla f(x) = Ax - b = 0,$$

cioè quando $Ax = b$. La soluzione del sistema lineare corrisponde quindi al minimo della funzione quadratica. Inoltre il *residuo* $Ax - b$ corrisponde al gradiente della funzione f, e ci dice quanto x è lontano dall'essere una soluzione.

8.2.1 Gradiente Coniugato

Iniziamo da un caso molto semplice

$$\min \frac{1}{2}\|x - b\|^2 = \min_x \frac{1}{2}(x_1^2 + \cdots + x_m^2) - (b_1 x_1 + \cdots + b_m x_m) + \cancel{cost}$$

Partiamo da una soluzione iniziale $x^{(0)} = 0$, e minimizziamo una coordinata alla volta. Chiamiamo $d^{(i)}$ la direzione di ricerca ad ogni passo: $d^{(i)} \propto x^i - x^{i-1}$, normalizzata in modo che $\|d^i\| = 1$.

Seguendo questo procedimento,

$$\begin{aligned}
x^{(1)} &= (b_1, 0, \ldots, 0)^T, \quad d^{(1)} = (1, 0, \ldots, 0)^T \\
x^{(2)} &= (b_1, b_2, 0, \ldots, 0)^T, \quad d^{(1)} = (0, 1, 0, \ldots, 0)^T \\
x^{(3)} &= (b_1, b_2, , b_3, 0, \ldots, 0)^T, \quad d^{(1)} = (0, 0, 1, 0 \ldots, 0)^T \\
&\vdots \\
x^{(n)} &= b, d^{(n)} = (0, \ldots, 0, 1)^T
\end{aligned}$$

Questo caso è molto semplice, e la soluzione era ovvia anche ad occhio, ma da questo esempio possiamo evidenziare due proprietà importanti:

- Le direzioni di ricerca sono tutte ortogonali: $d^{(i)T} d^{(j)} = 0$, s e$i \neq j$.
- Il procedimento finisce in un numero finito di passi e trova la soluzione esatta: infatti allo step i-esimo, $x^{(i)}$ è la approssimazione migliore appartenente a span$\{d^{(1)}, ..., d^{(i)}\}$, e quindi $x^{(n)}$ è l'approssimazione migliore in span$\{d^{(1)}, ..., d^{(n)}\} = \mathbb{R}^n$

Scegliere $d^{(i)} = e_i$ era in questo caso la scelta più naturale, ma avremmo potuto scegliere qualunque insieme di vettori ortogonali: Data la matrice ortogonale $U = (u_1|...|u_n)$, chiamiamo $y = U^T x = U^{-1} x$, e quindi $x = Uy$. Allora

$$\min_x \frac{1}{2} x^T x - b^T x = \min_y y^T U^T U y - b^T U y$$

Chiamiamo $g = U^T b$, e otteniamo

$$\min_y \frac{1}{2}(y_1^2 + \cdots + y_m^2) - (g_1 y_1 + \cdots + g_m y_m)$$

Nello "spazio di x" questo è equivalente a minimizzare x lungo le direzioni $u_i = U e_i$. Dopo k passi, x è la migliore approssimazione in span$\{u_1, ..., u_k\}$.

Passando al caso generale, per risolvere

$$\min_{x \in \mathbb{R}^m} \frac{1}{2} x^T A x - b^T x,$$

possiamo trovare la matrice di cambio di base R tale che $R^TR = A$, e cambiare base, ottenendo

$$\min_x \frac{1}{2}y^Ty - g^Ty$$

con $y = Rx$, $g = (R^T)^{-1}b$ Siccome non conosciamo R, e calcolarla sarebbe troppo costoso, in pratica minimizziamo x lungo le direzioni $(d_i)_{i=1}^m$, dove $u_i = Rd_i$, e le u_i sono le direzioni di ricerca ortogonali nello spazio y.
$u_i^Tu_j = 0 \implies p_i^TR^TRp_j = p_i^TAp_j = 0$, cioè le d_i sono A-*ortogonali*. Dopo k passaggi, $x^{(k)}$ è la migliore approssimazione in $\text{span}\{d_1, ..., d_k\}$. Perciò

$$x^{(k)} = \underset{\text{span}\{d_1,..,d_k\}}{\arg\min} \frac{1}{2}\|Rx - R^{-T}b\|^2 = \underset{\text{span}\{d_1,..,d_k\}}{\arg\min} \|R(x - R^{-1}R^{-T}b)\|^2 =$$
$$= \underset{\text{span}\{d_1,..,d_k\}}{\arg\min} \frac{1}{2}(x - x_*)^TR^TR(x - x_*) = \underset{\text{span}\{d_1,..,d_k\}}{\arg\min} \frac{1}{2}(x - x_*)^TA(x - x_*),$$

dove abbiamo chiamato $x_* = (R^{-1}R^{-T})b = (R^TR)^{-1}b = A^{-1}b$, cioè la soluzione del sistema lineare. La domanda che ci poniamo ora è come trovare le direzioni d_i. L'Algoritmo 3 ci mostra come scegliere le direzioni d_i in modo che siano A ortogonali. Per capire il perché di questa scelta dobbiamo introdurre gli spazi di Krylov.

Algoritmo 5 Metodo del gradiente coniugato

$\mathbf{x}_0 = \mathbf{0}, \mathbf{r}_0 = \mathbf{d}_0 = \mathbf{b}$;
for $j = 1{:}n$ **do**
  $\alpha_j = (\mathbf{r}_{j-1}^T\mathbf{r}_{j-1})/(\mathbf{d}_{j-1}^TA\mathbf{d}_{j-1})$; // exact LS (check!)
  $\mathbf{x}_j = \mathbf{x}_{j-1} + \alpha_j\mathbf{d}_{j-1}$;
  $\mathbf{r}_j = \mathbf{r}_{j-1} - \alpha_jA\mathbf{d}_{j-1}$; // residual update (check!)
  $\beta_j = (\mathbf{r}_j^T\mathbf{r}_j)/(\mathbf{r}_{j-1}^T\mathbf{r}_{j-1})$;
  $\mathbf{d}_j = \mathbf{r}_j + \beta_j\mathbf{d}_{j-1}$; // A-orthogonal (we'll see why)
end

8.2.2 Spazi di Krylov

Definizione. Data $A \in \mathbb{R}^{m\times m}$, $b \in \mathbb{R}^m$, $n \in \mathbb{N}$, lo spazio di Krylov associato a A e b è

$$\begin{aligned} K_n(A,b) &= \text{span}\{b, Ab, A^2b, ..., A^{n-1}b\} \\ &= \{v : v = p(A)b, \text{ dove } p(t) = c_0 + c_1 t + \cdots + c_{n-1}t^{n-1}\} \end{aligned}$$

Se A e b sono ovvi dal contesto, chiamiamo $K_n = K_n(A,b)$.

Proprietà utile: Se $v \in K_n(A,b)$, allora $Av \in K_n(A,b)$: infatti

$$Av = A(c_0 + c_1A + \cdots + c_{n-1}A^{n-1})b = (c_0A + c_1A^2 + \cdots + c_{n-1}A^n)b$$

Osservazione 10. I c_i sono unici se gli A^ib sono linearmente indipendenti.

Se gli A^ib sono linearmente indipendenti, e $v \longleftrightarrow (c_0, c_1, \ldots, c_{n-1},$ con $c_{n-1} \neq 0$, allora $v \in K_n$, ma $v \notin K_{n-1}$, e $Av \in K_{n+1}$, ma $Av \notin K_n$.

Sicuramente, esiste un numero naturale n_* che è il minimo tale che $A^{n_*} \in K_{n_*}$, e vale

$$dim(K_1) = 1 < dim(K_2) = 2 < \cdots < dim(K_{n_*}) = n_* = dim(K_{n_*+1}) = dim(K_{n_*+2}) = \ldots$$

Il seguente teorema mette in relazione il metodo del gradiente coniugato e gli spazi di Krylov:

Teorema 8.2.1. *Ad ogni passo n del gradiente coniugato, vale*

$$\begin{aligned} x_1, x_2, \ldots, x_n &\in K_n(A,b) \\ d_0, d_1, \ldots, d_{n-1} &\in K_n(A,b) \\ r_0, r_1, \ldots, r_{n-1} &\in K_n(A,b) \end{aligned}$$

Dimostrazione. $d_0 = r_0 = b \in K_1(A,b) \subseteq K_n(A,b)$, $x_0 = 0 \in K_n(A,b)$.

Per induzione su j:

$r_j = r_{j-1} - \alpha_j A d_{j-1}$.

Per ipotesi induttiva, $r_{j-1} \in K_j(A,b)$ e $d_{j-1} \in K_j(A,b)$, perciò $r_j \in K_{j+1}(A,b)$

$x_j = x_{j-1} + \alpha_j d_j$. Per ipotesi induttiva, $x_{j-1} \in K_{j-1}(A,b)$ e $d_{j-1} \in K_j(A,b)$, dunque $x_j \in K_j(A,b)$.

$d_j = r_j + \beta d_{j-1}$. Per ipotesi induttiva $d_{j-1} \in K_{j-1}(A,b)$, e abbiamo appena dimostrato che $r_j \in K_{j+1}(A,b)$, dunque $d_j \in K_{j+1}(A,b)$. □

Dalla dimostrazione si vede anche che per ogni j, $d_{j-1}, r_{j-1} \in K_j \setminus K_{j-1}$, e $x_j \in K_j \setminus K_{j-1}$, quindi tutte e tre le successioni di vettori del Teorema 8.2.1 sono basi degli spazi $K_n(A,b)$ per ogni n.

Come ultimo passo prima di dimostrare l'ottimalità del gradiente coniugato, mostriamo l'ortogonalità dei vettori:

Teorema 8.2.2. *Se $i \neq 0$, $r_i^T r_j = 0$, cioè gli r_i sono ortogonali, e $d_i^T A d_j = 0$, cioè i d_i sono $A-$ortogonali.*

Dimostrazione. Sia $i < j$, dimostriamo per induzione su j. Per $j = 0$ è banale. Sia $j \geq 1$. Mostriamo che vale $r_i^T r_j = 0$.

$r_i^T r_k = r_i^T r_{j-1} - \alpha_j r_i^T A d_{j-1}$. Quindi, se $i < j-1$, per ipotesi induttiva vale che $r_i^T r_{j-1} = 0$, e anche $r_i^T r_j = 0$ di conseguenza.

Se $i = j-1$,

$$r_{j-1}^T r_j = r_{j-1}^T (r_{j-1} - \alpha A d_{j-1}) = r_{j-1}^T r_{j-1} - \alpha_j r_{j-1}^T A d_{j-1}.$$

Per come l'abbiamo definita (Algoritmo 5), $\alpha_j = \frac{r_{j-1}^T r_{j-1}}{d_{j-1}^T A d_{j-1}}$, quindi

$$\alpha_j r_{j-1}^T A d_{j-1} = \frac{r_{j-1}^T r_{j-1}}{d_{j-1}^T A d_{j-1}} r_{j-1}^T A d_{j-1}$$

Ci resta quindi da dimostrare che $r_{j-1}^T A d_{j-1} = d_{j-1}^T A d_{j-1}$. Ma $d_{j-1} = r_{j-1} - \beta_{j-1} d_{j-2}$, quindi

$$d_{j.1}^T A d_{j-1} = r_{j-1}^T A d_{j-1} - \cancel{\beta_{j-1} \mathbf{d}_{j-2} A d_{j-1}} = r_{j-1}^T A d_{j-1}$$

siccome $d_{j-2} A d_{j-1} = 0$ per ipotesi induttiva. La dimostrazione che $d_i^T A d_j = 0$ usa le stesse tecniche, ed è analoga. □

Teorema 8.2.3. $x_j = \arg\min_{x \in K_j(A,b)} \|x - x_*\|_A^2$, *dove* $\|y\|_A = y^T A y$.

Dimostrazione. Sia $x_j + y$ un altro vettore in $K_j(A,b)$, quindi anche $y \in K_j(A,b)$.

$$\begin{aligned} \|x_* - (x_j + y)\|_A^2 &= (x_* - x_j - y)^T A (x_* - x_j - y) \\ &= (x_* - x_j)^T A (x_* - x_j) + y^T A y - 2 y^T A (x_* - x_j) \end{aligned}$$

Ora, $A(x_*) - A x_j = b - A x_j = r_j$, e poiché $y \in K_j(A,b)$, $y \in \operatorname{span}\{r_1, \ldots, r_{j-1}\}$, quindi $y^T A (x_* - x_j) = y^T r_j = 0$, per l'ortogonalità degli r_i. Dunque

$$\|x_* - (x_j + y)\|_A^2 = (x_* - x_j)^T A (x_* - x_j) + y^T A y > \|x_j - x_*\|_A^2$$

□

Ciò prova che il metodo del gradiente coniugato è il metodo migliore basato su combinazioni lineari e applicazioni di A, poiché ad ogni passo otteniamo la migliore approssimazione possibile.

8.2.3 Velocità di convergenza del gradiente coniugato

Abbiamo visto che in teoria, se $A \in \mathbb{R}^{m\times m}$, il gradiente coniugato termina in al più m passi: infatti, se $n > m$, sicuramente $K_n(A, b) = \mathbb{R}^m$, e quindi $r_n = 0$. In pratica usiamo il metodo come un qualsiasi metodo iterativo. Un primo risultato sulla convergenza è il seguente.

Teorema 8.2.4. *Per ogni $n \in \mathbb{N}$*

$$\frac{\|x_n - x_*\|_A}{\|x_0 - x_*\|_A} \leq \left(\frac{\sqrt{\lambda_{max}} - \sqrt{\lambda_{min}}}{\sqrt{\lambda_{max}} + \sqrt{\lambda_{min}}}\right)^n,$$

dove l_{max} e λ_{min} sono l'autovalore più grande / più piccolo di A.

Questo limite ci dice che la velocità di convergenza è lineare, ma non è ottimale. Le performance del gradiente coniugato dipendono infatti dalla distribuzione degli autovalori di A. In particolare, questo metodo è molto efficiente se gli autovalori sono divisi in *cluster* molto vicini:

Teorema 8.2.5.

$$\frac{\|x_n - x_*\|_A}{\|x_0 - x_*\|_A} \leq \min_{\substack{deg(q)\leq n \\ q(0)=1}} \max_{\lambda\in eig(A)} |q(\lambda)|$$

Cioè l'errore al passo n è limitato dal massimo errore che commettiamo se vogliamo interpolare le coppie $(0, \lambda_i)$, dove i λ_i sono gli autovalori di A, con un polinomio di grado n

8.2.4 Algoritmo di Arnoldi

Il metodo del gradiente coniugato funziona solo per una matrice A simmetrica e definita positiva. Per un sistema lineare generico, possiamo usare un metodo simile, sempre basato sugli spazi di Krylov.

Come nel gradiente coniugato, vorremmo calcolare una base do $K_n(A, b)$ $\{nb, Ab, \ldots, A^{n-1}b\}$ è una base, ma potrebbe essere molto mal-condizionata. Introduciamo un metodo per trovare sempre una base do $K_n(A, b)$

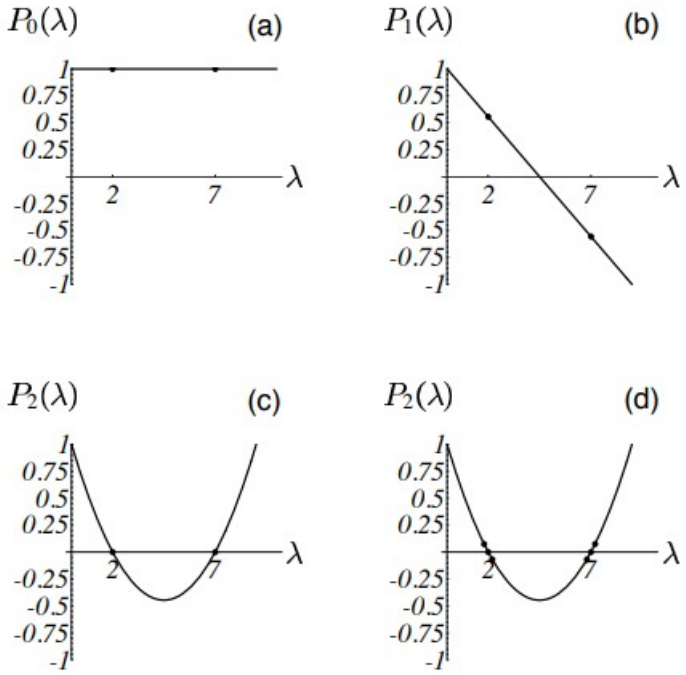

Figura 8.2: La convergenza del gradiente coniugato dipende da quanto vicino possiamo essere a 0 in corrispondenza di ogni autovalore di A: in questo caso gli autovalori sono 2 e 7, quindi un polinomio di grado 2 è sufficiente.

Ortogonalizzazione di Gram-Schmidt

Se $\{q_1, \ldots, q_j\}$ è una base ortonormale di $K_j(A, b)$, possiamo trovare facilmente q_{j+1} tale che $\{q_1 \ldots, q_j, q_{j+1}\}$ è una base di $K_{j+1}(A, b)$.

- Prendiamo $w = Aq_j$.
- Rendiamo w ortogonale a q_1: $w \leftarrow w - q_1\beta_1$, con $\beta_1 = q_1^T w$. In questo modo il nuovo w' è ortogonale a q_1 (semplice verifica)
- Ripetiamo il processo per $i = 2, \ldots, j$: $w \leftarrow w - q_i\beta_i$, con $\beta_i = q_i^T w$.
- Normalizziamo: $q_{j+1} \leftarrow w/\|w\|$.

Algoritmo di Arnoldi

Trova da 0 una base di $K_n(A, b)$, semplicemente applicando il processo appena visto n volte:

Prendiamo $q_1 = b/\|b\|$, e per $j = 2, \ldots, n$, prendiamo q_j risultato del processo appena visto. Il costo computazionale di queste operazioni è $O(mn^2)$, dove m è la dimensione di A, e n è la dimensione dello spazio di Krylov (cioè il numero di iterazioni del metodo).

Algoritmo di Arnolidi come fattorizzazione

Dal metodo appena illustrato, si vede che

$$Aq_j - \beta_1 q_1 - \cdots - \beta_j q_j = \beta_{j+1} q_{j+1}$$

con $\beta_{j+1} = 1/\|w\|$. In forma matriciale questo si riscrive come

$$Aq_j = (q_1|q_2|\ldots|q_{j+1}|q_{j+2}|\ldots|q_n) \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{j+1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Mettendo insieme tutti i q_j per $j = 1, \dots n$ a destra, si ottiene

$$A(q_1|q_2|\dots|q_n) = (q_1|q_2|\dots|q_{n+1}) \begin{pmatrix} \beta_1 & \beta_1 & \beta_1 & \dots & \beta_1 \\ \beta_2 & \beta_2 & \beta_2 & \dots & \beta_2 \\ 0 & \beta_3 & \beta_3 & \dots & \beta_3 \\ \vdots & 0 & \beta_4 & \dots & \beta_4 \\ \vdots & \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \dots & 0 & \beta_{n+1} \end{pmatrix}$$

Cioè $AQ_n = Q_{n+1}H_n$ Le dimensioni delle matrici sono giuste: $A \in \mathbb{R}^{m\times m}$, $Q_n \in \mathbb{R}^{m\times n}$, $Q_{n+1} \in \mathbb{R}^{m\times n+1}$, $H_n \in \mathbb{R}^{n+1\times n}$, dunque il risultato è una matrice $\in \mathbb{R}^{m\times n}$

Abbiamo quindi una base ben condizionata di $K_n(A, b)$, e possiamo calcolare l'approssimazione della soluzione di $Ax = b$ come

$$\min_{x\in K_n(A,b)} \|Ax - b\|_2 = Q_n y$$

per qualche $y \in \mathbb{R}^n$ (tutti i vettori di K_n si possono scrivere come immagine di Q_n, poiché questo corrisponde ad una combinazione lineare dei q_j, che sono una base di K_n). Troviamo quindi tale y:

$$\min_{x\in K_n(A,b)} \|Ax - b\|_2 = \min_{y\in\mathbb{R}^n} \|AQ_n y - b\|_2 = \min_{y\in\mathbb{R}^n} \|Q_{n+1}H_n y - b\|_2$$

Sia ora $Q_m = (Q_{n+1}/\tilde{Q})$, la matrice $m\times m$ ortonormale ottenuta completando

Q_{n+1} ad una base ortonormale di $\mathbb{R}^m$. Q_m è ortonormale, quindi

$$\begin{aligned}
\min_{y\in\mathbb{R}^n}\|Q_{n+1}H_n y - b\|_2 &= \min_{y\in\mathbb{R}^n}\|Q_m^T(Q_{n+1}H_n y - b)\|_2 \\
&= \min_y \left\| \left(\begin{array}{c} I_{n+1} \\ \hline 0 \end{array}\right) H_n y - \underbrace{\begin{pmatrix} \|b\| \\ 0 \\ \vdots \\ 0 \end{pmatrix}}_{q_1=b/\|b\|, q_j^T b=0 \text{ se } j\neq 1} \right\| \\
&= \min_y \left\| \left(\begin{array}{c} H_n y \\ \hline 0 \end{array}\right) - \begin{pmatrix} \|b\| \\ 0 \\ \vdots \\ 0 \end{pmatrix} \right\| \\
&= \min_y \|H_n y - \|b\| e_1\|
\end{aligned}$$

Il problema

$$\min_y \|H_n y - \|b\| e_1\| \equiv H_n y = \|b\| e_1 \tag{8.1}$$

può essere risolto in $O(n^3)$ tramite fattorizzazione QR. In genere $n << m$, quindi il costo è minore rispetto a quello dell'algoritmo di Arnoldi, $O(mn^2)$.

8.2.5 GMRES

Il metodo *Generalized Minimal Residual* è una versione ottimizzata del procedimento appena visto. Alcuni miglioramenti che possiamo fare sono:

1. Il residuo $\rho = \|Ax - b\|$ è disponibile senza il prodotto matrice vettore: infatti $\rho = \|H_n y - \|b\| e_1\|$

2. H_n ha una forma particolare: è "quasi triangolare superiore", cioè gli elementi sotto la prima sottodiagonale sono tutti nulli (H è una matrice di Hessemberg). Nella fattorizzazione QR della matrice, necessaria per risolvere il sistema (8.1), possiamo trarre vantaggio dalla struttura per far calare il costo da $O(n^3)$ a $O(n^2)$.

3. In realtà possiamo calcolare la fattorizzazione QR in modo incrementale, passando da $Q_n R_n = H_n \to Q_{n+1} R_{n+1} = H_{n+1}$. In questo modo non dobbiamo necessariamente fissare il valore di n all'inizio

Per quanto riguarda la convergenza del metodo abbiamo un risultato simile a quello del gradiente coniugato, che ci dice che più gli autovalori di A sono clusterizzati, migliori sono le performance dell'algoritmo. Notiamo che, in questo caso, gli autovalori saranno in generale numeri complessi, quindi intenderemo come valore assoluto l'usiale norma su $\mathbb{C}$, $|a + ib| = a^2 + b^2$.

Teorema 8.2.6.

$$\|Ax^{(n)} - b\| \leq \kappa(V) \min_{\substack{deg(q) \leq n \\ q(0)=1}} \max_{\lambda \in eig(A)} |q(\lambda)| \cdot \|b\|$$

dove V è la matrice che diagonalizza A, $A = VAV^{-1}$, e $\kappa(V)$ è il numero di condizionamento di V.

Ulteriori considerazioni

- La convergenza monotona decrescente è garantita, poiché $\rho_n = \min_{x \in K_n} \|Ax - b\|$
- Il costo di un'iterazione cresce con n, quindi dopo un numero di passi (ad esempio 50), è utile il restart cioè dato $x^{(n)}$, ricominciare risolvendo il problema $Az = b - Ax^{(n)}$
- Se nella procedura incontriamo una divisione per 0, siamo in realtà contenti perchè ciò vuol dire che $\min \|H_n y - \|b\| e_1\| = 0$, e quindi abbiamo risolto il problema ("lucky breakdown").

MINRES

Se A è simmetrica (e non definita positva, in caso contrario useremo il gradiente coniugato), anche H_n è simmetrica, quindi è tridiagonale. Perciò il costo dell'iterazione di Arnoldi per calcolare Q_n e H_n diventa $O(mn)$, diminuendo di un fattore lineare. GMRES per matrici simmetriche è detto MINRES.