Appunti dalle lezioni di Calcolo Numerico

A.A. 2013/2014

Indice

Elenco delle figure

Capitolo 2

Riassunto di Algebra Lineare

Un numero (scalare) intero, reale o complesso sarà in genere indicato da una lettera minuscola dell'alfabeto greco, per esempio:

$$\alpha \in \mathbb{I} \qquad \beta \in \mathbb{R} \qquad \alpha \in \mathbb{C}.$$

Un vettore, definito come una n-upla ordinata di numeri (e.g. reali), sarà indicato con una lettera minuscola dell'alfabeto inglese, usando le seguenti notazioni del tutto equivalenti:

$$x \in \mathbb{R}^n \qquad x = \begin{bmatrix} x_1 \\ . \\ . \\ . \\ x_n \end{bmatrix} \qquad x = \{x_i\}.$$

Il numero $x_i \in \mathbb{R}$ è chiamato la componente i-esima del vettore x.

Una matrice, definita da una tabella di numeri (e.g. reali) caratterizzata da n righe e m colonne, sarà indicata da una lettera maiuscola dell'alfabeto inglese, usando le seguenti notazioni del tutto equivalenti:

$$A_{[n\times m]} \qquad A = \begin{bmatrix} a_{11} & \dots & a_{1m} \\ . & \dots & \\ . & \dots & \\ . & \dots & \\ a_{n1} & \dots & a_{nm} \end{bmatrix} \qquad A = \{a_{ij}\}.$$

Il numero $a_{ij} \in \mathbb{R}$ è chiamato l'elemento ij-esimo della matrice A. In realtà noi utilizzeremo quasi esclusivamente matrici quadrate, per cui in genere si avrà $m = n$. Si noti che un vettore può essere considerato come una matrice avente una colonna e cioè $m = 1$. Per convenzione (nostra) un vettore è quindi sempre un vettore colonna. Tuttavia, quando si parlerà di matrici o di vettori si farà sempre riferimento alle

notazioni precedentemente definite. Tutte le proprietà che seguono sono ovviamente valide per vettori e matrici.

Somma di matrici. La matrice somma di due matrici della stessa dimensione è definita coma la matrice che si ottiene sommando ordinatamente le componenti, o in formula, date $A, B, C \in \mathbb{R}^{n\times m}$:

$$C = A \pm B := \{a_{ij} \pm b_{ij}\}.$$

Prodotto di una matrice per uno scalare. Il prodotto tra uno scalare e una matrice o un vettore è definito per componenti:

$$\alpha A = \{\alpha a_{ij}\}.$$

Matrice trasposta. La matrice $A^T \in \mathbb{R}^{n\times n}$ si chiama la matrice trasposta di A e si ottiene cambiando le righe con le colonne:

$$A = \{a_{ij}\} \qquad A^T = \{a_{ji}\}.$$

Per una matrice in campo complesso $A \in \mathbb{C}^{n\times n}$, l'operazione di trasposizione deve generalmente essere accompagnata dall'operazione di coniugazione complessa:

$$A = \{a_{ij}\} \qquad A^* = \overline{A^T} = \{\overline{a_{ji}}\}.$$

Matrice nulla. La matrice nulla è quella matrice che ha tutte le componenti uguali a zero, ed è l'elemento neutro della somma:

$$A = 0 \Longleftrightarrow a_{ij} = 0 \qquad i, j = 1, \ldots, n.$$

Matrice identità. La mtrice identità I e quella matrice quadrata che ha le componenti diagonali uguali a uno e tutte le altre nulle:

$$I_{[n\times n]} \qquad I := \begin{bmatrix} 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ . & . & \ldots & & . \\ 0 & 0 & \ldots & 1 & 0 \\ 0 & 0 & \ldots & 0 & 1 \end{bmatrix}.$$

Matrice positiva o non negativa. Una matrice è detta positiva (o non negativa) [1] se tutti i suoi elementi sono positivi o nulli con almeno un elemento positivo:

$$A > 0 \Rightarrow a_{ij} \geq 0.$$

[1] La proprietà di una matrice di essere *positiva* non va confusa con la proprietà di essere *definita positiva* che verrà definita più avanti.

Prodotto scalare tra due vettori. Si definisce prodotto scalare tra due vettori la somma dei prodotti delle componenti omonime:

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

Altre notazioni che useremo per il prodotto scalare sono:

$$\langle x, y \rangle \text{ oppure } x \cdot y.$$

In modo più formale, dato uno spazio vettoriale $V \subset \mathbb{C}^n$ (o $\mathbb{R}^n$), il prodotto scalare (o interno) tra due elementi $x, y \in V$, indicato con $\langle x, y \rangle$, è la mappa:

$$\langle \cdot, \cdot \rangle : V \times V \to \mathbb{C} \text{ (o } \mathbb{R})$$

che soddisfa alle seguenti proprietà definenti:

- (Simmetria Hermitiana) $\langle x, y \rangle = \overline{\langle y, x \rangle}$ dove $\overline{(\cdot)}$ indica l'operazione di coniugazione complessa;
- (linearità) $\langle \alpha x + \beta y, z \rangle = \alpha \langle x, z \rangle + \beta \langle y, z \rangle$;
- (positività) $\langle x, x \rangle > 0$ per ogni $x \in \mathbb{C}^n$ (o $\mathbb{R}^n$) e $\langle x, x \rangle = 0$ solo per $x = 0$.

Prodotto tra matrici. Il prodotto tra due matrici, detto anche prodotto righe-colonne, è dato da quella matrice che ha per componenti i prodotti scalari tra le righe della prima matrice e le colonne della seconda matrice pensate come vettore:

$$A_{[n \times p]} = \{a_{ij}\} \qquad B_{[p \times m]} = \{b_{ij}\} \qquad C_{[n \times m]} = \{c_{ij}\}$$

$$C = AB \qquad c_{ij} := \sum_{k=1,p} a_{ik} b_{kj}, \qquad i = 1, \ldots, n \qquad j = 1, \ldots, m.$$

Il prodotto matrice vettore è un caso particolare del prodotto tra matrice, considerando il vettore come una matrice $n \times 1$. E' facile quindi verificare che il prodotto scalare gode della seguente proprietà[2]:

$$\langle x, Ay \rangle = \left\langle A^T x, y \right\rangle \qquad \langle Ax, y \rangle = \left\langle x, A^T y \right\rangle.$$

[2]Ovviamente il prodotto scalare è commutativo e cioè: $\langle x, Ay \rangle = \langle Ay, x \rangle$ ovvero $x^T Ay = (Ay)^T x$

Determinante di una matrice. Data una matrice quadrata A, il suo determinante determinante, det A, è definito come lo scalare dato dalla somma di tutti prodotti ottenuti prendendo come fattore un elemento di ciascuna riga ed uno di ciascuna colonna:

$$\det A := \sum \pm a_{1,i_1} a_{2,i_2} \cdots a_{n,i_n},$$

dove $i_1, i_2, \ldots, i_n$ sono permutazioni distinte dei primi n numeri interi e il segno è dato dall'ordine della permutazione.

Inversa di una matrice quadrata. Data una matrice quadrata $A_{[n\times n]}$ se det $A \neq 0$, si definisce matrice inversa A^{-1}, se esiste, la matrice tale che:

$$A^{-1}A = AA^{-1} = I.$$

Matrice singolare. Una matrice è singolare se la sua inversa non esiste. Una matrice singolare ha determinante nullo e viceversa.

Matrice unitaria o ortogonale. Una matrice si dice unitaria o ortogonale se:

$$U^T = U^{-1}.$$

Proprietà delle operazioni tra matrici quadrate.

1. $AB \neq BA$ (Le matrici per le quali la proprietà commutativa vale sono dette *commutative*.)
2. $A + B = B + A$
3. $(A + B) + C = A + (B + C)$
4. $(AB)C = A(BC)$
5. $A(B + C) = AB + BC; (A + B)C = AC + BC$
6. $(AB)^T = B^T A^T$
7. $(AB)^{-1} = B^{-1}A^{-1}$
8. $(A^T)^{-1} = (A^{-1})^T$.

Matrice simmetrica. Una matrice si dice simmetrica se è uguale alla sua trasposta:

$$A = A^T;$$

si dice antisimmetrica se è opposta alla sua trasposta:

$$A = -A^T.$$

Ogni matrice può essere decomposta secondo la somma della sua parte simmetrica e della sua parte antisimmetrica:

$$A = \frac{1}{2}(A + A^T) + \frac{1}{2}(A - A^T).$$

Matrice definita positiva. Una matrice $A_{[n\times n]}$ si dice definita positiva (semi definita positiva) se:

$$x^T Ax = \langle x, Ax\rangle > 0 \qquad (\langle x, Ax\rangle \geq 0).$$

Matrice di tipo M o M-matrice. Una matrice è detta di tipo M se gode delle seguenti proprietà:

1. è non singolare;
2. tutti gli elementi della diagonale sono positivi;
3. gli elementi extra diagonali sono negativi o nulli.

Una M-matrice ha la proprietà che la sua inversa è positiva.

2.0.4 Spazi vettoriali, vettori linearmente dipendenti, basi

Spazio vettoriale. Uno *spazio vettoriale* V (sul campo scalare $\mathbb{R}$) è un insieme di vettori dove sono definite l'operazione di addizione tra due vettori e di moltiplicazione tra uno scalare (reale) e un vettore. Tali operazioni devono soddisfare le seguenti proprietà definenti[3] per ogni $x, y \in V$ e per ogni $\alpha, \alpha_1, \alpha_2 \in \mathbb{R}$:

1. $x + y = y + x$;
2. $x + (y + z) = (x + y) + z$;

[3]Si noti che tali proprietà agiscono in modo tale che la maggior parte delle operazioni elementari che generalmente facciamo sul campo degli scalari possano essere fatte anche su tale spazio

3. esiste un unico elemento nullo dell'addizione (il vettore "zero") tale che $x+0 = 0+x = x$;
4. per ogni x esiste un unico vettore $-x$ tale che $x + (-x) = 0$;
5. $1x = x$;
6. $(\alpha_1\alpha_2)x = \alpha_1(\alpha_2 x)$;
7. $\alpha(x+y) = \alpha x + \alpha y$;
8. $(\alpha_1 + \alpha_2)x = \alpha_1 x + \alpha_2 x$.

Per esempio, sono spazi vettoriali:

- $\mathbb{R}^k$, l'insieme di tutti i vettori a k componenti con le classiche operazioni di somma e prodotto per uno scalare;
- $\mathbb{R}^\infty$ l'insieme dei vettori a infinite componenti (di nuovo con le stesse operazioni di prima);
- lo spazio delle matrici $m \times n$; in questo caso i vettori sono matrici e le operazioni sono quelle definite nei paragrafi precedenti[4];
- lo spazio delle funzioni continute $f(x)$ definite nell'intervallo $0 \leq x \leq 1$ (ad esempio appartengono a tale spazio $f(x) = x^2$ e $g(x) = \sin(x)$ per le quali si ha che $(f+g)(x) = x^2 + sin(x)$ e ogni multiplo tipo $3x^2$ oppure $-\sin(x)$ sono ancora nello spazio). I vettori in questo caso sono funzioni quindi con "dimensione" infinita.

Sottospazio vettoriale. Un sottospazio $S \subset V$ dello spazio vettoriale V è un sottoinsieme di V che soddisfa alle relazioni:

1. per ogni $x, y \in S$ la loro somma $z = x + y$ è ancora un elemento di S;
2. il prodotto di ogni $x \in S$ per uno scalare $\alpha \in \mathbb{R}$ è un elemento di S: $z = \alpha x$, $z \in S$.

Si dice anche che il sottospazio S è un sottoinsieme di V "chiuso" rispetto alle operazioni di addizione e moltiplicazione per uno scalare. Un esempio di un sottospazio vettoriale è un piano, che è affine allo spazio vettoriale $\mathbb{R}^2$ se pensato isolato ma che è contenuto in $\mathbb{R}^3$.

[4] questo spazio è in qualche modo simile a $\mathbb{R}^{mn}$

Indipendenza lineare. Si dice che k vettori v_k sono *linearmente indipendenti* se tutte le loro combinazioni lineari (eccetto quella triviale a coefficienti nulli) sono non-nulle:

$$\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_k v_k \neq 0 \qquad \text{escludendo il caso} \qquad \alpha_1 = \alpha_2 = \cdots = \alpha_k = 0.$$

In caso contrario, i vettori si dicono linearmente dipendenti.

Si ha la seguente proprietà: un insieme di k vettori apartenenti a $\mathbb{R}^m$ devono necessariamente essere linearmente dipendenti se $k > m$.

Span. Se uno spazio vettoriale V è formato da tutte le combinazioni lineari dei vettori $v_1, v_2, \ldots, v_n$, si dice che questi vettori "generano" lo spazio V e si scrive:

$$V = \text{span}\{v_1, v_2, \ldots, v_n\}.$$

In altre parole ogni altro vettore di V può essere scritto come combinazione lineare dei vettori generatori:

$$w \in V \Rightarrow w = \alpha_1 v_1 + \ldots \alpha_n v_n = \sum_{i=1}^{n} \alpha_1 v_1.$$

Base. Una *base* dello spazio vettoriale V è l'insieme (minimale) dei vettori che:

1. sono linearmente indipendenti;

2. generano lo spazio V.

Dimensione di uno spazio vettoriale. Le basi di uno spazio vettoriale sono infinite. Ciascuna base contiene lo stesso numero di vettori. Tale numero è chiamato *dimensione* dello spazio V ($\dim V$.

Ad esempio, una base dello spazio tri-dimensionale $\mathbb{R}^3$ è costituita dall'insieme dei vettori coordinate e_1, e_2, e_3, dove $e_1 = (1,0,0)^T$, $e_2 = (0,1,0)^T$, $e_3 = (0,0,1)^T$, con ovvia estensione alla generica dimensione n e si scrive:

$$n = \dim(V).$$

Ogni insieme di vettori di V linearmente dipendenti può essere esteso ad una base (se necessario aggiungendo opportuni vettori). Viceversa, ogni insieme di vettori generatori di V può essere ridotto ad una base (se necessario eliminando dei vettori).

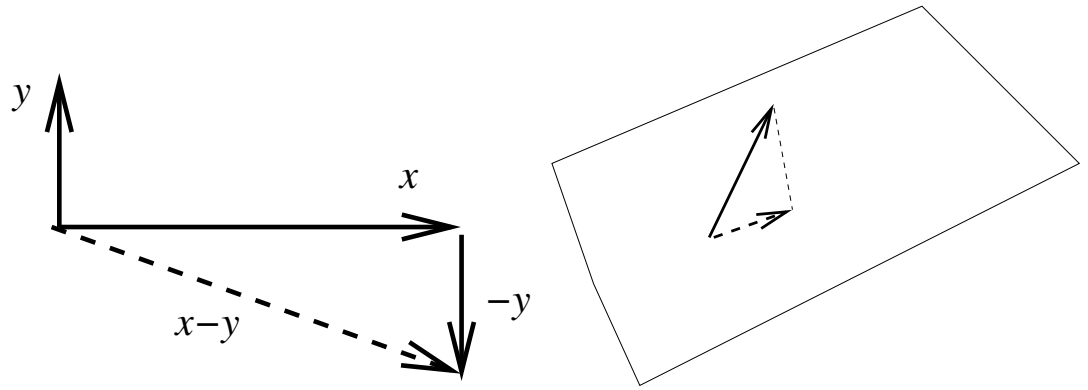

Figura 2.1: **A sinistra**: vettori ortogonali: x e y sono ortogonali. Applicando il Teorema di Pitagora alla coppia di vettori x e $-y$ che formano i cateti di un triangolo rettangolo e scrivendo l'uguaglianza (2.1) si ricava immediatamente che deve essere valida la (2.2). **A destra**: proiezione ortogonale del sottospazio V (formato da un solo vettore) nel sottospazio W (formato dal piano).

Ortogonalità tra vettori e sottospazi

Introduciamo il concetto di lunghezza di un vettore x che indichiamo con $\|x\|$ (si veda più avanti il paragrafo sulle norme di vettori). Visivamente, in $\mathbb{R}^2$, scomponendo il vettore nelle sue componenti lungo gli assi principali, $x = (x_1, x_2)$, si può definire la lunghezza usando il teorema di Pitagora, da cui si ha immediatamente:

$$\|x\|^2 = x_1^2 + x_2^2,$$

e per estensione diretta a $\mathbb{R}^n$:

$$\|x\|^2 = x_1^2 + x_2^2 + \ldots + x_n^2.$$

Vettori ortogonali. Sempre in $\mathbb{R}^2$, è intuitivo dire che due vettori x e y sono ortogonali se formano un tra lor un angolo rettangolo, ovvero usando il teorema di Pitagora, se (si veda Figura 2.1):

$$\|x\|^2 + \|y\|^2 = \|x - y\|^2 . \tag{2.1}$$

Dalla precedente, scritta per componenti, si verifica immediatamente che dovranno essere nulli i prodotti incrociati (somma dei doppi prodotti), da cui si ottiene la definizione generale di ortogonalità tra vettori di $\mathbb{R}^n$:

$$x^T y = \langle x, y \rangle = 0. \tag{2.2}$$

Tale quantità, il prodotto scalare, è anche chiamato prodotto interno. Si dimostra immediatamente che se n vettori sono mutuamente ortogonali, allora essi sono linearmente indipendenti.

Spazi ortogonali. due sottospazi V e W di $\mathbb{R}^n$ sono ortogonali se ogni vettore $v \in V$ è ortogonale a ogni vettore $w \in W$.

Complemento ortogonale. Dato un sottospazio $V \subset \mathbb{R}^n$, lo spazio di tutti i vettori ortogonali a tutti i vettori di V si dice complemento ortogonale di V e si denota con $V^\perp$.

Spazio nullo e spazio immagine di una matrice. L'insieme di tutti i vettori $x \in \mathbb{R}^n$ tali che $Ax = 0$ si chiama *spazio nullo* o *kernel* della matrice A e si indica con $\ker A$.

L'insieme di tutti i vettori $x \in \mathbb{R}^n$ tali che $Ax \neq 0$ si chiama *immagine* (o *Range*) della matrice A e si indica con $\mathrm{Ran}(A)$.

Si ha immediatamente che:

$$\dim(\ker(A)) + \dim(\mathrm{Ran}(A)) = n,$$

e che il rango di A è uguale alla dimensione del $\mathrm{Ran}(A)$.

2.0.5 Operatori di proiezione.

Un operatore di proiezione, o proiettore, P, è una trasformazione lineare $P : \mathbb{R}^n \to \mathbb{R}^n$ idempotente, cioè tale che:

$$P^2 = P. \tag{2.3}$$

Un tipico esempio di una operazione di proiezione riguarda l'operazione di proiezione di un vettore tridimensionale su un piano: dato il vettore $x = (x_1, x_2, x_3)^T$, il proiettore che lo trasforma nel vettore $\tilde{x} = (x_1, x_2, 0)^T$ è rappresentato dalla matrice:

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

Evidentemente P è un proiettore. Infatti:

$$P \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ 0 \end{pmatrix}$$

da cui immediatamente si ha $P^2 x = P(Px) = P\tilde{x} = \tilde{x}$.

Alcune proprietà di un proiettore P:

1. se P è un proiettore anche $(I - P)$ lo è;

2. $\ker(P) = \mathrm{Ran}(I - P)$;

3. $\ker(P) \cap \mathrm{Ran}(P) = 0$ (il vettore nullo).

4. $\mathbb{R}^n = \ker(P) \oplus \mathrm{Ran}(I - P)$;

Infatti, se P soddisfa (2.3), si ha immediatamente: $(I - P)(I - P)x = (I + P^2 - 2P)x = (I - P)x$, da cui discende la prima prorietà; la seconda è un'immediata conseguenza della prima. La terza proprietà si dimostra immediatamente dalla idempotenza di P: se $x \in \mathrm{Ran}(P)$ allora $Px = x$; se $x \in \ker(P)$ deve essere $Px = 0$, talchè $x = Px = 0$. Ovviamente ogni elemento di $\mathbb{R}^n$ può essere scritto come $x = Px + (I - P)x$, da cui si ricava immediatamente la quarta proprietà.

Siano dati due sottospazi $M \subset \mathbb{R}^n$ e $S \subset \mathbb{R}^n$, e sia $L = S^\perp$. Allora il proiettore P su M ortogonale al sottospazio L è la trasformazione lineare $u = Px$ $(\forall x \in \mathbb{R}^n)$ che soddisfa:

$$u \in M \tag{2.4}$$

$$x - u \perp L \tag{2.5}$$

Queste relazioni determinano rispettivamente il numero di gradi di libertà $m = \mathrm{Rank}(P) = \dim(M)$ e le m condizioni che definiscono Px. Si può dimostrare che queste relazioni definiscono compiutamente il proiettore, ovvero che, dati due sottospazi M e L di dimensione m è sempre possibile definire un proiettore su M ortogonale a L tramite le condizioni (2.4) e (2.5).

Lemma 2.0.1. *Dati due sottospazi di $\mathbb{R}^n$, M e L, aventi la stessa dimensione m, le seguenti due condizioni sono matematicamente equivalenti:*

1. *Non esistono vettori di M ortogonali a L;*

2. *per ogni $x \in \mathbb{R}^n$ esiste un unico vettore u che soddisfa* (2.4) *e* (2.5).

Dimostrazione. La prima condizione è equivalente alla condizione:

$$M \cap L^\perp = \{0\}.$$

Siccome L e $L \perp$ hanno dimensioni n e $n - m$, questo è equivalente a dire:

$$\mathbb{R}^n = M \oplus L^\perp.$$

Quindi, per ogni x esiste un unico vettore u appartenente a M tale che:

$$x = u + w \qquad\qquad w = x - u \in L^\perp.$$

□

Rappresentazione matriciale. Per poter esprimere le condizioni per definire un proiettore, Dobbiamo trovare due basi, una per M, che chiameremo $V = [v_1, \ldots, v_m]$, e una per L, che chiameremo $W = [w_1, \ldots, w_m]$. Tali basi sono bi-ortogonali se $\langle v_i, w_j \rangle = \delta_{ij}$, che scritta in forma matriciale diventa $W^T V = I$. Chiamiamo quindi Vy la rappresentazione di Px nella base V. Il vincolo di ortogonalità $x - Px \perp L$ si può tradurre nelle condizioni:

$$\langle (x - Vy), w_j \rangle = 0 \qquad \text{per } j = 1, \ldots, m,$$

ovvero:

$$W^T(x - Vy) = 0.$$

Se le due basi sono ortogonali si ha immediatamente che $y = W^T x$, e quindi $Px = VW^T x$, da cui ricaviamo la rappresentazione matriciale dell'operatore di proiezione:

$$P = VW^T.$$

Nel caso in cui le due basi non sono bi-ortogonali, l'espressione diventa invece:

$$P = V(W^T V)^{-1} W^T.$$

Sotto l'assunzione (ovvia) che non esistono vettori di M ortogonali a L, la matrice $W^T V$ di dimensione $m \times m$ è nonsingolare.

Proiezione ortogonale. Se V e W sono sottospazi di $\mathbb{R}^n$, una qualsiasi delle seguenti proprietà li caratterizza come ortogonali:

1. $W = V^\perp$ (W consiste di tutti i vettori ortogonali a V);
2. $V = W^\perp$ (V consiste di tutti i vettori ortogonali a W);
3. V e W sono ortogonali e $\dim V + \dim W = n$.

La proiezione ortogonale di un vettore $x \in V_1$ lungo la direzione del vettore $y \in V_2$ è data dal vettore:

$$y = \langle x, y \rangle \, y = yy^T v = Pv$$

In generale, un proiettore P si dice ortogonale se i due spazi M e L sono uguali, e quindi $\ker(P) = \mathrm{Ran}(P)^\perp$. Un proiettore che non è ortogonale si dice obliquo. Alla luce delle condizioni studiate in precedenza, un proiettore è caratterizzato per ogni $x \in \mathbb{R}^n$ dalle seguenti proprietà:

$$\begin{aligned} &Px \in M \\ &x - Px = (I - P)x \perp M \end{aligned}$$

ovvero

$$\begin{aligned} &Px \in M \\ &\langle (I-P)x, y\rangle = 0 \qquad\qquad \forall y \in M \end{aligned}$$

Si ha immediatamente il seguente:

Proposizione 2.0.2. *Un proiettore è ortogonale se e solo se è simmetrico (hermitiano).*

Dimostrazione. La trasformazione P^T, definita dall'aggiunto di P:

$$\left\langle P^T x, y\right\rangle = \langle x, Py\rangle \qquad \forall x, y$$

è anch'essa un proiettore. Infatti:

$$\left\langle (P^T)^2 x, y\right\rangle = \left\langle P^T x, Py\right\rangle = \left\langle x, P^2 y\right\rangle = \langle x, Py\rangle = \left\langle P^T x, y\right\rangle .$$

Di conseguenza,

$$\begin{aligned} \ker(P^T) &= \mathrm{Ran}(P)^\perp \\ \ker(P) &= \mathrm{Ran}(P^T)^\perp \end{aligned}$$

Per definizione, un proiettore è ortogonale se $\ker(P) = \mathrm{Ran}(P)^\perp$, talchè, se $P = P^T$ allora P è ortogonale. D'altro canto, se P è ortogonale, dalle precedenti si ha che $\ker(P) = \ker(P^T)$ e $\mathrm{Ran}(P) = \mathrm{Ran}(P^T)$. Siccome P è un proiettore, ed è quindi univocamente determinato dai suoi spazi nulli e dalla sua immagine, si deduce che $P = P^T$. □

Proprietà di ottimalità di un proiettore ortogonale

Lemma 2.0.3. *Siano $M \subset \mathbb{R}^n$ e P on proiettore su M. Allora per ogni $x \in \mathbb{R}^n$ si ha:*

$$\|x - Px\| < \|x - y\| \, \forall y \in M.$$

Dimostrazione. Dalle condizioni di ortogonalità si ottiene:

$$\|x-y\|_2^2 = \|x - Px + (Px - y)\|_2^2 = \|x - Px\|_2^2 + \|Px - y\|_2^2 .$$

da cui si ha immediatamente $\|x - Px\|_2 \leq \|x - y\|_2$ con l'uguaglianza che si verifica per $y = Px$. □

Corollario 2.0.4. *Sia M un sottospazio e x un vettore di $\mathbb{R}^n$. Allora*

$$\min_{y \in M} \|x - y\|_2 = \|x - \tilde{y}\|_2$$

se e solo se le seguenti condizioni sono entrambe soddisfatte:

$$\tilde{y} \in M, \qquad\qquad x - \tilde{y} \perp M.$$

2.0.6 Autovalori ed autovettori

Uno scalare λ ed un vettore $u \neq 0$ si dicono rispettivamente autovalore ed autovettore di una matrice quadrata A se soddisfano alla seguente relazione:

$$Au = \lambda u.$$

Si ricava facilmente che

$$(A - \lambda I)u = 0 \qquad \Rightarrow \qquad P(\lambda) = \det\,(A - \lambda I) = 0.$$

Dalla prima delle precedenti si vede immediatamente che gli autovettori sono definiti a meno di una costante moltiplicativa. Dalla seconda invece si vede che gli autovalori sono le radici di un polinomio di grado n a coefficienti reali (se gli elementi di A sono reali). Da quest'ultima osservazione e usando le proprietà delle radici di un polinomio si ricavano le seguenti proprietà degli autovalori:

$$\lambda \in \mathbb{C} \qquad \sum_{i=1}^{n} \lambda_i = \sum_{i=1}^{n} a_{ii} = \mathrm{Tr}\ A \qquad \prod_{i=1}^{n} \lambda_i = \det\ A \qquad A^m u = \lambda^m u.$$

Se esiste $\lambda = 0$, allora la matrice è singolare ($\det A = 0$). Secondo una comune notazione, tutti gli autovalori di una matrice A si indicano con $\lambda(A)$. Si dice anche che $\lambda(A)$ è lo spettro di A. Molto spesso si ordinano gli autovalori in maniera tale che

$$|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_n|,$$

per cui in generale la notazione $\lambda_1(A)$ indica l'autovalore di A massimo in modulo, mentre $\lambda_n(A)$ indica l'autovalore minimo (in modulo). Il modulo di $\lambda_1(A)$ è anche detto raggio spettrale di A e si indica con $\rho(A) = |\lambda_1(A)|$.

Trasformazioni di similitudine. Si dice che una matrice B è ottenuta da A tramite una trasformazione di similitudine se esiste una matrice non singolare S tale che:

$$B = S^{-1}AS.$$

Si vede che B e A hanno gli stessi autovalori mentre gli autovettori sono scalati dalla matrice S. Infatti:

$$\det\,(B - \lambda I) = \det\,(S^{-1}AS - \lambda S^{-1}S) = \det\ S^{-1}\det\,(A - \lambda I)\det\ S = \det\,(A - \lambda I),$$

e quindi

$$\begin{array}{rcl} Bu &=& \lambda u \\ Av &=& \lambda v \end{array} \Rightarrow \begin{array}{rcl} S^{-1}ASu &=& \lambda u \\ ASu &=& \lambda Su \end{array} \Rightarrow v = Su.$$

E' facile verificare che

$$\lambda(A) = \lambda(A^T) \qquad \lambda(AB) = \lambda(A^{-1}ABA) = \lambda(BA).$$

E' facile anche verificare che se A è definita positiva, allora $\lambda_i > 0$ $i = 1, \ldots, n$.

Se si indica con D la matrice (diagonale) formata dagli elementi di A sulla diagonale e con tutti gli elementi extradiagonali nulli:

$$D = \{d_{ij}\} \qquad d_{ij} = \begin{cases} a_{ii}, & \text{se } i = j, \\ 0, & \text{se } i \neq j. \end{cases} ;$$

si ha allora:

$$\lambda(D^{-1}A) = (\text{ponendo } S = D^{-\frac{1}{2}}) = \lambda(D^{\frac{1}{2}}D^{-1}AD^{-\frac{1}{2}}) = \lambda(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}).$$

Inoltre, la matrice $B = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ è definita positiva se lo è A. Infatti:

$$\langle x, Bx\rangle = \left\langle x, D^{-\frac{1}{2}}AD^{-\frac{1}{2}}x\right\rangle = \left\langle D^{-\frac{1}{2}}x, AD^{-\frac{1}{2}}x\right\rangle > 0.$$

Proprietà delle matrici simmetriche e diagonalizzazione.

- Gli autovalori ed autovettori di una matrice A simmetrica sono tutti reali.
- Gli autovalori ed autovettori di una matrice A antisimmetrica sono immaginari.
- Se A è definita positiva, $\lambda_i > 0$ per ogni $i = 1, \ldots, n$.
- Una matrice A si dice diagonalizzabile se esiste una matrice U non singolare tale che

  $$\Lambda = U^{-1}AU$$

  è una matrice diagonale. In questo caso è facile vedere che $\lambda_{ii} = \lambda_i$ sono gli autovalori di A e le colonne di U sono gli autovettori.
- Se A è simmetrica e definita positiva, è diagonalizzabile e la matrice U è unitaria (o ortogonale) ($U^{-1} = U^T$). Una matrice unitaria ha le colonne ortogonali tra di loro, per cui

  $$\langle u_i, u_j\rangle = \begin{cases} \neq 0, & \text{se } i = j, \\ = 0, & \text{se } i \neq j. \end{cases} ;$$

  e poichè in questo caso gli u_i sono autovettori di A, e sono definiti a meno di una costante moltiplicativa, si ha:

  $$\langle u_i, u_j\rangle \begin{cases} = 1, & \text{se } i = j, \\ = 0, & \text{se } i \neq j. \end{cases} .$$

  Si può concludere gli autovettori di matrici diagonalizzabili formano una base eventualmente ortonormale per lo spazio vettoriale $\mathbb{R}^n$. Questo significa che tutti i vettori di $\mathbb{R}^n$ possono essere espressi come combinazione lineare degli autovettori di A.

2.0.7 Norme di vettori e di matrici

Norme di vettori. Si definisce norma di un vettore x uno scalare reale che soddisfa alle seguenti relazioni:

1. $\|x\| > 0$, $\|x\| = 0$ se e solo se $x = 0$;
2. dato $\alpha \in \mathbb{R}$, $\|\alpha x\| = |\alpha| \, \|x\|$;
3. $\|x + y\| \leq \|x\| + \|y\|$;

Un esempio di norma di vettori generica è dato dalla norma-p:

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

per $p = 2$ si ha la classica norma euclidea $\|x\|_2 = \sqrt{\langle x, x \rangle}$; per $p = \infty$ si ha la norma massima $\|x\|_\infty = \max_i |x_i|$, per $p = 1$ si ha la norma assoluta $\|x\|_1 = \sum_{i=1}^{n} |x_i|$. Per $p = 2$ vale la disuguaglianza di Schwarz:

$$\langle x, y \rangle \leq \|x\|_2 \, \|y\|_2 \, .$$

Un'altra norma molto usata è la norma "energia", definita come il prodotto scalare tra il vettore x e il vettore Ax, dove la matrice A è una matrice simmetrica e definita positiva (se non lo fosse la proprietà 1 sopra non sarebbe soddisfatta):

$$\|x\|_A = \sqrt{\langle x, Ax \rangle}.$$

Norme di matrici. Si definisce norma di una matrice A uno scalare reale che soddisfa alle seguenti relazioni:

1. $\|A\| > 0$, $\|A\| = 0$ se e solo se $A = 0$;
2. dato $\alpha \in \mathbb{R}$, $\|\alpha A\| = |\alpha| \, \|A\|$;
3. $\|A + B\| \leq \|A\| + \|B\|$;
4. $\|AB\| \leq \|A\| \, \|B\|$;

Esempi di norme di matrici:

- norma di Frobenius: $\|A\|_2 = \sqrt{\text{Tr } (A^T A)} = \sqrt{\sum_{ij} |a_{ij}|^2}$;
- norma di Hilbert o norma spettrale: $\|A\| = \sqrt{\rho A^T A} = \sqrt{\left| \lambda_1 (A^T A) \right|}$.

Norme compatibili o indotte. Si dice che la norma di matrice $\|A\|$ è compatibile (o indotta da) con la norma di vettore $\|x\|$ se:

$$\|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|}.$$

Dimostriamo che la norma spettrale di matrice è compatibile con (indotta da) la norma euclidea di vettore. Infatti, data una matrice non singolare $A \in \mathbb{R}^{n \times n}$, si costruisce la matrice simmetrica e definita positiva $H = A^T A$. Essendo H diagonalizzabile, un generico vettore $x \in \mathbb{R}^n$ può essere pensato come combinazione lineare degli autovettori u_i di H:

$$x = c_1 u_1 + c_2 u_2 + \ldots + c_n u_n.$$

Poichè gli u_i sono ortonormali, si ottiene facilmente:

$$\begin{aligned} \langle x, Hx \rangle = \quad & x^T H x = (c_1 u_1 + c_2 u_2 + \ldots + c_n u_n)^T H (c_1 u_1 + c_2 u_2 + \ldots + c_n u_n) \\ = \quad & \lambda_1 |c_1|^2 + \ldots + \lambda_n |c_n|^2 \\ \leq \quad & \lambda_1 (|c_1|^2 + \ldots + |c_n|^2) = \lambda_1 \|x\|_2^2 , \end{aligned}$$

e da questa:

$$\lambda_1(A^T A) \geq \frac{\langle Ax, Ax \rangle}{\|x\|_2^2} = \frac{\|Ax\|_2^2}{\|x\|_2^2}.$$

La dimostrazione si completa prendendo la radice quadrata della precedente espressione.

Sistemi Lineari. Data una matrice A, di dimensioni $n \times n$ e non singolare, e un vettore $b \in \mathbb{R}^n$, si cerca il vettore $x \in \mathbb{R}^n$ che soddisfa:

$$Ax = b.$$

La soluzione formale di tale sistema è data da:

$$x^* = A^{-1} b.$$

Non è ragionevole trovare tale vettore, o un'approssimazione di esso, utilizzando la formula precedente, essendo il calcolo dell'inversa A^{-1} molto oneroso[5]

[5]Il modo più semplice per calcolare l'inversa è quella di risolvere n sistemi lineari, con un costo computazionale molto elevato. Ci sono altri metodi per calcolare l'inversa, ma sempre con costo molto alto rispetto alla soluzione di un sistema lineare.

In queste note si farà riferimento esclusivamente a metodi iterativi per la soluzione di sistemi lineari. In tali metodi si cerca di definire una successione di iterate (vettori) x_k $k > 0$ in maniera tale che

$$\lim_{k\to\infty} x_k = x^*. \tag{2.6}$$

Uno schema iterativo verrà terminato in pratica molto prima che la condizione precedente sia verificata. In effetti, non conoscendo x^*, sarà impossibile calcolare tale limite. Di solito si definisce il residuo come:

$$r_k = b - Ax_k,$$

per cui la condizione di convergenza (2.6) si traduce immediatamente dicendo che il residuo deve tendere a zero. L'iterazione verrà quindi terminata non appena la norma (qualsiasi) del residuo non diventi minore di una soglia predeterminata, chiamata tolleranza. In molti casi è meglio sostituire questa condizione con una relativa; l'iterazione termina quando il residuo iniziale è diminuito di un fattore τ:

$$\frac{\|r_k\|}{\|r_0\|} < \tau.$$

Definendo il vettore errore come la differenza tra la soluzione approssimata e la soluzione vera $e_k = x_k - x^*$ si può ricavare una relazione tra residuo ed errore:

$$\frac{\|e_k\|}{\|e_0\|} \leq \kappa(A)\frac{\|r_k\|}{\|r_0\|}.$$

dove il numero $k(A) = \|A\|\,\|A^{-1}\|$ è chiamato il numero di condizionamento della matrice A. Infatti:

$$r_k = b - Ax_k = -Ae_k.$$

da cui, utilizzando norme matriciali compatibili:

$$\|e_k\| = \left\|A^{-1}Ae_k\right\| \leq \left\|A^{-1}\right\|\,\|Ae_k\| = \left\|A^{-1}\right\|\,\|r_k\|\,,$$

e:

$$\|r_0\| = \|Ae_0\| \leq \|A\|\,\|e_0\|\,,$$

quindi:

$$\frac{\|e_k\|}{\|e_0\|} \leq \left\|A^{-1}\right\|\,\|A\|\,\frac{\|r_k\|}{\|r_0\|} = \kappa(A)\frac{\|r_k\|}{\|r_0\|}. \tag{2.7}$$

La condizione di terminazione sul residuo relativo così definito è scomoda perchè dipende fortemente dalla soluzione iniziale x_0. Si preferisce quindi rapportare la norma del residuo corrente alla norma del termine noto b, e cioè utilizzare la condizione:

$$\frac{\|r_k\|}{\|b\|} < \tau.$$

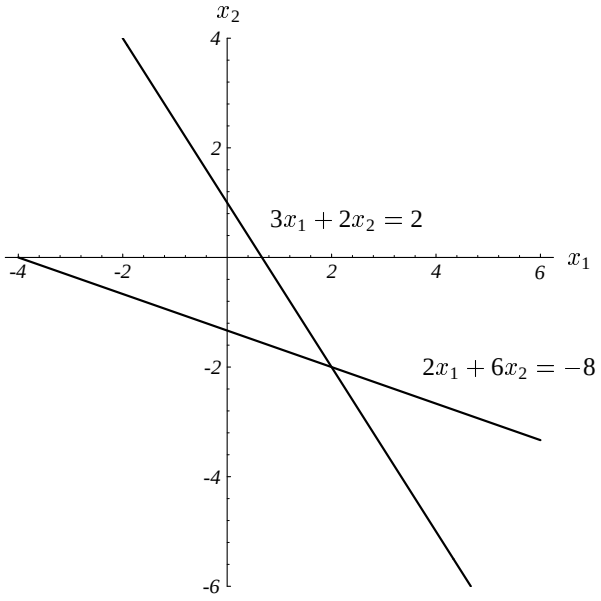

Figura 2.2: Interpretazione geometrica di un sistema lineare in $\mathbb{R}^2$. La soluzione del sistema è il punto di intersezione delle due rette e cioè il vettore $x^* = [2, -2]^T$.

Le due condizioni sono equivalenti qualora si scelga come soluzione iniziale $x_0 = 0$ (e quindi $r_0 = b$), una scelta molto diffusa.

Nel seguito faremo spesso riferimento al seguente esempio:

$$Ax = b \qquad \text{ove} \qquad A = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \qquad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad b = \begin{bmatrix} 2 \\ -8 \end{bmatrix} \tag{2.8}$$

che ha soluzione

$$x^* = \begin{bmatrix} 2 \\ -2 \end{bmatrix}.$$

In $\mathbb{R}^2$ l'interpretazione geometrica è facile: la soluzione del sistema è il punto di intersezione delle due rette le cui equazioni formano il sistema lineare, situazione che è disegnata in Fig. 2.2.

Capitolo 3

Metodi per la soluzione di sistemi lineari

Sia dato un sistema lineare:

$$Ax = b, \tag{3.1}$$

dove A è una matrice quadrata non singolare di dimensini $n \times n$ e x e b sono vettori in $\mathbb{R}^n$. Il problema che si vuole affrontare è quello di trovare il vettore x noti A e b.

3.1 Metodi lineari e stazionari

Si considerano in questo capitolo metodi iterativi della forma:

$$x_{k+1} = Ex_k + q, \tag{3.2}$$

dove $x_{k+1}, x_k, q \in \mathbb{R}^n$ e E è la matrice $(n \times n)$ di iterazione. In pratica, si vuole costruire una successione di vettori approssimanti $\{x_k\}$ che converga per $k \to \infty$ alla soluzione del problema (3.1):

$$x^* = A^{-1}b. \tag{3.3}$$

I metodi oggetto di studio in questo capitolo, rappresentati dalla (3.2), si dicono lineari e stazionari in quanto la matrice di iterazione E è stazionaria (costante) durante il processo iterativo e l'approssimazione x_k compare linearmente.

Per studiare la convergenza di questi schemi verifichiamo dapprima la loro consistenza. Sostituiamo quindi la (3.3) al posto di x_{k+1} e x_k in (3.2), ottenendo:

$$x^* = Ex^* + q \qquad\qquad q = (I - E)x^* = (I - E)A^{-1}b, \tag{3.4}$$

e osserviamo che tale espressione per q garantisce la consistenza forte dello schema (3.2).

Osservazione 3.1.1. Osserviamo che in un certo senso la matrice $I - E$ deve essere una approssimazione dell'inversa di A. Infatti, una matrice B si dice *inversa approssimata* di A se $\|I - BA\| < 1$.

Sotto la condizione $\|E\| < 1$, il lemma seguente ci garantisce che lo schema converge alla soluzione del sistema lineare.

Lemma 3.1.1 (Lemma di Banach)**.** *Sia E una matrice di dimensioni $n \times n$ con $\|E\| < 1$; allora $I - E$ è non singolare e*

$$\left\|(I - E)^{-1}\right\| \leq \frac{1}{1 - \|E\|}.$$

Dimostrazione. Per prima cosa dimostriamo che la serie di Neumann[1] $(I + E + E^2 + \ldots)$ converge a $(I - E)^{-1}$, cioè:

$$\sum_{i=0}^{\infty} E^i = (1 - E)^{-1}.$$

Infatti, definiamo la somma parziale S_k come:

$$S_k = \sum_{i=0}^{k} E^i.$$

Poichè $\|E^i\| \leq \|E\|^i$ e $\|E\| < 1$ per ipotesi, risulta immediatamente che, per $k < m$:

$$\|S_k - S_m\| \leq \sum_{i=k+1}^{m} \|E\|^i = (\text{serie geometrica}) = \|E\|^{k+1} \frac{1 - \|E\|^{m-k}}{1 - \|E\|},$$

che evidentemente converge a zero per $k, m \to \infty$. Quindi, per un noto teorema sulle successioni di Cauchy in spazi vettoriali, la successione S_k converge ad una matrice $n \times n$ che chiameremo S. Ora, siccome $S_{k+1} = ES_k + I$, al limite si ottiene $S = ES + I$, da cui immediatamente si ha $S = (I - E)^{-1}$.

La dimostrazione si completa osservando che

$$\left\|(I - E)^{-1}\right\| \leq \sum_{i=1}^{\infty} \|E\|^i = (1 - \|E\|)^{-1}.$$

□

Conseguenza diretta del lemma precedente è il seguente:

[1] La serie di Neumann è un'Estensione alle matrici della serie di Taylor $1 + x + x^2 + \ldots = 1/(1-x)$

Corollario 3.1.2. *Se* $\|E\| < 1$*, lo schema* (3.2) *converge a* $x = (I - E)^{-1}q$*.*

Quindi, se $\|E\| < 1$, e poiché per la consistenza $q = (I - E)A^{-1}b$, lo schema (3.2) converge a $x^* = A^{-1}b$, la soluzione vera del sistema lineare.

La condizione necessaria e sufficiente per la convergenza dello schema (3.2) è specificata in tutta generalità nel seguente:

Teorema 3.1.3. *Data una matrice reale* E *di dimensioni* $n \times n$*. L'iterazione* (3.2) *converge per ogni* $q \in \mathbb{R}^n$ *se e solo se* $\rho(E) = |\lambda_1(E)| < 1$*.*

Matrici la cui potenza k-esima tende ad annullarsi all'infinito si dicono *matrici convergenti*.

La condizione di convergenza specificata nel teorema precedente è indipendente dalle proprietà della matrice di iterazione, che non deve necessariamente essere ne' simmetrica ne' diagonalizzabile. Per brevità, e per capire fino in fondo il ruolo che autovalori e autovettori della matrice di iterazione giocano nella convergenza dello schema, dimostriamo il teorema 3.1.3 facendo delle assunzioni sulla matrice di iterazione E.

Procediamo nel modo classico andando ad analizzare la "stabilità" dello schema (3.2), essendo la consistenza garantita da (3.4). A tale scopo, definiamo il vettore errore $e_k = x^* - x_k$. La stabilità dello schema implica che l'errore deve rimanere limitato o meglio ancora tendere a zero all'aumentare di k, e cioè:

$$\lim_{k\to\infty} e_k = 0.$$

Per verificare questa condizione, dobbiamo ricavare una relazione tra gli errori a iterazioni successive. Dalla consistenza e linearità dello schema si ricava immediatamente che:

$$e_{k+1} = Ee_k = E^{k+1}e_0. \tag{3.5}$$

Prendendo la norma (ad es. la norma euclidea) dell'espressione precedente e usando la disuguaglianza di Schwartz, si ottiene immediatamente:

$$\|e_k\| \leq \|E\|^k \|e_0\|,$$

dove $\|E\|$ è la norma indotta da $\|e_0\|$. La convergenza del metodo è assicurata se $\|E\| < 1$, risultato uguale a quello ricavato con il lemma di Banach 3.1.1. Assumendo simmetrica la matrice di iterazione, e osservando che usando la norma euclidea dei vettori e la norma spettrale per le matrici si ha che $\|E\| = \sqrt{\lambda(E^TE)} = |\lambda(E)| = \rho(E)$, la condizione necessaria e sufficiente per la convergenza dello schema (3.2) è:

$$\rho(E) < 1. \tag{3.6}$$

Alternativamente, tala condizione si può dimostrare assumendo che la matrice di iterazione E sia diagonalizzabile[2], e cioè assumendo che gli autovettori di E possano essere usati come base per $\mathbb{R}^n$:

$$E = U\Lambda U^{-1},$$

con U la matrice le cui colonne sono gli autovettori di E e Λ la matrice diagonale contenente gli autovalori di E. In questo caso, il vettore errore iniziale si può espandere come combinazione lineare degli autovettori di E:

$$e_0 = \sum_{i=1}^{n} \gamma_i u_i = Ug \qquad U = [u_1, \ldots, u_n] \qquad g = \{\gamma_i\}.$$

Sostituendo in (3.5), si ottiene dunque:

$$e_k = E^k e_0 = \sum_{i=1}^{n} \gamma_i \lambda_i^k u_i = \lambda_1^k \sum_{i=1}^{n} \gamma_i \left(\frac{\lambda_i}{\lambda_1}\right)^k u_i,$$

avendo assunto l'ordinamento classico di autovalori e corrispondenti autovettori in ordine crescente in valore assoluto $|\lambda_1| < |\lambda_2| \leq |\lambda_3| \leq \ldots \leq |\lambda_n|$ prendendo la norma e notando che $|\lambda_i/\lambda_1| < 1$ per $i > 1$:

$$\|e_k\| \leq |\lambda_1|^k \sum_{i=1}^{n} \left|\frac{\lambda_i}{\lambda_1}\right|^k \|u_i\| \leq |\lambda_1|^k \|u_1\|, \tag{3.7}$$

da cui segue subito la (3.6).

Osservazione 3.1.2. Calcolando $\|e_{k+1}\| \,/\, \|e_k\|$ tramite l'equazione (3.7), e ricordando la definizione di ordine e costante asintotica di convergenza, si ricava subito che lo schema (3.2) converge con ordine 1 (lineare) e costante asintotica di convergenza $M = \rho(E)$.

Osservazione 3.1.3. Il raggio spettrale di una matrice quadrata A è definito dalle due condizioni equivalenti:

$$\rho(A) = \max_{\lambda \in \sigma_A} |\lambda(A)| = \lim_{n \to \infty} \|A^n\|^{\frac{1}{n}}$$

[2]Assunzione anche questa molto forte: in generale le matrici di iterazione non solo non saranno ne' diagonalizzabili ne' simmetriche ma saranno addirittura singolari!

3.1.1 Metodi lineari e stazionari classici

Ritorniamo ora a studiare i metodi di tipo (3.2) nella loro forma più classica. Un modo intuitivo per ricavare schemi lineari e stazionari è il seguente. Data una matrice non singolare P, chiamata "precondizionatore", si sommi a primo e secondo membro di (3.1) il vettore Px:

$$Px = Px - Ax + b = (P - A)x + b,$$

da cui si può ricavare il seguente schema iterativo:

$$x_{k+1} = (I - P^{-1}A)x_k + P^{-1}b. \tag{3.8}$$

che ha matrice di iterazione $E = I - P^{-1}A$ e che verifica la (3.4). Lo schema si può anche scrivere come:

$$x_{k+1} = x_k + P^{-1}r_k \qquad r_k = b - Ax_k.$$

Intuitivamente, guardando quest'ultima equazione, è immediato arguire che prendendo (idealmente) come precondizionatore $P = A$ si otterrebbe la soluzione x^* con una sola iterazione. Ovviamente questo non è concepibile in quanto il calcolo di A^{-1} corrisponde in termini di costo computazionale alla soluzione di n sistemi lineari. Cercheremo quindi di trovare delle matrici P "vicine" alla matrice A, che siano computazionalmente efficienti da calcolare e per le quali il prodotto $P^{-1}u$ (u vettore generico di $\mathbb{R}^n$) sia computazionalmente efficiente.

Un esempio semplice ma istruttivo di metodo lineare e stazionario è il metodo di Richardson, ottenuto da (3.8) prendendo $P = I$:

$$x_{k+1} = (I - A)x_k + b. \tag{3.9}$$

Si osservi che applicando il metodo di Richardson (3.9) al sistema "precondizionato":

$$P^{-1}Ax = P^{-1}b \tag{3.10}$$

si ottiene proprio il metodo riportato in (3.8). Quindi il metodo di Richardson con $E = I - P^{-1}A$ converge se $\rho(I - P^{-1}A) < 1$, o equivalentemente $\rho(P^{-1}A) < 1$.

Riprenderemo con più dettaglio lo schema di Richardson nel prosieguo. Vediamo ora alcuni classici esempi di matrici P. Tutti gli schemi seguenti si ricavano dallo "splitting" additivo della matrice del sistema $A = L + D + U$ con:

$$l_{ij} = \begin{cases} a_{ij}, & \text{if } i < j, \\ 0, & \text{if } i \geq j. \end{cases} \qquad d_{ij} = \begin{cases} a_{ii}, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases} \qquad u_{ij} = \begin{cases} 0, & \text{if } i \leq j, \\ a_{ij}, & \text{if } i > j. \end{cases}$$

Abbiamo i seguenti schemi:

Metodo di Jacobi ($P = D$). Prendendo $P = D$, si ottiene il metodo di Jacobi:

$$x_{k+1} = (I - D^{-1}A)x_k + D^{-1}b = x_k + D^{-1}r_k,$$

o, indicando con $x_{k+1,i}$ l'i-esimo elemento del vettore x_{k+1}, il metodo può essere scritto per componenti::

$$x_{k+1,i} = x_{k,i} + \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{n} a_{i,j}x_{k,j}\right) = \frac{1}{a_{ii}}\left(b_i - \sum_{\substack{j=1\\ j\neq i}}^{n} a_{i,j}x_{k,j}\right). \tag{3.11}$$

Metodo di Gauss-Seidel ($P = L + D$). Prendendo $P = L + D$, si ottiene

$$x_{k+1} = \left[I - (L+D)^{-1}A\right]x_k + (L+D)^{-1}b = x_k + (L+D)^{-1}r_k,$$

o, indicando con $x_{k+1,i}$ l'i-esimo elemento del vettore x_{k+1}, il metodo scritto per componenti diventa:

$$x_{k+1,i} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{i,j}x_{k+1,j} - \sum_{j=i+1}^{n} a_{i,j}x_{k,j}\right). \tag{3.12}$$

3.2 Metodi di rilassamento

La convergenza dei metodi studiati fino a qui può essere migliorata introducendo un parametro (detto di rilassamento) che dovrà essere identificato in modo da minimizzare il raggio spettrale della matrice di iterazione. A tal fine, riscriviamo il metodo (3.2) nel seguente modo:

$$x_{k+1} = (I - \alpha_k P^{-1}A)x_k + \alpha_k P^{-1}b = x_k + \alpha_k P^{-1}r_k, \tag{3.13}$$

con $\alpha_k \in \mathbb{R}$ parametro reale. Se $\alpha_k = \alpha$ è indipendente dall'iterazione k, lo schema è classificabile nell'ambito dei metodi stazionari (la matrice di iterazione non dipende da k). Altrimenti il metodo è "non stazionario". In questo caso la matrice di iterazione è:

$$E_k = I - \alpha_k P^{-1}A$$

L'algoritmo non stazionario può essere scritto nel modo seguente:

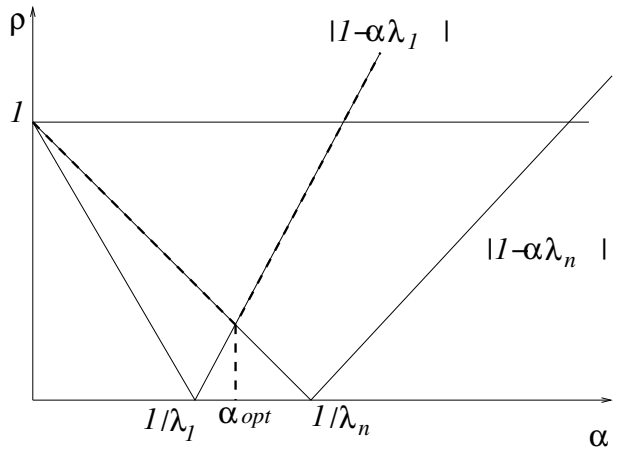

Figura 3.1: La curva $\rho(E_\alpha)$ in funzione di α

ALGORITHM RICHARDSON NON STAZIONARIO
Input: $x_0, nimax, toll$; $k = 0$;
$r_0 = b - Ax_0$.
FOR $k = 0, 1, \ldots$ fino a convergenza:

$$\begin{aligned} &1. \quad z_k &&= P^{-1} r_k && (3.14)\\ &2. \quad \alpha_k &&= \ldots && (3.15)\\ &3. \quad x_{k+1} &&= x_k + \alpha_k z_k && (3.16)\\ &4. \quad r_{k+1} &&= r_k - \alpha_k A z_k && (3.17) \end{aligned}$$

END FOR

dove il passo in (3.14), chiamato "applicazione del precondizionatore", si esegue in pratica risolvendo il sistema

$$P z_{k+1} = r_k,$$

e il passo in (3.15) dipende dal metodo.

Nel caso stazionario ($\alpha_k = \alpha$), si può studiare qual'è il valore ottimale del parametro, cioè il valore di α che minimizza il raggio spettrale della matrice di iterazione $\rho(E) = \rho(I - P^{-1}A)$. Infatti, notando che $\lambda(I - \alpha P^{-1}A) = 1 - \alpha\lambda(P^{-1}A)$ la condizione necessaria per la convergenza si può scrivere $|1 - \alpha\lambda_i| < 1$ per $i = 1, \ldots, n$, dove λ_i, l'i-esimo autovalore della matrice (nonsimmetrica) $P^{-1}A$, è in generale un mumero complesso. Questa disuguaglianza equivale a

$$(1 - \alpha \operatorname{Re}\lambda_i)^2 + \alpha^2 (\operatorname{Im}\lambda_i)^2 < 1,$$

da cui si ricava

$$\frac{\alpha|\lambda_i|^2}{2\,\mathrm{Re}\,\lambda_i} < 1 \qquad \forall i = 1, \ldots, n.$$

Nel caso in cui $\lambda_i(P^{-1}A) \in \mathbb{R}$ per ogni i, si ottiene che la condizione su α per la convergenza dello schema è:

$$0 < \alpha < \frac{2}{\lambda_{max}},$$

e si può ricavare il valore di α_{opt}, cioè il valore di α che minimizza il raggio spettrale della matrice di iterazione. Infatti si ha:

$$\rho(E_\alpha) = \max\left[|1 - \alpha\lambda_{min}|, |1 - \alpha\lambda_{max}|\right],$$

e il valore di α che minimizza la precedente è dato da:

$$\alpha_{opt} = \frac{2}{\lambda_{min} + \lambda_{max}}$$

Infatti, si può vedere facilmente dalla Figura 3.1 che il valore ottimale di α si trova nel punto di incontro delle curve $|1 - \alpha\lambda_{min}|$ e $|1 - \alpha\lambda_{max}|$, da cui il valore precedente. Il valore ottimale del raggio spettrale della matrice di iterazione si ricava immediatamente per sostituzione, ottenendo:

$$\rho_{opt} = \frac{\lambda_{max} - \lambda_{min}}{\lambda_{max} + \lambda_{min}} = \frac{\kappa(P^{-1}A) - 1}{\kappa(P^{-1}A) + 1} = \frac{\kappa(P^{-1}A) - 1}{\kappa(P^{-1}A) + 1}$$

Attualmente, tali schemi vengono raramente usati per la soluzione di sistemi lineari, ma sono assai utili come precondizionatori, come vedremo nel prossimo paragrafo, per il metodo del gradiente coniugato.

3.3 Metodi del Gradiente per la soluzione di sistemi lineari

Queste note sono un'elaborazione di un articolo di Jonathan Shewchuk, dal titolo esemplificativo *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain* [6], mediate con l'illuminante interpretazione geometrica riportata nel libro del Prof. Gambolati [1]. Per approfondimenti si rimanda il lettore al libro di C.T. Kelley [2] o meglio ancora al libro di Y. Saad [4].

3.3.1 Forme quadratiche

Data una matrice A di dimensione $n \times n$, un vettore $b \in \mathbb{R}^n$ e uno scalare $c \in \mathbb{R}$, una forma quadratica è una funzione quadratica di tutte le componenti $x_1, x_2, \ldots, x_n$:

$$f(x) : \mathbb{R}^n \to \mathbb{R}$$

definita da

$$f(x) = \frac{1}{2} x^T A x - b^T x + c = \frac{1}{2} \langle x, Ax \rangle - \langle b, x \rangle + c. \tag{3.18}$$

Siamo interessati a trovare il minimo di $f(x)$. Si dice che $f(x)$ ha un minimo in x^* se $f(x^*) < f(x)$ per tutti gli x nel dominio di f. Se tutte le derivate parziali di f sono nulle in x^*, allora x^* può essere un punto di minimo, un punto di massimo, oppure nessuno dei due. Ad esempio in $\mathbb{R}^2$, la funzione $z = f(x, y) = x^2 + y^2$ ha un punto di minimo assoluto in $(0, 0)$. D'altro canto, le derivate parziali della funzione $z = f(x, y) = xy$ si annullano entrambe nel punto $(0, 0)$, ma tale punto non è di minimo, (si chiama punto di stazionarietà). Infatti, come si vede facilmente, in tutti i punti (x, y) del primo e terzo quadrante (dove x e y hanno lo stesso segno) si ha $f(x, y) > 0 = f(0, 0)$, mentre per tutti i punti del secondo e quarto quadrante (dove x e y sono di segno opposto) si ha $f(x, y) < 0 = f(0, 0)$.

Prendiamo ora un vettore qualsiasi $v \in \mathbb{R}^n$ e uno scalare $\epsilon \in \mathbb{R}$ e formiamo un nuovo vettore $x^* + \epsilon v \in \mathbb{R}^n$. Perchè x^* sia punto di minimo assoluto dovrà essere:

$$f(x^* + \epsilon v) > f(x^*) \qquad \text{per ogni } \epsilon \text{ e ogni } v.$$

In particolare dovrà esserlo per ϵ piccolo e che tende a zero. Possiamo pensare ora la f come variabile di ϵ soltanto, per cui la condizione di stazionarietà per la $f(x)$ sarà quindi data da:

$$\frac{d}{d\epsilon} \left[f(x + \epsilon v) \right] |_{\epsilon=0} = 0,$$

che deve essere valida per ogni v. Utilizzando la regola di derivazione della funzione composta si può calcolare la precedente derivata ragionando componente per

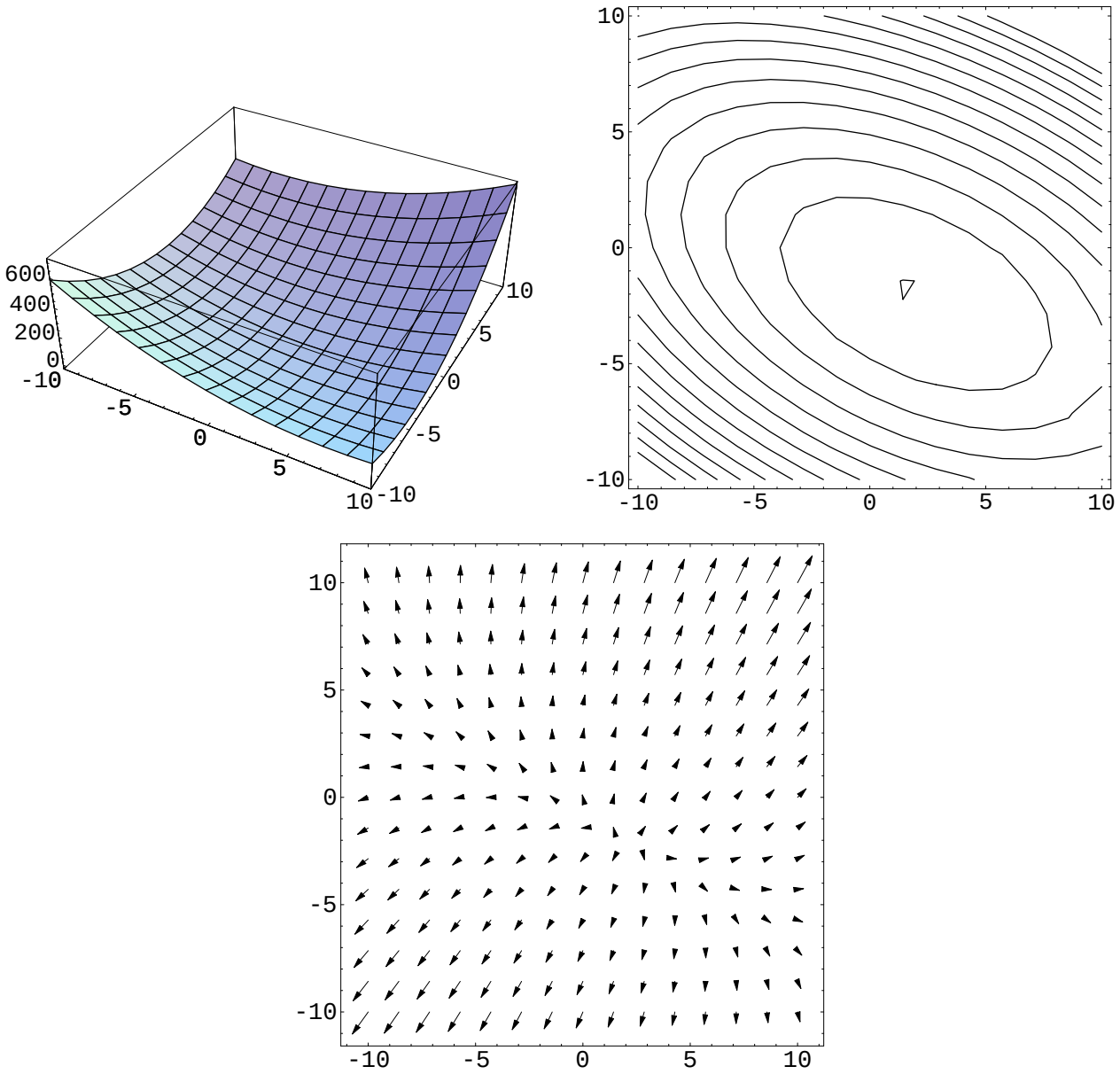

Figura 3.2: Forma quadratica corrispondente al sistema lineare (2.8). A sinistra in alto è rappersentato il grafico in $\mathbb{R}^2$; a destra in alto sono rappresentate le linee di livello; in basso il campo dei gradienti.

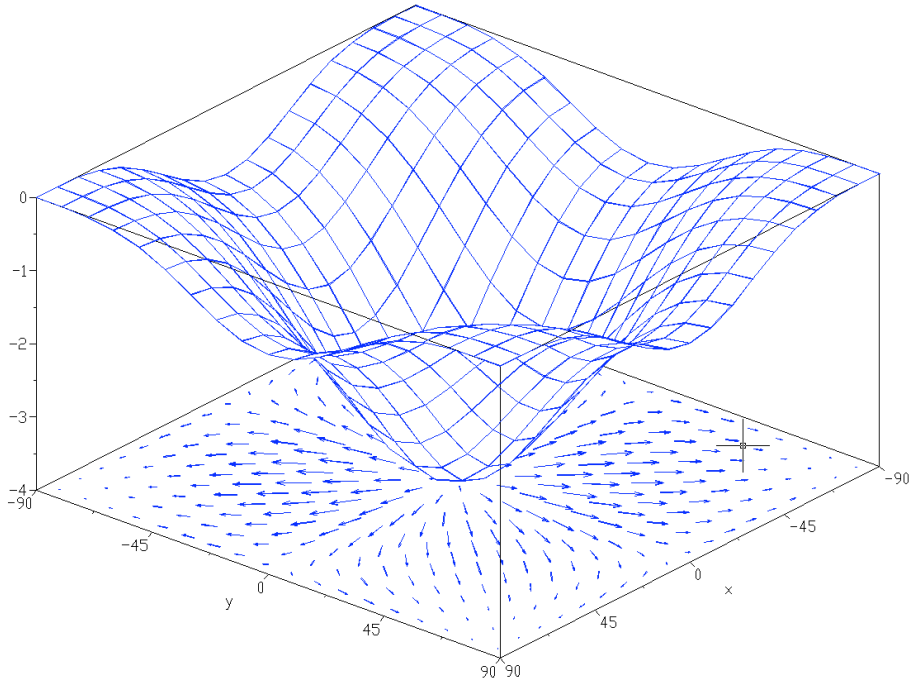

Figura 3.3: Grafico della funzione $f(x,y) = -(\cos^2 x + \cos^2 y)^2$ e del campo vettoriale $\nabla f(x,y) = (\partial f/\partial x, \partial f/\partial y)^T$

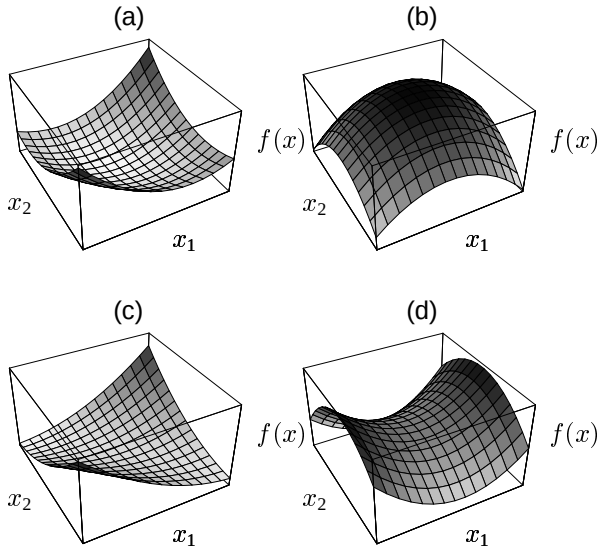

Figura 3.4: Grafico delle forme quadratiche (in $\mathbb{R}^2$) rappresentative di a) una matrice definita positiva, b) una matrice definita negativa, c) una matrice semidefinita positiva (in realtà è singolare e si nota la linea lungo la quale $f(x)$ è minima e che corrisponde alle inifite soluzione del sistema lineare corrispondente), d) una matrice indefinita (punto sella).

componente:

$$\frac{d}{d\epsilon}f(x_1, x_2, \ldots, x_n) = \frac{\partial f}{\partial x_1}\frac{dx_1}{d\epsilon} + \frac{\partial f}{\partial x_2}\frac{dx_2}{d\epsilon} + \cdots + \frac{\partial f}{\partial x_n}\frac{dx_n}{d\epsilon} = \frac{\partial f}{\partial x_1}v_1 + \frac{\partial f}{\partial x_2}v_2 + \cdots + \frac{\partial f}{\partial x_n}v_n.$$

Definendo il gradiente della funzione $f(x)$ come il vettore delle derivate parziali (si vedano le Figure 3.2 e 3.3):

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \cdot \\ \cdot \\ \cdot \\ \frac{\partial f}{\partial x_n} \end{bmatrix},$$

si ottiene immediatamente la condizione di minimizzazione di $f(x)$:

$$\frac{d}{d\epsilon}\left[f(x+\epsilon v)\right]\Big|_{\epsilon=0} = \nabla f(x)\cdot v = \langle \nabla f(x), v\rangle = \nabla f(x)^T v = 0. \tag{3.19}$$

Si noti che se $\|v\| = 1$ $\langle \nabla f(x), v\rangle = D_v f(x)$ è la derivata direzionale di f in x lungo la direzione v. D'altro canto, nel caso della nostra forma quadratica (3.18), possiamo

scrivere:

$$\begin{aligned} f(x+\epsilon v) &= \frac{1}{2}\langle (x+\epsilon v), A(x+\epsilon v)\rangle - \langle b, (x+\epsilon v)\rangle + c \\ &= \frac{1}{2}\left[\langle x, Ax\rangle + \epsilon\langle v, Ax\rangle + \epsilon\langle x, Av\rangle + \epsilon^2\langle v, Av\rangle\right] - \langle b, x\rangle - \epsilon\langle b, v\rangle + c, \end{aligned}$$

da cui:

$$\begin{aligned} \frac{d}{d\epsilon}\left[f(x+\epsilon v)\right]|_{\epsilon=0} &= \frac{1}{2}\left[\langle v, Ax\rangle + \langle x, Av\rangle\right] - \langle b, v\rangle \\ &= \left\langle \frac{A+A^T}{2}x, v\right\rangle - \langle b, v\rangle = 0. \end{aligned}$$

Di conseguenza:

$$\begin{aligned} \nabla f(x) &= \frac{A+A^T}{2}x - b \\ (\frac{A+A^T}{2}x &- b) \perp v, \end{aligned} \tag{3.20}$$

e quest'ultima è valida per ogni $v \in \mathbb{R}^n$, da cui immediatamente

$$\frac{A+A^T}{2}x = b,$$

e se $A = A^T$ si ottiene che la condizione di minimio (3.19) si può scrivere come:

$$\begin{aligned} \nabla f(x) &= Ax - b \\ Ax &= b. \end{aligned} \tag{3.21}$$

Questo ci dice che la forma quadratica $f(x)$ ha un minimo in corrispondenza della soluzione del sistema lineare $Ax = b$, se A è simmetrica, del sistema $\frac{1}{2}(A+A^T)x = b$ se A non è simmetrica.

3.3.2 Caso simmetrico $A = A^T$

Si può notare che il minimo della $f(x)$ definito sopra è globale se e solo se A è simmetrica e definita positiva. Infatti, perchè x^* sia minimo globale è necessario che $f(x^*+e) > f(x^*)$ per ogni vettore $e \in \mathbb{R}^n$. Quindi:

$$\begin{aligned} f(x^*+e) &= \frac{1}{2}\langle A(x^*+e), (x^*+e)\rangle - \langle b, (x^*+e)\rangle + c \\ &= \frac{1}{2}\left[\langle Ax^*, x^*\rangle + \langle Ax^*, e\rangle + \langle Ae, x^*\rangle + \langle Ae, e\rangle\right] - \langle b, x^*\rangle - \langle b, e\rangle + c \\ &= \frac{1}{2}\langle Ax^*, x^*\rangle - \langle b, x^*\rangle + c + \langle Ax^*, e\rangle + \frac{1}{2}\langle Ae, e\rangle - \langle b, e\rangle \\ &= f(x^*) + \frac{1}{2}\langle Ae, e\rangle, \end{aligned}$$

dove abbiamo usato il fatto che se $A = A^T$ si ha $\langle Az, w\rangle = \langle z, Aw\rangle$. Se A è definita positiva, $\langle Ae, e\rangle > 0$ da cui si ricava che $f(x^*)$ è minimo assoluto.

Nella Figura 3.2 viene rappresentato il grafico della $f(x)$ e il campo dei gradienti $\nabla f(x)$ nel caso particolare del sistema lineare dato dalla (2.8), con A matrice simmetrica. Si noti che le linee di livello che rappresentano la $f(x)$ (Figura 3.2 in alto a destra) sono delle ellissi i cui assi principali sono gli autovettori della matrice A. In $\mathbb{R}^n$ gli ellissi saranno degli iper-ellissoidi, e le linee di livello si trasformeranno in superfici di livello (o isosuperfici), l'interpretazione geometrica, seppur non rappresentabile graficamente, rimane la stessa. In altre parole, l'equazione $f(x) = C$ rappresenta un iper-ellissoide in $\mathbb{R}^n$ il cui centro coincide con il minimo di $f(x)$ e quindi con la soluzione del sistema lineare [3].

Per vedere questo si deve lavorare su un nuovo sistema di riferimento ottenuto come segue. Si noti che data una soluzione approssimata $x_k \neq x^*$, l'errore associato è rappresentato da:

$$e_k = x_k - x^*$$

Effettuiamo ora un cambio di variabili e poniamoci in un sistema di riferimento z dato da:

$$e_k = Uz, \tag{3.22}$$

dove z è la nuova variabile e U è la matrice le cui colonne sono gli autovettori u_i di A ordinati secondo gli autovalori di A (sempre positivi) decrescenti. La matrice U è unitaria (o ortogonale)[4] e cioè $U^{-1} = U^T$. Il nuovo sistema di riferimento ha l'origine coincidente con il centro dell'iper-ellissoide e gli assi coordinati coincidenti con gli autovettori di A, per cui si può scrivere la forma quadratica seguente:

$$\Phi(e_k) = \langle e_k, Ae_k\rangle = \langle Uz, AUz\rangle = z^T\Lambda z = \langle z, \Lambda z\rangle\,,$$

le cui superfici di livello sono rappresentate dalla seguente equazione:

$$\lambda_1 z_1^2 + \lambda_2 z_2^2 + \ldots + \lambda_n z_n^2 = \text{cost},$$

ovvero:

$$\frac{z_1^2}{\frac{\text{cost}}{\lambda_1}} + \frac{z_2^2}{\frac{\text{cost}}{\lambda_2}} + \cdots \frac{z_n^2}{\frac{\text{cost}}{\lambda_n}} = 1, \tag{3.23}$$

che è l'equazione in forma canonica dell'iper-ellissoide avente come lunghezza dei semi-assi:

$$a_1 = \sqrt{\frac{\text{cost}}{\lambda_1}}, \qquad a_2 = \sqrt{\frac{\text{cost}}{\lambda_2}}, \ldots, a_n = \sqrt{\frac{\text{cost}}{\lambda_n}}$$

[3] Si noti che il grafico $z = f(x)$ è rappresentabile in $\mathbb{R}^{n+1}$.

[4] Si ricordi che A è simmetrica e definita positiva, e quindi diagonalizzabile, da cui deriva che i suoi autovettori formano una base per lo spazio vettoriale $\mathbb{R}^n$.

Notiamo che la forma quadratica $\Phi(\cdot)$ ha come punto di minimo il vettore nullo, cioè $e_k = x_k - x^* = 0$, corrispondente proprio al minimo della $f(\cdot)$, ovvero la soluzione del sistema lineare. Infatti:

$$f(x_k) = f(x^* + e_k) = f(x^*) + \frac{1}{2}\langle e_k, Ae_k\rangle = \frac{1}{2}\Phi(e_k)$$

Inoltre, la $\Phi(\cdot)$ coincide con la $\Phi_3(x) = e^T Ae$ descritta sul libro di Gambolati [1] a meno di una costante moltiplicativa positiva.

Abbiamo quindi dimostrato che trovare la soluzione del sistema lineare $Ax = b$ è equivalente a trovare il punto di minimo della forma quadratica $f(x)$ ed è equivalente ad imporre l'ortogonalità del residuo a tutti i vettori di $\mathbb{R}^n$. In altre parole, per matrici simmetriche, i seguenti problemi sono equivalenti:

Problema 3.3.1 (sistema lineare)**.**
Trovare $x \in \mathbb{R}^n$ tale che:

$$Ax = b. \tag{S}$$

Problema 3.3.2 (di minimizzazione)**.**
Trovare $x \in \mathbb{R}^n$ tale che:

$$F(x) \leq F(v) \qquad \forall v \in \mathbb{R}^n. \tag{M}$$

Problema 3.3.3 (variazionale)**.**
Trovare $x \in \mathbb{R}^n$ tale che:

$$\langle b - Ax, v\rangle = 0 \qquad \forall v \in \mathbb{R}^n. \tag{V}$$

Metodo del gradiente o della discesa più ripida (o dello *steepest descent)*

Poichè il gradiente di f è interpretabile come la direzione di massimo aumento della funzione f, dalla (3.20) si vede immediatamente che il residuo del sistema lineare coincide con la direzione di discesa più ripida (Steepest Descent):

$$r = b - Ax = -\nabla f(x).$$

E' quindi intuitivo costruire un algoritmo basato sul calcolo di tale gradiente. Questo metodo, detto appunto metodo del gradiente o metodo della ricerca più ripida o dello Steepest Descent (SD), cerca di percorrere in ogni punto la direzione di discesa più ripida per arrivare al minimo assoluto. E' intuitivo pensare che tale algoritmo funzionerà solo se non esistono minimi relativi: in un minimo relativo, non è più necessariamente definita una direzione di discesa, e il metodo potrebbe trovarsi in condizioni di stallo.

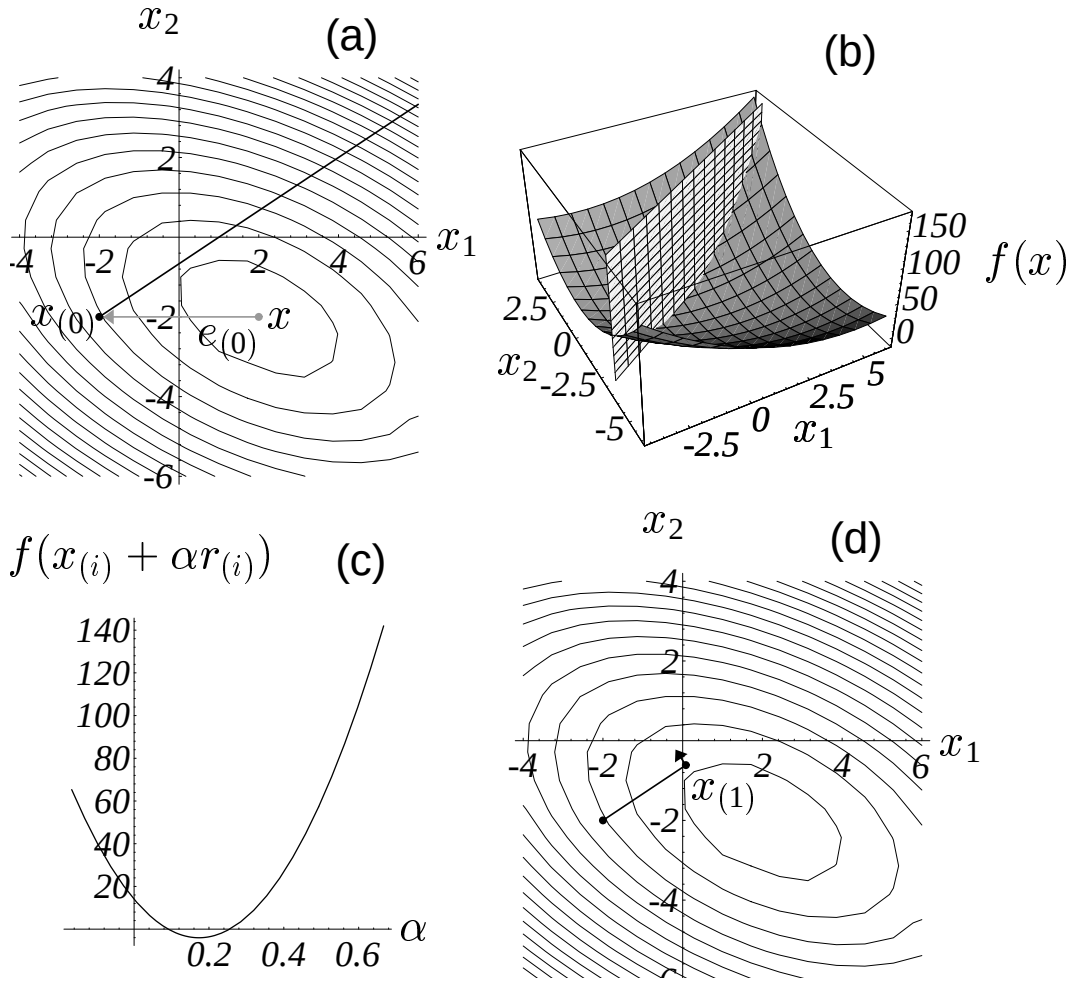

Figura 3.5: Interpretazione in $\mathbb{R}^2$ delle varie fasi dello schema dello steepest descent per la soluzione del sistema (2.8): a) la direzione di ricerca r_0 e l'errore iniziale e_0; b) la forma quadratica e il piano ortogonale a x_1 e x_2 passante per la direzione r_0; c) la funzione $f(\alpha_0)$ nella direzione r_0; d) la nuova iterata x_1 e la nuova direzione del gradiente.

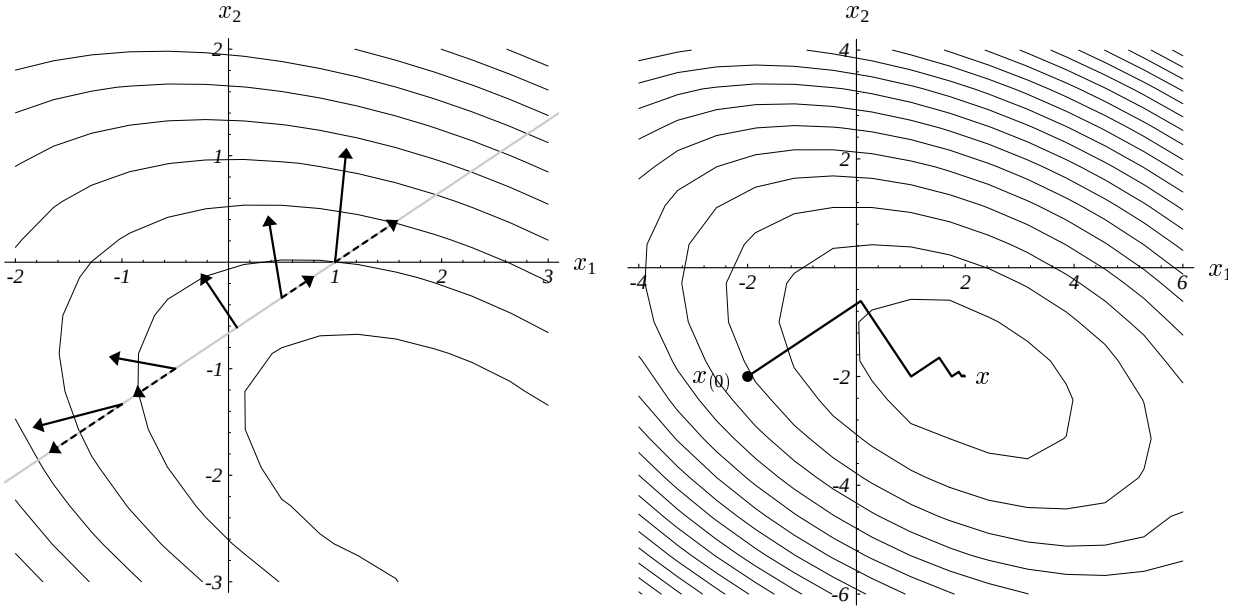

Figura 3.6: Interpretazione in $\mathbb{R}^2$ delle varie fasi dello schema dello steepest descent per la soluzione del sistema (2.8). A sinistra sono disegnati i vettori $\nabla f(x_k + \alpha_k r_k)$ lungo la direzione r_k; la $f(x_{k+1})$ è minima nel punto in cui $r_k \perp \nabla f(x_k + \alpha_k r_k)$. A destra sono rappresentate alcune iterazioni del metodo a partire dal punto iniziale $(-2, -2)^T$. Lo schema converge alla soluzione $(2, -2)^T$.

Costruiamo allora un algoritmo iterative basato su:

$$x_{k+1} = x_k + \alpha_k r_k,$$

dove l'indice k indica l'iterazione. Il valore di α_k viene determinato imponendo che lungo la direzione r_k il funzionale sia minimo (procedimento di line search, si veda la sua interpretazione geometrica riportata in Fig. 3.6(a,b)). Il vettore r_k individua la direzione di ricerca.

Per fare il conto dell'espressione di α_k, annulliamo semplicemente la derivata prima della forma quadratica pensata come funzione di α_k. Imponiamo cioè:

$$\frac{d}{d\alpha_k} f(x_k + \alpha_k r_k) = \langle \nabla f(x_k + \alpha_k r_k), r_k \rangle = \langle \nabla f(x_{k+1}), r_k \rangle = 0,$$

ma visto che $\nabla f(x_{k+1}) = -r_{k+1}$, si impone:

$$\langle r_{k+1}, r_k \rangle \Rightarrow \langle (b - Ax_{k+1}), r_k \rangle = 0,$$

e siccome $A = A^T$:

$$\langle b, r_k \rangle - \langle Ax_k, r_k \rangle - \alpha_k \langle Ar_k, r_k \rangle = \langle r_k, r_k \rangle - \alpha_k \langle Ar_k, r_k \rangle = 0,$$

da cui si ricava immediatamente:

$$\alpha_k = \frac{\langle r_k, r_k \rangle}{\langle Ar_k, r_k \rangle}. \tag{3.24}$$

L'algoritmo può quindi essere scritto nel modo seguente.

ALGORITHM SD
Input: $x_0, nimax, toll$; $k = 0$;
$r_0 = b - Ax_0$.
FOR $k = 0, 1, \ldots$ fino a convergenza:

1. $\alpha_k = \dfrac{\langle r_k, r_k \rangle}{\langle Ar_k, r_k \rangle}$ (3.25)
2. $x_{k+1} = x_k + \alpha_k r_k$ (3.26)
3. $r_{k+1} = r_k - \alpha_k Ar_k$ (3.27)

END FOR

In Fig. 3.6 a destra vengono rappresentate le prime 6 iterazioni del metodo per la soluzione del sistema (2.2).

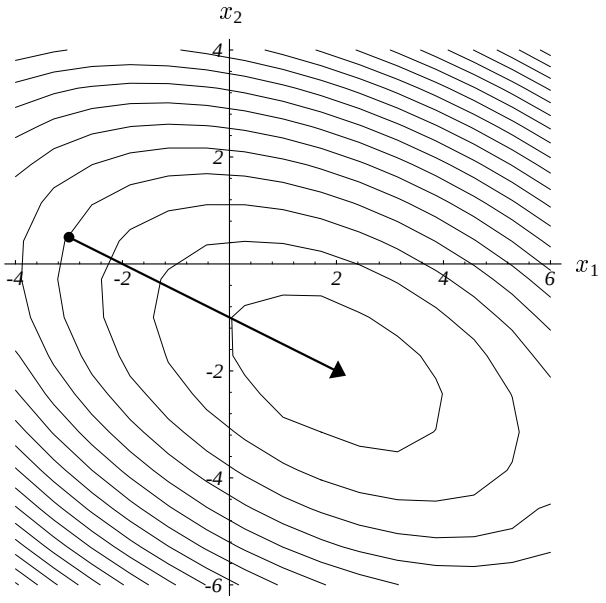

Figura 3.7: Il metodo dello steepest descent converge in una sola iterazione se la direzione del residuo iniziale coincide con quella di un autovettore.

Osservazione 3.3.4. Come caso "molto" particolare, supponiamo che alla iterazione k-esima l'errore e_k coincida con un autovettore della matrice A e chiamiamo λ_k il corrispondente autovalore, per cui:

$$r_k = -Ae_k = -\lambda_k e_k,$$

da cui si vede che il residuo è anch'esso un autovettore. Sostituendo nella (3.26) dell'ALGORITM SD si ottiene immediatamente:

$$e_{k+1} = e_k + \frac{\langle r_k, r_k \rangle}{\langle r_k, Ar_k \rangle} r_k = e_k + \frac{\lambda_k^2 \langle e_k, e_k \rangle}{\lambda_k^3 \langle e_k, e_k \rangle}(-\lambda_k e_k) = 0.$$

In altre parole, come si può vedere nell'interpretazione geometrica di Fig. 3.7, r_k è la direzione dell'autovettore e_k di A e coincide quindi con uno degli assi principali dell'iper-ellissoide e quindi passa per il centro (punto di minimo).

E' chiaro che per vedere il comportamento dell'errore nello schema SD si può sfruttare la relazione precedente utilizzando una rappresentazione dell'errore tramite la base data dagli autovettori di A. Per fare questo scriviamo:

$$\begin{aligned} e_k &= \sum_{j=1}^{n} \gamma_j u_j \\ r_k &= -Ae_k = -\sum_{j=1}^{n} \gamma_j \lambda_j u_j, \end{aligned} \tag{3.28}$$

ma poichè:

$$\|e_k\|^2 = \langle e_k, e_k \rangle = \sum_{j=1}^{n} \gamma_j^2 \qquad e \qquad \|e_k\|_A^2 = \langle e_k, Ae_k \rangle = \sum_{j=1}^{n} \gamma_j^2 \lambda_j,$$

e analogamente:

$$\|r_k\|^2 = \langle r_k, r_k \rangle = \sum_{j=1}^{n} \gamma_j^2 \lambda_j^2 \qquad e \qquad \|r_k\|_A^2 = \langle r_k, Ar_k \rangle = \sum_{j=1}^{n} \gamma_j^2 \lambda_j^3,$$

abbiamo immediatamente:

$$\begin{aligned} e_{k+1} &= e_k + \frac{\langle r_k, r_k \rangle}{\langle r_k, Ar_k \rangle} r_k \\ &= e_k + \frac{\sum_{j=1}^{n} \gamma_j^2 \lambda_j^2}{\sum_{j=1}^{n} \gamma_j^2 \lambda_j^3} r_k. \end{aligned}$$

Come visto prima, se e_k è tale che solo uno degli $\alpha_j \neq 0$ basta scegliere $\alpha_j = 1/\lambda_j$ e si ha convergenza immediata. Se d'altra parte tutti gli autovalori sono uguali, si ricava ancora che $e_{k+1} = 0$. In questo caso, infatti, gli iper-ellissoidi sono in realtà delle iper-sfere, per cui le direzioni del residuo puntano direttamente verso il centro. Si noti da ultimo che il termine

$$\frac{\langle r_k, r_k \rangle}{\langle r_k, Ar_k \rangle} = \frac{\sum_{j=1}^{n} \gamma_j^2 \lambda_j^2}{\sum_{j=1}^{n} \gamma_j^2 \lambda_j^3}$$

può essere intuitivamente visto come una media pesata degli inversi degli autovalori e i pesi γ_j^2 fanno sì che le componenti (intese come i vettori u_j di (3.28)) più "lunghe" di e_k vengano "accorciate" prima delle altre. Come risultato si ha che alcune delle componenti più "corte" potrebbero anche aumentare, da cui segue che la convergenza dello schema non è monotona, ma in genere non per molto.

Convergenza di SD

Per studiare formalmente la convergenza di SD abbiamo bisogno di utilizzare quella che viene chiamata la norma energia del vettore x, definita come:

$$\|x\|_A = \sqrt{\langle Ax, x \rangle} = \sqrt{\langle x, Ax \rangle} = \sqrt{x^T Ax}.$$

Per prima cosa vediamo che il minimo della forma quadratica $f(x_k)$ implica minimizzare la norma energia dell'errore $\|e_k\|_A$. Infatti, dalla definizione di errore e_k e usando il fatto che $A^T = A$, si ha immediatamente:

$$f(x_k) = f(x + e_k) = f(x) + \frac{1}{2} \langle Ae_k, e_k \rangle,$$

da cui, siccome $\frac{1}{2}\langle Ae_k, e_k\rangle \geq 0$ e $f(x)$ è il minimo assoluto della forma quadratica, segue l'affermazione. Quindi:

$$\begin{aligned}
\|e_{k+1}\|_A^2 &= \langle Ae_{k+1}, e_{k+1}\rangle = \langle A(e_k + \alpha_k r_k), (e_k + \alpha_k r_k)\rangle \\
&= \langle Ae_k, e_k\rangle + 2\alpha_k \langle Ae_k, r_k\rangle + \alpha_k^2 \langle Ar_k, r_k\rangle \\
&\quad \text{(ricordando che } Ae_k = -r_k\text{)} \\
&= \|e_k\|_A^2 - \frac{(\langle r_k, r_k\rangle)^2}{\langle Ar_k, r_k\rangle} \\
&= \|e_k\|_A^2 \left(1 - \frac{(\langle r_k, r_k\rangle)^2}{\langle Ar_k, r_k\rangle \langle Ae_k, e_k\rangle}\right) \\
&= \|e_k\|_A^2 \left(1 - \frac{(\sum_j \gamma_j^2 \lambda_j^2)^2}{(\sum_j \gamma_j^2 \lambda_j^3)(\sum_j \gamma_j^2 \lambda_j)}\right) \\
&= \omega^2 \|e_k\|_A^2,
\end{aligned}$$

con evidentemente ω^2 che deve essere minore di 1 per avere convergenza. In $\mathbb{R}^2$ è ragionevolmente facile trovare un significato per il numero ω^2. Infatti, se $\lambda_1 \geq \lambda_2$, definiamo il numero di condizionamento spettrale della matrice A come $\kappa(A) = \lambda_1/\lambda_2$ (in generale per una matrice A di dimensioni $n \times n$ sarà $\kappa(A) = \lambda_1/\lambda_n$), e definiamo $\mu = \gamma_2/\gamma_1$. Si noti che $\kappa(A)$ è chiamato il numero di condizionamento della matrice. Valori di $\kappa(A)$ relativamente grandi corrispondono a matrici relativamente malcondizionate. Allora:

$$\omega^2 = 1 - \frac{(\kappa(A)^2 + \mu^2)^2}{(\kappa(A) + \mu^2)(\kappa(A)^3 + \mu^2)}.$$

Analizzando questa relazione, si vede che il caso peggiore si ha quando $\kappa(A)^2 = \mu^2$, e cioè:

$$\omega^2 \leq 1 - \frac{4\kappa(A)^4}{(\kappa(A)^3 + \kappa(A)^2)(\kappa(A)^2 + \kappa(A))} = \left(\frac{\kappa(A) - 1}{\kappa(A) + 1}\right)^2.$$

In questo caso ω può essere visto come un'approssimazione della costante asintotica dell'errore:

$$\|e_{k+1}\|_A \leq \frac{\kappa(A) - 1}{\kappa(A) + 1} \|e_k\|_A,$$

e cioè:

$$\|e_k\|_A \leq \left(\frac{\kappa(A) - 1}{\kappa(A) + 1}\right)^k \|e_0\|_A,$$

e l'errore relativo sulla forma quadratica diventa:

$$\frac{f(x_k) - f(x)}{f(x_o) - f(x)} = \frac{\frac{1}{2}\langle e_k, Ae_k\rangle}{\frac{1}{2}\langle e_0, Ae_0\rangle} \leq \left(\frac{\kappa(A) - 1}{\kappa(A) + 1}\right)^{2k}. \tag{3.29}$$

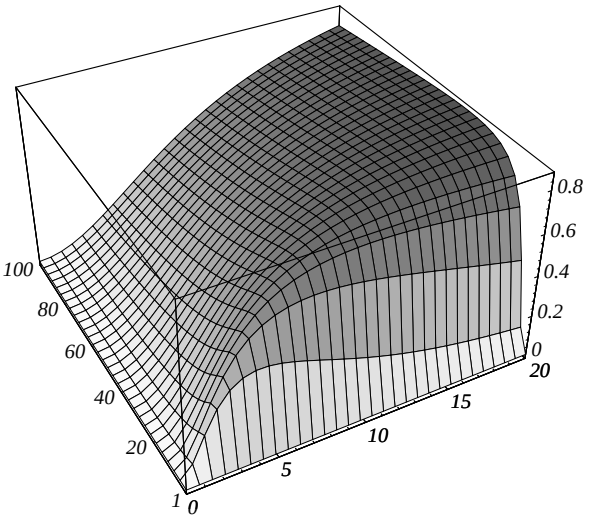

Figura 3.8: Andamento del fattore di convergenza ω.

Si può dimostrare che l'epressione precedentemente scritta per l'errore è valida anche in $\mathbb{R}^n$.

Osservazione 3.3.5. Il gradiente della forma quadratica è pari a $\nabla f(z) = \Lambda z$. La relazione di aggiornamento della soluzione approssimata x_{k+1} nel nuovo riferimento diventa quindi:

$$z_{k+1} = z_k - \alpha_k U^T U \Lambda z_k,$$

per cui se la soluzione iniziale z_0 (che coincide con l'errore iniziale) ha la componente i-esima nulla, anche z_1 avrà la componente i-esima nulla, e quindi $x_{1,i}$ sarà esatta. Se le componenti esatte sono molte, si avrà una notevole accelerazione della convergenza.

L'andamento di ω in funzione di $\kappa(A)$ e μ è graficato in Fig. 3.8. Si vede che se l'errore iniziale coincide con un autovettore, e quindi $\mu = 0$ (o ∞), $\omega = 0$ e la convergenza è istantanea. Lo stesso avviene per $\kappa(A) = 1$ e cioè se gli autovalori sono tutti uguali.

In Fig. 3.9 è invece illustrato il "cammino" di SD per vari valori di $\kappa(A)$ e μ nel sistema di riferimento definito dagli autovettori di A. Come si verifica da (3.29), la convergenza peggiora per valori di $\kappa(A)$ grandi, in corrispondenza al fatto che la matrice risulta malcondizionata. I primi due grafici in alto di Fig. 3.9 rappresentano degli esempi caratterizzati da $\kappa(A)$ grande; SD può convergere velocemente se il punto iniziale è fortunato (Fig. 3.9 alto a sinistra) ma in generale la convergenza è molto lenta (Fig. 3.9 alto a destra). Nei due grafici in basso, invece, l'iper-ellissoide è

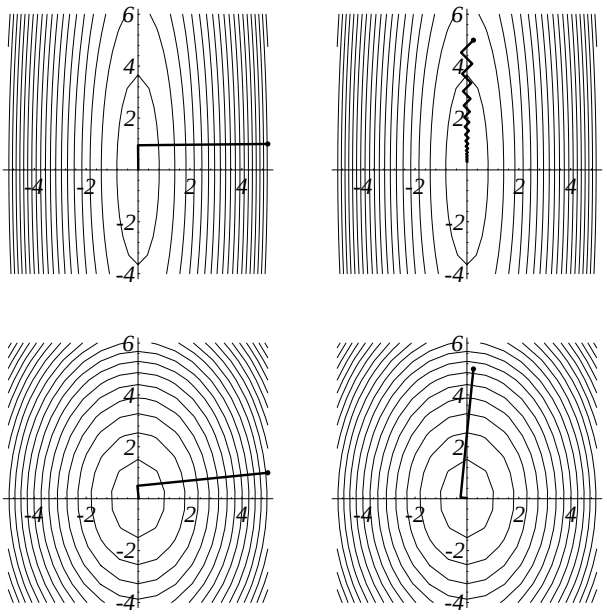

Figura 3.9: Esempi di convergenza del metodo di Steepes Descent in corrispondenza a valori estremi di $\kappa(A)$ e μ relativi alla figura 3.8. Le due figure in alto si riferiscono al casso di $\kappa(A)$ grande, mentre quelle in basso sono caratterizzate da un valore di $\kappa(A)$ vicino a 1.

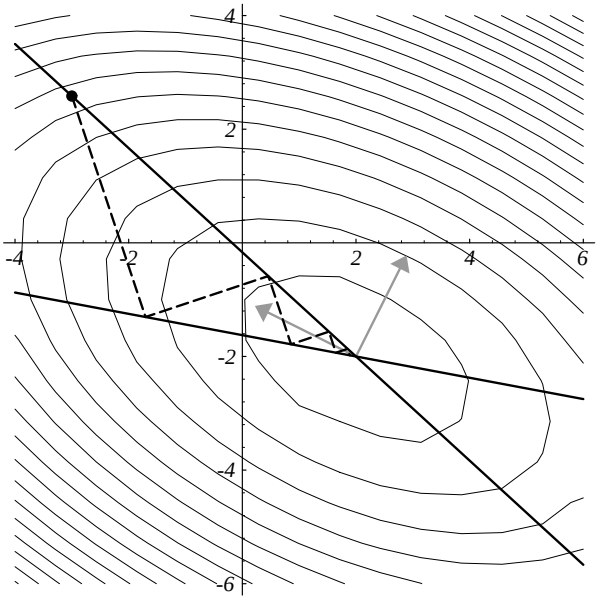

Figura 3.10: I punti iniziali peggiori per la convergenza di SD sono localizzati sulle linee continue nere. Le linee tratteggiate rappresentano il percorso delle iterate e formano un angolo di 45° rispetto agli assi dell'iper-ellissoide visualizzati con le freccie grigie. Per questo caso $\kappa(A) = 3.5$.

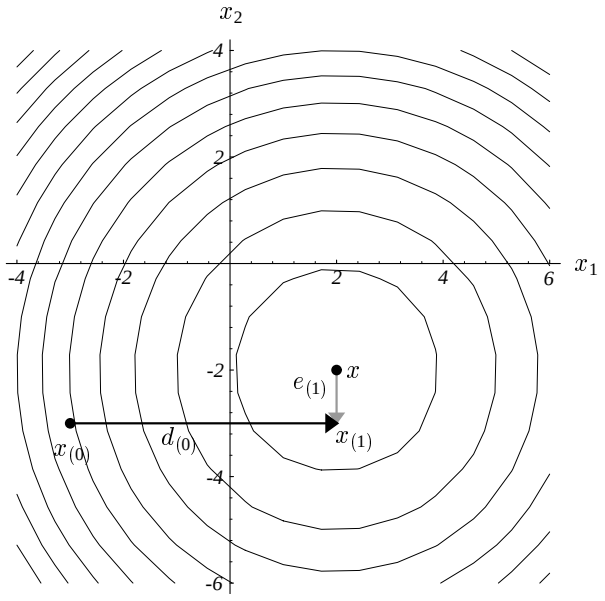

Figura 3.11: Utilizzando due direzioni ortogonali si arriva a convergenza in due iterazioni usando la direzione dell'errore e_k

quasi "sferico" (infatti $\kappa(A) \approx 1$) e la convergenza è veloce indipendentemente dalla soluzione iniziale.

Nella Fig. 3.10, corrispondente al solito sistema (2.8) per il quale $\kappa(A) = 3.5$, sono riportate le linee dei punti iniziali "peggiori". Le equationi delle linee sono date da $\gamma_2/\gamma_1 = \pm\kappa(A)$.

Il metodo del gradiente coniugato

Si è visto in precedenza che il metodo SD può convergere lentamente. In particolare, si é visto che due direzioni di ricerca (gradienti) successive sono ortogonali tra di loro (cfr. Fig. 3.10). Di conseguenza, solo due direzioni di discesa vengono utilizzate nell'intero processo, essendo r_k ortogonale a r_{k-1} e parallelo a r_{k-2}. Ovviamente, se le due direzioni di ricerca non passano per il punto di minimo del funzionale, il numero di iterazioni di SD per arrivare alla soluzione esatta è infinito. Per migliorare la convergenza è quindi necessario utilizzare direzioni di ricerca che non si ripetano.

Cerchiamo quindi di costruire uno schema che ad ogni iterazione generi una nuova direzione di ricerca diversa (ortogonale) da tutte le precedenti (e di conseguenza linearmente indipendente da tutte le altre). Lungo ogni direzione eseguiamo un passo di lunghezza tale da "allinearsi" al punto di minimo (il centro dell'iperelissoide) in quella direzione. Come conseguenza, nel caso peggiore la n-esima direzione di ricerca passa per il punto di minimo di $f(x)$ e lo schema in teoria converge con al massimo

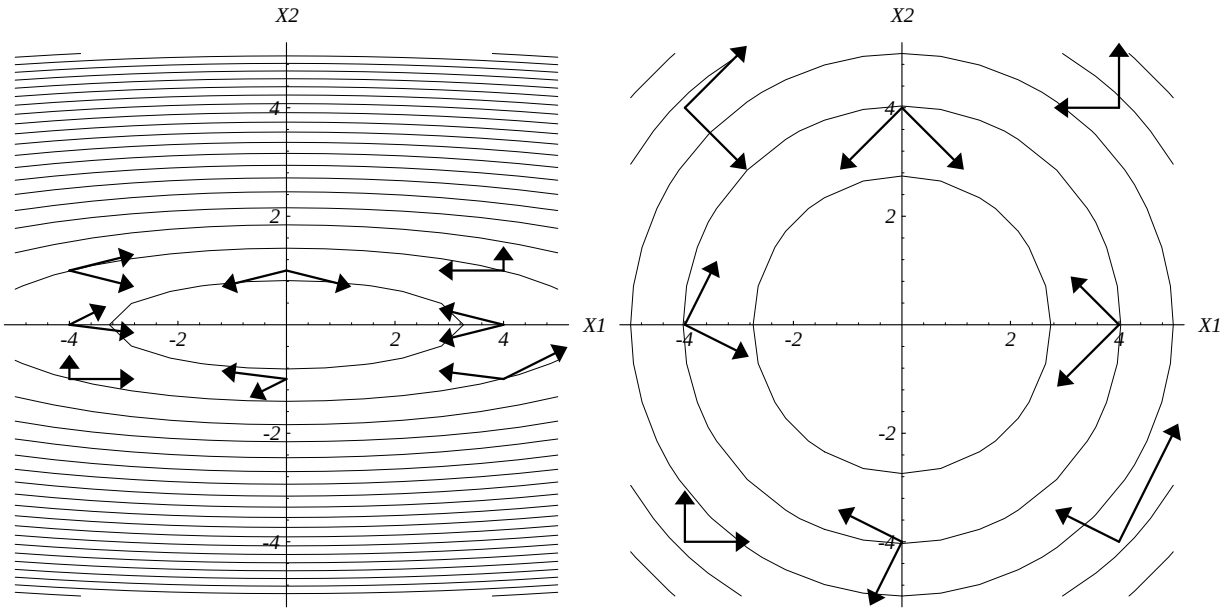

Figura 3.12: Sinistra: coppie di vettori A-coniugati; destra: gli stessi vettori in un piano deformato con la matrice A.

n iterazioni. Si dice allora che lo schema iterativo ha terminazione finita, intendendo cioè che lo schema ha un impianto simile ad uno schema iterativo (formule ricorsive per l'aggiornamento delle variabili) ma raggiunge la soluzione esatta in un numero predefinito di iterazioni. Per esempio, in $\mathbb{R}^2$, data una prima direzione (chiamiamola p_0) ad esempio parallela all'asse x, si esegue lungo essa un passo di lunghezza tale da fermarsi sul punto di intersezione tra la direzione p_0 e la normale ad essa passante per il centro. La direzione successiva (p_1) è parallela all'asse y e ci porta direttamente alla soluzione (si veda la Fig. 3.11). Notiamo anche che questa direzione è coincidente con la direzione dell'errore $e_{k+1} = x - x_{k+1}$. Possiamo scrivere questo come:

$$x_{k+1} = x_k + \alpha_k p_k, \tag{3.30}$$

e calcoliamo α_k in modo che e_{k+1} sia ortogonale a p_k:

$$\begin{aligned} \langle p_k, e_{k+1} \rangle &= 0 \\ \langle p_k, e_k + \alpha_k p_k \rangle &= 0 \qquad \text{dal'eq. (3.30)} \\ \alpha_k &= -\frac{\langle p_k, e_k \rangle}{\langle p_k, p_k \rangle}. \end{aligned}$$

Prima di procedere alla costruzione dello schema, vediamo alcune definizioni utili. Diciamo che due vettori x e y sono A-coniugati o A-ortogonali se:

$$\langle x, Ay \rangle = 0.$$

Si noti che se A è simmetrica e definita positiva, la formula precedente può essere usata come definizione di un nuovo prodotto scalare:

$$\langle x, y\rangle_A = \langle x, Ay\rangle = \langle Ax, y\rangle ,$$

che genera la norma (chiamata norma energia):

$$\|x\|_A = \sqrt{\langle x, x\rangle_A}.$$

Siano p_i, $i = 1, \ldots, n$, n vettori (direzioni) A-ortogonali, e cioè tali che

$$\langle p_i, Ap_j\rangle = \begin{cases} \neq 0, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases} . \tag{3.31}$$

Tali vettori sono linearmente indipendenti e quindi formano una base per $\mathbb{R}^n$. Infatti, se supponiamo che esistano delle costanti α_i tali per cui:

$$0 = \alpha_1 p_1 + \ldots + \alpha_n p_n;$$

moltiplicando per la matrice A (simmetrica e definita positiva) e poi scalarmente per p_i, $i = 1, \ldots, n$, si ottiene:

$$0 = \alpha_1 \langle p_i, Ap_1\rangle + \ldots + \alpha_n \langle p_i, Ap_n\rangle , \qquad i = 1, \ldots, n.$$

Dalla (3.31) si ottiene subito $\alpha_i = 0$.

Da quello che abbiamo visto in precedenza è facile arguire che se le isosuperfici (linee di livello del paraboloide identificato dalla forma quadratica) fossero sferiche il metodo SD convergerebbe in 1 iterazione. In questo caso, infatti, le direzioni del gradiente sono tutte ortogonali alle isosuperfici, per cui le direzioni calcolate da SD devono necessariamente passare per il centro. Guardando all'equazione (3.23), si nota che le isosuperfici sono sferiche se tutti gli autovalori λ_i sono uguali. E' possibile usare un cambiamento di variabili (cambiamento del sistema di riferimento) per trasformare gli iperellissoidi in ipersuperfici sferiche. Per fare ciò usiamo il fatto che A è simmetrica e definita positiva per cui $A = U\Lambda U^T$. Quindi:

$$\frac{1}{2}\langle x, Ax\rangle = \frac{1}{2}\langle y, y\rangle ,$$

dove

$$y = \Lambda^{\frac{1}{2}} U^T x.$$

Nelle nuove coordinate y le isosuperfici diventano quindi sferiche (basta sostituire opportunamente in (3.23)). L'idea è quindi di trovare delle direzioni q_i mutuamente

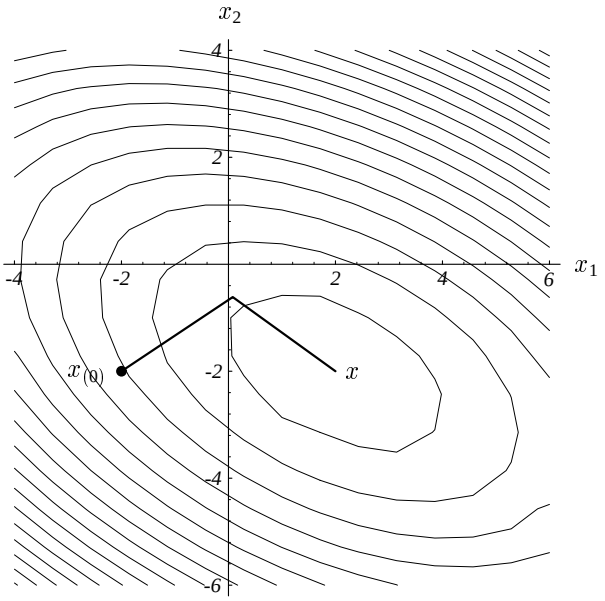

Figura 3.13: Interpretazione geometrica del metodo CG in $\mathbb{R}^2$: le due direzioni di ricerca sono A-coniugate, per cui la seconda necessariamente passa per il centro dell'elissoide, e quindi per il punto soluzione x^*.

ortogonali nel caso sferico e che quindi garantiscono la convergenza in al massimo n iterazioni. Tali direzioni dovranno quindi soddisfare:

$$\langle q_i, q_j \rangle = 0$$

per ogni indice i e j (diversi) compresi tra 1 e n. Possiamo ora tornare al caso "ellittico" notando che le direzioni q_i nel sistema "sferico" sono legate alle direzioni p_i nel sistema "ellittico" dalla relazione:

$$q_i = \Lambda^{\frac{1}{2}} U^T p_i;$$

la relazione di ortogonalità tra q_i e q_j in termini di p_i e p_j diventa:

$$\left\langle \Lambda^{\frac{1}{2}} U^T p_i, \Lambda^{\frac{1}{2}} U^T p_j \right\rangle = p_i^T U \Lambda^{\frac{1}{2}} \Lambda^{\frac{1}{2}} U^T p_j = p_i^T A p_j = \langle p_i, p_j \rangle_A = 0,$$

ovvero, p_i e p_j devono essere A-coniugati. Fig. 3.12 mostra la relazione che esiste tra due vettori A-coniugati rappresentati nel piano $\mathbb{R}^2$ oppure nel piano deformato utilizzando la matrice $A = U\Lambda U^T$. Si noti che la relazione di A-ortogonalità nello spazio vettoriale avente prodotto scalare standard ($\langle \cdot, \cdot \rangle$) si trasforma in una relazione di ortogonalità in uno spazio indotto dal prodotto scalare $\langle \cdot, \cdot \rangle_A$.

In sintesi, vogliamo calcolare n direzioni $p_o, \ldots, p_n$ che siano mutuamente A-coniugate, sulle quali andremo a cercare ogni volta il minimo della $f(x)$ utilizzando ancora la formula (3.24). Per fare questo, ovviamente, non possiamo servirci degli autovalori di A perchè sarebbe troppo costoso. L'interpretazione geometrica del metodo CG in $\mathbb{R}^2$ è mostrata in Fig. 3.13.

Andiamo a costruire quindi il seguente algoritmo del gradiente coniugato (CG):

ALGORITHM CG
Input: $x_0, nimax, toll$; $k = 0$;
$r_0 = b - Ax_0 = p_0$
FOR $k = 0, 1, \ldots$ fino a convergenza:

$$\alpha_k = \frac{\langle p_k, r_k \rangle}{\langle p_k, Ap_k \rangle} \tag{3.32}$$

$$x_{k+1} = x_k + \alpha_k p_k \tag{3.33}$$

$$r_{k+1} = r_k - \alpha_k A p_k \tag{3.34}$$

$$\beta_k = -\frac{\langle r_{k+1}, Ap_k \rangle}{\langle p_k, Ap_k \rangle} \tag{3.35}$$

$$p_{k+1} = r_{k+1} + \beta_k p_k \tag{3.36}$$

END FOR

Il coefficiente β_k è calcolato in modo da imporre la relazione di A-ortogonalità tra la direzione p_k e la nuova direzione p_{k+1}, cioè:

$$\langle p_{k+1}, Ap_k \rangle = \langle r_{k+1} + \beta_k p_k, Ap_k \rangle = 0,$$

da cui si ricava immediatamente la (3.35). Si può dimostrare per induzione che p_{k+1} è A-ortogonale anche alle precedenti direzioni e che i residui sono tra loro ortogonali:

$$\langle p_{k+1}, Ap_i \rangle = 0 \qquad i = 0, 1, \ldots, k \tag{3.37}$$

$$\langle r_{k+1}, r_i \rangle = 0 \qquad i = 0, 1, \ldots, k \tag{3.38}$$

Infatti, poniamo $p_0 = r_0$ $x_0 = p_0$. Quindi:

$\boxed{k = 0}$

$$\begin{aligned} x_1 &= x_0 + \alpha_0 p_0 \\ r_1 &= r_0 - \alpha_0 A p_0 \end{aligned}$$

imponiamo $\langle p_0, r_1 \rangle = 0$ ottenendo:

$$\langle p_0, r_0 - \alpha_0 A p_0 \rangle = 0 \qquad \Rightarrow \qquad \alpha_0 = \frac{\langle p_0, r_0 \rangle}{\langle p_0, Ap_0 \rangle}$$

scriviamo:

$$p_1 = r_1 + \beta_0 p_0$$

imponiamo $\langle p_0, Ap_1 \rangle = 0$ ottenendo:

$$\langle p_0, A(r_1 + \beta_0 p_0) \rangle = 0 \qquad \Rightarrow \qquad \beta_0 = -\frac{\langle p_0, Ar_0 \rangle}{\langle p_0, Ap_0 \rangle}$$

Si noti che $\langle r_0, r_1 \rangle = 0$.

$\boxed{k = 1}$

$$\begin{aligned} x_2 &= x_1 + \alpha_1 p_1 \\ r_2 &= r_1 - \alpha_1 A p_1 \end{aligned}$$

imponiamo $\langle p_1, r_2 \rangle = 0$ ottenendo:

$$\langle p_1, r_1 - \alpha_1 A p_1 \rangle = 0 \qquad \Rightarrow \qquad \alpha_1 = \frac{\langle p_1, r_1 \rangle}{\langle p_1, Ap_1 \rangle}$$

scriviamo:

$$p_2 = r_2 + \beta_1 p_1$$

imponiamo $\langle p_1, Ap_2 \rangle = 0$ ottenendo:

$$\langle p_1, A(r_2 + \beta_1 p_1) \rangle = 0 \qquad \Rightarrow \qquad \beta_1 = -\frac{\langle p_1, Ar_1 \rangle}{\langle p_1, Ap_1 \rangle}$$

Quindi si hanno i seguenti risultati:

$$\begin{aligned} \langle r_0, r_2 \rangle &= \langle p_0, r_2 \rangle = \langle p_0, r_1 - \alpha_1 A p_1 \rangle = 0 \\ \langle r_1, r_2 \rangle &= \langle r_2, p_1 - \beta_0 p_0 \rangle = 0 \\ \langle r_2, Ap_0 \rangle &= \frac{\langle r_2, r_0 - r_1 \rangle}{\alpha_0} = 0 \\ \langle p_0, Ap_2 \rangle &= \langle p_0, A(r_2 + \beta_1 p_1) \rangle = \langle r_2, Ap_0 \rangle + \beta_1 \langle p_0, Ap_1 \rangle = 0 \end{aligned}$$

$\boxed{k = \ldots}$

Alla fine del processo di induzione si vede chiaramente che:

$$\begin{aligned} \langle p_j, r_i \rangle &= 0 \qquad i > j \\ \langle r_i, r_j \rangle &= 0 \qquad i \neq j \\ \langle p_i, Ap_j \rangle &= 0 \qquad i \neq j. \end{aligned}$$

Si osservi che

$$\begin{aligned} x_2 &= x_0 + \alpha_0 r_0 + \alpha_1 p_1 \\ &= x_0 + (\alpha_0 + \alpha_1\beta_0) r_0 + \alpha_1 r_1 \\ &= x_0 + \gamma_1' r_0 + \gamma_2' A r_0 \\ &\dots \\ x_k &= x_0 + \sum_{i=0}^{k-1} \gamma_i A^i r_0 \end{aligned}$$

da cui si vede che la soluzione è data da una combinazione lineare dei vettori $A^i r_i$. Si dimostra che tali vettori sono linearmente indipendenti e generano uno spazio affine a $\mathbb{R}^k$ $\mathcal{K}_k \approx \mathbb{R}^k$, chiamato spazio di Krylov, per cui si può scrivere:

$$x_k \in x_o + \mathcal{K}_k,$$

dove

$$\mathcal{K}_k = \text{span}\left\{r_0, Ar_0, A^2 r_0, \dots, A^{k-1} r_0\right\}$$

Per questo il metodo del gradiente coniugato e i metodi derivati da esso si chiamono metodi di Krylov.

Si noti che, essendo i residui tra loro ortogonali (si veda la (3.38)), si verifica immediatamente che $r_n = 0$ e quindi $x_n = x^*$, e cioè il metodo CG in teoria ritorna la soluzione vera dopo n iterazioni. Si dice anche che CG è un metodo iterativo che ha terminazione finita, e per questo è "paragonabile" ad uno schema diretto, in quanto teoricamente la soluzione vera è raggiunta dopo un numero di operazioni finito e determinabile a priori. Questo fatto però è vero solo in teoria, perchè nella pratica le relazioni di ortogonalità precedentemente viste sono verificate solamente in maniera approssimata per la presenza degli errori di rappresentazione dei numeri all'elaboratore. Quindi lo schema va considerato alla stregua degli schemi iterativi. Inoltre, per questioni di costi computazionali, noi preferiremmo riuscire a raggiungere la tolleranza richiesta nell'errore con un numero di iterazioni molto inferiore a n. Per questo si ricorre alla tecnica del precondizionamento, che vedremo più avanti.

Convergenza di CG

Da quanto discusso prima, si deduce che la k-esima iterazione del metodo CG è il punto di minimo della forma quadratica $f(x)$ sul sottospazio $S = x_0 + \mathcal{K}_k$. Poichè la matrice A è simmetrica, minimizzare la $f(x)$ su S è equivalente a minimizzare $\|x - x^*\|_A$ sullo stesso sottospazio. Infatti, con pochi passaggi, si ricava (assumendo che la costante della forma quadratica sia $c = \|x^*\|_A$):

$$\|x - x^*\|_A^2 = \|b - Ax\|_{A^{-1}}^2 = 2f(x).$$

Da questo, segue immediatamente che

$$\|x_k - x^*\|_A \leq \|w - x^*\|_A$$

per ogni $w \in x_0 + \mathcal{K}_k$. Quindi, x_k è il vettore più vicino (se misurato in norma energia) alla soluzione x_* rispetto a tutti gli altri vettori appartenenti allo spazio $x_0 + \mathcal{K}_k$. D'altro canto, il vettore w può essere scritto in generale come:

$$w = x_0 + \sum_{j=0}^{k-1} \gamma_j A^j r_0$$

per opportuni coefficienti $\gamma_j \in \mathbb{R}$. Quindi

$$w - x^* = x_0 + \sum_{j=0}^{k-1} \gamma_j A^j r_0 - x^* = x_0 - x^* - \sum_{j=0}^{k-1} \gamma_j A^{j+1}(x_0 - x^*) = p(A)(x_0 - x^*)$$

e il polinomio $p(z) \in \mathcal{P}_{k,p(0)=1}$, appartiene cioè all'insieme dei polinomi di grado k e omogenei ($p_k(0) = 1$), ed è definito da:

$$p(z) = 1 + \sum_{j=0}^{k-1} \gamma_j z^{j+1}.$$

Si vede immediatamente che:

$$\|x_k - x^*\|_A = \min_{p \in \mathcal{P}_{k,p(0)=1}} \|p(A)(x_0 - x^*)\|_A . \tag{3.39}$$

Usando la decomposizione spettrale della matrice A, scritta come:

$$A^j = U \Lambda^j U^T,$$

si ottiene:

$$p(A) = U p(\lambda) U^T.$$

Poichè:

$$\|p(A)x\|_A = \left\|A^{1/2} p(A) x\right\|_2 \leq \|p(A)\|_2 \left\|A^{1/2} x\right\|_2 = \|p(A)\|_2 \|x\|_A ,$$

si ottiene dalla (3.39):

$$\|x_k - x^*\|_A \leq \|x_0 - x^*\|_A \min_{p \in \mathcal{P}_{k,p(0)=1}} \max_{z \in \lambda(A)} |p(z)|,$$

dove $\lambda(A)$, chiamato lo spettro di A, è l'insieme di tutti gli autovalori di A. Da qui l'importante risultato che fornisce una maggiorazione dell'errore relativo alla k-esima iterazione di CG:

$$\frac{\|x_k - x^*\|_A}{\|x_0 - x^*\|_A} \leq \max_{z \in \lambda(A)} |p_k(z)|,$$

dove $p_k(z)$ è un qualsiasi polinomio di grado k con $p_k(0) = 1$.

Utilizzando polinomi scelti opportunamente, si possono ricavare alcune propretà importanti di CG. Per esempio, è possibile far vedere la proprietà di terminazione finita. Per fare ciò si prenda il polinomio così definito:

$$p^*(z) = \prod_{i=1}^{N} \frac{\lambda_i - z}{\lambda_i},$$

e osservando che tale polinomio ha radici in corrispondenza agli autovalori di A, si ottiene immediatamente:

$$\|x_N - x^*\|_A \leq \max_{z \in \lambda(A)} |p_k^*(z)| = 0.$$

Con ragionamento simile, si vede che se A ha $m < N$ autovalori distinti, CG converge in al massimo m iterazioni. Lo stesso accade nel caso in cui il termine noto sia una combinazione lineare di m autovettori della matrice A, e si parta da $x_0 = 0$. Dalla simmetria della matrice e dalla compatibilità della norma matriciale di Hilbert con la norma euclidea dei vettori, ricordando la (2.7), si ottiene immediatamente:

$$\frac{\|r_k\|_2}{\|r_0\|_2} = \frac{\|b - Ax_k\|_2}{\|b - Ax_0\|_2} = \frac{\|A(x_k - x^*)\|_2}{\|A(x_0 - x^*)\|_2} \leq \sqrt{\frac{\lambda_1}{\lambda_N}} \frac{\|x_k - x^*\|_A}{\|x_0 - x^*\|_A},$$

dove λ_1 e λ_N sono rispettivamente l'autovalore massimo e quello minimo della matrice A.

Usando i polinomi di Chebyshev, è possibile dimostrare il seguente risultato di convergenza dello schema CG:

$$\frac{\|x_k - x^*\|_A}{\|x_0 - x^*\|_A} \leq 2 \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^k,$$

dove $\kappa(A) = \lambda_1/\lambda_N$ è l'indice di condizionamento della matrice A. Il risultato mostra che il numero di iterazioni che CG impiega per arrivare a convergenza è proporzionale alla radice del numero di condizionamento di A. Possiamo quindi calcolare il numero m di iterazioni necessario per ridurre l'errore iniziale di un fattore ϵ. Infatti basta imporre che:

$$2 \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^m < \epsilon,$$

da cui, prendendo i logaritmi e notando che

$$\log\left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)+1}}\right) \sim \frac{1}{\sqrt{\kappa(A)}}$$

per valori grandi di $\kappa(A)$, si ottiene immediatamente:

$$m \geq \frac{1}{2}\sqrt{\kappa(A)}\log\frac{2}{\epsilon}.$$

Si noti che la convergenza di CG risulta più veloce di quella di SD per la presenza della radice del numero di condizionamento di A (si veda l'eq. (3.29)). Inoltre, la convergenza è accelerata se $\kappa(A)$ si avvicina a 1, ovvero se $\lambda_1 \approx \lambda_N$. Questa osservazione ci permette di cercare di migliorare la convergenza del metodo tramite la tecnica di "precondizionamento", secondo la quale si cerca di trasformare il sistema originale in uno equivalente che ha indice di condizionamento il più possibile vicino a 1.

Precondizionamento di CG: Metodo PCG

Il precondizionamento del metodo CG serve a ridurre il numero di condizionamento della matrice A, in modo da accelerare la convergenza dello schema. Lo scopo è quello di arrivare ad una soluzione accettabile (i.e., con un residuo sufficientemente piccolo) in un numero di iterazioni che generalmente è molto minore della dimensione della matrice. Per fare questo riscriviamo il problema iniziale premoltiplicandolo per una matrice P simmetrica e definita positiva, che approssimi A ma sia facile da invertire, come discusso nel paragrafo 3.1.1. Riportiamo qui l'eq. (3.10):

$$P^{-1}Ax = P^{-1}b.$$

In realtà, per migliorare la convergenza di CG applicato a questo sistema bisogna richiedere:

$$1 \leq \kappa(P^{-1}A) \ll \kappa(A), \tag{3.40}$$

(idealmente $\kappa(P^{-1}A) \approx 1$). Il problema fondamentale con quanto scritto è che la nuova matrice $B = P^{-1}A$ non è più simmetrica nè positiva definita, per cui non si può applicare il metodo CG.

Si può però procedere ricordandoci dell'esistenza delle trasformazioni di similitudine, discusse nel paragrafo 2.0.6. E' possibile infatti scrivere la matrice di precondizionamento P come prodotto di due matrici che risultino facili da invertire. In tal caso otteniamo:

$$\begin{aligned} P &= E^T E \qquad & P^{-1} &= E^{-T}E^{-1} \\ B &= E^{-1}AE^{-T} \qquad & \lambda(B) &= \lambda(P^{-1}A) \end{aligned}$$

e il nuovo sistema da risolvere si può scrivere quindi:

$$By = c \qquad \text{con} \qquad B = E^{-1}AE^{-T} \qquad y = E^T x \qquad c = E^{-1}b. \tag{3.41}$$

Tale sistema ha le stesse soluzioni di quello originale e B è simmetrica e definita positiva con un indice di condizionamento minore di quello di A (vedi la (3.40)). Dal punto di vista geometrico, possiamo dire che gli iperelissoidi che rappresentano le curve di livello della forma quadratica che andiamo a minimizzare saranno molto più vicine a delle ipersfere nel caso del sistema precondizionato, con evidenti vantaggi in termini di allineamento precoce della direzione di ricerca con il centro di tali iperelissoidi (e quindi il minimo della forma quadratica).

Applicando il metodo di CG al nuovo sistema (3.41), con qualche passaggio si arriva alle seguenti equazioni del "Preconditioned Conjugate Gradient" (PCG) algotihm:

ALGORITHM PCG
Input: $x_0, nimax, toll$; $k = 0$;
$r_0 = b - Ax_0$ $p_0 = P^{-1}r_0$
FOR $k = 0, 1, \ldots$ fino a convergenza:

1. $\alpha_k = \dfrac{\langle p_k, r_k\rangle}{\langle p_k, Ap_k\rangle}$ (3.42)
2. $x_{k+1} = x_k + \alpha_k p_k$ (3.43)
3. $r_{k+1} = r_k - \alpha_k Ap_k$ (3.44)
4. $\beta_k = -\dfrac{\langle P^{-1}r_{k+1}, Ap_k\rangle}{\langle p_k, Ap_k\rangle}$ (3.45)
5. $p_{k+1} = P^{-1}r_{k+1} + \beta_k p_k$ (3.46)

END FOR

Si noti che è sufficiente applicare il precondizionatore solamente una volta al vettore r_{k+1}, salvando il risultato per utilizzarlo poi sia in (3.45) che in (3.46).

La definizione del precondizionatore più efficiente per un dato problema è generalmente difficile e non esistono metodi generali. Per i sistemi che scaturiscono dalla discretizzazione (agli elementi finiti o alle differenze finite) di equazioni differenziali, si usano spesso due tipi di precondizionatori: il semplice precondizionatore diagonale di Jacobi e la decomposta Incompleta di Cholesky (IC(0)).

Il precondizionatore diagonale di Jacobi (Jacobi diagonal scaling) corrisponde alla scelta:

$$P = D \qquad p_{ij} = \begin{cases} a_{ij}, & \text{se } i = j, \\ 0, & \text{se } i \neq j. \end{cases},$$

ed è di facilissima applicazione, anche se la sua efficienza non è elevatissima. Questo precondizionatore corrisponde ad una iterazione del metodo di Jacobi studiato nel paragrafo 3.1.1.

Il precondizionatore più frequentemente usato, perchè spesso più efficiente, è dato dalla decomposta Incompleta di Cholesky senza fill-in. Tale precondizionatore è facilmente definito a partire dalla fattorizzazione di Cholesky della matrice $A = LL^T$ imponendo a priori lo stesso pattern di sparsità della matrice A. In altre parole, detti l_{ij} gli elementi della matrice (triangolare) fattorizzata di Cholesky, il precondizionatore IC(0) è dato da:

$$P = EE^T \qquad e_{ij} = \begin{cases} l_{ij}, & \text{se } a_{ij} \neq 0, \\ 0, & \text{se } a_{ij} = 0. \end{cases} .$$

Implementazione all'elaboratore del PCG

In realtà l'implementazione del metodo PCG si deve fare prendendo in considerazione il fatto che le operazioni più costose sono il prodotto matrice vettore e l'applicazione del precondizionatore ad un vettore, per cui tali operazioni vanno effettuate solo se necessarie. Per questo, si preferisce salvare i vettori risultanti e riutilizzarli opportunamente. L'algoritmo completo può quindi essere scritto cosi':

ALGORITHM PCG
Input: $A, P, b, x_0, nimax, toll$; $k = 1$;
$x = x_0$, $r = r_0 = b - Ax_0$, $\rho_0 = \|r\|_2^2$
DO WHILE $\sqrt{\rho_{k-1}} > toll\,\|b\|_2$ and $k < nimax$:

1. $z = P^{-1}r$
2. $\tau_{k-1} = z^T r$
3. if $k = 1$ then
   $\beta = 0$
   $p = z$
   else
   $\beta = \tau_{k-1}/\tau_{k-2}$
   $p = z + \beta p$
   end if
4. $w = Ap$
5. $\alpha = \tau_{k-1}/p^T w$
6. $x = x + \alpha p$
7. $r = r - \alpha w$
8. $\rho_k = r^T r$
9. $k = k + 1$

END FOR

Questo algoritmo prevede di memorizzare la matrice A, il precondizionatore P e 6 vettori: b, x, r, z, p e w.

Usando il precondizionatore IC(0), il punto 1. dell'algoritmo precedente viene svolto risolvendo il sistema lineare $Pz = (EE^T)z = r$ tramite una sostituzione in avanti ed una indietro:

$$\begin{aligned} Ey &= r \\ E^T z &= y. \end{aligned}$$

Metodo delle correzioni residue

Il metodo delle correzioni residue (CR) si utilizza spesso per calcolare una soluzione iniziale per il PCG. In pratica tale metodo è una modifica del metodo iterativo di Richardson (si veda il paragrafo 3.1.1) che utilizza come matrice di iterazione la matrice

$$E = (I - P^{-1}A),$$

per cui, partendo dalla soluzione iniziale $x_0 = 0$ si hanno le seguenti iterate:

$$\begin{aligned} x_0 &= P^{-1}b \\ x_1 &= x_0 + P^{-1}r_0 \\ x_2 &= x_1 + P^{-1}r_1 \\ \dots & \\ x_{k+1} &= x_k + P^{-1}r_k. \end{aligned}$$

Di solito una iterazione è sufficiente per eliminare tutte le componenti dell'errore corrispondenti ad autovettori unitari della matrice $P^{-1}A$. Infatti, l'errore dello schema precedente può essere scritto come:

$$e_{k+1} = (I - P^{-1}A)^k e_0.$$

Notando che la matrice $P^{-1}A$ è diagonalizzabile, si vede immediatamente che l'errore alla k-esima iterazione è dato da

$$e_k = \sum_{j=1}^{n} (1 - \lambda_j)^k cj v_j,$$

essendo $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N$ gli autovalori di $P^{-1}A$ ordinati in senso decrescente e i vettori v_j, $j = 1, \ldots, N$ i corrispondenti autovettori. Si vede subito che una iterazione del metodo CR azzera tutte le componenti dell'errore relative agli autovettori associati agli autovalori di $P^{-1}A$ unitari.

Esempi di applicazione

Si riportano qui di seguito due esempi di applicazione del metodo PCG precondizionato con IC(0) (la decomposta incompleta di Cholesky).

Il primo esempio riguarda una matrice di dimensioni $n = 28600$, derivante dalla discretizzazione spaziale tramite il metodo agli elementi finiti misti (ibridizzati) di un'equazione ellittica a coefficienti variabili nello spazio. La matrice è simmetrica e definita positiva ed è caratterizzata da un numero di elementi non nulli pari a 201200, circa lo 0.02% degli elementi totali (n^2). La soluzione è stata ottenuta con un processore Pentium 4 a 2 GHz in circa 10 secondi. Il pattern degli elementi non nulli è mostrato in Figura 3.14.

Il secondo esempio è relativo ad una matrice di dimensioni $n = 80711$, avente un numero di elementi non nulli pari a 432000, circa lo 0.07% degli elementi totali (n^2) (Figura 3.15 . La matrice è relativa ad una discretizzazione di un sistema di equazioni di diffusione tramite un metodo misto di elementi finiti alla Galerkin con elementi triangolari e un metodo alle differenze finite. La soluzione ha impiegato circa 5 secondi.

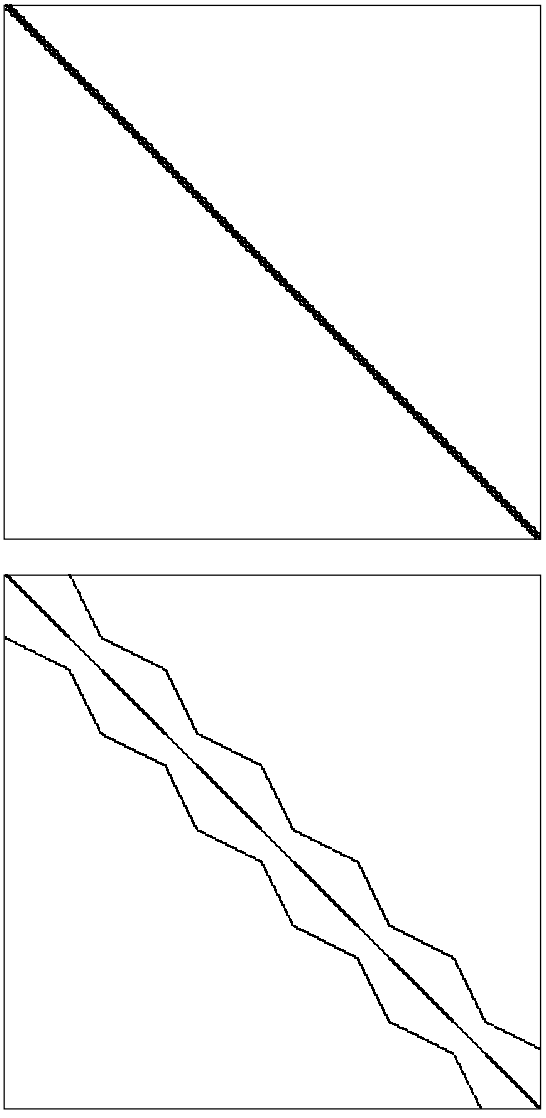

Figura 3.14: Pattern spaziali degli elementi non nulli della matrice $n = 28600$. A destra è disegnata l'intera matrice, a sinistra si riporta uno zoom del riquadro in alto a sinistra.

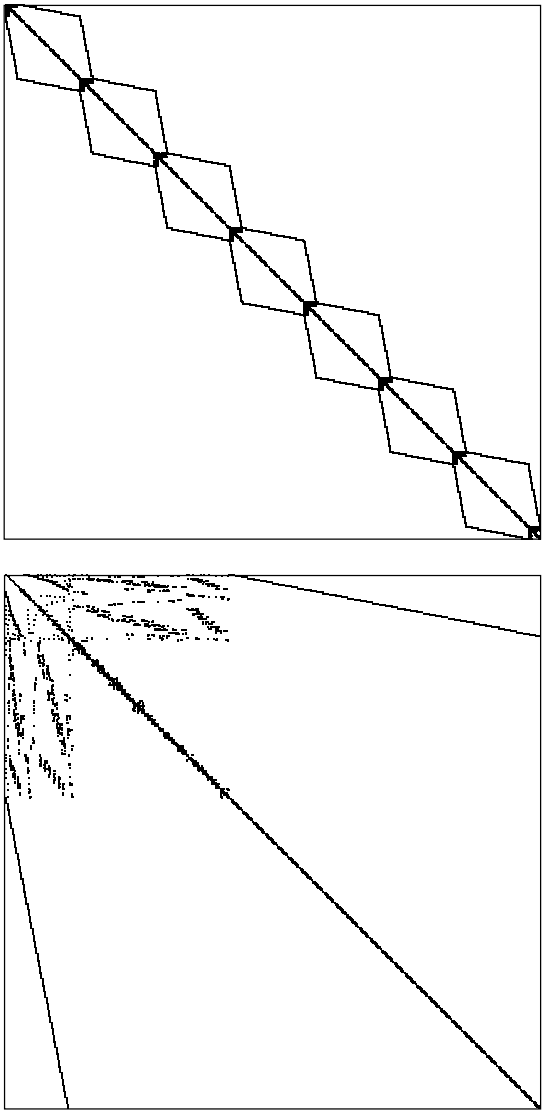

Figura 3.15: Pattern spaziali degli elementi non nulli della matrice $n = 80711$. A destra è disegnata l'intera matrice, a sinistra si riporta uno zoom del riquadro in alto a sinistra.

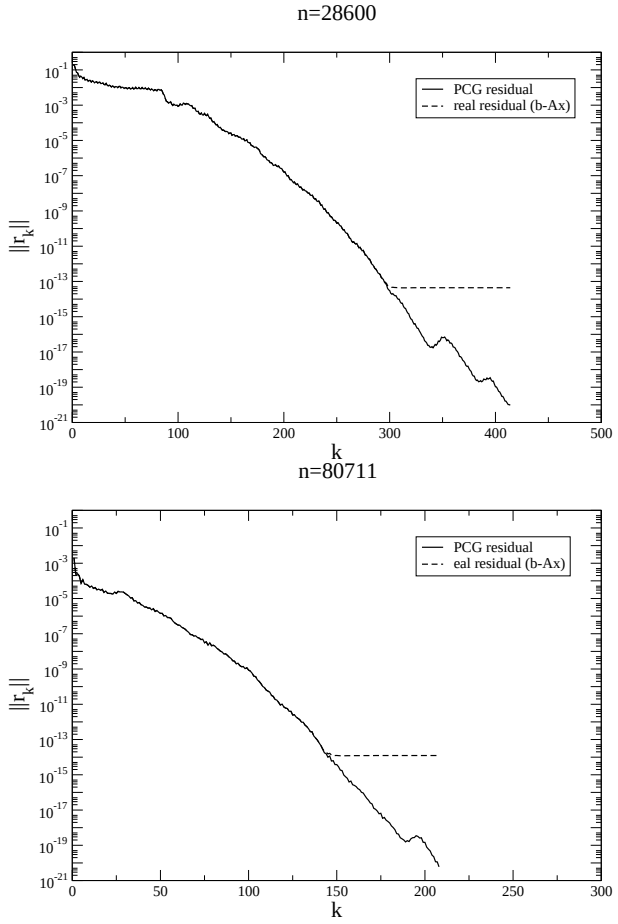

Figura 3.16: Profili di convergenza del metodo PCG con precondizionatore di Choleky per la matrice $n = 28600$ (sinistra) e $n = 80711$ (destra).

I profili di convergenza del metodo PCG per le due matrici sono riportati in Figura 3.16, a destra quello relativo alla prima matrice, e a sinistra quello relativo alla seconda. Si noti il comportamento del residuo calcolato tramite la formula ricorrente del PCG, che continua a decrescere, mentre il residuo vero si stabilizza alla precisione di macchina (circa 10^{-16}).

Implementazione pratica del metodo PCG

Per l'applicazione ottimale del PCG è necessario minimizzare i costi del prodotto matrice-vettore e dell'applicazione del precondizionatore. Ciò viene effettuato in general sfruttando la caratteristica di sparsità tipica delle matrici che derivano da discretizzazioni di equazioni differenziali e quindi memorizzando solamente gli elementi non nulli della matrice e effettuando le operazioni solo su di esssi.

Il modo in cui viene memorizzata la matrice A del sistema è detto Compressed Row Storage (CRS).

Caso non simmetrico. Data una generica matrice A quadrata di ordine n, la tecnica di memorizzazione CRS prevede la generazione di 3 vettori:

1. `SYSMAT`: vettore di numeri reali contenente gli nt coefficienti non nulli della matrice A memorizzati in successione per righe;

2. `JA`: vettore di numeri interi contenente gli nt indici di colonna dei corrispondenti elementi memorizzati in `SYSMAT`;

3. `IA`: vettore di numeri interi con $n+1$ componenti, contenente le posizioni in cui si trova in `SYSMAT` il primo elemento di ciascuna riga di A.

Il vettore `IA` è chiamato "vettore topologico" e talvolta indicato anche come `TOPOL`. L'uso congiunto di `IA` e `JA` consente di individuare qualsiasi elemento non nullo a_{ij} memorizzato in `SYSMAT`. Infatti, l'elemento a_{ij} si troverà in una posizione k del vettore `SYSMAT` compresa nell'intervallo $\texttt{IA}(i) \leq k \leq \texttt{IA}(i+1) - 1$ e tale per cui $\texttt{JA}(k) = j$. Queste due condizioni permettono di individuare univocamente k per cui $\texttt{SYSMAT}(k) = a_{ij}$. Si noti che l'occupazione di memoria si riduce da n^2 numeri reali (generalmente in doppia precisione) a nt numeri reali (tipicamente $nt < 0.05n^2$) e $nt + n + 1$ numeri interi.

Facciamo un esempio numerico per rendere pičhiaro il concetto. Si consideri la seguente matrice A di dimensione 6×6, sparsa e non simmetrica:

$$A = \begin{bmatrix} 4 & 0 & 7 & 0 & 0 & 0 \\ 3 & 4 & 0 & 2 & 0 & 0 \\ 0 & 2 & 4 & 0 & 4 & 0 \\ 0 & 0 & 7 & 4 & 0 & 1 \\ 1 & 0 & 0 & 3 & 4 & 0 \\ 1 & 1 & 0 & 0 & 3 & 4 \end{bmatrix}.$$

Gli elementi non nulli sono $nt = 18$, mentre $n^2 = 36$, pertanto il grado di sparsità di A è il 50%. Trascurando i coefficienti nulli, la matrice A diventa:

$$A' = \begin{bmatrix} 4 & & 7 & & & \\ 3 & 4 & & 2 & & \\ & 2 & 4 & & 4 & \\ & & 7 & 4 & & 1 \\ 1 & & & 3 & 4 & \\ 1 & 1 & & & 3 & 4 \end{bmatrix}.$$

Il vettore `SYSMAT` ha dimensione 18 e risulta:

$$\overbrace{\begin{array}{cccccccccccccccccc} 4 & 7 & 3 & 4 & 2 & 2 & 4 & 4 & 7 & 4 & 1 & 1 & 3 & 4 & 1 & 1 & 3 & 4 \end{array}}^{nt}.$$

Il vettore `JA` ha la stessa dimensione di `SYSMAT` e per ogni coefficiente di quest'ultimo individua l'indice di colonna:

$$\overbrace{\begin{array}{cccccccccccccccccc} 1 & 3 & 1 & 2 & 4 & 2 & 3 & 5 & 3 & 4 & 6 & 1 & 4 & 5 & 1 & 2 & 5 & 6 \end{array}}^{nt}.$$

Il vettore `IA`, infine, individua la posizione in `SYSMAT` del primo coefficiente di ciascuna riga:

$$\overbrace{\begin{array}{ccccccc} 1 & 3 & 6 & 9 & 12 & 15 & 19 \end{array}}^{n+1}.$$

Si noti che che le componenti di `IA` sono necessariamente ordinate in senso strettamente crescente e che $\texttt{IA}(n+1) = nt + 1$. Ciò si deduce immediatamente dal fatto che per individuare un elemento a_{ij} dell'ultima riga di A si deve cercare l'indice k della componente in `SYSMAT` nell'intervallo $\texttt{IA}(n) \leq k \leq \texttt{IA}(n+1) - 1$.

Caso simmetrico Nel caso in cui la matrice A sia simmetrica, come accade ad esempio nella soluzione agli elementi finiti dell'equazione differenziale ellittica di

Laplace, il risparmio computazionale derivato dal sistema CRS viene incrementato memorizzando la sola parte triangolare alta, inclusa la diagonale principale, di A. In altri termini, vengono memorizzati solo i coefficienti $a_{ij} \neq 0$ con $j \geq i$ e si sfrutta la proprietà A secondo cui $a_{ij} = a_{ji}$.

La memorizzazione di A viene effettuata sempre mediante i tre vettori `SYSMAT`, `JA` e `IA` definiti nel paragrafo precedente, in cui tuttavia nt rappresenta il numero di coefficienti non nulli della triangolare alta e `IA` contiene le posizioni in cui si trova in `SYSMAT` l'elemento diagonale di ciascuna riga di A.

Anche in questo caso facciamo un esempio numerico per rendere pičhiaro il concetto. Si consideri la seguente matrice A di dimensione 7×7, sparsa e simmetrica:

$$A = \begin{bmatrix} 10 & 0 & 0 & 0 & 1 & 0 & 2 \\ 0 & 1 & 0 & 3 & 0 & -1 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 & 1 \\ 0 & 3 & 0 & 2 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 5 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 20 \end{bmatrix}.$$

Gli elementi non nulli sono 19, mentre se fosse piena ne avremmo 49. Tuttavia i coefficienti da memorizzare sono solamente quelli relativi alla parte superiore di A, di seguito evidenziati:

$$A' = \begin{bmatrix} \boxed{10} & & & & \boxed{1} & & \boxed{2} \\ & \boxed{1} & & \boxed{3} & & \boxed{-1} & \\ & & \boxed{4} & & & & \boxed{1} \\ & 3 & & \boxed{2} & \boxed{1} & & \\ 1 & & & 1 & \boxed{5} & & \\ & -1 & & & & \boxed{1} & \\ 2 & & 1 & & & & \boxed{20} \end{bmatrix}.$$

Il vettore `SYSMAT` avrà perciò dimensione $nt = 13$ (7 elementi diagonali più 6 extra-diagonali) anziché 19, con un risparmio di memoria superiore al 30%, e sarà dato da:

$$\overbrace{10 \quad 1 \quad 2 \quad 1 \quad 3 \quad -1 \quad 4 \quad 1 \quad 2 \quad 1 \quad 5 \quad 1 \quad 20}^{nt}.$$

Il vettore `JA` con gli indici di colonna corrispondenti ha la stessa dimensione di `SYSMAT` e risulta:

$$\overbrace{1 \quad 5 \quad 7 \quad 2 \quad 4 \quad 6 \quad 3 \quad 7 \quad 4 \quad 5 \quad 5 \quad 6 \quad 7}^{nt}.$$

Infine, il vettore IA possiede sempre dimensione $n+1$ ed individua la posizione in SYSMAT degli elementi diagonali:

$$\overbrace{1 \quad 4 \quad 7 \quad 9 \quad 11 \quad 12 \quad 13 \quad 14}^{n+1}.$$

Ad esempio, IA(3)=7 significa che a_{33} è memorizzato in SYSMAT(7), come confermato anche dal fatto che JA(7)=3.

Come nel caso di matrici non simmetriche, si pone $\texttt{IA}(n+1) = nt+1$. Si noti anche che dovrà sempre essere IA(1)=1 e $\texttt{IA}(n) = nt$, nonché JA(1)=1 e $\texttt{JA}(nt) = n$.

Implementazione del prodotto matrice-vettore Lo schema del GCM necessita dell'operazione del prodotto matrice-vettore. L'implementazione di tale operazione al calcolatore risulta del tutto banale nel caso di memorizzazione tabellare della matrice, mentre è necessaria qualche attenzione qualora si utilizzi il sistema di rappresentazione CRS. Nel seguito distingueremo fra l'implementazione del prodotto matrice-vettore per matrici non simmetriche, più intuitivo, e per matrici simmetriche in cui si memorizza la sola triangolare alta.

Caso non simmetrico Si vuole calcolare il prodotto matrice-vettore:

$$Av = w \tag{3.47}$$

con A matrice quadrata non simmetrica di ordine n, v e w vettori in $\mathbb{R}^n$. Si ricorda che la componente i-esima di w è pari alla somma dei prodotti degli elementi della riga i di A per i corrispodenti elementi di v:

$$w_i = \sum_{j=1}^{n} a_{ij} v_j \tag{3.48}$$

Se la matrice è memorizzata in modo compatto, gli elementi a_{ij} vanno opportunamente ricercati in SYSMAT mediante l'uso di IA, mentre gli indici j relativi alle colonne si trovano nel vettore intero JA. In particolare, gli elementi di A appartenenti alla riga i sono memorizzati in corrispondenza agli indici k del vettore SYSMAT compresi, per definizione di IA, nell'intervallo $\texttt{IA}(i) \leq k \leq \texttt{IA}(i+1) - 1$. Gli indici di colonna j, di conseguenza, sono memorizzati in $\texttt{JA}(k)$.

Il prodotto matrice-vettore con A non simmetrica e memorizzata in forma compatta può quindi essere calcolato implementando il seguente algoritmo:

001 **Per** $i = 1, n$
002 azzero w_i

003 **Per** $k = \texttt{IA}(i), \texttt{IA}(i+1) - 1$
004 $j :=\texttt{JA}(k)$
005 $w_i := w_i + \texttt{SYSMAT}(k) \cdot v_j$
006 **Fine Per**
007 **Fine Per**

Si noti che è sempre utile azzerare il vettore soluzione prima di procedere all'implementazione del ciclo di sommatoria (riga 2), al fine di evitare l'uso improprio di valori precedentemente contenuti in w.

Caso simmetrico Si vuole ora calcolare il prodotto (3.47) in cui la matrice simmetrica A è memorizzata in formato CRS come descritto prima (e quindi memorizzando solo la parte triangolare. Poiché gli elementi a_{ij} con $j < i$ non sono ora immediatamente disponibili nella sequenza di coefficienti della riga i, conviene scrivere la definizione (3.48) come:

$$w_i = \sum_{j=1}^{i-1} a_{ji} v_j + \sum_{j=i}^{n} a_{ij} v_j \tag{3.49}$$

avendo sfruttato la condizione per cui $a_{ij} = a_{ji}$. Dalla (3.49) si deduce che il contributo a w_i relativo agli elementi con $j \geq i$ può essere implementato in maniera del tutto analoga a quanto fatto nel paragrafo precedente, mentre il contributo relativo agli elementi con $j < i$ può essere determinato selezionando in `SYSMAT` le componenti k per cui $\texttt{JA}(k) = i$, cioè appartenenti alla colonna i-esima. E' del tutto evidente, tuttavia, che questo modo di procedere, seppure intuitivo, risulta inapplicabile da un punto di vista pratico. Sarebbe, infatti, necessario procedere ad una ricerca su nt componenti per n volte, con un dispendio che renderebbe inutile il vantaggio fornito dalla memorizzazione compatta della matrice e, in certi casi, dalla convergenza accelerata del GCM.

Conviene osservare che gli elementi della triangolare alta della riga i, oltre a contribuire al calcolo di w_i, entrano in gioco, in virtdella simmetria di A, anche nel calcolo di w_j, con j pari all'indice di colonna dell'elemento considerato. All'interno del medesimo ciclo sulle righe i si aggiornerà pertanto non solo w_i ma anche tutti i w_j corrispondenti. L'algoritmo descritto nel precedente paragrafo viene quindi modificato nel modo seguente:

001 **Per** $i = 1, n$
002 azzero w_i
003 **Fine Per**

```
004     Per i = 1, n
005         k := IA(i)
006         w_i := w_i + SYSMAT(k) · v_i
007         Per k = IA(i) + 1, IA(i + 1) − 1
008             j := JA(k)
009             w_i := w_i + SYSMAT(k) · v_j
010             w_j := w_j + SYSMAT(k) · v_i
011         Fine Per
012     Fine Per
```

Si noti che, a differenza dell'algoritmo utilizzato per matrici non simmetriche, in questo caso l'azzeramento del vettore prodotto w va fatto con un ulteriore ciclo (righe 1-3) esterno a quello di calcolo di w. Inoltre, il contributo a w_i dato dall'elemento diagonale a_{ii} viene conteggiato a parte (riga 6) per evitare di considerarlo due volte nel ciclo successivo (corrisponde infatti al caso in cui $i = j$).

Calcolo del precondizionatore

Una delle chiavi del successo del GCM nella soluzione efficiente di sistemi lineari sparsi, simmetrici e definiti positivi sta nella possibilità di ottenere formidabili accelerazioni della convergenza mediante l'uso di opportune matrici di precondizionamento. La scelta di K^{-1} deve soddisfare alle seguenti caratteristiche:

- deve essere tale che il prodotto $P^{-1}A$ abbia lo spettro, cioè l'insieme degli autovalori, raggruppato attorno all'unità e comunque un numero di condizionamento spettrale inferiore a quello di A;
- il calcolo deve essere semplice e poco costoso per non appesantire lo schema;
- l'occupazione di memoria deve essere inferiore o al più paragonabile a quella della matrice del sistema allo scopo di non vanificare lo sforzo computazionale effettuato per la memorizzazione compatta.

Spesso le suddette caratteristiche confliggono fra loro e necessariamente la scelta di P^{-1} diventa il risultato di un compromesso. Per paradosso, la matrice che soddisfa completamente la prima richiesta è ovviamente A^{-1}, il cui costo ed occupazione di memoria (si ricordi che l'inversa di una matrice sparsa è generalmente una matrice piena) tuttavia rendono del tutto incalcolabile.

Una discreta accelerazione del metodo del GC è ottenuta scegliendo:

$$P^{-1} = D^{-1} \tag{3.50}$$

dove D è la matrice diagonale contenente gli elementi diagonali di A. In questo caso, il calcolo della matrice di precondizionamento e la sua applicazione nello schema del GCM risultano banali e vengono lasciati per esercizio al lettore. Il raggruppamento degli autovalori attorno all'unità di $D^{-1}A$, tuttavia, può essere limitato, specialmente se A presenta coefficienti extra-diagonali grandi rispetto agli elementi diagonali.

Risultati assai più significativi sono invece ottenuti assumendo:

$$P^{-1} = \left(\tilde{L}\tilde{L}^T\right)^{-1} \tag{3.51}$$

dove $\tilde{L}$, matrice triangolare bassa, è calcolata mediante la fattorizzazione incompleta di A, cioé la decomposta di Cholesky a cui viene assegnato il medesimo schema di sparsità di A. L'uso di questa matrice di precondizionamento, proposto negli anni '70 da Kershaw, si rivela generalmente un ottimo compromesso fra le contrapposte esigenze precedentemente discusse. Il calcolo di P^{-1} secondo la (3.51) e la sua applicazione nell'algoritmo del GCM verranno esaminati nei due paragrafi seguenti.

Fattore incompleto di Cholesky secondo Kershaw Il calcolo del fattore incompleto di Cholesky si basa sulla fattorizzazione triangolare di matrici simmetriche che viene di seguito brevemente richiamata. Poichè con la memorizzazione compatta di A vengono conservati i soli elementi appartenenti alla triangolare alta, conviene riscrivere la (3.51) come:

$$P^{-1} = \left(\tilde{U}^T\tilde{U}\right)^{-1} \tag{3.52}$$

dove $\tilde{U} = \tilde{L}^T$. Supponiamo che A sia una matrice 3×3 piena di cui svolgiamo per esteso la fattorizzazione:

$$\begin{bmatrix} u_{11} & 0 & 0 \\ u_{12} & u_{22} & 0 \\ u_{13} & u_{23} & u_{33} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{12} & a_{22} & a_{23} \\ a_{13} & a_{23} & a_{33} \end{bmatrix} \tag{3.53}$$

Si noti che nella (3.53) si è sfruttata la simmetria di A ed il fatto che i soli elementi a_{ij} memorizzati sono quelli per cui $j \geq i$.

Sviluppiamo il prodotto (3.53) procedendo secondo la successione delle righe di A. Per la prima riga vale:

$$\begin{aligned} a_{11} = u_{11}^2 \quad &\Rightarrow \quad u_{11} = \sqrt{a_{11}} \\ a_{12} = u_{11}u_{12} \quad &\Rightarrow \quad u_{12} = \frac{a_{12}}{u_{11}} \\ a_{13} = u_{11}u_{13} \quad &\Rightarrow \quad u_{13} = \frac{a_{13}}{u_{11}} \end{aligned} \tag{3.54}$$

Si può osservare che i coefficienti sottodiagonali di A vengono di volta in volta già utilizzati nel calcolo delle righe superiori a quella corrente. Si procede, pertanto, con le righe 2 e 3 considerando i soli termini relativi alla triangolare alta di A:

$$\begin{aligned} a_{22} = u_{12}^2 + u_{22}^2 \quad &\Rightarrow \quad u_{22} = \sqrt{a_{22} - u_{12}^2} \\ a_{23} = u_{12}u_{13} + u_{22}u_{23} \quad &\Rightarrow \quad u_{23} = \frac{1}{u_{22}}\left(a_{23} - u_{12}u_{13}\right) \\ a_{33} = u_{13}^2 + u_{23}^2 + u_{33}^2 \quad &\Rightarrow \quad u_{33} = \sqrt{a_{33} - u_{13}^2 - u_{23}^2} \end{aligned} \tag{3.55}$$

Dalle (3.54) e (3.55) si può facilmente generalizzare l'algoritmo di calcolo del fattore completo di Cholesky per una matrice di ordine n:

001 $u_{11} := \sqrt{a_{11}}$
002 **Per** $j = 2, n$
003 $\quad u_{1j} := a_{1j}/u_{11}$
004 **Fine Per**
005 **Per** $i = 2, n$
006 $\quad u_{ii} := \sqrt{a_{ii} - \sum_{l=1}^{i-1} u_{li}^2}$
007 $\quad$ **Per** $j = i + 1, n$
008 $\quad\quad u_{ij} := \left(a_{ij} - \sum_{l=1}^{i-1} u_{li}\, u_{lj}\right)/u_{ii}$
009 $\quad$ **Fine Per**
010 **Fine Per**

Il passaggio concettuale dal fattore completo di Cholesky a quello incompleto secondo Kershaw risulta a questo punto banale, in quanto all'algoritmo precedente è sufficiente aggiungere l'assegnazione:

$$a_{ij} = 0 \Rightarrow u_{ij} := 0 \tag{3.56}$$

Poiché A è memorizzata in modo compatto, e così anche $\tilde{U}$, si deduce immediatamente che, in virtù dell'assegnazione (3.56), i vettori `IA` e `JA` descrivono anche la topologia di $\tilde{U}$ per la quale sarà sufficiente adottare un vettore `PREC` contenente i coefficienti non nulli memorizzati sequenzialmente per righe.

Il nuovo algoritmo di calcolo del fattore incompleto di Cholesky secondo Kershaw con la memorizzazione in formato CRS risulta pertanto:

001 `PREC`(1) := $\sqrt{$`SYSMAT`(1)$}$
002 **Per** $k =$ `IA`(1)+1, `IA`(2)-1

003 $\quad$ $\texttt{PREC}(k) := \texttt{SYSMAT}(k)/\texttt{PREC}(1)$
004 **Fine Per**
005 **Per** $i = 2, n$
006 $\quad$ $k := \texttt{IA}(i)$
007 $\quad$ $l :=$ ogni $k < \texttt{IA}(i)$ per cui $\texttt{JA}(k) = i$
008 $\quad$ $\texttt{PREC}(k) := \sqrt{\texttt{SYSMAT}(k) - \sum_l \texttt{PREC}(l)^2}$
009 $\quad$ **Per** $k1 = \texttt{IA}(i) + 1, \texttt{IA}(i+1) - 1$
010 $\quad\quad$ $j := \texttt{JA}(k1)$
011 $\quad\quad$ $l1 :=$ ogni $k < \texttt{IA}(i)$ per cui $\texttt{JA}(k) = i$
012 $\quad\quad$ $l2 :=$ ogni $k < \texttt{IA}(i)$ per cui $\texttt{JA}(k) = j$
013 $\quad\quad$ $\texttt{PREC}(k1) := \left(\texttt{SYSMAT}(k1) - \sum_{l1,l2} \texttt{PREC}(l1) \cdot \texttt{PREC}(l2)\right) / \texttt{PREC}(k)$
014 $\quad$ **Fine Per**
015 **Fine Per**

Va osservato, tuttavia, che l'implementazione efficiente delle sommatorie in riga 8 e 13 sugli indici l, $l1$ ed $l2$ è in realtà non banale e va preferibilmente effettuata calcolando separatamente il contributo del coefficiente relativo al k corrente in modo da evitare le dispendiose ricerche previste in riga 7, 11 e 12. Tale implementazione è effettuata nella subroutine `KERSH` a disposizione dello studente.

Si deve infine ricordare che l'uso generalizzato delle (3.54) e (3.55) comporta l'estrazione di radici quadrate il cui argomento, nel caso di una fattorizzazione incompleta, non è più garantito essere positivo. In questo caso l'elemento diagonale in questione verrà posto pari ad un numero positivo arbitrario (ad esempio, l'ultimo coefficiente diagonale di $\tilde{U}$ non nullo). Si dimostra comunque teoricamente che se la matrice A è di tipo M, come ad esempio si verifica nella discretizzazione agli elementi finiti dell'equazione ellittica di Laplace se si utilizzano funzioni di base lineari definite su una triangolazione di Delaunay, tutte le radici da calcolare nel fattore incompleto esistono nel campo reale.

Applicazione della decomposta incompleta Mediante l'algoritmo sviluppato nel precedente paragrafo non si calcola esplicitamente la matrice di precondizionamento P^{-1} ma solamente il fattore incompleto $\tilde{U}$. E' quindi necessario sviluppare un algoritmo ad hoc che permetta di calcolare il generico prodotto $P^{-1}v$ senza generare esplicitamente P^{-1}. Sia quindi w il vettore in $\mathbb{R}^n$ risultato del prodotto $P^{-1}v$. Per definizione di P^{-1} si ha:

$$w = \left(\tilde{U}^T\tilde{U}\right)^{-1} v \tag{3.57}$$

cioè, premoltiplicando per P ambo i membri:

$$\left(\tilde{U}^T\tilde{U}\right) w = v \tag{3.58}$$

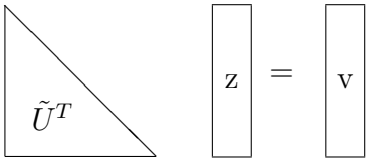

Figura 3.17: Schema della soluzione del sistema con sostituzioni in avanti.

Il calcolo di w viene quindi ricondotto alla soluzione di un sistema lineare la cui matrice è P. Poiché P è fattorizzabile nel prodotto di due matrici triangolari, il sistema (3.58) è risolvibile tramite sostituzioni in avanti e all'indietro. Posto:

$$\tilde{U}w = z \tag{3.59}$$

la (3.58) diventa:

$$\tilde{U}^T z = v \tag{3.60}$$

che si può facilmente risolvere con sostituzioni in avanti (Figura 3.17). Iniziando dalla prima componente, ricavata in modo immediato come:

$$z_1 = \frac{v_1}{u_{11}} \tag{3.61}$$

si ottiene con semplici calcoli la formula ricorrente:

$$z_i = \frac{1}{u_{ii}} \left(v_i - \sum_{j=1}^{i-1} u_{ji} z_j \right) \qquad i = 2, \ldots, n \tag{3.62}$$

Come osservato per il prodotto matrice-vettore e per la determinazione del fattore incompleto di Cholesky, la sommatoria contenuta in (3.62) non è banale da implementare in modo efficiente memorizzando le matrici secondo il sistema CRS. A tal proposito, risulta conveniente definire un vettore di accumulazione s nelle cui componenti viene aggiornato il prodotto contenuto nella sommatoria dell'equazione (3.62) procedendo per righe di $\tilde{U}$. L'algoritmo per il calcolo del vettore z può pertanto essere scritto come:

001 **Per** $j = 1, n$
002 azzero s_j
003 **Fine Per**
004 $z_1 := v_1/\texttt{PREC}(1)$

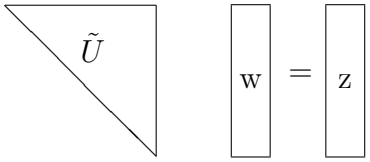

Figura 3.18: Schema della soluzione del sistema con sostituzioni all'indietro.

005 **Per** $i = 2, n$
006 $k := \texttt{IA}(i)$
006 **Per** $k1 = \texttt{IA}(i-1)+1,\ \texttt{IA}(i)-1$
007 $j := \texttt{JA}(k1)$
008 $s_j := s_j + \texttt{PREC}(k1) \cdot z_{i-1}$
009 **Fine Per**
010 $z_i := (v_i - s_i)\,/\texttt{PREC}(k)$
011 **Fine Per**

Ottenuto il vettore z si può infine calcolare w risolvendo il sistema (3.59) tramite sostituzioni all'indietro (Figura 3.18). La formula ricorrente si ricava in modo del tutto analogo a quanto fatto nelle equazioni (3.61) e (3.62) partendo in questo caso dalla componente n-esima:

$$w_n = \frac{z_n}{u_{nn}} \tag{3.63}$$

$$w_i = \frac{1}{u_{ii}} \left(z_i - \sum_{j=i+1}^{n} u_{ij} w_j \right) \qquad i = n-1, \ldots, 1 \tag{3.64}$$

L'implementazione della (3.18) risulta stavolta molto semplice anche memorizzando la matrice $\tilde{U}$ in forma compatta. Sempre utilizzando il vettore di accumulo s, l'algoritmo corrispondente può essere scritto nel modo seguente:

001 $w_n := z_n/\texttt{PREC}(nt)$
002 **Per** $i = n-1, 1$ con passo -1
003 azzero s_i
004 $k := \texttt{IA}(i)$
005 **Per** $k1 = \texttt{IA}(i)+1,\ \texttt{IA}(i+1)-1$
006 $j := \texttt{JA}(k1)$
007 $s_i := s_i + \texttt{PREC}(k1) \cdot w_j$

008 **Fine Per**
009 $w_i := (z_i - s_i)$ /PREC(k)
010 **Fine Per**

Per un confronto, viene messa a disposizione dello studente la subroutine LSOLVE che implementa gli algoritmi soprariportati.

4.8 La tecnica del precondizionamento nei metodi iterativi

La convergenza dei metodi iterativi per soluzione di sistemi lineari come di quelli per il calcolo di autovalori e autovettori è generalmente funzione del numero di condizionamento della matrice, il quale è a sua volta funzione degli auovalori di modulo massimo e minimo. Per diminuire il numero di condizionamento della matrice si usa la tecnica del "precondizonamento", che trasformare il problema in uno equivalente avente "comportamento spettrale" migliore.

4.8.1 Sistemi lineari

Metodo del Gradiente coniugato

Seguendo l'idea precedente, si vuole usare il metodo del gradiente coniugato per risolvere il sistema precondizionato:

$$By = c$$

ottenuto premoltiplicando il sistema lineare originale per l'inverso del precondizionatore P, e cioè:

$$P^{-1}Ax = P^{-1}b \qquad By = c \qquad \text{con } B = P^{-1}A, y = x, c = P^{-1}b,$$

con l'idea che:

$$1 \leq \kappa(P^{-1}A) \ll \kappa(A).$$

Procedendo in questo modo, non si mantengono le caratteristiche di simmetria e definizione positiva della matrice originaria A. Si potrebbe procedere assumendo $P = EE^T$ e lavorando quindi con la matrice $B' = E^{-T}AE^{-1}$ che è simile alla matrice B e quindi con lo stesso indice di condizionamento spettrale. Si dovrà quindi risolvere il sistema:

$$B'y = c' \qquad \text{con } B' = E^{-T}AE^{-1}, y = Ex, c = E^{-T}b.$$

Per scrivere le equazioni dell'algoritmo del Gradiente Coniugato Precondizionato (PCG) si può però procedere in maniera alternativa. Si può osservare il seguente:

Lemma 4.8.1. *La matrice $P^{-1}A$ è autoaggiunta rispetto al prodotto scalare $\langle \cdot, \cdot \rangle_P$.*

Dimostrazione. La dimostrazione è una semplice verifica diretta:

$$\begin{aligned} \left\langle P^{-1}Ax, y \right\rangle_P &= \left\langle PP^{-1}Ax, y \right\rangle = \langle Ax, y \rangle = \\ &= \left\langle x, PP^{-1}Ay \right\rangle = \left\langle x, P^{-1}, Ay \right\rangle_P . \end{aligned}$$

Lo stesso procedimento si può fare per $\langle \cdot, \cdot \rangle_A$. □

Il gradiente coniugato precondizionato equivale quindi ad applicare il metodo del gradiente coniugato utilizzando il prodotto scalare $\langle \cdot, \cdot \rangle_P$. Si denoti con r_k il residuoo del sistema lineare originale, $r_k (= b - Ax_k)$, e con z_k quello precondizionato, $z_k = P^{-1} r_k$. Si ha immediatamente che:

$$\langle z_k, z_k \rangle_P = \left\langle P^{-1} r_k, r_k \right\rangle .$$

Ad ogni passo del PCG si vuole minimizzare la forma quadratica:

$$f(x) = \frac{1}{2} \left\langle x, P^{-1} A x \right\rangle_P - \left\langle x, P^{-1} b \right\rangle_P + c,$$

ottenendo:

$$\alpha_k = \frac{\langle z_k, z_k \rangle_P}{\langle P^{-1} A p_k, p_k \rangle_P} = \frac{\langle P^{-1} r_k, r_k \rangle}{\langle A p_k, p_k \rangle};$$

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

$$r_{k+1} = r_k - \alpha_k A p_k.$$

La condizione di A-ortogonalità diventa ora:

$$\left\langle P^{-1} A p_k, p_k + 1 \right\rangle_P = 0 \Longrightarrow \beta_k = - \frac{\langle P^{-1} r_k, A p_k \rangle}{\langle p_k, A p_k \rangle};$$

e infine

$$p_{k+1} = P^{-1} r_{k+1} + \beta_k p_k.$$

L'algoritmo PCG si può quindi scrivere come (si veda anche pag. 71):

ALGORITHM PCG
Input: $x_0, nimax, toll$; $k = 0$;
$r_0 = b - Ax_0$ $p_0 = P^{-1} r_0$
FOR $k = 0, 1, \ldots$ fino a convergenza:

1. $\alpha_k = \dfrac{\langle p_k, r_k \rangle}{\langle p_k, A p_k \rangle}$
2. $x_{k+1} = x_k + \alpha_k p_k$
3. $r_{k+1} = r_k - \alpha_k A p_k$
4. $\beta_k = - \dfrac{\langle P^{-1} r_{k+1}, A p_k \rangle}{\langle p_k, A p_k \rangle}$
5. $p_{k+1} = P^{-1} r_{k+1} + \beta_k p_k$

END FOR

Metodo GMRES

Nel caso di sistema non simmetrico, si deve distinguere tra applicazione del precondizionatore sinistra ("left") o destra ("right") o divisa ("split"). Possiamo infatti riscrivere il sistema precondizionato nei seguenti modi:

$$\text{Left} \quad P_L^{-1}Ax = P_L^{-1}b \tag{4.11}$$

$$\text{Right} \quad AP_R^{-1}P_Rx = b \tag{4.12}$$

$$\text{Split} \quad P_L^{-1}AP_R^{-1}P_Rx = P_L^{-1}b \tag{4.13}$$

Con il metodo "Left" si arriva immediatamente al seguente algoritmo:

ALGORITHM PGMRES(m) (left)
Input: A, $r_0 = P^{-1}(b - Ax_0)$, $\beta = \|r_0\|_2$, $v_1 r_0/\beta$;
FOR $j = 1, 2, \ldots, m$:

1. $w = P^{-1}Av_j$;
2. FOR $i = 1, 2, \ldots, j$:
   $h_{ij} = \langle w, v_i \rangle$,
   $w = w - h_{ij}v_i$;
   END FOR
3. $h_{j+1,j} = \|w_j\|_2$;
4. $v_{j+1} = w/h_{j+1,j}$.

END FOR

$$V_m = [v_1, \ldots, v_m]$$

$$H_m = \{h_{ij}\}_{i=1,\ldots,j+1;j=1,\ldots,m}$$

$$y_m = \text{argmin}_y \|\beta e_1 - H_m y\|_2$$

$$x_m = x_0 + V_m y_m$$

Se convergenza raggiunta STOP altrimenti ricomincia con $x_0 = x_m$.

Lo spazio di Krylov minimizzato dalla procedura di Arnoldi è:

$$\mathcal{K} = \text{span}\left\{r_0, P^{-1}Ar_0, \ldots, \left(P^{-1}A\right)^{m-1} r_0\right\}.$$

Con il precondizionatore "Right", poniamo $y = Px$ e $AP^{-1}y = b$. Di conseguenza il residuo iniziale è $r_0 = b - Ax_0 = b - AP^{-1}y_0$. Alla fine del ciclo di Arnoldi si avrà dunque:

$$y_m = y_0 + V\eta = y_0 + \sum_{i=1}^{m} v_i\eta_i.$$

ovvero:

$$x_m = x_0 + P^{-1} \sum_{i=1}^{m} v_i \eta_i,$$

da cui scaturisce il seguente algoritmo:

ALGORITHM PGMRES(m) (right)
Input: A, $r_0 = P^{-1}(b - Ax_0)$, $\beta = \|r_0\|_2$, $v_1 r_0/\beta$;
FOR $j = 1, 2, \ldots, m$:

1. $w = AP^{-1}v_j$;
2. FOR $i = 1, 2, \ldots, j$:
   $h_{ij} = \langle w, v_i \rangle$,
   $w = w - h_{ij} v_i$;
   END FOR
3. $h_{j+1,j} = \|w_j\|_2$;
4. $v_{j+1} = w/h_{j+1,j}$.

END FOR

$$V_m = [v_1, \ldots, v_m]$$
$$H_m = \{h_{ij}\}_{i=1,\ldots,j+1;j=1,\ldots,m}$$
$$y_m = \text{argmin}_y \|\beta e_1 - H_m y\|_2$$
$$x_m = x_0 + P^{-1} V_m y_m$$

Se convergenza raggiunta STOP altrimenti ricomincia con $x_0 = x_m$.

Lo spazio di Krylov minimizzato dalla procedura di Arnoldi è:

$$\mathcal{K} = \text{span}\left\{r_0, AP^{-1}r_0, \ldots, \left(AP^{-1}\right)^{m-1} r_0\right\}$$

Il PGMRES "split" usa una combinazione dei due approcci.

E' chiaro da queste note le applicazioni "left" e "right" definiscono residui che sono scalati diversamente. In particolare, il PGMRES "left" itera utilizzando il residuo scalato con il precondizionatore:

$$r_m = z_m = P^{-1}(b - Ax_m),$$

mentre il PGMRES "right" usa il residuo non scalato:

$$r_m = (b - AP^{-1}y_m) = (b - Ax_m).$$

Di ciò dovrà essere necessariamente tenuto conto quando si confronta il residuo con la tolleranza per la verifica di convergenza dell'iterazione.

A parità di matrice precondizionante P le differenze sull'efficienza, e quindi sul numero di iterazioni sono minime, e si evidenziano quasi esclusivamente per matrici A assai malcondizionate.

In ogni caso possiamo apprezzare meglio le differenza guardando alle proprietà di minimizzazione. In particolare, l'approccio "left" risolve il seguente problema di minimo:

$$\min_{x \in x_0 + \mathcal{K}_m^L} \left\| P^{-1}(b - Ax) \right\|_2,$$

dove

$$\mathcal{K}_m^L = \text{span}\left\{ z_0, P^{-1}Az_0, \ldots, (P^{-1}A)^{m-1} z_0 \right\}$$

e $z_0 = P^{-1}r_0$. La soluzione approssimata è data quindi da:

$$x_m = x_0 + P^{-1} s_{m-1}\left(P^{-1}A\right) z_0.$$

dove s_{m-1} è il polinomio di grado $m-1$ che minimizza

$$\left\| z_0 - P^{-}1As\left(P^{-1}A\right) z_0 \right\|_2,$$

ovvero nelle variabili originali:

$$\left\| P^{-1}\left(r_0 - As\left(P^{-1}A\right) P^{-1} r_0\right) \right\|_2.$$

Il PGMRES "right" invece minimizza la norma del residuo non scalato:

$$\min_{y \in y_0 + \mathcal{K}_m^R} \left\| b - AP^{-1}y \right\|_2 = \left\| r_m \right\|_2,$$

dove:

$$\mathcal{K}_m^R = \text{span}\left\{ r_0, AP^{-1}r_0, \ldots, (AP^{-1})^{m-1} r_0 \right\}.$$

Premoltiplicando la precedente per P^{-1} si vede subito che la soluzione approssimata è data da:

$$x_m = x_0 + P^{-1} s_{m-1}\left(AP^{-1}\right) r_0.$$

dove s_{m-1} è il polinomio di grado $m-1$ che minimizza

$$\left\| r_0 - AP^{-1}s\left(P^{-1}A\right) r_0 \right\|_2.$$

4.8.2 Calcolo di autovalori e autovettori

Prendiamo ad esempio il precondizionamente per il metodo DACG. Si ricorda che l'applicazione di tale metodo equivale a risolvere on il metodo PCG il problema

$$\begin{cases} (A-\lambda_n I)e_k &= r_k \perp u_n \\ \langle e_k, u_n\rangle &= 0 \end{cases},$$

e che la convergenza dello schema CG per la minimizzazione del quoziente di Rayleigh dipende dal numero di condizionamento della matrice del sistema, e cioè:

$$k_0 = \frac{\lambda_{\max}(A-\lambda_n I)}{\lambda_{\min_{(A-\lambda_n I)}}} = \frac{\lambda_1-\lambda_n}{\lambda_{n-1}-\lambda_n}.$$

La tecnica del precondizionamento, anche in questo caso, vuole trasformare il problema $Au=\lambda u$ in un problema modificato equivalente $\tilde{A}\tilde{u}=\tilde{\lambda}\tilde{u}$, tale che

$$k\left(\tilde{A}-\tilde{\lambda}I\right) \ll k\left(A-\lambda I\right).$$

Scriviamo quindi $y=Ex$, con E matrice non singolare. Il quoziente di Rayleigh vale:

$$\begin{aligned} q(x) &= \frac{\langle x, Ax\rangle}{\langle x,x\rangle} = \frac{\langle E^{-1}y, AE^{-1}y\rangle}{\langle y,y\rangle} \\ &= \frac{\left\langle y, E^{-T}AE^{-1}y\right\rangle}{\langle y,y\rangle} = \tilde{q}(y). \end{aligned}$$

La matrice precondizionata diventa quindi:

$$\tilde{A}-\tilde{\lambda_n}I = E^{-T}(A-\lambda_n I)E^{-1} = (E^TE)^{-1})(A-\lambda_n I) = P^{-1}(A-\lambda_n I).$$

Nel caso ideale in cui $P=A$, otteniamo allora:

$$A^{-1}(A-\lambda_n I)u_j = (I-\lambda_n A^{-1})u_j = (1-\frac{\lambda_n}{\lambda_j})u_j \qquad \text{con} \qquad 0 \le 1-\frac{\lambda_n}{\lambda_j} \le 1.$$

Assumendo quindi $\lambda_1 \ge \lambda_2 \ge \ldots \ge \lambda_n$. si ottiene subito:

$$k_1 = k\left(A^{-1}(A-\lambda_n I)\right) = \frac{1-\lambda_n/\lambda_1}{1-\lambda_n/\lambda_{n-1}} = \frac{\lambda_{n-1}}{\lambda_1}k\left(A-\lambda_n I\right).$$

Nel caso molto frequente in cui $\lambda_n \ll \lambda_1$, si ha:

$$k_1 = \frac{1-\lambda_n/\lambda_1}{1-\lambda_n/\lambda_{n-1}} \approx \frac{1}{1-\lambda_n/\lambda_{n-1}} = \frac{\lambda_{n-1}}{\lambda_{n-1}-\lambda_n},$$

che mostra che $P=A$ è il precondizionatore "ottimale" perchè in questo caso la convergenza del DACG è indipendente da λ_1, e quindi dalla mesh. Ovviamente, prendere $P=A$ equivale a risolvere un sistema lineare (tipicamente con il metodo PCG) ogniqualvolta si debba applicare il precondizionatore ad un vettore, un'operazione molto costosa.

4.8.3 Calcolo del precondizionatore

Nel paragrafo 3.1.1 abbiamo studiato i metodi lineari e stazionari per risolvere un sistema lineare "precondizionato" con precondizionatore P. Tali schemi risolutivi erano del tipo:

$$x_{k+1} = (I - P^{-1}A)x_k + P^{-1}b,$$

ovvero

$$x_{k+1} = x_k + P^{-1}r_k \qquad r_k = b - Ax_k., \tag{4.14}$$

Si noti che l'applicazione di un precondizionatore equivale a risolvere un sistema lineare che ha come matrice proprio P e come termine noto il residuo corrente, esattamente quello che si deve fare ad ogni iterazione di (4.14). Possiamo quindi pensare di usare un prefissato numero di iterazioni (ad esempio 1) di un metodo lineare e stazionario per il vettore risultante dall'applicazione del precondizionatore al residuo r_k. Per esempio, utilizzando una iterazione del metodo di Jacobi, il precondizionatore diventa $P = D$, mentre usando una iterazione del metodo di Gauss-Seidel si ha $P = (L + D)$. Volendo mantenere simetria anche nell'applicazione del metodo, il precondizionatore di Gauss-Seidel simetrico diventa $P = (L + D)D^{-1}(D + U)$.

Come si vede da quest'ultima espressione del precondizionatore, esso si può sempre scrivere come prodotto di due matrici triangolari, $P = LU$ (nel caso di GS simmetrico, $L = (M + D)D^{-1}$ e $U = M + D$) con le matrici L e U che hanno lo stesso pattern di sparsità della matrice A. Questa osservazione dà ragione, almeno empiricamente, alla definizione dei precondizionatori tramite le cosidette fattorizzazioni incomplete, fattorizzazioni cioè nelle quali si azzerano gli elementi corrispondenti a elementi nulli della matrice originale A.

Le fattorizzazioni incomplete

Seguendo l'idea precedentemente abbozzata, la fattorizzazione $ILU(0)$ o ILU (incomplete LU) parte dalla fattorizzazione di Crout LU (o Banakiewitz) mantenendo nulli tutti gli elementi l_{ij} di posizione corrispondente a elementi nulli di A ($l_{ij} = 0 \Leftrightarrow a_{ij} = 0$), si impone cioè che il pattern dei nonzeri di $ILU(0)$ e quello di A coincidano: $NZ(ILU(0)) = NZ(A)$. Si noti che per matrici simmetriche, ovviamente si usa la fattorizzazione di Choleski e $U = L^T$. La decomposta incompleta corrispondente si indica con $IC(0)$.

Si ricorda che data una matrice sparsa il processo di fattorizzazione completo LU "riempie" la matrice. In altre parole, tutti gli elementi delle matrici L e U localizzati all'interno della banda della matrice A sono nonnulli. Le fattorizzazioni incomplete sono quindi formate da "sottomatrici" di L e di U con un numero di nonzeri assai ridotto rispetto alle corrispondenti matrici complete. Questo processo si chiama processo di "fill-in".

Si noti che pre-imporre al precondizionatore il pattern di sparsità di A scaturisce da un'osservazione di carattere completamente empirico, e non è detto che il pattern di A sia quello che fornisce la massima efficienza. Per questo motivo, si definisce spesso un "livello di fill-in" o di "riempimento" del pattern, $\rho \neq 0$, che indica intuitivamente gli elementi di posizione al di fuori del pattern di sparsità di A che verrebbero formati (cioè trasformati in elementi non nulli) durante il passo ρ del processo di eliminazione di Gauss. Più precisamente, un elemento $p_{ij} \in NZ(ILU(\rho))$ ha livello di fill-in pari a ρ se nel grafo $G(A) = (V, E)$ associato alla matrice A il percorso che congiunge i con j è costituito da $\rho + 1$ salti. Intuitivamente, ci si può aspettare che un elemento p_{ij} sia più importante quanto più piccolo è il suo livello di fill-in. Qundi nel precondizionatore $ILU(\rho)$ si accettano tutti gli elementi che hanno livello di fill-in $\leq \rho$.

Preimporre il livello di fill-in non garantisce l'ottimalità dell'elemento della matrice precondizionatore. Si ricorre allora alla tecnica del "dropping", si considerano cioè solo gli elementi p_{ij} che sono sufficientemente grandi. Per rendere questa condizione precisa, si può pensare di accettare l'elemento p_{ij} al livello di fill-in ρ se $|p_{ij}| \geq \tau \|a_{i,\cdot}\|_2$, con τ una tolleranza prefissata. Questo dà forma al precondizionatore chiamato $ILUT(\rho, \tau)$. Un avvertimento: il livello di fill-in ρ può essere contato a partire da $NZ(I)$ o da $NZ(A)$, per cui bisogna stare molto attenti alla definizione di livelloe ρ data in ciascuna implementazione di $ILUT(\rho, \tau)$. Nella pratica, i valori di ρ e τ sono determinati empiricamente.

L'approccio utilizzato in $ILUT(\rho, \tau)$ di considerare non-nulli solo gli elementi della fattorizzazione che soddisfano a certi criteri "a-posteriori" è spesso chiamato "post-filtration", si filtrano cioè gli elementi dopo averli generati. Si parla invece di "pre-filtration" quando, come nella $ILU(0)$ si pre-impone il pattern di sparsità. La domanda quindi che ci si pone è quindi "qual'è il pattern di sparsità ottimale". E' impossibile dare risposta a tale domanda, che intuitivamente dipende dal problema che si vuole risolvere (e.g. soluzione di un sistema lineare o calcolo di autovalori). Si può però cercare di dare una spiegazione intuitiva che possa aiutare a capire il problema. Per prima cosa dobbiamo cercare dei pattern di sparsità che possano essere dei candidati ragionevoli, ad esempio il pattern di A^2: $NZ(A^2)$. Per vedere che potenze di A hanno pattern più pieni di quelli di A basta ricordare la serie di Neumann:

$$(I - E)^{-1} = \sum_{i=0}^{\infty} E^i \qquad \|E\| < 1. \tag{4.15}$$

Si vede subito che i primi termini della sommatoria hanno patterns $NZ(E^0) = NZ(I)$, $NZ(E)$, $NZ(E^2)$, ... Poichè la matrice $(I - E)^{-2}$ è piena, si deduce che i pattern di sparsità delle potenze di E sono di dimensioni crescenti, e quindi le prime potenze sono quelle più importanti per la definizione del $NZ(\cdot)$. Questa proprietà si può rendere precisa a partire dal grafo associato.

L'applicazione ad un vettore di un precondizionatore in forma di fattorizzazione $P = LU$ richiede il processo di sostituzione in avanti e all'indietro:

$$z = P^{-1}r \qquad Pz = r \qquad \Rightarrow \qquad Ly = r \qquad Uz = y.$$

Si noti che queste due sostituzioni (in avanti e all'indietro) sono operazioni intrinsecamente scalari e di difficile parallelizzazione.

Precondizionatori polinomiali

Riprendiamo la serie di Neumann (4.15) con una scalatura ω:

$$I + E + E^2 + \ldots \qquad\qquad E = I - \omega A,$$

or, similarly:

$$\omega A = D - (D - \omega A) \qquad (\omega A)^{-1} = \left[I - (I - \omega D^{-1}A)\right]^{-1} D^{-1},$$

da cui otteniamo immediatamente

$$D^{-1}A = \omega^{-1}(I - E) \qquad\qquad E = I - \omega A.$$

Possiamo quindi definire il precondizionatore P troncando la serie alla potenza s, ottenendo:

$$P^{-1} = [I + E + E^2 + \ldots + E^s]D^{-1} = \frac{1}{\omega}(I - E^{s+1}).$$

L'applicazione di E^{s+1} ad un vettore è generalmente costose, essendo equivalente al costo di s prodotti matrice-vettore. Per questo motivo sono stati sperimentati in letteratura polinomi diversi, ad esempio Tchebytchev. La tencnica però non ha avuto particolare successo per la difficoltà che il polinomio che minimizza il numero di iterazioni per PCG o PGMRES non coincide con il polinomio che minimizza $k(A)$ o che ha le proprietà spettral migliori. Questa difficoltà ha sempre frenato lo sviluppo e l'uso dei precondizionatori polinomiali, anche se essendo la loro applicazione basata esclusivamente sul prodotto matrice-vettore, essi sono intrinsecamente paralleli, al contrario dei precondizionatori basati su fattorizzazioni.

Inverse approssimate

Un campo di ricerca particolarmente attivo è quello legato ai precondizionatori basati su inverse approssimate. Essi partono dall'idea di approssimare direttamente P^{-1} invece che approssimare P e poi farne l'inversa. I precondizionatori così pensati hanno il vantaggio che la loro applicazione ad un vettore, non dovendo ricorrere alle sostituzioni in avanti e all'indietro, è completamente parallelizzabie.

Inoltre, i precondizionatori basati sulle decomposte incomplete potrebbero essere generalmente meno "stabili", o meglio, si ha meno controllo sulla loro accuratezza. Infatti, la decomposta incompleta $\tilde{L}\tilde{U}$ soddisfa all'equazione:

$$A = \tilde{L}\tilde{U} + E, \qquad \tilde{L}^{-1}A\tilde{U}^{-1} = I + \tilde{L}^{-1}E\tilde{U}^{-1},$$

dove E è la matrice errore. Si noti che spesso E è "poco importante" rispetto a $\tilde{L}^{-1}E\tilde{U}^{-1}$.

Si ricorda qui che una matrice B è un'inversa approssimata di A se:

$$\|I - BA\| < 1.$$

Quindi si vuole trovare una matrice sparsa P tale che:

$$\begin{aligned} \text{Left-AI:} \quad & P = \text{argmin}_B F(B) = \|I - AB\|_F^2 , \\ \text{Right-AI:} \quad & P = \text{argmin}_B F(B) = \|I - BA\|_F^2 , \\ \text{Split-AI:} \quad & P = \text{argmin}_{B=LU} F(B) = \|I - LAU\|_F^2 . \end{aligned}$$

L'idea fondamentale è qualla di mantenere comunque l'inversa approssimata in forma di fattorizzazione di due fattori triangolari, ai quali si applica un pattern di sparsità sia con la pre-filtration che con la post-filtration.

Il processo di ottimizzazione può essere risolto in maniera approssimata effettuando per esempio alcune (poche) iterazioni del metodo dello Steepest Descent applicato globalmente a tutta la funzione, oppure applicato localmente a ciascuna componente di $F(B)$. Il processo è sicuramente parallelizzabile visto che

$$F(B) = \|I - AB\|_F^2 = \sum_{j=1}^{n} \|e_j - Ab_j\|_2^2 \qquad F_j b = \|e_j - Ab\|_2^2 \qquad j = 1, 2, \ldots, n.$$

Riordinamenti ottimali

Una pratica molto comune per migliorare l'efficienza dei precondizionatori è quella di ricorrere a riordinamenti o permutazioni di righe e colonne in maniera tale da minimizzare il fill-in. Infatti, considerando ad esempio il caso della $ILU(0)$, e cioè un precondizionatore P tale che $NZ(P) = NZ(A)$, si vede che il numero di elementi di P che venono trascurati dipende direttamente dal grado di fill-in causato durante il processo di decomposizione completo. Quindi, la minimizzazione del fill-in durante tale processo porta a trascurare un numero minimi do elementi nel precondizionatore e quindi intuitivamente a migliorarne le caratteristiche spettrali.

Uno degli schemi di riordinamento più usato è il "reverse Cathill-McKee" (RCM) che minimizza la dimensione della banda della matrice. Si consideri, per esempio,

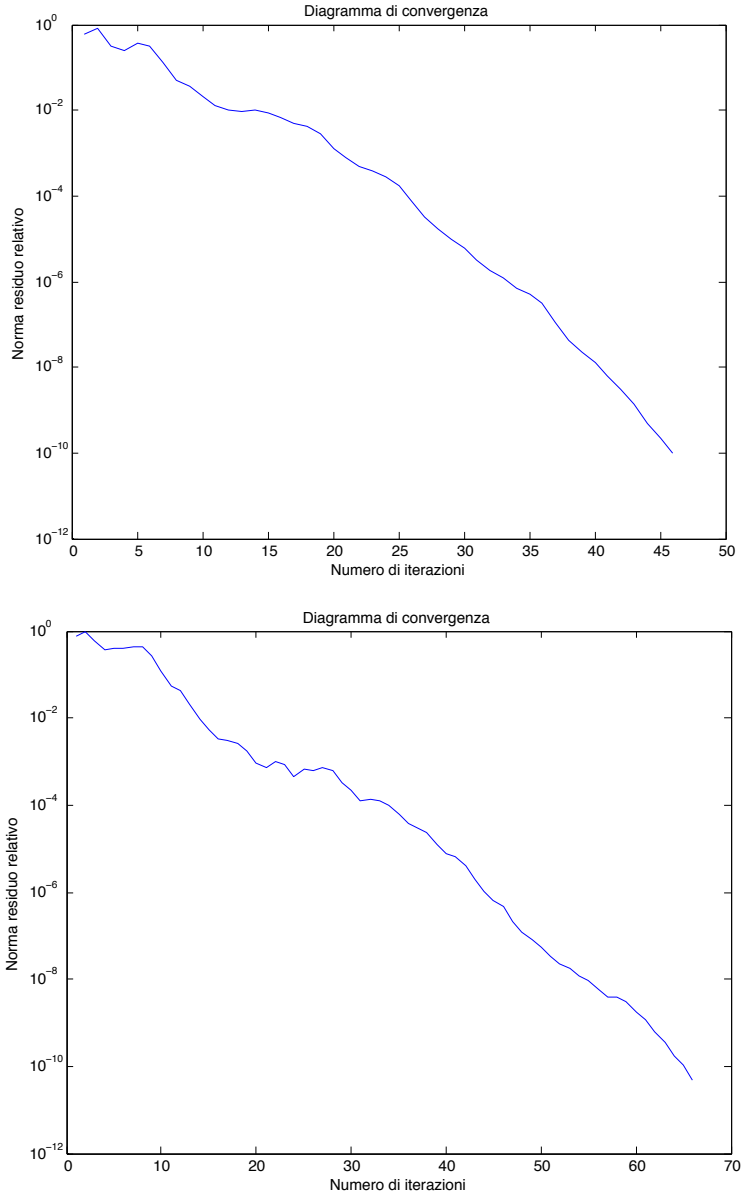

Figura 4.2: Profili di convergenza di PCG con precondizionatore $IC(0)$ con (sopra) e senza (sotto) riordinamento ottimale (Reverse Cathill-McKee)

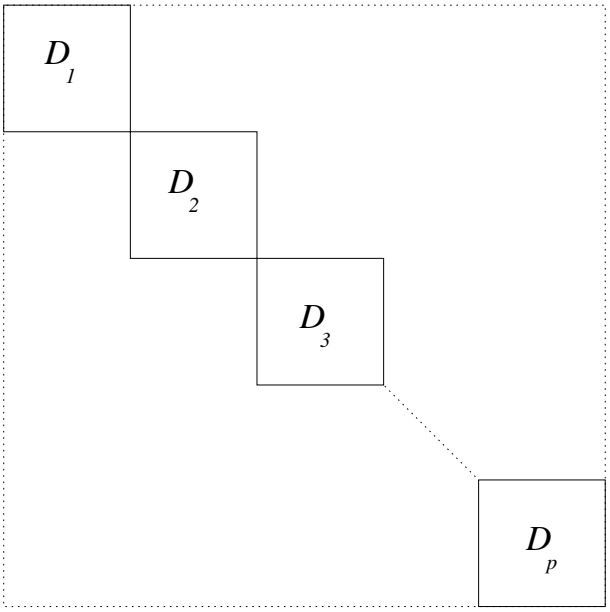

Figura 4.3: Esempio di precondizionatore diagonale a blocchi (o di Jacobi).

un caso di discretizzazione agli elementi finiti dell'equazione di Poisson:

$$\begin{aligned} -\Delta u &= f \qquad && x \in \Omega \subset \mathbb{R}^2 \\ u &= 0 && x \in \partial\Omega \end{aligned}$$

su una mesh di 1345 nodi (pari alla dimensione del sistema. In Figura 4.2 si riportano i grafici di convergenza del metodo PCG con precondizionatore $IC(0)$. Il grafico di sinistra mostra la convergenza ottenuta senza renumerazione, a destra il sistema è stato preliminarmente riordinato con RCM in modo da minimizzare la larghezza della banda. Si noti come si è ottenuto un risparmio nel numero di iterazioni maggiore del 30%.

Precondizionatori a blocchi

Una importante classe di precondizionatori sono i precondizionatori a blocchi. Per esempio, il precondizionatore di Jacobi a blocchi prende come matrice P la matrice fatta da sottomatrici diagonali della matrice A di dimensione piccola, come evidenziato in Figura 4.3.

L'iterazione di Jacobi a blocchi con rilassamento può essere formalizzata individuando p insiemi contenenti gli indici di riga che formano i p blocchi:

$$S_j = \{i : l_j \leq i \leq r_j\},$$

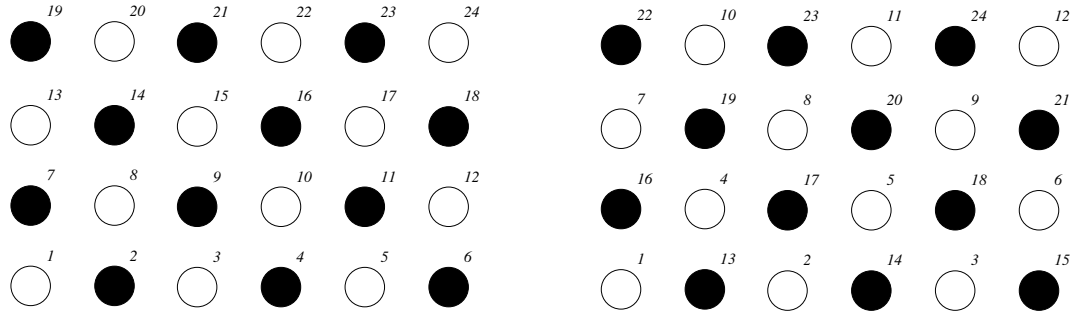

Figura 4.4: Colorazione "red-black" della matrice 24×24 (sinistra) e suo riordinamenta (a destra).

con:

$$l_1 = 1; \qquad r_p = n; \qquad r_j > l_{j+1}; \qquad 1 \leq j \leq p-1.$$

Definendo la matrice:

$$V_j = \left[e_{l_j}, e_{l_j+1}, \ldots, e_{r_j} \right],$$

l'iterazione di Jacobi con pesi ω_i si può scrivere come:

$$\begin{aligned} x_{k+1} &= x_k + \sum_{j=1}^{p} \omega_j V_j A_j^{-1} V_j^T r_k, \\ r_{k+1} &= \left[I - \sum_{j=1}^{p} \omega_j A V_j (V_j^T A_j V_j)^{-1} V_J^T \right] r_k. \end{aligned}$$

Quando i blocchi sono sufficientemente piccoli l'applicazione del precondizionatore viene fatta calcolando esplicitamente le inverse dei blocchi D_j. Altrimenti bisogna ricorrere ad un solutore sparso. L'efficienza del precondizionatore ovviamente dipende dalla dimensione dei blocchi, ma per problemi difficili (molto malcondizionati) il precondizionatore a blocchi non è consigliabile. Ovviamente è pensabile usare altri metodi lineari e stazionari a blocchi, ad esempio il metodo di Gauss Seidel, con considerazione del tutto analoghe a quelle fatte per il metodo di Jacobi.

Multicoloring: ordinamento "Red-Black"

Una tecnica molto usata per costruire matrici a blocchi utilizzabili per arriavare a precondizionatori efficienti è la tecnica del "multicoloring". In pratica si "colorano" (etichettano) i nodi del grafo di adiacenza della matrice in modo che due nodi vicini abbiano "colori" diversi. Per esempio, l'ordinamento "red-black" consiste nell'utilizzare due colori per individuare alternativamente i vertici di $G(A)$, come evidenziato in Figura 4.4, pannello di sinistra. Un riordinamento dei vertici del grafo come

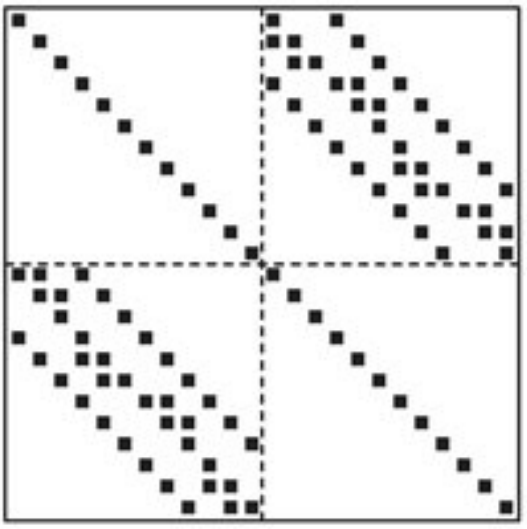

Figura 4.5: Sparsity pattern della matrice 24×24 dopo riordinamento red-black (da [4]).

mostrato nel pannello di destra delle figura porta ad una matrice riordinata con struttura a blocchi e blocchi diagonali sulla diagonale principale:

$$A = \begin{bmatrix} D_1 & B \\ C & D_2 \end{bmatrix} \tag{4.16}$$

il cui pattern di sparsità $NZ(A)$ è mostrato in Figura 4.5. Il sistema associato può quindi risolversi a blocchi formalmente andando a definire $x = (x_1, x_2)^T$ e $b = (b_1, b_2)^T$, che eliminando x_1 si ottiene immediatamente:

$$x_1 = D_1^{-1}(b_1 - Bx_2),$$

per cui il sistema si può ridurre risolvendo il seguente sistema sparso:

$$(D_2 - CD_1^{-1}B)x_2 = b_2 - CD_1^{-1}b_1.$$

Questo sistema ridotto è spesso facile da risolvere usando un semplice precondizionatore diagonale. E' spesso utilenon formare il sistema ridotto esplicitamente ma fare i prodotti matrice vettore memorizzando solo i blocchi D_1, D_2, B e C.

L'ordinamento red-black funziona però solo con matrici con struttura particolare. Per matrici sparse generali la struttura a blocchi mostrata in eq. (4.18) non è raggiungibile. Si può però arrivare sempre ad una struttura in cui il blocco $(1,1)$ è diagonale:

$$A = \begin{bmatrix} D_1 & B \\ C & E \end{bmatrix}, \tag{4.17}$$

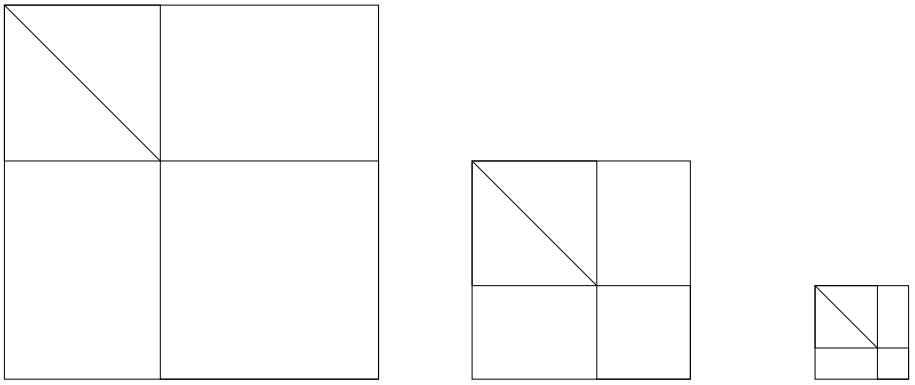

Figura 4.6: Processo di multi-eliminazione

Si può pensare di applicare il procedimento al blocco E e procedere quindi ricorsivamente fino ad arrivare ad un blocco finale sufficientemente piccolo da poterlo invertire esplicitamente, come mostrato graficamente in Figura 4.6, costruendo quindi un process di "multi-eliminazione".

Sia A_j la matrice ottenuta al livello (passo) j di tale processo, $j = 0, \ldots, n_{lev}$ con $A_0 = A$. Ad ogni livello la matrice A_j è permutata in modo da ottenere:

$$P_j A_j P_j^T = \begin{bmatrix} D_j & B_j \\ C_j & E_j \end{bmatrix},$$

e la nuova matrice ridotta diventa:

$$A_{j+1} = E_j - C_j D_j^{-1} B_j.$$

Questa eliminazione in realtà può essere vista come una sorta di fattorizzazione LU a blocchi:

$$P_j A_j P_j^T = \begin{bmatrix} D_j & B_j \\ C_j & E_j \end{bmatrix} == \begin{bmatrix} I & 0 \\ C_j D_j^{-1} & I \end{bmatrix} \times \begin{bmatrix} D_j & B_j \\ 0 & A_{j+1} \end{bmatrix}.$$

Il problema di questo procedimento è che le matrici A_{j+1} sono soggette a fill-in, diventano cioè sempre più piene. Occorre quindi trascurare elementi piccoli in questo processo di eliminazione per esempio eliminando tutti gli elementi "piccoli". Questa procedura è chiamata $ILUM(\tau)$ [4].

Usando di nuovo una ricorsione a blocchi si può proporre una permutazione della matrice A in cui il blocco D_1 non è diagonale ma diagonale a blocchi:

$$A = \begin{bmatrix} G & B \\ C & E \end{bmatrix} = \begin{bmatrix} I & 0 \\ EG^{-1} & I \end{bmatrix} \times \begin{bmatrix} G & B \\ 0 & S \end{bmatrix} \tag{4.18}$$

dove S è il complemento di Schur:

$$S = E - CG^{-1}B.$$

Fattorizzando quindi la matrice G con una decomposizione LU incompleta ($G = \tilde{L}\tilde{U} + E$, dove E contiene gli elementi di fill-in trascurati), si ottiene una approssimazione del tipo:

$$A = \begin{bmatrix} G & B \\ C & E \end{bmatrix} \approx \begin{bmatrix} \tilde{L} & 0 \\ E\tilde{U}^{-1} & I \end{bmatrix} \times \begin{bmatrix} I & 0 \\ 0 & S \end{bmatrix} \times \begin{bmatrix} \tilde{U} & \tilde{L}^{-1}B \\ 0 & I \end{bmatrix}$$

che è della forma LDU. Anche questo processo può essere generalizzato a blocchi come la $ILUM$. Questo modo di procedere è detto $ARMS$ (Algebraic Recursive Multilevel Solver), e può essere messo in connessione con metodi di tipo "multigrid" (si veda per maggiori dettagli [4]).

Precondizionatori per problemi indefiniti (saddle point problems)

Sia da risolvere il problema:

$$\begin{bmatrix} A & B^T \\ B & -C \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix},$$

dove A è $n \times n$, C è $m \times m$ e B è $m \times n$. Tipicamente $n > m$. Questi sistemi scaturiscono per esempio dalla discretizzazione di sistemi differenziali. E' questo il caso del problema della consolidazione delle rocce a seguito della estrazione di fluidi (acqua, gas, petrolio) che è governato dalle equazioni di Biot. Definiamo il precondizionatore ECP (Exact Constraint Preconditioner) come:

$$A \approx P = \begin{bmatrix} P_A & B^T \\ B & -C \end{bmatrix} = \begin{bmatrix} I & 0 \\ BP_A^{-1} & I \end{bmatrix} \begin{bmatrix} P_A & 0 \\ 0 & -S \end{bmatrix} \begin{bmatrix} I & P_A^{-1}B^T \\ 0 & I \end{bmatrix}$$

dove P_A è un'approssimazione simmetrica e definita positiva di A, e $S = C + BP_A^{-1}B^T$ é l'opposto del complemento di Schur di P. Un modo immediato per ottenere un buon precondizionatore è quello di definire $P_A = \mathrm{diag}(A)$. Si può in questo caso calcolare esplicitamente P^{-1}:

$$P^{-1} = \begin{bmatrix} I & -P_A^{-1}B^T \\ 0 & I \end{bmatrix} \begin{bmatrix} P_A^{-1} & 0 \\ 0 & -S^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -BP_A^{-1} & I \end{bmatrix}$$

Risolta che gli autovalori di $P_A^{-1}A$ sono tutti reali e positivi, con almento $n - m$ unitari. Gli altri autovalori sono più o meno vicini all'unità in funzione di quanto bene P_A approssima A. Si noti che ad ogni iterazione e.g. del PCG l'applicazione

del precondizionatore richiede la soluzione di un sistema che ha S come matrice, per cui il costo di un'iterazione è alto, ma la convergenza viene raggiunta con poche iterazioni.

Per diminuire il costo dell'applicazione di P si può pensare di sostituire a S una sua approssimazione $S \approx P_S$, ottenendo quello che si chiama un "Inexact Constraint Preconditioner" (ICP):

$$P_1 = \begin{bmatrix} P_A & B^T \\ B & S - P_s \end{bmatrix} = \begin{bmatrix} I & 0 \\ BP_A^{-1} & I \end{bmatrix} \begin{bmatrix} P_A & 0 \\ 0 & -P_s \end{bmatrix} \begin{bmatrix} I & P_A^{-1}B^T \\ 0 & I \end{bmatrix}$$

Oppure una versione cosidetta "triangolare":

$$P_2 = \begin{bmatrix} P_A & B^T \\ 0 & -P_s \end{bmatrix} = \begin{bmatrix} P_A & 0 \\ 0 & -P_s \end{bmatrix} \begin{bmatrix} I & P_A^{-1}B^T \\ 0 & I \end{bmatrix}$$

che va provata e il cui comportamente spettrale va analizzato. Esistono dei teoremi che determinano la clusterizzazione degli autovalori di P attorno a 1, e le prove sperimentali dimostrano che questi precondizionatori sono molto efficienti. La loro scalabilità parallela risulta però ancora limitata.

Bibliografia

[1] G. Gambolati. *Lezioni di metodi numerici per ingegneria e scienze applicate*. Cortina, Padova, Italy, 2 edition, 2002. 619 pp.

[2] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations*. SIAM, Philadelphia, 1995.

[3] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical Mathematics*. Springer, Berlin, Heidelberg, second edition, 2007.

[4] Y. Saad. *Iterative Methods for Sparse Linear Systems. Second edition*. SIAM, Philadelphia, PA, 2003.

[5] Y. Saad. *Numerical Methods for large eigenvalue problems. Second edition*. SIAM, Philadelphia, PA, 2011.

[6] J. R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical report, Pittsburgh, PA, USA, 1994.

[7] L. N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, 1997.