Indice

1. Sistema binario e sistema floating-point IEEE754

Sistema binario

La rappresentazione in base N di un numero reale x è data da:

$$x = \pm(x_n N^n + x_{n-1} N^{n-1} + \cdots + x_0 + x_{-1} N^{-1} + \cdots + x_{-r} N^{-r})$$

dove:

- $n \in \mathbb{N}$
- $r \in \mathbb{N} \cup +\infty$
- $x_j \in 0, 1, \ldots, N-1$ per ogni $j = n, n-1, \ldots, -r$

Per i computer, la base utilizzata è tipicamente $N = 2$, quindi le cifre sono $0, 1$.

Conversione da base 10 a base 2

Per la parte intera, si utilizza la divisione iterata per 2:

1. Si divide il numero per 2
2. Si annota il resto (0 o 1)
3. Si continua con il quoziente fino a ottenere 0

Per la parte frazionaria, si utilizza la moltiplicazione iterata per 2:

1. Si moltiplica la parte frazionaria per 2
2. Si annota la parte intera del risultato (0 o 1)

3. Si continua con la parte frazionaria del risultato fino a ottenere 0 o un ciclo

Esempio: $(10011010010)_2 = 2^{10} + 2^7 + 2^6 + 2^4 + 2^1 = 1024 + 128 + 64 + 16 + 2 = 1234_{10}$

Problema importante: Non tutti i numeri decimali hanno una rappresentazione binaria finita. Ad esempio, 0.1_{10} ha una rappresentazione binaria periodica:
$0.1_{10} = (0.00011001100110011\dots)_2$

IEEE754 - Rappresentazione in virgola mobile

Lo standard IEEE754 definisce la rappresentazione dei numeri in virgola mobile nei computer moderni.

Struttura generale

Un numero $x \neq 0$ in formato IEEE754 viene rappresentato come:

$$(x)_2 = (-1)^s \times 2^{e-b} \times 1.f$$

dove:

- s è il bit di segno (0 per positivo, 1 per negativo)
- e è l'esponente con bias b
- $1.f$ è la mantissa (parte frazionaria normalizzata)

Formati IEEE754

Formato a 32 bit (float):

- 1 bit per il segno
- 8 bit per l'esponente (e), con bias $b = 127$
- 23 bit per la mantissa (f)
- Range di e: $0 < e < 255$

Formato a 64 bit (double):

- 1 bit per il segno
- 11 bit per l'esponente (e), con bias $b = 1023$
- 52 bit per la mantissa (f)
- Range di e: $0 < e < 2047$

Casi speciali

1. **Zero**: $e = 0, f = 0$ (può essere +0 o -0 a seconda del bit di segno)
2. **Infinito**: $e = e_{max}, f = 0$ (+∞ o -∞ a seconda del bit di segno)
3. **NaN** (Not a Number): $e = e_{max}, f > 0$ (risultato di operazioni indefinite)

4. **Numeri denormalizzati**: $e = 0, f > 0$ (per rappresentare numeri molto piccoli)

Spaziatura e limiti

La spaziatura tra i numeri rappresentabili non è uniforme: è più densa vicino allo zero e meno densa per numeri di grande modulo.

Massimo e minimo modulo rappresentabile:

- Numero normalizzato più piccolo (in modulo): $2^{1-b} \approx 1.18 \times 10^{-38}$ (float)
- Numero normalizzato più grande (in modulo): $2^{e_{max}-b} \times (2 - 2^{-p}) \approx 3.4 \times 10^{38}$ (float)

2. Errori assoluto e relativo, errore di rappresentazione

Definizioni di errore

Sia $\tilde{x}$ un'approssimazione di x:

- **Errore assoluto**: $e_{abs} = |x - \tilde{x}|$
- **Errore relativo**: $e_{rel} = \frac{|x-\tilde{x}|}{|x|} = \frac{e_{abs}}{|x|}$ (per $x \neq 0$)

Errore di rappresentazione

Quando un numero reale x viene rappresentato in un computer come $fl(x)$, si verifica un errore di rappresentazione dovuto alla precisione finita.

Troncamento

Nel troncamento, tutte le cifre oltre quelle rappresentabili vengono semplicemente scartate.

Per esempio, se approssimiamo $\pi = 3.14159\ldots$ con 3 cifre decimali per troncamento, otteniamo $\tilde{\pi} = 3.141$.

Arrotondamento

Nell'arrotondamento, la cifra rappresentabile meno significativa viene incrementata se la cifra successiva è maggiore o uguale a 5.

Per esempio, se approssimiamo $\pi = 3.14159\ldots$ con 3 cifre decimali per arrotondamento, otteniamo $\tilde{\pi} = 3.142$.

Modello matematico dell'errore di rappresentazione

In generale, possiamo modellare la rappresentazione di un numero reale x in un computer come:

$$fl(x) = x(1 + \delta), \quad |\delta| \leq \varepsilon_{mach}$$

dove ε_{mach} è la precisione macchina, che rappresenta il più piccolo numero positivo tale che $fl(1+\varepsilon_{mach}) > 1$.

Per IEEE754:

- Float (32 bit): $\varepsilon_{mach} \approx 5.96 \times 10^{-8}$
- Double (64 bit): $\varepsilon_{mach} \approx 1.11 \times 10^{-16}$

3. Operazioni macchina, loro (non) proprietà, precisione macchina

Operazioni macchina

Le operazioni aritmetiche eseguite su numeri macchina $(\tilde{x}, \tilde{y})$ producono risultati approssimati. In generale, per un'operazione $\circ \in +, -, \times, \div$:

$$\tilde{x} \circ_M \tilde{y} = fl(\tilde{x} \circ \tilde{y}) = (\tilde{x} \circ \tilde{y})(1+\delta), \quad |\delta| \leq \varepsilon_{mach}$$

Non proprietà delle operazioni macchina

Le operazioni macchina non conservano tutte le proprietà algebriche delle operazioni esatte:

1. **Non associatività**: $(a +_M b) +_M c \neq a +_M (b +_M c)$
2. **Non distributività**: $a \times_M (b +_M c) \neq a \times_M b +_M a \times_M c$

La commutativià è generalmente conservata: $a +_M b = b +_M a$ e $a \times_M b = b \times_M a$.

Esempio di non associatività: Consideriamo $(10^{16} + 1) - 10^{16}$ e $10^{16} + (1 - 10^{16})$ in precisione doppia.

- Nel primo caso: $fl(10^{16} + 1) = 10^{16}$, quindi $fl(fl(10^{16} + 1) - 10^{16}) = 0$
- Nel secondo caso: $fl(1 - 10^{16}) = -10^{16}$, quindi $fl(10^{16} + fl(1 - 10^{16})) = 0$ Il risultato esatto sarebbe 1.

Precisione macchina

La precisione macchina ε_{mach} è fondamentale per quantificare l'accuratezza delle operazioni. In IEEE754:

- Per numeri normalizzati in formato a 32 bit: $\varepsilon_{mach} = 2^{-24} \approx 5.96 \times 10^{-8}$
- Per numeri normalizzati in formato a 64 bit: $\varepsilon_{mach} = 2^{-53} \approx 1.11 \times 10^{-16}$

4. Stabilità di un algoritmo, stabilità delle operazioni macchina e cancellazione numerica

Stabilità di un algoritmo

Un algoritmo è stabile se piccole perturbazioni nei dati di input producono piccole perturbazioni nei risultati.

Definizione formale: Un algoritmo A che risolve un problema P è stabile se, per ogni input x e perturbazione δx, esiste una costante K tale che:

$$\frac{|A(x+\delta x) - A(x)|}{|A(x)|} \leq K \frac{|\delta x|}{|x|}$$

Stabilità della somma

Consideriamo la somma di n numeri $S_n = \sum_{i=1}^{n} a_i$ e la sua approssimazione $\tilde{S}_n = \sum_{i=1}^{n} \tilde{a}_i$.

Teorema: Se $\tilde{a}_i = a_i(1+\delta_i)$ con $|\delta_i| \leq \delta$ per ogni i, allora:

$$|\tilde{S}_n - S_n| \leq \delta \sum_{i=1}^{n} |a_i|$$

Dimostrazione:

$$|\tilde{S}_n - S_n| = \left|\sum_{i=1}^{n} \tilde{a}_i - \sum_{i=1}^{n} a_i\right| = \left|\sum_{i=1}^{n} (\tilde{a}_i - a_i)\right| = \left|\sum_{i=1}^{n} a_i\delta_i\right| \leq \sum_{i=1}^{n} |a_i\delta_i| \leq \delta \sum_{i=1}^{n} |a_i|$$

Cancellazione numerica

La cancellazione numerica avviene quando si sottraggono due numeri quasi uguali, risultando in una significativa perdita di cifre significative.

Esempio: Calcoliamo $f(x) = \frac{1-\cos(x)}{x^2}$ per x molto piccolo.

Approccio diretto:

- Per $x = 10^{-8}$, $\cos(x) \approx 0.9999999999995$
- $1 - \cos(x) \approx 5 \times 10^{-16}$
- $f(x) \approx \frac{5 \times 10^{-16}}{10^{-16}} \approx 0.5$

Ma sappiamo che $\lim_{x \to 0} f(x) = \frac{1}{2}$, quindi il risultato è abbastanza accurato.

Usando l'identità trigonometrica $1 - \cos(x) = 2\sin^2(x/2)$:

- $f(x) = \frac{2\sin^2(x/2)}{x^2} = \frac{2\sin^2(x/2)}{4(x/2)^2} = \frac{1}{2} \cdot \frac{\sin^2(x/2)}{(x/2)^2}$
- Per x piccolo, $\frac{\sin^2(x/2)}{(x/2)^2} \approx 1$
- Quindi $f(x) \approx \frac{1}{2}$

Esempio di instabilità: Integrale iterato

Consideriamo il calcolo di $I_n = e^{-1} \int_0^1 x^n e^x dx$ per $n = 0, 1, \ldots, 40$ usando la formula ricorsiva:

$$I_n = 1 - nI_{n-1}$$

con $I_0 = \frac{1-e}{e}$.

L'algoritmo si dimostra instabile per n grande a causa dell'amplificazione degli errori di arrotondamento nella ricorsione.

5. Problema matematico, buona posizione

Definizione di problema matematico

Un problema matematico P può essere visto come una funzione $P : X \to Y$ che mappa uno spazio di input X a uno spazio di output Y.

Problema ben posto (secondo Hadamard)

Un problema $P : X \to Y$ è ben posto se:

1. **Esistenza**: Per ogni input $x \in X$ esiste almeno una soluzione $y \in Y$ tale che $P(x) = y$
2. **Unicità**: Per ogni input $x \in X$ esiste al più una soluzione $y \in Y$ tale che $P(x) = y$
3. **Continuità/Stabilità**: La soluzione $y = P(x)$ dipende con continuità dai dati di input x

Problema mal posto

Un problema che non soddisfa almeno una delle tre condizioni è detto mal posto.

Esempio di problema mal posto: Differenziazione numerica. Piccole perturbazioni nei dati possono portare a grandi cambiamenti nella derivata calcolata.

6. Condizionamento numerico assoluto e relativo di un problema ben posto

Condizionamento numerico

Il condizionamento numerico di un problema quantifica quanto le perturbazioni nei dati di input influenzano la soluzione.

Condizionamento assoluto

Sia $P : X \to Y$ un problema ben posto e sia $x \in X$. Il condizionamento assoluto di P in x è:

$$K_{abs}(P, x) = \lim_{\delta \to 0} \sup_{|\delta x| \leq \delta} \frac{|P(x + \delta x) - P(x)|}{\delta}$$

Se P è differenziabile in x, allora $K_{abs}(P, x) = |P'(x)|$.

Condizionamento relativo

Il condizionamento relativo di P in x è:

$$K_{rel}(P,x) = \lim_{\delta \to 0} \sup_{|\delta x| \leq \delta |x|} \frac{|P(x+\delta x) - P(x)|}{|P(x)|} \cdot \frac{|x|}{\delta |x|} = K_{abs}(P,x) \cdot \frac{|x|}{|P(x)|}$$

Problema ben condizionato vs mal condizionato

- Un problema è ben condizionato se $K_{rel}(P,x)$ è piccolo
- Un problema è mal condizionato se $K_{rel}(P,x)$ è grande

Esempio di problema mal condizionato: Calcolo delle radici di un polinomio di grado elevato. Piccole variazioni nei coefficienti possono causare grandi variazioni nelle radici.

7. Numero di condizionamento di una matrice e stima dell'errore relativo della soluzione di un sistema lineare

Numero di condizionamento di una matrice

Sia $A \in \mathbb{R}^{n \times n}$ invertibile. Il numero di condizionamento di A rispetto a una norma $|\cdot|$ è:

$$\text{cond}(A) = |A| \cdot |A^{-1}|$$

Proprietà del numero di condizionamento

1. $\text{cond}(A) \geq 1$ per qualsiasi matrice invertibile
2. $\text{cond}(A) = \text{cond}(A^{-1})$
3. $\text{cond}(cA) = \text{cond}(A)$ per qualsiasi $c \neq 0$
4. Se A è ortogonale, allora $\text{cond}_2(A) = 1$

Stima dell'errore relativo nella soluzione di sistemi lineari

Consideriamo il sistema $Ax = b$ con A invertibile e la sua versione perturbata $A\tilde{x} = \tilde{b}$.

Teorema: Se $\tilde{b} = b + \delta b$, allora:

$$\frac{|\tilde{x} - x|}{|x|} \leq \text{cond}(A) \cdot \frac{|\delta b|}{|b|}$$

Dimostrazione:

$$\tilde{x} - x = A^{-1}\tilde{b} - A^{-1}b \qquad = A^{-1}(\tilde{b} - b) \; = A^{-1}\delta b$$

Prendendo le norme:

$$|\tilde{x} - x| = |A^{-1}\delta b| \qquad \leq |A^{-1}| \cdot |\delta b|$$

Dividendo per $|x|$ e notando che $|b| = |Ax| \leq |A| \cdot |x|$:

$$\frac{|\tilde{x} - x|}{|x|} \leq |A^{-1}| \cdot \frac{|\delta b|}{|x|} \qquad = |A^{-1}| \cdot \frac{|\delta b|}{|b|} \cdot \frac{|b|}{|x|} \; \leq |A^{-1}| \cdot |A| \cdot \frac{|\delta b|}{|b|} \qquad = \text{cond}(A) \cdot \frac{|\delta b|}{|b|}$$

Caso in cui è perturbata anche la matrice

Se consideriamo perturbazioni sia nella matrice che nel termine noto, $(A + \delta A)\tilde{x} = \tilde{b}$, allora:

$$\frac{|\tilde{x} - x|}{|x|} \leq \frac{\text{cond}(A)}{1 - \text{cond}(A) \cdot \frac{|\delta A|}{|A|}} \left(\frac{|\delta b|}{|b|} + \frac{|\delta A|}{|A|} \right)$$

purché $\text{cond}(A) \cdot \frac{|\delta A|}{|A|} < 1$.

8. Algoritmi di sostituzione avanti e sostituzione indietro, condizioni per l'applicabilità

Algoritmo di sostituzione in avanti

Per risolvere un sistema triangolare inferiore $Lx = b$ con L avente elementi diagonali non nulli:

```
Per i = 1, 2, ..., n:
    x[i] = (b[i] - sum(L[i,j] * x[j] per j = 1, 2, ..., i-1)) / L[i,i]
```

Condizioni per l'applicabilità: L deve essere triangolare inferiore con elementi diagonali non nulli.

Algoritmo di sostituzione all'indietro

Per risolvere un sistema triangolare superiore $Ux = b$ con U avente elementi diagonali non nulli:

```
Per i = n, n-1, ..., 1:
    x[i] = (b[i] - sum(U[i,j] * x[j] per j = i+1, i+2, ..., n)) / U[i,i]
```

Condizioni per l'applicabilità: U deve essere triangolare superiore con elementi diagonali non nulli.

Complessità computazionale

Entrambi gli algoritmi hanno complessità $O(n^2)$, che è ottimale per questo tipo di problemi.

9. Fattorizzazione LU senza pivoting: algoritmo, limitazioni alla sua applicabilità e problematiche numeriche

Fattorizzazione LU

La fattorizzazione LU decompone una matrice quadrata A come prodotto di una matrice triangolare inferiore L con diagonale unitaria e una matrice triangolare superiore U:

$$A = LU$$

Algoritmo per la fattorizzazione LU senza pivoting

```
Per k = 1, 2, ..., n-1:
    Per i = k+1, k+2, ..., n:
        m[i,k] = A[i,k] / A[k,k]
        Per j = k+1, k+2, ..., n:
            A[i,j] = A[i,j] - m[i,k] * A[k,j]
```

Dopo l'esecuzione dell'algoritmo:

- La parte triangolare superiore di A diventa U
- La parte triangolare inferiore è sostituita dai valori $m[i,j]$ che formano L

Limitazioni e problematiche

1. **Limitazione algebrica**: L'algoritmo richiede che tutti i pivot (elementi diagonali) siano non nulli, altrimenti non è possibile completare la fattorizzazione.
2. **Problematica numerica**: Anche se tutti i pivot sono non nulli, se alcuni sono molto piccoli si possono verificare instabilità numeriche. Ad esempio, quando dividiamo per un pivot molto piccolo, i valori $m[i,k]$ possono diventare molto grandi, amplificando gli errori di arrotondamento.

Esempio: Consideriamo la matrice

$$A = \begin{pmatrix} 10^{-10} & 1 \\ 1 & 1 \end{pmatrix}$$

Se utilizziamo la fattorizzazione LU senza pivoting, otteniamo:

$$L = \begin{pmatrix} 1 & 0 \\ 10^{10} & 1 \end{pmatrix}, \quad U = \begin{pmatrix} 10^{-10} & 1 \\ 0 & -10^{10} + 1 \end{pmatrix}$$

L'elemento 10^{10} in L è molto grande e può causare instabilità numerica.

10. Algoritmo LU con pivoting parziale per righe: "idea" del pivoting e sua motivazione, output dell'algoritmo

Pivoting parziale per righe

L'idea del pivoting parziale è di scambiare le righe della matrice A in modo che, ad ogni passo k, il pivot $A[k,k]$ sia l'elemento di maggior modulo nella colonna k considerando solo le righe da k a n.

Motivazione

Il pivoting parziale per righe mira a:

1. Evitare la divisione per zeri o numeri molto piccoli
2. Ridurre gli errori di arrotondamento minimizzando l'amplificazione degli errori

Algoritmo LU con pivoting parziale

```
Per k = 1, 2, ..., n-1:
    Trova l'indice r ≥ k tale che |A[r,k]| = max{|A[i,k]| : i = k, k+1, ..., n}
    Se r ≠ k, scambia le righe k e r di A
    Registra lo scambio nella matrice di permutazione P
    Per i = k+1, k+2, ..., n:
        m[i,k] = A[i,k] / A[k,k]
        Per j = k+1, k+2, ..., n:
            A[i,j] = A[i,j] - m[i,k] * A[k,j]
```

Output dell'algoritmo

La fattorizzazione LU con pivoting parziale produce:

1. Una matrice di permutazione P
2. Una matrice triangolare inferiore L con diagonale unitaria
3. Una matrice triangolare superiore U

tali che:

$$PA = LU$$

Vantaggi rispetto alla fattorizzazione senza pivoting

1. Maggiore stabilità numerica
2. Garanzia che l'algoritmo possa essere completato per qualsiasi matrice invertibile
3. Controllo della crescita degli elementi di L

11. Punti fissi, lemma delle contrazioni con dimostrazione

Punti fissi

Definizione: Un punto x^* è un punto fisso di una funzione $F : D \to \mathbb{R}^n$ se $F(x^*) = x^*$.

Contrazioni

Definizione: Una funzione $F : D \to \mathbb{R}^n$ con $D \subset \mathbb{R}^n$ è una contrazione rispetto a una norma $|\cdot|$ se esiste una costante $L < 1$ tale che:

$$|F(x) - F(y)| \leq L|x - y|, \quad \forall x, y \in D$$

Lemma delle contrazioni

Teorema (Lemma delle contrazioni): Sia $F : \mathbb{R}^n \to \mathbb{R}^n$ una contrazione rispetto a una norma $|\cdot|$ con costante $L < 1$. Allora:

1. Esiste un unico punto fisso $x^* \in \mathbb{R}^n$ di F
2. Per ogni scelta di $x^{(0)} \in \mathbb{R}^n$, la successione $x^{(k)}{}_{k\in\mathbb{N}}$ definita da $x^{(k+1)} = F(x^{(k)})$ converge a x^*
3. Vale la stima di errore: $|x^{(k+1)} - x^*| \leq L|x^{(k)} - x^*|$, quindi $|x^{(k)} - x^*| \leq L^k|x^{(0)} - x^*|$
   Dimostrazione:
4. Dimostriamo prima che la successione $x^{(k)}$ è di Cauchy.

   Definiamo $s^{(k)} = x^{(k+1)} - x^{(k)}$. Notiamo che:

   $$x^{(k+l)} - x^{(k)} = \sum_{j=0}^{l-1} s^{(k+j)}$$

   $$|s^{(k)}| = |F(x^{(k)}) - F(x^{(k-1)})| \leq L|x^{(k)} - x^{(k-1)}| = L|s^{(k-1)}|$$

   Quindi:

   $$|x^{(k+l)} - x^{(k)}| \leq \sum_{j=0}^{l-1} |s^{(k+j)}| \leq \sum_{j=0}^{l-1} L^j|s^{(k)}| \leq |s^{(k)}|\frac{1 - L^l}{1 - L}$$

   Per k sufficientemente grande, $|x^{(k+l)} - x^{(k)}| < \epsilon$ per ogni l, quindi la successione è di Cauchy.

   In $\mathbb{R}^n$, ogni successione di Cauchy converge, quindi esiste $\lim_{k\to\infty} x^{(k)} = x^*$.
5. Dimostriamo che x^* è un punto fisso:

   $$x^* = \lim_{k\to\infty} x^{(k+1)} = \lim_{k\to\infty} F(x^{(k)}) = F(\lim_{k\to\infty} x^{(k)}) = F(x^*)$$

   dove abbiamo usato la continuità di F.
6. Dimostriamo l'unicità:

   Supponiamo che esistano due punti fissi x^* e y^* con $x^* \neq y^*$. Allora:

   $$|x^* - y^*| = |F(x^*) - F(y^*)| \leq L|x^* - y^*| < |x^* - y^*|$$

   Questa è una contraddizione, quindi $x^* = y^*$.
7. Dimostriamo la stima di errore:

   $$|x^{(k+1)} - x^*| = |F(x^{(k)}) - F(x^*)| \leq L|x^{(k)} - x^*|$$

Iterando questa disuguaglianza, otteniamo:

$$|x^{(k)} - x^*| \leq L^k |x^{(0)} - x^*|$$

12. Metodi iterativi lineari stazionari per soluzione di Ax=b

Introduzione ai metodi iterativi

I metodi iterativi costruiscono una successione di vettori $x^{(k)}{}_{k\in\mathbb{N}}$ che converge alla soluzione esatta x^* del sistema $Ax = b$.

Metodo iterativo lineare stazionario

Definizione: Un metodo iterativo lineare stazionario ha la forma:

$$x^{(k+1)} = F(x^{(k)}) = Ex^{(k)} + q, \quad k \in \mathbb{N}$$

dove $E \in \mathbb{R}^{n\times n}$ e $q \in \mathbb{R}^n$ sono indipendenti da k.

Convergenza

Teorema: Sia $|\cdot|$ una norma in $\mathbb{R}^n$ e $|\cdot|_*$ la norma indotta in $\mathbb{R}^{n\times n}$. Se $|E|_* < 1$, allora per ogni $x^{(0)} \in \mathbb{R}^n$, la successione generata dal metodo iterativo converge all'unico punto fisso $\bar{x}$ di F e vale la stima:

$$|x^* - x^{(k+1)}| \leq \|E\|_*^k |x^* - x^{(0)}|$$

Convergenza

Teorema: Sia $|\cdot|$ una norma in $\mathbb{R}^n$ e $|\cdot|_*$ la norma indotta in $\mathbb{R}^{n\times n}$. Se $|E|_* < 1$, allora per ogni $x^{(0)} \in \mathbb{R}^n$, la successione generata dal metodo iterativo converge all'unico punto fisso $\bar{x}$ di F e vale la stima:

$$|x^* - x^{(k+1)}| \leq \|E\|_*^k |x^* - x^{(0)}|$$

Consistenza

Affinché il metodo converga alla soluzione del sistema $Ax = b$, deve valere la condizione di consistenza:

$$(I_n - E)A^{-1}b = q$$

Metodo di Richardson

Il metodo di Richardson è definito come:

$$x^{(k+1)} = (I_n - A)x^{(k)} + b, \quad k \in \mathbb{N}$$

Questo corrisponde alla scelta $E = I_n - A$ e $q = b$.

Proprietà spettrali

La convergenza del metodo può essere analizzata tramite le proprietà spettrali della matrice di iterazione E:

$$x^{(k)} = E^k x^{(0)} + \left(\sum_{j=0}^{k-1} E^j \right) q$$

Se $E = P^{-1} \Lambda P$ con $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, allora:

$$\lim_{k \to \infty} x^{(k)} = P^{-1} \mathrm{diag} \left(\frac{1}{1-\lambda_1}, \ldots, \frac{1}{1-\lambda_n} \right) Pq = (I_n - E)^{-1} q$$

Il metodo converge se e solo se $\rho(E) < 1$, dove $\rho(E)$ è il raggio spettrale di E (il massimo dei moduli degli autovalori).

Precondizionamento

Per migliorare la convergenza, si può introdurre una matrice di precondizionamento P:

$$x^{(k+1)} = (I_n - P^{-1} A) x^{(k)} + P^{-1} b, \quad k \in \mathbb{N}$$

Metodi classici:

1. **Metodo di Jacobi**: $P = D$ (parte diagonale di A)
2. **Metodo di Gauss-Seidel**: $P = D + L$ (parte diagonale più triangolare inferiore di A)