

# Appunti di LPC

Matteo Zoppetti

## Indice

<b>1</b>	<b>Metodi di risoluzione di equazioni</b>	<b>2</b>
1.1	Metodo di bisezione . . . . .	2
1.2	Metodo del punto fisso . . . . .	2
1.3	Metodo di Newton . . . . .	3
1.4	Velocità di convergenza . . . . .	3
<b>2</b>	<b>Interpolazione di polinomi</b>	<b>4</b>
<b>3</b>	<b>Integrazione numerica</b>	<b>5</b>
3.1	Regola dei rettangoli . . . . .	5
3.2	Regola del punto medio . . . . .	5
3.3	Formula del trapezio . . . . .	5
3.4	Formula di Gauss . . . . .	6
3.5	Formula di Simpson . . . . .	6
3.6	Montecarlo . . . . .	6
<b>4</b>	<b>Risoluzione di sistemi lineari <math>Ax = b</math></b>	<b>7</b>
4.1	Metodo di Jacobi . . . . .	7
4.2	Metodo di Gauss-Seidel . . . . .	8
4.3	Algoritmo di Gauss . . . . .	8

# 1 Metodi di risoluzione di equazioni

Sia  $f : \mathbb{R} \rightarrow \mathbb{R}$  continua in  $[a, b]$  tale che  $f(a) \cdot f(b) < 0$

L'obiettivo degli algoritmi presentati è approssimare la soluzione  $\alpha \in [a, b] \mid f(\alpha) = 0$ . Notare che l'esistenza di  $\alpha$  è garantita dal teorema di esistenza degli zeri. Questi metodi funzionano costruendo successioni  $\{x_k\}_k$  che convergono ad  $\alpha$ .

## 1.1 Metodo di bisezione

Considero l'intervallo  $[a_0, b_0] = [a, b]$  e il punto medio  $\xi_0 = \frac{a_0+b_0}{2}$  ora, se  $f(\xi_0) \cdot f(b) < 0 \implies [a_1, b_1] = [\xi_0, b_0]$ , altrimenti  $[a_1, b_1] = [a_0, \xi]$ . Reitero il processo un numero arbitrario di volte per approssimare  $\alpha$

**Convergenza** Definiamo l'errore al passo  $k$  come  $e_k = |\alpha - \xi_k|$  e notiamo che  $e_k \leq \frac{b-a}{2^k} \leq \frac{b-a}{2^{k+1}} \rightarrow 0$  e che  $\lim_{k \rightarrow \infty} \xi_k = \alpha$ . Il metodo di bisezione ha convergenza lineare.

**Tolleranza** Se richiedo che  $e_k \leq tol$  ho bisogno di un numero di passaggi tali che  $\frac{b-a}{2^k} \leq tol$  e pertanto  $k \geq \lceil \log_2 \left( \frac{b-a}{tol} \right) \rceil$ . Da questo calcolo segue che ogni 4 iterazioni la stima di  $\alpha$  migliora di una cifra decimale.

### Criteri di arresto

- Residuo:  $e_k = |\alpha - \xi_k| \leq tol$
- Distanza dagli estremi:  $\frac{|b_k - a_k|}{|\xi_k|} \leq tol$
- Numero massimo di iterazioni

## 1.2 Metodo del punto fisso

Poniamo  $f(x), g(x) \mid g(x) = x + f(x)$ .

$$\implies f(\alpha) = 0 \iff g(\alpha) = \alpha$$

Il punto fisso di  $g$  è  $\bar{x} : \bar{x} = g(\bar{x})$ .

L'algoritmo consiste nel costruire una successione  $x_k \mid x_{k+1} = g(x_k)$ , con  $x_0$  input dell'utente.

**Convergenza** Consideriamo l'errore  $e_k$

$$e_{k+1} = |\alpha - x_{k+1}| = |g(\alpha) - g(x_k)| \leq |g'(\xi_k)| \cdot |\alpha - x_k|$$

Da cui segue che  $e_k \rightarrow 0$  se  $|g'(x)| < 1$ . L'algoritmo ha convergenza lineare, tanto più veloce quanto più  $|g'(x)|$  è piccola.

Per trovare una  $g$  adatta, pongo:

$$g(x) = x + \lambda f(x)$$

con  $\lambda : |g'(x)| < 1 \implies |1 + \lambda f'(x)| < 1$ .

### Criteri di arresto

- Numero massimo di iterazioni
- Residuo:  $e_k = |\alpha - x_k| \leq tol$
- Vicinanza tra iterate:  $\frac{|x_{k+1} - x_k|}{|x_{k+1}|} \leq tol$

### 1.3 Metodo di Newton

È un riadattamento del metodo del punto fisso. Si pone  $g(x) = x - \frac{f(x)}{f'(x)}$ . Da cui segue che  $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$ .

### Convergenza

$$0 = f(\alpha) = f(x_k) + (\alpha - x_k)f'(x_k) + O((\alpha - x_k)^2) \implies \alpha \approx x_k - \frac{f(x_k)}{f'(x_k)}$$

Dall'espansione con Taylor segue che il metodo diverge con velocità quadratica:  $e_k \leq C e_{k-1}^2$ . Il metodo non converge sempre: se la derivata è troppo piccola questo non converge. Per questo motivo spesso si usa il metodo di Newton con "warm start": qualche iterata con il metodo di bisezione e poi metodo di Newton.

I criteri di arresto sono i medesimi del metodo del punto fisso.

### 1.4 Velocità di convergenza

Un metodo ha convergenza di ordine  $p$  se:

$$\exists C > 0 \mid e_{k+1} \leq C \cdot e_k^p$$

Per calcolare sperimentalmente l'ordine di convergenza di un metodo di risoluzione, calcolo l'errore di ogni iterata  $e_k$  e osservo che:

$$\log(e_k) \approx \log(e_{k-1}^p) + \log(c) \implies p \approx \frac{\log(\frac{e_{k+1}}{e_k})}{\log(\frac{e_k}{e_{k+1}})}$$

## 2 Interpolazione di polinomi

È utilizzata per approssimare una qualche funzione  $f(x)$  con un polinomio  $p_n \in \mathbb{R}[x]_{\leq n}$ . Un altro uso comune è, dato un insieme di punti  $\{(x_i, y_i)\}_{i=0}^n$ , trovare un polinomio che meglio approssima questi dati (detti nodi), ovvero un polinomio tale che  $p_n(x_i) = y_i$ .

**Interpolazione di Lagrange** Data una griglia  $X = \{x_0, x_1, \dots, x_n\}$  con  $x_i < x_{i+1} \forall i \in \{0, 1, \dots, n\}$  Definiamo, sulla griglia  $X$  gli  $n+1$  polinomi di base di Lagrange come:

$$l_i(x) = \prod_{j \neq i} \frac{x - x_j}{x_i - x_j} \quad i \in \{0, 1, \dots, n\}$$

Tali polinomi sono tutti di grado  $n$ . Notiamo che:

$$l_i(x_j) = \delta_{ij} = \begin{cases} 0 & j \neq i \\ 1 & j = i \end{cases}$$

Si dimostra inoltre che tali polinomi sono una base dello spazio vettoriale  $\mathbb{R}[x]_{\leq n}$ .

Il polinomio di interpolazione di Lagrange sarà:

$$p_n(x) = \sum_{i=0}^n y_i \cdot l_i(x)$$

Si dimostra inoltre che tale polinomio è unico.

**Errore** Si considera l'errore come  $e_n(x) = f(x) - p_n(x)$ ,  $x = x_i \implies e_n(x) = 0$ . Inoltre si può dimostrare che:

$$e_n(x) = \prod_{i=0}^n (x - x_i) \frac{f^{(n+1)}(\xi)}{(n+1)!}$$

Si noti che non è vero che  $e_n \xrightarrow{n \rightarrow \infty} 0$ .

**Fenomeno di Runge** Avviene quando si approssimano funzioni a gradini. Per limitare la funzione nodale si scelgono i nodi di Chebychev:

$$x_i = \frac{a+b}{2} + \frac{b-a}{2} \cdot \cos\left(\frac{2k+1}{2(n+1)}\pi\right)$$

### 3 Integrazione numerica

$$I = \int_a^b f(x) dx = F(b) - F(a) \wedge F \in \{f : [a, b] \rightarrow \mathbb{R}\}$$

Formule di quadratura numerica:

$$I \approx Q(f) = \sum_{k=1}^M \omega_k f(x_k)$$

#### 3.1 Regola dei rettangoli

Considero  $f : [a, b] \rightarrow \mathbb{R}$  e divido  $[a, b]$  in  $M$  sottointervalli di uguale ampiezza  $h = \frac{b-a}{M}$ :

$$\Omega_k = [x_{k-1}, x_k]$$

Da cui segue che:

$$I = \int_a^b f(x) dx = \sum_{k=1}^M \int_{\Omega_k} f(x) dx$$

Approssimiamo  $I$  come:

$$I \approx \sum_{k=1}^M f(x_{k-1}) \cdot h = h \sum_{k=1}^M f(x_k)$$

**Convergenza** Dai nostri metodi speriamo che:

$$Q(f) \xrightarrow{h \rightarrow 0} I$$

Ovvero che:

$$err(h) = |I - Q(f)| \rightarrow_{h \rightarrow 0} 0$$

Con  $err(h) \leq C(a, b; f) \cdot h$  Ovvero quando dimezzo  $h$  dimezzo  $err(h)$ . Inoltre, la formula dei rettangoli ha ordine di accuratezza 1. La formula dei rettangoli è una formula ad un punto.

#### 3.2 Regola del punto medio

È analoga alla regola dei rettangoli, ma anziché considerare come altezza  $f(x_k)$ , considero  $f(c_k)$ , con  $c_k = \frac{x_k + x_{k-1}}{2}$ . Quindi:

$$Q(f) = h \sum_{k=1}^M f(c_k)$$

Questo metodo ha ordine di accuratezza 2:

$$err(h) \leq C(a, b; f) \cdot h^2$$

#### 3.3 Formula del trapezio

Considero le medesime premesse dei casi precedenti. Costruisco dei trapezi su ogni intervallo con basi  $f(x_k)$  e  $f(x_{k-1})$ .

$$Q(f) = \frac{h}{2} \sum_{k=1}^M [f(x_{k-1}) + f(x_k)]$$

La formula del trapezio è accurata di ordine 2. La formula del trapezio è a due punti.

### 3.4 Formula di Gauss

Considero i seguenti punti:

$$\begin{aligned}\gamma_{k1} &= x_{k-1} + \left(1 - \frac{1}{\sqrt{3}}\right) \frac{h}{2} \\ \gamma_{k2} &= x_{k-1} + \left(1 + \frac{1}{\sqrt{3}}\right) \frac{h}{2}\end{aligned}$$

Da cui segue che:

$$Q(f) = \frac{h}{2} \sum_{k=1}^M [f(\gamma_{k1}) + f(\gamma_{k2})]$$

Si può dimostrare che il metodo è accurato di ordine 4:

$$err(h) \leq C(a, b; f) \cdot h^4$$

### 3.5 Formula di Simpson

Considero  $f(x_{k-1})$ ,  $f(c_k)$  e  $f(x_k)$ . Su ogni  $\Omega_k$  costruisco quindi la parabola che interpola  $f(x)$ .

$$Q(f) = \frac{h}{6} \sum_{k=1}^M [f(x_{k-1}) + 4f(c_k) + f(x_k)]$$

Questa è una formula a tre punti con accuratezza di ordine 4.

### 3.6 Montecarlo

Al contrario delle formule precedenti, questa è una formula stocastica (non deterministica). Costruisco un rettangolo  $R = [a, b] \times [0, \max_{x \in [a, b]} f(x)]$ . Prendo dei punti casuali nel rettangolo,  $K$  cadranno sotto la funzione e  $N$  sono i totali.

$$Q(f) = \frac{K}{N} A_R$$

Dove  $A_R$  è l'area del rettangolo. Montecarlo è un metodo molto lento, ma il suo ordine di accuratezza non decresce in dimensioni superiori a 1.

## 4 Risoluzione di sistemi lineari $Ax = b$

I metodi per risolvere sistemi lineari sono suddivisi in metodi diretti e iterativi. I metodi diretti vanno a riempire la matrice (non efficienti con matrici sparse). I metodi diretti sono efficaci con sistemi piccoli ( $1000 \times 1000$ ) oppure quando è necessario risolvere molti sistemi del tipo  $Ax = b$ ,  $Ax = c$ ,  $Ax = d$ , ... Negli altri casi convengono metodi iterativi. In tutti i metodi esposti considereremo  $A \in M_{n,n}(\mathbb{R})$ ,  $b, x \in \mathbb{R}^n$

L'idea generale è "splittare" la matrice  $A$ :

$$\begin{aligned} A &= M - N \quad \wedge \quad M \in GL_n(\mathbb{R}) \\ \implies (M - N)x &= b \\ \implies x &= M^{-1}(Nx + b) \end{aligned}$$

Quest'ultima equazione ricorda il metodo del punto fisso: dato  $x^{(0)}$  iniziale definisco  $\forall k \geq 0$ :

$$x^{(k+1)} = M^{-1}(Nx^{(k)} + b)$$

Considero l'errore  $e^{(k)} = x^{(k)} - x$  e verifico se:

$$\begin{aligned} \lim_{k \rightarrow \infty} e^{(k)} &= \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \\ \implies \max_{i=1, \dots, n} |x_i^{(k)} - x_i| &\rightarrow 0 \end{aligned}$$

Per cui servirebbe il concetto di norma infinito  $\|\cdot\|_\infty$ .

$$\begin{aligned} x^{(k+1)} - x &= M^{-1}N(x^{(k)} - x) \\ \implies e^{(k+1)} &= (M^{-1}N)^k e^{(0)} \\ \implies \|e^{(k+1)}\|_\infty &\leq \|M^{-1}N\|_\infty^k \|e^{(0)}\|_\infty \xrightarrow{k \rightarrow \infty} 0 \iff \|M^{-1}N\|_\infty < 1 \end{aligned}$$

### 4.1 Metodo di Jacobi

Prendo  $M$  matrice diagonale con la diagonale di  $A$ ,  $N = M - A$

$$a_{ii}x_i^{(k+1)} = b_i - \sum_{\substack{j=1 \\ j \neq i}}^n a_{ij}x_j^{(k)}$$

Da cui segue, dividendo tutto per  $a_{ii}$

$$x_i^{(k+1)} = \frac{b_i}{a_{ii}} - \frac{1}{a_{ii}} \cdot \sum_{\substack{j=1 \\ j \neq i}}^n a_{ij}x_j^{(k)}$$

Una condizione sufficiente per la convergenza è che  $A$  sia a dominanza diagonale stretta, ovvero:

$$\sum_{\substack{j=1 \\ j \neq i}}^n a_{ij} < a_{ii}$$

Si noti che Jacobi ha bisogno di tenere in memoria 2 passi di vettori soluzioni.

## 4.2 Metodo di Gauss-Seidel

$A = M + N$  con  $M$  triangolare inferiore di  $A$  e  $N$  triangolare superiore con diagonale nulla.

$$Mx^{(k+1)} = b + Nx^{(k)}$$

L'equazione 1 viene risolta come per Jacobi:

$$a_{11}x_1^{(k+1)} = b_1 - \sum_{j=1}^n a_{1j}x_j^{(k)}$$

L'equazione 2 risulta:

$$a_{22}x_2^{(k+1)} = b_2 - \sum_{j=1}^n a_{2j}x_j^{(k)} - a_{21}x_1^{(k+1)}$$

E questo comportamento viene reiterato. Si noti che Gauss-Seidel ha bisogno di tenere in memoria solo un array di soluzioni.

**Criteri di arresto** Questi criteri valgono ovviamente solo per i metodi iterativi.

- numero di iterazioni
- $e^{(k)} < tol$
- check su elementi diagonali per verificare che non siano troppo piccoli

## 4.3 Algoritmo di Gauss

È un metodo diretto per risolvere sistemi lineari. Considero:

$$Ux = b, U \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n$$

Se  $U$  fosse triangolare superiore (o inferiore) la risoluzione risulta immediata:

$$x_n = \frac{b_n}{U_{nn}}$$

In caso il sistema non sia triangolare (a scala), si applicano operazioni elementari di riga fino a ridurre la matrice.

$$Ax = b \rightsquigarrow MAx = Ux = Mb$$
$$A = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \rightsquigarrow M \cdot \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix}$$

Se  $a_{11} \approx 0$  cerco  $\max_{k \in \{2, \dots, n\}} \{a_{k1}\}$ . Trovato questo elemento nella riga  $p$ , scambio la riga 1 con la riga  $p$ . Questa tecnica si chiama *pivoting*. Applicare il pivoting risulta numericamente più stabile. Dopo un'iterazione la matrice diventa:

$$A_2^{(1)} = A_2^{(0)} - m_{21}A_1^{(0)}$$



Con  $m_{21} = \frac{a_{21}}{a_{11}}$ . Per proseguire ad annullare tutti i termini  $a_{n1}$  trovo moltiplicatori del tipo  $m_{n1} = \frac{a_{n1}}{a_{11}}$ . La matrice  $A^{(1)}$  diventa quindi:

$$A^{(1)} = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ 0 & \dots & a_{2n} \\ \vdots & \ddots & \vdots \\ 0 & \dots & a_{nn}^{(1)} \end{bmatrix}$$