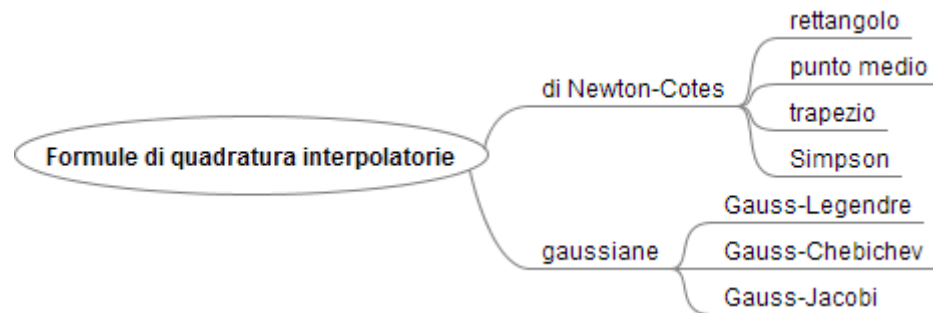
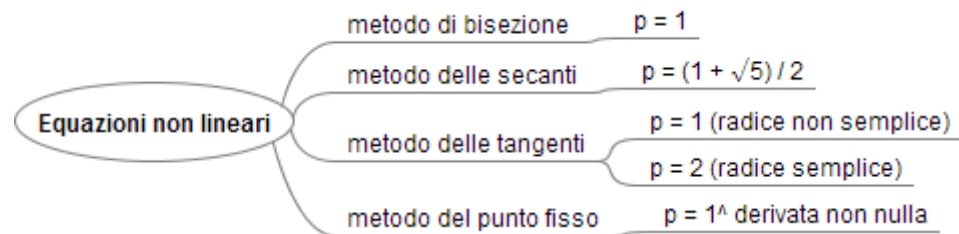
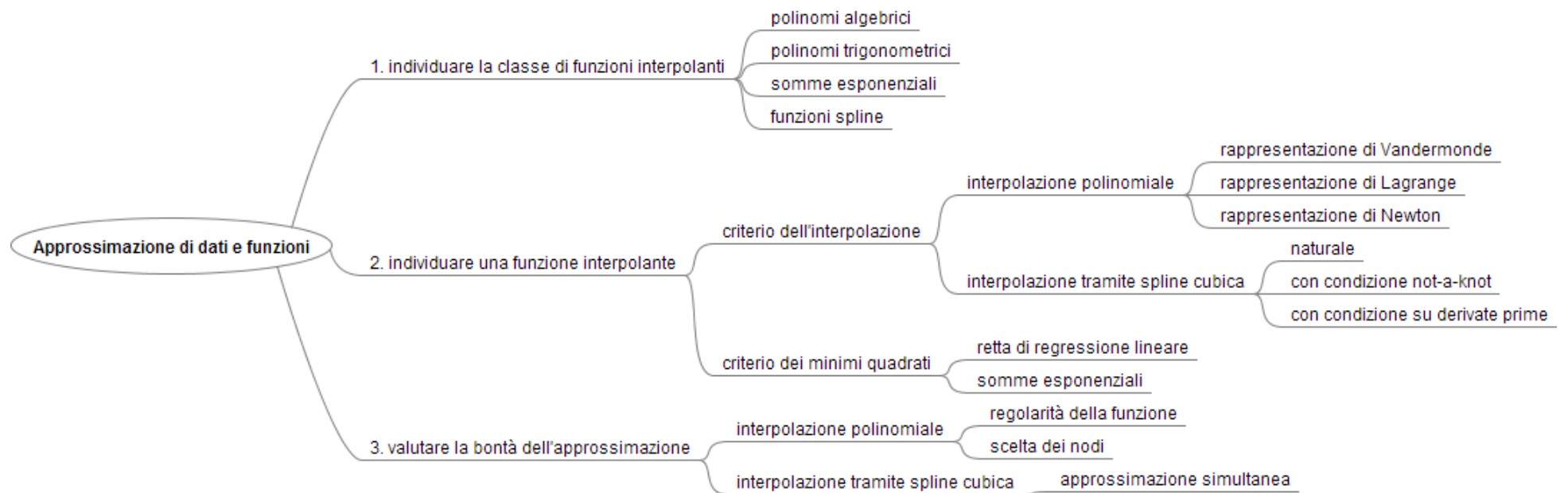


Errori di arrotondamento	Errore assoluto	Errore relativo
		$e_a = a - \bar{a} = p - \bar{p} \cdot N^q$
troncamento $ p - \bar{p} < N^{-t}$	$e_a < N^{q-t}$	$e_r < N^{1-t}$
arrotondamento $ p - \bar{p} \leq \frac{1}{2} N^{-t}$	$e_a \leq \frac{1}{2} N^{q-t}$	$e_r \leq \frac{1}{2} N^{1-t}$

Casi particolari metodi per la risoluzione numerica dei sistemi lineari

diagonale dominante per righe	<ul style="list-style-type: none"> • matrice non singolare → tecnica di sostituzione applicabile • il metodo di Jacobi è convergente • il metodo di Gauss-Seidel è convergente
diagonale dominante per colonne	<ul style="list-style-type: none"> • matrice non singolare → tecnica di sostituzione applicabile • pivoting superfluo • il metodo di Jacobi è convergente • il metodo di Gauss-Seidel è convergente
simmetrica definita positiva	<ul style="list-style-type: none"> • matrice non singolare → tecnica di sostituzione applicabile • pivoting superfluo • vale la fattorizzazione di Choleski • il metodo di Gauss-Seidel è convergente



- **linspace(a, b, n)** genera un vettore di $n + 1$ punti equispaziati nell'intervallo $[a, b]$
- **abs(<numero>)** calcola il valore assoluto
- **norm(A, 2/inf/1)** calcola la norma $2/\infty/1$ del vettore/matrice A
- **zeros(<nrighe>, <ncolonne>)** genera una matrice con tutti gli elementi = 0
- **ones(<nrighe>, <ncolonne>)** genera una matrice con tutti gli elementi = 1
- **eye(<nrighe>, <ncolonne>)** genera una matrice con tutti gli elementi sulla diagonale = 1
- **diag(<vettore>)** genera una matrice inserendo il vettore nella diagonale principale
- **diag(<matrice>)** genera un vettore estraendo la diagonale principale dalla matrice
- **sum(<matrice>, 1)** somma per colonne
- **sum(<matrice>, 2)** somma per righe
- **f = inline(' <espressione in x>')** dichiara una funzione
- **plot(x, y, '<stile>')** genera il grafico con x per ascisse e y per ordinate

2. SISTEMI LINEARI

- **cond(A, 2/inf/1)** calcola il numero di condizionamento in norma 2 del sistema $Ax = b$
- **hilb(n)** genera la matrice di Hilbert di ordine n
- **vander(x)** genera la matrice di Vandermonde associata al vettore x
- **x = A \ b** calcola la soluzione x di $Ax = b$ con il metodo di eliminazione di Gauss con pivoting parziale
- **[L, U, P] = lu(A)** calcola i fattori L, U, P della fattorizzazione $PA = LU$ di A
- **R = chol(A)** calcola il fattore R della fattorizzazione di Choleski $A = R^T R$ della matrice simmetrica definita positiva A

3. APPROSSIMAZIONE

- **p = polyval(c, z)** calcola i valori che il polinomio a coefficienti c assume nei punti di z
- **c = polyfit(x, y, n - 1)** calcola i coefficienti del polinomio di grado $n - 1$ interpolante gli n punti (x_i, y_i) con la matrice di Vandermonde
- **s = spline(x, y, z)** calcola i valori che assume in z la spline cubica interpolante gli n dati (x_i, y_i) secondo la condizione not-a-knot
- **s = spline(x, [yd0 y ydn], z)** calcola i valori che assume in z la spline cubica interpolante gli n dati (x_i, y_i) secondo la condizione sulle derivate prime (yd0 = derivata prima di f in x_0 , ydn = derivata prima di f in x_n)
- **c = polyfit(x, y, n)** calcola i coefficienti del polinomio di grado n interpolante gli m punti (x_i, y_i) secondo il criterio dei minimi quadrati

4. EQUAZIONI NON LINEARI

- **r = fzero(f, x0, toll)** calcola un'approssimazione r, con tolleranza assoluta toll, di una radice dell'equazione $f(x) = 0$ a partire dall'approssimazione iniziale x_0
- **r = roots(c)** calcola gli zeri del polinomio a coefficienti c (a partire da quello del termine di grado massimo)

5. CALCOLO DI INTEGRALI

- **[q, N] = quad(f, a, b, toll)** calcola con la formula composta di Simpson il valore approssimato q dell'integrale $\int_a^b f(x)dx$ con una tolleranza assoluta toll e con N valutazioni di funzione
- **[q, N] = quadl(f, a, b, toll)** calcola con una formula gaussiana il valore approssimato q dell'integrale $\int_a^b f(x)dx$ con una tolleranza assoluta toll e con N valutazioni di funzione