Laboratorio 11

Obiettivi

Approssimare il problema di Cauchy per un SISTEMA di E.D.O.:

f: $\Re^{d+1} \longrightarrow \Re^d$

u'(t) = f(t,u(t)), $t_0 \le t \le T$, $u(t) \in \Re^d$

$u(t_0)$ = v, $v \in \Re^d$

con un metodo **Multi – Passo esplicito**

Metodi multi passo

Assegnati i due polinomi caratteristici di grado p

$$\rho = \sum_{k=0}^{p} \rho_k x^{p-k} \quad , \quad \sigma = \sum_{k=0}^{p} \sigma_k x^{p-k} \, ,$$

un metodo **Multi – Passo** (per ora esplicito) **a p passi con τ fisso** si scrive:

$$\sum_{k=0}^{p} \rho_k U_{n-k} = \tau \sum_{k=0}^{p} \sigma_k f_{n-k}$$

dove si è posto $f_k = f(t_k , U_k)$.

Nota 1: allo scopo di utilizzare notazioni i cui indici coincidano con quelli che si useranno in C++, qui **le notazioni non coincidono con quelle utilizzate dal docente**: qui ρ_0 e σ_0 sono i coefficienti direttivi e non i termini noti. I coefficienti dei polinomi saranno memorizzati in due vettori, ro e sigma. (vedi il file multi_step_coefficienti.cpp allegato, contenente i coefficienti di alcuni metodi di Adams)

Nota 2: p è il numero dei passi all'indietro e il grado dei polinomi. La richiesta che il metodo sia esplicito si traduce nella condizione $\sigma_0 = 0$, ossia sigma[0]=0, ciò farà la differenza con il caso implicito. Per semplicità di programmazione e uniformità con il caso implicito, considereremo σ un polinomio di grado p con coefficiente direttivo nullo.

Esempio

Al metodo a due passi di Adams-Bashforth

$$u_n = u_{n-1} + \frac{\tau}{2}\left[3 f_{n-1} - f_{n-2}\right]$$

Si associano i vettori

ro = [1, -1, 0]

sigma = [0, 3/2, -1/2]

Alcune indicazioni su come impostare il lavoro:

- Può essere conveniente normalizzare i coefficienti di ρ e σ in modo che $\tilde{\rho}_0 = 1$ e, riscrivere il metodo come:

$$U_n = -\sum_{k=1}^{p} \tilde{\rho}_k U_{n-k} + \tau \sum_{k=0}^{p} \tilde{\sigma}_k f_{n-k}$$

- Per metodi con un numero di passi p > 1 sarà necessario calcolare $(p-1)$ valori iniziali aggiuntivi con un metodo diverso: si consiglia un Runge-Kutta di ordine elevato.

- Ad ogni passo si calcola un solo valore nuovo delle f_k, precisamente f_{n-1}, gli altri si salvano dai passi precedenti. Sarà utile avere 2 matrici Old_U e Old_effe che memorizzano i vecchi valori U_k e F_k che servono nei passi successivi.

- È utile avere una funzione che copia un vettore in un altro.

Schema di lavoro per l'implementazione del metodo

NB: in **grassetto rosso** si indica ciò che si intende fare nelle righe sottostanti

Dimensionamenti:

- tutto ciò che serve per un Runge-Kutta esplicito
- ro[p+1], sigma[p+1]
- Old_U[p][d], Old_effe[p][d],
- …

Inizializzazioni:

- t = t_0, u = *v*, …
- ro e sigma (vedi apposita funzione)
- normalizzazione di ro e sigma
- STAMPA t, u

Calcolo dei valori iniziali aggiuntivi con un metodo RK esplicito:

n = 0, …< (p-1) (ciclo ridotto sul tempo)

- memorizzo u in n-esima riga di Old_u
- calcolo f(t,u) memorizzandola in n-esima riga di Old_effe
- eseguo un passo di RK (NB: K[0] coincide con f(t,u) appena calcolata)
- incremento il tempo
- STAMPA t, u

end ciclo ridotto

Vero ciclo sul tempo:

n = p, …., N

- copio u in Old_u alla riga di indice **(n-1)%p**
- calcolo f(t,u) memorizzandola in Old_effe alla riga di indice **(n-1)%p**
- calcolo u nuovo usando i valori in Old_u e Old_effe associando i coefficienti nell'ordine giusto:

  su[d] , sf[d] vettori ausiliari per le due somme

  coefu[passi], coeff[passi] vettori ausiliari dei coefficienti

  for int j =1, … p

  int **k = (n-j)%p**;

  coeff [k] = sigma [j];

  coefu [k] = ro[j];

  end j

  prodotto vettore riga * matrice: coefu*Old_u → su

  prodotto vettore riga * matrice: coeff*Old_f   → sf u = - su + tau*sf; // qui sottinteso un ciclo su componenti vettore

- incremento il tempo
- STAMPA t, u

end ciclo su n

Dopo aver implementato il metodo e verificato il suo funzionamento su un esempio noto svolgere gli esercizi sottostanti

Esercizio 11.1

Inizializzare un metodo multi_passo di ordine p > 2 con un metodo RK di ordine q < (p – 1) applicati ad un problema sufficientemente regolare. Analizzare le conseguenze.

Esercizio 11.2

Approssimare il problema di Dahlquist con un metodo di Adams-Bashforth.. Provare con gli stessi valori di λ e N suggeriti nell'esercizio 8.2. Legare i risultati ottenuti ai domini di stabilità dei metodi (vedi funzione di Matlab *msregstab.m* scaricabile dal sito del corso).

Esercizio 11.3

Approssimare il problema di Dahlquist con λ = i, con il metodo Leap Frog:

$$u_n = u_{n-2} + 2\tau\, f_{n-1}$$

N = 10, 100, 1000, 10000. Visualizzare la traiettoria della soluzione approssimata.

Ripetere le medesime prove per λ = -1.

Mettere in relazione i risultati con il dominio di stabilità del metodo.