Progetto di Calcolo Numerico 2

Žana Ilić - 898373

Luglio 2020

Consideriamo il seguente problema ai valori iniziali:

$$u' = f(t,u) = \begin{pmatrix} -333.4 & 666.6 \\ 333.3 & -666.7 \end{pmatrix} u, \quad u(0) = \begin{pmatrix} 3 \\ 0 \end{pmatrix}$$

Prima vogliamo trovare soluzione esatta di nostro problema lineare e autonomo di dimensione $d = 2$. Troviamo autovalori e autovettori di matrice che chiamiamo A. Calcoliamo il polinomio caratteristico associato ad A:

$$p_A(\lambda) = \det(A - \lambda Id_2) = \begin{vmatrix} -333.4 - \lambda & 666.6 \\ 333.3 & -666.7 - \lambda \end{vmatrix} = \lambda^2 + 1000.1\lambda + 100$$

e vediamo che abbiamo due autovalori negativi $\lambda_1 = -0.1$ e $\lambda_2 = -1000$. Poi troviamo autovettori associati rispettivamente a λ_1 e λ_2:

$$\eta_1 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \quad \eta_2 = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

Soluzione generale del nostro problema ha la forma

$$u = c_1 e^{-0.1t} \begin{pmatrix} 2 \\ 1 \end{pmatrix} + c_2 e^{-1000t} \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

ma se applichiamo valori iniziali otteniamo che $c_1 = 1$ e $c_2 = -1$, cioè la soluzione esatta è:

$$u = e^{-0.1t} \begin{pmatrix} 2 \\ 1 \end{pmatrix} - e^{-1000t} \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

Possiamo scrivere la soluzione esatta per due componenti:

$$\begin{cases} u_1 = 2e^{-0.1t} + e^{-1000t} \\ u_2 = e^{-0.1t} - e^{-1000t} \end{cases}$$

Ora vogliamo trovare un metodo numerico esplicito adeguato per approssimare la soluzione esatta sull'intervallo $I = [0, 3]$. Implementiamo diversi metodi espliciti a un passo che hanno la forma generale:

$$U_{n+1} = U_n + \tau \sum_{i=1}^{s} b_i K_i, \quad K_i = f\left(t_n + c_i\tau, U_n + \tau \sum_{j=1}^{i-1} a_{ij} K_j\right)$$

e ogni metodo ha diversi coefficienti $A \in \mathbb{R}^{sxs}, b \in \mathbb{R}^s, c \in \mathbb{R}^s$ per numero di stadi $s \in \mathbb{N}$. Usiamo i metodi espliciti di Eulero di ordine 1, Heun di ordine 2 e Runge-Kutta di ordine 4 per approssimare la soluzione esatta con i passi temporali fissi:

$$\tau = \frac{1}{10}, \frac{1}{40}, \frac{1}{160}, \frac{1}{640}, \frac{1}{2560}, \frac{1}{10240}.$$

Poichè usiamo passi temporali fissi cioè griglia equidistante e sappiamo che l'intervallo è $I = [t_0, T] = [0, 3]$ e che il numero di passi è pari a $N = (T - t_0)/\tau$, per comodità in codice usiamo numero di passi N invece di valori di passi temporali fissi τ. Abbiamo che, rispettivamente per ogni valore di τ, valori di N sono:

$$N = 30, 120, 480, 1920, 7680, 30720.$$

In tabella vediamo errori finali - la norma 2 della differenza tra soluzione esatta e soluzione approssimata al tempo finale: $\|U_N - u(T)\|$.

N	Eulero Esplicito	Heun	Runge-Kutta 4
30	1.04609e+060	7.22863e+110	inf
120	inf	inf	nan
480	nan	nan	nan
1920	3.88283e-005	2.02242e-009	5.63535e-014
7680	9.70640e-006	1.26532e-010	1.42497e-013
30720	2.42656e-006	7.70450e-012	2.02078e-013

Dalla tabella sopra si può vedere che per valori grandi di passo temporale τ ovvero per i piccoli valori di passi N, errore finale tra soluzione esatta e soluzione approssimata è molto grande per ogni metodo. Invece, per i valori più piccoli di passo temporale τ e rispettivamente per i grandi valori di passi N vediamo che errore è il più piccolo per il metodo di Runge-Kutta di ordine

4. Questo metodo è il metodo migliore per approssimazione di soluzione esatta. Osserviamo anche che in metodo di Runge Kutta errore non diminuisce con lo stesso ordine e anzi cresce, per i valori $N = 7680$ e $N = 30720$. Errore teorico sarà rispettivamente quasi 2.2e-16 e 8.5e-19. Perchè τ decresce molto rispetto alla precisione di macchina, non vediamo questi risultati teorici sulla tabella.

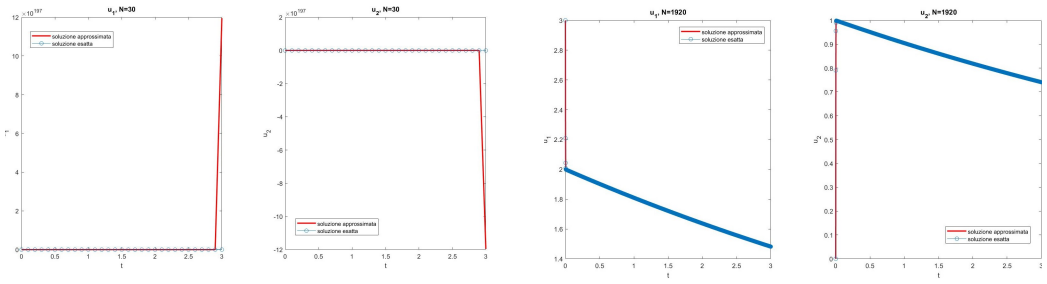

Figura 1: u_1 e u_2 con $\tau = 1/10$ e $\tau = 1/640$

Nelle figure sopra vediamo le due componenti della soluzione approssimata con il metodo di Runge-Kutta di ordine 4 per due diversi passi temporali. In blu si vede la soluzione esatta del problema. In rosso si vede la soluzione approssimata. Per $\tau = 1/10$, la soluzione approssimata di componenete u_1 va a più infinito, quindi abbiamo divergenza dalla soluzione esatta. La soluzione approssimata di componenete u_2 va a meno infinito e diverge dalla soluzione esatta. Concludiamo che per i grandi valori di τ soluzione approssimata dista troppo da soluzione esatta. Quando τ diventa più piccolo, soluzione approssimata ed esatta sono sovrapposte per entrambe i componenti, come per $\tau = 1/640$.

Ora vogliamo applicare i stessi metodi espliciti per approssimare la soluzione esatta ma ora in modo diverso. Vogliamo prima avere 512 passi di lunghezza $\tau = 1/10240$ che sono seguiti da passi di lunghezza $\tau = 1/10$. Nella figura sotto vediamo il comportamento di due componenti della soluzione approssimata in questo modo. La soluzione trovata è un po meglia di quella trovata con soli passi di lunghezza $\tau = 1/10240$, ma ancora dista troppo da soluzione esatta.

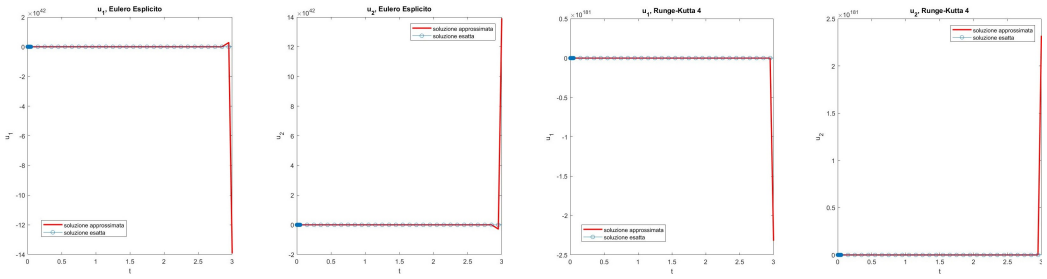

Figura 2: u_1 e u_2 con 512 passi di lunghezza $\tau = 1/10240$ e 30 passi di lunghezza $\tau = 1/10$ con metodo di Eulero Esplicito e Runge-Kutta 4

Alla fine vogliamo usare metodi di Runge-Kutta adattivi che usano due metodi, uno di ordine p e secondo di ordine $p-1$. Questi due metodi hanno passaggi comuni, cioè hanno la stessa forma, con uguale matrice A e vettore c ma con diversi vettori, b e $\hat{b}$. Usiamo il metodo di Eulero-Heun 2(1) e di Runge-Kutta-Fehlberg 5(4). Approssimando la soluzione con questi metodi, ad ogni passo vogliamo trovare passo temporale adattato in modo tale che errore stimato in tale passo, $err_n \approx \tau \| \sum_i^s (b_i - \hat{b}_i) K_i \|$, rimanga minore di tolleranza predefinita. Se errore è più grande della tolleranza, passo viene ripetuto con passo temporale più piccolo, che si ottiene moltiplicandolo per una costante uguale a $\sqrt[p]{toll/err_n}$. Se errore è minore della tolleranza, lunghezza di passo temporale viene aumentata per risparmiare tempo.

Approssimiamo il nostro problema con due metodi a passo adattivo. Nella tabella sotto possiamo vedere numeri di passi utilizzati ed errori finali ottenuti per diverse tolleranze. Poi approssimiamo il problema ancora una volta con metodi a passo fisso di ordine p - metodo di Heun e metodo di Fehlberg 5. Il numero di passi utilizzati deve essere uguale alle valutazioni di f in metodo adattivo diviso per numero di stadi del metodo. Così possiamo confrontare errori ottenuti con il metodo adattivo e il metodo a passo fisso.

toll	num. passi EE/Heun	errore EE/Heun	valutazioni di f	num. passi Heun	errore Heun
1e-3	1576	0.00018	3154	1577	2.99e-009
1e-6	3881	1.53e-007	7764	3882	4.96e-010
1e-9	88308	4.31e-011	176618	44154	3.85e-012

toll	num. passi Fehlberg54	errore Fehlberg54	valutazioni di f	num. passi Fehlberg5	errore Fehlberg5
1e-3	822	0.00021	4950	825	6.96e-014
1e-6	841	4.76e-008	5064	844	7.89e-014
1e-9	919	8.69e-011	5532	922	6.90e-014

Possiamo concludere che in entrambi i casi, il metodo a passo fisso ha errore di ordine di grandezza meno rispetto all'altro metodo. Metodi con passo adattivo sono più comodi, perchè così non dobbiamo cercare un numero di passi appropriato come per i altri metodi a passo fisso. Ma in questo caso vediamo che una volta che sappiamo il numero di passi neccessari, metodo a passo fisso, più precisamente, metodo di Fehlberg 5, con il numero di passi più piccolo ha errore finale il più piccolo. Nella figura sotto si vedono le due componenti della soluzione approssimata con il metodo di Runge-Kutta-Fehlberg 5(4) usando la tolleranza 1e-6.

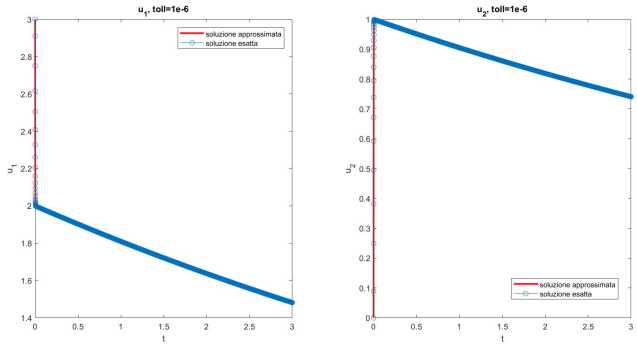

Figura 3: Metodo di Runge-Kutta-Fehlberg 5(4) con tolleranza 1e-6

Per quanto riguarda il codice, ho diviso il progetto in tre parti. All'inizio si può scegliere cosa si vuole fare. Scelta di metodo esplicito oppure adattivo, numero di stadi s e ordine p di metodo adattivo si scrivono "a mano" direttamente in funzione `main`. Il programma stampa i valori di tempo e di soluzione approssimata a video e su file di nome `progetto.dat`. Alla fine il programma stampa a video errore finale in norma due. Nel caso in cui si risolve terza parte dell'programma, stampa a video anche numero di valutazioni di f e numero di passi di relativo metodo a passo fisso. Si usa il codice Matlab `progetto.m` per stampare i vari grafici.