Simulazione dell'interazione tra due specie utilizzando le equazioni di Lotka-Volterra

Marco Biancalani, Oscar Fontanelli

30 August 2024

1 Modifiche rispetto alla versione precedente

Nella precedente versione del programma si era palesato un problema nel caso in cui, durante la simulazione, una delle due specie, o entrambe, si estinguessero. In tal caso, l'integrale primo risultava infinito dall'istante in cui avveniva l'estinzione in poi, quando in verità esso dovrebbe risultare impossibile, dato che si tratta di calcolare il logaritmo di argomento nullo, come si vede dalla formula riportata nel prossimo capitolo.

Per ovviare a questo problema abbiamo detto al programma di porre arbitrariamente l'integrale primo uguale a 0 nel caso in cui una delle specie diventasse estinta, un'altra soluzione poteva essere quella di far scrivere al programma "undefined" ma questo avrebbe portato a grossi problemi nel caso in cui si fosse voluto fare il grafico dei risultati ottenuti.

2 Introduzione teorica

Le equazioni di Lotka-Volterra servono a descrivere l'interazione tra due specie, la prima di prede, supponendo che esse abbiano una quantità illimitata di cibo, la seconda di predatori, la quale si nutre delle prede. Le equazioni sono valide sotto la condizione che siano solo due le specie presenti. Le due equazioni sono di seguito riportate:

$$\frac{dx}{dt} = (A - By(t))x(t)$$
$$\frac{dy}{dt} = (Cx(t) - D)y(t)$$

Dove x(t) ed y(t) sono, rispettivamente, il numero di prede ed il numero di predatori al tempo t, mentre i parametri stanno ad indicare, nel caso di A e C, quanto rapidamente riescano a riprodursi le due specie nel caso in cui esse abbiano trovato una fonte di sostentamento, B e D indicano invece il loro tasso di mortalità. Le due equazioni possono essere discretizzate per facilitare la costruzione di un programma che le utilizzi:

$$x_i = x_{i-1} + (A - By_{i-1})x_{i-1}\Delta t$$

$$y_i = y_{i-1} + (Cx_{i-1} - D)y_{i-1}\Delta t$$

Vengono definiti "Punti di equilibrio" della simulazione i seguenti:

$$e_1 = (0, 0)$$

$$e_2 = (\frac{D}{C}, \frac{A}{B})$$

Se la simulazione si trova in uno di questi due punti, il numero di prede e predatori rimane costante. Essi vengono ottenuti ponendo uguale a 0 le due equazioni differenziali di partenza. Un altro valore importante, che dovrebbe rimanere costante nel tempo è quello dell'integrale primo:

$$H(x, y) = -Dln(x) + Cx + By - Aln(y)$$

3 Struttura dell'archivio

L'archivio consegnato conterrà sette file principali:

- Relazione.pdf, quello che state leggendo in questo momento.
- main.cpp che contiene il Main, è la parte fondamentale del codice, permette di inserire parametri e valori iniziali in imput e in base a ciò calcolare i valori della simulazione
- lotka_volterra.hpp che contiene la dichiarazione dei vari metodi e attributi all'interno di una classe chiamata "Simulation".
- lotka_volterra.cpp, che contiene la definizione dei vari metodi contenuti nel file .hpp
- lotka_volterra_grapich1.cpp, che utilizza il programma Root per creare un grafico a dispersione in cui i punti hanno coordinata $(x(t), y(t))$, i punti si orienteranno nello stesso modo intorno al punto di equilibrio e_2 a prescindere dal valore dei parametri.
- lotka_volterra_grapich2.cpp sfrutta ancora Root, questa volta per creare un grafico che descriva l'andamento della popolazione delle due specie e del valore di H in funzione del tempo.
- lotka_volterra.test.cpp che contiene il programma di testing.

4 Compilazione e utilizzo

Per calcolare i valori della simulazione usare il comando:

g++ -Wall -Wextra -Wpedantic -Wconversion -Wsign-conversion -Wshadow -Wimplicit-fallthrough -Wextra-semi -Wold-style-cast -D_GLIBCXX_ASSERTIONS -fsanitize=address,undefined lotka_volterra.cpp main.cpp -o valuecalculator.

Una volta compilato, eseguire valuecalculator con il comando ./valuecalculator .Il programma vi chiederà di fornire in input i valori dei 4 parametri, la popolazione iniziale delle due specie e la durata della simulazione. Il programma ritornerà un file ValuesList.txt, in cui troverete 4 colonne di valori. La prima colonna conterrà l'indicazione sull'istante di tempo esaminato, accanto ad ogni valore di essa si troverà, nell'ordine, il numero di prede, di predatori, e l'integrale primo nell'istante di tempo specifico.

Utilizzando successivamente le due macro Root e il file ValuesList.txt, oltre al file e_2Coordinates.txt, ritornato anche questo dal programma, che contiene le coordinate del punto e_2, è possibile ottenere due rappresentazioni grafiche della simulazione, come riportato nella seconda sezione. Entrambe le rappresentazioni possono essere fatte sia in versione statica, in cui i punti vengono inseriti tutti in contemporanea, sia in versione dinamica, in cui i punti vengono posizionati nel tempo. Per utilizzare le macro aprite Root su MobaXTerm e caricate quella desiderata, usando il comando .L lotka_volterra_graphic1.cpp oppure .L lotka_volterra_graphic2.cpp. In seguito invocate la funzione main(). Root prenderà i dati per i grafici direttamente dal file ValuesList.txt e dal file e_2Coordinates.txt, quindi essi dovranno essere accessibili dal programma. Dopo aver scelto quale tipo di grafico si vuole visualizzare, root disegnerà il grafico e, una volta finita l'operazione, scriverà sul terminale "Simulation completed". Nel caso si volessero eseguire entrambe le macro sarà necessario riavviare Root.

Per quando riguarda l'utilizzo dei test, usare il comando di compilazione g++ -Wall -Wextra -Wpedantic -Wconversion -Wsign-conversion -Wshadow -Wimplicit-fallthrough -Wextra-semi -Wold-style-cast -D_GLIBCXX_ASSERTIONS -fsanitize=address,undefined lotka_volterra_tests.cpp lotka_volterra.cpp -o lotka_volterra.t.cpp. Eseguire poi scrivendo ./lotka_volterra.t.cpp

5 Funzionamento del programma

Il programma si basa sulla memorizzazione della popolazione delle due specie e dell'integrale primo per ogni istante di tempo, per fare ciò vengono utilizzati quattro accumulatori di tipo "vector". I vector delle due popolazioni vengono inizializzati alla popolazione iniziale che l'utente ha fornito tramite terminale, il vector relativo al tempo viene inizializzato a 0, metre il vettore relativo all'integrale primo viene inizializzato al valore dell'integrale all'istante $t = 0$. Successivamente il programma usa il metodo evolve() per calcolare la variazione delle due popolazioni, in base alle equazioni discretizzate riportate nell'Introduzione, e dell'integrale primo, dopo una variazione di tempo $\Delta t = 0.001$ i valori ottenuti vengono poi aggiunti ai vector. Un ciclo for permette di ripetere il metodo evolve() per ogni variazione di tempo Δt fino a che non si raggiunge l'istante di tempo indicato dall'utente. I valori ottenuti dal ciclo vengono via via aggiunti con un "pushback" ai vari vector, nel caso del vector per il tempo non si fa altro che andare avanti di un passo Δt. Quando l'ultimo istante di tempo raggiunge il tempo t fornito dall'utente il ciclo si arresta ed i 4 vector vengono trascritti in un file di testo chiamato, come detto in precedenza, ValuesList.txt. Oltre a

ciò il programma calcolerà anche il punto di equilibrio e_2 e lo trascriverà in un file chiamato e_2Coordinates.txt.

6 Precisazioni sui dati di input/output

Il programma è stato istruito sul rigettare qualsiasi dato in input che sia minore di 0, ma è comunque bene, al fine di ottenere dati soddisfacenti e veritieri, assegnare dei valori realistici, soprattutto ai parametri. I valori solitamente sono:

$$0,1 < A < 1,5; 0 < B < 10^{-3}; 0,1 < C < 1; 10^{-3} < D < 10^{-2}$$

Per quanto riguarda i dati di output, potrete notare che spesso l'integrale primo oscilla leggermente, questo fenomeno si nota particolarmente nel caso del secondo grafico, questo "difetto" è dovuto alla scarsa precisione del metodo di calcolo dell'integrale primo, calcolandolo con metodi più potenti sarebbe più possibile verificarne la costanza.

7 Testing

Per quanto riguarda i test abbiamo creato vari test case, nel primo abbiamo fornito dei parametri per i quali nessuna delle specie si estingue, siamo andati a verificare che i valori di x,y ed H siano quelli attesi in vari istanti di tempo, oltre a verificare le coordinate di e_2, viene poi eseguito un altro test con parametri diversi ma con risultato uguale al primo, ovvero che le due specie coesistono indefinitamente.

Un altro test verifica che, inseriti opportuni parametri, dopo un tempo sufficientemente lungo le prede si estinguono, e di conseguenza i predatori, trovandosi in assenza di fonti di sostentamento.

Il quarto test simula invece il caso opposto, ovvero quello in cui i predatori si estinguono, sempre se forniti opportuni parametri e dopo sufficiente tempo, in questo caso le prede, trovandosi in assenza di minacce, continueranno ad aumentare indefinitamente.

Il quinto test invece riguarda un punto di equilibrio, è stato inserito per verificare che, in caso il numero iniziale di prede e predatori sia 0, esso non cambi.