Laboratorio di Informatica T (Ch7)

Sistemi Dinamici Tempo-Discreti

(Old) News

L'anno scorso, questi signori hanno vinto un premio Nobel

Michael Rosbash, Jeffrey C. Hall, and Michael W. Young

Ritmo Circadiano

Ma cosa hanno fatto?

Hanno identificato i meccanismi fondamentali dietro il ritmo circadiano

- Il ritmo circadiano è il nostro "orologio biologico"
- Ha un periodo di circa 24 ore (circa-diem)
- Può adattarsi a segnali esterni (e.g. luce)

Più in dettaglio, come funziona?

- Una proteina "X" si accumula durante la notte…
- …E viene smaltita durante il giorno

Ma come fa ad invertirsi l'andamento? La reazione è autocatalitica

- La proteina "X" è contemporaneamente un reagente ed un prodotto
- La proteina "X" inibisce la propria produzione

Oscillatore di Van der Pol

Il ritmo circadiano è un esempio di oscillatore biologico

- Le vere reazioni che lo caratterizzano sono complesse…
- …Ma un modello semplificato è alla nostra portata!

Oscillatore di Van der Pol (discretizzato)

Consideriamo una versione discretizzata dell'oscillatore di Van der Pol:

$$x^{(t+1)} = x^{(t)} + hy^{(t)}$$
$$y^{(t+1)} = y^{(t)} + h\left(\mu(1 - x^{(t)2})y^{(t)} - x^{(t)}\right)$$

- x è la variabile che corrisponde alla proteina, y è ausiliaria
- Hanno un valore diverso per ogni istante di tempo t
- Le equazioni ci dicono come passare dallo stato al tempo t…
- …Allo stato al tempo $t + 1$

Sistemi Dinamici Tempo Discreti

Si dice sistema dinamico un sistema che ha uno stato che varia nel tempo

- Nel nostro caso, lo stato sono le due variabili x e y…
- …Che variano nel tempo per passi discreti

Per questa ragione, il nostro sistema dinamico si dice tempo-discreto

- Verso la fine del corso parleremo di sistemi dinamici tempo-continui…
- …Che sono definiti da equazione differenziali!

Sistemi Dinamici Tempo Discreti

Un sistema dinamico tempo discreto è caratterizzato dall'equazione:

$$x^{(t+1)} = f(x^{(t)})$$

- x è la variabile di stato (può essere vettoriale!)
- f si dice funzione di transizione

Per simulare l'andamento dello stato basta invocare ripetutamente f:

$$x_{cur} = x_0$$
$$\textbf{for } t \in \{1..n\}$$
$$\quad X_t = x_{cur} \text{ (memorizzo lo stato corrente)}$$
$$\quad x_{cur} = f(x_{cur})$$

- x_0 è il valore iniziale dello stato, x_{cur} quello corrente
- Alla fine, la matrice X contiene il risultato

Esempio: Oscillatore di Van Der Pol

Esempio: Oscillatore di Van Der Pol

Usiamo come esempio il nostro oscillatore:

$$x^{(t+1)} = x^{(t)} + hy^{(t)}$$
$$y^{(t+1)} = y^{(t)} + h\left(\mu(1 - x^{(t)2})y^{(t)} - x^{(t)}\right)$$

- Se conosciamo h, μ ed i valori iniziali di x e y…
- …Possiamo simulare l'andamento dello stato…
- …Utilizzando l'algoritmo appena visto

Esempio: Oscillatore di Van Der Pol

Per l'implementazione, iniziamo dalla funzione di transizione:

```
function Xf = f(X, h, mu)
    % "Spacchetto" lo stato (per comodità)
    x = X(1);
    y = X(2);
    % Calcolo lo stato futuro
    Xf(1) = x + h * y;
    Xf(2) = y + h * (mu * (1 - x^2)*y - x);
end
```

- `X` dovrà contenere lo stato corrente…
- …Nel nostro caso, un vettore con x e y
- La funzione restituisce i due elementi dello stato futuro in `Xf`

Esempio: Oscillatore di Van Der Pol

Quindi possiamo effettuare la simulazione in uno script:

Prima definiamo i dati del problema:

```
clear all

% Dati del problema
x0 = 1;
y0 = 1;
h = 0.05;
mu = 2;

...
```

Esempio: Oscillatore di Van Der Pol

Quindi possiamo effettuare la simulazione in uno script:

Poi effettuiamo la simulazione vera e propria:

```
...
% Simulazione
X = [];
T = 1:1000; % Istanti di tempo
xc = [x0, y0]; % Stato iniziale
for t = T
    X(t, :) = xc; % Salvo lo stato corrente
    xc = f(xc, h, mu); % Genero lo stato futuro
end
...
```

Esempio: Oscillatore di Van Der Pol

Quindi possiamo effettuare la simulazione in uno script:

Infine disegniamo l'andamento delle grandezze che ci interessano

```
...
% "Spacchetto" le due componenti dello stato
x = X(:, 1); % Prima colonna
y = X(:, 2); % Prima colonna

% Disegno l'andamento nel tempo
plot(T, x)
```

Potete trovare il codice nello start-kit

Esercizio: Comportamento dell'Oscillatore

Esercizio: Comportamento dell'Oscillatore

Studiate il comportamento dell'oscillatore

Il codice è disponibile nello start-kit nel file `es_van_der_pol.m`

- Cosa succede con i valori dei parametri suggeriti?
- Cosa succede variando x_0 e y_0?
- Cosa succede facendo crescere μ (in $[1, 6]$)?
- Cosa succede ponendo $\mu = 1$ e $h = 0.315$?
  - Suggerimento: simulate solo 300 iterazioni in questo caso

Soluzione

Risposte:

- Cosa succede con i valori dei parametri suggeriti?
  - Il sistema ha un comportamento periodico
- Cosa succede variando x_0 e y_0?
  - Niente! Il ciclo è robusto rispetto a variazioni dello stato iniziale
  - Si dice che il sistema ha un ciclo limite
- Cosa succede facendo crescere μ (in $[1, 6]$)?
  - Il ciclo diventa più "squadrato": l'oscillatore di Van der Pol…
  - …è nato come un modello del battito cardiaco!
- Cosa succede ponendo $\mu = 1$ e $h = 0.315$?
  - Emergono delle irregolarità non predicibili
  - Il comportamento diventa caotico

Esempio: Random Walk (2D)

Esempio: Random Walk (2D)

Consideriamo una particella in moto:

- La particella si muove su un piano (sx, dx, su, giù)
- La particella si muove per passi discreti $(+1, -1)$
- Il moto è soggetto a perturbazioni casuali (agitazione termica)
- Ad ogni passo la particella si sposta a sx/dx/su/giù con la stessa probabilità

Si modelli il moto come un sistema dinamico

- Si assuma che la particella parta in posizione `(0,0)`
- Si proceda ad effettuare una simulazione
- Si disegni la traiettoria della particella su un grafico
  - Traiettoria = sequenza degli stati visitati
  - In questo caso è semplicemente la sequenza delle posizioni

Esercizio: Random Walk (2D)

Nel file di script `random_walk2.m`, definite la funzione:

```
function xf = f(xc)
```

- Che calcoli lo stato futuro **`xf`** a partire da quello corrente **`xc`**
- Lo script invocherà ripetutamente f per simulare lo spostamento

Come scegliere la direzione?

- Possiamo usare **`rand`** per generare un numero v casuale in $(0, 1)$
- Se $v \in [0, 0.25)$ andiamo (e.g.) a sx
- Se $v \in (0.25, 0.5]$ andiamo (e.g.) su, etc.
- Così ogni direzione ha esattamente il 25% di probabilità!

Trovate il codice della soluzione nello start-kit

Esercizio: Modello di Ricker

Esercizio: Modello di Ricker

Modello di Ricker

È un sistema dinamico tempo-discreto caratterizzato dall'equazione:

$$x^{(t+1)} = x^{(t)} e^{r\left(1-\frac{x^{(t)}}{N}\right)}$$

È nato per modellare l'evoluzione di una popolazione di pesci

- x è il numero di individui
- r è un tasso di crescita
- N è la popolazione sostenibile

Il sistema può essere simulato in modo analogo a quanto visto

- Lo stato è uno scalare, il che rende le cose più semplici

Esercizio: Modello di Ricker

Il modello di Ricker può assumere una varietà di comportamenti:

Partite dal file `es_ricker.m` nello start-kit

- Definite la funzione (corrispondente alla funzione di transizione):

```
function xf = f(x, r, N)
```

- Nello script, simulate (e disegnate) 100 passi

Determinate per via empirica per quali valori di r:

- Il sistema converge ad uno stato stabile, senza oscillare
- Il sistema converge ad uno stato stabile, ma dopo aver oscillato
- Il sistema ha un comportamento periodico
- Il sistema ha un comportamento caotico

Esempio: Random Walk (1D)

Esempio: Random Walk (1D)

Consideriamo una particella in moto:

- La particella si muove su una linea (e.g. sx, dx)
- La particella si muove per passi discreti $(+1, -1)$
- Il moto è soggetto a perturbazioni casuali (agitazione termica)
- Ad ogni passo la particella si sposta a sx o dx con la stessa probabilità

Si modelli il moto come un sistema dinamico

- Si assuma che la particella parta in posizione 0
- Si proceda ad effettuare una simulazione
- Si disegni l'andamento della posizione nel tempo
  - In 1D, la traiettoria sarebbe poco leggibile!

Esercizio: Random Walk (2D)

Nel file di script `random_walk1.m`, definite:

La funzione di transizione:

```
function xf = f(xc)
```

- Che calcoli lo stato futuro `xf` a partire da quello corrente `xc`
- Lo script invocherà ripetutamente f per simulare lo spostamento

Per scegliere la direzione, usiamo la tecnica vista in `random_walk2.m`

- Stavolta è più facile: va scelta una direzione sola!

Esercizio: Modello di Beverton-Holt

Esercizio: Modello di Beverton-Holt

Sia data una popolazione che cresce secondo il modello

$$x^{(t+1)} = \frac{r\, x^{(t)}}{1 + \frac{x^{(t)}}{N}}$$

Dove:

- r indica il tasso di crescita (deve valere $r \in [1.0, 2.0]$)
- N indica un valore di popolazione…
- …che, se raggiunto, dimezza il tasso di crescita

Il modello è nato per applicazioni simili a quello di Ricker

Si implementi un simulatore per il modello

Esercizio: Modello di Beverton-Holt

In particolare, partite dal file `es_beverton_holt.m` nello start-kit:

Definite la funzione di transizione:

```
function xf = f(x, r, N)
```

- Che calcoli lo stato futuro **`xf`** a partire da quello corrente **`x`**
- Nello script, simulate (e disegnate) 100 passi

Studiate per via empirica il comportamento del sistema:

- Per quali valori di r la popolazione cresce?
- Per quali collassa?
- Il sistema può oscillare?
- Il sistema può assumere comportamento periodico?
- Il sistema può assumere comportamento caotico?

Esercizio: Linear Congruential Generator

Esercizio: Linear Congruential Generator

Il computer non gioca a dadi

- In un elaboratore tipico, ogni operazione è deterministica
- Dati gli stessi operandi, produce lo stesso risultato

Allora come fa Matlab a generare numeri casuali?

Risposta: barando! :-)

- Matlab usa una successione numerica con valori difficili da predire
- …E che soddisfano alcune proprietà statistiche
- La successione parte da un numero iniziale (detto "seme")…
- …Che tipicamente è l'orario corrente ;-)

Si parla di numeri pseudo-casuali

Esercizio: Linear Congruential Generator

Come esempio consideriamo il Linear Congruential Generator

È semplicemente la successione data da:

$$X_{n+1} = (aX_n + c) \ \mathbf{mod} \ m$$

- Con X_i, a, c ed m interi; il valore di partenza X_0 è noto

Ricordate che $x \mod y$ è il resto della divisione intera tra x e y

- In Matlab lo potete ottenere con `mod(x, y)`
- In caso di dubbi, usate `help`

Esercizio: Linear Congruential Generator

Nel file `es_lcg.m` si definisca la funzione:

```
function R = my_lcg(X0, N)
```

- Che, dato il valore (scalare) di X_0...
- ...Generi gli elementi da X_1 a X_N della successione...
- ...E li salvi nel vettore di ritorno `R`
- Si consideri $m = 16, a = 9, c = 3$

Nello script, si effettuino dei test:

- Si provi ad eseguire `my_lcg` in sequenza con diversi valori di `X0`

I valori prodotti possono sembrare casuali...

- Ma provate a disegnarli e ad aumentare il numero di steps...
- Noterete che in realtà la successione ha un periodo :-)

Esercizio: Modello di Shepherd

Esercizio: Modello di Shepherd

Sia data una popolazione che cresce secondo il modello

$$x^{(t+1)} = \frac{r\, x^{(t)}}{1 + \left(\frac{x^{(t)}}{N}\right)^2}$$

Dove:

- r indica il tasso di crescita (positivo)
- N indica un valore di popolazione…
- …che, se raggiunto, dimezza il tasso di crescita

Il modello è nato per applicazioni simili a quello di Ricker

Si implementi un simulatore per il modello

Esercizio: Modello di Shepherd

In particolare, partite dal file `es_shepherd.m` nello start-kit:

Definite la funzione di transizione:

```
function xf = f(x, r, N)
```

- Che calcoli lo stato futuro **`xf`** a partire da quello corrente **`x`**
- Nello script, simulate (e disegnate) 100 passi

Studiate per via empirica il comportamento del sistema:

- Per quali valori di r la popolazione si azzera?
- Per quali si assesta sul valore N?
- Il sistema può oscillare?
- Il sistema può assumere comportamento periodico o caotico?

Esercizio: Trasposizione di Matrice

Esercizio: Trasposizione di Matrice

Nel file `es_transpose.m` si definisca la funzione:

```
function B = my_transpose(A)
```

- Che calcoli la trasposizione della matrice **`A`**
- Si confrontino i risultati con l'operatore di trasposizione di Matlab

Esercizio: Conteggio di Elementi

Esercizio: Conteggio di Elementi

Nel file `es_count.m`, si definisca la funzione ausiliaria:

```
function [U, C] = my_count(V)
```

Che, dato un vettore di ingresso `V`, restituisca `U` e `C` tali che:

- `U` contenga gli elementi distinti di `V`
- Per ogni valore `v` in `U`, la cella corrispondente di `C`…
- …Contenga il numero di occorrenze di `v` in `V`

I.e. la funzione deve contare le occorrenze di ogni elemento.

- Si verifichi il funzionamento nella funzione principale
- Si utilizzi un vettore definito a mano

Suggerimento: potete ottenere `U` con la funzione `unique` di Matlab

Esercizio: Prodotto Cumulativo

Esercizio: Prodotto Cumulativo

Matlab fornisce la funzione:

```
function P = cumprod(V)
```

- Che in ogni elemento `P(ii)` del vettore restituito...
- ...riporta il prodotto degli elementi di `V` negli indici da `1` a `ii`

Quindi, per esempio:

```
cumprod([2, 4, 6]) % denota [2, 8=2*4, 48=2*4*6]
```

Esercizio: Prodotto Cumulativo

Nel file di funzione `es_cumprod.m`, si definisca una funzione:

```
function P = my_cumprod(V)
```

- Che replichi il comportamento di **`cumprod`**

Si verifichi la correttezza nella funzione principale:

- Si utilizzino dei vettori di numeri casuali
- Si confrontino i risultati con quelli di **`cumprod`** in Matlab

Esercizio: Bubble Sort

Esercizio: Bubble Sort

“Bubble sort” un algoritmo semplice per ordinare un vettore

- L’algoritmo scorre ripetutamente un vettore
- Ad ogni passaggio, si scambiano gli elementi consecutivi fuori ordine
- Si procede finché non sono più necessari scambi

Ricordate l’esempio dell’ordinare l’aula nella prima lezione?

- Bubble sort è l’algoritmo che avevamo pensato per risolverlo!

Esercizio: Bubble Sort

Ecco un possibile pseudo-codice

Sia dato un vettore V non necessariamente ordinato di lunghezza n

$$
\begin{array}{l}
\text{ordinato} = \texttt{false} \\
\textbf{for } j \in \{1..n\} \text{ (al più } n \text{ passaggi)} \\
\quad \textbf{for } i \in \{1..n-1\} \text{ (} n \text{ è la lunghezza, mi fermo un passo prima)} \\
\qquad \textbf{if } V_i > V_{i+1} \\
\qquad\quad \text{scambia gli elementi}
\end{array}
$$

- Ci sono varianti (molto) più efficienti, ma questa andrà bene per noi

La difficoltà è capire come effettuare lo scambio

- Suggerimento: potrebbe fare comodo introdurre una variabile di appoggio

Esercizio: Bubble Sort

Nel file `es_bubble.m`, definite la funzione:

```
function V = my_bubble(V)
```

- Che ordini **`V`** utilizzando bubble sort…
- …E quindi restituisca **`V`** stesso (vedi nome della variabile di ritorno)

Verificate la correttezza nella funzione principale `es_bubble`:

- Utilizzate vettori casuali
- Confrontate il risultato con quello della funzione predefinita **`sort`**:

```
function W = sort(V)
```