Modello di Fitzhugh-Nagumo e Studio della Sincronizzazione in una Rete di Hopfield

Azzani Alessandro, Carichini Daniele

11 maggio 2022

1 Introduzione

La prima modellizzazione del comportamento di un neurone valse il nobel per la fisiologia ad Alan Lloyd Hodgkin ed Andrew Huxley nel 1963. La complessità di questo modello è tale da non consentirne una risoluzione analitica, finora è stato studiato con successo grazie a numerose simulazioni numeriche confrontate con i dati sperimentali.

Una notevole semplificazione rispetto al modello di Hodgkin e Huxley è quello proposto da Richard FitzHugh nel 1961, chiamato modello di FitzHugh-Nagumo. Nella seguente tesina si presenta tale modello, dopodiché si studia il problema della sincronizzazione dei neuroni in un modello ulteriormente semplificato su una rete di Hopfield. La sincronizzazione dei neuroni è un problema interessante dal punto di vista medico poiché collegata a vari disturbi neurologici, come l'epilessia o il parkinson. Come ultimo obiettivo si avrà quindi quello di studiare la tendenza di una rete alla sincronizzazione in funzione di certi parametri in modo da simulare il comportamento della rete neuronale.

2 Modello di FitzHugh-Nagumo

Il *modello di FitzHugh-Nagumo* (FHN), è stato proposto per la prima volta nel 1961 da Richard FitzHugh [1] per simulare il comportamento di un sistema eccitabile, cioè un neurone. Può essere pensato come una proiezione bidimensionale le modello di Hodgkin-Huxley o come una versione modificata dell'oscillatore di Van Der Pol. Il circuito equivalente fu introdotto nel 1962 da Jin-ichi Nagumo, Suguru Arimoto, and Shuji Yoshizawa [2].

Il modello FHN descrive perciò il processo di depolarizzazione della membrana cellulare di un neurone, che provoca un improvviso potenziale elettrico lungo la membrana chiamato potenziale d'azione o spike e descritto dal modello biologico di un neurone [3].

Il comportamento del modello è descritto da due equazione differenziali ordinarie non-lineari, caratterizzate da due diverse scale di tempo:

$$\begin{cases} \epsilon \dot{u}(t) = u(t) - \frac{u^3(t)}{3} - v(t) + I_{ext}(t) \\ \dot{v}(t) = u(t) + a - bv(t) \end{cases} . \tag{1}$$

u rappresenta il potenziale sulla membrana cellulare ed è la variabile veloce, mentre v è la variabile lenta e introduce la refrattarietà del neurone dopo aver sparato. La differenza tra le due scale di tempo è introdotta da ϵ. I_{ext} è invece lo stimolo esterno, che nel caso di una rete neurale è il segnale proveniente dagli altri neuroni tramite le connessioni chiamate *sinapsi*. Infine a e b sono parametri del modello che regolano il regime in cui opererà il sistema.

Entrando più in dettaglio, per comprendere meglio il comportamento descritto dall'Equazione 1 è utile rappresentare le *nulcline* in un piano uv. In questo caso le nulcline sono le due curve lungo cui la derivata temporale delle variabili u e v sono nulle (senza considerare lo stimolo esterno I_{ext}):

$$\begin{cases} u - \frac{u^3}{3} - v = 0 \\ u + a - bv = 0 \end{cases} \implies \begin{cases} v = u - \frac{u^3}{3} \\ u = bv - a \end{cases} .$$

Perciò le nulcline sono rispettivamente una cubica e una retta che si intersecano in un punto fisso (u^*, v^*), che è stabile solo se $|u^*| > 1$ e diventa instabile per $|u^*| < 1$, caso in cui invece il circuito tende a un ciclo limite stabile. In Figura 1 sono rappresentate le nulcline e la loro intersezione nel punto fisso del sistema nel regime eccitabile, ossia quando $|u^*| > 1$.

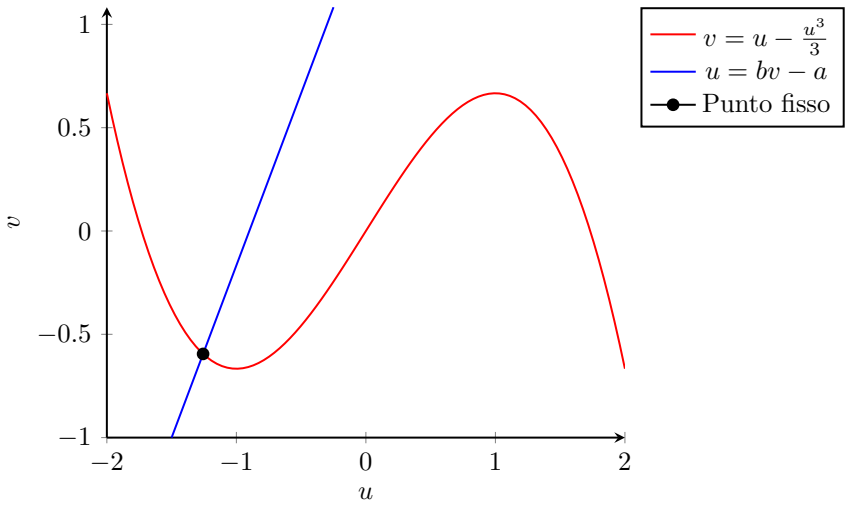


Figura 1: Spazio delle fasi vu raffigurante le nulcline del modello FHN. Sono stati posti $a = 0.9$ e $b = 0.6$.

Analizziamo ora come queste nulcline descrivono il moto del sistema. La nulclina di u (funzione rossa in Figura 1) ha un massimo in $(1, 2/3)$ e un minimo in $(-1, -2/3)$. Questi due punti dividono la curva in tre rami:

- ramo refrattario (a sinistra del minimo),
- ramo repulsivo (tra il minimo e il massimo),
- ramo attivo (a destra del massimo).

Immaginiamo che il sistema sia inizialmente fermo nel punto fisso (u^*, v^*) e che riceva uno stimolo esterno I_{ext}. Questo stimolo lo immaginiamo per semplicità positivo e limitato nel tempo, in modo da avere $\dot{u} > 0$ e da spostare il sistema dal punto fisso. Visto che le due variabili sono descritte da due scale di tempo diverse ed essendo $\epsilon \ll 1$, u sarà molto più veloce di v, quindi assumiamo che u si muova lungo una retta orizzontale nel piano vu, ossia a v costante.

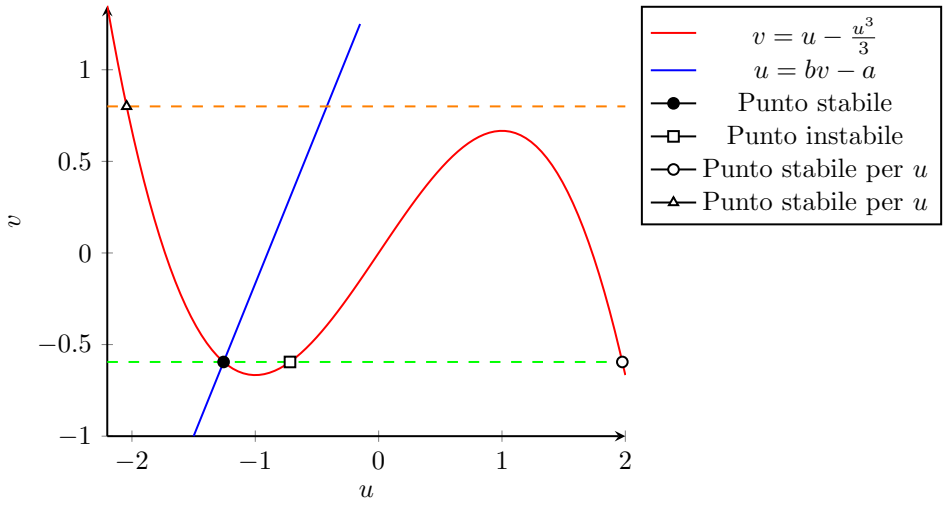


Figura 2: Andamento della variabile u per due diversi valori di v fissati.

Tramite lo stimolo esterno quindi il sistema tende a spostarsi orizzontalmente lungo la linea verde in Figura 2. Se lo stimolo I_{ext} è abbastanza intenso, allora il sistema riuscirà a superare il punto fisso instabile nel ramo repulsivo della nulclina di u e dirigersi verso il punto fisso sul ramo attivo. A questo punto u avrà raggiunto la sua nulclina e quindi avrà derivata temporale nulla; v, invece, tenderà a muoversi più lentamente lungo la nulclina di u in modo da avvicinarsi alla sua nulclina, ossia inizierà a muoversi sul ramo attivo verso l'alto. Chiaramente questo è possibile finché $v < \frac{2}{3}$, dopodiché il punto avrà superato il massimo e la variabile u non sarà più in equilibrio e si muoverà nuovamente lungo una retta orizzontale a v costante. Sta volta però u tenderà a muoversi verso sinistra lungo la retta arancione in modo da raggiungere l'unico punto stabile esistente quando $v > \frac{2}{3}$, ossia quello sul ramo refrattario. Infine, v tenderà

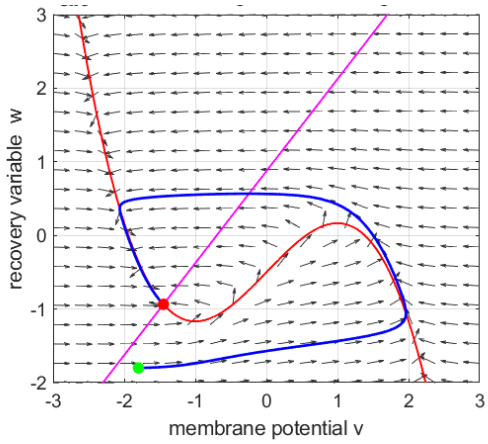


Figura 3: Dinamica del sistema nel piano vu

nuovamente a spostarsi lentamente verso la sua nulclina, che in questo caso vuol dire verso il basso fino a tornare sul punto stabile iniziale intersezione delle due nulcline. Si noti come lo stimolo esterno ha l'effetto di attivare il sistema solo quando si trova abbastanza vicino al punto stabile e quando I_{ext} è abbastanza intenso. Se queste due condizioni non sono soddisfatte, il sistema potrebbe non riuscire a "superare" il ramo repulsivo e quindi ad attivare il neurone. La dinamica del sistema nello spazio delle fasi vu è riportata in Figura 3.

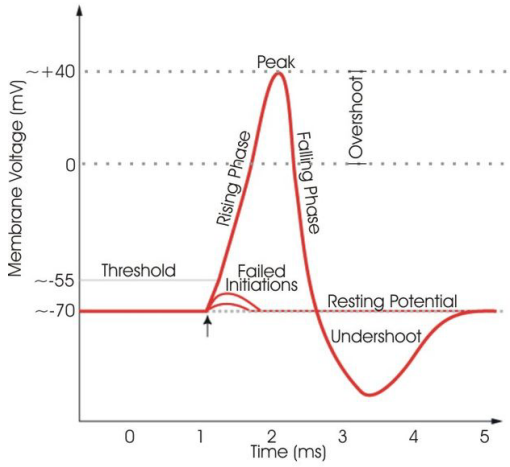


Figura 4: Spike del potenziale di membrana di un neurone

Infatti u è la variabile che rappresenta il potenziale di membrana del neurone. In Figura 4 è possibile apprezzare l'andamento nel tempo della variabile u in seguito ad uno stimolo esterno. Si noti come, se lo stimolo non è abbastanza

intenso non riesce ad attivare il neurone che quindi torna velocemente nella condizione iniziale. Se invece lo stimolo esterno riesce a superare il valore di soglia, allora il neurone si attiva e "spara", generando una spike. In seguito al raggiungimento del picco di potenziale però, il neurone avrà un tempo refrattario durante il quale, anche se sottoposto ad uno stimolo esterno, non riuscirà ad attivarsi.

3 Introduzione ai Grafi

Un *grafo* $\mathcal{G}$ è definito da un insieme di nodi $\mathcal{V}$ e di link $\mathcal{E}$. Un link (i, j) è una connessione tra l'i-esimo nodo v_i e il j-esimo v_j. Un grafo può essere *indiretto* se $(i, j) \in \mathcal{E} \iff (j, i) \in \mathcal{E}$, altrimenti è detto *diretto*. Nell'implementazione della rete cerebrale nella simulazione sono stati usati solo grafi diretti, quindi ci concentreremo su questi ultimi. Un esempio di grafo diretto è riportato in Figura 5.

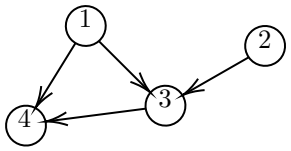


Figura 5: Grafo diretto

Dato un grafo, è possibile associargli la *matrice di adiacenza* A, definita come segue:

$$A_{ij} = \begin{cases} 1 & (i, j) \in \mathcal{E} \\ 0 & \text{altrimenti} \end{cases}.$$

Nel caso del grafo in Figura 5, A sarà

$$A = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

È possibile definire i *vicini* di un nodo v_i come l'insieme di nodi ai quali è connesso v_i. In altre parole $v_j \sim v_i$, cioè v_j è vicino di v_i, se e solo se $(i, j) \in \mathcal{E}$. Per esempio, nel grafo in Figura 5, i vicini di v_1 saranno v_3 e v_4, quindi $v_1 \sim v_3$ e $v_1 \sim v_4$.

Il *grado* di un nodo invece è il numero di vicini che il nodo ha. Nel caso di un grafo diretto, è possibile distinguere tra *in degree* e *out degree*, che sono, rispettivamente, il numero di connessioni entranti nel nodo e uscenti dal nodo. Per esempio, nel grafo in Figura 5 si avrà $\text{in-deg}(v_3) = 2$ e $\text{out-deg}(v_3) = 1$. Valgono inoltre le seguenti formule per calcolare l'in degree e l'out degree:

$$\text{in-deg}(v_i) = \sum_j A_{ji} \qquad \text{out-deg}(v_i) = \sum_j A_{ij}.$$

È inoltre possibile associare dei *pesi* w_{ij} ai link (i, j). Questi pesi rappresentano l'importanza della connessione tra i due nodi. Nel caso di un grafo con link pesati, spesso invece che scrivere la matrice di adiacenza A, si preferisce scrivere la *matrice dei pesi* W, dove $W_{ij} = w_{ij}$.

Un grafo è detto *bipartito* se l'insieme dei vertici $\mathcal{V}$ è divisibile in due insiemi disgiunti e indipendenti $\mathcal{V}_1$ e $\mathcal{V}_2$, tale che per ogni link $(i, j) \in \mathcal{E}$, si ha $v_i \in \mathcal{V}_1$ e $v_j \in \mathcal{V}_2$. In Figura 6 è mostrato un esempio di grafo bipartito.

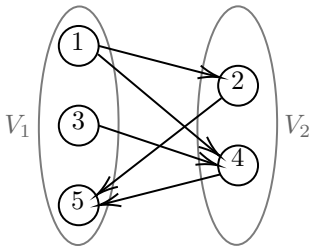


Figura 6: Grafo bipartito

Per quanto riguarda un grafo clusterizzato, non esiste una definizione rigorosa. Possiamo informalmente affermare che un grafo è *clusterizzato* quando l'insieme dei nodi $\mathcal{V}$ è formato da sottoinsiemi di nodi disgiunti $\mathcal{V}_1, \ldots, \mathcal{V}_n$, tali che i nodi dentro lo stesso sottoinsieme, detto *cluster*, sono altamente connessi rispetto alle connessioni tra i cluster. In Figura 7 è riportato un esempio di grafo diviso in tre cluster.

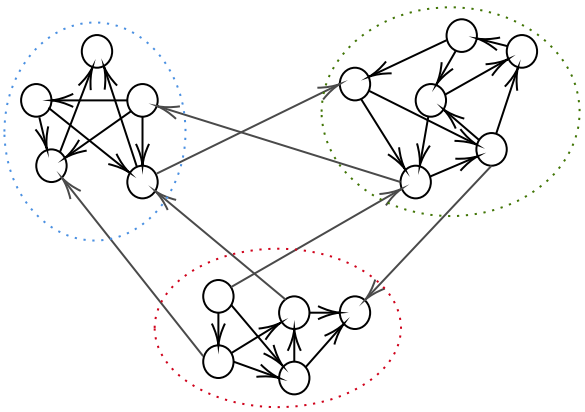


Figura 7: Grafo clusterizzato

4 Reti di Hopfield

Una *rete di Hopfield* è un tipo di rete neurale artificiale, nota per essere il modello di rete che simula le capacità del cervello umano di ricordare le cose

o di ricostruire le immagini distorte. Il fatto che dato uno stato iniziale la convergenza della rete ad un attrattore sia assicurata consente la definizione di un'*energia* associata ad ogni stato che tende ad un minimo nell'evoluzione. Questo fatto però non verrà discusso nella seguente tesi, in quanto considereremo delle reti di hopfield come modellizzazione di una rete di neuroni cerebrali per lo studio della loro sincronizzazione.

Il neurone nel modello di Hopfield è uno stato binario: 1 quando è acceso, 0 altrimenti. L'intera rete può essere rappresentata come un grafo pesato (si veda Sezione 3) i quali nodi sono i neuroni ed i link le connessioni tra essi. In questa rappresentazione a grafo ogni nodo avrà perciò associato uno stato binario s rappresentante lo stato del neurone e la dinamica di ogni nodo sarà data dalla seguente formula:

$$s_i(t+1) = \Theta\left(\sum_{v_j \sim v_i} w_{ji} s_j(t) - \Delta\right) \tag{2}$$

Dove Θ è la funzione di Heaviside e Δ è la soglia (il segnale minimo che un neurone deve ricevere per accendersi).

Il segno dei pesi degli edge determinerà se il neurone agirà da eccitatore o da inibitore, ovvero se tenderà ad accendere o spegnere i suoi vicini.

5 Implementazione del Codice

In questa Sezione sono discusse le principali scelte che sono state prese durante l'implementazione del codice. Innanzitutto, è stata creata una classe `Neuron` che contenesse le caratteristiche principali di ogni neurone, cioè `state`, che rappresenta lo stato del neurone, e `clock`, che è un timer che non permette al neurone di cambiare stato in modo da simulare sia lo stato attivo sia lo stato refrattario della dinamica del sistema descritto dal modello di FHN. Dopodichè il programma è stato implementato in modo tale da permettere simulazioni con reti con strutture a grafi classici, clusterizzati o bipartiti.

5.1 Grafo Classico

Il grafo è rappresentato tramite la matrice di adiacenza `adj` nella classe `Graph`. Ci sono sei parametri principali che definiranno la struttura del network e anche la sua dinamica:

- `n_nodes`: è il numero di nodi, e quindi di neuroni, della rete. Per la simulazione è posto intorno a 1000.
- `in_degree`: rappresenta il numero medio dell'in degree di ogni nodo. Per rendere il network più realistico si è scelto di far variare l'in-degree tra `in_degree - 2` e `in_degree + 1` tramite una distribuzione uniforme di interi. Si è scelto di porre `in_degree = 5` durante le simulazioni.

- `time_active`: è il tempo (ossia il numero di step) dopo l'attivazione durante i quali il neurone rimane attivo ($s = 1$) indipendentemente dal segnale che riceve dai suoi vicini. È stato posto `time_active = 2`.

- `time_passive`: è il periodo refrattario del neurone, cioè il numero di step che devono passare affinché il neurone sia pronto a ricevere un nuovo segnale. Durante questo periodo il neurone rimane quindi spento ($s = 0$). Durante le simulazioni è stato posto `time_passive = 1`. Il periodo della dinamica di un neurone è quindi `time_active` + `time_passive` = 3.

- `retard`: è il tempo che un segnale impiega a percorrere i dendriti, cioè a raggiungere il neurone successivo. Per ragioni che saranno chiare più avanti, `retard` deve essere sempre maggiore del periodo di un neurone, quindi si è posto `retard = 3`.

- `EI`: rappresenta il rapporto tra connessioni eccitatorie ($w > 0$) e inibitorie ($w < 0$), ossia il numero E/I. I pesi delle connessioni, infatti, sono tipicamente generati tramite una distribuzione uniforme di numeri reali tra `w_min = -0.5` e `w_max = 1`, e quindi con un rapporto $E/I =$ `w_max`$/|$`w_min`$| = 2$. Nel caso in cui si voglia inserire un diverso rapporto E/I, allora `w_max` verrà sovrascritto come `w_max` $=$ `EI` $\cdot$ $|$`w_min`$|$.

Questi sono i parametri che servono per generare il grafo e definire la dinamica del sistema. Infatti, verrà generata una matrice dei pesi `adj n_nodes` $\times$ `n_nodes`. Nel riempirla vengono estratti in modo casuale sia i due neuroni tra cui inserire una connessione sia il peso della connessione tramite una distribuzione uniforme tra `w_min` e `w_max`. Chiaramente si controlla di intercorrere in casi come quelli mostrati in Figura 8, cioè si controlla di non avere un link che connetta lo stesso nodo e di non avere due link tra gli stessi nodi.

Inoltre è stata implementata la funzione `normalize()` che normalizza l'output di ogni neurone tramite la formula

$$w'_{ij} = \frac{w_{ij}}{\sum_j w_{ij}}$$

dove w_{ij} rappresenta il peso della connessione tra l'i-esimo neurone e il j-esimo.

Un'altra componente che determina pesantemente l'evoluzione del sistema è il modo di iniziallizare il segnale. Nelle simulazione si usa la funzione `activate()`, che prende come variabile in ingresso `prob`, che rappresenta la probabilità che

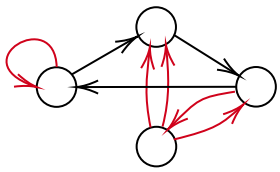


Figura 8: Casi evitati nella creazione del grafo

un neurone sia attivo al primo step. Inoltre, quando un neurone viene attivato al primo step, anche il suo `clock` non viene posto sempre uguale a `time_active`, ma estratto casualmente tra 0 e `time_active`, in modo da rendere più realistica anche l'inizializzazione.

Per quanto riguarda l'evoluzione della rete, ogni neurone evolve secondo il seguente processo: se é attivo resta attivo per un numero di iterazioni pari a `time_active` inviando il segnale ai neuroni vicini con pesi dati dai rispettivi elementi della matrice di adiacenza `adj`. Una volta che é passato un numero di iterazioni pari a `time_active` il neurone si spegne e resta tale per un tempo refrattario pari a `time_passive` iterazioni. Come già specificato, il ruolo di contare tali iterazioni è svolto dalla variabile `clock`, interna alla classe `Neuron`. Se invece il neurone é spento con $\texttt{clock} = 0$, dovrà percepire il segnale in entrata ad un numero di iterazioni precedenti pari a `retard` aggiornandosi seguendo la formula:

$$s_i(t) = \Theta\left(\sum_{v_j \sim v_i} w_{ji} s_j(t - \texttt{retard})\right) \tag{3}$$

Si noti perciò che il sistema dovrà avere una memoria degli stati passati pari a `retard`. Nel programma implementato l'intera dinamica di un'iterazione è trattata dalla funzione `next_step()`.

La misura della sincronizzazione ad un determinato step è eseguita dalla funzione `get_sync()`. La sincronizzazione `sync` della rete è calcolata attraverso la formula:

$$\texttt{sync} = \frac{\sum_i^N 2s_i - 1}{N}$$

dove s_i è lo stato dell'i-esimo neurone durante una certa iterazione e N è il numero di neuroni nella rete. In questo modo, se la rete ha raggiunto una sincronizzazione abbastanza buon vedremo la variabile `sync` alternasi tra valori vicini a -1 e 1.

Un'altra misura di sincronizzazione è eseguita da `get_average_sync()`, la quale restituisce la media del valore assoluto della sincronizzazione calcolata da `get_sync()` su un periodo del neurone, ossia su un numero di `time_active` + `time_passive` iterazioni.

Infine, siccome `get_average_sync()` non riesce a distinguere tra il caso di una completa sincronizzazione o completo spegnimento della rete, si è deciso di introdurre anche la funzione `get_activation_sync()`. Questa funzione restituisce, dopo aver fatto trascorrere un certo numero di iterazioni in modo da aver raggiunto una situazione periodica, il numero massimo di neuroni che si accendono simultaneamente (che passano quindi da uno stato spento a uno stato acceso), meno il numero di neuroni che invece si accendono in un altro momento diviso `time_active`+`time_passive`-1, tutto normalizzato per il numero totali di neuroni della rete. Scritto in formula, se $a_1, a_2, \ldots, a_n$ rappresentano il numero di neuroni che si accendono simultaneamente durante una certa iterazione,

$$\begin{pmatrix}
0 & 0 & 0 & 6.6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -148 & 0 & 0 & -143 & 0 & 0 & 0 \\
6.7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -149 & 0 & 0 \\
0 & 8.0 & 9.8 & 0 & 0 & 0 & 0 & -144 & 0 & 0 & -146 & 0 \\
0 & 0 & -147 & 0 & 0 & 0 & 6.4 & 0 & 0 & -142 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-148 & 0 & -147 & 0 & 0 & 0 & 0 & 8.6 & 0 & 0 & 0 & -144 \\
0 & 0 & 0 & 0 & 8.3 & 9.3 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -150 & 0 & 0 & 0 & 0 & 0 \\
-142 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6.6 & 0 & 0 & 9.9 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -142 & 0 & 8.1 & 0 & 0 \\
0 & -149 & 0 & 0 & -150 & 0 & 0 & 0 & 0 & 0 & 7.3 & 0
\end{pmatrix}$$

Figura 9: Esempio di matrice di adiacenza di grafo con 3 cluster da 4 neuroni ognuno. Le sottomatrici sono evidenziate con colori diversi. L'in degree è stato posto uguale a 1 e le connessioni tra i cluster a 5.

con $n = \texttt{time\_active} + \texttt{time\_passive}$, allora

$$\texttt{activation\_sync} = \frac{\max(a_1, a_2, \ldots, a_n) - \frac{\sum a_i - \max(a_1, a_2, \ldots, a_n)}{n-1}}{N},$$

dove N rappresenta il numero di neuroni della rete.

5.2 Grafo Clusterizzato

Per trattare grafi clusterizzati è stata implementata la classe `Cluster`, figlia della classe `Graph`. In questo modo si sono potute riutilizzare tutte le funzioni e le variabili appartenenti alla classe madre utili anche per la figlia, come ad esempio `next_step()`. Le nuove variabili in gioco sono tre in più:

- `num_clusters`: è il numero di clusters nel grafo.
- `nodes_in_cluster`: è il numero di neuroni in ogni cluster, che quindi sarà uguale per tutti.
- `inter_connections`: é la connettività tra cluster, ovvero il numero di link tra elementi appartenenti a diversi sottogruppi, anch'esso fissato.

La matrice di adiacenza di un grafo clusterizzato appare come una matrice quadrata di lato `num_clusters` $\times$ `nodes_in_cluster`, sulla cui diagonale sono visibili le matrici di adiacenza dei vari sottogruppi, che hanno elementi tutti positivi. Gli elementi esterni a tali sottomatrici sono per lo più nulli e rappresentano le connessioni tra i clusters, le quali sono solitamente poche e fortemente inibitorie. un esempio di matrice di adiacenza di grafo clusterizzato generato dal programma è riportato in Figura 9. La funzione di generare la matrice di adiacenza è svolta da `cluster_adj()`.

Un'altra funzione interna a `Cluster` che va a modificare la matrice di adiacenza è `cycle()`, la quale fa in modo che i cluster formino un anello con le

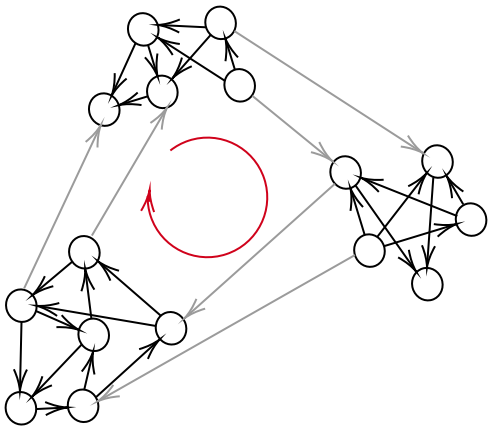


Figura 10: Grafo clusterizzato connesso ad anello

interconnessioni. Inoltre inverte i segni di tutti i link, rendendo le connessioni interne ad un sottogruppo negative e le connessioni esterne fortemente positive, come mostrato in Figura 10.

Per quanto riguarda la sincronizzazione, la funzione d'interesse è `print_sync()`, la quale stampa il valore della sincronizzazione già definita per il grafo classico, per ogni sottogruppo.

5.3 Grafo Bipartito

La classe `Bipartite` serve a trattare i grafi bipartiti. È classe figlia di `Graph`, perciò condivide tutti i suoi metodi. Chiaramente però la matrice di adiacenza verrà generata in modo diverso da `bipartite_adj()`. Anche in questo caso la matrice di adiacenza si presenterà con delle sottomatrici, che però saranno solo 2 e saranno disposte fuori dalla diagonale, come mostrato in Figura 11.

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 19 & 13 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 13 & 17 & 0 \\ 0 & 0 & 0 & 0 & 16 & 0 & 0 & 11 \\ 0 & 0 & 0 & 0 & 0 & 0 & 15 & 18 \\ 0 & 13 & 0 & 12 & 0 & 0 & 0 & 0 \\ 0 & 0 & 16 & 16 & 0 & 0 & 0 & 0 \\ 13 & 0 & 18 & 0 & 0 & 0 & 0 & 0 \\ 15 & 20 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Figura 11: Matrice di adiacenza per un grafo bipartito con 8 nodi ed in degree uguale a 2

Inoltre è differente il modo di calcolare la sincronizzazione nel caso di un grafo bipartito. Infatti, se S_1 e S_2 rappresentano la sincronizzazione nel primo sottoinsieme di nodi e nel secondo rispettivamente, allora la sincronizzazione totale si calcolerà come

$$\texttt{sync} = \frac{S_1 + S_2}{2}.$$

È stata anche implementata la funzione `print_syncs()` che restituisce i due valori della sincronizzazione per i due sottoinsiemi di nodi, calcolata come descritto in Sezione 5.1.

Inoltre è stata implementata una diversa funzione per l'inizializzazione, chiamata `bias_init()`, che tratta i due sottoinsieme di nodi e li inizializza in modo diverso con due differenti probabilità di accensione per ogni neurone.

6 Risultati

Nella seguente Sezione sono riportate le scelte principali ed i risultati ottenuti nelle simulazioni.

6.1 Rete a Grafo Classico

Per prima cosa si è provato a capire come la tendenza dei neuroni nella rete a sincronizzarsi dipendesse da alcuni parametri della rete stessa. Infatti, a seconda dei valori dei parametri che definiscono la rete, la dinamica poteva essere attratta verso una dinamica più sincronizzata, come nel caso di Figura 12, oppure verso una dinamica più casuale, come riportato in Figura 13. Per visualizzare qualitativamente il comportamento della rete, infatti, si è deciso di estrarre casualmente 40 neuroni e visualizzare la loro dinamica per un numero di iterazioni limitato, rappresentandoli con un quadratino giallo nel caso in cui durante quella iterazione fossero accesi o con un quadratino blu altrimenti.

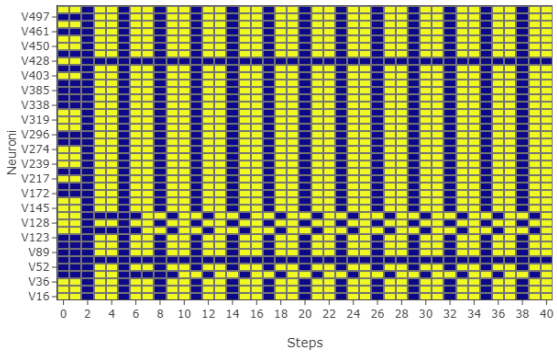


Figura 12: Esempio di dinamica sincronizzata

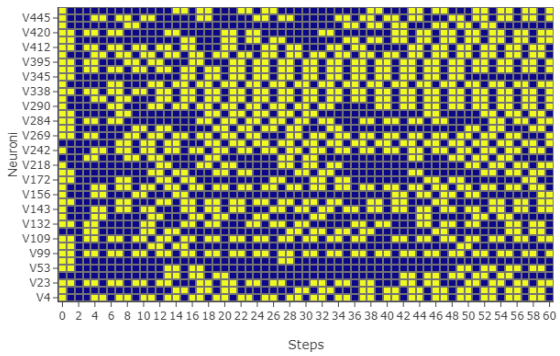


Figura 13: Esempio di dinamica casuale

Il primo parametro che si è voluto considerare è stato E/I, cioè il rapporto tra connessioni eccitatorie ($w > 0$) e inibitorie ($w < 0$) del grafo. Infatti, nello studio [4] è stato dimostrato che il rapporto E/I è costante per molti neuroni della corteccia cerebrale di topo e quindi anche dell'essere umano, dato che l'architettura fondamentale è fortemente conservata in tutti i mammiferi. Questo rapporto è cruciale per mantenere un equilibrio nel cervello. Nell'articolo viene anche affermato che disturbi come l'epilessia, l'autismo o la schizofrenia potrebbero dipendere, almeno in parte, dal mantenimento di un ottimale rapporto E/I. "Se questo rapporto è alterato, si altera anche la percezione del mondo del soggetto, che potrebbe non essere più in grado di regolare l'enorme flusso di stimolazioni che arrivano al cervello in una giornata normale" ha dichiarato Massimo Scanziani, autore principale dello studio. "Come conseguenza, il soggetto stesso potrebbe essere sopraffatto dalle stimolazioni o viceversa del tutto insensibile: ciò sarebbe evidente soprattutto nelle interazioni sociali, che richiedono una regolazione fine degli stimoli".

Per provare a studiare la tendenza della rete a sincronizzarsi in funzione del coefficiente E/I, si è deciso di raccogliere i dati dalle simulazioni in modo da poter disegnare un grafico che mostri l'andamento della sincronizzazione in funzione di E/I. In particolare, per ogni valore del coefficiente E/I è stato generato un grafo di 200 nodi, avente `in_degree` $= 6$. Sono riportati i grafici sia per la sincronizzazione calcolata tramite `get_average_sync()`, sia tramite `get_activation_sync()`, chiamate rispettivamente S_{media} e S_{attiv}. Per entrambi i grafici, in modo da far dipendere i risultati il meno possibile dall'inizializzazione iniziale, sono stati acquisti i valori di sincronizzazione dopo 300 step, quando quindi il sistema ha molto probabilmente raggiunto una dinamica periodica, ed è stata riportata la media tra 10 simulazioni con stessi parametri ma diversa inizializzazione data comunque da `activate()`, settando la probabilità di attivazione di un neurone nello stato iniziale a 20%. I risultati sono riportati in Figura 14.

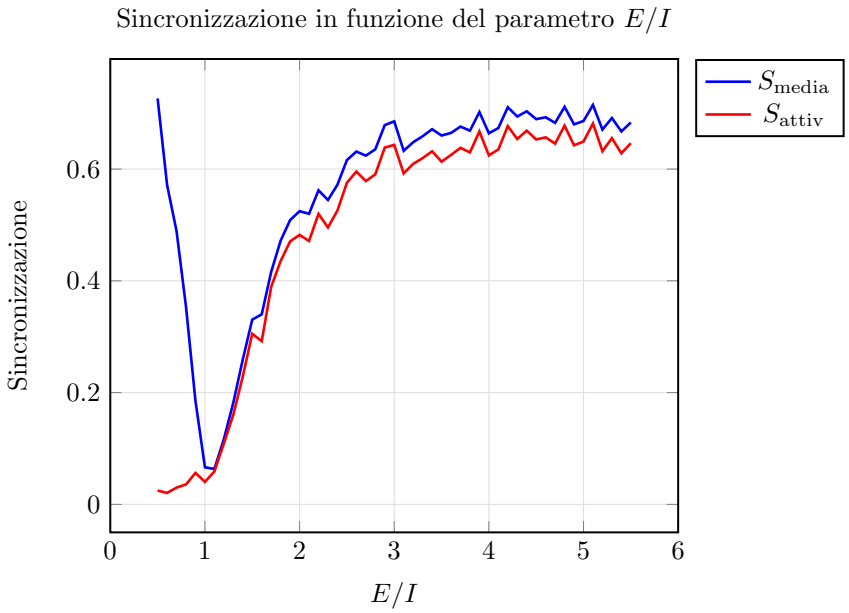


Figura 14: Grafico della media dei due diversi modi di calcolare la sincronizzazione raggiunta dalla rete, in funzione di diversi valori del parametro E/I

Si noti che siccome S_{media} calcola la media dei valori assoluti della sincronizzazione in un periodo, per valori di $E/I < 1$ restituisce numeri elevati poiché, essendo presenti più connessioni inibitorie che eccitatorie, la maggior parte dei neuroni sarà spenta, la sincronizzazione sarà quindi minore di zero ma contribuirà comunque nel calcolo di S_{media}. L'altro modo di misurare la sincronizzazione della rete invece non riporta questo problema, perciò anche quando molti neuroni non partecipano alla dinamica e rimangono spenti, S_{attiv} sarà molto vicina allo zero. Inoltre, a partire da $E/I \cong 1$, la sincronizzazione inizia ad aumentare rapidamente.

Essendo una tendenza che ricorda fortemente un'esponenziale, si è deciso di provare a fittare i dati con la funzione

$$f(x) = p_0 - p_1 \exp(-(x - p_2)),$$

dove p_0, p_1 e p_2 sono i parametri della funzione e x rappresenta il parametro E/I. In questo caso è stata calcolata la media per 15 simulazioni di reti per ogni valore del rapporto E/I, e tra questi 15 campioni è stata anche calcolata la deviazione standard, che è una misura che suggerisce quanto la rete, in quella particolare configurazione, dipenda dall'inizializzazione casuale. I risultati del fit sono riportati in Figura 15.

Successivamente si è tentato quindi di verificare quanto la sincronizzazione raggiunta dalla rete potesse dipendere dall'inizializzazione imposta, in particolare dalla percentuale di neuroni attivati nello stato iniziale. Quindi è stato

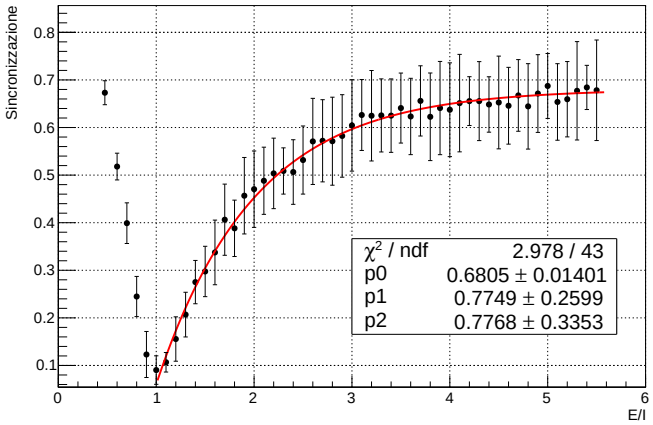


Figura 15: Fit esponenziale dei dati di sincronizzazione media in funzione di E/I

posto $E/I = 2$ e sono stati ricavati i dati della sincronizzazione in funzione della probabilità di accensione nello stato iniziale. I dati sono riportati nel grafico in Figura 16, insieme alle barre di errore stimate tramite la deviazione standard come in precedenza. La misura di sincronizzazione usata è stata `get_activation_sync()`, siccome permette di apprezzare anche il caso in cui molti neuroni rimangono spenti.

Si noti come la dinamica futura della rete dipende pesantemente da come viene inizializzata. In particolare, più neuroni vengono accesi all'inizio, meno la rete riesce a sincronizzarsi. Questo è dovuto al fatto che, se un neurone viene accesso all'inizio, questo non rimane per forza acceso per tutto il periodo `time_active`, ma per un periodo casuale tra 0 e `time_active`. Perciò, anche se vengono accesi molti neuroni, questi non saranno coordinati, e più alla prima iterazione si trovano già accesi, meno la rete avrà la possibilità di coordinarsi durante l'evoluzione. Se invece la probabilità di attivazione è bassa, molti neuroni saranno spenti e quindi stimolati ad accendersi da i pochi neuroni attivati, e questo restituisce più "margine" alla rete di cadere in una dinamica periodica sincronizzata.

Perciò, considerando che i nostri neuroni hanno un tempo di attivazione di 2 e refrattario di 1, un'inizializzazione in cui si accendono 2/3 dei neuroni dovrebbe essere quella che provoca una dinamica periodica meno sincronizzata possibile. Infatti, in questo modo, un terzo dei neuroni rimarrà spento nello stato iniziale, mentre i restanti si accenderanno. Di quelli che si accendono, però, solo la metà rimarrà nello stato eccitato per due iterazioni, mentre l'altra metà si spegnerà dopo una sola iterazione. Questa è la dinamica periodica più lontana dalla sincronizzazione. Il grafico dei due valori di sincronizzazione in funzione del parametro E/I con una probabilità di inizializzazione del 66% è

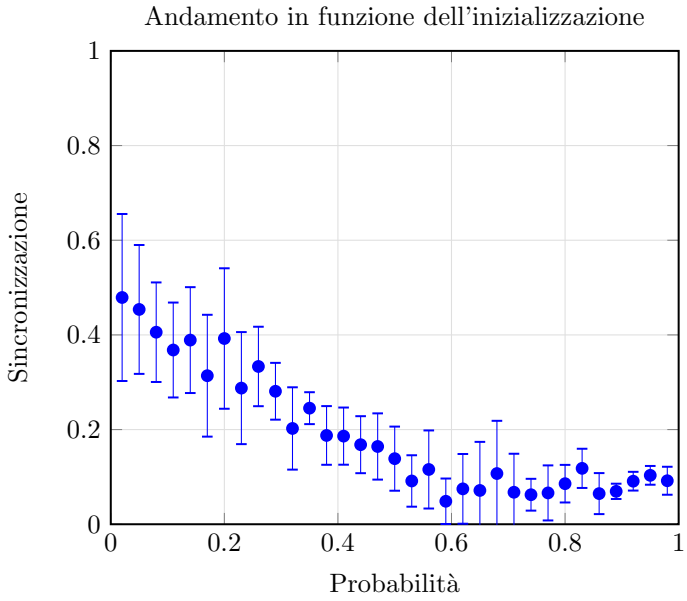


Figura 16: Dati sulla sincronizzazione dei neuroni in funzione della probabilità di attivazione iniziale

riportato in Figura 17. Si noti come, quando si ha a che fare con valori bassi di sincronizzazione, la cosa più attendibile da guardare sia in realtà il numero di neuroni che si attivano contemporaneamente, ossia la sincronizzazione calcolata dalla funzione `get_activation_sync()`. S_{media}, invece, quando $E/I > 3$, restituisce un valore di sincronizzazione di circa $1/3$. Questo perché, durante ogni step, $2/3$ dei neuroni sono accesi, mentre il restante terzo è spento, restituendo una sincronizzazione di $2/3 - 1/3 = 1/3$. Quando E/I si avvicina al valore 1, invece, i pesi delle connessioni sono equamente distribuiti tra negativi e positivi, e questo genera una dinamica desincronizzata, diversa da quella descritta sopra. Questo fenomeno è chiaramente apprezzabile in Figura 18.

Un metodo utilizzato per studiare la tendenza alla sincronizzazione della rete è stato quello di mandare un segnale in input durante la sua evoluzione. Per segnale si intende l'accensione di un certo numero di neuroni. L'intensità del segnale sarà quindi il numero di neuroni accesi. Un primo esempio si può osservare in figura 19, dove si è compiuto uno studio della sincronizzazione raggiunta dalla rete in funzione del rapporto E/I mandando un impulso al 100%. Chiaramente la situazione in tal caso è banale, in quanto tutti i neuroni sono attivati dal segnale ad un certo istante. Tuttavia si può osservare che la sincronizzazione non è una configurazione stabile per tutti gli E/I, infatti nell'evoluzione successiva al segnale la sincronizzazione è inferiore al 90% per tutti gli $E/I < 2$.

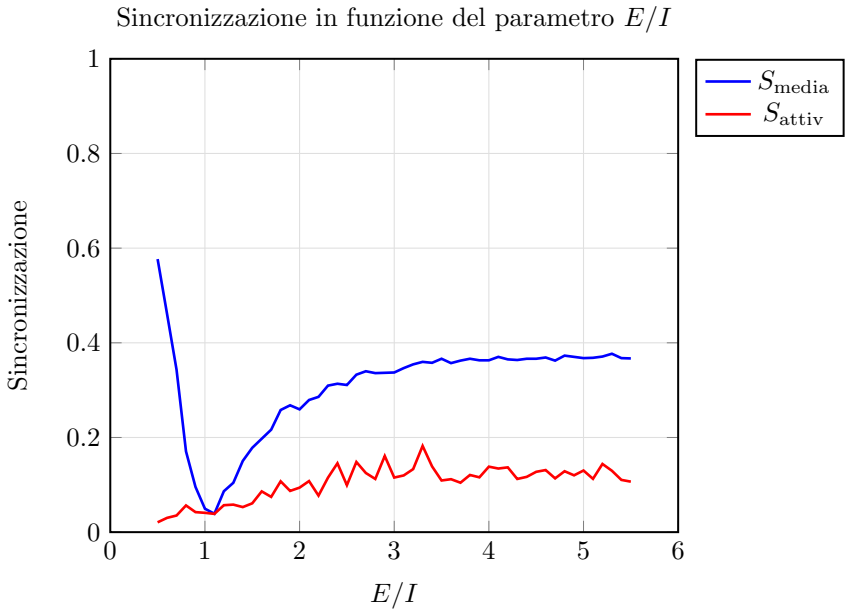


Figura 17: Grafico della sincronizzazione raggiunta dalla rete in funzione del parametro E/I, con una configurazione iniziale al 66% di attivazione

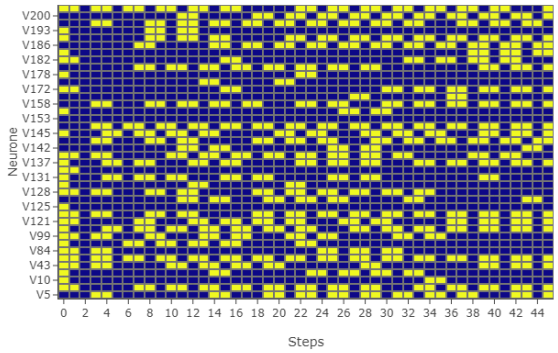


Figura 18: Dinamica ottenuta con un valore di $E/I = 1$ e inizializzazione al 66% di attivazione

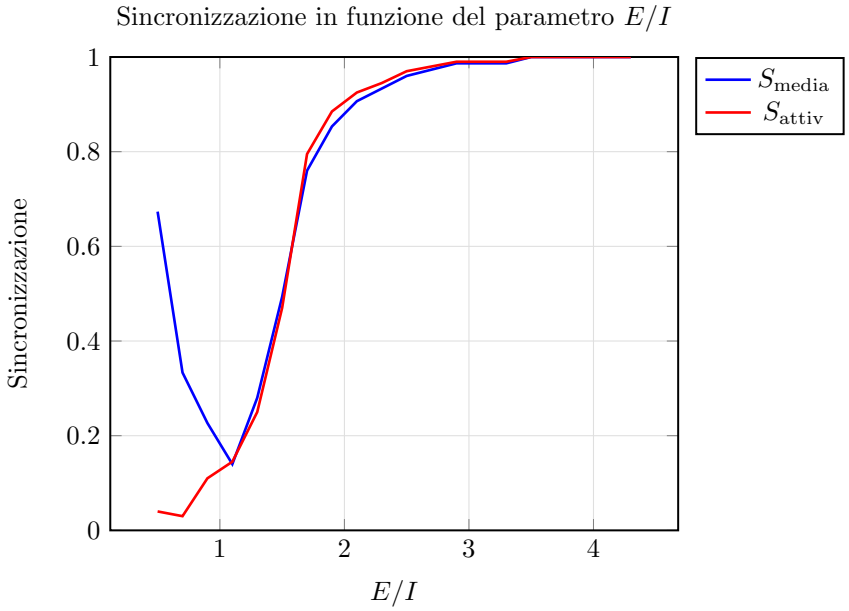


Figura 19: Grafico della sincronizzazione raggiunta dalla rete in funzione del parametro E/I in seguito a un segnale che ha accende tutti i neuroni della rete

Inoltre, in Figura 20 è raffigurato l'andamento della sincronizzazione calcolata tramite `get_activation_sync()`, per due diversi valori di E/I, in funzione della frazione del numero di neuroni che prendono parte al segnale. In particolare, il segnale è stato mandato per 10 volte, aspettando 21 iterazioni tra un segnale e il successivo.

6.2 Rete a Grafo Clusterizzato

Per studiare la dinamica nel caso di un grafo clusterizzato, per prima cosa si è generato un grafo con 3 cluster, 50 nodi ciascuno. L'in-degree è stato mantenuto uguale a 6 mentre le connessioni tra i cluster sono state poste a 200, il che vuol dire che in generale un neurone in un cluster è connesso a 4 neuroni di un diverso cluster. Poi è stata chiamata la funzione `cycle()`, la quale ricordiamo essere una funzione che rende le connessioni interne a ogni cluster negative mentre quelle tra i cluster fortemente positive e in modo da formare una struttura "ad anello". La dinamica ottenuta è riportata in Figura 21, si nota chiaramente come il segnale venga trasmetto tra il primo, il secondo e il terzo cluster per poi iniziare nuovamente il ciclo.

Sempre con un grafo clusterizzato come quello descritto in precedenza, ma senza chiamare la funzione `cycle()`, sono state eseguite ulteriori simulazioni per osservare la dinamica raggiunta dalla rete. In questo caso, quindi, il grafo ha le connessioni dentro a un cluster positive e quelle tra i cluster fortemente nega-

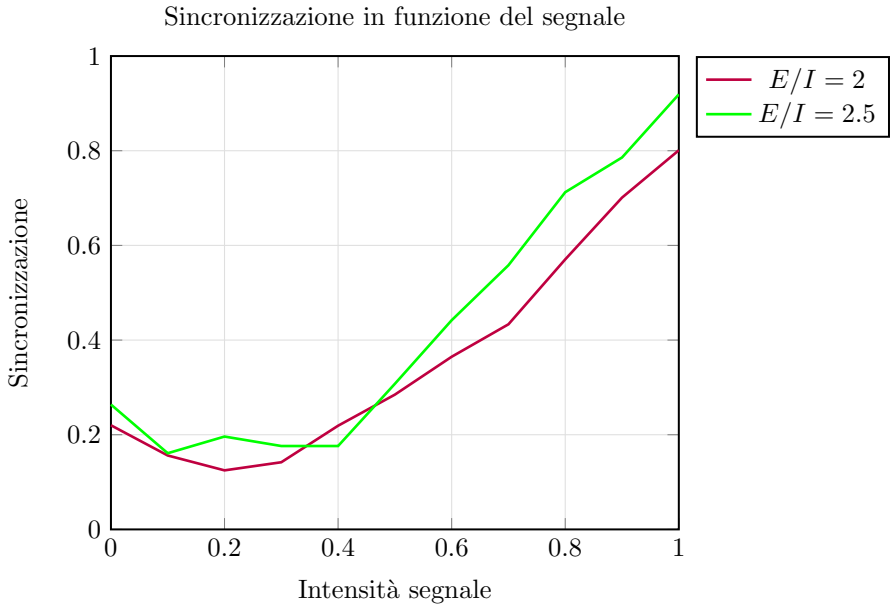


Figura 20: Grafico della sincronizzazione raggiunta dalla rete in funzione dell'intensità del segnale, cioè della frazione del numero di neuroni soggetti all'impulso, per due diversi valori del parametro E/I

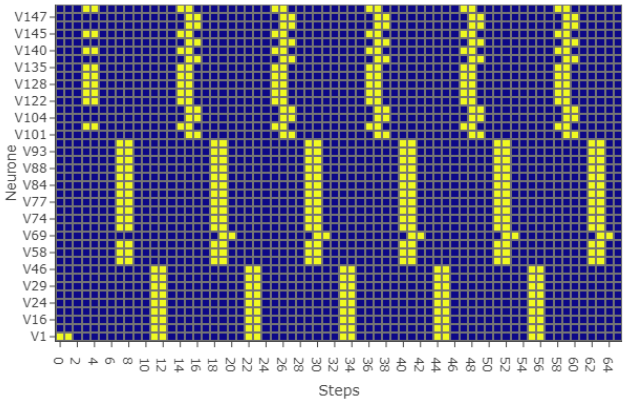


Figura 21: Dinamica nel caso di un grafo clusterizzato "ad anello"

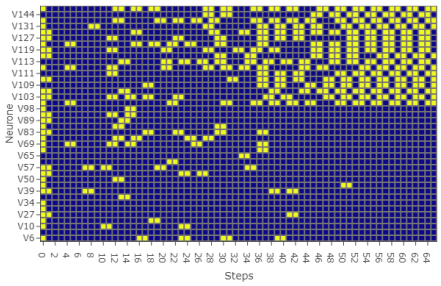
tive. Perciò, ci aspettiamo che in seguito ad un'inizializzazione casuale, la rete tenda a "spegnere" i cluster ad eccezione di uno che rimane "acceso". Inoltre, quale cluster rimanga effettivamente acceso è ovviamente casuale e dipende for-

temente dalle condizioni iniziali. In Figura 22 sono riportate alcune dinamiche ottenute in questo modo, con diversi valori iniziali della probabilità di attivazione. In Figura 22c, inoltre, è riportato il caso particolare in cui nessuno tra due cluster riesce effettivamente a prevalere sull'altro, e quindi a spegnerlo. In questo caso però la dinamica non sembra giunta a una situazione periodica e stabile nonostante il fatto che la dinamica, per come è implementato il programma, sia deterministica a fissate le condizione iniziali. In più, essendo lo spazio delle fasi della rete finito, nella sua evoluzione la rete è obbligata a tornare in un punto della traiettoria già attraversato e quindi a cadere in una situazione periodica. Per questo non può esistere una dinamica veramente casuale del nostro sistema e si considera la dinamica riportata in Figura 22c caotica.

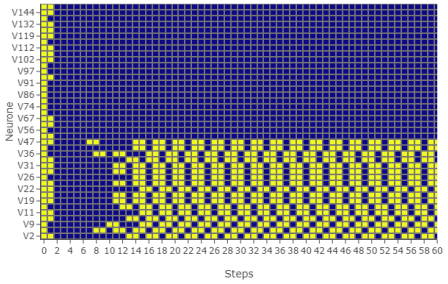
6.3 Rete a Grafo Bipartito

Nel caso di un grafo bipartito, ci si aspetta che il segnale "rimbalzi" tra i due sottoinsiemi di nodi. Anche in questo caso, però, la dinamica che segue il sistema dipenderà fortemente da come si inizializza la rete. Se infatti si inizializzano entrambi i sottoinsiemi di nodi ugualmente (quindi con lo stesso numero di neuroni attivi), la rete si comporterà come un grafo normale in sincronizzazione. Il segnale del primo sottoinsieme $\mathcal{V}_1$, infatti, attiverebbe il secondo sottoinsieme $\mathcal{V}_2$, e viceversa. Però questo avverrebbe contemporaneamente, generando una rete completamente sincronizzata, nonostante in realtà i neuroni in $\mathcal{V}_1$ non comunichino tra di loro, così come quelli in $\mathcal{V}_2$. Un esempio è mostrato in Figura 23.

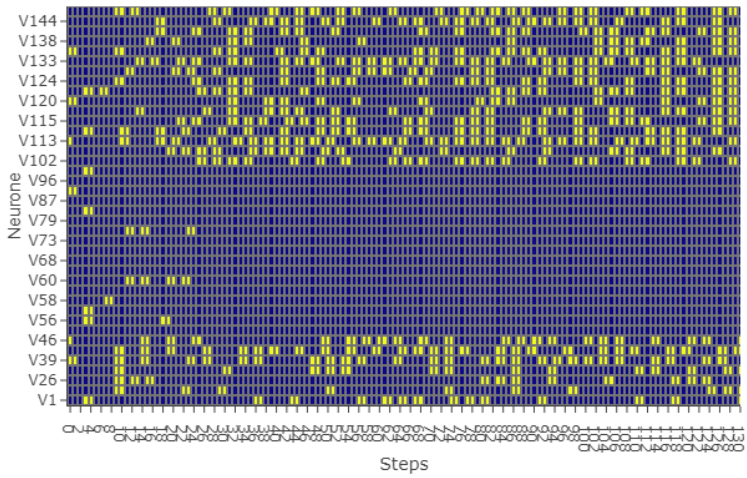
Se invece $\mathcal{V}_1$ e $\mathcal{V}_2$ vengono inizializzati in modo asimmetrico, allora il sistema tenderà verso una dinamica in cui quando un sottoinsieme è acceso, l'altro è spento. Chiaramente, più i sottoinsieme vengono inizializzati in modo asimmetrico, più questa dinamica sarà visibile e apprezzabile. Un esempio è riportato in Figura 24.



(a) Probabilità di inizializzazione = 70%



(b) Probabilità di inizializzazione = 100%



(c) Probabilità di inizializzazione = 20%

Figura 22: Simulazioni con grafo clusterizzato

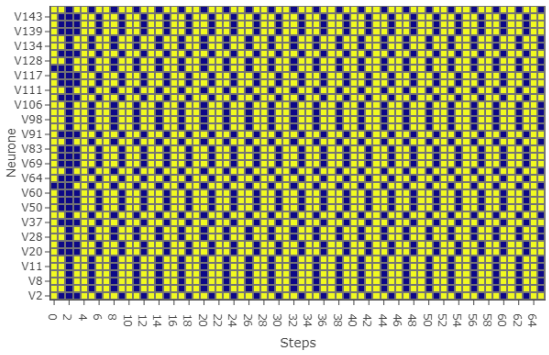


Figura 23: Dinamica su un grafo bipartito con l'80% di tutti i neuroni inizializzati

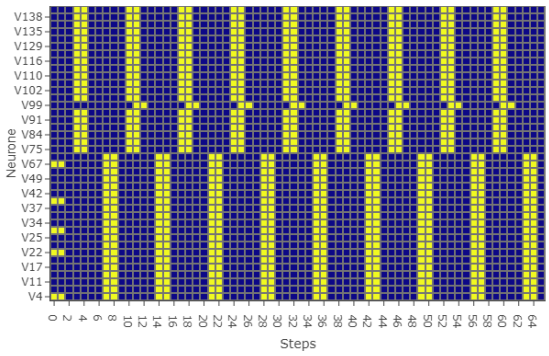


Figura 24: Dinamica su grafo bipartito con solo un sottoinsieme di nodi inizializzato

Riferimenti bibliografici

[1] Richard FitzHugh. ≪Impulses and Physiological States in Theoretical Models of Nerve Membrane≫. In: *Biophysical Journal* 1.6 (1961), pp. 445–466. ISSN: 0006-3495. DOI: https://doi.org/10.1016/S0006-3495(61)86902-6. URL: https://www.sciencedirect.com/science/article/pii/S0006349561869026.

[2] J. NAGUMOt et al. ≪An Active Pulse Transmission Line Simulating Nerve Axon≫. In: 2006.

[3] Wulfram Gerstner e Werner M. Kistler. ≪Spiking Neuron Models: Single Neurons, Populations, Plasticity≫. In: 2002.

[4] Scanziani M. Isaacson JS. ≪How inhibition shapes cortical activity≫. In: (2011). DOI: 10.1016/j.neuron.2011.09.027.