CAPITOLO 3

IMPLEMENTAZIONE DEL MODELLO E RISULTATI SPERIMENTALI

3.1 Implementazione del modello

Il modello è stato implementato su calcolatore mediante la realizzazione di uno script matlab. In questo paragrafo riporteremo e commenteremo parti del codice del programma.

La posizione delle particelle nello spazio è individuata delle tre coordinate x , y e z. Le matrice `X` di dimensione N*3, con N numero di nodi del modello è utilizzata per memorizzare la posizione delle particelle per ciascun passo della simulazione. Le tre colonne della riga *i-esima* memorizzano rispettivamente le coordinate x , y, z della particella i. La direzione delle particelle nelle spazio è individuata dai due angoli θ e φ (figura 3.1). I vettori `TETA` e `FI` entrambi di dimensione N*1 memorizzano i dati relativi alle due direzioni per ciascun istante di tempo. Come si è detto il modello parte da condizioni iniziali casuali per cui la matrice `X` e i due vettori `TETA` e `FI` sono inizializzati in maniera random mediante le seguenti istruzioni:

```
X=L*(rand(N,3)-0.5)
TETA= 2*pi*(rand(N,1)-0.5)
FI= 2*pi*(rand(N,1)-0.5)
```

Dove `L` è la dimensione del piano tridimensionale in cui si muovono le particelle. Come si vede viene sfruttata la funzione di libreria "`rand(N,3)`" che genera una matrice di dimensione N*3 con numeri scelti in maniera casuale con probabilità uniforme nell'intervallo [0,1]. Di conseguenza è assegnata a ciascuna particella una posizione iniziale scelta con probabilità uniforme tra `-L/2` e `+L/2`, viceversa i valori degli angoli sono estratti con probabilità univforme tra $-\pi$ e $+\pi$ (nella istruzione indicato con `pi`). Le particelle vengono inizializzate all'inizio di ciascuna prova della simulazione.

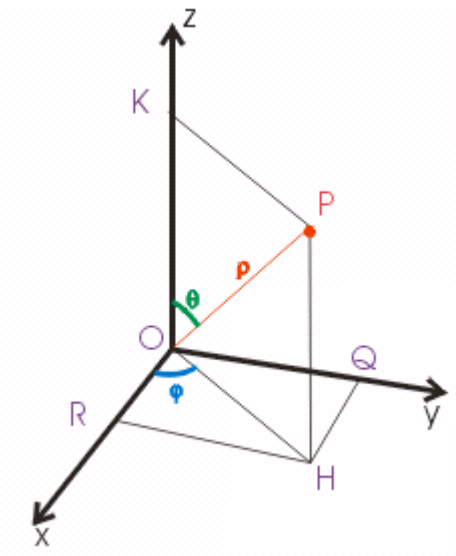


Figura 3.1 : la direzione di una particella generica è individuata dall'angolo θ (angolo in verde in figura) e dall'angolo φ (in blu in figura)

Ad ogni passo della simulazione si procede all'aggiornamento delle coordinate delle particelle secondo la regola imposta del modello. Ossia le particelle si muovono con velocità costante in modulo (v) , e assumono la direzione media delle particelle che si trovino a una distanza inferiore al raggio di interazione τ con l'aggiunta di qualche perturbazione di tipo casuale ($\Delta\theta$). Si assume per lo spazio tridimensionale un'ipotesi di mondo chiuso, quindi prima dell'aggiornamento delle coordinate quest'ultime sono normalizzate, utilizzando l'operazione modulo, rispetto alla dimensione L del piano tridimensionale:

```
X=mod(X,L);
X=X+[v*sin(FI).*cos(TETA)  v*sin(FI).*sin(TETA)  v*cos(FI)];
```

La prima istruzione realizza l'ipotesi di mondo chiuso, sfruttando la funzione di libreria "`mod(X,L)`" che sostituisce a ciascun elemento di `X` il resto della divisione intera tra l'elemento stesso e L. L'aggiornamento delle coordinate avviene scomponendo la velocità della particella $v_{i,}$ lungo le tre direzioni. La velocità v_i all'istante t è individuata dal modulo costante v e dai due angoli θ_i e φ_i (che individuano la direzione della

particella all'istante *t*). Di conseguenza l'aggiornamento delle tre coordinate della particella *i-esima* è dato dalle seguenti espressioni:

$$x_i(t+1) = x_i(t) + v * \sin(\phi_i) * \cos(\theta_i) * \Delta t$$
$$y_i(t+1) = y_i(t) + v * \sin(\phi_i) * \sin(\theta_i) * \Delta t$$
$$z_i(t+1) = z_i(t) + v * \cos(\phi_i) * \Delta t$$

Nel corso delle simulazioni all'intervallo Δt, ossia il tempo che intercorre tra due passi successivi della simulazione, è stato assegnato valore uno, quindi non compare nel codice, per ν viceversa si è scelta un valore pari a 0.03. Per $\nu \to \infty$ le particelle tendono a mischiarsi completamente dopo un passo della simulazione, viceversa un valore troppo basso renderebbe il modello statico. Un valore di ν pari a 0.03 rende le particelle abbastanza reattive tali da mutare la loro configurazione dopo pochi aggiornamenti della direzione.

Ad ogni passo della simulazione ciascuna particella aggiorna la propria direzione ponendola pari alla direzione media delle particelle che si trovano a distanza inferiore del raggio di interazione τ. Il prossimo passo, quindi, consiste nel valutare, per ciascuno dei membri del sistema, le particelle che appartengono al suo vicinato. Allo scopo s'introduce una matrice (`Vicinato`) di dimensione *N*N* che viene inizializzata come matrice nulla. L'elemento generico a_{ij} è posto poi a uno se le particelle *i* e *j* sono a una distanza inferiore del raggio di interazione, viceversa è zero se i due membri del sistema sono lontani.

Il controllo sulla vicinanza è effettuato tenendo conto anche dell'ipotesi di mondo chiuso con cui è stato implementato il sistema. Analizziamo meglio questo aspetto. La figura 3.2 rappresenta un piano *x-y* estratto dallo spazio tridimensionale, di conseguenza il piano ha dimensione pari a *L*. Focalizziamoci solo sulla coordinata *x* per semplicità. In figura sono mostrate tre diverse situazioni che possono verificarsi nel calcolo della distanza. Supponiamo di stare analizzando la particella 1 e di voler valutare la distanza dalla particella *2*. Siano x_i la coordinata *x* della particella *i*. La distanza diretta(indicata con *d* in figura) è semplicemente data dalla differenza tra le due coordinate (x_1 - x_2). Tuttavia, poiché vale l'ipotesi di spazio chiuso, la particella *2* è come se si trovasse anche nella posizione *2'* (in grigio in figura). Quindi, in realtà, la distanza *d* è maggiore della distanza effettiva tra le 2 particelle che è quella in grigio,

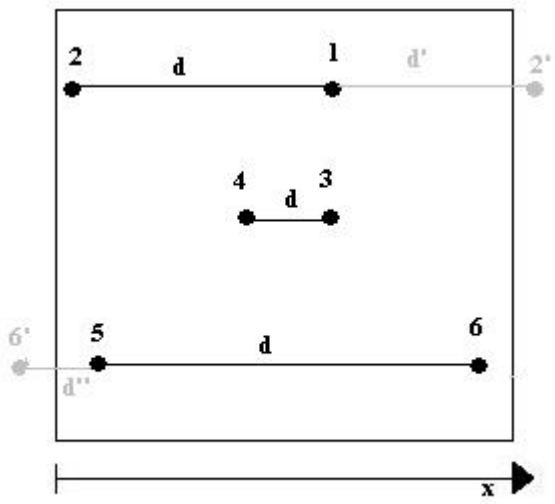


Figura 3.2: valutazione delle distanze nell'aggiornamento della matrice `Vicinato`

indicata in figura con *d',* che è data dall'espressione : $|x_1-(x_2+L)|$. Nel caso la particella in analisi sia la *3* e ne volessimo valutare la distanza dalla *4* in questo caso la distanza effettiva è quella diretta, data dalla differenze delle due coordinate: $|x_3-x_4|$. Ora soffermiamoci sulla particella *5* e valutiamo la distanza dalla particella *6*. In questo caso la distanza effettiva è quella indicata in figura con *d"* (in grigio) che è data dall'espressione: $|x_5-(x_6-L)|$. Occorre valutare di volta in volta quale delle tre espressioni produce il risultato minimo. Naturalmente il discorso va esteso per le altre due coordinate *y* e *z*. Il codice che effettua l'aggiornamento della matrice `Vicinato` è il seguente:

```
for i=1:N
   for j=1:N
      diffxmin=min([(X(j,1)-X(i,1))^2  X(j,1)-(X(i,1)-L))^2
(X(j,1)-(L+X(i,1)))^2]);
     diffymin=min([(X(j,2)-X(i,2))^2  (X(j,2)-(X(i,2)-L))^2
(X(j,2)-(L+X(i,2)))^2]);
     diffzmin=min([(X(j,3)-X(i,3))^2  (X(j,3)-(X(i,3)-L))^2
(X(j,3)-(L+X(i,3)))^2]);
   Vicinato(i,j)=(diffxmin+diffymin+diffzmin <=r^2);
   end
 end
```

le istruzioni `diffxmin,diffymin e diffzmin` valutano il minimo delle tre espressioni definite in precedenza relativamente alle tre dimensioni *x* e *y* e *z*, Infine l'ultima istruzione dei due cicli `for` effettua il confronto tra la distanza effettiva(calcolata utilizzando le distanze minime relative alle tre coordinate) e il raggio di interazione (`r` nel codice) .

Effettuata la valutazione del vicinato di ciascuna particella, si procede alla creazione delle connessioni di lungo raggio. Come si è detto ad ogni passo della simulazione sono create δ collegamenti di lungo raggio. Il codice che implementa questa operazione è il seguente:

```
for i=1:delta
  agg=round((N-1)*rand(1)+1);
  agg2=round((N-1)*rand(1)+1);
    while (Vicinato(agg,agg2)==1)
         agg=round((N-1)*rand(1)+1);
        agg2=round((N-1)*rand(1)+1);
    end
  Vicinato(agg,agg2)=1;
  Vicinato(agg2,agg)=1;
end
```

Il ciclo `for` esterno viene eseguito `delta` (δ) volte e ha lo scopo di creare una connessione di lungo raggio. Vengono infatti estratti due particelle in maniera casuale(`agg` e `agg2`) ma che siano lontane, e, quindi, per le quali quindi `Vicinato(agg,agg2)` sia zero. Il ciclo `while`, infatti, continua a estrarre le particelle fino a che non ne trova due lontane. Infine le particelle estratte sono rese connesse mediante le istruzioni `Vicinato(agg,agg2)=1` e `Vicinato(agg2,agg)=1`. La procedura è poi ripetuta per ciascuna delle delta connessioni di lungo raggio.

Il prossimo passo è quantificare il numero di vicini relativi a ciascuna particella. Questo è ottenuto tramite l'istruzione :

```
D=diag(sum(Vicinato,2));
```

Si utilizza la funzione di libreria "`sum()`" che restituisce un vettore con la somma degli elementi lungo la dimensione inviata come secondo parametro della funzione. In questo caso la dimensione è 2 , quindi la somma è per righe. Tuttavia in questo caso la scelta della dimensione di somma è del tutto indifferente, essendo la matrice `Vicinato` simmetrica. La funzione "`diag()`" posiziona gli elementi del vettore, che viene passato alla funzione come parametro, sulla diagonale di una matrice *N*N*. Tale istruzione quindi restituisce una matrice(`D`), i cui elementi sono tutti nulli tranne quelli della diagonale e in cui l'elemento *i-esimo* della diagonale contiene il numero di particelle vicine alla particella i-esima(il numero comprende la particella stessa).

A questo punto di può procedere all'aggiornamento della direzione di ciascuna particella. Per ciascuna particella l'aggiornamento dei due angoli avviene secondo le seguenti espressioni:

$$\theta_i(t+1) = \arctan\left(\left\langle \frac{\sin(\theta(t))}{\cos(\theta(t))} \right\rangle_\tau\right) + \Delta\theta$$

$$\phi_i(t+1) = \arctan\left(\left\langle \frac{\sin(\phi(t))}{\cos(\phi(t))} \right\rangle_\tau\right) + \Delta\phi$$

Quindi si valuta la media del seno e coseno degli angoli θ e φ delle particelle appartenenti al vicinato e dunque viene estratta l'arcotangente del rapporto delle medie relative a seno e coseno. Il codice che implementa queste operazione è il seguente:

```
dTETA=eta*rand(N,1)-eta/2;
TETAcos=inv(D)*Vicinato*cos(TETA);
TETAsin=inv(D)*Vicinato*sin(TETA);
TETA=atan2(TETAsin, TETAcos)+dTETA;
```

La prima istruzione estrae il disturbo casuale $\Delta\theta$ (`dTETA`) con probabilità uniforme nell'intervallo [$-\eta/2$, $+\eta/2$] , la seconda e terza istruzione calcolano la media relativa rispettivamente al coseno e al seno dell'angolo θ delle particelle del vicinato. Infine la quarta istruzione calcola il valore aggiornato di θ, calcolando l'arcotangente del

rapporto delle due medie e sommando a questo valore il disturbo derivante dal rumore. Il codice relativo all'angolo φ è del tutto analogo a quello riportato per l'angolo θ.

A questo punto possono essere valutati per ciascun passo della simulazione i parametri d'analisi. Consideriamo il parametro d'ordine v_a definito dall'espressione:

$$v_a = \frac{1}{Nv} \left| \sum_{i=1}^{N} v_i \right|$$

Dove v_i è la velocità della particella i-esima. Le tre coordinate lungo gli assi *x,y* e *z* della generica velocità v_i della particella i-esima sono:

$$\begin{aligned} Vx_i &= v * \sin(\phi_i) * \cos(\theta_i) \\ Vy_i &= v * \sin(\phi_i) * \sin(\theta_i) \\ Vz_i &= v * \cos(\phi_i) \end{aligned}$$

Sostituendo nell'espressione di v_a si ottiene:

$$v_a = \frac{1}{Nv} \left| \sum_{i=1}^{N} v * \sin(\phi_i) * \cos(\theta_i) * ax + v * \sin(\phi_i) * \sin(\theta_i) * ay + v * \cos(\phi_i) * az \right|$$

Dove *ax*, *ay* e *az* sono rispettivamente i versori dell'asse *x*, *y* e *z*. Il fattore v è comune a tutti i membri della sommatoria, è possibile, quindi, metterlo in evidenza e portarlo fuori dalla sommatoria. Portato fuori si semplifica con l'identico fattore v presente al denominatore:

$$v_a = \frac{1}{N} \left| \sum_{i=1}^{N} \sin(\phi_i) * \cos(\theta_i) * ax + \sin(\phi_i) * \sin(\theta_i) * ay + \cos(\phi_i) * az \right|$$

Sostituendo al modulo la corrispondente espressione si ottiene:

$$v_a = \frac{1}{N} \sqrt{\left(\sum_{i=1}^{N} \sin(\phi_i) * \cos(\theta_i) \right)^2 + \left(\sum_{i=1}^{N} \sin(\phi_i) * \sin(\theta_i) \right)^2 + \left(\sum_{i=1}^{N} \cos(\phi_i) \right)^2}$$

Il codice corrispondente all'espressione è il seguente:

```
va=[va 1/N*sqrt((sum(cos(FI)))^2 +
(sum(sin(TETA).*sin(FI)))^2 + (sum(cos(TETA).*sin(FI)))^2)]
```

L'istruzione aggiunge in coda al vettore va il valore del parametro calcolato nel passo attuale della simulazione.

Lo scopo della nostra analisi è lo studio del modello di Vicseck, applicando il paradigma delle reti dinamiche. Fino a adesso abbiano analizzato l'implementazione del modello. Lo scopo della nostra ricerca sarà anche la valutazione della variazione dei parametri caratteristici delle reti in relazione all'evoluzione del sistema e quindi in corrispondenza delle variazione del parametro v_a, che quantifica lo stato di ordinamento del sistema. Rappresenteremo la rete in termine della sua matrice delle adiacenze. La matrice `Vicinato` tiene traccia delle connessioni esistenti tra i vari nodi ed è modificata dinamicamente ad ogni passo della simulazione. Tale matrice può essere considerata,a tutti gli effetti, la matrice delle adiacenza della rete. Dunque valuteremo i parametri della rete facendo riferimento alla matrice `Vicinato`.

Il primo su cui ci soffermeremo è il grado di ciascun nodo della rete. Esso è stato definito come il numero di connessioni che insistono su un nodo. La sua espressione in termini della matrice delle adiacenze è:

$$k_i = \sum_{j \in N} a_{ij}$$

Questa misura è già stata valutata nel calcolo della direzione media dei nodi vicini. E' stato, infatti, calcolato il numero di vicini o, in maniera equivalente, il numero di nodi connessi relativo a ciascun nodo, che non è altro che il grado di ciascun nodo, ed è stato memorizzato sulla diagonale della matrice `D`. Il seguente ciclo `for` crea un vettore di dimensione *N*(vettore `K`) in cui l'elemento i-esimo rappresenta il grado del nodo i estraendo il valore dalla matrice `D`:

```
for i=1:N
   K(i)=D(i,i)
end
AvgK=[AvgK (sum(K)/N)]
```

L'ultima istruzione che segue il ciclo `for` calcola il grado medio facendo la sommatoria dei gradi di ciascun nodo, utilizzando la funzione di libreria "`sum()`", e facendone il rapporto con il numero dei nodi *N*. Il valore è poi posto in coda al vettore `AvgK`.

La valutazione degli altri due parametri su cui ci soffermiamo (avarage shorth path e coefficiente di clustering) è effettuata tramite l'ausilio di un applicativo esterno: il Pajek [11]. Il pajek è un programma per lo studio delle reti complesse e dispone di numerose funzionalità. In particolare noi lo utilizzeremo per il calcolo della matrice dei percorsi più brevi, in cui ciascun elemento a_{ij} è la lunghezza del percorso più breve tra il nodo *i* e *j*, e per il calcolo del coefficiente di clustering c_i di ciascun nodo. Lo script matlab "pajek" richiamato all'interno del programma principale ha lo scopo di scrivere la rappresentazione della rete in un formato intellegibile al programma, quindi l'applicativo è lanciato in esecuzione in background, dunque lo script "elaborazione" si occupa di estrarre i dati di nostro interesse dal file di output prodotto dal pajek.

Il pajek utilizza dei file testuali per rappresentare una rete. Il formato generico di tali file è:

```
*vertices n
1 "nome nodo"
..............
n "nome nodo n"
*edges
  i j
.................
```

dove `n` è il numero dei nodi della rete. Il file è costituito da un elenco numerato dei nodi con le rispettive etichette, e da un elenco di coppie di nodi che rappresentano le coppie di nodi connessi da un arco (elencate dopo la stringa `*edges`). Il compito dello script "pajek" è quello di scrivere un file con questa struttura che rappresenti la topologia della rete in ciascun passo della simulazione servendosi della matrice delle adiacenze. La parte più significativa del codice che crea il file è la seguente:

```
[l,m]=find(Vicinato);
Z=[l,m];
d=size(Z);
```

```
for I= 1:d(1)
fprintf(z,'%d%d%c%c',Z(I,1),Z(I,2),char(13),char(10));
end
```

Questa parte di codice serve ad individuare e scrivere sul file le coppie di nodi connessi, quindi gli archi della rete. La prima istruzione utilizza la funzione "`find()`" per individuare le coppie connesse. La funzione restituisce le coppie di indici i e j tali che $a_{ij} \neq 0$, dove a_{ij} rappresenta l'elemento generico della matrice che viene passata come parametro alla funzione (nel nostro caso la matrice delle adiacenze `Vicinato`) Il ciclo `for` scrive poi le varie coppie nel file, che era stato aperto con istruzioni precedenti a quelle riportate.

Il codice dello script "elaborazione" ha lo scopo di estrarre i dati di nostro interesse dai file di output del programma pajek. Quest'ultimo produce due file : "short.net", che contiene la matrice dei percorsi più brevi e il file " clustering.vec" che contiene il coefficiente di clustering relativo a ciascun nodo della rete. La parte di codice relativa all'estrazione dei coefficienti di clustering è:

```
z=fopen('clustering.vec','r');
c=textscan(z,'%f','headerLines',1);
fclose(z);
c=c{1,1};
C=sum(c)/N;
```

La variabile `c` memorizza il vettore dei coefficienti di clustering prelevandolo da file tramite l'istruzione di libreria "`textscan()`". L'ultima istruzione riportata salva nella variabile `C` il coefficiente di clustering medio della rete. Le istruzione per l'estrazione della matrice dei percorsi più brevi sono analoghe, la matrice è estratta da file e salvata nella variabile `Short`. Quindi viene calcolata la distanza media e salvata nella variabile `AvgPath`:

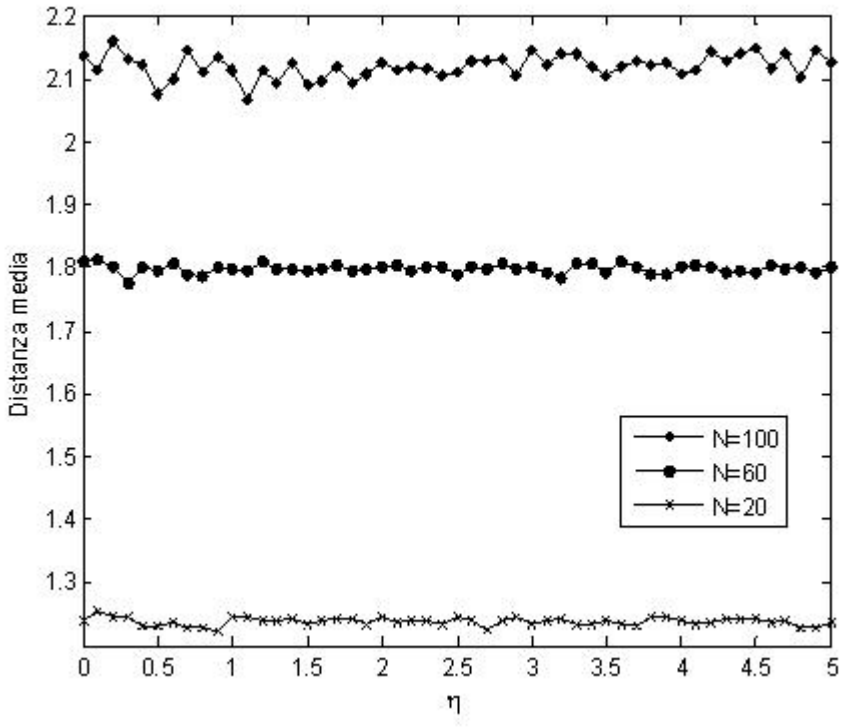
```
AvgPath=sum(Short,1);
AvgPath=sum(AvgPath)/(N*(N-1));
```

3.2 Risultati sperimentali.

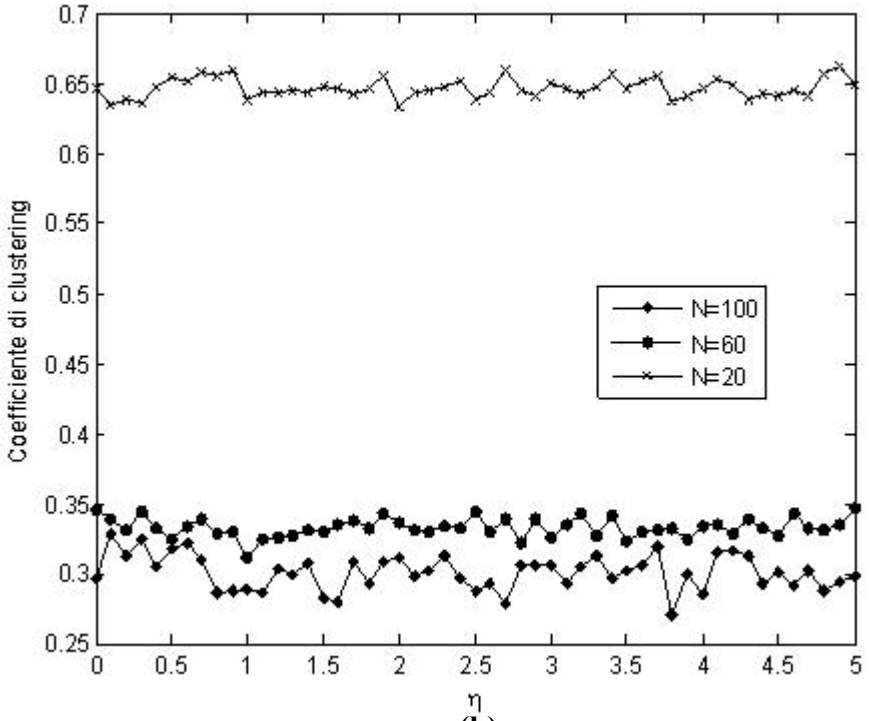
Partiamo dunque dalla valutazione della dinamica dei parametri introdotti, in corrispondenza di diversi valori del rumore. In tabella 3.1 è mostrato l'elenco completo dei dati della simulazione:

PARAMETRO	VALORE
Raggio di interazione τ	1
Densità ρ	4
Numero di nodi N	20-60-100
Modulo della velocità v	0.03
Dimensione dello spazio L	2.924
Intervallo di tempo Δt	1
Rumore η	[0,5]

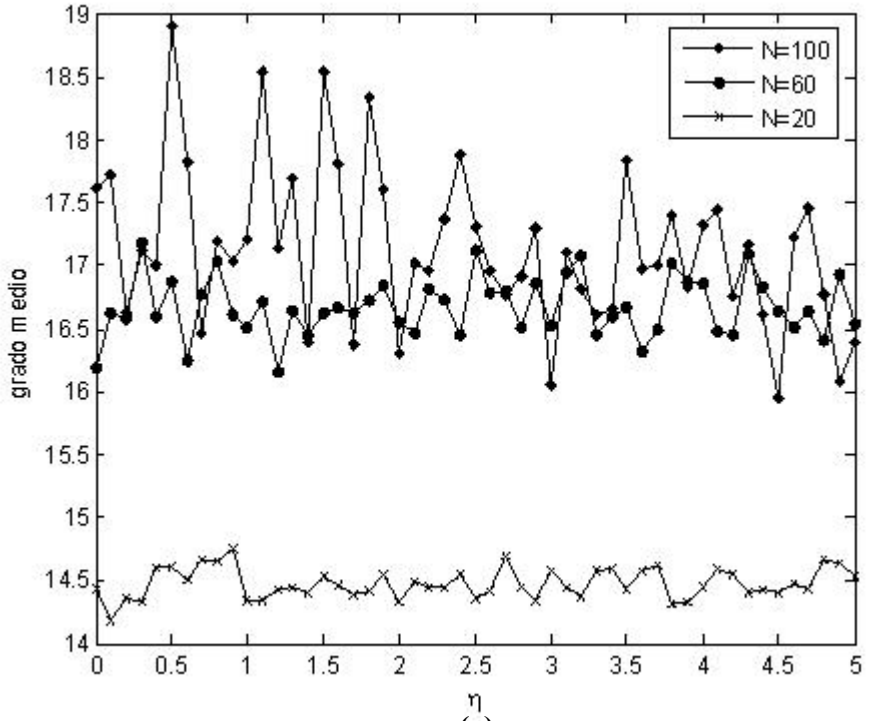
Tabella 3.1 : dati relativi alla prima simulazione



(a)



(b)



(c)

Figura 3.3 : andamento della distanza media (a) , del coefficiente di clustering medio(b) e del grado medio dei nodi(c) al variare del rumore η. I risultati sono mediati su cinque prove.

La figura 3.3 mostra i risultati della simulazione medianti su cinque prove. Le prove sono state effettuate a densità costante(ρ=4) per diversi valori di *N*. Il rumore η assume valore compreso nell'intervallo [0,5].

In figura 3.3(a) è mostrato l'andamento della distanza media al variare del rumore. Le tre curve sono pressoché costanti al variare del rumore. Questo è dovuto al fatto che il valore del parametro è stato valutato sul sistema a regime, quindi quando il valore del parametro d'ordinamento v_a ha raggiunto il suo valore stabile. In queste condizioni anche la distanza media assume un valore di regime, che non risulta particolarmente influenzato dal rumore. Le tre curve si riferiscono a valori diversi di *N*, da cui la differenza nel valore assunto. Infatti, una rete con un numero maggiore di nodi presenterà un valore della distanza media superiore, anche se il valore assunto dal parametro nei due casi estremi *N*=20 e *N*=100 non è in realtà molto differente. Questa è una caratteristica tipica delle reti di tipo di random(le condizioni iniziali del sistema sono estratte casualmente), in cui la distanza media si mantiene basso anche con un notevole aumento del numero dei nodi N.

La figura 3.3(b) mostra l'andamento del coefficiente di clustering medio dei nodi al variare dei nodi. Anche in questo caso valgono le considerazioni fatte per la distanza media. Il valore del parametro è valutato con il sistema a regime, e risulta non particolarmente influenzato dal valore del rumore. Come in tutte le reti random, la rete non risulta molto clusterizzata, infatti il valore del coefficiente di clustering è basso nei casi *N*=100, e *N*=60 mentre il caso *N*=20 presenta un valore medio che però è dovuto unicamente al piccolo numero di nodi. Una rete con un numero minori di nodi risulta ovviamente più clusterizzata.

La figura 3.3(c) mostra l'evoluzione del grado medio. La curva *N*=100 ha un andamento rumoroso, mentre le curve *N*=20 e *N*=60 presentano un andamento più regolare. Anche in questo caso il parametro è valutato con in sistema a regime e come nei due casi precedenti non si rileva una dipendenza netta dal rumore. Il valore assunto è piuttosto dipendente dal numero *N* di nodi. Una rete con un numero maggiore di nodi tenderà, ovviamente, ad essere caratterizzata da un grado medio superiore rispetto a reti con un minor numero di nodi

Valutiamo adesso l'effetto sul sistema dell'introduzione delle connessioni di lungo raggio. La tabella 3.2 mostra i dati della simulazione:

PARAMETRO	VALORE
Raggio di interazione τ	1
Densità ρ	4
Numero di nodi N	100
Modulo della velocità ν	0.03
Dimensione dello spazio L	2.924
Intervallo di tempo Δt	1
Rumore η	1
Connessioni di lungo raggio δ	[0,10]

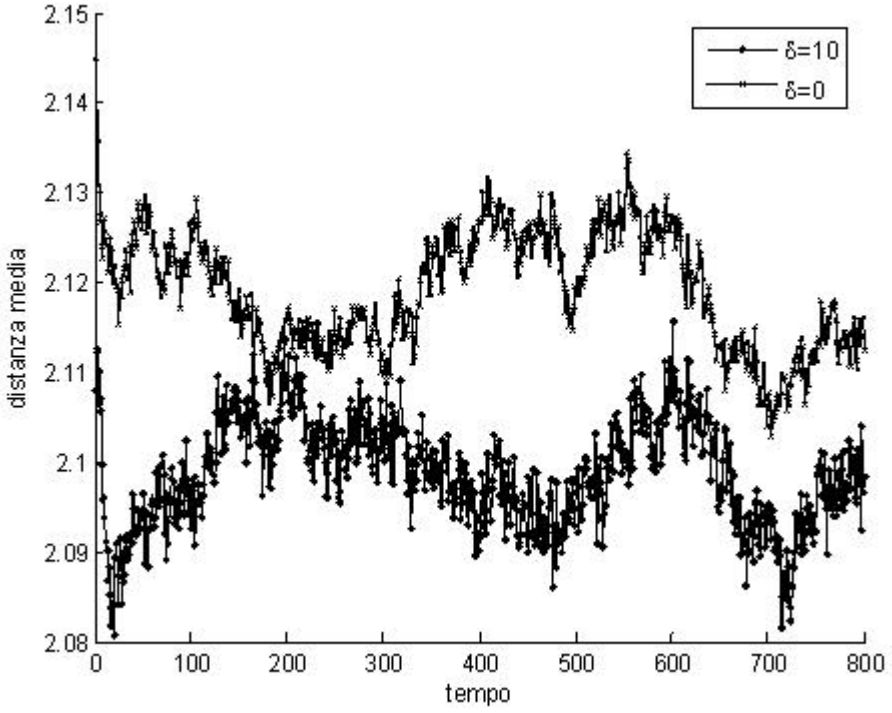
Tabella 3.2 : dati relativi alla seconda simulazione: la densità è costante nel corso delle prove , il rumore è costante è pari a uno , il numero di nodi è costante ed è pari a 100. Si utilizza un numero di connessioni di lungo raggio δ che va da zero a 10 nelle varie prove.

La figura 3.4 mostra l'andamento dei parametri analizzati in funzione del tempo. Sono riportate solo le curve relativa a zero connessioni di lungo raggio per ciascun passo della simulazione(quindi per ogni secondo poiché Δt è pari a uno), e la curva relativa a dieci connessioni.

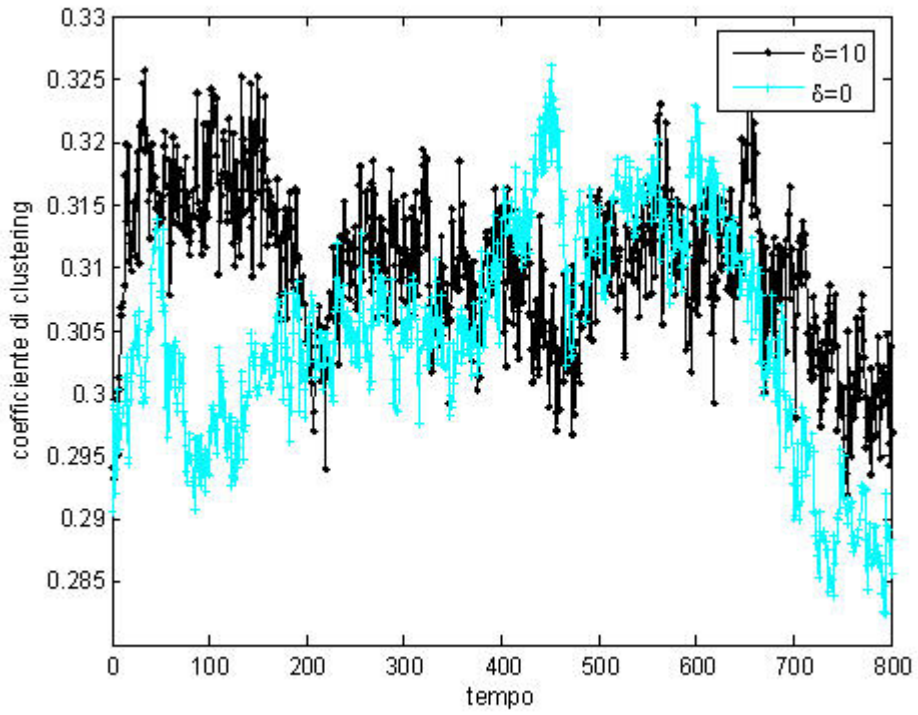
La figura 3.4(a) rappresenta l'andamento della distanza media.. L'introduzione delle connessioni di lungo raggio porta ad un decremento della distanza media. Questo è dovuto all'effetto "small word" prodotto dall'introduzione dei collegamenti a distanza.

La figura 3.4(b) mostra l'andamento del coefficiente di clustering in funzione del tempo sempre utilizzando zero e dieci connessioni di lungo raggio. L'effetto delle connessioni porta ad un sensibile aumento del coefficiente di clustering, anche se l'andamento è piuttosto rumoroso.

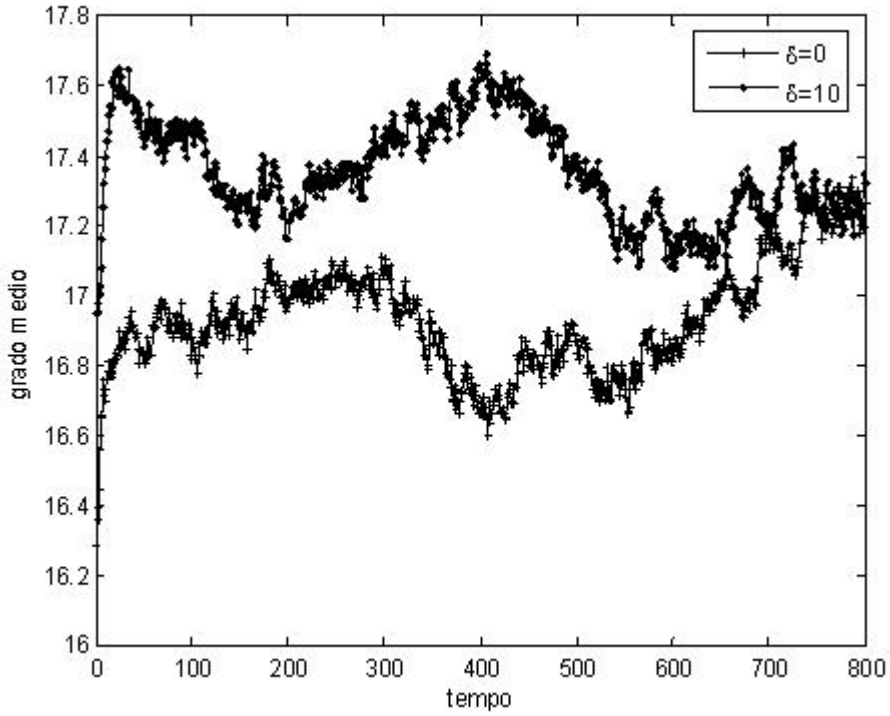
In figura 3.4(c) sono rappresentate le due curve relative al grado medio dei nodi, sempre nei due casi di analisi di nessuna connessione e dieci connessioni. Inizialmente l'introduzione delle connessioni porta ad un valore superiore del grado, dipendente dal fatto che il sistema risulta maggiormente connesso. Tuttavia nel corso dell'evoluzione del sistema gli stessi valori del parametro sono raggiunti anche dal sistema privo di connessioni di lungo raggio, ma indubbiamente, il sistema con i collegamenti a distanza ha una dinamica relativa al parametro più veloce.


δ=10
δ=0
distanza media
tempo
2.15
2.14
2.13
2.12
2.11
2.1
2.09
2.08
0
100
200
300
400
500
600
700
800

(a)


δ=10
δ=0
coefficiente di clustering
tempo
0.33
0.325
0.32
0.315
0.31
0.305
0.3
0.295
0.29
0.285
0
100
200
300
400
500
600
700
800

(b)



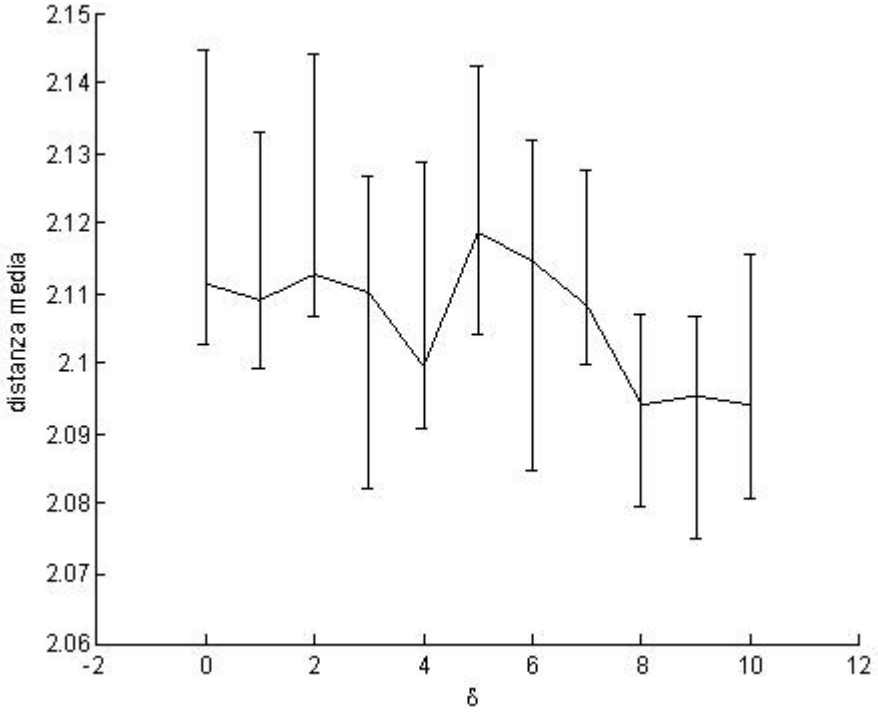
(c)

Figura 3.4: andamento della distanza media (a) , del coefficiente di clustering medio(b) e del grado medio dei nodi(c) in funzione del tempo utilizzando zero e dieci connessioni di lungo raggio. I risultati sono mediati su cinque prove.

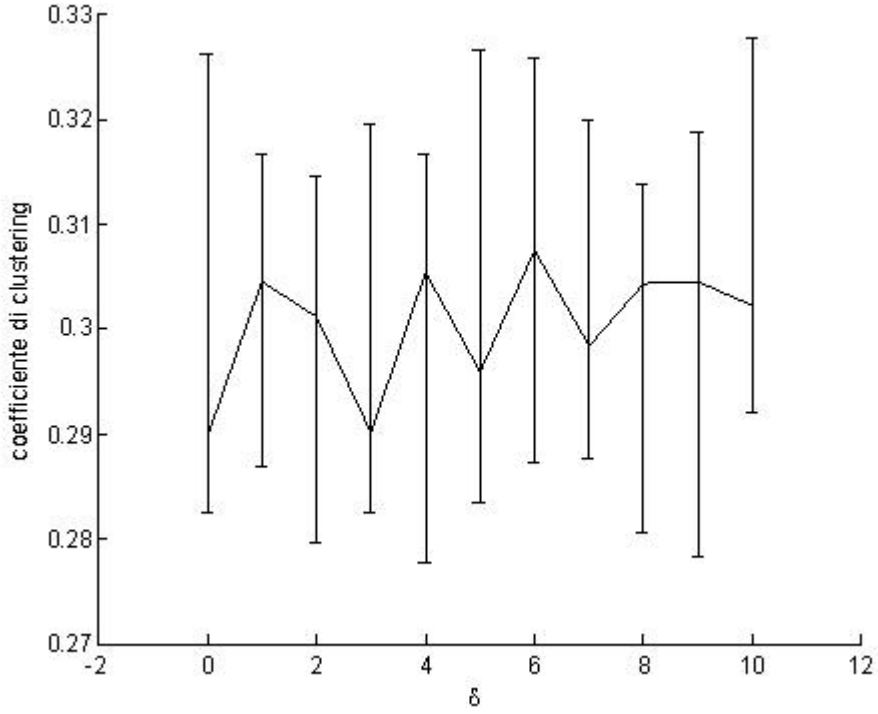
.

Valutiamo anche l'andamento dei parametri al variare del numero di connessioni di lungo raggio stabilite ad ogni passo della simulazione la figura 3.5 mostra il valore a regime dei parametri d'analisi in funzione di δ.

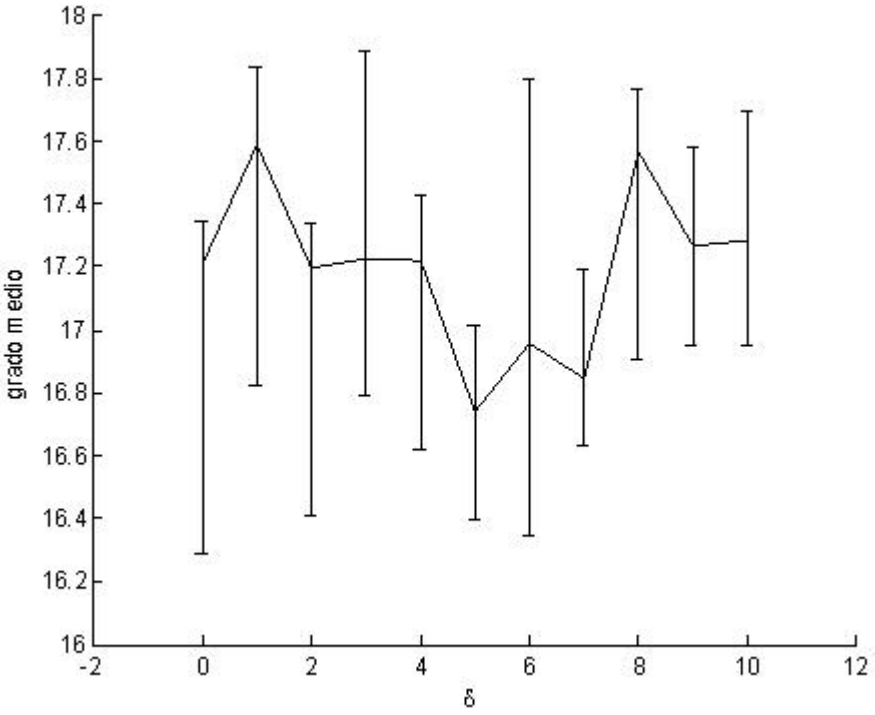
La figura 3.5(a) rappresenta la curva relativa alla distanza media in funzione del numero di connessioni stabilite. Si rileva un andamento decrescente del parametro che tende a diminuire, anche se solo sensibilmente, all'aumentare delle connessioni di lungo raggio. La figura 3.5(b) è invece relativa al coefficiente di clustering. Anche qui si nota un aumento, in realtà molto contenuto, all'aumentare del numero di connessioni. L'andamento relativo al grado dei nodi(figura 3.5(c)) non è ben definito , la curva ha , infatti un andamento frastagliato, e si mantiene su valori, pressoché molto simili.


distanza media
δ
2.15
2.14
2.13
2.12
2.11
2.1
2.09
2.08
2.07
2.06
-2
0
2
4
6
8
10
12

(a)


coefficiente di clustering
δ
0.33
0.32
0.31
0.3
0.29
0.28
0.27
-2
0
2
4
6
8
10
12

(b)



(c)

Figura 3.5: andamento della distanza media (a) , del coefficiente di clustering medio (b) e del grado medio dei nodi (c) in funzione di δ . I risultati sono mediati su cinque prove.