Inferenza statistica e machine learning in fisica statistica: il problema di Ising inverso

Il modello vettoriale di Potts a quattro colori

SAPIENZA
UNIVERSITÀ DI ROMA

Percorso di eccellenza in Fisica 2023
Docente: Luca Leuzzi

Luca Maria Del Bono (delbono.1859689@studenti.uniroma1.it)

Introduzione

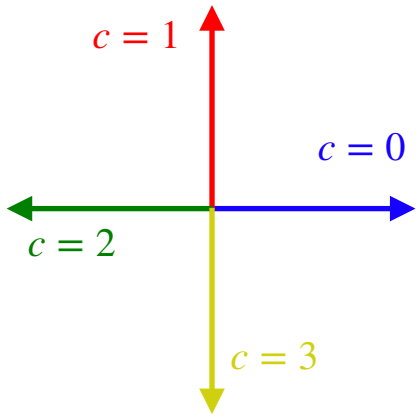
Il Modello Vettoriale di Potts a 4 colori

Modello vettoriale di Potts: le variabili

Spin di norma unitaria $\vec{s}$ orientati ad angoli $\theta_c = \frac{2\pi}{q}c, \;\; c = 0, 1, \ldots, q-1$

Nel nostro caso $q = 4$, quindi

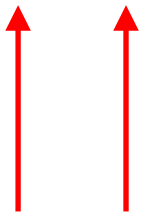
$$\vec{s} \in \left\{ \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ -1 \end{bmatrix} \right\}$$



Modello vettoriale di Potts: l'Hamiltoniana

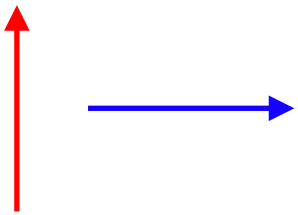
$$\mathcal{H} \equiv -J \sum_{\langle ij \rangle} \vec{s}_i \cdot \vec{s}_j = -J \sum_{\langle ij \rangle} \cos(\theta_i - \theta_j)$$

Favorita



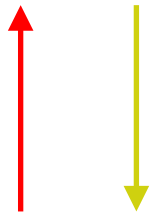
$$-J\ \vec{s}_i \cdot \vec{s}_j = -1$$

Neutra



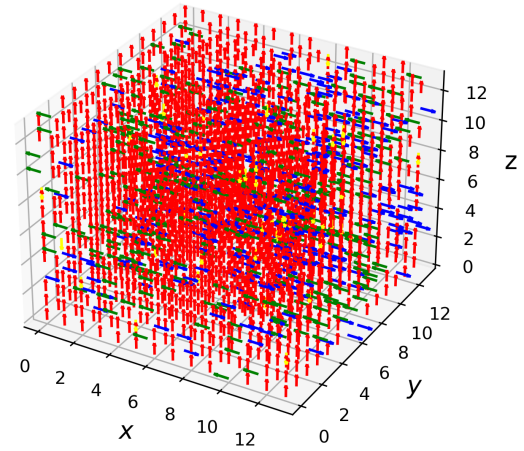
$$-J\ \vec{s}_i \cdot \vec{s}_j = 0$$

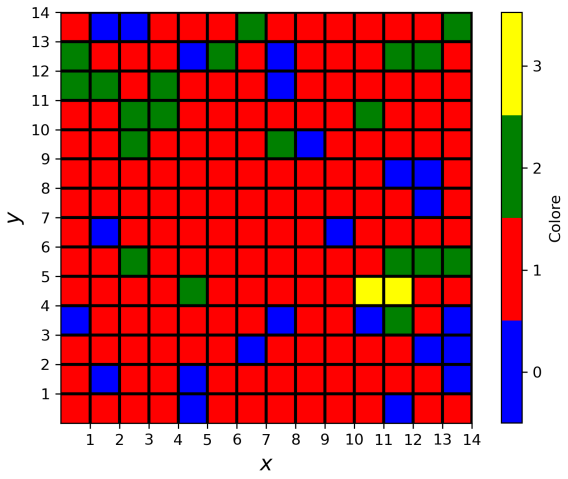
Sfavorita



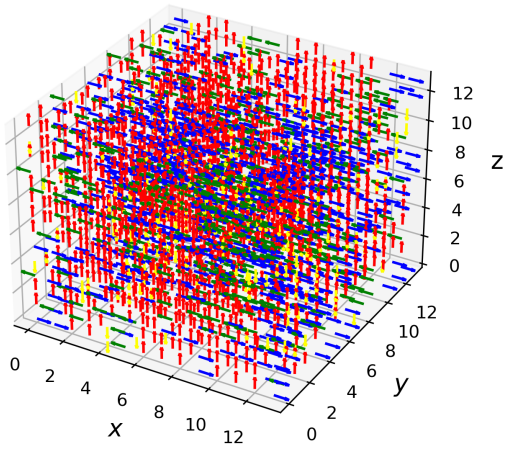
$$-J\ \vec{s}_i \cdot \vec{s}_j = +1$$

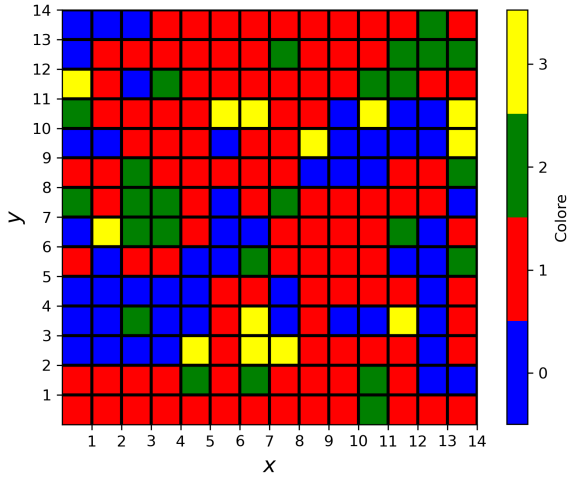
Comportamenti per alta e bassa temperatura



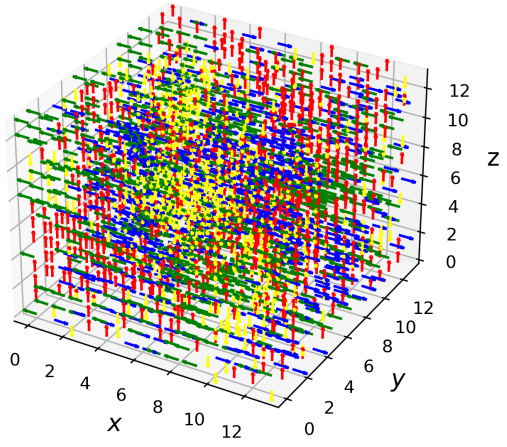


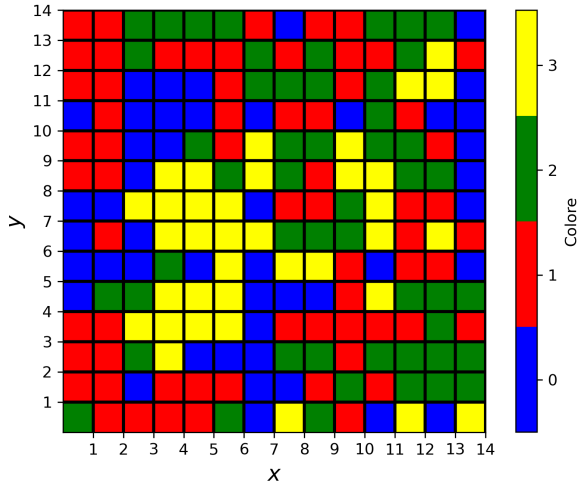
$T = 2.00J$





$T = 2.25J$





$T = 2.35J$

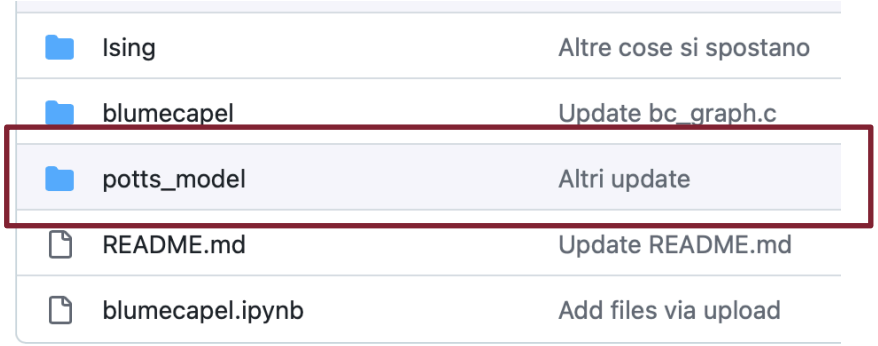
Inferenza tramite algoritmo Mean Field

Utilizzo del Codice

Scaricare la directory

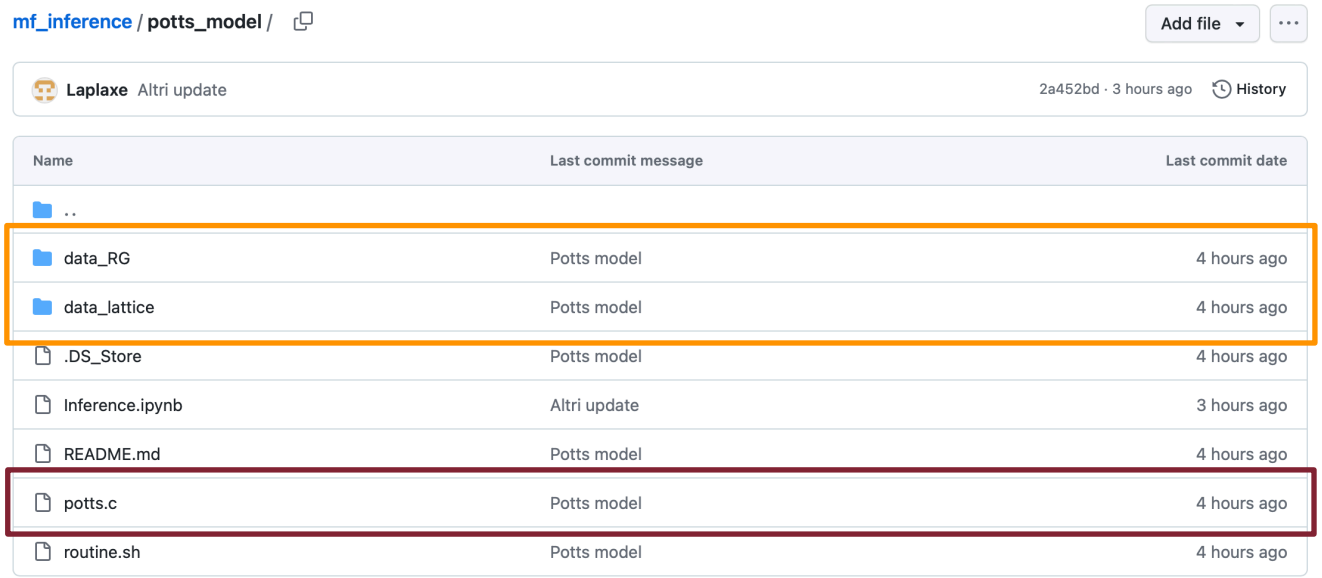
Scaricate la repository GitHub da: https://github.com/bsfn1844815/mf_inference.git

La directory relativa al modello di Potts è potts_model



Generare il dataset delle configurazioni

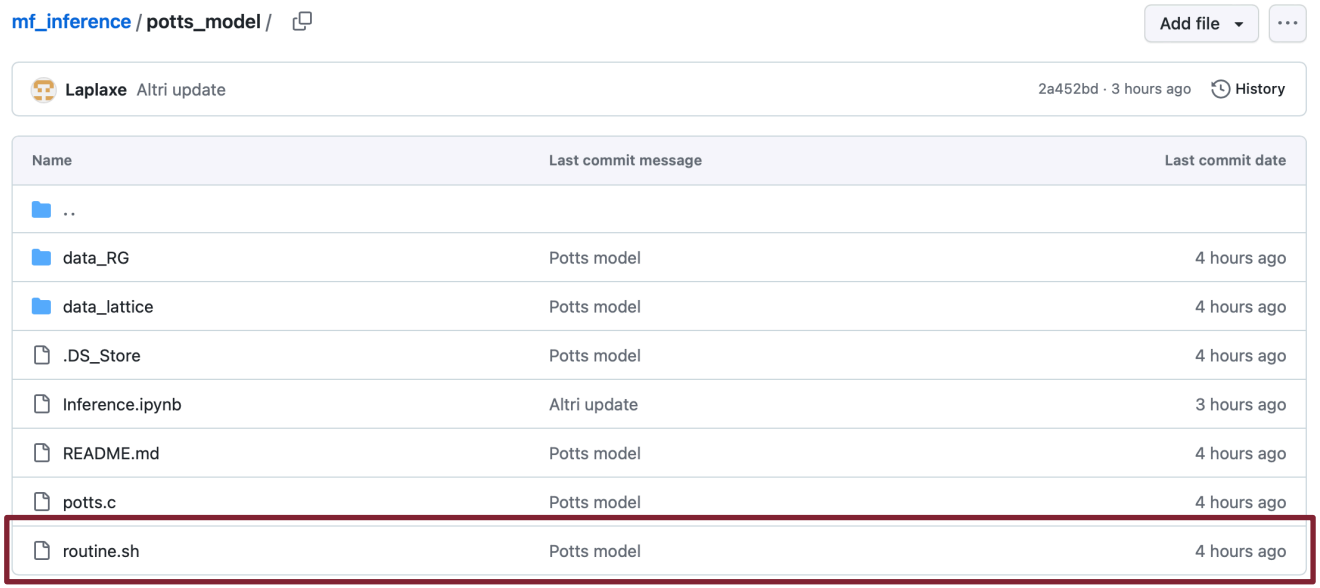
Per generare delle configurazioni, usate il programma potts.c



Potete anche utilizzare le configurazioni già generate contenute in data_RG e data_lattice

Generare il dataset delle configurazioni

Per generare dati in sequenza a varie temperature, potete utilizzare lo script bash routine.sh



Utilizzare potts.c

Nel programma avete a disposizione alcuni #define per settare i parametri della simulazione

```
#define L 3
#define TERMALIZE_SWEEPS 100000
#define NUM_SWEEPS 100000
```

Potete settare la taglia lineare del sistema, il numero di sweep della termalizzazione e il numero di sweep per cui vengono generati i dati

```
#undef REGULAR_LATTICE
#define ER_GNM_GRAPH
```

Potete scegliere se utilizzare un lattice regolare o un grafo di Erdős–Rényi

```
#define START_AT_INFINITE_TEMPERATURE
#undef START_AT_ZERO_TEMPERATURE
```

Potete scegliere se partire fa temperatura infinita (i.e. gli spin hanno orientazione casuale) o da temperatura nulla (i.e. tutti gli spin hanno $c = 0$)

Utilizzare potts.c

Dopo aver compilato il programma ed aver ottenuto l'eseguibile, potete lanciare la simulazione tramite

>>> *./nome_eseguibile <T> <J>*

Inserendo al posto di *<T>* e *<J>* i valori di temperatura e coupling che volete utilizzare

Utilizzare potts.c

Il programma produce 4 file .dat

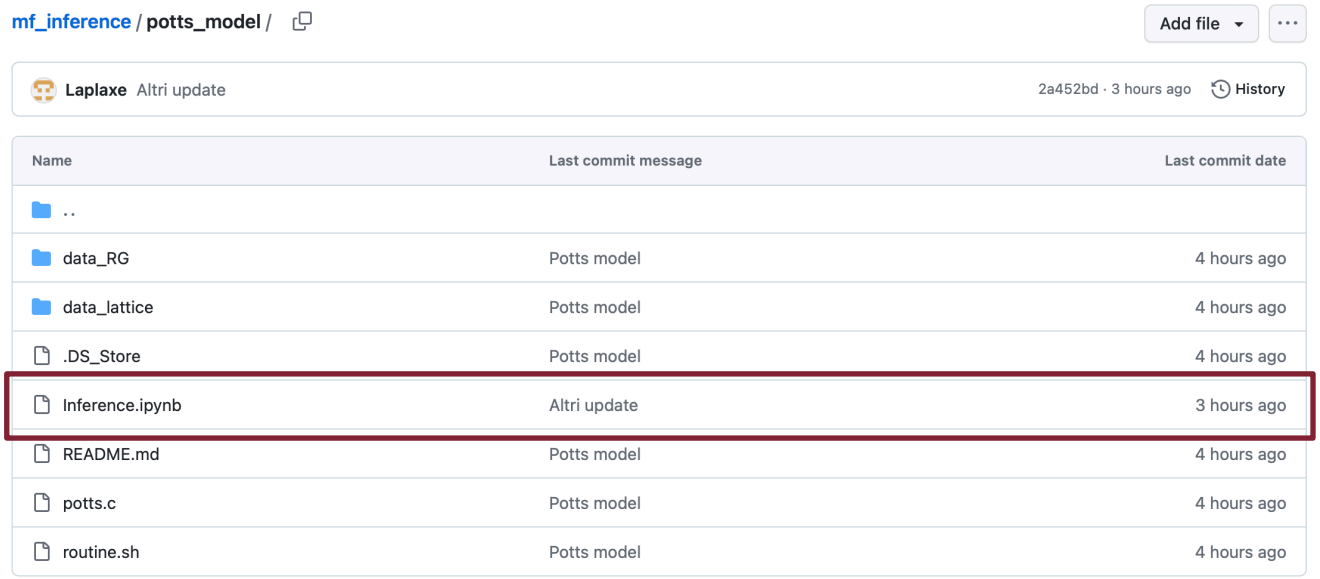
neighbours_L3_T1.50_J1.00.dat — File che contiene alla riga $i-$esima i vicini dello spin i

interaction_L3_T1.50_J1.00.dat — File che contiene la matrice di interazione J_0

config_L3_T1.50_J1.00.dat — File che contiene la configurazioni generate

Utilizzare il programma di inferenza

Le funzioni per fare inferenza sulle configurazioni generate sono contenute nel Jupyter Notebook Inference.ipynb



Nel notebook sono anche presenti alcuni grafici

Utilizzare Inference.ipynb

Le funzioni per fare inferenza sulle configurazioni generate sono contenute nel Jupyter Notebook Inference.ipynb

De facto, l'implementazione dell'algoritmo Mean Field avviene tramite la funzione MF_J

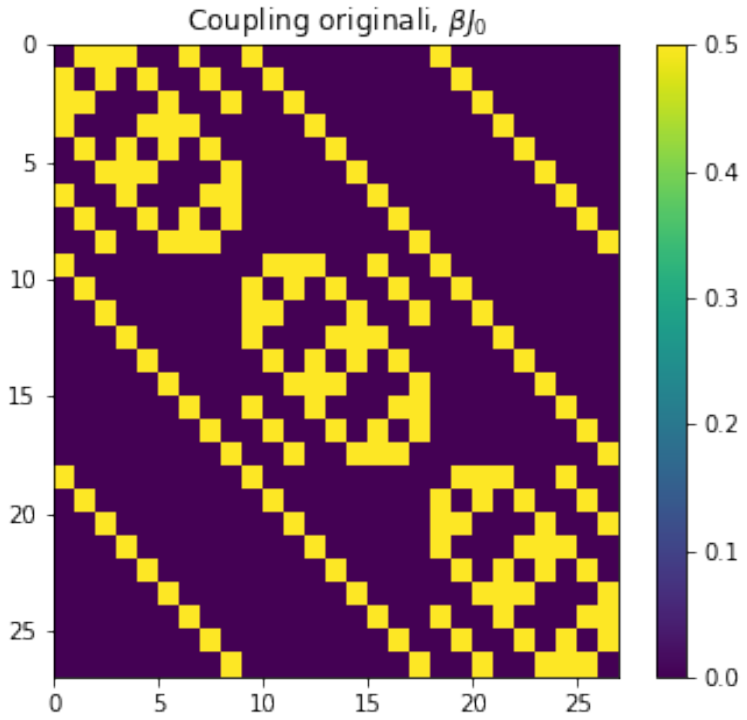
```
#Implementazione usando 'cov'

def MF_J(data, N):
    Cx = np.cov(np.split(data[:,2], int(len(data)/N)), rowvar = False) #Covarianza per la componente x
    Cy = np.cov(np.split(data[:,3], int(len(data)/N)), rowvar = False) #Covarianza per la componente y
    C = Cx+Cy
    J = - np.linalg.inv(C)
    return J
```

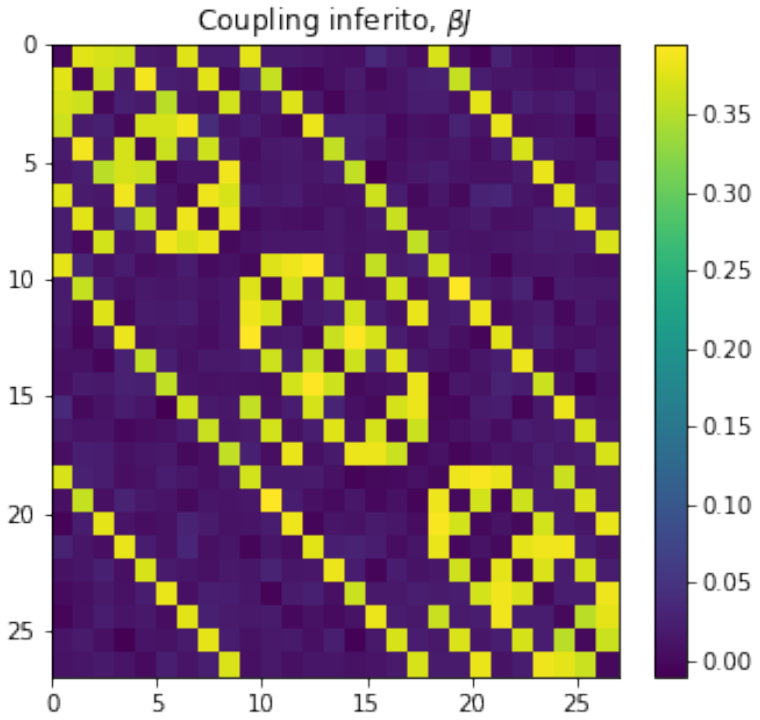
Inferenza tramite algoritmo Mean Field

Alcuni risultati

Coupling inferiti βJ vs coupling originali βJ_0

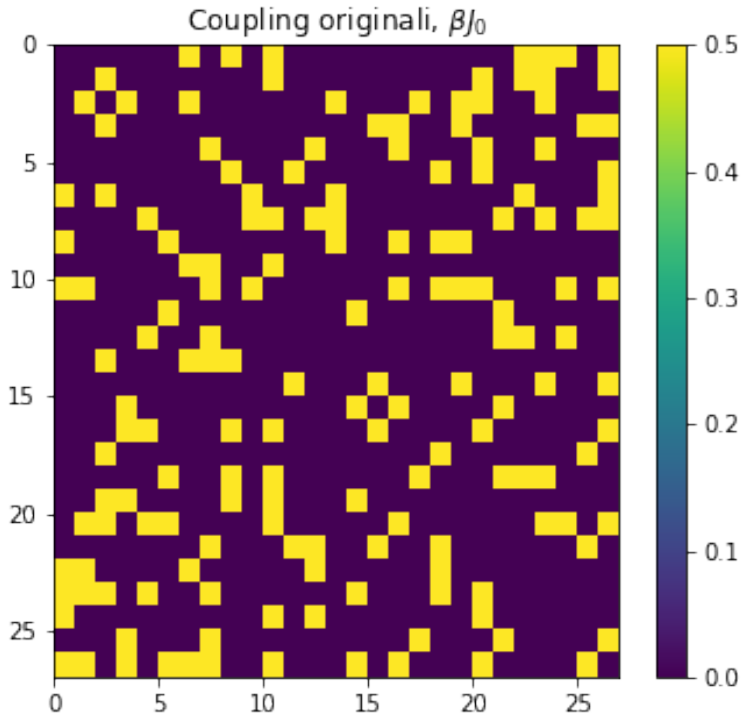
Reticolo cubico

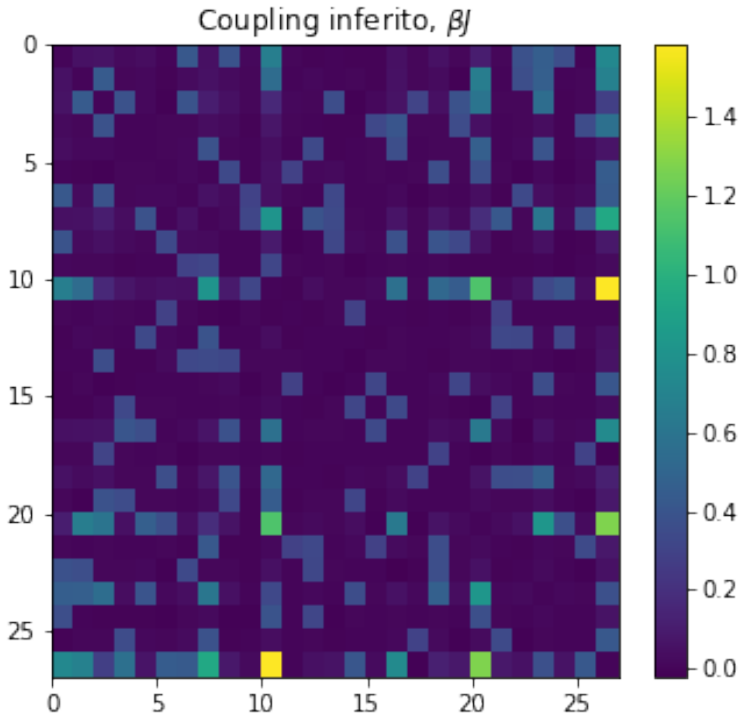




Coupling inferiti βJ vs coupling originali βJ_0

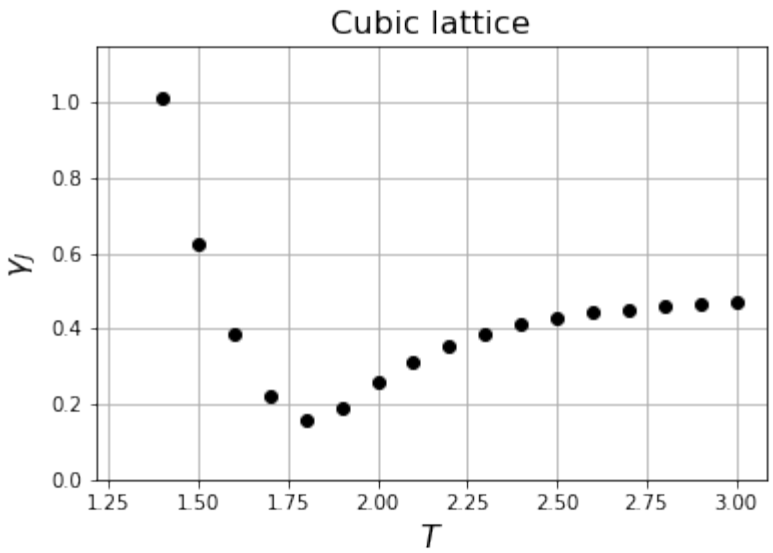
Reticolo cubico



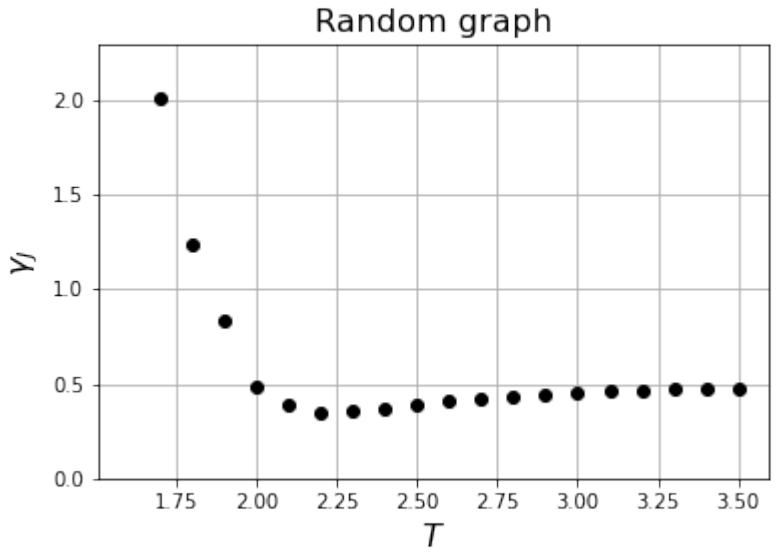


Errore γ_J vs T

Reticolo cubico



Grafo random



$$\gamma_J = \sqrt{\frac{\sum_{i<j}(J_{ij} - J_{0,ij})^2}{\sum_{i<j} J_{0,ij}^2}}$$