PERCOLAZIONE: ANALISI TRAMITE CLUSTERING SU CUBO BINARIO

Alessandro Pavesi

Modellazione e Simulazioni Numeriche

Introduzione

In questa relazione tratteremo l'argomento della percolazione definita come *il movimento di un fluido attraverso un materiale poroso.*

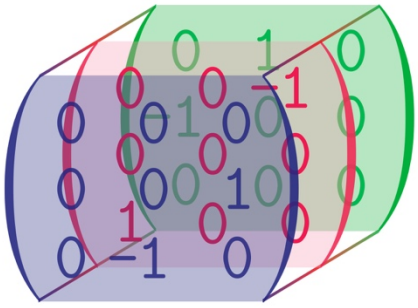
Studieremo il problema da un punto di vista simulativo, creando un modello di un materiale generico e ponendo l'attenzione sulla situazione di interesse: il fluido riesce a filtrare da una parte all'altra del materiale? Qual è il suo percorso attraverso il materiale? Cioè, qual è l'insieme di molecole che attraversa?

Per studiare la percolazione ci avvarremo della tecnica Clustering definita come *insieme di tecniche volte alla selezione e raggruppamento di elementi omogenei in un insieme di dati*.

Spiegazione

Creando un modello per il movimento di un fluido attraverso un materiale si deve tenere conto del numero di molecole all'interno del materiale, della loro disposizione e del loro movimento; studiando le stesse caratteristiche per il fluido e mettendo insieme i dati si può capire se fluisce attraverso da parte a parte o no.
Non disponendo di modelli realistici riferiti alla realtà e nemmeno le conoscenze per crearli ed elaborarli si deve semplificare la rappresentazione ragionando diversamente.

Si pensi al materiale come ad un cubo, questo cubo è necessariamente pieno di molecole come detto; intendendo queste molecole come un insieme ordinato di *siti* in modo da creare una matrice 3D. Ogni sito è caratterizzato da un valore, 0 o 1 che indica se il fluido passa o meno per esso. Questo valore è chiamata *variabile caratteristica*. Si definisce un sito con valore 1 *colorato* e con valore 0 *non colorato*, in questi termini possiamo parlare di *colorazione* del cubo.
A questo punto si ha una matrice binaria di 3 dimensioni che rappresenta il materiale, ogni sito è il modello di una particella e il valore ad essi associato caratterizza il materiale.



I valori assegnati ai siti sono la chiave per studiare il passaggio del fluido, partendo da un lato del cubo si cerca un sito colorato, questo vuol dire che il fluido passa per quel sito, si analizzano quindi i siti adiacenti, sull'asse x, y e z, alla ricerca di un sito con valore 1; se si riesce ad arrivare sulla faccia opposta del cubo seguendo un percorso continuo allora è avvenuta *percolazione*.

È possibile ovviamente che esistano più percorsi continui per arrivare da una faccia all'altra del cubo, come è possibile che ci siano percorsi che si interrompono e non riescano a continuare. Per

identificare i vari percorsi si assegna ad ogni sito l'etichetta riferita al percorso di cui fa parte, questa è l'utilizzo della tecnica *Clustering*. Un insieme di siti rappresentato dalla stessa etichetta è un *Cluster*.

Il procedimento è diviso tra la creazione del cubo e la sua colorazione e tra l'analisi dei cluster trovati. Durante l'analisi se si trovano molti siti con valore 1 significa che esiste una probabilità alta di percolazione, al contrario con pochi siti di valore 1 la percolazione sarà meno probabile.
La domanda che si pone quindi è: quanti siti devo colorare affinché ci sia percolazione?
In altri termini, qual è la probabilità minima di colorazione dei siti per cui avviene sicuramente percolazione?

Quindi, si crea un cubo di lato L, lo si colora con una probabilità P e utilizzando un algoritmo di Cluster Finding si identificano tutti i percorsi (cluster). Infine si verifica se in una parete del cubo e nella sua opposta esiste la stessa etichetta, ciò significa che un cluster è continuo da parete a parete e quindi è avvenuta percolazione.

Obbiettivo

L'obbiettivo è quello di verificare statisticamente quante volte un fluido riesce ad attraversare da una parte all'altra il materiale, verificando qual è la probabilità di colorazione minima dei siti del cubo per cui ho sicuramente percolazione, chiamando questo valore *soglia di percolazione*.
Per fare ciò si conducono varie simulazioni prendendo in considerazione varie probabilità di colorazione e varie dimensioni del cubo.

Definizioni Variabili

Le variabili utilizzate sono:

- P: probabilità di colorazione dei siti
- L: lato del cubo (# siti per lato)
- N: numero di prove da effettuare per ogni probabilità
- pMin: probabilità di partenza
- pMax: probabilità di arrivo
- pStep: step di avanzamento

Le variabili studiate sono:

- frequenza di percolazione (PP): definisce la frequenza di avvenuta percolazione.
- soglia di percolazione: probabilità minima in cui ho cubo teoricamente infinito ha sicuramente percolazione.
- P1: definisce la probabilità che un sito qualsiasi appartenga al cluster di dimensioni maggiori.
- P2: definisce la probabilità che un sito colorato appartenga al cluster di dimensioni maggiori
- P3: definisce la probabilità che un sito effettivamente colorato appartenga al cluster di dimensioni massime. Questa funzione è molto simile a P2 ed effettivamente il grafico è sovrapponibile, questo perché rappresentano le stesse informazioni con la differenza che P2 è un calcolo basato sulla probabilità di colorazione dei cluster mentre P3 si basa sul numero esatto di cluster colorati.

- RACS (Reduced Average Cluster Size): rappresenta il numero medio della dimensione dei cluster senza contare il cluster massimo.

Limiti

Utilizzando un cubo in tre dimensioni i siti sono aumentati esponenzialmente. Questo ha causato un rallentamento significativo nelle varie operazioni di creazione e ricerca dei cluster e nei tempi di elaborazione dell'algoritmo rispetto alla situazione a due dimensioni studiata a lezione. Si deve quindi limitare la dimensione del cubo e il numero di prove effettuate su esso, cercando di avere risultati utili in tempi ragionevoli.
Il numero di prove per ogni reticolo è di ordine 1e4.
Il lato del cubo su cui si effettuano le simulazioni è compreso tra 3 e 15 siti.
La probabilità varia da 0 a 1 con step intermedi di 0.01.

Algoritmo

L'algoritmo è diviso in due funzioni con compiti distinti:

- clusterExam3D(L, p) è la funziona dedicata alla creazione del cubo e al clustering. Funziona attribuendo ad ogni sito un indice continuo che parte dal sito in alto a sinistra della prima dimensione del cubo fino al sito in basso a destra dell'ultima dimensione. Per ogni sito colorato si valutano i vicini controllando gli assi x, y e z e aggiungendo ad una coda i siti colorati trovati. Si etichettano i siti adiacenti con la stessa etichetta e si aumenta il numero della dimensione del cluster. Si eseguono questi passaggi per ogni sito all'interno della coda e per ogni sito non ancora etichettato. In questo modo viene creato un cubo delle stesse dimensioni del cubo di partenza con all'interno le etichette dei cluster trovati.
- multiPercExam3D(L,pMin, pMax, pStep, N) è la funzione dedicata a condurre l'esperimento e a estrarne i parametri di interesse. Per ogni step incluso tra pMin e pMax ripete l'esperimento N volte. Questo consiste nella chiamata alla funzione clusterExam3D() e nel cercare tramite il confronto dei cluster di due lati opposti del cubo se è avvenuta percolazione. Quindi si occupa di calcolare il numero di volte in cui ha percolato e di P1, P2, P3 e RACS per ogni cubo. Per ogni probabilità di colorazione presa in considerazione si calcola la media e l'errore della frequenza di percolazione e di P1, P2, P3 e RACS sugli N dati.

  Sono stati creati altri algoritmi per la creazione di grafi e per il calcolo del tempo di elaborazione oltre che ad algoritmi per la rappresentazione 2D per il confronto dei dati.

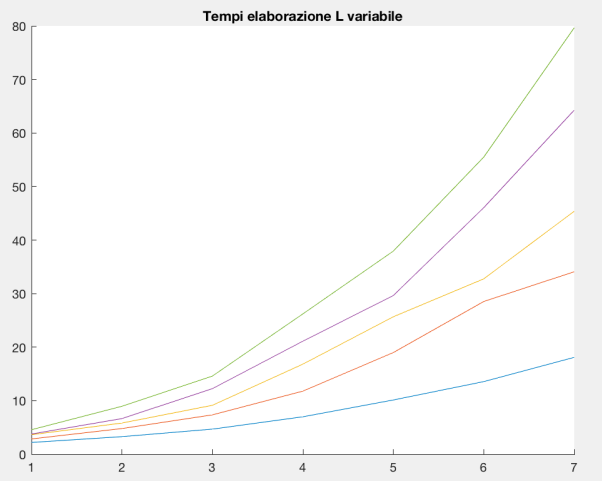
Allego gli algoritmi descritti a fine relazione.

Analisi tempo di elaborazione

I tempi di elaborazione sono stati calcolati tramite la funzione timeit() di Matlab.

P fisso e L variabile

Tempi di elaborazione per P=[.11, .25, .33, .45, .60] e L=[3, …, 10]:



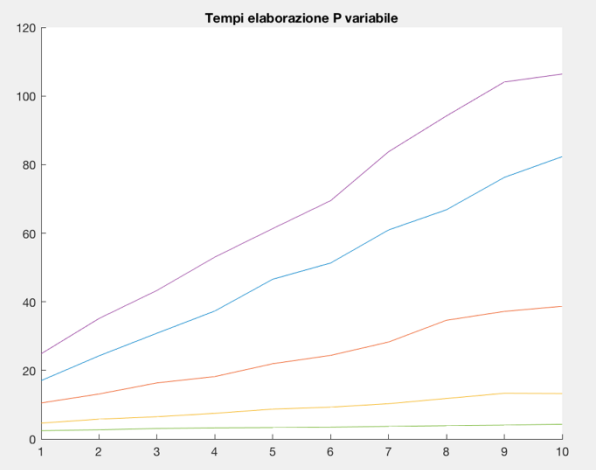
- blu: 0.11
- arancione: 0.25
- giallo: 0.33
- viola: 0.45
- verde: 0.60

Sull'asse X sono rappresentati i valori di L (1=3, 2=4, 2=5 e cosi via…).
Sull'asse Y sono rappresentati i secondi.

Viene evidenziato come all'aumentare della probabilità anche il tempo di elaborazione aumenta. Differentemente dal tempo di elaborazione rispetto alla dimensione del cubo il tempo aumenta di un fattore costante per ogni probabilità.

L fisso e P variabile

Tempi di elaborazione per L=[5,7,9,10,12] e P=[0.10, …, 0.55] con step di 0.05:



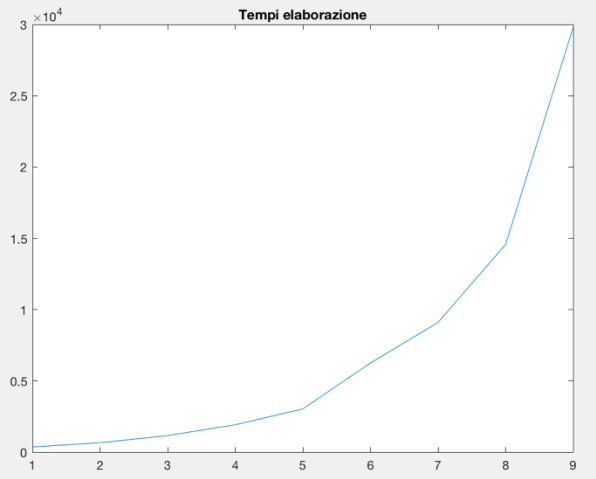
- verde: 5
- giallo: 7
- arancione: 9
- blu: 10
- viola: 12

Sull'asse X sono rappresentati i valori di P (1=0.10, 2=0.15, 2=0.20 e cosi via…).
Sull'asse Y sono rappresentati i secondi.

Si nota come l'incremento della probabilità di colorazione ha un effetto di aumentare linearmente il tempo di elaborazione.

Tempi di elaborazione della simulazione

L=[3,4,5,6,7,9,10,12,15] e P=[0, …,1]:



- T3 = 364.55 secondi, ~ 6 minuti.
- T4 = 661.87 secondi, ~ 11 minuti.
- T5 = 1162.18 secondi, ~ 19 minuti.
- T6 = 1922.73 secondi, ~ 32 minuti.
- T7 = 3032.77 secondi, ~ 50 minuti.
- T9 = 6254.22 secondi, ~ 1 ora e 40 minuti.
- T10 = 9098.19 secondi, ~ 2 ore e 30 minuti.
- T12 = 14569.39 secondi, ~ 4 ore.
- T15 = 29800.53 secondi, ~ 8 ore e 20 minuti.
- Sull'asse X sono rappresentati i valori di L (1=3, 2=4, 3=5 e cosi via…).
- Sull'asse Y sono rappresentati i secondi.

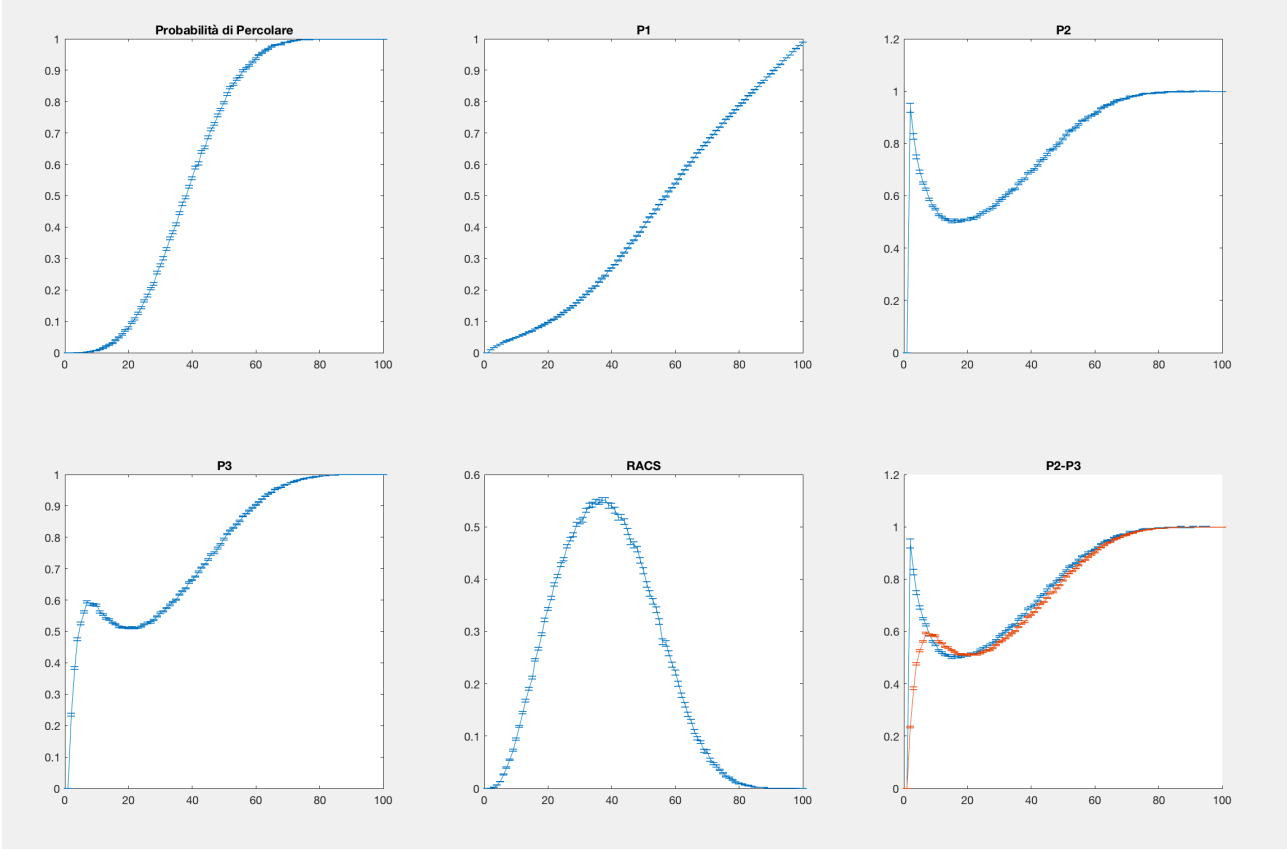
Si nota come il grafico ha andamento esponenziale rispetto alla dimensione del cubo. Questo perché l'aumento di siti aumenta i controlli da eseguire per il cluster finding.

Analisi numerica: test L fisso e P variabile tra 0 e 1

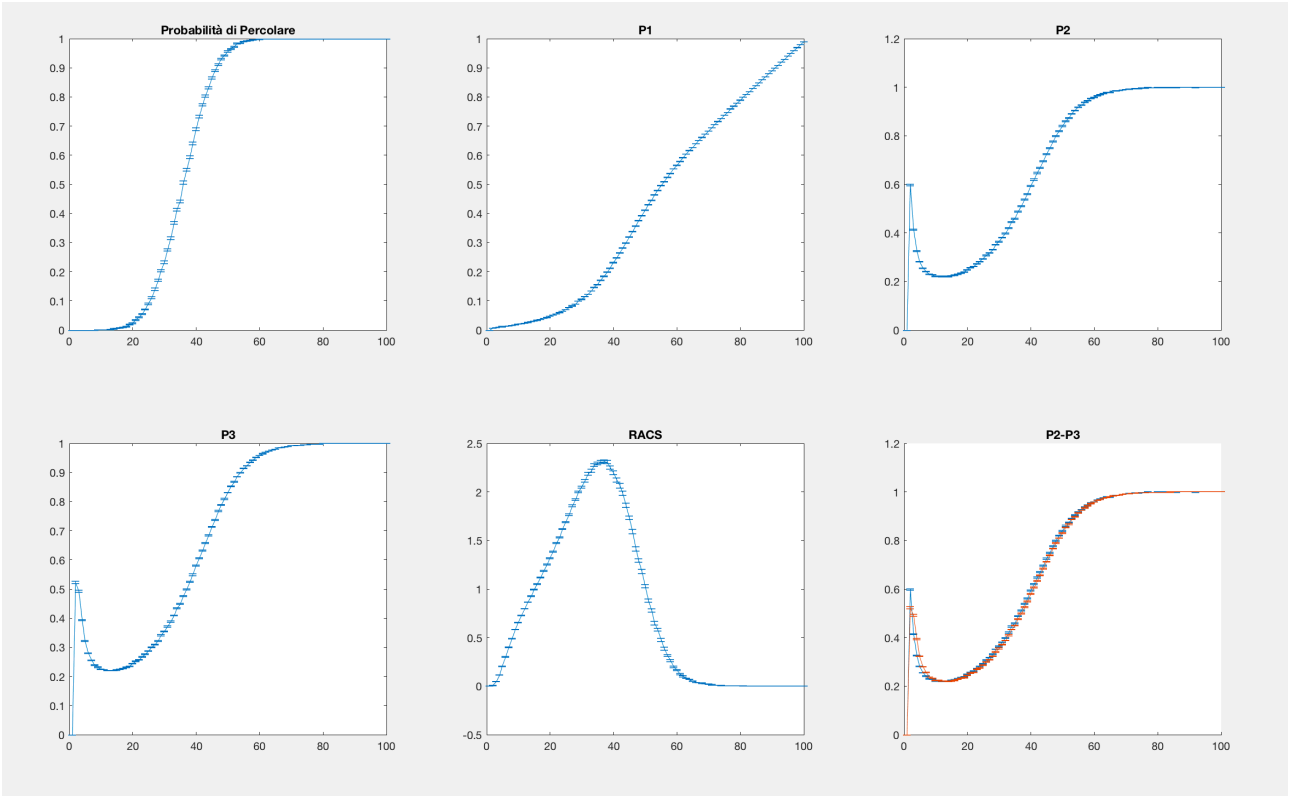
I grafici di PP, P1, P2, P3 hanno sull'asse X i valori di P e sull'asse Y i valori di PP. Il grafico di RACS ha sull'asse X i valori di P mentre sull'asse i valori di Y la dimensione dei cluster.

Vengono presentati i vari grafici per ogni dimensione del cubo studiata.

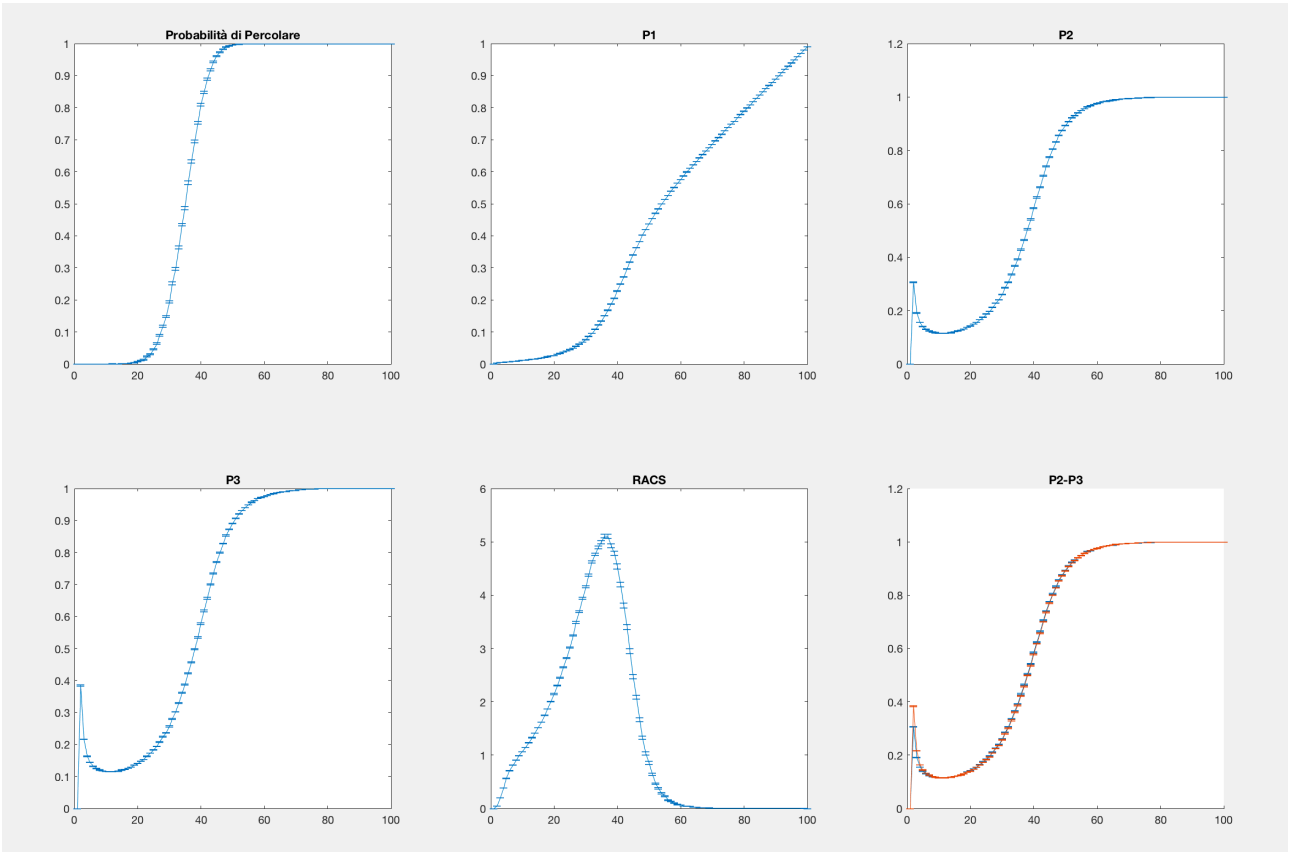
L = 3



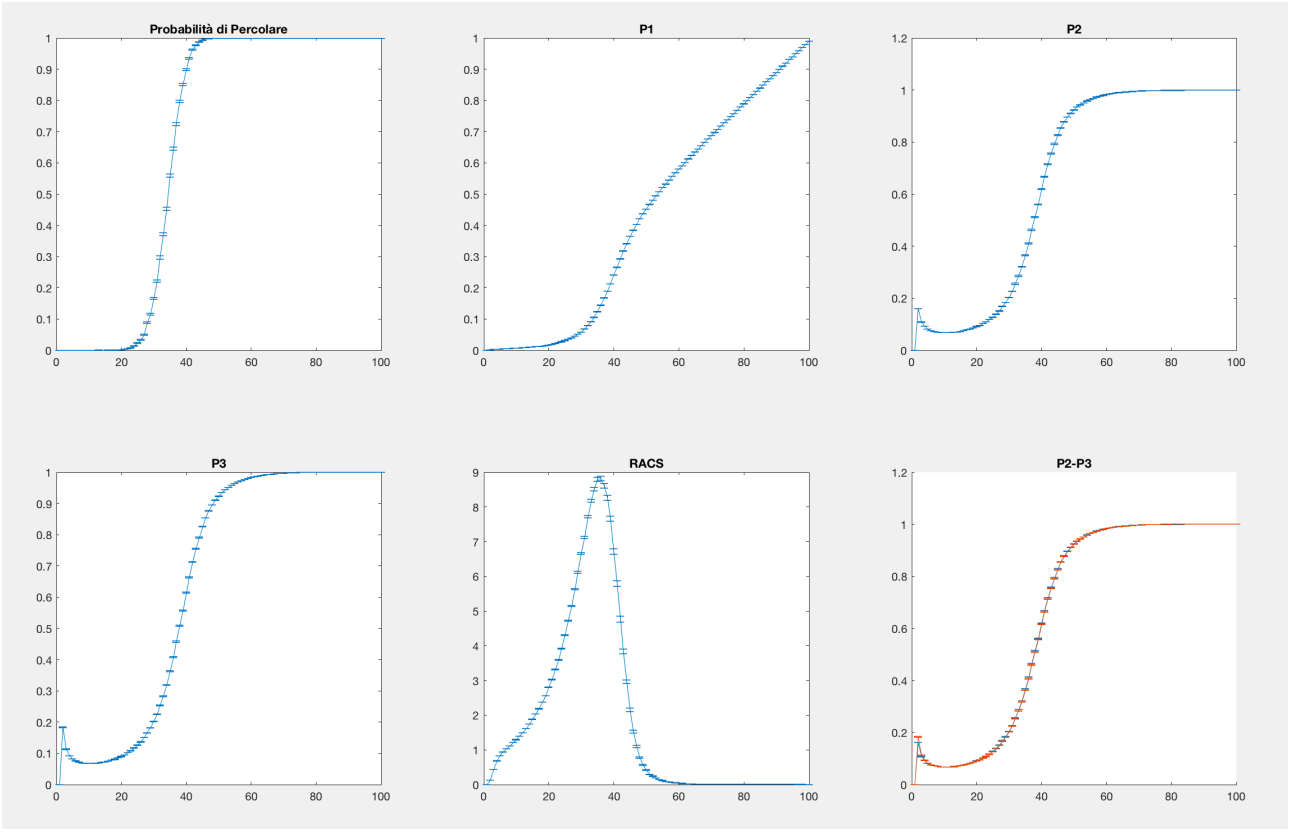
L = 5


Probabilità di Percolare
P1
P2
P3
RACS
P2-P3

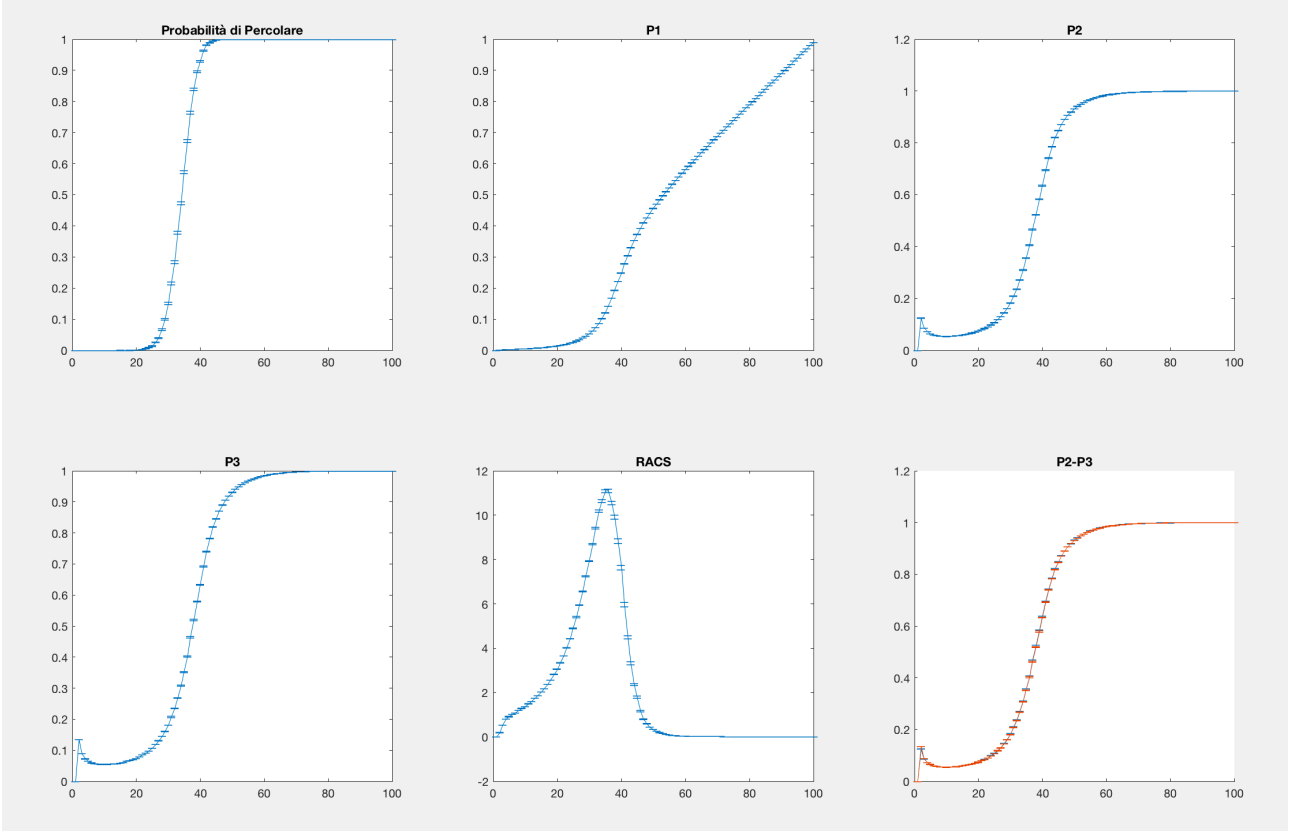
L = 7


Probabilità di Percolare
P1
P2
P3
RACS
P2-P3

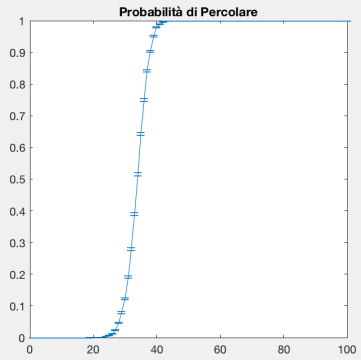
L = 9

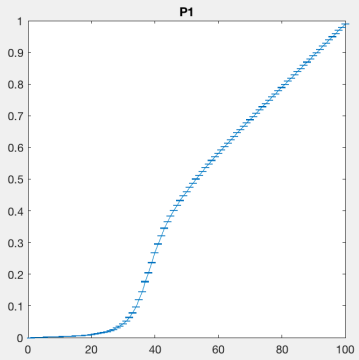

Probabilità di Percolare
P1
P2
P3
RACS
P2-P3

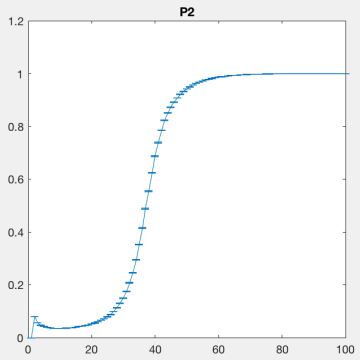
L = 10

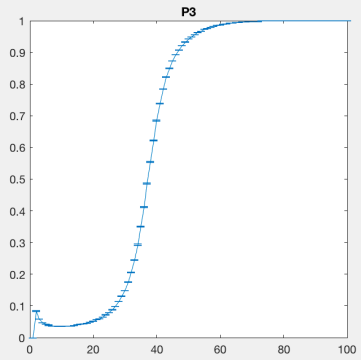

Probabilità di Percolare
P1
P2
P3
RACS
P2-P3

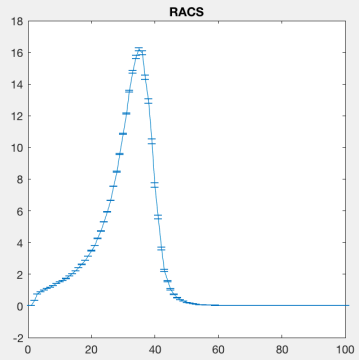
L = 12

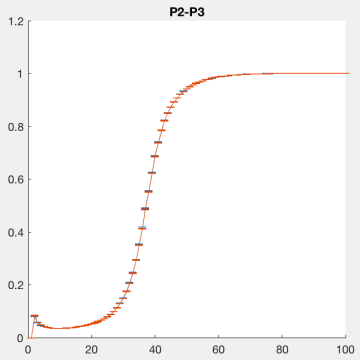

Probabilità di Percolare


P1

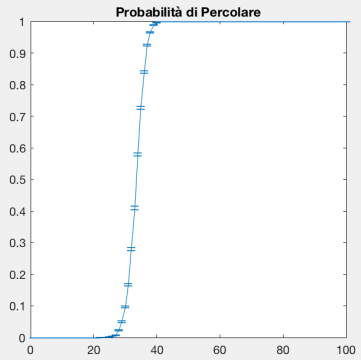

P2

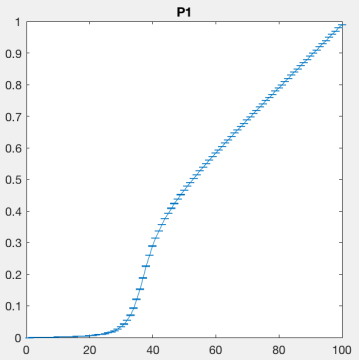

P3

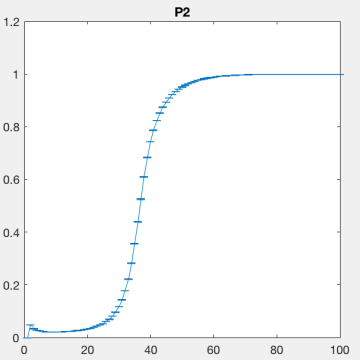

RACS

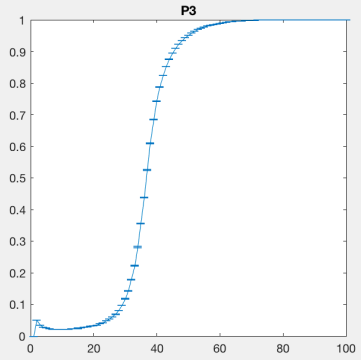

P2-P3

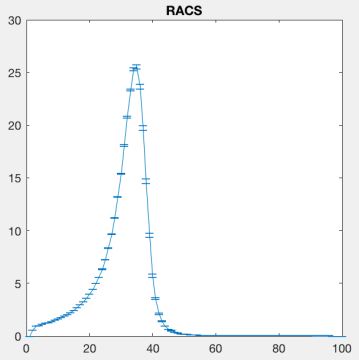
L = 15

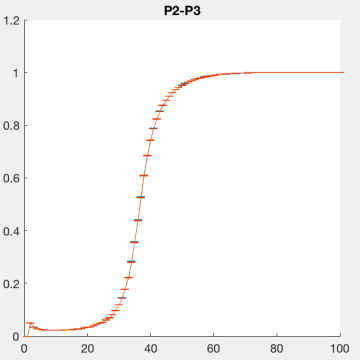

Probabilità di Percolare


P1


P2


P3


RACS


P2-P3

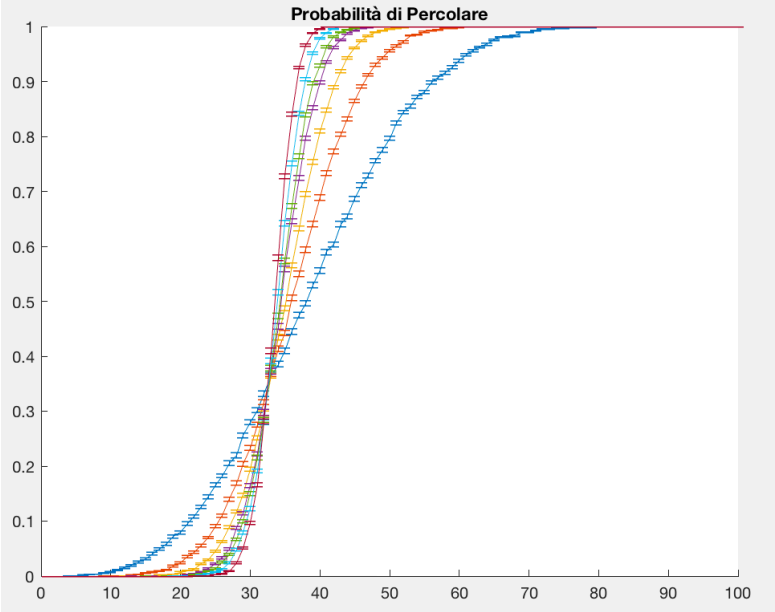
Confronti numerici: test L fisso e P variabile tra 0 e 1

Dopo aver mostrato i dati per ogni singola dimensione presa in esame confrontiamo i dati per le variabili studiate. I grafici di PP, P1, P2, P3 hanno sull'asse X i valori di P e sull'asse Y i valori di PP. Il grafico di RACS ha sull'asse X i valori di P mentre sull'asse i valori di Y la dimensione dei cluster. Ogni dimensione ha un colore specifico:

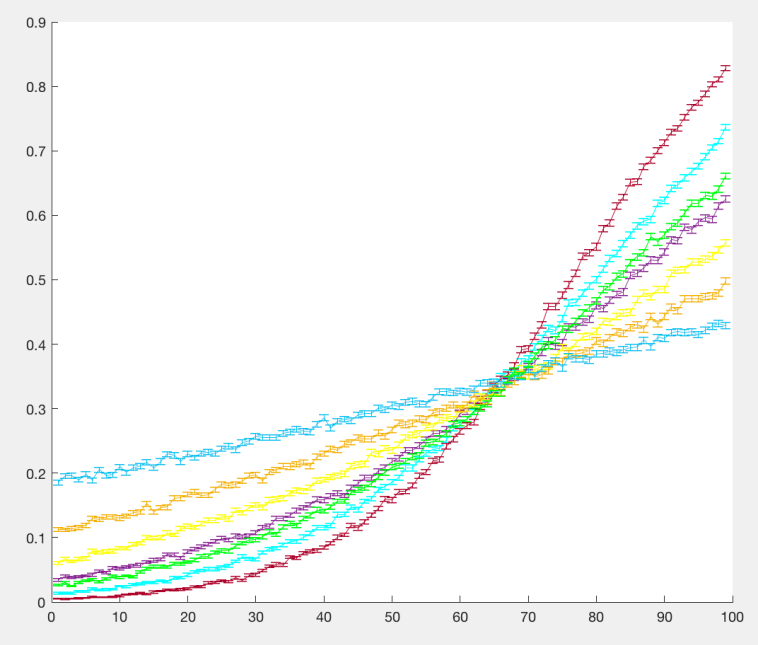
- L=3 è il colore Blu
- L=5 è il colore Arancione
- L=7 è il colore Giallo
- L=9 è il colore Viola
- L=10 è il colore Verde
- L=12 è il colore Azzurro
- L=15 è il colore Magenta

Analisi dei grafici:

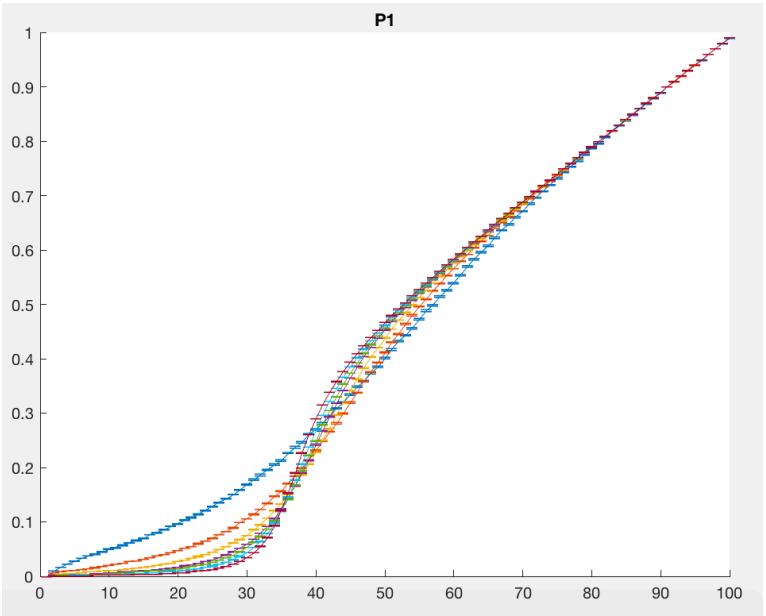
- la frequenza di percolazione crea un grafico sigmoidale, aumentando la dimensione del cubo la crescita della funzione è sempre più rapida. Questo indica che se si potesse aumentare la dimensione del cubo all'infinito si avrebbe non più una sigmoide ma una funzione lineare con valori 0 e 1 e un punto di discontinuità esattamente nel valore della soglia di percolazione.



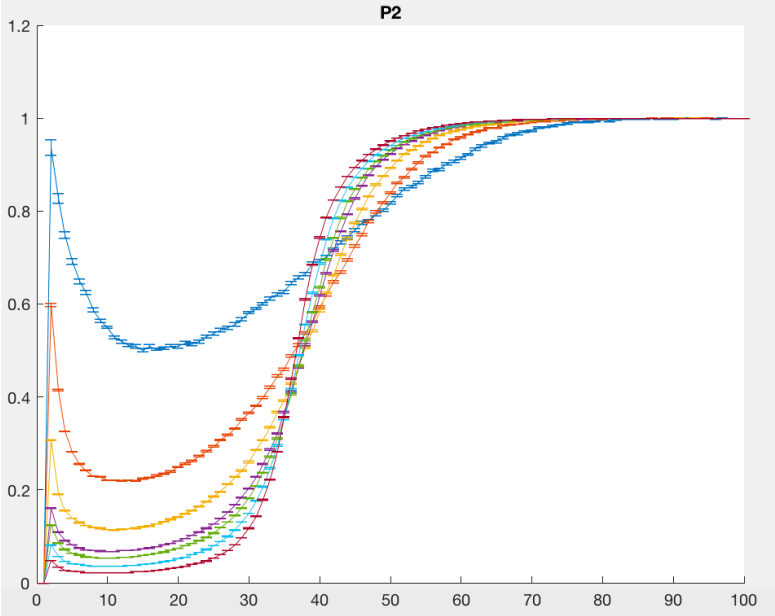
- Si mostra l'ingrandimento del punto di flesso della sigmoide. L'asse X ha valori compresi tra 0.25 e 0.35 con step di avanzamento di 0.001, mentre l'asse Y è la frequenza di percolazione. Il punto in cui le funzioni si incrociano è il punto di discontinuità all'interno del grafico. Si ricordi che se si facesse tendere la dimensione del cubo all'infinito si avrebbe un grafico lineare con un punto di discontinuità nell'esatto punto di soglia. Si può dedurre per cui che la soglia di percolazione è compresa tra 0.315 e 0.32.



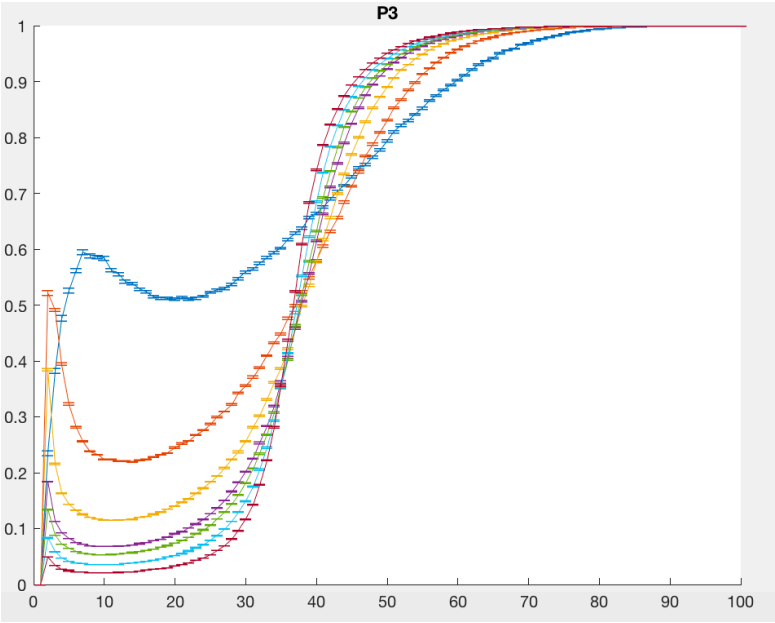
- la variabile P1 rappresenta la probabilità che un sito qualunque appartenga al cluster di dimensione massima. Si nota che per probabilità di colorazione basse, P1 rimane vicina allo zero questo perché ci sono molti siti di piccole dimensioni e soltanto alcuni di dimensioni più grosse (se non soltanto uno) quindi i siti tendono a creare cluster distinti. Avvicinandosi al punto di soglia si ha un aumento di P1 che, superato essa, diventa praticamente la bisettrice del quadrante ad indicare che la probabilità di appartenere al cluster di dimensioni massime cresce proporzionalmente alla probabilità di colorazione, fino a diventare massima.



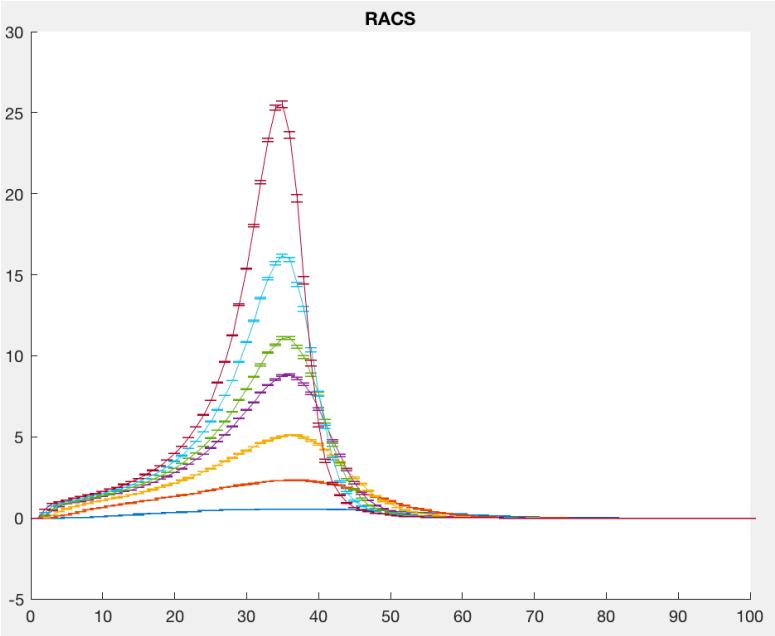
- la variabile P2 indica la probabilità che un sito colorato appartenga al cluster di dimensioni massime. I primi dati, distanti dall'andamento successivo della funzione, derivano dalle dimensioni relativamente piccole del cubo in questione. Focalizzando l'attenzione sui dati prima della soglia notiamo che i valori sono molto bassi a causa della poca probabilità di colorazione e quindi dei pochi siti aggregati e dei molti cluster di piccole dimensioni. Oltrepassando la soglia vediamo una percentuale sostanzialmente vicino alla probabilità massima a significare che normalmente esiste un cluster di dimensioni massime che prende la maggioranza dei siti colorati.



- la variabile P3 indica la probabilità che un sito appartenga al cluster massimo basata sul valore reale dei siti colorati, molto simile quindi alla funzione di P2. Se sovrapposte (come nei casi singoli visti sopra) sono quasi coincidenti. Le considerazioni fatte per P2 sono valide anche per P3.



- la variabile RACS rappresenta il valore medio ridotto della dimensione del cluster, calcolata quindi senza la dimensione del cluster massimo. La curva di questo grafo ha il suo valore massimo nel punto di soglia, questo perché la soglia di percolazione è il punto in cui abbiamo la numerosità massima di cluster distinti, mentre appena superata la soglia notiamo che la funzione si appiattisce rapidamente sull'asse X ad indicare che i siti appartengono al cluster massimo e la numerosità di cluster distinti è minima.

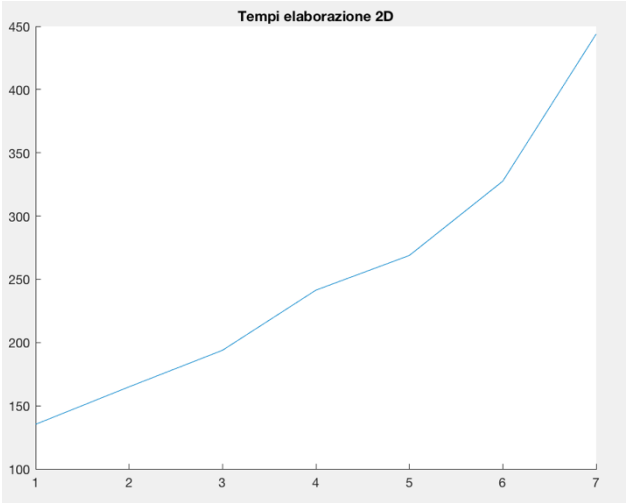


Si nota come in tutti i grafici vi è un cambiamento nell'andamento della funziona intorno alla soglia di percolazione. Ogni singolo grafico quindi può indicare questo valore.

Breve analisi bi dimensionale

Utilizzando una matrice di due dimensioni ed elaborandola con lo stesso obbiettivo si troveranno risultati diversi. La prima considerazione è la differenza di struttura su cui effettuare test, lavorando in 2 dimensioni si utilizza una matrice binaria.

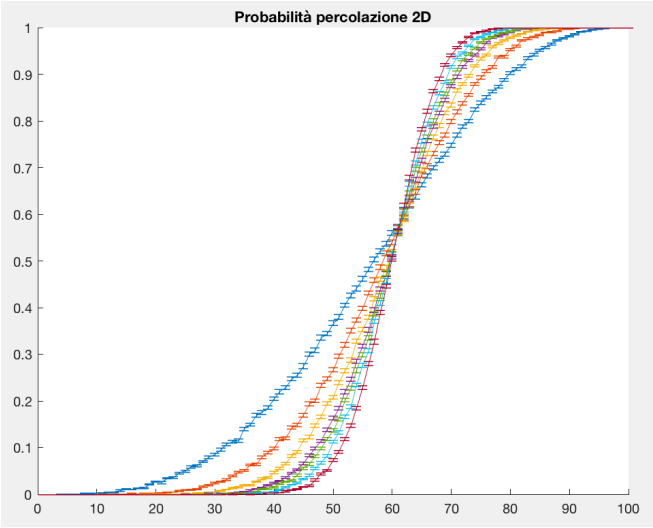
Questi grafici sono creati con gli stessi dati con cui sono state fatte le altre prove:
L=[3,4,5,6,7,9,10,12,15], P=[0, …,1] e N=1e4.
I tempi di elaborazione diminuiscono:



L'asse delle Y indica un valore massimo di 450 secondi confronto ai circa 30000 secondi calcolati con l'elaborazione 3D.

Le matrici possono avere quindi lati di dimensioni molto maggiori confronto al cubo anche di un fattore 100, così come il numero di prove possibili per ogni cubo può passare da 1e4 a 1e6 e rimanere in tempi di elaborazione comparabili.

La soglia di percolazione cambia passando ad un valore compreso tra 0.61 e 0.62:

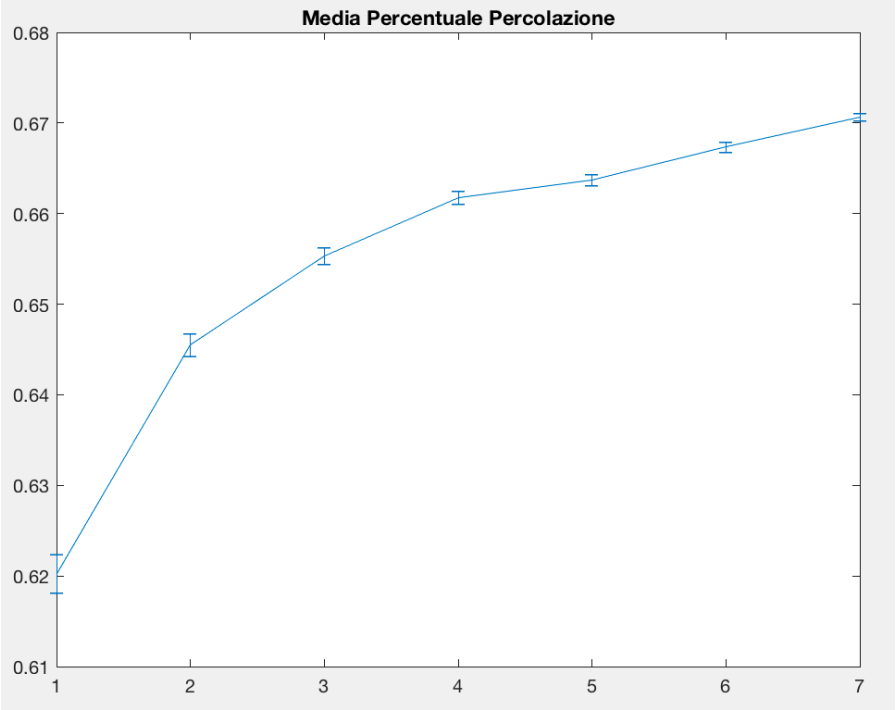


Per avere percolazione quindi è necessario un numero più alto di siti colorati.

Conclusioni

L'obbiettivo di questa relazione è cercare la soglia di percolazione all'interno di un cubo di 3 dimensioni. Si è verificato che il valore è compreso tra 0.315 e 0.32. Questo valore potrebbe diventare più preciso tramite l'elaborazione di cubi con dimensioni maggiori (L). Un altro metodo sarebbe quello di aumentare il numero di prove per ogni reticolo (N). Purtroppo in questa analisi non sono stati presi in considerazione valori più grandi a causa della potenza di calcolo limitata e del tempo impiegato. Tornando alla soglia di percolazione, il valore è circa la metà del valore di soglia per una matrice in due dimensioni, spiegabile dal fatto che i siti adiacenti aumentano e per arrivare da una parete all'altra del cubo è possibile creare percorsi che si muovono attraverso le varie dimensioni del cubo. Si è notato inoltre che aumentando la probabilità di colorazione si viene a formare un cluster che include la maggioranza dei siti.

La media della probabilità di percolazione utilizzando un cubo in tre dimensioni è compresa tra 0.66 e 0.67, molto maggiore della media di probabilità su una matrice che è compresa tra 0.38 e 0.40. Questo risultato deriva da una maggiore possibilità di espandersi e girare intorno ad ostacoli da parte del fluido.



Sull'asse X abbiamo le varie dimensioni di L, sull'asse Y abbiamo le percentuali di percolazione.

Bibliografia

- https://www.matematicamente.it/rivista-il-magazine/numero-9-aprile-2009/112-data-mining-esplorando-le-miniere-alla-ricerca-della-conoscenza-nascosta-clustering/
- dispense del corso

clusterExam3D

```
function M = clusterExam3D(N, p)
% Ricerca di cluster all'interno di una matrice N*N con probabilit‡ p
% che un sito sia colorato
% Ritorna una struttura dati:
% .m Ë la matrice3D iniziale
% .e Ë la matrice3D con le etichette
% .d Ë un vettore con le dimensioni dei cluster trovati

% Creo una matrice 3D N*N*N in cui visualizzo i siti secondo la
% probabilita p
d=zeros(N,N,N);
M.m = [];
for k=1:N
    d(:,:,k)=rand(N)<p;
end
M.m = d;
 % Creo una matrice 3D N*N*N in cui inserirÚ le etichette
d=zeros(N,N,N);
for k=1:N
    d(:,:,k)=0;
end
M.e = d;
M.d = [];
dim = N;
% contiene la matrice degli indici
I = reshape(1:N^3,dim,dim,dim);

% etichetta di partenza
etichetta = 0;
% alloco la memoria per la coda
% all'interno della coda ci saranno gli indici vicini all'indice che
% siamo valutando che dovranno essere valutati
coda = zeros(1, N^3);

% ciclo tutti gli elementi della matrice
for i = 1:N^3
    % se l'elemento della matrice non Ë etichettato ed Ë valorizzato 1
    % procedo con la vlautazione
    if (~M.e(i) && M.m(i))
        % lunghezza coda
        len_coda = 1;
        % inizializzo la coda
        coda(len_coda) = i;
        % etichetta ++
        etichetta = etichetta + 1;
        % nella matrice delle etichette segno con l'etichetta corrente
        % il sito che sto valutando
        M.e(i) = etichetta;
        % inizializzo contatore coda
        j = 1;
        % ciclo la coda
        while j <= len_coda
            % estraggo sito dalla coda
            sito = coda(j);

            % creo i vicini per il sito che sto valutando
            % creo gli indici di vicini
            % destro
            nnr = [I(:, 2:end, ceil(sito/(N)^2)) zeros(dim, 1, 1)];
            % sinistro
```

```
nnl = [zeros(dim,1,1) I(:, 1:end-1, ceil(sito/(N)^2))];
% su
nnu = [zeros(1,dim,1); I(1:end-1, :, ceil(sito/(N)^2))];
% giu
nnd = [I(2:end, :, ceil(sito/(N)^2)); zeros(1,dim,1)];
% avanti (3D cioË verso il lettore)
if ceil(sito/(N)^2)-1 > 0 && ceil(sito/(N)^2)-1 <= N
    nna = I(:, :, ceil(sito/(N)^2)-1);
else
    nna = zeros(dim,dim);
end
% indietro (3D cioË verso il dentro dello schermo)
if ceil(sito/(N)^2)+1 > 0 && ceil(sito/(N)^2)+1 <= N
    nni = I(:, :, ceil(sito/(N)^2)+1);
else
    nni = zeros(dim,dim);
end

% per prendere l'indice corretto all'interno delle
% tabelle di vicini devo prendere il sito 2D dal sito
% 3D
if sito >= N^2
    sito2D = sito-N^2*(ceil(sito/N^2)-1);
else
    sito2D = sito;
end

% se il SITO SOPRA esiste, se il sito sopra Ë 1, se il sito
% sopra non ha etichetta
% allora Ë un sito da valutare e lo inserisco nella coda
if nnu(sito2D) && M.m(nnu(sito2D)) && ~M.e(nnu(sito2D))
    % aumento la lunghezza della coda
    len_coda = len_coda + 1;
    % aggiungo alla coda il sito sopra
    coda(len_coda) = nnu(sito2D);
    % assegno al sito sopra l'etichetta corrente
    M.e(nnu(sito2D)) = etichetta;
end

% se il SITO SOTTO esiste, Ë 1 e non ha etichetta
% allora Ë un sito da valutare e lo inserisco nella coda
if nnd(sito2D) && M.m(nnd(sito2D)) && ~M.e(nnd(sito2D))
    len_coda = len_coda + 1;
    % aggiungo alla coda il sito destro
    coda(len_coda) = nnd(sito2D);
    % assegno al sito giu l'etichetta corrente
    M.e(nnd(sito2D)) = etichetta;
end

% se il SITO SINISTRO esiste, se Ë 1 e non ha etichetta
% allora Ë un sito da valutare e lo inserisco nella coda
if nnl(sito2D) && M.m(nnl(sito2D)) && ~M.e(nnl(sito2D))
    len_coda = len_coda + 1;
    % aggiungo alla coda il sito sinistro
    coda(len_coda) = nnl(sito2D);
    % assegno al sito sinistro l'etichetta corrente
    M.e(nnl(sito2D)) = etichetta;
end

% se il SITO DESTRO esiste, Ë 1 e non ha etichetta
% allora Ë un sito da valutare e lo inserisco nella coda
if nnr(sito2D) && M.m(nnr(sito2D)) && ~M.e(nnr(sito2D))
```

```
                    len_coda = len_coda + 1;
                    % aggiungo alla coda il sito destro
                    coda(len_coda) = nnr(sito2D);
                    % assegno al sito destro l'etichetta corrente
                    M.e(nnr(sito2D)) = etichetta;
                end

                % se il SITO AVANTI esiste, Ë 1 e non ha etichetta
                % allora Ë un sito da valutare e lo inserisco nella coda
                %sito
                if nna(sito2D) && M.m(nna(sito2D)) && ~M.e(nna(sito2D))
                    len_coda = len_coda + 1;
                    % aggiungo alla coda il sito destro
                    coda(len_coda) = nna(sito2D);
                    % assegno al sito destro l'etichetta corrente
                    M.e(nna(sito2D)) = etichetta;
                end

                % se il SITO INDIETRO esiste, Ë 1 e non ha etichetta
                % allora Ë un sito da valutare e lo inserisco nella coda
                %M.e(nni(sito))
                if nni(sito2D) && M.m(nni(sito2D)) && ~M.e(nni(sito2D))
                    len_coda = len_coda + 1;
                    % aggiungo alla coda il sito destro
                    coda(len_coda) = nni(sito2D);
                    % assegno al sito destro l'etichetta corrente
                    M.e(nni(sito2D)) = etichetta;
                end

                % aumento l'indice della coda
                j = j + 1;
            end
            % aumento la dimensione del cluster
            M.d = [M.d, len_coda];
        end
    end

end
```

multiPercExam3D

```
function [MultiPperc, MultiErrPperc, MultimP1Perc, MultimP2Perc, MultimP3Perc,
MultimRacsPerc, MultiErrP1Perc, MultiErrP2Perc, MultiErrP3Perc,
MultiErrRacsPerc] = multiPercExam3D(L,pMin, pMax, pStep, N)
    % Programma per verificare se c'Ë percolazione N volte sull'intervallo
    % pMin - pMax di probabilit‡ di colorazione del sito
    % L Ë la dimensione della mia matrice
    % PpMin Ë la probabilit‡ di partenza
    % N Ë il numero di volte che devo provare.
    % pStep Ë lo step di avanzamento della probabilit‡

    % inizializzazione variabili
    % numero prove con p diverso
    if pMax < pMin
        return
    end
    if pMax == pMin
        deltaP = 1;
    else
        deltaP = floor((pMax-pMin)/pStep);
    end

    % variabili per la frequenza di percolare/valor medio e la sua varianza
    MultiPperc = zeros(1,deltaP);
    MultiErrPperc = zeros(1,deltaP);
    % MultivarPperc = zeros(1,deltaP);
    MultinClusterPerc = zeros(1,deltaP);

    % variabili per la media e la varianza di p1,p2,p3 e racs
    MultimP1Perc = zeros(1,deltaP);
    % MultivarP1Perc = zeros(1,deltaP);
    MultiErrP1Perc = zeros(1,deltaP);

    MultimP2Perc = zeros(1,deltaP);
    % MultivarP2Perc = zeros(1,deltaP);
    MultiErrP2Perc = zeros(1,deltaP);

    MultimP3Perc = zeros(1,deltaP);
    % MultivarP3Perc = zeros(1,deltaP);
    MultiErrP3Perc = zeros(1,deltaP);

    MultimRacsPerc = zeros(1,deltaP);
    % MultivarRacsPerc = zeros(1,deltaP);
    MultiErrRacsPerc = zeros(1,deltaP);

    % ripeto l'esperimento N volte con p variabile a step di 0.01
    for z=1:deltaP
        p = pMin+(pStep*(z-1))
        % mi salvo per ogni prova se ha percolato o no
        xPerc = zeros(1,N);
        nClusterPerc = zeros(1,N);
        % mi salvo per ogni prova le variabili p1,p2,p3,racs
        p1Perc = zeros(1,N);
        p2Perc = zeros(1,N);
        p3Perc = zeros(1,N);
        racsPerc = zeros(1,N);

        % ripeto l'esperimento della percolazione N volte per un p
        for i=1:N
            r = clusterExam3D(L, p);
            % prendiamo le etichette della prima e ultima riga di ogni
```

```
            % dimensione
            % dentro cOne ci metto le etichette dei cluster trovati nella prima
            % colonna di ogni dimensione
            cOne = [];
            % dentro cLast ci metto le etichette dei cluster trovati nella
            % ultima colonna di ogni dimensione
            cLast = [];
            for k=1:L
                % prendo le etichette dalla prima colonna
                uOne = unique(r.e(:, 1, k));
                % unique mi da la colonna, a me serve un vettore quindi faccio
                % questo for anche se non Ë bello
                uOneArray = [];
                for w=1:length(uOne)
                    uOneArray = [uOneArray uOne(w)];
                end
                cOne = [cOne uOneArray];

                % prendo le etichette dalla ultima colonna
                uLast = unique(r.e(:, end, k));
                uLastArray = [];
                for w=1:length(uLast)
                    uLastArray = [uLastArray uLast(w)];
                end
                cLast = [cLast uLastArray];
            end
            cOne = unique(cOne);
            cLast = unique(cLast);
            % elimino l'etichetta 0
            cOne(cOne==0) = [];
            cLast(cLast==0) = [];

            % salvo quante percolazioni ho avuto
            nPerc = 0;
            % verifichiamo se c'Ë intersezione tra le etichette
            for j=1:length(cOne)
                % ciclo le etichette dentro cOne e se la stessa etichetta Ë
                % dentro cLast allora c'Ë percolazione
                % quindi conto la percolazione
                if sum(cOne(j) == cLast)
                    nPerc = nPerc + 1;
                end
            end

            % controllo per gestire l'assenza di cluster
            if r.d
                % MI SALVO LE INFORMAZIONI UTILI ALL'ANALISI:
                maxCluster = max(r.d);
                numeroSiti = L*L*L; % cubo in 3D

                % 1. dimensione massima cluster / numero siti
                p1 = maxCluster/numeroSiti;

                % 2. dimensione massima cluster / (prob. colorazione * numero
siti)
                p2 = maxCluster/(p*numeroSiti);

                % 3. dimensione massima cluster / somma su tutti i cluster(dim.
                % cluster * # cluster di quella dimensione)
                sommaCluster = 0;
                for w = 1: max(unique(r.d))
                    sommaCluster = sommaCluster + sum(r.d==w)*w;
```

```
                end
                p3 = maxCluster/sommaCluster;

                % 4. RACS = somma su tutti i cluster diversi dal
massimo(dimensione
                % cluster * (dim cluster*# cluster di quella dimensione))
                % / somma su tutti i cluster(dim. cluster * # cluster di quella
dimensione)
                racsNum = 0;
                for w = 1:(max(unique(r.d))-1)
                    racsNum = racsNum + w*(sum(r.d==w)*w);
                end
                racs = racsNum/sommaCluster;

                % mi salvo le info
                p1Perc(i) = p1;
                p2Perc(i) = p2;
                p3Perc(i) = p3;
                racsPerc(i) = racs;
            else
                p1Perc(i) = 0;
                p2Perc(i) = 0;
                p3Perc(i) = 0;
                racsPerc(i) = 0;
            end

            % 5. devo sapere se c'Ë percolazione o no
            if nPerc
                xPerc(i) = 1;
            end
            % 6. mantengo l'informazione di quanti cluster percolanti
            % sono stati creati
            nClusterPerc(i) = nPerc;
        end
        % MI SALVO I DATI PER OGNI P

        % calcoliamo la frequenza di percolare, la sua varianza e il numero
        % di cluster percolanti medio
        MultiPperc(z) = mean(xPerc);
        %MultivarPperc(z) = var(xPerc)/N;
        MultiErrPperc(z) = std(xPerc)/sqrt(N);
        MultinClusterPerc(z) = mean(nClusterPerc);

        % calcoliamo la media e la varianza per p1,p2,p3 e racs e l'errore
        MultimP1Perc(z) = mean(p1Perc);
        %MultivarP1Perc(z) = var(p1Perc)/N;
        MultiErrP1Perc(z) = std(p1Perc)/sqrt(N);

        MultimP2Perc(z) = mean(p2Perc);
        %MultivarP2Perc(z) = var(p2Perc)/N;
        MultiErrP2Perc(z) = std(p2Perc)/sqrt(N);

        MultimP3Perc(z) = mean(p3Perc);
        %MultivarP3Perc(z) = var(p3Perc)/N;
        MultiErrP3Perc(z) = std(p3Perc)/sqrt(N);

        MultimRacsPerc(z) = mean(racsPerc);
        %MultivarRacsPerc(z) = var(racsPerc)/N;
        MultiErrRacsPerc(z) = std(racsPerc)/sqrt(N);
    end

end
```