Laboratorio di Calcolo per Fisici,
I esame appello invernale AA 2019-20

Canale Pb-Z

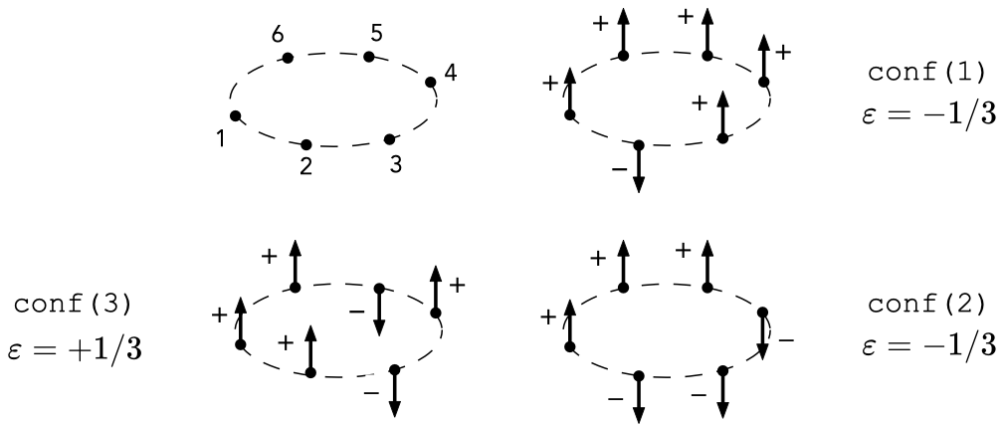
Scopo della prova di esame è lo studio della distribuzione delle energie di un modello di Ising unidimensionale. Per svolgere l'esercitazione avete 3 ore; sono concessi libri di testo e appunti, ma **l'uso di cellulari, laptop e tablet non è ammesso, pena l'annullamento del compito.**
Il programma va scritto e salvato esclusivamente sul computer del laboratorio, utilizzando lo username fornito dal/la docente.
I file vanno salvati all'interno della cartella home; i file `c`, `py` e `png` si chiameranno `nomecognome.c`, `nomecognome.py` e `nomecognome.png` rispettivamente (nello scrivere nome e cognome omettere eventuali caratteri speciali come apostrofi e accenti). Per sicurezza inserite nelle prime righe del file .c tre righe di commento contenenti il vostro nome, cognome e numero di matricola.

► **Esercizio:**

Il *modello di Ising*, nato per il ferromagnetismo nella materia, descrive molti altri fenomeni collettivi, dalle reti neurali alla disposizione degli uccelli in uno stormo. Qui ne vediamo una versione semplificata: una catena unidimensionale fatta di N siti consecutivi $1, 2, ..., N$, chiusa su se stessa in modo tale che dopo il sito N ci sia di nuovo il sito 1. In figura è mostrato il caso $N=6$. Su ciascun sito $i = 1, 2, ..., N$ è definita una variabile chiamata *spin*, che assume solo due valori interi: $S_i = \pm 1$. Se chiamiamo *configurazione* ciascuna delle possibili scelte dei valori degli N spin, è facile convincersi che, per una catena di N siti, esistono in tutto 2^N possibili configurazioni. Delle 64 configurazioni del caso $N=6$, tre esempi sono mostrati in figura (con gli spin $+1$ disegnati come frecce in alto e quelli -1 come frecce in basso). A ciascuna configurazione è associata un'*energia*, che definiamo

$$\varepsilon = -\frac{1}{N} \sum_{\langle i,j \rangle} S_i S_j \,, \tag{1}$$

dove il simbolo $\langle i, j \rangle$ indica che la sommatoria è estesa solo a tutte le possibili coppie i, j di siti primi vicini. Per $N = 6$, ad esempio, $\varepsilon = -(1/6)\,(S_1S_2 + S_2S_3 + S_3S_4 + S_4S_5 + S_5S_6 + S_6S_1)$; alle config-



urazioni mostrate in figura, cioè `conf(1)`$=+1,-1,+1,+1,+1,+1$, `conf(2)`$=+1,-1,-1,-1,+1,+1$, `conf(3)`$=+1,+1,-1,+1,-1,+1$, corrispondono quindi i valori di energia ε indicati vicino a ciascuna.

Si scriva un programma `c` che, per una catena di Ising chiusa su se stessa come quella appena descritta, ma con $N = 12$ siti, valuti prima esattamente e poi statisticamente la distribuzione delle *energie*, e alla fine confronti graficamente, con `python`, le due stime, attraverso i seguenti passaggi:

1. Una direttiva al precompilatore fissa le dimensioni massime degli array che verranno utilizzati in modo che, ad esempio, l'array `conf` possa contenere la configurazione di una catena con $N=12$.

2. Il programma valuta la distribuzione esatta delle *energie* ε con un ciclo che:

   - genera progressivamente ciascuna delle 2^N possibili *configurazioni*, salvandola in un array `conf` di tipo e dimensione opportuna – si veda il *suggerimento* nella pagina successiva.
   - calcola per ciascuna di esse, attraverso una funzione `epsilon` (di tipo e parametri opportuni), l'*energia* cosí come definita dall'equazione (1) della pagina precedente.
   - salva in un opportuno array `energie1` la lista delle 2^N *energie* man mano ottenute
   - le prime 10 volte (cioè per le prime dieci configurazioni) stampa su terminale la configurazione con vicino la corrispondente energia, usando il seguente formato di stampa: `Configurazione n.  8 |1|1|1|-1|-1|1|1|1|1|1|1|1| Eps=-0.66666666`

   e alla fine scrive un file `histo1.dat` che nella prima colonna contiene l'*energia* e nella seconda la sua frequenza (numero di volte che capita quell'*energia* diviso per 2^N).

3. Il programma chiede di inserire un numero $M \in [10^3, 10^4]$ di *configurazioni* da utilizzare ai fini di una stima statistica della distribuzione delle *energie*, controlla che sia compreso tra 1000 e 10000 e, in caso contrario, itera la richiesta.

4. Il programma produce un campione statistico della distribuzione delle *energie*, ripetendo M volte i seguenti passaggi:

   - attraverso una funzione `spinconf`, di tipo e parametri opportuni, genera in modo casuale una configurazione di N spin, ciascuno dei quali con uguale probabilità di valere $+1$ o -1.
   - le prime 10 volte (cioè per le prime dieci configurazioni) stampa su terminale la configurazione con vicino la corrispondente energia, usando lo stesso formato di stampa suggerito al punto 2.
   - attraverso la funzione `epsilon` già definita e usata in precedenza, calcola l'*energia* della configurazione appena generata.
   - salva in un opportuno array `energie2` la lista delle M *energie* man mano ottenute.

   e alla fine scrive un file `histo2.dat` che nella prima colonna contiene l'*energia* e nella seconda la sua frequenza (numero di volte che capita quell'*energia* diviso per M).

5. Eseguire ora il programma `c` sopra descritto, dopodiché, tramite uno script `python`, produrre un unico grafico (dotato di titolo, label degli assi e tutte le didascalie eventualmente necessarie a comprenderne il senso a prima vista), da salvare nel file `nome_cognome.png`, nel quale vengono confrontati i due istogrammi relativi ai dati dei file `histo1.dat` e `histo2.dat` .

Suggerimento: Il seguente blocco di codice `c` genera tutte le configurazioni possibili di una catena di spin unidimensionale con $N = 6$, salvandole in un vettore `conf` di lunghezza $N = 6$:

```
int conf[6]={0.}, iconf=0;
int i1,i2,i3,i4,i5,i6;

for(i1=1;i1<=2;i1++)
{
    for(i2=1;i2<=2;i2++)
    {
         for(i3=1;i3<=2;i3++)
         {
             for(i4=1;i4<=2;i4++)
             {
                 for(i5=1;i5<=2;i5++)
                 {
                     for(i6=1;i6<=2;i6++)
                     {
                     iconf++;
                     conf[0]=pow(-1.0,i1);
                     conf[1]=pow(-1.0,i2);
                     conf[2]=pow(-1.0,i3);
                     conf[3]=pow(-1.0,i4);
                     conf[4]=pow(-1.0,i5);
                     conf[5]=pow(-1.0,i6);
//una volta generato conf qui lo si puo' stampare o si puo' calcolare l'energia.

                     }
                 }
             }
         }
    }
}
```