Ising-modell és a Metropolis-Hastings algoritmus

Tekintsük az előadáson tanulmányozott 1D Ising láncot!

A programfejlesztés során kezdetben egy kis, mégpedig 64 spinből álló rendszerrel dolgozzunk! A rendszer energiáját és a k_BT termikus energiát (hőmérsékletet) J egységekben mérjük, ami azt jelenti, hogy a J helyébe írhatunk 1-et.

Az egyes feladatrészeket egymás után valósítsa meg a beadott program!

A következők legyenek a parancssori argumentumok:

lánc_hossza lépések_száma k_BT_értéke

a) Programozzuk le a Metropolis-Hastings algoritmust, amelyet az előadáson tárgyaltunk. Tegyük ezt az algoritmus külön függvénybe. Ezután futtassuk 10000 lépésen keresztül! Használjunk egy 64 spinből álló Ising láncot nyitott határfeltétellel! Indítsuk a rendszert az előadáson definiált "hideg" kezdőállapotból! Készítsük el az algoritmus által szolgáltatott E_i energiaértékek gyakorisági hisztogramját $k_BT = 1$ és $k_BT = 3$ esetekre! Ehhez írassuk ki egy fájlba a rendszer energiáját minden Metropolis-Hasting lépés után új sorba! Ez persze úgy értendő, hogy ha elfogadtuk az új energiaértéket, akkor azt írjuk ki, ha nem fogadtuk el, akkor az előző lépésből megtartott értéket írjuk ki. Abrázoljuk egy results.ipynb notebookban az értékeket hisztogramon az alábbi módon:(A)

In [ ]:
```
%pylab inline
#------------
x1=loadtxt("file1.dat")
x2=loadtxt("file2.dat")
figsize(8,6)
xlim(-64,0)
hist(x1,32)
hist(x2,32);
```

b) Ismételjük meg az a) pontbeli számolást a "forró" kezdőállapotból indulva! Az ábrázolás alá írjuk le, hogy ugyanazt az eloszlást kaptuk-e! (A)

c) Ábrázoljuk a) részben definiált rendszer "időfejlődését" 256 lépésen keresztül! Ezt a következőképp tegyük:

Minden egyes iterációs lépésben irassuk ki a lánc minden egyes spinjének az értékét egy fájlba! Egy lépéshez tartozó spin értékek egy sorban szerepeljenek, space-vel elválasztva! Ezután ábrázoljuk az eredményeket a results.ipynb-ben! A vízszintes tengely az iterációs lépés számának, a függőleges tengely pedig a spin helyét jelző indexnek feleljen meg! Ezt pl. az alábbi utasításokkal lehet megvalósítani. Próbáljuk ezt ki két különböző hőmérsékletnél, pl. $k_BT = 1$ és $k_BT = 3$.(A)

In [ ]:
```
smx=loadtxt("chain-evolve.dat")
figsize(16,4)
matshow(smx.T);
```

d) Határozzuk meg a rendszer teljes energiájának hőmérsékletfüggését! Ehhez először ábrázoljuk az a) részben kapott energiaértékeket a lépészám függvényében! Láthathjuk, hogy néhányszáz lépés után az energia értéke egy átlag körül fluktuál. Ezt nevezzük termalizációnak.

A program számolja ki az átlagenergiát a termalizáció beállta után (a sok egymás után energiaérték kiíratása nélkül) pl. 2000 itarációra! Ezt ismételjük meg pl. 20 különböző hőmérséklet értékre a $k_BT = 0.1, \dots 4.0$ intervallumban! Ábrázoljuk a teljes energiát a k_BT függvényében a results.ipynb-ben! Hasonlítsuk össze az analitikus számolásból kapható eredménnyel, ami $\langle E \rangle = -NJ \tanh\left(\frac{J}{k_BT}\right)$! (T)