LCPB 23-24 exercise 4 (Restricted Boltzmann Machines, RBMs)

We want to study the performances of an RBM, and, by looking at its learned weights and biases, better understand the correlations in the data (from file **x_RBM_2024_exercise.dat**, $N=10^4$ configurations with L=10 bits). Use an RBM with M=3 hidden units.

1.
Increase the number of contrastive divergence steps from n=1 to n=5.

2.
Compute the log-likelihood $\mathscr{L}$ during the training, at every epoch, or every minibatch update if it reaches a maximum already in the first epoch. Use “t” as an index of this “time”, indicating the unit in the figures.

5.
Try RBMs with different numbers of hidden units: M=1, 2, 3 (done above), 4, 5, and 6.

3.
for M=3, plot $\mathscr{L}$ as a function of “t”, comparing the two contrastive divergence cases (n=1 and n=5). Then, for n=1, plot $\mathscr{L}$ as a function of “t”, comparing the two cases with different M.

4.
From the weights learned by the RBM, guess the structure of the data.

———

To compute $\mathscr{L}$, consider full configurations x=(v,h) and their energy E(x). With L visible units and M hidden units, we have 2^{L+M} possible configurations. The partition function Z is the sum of all their Boltzmann weights $e^{-E(x)}$. The average expectation of the energy according to data is computed by averaging the energy of each data sample v_n completed with all 2^M possible hidden states. Thus, in total there are $N*2^M$ configurations to use in the $\langle E \rangle_{data}$ average.

The package **itertools** is useful for generating the possible configurations.
https://docs.python.org/3/library/itertools.html

```
Q=4
import itertools as it
conf = it.product((0,1), repeat=Q)
all_conf=list(conf)
for x in all_conf:
    print(x)
```