Deep Learning Assignment 2

Flavie VAMPOUILLE – MSc in DS&BA (ESSEC/CENTRALE)

2. Analytic exercise

Given a set of training samples X={x^1, … , x^M}, the log likelihood of X given W is:

$$S(X, W) = log\ \prod_{m=1}^{M} P(x^m; W)$$

$$= \sum_{m=1}^{M} log\ P(x^m; W)$$

$$= \sum_{m=1}^{M} \log\ [\sum_h \frac{1}{Z(W)} \exp\big(-E(x^m, h; W)\big)\ ]$$

$$= \sum_{m=1}^{M} [\ \log\ \sum_h \exp\big(-E(x^m, h; W)\big) - \log\big(Z(W)\big)\ ]$$

where: $Z(W) = \sum_{x,h} \exp[-E(x, h; W)]$

Taking the partial derivative of $\log\big(Z(W)\big)$ with respect to the weight $w_{k,l}$, we obtain:

$$\frac{\partial \log(Z(W))}{\partial w_{k,l}} = \frac{1}{Z(W)} \sum_{x,h} \frac{\partial}{\partial w_{k,l}} \exp[-E(x, h; W)]$$

$$= \frac{1}{Z(W)} \sum_{x,h} \exp[-E(x, h; W)] \, . \frac{\partial\ E(x,h;W)}{\partial w_{k,l}}$$

and $E(x, h; W) = -\frac{1}{2}\ y^T W y$ (with W symmetric), where: $y \equiv (x, h)$

Thus

$$\frac{\partial \log(Z(W))}{\partial w_{k,l}} = \frac{1}{Z(W)} \sum_{x,h} \exp[-E(x, h; W)]\ y_k\ y_l$$

$$= \sum_{x,h} \mathrm{P(x, h; W)}\ .\ y_k\ y_l$$

$$= \ < y_k\ y_l >_{\cdot P(x,h;W)}$$

Thus

$$\sum_{m=1}^{M}\left(-\frac{\partial \log(Z(W))}{\partial w_{k,l}}\right) = -\sum_{m=1}^{M} < y_k\, y_l >_{\cdot P(x,h;W)}$$

<u>Taking the partial derivative of log $\sum_h \exp\left(-E(x^m,h;W)\right)$ with respect to the weight $w_{k,l}$, we obtain:</u>

$$\frac{\partial \log \sum_h \exp\left(-E(x^m,h;W)\right)}{\partial w_{k,l}} = \frac{1}{\sum_h \exp\left(-E(x^m,h;W)\right)} \cdot \frac{\partial \sum_h \exp\left(-E(x^m,h;W)\right)}{\partial w_{k,l}}$$

$$= \frac{1}{\sum_h \exp\left(-E(x^m,h;W)\right)} \cdot \sum_h \frac{\partial \exp\left(-E(x^m,h;W)\right)}{\partial w_{k,l}}$$

$$= \frac{1}{\sum_h \exp\left(-E(x^m,h;W)\right)} \cdot \sum_h \exp[-E(x^m,h;W)]\, \frac{\partial E(x^m,h;W)}{\partial w_{k,l}}$$

$$= \frac{\partial E(x^m,h;W)}{\partial w_{k,l}}$$

$$= y_k^m\, y_l^m \qquad \text{where } y^m \equiv (x^m, \text{h})$$

$$= < y_k\, y_l >_{\cdot P(h;x^m,W)}$$

Thus

$$\sum_{m=1}^{M} \frac{\partial \log \sum_h \exp\left(-E(x^m,h;W)\right)}{\partial w_{k,l}} = \sum_{m=1}^{M} < y_k\, y_l >_{\cdot P(h;x^m,W)}$$

<u>The derivative of the log likelihood is therefore:</u>

$$\frac{\partial S(X,W)}{\partial w_{k,l}} = \sum_{m=1}^{M} \left[< y_k\, y_l >_{\cdot P(h;x^m,W)} - < y_k\, y_l >_{\cdot P(h,x;W)} \right]$$

2. Exact summations

In boltzmann.m

update the partition function

```
%% energy of all network states
ener_all    = -0.5 * sum ( states * W_all .*states , 2 ) ;

%% partition function
Z           = sum ( exp(-ener_all) ) ;

%% probability of any state
P           = exp(-ener_all) / Z ;
```

'awake' phase:

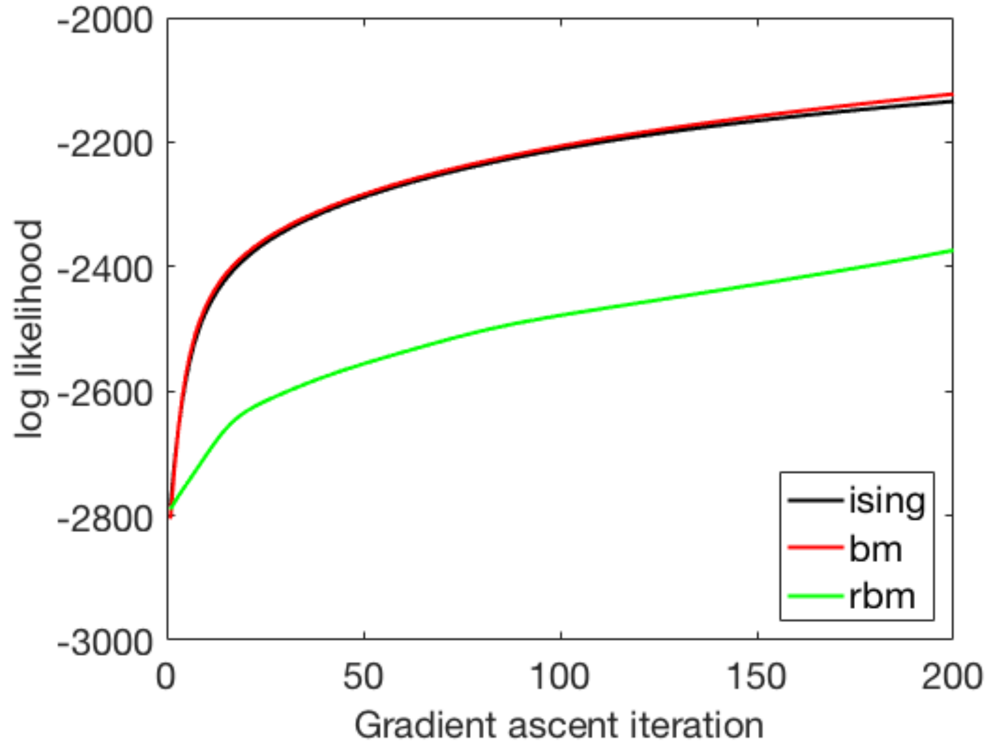
```
case 'ising'

%% energy of observed state
ener_clamped = -0.5 * ( state_o * W_all * state_o' )  ;

%% contribution of m-th sample to 'awake' matrix
E_positive   = ( state_o' * state_o) ;


case {'bm','rbm'}

ener_clamped = diag ( -0.5 * state_clamped * W_all * state_clamped' ) ;

%% P(h|state_o): store as an 2^M vector, M being the #of hidden variables
P_n          = exp(-ener_clamped) / sum ( exp(-ener_clamped) ) ;
```

And

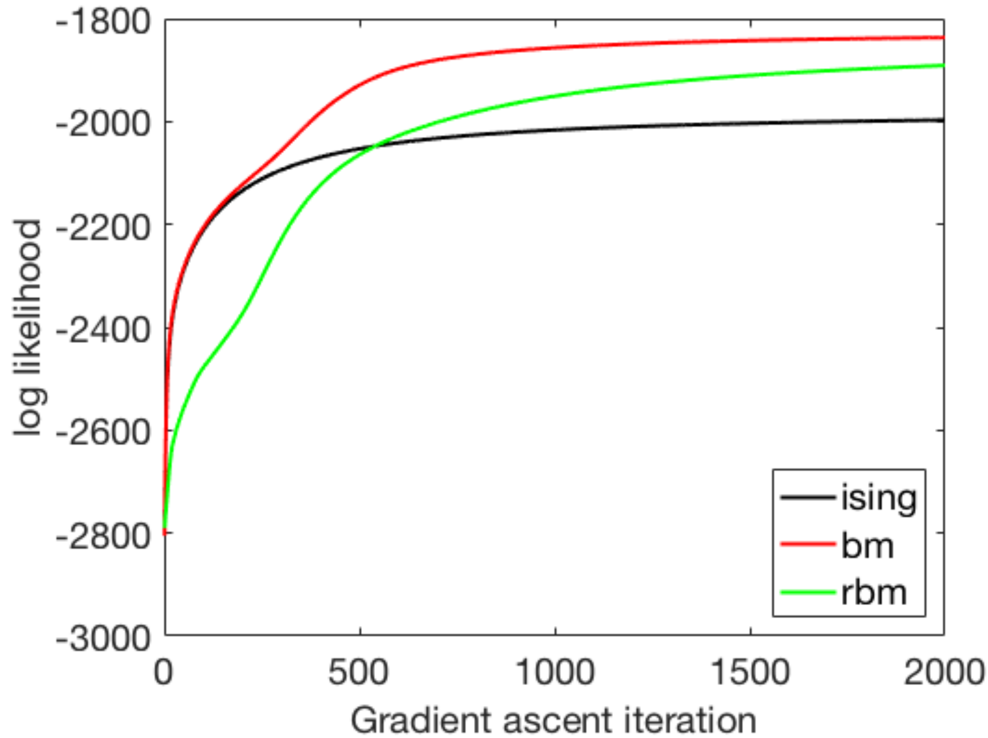
```
Ps(it,m) = sum ( exp(-ener_clamped) ) / Z ;
```

Log likelihood as a function of iterations using brute force estimation of the gradient for the Ising Model, the Boltzmann Machine and the Restricted Boltzmann Machine.

It_max = 200 // n_hidden_wt = 8



It_max = 2000 // n_hidden_wt = 8



Observations

We can see that the log-likelihood estimation is higher for the Boltzmann Machine than for the Ising Model and the Restricted Boltzmann Machine. Ising Model and RBM being subcase of BM this could be expected (BM has more degree of freedom). We can also see that the convergence rate for BM and Ising Model is better than for RBM, but the Ising Model scores the lowest log-likelihood.

2. Block-Gibbs sampling and Contrastive Divergence

The activity rule of Botzmann machine is (with activation a_i):

- Set x_i = +1 with probability $\frac{1}{1+\,e^{-2a_i}}$
- Set x_i = -1 else

Thus, the block_gibbs function can be written as below:

Function block_gibbs.m

```
function [ x , ph ] = block_gibbs ( x0 , W , L )

x = x0 ;

for l = 1:L

    % probability that 'position' of h will be +1 given x
    ph          = 1 ./ ( 1 + exp(-2*x*W) ) ;
    % decide whether to set 'position' of h to +1 or -1
    rand1       = rand(1,length(ph)) ;
    h           = double(ph>rand1) - ( 1 - double(ph>rand1) ) ;

    % probability that 'position' of x will be +1 given h
    px          = 1 ./ ( 1 + exp(2*h*W') ) ;
    % decide whether to set 'position' of x to +1 or -1
    rand2       = rand(1,length(px)) ;
    x           = double(px>rand2) - ( 1 - double(px>rand2) ) ;

end

ph = 1 ./ ( 1 + exp(-2*x*W) ) ;

end
```

In boltzmann.m

‘awake’ phase

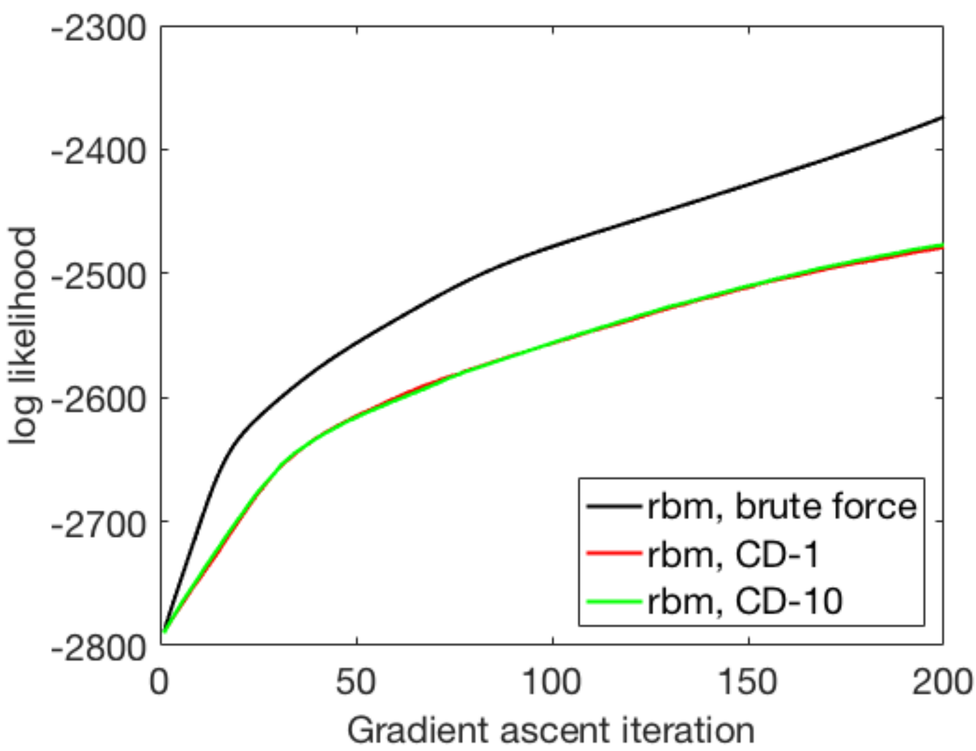
```
case {'rbm-cd-1','rbm-cd-10'}

%% update the observed-to-hidden connections
ener_clamped     = diag ( -0.5 * state_j * W_all * state_j' ) ;

P_n              = exp(-ener_clamped) / Z ;

%% positive phase: P(h|v), [M x 1] vector of posteriors
P_hidden_0   = 1 ./ ( 1 + exp(-2*state_o*W_inter) ) ;

%% 'awake' phase
<h_i v_j>_{P(h|v^i)} = sum_{label_h = -1/1} label_h * P(h_i = label_h|v)
*  v_j
E_positive   = state_o' * (2*P_hidden_0-1) ;
```

‘Dream’ phase with Contrastive Divergence:

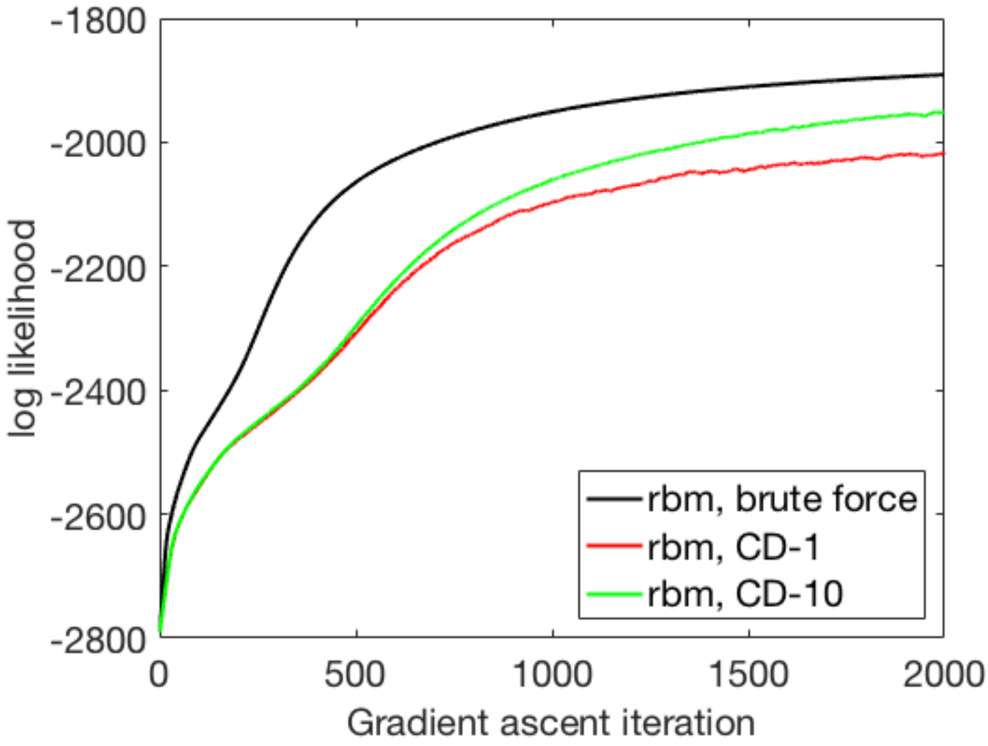
```
E_negative            = obs_K' * (2*P_hidden_K-1) ;


%% brute force summation for 'dream' part
E_dream     = states' * (P .* states) ;
```

Log likelihood as a function of iterations using contrastive divergence for the Restricted Boltzmann machine.

It_max = 200   //   n_hidden_wt = 8



It_max = 2000 // n_hidden_wt = 8



Observations

Note: the graphs obtain are not the same than the ones asked, so there must be a mistake somewhere, but the log-likelihood are nevertheless coherent with what could be expected.

In contrastive divergence, we would like that the Gibbs sampling don't change too much the distribution over the visible variables. We use the block_gibbs function to update the weights and as we could have expected, the log-likelihood for rbm-cd-1 and rbm-cd-10 are smaller than the log-likelihood of the rbm-brute force method (which is the real value) and we obtain a better approximation when the number K increase.