Poetic interpretation of BM learning:

When the BM is "awake", it measures
i.e.,
gets input
from the world
real-world correlations $<x_i x_j>_D$ uses them to adjust the weights
when it is "asleep", it does not adjust the weights - it "dreams" about the world & computes $<x_i x_j>_p$ (i.e., its "idea" of the world). When $<x_i x_j>_D = <x_i x_j>_p$, the two views are balanced.

However, the "world" is represented by just two-point correlations $<x_i x_j>_D$, seems to be too poor to really capture the richness of the world.

For example, consider a "shifter ensemble" of images:

\[
\begin{array}{cccc}
\vdots & \vdots & \vdots & \vdots \\
\hline
1 \rightarrow & 2 \rightarrow & \ldots & L \\
\hline
1 \rightarrow & 2 \rightarrow & \ldots & L \\
\end{array}
\]

Then, away from the boundaries:

\[
\begin{align*}
<x_j x_{j+1}> &= \frac{1}{3} \\
<x_j x_{j+L-1}> &= \frac{1}{3} \\
<x_j x_{j+L+1}> &= \frac{1}{3} \\
\text{all others are } &= 0
\end{align*}
\]
This seems too poor to describe the images \( \Rightarrow \) need higher-order statistics:

\[
P(x) = \frac{1}{Z} e^{-\frac{1}{2} \sum_{ij} w_{ij} x_i x_j + \frac{1}{6} \sum_{ijk} v_{ijk} x_i x_j x_k + \ldots}
\]

higher-order BM

Can get \( \frac{\partial}{\partial w_{ij}} \log Z \), \( \frac{\partial}{\partial v_{ijk}} \log Z \), etc. do gibbs sampling

[ but there are too many parameters ]

In general:

Idea: (due to Hinton & Sejnowski, 1986)
introduce hidden variables to model higher-order correlations.

BM with hidden units [restricted BM]

\[
\tilde{y} = \begin{cases} \tilde{x} & \text{visible nodes state (} M_1 \text{) vector} \\ \tilde{h} & \text{hidden nodes state (} M_2 \text{) vector} \end{cases}
\]

\( M_1 + M_2 \) vector

In particular, when visible nodes are "clamped" at \( \tilde{x}^{(n)} \Rightarrow \tilde{y}^{(n)} = (\tilde{x}^{(n)}, \tilde{h}) \).

Then

\[
P(\tilde{x}^{(n)}) = \sum_{\tilde{h}} P(\tilde{x}^{(n)}, \tilde{h}) = \frac{1}{Z} \sum_{\tilde{h}} e^{-\frac{1}{2} \tilde{y}^{(n)T} W \tilde{y}^{(n)}}
\]

\[
Z = \sum_{\tilde{x}, \tilde{h}} e^{-\frac{1}{2} \tilde{y}^{(n)T} W \tilde{y}^{(n)}}
\]

\( \Rightarrow \) particle function
As before, consider

$$
\frac{\partial}{\partial W_{ij}} \log L = \sum_n \frac{\partial}{\partial W_{ij}} \left\{ \log Z_{\mathbf{x}^{(n)}} - \log Z \right\}
$$

$$
J = \prod_{n=1}^{N} P(\mathbf{x}^{(n)})
$$

$$
= \sum_n \left\{ \frac{1}{Z_{\mathbf{x}^{(n)}}} \sum_{h} y_i^{(n)} y_j^{(n)} e^{\frac{1}{2} \mathbf{y}^{(n)^T W y^{(n)}} - \langle x_i x_j \rangle p(\mathbf{x}, \mathbf{h})} \right\}
$$

as before

$$
\frac{\sum_{n} y_i^{(n)} y_j^{(n)} e^{\frac{1}{2} \mathbf{y}^{(n)^T W y^{(n)}}}}{\sum_{h} e^{\frac{1}{2} \mathbf{y}^{(n)^T W y^{(n)}}}} = \sum_{h} y_i^{(n)} y_j^{(n)} P(h|x^{(n)}) = \sum_{h} y_i^{(n)} y_j^{(n)} P(h|x^{(n)}) = \langle y_i y_j \rangle P(h|x^{(n)})
$$

$$
= \sum_n \left\{ \langle y_i y_j \rangle p(h|x^{(n)}) - \langle y_i y_j \rangle P(\mathbf{h}|\mathbf{x}^{(n)}) \right\}
$$

estimate by unrestricted Gibbs sampling
with \( \mathbf{x}^{(n)} \) fixed
(only hidden spins flipped)
**Application of BM in neural networks (NN)**

Hinton & Salakhutdinov, Science 2006

**Idea:** build a multi-layer NN, pre-train intermediate layers using BMs, then refine the weights by backpropagation.

Consider data that can be represented as binary vectors, e.g. images (0,1) (or vector of spins)

**RBMs**

$$E(\mathbf{v}, \mathbf{h}) = - \sum_i b_i v_i - \sum_{j} b_j h_j - \sum_{i,j} v_i h_j w_{ij}$$

given pixel states,

(1) driven by data

$$h_j = \begin{cases} 1, & \sigma(b_j + \sum_i v_i w_{ij}) \\ 0, & \text{otherwise} \end{cases}$$

(2) "confabulation"

$$v_i = \begin{cases} 1, & \sigma(b_i + \sum_j h_j w_{ij}) \\ 0, & \text{otherwise} \end{cases}$$
(3) \( h_j = \begin{cases} 1, & \sum_i v_i w_{ij} \\ 0, & \text{otherwise} \end{cases} \), driven by confabulation record \( v_i h_j \).

Repeat many times, compute
\( <v_i h_j>_{\text{data}} \) & \( <v_i h_j>_{\text{recon}} \)

Finally, adjust weights:
\[
\Delta w_{ij} = \eta \left( <v_i h_j>_{\text{data}} - <v_i h_j>_{\text{recon}} \right)
\]
learning rate

Iterate to convergence.

Next, make the hidden units the visible units of the next RBM.

Note: \( E(\vec{v}, \vec{h}) = - \sum_i v_i \left[ b_i + \sum_j h_j w_{ij} \right] + \text{const}(\vec{v}) \)

loca-field field for \( v_i \)

Then, for all other spins fixed
\[
P(\bar{v}_i = +1) = \frac{\mathcal{G}(b_i + \sum_j h_j w_{ij})}{\mathcal{G}(b_i + \sum_j h_j w_{ij}) + 1}
\]
\( v_i = 1 \) state \( v_i = 0 \) state

\[
P(\bar{v}_i = 0) = 1 - P(\bar{v}_i = +1)
\]
same as (**)

-5-
Likewise,

\[ E(\vec{w}, h) = -\sum_j h_j \left[ b_j + \sum_i a_i w_{ij} \right] + \text{const}(h) \]

local field

for \( h_j \)

leading to (*)

Finally, the whole architecture:

\[
\begin{align*}
\text{RBM} & \implies \begin{cases} 
2000 \\
\uparrow \vec{w}_1 \\
2000 \\
1000 \\
\uparrow \vec{w}_2 \\
2000 \\
500 \\
\uparrow \vec{w}_3 \\
500 \\
\uparrow \vec{w}_4 \\
1000 \\
\uparrow \vec{w}_5
\end{cases} \\
\text{output} & \implies \begin{cases} 
\text{decoder} \\
\text{encoder} \\
\text{data}
\end{cases}
\end{align*}
\]

Unrolling:

\text{For backpropagation, replace stochastic units with } 6 \text{-units with local fields as activations}

\text{Minimize the error between output & data by backpropagation with conjugate gradients used on 10^3 data vectors at a time.}

-6-