Computational neuroscience

Lecturer: Bert Kappen
Radboud University Nijmegen, Netherlands

September 27, 2017
Course setup

- 9 ec course, until mid January

- Two lecturers: Bert Kappen and Paul Tiesinga, 7 lectures each

- Examination based on computer exercises. One point bonus for weekly exercises

- Instructors: Elisabeth Noordanus

- All course materials (slides, exercises) and schedule via http://www.snn.ru.nl/~bertk/comp_neuroscience/
Neural information processing is noisy

When a neuron is presented repeatedly with the same stimulus, the response of the neuron is not identical.

Monkey cortical area V4. response to 1500 ms flashed grating. Dots show spikes of the neuron. Different lines are different trials, which are each slightly different. Summing the aligned trials produces the poststimulus time histogram (PSTH). The two lower plots illustrate an average firing rate obtained from the raster diagrams using Gaussian smooting with $\sigma$ set to 20 and 200 msec, respectively.

The instantaneous firing rate is reproducible.
An important source of the noise is the unreliable response of synapses. Layer 5 pyramidal neuron from somatosensory cortex of rat. c) shows in repeated trials the response of a single synapse to a regular train of presynaptic spikes. As can be seen, synaptic response is very unreliable. d) shows the same experiment after a period of vigorous paired stimulation of the pre- and post-synaptic cell (b). One sees that as a result of the paired stimulation the reliability of the first synaptic response is greatly enhanced. e) This effect lasts for hours.
This unreliability can in turn be related to the stochastic nature of the channels in the membrane.

![Ionic current across patch of excitable membrane (rat).](image)

Another cause of the unreliable neural firing is that the input to the neuron is not only from the stimulus, but also from active neurons that are connected to it. Since the response of the neuron is noisy, it differs from trial to trial.
Interspike intervals

Instead of measuring the instantaneous firing rate of the neuron, we can also measure the time between subsequent spikes, called the inter-spike intervals (ISIs).

A) Spontaneous activity in the auditory nerve is quite variable between fibers, but all interval histograms can be described by an exponential distribution (left), and show independence of subsequent intervals (right).

B) In the cochlear nucleus, a wide range of interval distributions are found ranging from exponential to nearly Gaussian.

Subsequent ISIs can be correlated or not.
Rate code or spike timing?

Fly H1 neuron detects visual motion

Reaction time around 30 ms
With max 100-200 Hz just handful of spikes

Bialek, Rieke, van Steveninck 1990
Poisson Processes

The Poisson process is the simplest stochastic point process that you can think of. The probability of firing in any short time interval \([t, t + \delta t]\) is constant and equal to \(\lambda \delta t\).

Define \(p_x(t) = \text{Prob}\{\text{cell fires exactly } x\ \text{times in } [0, t]\}\). Then,

\[
p_0(t + \delta t) = p_0(t)(1 - \lambda \delta t)
\]

Since \(p_0\) is a smooth function of \(t\), we can use the Taylor series expansion and write:

\[
p_0(t + \delta t) = p_0(t) + \delta t \frac{\partial p_0(t)}{\partial t}
\]

Combining these last two equations, we obtain \(\frac{\partial p_0(t)}{\partial t} = -\lambda p_0(t)\). This can be easily solved to yield

\[
p_0(t) = \exp(-\lambda t)
\]

since \(p_0(0) = 1\).
We can now repeat this reasoning to compute \( p_x(t) \).

\[
p_x(t + \delta t) = p_x(t)(1 - \lambda \delta t) + p_{x-1}(t)\lambda\delta t
\]

\[
\frac{\partial p_x(t)}{\partial t} = -\lambda p_x(t) + \lambda p_{x-1}(t)
\]

In words, the probability to obtain \( x \) spikes in \([0, t + \delta t]\), is either to obtain \( x \) spikes in \([0, t]\) and no spikes in \([t, t + \delta t]\) or \( x - 1 \) spikes in \([0, t]\) and one spike in \([t, t + \delta t]\). By direct substitution, one can easily verify that the solution is given by

\[
p_x(t) = \frac{(\lambda t)^x}{x!} \exp(-\lambda t)
\]

The expected number of spikes at time \( t \) is

\[
\langle x \rangle_t = \sum_{x=0}^{\infty} xp_x(t) = \lambda t
\]

The fluctuation in the number of spikes is given by

\[
\sigma_t^2 = \langle x^2 \rangle_t - \langle x \rangle_t^2 = \lambda t
\]
Interval distribution

The length of the interspike intervals (ISIs) can be computed for the Poisson process. The probability to observe an ISI of length $t$ is denoted by $I(t)$ and is related to $p_0(t)$ in the following way. \[ I(t)\delta t = \text{Prob}\{\text{no firing in } [0, t]\} \times \text{Prob}\{\text{firing in } [t, t + \delta t]\} = \lambda p_0(t)\delta t. \]

Thus,
\[ I(t) = \lambda \exp(-\lambda t) \]

The mean ISI is given by
\[ \langle t \rangle = \int_0^\infty tI(t)dt = \lambda^{-1} \]

The variance in $I$ is given by
\[ \sigma_t^2 = \langle t^2 \rangle - \langle t \rangle^2 = \lambda^{-2} \]

The 'coefficient of variation' CV is defined as
\[ CV = \frac{\sigma_t}{\langle t \rangle}. \]

$CV = 1$ for Poisson processes.
The coefficient of variation

(left) Firing statistics of neurons in areas V1 (monkey; fixation task; cells were stimulated by flashed bars, Knierim and Van Essen 1992) and MT (monkey; discrimination task of movements in random dot patterns). A,B) Sample spike train C,D) PSTH; E,F) ISIs. (right) A,B) Instantaneous firing rates of V1 and MT cell. PSTHs were segmented according to instantaneous firing rate in 10 sets (0-9). C-E) ISI histograms are computed for different rates (segments 2, 5 and 8 shown).
CV as a function of firing rate. Values between 20-30 msec are underestimated and are probability consistent with CV=1. For low rates, the data are in disagreement with the Poisson ansatz.

We see that the measured CV is typically smaller than 1, and can therefore not be explained by the Poisson process. Simple extensions of the basic Poisson process can however explain these data, as we will show now.
Integration reduces CV

Consider that the neuron receives input, which are independent Poisson spike trains with identical rates $\lambda$, and that it requires $N$ of such input spikes before the neuron fires. The ISI distribution is now given by:

$$I_N(t)\delta t = \text{Prob}\{N-1 \text{ firings in } [0, t]\} \times \text{Prob}\{\text{firing in } [t, t + \delta t]\} = \lambda p_{N-1}(t)\delta t.$$  

Thus,

$$I_N(t) = \frac{\lambda^N t^{N-1}}{(N-1)!} \exp(-\lambda t)$$

which is known as the Gamma distribution.

The mean and variance are given by

$$\langle t \rangle_N = \frac{N}{\lambda} \quad \sigma_t^2 = \frac{N}{\lambda^2} \quad (1)$$

Refractoriness increases

$$\langle t \rangle_N \rightarrow \langle t \rangle_N + t_0$$

and $\sigma_t^2$ is unaltered.
The 'coefficient of variation' CV is therefore

\[ CV = \frac{\sqrt{N}}{N + \lambda t_0} \]

CV as predicted by the above model for different values of \( N (= N_{th}) \) and \( t_0 \).

We see that neural integration (large \( N \)) is in disagreement with the observed large CV. If the neural code uses the (temporal average) firing rate as the carrier of information, then the neuron must integrate incoming activity over time to extract this rate from the incoming spike train. We see that integration implies small CV, whereas the data show rather large CV. Therefore we may conclude that the idea that mean firing rate is the carrier of information is contradicted by the above data.
ISI distributions

For a Poisson process the ISI distribution is given by

\[ I(t) = \lambda \exp(-\lambda t) \]

Consider now the probability density \( I_k(t) \) for the interval between a spike and the \( k \)th that follows it. This is the Gamma distribution as given before:

\[ I_k(t) = \frac{\lambda^k t^{k-1}}{(k-1)!} \exp(-\lambda t) \]

For large \( k \), \( I_k(t) \) tends towards a Gaussian distribution (259-2) (exercise).
Cell R-4-10 has a more or less Gaussian ISI. The convolution of a Gaussian is again a Gaussian. Therefore, $I_k(t)$ remains Gaussian (R-4-10).
For cell 240-1 we observe that the shape of $I_k(t)$ is independent of $k$:

$$I_2(t) = I(t) * I(t) = \int_0^t dt' I(t - t')I(t')$$

$$= \frac{1}{2}I(t/2)$$

Which distribution has the property that, when convoluted with itself, is of the same functional form? the Gaussian distribution, the Gauchy distribution and the so-called stable distribution of order $1/2$.

The distribution of order $1/2$ is defined only on the interval $x \geq 0$ and has one of the longest tails to be found in probability theory. It gives the probability density of first passage times in a one-dimensional random walk.
The size of a PSP is of the order of 0.05-2 mV.

The PSP is a stochastic event: it either happens or it does not with probability anywhere between 0.1 and 0.9 and depends also on recent pre- and post-synaptic cell activity.

The effect of synaptic input on neuron output requires compartmental modeling of the geometry of the cell. The overall picture that emerges is that the local PSP at the synapse propagates to the cell body with a delay of 1-2 msec and shows a temporal dispersion of about 10 msec, strongly reducing high frequency components above 50 Hz.
First passage times

The first passage time distribution can be formulated as a random walk problem, which we here describe in terms of a simplified neuron model. This model mimics the linear and threshold behaviour of real neurons.

1. Let the electrical state of the neuron be specified by a single number, the membrane potential $v$.

2. Choose a particular point ($v = 0$) as the resting potential.

3. Assume that each incoming EPSP increments $v$ by one and each IPSP decreases $v$ by one. These presynaptic events are random and independent:

\[ v_{t+1} = v_t + \xi \]

where, $\xi = \pm 1$ is a random variable with mean value $\mu$ and variance $\sigma^2$.

4. Choose another point $v = v_{th}$ as the threshold. Whenever $v$ reaches the threshold, a spike is emitted and $v$ is reset to 0.
**Diffusion**

First consider that there is no threshold, and we wish to compute the probability $p(t, v|t_0, v_0)$, which is the probability that at time $t$ the membrane potential is at $v$, given that at time $t_0$ the value was $v_0$.

Since $v_t$ is a sum of $t - t_0$ independent contributions, $p(t, v|t_0, v_0)$ is Gaussian:

$$\langle v_{t+1} \rangle = \langle v_t \rangle + \mu, \quad \langle v \rangle_t = v_0 + \mu(t - t_0)$$

We compute the time dependence of the variance:

$$\sigma^2_{t+1} = \sigma^2_t + \sigma^2, \quad \sigma^2_t = \sigma^2(t - t_0)$$

Thus the diffusion process is described by a Gaussian distribution with time dependent mean and variance:

$$p(t, v|t_0, v_0) = \frac{1}{\sqrt{2\pi(t - t_0)\sigma}} \exp \left( - \frac{(v - \langle v \rangle_t)^2}{2\sigma^2(t - t_0)} \right)$$
First passage time distribution

We can now compute $\rho(t)$: the probability distribution of the ISIs for the first passage time problem. We make use of the following identity:

$$p(t, v_{th}|0, 0) = \int_0^t dt' p(t, v_{th}|t', v_{th}) \rho(t')$$

$\rho(t')$ is the probability to arrive after time $t'$ for the first time to $v_{th}$, starting at time $t = 0$ at $v = 0$.

The equation states that $\rho(t)$ is related to the diffusion distribution $p(t, v_{th}|0, 0)$ by first moving in time $t'$ from 0 to $v_{th}$ (described by $\rho(t')$) and then at time $t$ visiting the threshold again (described by $p(t, v_{th}|t', v_{th})$).
First passage time distribution

The solution is given by

\[ \rho(t) = \frac{v_{th}}{\sqrt{2\pi}\sigma t^{3/2}} \exp \left( -\frac{(v_{th} - \mu t)^2}{2\sigma^2 t} \right) \]
Laplace transformation

If $f(t)$ is a function of $t$, then the Laplace transform of $f$, denoted by $\hat{f}$ is given by

$$
\hat{f}(s) = \int_0^\infty f(t) \exp(-st)dt
$$

The Laplace transform is only defined for $\text{Re } s > 0$. The inverse transformation is given by

$$
f(t) = \frac{1}{2\pi i} \int_{c-i\infty}^{c+i\infty} ds \hat{f}(s) \exp(st)
$$

with $c$ some positive real number.

The Laplace transformation can be useful to solve integral equations of certain type. For instance, if

$$
f_1(t) = \int_0^t dt' f_2(t-t') f_3(t'),
$$

then

$$
\hat{f}_1(s) = \hat{f}_2(s) \hat{f}_3(s)
$$

Some useful Laplace relations are:
We use the Laplace transformation to solve the first passage time problem for \( \mu = 0 \). We identify

\[
\begin{align*}
f_1(t) &= p(t, v_{\text{th}}|0, 0) = \frac{1}{\sqrt{2\pi t} \sigma} \exp \left( -\frac{v_{\text{th}}^2}{2\sigma^2 t} \right) \quad \hat{f}_1(s) = \frac{1}{\sqrt{2s} \sigma} \exp \left( -\sqrt{2s} \frac{v_{\text{th}}}{\sigma} \right) \\
f_2(t - t') &= p(t, v_{\text{th}}|t', v_{\text{th}}) = \frac{1}{\sqrt{2\pi(t - t')} \sigma} \quad \hat{f}_2(s) = \frac{1}{\sqrt{2s} \sigma} \\
f_3(t') &= \rho(t')
\end{align*}
\]
Thus,

\[ \hat{\rho}(s) = \frac{\hat{f}_1(s)}{\hat{f}_2(s)} = \exp \left( -\sqrt{2s} \frac{v_{th}}{\sigma} \right) \]

\[ \rho(t) = \frac{v_{th}}{\sqrt{2\pi\sigma t^{3/2}}} \exp \left( -\frac{v_{th}^2}{2\sigma^2 t} \right) \]
Scale invariance

It is easy to see that without drift ($\mu = 0$), $\rho(t)$ is invariant under convolution with itself. The sum of two subsequent intervals is distribution as

$$\rho_2(t) = \int_0^t dt' \rho(t - t')\rho(t') = \frac{1}{4}\rho(t/4)$$

(See handouts_cns) So $\rho_2(t)$ has the same shape as $\rho(t)$. However, note that the scaling factor 4 is different from the scaling factor 2 that we observe experimentally.

NB: For large $k$, $\rho_k$ does not become Gaussian as observed experimentally.
Approximate scale invariance.

When $\mu \neq 0$, the first passage time distribution is unfortunately no longer invariant under convolution with itself. However, it is easy to show that for a certain range of parameters there is approximate shape invariance with $2\times$ expansion of the time scale.

Simulated ISIs from first passage time distribution with drift $\mu = 1/16$ and threshold $v_{\text{th}} = 32$ shows approximate invariance under convolution with $2\times$ expansion of the time scale.
Fit to data

The first passage time distribution can be fitted to the data and has two free parameters:

$$\rho(t) \propto \frac{1}{t^{3/2}} \exp \left( -\frac{a}{t} - bt \right)$$

and fit great variety of neuron data.
The integrate and fire model improves the random walk model, by introducing a 'leak' term in the dynamics that drives the membrane potential back to its resting value. For small currents the response is well described by a linear differential equation of the form

\[ C \frac{dV}{dt} = -\frac{V}{R} + I(t) \]

When \( V > V_{th} \) a spike is generated and \( V \to V_{rest} = 0 \).
For constant current \((I(t) = I_0, t > 0, V(0) = 0)\) the differential equation in the absence of a threshold can be easily solved:

\[
V(t) = I_0 R \left(1 - e^{-t/\tau}\right), \quad \tau = RC
\]

Thus, for large \(t\) the membrane potential saturates at a value \(V(\infty) = I_0 R\). If \(V(\infty) > V_{th}\) a spike is produced at time \(t^*\) which satisfies

\[
V_{th} = I_0 R \left(1 - e^{-t^*/\tau}\right),
\]

or

\[
t^* = \tau_{ref} - \tau \log \left(1 - \frac{V_{th}}{I_0 R}\right)
\]
Summary

1. Neuron spiking is noisy and requires a statistical description.
   
   (a) If many excitatory pre-synaptic events contribute to a spike (integration) one expects a low coefficient of variation, which is not observed experimentally (Poisson process, Gamma distribution)
   
   (b) If both excitatory and inhibitory events contribute, the spike timing shows large variance (first passage time distribution).
   
   (c) Most observed ISI distributions can be modeled with a Poisson, Gaussian or first passage time distribution. This assumes independent subsequent ISIs.
   
   (d) When non-stationary input is presented, the distribution can be rather more complicated (multi-modal). See Gerstein 1964.

2. We have introduced two simplified neuron models: FPT and IF. The first can be solved analytically, the second not. Both are approximate at best:
   
   (a) Real neurons have a complex geometry, and the spike generation process can initiate at various loci.
   
   (b) We ignore the fact that the size of the PSP depends on the membrane potential ($I = g(V)(V - V_{rev})$).
A very simple model that captures the input output relation is to state that in each small but finite time interval $\Delta t$ the neuron can either emit one spike or no spike ($y = 1, 0$) and the probability of $y = 1$ is a non-linear function of the input current $I$:

$$p(y = 1) = \sigma(I), \quad \sigma(x) = \frac{1}{2}(1 + \tanh(x)).$$

The maximal firing frequency is $1/\Delta t$. 
In a network, the input to neuron $i$

$$I_i(t) = \sum_{j \neq i} I_{ij} y_j(t) + \Theta_i$$

$\sigma(\Theta_i)$ is equal to the firing rate of the neuron in the absence of any input.

$t$ labels the discretized time in units of $\Delta t$.

$$p(y_i = 1, t + 1|y(t)) = \sigma(\sum_{j \neq i} I_{ij} y_j(t) + \Theta_i)$$
McCulloch-Pitts neurons

The McCulloch-Pitts neuron model. When the total input activity exceeds a threshold the output of the neuron is 1, otherwise it is 0.

\[
\lim_{\beta \to \infty} \sigma(\beta h_i(y)) = \Theta(h_i(y))
\]
Logical circuits:

\[ y = \Theta(y_1 + y_2 - \theta) \]

\( \theta = 3/2 \): logical AND
\( \theta = 1/2 \): logical OR

McCulloch-Pitts neurons can compute any logical function.
The Perceptron

Relevant in history of pattern recognition and neural networks.

- Perceptron learning rule + convergence, Rosenblatt (1962)
- Perceptron critique (Minsky and Papert, 1969) → "Dark ages of neural networks"
- Revival in the 80’s: Backpropagation
The Perceptron

\[ y(x) = \text{sign}(w^T \phi(x)) \]

where

\[ \text{sign}(a) = \begin{cases} 
+1, & a \geq 0 \\
-1, & a < 0.
\end{cases} \]

and \( \phi(x) \) is a feature vector (e.g. hard wired neural network).

---

**Figure 3.10.** The perceptron network used a fixed set of processing elements, denoted \( \phi_j \), followed by a layer of adaptive weights \( w_j \) and a threshold activation function \( g(\cdot) \). The processing elements \( \phi_j \) typically also had threshold activation functions, and took inputs from a randomly chosen subset of the pixels of the input image.
The Perceptron

Ignore $\phi$, ie. consider inputs $x^\mu$ and outputs $t^\mu = \pm 1$
Define $w^T x = \sum_{j=1}^{n} w_j x_j + w_0$. Then, the learning condition becomes

$$\text{sign}(w^T x^\mu) = t^\mu, \quad \mu = 1, \ldots, P$$

We have

$$\text{sign}(w^T x^\mu t^\mu) = 1 \quad \text{or} \quad w^T z^\mu > 0$$

with $z_j^\mu = x_j^\mu t^\mu$. 
**Linear separation**

Classification depends on sign of $w^T x$. Thus, decision boundary is hyper plane:

$$0 = w^T x = \sum_{j=1}^{n} w_j x_j + w_0$$

Perceptron can solve linearly separable problems.

AND problem is linearly separable.

XOR problem and linearly dependent inputs not linearly separable.
Perceptron learning rule

Learning successful when
\[ w^T z^\mu > 0, \text{ all patterns} \]

Learning rule is 'Hebbian':

\[
\begin{align*}
  w_j^{\text{new}} &= w_j^{\text{old}} + \Delta w_j \\
  \Delta w_j &= \eta \Theta(-w^T z^\mu) x_j^\mu t^\mu = \eta \Theta(-w^T z^\mu) z_j^\mu
\end{align*}
\]

\(\eta\) is the learning rate.

Depending on the data, there may be many or few solutions to the learning problem (or none at all)
The quality of the solution is determined by the worst pattern. Since the solution does not depend on the size of $w$:

$$D(w) = \frac{1}{|w|} \min_{\mu} w^T z^\mu$$

Acceptable solutions have $D(w) > 0$.

The best solution is given by $D_{\text{max}} = \max_w D(w)$. 
$D_{\text{max}} > 0$ iff the problem is linearly separable.
Convergence of Perceptron rule

Assume that the problem is linearly separable, so that there is a solution $w^*$ with $D(w^*) > 0$.

At each iteration, $w$ is updated only if $w \cdot z^\mu < 0$. Let $M^\mu$ denote the number of times pattern $\mu$ has been used to update $w$. Thus,

$$w = \eta \sum_{\mu} M^\mu z^\mu$$

Consider the quantity

$$-1 < \frac{w \cdot w^*}{|w||w^*|} < 1$$

We will show that

$$\frac{w \cdot w^*}{|w||w^*|} \geq \mathcal{O}(\sqrt{M}),$$

with $M = \sum_{\mu} M^\mu$ the total number of iterations.

Therefore, $M$ can not grow indefinitely. Thus, the perceptron learning rule converges in a finite number of steps when the problem is linearly separable.
Proof:

\[ w \cdot w^* = \eta \sum_{\mu} M^\mu z^\mu \cdot w^* \geq \eta M \min_{\mu} z^\mu \cdot w^* \]

\[ = \eta M D(w^*) |w^*| \]

\[ \Delta |w|^2 = |w + \eta z^\mu|^2 - |w|^2 = 2\eta w \cdot z^\mu + \eta^2 |z^\mu|^2 \]

\[ \leq \eta^2 |z^\mu|^2 = \eta^2 N \]

\[ |w| \leq \eta \sqrt{NM} \]

Thus,

\[ 1 \geq \frac{w \cdot w^*}{|w||w^*|} \geq \sqrt{M} \frac{D(w^*)}{\sqrt{N}} \]

Number of weight updates:

\[ M \leq \frac{N}{D^2(w^*)} \]
Capacity of the Perceptron

Consider $P$ patterns in $N$ dimensions in general position:
- no subset of size less than $N$ is linearly dependent.
- general position is necessary for linear separability

Question: What is the probability that a problem of $P$ samples in $N$ dimensions is linearly separable?
Define $C(P, N)$ the number of linearly separable colorings on $P$ points in $N$ dimensions, with separability plane through the origin. Then (Cover 1966):

$$C(P, N) = 2 \sum_{i=0}^{N-1} \binom{P - 1}{i}$$

When $P \leq N$ small, then $C(P, N) = 2 \sum_{i=0}^{P-1} \binom{P - 1}{i} = 2(1 + 1)^{P-1} = 2^P$

When $P = 2N$, then 50 % is linearly separable: $C(P, N) = 2 \sum_{i=0}^{N-1} \binom{2N - 1}{i} = \sum_{i=0}^{2N-1} \binom{2N - 1}{i} = 2^{2N-1} = 2^{P-1}$
Proof by induction.

Add one point $X$. The set $C(P, N)$ consists of
- colorings with separator through $X$ (A)
- rest (B)

Thus,

\[ C(P + 1, N) = 2A + B = C(P, N) + A \]
\[ = C(P, N) + C(P, N - 1) \]

Yields

\[ C(P, N) = 2 \sum_{i=0}^{N-1} \binom{P - 1}{i} \]
Input-output behaviour:

**Linear units**

\[ y^\mu = w_0 + \sum_{j=1}^{n} w_j x_j^\mu = \sum_{j=0}^{n} w_j x_j \quad x_0 = 1 \]

Desired behaviour: \( y^\mu = t^\mu, \mu = 1, \ldots, P. \)

Define a learning rule as gradient descent on a cost function:

\[ E(w) = \frac{1}{2} \sum_{\mu=1}^{P} \left( t^\mu - \sum_{j=0}^{n} w_j x_j^\mu \right)^2 \]

\[ \Delta w_i = -\eta \frac{\partial E}{\partial w_i} = \eta \sum_{\mu} \left( t^\mu - \sum_{j} w_j x_j^\mu \right) x_i^\mu \]

\[ w_i = w_i + \Delta w_i \]
Parameter optimization

$E(w)$ is minimal when $\nabla E(w) = 0$, but not vice versa!

As a consequence, gradient based methods find a local minimum, not necessarily the global minimum.
Gradient descent optimization

The simplest procedure to optimize $E$ is to start with a random $w$ and iterate

$$w^\tau + 1 = w^\tau - \eta \nabla E(w^\tau)$$

This is called batch learning, where all training data are included in the computation of $\nabla E$.

Does this algorithm converge? Yes, if $\epsilon$ is "sufficiently small" and $E$ bounded from below.

Proof: Denote $\Delta w = -\eta \nabla E$.

$$E(w + \Delta w) \approx E(w) + (\Delta w)^T \nabla E = E(w) - \eta \sum_i \left( \frac{\partial E}{\partial w_i} \right)^2 \leq E(w)$$

In each gradient descent step the value of $E$ is lowered. Since $E$ bounded from below, the procedure must converge asymptotically.
Convergence of gradient descent in a quadratic well

\[ E(w) = \frac{1}{2} \sum_i \lambda_i w_i^2 \]

\[ \Delta w_i = -\eta \frac{\partial E}{\partial w_i} = -\eta \lambda_i w_i \]

\[ w_i^{\text{new}} = w_i^{\text{old}} + \Delta w_i = (1 - \eta \lambda_i) w_i \]

Convergence when \(|1 - \eta \lambda_i| < 1\). Oscillations when \(1 - \eta \lambda_i < 0\).

Optimal learning parameter depends on curvature of each dimension.
Non-linear units/logistic regression

When the transfer function is non-linear,

\[ y^\mu = g(h^\mu), \quad h^\mu = \sum_j w_j x_j^\mu \]

the cost criterion for learning becomes

\[ E(w) = \frac{1}{2} \sum_\mu (t^\mu - y^\mu)^2 = \frac{1}{2} \sum_\mu \left( t^\mu - g \left( \sum_j w_j x_j^\mu \right) \right)^2 \]

\[ \Delta w_i = -\eta \frac{\partial E}{\partial w_i} = \sum_\mu (t^\mu - y^\mu) g'(h^\mu)x_i^\mu \]
Non-linear units/logistic regression

For $y = \pm 1$ we define

$$p(y|x) = \sigma(hy) \quad h = \sum_j w_j x_j$$

$$\sigma(x) = (1 + e^{-2x})^{-1} = \frac{1}{2}(1 + \tanh(x)).$$

Note that $p(y = -1|x) = \sigma(-h) = 1 - \sigma(h) = 1 - p(y = 1|x)$.

When the target distribution is given by $q(y|x^\mu), \mu = 1, \ldots, N$, we use the KL divergence as cost function for learning:

$$E = \sum_\mu \sum_{y=\pm 1} q(y|x^\mu) \log \left( \frac{q(y|x^\mu)}{p(y|x^\mu)} \right) = -\sum_\mu \sum_{y=\pm 1} q(y|x^\mu) \log p(y|x^\mu) + const.$$
Gradient descent on this learning rule yields $^1$ $^2$

$$\frac{\partial p(t^\mu|x^\mu)}{\partial w_j} = 2p(t^\mu|x^\mu)p(-t^\mu|x^\mu)t^\mu x^\mu_j$$

$$\frac{\partial E}{\partial w_j} = -2 \sum_{\mu} p(-t^\mu|x^\mu)t^\mu x^\mu_j$$

$$= - \sum_{\mu} (t^\mu - \langle y^\mu \rangle)x^\mu_j$$

This is equivalent with HKP (5.57-58).

$^1$

$$\frac{d}{dx} \sigma(x) = 2(1 + e^{-2x})^{-2}e^{-2x} = 2\sigma(x) \frac{e^{-2x}}{1 + e^{-2x}} = 2\sigma(x)\sigma(-x)$$

$^2$

$$\langle y^\mu \rangle \equiv \sum_{y=\pm 1} yp(y|x^\mu) = \sum_{y=\pm t^\mu} yp(y|x^\mu) = t^\mu p(t^\mu|x^\mu) - t^\mu p(-t^\mu|x^\mu) = t^\mu (1 - 2p(-t^\mu|x^\mu))$$

$$\langle y^\mu \rangle - t^\mu = -2t^\mu p(-t^\mu|x^\mu)$$
Network training

Regression: $t_n$ continue valued, $h_2(x) = x$ and one usually minimizes the squared error (one output)

$$E(w) = \frac{1}{2} \sum_{n=1}^{N} (y(x_n, w) - t_n)^2$$

$$= - \log \prod_{n=1}^{N} \mathcal{N}(t_n | y(x_n, w), \beta^{-1}) + \ldots$$

Classification: $t_n = 0, 1$, $h_2(x) = \sigma(x)$, $y(x_n, w)$ is probability to belong to class 1.

$$E(w) = - \sum_{n=1}^{N} \{t_n \log y(x_n, w) + (1 - t_n) \log(1 - y(x_n, w))\}$$

$$= - \log \prod_{n=1}^{N} y(x_n, w)^{t_n}(1 - y(x_n, w))^{1-t_n}$$
Network training

More than two classes: consider network with $K$ outputs. $t_{nk} = 1$ if $x_n$ belongs to class $k$ and zero otherwise. $y_k(x_n, w)$ is the network output

$$E(w) = - \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \log p_k(x_n, w)$$

$$p_k(x, w) = \frac{\exp(y_k(x, w))}{\sum_{k'=1}^{K} \exp(y_{k'}(x, w))}$$
Parameter optimization

$E(w)$ is minimal when $\nabla E(w) = 0$, but not vice versa!

As a consequence, gradient based methods find a local minimum, not necessary the global minimum.
Gradient descent optimization

The simplest procedure to optimize $E$ is to start with a random $\mathbf{w}$ and iterate

$$
\mathbf{w}^{\tau+1} = \mathbf{w}^{\tau} - \eta \nabla E(\mathbf{w}^{\tau})
$$

This is called batch learning, where all training data are included in the computation of $\nabla E$.

Does this algorithm converge? Yes, if $\epsilon$ is "sufficiently small" and $E$ bounded from below.

Proof: Denote $\Delta \mathbf{w} = -\eta \nabla E$.

$$
E(\mathbf{w} + \Delta \mathbf{w}) \approx E(\mathbf{w}) + (\Delta \mathbf{w})^T \nabla E = E(\mathbf{w}) - \eta \sum_i \left( \frac{\partial E}{\partial w_i} \right)^2 \leq E(\mathbf{w})
$$

In each gradient descent step the value of $E$ is lowered. Since $E$ bounded from below, the procedure must converge asymptotically.
Convergence of gradient descent in a quadratic well

\[ E(w) = \frac{1}{2} \sum_i \lambda_i w_i^2 \]

\[ \Delta w_i = -\eta \frac{\partial E}{\partial w_i} = -\eta \lambda_i w_i \]

\[ w_i^{\text{new}} = w_i^{\text{old}} + \Delta w_i = (1 - \eta \lambda_i) w_i \]

Convergence when \(|1 - \eta \lambda_i| < 1\). Oscillations when \(1 - \eta \lambda_i < 0\).

Optimal learning parameter depends on curvature of each dimension.
Learning with momentum

One solution is adding momentum term:

\[
\Delta w_{t+1} = \Delta w_t - \eta \nabla E(w_t) + \alpha \Delta w_t \\
= \Delta w_t - \eta \nabla E(w_t) + \alpha (-\eta \nabla E(w_{t-1}) + \alpha (-\eta \nabla E(w_{t-2}) + \ldots)) \\
= \eta \sum_{k=0}^{t} \alpha^k \nabla E(w_{t-k})
\]

Consider two extremes:

**No oscillations** all derivative are equal:

\[
\Delta w_{t+1} \approx -\eta \nabla E \sum_{k=0}^{t} \alpha^k = -\eta \frac{\partial E}{1 - \alpha\partial w}
\]

results in acceleration
**Oscillations** all derivatives are equal but have opposite sign:

\[
\Delta w(t + 1) \approx -\eta \nabla E \sum_{k=0}^{t} (-\alpha)^k = -\frac{\eta}{1 + \alpha \partial w} \frac{\partial E}{\partial w}
\]

results in deceleration
Newton's method

One can also use Hessian information for optimization. As an example, consider a quadratic approximation to $E$ around $w_0$:

$$E(w) = E(w_0) + b^T(w - w_0) + \frac{1}{2}(w - w_0)H(w - w_0)$$

$$b_i = \frac{\partial E(w_0)}{\partial w_i} \quad H_{ij} = \frac{\partial^2 E(w_0)}{\partial w_i \partial w_j}$$

$$\nabla E(w) = b + H(w - w_0)$$

We can solve $\nabla E(w) = 0$ and obtain

$$w = w_0 - H^{-1}\nabla E(w_0)$$

This is called Newton's method.

Quadratic approximation is exact when $E$ is quadratic, so convergence in one step.
Line search

Another solution is line optimisation:

$$w_1 = w_0 + \lambda d_0, \quad d_0 = \nabla E(w_0)$$

$\lambda$ is found by a one dimensional optimisation

$$0 = \frac{\partial}{\partial \lambda} E(w_0 + \lambda d_0) = d_0 \cdot \nabla E(w_1) = d_0 \cdot d_1$$

Therefore, subsequent search directions are orthogonal.


Conjugate gradient descent

We choose as new direction a combination of the gradient and the old direction

\[ d'_1 = \nabla E(w_1) + \beta d_0 \]

Line optimisation \( w_2 = w_1 + \lambda d'_1 \) yields \( \lambda \) such that \( d'_1 \cdot \nabla E(w_2) = 0 \).

The direction \( d'_1 \) is found by demanding that \( \nabla E(w_2) \approx 0 \) also in the 'old' direction \( d_0 \):

\[
0 = d_0 \cdot \nabla E(w_2) \approx d_0 \cdot (\nabla E(w_1) + \lambda H(w_1)d'_1)
\]

or

\[
d_0 H(w_1)d'_1 = 0
\]

\( d_0, d'_1 \) are said to be conjugate.
Polak-Ribiere rule

The conjugate directions can be computed without computing the Hessian matrix, for instance using the Polak-Ribiere rule:\(^3\)

\[
\beta = \frac{(\nabla E(w_1) - \nabla E(w_0)) \cdot \nabla E(w_1)}{\|\nabla E(w_0)\|^2}
\]

For quadratic problems, it can be proven that this rule keeps the last \(n\) directions all mutually conjugate [Press et al., 1996]

\[
d_i^T H d_j = 0 \quad i, j = 1, \ldots, n
\]

---

\(^3\)We need \(0 = d_0^T H(w_1) d_1'\). We use \(\nabla E(w_0) \approx \nabla E(w_1) + (w_0 - w_1)^T H(w_1) = \nabla E(w_1) - d_0^T H(w_1)\) and \(d_1' = \nabla E(w_1) + \beta d_0\). Then

\[
0 = d_0^T H(w_1) d_1' = (\nabla E(w_1) - \nabla E(w_0)) \cdot (\nabla E(w_1) + \beta d_0) = (\nabla E(w_1) - \nabla E(w_0)) \cdot \nabla E(w_1) - \beta \|\nabla E(w_0)\|^2
\]
Stochastic gradient descent

One can also consider on-line learning, where only one or a subset of training patterns is considered for computing $\nabla E$.

$$E(w) = \sum_n E_n(w) \quad w_{t+1} = w_t - \alpha_t \nabla E_n(w^\tau)$$

May be efficient for large data sets. This results in a stochastic dynamics in $w$ that can help to escape local minima.
Robbins Monro

Method of *stochastic approximation* originally due to Robbins and Monro 1951:
- Solve $M(x) = a$ with $M(x) = \langle N(x, \xi) \rangle$.
- Iterate $x_{t+1} = x_t + \alpha_t(a - N(x, \xi))$
- Convergence requires

$$\sum_t \alpha_t = \infty \quad \sum_t \alpha_t^2 < \infty$$

For instance $\alpha_t = 1/t$.

Application to stochastic gradient descent:
- $\nabla E(w) = 0$ with $\nabla E(w) = \sum_n \nabla E_n(w)$
- Iterate $w_{t+1} = w_t - \eta_t \nabla E_n(w)$

Extensions of SGD and comparisons see [Sohl-Dickstein et al., 2013].
Feed-forward Network functions

We extend the previous regression model with fixed basis functions

\[ y(x, w) = f \left( \sum_{j=1}^{M} w_j \phi_j(x) \right) \]

to a model where \( \phi_j \) is adaptive:

\[ \phi_j(x) = h(\sum_{i=0}^{D} w_{ji}^{(1)} x_i) \]
Feed-forward Network functions

In the case of $K$ outputs

$$y_k(x, w) = h_2 \left( \sum_{j=1}^{M} w^{(2)}_{kj} h_1 \left( \sum_{i=0}^{D} w^{(1)}_{ji} x_i \right) \right)$$

$h_2(x)$ is $\sigma(x)$ or $x$ depending on the problem. $h_1(x)$ is $\sigma(x)$ or $\tanh(x)$.

Left) Two layer architecture. Right) general feed-forward network with skip-layer connections.

If $h_1, h_2$ linear, the model is linear. If $M < D, K$ it computes principle components (Bishop section 12.4.2).
Feed-forward Network functions

Two layer NN with 3 'tanh' hidden units and linear output can approximate many functions. \( x \in [-1, 1] \), 50 equally spaced points. From left to right: \( f(x) = x^2, \sin(x), |x|, \Theta(x) \). Dashed lines are outputs of the 3 hidden units.

Two layer NN with two inputs and 2 'tanh' hidden units and sigmoid output for classification. Dashed lines are hidden unit activities.

Feed-forward neural networks have good approximation properties.
**Weight space symmetries**

For any solutions of the weights, there are many equivalent solutions due to symmetry:
- for any hidden unit $j$ with $\tanh$ activation function, change $w_{ji} \rightarrow -w_{ji}$ and $w_{kj} \rightarrow -w_{kj}$: $2^M$ solutions
- rename the hidden unit labels: $M!$ solutions

Thus a total of $M!2^M$ equivalent solutions, not only for $\tanh$ activation functions.
5.3.1

Error backpropagation

Error is sum of error per pattern

\[ E(w) = \sum_n E^n(w) \quad \text{with} \quad E^n(w) = \frac{1}{2} \| y(x_n, w) - t_n \|^2 \]

\[ y_k(x, w) = h_2 \left( w_{k0} + \sum_{j=1}^M w_{kj} h_1 \left( w_{j0} + \sum_{i=1}^D w_{ji} x_i \right) \right) \]

\[ = h_2(a_k) \]

\[ a_k = w_{k0} + \sum_{j=1}^M w_{kj} h_1(a_j) = \sum_{j=0}^M w_{kj} h_1(a_j) \quad h_1(a_0) = 1 \]

\[ a_j = w_{j0} + \sum_{i=1}^D w_{ji} x_i = \sum_{i=0}^D w_{ji} x_i \quad x_0 = 1 \]

\[ i \text{ labels inputs, } j \text{ labels hiddens, } k \text{ labels outputs.} \]
Error backpropagation

We do each pattern separately, so we consider $E^n$

$$
y_k(x^n, w) = h_2(a^n_k) = h_2 \left( \sum_{j=0}^{M} w_{kj} h_1(a^n_j) \right) = h_2 \left( \sum_{j=0}^{M} w_{kj} h_1 \left( \sum_{i=0}^{D} w_{ji} x^n_i \right) \right)
$$

$$\frac{\partial E^n}{\partial w_{kj}} = (y^n_k - t^n_k) \frac{\partial y^n_k}{\partial w_{kj}} = (y^n_k - t^n_k) h'_2(a^n_k) \frac{\partial a^n_k}{\partial w_{kj}} = (y^n_k - t^n_k) h'_2(a^n_k) h_1(a^n_j)
$$

$$= \delta^n_k h_1(a^n_j)$$

$$\delta^n_k = (y^n_k - t^n_k) h'_2(a^n_k)$$

$$\frac{\partial E^n}{\partial w_{ji}} = \sum_{k=1}^{K} (y^n_k - t^n_k) \frac{\partial y^n_k}{\partial w_{ji}} = \sum_{k=1}^{K} (y^n_k - t^n_k) h'_2(a^n_k) \frac{\partial a^n_k}{\partial w_{ji}}$$

$$= \sum_{k=1}^{K} \delta^n_k w_{kj} h'_1(a^n_j) \frac{\partial a^n_j}{\partial w_{ji}} = \sum_{k=1}^{K} \delta^n_k w_{kj} h'_1(a^n_j) x^n_i = \delta^n_j x^n_i$$

$$\delta^n_j = h'_1(a^n_j) \sum_{k=1}^{K} \delta^n_k w_{kj}$$
The back propagation extends to arbitrary layers:

1. $z_i^n = x_i^n$ forward propagation all activations $z_j^n = h_1(a_j^n)$ and $z_k^n = h_2(a_k^n)$, etc.

2. Compute the $\delta_k^n$ for the output units, and back-propagate the $\delta$ to obtain $\delta_j^n$ each hidden unit $j$

3. $\partial E^n / \partial w_{kj} = \delta_k^n z_j^n$ and $\partial E^n / \partial w_{ji} = \delta_j^n z_i^n$

4. for batch mode, $\partial E / \partial w_{ji} = \sum_n \partial E^n / \partial w_{ji}$

$E$ is a function of $\mathcal{O}(|w|)$ variables. In general, the computation of $E$ requires $\mathcal{O}(|w|)$ operations. The computation of $\nabla E$ would thus require $\mathcal{O}(|w|^2)$ operations.

The backpropagation method allows to compute $\nabla E$ efficiently, in $\mathcal{O}(|w|)$ operations.
Regularization

Complexity of neural network solution is controlled by number of hidden units

sum squared test error for different number of hidden units and different weight initializations. Error is also affected by local minima.
Part of the cause of local minima is the saturation of the sigmoid functions \( \tanh(\sum w_{ij}x_j) \). When \( w_{ij} \) becomes large, any change in its value hardly affects the output, implying \( \nabla_{ij}E = 0 \).

One can partly prevent this from happening by

- choosing \( \tanh \) instead of \( \sigma \) transfer functions and scaling of inputs and outputs with mean zero and standard deviation one
- proper initialisation of \( w_{ij} \) with mean zero and standard deviation of order \( 1/\sqrt{n_1} \), with \( n_1 \) the number of inputs to neuron \( i \).
- add regularizer such as \( \sum_i w_i^2 \) to cost keeps weights small
- dropouts, other transfer functions, adding noise, ....
MLPs are universal approximators

Consider $2^n$ binary patterns in $n$ dimensions and two classes:

$$x^\mu \rightarrow c^\mu = \pm 1, \quad x_i^\mu = \pm 1$$

Use $2^n$ hidden units, labeled $j = 0, \ldots, 2^n - 1$, $i$ labels input. Set

$$w_{ji} = b \quad \text{if $i$th digit in binary repr. of $j$ is 1}$$

$$w_{ji} = -b \quad \text{else}$$

<table>
<thead>
<tr>
<th>$j$</th>
<th>binary</th>
<th>$w_{j1}$</th>
<th>$w_{j2}$</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$\sum_i w_{0i} x_i$</th>
<th>$w_{1i} x_i$</th>
<th>$w_{2i} x_i$</th>
<th>$w_{3i} x_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00</td>
<td>-b</td>
<td>-b</td>
<td>-1</td>
<td>-1</td>
<td>2b</td>
<td>0</td>
<td>0</td>
<td>-2b</td>
</tr>
<tr>
<td>1</td>
<td>01</td>
<td>-b</td>
<td>b</td>
<td>-1</td>
<td>1</td>
<td>0</td>
<td>2b</td>
<td>-2b</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>b</td>
<td>-b</td>
<td>1</td>
<td>-1</td>
<td>0</td>
<td>-2b</td>
<td>2b</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>b</td>
<td>b</td>
<td>1</td>
<td>1</td>
<td>-2b</td>
<td>0</td>
<td>0</td>
<td>2b</td>
</tr>
</tbody>
</table>
MLPs are universal approximators

Use threshold of \((n - 1)b\) at each hidden unit.  
\[ z_j = \Theta[\sum_i w_{ji} x_i - (n - 1)b] \]

The remaining problem has \(p = 2^n\) patterns in \(2^n\) dimensions and is linearly separable.

Define \(c = \text{sign} [\sum_{j=0}^3 w_j z_j]\).

<table>
<thead>
<tr>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(z_0)</th>
<th>(z_1)</th>
<th>(z_2)</th>
<th>(z_3)</th>
<th>(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>\text{sign}[w_0]</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>\text{sign}[w_1]</td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>\text{sign}[w_2]</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>\text{sign}[w_3]</td>
</tr>
</tbody>
</table>

The combination of linear summation and non-linear functions can create many different functions.

- The MLP with a single hidden layer can map any continuous function [Cybenko, 1989, Hornik et al., 1989]
- Smooth
Emergence of simple-cell receptive field properties by learning a sparse code for natural images [Olshausen et al., 1996]

Efficient coding following Barlow’s principle of redundancy reduction [Barlow, 1961, Barlow et al., 1989]:

"a useful goal of sensory coding is to transform the input in such a manner that reduces the redundancy due to complex statistical dependencies in the input”

Entropy of a code:

\[ H(a_1, \ldots, a_n) \leq \sum_{i=1}^{n} H(a_i) \]

LHS Entropy=Information is maximised when \( a_i \) are independent.
Properties of receptive fields are:
- spatially localized
- oriented
- bandpass (selective to structure at different scales)

Idea:
- Describe image as a linear superposition of basic components,

\[ I(x, y) = \sum_i a_i \phi_i(x, y) \]

which are adapted to best account for the statistics of the data.
- Sparsity: each image requires only a small subset of these components
PCA

Consider an ensemble of images: $I_n(z), n = 1, \ldots, N$ with $z = (x, y)$. Subtract the mean value such that $\sum_n I_n(z) = 0$. Compute the covariance matrix

$$C(z, z') = \frac{1}{N} \sum_n I_n(z) I_n(z') \quad (\equiv \langle I(z) I(z') \rangle)$$

Compute the eigenfunctions

$$C \phi_i = \lambda_i \phi_i \quad \int dz' C(z, z') \phi_i(z') = \lambda_i \phi_i(z) \quad \left( \int dz' = \sum_{z'} \right)$$

Eigenfunctions are ortho-normal: $\int dz \phi_i(z) \phi_j(z) = \delta_{ij}$.

The image can be approximately represented in terms of the $K$ components with largest eigenvalues:

$$I(z) = \sum_{i=1}^{\infty} a_i \phi_i(z) \approx \sum_{i=1}^{K} a_i \phi_i(z) \quad a_i = \int dz I(z) \phi_i(z)$$
The components are independent. The transformation from $I(z) \rightarrow a_{1:K}$ is a decorrelation or whitening.

$$\langle a_i a_j \rangle = \int dz \int dz' \phi_i(z) \phi_j(z') \langle I(z) I(z') \rangle = \int dz \int dz' \phi_i(z) \phi_j(z') C(z, z')$$

$$= \int dz \phi_i(z) \lambda_j \phi_j(z) = \lambda_j \delta_{ij}$$

Eigenfunctions are not localized and do not resemble cortical receptive fields.
Finding a good code can be formulated as an optimisation problem:

\[ E = \left\langle \frac{1}{2} \int dz \left( I(z) - \sum_i a_i \phi_i(z) \right)^2 \right\rangle + \lambda \sum_i S \left( \frac{a_i}{\sigma} \right) \]

with \( S(\cdot) = |\cdot| \) or other non-linearity. \( \left\langle \cdot \right\rangle \) is average over images.

The non-linearity \( S \) favours sparse solutions.
Learning

For given $\phi_i(x, y)$, optimize $a_i$ by gradient descent: $\Delta a_i = -\eta \frac{\partial E}{\partial a_i}$ with

$$\frac{\partial E}{\partial a_i} = \left\langle -\int dz \left( I(z) - \sum_j a_j \phi_j(z) \right) \phi_i(z) \right\rangle + \frac{\lambda}{\sigma} S' \left( \frac{a_i}{\sigma} \right)$$

$$= -b_i + \sum_j a_j C_{ij} + \frac{\lambda}{\sigma} S' \left( \frac{a_i}{\sigma} \right)$$

with $b_i = \int dz \left\langle I(z) \right\rangle \phi_i(z)$ and $C_{ij} = \int dz \phi_i(z) \phi_j(z)$.

Update $\phi_i(x, y)$ according to gradient descent:

$$\Delta \phi_i(z) = -\eta \frac{\partial E}{\partial \phi_i(z)} = \eta a_i \left( \left\langle I(z) \right\rangle - \sum_j a_j \phi_j(z) \right)$$

Repeat until convergence.
Learned features $\phi_i$ equal the basic components that generate the training samples. Learning discovers the mechanism that generated the data.
Training $16 \times 16$ patches of 10 natural images of size $512 \times 512$

Basis functions are oriented, localised and at different spatial scale. Learning discovers the 'building blocks' of natural images. Spatial scales range is approx. 1.1 octave and aspect ratio of 1.3 (simple cell receptive fields 1.5 and 2, respectively)
Training $16 \times 16$ patches of 10 natural images of size $512 \times 512$

Histogram of activities $a_i$ after training (solid) and before training (dashed) averaged over all images show sparsity.
In each time interval $\Delta t$ the neuron can either emit one spike or no spike ($y = 1, 0$) The probability of $y = 1$ for neuron $i$ is

$$p(y_i = 1, t + 1|y(t)) = \sigma \left( \sum_{j \neq i} I_{ij} y_j(t) + \Theta_i \right), \quad \sigma(x) = \frac{1}{2} (1 + \tanh(x)).$$

$t$ labels the discretized time in units of $\Delta t$. The maximal firing frequency is $1/\Delta t$. 
Replace the binary variables $y_i = 0, 1$ by the variables $s_i = \pm 1$ using the relation $y_i = \frac{1}{2}(s_i + 1)$.

$$p(s'_i, t + 1|s, t) = \sigma(s'_ih_i(s(t))) \quad h_i(s) = \sum_{j \neq i} w_{ij}s_j + \theta_i$$

$$w_{ij} = \frac{1}{2}I_{ij} \quad \theta_i = \Theta_i + \frac{1}{2} \sum_{j \neq i} I_{ij}$$
Network dynamics

Neuron dynamics:

\[ p(s'_i | s) = \frac{1}{2} (1 + \tanh(h_is'_i)) \]

\[ h_i = \sum_j w_{ij}s_j + \theta_i \]

Neurons are updated sequentially.

Transitions are allowed from state \( s \) to any of its nearest neighbors \( s' = F_i s \)

\[ T(F_is | s) = \frac{1}{n} p(s'_i | s) \quad T(s | s) = 1 - \sum_i T(F_is | s) \]
Boltzmann-Gibbs distributions

The stationary distribution satisfies:

\[ \sum_{s'} T(s|s')p(s') = p(s) = \sum_{s'} T(s'|s)p(s) \]

Detailed balance:

\[ T(s|s')p(s') = T(s'|s)p(s) \quad \text{for all } s, s'. \]

If detailed balance holds, it implies that \( p(s) \) is a stationary distribution of \( T \). However, the reverse is not true.
For sequential dynamics \( s' = F_is \):

\[
T(F_is|s) = \frac{1}{n} p(s'|s) \Rightarrow \frac{T(s|F_is)}{T(F_is|s)} = \exp(2h_is_i).
\]

Consider the Boltzmann-Gibbs distribution

\[
p(s) = \frac{1}{Z} \exp(-E(s))
\]

\[
-E(s) = \frac{1}{2} \sum_{ij} w_{ij} s_i s_j + \sum_i \theta_i s_i.
\]

\[
Z = \sum_s \exp(-E(s))
\]

Depends only on symmetric part of \( w_{ij} \).

\[
\frac{p(s)}{p(F_is)} = \exp(2(\sum_j w_{ij}^s s_j + \theta_i) s_i).
\]

Thus, \( p(s) \) satisfies detailed balance when \( w_{ij} \) symmetric.
Intractability

From the Boltzmann distribution, one can compute simple statistics such as

\[ \langle s_i \rangle = \sum_s s_i p(s) \text{ or } \langle s_i s_j \rangle = \sum_s s_i s_j p(s) \]

These sums involve $2^n$ terms and can in general not be simplified unless eg. $p(s) = \prod_j p_j(s_j)$:

\[ \langle s_i \rangle = \sum_s s_i \prod_j p_j(s_j) = \sum_{s_i} s_i p_i(s_i) \prod_{j \neq i} \sum_{s_j} p_j(s_j) \]

Complexity is essentially in $Z$:

\[
\langle s_i \rangle = \frac{1}{Z} \frac{\partial Z}{\partial \theta_i} \\
\langle s_i s_j \rangle = \frac{1}{Z} \frac{\partial^2 Z}{\partial \theta_i \partial \theta_j}
\]
Mean field theory

As for the Ising model, we choose for $q(s)$ a factorized distribution:

$$q(s) = \prod_i \frac{1}{2}(1 + m_i s_i), \quad m_i = \langle s_i \rangle_q$$

Minimizing $KL(q|p) = \sum_s q(s) \log \frac{q(s)}{p(s)}$ wrt $q$ yields the MF equations:

$$m_i = \tanh \left( \sum_{j=1}^n w_{ij} m_j + \theta_i \right)$$
Linear response correction

We can also compute the correlations in the mean field approximation. The crucial observation is that both the mean firing rates and the correlations can be computed as derivatives of the partition function:

\[
\langle s_i \rangle = \frac{\partial \log Z}{\partial \theta_i}
\]

\[
\chi_{ij} = \frac{\partial^2 \log Z}{\partial \theta_i \partial \theta_j}
\]

with the correlations \( \chi_{ij} = \langle s_i s_j \rangle - \langle s_i \rangle \langle s_j \rangle \). Thus,

\[
\chi_{ij} = \frac{\partial \langle s_i \rangle}{\partial \theta_j} \approx \frac{\partial m_i}{\partial \theta_j}
\]

Because the mean field equations give us an implicit relation between \( m_i \) and \( \theta_j \), we can
derive

$$
\delta m_i = \frac{1}{1 - m_i^2} \left( \sum_j w_{ij} \delta m_j + \delta \theta_i \right)
$$

$$
\frac{\partial \theta_i}{\partial m_j} = \frac{\delta_{ij}}{1 - m_i^2} - w_{ij} \equiv A_{ij} = \chi_{ij}^{-1}
$$

Thus,

$$
\langle s_i \rangle \approx m_i \quad \langle s_i s_j \rangle \approx m_i m_j + A_{ij}^{-1}
$$
Boltzmann Machines

The basic idea is to treat Boltzmann-Gibbs distribution as a statistical model $p(s|w, \theta)$, and use standard statistical tools to estimate its parameters.

Simplest case: no hidden units.

Learning: given a set of $P$ training patterns $s^{\mu} = (s^{\mu}_1, \ldots, s^{\mu}_n)$ with $\mu = 1, \ldots, P$, find $w, \theta$ such that the $p(s|w, \theta)$ 'best' describes these data.

$$L(w, \theta) = \frac{1}{P} \sum_{\mu} \log p(s^{\mu}_1, \ldots, s^{\mu}_n|w, \theta)$$

and maximize this function wrt to $w$ and $\theta$. 
We absorbe the thresholds:

\[
p(s|w) = \frac{1}{Z} \exp \left( \frac{1}{2} \sum_{i=0}^{n} \sum_{j=0}^{n} w_{ij} s_i s_j \right)
\]

\[
= \frac{1}{Z} \exp \left( \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} s_i s_j + \sum_{i=1}^{n} \theta_i s_i \right)
\]

with \(w_{i0} = \theta_i\) and \(s_0 = 1\).

\[
\frac{\partial \log p(s|w)}{\partial w_{ij}} = -\frac{\log Z}{\partial w_{ij}} + s_i s_j = -\langle s_i s_j \rangle + s_i s_j
\]

\[
\frac{\partial L}{\partial w_{ij}} = \frac{1}{P} \sum_{\mu} \frac{\partial \log p(s^{\mu}|w)}{\partial w_{ij}} = -\langle s_i s_j \rangle + \frac{1}{P} \sum_{\mu} s^{\mu}_i s^{\mu}_j = -\langle s_i s_j \rangle + \langle s_i s_j \rangle_c
\]
This maximization can be easily performed by computing the gradients of $L$ wrt $w_{ij}$ and $\theta_i$:

\[
\frac{\partial L}{\partial \theta_i} = \left( \langle s_i \rangle_c - \langle s_i \rangle \right),
\]

\[
\frac{\partial L}{\partial w_{ij}} = \left( \langle s_is_j \rangle_c - \langle s_is_j \rangle \right) i \neq j.
\]

Free expectations:

\[
\langle s_i \rangle = \sum_s s_i p(s), \quad \langle s_is_j \rangle = \sum_s s_is_j p(s)
\]

Clamped expectations:

\[
\langle s_i \rangle_c = \frac{1}{P} \sum_{s_i^\mu} s_i^\mu, \quad \langle s_is_j \rangle_c = \frac{1}{P} \sum_{s_i^\mu s_j^\mu} s_i^\mu s_j^\mu
\]
The simplest learning procedure is gradient ascent: start at $t = 0$ with a random initial value of all weights and thresholds and to iteratively change these values in the direction of their gradients:

$$w_{ij}(t+1) = w_{ij}(t) + \eta \frac{\partial L}{\partial w_{ij}}$$

$$\theta_i(t+1) = \theta_i(t) + \eta \frac{\partial L}{\partial \theta_i}$$

which increases $L$ at each step (for sufficiently small $\eta$).

Learning terminates when the gradients are zero, i.e. at a local maximum. The first and second order statistics of the Boltzmann distribution $p$ and of the data are equal.
Hidden units

Denote visible units by \( x \), hidden units by \( h \) and total state \( y = (x, h) \). The likelihood is given as:

\[
L(w) = \frac{1}{P} \sum_{\mu} \log p(x^\mu | w) = \frac{1}{P} \sum_{\mu} \log \sum_h p(x^\mu, h | w)
\]

\[
\frac{\partial L(w)}{\partial w_{ij}} = \frac{1}{P} \sum_{\mu} \frac{1}{p(x^\mu | w)} \sum_h \frac{\partial p(x^\mu, h | w)}{\partial w_{ij}} = \frac{1}{P} \sum_{\mu, h} \frac{p(x^\mu, h | w)}{p(x^\mu | w)} \frac{\partial \log p(x^\mu, h | w)}{\partial w_{ij}}
\]

\[
= \frac{1}{P} \sum_{\mu, h} p(h | x^\mu, w) \frac{\partial \log p(x^\mu, h | w)}{\partial w_{ij}} = \frac{1}{P} \sum_{\mu, h} p(h | x^\mu, w) \left( -\langle y_i y_j \rangle + y_i^\mu y_j^\mu \right)
\]

\[
= -\langle y_i y_j \rangle + \langle y_i y_j \rangle_c
\]

with \( y^\mu = (x^\mu, h) \).
Hidden units

When \(i, j \in x\):

\[
\langle y_i y_j \rangle = \langle x_i x_j \rangle, \quad \langle x_i x_j \rangle_c = \frac{1}{P} \sum_{\mu} x^\mu_i x^\mu_j
\]

When \(i, j \in h\):

\[
\langle y_i y_j \rangle = \langle h_i h_j \rangle, \quad \langle y_i y_j \rangle_c = \frac{1}{P} \sum_{\mu} \langle h_i h_j \rangle_\mu, \quad \langle h_i h_j \rangle_\mu = \sum_h p(h|x^\mu, w)h_i h_j
\]

When \(i \in x, j \in h\):

\[
\langle y_i y_j \rangle = \langle x_i h_j \rangle, \quad \langle y_i y_j \rangle_c = \frac{1}{P} \sum_{\mu} x^\mu_i \langle h_j \rangle_\mu, \quad \langle h_j \rangle_\mu = \sum_h p(h|x^\mu, w)h_j
\]
Mean field learning

One can use the mean field and linear response approach to approximate the learning rule.

In the absence of hidden units we can directly solve the fixed point equations. Define

$$C_{ij} = \langle s_is_j \rangle_c - \langle s_i \rangle_c \langle s_j \rangle_c$$

From $\frac{\partial L}{\partial \theta_i} = 0$ we compute $m$:

$$\langle s_i \rangle = \langle s_i \rangle_c \quad m_i \approx \langle s_i \rangle_c$$

From $\frac{\partial L}{\partial w_{ij}} = 0$ and LR approximation we compute $w$:

$$\langle s_is_j \rangle = \langle s_is_j \rangle_c \quad \chi_{ij} = C_{ij}$$

$$\chi_{ij}^{-1} = C_{ij}^{-1} \quad w_{ij} = \frac{\delta_{ij}}{1 - m_i^2} - (C^{-1})_{ij}$$
From MF equations we compute $\theta$:

$$\theta_i = \tanh^{-1}(m_i) - \sum_{j=1}^{n} w_{ij} m_j$$
Classification of digits
Sample of 60000 training patterns and 10000 test patterns of the $28 \times 28$ handwritten digits of the U.S. Postal Service Office of Advanced Technology. Patterns are binary and 10 % pixel noise is added.
Learning procedure

One Boltzmann Machine per digit. Per digit, we use approximately 6000 patterns for training. We thus obtain 10 Boltzmann distributions, each with its own parameters $W^\alpha = (w^\alpha_{ij}, \theta^\alpha_i)$, $\alpha = 1, \ldots, 10$.

We classify each pattern $s$ to the model $\alpha$ with the highest probability:

$$\text{class}(s) = \arg\max_{\alpha} p_\alpha(s),$$

$$p_\alpha(s) = \frac{1}{Z(W^\alpha)} \exp\left(\frac{1}{2} \sum_{ij} w^\alpha_{ij} s_i s_j + \theta^\alpha_i s_i \right)$$

The normalization $Z(W^\alpha)$ is intractable and depends on $\alpha$ and therefore affects classification. We use its mean field approximation $\log Z \approx -F$.

Test the performance on 500 of the 10000 test patterns classifies 45 incorrect.

Compare with simple template matching on the mean image yields 123 errors.
Markov processes

Given a Markov process, when is the stationary distribution unique? What are the dominant relaxation times? To answer these questions we make a brief digression.

Let $S$ denote the set of all state vectors $s$. $s \in S$ is a binary vector of length $n$ and thus $s$ can take on $2^n$ different values. Therefore, $p_t(s)$ is a vector of length $2^n$ and $T(s'|s)$ is a $2^n \times 2^n$ matrix.

Since $p_t(s)$ denotes a probability vector, it must satisfy $\sum_s p_t(s) = 1$. In addition, $T(s'|s)$ is a probability vector in $s'$ for each value of $s$ and therefore each column must add up to one:

$$\sum_{s'} T(s'|s) = 1.$$ 

Matrices with this property are called stochastic matrices.

Let us denote the eigenvalues and left and right eigenvectors of $T$ by $\lambda_\alpha, l_\alpha, r_\alpha, \alpha =$
1, \ldots, 2^n$, respectively. In matrix notation we have

\[ Tr_\alpha \quad = \quad \lambda_\alpha r_\alpha \]
\[ l_\alpha^\dagger T \quad = \quad \lambda_\alpha l_\alpha^\dagger \]

Since \( T \) is a non-symmetric matrix, the left and right eigenvectors are different, non-orthogonal and complex valued. \( ^\dagger \) denotes complex conjugation and transpose. The eigenvalues are complex valued. Under rather general conditions each set of eigenvectors spans a non-orthogonal basis of \( \mathbb{C}^{2^n} \). These two bases are dual in the sense that:

\[ l_\alpha^\dagger r_\beta = \delta_{\alpha\beta}. \]

We can therefore expand \( T \) on the basis of its eigenvectors:

\[ T = \sum_{\alpha=1}^{2^n} \lambda_\alpha r_\alpha l_\alpha^\dagger \]

---

\(^4\)In general, the number of eigenvalues of \( T \) can be less than \( 2^n \). However, for our purposes we can ignore this case
At \( t = 0 \) the network is in a state \( s^0 \):

\[
p_{t=0}(s) = \delta_{s, s^0}
\]

At some later time:

\[
p_t = T^t p_0 = \sum_{\alpha} \lambda_{\alpha}^t r_{\alpha}(l_{\alpha}^\dagger p_0)
\]

Stationarity:

\[
Tp_\infty = p_\infty.
\]

Thus, the stationary distribution is a right eigenvector of \( T \) with eigenvalue 1.

The time to approach stationarity is also given by the eigenvalues of \( T \). In particular, each eigenvalue whose norm \(|\lambda_{\alpha}| < 1\) corresponds to a transient mode with \textit{relaxation time}

\[
\tau_{\alpha} = \frac{-1}{\log \lambda_{\alpha}}.
\]

We will now study under which conditions \( p_\infty \) is unique.
Some properties

A Markov process is called *irreducible*, or *ergodic*, on a subset of states $C \subset S$ if for any state $s \in C$ there is a finite probability to visit any other state $s' \in C$:

$$s = s^0, s^1, \ldots, s^k = s'$$

with $T(s^i|s^{i-1}) > 0$, $i = 1, \ldots, k$.

A subset of states $C \subset S$ is called *closed* when the Markov process can never escape from $C$, once entered:

$$T(s'|s) = 0 \quad \text{for all } s \in C, s' \not\in C.$$  

In general, we can decompose the state space $S$ uniquely into closed irreducible subsets

$$S = \mathcal{T} \cup C_1 \cup C_2 \ldots,$$

where $\mathcal{T}$ is a set of *transient states* and the $C_i$ are closed irreducible sets.
Perron-Frobenius theorem

An irreducible Markov process $T$ of periodicity $d$ has $d$ eigenvalues given by

$$
\lambda_m = \exp(2\pi im/d), m = 0, \ldots, d - 1,
$$

and all remaining eigenvalues of $T$ are inside the unit circle in the complex plane: $|\lambda_\alpha| < 1$.

In particular, $T$ has exactly one eigenvalue 1. Its corresponding right eigenvector is equal to the (unique) stationary distribution (ergodicity).

The left eigenvector with eigenvalue 1 is $\propto (1, \ldots, 1)$.
Non-ergodic behavior

A non-irreducible or non-ergodic Markov process has more than one eigenvalue 1 and therefore more than one left and right eigenvector with eigenvalue 1. Let us denote these eigenvectors by $l_1, \ldots, l_k$ and $r_1, \ldots, r_k$, respectively. Any linear combination of the right eigenvectors

$$p_\infty = \sum_{\alpha=1}^{k} \rho_\alpha r_\alpha$$

is therefore a stationary distribution, assuming proper normalization: $p_\infty(s) \geq 0$ for all $s$ and $\sum_s p_\infty(s) = 1$. Thus, there exists a manifold of dimension $k - 1$ of stationary distributions.
The $k$ left eigenvectors with eigenvalue 1 encode *invariants* of the dynamics. Define

$$L_\alpha(p_t) = l_\alpha^\dagger p_t, \alpha = 1, \ldots, k.$$  

Then $L_\alpha$ is invariant under the Markov dynamics:

$$L_\alpha(p_{t+1}) = l_\alpha^\dagger p_{t+1} = l_\alpha^\dagger T p_t = l_\alpha^\dagger p_t = L_\alpha(p_t).$$  

- $k = 1$ ensures invariance of normalisation: $l_1 \propto (1, \ldots, 1)$
- the remaining $k - 1$ invariants are determined by the initial distribution $p_0$. They parametrize the stationary manifold and determine uniquely the stationary distribution.
Stationary distribution depends on initial state

From

$$p_\infty = \sum_{\alpha=1}^{k} \rho_\alpha r_\alpha$$

$$l_\alpha^r r_\beta = \delta_{\alpha\beta}$$

we obtain

$$L_\alpha = l_\alpha^r p_0 = l_\alpha^r p_\infty = \rho_\alpha$$

Thus, the stationary state depends on the initial state as:

$$p_\infty = \sum_{\alpha=1}^{k} (l_\alpha^r p_0) r_\alpha.$$ 

Note, that in the ergodic case ($k = 1$) the dependence on the initial state disappears, as it should, since $l_1^r p_0 = 1$ for any initial distribution $p_0$.
Two neurons sequential dynamics

Consider a network of two neurons connected symmetrically by a synaptic weight $w = w_{12} = w_{21}$ and $\theta_1 = \theta_2 = 0$. First consider sequential dynamics. The transition matrix $T$ has 4 eigenvalues.

- $w$ small: ergodic, unique stationary distribution Relaxation time determined by largest eigenvalue.
• $w$ large: the relaxation time becomes infinite because a second eigenvalue approaches 1. Some transitions require infinite time and therefore ergodicity is broken. Two ergodic components (states $(1,1)$ and $(-1,-1)$).
Some additional properties

- symmetric networks have real eigenvalues (both parallel and sequential dynamics)

- for sequential dynamics (symmetric or asymmetric) all eigenvalues are within the circle centered at \( \frac{1}{2} + 0i \) with radius \( \frac{1}{2} \). Therefore, sequential dynamics has always periodicity 1.

- parallel dynamics with symmetric weights has at most periodicity 2

- Parallel dynamics with asymmetric weights can have arbitrary periodicity
Summary

1. The behaviour of a network of stochastic neurons can be described as a first order Markov process.

2. The transition matrix can be analysed in terms of its eigenvalues and eigenvectors. Eigenvalues close to 1, imply long convergence times.

3. Ergodic networks cannot be used as memory, because of the unique asymptotic state.

4. For non-ergodic networks, the asymptotic behaviour of the network depends then on its initial state and can be used as a memory
Weak pairwise correlations imply strongly correlated network states in a neural population

Analysis of spike data from multiple neurons in salamander retina.

Shows that observed correlation structure is well captured by Ising model with pairwise interactions.
a, A segment of the simultaneous responses of 40 retinal ganglion cells in the salamander to a natural movie clip. b, Discretization of population spike trains into a binary pattern is shown for the green boxed area in a. c, Example cross-correlogram between two neurons with strong correlations. Inset shows the same cross-correlogram on an expanded time scale in ms. d, Histogram of correlation coefficients for all pairs of 40 cells from a. e, Probability distribution of synchronous spiking events in the 40 cell population in response to a long natural movie (red) approximates an exponential (dashed red). The distribution of synchronous events for the same 40 cells after shuffling each cell’s spike train to eliminate all correlations (blue), compared to the Poisson distribution (dashed light blue). f, The rate of occurrence of each pattern predicted if all cells are independent is plotted against the measured rate. Each dot stands for one of the $2^{10} = 1024$ possible binary activity patterns for 10 cells. Black line shows equality.
Observations

Most pairs of cells are weakly correlated (fig 1.d).

But correlations are not in agreement with independent neurons (fig. 1.e).

Word (=10 bit activity pattern) frequencies do not follow independent Poisson frequencies (fig. 1.f).
Boltzmann Machine

Approach:

- Consider subsets of 10 cells out of 40 cells (intractability)
- Learn Boltzmann Machine
Boltzmann distribution

\[ p(s) = \frac{1}{Z} \exp \left( \sum_i h_i s_i + \frac{1}{2} \sum_{ij} J_{ij} s_i s_j \right) \]

a, Example of \( J_{ij} \) and \( h_i \) for a group of 10 cells. b, Histograms of \( h_i \) and \( J_{ij} \) values from 250 different groups of 10 cells. c, Non-trivial relation between coupling and correlation. Left: Cells A and B are not connected, but correlated through C. Right: Cells A and B are connected but their correlation is frustrated through C. 40% of triplets are frustrated. d, Interaction strength \( J_{ij} \) plotted against the correlation coefficient \( C_{ij} \); each point shows the value for one cell pair averaged over many different groups of cells (190 pairs from 250 groups), and error bars show standard deviations.
Observations

$h_i$ negative for most cells for low mean activity. Most correlations are positive. (fig.3.b).

Non-trivial relation between couplings $J_{ij}$ and correlations $C_{ij}$.
a, As in fig. 1, the rate of occurrence of each word predicted from the Ising model P2 is plotted against the measured rate (red dots). For comparison, the independent model P1 is also plotted (from Fig. 1f; grey dots). b, Histogram of KL divergences for model P1 and P2; data from 250 groups.

Ising model predicts very well empirical data.
Deep belief networks [Hinton and Salakhutdinov, 2006]

Finding important features in data helps to build better pattern recognition systems.

The problem can be viewed as a dimension reduction problem:
- PCA
- (non-linear) auto-encoders

Optimization is difficult with many layers. Idea:
- initialise by training RBMs for pairs of layers
- train with gradient descent/backpropagation
Restricted BM and contrastive divergence

A layer of visible neurons \((v_i, 'pixels')\) and a layer of hidden neurons \((h_j, 'features')\) with between-layer connections only.

\[
E(v,h) = - \sum_i b_i v_i - \sum_j b_j h_j - \sum_{i,j} w_{ij} v_i h_j \quad v_i, h_j \text{ binary}
\]

BM learning rule

\[
\Delta w_{ij} = \eta \left( \langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{free}} \right)
\]

\[
\langle v_i h_j \rangle_{\text{data}} = \sum_{\mu \in \text{data}} v_i^\mu \langle h_j \rangle_{v^\mu} \quad \langle h_j \rangle_v = \sigma(\sum_i v_i^\mu w_{ij} + b_j)
\]

\[
\langle v_i h_j \rangle_{\text{free}} = \sum_{v,h} v_i h_j p(v,h)
\]

Contrastive divergence approximates \(\langle v_i h_j \rangle_{\text{free}}\) by sampling a few steps:

\[
v^\mu \rightarrow h^\mu \rightarrow (v^\mu)' \rightarrow (h^\mu)' \quad \langle v_i h_j \rangle_{\text{free}} \approx \sum_{\mu} (v^\mu)'_i (h^\mu)'_j
\]
DBN construction
Experiments: MNIST data

784 − 1000 − 500 − 250 − 30 autoencoder
60,000 training images and 10,000 test images

Top to bottom: test samples; DBN; logistic PCA with 30 components; PCA with 30 components. The average squared errors for the last three rows are 3.00, 8.01, and 13.87.
Experiments: MNIST data

Fig. 3. (A) The two-dimensional codes for 500 digits of each class produced by taking the first two principal components of all 60,000 training images. (B) The two-dimensional codes found by a 784-1000-500-250-2 autoencoder. For an alternative visualization, see (B).

DBN yields better visualisation than PCA
Experiments: documents

Each newswire story was represented as a 2000 dimensional vector of common word counts.

Training:
- 2000-500-250-125-2 autoencoder
- 402,207 training documents, 402,207 test documents

Spiking neurons [O’Connor et al., 2013]

The DBN functionality has recently been reproduced using leaky integrate and fire neurons.
- Siegert neurons have rate input-output transfer functions equivalent to LIF neurone with poisson inputs.
- train on 120.000 MNIST images using sigmoid or Siegert neurons
- tested using sigmoid, Siegert or Siegert-Poisson rate neurons.

Table 2 | Classification performance on the MNIST test set for two time-stepped and one event-based LIF neuron model.

<table>
<thead>
<tr>
<th>Neuron model</th>
<th>Domain</th>
<th>% correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sigmoid-Binary</td>
<td>time-step</td>
<td>97.48</td>
</tr>
<tr>
<td>Siegert</td>
<td>time-step</td>
<td>95.2</td>
</tr>
<tr>
<td>LIF</td>
<td>event-based</td>
<td>94.09</td>
</tr>
</tbody>
</table>

Inputs for the event-based model were simulated Poisson spike trains (see Section 2.4).
Mining for Structure

Massive increase in both computational power and the amount of data available from web, video cameras, laboratory measurements.

Images & Video
- flickr
- Google
- YouTube

Text & Language
- Wikipedia
- Associated Press

Speech & Audio
- Gene Expression
- Climate Change
- Geological Data

Product Recommendation
- Amazon
- Netflix
- eBay

Relational Data/Social Network
- Facebook
- Twitter

- Develop statistical models that can discover underlying structure, cause, or statistical correlation from data in unsupervised or semi-supervised way.
- Multiple application domains.
Caption Generation with Visual Attention

A man riding a horse in a field.

Xu et al., ICML 2015
Challenges - I

Very different input representations

- Images – real-valued, dense
- Text – discrete, sparse

Difficult to learn cross-modal features from low-level representations.
Multimodal DBM

Gaussian model

Dense, real-valued image features

\( V_{image} \)

Replicated Softmax

Word counts

\( V_{text} \)

(Srivastava & Salakhutdinov, NIPS 2012, ICLR 2014)
Text Generated from Images

**Given**
- dog, cat, pet, kitten, puppy, ginger, tongue, kitty, dogs, furry
- sea, france, boat, mer, beach, river, bretagne, plage, brittany
- portrait, child, kid, ritratto, kids, children, boy, cute, boys, italy

**Generated**
- insect, butterfly, insects, bug, butterflies, lepidoptera
- graffiti, streetart, stencil, sticker, urbanart, graff, sanfrancisco
- canada, nature, sunrise, ontario, fog, mist, bc, morning

Bert Kappen
Text Generated from Images

**Given**
- Portrait, women, army, soldier, mother, postcard, soldiers
- Obama, barackobama, election, politics, president, hope, change, sanfrancisco, convention, rally
- Water, glass, beer, bottle, drink, wine, bubbles, splash, drops, drop
Human-level control through deep reinforcement learning

http://www.nature.com/nature/journal/v518/n7540/full/nature14236.html#videos
References


Bert Kappen