Advanced Machine Learning

Bert Kappen
SNN Radboud University, Nijmegen
Gatsby Unit, UCL London

January 6, 2020
Course setup

- 6 ec course
- examination based on computer exercises
- weekly exercises discussed in tutorial class
- All course materials (slides, exercises) and schedule via [http://www.snn.ru.nl/~bertk/machinelearning/adv_ml.html](http://www.snn.ru.nl/~bertk/machinelearning/adv_ml.html)
Chapter 29 and 30: Monte Carlo sampling

- Uniform sampling
- Importance and Rejection sampling
- Metropolis-Hasting method
- Detour on Markov processes, stationary distribution, ergodicity
- Gibbs sampling
- Hybrid Monte Carlo
The problems

1. generate samples \(\{x^r\}, r = 1, \ldots, R\) from \(p(x)\)

2. estimate

\[
\Phi = \langle \phi(x) \rangle = \int d^n x p(x) \phi(x)
\]

We focus on 1, since 1 solves 2:

\[
\hat{\Phi} = \frac{1}{R} \sum_r \phi(x^r)
\]

\[
\langle \hat{\Phi} \rangle = \Phi
\]

\[
var(\hat{\Phi}) = \frac{\sigma^2}{R}, \quad \sigma^2 = \int d^n x p(x) (\phi(x) - \Phi)^2
\]

when \(\{x^r\}\) independent.
An analogy

Imagine a lake with coordinates \( x = (x_1, x_2) \). You wish to estimate the average plankton concentration

\[
I = \frac{1}{Z} \int d x p^*(x) \phi(x) \quad Z = \int d x p^*(x)
\]

\( p^*(x) \) is the depth of the lake at \( x \). \( \phi(x) \) is the plankton concentration at \( x \). \( Z \) is the total volume of the lake.

Go out with your boat to locations \( x \), measure the depth with a plumbline and the local plankton concentration.

Problem 1 is to sample uniform \( 1 \text{cm}^3 \) from the volume (= sample \( \propto p(x) \) from the surface).
Uniform sampling:

\[ \{ x^r \}, \ r = 1, \ldots, R \]

requires \( a \) samples per dimension \( \to a^n \) samples.

For learning or inference, number of parameters

\[ n = 100 - 1000. \]
Sampling from a multi-variate Gaussian

How to sample from a multi-variate Gaussian distribution $\mathcal{N}(x|m, S)$?

Define $C$ such that $CC^T = S$ and define $y = C^{-1}(x - m)$. Since this is a linear transformation, $y$ is also Gaussian distributed with

$$
\mathbb{E}y = C^{-1}(\langle x \rangle - m) = 0
$$

$$
\mathbb{V}(y) = \mathbb{E}yy^T = C^{-1}\mathbb{E}(x - m)(x - m)^T (C^{-1})^T = C^{-1}S (C^{-1})^T = 1
$$

Thus sample from $y$ can be obtained by sampling each component independently:

$$
p(y) = \prod_i \mathcal{N}(y_i|0, 1)
$$

Sampling from a one dimensional Gaussian can be done by sampling from a uniform distribution using the Box-Muller transformation (Exercise).

Sample from $x$ is obtained by sampling from $y$ and $x = Cy + m$. 
Typical set

Consider the Ising model

\[ p(s|w) = \frac{\exp\left(\frac{1}{2} \sum_{ij} s_is_jw_{ij}\right)}{Z} \]

\( s_i = \pm 1, \quad i = 1, \ldots, n. \) This distribution is intractable to compute, due to the normalisation

\[ Z = \sum_{s_1} \cdots \sum_{s_n} \exp\left(\frac{1}{2} \sum_{ij} s_is_jw_{ij}\right) \]

Total number of states is \( 2^n \), but most probability is concentrated in the so-called typical set \( T \).
Typical set

Consider $X$ from alphabet $A_X = \{a_1, \ldots, a_M\}$. Draw long string of $N$ values. Value $a_i$ occurs approximately $p_iN$ times.

Probability

$$P(X_1, \ldots, X_N) \approx p_1^{p_1N} \cdots p_M^{p_MN} = 2^N \sum_{j=1}^M p_i \log_2 p_i = 2^{-NH}$$

with $H = -\sum_i p_i \log p_i$ the entropy of the distribution.

For $N = 1$, $P(X) \approx 2^{-H}$ when $X$ typical and $P(X) \approx 0$ otherwise. Thus typical size $|T| \approx 2^H$.

Thus, in the case $X = (X_1, \ldots, X_n)$ with $X_i \pm 1$, the probability to hit the typical set is

$$p = \frac{2^H}{2^n}$$

If one draws $R$ samples uniform, the expected number of hits to the typical set is

$$R_{\text{hit}} = R \frac{2^H}{2^n}$$
To ensure that $R_{\text{hit}} \gg 1$ one thus finds

$$R \gg 2^{n-H}$$

What is $H$?

- For high temperature (noise) $H \approx n$. Uniform sampling feasible
- For low temperature $H \ll n \Rightarrow R = O(2^n)$

Uniform sampling only works for uniform distributions.
Importance sampling

Sample from another distribution $q(x)$.

Often one can propose a sample density that is
1) better than uniform and
2) easy to sample from.

For instance, a (spherical) Gaussian:

$$Q^*(x) \propto \exp(- \sum_i x_i^2 / 2)$$
Consider simple 1-d sampling problem. Given \( p(x) \), compute

\[
\Phi = \text{Prob}(x < 0) = \int_{-\infty}^{\infty} \phi(x)p(x)\,dx
\]

with \( \phi(x) = 0, 1 \) if \( x > 0, x < 0 \), respectively.

Naive method: generate \( N \) samples \( X_i \sim p \)

\[
\hat{\Phi} = \frac{1}{N} \sum_{i=1}^{N} \phi(X_i)
\]
Importance sampling

Consider another distribution $q(x)$. Then

$$
\Phi = \text{Prob}(x < 0) = \int_{-\infty}^{\infty} \phi(x) \frac{p(x)}{q(x)} q(x) dx
$$

Importance sampling: generate $N$ samples $X_i \sim q$

$$
\hat{\Phi} = \frac{1}{N} \sum_{i=1}^{N} \phi(X_i) \frac{p(X_i)}{q(X_i)}
$$

Unbiased (= correct) for any $q$!
Optimal importance sampling

The distribution

\[ q^*(x) = \frac{p(x)\phi(x)}{\Phi} \]

is the optimal importance sampler.

One sample \( X \sim q^* \) is sufficient to estimate \( a \):

\[ \hat{\Phi} = \phi(X) \frac{p(X)}{q^*(X)} = \Phi \]
Choose sufficiently broad importance sampler

Toy 1-d amino acid sampling problem, showing effect of poor importance sampler. Left: Gaussian suggests convergence after 500 iterations. Right: Cauchy is broader distribution and shows more robust behaviour.
Normalization

Typically, $p(x)$ can only be easily computed, up to a constant:

$$p(x) = \frac{p^*(x)}{Z} \quad Z = \int dx p^*(x)$$

For instance

$$p(w|D) = \frac{p(D|w)p(w)}{p(D)} \quad p(D) = \int dw p(D|w)p(w)$$

Estimate both numerator and denominator by sampling.

$$\Phi = \int dx p(x)\phi(x) = \frac{\int dx p^*(x)\phi(x)}{\int dx p^*(x)} = \frac{\int dx q(x)\frac{p^*(x)}{q(x)}\phi(x)}{\int dx q(x)\frac{p^*(x)}{q(x)}}$$

Sample $\{x^r\}$ from $q(x)$ and compute

$$w_r = \frac{p^*(x^r)}{q(x^r)} \quad \hat{\Phi} = \frac{\sum_r w_r \phi(x^r)}{\sum_r w_r}$$
Importance sampling

The estimate is biased.

$$\mathbb{E} \hat{\Phi} = \mathbb{E} \left( \frac{\sum_{r=1}^{N} w_r \phi(x^r)}{\sum_{r=1}^{N} w_r} \right) \neq \frac{\mathbb{E} \sum_{r=1}^{N} w_r \phi(x^r)}{\mathbb{E} \sum_{r=1}^{N} w_r} = \frac{N \int dx p^*(x) \phi(x)}{N \int dx p^*(x)} = \Phi$$

However, for large $N$:

$$\sum_{r=1}^{N} w_r = N \mathbb{E} w_r + O \left( \sqrt{N} \right) \approx N \mathbb{E} w_r$$

$$\mathbb{E} \hat{\Phi} \approx \mathbb{E} \left( \frac{\sum_{r=1}^{N} w_r \phi(x^r)}{N \mathbb{E} w_r} \right) = \frac{\mathbb{E} \sum_{r=1}^{N} w_r \phi(x^r)}{N \int dx p^*(x)} = \frac{N \int dx p^*(x) \phi(x)}{N \int dx p^*(x)} = \Phi$$

The estimator $\hat{\Phi}$ is asymptotically unbiased.
Choose, $c$ such that for all $x : cQ^*(x) > P^*(x)$

- generate $x$ from $Q^*(x)$
- generate $u$ uniform from $[0, cQ^*(x)]$
- if $u > P^*(x)$ reject $x$, otherwise accept $x$

This procedure samples $P^*(x)$ because $(x, u)$ uniform from light grey area.

$$
\hat{\Phi} = \sum_r \phi(x^r) \rightarrow \int dx \phi(x)p(x)
$$
Let $p(x)$ and $q(x)$ be spherical Gaussians in $n$ dimensions with mean 0 and $\sigma_q = 1.01 \sigma_p$. 
Since

\[ q(0) = \left( \frac{1}{\sqrt{2\pi\sigma_q^2}} \right)^n \quad p(0) = \left( \frac{1}{\sqrt{2\pi\sigma_p^2}} \right)^n \]

then

\[ c = \frac{p(0)}{q(0)} = \left( \frac{\sigma_q}{\sigma_p} \right)^n = 1.01^n \]

With \( n = 1000 \) we find \( c=20.000 \).

Acceptance rate = \( \frac{\text{volume} \ p}{\text{volume} \ c q} = \frac{1}{c} \)

Thus rejection sampling is inefficient in high dimensions.

A similar argument holds for importance sampling.
**Metropolis algorithm**

The Metropolis algorithm (1956) considers a sampling density which depends on the current sample value: \( q(x|x') \).

Initialize in some random state \( x^1 \)
At iteration $r$, sample $x'$ from $q(x'|x^r)$ and compute

$$a = \frac{p^*(x')q(x^r|x')}{{p^*(x^r)q(x'|x^r)}}$$

If $a \geq 1$, accept $x'$ as the new state: $x^{r+1} = x'$

Else, accept $x'$ as the new state with probability $a$

If accept: $x^{r+1} = x'$, else $x^{r+1} = x^r$
Markov processes

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.

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 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

$$
Tr_\alpha = \lambda_\alpha r_\alpha \\
\dagger_l \lambda_\alpha = \lambda_\alpha \dagger_l
$$

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 $C^{2n}$. These two bases are dual in the sense that:

$$
\dagger_l r_\beta = \delta_{\alpha \beta}.
$$

We can therefore expand $T$ on the basis of its eigenvectors:

$$
T = \sum_{\alpha=1}^{2^n} \lambda_\alpha r_\alpha \dagger_l
$$

---

1In 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:

$$p_t = T^t p_0 = \sum_{\alpha} \lambda^t_\alpha r_\alpha (l_\alpha^\dagger p_0)$$

Stationarity:

$$T p_\infty = p_\infty.$$ 

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 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:

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

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

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

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, with parameters $\rho_\alpha$ such that $p_\infty(s) \geq 0$ for all $s$ and proper normalization: $\sum_s p_\infty(s) = 1$. 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

\[ p_\infty = \sum_{\alpha=1}^{k} \rho_\alpha r_\alpha \]
\[ l_\alpha^\dagger r_\beta = \delta_{\alpha\beta} \]

we obtain

\[ L_\alpha = l_\alpha^\dagger p_0 = l_\alpha^\dagger p_\infty = \rho_\alpha \]

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

\[ p_\infty = \sum_{\alpha=1}^{k} (l_\alpha^\dagger p_0) r_\alpha. \]

Note, that in the ergodic case \((k = 1)\) the dependence on the initial state disappears, as it should, since \(l_1^\dagger p_0 = 1\) for any initial distribution \(p_0\)
Detailed balance

Stationarity:

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

There always exists a stationary solution $p(s)$.

The Markov process $T$ satisfies detailed balance if $\exists p$ such that

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

Detailed balance is a property of the Markov dynamics $T$. It is not true in general.

DB holds for thermodynamic systems and Metropolis Hastings, not for chaotic systems.

If DB $p(s)$ is a stationary distribution of $T$. The reverse is not true.
Summary

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

2. Ergodic systems have unique stationary distribution.

3. Non-ergodic systems have multiple stationary distributions. Their asymptotic behaviour depends on the initial distribution.
Convergence of Metropolis algorithm

Metropolis algorithm is example of Markov process. Given two states $x$ and $x'$. Define

$$a_{x'x} = \frac{p^*(x')q(x|x')}{p^*(x)q(x'|x)}, \quad a_{xx'} = \frac{1}{a_{x'x}}$$

Suppose $a_{x'x} \geq 1$. Then

**Given $x$, the probability to accept $x'$ is**

$$T(x'|x) = q(x'|x)$$

**Given $x'$, the probability to accept $x$ is**

$$T(x|x') = q(x|x')a_{xx'}$$

$$\frac{T(x'|x)}{T(x|x')} = a_{x'x} \frac{q(x'|x)}{q(x|x')} = \frac{p^*(x')}{p^*(x)} = \frac{p(x')}{p(x)}$$

i.e. detailed balance. This implies that the process $T(x'|x)$ converges to $p(x)$. 
Metropolis Hasting for the Ising model

Use MH to sample from the Ising model

\[ p(x) = \frac{1}{Z} \exp (-E(x)) \quad E(x) = -\frac{1}{2} \sum_{i \neq j} w_{ij}x_i x_j = -\sum_{(ij)} w_{ij}x_i x_j \]

Consider \( q(x'|x) \) to implement single spin flips. Define \( F_i x \) is the vector of spins obtained by flipping bit \( i \). Then

\[ q(F_i x|x) = \frac{1}{n} \quad q(x|x) = 0 \]

The MH ratio for the transition from state \( x \) to state \( F_i x \) is

\[ a_{x,F_i x} = \frac{p^*(F_i x)}{p^*(x)} = \exp (-\Delta E_{x,F_i x}) \]

\[ \Delta E_{x,F_i x} = E(F_i x) - E(x) = 2x_i \sum_{j \neq i} w_{ij}x_j \]
Example: 2 spins

\[ E = -wx_1x_2. \]

There are four states \( \alpha = 1:4 \) with energies:

\[ E_{1:4} = \{ E_{++}, E_{+-}, E_{-+}, E_{--} \} = \{-w, w, w, -w\} \]

We define \( \Delta E_{\alpha,\beta} = E_\beta - E_\alpha \):

\[
\Delta E = \begin{pmatrix}
0 & 2w & 2w & 0 \\
-2w & 0 & 0 & -2w \\
-2w & 0 & 0 & -2w \\
0 & 2w & 2w & 0
\end{pmatrix}
\]

We define \( q_{\alpha,\beta} = q(\beta|\alpha) \):

\[
q = \frac{1}{2} \begin{pmatrix}
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 \\
0 & 1 & 1 & 0
\end{pmatrix}
\]
Example: 2 spins

Assume $w > 0$. We have for $\alpha \neq \beta$ $a_{\alpha,\beta} = \min(\exp(-2\Delta E_{\alpha,\beta}), 1)$:

$$a = \begin{pmatrix}
* & e^{-2w} & e^{-2w} & 1 \\
1 & * & 1 & 1 \\
1 & 1 & * & 1 \\
1 & e^{-2w} & e^{-2w} & *
\end{pmatrix}$$

For $\alpha \neq \beta$: $T_{\alpha,\beta} = T(\beta|\alpha) = a_{\alpha,\beta} q_{\alpha,\beta}$:

$$T = \begin{pmatrix}
1 - e^{-2w} & \frac{1}{2} e^{-2w} & \frac{1}{2} e^{-2w} & 0 \\
\frac{1}{2} & 0 & 0 & \frac{1}{2} \\
\frac{1}{2} & 0 & 0 & \frac{1}{2} \\
0 & \frac{1}{2} e^{-2w} & \frac{1}{2} e^{-2w} & 1 - e^{-2w}
\end{pmatrix}$$

Diagonal terms follow from $\sum_{\beta} T_{\alpha,\beta} = 1$.

Interpretation of $T$: from high energy states 2,3 ((+, −) and (−, +)) spins transit with equal probability to one of the two low energy states 1,4 ((+, +) and (−, −)). From low energy states 1,4 their is finite probability to transit to high energy states (decreasing with $w$).
Example: 2 spins

Eigenvalues of $T$ versus $w$. For low $w$ there is one eigenvalue 1 and is ergodic. For high $w$ there are two eigenvalues 1, system is non-ergodic.
Gibbs sampling

- Consider only change of one element $i$ of $(x_1, \ldots, x_n)$ at the time. Update $x \rightarrow x'$ according to

$$q(x'|x) = p(x'_i|x_{\setminus i}) \delta_{x_{\setminus i}, x'_i}$$

with $p(x'_i|x_{\setminus i})$ the conditional distribution from the target distribution $p(x)$.

- Accept: $x^{r+1} = x'$

The one dimensional sampling can be done using for instance Rejection sampling.
Gibbs sampling

Each Gibbs sampling is a MH step with acceptance $a = 1$:

\[
q(x'|x) = p(x'_i|x_{-i}) \delta_{x_{-i}, x'_i}
\]

\[
a = \frac{p(x') q(x|x')}{p(x) q(x'|x)} = \frac{p(x'_i|x_{-i}) p(x'_{-i}) p(x_i|x'_i)}{p(x_i|x_{-i}) p(x_{-i}) p(x'_i|x_{-i})} = \frac{p(x'_i|x_{-i}) p(x_i|x_{-i})}{p(x_i|x_{-i}) p(x'_i|x_{-i})} = 1
\]
Gibbs sampling for the Ising model

\[ p(x) = \frac{1}{Z} \exp(-E(x)) \quad E(x) = -\sum_{(ij)} w_{ij} x_i x_j \]

\[ p(x_i|x_{\setminus i}) = \frac{p(x_i, x_{\setminus i})}{p(x_{\setminus i})} = \frac{e^{-E(x_i, x_{\setminus i})}}{e^{-E(x_i=1, x_{\setminus i})} + e^{-E(x_i=-1, x_{\setminus i})}} = \frac{e^{x_i h_i}}{e^{h_i} + e^{-h_i}} = \sigma(x_i h_i) \]

with \( h_i = \sum_{j \neq i} w_{ij} x_j \).
The detailed balance is the same

\[
\frac{\text{rate } ++ \text{ to } -} {\text{rate } - \text{ to } ++} = \frac{\sigma(-w)}{\sigma(w)} = \frac{1/2}{1/2e^{2w}} = \frac{1/2e^{-2w}}{1/2} = e^{-2w}
\]
Correlations slow down sampling

When $q(x'|x)$ is Gaussian centered on $x$, $rac{q(x'|x)}{q(x|x')}$ independent of $x, x'$:

$$a_{x'x} = \frac{p^*(x')}{p^*(x)}$$

$\epsilon$ large:
Acceptance rate $a_{x'x}$ small.

$\epsilon$ small:
Strong dependence on starting value. Many samples needed to sample.
Correlations slow down sampling

With step $\epsilon$ random, the particle moves a distance $L \approx \sqrt{T}\epsilon$ in $T$ iterations.

If largest length scale is $L$ then

$$T \approx \left( \frac{L}{\epsilon} \right)^2$$
Correlations slow down sampling

Convergence slow when variables correlated: When two variables have marginal width $L$ and conditional width $\epsilon$, the number of iterations scales as $T \approx L^2/\epsilon^2$. 
The Hybrid Monte Carlo Method

Let

\[ P(x) = \frac{e^{-E(x)}}{Z} \]

with \( E \) and its gradient \( \frac{\partial E}{\partial q_i} \) easy to compute.

Gradient information reduces random walk behaviour in Metropolis method.

Double the state space by introducing for each \( x_i \) a momentum \( p_i \). Define the Hamiltonian

\[
H(p, x) = E(x) + \frac{\alpha}{2} \sum_i p_i^2
\]

\[
P_H(p, x) = \frac{1}{Z_H} \exp \left( -E(x) - \frac{\alpha}{2} \sum_i p_i^2 \right)
\]

The marginal is \( p(x) = \int dp P_H(p, x) \).
Sample from $P_H$ using the Hamiltonian dynamics and Metropolis-Hasting:

$$\frac{dp_i}{dt} = -\frac{\partial H}{\partial x_i} \quad \frac{dx_i}{dt} = \frac{\partial H}{\partial p_i}$$

leaves $H$ invariant.

$$\frac{dH}{dt} = \sum_i \frac{dH}{dp_i} \frac{dp_i}{dt} + \frac{dH}{dx_i} \frac{dx_i}{dt} = 0$$
Ch. 30.1

Pseudo code

Choose initial $x^1$.

For $t = 1 : T$:

1. choose $p^t$ from $\mathcal{N}(0, \alpha^{-1})$, giving $(x^t, p^t)$

2. run Hamilton dynamics, giving $(x', p')$

3. Metropolis step: accept $(x^{t+1}, p^{t+1}) = (x', p')$ as new state with probability

$$
\min(1, a) \quad a = \frac{P_H(x', p')}{P_H(x, p)} = \frac{e^{-H(x', p')}}{e^{-H(x, p)}}
$$

4. On rejection, $(x^{t+1}, p^{t+1}) = (x^t, p^t)$

NB: $a = 1$ by construction in theory, but deviates from 1 due to numerical integration errors.
Pseudo code

```
g = gradE (x);  # set gradient using initial x
E = findE(x);  # set objective function too

for l = 1:L  # loop L times
  p = randn(size(x));  # initial momentum is Normal(0,1)
  H = p' * p / 2 + E;  # evaluate H(x,p)

  xnew = x;  gnew = g;
  for tau = 1:tau  # make Tau ‘leapfrog’ steps
    p = p - epsilon * gnew / 2;  # make half-step in p
    xnew = xnew + epsilon * p;  # make step in x
    gnew = gradE(xnew);  # find new gradient
    p = p - epsilon * gnew / 2;  # make half-step in p
  endfor

  Enew = findE(xnew);  # find new value of H
  Hnew = p' * p / 2 + Enew;
  dH = Hnew - H;  # Decide whether to accept

  if (dH < 0) accept = 1;
  elseif (rand() < exp(-dH)) accept = 1;
  else accept = 0;
  endif

  if (accept)
    g = gnew;  x = xnew;  E = Enew;
  endif
endfor
```
Example

Consider the double well cost $E(x) = \left(x^2 - 1\right)^2$.

The Hamiltonian $H(x, p) = E(x) + \frac{1}{2} \alpha p^2$ has the form.

$\alpha$ large yields large $p$ values and dynamical trajectories to large distances.
Example 2

\[ H(x, p) = \frac{1}{2} \beta_1 x_1^2 + \frac{1}{2} \beta_2 x_2^2 + \frac{1}{2} (p_1^2 + p_2^2) \]

The dynamics is

\[
\begin{align*}
\dot{x}_i &= \frac{\partial H(x, p)}{\partial p_i} = p_i \\
\dot{p}_i &= -\frac{\partial H(x, p)}{\partial x_i} = -\beta_i x_i \\
\ddot{x}_i &= -\beta_i x_i
\end{align*}
\]

The solution is of the form \( x_i(t) = \sin \left( \sqrt{\beta_i} t \right) \).

If \( \beta_1 \gg \beta_2 \), oscillations in \( x_1 \) much faster than in \( x_2 \).
Comparison of HMC and Metropolis

Two dimensional elongated Gaussian distribution. a-b) Hybrid Monte Carlo method c-d) Metropolis method.
Perceptron/logistic regression

\[ p(t = 1|x, w) = \sigma(\vec{w} \cdot \vec{x}) \]

\[ \sigma(x) = \frac{1}{1 + \exp(-2x)} \]

\[ \vec{w} \cdot \vec{x} = w_0 + w_1 x_1 + w_2 x_2 \]

\[ \mathbf{w} = (0, 2) \]
Perceptron/logistic regression

- $w = (-2, 3)$
- $w = (0, 2)$
- $w = (1, 0)$
- $w = (2, -2)$
- $w = (2, 2)$
- $w = (3, 0)$
- $w = (5, 1)$
- $w = (5, 4)$
- $w = (1, 4)$
Learning as inference

Data set: \( \{ x^\mu, t^\mu \}, \mu = 1, \ldots, P \) with \( t^\mu = \pm 1 \).

Probability of data point under the model: \( p(t^\mu|x^\mu, w) = \sigma(t^\mu w \cdot x^\mu) \)

Likelihood:

\[
p(D|w) = \prod_\mu p(t^\mu|x^\mu, w) = \exp(-G(w)) \quad G(w) = - \sum_\mu \log(p(t^\mu|x^\mu, w))
\]

Prior:

\[
p(w) = \frac{\exp(-\alpha E_w(w))}{Z_w(\alpha)} \quad E_w(w) = \sum_i w_i^2
\]

makes solutions with small weights more probable.

Posterior:

\[
p(w|D) = \frac{p(D|w)p(w)}{p(D)} \propto \exp(-M(w))
\]

\[
M(w) = G(w) + \alpha E_w(w)
\]
ML versus Bayesian

Standard in neural network learning is to compute the maximum likelihood or maximum posterior solution. For new test point $a$

$$D \rightarrow w_{ml}$$

$$p(t|a) = p(t|a, w_{ml})$$

Bayesian approach requires integration over multiple solutions:

$$D \rightarrow p(w|D)$$

$$p(t|a) = \int dw p(w|D)p(t|a, w) = \langle p(t|a, w) \rangle_{p(w|D)}$$
The maximum posterior solution

Minimizing the cost function $M(w)$ yields smoother solutions for larger $\alpha$. 
The full Bayesian solution

<table>
<thead>
<tr>
<th>Data set</th>
<th>Likelihood</th>
<th>Probability of parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N = 0$</td>
<td>(constant)</td>
<td></td>
</tr>
<tr>
<td>$N = 2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$N = 4$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$N = 6$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The full Bayesian solution

\[ \alpha = 0.01, \quad q(w'|w) = N(w'|w, \sigma), \quad \sigma = 0.1 \]

\[ p(t|x) = \int dw p(t|x, w)p(w|D) \approx \frac{1}{R} \sum_r p(t|x, w^r) \]
Ch. 31 Ising model

Ising model is a probability distribution over vectors of binary variables. \( x = (x_1, \ldots, x_N) \) with \( x_i = \pm 1 \).

\[
p(x) = \frac{1}{Z} \exp(-\beta E(x)) \quad E(x) = -\frac{1}{2} \sum_{i \neq j} J_{ij} x_i x_j - \sum_i h_i x_i
\]

\[
Z = \sum_x \exp(-\beta E(x)) = \sum_{x_1} \cdots \sum_{x_N} \exp(-\beta E(x))
\]

\( \beta = 1/kT \) is the inverse temperature, can be absorbed in the coupling.\(^2\)

Why study the Ising model:
- phase transition: qualitative change when going from high temperature to low temperature
- relation to attractor neural networks (hopfield models) and Boltzmann Machines

\(^2\)It is sometimes convenient to write \( \frac{1}{2} \sum_{i \neq j} J_{ij} x_i x_j = \sum_{(ij)} J_{ij} x_i x_j \).
Some basic relations

The mean energy of the spin system $\langle E \rangle = \sum_x E(x)p(x)$ can be computed from $Z$:

$$\frac{\partial \log Z}{\partial \beta} = \frac{1}{Z} \frac{\partial Z}{\partial \beta} = \frac{1}{Z} \sum_x -E(x) \exp(-\beta E(x)) = -\langle E \rangle$$

The energy fluctuations are computed similarly:

$$\frac{\partial^2 \log Z}{\partial \beta^2} = \frac{\partial}{\partial \beta} \left( \frac{1}{Z} \frac{\partial Z}{\partial \beta} \right) = -\frac{1}{Z^2} \left( \frac{\partial Z}{\partial \beta} \right)^2 + \frac{1}{Z} \frac{\partial^2 Z}{\partial \beta^2}$$

$$= -\langle E \rangle^2 + \frac{1}{Z} \sum_x E(x)^2 \exp(-\beta E(x)) = -\langle E \rangle^2 + \langle E^2 \rangle = \nabla(E)$$

One calls $F = -\frac{1}{\beta} \log Z$ the free energy, or a log partition sum.

$$F = \langle E \rangle - \frac{1}{\beta} H \quad \text{ex.31.1}$$
Metropolis Hasting for the Ising model

Use MH to sample from the Ising model

\[ p(x) = \frac{1}{Z} \exp(-E(x)) \]
\[ E(x) = -\sum_{(ij)} w_{ij} x_i x_j \]

Consider \( q(x'|x) \) to implement single spin flips. Define \( F_i x \) is the vector of spins obtained by flipping bit \( i \). Then

\[ q(F_i x|x) = \frac{1}{n} \quad q(x|x) = 0 \]

The MH ratio for the transition from state \( x \) to state \( F_i x \) is

\[ a_{x,F_i x} = \frac{p_i^*(F_i x)}{p_i^*(x)} = \exp\left(-\Delta E_{x,F_i x}\right) \]
\[ \Delta E_{x,F_i x} = E(F_i x) - E(x) = 2x_i \sum_{j \neq i} w_{ij} x_j \]
Ferro-magnet

Rectangular grid with $J = 1$ and $h_i = 0$ Ferro-magnet, periodic boundary.

Run system at different temperatures $\beta = 0.03 \rightarrow 10 \rightarrow 0.03$ (check for hysteresis)
Run $T$ iterations at each temperature, discard first $1/3$ for 'burn in'.
record $\langle E \rangle, \text{var}(E), \langle m^2 \rangle$ as time averages:

\[
\langle E \rangle = \frac{1}{T} \sum_{t=1}^{T} E(x_t)
\]

\[
\langle m^2 \rangle = \frac{1}{T} \sum_{t=1}^{T} m^2(x_t) \\
m(x_t) = \frac{1}{N} \sum_{i=1}^{N} x_i
\]
Ferro-magnet

Left: Mean energy. Each spin has 4 links, thus $E$ has $2N$ terms.
- at low temperature $\langle E \rangle / N = -2 \langle x_i x_j \rangle = -2$ (for one of the two ground states).
- At high temperature $\langle E \rangle / N = -2 \langle x_i x_j \rangle \approx 0$ and $\nabla (E / N) \propto 1 / N$.
- at intermediate temperature $\nabla (E / N)$ increases with $N$

Right: Mean squared magnetisation
- is zero for high temperature
- and 'breaks' to $\pm 1$ for low temperature.
Ferro-magnet

Energy fluctuations increase around the critical temperature
Anti-ferromagnet

Anti-ferromagnet has $J = -1$ between neighbors on the grid.

Ground states of the anti-ferromagnet are one-to-one with ground states in ferromagnet.

We can divide the spins into two groups $B, W$ according to the checker board.

\[
E_{\text{anti}}(x_B, x_W) = - \sum_{i \in B} \sum_{j \in W} (J = -1) x_i x_j = - \sum_{i \in B} \sum_{j \in W} (J = 1) x_i (x_j' = -x_j) \\
= E_{\text{ferro}}(x_B, -x_W)
\]

The ferro and anti-ferro magnetic system have the same distribution of energies $E(x)$ by relabeling $x$. 
Anti-ferromagnet

When the number of spins per direction is odd, this is no longer true.

Rectangular grid of 5 × 5 spins with ferromagnetic (left) and anti-ferromagnetic (right) interaction.

Not all bonds $J_{x_i x_j}$ can be minimized. This is a boundary effect that decreases with system size.
Frustration: Anti-ferromagnetic triangular Ising model

When $J = -1$, $E(x)$ cannot be minimized by minimizing all link energies $-Jx_i x_j = x_i x_j$.

Of the eight possible configurations of three spins, six have energy $-1$ and two have energy $+3$.

As a result: no 'clear cut' unique minimal energy state(s) but many states with similar low energy (Spin glass).
Frustration: Anti-ferromagnetic triangular Ising model

Energy (top), fluctuations (middle), heat capacity (bottom) for ferromagnet $J = 1$ (left) and anti-ferromagnet ($J = -1$) on triangular lattice.

Note: different low energies due to frustration, absence of peak in $\nabla(E)$ indicates no phase transition to state with long range order.
Direct computation of partition function

The summation of the partition sum $Z$ can be performed in the 2-dimensional Ising model on a lattice of size $W \times C$.

$$Z = \sum_x \exp(-\beta E(x))$$

$$= \sum_{s_1} \sum_{s_2} \ldots \sum_{s_C} \exp\left(-\beta \sum_{c=1}^C E(s_c, s_{c+1})\right)$$

$s_c$ is a binary vector of length $W$, $\sum s_c$ contains $2^W$ terms. The energy is

$$E(s_c, s_{c+1}) = - \sum_{i \in c, j \in c+1} J x_i x_j - \frac{1}{4} \sum_{i,j \in c} J x_i x_j - \frac{1}{4} \sum_{i,j \in c+1} J x_i x_j$$

$$Z = \sum_{s_1} \sum_{s_2} \ldots \sum_{s_C} M_{s_1,s_2} M_{s_2,s_3} \ldots M_{s_{C-1},s_C} M_{s_C,s_1} = \sum_{s_1} \left( M^C \right)_{s_1,s_1}$$

$$= \text{Trace}(M^C) \quad M_{s_c,s_{c+1}} = \exp(-\beta E(s_c, s_{c+1}))$$

$M$ is $2^W \times 2^W$ matrix.
Direct computation of partition function

\[
\lim_{C \to \infty} Z = \lim_{C \to \infty} \text{Trace}(M^C) = \lim_{C \to \infty} \sum_{\alpha} \mu\alpha^C = \mu_{\text{max}}^C
\]

with \(\mu_{\text{max}}\) the largest eigenvalue of \(M\).

Free energy per spin

\[
f = -\frac{1}{\beta WC} \log Z = -\frac{1}{\beta W} \log \mu_{\text{max}}
\]

1-3: \(F = \langle E \rangle - \frac{1}{\beta} H\), high temperature \(\langle E \rangle \approx 0\) and \(f \approx -\frac{1}{\beta} H \approx -\frac{1}{\beta} \log 2\). Entropy \(H = -\frac{\partial F}{\partial T}\), (Ex. 31.1) low temperature slope is zero for ferro and non-zero for triang. anti-ferro.

4: Mean energy of frustrated system at low temperature is higher.
Direct computation of partition function

\[ \text{var}(E) = \frac{\partial^2 \log Z}{\partial \beta^2} \]

versus temperature shows signs of phase transition for rectangular and triangular ferromagnet, but not for the frustrated triangular anti-ferromagnet.
Simulated annealing

Approximate an optimization problem by a sequence of sampling problems:

$$\min_x E(x) \rightarrow p(x) = \frac{1}{Z} e^{-E(x)/T}$$

Initially use large $T$ and decrease to $T = 0$ (‘cooling’ from high to low temperature).
Exercise Simulated annealing

Write your own SA algorithm to minimize $E(x) = -\frac{1}{2}x'wx = -\sum_{i,j} x_i x_j w_{ij}$.

If $w_{ij} > 0$ random, optimization is easy (ferromagnetic). There are two global minima.
If $w_{ij}$ has both signs, optimization is hard (spin glass). There are many minima of approx equal quality.

Initial temperature is set $T_0 = \max(dE)$, where $dE = E(x') - E(x)$ when $x \rightarrow x'$. As a result,

$$a = \frac{p^*(x')}{p^*(x)} = \exp(-dE/T_0) > \exp(-1)$$

$\beta_0 = 1/T_0$. Use $T = 1000$ samples at each temperature.

while $\text{std}(E) > 0$ do
- generate $x_{1:T}$. Compute $E(x_{1:T})$.
- Compute mean and std of $E$.
- $\beta = 1.05\beta$
Variational Inference

Outline

• Laplace approximation
  – application to Bayesian posterior

• variational methods
  – binary variables: Ising model, image denoising, dense networks
  – general case, structured approximations
  – approximate the Bayesian posterior for a Gaussian problem

• Belief propagation
Laplace approximation

The Laplace approximation approximates a given a probability density $p(x)$ by a Gaussian distribution centered on the maximum $x^* = \text{argmax}_x p(x)$.

Without loss of generality we can write $p(x) = \frac{1}{Z} e^{-E(x)}$. Expand $E$ around $x^*$ to second order:

$$E(x) \approx E(x^*) + \sum_i (x_i - x_i^*) \frac{\partial E(x)}{\partial x_i} \bigg|_{x=x^*} + \frac{1}{2} \sum_{ij} (x_i - x_i^*)(x_j - x_j^*) \frac{\partial^2 E}{\partial x_i \partial x_j} \bigg|_{x=x^*}$$

Note, that $\frac{\partial E(x)}{\partial x_i} \bigg|_{x=x^*} = 0$ and define $H_{ij} = \frac{\partial^2 E}{\partial x_i \partial x_j} \bigg|_{x=x^*}$. Then

$$p(x) \approx \frac{1}{Z} e^{-E(x^*)} e^{-\frac{1}{2}(x-x^*)^T H (x-x^*)} = \frac{1}{\sqrt{\det(2\pi H^{-1})}} e^{-\frac{1}{2}(x-x^*)^T H (x-x^*)}$$

The Laplace approximation is not necessarily the 'best' approximation (for instance in terms of KL divergence), but is easy to compute.

$$3Z = \int dx e^{-E(x)} \approx e^{-E(x^*)} \int e^{-\frac{1}{2}(x-x^*)^T H (x-x^*)} = e^{-E(x^*)} \sqrt{\det(2\pi H^{-1})}$$
Laplace approximation for Bayesian posterior

Consider the Bayesian learning of the perceptron discussed in MK 39 and MK 41:

\[ p(D|w) = \exp(-G(w)) \quad G(w) = - \sum_{\mu} \log(p(t^\mu|x^\mu, w)) \]

\[ p(w|\alpha) \propto \frac{\exp(-\alpha \sum_i w_i^2)}{Z_w(\alpha)} \]

\[ p(w|D, \alpha) = \frac{p(D|w)p(w)}{p(D)} = \frac{1}{Z} \exp(-E(w)) \quad E(w) = G(w) + \alpha \sum_i w_i^2 \]

We wish to compute the Bayesian prediction the probability of class \( t = 0, 1 \) for a new data point

\[ p(t|x, D, \alpha) = \int dw p(t|x, w)p(w|D, \alpha) \]

\[ ^4 \text{We use } t = 0, 1 \text{ and } p(t = 1|x, w) = \sigma(w \cdot x) \text{ with } \sigma(x) = 1/(1 + e^{-x}). \]
Laplace approximation for Bayesian posterior

We use the Laplace approximation

\[ p(t|x, D, \alpha) \approx \int dw p(t|x, w) \mathcal{N}(w|w^*, H) \]

with \( \mathcal{N}(w|w^*, H) \) a Gaussian centered on the maximum posterior solution \( w^* = \text{argmin}_w E(w) \). \( w^* \) can be found by gradient descend.

The Hessian is

\[
H_{ij} = \left. \frac{\partial^2 E}{\partial w_i \partial w_j} \right|_{w=w^*} = \sum_{\mu} \sigma(h^\mu)\sigma(-h^\mu) x_i^\mu x_j^\mu + 2\alpha \delta_{ij} \quad h^\mu = \sum_{i=0}^{d} w^*_i x_i^\mu
\]

\( H^{-1} \) is the covariance matrix of the posterior on \( w \) in the Gaussian approximation, which we can interpret as error bars on \( w \). Note, that \( H^{-1} \to 0 \) when the number of data \( p \to \infty \).

The function \( p(t=1|x, w) = \sigma(a) \) with \( a = w \cdot x \). Since \( w \) is Gaussian, so is \( a \) and

\[
\mathbb{E}(a) = w^* \cdot x = a^* \quad \mathbb{V}(a) = x^T \mathbb{V}(w)x = x^T H^{-1} x = s^2
\]
Laplace approximation for Bayesian posterior

Thus the integral is reduced to a one dimensional integral

\[ p(t = 1|x, D, \alpha) \approx \int da \sigma(a) N(a|a^*, s^2) \]

The last step is to note that the Gaussian integral of a sigmoid function

\[ \psi(a^*, s^2) = \int da \sigma(a) N(a|a^*, s^2) \approx \sigma(\kappa(s)a^*) \quad \kappa(s) = 1 / \sqrt{1 + \pi s^2 / 8} \]

Left: \( \psi(a^*, s^2) \). Right: \( \psi(a^*, s^2 = 4) \) and \( \sigma(\kappa(2)a^*) \) versus \( a^* \).
Laplace approximation for Bayesian posterior

(a) A projection of the Gaussian approximation onto the \((w_1, w_2)\) plane of weight space. The one- and two-standard-deviation contours are shown. Also shown are the trajectory of the optimizer, and the Monte Carlo methods samples. (b) The predictive function obtained from the Gaussian approximation and the further approximation for the sigmoid integral (41.30).
Variational methods

Recall the Kulback-Leibler divergence (aka relative entropy) between two distributions $p, q$ over the same domain $\mathcal{A}$:

$$KL(q|p) = \sum_{x \in \mathcal{A}} q(x) \log \frac{q(x)}{p(x)}$$

Properties:

$$KL(q|p) \neq KL(p|q) \quad KL(q|p) \geq 0 \quad KL(q|p) = 0 \iff p = q$$

Often, distributions are of the form $p(x) = \frac{p^*(x)}{Z}$, where $p^*(x)$ is easy to evaluate for any $x$, but $Z = \sum_x p^*(x)$ is intractable.  

Efficient approximation of $Z$ and other 'statistics' of interest.

---

For instance, the Ising model:

$$p(x) = \frac{1}{Z} \exp (-\beta E(x)) \quad E(x) = -\frac{1}{2} \sum_{i \neq j} J_{ij} x_i x_j - \sum_i h_i x_i$$

$$Z = \sum_x \exp (-\beta E(x))$$
Jensen’s inequality

A function $f$ is convex if

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda) f(x_2)$$

This generalises to

$$f\left(\sum_{i=1}^{n} \lambda_i x_i\right) \leq \sum_{i=1}^{n} \lambda_i f(x_i) \quad \sum_i \lambda_i = 1$$

$$f\left(\sum_x q(x)x\right) \leq \sum_x q(x)f(x) \quad f(\mathbb{E}x) \leq \mathbb{E}f(x)$$

The property $KL(p|q) \geq 0$ follows from the Jensen inequality and the convexity of $-\log x$ (exercise).
Variational methods

Given some distribution \( p(x) = \frac{1}{Z} p^*(x) \). We wish to approximate \( p \) by \( q \), where \( q \) is a simpler distribution. We do this by minimizing

\[
KL(q|p) = \langle \log q \rangle_q - \langle \log p \rangle_q = \langle \log q \rangle_q - \langle \log p^* \rangle_q + \log Z \geq 0
\]

\[
\log Z \geq -\langle \log q \rangle_q + \langle \log p^* \rangle_q
\]

with respect to \( q \). Typically, \( q \) is a simple parametrized distribution and the optimal solution has \( KL(q|p) > 0 \).

In physics the notation is \( p^*(x) = e^{-E(x)} \) with \( E \) the energy. Then \( F = -\log Z \) is called the free energy and \( \tilde{F}(q) = \langle \log q \rangle_q + \langle E \rangle_q \) is called the variational free energy.

Minimizing \( KL(q|p) \) is equivalent to minimizing \( \tilde{F}(q) \):

\[
KL(q|p) = \tilde{F}(q) + \log Z
\]
Variational methods

We can use the variational bound to approximate the partition sum of a spin model

\[ p(x) = \frac{1}{Z} e^{-E(x)} \quad Z \approx e^{-\tilde{F}(q)} \quad \left( Z \geq e^{-\tilde{F}(q)} \right) \]

We can then also use \( q \) to approximate the statistics of \( p \) (ie. \( \langle x_i \rangle_p \approx \langle x_i \rangle_q \) etc.)

or to bound the evidence in Bayesian model comparison

\[ p(w|D) = \frac{p(D|w)p(w)}{p(D)} \quad p(D) \approx e^{-\tilde{F}(q)} \]
**Difference $KL(q|p)$ and $KL(p|q)$**

We illustrate the difference between $KL(q|p)$ and $KL(p|q)$ with

$$p(x) = \frac{1}{2} \mathcal{N}(x| - \mu, \sigma^2) + \frac{1}{2} \mathcal{N}(x| \mu, \sigma^2) \quad q(x) = \mathcal{N}(x|m, s^2)$$

with $\mu \ll \sigma$ (well separated Gaussians).

Minimizing $KL(q|p)$ finds one of two solution $m \approx \pm \mu$ and $s^2 = \sigma^2$.

Minimizing $KL(p|q)$:

$$KL(p|q) = \langle \log p \rangle_p - \langle \log q \rangle_p = \frac{1}{2s^2} \langle (x - m)^2 \rangle_p + \frac{1}{2} \log s^2$$

The minimum is $m = \langle x \rangle_p$ and $s^2 = \langle (x - m)^2 \rangle_p$. This solution is known as moment matching: the distribution $q$ is such that its (first and second) moments match those of $p$. 
The (naive) mean field approximation for spin system

Consider \( x = (x_1, \ldots, x_n) \) with \( x_i = \pm 1 \) and

\[
p(x) = \frac{1}{Z} \exp(-E(x)) \quad E(x) = -\sum_{i>j} w_{ij} x_i x_j - \sum_i h_i x_i \quad Z = \sum_x \exp(-E(x))
\]

The simplest possible \( q \) is factorized:

\[
q(x) = \prod_{i=1}^n q_i(x_i) \quad q_i(x) = \frac{\exp(a_i x)}{\exp(a_i x) + \exp(-a_i x)} = \sigma(a_i x)
\]

The variational free energy \( \tilde{F}(q) = \langle E \rangle_q - S \).

\[
\langle x_i \rangle_q = q_i(1) - q_i(-1) = \tanh(a_i) \equiv m_i \quad \langle x_i x_j \rangle_q = \langle x_i \rangle \langle x_j \rangle = m_i m_j
\]

\[
\langle E \rangle_q = -\sum_{(ij)} w_{ij} \langle x_i x_j \rangle_q - \sum_i h_i \langle x_i \rangle_q = -\sum_{(ij)} w_{ij} m_i m_j - \sum_i h_i m_i
\]

with \( \langle \cdot \rangle = \langle \cdot \rangle_q \). We can express \( q_i(x) = \frac{1}{2}(1 + m_i x_i) \).
The (naive) mean field approximation for spin system

\[ S = - \sum_x q(x) \log q(x) = - \sum_x q(x) \sum_{i=1}^n \log q_i(x_i) \]

\[ = - \sum_{i=1}^n \sum_x q(x) \log q_i(x_i) = - \sum_{i=1}^n \sum_{x_i} q_i(x_i) \log q_i(x_i) \]

\[ \sum_{x_i} q_i(x_i) \log q_i(x_i) = \frac{1}{2} \left( (1 + m_i) \log \frac{1}{2}(1 + m_i) + (1 - m_i) \log \frac{1}{2}(1 - m_i) \right) \]

\[ \sum_x q(x)f(x) = \sum_{x_i} q(x, x_{-i}) f(x_i) = \sum_{x_i} q(x_i) f(x_i) = \sum_{x_i} q_i(x_i) f(x_i) \]
The variational free energy is a function of $m_1:n$:

$$
\tilde{F}(m) = -\sum_{(ij)} w_{ij}m_im_j - \sum_i h_im_i \\
+ \sum_i \frac{1}{2}\left((1 + m_i) \log \frac{1}{2}(1 + m_i) + (1 - m_i) \log \frac{1}{2}(1 - m_i)\right)
$$

We find the minimum by differentiation:

$$
\frac{\partial \tilde{F}(m)}{\partial m_i} = -\sum_j w_{ij}m_j - h_i + \frac{1}{2} \log \frac{1 + m_i}{1 - m_i}
$$

It is easy to show that $\frac{1}{2} \log \frac{1 + m_i}{1 - m_i} = a_i = \tanh^{-1}(m_i)$. Thus,

$$
m_i = \tanh \left( \sum_j w_{ij}m_j + h_i \right)
$$

are the mean field equations.
Image denoising

Consider a binary image $y$ generated by corrupting a clean image $x$; our interest is to recover the clean image given the corrupted image.

A noisy pixel generating process takes each clean pixel $x_i \in \pm1$ and flips:

$$p(y|x) = \prod_i p(y_i|x_i), \quad p(y_i|x_i) \propto e^{\gamma y_i x_i}$$

Clean images are reasonably smooth and can be described using a MRF:

$$p(x) \propto e^{\sum_{i \sim j} w_{ij} x_i x_j}$$

for settings of $w_{ij} > 0$, for neighbouring $i$ and $j$, and $w_{ij} = 0$ otherwise.

$$p(x, y) = p(x) \prod_i p(y_i|x_i) = \frac{1}{Z} e^{\sum_{i \sim j} w_{ij} x_i x_j + \sum_i \gamma y_i x_i}$$

The posterior is given by

$$p(x|y) = \frac{p(y|x)p(x)}{\sum_x p(y|x)p(x)} \propto e^{\sum_{i \sim j} w_{ij} x_i x_j + \sum_i \gamma y_i x_i}$$

Interested in $p(x_i|y)$ and $Z$ – useful in model comparison.
Image Restoration

Provide noisy image \( y \). Compute mean field approximation \( \langle x_i \rangle \). Threshold these values to 0, 1. Alternative is to use Iterated Conditional Modes (BRML 28.9)
The (naive) mean field approximation the Ising model

In the case of the 2d Ising model, all couplings $w_{ij} = w$ for neighbouring spins and zero otherwise and the external field $h_i = h$. The variational distribution has now only a single variable:

$$q(x|a) = \frac{1}{Z} \exp(\sum_i ax_i) \quad m = \tanh(a)$$

The mean field equation becomes

$$m = \tanh(Cwm + h)$$

with $C$ the number of neighbours in the lattice (4 for square lattice).

MF estimate of critical temperature is $T_c = 4$, exact value is $T_c = 2.27$

For $T > T_c$ there $\tilde{F}(m)$ is convex and has one minimum, for $T < T_c$ $\tilde{F}(m)$ is not convex and has 2 minima.
The (naive) mean field approximation the Ising model

Comparison of MF approximation to exact solution for $8 \times 8$ ferromagnetic Ising model. MF free energy is higher than true free energy. Energy fluctuations are not well predicted.

Mean field approximation improves when number of connections increase, and becomes exact for the fully connected system in the large $N$ limit.
Mean field approximation works well for dense models

Consider a distribution over $d$ binary variables

$$p(x) = \frac{1}{Z} \exp \left( \sum_{ij} w_{ij} x_i x_j \right)$$

If each variable is connected to a large number of neighbors $O(d)$, the individual couplings must scale as $O(d^{-1})$. In this case $p(x)$ will be approximately factorized when $d \to \infty$.

The argument is that the independency assumption is satisfied selfconsistently. In the following way.

Define $z_i = \sum_j w_{ij} x_j$. We then have

$$\langle z_i \rangle = \sum_j w_{ij} \langle x_j \rangle = O(1) \quad \mathbb{V}(z_i) = \sum_j \mathbb{V}(w_{ij} x_j) = O(d^{-1})$$

Therefore $z_i \approx \langle z_i \rangle$ for large $d$. 
Thus

\[ p(x) = \frac{1}{Z} \exp \left( \sum_i x_i z_i \right) \approx \frac{1}{Z} \exp \left( \sum_i x_i \langle z_i \rangle \right) = \prod_i p_i(x_i) \]
General mean field equations

In most applications of the variational approximation \(q\) is factorized: \(q(x) = \prod_i q_i(x_i)\).

The KL divergence is

\[
KL(q|p) = \langle \log q \rangle_q - \langle \log p \rangle_q = \sum_i \sum_{x_i} q_i(x_i) \log q_i(x_i) - \sum_x \prod_i q_i(x_i) \log p(x)
\]

Differentiating with respect to \(q_i(x_i)\) we get

\[
\frac{\partial KL(q|p)}{\partial q_i(x_i)} = \log q_i(x_i) + 1 - \sum_{x \setminus i} \prod_{j \neq i} q_j(x_j) \log p
\]

Remembering the Lagrange multiplier to ensure that \(q_i\) is normalized, the solution is thus

\[
q_i(x_i) \propto \exp\left(\langle \log p \rangle_{\prod_{j \neq i} q_j}\right)
\]

\(^7\)Sums are replaced by integrals if \(x\) is continuous
NB one can replace $p(x)$ by its unnormalized version $p^*(x)$ to compute its mean field approximation.
Convergence of mean field equations using local updates

The sequential update scheme converges to a local minimum of $KL(q|p)$.

Proof:

$$ KL(q|p) = \sum_i \sum_{x_i} q_i(x_i) \log q_i(x_i) - \sum_x \prod_i q_i(x_i) \log p(x) $$

is a convex function in $q_i(x_i)$ for given $q_j(x_j)$, $j \neq i$. The normalization constraint is linear in $q_i(x_i)$.

Therefore updating $q_i(x_i)$ is equivalent to a minimization of $KL(q|p)$ with respect to $q_i(x_i)$ and thus

$$ KL(q_i^{\text{new}} \prod_{j \neq i} q_j^{\text{old}} | p) \leq KL(q_i^{\text{old}} \prod_{j \neq i} q_j^{\text{old}} | p) $$
(Non)-convexity of $KL(q|p)$

$KL(q|p) = \tilde{F}(m_1, m_2) = -wm_1m_2 - H(q_1) - H(q_2) \text{ vs. } \frac{1}{2}(1 + m_1), \frac{1}{2}(1 + m_1) \text{ for } E(x) = -wx_1x_2.$

$KL(q|p) = \tilde{F}(q_1, q_2)$ convex in $m_1$ for fixed $m_2$ and visa versa, but not convex in $(m_1, m_2)$. There may be more than one solution (local minima)
Structured Variational Inference

We need the entropy and energy to be tractable. For the binary MRF case, we can consider richer $q$ distributions. A richer $q$ gives a better approximation and a tighter bound.

**Classes of tractable $q$**

(a) diagram

(b) diagram

(c) diagram

**Figure:** (a): Naive Mean Field approximation $q(x) = \prod_i q_i(x_i)$. (b): A spanning tree approximation. (c): A hypertree approximation.
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,

\[
\langle s_i \rangle \approx m_i \quad \langle s_i s_j \rangle \approx m_i m_j + A_{ij}^{-1} \quad A_{ij} = \frac{\delta_{ij}}{1 - m_i^2} - w_{ij}
\]
Bayesian posterior for Gaussian

Hypothesis space for a Gaussian distribution in 1 dimension:

\[ p(x|\mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \]
Bayesian posterior for Gaussian

Give some data, we can evaluate $p(\{x_n\}_{n=1}^N | \mu, \sigma)$
Bayesian posterior for Gaussian: ML solution

The log likelihood is

$$\log p({x_n}_{n=1}^N | \mu, \sigma) = -N \log (\sqrt{2\pi\sigma}) - \frac{1}{2\sigma^2} \sum_n (x_n - \mu)^2$$

The maximum likelihood solution is

$$\mu = \frac{1}{N} \sum_n x_n \quad \sigma^2 = \frac{1}{N} \sum_n (x_n - \mu)^2$$

$\mu$ independent of $\sigma$, but $\sigma$ depends on $\mu$. 

![Graphs and plots (a), (b), and (c)](image)
Bayesian posterior for Gaussian: Variational solution

In Bayesian learning we consider the posterior:

\[ p(w|D) = \frac{p(D|w)p(w)}{p(D)} \]

We define the variational approximation by introducing the variational distribution \( q(w) \) and minimize

\[
KL(q|p) = \tilde{F}(q) + \log Z
\]

\[
\tilde{F}(q) = \langle \log q \rangle_q - \langle \log p^* \rangle_q = \int dq(w) \log \frac{q(w)}{p(D|w)p(w)}
\]
Bayesian posterior for Gaussian: Variational solution

Given the Gaussian distribution \( \beta = \frac{1}{\sigma^2} \):  

\[
p(x|\mu, \beta) = \sqrt{\frac{\beta}{2\pi}} \exp\left(-\frac{1}{2}\beta(x - \mu)^2\right)
\]

posterior for \( N \) data points \( D = \{x_n\}_{n=1}^N \) is  

\[
p(\mu, \beta|D) \propto \left(\frac{\beta}{2\pi}\right)^{N/2} \exp\left(-\frac{1}{2}\beta(N(\mu - \bar{x})^2 + S)\right)\beta^{-1}
\]

with \( \bar{x} = \frac{1}{N} \sum_n x_n \), \( S = \sum_n (x_n - \bar{x})^2 \), flat prior on \( \mu \) and \( \beta^{-1} \) the ’non-informative prior’ on \( \beta \).

---

8 The non-informative prior is defined in the log domain. Define \( x = \log \sigma \) then  

\[
p_x(x) = p_\sigma(\sigma) \left| \frac{d\sigma}{dx} \right|
\]

When \( p_x(x) \) uniform, since \( \left| \frac{d\sigma}{d\sigma} \right| = \sigma^{-1} \) we obtain \( p_\sigma(\sigma) \propto \sigma^{-1} \). In terms of \( \beta \), we get \( p_\beta(\beta) \propto \beta^{-1} \). See also sheet on the Gamma distribution.
Bayesian posterior for Gaussian: Variational solution

We approximate the posterior by a factorized variational distribution $q_\mu(\mu)q_\beta(\beta)$.

$$\tilde{F}(q) = \int d\mu q_\mu(\mu) \log q_\mu(\mu) + \int d\beta q_\beta(\beta) \log q_\beta(\beta) - \int d\mu d\sigma q_\mu(\mu)q_\beta(\beta) \log p(\mu, \beta|D)$$

We optimize with respect to $q_\mu(\mu)$ subject to normalization constraint:

$$\frac{\partial \tilde{F}}{\partial q_\mu(\mu)} = \log q_\mu(\mu) + 1 - \int d\beta q_\beta(\beta) \log p(\mu, \beta|D)$$

$$= \log q_\mu(\mu) + \frac{1}{2} \bar{\beta}N(\mu - \bar{x})^2 + \text{const.}$$

with $\bar{\beta} = \int d\beta q_\beta(\beta) \beta$ and the constant term contains all terms independent of $\mu$.

Thus, the solution $q_\mu$ is a Gaussian distribution with mean $\bar{x}$ and variance $(N\bar{\beta})^{-1}$. 
We optimize with respect to $q_\beta(\beta)$ subject to normalization constraint:

$$\frac{\partial \tilde{F}}{\partial q_\beta(\beta)} = \log q_\beta(\beta) + 1 - \int d\mu q_\mu(\mu) \log p(\mu, \beta|D)$$

$$= \log q_\beta(\beta) - \left(\frac{N}{2} - 1\right) \log \beta + \frac{1}{2} \beta \left(\frac{1}{\beta} + \bar{S}\right) + \text{const.}$$

$$= \log q_\beta(\beta) - \log \left(\beta^{c-1} e^{-\beta/s}\right) + \text{const.}$$

with the constant term contains all terms independent of $\beta$. In the second step we used $\int d\mu q_\mu(\mu)(\mu - \bar{x})^2 = (N\bar{\beta})^{-1}$ and we defined $c = \frac{N}{2}$, $\frac{1}{s} = \frac{1}{2} \left(\frac{1}{\beta} + \bar{S}\right)$. Thus, the solution $q_\beta$ is a Gamma distribution $\Gamma(\beta|c, s)$ with parameters $c, s$. 
Closed form solution

We can solve for $\bar{\beta}$ self-consistently.

$$\bar{\beta} = \int d\beta \Gamma(\beta|c, s)\beta = cs = \frac{N}{\frac{1}{\beta} + S}$$

which has solution $\bar{\beta} = \frac{N-1}{S}$.

The final solution is

$$q_\mu(\mu) = \mathcal{N}\left(\mu|\bar{x}, \frac{S}{N(N-1)}\right)$$

$$q_\beta(\beta) = \Gamma(\beta|c, b) \quad c = \frac{N}{2} \quad \frac{1}{b} = \frac{NS}{2(N-1)}$$

Note, that the variational approximation has mean $\bar{\mu} = \bar{x}$ and $\bar{\sigma}^2 = \frac{S}{N-1}$. 
Bayesian posterior for Gaussian: Variational solution

Solid: $p(\mu, \sigma|D)$. Dashed: Iterative optimization of $q_\mu$ (b,d) and $q_\sigma$ (c,e) from initial (a) converges in 15 iterations (f)

The solution can also be obtained numerically by optimizing for $q_\mu$ for fixed $q_\beta$ and vise versa.
Gamma prior

The Gamma distribution is defined for positive quantities:

$$\Gamma(x|c, s) = \frac{1}{Z} \left(\frac{x}{s}\right)^{c-1} \exp\left(-\frac{x}{s}\right)$$

$$Z = \Gamma(c)s$$

When $y = \log x$ then

$$p(y) = p(x) \left|\frac{dx}{dy}\right| = \frac{1}{\Gamma(c)} \left(\frac{x(y)}{s}\right)^c \exp\left(-\frac{xx(y)}{s}\right)$$

Gamma distribution with $(s, c) = (1, 3)$ and $(s, c) = (10, 0.3)$. Top: linear vertical scale. Bottom: log vertical scale. Left: versus $x$. Right: versus log $x$.

$\Gamma(x|c, s)$ may be peaked at $x = 0$ this peak is absent in the log basis.

When $c \to 0$ with $sc = 1$.

$$\Gamma(x|c, s) = \frac{1}{Z} \left(\frac{x}{s}\right)^{c-1} \exp\left(-\frac{x}{s}\right) = \frac{1}{Z} c^{c-1} \left(\frac{x}{sc}\right)^{c-1} \exp\left(-\frac{x}{sc}\right)^c \propto x^{-1}$$

The non-informative prior has no scale $s$ and is 'flat' in the log basis.
Belief propagation

Consider a distribution

\[ p(x_1, x_2, x_3, x_4, x_5) = \frac{1}{Z} \phi(x_1, x_2) \phi(x_2, x_3) \phi(x_3, x_4) \phi(x_4, x_5) \]

and we wish to compute

\[ p(x_3) = \sum_{x_1, x_2, x_4, x_5} p(x_1, x_2, x_3, x_4, x_5) \]

\[ \propto \sum_{x_2} \sum_{x_1} \phi(x_1, x_2) \phi(x_2, x_3) \sum_{x_4} \sum_{x_5} \phi(x_4, x_5) \phi(x_3, x_4) = \mu_{2\to3}(x_3) \mu_{4\to3}(x_3) \]
Belief propagation

Define messages

\[ \mu_{i \to j}(x_j) = \sum_{x_i} \phi(x_i, x_j) \prod_{k \in \mathcal{N}_i \setminus j} \mu_{k \to i}(x_i) \]  

(1)

BP is exact on trees (graphs without loops).

Loopy belief propagation is the same algorithm on a loopy graph. Initialize \( \mu_{i \to j}(x_j) = 1 \) and iterate until convergence (which is not guaranteed!).
A more formal derivation: the Bethe free energy

[Yedidia et al., 2005]

Consider a binary spin model

\[ p(x) = \frac{1}{Z} \exp(-E(x)) \quad E(x) = -\sum_{(i,j)} w_{ij}x_i x_j - \sum_i w_i x_i \]

The variational free energy is

\[ F(q) = \langle E \rangle_q + \langle \log q \rangle_q \]

Since \( E \) contains single spin and pair spin terms only,

\[ \langle E \rangle_q = -\sum_{(i,j)} w_{ij} \langle x_i x_j \rangle_{q_{ij}} - \sum_i w_i \langle x_i \rangle_{q_i} \]

we only have to specify \( q_i(x_i), q_{ij}(x_i, x_j) \) to compute \( \langle E \rangle \).
The Bethe approximation

Exact computation of \( \langle \log q \rangle_q = \sum_x q(x) \log q(x) \) is intractable.

The Bethe approximation approximates the entropy by the sum of entropies of the pair interactions, minus the overcounting of the single spin terms. Denote \( a = (i, j) \), \( x_a = (x_i, x_j) \), \( q_a(x_a) = q_{ij}(x_i, x_j) \).

\[
\langle \log q \rangle_q \approx \sum_a \langle \log q_a \rangle_{q_a} - \sum_i (n_i - 1) \langle \log q_i \rangle_{q_i}
\]

\( n_i \) is the number of neighbor spins of spin \( i \).

Thus

\[
F_{Bethe}(\{q_i, q_a\}) = - \sum_a \langle x_a \rangle_{q_a} - \sum_i \langle x_i \rangle_{q_i} + \sum_a \langle \log q_a \rangle_{q_a} - \sum_i (n_i - 1) \langle \log q_i \rangle_{q_i}
\]

\( F \) is easy to evaluate.

Note, that \( F \) is a difference of convex functions and is therefore not convex, ie. it may have multiple local minima.
Optimizing the Bethe free energy

The solution is obtained by minimizing $F_{\text{Bethe}}(\{q_i, q_a\})$ with respect to all $\{q_i, q_a\}$ subject to the constraints

\[
q_i(x_i) = \sum_{x_j} q_{ij}(x_i, x_j) \quad \sum_{x_i} q_i(x_i) = 1 \quad \sum_{x_a} q_a(x_a) = 1
\]

yields the BP equations.
BP equations with external fields

The spin model can be written as

\[ p(x) = \frac{1}{Z} \exp \left( \sum_{(ij)} w_{ij} x_i x_j + \sum_i w_i x_i \right) = \frac{1}{Z} \prod_a \phi_a(x_a) \prod_i \phi_i(x_i) \]

\[ \phi_a(x_a) = e^{w_{ij} x_i x_j} \quad \phi_i(x_i) = e^{w_i x_i} \]

Define \( \phi'_a(x_a) = \phi_a(x_a) \prod_{i \in a} \phi_i(x_i)^{c_i} \). Then

\[ \prod_{a} \prod_{i \in a} \phi_i(x_i)^{c_i} = \prod_i \prod_{a \ni i} \phi_i(x_i)^{c_i} = \prod_i \phi_i(x_i)^{n_i c_i} \]

with \( n_i \) the number of interaction terms that spin \( i \) participates in. Thus, when \( c_i = 1/n_i \) we get

\[ p(x) = \frac{1}{Z} \prod_a \phi'_a(x_a) \]

and we can use the BP equations Eq. [1].
Further reading

Boltzmann Machines

The basic idea is to treat Boltzmann-Gibbs distribution of the Ising model 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 \( s_0 = 1 \).

\[
\frac{\partial \log p(s|w)}{\partial w_{ij}} = -\frac{\log Z}{\partial w_{ij}} + s_i s_j = -\left< s_i s_j \right> + s_i s_j
\]

\[
\frac{\partial L}{\partial w_{ij}} = \frac{1}{P} \sum_{\mu} \frac{\partial \log p(s^\mu|w)}{\partial w_{ij}} = -\left< s_i s_j \right> + \frac{1}{P} \sum_{\mu} s_i^{\mu} s_j^{\mu} = -\left< s_i s_j \right> + \left< s_i s_j \right>_c
\]
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:

\[
\langle s_i \rangle_c = \frac{1}{P} \sum_{i} s_i^\mu, \quad \langle s_is_j \rangle_c = \frac{1}{P} \sum_{i} s_i^\mu s_j^\mu
\]
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.
Neuron spike data from salamander retina

Schneidmann 2006
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

\[
C_{ij} = \langle s_i s_j \rangle_c - \langle s_i \rangle_c \langle s_j \rangle_c
\]

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 \( w \):

\[
\langle s_i s_j \rangle = \langle s_i s_j \rangle_c \quad \chi_{ij} = C_{ij}
\]

\[
\chi^{-1}_{ij} = C^{-1}_{ij} \quad w_{ij} = \frac{\delta_{ij}}{1 - m_i^2} - (C^{-1})_{ij}
\]
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.
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 $i, j \in x$:

$$\langle y_i y_j \rangle = \langle x_i x_j \rangle \quad \langle x_i x_j \rangle_c = \frac{1}{P} \sum_{\mu} x_i^\mu x_j^\mu$$

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 $i \in x, j \in h$:

$$\langle y_i y_j \rangle = \langle x_i h_j \rangle \quad \langle y_i y_j \rangle_c = \frac{1}{P} \sum_{\mu} x_i^\mu \langle h_j \rangle_\mu \quad \langle h_j \rangle_\mu = \sum_h p(h|x^\mu, w)h_j$$
Auto encoders [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
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 (8).

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

Summary

Boltzmann machines is a learnable Ising model.
- simple stochastic neural network model
- component in deep learning (RBM)

Learning by
- MCMC is generally (too) slow. Claimed to be faster for RBM using contrastive divergence
- Mean field learning is fast, but may be (too) inaccurate
Mining for Structure

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

- 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.

- sunset, pacific ocean, baker beach, seashore, ocean
Multimodal DBM

$h^3$

$h^2$

$h^1$

Gaussian model

Dense, real-valued image features

$v_{image}$

$V_{image}$

Replicated Softmax

Word counts

$v_{text}$

$V_{text}$

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

Given: dog, cat, pet, kitten, puppy, ginger, tongue, kitty, dogs, furry

Generated: insect, butterfly, insects, bug, butterflies, lepidoptera

Given: sea, france, boat, mer, beach, river, bretagne, plage, brittany

Generated: graffiti, streetart, stencil, sticker, urbanart, graff, sanfrancisco

Given: portrait, child, kid, ritratto, kids, children, boy, cute, boys, italy

Generated: canada, nature, sunrise, ontario, fog, mist, bc, morning
Text Generated from Images

**Given**
- portrait, women, army, soldier, mother, postcard, soldiers

**Generated**
- 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
Ch. 20: Clustering

Clustering: group things that a similar

- "brown things that run away" $\rightarrow$ "animals"
- "green things that dont run away" $\rightarrow$ "plants"

Clustering is good for prediction and communication

One type of clustering is called vector quantisation:

- find $k(\vec{x})$, $\vec{m}^k$ that encode each $\vec{x}$ in a label $k = 1, \ldots, K$, for instance minimize

$$D = \sum_{\vec{x}} P(\vec{x}) \frac{1}{2} \left( \vec{m}^k(\vec{x}) - \vec{x} \right)^2$$
**K-means clustering**

**Initialization.** Set $K$ means $\{m^{(k)}\}$ to random values.

**Assignment step.** Each data point $n$ is assigned to the nearest mean. We denote our guess for the cluster $k^{(n)}$ that the point $x^{(n)}$ belongs to by $\hat{k}^{(n)}$.

$$\hat{k}^{(n)} = \arg \min_k \{d(m^{(k)}, x^{(n)})\}.$$  \hspace{1cm} (20.3)

An alternative, equivalent representation of this assignment of points to clusters is given by ‘responsibilities’, which are indicator variables $r_k^{(n)}$. In the assignment step, we set $r_k^{(n)}$ to one if mean $k$ is the closest mean to datapoint $x^{(n)}$; otherwise $r_k^{(n)}$ is zero.

$$r_k^{(n)} = \begin{cases} 1 & \text{if } \hat{k}^{(n)} = k \\ 0 & \text{if } \hat{k}^{(n)} \neq k. \end{cases}$$  \hspace{1cm} (20.4)

*What about ties?* We don’t expect two means to be exactly the same distance from a data point, but if a tie does happen, $\hat{k}^{(n)}$ is set to the smallest of the winning $\{k\}$.

**Update step.** The model parameters, the means, are adjusted to match the sample means of the data points that they are responsible for.

$$m^{(k)} = \frac{\sum_n r_k^{(n)} x^{(n)}}{R^{(k)}}.$$  \hspace{1cm} (20.5)

where $R^{(k)}$ is the total responsibility of mean $k$,

$$R^{(k)} = \sum_n r_k^{(n)}.$$

*What about means with no responsibilities?* If $R^{(k)} = 0$, then we leave the mean $m^{(k)}$ where it is.

Repeat the assignment step and update step until the assignments do not change.
**K-means clustering**

Two clusters:

![Data Diagram](image)

![Assignment Diagram](image)

![Update Diagram](image)
$K$-means clustering

Results depend on initial condition

Run 1

Run 2

Run 1: convergence after 5 iterations; 4 equally sized clusters
Run 2: convergence after 6 iterations; 1 large cluster, 3 small ones
**K-means clustering**

Ad-hoc features:

- which distance to use?
- which value of $K$?
- $K$-means does not take 'weight' of cluster into account.

(a) 75 data points from two gaussians. Left cluster has 5 times more data than right cluster. (b) Clustering ($K = 2$) yields wrong assignments of data points (+’s).
**K-means clustering**

Ad-hoc features:

- which distance to use?
- which value of $K$?
- $K$-means does not take ’shape’ of cluster into account.

(a) two elongated clusters are clustered wrongly ($K = 2$ b).
Soft $K$-means clustering

**Assignment step.** Each data point $x^{(n)}$ is given a soft ‘degree of assignment’ to each of the means. We call the degree to which $x^{(n)}$ is assigned to cluster $k$ the responsibility $r_k^{(n)}$ (the responsibility of cluster $k$ for point $n$).

$$r_k^{(n)} = \frac{\exp \left( -\beta d(m^{(k)}, x^{(n)}) \right)}{\sum_{k'} \exp \left( -\beta d(m^{(k')}, x^{(n)}) \right)}. \quad (20.7)$$

The sum of the $K$ responsibilities for the $n$th point is 1.

**Update step.** The model parameters, the means, are adjusted to match the sample means of the data points that they are responsible for.

$$m^{(k)} = \frac{\sum_n r_k^{(n)} x^{(n)}}{R^{(k)}} \quad (20.8)$$

where $R^{(k)}$ is the total responsibility of mean $k$,

$$R^{(k)} = \sum_n r_k^{(n)}. \quad (20.9)$$

The update is identical. The only difference is that the responsibilities take values between 0 and 1.
Circles indicate length scale $\sigma = 1/\sqrt{\beta}$. Bifurcation as a function of $\beta$. See Bert Kappen ML 150
[Rose et al., 1992].
Exact inference for Gaussian mixture

\[
p(x|\pi_1, \pi_2, \mu_1, \mu_2, \sigma_1, \sigma_2) = \frac{\pi_1}{\sqrt{2\pi\sigma_1}} \exp\left(-\frac{(x - \mu_1)^2}{2\sigma_1^2}\right) + \frac{\pi_2}{\sqrt{2\pi\sigma_2}} \exp\left(-\frac{(x - \mu_2)^2}{2\sigma_2^2}\right)
\]

Weight of the mixture components is \(\pi_1, \pi_2 = 0.6, 0.4\) in the top half and \(0.8, 0.2\) in the bottom half. Means \(\mu_1\) and \(\mu_2\) vary horizontally, and standard deviations \(\sigma_1\) and \(\sigma_2\) vary vertically.
Exact inference for Gaussian mixture

\[ p(x|\pi_1, \pi_2, \mu_1, \mu_2, \sigma_1, \sigma_2) = \frac{\pi_1}{\sqrt{2\pi\sigma_1}} \exp\left( -\frac{(x - \mu_1)^2}{2\sigma_1^2} \right) + \frac{\pi_2}{\sqrt{2\pi\sigma_2}} \exp\left( -\frac{(x - \mu_2)^2}{2\sigma_2^2} \right) \]

Bayesian posterior is distribution over 5 dimensional continuous space.
**Maximum likelihood for mixture of Gaussians**

Mixture distribution of $K$ Gaussians

\[
p(x|\mu, \sigma) = \sum_{k=1}^{K} p_k \mathcal{N}(x|\mu_k, \sigma_k)
\]

$\mathcal{N}(x|\mu_k, \sigma_k)$ is the Normal distribution with mean $\mu_k$ and variance $\sigma_k^2$ and $\sum_k p_k = 1$.

Introduce hidden variable $k = 1, 2, \ldots, K$ and define

\[
p(x, k) = p_k \mathcal{N}(x|\mu_k, \sigma_k)
\]

then $p(x) = \sum_k p(x, k)$.

Given all parameters $\mu_k, \sigma_k, p_k$ we compute responsibilities for each data point:

\[
p_{k|n} = p(k|x_n) = \frac{p(x_n, k)}{p(x_n)}
\]

$0 \leq p_{k|n} \leq 1$ and $\sum_{k=1}^{K} p_{k|n} = 1$. 
Maximum likelihood for mixture of Gaussians

Define

\[ L = \sum_{n=1}^{N} \log p(x_n) \quad p(x_n) = \sum_k p(x_n, k) \]

Then

\[ \frac{\partial L}{\partial \mu_k} = \sum_n \frac{1}{p(x_n)} \frac{\partial p(x_n, k)}{\partial \mu_k} = \sum_n \frac{p(x_n, k)}{p(x_n)} \frac{\partial \log p(x_n, k)}{\partial \mu_k} = \sum_n p_{k|n} \frac{x_n - \mu_k}{\sigma_k^2} \]

\[ H_{kl} = \frac{\partial^2 L}{\partial \mu_k \partial \mu_l} \approx -\sum_n p_{k|n} \frac{1}{\sigma_k^2} \delta_{kl} \]

Using Newtons method the update becomes:

\[ \mu_k' = \mu_k - \sum_l H_{kl} \frac{\partial L}{\partial \mu_l} = \mu_k + \frac{\sum_n p_{k|n}(x_n - \mu_k)}{\sum_n p_{k|n}} = \frac{\sum_n p_{k|n} x_n}{\sum_n p_{k|n}} \]
Similar for $\sigma_k, p_k$

This is identical to the soft clustering algorithm.
Maximum likelihood for mixture of $K$ spherical Gaussians

Assignment step. The responsibilities are

$$r_k^{(n)} = \frac{\pi_k \frac{1}{(\sqrt{2\pi}\sigma_k)^I} \exp \left( -\frac{1}{\sigma_k^2} d(m^{(k)}, x^{(n)}) \right)}{\sum_{k'} \pi_{k'} \frac{1}{(\sqrt{2\pi}\sigma_{k'})^I} \exp \left( -\frac{1}{\sigma_{k'}^2} d(m^{(k')}, x^{(n)}) \right)}$$

(22.22)

where $I$ is the dimensionality of $x$.

Update step. Each cluster’s parameters, $m^{(k)}$, $\pi_k$, and $\sigma_k^2$, are adjusted to match the data points that it is responsible for.

$$m^{(k)} = \frac{\sum_n r_k^{(n)} x^{(n)}}{R^{(k)}}$$

(22.23)

$$\sigma_k^2 = \frac{\sum_n r_k^{(n)} (x^{(n)} - m^{(k)})^2}{IR^{(k)}}$$

(22.24)

$$\pi_k = \frac{R^{(k)}}{\sum_k R^{(k)}}$$

(22.25)

where $R^{(k)}$ is the total responsibility of mean $k$,

$$R^{(k)} = \sum_n r_k^{(n)}.$$  

(22.26)
Maximum likelihood for mixture of $K$ spherical Gaussians

Performance on same examples as before shows that size of cluster is properly found but requires more iterations.

$K = 2$ with soft $K$ max algorithm with spherical Gaussians works well
Maximum likelihood for mixture of $K$ axis-aligned Gaussians

\[
  r_k^{(n)} = \frac{\pi_k}{\prod_{i=1}^{I} \sqrt{2\pi \sigma_i^{(k)}}} \exp \left( -\sum_{i=1}^{I} \frac{(m_i^{(k)} - x_i^{(n)})^2}{2(\sigma_i^{(k)})^2} \right)
  \sum_{k'} (\text{numerator, with } k' \text{ in place of } k) \tag{22.27}
\]

\[
  \sigma_i^{(k)} = \frac{\sum_{n} r_k^{(n)} (x_i^{(n)} - m_i^{(k)})^2}{R^{(k)}} \tag{22.28}
\]
Maximum likelihood for mixture of $K$ axis-aligned Gaussians

Performance on same examples as before shows that size and shape of cluster is properly found.
Overfitting: When a cluster covers only 1 data point the likelihood can be made arbitrary large by shrinking the variance to zero.

The posterior density has infinite spikes, and its integrated probability is small.

Adding a prior and computing $\max_\theta p(\text{Data}|\theta)p(\theta)$ does not resolve the problem.
EM

The original EM method was introduced in [Baum et al., 1970].

Consider a model \( p(x, k|\theta) \). The problem is to find \( \theta \) that maximizes the observed data likelihood \( L(\theta) = \sum_x q(x) \log p(x|\theta) \) with \( q(x) \) the empirical distribution of \( x \). Taking the gradient wrt \( \theta \) and setting to zero, we find

\[
\frac{\partial L}{\partial \theta} = \sum_x q(x) \sum_k p(k|x, \theta) \frac{d \log p(x, k|\theta)}{d\theta}
\]

which often converges slow.

Instead of maximizing \( L \) we compute a lower bound and maximize that. For given \( x \):

\[
\log p(x|\theta) = \log \sum_k p(x, k|\theta) = \log \sum_k s_x(k) \frac{p(x, k|\theta)}{s_x(k)} \geq \sum_k s_x(k) \log \frac{p(x, k|\theta)}{s_x(k)} = \tilde{L}_x
\]

\[
L = \sum_x q(x) \log p(x|\theta) \geq \sum_x q(x) \tilde{L}_x = \tilde{L}
\]
The bound can be understood in terms of KL divergence:

\[
L - \tilde{L} = \sum_x q(x) \log p(x) - \sum_x q(x) \sum_k s_x(k) \log \frac{p(x, k)}{s_x(k)}
\]

\[
= \sum_x q(x) \sum_k s_x(k) \log \frac{s_x(k)}{p(k|x)} = \sum_x q(x) KL(s_x|p(·|x))
\]

The bound follows thus also from \( KL \geq 0 \).

The best bound is when \( s(k) \) is chosen to maximize \( \tilde{L}_x \) independently for each \( x \).
**EM**

The E step consists of maximizing $\tilde{L}_x$ wrt $s_x(k)$ for given $p(x, k|\theta_{\text{old}})$. When $s_x(k)$ is unparametrized (a list of $K$ numbers) the result is simply Bayes rule:

$$E\text{ step: } s_x(k) = \frac{p(x, k|\theta_{\text{old}})}{p(x|\theta_{\text{old}})} = p(k|x, \theta_{\text{old}})$$

Substitution gives

$$\tilde{L}_x = \sum_k p(k|x, \theta_{\text{old}}) \log \frac{p(x, k|\theta)}{p(k|x, \theta_{\text{old}})} \quad \tilde{L} = \sum_x q(x) \tilde{L}_x$$

The M step is to maximize $\tilde{L}$ wrt $\theta$ for fixed $\theta_{\text{old}}$. The fixed point equation is

$$M\text{ step: } \sum_x q(x) \sum_k p(k|x, \theta_{\text{old}}) \frac{\partial \log p(x, k|\theta)}{\partial \theta} = 0$$

which can often be solved for $\theta$ in terms of $\theta_{\text{old}}$. (Compare with the gradient $\frac{\partial L}{\partial \theta}$.)
EM does coordinate ascent on $\tilde{L}$ alternating in directions $s_x(k)$ (E step) and $\theta$ (M step).
Increasing $\tilde{L}$ either increases $L$ or tightens the bound on $L$. 
Maximum likelihood via EM for mixture of $K$ spherical Gaussians

Clustering: $x^n \in \mathbb{R}^d, n = 1, \ldots, N$ is observed data; $k = 1, \ldots, K$ are the cluster labels.

$$p(x, k|\theta) = p_k p(x|k, \theta) \quad p(x|k, \theta) = \mathcal{N}(x|\mu_k, \Sigma_k) \quad p_k = \frac{1}{K}$$

$\theta$ is $\{\mu_k, \Sigma_k\}, k = 1, \ldots, K$ with $\Sigma_k = \sigma_k I_d$ (spherical Gaussians)
E step: \[ s_n(k) = p(k|x^n, \theta^{old}) = \frac{\mathcal{N}(x^n|\mu_k^{old}, \Sigma_k^{old})}{\sum_{k'} \mathcal{N}(x^n|\mu_{k'}^{old}, \Sigma_{k'}^{old})} = r_k^n \]

M step: \[ 0 = \sum_x q(x) \sum_{k'} p(k'|x, \theta^{old}) \frac{\partial \log p(x, k'|\theta)}{\partial \{\mu_k, \sigma^2_k\}} = \sum_n s_n(k) \frac{\partial \log \mathcal{N}(x^n|\mu_k, \Sigma_k)}{\partial \{\mu_k, \sigma^2_k\}} \]

\[ \frac{\partial \log \mathcal{N}(x^n|\mu_k, \Sigma_k)}{\partial \mu_k} = -\Sigma_k^{-1}(x^n - \mu_k) = 0 \quad \rightarrow \quad \mu_k = \frac{\sum_n s_n(k)x^n}{\sum_n s_n(k)} \]

\[ \frac{\partial \log \mathcal{N}(x^n|\mu_k, \Sigma_k)}{\partial \sigma^2_k} = \frac{1}{2\sigma^4_k} \left( ||x^n - \mu_k||^2 - d\sigma^2_k \right) = 0 \quad \rightarrow \quad \sigma^2_k = \frac{1}{d} \frac{\sum_n s_n(k)||x^n - \mu_k||^2}{\sum_n s_n(k)} \]
Chapter 24: Marginalisation

Given the Gaussian distribution

\[ p(x|\mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right) \]

the log likelihood for \( N \) data points is

\[
\log p(\{x_n\}_{n=1}^{N}|\mu, \sigma) = -N \log(\sqrt{2\pi}\sigma) - \sum_n (x_n - \mu)^2 / 2\sigma^2
\]

\[ = -N \log(\sqrt{2\pi}\sigma) - \left[N(\mu - \bar{x})^2 + S\right] / 2\sigma^2 \]

with \( S = \sum_n (x_n - \bar{x})^2 \).

The posterior over \( \mu, \sigma \) assuming flat priors is Gaussian in \( \mu \) but not in \( \sigma \).

The maximum posterior likelihood is given by \( \{\mu, \sigma\}_{ML} = \{\bar{x}, \sigma_N = \sqrt{S/N}\} \).

\( \mu_{ML} \) is unbiased: \( \mathbb{E}\mu_{ML} = \mathbb{E}x_n = \mu \)

\( \sigma_{ML}^2 \) is biased: \( \mathbb{E}\sigma_{ML}^2 = \frac{N-1}{N} \sigma^2 \) (show!)
Chapter 24: Marginalisation

Estimate of $\mu$ independent of $\sigma$
Chapter 24: Marginalisation

Estimating $\sigma$ depends on $\mu$. 
Chapter 24: Marginalisation

Bayesian answer is unbiased. This requires marginalisation over $\mu$.

$$
p(\mu, \sigma|x_n^{N\in\mathbb{N}}) \rightarrow p(\sigma|x_n^{N\in\mathbb{N}}) = \int d\mu p(\mu, \sigma|x_n^{N\in\mathbb{N}})
$$

$$
p(\mu, \sigma|x_n^{N\in\mathbb{N}}) \propto (2\pi\sigma^2)^{-N/2} \exp\left(-\frac{N(\mu - \bar{x})^2 + S}{2\sigma^2}\right)
$$

$$
p(\sigma|x_n^{N\in\mathbb{N}}) = \int d\mu p(\mu, \sigma|x_n^{N\in\mathbb{N}})
$$

$$
\int \exp\left(-N(\mu - \bar{x})^2/2\sigma^2\right) = \sqrt{2\pi\sigma^2/N}
$$

$$
p(\sigma|x_n^{N\in\mathbb{N}}) \propto (2\pi\sigma^2)^{-(N-1)/2} \exp\left(-S/2\sigma^2\right)
$$

The most likely value is $\sigma_{N-1} = \sqrt{S/(N-1)}$. See bottom right figure previous page.
Chapter 27: Laplace method

When computing marginal:

\[ p(x_1) = \int dx_2 \ldots dx_N p(x_1, \ldots, x_N) \]

One can use the Laplace approximation:

\[
\begin{align*}
Z &= \int dx f(x) \\
\log f(x) &\approx \log f(x_0) - \frac{c}{2} (x - x_0)^2 \\
Z &\approx \int dx \exp \left( \log f(x_0) - \frac{c}{2} (x - x_0)^2 \right) = f(x_0) \sqrt{\frac{2\pi}{c}}
\end{align*}
\]

\[ c = -\frac{d^2 \log f(x_0)}{dx^2} \]
Chapter 27: Laplace method

In $N$ dimensions:

$$\log f(x) \approx \log f(x_0) - \frac{1}{2}(x - x_0)^T A(x - x_0) \quad A(x_0) = -\frac{d^2 \log f(x_0)}{dx_idx_j}$$

$$Z \approx f(x_0) \int dx \exp \left( -\frac{1}{2}(x - x_0)^T A(x - x_0) \right) = f(x_0) \sqrt{\frac{(2\pi)^N}{\det A}}$$
Summary

Inference is parameter estimation can be done

- full Bayesian. Requires integration over model parameters
- or maximum likelihood approach. Requires optimization

Bayesian approach

- Explicit solution (Gaussian case, exponential models with conjugate priors)
- Monte Carlo sampling
- variational methods
- Laplace approximation (= similar to maximum likelihood)
Summary

Maximum likelihood approach (may be biased)

- Simple models explicit solution possible (Gaussian)
- Models with latent variables (clustering, but also, Boltzmann Machines, deep networks, etc) often use EM approach

EM maximized a lower bound on the likelihood via two operations

- E step computes a conditional distribution $p(x|y)$ on latent variable $x$ given observation $y$; use exact (clustering) MCMC or variational
- M step improves or maximizes (clustering) parameters
Given a current state and a future desired state, what is the best/cheapest/fastest way to get there.
Why stochastic optimal control?
Optimal control theory

Hard problems:
- a learning and exploration problem
- a stochastic optimal control computation
- a representation problem $u(x, t)$
Outline

Optimal control theory, discrete time
- Introduction of delayed reward problem in discrete time;
- Dynamic programming solution

Optimal control theory, continuous time
- Pontryagin maximum principle;

Stochastic optimal control theory
- Stochastic differential equations
- Kolmogorov and Fokker-Plack equations
- Hamilton-Jacobi-Bellman equation
- LQ control, Ricatti equation;
- Portfolio selection

Path integral/KL control theory
- Importance sampling
- KL control theory
Material


• Dimitri Bertsekas, Dynamic programming and optimal control

• [http://www.snn.ru.nl/~bertk/machinelearning/](http://www.snn.ru.nl/~bertk/machinelearning/)
Optimal control theory: Optimize sum of a path cost and end cost. Result is optimal control sequence and optimal trajectory.

Input: Cost function.
Output: Optimal trajectory and controls.
Introduction

Control problems are delayed reward problems:

- Motor control: devise a sequence of motor commands to reach a goal
- Finance: devise a sequence of buy/sell commands to maximize profit
- Learning, exploration vs. exploitation
Types of optimal control problems

Finite horizon (fixed horizon time):

- Dynamics and environment may depend on time.
- Optimal control depends explicitly on time.

Finite horizon (moving horizon):

- Dynamics and environment are static.
- Optimal control is time independent.

Infinite horizon:

- discounted reward, Reinforcement learning
- average reward

Other issues:
• discrete vs. continuous state
• discrete vs. continuous time
• observable vs. partial observable
• noise
**Discrete time control**

Consider the control of a discrete time deterministic dynamical system:

\[ x_{t+1} = x_t + f(t, x_t, u_t), \quad t = 0, 1, \ldots, T - 1 \]

\( x_t \) describes the *state* and \( u_t \) specifies the *control* or *action* at time \( t \).

Given \( x_{t=0} = x_0 \) and \( u_{0:T-1} = u_0, u_1, \ldots, u_T - 1 \), we can compute \( x_{1:T} \).

Define a cost for each sequence of controls:

\[
C(x_0, u_{0:T-1}) = \phi(x_T) + \sum_{t=0}^{T-1} R(t, x_t, u_t)
\]

The problem of optimal control is to find the sequence \( u_{0:T-1} \) that minimizes \( C(x_0, u_{0:T-1}) \).
Dynamic programming

Find the minimal cost path from A to J.

\[ C(J) = 0, \quad C(H) = 3, \quad C(I) = 4 \]

\[ C(F) = \min(6 + C(H), 3 + C(I)) \]
Discrete time control

The optimal control problem can be solved by dynamic programming. Introduce the *optimal cost-to-go*:

\[
J(t, x_t) = \min_{u_{t:T-1}} \left( \phi(x_T) + \sum_{s=t}^{T-1} R(s, x_s, u_s) \right)
\]

which solves the optimal control problem from an intermediate time \( t \) until the fixed end time \( T \), for all intermediate states \( x_t \).

Then,

\[
J(T, x) = \phi(x) \\
J(0, x) = \min_{u_{0:T-1}} C(x, u_{0:T-1})
\]
Discrete time control

One can recursively compute $J(t, x)$ from $J(t + 1, x)$ for all $x$ in the following way:

$$J(t, x_t) = \min_{u_t: T-1} \left( \phi(x_T) + \sum_{s=t}^{T-1} R(s, x_s, u_s) \right)$$

$$= \min_{u_t} \left( R(t, x_t, u_t) + \min_{u_{t+1}: T-1} \left[ \phi(x_T) + \sum_{s=t+1}^{T-1} R(s, x_s, u_s) \right] \right)$$

$$= \min_{u_t} \left( R(t, x_t, u_t) + J(t + 1, x_{t+1}) \right)$$

$$= \min_{u_t} \left( R(t, x_t, u_t) + J(t + 1, x_t + f(t, x_t, u_t)) \right)$$

This is called the *Bellman Equation*.

Computes $u$ as a function of $x, t$ for all intermediate $t$ and all $x$. 
Discrete time control

The algorithm to compute the optimal control $u^{*}_{0:T-1}$, the optimal trajectory $x^{*}_{1:T}$ and the optimal cost is given by

1. Initialization: $J(T, x) = \phi(x)$

2. Backwards: For $t = T - 1, \ldots, 0$ and for all $x$ compute

   \[ u^{*}_t(x) = \arg\min_u \{ R(t, x, u) + J(t + 1, x + f(t, x, u)) \} \]

   \[ J(t, x) = R(t, x, u^{*}_t) + J(t + 1, x + f(t, x, u^{*}_t)) \]

3. Forwards: For $t = 0, \ldots, T - 1$ compute

   \[ x^{*}_{t+1} = x^{*}_t + f(t, x^{*}_t, u^{*}_t(x^{*}_t)) \]

NB: the backward computation requires $u^{*}_t(x)$ for all $x$. 
Stochastic case

\[ x_{t+1} = x_t + f(t, x_t, u_t, w_t) \quad t = 0, \ldots, T - 1 \]

At time \( t \), \( w_t \) is a random value drawn from a probability distribution \( p(w) \).

For instance,

\[ x_{t+1} = x_t + w_t, \quad x_0 = 0 \]

\[ w_t = \pm 1, \quad p(w_t = 1) = p(w_t = -1) = 1/2 \]

\[ x_t = \sum_{s=0}^{t-1} w_s \]

Thus, \( x_t \) random variable and so is the cost

\[ C(x_0) = \phi(x_T) + \sum_{t=0}^{T-1} R(t, x_t, u_t, \xi_t) \]
Stochastic case

\[
C(x_0) = \left\langle \phi(x_T) + \sum_{t=0}^{T-1} R(t, x_t, u_t, \xi_t) \right\rangle
\]

\[
= \sum_{w_0:T-1} \sum_{\xi_0:T-1} p(w_0:T-1) p(\xi_0:T-1) \left\langle \phi(x_T) + \sum_{t=0}^{T-1} R(t, x_t, u_t, \xi_t) \right\rangle
\]

with \(\xi_t, x_t, w_t\) random. Closed loop control: find functions \(u_t(x_t)\) that minimizes the remaining expected cost when in state \(x\) at time \(t\). \(\pi = \{u_0(\cdot), \ldots, u_{T-1}(\cdot)\}\) is called a policy.

\[
x_{t+1} = x_t + f(t, x_t, u_t(x_t), w_t)
\]

\[
C_\pi(x_0) = \left\langle \phi(x_T) + \sum_{t=0}^{T-1} R(t, x_t, u_t(x_t), \xi_t) \right\rangle
\]

\(\pi^* = \arg\min_{\pi} C_\pi(x_0)\) is optimal policy.
Stochastic Bellman Equation

\[ J(t, x_t) = \min_{u_t} \langle R(t, x_t, u_t, \xi_t) + J(t + 1, x_t + f(t, x_t, u_t, w_t)) \rangle \]

\[ J(T, x) = \phi(x) \]

\( u_t \) is optimized for each \( x_t \) separately. \( \pi = \{u_0, \ldots, u_{T-1}\} \) is optimal a policy.
Inventory problem

- $x_t = 0, 1, 2$ stock available at the beginning of period $t$.
- $u_t$ stock ordered at the beginning of period $t$. Maximum storage is 2: $u_t \leq 2 - x_t$.
- $w_t = 0, 1, 2$ demand during period $t$ with $p(w = 0, 1, 2) = (0.1, 0.7, 0.2)$; excess demand is lost.
- $u_t$ is the cost of purchasing $u_t$ units. $(x_t + u_t - w_t)^2$ is cost of stock at end of period $t$.

\[
x_{t+1} = \max(0, x_t + u_t - w_t)
\]

\[
C(x_0, u_{0:T-1}) = \left\langle \sum_{t=0}^{t=2} u_t + (x_t + u_t - w_t)^2 \right\rangle
\]

Planning horizon $T = 3$. 
Inventory problem
Apply Bellman Equation

\[ J_t(x_t) = \min_{u_t} \langle R(x_t, u_t, w_t) + J_{t+1}(f(x_t, u_t, w_t)) \rangle \]

\[ R(x, u, w) = u + (x + u - w)^2 \]

\[ f(x, u, w) = \max(0, x + u - w) \]

Start with \( J_3(x_3) = 0, \forall x_3 \).
Dynamic programming in action

Assume we are at stage \( t = 2 \) and the stock is \( x_2 \). The cost-to-go is what we order \( u_2 \) and how much we have left at the end of period \( t = 2 \).

\[
J_2(x_2) = \min_{0 \leq u_2 \leq 2-x_2} \left( u_2 + (x_2 + u_2 - w_2)^2 \right)
\]

\[
= \min_{0 \leq u_2 \leq 2-x_2} \left( u_2 + 0.1 \times (x_2 + u_2)^2 + 0.7 \times (x_2 + u_2 - 1)^2 
+ 0.2 \times (x_2 + u_2 - 2)^2 \right)
\]

\[
J_2(0) = \min_{0 \leq u_2 \leq 2} \left( u_2 + 0.1 \times u_2^2 + 0.7 \times (u_2 - 1)^2 + 0.2 \times (u_2 - 2)^2 \right)
\]

\[
u_2 = 0 : \quad rhs = 0 + 0.7 \times 1 + 0.2 \times 4 = 1.5
\]

\[
u_2 = 1 : \quad rhs = 1 + 0.1 \times 1 + 0.2 \times 1 = 1.3
\]

\[
u_2 = 2 : \quad rhs = 2 + 0.1 \times 4 + 0.7 \times 1 = 3.1
\]

Thus, \( u_2(x_2 = 0) = 1 \) and \( J_2(x_2 = 0) = 1.3 \)
Inventory problem

The computation can be repeated for $x_2 = 1$ and $x_2 = 2$, completing stage 2 and subsequently for stage 1 and stage 0.

<table>
<thead>
<tr>
<th>Stock</th>
<th>Stage 0 Cost-to-go</th>
<th>Stage 0 Optimal stock to purchase</th>
<th>Stage 1 Cost-to-go</th>
<th>Stage 1 Optimal stock to purchase</th>
<th>Stage 2 Cost-to-go</th>
<th>Stage 2 Optimal stock to purchase</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3.67</td>
<td>1</td>
<td>2.5</td>
<td>1</td>
<td>1.3</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2.67</td>
<td>0</td>
<td>1.2</td>
<td>0</td>
<td>0.3</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2.608</td>
<td>0</td>
<td>1.68</td>
<td>0</td>
<td>1.1</td>
<td>0</td>
</tr>
</tbody>
</table>
Exercise: Two ovens

A certain material is passed through a sequence of two ovens. Aim is to reach pre-specified final product temperature $x^*$ with minimal oven energy.

$\dot{x}_0, 1, 2$ are the product temperatures initially, after passing through oven 1 and after passing through oven 2. $u_{0,1}$ are the oven temperatures. The dynamics is

$$
x_{t+1} = (1 - a)x_t + au_t \quad t = 0, 1
$$

$$
C = r(x_2 - x^*)^2 + u_0^2 + u_1^2
$$

- Find the optimal control solution $u_0, u_1$.
- Show that adding mean zero noise to the dynamics ($x_{t+1} = (1 - a)x_t + au_t + w_t$ with $\langle w_t \rangle = 0$), does not change the optimal control solution.
Example: Two ovens

End cost-to-go is $J(2, x_2) = r(x_2 - x^*)^2$.

\[
J(1, x_1) = \min_{u_1} \left( u_1^2 + J(2, x_2) \right) = \min_{u_1} \left( u_1^2 + r((1 - a)x_1 + au_1 - x^*)^2 \right)
\]

\[
u_1 = \mu_1(x_1) = \frac{ra(x^* - (1 - a)x_1)}{1 + ra^2}
\]

\[
J(1, x_1) = \frac{r((1 - a)x_1 - x^*)^2}{1 + ra^2}
\]

\[
J(0, x_0) = \min_{u_0} \left( u_0^2 + J(1, x_1) \right) = \min_{u_0} \left( u_0^2 + r((1 - a)x_1 - x^*)^2 \right)
\]

\[
= \min_{u_0} \left( u_0^2 + \frac{r((1 - a)((1 - a)x_0 + au_0) - x^*)^2}{1 + ra^2} \right)
\]

\[
u_0 = \mu_0(x_0) = \frac{r(1 - a)a(x^* - (1 - a)^2 x_0)}{1 + ra^2(1 + (1 - a)^2)}
\]

\[
J(0, x_0) = \frac{r((1 - a)^2 x_0 - x^*)^2}{1 + ra^2(1 + (1 - a)^2)}
\]
• **Linear Quadratic Control:** Solution can be obtained in closed form because problem is linear quadratic.

• **Certainty equivalence:** Optimal control solution is unaffected by noise:

\[
\begin{align*}
x_{t+1} &= (1 - a)x_t + au_t + w_t \quad t = 0, 1 \\
C &= r(x_2 - x^*)^2 + u_0^2 + u_1^2
\end{align*}
\]

with \( \langle w_t \rangle = 0 \). Then

\[
J(1, x_1) = \min_{u_1} \left( u_1^2 + \langle r((1 - a)x_1 + au_1 + w_1 - x^*)^2 \rangle \right)
\]

\[
= \min_{u_1} \left( u_1^2 + r((1 - a)x_1 + au_1 - x^*)^2 + r \langle w_1 \rangle^2 \right)
\]
Continuous limit

Replace $t + 1$ by $t + dt$ with $dt \to 0$.

$$x_{t+dt} = x_t + f(x_t, u_t, t)dt$$

$$C(x_0, u_{0\to T}) = \phi(x_T) + \int_0^T d\tau R(\tau, x(\tau), u(\tau))$$

Assume $J(x, t)$ is smooth.

$$J(t, x) = \min_u (R(t, x, u)dt + J(t + dt, x + f(x, u, t)dt))$$

$$\approx \min_u (R(t, x, u)dt + J(t, x) + \partial_t J(t, x)dt + \partial_x J(t, x)f(x, u, t)dt)$$

$$-\partial_t J(t, x) = \min_u (R(t, x, u) + f(x, u, t)\partial_x J(x, t))$$

with boundary condition $J(x, T) = \phi(x)$. 
Continuous limit

\[-\partial_t J(t, x) = \min_u (R(t, x, u) + f(x, u, t)\partial_x J(x, t))\]

with boundary condition \(J(x, T) = \phi(x)\).

This is called the \textit{Hamilton-Jacobi-Bellman Equation}.

Computes the \textit{anticipated potential} \(J(t, x)\) from the future potential \(\phi(x)\).
Example: Mass on a spring

The spring force $F_z = -z$ towards the rest position and control force $F_u = u$. 

Newton’s Law

$$F = -z + u = m\ddot{z}$$

with $m = 1$.

Control problem: Given initial position and velocity $z(0) = \dot{z}(0) = 0$ at time $t = 0$, find the control path $-1 < u(0 \rightarrow T) < 1$ such that $z(T)$ is maximal.
Example: Mass on a spring

Introduce $x_1 = z, x_2 = \dot{z}$, then

\[
\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -x_1 + u
\end{align*}
\]

The end cost is $\phi(x) = -x_1$; path cost $R(x, u, t) = 0$.

The HJB takes the form:

\[
-\partial_t J = \min_u \left( x_2 \frac{\partial J}{\partial x_1} - x_1 \frac{\partial J}{\partial x_2} + \frac{\partial J}{\partial x_2} u \right) = x_2 \frac{\partial J}{\partial x_1} - x_1 \frac{\partial J}{\partial x_2} - \left| \frac{\partial J}{\partial x_2} \right|, \quad u = -\text{sign} \left( \frac{\partial J}{\partial x_2} \right)
\]
Example: Mass on a spring

We try $J(t, x) = \psi_1(t)x_1 + \psi_2(t)x_2 + \alpha(t)$. The HJBE reduces to the ordinary differential equations

$$
\dot{\psi}_1 = \psi_2 \\
\dot{\psi}_2 = -\psi_1 \\
\dot{\alpha} = -|\psi_2|
$$

These equations must be solved for all $t$, with final boundary conditions $\psi_1(T) = -1, \psi_2(T) = 0$ and $\alpha(T) = 0$.

Note, that the optimal control only requires $\partial_x J(x, t)$, which in this case is $\psi(t)$ and thus we do not need to solve $\alpha$. The solution for $\psi$ is

$$
\psi_1(t) = -\cos(t - T) \\
\psi_2(t) = \sin(t - T)
$$
Example: Mass on a spring

The optimal control is

\[ u(x, t) = -\text{sign}(\psi_2(t)) = -\text{sign}(\sin(t - T)) \]

As an example consider \( T = 2\pi \). Then, the optimal control is

\[ u = -1, \quad 0 < t < \pi \]
\[ u = 1, \quad \pi < t < 2\pi \]
Pontryagin minimum principle

The HJB equation is a PDE with boundary condition at future time. The PDE is solved using discretization of space and time.

The solution is an optimal cost-to-go for all $x$ and $t$. From this we compute the optimal trajectory and optimal control.

An alternative approach is a variational approach that directly finds the optimal trajectory and optimal control.
Pontryagin minimum principle

We can write the optimal control problem as a constrained optimization problem with independent variables \( u(0 \rightarrow T) \) and \( x(0 \rightarrow T) \)

\[
\min_{u(0\rightarrow T),x(0\rightarrow T)} \phi(x(T)) + \int_0^T dt R(x(t), u(t), t)
\]

subject to the constraint

\[
\dot{x} = f(x, u, t)
\]

and boundary condition \( x(0) = x_0 \).

Introduce the Lagrange multiplier function \( \lambda(t) \):

\[
C = \phi(x(T)) + \int_0^T dt \left[ R(t, x(t), u(t)) - \lambda(t) f(t, x(t), u(t)) - \dot{x}(t) \right]
\]

\[
= \phi(x(T)) + \int_0^T dt \left[ -H(t, x(t), u(t), \lambda(t)) + \lambda(t) \dot{x}(t) \right]
\]

\[-H(t, x, u, \lambda) = R(t, x, u) - \lambda f(t, x, u)\]
Derivation PMP

The solution is found by extremizing $C$. This gives a necessary but not sufficient condition for a solution.

If we vary the action wrt to the trajectory $x$, the control $u$ and the Lagrange multiplier $\lambda$, we get:

$$
\delta C = \phi_x(x(T))\delta x(T)
+ \int_0^T dt[-H_x\delta x(t) - H_u\delta u(t) + (-H_\lambda + \dot{x}(t))\delta \lambda(t) + \lambda(t)\delta \dot{x}(t)]
= (\phi_x(x(T)) + \lambda(T))\delta x(T)
+ \int_0^T dt\left[(-H_x - \dot{\lambda}(t))\delta x(t) - H_u\delta u(t) + (-H_\lambda + \dot{x}(t))\delta \lambda(t)\right]
$$

For instance, $H_x = \frac{\partial H(t, x(t), u(t), \lambda(t))}{\partial x(t)}$.

We can solve $H_u(t, x, u, \lambda) = 0$ for $u$ and denote the solution as

$$u^*(t, x, \lambda)$$
Assumes $H$ convex in $u$.

The remaining equations are

\[
\dot{x} = H_\lambda(t, x, u^*(t, x, \lambda), \lambda) \\
\dot{\lambda} = -H_x(t, x, u^*(t, x, \lambda), \lambda)
\]

with boundary conditions

\[
x(0) = x_0 \quad \lambda(T) = -\phi_x(x(T))
\]

Mixed boundary value problem.
Again mass on a spring

Problem

\[ \dot{x}_1 = x_2, \quad \dot{x}_2 = -x_1 + u \]
\[ R(x, u, t) = 0 \quad \phi(x) = -x_1 \]

Hamiltonian

\[ H(t, x, u, \lambda) = -R(t, x, u) + \lambda^t f(t, x, u) = \lambda_1 x_2 + \lambda_2 (-x_1 + u) \]
\[ H^*(t, x, \lambda) = \lambda_1 x_2 - \lambda_2 x_1 - |\lambda_2| \quad u^* = -\text{sign}(\lambda_2) \]

The Hamilton equations

\[ \dot{x} = \frac{\partial H^*}{\partial \lambda} \quad \Rightarrow \quad \dot{x}_1 = x_2, \quad \dot{x}_2 = -x_1 - \text{sign}(\lambda_2) \]
\[ \dot{\lambda} = -\frac{\partial H^*}{\partial x} \quad \Rightarrow \quad \dot{\lambda}_1 = \lambda_2, \quad \dot{\lambda}_2 = -\lambda_1 \]

with \( x(t = 0) = x_0 \) and \( \lambda(t = T) = (1, 0) \).
Example

Consider the control problem:

\[ dx = u dt \]

\[ C = \frac{\alpha}{2} x(T)^2 + \int_{t_0}^{T} dt \frac{1}{2} u(t)^2 \]

with initial condition \( x(t_0) \).

Solve the control problem using the PMP formalism.
Solution

The PMP recipe is

1. Construct the Hamiltonian

\[ H(t, x, u, \lambda) = -R(t, x, u) + \lambda f(t, u, x) = -\frac{1}{2}u^2 + \lambda u \]

2. Construct the optimized Hamiltonian

\[ H^*(t, x, \lambda) = H(t, x, u^*, \lambda) = \frac{1}{2}\lambda^2 \quad u^* = \lambda \]

3. Solve the Hamilton equations of motion

\[ \frac{dx}{dt} = \frac{\partial H^*}{\partial \lambda} = \lambda \]
\[ \frac{d\lambda}{dt} = -\frac{\partial H^*}{\partial x} = 0 \]
with boundary conditions \( x(t_0) \) and \( \lambda(t = T) = -\alpha x(T) \)\(^9\). The solution for \( \lambda \) is constant \( \lambda(t) = \lambda = -\alpha x(T) \). The solution for \( x(t) \) is

\[
x(t) = x(t_0) + \lambda(t - t_0)
\]

Combining these two results, we get \( \lambda = -\alpha x(T) = -\alpha(x(t_0) + \lambda(T - t_0)) \), or

\[
\lambda = \frac{-\alpha x(t_0)}{1 + \alpha(T - t_0)}
\]

Since \( u^* = \lambda \), this is the optimal control law.

\(^9\)Note, that \( \phi(x) = \frac{\alpha}{2}x^2 \) so that \( \phi_x = \alpha x \).
Relation to classical mechanics

The equations look like classical mechanics

\[
\begin{align*}
\dot{x} &= H_\lambda(t, x, u^*(t, x, \lambda), \lambda) \quad x(0) = x_0 \\
\dot{\lambda} &= -H_x(t, x, u^*(t, x, \lambda), \lambda) \quad \lambda(T) = -\phi_x(x(T)) 
\end{align*}
\]

In classical mechanics $H$ is called the Hamiltonian. Consider the time evolution of $H$:

\[
\begin{align*}
\dot{H} &= H_t + H_u u + H_x \dot{x} + H_\lambda \dot{\lambda} = H_t \\
H(t, x, u, \lambda) &= -R(t, x, u) + \lambda f(t, u, x)
\end{align*}
\]

So, for problems where $R, f$ do not explicitly depend on time, $H$ is a constant of the motion.
Example

Consider the control problem:

\[ dx = u \, dt \]

\[ C = \int_{t_0}^{t'} dt \left( \frac{1}{2} u(t)^2 + V(x(t)) \right) \]

with initial condition \( x(t_0) \).

1. \( H(x, u, \lambda) = -\frac{1}{2} u^2 - V(x) + \lambda u \)

2. \( u^* = \lambda, \quad H^*(x, \lambda) = \frac{1}{2} \lambda^2 - V(x) \)

3. \( \dot{x} = \frac{\partial H^*}{\partial \lambda} = \lambda \quad \dot{\lambda} = -\frac{\partial H^*}{\partial \lambda} = \frac{\partial V(x)}{\partial x} \)

Control cost \( V \) play role of *minus* potential energy.
Control solution has constant *difference* of kinetic energy and state cost.
Comments

The solution of the HJB PDE is expensive.

The PMP method is computationally less complicated than the HJB method because it does not require discretisation of the state space.

HJB generalizes to the stochastic case, PMP does not (at least not easy).
Stochastic control
Stochastic differential equations

Consider the random walk on the line:

\[ X_{t+1} = X_t + \xi_t \quad \xi_t = \pm 1 \]

with \( x_0 = 0 \). We can compute

\[ X_t = \sum_{i=1}^{t} \xi_i \]

Since \( x_t \) is a sum of random variables, \( x_t \) becomes Gaussian distributed with

\[ \mathbb{E} x_t = \sum_{i=1}^{t} \mathbb{E} \xi_i = 0 \]

\[ \nabla x_t = \sum_{i,j=1}^{t} \nabla \xi_i = t \]

Note, that the fluctuations \( \propto \sqrt{t} \).
Stochastic differential equations

In the continuous time limit we define

\[ dX_t = X_{t+dt} - X_t = dW_t \]

with \( dW_t \) an infinitesimal mean zero Gaussian variable: \( \mathbb{E}dW_t = 0, \mathbb{V}dW_t = \nu dt \).

Then with initial condition \( x_1 \) at \( t_1 \)

\[ X_t = x_1 + \int_{t_1}^t dW_s \quad \mathbb{E}X_t = x_0 \quad \mathbb{V}X_t = \nu t \]

is called a Wiener process or Brownian motion.

Since the increments are independent, \( X_t \) is Gaussian distributed

\[ p(x_2, t_2 | x_1, t_1) = \frac{1}{\sqrt{2\pi\nu(t_2 - t_1)}} \exp \left( -\frac{(x_2 - x_1)^2}{2\nu(t_2 - t_1)} \right) \]
Stochastic differential equations

Consider the stochastic differential equation

\[ dX_t = f(X_t, t)dt + dW_t \]

\( W_t \) is a Wiener process.

In this case \( \rho(x_2, t_2|x_1, t_1) \) may be very complex and is generally not known.

Define \( \rho(x, t) = p(x, t|x_0, 0) \). Then (Fokker-Planck forward equation)

\[
\frac{\partial}{\partial t} \rho(x, t) = -\nabla (f(x, t)\rho(x, t)) + \frac{1}{2} \nu \nabla^2 \rho(x, t), \quad \rho(x, 0) = \delta(x - x_0)
\]

Define \( \psi(x, t) = p(z, T|x, t) \). Then (Kolmogorov backward equation)

\[
-\frac{\partial}{\partial t} \psi(x, t) = f(x, t) \nabla \psi(x, t) + \frac{1}{2} \nu \nabla^2 \psi(x, t) \quad \psi(x, T) = \delta(z - x)
\]
Example: Brownian motion

\[ X_t = x_0 + \int_0^t dW_s \]

\[ \rho(x, t) = p(x, t|x_0, 0) = \frac{1}{\sqrt{2\pi \nu t}} \exp\left( -\frac{(x - x_0)^2}{2\nu t} \right) \]

\[ \psi(x, t) = p(z, T|x, t) = \frac{1}{\sqrt{2\pi \nu (T - t)}} \exp\left( -\frac{(x - z)^2}{2\nu (T - t)} \right) \]
Stochastic optimal control

Consider a stochastic dynamical system

$$dX_t = f(t, X_t, u)dt + dW_t$$

$W_t$ is a Wiener process with $\mathbb{E}dW_t^2 = \nu(t, x, u)dt$. \[^{10}\]

The cost becomes an expectation:

$$C(t, x, u) = \mathbb{E}\left(\phi(X_T) + \int_t^T d\tau R(t, X_t, u(X_t, t))\right)$$

over all stochastic trajectories starting at $x$ with control function $u(\cdot, t)$.

Optimize with respect to the set of functions $u(\cdot, t)$.

\[^{10}\]Our notation is for one dimensional $X$, but the theory generalizes trivially to higher dimension.
Stochastic optimal control

We obtain the Bellman recursion

\[
J(t, x_t) = \min_u R(t, x_t, u_t)dt + \mathbb{E}J(t + dt, X_{t+dt})
\]

\[
J(t + dt, x_t + dX_t) = J(t, x_t) + dt\partial_t J(t, x_t) + dX_t \partial_x J(t, x_t) + \frac{1}{2}dX_t^2 \partial_x^2 J(t, x_t)
\]

\[
\mathbb{E}J(t + dt, x_t + dX_t) = J(t, x_t) + dt\partial_t J(t, x_t) + f dt \partial_x J(t, x_t) + \frac{1}{2}\nu dt \partial_x^2 J(t, x_t)
\]

because \( \mathbb{E}dX_t = f dt \) and \( \mathbb{E}dX_t^2 = \nu dt + (f dt)^2 = \nu dt + O(dt^2) \).

Thus (Stochastic Hamilton-Jacobi-Bellman equation)

\[
-\partial_t J(t, x) = \min_u \left( R(t, x, u) + f(x, u, t)\partial_x J(x, t) + \frac{1}{2}\nu(t, x, u)\partial_x^2 J(x, t) \right)
\]

with boundary condition \( J(x, T) = \phi(x) \).
Linear Quadratic control

The dynamics is linear

$$dX_t = [A(t)X_t + B(t)u_t + b(t)]dt + \sum_{j=1}^{m} (C_j(t)X_t + D_j(t)u_t + \sigma_j(t))dW_j, \quad \langle dW_jdW_{j'} \rangle = \delta_{jj'}$$

The cost function is quadratic

$$\phi(x) = \frac{1}{2}x'Gx$$
$$R(x, u, t) = \frac{1}{2}x'Q(t)x + u'S(t)x + \frac{1}{2}u'R(t)u$$

In this case the optimal cost-to-go is quadratic in $x$:

$$J(t, x) = \frac{1}{2}x'P(t)x + \alpha'(t)x + \beta(t)$$
$$u_t = -\Psi(t)x_t - \psi(t)$$
Substitution in the HJB equation yields ODEs for $P, \alpha, \beta$:

\[
-\dot{P} = PA + A'P + \sum_{j=1}^{m} C'_jPC_j + Q - \hat{S}'\hat{R}^{-1}\hat{S}
\]

\[
-\dot{\alpha} = [A - B\hat{R}^{-1}\hat{S}]'\alpha + \sum_{j=1}^{m} [C_j - D_j\hat{R}^{-1}\hat{S}]'P\sigma_j + Pb
\]

\[
\dot{\beta} = \frac{1}{2} \left| \sqrt{\hat{R}\psi} \right|^2 - \alpha' b - \frac{1}{2} \sum_{j=1}^{m} \sigma'_jP\sigma_j
\]

\[
\dot{\hat{R}} = R + \sum_{j=1}^{m} D'_jPD_j
\]

\[
\hat{S} = B'P + S + \sum_{j=1}^{m} D'_jPC_j
\]

\[
\Psi = \hat{R}^{-1}\hat{S}
\]

\[
\psi = \hat{R}^{-1}(B'\alpha + \sum_{j=1}^{m} D'_jP\sigma_j)
\]

with $P(t_f) = G$ and $\alpha(t_f) = \beta(t_f) = 0$. 

Bert Kappen ML 227
**Example**

Find the optimal control for the dynamics

\[
dX_t = u dt + dW_t, \quad \langle dW_t^2 \rangle = \nu dt
\]

\[
C = \left\langle \frac{1}{2} Gx(T)^2 + \int_0^T dt \frac{1}{2} u(x,t)^2 \right\rangle
\]

with end cost \( \phi(x) = \frac{1}{2} Gx^2 \) and path cost \( R(x,u) = \frac{1}{2} u^2 \).

\( A = 0, B = 1, b = 0, C = D = 0, \sigma_j = \sqrt{\nu}, m = 1, \hat{R} = 1, \hat{S} = P, \Psi = P, \psi = \alpha \)

The Ricatti equations reduce to

\[
\dot{P} = P^2 \quad P(T) = G
\]

\[
\dot{\alpha} = P\alpha \quad \alpha(T) = 0
\]

\[
\dot{\beta} = \frac{1}{2} \alpha^2 - \frac{1}{2} \nu P
\]
The solution is $\alpha(t) = 0$ and
\[
P(t) = \frac{1}{c - t} \quad \frac{1}{c - T} = G
\]
and $\beta$ not relevant.

\[
u(x, t) = -P(t)x - \alpha(t) = -\frac{Gx}{1 + G(T - t)}
\]

Compare with deterministic case considered earlier, is identical due to certainty equivalence.
When $G \to \infty$ we obtain the Brownian bridge. The control law and dynamics becomes

\[ dx = u dt + d\xi \]
\[ u = \frac{-x(t_0)}{T - t_0} \]

$x(T) \to 0 \text{ w.p. } 1$. 
Example

Find the optimal control for the dynamics

\[ dX_t = u dt + dW_t, \quad \langle dW_t^2 \rangle = v dt \]

with end cost \( \phi(x) = 0 \) and path cost \( R(x, u) = \frac{1}{2}(Qx^2 + Ru^2) \).

The Ricatti equations reduce to

\[
\begin{align*}
-\dot{P} &= Q - R^{-1}P^2 \\
-\dot{\alpha} &= -R^{-1}P \alpha = 0 \\
\dot{\beta} &= -\frac{1}{2}vP
\end{align*}
\]

with \( P(T) = \alpha(T) = \beta(T) = 0 \) and

\[ u(x, t) = -R^{-1}P(t)x \]
The solution is

\[
P(t) = \sqrt{RQ} \tanh \left( \frac{\sqrt{Q}}{R} (T - t) \right)
\]

\[
\alpha(t) = 0
\]

\[
\beta(t) = \frac{1}{2} \nu R \log \cosh \left( \frac{\sqrt{Q}}{R} (T - t) \right)
\]

\[
\Psi(t) = R^{-1} P(t) \quad \psi(t) = 0
\]

The control is given by Eq. (2):

\[
u(x, t) = -R^{-1} P(t)x
\]

(2)
Comments

Note, that in the last example the optimal control is independent of $\nu$, i.e. optimal stochastic control equals optimal deterministic control.

In general:

- If $C_j = D_j = 0$ (only 'additive noise') $\dot{P}, \dot{\alpha}$ independent of noise $\sigma$, $\dot{\beta}$ depends on $\sigma$, but control independent of $\beta$. Thus control independent of $\sigma$ (certainty equivalence)

- If $C_j \neq 0$ or $D_j \neq 0$, control depends on $C_j, D_j, \sigma_j$ (no certainty equivalence)
Example: Portfolio selection

Consider a market with \( p \) stocks and one bond. The bond price process is subject to the following deterministic ordinary differential equation:

\[
dP_0(t) = r(t)P_0(t)dt, \quad P_0(0) = p_0 > 0
\]  

(3)

The other assets have price processes \( P_i(t), i = 1, \ldots, p \) satisfying stochastic differential equations

\[
dP_i(t) = P_i(t) \left( b_i(t)dt + \sum_{j=1}^{m} \sigma_{ij}(t)d\xi_j(t) \right), \quad P_i(0) = p_i > 0
\]  

(4)

Consider an investor whose total wealth at time \( t \) is denoted by \( x(t) \)

\[
x(t) = \sum_{i=0}^{p} N_i(t)P_i(t)
\]  

(5)

\[11\] This section is from [?] section 6.8 (pg. 335).
with \( N_i \) the number of stocks/bond of type \( i \). For given \( N_i(t) \),

\[
dx(t) = \sum_{i=0}^{p} N_i(t) dP_i(t) = \left( r(t) x(t) + \sum_{i=1}^{p} (b_i(t) - r(t)) u_i(t) \right) dt + \sum_{i=1}^{p} \sum_{j=1}^{m} \sigma_{ij}(t) u_i(t) d\xi_j(t)
\]

with \( u_i(t) = N_i(t) P_i(t), i = 1, \ldots, p \) the rescaled control variable.

The objective of the investor is to maximize the mean terminal wealth \( \langle x(t_f) \rangle \) and minimize at the same time the variance

\[
\Sigma^2 = \langle x(t_f)^2 \rangle - \langle x(t_f) \rangle^2
\]

This is a multi-objective optimization problem with an efficient frontier of optimal solutions: for each given mean there is a minimal variance.

These pairs can be found by minimizing the single objective criterion

\[
\mu \Sigma^2 - \langle x(t_f) \rangle
\]

for different values of the weighting factor \( \mu \).
This objective, however, is not an expectation value of some stochastic quantity due to the $\langle \cdot \rangle^2$ term. Consider a slightly different problem, minimizing the objective

$$\left\langle \mu x(t_f)^2 - \lambda x(t_f) \right\rangle$$

(8)

which is of the standard stochastic optimization form. One can show that one can construct a solution of Problem 7 by solving problem 8 for suitable $\lambda(\mu)$.\(^{12}\)

Our goal is thus to minimize eq. 8 subject to the stochastic dynamics eq. 6.

This is an LQ problem. The solution is computed from the Ricatti equations

$$u_i(x, t) = \psi_i(t)x + \phi_i(t)$$

As an example we consider the simplest possible case: $p = m = 1$ and $r, b, \sigma$ independent of time.

\(^{12}\) and finding $\lambda$ from

$$\lambda = 1 + 2\mu \left\langle x(t_f) \right\rangle (\lambda, \mu)$$

([?] Theorem 8.2 pg. 338)
Parameter values are: \( p = m = 1 \). Trading period is one year weekly. annual bond rate 5 % \((r = 0.0009758)\), annual expected stock rate is 10 % \((b = 0.0019)\), volatility \( \sigma = 2b \). \( x_0 = 2 \). Shows \( \text{var } x \) versus \( \langle x \rangle \) scatter plot for various values of \( \mu \). Small \( \mu \) corresponds to risky investments with high expected return and large fluctuation. \( \mu \rightarrow \infty \) corresponds to riskless investment in bond only and a return of 5 %.

\( \mu = 10 \) corresponds to \( \langle x \rangle = 3 \) and \( \sqrt{\text{var}} = 0.2 \).
Simulation of optimal control with $\mu = 10$, The optimal strategy is to borrow many stocks and sell them as soon as the objective is achieved.

Indeed, $\langle x \rangle = 3$ as expected. The strategy to get at this 50 % increase in wealth is to buy many stocks and hope they will give the expected wealth increase. As soon as this occurs, all stocks are sold and the money is put in the bank.  

\[ (2 + 50)(1 + bt) - 50(1 + rt) = 3 \quad 50(b - r)t \approx 1 \]
Path integral control

The $n$-dimensional path integral control problem is defined as

$$dX_t = f(X_t, t)dt + g(x, t)(u(X_t, t)dt + dW_t)$$

$$C(t, x, u) = \mathbb{E}\left(\phi(X_T) + \int_t^T dsV(X_s, s) + \frac{1}{2}u'(X_s, s)Ru(X_s, s)\right)$$

with $\mathbb{E}dW_t dW'_t = \nu dt$. $g$ is $n \times m$ matrix, $\nu$ is $m \times m$ matrix and $u, dW_t$ are $m$ dimensional.

The cost is an expectation over all stochastic trajectories starting at $x$ with control function $u(x, t)$.

The stochastic HJB equation becomes

$$-\partial_t J = \min_u \left(\frac{1}{2}u'Ru + V + (\nabla J)'(f + gu) + \frac{1}{2}\text{Tr}\left(gvg'\nabla^2 J\right)\right)$$

which we need to solve with end boundary condition $J(x, t_f) = \phi(x)$ for all $x$. 
Path integral control

Minimization wrt $u$ yields:

\[
  u = -R^{-1}g' \nabla J
\]

\[
  -\partial_t J = -\frac{1}{2} (\nabla J)' g R^{-1} g' (\nabla J) + V + (\nabla J)' f + \frac{1}{2} \text{Tr} \left( gg' \nabla^2 J \right)
\]

Define $\psi(x, t)$ through $J(x, t) = -\lambda \log \psi(x, t)$ and impose a relation between $R$ and $\nu$:

\[
  R = \lambda \nu^{-1}
\]

with $\lambda$ a positive number.
Path integral control

Then the HJB becomes linear in $\psi$

$$-\partial_t \psi = \left( -\frac{V}{\lambda} + f' \nabla + \frac{1}{2} \text{Tr}(g \nu g' \nabla^2) \right) \psi$$

with end condition $\psi(x, T) = \exp(-\phi(x)/\lambda)$\(^{15}\)

\(^{15}\) We sketch the derivation for $g = 1$.

$$-\frac{1}{2} (\nabla J)' R^{-1} (\nabla J) + \frac{1}{2} \text{Tr}(\nu \nabla^2 J) = -\frac{1}{2} \sum_{ij} \nabla_i J R_{ij}^{-1} \nabla_j J + \frac{1}{2} \lambda \sum_{ij} R_{ij}^{-1} \nabla_{ij} J$$

$$= \frac{1}{2} \sum_{ij} R_{ij}^{-1} \left( -\nabla_i J \nabla_j J + \lambda \nabla_{ij} J \right)$$

$$= \frac{1}{2} \sum_{ij} R_{ij}^{-1} \left( -\lambda^2 \frac{1}{\psi} \nabla_{ij} \psi \right)$$

since

$$-\nabla_i J \nabla_j J = -\lambda^2 \frac{1}{\psi^2} \nabla_i \psi \nabla_j \psi$$

$$\nabla_{ij} J = -\lambda \nabla_i \nabla_j \log \psi = -\lambda \nabla_i \left( \frac{1}{\psi} \nabla_j \psi \right) = \lambda \frac{1}{\psi^2} \nabla_i \psi \nabla_j \psi \nabla \psi - \frac{1}{\psi} \nabla_{ij} \psi$$
Path integral control

We identify $\psi(x, t) \propto p(z, T|x, t)$, then the linear Bellman equation

$$-\partial_t \psi = \left( -\frac{V}{\lambda} + f' \nabla + \frac{1}{2} \text{Tr}(g v g' \nabla^2) \right) \psi$$

can be interpreted as a Kolmogorov backward equation for the process

$$dx_i = f_i(x, t) dt + \sum_a g_{ia}(x, t) d\xi_a$$

$$x(t) = \uparrow \text{ with probability } V(x, t) dt / \lambda$$

$$x(T) = \uparrow \text{ with probability } \phi(x) / \lambda$$

The corresponding forward equation is

$$\partial_t \rho = -\frac{V}{\lambda} \rho - \nabla (f \rho) + \frac{1}{2} \text{Tr} \nabla^2 g v g' \rho$$

with $\rho(x, t) = p(x, t|z, 0)$ and $\rho(x, 0) = \delta(x - z)$. 
Feynman-Kac formula

Denote $Q(\tau|x, s)$ the distribution over uncontrolled trajectories that start at $x, t$:

$$dx = f(x, t) dt + g(x, t) d\xi$$

with $\tau$ a trajectory $x(t \to T)$. Then

$$\psi(x, t) = \int dQ(\tau|x, t) \exp\left(-\frac{S(\tau)}{\lambda}\right)$$

$$S(\tau) = \phi(x(T)) + \int_t^T ds V(x(s), s)$$

$\psi$ can be computed by forward sampling the uncontrolled process.
Uncontrolled dynamics specifies distribution $q(\tau|x, t)$ over trajectories $\tau$ from $x, t$.

Cost for trajectory $\tau$ is $S(\tau|x, t) = \phi(x_T) + \int_t^T ds V(x_s, s)$.

Find optimal distribution $p(\tau|x, t)$ that minimizes $\mathbb{E}_p S$ and is 'close' to $q(\tau|x, t)$. 
KL control

Find $p^*$ that minimizes

$$C(p) = KL(p|q) + \mathbb{E}_p S \quad KL(p|q) = \int d\tau p(\tau|x, t) \log \frac{p(\tau|x, t)}{q(\tau|x, t)}$$

The optimal solution is given by

$$p^*(\tau|x, t) = \frac{1}{\psi(x, t)} q(\tau|x, t) \exp(-S(\tau|x, t))$$

$$\psi(x, t) = \int d\tau q(\tau|x, t) \exp(-S(\tau|x, t)) = \mathbb{E}_q e^{-S}$$

The optimal cost is:

$$C(p^*) = - \log \psi(x, t)$$
Controlled diffusions

In the case of controlled diffusions, \( p(\tau|x, t) \) is parametrised by functions \( u(x, t) \), \( q(\tau|x, t) \) corresponds to \( u(x, t) = 0 \):

\[
\begin{align*}
dX_t &= f(X_t, t)dt + g(X_t, t)(u(X_t, t)dt + dW_t) & \mathbb{E}(dW_idW_j) &= \nu_{ij}dt \\
C(p) &= \mathbb{E}_p\left(\int dt \frac{1}{2}u(X_t, t)'\nu^{-1}u(X_t, t) + S(\tau|x, t)\right)
\end{align*}
\]

\( J(x, t) = -\log \psi(x, t) \) is the solution of the Bellman equation.

\( p^* \) is generated by optimal control \( u^*(x, t) \):

\[
\begin{align*}
u^*(x, t)dt &= \mathbb{E}_{p^*}(dW_t) = \frac{\mathbb{E}_q(dWe^{-S})}{\mathbb{E}_q(e^{-S})}
\end{align*}
\]

\( \psi, u^* \) can be computed by forward sampling from \( q \).
Consider a stochastic dynamical system

\[ dX_t = f(X_t, u)dt + g(X_t)dW_t \quad \mathbb{E}(dW_{t,i}dW_{t,j}) = \nu_{ij}dt \]

Given \( X_0 \) find control function \( u(x, t) \) that minimizes the expected future cost

\[ C = \mathbb{E}\left( \phi(X_T) + \int_0^T dt R(X_t, u(X_t, t)) \right) \]
Standard approach: define $J(x, t)$ is optimal cost-to-go from $x, t$.

$$J(x, t) = \min_u u_{t:T} \mathbb{E}_u \left( \phi(X_T) + \int_t^T dt R(X_t, u(X_t, t)) \right) \quad X_t = x$$

$J$ satisfies a partial differential equation

$$-\partial_t J(t, x) = \min_u \left( R(x, u) + f(x, u) \nabla_x J(x, t) + \frac{1}{2} \text{Tr}(g v g' \nabla_x^2 J(x, t)) \right) \quad J(x, T) = \phi(x)$$

with $u = u(x, t)$. This is HJB equation. Optimal control $u^*(x, t)$ defines distribution over trajectories $p^*(\tau) (= p(\tau | x_0, 0))$. 
Path integral control theory

\[ dX_t = \frac{f(X_t)dt}{f(X_t,u)dt} + g(X_t)u(X_t, t)dt + g(X_t)dW_t \quad X_0 = x_0 \]

Goal is to find function \( u(x, t) \) that minimizes

\[ C = \mathbb{E} \left\{ \phi(X_T) + \int_0^T dt \left( V(X_t, t) + \frac{1}{2}u(X_t, t)^2 \right) \right\} = \mathbb{E} \left( S(\tau) + \int_0^T dt \frac{1}{2}u(X_t, t)^2 \right) \]

\[ S(\tau) = \phi(X_T) + \int_0^T V(X_t, t) \]
Equivalent formulation: Find distribution over trajectories $p$ that minimizes $C(p)$

$$C(p) = \int d\tau p(\tau) \left( S(\tau) + \log \frac{p(\tau)}{q(\tau)} \right)$$

$q(\tau|x_0, 0)$ is distribution over uncontrolled trajectories.

The optimal solution is given by $p^*(\tau) = \frac{1}{\psi} q(\tau) e^{-S(\tau)}$
Path integral control theory

Equivalent formulation: Find distribution over trajectories $p$ that minimizes

$$C(p) = \int d\tau p(\tau) \left( S(\tau) + \log \frac{p(\tau)}{q(\tau)} \right)$$

$q(\tau|x_0, 0)$ is distribution over uncontrolled trajectories.

The optimal solution is given by \( p^*(\tau) = \frac{1}{\psi} q(\tau) e^{-S(\tau)} = p(\tau|u^*) \).

Equivalence of optimal control and discounted cost (Girsanov)
The optimal control cost is $C(p^*) = -\log \psi = J(x_0, 0)$ with

$$\psi = \int d\tau q(\tau) e^{-S(\tau)} = \mathbb{E}_q e^{-S}$$

$J(x, t)$ can be computed by forward sampling from $q$. 
Delayed choice

Time-to-go $T = 2 - t$.

$$J(x, t) = -\nu \log \mathbb{E}_q \exp(-\phi(X_2)/\nu)$$

Decision is made at $T = \frac{1}{\nu}$
Delayed choice

Time-to-go \( T = 2 - t \).

\[
J(x, t) = -\nu \log \mathbb{E}_q \exp(-\phi(X_2)/\nu)
\]

"When the future is uncertain, delay your decisions."
Delayed choice (details)

\[ dX_t = u dt + dW_t \quad \mathbb{E}dW^2_t = \nu dt \]

\( V = 0 \), path cost is \( \frac{1}{2}u^2 \) and end cost \( \phi(z = \pm 1) = 0, \phi(z) = \infty \) else encodes two targets at \( z = \pm 1 \) at \( t = T \).

PI recipe:

1. \[
\psi(x, t) = \int dQ(\tau|x, t) \exp(-S(\tau)/\lambda) \\
S(\tau) = \phi(x(T)) \\
\psi(x, t) = \int dz q(z, T|x, t) \exp(-\phi(z)/\lambda) = q(1, T|x, t) + q(-1, T|x, t) \\
q(z, T|x, t) = N(z|x, \nu(T-t))
\]

2. Compute

\[
J(x, t) = -\lambda \log \psi(x, t) = \frac{1}{T-t} \left( \frac{1}{2}x^2 - \nu(T-t) \log 2 \cosh \frac{x}{\nu(T-t)} \right)
\]
3.

\[ u(x, t) = -\nabla J(x, t) = \frac{1}{T-t} \left( \tanh \frac{x}{v(T-t)} - x \right) \]
Coordination of UAVs

$\approx 10,000$ trajectories per iteration, 3 iterations per second.

Video at: [http://www.snn.ru.nl/~bertk/control_theory/PI_quadrotors.mp4](http://www.snn.ru.nl/~bertk/control_theory/PI_quadrotors.mp4)

Gomez et al. 2015
Coordination of UAVs

Chao Xu ACC 2017
Importance sampling and control

\[ \psi(x, t) = \mathbb{E}_q e^{-S} \quad S(\tau|x, t) = \phi(x_T) + \int_t^T dsV(x_s, s) \]

Sampling is 'correct' but inefficient.
”To compute or not to compute, that is the question”

There are two extreme approaches to compute actions:

- precompute the appropriate action \( u(x) \) for any possible situation \( x \). Complex to learn and to store. Fast to execute
- compute the appropriate action \( u(x) \) for the current situation \( x \). Low learning and storage cost. Slow execution.

Intuitively, one can imagine that the most efficient approach is to combine both ideas (like ’just-in-time’ manufacturing):

- precompute ’basic motor skills’, the ’halffabrikaat’
- compute the appropriate action \( u(x) \) from the basic motor skills
Importance sampling

Consider simple 1-d sampling problem. Given $q(x)$, compute

$$a = \text{Prob}(x < 0) = \int_{-\infty}^{\infty} I(x)q(x)dx$$

with $I(x) = 0, 1$ if $x > 0, x < 0$, respectively.

Naive method: generate $N$ samples $X_i \sim q$

$$\hat{a} = \frac{1}{N} \sum_{i=1}^{N} I(X_i) \quad \mathbb{E}\hat{a} = a \quad \text{Var}(\hat{a}) = \frac{1}{N} \text{Var}(I)$$
Importance sampling

Consider another distribution \( p(x) \). Then

\[
a = \text{Prob}(x < 0) = \int_{-\infty}^{\infty} I(x) \frac{q(x)}{p(x)} p(x) \, dx
\]

Importance sampling: generate \( N \) samples \( X_i \sim p \)

\[
\hat{a} = \frac{1}{N} \sum_{i=1}^{N} I(X_i) \frac{q(X_i)}{p(X_i)} \quad \mathbb{E}\hat{a} = a \quad \text{Var}(\hat{a}) = \frac{1}{N} \text{Var}\left( \frac{I_p}{q} \right)
\]

Unbiased (= correct) for any \( p \)
Optimal importance sampling

The distribution

\[ p^*(x) = \frac{q(x)I(x)}{a} \]

is the optimal importance sampler.

One sample \( X \sim p^* \) is sufficient to estimate \( a \):

\[ \hat{a} = I(X) \frac{q(X)}{p^*(X)} = a \quad \mathbb{E}\hat{a} = a \quad Var(\hat{a}) = 0 \]
Estimating $\psi = \mathbb{E}e^{-S}$

Sample $N$ trajectories from uncontrolled dynamics

$$\tau_i \sim q(\tau) \quad w_i = e^{-S(\tau_i)} \quad \hat{\psi} = \frac{1}{N} \sum_i w_i$$

$\hat{\psi}$ unbiased estimate of $\psi$.

Sampling efficiency is inversely proportional to variance in (normalized) $w_i$.

$$ESS = \frac{N}{1 + N^2 \text{Var}(w)}$$
Sample $N$ trajectories from controlled dynamics and reweight yields unbiased estimate of cost-to-go:

$$
\tau_i \sim p(\tau) \quad w_i = e^{-S(\tau_i)} \frac{q(\tau_i)}{p(\tau_i)} = e^{-S_u(\tau_i)} \quad \hat{\psi} = \frac{1}{N} \sum_i w_i
$$

$$
S_u(\tau) = S(\tau) + \int_0^T dt \frac{1}{2} u(X_t, t)^2 + \int_0^T u(X_t, t) dW_t
$$
Importance sampling

\[ S_u(\tau) = S(\tau) + \int_0^T dt \frac{1}{2} u(X_t, t)^2 + \int_0^T u(X_t, t) dW_t \]

Thm:
- Better \( u \) (in the sense of optimal control) provides a better sampler (in the sense of effective sample size).
- Optimal \( u = u^* \) (in the sense of optimal control) requires only one sample and \( S_u(\tau) \) deterministic!

Thijssen, Kappen 2015
Proof

Control cost is $C(p) = \mathbb{E}_p \left( S(\tau) + \log \frac{p(\tau)}{q(\tau)} \right) = \mathbb{E} S_u$

Using Jensen’s inequality:

$$C^* = -\log \sum_{\tau} q(\tau) e^{-S(\tau)} = -\log \sum_{\tau} p(\tau) e^{-S(\tau) - \log \frac{p(\tau)}{q(\tau)}} \leq \sum_{\tau} p(\tau) \left( S(\tau) + \log \frac{p(\tau)}{q(\tau)} \right) = C(p)$$
Proof

Control cost is $C(p) = \mathbb{E}_p \left( S(\tau) + \log \frac{p(\tau)}{q(\tau)} \right) = \mathbb{E} S_u$

Using Jensen’s inequality:

$$C^* = - \log \sum_\tau q(\tau) e^{-S(\tau)} = - \log \sum_\tau p(\tau) e^{-S(\tau)} - \log \frac{p(\tau)}{q(\tau)} \leq \sum_\tau p(\tau) \left( S(\tau) + \log \frac{p(\tau)}{q(\tau)} \right) = C(p)$$

The inequality is saturated when $S(\tau) + \log \frac{p(\tau)}{q(\tau)}$ has zero variance: left and right side evaluate to $S(\tau) + \log \frac{p(\tau)}{q(\tau)}$.

This is realized when $p = p^*$ \[17\].
Example

Geometric Brownian motion on the interval $t = 0$ to $T$.

$$dX_t = X_t (u(tX_t, t) dt + dW_t),$$

$$C = \mathbb{E} \frac{1}{2} \log(X_T)^2$$

$$u(x, t) = a(t) + b(t)x + c(t)x^2$$

<table>
<thead>
<tr>
<th></th>
<th>$u = 0$</th>
<th>constant</th>
<th>linear</th>
<th>quadratic</th>
<th>optimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C$</td>
<td>7.526</td>
<td>5.139</td>
<td>1.507</td>
<td>1.461</td>
<td>1.420</td>
</tr>
<tr>
<td>FES(%)</td>
<td>34.3</td>
<td>42.08</td>
<td>87.5</td>
<td>95.2</td>
<td>99.3</td>
</tr>
</tbody>
</table>
The Path Integral Cross Entropy (PICE) method

We wish to estimate

$$\psi = \int d\tau q(\tau)e^{-S(\tau)}$$

The optimal (zero variance) importance sampler is $p^*(\tau) = \frac{1}{\psi}q(\tau)e^{-S(\tau)}$.

We approximate $p^*(\tau)$ with $p_u(\tau)$, where $u(x, t|\theta)$ is a parametrized control function.

Following the Cross Entropy method, we minimise $KL(p^*|p_u)$.

$$KL(p^*|p_u) \propto \int d\tau p^*(\tau) \log p_u(\tau)$$

We estimate $KL(p^*|p_u)$ and its gradients with importance sampling (use current best controller).

$$\Delta \theta \propto -\frac{\partial KL(p^*|p_u)}{\partial \theta} \propto -\mathbb{E}_u e^{-S_u} \int_0^T dW_t \frac{\partial u(X_t, t|\theta)}{\partial \theta}$$
$u(x, t|\theta)$ is arbitrary.

Estimate gradient by sampling.
Adaptive importance sampling

for $k = 0, \ldots$ do

$\text{data}_k = \text{generate\_data}(\text{model}, u_k)$  \hspace{1cm} % Importance sampler

$u_{k+1} = \text{learn\_control}(\text{data}_k, u_k)$  \hspace{1cm} % Gradient descent

end for

Parallel sampling
Parallel gradient computation

[Diagram showing a network with nodes labeled "MC" and "Δθ" interconnected by arrows and a square box labeled "D" leading into the network.]
Inverted pendulum

Simple 2nd order pendulum with noise, $X = (\alpha, \dot{\alpha})$

$$\ddot{\alpha} = -\cos \alpha + u \quad C = \mathbb{E} \int_0^T dt V(X_t) + \frac{1}{2} u(X_t, t)^2$$

Naive grid: $u(x) = \sum_k u_k \delta_{x,x_k}$.

$ESS < 1$ due to time discretization, finite sample size effects and $u(x, t) = u(x)$.

Illustration of gradient descent learning Eq. ?? for a second order inverted pendulum problem. Left: Entropic sample size versus importance sampling iteration.
Middle: Optimal cost to go versus importance sampling iteration. Right: Optimal control solution $\hat{u}(x_1, x_2)$ versus $x_1, x_2$ with $0 \leq x_1 \leq 2\pi$ and $-2 \leq x_2 \leq 2$. 
Acrobot

2 DOF, second order, under actuated, continuous stochastic control problem. Task is swing-up from down position.

\begin{align*}
    d_{11} \ddot{q}_1 + d_{12} \ddot{q}_2 + h_1 + \phi_1 &= 0 \\
    d_{21} \ddot{q}_1 + d_{22} \ddot{q}_2 + h_2 + \phi_2 &= \tau,
\end{align*}

where

\begin{align*}
    d_{11} &= m_1 l_1^2 + m_2 (l_1^2 + l_2^2 + 2l_1 l_2 \cos(q_2)) + l_1 + l_2 \\
    d_{12} &= m_2 l_2^2 + l_2 \\
    d_{21} &= m_2 l_2^2 + l_1 l_2 \cos(q_2) + l_2 \\
    d_{22} &= m_2 l_2^2 + l_1 l_2 \cos(q_2) + l_2 \\
    h_1 &= -m_2 l_2 \sin(q_2) \dot{q}_2^2 + 2m_2 l_2 \sin(q_2) \dot{q}_2 \dot{q}_1 \\
    h_2 &= m_2 l_2 \sin(q_2) \dot{q}_1^2 \\
    \phi_1 &= (m_1 l_1 + m_2 l_2) \cos(q_1) + m_2 l_2 \cos(q_1 + q_2) \\
    \phi_2 &= m_2 l_2 \cos(q_1 + q_2).
\end{align*}
Acrobot (details)

\[ q_1(0) = q_2(0) = -\pi/2, \quad \dot{q}_1(0) = \dot{q}_2(0) = 0, \text{ maximize final height} \]

\[ H = l_1 \sin q_1(T) + l_2 \sin q_2(T) \]
Acrobot (details)

\[ d_{11}(q)\ddot{q}_1 + d_{12}(q)\ddot{q}_2 + h_1(q, \dot{q}) + \phi_1(q) = 0 \]
\[ d_{21}(q)\ddot{q}_1 + d_{22}\ddot{q}_2 + h_2(q, \dot{q}) + \phi_2(q) = u \]

We can write these equations in standard form

\[ dx_i = f_i(x)dt + g_i(x)udt \]

with \( x_1 = q_1, x_2 = q_2, x_3 = \dot{q}_1, x_4 = \dot{q}_2 \) and

\[
\begin{align*}
  f_1(x) &= x_3 \\
  f_2(x) &= x_4 \\
  f_3(x) &= \frac{-d_{22}(h_1+\phi_1)+d_{12}(h_2+\phi_2)}{D} \\
  f_4(x) &= \frac{d_{12}(h_1+\phi_1)-d_{11}(h_2+\phi_2)}{D} \\
  g_1(x) &= 0 \\
  g_2(x) &= 0 \\
  g_3(x) &= -\frac{d_{12}}{D} \\
  g_4(x) &= \frac{d_{11}}{D}
\end{align*}
\]
100 iterations. At each iteration 50 stochastic trajectories were generated. The new control was computed from a deterministic trajectory. Noise was lowered at each iteration. Top left: final height for each stochastic trajectory for each iteration (red) and for each deterministic solution (blue).
Neural network 10 layers, 25 neurons per layer. Input is sin and cosine of both angles as well as angular velocity. No time as input. 100 iterations, with 10000 rollouts per iteration. Annealing such that ESS larger than 10 %. Took around 15 min with 100 cpu.
Integrated sensorimotor control

 Initialize control $u_0$
 for $t = 0, \ldots$ do
  $data_t = \text{act\_in\_the\_world}(u_t)$
  $model_t = \text{learn\_model}(u_t, data_t)$
  $u_{t+1} = \text{compute\_control}(model_t)$
 end for

 compute\_control
 for $k = 0, \ldots$ do
  $data_k = \text{generate\_data}(model, u_k)$ % Monte Carlo importance sampler
  $u_{k+1} = \text{learn\_control}(data_k, u_k)$ % Deep or recurrent learning
 end for
Integrated sensorimotor control

Initialize control $u_0$

\[
\text{for } t = 0, \ldots \text{ do} \\
\quad \text{data}_t = \text{act\_in\_the\_world}(u_t) \\
\quad \text{model}_t = \text{learn\_model}(u_t, \text{data}_t) \\
\quad u_{t+1} = \text{compute\_control(\text{model}_t)} \\
\text{end for}
\]

\text{compute\_control}

\[
\text{for } k = 0, \ldots \text{ do} \\
\quad \text{data}_k = \text{generate\_data(\text{model}, u_k)} \quad \% \text{Monte Carlo importance sampler} \\
\quad u_{k+1} = \text{learn\_control(\text{data}_k, u_k)} \quad \% \text{Deep or recurrent learning} \\
\text{end for}
\]

- generate infinite data to learn infinitely complex
Integrated sensorimotor control

Initialize control \( u_0 \)

\textbf{for} \( t = 0, \ldots \) \textbf{do}

\begin{align*}
data_t & = \text{act}\_\text{in}\_\text{the}\_\text{world}(u_t) \\
model_t & = \text{learn}\_\text{model}(u_t, data_t) \\
u_{t+1} & = \text{compute}\_\text{control}(model_t)
\end{align*}

\textbf{end for}

\textbf{compute}\_\text{control}

\textbf{for} \( k = 0, \ldots \) \textbf{do}

\begin{align*}
data_k & = \text{generate}\_\text{data}(model, u_k) \quad \% \text{Monte Carlo importance sampler} \\
u_{k+1} & = \text{learn}\_\text{control}(data_k, u_k) \quad \% \text{Deep or recurrent learning}
\end{align*}

\textbf{end for}

\begin{itemize}
  \item generate infinite data to learn infinitely complex
  \item \( data_t \) and \( data_k \) are the two realities of the brain
\end{itemize}
Towards sensorimotor integration

The brain is a Monte Carlo sampler

- Perception: Bayesian posterior computation
- Action: solving an optimal control problem through sampling

Both require the learning of a world model
Towards sensorimotor integration

The brain is a Monte Carlo sampler

- Perception: Bayesian posterior computation
- Action: solving an optimal control problem through sampling

Both require the learning of a world model

Action computation is optimized by adaptive importance sampling,

- this is a type of motor learning
- but is complemented by sampling ('halffabrikaat')
Towards sensorimotor integration

The brain is a Monte Carlo sampler

- Perception: Bayesian posterior computation
- Action: solving an optimal control problem through sampling

Both require the learning of a world model

Action computation is optimized by adaptive importance sampling,

- this is a type of motor learning
- but is complemented by sampling (‘halffabrikaat’)

Many open problems

- Sensing, acting interdependence
- action hierarchies in terms of action building blocks
Thank you!


www.snn.ru.nl/~bertk
KL control theory

$x$ denotes state of the agent and $x_{1:T}$ is a path through state space from time $t = 1$ to $T$.

$q(x_{1:T}|x_0)$ denotes a probability distribution over possible future trajectories given that the agent at time $t = 0$ is in state $x_0$, with

$$q(x_{1:T}|x_0) = \prod_{t=0}^{T} q(x_{t+1}|x_t)$$

$q(x_{t+1}|x_t)$ implements the allowed moves.

$R(x_{1:T}) = \sum_{t=1}^{T} R(x_t)$ is the total cost when following path $x_{1:T}$.

The KL control problem is to find the probability distribution $p(x_{1:T}|x_0)$ that minimizes

$$C(p|x_0) = \sum_{x_{1:T}} p(x_{1:T}|x_0) \left( \log \frac{p(x_{1:T}|x_0)}{q(x_{1:T}|x_0)} + R(x_{1:T}) \right) = KL(p||q) + \langle R \rangle_p$$
KL control theory

\( p(x_{1:T}|x_0) \) and \( q(x_{1:T}|x_0) \) distributions over trajectories.

Given \( q \), find \( p \) that minimizes

\[
C(p|x_0) = KL(p||q) - \langle R \rangle_p
\]

The solution and the optimal control cost are

\[
\begin{align*}
  p(x_{1:T}|x_0) &= \frac{1}{Z(x_0)} q(x_{1:T}|x_0) \exp(R(x_{1:T})) \\
  C &= -\log Z(x_0) \\
  Z(x_0) &= \sum_{x_{1:T}} q(x_{1:T}|x_0) \exp(R(x_{1:T}))
\end{align*}
\]

NB: \( Z(x_0) \) is an integral over paths.
KL control theory

The optimal control at time $t = 0$ is given by

$$p(x_1|x_0) = \sum_{x_{2:T}} p(x_{2:T}|x_0) \propto q(x_1|x_0) \exp(R(x_1))\beta_1(x_1)$$

with $\beta_t(x)$ the backward messages.

$$\beta_T(x_T) = 1$$
$$\beta_{t-1}(x_{t-1}) = \sum_{x_t} q(x_t|x_{t-1}) \exp(R(x_t))\beta_t(x_t)$$
Multi Agent cooperative game

Model of cooperation: either hunt a hare alone or a stag together.

<table>
<thead>
<tr>
<th></th>
<th>Stag</th>
<th>Hare</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stag</td>
<td>3, 3</td>
<td>0, 1</td>
</tr>
<tr>
<td>Hare</td>
<td>1, 0</td>
<td>1, 1</td>
</tr>
</tbody>
</table>

We define the KL-stag-hunt game as a multi-agent version where agents move on a grid to hunt stag or hare.
Approximate inference of the KL-stag-hunt problem

$\lambda = 10$

$\lambda = 0.1$

$M = 10$ agents, $N = 400$ locations, $10^{26}$ states per time slice

Sequential BP. If converges, converges in less than 500 iterations. Trajectories are marginal beliefs.
Phase transition (?)

\[ \frac{\partial}{\partial \lambda} \log Z_{BP} \]

Payoff dominant vs. Risk dominant

# BP iterations

\[ \text{M} = 4 \quad \text{M} = 10 \]
Average cost KL control (Todorov 2006)

When $T \to \infty$ and $q$ ergodic the backward message recursion

$$
\beta_{t-1}(x_{t-1}) = \sum_{x_t} H(x_{t-1}, x_t) \beta_t(x_t) \quad H(x, y) = q(y|x) \exp(R(y))
$$

becomes the computation of the Perron-Frobenius eigen pair $(\beta(\cdot), \lambda)$:

$$
H\beta = \lambda \beta \quad H(x, y) = q(y|x) \exp(R(x))
$$

The optimal control satisfies

$$
p(y|x) = q(y|x) \exp(R(x)) \frac{\beta(y)}{\lambda \beta(x)}
$$

$$
C = -\log \lambda
$$

$$
J(x) = -\log \beta(x)
$$
KL-learning [Bierkens, Kappen 2012]

- Goal: find Perron-Frobenius solution $Hz = \lambda z$, with $H = [\mathcal{q}(y|x) \exp(-R(x))]$, while stepping through state space according to $q$ and observing incurred cost.

- Algorithm (KL-learning):
  
  $z \leftarrow (1/n, \ldots, 1/n)$, $\lambda > 0$, $x \leftarrow$ any state
  
  for $m = 1 : M$ do
    $y \leftarrow$ independent draw from $q(\cdot|x)$
    $\Delta \leftarrow \exp(-R(x))z(y)/\lambda - z(x)$
    $z(x) \leftarrow z(x) + \gamma \Delta$
    $\lambda \leftarrow \lambda + \gamma \Delta$
    $x \leftarrow y$
  
  end for

- Invariants: $z > 0$, $\lambda = \|z\|_1$.

Generalization of $z$-learning (Todorov) to $\lambda \neq 1$
Numerical experiment
Summary and discussion

Control as inference links control to machine learning and statistical physics
- efficient computational methods
Summary and discussion

Control as inference links control to machine learning and statistical physics
- efficient computational methods
- insight in the role of noise: phase transitions (delayed choice and collaboration)
Summary and discussion

Control as inference links control to machine learning and statistical physics
- efficient computational methods
- insight in the role of noise: phase transitions (delayed choice and collaboration)
- favorable comparison with state-of-the-art RL methods in robotics (Theodorou 2010-2012)
Other topics

- Variational approximation, $n$ joint arm (Kappen tutorial 2011)
- Sampling approach to control of robotics arm (van den Broek 2011)
- Coordination of continuous agents using MF and BP (Wiegerinck et al. 2006, van den Broek et al. 2006)
- Risk sensitive path integral control (van den Broek 2010)
- Inference and control (Kappen tutorial 2011)
The variational method

Consider an arm consisting of \( n \) joints of length 1. The location of the \( i \)th joint in the 2d plane is

\[
    x_i = \sum_{j=1}^{i} \cos \theta_i \quad y_i = \sum_{j=1}^{i} \sin \theta_i
\]

with \( i = 1, \ldots, n \). Each of the joint angles is controlled by a variable \( u_i \). The dynamics of each joint is

\[
    d\theta_i = u_i dt + d\xi_i, \quad i = 1, \ldots, n
\]

with \( d\xi_i \) independent Gaussian noise with \( \langle d\xi_i^2 \rangle = \nu dt \). Denote by \( \vec{\theta} \) the vector of joint angles, and \( \vec{u} \) the vector of controls.
The variational method

The expected cost for the control path $\vec{u}_{t:T}$ is

$$C(\vec{\theta}, t, \vec{u}_{t:T}) = \left\langle \phi(\theta(T)) + \int_t^T \frac{1}{2} \vec{u}'(t)\vec{u}(t) \right\rangle$$

$$\phi(\vec{\theta}) = \frac{\alpha}{2} \left( (x_n(\vec{\theta}) - x_{\text{target}})^2 + (y_n(\vec{\theta}) - y_{\text{target}})^2 \right)$$

with $x_{\text{target}}, y_{\text{target}}$ the target coordinates of the end joint.
The variational method

Because $V = 0, f = 0, g = 1$, the solution to uncontrolled dynamics is Gaussian

$$
\psi(\vec{\theta}, t) = \int d\vec{\theta} \left( \frac{1}{\sqrt{2\pi \nu(T - t)}} \right)^n \exp \left( - \sum_{i=1}^{n} \left( \theta_i - \theta_i^0 \right)^2 / 2\nu(T - t) - \phi(\vec{\theta}) / \nu \right)
$$

The control at time $t$ for all components $i$ is computed from Eq. ?? and is given by

$$
u_i = \frac{1}{T - t} \left( \langle \theta_i \rangle - \theta_i^0 \right)
$$

where $\langle \theta_i \rangle$ is the expectation value of $\theta_i$ computed wrt the probability distribution

$$
p(\vec{\theta}) = \frac{1}{\psi(\vec{\theta}, t)} \exp \left( - \sum_{i=1}^{n} \left( \theta_i - \theta_i^0 \right)^2 / 2\nu(T - t) - \phi(\vec{\theta}) / \nu \right)
$$

This is not exactly correct because $\theta$ is a periodic variable. One should use the solution to diffusion on a circle instead. We can ignore this as long as $\sqrt{\nu(T - t)}$ is small compared to $2\pi$. 

---

Bert Kappen

ML 305
The variational method

We compute the expectations $\langle \vec{\theta} \rangle$ by introducing a factorized Gaussian variational distribution $q(\vec{\theta}) = \prod_{i=1}^{n} \mathcal{N}(\theta_i | \mu_i, \sigma_i)$. We compute $\mu_i$ and $\sigma_i$ by by minimizing the KL divergence between $q(\vec{\theta})$ and $p(\vec{\theta})$:

$$KL = \int d\theta q(\theta) \log \frac{q(\theta)}{p(\theta)}$$

$$= - \sum_{i=1}^{n} \log \sqrt{2\pi\sigma_i^2} + \log \psi(\vec{\theta}_0, t) + \frac{1}{2\nu(T - t)} \sum_{i=1}^{n} (\sigma_i^2 + (\mu_i - \theta_i^0)^2) + \frac{1}{\nu} \phi(\vec{\theta}) \rangle_q$$

where we omit irrelevant constants. $\phi(\vec{\theta})$ can be computed in closed form. Setting the derivative of the KL with respect to $\mu_i$ and $\sigma_i^2$ equal to zero:

$$\mu_i \leftarrow \theta_i^0 + \alpha(T - t) \left( \sin \mu_i e^{-\sigma_i^2/2} (\langle x_n \rangle - x_{\text{target}}) - \cos \mu_i e^{-\sigma_i^2/2} (\langle y_n \rangle - y_{\text{target}}) \right)$$

$$\frac{1}{\sigma_i^2} \leftarrow \frac{1}{\nu} \left( \frac{1}{T - t} + \alpha e^{-\sigma_i^2} - \alpha (\langle x_n \rangle - x_{\text{target}}) \cos \mu_i e^{-\sigma_i^2/2} - \alpha (\langle y_n \rangle - y_{\text{target}}) \sin \mu_i e^{-\sigma_i^2/2} \right)$$

Bert Kappen ML 306
The variational method
The variational method

(a) $t = 0.05$

(b) $t = 0.55$

(c) $t = 1.8$

(d) $t = 2.0$
The variational method

Note, that the computation of $\langle \theta_i \rangle$ solves the coordination problem between the
different joints. Once $\langle \theta_i \rangle$ is known, each $\theta_i$ is steered independently to its target value $\langle \theta_i \rangle$ using the control law Eq. 9. The computation of $\langle \theta_i \rangle$ in the variational approximation is very efficient and can be used to control arms with hundreds of joints.
Coordination of agents

$n$ agents with independent dynamics

\[ dx_\alpha = (f_\alpha(x_\alpha, t) + u_\alpha) + d\xi_\alpha, \quad \alpha = 1, \ldots, n \]

should coordinate their actions to minimize a cost at a future time $t = T$:

\[ \phi(y_1, \ldots, y_n) \quad y_\alpha \in \{z_1, \ldots, z_k\} \]

and $\phi = \infty$ elsewhere.
Coordination of agents

Then,

\[
\Psi(x_1, \ldots, x_n, t) = \int dy_1 \ldots dy_n \prod_{\alpha} \rho(y_\alpha, T|x_\alpha, t) \exp(-\phi(y_1, \ldots, y_n)/\nu)
\]

\[
= \sum_{\vec{y}} \exp(-E(\vec{y}|\vec{x}, t)/\nu)
\]

\[
p(\vec{y}) = \frac{1}{Z} \exp(-E(\vec{y}|\vec{x}, t)/\nu)
\]

\[
u_\alpha(\vec{x}, t) = -\partial_{x_\alpha} J = \left\langle \frac{\partial \log \rho(y_\alpha, T|x_\alpha, t)}{\partial x_\alpha} \right\rangle
\]

with \(\vec{x} = (x_1, \ldots, x_n), \vec{y} = (y_1, \ldots, y_n)\).

\(E\) has a graphical model structure if \(\phi\) has.
Pseudo code

Loop:

1. Compute the cost and its log derivative for each agent to move to each target:

   \[ \rho(z_i, T|x_\alpha, t), \quad i = 1, \ldots, k, \quad \alpha = 1, \ldots, n \]

   This path integral can be estimated using MC sampling or variational approximation.

2. Compute \( u_\alpha \) using graphical model inference in \( p(\vec{y}) \) (exact, BP, MF).
A simple 1d example

Intrinsic dynamics $f_\alpha = 0, V(x_1, \ldots, x_n) = 0$:

$$p(y_\alpha, T|x_\alpha, t) \propto \exp(- (y_\alpha - x_\alpha)^2 / 2\nu(T - t))$$

End cost $\phi(y_1, \ldots, y_n) = \sum_{j=1}^{k} (n_j(y) - n_j)^2$, with $n_j(y)$ the # of agents that go to target $j$.

Optimal control is for agent $\alpha$ is

$$u_\alpha = \frac{1}{T - t} (\langle y_\alpha \rangle - x_\alpha)$$
A simple 1d example

(i) Agent predicted target $\langle y_\alpha \rangle$

(j) Agent position $x$
A simple 1d example

**Control cost**
greedy control (red)
MF control (blue)
BP control (green)

**CPU time**
extact control (black)
MF control (blue)
BP control (green)
greedy control (red)
Nonlinear Coordination

Agents $a = 1, \ldots, n$ in $2D$:

\[
\begin{align*}
  dx_a(t) &= v_a(t) \cos \varphi_a(t) \, dt \\
  dy_a(t) &= v_a(t) \sin \varphi_a(t) \, dt \\
  dv_a(t) &= u_a(t) dt + d\xi_a(t) \\
  d\varphi_a(t) &= \omega_a(t) dt + d\zeta_a(t)
\end{align*}
\]

Initial states $O$, $v_a(0) = 0$, $\varphi_a(0) = 0$

Targets $X$, $v_a(T) = 0$, $\varphi_a(T) = 0$

Sample paths specified at $t_i = t + i \, dt$,

$i = 0, \ldots, 6$, $dt = (T - t)/6$
Inference methods:

- Junction Tree (· − ·)
- MF (—)

(100 sample paths per agent-target)

JT: exponential in number of agents
   (intractable for # agents > 10)

MF: polynomial in number of agents
Risk sensitive control

It is relatively straightforward to generalize the path integral method to optimize a cost of the form

\[ \tilde{C} = \phi(x_T) + \int \frac{1}{2} u'R u + V(x) \]

\[ C = \frac{1}{\theta} \log \langle \exp(\theta \tilde{C}) \rangle \]

For \( \theta = 0 \) the risk neutral control is recovered. For \( \theta \) small:

\[ C = \langle \tilde{C} \rangle + \frac{\theta}{2} \left( \langle \tilde{C}^2 \rangle - \langle \tilde{C} \rangle^2 \right) + h.o. \]

\( \theta > 0 \) is risk averse, \( \theta < 0 \) is risk seeking.

vd Broek et al. UAI 2010
Risk sensitive control

We illustrate the behavior for the (well known) LQ case. \( V = f = 0, \phi = \alpha/2x^2. \)

The optimal control is given by

\[
u = \frac{-\alpha x}{R + \alpha(T - t)(1 - \nu R \theta)}\]

For \( \theta < 0 \) control is weaker
For \( 0 < \theta < 1/R \nu \) control is stronger
In both cases control increases with time.

For \( \theta > 1/R \nu \), control is only well-defined when the denominator is positive:

\[
\alpha(T - t) < \frac{R}{\nu R \theta - 1}
\]

Control decreases with time. For larger time-to-go, the expected cost is infinite.

vd Broek et al. UAI 2010
Inference and control

As an example of the intricacies of joint inference and control, consider the simple LQ control problem [? , ?]

\[ \begin{align*}
   dx &= \alpha u dt + d\xi \\
   C(x_0, \theta_0, u(0 \rightarrow T)) &= \left\{ \phi(x(T)) + \int_0^T dt R(x, u, t) \right\}
\end{align*} \]

(11) (12)

with \( \alpha \) unobserved and \( x \) observed. Path cost \( R(x, u, t) \) and end cost \( \phi(x) \) and noise variance \( \nu \) are given.

Although \( \alpha \) is unobserved, we have a means to observe \( \alpha \) indirectly through the sequence \( x_t, u_t, t = 0, \ldots \). Each time step we observe \( dx \) and \( u \) and we can thus update our belief about \( \alpha \) using the Bayes formula:

\[ p_{t+dt}(\alpha|dx, u) \propto p(dx|\alpha, u)p_t(\alpha) \]

(13)

\( p(dx|\alpha, u) \) is Normal in \( dx \) with variance \( \nu dt \)
\( p_t(\alpha) \) our belief at time \( t \) about the values of \( \alpha \)
The information that we receive about $\alpha$ increases with $u$, because the $\alpha u dt$ term dominates the $d\xi$ term. However, large $u$ values are more costly and also may drive us away from our target state $x(T)$.

Thus, the optimal control is a balance between optimal inference and minimal control cost.

The solution is to augment the state space with parameters $\theta_t$ (sufficient statistics) that describe $p_t(\alpha) = p(\alpha|\theta_t)$ and $\theta_0$ known, which describes our initial belief in the possible values of $\alpha$. The cost that must be minimized is

$$C(x_0, \theta_0, u(0 \rightarrow T)) = \langle \phi(x(T)) + \int_0^T dtR(x, u, t) \rangle$$

(14)

where the average is with respect to the noise $d\xi$ as well as the uncertainty in $\alpha$.

NB: the average over $\alpha$ depends on $\theta_t$ which is not known beforehand.
For simplicity, consider the example that $\alpha$ attains only two values $\alpha = \pm 1$. Then $p_t(\alpha|\theta) = \sigma(\alpha \theta)$, with the sigmoid function $\sigma(x) = \frac{1}{2}(1 + \tanh(x))$. The update equation Eq. 13 implies a dynamics for $\theta$:

$$d\theta = \frac{u}{\nu} dx = \frac{u}{\nu}(\alpha u dt + d\xi)$$

With $z_t = (x_t, \theta_t)$ we obtain a standard HJB Eq.

$$-\partial_t J(t, z) dt = \min_u \left( R(t, x, u) dt + \langle dz \rangle_z \partial_z J(z, t) + \frac{1}{2} \langle d\xi^2 \rangle_z \partial^2_z J(z, t) \right)$$

with boundary condition $J(z, T) = \phi(x)$ (NB independent of $\theta$).

---

19 The rhs of the Bayes rule is

$$p(dx|\alpha, u)p(\alpha|\theta_t) \propto \exp\left(-\frac{(dx - \alpha u dt)^2}{2\nu dt}\right)\exp(\alpha \theta_t) \propto \exp\left(\frac{dx u}{\nu} + \alpha \theta_t\right) = \exp\left(\alpha \left(\theta_t + \frac{dx u}{\nu}\right)\right)$$
The result is

$$-\partial_t J = \min_u \left( R(x, u, t) + \bar{\alpha} u \partial_x J + \frac{u^2 \bar{\alpha}}{\nu} \partial_\theta J + \frac{1}{2} \nu \partial_x^2 J + \frac{1}{2} \nu \partial_\theta^2 J + u \partial_x \partial_\theta J \right)$$

with boundary conditions $J(x, \theta, T) = \phi(x)$.  

Thus, the dual control problem (joint inference on $\alpha$ and control problem on $x$) has become an ordinary control problem in $x, \theta$ (Florentin, 1962).

Note that if $R, \phi$ are quadratic and $\alpha$ is known, this is an LQ problem. However, when $\alpha$ is not known, the corresponding dual control problem is not LQ (because of the additional $u$ dependent terms).

\[20\] The expectation values appearing in this equation are conditioned on $(x_t, \theta_t)$ and are averages over $p(\alpha|\theta_t)$ and the Gaussian noise. $\langle dx \rangle_{x,\theta} = \bar{\alpha} u dt$, $\langle d\theta \rangle_{x,\theta} = \bar{\alpha} \nu dt$, $\langle dx^2 \rangle_{x,\theta} = \nu dt$, $\langle d\theta^2 \rangle_{x,\theta} = \frac{u^2}{\nu} dt$, $\langle dx d\theta \rangle = u dt$, with $\bar{\alpha} = \tanh(\theta)$ the expected value of $\alpha$ for a given value $\theta$. 

Bert Kappen
Probing

Dual control solution with end cost $\phi(x) = x^2$ and path cost $\int_t^f dt'\frac{1}{2}u(t')^2$ and $\nu = 0.5$. Plot shows the deviation of the control from the certain case: $u_t(x, \theta)/u_t(x, \theta = \pm\infty)$ as a function of $\theta$ for different values of $t$ and $x = 2$. The curves with the larger values are for larger times-to-go.

’Probing’: $u$ is much larger when $\alpha$ is uncertain ($\theta$ small) then when $\alpha$ is certain $\theta = \pm\infty$. 
Symmetry breaking and non-differentiability of $J$

The observed probing behavior arises as the result of a symmetry breaking in the right hand side of the Bellman equation.
Figure 1: Rhs of the Bellman equation as a function of $u$ and its derivative for $\theta = 0$. The different curves correspond to different values of $x$. Explorative behavior ($u_0$) arises in the no-knowledge state $\theta = 0$ by proposing non-zero controls. The singularity is absent at $t = T - 2$ and present starting from $t = T - 3$. 

$J(u)$

$\frac{dJ(u)}{du}$
Symmetry breaking and non-differentiability of $J$

As a result of the local minima in the Bellman optimization, the optimal value function is not differentiable.

The optimal cost-to-go is convex in the belief [?].

Left) $J_t(x, \theta)$ for $t = T - 2$, $x = -2$ (grey) and $t = T - 2$, $x = -6$ (black) versus $\theta$ Right) Same as left, but as a function of the belief $p = p(b = 1|\theta)$. 
Controlled noisy Lorenz attractor

\[ u(t, x) = A(t)x + b(t). \quad N = 6000 \]
Further reading

http://www.snn.ru.nl/~bertk/
http://www.snn.ru.nl

References


