Deep Learning - Foundations and Concepts

Chapter 14. Sampling

nonlineark@github

March 30, 2025

Outline

Expectations

For some applications the goal is to evaluate expectations with respect to the distribution. Suppose we wish to find the expectation of a function $f(z)$ with respect to a probability distribution $p(z)$:

$$E(f) = \int f(z)p(z)\mathrm{d}z$$

The general idea behind sampling methods is to obtain a set of samples $z^{(l)}$ drawn independently from the distribution $p(z)$. This allows the expectation to be approximated by a finite sum:

$$\bar{f} = \frac{1}{L}\sum_{l=1}^{L} f(z^{(l)})$$

Expectations

Let's calculate the expectation and variance of $\bar{f}$:

$$E(\bar{f}) = E(\frac{1}{L}\sum_{l=1}^{L} f(z^{(l)})) = E(f)$$

$$E(\bar{f}^2) = E(\frac{1}{L^2}\sum_{l,l'} f(z^{(l)})f(z^{(l')})) = (E(f))^2 + \frac{1}{L}\mathrm{var}(f)$$

$$\mathrm{var}(\bar{f}) = E(\bar{f}^2) - (E(\bar{f}))^2 = \frac{1}{L}\mathrm{var}(f)$$

Which shows that:

- $\bar{f}$ is an unbiased estimator of $E(f)$.
- Due to the linear decrease of the variance with increasing L, in principle, high accuracy may be achievable with a relatively small number of samples $z^{(l)}$.

Standard distributions

Problem

Suppose that z is uniformly distributed over the interval $(0, 1)$. Given a probability density function p, find a function g such that the random variable $y = g(z)$ has p as its probability density function.

Standard distributions

Let U be the probability density function of the uniform distribution over the interval $(0, 1)$, we have:

$$
\begin{aligned}
p(y)\mathrm{d}y &= U(z)\mathrm{d}z \\
f(y_0) = \int_{-\infty}^{y_0} p(y)\mathrm{d}y &= \int_{-\infty}^{z_0} U(z)\mathrm{d}z = z_0 \\
y_0 &= f^{-1}(z_0)
\end{aligned}
$$

So we have to transform the uniformly distributed random numbers using a function that is the inverse of the cumulative distribution function of the desired probability density function.

Standard distributions

Some examples:

- Exponential distribution $p(y) = \lambda \exp(-\lambda y)$:
  - $z = f(y) = \int_0^y p(t)\mathrm{d}t = 1 - \exp(-\lambda y)$.
  - $y = -\frac{1}{\lambda} \log(1 - z)$.
- Cauchy distribution $p(y) = \frac{1}{\pi} \frac{1}{1+y^2}$:
  - $z = f(y) = \int_{-\infty}^y p(t)\mathrm{d}t = \frac{1}{\pi} \arctan y + \frac{1}{2}$.
  - $y = \tan(\pi(z - \frac{1}{2}))$.

Standard distributions

The generalization to multiple variables involves the Jacobian of the change of variables, so that:

$$p_Y(y_1, \ldots, y_M) = p_Z(z_1, \ldots, z_M) \left| \frac{\partial(z_1, \ldots, z_M)}{\partial(y_1, \ldots, y_M)} \right|$$

Standard distributions

The Box-Muller method for generating samples from a Gaussian distribution. First, suppose we generate pairs of uniformly distributed random numbers $z_1, z_2 \in (-1, 1)$. Next, we discard each pair unless it satisfies $z_1^2 + z_2^2 \leq 1$. This leads to a uniform distribution of points inside the unit circle with $p_Z(z_1, z_2) = \frac{1}{\pi}$. Then, for each pair z_1, z_2 we evaluate the quantities:

$$y = z\frac{\sqrt{-4\log||z||}}{||z||}$$

The joint distribution of y_1 and y_2 is given by:

$$p_Y(y_1, y_2) = p_Z(z_1, z_2)|\frac{\partial(z_1, z_2)}{\partial(y_1, y_2)}| = (\frac{1}{\sqrt{2\pi}}\exp(-\frac{y_1^2}{2}))(\frac{1}{\sqrt{2\pi}}\exp(-\frac{y_2^2}{2}))$$

So y_1 and y_2 are independent and each has a Gaussian distribution with zero mean and unit variance.

Rejection sampling

Suppose that:

- We wish to sample from a distribution $p(z)$, and sampling directly from $p(z)$ is difficult.
- We are easily able to evaluate $p(z)$ for any given value of z, up to some normalizing constant Z_p, so that $p(z) = \frac{1}{Z_p}\tilde{p}(z)$, where $\tilde{p}(z)$ can readily be evaluated, but Z_p is unknown.

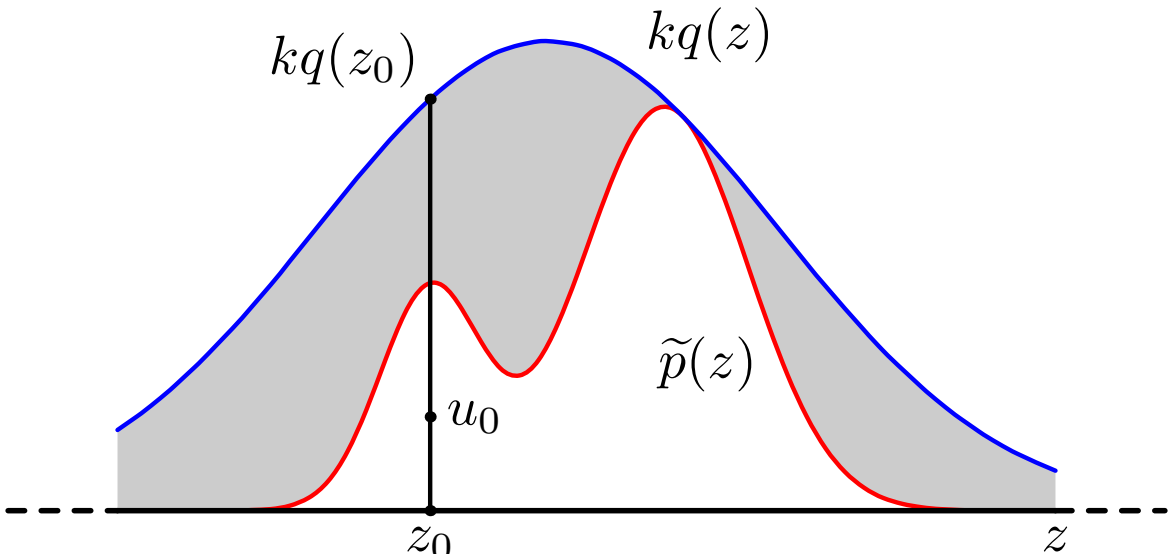
Rejection sampling

To apply rejection sampling:

- Find a simpler distribution $q(z)$, called a proposal distribution, from which we can readily draw samples.
- Introduce a constant k whose value is chosen such that $kq(z) \geq \tilde{p}(z)$ for all values of z.
- Generate a number z_0 from the distribution $q(z)$.
- Generate a number u_0 from the uniform distribution over $[0, kq(z_0)]$.
- If $u_0 > \tilde{p}(z_0)$ then the sample is rejected, otherwise u_0 is retained.
- The corresponding z values in the remaining pairs are distributed according to $p(z)$.

Rejection sampling

Figure: Illustration of the rejection sampling method



Rejection sampling

Let's verify the correctness of the rejection sampling method. Suppose that random variable Z is distributed according to $q(z)$, and random variable U is uniformly distributed over $[0, kq(Z)]$. We want to calculate the probability density function of the random variable $Z|0 \leq U \leq \tilde{p}(Z)$:

$$\begin{aligned} P(Z \in E|0 \leq U \leq \tilde{p}(Z)) &= \frac{P(Z \in E, 0 \leq U \leq \tilde{p}(Z))}{P(0 \leq U \leq \tilde{p}(Z))} \\ &= \frac{\int_E q(z) \frac{\tilde{p}(z)}{kq(z)} \mathrm{d}z}{\int_{\mathbb{R}} q(z) \frac{\tilde{p}(z)}{kq(z)} \mathrm{d}z} \\ &= \int_E p(z) \mathrm{d}z \end{aligned}$$

We see that the random variable $Z|0 \leq U \leq \tilde{p}(Z)$ is indeed distributed according to $p(z)$.

Rejection sampling

Let's calculate the probability that a sample will be accepted:

$$P_{\text{accept}} = \int_{\mathbb{R}} q(z) \frac{\tilde{p}(z)}{kq(z)} \mathrm{d}z = \frac{Z_p}{k}$$

We see that the constant k should be as small as possible subject to the limitation that $kq(z)$ must be nowhere less than $\tilde{p}(z)$.

Adaptive rejection sampling

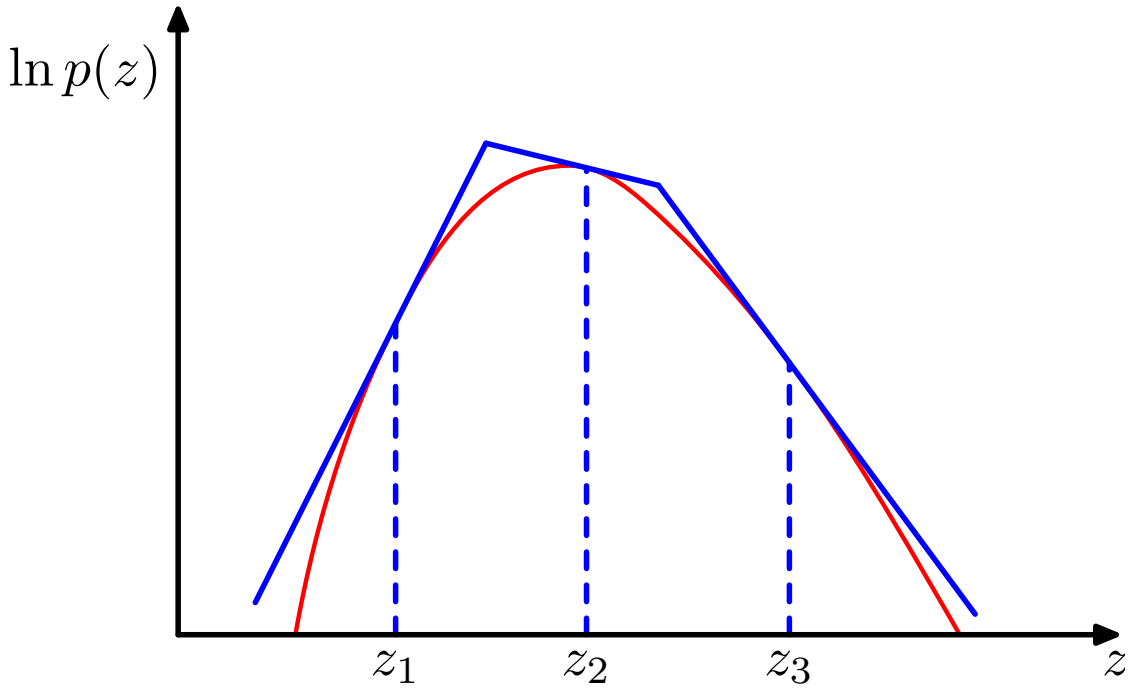
- In many instances, it can be difficult to determine a suitable analytic form for the envelope distribution $q(z)$.
- An alternative approach is to construct the envelope function on the fly based on measured values of the distribution $p(z)$.
- Constructing an envelope function is particularly straightforward when $p(z)$ is log concave.

Adaptive rejection sampling

- Evaluate the function $\log p(z)$ and its gradient at some initial set of grid points.
- The intersections of the resulting tangent lines are used to construct the envelope function.
- Draw a sample value from the envelope distribution. This is straightforward because the envelope function comprises a piecewise exponential distribution.
- If the sample is accepted, then it will be a draw from the desired distribution.
- If the sample is rejected, then it is incorporated into the set of grid points, a new tangent line is computed, and the envelope function is thereby refined.

Adaptive rejection sampling

Figure: Illustration of the construction of an envelope function for adaptive rejection sampling



Importance sampling

The technique of importance sampling provides a framework for approximating expectations directly but does not itself provide a mechanism for drawing samples from a distribution $p(z)$.

Suppose we wish to calculate the expectation of a function $f(z)$ with respect to a distribution $p(z)$:

- The distribution $p(z)$ can be evaluated only up to a normalization constant, so that $p(z) = \frac{\tilde{p}(z)}{Z_p}$, where Z_p is unknown.
- Because it is difficult to draw samples directly from $p(z)$, we rely on a proposal distribution $q(z)$ from which it is easy to draw samples.
- The distribution $q(z)$ also has an unknown normalization constant Z_q, so that $q(z) = \frac{\tilde{q}(z)}{Z_q}$.

Importance sampling

Let's calculate the expectation of $f(z)$ with respect to $p(z)$:

$$E(f) = \int f(z)p(z)\mathrm{d}z = \int f(z)\frac{p(z)}{q(z)}q(z)\mathrm{d}z$$

$$\approx \frac{1}{L}\sum_{l=1}^{L} f(z^{(l)})\frac{p(z^{(l)})}{q(z^{(l)})} = \frac{Z_q}{Z_p}\frac{1}{L}\sum_{l=1}^{L} f(z^{(l)})\frac{\tilde{p}(z^{(l)})}{\tilde{q}(z^{(l)})}$$

$$\frac{Z_p}{Z_q} = \frac{1}{Z_q}\int \tilde{p}(z)\mathrm{d}z = \frac{1}{Z_q}\int \frac{\tilde{p}(z)}{q(z)}q(z)\mathrm{d}z$$

$$\approx \frac{1}{Z_q}\frac{1}{L}\sum_{l=1}^{L}\frac{\tilde{p}(z^{(l)})}{q(z^{(l)})} = \frac{1}{L}\sum_{l=1}^{L}\frac{\tilde{p}(z^{(l)})}{\tilde{q}(z^{(l)})}$$

where the samples $\{z^{(l)}\}$ are drawn from $q(z)$.

Importance sampling

Let:

$$\tilde{r}_l = \frac{\tilde{p}(z^{(l)})}{\tilde{q}(z^{(l)})}$$

$$w_l = \frac{\tilde{r}_l}{\sum_{l'} \tilde{r}_{l'}}$$

we see that:

$$E(f) \approx \frac{\sum_{l=1}^{L} f(z^{(l)}) \tilde{r}_l}{\sum_{l=1}^{L} \tilde{r}_l} = \sum_{l=1}^{L} w_l f(z^{(l)})$$

Sampling-importance-resampling

- Draw L samples $z^{(1)}, \ldots, z^{(L)}$ from $q(z)$.
- Construct weights $w_1, \ldots, w_L$ using $w_l = \frac{\tilde{r}_l}{\sum_{l'} \tilde{r}_{l'}} = \frac{\tilde{p}(z^{(l)})/q(z^{(l)})}{\sum_{l'} \tilde{p}(z^{(l')})/q(z^{(l')})}$.
- Draw L samples from the discrete distribution $(z^{(1)}, \ldots, z^{(L)})$ with probabilities given by the weights $(w_1, \ldots, w_L)$.

Sampling-importance-resampling

Let's verify the correctness of the sampling-importance-resampling method.

$$P(z \leq a) = \sum_{l=1}^{L} I(z^{(l)} \leq a) w_l = \frac{\sum_{l=1}^{L} I(z^{(l)} \leq a) \frac{\tilde{p}(z^{(l)})}{q(z^{(l)})}}{\sum_{l=1}^{L} \frac{\tilde{p}(z^{(l)})}{q(z^{(l)})}}$$

where I is the indicator function. Taking the limit $L \to \infty$:

$$P(z \leq a) = \frac{\int I(z \leq a) \frac{\tilde{p}(z)}{q(z)} q(z) \mathrm{d}z}{\int \frac{\tilde{p}(z)}{q(z)} q(z) \mathrm{d}z} = \frac{\int_{-\infty}^{a} \tilde{p}(z) \mathrm{d}z}{\int_{-\infty}^{\infty} \tilde{p}(z) \mathrm{d}z} = \int_{-\infty}^{a} p(z) \mathrm{d}z$$

Markov Chain Monte Carlo

Suppose that:

- We wish to sample from a distribution $p(z)$, and sampling directly from $p(z)$ is difficult.
- We are easily able to evaluate $p(z)$ for any given value of z, up to some normalizing constant Z_p, so that $p(z) = \frac{1}{Z_p}\tilde{p}(z)$, where $\tilde{p}(z)$ can readily be evaluated, but Z_p is unknown.
- We maintain a record of the current state $z^{(\tau)}$, and the proposal distribution $q(z|z^{(\tau)})$ is conditioned on this current state.
- At each cycle of the algorithm, we generate a candidate sample z^* from the proposal distribution and then accept the sample according to an appropriate criterion.

The Metropolis algorithm

In the basic Metropolis algorithm, we assume that the proposal distribution is symmetric, that is $q(z_A|z_B) = q(z_B|z_A)$ for all values of z_A and z_B. The candidate sample is then accepted with probability:

$$A(z^*, z^{(\tau)}) = \min(1, \frac{\tilde{p}(z^*)}{\tilde{p}(z^{(\tau)})})$$

As we will see, if $q(z_A|z_B)$ is positive for any values of z_A and z_B, the distribution of $z^{(\tau)}$ tends to $p(z)$ as $\tau \to \infty$.

The Metropolis algorithm

Algorithm 1: Metropolis sampling

$z_{\text{prev}} \leftarrow z^{(0)}$;
for $\tau \leftarrow 1$ **to** T **do**
    $z^* \sim q(z|z_{\text{prev}})$;
    $u \sim \mathcal{U}(0, 1)$;
    if $\frac{\tilde{p}(z^*)}{\tilde{p}(z_{\text{prev}})} > u$ **then**
        $z_{\text{prev}} \leftarrow z^*$;
    end
end
return z_{prev};

Markov chains

For a first-order Markov chain $z^{(1)}, \ldots, z^{(M)}, \ldots$:

- The transition probability $T_m(z^{(m)}, z^{(m+1)})$ from $z^{(m)}$ to $z^{(m+1)}$ is defined as $p(z^{(m+1)}|z^{(m)})$.
- A Markov chain is called homogeneous if the transition probabilities are the same for all m.
- A distribution is said to be invariant with respect to a Markov chain if the marginal distributions $p(z^{(m)})$ are invariant.

Markov chains

A transition probability $T(z, z')$ is said to satisfy detailed balance with respect to a distribution $p(z)$ if:

$$p(z)T(z, z') = p(z')T(z', z)$$

It is easily seen that $p(z)$ is invariant:

$$\int p(z)T(z, z')\mathrm{d}z = \int p(z')T(z', z)\mathrm{d}z = p(z') \int p(z|z')\mathrm{d}z = p(z')$$

A Markov chain that respects detailed balance is said to be reversible.

Markov chains

Our goal is to use Markov chains to sample from a given distribution $p^*(z)$:

- We can achieve this if we set up a Markov chain such that $p^*(z)$ is invariant.
- We must also require that for $m \to \infty$, the distribution $p(z^{(m)})$ converges to $p^*(z)$, irrespective of the choice of initial distribution $p(z^{(0)})$. This property is called ergodicity, and the invariant distribution is then called the equilibrium distribution.

The Metropolis-Hastings algorithm

The Metropolis-Hastings algorithm, which is a generalization of the basic Metropolis algorithm, applies when the proposal distribution is no longer a symmetric function of its arguments. The candidate sample is accepted with probability:

$$A_k(z^*, z^{(\tau)}) = \min(1, \frac{\tilde{p}(z^*)q_k(z^{(\tau)}|z^*)}{\tilde{p}(z^{(\tau)})q_k(z^*|z^{(\tau)})})$$

Here k labels the members of the set of possible transitions being considered. For a symmetric proposal distribution, the Metropolis-Hastings criterion reduces to the standard Metropolis criterion.

The Metropolis-Hastings algorithm

Algorithm 2: Metropolis-Hastings sampling

$z_{\text{prev}} \leftarrow z^{(0)}$;
for $\tau \leftarrow 1$ **to** T **do**
    $k \leftarrow M(\tau)$;
    $z^* \sim q_k(z|z_{\text{prev}})$;
    $u \sim \mathcal{U}(0,1)$;
    if $\frac{\tilde{p}(z^*)q_k(z_{\text{prev}}|z^*)}{\tilde{p}(z_{\text{prev}})q_k(z^*|z_{\text{prev}})} > u$ **then**
        $z_{\text{prev}} \leftarrow z^*$;
    end
end
return z_{prev};

The Metropolis-Hastings algorithm

We can show that $p(z)$ is an invariant distribution of the Markov chain defined by the Metropolis-Hastings algorithm by showing that detailed balance is satisfied:

$$\begin{aligned} p(z)T_k(z, z') &= p(z)q_k(z'|z)A_k(z', z) \\ &= \min(p(z)q_k(z'|z), p(z')q_k(z|z')) \\ &= \min(p(z')q_k(z|z'), p(z)q_k(z'|z)) \\ &= p(z')q_k(z|z')A_k(z, z') \\ &= p(z')T_k(z', z) \end{aligned}$$

Gibbs sampling

Algorithm 3: Gibbs sampling

for $\tau \leftarrow 1$ **to** T **do**
  for $m \leftarrow 1$ **to** M **do**
    $z_m \sim p(z_m | \{z_{m' \neq m}\})$;
  end
end
return $\{z_1, \ldots, z_M\}$;

Gibbs sampling

Let's first verify that the distribution $p(z)$ is an invariant of each of the Gibbs sampling steps individually. Suppose at a certain step, z is transitioned to z', and the variable that has been updated is z_m:

$$\begin{aligned}
&\int T(z, z')p(z)\mathrm{d}z \\
&= \int p(z'_m|z'_{m'\neq m})p(z'_1, \ldots, z'_{m-1}, z_m, z'_{m+1}, \ldots, z'_M)\mathrm{d}z_m \\
&= p(z'_m|z'_{m'\neq m})p(z'_{m'\neq m}) = p(z')
\end{aligned}$$

To ensure that the Gibbs sampling procedure samples from the correct distribution, it has to be ergodic. A sufficient condition for ergodicity is that none of the conditional distributions are anywhere zero.

Gibbs sampling

The Gibbs sampling procedure is a particular instance of the Metropolis-Hastings algorithm. Consider a Metropolis-Hastings sampling step, at which z is transitioned to z^*, and the variable that has been updated is z_m. The factor that determines the acceptance probability is given by:

$$\begin{aligned} A(z^*, z) &= \min(1, \frac{p(z^*)q_m(z|z^*)}{p(z)q_m(z^*|z)}) \\ &= \min(1, \frac{p(z_m^*|z_{m' \neq m}^*)p(z_{m' \neq m}^*)p(z_m|z_{m' \neq m}^*)}{p(z_m|z_{m' \neq m})p(z_{m' \neq m})p(z_m^*|z_{m' \neq m})}) \\ &= 1 \end{aligned}$$

Thus the Metropolis-Hastings steps are always accepted.

Gibbs sampling

One approach to reducing the random walk behavior in Gibbs sampling is called over-relaxation. It applies to problems for which the conditional distributions are Gaussian. At each step of the Gibbs sampling algorithm, the conditional distribution for a particlar component z_m has some mean μ_m and some variance σ_m^2. In the over-relaxation framework, the value of z_m is replaced with:

$$z'_m = \mu_m + \alpha_m(z_m - \mu_m) + \sigma_m\sqrt{1-\alpha_m^2}\nu$$

where ν is a Gaussian random variable with zero mean and unit variance, and α is a parameter such that $-1 < \alpha < 1$. This step leaves the desired distribution invariant because z'_m also has mean μ_m and variance σ_m^2. The effect of over-relaxation is to encourage directed motion through state space when the variables are highly correlated.

Ancestral sampling

For a directed graph with no observed variables, it is straightforward to sample from the joint distribution using the following ancestral sampling approach:

$$p(z) = \prod_{m=1}^{M} p(z_m|\mathrm{pa}(m))$$

To obtain a sample from the joint distribution, we make one pass through the set of variables in the order $z_1, \ldots, z_M$ sampling from the conditional distributions $p(z_m|\mathrm{pa}(m))$.

Ancestral sampling

Consider a directed graph in which some of the nodes, which comprise the evidence set $\mathcal{E}$, are instantiated with observed values. The likelihood weighted sampling method:

- For each variable in turn:
  - If that variable is in the evidence set, the it is just set to its instantiated value.
  - If it is not in the evidence set, then it is sampled from the conditional distribution $p(z_m|\mathrm{pa}(m))$.
- The weighting associated with the resulting sample z is then given by $r(z) = \prod_{z_m \notin \mathcal{E}} \frac{p(z_m|\mathrm{pa}(m))}{p(z_m|\mathrm{pa}(m))} \prod_{z_m \in \mathcal{E}} \frac{p(z_m|\mathrm{pa}(m))}{1} = \prod_{z_m \in \mathcal{E}} p(z_m|\mathrm{pa}(m))$.

Langevin sampling

- The Metropolis-Hastings algorithm can be relatively inefficient since the proposal distribution is often a simple, fixed distribution that can generate updates in any direction in the data space, leading to a random walk.
- We can introduce Markov chain sampling algorithms that make use of the gradient of the probability density with respect to the data vector so as to take steps that preferentially move towards regions of higher probability.
- Langevin sampling avoids the use of an acceptance test, and arises in the context of machine learning models defined in terms of energy functions.

Energy-based models

- The energy function $E(x; w)$ is a real valued function of its arguments with no other constraints.
- The exponential $\exp(-E(x; w))$ can be viewed as an unnormalized probability distribution over x:
  - The introduction of the minus sign means that higher values of energy correspond to lower values of probability.
- We can then define a normalized distribution using $p(x; w) = \frac{1}{Z(w)} \exp(-E(x; w))$:
  - $Z(w)$, known as the partition function, is defined by $Z(w) = \int \exp(-E(x; w))\mathrm{d}x$.
- The energy function is often modelled using a deep neural network with input vector x and a scalar output $E(x; w)$, where w represents the weights and biases in the network.

Energy-based models

- The big advantage of energy-based models is their flexibility in that they bypass the requirement for normalization.
- A corresponding disadvantage is that since the normalizing constant is unknown, they can be more difficult to train:
  - To compute the gradient of the log likelihood with respect to w, we need to know the form of $Z(w)$.
  - However, for many choices of the energy function $E(x; w)$, it will be impractical to evaluate the partition function.

Maximizing the likelihood

We can make use of Monte Carlo sampling methods to approximate the gradient of the log likelihood with respect to the model parameters. For a single data point x:

$$\nabla_w \log p(x; w) = -\nabla_w E(x; w) - \nabla_w \log Z(w)$$

Assume that the data points from the training set are drawn independently from some unknown distribution $p_{\mathcal{D}}(x)$:

$$E_{x \sim p_{\mathcal{D}}(x)}(\nabla_w \log p(x; w)) = -E_{x \sim p_{\mathcal{D}}(x)}(\nabla_w E(x; w)) - \nabla_w \log Z(w)$$

Maximizing the likelihood

Let's calculate $-\nabla_w \log Z(w)$:

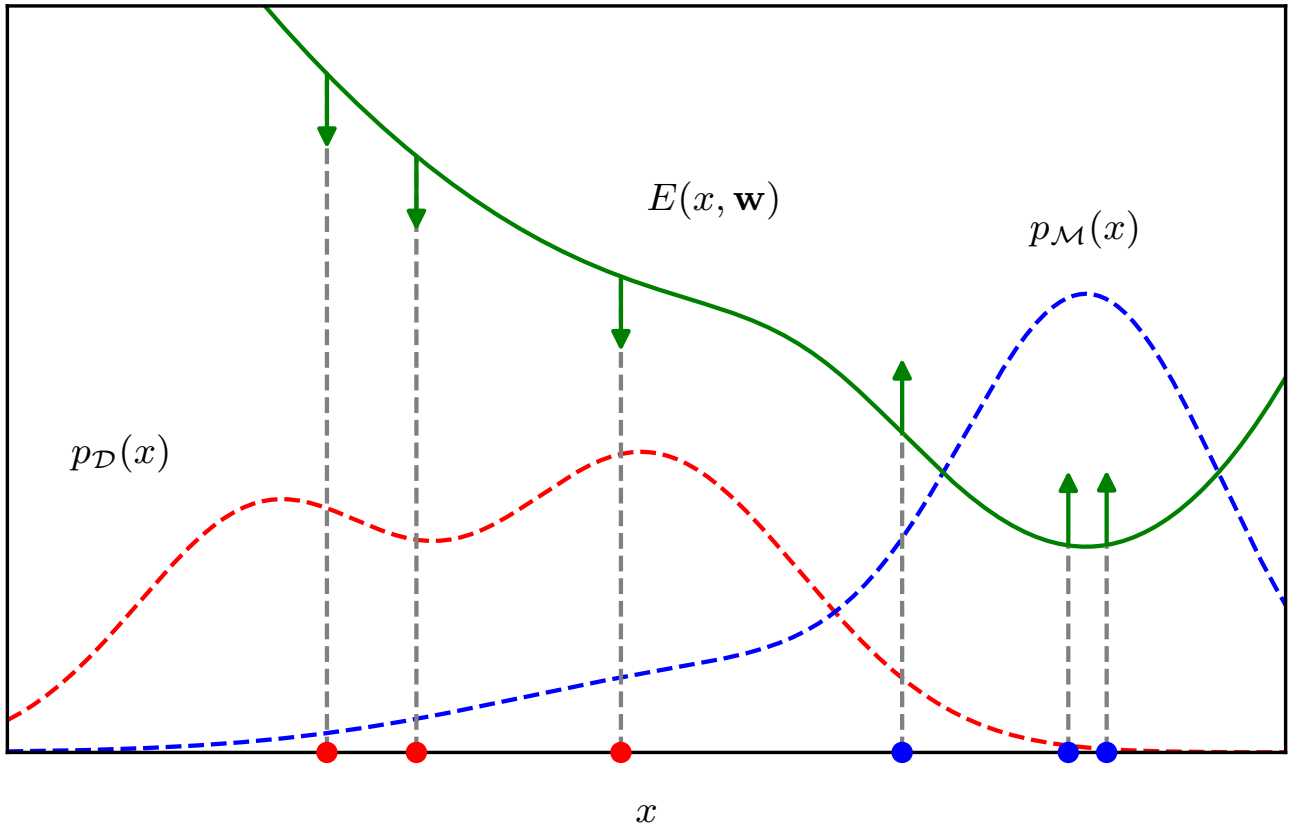
$$
\begin{aligned}
-\nabla_w \log Z(w) &= -\frac{1}{Z(w)} \nabla_w \int \exp(-E(x;w)) \mathrm{d}x \\
&= -\frac{1}{Z(w)} \int \exp(-E(x;w))(-\nabla_w E(x;w)) \mathrm{d}x \\
&= \int \nabla_w E(x;w) p(x;w) \mathrm{d}x = E_{x \sim p_{\mathcal{M}}(x)}(\nabla_w E(x;w))
\end{aligned}
$$

where the model distribution $p(x;w)$ is written as $p_{\mathcal{M}}(x)$ in the expectation to make it clear. Now we have:

$$
\begin{aligned}
&E_{x \sim p_{\mathcal{D}}(x)}(\nabla_w \log p(x;w)) \\
&= -E_{x \sim p_{\mathcal{D}}(x)}(\nabla_w E(x;w)) + E_{x \sim p_{\mathcal{M}}(x)}(\nabla_w E(x;w))
\end{aligned}
$$

Maximizing the likelihood

Figure: Illustration of the training of an energy-based model by maximizing the likelihood



Langevin dynamics

When training an energy-based model, we need to approximate the two terms on the right-hand side. For any given value of x, we can evaluate $\nabla_w E(x; w)$ using automatic differentiation. For the first term, we can use the training data set to estimate the expectation over x:

$$E_{x \sim p_{\mathcal{D}}(x)}(\nabla_w E(x; w)) \approx \frac{1}{N} \sum_{n=1}^{N} \nabla_w E(x_n; w)$$

Langevin dynamics

For the second term, we need to draw samples from the model distribution. This can be done using Markov chain Monte Carlo methods. One popular approach is called Langevin sampling. We start by drawing an initial value $x^{(0)}$ from a prior distribution, and then we iterate the following Markov chain steps:

$$x^{(\tau+1)} = x^{(\tau)} + \eta \nabla_x \log p(x^{(\tau)}; w) + \sqrt{2\eta}\epsilon^{(\tau)}$$

where $\epsilon^{(\tau)} \sim \mathcal{N}(\epsilon; 0, I)$ are independent samples from a zero-mean, unit-covariance Gaussian distribution, and the parameter η controls the step size. It can be shown that, in the limits of $\eta \to 0$ and $\tau \to \infty$, the value of $x^{(\tau)}$ is an independent sample from the distribution $p(x)$.

Langevin dynamics

One more word about $\nabla_x \log p(x; w)$, which is called the score function:

$$\begin{aligned}
\nabla_x \log p(x; w) &= \frac{1}{p(x; w)} \nabla_x \left(\frac{1}{Z(w)} \exp(-E(x; w)) \right) \\
&= \frac{1}{p(x; w)} \frac{1}{Z(w)} \exp(-E(x; w))(-\nabla_x E(x; w)) \\
&= -\nabla_x E(x; w)
\end{aligned}$$

Langevin dynamics

Algorithm 4: Langevin sampling

$x \leftarrow x_0$;

for $\tau \leftarrow 1$ **to** T **do**

$\epsilon \sim \mathcal{N}(\epsilon; 0, I)$;

$x \leftarrow x + \eta \nabla_x \log p(x; w) + \sqrt{2\eta}\epsilon$;

end

return x;

Langevin dynamics

We can repeat the Langevin sampling process to generate a set of samples $\{x_1, \ldots, x_M\}$ from the model distribution and then approximate the second term using:

$$E_{x \sim p_{\mathcal{M}}(x)}(\nabla_w E(x; w)) \approx \frac{1}{M} \sum_{m=1}^{M} \nabla_w E(x_m; w)$$

Langevin dynamics

Running long Markov chains to generate independent samples can be computationally expensive, and so we need to consider practical approximations. One approach is called contrastive divergence:

- Running a Monte Carlo chain starting with one of the training data points x_n.
- Running for only a few steps of Monte Carlo, perhaps even as few as one step.
- The resulting sample will be far from unbiased and will lie close to the data manifold.
- This can prove effective for tasks such as discrimination but is expected to be less effective in learning a generative model.