MTH210a: Statistical Computing

Instructor: Dootika Vats

Semester II, 2022 - 2023

Contents

1 Lecture-wise Summary

Lec No.	Date	Topic
1	Jan 9	FCH and Pseudorandom number generation
2	Jan 9	Pseudorandom numbers and inverse transform
3	Jan 10	Accept-Reject
4	Jan 11	Accept-Reject and composition method
5	Jan 16	Continuous: inverse transform and accept-reject
6	Jan 17	Accept-reject
7	Jan 18	Accept-reject
8	Jan 23	Box-Muller, Ratio-of-uniforms, Quiz 1
9	Jan 24	Ratio-of-uniforms
10	Jan 25	Ratio-of-uniforms, Miscellaneous sampling
11	Jan 30	Multivariate normal sampling, Simple Monte Carlo
12	Jan 31	Simple Monte Carlo, simple importance sampling
13	Feb 1	Simple importance sampling
14	Feb 6	Optimal proposal example, Weighted importance sampling
15	Feb 7	Weighted importance sampling
16	Feb 8	Likelihood function, Maximum Likelihood Estimator
17	Feb 13	MLE and linear regression
18	Feb 14	Penalized regression, Quiz 2
19	Feb 15	Newton-Raphson Algorithm
–	Feb 20 - 24	Mid-sem Exam week
20	Feb 27	Newton-Raphson Algorithm
21	Feb 28	Gradient Ascent Algorithm
22	Feb 28 (extra class)	Logistic regression GA algorithm
23	Mar 1	More Logistic regression
	Mar 6 - 17	Mid-sem Recess + 1 week I was away
24	Mar 20	MM Algorithm
25	Mar 21	Bridge regression
26	Mar 21 (extra class)	EM Algorithm
27	Mar 22	EM Algorithm
28	Mar 27	Gaussian Mixture Model
29	Mar 28	Gaussian Mixture
30	Mar 28 (extra class)	Gaussian Mixture still
31	Mar 29	Censored data example
32	Apr 3	Loss functions
33	Apr 5	Cross-validation
34	Apr 10	Bootstrapping
35	Apr 11	Bootstrapping

2 Pseudorandom Number Generation

The building block of computational simulation is the generation of uniform random numbers. If we can draw from $U(0,1)$, then we can draw from *most* other distributions. Thus the construction of sampling from $U(0,1)$ requires special attention.

Computers can generate numbers between $(0,1)$, which although are not exactly random (and in fact deterministic), but have the appearance of being $U(0,1)$ random variables. These draws from $U(0,1)$ are *pseudorandom* draws.

The goal in *pseudorandom* generation is to draw

$$X_1, \dots, X_n \overset{\text{approx iid}}{\sim} U(0,1)\,.$$

The resultant sample is as uniformly distributed as possible, and as independent as possible. We will learn about two different pseudorandom generators. These are very basic ones that are actually not really used in real life, but make our point well.

Note: After this lecture, we will always assume that all $U(0,1)$ draws are exactly iid and perfectly random. We will forget that they are infact, pseudorandom. Pseudorandom generation is a whole field in itself; for more on this, checkout CS744 at IITK.

2.1 Multiplicative congruential method

A common algorithm to generate a sequence $\{x_n\}$ is the *multiplicative congruential method*:

1. Set *seed* x_0, and positive integers a, m.
2. Obtain $x_t = a\,x_{t-1} \mod m$
3. Return sequence x_t/m for $t = 1, \dots, n$.

Since $x_t \in \{0, 1, \dots m-1\}$, $x_t/m \in (0,1)$. Also note that after some finite number of steps $< m$, the algorithm will repeat itself, since when a seed x_0 is set, a deterministic sequence of numbers follows. Naturally, to allow for the sequence x_t to mimic uniform and random draws m should be large. Naturally, both a and m should be chosen to be large so as to avoid repetition. Typically m should be a large prime number.

Example 1. Set $a = 123$ and $m = 10$, and let $x_0 = 7$. Then
$x_1 = 123 * 7 \mod 10 = 1$

$$x_2 = 123 * 1 \mod 10 = 3$$
$$x_3 = 123 * 3 \mod 10 = 9$$
$$x_4 = 123 * 9 \mod 10 = 7$$
$$x_5 = 123 * 7 \mod 10 = 1$$
$$\vdots$$

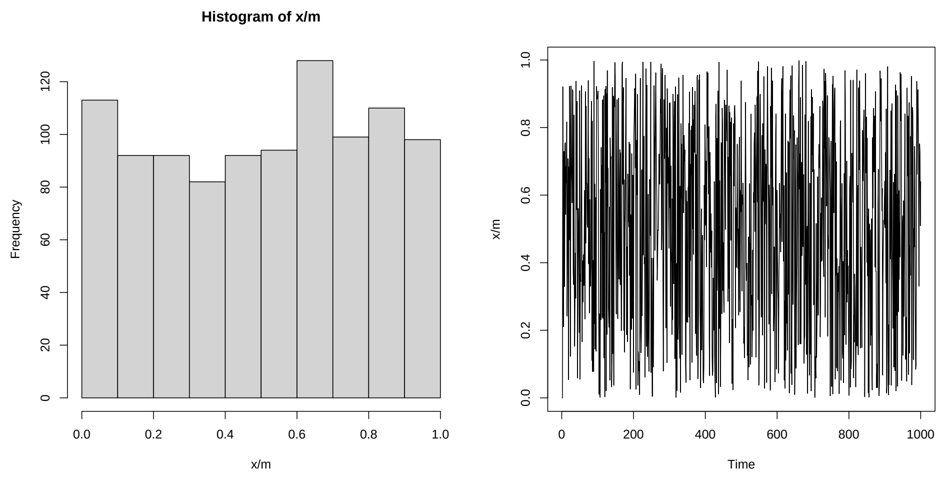
■

Thus, we see that the above choices of a, m, x_0 repeats itself. It is also recommended that a is large to ensure large jumps, and reduce "dependence" in the sequence. Based on the bits of your machine, it is recommended to set $m = 2^{31} - 1$ and $a = 7^5$. Notice that both are large.

```r
m <- 2^(31) - 1
a <- 7^5
x <- numeric(length = 1e3)
x[1] <- 7

for(i in 2:1e3)
{
  x[i] <- (a * x[i-1]) %% m
}
par(mfrow = c(1,2))
hist(x/m) # looks close to uniformly distributed
plot.ts(x/m) # look like it's jumping around too
```

The histogram shows roughly "uniform" distribution of the samples and the trace plot shows the lack of dependence between samples.



Any pseudorandom generation method should satisfy:

1. for any initial seed, the resultant sequence has the "appearance" of being IID from Uniform$[0, 1]$.
2. for any initial seed, the number of values generated before repetition begins is large
3. the values can be computed efficiently.

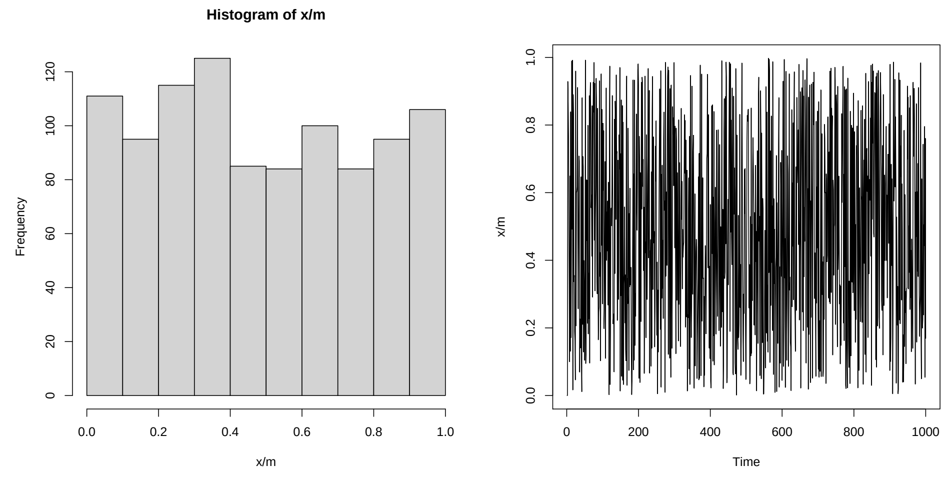
2.2 Mixed Congruential Generator

Notice that in the previous method, if we set the seed to be zero, the algorithms fails! To combat this, there is another method, the *mixed congruential generator*:

1. Set seed x_0, and positive integers a, c, m.
2. $x_t = (a\, x_{t-1} + c) \mod m$
3. Return sequence x_t/m for $t = 1, \dots, n$.

```
m <- 2^(31) - 1
a <- 7^5
c <- 2^(10) - 1
x <- numeric(length = 1e3)
x[1] <- 7

for(i in 2:1e3)
{
  x[i] <- (c + a * x[i-1]) %% m
}
par(mfrow = c(1,2))
hist(x/m) # looks close to uniformly distributed
plot.ts(x/m) # look like it's jumping around too
```

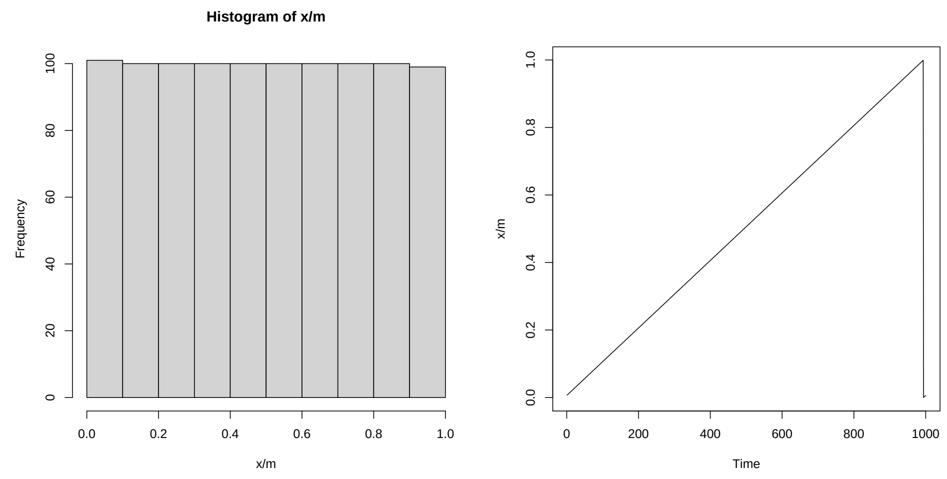


We must be cautious not to be happy with a just a histogram. A histogram shows that the empirical distribution of all samples is uniformly distributed. But we can still get a uniform looking histogram if we set $a = 1$, $m = 1e3$ and $c = 1$.

```r
m <- 1e3
a <- 1
c <- 1
x <- numeric(length = 1e3)
x[1] <- 7

for(i in 2:1e3)
{
  x[i] <- (c + a * x[i-1]) %% m
}
par(mfrow = c(1,2))
hist(x/m) # looks VERY uniformly distributed
plot.ts(x/m) # Clearly "dependent" samples
```

Although a histogram shows an almost perfect uniform distribution, the trace plot shows that the draws don't behave like they are independent.



We could also use

$$x_n = (a_1 x_{n-1} + a_2 x_{n-2} + \cdots + a_k x_{n-k} + c) \mod m\,,$$

but this requires more flops from the computer, and so is not as computationally viable.

We claim that these methods return "good" pseudosamples, in the sense of the three points stated above. There are statistical hypothesis tests, like the Kolmogorov-Smirnov test, one can do to test whether a sample is truly random: independent and identically distributed.

`runif()` in R uses the Mersenne-Twister generator by default (we will not go into this), but there are options to use other generators. After this, we will assume that `runif()` returns truly iid samples from $U(0,1)$.

2.3 Generating $U(a,b)$

Suppose we can draw from $U(a,b)$ for any $a,b \in \mathbb{R}$. But we only know how to draw from $U(0,1)$. Note that if $U \sim U(0,1)$, then for any a,b,

$$(b-a)U + a \sim U(a,b) \quad .$$

That means, we can draw $U \sim U(0,1)$ and set $X = (b-a)U + a$. Then $X \sim U(a,b)$.

```r
# Try for yourself

set.seed(1)
repeats <- 1e4
b <- 10
a <- 5
U <- runif(repeats, min = 0, max = 1)
X <- (b - a) * U + a #R is vectorized

hist(X)
```

Questions to think about

- Given a sample of pseudorandom draws from $U(0,1)$ and perfectly IID draws from $U(0,1)$, would you be able to tell the difference?
- Could we obtain uniform samples from $\mathbb{R}$?

2.4 Exercises

1. (Using R) Consider the multiplicative congruential method. For a, m positive integers

$$x_n = a x_{n-1} \mod m \,.$$

   (a) Set seed $x_0 = 5$, $m = 10^4, a = 2$. Generate $n = 10^4$ pseudorandom numbers using the above method. Does this look like a (pseudo) random sample from Uniform$[0,1]$? Maybe plot a histogram to see the empirical distribution.

   (b) Now look at only the first 10 numbers: $x_1, x_2, \ldots, x_{10}$. What is the problem here?

   (c) How can you fix the problem noted in the previous step?

3 Generating Discrete Random Variables

Suppose X is a discrete random variable having probability mass function

$$\Pr(X = x_j) = p_j \quad j = 0, 1, \ldots, \quad \sum p_j = 1 \,.$$

Examples of such random variables are: Bernoulli, Poisson, Geometric, Negative Binomial, Binomial, etc. We will learn two methods to draw samples realizations of this discrete random variable:

1. Inverse transform method
2. The acceptance-rejection technique

3.1 Inverse transform method

Let's demonstrate the inverse transform method with an example first.

Example 2 (Bernoulli distribution)**.** If $X \sim \text{Bern}(p)$, then

$$\Pr(X = 1) = p \qquad \text{and} \qquad \Pr(X = 0) = 1 - p := q \,.$$

Let $U \sim U[0, 1]$. Define

$$X = \begin{cases} 0 & \text{if } U \leq q \\ 1 & \text{if } q < U \leq 1 \end{cases} .$$

Then $X \sim \text{Bern}(p)$.

Proof. To show the result we only need to show that $\Pr(X = 1) = p$ and $\Pr(X = 0) = 1 - p$. Recall that by the cumulative distribution function of $U[0, 1]$, for any $0 < t < 1$, $\Pr(U \leq t) = t$. Using this,

$$\Pr(X = 0) = \Pr(U \leq q) = q \,,$$

and also

$$\Pr(X = 1) = \Pr(q < U \leq 1) = 1 - q = p \,.$$

■

■

Algorithm 1 Inverse transform for Bern(p)

```
1: Draw $U \sim U[0,1]$
2: if $U < q$ then $X = 0$ else $X = 1$
```

Inverse transform method: The principles used in the above example can be extended to any generic discrete distribution. For a distribution with mass function

$$\Pr(X = x_j) = p_j \qquad \text{for } j = 0, 1, \dots \quad \text{with} \quad \sum_{j=0} p_j = 1\,.$$

Let $U \sim U[0,1]$. Set X to be

$$X = \begin{cases} x_0 & \text{if } U \le p_0 \\ x_1 & \text{if } p_0 < U \le p_0 + p_1 \\ x_2 & \text{if } p_0 + p_1 < U \le p_0 + p_1 + p_2 \\ \vdots & \\ x_j & \text{if } \sum_{i=0}^{j-1} p_i < U \le \sum_{i=0}^{j} p_i \end{cases}.$$

This works because

$$\Pr(X = x_j) = \Pr\left(\sum_{i=0}^{j-1} p_i < U \le \sum_{i=0}^{j} p_i\right) = \sum_{i=0}^{j} p_i - \sum_{i=0}^{j-1} p_i = p_j\,.$$

This method is called the *Inverse transform method* since the algorithm is essentially looking at the inverse cumulative distribution function of the random variable.

Example 3 (Poisson random variables)**.** The probability mass function for the Poisson random variable is

$$\Pr(X = i) = p_i = \frac{e^{-\lambda}\lambda^i}{i!} \quad i = 0, 1, 2, \dots,$$

Algorithm 2 Inverse transform for Poisson(λ)

```
1: Draw $U \sim U[0,1]$
2: if $U \leq p_0$ then
3:     $X = 0$
4: else if $U \leq p_0 + p_1$ then
5:     $X = 1$
6:     ...
7: else if $U \leq \sum_{i=1}^{j} p_i$ then
8:     $X = j$
9:     ...
```

However, Algorithm 2 outlines a challenge in implementing this algorithm.

Q. What happens when λ is large?

A Poisson(λ) distribution with a large λ will yield p_j to be small when j is small. This implies Algorithm 2 can be quite slow here. We will therefore discuss a more computationally efficient algorithm.

We know that most likely, a realization from Poisson will be closer to λ, so it will be beneficial to start from around λ. Set $I = \lfloor \lambda \rfloor$, and check whether

$$\sum_{i=0}^{I-1} p_i < U \leq \sum_{i=0}^{I} p_i \,.$$

If it is, then return $X = I$. Else, if $U > \sum_{i=1}^{I} p_i$, then increase I, otherwise, decrease I and check again. ■

Questions to think about

- What other example can you think of where the inverse transform method could take a lot of time?
- Can you try and implement this for a Binomial random variable?

3.2 Accept-Reject for Discrete Random Variables

Although we can draw from any discrete distribution using the inverse transform method, you can imagine that for distributions on countably infinite spaces (like the Poisson distribution), the inverse transform method may be very expensive. In such situations, acceptance-rejection sampling may be more reliable.

Let $\{p_j\}$ denote the pmf of the target distribution with $\Pr(X = a_j) = p_j$ and let $\{q_j\}$ denote the pmf of another distribution with $\Pr(Y = a_j) = q_j$. Suppose you can efficiently draw from $\{q_j\}$ and you want to draw from $\{p_j\}$. Let c be a constant such that

$$\frac{p_j}{q_j} \le c < \infty \quad \text{for all } j \text{ such that } p_j > 0\,.$$

If we can find such a $\{q_j\}$ and c, then we can implement an *Acceptance-Rejection* or *Accept-Reject* sampler. The idea is to draw samples from $\{q_j\}$ and accept these samples if they seem likely to be from $\{p_j\}$.

Note: When $\{p_j\}$ has a finite set of states, c is always finite (since the maximum exists). However, when target distribution does not have a finite set of states, then c need not be finite, and accept-reject is not possible.

Algorithm 3 Acceptance-Rejection sampler to draw 1 sample from $\{p_j\}$

```
1: Draw $U \sim U[0,1]$
2: Simulate $Y = y$ with probability mass function $q_y$
3: if $U \le \frac{p_y}{cq_y}$ then
4:     Return $X = y$ and stop
5: else
6:     Goto step 1
```

Theorem 1. When c is finite, the Accept-Reject method generates a random variable with probability

$$\Pr(X = a_j) = p_j\,.$$

Further, the number of iterations needed to generate an acceptance is distributed as Geometric$(1/c)$.

Proof. First, we look at the second statement. We note that the number of iterations required to stop the algorithm is clearly geometrically distributed by the definition of the geometric distribution – the distribution of the number of Bernoulli trials needed to get one success (with support $1, 2, 3...$).

We will show that the probability of success is $1/c$. “Success” here is an acceptance. First, consider

$$\begin{aligned}\Pr(Y = a_j, \text{accepted}) &= \Pr(Y = a_j)\ \Pr(\text{Accept} \mid Y = a_j)\\ &= q_j \Pr\left(U \leq \frac{p_j}{cq_j} \mid Y = a_j\right)\\ &= q_j \frac{p_j}{cq_j} = \frac{p_j}{c}\,.\end{aligned}$$

Using this we can calculate the marginal distribution of accepting:

$$\Pr(\text{accept}) = \sum_j \Pr(Y = a_j, \text{accept}) = \sum_j \frac{p_j}{c} = \frac{1}{c}\,.$$

Thus, the second statement is proved. We will now use this to show the main statement. Note that

$$\begin{aligned}\Pr(X = a_j) &= \sum_{n=1}^{\infty} \Pr(a_j \text{ accepted on iteration } n)\\ &= \sum_{n=1}^{\infty} \Pr(\text{No acceptance until iteration } n-1)\ \Pr\left(Y = a_j, \text{accept}\right)\\ &= \underbrace{\sum_{n=1}^{\infty} \left(1 - \frac{1}{c}\right)^{n-1}}_{c}\ \frac{p_j}{c}\\ &= p_j\,.\end{aligned}$$

This completes the proof. □

Note: Since the probability of acceptance in any loop is $1/c$, the expected number of loops for one acceptance is c. The larger c is, the more expensive the algorithm.

One important thing to note is that within the support $\{a_j\}$ of $\{p_j\}$, the proposal distribution must always be positive. That is, for all a_j in the support of $\{p_j\}$, $\Pr(Y = a_j) = q_j > 0$. In other words, a proposal distribution must have support *larger* than the target distribution.

Example 4 (Sampling from Binomial using AR)**.** The binomial distribution has pmf

$$\Pr(X = x) = \binom{n}{x}(1-p)^{n-x}p^x \qquad \text{for } x = 0, 1, \dots, n\,.$$

We will use AR to simulate draws from Binomial(n, p). The first task is to choose a proposal distribution. We could use any of Poisson, negative-binomial, or geometric distributions. We cannot use Bernoulli, since the support of Bernoulli does not contain the support of Binomial.

We choose to use the geometric distribution, but we must be a little careful.

We use the version of geometric distribution that is defined as the number of failures before the first success, so that the support of the geometric distribution has 0 in it. The pmf of the geometric distribution is

$$\Pr(X = x) = (1-p)^x p \qquad x = 0, 1, \dots\,.$$

We will first find c. Note that

$$\begin{aligned}\frac{p(x)}{q(x)} &= \frac{\binom{n}{x}(1-p)^{n-x}p^x}{(1-p)^x p} \\ &= \binom{n}{x}(1-p)^{n-2x}p^{x-1}\,.\end{aligned}$$

Set

$$c = \max_{x=0,1,\dots,n} \binom{n}{x}(1-p)^{n-2x}p^{x-1}\,.$$

For $n = 10, p = 0.25$, we yield $c = 2.373\dots$.

To be safe (since we don't know all the decimal points), we may set c to be slightly larger (say $c = 2.5$) as c just needs to be an upper bound. Once c is known, the AR algorithm can be implemented simply as described. Now here, we would expect, on average, 2.5 values of Geometric random variables to be proposed until *one* acceptance.

Note, that c depends on both n and p. Particularly, if n is large, then c increases drastically. A way to understand this is then the mean of the target distribution (np) can be much larger than the mean of the proposal, $(1-p)/p$. In this case, this implies that the bulk of the mass of the target distribution is far away from the bulk of the mass of the proposal distribution. This is not ideal. We want the pmf of the proposal and target to match each other as much as possible, so that c is close to 1. This suggests, that we may **not** want to choose the same p in the proposal distribution!

A possible fix, is to consider a Geometric(p^*) proposal where p^* is such that

$$np = \frac{1-p^*}{p^*} \Rightarrow p^* = \frac{1}{np+1}$$

In this case, we have

$$\frac{p(x)}{q(x)} = \frac{\binom{n}{x}(1-p)^{n-x}p^x}{(1-p^*)^x p^*}$$

the maximum over $\{0, 1, \ldots, n\}$ can be determined on the computer. For $n = 100$ and $p = .25$, the old bound is 1028.497 and the new one is 6.0455, which is much more efficient! ■

Example 5. (Geometric Random Variable) We consider the Geometric random variable with pmf (trails until x failures)

$$\Pr(X = x) = (1-p)^x p \qquad x = 0, 1, 2, \ldots$$

We cannot use Binomial as a proposal, but we can use Poisson. Let us consider the Poisson(λ) proposal. The Poisson random variable has pmf

$$\Pr(X = x) = \frac{e^{-\lambda}\lambda^x}{x!} \qquad x = 0, 1, 2, \ldots .$$

First step is to find c, if it exists

$$\begin{aligned} \frac{p(x)}{q(x)} &= \frac{(1-p)^x p}{\frac{e^{-\lambda}\lambda^x}{x!}} \\ &= \frac{p}{e^{-\lambda}} \left(\frac{1-p}{\lambda}\right)^x x! \,. \end{aligned}$$

For small values of λ ($< 1-p$), the above clearly diverges as x increases, thus the maximum doesn't exist. This is true for large values of λ as well. To see, this (intuitively), consider the Stirling's approximation of the factorial:

$$\log(x!) \approx x\log(x) - x \Rightarrow x! \approx e^{x\log x - x} \,.$$

Using this:

$$\begin{aligned}
\frac{p(x)}{q(x)} &= \frac{p}{e^{-\lambda}}\left(\frac{1-p}{\lambda}\right)^x x! \\
&= \frac{p}{e^{-\lambda}}\left(\frac{1-p}{\lambda}\right)^x e^{x\log x - x} \\
&= \frac{p}{e^{-\lambda}}\left(\frac{(1-p)e^{\log(x)}}{e\lambda}\right)^x
\end{aligned}$$

Thus, no matter how large λ is, eventually as x increases $e^{\log(x)}$ will be larger than λ and the ratio will diverge. Thus, this proposal does not allow an AR for the Geomtric distribution. ■

Question to think about

- Why is c always greater than 1?
- What happens when c is large or small?

3.3 The Composition Method

We have now learned two algorithms for sampling from a discrete distribution: the inverse transform method and the accept-reject algorithm. The inverse transform method can be used for *any* distribution and the accept-reject can be efficient if used properly.

For certain special distributions, it is easier to use a *composition method* for sampling.

Suppose we have an efficient way of simulating random variables from two pmfs $\{p_j^{(1)}\}$ and $\{p_j^{(2)}\}$, and we want to simulate from

$$\Pr(X = j) = \alpha p_j^{(1)} + (1-\alpha)p_j^{(2)} \quad j \geq 0 \;\text{ where } 0 < \alpha < 1\,.$$

First you should note that the above *composition pmf* is a valid pmf since $\sum_j \Pr(X = j) = 1$. How would we sample in such a situation?

Let $X_1 \sim P^{(1)}$ and $X_2 \sim P^{(2)}$. Set

$$X = \begin{cases} X_1 & \text{with probability } \alpha \\ X_2 & \text{with probability } 1-\alpha \end{cases}.$$

Algorithm 4 Composition method

```
1: Draw $U \sim U[0,1]$
2: if $U \leq \alpha$ then simulate $X_1 \sim P^{(1)}$ else simulate $X_2$ and stop
```

Proof. Consider

$$
\begin{aligned}
&\Pr(X = j) \\
&= \Pr(X = j, U \leq \alpha) + \Pr(X = j, \alpha < U \leq 1) \qquad \text{(by law of total probability)} \\
&= \Pr(X = j \mid U \leq \alpha) \Pr(U \leq \alpha) + \Pr(X = j \mid \alpha < U \leq 1) \Pr(\alpha < U \leq 1) \\
&= \Pr(X_1 = j) \Pr(U \leq \alpha) + \Pr(X_2 = j) \Pr(\alpha < U \leq 1) \qquad \text{(by independence of } U \text{ and } X_1, X_2\text{)} \\
&= \alpha p_j^{(1)} + (1-\alpha) p_j^{(2)} .
\end{aligned}
$$

□

We can set this up more generally for k different distributions. In general, $F_i, i = 1, \ldots, k$ are distribution functions, and α_i are such that $0 < \alpha_i < 1$ for all i and $\sum_i \alpha_i = 1$. The composition (or mixture) distribution is

$$F(x) = \sum_{i=1}^{k} \alpha_i F_i(x) .$$

Let $X_i \sim F_i$. To simulate from the composition F, set

$$X = \begin{cases} X_1 & \text{with probability } \alpha_1 \\ X_2 & \text{with probability } \alpha_2 \\ \vdots & \\ X_k & \text{with probability } \alpha_k \end{cases} .$$

Example 6 (Zero inflated Poisson distribution)**.** A Poisson(λ) distribution usually has a small mass at 0. But sometimes, we need a counting distribution with large mass at 0. For example, consider the random variable X being the number of COVID-19 patients tested positive every hour. Many hours of the day this number may be 0, and then this number can be quite high for some hours.

In such a case, we may use the *zero inflated Poisson distribution* (ZIP). Recall that if

$X \sim \text{Poisson}(\lambda)$

$$\Pr(X = k) = e^{-\lambda}\frac{\lambda^k}{k!} \quad k = 0, 1, \dots .$$

If $X \sim \text{ZIP}(\delta, \lambda)$ for $\delta > 0$

$$\Pr(X = k) = \begin{cases} \delta + (1-\delta)e^{-\lambda} & \text{if } k = 0 \\ (1-\delta)e^{-\lambda}\dfrac{\lambda^k}{k!} & \text{if } k = \{1, 2, \dots\} \end{cases} .$$

Note that the mean of a ZIP is $(1-\delta)\lambda < \lambda$ since more mass is given at 0. We will use the composition method to sample from the ZIP distribution. To sample from a ZIP, first $p_j^{(1)}$ be defined as

$$\Pr(X_1 = 0) = 1 \quad \text{and} \quad \Pr(X_1 \neq 0) = 0\,,$$

and let $X_2 \sim \text{Poisson}(\lambda)$. Define the pmf:

$$\Pr(X = k) = \delta p_k^{(1)} + (1-\delta)p_k^{(2)}\,.$$

Then $X \sim \text{ZIP}(\delta, \lambda)$. To see this, plug in $k = 0$ and $k = 1, 2, \dots$ above:

Algorithm 5 Zero inflated Poisson distribution

```
1: Draw $U \sim U[0, 1]$
2: if $U \leq \delta$ then $X = 0$ else simulate $X \sim \text{Poisson}(\lambda)$
```

■

Other composition or mixture distributions are also possible. Think about Zero-inflated Binomial, Zero-inflated Geometric, 2-inflated Poisson, etc.

3.4 Exercises

1. Show that if $U \sim U(0, 1)$, then for any a, b,

$$(b - a) * U + a \sim U(a, b)$$

2. Use the inverse transform method to sample from a geometric distribution, where

for $0 < p < 1$ and $q = 1 - p$,

$$\Pr(X = i) = pq^{i-1}, \quad i \geq 1, \quad \text{where } q = 1 - p\,.$$

3. We want to draw a sample from the random vector $(X, Y)^\top$, that follows the distribution with joint probability mass function

$$P(X = i, Y = j) = \theta_{i,j} \qquad \text{where } \; i, j \in \{0, 1\}\,.$$

Here $\sum_{i,j} \theta_{i,j} = 1$. Write an inverse-transform algorithm to draw realizations of $(X, Y)^\top$.

4. List as many appropriate proposal distributions as you can think of for the following target distributions:
   - Binomial
   - Bernoulli
   - Geometric
   - Negative Binomial
   - Poisson

5. (Using R) Draw 10,000 draws from a Binomial(20, .75) distribution using an accept-reject sampler.

6. In an accept-reject algorithm, we need to find c such that

$$\frac{p_j}{q_j} \leq c \quad \text{forall } j \text{ for which } p_i > 0\,.$$

And, the probability of accepting in any iteration is $1/c$. Why is c guaranteed to be more than 1?

7. Simulate from a Negative Binomial(n, p) using the inverse transform and accept-reject methods. Implement in R with $n = 10$ successes and $p = .30$.

8. Simulate from the following "truncated Poisson distribution" with pmf:

$$\Pr(X = i) = \frac{e^{-\lambda}\lambda^i / i!}{\sum_{j=0}^{m} e^{-\lambda}\lambda^j / j!} \quad i = 0, 1, 2, \ldots, m\,.$$

Implement in R with $m = 30$ and $\lambda = 20$.

9. Suppose we want to obtain samples from a discrete distribution with pmf $\{p_i\}$. We use accept-reject with proposal distribution with pmf $\{q_i\}$, such that for some $\alpha \in \mathbb{R}$:
$$\frac{p_i}{q_i} \propto i^{\alpha} \qquad i = 1, 2, \dots, ,$$
For what values of α would this AR algorithm work?

10. Suppose we want to obtain samples from a discrete distribution with pmf $\{p_i\}$. Two possible proposal distributions are $\{q_i^{(1)}\}$ and $\{q_2^{(2)}\}$, yielding AR bounds c_1 and c_2 such that $c_1 > c_2$. Which proposal distribution is better?

11. Implement a an algorithm to sample from a Zero Inflated Binomial distribution. Can you think of an application of such a distribution?

4 Generating continuous random variables

Similar to generating discrete random variables, there are various methods for generating continuous random variables. We will discuss three main methods:

1. Inverse transform
2. The accept-reject method
3. Ratio of uniforms

We will also discuss a few special samplers.

4.1 Inverse transform

The principles of the inverse transform method for discrete distributions, apply similarly to continuous random variables. Consider a random variable X with probability density function $f(x)$ so that $f(x) \geq 0$, $\int_{-\infty}^{\infty} f(x) = 1$ with distribution function

$$F(x) = \int_{-\infty}^{x} f(x)\, dx\,.$$

The following theorem will be the foundation for the inverse transform method.

Theorem 2. Let $U \sim U[0,1]$. For any continuous distribution F, a random variable $X = F^{-1}(U)$ has distribution F.

Proof. Let F_X be the distribution function of $X = F^{-1}(U)$. We need to show that $F_X = F$. Note that for any $x \in \mathbb{R}$,

$$\begin{aligned}
F_X(x) &= \Pr(X \leq x) \\
&= \Pr(F^{-1}(U) \leq x) \\
&= \Pr(F(F^{-1}(U)) \leq F(x)) \qquad \text{(Since } F \text{ is non-decreasing)} \\
&= \Pr(U \leq F(x)) \\
&= F(x)\,.
\end{aligned}$$

□

The above theorem then implies that if we can invert the CDF function, then we can obtain random draws from that random variable.

Example 7. Exponential(1)**:** For the Exponential(1) distribution, the cdf is $F(x) = 1 - e^{-x}$. Thus,

$$F^{-1}(u) = -\log(1-u)\,.$$

To generate $X \sim \text{Exp}(1)$ we can thus use the following algorithm:

Algorithm 6 Exponential(1) Inverse transform

```
1: Generate $U \sim U[0,1]$
2: Set $X = -\log(1-U) \sim \text{Exp}(1)$
```

Similarly, we can draw from an Exponential(λ) distribution. ■

Example 8. Cauchy distribution: Cauchy distribution has pdf

$$f(x) = \frac{1}{\pi}\frac{1}{(1+x^2)}\,,$$

and

$$u = F(x) = \int_{-\infty}^{x} f(y)dy = \frac{1}{\pi}\arctan(x) + \frac{1}{2}\,.$$

So, $F^{-1}(u) = \tan(\pi(u - .5))$.

Algorithm 7 Cauchy distribution

```
1: Generate $U \sim U[0,1]$
2: Set $X = \tan(\pi(U - .5) \sim \text{Cauchy}$
```

■

Example 9. Gamma distribution: The CDF of a Gamma(n, λ) distribution is

$$F(x) = \int_0^x \frac{\lambda e^{-\lambda y}(\lambda y)^{n-1}}{\Gamma(n)}dy\,.$$

Here, we don't know the CDF in closed form and thus cannot analytically find the inverse. This is an example where the inverse transform method cannot work in practice (even though it works theoretically). Thus, unlike the discrete case, this genuinely motivates the need for another method to sample from a distribution.

■

Questions to think about

- The CDF $F(x)$ is a deterministic function, so how is $F^{-1}(U)$ a random quantity?
- Can we use the inverse transform method to generate sample from a normal distribution?

4.2 Accept-reject method

Suppose we cannot generate from distribution F with pdf $f(x)$, like the Gamma distribution example in inverse transform. We can use accept-reject in a similar way as the discrete case. That is, we choose an appropriate *proposal distribution* with density $g(x)$, and accept or reject it based on certain probabilities.

Let the support of F be $\mathcal{X}$ and choose a proposal distribution G with density $g(x)$ *whose support is larger or the same as the support of F.* That is, if $\mathcal{Y}$ is the support of G then, $\mathcal{X} \subseteq \mathcal{Y}$. If we can fine c such that

$$\sup_{x \in \mathcal{X}} \frac{f(x)}{g(x)} \leq c\,,$$

then an accept-reject sampler can be implemented.

Algorithm 8 Accept-reject for continuous random variables

```
1: Draw $U \sim U[0,1]$
2: Draw proposal $Y \sim G$, independently
3: if $U \leq \frac{f(Y)}{c\,g(Y)}$ then
4:     Return $X = Y$
5: else
6:     Go to Step 1.
```

Theorem 3. Algorithm 8 returns $X \sim F$. Further, the number of loops the AR algorithm takes to return X is distributed Geometric($1/c$).

Proof. Consider any set B in $\mathcal{X}$. We will show that

$$\Pr(X \in B) = F(B)\,.$$

First, we consider the probability of acceptance:

$$\begin{aligned}
\Pr(\text{accept}) &= \Pr\left(U \leq \frac{f(Y)}{cg(Y)}\right) \\
&= \mathrm{E}\left[I\left(U \leq \frac{f(Y)}{cg(Y)}\right)\right] \\
\\
&= \mathrm{E}\left[\mathrm{E}\left[I\left(U \leq \frac{f(Y)}{cg(Y)}\right) \mid Y\right]\right] && \text{using iterated expectations} \\
&= \mathrm{E}\left[\Pr\left(U \leq \frac{f(Y)}{cg(Y)} \mid Y\right)\right] \\
&= \mathrm{E}\left[\frac{f(Y)}{cg(Y)}\right] \\
&= \int_{\mathcal{Y}} \frac{f(y)}{cg(y)} g(y)dy \\
&= \frac{1}{c}\int_{\mathcal{Y}} f(y)dy \\
&= \frac{1}{c}\int_{\mathcal{X}} f(y)dy + \frac{1}{c}\int_{\mathcal{Y}/\mathcal{X}} f(y)dy && \text{since } \mathcal{Y} \subseteq \mathcal{X} \\
&= \frac{1}{c}\,.
\end{aligned}$$

Now that we have this established, consider

$$\begin{aligned}
\Pr(X \in B) &= \Pr(Y \in B \mid \text{accept}) \\
&= \frac{\Pr\left(Y \in B, U < \frac{f(Y)}{cg(Y)}\right)}{\Pr(\text{accept})} \\
&= c \cdot \mathrm{E}\left[\mathrm{E}\left[I\left(Y \in B, U < \frac{f(Y)}{cg(Y)}\right) \mid Y\right]\right] \\
&= c \cdot \mathrm{E}\left[I\left(Y \in B\right)\mathrm{E}\left[I\left(U < \frac{f(Y)}{cg(Y)}\right) \mid Y\right]\right] \\
&= c \cdot \mathrm{E}\left[I\left(Y \in B\right)\frac{f(Y)}{cg(Y)}\right] \\
&= c \cdot \int_B \frac{f(y)}{cg(y)} g(y)dy \\
&= \int_B f(y)dy \\
&= F(B)\,.
\end{aligned}$$

□

From the proof, we know that Pr(*accept*) $= 1/c$, and so, just like the discrete example, the number of attempts it takes to generate an acceptance is distributed Geometric($1/c$). Thus

$$\textit{Expected number of loops for an acceptance is} = c\,.$$

Accept-reject method: intuition

At a proposed value y:

- if $f(y)$ is large but $g(y)$ is small means this value will not be proposed often and is a good value to accept for f, so higher probability of accepting it.
- if $f(y)$ is small but $g(y)$ is large, then this value will be proposed often but is unlikely for f, so accept this value less often.

We can choose any g we want as long its support is larger than other the support of f, and the resulting c is finite. However, some gs will be better than other gs, based on the expected number of iterations, c.

Example 10. Beta distribution: Consider the beta distribution Beta$(4, 3)$, where

$$f(x) = \frac{\Gamma(7)}{\Gamma(4)\Gamma(3)} x^{4-1}(1-x)^{3-1} \quad 0 < x < 1; \quad .$$

Consider a uniform proposal distribution. So that $G = U(0, 1)$ and

$$g(x) = 1 \qquad \text{for } x \in (0, 1)\,.$$

Note that, $\mathcal{X} = \mathcal{Y}$ in this case. For this choice of g,

$$\sup_{x\in(0,1)} \frac{f(x)}{g(x)} = \sup_{x\in(0,1)} f(x)$$

We can show that maximum of $f(x)$ occurs at $x = 3/5$ and

$$\sup_{x\in(0,1)} \frac{f(x)}{g(x)} = \sup_{x\in(0,1)} f(x) = 60\left(\frac{3}{5}\right)^3\left(\frac{2}{5}\right)^2 = 2.0736 = c\,.$$

Algorithm 9 Accept-reject for Beta$(4,3)$

```
1: Draw $U \sim U[0,1]$
2: Draw proposal $Y \sim U(0,1)$
3: if $U \leq \dfrac{f(Y)}{c\,g(Y)}$ then
4:     Return $X = Y$
5: else
6:     Go to Step 1.
```

■

In order to choose a good proposal distribution (that yields a finite c), it is important to choose a g so that it has <u>"fatter tails"</u> than f. This ensures that as $x \to \infty$ or $x \to \infty$, g dominates f, so that $c \to 0$ in the extremes, rather than blow up.

Example 11. Normal distribution

The target density function is

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\,.$$

We know that the t-distribution has the right support and fatter tails and the "fattest" t distribution is with degrees of freedom 1, which is Cauchy. The pdf of a Cauchy distribution is

$$g(x) = \frac{1}{\pi}\frac{1}{1+x^2}\,.$$

(We know we can sample from Cauchy using inverse transform, so that is easy.) We will need to find the supremum of the ratio of the densities. Consider

$$\frac{f(x)}{g(x)} = \frac{\pi}{\sqrt{2\pi}}(1+x^2)e^{-x^2/2}\,.$$

When $x \to \infty, -\infty$, $e^{-x^2/2}$ decreases more rapidly than x^2 increases, the ratio tends to zero. This can be shown more formally using L'Hopital's rule.

Taking a derivative of the above ratio and setting it to 0 (and checking the second derivative condition), yields that the supremum above occurs at $x = -1, 1$, so

$$\sup_{x\in\mathbb{R}} \frac{f(x)}{g(x)} = \frac{f(1)}{g(1)} = \sqrt{2\pi}e^{-1/2} \approx 1.746 \Rightarrow c = 1.80\,.$$

The actual algorithm can now be implemented similarly as before.

Note: Consider if the target distribution was Cauchy, and the proposal was $N(0,1)$? The ratio would clearly diverge as $x \to -\infty, \infty$, and thus an accept-reject sampler would not be possible. ■

Example 12. Sampling from a uniform circle Consider a unit circle centered at $(0,0)$:

$$x^2 + y^2 < 1 \qquad -1 < x, y < 1\,.$$

We are interested in sampling uniformly from within this circle. Since the area of the circle is π, the target density is

$$f(x,y) = \frac{1}{\pi} I(x^2 + y^2 < 1)\,.$$

Consider the uniform distribution on the square as a proposal distribution

$$g(x,y) = \frac{1}{4} I(-1 < x < 1) I(-1 < y < 1)\,.$$

First, we will find c. For x, y such that $x^2 + y^2 < 1$,

$$\frac{f(x,y)}{g(x,y)} = \frac{4}{\pi} I(x^2 + y^2 < 1) \leq \frac{4}{\pi} := c\,.$$

Next, note that

$$\frac{f(x,y)}{cg(x,y)} = \frac{4}{\pi} I(x^2 + y^2 < 1) \frac{\pi}{4} = I(x^2 + y^2 < 1)\,.$$

So for any (x, y) drawn from within the square, the ratio will be either 1 or 0, thus, no need to draw a uniform at all!

Note: How do we draw uniformly from within the box? Note that

$$g(x,y) = \left[\frac{1}{2} I(-1 < x < 1)\right] \left[\frac{1}{2} I(-1 < y < 1)\right] = g_1(x) \cdot g_1(y)\,,$$

where g_1 is a density of a $U(-1,1)$ random variable. Thus, with two independent draws $U_1, U_2 \overset{\text{iid}}{\sim} U(-1,1)$, we have $U_1 \times U_2$ being a draw from the uniform box.

Algorithm 10 Accept-reject for Uniform distribution on a circle

```
1: Draw proposal $(U_1, U_2) \sim U(-1,1) \times U(-1,1)$
2: if $U_1^2 + U_2^2 \leq 1$ then
3:     Return $(X, Y) = (U_1, U_2)$
4: else
5:     Go to Step 1.
```

■

Questions to think about

- In A-R, do we want c to be large or small?
- Why is c guaranteed to be more than 1?
- How can you decide whether one proposal distribution is better than another proposal distribution?
- Try implementing the circle/square example in 3 dimension, 4 dimensions, and a general p dimensions. What happens to c?
- Can a similar A-R algorithm be implemented for Beta(m,n) for all $m, n \in \mathbb{Z}$?

4.2.1 Choosing a proposal

Sometimes it is difficult to find a good proposal or even one that works! That is, for a target density $f(x)$ it can sometimes be challenging to find a proposal density $g(x)$ such that

$$\sup_x \frac{f(x)}{g(x)} < \infty$$

Here are certain examples of when it may be difficult / impossible to implement accept-reject.

Example 13 (Beta)**.** Consider a Beta(m,n)

$$f(x) = \frac{\Gamma(m+n)}{\Gamma(m)\Gamma(n)} x^{m-1}(1-x)^{n-1}$$

Depending on m and n, the Beta distribution can behave quite differently. Particularly, note that when both $m, n < 1$ the Beta density function is unbounded!

When $m, n < 1$, if we use a uniform proposal distribution

$$\sup_{x \in (0,1)} \frac{f(x)}{g(x)} = \sup_{x \in (0,1)} \frac{\Gamma(m+n)}{\Gamma(m)\Gamma(n)} x^{m-1}(1-x)^{n-1} = \infty \,.$$

So a Uniform distribution will not work. In fact, any proposal distribution with a bounded density will not work. So this is an example of a distribution where it is difficult to find a good proposal distribution.

However, when say $n \geq 1$, then

$$\begin{aligned} f(x) &= \frac{\Gamma(m+n)}{\Gamma(m)\Gamma(n)} x^{m-1}(1-x)^{n-1} \\ &\leq \frac{\Gamma(m+n)}{\Gamma(m)\Gamma(n)} x^{m-1} \,. \end{aligned}$$

If we look at the upper bound, the function x^{m-1} on $x \in (0,1)$ can define a valid distribution if normalized. So, consider $g(x) = mx^{m-1}$, which is a proper density on $0 \leq x \leq 1$, and

$$\frac{f(x)}{g(x)} \leq m \frac{\Gamma(m+n)}{\Gamma(m)\Gamma(n)} := c$$

This Accept-Reject sampler can be implemented easily. Similarly if $m \geq 1$. Thus, an AR sampler is easier to implement here if one of m or n is more than (or equal to) 1. ■

Example 14 (Accept-Reject for Cauchy target)**.** Consider the target density

$$f(x) = \frac{1}{\pi} \frac{1}{1+x^2} \quad x \in \mathbb{R} \,,$$

The Cauchy distribution is known to have "fat tails" so that as $x \to \pm\infty$, the density function reduces to zero slowly. This means that it is very challenging to find $g(x)$ that "dominates" the density in the tails.

For example, let the proposal be $N(0,1)$. As discussed before, we will see the ratio of the densities be

$$\frac{f(x)}{g(x)} = \frac{2\pi}{\pi}\frac{e^{x^2/2}}{1+x^2} \to \infty \text{ as } x \to \pm\infty!$$

In fact, as far we know, there are no possible standard accept-reject algorithms possible here. ■

4.2.2 Choosing parameters for a fixed proposal family

If you have chosen a family of proposal distributions that you know gives a finite c, it may be unclear what the best parameters for that proposal distribution is. That is, if the target $f(x)$ and the proposal density is $g(x \mid \theta)$ (where θ is a parameter you can change, to change the behaviour of the proposal, then you want to find a value of the parameter θ so that the resulting proposal is the "best".

Notice that the upper bound will be a function of θ, so that

$$\sup_x \frac{f(x)}{g(x|\theta)} \le c(\theta)\,.$$

The value $c(\theta)$ is the expected number of loops for the accept-reject algorithm. Since we want this to be small, the best proposal density within this family would be the one that minimizes $c(\theta)$, so set

$$\theta^* := \arg\min_\theta c(\theta)\,.$$

Example 15 (Gamma distribution)**.** Consider the target distribution Gamma(α, β)

$$f(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}\,.$$

Further, suppose we want to use an Exp(λ) proposal. Then

$$g(x|\lambda) = \lambda e^{-\lambda x}\,.$$

We can now find $c(\lambda)$,

$$\frac{f(x)}{g(x)} = \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{x^{\alpha-1}e^{-\beta x}}{\lambda e^{-\lambda x}}$$

$$= \frac{\beta^\alpha}{\lambda\Gamma(\alpha)} x^{\alpha-1} e^{-x(\beta-\lambda)} .$$

First note that no matter what λ is, if $0 < \alpha < 1$, then $f(x)/g(x) \to \infty$ as $x \to 0$. So accept-reject with this proposal won't work!

However, when $\alpha \geq 1$, then $x^{\alpha-1}$ increases, so we want to choose λ such that $e^{-x(\beta-\lambda)}$ decreases (since exponential decay is more powerful than polynomial increase) (of course, you should show this more mathematically). Thus we want $\beta > \lambda$!

Thus, we restriction attention to $\alpha \geq 1$, $\lambda < \beta$,

$$c(\lambda) = \sup_x \frac{f(x)}{g(x)} = \sup_{x>0} \frac{\beta^\alpha}{\lambda\Gamma(\alpha)} x^{\alpha-1} e^{-x(\beta-\lambda)}$$

which you can show, occurs at

$$x = \frac{\alpha - 1}{\beta - \lambda} ,$$

for which

$$c(\lambda) = \frac{\beta^\alpha}{\lambda\Gamma(\alpha)} \left(\frac{\alpha - 1}{\beta - \lambda} \right)^{\alpha-1} e^{1-\alpha} ,$$

which is minimized for

$$\lambda = \beta/\alpha.$$

Thus, the optimal exponential proposal for the Gamma$(\alpha, \beta), \alpha > 1$ is Exp(β/α). ■

Questions to think about

- How would you implement accept-reject for Gamma(α, β) for $0 < \alpha < 1$?

4.3 The Box-Muller transformation for $N(0, 1)$.

A classical method to generate samples from $N(0, 1)$ is the Box-Muller transformation method. Here, we will draw random variables (R^2, Θ) from a certain distribution in the polar coordinate system, and then use a transformation h, so that $h(R^2, \Theta) \sim N(0, 1)$. First, we will need some theory for this.

Let X and $Y \overset{\text{iid}}{\sim} N(0, 1)$. The joint density of (X, Y) is

$$f(x, y) = \frac{1}{2\pi} e^{-x^2/2} e^{-y^2/2} \quad x \in \mathbb{R}, y \in \mathbb{R}.$$

Let (R^2, Θ) denote the polar coordinates of (X, Y) so that, $X = R \cos \Theta$ and $Y =$

$R\sin\Theta$; here the support of R is $(0,\infty)$ and the support of Θ is $(0, 2\pi)$. Then,

$$R^2 = X^2 + Y^2 \qquad \tan\Theta = \frac{Y}{X}\,.$$

Notationally, we denote a realization from (R^2, Θ) as (d, θ) and find the joint density of $f(d,\theta)$. Thus, let $d = x^2 + y^2$ and $\theta = \tan^{-1}(y/x)$. We know that the density for (d, θ) can be found by

$$f(d,\theta) = |J| f(x,y) \qquad \text{where } J = \begin{vmatrix} \frac{\partial x}{\partial d} & \frac{\partial y}{\partial d} \\ \frac{\partial x}{\partial \theta} & \frac{\partial y}{\partial \theta} \end{vmatrix}$$

Solving for J,

$$J = \begin{vmatrix} \frac{\partial \sqrt{d}\cos\theta}{\partial d} & \frac{\partial \sqrt{d}\sin\theta}{\partial d} \\ \frac{\partial \sqrt{d}\cos\theta}{\partial \theta} & \frac{\partial \sqrt{d}\sin\theta}{\partial \theta} \end{vmatrix} = \begin{vmatrix} \frac{1}{2}\frac{\cos\theta}{\sqrt{d}} & \frac{1}{2}\frac{\sin\theta}{\sqrt{d}} \\ -\sqrt{d}\sin\theta & \sqrt{d}\cos\theta \end{vmatrix} = \frac{1}{2}\,.$$

Since $d = x^2 + y^2$, the joint density of (R^2, Θ) is $f(d, \theta)$ with

$$\begin{aligned} f(d,\theta) &= \frac{1}{2}\frac{1}{2\pi}e^{-d/2} \qquad 0 < d < \infty, 0 < \theta < 2\pi \\ &= \underbrace{\frac{1}{2\pi} I(0 < \theta < 2\pi)}_{U(0,2\pi)} \; \underbrace{\frac{1}{2}e^{-d/2} I(0 < d < \infty)}_{\text{Exp}(2)} \end{aligned}$$

This is a separable density, so R^2 and Θ are independent, and $\Theta \sim U[0, 2\pi]$ and $R^2 \sim \text{Exp}(2)$.

To generate from Exp(2), we can use an inverse transform method. If $U \sim U(0,1)$, then by the inverse transform method, $-2\log U \sim \text{Exp}(2)$ (verify for yourself). To generate from $U(0, 2\pi)$, we know if $U \sim U(0,1)$, then $2\pi U \sim U(0,2\pi)$. The Box-Muller algorithm then is given in Algorithm 11 which produces X and Y from $N(0,1)$ indendently.

Algorithm 11 Box-Muller algorithm for $N(0,1)$

```
1: Generate $U_1$ and $U_2$ from $U[0,1]$ independently
2: Set $R^2 = -2\log U_1$ and $\Theta = 2\pi U_2$
3: Set $X = R\cos(\Theta) = \sqrt{-2\log U_1}\cos(2\pi U_2)$
4: and $Y = R\sin(\Theta) = \sqrt{-2\log U_1}\sin(2\pi U_2)$ .
```

4.4 Ratio-of-Uniforms

Ratio-of-uniforms is a powerful, however not so popular method to generate samples for a continuous random variables. When it works, it can work really well. The method is based critically on the following theorem.

Theorem 4. Let $f(x)$ be a target density with distribution function F. Define the set

$$C = \left\{ (u, v) : 0 \leq u \leq \sqrt{f\left(\frac{v}{u}\right)} \right\} .$$

If C bounded, let (U, V) be uniformly distributed over the set C; then $V/U \sim F$.

Proof. We will show that the density of $Z = V/U$ is $f(z)$. Note that by definition, the joint density of (U, V) is

$$f_{(U,V)}(u, v) = \frac{1}{\int\int_C du\, dv} I\left\{(u, v) \in C\right\} .$$

Consider transformation $(U, V) \mapsto (U, Z)$ with $Z = V/U$. Then $U = U$ and $V = UZ$. It's easy to see that the Jacobian for this transformation is U. So

$$f_{(U,Z)}(u, z) = \frac{u}{\int\int_C du\, dv} I\left\{0 \leq u \leq f^{1/2}(z)\right\} .$$

Now that we have the joint distribution of (U, Z), all we need to show is that the marginal distribution of Z is F. Finding the marginal distribution of $Z = V/U$, we integrate out U,

$$\begin{aligned} f_Z(z) &= \int \frac{u}{\int\int_C du\, dv} I\left\{0 \leq u \leq f^{1/2}(z)\right\} du \\ &= \frac{1}{\int\int_C du\, dv} \int_0^{f^{1/2}(z)} u du \\ &= \frac{f(z)}{2\int\int_C du\, dv} . \end{aligned}$$

Since $f_Z(z)$ and $f(z)$ are both densities, this implies that

$$1 = \int f_Z(z) dz = \frac{\int f(z) dz}{2\int\int_C du\, dv} = \frac{1}{2\int\int_C du\, dv} \Rightarrow \int\int_C du\, dv = \frac{1}{2}$$

This implies $f_Z(z) = f(z)$. Thus, $Z = V/U$ has the desired distribution. $\square$

So if we can draw $(U, V) \sim \text{Unif}(C)$, then $V/U \sim F$. But C looks quite complicated, so how do we uniformly draw from C?

Think back to the AR technique used to draw uniformly from a circle! If C is a bounded set, then if we enclose C in a rectangle, we can use accept-reject to draw uniform draws from C! So, the task is to find $[0, a] \times [b, c]$ such that

$$0 \le u \le a \quad b \le v \le c \quad \text{ for all } (u, v) \in C\,.$$

We just need to find any such a, b, c.

First, note that if $\sup_x f^{1/2}(x)$ exists, then

$$0 \le u \le f^{1/2}\left(\frac{v}{u}\right) \le \sup_x f^{1/2}(x) =: a\,.$$

Note now that inside C, if $x = v/u \Rightarrow v/x = u \le f^{1/2}(x)$. This implies that

$$\frac{v}{x} \le f^{1/2}(x)\,.$$

Now for:

$$\begin{aligned} x \ge 0: &\quad v \le x f^{1/2}(x) \le \sup_{x \ge 0} x f^{1/2}(x) =: c \\ x \le 0: &\quad v \ge x f^{1/2}(x) \ge \inf_{x \le 0} x f^{1/2}(x) =: b\,. \end{aligned}$$

Note that if $\sqrt{f(x)}$ or $x^2 f(x)$ are unbounded, then C is unbounded, and the method cannot work. Now that we have found the rectangle: $[0, a] \times [b, c]$, we can propose from the rectangle, check if the proposed value is in the region C; if it is, we accept it and return V/U. This leads to the following algorithm:

Algorithm 12 Ratio-of-Uniforms

```
1: Generate $(U, V) \sim U[0, a] \times U[b, c]$
2: If $U \le \sqrt{f(V/U)}$, then set $X = V/U$.
3: Else go to 1.
```

Steps 1 and 2 in Algorithm 12 are implementing an Accept-Reject to sample uniformly from C. To understand how effective this algorithm will be, we can calculate the probability of acceptance for the AR. First, note that

$$\sup_{(u,v)\in C} \frac{f(u,v)}{g(u,v)} = \sup_{(u,v)\in C} \frac{\frac{I((u,v)\in C)}{\int_C dudv}}{\frac{1}{a*(c-b)}} = 2a(c-b)$$

Thus,

$$\Pr(\text{Accepting for AR in RoU}) = \frac{1}{2a(c-b)}\,.$$

So if a is large and/or $(c-b)$ is large, the probability is small, and thus the algorithm will take a large number of loops to yield one acceptance.

Example 16 (Exponential(1))**.**

$$f(x) = e^{-x} \quad x \geq 0$$

Here,

$$C = \left\{(u,v) : 0 \leq u \leq e^{-v/2u}\right\}.$$

Since $e^{-x/2}$ is a decreasing function, $a = \sup_x e^{-x/2} = 1$. Additionally,

$$b = \inf_{x\leq 0} xe^{-x/2} = 0 \quad \text{(since support is } x \geq 0)$$

and

$$c = \sup_{x\geq 0} xe^{-x/2} \Rightarrow c = 2e^{-1} \quad \text{(show for yourself)}\ .$$

So we sample from $U[0,1] \times [0, 2/e]$ and then implement accept-reject. ■

Example 17 (Normal(θ, σ^2))**.** The target density is:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\theta)^2/2\sigma^2}\,.$$

The set C is

$$C = \left\{(u,v) : 0 \leq u \leq \left(\frac{1}{2\pi\sigma^2}\right)^{1/4} e^{-(v-u\theta)^2/4\sigma^2u^2}\right\}$$

In order to draw the region later, we need to rearrange the bound above, which gives us (by taking log):

$$(v - \theta u)^2 \leq 4\sigma^2 u^2 \left(\log u + \frac{1}{2} \log(2\pi\sigma^2) \right) .$$

The above defines the region C. Now, in order to bound the region C, we find the limits a, b, c: (The following calculations were incorrect in class. I have now fixed it here.)

$$a = \sup_{x \in \mathbb{R}} (2\pi\sigma^2)^{-1/4} e^{-(x-\theta)^2/4\sigma^2} = (2\pi\sigma^2)^{-1/4}$$

$$b = \inf_{x \leq 0} \left(\frac{1}{2\pi\sigma^2} \right)^{1/4} x e^{-(x-\theta)^2/4\sigma^2} \qquad \text{and} \qquad c = \sup_{x \geq 0} \left(\frac{1}{2\pi\sigma^2} \right)^{1/4} x e^{-(x-\theta)^2/4\sigma^2}$$

First, we find b and then c will follow similarly. Note that b will be non-positive, and thus, to find the infimum, we first take negative and then log. That is, let for $x < 0$, let

$$A(x) = \left(\frac{1}{2\pi\sigma^2} \right)^{1/4} (-x) e^{-(x-\theta)^2/4\sigma^2}$$

Then $A(x)$ is non-negative, and we want to find the supremem of $A(x)$ for $x \leq 0$. Taking log:

$$\begin{aligned} \log(A(x)) &= -\frac{1}{4} \log(2\pi\sigma^2) + \log(-x) - \frac{(x-\theta)^2}{4\sigma^2} \\ \Rightarrow \frac{d \log(A(x))}{dx} &= \frac{1}{x} - \frac{(x-\theta)}{2\sigma^2} \overset{set}{=} 0 \\ \Rightarrow x &= \frac{\theta \pm \sqrt{\theta^2 + 8\sigma^2}}{2} \end{aligned}$$

Now, we need to decide which of $\pm$ would be choose. Note that $\sqrt{\theta^2 + 8\sigma^2} > \theta$. Hence, since we are taking $\sup_{x \geq 0} A(x)$, we obtain

$$x_b := \frac{\theta - \sqrt{\theta^2 + 8\sigma^2}}{2}$$

Thus,

$$b = x_b f^{1/2}(x_b)$$

Similarly, we obtain

$$x_c := \frac{\theta + \sqrt{\theta^2 + 8\sigma^2}}{2\sigma^2}$$

with

$$c = x_c f^{1/2}(x_c)\,.$$

All that needs to be done now is to implement Algorithm 12 with these values of a, b, c, given the values of θ and σ^2

■

Questions to think about

1. Construct a similar RoU sampler for Cauchy distribution.
2. Why does RoU fail when C is unbounded?
3. For $N(0,1)$ between RoU and AR using Cauchy proposal, which is more efficient, in terms of the expected number of uniforms required for one acceptance?

4.5 Miscellaneous methods in sampling

4.5.1 Known relationships

It is always useful to remember the relationships between different distributions.

1. **Binomial distribution**: We know that if $Y_1, Y_2, \dots, Y_n \overset{iid}{\sim} \text{Bern}(p)$, then

   $$X = Y_1 + Y_2 + \dots Y_n \sim \text{Bin}(n,p)\,.$$

   So, we can simulate n Bernoulli variables, add them up, and we have a realization from a Binomial(n, p).

2. **Negative binomial distribution**: Number of failures until the rth success. So possibly related to geometric! If $Y_1, Y_2, \dots, Y_r \overset{iid}{\sim} \text{Geom}(p)$ (on failures), then

   $$X = Y_1 + Y_2 + \cdots + Y_r \sim NB(r,p)\,.$$

3. **Beta distribution** If $X \sim \text{Gamma}(a, 1)$ and $Y \sim \text{Gamma}(b, 1)$, then

   $$\frac{X}{X+Y} \sim \text{Beta}(a,b)\,.$$

4. **Dirichlet distribution :** The Dirichlet distribution is a distribution over pmf.

$$f(x_1, x_2, \dots, x_k) = \frac{\Gamma(\alpha_1 + \cdots + \alpha_k)}{\prod_{i=1}^k \Gamma(\alpha_i)} \prod_{i=1}^k x_i^{\alpha_i - 1} \quad 0 \le x_i \le 1, \sum_{i=1}^k x_i = 1\,.$$

The Dirichlet distribution is a generalization of the Beta distribution. Similarly,

$$\begin{aligned} Y_1 &\sim \text{Gamma}(\alpha_1, 1) \\ Y_2 &\sim \text{Gamma}(\alpha_2, 1) \\ &\vdots \\ Y_k &\sim \text{Gamma}(\alpha_k, 1) \end{aligned}$$

Let

$$X_i = \frac{Y_i}{\sum_{i=1}^k Y_i}\,.$$

Then $(X_1, \dots, X_k) \sim \text{Dir}(\alpha_1, \alpha_2, \dots, \alpha_k)$.

5. **Chi-squared distribution**: If $Y_1, Y_2, \dots, Y_k \overset{iid}{\sim} N(0,1)$, then

$$X = Y_1^2 + Y_2^2 + \cdots + Y_k^2 \sim \chi_k^2\,.$$

This way we can simulate χ^2 distributions with integer degrees of freedom.

6. ***t*-distribution** Let $Z \sim N(0,1)$ and $Y \sim \chi_k^2$, then

$$X = \frac{Z}{\sqrt{\frac{Y}{k}}} \sim t_k\,.$$

7. **Location-scale family**: Let F be a distribution in the location-scale family. Then, if Z has CDF $F_Z(z)$ in the sense that it doesn't have any parameters. Then for $\mu \in R$ and $\sigma > 0$,

$$Y = \mu + \sigma Z \text{ has CDF } F_Y(y) = F_Z\left(\frac{z - \mu}{\sigma}\right)\,.$$

If Z has pdf $f(z)$ then Y has pdf $\sigma^{-1} f((z - \mu)/\sigma)$.

So, if $Z \sim N(0,1)$, then $Y = \mu + \sigma Z \sim N(\mu, \sigma^2)$.

4.5.2 Sampling from mixture distributions

Mixture distributions for continuous densities is very similar to the discrete distributions. For $j = 1, \dots, k$, let $F_j(x)$ be a distribution function and let $f_j(x)$ be the corresponding density function. A mixture distribution function, F is

$$F(x) = \sum_{j=1}^{k} p_j F_j(x) \quad \text{where } p_j > 0, \sum_j p_j = 1$$

and the corresponding density is

$$f(x) = \sum_{j=1}^{k} p_j f_j(x)\,.$$

If we can draw from each F_j then we can draw from the mixture distribution F as well using the same steps as in the discrete case.

Algorithm 13 Sampling from a mixture distribution

```
1: Generate $U \sim U[0,1]$
2: If $U < p_1$, generate from $F_1$
3: If $U < p_1 + p_2$, generate from $F_2$
4: ...
5: If $U < \sum_{i=1}^{k-1} p_i$, generate from $F_{k-1}$
6: Else, generate from $F_k$ .
```

Example 18 (Mixture of normals)**.** Consider two normal distributions $N(\mu_1, \sigma_1^2)$ and $N(\mu_1, \sigma_2^2)$. For some $0 < p < 1$, the mixture density is

$$\begin{aligned} f(x) &= p f_1(x; \mu_1, \sigma_1^2) + (1-p) f_2(x; \mu_2, \sigma_2^2) \\ &= p \frac{1}{\sqrt{2\pi\sigma_1^2}} \exp\left\{ -\frac{1}{2} \left(\frac{x - \mu_1}{\sigma_1} \right)^2 \right\} + (1-p) \frac{1}{\sqrt{2\pi\sigma_2^2}} \exp\left\{ -\frac{1}{2} \left(\frac{x - \mu_2}{\sigma_2} \right)^2 \right\} \end{aligned}$$

Mixture distributions are particularly useful for clustering problems and we will come back to them again in the data analysis part of the course. If we want to sample from this distribution

Algorithm 14 Sampling from a Gaussian mixture

```
1: Generate $U \sim U[0,1]$
2: If $U < p$, generate $N(\mu_1, \sigma_1^2)$ (using location-scale family trick)
3: Otherwise, generate $N(\mu_2, \sigma_2^2)$.
```

■

4.5.3 Multidimensional target

We have almost entirely focused on univariate densities, but most often interest is in multivariate/multidimensional target distribution.

- **Conditional Distribution:** Consider a variable $\mathbf{X} = (X_1, X_2, \dots, X_k)$, with a joint pdf
$$f(\mathbf{x}) = f(x_1, x_2, \dots, x_k)\,.$$
We can use conditional distribution properties:
$$f(\mathbf{x}) = f_{X_1}(x_1) f_{X_2|X_1}(x_2) \dots f_{X_k|X_1,\dots,X_{k-1}}(x_k)\,.$$

Algorithm 15 Sampling $\mathbf{X}$ using conditional distributions

```
1: Generate $X_1 \sim f_{X_1}(x_1)$
2: Generate $X_2 \sim f_{X_2|X_1}(x_2)$
3: Generate $X_3 \sim f_{X_3|X_2,X_1}(x_3)$
4: $\vdots$
5: Generate $X_n \sim f_{X_k|X_{k-1},\dots,X_1}(x_k)$
6: Return $\mathbf{X} = (X_1, \dots, X_k)$
```

- **Multivariate normal:** Consider sampling from a $N_k(\mu, \Sigma)$ where Σ is positive definite. Then for $|\cdot|$ denoting determinant,
$$f_{\mathbf{X}}(\mathbf{x}) = \left(\frac{1}{2\pi}\right)^{k/2} |\Sigma|^{-1/2} \exp\left\{-\frac{(\mathbf{x}-\mu)^T \Sigma^{-1} (\mathbf{x}-\mu)}{2}\right\}\,,$$

is the density of a multivariate normal distribution with mean μ and covariance Σ. First, note that since Σ is a positive-definite (symmetric) matrix, we can use the eigenvalue decomposition

$$\Sigma = Q\Lambda Q^{-1}$$

where Q is the matrix of eigenvectors and since Σ is symmetric, Q is guaranteed to be an orthongal matrix so that $Q^{-1} = Q^T$ and Λ is a diagonal matrix of eigenvalues. Then, we can define the *square-root* of Σ as

$$\Sigma^{1/2} := Q\Lambda^{1/2}Q^{-1}\,,$$

so that

$$\Sigma^{1/2}\Sigma^{1/2} = Q\Lambda^{1/2}Q^{-1}Q\Lambda^{1/2}Q^{-1} = Q\Lambda Q^{-1}\,.$$

Similarly, the inverse square-root is

$$\Sigma^{-1/2} = Q\Lambda^{-1/2}Q^{-1}\,,$$

Set $\mathbf{Z} = \Sigma^{-1/2}(\mathbf{X} - \mu)$. Then

$$\mathbf{Z} \sim N_k(0, I_k)\,.$$

That is, $\mathbf{Z}$ is a k-dimensional multivariate normal distribution with an identity covariance matrix. Which implies if $\mathbf{Z} = (Z_1, \dots, Z_k)$, then $\text{Cov}(Z_i, Z_j) = 0$ for all $i \neq j$.

For the normal distribution, if the covariance is zero, then the random variables are independent! This isn't true in general but is true for normal random variables.

So, to sampling from $N_k(\mu, \Sigma)$, we can sample $Z_1, Z_2, \dots, Z_k \overset{iid}{\sim} N(0, 1)$, and set $\mathbf{Z} = (Z_1, \dots, Z_k)$. Then

$$\mathbf{X} := \mu + \Sigma^{1/2}\mathbf{Z} \sim N_k(\mu, \Sigma)\,.$$

Then $\mathbf{Z} \sim N_k(\mu, \Sigma)$.

Questions to think about

- Can you construct a zero-inflated normal distribution and find a suitable application of it?

4.6 Exercises

1. Using the inverse transform method, simulate from $\text{Exp}(\lambda)$ for any $\lambda > 0$. Implement this for $\lambda = 5$.

2. Use the inverse transform method to obtain samples from the $\text{Weibull}(\alpha, \lambda)$

$$f(x) = \alpha\lambda x^{\alpha-1}e^{-\lambda x^{\alpha}}\,, \qquad x > 0\,.$$

3. (Ross 5.1) Give a method for generating a random variable having density function

$$f(x) = \frac{e^x}{e-1} \qquad 0 \leq x \leq 1\,.$$

4. (Ross 5.2) Give a method for generating a random variable having density function

$$f(x) = \begin{cases} \dfrac{x-2}{2} & \text{if } 2 \leq x \leq 3 \\ \dfrac{2-x/3}{2} & \text{if } 3 \leq x \leq 6 \end{cases}$$

5. (Ross 5.3) Use the inverse transform method to generate a random variable having distribution function

$$F(x) = \frac{x^2 + x}{2} \qquad 0 \leq x \leq 1\,.$$

6. Sample following the following distribution using two different methods:

$$f(x) = \begin{cases} \dfrac{3}{4}(1-x^2) & \text{if } x \in (-1, 1) \\ 0 & \text{otherwise} \end{cases}$$

7. (Ross 5.6) Let X be an $\text{Exp}(1)$. Provide an efficient algorithm for simulating a random variable whose distribution is the conditional distribution of X given

that $X < 0.05$. That is, its density function is

$$f(x) = \frac{e^{-x}}{1 - e^{-0.05}} \qquad 0 < x < 0.05\,.$$

Using `R` generate 1000 such random variables and use them to estimate $\mathrm{E}[X \mid X < 0.05]$.

8. (Ross 5.7) Suppose it is relatively easy to generate random variables from any of the distributions F_i, $i = 1, \ldots, k$. How could we generate a random variable from the distribution function

$$F(x) = \sum_{i=1}^{n} p_i F_i(x)\,,$$

where $p_i \geq 0$ and $\sum p_i = 1$.

9. (Ross 5.8) Using the previous exercise, provide algorithms for generating random variables from the following distributions:

(a) $F(x) = \frac{x+x^3+x^5}{3}, 0 \leq x \leq 1$.

(b) $F(x) = \begin{cases} \frac{1-e^{-2x}+2x}{3} & \text{if } x \in (0,1) \\ \frac{3-e^{-2x}}{3} & \text{if } x \in [1,\infty) \end{cases}$

10. (Ross 5.9) Give a method to generate a random variable with distribution function

$$F(x) = \int_0^\infty x^y e^{-y} dy \qquad 0 \leq x \leq 1$$

11. (Ross 5.15) Give two methods for generating a random variable with density function

$$f(x) = xe^{-x}, 0 \leq x < \infty\,.$$

12. (Ross 5.18) Give an algorithm for generating a random variable having density function

$$f(x) = 2xe^{-x^2}\,, \qquad x > 0\,.$$

13. (Ross 5.19) Show how to generate a random variable who distribution function is

$$F(x) = \frac{x + x^2}{2}\,, \qquad 0 \leq x \leq 1$$

using the inverse transform, accept-reject, composition method.

14. (Ross 5.20) Use the AR method to find an efficient way to generate a random variable having density function

$$f(x) = \frac{(1+x)e^{-x}}{2} \quad 0 < x < \infty\,.$$

15. (Ross 5.21) Consider the target density to be a truncated Gamma$(\alpha, 1)$, $\alpha < 1$ defined on (a, ∞) for some $a > 0$. Suppose the proposal distribution is a truncated exponential(λ), defined on the same (a, ∞). What is the best λ to use?

16. (Using R)

    (a) Implement an accept-reject sampler to sample uniformly from the circle $\{x^2 + y^2 \leq 1\}$ and obtain 10000 samples and estimate the probability of acceptance. Does it approximately equal $\pi/4$?

    (b) Now consider sampling uniformly from a p-dimensional sphere (a circle is $p = 2$). Consider a p-vector $\mathbf{x} = (x_1, x_2, \dots, x_p)$ and let $\|\cdot\|$ denote the Euclidean norm. The pdf of this distribution is

    $$f(\mathbf{x}) = \frac{\Gamma\left(\frac{p}{2}+1\right)}{\pi^{p/2}} I\{\|\mathbf{x}\| \leq 1\}\,.$$

    Use a uniform p-dimensional hypercube to sample uniformly from this sphere. Implement this for $p = 3, 4, 5$, and 6. What happens as p increases?

17. (Using R)

    (a) Using accept-reject and a standard normal proposal, obtain samples from a truncated standard normal distribution with pdf:

    $$f(x) = \frac{1}{\Phi(a) - \Phi(-a)} \frac{1}{\sqrt{2\pi}} e^{-x^2/2} I(-a < x < a)\,,$$

    where $\Phi(\cdot)$ is the CDF of a standard normal distribution. Run for $a = 4$ and $a = 1$. What are the differences between the two settings.

    (b) Now consider a multivariate truncated normal distribution, where for $\mathbf{x} = (x_1, x_2, \dots, x_p)$, the pdf is

    $$f(\mathbf{x}) = \left(\frac{1}{\Phi(a) - \Phi(-a)}\right)^p \left(\frac{1}{\sqrt{2\pi}}\right)^p e^{-\mathbf{x}^T\mathbf{x}/2} I(-a < \mathbf{x} < a)\,.$$

Implement an accept-reject sampler with proposal distribution $N_p(0, I)$ with $a = 4$ and $p = 3, 10$ and with $a = 1$ and $p = 3, 10$. Describe the differences between these settings.

18. Implement an accept-reject sampler to draw from a Gamma$(\alpha, 1)$ for $\alpha > 1$. Using the above method, can you draw samples from Gamma(α, β), for any β?

19. In accept-reject sampling, why is $c \geq 1$?

20. Use ratio-of-uniforms method to sample from a truncated exponential distribution with density
$$f(x) = \frac{e^{-x}}{1 - e^{-a}} \quad 0 < x < a\,.$$
How efficient is this algorithm?

21. Use ratio-of-uniforms method to sample from the distribution with density
$$f(x) = \frac{1}{x^2} \quad x \geq 1\,.$$

22. Use ratio-of-uniforms method to draw samples from a t_ν distribution for $\nu \geq 1$.

23. (Zero-inflated Gamma distribution) Suppose you are an auto-insurance company and you want to study the cost of claims associated with each customer. That is, each customer, if they have an accident, will come to you and claim insurance money reimbursement for the accident. So

    Let X = insurance money asked for by a customer in a month.

    However, most customers will not enter into any accidents, so they will claim Rs 0. But when they do, they will claim reimbursement for some amount of money that, say, will follow a Gamma distribution.

    The density function can be defined as follows for $0 < p < 1$
$$f(x) = p\mathbb{I}(x = 0) + (1 - p)\frac{\beta^\alpha}{\Gamma(\alpha)}x^{\alpha-1}e^{-x\beta}\,.$$
    Provide an algorithm to sample this random variable.

5 Importance Sampling

We have so far learned many (many!) ways of sampling from different distributions. These sampling methodologies are particularly useful when we want to estimate characteristic of F. Using computer simulated samples from F to estimate characteristics of F is broadly termed as *Monte Carlo*

5.1 Simple Monte Carlo

Suppose F is a distribution with density f. We are interested in estimating the expectation of a function $h : \mathcal{X} \to \mathbb{R}$ with respect to F. That is, we want to estimate

$$\theta := \mathrm{E}_F[h(X)] = \int_{\mathcal{X}} h(x) f(x)\, dx < \infty \,,$$

we assume that θ is finite. We also assumed that

$$\sigma^2 = \mathrm{Var}_F(h(X)) < \infty \,.$$

Note: there is no "data" here, there is just an integral! We are just interested in estimating an annoying integral.

Note: notation $E_F[X]$ means the expectation is with respect to F. From now on, it is very important to keep track of what the expectation is with respect to.

Suppose we can draw iid samples $X_1, \dots, X_N \overset{\text{iid}}{\sim} F$ (this we can do using the many methods we have learned). Then, by the weak law of large numbers, as $N \to \infty$,

$$\hat{\theta} = \frac{1}{N} \sum_{t=1}^{N} h(X_t) \xrightarrow{p} \theta \,.$$

In addition, we can find the variance of the estimator:

$$\begin{aligned}
\mathrm{Var}(\hat{\theta}) &= \mathrm{Var}\left(\frac{1}{N} \sum_{t=1}^{N} h(X_t) \right) \\
&= \frac{1}{N^2} \sum_{t=1}^{N} \mathrm{Var}_F(h(X_t)) \qquad \text{because of independence} \\
&= \frac{\mathrm{Var}_F(h(X_1))}{N} \qquad \text{because of identical}
\end{aligned}$$

$$= \frac{\sigma^2}{N} \,.$$

Naturally, a central limit theorem also holds if $\sigma^2 < \infty$, so that as $N \to \infty$

$$\sqrt{N}(\hat{\theta} - \theta) \xrightarrow{d} N(0, \sigma^2) \,,$$

This central limit theorem gives us an expected behavior of $\hat{\theta}$ for large values of n.

Q. But is there a way we can obtain a better estimator of θ?

A. Possibly by using importance sampling.

5.2 Simple importance sampling

Our goal is the same. For $h : \mathcal{X} \to \mathbb{R}$, we want to estimate $\theta = \mathrm{E}_F[h(X)]$. Similar to the the accept-reject sampler, we will choose a proposal distribution. Let G be a distribution with density g defined on $\mathcal{X}$ so that,

$$\begin{aligned}
\mathrm{E}_F[h(X)] &= \int_{\mathcal{X}} h(x) f(x) dx \\
&= \int_{\mathcal{X}} \frac{h(x) f(x)}{g(x)} g(x)\, dx \\
&= \mathrm{E}_G\left[\frac{h(Z) f(Z)}{g(Z)}\right], \qquad Z \sim G
\end{aligned}$$

If $Z_1, \dots, Z_N \overset{\text{iid}}{\sim} G$, then an estimator of θ is

$$\hat{\theta}_g = \frac{1}{N} \sum_{t=1}^{N} \frac{h(Z_t) f(Z_t)}{g(Z_t)} \,.$$

The estimator $\hat{\theta}_g$ is the *importance sampling estimator*, the method is called *importance sampling* and G is the *importance distribution.*

Let

$$w(Z_t) = \frac{f(Z_t)}{g(Z_t)}$$

be the weights assigned to each point Z_t. Then $\hat{\theta}_g$ is a weighted average of of $h(Z_t)$. Intuitively, this means that depending on how likely a sampled value is for f and g, a weight is assigned to that value.

Example 19 (Moments of Gamma distribution)**.** Suppose we want to estimate the kth moment of a Gamma distribution. That is, let F be the density of a Gamma(α, β) distribution. Then

$$\theta = \int_0^\infty x^k \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} dx\,.$$

Suppose we set G to be also an Exponential(λ) distribution. Let $Z_1, \dots, Z_N \sim \text{Exp}(\lambda)$

$$\begin{aligned}
\hat{\theta}_g &= \frac{1}{N} \sum_{t=1}^{N} \left[\frac{h(Z_t) f(Z_t)}{g(Z_t)} \right] \\
&= \frac{1}{N} \sum_{t=1}^{N} \left[\frac{\beta^\alpha}{\Gamma(\alpha)} \frac{Z_t^k Z_t^{\alpha-1} e^{-\beta Z_t}}{\lambda e^{-\lambda Z_t}} \right] .
\end{aligned}$$

■

So we have now constructed an alternative estimator of θ. In fact, a different choice of G will yield a different estimator. It is now important to study the properties of this importance sampling estimator. We study a sequence of properties.

Theorem 5 (Unbiasedness)**.** The importance sampling estimator $\hat{\theta}_g$ is unbiased for θ.

Proof. To show an estimator is unbiased, we need to show that $\text{E}[\hat{\theta}_g] = \theta$. Consider

$$\begin{aligned}
\text{E}\left[\hat{\theta}_g\right] &= \text{E}_G \left[\frac{1}{N} \sum_{t=1}^{N} \frac{h(Z_t) f(Z_t)}{g(Z_t)} \right] \\
&= \frac{1}{N} \sum_{t=1}^{N} \text{E}_G \left[\frac{h(Z_t) f(Z_t)}{g(Z_t)} \right] \\
&= \frac{1}{N} \sum_{t=1}^{N} \text{E}_G \left[\frac{h(Z_1) f(Z_1)}{g(Z_1)} \right] \\
&= \int_{\mathcal{X}} \frac{h(z) f(z)}{g(z)} g(z)\, dz \\
&= \int_{\mathcal{X}} h(z) f(z) dz \\
&= \theta\,.
\end{aligned}$$

□

Theorem 6. *The importance sampling estimator is consistent for θ. That is, as $N \to \infty$,*

$$\hat{\theta}_g \xrightarrow{p} \theta \,.$$

Proof. Note that $\hat{\theta}_g$ is just a sample average:

$$\hat{\theta}_g = \frac{1}{N} \sum_{t=1}^{N} \frac{h(Z_t) f(Z_t)}{g(Z_t)} \,.$$

The law of large numbers applies to any sample average whose expectation is finite. So by the law of large numbers, as $N \to \infty$,

$$\hat{\theta}_g \xrightarrow{p} \mathrm{E}[\hat{\theta}_g] = \theta \,.$$

□

This means that as we get more and more samples from G, our estimator will get increasingly closer to the truth.

However, we should never be happy with a point estimator!

It is essential to quantify the variability in our estimator $\hat{\theta}_g$ in order to ascertain how "erratic" or "stable" the estimator is. We also want to establish expected behavior for $\hat{\theta}_g$, but *does a central limit theorem hold?*. Notice that a simple Monte Carlo is just a sample average, so we should be able to directly apply the CLT result, if the variance is finite. Note that, the variance of $\hat{\theta}_g$ is

$$\mathrm{Var}(\hat{\theta}_g) = \mathrm{Var}_g \left(\frac{1}{N} \sum_{t=1}^{N} \frac{h(Z_t) f(Z_t)}{g(Z_t)} \right) = \frac{1}{N} \mathrm{Var}_g \left(\frac{h(Z_1) f(Z_1)}{g(Z_1)} \right) =: \frac{\sigma_g^2}{N} \,.$$

A central limit theorem will hold if $\sigma_g^2 = \mathrm{Var}_g \left(\dfrac{h(Z_1) f(Z_1)}{g(Z_1)} \right) < \infty$.

So the question is, when is this finite?

The following theorem provides a sufficient condition.

Theorem 7. Suppose $\sigma^2 = \text{Var}_F(h(X)) < \infty$. If g is chosen such that

$$\sup_{z \in \mathcal{X}} \frac{f(z)}{g(z)} \leq M < \infty$$

then

$$\sigma_g^2 < \infty \,.$$

Proof. First note that if the variance of a random variable is finite, this is equivalent to saying that the second moment of that variable is finite. So, consider the second moment of $\frac{h(Z)f(Z)}{g(Z)}$ where $Z \sim G$.

$$\begin{aligned}
\text{E}_G\left[\left(\frac{h(Z)f(Z)}{g(Z)}\right)^2\right] &= \int_{\mathcal{X}} \frac{h(z)^2 f(z)^2}{g(z)^2} g(z) dz \\
&= \int_{\mathcal{X}} h(z)^2 \, \frac{f(z)}{g(z)} \, f(z) dz \\
&\leq M \int_{\mathcal{X}} h(z)^2 f(z) dz \\
&= M \, \text{E}_F(h(X)^2) < \infty \quad \text{ by assumption} \,.
\end{aligned}$$

$\square$

Thus, if an accept-reject is possible for the propogal G, then a simple importance sampling estimator of θ, with a finite variance, is also possible. Now, we have a central limit theorem that can hold. Recall

$$\sigma_g^2 = \text{Var}_G\left(\frac{h(Z)f(Z)}{g(Z)}\right) \,. \tag{1}$$

By the CLT, if $\sigma_g^2 < \infty$, then as $N \to \infty$,

$$\sqrt{N}(\hat{\theta}_g - \theta) \xrightarrow{d} N(0, \sigma_g^2) \,. \tag{2}$$

Further, an estimator of σ_g^2 is easily available since we have N samples of $h(Z)f(Z)/g(Z)$ available. Thus, an estimator of σ_g^2 is the sample variance from all the samples:

$$\hat{\sigma}_g^2 := \frac{1}{N-1}\left(\frac{h(Z_t)f(Z_t)}{g(Z_t)} - \hat{\theta}_g\right)^2 \,.$$

Example 20 (Gamma continued)**.** Recall from the accept-reject example for Gamma(α, β) with $\alpha \geq 1$ and Exponential(λ) proposal for an accept-reject sampler will work only if $\lambda < \beta$. That means, when $\lambda < \beta$, there exists a finite M, and the importance sampling estimator will have a finite variance. ■

Questions to think about

1. Can we construct G so that its support, $\mathcal{Y}$ is larger than $\mathcal{X}$?
2. Check what happens with $\beta = \lambda$ in this simulation.
3. Why would a CLT be useful here?
4. How would we check whether this importance sampler is better than IID Monte Carlo?

5.2.1 Optimal proposals

How do we choose the importance distribution g? The proposal g should be chosen so that:

- Sampling from G is relatively easy
- $\text{Var}_g(\hat{\theta}_g) = \sigma_g^2/N$ is smaller than regular Monte Carlo variance estimator.

Note that, one reason to use importance sampling would be to obtain smaller variance estimators than the original. So, if we can choose g such that σ_g^2 is minimized that would be ideal!

Let's see this term:

$$\sigma_g^2 = \text{Var}_G\left(\frac{h(Z)f(Z)}{g(Z)}\right) = \text{E}_G\left[\frac{h(Z)^2 f(Z)^2}{g(Z)^2}\right] - \theta^2 = \underbrace{\int_{\mathcal{X}} \frac{h(z)^2 f(z)^2}{g(z)} dz}_{A} - \theta^2$$

For the above to be small, term A should be close to θ^2. This logic leads to the following theorem.

Theorem 8. If $\int_{\mathcal{X}} |h(x)| f(x) dx \neq 0$, the importance density g^* that minimizes σ_g^2 is

$$g^*(z) = \frac{|h(z)|f(z)}{\text{E}_F\left[|h(x)|\right]}\,.$$

Proof. Consider the above importance density. The second moment of the importance sampling estimator with this density is:

$$
\begin{aligned}
&\theta^2 + \sigma^2_{g^*} \\
&= \mathrm{E}_{G^*}\left[\left(\frac{h(Z)f(Z)}{g^*(Z)}\right)^2\right] \\
&= \int_{\mathcal{X}} \frac{h(z)^2 f(z)^2}{g^*(z)^2} g^*(z)dz \\
&= \int_{\mathcal{X}} \frac{h(z)^2 f(z)^2}{|h(z)|f(z)} \cdot \mathrm{E}_F\left[|h(x)|\right] dz \\
&= \mathrm{E}_F\left[|h(x)|\right] \int_{\mathcal{X}} |h(z)|f(z)dz \\
&= \left[\int_{\mathcal{X}} |h(z)|f(z)dz\right]^2 \\
&= \left[\int_{\mathcal{X}} \frac{|h(z)|f(z)}{g(z)} g(z)dz\right]^2 \quad \text{for any other } g \text{ defined on } \mathcal{X} \\
&= \left(\mathrm{E}_G\left[\frac{|h(z)|f(z)}{g(z)}\right]\right)^2 \\
&\leq \mathrm{E}_G\left[\frac{h(z)^2 f(z)^2}{g^2(z)}\right] \quad \text{By Jensen's inequality: for a convex function } \phi,\ \phi(E[x]) \leq E(\phi(x)) \\
&= \theta^2 + \sigma^2_g\,.
\end{aligned}
$$

Thus, for any generic proposal g defined on $\mathcal{X}$, we have

$$\sigma^2_{g^*} \leq \sigma^2_g\,.$$

Since this is true for all g, this implies that g^* produces the smallest $\sigma^2_{g^*}$. □

Note that, with this choice of proposal,

$$
\begin{aligned}
\sigma^2_{g^*} &= \mathrm{Var}_{g^*}\left(\frac{h(Z)f(Z)}{g^*(Z)}\right) \\
&= \mathrm{E}_F\left[|h(x)|\right]^2 \mathrm{Var}_{G^*}\left(\frac{h(Z)f(Z)}{|h(Z)|f(Z)}\right) \\
&= \mathrm{E}_F\left[|h(z)|\right]^2 \mathrm{Var}_{G^*}\left(\frac{h(Z)f(Z)}{|h(Z)|f(Z)}\right)\,.
\end{aligned}
$$

If on the support $\mathcal{X}$, $h(Z) = |h(Z)|$, then the variance of the importance sampling estimator is zero!

Example 21 (Gamma distribution)**.** Consider estimating moments of a Gamma(α, β) distribution. We actually know the optimal importance distribution here! For estimating the kth moment

$$\begin{aligned} g^*(z) &\propto |h(z)|f(z) \\ &= |x^k| x^{\alpha-1} \exp\{-\beta x\} \\ &= x^{\alpha+k-1} \exp\{-\beta x\} \ . \end{aligned}$$

So the optimum importance distribution is Gamma$(\alpha+k, \beta)$. The variance in this case of the estimator will be 0. ■

Example 22 (Mean of standard normal)**.** Let $h(x) = x$ and let $f(x)$ be the density of a standard normal distribution. So we are interested in estimating the mean of the standard normal distribution. The universally optimal proposal in this case is

$$g^*(x) = \frac{|x|e^{-x^2/2}}{\int |x|e^{-x^2/2} dx}$$

But it may be quite challenging to draw samples from the above distribution! In order for importance sampling to be useful, we need not find the optimal proposal, as long as we can find a *more* efficient proposal than sampling from the target.

Consider an importance distribution of $N(0, \sigma^2)$ for some $\sigma^2 > 0$. The variance of the importance estimator is

$$\begin{aligned} \sigma_g^2 &= \int_{-\infty}^{\infty} \frac{h(x)^2 f(x)^2}{g(x)} dx \\ &= \int_{-\infty}^{\infty} x^2 \frac{\sigma}{\sqrt{2\pi}} \exp\left\{\frac{x^2}{2\sigma^2} - x^2\right\} dx \\ &= \sigma \int_{-\infty}^{\infty} x^2 \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{x^2}{2}\left(2 - \frac{1}{\sigma^2}\right)\right\} dx \\ &= \frac{\sigma}{\sqrt{2-\sigma^{-2}}} \int_{-\infty}^{\infty} x^2 \cdot \underbrace{\sqrt{\frac{2-\sigma^{-2}}{2\pi}} \exp\left\{-\frac{x^2}{2}\left(2 - \frac{1}{\sigma^2}\right)\right\}}_{\text{density of } N(0,(2-\sigma^{-2})^{-1}) \text{ if } \sigma^2 > 1/2} dx \\ &= \frac{\sigma}{\sqrt{2-\sigma^{-2}}} \frac{1}{2-\sigma^{-2}} \\ &= \frac{\sigma}{(2-\sigma^{-2})^{3/2}} \qquad \text{if } \sigma^2 > 1/2 \end{aligned}$$

else if $\sigma^2 < 1/2$, the integral diverges and the variance is infinite. Also, minimizing the variance:

$$\arg \min_{\sigma > \sqrt{1/2}} \frac{\sigma}{(2 - \sigma^{-2})^{3/2}} = \sqrt{2}\,.$$

Thus the optimal proposal has standard deviation $\sigma = \sqrt{2}$, not 1! Also, at $\sigma^2 = 2$, the variance is .7698 which is less than 1. ■

5.2.2 Questions to think about

- Does this mean that $N(0, 2)$ is the optimal proposal for estimating the mean of a standard normal?
- What is the optimal proposal within the class of *Beta* proposals for estimating the mean of a Beta distribution?

5.3 Weighted Importance Sampling

Often for many distributions, we do not know the target distribution fully, but only know it up to a normalizing constant. That is, for some unknown a, the target density is

$$f(x) = a\tilde{f}(x)$$

and for some known or unknown b, the proposal density is

$$g(x) = b\tilde{g}(x)$$

For simplicity (or rather uniformity in complexity), we will assume that b is unknown. Even though f is not fully known, we are interested in expectations under f. Suppose for some function h, the following integral is of interest:

$$\theta := \int_{\mathcal{X}} h(x)f(x)dx\,.$$

We still want to use g as the importance distribution, so that

$$\theta = \int_{\mathcal{X}} \frac{h(x)f(x)}{g(x)} g(x)\,dx\,.$$

Since a and b are unknown, we can't evaluate $f(x)$ and $g(x)$. So our original estimator does not work anymore! We can evaluate $\tilde{f}(x)$ and $\tilde{g}(x)$. So if we can estimate a and b

as well, that will allow us estimate θ. Instead, we will estimate b/a, which also works!

Consider $Z_1, \dots, Z_t \overset{\text{iid}}{\sim} G$. The *weighted importance sampling* estimator of θ is

$$\hat{\theta}_w := \frac{\displaystyle\sum_{t=1}^{N} \frac{h(Z_t)\tilde{f}(Z_t)}{\tilde{g}(Z_t)}}{\displaystyle\sum_{t=1}^{N} \frac{\tilde{f}(Z_t)}{\tilde{g}(Z_t)}}.$$

Theorem 9. *The weighted importance sampling estimator is consistent. So as $N \to \infty$, $\hat{\theta}_w \xrightarrow{p} \theta$.*

Proof. Intuitively, both the numerator and the denominator are average, so the LLN applies to both individually. Then we may use Slutsky's theorem. This is the plan.

As a first step, note that as $N \to \infty$, by the law of large numbers

$$\frac{1}{N}\sum_{t=1}^{N} \frac{h(Z_t)\tilde{f}(Z_t)}{\tilde{g}(Z_t)} \xrightarrow{p} \mathrm{E}_G\left[\frac{h(Z)\tilde{f}(Z)}{\tilde{g}(Z)}\right] \quad \text{and} \quad \frac{1}{N}\sum_{t=1}^{N} \frac{\tilde{f}(Z_t)}{\tilde{g}(Z_t)} \xrightarrow{p} \mathrm{E}_G\left[\frac{\tilde{f}(Z)}{\tilde{g}(Z)}\right]$$

By an application of Slutsky's theorem, as $N \to \infty$

$$\hat{\theta}_w \xrightarrow{p} \frac{\mathrm{E}_G\left[\dfrac{h(Z)\tilde{f}(Z)}{\tilde{g}(Z)}\right]}{\mathrm{E}_G\left[\dfrac{\tilde{f}(Z)}{\tilde{g}(Z)}\right]}.$$

We need to show that

$$\theta = \frac{\mathrm{E}_G\left[\dfrac{h(Z)\tilde{f}(Z)}{\tilde{g}(Z)}\right]}{\mathrm{E}_G\left[\dfrac{\tilde{f}(Z)}{\tilde{g}(Z)}\right]}.$$

So we will first find both expectations. First

$$\begin{aligned}\mathrm{E}_G\left[\frac{h(Z)\tilde{f}(Z)}{\tilde{g}(Z)}\right] &= \int_{\mathcal{X}} \frac{h(z)\tilde{f}(z)}{\tilde{g}(z)}\, g(z)dz \\ &= \frac{b}{a}\int_{\mathcal{X}} \frac{h(z)f(z)}{g(z)}\, g(z)dz\end{aligned}$$

$$= \frac{b}{a}\theta\,.$$

Second,

$$\begin{aligned} \mathrm{E}_G\left[\frac{\tilde{f}(Z)}{\tilde{g}(Z)}\right] &= \int_{\mathcal{X}} \frac{\tilde{f}(z)}{\tilde{g}(z)} g(z)dz \\ &= \frac{b}{a}\int_{\mathcal{X}} f(z)dz = \frac{b}{a}\,. \end{aligned}$$

So,

$$\frac{\mathrm{E}_G\left[\dfrac{h(Z)\tilde{f}(Z)}{\tilde{g}(Z)}\right]}{\mathrm{E}_G\left[\dfrac{\tilde{f}(Z)}{\tilde{g}(Z)}\right]} = \frac{\frac{b}{a}\theta}{\frac{b}{a}} = \theta\,.$$

□

We will denote

$$w(Z) = \frac{\tilde{f}(Z)}{\tilde{g}(Z)}\,.$$

Then $w(Z)$ is called the un-normalized importance sampling weight.

Thus, even though we do not know a and b, the weighted importance sampling estimator converges to the right quantity and is consistent. However, not knowing b and a comes at a cost. Unlike the simple importance sampling estimator, the weighted importance sampling estimator $\hat{\theta}_w$ is not unbiased.

To see this heuristically, let's assume (just for theory, we actually don't do this) that we obtained two different samples from G for the numerator and the denominator. That is, assume that $Z_1, \dots, Z_N \overset{\text{iid}}{\sim} G$ and $T_1, \dots, T_N \overset{\text{iid}}{\sim} G$. This allows us to break the expectation as follows:

$$\begin{aligned} \mathrm{E}_G\left[\frac{\sum_{t=1}^N h(Z_t)w(Z_t)}{\sum_{t=1}^N w(T_t)}\right] &= \mathrm{E}_G\left[\sum_{t=1}^N h(Z_t)w(Z_t)\right]\mathrm{E}_G\left[\frac{1}{\sum_{t=1}^N w(T_t)}\right] \\ &\geq \mathrm{E}_G\left[\sum_{t=1}^N h(Z_t)w(Z_t)\right]\frac{1}{\mathrm{E}_G\left[\sum_{t=1}^N w(T_t)\right]} \qquad \text{By Jensen's inequality} \end{aligned}$$

$$= \theta\,,$$

where the equality only holds for a constant function only. Thus, even if we have two independent samples for the numerator and denominator, we will still not obtain an unbiased estimator. This is then certainly true for when we use the same sample as well. However, the proof of this is complex, and outside the scope of the course.

Further, an exact expression for the variance is difficult to obtain. Even if the numerator and estimator are assumed to be from independent samples, we may, after a series of approximations (omitted), obtain:

$$\begin{aligned}
\operatorname{Var}\left[\hat{\theta}_w\right] &\approx \frac{\operatorname{Var}_G\left(h(Z)w(Z)\right)}{\left[\mathrm{E}_G(w(Z))\right]^2}\left(1+\frac{\operatorname{Var}_G(w(Z))}{\left[\mathrm{E}_G(w(Z))\right]^2}\right)\\
&= \frac{\operatorname{Var}_G\left(h(Z)w(Z)\right)}{b^2/a^2}\left(1+\frac{\operatorname{Var}_G(w(Z))}{\left[\mathrm{E}_G(w(Z))\right]^2}\right)\\
&= \operatorname{Var}_G\left(ah(Z)w(Z)/b\right)\left(1+\frac{\operatorname{Var}_G(w(Z))}{\left[\mathrm{E}_G(w(Z))\right]^2}\right)\\
&= \operatorname{Var}_G\left(h(Z)\frac{f(Z)}{g(Z)}\right)\left(1+\frac{\operatorname{Var}_G(w(Z))}{\left[\mathrm{E}_G(w(Z))\right]^2}\right)\\
&= \sigma_g^2\left(1+\frac{\operatorname{Var}_G(w(Z))}{\left[\mathrm{E}_G(w(Z))\right]^2}\right)\\
&\geq \sigma_g^2\,,
\end{aligned}$$

where recall that σ_g^2 is the variance coming from the simple importance sampling estimator. This seems to indicate that weighted importance sampling is always worse than simple importance sampling, <u>but in reality this is not true.</u> Here we assumed two independent samples in the approximation above. Since the samples from the denominator and numerator are the same, it is *sometimes* possible for the numerator and denominator to be negatively correlated, so that weighted importance sampling is better!

<u>NOTE:</u> Unlike simple importance sampling, the asymptotic normality of the weighted importance sampling estimator is challenging to verify, and the variance may not always be finite.

However, unlike simple importance sampling, estimating the variance $\operatorname{Var}\left[\hat{\theta}_w\right]$ is no longer straightforward here!

Example 23. Consider estimating

$$\theta = \int_0^\pi \int_0^\pi xy\pi(x,y)dxdy\,,$$

where

$$\pi(x,y) \propto e^{(\sin(xy))} \quad 0 \leq x, y \leq \pi$$

Notice that here, the target distribution is bivariate, but the function $h(x,y) = xy$ is a univariate mapping. Further, the target distribution is not a product of two marginals, so we have to implement multivariate importance sampling. Also, we do not know the normalizing constants. So for some unknown $a > 0$

$$\pi(x,y) = a\, e^{(\sin(xy))} \quad 0 \leq x, y \leq \pi$$

We will use a weighted importance sampling distribution. Consider the importance distribution that is uniform on the box: $U[0,\pi] \times U[0,\pi]$. So that

$$g(z,t) = \frac{1}{\pi^2} I(0 < z < \pi) I(0 < t < \pi)\,.$$

Since, we will assume b is unknown, we realize that

$$\tilde{g}(z,t) = 1 \cdot I(0 < z < \pi) I(0 < t < \pi)\,.$$

Sample $(Z_1, T_1), \dots, (Z_N, T_N) \sim U[0,\pi] \times U[0,\pi]$. The weights are

$$w(Z_t, T_t) = \frac{\tilde{f}(Z_t, T_t)}{\tilde{g}(Z_t, T_t)} = e^{\sin(Z_t T_t)}\,.$$

The final estimator is

$$\hat{\theta}_w = \frac{\sum_{t=1}^N Z_t T_t w(Z_t, T_t)}{\sum_{t=1}^N w(Z_t, T_t)} = \frac{\sum_{t=1}^N Z_t T_t e^{\sin(Z_t T_t)}}{\sum_{t=1}^N e^{\sin(Z_t T_t)}}\,.$$

■

5.4 Questions to think about

- How would you choose a good proposal for weighted importance sampling? Would finding a proposal that yields a small variance suffice?
- Do you have intuition as to why often the variance of the weight importance estimator is larger than the variance of the simple importance sampler for the same importance proposal?

5.5 Exercises

Still working on below. Will add some more exercises by 8th Feb.

1. Estimate $\int_0^1 e^x dx$ using importance sampling.
2. Estimate $\int_{-\infty}^{\infty} e^{-x^2/2}$ using importance sampling.
3. The inverse Gaussian distribution has density

$$f(x) = \sqrt{\frac{\lambda}{2\pi x^3}} \exp\left\{-\frac{\lambda(x-\mu)^2}{2\mu^2 x}\right\} \qquad x > 0 \text{ and } \mu, \lambda > 0\,.$$

   We are interested in estimating the moment generating function of this distribution, $\mathrm{E}_F[e^{tX}]$ for $t \in \mathbb{R}$. Sampling from this Inverse Gaussian distribution can be quite challenging, so we will use importance sampling instead.

   For $\mu = 1$ and $\lambda = 3$, using importance sampling with importance distribution Gamma$(10, 3)$ (rate $= 3$), make a plot with t on the x-axis and the importance sampling estimate of $\mathrm{E}_F[e^{tX}]$ on the y-axis.

   NOTE: First, use the $Z_1, Z_2, \dots, Z_N$ points for all chosen values of t, and then use different importance samples for different values of t.

4. Consider the problem of estimating the k-th moment of a Beta(α, β) distribution. For which values of α and β are we sure to obtain importance estimators of the kth moment with a finite variance for a uniform proposal distribution.
5. In the previous problem, give other examples of importance proposal distributions that will give finite variance of the importance estimator? For what values of α and β is the finite variance of the estimator not guaranteed?
6. Consider estimating the mean of a standard Cauchy distribution using importance sampling with a normal proposal distribution. Does the estimator have

finite variance?

7. For estimating kth moment of a Gamma(α, β) with $\alpha > 1$ with the importance distribution Exp(λ), show that the importance sampling estimator has infinite variance when $\lambda > \beta$.

8. Considering a density $f(x)$ and an importance proposal $g(x)$. Suppose

$$\sup_x \frac{f(x)}{g(x)} < \infty\,.$$

In order to estimate the mean of the target density, is there any benefit to using importance sampling over accept-reject sampling?

9. For a target distribution f and a proposal g, if

$$\sup_x \frac{f(x)}{g(x)} < \infty$$

then we know that the simple importance estimator has finite variance. Does the weighted importance estimator also have finite variance?

10. For some known $y_i \in \mathbb{R}$, $i = 1, \dots, n$ and some $\nu > 2$, suppose the target density is

$$f(x) \propto e^{-x^2/2} \prod_{i=1}^{n} \left(1 + \frac{(y_i - x)^2}{\nu}\right)^{-(\nu+1)/2}\,.$$

Generate ys using the following code for $\nu = 5$

```
set.seed(1)
n <- 50
nu <- 5
y <- rt(n, df = nu)
```

Implement an importance sampling estimator with a $N(0,1)$ proposal to estimate the first moment of this distribution. Does the weighted importance sampling estimator seem to have finite variance? What happens if $\nu = 1$ and $\nu = 2$?

11. Suppose interest is in estimating

$$\theta = \int_0^{10} \exp\left\{-2|x-5|\right\} dx\,.$$

- What is the optimal simple importance proposal distribution? (*Hint:* look up Laplace distribution) and what is the corresponding simple importance sampling estimator?

- Implement a weighted importance sampling procedure with the same proposal distribution from above. How do the final estimators compare?

The quantity θ can be written as

$$\theta = \int_0^{10} 10 \exp\{-2|x-5|\} f(x) dx\,,$$

where $\pi(x)$ is the density of a $U[0,10]$. We know we can do IID sampling that is, sample $X_1, X_2, \dots, X_N$ from Unif$[0,10]$ and estimate θ. But this will be simple Monte Carlo and not importance sampling. Using importance sampling, we can reduce the variance. First, note that the optimal importance distribution here is

$$g^*(z) = \frac{10\exp\{-2|z-5|\}\frac{1}{10}}{\theta} \qquad z \in (0,10)\,.$$

The function h is identical to the density of a Laplace (Double exponential) distribution. For a Laplace random variable with parameters μ and b

$$l(x) = \frac{1}{2b}\exp\left\{-\frac{|z-\mu|}{b}\right\} \quad -\infty < z < \infty$$

So $h(x)$ is the density of Laplace$(5, 1/2)$, but truncated between 0 and 10. So the optimum proposal is a truncated on (0, 10) Laplace$(5, 1/2)$.

You can simulate $Z_1, Z_2, \dots Z_N$ from this g^* using accept-reject algorithm (recall previous exercises) and then the optimal importance sampling estimator of θ is

$$\hat{\theta}_{g^*} = \frac{1}{N}\sum_{t=1}^{N}\frac{h(Z_t)f(Z_t)}{g^*(Z_t)} = \frac{1}{N}\sum_{t=1}^{N}\frac{10\exp\{-2|z-5|\}\frac{1}{10}}{\frac{10\exp\{-2|z-5|\}\frac{1}{10}}{\theta}} = \theta!$$

So notice that for the optimal proposal, the IS estimator is the quantity we want itself! Thus it is impossible to implement the simple importance sampler here.

However, we can do a weighted importance sampling estimator since the normalizing constant for g^* (which is θ) is unknown. So we have

$$g^*(z) = \underbrace{\frac{1}{\theta}}_{b} \underbrace{\exp\{-2|z-5|\}}_{\tilde{g}^*} \qquad z \in (0, 10)$$

We will assume $\tilde{f} = f$, that is $a = 1$.

$$\begin{aligned} \hat{\theta}_w &= \frac{\sum_{t=1}^N \frac{h(Z_t)\tilde{f}(Z_t)}{\tilde{g}^*(Z_t)}}{\sum_{t=1}^N \frac{\tilde{f}(Z_t)}{\tilde{g}^*(Z_t)}} \\ &= \frac{\sum_{t=1}^N \frac{\exp\{-2|Z_t-5|\}}{\exp\{-2|Z_t-5|\}}}{\sum_{t=1}^N \frac{10^{-1}}{\exp\{-2|Z_t-5|\}}} \\ &= 10\frac{N}{\sum_{t=1}^N \exp\{2|Z_t-5|\}} \end{aligned}$$

6 Likelihood Based Estimation

We have learned a fair amount about sampling from various distributions and estimating integrals. For the next few weeks we will focus our attention to optimization methods for certain statistical procedures.

Before we study optimization, we want to motivate why exactly optimization is useful to statisticians. One common use of optimization in statistics is when obtaining a maximum likelihood estimator (MLE) for a parameter. Thus, we first introduce MLE below briefly, before going into optimization methods.

6.1 Likelihood Function

Suppose $X_1, X_2, \dots, X_n$ is a random sample from a distribution with density $f(x|\theta)$ for $\theta \in \Theta$, where Θ is the parameter space. The "x given θ" implies that given a particular value of θ, $f(\cdot|\theta)$ defines a density. $f(\cdot|\theta)$ is also written sometimes as $f_\theta(x)$.

The parameter θ can be a vector of parameters. After having obtained *real data*, from F, we want to

1. estimate θ
2. and assess the quality of the estimator of θ.

A useful method of estimating θ is the method of *maximum likelihood estimation*. Let $\mathbf{X} = (X_1, \dots, X_n)$. The idea is that we define a function $L(\theta|\mathbf{X})$ which measures "how likely is a particular value of θ given the data observed". In general $L(\theta|\mathbf{X} = \mathbf{x})$ is the joint distribution of all the Xs

$$L(\theta|\mathbf{X} = \mathbf{x}) = f(\mathbf{x}|\theta) = f(x_1, x_2, \dots, x_n|\theta)\,.$$

When the sample is independent as well, this likelihood becomes

$$L(\theta|\mathbf{X}) = \prod_{i=1}^{n} f(x_i|\theta)\,.$$

It is important to note that $L(\theta|\mathbf{x})$ is not a distribution over θ, it is just a function of θ. It is a function that quantifies how likely a value of θ is.

Example 24. Suppose we obtain $X_1, X_2 \overset{\text{iid}}{\sim} N(\theta, 1)$, where we don't know θ. Suppose we obtain $X_1 = 2 =: x_1$ and $X_2 = 3 =: x_2$. Then the likelihood function is:

$$\begin{aligned} L(\theta|x_1, x_2) &= f(x_1|\theta) \cdot f(x_2|\theta) \\ &= f(2|\theta) f(5|\theta) \\ &= \frac{1}{2\pi} \exp\left\{ -\frac{(2-\theta)^2}{2} - \frac{(3-\theta)^2}{2} \right\} . \end{aligned}$$

We can plot the above function of θ for different values of θ to understand what the likelihood of every value of θ is. ■

6.2 Maximum Likelihood Estimation

The "most likely" value of θ having observed the data is the value that maximizes the likelihood

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta \in \Theta} L(\theta|\mathbf{x}) .$$

$\hat{\theta}_{\text{MLE}}$ is called the maximum likelihood estimator of θ. As you can understand, maximizing the function $L(\theta|\mathbf{x})$ may be complex for some problems and not possible to do analytically. This is where we require numerical optimization methods to help us obtain these MLEs. MLEs have nice theoretical properties and you will learn them in MTH211a or MTH418a.

Before we continue with some examples, we recall a few definitions:

Definition 1. Concave function (one-dimension): a function $h(x)$ is concave if $h''(x) \leq 0$ for all x. If the equality never holds, it is strictly concave

Definition 2. Concave function : a function $h(\mathbf{x})$ is concave if the Hessian of the matrix, $\nabla^2 h(\mathbf{x})$, is negative semi-definite for all $\mathbf{x}$. That is, if all eigenvalues of the Hessian are non-positive.

6.2.1 Examples

Example 25 (Bernoulli)**.** Let $X_1, \dots, X_n \overset{iid}{\sim} \text{Bern}(p)$, $0 \leq p \leq 1$. Then the likelihood is

$$L(p|\mathbf{x}) = \prod_{i=1}^{n} \Pr(X_i = x_i|p)$$

$$
\begin{aligned}
&= \prod_{i=1}^{n} \left[p^{x_i}(1-p)^{1-x_i} \right] \\
&= p^{\sum x_i}(1-p)^{n-\sum x_i}\,.
\end{aligned}
$$

To obtain the MLE of θ, we will maximize the likelihood. Note that maximizing the likelihood is the same as maximizing the log of the likelihood, but the calculations are easier after taking a log. So we take a log:

$$
\begin{aligned}
\Rightarrow l(p) := \log L(p|\mathbf{x}) &= \left(\sum_{i}^{n} x_i \right) \log p + \left(n - \sum_{i}^{n} x_i \right) \log(1-p) \\
\Rightarrow \frac{dl(p)}{dp} &= \frac{\sum x_i}{p} - \frac{n - \sum x_i}{1-p} \overset{\text{set}}{=} 0 \\
\Rightarrow \hat{p} &= \frac{1}{n} \sum_{t=1}^{n} x_i\,.
\end{aligned}
$$

Taking the second derivative, we obtain

$$
\frac{d^2 l(p)}{dp^2} = -\frac{\sum x_i}{p^2} - \frac{n - \sum x_i}{(1-p)^2} < 0 \qquad \text{for all } p\,.
$$

Thus, the likelihood is concave, and $\hat{p}$ is a global maxima.

In any example, if I do not check the second derivative, you HAVE to check it for yourself.

Thus,

$$
\hat{p}_{\text{MLE}} = \frac{1}{n} \sum_{t=1}^{n} x_i\,.
$$

■

Example 26 (Two parameter exponential)**.** The density of a two parameter exponential distribution is

$$
f(x|\mu, \lambda) = \lambda e^{-\lambda(x-\mu)} \quad x \geq \mu, \quad \mu \in \mathbb{R}, \lambda > 0\,.
$$

We want to compute the MLEs of both λ and μ. The likelihood is

$$
L(\lambda, \mu|\mathbf{x}) = \prod_{t=1}^{n} f(x_i|\mu, \lambda)
$$

$$= \prod_{t=1}^{n} \lambda e^{-\lambda(x_i-\mu)}\, I(x_i \geq \mu)$$
$$= \lambda^n \exp\left\{-\lambda\left(\sum_{i=1}^{n} x_i - n\mu\right)\right\} I(x_1,\ldots,x_n \geq \mu) \quad \forall\mu \text{ and } \lambda > 0\,.$$

But if $X_1,\ldots,X_n \geq \mu \Rightarrow \min\{X_i\} \geq \mu$. So

$$L(\lambda,\mu|\mathbf{x}) = \lambda^n \exp\left\{-\lambda\left(\sum_{i} x_i - n\mu\right)\right\} I\left(\min_i\{x_i\} \geq \mu\right) \quad \forall\mu \text{ and } \lambda > 0\,.$$

We will first try to maximize with respect to μ and then with respect to λ. Note that $L(\lambda,\mu|\mathbf{x})$ is an increasing function of μ within the restriction. So that the MLE of μ is the largest value in the support of μ where $\mu \leq \min\{X_i\}$. So

$$\hat{\mu}_{\text{MLE}} = \min_{1\leq i\leq n}\{X_i\} =: X_{(1)}\,.$$

Next, note that

$$\begin{aligned}
L(X_{(1)},\lambda|\mathbf{x}) &= \lambda^n \exp\left\{-\lambda\left(\sum_i X_i - nX_{(1)}\right)\right\}\\
\Rightarrow l(X_{(1)},\lambda) := \log L(X_{(1)},\lambda|\mathbf{x}) &= n\log\lambda - \lambda\left(\sum X_i - nX_{(1)}\right)\\
\Rightarrow \frac{d\,l}{d\,\lambda} &= \frac{n}{\lambda} - \left(\sum X_i - nX_{(1)}\right) \overset{\text{set}}{=} 0 \quad \text{and}\\
\frac{d^2\,l}{d\,\lambda^2} &= -\frac{n}{\lambda^2} < 0\,.
\end{aligned}$$

So, the log-likelihood function is concave, thus there is a unique maximum. Set

$$\begin{aligned}
\frac{d\,l}{d\,\lambda} &= 0\\
\Rightarrow \frac{n}{\lambda} &= \sum_{t=1}^{n} X_i - nX_{(1)}\\
\Rightarrow \hat{\lambda}_{\text{MLE}} &= \frac{n}{\sum X_i - nX_{(1)}}\,.
\end{aligned}$$

■

6.3 Regression

We will focus a lot on variants of linear regression and thus it is important to setup the premise of linear regression.

Let $Y_1, Y_2, \dots, Y_n$ be observations known as the *response*. Let $x_i = (x_{i1}, \dots x_{ip})^T \in \mathbb{R}^p$ be the ith corresponding vector of covariates for the ith observation. Let $\beta \in \mathbb{R}^p$ be the *regression coefficient* so that for $\sigma^2 > 0$,

$$Y_i = x_i^T \beta + \epsilon_i \quad \text{where } \epsilon_i \sim N(0, \sigma^2)\,.$$

Let $X = (x_1^T, x_2^T, \dots, x_n^T)^T$. In vector form we have,

$$Y = X\beta + \epsilon \sim N_n(X\beta, \sigma^2 I_n)\,.$$

Note: I use capital Y to denote the population random variable and will use the small y to denote realized observations.

The linear regression model is built to estimate β, which measures the linear effect of X on Y. There is much more to linear regression and multiple courses are required to study all aspects of it. However, here we will just focus on the mathematical properties and optimization tools required to study them.

Example 27 (MLE for Linear Regression)**.** In order to understand the linear relationship between X and β, we will need to estimate β. Since we assume that the errors are normally distributed, we have a distribution available for Ys and we may use the method of MLE. We have

$$\begin{aligned} L(\beta, \sigma^2 | y) &= \prod_{t=1}^{n} f(y_i | X, \beta, \sigma^2) \\ &= \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left\{-\frac{1}{2}\frac{(y - X\beta)^T(y - X\beta)}{\sigma^2}\right\} \\ \Rightarrow l(\beta, \sigma^2) := \log L(\beta, \sigma^2|y) &= -\frac{1}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2}\frac{(y - X\beta)^T(y - X\beta)}{\sigma^2} \end{aligned}$$

Note that

$$(y - X\beta)^T(y - X\beta) = (y^T - \beta^T X^T)(y - X\beta)$$

$$
\begin{aligned}
&= y^T y - y^T X\beta - \beta^T X^T y + \beta^T X^T X\beta \\
&= y^T y - 2\beta^T X^T y + \beta^T X^T X\beta \,.
\end{aligned}
$$

Using this we have (recall your multivariable calculus courses)

$$
\begin{aligned}
\frac{dl}{d\beta} &= -\frac{1}{2\sigma^2}\left[-2X^T y + 2X^T X\beta\right] = \frac{X^T y - X^T X\beta}{2\sigma^2} \overset{set}{=} 0 \\
\frac{dl}{d\sigma^2} &= -\frac{n}{2\sigma^2} + \frac{(y - X\beta)^T(y - X\beta)}{2\sigma^4} \overset{set}{=} 0 \,.
\end{aligned}
$$

The first equation leads to $\hat{\beta}_{\text{MLE}}$ satisfying

$$
X^T y - X^T X\hat{\beta}_{\text{MLE}} = 0 \Rightarrow \hat{\beta}_{\text{MLE}} = (X^T X)^{-1} X^T y \,,
$$

if $(X^T X)^{-1}$ exists. And $\hat{\sigma}^2_{MLE}$ is

$$
\hat{\sigma}^2_{MLE} = \frac{(y - X\hat{\beta}_{\text{MLE}})^T(y - X\hat{\beta}_{\text{MLE}})}{n} \,.
$$

Verify: that the Hessian matrix is negative definite, and thus the objective function is concave.

Note: What if $(X^T X)^{-1}$ does not exist?

For example, if $p > n$, then the number of observations is less than the number of parameters, and since X is $n \times p$, $(X^T X)$ is $p \times p$ of rank $n < p$. So $X^T X$ is not full rank and cannot be inverted. In this case, the MLE does not exist and other estimators need to be constructed. This is one of the motivations of *penalized regression*, which we will discuss in detail. ■

6.4 Penalized Regression

Note that in the Linear regression setup, the MLE for β satisfied:

$$
\hat{\beta}_{\text{MLE}} = \arg\min_{\beta} (y - X\beta)^T (y - X\beta)
$$

Suppose X is such that $(X^T X)$ is not invertible, then we don't know how to estimate β. In such cases, we may used *penalized likelihood*, that penalizes the coefficients β so that some of the βs are "pushed towards zero". The corresponding Xs to those small βs are essentially not important, removing singularity from $X^T X$.

Instead of looking at the likelihood, we consider a penalized likelihood. Since the optimization of $L(\beta|y)$ only depends on $(y-X\beta)^T(y-X\beta)$ term, a penalized (negative) log-likelihood is used and the final penalized (negative) log-likelihood is

$$Q(\beta) = -\log L(\beta|y) + P(\beta)$$

Here $P(\beta)$ is a penalization function. Note that since we are now looking at the *negative* log-likelihood, we now want to minimize $Q(\beta)$. The penalization function assigns large values for large β, so that the optimization problem favors small values of β.

There are *many* ways of penalizing β and each method yields a different estimator. A popular one is the *ridge* penalty.

Example 28 (Ridge Regression)**.** The ridge penalization term is $P(\beta) = \lambda\beta^T\beta/2$ for some $\lambda > 0$ for

$$Q(\beta) = \frac{(y - X\beta)^T(y - X\beta)}{2} + \frac{\lambda}{2}\beta^T\beta\,.$$

We will minimize $Q(\beta)$ over the space of β and since we are adding an arbitrary term that depends on the size of β, smaller sizes of β will be preferred. Small sizes of β means X are less important, and this will eventually nullify the singularity in X^TX. The larger λ is, the more "penalization" there is for large values of β; λ is typically user-chosen. We will study choosing λ when we cover "cross-validation" later.

We are now interested in finding:

$$\hat\beta = \arg\min_{\beta}\left\{\frac{(y - X\beta)^T(y - X\beta)}{2} + \frac{\lambda}{2}\beta^T\beta\right\}\,.$$

To carry out the minimization, we take the derivative:

$$\begin{aligned}\frac{dQ(\beta)}{d\beta} &= \frac{1}{2}(-2X^Ty + 2X^TX\beta) + \lambda\beta \overset{set}{=} 0\\ &\Rightarrow (X^TX + \lambda I_p)\hat\beta - X^Ty = 0\\ &\Rightarrow \hat\beta_{\text{ridge}} = (X^TX + \lambda I_p)^{-1}X^Ty\,.\end{aligned}$$

Note: <u>Verify that the Hessian matrix is positive definite for yourself.</u>

Note that $(X^TX + \lambda I_p)$ is always positive definite for $\lambda > 0$ since for any $a \in \mathbb{R}^p \neq 0$

$$a^T(X^TX + \lambda I_p)a = a^TX^TXa + \lambda a^Ta > 0$$

Thus, the final ridge solution always exists even if X^TX is not invertible.

Pros:
We have an estimate of β!
In terms of a certain criterion (we will learn later), we actually do better than non-penalized estimation even when (X^TX) is invertible.

Cons:
The estimator is not an MLE, so we cannot use distributional properties to construct confidence intervals. This is a big problem, and is addressed by bootstrapping, which we will get to.

We will study one more penalization method later. ■

Questions to think about

1. Under the normal likelihood, what is the distribution of $\hat{\beta}_{\text{MLE}}$ (when it exists) and $\hat{\beta}_{\text{ridge}}$? Are they unbiased? Which one has a smaller variance (covariance)?
2. What other penalization functions can you think of? Recall that $\beta^T\beta = \|\beta\|_2^2$.

6.5 No closed-form MLEs

Obtaining MLE estimates for a problem requires maximizing the likelihood. However, it is possible that no analytical form of the maxima is possible!

This is a common challenge in many models and estimation problems, and requires sophisticated optimization tools. In the next few weeks, we will go over some of these optimization methods.

Example 29 (Gamma Distribution)**.** Let $X_1, \dots, X_n \overset{iid}{\sim} \text{Gamma}(\alpha, 1)$. The likelihood function is

$$\begin{aligned} L(\alpha|x) &= \prod_{i=1}^{n} \frac{1}{\Gamma(\alpha)} x_i^{\alpha-1} e^{-x_i} \\ &= \frac{1}{\Gamma(\alpha)^n} e^{-\sum x_i} \prod_{i=1}^{n} x_i^{\alpha-1} \end{aligned}$$

$$\Rightarrow l(\alpha) := \log L(\alpha|x) = -n\log(\Gamma(\alpha)) + (\alpha-1)\sum_{i=1}^{n}\log x_i - \sum_{i=1}^{n} x_i$$

Taking first derivative

$$\frac{dl(\alpha)}{d\alpha} = -n\frac{\Gamma'(\alpha)}{\Gamma(\alpha)} + \sum_{i=1}^{n}\log x_i \overset{set}{=} 0$$

Solving the above analytically is not possible. In fact, the form of $\frac{\Gamma'(\alpha)}{\Gamma(\alpha)}$ is challenging to write analytically.

However, taking second derivative

$$\frac{d^2\,l(\alpha)}{d\alpha^2} = -n\frac{d^2}{d\alpha^2}\log(\Gamma(\alpha)) < 0 \quad \text{(polygamma function of order 1 is} > 0)$$

In the above, we use that

$$\frac{d^2}{d\alpha^2}\log(\Gamma(\alpha))$$

is the *polygamma function of order 1*, which is always positive (look it up). So we know that the function is concave and a unique maximum exists, but not available in closed form.

We cannot get an analytical form of the MLE for α. In such cases, we will use optimization methods. ■

6.6 Exercises

1. *Simple linear regression*: Load the `cars` dataset in R:

   ```
   data(cars)
   ```

   Fit a linear regression model using maximum likelihood with response y being the distance and x being speed. Remember to include an intercept term in X by making the first column as a column of 1s. *Do not use inbuilt functions in R to fit the model.*

2. *Multiple linear regression*: Load the `fuel2001` dataset in R:

   ```
   fuel2001 <- read.csv("https://dvats.github.io/assets/fuel2001.csv",
   row.names = 1)
   ```

   Fit the linear regression model using maximum likelihood with response `FuelC`.

Remember to include an intercept in X.

3. *Simulating data in R:*

   Let $X \in \mathbb{R}^{n \times p}$ be the design matrix, where all entries in its first column equal one (to form an intercept). Let $x_{i,j}$ be the (i,j)th element of X. For the *ith* case, $x_{i1} = 1$ and $x_{i2}, \ldots, x_{ip}$ are the values of the $p-1$ predictors. Let y_i be the response for the ith case and define $y = (y_1, \ldots, y_n)^T$. The model assumes that y is a realization of the random vector

   $$Y \sim N_n(X\beta_*, \sigma_*^2 I_n),$$

   where $\beta_* \in \mathbb{R}^p$ are unknown regression coefficients and $\sigma_*^2 > 0$ is the unknown variance.

   For our simulation, let's pick $n = 50, p = 5, \sigma^2 = 1/2$ and generate the entries of β_* as p independent draws from $N(0,1)$:

```
set.seed(1)
n <- 50
p <- 5
sigma2.star <- 1/2
beta.star <- rnorm(p)
beta.star  # to output
[1] -0.6264538  0.1836433 -0.8356286  1.5952808  0.3295078
```

   We will create the design matrix $X \in \mathbb{R}^{n \times p}$, so that $x_{i1} = 1$ and the other entries are from $N(0,1)$.

```
X <- cbind(1, matrix(rnorm(n*(p-1)), nrow = n, ncol = (p-1)))
```

   Now we will generate a realization of $Y \sim N_n(X\beta_*, \sigma_*^2 I_n)$:

```
y <-  X %*% beta.star + rnorm(n, mean = 0, sd = sqrt(sigma2.star))
```

   In this way we have generate *simulated* data to be used in regression.

4. Find the MLE estimator of β and σ^2 from the previous dataset. Is it close to β_* and σ_*^2? Find the ridge regression solution with $\lambda = 0.01, 1, 10, 100$.

5. *Regression: an equivalent optimization*

   In our original setup $X \in \mathbb{R}^{n \times p}$, all entries in its first column equal to one to form an intercept. The MLE estimate (when it exists) is

$$\hat{\beta} = \arg\min_{\beta \in \mathbb{R}^p} (y - X\beta)^T (y - X\beta)\,.$$

Define X_{-1} be the matrix X with its first column removed. Let $\bar{y} = n^{-1}\sum_{i=1}^n y_i$ and $\bar{x}^T = n^{-1}1_n^T X_{-1} = (n^{-1}\sum_{i=1}^n x_{i2}, \dots, n^{-1}\sum_{i=1}^n x_{ip})$. Let $\tilde{y} = (y_1 - \bar{y}, \dots, y_n - \bar{y})^T$ and $\tilde{X} = X_{-1} - 1_n\bar{x}^T$. Then $\tilde{y}$ is the centered response and $\tilde{X}$ is the centered design matrix.

Suppose that

$$\hat{\beta}_{-1} = \arg\min_{\tilde{\beta} \in \mathbb{R}^{p-1}} (\tilde{y} - \tilde{X}\tilde{\beta})^T (\tilde{y} - \tilde{X}\tilde{\beta})$$
$$\hat{\beta}_1 = \bar{y} - \bar{x}^T \hat{\beta}_{-1}\,.$$

Then $(\hat{\beta}_1, \hat{\beta}_{-1}^T)^T$ is equivalent to $\hat{\beta}$ above. Verify this for the dataset generated in Exercise 3.

6. *Logistic Regression:* Often in regression, the response may be a 0 or 1. That is, the response is a Bernoulli random variable. Let the covariate vector for the ith observation be $x_i = (1, x_{i2}, \dots, x_{ip})^T$. Suppose y_i is a realization of Y_i where

$$Y_i \sim \text{ Bern}\left(\frac{\exp(x_i^T\beta)}{1 + \exp(x_i^T\beta)}\right)\,.$$

Find the MLE of β_*. Does a closed form solution exist?

7 Numerical optimization methods

When obtaining true optimizer is not available analytically, we may use numerical method. A general numerical optimization problem is framed in the following way. Let $f(\theta)$ be an *objective function* that is the main function of interest and needs to be either maximized or minimized. Then, we want to solve the following maximization

$$\theta_* = \arg\max_{\theta} f(\theta)$$

The algorithms we will learn are such that they generate a sequence of $\{\theta_{(k)}\}$ such that the goal is for $\theta_{(k)} \to \theta_*$ in a deterministic manner (non-random convergence).

All methods that we will learn will find a local optima. Some will guarantee a local maxima, but not guarantee a global maxima, some will guarantee a local optima (so max or min), but not a global maxima. <u>If the objective function is concave, then all methods will guarantee a global maxima!</u>

Recall that a (univariate) function f is concave if $f'' < 0$ and a (multivariate) function f is concave if its Hessian is negative definite: for all $a \neq 0 \in \mathbb{R}^p$,

$$a^T \left[\nabla^2 f\right] a < 0\,.$$

7.1 Newton-Raphson's method

To solve this optimization problem, consider starting at a point $\theta_{(0)}$. Then subsequent elements of the sequence are determined in the following way.

Suppose that the objective function f is such that a second derivative exists. Since $f(\theta)$ is maximized at the unknown θ_*,

$$f'(\theta_*) = 0.$$

Applying a first order Taylor's series expansion to f' (and ignoring higher orders) to $f'(\theta_*)$ about the current iterate $\theta_{(k)}$

$$\begin{aligned} f'(\theta_*) &\approx f'(\theta_{(k)}) + (\theta_* - \theta_{(k)})f''(\theta_{(k)}) = 0 \\ \Rightarrow \theta_* &\approx \theta_{(k)} - \frac{f'(\theta_{(k)})}{f''(\theta_{(k)})}\,, \end{aligned}$$

where the approximation is best when $\theta_{(k)} = \theta^*$ and the approximation is weak when $\theta_{(k)}$ is far from θ_*. Thus, if we start from an arbitrary point using successive updates of the right hand side, we will get closer and closer to θ_*.

Using the above argument, the Newton-Raphson method was constructed:

$$\theta_{(k+1)} = \theta_{(k)} - \frac{f'(\theta_{(k)})}{f''(\theta_{(k)})}$$

Intuitively, when $f'(\theta_k) < 0$, the function is increasing at $\theta_{(k)}$, and thus Newton-Raphson increases $\theta_{(k)}$; and vice-versa.

You stop iterating when $|\theta_{(k+1)} - \theta_{(k)}| < \epsilon$ for some chosen tolerance ϵ or $|f'(\theta_{(k+1)})| \approx 0$.

> *If the objective function is concave, the N-R method will converge to the global maxima. Otherwise it converges to a local $\underline{\textit{optima}}$ or diverges!*

Example 30 (Gamma distribution continuted)**.** Our objective function is the log-likelihood:

$$f(\alpha) = -n\log(\Gamma(\alpha)) + (\alpha - 1)\sum_{i=1}^{n}\log x_i - \beta\sum x_i$$

First derivative

$$f'(\alpha) = -n\frac{\Gamma'(\alpha)}{\Gamma(\alpha)} + \sum_{i=1}^{n}\log X_i$$

Second derivative

$$f''(\alpha) = -n\frac{d^2}{d\alpha^2}\log(\Gamma(\alpha)) < 0\,.$$

Thus the log-likelihood is concave, which implies there is a global maxima! The Newton-Raphson algorithm will converge to this global maxima.

Start with a reasonable starting value: α_0. Then iterate with

$$\alpha_{(k+1)} = \alpha_{(k)} - \frac{f'(\alpha_{(k)})}{f''(\alpha_{(k)})}$$

Polygamma functions are in `psi` function in the `pracma` R package.

What is a good starting value α_0? Well, we know that the mean of a Gamma$(\alpha, 1)$ is α, so a good starting value is $\alpha_0 = n^{-1}\sum_{i=1}^{n} X_i$.

Note the impact of the sample size of the original data. The Newton-Raphson algorithm converges to the MLE. If the data size is small, the MLE may not be close to the truth.

This is why we see that the blue and red lines are far from each other. However, when increase the observed data to be 1000 observations, the consistency of the MLE should kick in and we expect to see the blue and red lines to be similar (below).

■

Example 31 (Location Cauchy distribution)**.** Consider the location Cauchy distribution with mode at $\mu \in \mathbb{R}$. The goal is to find the MLE for μ.

$$f(x|\mu) = \frac{1}{\pi} \frac{1}{(1 + (x - \mu)^2)} \,.$$

First, we find the log-likelihood

$$L(\mu|X) = \prod_{i=1}^{n} f(X_i|\mu) = \pi^{-n} \prod_{i=1}^{n} \frac{1}{1 + (x_i - \mu)^2}$$

$$\Rightarrow l(\mu) := \log L(\mu|X) = -n \log \pi - \sum_{t=1}^{n} \log(1 + (X_i - \mu)^2) \,.$$

It is evident that a closed form solution is difficult. So, we find the derivates.

$$l'(\mu) = 2 \sum_{i=1}^{n} \frac{X_i - \mu}{1 + (X_i - \mu)^2} \,.$$

$$l''(\mu) = 2 \sum_{i=1}^{n} \left[2 \frac{(X_i - \mu)^2}{[1 + (X_i - \mu)^2]^2} - \frac{1}{1 + (X_i - \mu)^2} \right] ,$$

which may be positive or negative. So this is not a concave function. This implies that Newton-Raphson is not guaranteed to converge to the global maxima! It may not even converge to a local maxima, and may converge to a local minima. Thus, we will need to be careful in choosing starting values.

1. Set $\mu_0 = \text{Median}(X_i)$ since the mean of Cauchy does not exist and the Cauchy centered at μ is symmetric around μ.

2. Determine:
$$\mu_{(k+1)} = \mu_{(k)} - \frac{f'(\mu_{(k)})}{f''(\mu_{(k)})}$$

3. Stop when $|l'(\mu_{(k+1)}| < \epsilon$ for a chosen tolerance level ϵ.

■

Newton-Raphson in Higher Dimensions

The NR method can be found in the same way using the multivariate Taylor's expansion. Let $\theta = (\theta_1, \theta_2, \ldots, \theta_p)$. Then first let ∇f denote the gradient of f and $\nabla^2 f$ denote the Hessian. So

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial \theta_1} \\ \vdots \\ \frac{\partial f}{\partial \theta_p} \end{bmatrix} \quad \text{and} \quad \nabla^2 f = \begin{bmatrix} \frac{\partial^2 f}{\partial \theta_1^2} & \frac{\partial^2 f}{\partial \theta_1 \theta_2} & \cdots \\ \vdots & \vdots & \vdots \\ \frac{\partial^2 f}{\partial \theta_p \theta_1} & \cdots & \vdots \frac{\partial^2 f}{\partial \theta_p^2} \end{bmatrix}$$

Then, the function f is concave if $\nabla^2 f$ is negative definite. *So always check that first to know if there is a unique maximum.*

Using a similar multivariate Taylor series expansion, the Newton-Raphson update solves the system of linear equations

$$\nabla f(\theta_{(k)}) + \nabla^2 f(\theta_k)(\theta_{k+1} - \theta_k) = 0\,.$$

If $\nabla^2 f(\theta_{(k)})$ is invertible, then you have that

$$\theta_{(k+1)} = \theta_{(k)} - \left[\nabla^2 f(\theta_{(k)})\right]^{-1} \nabla f(\theta_{(k)})\,.$$

Iterations are stopped with when $\|\nabla f(\theta_{(k+1)})\| < \epsilon$ for some chosen tolerance level, ϵ.

Example 32 (Ridge regression)**.** In the ridge regression problem, we have an analytical solution available. But assume that we did not have the analytical solution. In that case we have our objective function to *minimize*:

$$Q(\beta) = \frac{(y - X\beta)^T (y - X\beta)}{2} + \frac{\lambda}{2}\beta^T\beta\,.$$

We are now interested in finding:

$$\hat{\beta} = \arg\min_{\beta} \left\{ \frac{(y - X\beta)^T (y - X\beta)}{2} + \frac{\lambda}{2}\beta^T\beta \right\}\,.$$

Recall that we obtained

$$\nabla Q(\beta) = (X^T X + \lambda I_p)\beta - X^T y$$

and the Hessian was

$$\nabla^2 Q(\beta) = (X^T X + \lambda I_p)$$

So the iterates of Newton-Raphson are then

$$\begin{aligned}
\beta_{k+1} &= \beta_{(k)} - (X^T X + \lambda I_p)^{-1}((X^T X + \lambda I_p)\beta_{(k)} - X^T y) \\
&= \beta_{(k)} - \beta_{(k)} + (X^T X + \lambda I_p)^{-1} X^T y \\
&= (X^T X + \lambda I_p)^{-1} X^T y
\end{aligned}$$

This simplification gives us the actual right solution! So one NR step will lead us to the analytical solution in this case! ■

Still working on this

Questions

1. Can you implement the the Newton-Raphson procedure for linear regression and ridge regression?
2. What are some of the issues in implementing Newton-Raphson? Can we use it for any problem?
3. If the function is not concave and different starting values yield convergence to different points (or divergence), then what do we do?

7.2 Gradient Ascent (Descent)

For concave objective functions, Newton-Raphson is essentially the best algorithm. However, there are a few flaws of the algorithm that make it challenging to use it more generally:

- when the objective function is not concave, NR is not guaranteed to even converge.

- when the objective function is complicated and high-dimensional, finding the Hessian, and inverting it repeatedly may be expensive.

In such a case, gradient ascent (or gradient descent if the problem is a minimizing problem) is a useful alternative as it does not require the Hessian.

Consider the objective function $f(\theta)$ that we want to maximize and suppose θ_* is the true maximum. Then, by the Taylor series approximation at a fixed θ_0

$$f(\theta) \approx f(\theta_0) + f'(\theta_0)(\theta - \theta_0) + \frac{f''(\theta_0)}{2}(\theta - \theta_0)^2$$

If $f''(\theta)$ is unavailable or we don't want to use it, consider assuming that the double derivative is a negative constant:$f''(\theta) = -1/t$ for some $t > 0$. That is, assume that f is quadratic and concave. Then,

$$f(\theta) \approx f(\theta_0) + f'(\theta_0)(\theta - \theta_0) - \frac{1}{2t}(\theta - \theta_0)^2$$

Maximizing $f(\theta)$ and using this crude approximation would imply maximizing the right hand side. Taking the derivative with respect to θ setting it to zero:

$$f'(\theta_0) - \frac{\theta - \theta_0}{t} \overset{set}{=} 0 \Rightarrow \theta = \theta_0 + tf'(\theta_0)\,.$$

Using this intuition, given a $\theta_{(k)}$, the gradient ascent algorithm does the update

$$\theta_{(k+1)} = \theta_{(k)} + tf'(\theta_{(k)})\,,$$

for $t > 0$. The iteration can be stopped when $|\theta_{(k+1)} - \theta_{(k)}| < \epsilon$ for $\epsilon > 0$ or when $|f'(\theta_{k+1})| \approx 0$.

For concave functions, there exists a t such that gradient ascent converges to the global maxima. In general (when the function is not concave), there exists a t such that gradient ascent converges to a local maxima, as long as you don't start from a local minima.

The algorithm essentially does a local concave quadratic approximation at the current point $\theta_{(k)}$ and then maximizes that quadratic equation. The value of t indicates how far do we want to jump and is a tuning parameter. If t is large, we take big jumps, as opposed to t small, where the jumps are smaller.

Example 33 (Location Cauchy distribution)**.** Recall the location Cauchy distribution with mode at $\mu \in \mathbb{R}$, where the log-likelihood is not guaranteed to be concave and thus Newton-Raphson is difficult to use. The log-likelihood was

$$L(\mu|X) = \prod_{i=1}^{n} f(X_i|\mu) = \pi^{-n} \prod_{i=1}^{n} \frac{1}{1 + (x_i - \mu)^2}$$

$$\Rightarrow l(\mu) := \log L(\mu|X) = -n \log \pi - \sum_{t=1}^{n} \log(1 + (X_i - \mu)^2)\,.$$

We found the first derivative:

$$l'(\mu) = 2 \sum_{i=1}^{n} \frac{X_i - \mu}{1 + (X_i - \mu)^2}\,.$$

I choose $t = 0.3$ in this case so that we obtain the following gradient ascent iterative scheme:

$$\mu_{(k+1)} = \mu_{(k)} + (0.3) \left(2 \sum_{i=1}^{n} \frac{X_i - \mu}{1 + (X_i - \mu)^2}\,. \right)\,.$$

1. Set $\mu_0 = \text{Median}(X_i)$ since the mean of Cauchy does not exist and the Cauchy centered at μ is symmetric around μ.

2. Determine:
$$\mu_{(k+1)} = \mu_{(k)} + (0.3) \left(2 \sum_{i=1}^{n} \frac{X_i - \mu}{1 + (X_i - \mu)^2}\,. \right)\,.$$

3. Stop when $|l'(\mu_{(k+1)})| < \epsilon$ for a chosen tolerance level ϵ.

■

Gradient Ascent in Higher Dimensions

By a multivariate Taylor series expansion, we can obtain a similar motivation and the iteration in the algorithm is:

$$\theta_{(k+1)} = \theta_{(k)} + t \nabla f(\theta_{(k)})\,.$$

Example 34 (Logistic regression)**.** We have studied linear regression for modeling continuous responses. But when Y is a response of 1s and 0s (Bernoulli) then assuming the Ys are normally distributed is not appropriate. Instead, when the ith covariate is $(x_{i1}, \dots, x_{ip})^T$, then for $\beta \in \mathbb{R}^p$ logistic regression assumes the model

$$Y_i \sim \text{Bern}\left(\frac{e^{x_i^T\beta}}{1 + e^{x_i^T\beta}}\right).$$

In other words, the probability that any response takes the value 1 is

$$\Pr(Y_i = 1) = \frac{e^{x_i^T\beta}}{1 + e^{x_i^T\beta}} =: p_i\,.$$

Our goal is to obtain the MLE of β. As usual, first we write down the log-likelihood.

$$\begin{aligned}
L(\beta|Y) &= \prod_{i=1}^{n} (p_i)^{y_i} (1 - p_i)^{1-y_i} \\
\Rightarrow l(\beta) &= \sum_{i=1}^{n} y_i \log p_i + \sum_{i=1}^{n} (1 - y_i)\log(1 - p_i) \\
&= \sum_{i=1}^{n} \log(1 - p_i) + \sum_{i=1}^{n} y_i \left(\log p_i - \log(1 - p_i)\right) \\
&= -\sum_{i=1}^{n} \log\left(1 + \exp(x_i^T\beta)\right) + \sum_{i=1}^{n} y_i x_i^T\beta
\end{aligned}$$

In order to understand taking the derivative, we can do is elementwise for each β_s or do it using matrix calculus. Let's first do it element-wise. In order to do it element-wise, we can first solve the above

$$l(\beta) = -\sum_{i=1}^{n} \log\left(1 + \exp\left(\sum_j x_{ij}\beta_j\right)\right) + \sum_{i=1}^{n}\sum_{j=1}^{p} y_i x_{ij}\beta_j$$

$$\frac{\partial l(\beta)}{\beta_s} = \sum_{i=1}^{n} y_i x_{is} - \sum_{i=1}^{n} \frac{x_{is} e^{\sum_j x_{ij}\beta_j}}{1 + e^{\sum_j x_{ij}\beta_j}}$$

Assembling the whole vector for all j

$$\Rightarrow \nabla l(\beta) = \sum_{i=1}^{n} x_i \left[y_i - \frac{1}{1 + e^{-x_i^T\beta}}\right]$$

The above can be done directly for the full vector as well in one go:

$$\begin{aligned}\nabla l(\beta) &= \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} \frac{x_i e^{x_i^T\beta}}{1+e^{x_i^T\beta}} \\ &= \sum_{i=1}^{n} x_i \left[y_i - \frac{e^{x_i^T\beta}}{1+e^{x_i^T\beta}} \right] \\ &= \sum_{i=1}^{n} x_i \left[y_i - \frac{1}{1+e^{-x_i^T\beta}} \right] \overset{set}{=} 0\end{aligned}$$

An analytical solution here is not possible, thus a numerical optimization tool is required. In order to know what kind of function we have, we also obtain the Hessian. In order to calculate the Hessian, we can, once again do this element-wise or do it together. Let's do it elementwise: For $j \neq k$

$$\begin{aligned}\frac{\partial^2 l(\beta)}{\beta_k \beta_s} &= \frac{\partial}{\partial \beta_k} \left[\sum_{i=1}^{n} y_i x_{is} - \sum_{i=1}^{n} \frac{x_{is} e^{\sum_j x_{ij}\beta_j}}{1+e^{\sum_j x_{ij}\beta_j}} \right] \\ &= -\frac{\partial}{\partial \beta_k} \left[\sum_{i=1}^{n} \frac{x_{is} e^{\sum_j x_{ij}\beta_j}}{1+e^{\sum_j x_{ij}\beta_j}} \right] \\ &= -\sum_{i=1}^{n} x_{is} e^{\sum_j x_{ij}\beta_j} \frac{x_{ik}}{(1+e^{\sum_j x_{ij}\beta_j})^2} \\ &= -\sum_{i=1}^{n} x_{ik} x_{is} \frac{e^{\sum_j x_{ij}\beta_j}}{(1+e^{\sum_j x_{ij}\beta_j})^2} \\ &= -X^T W X\,,\end{aligned}$$

where W is the $n \times n$ diagonal matrix with diagonal elements $e^{x_i^T\beta}/(1+e^{x_i^T\beta})^2$.

Note: You can verify by studying the Hessian above that it is negative semi-dfinite, and thus that the likelihood function is indeed concave and thus we can use either Newton-Raphson or Gradient Ascent successfully. We will use gradient ascent, and you should on your own, implement N-R.

Gradient Ascent for Logistic regression:

1. Set $\beta_{(0)} = \mathbf{0}_p$ (since there is no information available, and it is reasonable to assume that none of the covariates are important).
2. Set some appropriate t.

3. For iteration $k+1$:

$$\beta_{(k+1)} = \beta_{(k)} + t\left[\sum_{i=1}^{n} x_i \left[y_i - \frac{1}{1+e^{-x_i^T\beta_{(k)}}}\right]\right]$$

4. Stop when $|\nabla f'(\beta_{(k)}| < \epsilon$.

$\blacksquare$

Example 35 (Probit Regression)**.** Not completed in class, do it yourself. Logistic regression is only one way to model the probabilities. For p_i, we may actually use any mapping that will transport $x_i^T\beta$ to a probability p_i. This can be done through a CDF function.

For $i = 1, \dots, n$, let $x_i = (1, x_{i2}, \dots, x_{ip})^T$ be the vector of covariates for the ith observation and $\beta \in \mathbb{R}^p$ be the corresponding vector of regression coefficients. Suppose response y_i is a realization of Y_i with

$$Y_i \sim \text{Bern}\left(\Phi(x_i^T\beta)\right),$$

where $\Phi(\cdot)$ is the CDF of a standard Normal distribution. We want to find the MLE of β. The likelihood is given by

$$L(\beta) = \prod_{i=1}^{N} \Phi(x_i^T\beta)^{y_i}(1-\Phi(x_i^T\beta))^{(1-y_i)}.$$

We know that $1-\Phi(x) = \Phi(-x)$. Let $z_i = 2y_i - 1$, which implies $y_i = 1 \implies z_i = 1$ and $y_i = 0 \implies z_i = -1$. This transformation will help us.

So for $y_i = 1, \Phi(x_i^T\beta)^{y_i} = \Phi(z_i x_i^T\beta)^{y_i}$, and for $y_i = 0, \Phi(x_i^T\beta)^{y_i} = 1 = \Phi(z_i x_i^T\beta)^{y_i}$. Similarly, for $y_i = 0, \Phi(-x_i^T\beta)^{(1-y_i)} = \Phi(z_i x_i^T\beta)^{(1-y_i)}$, and for $y_i = 1, \Phi(-x_i^T\beta)^{(1-y_i)} = 1 = \Phi(z_i x_i^T\beta)^{(1-y_i)}$.

Therefore,

$$\begin{aligned}\Phi(x_i^T\beta)^{y_i} &= \Phi(z_i x_i^T\beta)^{y_i}\\ (1-\Phi(x_i^T\beta))^{(1-y_i)} &= \Phi(z_i x_i^T\beta)^{(1-y_i)}\end{aligned}$$

As a consequence, likelihood can be written concisely as

$$\begin{aligned} L(\beta) &= \prod_{i=1}^{N} \Phi(z_i x_i^T \beta)^{y_i} \Phi(z_i x_i^T \beta)^{(1-y_i)} \\ &= \prod_{i=1}^{N} \Phi(z_i x_i^T \beta) \end{aligned}$$

Let the log likelihood be denoted by $l(\beta)$. We have

$$l(\beta) = \sum_{i=1}^{N} \log(\Phi(z_i x_i^T \beta)) \,.$$

For finding the optimum value of regression coefficient that minimizes the log-likelihood, we use the iterative Newton-Raphson method. The gradient and Hessian calculations are as follows:

$$\begin{aligned} \nabla l(\beta) = D &= \frac{d}{d\beta} \sum_{i=1}^{N} \log(\Phi(z_i x_i^T \beta)) \\ &= \sum_{i=1}^{N} \frac{f(z_i x_i^T \beta) z_i x_i}{\Phi(z_i x_i^T \beta)} \,. \end{aligned}$$

$$\begin{aligned} \nabla^2 l(\beta) = DD &= \frac{d}{d\beta} \sum_{i=1}^{N} \frac{f(z_i x_i^T \beta) z_i x_i}{\Phi(z_i x_i^T \beta)} \\ DD_{pq} &= \sum_{i=1}^{N} \left[\frac{d}{d\beta_q} \frac{f(z_i x_i^T \beta) z_i x_{ip}}{\Phi(z_i x_i^T \beta)} \right] \\ &= \sum_{i=1}^{N} \left[-\frac{f(z_i x_i^T \beta)^2}{\Phi(z_i x_i^T \beta)^2} (z_i x_{iq})(z_i x_{ip}) + \frac{f'(z_i x_i^T \beta)}{\Phi(z_i x_i^T \beta)} (z_i x_{iq})(z_i x_{ip}) \right] \end{aligned}$$

We can use that $f'(x) = \frac{-x}{\sqrt{2\pi}} e^{-x^2/2} = -x f(x)$. This gives us the following formulation for Hessian matrix

$$DD_{pq} = -\sum_{i=1}^{N} \left[x_{ip} \left(\frac{f(z_i x_i^T \beta)}{\Phi(z_i x_i^T \beta)} \right)^2 x_{iq} + x_{ip} \left(\frac{f(z_i x_i^T \beta)(z_i x_i^T \beta)}{\Phi(z_i x_i^T \beta)} \right) x_{iq} \right]$$

Therefore,

$$DD = -\sum_{i=1}^{N} x_i \left[\left(\frac{f(z_i x_i^T \beta)}{F(z_i x_i^T \beta)} \right)^2 + \left(\frac{f(z_i x_i^T \beta)(z_i x_i^T \beta)}{\Phi(z_i x_i^T \beta)} \right) \right] x_i^T$$

One can show that the above is negative definite, and thus the function is concave (this will be an exercise). Now we can apply Gradient Ascent algorithm or Newton-Raphson. ■

Still working on this

7.3 MM Algorithm

Consider obtaining a solution to

$$\theta_* = \arg\max_\theta f(\theta)$$

The "Minorize/Maximize algorithm" algorithm at a current iterate, finds a "minorizing" function at that point, and then maximizes that minorizing function. That is, at any given iteration, consider a *minorizing function* $\tilde{f}(\theta|\theta_{(k)})$ such that:

- $f(\theta_k) = \tilde{f}(\theta_k|\theta_k)$
- $f(\theta) \geq \tilde{f}(\theta|\theta_k)$ for all other θ

Then, $\theta_{(k+1)}$ is obtained as

$$\theta_{(k+1)} = \arg\max_\theta \tilde{f}(\theta|\theta_{(k)}) \, .$$

The algorithm has the ascent property in that every update increases the objective value. That is,

$$\begin{aligned} f(\theta_{(k+1)}) &\geq \tilde{f}(\theta_{(k+1)} \mid \theta_{(k)}) \\ &\geq \tilde{f}(\theta_{(k)} \mid \theta_{(k)}) \\ &= f(\theta_{(k)}) \, . \end{aligned}$$

Thus, if we want to maximize $f(\theta)$, we may find a minorizing function for it, and then repeatedly maximize it. The key to implementing the MM algorithm is to finding a good *minorizing* function. This can be done in a few different ways and generally application specific.

Note: When minimizing an objective function, we the opposite: we find a *majorizing function* and then minimize it.

The question is, how to construct such minorizing functions? This is typically done on a case-by-case basis. Many inequalities are used:

- Jensen's inequality
- Cauchy Schwartz inequliaty
- Arithmetic mean-geometric mean inequalities.

One common way of implementing MM-algorithm is to use the remainder form of Taylor series expansion:

$$f(\theta) = f(\theta_{(k)}) + f'(\theta_{(k)})(\theta - \theta_{(k)}) + \frac{1}{2}f''(z)(\theta - \theta_k)^2 \,,$$

where z is some constant between θ_k and θ (by the mean value theorem).

If we can lower bound $f''(z) > L$, then

$$\tilde{f}(\theta|\theta_{(k)}) = f(\theta_{(k)}) + f'(\theta_{(k)})(\theta - \theta_{(k)}) + \frac{1}{2}L(\theta - \theta_k)^2$$

and the iterates are

$$\theta_{(k+1)} = \theta_{(k)} - \frac{f'(\theta_{(k)})}{L} \,.$$

In some way, this is an informed way of choosing the learning rate for gradient ascent!

Example 36 (Location Cauchy distribution)**.** As before, the log-likelihood is

$$f(\mu) = \log L(\mu|X) = -n\log\pi - \sum_{t=1}^{n}\log(1 + (X_i - \mu)^2)\,,$$

and derivatives

$$f'(\mu) = 2\sum_{i=1}^{n}\frac{X_i - \mu}{1 + (X_i - \mu)^2}\,,$$

$$f''(\mu) = 2\sum_{i=1}^{n}\left[\frac{(X_i - \mu)^2 - 1}{[1 + (X_i - \mu)^2]^2}\right]\,.$$

Consider the Taylor's expansion:

$$f(\mu) = f(\mu_{(k)}) + f'(\mu_{(k)})(\mu - \mu_{(k)}) + \frac{1}{2}f''(\mu_{(k)})(\mu - \mu_{(k)})^2$$

Note that for any i

$$\begin{aligned} 2\left[\frac{(X_i - \mu)^2 - 1}{[1 + (X_i - \mu)^2]^2}\right] &\geq 2\left[\frac{-1}{[1 + (X_i - \mu)^2]^2}\right] \\ &\geq -2 := L\,. \end{aligned}$$

This implies that $f''(\mu) \geq -2n$. So the iterations are,

$$\mu_{(k+1)} = \mu_{(k)} + \frac{1}{n}\sum_{i=1}^{n} \frac{X_i - \mu_{(k)}}{1 + (X_i - \mu_{(k)})^2}\,.$$

■

The main utility of MM algorithm is that it can be used when the gradient of the objective function is unavailable. An excellent example of this is the following Bridge Regression problem.

Example 37 (Bridge regression)**.** Recall the case of the penalized (negative) log-likelihood problem during ridge regression, where the objective function was:

$$Q(\beta) = \frac{(y - X\beta)^T(y - X\beta)}{2} + \frac{\lambda}{2}\sum_{i=1}^{p} \beta_i^2\,.$$

By changing the penalty function, the above ridge regression objective function can be generalized as *bridge regression* when the objective function is

$$Q_B(\beta) = \frac{(y - X\beta)^T(y - X\beta)}{2} + \frac{\lambda}{\alpha}\sum_{i=1}^{p} |\beta_i|^\alpha\,,$$

for $\alpha \in [1, 2]$ and $\lambda > 0$. When $\alpha = 2$, this is ridge regression, and when $\alpha = 1$, this is the popular *lasso* regression. Different choices of α, lead to different style of penalization. For a given λ, smaller values of α push the estimates closer towards zero.

We need to find the *bridge regression estimates*

$$\arg\min_{\beta} Q_B(\beta).$$

First note that, $(y - X\beta)^T(y - X\beta)$ is a convex function and $|\beta_i|^\alpha$ is convex for $\alpha \geq 1$. Since the sum of positive convex functions is convex, the objective function is convex, thus our optimization algorithms will find a global *minima*.

Note that for $\alpha = 1$, the objective function is not differentiable at 0, and for $\alpha \in (1, 2)$, the function is not twice differentiable at 0. Thus, using Newton-Raphson and gradient descent is not possible. We will instead use an MM algorithm. Since this is a minimization problem, we will find a *majorizing function* and then minimize the majorizing function.

We will try to find a majorizing function that upper bounds the objective $Q_B(\beta)$, and then minimize the majorizing function. Intuitively, optimizing the majorizing function will again require derivatives of the majorizing function. Thus our goal is to find a majorizing function that *does not* contain an absolute value, and is thus differentiable.

Note further that $(y - X\beta)^T(y - X\beta)$ term is well behaved and quadratic. Thus, we are not inclined to look at this part. Instead we would like to lower bound $\sum_{i=1}^p |\beta_i|^\alpha$. Consider a function $h(u) = u^{\alpha/2}$ for $u \geq 0$ and see that

$$h'(u) = \frac{\alpha}{2} u^{\alpha/2 - 1}$$

and

$$h''(u) = \frac{\alpha}{2}\left(\frac{\alpha}{2} - 1\right) u^{\alpha/2-2} \leq 0$$

so $h(u)$ is a concave function for $\alpha \in [1, 2]$. For a concave function, by the "Rooftop theorem", the first order Taylor series creates a tangent line that is above the function. Thus for a u^*,

$$h(u) \leq h(u^*) + h'(u^*)(u - u^*) = h(u^*) + \frac{\alpha}{2}(u^*)^{\alpha/2-1}(u - u^*)\,.$$

For any given iteration of the optimization, given $\beta_{(k)}$, taking $u = |\beta_i|^2$ and $u^* = |\beta_{i,(k)}|^2$ where β_i is the ith component of the vector β. Then,

$$\begin{aligned} h(u) = |\beta_i|^\alpha &\leq |\beta_{i,(k)}|^\alpha + \frac{\alpha}{2}|\beta_{i,(k)}|^{\alpha-2}\left(\beta_i^2 - \beta_{i,(k)}^2\right) \\ &= |\beta_{i,(k)}|^\alpha - \frac{\alpha}{2}|\beta_{i,(k)}|^\alpha + \frac{\alpha}{2}|\beta_{i,(k)}|^{\alpha-2}\beta_i^2 \end{aligned}$$

$$= \text{constants} + \frac{m_{i,(k)}}{2}\beta_i^2$$

where $m_{i,(k)} = \alpha|\beta_{i,(k)}|^{\alpha-2}$. (You will see that the constants will not be important.)

Now that we have upper bounded the the penalty function, we have an upper bound on the full objective function! So, the objective function can be bounded above by:

$$Q_B(\beta) \leq \text{constants} + \frac{(y - X\beta)^T(y - X\beta)}{2} + \frac{\lambda}{2\alpha}\sum_{j=1}^{p} m_{j,(k)}\beta_j^2 \,.$$

Why is this upper bound useful?

- Remember that at any given iteration, the optimization is with respect to β. Thus, the constants are truly constants.
- The upper bound has no absolute values and is easily differentiable!
- Recall that we obtained the upper bound function using a derivative of $h(u)$. This derivative is not defined at $u = 0$. However, we're only using the derivative function at $u^* = |\beta_{i,(k)}|^2$, which is the previous iteration. So as long as we DO NOT START at zero, this upper bound is valid.
- Finally, the upper bound is easily optimizable, as it is similar to ridge. (See below)

The objective function is similar to ridge regression, except it is "weighted". Following the same steps as in ridge optimization, you can show that the minimum occurs at

$$\beta_{(k+1)} = (X^TX + \lambda M_{(k)})^{-1}X^Ty \,,$$

where $M_{(k)} = \text{diag}(m_{1,(k)}/\alpha, m_{2,(k)}/\alpha, \ldots, m_{p,(k)}/\alpha)$. Note that here $M_{(k)}$ is what drives the direction of the optimization.

■

Questions to think about

- How do you think we can choose α in any given problem?
- How do you think we can choose λ in any given problem?
- Will the MM algorithm always converge to a global maxima?

7.4 Exercises

1. Find the MLE of (α, μ) for a Pareto distribution with density

$$f(x) = \frac{\alpha\mu^\alpha}{x^{\alpha+1}} \quad x \geq \mu, \quad \mu, \alpha > 0\,.$$

   Do you need numerical procedures to calculate the MLE here?

2. Consider objective function of the form $f(\theta) = a\theta^2 + b\theta + c$. Writes the iterates of the Newton-Raphson algorithm. In how many iterates do you expect NR to converge for any starting value $\theta_{(0)}$?

3. (Using R) Find the MLE of (μ, σ^2) for the $N(\mu, \sigma^2)$ distribution using Newton-Raphson's method. Compare with the closed form estimates.

4. (Using R) Using both Newton-Raphson and gradient ascent algorithm, maximize objective function

$$f(x) = \cos(x) \quad x \in [-\pi, 3\pi]\,.$$

5. Consider estimating the first moment of Exp(λ) using simple importance sampling with a Gamma(α, β) proposal distribution. The density of an exponential is

$$\pi(x) = \lambda e^{-\lambda x} \quad x > 0\,,$$

   and the density of Gamma(α, β) is

$$g(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{\beta x} \quad x > 0\,.$$

   (a) Construct the simple importance sampling expression for general α, β, and λ?

   (b) Denote the variance of the estimator in (a) as $\kappa_\lambda(\alpha, \beta)$. For what values of α and β is $\kappa_\lambda(\alpha, \beta)$ infinite? You will have have to solve the integral here.

   (c) Set $\beta = 1$. Describe an algorithm for obtaining the best proposal within this family of proposals. Give all details. Use the above constructed algorithm to obtain the best Gamma(α, 1) for $\lambda = 5$. Here you may require an optimization algorithm.

   (d) Is the proposal obtained in (d) the universally optimal proposal for this problem?

6. Generate data according to the logistic regression model above with $n = 50$, and use Newton-Raphson's and gradient ascent algorithm to find the MLE.

7. Implement Newton-Raphson and gradient ascent to find the MLE of (α, β) for a Beta(α, β) distribution. Generate your own data to implement this in R.

8. (Using R) Find the MLE of a $\Gamma(\alpha, 1)$ distribution using Newton-Raphson's method. Set $\alpha = 4$ and $n = 10$ and generate your own data. Rerun the Newton-Raphson's algorithm with different starting values.

9. (Using R) Find the MLE of a location Cauchy distribution with density

$$f(x) = \frac{1}{\pi}\frac{1}{1 + (x-\mu)^2} \quad \mu \in \mathbb{R}, x \in \mathbb{R}\,.$$

   Set $\mu = -2$ and $n = 4, 100$, and run the Newton-Raphson's algorithm with different starting values. Are starting values more impactful here than compared to the Gamma problem? Why or why not? Now repeat the same for the gradient ascent algorithm.

10. (Modified Newton-Raphson): It is possible to "overshoot" when using Newton-Raphson's algorithm, when the objective function is not concave. In these scenarios, we use a modified Newton-Raphson's approach, with step factors.

    Suppose the kth iteration is such that $f'(\theta_{(k)}) > 0$ (that is the function is increasing at $\theta_{(k)}$), and NR takes us to $\theta_{(k+1)}$ where $f'(\theta_{(k+1)}) < 0$, so that now the function is decreasing. This means we may have overshot! As a compromise, we may want to implement the following algorithm:

$$\theta_{(k+1)} = \theta_{(k)} - \lambda_{(k)}\,\frac{f'(\theta_{(k)})}{f''(\theta_{(k)})}$$

    where $\lambda_{(k)}$ is a step-factor sequence chosen at every iteration so that

$$f(\theta_{(k+1)}) > f(\theta_{(k)})\,.$$

    That is, the next value must be such that we achieve an increase in the objective function. You can choose $\lambda_{(k)}$ so that

    (a) If $f(\theta_{(k+1)}) > f(\theta_k)$, then $\lambda_{(k)} = 1$. Else $\lambda_{(k)} = 1/2$, and recalculate $f(\theta_{(k+1)})$

    (b) If $f(\theta_{(k+1)}) > f(\theta_k)$, then continue, else set $\lambda_{(k)} = 1/2^2$ and so on...

Implement this modified algorithm for the Gamma and Cauchy examples.

11. Consider the logistic regression model: for $i = 1, \dots, n$, let $x_i = (1, x_{i2}, \dots, x_{ip})^T$ be the vector of covariates for the ith observation and $\beta \in \mathbb{R}^p$ be the corresponding vector of regression coefficients. Suppose response y_i is a realization of Y_i with

$$Y_i \sim \text{Bern}\,(p_i) \quad \text{where} \quad p_i = \frac{\exp(-x_i^T\beta)}{1 + \exp(-x_i^T\beta)}\,.$$

   (a) Write the negative log-likelihood, $-l(\beta)$.

   (b) Consider the ridge logistic regression problem, which *minimizes* the following penalized negative log-likelihood:

$$Q(\beta) = -l(\beta) + \frac{\lambda}{2}\sum_{i=1}^{p}\beta_i^2\,.$$

   Is $Q(\beta)$ a convex function in β?

   (c) Write the Newton-Raphson algorithm for minimizing $Q(\beta)$. Write all steps clearly.

12. Consider a typical Poisson regression model. For $i = 1, \dots, n$, let $x_i \in \mathbb{R}^p$ be the vector of covariates for the ith observation and $\beta \in \mathbb{R}^p$ are the corresponding regression coefficients. Let y_i be the observed response, which is count data. That is,

$$Y_i \sim \text{Poisson}\left(e^{x_i^T\beta}\right)\,.$$

   (a) Find the joint likelihood $L(\beta \mid y_1, \dots y_n)$ and obtain the maximum likelihood estimator of β. If not available in closed-form, present the complete optimization algorithm to obtain $\hat{\beta}_{\text{MLE}}$.

8 The EM algorithm

An important application of the MM algorithm is the Expectation-Maximization (EM) algorithm. Since the EM algorithm is an integral part of statistics on its own, we study it separately. We will first motivate the EM algorithm with an example.

8.1 Motivating example: Gaussian mixture models

Suppose $X_1, X_2, \dots, X_n \overset{iid}{\sim} F$, where F is mixture of two normal distributions so that the density is:

$$f(x \mid \mu_1, \mu_2, \sigma_1^2, \sigma_2^2, p^*) = p^* f_1(x \mid \mu_1, \sigma_1^2) + (1 - p^*) f_2(x \mid \mu_2, \sigma_2^2)\,,$$

where $f_i(x \mid \mu_i, \sigma_i^2)$ is the density of $N(\mu_i, \sigma_i^2)$ distribution for $i = 1, 2$. Data following such distributions arises often in real life. For instance, suppose, we collect batting averages for players in cricket. Then bowlers are expected to have low averages and batters are expected to have high averages, created a mixture-like distribution.

Given data, suppose we wish to find the maximum likelihood estimates of all 5 parameters: $(\mu_1, \mu_2, \sigma_1^2, \sigma_2^2, p^*)$. That is, we want to maximize:

$$\begin{aligned} l(\mu_1, \mu_2, \sigma_1^2, \sigma_2^2, p^* | X) &= \sum_{i=1}^{n} \log f(x_i \mid \mu_1, \mu_2, \sigma_1^2, \sigma_2^2, p^*) \\ &= \sum_{i=1}^{n} \log \left[p^* f_1(x \mid \mu_1, \sigma_1^2) + (1 - p^*) f_2(x \mid \mu_2, \sigma_2^2) \right]\,. \end{aligned}$$

There is no analytical solution to the above optimization problem and we have to resort to numerical techniques. We can certainly think of implementing gradient-based estimation methods here. However, instead of trying to use gradient-based tools, we use a common trick called the <u>*latent variable*</u> or <u>*missing data*</u> trick.

Recall that the data likelihood is a mixture of Gaussians. An interpretation of this is that with probability p^*, any observed X_i is from f_1 and with probability $1 - p^*$ it is from f_2.

Suppose we have the information about the the class of each x_i (classes being class 1 and class 2). Thus, suppose the *complete data* was of the form

$$(X_1, Z_1), (X_2, Z_2), \dots, (X_n, Z_n)\,,$$

where each $Z_i = k$ means that X_i is from population k. If this complete data is available to us, then first note that the joint probability density/mass function is

$$f(x_i, z_i = k) = f(x_i | z_i = k) \Pr(Z_i = k) \, .$$

Suppose $\mathcal{D}_1 = \{i : 1 \leq i \leq n, z_i = 1\}$ and $\mathcal{D}_2 = \{i : 1 \leq i \leq n, z_i = 2\}$, with cardinality d_1 and d_2 respectively. The set $\mathcal{D}_1$ and $\mathcal{D}_2$ have the indices of the data that belong to each component of the mixture.

Then the likelihood of the $\underline{\text{complete data}}$ is

$$\begin{aligned}
&L(\mu_1, \mu_2, \sigma_1^2, \sigma_2^2, p^* | X, Z) \\
&= \prod_{i=1}^{n} f(x_i, z_i) \\
&= \prod_{i \in \mathcal{D}_1} f(x_i, z_i = 1) \prod_{j \in \mathcal{D}_2} f(x_i, z_i = 2) \\
&= \prod_{i \in \mathcal{D}_1} f(x_i | z_i = 1) \Pr(Z_i = 1) \prod_{i \in \mathcal{D}_2} f(x_i | z_i = 2) \Pr(Z_i = 2) \\
&= \prod_{i \in \mathcal{D}_1} \left[p^* f_1(x_i | \mu_1, \sigma_1^2) \right] \prod_{j \in \mathcal{D}_2} \left[f_2(x_i | \mu_2, \sigma_2^2)(1 - p^*) \right] \\
&= (p^*)^{d_1} (1 - p^*)^{d_2} \prod_{i \in \mathcal{D}_1} \left[f_1(x_i | \mu_1, \sigma_1^2) \right] \prod_{i \in \mathcal{D}_2} \left[f_2(x_i | \mu_2, \sigma_2^2) \right] \, .
\end{aligned}$$

This means that the log-likelihood of the $\underline{\text{complete data}}$ is

$$\Rightarrow \log L = d_1 \log(p^*) + d_2 \log(1 - p^*) + \sum_{i \in \mathcal{D}_1} \log f_1(x_i | \mu_1, \sigma_1^2) + \sum_{i \in \mathcal{D}_2} \log f_2(x_i | \mu_2, \sigma_2^2) \, .$$

This *complete* log-likelihood is in a far nicer format, so that closed-form estimates are available.

Differentiating with respect to p^*, we get

$$\frac{\partial l}{\partial p^*} = \frac{d_1}{p^*} - \frac{d_2}{1 - p^*} \overset{\text{set}}{=} 0 \, .$$

This gives that the MLE for p^* is

$$\hat{\pi}^* = \frac{d_1}{d_1 + d_2} = \frac{d_1}{n} \, .$$

You can also see that the MLEs for $\mu_1, \mu_2, \sigma_1^2, \sigma_2^2$ have all been isolated so that the MLEs can be easily obtained from the usual Gaussian likelihood.

$$\hat{\mu}_1 = \frac{1}{d_1} \sum_{i \in \mathcal{D}_1} X_i \quad , \quad \hat{\mu}_2 = \frac{1}{d_2} \sum_{i \in \mathcal{D}_2} X_i$$

$$\sigma_1^2 = \frac{1}{d_1} \sum_{i \in \mathcal{D}_1} (X_i - \hat{\mu}_1)^2 \quad , \quad \sigma_2^2 = \frac{1}{d_2} \sum_{i \in \mathcal{D}_2} (X_i - \hat{\mu}_2)^2 \, .$$

(Of course, you must verify the second derivative condition).

However, remember that the actual Zs are **unknown** to us, so we can't actually calculate d_1 and d_2, and the above cannot be evaluated. The EM algorithm will solve this problem by estimating the <u>unobserved</u> Z_i corresponding to each Z_i in an iterative manner.

We will come back to this Gaussian problem again.

8.2 The Expectation-Maximization Algorithm

Suppose, we have a vector of parameters θ, and we have only observed the marginal data $X_1, \dots, X_n$ from the complete data (X_i, Z_i). The objective is to maximize is

$$l(\theta|X) = \log f(\mathbf{x}|\theta) = \log \int f(\mathbf{x}, \mathbf{z}|\theta) d\nu_z \, ,$$

where the $\int \cdot \, d\nu_z$ denotes integral or summation based on whether Z is continuous or discrete.

The EM algorithm produces iterates $\{\theta_{(k)}\}$ in order to solve the above maximization problem. Writing the target objective function in this form allows us to do the optimization using EM. The EM algorithm iterates through an "E" step (Expectation) and an "M" step (maximization). Consider a starting value θ_0. Then for any $(k+1)$ iteration

1. **E-Step**: Compute the expectation of the complete expected log-likelihood:

$$q(\theta|\theta_{(k)}) = \mathrm{E}_{Z|X}\Big[\log f(\mathbf{x}, \mathbf{z}|\theta) \mid X = x, \theta_{(k)} \Big]$$

   where the expectation is computed with respect to the conditional distribution of Z given $X = x$ for the current iterate $\theta_{(k)}$.

2. **M-Step**: Compute
$$\theta_{(k+1)} = \arg\max_{\theta \in \Theta} q\left(\theta|\theta_{(k)}\right) .$$

3. Stop when $\|\theta_{(k+1)} - \theta_{(k)}\| < \epsilon$.

The following theorem will convince us that running the above algorithm will ensure the $\theta_{(k)}$ converges to a local maxima. The trick to implementing the EM algorithm for a general problem is in finding an appropriate joint distribution (X, Z) for which the E-step is computable.

Theorem 10. The EM algorithm is an MM algorithm and thus has the ascent property.

Proof. In order to show the result we first need to find a minorizing function.

The objective function is $\log f(\mathbf{x}|\theta)$. So we need to find $\tilde{f}(\theta|\theta_{(k)})$ such that $\tilde{f}(\theta_{(k)}|\theta_{(k)}) = \log f(\mathbf{x}|\theta_{(k)})$ and in general
$$\tilde{f}(\theta|\theta_{(k)}) \leq \log f(\mathbf{x}|\theta)$$

We will show that $\tilde{f}(\theta|\theta_{(k)})$ is such that
$$\tilde{f}(\theta|\theta_{(k)}) = q(\theta|\theta_{(k)}) \; + \text{constants}.$$

Then, maximizing $\tilde{f}(\theta|\theta_{(k)})$ is equivalent to maximizing $q(\theta|\theta_{(k)})$ (the M step of both EM and MM).

Let
$$\tilde{f}(\theta|\theta_{(k)}) = \int_z \log\{f(\mathbf{x},\mathbf{z}|\theta)\} f(\mathbf{z}|\mathbf{x},\theta_{(k)})dz + \log f(\mathbf{x}|\theta_{(k)}) - \int_z \log\{f(\mathbf{x},\mathbf{z}|\theta_{(k)})\} f(\mathbf{z}|\mathbf{x},\theta_{(k)})dz \,.$$

(The proof technique is setup for continuous Z, but the same proof works for discrete Z as well.)

Naturally, we can see that at $\theta = \theta_{(k)}$, $\tilde{f}(\theta_{(k)}|\theta_{(k)}) = f(\mathbf{x}|\theta_{(k)})$. We will now show that minorizing property.

$$\begin{aligned}&\tilde{f}(\theta|\theta_{(k)}) \\ &= \int_z \log\{f(\mathbf{x},\mathbf{z}|\theta)\} f(\mathbf{z}|\mathbf{x},\theta_{(k)})dz + \log f(\mathbf{x}|\theta_{(k)}) - \int_z \log\{f(\mathbf{x},\mathbf{z}|\theta_{(k)})\} f(\mathbf{z}|\mathbf{x},\theta_{(k)})dz\end{aligned}$$

$$= \int_z \log\{f(\mathbf{x},\mathbf{z}|\theta)\} f(\mathbf{z}|\mathbf{x},\theta_{(k)})dz + \int_z \log f(\mathbf{x}|\theta_{(k)}) f(\mathbf{z}|\mathbf{x},\theta_{(k)})dz - \int_z \log\{f(\mathbf{x},\mathbf{z}|\theta_{(k)})\} f(\mathbf{z}|\mathbf{x},\theta_{(k)})dz$$

$$= \int_z \log\left\{\frac{f(\mathbf{x},\mathbf{z}|\theta)\, f(\mathbf{x}|\theta_{(k)})}{f(\mathbf{x},\mathbf{z}|\theta_{(k)})}\right\} f(\mathbf{z}|\mathbf{x},\theta_{(k)})$$

$$= \int_z \log\left\{\frac{f(\mathbf{x},\mathbf{z}|\theta)\, f(\mathbf{x}|\theta_{(k)})}{f(\mathbf{x},\mathbf{z}|\theta_{(k)})}\right\} f(\mathbf{z}|\mathbf{x},\theta_{(k)}) + \log f(\mathbf{x}|\theta) - \log f(\mathbf{x}|\theta)$$

$$= \int_z \log\left\{\frac{f(\mathbf{x},\mathbf{z}|\theta)\, f(\mathbf{x}|\theta_{(k)})}{f(\mathbf{x},\mathbf{z}|\theta_{(k)}) f(\mathbf{x}|\theta)}\right\} f(\mathbf{z}|\mathbf{x},\theta_{(k)}) + \log f(\mathbf{x}|\theta)$$

By Jensen's inequality,

$$\begin{aligned}
&\le \log\left[\int_z \left\{\frac{f(\mathbf{x},\mathbf{z}|\theta)\, f(\mathbf{x}|\theta_{(k)})}{f(\mathbf{x},\mathbf{z}|\theta_{(k)}) f(\mathbf{x}|\theta)}\right\} f(\mathbf{z}|\mathbf{x},\theta_{(k)})\right] + \log f(\mathbf{x}|\theta)\\
&= \log\left[\int_z \frac{f(\mathbf{z}|\mathbf{x},\theta)}{f(\mathbf{z}|\mathbf{x},\theta_{(k)})} f(\mathbf{z}|\mathbf{x},\theta_{(k)})\right] + \log f(\mathbf{x}|\theta)\\
&= \log \int_z f(\mathbf{z}|\mathbf{x},\theta)dz + \log f(\mathbf{x}|\theta)\\
&= \log f(\mathbf{x}|\theta)\,.
\end{aligned}$$

Thus, $\tilde{f}(\theta|\theta_{(k)})$ is a minorizing function, and the next iterate is

$$\theta_{(k+1)} = \arg\max_{\theta} \tilde{f}(\theta|\theta_{(k)}) = \arg\max_{\theta} q(\theta|\theta_{(k)})$$

□

8.3 (Back to) Gaussian mixture likelihood

We will look at the general setup of C groups, so that the density for $X_1,\dots,X_n$ is

$$f(x \mid \theta) = \sum_{j=1}^{C} \pi_j f_j(x \mid \mu_j, \sigma_j^2)\,,$$

where $\theta = (\mu_1, \dots, \mu_C, \sigma_1^2, \dots, \sigma_C^2, \pi_1, \dots, \pi_{C-1})$. The setup is the same as before, and suppose we the <u>*complete data*</u> (X_i, Z_i) where

$$[X_i \mid Z_i = c] \sim N(\mu_c, \sigma_c^2) \quad \text{and} \quad \Pr(Z_i = c) = \pi_c .$$

To implement the EM algorithm for this example, we first need to find $q(\theta|\theta_{(k)})$, which requires finding the distribution of $Z|X$. This can be done by Bayes' theorem, since

$$\Pr(Z = c \mid X = x_i) = \frac{f(x_i \mid Z = c) \Pr(Z = c)}{f(x_i)} = \frac{f_c(x_i \mid \mu_c, \sigma_c^2)\ \pi_c}{\sum_{j=1}^{C} f_j(x_i \mid \mu_j, \sigma_j^2)\pi_j} =: \gamma_{i,c} .$$

So for any kth iterate with current step $\theta_{(k)} = (\mu_{1,k}, \mu_{2,k}, \sigma_{1,k}^2, \sigma_{2,k}^2, p_k^*)$, we have

$$\Pr(Z = c \mid X = x_i, \theta_{(k)}) = \frac{f_c(x_i \mid \mu_{c,k}, \sigma_{c,k}^2)\pi_{c,k}}{\sum_{j=1}^{C} f_j(x_i \mid \mu_{j,k}, \sigma_{j,k}^2)\pi_{j,k}} := \gamma_{i,c,k} .$$

<u>**NOTE:**</u> $\gamma_{i,c}$ are itself quantities of interest since they tell us the probability of the ith observation being in class c. In this way, at the end we get probabilities of association for each data point that inform us about the classification of the observations.

Next,

$$\begin{aligned}
q(\theta \mid \theta_{(k)}) &= \mathrm{E}_{Z|X}\left[\log f(\mathbf{x}, \mathbf{z}|\theta) \mid X = x, \theta_{(k)}\right] \\
&= \mathrm{E}_{Z|X}\left[\sum_{i=1}^{n} \log f(x_i, z_i|\theta) \mid X = x, \theta_{(k)}\right] \\
&= \sum_{i=1}^{n} \mathrm{E}_{Z_i|X_i}\left[\log f(x_i, z_i|\theta) \mid X = x_i, \theta_{(k)}\right] \\
&= \sum_{i=1}^{n}\sum_{c=1}^{C} \left[\log f(x_i, z_i = c|\theta)\right] \Pr(Z = c \mid X = x_i, \theta_{(k)}) \\
&= \sum_{i=1}^{n}\sum_{c=1}^{C} \left[\log f(x_i|, z_i = c, \theta) \Pr(z_i = \pi_c)\right] \Pr(Z = c \mid X = x_i, \theta_{(k)}) \\
&= \sum_{i=1}^{n}\sum_{c=1}^{C} \log\left\{f_c(x_i|\mu_c, \sigma_c^2)\pi_c\right\} \underbrace{\frac{f_c(x_i|\mu_{c,k}, \sigma_{c,k}^2)\pi_{c,k}}{\sum_{j=1}^{C} f_j(x_i|\mu_{j,k}, \sigma_{j,k}^2)\pi_{j,k}}}_{\gamma_{i,c,k}} .
\end{aligned}$$

This is expectation is in a complicated form, but we have it available! Notice that we will need to store the value of $\gamma_{i,c,k}$ in order to implement the E-step:

$$\gamma_{i,c,k} = \frac{f_c(x_i \mid \mu_{c,k}, \sigma^2_{c,k})\pi_{c,k}}{\sum_{j=1}^C f_j(x_i \mid \mu_{j,k}, \sigma^2_{j,k})\pi_{j,k}} \, .$$

This completes the E-step. We move on to the M-step. To complete the M-step

$$\theta_{(k+1)} = \arg\max q(\theta \mid \theta_{(k)}) \, .$$

$$\begin{aligned} q(\theta \mid \theta_{(k)}) &= \sum_{i=1}^n \sum_{c=1}^C \left\{ \log f_c(x_i \mid \mu_c, \sigma^2_c) + \log \pi_c \right\} \gamma_{i,c,k} \\ &= \sum_{i=1}^n \sum_{c=1}^C \left[-\frac{1}{2}\log(2\pi) - \frac{1}{2}\log\sigma^2_c - \frac{(X_i - \mu_c)^2}{2\sigma^2_c} + \log\pi_c \right] \gamma_{i,c,k} \\ &= \text{const} - \frac{1}{2}\sum_{i=1}^n \sum_{c=1}^C \log\sigma^2_c \gamma_{i,c,k} - \sum_{i=1}^n \sum_{c=1}^C \frac{(X_i - \mu_c)^2}{2\sigma^2_c}\gamma_{i,c,k} + \sum_{i=1}^n \sum_{c=1}^C \log\pi_c \, \gamma_{i,c,k} \, . \end{aligned}$$

Taking derivatives and setting to 0, we get that for any c,

$$\frac{\partial q}{\partial \mu_c} = \sum_{i=1}^n \frac{(x_i - \mu_c)\,\gamma_{i,c,k}}{\sigma^2_c} \overset{\text{set}}{=} 0 \;\Rightarrow\; \mu_{c,(k+1)} = \frac{\sum_{i=1}^n \gamma_{i,c,k}\; x_i}{\sum_{i=1}^n \gamma_{i,c,k}} \, , \tag{3}$$

and the second derivative is clearly negative. For σ^2_C,

$$\frac{\partial L}{\partial \sigma^2_c} = -\frac{1}{2}\sum_{i=1}^n \frac{\gamma_{i,c,k}}{\sigma^2_c} + \sum_{i=1}^n \frac{(X_i - \mu_c)^2}{2\sigma^4_c}\gamma_{i,c,k} \overset{\text{set}}{=} 0 \;\Rightarrow\; \sigma^2_{c,(k+1)} = \frac{\sum_{i=1}^n \gamma_{i,c,k}(x_i - \mu^2_{c,(k+1)})}{\sum_{i=1}^n \gamma_{i,c,k}} \, . \tag{4}$$

(You can show for yourself that the second derivative at this solutions is negative.)

For π_c note that the optimization requires a constraint, since $\sum_{c=1}^C \pi_c = 1$. So we will use Lagrange multipliers. The modified objective function, for $\lambda > 0$ is

$$\tilde{q}(\theta \mid \theta_{(k)}) = q(\theta \mid \theta_{(k)}) - \lambda \left(\sum_{c=1}^C \pi_c - 1 \right) \, .$$

Taking derivative

$$\begin{aligned}
&\Rightarrow \frac{\partial \tilde{q}}{\partial \pi_c} = \sum_{i=1}^{n} \frac{\gamma_{i,c,k}}{\pi_c} - \lambda \overset{\text{set}}{=} 0 \\
&\Rightarrow \pi_c = \sum_{i=1}^{n} \frac{\gamma_{i,c,k}}{\lambda} \\
&\Rightarrow \sum_{c=1}^{C} \pi_c = \sum_{c=1}^{C} \sum_{i=1}^{n} \frac{\gamma_{i,c,k}}{\lambda} \\
&\Rightarrow 1 = \frac{1}{\lambda} \sum_{i=1}^{n} 1 \\
&\Rightarrow \lambda = n \\
&\Rightarrow \pi_{c,(k+1)} = \frac{1}{n} \sum_{i=1}^{n} \gamma_{i,c,k} \,. \qquad (5)
\end{aligned}$$

(and the second derivative is clearly negative). Thus equations (3), (3) and (5) provide the iterative updates for the parameters. The final algorithm is:

Algorithm 16 EM Algorithm for Mixture of Gaussians

```
1: Set the initial value: $\theta_{(0)} = (\mu_{1,(0)}, \dots, \mu_{C,(0)}, \sigma^2_{1,(0)}, \sigma^2_{C,(0)}, \pi_{1,(0)}, \dots, \pi_{C,(0)})$
2: For all $c = 1, \dots, C$ and all $i = 1, \dots, n$, calculate
       $$\gamma_{i,c,(k)} = \frac{f_c(x_i \mid \mu_{c,(k)}, \sigma^2_{c,(k)})\pi_{c,(k)}}{\sum_{j=1}^{C} f_j(x_i \mid \mu_{j,(k)}, \sigma^2_{j,(k)})\pi_{j,(k)}} .$$
3: For all $c = 1, \dots, C$
       $$\mu_{c,(k+1)} = \frac{\sum_{i=1}^{n} \gamma_{i,c,(k)} \; x_i}{\sum_{i=1}^{n} \gamma_{i,c,(k)}}$$
       $$\sigma^2_{c,(k+1)} = \frac{\sum_{i=1}^{n} \gamma_{i,c,(k)}(x_i - \mu^2_{c,(k+1)})}{\sum_{i=1}^{n} \gamma_{i,c,(k)}}$$
       $$\pi_{c,(k+1)} = \frac{1}{n} \sum_{i=1}^{n} \gamma_{i,c,(k)}$$
4: Stop when $\|\theta_{(k+1)} - \theta_{(k)}\| < \epsilon$.
```

Note: The target likelihood $f(\mathbf{x}|\theta)$ is not concave, so the algorithm is **not** guaranteed to converge to a global maxima.

Questions to think about

- What happens when you change the starting values drastically?
- What happens when you increase the number of clusters C?
- Can you setup the EM algorithm for multivariate normal distributions?

8.4 EM Algorithm for Censored Data

The EM algorithm is particularly employed when dealing with *censored* data: censored data is when the realization of a random variable is only partially known. Consider the following example.

A light bulb company is testing the failure times of their bulbs and know that failure times follow Exp(λ) for some $\lambda > 0$. They test n light bulbs, so the failure time of each light bulb is

$$Z_1, \dots, Z_n \overset{\text{iid}}{\sim} \text{Exp}(\lambda)\,.$$

However, officials recording these failure times walked into the room only at time T and observed that $m < n$ of the bulbs had already failed. Their failure time cannot be recorded. Define $E_j = I(Z_j < T)$, so observed data is

$$E_1 = 1, \dots, E_m = 1, Z_{m+1}, Z_{m+2}, \dots, Z_n\,.$$

Note that $E_i \sim \text{Bern}(p)$ where $p = \Pr(E_i = 1) = \Pr(Z_i \leq T) = 1 - e^{-\lambda T}$ (from the CDF of an exponential distribution). Our goal is to find the MLE for λ. Note that

- If we ignore the first m light bulbs, then not only do we have a smaller sample size, but we also have a biased sample which do not contain the bottom tail of the distribution of failure times.

So we must account for the "missing data". Let us first write down the observed data likelihood.

$$\begin{aligned}
L(\lambda|E_1, \dots, E_m, Z_{m+1}, \dots, Z_n) &= f(E_1, \dots, E_m, Z_{m+1}, \dots, Z_n|\lambda)\\
&= \prod_{i=1}^{m} \Pr(E_i = 1) \cdot \prod_{j=m+1}^{n} f(z_j|\lambda)\\
&= \prod_{i=1}^{m} \left(1 - e^{-\lambda T}\right) \prod_{j=m+1}^{n} \lambda \exp\left\{-\lambda Z_j\right\}
\end{aligned}$$

$$= (1 - e^{-\lambda T})^m \lambda^{n-m} \exp\left\{-\lambda \sum_{j=m+1}^{n} Z_j\right\}.$$

Closed-form MLEs are difficult here, and some sort of numerical optimization is useful. Of course, here, we cn very easily implement our gradient-based methods. But, we will resort to the EM algorithm, as that will give us something extra since that has the added advantage that the estimates of $Z_1, \dots, Z_m$ may be obtained as well.

Also, note that if we choose to "throw away" the censored data, then the likelihood is

$$L_{bad}(\lambda | Z_{m+1} \dots Z_n) = \lambda^{n-m} \exp\left\{-\lambda \sum_{j=m+1}^{n} Z_j\right\}$$

and the MLE is

$$\lambda_{\text{MLE, bad}} = \frac{n-m}{\sum_{j=m+1}^{n} Z_j}$$

The above MLE is a bad estimator since the data thrown away is not censored at random, and in fact, all those bulbs fused early. So the bulb company cannot just throw that data away, as that would be dishonest!

Now we implement the EM algorithm for this example. First, note that the complete unobserved data is $Z_1, \dots, Z_n$ and the complete log likelihood is

$$\log L_{\text{comp}}(\lambda | Z_1, \dots, Z_n) = \log f(\mathbf{z}|\lambda) = \log\left\{\prod_{i=1}^{n} \lambda e^{-\lambda z_i}\right\} = n \log\lambda - \lambda \sum_{i=1}^{n} z_i \,.$$

In order to implement the EM algorithm, we need the conditional distribution of the unobserved data, given the observed data. Unobserved data is $Z_1, \dots, Z_m$:

$$\begin{aligned}
&f(Z_1, \dots, Z_m \mid E_1, E_2, \dots, E_m, Z_{m+1}, \dots, Z_n)\\
&= f(Z_1, \dots, Z_m \mid E_1, \dots, E_m, Z_{m+1}, \dots, Z_n)\\
&= f(Z_1, \dots, Z_m \mid E_1, \dots, E_m)\\
&= \prod_{i=1}^{m} f(Z_i \mid E_i)\\
&= \prod_{i=1}^{m} f(Z_i \mid Z_i \le T)\\
&= \prod_{i=1}^{m} \frac{\lambda e^{-\lambda Z_i}}{1 - e^{-\lambda T}} \mathbb{I}(Z_i \le T)\,.
\end{aligned}$$

Further,

$$\begin{aligned}\mathrm{E}\left[Z_i|E_i = 1\right] &= \mathrm{E}\left[Z_i|Z_i \le T\right]\\ &= \int_0^T z_i \frac{\lambda e^{-\lambda z_i}}{1-e^{-\lambda T}} = \cdots = \frac{1}{\lambda} - \frac{Te^{-\lambda T}}{1-e^{-\lambda T}}\,.\end{aligned}$$

Once we have the conditional likelihood of the unobserved observations, we are rready to implement EM algorithm. Implementing the EM steps now

1. **E-Step**: In the E-step, we find the expectation of the complete log likelihood under $Z_{1:m}|(E_{1:m}, Z_{(m+1):n})$. That is

$$\begin{aligned}q(\lambda \mid \lambda_{(k)}) &= \mathrm{E}\Big[\log f(Z_1,\dots,Z_n|\lambda) \mid E_1,\dots,E_m,Z_{m+1},\dots,Z_n\Big]\\ &= n\log\lambda - \lambda \mathrm{E}_{\lambda_{(k)}}\left[\sum_{i=1}^n Z_i|E_1=1,\dots,E_m=1,Z_{m+1},\dots,Z_n\right]\\ &= n\log\lambda - \lambda\sum_{i=m+1}^n Z_i - \lambda\sum_{i=1}^m [\mathrm{E}[Z_i|Z_i\le T]\end{aligned}$$

2. **M-Step**: To implement the M-step:

$$\lambda_{(k+1)} = \arg\max_{\lambda}\left[n\log\lambda - \lambda\sum_{i=m+1}^n Z_i - \lambda\sum_{i=1}^m [\mathrm{E}[Z_i|Z_i\le T]\,.\right]$$

It is then easy to show that the M step makes the following update (show by yourself):

$$\lambda_{(k+1)} = \frac{n}{\sum_{i=m+1}^n Z_i + \sum_{i=1}^m [\mathrm{E}[Z_i|Z_i\le T]} = \frac{n}{\sum_{i=m+1}^n Z_i + m\left[\dfrac{1}{\lambda_{(k)}} - \dfrac{Te^{-\lambda_{(k)}T}}{1-e^{-\lambda_{(k)}T}}\right]}$$

Questions to think about

- If the E-step is not solvable, in the sense that the expectation is not available, what can we do?
- What is an added benefit of doing EM rather than NR or Gradient Ascent on the observed likelihood?

8.5 Exercises

1. Following the steps from class, write the EM algorithm for a mixture of K Gaussians, for any general K. That is, the distribution is

$$f(x|\theta) = \sum_{k=1}^{K} \pi_k f_k(x|\mu_k, \sigma_k^2)\,.$$

2. (Using R) Consider the `faithful` dataset in R, which contains waiting time between eruptions and the duration of each eruption for the Old Faithful geyser in Yellowstone National Park, Wyoming, USA. First, run the following code

```
data(faithful)
plot(density(faithful$eruptions))
```

   You will see that the length of the eruptions looks like a bimodal distribution. For any given eruption, let X_i be the eruption time. Let

$$Z_i = \begin{cases} 1 & X_i \text{ has short eruptions} \\ 2 & X_i \text{ has long eruptions} \end{cases}$$

   Thus Z_i is a *latent* variable which is not observed. Let π_1 and π_2 be the probability of short and long eruptions, respectively. Assume that the joint distribution of (X, Z) is

$$f(x, z|\theta) = \pi_1 f_1(x|\mu_1, \sigma_1^2) I(Z = 1) + \pi_2 f_2(x|\mu_2, \sigma_2^2) I(Z = 2)\,.$$

   Implement the EM algorithm for this example.

3. (Using R) For the same dataset `faithful`, we will fit a *multivariate* mixture of Gaussians for both the eruption time and waiting times. Let X_i be the eruption time and Y_i be the waiting time for the ith eruption. Let

$$Z_i = \begin{cases} 1 & X_i \text{ and } Y_i \text{ has short eruptions and short wait times} \\ 2 & X_i \text{ and } Y_i \text{ has long eruptions and long wait times} \end{cases}.$$

   First, we want to find the EM steps for this. The joint distribution of the observed

$t = (x, y)$ and the latent variable z is

$$f(t, z|\theta) = \pi_1 f_1(t|\mu_1, \Sigma_1) I(Z = 1) + \pi_2 f_2(t|\mu_2, \Sigma_2^2) I(Z = 2)\,,$$

where $\mu_c \in \mathbb{R}^2$, $\Sigma_c \in \mathbb{R}^{2\times 2}$ and

$$f_c(t \mid \mu_c, \sigma_c^2) = \left(\frac{1}{2\pi}\right) \frac{1}{|\Sigma_c|^{1/2}} \exp\left\{-\frac{(t-\mu_c)^T \Sigma_c^{-1} (t-\mu_c)}{2}\right\}.$$

Similar to the one dimensional case, set up the EM algorithm for this two-dimensional case, and then implement this on the Old Faithful dataset.

4. Repeat the previous exercises for four latent class defined as

$$Z_i = \begin{cases} 1 & X_i \text{ and } Y_i \text{ has short eruptions and short wait times} \\ 2 & X_i \text{ and } Y_i \text{ has long eruptions and long wait times} \\ 3 & X_i \text{ has short eruptions and } Y_i \text{ has long wait times} \\ 4 & X_i \text{ has long eruptions and } Y_i \text{ has short wait times} \end{cases}.$$

5. (EM algorithm for multinomial) Suppose $y = (y_1, y_2, y_3, y_4)$ has a multinomial distribution with probabilities

$$\left(\frac{1}{2} + \frac{\theta}{4}, \frac{1-\theta}{4}, \frac{1-\theta}{4}, \frac{\theta}{4}\right).$$

The joint distribution of y is

$$g(y \mid \theta) = \frac{(\sum y_i)!}{\prod_{i=1}^4 y_i!} \left(\frac{1}{2} + \frac{\theta}{4}\right)^{y_1} \left(\frac{1-\theta}{4}\right)^{y_2} \left(\frac{1-\theta}{4}\right)^{y_3} \left(\frac{\theta}{4}\right)^{y_4}.$$

Suppose you observe $y = (125, 18, 20, 34)$. Also, suppose that the complete data is $(z_1, z_2, y_2, y_3, y_4)$ where $z_1 + z_2 = x_1$. That is, the first variable y_1 is broken into two groups, with the new probabilities are

$$\left(\frac{1}{2}, \frac{\theta}{4}, \frac{1-\theta}{4}, \frac{1-\theta}{4}, \frac{\theta}{4}\right).$$

The complete data distribution is

$$f(z, y \mid \theta) = \frac{(z_1 + z_2 + y_2 + y_3 + y_4)!}{z_1! z_2! y_2! y_3! y_4!} \left(\frac{1}{2}\right)^{x_1} \left(\frac{\theta}{4}\right)^{x_2} \left(\frac{1-\theta}{4}\right)^{y_2} \left(\frac{1-\theta}{4}\right)^{y_3} \left(\frac{\theta}{4}\right)^{y_4}.$$

The E-Step is

$$q(\theta|\theta_{(k)}) = \mathrm{E}_{\theta_{(k)}}\left[\log f(z, y \mid \theta) \mid y_1, y_2, y_3, y_4\right] .$$

Write the above expectation explicitly, and then write the M-step. Implement this in R and return the estimate of θ.

6. Suppose the expectation in the E-step is not tractable. That is the expectations is not available in closed form. In this case, one may replace the expectation with a Monte Carlo estimate. This is called *Monte Carlo EM*. Repeat Exercise 5 using Monte Carlo EM.

7. Will the above algorithm have the ascend property.

8. Consider a two component Gamma($\alpha_i, 1$) mixture density

$$f(x; \alpha_1, \alpha_2, \pi_1, \pi_2) = \pi_1 f_1(x; \alpha_1) + \pi_2 f_2(x; \alpha_2) ,$$

where $\pi_1, \pi_2, \alpha_1, \alpha_2 > 0$ and $\pi_1 = 1 - \pi_2$. Recall that:

$$f_i(x; \alpha_i) = \frac{1}{\Gamma(\alpha_i)} x^{\alpha - 1} e^{-x} .$$

   (a) Construct an EM algorithm to obtain the MLE of $\alpha_1, \alpha_2, \pi_1, \pi_2$. Present all details and do not skip steps.

   (b) Implement the algorithm on the `eruptions` dataset. What are your final estimates of $\alpha_1, \alpha_2, \pi_1, \pi_2$? Is there much difference in the clusters from the Gaussian mixture model?

9 Choosing Tuning Parameters

Still working on this We will take a break from optimization methods to discuss an important practical question:

How do we choose model tuning parameters?

In ridge/bridge/lasso regression, we require choosing λ and/or α. In Gaussian mixture models, we need to choose the number of clusters C.

9.1 Components in Gaussian Mixture Model

Suppose for observed data $X_1, X_2, \dots, X_n$ our goal is to cluster these observations using a Gaussian Mixture Model:

$$f(x \mid \theta) = \sum_{j=1}^{C} \pi_j f_j(x \mid \mu_j, \sigma_j^2)\,,$$

In the above C denotes the number of components/classes/groups/populations/cluster. The choice of C depends on us, so the question is how to choose C?

Recall that we use the log-likelihood to see whether our estimates are good or not.A larger value of the log-likelihood means better models! Thus, in principle, we should be able to use negative log-likelihood to see how bad our estimates are. However, the negative log-likelihood will keep reducing always as C increases, since each data point will want to be it's own cluster.

An alternative model selection procedure is used, called the *Bayesian Information Criterion*. The AIC is a function of the log-likelihood and a penalty term, penalizing the number of parameter the algorithm has to estimate. So if C increases, the number of parameters to be estimated increases, and a penalty is imposed. That is,

$$\text{BIC}(\hat{\theta} \mid x) = 2 \log l(\hat{\theta}|x) - \log(n)K$$

where K is again the number of parameters being estimated. The BIC decreases when the number of clusters increase and for large data.. This makes sense since when large number of data are available, we should be able to find simpler models.

Note: We want to choose C such that the value of BIC is *maximized.*

9.2 Loss functions

A loss function is a measure of error of an estimator from the true value. Having observed data, y, let $\hat{f}$ be the model fit. Then a loss function is a function $L(y, \hat{f})$ that quantifies the *distance* between y and $\hat{f}$. The form of the loss function may depend on the type of data. We will focus only on binary/Gaussian regression and the Gaussian mixture models.

- **Linear regression:** In penalized or non-penalized regression, we have an observation y, covariates X, and a regression coefficient, β. Let $\hat{\beta}$ be the estimated regression coefficient – this could be obtained via ridge, bridge, or regular regression. Then the estimated response is

  $$\hat{f}(x) = x^T \hat{\beta}\,.$$

  A loss function measures the error between y and the estimated response $\hat{y} = \hat{f}(x)$. For continuous response y, there are two popular loss functions:

  - Squared error: $L(y, \hat{f}(x)) = (y - \hat{f}(x))^2$
  - Absolute error: $L(y, \hat{f}(x)) = |y - \hat{f}(x)|$

  We will focus mainly on the squared error loss.

- **Binary data classification**: For binary data models, like logistic regression, a popular loss function is the *misclassification* also known as the $0-1$ loss. Given response y which are Bernoulli(p) where

  $$p = \frac{e^{x^T\beta}}{1 + e^{x^T\beta}}\,,$$

  we can find the estimated p, $\hat{p}$ by setting

  $$\hat{p} = \frac{e^{x^T\hat{\beta}_{\text{MLE}}}}{1 + e^{x^T\hat{\beta}_{\text{MLE}}}}\,,$$

  where $\hat{\beta}_{\text{MLE}}$ are the MLE estimates obtained using an optimization algorithm. These $\hat{p}$ are the estimated probability of success. Set $\hat{f}(x) = 1 \cdot \mathbb{I}\{\hat{p} \geq .5\} + 0 \cdot \mathbb{I}\{\hat{p} < .5\}$. (The cutoff, .5, is the default cutoff, but can be changed depending on the problem). Then the *misclassification* loss function checks whether the model has misclassified the observation

$$\text{Misclassification or } 0-1 \text{ loss} : L(y, \hat{f}(x)) = \mathbb{I}(y \neq \hat{f}(x))\,.$$

How do we use these loss functions?

Given a dataset, we are interested in estimating the *test error* which is the expected loss, of an independent dataset given estimates from the current dataset. If our given dataset is $\mathcal{D}$

$$\text{Err}_{\mathcal{D}} = \text{E}\left[L(y, \hat{f}(x)) | \mathcal{D}\right]\,,$$

where $\hat{f}(x)$ denotes the estimated y for a new independent dataset, given estimators from the current dataset. For example, using $\hat{\beta}$ from a given dataset $\mathcal{D}$, then $\hat{f}(x) = x_{\text{new}}\hat{\beta}$.

9.3 Cross-validation

In many data analysis techniques, there are tuning/nuisance parameters that need to be chosen in order to fit the model. How can we choose these nuisance parameters? For example:

- λ, the tuning parameter in penalized regression
- α, the bridge regression penalization factor

Or, we can fit multiple different models to the same dataset. For example, we could fit linear regression, bridge regression, or ridge regression. Which fit is the best? How can we make that assessment?

Cross-validation provides a way to estimate the *test error* without requiring new data. In cross-validation, our original dataset is broken into chunks in order to emulate independent datasets. Then the model is fit on some chunks and tested on other chunks, with the loss recorded. The way the data is broken into chunks can lead to different methods of cross-validation.

9.3.1 Holdout method

One way is to use the holdout method. In this,

- Choose a loss function
- We randomly split the data into two parts: a *training set* and a *test set*.
- Fit the model on the training set, and obtain estimates of the observation y for the test set. Compute loss on the test set.
- Repeat the process for different models and choose the model with smallest error.

This method has two drawbacks:

1. We do not often have the luxury of "throwing away" observations for the test data, specially in small data problems.
2. It is possible, that just by chance a bad split in the data makes the test/train split data drastically different.

The main advantage is that, compared to the methods that follow, this is fairly cheap.

9.3.2 Leave-one-out Cross-validation

In leave-one-out cross-validation (LOOCV), the data $\mathcal{D}$ of size n is randomly split into a *training set* of size $n-1$ and a *test set* of size 1. This is repeated for systematically all observations so that there are n such splits possible.

For each split, the test error is estimated, and the average error over all splits is calculated, which estimates the expected test error for a model fit $\hat{f}(x)$. Let $\hat{f}^{-i}(x_i)$ denote the predicted value of y_i using the model that removes the ith data point. Then CV estimate of the prediction error is

$$\text{CV}_1(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{f}^{-i}(x_i)) \qquad \approx \text{E}\left[L(y, \hat{f}(x))\right] .$$

Note that each $\hat{f}^{-i}(x_i)$ represents model fits using different datasets, with testing on one observation.

$\text{CV}_1(\hat{f})$ can be calculated for different models or tuning parameters. Let γ denote a generic tuning parameter utilized in determining the model fit $\hat{f}$, and let $\hat{f}^{-i}(x_i, \gamma)$ denote the predicted value for y_i using a training data that doesn't include the ith observation and uses γ as a tuning parameter. Then:

$$\text{CV}_1(\hat{f}, \gamma) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{f}^{-i}(x_i, \gamma)) .$$

The chosen model is the one with γ such that

$$\gamma_{\text{chosen}} = \arg\min_{\gamma} \left\{ \text{CV}_1(\hat{f}, \gamma) \right\}$$

The final model is $\hat{f}(X, \gamma_{\text{chosen}})$ fit to *all* the data. In this way we can accomplish two things: obtain an estimate of the prediction error and choose a model.

Points:

- $\text{CV}_1(\hat{f})$ is an approximately unbiased estimator of the test error. This is because the expectation is being evaluated over n samples of the data that are very close to the original data $\mathcal{D}$.
- LOOCV is computationally burdensome since the model is fit n times for each θ. If the number of possible values of γ is large, this becomes computationally expensive.

9.3.3 K-fold cross-validation

The data is randomly split into K roughly equal-sized parts. For any kth split, the rest of the $K-1$ parts make up the *training set* and the model is fit to the *training set*. We then estimate the prediction error for each element in the kth part. Repeating this for all $k = 1, 2, \dots, K$ parts, we have an estimate of the prediction error.

Let $\kappa : \{1, \dots, N\} \mapsto \{1, \dots, K\}$ indicates the partition to which each ith observation belongs. Let $\hat{f}^{-\kappa(i)}(x)$ be the fitted function for the $\kappa(i)th$ partition removed. Then, the estimated prediction error is

$$\text{CV}_K(\hat{f}, \gamma) = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{n/K} \sum_{i \in k^{\text{th}}\text{split}} L(y_i, \hat{f}^{-\kappa(i)}(x_i, \gamma)) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}^{-\kappa(i)}(x_i, \gamma)) \,.$$

The chosen model is the one with γ such that

$$\theta_{\text{chosen}} = \arg\min_{\gamma} \left\{ \text{CV}_K(\hat{f}, \gamma) \right\}$$

The final model is $\hat{f}(X, \gamma_{\text{chosen}})$ fit to *all* the data.

Points:

- For small K, the bias in estimating the true test error is large since each training data is quite different from the given dataset $\mathcal{D}$.

- The computational burden is lesser when K is small.

Usually, for large datasets, 10-fold or 5-fold CV is common. For small datasets, LOOCV is more common.

Questions to think about

- Which algorithms do you think would be time-consuming to do LOOCV for?

9.4 Bootstrapping

We have discussed cross-validation, which we use to choose model tuning parameters. However, once the final model is fit, we would like to do *inference*. That is, we want to account for the variability of the final estimators obtained, and potentially do testing.

If our estimates are MLEs then we know that under certain important conditions, MLEs have asymptotic normality, that is

$$\sqrt{n}(\hat{\theta}_{\text{MLE}} - \theta) \xrightarrow{d} N(0, \sigma^2_{\text{MLE}})\,,$$

where σ^2_{MLE} is the inverse Fisher information. Then, if we can estimate σ^2_{MLE}, we can construct asymptotically normal confidence intervals:

$$\hat{\theta}_{\text{MLE}} \pm z_{1-\alpha/2}\sqrt{\frac{\hat{\sigma}^2_{MLE}}{n}}\,.$$

We can also conduct hypothesis tests etc and go on to do regular statistical analysis. But sometimes we cannot use an asymptotic distribution:

1. when our estimates are not MLEs, like ridge and bridge regression
2. when the assumptions for asymptotic normality are not satisfied (I haven't shared these assumptions)
3. when n is not large enough for asymptotic normality to hold

In Bootstrapping, we approximate the distribution of $\hat{\theta}$, and from there we will obtain a confidence intervals.

Suppose $\hat{\theta}$ is some estimator of θ from sample $X_1, \dots, X_n \overset{iid}{\sim} F$. Then since $\hat{\theta}$ is random it has a sampling distribution G_n that is unknown. If asymptotic normality holds, then $G_n \approx N(\cdot, \cdot)$ for large enough n, but in general we may not know much about G_n. If we could obtain many (say, B) similar datasets, we could obtain an estimate from each of those B datasets:

$$\hat{\theta}_1, \dots, \hat{\theta}_B \overset{iid}{\sim} G_n \,.$$

Once we have B realizations from G_n, we can easily estimate characteristics about G_n, like the overall mean, variance, quantiles, etc.

Thus, in order to learn things about the sampling distribution G_n, our goal is to draw more samples of such data. But this, of course is not easy in real-data scenarios. We could obtain more Monte Carlo datasets from F, but we typically do not know the true F.

In bootstrap, instead of obtaining typical Monte Carlo datasets, we will repeatedly "resample" from our current dataset. This would give us an approximate sample from our distribution G_n, and we could estimate characteristics of this distribution! This resampling using information from the current data is called *bootstrapping*. We will study two popular bootstrap methods: *nonparameteric bootstrap* and *parametric bootstrap*.

9.4.1 Nonparametric Bootstrap

In nonparametric bootstrap, we resample data of size n from within $\{X_1, X_2, \dots, X_n\}$ (with replacement) and obtain estimates of θ using these samples. That is

$$\begin{aligned}
\text{Bootstrap sample 1:} &\qquad X^*_{11}, X^*_{21}, \dots, X^*_{n1} \Rightarrow \hat{\theta}^*_1 \\
\text{Bootstrap sample 2:} &\qquad X^*_{12}, X^*_{22}, \dots, X^*_{n2} \Rightarrow \hat{\theta}^*_2 \\
&\qquad\qquad\qquad\vdots \\
\text{Bootstrap sample B:} &\qquad X^*_{1B}, X^*_{2B}, \dots, X^*_{nB} \Rightarrow \hat{\theta}^*_B \,.
\end{aligned}$$

Each sample is called a bootstrap sample, and there are B bootstrap samples. Now, the idea is that $\hat{\theta}^*_1, \dots \hat{\theta}^*_B$ are B approximate samples from the distribution of $\hat{\theta}, G_n$. That is

$$\hat{\theta}^*_1, \dots \hat{\theta}^*_B \approx G_n \,.$$

If we want to know the variance of $\hat{\theta}$, then the bootstrap estimate of the variance is

$$\widehat{\text{Var}(\hat{\theta})} = \frac{1}{B-1} \sum_{b=1}^{B} (\hat{\theta}^*_b - B^{-1} \sum_k \hat{\theta}^*_k)^2 \,.$$

Similarly, we may want to construct a $100(1-\alpha)\%$ confidence interval for $\hat{\theta}$. A $100(1-\alpha)\%$ confidence interval is the random interval (L, U) such that

$$\Pr((L, U) \text{ contains } \theta) = 1 - \alpha \qquad .$$

Note that here L and U are random and θ is fixed. We can find the confidence interval by looking at the quantiles of the distribution of $\hat{\theta}$, G_n. Since we have bootstrap samples from G_n, we can estimate these quantiles!. So if we order the bootstrap estimates

$$\hat{\theta}^*_{(1)} < \hat{\theta}^*_{(2)} < \cdots < \hat{\theta}^*_{(B)} ,$$

and set L to be the $(\alpha/2)$th ordered statistic and U to be $(1-\alpha/2)$th order statistic, we get:

$$L = \hat{\theta}^*_{\lfloor \alpha/2 * B \rfloor} \quad \text{and } U = \quad \hat{\theta}^*_{\lfloor 1-\alpha/2 * B \rfloor} .$$

Then $\left(\hat{\theta}^*_{\lfloor \alpha/2*B \rfloor}, \hat{\theta}^*_{\lfloor 1-\alpha/2*B \rfloor}\right)$, is a $100(1-\alpha)\%$ bootstrap confidence interval.

Example 38 (Median of Gamma(a, b))**.** Let $X_1, \dots, X_n \overset{iid}{\sim} \text{Gamma}(a, b)$. The median of this distribution (θ) does not have a closed form expression, but suppose we are interested in estimating it with the sample median. That is

$$\hat{\theta} = \text{Sample Median}(X_1, X_2, \dots, X_n) \sim G_n$$

Technically, there is a result of asymptotic normality of the sample median so that $G_n \approx$ normally distribution for large n. However, we may not have enough samples for the asymptotic normality to hold reasonably. Thus, instead here we implement the nonparametric bootstrap:

$$\begin{aligned} X^*_{11}, X^*_{21}, \dots, X^*_{n1} &\Rightarrow \hat{\theta}^*_1 \\ X^*_{12}, X^*_{22}, \dots, X^*_{n2} &\Rightarrow \hat{\theta}^*_2 \\ &\vdots \\ X^*_{1B}, X^*_{2B}, \dots, X^*_{nB} &\Rightarrow \hat{\theta}^*_B \end{aligned}$$

$$.$$

And find $\alpha/2$ and $1-\alpha/2$ sample quantiles from $\hat{\theta}^*_i, i = 1, \dots, B$. Then $\left(\hat{\theta}^*_{\lfloor \alpha/2*B \rfloor}, \hat{\theta}^*_{\lfloor 1-\alpha/2*B \rfloor}\right)$, is a $100(1-\alpha)\%$ bootstrap confidence interval. ■

9.4.2 Parametric Bootstrap

Suppose $X_1, \dots, X_n \sim F(\theta)$, where θ is a parameter we can estimate. Let $\hat{\theta}$ be a chosen estimator of θ. Instead of resampling within our data, in *parametric* bootstrap, we use our estimator of θ to obtain computer generated samples from $F(\hat{\theta})$:

$$\begin{gathered}
X_{11}^*, X_{21}^*, \dots, X_{n1}^* \sim F(\hat{\theta}) \Rightarrow \hat{\theta}_1^* \\
X_{12}^*, X_{22}^*, \dots, X_{n2}^* \sim F(\hat{\theta}) \Rightarrow \hat{\theta}_2^* \\
\vdots \\
X_{1B}^*, X_{2B}^*, \dots, X_{nB}^* \sim F(\hat{\theta}) \Rightarrow \hat{\theta}_B^*
\end{gathered}$$

And again, we find the $\alpha/2$ and $1-\alpha/2$ quantiles of the $\hat{\theta}_i^*$s so that $\left(\hat{\theta}^*_{\lfloor \alpha/2*B \rfloor}, \hat{\theta}^*_{\lfloor 1-\alpha/2*B \rfloor}\right)$, is a $100(1-\alpha)\%$ bootstrap confidence interval.

Example 39 (Coefficient of variation)**.** For a population with variance σ^2 and mean μ, its coefficient of variation is defined as

$$\theta = \frac{\sigma}{\mu}\,.$$

It essentially tells us what is the deviation of the population compared to its mean. For $F = N(\mu, \sigma^2)$, and let $X_1, \dots, X_n \overset{iid}{\sim} N(\mu, \sigma^2)$. We want to estimate θ, the coefficient of variation. The obvious estimator is the sample standard deviation by the sample mean:

$$\hat{\theta} = \frac{\sqrt{s^2}}{\bar{X}} \qquad \text{where } s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2\,.$$

$\hat{\theta}$ is a complicated estimator, and it is unclear if it has some known distribution. We may then understand its sampling distribution based on doing a parametric bootstrap method:

$$\begin{gathered}
X_{11}^*, X_{21}^*, \dots, X_{n1}^* \sim N(\bar{X}, s^2) \Rightarrow \hat{\theta}_1^* \\
X_{12}^*, X_{22}^*, \dots, X_{n2}^* \sim N(\bar{X}, s^2) \Rightarrow \hat{\theta}_2^* \\
\vdots \\
X_{1B}^*, X_{2B}^*, \dots, X_{nB}^* \sim N(\bar{X}, s^2) \Rightarrow \hat{\theta}_B^*\,.
\end{gathered}$$

And find $\alpha/2$ and $1-\alpha/2$ sample quantiles from $\hat{\theta}^*_i, i = 1, \dots, B$. ■

Questions to think about

- What happens if we increase or decrease B?
- How will you use bootstrapping to obtain confidence intervals for bridge regression coefficients?
- Do we need bootstrapping to obtain confidence intervals for ridge regression estimates?

9.5 Exercises

1. *Comparing different cross-validation techniques*: Generate a dataset using the following code:

   ```
   set.seed(10)
   n <- 100
   p <- 50
   sigma2.star <- 4
   beta.star <- rnorm(p, mean = 2)
   beta.star
   ```

   Generate a dataset (y, X) to fit the linear regression model

   $$y = X\beta + \epsilon\,.$$

   Implement the holdout method, leave-one-out, 10-fold, and 5-fold cross-validation over 500 replications. Keep a track of the CV error from each method and compare the performance of all cross-validation methods.

2. *cars dataset*: Estimate the prediction error for the cars dataset using 10-fold, 5-fold, and LOOCV.

3. *mtcars dataset*: Consider the `mtcars` dataset from 1974 Motor Trend US magazine, that comprises fuel consumption and 10 aspects of automobile design and performance for 32 automobiles. Load the dataset using

   ```
   data(mtcars)
   ```

   There are 10 covariates in the dataset, and `mpg` (miles per gallon) is the response variable. Fit a ridge regression model for this dataset and find an optimal λ using 1-fold, 5-fold, and LOOCV cross-validation. Choose the best $\lambda \in \{10^{-8}, 10^{-7.5}, \dots, 10^{7.5}, 10^{8}\}$. Make sure you make the X matrix such that the first column is a column of 1s.

4. *Seeds dataset:* Download the seeds dataset from

   `https://archive.ics.uci.edu/ml/datasets/seeds`

   This dataset contains information about *three* varieties of wheat (last column of the dataset). There are 7 covariate information. Fit a 7-dimensional Gaussian mixture model algorithm with $C = 3$ and estimate the mis-classification rate using cross-validation.

5. *Seeds dataset:* For the same dataset, with $C = 3$, use cross-validation to find out which of the 7 covariates best helps identify between the three kinds of wheat.

6. Generate n observations from a Normal distribution with mean μ and variance σ^2. Use code below:

   ```
   set.seed(1)
   mu <- 5
   sig2 <- 3
   n <- 100
   my.samp <- rnorm(n, mean = mu, sd = sqrt(sig2))
   ```

   Construct bootstrap confidence intervals for estimating the mean of a normal distribution using both parameteric and nonparametric bootstrap methods and compare the confidence intervals with the usual normal distribution confidence intervals for μ.

7. Repeat the previous exercise for estimating the mean of t_{10} distribution from sample of size 50. Are the bootstrap confidence intervals similar to the intervals using CLT? Why or why not?

8. Repeat again to estimate the mean of a Gamma$(.05, 1)$ distribution from a sample of size $n = 50$. Are the bootstrap confidence intervals similar to the intervals using CLT? Why or why not?

9. For Exercise 1, fit a bridge regression model with $\lambda = 5$, $\alpha = 1.5$, and construct 95% parametric and nonparametric bootstrap confidence intervals for of the 50 βs. In repeated simulations, what is the coverage probability of each the con-

fidence intervals. What percentage of the confidence intervals contain the true vector of β, `beta.star`?

10. Obtain 95% bootstrap confidence intervals for each ridge regression coefficient β for the chosen λ value in the *mtcars* Exercise 3.

10 Stochastic optimization methods

We go back to optimization this week. The reason we took a break from optimization is because we will focus on stochastic optimization methods, which will lead the discussion into other stochastic methods. We will cover two topics:

1. Stochastic gradient ascent - used for large scale data problems
2. Simulated annealing - used for non-convex objective functions

Our goal is the same as before: for an objective function $f(\theta)$, our goal is to find

$$\theta^* = \arg\max_\theta f(\theta)\,.$$

10.1 Stochastic gradient ascent

Recall, in order to maximize the objective function the gradient ascent algorithm does the following update:

$$\theta_{(k+1)} = \theta_{(k)} + t\nabla f(\theta_{(k)})\,,$$

where $\nabla f(\theta_{(k)})$ is the gradient vector. Now, in many statistics problems, the objective function is the log-likelihood (for some density $\tilde{f}$). That is, we have $X_1, X_2, \dots, X_n \overset{\text{iid}}{\sim} \tilde{F}$ with density function $\tilde{f}$. Then interest is in :

$$\theta^* = \arg\max_\theta \left\{ \sum_{i=1}^n \log \tilde{f}(x_i|\theta) \right\} = \arg\max_\theta \underbrace{\left\{ \frac{1}{n}\sum_{i=1}^n \log \tilde{f}(x_i|\theta) \right\}}_{f(\theta)}\,.$$

Now due to the objective function being an average, we have the following:

$$\begin{aligned} f(\theta) &= \frac{1}{n}\sum_{i=1}^n \log \tilde{f}(\theta|x_i) \\ \Rightarrow \nabla f(\theta) &= \frac{1}{n}\sum_{i=1}^n \nabla \left[\log \tilde{f}(\theta|x_i)\right]\,. \end{aligned}$$

That is, in order to implement a gradient ascent step, the gradient of the log-likelihood is calculated for the whole data. However, consider the following two situations

- the data size n and/or dimension of θ are prohibitively large so that calculating the full gradient multiple times is infeasible

- the data is not available at once! In many online data situations, the full data set is not available, but comes in sequentially. Then, the full data gradient vector is not available.

In such situations, when the full gradient vector is unavailable, we may replace it with the *estimate* the gradient. Suppose i_k is a randomly chosen index in $\{1, \dots, n\}$. Then

$$\mathbb{E}\left[\underbrace{\nabla\left[\log \tilde{f}(x_{i_k}|\theta)\right]}_{\text{Estimate of full gradient}}\right] = \frac{1}{n}\sum_{i=1}^{n}\left[\nabla \log \tilde{f}(x_i|\theta)\right] .$$

Thus, $\nabla \log\left[\tilde{f}(x_{i_k}|\theta)\right]$ is an unbiased estimator of the complete gradient, but uses *only one data point*. Replacing the complete gradient with this estimate yields the *stochastic gradient ascent* update:

$$\theta_{(k+1)} = \theta_{(k)} + t\left\{\nabla\left[\log \tilde{f}(x_{i_k}|\theta_{(k)})\right]\right\} ,$$

where i_k is a randomly chosen index. This randomness in choosing the index makes this a *stochastic algorithm*.

- advantage: it is much cheaper to implement since only one-data point is required for gradient evaluation
- disadvantage it may require larger k for convergence to the optimal solution
- disadvantage as k increases, $\theta_{(k+1)} \not\to \theta^*$. Rather, after some initial steps, $\theta_{(k+1)}$ oscillates around θ^*.

After K iterations, the final estimate of θ^* is

$$\hat{\theta}^* = \frac{1}{K}\sum_{k=1}^{K}\theta_{(k+1)} .$$

However, since each step involves estimating data gradient, variability in updates of $\theta_{(k)}$ is larger than using gradient ascent. To stabilize this behavior, often `mini-batch` stochastic gradient is used.

10.1.1 Mini-batch stochastic gradient ascent

Let I_k be a random subset of $\{1, \dots, n\}$ of size b. Then, the mini-batch stochastic gradient ascent algorithm implements the following update:

$$\theta_{(k+1)} = \theta_{(k)} + t \left[\frac{1}{b} \sum_{i \in I_k} \nabla \left[\log \tilde{f}(\theta_{(k)} | x_i) \right] \right] .$$

The mini-batch stochastic gradient estimate of θ^* after K updates is

$$\hat{\theta}^* = \frac{1}{K} \sum_{k=1}^{K} \theta_{(k)} .$$

There are not a lot of clear rules about terminating the algorithm in stochastic gradient. Typically, the number of iterations $K = n$, so that one full pass at the data is implemented.

10.1.2 Logistic regression

Recall the logistic regression setup where for a response Y and a covariate matrix X,

$$Y_i \sim \text{Bern}\left(\frac{e^{x_i^T \beta}}{1 + e^{x_i^T \beta}} \right) .$$

In order to find the MLE for β, we obtain the log-likelihood.

$$\begin{aligned} L(\beta | Y) &= \prod_{i=1}^{n} (p_i)^{y_i} (1 - p_i)^{1 - y_i} \\ \Rightarrow \frac{1}{n} \log \tilde{f}(\beta) &= -\frac{1}{n} \sum_{i=1}^{n} \log \left(1 + \exp(x_i^T \beta) \right) + \frac{1}{n} \sum_{i=1}^{n} y_i x_i^T \beta \end{aligned}$$

Taking derivative:

$$\nabla \left[\frac{1}{n} \log \tilde{f}(\beta) \right] = \frac{1}{n} \sum_{i=1}^{n} x_i \left[y_i - \frac{e^{x_i^T \beta}}{1 + e^{x_i^T \beta}} \right] .$$

As noted earlier, the target objective is concave, thus a global optima exists and the gradient ascent algorithm will converge to the MLE. We will implement the stochastic gradient ascent algorithm here. The stochastic gradient ascent algorithm proceeds in the folllowing way, for a randomly chosen index i_k,

$$\beta_{(k+1)} = \beta_{(k)} + t\left[x_{i_k}\left(y_{i_k} - \frac{e^{x_{i_k}^T\beta}}{1+e^{x_{i_k}^T\beta}}\right)\right]$$

NOTE: Here we chose a fixed learning rate t. A common strategy is to choose a learning rate t_k that reduces to 0 as k increases.

10.2 Simulated annealing

Last lecture we went over the stochastic gradient ascent algorithm: the merit of this algorithm was its use in online sequential data and for large data set problem.

This lecture focuses on simulated annealing, an algorithm particularly useful for non-concave objective functions. Our goal is the same as before: for an objective function $f(\theta)$, our goal is to find

$$\theta^* = \arg\max_\theta f(\theta)\,.$$

Recall that when the objective function is non-concave, all of the methods we've discussed cannot escape out of a local maxima. This creates challenges in obtaining global maximas. This is where the method of *simulated annealing* has an advantage over other methods.

Consider an objective function $f(\theta)$ to maximize. Note that maximizing $f(\theta)$ is equivalent to maximizing $\exp(f(\theta))$. The idea in simulated annealing is that, instead of trying to find a maxima directly, we will obtain samples from the density

$$\pi(\theta) \propto \exp(f(\theta))\,.$$

Samples collected from $\pi(\theta)$ are likely to be from areas near the maximas. However, obtaining samples from $\pi(\theta)$ means there will be samples from low probability areas as well. So how do we force samples to come from areas near the maximas?

Consider for $T > 0$, we may also look at:

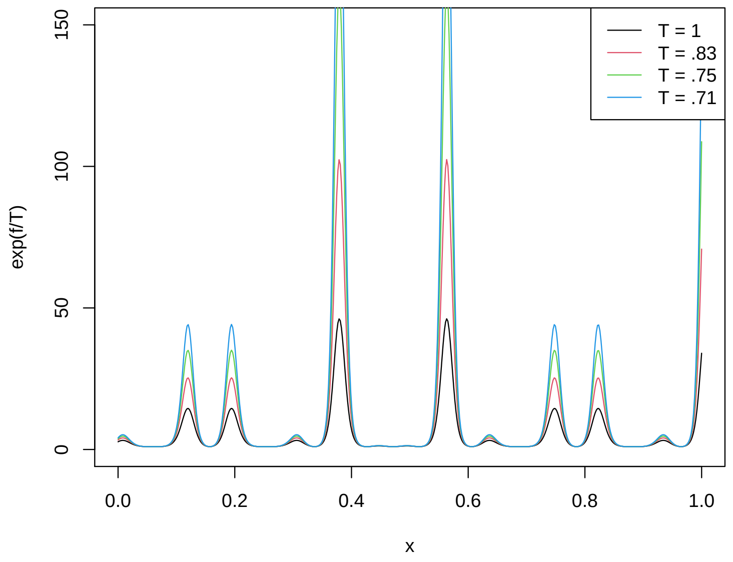
$$\arg\max_\theta f(\theta) = \arg\max_\theta e^{\{f(\theta)/T\}}\,.$$

For $0 < T < 1$, the objective function's modes are exaggerated there-by amplifying the maximas. This feature will help us in trying to "push" the sampling to areas of high-probability.

Example 40. Consider the following objective function

$$f(\theta) = [\cos(50\theta) + \sin(20\theta)]^2 \, I(0 < \theta < 1)$$

Below is a plot of $e^{f(\theta)/T}$ for various values of T.



■

In simulated annealing, this feature is utilized so that every subsequent sample is drawn from an increasingly concentrated distribution. That is, at a time point k, a sample will be drawn from

$$\pi_{k,T}(\theta) \propto e^{f(\theta)/T_k} \,,$$

where T_k is a decreasing sequence.

How do we generate these samples?

Certainly, we can try and use accept-reject or another Monte Carlo sampling method, but such methods cannot be implemented generally. Note that for any θ', θ

$$\frac{\pi_{k,T}(\theta')}{\pi_{k,T}(\theta)} = \exp\left\{\frac{f(\theta') - f(\theta)}{T_k}\right\} .$$

Let G be a proposal distribution with density $g(\theta'|\theta)$ so that $g(\theta'|\theta) = g(\theta|\theta')$. Such a proposal distribution is a symmetric proposal distribution. Further, given a value of θ_k, θ' is sampled using density $g(\cdot|\theta_k)$.

Algorithm 17 Simulated Annealing algorithm

```
1: For $k = 1, \dots, N$, repeat the following:
2: Generate $\theta' \sim G(\cdot|\theta_k)$ and generate $U \sim U(0,1)$
3: Let $\alpha = \min\left\{1, \exp\left\{\frac{f(\theta') - f(\theta)}{T_k}\right\}\right\}$.
4: If $U < \alpha$, then let $\theta_{k+1} = \theta'$
5:     Else $\theta_{k+1} = \theta_k$.
6: Update $T_{k+1}$
7: Store $\theta_{k+1}$ and $e^{f(\theta_{k+1})}$.
8: Return $\theta^* = \theta_{k^*}$ where $k^*$ is such that $k^* = \arg\max_k e^{f(\theta_k)}$
```

Thus, if the proposed value is such that $f(\theta') > f(\theta)$, then $\alpha = 1$ and the move is always accepted. The reason simulated annealing works is because when θ' is such that $f(\theta') < f(\theta)$, even then, the move is accepted with probability α.

Thus, there is always a chance to move out of local maximas.

Essentially, each θ_k is approximately distributed as $\pi_{k,T}$, and as $T_k \to 0$, $\pi_{k,T}$ puts more and more mass on the maximas, thus, θ_k will typically be getting increasingly closer to θ^*.

- Typically, $G(\cdot|\theta)$ is $U(\theta - r, \theta + r)$ or $N(\theta, r)$ which are both valid symmetrical proposals. The parameter r dictates how far/close the proposed values will be.
- T_k is often called the *temperature* parameter. A common value of $T_k = d/\log(k)$ for some constant d.