Missing Data

There are whole courses in treating missing data. In this week, I'm just going to show you one general approach called the EM algorithm because it heavily builds on what we already know (maximum likelihood and expectation).

Suppose that you have to estimate the mean of some data where one datum is missing. Since highschool (or before) you often just ignore it and take the mean of what's left as your estimate. Alternatively, you replace the datum with the average of the remaining data and take the average of everything (this happens to give you the same answer).

In this course, we know the average can be quite sophisticated when we use the expected value. Furthermore, if we have other data to take into account, we can take a conditional expected value (given the other data from the observation).

Consider the following example. We won't go into linear regression, but you should see what the relationship will be.

Example 1

Consider the data below. What is a good estimate for the mean of y?

x	y
1	9.94
2	19.98
3	30.16
4	40.01
5	50.01
6	60.17
7	70.05
8	79.87
9	
10	99.96

plug in the avg of remaining observed data?

The mean of the observed y is 51.1277778. Would it make sense to take the average of y by simply omitting the observed data, or to replace it with the average of remaining y? It should be obvious that $\mathrm{E}(Y|x) \approx 10x$ and it would make sense to use the value of 90. If we did that, we would obtain the estimate of the average of y as being 55.015, larger because we guessed the value of y would be larger than the mean of the rest, because of its corresponding value of x.

Alternatively, we could guessing the value of y by looking at similar values of x and maybe taking some kind of weighted average, which is called a nearest-neighbours approach. It's very similar to what we did above.

Censoring Data

Example 2

We want to estimate the average time we can use a light chain to hold a heavy load. Suppose we use 10 lengths of chains to hold heavy loads. Some fail (break) after a number of hours. We assume that at some point, each light chain will fail given enough time under load, but we only observed data (next table) up to 15 hours. Find the function that should be maximized to estimate θ and estimate θ

Statistically, we expect that some chains may last a long time but we can't wait until the last chain breaks; we have to end our experiment early. The data from the chains still holding the load have missing values associated with them, but we have still observed *some* information: their value (time until failure) is larger than the time we have been watching so far (15 hours). Such instances are called right-censored data (on a graph you know it's further to the right but you don't know where). You can also have left-censored data (the value is some unknown value below some threshold) or doubly-censored data (the value is outside of an interval).

Consider data from an exponential, right censored after 15 hours under load.

chainLabel	failureTime
1	10.12
2	6.92
3	
4	0.38
5	0.67
6	3.80
7	3.77
8	1.74
9	
10	0.35

The observed data points are all up to 15 (we observed failure within 15 hours). In our case have 2 censored data points so they would have failed some time after 15 hours and if we observed it, they would be at least 15.

Obviously, the average time can't just be from the observed value: we know that we would underestimate if we did that. There is a relationship between the data missing and its value. Therefore, these values are not missing (completely) at random, instead they are just missing at random.

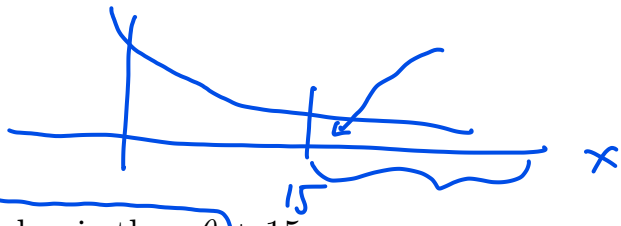
Let $X \sim \exp(\theta)$ where $f(x) = \frac{1}{\theta}e^{-\frac{x}{\theta}}$ represent the time until failure. We haven't observed (or not observed) data from this distribution.

The usual log-likelihood is $\ell(\theta) = -n\log(\theta) - \frac{1}{\theta}\sum(x_i)$ and it should be obvious our sufficient statistic is $\sum(x_i)$.

However, we can't differentiate yet as not all of the numbers in our log likelihood are proper numbers. In particular, some of the x_i are realized and some remain random/unobserved. Our likelihood shouldn't have random variables in it before we differentiate. Therefore, instead of maximizing $\ell(\theta)$, let us maximize $E(\ell(\theta))$ which replaces missing sufficient statistics with their expected values. In our case, it happens to be the same as replacing each missing value with its expected value. However, as we will see, that is not not the average of the data. Note that

$$\sum x = \sum_{i \text{ observed}} x_i + \sum_{j \text{ censored}} X_j$$

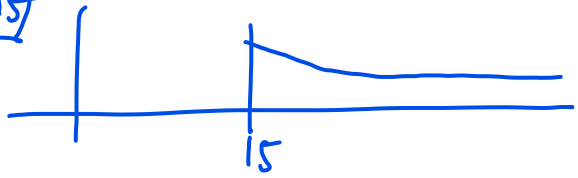
so we must replace $\sum_{j \text{ censored}} X_j$ with $\mathrm{E}\left(\sum_{j \text{ censored}} X_j\right) = \sum_{j \text{ censored}} \mathrm{E}(X_j)$. A caveat here is that we would of course think this is just θ but we know that each of this missing values is more than 15. So we really need $\mathrm{E}(X_j | X_j > 15)$.



Show that distribution of $X|X > 15$ is
$f(x|x > 15) = e^{\frac{15}{\theta}} \times \frac{1}{\theta} e^{-\frac{x}{\theta}}$ and its expected value is then $\theta + 15$

$f(x,\ x>15) = c \cdot f(x)$ $x>15$

let $Y = x|x>15$



$$\int_{15}^{\infty} c \cdot f(x)\,dx = 1$$

$$= \int_{15}^{\infty} c \cdot \frac{1}{\theta} e^{-x/\theta}\,dx = 1$$

$$= -c \cdot e^{-x/\theta}\Big|_{15}^{\infty} = 1$$

$$= 0 - (-c)\, e^{-15/\theta} = 1$$

$$c = e^{15/\theta}$$

Can show through integration by parts

$$E(X|x>15) = \int_{15}^{\infty} x \cdot e^{15/\theta} \cdot \frac{1}{\theta} e^{-x/\theta}\,dx$$

$$= \vdots$$

$$= 15 + \theta$$

You can easily show from stat2000 that the distribution of $X|X > 15$ is $f(x|x > 15) = e^{\frac{15}{\theta}} \times \frac{1}{\theta} e^{-\frac{x}{\theta}}$ and its expected value is then $\theta + 15$, either by manually integrating or using an online integrator. One important thing to notice here is that the expected value is a function of θ itself, which means if we put a guess in for the expected value, the value of θ will change and then our guess for the expected value itself should update. At the end, we will denote the current guess as $\theta^{(\text{old})}$ and $E(X|X > 15) = 15 + \theta^{(\text{old})}$.

Letting n_c represent the number of censored data points below,

$$\begin{aligned}
\ell(\theta) &= -n\log(\theta) - \frac{1}{\theta}\left(\sum_{i \text{ obs}} x_i + \sum_{j \text{ censored}} x_j\right) \\
\mathrm{E}(\ell(\theta)) &= -n\log(\theta) - \frac{1}{\theta}\left(\sum_{i \text{ obs}} x_i + \sum_{j \text{ censored}} \mathrm{E}(X|X > 15)\right) \\
&= -n\log(\theta) - \frac{1}{\theta}\left(\sum_{i \text{ obs}} x_i + n_c \mathrm{E}(X|X > 15)\right)
\end{aligned}$$

We now differentiate this function, which does not have random variables, to obtain a function that's like a score funtion

$$\begin{aligned}
\frac{\partial}{\partial\theta}\mathrm{E}(\ell(\theta)) &= -\frac{n}{\theta} + \frac{1}{\theta^2}\left(\sum_{i \text{ obs}} x_i + n_c \mathrm{E}(X|X > 15)\right) \\
&= \frac{1}{\theta^2}\left(-n\theta + n_c \mathrm{E}(X|X > 15) + \sum_{i \text{ obs}} x_i\right)
\end{aligned}$$

Setting this equal to zero gives

$$\hat{\theta} = \frac{1}{n}(n_c \mathrm{E}(X|X > 15) + \sum_{\text{obs}} x_i)$$

where $\mathrm{E}(X|X > 15) = 15 + \theta^{(\text{old})}$. Note that this literally replaces each missing datum with the expected value of a data point that is more than 15. In another distribution that does not have the memoryless property of the exponential, this expected value would be something else, but you can guess the estimate of $\hat{\theta}$ could otherwise be the same. Code for the EM is shown below.

```r
#x already defined
nc=sum(is.na(x));
sumObservedx = sum(x,na.rm=T)

theta = mean(x,na.rm=T) #initial guess is the mean of observeddata (underestimated)
print(theta)
## [1] 3.469507


for (i in 1:14){
  expectedvalue = 15+theta                          #e step - calculate expected value
  theta=1/n * (  nc*(expectedvalue)+sumObservedx  ) #m step - find the "MLE"
  print(theta)
}
## [1] 6.469507
## [1] 7.069507
## [1] 7.189507
## [1] 7.213507
## [1] 7.218307
## [1] 7.219267
## [1] 7.219459
## [1] 7.219497
## [1] 7.219505
## [1] 7.219507
## [1] 7.219507
## [1] 7.219507
## [1] 7.219507
## [1] 7.219507
```

In our example You can see that we reach a fairly stable value quickly. Before I deleted the data, the actual mean of the sample was 7.6419552 which the EM algorithm estimates better than the original estimate of only the observed data, 3.4695069.

Definition 1: EM algorithm

The **EM algorithm** is a method of finding MLEs in light of missing data. Rather than finding the maximum of the log-likelihood, $\ell(\theta)$, one finds the maximum of $\mathrm{E}(\ell(\theta))$, replacing the missing sufficient statistics in $\ell(\theta)$ with their expected values.

$\Rightarrow E(x_i^2) \neq (E(x_i))^2$

The function $\mathrm{E}(\ell(\theta))$ is typically denoted and referred to as $Q(\theta)$ or $Q(\theta, \theta^{(\text{old})})$ since the expected values themselves contain $\theta^{(\text{old})}$.

The **E-step** is creating the new function to find the maximum of, $Q(\theta, \theta^{(\text{old})}) = \mathrm{E}(\ell(\theta))$ instead of $\ell(\theta))$, and then

the **M-step** is finding the maximum of Q, treating any expected values (and the old estimates of parameter) as constants.

Notes

- The name comes from taking an expected value and maximizing, rather than maximizing the likelihood directly
- Expected values may be challenging, for example if we required $\mathrm{E}(X_i^2)$ we would want the second moment, not just the square of the first moment.
- Often, the expected value depends on the value of θ, and maximized value of θ depends on expected values, and no way to solve for θ. It can be shown that with a good starting value, the algorithm can iterate back and forth until convergence.

Example 3: Mixture Model

Someone observes data coming from one of two different Poisson processes, but doesn't know which observation belongs to which process. Each process corresponds to a parameter $\lambda_j, j \in \{1, 2\}$ so the distribution to use in the likelihood function is either $P(X = x_i|\lambda_1) = \frac{1}{x_i!}e^{-\lambda_1}\lambda_1^{x_i}$ or $P(X = x_i|\lambda_2) = \frac{1}{x_i!}e^{-\lambda_2}\lambda_1^{x_i}$ Find the EM algorithm estimates for each λ_j. Hint: introduce z_i an indicator for whether x_i belongs to the first coin or not.

We will assume that the coin flip can only correspond to one coin, and hence, one pmf, so we can say that if z_i is the indicator for which process it came from ($z_i = 1$ if it came from process 1 and 0 otherwise). This gives a mass function often

$$f(x_i) = (f_0(x_i))^{z_i} (f_1(x_i))^{1-z_i}$$

where z_i is a (missing) value denoting which coin the flip came from. The likelihood is then

$$\mathcal{L} = \prod_i f(x_i) = \prod_i (f_0(x_i))^{z_i} (f_1(x_i))^{1-z_i}$$

$$\ell(\lambda_1 \lambda_2) = \sum_i z_i \Big(-\lambda_1 + x_i \log \lambda_1 - \log(x_i!) \Big) + (1 - z_i) \Big(-\lambda_2 + x_i \log \lambda_2 - \log(x_i!) \Big)$$

$$\frac{\partial}{\partial \lambda_1} \ell = \sum_i z_i \Big(-1 + x_i \lambda_1 \Big)$$

Setting this equal to zero yields $\hat{\lambda}_1 = \frac{\sum z_i x_i}{\sum z_i}$. Doing the same for λ_1 yields the similar estimate $\hat{\lambda}_1 = \frac{\sum (1-z_i) x_i}{\sum (1-z_i)}$. This is satisfying as for for the binary versions of the variables, these are the average of the data coming from each process.

The issue with the above is that z can be considered random and so our derivatives are not appropriate unless we deal with expected values for the missing sufficient statistics.

$$Q = \ell(\lambda_1, \lambda_2) = \sum_i \mathrm{E}(z_i) \Big(-\lambda_1 + x_i \log \lambda_1 - \log(x_i!) \Big) + (1 - \mathrm{E}(z_i)) \Big(-\lambda_2 + x_i \log \lambda_2 - \log(x_i!) \Big)$$

The expected value of this indicator will come from Bayes' theorem (we will skip the derivation in this course):

$$\mathrm{E}(z_i | \lambda_1^{(\text{old})}, \lambda_2^{(\text{old})}) = \frac{\overline{z} f_1(x_i)}{\overline{z} f_1(x_i) + (1 - \overline{z}) f_2(x_i)}; \qquad \overline{z} = \frac{1}{n} \sum_i z_i$$

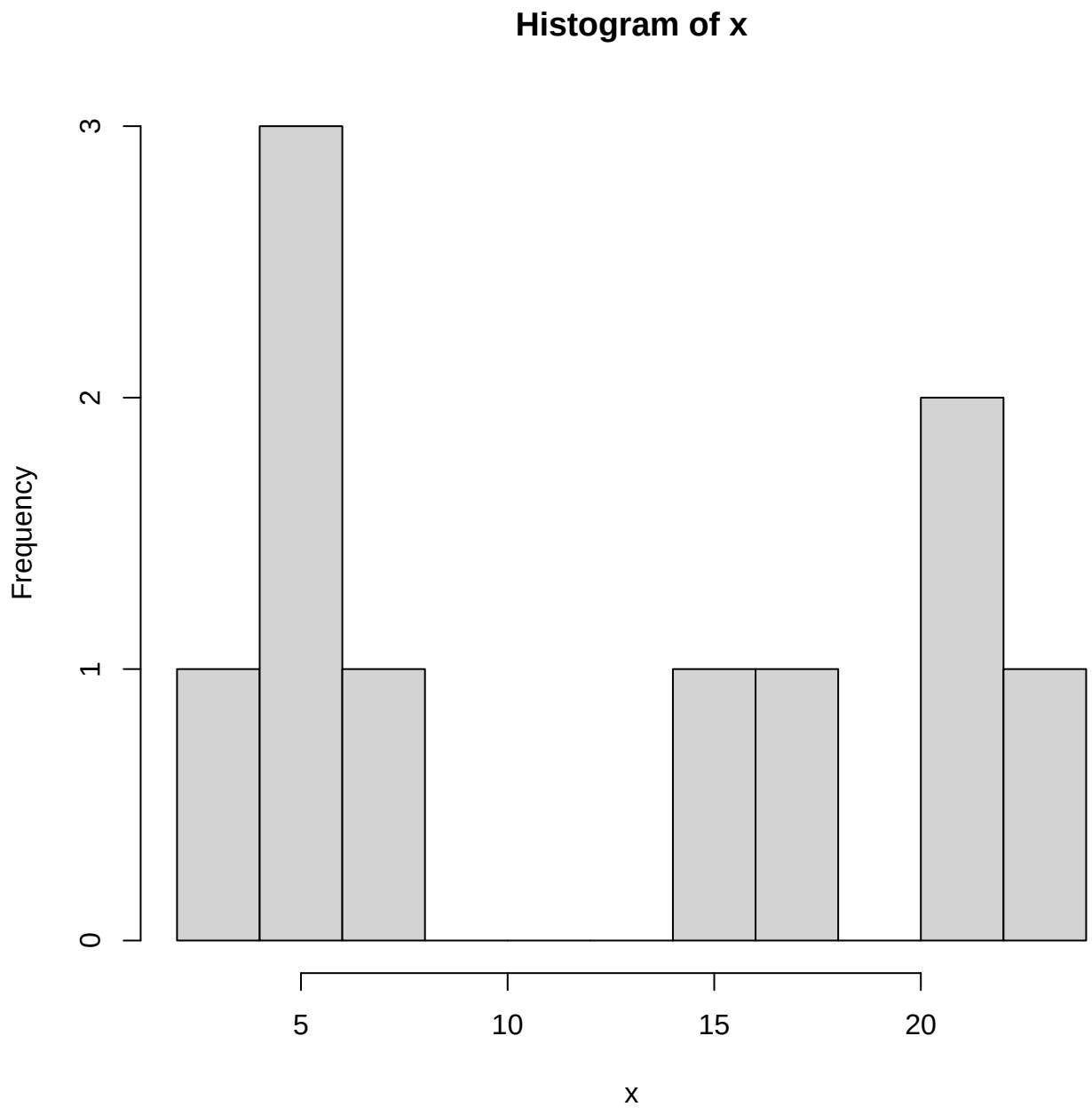
Remember that $\lambda_1^{(\text{old})}$ and $\lambda_2^{(\text{old})}$ are inside of f_1 and f_2, respectively, which puts them inside of $\mathrm{E}(z_i)$, which is how they exist inside of Q. It's much too long a line to put it all together and I think it makes it more confusing to read.

For the algorithm, we make some guess about z (even randomly assign) and then estimate each λ_j. From that, we recalculate $\mathrm{E}(z_i)$ for every observation and use

that value in place of z_i in the estimates of the λ_js, that is: $\hat{\lambda}_1 = \frac{\sum \mathrm{E}(z_i)x_i}{\sum \mathrm{E}(z_i)}$ and $\hat{\lambda}_1 = \frac{\sum (1-\mathrm{E}(z_i))x_i}{\sum (1-\mathrm{E}(z_i))}$. This would repeat until convergence. On a test or a quiz, I would only have you write the updates for the parameter estimates and expected values (as above). The rest is trivial computer coding. For your interest, though, here's an example in R.

```
set.seed(1234)
x1=rpois(5,lambda=5)
x2=rpois(5,lambda=20)
x=c(x1,x2)#put the two collections of data together
n=length(x)

#this generates data from pois(5) and pois(20)
hist(x,breaks=10)
```




```
#you can see there are two subgroups in the histogram, but the algorithm doesn't know tha
#the algorithm also doesn't know the order of data

#create poisson pmf
f=function(x,lambda)exp(-lambda)*lambda^x/factorial(x)

#randomly create group memberships and first estimates of parameters
z=runif(n)
```

```r
lambda1 = sum(z*x)/sum(z)
lambda2 = sum((1-z)*x)/sum(z)

updatez = function(lambda1,lambda2){ # helper function
 meanz = mean(z)
 meanz*f(x,lambda1) / (meanz*f(x,lambda1) + (1-meanz)*f(x,lambda2)  )
}

for (i in 1:10){
  z=updatez(lambda1,lambda2)
  lambda1 = sum(z*x)/sum(z)
  lambda2 = sum((1-z)*x)/sum(z)
  print(c(lambda1,lambda2))
}
## [1] 8.492906 9.799276
## [1]  9.198096 13.342922
## [1]  6.029193 19.642364
## [1]  5.211877 19.998477
## [1]  5.201286 20.028233
## [1]  5.201256 20.027825
## [1]  5.201255 20.027836
## [1]  5.201255 20.027836
## [1]  5.201255 20.027836
## [1]  5.201255 20.027836


c(mean(x1),mean(x2))#actual means of the poisson
## [1]  5.2 20.0


#algorithm guess of the probability observation belonged to group 1
round(z,4)
##  [1] 1.0000 0.9988 0.9997 0.9988 0.9954 0.0000 0.0012 0.0000 0.0003 0.0000
```

The interesting thing about this mixture model example (subgroups) is that there are two types of data: the observed data and the values we typically perform math and inference on (the x) and the group class label that we created (the z). Notice that the *entire* z variable is completely missing (literally half our 'data'), but we still correctly determined the mean of each group, and we estimated which observation came from which process very well (the first five z are nearly 1, which mean they're in one group, and the next five z are nearly 0, which means they're in the other

group). E(z_i) is a posterior probability of belonging to the first group. When data are not so clearly separated, the values of the z are correspondingly less clearcut but surprisngly, parameters are typically still estimated fairly well. You can try it yourself by repeating the above with a larger value for the first mean.