BAS 471 Spring 2023 Homework on Unit 8 - Estimating Parameters of Probability Distributions

Ryan Curling

4/20/2023

Note: these are your homework problems

Reminder about collaboration policy. You can develop a common set of R code with you and your friends. However, anything that is written interpretation, i.e., anything that follows a **Response:** needs to be written up in your own words. Homeworks that look to be near copy/pastes of each other will receive substantially reduced credit.

Note: For this assignment, you'll be required to have a few screenshots from Wolfram Alpha as well as a few pictures of work you are to perform on pen and paper. For these parts, pasting a photo of work into your Word document after it knits is fine!

Question 1 (fitdist to the rescue)

You'll need the data in the `BAS471-HWonUnit8F22.RData` saved environment file to complete this problem. You know the drill:

- Make sure the .RData file is in your 471 folder
- Make sure to do a Save As on this .Rmd file and forcibly put it in your 471 folder
- Make sure your working directory is your 471 folder too (Session, Set Working Directory)

```
load("BAS471-HWonUnit8S23.RData")
```

Let's find the MLEs for a few niche distributions where the mathematics of deriving the general expressions for the MLE would have been extremely difficult (if not impossible), but finding a numerical value for the MLE for a given dataset is easy with `fitdist`.

For each distribution:

1) Define `dcustom` (you can have a different name if you want, but it does have to start with `d`), taking as its first argument `x` and 2nd, 3rd, etc. arguments the names of the parameters of the probability distribution.

2) Show that you pass the sanity check for `dcustom` by evaluating `dcustom` at the given value of x and set of parameters (the value of the PMF or PDF should be printed out and knitted into the document).

3) Run `fitdist`, left-arrowing the result into `FIT`. Then, print out the contents of `FIT` to the screen. Remember to ignore warnings of `The pcustom function must be defined` since we don't actually need `pcustom` to get the MLE estimates (it's only needed for QQ plots).

4) Make a histogram of the data with `hist` (add in `freq=FALSE`), then use `curve` to overplot the fitted PDF. Although the quality of fit is better assessed with the QQ plot, this at least establishes that the fit is decent! For `curve`, the syntax should be:

- one parameter: `curve(dcustom(x,FIT$estimate[1]),add=TRUE,col="red")`
- two parameters:
  `curve(dcustom(x,FIT$estimate[1],FIT$estimate[2]),add=TRUE,col="red")`
- three parameters:
  `curve(dcustom(x,FIT$estimate[1],FIT$estimate[2],FIT$estimate[3]),add=TRUE,col="red")`

This way, the code automatically extracts the best fitting parameters (the MLEs) from the fit!

Refer to examples in the in-class Worksheet: `BAS471-Unit8-Estimation.Rmd` if you need a reminder for how to do any of these things!

a) The notorious zoo lacks distributions that are skewed right (a useful feature to have for distributions in business analytics) while allowing negative values. Such a distribution may be useful for modeling checking account balances (where some people are temporarily overdrawn and have negative balances).

One distribution does allow for both. Let's call it the "Jackson" distribution (not its official name). The Jackson distribution has two flavors (mouse over to see the equation; if the equation doesn't pop up, try restarting RStudio and/or clicking the double up/down arrows on the far right side of the equation box to expand it):

$$Flavor\ 1 \qquad f(x) = \frac{ae^{-x}}{\left(1+e^{-x}\right)^{a+1}} \qquad -\infty < x < \infty$$

$$Flavor\ 2 \qquad f(x) = \frac{abe^{-bx}}{\left(1+e^{-bx}\right)^{a+1}} \qquad -\infty < x < \infty$$

You'll notice that Flavor 1 is a special case of Flavor 2 when b=1.

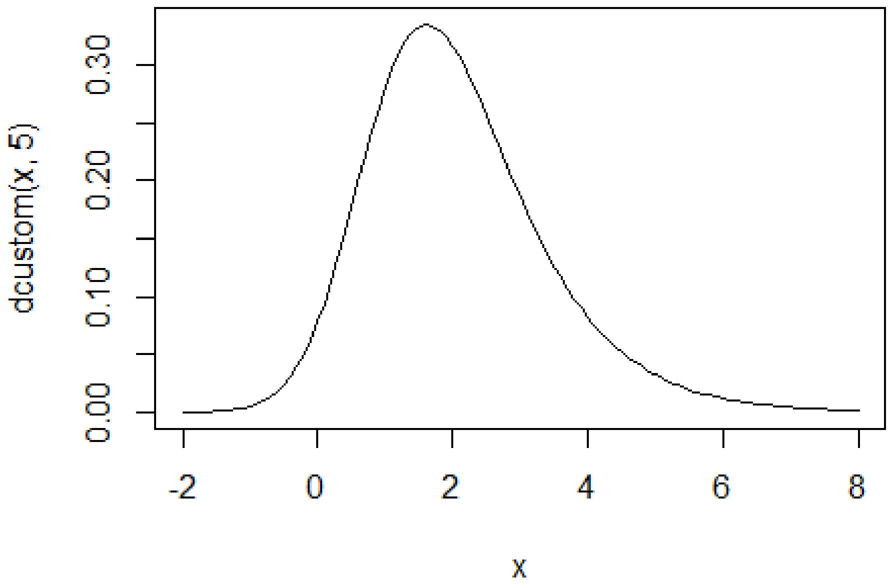
a1) Find the MLE for a (Flavor 1) using the values in `jackson.data`. Sanity check: close to 3.9. Try a starting guess of 2.

a2) Find the MLEs for a and b (Flavor 2) using the values in `jackson.data`. Sanity check: close to 7.9 and 1.8, respectively. Try starting guesses of 4 for both parameters.

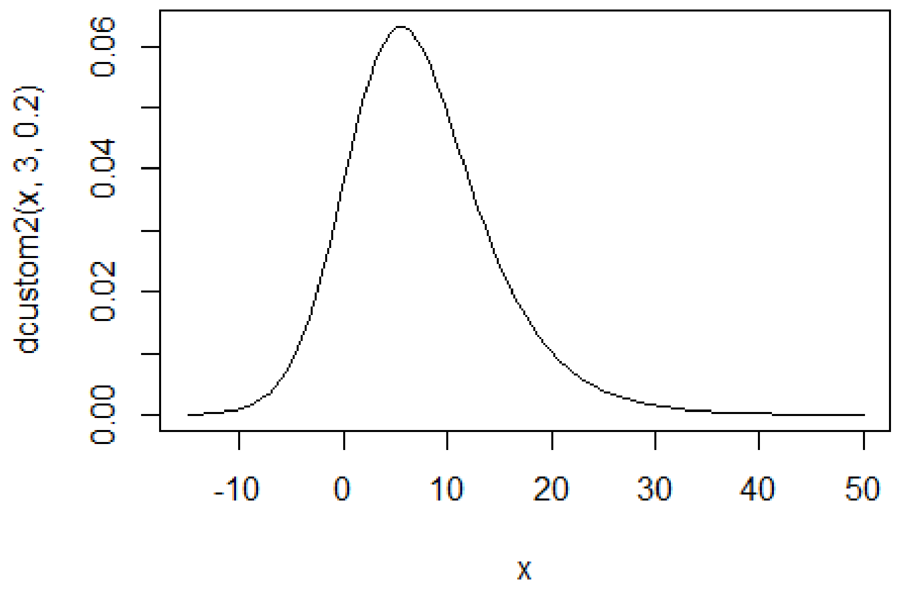
a3) Look at the AICs of the fits and comment on which is preferred (remember, if the AICs indicate they are comparable, the suggestion is to pick the simpler model)

Response:

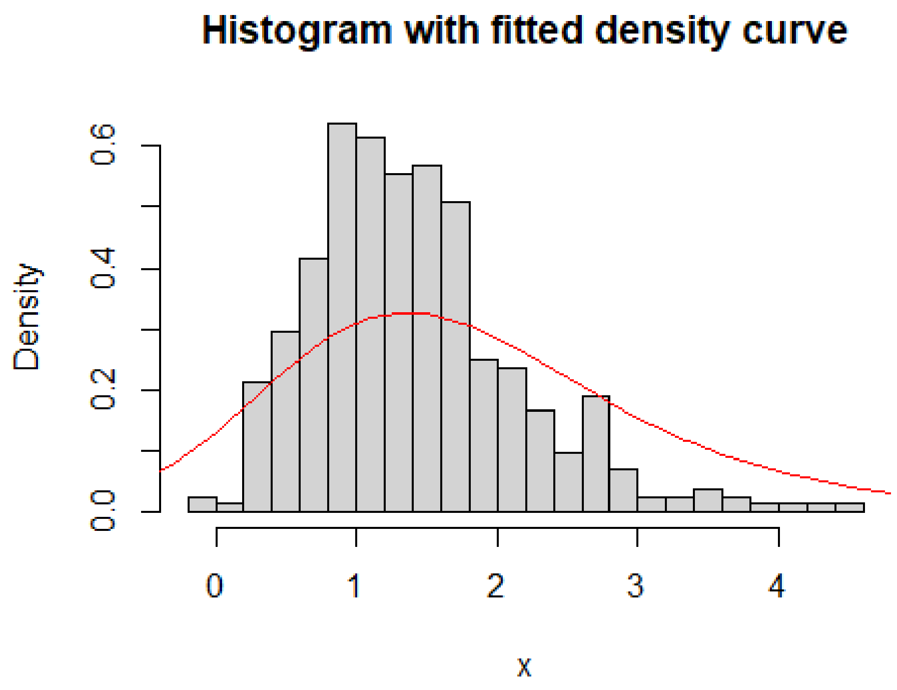
```
#Part a1 (Flavor 1); only 1 parameter a
dcustom <- function(x,a) {
  numerator <- a * exp(-x)
  denom <- (1+exp(-x))^(a+1)
  return(numerator/denom)
}
#Uncomment the sanity check to show you pass it
dcustom(-0.2,a=4)
## [1] 0.09031967
#0.09031967

#Uncomment to see a curve if you're interested; this doesn't have to be in
writeup
curve(dcustom(x,5),from=-2,to=8)
```



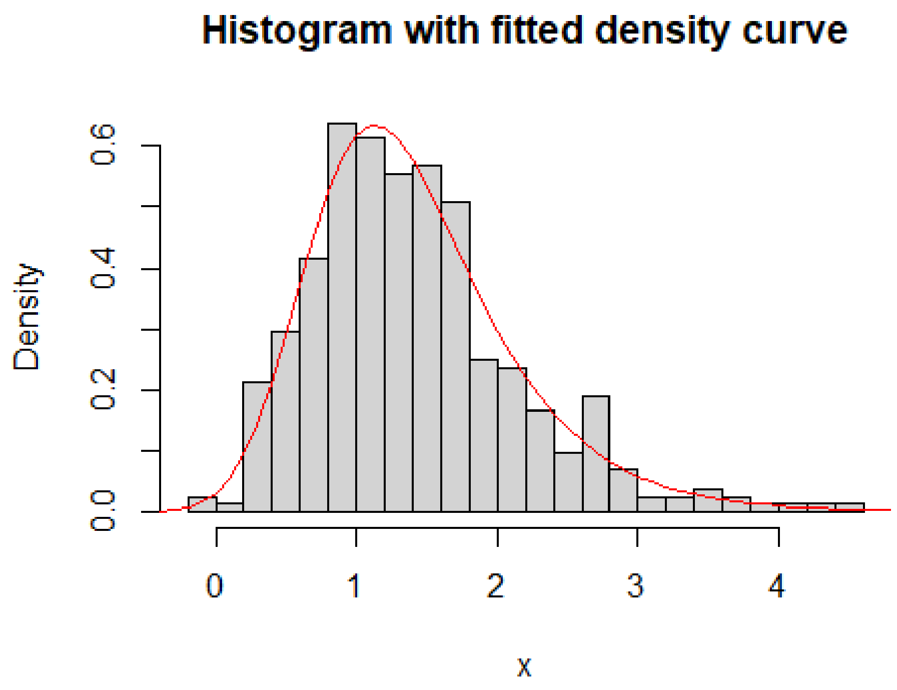
```
#Finding MLE for jackson.data

fita1 <- fitdist(jackson.data, dcustom, start = list(a = 2), method = "mle")
## Warning in fitdist(jackson.data, dcustom, start = list(a = 2), method =
"mle"):
## The pcustom function must be defined
print(fita1)
## Fitting of the distribution ' custom ' by maximum likelihood
## Parameters:
##   estimate Std. Error
## a 3.911703  0.1901935



#Part a2 (Flavor 2); two parameters a and b

dcustom2 <- function(x, a, b){
  numerator <- a * b * exp(-b*x)
  denominator <- (1 + exp(-b*x))^(a+1)
  return(numerator/denominator)
}

#Uncomment the sanity check to show you pass it
dcustom2(-0.2,a=8,b=4)
## [1] 0.001884354
#0.001884354

#Uncomment to see a curve if you're interested; this doesn't have to be in
writeup

curve(dcustom2(x,3,.2),from=-15,to=50)
```



```
fita2 <- fitdist(jackson.data, dcustom2, start = list(a=4, b=4),
method="mle")
## Warning in fitdist(jackson.data, dcustom2, start = list(a = 4, b = 4),
method =
## "mle"): The pcustom2 function must be defined
print(fita2)
## Fitting of the distribution ' custom2 ' by maximum likelihood
## Parameters:
##   estimate Std. Error
## a 7.934549 0.54995029
## b 1.833697 0.06410684

#Histogram plot dcustom

hist(jackson.data, freq = FALSE, breaks = 30, xlab = "x", main = "Histogram
with fitted density curve")
curve(dcustom(x, fita1$estimate[1]), from = -2, to = 8, add = TRUE, col =
"red")
```



```r
#Histogram plot dcustom2

hist(jackson.data, freq = FALSE, breaks = 30, xlab = "x", main = "Histogram
with fitted density curve")
curve(dcustom2(x, fita2$estimate[1], fita2$estimate[2]), from = -2, to = 8,
add = TRUE, col = "red")
```



```
#Checking AICs

fita1$aic
## [1] 1105.743
fita2$aic
## [1] 883.489
```

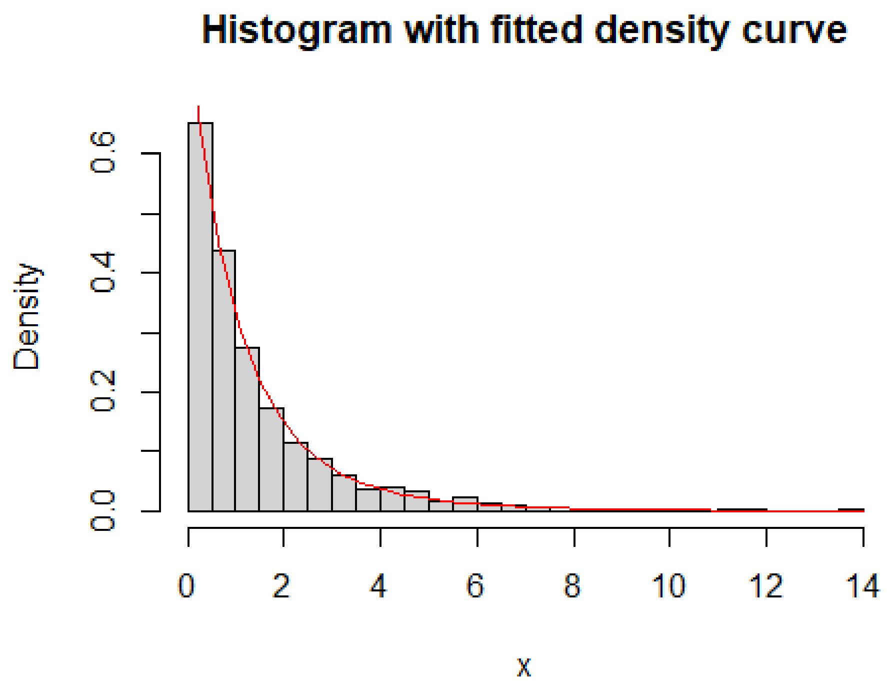
b) The Lomax distribution is a heavy-tail probability distribution used in business, economics, actuarial science, queueing theory, and Internet traffic modeling. Mouse over to see the equation; if the equation doesn't pop up, try restarting RStudio and/or clicking the double up/down arrows on the far right side of the equation box to expand it.

$$f(x) = \frac{ab^a}{(x+b)^{a+1}} \qquad x > 0$$

Find the MLEs for a and b using the values in `lomax.data`. Sanity check: the MLE for a is close to 6.0 while the MLE for b is near 7.2. Try starting guesses of 7 for each parameter.

```
dcustom <- function(x,a,b) {
  num <- a * b^a
  dem <- (x+b)^(a+1)
  return(num/dem)

  }
```

```r
#Uncomment the sanity check to show you pass it

dcustom(2,a=3,b=2.3)
## [1] 0.1067655
#0.1067655

fit1b <- fitdist(lomax.data,dcustom, start=list(a=7,b=7),method = "mle")
## Warning in fitdist(lomax.data, dcustom, start = list(a = 7, b = 7), method
=
## "mle"): The pcustom function must be defined

#MLE for a and b parameters

fit1b$estimate
##        a        b
## 6.027370 7.220715

#Histogram and curve plot
hist(lomax.data, freq = FALSE, breaks = 30, xlab = "x", main = "Histogram
with fitted density curve")
curve(dcustom(x,fit1b$estimate[1],fit1b$estimate[2]),add=TRUE,col="red")
```



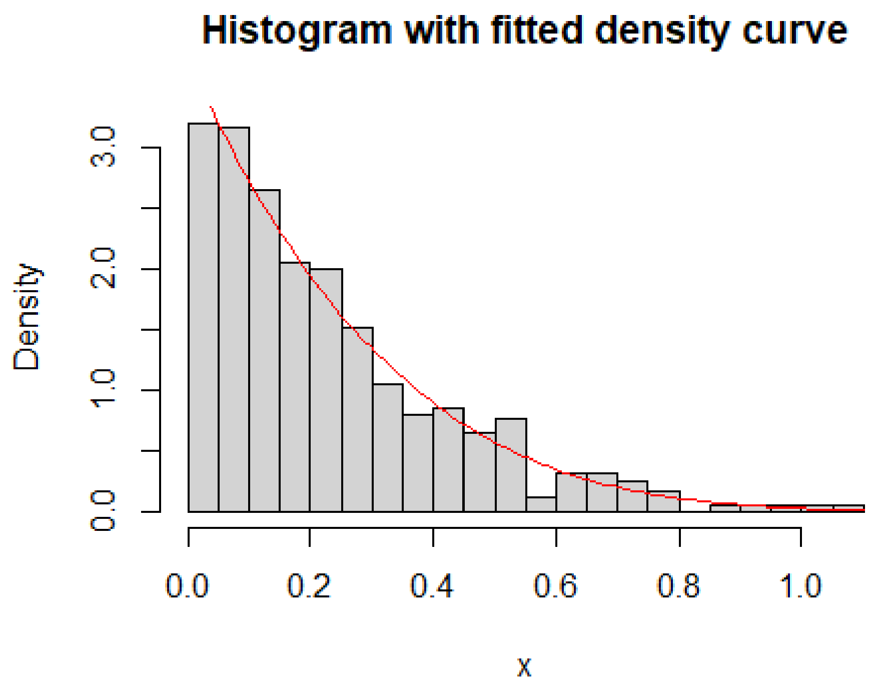
c) The “Cannon” distribution (not its real name) can serve as an alternative to the Weibull when modeling times to failure (e.g., of lightbulbs, jet engines, computer parts, etc.). The Weibull has the ability to model either an increasing or decreasing instantaneous failure rate, but not both. The Cannon allows cases where the failure rate is high at early times, small for mid times, and high at late times (a U shape). This could be useful for modeling the lifetimes of electrical components where parts often fail in the first few moments of operation, but if the part manages to work “for a while”, chances are it will continue working for a very long time.

The Cannon has three parameters with the following PDF. Mouse over to see the equation; if the equation doesn’t pop up, try restarting RStudio and/or clicking the double up/down arrows on the far right side of the equation box to expand it.

$$f(x) = \left(a + bx + cx^2\right)e^{-\left(ax+bx^2/2+cx^3/3\right)} \qquad x > 0$$

Find the MLEs for a, b, and c using the values in `cannon.data`. Sanity check: close to 3.74, 2.4, and 2.8, respectively. Try starting guesses of 3.5 for each parameter.

```
dcustom <- function(x,a,b,c) {

  part1 <- a + b*x + (c*x^2)
  part2 <- exp(-a*x - b*x^2/2 - c*x^3/3)
  return(part1 * part2)

}

#Uncomment the sanity check to show you pass it

dcustom(1.2,4,2,1)
## [1] 0.008593397
#0.008593397

fit2c <- fitdist(cannon.data,dcustom,start = list(a=3.5,b=3.5,c=3.5),method =
"mle")
## Warning in fitdist(cannon.data, dcustom, start = list(a = 3.5, b = 3.5, :
The
## pcustom function must be defined

print(fit2c)
## Fitting of the distribution ' custom ' by maximum likelihood
## Parameters:
##   estimate Std. Error
## a 3.748258  0.3802503
## b 2.406161  3.2530206
## c 2.798363  5.1105452

#Histogram and curve plot
hist(cannon.data, freq = FALSE, breaks = 30, xlab = "x", main = "Histogram
with fitted density curve")
curve(dcustom(x,fit2c$estimate[1],fit2c$estimate[2],fit2c$estimate[3]),add=TR
UE,col="red")
```



Question 2 (math time)

Consider the following simple model for how knowledge of a new product spreads by "word of mouth" in a small town.

- At first, only one person knows about the new product. This individual tells a random number of new individuals (0, 1, 2, 3, …). On average, the number of individuals this person tells is a little less than 1 (e.g., there's a large probability that the person tells no one).
- Each new person who has now heard of the product tells an additional random number of new individuals.
- The PMFs describing the random number of new individuals told are all identical.

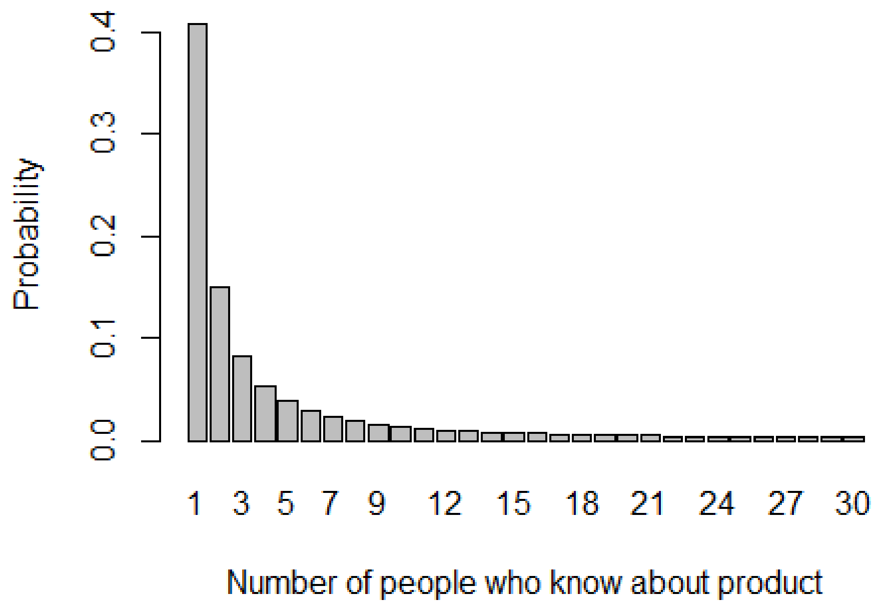
Eventually, the spread by "word of mouth" will stop. Let X be the *total* number of people who hear about the product. Possible values are x = 1, 2, 3, …. The probability mass function (PMF) of X is (mouse over to see the equation; if the equation doesn't pop up, try restarting RStudio and/or clicking the double up/down arrows on the far right side of the equation box to expand it):

$$P(X = x) = \frac{e^{-kx}(kx)^{x-1}}{x!} \qquad x = 1, 2, 3, \ldots$$

The value of k is the “shape” parameter of the distribution and controls what the PMF looks like.

If $k = 0.9$, the following vectors x and PMF give one example of the PMF.

```
x <- 1:30; k <- 0.9; PMF <- exp(-k*x)*(k*x)^(x-1)/factorial(x)
barplot(PMF,names.arg=x,ylab="Probability",xlab="Number of people who know
about product")
```



Let’s imagine that you do not know k, but you want to estimate it from a random sample of data values.

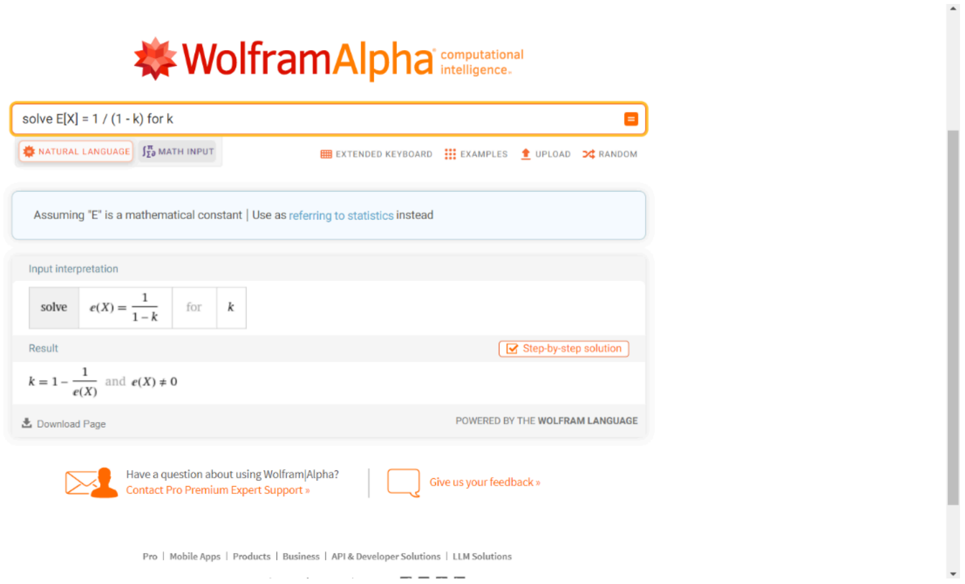
- The number of people that heard about product 1 might have been $x_1 = 12$
- The number of people that heard about product 2 might have been $x_2 = 32$
- The number of people that heard about product 3 might have been $x_3 = 26$
- etc. A total of n newly released products are considered.

a) Using some advanced math tricks (Wolfram-Alpha doesn’t work well with this PMF), it can be shown that the expected value of X is:

$$E[X] = \frac{1}{1-k}$$

Find the Method of Moments estimator for k. In other words, what function of $\bar{X}$ does MoM suggest we use to estimate k?

Insert Screenshot from Wolfram-Alpha, a photo of your work on scratch paper, or type out by hand (not too hard of an equation to solve)


solve E[X] = 1 / (1 - k) for k

Assuming "E" is a mathematical constant | Use as referring to statistics instead

Input interpretation

solve $e(X) = \frac{1}{1-k}$ for k

Result

$k = 1 - \frac{1}{e(X)}$ and $e(X) \neq 0$

NATURAL LANGUAGE
MATH INPUT
EXTENDED KEYBOARD
EXAMPLES
UPLOAD
RANDOM
Step-by-step solution
Download Page
POWERED BY THE WOLFRAM LANGUAGE
Have a question about using Wolfram|Alpha?
Contact Pro Premium Expert Support »
Give us your feedback »
Pro | Mobile Apps | Products | Business | API & Developer Solutions | LLM Solutions

b) Let $\{X_1, X_2, ..., X_n\}$ be a random sample from this PMF (e.g. X_1 is the number of people who heard about product 1, X_2 is the number of people who heard about product 2, etc.; each result is independent). On a piece of scratch paper, write out the likelihood of a random sample (the equations for the PMF multiplied by each other), then use some algebra to group terms, combine powers and/or exponents, and simplify to show that the likelihood is (mouse over to see the equations; if the equation doesn't pop up, try restarting RStudio and/or clicking the double up/down arrows on the far right side of the equation box to expand it):

$$L = \frac{e^{-kS} \cdot k^{S-n} \cdot P}{Q}$$

where (omitted subscripts and superscripts)

$$S = \sum x_i$$

$$P = \prod x_i^{x_i - 1}$$

$$Q = \prod x_i!$$

Response: Insert a screenshot of your work on scratch paper showing how to get the likelihood

c) Maximize the likelihood in Wolfram to find the expression for the MLE (it will involve S but not P or Q). Translate into English (e.g., ``two times the sample average") what the MLE says to do with the data (i.e., the "simple rule") to find the MLE for r.

Response: Insert Screenshot from Wolfram-Alpha

d) A big theme of BAS 471 is that *you* get to decide which of two things you want to be good at: math or programming. Most of us choose the latter! While `fitdist` doesn't provide us the "simple rule" of what to do with the data to find the MLE, it at least gives us an answer for the MLE on a case-by-case, dataset-by-dataset basis.

Define `dcustom` with arguments `x` and `k` (you can copy/paste the formula for the PMF in the first R chunk), then run `fitdist` to find the MLE for the data stored in `word.of.mouth` (use an initial guess of k=0.6). Remember, you can ignore the Warning `The pcustom function must be defined`.

Then, show that the value of the MLE from `fitdist` (close to 0.8) is essentially identical to the value you get when you plug in the relevant numbers into the equation for the MLE in part (c). Note: `word.of.mouth` lives in the saved environment file `BAS471-HWonUnit8S23.RData`. The code for loading that into the environment is in the code chunk at the beginning of the document and reproduced here.

```
load("BAS471-HWonUnit8S23.RData") #Instruction on how to load at top of Q1


# Define the custom distribution function
dcustom <- function(x, k) {
  exp(-k * x) * (k * x)^(x - 1) / factorial(x)
}

dcustom(12,.88)  #This serves as a sanity check to make sure your dcustom is
correct
## [1] 0.009858626
#0.009858626

# Fit the distribution using fitdist
fit2d <- fitdist(word.of.mouth,dcustom, start = list(k = 0.6), method="mle")
## Warning in fitdist(word.of.mouth, dcustom, start = list(k = 0.6), method =
## "mle"): The pcustom function must be defined

# Print the result
print(fit2d)
## Fitting of the distribution ' custom ' by maximum likelihood
## Parameters:
```

```
##    estimate Std. Error
## k 0.8026314 0.04595847
```

Question 3 (math time)

Planned obsolescence describes the practice of designing products to break quickly or become obsolete in the short to mid-term. The general idea behind this is to encourage sales of new products and upgrades, a practice that has been banned in some countries.

Imagine that an unscrupulous maker of printer ink cartridges has made it so that after `q` refills, the cartridge becomes unusable. It's perfectly possible for the cartridge to become unusable before refill `q` due to natural causes.

Let `X` be the number of refills before the cartridge becomes unusable. The possible number of refills before the cartridge becomes unusable are x = 0, 1, 2, …, q-1, q.

You believe that `X` has the PMF (mouse over to see the equation; if the equation doesn't pop up, try restarting RStudio and/or clicking the double up/down arrows on the far right side of the equation box to expand it):

$$P(X = x) = \frac{1}{2-\frac{1}{2^q}} \cdot \frac{1}{2^x} \qquad x = 0, 1, 2, ..., q$$

Your goal is the find the maximum likelihood estimator of `q`. The standard procedure of using calculus (Wolfram-Alpha or R) to maximize the likelihood won't work here! Whenever the value of x depends on the unknown parameter that we are trying to estimate (`q` here), the likelihood might peak at a place where the slope of the tangent line *isn't* 0 (which is what Wolfram-Alpha and R try to find). Let's use logic instead to find the MLE!

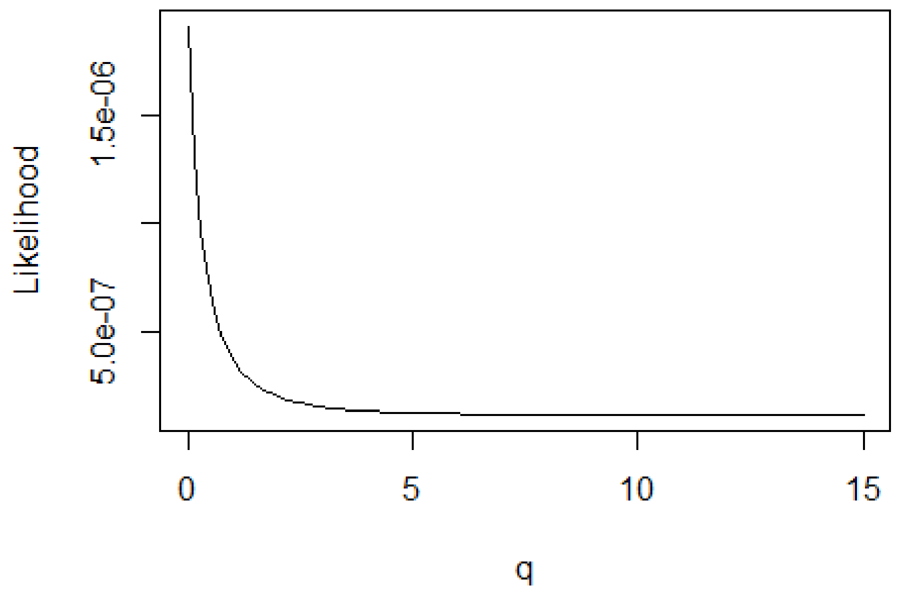
a) You buy four cartridges. The observed number of refills before they become unusable are $x_1 = 5, x_2 = 10, x_3 = 4, x_4 = 0$. Show that the likelihood of this sample is:

$$L = \frac{1}{2^{19}} \cdot \frac{1}{\left(2-\frac{1}{2^q}\right)^4}$$

Response: Insert a screenshot of your work on scratch paper showing how to get the likelihood

b) We can plot the likelihood of the data vs. `q` to help find the MLE. The chunk below plots the likelihood for values of q from 0 to 15 to give you an idea of what the likelihood looks like (you can expand the `to` and `from` arguments if you wish to see more).

```
#Remember curve always needs you to write the function in terms of x, even
though our function is of q
curve( 1/2^19 *1/(2-1/2^x)^4, from=0,to=15,xlab="q",ylab="Likelihood")
```



Let's use the shape the likelihood function to help choose q.

Response: Replace the "blank" with either the word "SMALL" or "LARGE". To maximize the likelihood, we need to choose q as _____ as possible.

c) The observed data is $x_1 = 5, x_2 = 10, x_3 = 4, x_4 = 0$. Certain values of q make the probability model unable to produce the collected value. Thus, logic dictates that there is a limit on how small or how large q can be. For example, we can outright eliminate the possibility that q equals 9.8 (think why that is). Thus, determine the MLE for q and explain your reasoning. In general if $x_1, x_2, ..., x_n$ is a random sample from this PDF, what will be the "formula" for the MLE of q (i.e., what needs to be calculated from the dataset to get the MLE)?

Response:

Question 4

The binomial distribution tells us the number of "successes" in a sequence of n independent trials, where each trial has a probability p of success. For example, if a customer is sent n special coupons, and each coupon has an independent probability p of being redeemed, then the total number of coupons a customer redeems would have a binomial distribution.

But what if we wanted to model the number of coupons used among coupon-redeemers (i.e., only those customers that have redeemed at least one coupon)? A "zero-truncated Binomial distribution" may work!

The zero-truncated Poisson gives a good model when a random quantity *would* be Poisson, but zero is not a possible value (e.g. number of items in shopping carts at check out). The zero-truncated Binomial gives a good model when the random quantity *would* be Binomial, but zero is not a possible value (e.g. number of coupons out of 15 that are used among people who have redeemed at least one).

With some math, it's possible to show that the PMF of a zero-truncated Binomial is:

$$P(X = x) = \frac{1}{1-(1-p)^n} \cdot \frac{n!}{x!(n-x)!} p^x (1 - p)^{n-x} \qquad x = 1, 2, ..., n - 1, n$$

Wolfram-Alpha (`sum x * 1/(1-(1-p)^n) * n!/(x!(n-x)!) p^x *(1-p)^(n-x) from x=1 to x=n assuming 0 < p < 1`) shows that the equation for the expected value is (mouse over to see the equation; if the equation doesn't pop up, try restarting RStudio and/or clicking the double up/down arrows on the far right side of the equation box to expand it):

$$\mu = E[X] = \sum x \cdot P(X = x) = \sum_{i=1}^{n} x \cdot \frac{1}{1-(1-p)^n} \cdot \frac{n!}{x!(n-x)!} p^x (1 - p)^{n-x} = \frac{np}{1-(1-p)^n}$$

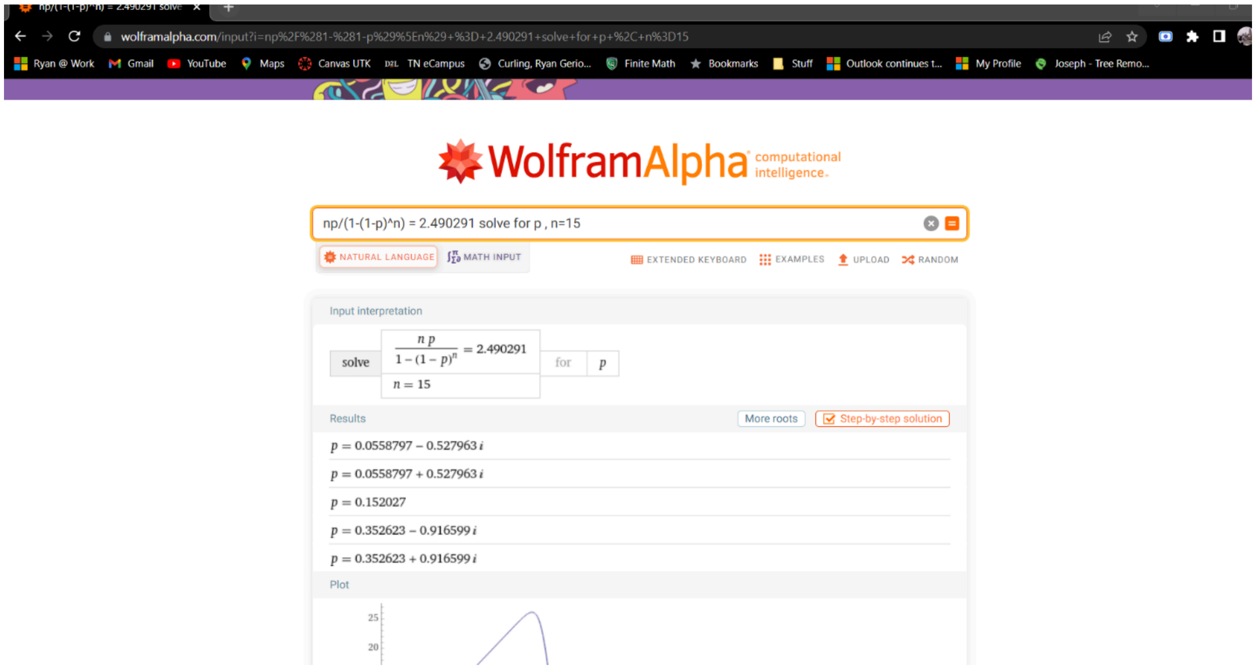
For example, a zero-truncated Binomial with $n = 10$ and $p = 0.2$ would have an average value of `10*.2/(1-(1-.2)^10)` = 2.24.

The data in the vector `coupons` (in the `BAS471-HWonUnit8F22.RData` file) contains the number of coupons (out of $n = 15$) that 206 coupon-redeeming shoppers of Kroger ended up using. You'll see that the average is close to 2.49.

Assume that the data comes from *some* zero-truncated Binomial distribution with $n = 15$, we just don't know what "flavor" it is (i.e., its value of p).

a) Find the Method of Moments estimator for p by having Wolfram solve the equation $E[X] = X^{-}$ for p. You'll need to replace X^{-} with the *exact* sample average from the data in this equation (measured in R), then have Wolfram solve the equation for p (replace n with 15). Include your screenshot from Wolfram-Alpha. Sanity check: a little above 0.15.

Response: Screenshot from Wolfram


wolframalpha.com/input?i=np%2F%281-%281-p%29%5En%29+%3D+2.490291+solve+for+p+%2C+n%3D15

np/(1-(1-p)^n) = 2.490291 solve for p , n=15

Input interpretation

solve $\frac{n\,p}{1-(1-p)^n} = 2.490291$, $n = 15$ for p

Results

$p = 0.0558797 - 0.527963\,i$

$p = 0.0558797 + 0.527963\,i$

$p = 0.152027$

$p = 0.352623 - 0.916599\,i$

$p = 0.352623 + 0.916599\,i$

Plot

```
#Get the average value of the data in coupons; needed for Wolfram
xbar <- mean(coupons)
xbar
## [1] 2.490291
```

b) Find the maximum likelihood estimator for p using `fitdist`. Note: you'll need to define the "d" version of the function for the PMF (`dcustom`); have it take arguments `x` and `p` (replace n in the equation with 15). Transcribe the formula for the PMF inside the curly brackets. You can write `x!` as `factorial(x)`, etc. Try a starting guess of `p` of 0.2. Sanity check: the MLE is a little above 0.152. (for large sample sizes, the MLE and MOM for the zero-truncated Binomial are essentially the same value)! Remember, you can ignore the Warning `The pcustom function must be defined.`

```
dcustom <- function(x,p) {
  numerator <- factorial(15) * p^x * (1 - p)^(15 - x)
  denominator <- factorial(x) * factorial(15 - x) * (1 - (1 - p)^15)
  return(numerator / denominator)
  }
#Sanity check to make sure you defined dcustom correctly
dcustom(2,0.3)
## [1] 0.09199688
#0.09199688

fit4b <- fitdist(coupons, dcustom, start= list(p=0.2),method = "mle")
## Warning in fitdist(coupons, dcustom, start = list(p = 0.2), method =
"mle"): The
## pcustom function must be defined
fit4b$estimate
##         p
## 0.1520266
```

c) Of special interest is the probability that $X = 8$, i.e., the probability that exactly 8 coupons are redeemed (imagine the customer analytics team has a goal for coupon design so that this probability is 0.1% or larger). In theory, you *could* estimate this as the fraction of customers in the data who used 8 coupons:

```
mean(coupons==8)
## [1] 0
```

However, this estimation procedure doesn't work for this dataset since 8 hasn't yet been observed (though it's definitely a possible value). No worries though! Because we have more information about the probability distribution creating the data (it's coming from a zero-truncated binomial), we still have a way to estimate P(X=8)!

Utilizing the formula for the PMF and plugging in $x = 8$, we find that $P(X = 8)$ can be written as some sophisticated function of p (mouse over to see the equation; if the equation doesn't pop up, try restarting RStudio and/or clicking the double up/down arrows on the far right side of the equation box to expand it):

$$P(X = 8) = \frac{1}{1-(1-p)^{15}} \cdot \frac{15!}{8!(15-8)!} p^8 (1 - p)^{15-8} = \frac{6435p^8(1-p)^7}{1-(1-p)^{15}}$$

Find the MLE of this *function* of p, i.e. the MLE of probability that a customer redeems 8 coupons. You'll find that it is small (about 0.0006), so it's not a huge surprise that 8 hadn't been observed in the data.

```
dcustom <- function(x,p){
  numerator <- 6435* (p^8) *(1-p)^7
  denominator <- 1- (1-p)^15
  (numerator/denominator)
}


fit3c <- fitdist(coupons, dcustom, start = list(p=0.2), method = "mle")
## Warning in fitdist(coupons, dcustom, start = list(p = 0.2), method =
"mle"): The
## dcustom function should return a zero-length vector when input has length
zero
## Warning in fitdist(coupons, dcustom, start = list(p = 0.2), method =
"mle"): The
## pcustom function must be defined
print(fit3c)
## Fitting of the distribution ' custom ' by maximum likelihood
## Parameters:
##    estimate Std. Error
## p 0.5333275  0.1288231
```