Assignment 9

Due Friday, November 6, 11:59 PM CT

Kyle yeo

Files

- The data are in files *geissler.csv* and *french-children.csv*.
- R Code from lecture for the beta-binomial model is in the file *beta-binomial.R*.

Problems

1

Summarize the Geissler data set for families of size of 5 (which is the distribution of boys and girls among the first five children in families with six or more children in Saxony over the time period) with the following calculations: find the number of families, total number of children, number of boys (sex assigned at birth), number of girls, the proportion of each, and the observed sex ratio (boys per 100 girls). Display the summary.

```
geissler <- read.csv("C:/stat_240/data/geissler.csv")

geissler1 <- geissler %>%
  filter(size == 5) %>%
  summarize(
    familiy_size = sum(freq),
    boys = sum(boys*freq),
    girls = sum(girls*freq),
    total = sum(size*freq),
    p_boy = boys/total,
    p_girl = girls/total,
    sex_ratio = 100*boys/girls

  )
geissler1
```

```
##   familiy_size   boys  girls  total     p_boy    p_girl sex_ratio
## 1        95390 245215 231735 476950 0.5141315 0.4858685   105.817
```

2

Fit the simple binomial and beta-binomial models to this data for the number of boys in the family using maximum likelihood. Describe how the assumptions between the two models differ, and how to interpret what this difference implies about the distributions of the numbers of boys and girls among the first five children in this population. Report all parameter estimates for each model and the log-likelihood of each model.

```
size5 <- geissler %>%
  filter(size==5) %>%
  mutate(prop = freq/sum(freq)) %>%
  mutate(boys = boys*freq)


x5 <- size5 %>%
  pull(freq)
## Simple binomial
```

```r
p_hat <- sum(x5*(0:5))/(5*sum(x5))
p_hat
```

```
## [1] 0.5141315
```

```r
logl_1 <- sum(x5*dbinom(0:5,5,p_hat,log=TRUE))
logl_1
```

```
## [1] -146590.6
```

```r
## Beta binomial
bb_5 <- mlebb(x5)
bb_5
```

```
## # A tibble: 1 x 6
##      mu   phi alpha  beta     logl convergence
##   <dbl> <dbl> <dbl> <dbl>    <dbl>       <int>
## 1 0.514  138.  71.1  67.2 -146566.           0
```

For simple binomial, p_hat = 0.5141315, logl_1 = -146590.6 For beta binomial, alpha = 71.1459, beta = 67.23828, logl = -146566.4

Basic assumptions for binomial model are followings: 1. Binary outcomes for each trial(boy and girl) 2. Independence (sex of early trials do not affect subsequent ones) 3. Fixed sample size of 5 4. Same probability of a boy for each child.

However, assumption for beta binomial is different from above i.e. the probability is not fixed and it has different value of p for each family. Therefore, we need to consider different values of p in each family when we use beta binomial model, then the assumption for simp le binomial works within each families. To sum up, we need to keep in mind that for beta binomial model, the distributions of the numbers of boys and girls among the first five children in this population has different p values for each family.

3

> Using results from the previous problem, test the null hypothesis of the binomial model versus the alternative hypothesis of the beta-binomial model. Report a test statistic, the sampling distribution of the test statistic assuming the null hypothesis is true, and a numerical estimate of the p-value. Interpret the results of this hypothesis text in context. For the fitted beta-binomial model, graph the beta density using the estimated parameter values. Interpret the meaning of this graph in context.

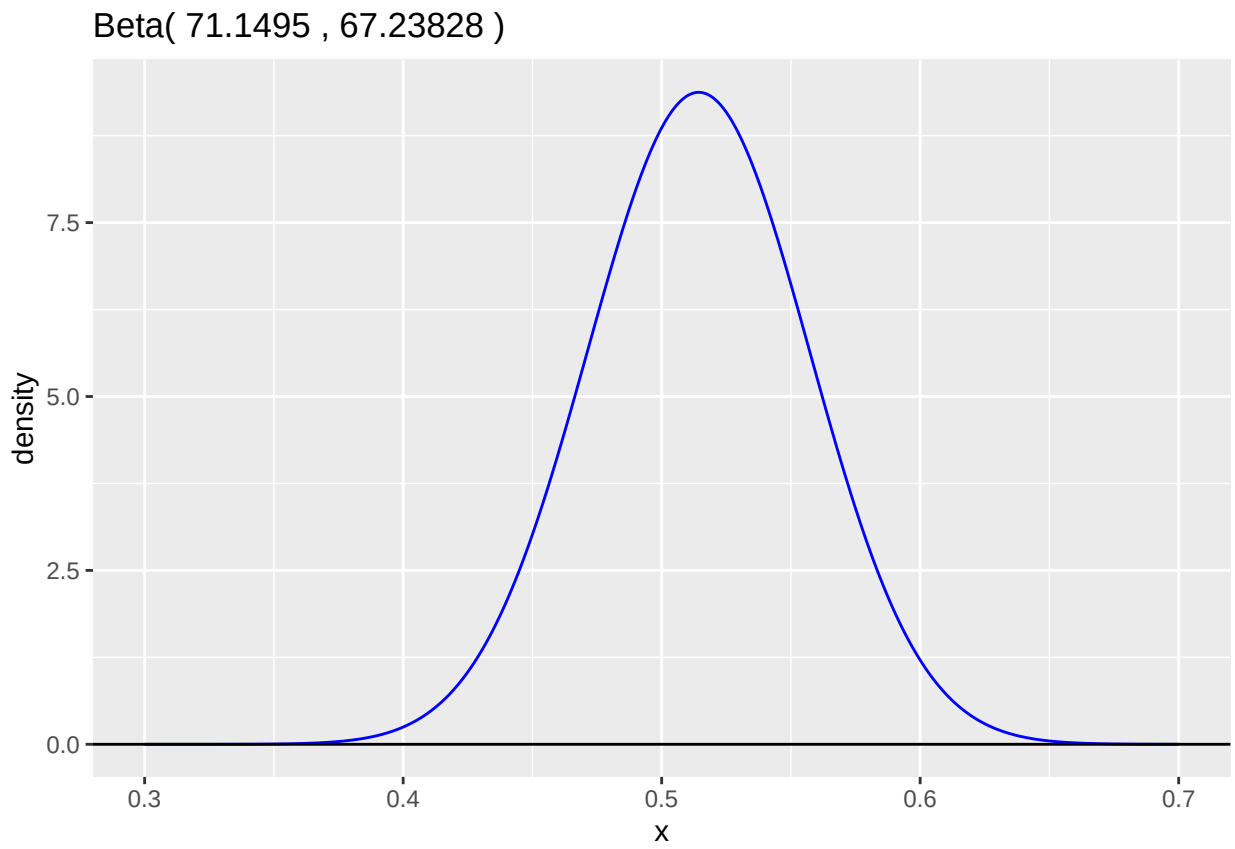
```r
G <- -2 * (logl_1 - bb_5$logl)
G
```

```
## [1] 48.40527
```

```r
p_value_1 <- 1 - pchisq(G,1)
p_value_1
```

```
## [1] 3.466338e-12
```

```r
gbeta(alpha = 71.1495,beta = 67.23828,a=0.3,b=0.7)
```



Sampling distribution of the test statistic : 48.40527 p-value : 3.466338e-12 P is small enough to reject the null hypothesis, therefore, beta binomial method might fit better. The sampling distribution of the test statistic G is approximately chi-squared with one degree of freedom.

4

Using the French family data in the file *french-children.csv*, make the following calculations.

Be sure to read the *Course Notes* description of the data as the format is different than the Geissler data. Specifically, each row specifies the number of families (in 1000s) with a child born given the previous number of boys and girls in the family, and the proportion of boys among those children. Each new child is only counted once and each family will appear each time there is a new child added.

- Find the total number of families, boys, girls, children, and average number of children per family.
- Find the proportion of boys, the proportion of girls, and the sex ratio (# of boys per 100 girls, sexes assigned at birth).
- Determine the number of children for each birth order (first, second, third, and so on) in the data set and count the number of boys and girls in each.
- Calculate the proportion of girls for each birth order and plot these proportions by birth order. Use the size attribute to signify the number of children.
  - Is there a pattern in this data?

```
french_familiy <- read_csv("C:/stat_240/data/french-children.csv")


french4 <- french_familiy %>%
  mutate(families = 1499*1000,
```

```
         num_boys = count*1000*p_boy,
         total_boys = sum(num_boys),
         num_girls = count*1000*(1-p_boy),
         total_girls = sum(num_girls),
         avg = (total_boys+total_girls)/families) %>%
  mutate(p_boy2 = total_boys/(total_boys+total_girls),
         p_girl = total_girls/(total_boys+total_girls),
         sex_ratio = (total_boys/total_girls)*100) %>%
  mutate(birth_order = case_when(girls+boys == 0 ~ "first",
                                 girls+boys == 1 ~ "second",
                                 girls+boys == 2 ~ "third",
                                 girls+boys == 3 ~ "fourth",
                                 girls+boys == 4 ~ "fifth",
                                 girls+boys == 5 ~ "sixth",
                                 girls+boys == 6 ~ "seventh",
                                 girls+boys == 7 ~ "eighth",
                                 girls+boys == 8 ~ "ninth",
                                 girls+boys == 9 ~ "tenth",
                                 girls+boys == 10 ~ "eleventh"))
french4
```

```
## # A tibble: 46 x 14
##    girls  boys count p_boy families num_boys total_boys num_girls total_girls
##    <dbl> <dbl> <dbl> <dbl>    <dbl>    <dbl>      <dbl>     <dbl>       <dbl>
##  1     0     0  1499 0.514  1499000  771086.   1985309.   727914.    1882691.
##  2     0     1   552 0.519  1499000  286322.   1985309.   265678.    1882691.
##  3     0     2   163 0.527  1499000   85901    1985309.    77099     1882691.
##  4     0     3    45 0.525  1499000   23625    1985309.    21375     1882691.
##  5     0     4    13 0.544  1499000    7072    1985309.     5928     1882691.
##  6     0     5     4 0.535  1499000    2140    1985309.     1860     1882691.
##  7     0     6     2 0.521  1499000    1042    1985309.      958     1882691.
##  8     1     0   506 0.509  1499000  257453.   1985309.   248547.    1882691.
##  9     1     1   290 0.512  1499000  148364    1985309.   141636     1882691.
## 10     1     2   114 0.514  1499000   58596    1985309.    55404     1882691.
## # ... with 36 more rows, and 5 more variables: avg <dbl>, p_boy2 <dbl>,
## #   p_girl <dbl>, sex_ratio <dbl>, birth_order <chr>
```

```
final_french <- french4 %>%
  group_by(birth_order) %>%
  summarise(total_c = sum(num_girls + num_boys),
            num_girls = sum(num_girls),
            num_boys = sum(num_boys),
            p_girl = num_girls/total_c) %>%
  arrange(desc(total_c))

final_french
```

```
## # A tibble: 11 x 5
##    birth_order total_c num_girls num_boys p_girl
##    <chr>         <dbl>     <dbl>    <dbl>  <dbl>
##  1 first       1499000   727914.  771086.  0.486
##  2 second      1058000   514225.  543775.  0.486
##  3 third        595000   289451   305549   0.486
##  4 fourth       303000   148503   154497   0.490
##  5 fifth        164000    80412    83588   0.490
```

```
##  6 sixth          96000    47092    48908   0.491
##  7 seventh        63000    30772    32228   0.488
##  8 eighth         40000    19652    20348   0.491
##  9 ninth          26000    12971    13029   0.499
## 10 tenth          16000     7776     8224   0.486
## 11 eleventh        8000     3923     4077   0.490
```

```
ggplot(final_french) +
  geom_point(aes(x = factor(final_french$birth_order, levels = c("first", "second", "third", "fourth",
  xlab("birth_order") +
  ylab("proportion of girls") +
  ggtitle("The relationship between birth order and proportion of girls")
```

As birth_order increases, the proportion of girls tends to increases until the ninth birth order, while the number of children tends to decreases.

5

Using the French family data in the file *french-children.csv*, make the following calculations.

- Determine the number of families with each number of children represented in the data and report these results in a table.
  - The table will have two columns, one for the number of children and one for the number of families with that number of children.
- Create a table with the same structure as the Geissler data with columns `boys`, `girls`, `size`, and `n` so that each row counts the number of families (`n`) in the data set with that number of boys and girls, where size is the number of children in the family. Display the subset of the table for all cases where

the number of boys and girls are the same.

- *(Hint: This last part is tricky. For example, the number of families with exactly 2 boys and 2 girls IS EQUAL TO the number of families who had a boy as the 4th child when they previously had one boy and two girls PLUS the number of families who had a girl as the fourth child when they previously had two boys and one girl MINUS the number of families that previously had two boys and two girls that had another child. A for loop may come in handy.)*

```
french_5_a <- french4 %>%
  mutate(n = girls+boys+1) %>%
  group_by(n)%>%
  mutate(count = sum(count))%>%
  select(count,n)%>%
  distinct()

french_5_a$num = c(abs(diff(french_5_a$count)),8)
french_5 <- french_5_a %>%
  select(-count)
french_5
```

```
## # A tibble: 11 x 2
## # Groups:   n [11]
##        n   num
##    <dbl> <dbl>
##  1     1   441
##  2     2   463
##  3     3   292
##  4     4   139
##  5     5    68
##  6     6    33
##  7     7    23
##  8     8    14
##  9     9    10
## 10    10     8
## 11    11     8
```

```
french_5_b <- french_familiy %>%
  mutate(families = 1499*1000,
         num_b = count*1000*p_boy,
         num_g = count*1000*(1-p_boy),
         total_b = sum(num_b),
         total_g = sum(num_g),
         avg = (total_b+total_g)/families,
         p_boy = total_b/(total_b+total_g),
         p_girl = total_g/(total_b+total_g),
         sex_ratio = (total_b/total_g)*100,
         children = (num_b+num_g))

french_5_b
```

```
## # A tibble: 46 x 13
##    girls  boys count p_boy families   num_b   num_g total_b total_g   avg p_girl
##    <dbl> <dbl> <dbl> <dbl>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>  <dbl>
##  1     0     0  1499 0.513  1499000 771086. 727914.  1.99e6  1.88e6  2.58  0.487
##  2     0     1   552 0.513  1499000 286322. 265678.  1.99e6  1.88e6  2.58  0.487
##  3     0     2   163 0.513  1499000  85901   77099   1.99e6  1.88e6  2.58  0.487
```

```
##  4     0     3    45 0.513  1499000  23625   21375   1.99e6  1.88e6  2.58  0.487
##  5     0     4    13 0.513  1499000   7072    5928   1.99e6  1.88e6  2.58  0.487
##  6     0     5     4 0.513  1499000   2140    1860   1.99e6  1.88e6  2.58  0.487
##  7     0     6     2 0.513  1499000   1042     958   1.99e6  1.88e6  2.58  0.487
##  8     1     0   506 0.513  1499000 257453. 248547.  1.99e6  1.88e6  2.58  0.487
##  9     1     1   290 0.513  1499000 148364  141636   1.99e6  1.88e6  2.58  0.487
## 10     1     2   114 0.513  1499000  58596   55404   1.99e6  1.88e6  2.58  0.487
## # ... with 36 more rows, and 2 more variables: sex_ratio <dbl>, children <dbl>
```

```
french_5_c <- geissler %>%
  select(girls,boys)%>%
  mutate(size = boys+girls) %>%
  mutate(n = 0)

french_5_c
```

```
##    girls boys size n
## 1      1    0    1 0
## 2      2    0    2 0
## 3      3    0    3 0
## 4      4    0    4 0
## 5      5    0    5 0
## 6      6    0    6 0
## 7      7    0    7 0
## 8      8    0    8 0
## 9      9    0    9 0
## 10    10    0   10 0
## 11    11    0   11 0
## 12    12    0   12 0
## 13     0    1    1 0
## 14     1    1    2 0
## 15     2    1    3 0
## 16     3    1    4 0
## 17     4    1    5 0
## 18     5    1    6 0
## 19     6    1    7 0
## 20     7    1    8 0
## 21     8    1    9 0
## 22     9    1   10 0
## 23    10    1   11 0
## 24    11    1   12 0
## 25     0    2    2 0
## 26     1    2    3 0
## 27     2    2    4 0
## 28     3    2    5 0
## 29     4    2    6 0
## 30     5    2    7 0
## 31     6    2    8 0
## 32     7    2    9 0
## 33     8    2   10 0
## 34     9    2   11 0
## 35    10    2   12 0
## 36     0    3    3 0
## 37     1    3    4 0
## 38     2    3    5 0
```

```
## 39     3    3    6 0
## 40     4    3    7 0
## 41     5    3    8 0
## 42     6    3    9 0
## 43     7    3   10 0
## 44     8    3   11 0
## 45     9    3   12 0
## 46     0    4    4 0
## 47     1    4    5 0
## 48     2    4    6 0
## 49     3    4    7 0
## 50     4    4    8 0
## 51     5    4    9 0
## 52     6    4   10 0
## 53     7    4   11 0
## 54     8    4   12 0
## 55     0    5    5 0
## 56     1    5    6 0
## 57     2    5    7 0
## 58     3    5    8 0
## 59     4    5    9 0
## 60     5    5   10 0
## 61     6    5   11 0
## 62     7    5   12 0
## 63     0    6    6 0
## 64     1    6    7 0
## 65     2    6    8 0
## 66     3    6    9 0
## 67     4    6   10 0
## 68     5    6   11 0
## 69     6    6   12 0
## 70     0    7    7 0
## 71     1    7    8 0
## 72     2    7    9 0
## 73     3    7   10 0
## 74     4    7   11 0
## 75     5    7   12 0
## 76     0    8    8 0
## 77     1    8    9 0
## 78     2    8   10 0
## 79     3    8   11 0
## 80     4    8   12 0
## 81     0    9    9 0
## 82     1    9   10 0
## 83     2    9   11 0
## 84     3    9   12 0
## 85     0   10   10 0
## 86     1   10   11 0
## 87     2   10   12 0
## 88     0   11   11 0
## 89     1   11   12 0
## 90     0   12   12 0
```

```
for (i in (1:90)){
  number_b = french_5_c$boys[i]
  number_g = french_5_c$girls[i]
  a = filter(french_5_b,
             boys == (number_b - 1),
             girls == number_g)%>%
    pull(num_b)
  b = filter(french_5_b,
             boys == (number_b),
             girls == (number_g-1))%>%
    pull(num_g)
  c = filter(french_5_b,
             boys == (number_b),
             girls == (number_g))%>%
    pull(children)

  children = 0
  if(length(a)>0)
    children = a + children
  if(length(b)>0)
    children = b + children
  if(length(c)>0)
    children = children - c
  french_5_c$n[i] = children
}

french_5_c %>%
  filter(n>0)
```

```
##    girls boys size        n
## 1      1    0    1 221914.4
## 2      2    0    2 106547.2
## 3      3    0    3  33716.0
## 4      4    0    4   8759.0
## 5      5    0    5   2170.0
## 6      6    0    6    539.0
## 7      7    0    7    516.0
## 8      0    1    1 219085.6
## 9      1    1    2 233130.4
## 10     2    1    3 105920.0
## 11     3    1    4  33206.0
## 12     4    1    5  10096.0
## 13     5    1    6   2531.0
## 14     6    1    7   1544.0
## 15     7    1    8   1036.0
## 16     0    2    2 123322.4
## 17     1    2    3 111463.0
## 18     2    2    4  49439.0
## 19     3    2    5  19314.0
## 20     4    2    6   8343.0
## 21     5    2    7   3479.0
## 22     6    2    8    982.0
## 23     7    2    9   1563.0
## 24     0    3    3  40901.0
```

```
## 25     1    3    4  36971.0
## 26     2    3    5  20888.0
## 27     3    3    6  10197.0
## 28     4    3    7   4676.0
## 29     5    3    8   2460.0
## 30     6    3    9   1515.0
## 31     7    3   10   1497.0
## 32     0    4    4  10625.0
## 33     1    4    5  12460.0
## 34     2    4    6   7990.0
## 35     3    4    7   6000.0
## 36     4    4    8   2945.0
## 37     5    4    9   1866.0
## 38     6    4   10   1943.0
## 39     7    4   11   1028.0
## 40     0    5    5   3072.0
## 41     1    5    6   3260.0
## 42     2    5    7   4054.0
## 43     3    5    8   2909.0
## 44     4    5    9   1996.0
## 45     5    5   10   1980.0
## 46     6    5   11   2430.0
## 47     0    6    6    140.0
## 48     1    6    7   1689.0
## 49     2    6    8   2033.0
## 50     3    6    9   1506.0
## 51     4    6   10    999.0
## 52     5    6   11   2979.0
## 53     0    7    7   1042.0
## 54     1    7    8   1635.0
## 55     2    7    9   1554.0
## 56     3    7   10   1581.0
## 57     4    7   11   1563.0
```

```
french_5_d <- french_5_c[ which(french_5_c$girls == french_5_c$boys), ]

french_5_d
```

```
##    girls boys size        n
## 14     1    1    2 233130.4
## 27     2    2    4  49439.0
## 39     3    3    6  10197.0
## 50     4    4    8   2945.0
## 60     5    5   10   1980.0
## 69     6    6   12      0.0
```

6

Using the data set of single-birth French families, determine for families with `b` boys and `g` girls the proportion of families which have a subsequent child. This will be a table with columns `boys`, `girls`, and a column for the proportion. Display a subset of these proportions in a reshaped table with one row for the number of previous girls (ranging from 0 to 4) and one column for the number of previous boys (also ranging from 0 to 4). Do you agree or disagree with this statement: families with more boys than girls are more likely to continue to have additional children. Use evidence from the displayed table to justify your response.

```
french_6_a <- french_familiy %>%
  mutate(children = count*1000)

french_6_a
```

```
## # A tibble: 46 x 5
##    girls  boys count p_boy children
##    <dbl> <dbl> <dbl> <dbl>    <dbl>
##  1     0     0  1499 0.514  1499000
##  2     0     1   552 0.519   552000
##  3     0     2   163 0.527   163000
##  4     0     3    45 0.525    45000
##  5     0     4    13 0.544    13000
##  6     0     5     4 0.535     4000
##  7     0     6     2 0.521     2000
##  8     1     0   506 0.509   506000
##  9     1     1   290 0.512   290000
## 10     1     2   114 0.514   114000
## # ... with 36 more rows
```

```
french_6_b <- french_6_a %>%
  select(boys,girls,children)
french_6_b
```

```
## # A tibble: 46 x 3
##     boys girls children
##    <dbl> <dbl>    <dbl>
##  1     0     0  1499000
##  2     1     0   552000
##  3     2     0   163000
##  4     3     0    45000
##  5     4     0    13000
##  6     5     0     4000
##  7     6     0     2000
##  8     0     1   506000
##  9     1     1   290000
## 10     2     1   114000
## # ... with 36 more rows
```

```
french_6_c <- french_5_c%>%
  select(-size)
french_6_c
```

```
##    girls boys        n
## 1      1    0 221914.4
## 2      2    0 106547.2
## 3      3    0  33716.0
## 4      4    0   8759.0
## 5      5    0   2170.0
## 6      6    0    539.0
## 7      7    0    516.0
## 8      8    0      0.0
## 9      9    0      0.0
## 10    10    0      0.0
## 11    11    0      0.0
```

```
## 12    12    0      0.0
## 13     0    1 219085.6
## 14     1    1 233130.4
## 15     2    1 105920.0
## 16     3    1  33206.0
## 17     4    1  10096.0
## 18     5    1   2531.0
## 19     6    1   1544.0
## 20     7    1   1036.0
## 21     8    1      0.0
## 22     9    1      0.0
## 23    10    1      0.0
## 24    11    1      0.0
## 25     0    2 123322.4
## 26     1    2 111463.0
## 27     2    2  49439.0
## 28     3    2  19314.0
## 29     4    2   8343.0
## 30     5    2   3479.0
## 31     6    2    982.0
## 32     7    2   1563.0
## 33     8    2      0.0
## 34     9    2      0.0
## 35    10    2      0.0
## 36     0    3  40901.0
## 37     1    3  36971.0
## 38     2    3  20888.0
## 39     3    3  10197.0
## 40     4    3   4676.0
## 41     5    3   2460.0
## 42     6    3   1515.0
## 43     7    3   1497.0
## 44     8    3      0.0
## 45     9    3      0.0
## 46     0    4  10625.0
## 47     1    4  12460.0
## 48     2    4   7990.0
## 49     3    4   6000.0
## 50     4    4   2945.0
## 51     5    4   1866.0
## 52     6    4   1943.0
## 53     7    4   1028.0
## 54     8    4      0.0
## 55     0    5   3072.0
## 56     1    5   3260.0
## 57     2    5   4054.0
## 58     3    5   2909.0
## 59     4    5   1996.0
## 60     5    5   1980.0
## 61     6    5   2430.0
## 62     7    5      0.0
## 63     0    6    140.0
## 64     1    6   1689.0
## 65     2    6   2033.0
```

```
## 66     3    6   1506.0
## 67     4    6    999.0
## 68     5    6   2979.0
## 69     6    6      0.0
## 70     0    7   1042.0
## 71     1    7   1635.0
## 72     2    7   1554.0
## 73     3    7   1581.0
## 74     4    7   1563.0
## 75     5    7      0.0
## 76     0    8      0.0
## 77     1    8      0.0
## 78     2    8      0.0
## 79     3    8      0.0
## 80     4    8      0.0
## 81     0    9      0.0
## 82     1    9      0.0
## 83     2    9      0.0
## 84     3    9      0.0
## 85     0   10      0.0
## 86     1   10      0.0
## 87     2   10      0.0
## 88     0   11      0.0
## 89     1   11      0.0
## 90     0   12      0.0
```

```
french_6_d <- inner_join(french_6_b, french_6_c)%>%
  mutate(x = children/(children+n))%>%
  select(boys,girls,x)
french_6_d
```

```
## # A tibble: 45 x 3
##     boys girls     x
##    <dbl> <dbl> <dbl>
##  1     1     0 0.716
##  2     2     0 0.569
##  3     3     0 0.524
##  4     4     0 0.550
##  5     5     0 0.566
##  6     6     0 0.935
##  7     0     1 0.695
##  8     1     1 0.554
##  9     2     1 0.506
## 10     3     1 0.538
## # ... with 35 more rows
```

```
french_6_e <- french_6_d %>%
  pivot_wider(names_from = boys, values_from =x)%>%
  select(1,8,2,3,4,5)%>%
  filter(!(girls == 5|girls == 6))
french_6_e
```

```
## # A tibble: 5 x 6
##   girls    `0`   `1`   `2`   `3`   `4`
##   <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
```

```
## 1     0 NA     0.716 0.569 0.524 0.550
## 2     1  0.695 0.554 0.506 0.538 0.562
## 3     2  0.571 0.503 0.548 0.590 0.652
## 4     3  0.523 0.534 0.600 0.651 0.647
## 5     4  0.533 0.581 0.609 0.702 0.731
```

I disagree with the statement since we can see that the only case when boy = 1 is necessary to the statement.