R Basics

Aleksandr Fisher

Getting the Basics

- Don't forget to check and set your working directory
- R can't find files that aren't there

Getting the Basics

- The Behavioral Risk Factor Surveillance System (BRFSS) is an annual telephone survey of 350,000 people in the United States. The BRFSS Web site contains a complete description of the survey, including the research questions that motivate the study and many interesting results derived from the data.

```
library(tidyverse)
source("http://www.openintro.org/stat/data/cdc.R")
CDC <- tbl_df(cdc)
class(cdc)
```

```
## [1] "data.frame"
```

Explore the data

- You can see and inspect the data set comfortably in RStudio with the View() command, which invokes a spreadsheet-style data viewer on a matrix-like R object.

```
CDC
```

```
## # A tibble: 20,000 x 9
##    genhlth   exerany hlthplan smoke100 height weight wtdesire   age gender
##    <fct>       <dbl>    <dbl>    <dbl>  <dbl>  <int>    <int> <int> <fct>
##  1 good            0        1        0     70    175      175    77 m
##  2 good            0        1        1     64    125      115    33 f
##  3 good            1        1        1     60    105      105    49 f
##  4 good            1        1        0     66    132      124    42 f
##  5 very good       0        1        0     61    150      130    55 f
##  6 very good       1        1        0     64    114      114    55 f
##  7 very good       1        1        0     71    194      185    31 m
##  8 very good       0        1        0     67    170      160    45 m
##  9 good            0        1        1     65    150      130    27 f
## 10 good            1        1        0     70    180      170    44 m
## # ... with 19,990 more rows
```

Explore the data

- After loading the data and converting it into a tibble, one should inspect the data to get some understanding about the structure and content. Common funtions for these tasks are:
- `<name-of-data-tibble>`: Display the first 10 rows and all columns that fit on one screen. It also prints an abbreviated description of the column type.
- `head(<name-of-df>)`, `tail(<name-of-df>)`: Return the first or last part. Use these commands if it is not a tibble but a data frame
- `dim()`: Retrieve the dimension
- `names()`: Get the names

Explore the data

- `str()`: Display compactly the internal structure
- `glimpse()`: is the dplyr-version of str() showing values of each variable the whole sceen width, but does not display the number of levels and names of factor variables. But this feature of str() cannot be displayed completly with either many or long levels names.
- `View()`: With RStudio you can see and inspect the data set comfortably. The View() function invokes a spreadsheet-style data viewer.

Install Packages

- When you download R from the Comprehensive R Archive Network (CRAN), you get that "base" R system
- The base R system comes with basic functionality; implements the R language
- One reason R is so useful is the large collection of packages that extend the basic functionality of R
- R packages are developed and published by the larger R community

Install Packages

- Packages can be installed with the `install.packages()` function in R
- To install a single package, pass the name of the lecture to the install.packages() function as the first argument
- You can install multiple R packages at once with a single call to install.packages()
- `install.packages(c("dplyr", "ggplot2", "devtools"))`

Loading R Packages

- Installing a package does not make it immediately available to you in R; you must load the package
- The library() function is used to load packages into R
- The following code is used to load the ggplot2 package into R

```
library(ggplot2)
```

- NOTE: Do not put the package name in quotes!

Arbuthnot Dataset

- The Arbuthnot data set refers to Dr. John Arbuthnot, an 18th century physician, writer, and mathematician. He was interested in the ratio of newborn boys to newborn girls, so he gathered the baptism records for children born in London for every year from 1629 to 1710. We can view the data by typing its name into the console.
- Read data from online with the code below:

```
source("http://www.openintro.org/stat/data/arbuthnot.R")
```

Look as Data

```r
dim(arbuthnot) #get number of rows and columns
```

```
## [1] 82  3
```

```r
glimpse(arbuthnot) #get the structure of the data
```

```
## Rows: 82
## Columns: 3
## $ year  <int> 1629, 1630, 1631, 1632, 1633, 1634, 1635, 1636, 1637, 1638, 1...
## $ boys  <int> 5218, 4858, 4422, 4994, 5158, 5035, 5106, 4917, 4703, 5359, 5...
## $ girls <int> 4683, 4457, 4102, 4590, 4839, 4820, 4928, 4605, 4457, 4952, 4...
```

Look as Data

```r
names(arbuthnot)
```

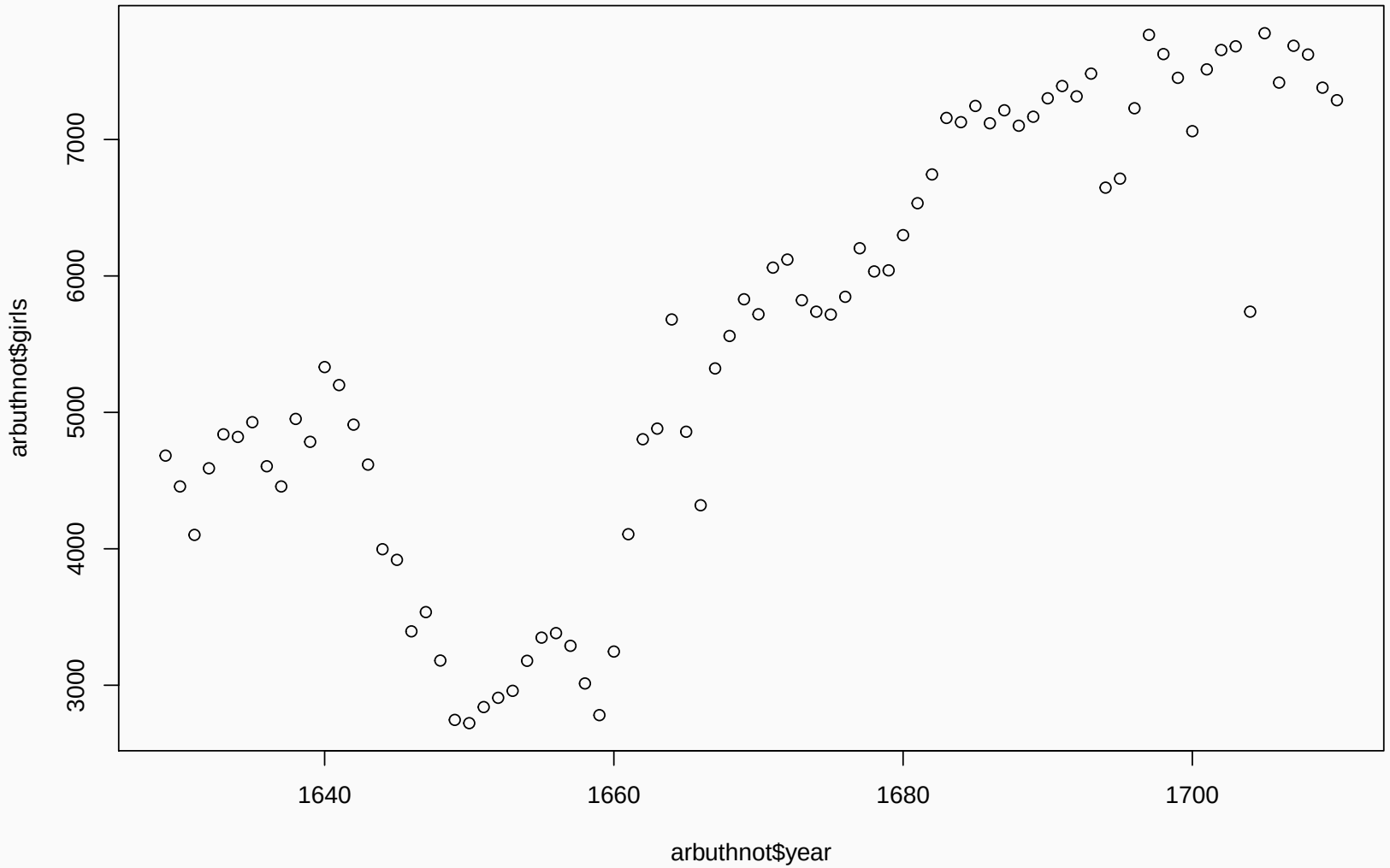
```
## [1] "year"  "boys"  "girls"
```

```r
#We can access the data in a single column of a data frame separately.
arbuthnot$boys
```

```
##  [1] 5218 4858 4422 4994 5158 5035 5106 4917 4703 5359 5366 5518 5470 5460 4793
## [16] 4107 4047 3768 3796 3363 3079 2890 3231 3220 3196 3441 3655 3668 3396 3157
## [31] 3209 3724 4748 5216 5411 6041 5114 4678 5616 6073 6506 6278 6449 6443 6073
## [46] 6113 6058 6552 6423 6568 6247 6548 6822 6909 7577 7575 7484 7575 7737 7487
## [61] 7604 7909 7662 7602 7676 6985 7263 7632 8062 8426 7911 7578 8102 8031 7765
## [76] 6113 8366 7952 8379 8239 7840 7640
```

Plot Data - Base R

```
plot(x = arbuthnot$year, y = arbuthnot$girls)
```

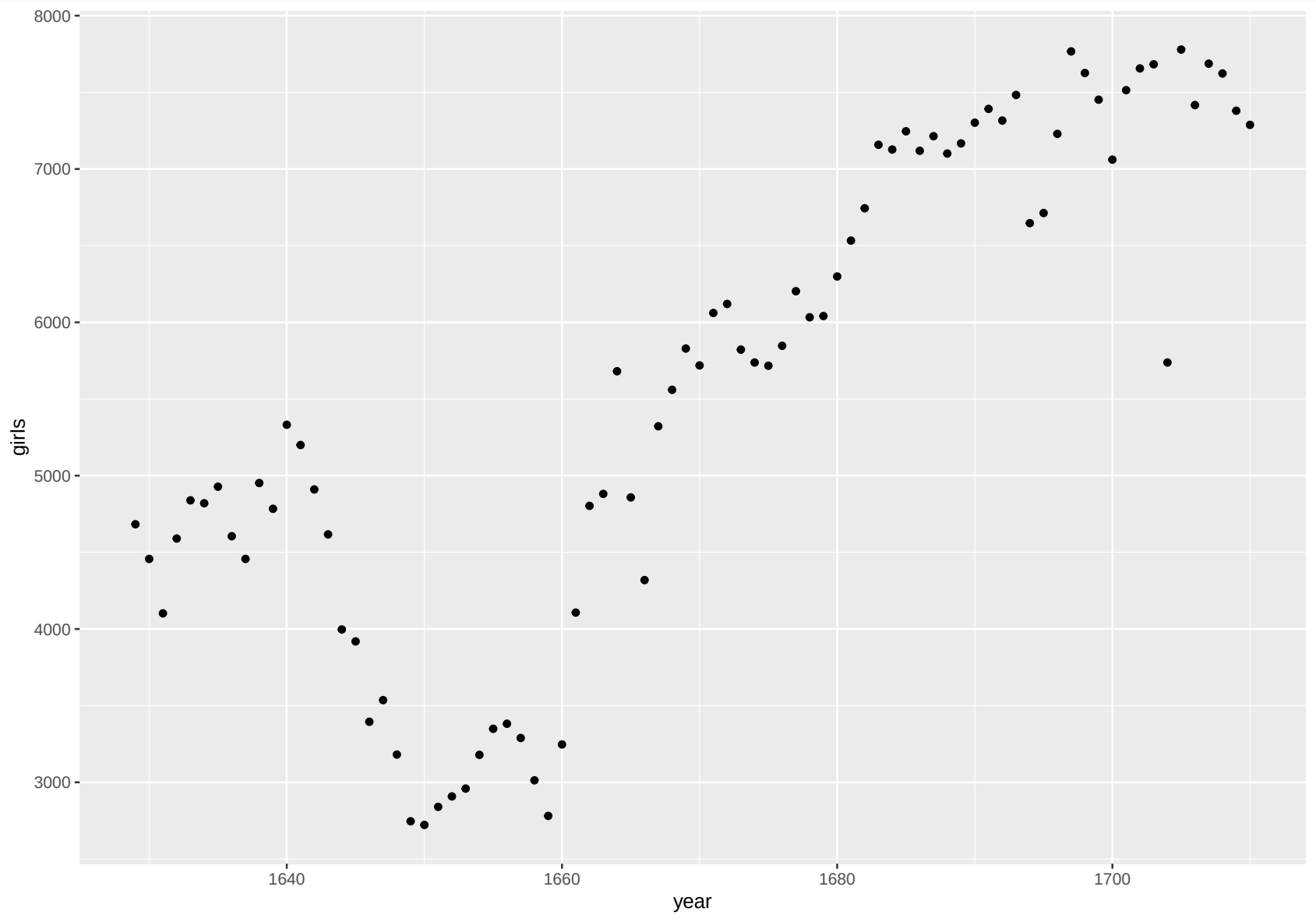


Plot Data - qplot R

- R has some powerful functions for making graphics. We can create a simple plot of the number of girls baptized per year with qplot.

```
qplot(x = year, y = girls, data = arbuthnot)
```

Plot Data - qplot R

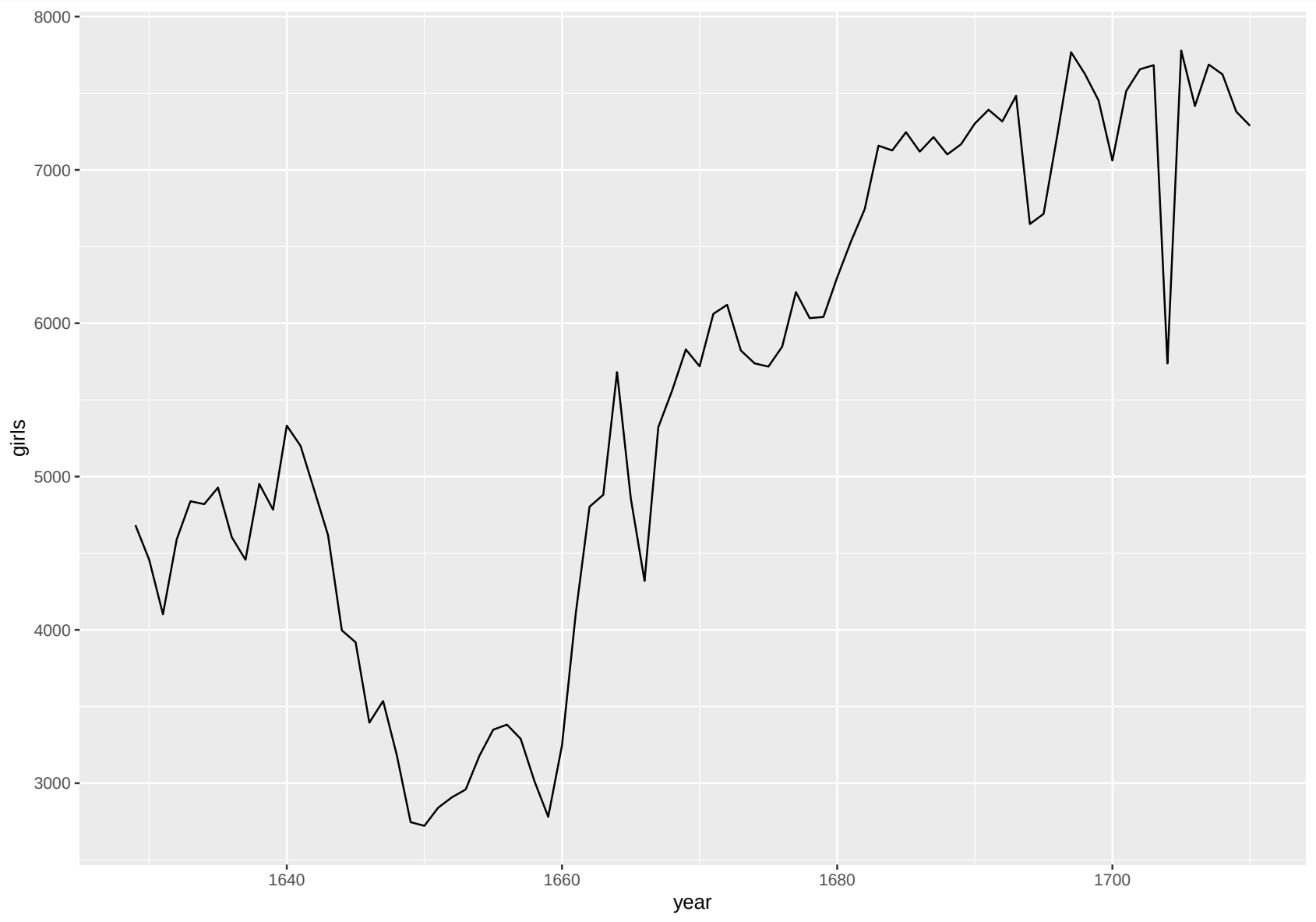


Plot Data - line graph

```
plot(x = arbuthnot$year, y = arbuthnot$girls, type = "l")
```

Plot Data - line graph

```
qplot(x = year, y = girls, data = arbuthnot, geom = "line")
```



Manipulating Data - Creating New Variables

- Now, suppose we want to plot the total number of baptisms. To compute this, we could use the fact that R is really just a big calculator. We can type in mathematical expressions like

```
5218 + 4683
```

```
## [1] 9901
```

```
arbuthnot$boys + arbuthnot$girls
```

```
##  [1]  9901  9315  8524  9584  9997  9855 10034  9522  9160 10311 10150 10850
## [13] 10670 10370  9410  8104  7966  7163  7332  6544  5825  5612  6071  6128
## [25]  6155  6620  7004  7050  6685  6170  5990  6971  8855 10019 10292 11722
## [37]  9972  8997 10938 11633 12335 11997 12510 12563 11895 11851 11775 12399
## [49] 12626 12601 12288 12847 13355 13653 14735 14702 14730 14694 14951 14588
## [61] 14771 15211 15054 14918 15159 13632 13976 14861 15829 16052 15363 14639
## [73] 15616 15687 15448 11851 16145 15369 16066 15862 15220 14928
```

Manipulating Data - Creating New Variables (mutate)

- We'll be using this new vector to generate some plots, so we'll want to save it as a permanent column in our data frame.

```
arbuthnot <- arbuthnot %>%
  mutate(total = boys + girls)
```

Piping Operator

- The %>% operator is called the piping operator. It takes the output of the previous expression and pipes it into the first argument of the function in the following one. To continue our analogy with mathematical functions, `x %>% f(y)` is equivalent to `f(x, y)`.

Piping Operator

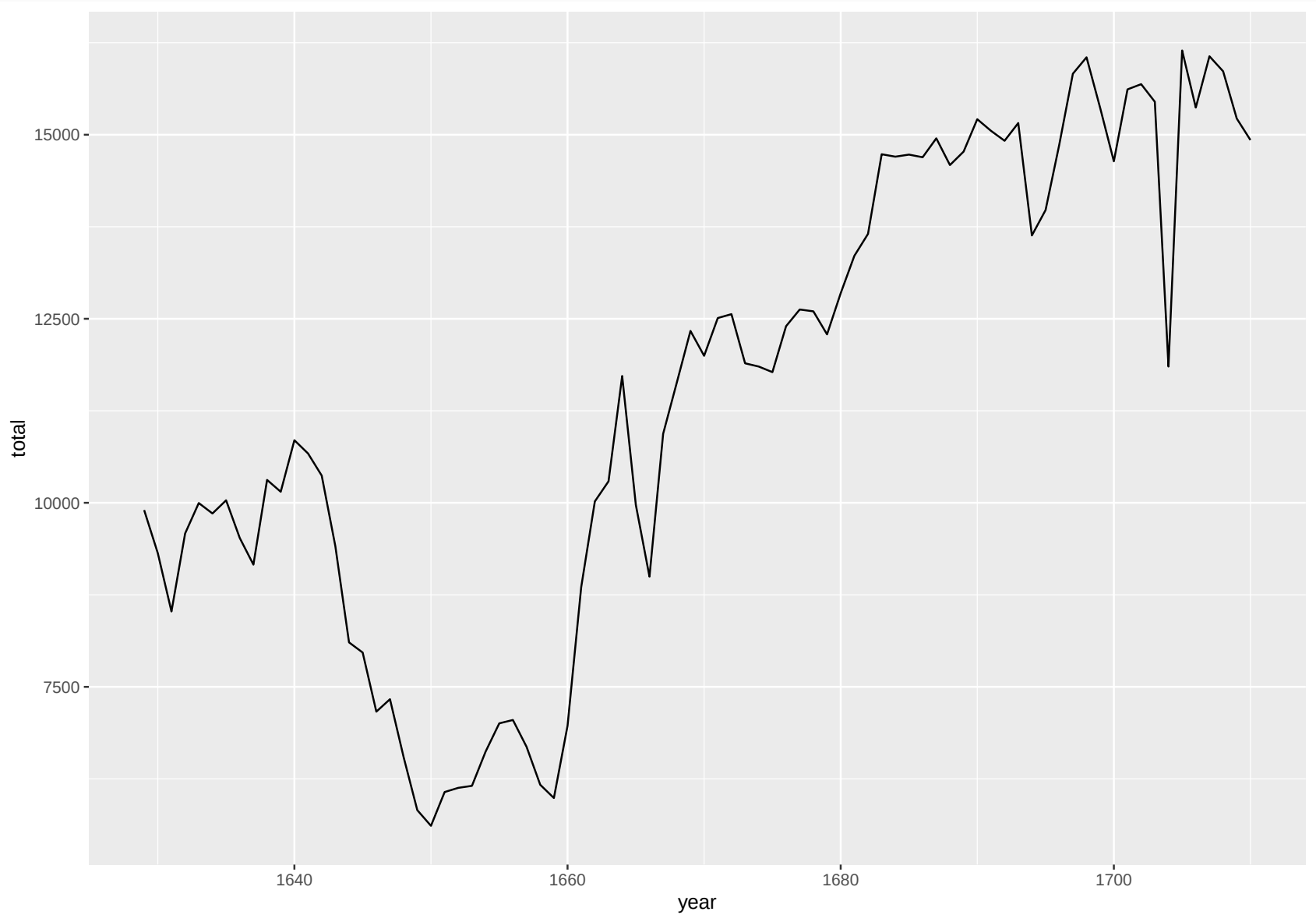
- **A note on piping:** Note that we can read these three lines of code as the following:
- "Take the arbuthnot dataset and pipe it into the mutate function. Mutate the arbuthnot data set by creating a new variable called total that is the sum of the variables called boys and girls.
- Then assign the resulting dataset to the object called arbuthnot, i.e. overwrite the old arbuthnot dataset with the new one containing the new variable."

Manipulating Data - Adding in plot

```
plot(arbuthnot$year, arbuthnot$boys + arbuthnot$girls, type = "l")
```

Manipulating Data - Adding in qplot

```
qplot(x = year, y = total, data = arbuthnot, geom = "line")
```



Manipulating Data - creating ratios

```
5218 / 4683
```

```
## [1] 1.114243
```

Manipulating Data - creating ratios

```
arbuthnot$boys / arbuthnot$girls
```

```
##  [1] 1.114243 1.089971 1.078011 1.088017 1.065923 1.044606 1.036120 1.067752
##  [9] 1.055194 1.082189 1.121656 1.034884 1.051923 1.112016 1.038120 1.027521
## [17] 1.032661 1.109867 1.073529 1.057215 1.121267 1.061719 1.137676 1.107290
## [25] 1.080095 1.082416 1.091371 1.084565 1.032533 1.047793 1.153901 1.146905
## [33] 1.156075 1.085988 1.108584 1.063369 1.052697 1.083121 1.055242 1.092266
## [41] 1.116143 1.097744 1.064016 1.052778 1.043112 1.065354 1.059647 1.120575
## [49] 1.035467 1.088679 1.034100 1.039530 1.044237 1.024466 1.058536 1.062860
## [57] 1.032846 1.064054 1.072498 1.054359 1.060974 1.083128 1.036526 1.039092
## [65] 1.025792 1.050850 1.081931 1.055748 1.037981 1.104904 1.061594 1.073219
## [73] 1.078254 1.048981 1.010673 1.065354 1.075460 1.072132 1.090022 1.080808
## [81] 1.062331 1.048299
```

```
arbuthnot <- arbuthnot %>%
  mutate(boy_to_girl_ratio = boys / girls)
```

Manipulating Data - ratios using mutate

```
arbuthnot <- arbuthnot %>%
  mutate(boy_ratio = boys / total)
```

Manipulating Data - True/False

```
arbuthnot$boys > arbuthnot$girls
```

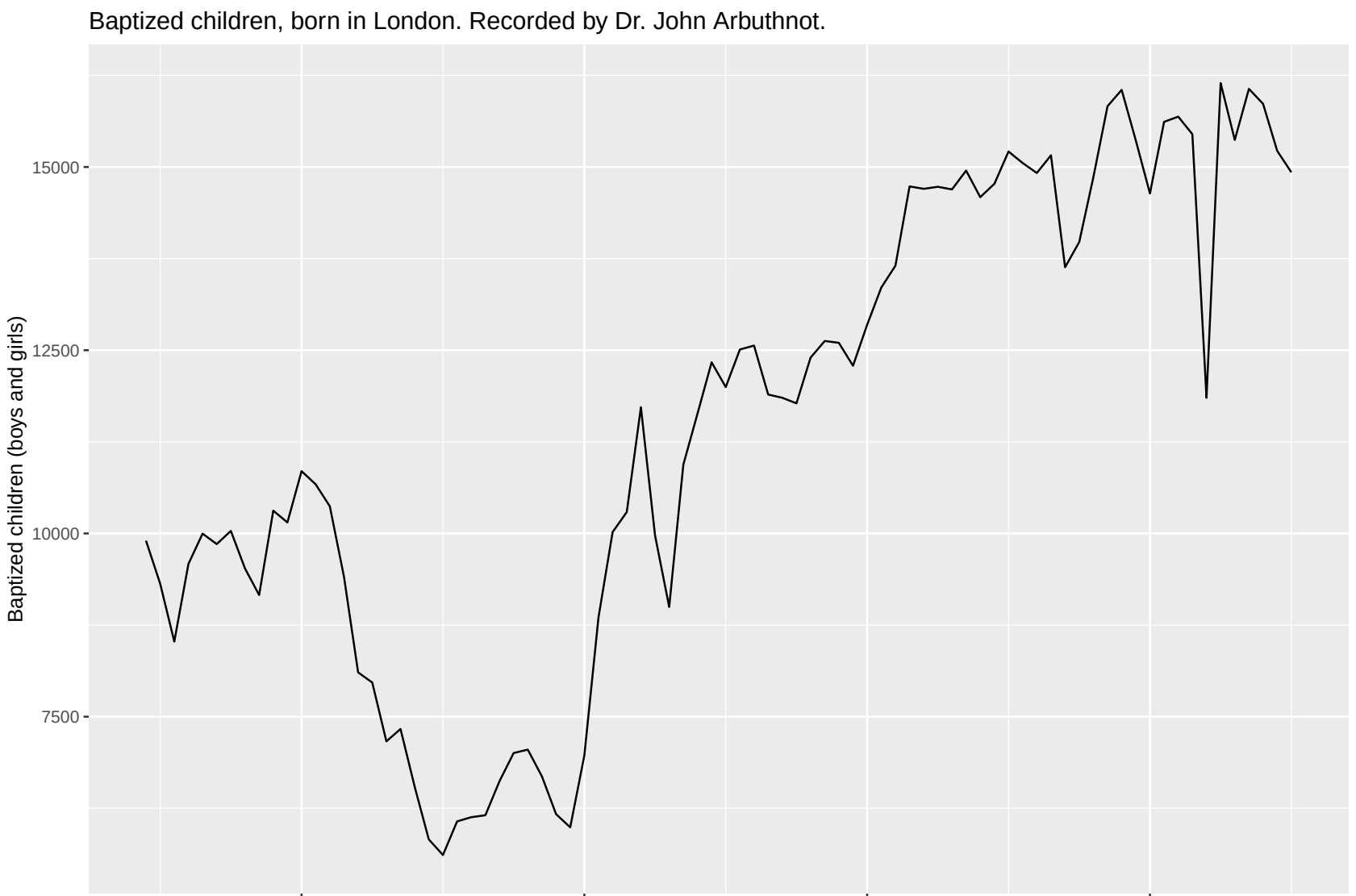
```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [16] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [31] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [46] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [61] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [76] TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

Manipulating Data

- Make a plot that displays the boy-to-girl ratio for every year in the data set. What do you see? Does Arbuthnot's observation about boys being born in greater proportion than girls hold up in the U.S.? Include the plot in your response.
- In what year did we see the most total number of births in the U.S.? You can refer to the help files or the R reference card http://cran.r-project.org/doc/contrib/Short-refcard.pdf to find helpful commands.

Some More Plots

```
arbuthnot %>% ggplot(aes(year, total)) +
        geom_line() +
        xlab("Year") + ylab("Baptized children (boys and girls)") +
        ggtitle("Baptized children, born in London. Recorded by Dr. John Arbuthnot.")
```

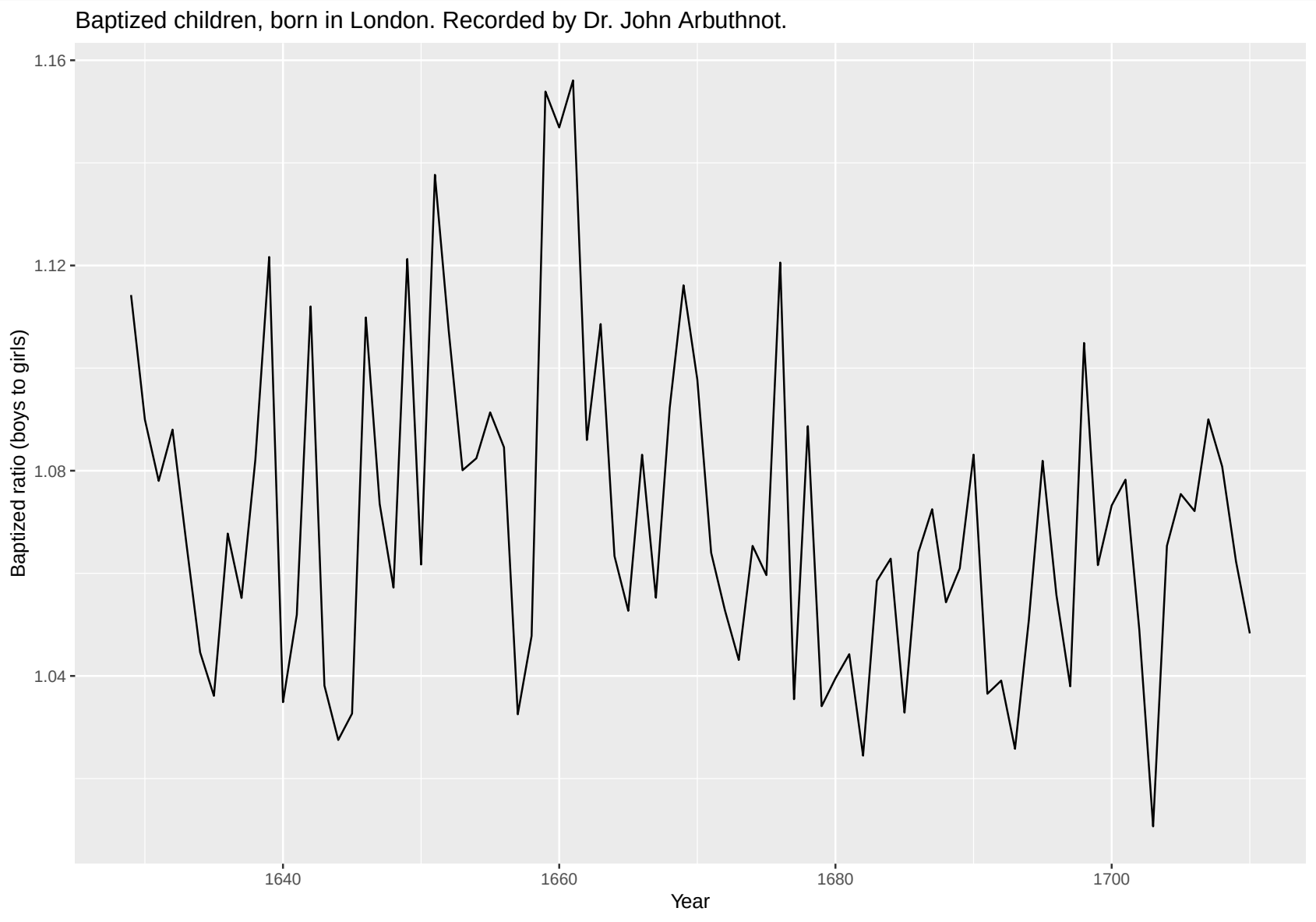


Some More Plots

- Similarly to how we computed the total number of births, we can compute different kinds of ratios (boys to girls, boys to total, girls to total).

```
arbuthnot <- arbuthnot %>%
  mutate(boy_to_girl_ratio = boys / girls)
arbuthnot <- arbuthnot %>%
  mutate(boy_ratio = boys / total)
arbuthnot <- arbuthnot %>%
  mutate(girl_ratio = girls / total)
arbuthnot %>%
        ggplot(aes(year, boy_to_girl_ratio)) +
        geom_line() +
        xlab("Year") + ylab("Baptized ratio (boys to girls)") +
        ggtitle("Baptized children, born in London. Recorded by Dr. John Arbuthnot.")
```

Ratio Plot



- I am now providing a short version of the plot graph with the