C91AR | Advanced Statistics using R

Lecture 6: Data Simulation for Correlation in R

Dr Pete McKenna

2025-02-28

Contents

1 Todays packages

```
# load packages
pacman::p_load(tidyverse,
               corrr,
               psych,
               tidyplots)
```

2 What's the deal with data simulation?

- Simulating data is a really useful skill for research and teaching.
- For research, it can help you to plan and prepare analysis scripts (e.g., for pre-registration), instead of waiting until data collection is complete.
  - This in turn can help you select the correct statistical test.
- For teaching, you can create topic-relevant datasets for specific courses.
- More generally, creating and working with simulated data helps to develop your understanding of statistical concepts.

3 OK, so what is today's session going to cover?

- The formula for correlation
- The related R code for simulating bivariate (i.e., includes 2 variables) data
- How to plot simulated data

4 Simulating Univariate data

- Data along a normal distribution can be simulated using the `rnorm` function, like so:

```
set.seed(385) # seed a random number for reproducibility


# Say we wanted to simulate response time to a cogntive test
# where the mean = 1000, the SD = 50, from 150 participants


rt_sim <-
  tibble(                           # create data table object
  control_rt = rnorm(mean = 1000, # state the mean
                     sd = 500,     # provide the sd
                     n = 150))     # supply the sample size
```

5 Examine the data

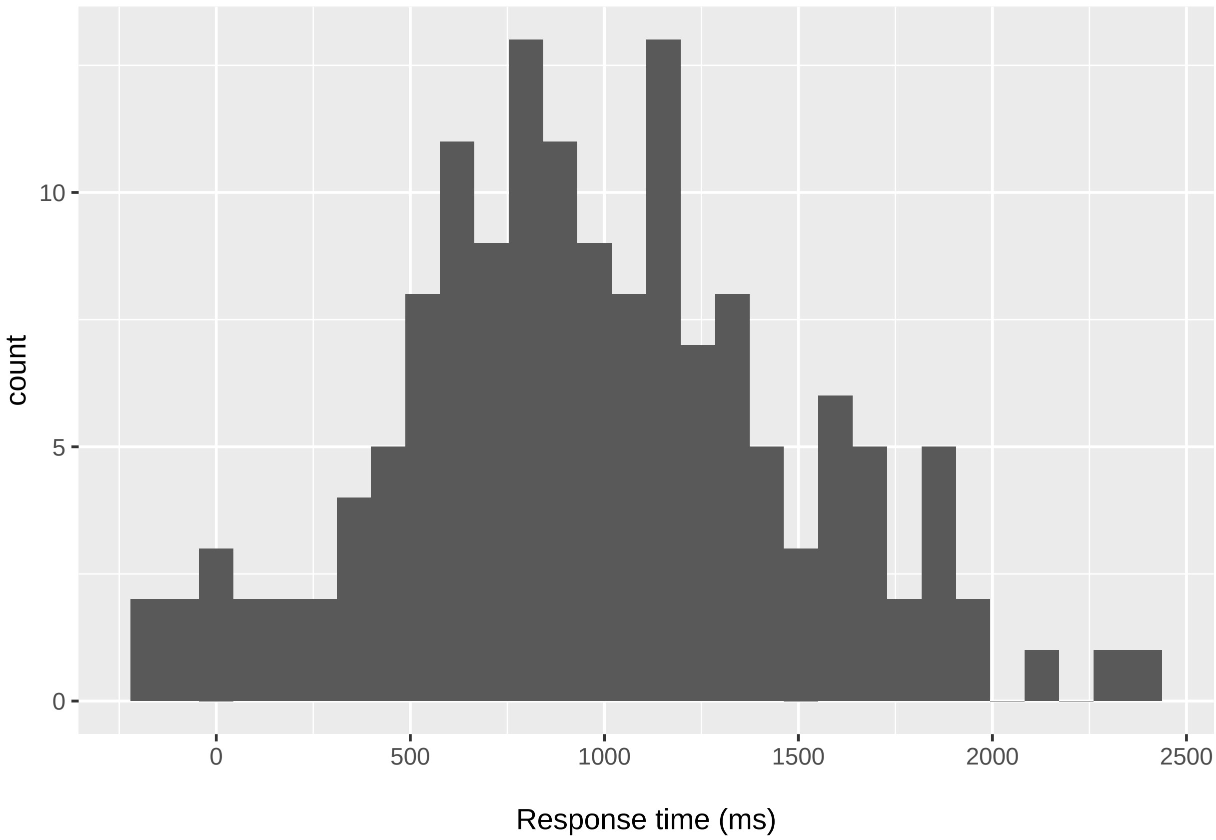
```
headTail(rt_sim)
```

```
##    control_rt
```

```
## 1    1443.73
## 2    1568.94
## 3     808.48
## 4     699.91
## 5        ...
## 6     428.12
## 7     773.09
## 8     663.01
## 9     859.12
```

- Here's our data plotted in a histogram.

```
# generate the histogram with ggplot2
ggplot(data = rt_sim,                        # specify data source
       mapping = aes(x = control_rt)) + # specify x and y mapping
  geom_histogram() +                         # add histogram layer
  labs(x = "\nResponse time (ms)")           # supply axis labels
```



6 How NOT To Simulate Bivariate data

- However, to simulate the distribution of two related variables you can't just run `rnorm` twice as you will end up with two variables that are unrelated, with a correlation of (near) zero.

```
# simulate two separate datasets
data <-
  tibble(
  control_rt = rnorm(mean = 1000,
                     sd = 500,
                     n = 150),
  treatment_rt = rnorm(mean = 1200,
                       sd = 700,
                       n = 150))

# run a correlation
cor(data$control_rt,   # Pearson is the default
    data$treatment_rt)
```

```
## [1] -0.0373358
```

7 Our first attempt at simulating multivariate data

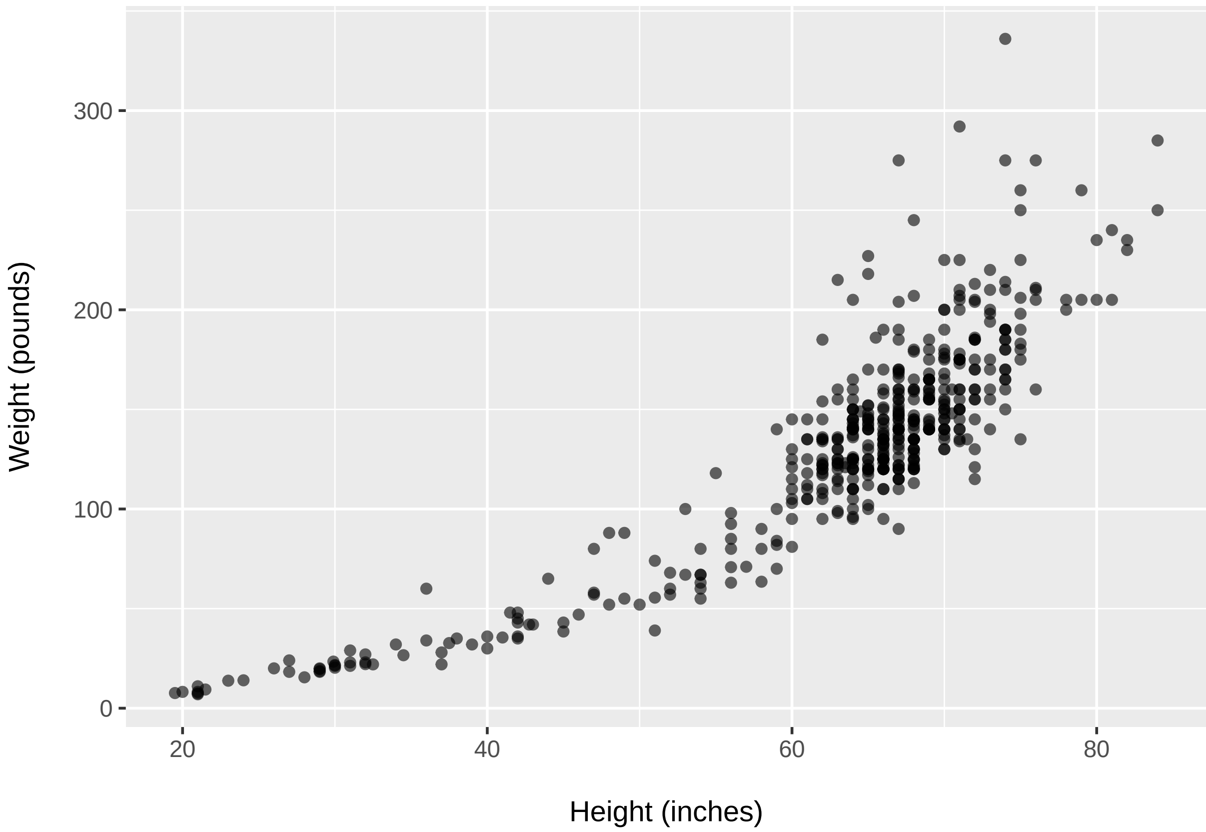
- Let's start by simulating some data representing hypothetical humans and their height and weight.
- We know these things are correlated.
- What we need to be able to simulate are the **means**, **standard deviations**, and the **correlations between these two variables**.
- I'm using a dataset of heights and weights - link on final slide.

```
# read in the data
# make sure you include the full file name and type (e.g., .csv) in the quotes
handw <-
  read_csv("data_raw/heights_and_weights.csv",
           col_types = "dd")               # specify that both cols are `double` type



# peek into heights and weights dataset
glimpse(handw)
```

```
## Rows: 475
## Columns: 2
## $ height_in  <dbl> 63, 67, 71, 71, 56, 67, 65, 74, 64, 62, 60, 47, 38, 67, 67,~
## $ weight_lbs <dbl> 130, 169, 178, 225, 98, 204, 145, 180, 150, 136, 110, 80, 3~
```

8 Scatter plot the heights and weights data

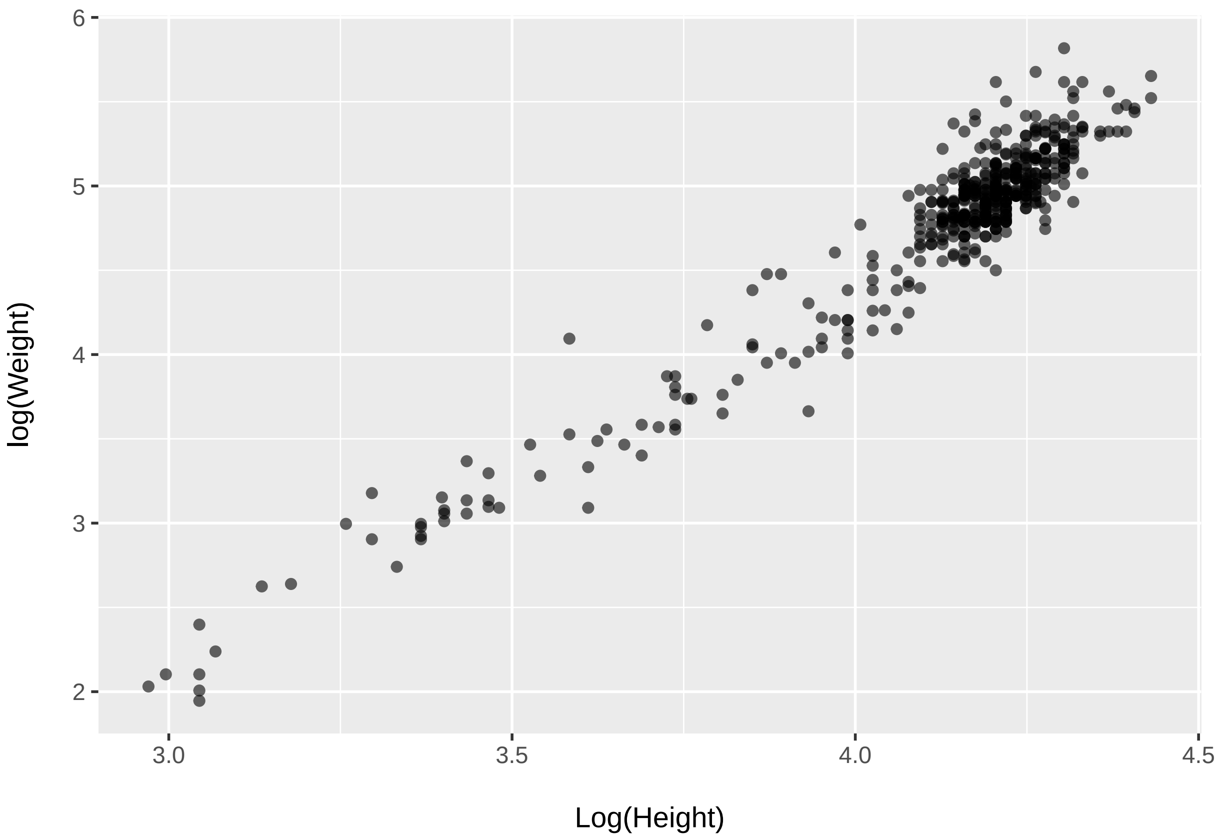
```r
# generate the scatter plot with ggplot2
ggplot(data = handw,                                     # specify data source
       mapping = aes(x = height_in, y = weight_lbs)) +  # specify x and y axis
  geom_point(alpha = .6) +                               # modify transparency of points
  labs(x = "\nHeight (inches)",                          # supply axis label names
       y = "Weight (pounds)\n")
```



- It is evident from the scatter plot of the distribution that the relationship between heights and weights is not quite linear, so let's log transform the variables.

9 Scatter plot of the log of handw data

```
# add log transformed vectors to dataset
handw_log <-
  handw |>
  mutate(hlog = log(height_in),   # create new variable `hlog` which is the log of height_in
         wlog = log(weight_lbs)) # create new variable `wlog` which is the log of weight_lbs

# generate a scatter plot of the log data
ggplot(data = handw_log,                      # don't forget to enter the new dataset
       mapping = aes(x = hlog, y = wlog)) +
  geom_point(alpha = .6) +
  labs(x = "\nLog(Height)",
       y = "log(Weight)\n")
```



10 Using the MASS::mvrnorm command

- The MASS package provides a function mvronrm which stands for multivariate + rnorm.
- MASS is a large package in R, so for efficiency let's only load the required components by using the argument MASS::mvrnorm instead of library("MASS").

- This is also a handy way to proceed as there are some annoying package conflicts between `dplyr` and `MASS` that we want to avoid.

11 `MASS::mvrnorm` arguments

- The three arguments to take note of are:
  - `n` = number of samples required
  - `mu` = a vector giving the means of the variables
  - `Sigma` = a positive-definite symmetric matrix specifying the covariance of the variables
- *Positive-Definite Symmetric Matrix*
  - A covariance matrix (also known as the variance-covariance matrix) specifying the variances of the individual variables and their inter-relationships.
  - It is essentially a multi-dimensional version of standard deviation.

11.1 Out of interest, what about the relationship between 2+ variables?

For a multivariate distribution with more than two variables you need

- The means for all of the variables.
- Their standard deviations.
- All possible pairwise correlations between the variables.

12 Matrix Calculations for the *Sigma* argument

A covariance matrix can be calculated using the following formula:

$$\sum = \begin{pmatrix} \sigma_x^2 & \rho_{xy}\sigma_x\sigma_y \\ \rho_{yx}\sigma_y\sigma_x & \sigma_y^2 \end{pmatrix}$$

- σ_x^2 = squared SD for x.
- σ_y^2 = squared SD y.
- $\rho_{xy}\sigma_x\sigma_y$ = the co-variances (i.e., the correlation multiplied by the two standard deviations, shown in the off-diagonal.
- It is worth saying here that the **covariance is just the correlation times the product of the two standard deviations.**

13 Gathering the statistics

- Let's start by gathering the statistics we need to simulate the data using `MASS::mvrnorm`.
- Remember, we need the
  - `mean`
  - `sd`
  - `Sigma`
- We will continue with the log of the data, as the relationship is more linear.

```
# calculate means and sd
handw_log |>
  summarise(mean_h = mean(hlog),
            sd_h = sd(hlog),
            mean_w = mean(wlog),
            sd_w = sd(wlog)) |>
  mutate_if(is.numeric, round, digits = 2) # round to 2 decimal places
```

```
## # A tibble: 1 x 4
##   mean_h  sd_h mean_w  sd_w
##    <dbl> <dbl>  <dbl> <dbl>
## 1   4.11  0.26   4.74  0.65
```

```
# calculate correlation
cor(handw_log$hlog, handw_log$wlog)
```

```
## [1] 0.9615714
```

14 Calculation output

- $\bar{x} = 4.11, \sigma_x = 0.26$ (mean and SD of log height)
- $\bar{y} = 4.74, \sigma_y = 0.65$ (mean and SD of log weight)
- $\rho_{xy} = 0.96$ (correlation between the two)

15 Calculating *Sigma* for `MASS:mvrnorm`

- We now have all of the information we need to simulate the height and weight of, let's say 500 humans.

- One last piece in the puzzle is to create the covariance matrix to supply to the `Sigma` argument.
- Let's plug in the output values we calculated previously into the covariance matrix formula.

16 Enter values into the formula

$$\sum = \begin{pmatrix} \sigma_x^2 & \rho_{xy}\sigma_x\sigma_y \\ \rho_{yx}\sigma_y\sigma_x & \sigma_y^2 \end{pmatrix}$$

- So plugging in the values we got above, our covariance matrix should be

$$\sum = \begin{pmatrix} .26^2 & (.96)(.26)(.65) \\ (.96)(.65)(.26) & .65^2 \end{pmatrix} = \begin{pmatrix} .067 & .162 \\ .162 & .423 \end{pmatrix}$$

17 Create covariance matrix for the `MASS::mvrnorm` argument in R

```
# define and store covariances
my_cov <-
  .96 * .26 * .65 # log of correlation times hlog_sd times wlog_sd

# use the matrix function to define our sigma
my_sigma <-
  matrix(c(.26^2, my_cov,
           my_cov, .65^2),
         ncol = 2)

my_sigma          # print the matrix
```

```
##         [,1]    [,2]
## [1,] 0.06760 0.16224
## [2,] 0.16224 0.42250
```

17.1 Some notes about the `matrix` function

- The first argument is a vector of values, which we created using `c()`.
- The `ncol` argument specifies how many columns the matrix should have.
- `matrix` fills the elements of the matrix by column by column, rather than row by row.
- You can change this behaviour if desired by changing the byrow argument to `byrow = TRUE`.

18 Simulate data

OK, so now we have `my_sigma` we're ready to use `MASS::mvrnorm`. Let's test it by creating 6 synthetic humans.

```r
# pass the names vector c(height = 4.11, weight = 4.74)
# for mu gives us column names in the output
log_ht_wt <-
  MASS::mvrnorm(n = 6,                    # our 6 synthetic humans
                mu = c(height = 4.11,     # log mean of height
                       weight = 4.74),    # log mean of weight
                Sigma = my_sigma)         # our positive-definitive matrix

# view the output
log_ht_wt
```

```
##        height   weight
## [1,] 4.046250 4.964822
## [2,] 3.946070 4.613247
## [3,] 4.582814 5.702836
## [4,] 4.520118 5.680369
## [5,] 4.134257 4.573500
## [6,] 4.427730 5.442189
```

- `MASS::mvrnorm` returns a matrix with a row for each simulated human, with the first column representing the log height and the second representing the log weight.
- But the log heights and weights are not very useful to us, so let's transform them back using the `exp()`, which is the inverse of `log()` transform.

```r
exp(log_ht_wt)
```

```
##        height   weight
## [1,] 57.18263 143.2831
## [2,] 51.73166 100.8109
## [3,] 97.78919 299.7163
## [4,] 91.84645 293.0574
## [5,] 62.44320  96.8826
## [6,] 83.74115 230.9473
```

```r
# remember height is measured in inches
# weight is measured in pounds
```

19 Simulation considerations

- Note, that there will be some unusual observations generated, with strangely high or low values for height and weight.
- However, you can rest easy knowing that the weight/height relationship, as specified by the variance vocariance matrix, has been preserved.

20 Simulating a large dataset

Finally, let's simulate a group of 500 humans, convert their values from the log space to the real space (e.g., inches and pounds), and plot a comparison between the original data and our simulated data.

```r
# simulate new humans
new_humans <-
  MASS::mvrnorm(n = 500,
                mu = c(height_in = 4.11,
                       weight_lbs = 4.74),
                Sigma = my_sigma) |>
  exp() |>                          # back-transform from log space to real space
  as_tibble() |>                    # make a tibble for plotting
  mutate(type = "simulated")         # vector labelling data as simulated

# examine data
headTail(new_humans)
```

```
##   height_in weight_lbs      type
## 1      63.1     106.76 simulated
## 2    131.15     655.42 simulated
## 3     80.06     320.35 simulated
## 4     43.88      69.69 simulated
## 5       ...        ...      <NA>
## 6     67.52      145.5 simulated
## 7      81.2     199.54 simulated
## 8     50.54      93.26 simulated
## 9     66.82     145.88 simulated
```

21 Combine the real and simulated data

```
# combine real and simulated datasets
# Note: `handw` is a table containing data from heights_and_weights.csv
all_data <-
  bind_rows(handw |>
              mutate(type = "real"), # vector labelling data as real
              new_humans)
```

- What do you think `bind_rows` is doing here?

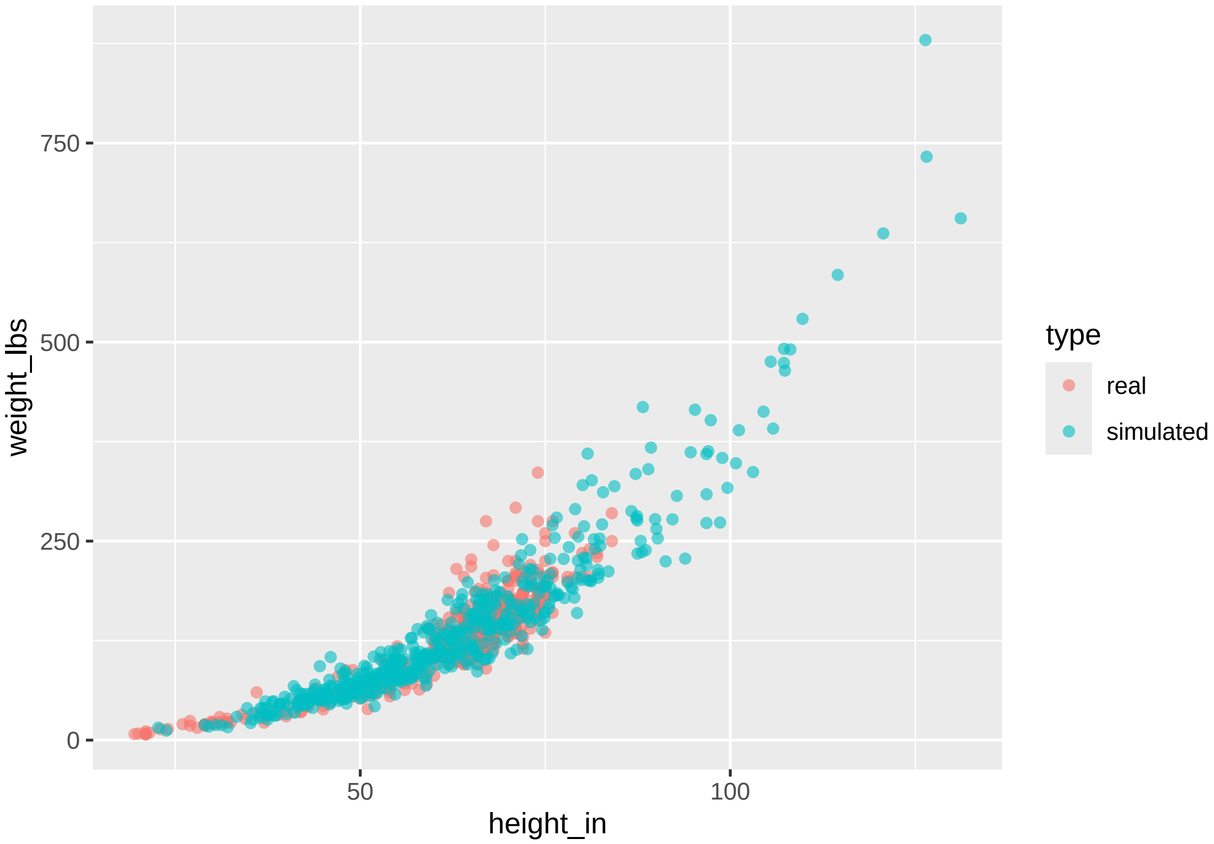
22 Examined combined data

```
# examine tibble
headTail(all_data)
```

```
##   height_in weight_lbs      type
## 1        63        130      real
## 2        67        169      real
## 3        71        178      real
## 4        71        225      real
## 5       ...        ...      <NA>
## 6     67.52      145.5 simulated
## 7      81.2     199.54 simulated
## 8     50.54      93.26 simulated
## 9     66.82     145.88 simulated
```

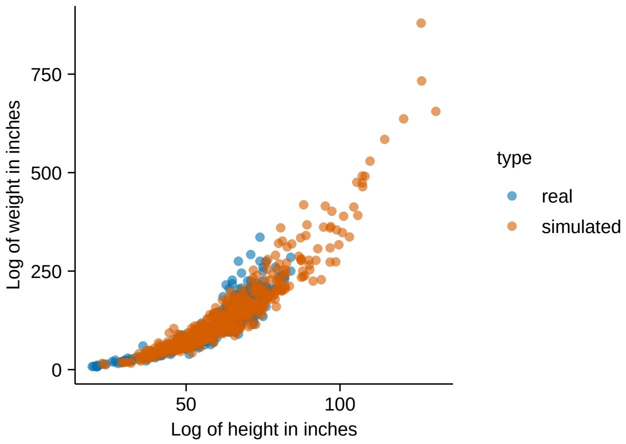
23 Plot the real and simulated data

```
# plot the data
ggplot(all_data,
       aes(x = height_in,
           y = weight_lbs)) +
  geom_point(aes(colour = type),
             alpha = .6)
```



24 With tidyplots

```
# plot the data
all_data |>
  tidyplot(x = height_in,
           y = weight_lbs,
           color = type) |>
  add_data_points(alpha = .6) |>
  adjust_x_axis_title("Log of height in inches") |>
  adjust_y_axis_title("Log of weight in inches")
```

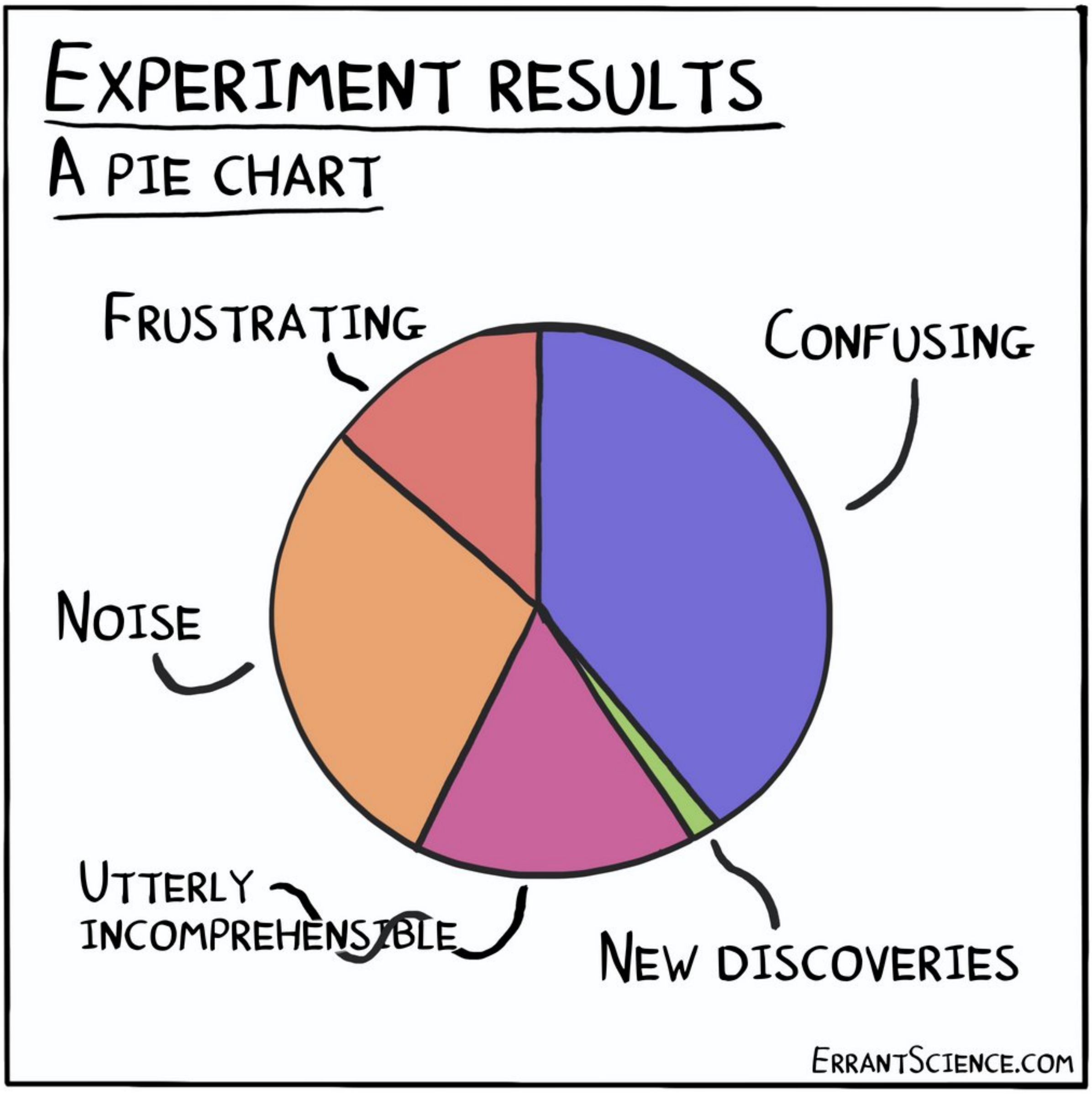


25 Save the simulated dataset

```
# write data as a csv into "data" folder
write_csv(all_data, "data_tidy/simulated_handw.csv")
```

26 Round-up and conclusion

- In the process of showing you how to simulate bivariate data you become more familiar with covariance and matrix calculations.
- This is just an example of how data simulation can develop statistical your expertise.
- The end result is something you can use for research (e.g., your analysis script) or for teaching your class (e.g., the dataset).



27 Thank you!

*Inspiration for today's session on Data Simulation:*Learning Statistical Models Through Simulation in R: Correlation and Regression

PsyPag & MSCP-Section Simulation Summer School

Data Simulation Workshops

Heights and weights dataset