simon-5505-04-slides

Topics to be covered

- What you will learn
  - Review statistics for continuous outcomes
  - Counts, proportions, and percentages
  - Crosstabulations
  - Bar plots
  - New categorical variables

The Titanic dataset, 1

The Titanic was a large cruise ship, the biggest of its kind in 1912. It was thought to be unsinkable, but when it set sail from England to America in its maiden voyage, it struck an iceberg and sank, killing many of the passengers and crew. You can get fairly good data on the characteristics of passengers who died and compare them to those that survived. The data indicate a strong effect due to age and gender, representing a philosophy of “women and children first” that held during the boarding of life boats.

Speaker notes

Here is the description of the Titanic dataset, found in the data dictionary that I created.

The Titanic dataset, 2

```
Rows: 1,313
Columns: 5
$ name     <chr> "Allen, Miss Elisabeth Walton", "Allison, Miss Helen
Loraine"…
$ pclass   <chr> "1st", "1st", "1st", "1st", "1st", "1st", "1st", "1st",
"1st"…
$ age      <dbl> 29.00, 2.00, 30.00, 25.00, 0.92, 47.00, 63.00, 39.00, 58.00,
…
$ sex      <chr> "female", "female", "male", "female", "male", "male",
"female…
$ survived <dbl> 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1,
1…
```

Speaker notes

Here are the key variables.

Peek at the bottom

```
ti |>
  slice_tail(n=10)
```

Speaker notes

Comments on the code and interpretation of the output appears in simon-5505-04-demo.

Descriptive statistics on age

```
ti |>
  summarize(
    age_mean=round(mean(age, na.rm=TRUE), 1),
    age_sd=round(sd(age, na.rm=TRUE), 1),
    age_min=min(age, na.rm=TRUE),
    age_max=max(age, na.rm=TRUE),
    age_missing=sum(is.na(age)))
```

Print out information on youngest and oldest passengers

```
ti |>
  slice_min(age, na_rm=TRUE)

ti |>
  slice_max(age, na_rm=TRUE)
```

Break #1

- What you have learned
  - Review statistics for continuous outcomes
- What’s coming next
  - Counts, proportions, and percentages

Categorical versus continuous variables

- Categorical
  - Small number of possible values
  - Each value has a name or label
- Continuous
  - Large number of possible values
  - Potentially any value in a range.

Speaker notes

A categorical variable is defined (loosely) as a variable that has a small number of possible values. Each value is usually associated with a particular category or label. In contrast, a continuous variable is defined as a variable that has a large number of possible values, potentially any value in a particular interval.

In a previous module, almost all of the variables that you used were continuous. Today, almost all of the variables that you will use will be categorical.

The distinction between continuous and categorical variables is important in deciding what types of descriptive and inferential statistics you should use. But, there is often gray and fuzzy line between categorical and continuous variables. Don’t worry too much about this today. If you’re not sure whether a variable is categorical or continuous, try some simple descriptive statistics and graphs appropriate for categorical data and then try some simple descriptive statistics and graphs for continuous data. You will usually figure out quickly whether treating your variable as categorical or continuous makes the most sense.

A hypothetical dataset

```
# A tibble: 10 × 1
   grp
   <chr>
 1 a
 2 b
 3 b
 4 c
 5 c
 6 c
 7 d
 8 d
 9 d
10 d
```

Counts and percentages, 1

```
hypo_1 |>
  count(grp)
```

```
# A tibble: 4 × 2
  grp       n
  <chr> <int>
1 a         1
2 b         2
3 c         3
4 d         4
```

Speaker notes

For categorical variables, you should first get frequency counts. A mean and standard deviation are usually meaningless for categorical data.

Unlike most other statistical packages, R tends to have a minimalist approach to statistics. If you asked for frequency counts in SAS or SPSS, these systems would automatically add percentages. R doesn't add percentages automatically.

This is something that you will either love or hate. You might think that SAS and SPSS are more thoughtful because almost every time you want a count, you'd also want the corresponding percentage. Or you might find it annoying to tell those programs to not clutter up your output with information you didn't want.

Personally, I don't like software deciding for me what I want. I'd rather ask for percentages explicitly when I need them rather than have them come as the default.

Now this is a rather trivial issue, but it does illustrate an important difference in philosophy. R makes you ask for the extras that you might need. SAS and SPSS force you to ask to NOT include things that they think are important.

Counts and percentages, 2

```
hypo_1 |>
  count(grp) |>
  mutate(total=sum(n)) |>
  mutate(pct=100*n/total)
```

```
# A tibble: 4 × 4
  grp       n total   pct
  <chr> <int> <int> <dbl>
1 a         1    10    10
2 b         2    10    20
3 c         3    10    30
4 d         4    10    40
```

Speaker notes

Most other programs will compute percentages automatically or as part of the same procedure that gave you the counts. As I have mentioned many times, R takes a minimalist approach. It gives you counts with one function but if you want percentages, you have to ask for it. In this example, you add a couple of mutate functions.

Counts and percentages, 3

```
hypo_1 |>
  count(grp) |>
  mutate(total=sum(n)) |>
  mutate(pct=100*n/total) |>
  mutate(pct=glue("{n}/{total} ({round(pct)}%)")) |>
  select(-n, -total)
```

```
# A tibble: 4 × 2
  grp   pct
  <chr> <glue>
1 a     1/10 (10%)
2 b     2/10 (20%)
3 c     3/10 (30%)
4 d     4/10 (40%)
```

Speaker notes

Another nice feature of R is that if you don’t like what the output looks like, you have many different ways to improve on it.

Break #2

- What you have learned
  - Counts, proportions, and percentages
- What’s coming next
  - Crosstabulations

Hypothetical dataset

```
# A tibble: 10 × 2
   intervention result
   <chr>        <chr>
 1 c            f
 2 c            s
 3 c            s
 4 t            f
 5 t            f
 6 t            f
 7 t            s
 8 t            s
 9 t            s
10 t            s
```

Speaker notes

Here is a simple hypothetical data set. The variable “intervention” represents whether a patient was a “c” or a “t”. Think of this as control and treatment. The variable “result” represents whether the patient was an “f” or an “s”. Maybe think of this as failure and success. The first row represents the single patient that is a “cf” or a control failure. The last four rows represent patients that are a “ts” or a treatment success.

Counts by intervention and result

```
hypo_2 |>
  count(intervention, result)
```

```
# A tibble: 4 × 3
  intervention result     n
  <chr>        <chr>  <int>
1 c            f          1
2 c            s          2
3 t            f          3
4 t            s          4
```

Speaker notes

A crosstabulation shows counts across the combination of two different categorical variables. Only one patient was a control failure, two were control successes, three were treatment failures and four were treatment successes.

Counts by result and intervention

```
hypo_2 |>
  count(result, intervention)
```

```
# A tibble: 4 × 3
  result intervention     n
  <chr>  <chr>        <int>
1 f      c                1
2 f      t                3
3 s      c                2
4 s      t                4
```

Speaker notes

There is a subtle change when you switch the order of the two arguments in the count function, but the data is essentially equivalent.

Rectangular layout, rows are result

```
hypo_2 |>
  count(intervention, result) |>
  pivot_wider(
    names_from=intervention,
    values_from=n)
```

```
# A tibble: 2 × 3
  result     c     t
  <chr>  <int> <int>
1 f          1     3
2 s          2     4
```

Speaker notes

Most other programs arrange the counts automatically in a rectangular grid. In this case where you have two levels for each categorical variable, it would be a square grid, but that’s a nitpick.

R is a minimalist package and you have to convert the counts from a single column to a rectangular grid with an extra function, pivot_wider.

Rectangular layout, rows are outcome

```
hypo_2 |>
  count(intervention, result) |>
  pivot_wider(
    names_from=result,
    values_from=n)
```

```
# A tibble: 2 × 3
  intervention     f     s
  <chr>        <int> <int>
1 c                1     2
2 t                3     4
```

Conditional probabilities, 1

```
hypo_2 |>
  count(intervention, result) |>
    group_by(intervention) |>
    mutate(total=sum(n)) |>
    mutate(pct=round(100*n/total)) -> prob_1

prob_1
```

```
# A tibble: 4 × 5
# Groups:   intervention [2]
  intervention result     n total   pct
  <chr>        <chr>  <int> <int> <dbl>
1 c            f          1     3    33
2 c            s          2     3    67
3 t            f          3     7    43
4 t            s          4     7    57
```

Speaker notes

There are two different ways to compute conditional probabilities. The first is to divide by the number of controls (3) or the number of treated (7).

Conditional probabilities, 2

```
hypo_2 |>
  count(intervention, result) |>
    group_by(result) |>
    mutate(total=sum(n)) |>
    mutate(pct=round(100*n/total)) -> prob_2

prob_2
```

```
# A tibble: 4 × 5
# Groups:   result [2]
  intervention result     n total   pct
  <chr>        <chr>  <int> <int> <dbl>
1 c            f          1     4    25
2 c            s          2     6    33
3 t            f          3     4    75
4 t            s          4     6    67
```

Speaker notes

You could also divide the number of failures (4) or the number of successes (6). This is more than a trivial distinction.

Interpretation, 1

```
# A tibble: 4 × 6
# Groups:   intervention [2]
  intervention result     n total   pct interpretation
  <chr>        <chr>  <int> <int> <dbl> <chr>
1 c            f          1     3    33 P[f|c]
2 c            s          2     3    67 P[s|c]
3 t            f          3     7    43 P[f|t]
4 t            s          4     7    57 P[s|t]
```

Speaker notes

These probabilities are conditional probabilities. You are looking at the probability of failure or success conditional on the intervention. The overall probability of success is 60%, but when you restrict your attention to just the controls, it jumps to 67%. When you restrict your attention to just the treatment it drops to 57%. The difference is 10%, showing that compared to the controls, the treatment is much less effective.

Interpretation, 2

```
# A tibble: 4 × 6
# Groups:   result [2]
  intervention result     n total   pct interpretation
  <chr>        <chr>  <int> <int> <dbl> <chr>
1 c            f          1     4    25 P[c|f]
2 c            s          2     6    33 P[c|s]
3 t            f          3     4    75 P[t|f]
4 t            s          4     6    67 P[t|s]
```

Rectangular arangement of probabilities, 1

```
# A tibble: 2 × 3
# Groups:   intervention [2]
  intervention     f     s
  <chr>        <dbl> <dbl>
1 c               33    67
2 t               43    57
```

Speaker notes

Here is where it gets tricky. The percentages can be arranged with this orientation…

Rectangular arangement of probabilities, 2

```
# A tibble: 2 × 3
  result     c     t
  <chr>  <dbl> <dbl>
1 f         33    43
2 s         67    57
```

Speaker notes

… or this orientation. Which is better?

While both orientations are okay, I have a definite preference for the first one. It all relates to the proximity principle. Place your most interesting comparisons in close proximity.

What is the most interesting comparison? It is the how much larger the success rate in the control group (67%) is compared to the success rate in the treatment group (57%).

The proximity principle, 1

```
# A tibble: 2 × 3
# Groups:   intervention [2]
  intervention     f     s
  <chr>        <dbl> <dbl>
1 c                   67
2 t                   57
```

```
# A tibble: 2 × 3
  result           c     t
  <chr>        <dbl> <dbl>
1 f
2 s               67    57
```

Speaker notes

Here are the two tables with only 67% and the 57% showing. Notice how much closer the two numbers are when one is directly beneath the other rather than the two numbers side by side.

The proximity principle, 2

```
          1256
           123


1256   123
```

Speaker notes

This is a general rule for other numbers as well. Notice how the extra digit in the 1256 stands out when you have 983 directly beneath it compared to when the two numbers are side by side.

There will always be exceptions to the proximity principle, but it is always worth considering.

Which percentages should you use

- General guidance
  - Set the rows to your treatment/exposure
  - Set the columns to your outcome
  - Compute row percentages
- why not try several formats
  - Revised your tables as often as you revise your writing

Speaker notes

It’s beyond the scope of this class, but with crosstabulations, you have choices as to what should be the rows and what should be the columns. Then you can compute row, column, or cell percentages.

I’ve found that nine times out of ten, the best choice depends on what is your treatment variable and what is your outcome. It usually works best if you place the treatment variable in the rows and the outcome in the columns and compute row percentages. That shows how often you see a particular outcome in the treatment group and the percentage in the control group is right beneath it.

That being said, I would also encourage you to try several different approaches.

Break #3

- What you have learned
  - Crosstabulations
- What’s coming next
  - Bar plots

Hypothetical datasets, 3

```
# A tibble: 9 × 2
  intervention result
  <chr>        <chr>
1 c            f
2 c            f
3 c            s
4 t1           f
5 t1           s
6 t1           s
7 t1           s
8 t2           f
9 t2           s
```

Barplot of counts, code

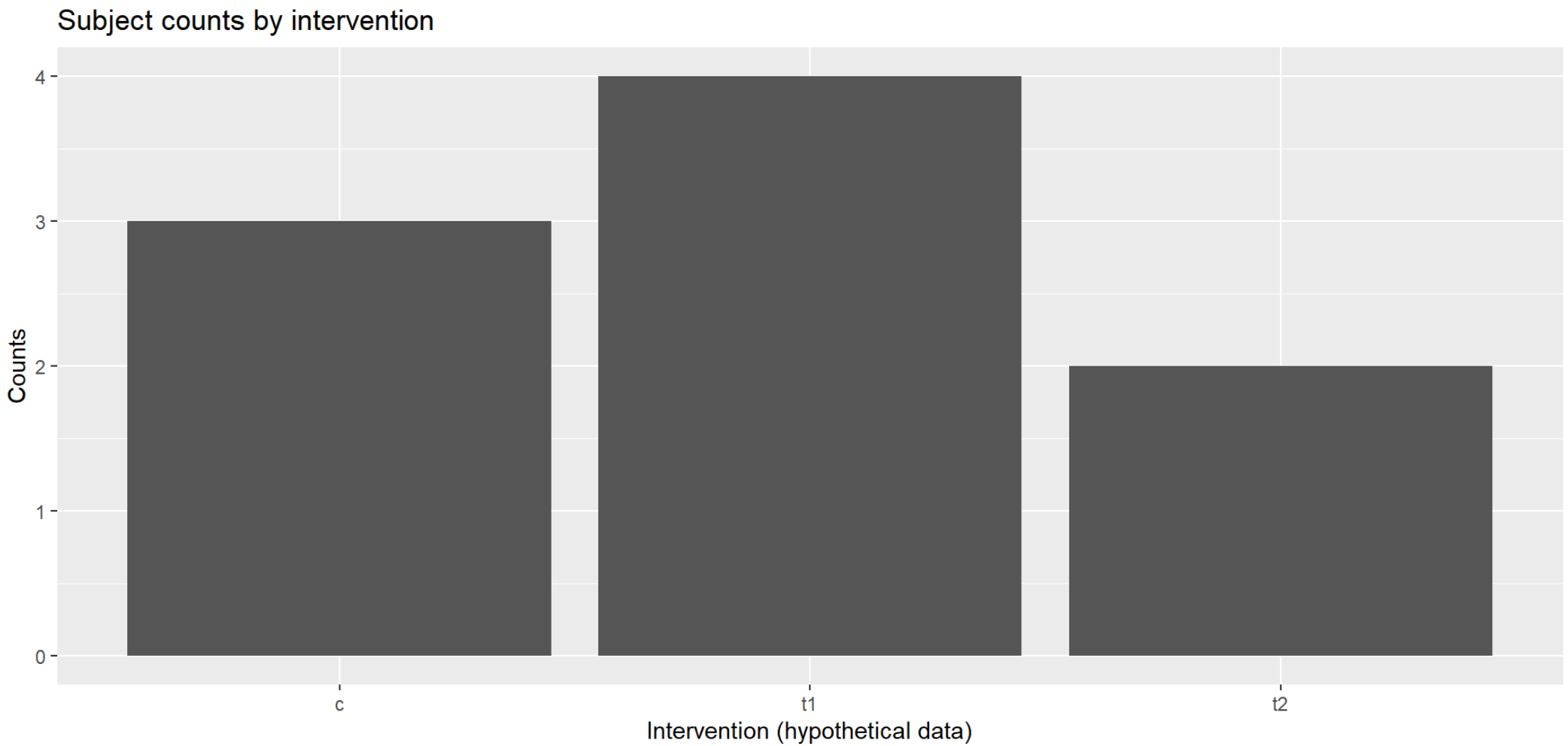
```
hypo_3 |>
  ggplot() +
  aes(x=intervention) +
  geom_bar() +
  labs(
    title = "Subject counts by intervention",
    x = "Intervention (hypothetical data)",
    y = "Counts",
    caption = "Simon, 2025-04-17") -> bar_1
```

Speaker notes

I’m not a big fan of bar plots, but they sometimes have their uses. You can get a barplot for the frequency count.

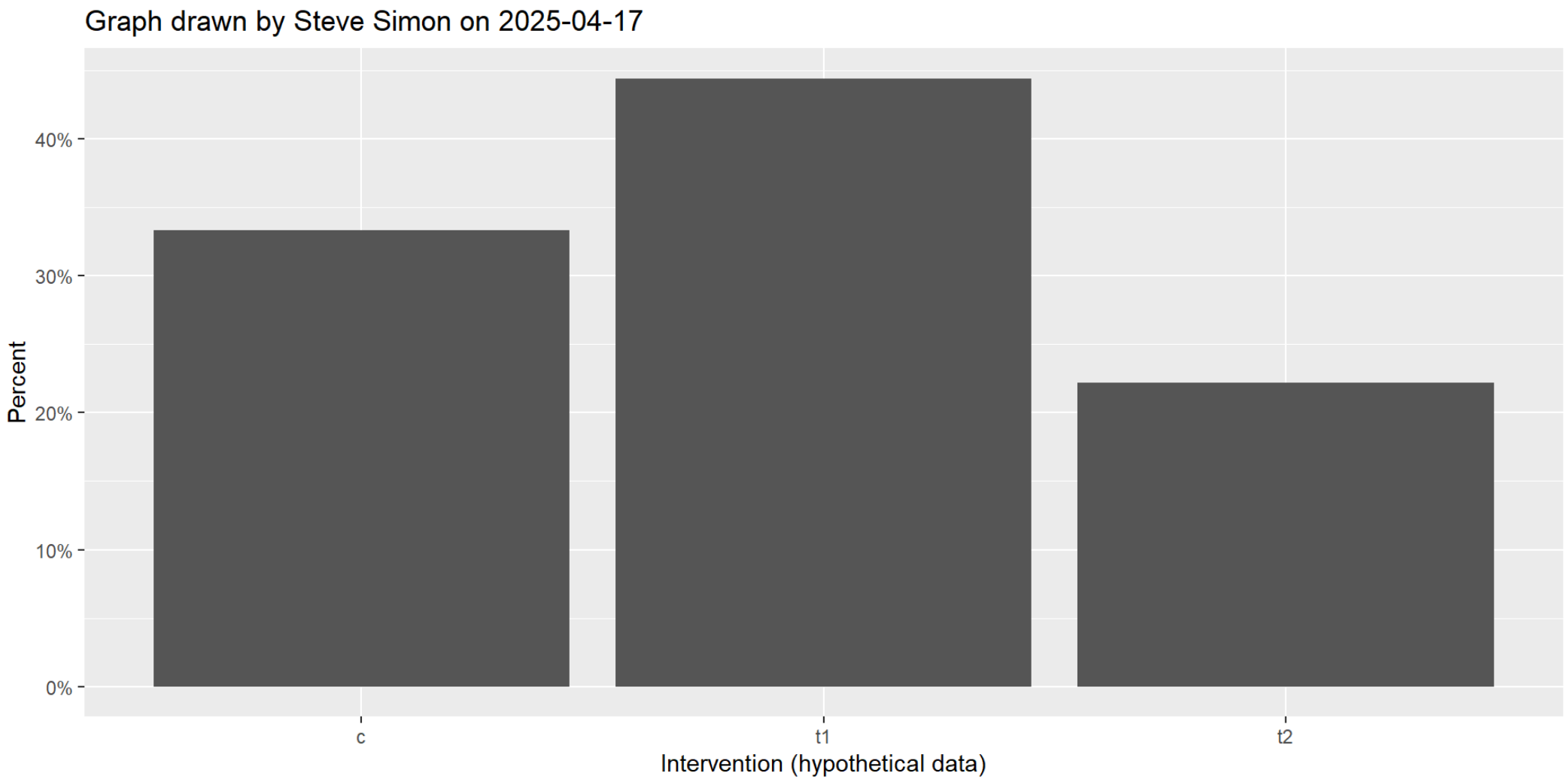
Because of the way Rmarkdown displays graphs, I have to put the code on a separate slide from the graph.

Barplot of counts, plot



Simon, 2025-04-17

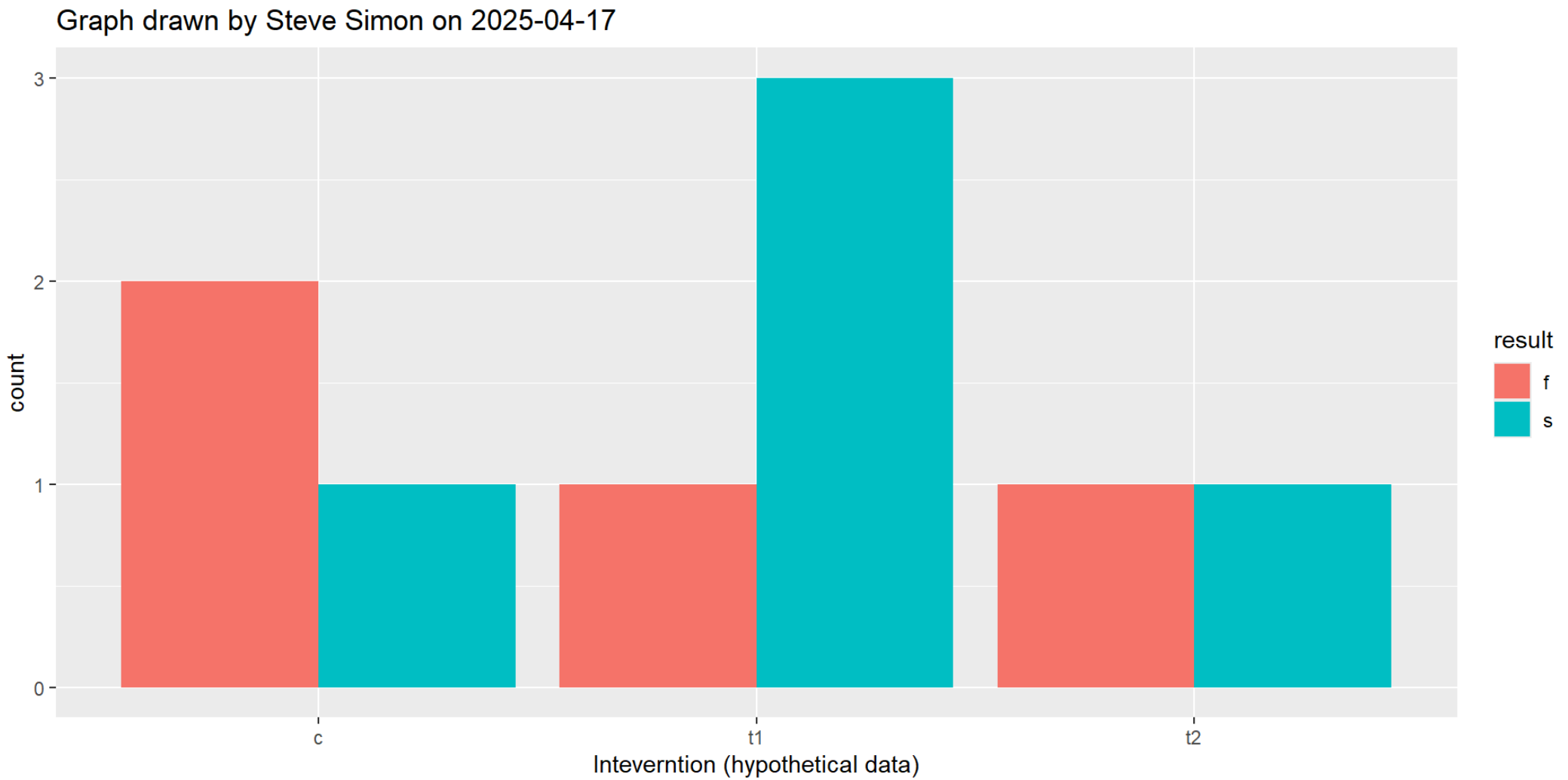
Barplot of percents



Speaker notes

Here is what the plot looks like.

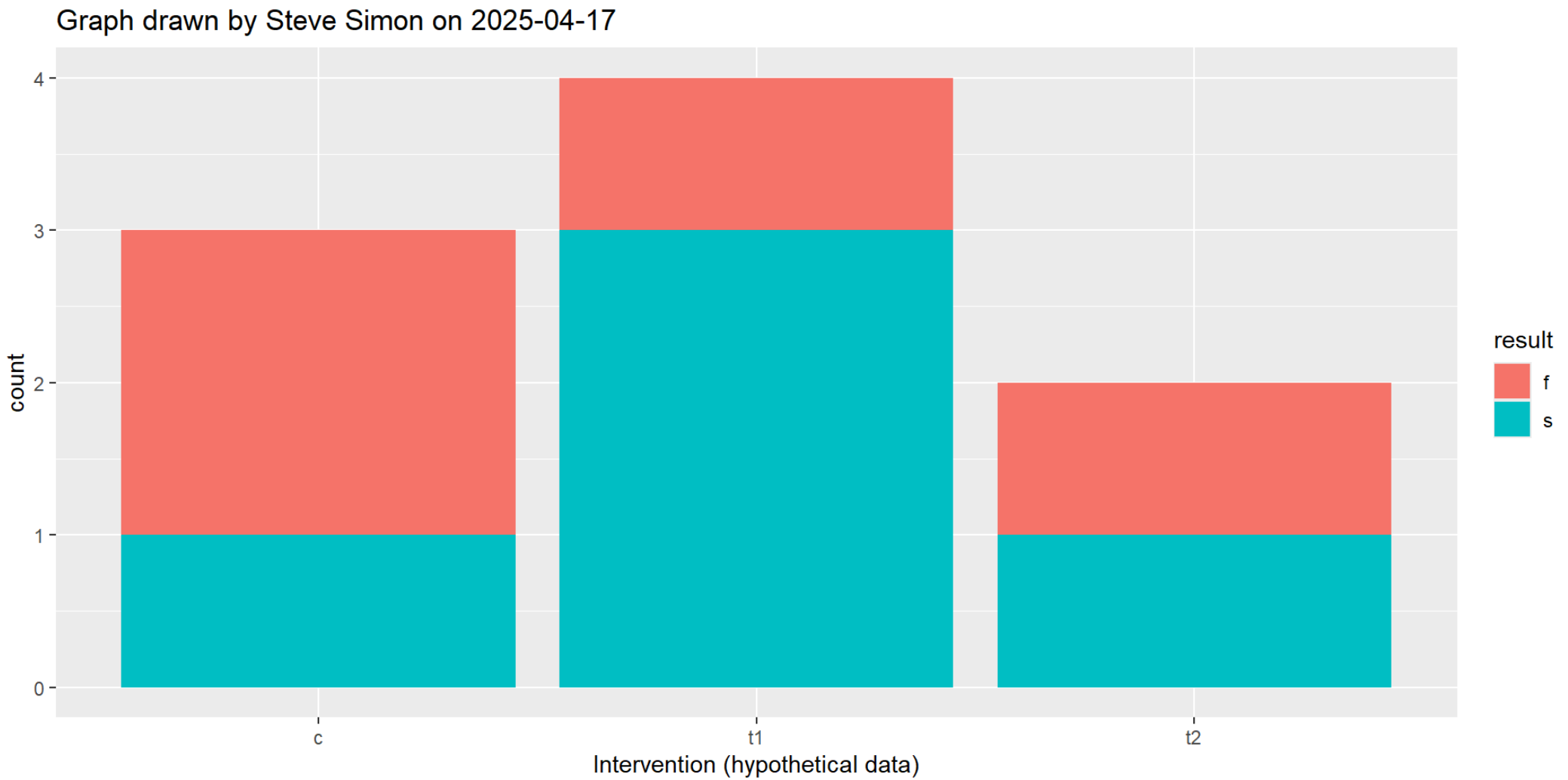
Barplot with two categorical counts, 1



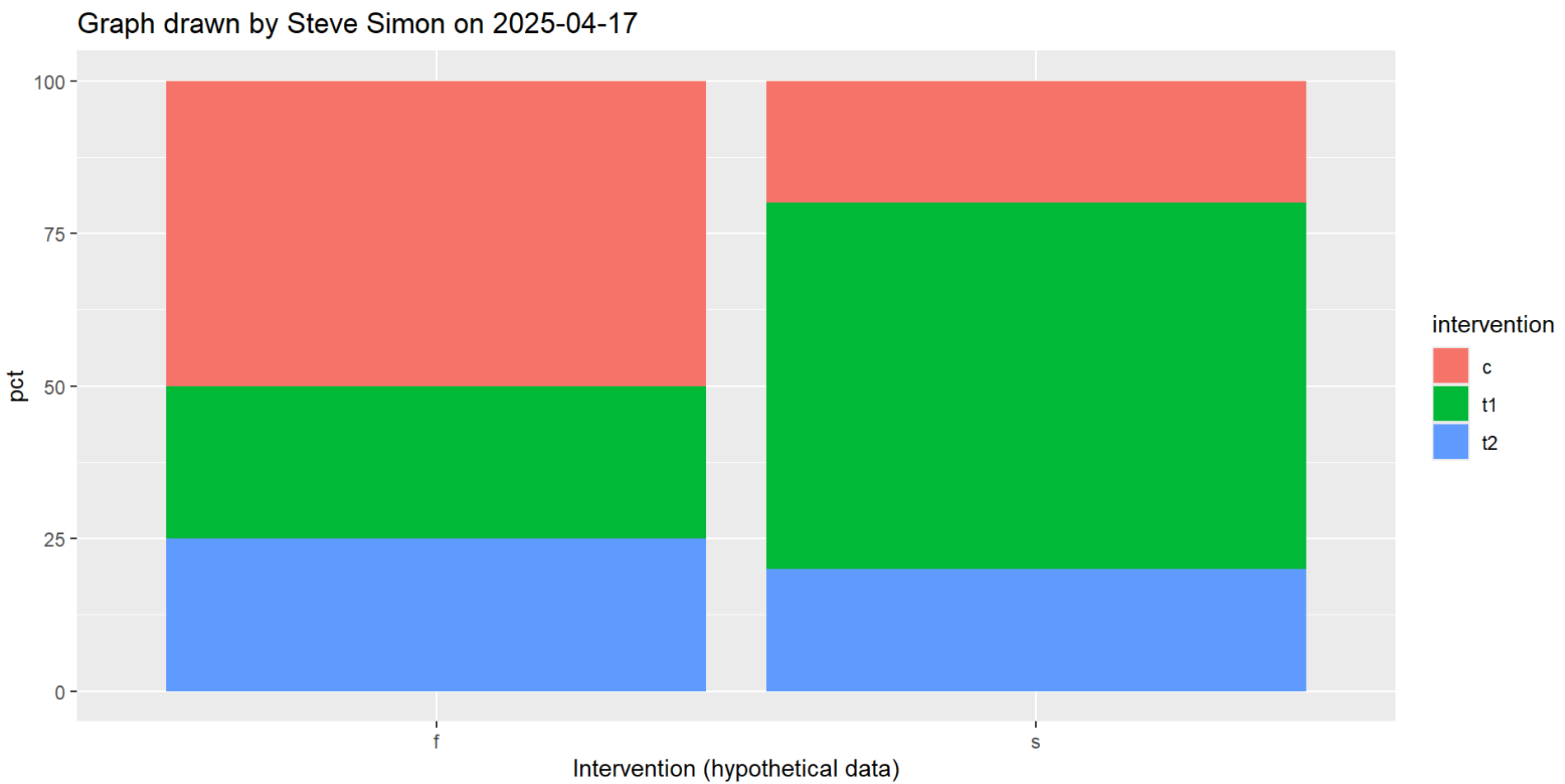
Speaker notes

Here is what the plot looks like.

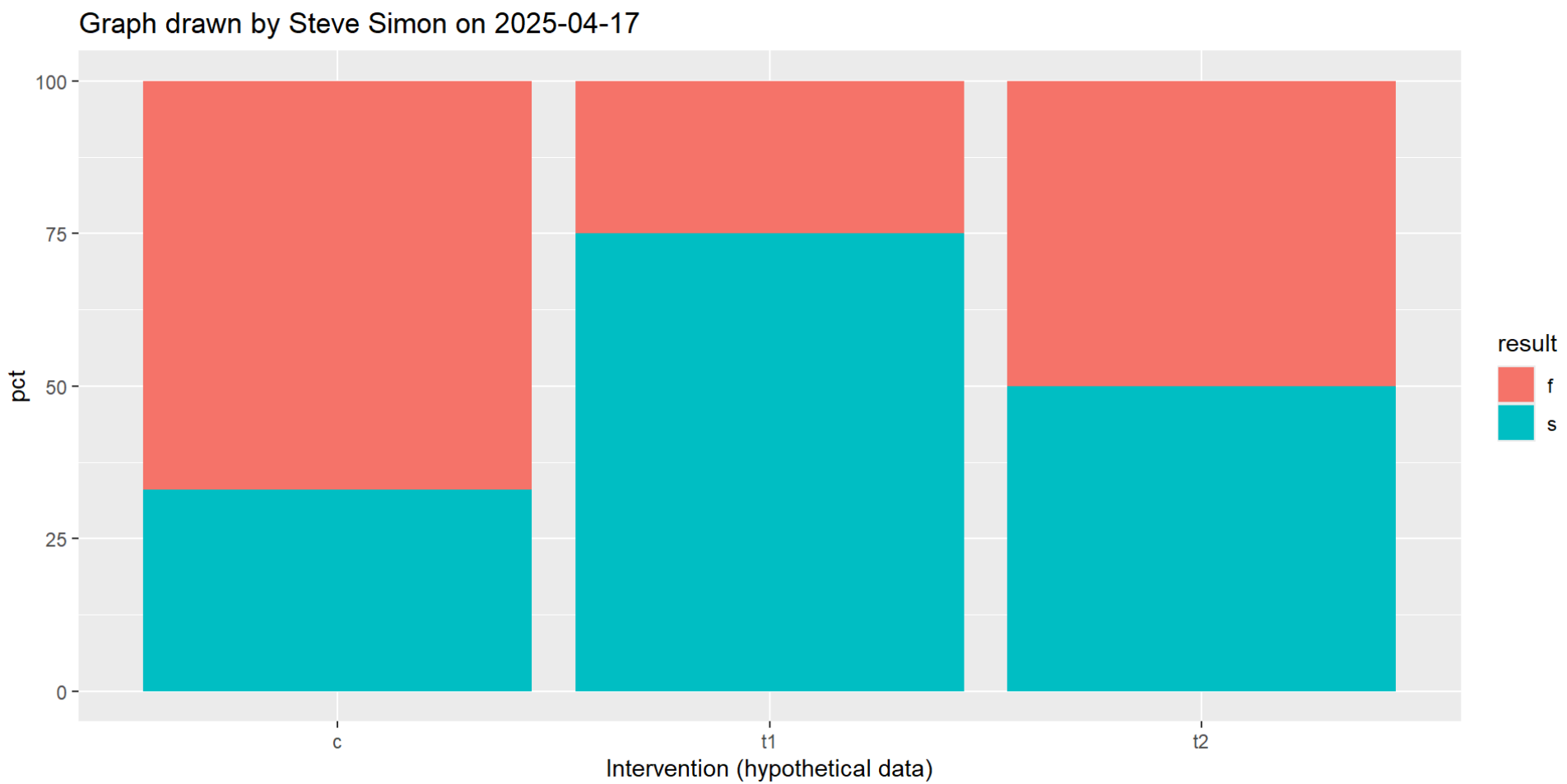
Barplot with two categorical counts, 2



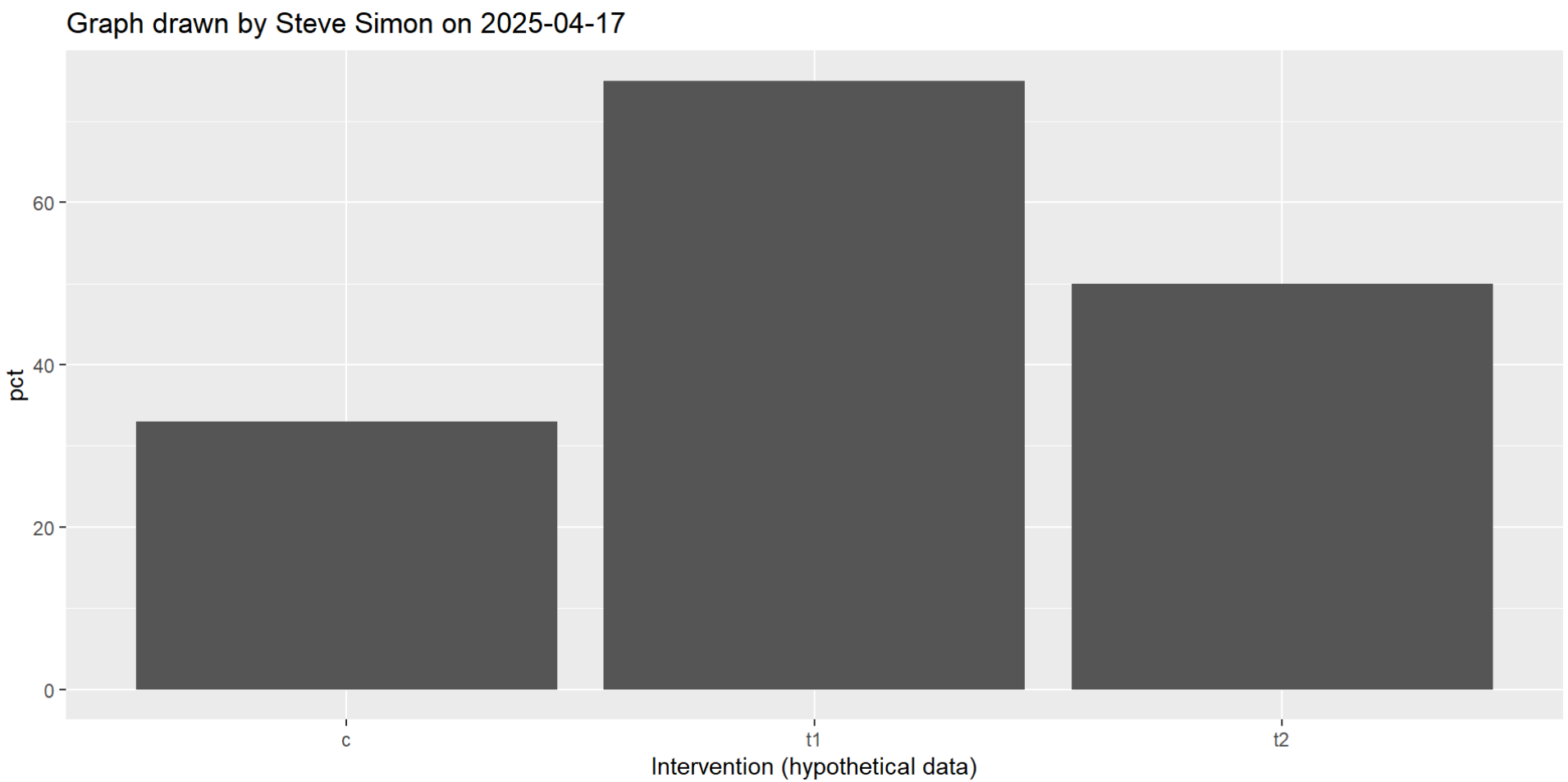
Barplot with two categorical percentages, 1



Barplot with two categorical percentages, 2



Barplot with two categorical percentages, 2



Break #4

- What you have learned
  - Bar plots
- What’s coming next
  - New categorical variables

Converting a continuous variable to categorical

```
ti_1 |>
    mutate(child=case_when(
    is.na(age) ~ "Unknown",
    age <=  0  ~ "Invalid",
    age >= 91  ~ "Invalid",
    age <  18  ~ "Yes",
    age >= 18  ~ "No")) -> ti_2
```

Quality check, 1

```
ti_2 |>
  group_by(child) |>
    summarize(
        age_min=min(age),
        age_max=max(age))
```

Combining categories

```
ti_2 |>
    mutate(third_class = case_when(
    is.na(pclass) ~ NA,
    pclass=="1st" ~ 0,
    pclass=="2nd" ~ 0,
    pclass=="3rd" ~ 1)) -> ti_3
```

Quality check, 2

```
ti_3 |>
    count(third_class, pclass)
```

Summary

- What you have learned
  - Review statistics for continuous outcomes
  - Counts, proportions, and percentages
  - Crosstabulations
  - Bar plots
  - New categorical variables