
Do not use averages with Likert scale data

with examples using



Dwight Barry

Do not use averages with Likert scale data

Dwight Barry, PhD

Enterprise Analytics
Seattle Children's Hospital
Seattle, WA, USA

Twitter: @healthstatsdude

About

This is a short overview of why averages don't work well for evaluating Likert scale or other ordinal-scale data, and what to do instead, with examples using R. While the examples are focused on healthcare surveys, the lessons apply to any use of ordinal scale data.

Note: all of the data in this document is fake, created specifically to illustrate particular points.

Website: <https://bookdown.org/Rmadillo/likert/>

Github: <https://github.com/Rmadillo/likert>

Cover image: Gustave Doré, 1863. Illustration 12 for Cervantes's *Don Quixote*. Public Domain.



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 License.

Summary

- Likert and similar ordinal-level scales have a variety of uses, particularly within surveys. They also occur in clinical care, for example, in the use of pain scores.
- When evaluated improperly—particularly through the use of averages—the results can be strikingly misleading. Obviously, misleading results could drive or promote action where none is warranted, and vice versa.
- In nearly all cases, not only is it mathematically wrong, **taking the average of a Likert-scale variable will *not* provide useful answers** to the questions managers can use to make actionable decisions. In essence, the use of averages cannot account for the importance of capturing and understanding variability. Analysts should strive to avoid their use in any reporting solution or analytic product that uses ordinal-scale data.
- Better ways to represent ordinal-value results include histograms of the values themselves, the use of well-supported “top-box”-type proportions, and/or bar charts of percentage by score or score category (e.g., favorable/neutral/unfavorable).
- “Statistical significance” on changes or differences between response groups’ medians or distribution shift can be assessed through non-parametric frequentist tests (e.g., permutation, Mann-Whitney-Wilcoxon), Information Theory, or Bayesian analysis. *t*-tests should never be used on Likert scales because ordinal data does not meet the assumptions of a *t*-test (and when using frequentist tools, one must *also* account for multiple testing to reduce the chance of false positives).
- A good way to remember not to use means on Likert scale data is to think: The average of *Agree* and *Strongly Agree* is **not** *Agree-And-A-Half*.

Data and R packages

```
#### Packages ####
library(grid)
library(nnet)
library(coin)
library(boot)
library(simpleboot)
library(knitr)
library(ggplot2)
library(dplyr)
library(AICcmodavg)
library(polycor)
library(likert)
library(MASS)
library(ordinal)

#### Basic example data set ####
person = c('A','B','C','D','E','F')

# Original
year1 = c(5,4,4,4,4,4)
year2 = c(2,5,5,5,5,4)
year3 = c(3,5,5,5,5,3)
year4 = c(1,5,5,5,5,5)

# A more obvious version
# year1 = c(3,3,3,3,3,3)
# year2 = c(4,4,4,2,2,2)
# year3 = c(5,4,3,3,2,1)
# year4 = c(5,5,5,1,1,1)

ex_1 = data.frame(person, year1, year2, year3, year4)

ex_1_long = reshape2::melt(ex_1)

#### Larger example data set ####

set.seed(29)

md = data.frame(Group = as.character("MD"),
  Response1 = ordered(sample(1:5, 100, replace=T, prob=c(.1,.1,.1,.2,.5))),
  Response2 = ordered(sample(1:5, 100, replace=T, prob=c(.1,.3,.3,.25,.15))))
rn = data.frame(Group = as.character("RN"),
  Response1 = ordered(sample(1:5, 100, replace=T, prob=c(.1,.1,.5,.2,.1))),
  Response2 = ordered(sample(1:5, 100, replace=T, prob=c(.1,.15,.45,.15,.15))))

both = rbind(md, rn)

# Add some NAs
make_NAs = sample(1:200, 15, replace=F)
both$Response1[make_NAs] = NA
```

```

make_NAs2 = sample(1:200, 15, replace=F)
both$Response2[make_NAs2] = NA

# Add question names to data
names(both) = c("EmployeeType",
                "My team works well together.",
                "I have the tools I need to do my job.")

#### Dashboarding pain scores example ####

# Create list for random pain scores
pain_list = list()

for(i in 1:24){
  set.seed(i)
  pain_level = ordered(sample(c("Low", "Medium", "High"), size = sample(10:30),
                             replace = T, prob = c(.15, .45, .40)), levels = c("Low", "Medium", "High"))
  pain_list[[i]] = table(pain_level)
}

# Unlist into a data frame
pain_df = data.frame(matrix(unlist(pain_list), nrow=24, byrow=T))
colnames(pain_df) = c("Low", "Medium", "High")

# Add some months
pain_scores = data.frame(Month = seq(as.Date("2014-10-01"), by = "month",
                                     length.out = 24), pain_df)

# Melt into long form, I really should learn tidyr
pain_scores = reshape2::melt(pain_scores, id.vars = "Month",
                             variable.name = "Pain_Group", value.name = "Count")

# Summarize to get counts and percentages
surgeries_pain = pain_scores %>%
  group_by(Month) %>%
  mutate(Surgeries = sum(Count), percent = (Count / sum(Count)),
         cumsum = cumsum(percent))

#### For use with chi-square and regression models ####

# Get rid of NAs
both2 = na.omit(both)

# Rename columns to something more R-friendly
names(both2) = c("EmployeeType", "Teamwork", "Tools")

# Reverse the levels so 5 will be at top of mosaic plot
both2$Teamwork = ordered(both2$Teamwork, levels = c("5", "4", "3", "2", "1"))

# Make a table object
both2_tab = xtabs(~ both2$EmployeeType + both2$Teamwork)

# For multinomial and prop odds models

```

```

both3 = both2

# Bring axis back to normal
both3$Teamwork = ordered(both3$Teamwork, levels = c("1", "2", "3", "4", "5"))

# Data frame for proportional odds regression
Teamwork_tab_long = both3[,1:2] %>%
  group_by(EmployeeType, Teamwork) %>%
  summarize(Count = n())

# Function to turn counts into rows I found laying around the web somewhere
countsToCases = function(x, countcol = "Count") {
  # Get the row indices to pull from x
  idx = rep.int(seq_len(nrow(x)), x[[countcol]])
  # Drop count column
  x[[countcol]] = NULL
  # Get the rows from x
  x[idx, ]
}

# Make a data frame for prop odds
Teamwork_tab_long$Teamwork_Group = as.numeric(Teamwork_tab_long$Teamwork)
Teamwork_tab_long$Teamwork = ordered(Teamwork_tab_long$Teamwork)
tab_df = data.frame(countsToCases(Teamwork_tab_long, countcol="Count"))

```

Why not? A simple example

Take a simple example where a group of 6 people take the same survey for 4 years, and the mean results for an important question, such as “my team works well together”, are as follows:

Year	Mean
Year 1	4.17
Year 2	4.33
Year 3	4.33
Year 4	4.33

From these values, one might conclude that there is an improvement from year 1 to year 2, and no change year-over-year after year 2.

The values that created the above results are as follows:

Individual	Year 1	Year 2	Year 3	Year 4
A	5	2	3	1
B	4	5	5	5
C	4	5	5	5
D	4	5	5	5
E	4	5	5	5
F	4	4	3	5

You might already see how management decisions would be made differently based on whether one had just the means or had the complete data.

However, in the latter case, you risk reducing or eliminating anonymity, which is essential to get respondents to answer truthfully (not to mention being unethical). Further, poring over tables of answers for many people for long surveys makes that approach practically infeasible.

Visualizing the results in ways that capture a more complete story provides an answer to both issues, as well as providing decision-makers with truly actionable information.

Always visualize

Histograms

Histograms of the actual score values are the best way to visualize Likert data—they have two real axes, showing counts by score value or category, so you can parse the visual and understand the results very quickly. Using the same data as above, you can instantly see that the “improvement” in year 2 was perhaps not an improvement after all: while most respondents appear to be satisfied above what they thought in year 1, one respondent may be at risk of leaving.

```
# Figure 1
ggplot(ex_1_long, aes(value)) +
  geom_histogram(binwidth=1) +
  facet_wrap(~variable, ncol=1) +
  xlab("Likert Scale Value") +
  theme_bw()
```

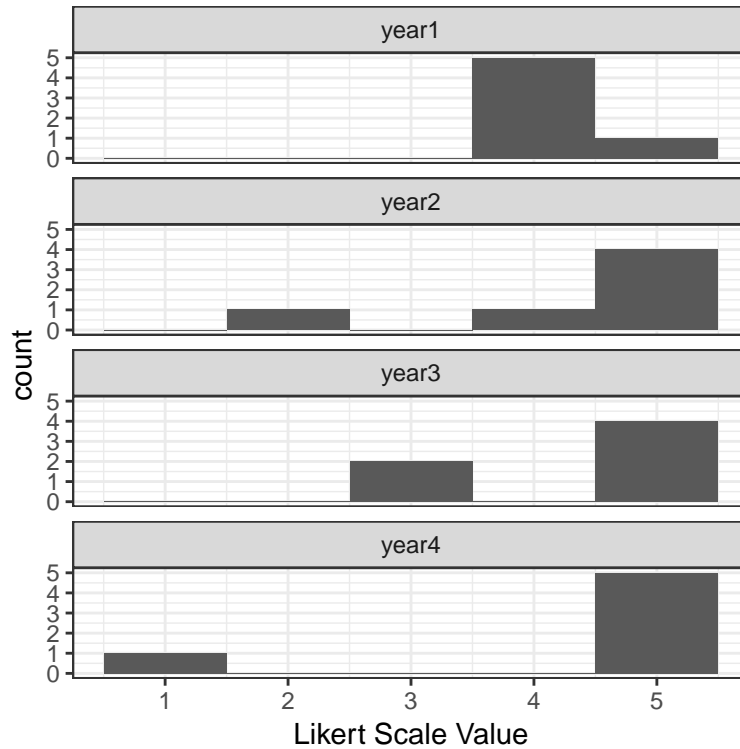


Figure 1: Histogram of simple example Likert scale data.

Likert charts

The main disadvantage of histograms is space; Likert charts—which are in essence just stacked bar charts—are far more compact. The disadvantage is that it takes slightly longer for a user to parse them, but when faced with lots of questions or groupings, they tend to be a better option.

There are two kinds of Likert charts—those that use a center line for a point of reference, and those that do not, in which case they are simply percentage bar charts for individual questions or are mosaic plots when comparing groupings. In the graphs below, each score value has its own color, and each score category—e.g., unfavorable is 1-2, neutral is 3, and favorable is 4-5 on a 5-point scale—is summarized by a percentage value at the left, middle/interior, and right sides of the bar, respectively.

The `likert` package provides some out-of-the-box plots for this type of data, though you must convert the numeric values to factors for it to work.

```
# Covert values to factors
ex_1[2:5] = lapply(ex_1[2:5], factor, levels = 1:5)

# Create a likert object
ex_1_likert = likert(ex_1[2:5])

# Figure 2
plot(ex_1_likert, ordered = FALSE, group.order = names(ex_1[2:5]))

# Figure 3
plot(ex_1_likert, ordered = FALSE, centered = FALSE, group.order=names(ex_1[2:5]))
```

Neither Likert chart type is as clear as the histogram at making the results immediately understandable, but again, histograms take more space, and busy decision makers often need to see the forest (all the questions)

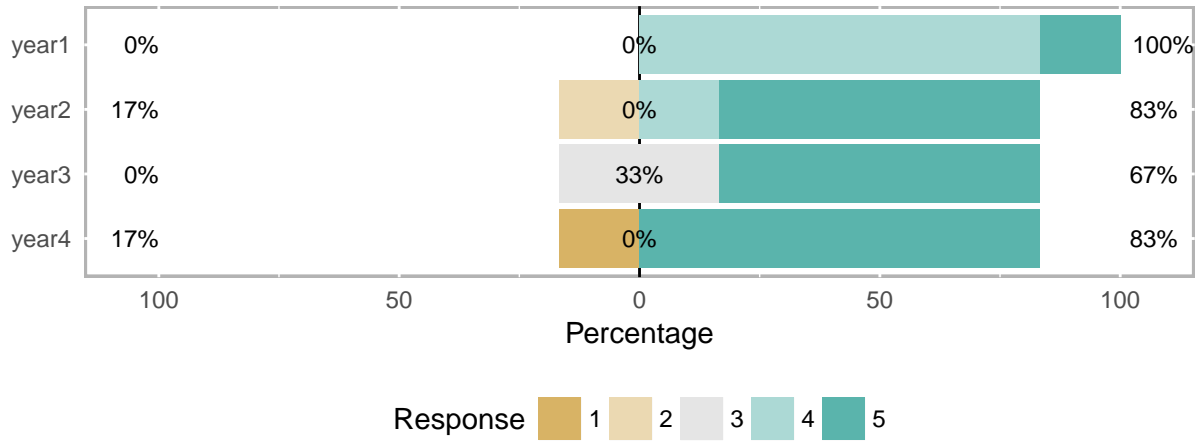


Figure 2: Centered Likert chart.

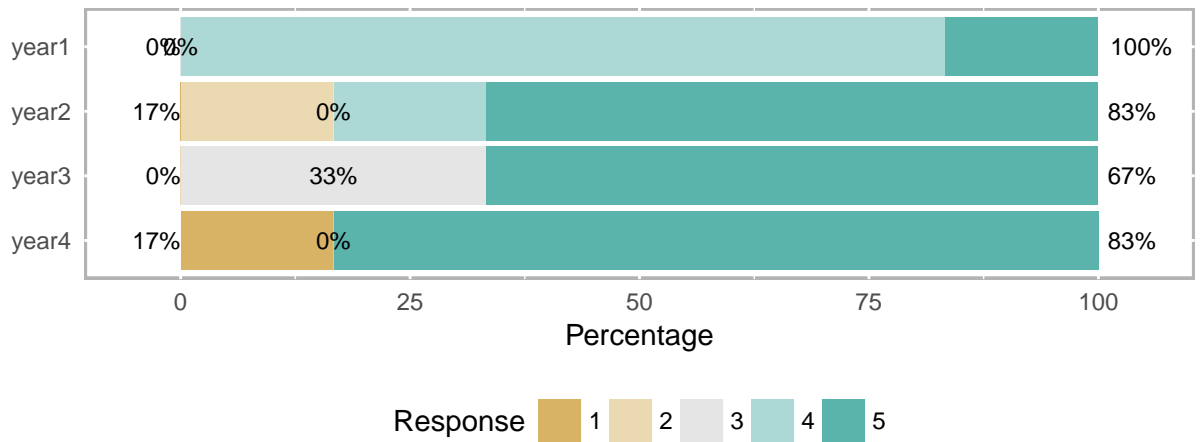


Figure 3: Uncentered Likert chart (aka percent bar chart).

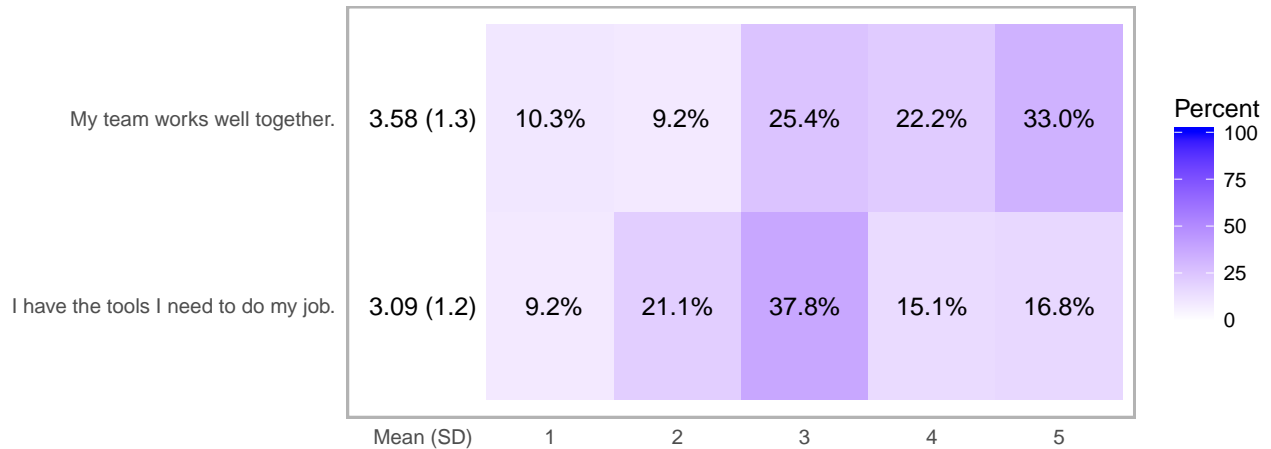


Figure 4: Heatmap of the response frequency for two different questions.

at the expense of some trees (each question). In this case, analysts might use the histograms to explore potentially important results themselves, and then use Likert charts in a report with some strategically-placed text highlighting important patterns they found with the histograms.

Other ordinal-scale visualizations

We'll use a larger data set in this section to illustrate other visualizations of ordinal data.

```
# Create likert object for example data set 2
ex_2_likert = likert(both[2:3])
```

Heatmap

Figure 4 shows a heatmap of the response frequency for two different questions (e.g., as within a single year's survey). While the use of means and SDs is inappropriate, this particular example directly illustrates why those values don't capture the response patterns in the data.

```
# Figure 4
plot(ex_2_likert, type = "heat")
```

Likert chart with response count histograms

The same data as seen in the heatmap above is more clearly shown with a Likert chart. Another option available with `likert` objects is to include count histogram to show number of responses and non-answers for each question.

```
# Figure 5
plot(ex_2_likert, include.histogram = TRUE)
```

Likert chart with subgroups

Subgroups can sometimes reveal patterns not seen in aggregate data. For example, compare the overall results for “My team works well together” in Figure 5 with the responses when broken out by the subgroups of MDs and RNs in Figure 6.

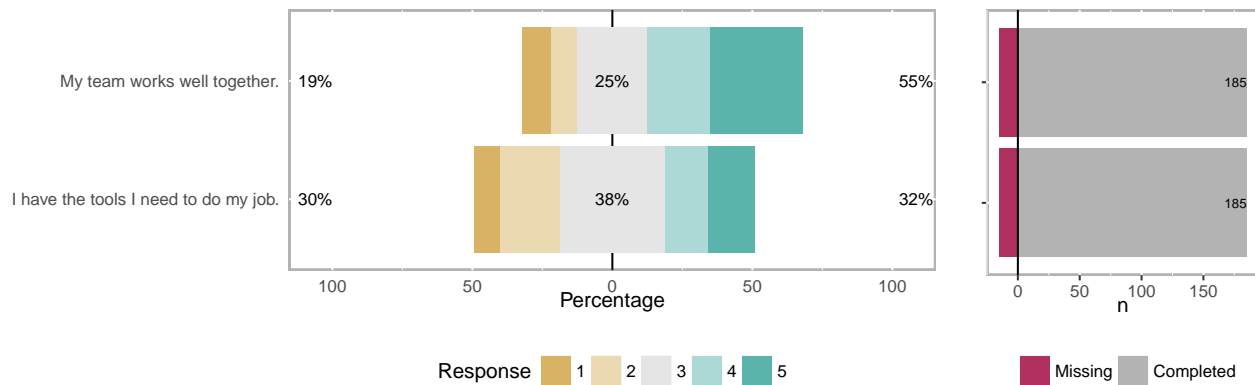


Figure 5: Likert chart of the response frequency for two different questions.

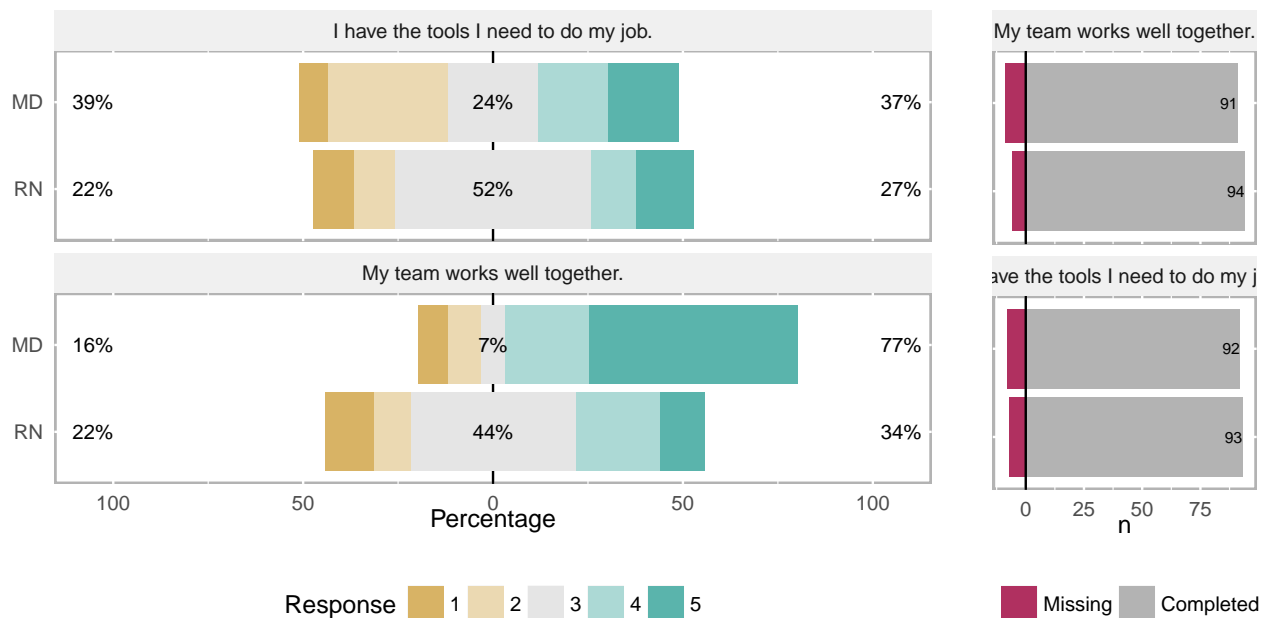


Figure 6: Likert chart of the response frequency for two different questions, grouped by MDs and RNs.

```
# Create likert object with groupings included
both_likert_2 = likert(both[, c(2:3), drop = FALSE], grouping = both$EmployeeType)

# Figure 6
plot(both_likert_2, include.histogram = TRUE)
```

Density histograms

While using a density plot on ordinal data is also statistically inappropriate, it can be a useful tool for an analyst. Bar histograms are difficult to overlay subgroups or different years for a direct comparison, so must be separated into facets instead, as was seen in Figure 1. Density plots are easier to overlay to show these comparisons, so while not appropriate for a report or dashboard, they can be really useful tools for an analyst during the exploration phase.

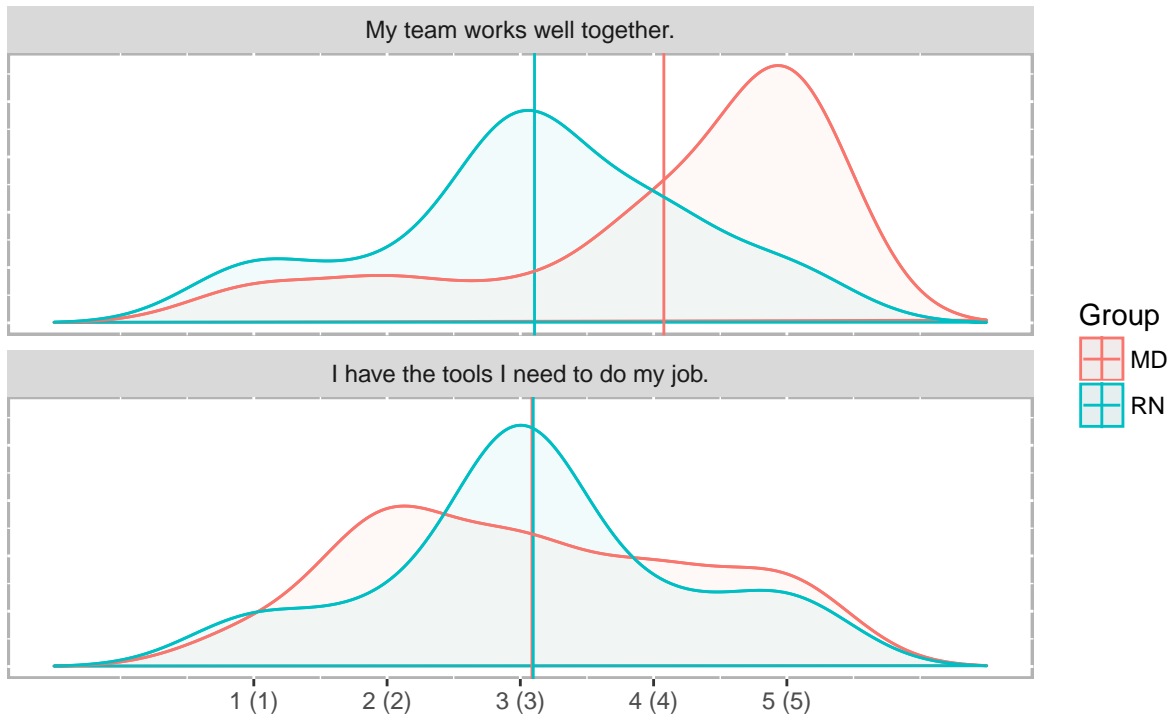


Figure 7: Density histograms of the response frequency for two different questions, with a grouping variable (MDs and RNs).

```
# Figure 7
plot(both_likert_2, type = "density")
```

Scatterplots & ordinal correlation

Although traditionally many analysts used non-parametric correlation like Spearman's or Kendall's, polychoric correlation is the proper tool to assess similarities between Likert scale results. (Polyserial correlation is used when one variable is numeric and the other is ordinal.)

```
poly_c_both = polychor(both[,2], both[,3])
```

The polychoric correlation coefficient between “My team works well together” and “I have the tools I need to do my job” is 0.0579. As expected, that suggests that there is no relationship between the responses to these two questions. It's fairly easy to see this lack of relationship in a scatterplot, with the points jittered to give a sense of response density.

```
# Figure 8
ggplot(both, aes(both[,2], both[,3], group=EmployeeType, color=EmployeeType)) +
  geom_jitter(na.rm=TRUE, width = 0.15, height = 0.15, alpha=0.6, size=3) +
  xlab("My team works well together") +
  ylab("I have the tools I need to do my job") +
  coord_equal()
```

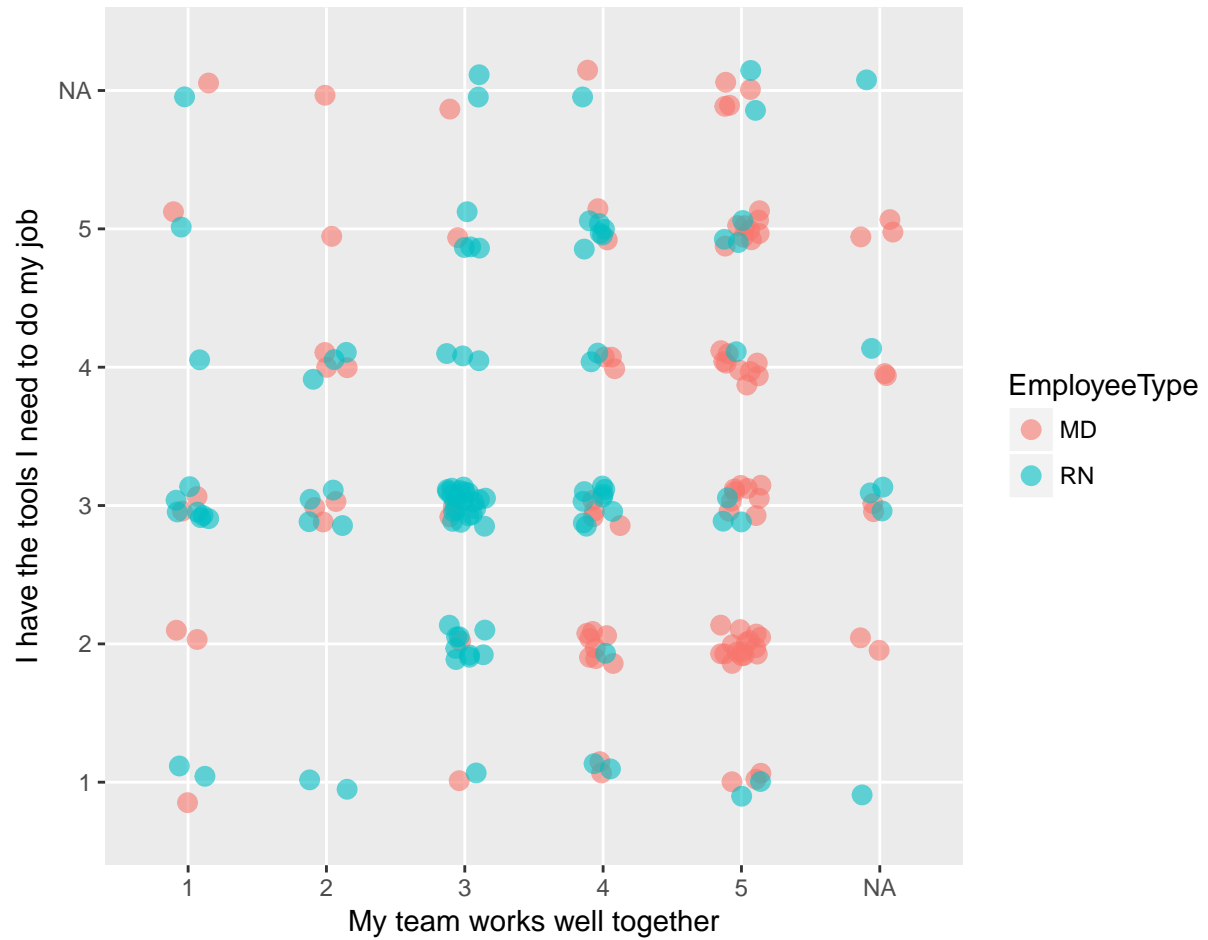


Figure 8: Scatterplot of ordinal comparisons (jittered to show point density) between the questions 'My team works well together' and 'I have the tools I need to do my job'.

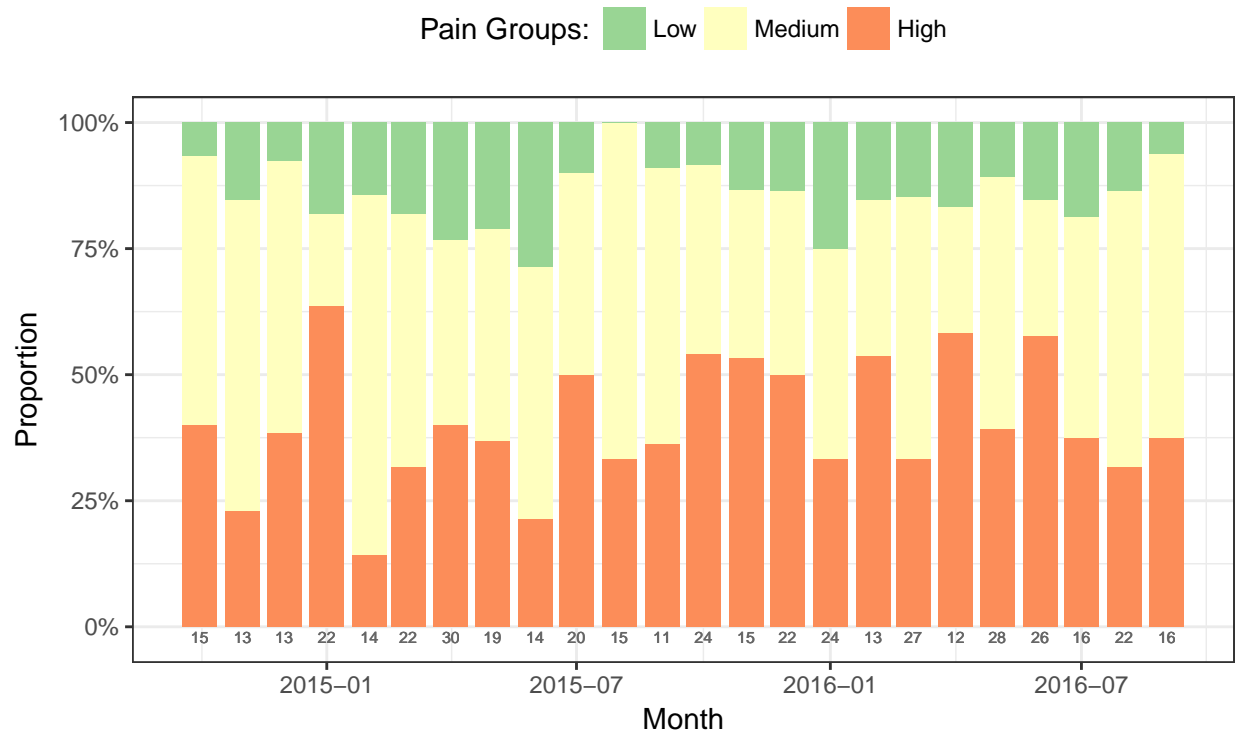


Figure 9: Monitoring maximum pain scores with a stacked percentage bar chart. Total surgeries performed that month occur just below each bar.

Monitoring ordinal data

In some cases, you may want to monitor data that uses an ordinal scale. If your metric is a “top box” type of score, a simple line chart can show that data over time; if it’s monitoring a stable process, statistical process control methods can be used as well.

If you want to monitor a more complete view of the data, a stacked percentage bar chart shows you trends across the time series.

Pain scores are a common outcome measure in surgeries, which are usually recorded on an intensity scale of 1-10, with 10 being the highest pain imaginable. Many researchers and quality improvement analysts collapse those values into Low (1-3), Medium (4-7), and High (8-10) categories, as the meaning of the exact values varies from patient to patient as well as between clinicians.

In this example (Figure 9), the maximum pain score for each patient in the 24 hours following a surgery are recorded, and assigned to a pain category. The total number of patients with maximum pain scores in each pain category are summed each month.

Figure 9

```
ggplot(surgeries_pain, aes(x = Month, y = Count, fill = Pain_Group)) +
  geom_bar(position = "fill", stat = "identity") +
  scale_y_continuous(labels = scales::percent) +
  scale_fill_brewer(name = "Pain Groups:", type = "div", palette = "Spectral", direction = -1) +
  geom_text(aes(y = -0.02, label = Surgeries), size = 2, color = "gray40") +
  ylab("Proportion") +
  theme_bw() +
  theme(legend.position = "top")
```

Neutral scores matter

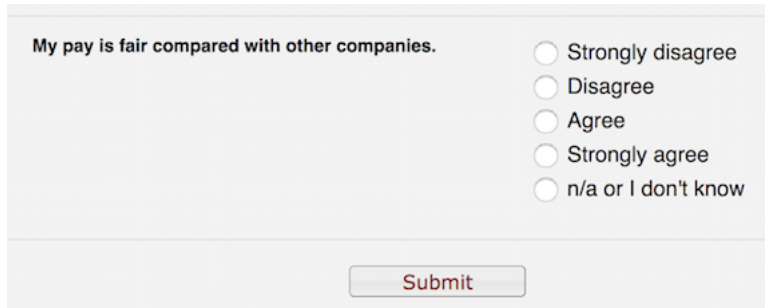
You might have noticed in some surveys that there is often no “neutral” or “undecided” category included in the middle of the scale, e.g., what’s usually a 3 on a 5-point Likert scale. Sometimes it is placed at the end of the scale, and sometimes it is eliminated entirely. The reason for this is that those terms can sometimes be interpreted in a variety of ways; for example, with a question such as “My pay is fair compared with other companies”, a *Neutral* response could indicate “I’m neutral on this”, “yes, I guess so”, “I don’t know”, “it’s neither fair nor unfair”, “I don’t want to answer”, “I’m not sure what ‘fair’ means”, and any number of ideas that don’t necessarily indicate a true neutral opinion.

When a question has a response option where this type of ambiguity exists, a mean value will tend toward the that option because of this bias, unless of course the mean is already at that value. However, when *Neutral* is marked as 3, and when valid responses tend towards 4s and 5s, these ambiguous responses will drag down the average (and vice versa for responses heavy with 1s and 2s). Of course, you shouldn’t use means anyway, as we’ve seen above, but many reports do—so understanding this effect is important toward interpreting the results in a useful way.

Use of a median is somewhat resistant to this problem, though you still won’t know whether the middle values are valid responses or accidents of interpretation.

When you see an “undecided” or “N/A” response placed at the end of the scale or missing entirely, it is usually (but not always!) a sign that the survey creator understands this problem.

Sometimes, of course, *Neutral* can be a completely reasonable and unambiguous response to a question. Context matters; while it’s easiest for survey creators and scanning software to use the same scale for large numbers of questions, it is important that the analyst understand the extent to which *Neutral* and similar types of responses are a valid part of the measurement scale for each question.



My pay is fair compared with other companies.

☐ Strongly disagree

☐ Disagree

☐ Agree

☐ Strongly agree

☐ n/a or I don't know

Submit

How many respondents are enough?

It’s common to think: “We surveyed everyone in this department, therefore the results we see must be correct.” However, how people responded to surveys depends on many factors—such as mood the date the survey is taken, recent events in life and in work, changes in organizational structure, and any number of other factors—and many internal surveys are given only once a year. Thus, survey results are really a *sample* of attitudes and opinions, subject to random events and natural fluctuations.

Typical practice at some companies is to expose summary results for groups with six or more people. While this helps preserve some anonymity, it does not include enough responses to ensure the overall response is stable. Comparisons over time or across groups that are not based on stable results can lead to conclusions about differences that may or may not reflect reality.

In this context, *stable* means that the data accurately represent true changes (or lack of change) in the question at hand. It’s basically impossible to distinguish natural variation from real change when you have

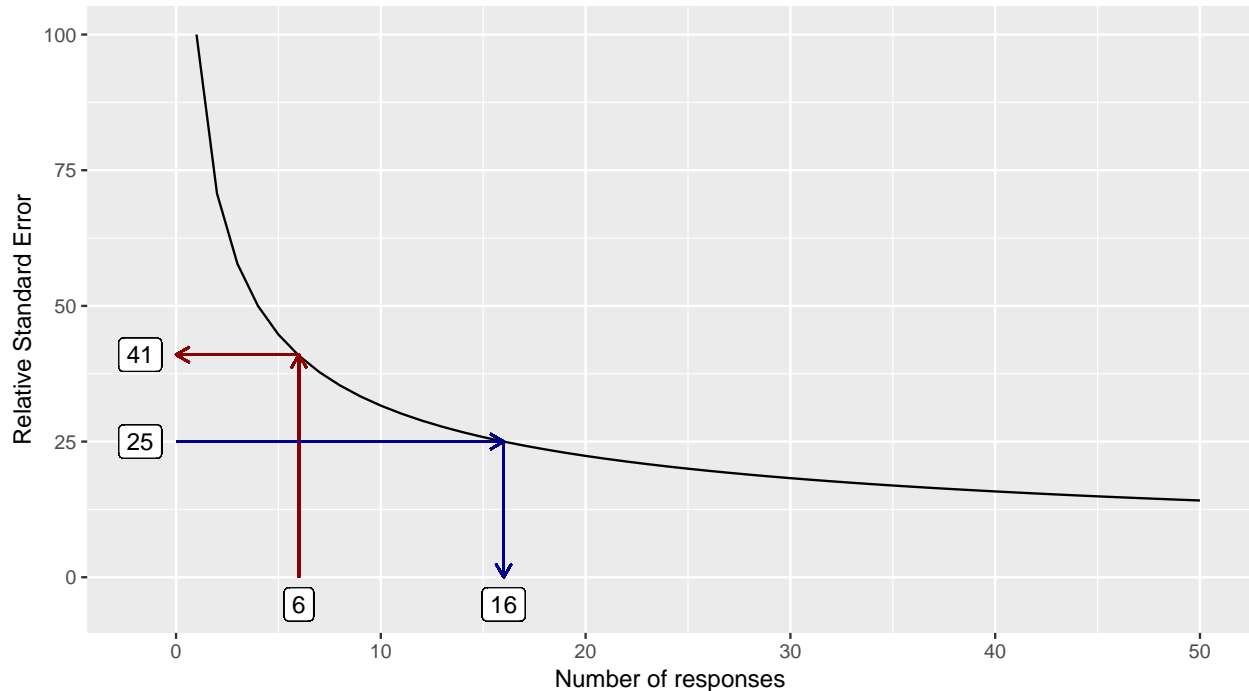


Figure 10: The relative standard error function.

small numbers of respondents. As a result, the National Center for Health Statistics, for example, does not publish results with less than 20 distinct cases or responses.

The relative standard error (RSE) is the metric used to evaluate whether you have enough values for the results to be stable. The standard error is an estimate of the likely difference between the results and the true value (which in surveys, even of complete populations, can't be known exactly due to the reasons mentioned above). The *relative* standard error is the standard error expressed as a percent of the measure or number of responses, which is a constant function: $\frac{1}{\sqrt{\text{responses}}} * 100$. This function can be seen in the graph on the next page.

Generally, you want RSE values less than 20-25% to have some confidence that your results are stable.

The RSE-response count function is shown in Figure 10. The RSE associated with the use of six responses is marked with dark red, and the response count associated with an RSE of 25% is marked with dark blue.

Is there a *significant* difference?

Many decision-makers want to know if a result is “significantly different” from, say, the same response from a previous time period, or between a couple of subgroups in the same response, typically asking for identification of questions in which $p < 0.05$ using a *t*-test. Unfortunately, this is mostly useless, for two reasons.

First, acting as if Likert or other ordinal scales are continuous level data leads to many problems of interpretation (see the *Appendix* for a summary table of measurement scales and appropriate statistics). There has been controversy over this distinction for many decades; however, a great way to understand the conceptual problem is to realize that the mean of *Agree* and *Strongly Agree* is **not** *Agree-And-A-Half*—it just makes no sense.

A subsequent argument might be that, no, it's not conceptually accurate, but it provides a sense for directional changes. However, such results still run into problems of interpretation: if you go from 4.17 to 4.33, have you

gone from *Agree.17* to *Agree.33*? What does such an “improvement” mean, in practical terms? All you can accurately say is that both values are most consistent with an *Agree* opinion.

Specifically in the medicine/healthcare context, Kuzon et al. state that the use of parametric statistics on ordinal data (such calculating a mean or using a *t*-test) is the first of “The seven deadly sins of statistical analysis”. Don’t “sin” and you don’t have to worry about whether your results are illegitimate.

There are a few ways around this problem: 1) use medians or other quantiles and test for differences in those statistics (these differences are best assessed via bootstrap or permutation testing), 2) test whether the distribution has shifted (Mann-Whitney-Wilcoxon or χ^2 tests), or 3) use more advanced techniques such as multinomial or proportional-odds regression (see the *Advanced* section, below). These options are the more statistically-correct ways to do it (as opposed a *t*-test).

However, even if you are using the correct tests, the multiple-testing problem remains if you are using traditional/frequentist inference. Make sure you consider the possibility of false-positives in any interpretation of mass-testing results, or use other inferential approaches such as Bayesian or Information-Theoretic instead (the *Advanced* section, below, uses an AIC-based Information-Theoretic approach for the model results compared against a no-difference model).

Permutation & Mann-Whitney tests

So, using the simple example from Chapter 1, we might want to know whether the median is statistically different between year 1 (Median = 4) and year 2 (Median = 5). Running a permutation test gives us the following results:

```
# Subset to years 1 and 2
ex_1_long_y12 = filter(ex_1_long, variable == "year1" | variable == "year2")

# Permutation test
oneway_test(value ~ variable, data = ex_1_long_y12, distribution = "exact")

>
> Exact Two-Sample Fisher-Pitman Permutation Test
>
> data: value by variable (year1, year2)
> Z = -0.33333, p-value = 1
> alternative hypothesis: true mu is not equal to 0
```

While our effect size is “1”—more accurately, *Agree* to *Strongly Agree*—the *p*-value of the test is very large (basically 1), so we cannot say that this difference is “statistically significant”.

We could also ask, “has the distribution shifted?”, which would involve using the Mann-Whitney-Wilcoxon test:

```
wilcox.test(value ~ variable, data = ex_1_long_y12)

>
> Wilcoxon rank sum test with continuity correction
>
> data: value by variable
> W = 11.5, p-value = 0.285
> alternative hypothesis: true location shift is not equal to 0
```

The *p*-value is non-significant, so the difference between year 1 and year 2 can’t be assumed to be a statistically significant change.

Looking at the raw data or graphs seen earlier, a decision-maker might be justified in wanting to act, but the analysis suggests that the difference is not statistically significant.

This leads us to the second problem with using p -values for determining whether a statistically-significant difference has occurred: sample size.

p -values are directly dependent on sample size. If your sample is large enough, you are guaranteed to have a small p -value. If your sample is small, whether or not you get a significant p -value depends on the scale of difference between the groups, i.e., the effect size.

For example, consider the following examples evaluating the number of people who answer *Agree* or *Strongly Agree* (the “favorable” score group) to a question:

Example	Favorable	Total Answers	Effect size	p -value
1	15	20	75%	0.04
2	114	200	57%	0.04
3	1,046	2,000	52%	0.04
4	1,001,450	2,000,000	50%	0.04

With 15 of 20 people selecting a favorable value on the Likert scale, we have an effect size of 75%, which is certainly an effect worth taking seriously. That value is also a statistically significant difference ($p < 0.05$), which supports the idea that the majority has a favorable opinion. With a couple of thousand responses (example 3), we again have a statistically significant difference, but the effect size is now only 52%, close enough to even-preference as to be *practically* the same. In medical terms, we might think of this as statistically significant but clinically irrelevant.

Effect sizes & CIs

For these reasons—and many others outside the scope of these guidelines—statisticians are moving away from the use of p -values. In frequentist statistics, these are being replaced by the use of effect sizes and confidence intervals (CIs); these provide information on both on the precision of the estimated difference, as well as whether the difference can be considered statistically distinct. If the CI includes 0, the difference is not-significant. Regardless of the location of 0, the width of the CI tells you how precise your estimate is.

```
median_diff = two.boot(year1, year2, median, R=1000)
cat(paste0("Difference in medians is ", abs(median_diff$t0), "."))
```

```
> Difference in medians is 1.
```

```
boot.ci(median_diff, type = "perc")
```

```
> BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
> Based on 1000 bootstrap replicates
>
> CALL :
> boot.ci(boot.out = median_diff, type = "perc")
>
> Intervals :
> Level      Percentile
> 95%      (-1,  1 )
> Calculations and Intervals on Original Scale
```

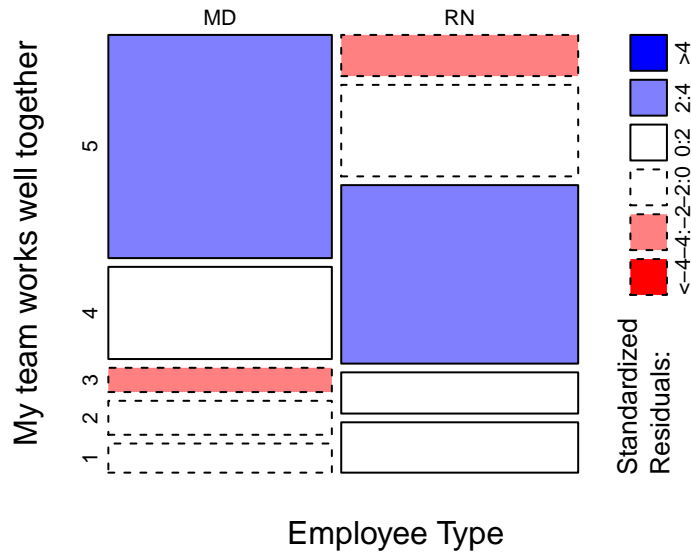


Figure 11: Mosaic plot between Employee Type and responses to the 'My team works well together' question.

Here, we see that the effect size is a difference in medians of 1, but the confidence interval on that effect size goes from -1 to +1, i.e. is consistent with any score difference between *Neutral* and *Strongly Agree*. Since that CI includes 0, we can't say that the change from median of *Agree* to a median of *Strongly Agree* is statistically different, though again, sample size matters—one would probably like to try to intervene based on the one respondent who dropped down to 2 (*Disagree*) anyway.

χ^2 test

While Mann-Whitney-Wilcoxon (sometimes known as the Mann-Whitney *U*-test) is the test most often used with differences between ordinal distributions, there are other options that can tell you whether a measured difference between groups is statistically different.

The old stand-by in this case is the χ^2 test, which is often best visualized with a mosaic plot (Figure 11).

```
# Figure 11
mosaicplot(both2_tab, shade = T, main="", xlab="Employee Type",
  ylab="My team works well together")
```

```
# Chi-square test
chisq.test(both2_tab)
```

```
>
> Pearson's Chi-squared test
>
> data: both2_tab
> X-squared = 52.809, df = 4, p-value = 9.344e-11
```

Multinomial regression

The multinomial regression model is a more powerful (and more modern) version of the χ^2 test. Here, we're using an AIC-based information-theoretic approach to determine whether the data as we see it is as likely

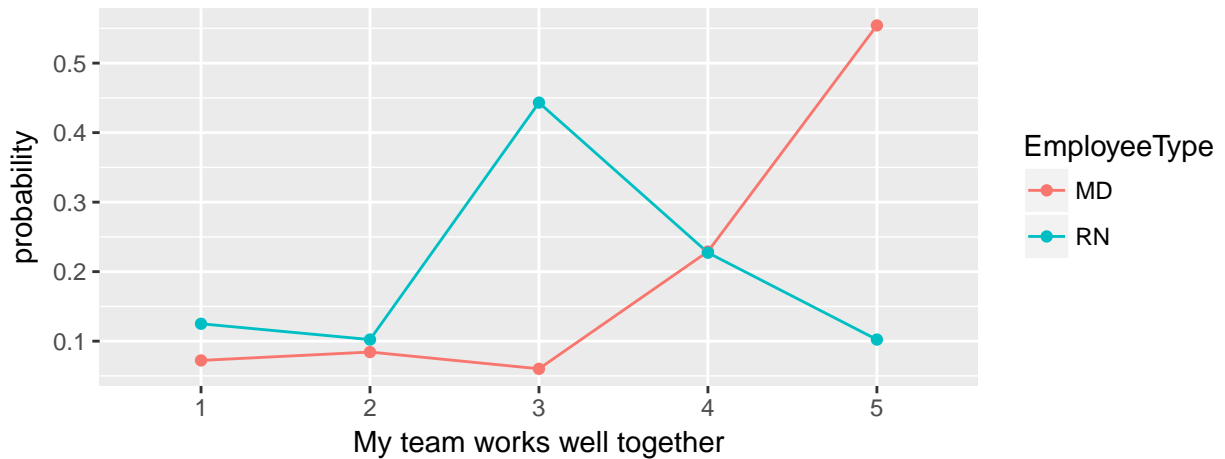


Figure 12: Multinomial regression between Employee Type and responses to the 'My team works well together' question, with information-theoretic table for multi-model inference.

a model as a model that suggests no difference between MD and RN responses. The probabilities can be plotted with a line chart for easy comparison.

```
# Multinomial regression
multnom_both = multinom(Teamwork ~ EmployeeType, data = both3, trace = FALSE)
multnom_both_1 = multinom(Teamwork ~ 1, data = both3, trace = FALSE)

# New data for prediction
df_both = data.frame(EmployeeType = rep(c("MD", "RN"), each = 5),
  Teamwork = rep(c(1:5), 2))

# Get probabilities
multnom_both_probs = cbind(df_both, predict(multnom_both,
  newdata = df_both, type = "probs", se = TRUE))

# Clean up, ugh
multnom_both_probs = multnom_both_probs[, -2]
multnom_both_probs = unique(multnom_both_probs)

# Make data frame for ggplot, probably should figure out tidyr
multnom_both_probs_df = reshape2::melt(multnom_both_probs,
  id.vars = "EmployeeType", variable.name = "Teamwork",
  value.name = "probability")

# Figure 12
ggplot(multnom_both_probs_df, aes(x = Teamwork, y = probability,
  color = EmployeeType, group = EmployeeType)) +
  geom_line() +
  geom_point() +
  xlab("My team works well together")
```

There appears to be an effect based on Employee Type; the AIC weight for that model is practically 1, making it clearly the best model of the model set.

```
# AICc table
mod_set = list()
```

```
mod_set[[1]] = multinom_both
mod_set[[2]] = multinom_both_1
```

```
kable(aictab(mod_set, modnames = c("Employee Type", "Null Model")))
```

Modnames	K	AICc	Delta_AICc	ModelLik	AICcWt	LL	Cum.Wt
Employee Type	8	472.0249	0.00000	1	1	-227.5680	1
Null Model	4	522.0647	50.03976	0	0	-256.9118	1

Proportional-odds regression

If you can meet the assumptions, the proportional-odds regression is more powerful than the multinomial model, as it can take into account the ordered nature of the ordinal scale.

Again, we use the information-theoretic approach to determine whether there is an effect based on Employee Type, again plotted with a line chart for easy comparison. And again the AICc results suggest an effect is present. The probabilities are slightly different from the multinomial model, but may be more accurate since we are now accounting for the ordered nature of the response values.

```
# Proportional odds regression with polr
polr_both = polr(Teamwork ~ EmployeeType, data = Teamwork_tab_long,
  weight = Count)

# New data for prediction, same as multinom
df_both = data.frame(EmployeeType = rep(c("MD", "RN"), each = 5),
  Teamwork = rep(c(1:5), 2))

# Get probabilities
polr_both_probs = cbind(df_both, predict(polr_both, newdata = df_both,
  type = "probs", se = TRUE))

# Clean up, ugh
polr_both_probs = polr_both_probs[,-2]
polr_both_probs = unique(polr_both_probs)

# Make data frame for ggplot, probably should figure out tidyr
polr_both_probs_df = reshape2::melt(polr_both_probs, id.vars = "EmployeeType",
  variable.name = "Teamwork", value.name = "probability")

# Figure 13
ggplot(polr_both_probs_df, aes(x = Teamwork, y = probability,
  color = EmployeeType, group = EmployeeType)) +
  geom_line() +
  geom_point() +
  xlab("My team works well together")

# I need to better understand diffs between polr and clm
# because polr objects don't play well with the AICcmodavg package
# Coefs/thresholds are exactly the same, though
fm1 = clm(Teamwork ~ EmployeeType, data=tab_df)
```

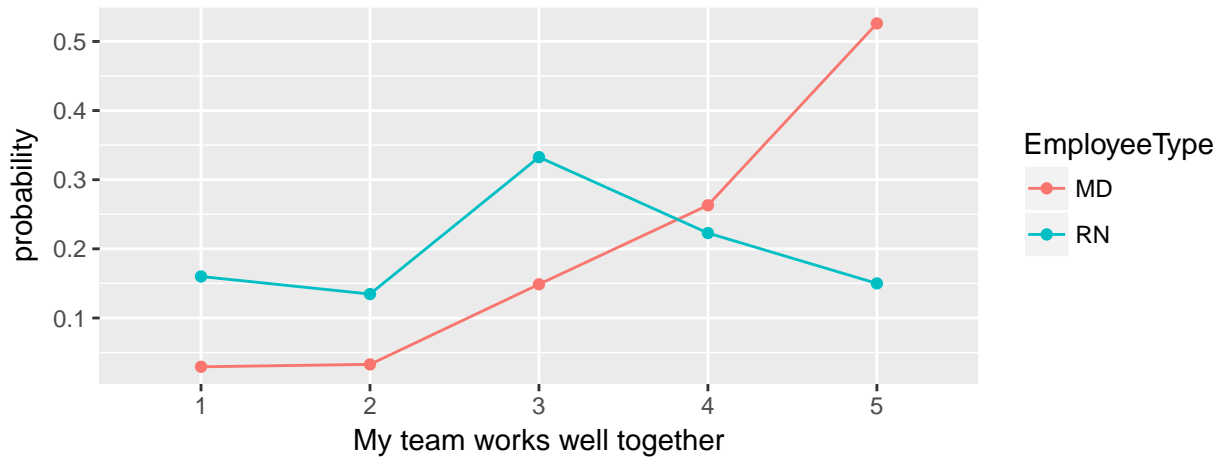


Figure 13: Proportional odds regression between Employee Type and responses to the 'My team works well together' question, with information-theoretic table for multi-model inference.

```
# Null model
fm2 = clm(Teamwork ~ 1, data=tab_df)

# AICc table
mod_set = list()
  mod_set[[1]] = fm1
  mod_set[[2]] = fm2

kable(aictab(mod_set, modnames = c("Employee Type", "Null Model")))
```

Modnames	K	AICc	Delta_AICc	ModelLik	AICcWt	LL	Cum.Wt
Employee Type	5	485.9008	0.00000	1	1	-237.7686	1
Null Model	4	522.0647	36.16387	0	0	-256.9118	1

We can test the assumption of proportional odds with the `anova` function. There's no evidence of a difference, suggesting that the assumption is met.

```
fm3 = clm(Teamwork ~ EmployeeType, data=tab_df, threshold="equidistant")
anova(fm1, fm3)
```

```
> Likelihood ratio tests of cumulative link models:
>
>      formula:                link: threshold:
> fm3 Teamwork ~ EmployeeType logit equidistant
> fm1 Teamwork ~ EmployeeType logit flexible
>
>      no.par    AIC  logLik LR.stat df Pr(>Chisq)
> fm3       3 485.94 -239.97
> fm1       5 485.54 -237.77  4.407  2    0.1104
```

If the concepts or ideas in this section are confusing, it's probably worth consulting a statistician for help evaluating your data with these tools.

A final word

“But everybody does it!” Yeah, and they’re wrong.

“But it’s directionally correct!” When a concept is based on discrete categories, “in-between” states are simply incoherent. The light’s either on or off, practically speaking.

“But I want to!” No. Just. No.

Appendix: Measurement Levels & Summary Statistics

Statistic / Parameter	Categorical <i>Nominal</i>	Ranked <i>Ordinal</i>	Discrete/Counts <i>Interval/Ratio</i>	Continuous <i>Interval/Ratio</i>
Data set size (n)	Y	Y	Y	Y
Percent / Frequency	Y	Y	Y	Y
Count or rate	Y	Y	Y	Y
Categories (levels)	Y	Y	Y	Y
Mode	Y	Y	Y	Y
Median	<i>No</i>	Y	Y	Y
Interquartile range	<i>No</i>	Y	Y	Y
Median absolute deviation	<i>No</i>	Y	Y	Y
Range	<i>No</i>	Y	Y	Y
Minimum/maximum value	<i>No</i>	Y	Y	Y
Quantiles	<i>No</i>	Y	Y	Y
Mean (average)	<i>No</i>	<i>No</i>	Y	Y
Standard deviation	<i>No</i>	<i>No</i>	Y*	Y*
Coefficient of variation	<i>No</i>	<i>No</i>	Y*	Y*

* You must use the correct distribution (proper mean-variance relationship) to ensure you get the correct standard deviation; most software defaults to calculating the standard deviation for a normally-distributed sample, which could be incorrect for certain kinds of count, rate, or proportion data, for example.