Statistical Report on Pain Management

Amanda Waldron

Fri Mar 11 2022

Contents

You are a research scientist at a hospital conducting medical research on pain management. Your colleague, a biologist, has concocted a new medication that has potential for decreasing the amount of pain patients in the burn victims unit feel when they go through their daily wound cleaning. Together, you conducted a randomized controlled trial where you randomly assigned 300 burn victims (with their consent) to one of three groups: medicine group, control group, and placebo group. The medicine group received the recommended dose of the new pain medication, the placebo group received a fake lookalike of the medication that was actually just flavorless sugar, and the control group received nothing. The patients in all 3 groups then provided their self-reported pain intensity at the end of their wound cleaning session on a 10 point Likert scale ranging from 1 (not very painful) to 10 (extremely painful). Your job is to determine whether this new medicine shows efficacy at reducing the pain caused by daily, necessary cleaning of burn wounds, over and above the effect of placebo.

1 Prepare R Session For Analysis (Section Total: 2pt)

1.0.1 (1pt.) Load the following packages into the library. Install packages that you don't already have in the console.

- tidyr
- dplyr
- DescTools
- jtools
- ggplot2

```
library(tidyr)
library(dplyr)
library(DescTools)
library(jtools)
library(ggplot2)
tinytex::install_tinytex()
```

1.0.2 (1pt) We have a csv file called "HW4.data" (available on Blackboard). Read this file into R Studio.

```
Ravenclaw <- read.csv("HW4.data.csv")
```

2 Inspect Data (Section Total: 13pt)

2.0.1 (3pt) To run an analysis using the aov or lm command, you will need to provide the argument: DV ~ IV. View the first few rows of our dataset. Can we do that? Why or why not? If not, what format does our data need to be in order to use the aov command?

```
Ravenclaw %>% head()
```

```
##   control.pain medicine.pain placebo.pain
## 1            9             6           7
## 2            6             6           9
```

```
## 3            9            5            7
## 4            5            5            8
## 5            8            4            6
## 6            9            8            8
```

The data is organized in such a way that the dependent and independent variable levels are combined into a 3 separate lists. It is organized by the separate levels as separate categories and the pain scores are divided into separate groups. It is not organized in a DV ~ IV fashion. Instead it is "control.pain" instead of pain ratings being separated from each level of the independent variable. Therefore we can not run an anlysis as of yet. The data needs to be reorganized into separate concise independent variables: control, medicine and placebo and then the pain ratings.Essentially, it would be ideal and simpler to organize the different conditions into one category(in other words each level) such as med.condition and the dependent variable(pain.scores) should have their own separate category corresponding to the conditions.That would then allow us to conduct an aov & lm command.

2.0.2 (2pt) Convert the data from wide to long format. When you do this, name the key "med.condition" and the value "pain.score". After, show the first few rows of the data.

```
Ravenclaw1 <- Ravenclaw

Ravenclaw1 %>% tidyr::gather(key = "med.condition", value = "pain.score") %>% head
```

```
##   med.condition pain.score
## 1  control.pain          9
## 2  control.pain          6
## 3  control.pain          9
## 4  control.pain          5
## 5  control.pain          8
## 6  control.pain          9
```

```
Ravenclaw1 <- Ravenclaw1 %>% gather(key = "med.condition", value = "pain.score")
Ravenclaw1 %>% head
```

```
##   med.condition pain.score
## 1  control.pain          9
## 2  control.pain          6
## 3  control.pain          9
## 4  control.pain          5
## 5  control.pain          8
## 6  control.pain          9
```

2.0.3 (2pt) Relabel the values in the med.condition variable to get rid of ".pain" at the end of each label. For example, control.pain should simply read "control".

```
Ravenclaw1 <- Ravenclaw1 %>%
mutate(med.condition = ifelse(Ravenclaw1$med.condition == "control.pain","control",
                              ifelse(Ravenclaw1$med.condition ==
                                     "placebo.pain","placebo","medicine")))


Ravenclaw1 %>% head()
```

```
##   med.condition pain.score
## 1       control          9
## 2       control          6
## 3       control          9
## 4       control          5
## 5       control          8
## 6       control          9
```

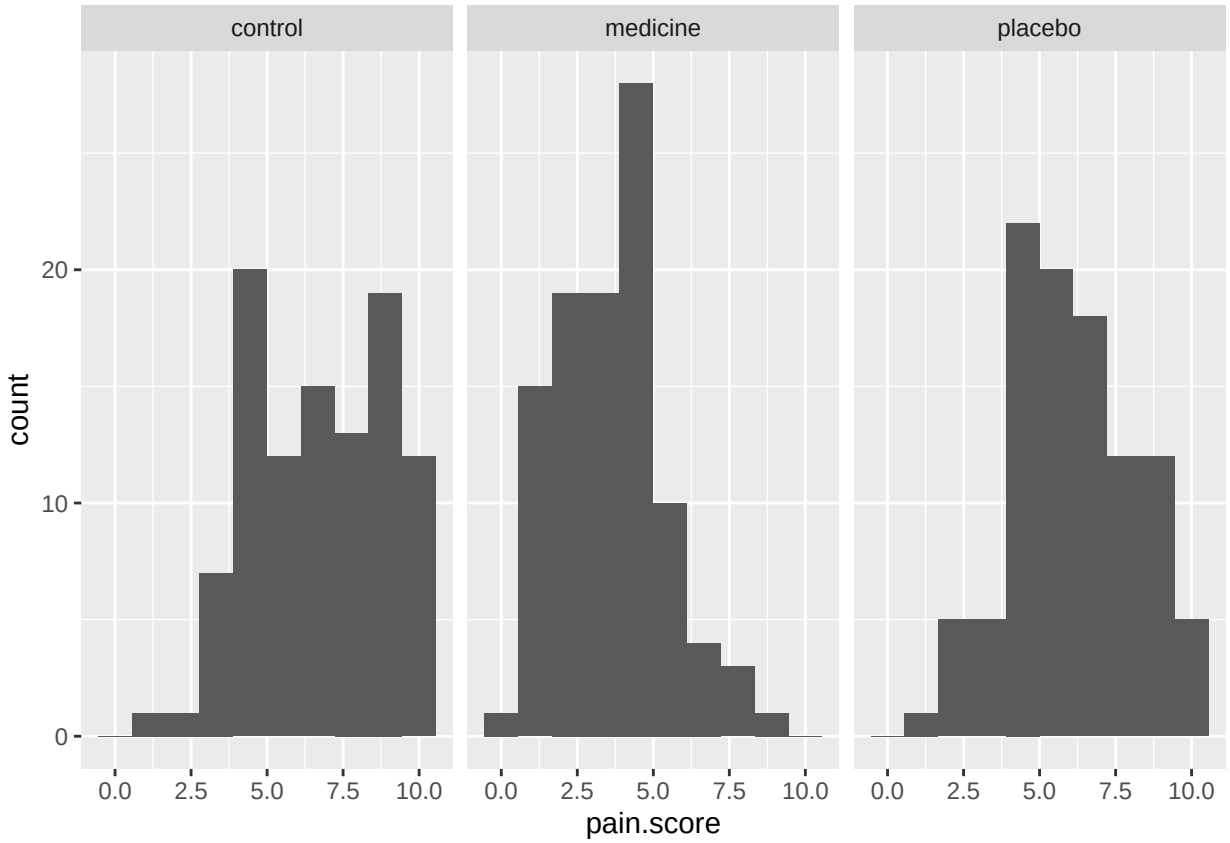
2.0.4 (2pt) Show the means and standard deviations of each med.condition group.

```
Ravenclaw1.desc <-
  Ravenclaw1 %>%
  group_by(med.condition) %>%
  summarise(means = mean(pain.score), SEs = sd(pain.score) / sqrt(30))
Ravenclaw1.desc
```

```
## # A tibble: 3 x 3
##   med.condition means   SEs
##   <chr>         <dbl> <dbl>
## 1 control        6.87 0.412
## 2 medicine       3.58 0.363
## 3 placebo        6.28 0.386
```

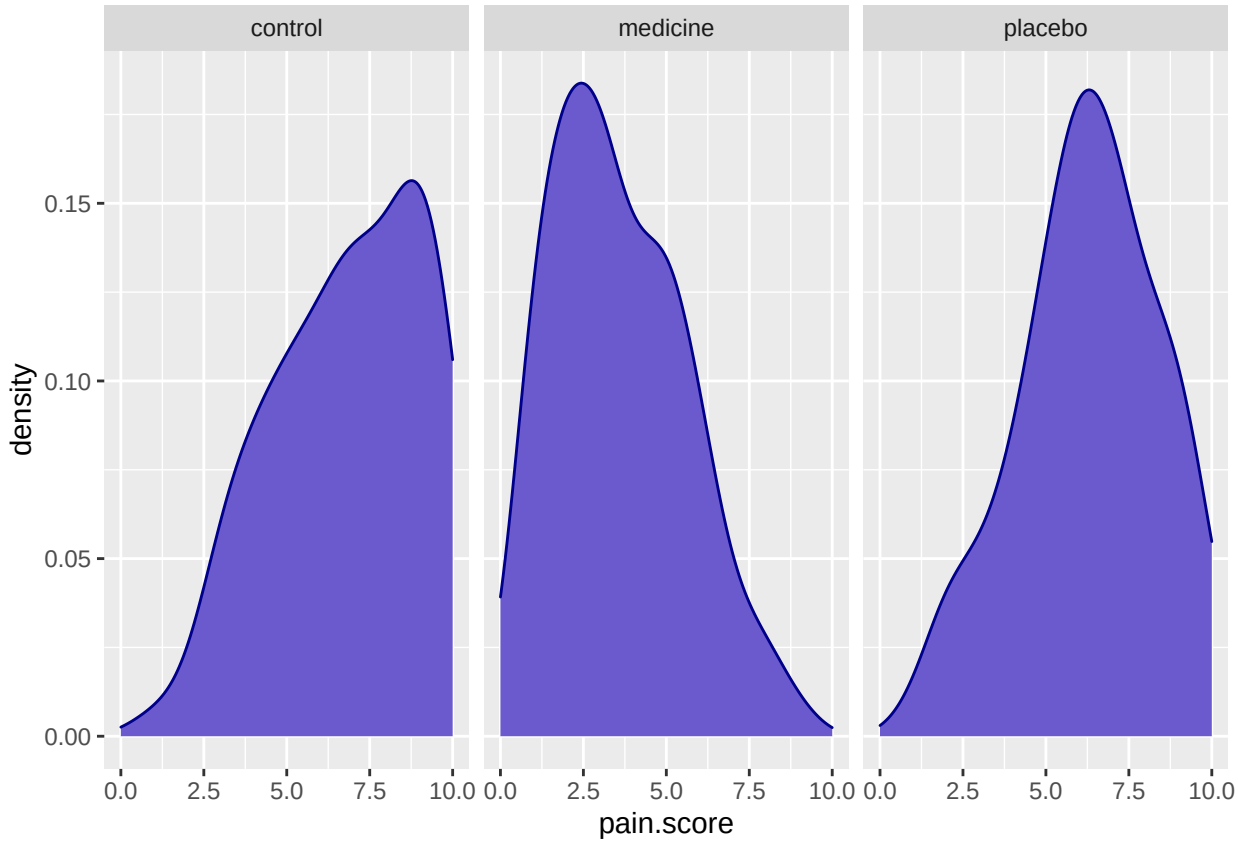
2.0.5 (2pt) Use the ggplot2 package to create histograms of pain.score for each med.condition group. Set the number of bins to 10 (use R documentation, ?geom_histogram, to figure out how).

```
ggplot(Ravenclaw1, aes(x = pain.score)) +
  geom_histogram(bins = 10) +
  facet_wrap(facets = "med.condition", nrow = 1)
```



2.0.6 (2pt) Use the ggplot2 to create density plots (i.e. histograms, but smooth) of pain.score for each med.condition group.

```
ggplot(Ravenclaw1, aes(x = pain.score)) +
  geom_density(color = "darkblue", fill= "slateblue") +
  facet_wrap(facets = "med.condition", nrow = 1)
```



3 Statistical Analyses (Section Total: 33pt)

3.1 Linear Regression (Section Subtotal: 9pt)

3.1.1 (2pt) Conduct a simple linear regression, and name it "my.reg". In the line below below that, pipe my.reg to the summ(confint = TRUE, digits = 4) command to get neat, summary output.

```
my.reg <- lm(pain.score ~ med.condition, data = Ravenclaw1)

my.reg %>% summ(confint = TRUE,digits = 4)
```

```
## MODEL INFO:
## Observations: 300
## Dependent Variable: pain.score
## Type: OLS linear regression
##
## MODEL FIT:
## F(2,297) = 68.2086, p = 0.0000
## R² = 0.3147
## Adj. R² = 0.3101
##
```

```
## Standard errors: OLS
## --------------------------------------------------------------------------
##                               Est.      2.5%     97.5%     t val.        p
## -------------------------- --------- --------- --------- ---------- --------
## (Intercept)                  6.8700    6.4520    7.2880    32.3452   0.0000
## med.conditionmedicine       -3.2900   -3.8811   -2.6989   -10.9530   0.0000
## med.conditionplacebo        -0.5900   -1.1811    0.0011    -1.9642   0.0504
## --------------------------------------------------------------------------
```

3.1.2 (1pt) What does the estimate of the intercept represent?

The estimate (6.87) of the intercept represents the control level of the med.condition.

3.1.3 (1pt) What does the estimate of med.conditionmedicine represent?

The estimate of the med.conditionmedicine (-3.29) represents the slope of that level of med.conditon.Also the slope of the line tells us the rate of change of y relative to x.

3.1.4 (1pt) What does the estimate of med.conditionplacebo represent?

The estimate of the med.conditionplacebo (-0.59) represents the slopes of that level of med.condition.Also the slope of the line tells us the rate of change of y relative to x.

3.1.5 (1pt) Using just the estimates from the regression output and the linear model formula, show some math that calculates what the expected value of pain.score is before taking into account the effect of medication and placebo. Hint: use dummy coding for the x-values.

```
6.87 + (-3.29 * 0) + (-0.59 * 0)
```

```
## [1] 6.87
```

3.1.6 (1pt) Repeat the last question, except now calculate the expected value of pain.score after taking into account the effect of medication.

```
6.87 + (-3.29 * 1) + (-0.59 * 0)
```

```
## [1] 3.58
```

3.1.7 (2pt) Repeat the last question, except now calculate the expected value of pain.score after taking into account the effect of BOTH medication and placebo.

```
6.87 + (-3.29 * 1) + (-0.59 * 1)
```

```
## [1] 2.99
```

3.2 Analysis of Variance (Section Subtotal: 24pt)

3.2.1 (1pt) Conduct an ANOVA using the aov() command and store it under the name "my.anova". In the line below, pipe my.anova to the summary command to show the ANOVA output.

```
my.anova <- aov(pain.score ~ med.condition, Ravenclaw1)
my.anova %>% summary
```

```
##                Df Sum Sq Mean Sq F value Pr(>F)
## med.condition   2  615.4  307.70   68.21 <2e-16 ***
## Residuals     297 1339.8    4.51
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3.2.2 (3pt) Give a precise definition of the p-value from the F Test provided in your ANOVA output.

The p value from the F test in the Anova output indicates the probability of observing a relationship between med.condition and pain.score this large or larger just due to sampling variability if the null were true.If the p-value is very low (< alpha level), you reject the null hypothesis and conclude that there's a statistically significant difference.

3.2.3 (1pt) Recall that you can get the residuals (the difference between the expected values predicted by the regression and the actual values) by calling resid(my.reg). Use that command to get the residuals from my.reg. Square the residuals, and take their sum. What does this number correspond to in the ANOVA output?

```
resid(my.reg) %>% head
```

```
##     1     2     3     4     5     6
##  2.13 -0.87  2.13 -1.87  1.13  2.13
```

```
my.reg$residuals^2 %>% sum
```

```
## [1] 1339.83
```

This number(1339.83) corresponds to the Sum of Squares residual.

3.2.4 (1pt) Divide the Sum Sq of the Residuals by the Residuals Df (degrees of freedom). What does this number correspond to in the ANOVA output?

```
(my.reg$residuals^2 %>% sum) / my.reg$df.residual
```

```
## [1] 4.511212
```

This number(4.51) corresponds to the Mean Square Residual in the ANOVA output.

3.2.5 (1pt) Divide the Sums Sq of med.condition by its Df. What does this number correspond to in the ANOVA output?

```
615.4/2
```

```
## [1] 307.7
```

This number(307.7) corresponds to the Mean Square for the med.condition in the ANOVA output.

3.2.6 (1pt) Divide the Mean Squares of med.condition by Mean Sq of the Residuals. What does this number correspond to in the ANOVA output (with rounding error)?

```
307.70 / 4.51
```

```
## [1] 68.22616
```

This returns back our F value(68.22616 with rounding error) in the ANOVA output which makes sense because this was most definitely the formula for an F test.

3.2.7 (2pt) Divide the Sum Sq of med.condition by the total Sum Sq (i.e. Sum Sq of med.condition + Sum Sq Residuals). What does this number correspond to in the regression output? What does it mean conceptually?

```
615.4 + 1339.8
```

```
## [1] 1955.2
```

```
615.4 / 1955.2
```

```
## [1] 0.3147504
```

This number corresponds to R^2 in our regression output.Conceptually it means that SSmed.condition/SStotal will give us our effect size. This actually makes sense because this is the exact formula we us for R^2 in an ANOVA analysis.It comes full circle that the same R^2 that is calculated in our regression analysis is the exact same as the R^2 that computed here from our ANOVA. This demonstrates that both of these tests are equal to each other and are essentially different methods of analysis(testing under the null so testing the same thing) that will retrun the same values and results just in different places in the outputs.

3.2.8 (1pt) If the Mean Sq of the Residuals from the ANOVA output represents the *pooled* residual variance (i.e. the average variance across our three groups), then you can take the square root of this number to get the pooled standard deviation. Do so below.

```
sqrt(4.51)
```

```
## [1] 2.123676
```

3.2.9 (3pt) Recall that the formula for Cohen's D is (mean difference) divided by (pooled standard deviation). You just calculated the pooled standard deviation in the last question. Using the estimates from your regression output, calculate Cohen's D for each group comparison. Note that for the placebo vs medicine comparison, you will have to first calculate the group mean for the placebo group and medicine group using the regression estimates, then subtract medicine mean from placebo mean.

```
#my.reg %>% summ(confint= TRUE)
# medicine vs control group
-3.29/2.123676
```

```
## [1] -1.549201
```

```
# placebo vs control group
-0.59/2.123676
```

```
## [1] -0.2778202
```

```
# placebo vs medicine group
#Control(intercept) - medicine
6.87 - 3.29
```

```
## [1] 3.58
```

```
#Control(intercept) - placebo
6.87 - 0.59
```

```
## [1] 6.28
```

```
#Medicine - Placebo
3.58 - 6.28
```

```
## [1] -2.7
```

```
-2.7/2.123676
```

```
## [1] -1.27138
```

3.2.10 (1pt) In the regression output, we have the answers to most questions we would ask about our data: is there a statistically significant difference between the control group and medicine group, even after accounting placebo? Is there a statistically significant effect of placebo over and above the medicine group? Given the meaning of the estimates, we already have the answer to those questions. But by convention, psychologists often prefer ANOVAs to test for a statistically significant difference among any of the group means, then use a post-hoc test to look for differences among the groups, such as Tukey Test. The advantage of this approach is the the Tukey test controls for alpha inflation, or the increased probability of rejecting a null hypothesis when we should not have (Type I error) due to the shear number of tests we conducted. The p-values are "alpha-adjusted". Of course, you could just adjust the p-values from the regression, but that's another story! For now, let's conduct the Tukey Test. Pipe my.anova to the TukeyHSD() command from the DescTools package.

```
my.anova %>% TukeyHSD
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = pain.score ~ med.condition, data = Ravenclaw1)
##
## $med.condition
##                     diff       lwr        upr     p adj
## medicine-control -3.29 -3.997537 -2.5824628 0.0000000
## placebo-control  -0.59 -1.297537  0.1175372 0.1229613
## placebo-medicine  2.70  1.992463  3.4075372 0.0000000
```

3.2.11 (3pt) According to the Tukey Test, what is the (alpha-adjusted) probability of observing a difference between the means of the placebo and control groups of -0.59 or larger just due to sampling variability alone if the null hypothesis of no difference were true?

p = 0.122,which is not significant meaning that the probability of observing a difference between the means of the placebo and control groups of -0.59 or larger just due to sampling variability alone if the null hypothesis of no difference were true is p= 0.122, we fail to reject the null hypothesis of no difference.

3.2.12 (3pt) Using an alpha of 0.05, what do we conclude was the cause of the difference between the means of placebo and control?

The cause of the difference between the means of placebo and control can be deemed to be that the placebo might have had an effect due to the participants being under the impression that they are in fact taking medication thus they could be more likely to have a lower pain score. BUT when we do compare our p value of 0.122 against the alpha of 0.05, there is no significance of this finding so we can conclude that this most likely occurred by chance and of course due to the placebo effect.

3.2.13 (3pt) Explain to your colleague what the results of the analysis were. Was the medication effective? If so, was the effect of the medication clinically significant (i.e. large enough to be noticeable in the real world)? Was there evidence for a placebo effect, such that even taking fake lookalike medication reduces pain? Consider the both the significance tests (aka p-values and confidence intervals) and effect sizes when communicating your conclusions.

The results indicate that there was a effect of medication on pain scores.Also, the effect of medication was found to be significant as shown by our analysis. When referring to our post-hoc analysis, there was a significant effect found in the difference of means in the medicine-placebo and medicine-control groups, p > .01/p > .05.The corresponding confidence intervals for both of these groups actually do capture the true population values which only support our findings of the effects of medication and thus more applicable to the real world. There was evidence for a placebo effect found in the placebo-control group in our post-hoc analysis because of the difference in means but that finding was not significant as shown by the corresponding p value compared to the alpha.The corresponding confidence intervals for this placebo-control group included 0 which shows that it is less likely to capture the true population value; plus it did not capture the true population value by just looking at the outputs. Therefore, the fake lookalike medication appears to reduce pain but those findings were not significant thus it could have happened by chance and just due to the belief of the participants that they were taking a medication that would help with their pain causing their pain scores to be different. The effect size was not as large as we would like to see at 31% because this means that a whopping 69% of pain.score is not explained by med.condition. That effect size needs to be looked into more as we need to understand what else can contribute to the pain.scores in future research.Overall, we found significant support for the effect of the medication relieving pain.

4 Making Publication-Ready Plots (12pts)

4.0.1 Make a violin plot with boxplot overlays visualizing the differences in pain.scale among the three groups. For this plot, do the following:

- (1pt) put med.condition on the x-axis, pain.score on the y-axis, and med.condition as the fill.
- (1pt) suppress the legend for both violin and boxplots.
- (1pt) adjust the violin plots to be see-through so that the boxplots clearly stand out against them.
- (1pt) adjust the width of the boxplots to look nice on top of the violin plots.
- (1pt) use aes(middle = mean(pain.score)) to make the black line in the box plot represent the mean instead of the median.
- (1pt) choose new colors for the fill using scale_fill_manual. Make sure that there is enough difference in lightness among your colors so that you could clearly distinguish between groups if the plot were printed in grey-scale (which often happens). You can view a list of ggplot's color options here: http://sape.inf.usi.ch/quick-reference/ggplot2/colour
- (1pt) add the caption "Figure 1. Violin plot with box-and-whisker overlays. The middle value on the box-and-whisker plots are the group means."
- (1pt) relabel the y-axis "Pain Score".
- (1pt) relabel the x-axis "Group Assignment".
- (1pt) add a theme of your choosing.
- (2pt) do this last: using the theme() command (which is different from choosing a theme), move the plot caption all the way to the left. Also make all text in the plot the Times font family.

Note that the order of when you add elements to the plot using "+" matters: ggplot move chronologically. So if you want the boxplot on top of the violin plot (rather than behind it), add the boxplot after you have the violin plot.

```r
# Note: add "\n" right before "means", or press "enter" in the typed caption below
# right before means. Either of these will add a line break to the caption right
# before the word "means".

library("extrafont")
```

Registering fonts with R

```r
#font_import()
#loadfonts

my.violin <- ggplot(Ravenclaw1, aes(x=med.condition, y=pain.score, fill=med.condition,
                                    aes(middle = mean(pain.score))))+
geom_violin(alpha=0.5)+
geom_boxplot(width=0.7)+
labs(title= "Med.Condition & Pain Score Violin Plot", caption= "Figure 1. Violin plot with box-and-whisk
     x= "Group Assignment", y="Pain Score")+
theme(legend.position = "none",
      plot.caption = element_text(hjust= 0))
theme_classic()
```

```
## List of 93
##  $ line                        :List of 6
##   ..$ colour       : chr "black"
##   ..$ size         : num 0.5
##   ..$ linetype     : num 1
##   ..$ lineend      : chr "butt"
##   ..$ arrow        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_line" "element"
##  $ rect                        :List of 5
##   ..$ fill         : chr "white"
##   ..$ colour       : chr "black"
##   ..$ size         : num 0.5
##   ..$ linetype     : num 1
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_rect" "element"
##  $ text                        :List of 11
##   ..$ family       : chr ""
##   ..$ face         : chr "plain"
##   ..$ colour       : chr "black"
##   ..$ size         : num 11
##   ..$ hjust        : num 0.5
##   ..$ vjust        : num 0.5
##   ..$ angle        : num 0
##   ..$ lineheight   : num 0.9
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ title                       : NULL
```

```
##  $ aspect.ratio              : NULL
##  $ axis.title                : NULL
##  $ axis.title.x              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2.75points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.top          :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2.75points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.bottom       : NULL
##  $ axis.title.y              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : num 90
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.75points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.y.left         : NULL
##  $ axis.title.y.right        :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : num -90
```

```
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.75points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text                :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : chr "grey30"
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2.2points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.top          :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2.2points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.bottom       : NULL
##  $ axis.text.y              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
```

```
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.2points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.y.left          : NULL
##  $ axis.text.y.right         :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.2points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.ticks                :List of 6
##   ..$ colour       : chr "grey20"
##   ..$ size         : NULL
##   ..$ linetype     : NULL
##   ..$ lineend      : NULL
##   ..$ arrow        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_line" "element"
##  $ axis.ticks.x              : NULL
##  $ axis.ticks.x.top          : NULL
##  $ axis.ticks.x.bottom       : NULL
##  $ axis.ticks.y              : NULL
##  $ axis.ticks.y.left         : NULL
##  $ axis.ticks.y.right        : NULL
##  $ axis.ticks.length         : 'simpleUnit' num 2.75points
##   ..- attr(*, "unit")= int 8
##  $ axis.ticks.length.x       : NULL
##  $ axis.ticks.length.x.top   : NULL
##  $ axis.ticks.length.x.bottom: NULL
##  $ axis.ticks.length.y       : NULL
##  $ axis.ticks.length.y.left  : NULL
##  $ axis.ticks.length.y.right : NULL
##  $ axis.line                 :List of 6
##   ..$ colour       : chr "black"
##   ..$ size         : 'rel' num 1
##   ..$ linetype     : NULL
##   ..$ lineend      : NULL
##   ..$ arrow        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_line" "element"
##  $ axis.line.x               : NULL
```

```
##  $ axis.line.x.top           : NULL
##  $ axis.line.x.bottom        : NULL
##  $ axis.line.y               : NULL
##  $ axis.line.y.left          : NULL
##  $ axis.line.y.right         : NULL
##  $ legend.background         :List of 5
##   ..$ fill         : NULL
##   ..$ colour       : logi NA
##   ..$ size         : NULL
##   ..$ linetype     : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_rect" "element"
##  $ legend.margin             : 'margin' num [1:4] 5.5points 5.5points 5.5points 5.5points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing            : 'simpleUnit' num 11points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing.x          : NULL
##  $ legend.spacing.y          : NULL
##  $ legend.key                : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.key.size           : 'simpleUnit' num 1.2lines
##   ..- attr(*, "unit")= int 3
##  $ legend.key.height         : NULL
##  $ legend.key.width          : NULL
##  $ legend.text               :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.text.align         : NULL
##  $ legend.title              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.title.align        : NULL
##  $ legend.position           : chr "right"
##  $ legend.direction          : NULL
```

```
##  $ legend.justification      : chr "center"
##  $ legend.box                : NULL
##  $ legend.box.just           : NULL
##  $ legend.box.margin         : 'margin' num [1:4] 0cm 0cm 0cm 0cm
##   ..- attr(*, "unit")= int 1
##  $ legend.box.background     : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.box.spacing        : 'simpleUnit' num 11points
##   ..- attr(*, "unit")= int 8
##  $ panel.background          :List of 5
##   ..$ fill         : chr "white"
##   ..$ colour       : logi NA
##   ..$ size         : NULL
##   ..$ linetype     : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_rect" "element"
##  $ panel.border              : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ panel.spacing             : 'simpleUnit' num 5.5points
##   ..- attr(*, "unit")= int 8
##  $ panel.spacing.x           : NULL
##  $ panel.spacing.y           : NULL
##  $ panel.grid                :List of 6
##   ..$ colour       : chr "grey92"
##   ..$ size         : NULL
##   ..$ linetype     : NULL
##   ..$ lineend      : NULL
##   ..$ arrow        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_line" "element"
##  $ panel.grid.major          : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ panel.grid.minor          : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ panel.grid.major.x        : NULL
##  $ panel.grid.major.y        : NULL
##  $ panel.grid.minor.x        : NULL
##  $ panel.grid.minor.y        : NULL
##  $ panel.ontop               : logi FALSE
##  $ plot.background           :List of 5
##   ..$ fill         : NULL
##   ..$ colour       : chr "white"
##   ..$ size         : NULL
##   ..$ linetype     : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_rect" "element"
##  $ plot.title                :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 1.2
##   ..$ hjust        : num 0
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
```

```
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 5.5points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ plot.title.position       : chr "panel"
##  $ plot.subtitle             :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 5.5points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ plot.caption              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : num 1
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 5.5points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ plot.caption.position     : chr "panel"
##  $ plot.tag                  :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 1.2
##   ..$ hjust        : num 0.5
##   ..$ vjust        : num 0.5
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ plot.tag.position         : chr "topleft"
##  $ plot.margin               : 'margin' num [1:4] 5.5points 5.5points 5.5points 5.5points
##   ..- attr(*, "unit")= int 8
##  $ strip.background          :List of 5
##   ..$ fill         : chr "white"
```

```
##   ..$ colour       : chr "black"
##   ..$ size         : 'rel' num 2
##   ..$ linetype     : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_rect" "element"
##  $ strip.background.x        : NULL
##  $ strip.background.y        : NULL
##  $ strip.placement           : chr "inside"
##  $ strip.text                :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : chr "grey10"
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 4.4points 4.4points 4.4points 4.4points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ strip.text.x              : NULL
##  $ strip.text.y              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : num -90
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ strip.switch.pad.grid     : 'simpleUnit' num 2.75points
##   ..- attr(*, "unit")= int 8
##  $ strip.switch.pad.wrap     : 'simpleUnit' num 2.75points
##   ..- attr(*, "unit")= int 8
##  $ strip.text.y.left         :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : num 90
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
```

```
##  - attr(*, "complete")= logi TRUE
##  - attr(*, "validate")= logi TRUE
```

```
my.violin + scale_fill_brewer(palette="Dark2")
```

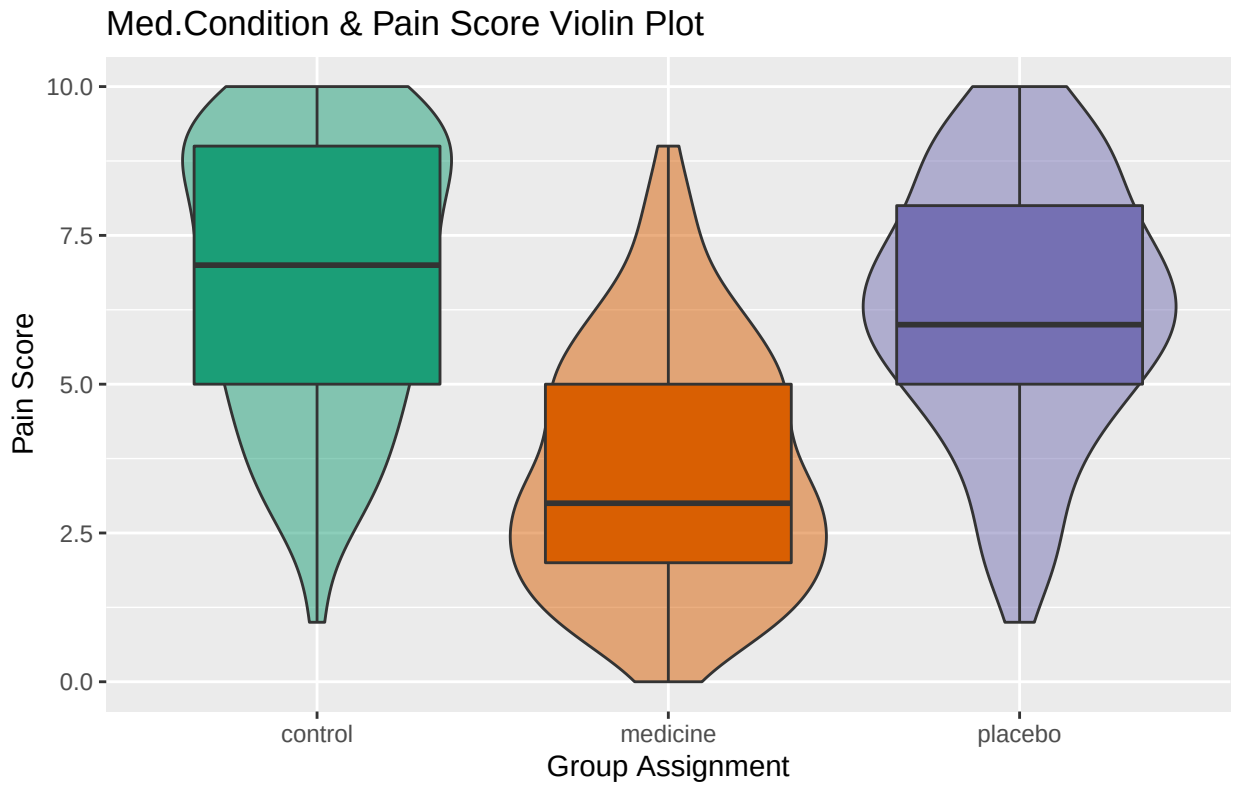


Figure 1. Violin plot with box−and−whisker overlays. The middle value on the box−and−whisker plots are the means.