# Supplemental Material for 'Optimizing Model Performance and Fairness Through Evolved Sample Weights'

Anil Kumar Saini, Jose Guadalupe Hernandez, Emily F. Wong, Jason H. Moore

2024-08-01

# Contents

# Chapter 1

# Introduction

This is not intended as a stand-alone document, but as a companion to our manuscript.

## 1.1 Contributing authors

- Anil Kumar Saini
- Jose Guadalupe Hernandez
- Emily F. Wong
- Jason H. Moore

## 1.2 About our supplemental material

As you may have noticed (unless you're reading a pdf version of this), our supplemental material is hosted using GitHub pages. We compiled our data analyses and supplemental documentation into this nifty web-accessible book using bookdown.

The code used for this supplemental material can be found in this GitHub repository.

Our supplemental material includes the following:

- Metric defintions (Section **??**)

- Heart disease results (Section 3)
- Student math results (Section 4)
- Student por results (Section 5)

- CreditG results (Section 6)
- Titanic results (Section 7)
- US Crime results (Section 8)
- Compas Violent results (Section 9)
- NLSY results (Section 10)
- Compas results (Section 11)
- Speed dating results (Section 12)
- PMAD EPDS results (Section 13)
- PMAD PHQ results (Section 14)

## 1.3 Supplemental material setup

### 1.3.1 Required packages and variables

Variable set up.

```
library(ggplot2)
library(cowplot)
library(dplyr)
library(PupillometryR)

NAMES <- c('Evolved','Calculated','None')
TASKS <- c('heart_disease', 'student_math', 'student_por', 'creditg', 'titanic', 'us_c
SHAPE <- c(21,24,22)
cb_palette <- c('#D81B60','#1E88E5','#FFC107')
TSIZE <- 19

p_theme <- theme(
  plot.title = element_text( face = "bold", size = 22, hjust=0.5),
  panel.border = element_blank(),
  panel.grid.minor = element_blank(),
  legend.title=element_text(size=18),
  legend.text=element_text(size=18),
  axis.title = element_text(size=18),
  axis.text = element_text(size=14),
  legend.position="bottom",
  panel.background = element_rect(fill = "#f1f2f5",
                                  colour = "white",
                                  linewidth = 0.5, linetype = "solid")
)

testing <- read.csv(paste('./', 'hv_test.csv', sep = "", collapse = NULL), header = TRU
testing$exp <- gsub('Evolved Weights', 'Evolved', testing$ex)
testing$exp <- gsub('Calculated Weights', 'Calculated', testing$ex)
```

```
testing$exp <- gsub('No Weights', 'None', testing$ex)
testing$exp <- factor(testing$exp, levels = NAMES)
```

### 1.3.2 Helper functions

Function to plot hypervolume results

```
  # function to plot hyper-volume data
  volume_plotter <- function(data, id)
  {
    ggplot(data, aes(x = exp, y = hv, color = exp, fill = exp, shape = exp)) +
    geom_flat_violin(position = position_nudge(x = .1, y = 0), scale = 'width', alpha = 0.2, widt
    geom_boxplot(color = 'black', width = .07, outlier.shape = NA, alpha = 0.0, size = 1.0, posit
    geom_point(position = position_jitter(width = 0.02, height = 0.0001), size = 1.5, alpha = 1.0
    scale_y_continuous(
      name="Volume",
    ) +
    scale_x_discrete(
      name="Strategy"
    )+
    scale_shape_manual(values=SHAPE, name="Weight\nStrategy") +
    scale_colour_manual(values = cb_palette, name="Weight\nStrategy") +
    scale_fill_manual(values = cb_palette, name="Weight\nStrategy") +
    ggtitle(TASKS[id])+
    p_theme + coord_flip()
  }
```

Function to summarize hypervolume results

```
# function to plot hyper-volume data
volume_summarize <- function(data)
  {
    data %>%
    group_by(exp) %>%
    dplyr::summarise(
      count = n(),
      na_cnt = sum(is.na(hv)),
      min = min(hv, na.rm = TRUE),
      median = median(hv, na.rm = TRUE),
      mean = mean(hv, na.rm = TRUE),
      max = max(hv, na.rm = TRUE),
      IQR = IQR(hv, na.rm = TRUE)
    )
  }
```

# Chapter 2

# Bias defintions

Multiple metrics exist to measure the fairness of predictions made by a machine learning model. Each metric is defined in relation to a specific application context and attempts to quantify different properties (false negative rate, accuracy, etc.) of the predictions for people belonging to different groups. Different metrics try to quantify different properties (false negative rate, accuracy, etc.) of the predictions for people belonging to different groups. For example, 'Demographic parity' measures whether the acceptance rates (proportion of individuals belonging to the group receiving positive prediction) are the same for all groups. 'Error rate parity' measures whether the false positive and false negative rates in all groups are equal, and 'Predictive parity' ensures an equal positive prediction rate across all groups. Here, we discuss in detail two commonly used metrics to measure the fairness in the predictions of a given model : 'Subgroup False Positive Fairness', and 'Subgroup False Negative Fairness'. Before delving into the definitions of the above-mentioned metrics, we describe some terminologies here. Let $\mathcal{D} = \{(X, X', Y)_i\}_{i=1}^N$ be the dataset under consideration. For each data point $(X, X', Y)$, $X \in \mathcal{X}^d$ contains values corresponding to $d$ *non-sensitive* features, $X' \in \mathcal{X}'^p$ contains values corresponding to $p$ *sensitive* features, and $Y$ contains the target variables. Features deemed 'sensitive', or 'protected', such as race, sex, and gender, are classified as sensitive features ($X'$). Here, we would assume $X'$ and $X$ do not overlap, and therefore, $X + X'$ would give us the full feature set for a particular data point. Based on the values of sensitive attributes, each data point can fall into one of the groups defined by those sensitive attributes. For example, 'Black women younger than 25' would be one of the groups when the sensitive attributes are race, gender, and age. Let $G \in \mathcal{G}$ be one such group. We show the membership to this group by $X' \in G$. Finally, let $\hat{Y} \in \{0, 1\}$ be the predicted target value output by the classifier. Finally, let $R(X, X') \in [0.0, 1.0]$ be the risk score output by a given ML model, $\hat{Y} \in \{0, 1\}$ is the predicted target value, and for simplicity, also the classifier, formed by applying a threshold on $R(X, X')$. False Positive Subgroup Fairness and False Positive Subgroup Fairness capture the maximum deviation of a model's performance among any one group in $\mathcal{G}$, normalized by the probability of observing an individual from that group in the negative or positive labels, respectively. Since in most scenarios, we would want the model to perform similarly in all groups, lower values on these metrics denote more fair models. For a dataset $\mathcal{D}$, and risk model $R(X, X')$, the following are the definitions. False Positive (FP) Rate : False positive (FP) rate can be defined as

$$FP(R) = Pr[\hat{Y} = 1 | Y = 0].$$

And the False Positive Rate for a group $G$ can be defined as

$$FP(R, G) = Pr[\hat{Y} = 1 | Y = 0, X' \in G].$$

False Negative (FN) Rate : False positive (FP) rate can be defined as

$$FN(R) = Pr[\hat{Y} = 0 | Y = 1].$$

And the False Negative Rate for a group $G$ can be defined as

$$FN(R, G) = Pr[\hat{Y} = 1 | Y = 0, X' \in G].$$

False Positive Subgroup Fairness : Let the probability of getting negative labels in group G be

$$\alpha_{FP}(G) = Pr[X' \in G, Y = 0].$$

We also define the absolute difference in false positive rate between the whole population and for a specific group G as

$$\beta(R, G) = |FP(R) - FP(R, G)|.$$

Then the False Positive Subgroup Fairness (FPSF) is given by

$$FPSF(D, R) = \max_{G \in \mathcal{G}} \alpha_{FP}(G) \beta(R, G).$$

False Negative Subgroup Fairness : Let the probability of getting positive labels in group G be

$$\alpha_{FN}(G) = Pr[X' \in G, Y = 1].$$

We also define the absolute difference in false negative rate between the whole population and for a specific group G as

$$\beta(R, G) = |FN(R) - FN(R, G)|.$$

Then the False Positive Subgroup Fairness (FPSF) is given by

$$FNSF(D, R) = \max_{G \in \mathcal{G}} \alpha_{FN}(G) \beta(R, G).$$

# Chapter 3

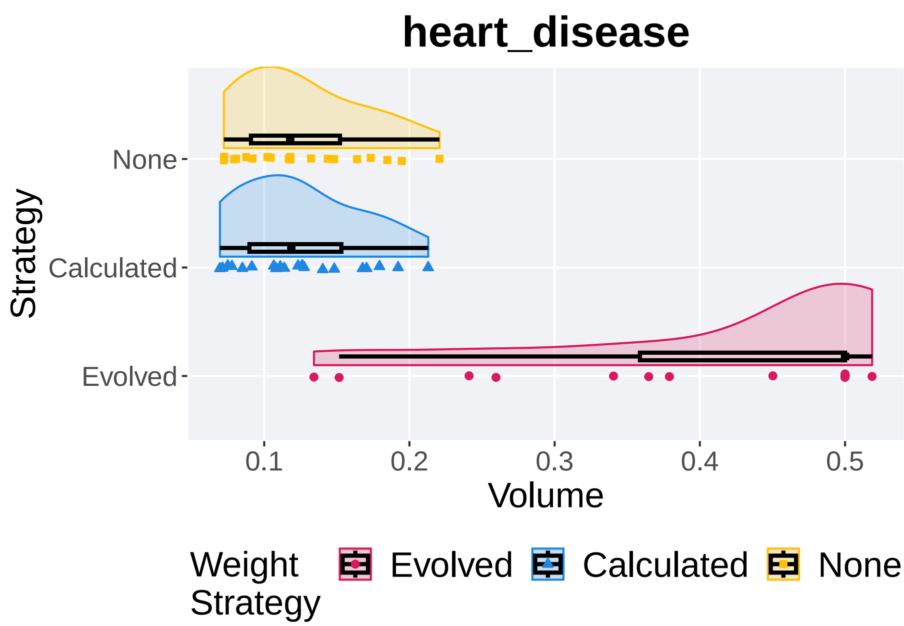
# Heart Disease

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `heart_disease` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "heart_disease")
```

## 3.1 Hypervolume

```
volume_plotter(data,1)
```




### 3.1.1 Summary stats

```r
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt    min median  mean   max    IQR
##   <fct>      <int>  <int>  <dbl>  <dbl> <dbl> <dbl>  <dbl>
## 1 Evolved       20      0 0.134   0.5   0.417 0.519 0.141
## 2 Calculated    20      0 0.0695  0.119 0.125 0.213 0.0633
## 3 None          20      0 0.0722  0.118 0.126 0.221 0.0613
```

### 3.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```r
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 34.987, df = 2, p-value = 2.528e-08
```

### 3.1.3 Pairwise wlcoxon test

```
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 4.5e-07 -
## None       4.5e-07 1
##
## P value adjustment method: bonferroni
```

# Chapter 4

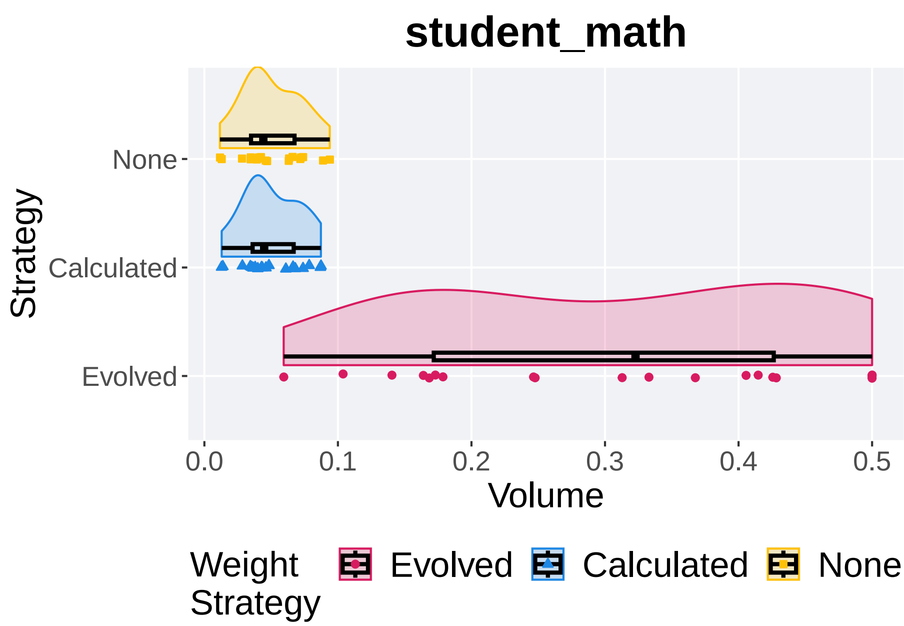
# Student Math

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `student_math` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "student_math")
```

## 4.1 Hypervolume

```
volume_plotter(data,2)
```




### 4.1.1 Summary stats

```
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt    min median   mean    max    IQR
##   <fct>      <int>  <int>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 Evolved       20      0 0.0594 0.323  0.308  0.5    0.255
## 2 Calculated    20      0 0.0129 0.0448 0.0504 0.0873 0.0307
## 3 None          20      0 0.0116 0.0441 0.0503 0.0939 0.0326
```

### 4.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 36.282, df = 2, p-value = 1.323e-08
```

### 4.1.3 Pairwise wlcoxon test

```
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 3.3e-07 -
## None       3.3e-07 1
##
## P value adjustment method: bonferroni
```

# Chapter 5

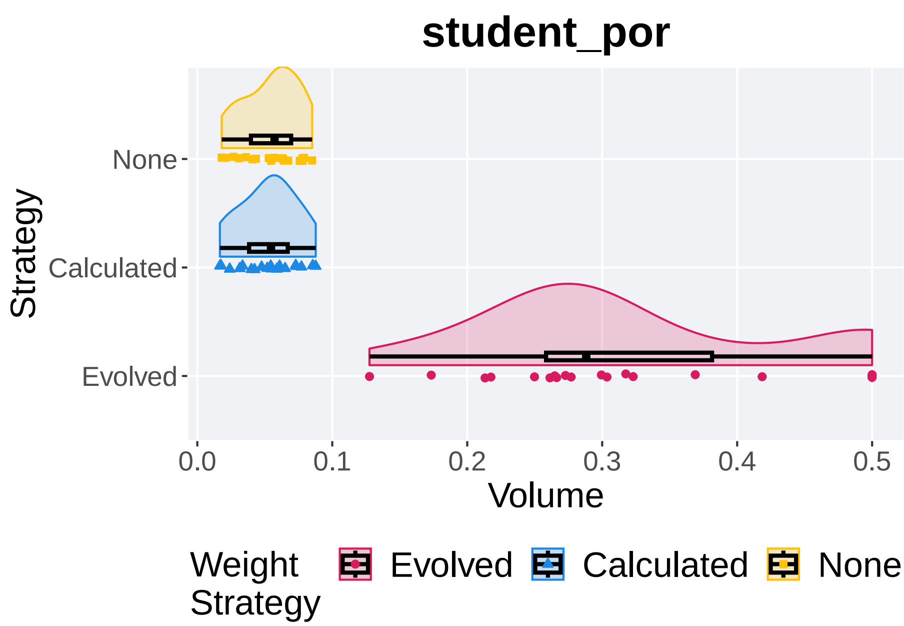
# Student Por

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `student_por` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "student_por")
```

## 5.1 Hypervolume

```
volume_plotter(data,3)
```




### 5.1.1 Summary stats

```
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt    min median   mean    max    IQR
##   <fct>      <int>  <int>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 Evolved       20      0 0.128  0.288  0.318  0.5    0.123
## 2 Calculated    20      0 0.0168 0.0546 0.0528 0.0878 0.0286
## 3 None          20      0 0.0181 0.0573 0.0547 0.0851 0.0298
```

### 5.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 39.429, df = 2, p-value = 2.742e-09
```

### 5.1.3 Pairwise wlcoxon test

```
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 1e-07   -
## None       1e-07   1
##
## P value adjustment method: bonferroni
```

# Chapter 6

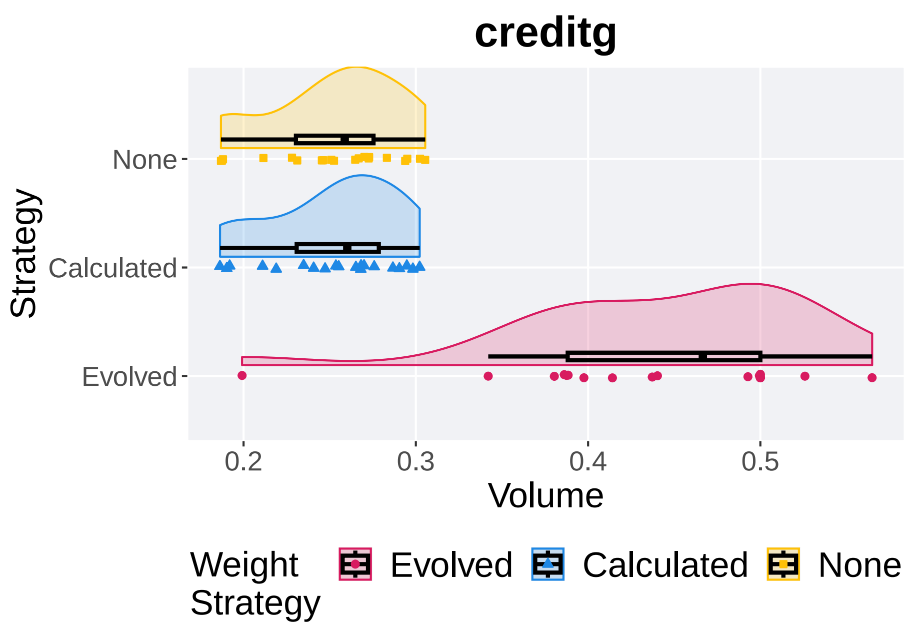
# CreditG

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `creditg` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "creditg")
```

## 6.1 Hypervolume

```
volume_plotter(data,4)
```




### 6.1.1 Summary stats

```
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt   min median  mean   max    IQR
##   <fct>      <int>  <int> <dbl>  <dbl> <dbl> <dbl>  <dbl>
## 1 Evolved       20      0 0.199  0.467 0.443 0.565 0.112
## 2 Calculated    20      0 0.186  0.260 0.252 0.302 0.0477
## 3 None          20      0 0.187  0.259 0.253 0.305 0.0450
```

### 6.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 32.972, df = 2, p-value = 6.922e-08
```

### 6.1.3 Pairwise wlcoxon test

```
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                    paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 1.1e-06 -
## None       1.1e-06 1
##
## P value adjustment method: bonferroni
```

# Chapter 7

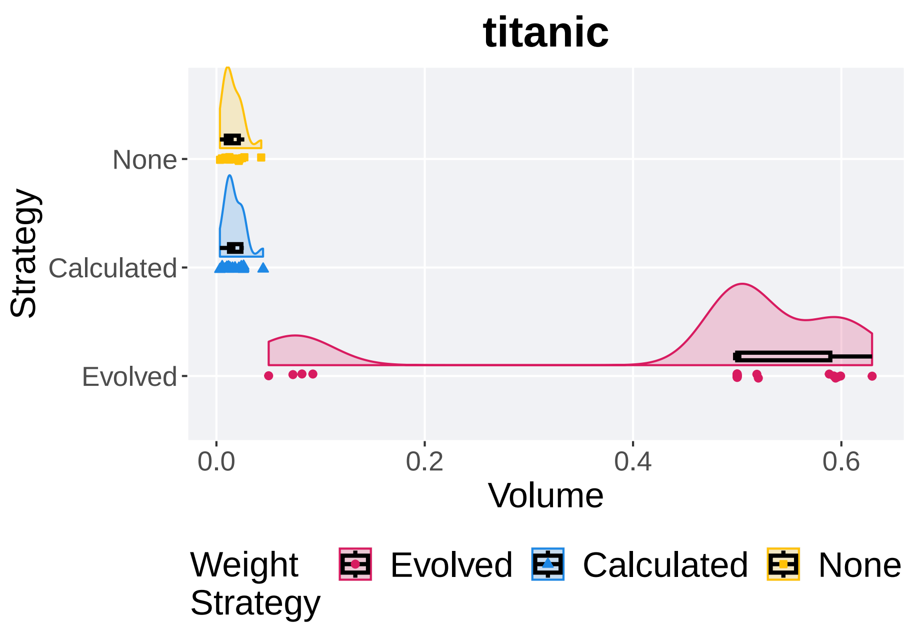
# Titanic

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `titanic` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "titanic")
```

## 7.1 Hypervolume

```
volume_plotter(data,5)
```




### 7.1.1 Summary stats

```
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt     min median   mean    max    IQR
##   <fct>      <int>  <int>   <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 Evolved       20      0 0.0502  0.5    0.447  0.629  0.0894
## 2 Calculated    20      0 0.00334 0.0143 0.0171 0.0448 0.0119
## 3 None          20      0 0.00340 0.0126 0.0157 0.0430 0.0125
```

### 7.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 39.658, df = 2, p-value = 2.445e-09
```

### 7.1.3 Pairwise wlcoxon test

```
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 9e-08   -
## None       9e-08   0.74
##
## P value adjustment method: bonferroni
```

# Chapter 8

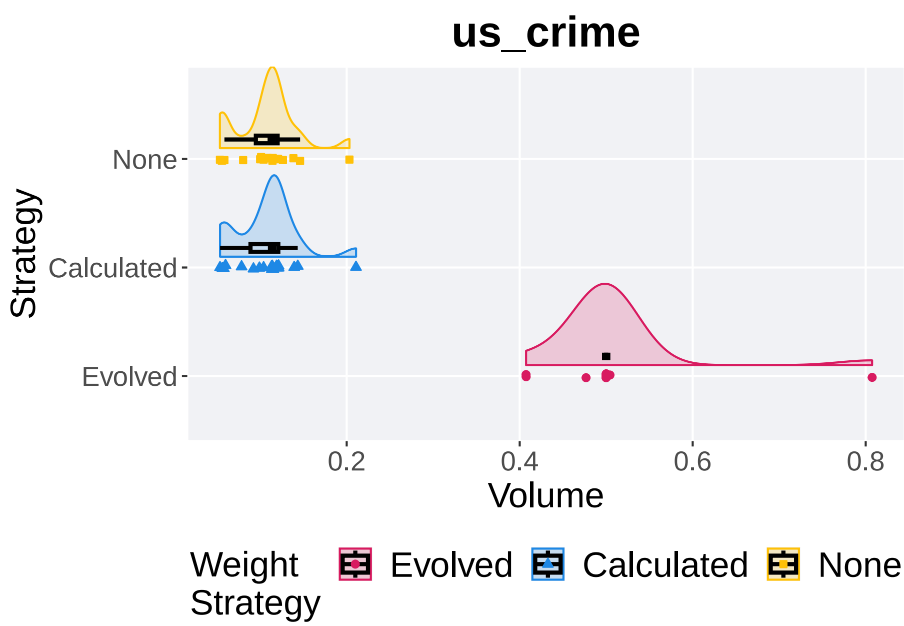
# US Crime

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `us_crime` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "us_crime")
```

## 8.1 Hypervolume

```
volume_plotter(data,6)
```




### 8.1.1 Summary stats

```
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt    min median  mean   max    IQR
##   <fct>      <int>  <int>  <dbl>  <dbl> <dbl> <dbl>  <dbl>
## 1 Evolved       20      0 0.407   0.5   0.505 0.807 0
## 2 Calculated    20      0 0.0536  0.114 0.108 0.211 0.0322
## 3 None          20      0 0.0534  0.113 0.107 0.203 0.0252
```

### 8.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 39.978, df = 2, p-value = 2.084e-09
```

### 8.1.3 Pairwise wlcoxon test

```
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 4.4e-08 -
## None       4.4e-08 1
##
## P value adjustment method: bonferroni
```

# Chapter 9

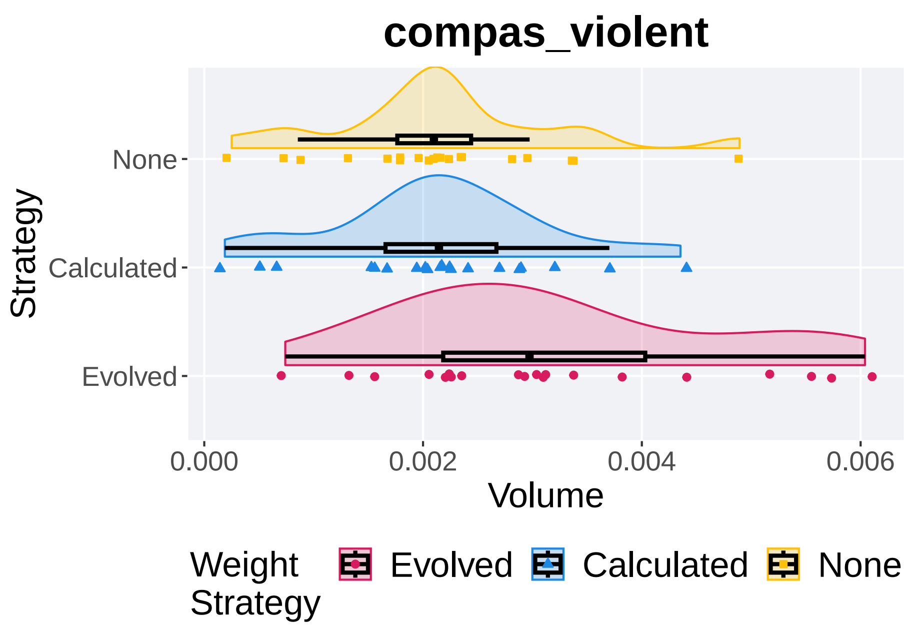
# Compas Violent

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `compas_violent` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "compas_violent")
```

## 9.1 Hypervolume

```
volume_plotter(data,7)
```




### 9.1.1 Summary stats

```
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt      min  median    mean     max      IQR
##   <fct>      <int>  <int>    <dbl>   <dbl>   <dbl>   <dbl>    <dbl>
## 1 Evolved       20      0 0.000741 0.00297 0.00319 0.00604 0.00185
## 2 Calculated    20      0 0.000188 0.00215 0.00215 0.00435 0.00101
## 3 None          20      0 0.000251 0.00210 0.00217 0.00489 0.000675
```

### 9.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 6.7764, df = 2, p-value = 0.03377
```

### 9.1.3 Pairwise wlcoxon test

```
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum exact test
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 0.034   -
## None       0.039   1.000
##
## P value adjustment method: bonferroni
```

# Chapter 10

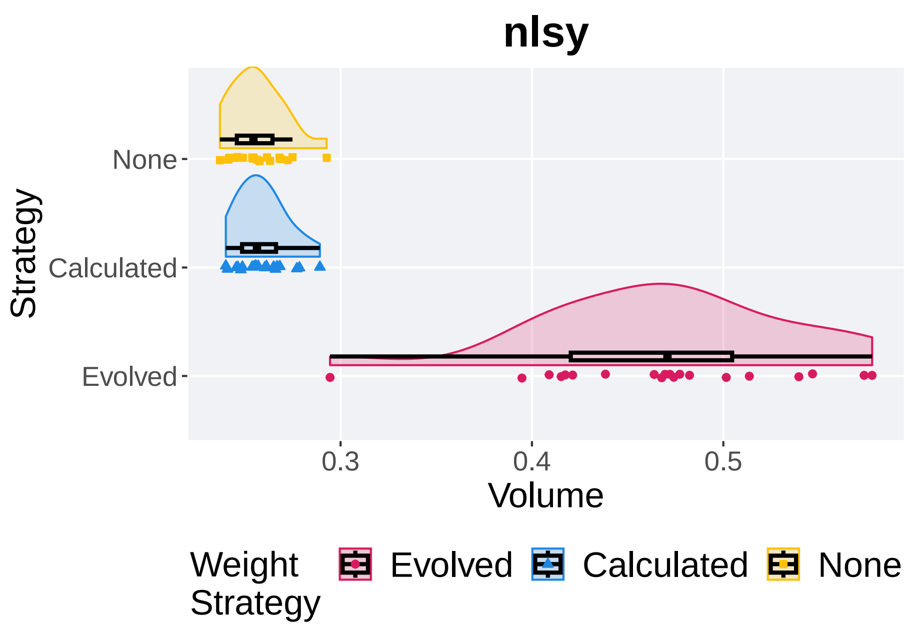
# NLSY

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `nlsy` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "nlsy")
```

## 10.1 Hypervolume

```
volume_plotter(data,8)
```




### 10.1.1 Summary stats

```
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt   min median  mean   max    IQR
##   <fct>      <int>  <int> <dbl>  <dbl> <dbl> <dbl>  <dbl>
## 1 Evolved       20      0 0.294  0.471 0.468 0.578 0.0843
## 2 Calculated    20      0 0.240  0.256 0.259 0.289 0.0177
## 3 None          20      0 0.237  0.254 0.256 0.293 0.0186
```

### 10.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 39.518, df = 2, p-value = 2.623e-09
```

### 10.1.3 Pairwise wlcoxon test

```
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum exact test
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 2.2e-11 -
## None       2.2e-11 0.82
##
## P value adjustment method: bonferroni
```

# Chapter 11

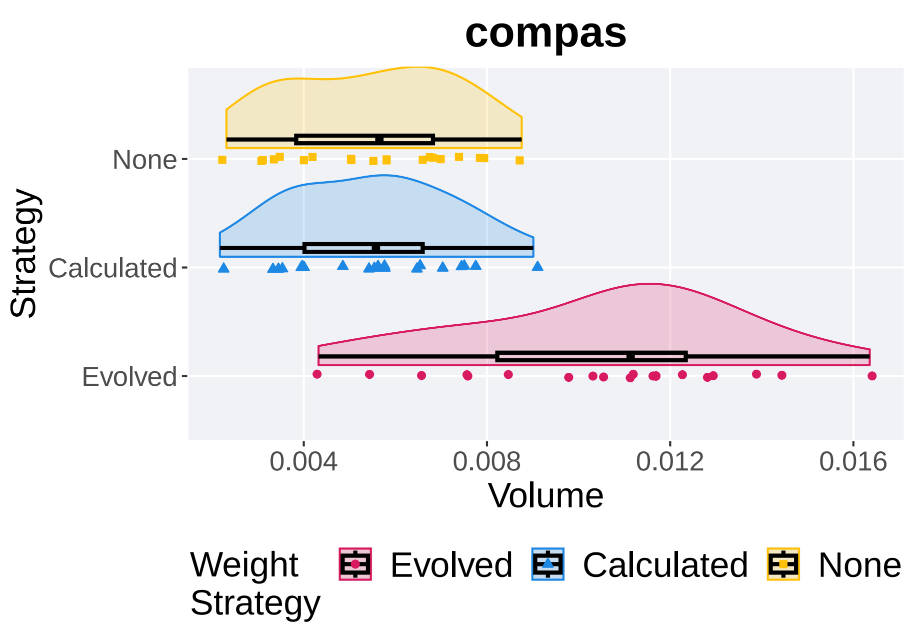
# Compas

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `compas` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "compas")
```

## 11.1 Hypervolume

```
volume_plotter(data,9)
```




### 11.1.1 Summary stats

```
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt     min  median    mean     max     IQR
##   <fct>      <int>  <int>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 Evolved       20      0 0.00432 0.0111  0.0105  0.0164  0.00411
## 2 Calculated    20      0 0.00217 0.00558 0.00546 0.00901 0.00258
## 3 None          20      0 0.00231 0.00565 0.00548 0.00876 0.00299
```

### 11.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 26.298, df = 2, p-value = 1.947e-06
```

### 11.1.3 Pairwise wlcoxon test

```r
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum exact test
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 1.4e-06 -
## None       3.6e-06 1
##
## P value adjustment method: bonferroni
```

# Chapter 12

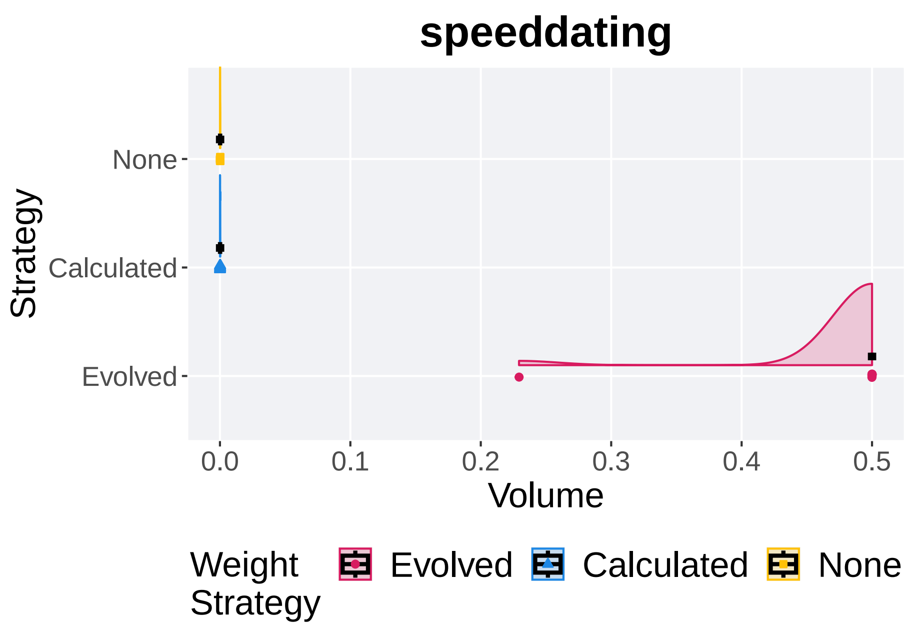
# Speeddating

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `speeddating` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "speeddating")
```

## 12.1 Hypervolume

```
volume_plotter(data,10)
```




### 12.1.1 Summary stats

```
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt       min   median     mean      max      IQR
##   <fct>      <int>  <int>     <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 Evolved       20      0 0.229     0.5      0.486    0.5      0
## 2 Calculated    20      0 0.0000388 0.000102 0.000114 0.000239 0.000121
## 3 None          20      0 0.0000297 0.000109 0.000124 0.000255 0.000122
```

### 12.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 40.672, df = 2, p-value = 1.473e-09
```

### 12.1.3 Pairwise wlcoxon test

```r
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 1.7e-08 -
## None       1.7e-08 1
##
## P value adjustment method: bonferroni
```

# Chapter 13

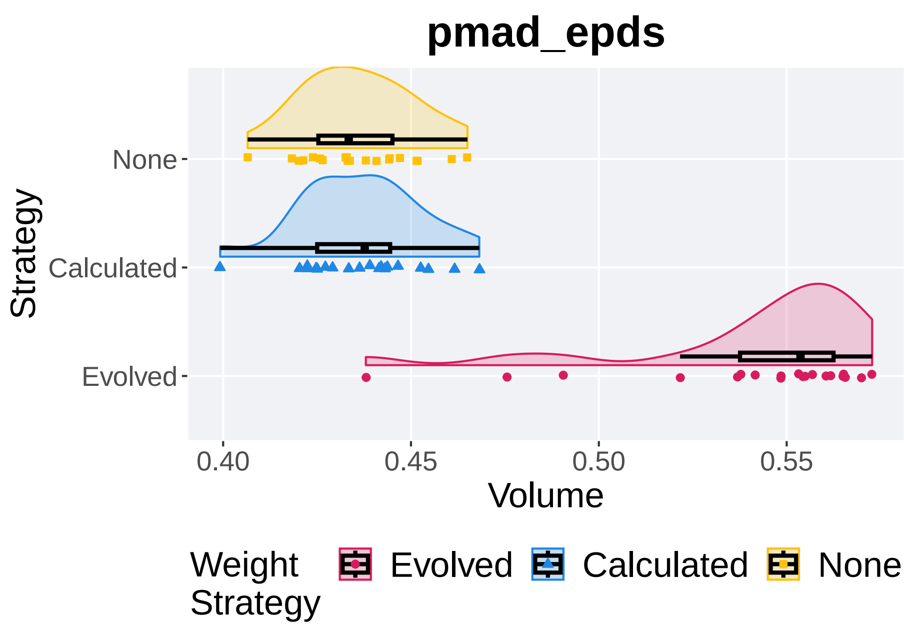
# PMAD EPDS

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `pmad_epds` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "pmad_epds")
```

## 13.1 Hypervolume

```
volume_plotter(data,11)
```




### 13.1.1 Summary stats

```
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt   min median  mean   max    IQR
##   <fct>      <int>  <int> <dbl>  <dbl> <dbl> <dbl>  <dbl>
## 1 Evolved       20      0 0.438  0.554 0.541 0.573 0.0249
## 2 Calculated    20      0 0.399  0.438 0.437 0.468 0.0194
## 3 None          20      0 0.407  0.433 0.436 0.465 0.0197
```

### 13.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 35.731, df = 2, p-value = 1.742e-08
```

### 13.1.3 Pairwise wlcoxon test

```
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum exact test
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 3.0e-09 -
## None       2.1e-09 1
##
## P value adjustment method: bonferroni
```

# Chapter 14

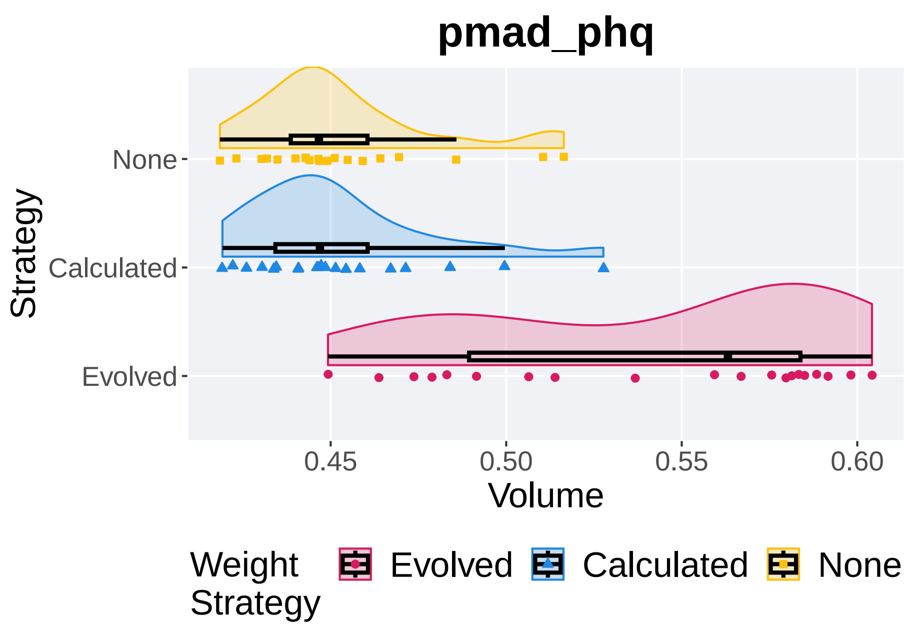
# PMAD PHQ

Here we report the **hypervolume** achived by evaluating the performance of each solution wihtin the Pareto front on the test set of the `pmad_phq` dataset.

```
# heart-disease data
data <- filter(testing, dataset == "pmad_phq")
```

## 14.1 Hypervolume

```
volume_plotter(data,13)
```




### 14.1.1 Summary stats

```
volume_summarize(data)
```

```
## # A tibble: 3 x 8
##   exp        count na_cnt   min median  mean   max    IQR
##   <fct>      <int>  <int> <dbl>  <dbl> <dbl> <dbl>  <dbl>
## 1 Evolved       20      0 0.449  0.563 0.541 0.604 0.0944
## 2 Calculated    20      0 0.419  0.447 0.452 0.528 0.0263
## 3 None          20      0 0.418  0.447 0.453 0.516 0.0218
```

### 14.1.2 Kruskal-Wallis test

Detected differences between weight strategies.

```
kruskal.test(hv ~ exp, data = data)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hv by exp
## Kruskal-Wallis chi-squared = 29.615, df = 2, p-value = 3.708e-07
```

### 14.1.3 Pairwise wlcoxon test

```
pairwise.wilcox.test(x = data$hv, g = data$exp, p.adjust.method = "bonferroni",
                    paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum exact test
##
## data:  data$hv and data$exp
##
##            Evolved Calculated
## Calculated 2.5e-07 -
## None       3.2e-07 1
##
## P value adjustment method: bonferroni
```