

Supplemental Material for ‘Hidden Lexicase
Selection Parameters: Varying Population Size
and Test Case Redundancy with Diagnostic
Metrics’

Jose Guadalupe Hernandez, Anil Kumar Saini, Jason H. Moore

2024-05-15

Contents

1	Introduction	5
1.1	About our supplemental material	5
1.2	Contributing authors	5
2	Exploitation rate results	7
2.1	Analysis setup	7
2.2	Performance over time	8
2.3	Best performance evolved	9
2.4	Evaluation satisfactory solution if found	11
3	Contradictory objectives 100 results	15
3.1	Analysis setup	15
3.2	Activation gene coverage	16
3.3	Satisfactory trait coverage	17
4	Mutation On and Off	23
4.1	Analysis setup	23
4.2	Population size 100	24
4.3	Population size 500	27
4.4	Population size 1000	30
5	Contradictory objectives 200 results	35
5.1	Analysis setup	35
5.2	Activation gene coverage	36
5.3	Satisfactory trait coverage	37

6	Contradictory objectives 300 results	43
6.1	Analysis setup	43
6.2	Activation gene coverage	44
6.3	Satisfactory trait coverage	45
7	Contradictory objectives 500 results	51
7.1	Analysis setup	51
7.2	Activation gene coverage	52
7.3	Satisfactory trait coverage	53

[illegible]

Chapter 1

Introduction

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

1.1 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:

- Exploitation rate results (Section 2)
- Standard contradictory rates results (Section 3)
- Mutation On and Off results (Section 4)
- Contradictory rates results w/ 100 redundant test cases (Section 5)
- Contradictory rates results w/ 200 redundant test cases (Section 6)
- Contradictory rates results w/ 400 redundant test cases (Section 7)

1.2 Contributing authors

- Jose Guadalupe Hernandez
- Anil Kumar Saini
- Jason H. Moore

Chapter 2

Exploitation rate results

Here we report the **performance** and evaluation a **satisfactory solution** was found on the exploitation rate diagnostic. 50 replicates were conducted for each population size explored. Performance is defined at the average trait performance, where we collect the best performing solution in each generation over time and the best performing solution evolved. A satisfactory solution is defined as a solution that has a phenotype with all traits greater than or equal to 99.0.

2.1 Analysis setup

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

# over time data
over_time <- read.csv("../Paper_Data/Exploitation/ot.csv", header = TRUE, stringsAsFactors = FALSE)
over_time$pop_size <- factor(over_time$pop_size, levels = NAMES)

# best performance data
best <- read.csv("../Paper_Data/Exploitation/best.csv", header = TRUE, stringsAsFactors = FALSE)
best$pop_size <- factor(best$pop_size, levels = NAMES)

# get the data
ssf <- read.csv("../Paper_Data/Exploitation/ssf.csv", header = TRUE, stringsAsFactors = FALSE)
ssf$pop_size <- factor(ssf$pop_size, levels = NAMES)
ssf <- filter(ssf, evaluation <= 1.5*10^9)
```

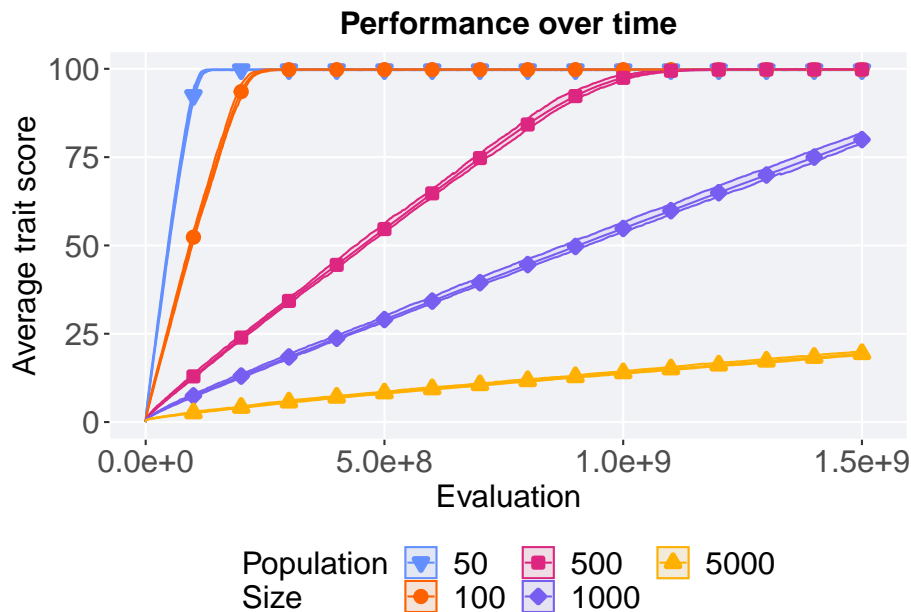
2.2 Performance over time

Performance of the best solution in the population at each generation over time.

```
# aggregate
lines = over_time %>%
  group_by(pop_size, eval) %>%
  dplyr::summarise(
    min = min(performance),
    mean = mean(performance),
    max = max(performance)
  )
lines$pop_size <- factor(lines$pop_size, levels = NAMES)

ggplot(lines, aes(x=eval, y=mean, group = pop_size, fill = pop_size, color = pop_size,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %% 100000000 == 0 & eval != 0), size = 1.0, str
  scale_y_continuous(
    name="Average trait score",
    limits=c(0, 100),
    breaks=seq(0,100, 25),
    labels=c("0", "25", "50", "75", "100")
  ) +
  scale_x_continuous(
    name="Evaluation",
    labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
    limits = c(0,1520000000)

  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Performance over time')+
  p_theme +
  guides(
    shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
  )
```

2.3 Best performance evolved

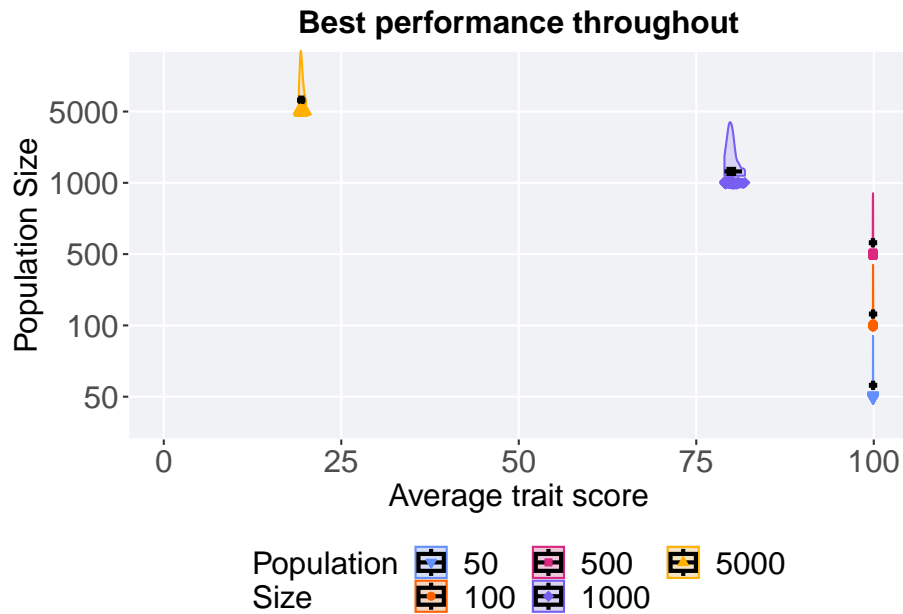
Performance of the best solution found throughout the entire evolutionary run.

```
ggplot(best, aes(x = pop_size, y = performance, color = pop_size, fill = pop_size, shape = pop_size)) +
  geom_flat_violin(position = position_nudge(x = .1, y = 0), scale = 'width', alpha = 0.2, width = 1) +
  geom_boxplot(color = 'black', width = .07, outlier.shape = NA, alpha = 0.0, size = 1.0, position = position_nudge(x = .1, y = 0)) +
  geom_point(position = position_jitter(width = 0.02, height = 0.0001), size = 1.5, alpha = 1.0) +
  scale_y_continuous(
    name="Average trait score",
    limits=c(0, 100),
    breaks=seq(0,100, 25),
    labels=c("0", "25", "50", "75", "100")
  ) +
  scale_x_discrete(
    name="Population Size"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best performance throughout') +
  p_theme+ coord_flip() +
  guides(
```

```

shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
)

```



2.3.1 Summary statistics

```

best %>%
  group_by(pop_size) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(performance)),
    min = min(performance, na.rm = TRUE),
    median = median(performance, na.rm = TRUE),
    mean = mean(performance, na.rm = TRUE),
    max = max(performance, na.rm = TRUE),
    IQR = IQR(performance, na.rm = TRUE)
  )

```

A tibble: 5 x 8

pop_size	count	na_cnt	min	median	mean	max	IQR
<fct>	<int>	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>

```
## 1 50      50      0 99.9  99.9  99.9  99.9 0.0149
## 2 100     50      0 99.9  99.9  99.9  99.9 0.0166
## 3 500     50      0 99.9  99.9  99.9  99.9 0.0235
## 4 1000    50      0 79.0  79.9  80.0  81.8 0.779
## 5 5000    50      0 19.0  19.4  19.4  20.0 0.299
```

2.3.2 Kruskal-Wallis test

```
kruskal.test(performance ~ pop_size, data = best)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  performance by pop_size
## Kruskal-Wallis chi-squared = 198.29, df = 4, p-value < 2.2e-16
```

2.3.3 Pairwise wilcoxon test

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

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  best$performance and best$pop_size
##
##      50      100      500      1000
## 100 1.00000 -          -          -
## 500 0.00078 0.00059 -          -
## 1000 < 2e-16 < 2e-16 < 2e-16 -
## 5000 < 2e-16 < 2e-16 < 2e-16 < 2e-16
##
## P value adjustment method: bonferroni
```

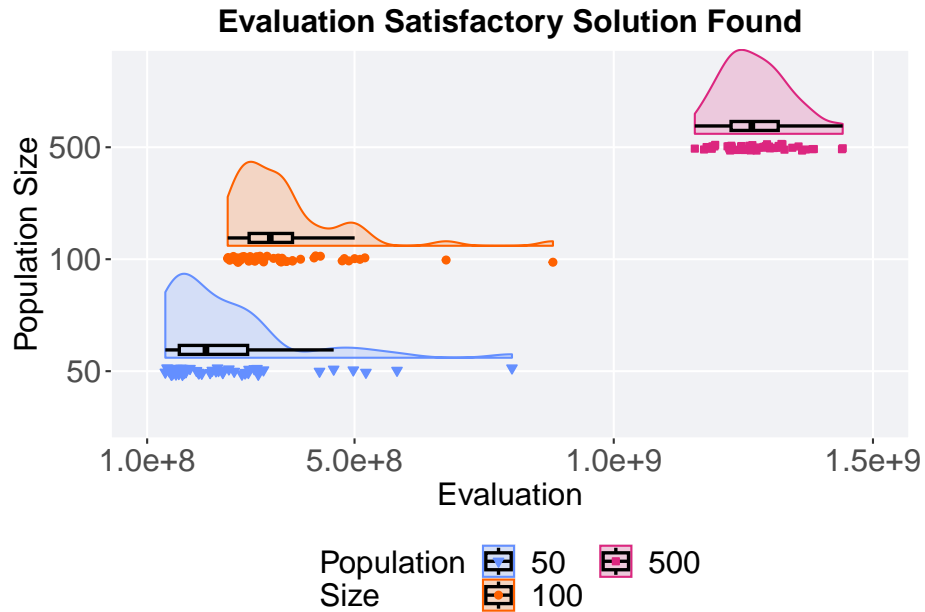
2.4 Evaluation satisfactory solution if found

Evaluation a satisfactory solution is found for each population size.

```

ggplot(ssf, aes(x = pop_size, y = evaluation, color = pop_size, fill = pop_size, shape = pop_size)) +
  geom_flat_violin(position = position_nudge(x = 0.12, y = 0), scale = 'width', alpha = 0.5) +
  geom_boxplot(color = 'black', width = .08, outlier.shape = NA, alpha = 0.0, size = 0.5) +
  geom_point(position = position_jitter(width = 0.03, height = 0.000001), size = 1.5, alpha = 0.5) +
  scale_y_continuous(
    name = 'Evaluation',
    breaks = c(100000000, 500000000, 1000000000, 1500000000),
    labels = c('1.0e+8', '5.0e+8', '1.0e+9', '1.5e+9'),
    limits = c(100000000, 1500000000)
  ) +
  scale_x_discrete(
    name = "Population Size",
  ) +
  scale_shape_manual(values = SHAPE, ) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle(bquote('Evaluation Satisfactory Solution Found')) +
  p_theme + coord_flip() +
  guides(
    shape = guide_legend(nrow = 2, title.position = "left", title = 'Population\nSize'),
    color = guide_legend(nrow = 2, title.position = "left", title = 'Population\nSize'),
    fill = guide_legend(nrow = 2, title.position = "left", title = 'Population\nSize')
  )

```



2.4.1 Summary statistics

```
ssf %>%
  group_by(pop_size) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(evaluation)),
    min = min(evaluation, na.rm = TRUE),
    median = median(evaluation, na.rm = TRUE),
    mean = mean(evaluation, na.rm = TRUE),
    max = max(evaluation, na.rm = TRUE),
    IQR = IQR(evaluation, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
```

##	pop_size	count	na_cnt	min	median	mean	max	IQR
##	<fct>	<int>	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
##	1 50	50	0	135075000	213407500	251829000	803685000	131750000
##	2 100	50	0	255330000	337500000	364654000	882710000	83672500
##	3 500	50	0	1156350000	1266325000	1275005000	1441600000	90962500

2.4.2 Kruskal-Wallis test

```
kruskal.test(evaluation ~ pop_size, data = ssf)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  evaluation by pop_size
## Kruskal-Wallis chi-squared = 113.38, df = 2, p-value < 2.2e-16
```

2.4.3 Pairwise wilcoxon test

```
pairwise.wilcox.test(x = ssf$evaluation, g = ssf$pop_size, p.adjust.method = "bonferroni",
  paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
```

```
## data:  ssf$evaluation and ssf$pop_size
##
##      50      100
## 100 3.1e-08 -
## 500 < 2e-16 < 2e-16
##
## P value adjustment method: bonferroni
```

Chapter 3

Contradictory objectives 100 results

Here we report the **activation gene coverage** and **satisfactory trait coverage** was found on the contradictory objectives diagnostic. 50 replicates were conducted for each population size explored. Activation gene coverage is calculated by finding all the unique activation genes found within a given population. Satisfactory trait coverage is calculated by finding all the unique satisfactory traits found within a given population.

3.1 Analysis setup

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

# over time data
over_time <- read.csv("../Paper_Data/Contradictory-100/ot.csv", header = TRUE, stringsAsFactors = FALSE)
over_time$pop_size <- factor(over_time$pop_size, levels = NAMES)

# best performance data
best <- read.csv("../Paper_Data/Contradictory-100/best.csv", header = TRUE, stringsAsFactors = FALSE)
best$pop_size <- factor(best$pop_size, levels = NAMES)
```

3.2 Activation gene coverage

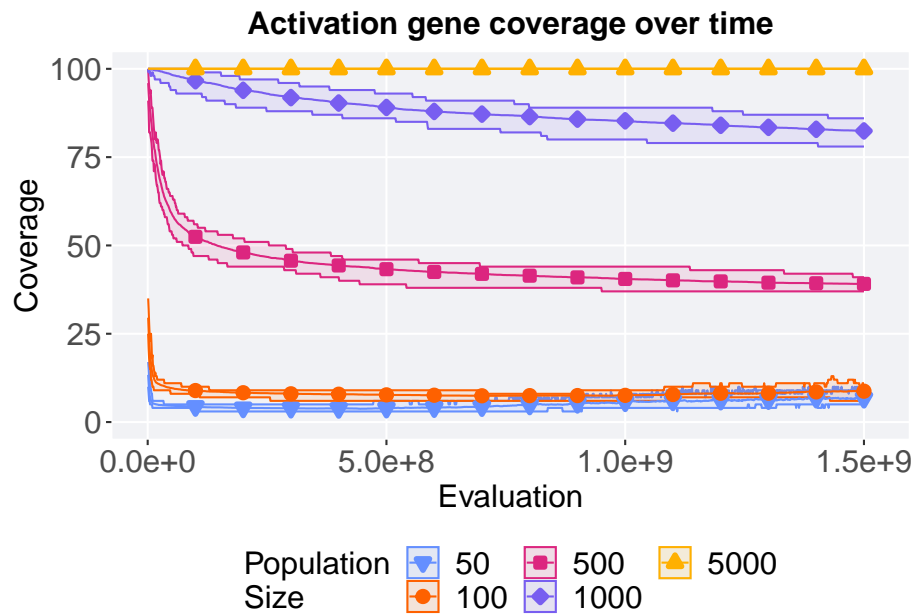
3.2.1 Coverage over time

Performance of the best solution in the population at each generation over time.

```
# aggregate
lines = filter(over_time, eval != 0) %>%
  group_by(pop_size, eval) %>%
  dplyr::summarise(
    min = min(activation_coverage),
    mean = mean(activation_coverage),
    max = max(activation_coverage)
  )
lines$pop_size <- factor(lines$pop_size, levels = NAMES)

ggplot(lines, aes(x=eval, y=mean, group = pop_size, fill = pop_size, color = pop_size,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %% 100000000 == 0 & eval != 0), size = 1.0, str
  scale_y_continuous(
    name="Coverage",
    limits=c(0, 100),
    breaks=seq(0,100, 25),
    labels=c("0", "25", "50", "75", "100")
  ) +
  scale_x_continuous(
    name="Evaluation",
    labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
    limits = c(0,1520000000)

  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Activation gene coverage over time')+
  p_theme +
  guides(
    shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
  )
```

3.3 Satisfactory trait coverage

3.3.1 Coverage over time

Satisfactory trait coverage over time.

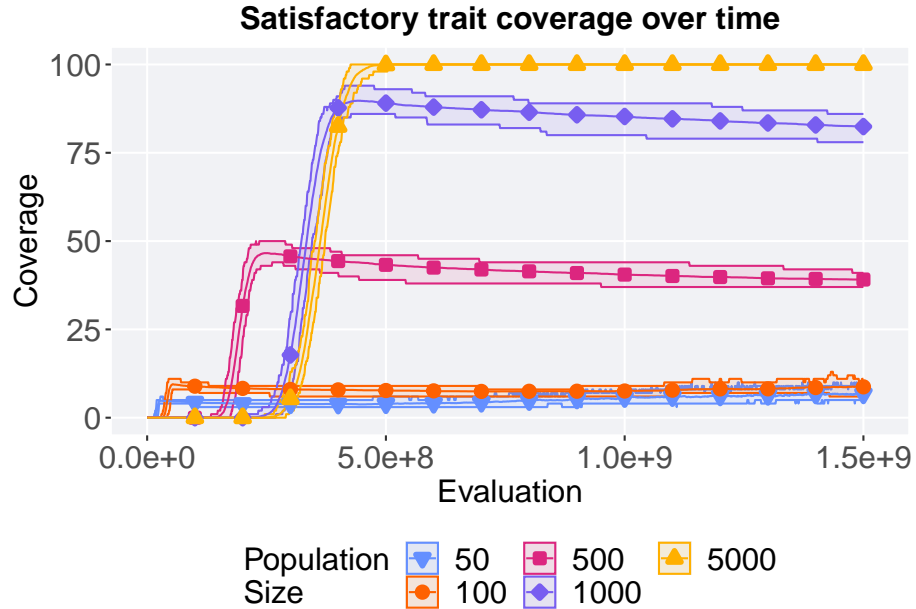
```
# aggregate
lines = over_time %>%
  group_by(pop_size, eval) %>%
  dplyr::summarise(
    min = min(satisfactory_coverage),
    mean = mean(satisfactory_coverage),
    max = max(satisfactory_coverage)
  )
lines$pop_size <- factor(lines$pop_size, levels = NAMES)

ggplot(lines, aes(x=eval, y=mean, group = pop_size, fill = pop_size, color = pop_size, shape = pop_size)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %>% 100000000 == 0 & eval != 0), size = 1.0, stroke = 2.0,
  scale_y_continuous(
    name="Coverage",
    limits=c(0, 100),
```

```

breaks=seq(0,100, 25),
labels=c("0", "25", "50", "75", "100")
) +
scale_x_continuous(
  name="Evaluation",
  labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
  limits = c(0,1520000000)
) +
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle('Satisfactory trait coverage over time')+
p_theme +
guides(
  shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
  color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
  fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
)

```

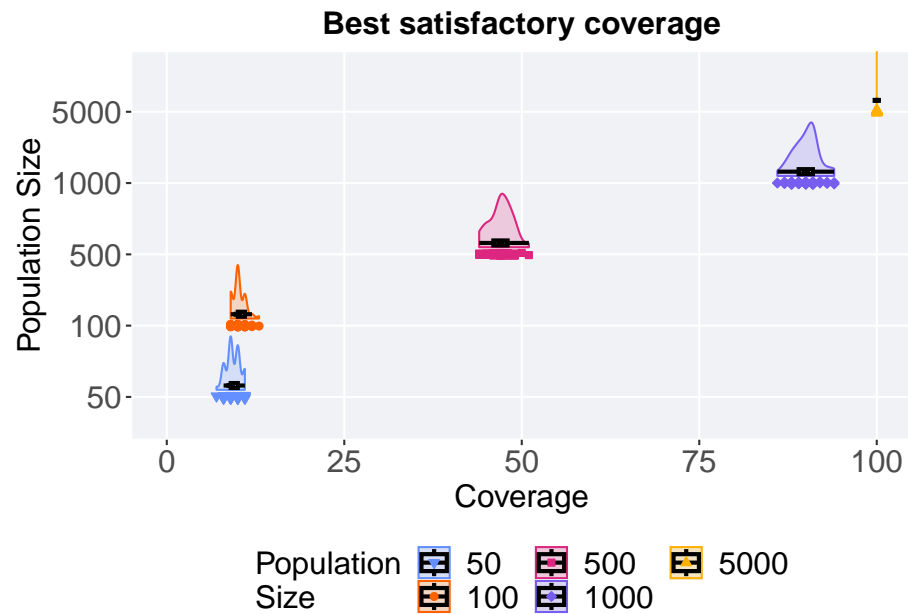


3.3.2 Best satisfactory trait coverage found throughout run

Satisfactory trait coverage of the best population found throughout an evolutionary run.

```
ggplot(best, aes(x = pop_size, y = coverage, color = pop_size, fill = pop_size, shape = pop_size))
  geom_flat_violin(position = position_nudge(x = .1, y = 0), scale = 'width', alpha = 0.2, width
  geom_boxplot(color = 'black', width = .07, outlier.shape = NA, alpha = 0.0, size = 1.0, position
  geom_point(position = position_jitter(width = 0.02, height = 0.0001), size = 1.5, alpha = 1.0)
  scale_y_continuous(
    name="Coverage",
    limits=c(0, 100),
    breaks=seq(0,100, 25),
    labels=c("0", "25", "50", "75", "100"),
  ) +
  scale_x_discrete(
    name="Population Size"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory coverage') +
  p_theme + coord_flip() +
  guides(
    shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
  )
```

```
## Warning: Removed 26 rows containing missing values ('geom_point()').
```



3.3.2.1 Summary statistics

```
best %>%
  group_by(pop_size) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(coverage)),
    min = min(coverage, na.rm = TRUE),
    median = median(coverage, na.rm = TRUE),
    mean = mean(coverage, na.rm = TRUE),
    max = max(coverage, na.rm = TRUE),
    IQR = IQR(coverage, na.rm = TRUE)
  )
```

A tibble: 5 x 8

##	pop_size	count	na_cnt	min	median	mean	max	IQR
##	<fct>	<int>	<int>	<int>	<dbl>	<dbl>	<int>	<dbl>
## 1	50	50	0	7	9	9.36	11	1
## 2	100	50	0	9	10	10.1	13	1
## 3	500	50	0	44	47	47.0	51	2
## 4	1000	50	0	86	90	90.0	94	2
## 5	5000	50	0	100	100	100	100	0

3.3.2.2 Kruskal-Wallis test

```
kruskal.test(coverage ~ pop_size, data = best)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: coverage by pop_size
## Kruskal-Wallis chi-squared = 232.33, df = 4, p-value < 2.2e-16
```

3.3.2.3 Pairwise wilcoxon test

```
pairwise.wilcox.test(x = best$coverage, g = best$pop_size, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: best$coverage and best$pop_size
##
##      50      100      500      1000
## 100 0.0019 -      -      -
## 500 <2e-16 <2e-16 -      -
## 1000 <2e-16 <2e-16 <2e-16 -
## 5000 <2e-16 <2e-16 <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```


Chapter 4

Mutation On and Off

Here we report the **activation gene coverage** and **satisfactory trait coverage** was found on the contradictory objectives diagnostic for runs with mutation on and with mutation off. 50 replicates were conducted for each population size explored. Activation gene coverage is calculated by finding all the unique activation genes found within a given population. Satisfactory trait coverage is calculated by finding all the unique satisfactory traits found within a given population.

4.1 Analysis setup

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

MUTATION <- c('No', 'Yes')

# mutation data
m_data_dir <- '../Paper_Data/Mutation-On-Off/On/Contradictory-100/'
m_over_time <- read.csv(paste(m_data_dir, 'ot.csv', sep = "", collapse = NULL), header = TRUE, stringsAsFactors = FALSE)
m_over_time$pop_size <- factor(m_over_time$pop_size, levels = NAMES)
m_over_time$mutation <- 'Yes'

# no mutation data
n_data_dir <- '../Paper_Data/Mutation-On-Off/Off/Contradictory-100/'
n_over_time <- read.csv(paste(n_data_dir, 'ot.csv', sep = "", collapse = NULL), header = TRUE, stringsAsFactors = FALSE)
n_over_time$pop_size <- factor(n_over_time$pop_size, levels = NAMES)
```

```
n_over_time$mutation <- 'No'

# combine all data
over_time <- rbind(m_over_time, n_over_time)
over_time$mutation <- factor(over_time$mutation, levels=MUTATION)
```

4.2 Population size 100

4.2.1 Activation gene coverage over time

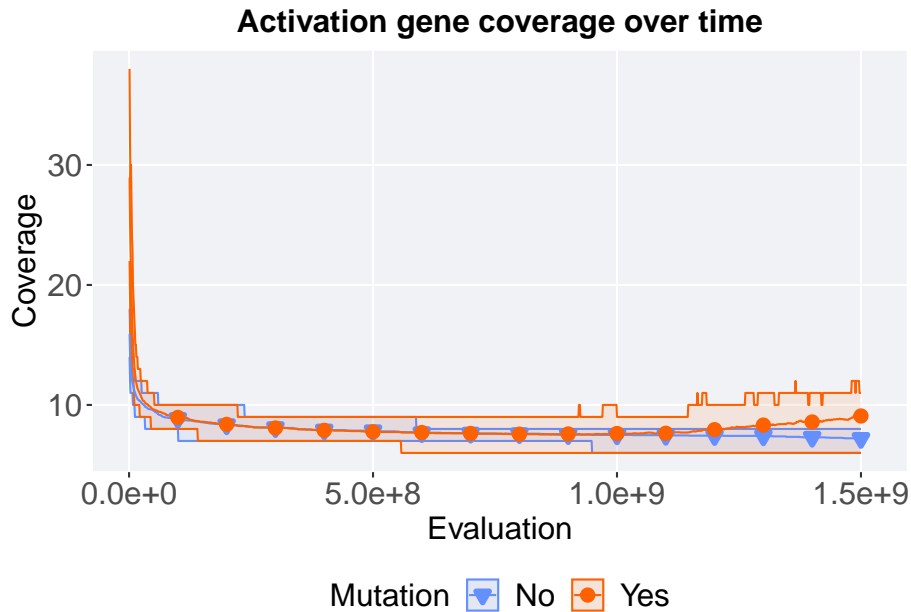
Activation gene coverage found in each generation.

```
# aggregate
lines = filter(over_time, pop_size == 100 & eval > 0) %>%
  group_by(mutation, eval) %>%
  dplyr::summarise(
    min = min(activation_coverage),
    mean = mean(activation_coverage),
    max = max(activation_coverage)
  )
lines$mutation <- factor(lines$mutation, levels = MUTATION)

ggplot(lines, aes(x=eval, y=mean, group = mutation, fill = mutation, color = mutation,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %% 100000000 == 0 & eval != 0), size = 1.0, stroke = "red") +
  scale_y_continuous(
    name="Coverage",
  ) +
  scale_x_continuous(
    name="Evaluation",
    labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
    limits = c(0, 1520000000)
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Activation gene coverage over time') +
  p_theme +
  guides(
    shape=guide_legend(nrow=1, title.position = "left", title = 'Mutation'),
    color=guide_legend(nrow=1, title.position = "left", title = 'Mutation'),
```



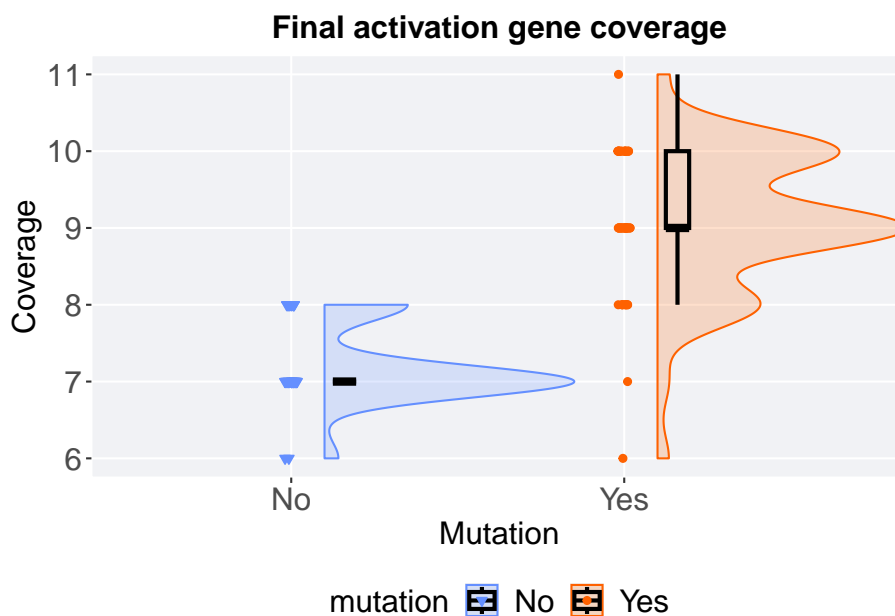
```
fill=guide_legend(nrow=1, title.position = "left", title = 'Mutation')
)
```



4.2.2 Final activation gene coverage

Activation gene coverage found in the final population.

```
filter(over_time, pop_size == 100 & eval == 1.5 * 10^9) %>%
  ggplot(., aes(x = mutation, y = activation_coverage, color = mutation, fill = mutation, shape =
  geom_flat_violin(position = position_nudge(x = .1, y = 0), scale = 'width', alpha = 0.2, width
  geom_boxplot(color = 'black', width = .07, outlier.shape = NA, alpha = 0.0, size = 1.0, positio
  geom_point(position = position_jitter(width = 0.02, height = 0.0001), size = 1.5, alpha = 1.0)
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Mutation"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme
```



4.2.2.1 Summary statistics

```
filter(over_time, pop_size == 100 & eval == 1.5 * 10^9) %>%
  group_by(mutation) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(activation_coverage)),
    min = min(activation_coverage, na.rm = TRUE),
    median = median(activation_coverage, na.rm = TRUE),
    mean = mean(activation_coverage, na.rm = TRUE),
    max = max(activation_coverage, na.rm = TRUE),
    IQR = IQR(activation_coverage, na.rm = TRUE)
  )
```

```
## # A tibble: 2 x 8
##   mutation count na_cnt  min median  mean  max  IQR
##   <fct>    <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 No         50     0     6     7  7.2    8     0
## 2 Yes        50     0     6     9  9.08   11     1
```

4.2.2.2 Wilcoxon test

```
final_100 <- filter(over_time, pop_size == 100 & eval == 1.5 * 10^9)
wilcox.test(final_100$activation_coverage ~ final_100$mutation,
            paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Wilcoxon rank sum test with continuity correction
##
## data: final_100$activation_coverage by final_100$mutation
## W = 133, p-value = 6.163e-16
## alternative hypothesis: true location shift is less than 0
```

4.3 Population size 500

4.3.1 Activation gene coverage over time

Activation gene coverage found in each generation.

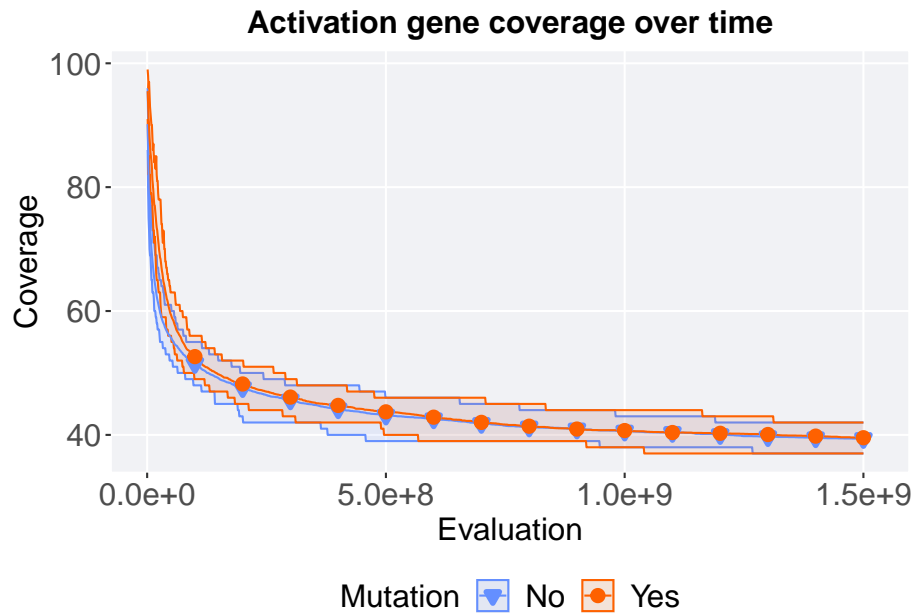
```
# aggregate
lines = filter(over_time, pop_size == 500 & eval > 0) %>%
  group_by(mutation, eval) %>%
  dplyr::summarise(
    min = min(activation_coverage),
    mean = mean(activation_coverage),
    max = max(activation_coverage)
  )
lines$mutation <- factor(lines$mutation, levels = MUTATION)

ggplot(lines, aes(x=eval, y=mean, group = mutation, fill = mutation, color = mutation, shape = mu
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %% 100000000 == 0 & eval != 0), size = 1.0, stroke = 2.0,
  scale_y_continuous(
    name="Coverage",
  ) +
  scale_x_continuous(
    name="Evaluation",
    labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
    limits = c(0, 1520000000)
  ) +
```

```

scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle('Activation gene coverage over time')+
p_theme +
guides(
  shape=guide_legend(nrow=1, title.position = "left", title = 'Mutation'),
  color=guide_legend(nrow=1, title.position = "left", title = 'Mutation'),
  fill=guide_legend(nrow=1, title.position = "left", title = 'Mutation')
)

```



4.3.2 Final activation gene coverage

Activation gene coverage found in the final population.

```

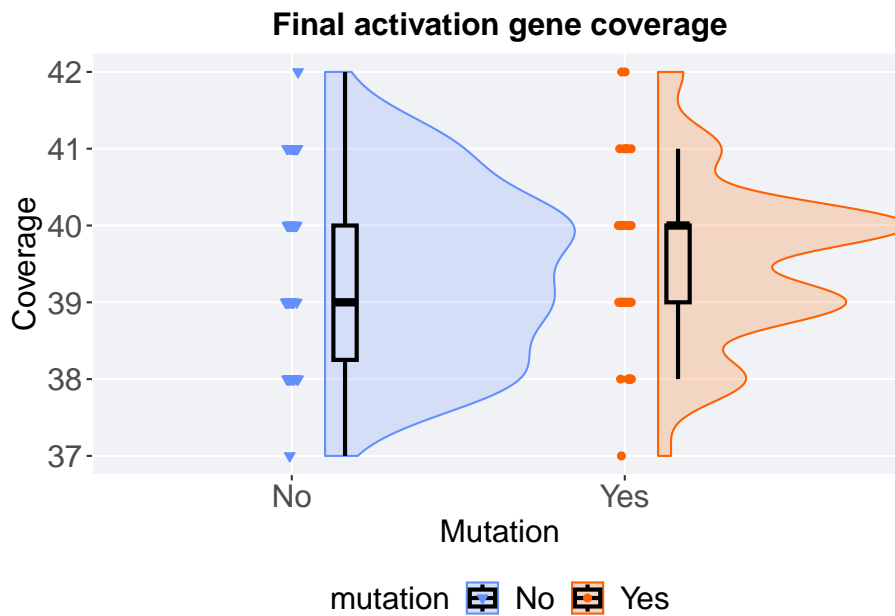
filter(over_time, pop_size == 500 & eval == 1.5 * 10^9) %>%
  ggplot(., aes(x = mutation, y = activation_coverage, color = mutation, fill = mutation)) +
  geom_flat_violin(position = position_nudge(x = .1, y = 0), scale = 'width', alpha = 0.5) +
  geom_boxplot(color = 'black', width = .07, outlier.shape = NA, alpha = 0.0, size = 1) +
  geom_point(position = position_jitter(width = 0.02, height = 0.0001), size = 1.5, alpha = 0.5) +
  scale_y_continuous(
    name="Coverage"
  ) +

```

```

scale_x_discrete(
  name="Mutation"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Final activation gene coverage')+
p_theme

```



4.3.2.1 Summary statistics

```

filter(over_time, pop_size == 500 & eval == 1.5 * 10^9) %>%
group_by(mutation) %>%
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(activation_coverage)),
  min = min(activation_coverage, na.rm = TRUE),
  median = median(activation_coverage, na.rm = TRUE),
  mean = mean(activation_coverage, na.rm = TRUE),
  max = max(activation_coverage, na.rm = TRUE),
  IQR = IQR(activation_coverage, na.rm = TRUE)
)

```

```
## # A tibble: 2 x 8
##   mutation count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 No           50     0    37    39  39.4   42  1.75
## 2 Yes          50     0    37    40  39.5   42   1
```

4.3.2.2 Wilcoxon test

```
final_500 <- filter(over_time, pop_size == 500 & eval == 1.5 * 10^9)
wilcox.test(final_500$activation_coverage ~ final_100$mutation,
             paired = FALSE, conf.int = FALSE, alternative = 't')
```

```
##
## Wilcoxon rank sum test with continuity correction
##
## data: final_500$activation_coverage by final_100$mutation
## W = 1164, p-value = 0.5402
## alternative hypothesis: true location shift is not equal to 0
```

4.4 Population size 1000

4.4.1 Activation gene coverage over time

Activation gene coverage found in each generation.

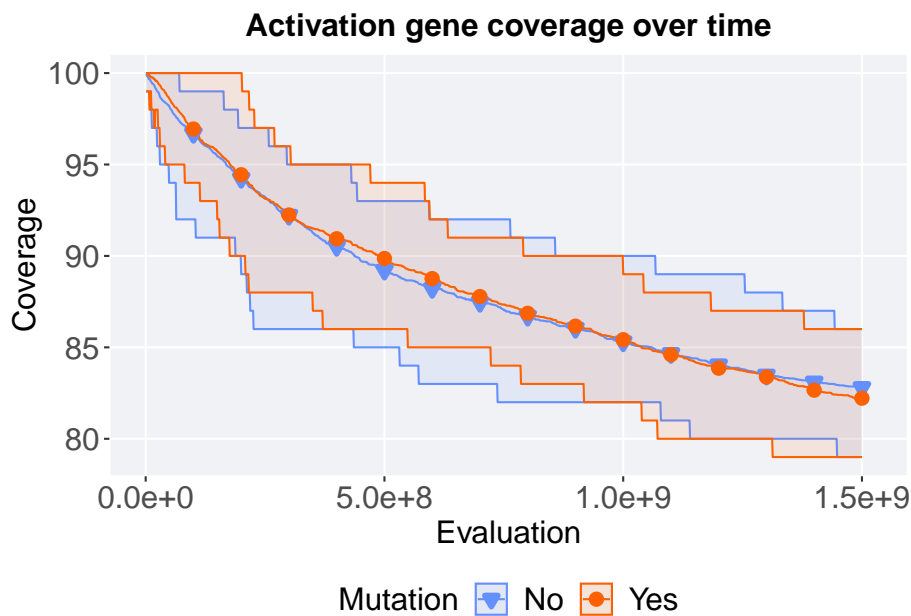
```
# aggregate
lines = filter(over_time, pop_size == 1000 & eval > 0) %>%
  group_by(mutation, eval) %>%
  dplyr::summarise(
    min = min(activation_coverage),
    mean = mean(activation_coverage),
    max = max(activation_coverage)
  )
lines$mutation <- factor(lines$mutation, levels = MUTATION)

ggplot(lines, aes(x=eval, y=mean, group = mutation, fill = mutation, color = mutation,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %% 100000000 == 0 & eval != 0), size = 1.0, str
  scale_y_continuous(
    name="Coverage",
```

```

) +
scale_x_continuous(
  name="Evaluation",
  labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
  limits = c(0, 1520000000)
) +
scale_shape_manual(values=SHAPE) +
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle('Activation gene coverage over time') +
p_theme +
guides(
  shape=guide_legend(nrow=1, title.position = "left", title = 'Mutation'),
  color=guide_legend(nrow=1, title.position = "left", title = 'Mutation'),
  fill=guide_legend(nrow=1, title.position = "left", title = 'Mutation')
)

```



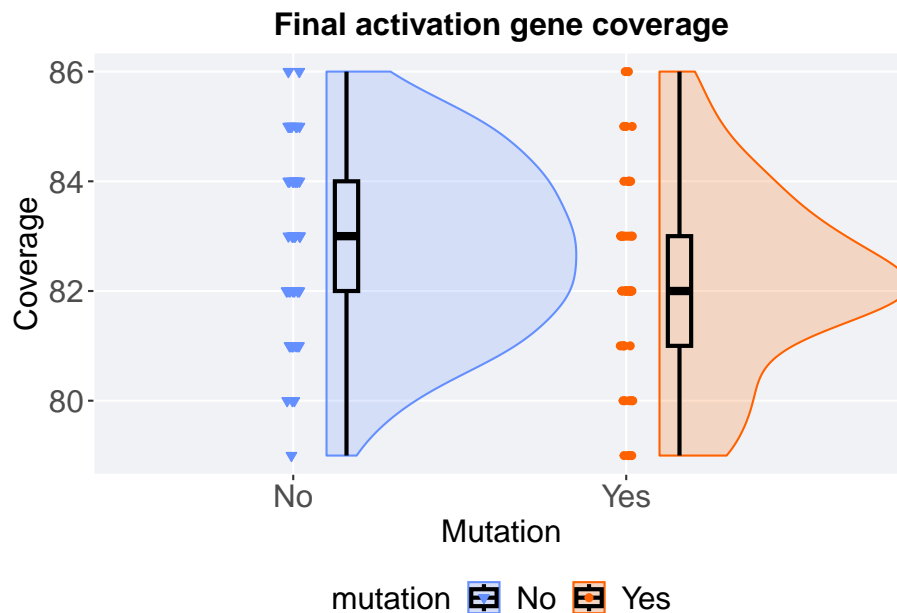
4.4.2 Final activation gene coverage

Activation gene coverage found in the final population.

```

filter(over_time, pop_size == 1000 & eval == 1.5 * 10^9) %>%
  ggplot(., aes(x = mutation, y = activation_coverage, color = mutation, fill = mutation)) +
  geom_flat_violin(position = position_nudge(x = .1, y = 0), scale = 'width', alpha = 0.5) +
  geom_boxplot(color = 'black', width = .07, outlier.shape = NA, alpha = 0.0, size = 1) +
  geom_point(position = position_jitter(width = 0.02, height = 0.0001), size = 1.5, alpha = 0.5) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Mutation"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  theme

```



4.4.2.1 Summary statistics

```

filter(over_time, pop_size == 1000 & eval == 1.5 * 10^9) %>%
  group_by(mutation) %>%
  dplyr::summarise(

```



```

count = n(),
na_cnt = sum(is.na(activation_coverage)),
min = min(activation_coverage, na.rm = TRUE),
median = median(activation_coverage, na.rm = TRUE),
mean = mean(activation_coverage, na.rm = TRUE),
max = max(activation_coverage, na.rm = TRUE),
IQR = IQR(activation_coverage, na.rm = TRUE)
)

```

```

## # A tibble: 2 x 8
##   mutation count na_cnt   min median   mean   max   IQR
##   <fct>     <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 No           50     0    79    83  82.8   86     2
## 2 Yes          50     0    79    82  82.2   86     2

```

4.4.2.2 Wilcoxon test

```

final_1000 <- filter(over_time, pop_size == 1000 & eval == 1.5 * 10^9)
wilcox.test(final_1000$activation_coverage ~ final_100$mutation,
             paired = FALSE, conf.int = FALSE, alternative = 't')

##
## Wilcoxon rank sum test with continuity correction
##
## data: final_1000$activation_coverage by final_100$mutation
## W = 1480, p-value = 0.1079
## alternative hypothesis: true location shift is not equal to 0

```


Chapter 5

Contradictory objectives 200 results

Here we report the **activation gene coverage** and **satisfactory trait coverage** was found on the contradictory objectives diagnostic. 50 replicates were conducted for each population size explored. Activation gene coverage is calculated by finding all the unique activation genes found within a given population. Satisfactory trait coverage is calculated by finding all the unique satisfactory traits found within a given population.

5.1 Analysis setup

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

# over time data
over_time <- read.csv("../Paper_Data/Contradictory-200/ot.csv", header = TRUE, stringsAsFactors = FALSE)
over_time$pop_size <- factor(over_time$pop_size, levels = NAMES)

# best performance data
best <- read.csv("../Paper_Data/Contradictory-200/best.csv", header = TRUE, stringsAsFactors = FALSE)
best$pop_size <- factor(best$pop_size, levels = NAMES)
```

5.2 Activation gene coverage

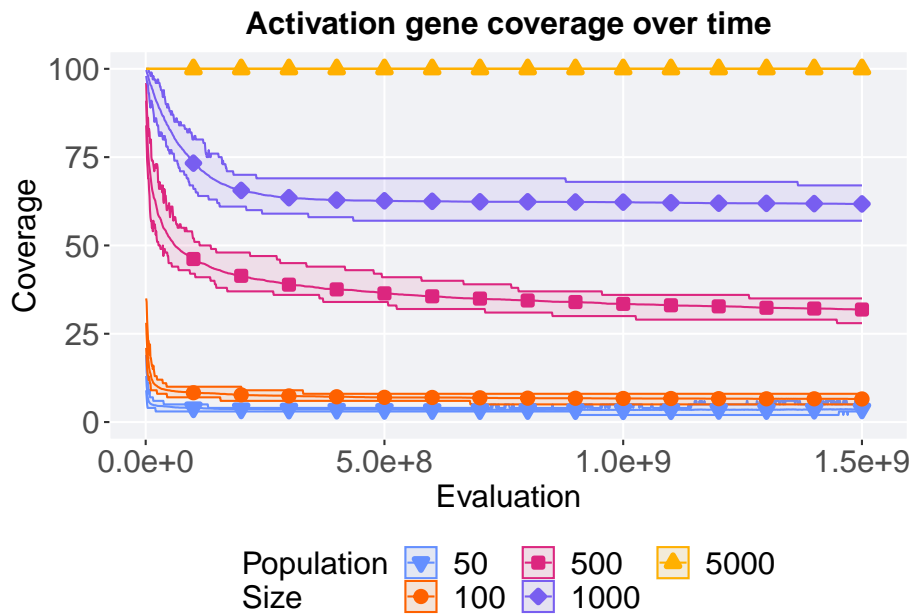
5.2.1 Coverage over time

Performance of the best solution in the population at each generation over time.

```
# aggregate
lines = filter(over_time, eval != 0) %>%
  group_by(pop_size, eval) %>%
  dplyr::summarise(
    min = min(activation_coverage),
    mean = mean(activation_coverage),
    max = max(activation_coverage)
  )
lines$pop_size <- factor(lines$pop_size, levels = NAMES)

ggplot(lines, aes(x=eval, y=mean, group = pop_size, fill = pop_size, color = pop_size,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %% 100000000 == 0 & eval != 0), size = 1.0, str
  scale_y_continuous(
    name="Coverage",
    limits=c(0, 100),
    breaks=seq(0,100, 25),
    labels=c("0", "25", "50", "75", "100")
  ) +
  scale_x_continuous(
    name="Evaluation",
    labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
    limits = c(0,1520000000)

  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Activation gene coverage over time')+
  p_theme +
  guides(
    shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
  )
```



5.3 Satisfactory trait coverage

5.3.1 Coverage over time

Satisfactory trait coverage over time.

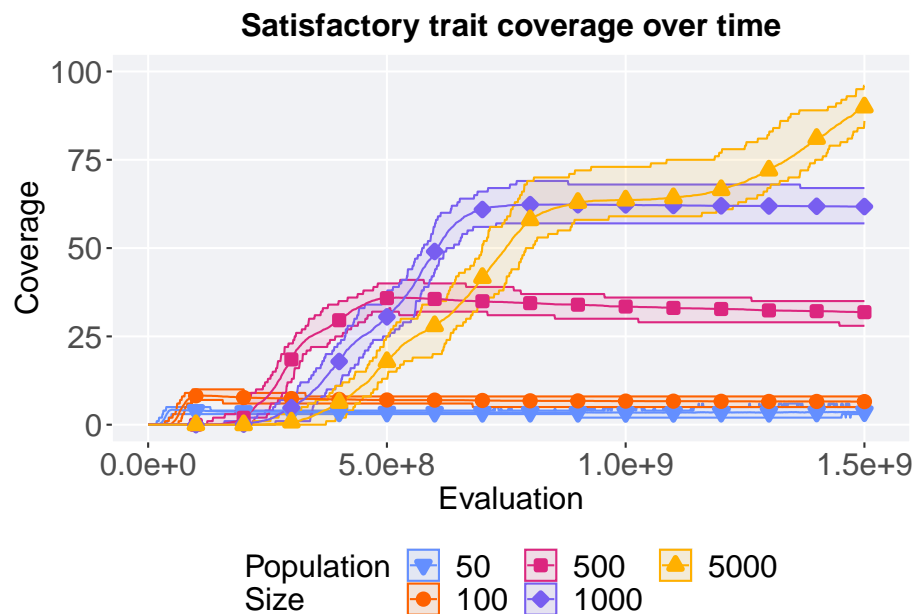
```
# aggregate
lines = over_time %>%
  group_by(pop_size, eval) %>%
  dplyr::summarise(
    min = min(satisfactory_coverage),
    mean = mean(satisfactory_coverage),
    max = max(satisfactory_coverage)
  )
lines$pop_size <- factor(lines$pop_size, levels = NAMES)

ggplot(lines, aes(x=eval, y=mean, group = pop_size, fill = pop_size, color = pop_size, shape = pop_size)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %>% 100000000 == 0 & eval != 0), size = 1.0, stroke = 2.0,
  scale_y_continuous(
    name="Coverage",
    limits=c(0, 100),
```

```

breaks=seq(0,100, 25),
labels=c("0", "25", "50", "75", "100")
) +
scale_x_continuous(
  name="Evaluation",
  labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
  limits = c(0,1520000000)
) +
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle('Satisfactory trait coverage over time')+
p_theme +
guides(
  shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
  color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
  fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
)

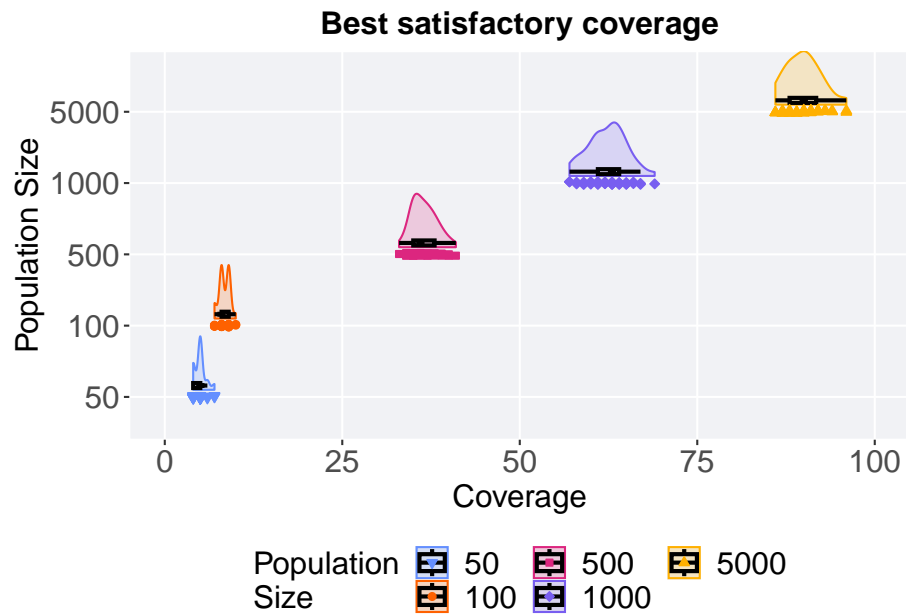
```



5.3.2 Best satisfactory trait coverage found throughout run

Satisfactory trait coverage of the best population found throughout an evolutionary run.

```
ggplot(best, aes(x = pop_size, y = coverage, color = pop_size, fill = pop_size, shape = pop_size))
  geom_flat_violin(position = position_nudge(x = .1, y = 0), scale = 'width', alpha = 0.2, width
  geom_boxplot(color = 'black', width = .07, outlier.shape = NA, alpha = 0.0, size = 1.0, position
  geom_point(position = position_jitter(width = 0.02, height = 0.0001), size = 1.5, alpha = 1.0)
  scale_y_continuous(
    name="Coverage",
    limits=c(0, 100),
    breaks=seq(0,100, 25),
    labels=c("0", "25", "50", "75", "100"),
  ) +
  scale_x_discrete(
    name="Population Size"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory coverage') +
  p_theme + coord_flip() +
  guides(
    shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
  )
```



5.3.2.1 Summary statistics

```
best %>%
  group_by(pop_size) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(coverage)),
    min = min(coverage, na.rm = TRUE),
    median = median(coverage, na.rm = TRUE),
    mean = mean(coverage, na.rm = TRUE),
    max = max(coverage, na.rm = TRUE),
    IQR = IQR(coverage, na.rm = TRUE)
  )
```

A tibble: 5 x 8

##	pop_size	count	na_cnt	min	median	mean	max	IQR
##	<fct>	<int>	<int>	<int>	<dbl>	<dbl>	<int>	<dbl>
## 1	50	50	0	4	5	4.94	7	1
## 2	100	50	0	7	8	8.38	10	1
## 3	500	50	0	33	36	36.5	41	3
## 4	1000	50	0	57	63	62.3	69	3
## 5	5000	50	0	86	90	90.0	96	3.75

5.3.2.2 Kruskal-Wallis test

```
kruskal.test(coverage ~ pop_size, data = best)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: coverage by pop_size
## Kruskal-Wallis chi-squared = 239.68, df = 4, p-value < 2.2e-16
```

5.3.2.3 Pairwise wilcoxon test

```
pairwise.wilcox.test(x = best$coverage, g = best$pop_size, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: best$coverage and best$pop_size
##
##      50      100      500      1000
## 100 <2e-16 -      -      -
## 500 <2e-16 <2e-16 -      -
## 1000 <2e-16 <2e-16 <2e-16 -
## 5000 <2e-16 <2e-16 <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```


Chapter 6

Contradictory objectives 300 results

Here we report the **activation gene coverage** and **satisfactory trait coverage** was found on the contradictory objectives diagnostic. 50 replicates were conducted for each population size explored. Activation gene coverage is calculated by finding all the unique activation genes found within a given population. Satisfactory trait coverage is calculated by finding all the unique satisfactory traits found within a given population.

6.1 Analysis setup

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

# over time data
over_time <- read.csv("../Paper_Data/Contradictory-300/ot.csv", header = TRUE, stringsAsFactors = FALSE)
over_time$pop_size <- factor(over_time$pop_size, levels = NAMES)

# best performance data
best <- read.csv("../Paper_Data/Contradictory-300/best.csv", header = TRUE, stringsAsFactors = FALSE)
best$pop_size <- factor(best$pop_size, levels = NAMES)
```

6.2 Activation gene coverage

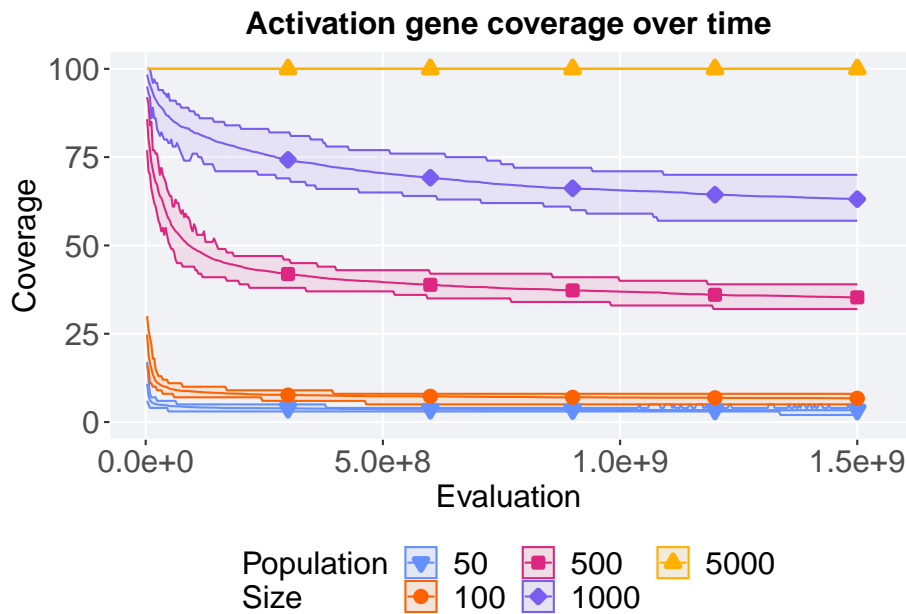
6.2.1 Coverage over time

Performance of the best solution in the population at each generation over time.

```
# aggregate
lines = filter(over_time, eval != 0) %>%
  group_by(pop_size, eval) %>%
  dplyr::summarise(
    min = min(activation_coverage),
    mean = mean(activation_coverage),
    max = max(activation_coverage)
  )
lines$pop_size <- factor(lines$pop_size, levels = NAMES)

ggplot(lines, aes(x=eval, y=mean, group = pop_size, fill = pop_size, color = pop_size,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %% 100000000 == 0 & eval != 0), size = 1.0, str
  scale_y_continuous(
    name="Coverage",
    limits=c(0, 100),
    breaks=seq(0,100, 25),
    labels=c("0", "25", "50", "75", "100")
  ) +
  scale_x_continuous(
    name="Evaluation",
    labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
    limits = c(0,1530000000)

  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Activation gene coverage over time')+
  p_theme +
  guides(
    shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
  )
```



6.3 Satisfactory trait coverage

6.3.1 Coverage over time

Satisfactory trait coverage over time.

```
# aggregate
lines = over_time %>%
  group_by(pop_size, eval) %>%
  dplyr::summarise(
    min = min(satisfactory_coverage),
    mean = mean(satisfactory_coverage),
    max = max(satisfactory_coverage)
  )
lines$pop_size <- factor(lines$pop_size, levels = NAMES)

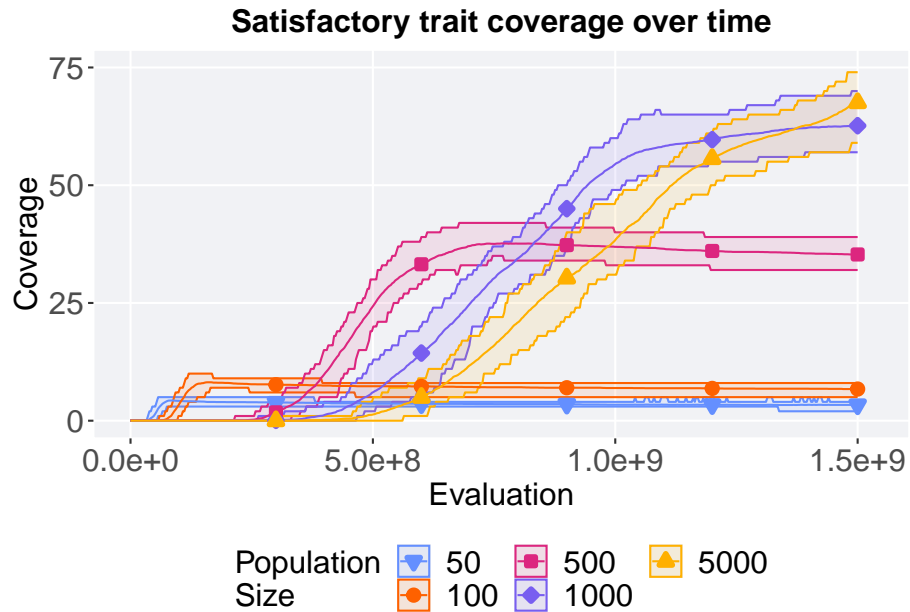
ggplot(lines, aes(x=eval, y=mean, group = pop_size, fill = pop_size, color = pop_size, shape = pop_size)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %% 100000000 == 0 & eval != 0), size = 1.0, stroke = 2.0,
    scale_y_continuous(
      name="Coverage",
      limits=c(0, 75),

```

```

breaks=seq(0,75, 25),
labels=c("0", "25", "50", "75")
) +
scale_x_continuous(
  name="Evaluation",
  labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
  limits = c(0,1530000000)
) +
scale_shape_manual(values=SHAPE) +
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle('Satisfactory trait coverage over time') +
p_theme +
guides(
  shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
  color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
  fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
)

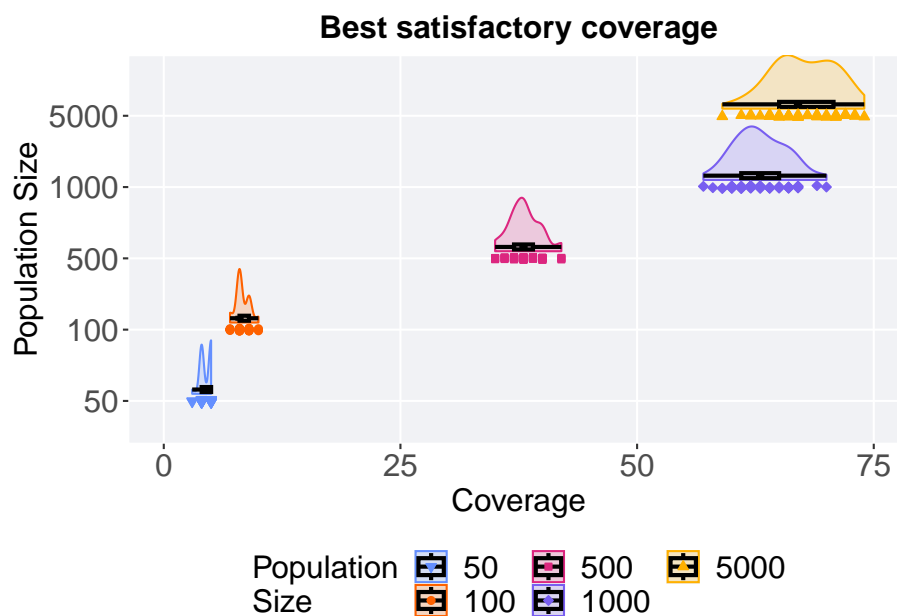
```



6.3.2 Best satisfactory trait coverage found throughout run

Satisfactory trait coverage of the best population found throughout an evolutionary run.

```
ggplot(best, aes(x = pop_size, y = coverage, color = pop_size, fill = pop_size, shape = pop_size)) +
  geom_flat_violin(position = position_nudge(x = .1, y = 0), scale = 'width', alpha = 0.2, width = 1) +
  geom_boxplot(color = 'black', width = .07, outlier.shape = NA, alpha = 0.0, size = 1.0, position = position_nudge(x = .1, y = 0)) +
  geom_point(position = position_jitter(width = 0.02, height = 0.0001), size = 1.5, alpha = 1.0) +
  scale_y_continuous(
    name="Coverage",
    limits=c(0, 75),
    breaks=seq(0,75, 25),
    labels=c("0", "25", "50", "75")
  ) +
  scale_x_discrete(
    name="Population Size"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory coverage') +
  p_theme + coord_flip() +
  guides(
    shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
  )
```



6.3.2.1 Summary statistics

```
best %>%
  group_by(pop_size) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(coverage)),
    min = min(coverage, na.rm = TRUE),
    median = median(coverage, na.rm = TRUE),
    mean = mean(coverage, na.rm = TRUE),
    max = max(coverage, na.rm = TRUE),
    IQR = IQR(coverage, na.rm = TRUE)
  )
```

```
## # A tibble: 5 x 8
##   pop_size count na_cnt  min median  mean  max  IQR
##   <fct>    <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 50         50      0     3   4.5  4.46     5     1
## 2 100        50      0     7     8   8.3    10     1
## 3 500        50      0    35   38  38.1    42     2
## 4 1000       50      0    57   63  63.0    70     4
## 5 5000      50      0    59   67  67.5    74     5.75
```


6.3.2.2 Kruskal-Wallis test

```
kruskal.test(coverage ~ pop_size, data = best)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: coverage by pop_size
## Kruskal-Wallis chi-squared = 233.56, df = 4, p-value < 2.2e-16
```

6.3.2.3 Pairwise wilcoxon test

```
pairwise.wilcox.test(x = best$coverage, g = best$pop_size, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: best$coverage and best$pop_size
##
##      50      100      500      1000
## 100 < 2e-16 -      -      -
## 500 < 2e-16 < 2e-16 -      -
## 1000 < 2e-16 < 2e-16 < 2e-16 -
## 5000 < 2e-16 < 2e-16 < 2e-16 2.3e-08
##
## P value adjustment method: bonferroni
```


Chapter 7

Contradictory objectives 500 results

Here we report the **activation gene coverage** and **satisfactory trait coverage** was found on the contradictory objectives diagnostic. 50 replicates were conducted for each population size explored. Activation gene coverage is calculated by finding all the unique activation genes found within a given population. Satisfactory trait coverage is calculated by finding all the unique satisfactory traits found within a given population.

7.1 Analysis setup

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

# over time data
over_time <- read.csv("../Paper_Data/Contradictory-500/ot.csv", header = TRUE, stringsAsFactors = FALSE)
over_time$pop_size <- factor(over_time$pop_size, levels = NAMES)

# best performance data
best <- read.csv("../Paper_Data/Contradictory-500/best.csv", header = TRUE, stringsAsFactors = FALSE)
best$pop_size <- factor(best$pop_size, levels = NAMES)
```

7.2 Activation gene coverage

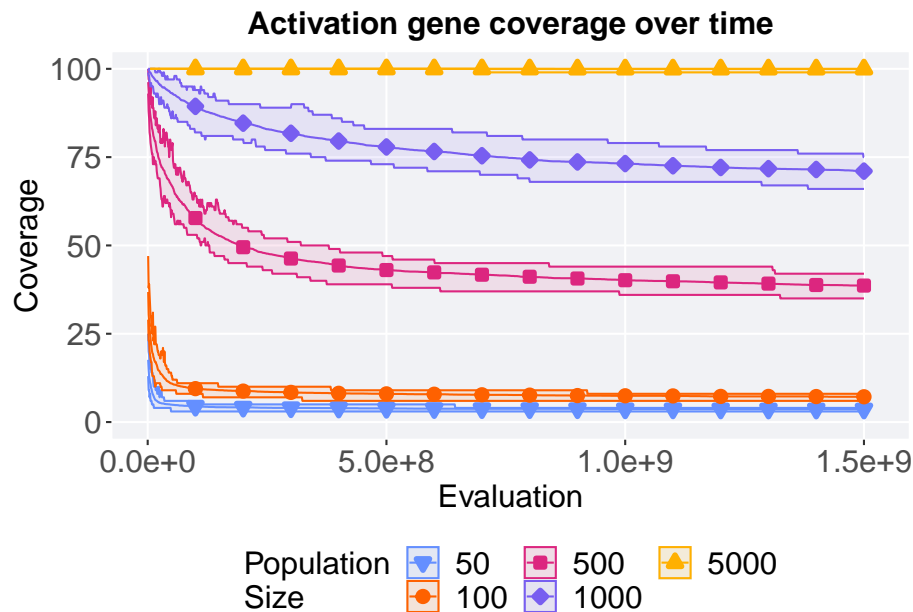
7.2.1 Coverage over time

Performance of the best solution in the population at each generation over time.

```
lines = filter(over_time, eval != 0) %>%
  group_by(pop_size, eval) %>%
  dplyr::summarise(
    min = min(activation_coverage),
    mean = mean(activation_coverage),
    max = max(activation_coverage)
  )
lines$pop_size <- factor(lines$pop_size, levels = NAMES)

ggplot(lines, aes(x=eval, y=mean, group = pop_size, fill = pop_size, color = pop_size,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %% 100000000 == 0 & eval != 0), size = 1.0, str
  scale_y_continuous(
    name="Coverage",
    limits=c(0, 100),
    breaks=seq(0,100, 25),
    labels=c("0", "25", "50", "75", "100")
  ) +
  scale_x_continuous(
    name="Evaluation",
    labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
    limits = c(0,1520000000)

  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Activation gene coverage over time')+
  p_theme +
  guides(
    shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
  )
```



7.3 Satisfactory trait coverage

7.3.1 Coverage over time

Satisfactory trait coverage over time.

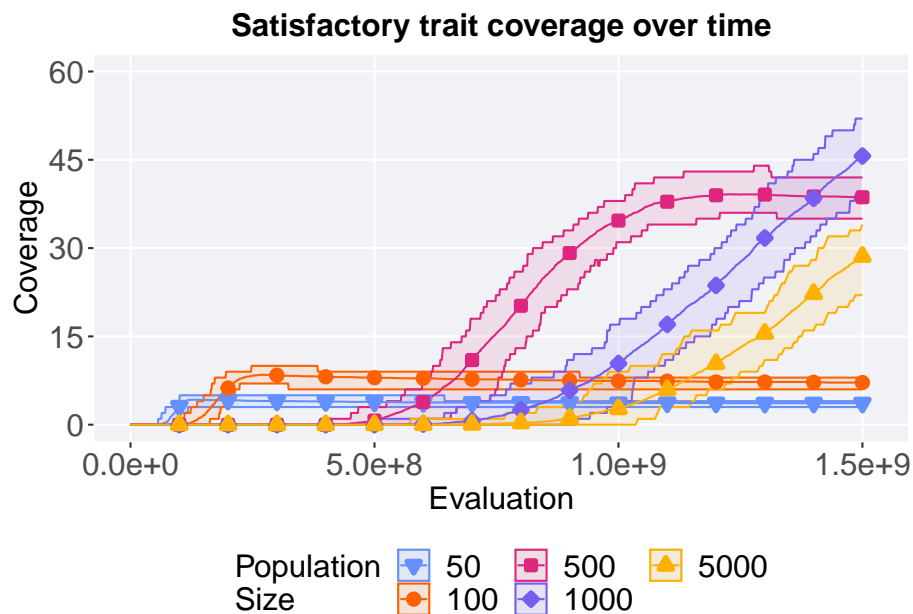
```
# aggregate
lines = over_time %>%
  group_by(pop_size, eval) %>%
  dplyr::summarise(
    min = min(satisfactory_coverage),
    mean = mean(satisfactory_coverage),
    max = max(satisfactory_coverage)
  )
lines$pop_size <- factor(lines$pop_size, levels = NAMES)

ggplot(lines, aes(x=eval, y=mean, group = pop_size, fill = pop_size, color = pop_size, shape = pop_size)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(linewidth = 0.5) +
  geom_point(data = filter(lines, eval %% 100000000 == 0 & eval != 0), size = 1.0, stroke = 2.0,
  scale_y_continuous(
    name="Coverage",
    limits=c(0, 60),
```

```

breaks=seq(0,60, 15),
labels=c("0", "15", "30", "45", "60")
) +
scale_x_continuous(
  name="Evaluation",
  labels = c('0.0e+0', '5.0e+8', '1.0e+9', '1.5e+9'),
  limits = c(0,1520000000)
) +
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle('Satisfactory trait coverage over time')+
p_theme +
guides(
  shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
  color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
  fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
)

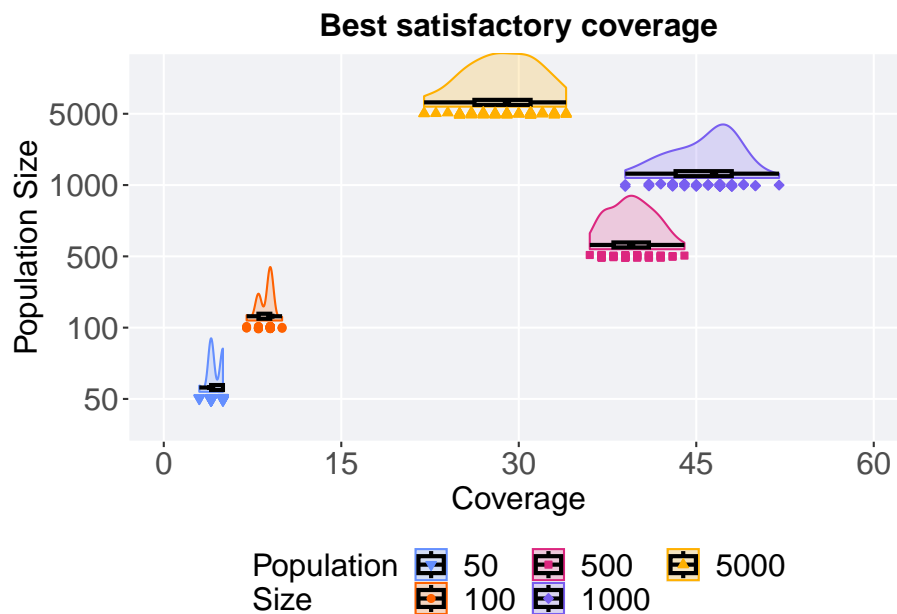
```



7.3.2 Best satisfactory trait coverage found throughout run

Satisfactory trait coverage of the best population found throughout an evolutionary run.

```
ggplot(best, aes(x = pop_size, y = coverage, color = pop_size, fill = pop_size, shape = pop_size))
  geom_flat_violin(position = position_nudge(x = .1, y = 0), scale = 'width', alpha = 0.2, width
  geom_boxplot(color = 'black', width = .07, outlier.shape = NA, alpha = 0.0, size = 1.0, position
  geom_point(position = position_jitter(width = 0.02, height = 0.0001), size = 1.5, alpha = 1.0)
  scale_y_continuous(
    name="Coverage",
    limits=c(0, 60),
    breaks=seq(0,60, 15),
    labels=c("0", "15", "30", "45", "60")
  ) +
  scale_x_discrete(
    name="Population Size"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory coverage') +
  p_theme + coord_flip() +
  guides(
    shape=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    color=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize'),
    fill=guide_legend(nrow=2, title.position = "left", title = 'Population\nSize')
  )
```



7.3.2.1 Summary statistics

```
best %>%
  group_by(pop_size) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(coverage)),
    min = min(coverage, na.rm = TRUE),
    median = median(coverage, na.rm = TRUE),
    mean = mean(coverage, na.rm = TRUE),
    max = max(coverage, na.rm = TRUE),
    IQR = IQR(coverage, na.rm = TRUE)
  )
```

A tibble: 5 x 8

##	pop_size	count	na_cnt	min	median	mean	max	IQR
##	<fct>	<int>	<int>	<int>	<dbl>	<dbl>	<int>	<dbl>
## 1	50	50	0	3	4	4.36	5	1
## 2	100	50	0	7	9	8.62	10	1
## 3	500	50	0	36	39.5	39.6	44	3
## 4	1000	50	0	39	46.5	45.7	52	4.75
## 5	5000	50	0	22	29	28.6	34	4.75

7.3.2.2 Kruskal-Wallis test

```
kruskal.test(coverage ~ pop_size, data = best)

##
## Kruskal-Wallis rank sum test
##
## data: coverage by pop_size
## Kruskal-Wallis chi-squared = 237.63, df = 4, p-value < 2.2e-16
```

7.3.2.3 Pairwise wilcoxon test

```
best$pop_size <- factor(best$pop_size, levels = c(1000,500,5000,100,50))
pairwise.wilcox.test(x = best$coverage, g = best$pop_size, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: best$coverage and best$pop_size
##
##      1000      500      5000      100
## 500 5.7e-14 -          -          -
## 5000 < 2e-16 < 2e-16 -          -
## 100  < 2e-16 < 2e-16 < 2e-16 -
## 50   < 2e-16 < 2e-16 < 2e-16 < 2e-16
##
## P value adjustment method: bonferroni
```