

# Diagnosing Island Supplemental Material

Jose Guadalupe Hernandez

2023-03-06



# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	Computer Setup . . . . .	5
1.2	Experimental setup . . . . .	5
<b>2</b>	<b>BASE: Exploitation rate results</b>	<b>11</b>
2.1	Analysis dependencies . . . . .	11
2.2	Truncation selection . . . . .	11
2.3	Tournament selection . . . . .	15
2.4	Lexicase selection . . . . .	18
<b>3</b>	<b>BASE: Ordered exploitation results</b>	<b>23</b>
3.1	Analysis dependencies . . . . .	23
3.2	Truncation selection . . . . .	23
3.3	Tournament selection . . . . .	27
3.4	Lexicase selection . . . . .	30
<b>4</b>	<b>BASE: Contradictory objectives results</b>	<b>39</b>
4.1	Analysis dependencies . . . . .	39
4.2	Truncation selection . . . . .	39
4.3	Tournament selection . . . . .	49
4.4	Lexicase selection . . . . .	58
<b>5</b>	<b>BASE: Multi-path exploration results</b>	<b>67</b>
5.1	Analysis dependencies . . . . .	67
5.2	Truncation selection . . . . .	67
5.3	Tournament selection . . . . .	78
5.4	Lexicase selection . . . . .	89
<b>6</b>	<b>MI50: Exploitation rate results</b>	<b>99</b>
6.1	Analysis dependencies . . . . .	99
6.2	Truncation selection . . . . .	99
6.3	Tournament selection . . . . .	103
6.4	Lexicase selection . . . . .	106

<b>7</b>	<b>MI50: Ordered exploitation results</b>	<b>111</b>
7.1	Analysis dependencies . . . . .	111
7.2	Truncation selection . . . . .	111
7.3	Tournament selection . . . . .	115
7.4	Lexicase selection . . . . .	118
<b>8</b>	<b>MI50: Contradictory objectives results</b>	<b>127</b>
8.1	Analysis dependencies . . . . .	127
8.2	Truncation selection . . . . .	127
8.3	Tournament selection . . . . .	137
8.4	Lexicase selection . . . . .	146
<b>9</b>	<b>MI50: Multi-path exploration results</b>	<b>155</b>
9.1	Analysis dependencies . . . . .	155
9.2	Truncation selection . . . . .	155
9.3	Tournament selection . . . . .	166
9.4	Lexicase selection . . . . .	177
<b>10</b>	<b>MI5000: Exploitation rate results</b>	<b>187</b>
10.1	Analysis dependencies . . . . .	187
10.2	Truncation selection . . . . .	187
10.3	Tournament selection . . . . .	191
10.4	Lexicase selection . . . . .	194
<b>11</b>	<b>MI5000: Ordered exploitation results</b>	<b>199</b>
11.1	Analysis dependencies . . . . .	199
11.2	Truncation selection . . . . .	199
11.3	Tournament selection . . . . .	203
11.4	Lexicase selection . . . . .	206
<b>12</b>	<b>MI5000: Contradictory objectives results</b>	<b>215</b>
12.1	Analysis dependencies . . . . .	215
12.2	Truncation selection . . . . .	215
12.3	Tournament selection . . . . .	225
12.4	Lexicase selection . . . . .	234
<b>13</b>	<b>MI5000: Multi-path exploration results</b>	<b>243</b>
13.1	Analysis dependencies . . . . .	243
13.2	Truncation selection . . . . .	243
13.3	Tournament selection . . . . .	254
13.4	Lexicase selection . . . . .	265

# Chapter 1

## Introduction

This is the supplemental material associated with the 6th chapter in my dissertation.

### 1.1 Computer Setup

These analyses were conducted in the following computing environment:

```
print(version)
```

```
##  
## platform      _  
## arch          x86_64-pc-linux-gnu  
## os            linux-gnu  
## system        x86_64, linux-gnu  
## status        Patched  
## major         4  
## minor         2.2  
## year          2022  
## month         11  
## day           10  
## svn rev       83330  
## language      R  
## version.string R version 4.2.2 Patched (2022-11-10 r83330)  
## nickname      Innocent and Trusting
```

### 1.2 Experimental setup

Setting up required variables variables.

```

# libraries we are using
library(ggplot2)
library(cowplot)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##      filter, lag

## The following objects are masked from 'package:base':
##
##      intersect, setdiff, setequal, union

library(PupillometryR)

## Loading required package: rlang

p_theme <- theme(
  text = element_text(size = 28),
  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 = 14),
  axis.title = element_text(size = 18),
  axis.text = element_text(size = 16),
  legend.position = "bottom",
  panel.background = element_rect(fill = "#f1f2f5",
                                   colour = "white",
                                   size = 0.5, linetype = "solid")
)

## Warning: The `size` argument of `element_rect()` is deprecated as of ggplot2 3.4.0.
## i Please use the `linewidth` argument instead.

# default variables

MODEL = c('EA', 'IS', 'NMIS')
EXPERIMENTS = c('BASE-EXPERIMENTS/', 'MI50/', 'MI5000/')
SCHEME = c('TRUNCATION', 'TOURNAMENT', 'LEXICASE')
DIAGNOSTIC = c('EXPLOITATION_RATE', 'ORDERED_EXPLOITATION', 'CONTRADICTION_OBJECTIVES')
DIMENSIONALITY = 100
cb_palette <- c('#D81B60', '#1E88E5', '#FFC107')
SHAPE = c(15, 16, 17)
TSIZE = 20

```

```

GENERATIONS = 50000

# data related
DATA_DIR = '/opt/Diagnosing-Island-Structures/DATA-FINAL/'

# go through each diagnostic and collect over time data for cross comparison (cc)
base_over_time = data.frame()
mi50_over_time = data.frame()
mi5000_over_time = data.frame()
print('over time data')

## [1] "over time data"
for(model in MODEL)
{
  print(model)
  for(scheme in SCHEME)
  {
    base_dir = paste(DATA_DIR,EXPERIMENTS[1],model,'/over-time-',scheme, '.csv', sep = "", collapse = "/")
    base_over_time = rbind(base_over_time, read.csv(base_dir, header = TRUE, stringsAsFactors = F))

    mi50_dir = paste(DATA_DIR,EXPERIMENTS[2],model,'/over-time-',scheme, '.csv', sep = "", collapse = "/")
    mi50_over_time = rbind(mi50_over_time, read.csv(mi50_dir, header = TRUE, stringsAsFactors = F))

    mi5000_dir = paste(DATA_DIR,EXPERIMENTS[3],model,'/over-time-',scheme, '.csv', sep = "", collapse = "/")
    mi5000_over_time = rbind(mi5000_over_time, read.csv(mi5000_dir, header = TRUE, stringsAsFactors = F))

  }
}

## [1] "EA"
## [1] "IS"
## [1] "NMIS"

colnames(base_over_time)[colnames(base_over_time) == "SEL"] = 'Selection\nScheme'
base_over_time$Structure <- factor(base_over_time$Structure, levels = MODEL)
base_over_time$sel_pre = base_over_time$sel_pre * -1.0

colnames(mi50_over_time)[colnames(mi50_over_time) == "SEL"] = 'Selection\nScheme'
mi50_over_time$Structure <- factor(mi50_over_time$Structure, levels = MODEL)
mi50_over_time$sel_pre = mi50_over_time$sel_pre * -1.0

colnames(mi5000_over_time)[colnames(mi5000_over_time) == "SEL"] = 'Selection\nScheme'
mi5000_over_time$Structure <- factor(mi5000_over_time$Structure, levels = MODEL)
mi5000_over_time$sel_pre = mi5000_over_time$sel_pre * -1.0

```

```

# go through each diagnostic and collect best over time for cross comparison (cc)
base_best = data.frame()
mi50_best = data.frame()
mi5000_best = data.frame()
print('best data')

## [1] "best data"
for(model in MODEL)
{
  print(model)
  for(scheme in SCHEME)
  {
    base_dir = paste(DATA_DIR,EXPERIMENTS[1],model,'/best-',scheme, '.csv', sep = "", collapse = "/")
    base_best = rbind(base_best, read.csv(base_dir, header = TRUE, stringsAsFactors = FALSE))

    mi50_dir = paste(DATA_DIR,EXPERIMENTS[2],model,'/best-',scheme, '.csv', sep = "", collapse = "/")
    mi50_best = rbind(mi50_best, read.csv(mi50_dir, header = TRUE, stringsAsFactors = FALSE))

    mi5000_dir = paste(DATA_DIR,EXPERIMENTS[3],model,'/best-',scheme, '.csv', sep = "", collapse = "/")
    mi5000_best = rbind(mi5000_best, read.csv(mi5000_dir, header = TRUE, stringsAsFactors = FALSE))
  }
}

## [1] "EA"
## [1] "IS"
## [1] "NMIS"

colnames(base_best)[colnames(base_best) == "SEL"] = 'Selection\nScheme'
base_best$Structure <- factor(base_best$Structure, levels = MODEL)

colnames(mi50_best)[colnames(mi50_best) == "SEL"] = 'Selection\nScheme'
mi50_best$Structure <- factor(mi50_best$Structure, levels = MODEL)

colnames(mi5000_best)[colnames(mi5000_best) == "SEL"] = 'Selection\nScheme'
mi5000_best$Structure <- factor(mi5000_best$Structure, levels = MODEL)

# get generation a satisfactory solution is found for cross comparison (cc)
base_ssf = data.frame()
mi50_ssf = data.frame()
mi5000_ssf = data.frame()
print('ssf data')

## [1] "ssf data"
for(model in MODEL)
{

```



```

print(model)
for(scheme in SCHEME)
{
  base_dir = paste(DATA_DIR,EXPERIMENTS[1],model,'/ssf-',scheme, '.csv', sep = "", collapse = M
  base_ssf = rbind(base_ssf, read.csv(base_dir, header = TRUE, stringsAsFactors = FALSE))

  mi50_dir = paste(DATA_DIR,EXPERIMENTS[2],model,'/ssf-',scheme, '.csv', sep = "", collapse = M
  mi50_ssf = rbind(mi50_ssf, read.csv(mi50_dir, header = TRUE, stringsAsFactors = FALSE))

  mi5000_dir = paste(DATA_DIR,EXPERIMENTS[3],model,'/ssf-',scheme, '.csv', sep = "", collapse = M
  mi5000_ssf = rbind(mi5000_ssf, read.csv(mi5000_dir, header = TRUE, stringsAsFactors = FALSE))
}
}

## [1] "EA"
## [1] "IS"
## [1] "NMIS"

colnames(base_ssf)[colnames(base_ssf) == "SEL"] = 'Selection\nScheme'
base_ssf$Structure <- factor(base_ssf$Structure, levels = MODEL)

colnames(mi50_ssf)[colnames(mi50_ssf) == "SEL"] = 'Selection\nScheme'
mi50_ssf$Structure <- factor(mi50_ssf$Structure, levels = MODEL)

colnames(mi5000_ssf)[colnames(mi5000_ssf) == "SEL"] = 'Selection\nScheme'
mi5000_ssf$Structure <- factor(mi5000_ssf$Structure, levels = MODEL)

```



## Chapter 2

# BASE: Exploitation rate results

Here we present the results for **best performances** found by each selection scheme replicate on the exploitation rate diagnostic with our base configurations. For our base configuration, we assume that there are migrations every 500 generations, 4 islands, and a ring topology. When migrations occur, we swap two individuals (same position on each island) and guarantee that no solution can return to the same island. Best performance found refers to the largest average trait score found in a given population. Note that performance values fall between 0.0 and 100.0.

### 2.1 Analysis dependencies

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

### 2.2 Truncation selection

Here we analyze how the different population structures affect truncation selection (size 8) on the exploitation rate diagnostic.

#### 2.2.1 Performance over time

```
lines = filter(base_over_time, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TRUNCATION') %>%
  group_by(Structure, Generations) %>%
```

```

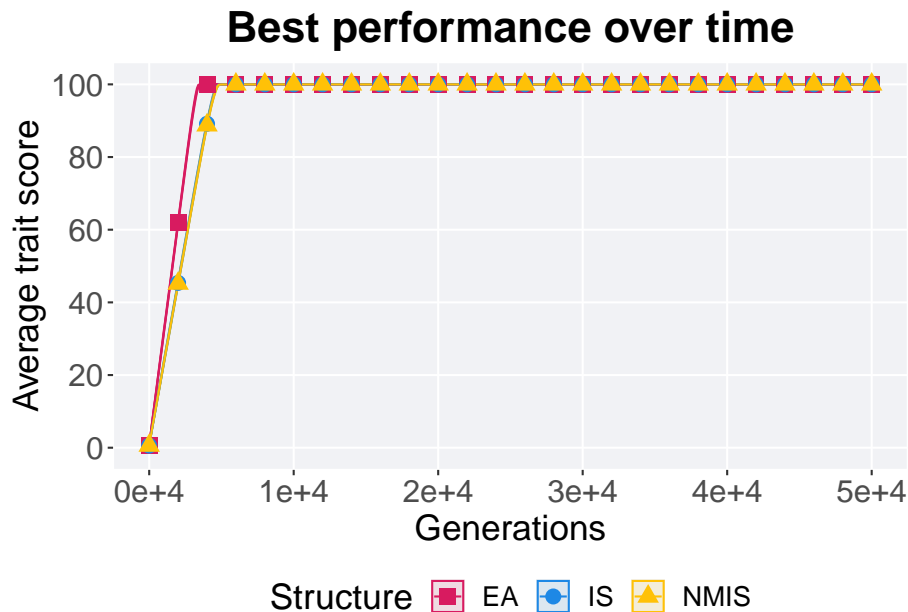
dplyr::summarise(
  min = min(pop_fit_max) / DIMENSIONALITY,
  mean = mean(pop_fit_max) / DIMENSIONALITY,
  max = max(pop_fit_max) / DIMENSIONALITY
)
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0,
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Best performance over time") +
  p_theme

```

```

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.

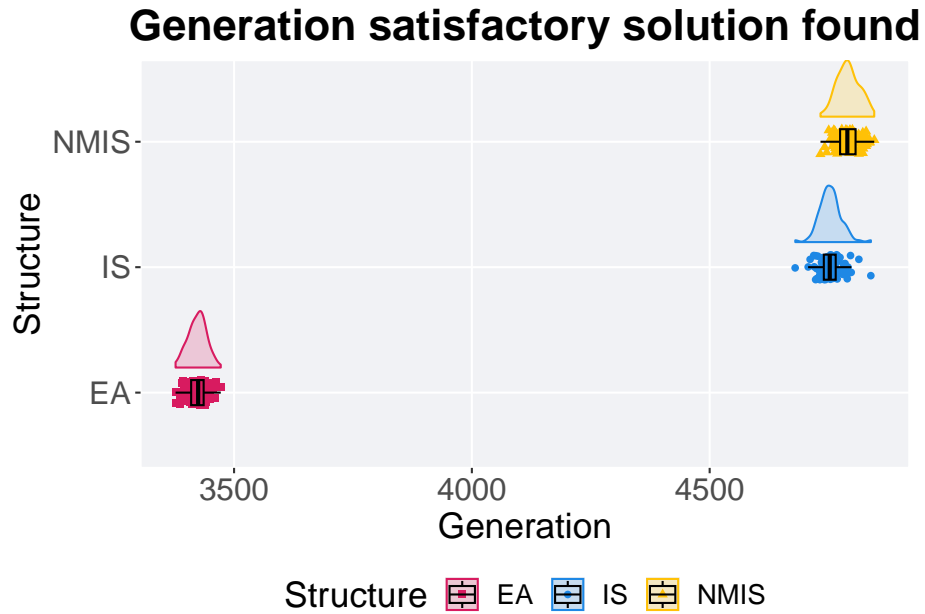
```



### 2.2.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(base_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TRUNCATION') %>%
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Generation"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Generation satisfactory solution found') +
  p_theme + coord_flip()
```



### 2.2.3 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(base_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TRUE')
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0  3377  3424.  3423.  3472  26.2
## 2 IS          100     0  4680  4752.  4754.  4839   25
## 3 NMIS        100     0  4733  4790.  4791.  4846  32.5
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)

##
## Kruskal-Wallis rank sum test
##
## data: Generations by Structure
## Kruskal-Wallis chi-squared = 237.99, df = 2, p-value < 2.2e-16

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: ssf$Generations and ssf$Structure
##
##      EA      IS
## IS  <2e-16 -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 2.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the exploitation rate diagnostic.

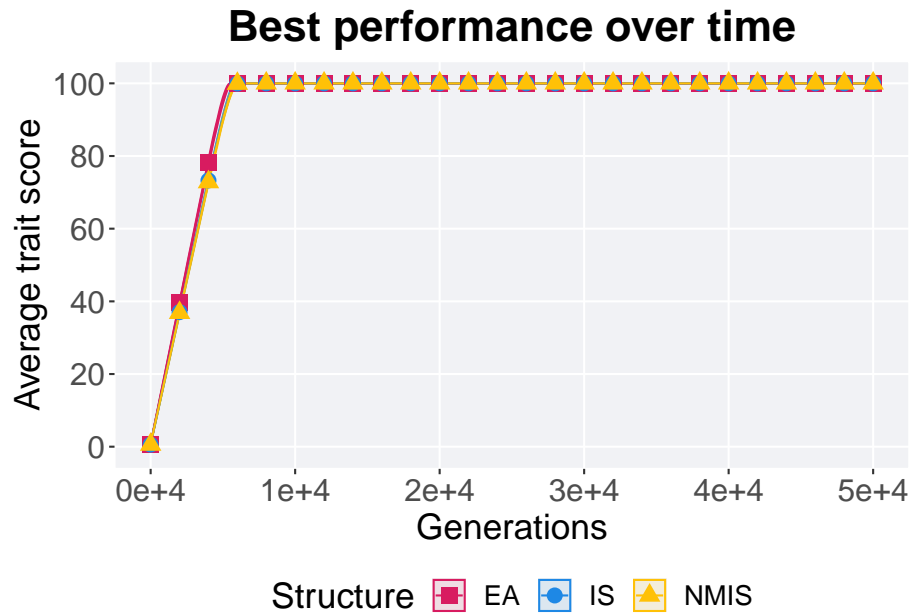
### 2.3.1 Performance over time

```
lines = filter(base_over_time, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TOURNAMENT')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.0)
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
```

```

labels=c("0", "20", "40", "60", "80", "100")
) +
scale_x_continuous(
  name="Generations",
  limits=c(0, 50000),
  breaks=c(0, 10000, 20000, 30000, 40000, 50000),
  labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
) +
scale_shape_manual(values=SHAPE) +
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle("Best performance over time") +
p_theme

```



### 2.3.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```

filter(base_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TOURNAMENT')
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +

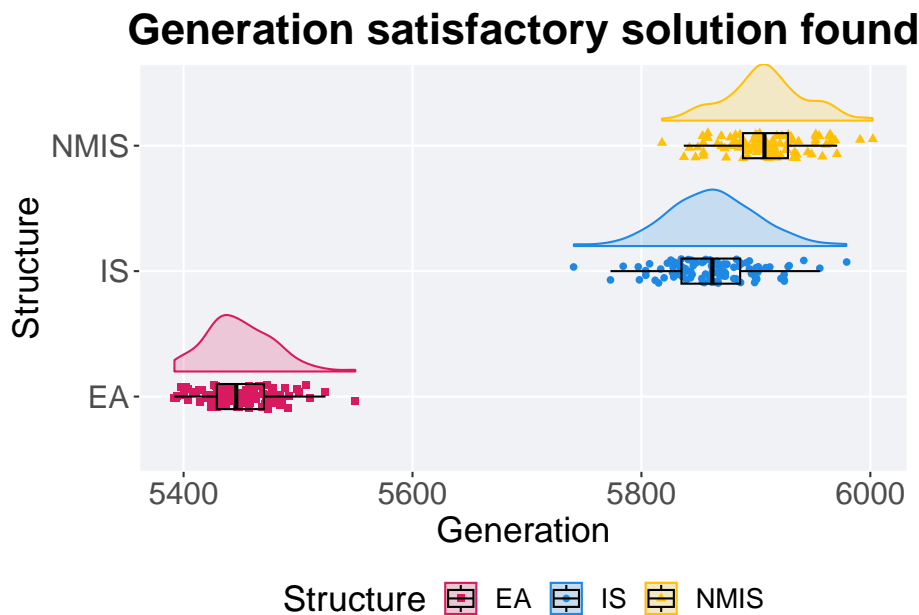
```



```

scale_y_continuous(
  name="Generation"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Generation satisfactory solution found')+
p_theme + coord_flip()

```



### 2.3.3 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(base_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TOURNAMENT' &
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),

```

```

max = max(Generations, na.rm = TRUE),
IQR = IQR(Generations, na.rm = TRUE)
)

```

```

## # A tibble: 3 x 8
##   Structure count na_cnt   min median  mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0  5392  5446  5449.  5550  41.2
## 2 IS          100     0  5741  5862  5862.  5979  51.2
## 3 NMIS        100     0  5818  5908. 5909.  6002  39.2

```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```

kruskal.test(Generations ~ Structure, data = ssf)

```

```

##
##  Kruskal-Wallis rank sum test
##
## data:  Generations by Structure
## Kruskal-Wallis chi-squared = 226.27, df = 2, p-value < 2.2e-16

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```

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

```

```

##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  ssf$Generations and ssf$Structure
##
##      EA      IS
## IS   < 2e-16 -
## NMIS < 2e-16 1.1e-14
##
## P value adjustment method: bonferroni

```

## 2.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the exploitation rate diagnostic.

### 2.4.1 Performance over time

```

lines = filter(base_over_time, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme`
group_by(Structure, Generations) %>%
dplyr::summarise(

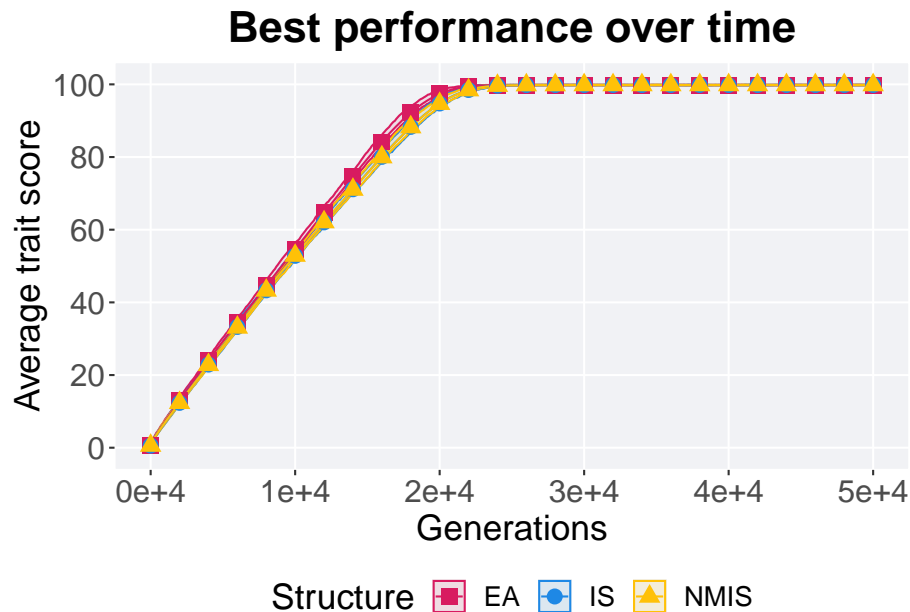
```

```

    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")

  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Best performance over time") +
  p_theme

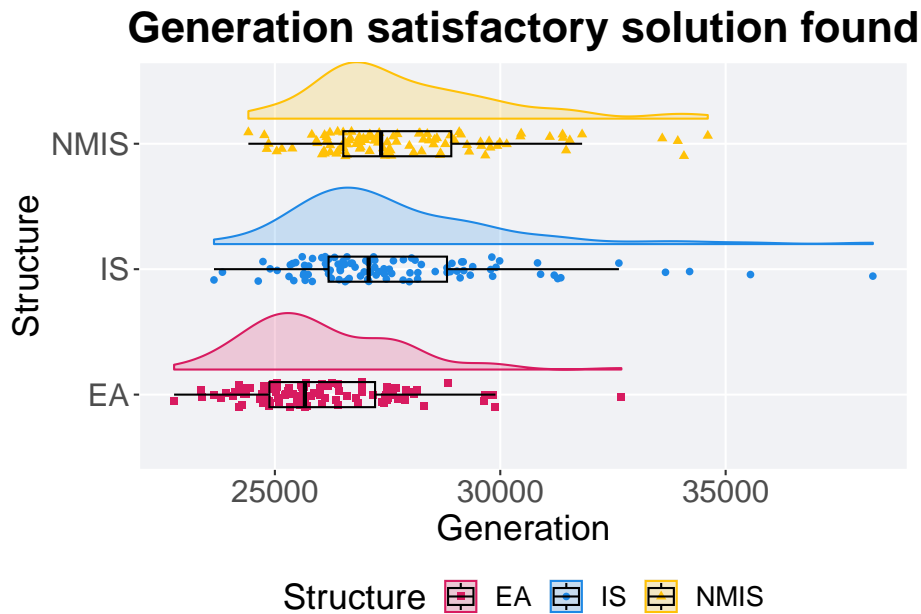
```



### 2.4.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(base_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'LEXICASE')
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Generation"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Generation satisfactory solution found') +
  p_theme + coord_flip()
```



### 2.4.3 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(base_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'LEXICASE' & Ge
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0 22764 25666. 26026. 32687 2344
## 2 IS          100     0 23649 27080. 27635. 38266 2628.
## 3 NMIS        100     0 24412 27358. 27906. 34604 2396.
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```
##  
## Kruskal-Wallis rank sum test  
##  
## data: Generations by Structure  
## Kruskal-Wallis chi-squared = 52.814, df = 2, p-value = 3.401e-12
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

```
##  
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction  
##  
## data: ssf$Generations and ssf$Structure  
##  
##      EA      IS  
## IS  3.0e-08 -  
## NMIS 2.2e-11 0.24  
##  
## P value adjustment method: bonferroni
```

## Chapter 3

# BASE: Ordered exploitation results

Here we present the results for **best performances** found by each selection scheme replicate on the ordered exploitation diagnostic with our base configurations. Best performance found refers to the largest average trait score found in a given population. Note that performance values fall between 0.0 and 100.0. For our base configuration, we execute migrations every 500 generations and there are 4 islands in a ring topology. When migrations occur, we swap two individuals (same position on each island) and guarantee that no solution can return to the same island.

### 3.1 Analysis dependencies

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

### 3.2 Truncation selection

Here we analyze how the different population structures affect truncation selection (size 8) on the ordered exploitation diagnostic.

#### 3.2.1 Performance over time

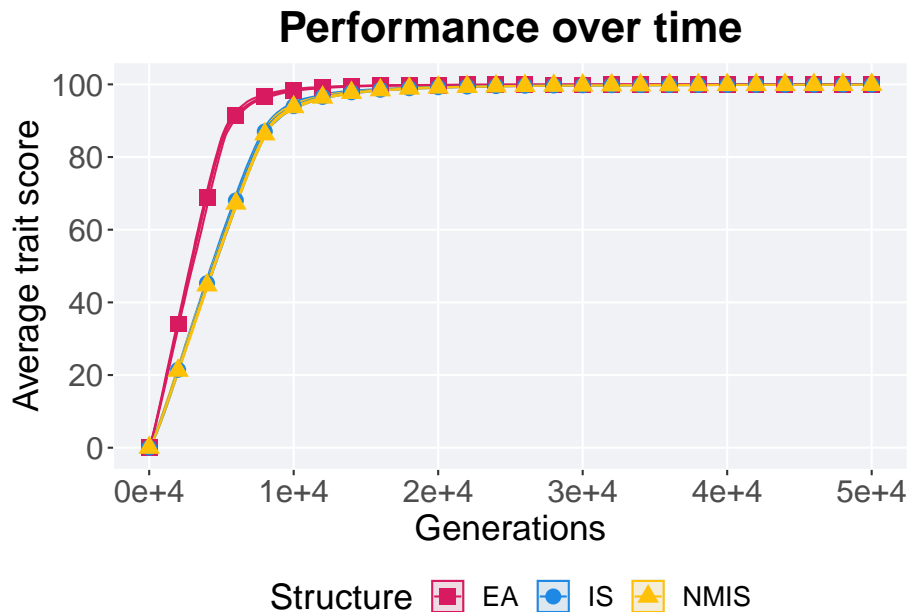
```
lines = filter(base_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TRU
  group_by(Structure, Generations) %>%
```

```

dplyr::summarise(
  min = min(pop_fit_max) / DIMENSIONALITY,
  mean = mean(pop_fit_max) / DIMENSIONALITY,
  max = max(pop_fit_max) / DIMENSIONALITY
)
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0,
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme

```

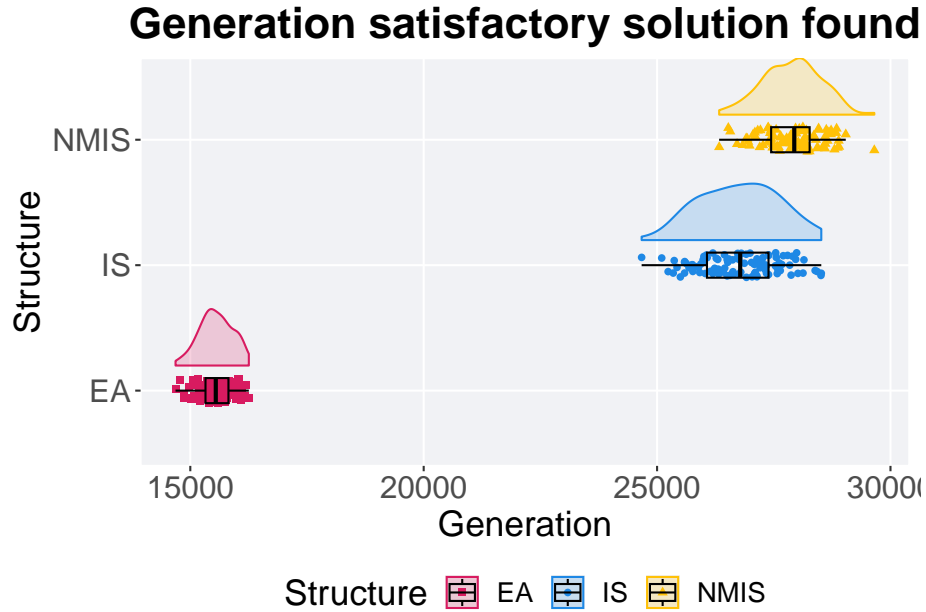




### 3.2.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(base_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TRUNCATION') %>%
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Generation"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Generation satisfactory solution found') +
  p_theme + coord_flip()
```



#### 3.2.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(base_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == '1')
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt  min median  mean  max  IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0 14684 15546 15554. 16254 492.
## 2 IS          100     0 24669 26780. 26767. 28518 1318.
## 3 NMIS        100     0 26330 27939 27888. 29654 825.
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)

##
## Kruskal-Wallis rank sum test
##
## data: Generations by Structure
## Kruskal-Wallis chi-squared = 231.88, df = 2, p-value < 2.2e-16

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: ssf$Generations and ssf$Structure
##
##      EA      IS
## IS  <2e-16 -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

### 3.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the ordered exploitation diagnostic.

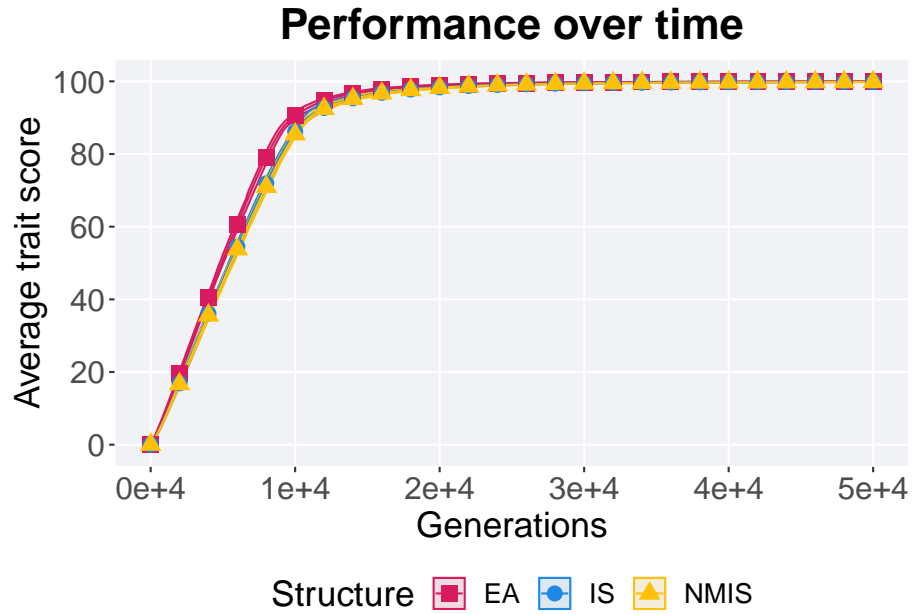
#### 3.3.1 Performance over time

```
lines = filter(base_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TOURNAMENT')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.0)
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
```

```

labels=c("0", "20", "40", "60", "80", "100")
) +
scale_x_continuous(
  name="Generations",
  limits=c(0, 50000),
  breaks=c(0, 10000, 20000, 30000, 40000, 50000),
  labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
) +
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle("Performance over time") +
p_theme

```



### 3.3.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```

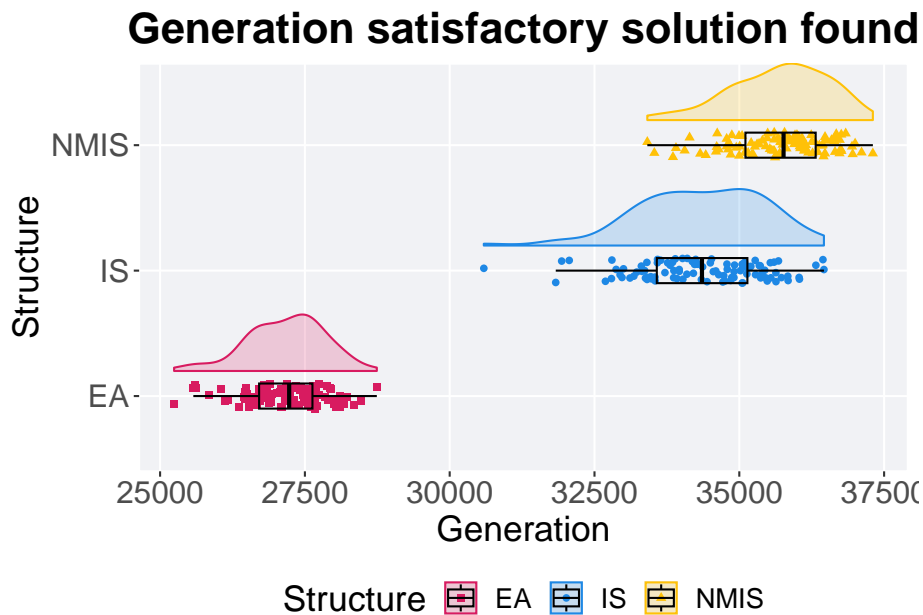
filter(base_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TOURNAMENT')
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +

```

```

scale_y_continuous(
  name="Generation"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Generation satisfactory solution found')+
p_theme + coord_flip()

```



### 3.3.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(base_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TOURNAMENT')
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),

```

```

max = max(Generations, na.rm = TRUE),
IQR = IQR(Generations, na.rm = TRUE)
)

```

```

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0 25242 27228. 27172. 28742  921.
## 2 IS          100     0 30589 34356. 34349. 36461 1564.
## 3 NMIS        100     0 33412 35764  35692. 37306 1213

```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```

##
##   Kruskal-Wallis rank sum test
##
## data:   Generations by Structure
## Kruskal-Wallis chi-squared = 229.49, df = 2, p-value < 2.2e-16

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```

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

```

```

##
##   Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:   ssf$Generations and ssf$Structure
##
##           EA           IS
## IS    < 2e-16 -
## NMIS < 2e-16 2.8e-16
##
## P value adjustment method: bonferroni

```

## 3.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the ordered exploitation diagnostic.

### 3.4.1 Performance over time

```

lines = filter(base_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nSche
group_by(Structure, Generations) %>%
dplyr::summarise(

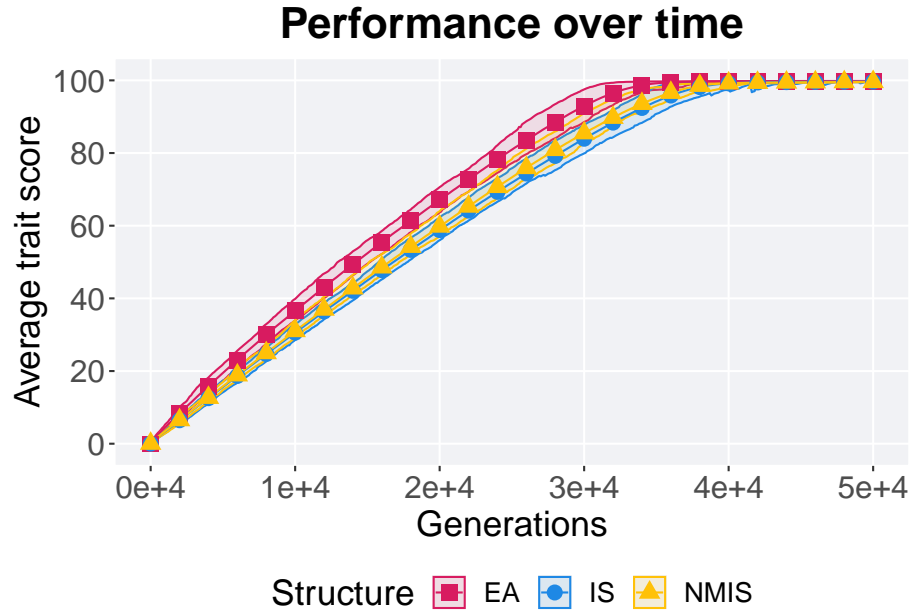
```

```

    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")

  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme

```

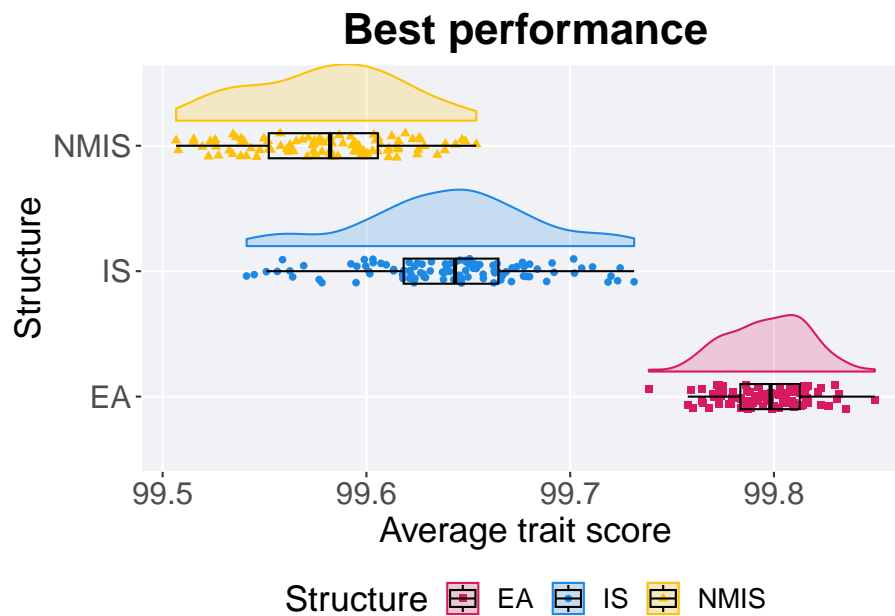


### 3.4.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(base_best, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEXIC
ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Stru
geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
scale_y_continuous(
  name="Average trait score"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Best performance')+
p_theme + coord_flip()
```





### 3.4.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(base_best, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LE
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VA)),
    min = min(VA, na.rm = TRUE) / DIMENSIONALITY,
    median = median(VA, na.rm = TRUE) / DIMENSIONALITY,
    mean = mean(VA, na.rm = TRUE) / DIMENSIONALITY,
    max = max(VA, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VA, na.rm = TRUE) / DIMENSIONALITY
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0  99.7  99.8  99.8  99.8 0.0291
## 2 IS          100     0  99.5  99.6  99.6  99.7 0.0465
## 3 NMIS        100     0  99.5  99.6  99.6  99.7 0.0535
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VA ~ Structure, data = performance)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  VA by Structure
## Kruskal-Wallis chi-squared = 235.04, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

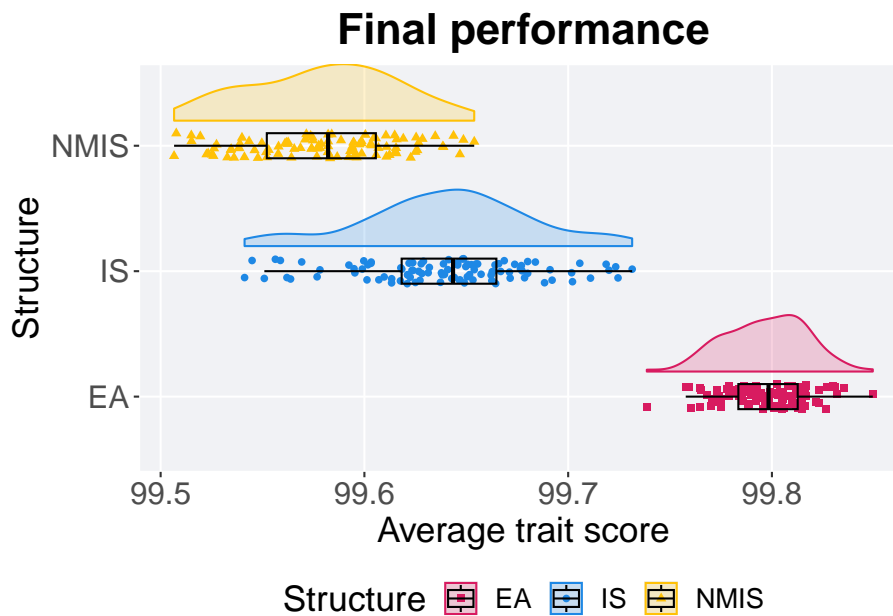
```
pairwise.wilcox.test(x = performance$VA, g = performance$Structure, p.adjust.method =
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VA and performance$Structure
##
##      EA      IS
## IS   <2e-16 -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

### 3.4.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(base_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'I
ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fi
geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
scale_y_continuous(
  name="Average trait score"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Final performance')+
p_theme + coord_flip()
```



### 3.4.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(base_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` =
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_fit_max)),
    min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA         100     0  99.7  99.8  99.8  99.8 0.0291
## 2 IS         100     0  99.5  99.6  99.6  99.7 0.0465
## 3 NMIS       100     0  99.5  99.6  99.6  99.7 0.0535
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 235.02, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.m = "none",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$pop_fit_max and performance$Structure
##
##      EA      IS
## IS    <2e-16 -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

### 3.4.4 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

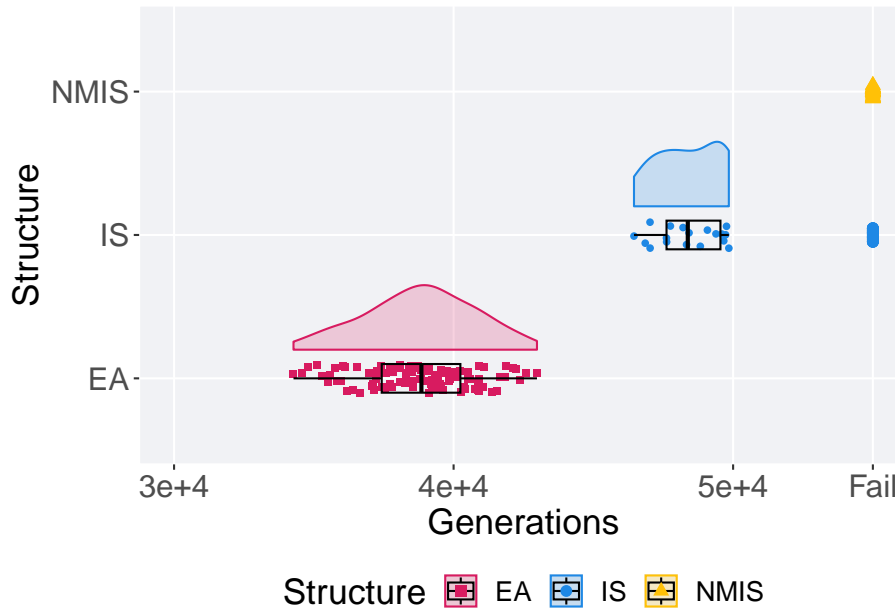
```
lex_fail = filter(base_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEXICAL')
lex_fail$Generations = 55000
lex_fail$Structure <- factor(lex_fail$Structure, levels = MODEL)

filter(base_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEXICAL') %>%
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = .5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  geom_point(data = lex_fail, aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Generations",
    limits=c(30000, 55000),
    breaks=c(30000, 40000, 50000, 55000),
    labels=c("3e+4", "4e+4", "5e+4", "Fail")
  ) +
  scale_x_discrete(
```

```

  name="Structure"
) +
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
p_theme + coord_flip()

```



#### 3.4.4.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(base_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEXICASE' &
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )

```

## # A tibble: 2 x 8

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

```
## 1 EA          100      0 34272 38848 38795. 42983 2814
## 2 IS          18      0 46454 48378. 48402. 49847 1929
```

Kruskal-Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  Generations by Structure
## Kruskal-Wallis chi-squared = 45.378, df = 1, p-value = 1.624e-11
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  ssf$Generations and ssf$Structure
##
##      EA
## IS 8.3e-12
##
## P value adjustment method: bonferroni
```

## Chapter 4

# BASE: Contradictory objectives results

Here we present the results for the **satisfactory trait coverage** and **activation gene coverage** generated by each selection scheme replicate on the contradictory objectives diagnostic with our base configurations. Note both of these values are gathered at the population-level. Activation gene coverage refers to the count of unique activation genes in a given population; this gives us a range of integers between 0 and 100. Satisfactory trait coverage refers to the count of unique satisfied traits in a given population; this gives us a range of integers between 0 and 100. For our base configuration, we execute migrations every 500 generations and there are 4 islands in a ring topology. When migrations occur, two individuals are swapped (same position on each island) and guarantee that no solution can return to its original island.

### 4.1 Analysis dependencies

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

### 4.2 Truncation selection

Here we analyze how the different population structures affect truncation selection (size 8) on the contradictory objectives diagnostic.

## 4.2.1 Satisfactory trait coverage

Satisfactory trait coverage analysis.

### 4.2.1.1 Coverage over time

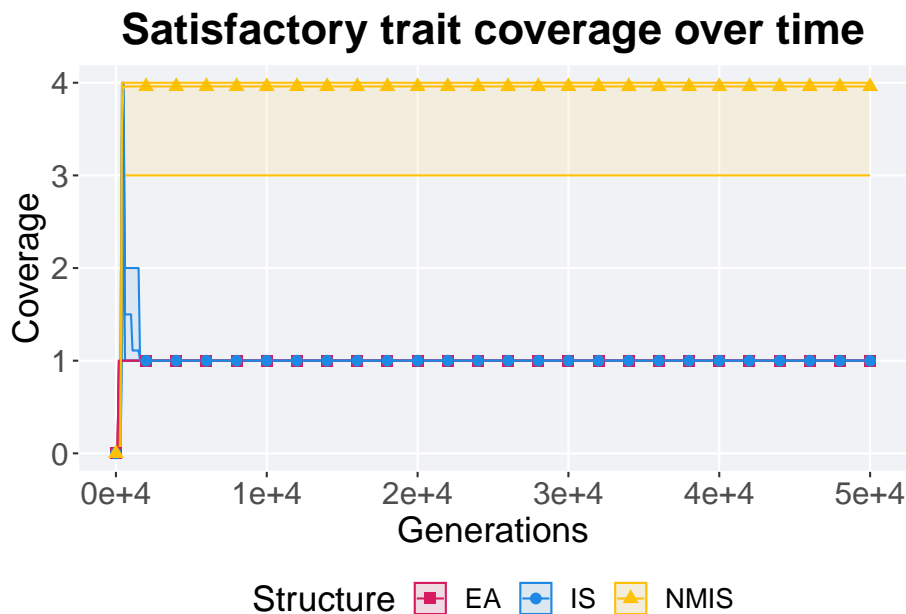
Satisfactory trait coverage over time.

```
lines = filter(base_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\n`
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_sat_cov),
    mean = mean(pop_sat_cov),
    max = max(pop_sat_cov)
  )
```

## `summarise()` has grouped output by 'Structure'. You can override using the  
## `.groups` argument.

```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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
```

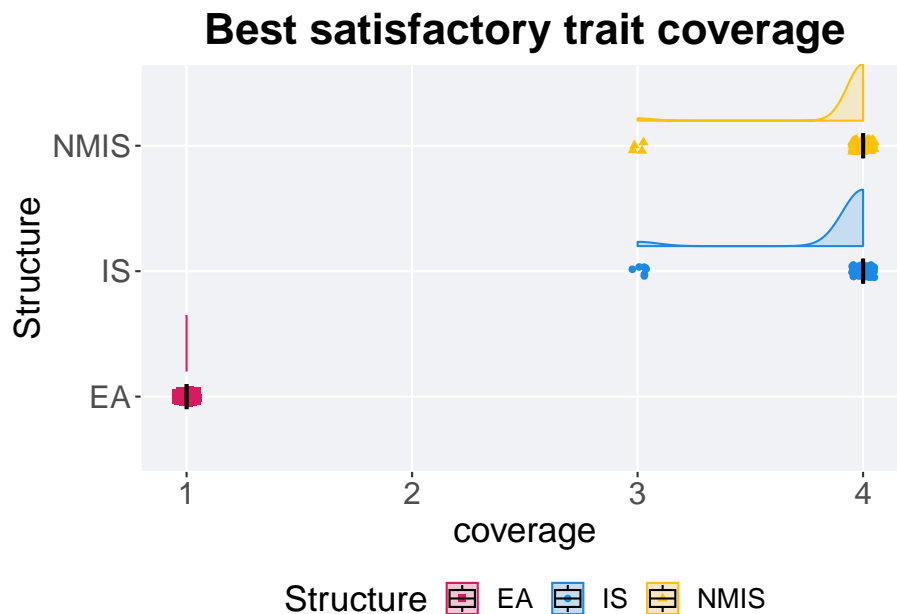




#### 4.2.1.2 Best coverage throughout

Best satisfactory trait coverage throughout 50,000 generations.

```
### best satisfactory trait coverage throughout
filter(base_best, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` == 'TRUNCATION')
ggplot(., aes(x = Structure, y = VAL, color = Structure, fill = Structure, shape = Structure))
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory trait coverage') +
  p_theme + coord_flip()
```



#### 4.2.1.2.1 Stats

Summary statistics for the best satisfactory trait coverage.

```
### best
coverage = filter(base_best, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nSc
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE),
    median = median(VAL, na.rm = TRUE),
    mean = mean(VAL, na.rm = TRUE),
    max = max(VAL, na.rm = TRUE),
    IQR = IQR(VAL, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     3     4   3.93     4     0
## 3 NMIS        100     0     3     4   3.96     4     0
```

Kruskal–Wallis test provides evidence of difference among satisfactory trait

coverage.

```
kruskal.test(VAL ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: VAL by Structure
## Kruskal-Wallis chi-squared = 279.71, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage.

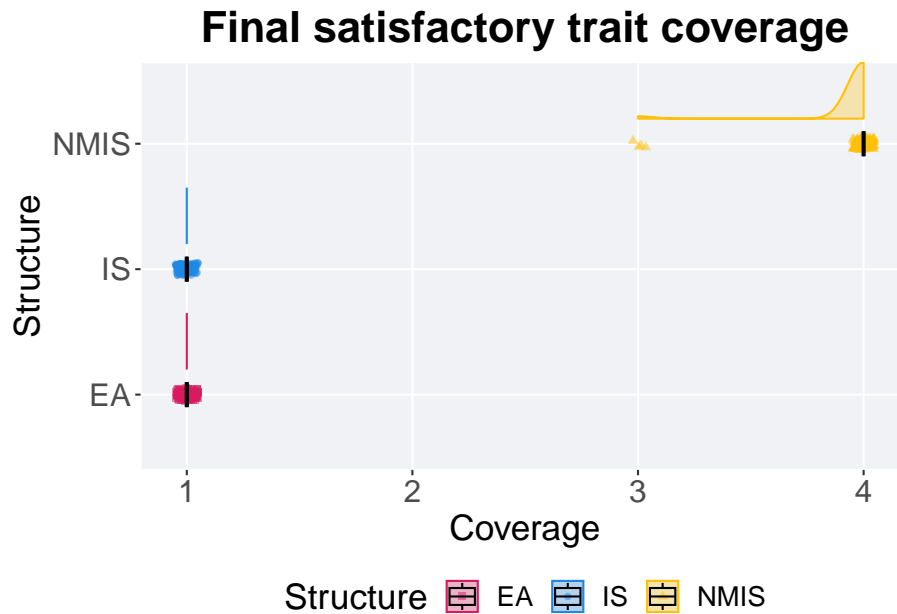
```
pairwise.wilcox.test(x = coverage$VAL, g = coverage$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$VAL and coverage$Structure
##
##      EA      IS
## IS  <2e-16 -
## NMIS <2e-16 0.53
##
## P value adjustment method: bonferroni
```

#### 4.2.1.3 End of 50,000 generations

Satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
filter(base_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'TRUNCAT
  ggplot(., aes(x = Structure, y = pop_sat_cov, color = Structure, fill = Structure, shape = Stru
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE)+
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final satisfactory trait coverage')+
  p_theme + coord_flip()
```



#### 4.2.1.3.1 Stats

Summary statistics for satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
coverage = filter(base_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_sat_cov)),
    min = min(pop_sat_cov, na.rm = TRUE),
    median = median(pop_sat_cov, na.rm = TRUE),
    mean = mean(pop_sat_cov, na.rm = TRUE),
    max = max(pop_sat_cov, na.rm = TRUE),
    IQR = IQR(pop_sat_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     3     4    3.96     4     0
```

Kruskal-Wallis test provides evidence of difference among satisfactory trait coverage in the population at the end of 50,000 generations.

```
kruskal.test(pop_sat_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_sat_cov by Structure
## Kruskal-Wallis chi-squared = 297.1, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage in the population at the end of 50,000 generations.

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

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_sat_cov and coverage$Structure
##
##      EA      IS
## IS   1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

### 4.2.2 Activation gene coverage

Activation gene coverage analysis.

#### 4.2.2.1 Coverage over time

Activation gene coverage over time.

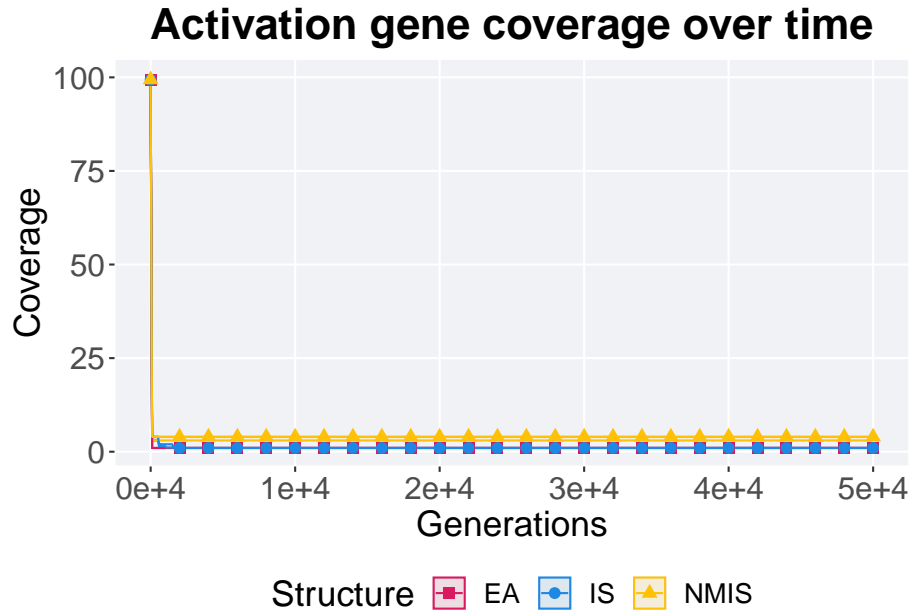
```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` ==
              group_by(Structure, Generations) %>%
              dplyr::summarise(
                min = min(pop_act_cov),
                mean = mean(pop_act_cov),
                max = max(pop_act_cov)
              )
```

```
## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.
```

```

ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
    scale_y_continuous(
      name="Coverage"
    ) +
    scale_x_continuous(
      name="Generations",
      limits=c(0, 50000),
      breaks=c(0, 10000, 20000, 30000, 40000, 50000),
      labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
    ) +
    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

```



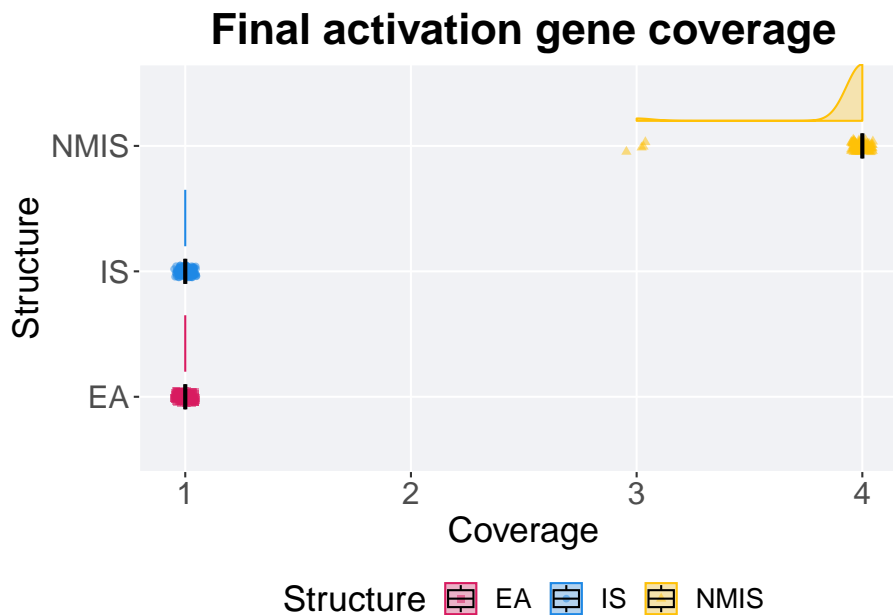
#### 4.2.2.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```

### end of run
filter(base_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` == 'TRUNCAT
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Stru
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE)+
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage')+
  p_theme + coord_flip()

```



#### 4.2.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(base_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme`
coverage %>%
  group_by(Structure) %>%

```

```
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(pop_act_cov)),
  min = min(pop_act_cov, na.rm = TRUE),
  median = median(pop_act_cov, na.rm = TRUE),
  mean = mean(pop_act_cov, na.rm = TRUE),
  max = max(pop_act_cov, na.rm = TRUE),
  IQR = IQR(pop_act_cov, na.rm = TRUE)
)
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     3     4    3.96     4     0
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
##   Kruskal-Wallis rank sum test
##
## data:  pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 297.1, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```
pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method =
  paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
##   Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS    1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```



## 4.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the contradictory objectives diagnostic.

### 4.3.1 Satisfactory trait coverage

Satisfactory trait coverage analysis.

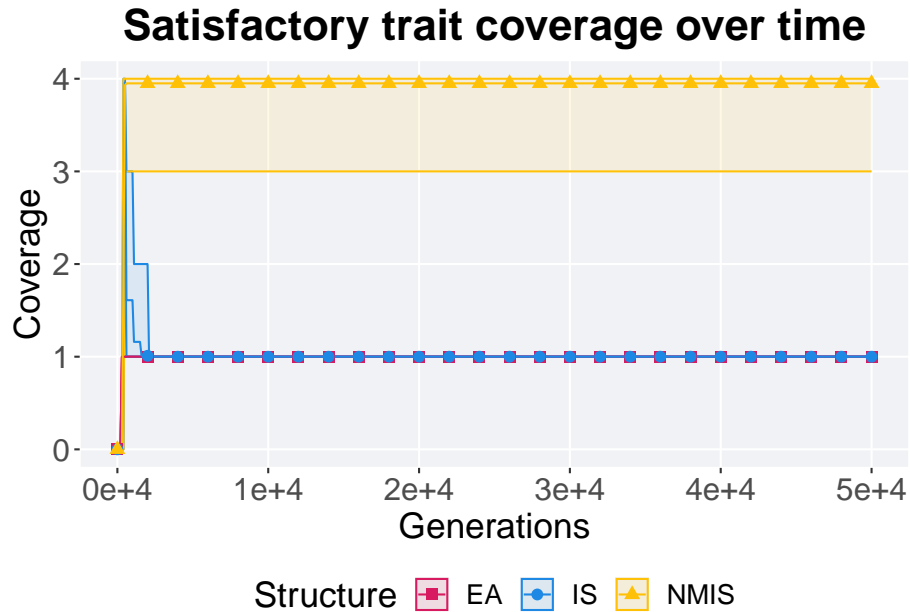
#### 4.3.1.1 Coverage over time

Satisfactory trait coverage over time.

```
lines = filter(base_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` ==
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_sat_cov),
    mean = mean(pop_sat_cov),
    max = max(pop_sat_cov)
  )

## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.

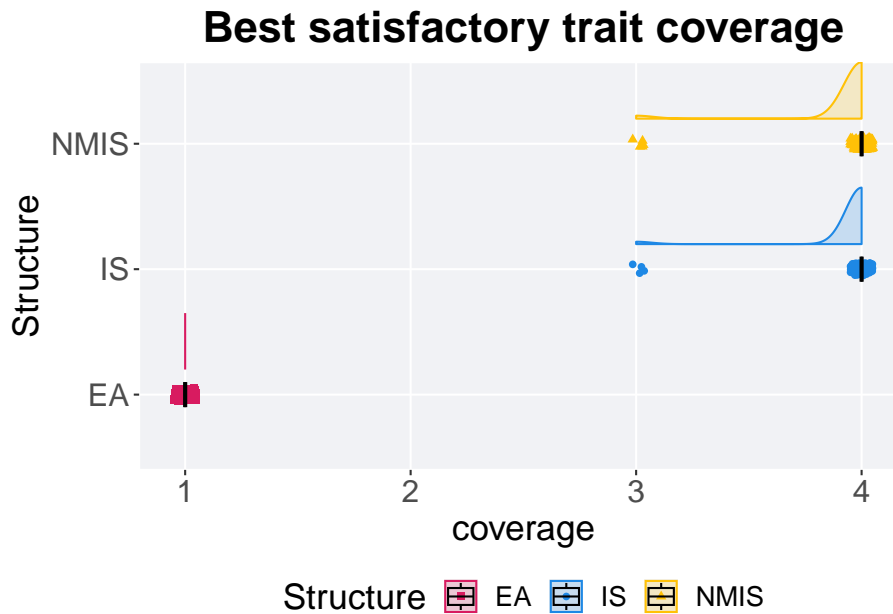
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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
```



#### 4.3.1.2 Best coverage throughout

Best satisfactory trait coverage throughout 50,000 generations.

```
### best satisfactory trait coverage throughout
filter(base_best, Diagnostic == 'CONTRADICTION OBJECTIVES' & `Selection\nScheme` == 'T
ggplot(., aes(x = Structure, y = VAL, color = Structure, fill = Structure, shape = S
geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0
geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
scale_y_continuous(
  name="coverage"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Best satisfactory trait coverage')+
p_theme + coord_flip()
```



#### 4.3.1.2.1 Stats

Summary statistics for the best satisfactory trait coverage.

```
### best
coverage = filter(base_best, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` == 'T
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE),
    median = median(VAL, na.rm = TRUE),
    mean = mean(VAL, na.rm = TRUE),
    max = max(VAL, na.rm = TRUE),
    IQR = IQR(VAL, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA         100     0     1     1     1     1     0
## 2 IS         100     0     3     4    3.96     4     0
## 3 NMIS        100     0     3     4    3.95     4     0
```

Kruskal–Wallis test provides evidence of difference among satisfactory trait

coverage.

```
kruskal.test(VAL ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: VAL by Structure
## Kruskal-Wallis chi-squared = 282.81, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage.

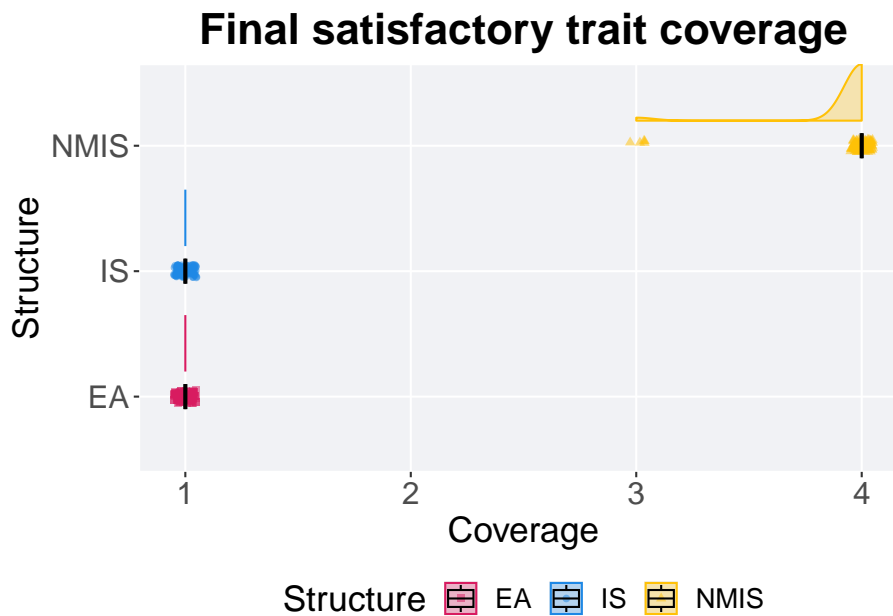
```
pairwise.wilcox.test(x = coverage$VAL, g = coverage$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$VAL and coverage$Structure
##
##      EA      IS
## IS   <2e-16 -
## NMIS <2e-16 1
##
## P value adjustment method: bonferroni
```

#### 4.3.1.3 End of 50,000 generations

Satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
filter(base_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'Final')
ggplot(., aes(x = Structure, y = pop_sat_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final satisfactory trait coverage') +
  p_theme + coord_flip()
```



#### 4.3.1.3.1 Stats

Summary statistics for satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
coverage = filter(base_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme`
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_sat_cov)),
    min = min(pop_sat_cov, na.rm = TRUE),
    median = median(pop_sat_cov, na.rm = TRUE),
    mean = mean(pop_sat_cov, na.rm = TRUE),
    max = max(pop_sat_cov, na.rm = TRUE),
    IQR = IQR(pop_sat_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt  min median  mean  max  IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     3     4    3.95     4     0
```

Kruskal-Wallis test provides evidence of difference among satisfactory trait coverage in the population at the end of 50,000 generations.

```
kruskal.test(pop_sat_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_sat_cov by Structure
## Kruskal-Wallis chi-squared = 296.65, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage in the population at the end of 50,000 generations.

```
pairwise.wilcox.test(x = coverage$pop_sat_cov, g = coverage$Structure, p.adjust.method = 'g',
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_sat_cov and coverage$Structure
##
##      EA      IS
## IS   1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

### 4.3.2 Activation gene coverage

Activation gene coverage analysis.

#### 4.3.2.1 Coverage over time

Activation gene coverage over time.

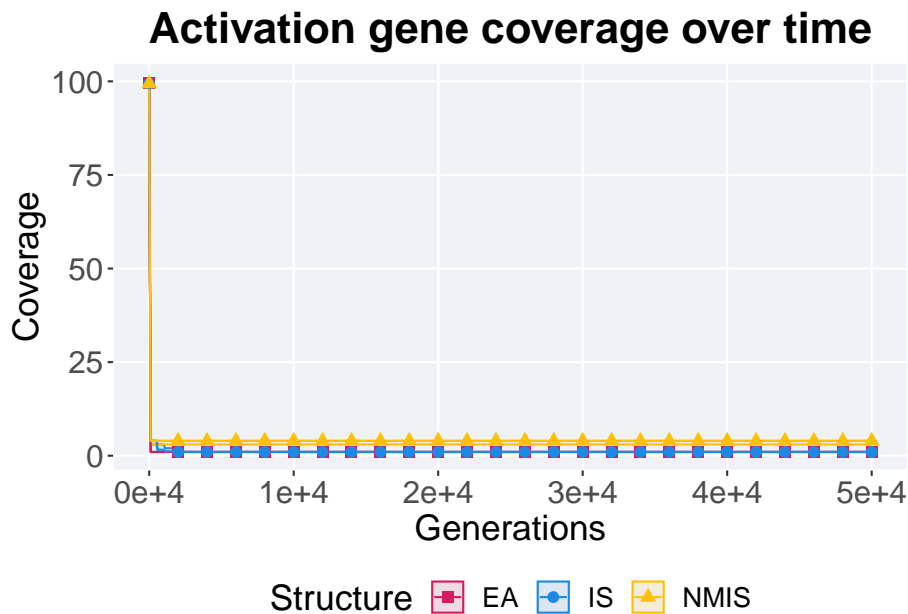
```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection` == 'n')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )
```

```
## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.
```

```

ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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

```



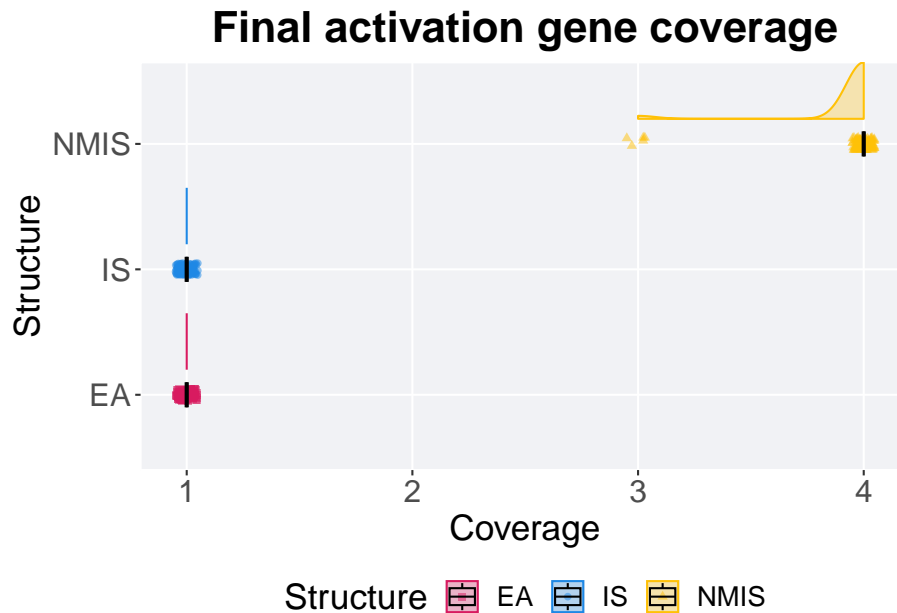
#### 4.3.2.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```

### end of run
filter(base_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` =
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, sl
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha =
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme + coord_flip()

```



#### 4.3.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(base_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection
coverage %>%
  group_by(Structure) %>%

```



```
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(pop_act_cov)),
  min = min(pop_act_cov, na.rm = TRUE),
  median = median(pop_act_cov, na.rm = TRUE),
  mean = mean(pop_act_cov, na.rm = TRUE),
  max = max(pop_act_cov, na.rm = TRUE),
  IQR = IQR(pop_act_cov, na.rm = TRUE)
)
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     3     4   3.95     4     0
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 296.65, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

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

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS    1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 4.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the contradictory objectives diagnostic.

### 4.4.1 Satisfactory trait coverage

Satisfactory trait coverage analysis.

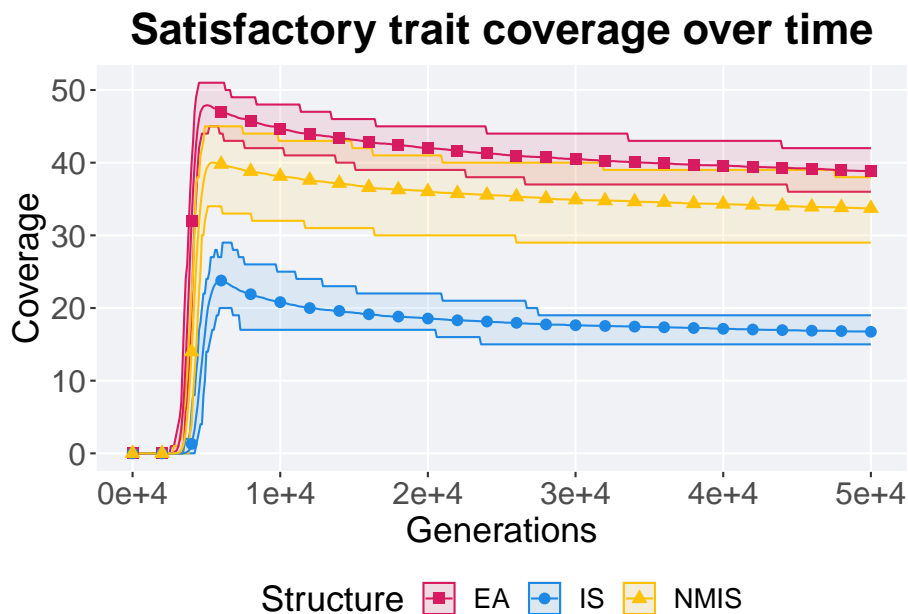
#### 4.4.1.1 Coverage over time

Satisfactory trait coverage over time.

```
lines = filter(base_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\n
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_sat_cov),
    mean = mean(pop_sat_cov),
    max = max(pop_sat_cov)
  )

## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.

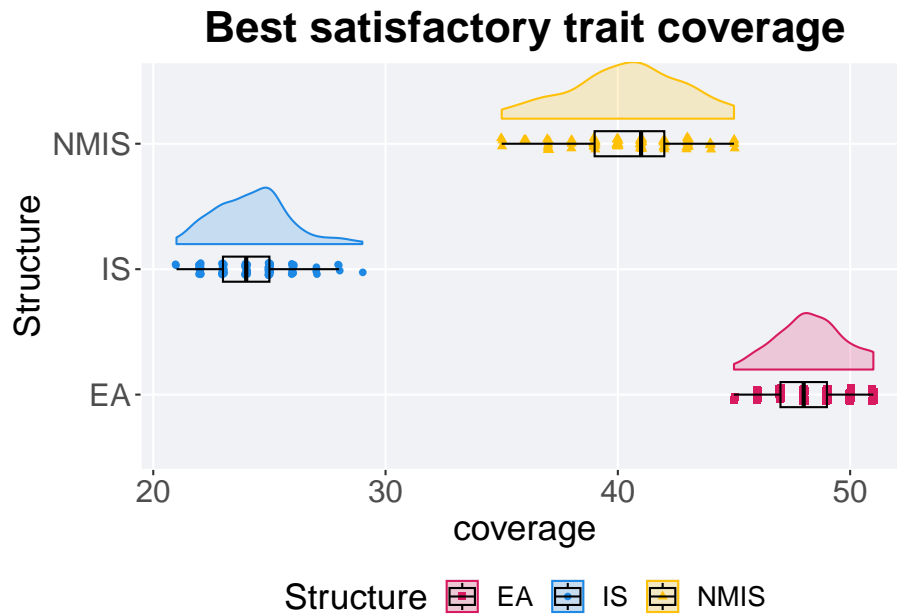
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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
```



#### 4.4.1.2 Best coverage throughout

Best satisfactory trait coverage throughout 50,000 generations.

```
### best satisfactory trait coverage throughout
filter(base_best, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE' &
  ggplot(., aes(x = Structure, y = VAL, color = Structure, fill = Structure, shape = Structure))
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory trait coverage') +
  p_theme + coord_flip()
```



#### 4.4.1.2.1 Stats

Summary statistics for the best satisfactory trait coverage.

```
### best
coverage = filter(base_best, Diagnostic == 'CONTRADICTION OBJECTIVES' & `Selection\nSc
coverage$Structure = factor(coverage$Structure, levels=c('EA','NMIS','IS'))
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE),
    median = median(VAL, na.rm = TRUE),
    mean = mean(VAL, na.rm = TRUE),
    max = max(VAL, na.rm = TRUE),
    IQR = IQR(VAL, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int>  <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0    45    48  48.3    51     2
## 2 NMIS        100     0    35    41  40.4    45     3
## 3 IS          100     0    21    24  24.2    29     2
```

Kruskal-Wallis test provides evidence of difference among satisfactory trait coverage.

```
kruskal.test(VAL ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: VAL by Structure
## Kruskal-Wallis chi-squared = 266.69, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage.

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

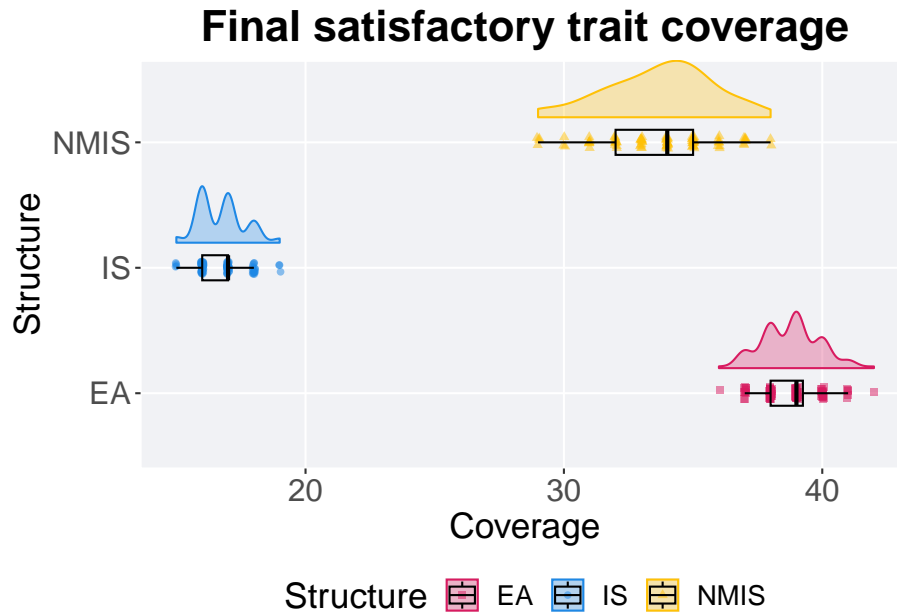
```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$VAL and coverage$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

#### 4.4.1.3 End of 50,000 generations

Satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
filter(base_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE')
ggplot(., aes(x = Structure, y = pop_sat_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final satisfactory trait coverage') +
```

```
p_theme + coord_flip()
```



#### 4.4.1.3.1 Stats

Summary statistics for satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
coverage = filter(base_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection` == 'EA')
coverage$Structure = factor(coverage$Structure, levels=c('EA', 'NMIS', 'IS'))
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_sat_cov)),
    min = min(pop_sat_cov, na.rm = TRUE),
    median = median(pop_sat_cov, na.rm = TRUE),
    mean = mean(pop_sat_cov, na.rm = TRUE),
    max = max(pop_sat_cov, na.rm = TRUE),
    IQR = IQR(pop_sat_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0    36    39  38.8    42  1.25
```

```
## 2 NMIS      100      0    29    34 33.7    38 3
## 3 IS        100      0    15    17 16.7    19 1
```

Kruskal–Wallis test provides evidence of difference among satisfactory trait coverage in the population at the end of 50,000 generations.

```
kruskal.test(pop_sat_cov ~ Structure, data = coverage)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  pop_sat_cov by Structure
## Kruskal-Wallis chi-squared = 265.34, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage in the population at the end of 50,000 generations.

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

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_sat_cov and coverage$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 4.4.2 Activation gene coverage

Activation gene coverage analysis.

### 4.4.2.1 Coverage over time

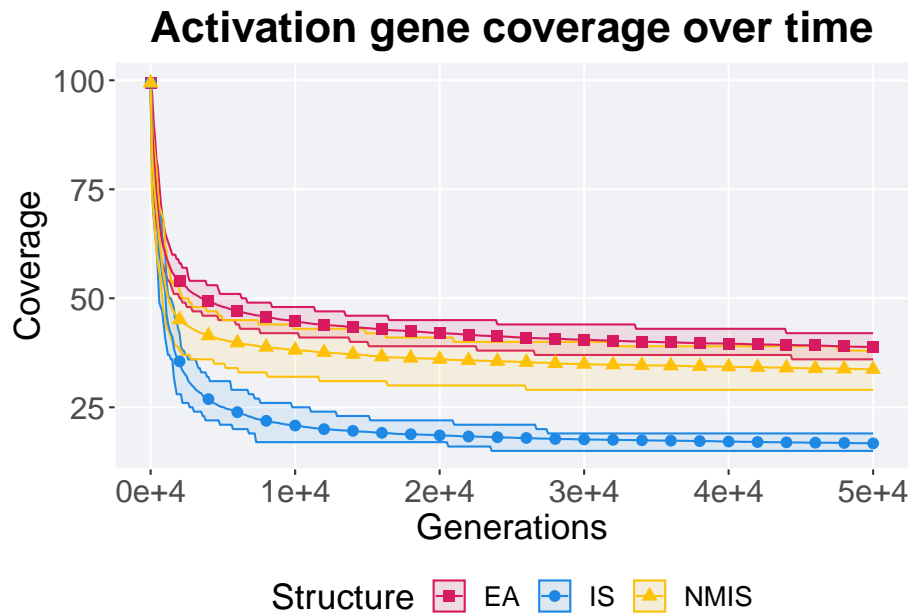
Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` ==
               group_by(Structure, Generations) %>%
               dplyr::summarise(
                 min = min(pop_act_cov),
                 mean = mean(pop_act_cov),
                 max = max(pop_act_cov)
               )
```

## `summarise()` has grouped output by 'Structure'. You can override using the

## ``groups`` argument.

```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
    scale_y_continuous(
      name="Coverage"
    ) +
    scale_x_continuous(
      name="Generations",
      limits=c(0, 50000),
      breaks=c(0, 10000, 20000, 30000, 40000, 50000),
      labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
    ) +
    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
```



#### 4.4.2.2 End of 50,000 generations

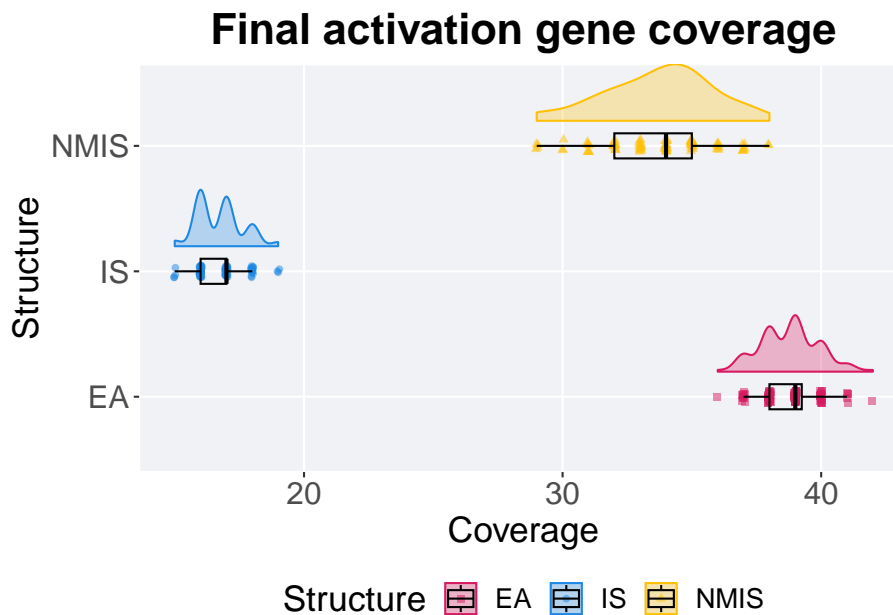
Activation gene coverage in the population at the end of 50,000 generations.



```

### end of run
filter(base_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE')
ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme + coord_flip()

```



#### 4.4.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(base_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE')
coverage$Structure = factor(coverage$Structure, levels=c('EA', 'NMIS', 'IS'))
coverage %>%

```

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

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA         100     0    36    39  38.8   42  1.25
## 2 NMIS       100     0    29    34  33.7   38   3
## 3 IS         100     0    15    17  16.7   19   1
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 265.34, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```
pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method =
  paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_act_cov and coverage$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## Chapter 5

# BASE: Multi-path exploration results

Here we present the results for the **best performances** and **activation gene coverage** generated by each selection scheme replicate on the multi-path exploration diagnostic. Best performance found refers to the largest average trait score found in a given population. Note that activation gene coverage values are gathered at the population-level. Activation gene coverage refers to the count of unique activation genes in a given population; this gives us a range of integers between 0 and 100.

### 5.1 Analysis dependencies

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

### 5.2 Truncation selection

Here we analyze how the different population structures affect truncation selection (size 8) on the contradictory objectives diagnostic.

#### 5.2.1 Performance

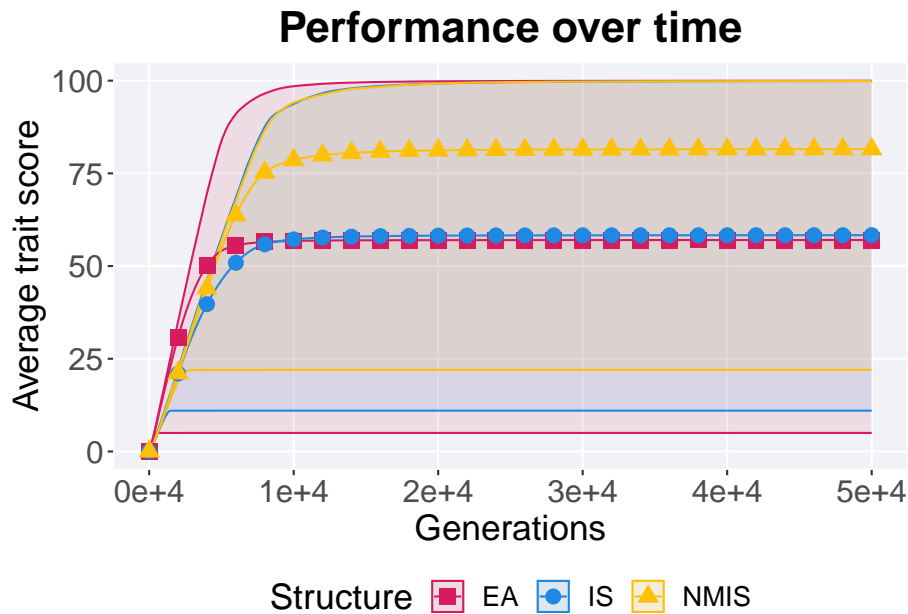
##### 5.2.1.1 Performance over time

```

lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nSch
group_by(Structure, Generations) %>%
dplyr::summarise(
  min = min(pop_fit_max) / DIMENSIONALITY,
  mean = mean(pop_fit_max) / DIMENSIONALITY,
  max = max(pop_fit_max) / DIMENSIONALITY
)
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0,
scale_y_continuous(
  name="Average trait score"
) +
scale_x_continuous(
  name="Generations",
  limits=c(0, 50000),
  breaks=c(0, 10000, 20000, 30000, 40000, 50000),
  labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")

) +
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle("Performance over time") +
p_theme

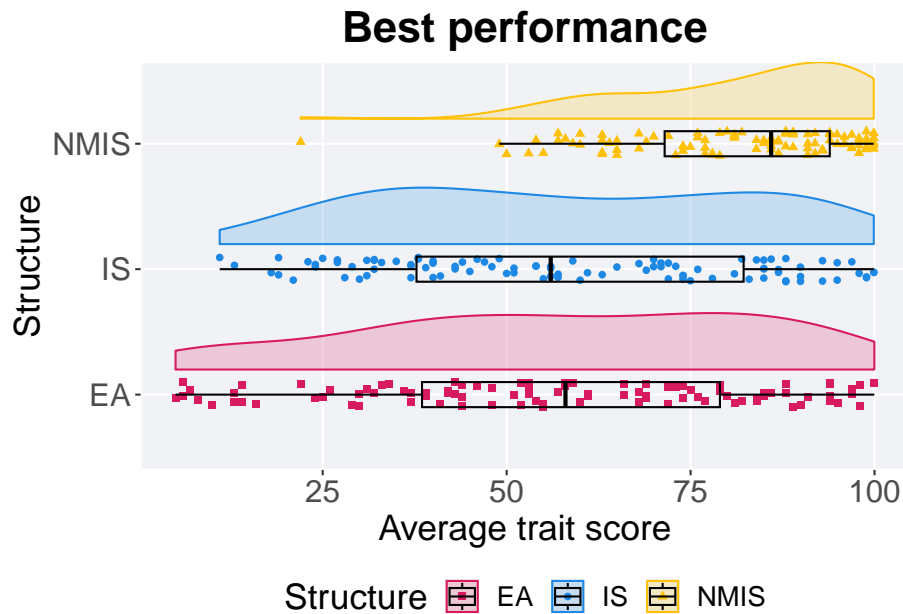
```



#### 5.2.1.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(base_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TRUNCATION' & V
ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Structure, sha
geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
scale_y_continuous(
  name="Average trait score"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Best performance')+
p_theme + coord_flip()
```



#### 5.2.1.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(base_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nSc
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE) / DIMENSIONALITY,
    median = median(VAL, na.rm = TRUE) / DIMENSIONALITY,
    mean = mean(VAL, na.rm = TRUE) / DIMENSIONALITY,
    max = max(VAL, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VAL, na.rm = TRUE) / DIMENSIONALITY
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0    5   58.0  57.0  100.  40.5
## 2 IS          100     0   11   56.0  58.3  99.9  44.5
## 3 NMIS        100     0  22.0  85.9  81.5  99.9  22.4
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VA ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  VAL by Structure
## Kruskal-Wallis chi-squared = 57.688, df = 2, p-value = 2.973e-13
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

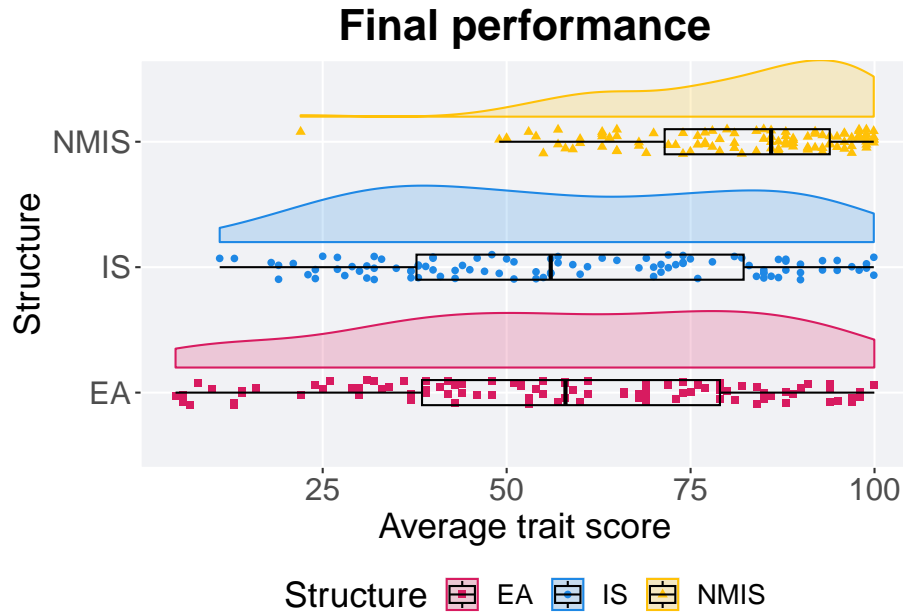
```
pairwise.wilcox.test(x = performance$VAL, g = performance$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VAL and performance$Structure
##
##      EA      IS
## IS   1      -
## NMIS 4.3e-11 1.3e-10
##
## P value adjustment method: bonferroni
```

### 5.2.1.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TRUNCATION') +
  ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fill = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final performance') +
  p_theme + coord_flip()
```



#### 5.2.1.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection` == 'First generation')
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_fit_max)),
    min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0    5   58.0  57.0  100.  40.5
## 2 IS          100     0   11   56.0  58.3  99.9  44.5
## 3 NMIS        100     0  22.0  85.9  81.5  99.9  22.4
```

Kruskal–Wallis test provides evidence of difference among selection schemes.



```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 57.688, df = 2, p-value = 2.973e-13
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

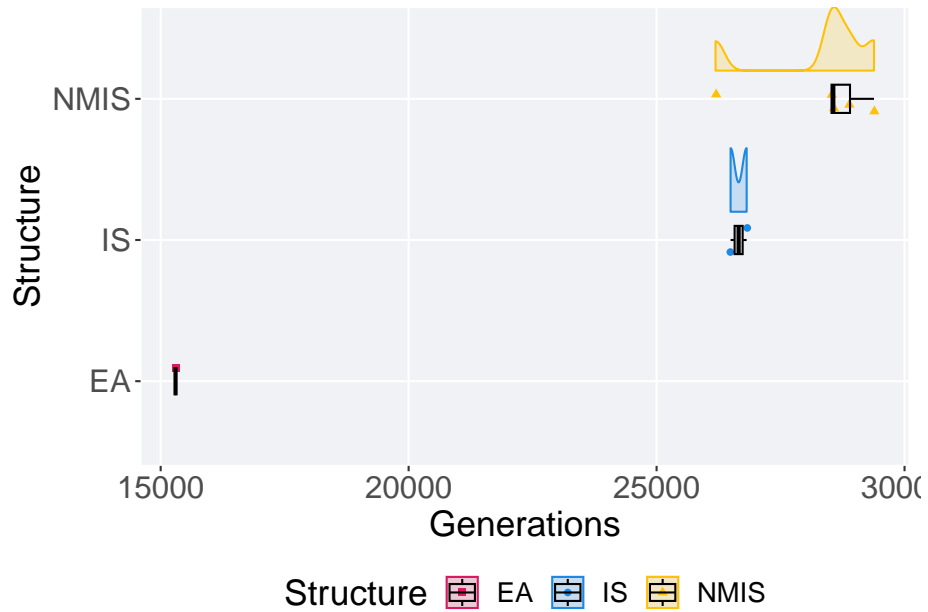
```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.method = "b"
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$pop_fit_max and performance$Structure
##
##      EA      IS
## IS   1      -
## NMIS 4.3e-11 1.3e-10
##
## P value adjustment method: bonferroni
```

### 5.2.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(base_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TRUNCATION' & Gen
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Stru
    geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
    geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
    geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
    scale_shape_manual(values=SHAPE)+
    scale_y_continuous(
      name="Generations"
    ) +
    scale_x_discrete(
      name="Structure"
    ) +
    scale_colour_manual(values = cb_palette) +
    scale_fill_manual(values = cb_palette) +
    p_theme + coord_flip()
```



### 5.2.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(base_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int>  <int> <int>  <dbl> <dbl> <int> <dbl>
## 1 EA          1      0 15300  15300 15300  15300    0
## 2 IS          2      0 26492  26654 26654  26816   162
## 3 NMIS        5      0 26188  28563 28313. 29384   372
```

Kruskal–Wallis test provides evidence of no difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)

##
## Kruskal-Wallis rank sum test
##
## data: Generations by Structure
## Kruskal-Wallis chi-squared = 3.3833, df = 2, p-value = 0.1842
```

### 5.2.3 Activation gene coverage

Activation gene coverage analysis.

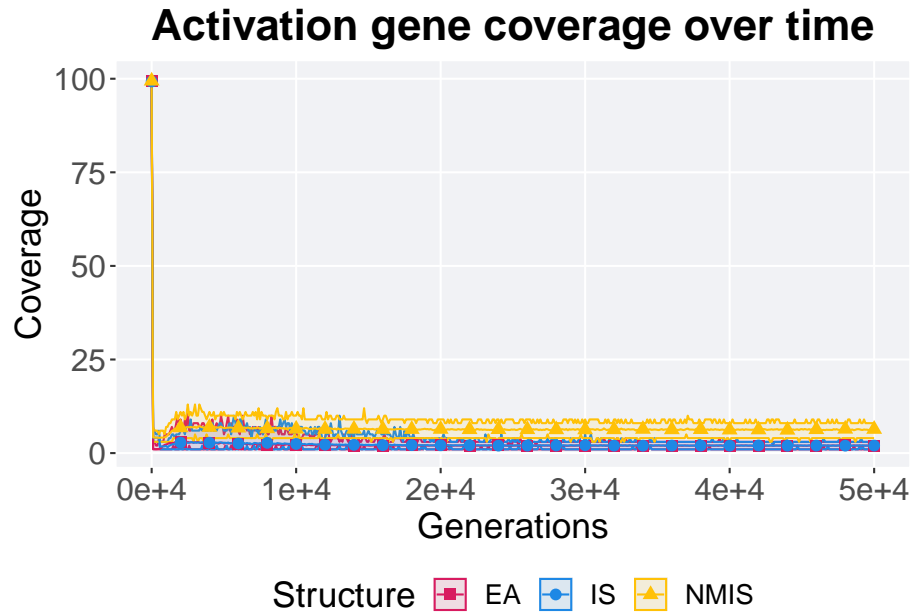
#### 5.2.3.1 Coverage over time

Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TF
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )

## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.

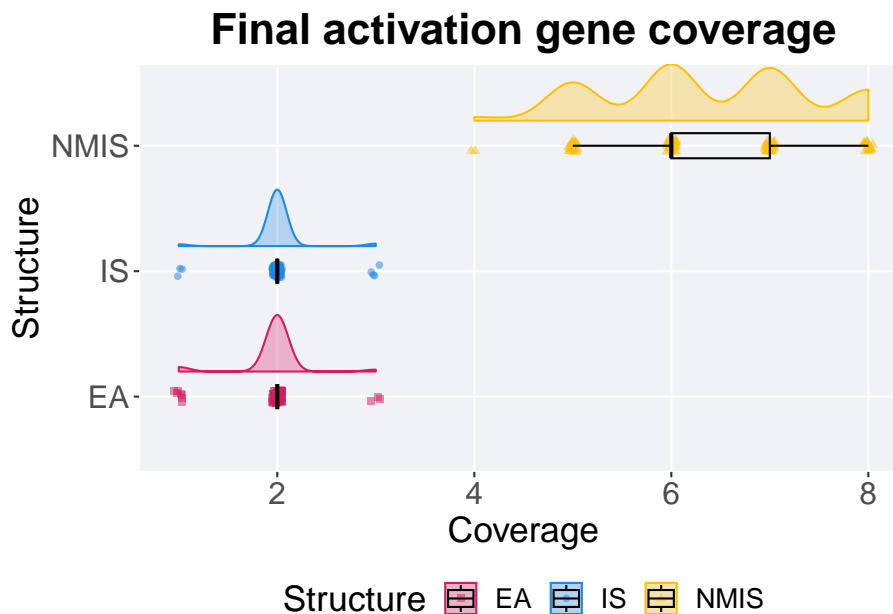
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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
```



#### 5.2.3.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```
### end of run
filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure),
    geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
    geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
    geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
    scale_shape_manual(values=SHAPE) +
    scale_y_continuous(
      name="Coverage"
    ) +
    scale_x_discrete(
      name="Structure"
    ) +
    scale_colour_manual(values = cb_palette) +
    scale_fill_manual(values = cb_palette) +
    ggtitle('Final activation gene coverage') +
    p_theme + coord_flip()
```



#### 5.2.3.2.1 Stats

Summary statistics for activation gene coverage.

```
coverage = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_act_cov)),
    min = min(pop_act_cov, na.rm = TRUE),
    median = median(pop_act_cov, na.rm = TRUE),
    mean = mean(pop_act_cov, na.rm = TRUE),
    max = max(pop_act_cov, na.rm = TRUE),
    IQR = IQR(pop_act_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt  min median  mean  max  IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA         100     0     1     2  1.96     3     0
## 2 IS         100     0     1     2  2.01     3     0
## 3 NMIS       100     0     4     6  6.38     8     1
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)

##
## Kruskal-Wallis rank sum test
##
## data: pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 258.93, df = 2, p-value < 2.2e-16

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on
activation gene coverage.

pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method
                    paired = FALSE, conf.int = FALSE, alternative = 'g')

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS  0.34    -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 5.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the contradictory objectives diagnostic.

### 5.3.1 Performance

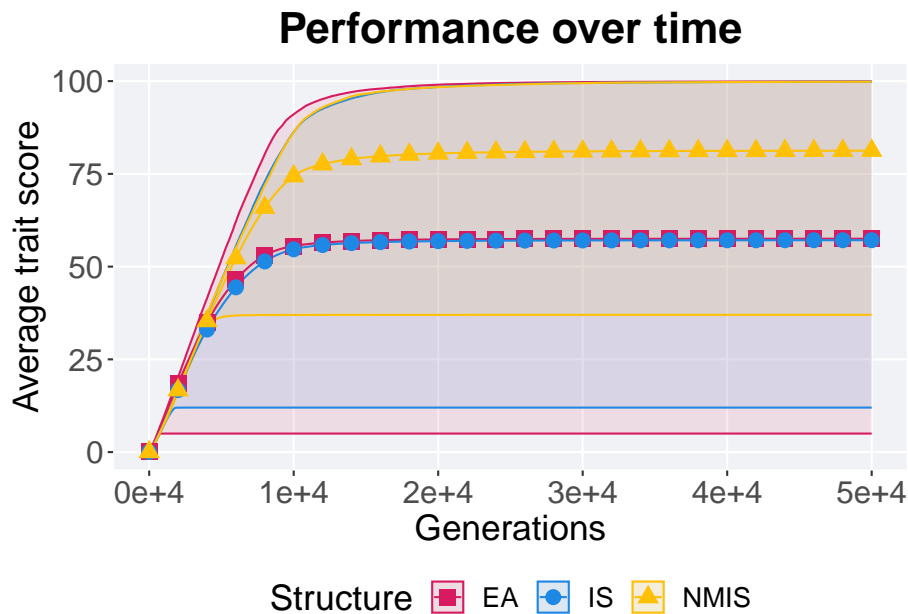
#### 5.3.1.1 Performance over time

```
lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection` ~ Sch
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %>= 2000 == 0), size = 2.5, stroke = 2.0,
  scale_y_continuous(
```

```

    name="Average trait score"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme

```



#### 5.3.1.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```

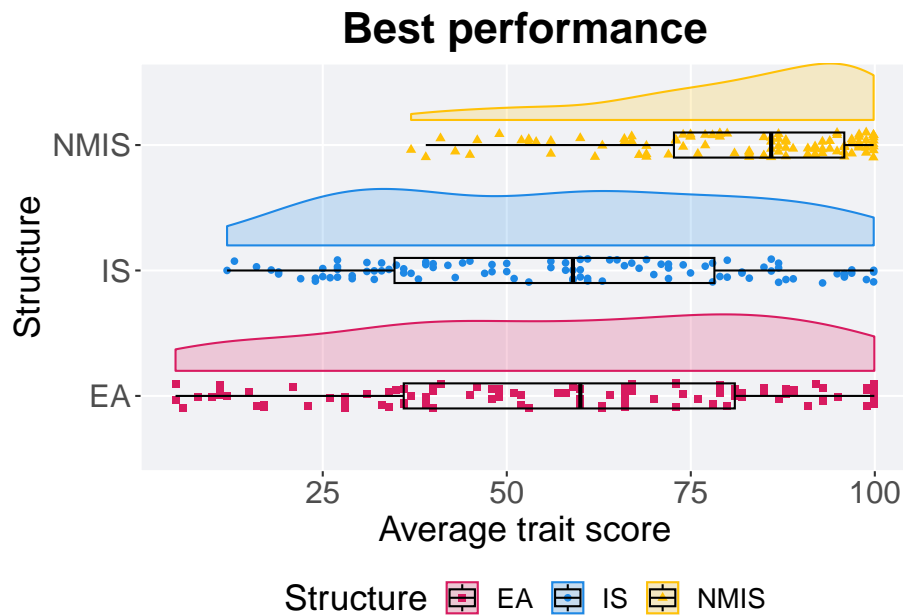
filter(base_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT' & V
  ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Structure, sha
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +

```

```

scale_y_continuous(
  name="Average trait score"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Best performance')+
p_theme + coord_flip()

```



#### 5.3.1.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

performance = filter(base_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nSc
performance %>%
group_by(Structure) %>%
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(VAL)),
  min = min(VAL, na.rm = TRUE) / DIMENSIONALITY,
  median = median(VAL, na.rm = TRUE) / DIMENSIONALITY,
  mean = mean(VAL, na.rm = TRUE) / DIMENSIONALITY,

```



```

    max = max(VA, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VA, na.rm = TRUE) / DIMENSIONALITY
  )

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0     5    60.0  57.5  99.9  45.0
## 2 IS          100     0    12    59.0  57.1  99.9  43.5
## 3 NMIS        100     0   37.0   85.9  81.2  99.8  23.1

```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VA ~ Structure, data = performance)
```

```

##
##  Kruskal-Wallis rank sum test
##
## data:  VA by Structure
## Kruskal-Wallis chi-squared = 52.543, df = 2, p-value = 3.895e-12

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

```

##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VA and performance$Structure
##
##      EA      IS
## IS    1      -
## NMIS 5.9e-09 5.3e-11
##
## P value adjustment method: bonferroni

```

### 5.3.1.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```

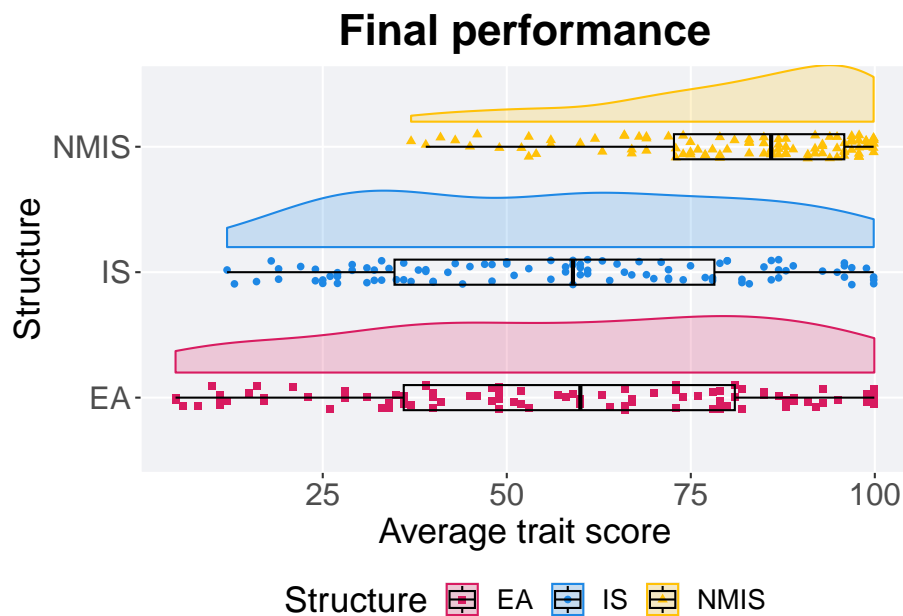
filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT')
ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fill = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  )

```

```

) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Final performance')+
p_theme + coord_flip()

```



#### 5.3.1.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

performance = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection` == 'Satisfactory')
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_fit_max)),
    min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
  )

```

```
)

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0    5    60.0  57.5  99.9  45.0
## 2 IS          100     0   12    59.0  57.1  99.9  43.5
## 3 NMIS        100     0  37.0   85.9  81.2  99.8  23.1
```

Kruskal-Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 52.543, df = 2, p-value = 3.895e-12
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.method = "b",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$pop_fit_max and performance$Structure
##
##      EA      IS
## IS    1      -
## NMIS 5.9e-09 5.3e-11
##
## P value adjustment method: bonferroni
```

### 5.3.2 Generation satisfactory solution found

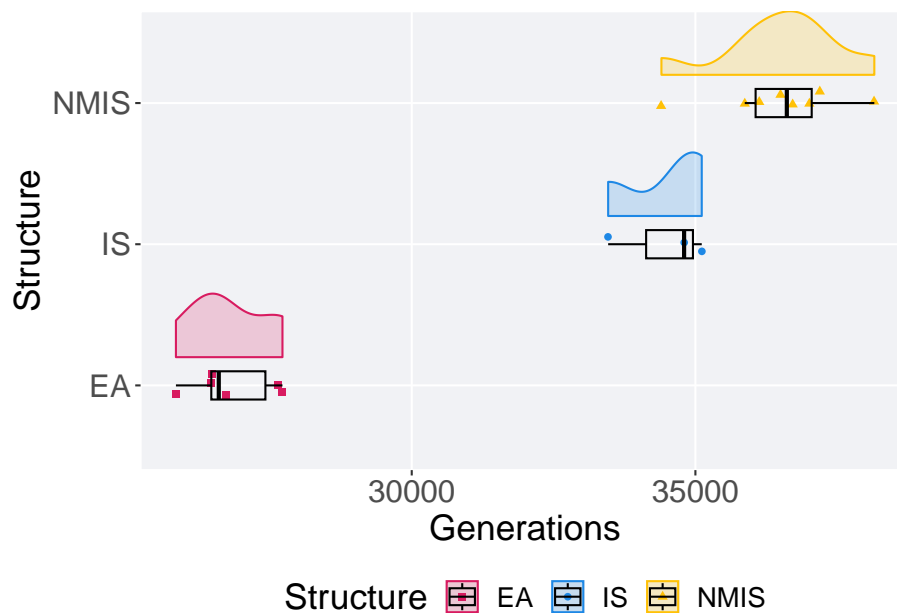
First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(base_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT') & Ger
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Generations"
```

```

) +
scale_x_discrete(
  name="Structure"
) +
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
p_theme + coord_flip()

```



### 5.3.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(base_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )

```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          6      0 25843 26598. 26813 27721  954
## 2 IS          3      0 33462 34801  34458. 35112  825
## 3 NMIS        8      0 34401 36612. 36496. 38154  989.
```

Kruskal-Wallis test provides evidence of no difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  Generations by Structure
## Kruskal-Wallis chi-squared = 12.797, df = 2, p-value = 0.001664
pairwise.wilcox.test(x = ssf$Generations, g = ssf$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum exact test
##
## data:  ssf$Generations and ssf$Structure
##
##      EA      IS
## IS  0.036 -
## NMIS 0.001 0.073
##
## P value adjustment method: bonferroni
```

### 5.3.3 Activation gene coverage

Activation gene coverage analysis.

#### 5.3.3.1 Coverage over time

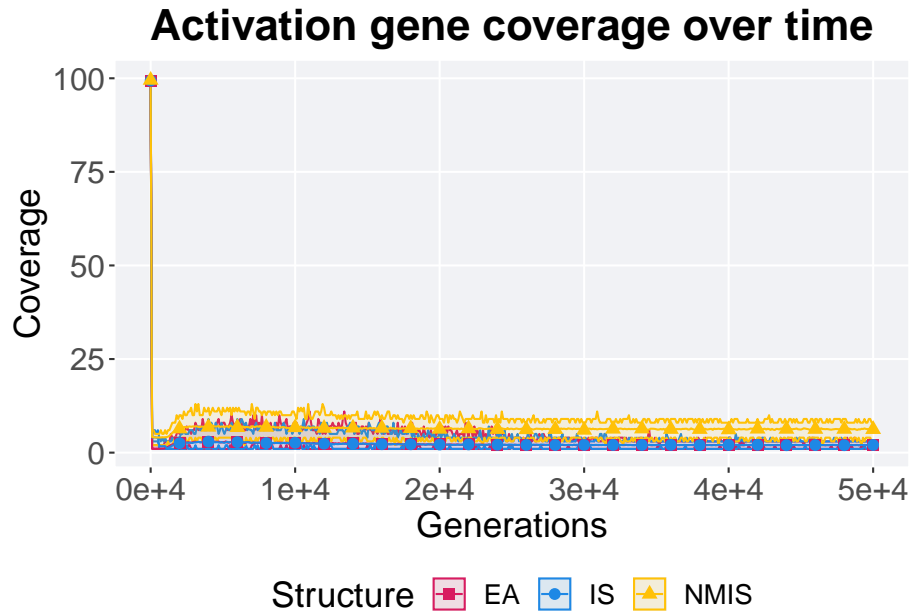
Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )
```

## `summarise()` has grouped output by 'Structure'. You can override using the

## `` .groups`` argument.

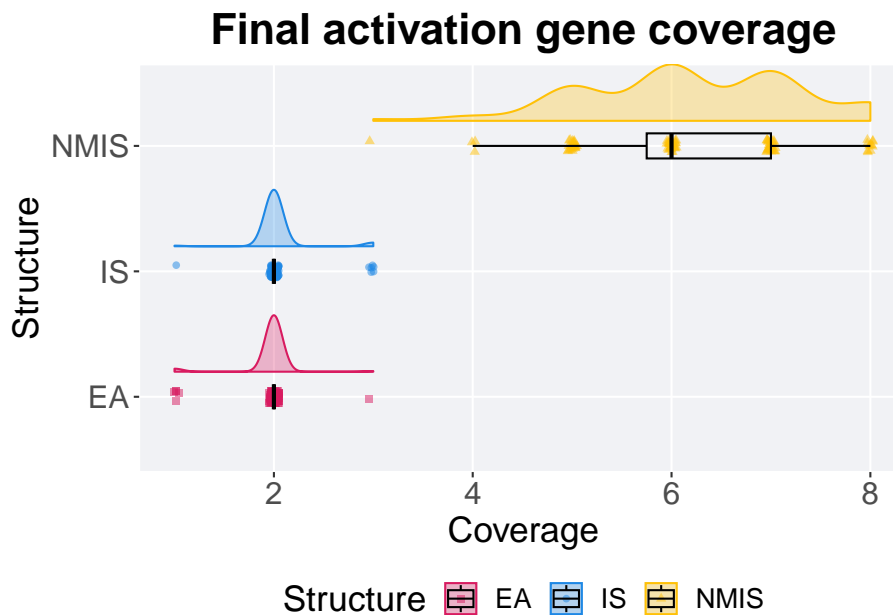
```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
    scale_y_continuous(
      name="Coverage"
    ) +
    scale_x_continuous(
      name="Generations",
      limits=c(0, 50000),
      breaks=c(0, 10000, 20000, 30000, 40000, 50000),
      labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
    ) +
    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
```



### 5.3.3.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```
### end of run
filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT')
ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme + coord_flip()
```



#### 5.3.3.2.1 Stats

Summary statistics for activation gene coverage.

```
coverage = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT')
coverage %>%
  group_by(Structure) %>%
```

```
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(pop_act_cov)),
  min = min(pop_act_cov, na.rm = TRUE),
  median = median(pop_act_cov, na.rm = TRUE),
  mean = mean(pop_act_cov, na.rm = TRUE),
  max = max(pop_act_cov, na.rm = TRUE),
  IQR = IQR(pop_act_cov, na.rm = TRUE)
)
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     2  1.96     3     0
## 2 IS          100     0     1     2  2.05     3     0
## 3 NMIS        100     0     3     6  6.22     8  1.25
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 264.53, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```
pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method =
  paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS  0.019  -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

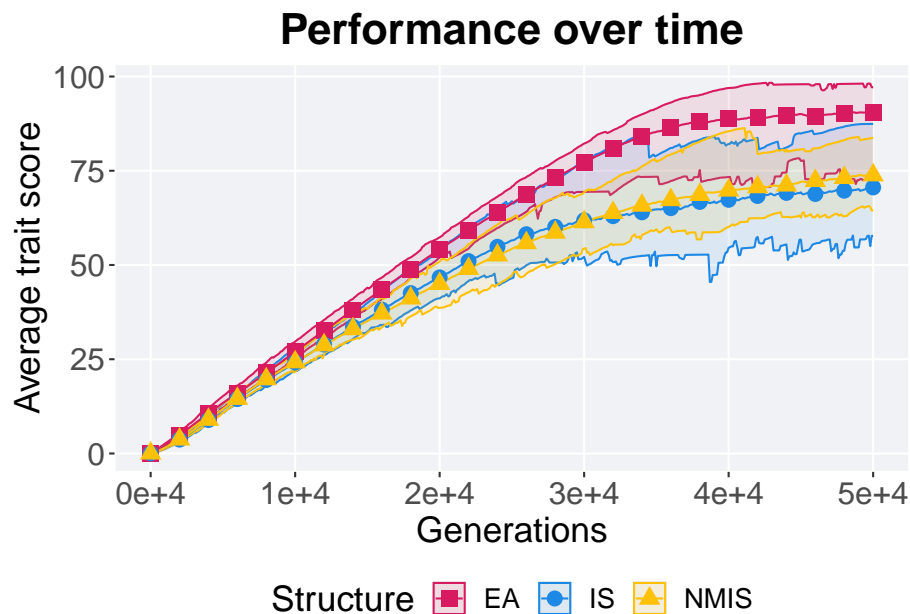


## 5.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the contradictory objectives diagnostic.

### 5.4.1 Performance

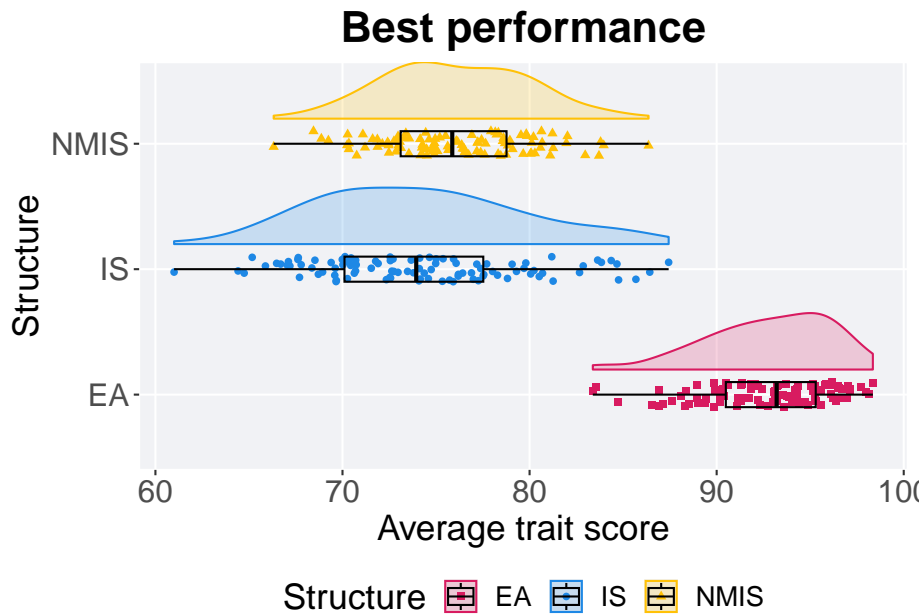
```
lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'LE
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme
```



#### 5.4.1.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(base_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'LEXI
  ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Stru
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best performance')+
  p_theme + coord_flip()
```



#### 5.4.1.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(base_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'I
performance$Structure = factor(performance$Structure, levels=c('EA', 'NMIS', 'IS'))
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE) / DIMENSIONALITY,
    median = median(VAL, na.rm = TRUE) / DIMENSIONALITY,
    mean = mean(VAL, na.rm = TRUE) / DIMENSIONALITY,
    max = max(VAL, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VAL, na.rm = TRUE) / DIMENSIONALITY
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA         100     0  83.4  93.2  92.8  98.4  4.80
## 2 NMIS       100     0  66.3  75.9  76.1  86.4  5.66
## 3 IS        100     0  61.0  73.9  74.1  87.4  7.42
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VA ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  VA by Structure
## Kruskal-Wallis chi-squared = 202.16, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

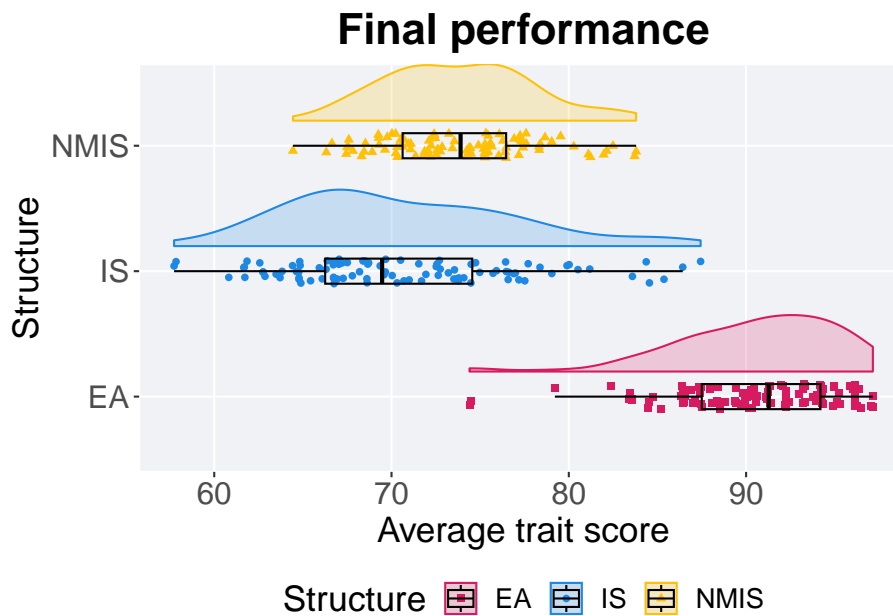
```
pairwise.wilcox.test(x = performance$VA, g = performance$Structure, p.adjust.method =
                    paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VA and performance$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 0.0032
##
## P value adjustment method: bonferroni
```

### 5.4.1.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fi
geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
scale_y_continuous(
  name="Average trait score"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Final performance')+
p_theme + coord_flip()
```



#### 5.4.1.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme`
performance$Structure = factor(performance$Structure, levels=c('EA','NMIS','IS'))
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_fit_max)),
    min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA         100     0  74.4  91.3  90.6  97.2  6.69
## 2 NMIS       100     0  64.4  73.9  73.8  83.8  5.84
## 3 IS         100     0  57.7  69.5  70.6  87.4  8.30
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 198.85, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.m = 'p',
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$pop_fit_max and performance$Structure
##
##      EA      NMIS
## NMIS < 2e-16 -
## IS   < 2e-16 1.6e-05
##
## P value adjustment method: bonferroni
```

### 5.4.2 Activation gene coverage

Activation gene coverage analysis.

#### 5.4.2.1 Coverage over time

Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection`\nSch
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )
```

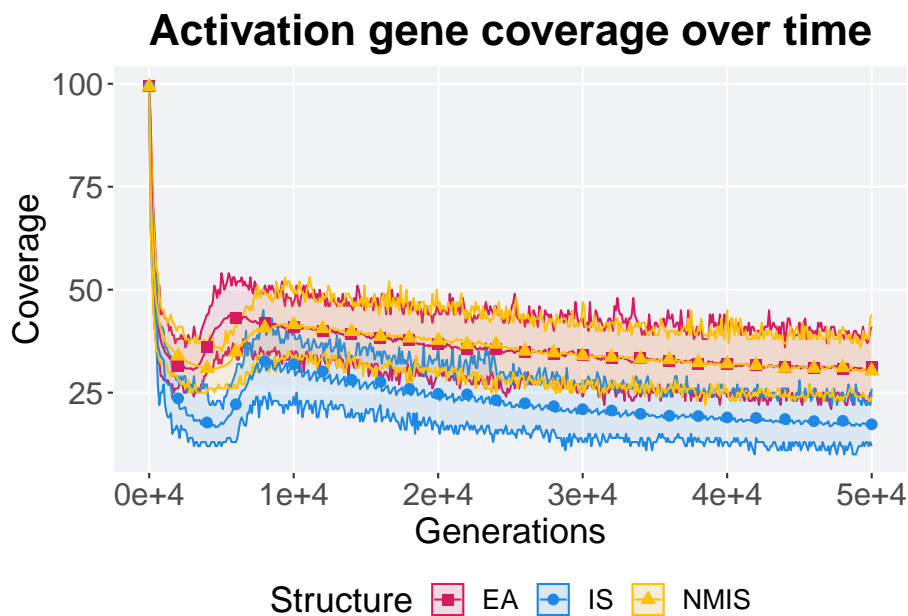
```
## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.
```

```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
```

```

scale_y_continuous(
  name="Coverage"
) +
scale_x_continuous(
  name="Generations",
  limits=c(0, 50000),
  breaks=c(0, 10000, 20000, 30000, 40000, 50000),
  labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
) +
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

```



#### 5.4.2.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```

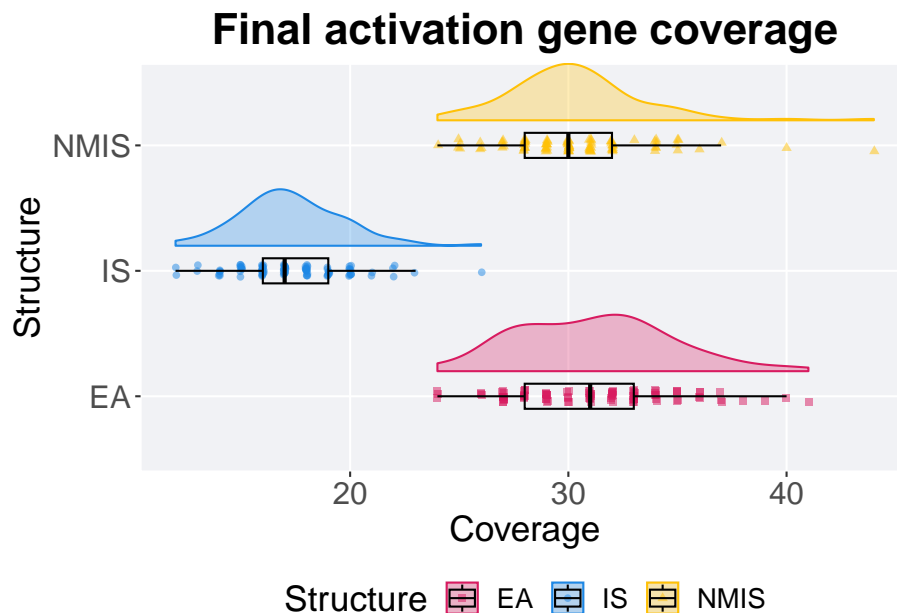
### end of run
filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'LEXICASE')
ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +

```

```

geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
scale_shape_manual(values=SHAPE) +
scale_y_continuous(
  name="Coverage"
) +
scale_x_discrete(
  name="Structure"
) +
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle('Final activation gene coverage') +
p_theme + coord_flip()

```



#### 5.4.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(base_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection` == 'n')
coverage$Structure = factor(coverage$Structure, levels=c('EA', 'NMIS', 'IS'))
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_act_cov)),
    min = min(pop_act_cov, na.rm = TRUE),

```



```

median = median(pop_act_cov, na.rm = TRUE),
mean = mean(pop_act_cov, na.rm = TRUE),
max = max(pop_act_cov, na.rm = TRUE),
IQR = IQR(pop_act_cov, na.rm = TRUE)
)

```

```

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int>   <dbl> <dbl> <int> <dbl>
## 1 EA          100     0    24     31  31.2   41     5
## 2 NMIS         100     0    24     30  30.3   44     4
## 3 IS          100     0    12     17  17.3   26     3

```

Kruskal-Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```

##
## Kruskal-Wallis rank sum test
##
## data:  pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 201.31, df = 2, p-value < 2.2e-16

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```

pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')

```

```

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_act_cov and coverage$Structure
##
##      EA      NMIS
## NMIS 0.077  -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni

```



## Chapter 6

# MI50: Exploitation rate results

Here we present the results for **best performances** found by each selection scheme replicate on the exploitation rate diagnostic with configurations presented below. For our the configuration of these experiments, we execute migrations every 50 generations and there are 4 islands in a ring topology. When migrations occur, we swap two individuals (same position on each island) and guarantee that no solution can return to the same island. Best performance found refers to the largest average trait score found in a given population. Note that performance values fall between 0.0 and 100.0.

### 6.1 Analysis dependencies

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

### 6.2 Truncation selection

Here we analyze how the different population structures affect truncation selection (size 8) on the exploitation rate diagnostic.

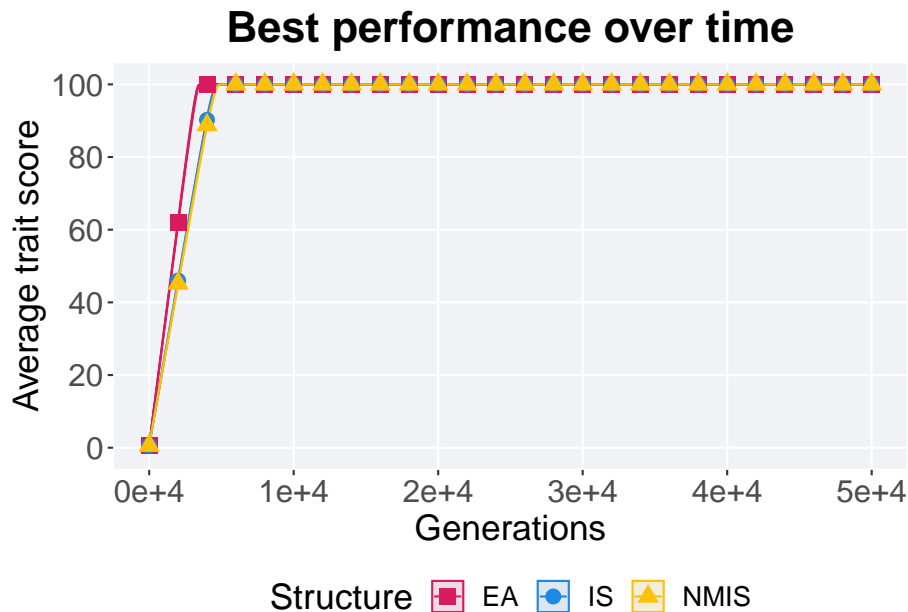
#### 6.2.1 Performance over time

```
lines = filter(mi50_over_time, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TRUNCATION')
group_by(Structure, Generations) %>%
```

```

dplyr::summarise(
  min = min(pop_fit_max) / DIMENSIONALITY,
  mean = mean(pop_fit_max) / DIMENSIONALITY,
  max = max(pop_fit_max) / DIMENSIONALITY
)
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0,
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Best performance over time") +
  p_theme

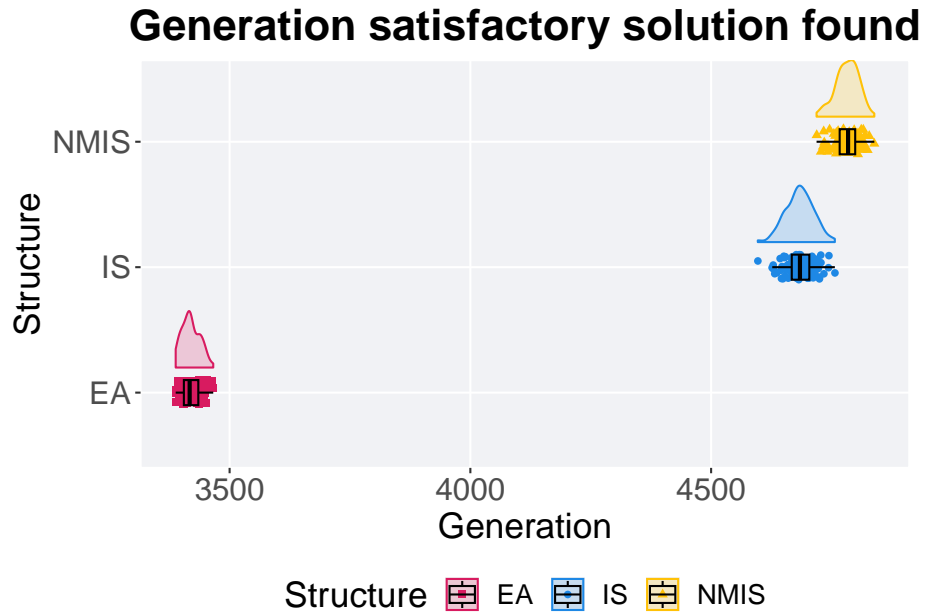
```



### 6.2.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi50_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TRUNCATION') %>%
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Generation"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Generation satisfactory solution found') +
  p_theme + coord_flip()
```



### 6.2.3 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(mi50_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TRUE')
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int>  <dbl> <dbl> <int> <dbl>
## 1 EA          100     0  3388  3417  3420.  3466   30
## 2 IS          100     0  4597  4684. 4684.  4757  36.5
## 3 NMIS        100     0  4719  4784. 4783.  4839  32.2
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)

##
## Kruskal-Wallis rank sum test
##
## data: Generations by Structure
## Kruskal-Wallis chi-squared = 264.73, df = 2, p-value < 2.2e-16

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: ssf$Generations and ssf$Structure
##
##      EA      IS
## IS  <2e-16 -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 6.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the exploitation rate diagnostic.

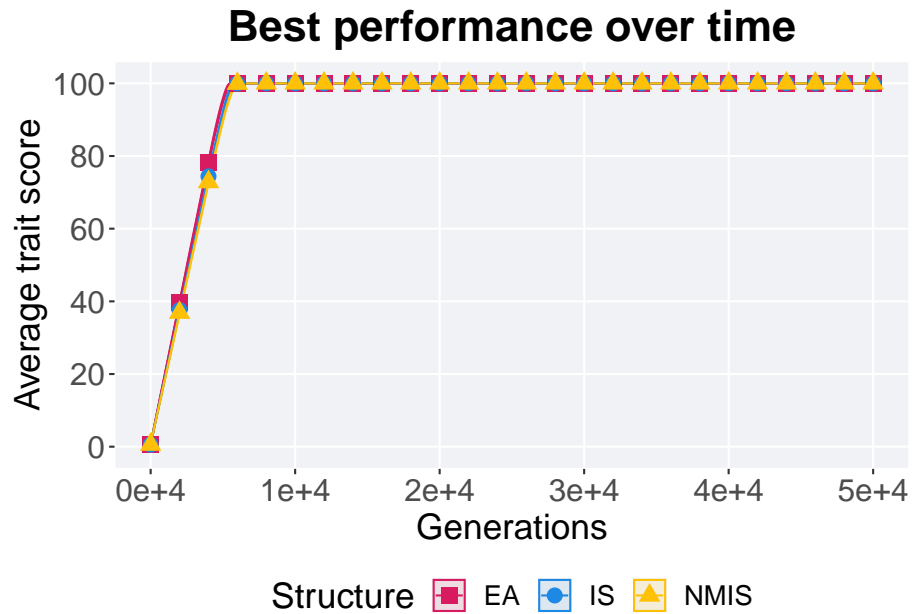
### 6.3.1 Performance over time

```
lines = filter(mi50_over_time, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TOURNAMENT')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.0)
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
```

```

labels=c("0", "20", "40", "60", "80", "100")
) +
scale_x_continuous(
  name="Generations",
  limits=c(0, 50000),
  breaks=c(0, 10000, 20000, 30000, 40000, 50000),
  labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
) +
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle("Best performance over time") +
p_theme

```



### 6.3.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```

filter(mi50_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TOURNAMENT')
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +

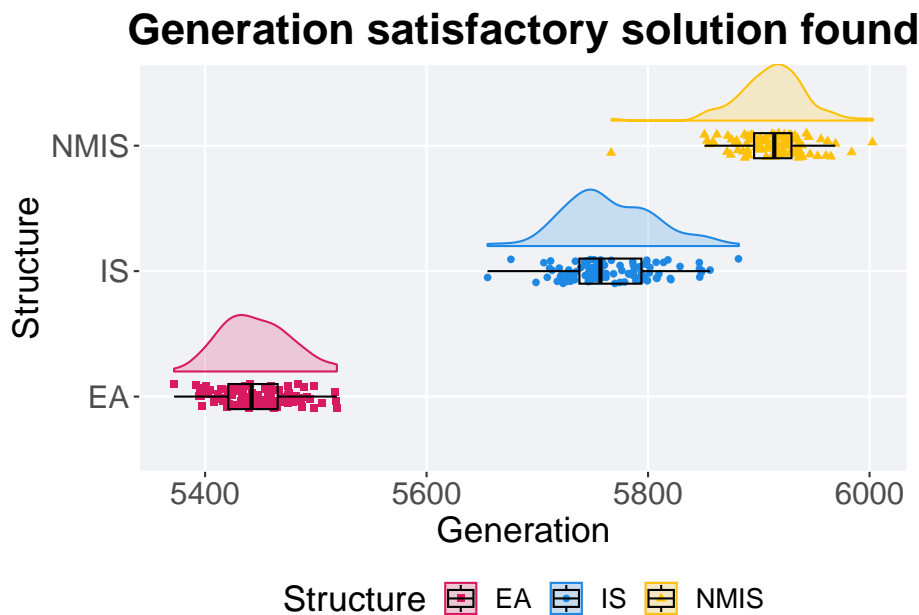
```



```

scale_y_continuous(
  name="Generation"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Generation satisfactory solution found')+
p_theme + coord_flip()

```



### 6.3.3 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(mi50_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TOURNAMENT' &
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),

```

```

max = max(Generations, na.rm = TRUE),
IQR = IQR(Generations, na.rm = TRUE)
)

```

```

## # A tibble: 3 x 8
##   Structure count na_cnt   min median  mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0  5372  5442 5446.  5519  44.5
## 2 IS          100     0  5655  5757 5765.  5882   56
## 3 NMIS        100     0  5767  5914 5912.  6003  33.8

```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```

##
##  Kruskal-Wallis rank sum test
##
## data:  Generations by Structure
## Kruskal-Wallis chi-squared = 264.22, df = 2, p-value < 2.2e-16

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```

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

```

```

##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  ssf$Generations and ssf$Structure
##
##      EA      IS
## IS  <2e-16 -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni

```

## 6.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the exploitation rate diagnostic.

### 6.4.1 Performance over time

```

lines = filter(mi50_over_time, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme`
group_by(Structure, Generations) %>%
dplyr::summarise(

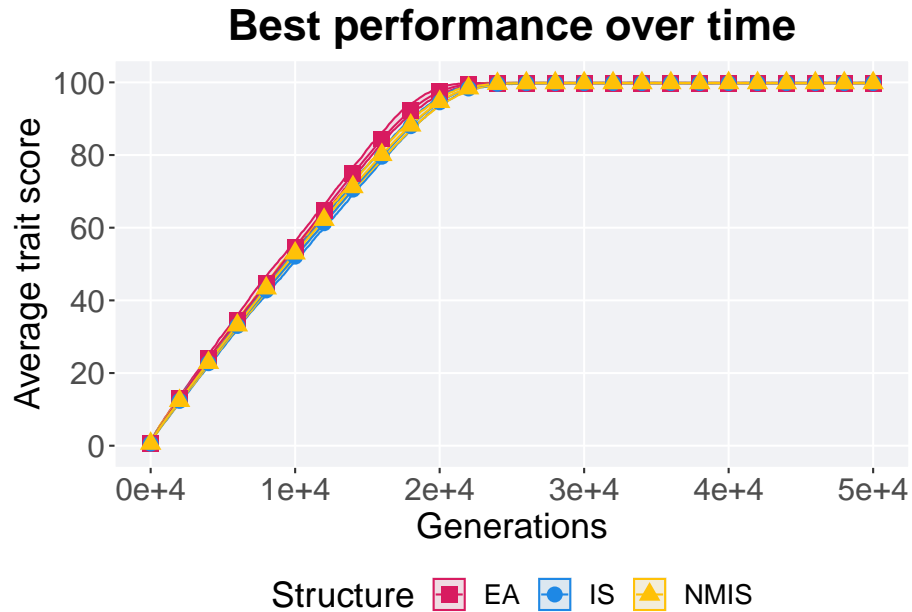
```

```

    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")

  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Best performance over time") +
  p_theme

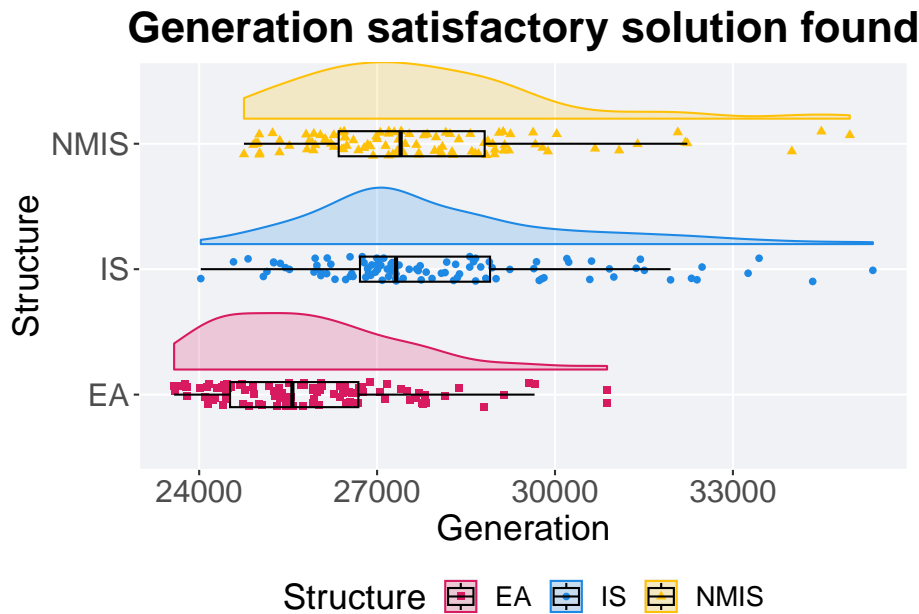
```



#### 6.4.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi50_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'LEXICASE')
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Generation"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Generation satisfactory solution found') +
  p_theme + coord_flip()
```



### 6.4.3 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(mi50_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'LEXICASE' & Ge
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt  min median  mean  max  IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA         100     0 23577 25572 25861. 30878 2163.
## 2 IS         100     0 24027 27320 28031. 35360 2194.
## 3 NMIS       100     0 24755 27398. 27747. 34971 2462.
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```
##  
##  Kruskal-Wallis rank sum test  
##  
## data:  Generations by Structure  
## Kruskal-Wallis chi-squared = 69.626, df = 2, p-value = 7.601e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

```
##  
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction  
##  
## data:  ssf$Generations and ssf$Structure  
##  
##          EA          IS  
## IS    1.2e-13 -  
## NMIS 7.0e-12 1  
##  
## P value adjustment method: bonferroni
```

## Chapter 7

# MI50: Ordered exploitation results

Here we present the results for **best performances** found by each selection scheme replicate on the ordered exploitation diagnostic with configurations presented below. Best performance found refers to the largest average trait score found in a given population. Note that performance values fall between 0.0 and 100.0. For our the configuration of these experiments, we execute migrations every 50 generations and there are 4 islands in a ring topology. When migrations occur, we swap two individuals (same position on each island) and guarantee that no solution can return to the same island.

### 7.1 Analysis dependencies

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

### 7.2 Truncation selection

Here we analyze how the different population structures affect truncation selection (size 8) on the ordered exploitation diagnostic.

#### 7.2.1 Performance over time

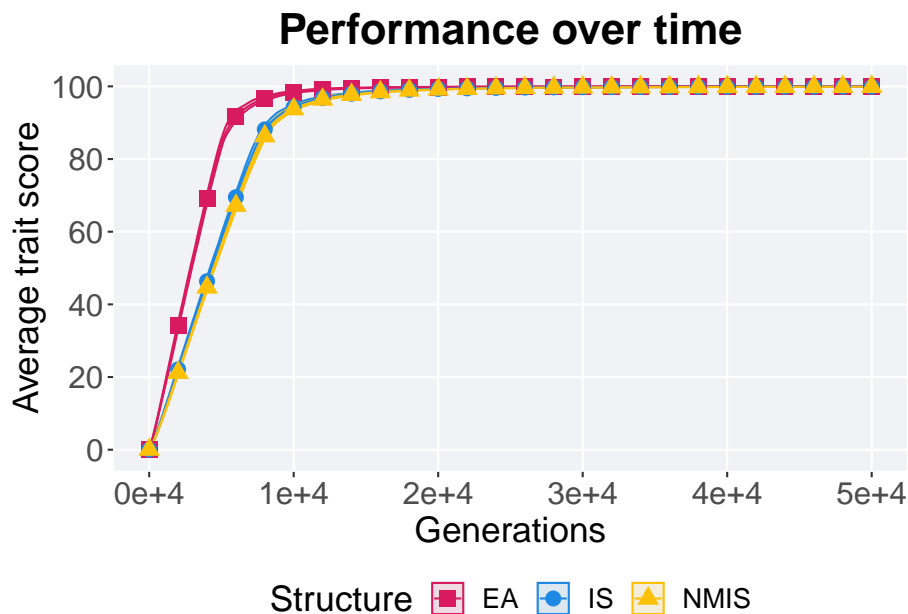
```
lines = filter(mi50_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TRU
  group_by(Structure, Generations) %>%
```

```

dplyr::summarise(
  min = min(pop_fit_max) / DIMENSIONALITY,
  mean = mean(pop_fit_max) / DIMENSIONALITY,
  max = max(pop_fit_max) / DIMENSIONALITY
)
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0,
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme

```



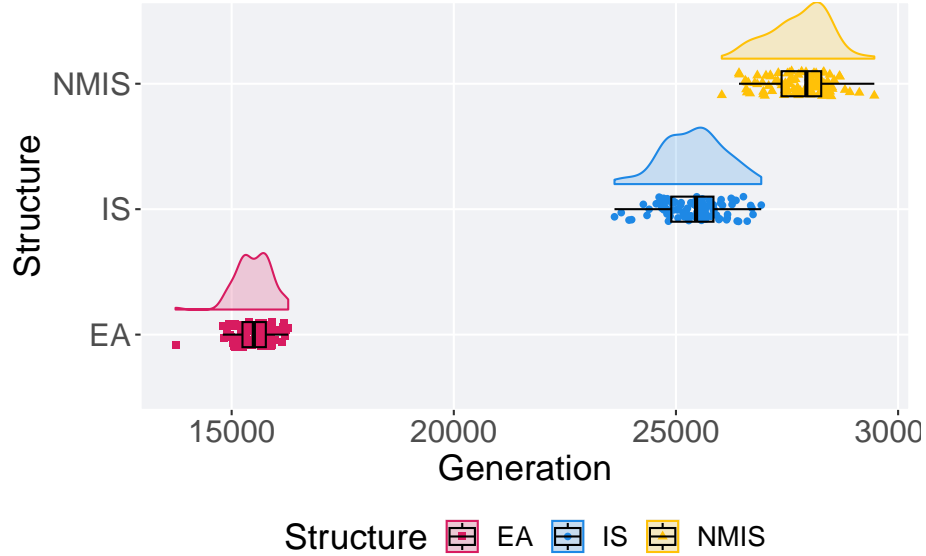


### 7.2.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi50_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TRUNCATION') %>%
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Generation"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Generation satisfactory solution found') +
  p_theme + coord_flip()
```

## Generation satisfactory solution found



### 7.2.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(mi50_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == '1')
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int>  <int> <int>  <dbl> <dbl> <int> <dbl>
## 1 EA          100      0 13737 15500. 15493. 16273  526.
## 2 IS          100      0 23617 25453  25405. 26920  950
## 3 NMIS        100      0 26032 27935  27781. 29465  892.
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)

##
## Kruskal-Wallis rank sum test
##
## data: Generations by Structure
## Kruskal-Wallis chi-squared = 264.17, df = 2, p-value < 2.2e-16

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: ssf$Generations and ssf$Structure
##
##      EA      IS
## IS  <2e-16 -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 7.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the ordered exploitation diagnostic.

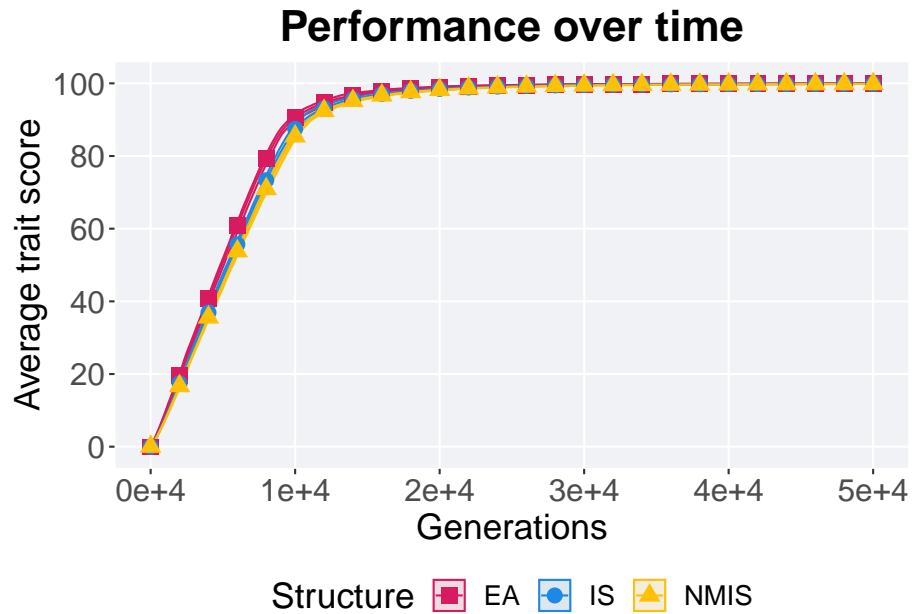
### 7.3.1 Performance over time

```
lines = filter(mi50_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TOURNAMENT')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.0)
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
```

```

labels=c("0", "20", "40", "60", "80", "100")
) +
scale_x_continuous(
  name="Generations",
  limits=c(0, 50000),
  breaks=c(0, 10000, 20000, 30000, 40000, 50000),
  labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
) +
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle("Performance over time") +
p_theme

```



### 7.3.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```

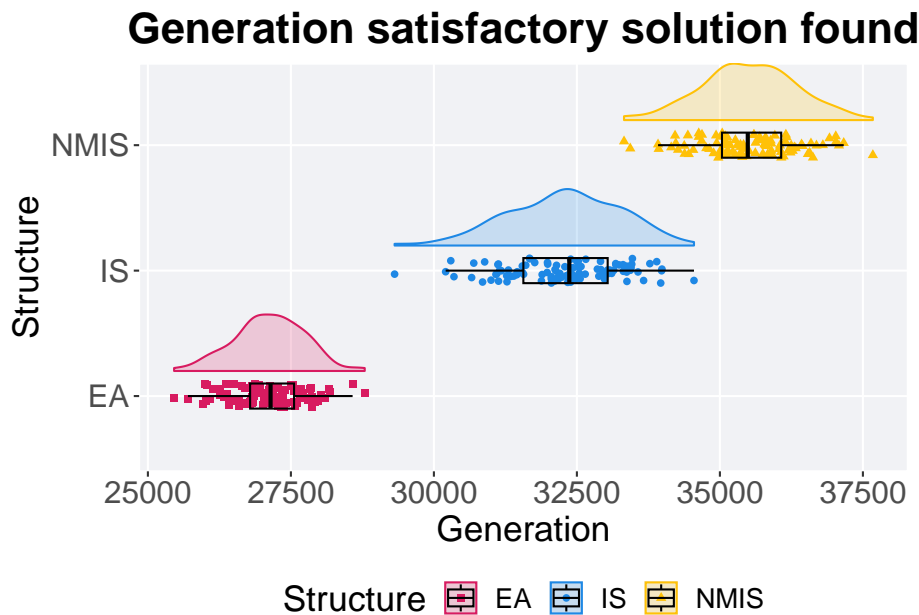
filter(mi50_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TOURNAMENT')
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +

```

```

scale_y_continuous(
  name="Generation"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Generation satisfactory solution found')+
p_theme + coord_flip()

```



#### 7.3.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(mi50_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TOURNAMENT')
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),

```

```

max = max(Generations, na.rm = TRUE),
IQR = IQR(Generations, na.rm = TRUE)
)

```

```

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0 25458 27144. 27124. 28791  769.
## 2 IS          100     0 29313 32368. 32281. 34547 1474
## 3 NMIS        100     0 33324 35488. 35510. 37674 1035.

```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```

##
##  Kruskal-Wallis rank sum test
##
## data:  Generations by Structure
## Kruskal-Wallis chi-squared = 264.58, df = 2, p-value < 2.2e-16

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```

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

```

```

##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  ssf$Generations and ssf$Structure
##
##      EA      IS
## IS  <2e-16 -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni

```

## 7.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the ordered exploitation diagnostic.

### 7.4.1 Performance over time

```

lines = filter(mi50_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScher
group_by(Structure, Generations) %>%
dplyr::summarise(

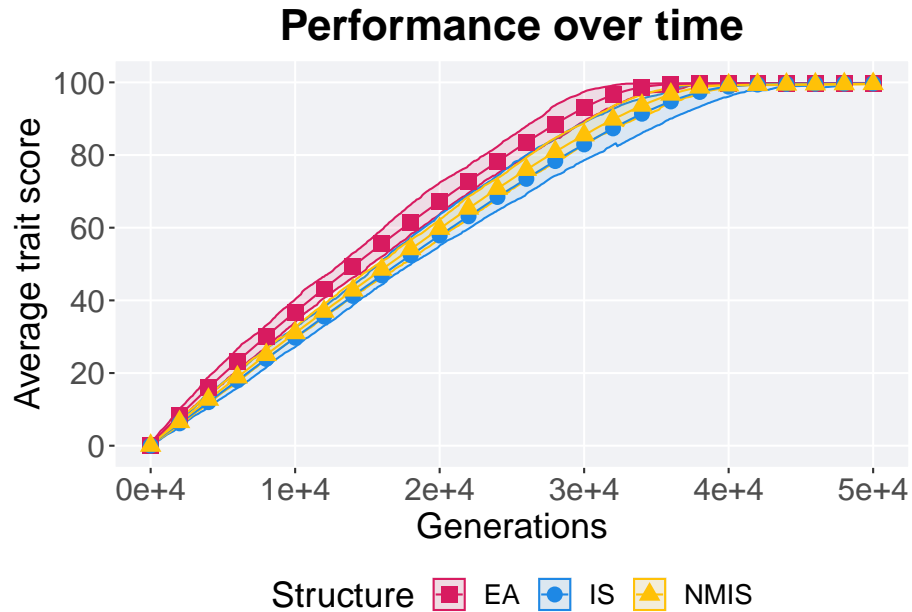
```

```

    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")

  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme

```

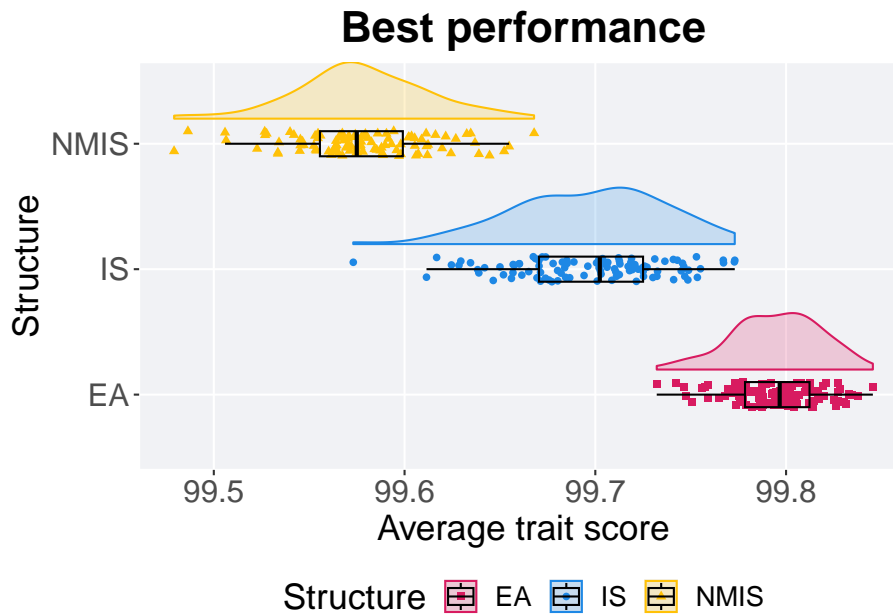


### 7.4.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi50_best, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEXIC
  ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Stru
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best performance')+
  p_theme + coord_flip()
```





#### 7.4.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi50_best, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LE
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VA)),
    min = min(VA, na.rm = TRUE) / DIMENSIONALITY,
    median = median(VA, na.rm = TRUE) / DIMENSIONALITY,
    mean = mean(VA, na.rm = TRUE) / DIMENSIONALITY,
    max = max(VA, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VA, na.rm = TRUE) / DIMENSIONALITY
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0  99.7  99.8  99.8  99.8 0.0338
## 2 IS          100     0  99.6  99.7  99.7  99.8 0.0545
## 3 NMIS        100     0  99.5  99.6  99.6  99.7 0.0435
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VA ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  VA by Structure
## Kruskal-Wallis chi-squared = 259.68, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

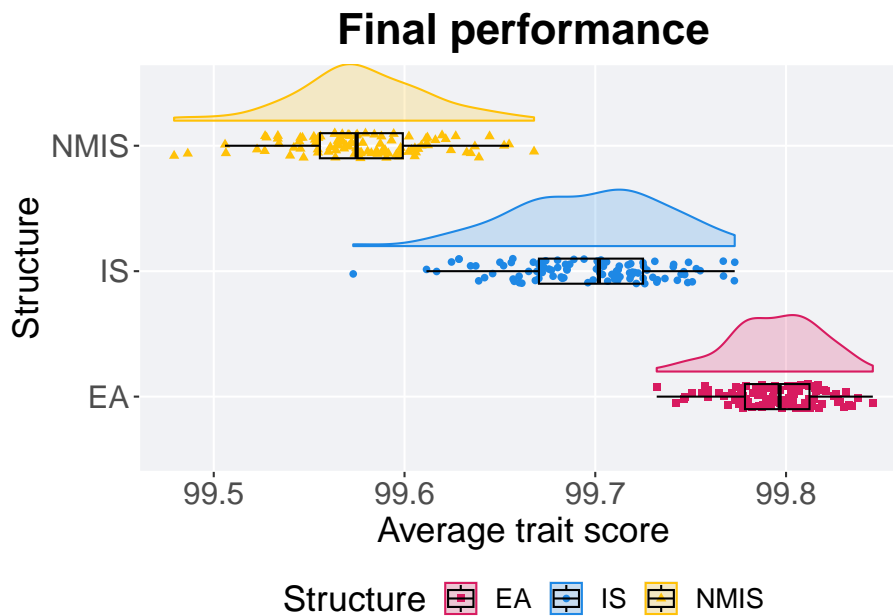
```
pairwise.wilcox.test(x = performance$VA, g = performance$Structure, p.adjust.method =
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VA and performance$Structure
##
##      EA      IS
## IS   <2e-16 -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

### 7.4.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi50_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'I
ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fi
geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
scale_y_continuous(
  name="Average trait score"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Final performance')+
p_theme + coord_flip()
```



#### 7.4.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi50_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` =
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_fit_max)),
    min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0  99.7  99.8  99.8  99.8 0.0338
## 2 IS          100     0  99.6  99.7  99.7  99.8 0.0545
## 3 NMIS        100     0  99.5  99.6  99.6  99.7 0.0435
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 259.68, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.m =
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$pop_fit_max and performance$Structure
##
##      EA      IS
## IS    <2e-16 -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

#### 7.4.4 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

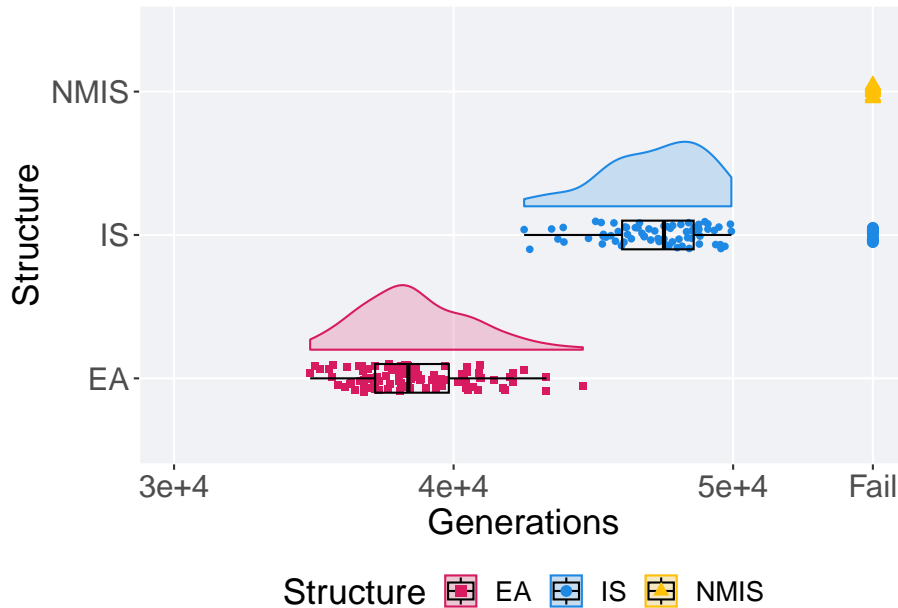
```
lex_fail = filter(mi50_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme`
lex_fail$Generations = 55000
lex_fail$Structure <- factor(lex_fail$Structure, levels = MODEL)

filter(mi50_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEXICAL') %>%
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
    geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = .5) +
    geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
    geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
    geom_point(data = lex_fail, aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
    scale_shape_manual(values=SHAPE) +
    scale_y_continuous(
      name="Generations",
      limits=c(30000, 55000),
      breaks=c(30000, 40000, 50000, 55000),
      labels=c("3e+4", "4e+4", "5e+4", "Fail")
    ) +
    scale_x_discrete(
```

```

  name="Structure"
) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  p_theme + coord_flip()

```



#### 7.4.4.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(mi50_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEXICASE' &
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )

```

## # A tibble: 2 x 8

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

```
## 1 EA          100      0 34868 38382. 38649. 44624 2638.
## 2 IS          70      0 42523 47526. 47195. 49938 2560.
```

Kruskal-Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  Generations by Structure
## Kruskal-Wallis chi-squared = 122.11, df = 1, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  ssf$Generations and ssf$Structure
##
##      EA
## IS <2e-16
##
## P value adjustment method: bonferroni
```

## Chapter 8

# MI50: Contradictory objectives results

Here we present the results for the **satisfactory trait coverage** and **activation gene coverage** generated by each selection scheme replicate on the contradictory objectives diagnostic the configurations presented below. Note both of these values are gathered at the population-level. Activation gene coverage refers to the count of unique activation genes in a given population; this gives us a range of integers between 0 and 100. Satisfactory trait coverage refers to the count of unique satisfied traits in a given population; this gives us a range of integers between 0 and 100. For our the configuration of these experiments, we execute migrations every 50 generations and there are 4 islands in a ring topology. When migrations occur, two individuals are swapped (same position on each island) and guarantee that no solution can return to its original island.

### 8.1 Analysis dependencies

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

### 8.2 Truncation selection

Here we analyze how the different population structures affect truncation selection (size 8) on the contradictory objectives diagnostic.

## 8.2.1 Satisfactory trait coverage

Satisfactory trait coverage analysis.

### 8.2.1.1 Coverage over time

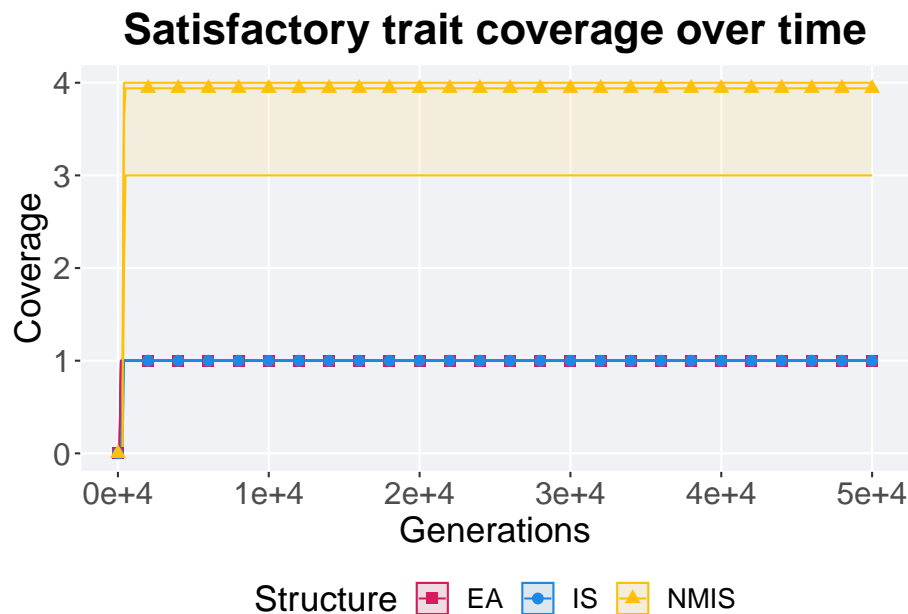
Satisfactory trait coverage over time.

```
lines = filter(mi50_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\n`
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_sat_cov),
    mean = mean(pop_sat_cov),
    max = max(pop_sat_cov)
  )
```

## `summarise()` has grouped output by 'Structure'. You can override using the  
## `.groups` argument.

```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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
```

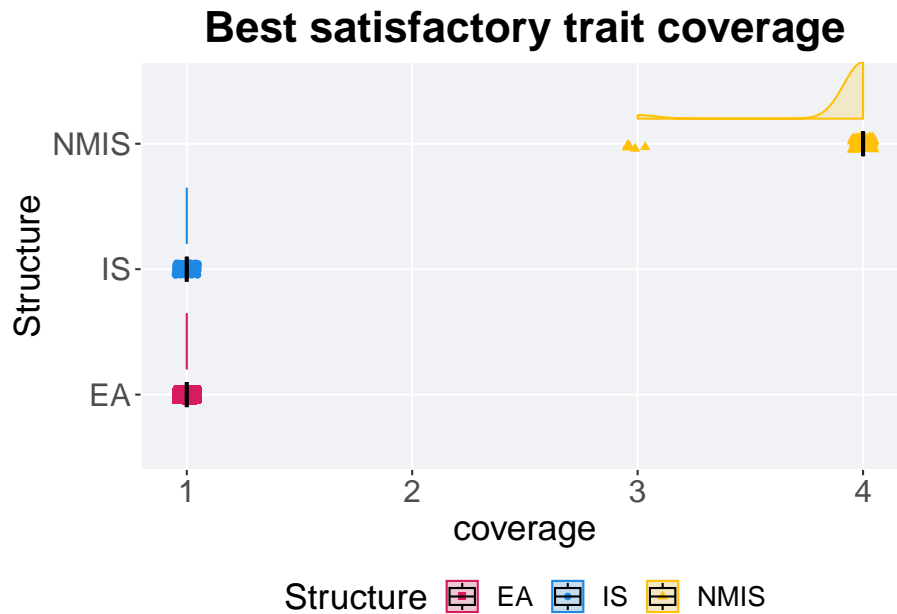




#### 8.2.1.2 Best coverage throughout

Best satisfactory trait coverage throughout 50,000 generations.

```
### best satisfactory trait coverage throughout
filter(mi50_best, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` == 'TRUNCATION')
ggplot(., aes(x = Structure, y = VAL, color = Structure, fill = Structure, shape = Structure))
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory trait coverage') +
  p_theme + coord_flip()
```



#### 8.2.1.2.1 Stats

Summary statistics for the best satisfactory trait coverage.

```
### best
coverage = filter(mi50_best, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nSc
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE),
    median = median(VAL, na.rm = TRUE),
    mean = mean(VAL, na.rm = TRUE),
    max = max(VAL, na.rm = TRUE),
    IQR = IQR(VAL, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     3     4   3.94     4     0
```

Kruskal–Wallis test provides evidence of difference among satisfactory trait

coverage.

```
kruskal.test(VAL ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: VAL by Structure
## Kruskal-Wallis chi-squared = 296.22, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage.

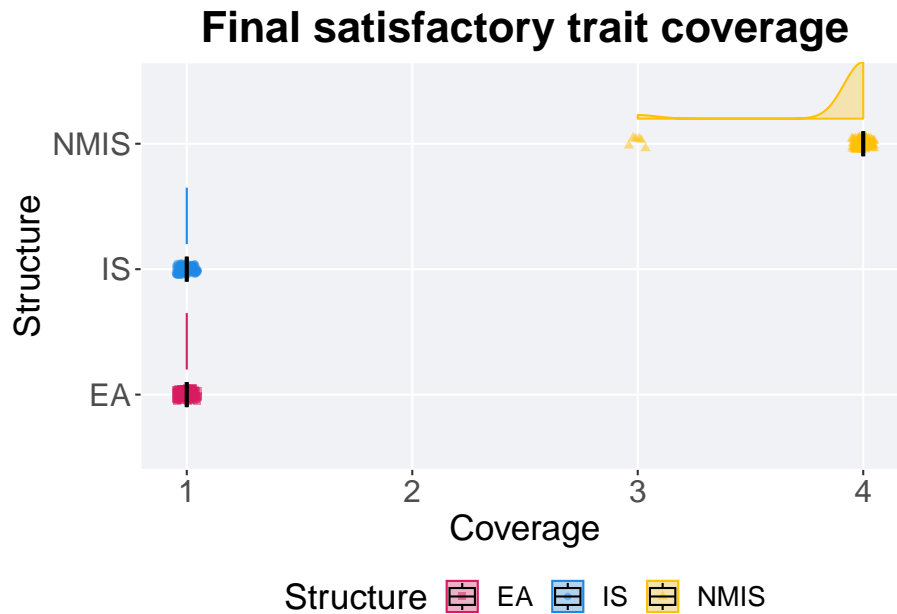
```
pairwise.wilcox.test(x = coverage$VAL, g = coverage$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$VAL and coverage$Structure
##
##      EA      IS
## IS   1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

### 8.2.1.3 End of 50,000 generations

Satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
filter(mi50_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'TRUNCAT
  ggplot(., aes(x = Structure, y = pop_sat_cov, color = Structure, fill = Structure, shape = Stru
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE)+
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final satisfactory trait coverage')+
  p_theme + coord_flip()
```



#### 8.2.1.3.1 Stats

Summary statistics for satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
coverage = filter(mi50_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection` == 'Satisfactory')
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_sat_cov)),
    min = min(pop_sat_cov, na.rm = TRUE),
    median = median(pop_sat_cov, na.rm = TRUE),
    mean = mean(pop_sat_cov, na.rm = TRUE),
    max = max(pop_sat_cov, na.rm = TRUE),
    IQR = IQR(pop_sat_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     3     4    3.94     4     0
```

Kruskal-Wallis test provides evidence of difference among satisfactory trait coverage in the population at the end of 50,000 generations.

```
kruskal.test(pop_sat_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_sat_cov by Structure
## Kruskal-Wallis chi-squared = 296.22, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage in the population at the end of 50,000 generations.

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

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_sat_cov and coverage$Structure
##
##      EA      IS
## IS   1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 8.2.2 Activation gene coverage

Activation gene coverage analysis.

### 8.2.2.1 Coverage over time

Activation gene coverage over time.

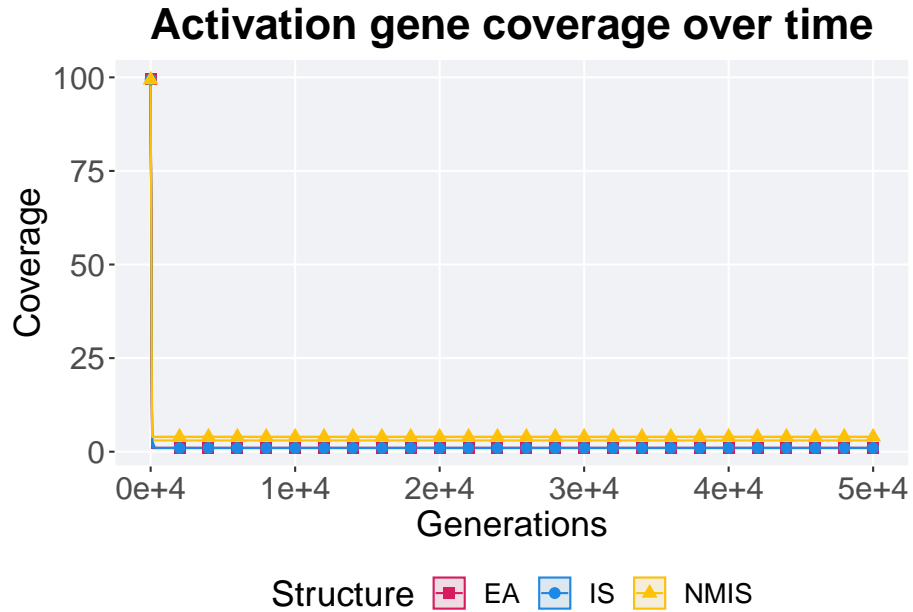
```
# data for lines and shading on plots
lines = filter(mi50_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` ==
               group_by(Structure, Generations) %>%
               dplyr::summarise(
                 min = min(pop_act_cov),
                 mean = mean(pop_act_cov),
                 max = max(pop_act_cov)
               )
```

```
## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.
```

```

ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
    scale_y_continuous(
      name="Coverage"
    ) +
    scale_x_continuous(
      name="Generations",
      limits=c(0, 50000),
      breaks=c(0, 10000, 20000, 30000, 40000, 50000),
      labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
    ) +
    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

```



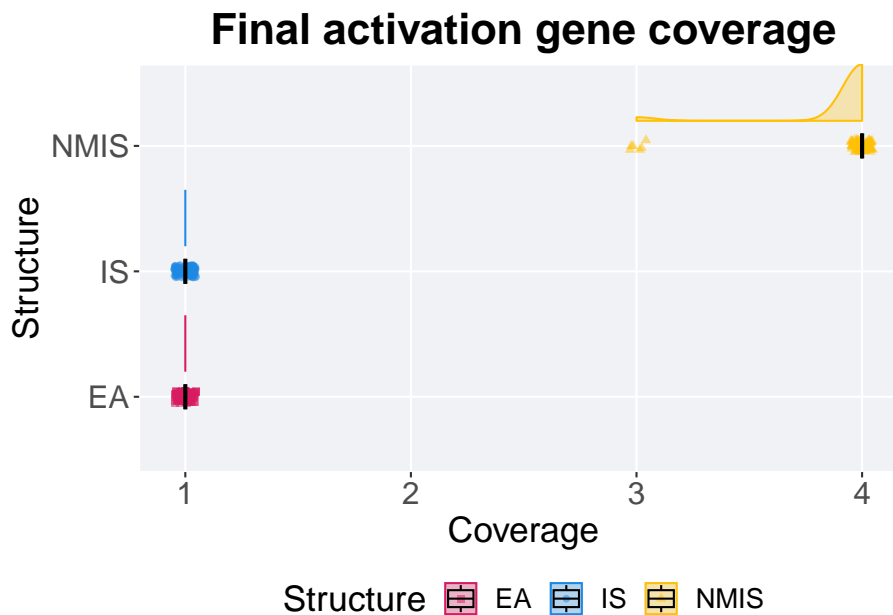
### 8.2.2.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```

### end of run
filter(mi50_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` == 'TRUNCAT
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Stru
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE)+
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage')+
  p_theme + coord_flip()

```



#### 8.2.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(mi50_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme`
coverage %>%
  group_by(Structure) %>%

```

```
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(pop_act_cov)),
  min = min(pop_act_cov, na.rm = TRUE),
  median = median(pop_act_cov, na.rm = TRUE),
  mean = mean(pop_act_cov, na.rm = TRUE),
  max = max(pop_act_cov, na.rm = TRUE),
  IQR = IQR(pop_act_cov, na.rm = TRUE)
)
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     3     4    3.94     4     0
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 296.22, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```
pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method =
  paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS   1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```



## 8.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the contradictory objectives diagnostic.

### 8.3.1 Satisfactory trait coverage

Satisfactory trait coverage analysis.

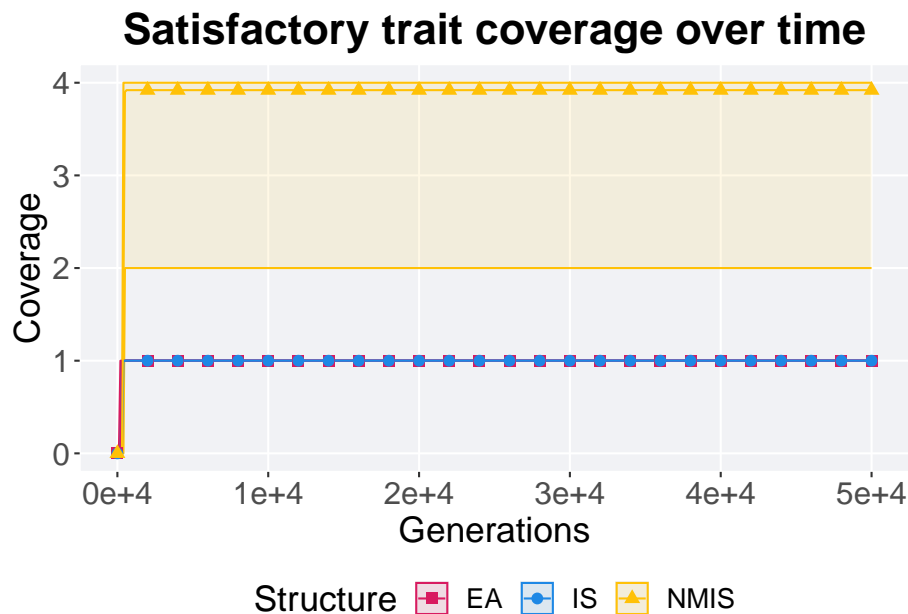
#### 8.3.1.1 Coverage over time

Satisfactory trait coverage over time.

```
lines = filter(mi50_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` ==
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_sat_cov),
    mean = mean(pop_sat_cov),
    max = max(pop_sat_cov)
  )

## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.

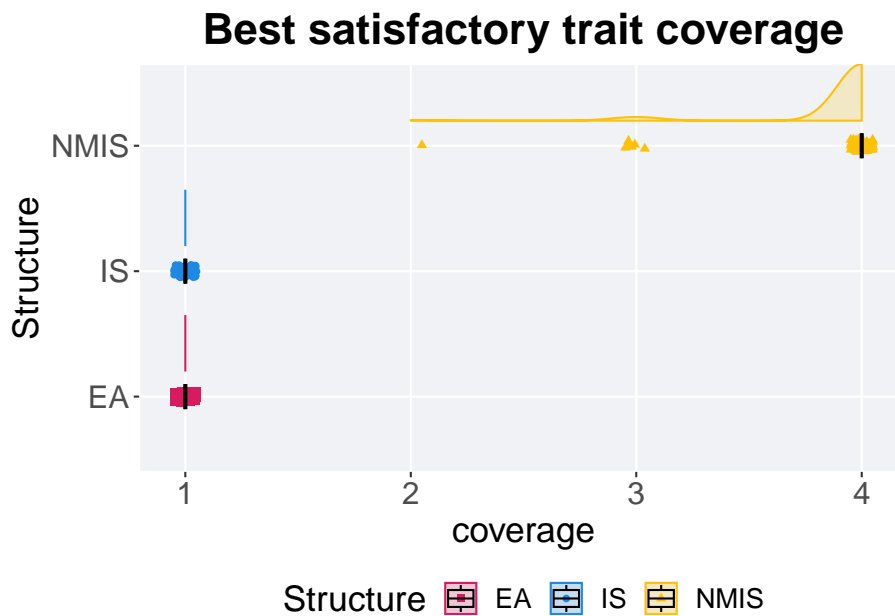
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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
```



#### 8.3.1.2 Best coverage throughout

Best satisfactory trait coverage throughout 50,000 generations.

```
### best satisfactory trait coverage throughout
filter(mi50_best, Diagnostic == 'CONTRADICTION OBJECTIVES' & `Selection\nScheme` == 'T
  ggplot(., aes(x = Structure, y = VAL, color = Structure, fill = Structure, shape = S
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha =
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory trait coverage') +
  p_theme + coord_flip()
```



#### 8.3.1.2.1 Stats

Summary statistics for the best satisfactory trait coverage.

```
### best
coverage = filter(mi50_best, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` == 'T
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE),
    median = median(VAL, na.rm = TRUE),
    mean = mean(VAL, na.rm = TRUE),
    max = max(VAL, na.rm = TRUE),
    IQR = IQR(VAL, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA         100     0     1     1     1     1     0
## 2 IS         100     0     1     1     1     1     0
## 3 NMIS       100     0     2     4    3.92    4     0
```

Kruskal–Wallis test provides evidence of difference among satisfactory trait

coverage.

```
kruskal.test(VAL ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: VAL by Structure
## Kruskal-Wallis chi-squared = 295.79, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage.

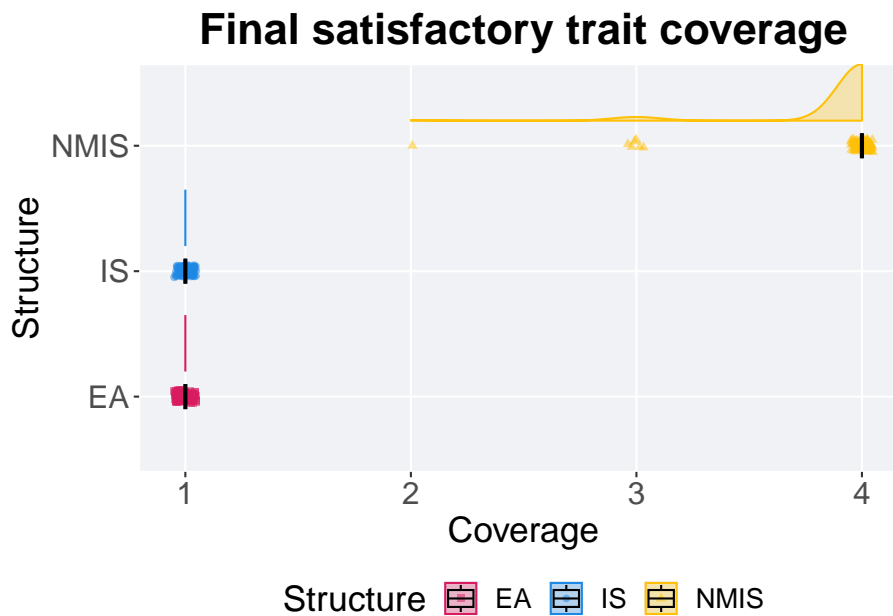
```
pairwise.wilcox.test(x = coverage$VAL, g = coverage$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$VAL and coverage$Structure
##
##      EA      IS
## IS    1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

### 8.3.1.3 End of 50,000 generations

Satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
filter(mi50_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'Final') %>%
  ggplot(., aes(x = Structure, y = pop_sat_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final satisfactory trait coverage') +
  p_theme + coord_flip()
```



#### 8.3.1.3.1 Stats

Summary statistics for satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
coverage = filter(mi50_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme`
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_sat_cov)),
    min = min(pop_sat_cov, na.rm = TRUE),
    median = median(pop_sat_cov, na.rm = TRUE),
    mean = mean(pop_sat_cov, na.rm = TRUE),
    max = max(pop_sat_cov, na.rm = TRUE),
    IQR = IQR(pop_sat_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt  min median  mean  max  IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     2     4    3.92    4     0
```

Kruskal–Wallis test provides evidence of difference among satisfactory trait coverage in the population at the end of 50,000 generations.

```
kruskal.test(pop_sat_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_sat_cov by Structure
## Kruskal-Wallis chi-squared = 295.79, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage in the population at the end of 50,000 generations.

```
pairwise.wilcox.test(x = coverage$pop_sat_cov, g = coverage$Structure, p.adjust.method =
                    paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_sat_cov and coverage$Structure
##
##      EA      IS
## IS   1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

### 8.3.2 Activation gene coverage

Activation gene coverage analysis.

#### 8.3.2.1 Coverage over time

Activation gene coverage over time.

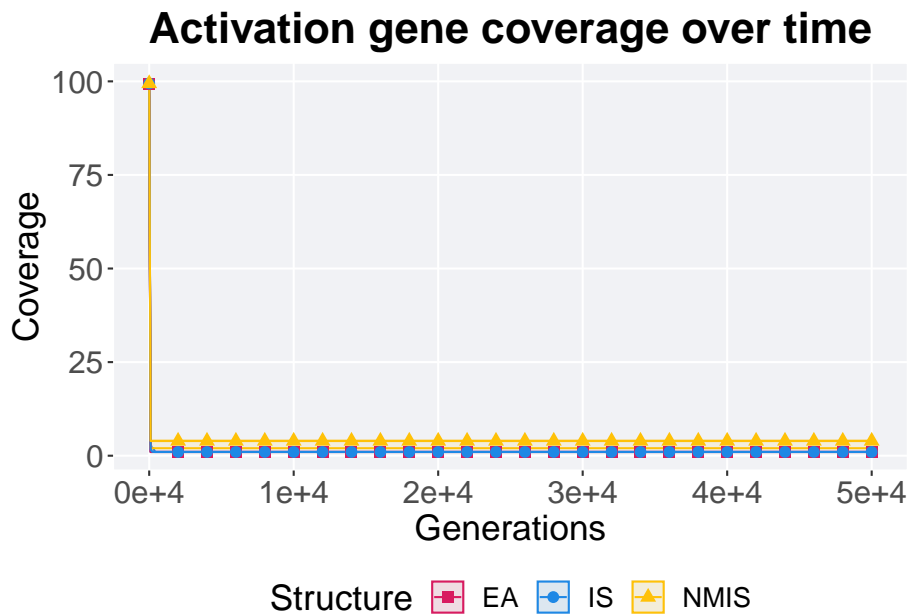
```
# data for lines and shading on plots
lines = filter(mi50_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection` == 'n')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )
```

```
## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.
```

```

ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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

```



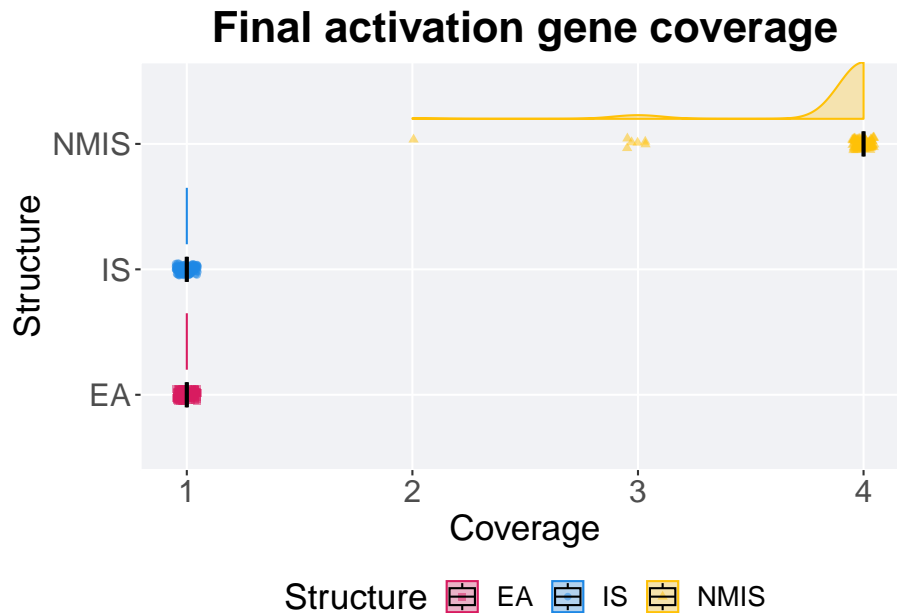
### 8.3.2.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```

### end of run
filter(mi50_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` =
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, sl
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha =
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme + coord_flip()

```



#### 8.3.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(mi50_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection
coverage %>%
  group_by(Structure) %>%

```



```
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(pop_act_cov)),
  min = min(pop_act_cov, na.rm = TRUE),
  median = median(pop_act_cov, na.rm = TRUE),
  mean = mean(pop_act_cov, na.rm = TRUE),
  max = max(pop_act_cov, na.rm = TRUE),
  IQR = IQR(pop_act_cov, na.rm = TRUE)
)
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     2     4    3.92     4     0
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 295.79, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

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

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS    1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 8.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the contradictory objectives diagnostic.

### 8.4.1 Satisfactory trait coverage

Satisfactory trait coverage analysis.

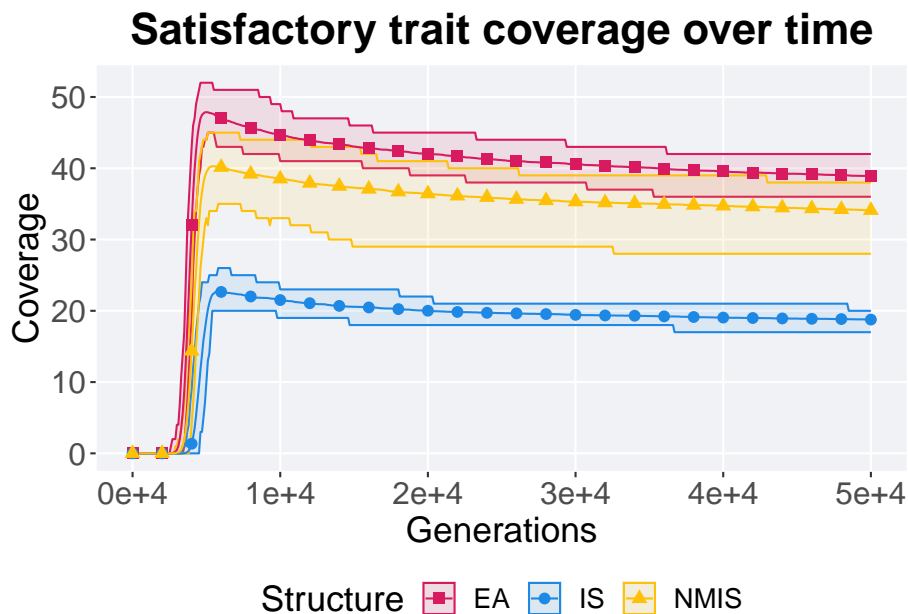
#### 8.4.1.1 Coverage over time

Satisfactory trait coverage over time.

```
lines = filter(mi50_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\n
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_sat_cov),
    mean = mean(pop_sat_cov),
    max = max(pop_sat_cov)
  )

## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.

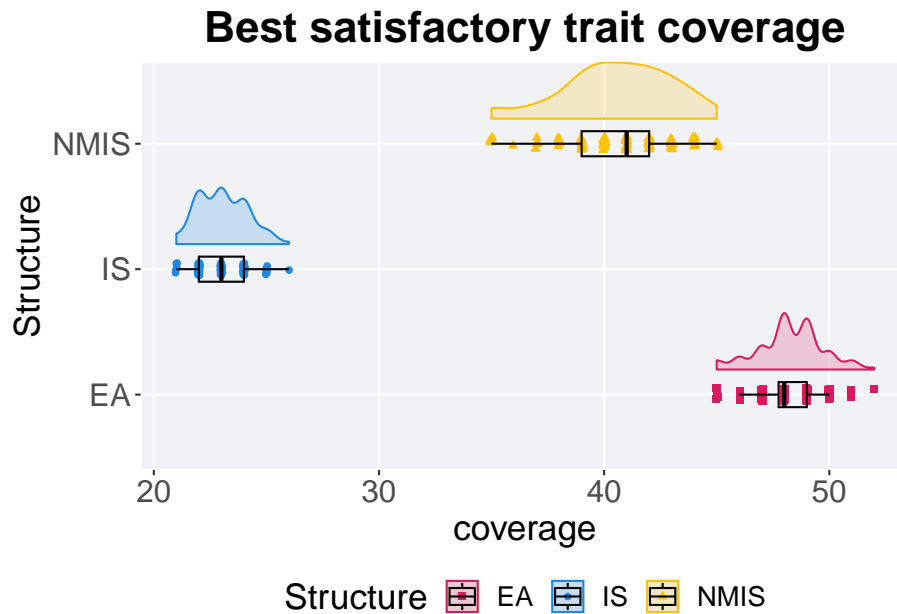
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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
```



#### 8.4.1.2 Best coverage throughout

Best satisfactory trait coverage throughout 50,000 generations.

```
### best satisfactory trait coverage throughout
filter(mi50_best, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE' &
  ggplot(., aes(x = Structure, y = VAL, color = Structure, fill = Structure, shape = Structure))
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory trait coverage') +
  p_theme + coord_flip()
```



#### 8.4.1.2.1 Stats

Summary statistics for the best satisfactory trait coverage.

```
### best
coverage = filter(mi50_best, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nSc
coverage$Structure = factor(coverage$Structure, levels=c('EA','NMIS','IS'))
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE),
    median = median(VAL, na.rm = TRUE),
    mean = mean(VAL, na.rm = TRUE),
    max = max(VAL, na.rm = TRUE),
    IQR = IQR(VAL, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0    45    48  48.2    52  1.25
## 2 NMIS        100     0    35    41  40.6    45    3
## 3 IS          100     0    21    23  23.0    26    2
```

Kruskal-Wallis test provides evidence of difference among satisfactory trait coverage.

```
kruskal.test(VAL ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: VAL by Structure
## Kruskal-Wallis chi-squared = 267.02, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage.

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

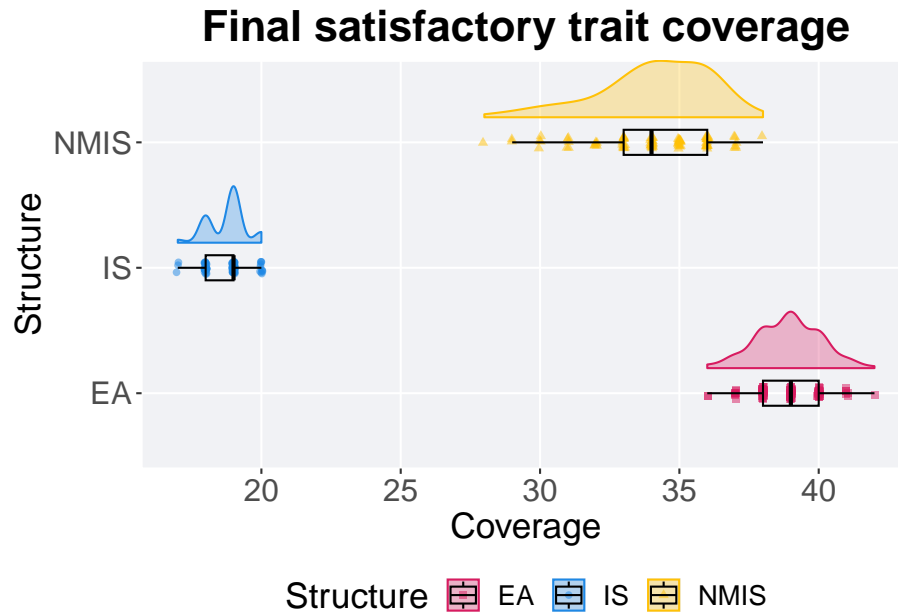
```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$VAL and coverage$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

#### 8.4.1.3 End of 50,000 generations

Satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
filter(mi50_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE')
ggplot(., aes(x = Structure, y = pop_sat_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final satisfactory trait coverage') +
```

```
p_theme + coord_flip()
```



#### 8.4.1.3.1 Stats

Summary statistics for satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
coverage = filter(mi50_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection` == 'MI50')
coverage$Structure = factor(coverage$Structure, levels=c('EA', 'NMIS', 'IS'))
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_sat_cov)),
    min = min(pop_sat_cov, na.rm = TRUE),
    median = median(pop_sat_cov, na.rm = TRUE),
    mean = mean(pop_sat_cov, na.rm = TRUE),
    max = max(pop_sat_cov, na.rm = TRUE),
    IQR = IQR(pop_sat_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0    36    39  38.9    42     2
```

## 2 NMIS	100	0	28	34	34.1	38	3
## 3 IS	100	0	17	19	18.8	20	1

Kruskal–Wallis test provides evidence of difference among satisfactory trait coverage in the population at the end of 50,000 generations.

```
kruskal.test(pop_sat_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_sat_cov by Structure
## Kruskal-Wallis chi-squared = 266.88, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage in the population at the end of 50,000 generations.

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

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_sat_cov and coverage$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 8.4.2 Activation gene coverage

Activation gene coverage analysis.

### 8.4.2.1 Coverage over time

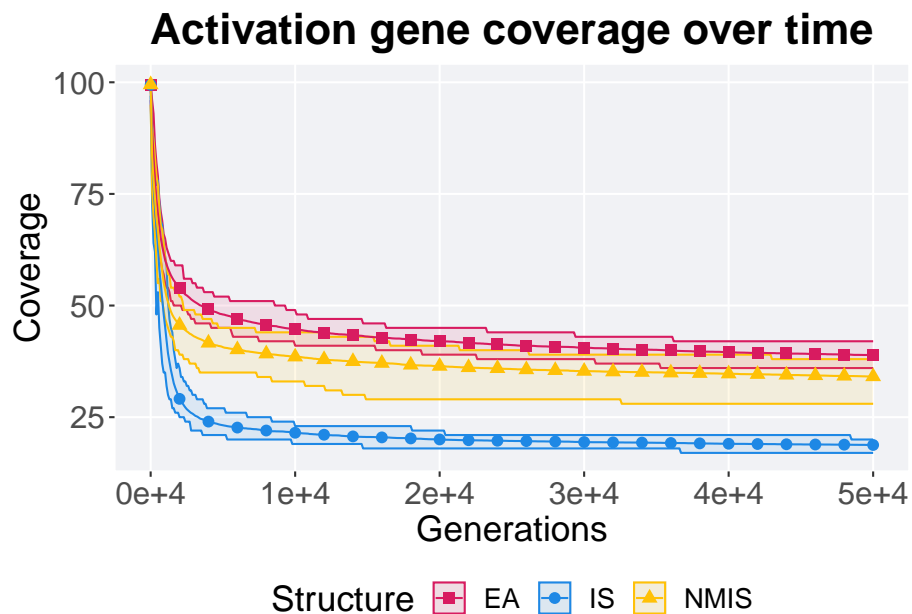
Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(mi50_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` ==
              group_by(Structure, Generations) %>%
              dplyr::summarise(
                min = min(pop_act_cov),
                mean = mean(pop_act_cov),
                max = max(pop_act_cov)
              )
```

## `summarise()` has grouped output by 'Structure'. You can override using the

## ``groups`` argument.

```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
    scale_y_continuous(
      name="Coverage"
    ) +
    scale_x_continuous(
      name="Generations",
      limits=c(0, 50000),
      breaks=c(0, 10000, 20000, 30000, 40000, 50000),
      labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
    ) +
    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
```



#### 8.4.2.2 End of 50,000 generations

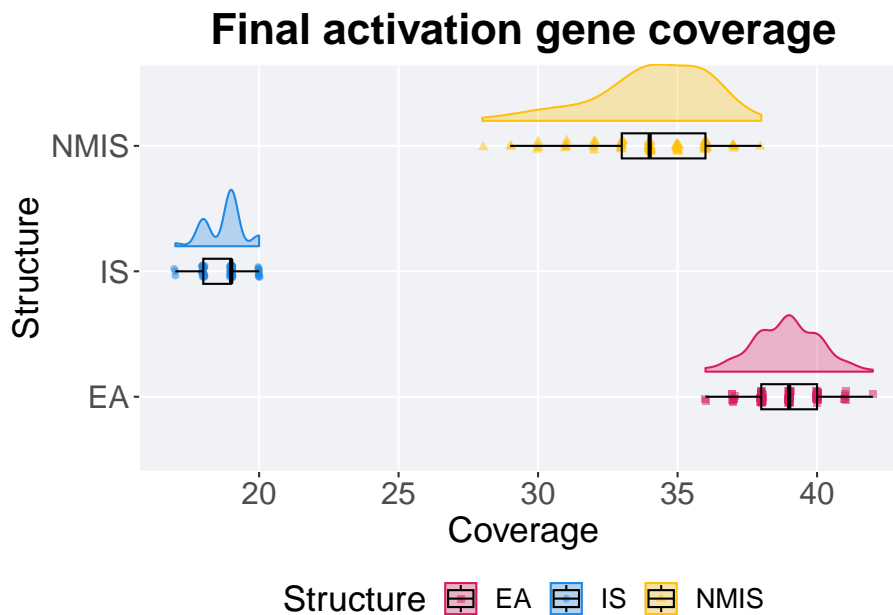
Activation gene coverage in the population at the end of 50,000 generations.



```

### end of run
filter(mi50_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE') %>%
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme + coord_flip()

```



#### 8.4.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(mi50_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE') %>%
  coverage$Structure = factor(coverage$Structure, levels=c('EA', 'NMIS', 'IS'))
coverage %>%

```

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

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0    36    39  38.9    42     2
## 2 NMIS         100     0    28    34  34.1    38     3
## 3 IS          100     0    17    19  18.8    20     1
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 266.88, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```
pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method =
  paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_act_cov and coverage$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## Chapter 9

# MI50: Multi-path exploration results

Here we present the results for the **best performances** and **activation gene coverage** generated by each selection scheme replicate on the multi-path exploration diagnostic with configurations presented below. For our the configuration of these experiments, we execute migrations every 50 generations and there are 4 islands in a ring topology. Best performance found refers to the largest average trait score found in a given population. Note that activation gene coverage values are gathered at the population-level. Activation gene coverage refers to the count of unique activation genes in a given population; this gives us a range of integers between 0 and 100.

### 9.1 Analysis dependencies

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

### 9.2 Truncation selection

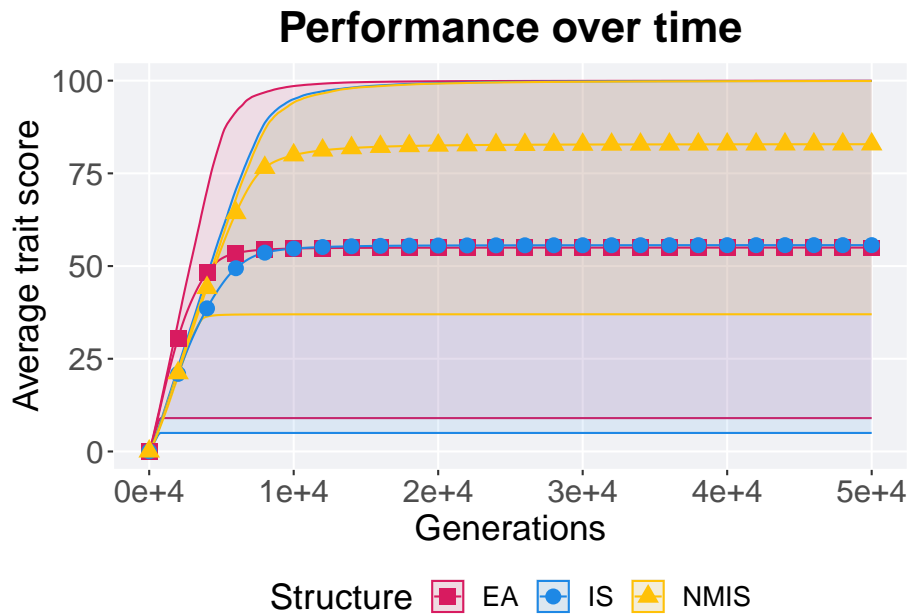
Here we analyze how the different population structures affect truncation selection (size 8) on the contradictory objectives diagnostic.

### 9.2.1 Performance

```

lines = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nSch
group_by(Structure, Generations) %>%
dplyr::summarise(
  min = min(pop_fit_max) / DIMENSIONALITY,
  mean = mean(pop_fit_max) / DIMENSIONALITY,
  max = max(pop_fit_max) / DIMENSIONALITY
)
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
geom_line(size = 0.5) +
geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0,
scale_y_continuous(
  name="Average trait score"
) +
scale_x_continuous(
  name="Generations",
  limits=c(0, 50000),
  breaks=c(0, 10000, 20000, 30000, 40000, 50000),
  labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
) +
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle("Performance over time") +
p_theme

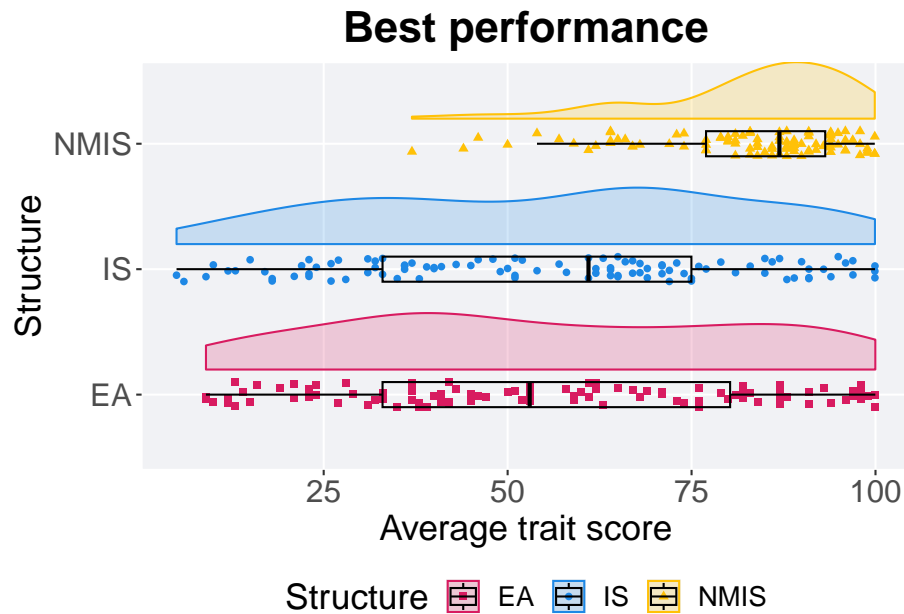
```



#### 9.2.1.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi50_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TRUNCATION' & V
ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Structure, sha
geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
scale_y_continuous(
  name="Average trait score"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Best performance')+
p_theme + coord_flip()
```



#### 9.2.1.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi50_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nSc
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE) / DIMENSIONALITY,
    median = median(VAL, na.rm = TRUE) / DIMENSIONALITY,
    mean = mean(VAL, na.rm = TRUE) / DIMENSIONALITY,
    max = max(VAL, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VAL, na.rm = TRUE) / DIMENSIONALITY
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0  9.00  53.0  55.0 100.  47.2
## 2 IS          100     0   5    61.0  55.6  99.9  42.0
## 3 NMIS        100     0 37.0   86.9  82.9  99.9  16.2
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VA ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  VAL by Structure
## Kruskal-Wallis chi-squared = 67.87, df = 2, p-value = 1.829e-15
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

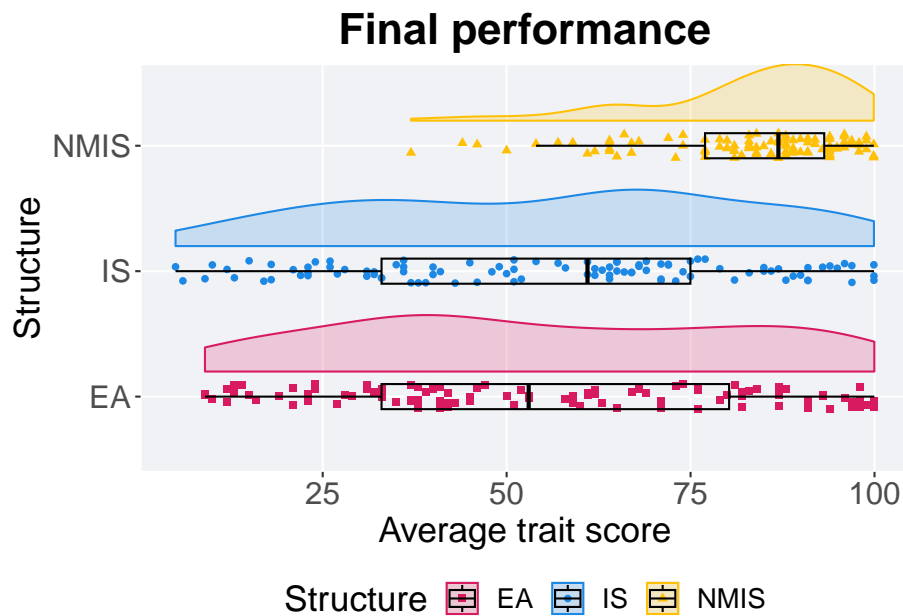
```
pairwise.wilcox.test(x = performance$VAL, g = performance$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VAL and performance$Structure
##
##      EA      IS
## IS   1      -
## NMIS 3.0e-12 7.9e-13
##
## P value adjustment method: bonferroni
```

### 9.2.1.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TRUNCATION') +
  ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fill = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final performance') +
  p_theme + coord_flip()
```



#### 9.2.1.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection` == 'Selection')
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_fit_max)),
    min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0  9.00  53.0  55.0 100.  47.2
## 2 IS          100     0   5    61.0  55.6  99.9  42.0
## 3 NMIS        100     0 37.0   86.9  82.9  99.9  16.2
```

Kruskal–Wallis test provides evidence of difference among selection schemes.



```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 67.87, df = 2, p-value = 1.829e-15
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

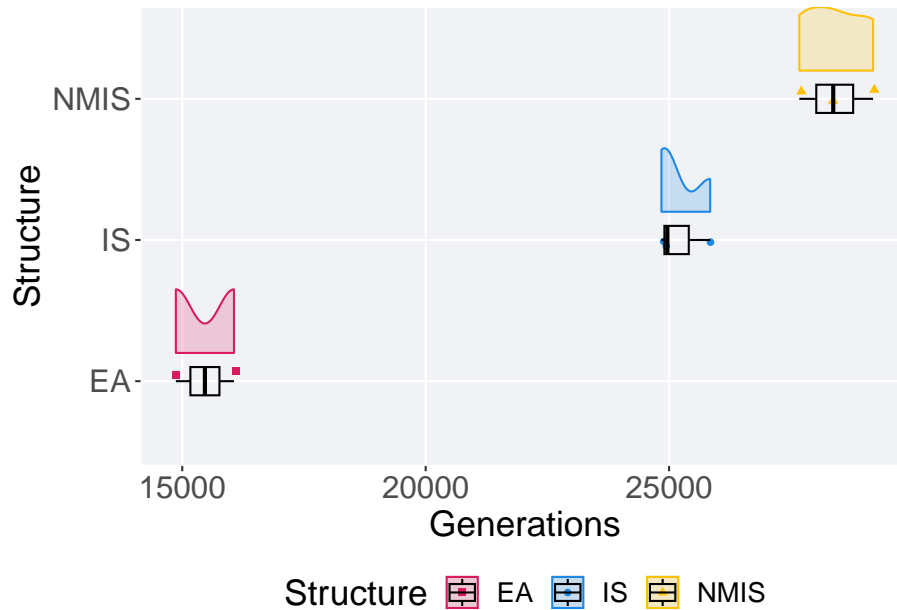
```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.method = "b"
                    paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$pop_fit_max and performance$Structure
##
##      EA      IS
## IS   1      -
## NMIS 3.0e-12 7.9e-13
##
## P value adjustment method: bonferroni
```

### 9.2.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi50_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TRUNCATION' & Gen
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Stru
    geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
    geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
    geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
    scale_shape_manual(values=SHAPE)+
    scale_y_continuous(
      name="Generations"
    ) +
    scale_x_discrete(
      name="Structure"
    ) +
    scale_colour_manual(values = cb_palette) +
    scale_fill_manual(values = cb_palette) +
    p_theme + coord_flip()
```



### 9.2.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(mi50_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          2      0 14868 15465 15465 16062  597
## 2 IS          3      0 24843 24965 25217. 25844  500.
## 3 NMIS        3      0 27675 28372 28412. 29190  758.
```

Kruskal–Wallis test provides evidence of no difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)

##
## Kruskal-Wallis rank sum test
##
## data: Generations by Structure
## Kruskal-Wallis chi-squared = 6.25, df = 2, p-value = 0.04394
```

### 9.2.3 Activation gene coverage

Activation gene coverage analysis.

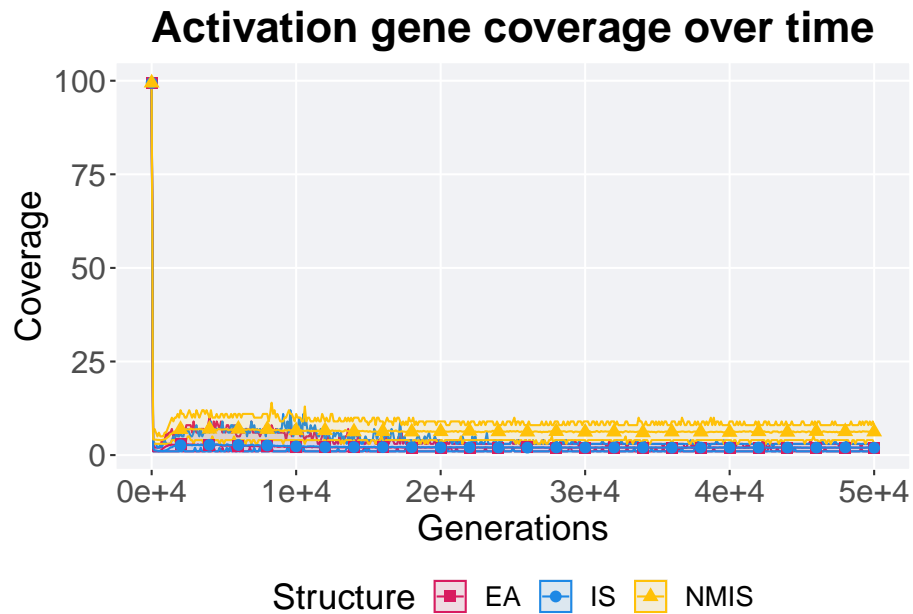
#### 9.2.3.1 Coverage over time

Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TF
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )

## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.

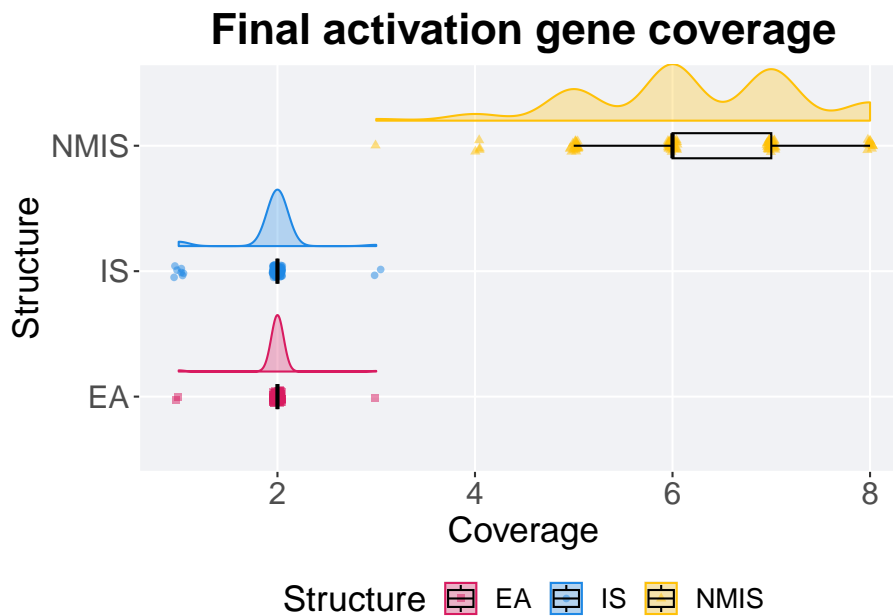
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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
```



#### 9.2.3.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```
### end of run
filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, size = 1.5, alpha = 0.5)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme + coord_flip()
```



#### 9.2.3.2.1 Stats

Summary statistics for activation gene coverage.

```
coverage = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_act_cov)),
    min = min(pop_act_cov, na.rm = TRUE),
    median = median(pop_act_cov, na.rm = TRUE),
    mean = mean(pop_act_cov, na.rm = TRUE),
    max = max(pop_act_cov, na.rm = TRUE),
    IQR = IQR(pop_act_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA         100     0     1     2  1.99     3     0
## 2 IS         100     0     1     2  1.95     3     0
## 3 NMIS       100     0     3     6  6.23     8     1
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 265.48, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```
pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method =
paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS   1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 9.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the contradictory objectives diagnostic.

### 9.3.1 Performance

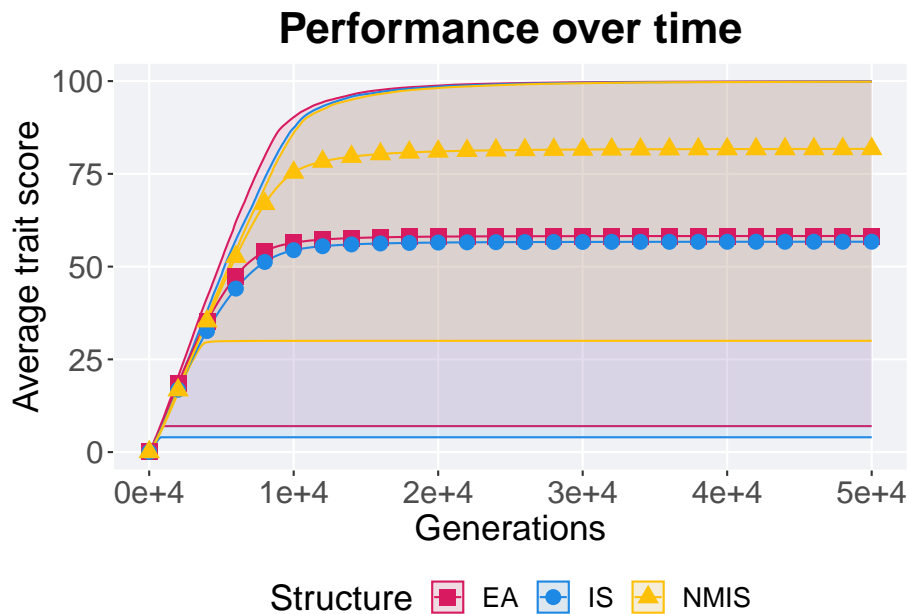
#### 9.3.1.1 Performance over time

```
lines = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection` ~ Sch
group_by(Structure, Generations) %>%
dplyr::summarise(
  min = min(pop_fit_max) / DIMENSIONALITY,
  mean = mean(pop_fit_max) / DIMENSIONALITY,
  max = max(pop_fit_max) / DIMENSIONALITY
)
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
geom_line(size = 0.5) +
geom_point(data = filter(lines, Generations %>= 2000 == 0), size = 2.5, stroke = 2.0,
scale_y_continuous(
```

```

    name="Average trait score"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme

```



#### 9.3.1.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```

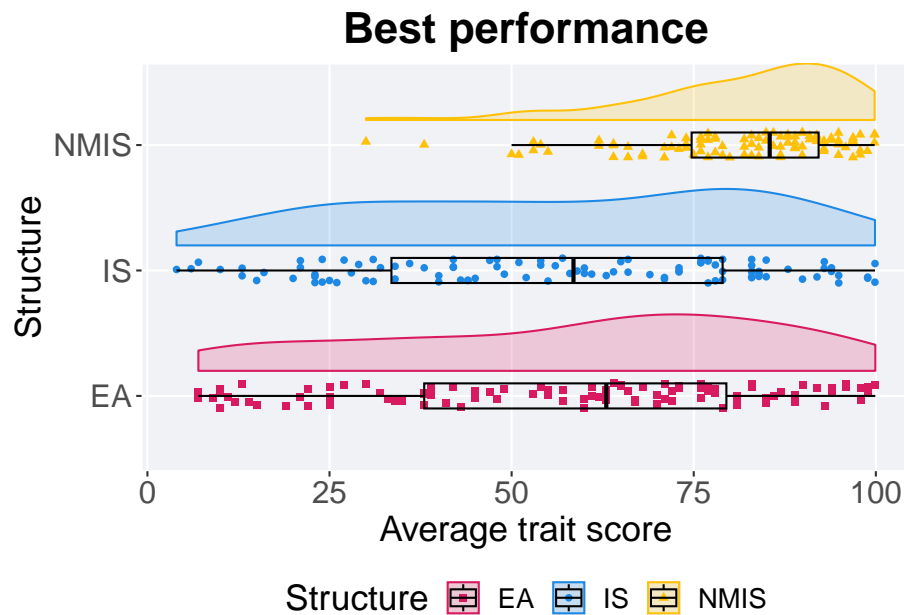
filter(mi50_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT' & V
  ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Structure, sha
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +

```

```

scale_y_continuous(
  name="Average trait score"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Best performance')+
p_theme + coord_flip()

```



#### 9.3.1.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

performance = filter(mi50_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nSc
performance %>%
group_by(Structure) %>%
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(VAL)),
  min = min(VAL, na.rm = TRUE) / DIMENSIONALITY,
  median = median(VAL, na.rm = TRUE) / DIMENSIONALITY,
  mean = mean(VAL, na.rm = TRUE) / DIMENSIONALITY,

```



```

    max = max(VAL, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VAL, na.rm = TRUE) / DIMENSIONALITY
  )

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0  7.00  63.0  58.2  99.9  41.5
## 2 IS          100     0   4    58.5  56.7  99.9  45.5
## 3 NMIS        100     0 30.0   85.4  81.7  99.8  17.4

```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VAL ~ Structure, data = performance)
```

```

##
##  Kruskal-Wallis rank sum test
##
## data:  VAL by Structure
## Kruskal-Wallis chi-squared = 56.11, df = 2, p-value = 6.546e-13

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

```

##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VAL and performance$Structure
##
##      EA      IS
## IS    1      -
## NMIS 4.2e-10 5.1e-11
##
## P value adjustment method: bonferroni

```

### 9.3.1.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```

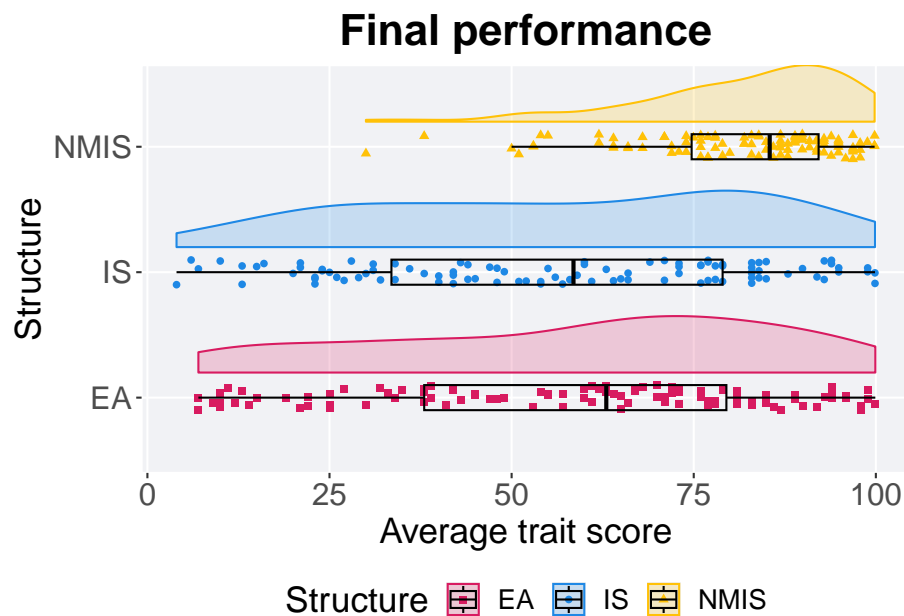
filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT')
ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fill = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  )

```

```

) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Final performance')+
p_theme + coord_flip()

```



### 9.3.1.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

performance = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection` == 'Satisfactory')
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_fit_max)),
    min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
  )

```

```
)
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int>  <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA         100      0  7.00  63.0  58.2  99.9  41.5
## 2 IS         100      0  4      58.5  56.7  99.9  45.5
## 3 NMIS       100      0 30.0   85.4  81.7  99.8  17.4
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 56.11, df = 2, p-value = 6.546e-13
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.method = "b",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$pop_fit_max and performance$Structure
##
##      EA      IS
## IS    1      -
## NMIS 4.2e-10 5.1e-11
##
## P value adjustment method: bonferroni
```

### 9.3.2 Generation satisfactory solution found

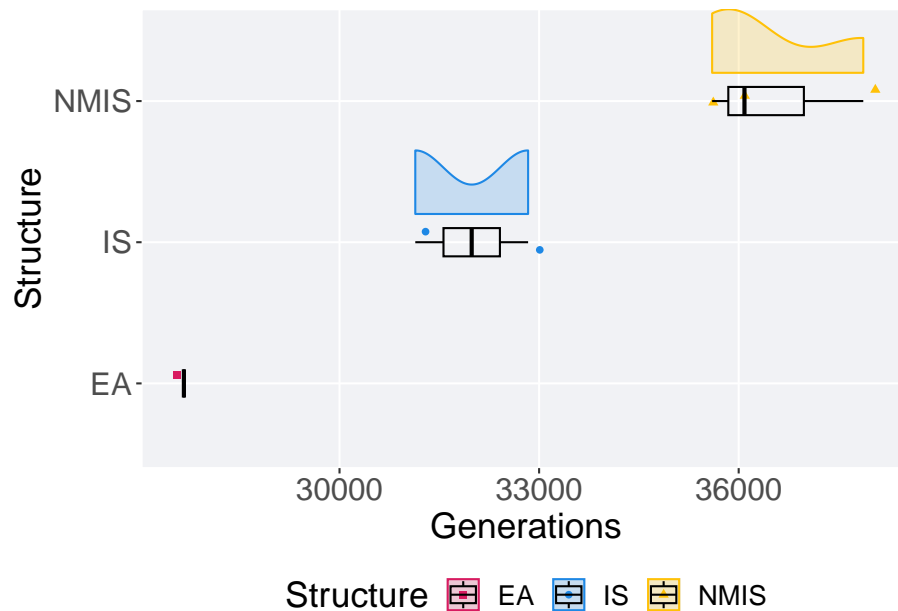
First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi50_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT') & Ger
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Generations"
```

```

) +
scale_x_discrete(
  name="Structure"
) +
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
p_theme + coord_flip()

```



### 9.3.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(mi50_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )

```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          1     0 27661 27661 27661 27661     0
## 2 IS          2     0 31139 31987 31987 32835   848
## 3 NMIS        3     0 35601 36087 36520. 37873  1136
```

Kruskal-Wallis test provides evidence of no difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  Generations by Structure
## Kruskal-Wallis chi-squared = 4.2857, df = 2, p-value = 0.1173
pairwise.wilcox.test(x = ssf$Generations, g = ssf$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum exact test
##
## data:  ssf$Generations and ssf$Structure
##
##      EA   IS
## IS  1.00  -
## NMIS 0.75 0.30
##
## P value adjustment method: bonferroni
```

### 9.3.3 Activation gene coverage

Activation gene coverage analysis.

#### 9.3.3.1 Coverage over time

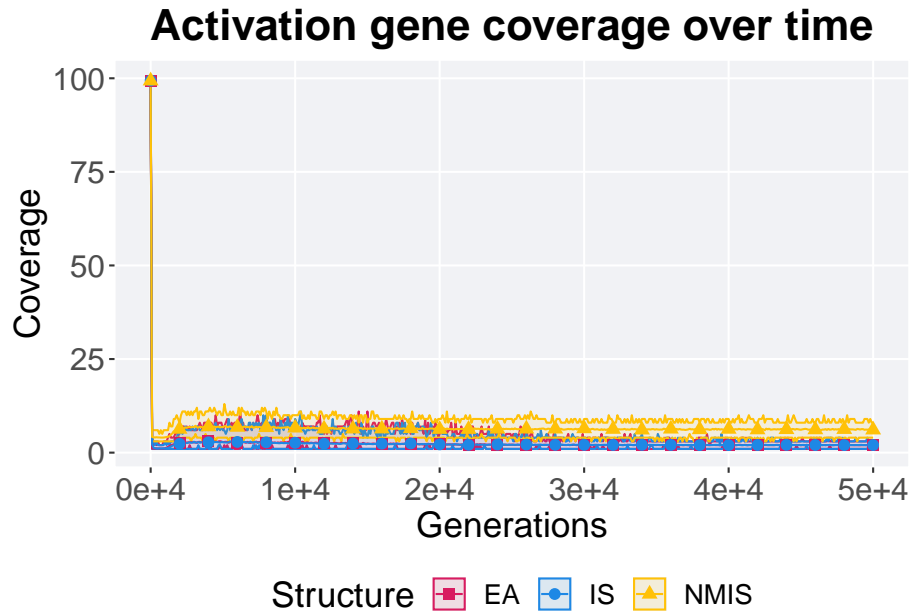
Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )
```

## `summarise()` has grouped output by 'Structure'. You can override using the

## ``groups`` argument.

```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
    scale_y_continuous(
      name="Coverage"
    ) +
    scale_x_continuous(
      name="Generations",
      limits=c(0, 50000),
      breaks=c(0, 10000, 20000, 30000, 40000, 50000),
      labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
    ) +
    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
```



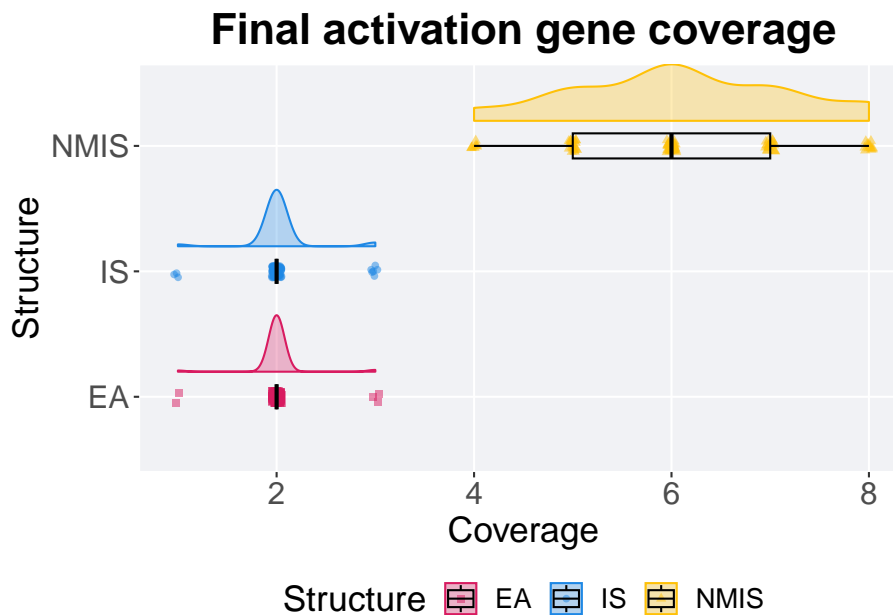
### 9.3.3.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```

### end of run
filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT')
ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme + coord_flip()

```



#### 9.3.3.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT')
coverage %>%
  group_by(Structure) %>%

```

```
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(pop_act_cov)),
  min = min(pop_act_cov, na.rm = TRUE),
  median = median(pop_act_cov, na.rm = TRUE),
  mean = mean(pop_act_cov, na.rm = TRUE),
  max = max(pop_act_cov, na.rm = TRUE),
  IQR = IQR(pop_act_cov, na.rm = TRUE)
)
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     2  2.01     3     0
## 2 IS          100     0     1     2  2.03     3     0
## 3 NMIS        100     0     4     6  6.09     8     2
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 262.68, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```
pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method =
  paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS  0.88    -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

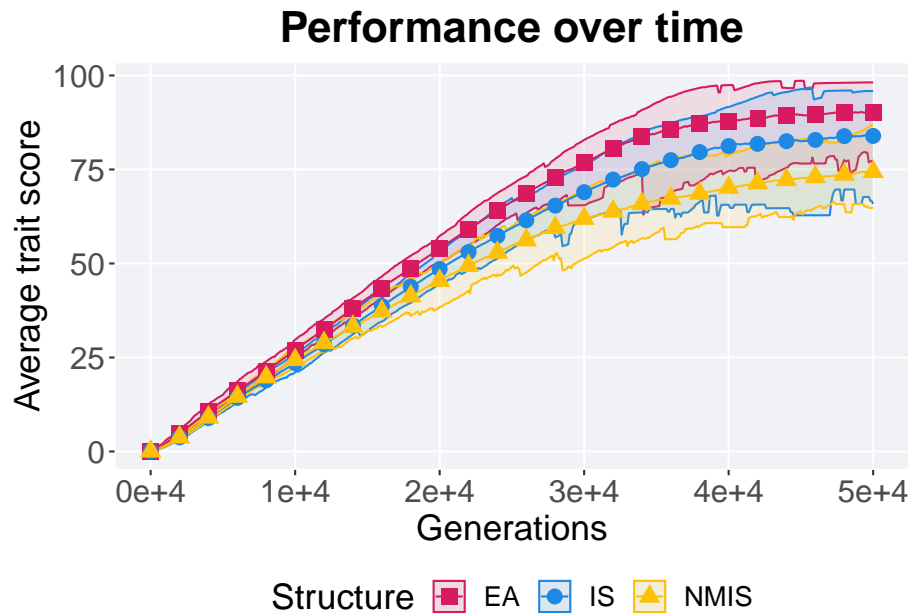


## 9.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the contradictory objectives diagnostic.

### 9.4.1 Performance

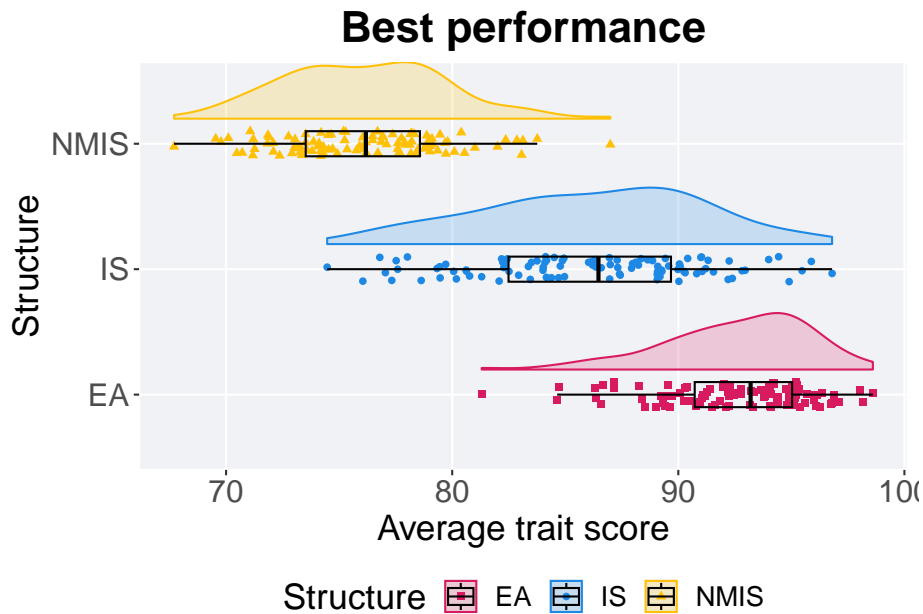
```
lines = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'LE
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme
```



#### 9.4.1.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi50_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'LEXI
  ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Stru
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best performance')+
  p_theme + coord_flip()
```



#### 9.4.1.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi50_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'I
performance$Structure = factor(performance$Structure, levels=c('EA', 'NMIS', 'IS'))
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE) / DIMENSIONALITY,
    median = median(VAL, na.rm = TRUE) / DIMENSIONALITY,
    mean = mean(VAL, na.rm = TRUE) / DIMENSIONALITY,
    max = max(VAL, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VAL, na.rm = TRUE) / DIMENSIONALITY
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA         100     0  81.3  93.2  92.6  98.6  4.31
## 2 NMIS       100     0  67.7  76.2  76.1  87.0  5.05
## 3 IS         100     0  74.5  86.5  86.1  96.8  7.18
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VAL ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: VAL by Structure
## Kruskal-Wallis chi-squared = 217.42, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

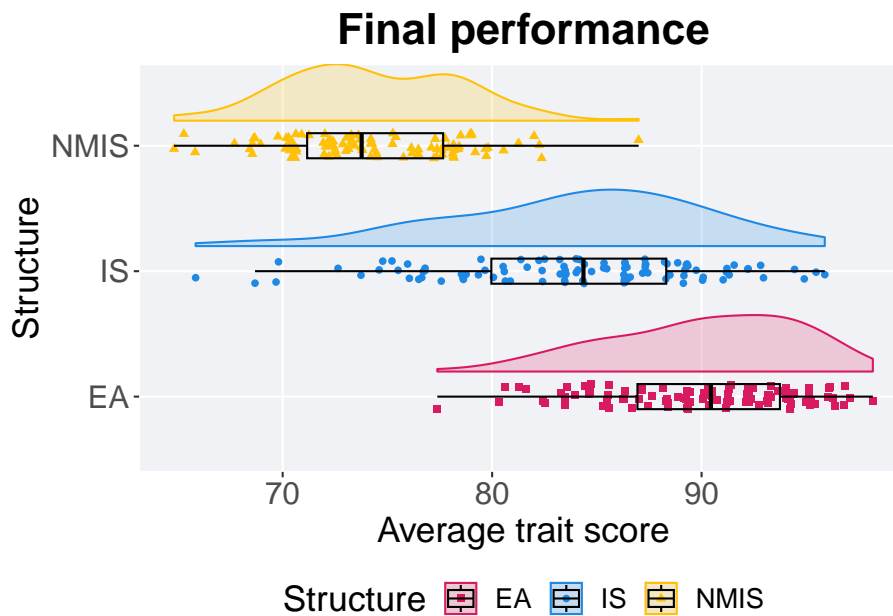
```
pairwise.wilcox.test(x = performance$VAL, g = performance$Structure, p.adjust.method =
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$VAL and performance$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 1
##
## P value adjustment method: bonferroni
```

### 9.4.1.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
       ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fi
       geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
       geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
       geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
       scale_y_continuous(
         name="Average trait score"
       ) +
       scale_x_discrete(
         name="Structure"
       ) +
       scale_shape_manual(values=SHAPE) +
       scale_colour_manual(values = cb_palette, ) +
       scale_fill_manual(values = cb_palette) +
       ggtitle('Final performance') +
       p_theme + coord_flip()
```



#### 9.4.1.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme`
performance$Structure = factor(performance$Structure, levels=c('EA', 'NMIS', 'IS'))
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_fit_max)),
    min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA         100     0  77.4  90.4  90.1  98.2  6.80
## 2 NMIS       100     0  64.8  73.8  74.4  87.0  6.49
## 3 IS         100     0  65.8  84.4  83.9  95.9  8.35
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 186.92, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.m = 'p.adjust.method'
paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$pop_fit_max and performance$Structure
##
## EA NMIS
## NMIS < 2e-16 -
## IS 2.5e-12 1
##
## P value adjustment method: bonferroni
```

## 9.4.2 Activation gene coverage

Activation gene coverage analysis.

### 9.4.2.1 Coverage over time

Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection`\nSch
group_by(Structure, Generations) %>%
dplyr::summarise(
  min = min(pop_act_cov),
  mean = mean(pop_act_cov),
  max = max(pop_act_cov)
)
```

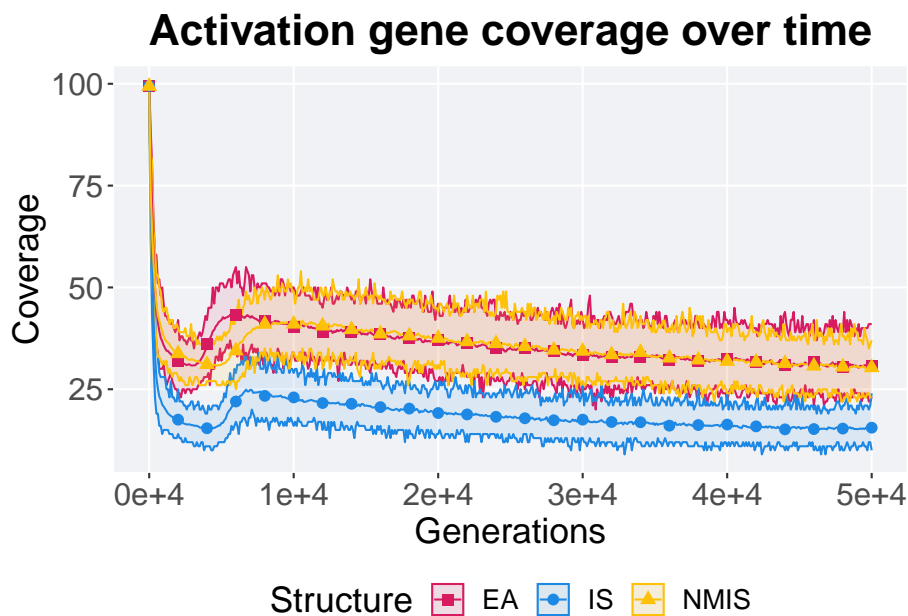
```
## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.
```

```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
geom_line(size = 0.5) +
geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
```

```

scale_y_continuous(
  name="Coverage"
) +
scale_x_continuous(
  name="Generations",
  limits=c(0, 50000),
  breaks=c(0, 10000, 20000, 30000, 40000, 50000),
  labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
) +
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

```



#### 9.4.2.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```

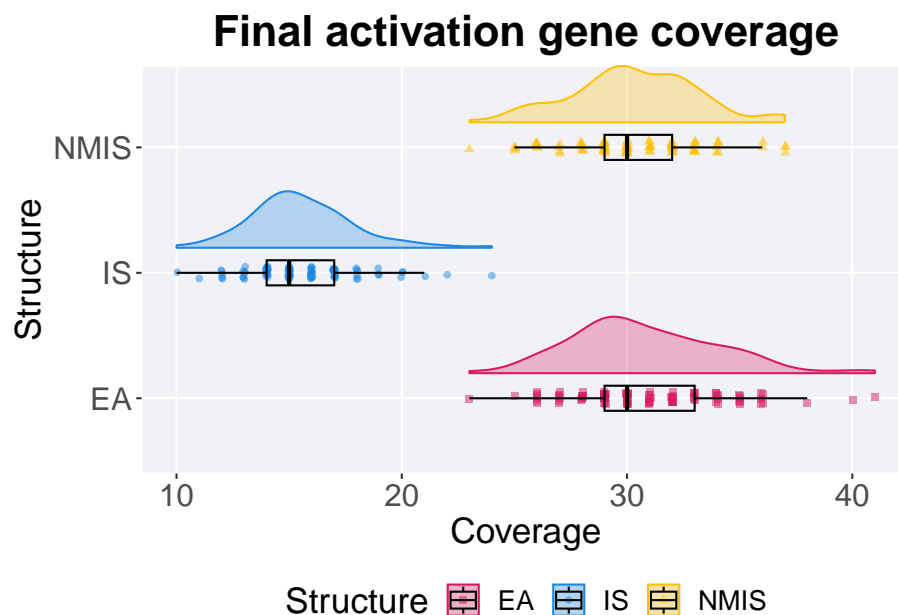
### end of run
filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'LEXICASE')
ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +

```

```

geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
scale_shape_manual(values=SHAPE) +
scale_y_continuous(
  name="Coverage"
) +
scale_x_discrete(
  name="Structure"
) +
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle('Final activation gene coverage') +
p_theme + coord_flip()

```



#### 9.4.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(mi50_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection` == 'n')
coverage$Structure = factor(coverage$Structure, levels=c('EA', 'NMIS', 'IS'))
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_act_cov)),
    min = min(pop_act_cov, na.rm = TRUE),

```



```

median = median(pop_act_cov, na.rm = TRUE),
mean = mean(pop_act_cov, na.rm = TRUE),
max = max(pop_act_cov, na.rm = TRUE),
IQR = IQR(pop_act_cov, na.rm = TRUE)
)

```

```

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0    23    30  30.8    41     4
## 2 NMIS         100     0    23    30  30.3    37     3
## 3 IS          100     0    10    15  15.6    24     3

```

Kruskal-Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```

##
## Kruskal-Wallis rank sum test
##
## data:  pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 200.36, df = 2, p-value < 2.2e-16

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```

pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')

```

```

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_act_cov and coverage$Structure
##
##      EA      NMIS
## NMIS 0.81    -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni

```



## Chapter 10

# MI5000: Exploitation rate results

Here we present the results for **best performances** found by each selection scheme replicate on the exploitation rate diagnostic with configurations presented below. For our the configuration of these experiments, we execute migrations every 50 generations and there are 4 islands in a ring topology. When migrations occur, we swap two individuals (same position on each island) and guarantee that no solution can return to the same island. Best performance found refers to the largest average trait score found in a given population. Note that performance values fall between 0.0 and 100.0.

### 10.1 Analysis dependencies

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

### 10.2 Truncation selection

Here we analyze how the different population structures affect truncation selection (size 8) on the exploitation rate diagnostic.

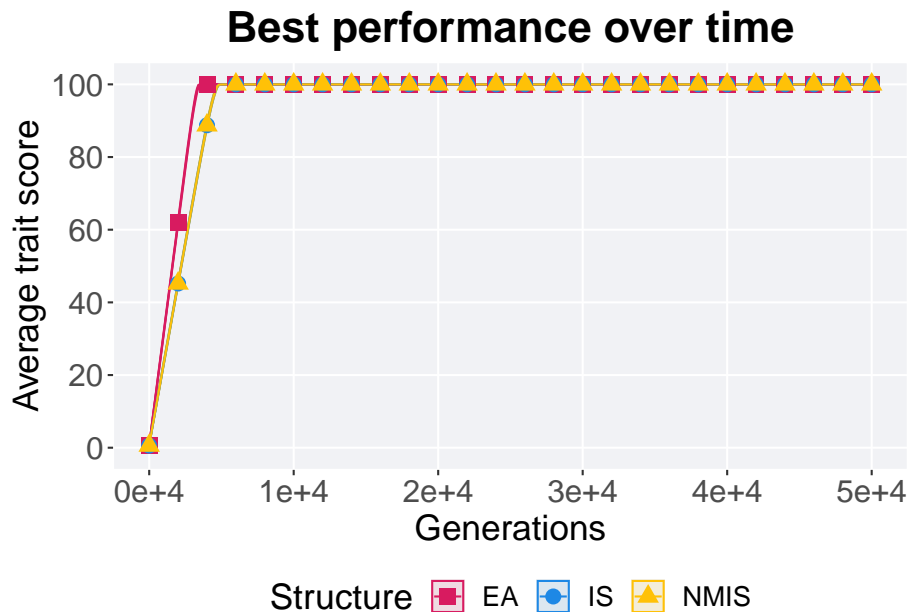
#### 10.2.1 Performance over time

```
lines = filter(mi5000_over_time, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TRUNCATION') %>%
  group_by(Structure, Generations) %>%
```

```

dplyr::summarise(
  min = min(pop_fit_max) / DIMENSIONALITY,
  mean = mean(pop_fit_max) / DIMENSIONALITY,
  max = max(pop_fit_max) / DIMENSIONALITY
)
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0,
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Best performance over time") +
  p_theme

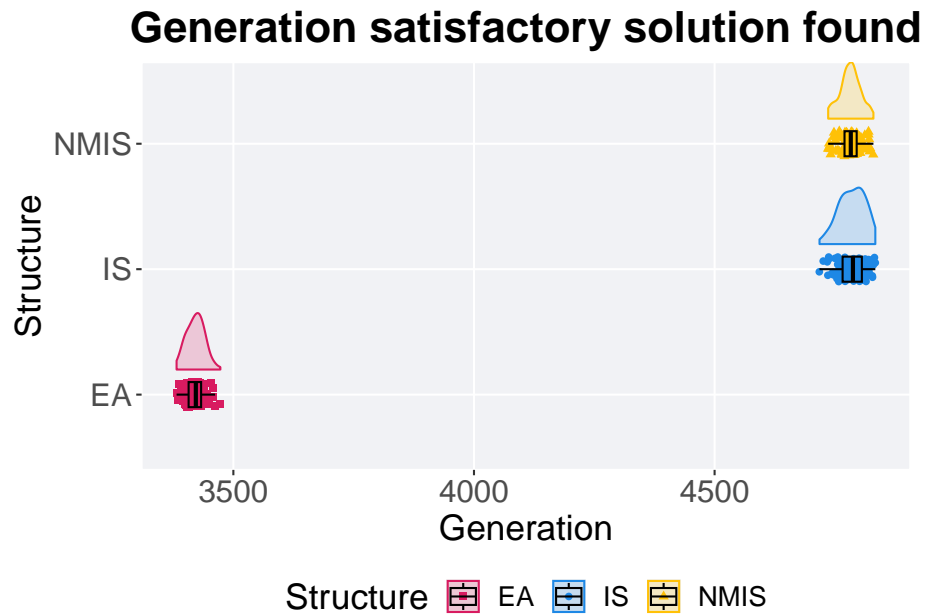
```



### 10.2.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi5000_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TRUNCATION') %>%
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Generation"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Generation satisfactory solution found') +
  p_theme + coord_flip()
```



### 10.2.3 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(mi5000_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TI')
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int>  <int> <int>  <dbl> <dbl> <int> <dbl>
## 1 EA          100      0  3382  3422  3421.  3473   26
## 2 IS          100      0  4718  4788. 4786.  4834   40
## 3 NMIS        100      0  4736  4783  4782.  4830  25.2
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)

##
## Kruskal-Wallis rank sum test
##
## data: Generations by Structure
## Kruskal-Wallis chi-squared = 200.03, df = 2, p-value < 2.2e-16

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: ssf$Generations and ssf$Structure
##
##      EA      IS
## IS  <2e-16 -
## NMIS <2e-16 1
##
## P value adjustment method: bonferroni
```

## 10.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the exploitation rate diagnostic.

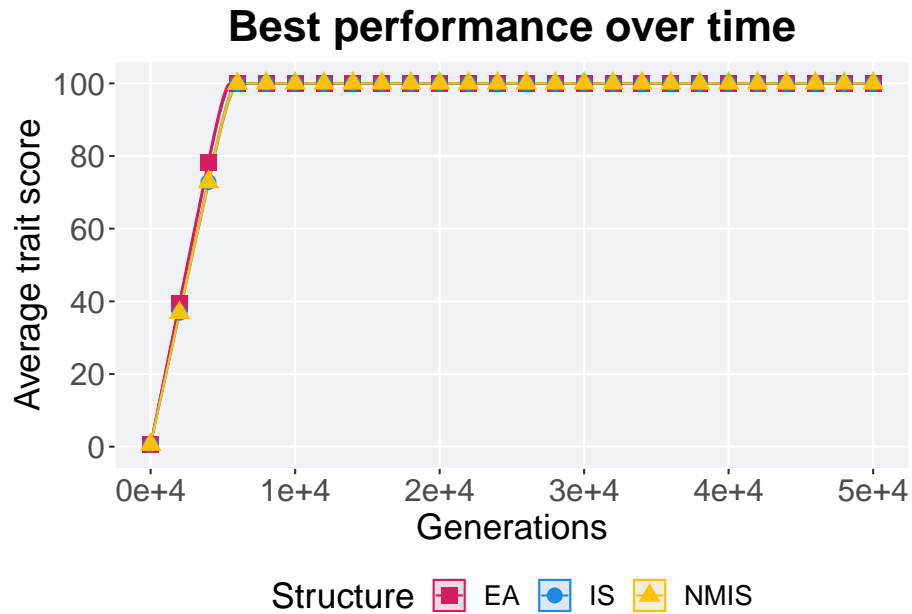
### 10.3.1 Performance over time

```
lines = filter(mi5000_over_time, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TOURNAMENT')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.0)
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
```

```

labels=c("0", "20", "40", "60", "80", "100")
) +
scale_x_continuous(
  name="Generations",
  limits=c(0, 50000),
  breaks=c(0, 10000, 20000, 30000, 40000, 50000),
  labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
) +
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle("Best performance over time") +
p_theme

```



### 10.3.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```

filter(mi5000_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TOURNAMENT')
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +

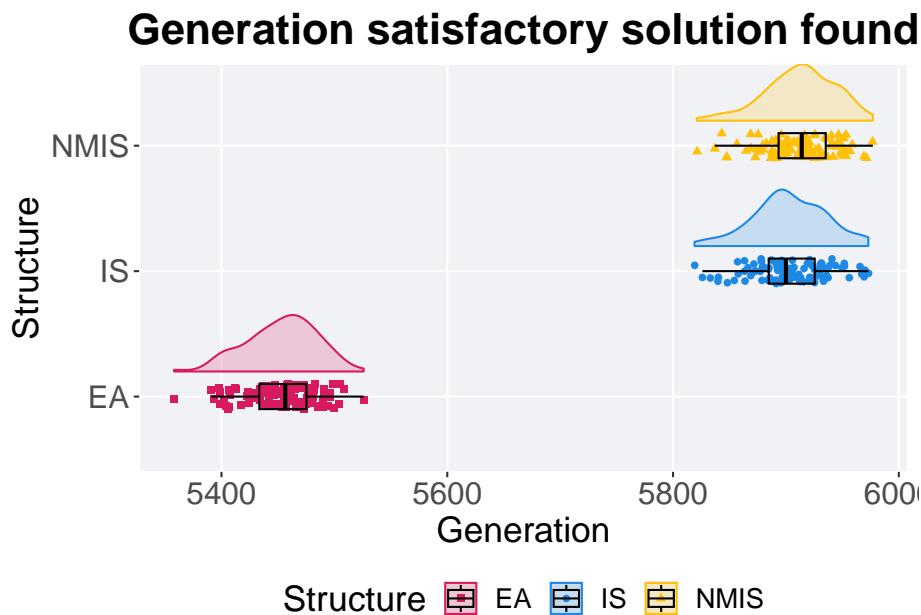
```



```

scale_y_continuous(
  name="Generation"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Generation satisfactory solution found')+
p_theme + coord_flip()

```



### 10.3.3 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(mi5000_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'TOURNAMENT')
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),

```

```

max = max(Generations, na.rm = TRUE),
IQR = IQR(Generations, na.rm = TRUE)
)

```

```

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0  5358  5456. 5453.  5526  41.8
## 2 IS          100     0  5819  5900  5903.  5973  40.8
## 3 NMIS        100     0  5821  5914  5912.  5977  41.8

```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```

##
##  Kruskal-Wallis rank sum test
##
## data:  Generations by Structure
## Kruskal-Wallis chi-squared = 201.55, df = 2, p-value < 2.2e-16

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```

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

```

```

##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  ssf$Generations and ssf$Structure
##
##      EA      IS
## IS  <2e-16 -
## NMIS <2e-16 0.04
##
## P value adjustment method: bonferroni

```

## 10.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the exploitation rate diagnostic.

### 10.4.1 Performance over time

```

lines = filter(mi5000_over_time, Diagnostic == 'EXPLOITATION_RATE') & `Selection` %>%
  group_by(Structure, Generations) %>%
  dplyr::summarise(

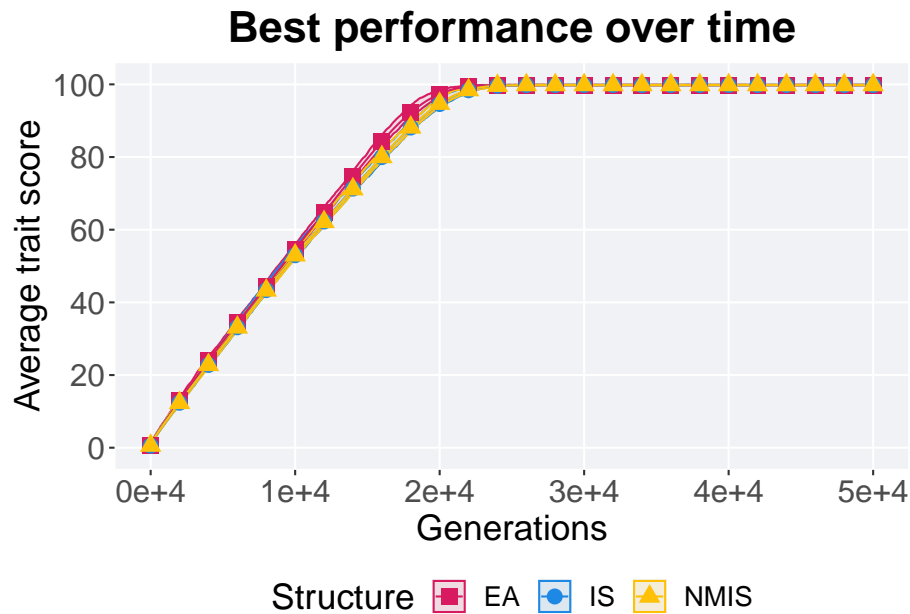
```

```

    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")

  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Best performance over time") +
  p_theme

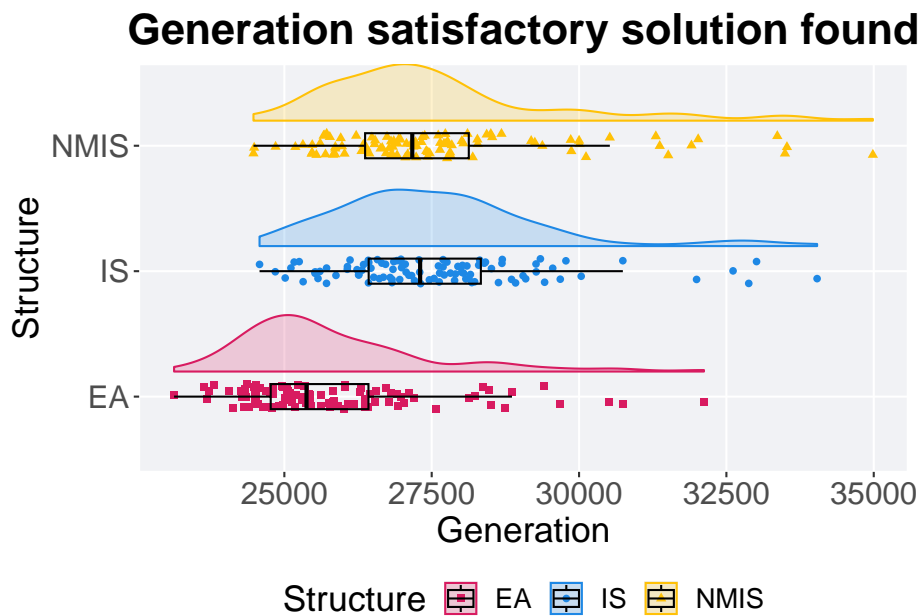
```



#### 10.4.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi5000_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'LEXICAS')
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Generation"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Generation satisfactory solution found') +
  p_theme + coord_flip()
```



### 10.4.3 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(mi5000_ssf, Diagnostic == 'EXPLOITATION_RATE' & `Selection\nScheme` == 'LEXICASE' &
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0 23129 25376 25814. 32119 1658.
## 2 IS          100     0 24579 27304. 27591. 34039 1903.
## 3 NMIS        100     0 24476 27170. 27601. 34985 1759.
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)

##
##  Kruskal-Wallis rank sum test
##
## data:  Generations by Structure
## Kruskal-Wallis chi-squared = 72.912, df = 2, p-value < 2.2e-16

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  ssf$Generations and ssf$Structure
##
##          EA          IS
## IS    1.1e-13 -
## NMIS 5.6e-13 1
##
## P value adjustment method: bonferroni
```

## Chapter 11

# MI5000: Ordered exploitation results

Here we present the results for **best performances** found by each selection scheme replicate on the ordered exploitation diagnostic with configurations presented below. Best performance found refers to the largest average trait score found in a given population. Note that performance values fall between 0.0 and 100.0. For our the configuration of these experiments, we execute migrations every 50 generations and there are 4 islands in a ring topology. When migrations occur, we swap two individuals (same position on each island) and guarantee that no solution can return to the same island.

### 11.1 Analysis dependencies

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

### 11.2 Truncation selection

Here we analyze how the different population structures affect truncation selection (size 8) on the ordered exploitation diagnostic.

#### 11.2.1 Performance over time

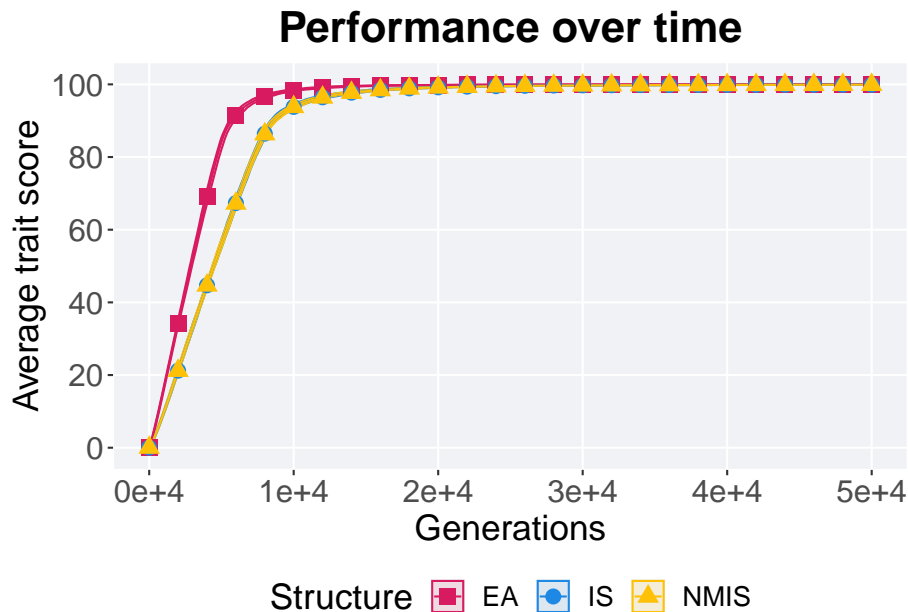
```
lines = filter(mi5000_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'T
  group_by(Structure, Generations) %>%
```

```

dplyr::summarise(
  min = min(pop_fit_max) / DIMENSIONALITY,
  mean = mean(pop_fit_max) / DIMENSIONALITY,
  max = max(pop_fit_max) / DIMENSIONALITY
)
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0,
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme

```



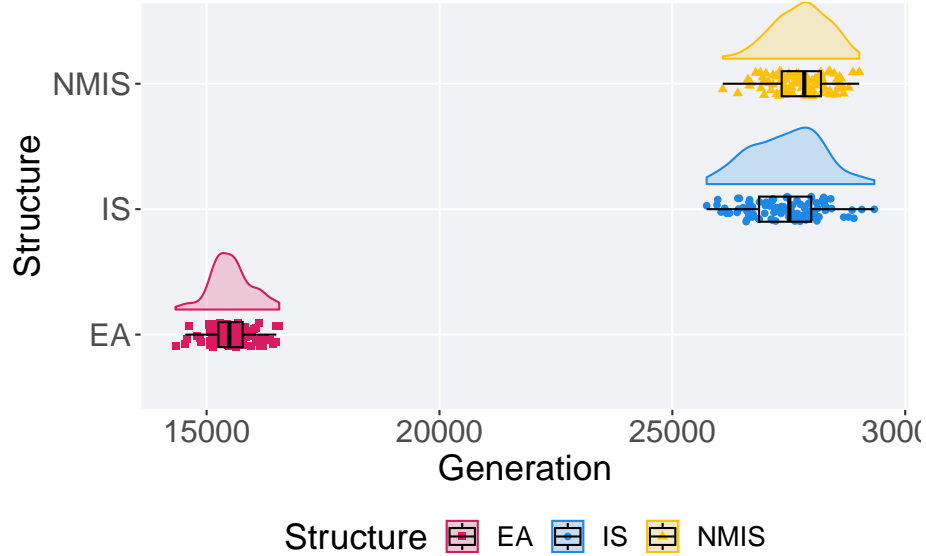


### 11.2.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi5000_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TRUNCATION') %>
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Generation"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Generation satisfactory solution found') +
  p_theme + coord_flip()
```

## Generation satisfactory solution found



### 11.2.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(mi5000_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` ==
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0 14333 15487 15522. 16559 521.
## 2 IS          100     0 25736 27512. 27446. 29337 1121.
## 3 NMIS        100     0 26084 27832 27762. 29013 844.
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)

##
## Kruskal-Wallis rank sum test
##
## data: Generations by Structure
## Kruskal-Wallis chi-squared = 203.38, df = 2, p-value < 2.2e-16

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: ssf$Generations and ssf$Structure
##
##      EA      IS
## IS  <2e-16 -
## NMIS <2e-16 0.0039
##
## P value adjustment method: bonferroni
```

## 11.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the ordered exploitation diagnostic.

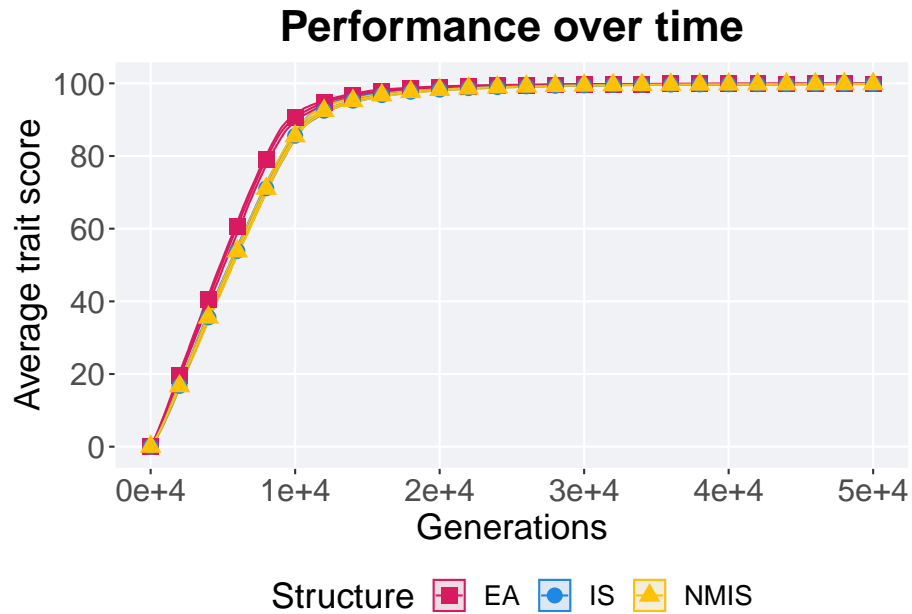
### 11.3.1 Performance over time

```
lines = filter(mi5000_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'Tournament')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.0)
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
```

```

labels=c("0", "20", "40", "60", "80", "100")
) +
scale_x_continuous(
  name="Generations",
  limits=c(0, 50000),
  breaks=c(0, 10000, 20000, 30000, 40000, 50000),
  labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
) +
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle("Performance over time") +
p_theme

```



### 11.3.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```

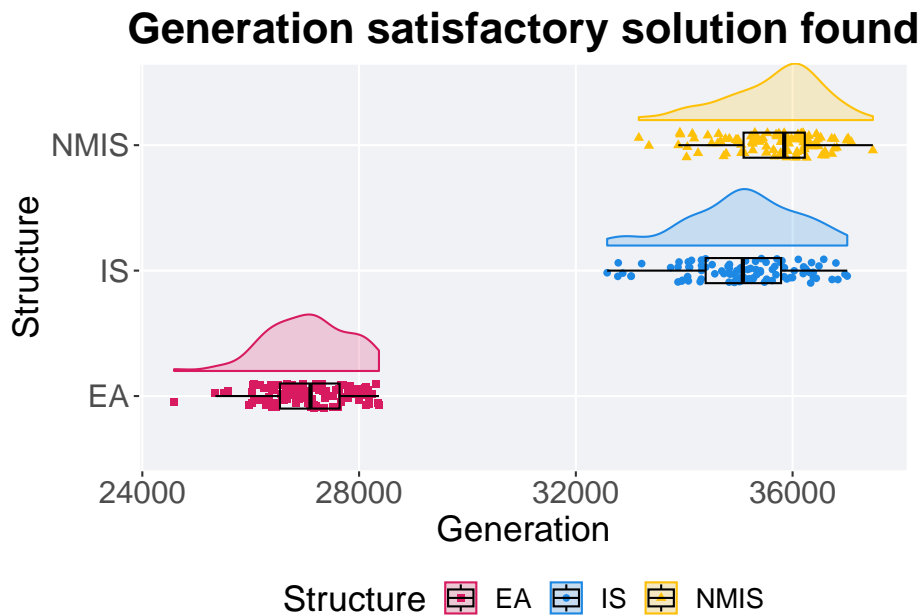
filter(mi5000_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TOURNAMENT')
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +

```

```

scale_y_continuous(
  name="Generation"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Generation satisfactory solution found')+
p_theme + coord_flip()

```



#### 11.3.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(mi5000_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'TOURNAMENT')
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),

```

```

max = max(Generations, na.rm = TRUE),
IQR = IQR(Generations, na.rm = TRUE)
)

```

```

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0 24586 27104. 27057. 28367 1101.
## 2 IS          100     0 32578 35082  35088. 37009 1392
## 3 NMIS        100     0 33162 35845  35659. 37481 1130.

```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```

##
##   Kruskal-Wallis rank sum test
##
## data:   Generations by Structure
## Kruskal-Wallis chi-squared = 206.91, df = 2, p-value < 2.2e-16

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```

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

```

```

##
##   Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:   ssf$Generations and ssf$Structure
##
##           EA           IS
## IS    < 2e-16 -
## NMIS < 2e-16 5.6e-05
##
## P value adjustment method: bonferroni

```

## 11.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the ordered exploitation diagnostic.

### 11.4.1 Performance over time

```

lines = filter(mi5000_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScore` > 0.5)
group_by(Structure, Generations) %>%
dplyr::summarise(

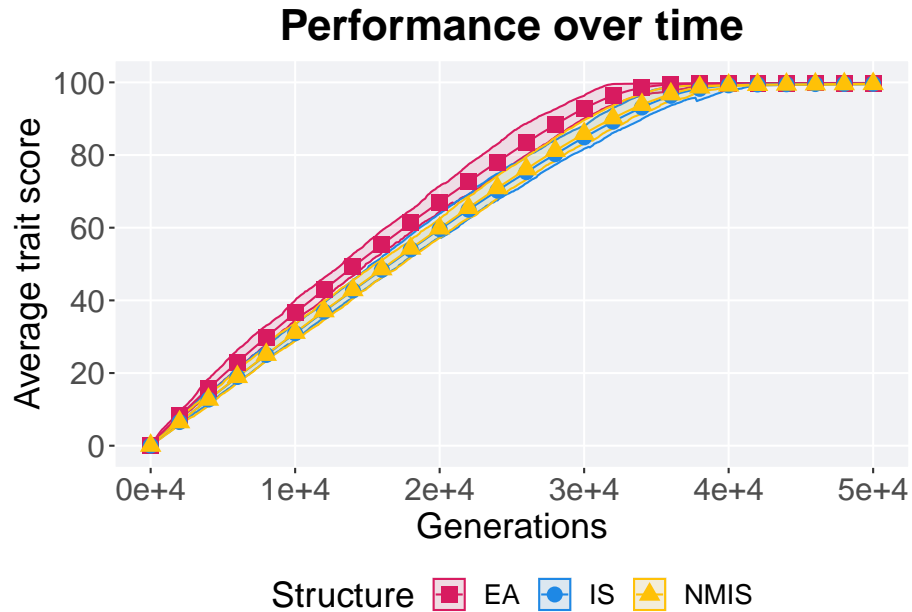
```

```

    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Average trait score",
    limits=c(-1, 101),
    breaks=seq(0,100, 20),
    labels=c("0", "20", "40", "60", "80", "100")
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")

  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme

```

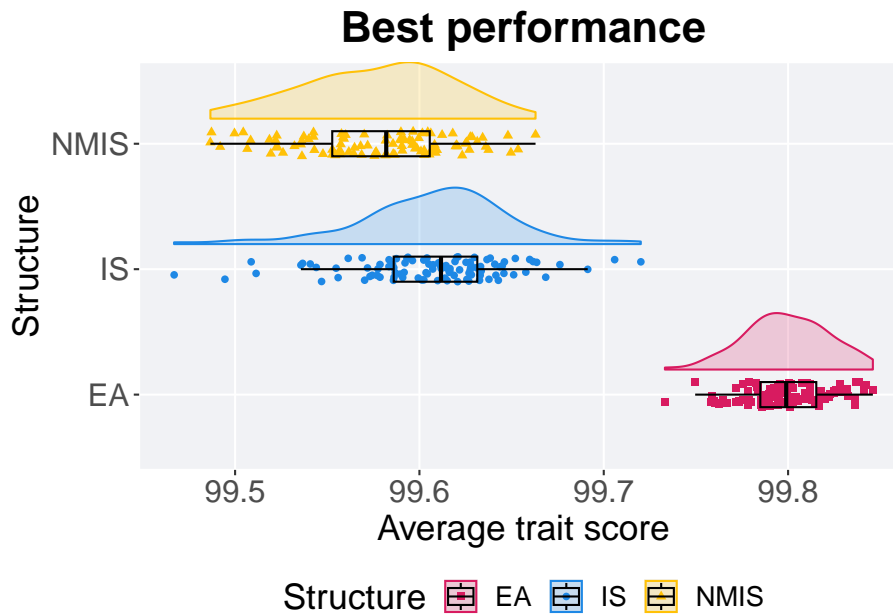


### 11.4.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi5000_best, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEX')
ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best performance') +
  p_theme + coord_flip()
```





#### 11.4.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi5000_best, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == '
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE) / DIMENSIONALITY,
    median = median(VAL, na.rm = TRUE) / DIMENSIONALITY,
    mean = mean(VAL, na.rm = TRUE) / DIMENSIONALITY,
    max = max(VAL, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VAL, na.rm = TRUE) / DIMENSIONALITY
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0  99.7  99.8  99.8  99.8 0.0303
## 2 IS          100     0  99.5  99.6  99.6  99.7 0.0454
## 3 NMIS        100     0  99.5  99.6  99.6  99.7 0.0529
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VAL ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: VAL by Structure
## Kruskal-Wallis chi-squared = 211.11, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

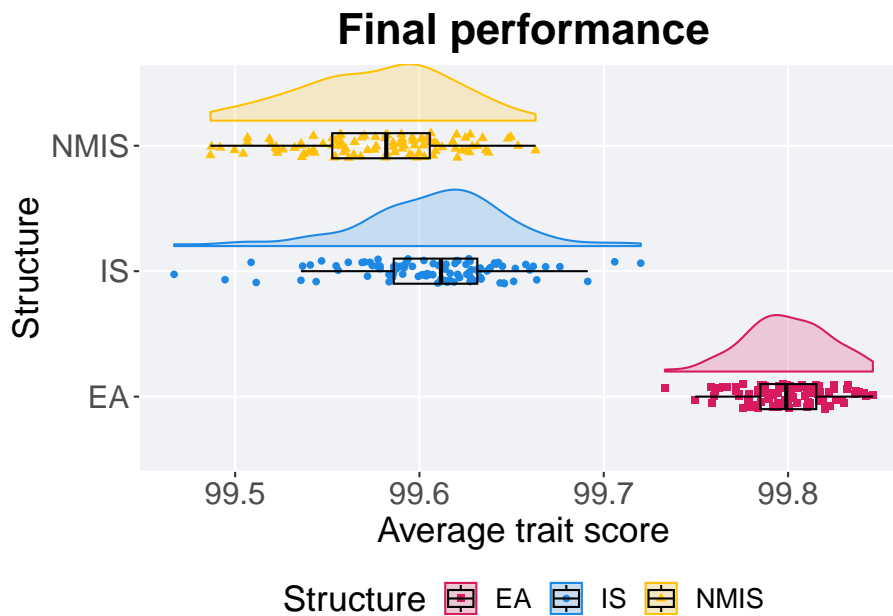
```
pairwise.wilcox.test(x = performance$VAL, g = performance$Structure, p.adjust.method =
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$VAL and performance$Structure
##
##      EA      IS
## IS    < 2e-16 -
## NMIS < 2e-16 4.1e-07
##
## P value adjustment method: bonferroni
```

### 11.4.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi5000_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` ==
       ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fi
       geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
       geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
       geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
       scale_y_continuous(
         name="Average trait score"
       ) +
       scale_x_discrete(
         name="Structure"
       ) +
       scale_shape_manual(values=SHAPE)+
       scale_colour_manual(values = cb_palette, ) +
       scale_fill_manual(values = cb_palette) +
       ggtitle('Final performance')+
       p_theme + coord_flip()
```



#### 11.4.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi5000_over_time, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme`
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_fit_max)),
    min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0  99.7  99.8  99.8  99.8 0.0303
## 2 IS          100     0  99.5  99.6  99.6  99.7 0.0454
## 3 NMIS        100     0  99.5  99.6  99.6  99.7 0.0529
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 211.12, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.m = "none",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$pop_fit_max and performance$Structure
##
##      EA      IS
## IS    <2e-16 -
## NMIS <2e-16 4e-07
##
## P value adjustment method: bonferroni
```

#### 11.4.4 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

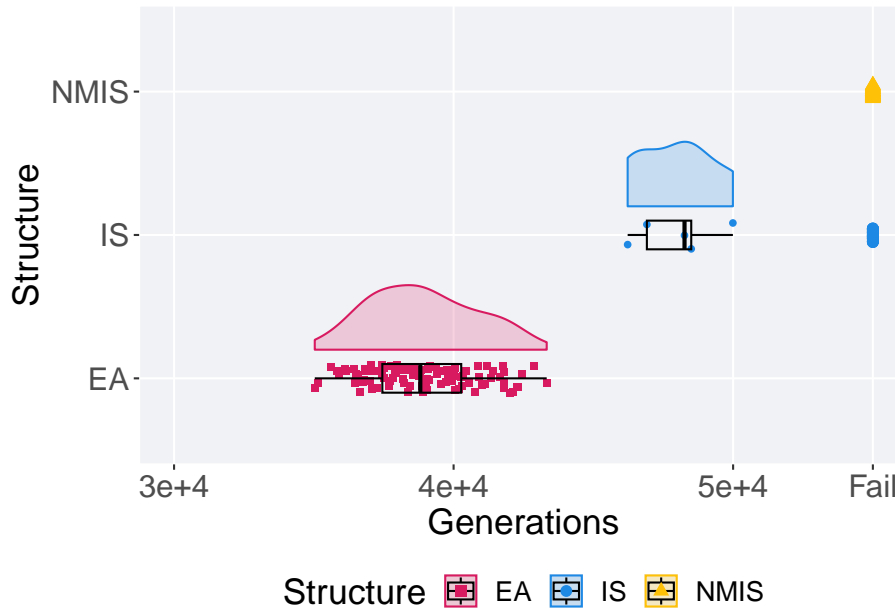
```
lex_fail = filter(mi5000_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEXICOM')
lex_fail$Generations = 55000
lex_fail$Structure <- factor(lex_fail$Structure, levels = MODEL)

filter(mi5000_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEXICOM') %>%
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
    geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = .5) +
    geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
    geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
    geom_point(data = lex_fail, aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
    scale_shape_manual(values=SHAPE) +
    scale_y_continuous(
      name="Generations",
      limits=c(30000, 55000),
      breaks=c(30000, 40000, 50000, 55000),
      labels=c("3e+4", "4e+4", "5e+4", "Fail")
    ) +
    scale_x_discrete(
```

```

  name="Structure"
) +
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
p_theme + coord_flip()

```



#### 11.4.4.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(mi5000_ssf, Diagnostic == 'ORDERED_EXPLOITATION' & `Selection\nScheme` == 'LEXICASE')
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )

```

```

## # A tibble: 2 x 8
##   Structure count na_cnt  min median  mean  max  IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>

```

```
## 1 EA          100      0 35033  38809 38906. 43331 2822.
## 2 IS           5      0 46227  48262 47980. 49992 1587
```

Kruskal-Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  Generations by Structure
## Kruskal-Wallis chi-squared = 14.151, df = 1, p-value = 0.0001687
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  ssf$Generations and ssf$Structure
##
##      EA
## IS 8.7e-05
##
## P value adjustment method: bonferroni
```

## Chapter 12

# MI5000: Contradictory objectives results

Here we present the results for the **satisfactory trait coverage** and **activation gene coverage** generated by each selection scheme replicate on the contradictory objectives diagnostic the configurations presented below. Note both of these values are gathered at the population-level. Activation gene coverage refers to the count of unique activation genes in a given population; this gives us a range of integers between 0 and 100. Satisfactory trait coverage refers to the count of unique satisfied traits in a given population; this gives us a range of integers between 0 and 100. For our the configuration of these experiments, we execute migrations every 50 generations and there are 4 islands in a ring topology. When migrations occur, two individuals are swapped (same position on each island) and guarantee that no solution can return to its original island.

### 12.1 Analysis dependencies

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

### 12.2 Truncation selection

Here we analyze how the different population structures affect truncation selection (size 8) on the contradictory objectives diagnostic.

## 12.2.1 Satisfactory trait coverage

Satisfactory trait coverage analysis.

### 12.2.1.1 Coverage over time

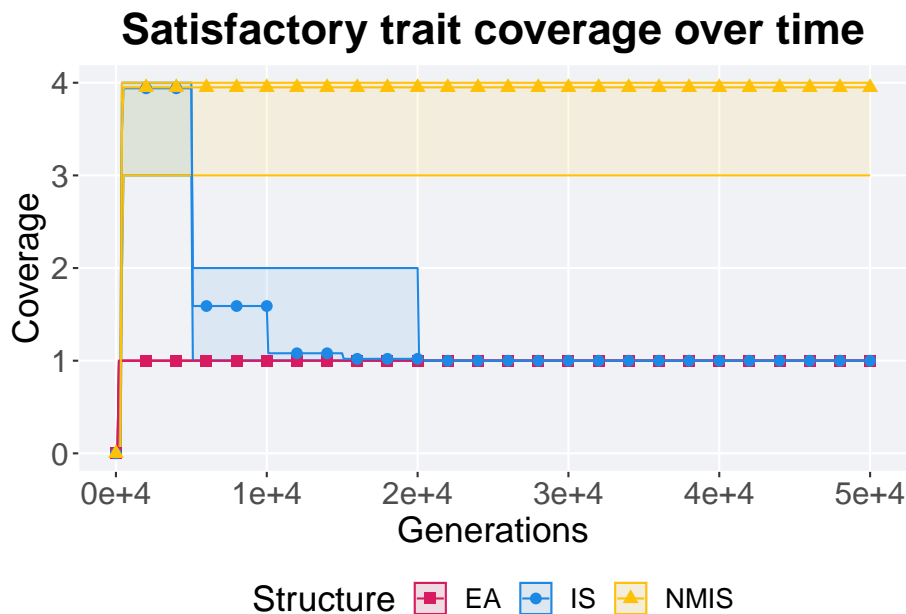
Satisfactory trait coverage over time.

```
lines = filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection` == 'Satisfactory')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_sat_cov),
    mean = mean(pop_sat_cov),
    max = max(pop_sat_cov)
  )
```

## `summarise()` has grouped output by 'Structure'. You can override using the  
## `.groups` argument.

```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
    scale_y_continuous(
      name="Coverage"
    ) +
    scale_x_continuous(
      name="Generations",
      limits=c(0, 50000),
      breaks=c(0, 10000, 20000, 30000, 40000, 50000),
      labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
    ) +
    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
```

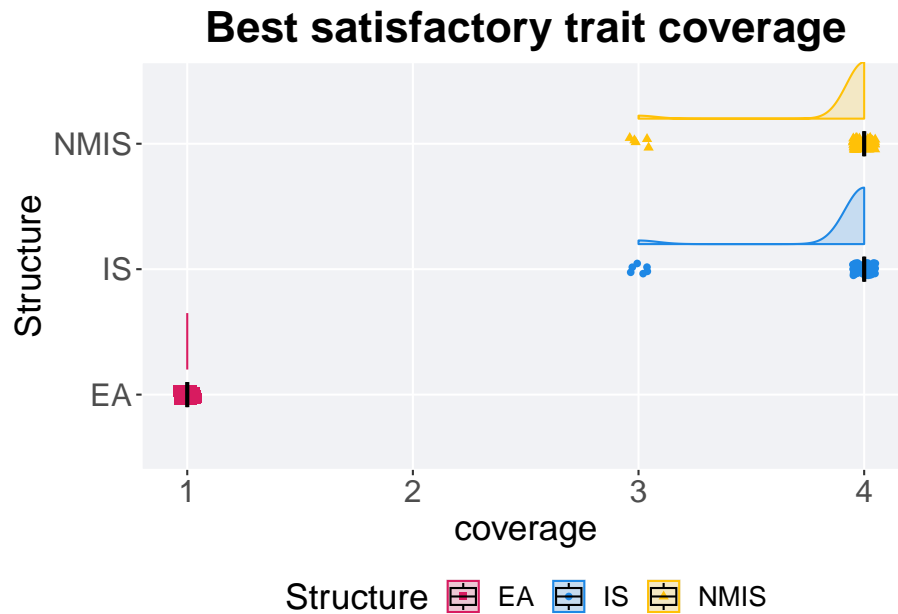




#### 12.2.1.2 Best coverage throughout

Best satisfactory trait coverage throughout 50,000 generations.

```
### best satisfactory trait coverage throughout
filter(mi5000_best, Diagnostic == 'CONTRADICTION' & `Selection\nScheme` == 'TRUNCATION')
ggplot(., aes(x = Structure, y = VAL, color = Structure, fill = Structure, shape = Structure))
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory trait coverage') +
  p_theme + coord_flip()
```



#### 12.2.1.2.1 Stats

Summary statistics for the best satisfactory trait coverage.

```
### best
coverage = filter(mi5000_best, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection` == 'n')
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE),
    median = median(VAL, na.rm = TRUE),
    mean = mean(VAL, na.rm = TRUE),
    max = max(VAL, na.rm = TRUE),
    IQR = IQR(VAL, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     3     4   3.94     4     0
## 3 NMIS        100     0     3     4   3.95     4     0
```

Kruskal–Wallis test provides evidence of difference among satisfactory trait

coverage.

```
kruskal.test(VAL ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: VAL by Structure
## Kruskal-Wallis chi-squared = 279.63, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage.

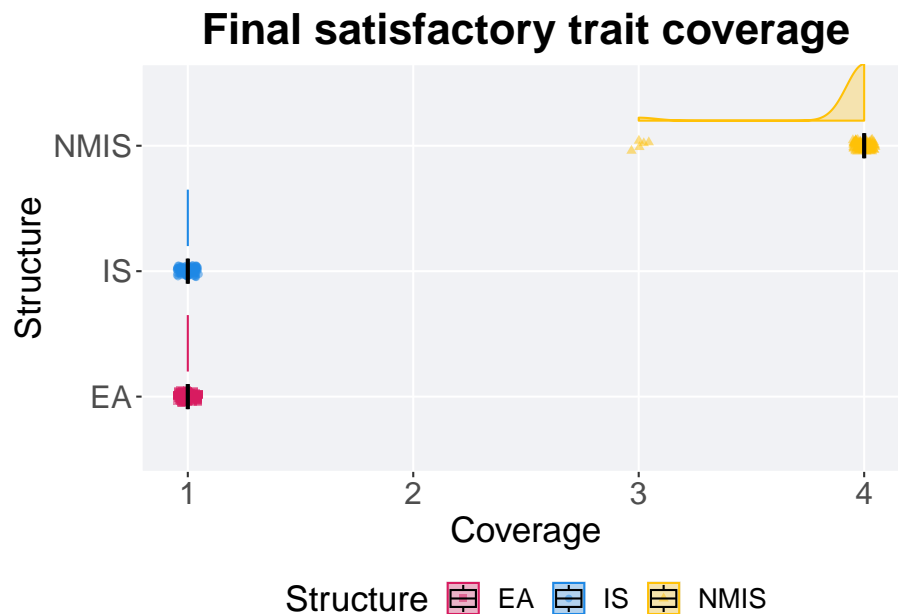
```
pairwise.wilcox.test(x = coverage$VAL, g = coverage$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$VAL and coverage$Structure
##
##      EA      IS
## IS  <2e-16 -
## NMIS <2e-16 1
##
## P value adjustment method: bonferroni
```

### 12.2.1.3 End of 50,000 generations

Satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'TRUNC
  ggplot(., aes(x = Structure, y = pop_sat_cov, color = Structure, fill = Structure, shape = Stru
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE)+
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final satisfactory trait coverage')+
  p_theme + coord_flip()
```



#### 12.2.1.3.1 Stats

Summary statistics for satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
coverage = filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Select
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_sat_cov)),
    min = min(pop_sat_cov, na.rm = TRUE),
    median = median(pop_sat_cov, na.rm = TRUE),
    mean = mean(pop_sat_cov, na.rm = TRUE),
    max = max(pop_sat_cov, na.rm = TRUE),
    IQR = IQR(pop_sat_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     3     4    3.95     4     0
```

Kruskal-Wallis test provides evidence of difference among satisfactory trait coverage in the population at the end of 50,000 generations.

```
kruskal.test(pop_sat_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_sat_cov by Structure
## Kruskal-Wallis chi-squared = 296.65, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage in the population at the end of 50,000 generations.

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

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_sat_cov and coverage$Structure
##
##      EA      IS
## IS   1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

### 12.2.2 Activation gene coverage

Activation gene coverage analysis.

#### 12.2.2.1 Coverage over time

Activation gene coverage over time.

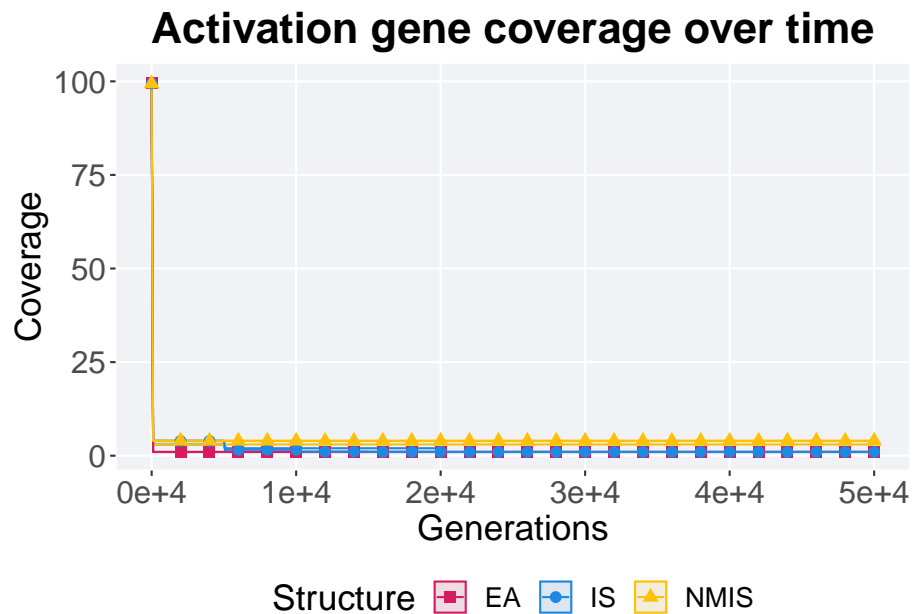
```
# data for lines and shading on plots
lines = filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` =
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )
```

```
## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.
```

```

ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
    scale_y_continuous(
      name="Coverage"
    ) +
    scale_x_continuous(
      name="Generations",
      limits=c(0, 50000),
      breaks=c(0, 10000, 20000, 30000, 40000, 50000),
      labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
    ) +
    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

```



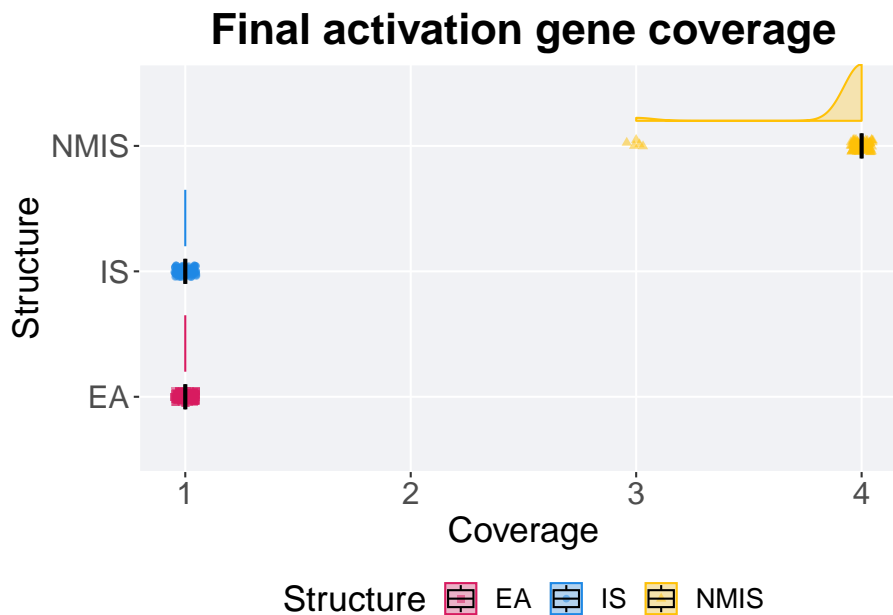
#### 12.2.2.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```

### end of run
filter(mi5000_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` == 'TRUNC
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Stru
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE)+
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage')+
  p_theme + coord_flip()

```



#### 12.2.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(mi5000_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme`
coverage %>%
  group_by(Structure) %>%

```

```
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(pop_act_cov)),
  min = min(pop_act_cov, na.rm = TRUE),
  median = median(pop_act_cov, na.rm = TRUE),
  mean = mean(pop_act_cov, na.rm = TRUE),
  max = max(pop_act_cov, na.rm = TRUE),
  IQR = IQR(pop_act_cov, na.rm = TRUE)
)
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     3     4    3.95     4     0
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
##   Kruskal-Wallis rank sum test
##
## data:  pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 296.65, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```
pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method =
  paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
##   Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS    1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```



## 12.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the contradictory objectives diagnostic.

### 12.3.1 Satisfactory trait coverage

Satisfactory trait coverage analysis.

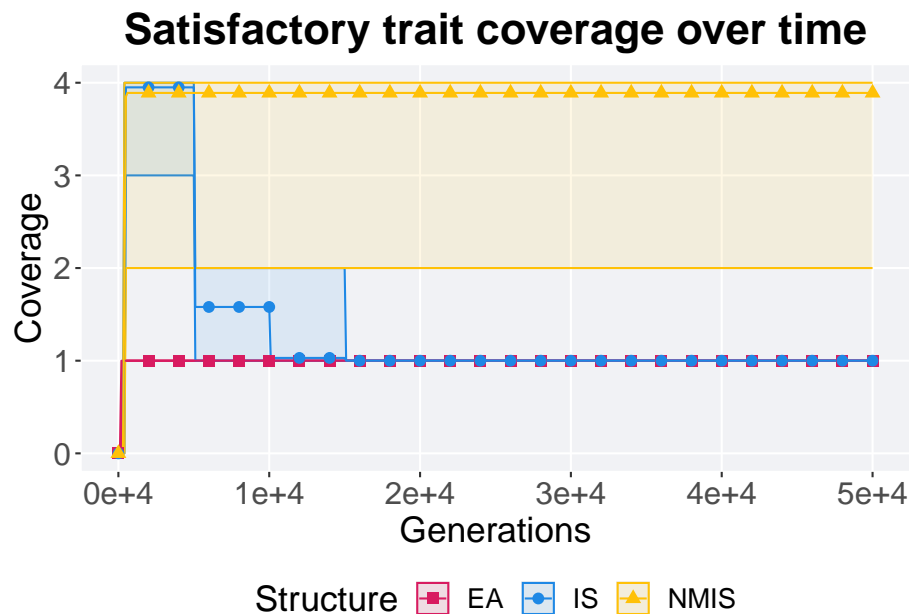
#### 12.3.1.1 Coverage over time

Satisfactory trait coverage over time.

```
lines = filter(mi5000_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` =
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_sat_cov),
    mean = mean(pop_sat_cov),
    max = max(pop_sat_cov)
  )

## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.

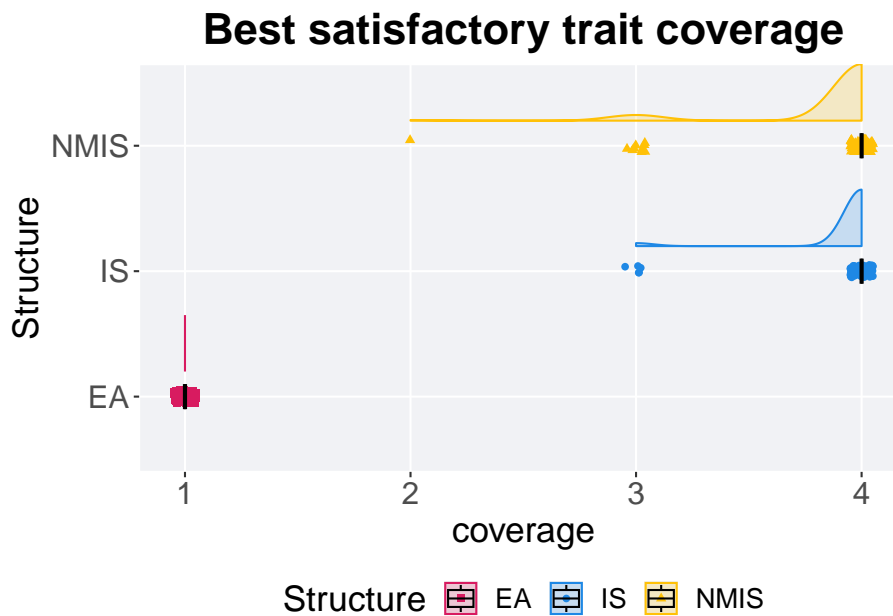
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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
```



#### 12.3.1.2 Best coverage throughout

Best satisfactory trait coverage throughout 50,000 generations.

```
### best satisfactory trait coverage throughout
filter(mi5000_best, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` ==
  ggplot(., aes(x = Structure, y = VAL, color = Structure, fill = Structure, shape = S
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory trait coverage') +
  p_theme + coord_flip()
```



#### 12.3.1.2.1 Stats

Summary statistics for the best satisfactory trait coverage.

```
### best
coverage = filter(mi5000_best, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` ==
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE),
    median = median(VAL, na.rm = TRUE),
    mean = mean(VAL, na.rm = TRUE),
    max = max(VAL, na.rm = TRUE),
    IQR = IQR(VAL, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA         100     0     1     1     1     1     0
## 2 IS         100     0     3     4    3.95     4     0
## 3 NMIS       100     0     2     4    3.89     4     0
```

Kruskal–Wallis test provides evidence of difference among satisfactory trait

coverage.

```
kruskal.test(VAL ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: VAL by Structure
## Kruskal-Wallis chi-squared = 273.91, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage.

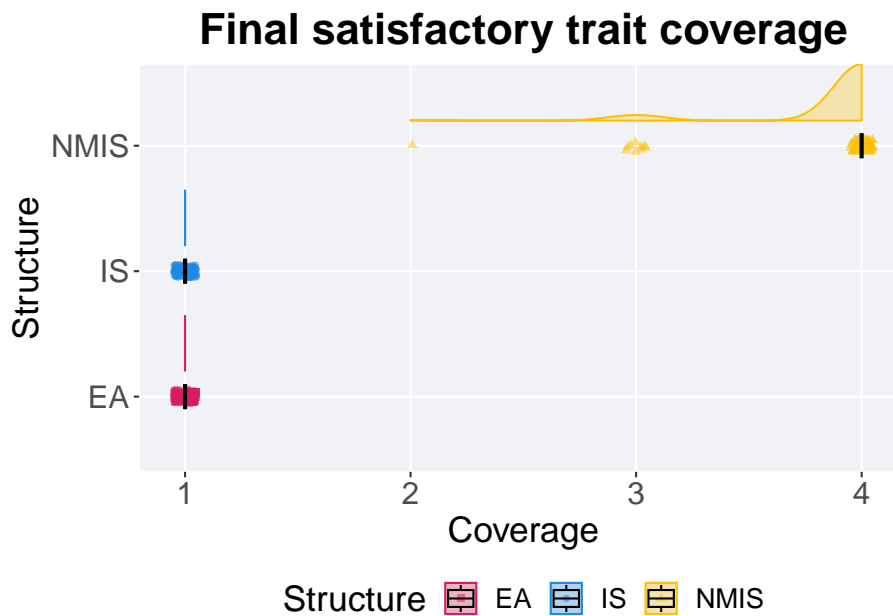
```
pairwise.wilcox.test(x = coverage$VAL, g = coverage$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$VAL and coverage$Structure
##
##      EA      IS
## IS   <2e-16 -
## NMIS <2e-16 1
##
## P value adjustment method: bonferroni
```

### 12.3.1.3 End of 50,000 generations

Satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'Satisfactory') %>%
  ggplot(., aes(x = Structure, y = pop_sat_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final satisfactory trait coverage') +
  p_theme + coord_flip()
```



#### 12.3.1.3.1 Stats

Summary statistics for satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
coverage = filter(mi5000_over_time, Diagnostic == 'CONTRADICTION_OBJECTIVES' & `Selection\nScheme` == 'EA')
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_sat_cov)),
    min = min(pop_sat_cov, na.rm = TRUE),
    median = median(pop_sat_cov, na.rm = TRUE),
    mean = mean(pop_sat_cov, na.rm = TRUE),
    max = max(pop_sat_cov, na.rm = TRUE),
    IQR = IQR(pop_sat_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt  min median  mean  max  IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     2     4    3.89    4     0
```

Kruskal–Wallis test provides evidence of difference among satisfactory trait coverage in the population at the end of 50,000 generations.

```
kruskal.test(pop_sat_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_sat_cov by Structure
## Kruskal-Wallis chi-squared = 294.58, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage in the population at the end of 50,000 generations.

```
pairwise.wilcox.test(x = coverage$pop_sat_cov, g = coverage$Structure, p.adjust.method =
                    paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_sat_cov and coverage$Structure
##
##      EA      IS
## IS   1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

### 12.3.2 Activation gene coverage

Activation gene coverage analysis.

#### 12.3.2.1 Coverage over time

Activation gene coverage over time.

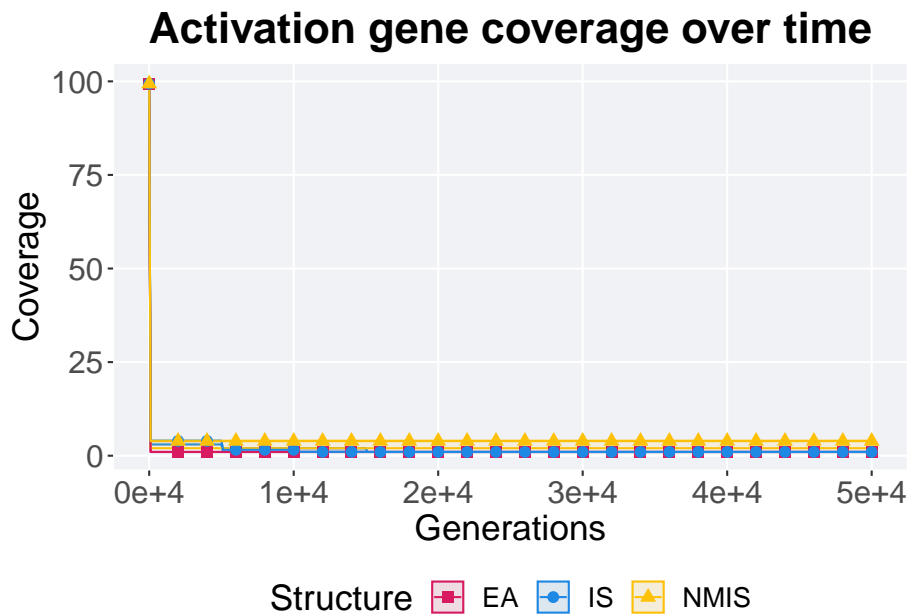
```
# data for lines and shading on plots
lines = filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection` == 'Satisfactory')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )
```

```
## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.
```

```

ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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

```



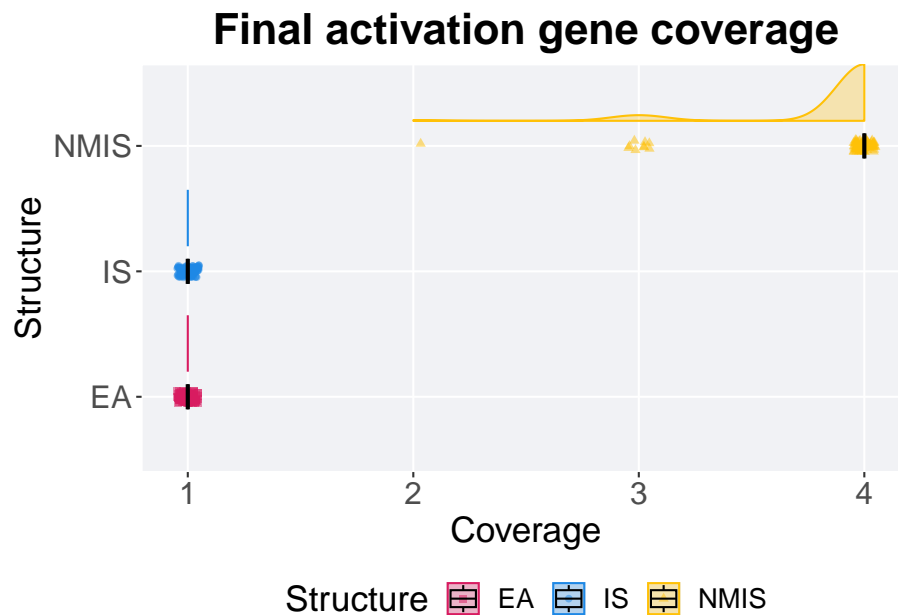
### 12.3.2.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```

### end of run
filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'IS')
ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme + coord_flip()

```



#### 12.3.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'IS')
coverage %>%
  group_by(Structure) %>%

```



```
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(pop_act_cov)),
  min = min(pop_act_cov, na.rm = TRUE),
  median = median(pop_act_cov, na.rm = TRUE),
  mean = mean(pop_act_cov, na.rm = TRUE),
  max = max(pop_act_cov, na.rm = TRUE),
  IQR = IQR(pop_act_cov, na.rm = TRUE)
)
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     1     1     1     0
## 2 IS          100     0     1     1     1     1     0
## 3 NMIS        100     0     2     4    3.89     4     0
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 294.58, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

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

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS    1      -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 12.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the contradictory objectives diagnostic.

### 12.4.1 Satisfactory trait coverage

Satisfactory trait coverage analysis.

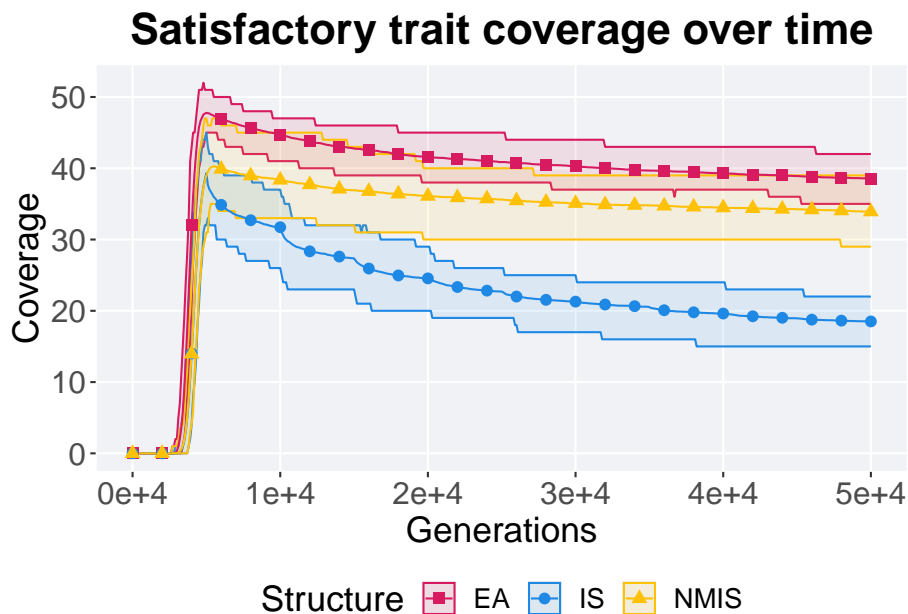
#### 12.4.1.1 Coverage over time

Satisfactory trait coverage over time.

```
lines = filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection` == 'Lexicase')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_sat_cov),
    mean = mean(pop_sat_cov),
    max = max(pop_sat_cov)
  )

## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.

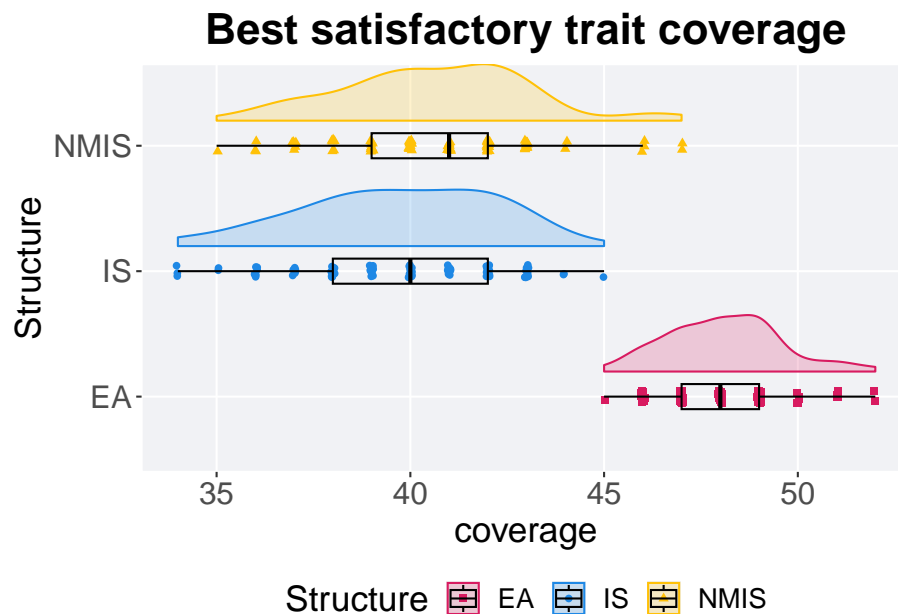
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
    scale_y_continuous(
      name="Coverage"
    ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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
```



#### 12.4.1.2 Best coverage throughout

Best satisfactory trait coverage throughout 50,000 generations.

```
### best satisfactory trait coverage throughout
filter(mi5000_best, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE')
ggplot(., aes(x = Structure, y = VAL, color = Structure, fill = Structure, shape = Structure))
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best satisfactory trait coverage') +
  p_theme + coord_flip()
```



#### 12.4.1.2.1 Stats

Summary statistics for the best satisfactory trait coverage.

```
### best
coverage = filter(mi5000_best, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection`\n
coverage$Structure = factor(coverage$Structure, levels=c('EA','NMIS','IS'))
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE),
    median = median(VAL, na.rm = TRUE),
    mean = mean(VAL, na.rm = TRUE),
    max = max(VAL, na.rm = TRUE),
    IQR = IQR(VAL, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0    45     48  48.1    52     2
## 2 NMIS        100     0    35     41  40.6    47     3
## 3 IS          100     0    34     40  39.7    45     4
```

Kruskal-Wallis test provides evidence of difference among satisfactory trait coverage.

```
kruskal.test(VAL ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: VAL by Structure
## Kruskal-Wallis chi-squared = 200.46, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage.

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

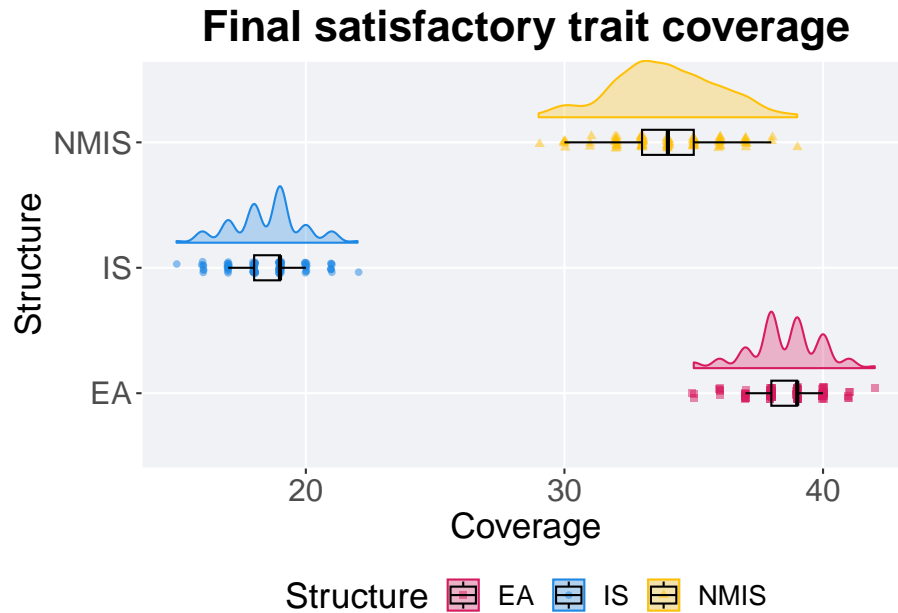
```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$VAL and coverage$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 0.025
##
## P value adjustment method: bonferroni
```

#### 12.4.1.3 End of 50,000 generations

Satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE')
ggplot(., aes(x = Structure, y = pop_sat_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final satisfactory trait coverage') +
```

```
p_theme + coord_flip()
```



#### 12.4.1.3.1 Stats

Summary statistics for satisfactory trait coverage in the population at the end of 50,000 generations.

```
### end of run
coverage = filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Select.
coverage$Structure = factor(coverage$Structure, levels=c('EA', 'NMIS', 'IS'))
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_sat_cov)),
    min = min(pop_sat_cov, na.rm = TRUE),
    median = median(pop_sat_cov, na.rm = TRUE),
    mean = mean(pop_sat_cov, na.rm = TRUE),
    max = max(pop_sat_cov, na.rm = TRUE),
    IQR = IQR(pop_sat_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0    35    39  38.5    42     1
```

```
## 2 NMIS      100      0    29    34 33.9    39    2
## 3 IS        100      0    15    19 18.5    22    1
```

Kruskal–Wallis test provides evidence of difference among satisfactory trait coverage in the population at the end of 50,000 generations.

```
kruskal.test(pop_sat_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  pop_sat_cov by Structure
## Kruskal-Wallis chi-squared = 259.02, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on satisfactory trait coverage in the population at the end of 50,000 generations.

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

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_sat_cov and coverage$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 12.4.2 Activation gene coverage

Activation gene coverage analysis.

### 12.4.2.1 Coverage over time

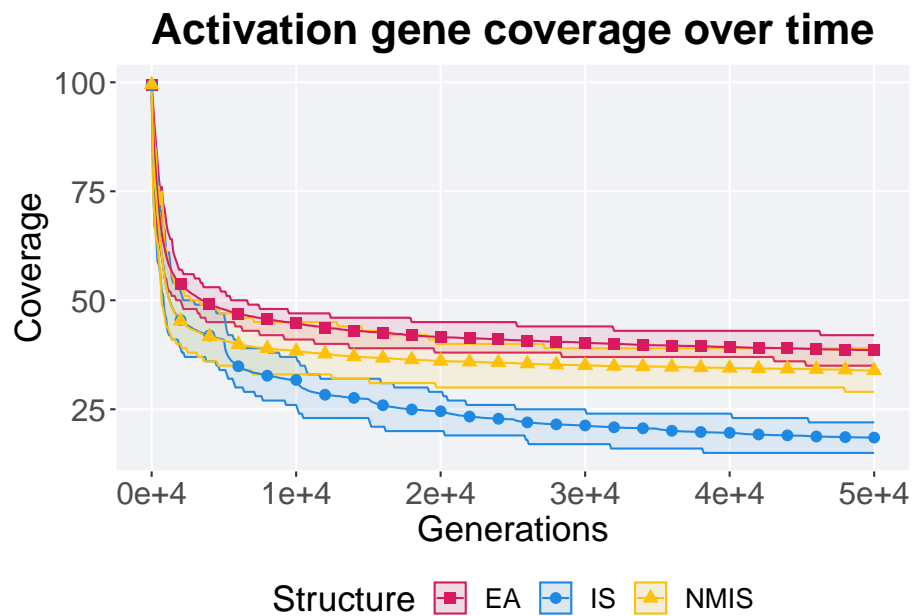
Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(mi5000_over_time, Diagnostic == 'CONTRADICTORY_OBJECTIVES' & `Selection\nScheme` == 'Selection')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )
```

## `summarise()` has grouped output by 'Structure'. You can override using the

```
## `.groups` argument.
```

```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
    scale_y_continuous(
      name="Coverage"
    ) +
    scale_x_continuous(
      name="Generations",
      limits=c(0, 50000),
      breaks=c(0, 10000, 20000, 30000, 40000, 50000),
      labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
    ) +
    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
```



#### 12.4.2.2 End of 50,000 generations

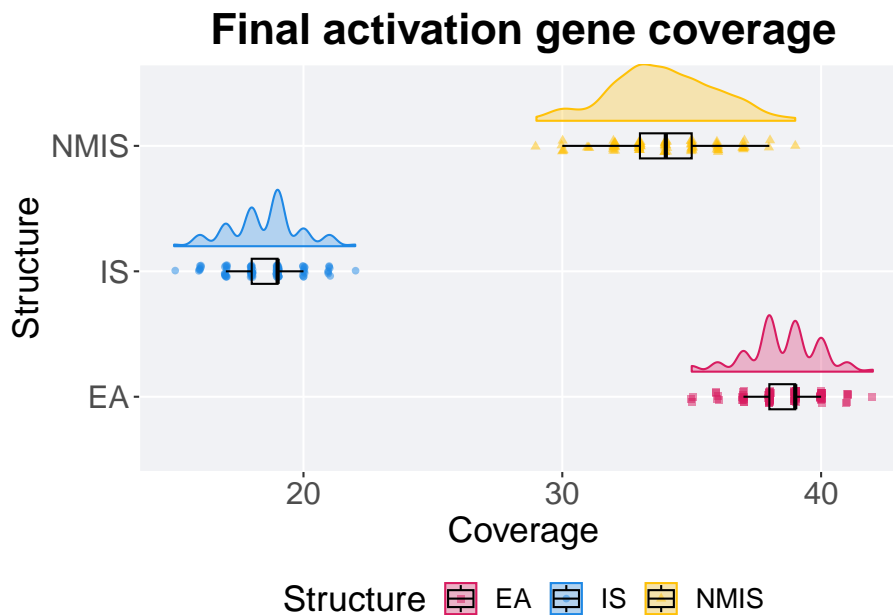
Activation gene coverage in the population at the end of 50,000 generations.



```

### end of run
filter(mi5000_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE') %>%
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme + coord_flip()

```



#### 12.4.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(mi5000_over_time, Diagnostic == 'CONTRADICTIONARY_OBJECTIVES' & `Selection\nScheme` == 'LEXICASE') %>%
  coverage$Structure = factor(coverage$Structure, levels=c('EA', 'NMIS', 'IS'))
coverage %>%

```

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

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA         100     0    35    39  38.5   42     1
## 2 NMIS       100     0    29    34  33.9   39     2
## 3 IS        100     0    15    19  18.5   22     1
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 259.02, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```
pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method =
  paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_act_cov and coverage$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## Chapter 13

# MI5000: Multi-path exploration results

Here we present the results for the **best performances** and **activation gene coverage** generated by each selection scheme replicate on the multi-path exploration diagnostic with configurations presented below. For our the configuration of these experiments, we execute migrations every 50 generations and there are 4 islands in a ring topology. Best performance found refers to the largest average trait score found in a given population. Note that activation gene coverage values are gathered at the population-level. Activation gene coverage refers to the count of unique activation genes in a given population; this gives us a range of integers between 0 and 100.

### 13.1 Analysis dependencies

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

### 13.2 Truncation selection

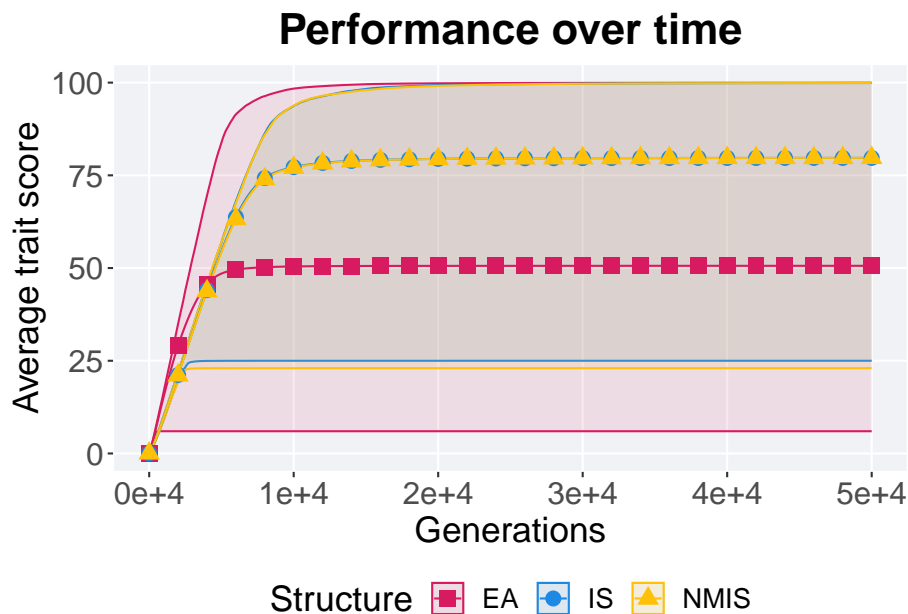
Here we analyze how the different population structures affect truncation selection (size 8) on the contradictory objectives diagnostic.

### 13.2.1 Performance

```

lines = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection`\nS
group_by(Structure, Generations) %>%
dplyr::summarise(
  min = min(pop_fit_max) / DIMENSIONALITY,
  mean = mean(pop_fit_max) / DIMENSIONALITY,
  max = max(pop_fit_max) / DIMENSIONALITY
)
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0,
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme

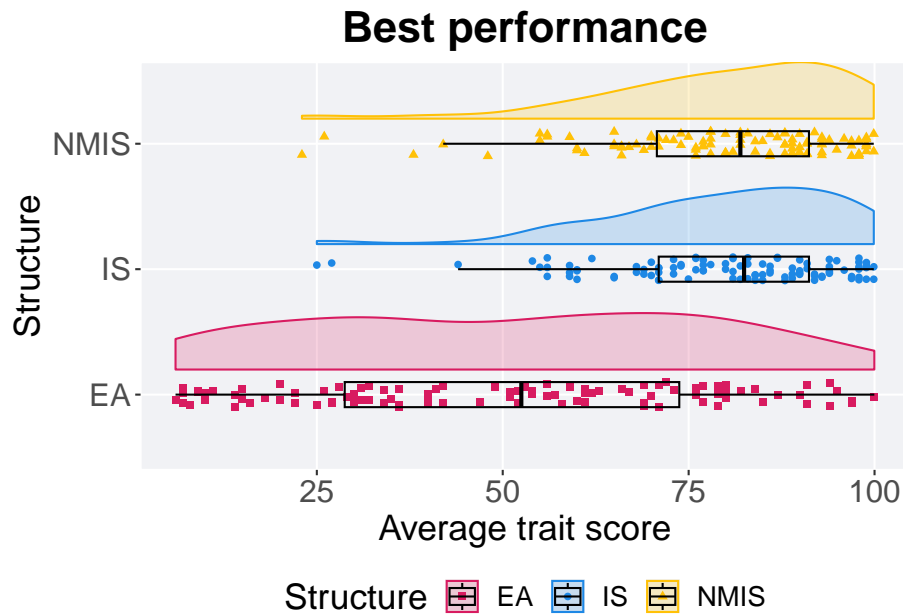
```



#### 13.2.1.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi5000_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TRUNCATION' &
  ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best performance') +
  p_theme + coord_flip()
```



#### 13.2.1.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi5000_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\`n
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE) / DIMENSIONALITY,
    median = median(VAL, na.rm = TRUE) / DIMENSIONALITY,
    mean = mean(VAL, na.rm = TRUE) / DIMENSIONALITY,
    max = max(VAL, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VAL, na.rm = TRUE) / DIMENSIONALITY
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0     6   52.5  50.6  100.   45.0
## 2 IS          100     0   25.0  82.5  79.7  99.9   20.2
## 3 NMIS        100     0   23.0  82.0  79.7  99.9   20.5
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VA ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  VAL by Structure
## Kruskal-Wallis chi-squared = 75.468, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

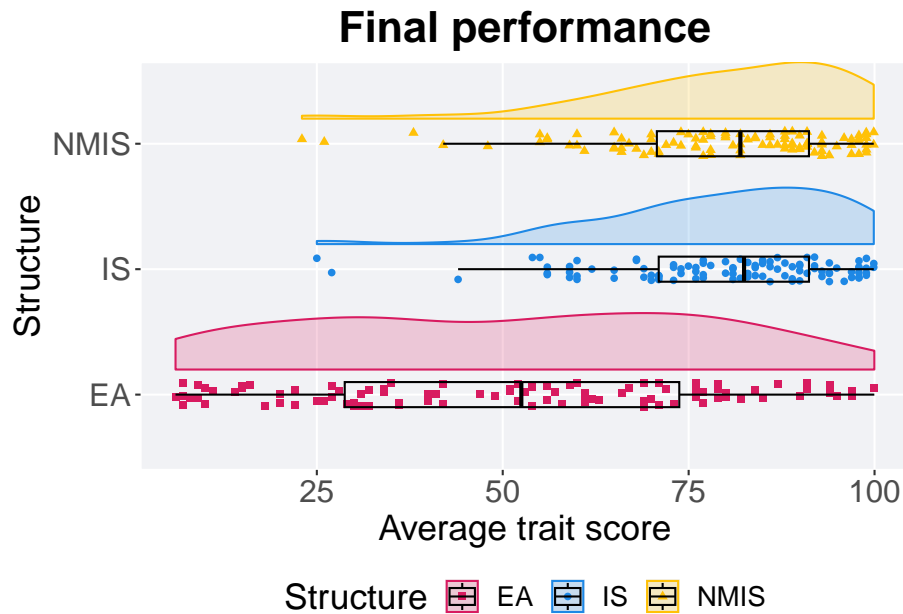
```
pairwise.wilcox.test(x = performance$VAL, g = performance$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VAL and performance$Structure
##
##      EA      IS
## IS  8.2e-14 -
## NMIS 8.7e-14 1
##
## P value adjustment method: bonferroni
```

### 13.2.1.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TRUNCATION') %>%
  ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fill = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final performance') +
  p_theme + coord_flip()
```



#### 13.2.1.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Select.
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_fit_max)),
    min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0    6   52.5  50.6  100.   45.0
## 2 IS          100     0  25.0  82.5  79.7  99.9   20.2
## 3 NMIS        100     0  23.0  82.0  79.7  99.9   20.5
```

Kruskal–Wallis test provides evidence of difference among selection schemes.



```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 75.468, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

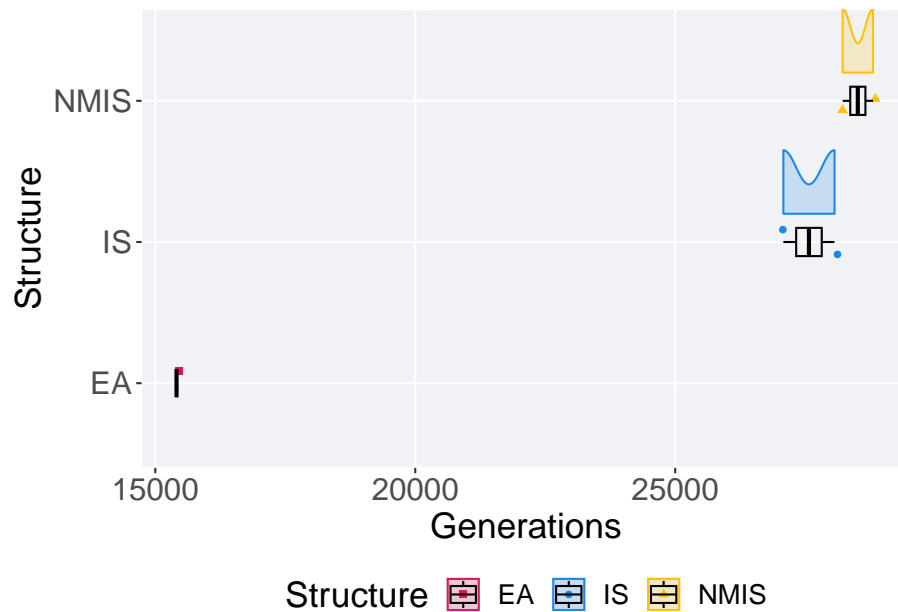
```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.method = "b"
                    paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$pop_fit_max and performance$Structure
##
##      EA      IS
## IS  8.2e-14 -
## NMIS 8.7e-14 1
##
## P value adjustment method: bonferroni
```

### 13.2.2 Generation satisfactory solution found

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi5000_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TRUNCATION' & C
ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Stru
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE)+
  scale_y_continuous(
    name="Generations"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  p_theme + coord_flip()
```



### 13.2.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
ssf = filter(mi5000_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int>  <int> <int>  <dbl> <dbl> <int> <dbl>
## 1 EA          1      0 15409 15409 15409 15409     0
## 2 IS          2      0 27080 27572 27572 28064    492
## 3 NMIS        2      0 28220 28512 28512 28804    292
```

Kruskal–Wallis test provides evidence of no difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)

##
## Kruskal-Wallis rank sum test
##
## data: Generations by Structure
## Kruskal-Wallis chi-squared = 3.6, df = 2, p-value = 0.1653
```

### 13.2.3 Activation gene coverage

Activation gene coverage analysis.

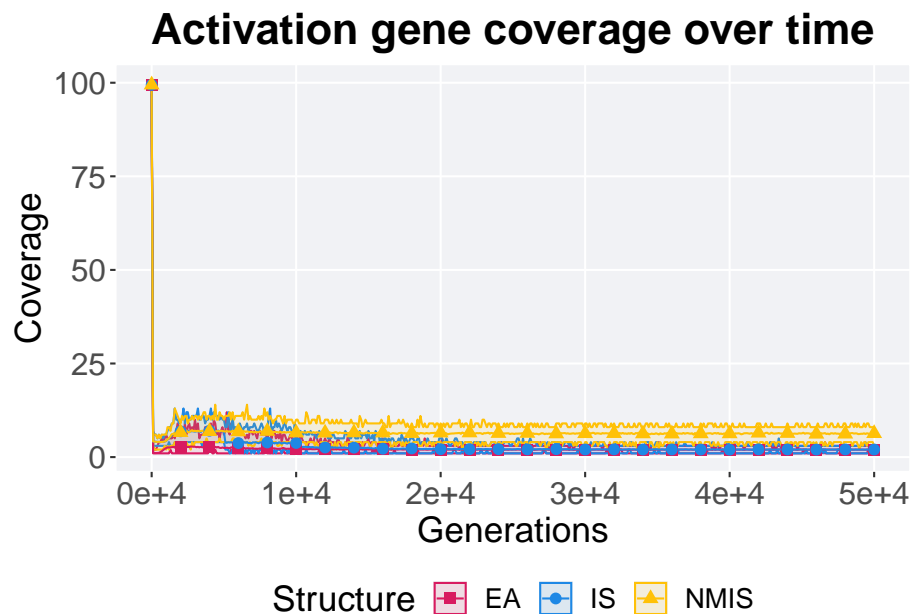
#### 13.2.3.1 Coverage over time

Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == '
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )

## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.

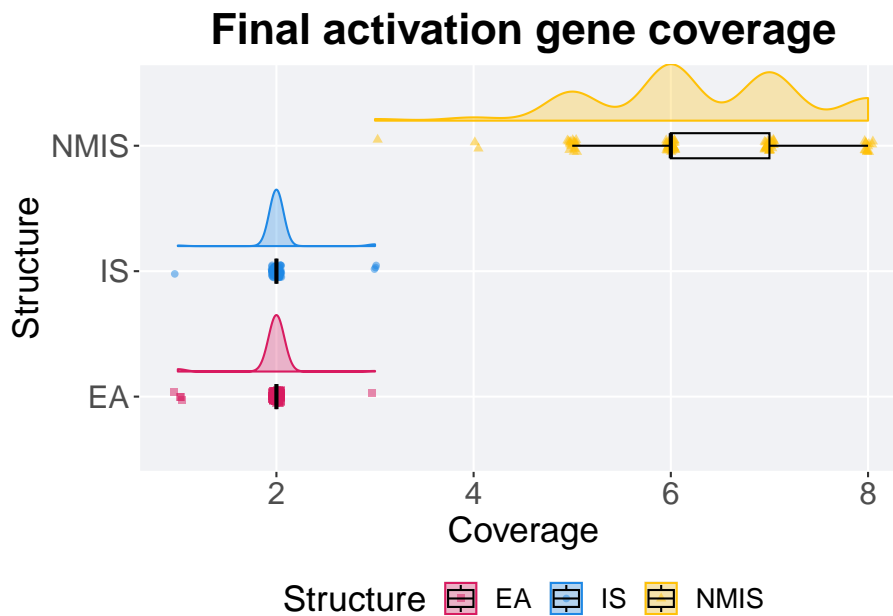
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  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
```



#### 13.2.3.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```
### end of run
filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme + coord_flip()
```



#### 13.2.3.2.1 Stats

Summary statistics for activation gene coverage.

```
coverage = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` =
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_act_cov)),
    min = min(pop_act_cov, na.rm = TRUE),
    median = median(pop_act_cov, na.rm = TRUE),
    mean = mean(pop_act_cov, na.rm = TRUE),
    max = max(pop_act_cov, na.rm = TRUE),
    IQR = IQR(pop_act_cov, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt  min median  mean  max  IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA         100     0     1     2  1.97     3     0
## 2 IS         100     0     1     2  2.02     3     0
## 3 NMIS       100     0     3     6  6.33     8     1
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 269.84, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```
pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method =
                    paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS  0.15    -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

## 13.3 Tournament selection

Here we analyze how the different population structures affect tournament selection (size 8) on the contradictory objectives diagnostic.

### 13.3.1 Performance

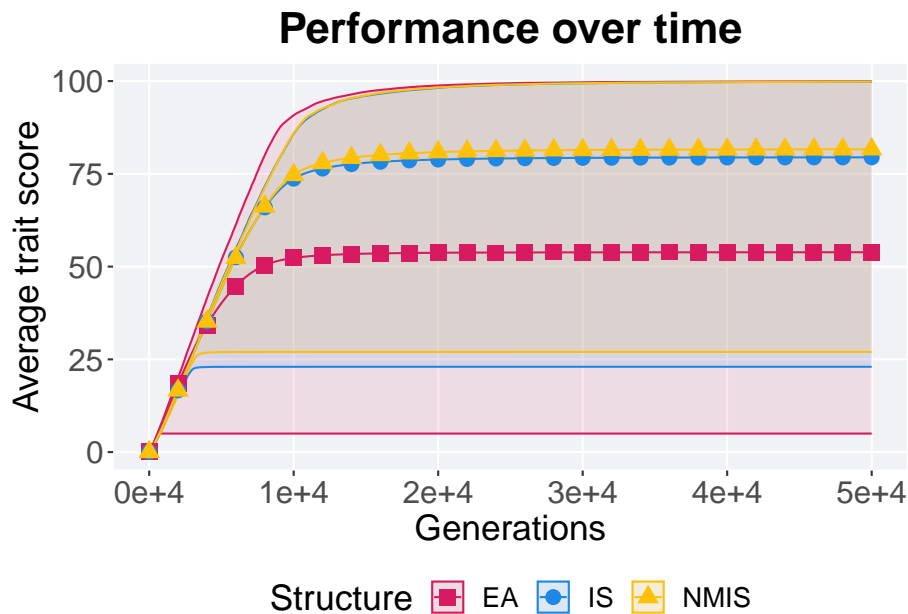
#### 13.3.1.1 Performance over time

```
lines = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection` == 'Tournament')
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %>= 2000), size = 2.5, stroke = 2.0,
            scale_y_continuous(
```

```

    name="Average trait score"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme

```



#### 13.3.1.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```

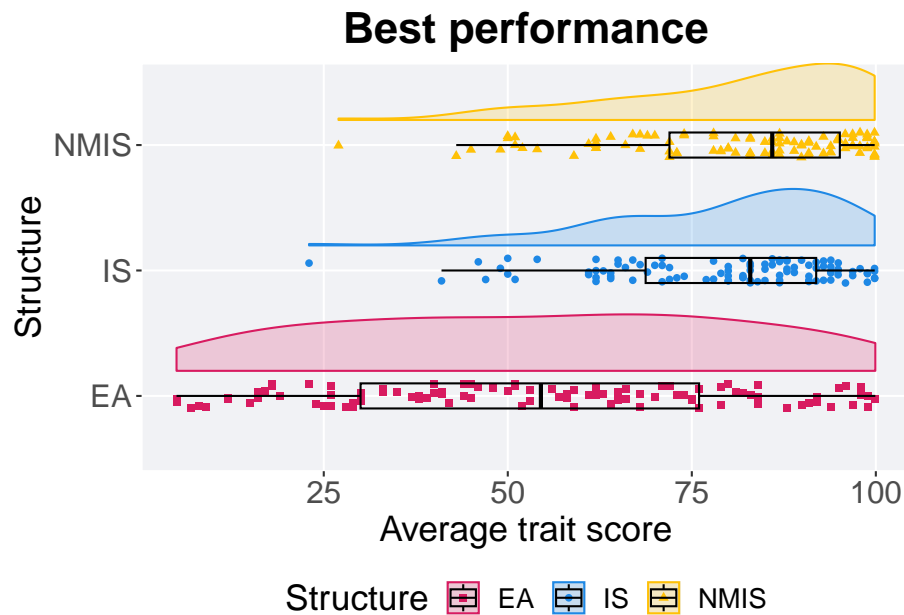
filter(mi5000_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT' &
  ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +

```

```

scale_y_continuous(
  name="Average trait score"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Best performance')+
p_theme + coord_flip()

```



#### 13.3.1.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

performance = filter(mi5000_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection` == 'n')
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VALUE)),
    min = min(VALUE, na.rm = TRUE) / DIMENSIONALITY,
    median = median(VALUE, na.rm = TRUE) / DIMENSIONALITY,
    mean = mean(VALUE, na.rm = TRUE) / DIMENSIONALITY,

```



```

    max = max(VAL, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VAL, na.rm = TRUE) / DIMENSIONALITY
  )

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100     0     5    54.5  53.9  99.9  46.0
## 2 IS          100     0   23.0  82.9  79.5  99.8  23.2
## 3 NMIS        100     0   27.0  85.9  81.6  99.8  23.1

```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VAL ~ Structure, data = performance)
```

```

##
##  Kruskal-Wallis rank sum test
##
## data:  VAL by Structure
## Kruskal-Wallis chi-squared = 63.856, df = 2, p-value = 1.361e-14

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

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

```

##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VAL and performance$Structure
##
##      EA      IS
## IS  6.2e-11 -
## NMIS 1.5e-12 0.37
##
## P value adjustment method: bonferroni

```

### 13.3.1.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```

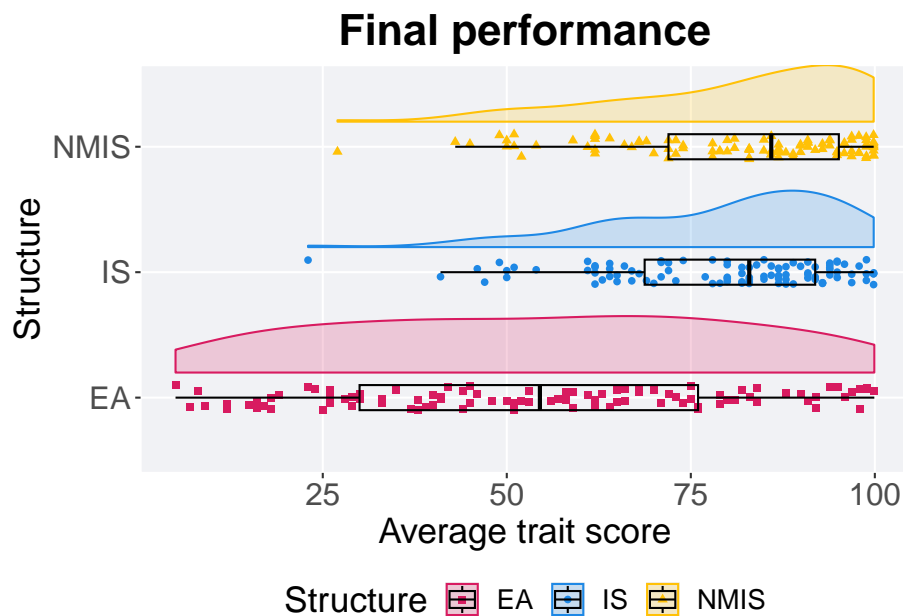
filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT')
ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fill = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  )

```

```

) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Final performance')+
p_theme + coord_flip()

```



#### 13.3.1.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

performance = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Select.
performance %>%
group_by(Structure) %>%
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(pop_fit_max)),
  min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
  median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
  mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
  max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
  IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
)

```

```
)

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int>  <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA          100      0     5    54.5  53.9  99.9  46.0
## 2 IS          100      0    23.0  82.9  79.5  99.8  23.2
## 3 NMIS        100      0    27.0  85.9  81.6  99.8  23.1
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 63.856, df = 2, p-value = 1.361e-14
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.method = "b",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$pop_fit_max and performance$Structure
##
##      EA      IS
## IS  6.2e-11 -
## NMIS 1.5e-12 0.37
##
## P value adjustment method: bonferroni
```

### 13.3.2 Generation satisfactory solution found

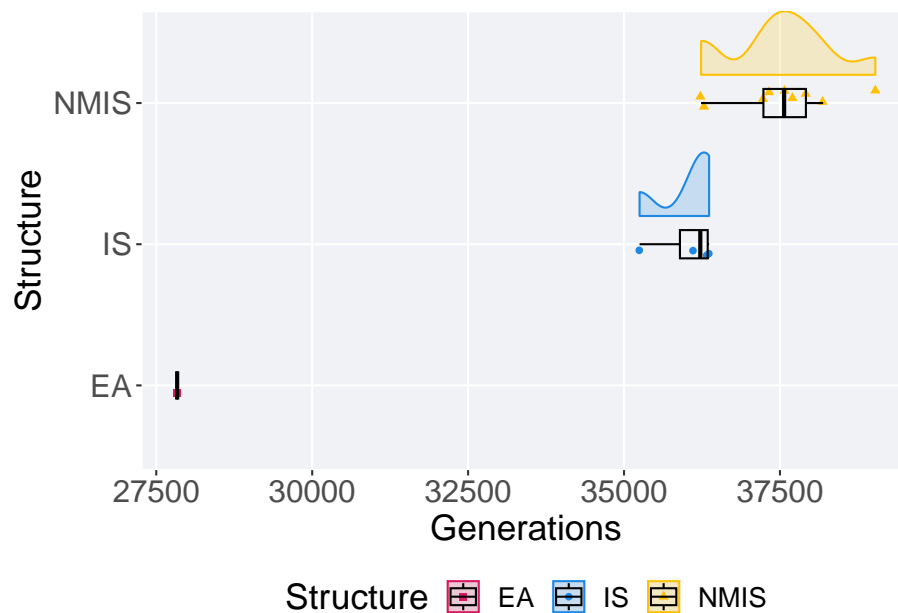
First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi5000_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT') &
  ggplot(., aes(x = Structure, y = Generations, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.2) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Generations"
```

```

) +
scale_x_discrete(
  name="Structure"
) +
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
p_theme + coord_flip()

```



### 13.3.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```

ssf = filter(mi5000_ssf, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
ssf %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(Generations)),
    min = min(Generations, na.rm = TRUE),
    median = median(Generations, na.rm = TRUE),
    mean = mean(Generations, na.rm = TRUE),
    max = max(Generations, na.rm = TRUE),
    IQR = IQR(Generations, na.rm = TRUE)
  )

```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          1      0 27835 27835 27835 27835  0
## 2 IS          4      0 35249 36223 36015. 36364 445.
## 3 NMIS        9      0 36235 37567 37498. 39029 681
```

Kruskal-Wallis test provides evidence of no difference among selection schemes.

```
kruskal.test(Generations ~ Structure, data = ssf)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  Generations by Structure
## Kruskal-Wallis chi-squared = 6.6444, df = 2, p-value = 0.03607
pairwise.wilcox.test(x = ssf$Generations, g = ssf$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum exact test
##
## data:  ssf$Generations and ssf$Structure
##
##      EA   IS
## IS  0.60 -
## NMIS 0.30 0.05
##
## P value adjustment method: bonferroni
```

### 13.3.3 Activation gene coverage

Activation gene coverage analysis.

#### 13.3.3.1 Coverage over time

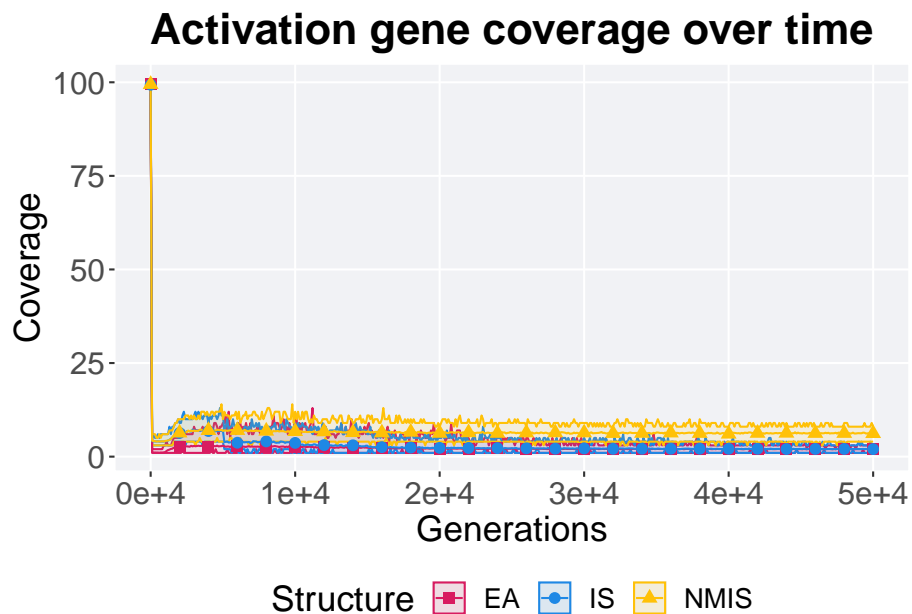
Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == '
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )
```

## `summarise()` has grouped output by 'Structure'. You can override using the

## ``groups`` argument.

```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure)) +
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
    scale_y_continuous(
      name="Coverage"
    ) +
    scale_x_continuous(
      name="Generations",
      limits=c(0, 50000),
      breaks=c(0, 10000, 20000, 30000, 40000, 50000),
      labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
    ) +
    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
```



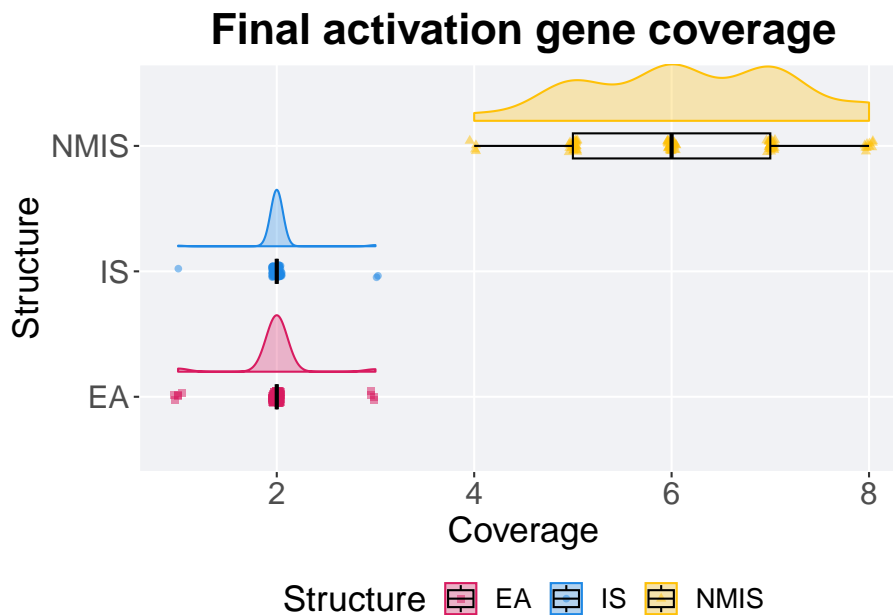
### 13.3.3.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```

### end of run
filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT') %>%
  ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_shape_manual(values=SHAPE) +
  scale_y_continuous(
    name="Coverage"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Final activation gene coverage') +
  p_theme + coord_flip()

```



#### 13.3.3.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'TOURNAMENT') %>%
  coverage %>%
  group_by(Structure) %>%

```

```
dplyr::summarise(
  count = n(),
  na_cnt = sum(is.na(pop_act_cov)),
  min = min(pop_act_cov, na.rm = TRUE),
  median = median(pop_act_cov, na.rm = TRUE),
  mean = mean(pop_act_cov, na.rm = TRUE),
  max = max(pop_act_cov, na.rm = TRUE),
  IQR = IQR(pop_act_cov, na.rm = TRUE)
)
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int> <dbl> <dbl> <int> <dbl>
## 1 EA          100     0     1     2  1.99     3     0
## 2 IS          100     0     1     2  2.01     3     0
## 3 NMIS        100     0     4     6  6.2      8     2
```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 265.43, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```
pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method =
  paired = FALSE, conf.int = FALSE, alternative = 'g')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: coverage$pop_act_cov and coverage$Structure
##
##      EA      IS
## IS  0.85    -
## NMIS <2e-16 <2e-16
##
## P value adjustment method: bonferroni
```

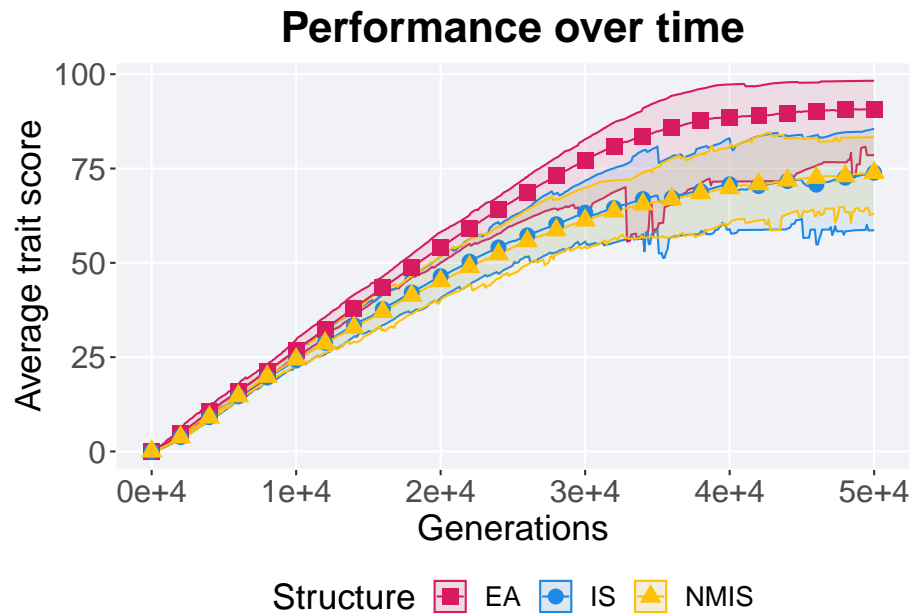


## 13.4 Lexicase selection

Here we analyze how the different population structures affect standard lexicase selection on the contradictory objectives diagnostic.

### 13.4.1 Performance

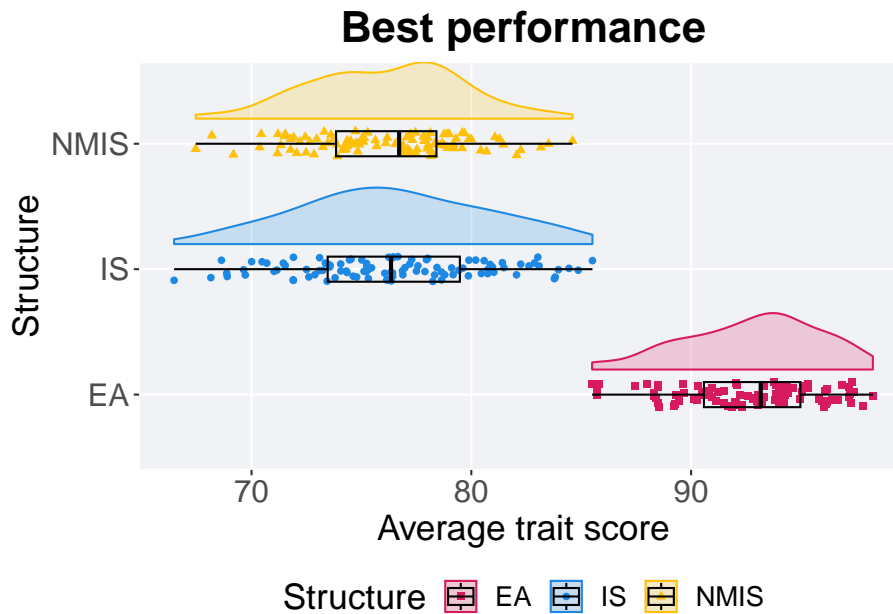
```
lines = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == '
  group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_fit_max) / DIMENSIONALITY,
    mean = mean(pop_fit_max) / DIMENSIONALITY,
    max = max(pop_fit_max) / DIMENSIONALITY
  )
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = Structure,
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 2.5, stroke = 2.0, alpha = 1.
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_continuous(
    name="Generations",
    limits=c(0, 50000),
    breaks=c(0, 10000, 20000, 30000, 40000, 50000),
    labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
  ) +
  scale_shape_manual(values=SHAPE)+
  scale_colour_manual(values = cb_palette) +
  scale_fill_manual(values = cb_palette) +
  ggtitle("Performance over time") +
  p_theme
```



#### 13.4.1.2 Best performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi5000_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'LE')
ggplot(., aes(x = Structure, y = VAL / DIMENSIONALITY, color = Structure, fill = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.5) +
  geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
  geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
  scale_y_continuous(
    name="Average trait score"
  ) +
  scale_x_discrete(
    name="Structure"
  ) +
  scale_shape_manual(values=SHAPE) +
  scale_colour_manual(values = cb_palette, ) +
  scale_fill_manual(values = cb_palette) +
  ggtitle('Best performance') +
  p_theme + coord_flip()
```



#### 13.4.1.2.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi5000_best, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
performance$Structure = factor(performance$Structure, levels=c('EA', 'NMIS', 'IS'))
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(VAL)),
    min = min(VAL, na.rm = TRUE) / DIMENSIONALITY,
    median = median(VAL, na.rm = TRUE) / DIMENSIONALITY,
    mean = mean(VAL, na.rm = TRUE) / DIMENSIONALITY,
    max = max(VAL, na.rm = TRUE) / DIMENSIONALITY,
    IQR = IQR(VAL, na.rm = TRUE) / DIMENSIONALITY
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA         100     0  85.5   93.2  92.8  98.3  4.39
## 2 NMIS       100     0  67.5   76.7  76.2  84.6  4.57
## 3 IS        100     0  66.5   76.3  76.4  85.5  6.01
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(VA ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data:  VA by Structure
## Kruskal-Wallis chi-squared = 199.33, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

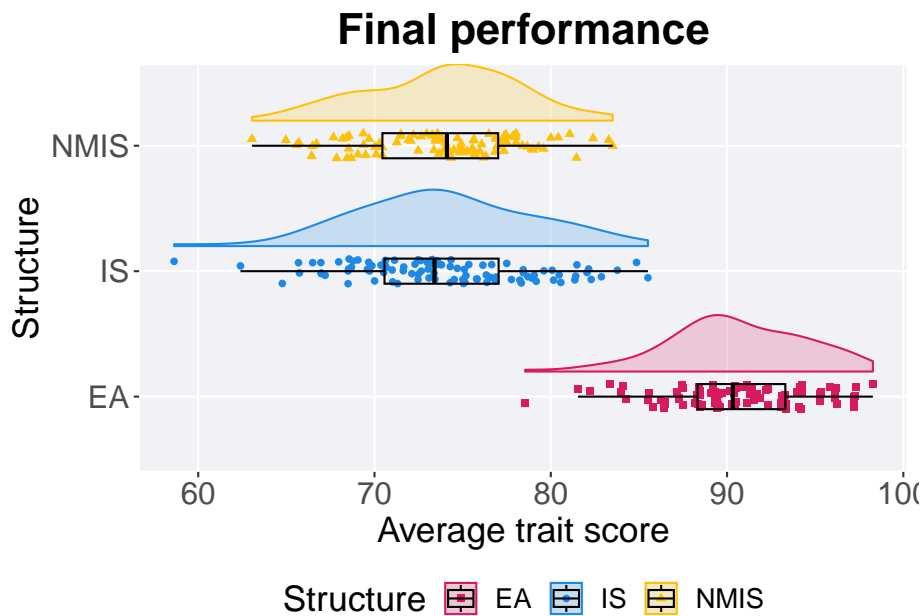
```
pairwise.wilcox.test(x = performance$VA, g = performance$Structure, p.adjust.method =
                    paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  performance$VA and performance$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 1
##
## P value adjustment method: bonferroni
```

### 13.4.1.3 Final performance

First generation a satisfactory solution is found throughout the 50,000 generations.

```
filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` ==
ggplot(., aes(x = Structure, y = pop_fit_max / DIMENSIONALITY, color = Structure, fi
geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0
geom_point(position = position_jitter(width = .1), size = 1.5, alpha = 1.0) +
geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
scale_y_continuous(
  name="Average trait score"
) +
scale_x_discrete(
  name="Structure"
)+
scale_shape_manual(values=SHAPE)+
scale_colour_manual(values = cb_palette, ) +
scale_fill_manual(values = cb_palette) +
ggtitle('Final performance')+
p_theme + coord_flip()
```



#### 13.4.1.3.1 Stats

Summary statistics for the first generation a satisfactory solution is found.

```
performance = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'EA')
performance$Structure = factor(performance$Structure, levels=c('EA', 'NMIS', 'IS'))
performance %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_fit_max)),
    min = min(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    median = median(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    mean = mean(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    max = max(pop_fit_max / DIMENSIONALITY, na.rm = TRUE),
    IQR = IQR(pop_fit_max / DIMENSIONALITY, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 EA         100     0  78.5  90.3  90.5  98.3  4.98
## 2 NMIS       100     0  63.0  74.1  73.9  83.5  6.57
## 3 IS         100     0  58.6  73.4  73.9  85.5  6.47
```

Kruskal–Wallis test provides evidence of difference among selection schemes.

```
kruskal.test(pop_fit_max ~ Structure, data = performance)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pop_fit_max by Structure
## Kruskal-Wallis chi-squared = 196.97, df = 2, p-value < 2.2e-16
```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction.

```
pairwise.wilcox.test(x = performance$pop_fit_max, g = performance$Structure, p.adjust.m = 'p.adjust.method',
paired = FALSE, conf.int = FALSE, alternative = 'l')
```

```
##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data: performance$pop_fit_max and performance$Structure
##
##      EA      NMIS
## NMIS <2e-16 -
## IS   <2e-16 1
##
## P value adjustment method: bonferroni
```

### 13.4.2 Activation gene coverage

Activation gene coverage analysis.

#### 13.4.2.1 Coverage over time

Activation gene coverage over time.

```
# data for lines and shading on plots
lines = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nS
group_by(Structure, Generations) %>%
  dplyr::summarise(
    min = min(pop_act_cov),
    mean = mean(pop_act_cov),
    max = max(pop_act_cov)
  )
```

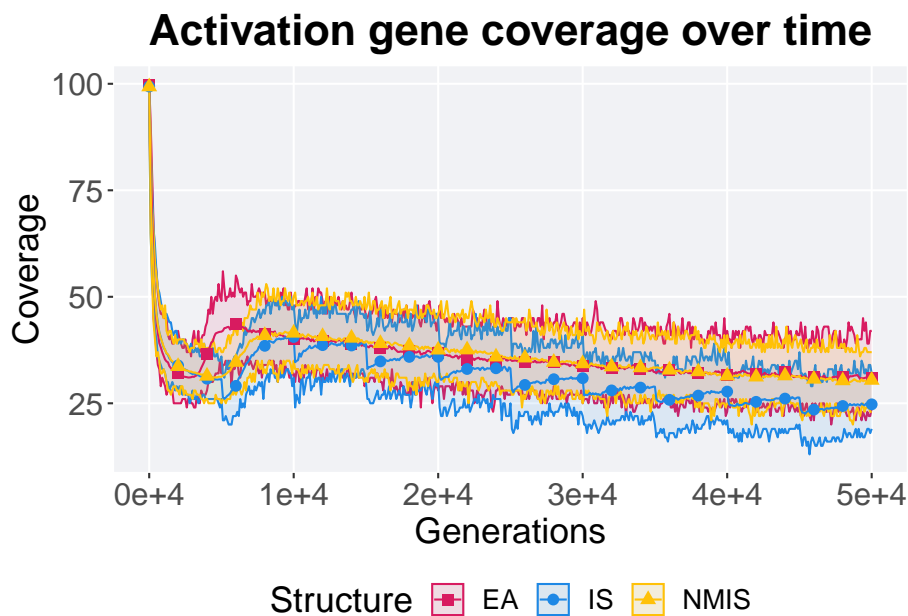
```
## `summarise()` has grouped output by 'Structure'. You can override using the
## `.groups` argument.
```

```
ggplot(lines, aes(x=Generations, y=mean, group = Structure, fill = Structure, color = S
  geom_ribbon(aes(ymin = min, ymax = max), alpha = 0.1) +
  geom_line(size = 0.5) +
  geom_point(data = filter(lines, Generations %% 2000 == 0), size = 1.5, stroke = 2.0,
```

```

scale_y_continuous(
  name="Coverage"
) +
scale_x_continuous(
  name="Generations",
  limits=c(0, 50000),
  breaks=c(0, 10000, 20000, 30000, 40000, 50000),
  labels=c("0e+4", "1e+4", "2e+4", "3e+4", "4e+4", "5e+4")
) +
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

```



#### 13.4.2.2 End of 50,000 generations

Activation gene coverage in the population at the end of 50,000 generations.

```

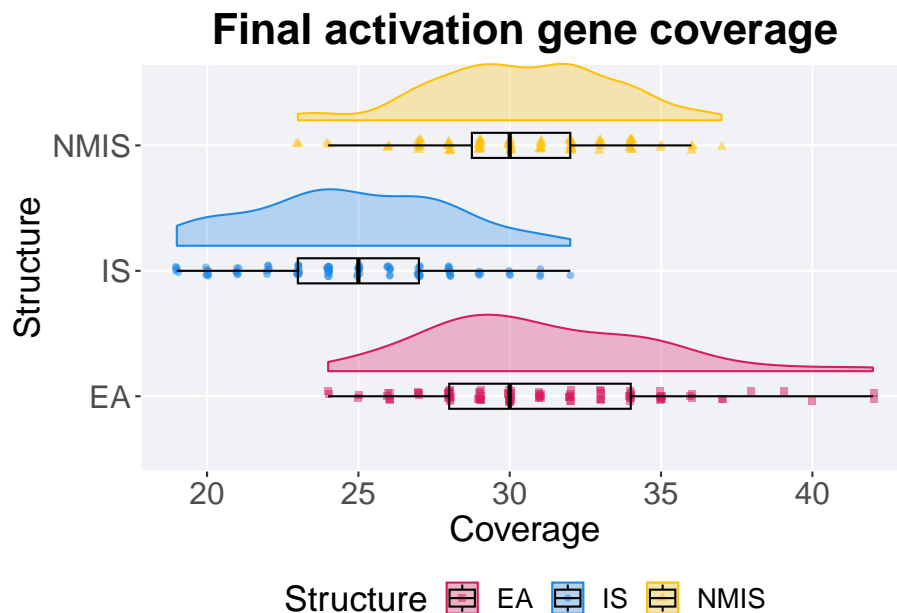
### end of run
filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection\nScheme` == 'LEXICASE')
ggplot(., aes(x = Structure, y = pop_act_cov, color = Structure, fill = Structure, shape = Structure)) +
  geom_flat_violin(position = position_nudge(x = .2, y = 0), scale = 'width', alpha = 0.3) +
  geom_point(position = position_jitter(height = .05, width = .05), size = 1.5, alpha = 0.5) +

```

```

geom_boxplot(color = 'black', width = .2, outlier.shape = NA, alpha = 0.0) +
scale_shape_manual(values=SHAPE) +
scale_y_continuous(
  name="Coverage"
) +
scale_x_discrete(
  name="Structure"
) +
scale_colour_manual(values = cb_palette) +
scale_fill_manual(values = cb_palette) +
ggtitle('Final activation gene coverage') +
p_theme + coord_flip()

```



#### 13.4.2.2.1 Stats

Summary statistics for activation gene coverage.

```

coverage = filter(mi5000_over_time, Diagnostic == 'MULTIPATH_EXPLORATION' & `Selection` == 'Final')
coverage$Structure = factor(coverage$Structure, levels=c('EA','NMIS','IS'))
coverage %>%
  group_by(Structure) %>%
  dplyr::summarise(
    count = n(),
    na_cnt = sum(is.na(pop_act_cov)),
    min = min(pop_act_cov, na.rm = TRUE),

```



```

median = median(pop_act_cov, na.rm = TRUE),
mean = mean(pop_act_cov, na.rm = TRUE),
max = max(pop_act_cov, na.rm = TRUE),
IQR = IQR(pop_act_cov, na.rm = TRUE)
)

```

```

## # A tibble: 3 x 8
##   Structure count na_cnt   min median   mean   max   IQR
##   <fct>      <int> <int> <int>   <dbl> <dbl> <int> <dbl>
## 1 EA          100     0    24     30    31    42    6
## 2 NMIS        100     0    23     30   30.4    37   3.25
## 3 IS          100     0    19     25   24.8    32    4

```

Kruskal–Wallis test provides evidence of difference among activation gene coverage.

```
kruskal.test(pop_act_cov ~ Structure, data = coverage)
```

```

##
## Kruskal-Wallis rank sum test
##
## data:  pop_act_cov by Structure
## Kruskal-Wallis chi-squared = 130.57, df = 2, p-value < 2.2e-16

```

Results for post-hoc Wilcoxon rank-sum test with a Bonferroni correction on activation gene coverage.

```

pairwise.wilcox.test(x = coverage$pop_act_cov, g = coverage$Structure, p.adjust.method = "bonferroni",
                     paired = FALSE, conf.int = FALSE, alternative = 'l')

```

```

##
## Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  coverage$pop_act_cov and coverage$Structure
##
##      EA      NMIS
## NMIS 0.81    -
## IS   <2e-16 <2e-16
##
## P value adjustment method: bonferroni

```